Departamento de Física, Facultad de Ciencias, Universidad de Chile.
Las Palmeras 3425, Ñuñoa. Casilla 653, Correo 1, Santiago
FONO: 562 678 7276 FAX: 562 271 2973
E-MAIL: secretaria@fisica.ciencias.uchile.cl

Apuntes de un curso de

INTRODUCCIÓN A LA FÍSICA MATEMÁTICA

segunda edición

José Rogan C.
Víctor Muñoz G.

Índice

Índice de figuras

Parte I

Análisis Vectorial.

Capítulo 1

Análisis vectorial.

versión final 2.05-260903[1]

1.1. Definiciones, una aproximación elemental.

En ciencia e ingeniería encontramos frecuentemente cantidades que tienen magnitud y sólo magnitud: la masa, el tiempo o la temperatura. Estas cantidades las llamamos *escalares*. En contraste, muchas cantidades físicas interesantes tienen magnitud y, adicionalmente, una dirección asociada. Este segundo grupo incluye el desplazamiento, la velocidad, la aceleración, la fuerza, el momento, el momento angular. Cantidades con magnitud y dirección serán llamadas cantidades *vectoriales*. Usualmente, en tratamientos elementales, un vector es definido como una cantidad que tiene magnitud y dirección. Para distinguir vectores de escalares, identificaremos las cantidades vectoriales con una flecha arriba, $\vec{V}$, o bien, con letra en "negrita", esto es, $\mathbf{V}$.

Un vector puede ser convenientemente representado por una flecha con largo proporcional a la magnitud. La dirección de la flecha da la dirección del vector, el sentido positivo de la dirección es indicado por la punta. En esta representación la suma vectorial

$$\vec{C} = \vec{A} + \vec{B} \ , \tag{1.1}$$

consiste en poner la parte posterior del vector $\vec{B}$ en la punta del vector $\vec{A}$. El vector $\vec{C}$ es

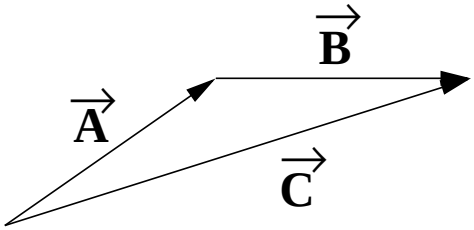


Figura 1.1: Ley del triángulo para la suma de vectores.

el representado por una flecha dibujada desde la parte posterior de $\vec{A}$ hasta la punta de $\vec{B}$.

[1]Este capítulo está basado en el primer capítulo del libro: *Mathematical Methods for Physicists, fourth edition* de George B. Arfken & Hans J. Weber, editorial Academic Press.

Este procedimiento, la ley del triángulo para la suma, le asigna un significado a la ecuación (1.1) y es ilustrado en la figura 1.1. Al completar el paralelogramo, vemos que

$$\vec{C} = \vec{A} + \vec{B} = \vec{B} + \vec{A} \, , \tag{1.2}$$

como muestra la figura 1.2. En otras palabras la suma vectorial es *conmutativa*

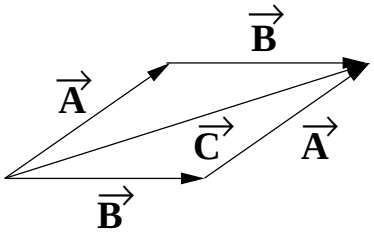


Figura 1.2: Ley del paralelogramo para suma de vectores.

Para la suma de tres vectores

$$\vec{D} = \vec{A} + \vec{B} + \vec{C} \, ,$$

figura 1.3, podemos primero sumar $\vec{A}$ y $\vec{B}$

$$\vec{A} + \vec{B} = \vec{E} \, .$$

Entonces a esta suma le agregamos $\vec{C}$

$$\vec{D} = \vec{E} + \vec{C} \, .$$

Similarmente, podemos primero sumar $\vec{B}$ y $\vec{C}$

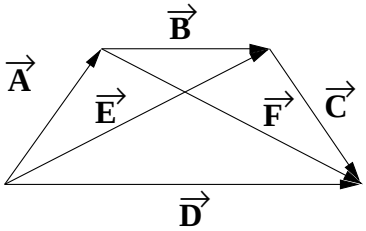


Figura 1.3: La suma de vectores es asociativa.

$$\vec{B} + \vec{C} = \vec{F} \, .$$

Entonces

$$\vec{D} = \vec{A} + \vec{F} \, .$$

En términos de la expresión original,

$$(\vec{A} + \vec{B}) + \vec{C} = \vec{A} + (\vec{B} + \vec{C}) \, .$$

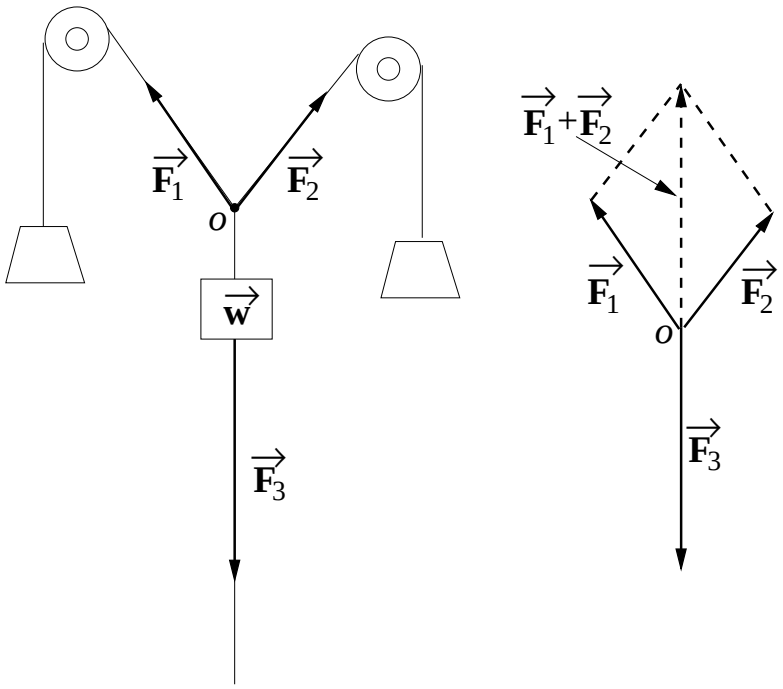


Figura 1.4: Equilibrio de fuerzas. $\vec{F}_1 + \vec{F}_2 = -\vec{F}_3$.

La suma de vectores es *asociativa*.

Un ejemplo físico directo de la ley de suma del paralelogramo es provisto por un peso suspendido de dos cuerdas. Si el punto de juntura (o en la figura 1.4) está en equilibrio, el vector suma de las dos fuerzas, $\vec{F}_1$ y $\vec{F}_2$, debe justo cancelarse con la fuerza hacia abajo $\vec{F}_3$. Aquí la ley de suma del paralelogramo es sujeta a una verificación experimental inmediata.[2]

La resta puede ser manejada definiendo el negativo de un vector como un vector de la misma magnitud pero con la dirección contraria. Entonces

$$\vec{A} - \vec{B} = \vec{A} + (-\vec{B}) \ .$$

En la figura 1.3

$$\vec{A} = \vec{E} - \vec{B}$$

Notemos que los vectores son tratados como objetos geométricos que son independientes de cualquier sistema de coordenadas. Realmente, no hemos presentado aún un sistema de coordenadas. Este concepto de independencia de un particular sistema de coordenadas es desarrollado en detalle en la próxima sección.

La representación del vector $\vec{A}$ por una flecha sugiere una segunda posibilidad. La flecha $\vec{A}$ (figura 1.5), partiendo desde el origen,[3] y terminando en el punto (A_x, A_y, A_z). Así, acordamos

[2]Estrictamente hablando la ley de suma del paralelogramo fue introducida como definición. Los experimentos muestran que si suponemos que las fuerzas son cantidades vectoriales y las combinamos usando la ley de suma del paralelogramo, la condición de una fuerza resultante nula como condición de equilibrio es satisfecha.

[3]Se verá que se puede partir desde cualquier punto en el sistema de referencias cartesiano, hemos elegido

que los vectores partan en el origen y su final puede ser especificado dando las coordenadas cartesianas (A_x, A_y, A_z) de la punta de la flecha.

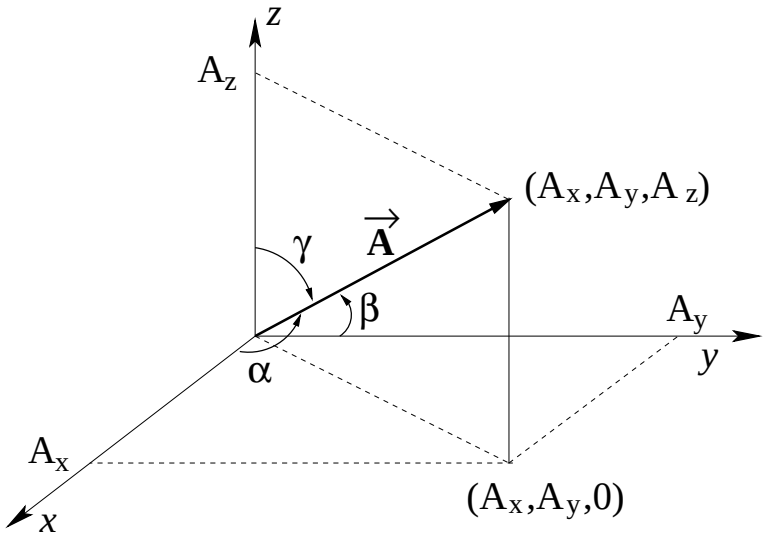


Figura 1.5: Componentes cartesianas y cosenos directores de $\vec{A}$.

Aunque $\vec{A}$ podría haber representado cualquier cantidad vectorial (momento, campo eléctrico, etc.) existe una cantidad vectorial importante, el desplazamiento desde el origen al punto (x, y, z), que es denotado por el símbolo especial $\vec{r}$. Entonces tenemos la elección de referirnos al desplazamiento como el vector $\vec{r}$ o por la colección (x, y, z), que son las coordenadas del punto final:

$$\vec{r} \leftrightarrow (x, y, z) \ . \tag{1.3}$$

Usando r para la magnitud del vector $\vec{r}$ encontramos que la figura 1.5 muestra que las coordenadas del punto final y la magnitud están relacionadas por

$$x = r\cos\alpha \ , \quad y = r\cos\beta \ , \quad z = r\cos\gamma \ . \tag{1.4}$$

Los $\cos\alpha$, $\cos\beta$ y $\cos\gamma$ son llamados los *cosenos directores*, donde α es el ángulo entre el vector dado y el eje-x positivo, y así sucesivamente. Un poco de vocabulario: las cantidades A_x, A_y y A_z son conocidas como las *componentes* (cartesianas) de $\vec{A}$ o las *proyecciones* de $\vec{A}$.

Así, cualquier vector $\vec{A}$ puede ser resuelto en sus componentes (o proyecciones sobre los ejes coordenados) produciendo $A_x = A\cos\alpha$, etc., como en la ecuación (1.4). Podemos elegir referirnos al vector como un cantidad simple $\vec{A}$ o por sus componentes (A_x, A_y, A_z). Notemos que el subíndice x en A_x denota la componente x y no la dependencia sobre la variable x. La componente A_x puede ser función de x, y y z tal que $A_x(x, y, z)$. La elección entre usar $\vec{A}$ o sus componentes (A_x, A_y, A_z) es esencialmente una elección entre una representación geométrica y una algebraica. En el lenguaje de teoría de grupos (capítulo 4), las dos representaciones son *isomórficas*.

el origen por simplicidad. Esta libertad de desplazar el origen del sistema de coordenadas sin afectar la geometría es llamada **invariancia translacional**.

Usaremos la representación que nos convenga más en cada caso. La representación geométrica "flecha en el espacio" puede ayudar en la visualización. El conjunto algebraico de componentes es usualmente mucho más conveniente para calcular.

Los vectores entran en la Física de dos maneras distintas : (1) Un vector $\vec{A}$ puede representar una sola fuerza actuando sobre un único punto. La fuerza de gravedad actuando sobre el centro de gravedad ilustra esta forma. (2) Un vector $\vec{A}$ puede estar definido sobre una región extendida; esto es, $\vec{A}$ y sus componentes son funciones de la posición: $A_x = A_x(x, y, z)$, y así sucesivamente. Ejemplos de este tipo incluyen el campo de velocidades de un fluido. Algunos autores distinguen estos dos casos refiriéndose a los vectores definidos sobre una región como un campo vectorial. Los conceptos de campos vectoriales como funciones de la posición serán extremadamente importantes más adelante.

En este punto es conveniente presentar los vectores unitarios a lo largo de cada eje coordenado. Sea $\hat{x}$ un vector de magnitud uno apuntando en la dirección x-positiva, $\hat{y}$, un vector de magnitud uno apuntando en la dirección y-positiva, y $\hat{z}$, un vector de magnitud unitaria en la dirección z-positiva. Entonces $\hat{x}A_x$, es un vector con magnitud igual a A_x, en la dirección de x-positiva. Por suma de vectores

$$\vec{A} = \hat{x}A_x + \hat{y}A_y + \hat{z}A_z \ , \tag{1.5}$$

lo cual establece que un vector es igual a la suma vectorial de sus componentes. Notemos que si $\vec{A}$ se anula, todas sus componentes deben anularse individualmente; esto es, si

$$\vec{A} = 0 \ , \qquad \text{entonces } A_x = A_y = A_z = 0 \ .$$

Finalmente, por el Teorema de Pitágoras, la magnitud del vector $\vec{A}$ es

$$A = (A_x^2 + A_y^2 + A_z^2)^{1/2} \ . \tag{1.6}$$

Esta resolución de un vector en sus componentes puede ser llevada a cabo en una variedad de sistemas de coordenadas, como mostramos en el próximo capítulo. Aquí nos restringiremos a coordenadas cartesianas.

La ecuación (1.5) es realmente la afirmación de que los tres vectores unitarios $\hat{x}$, $\hat{y}$ y $\hat{z}$ "abarcan" nuestro espacio real tridimensional. Cualquier vector constante puede ser escrito como una combinación lineal de $\hat{x}$, $\hat{y}$ y $\hat{z}$. Ya que $\hat{x}$, $\hat{y}$ y $\hat{z}$ son linealmente independientes (ninguno es una combinación lineal de los otros dos), ellos forman una *base* para el espacio real tridimensional.

Como un reemplazo a la técnica gráfica, la suma y resta de vectores puede ser llevada a cabo en términos de sus componentes. Para $\vec{A} = \hat{x}A_x + \hat{y}A_y + \hat{z}A_z$ y $\vec{B} = \hat{x}B_x + \hat{y}B_y + \hat{z}B_z$,

$$\vec{A} \pm \vec{B} = \hat{x}(A_x \pm B_x) + \hat{y}(A_y \pm B_y) + \hat{z}(A_z \pm B_z) \ . \tag{1.7}$$

Deberíamos enfatizar aquí que los vectores unitarios $\hat{x}$, $\hat{y}$, y $\hat{z}$ son usados por conveniencia. Ellos no son esenciales; podemos describir vectores y usarlos enteramente en términos de sus componentes: $\vec{A} \leftrightarrow (A_x, A_y, A_z)$. Esta manera de acercarse es la más poderosa, definiciones más sofisticadas de vectores serán discutidas en las próximas secciones.

Hasta aquí hemos definido las operaciones de suma y resta de vectores. Tres variedades de multiplicaciones son definidas sobre las bases de sus aplicaciones: un producto interno o escalar, un producto propio del espacio tridimensional, y un producto externo o directo que produce un tensor de rango dos. La división por un vector no está definida.

1.2. Rotación de los ejes de coordenadas.

En la sección anterior los vectores fueron definidos o representados en dos maneras equivalentes: (1) geométricamente, especificando la magnitud y la dirección con una flecha y (2) algebraicamente, especificando las componentes relativas a ejes de coordenadas cartesianos. Esta segunda definición es adecuada para el análisis vectorial del resto del capítulo. En esta sección propondremos dos definiciones ambas más sofisticadas y poderosas. Primero, el campo vectorial está definido en términos del comportamiento de las componentes de los vectores bajo la rotación de los ejes de coordenadas. Este acercamiento teórico de transformación nos llevará al análisis tensorial en el capítulo 2 y a la teoría de grupo de las transformaciones en el capítulo 4. Segundo, la definición de componentes de la sección anterior será refinada y generalizada de acuerdo a los conceptos matemáticos de espacio vectorial. Este acercamiento nos llevará a los espacios vectoriales de funciones incluyendo el espacio de Hilbert.

La definición de vector como una cantidad con magnitud y dirección se quiebra en trabajos avanzados. Por otra parte, encontramos cantidades, tales como constantes elásticas e índices de refracción en cristales anisotrópicos, que tienen magnitud y dirección *pero no son vectores*. Por otro lado, nuestra aproximación ingenua es inconveniente para generalizar y extenderla a cantidades más complejas. Obtendremos una nueva definición de campo vectorial, usando nuestro vector desplazamiento $\vec{r}$ como un prototipo.

Hay una importante base física para el desarrollo de una nueva definición. Describimos nuestro mundo físico por matemáticas, pero él y cualquier predicción Física que podamos hacer debe ser independiente de nuestro análisis matemático. Algunos comparan el sistema físico a un edificio y el análisis matemático al andamiaje usado para construir el edificio. Al final el andamiaje es sacado y el edificio se levanta.

En nuestro caso específico suponemos que el espacio es isotrópico; esto es, no hay direcciones privilegiadas o, dicho de otra manera, todas las direcciones son equivalentes. Luego, cualquier sistema físico que está siendo analizado o más bien las leyes físicas que son enunciadas no pueden ni deben depender de nuestra elección u *orientación* de los ejes coordenados.

Ahora, volveremos al concepto del vector $\vec{r}$ como un objeto geométrico independiente del sistema de coordenadas. Veamos un $\vec{r}$ en dos sistemas diferentes, uno rotado respecto al otro.

Por simplicidad consideremos primero el caso en dos dimensiones. Si las coordenadas x e y son rotadas en el sentido contrario a los punteros del reloj en un ángulo φ, *manteniendo* $\vec{r}$ *fijo* (figura 1.6) obtenemos las siguientes relaciones entre las componentes resueltas en el sistema original (sin primas) y aquellas resueltas en el nuevo sistema rotado (con primas):

$$\begin{aligned} x' &= \quad x\cos\varphi + y\,\mathrm{sen}\,\varphi \ , \\ y' &= -x\,\mathrm{sen}\,\varphi + y\cos\varphi \ . \end{aligned} \tag{1.8}$$

Vimos, en la sección anterior, que un vector podría ser representado por las coordenadas de un punto; esto es, las coordenadas eran proporcionales a los componentes del vector. En consecuencia las componentes de un vector deberían transformar bajo rotaciones como las coordenadas de un punto (tal como $\vec{r}$). Por lo tanto cualquier par de cantidades $A_x(x,y)$ y $A_y(x,y)$ en el sistema de coordenadas xy son transformadas en (A'_X, A'_y) por esta rotación del sistema de coordenadas con

$$\begin{aligned} A'_x &= \quad A_x\cos\varphi + A_y\,\mathrm{sen}\,\varphi \ , \\ A'_y &= -A_x\,\mathrm{sen}\,\varphi + A_y\cos\varphi \ , \end{aligned} \tag{1.9}$$

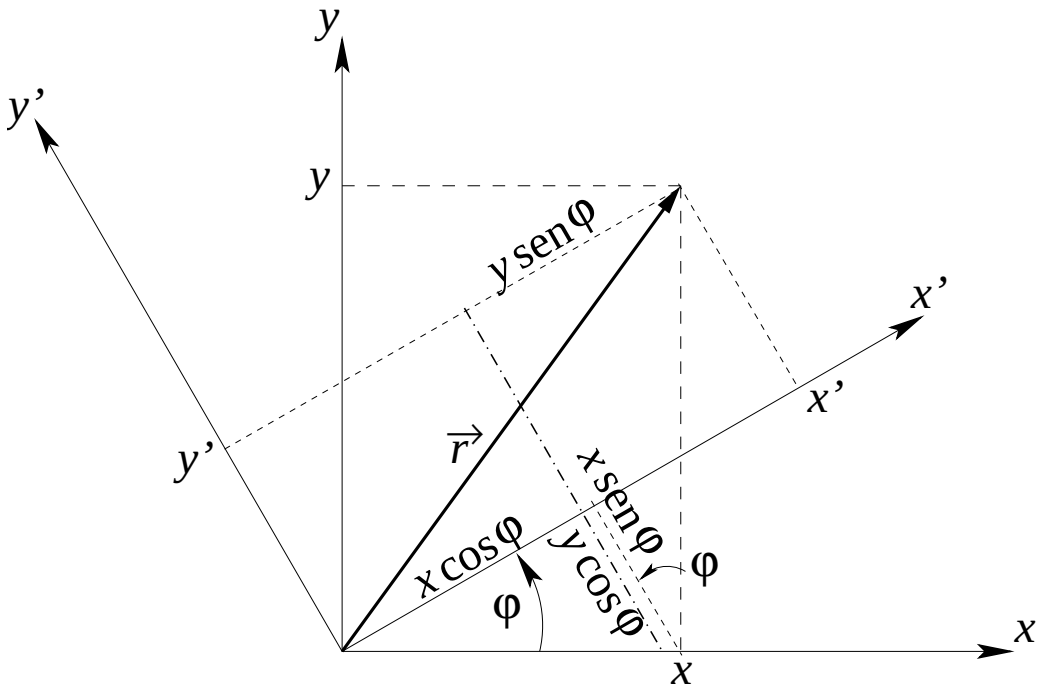


Figura 1.6: Rotación de los ejes coordenados cartesianos respecto del eje z.

definimos[4] A_x y A_y como las componentes de un vector $\vec{A}$. Nuestro vector está definido ahora en términos de la transformación de sus componentes bajo rotación del sistema de coordenadas. Si A_x y A_y transforman de la misma manera como x e y, las componentes del vector de desplazamiento bidimensional, ellas son entonces las componentes del vector $\vec{A}$. Si A_x y A_y no muestran esta forma invariante cuando las coordenadas son rotadas, ellas no forman un vector.

Las componentes del campo vectorial A_x y A_y que satisfacen las ecuaciones de definición (1.9), están asociadas a una magnitud A y a una dirección en el espacio. La magnitud es una cantidad escalar, invariante a la rotación del sistema de coordenadas. La dirección (relativa al sistema sin primas) es asimismo invariante ante la rotación del sistema de coordenadas. El resultado de todo esto es que las componentes de un vector pueden variar de acuerdo a la rotación del sistema de coordenadas con primas. Esto es lo que dicen las ecuaciones (1.9). Pero la variación con el ángulo es tal que las componentes en el sistema de coordenadas rotado A'_x y A'_y definen un vector con la misma magnitud y la misma dirección que el vector definido por las componentes A_x y A_y relativas al eje de coordenadas xy. Las componentes de $\vec{A}$ en un sistema de coordenadas particular constituye la *representación* de $\vec{A}$ en el sistema de coordenadas. Las ecuaciones (1.9), que corresponden a las relaciones de transformación, son una garantía de que la identidad $\vec{A}$ es preservada de la rotación del sistema de coordenadas. Para ir a tres y, luego, a cuatro dimensiones, encontramos conveniente usar una notación más

[4]La definición correspondiente de una cantidad escalar es $S' = S$, esto es, invariante ante rotaciones de los ejes de coordenadas.

compacta. Sea

$$\begin{aligned} x &\to x_1 \\ y &\to x_2 \end{aligned} \tag{1.10}$$

$$\begin{aligned} a_{11} &= \cos\varphi\ , & a_{12} &= \operatorname{sen}\varphi\ , \\ a_{21} &= -\operatorname{sen}\varphi\ , & a_{22} &= \cos\varphi\ . \end{aligned} \tag{1.11}$$

Las ecuaciones (1.8) transforman como

$$\begin{aligned} x_1' &= a_{11}x_1 + a_{12}x_2\ , \\ x_2' &= a_{21}x_1 + a_{22}x_2\ . \end{aligned} \tag{1.12}$$

Los coeficientes a_{ij} pueden ser interpretados como los cosenos directores, es decir, el coseno del ángulo entre x_i' y x_j; esto es,

$$\begin{aligned} a_{12} &= \cos(x_1', x_2) = \operatorname{sen}\varphi, \\ a_{21} &= \cos(x_2', x_1) = \cos\left(\varphi + \frac{\pi}{2}\right) = -\operatorname{sen}\varphi\ . \end{aligned} \tag{1.13}$$

La ventaja de la nueva notación[5]es que nos permite usar el símbolo suma $\sum$ y reescribir las ecuaciones (1.12) como

$$x_i' = \sum_{j=1}^{2} a_{ij}x_j\ , \qquad i = 1, 2\ . \tag{1.14}$$

Note que i permanece como un parámetro que da origen a una ecuación cuando este conjunto es igual a 1 y a una segunda ecuación cuando este conjunto es igual a 2. El índice j, por supuesto, es un índice de suma, es un índice mudo, como una variable de integración, j puede ser reemplazada por cualquier otro símbolo.

La generalización para tres, cuatro, o N dimensiones es ahora muy simple. El conjunto de N cantidades, V_j, se dice que son las componentes de un vector de N dimensiones, $\vec{V}$, si y sólo si sus valores relativos a los ejes coordenados rotados están dados por

$$V_i' = \sum_{j=1}^{N} a_{ij}V_j\ , \qquad i = 1, 2, \ldots, N\ . \tag{1.15}$$

Como antes, a_{ij} es el coseno del ángulo entre x_i' y x_j. A menudo el límite superior N y el correspondiente intervalo de i no será indicado. Se da por descontado que se sabe en qué dimensión se está trabajando.

[5]Podemos extrañarnos por qué hemos sustituido un parámetro φ por cuatro parámetros a_{ij}. Claramente, los a_{ij} no constituyen un conjunto mínimo de parámetros. Para dos dimensiones los cuatro a_{ij} están sujetos a tres restricciones dadas en la ecuación (1.19). La justificación para el conjunto redundante de cosenos directores es la conveniencia que provee. Se tiene la esperanza que esta conveniencia sea más aparente en los próximos capítulos. Para rotaciones en tres dimensiones son nueve los a_{ij}, pero solamente tres de ellos son independientes, existen además descripciones alternativas como: los ángulos de Euler, los cuaterniones o los parámetros de Cayley-Klein. Estas alternativas tienen sus respectivas ventajas y sus desventajas.

A partir de la definición de a_{ij} como el coseno del ángulo entre la dirección positiva de x_i' y la dirección positiva de x_j podemos escribir (coordenadas cartesianas)[6]

$$a_{ij} = \frac{\partial x_i'}{\partial x_j} \ . \tag{1.16}$$

Usando la rotación inversa ($\varphi \to -\varphi$) produce

$$x_j = \sum_{j=1}^{2} a_{ij} x_i' \ , \quad \text{o} \quad \frac{\partial x_j}{\partial x_i'} = a_{ij} \ . \tag{1.17}$$

Notemos que estas son derivadas parciales. Por uso de las ecuaciones (1.16), (1.17) y (1.15) se convierten en

$$V_i' = \sum_{j=1}^{N} \frac{\partial x_i'}{\partial x_j} V_j = \sum_{j=1}^{N} \frac{\partial x_j}{\partial x_i'} V_j \ . \tag{1.18}$$

Los cosenos directores a_{ij} satisfacen una *condición de ortogonalidad*

$$\sum_i a_{ij} a_{ik} = \delta_{jk} \ , \tag{1.19}$$

o, equivalentemente,

$$\sum_i a_{ji} a_{ki} = \delta_{jk} \ . \tag{1.20}$$

El símbolo δ_{ij} es la delta de Kronecker definida por

$$\delta_{ij} = \begin{cases} 1 & \text{para } i = j \ , \\ 0 & \text{para } i \neq j \ . \end{cases} \tag{1.21}$$

Podemos fácilmente verificar las ecuaciones (1.19) y (1.20) manteniéndose en el caso de dos dimensiones y sustituyendo los específicos a_{ij} desde la ecuación (1.11). El resultado es la bien conocida identidad $\text{sen}^2\,\varphi + \cos^2\varphi = 1$ para el caso no nulo. Para verificar la ecuación (1.19) en forma general, podemos usar la forma en derivadas parciales de las ecuaciones (1.16) y (1.17) para obtener

$$\sum_i \frac{\partial x_j}{\partial x_i'} \frac{\partial x_k}{\partial x_i'} = \sum_i \frac{\partial x_j}{\partial x_i'} \frac{\partial x_i'}{\partial x_k} = \frac{\partial x_j}{\partial x_k} \ . \tag{1.22}$$

El último paso sigue las reglas usuales para las derivadas parciales, suponiendo que x_j es una función de x_1', x_2', x_3', etc. El resultado final, $\partial x_j/\partial x_k$, es igual a δ_{ij}, ya que x_j y x_k son coordenadas lineales ($j \neq k$) que se suponen perpendiculares (en dos o tres dimensiones) u ortogonales (para cualquier número de dimensiones). Equivalentemente, podemos suponer que x_j y x_k con $j \neq k$ son variables totalmente independientes. Si $j = k$, la derivada parcial es claramente igual a 1. Al redefinir un vector en términos de cómo sus componentes transforman bajo rotaciones del sistema de coordenadas, deberíamos enfatizar dos puntos:

[6] Diferenciando $x_i' = \sum a_{ik} x_k$ con respecto a x_j.

1.- Esta definición se desarrolla porque es útil y apropiada en describir nuestro mundo físico. Nuestras ecuaciones vectoriales serán independientes de cualquier sistema particular de coordenadas. (El sistema de coordenadas no necesita ser cartesiano.) La ecuación vectorial siempre puede ser expresada en algún sistema particular de coordenadas y, para obtener resultados numéricos, deberíamos finalmente expresarla en algún sistema de coordenadas específico.

2.- Esta definición se puede generalizar abriendo la rama de la matemática conocida como análisis tensorial, (próximo capítulo).

El comportamiento de las componentes vectoriales bajo rotaciones de las coordenadas es usado en al sección 1.3 para probar que un producto escalar es un escalar, en la sección 1.4 para probar que un producto vectorial es un vector, y en la sección 1.6 para mostrar que la gradiente de un escalar, $\vec{\boldsymbol{\nabla}}\psi$, es un vector. Lo que resta de este capítulo deriva de las definiciones menos restrictivas de vector que las dadas en la sección 1.1.

1.2.1. Vectores y espacios vectoriales.

Es habitual en matemáticas etiquetar un vector en tres dimensiones $\vec{x}$ por un triplete ordenado de números reales (x_1, x_2, x_3). El número x_n es llamado la componente n del vector $\vec{x}$. El conjunto de todos los vectores (que obedecen las propiedades que siguen) forman un espacio vectorial sobre los reales en este caso de tres dimensiones. Atribuimos cinco propiedades a nuestros vectores: si $\vec{x}$ =(x_1, x_2, x_3) e $\vec{y} = (y_1, y_2, y_3)$,

1. Igualdad entre vectores : $\vec{x} = \vec{y}$ significa $x_i = y_i$, para i =1, 2, 3.

2. Suma de vectores: $\vec{x} + \vec{y} = \vec{z}$ significa $x_i + y_i = z_i$, para i =1,2,3.

3. Multiplicación por un escalar: $a\vec{x} \leftrightarrow (ax_1, ax_2, ax_3)$ con $a \in \mathbb{R}$.

4. El negativo de un vector o inverso aditivo: $-\vec{x} = (-1)\vec{x} \leftrightarrow (-x_1, -x_2, -x_3)$.

5. Vector nulo: aquí existe un vector nulo $\vec{0} \leftrightarrow (0, 0, 0)$.

Ya que nuestras componentes vector son números reales, las siguientes propiedades también se mantienen:

1. La suma de vectores es conmutativa: $\vec{x} + \vec{y} = \vec{y} + \vec{x}$.

2. La suma de vectores es asociativa: $(\vec{x} + \vec{y}) + \vec{z} = \vec{x} + (\vec{y} + \vec{z})$.

3. La multiplicación por escalar es distributiva: $a(\vec{x} + \vec{y}) = a\vec{x} + a\vec{y}$, y también se cumple $(a + b)\vec{x} = a\vec{x} + b\vec{x}$.

4. La multiplicación por escalar es asociativa: $(ab)\vec{x} = a(b\vec{x})$.

Además, el vector nulo $\vec{0}$ es único, tal como lo es el inverso aditivo de un vector $\vec{x}$ dado.

Esta formulación nos permite formalizar la discusión de componentes de la sección 1.1. Su importancia radica en las extensiones, las cuales serán consideradas en los capítulos posteriores. En el capítulo (4), mostraremos que los vectores forman tanto un grupo Abeliano bajo la suma, como un espacio lineal con las transformaciones en el espacio lineal descrito por matrices. Finalmente, y quizá lo más importante, para la Física avanzada el concepto de vectores presentado aquí puede ser generalizado a números complejos, funciones, y a un número infinito de componentes. Esto tiende a espacios de funciones de dimensión infinita, espacios de Hilbert, los cuales son importantes en la teoría cuántica moderna.

1.3. Producto escalar o producto punto.

Habiendo definido vectores, debemos proceder a combinarlos. Las leyes para combinar vectores deben ser matemáticamente consistentes. A partir de las posibilidades que existen seleccionamos dos que son tanto matemática como físicamente interesantes. Una tercera posibilidad es introducida en el capítulo 2, en la cual formaremos tensores.

La proyección de un vector $\vec{A}$ sobre los ejes coordenados, la cual define su componente cartesiana en la ecuación (1.4), es un caso especial del producto escalar de $\vec{A}$ y los vectores unitarios coordenados,

$$A_x = A\cos\alpha \equiv \vec{A}\cdot\hat{x}\ , \quad A_y = A\cos\beta \equiv \vec{A}\cdot\hat{y}\ , \quad A_z = A\cos\gamma \equiv \vec{A}\cdot\hat{z}\ . \tag{1.23}$$

Al igual que la proyección es lineal en $\vec{A}$, deseamos que el producto escalar de dos vectores sea lineal en $\vec{A}$ y $\vec{B}$, es decir, obedezca las leyes de distributividad y asociatividad

$$\vec{A}\cdot(\vec{B}+\vec{C}) = \vec{A}\cdot\vec{B} + \vec{A}\cdot\vec{C}\ , \tag{1.24}$$

$$\vec{A}\cdot y\vec{B} = (y\vec{A})\cdot\vec{B} = y\vec{A}\cdot\vec{B}\ , \tag{1.25}$$

donde y es un número. Ahora podemos usar la descomposición de $\vec{B}$ en sus componentes cartesianas de acuerdo a la ecuación (1.5), $\vec{B} = B_x\hat{x} + B_y\hat{y} + B_z\hat{z}$, para construir el producto escalar general o producto punto de los vectores $\vec{A}$ y $\vec{B}$ como

$$\begin{aligned}\vec{A}\cdot\vec{B} &= \vec{A}\cdot(B_x\hat{x} + B_y\hat{y} + B_z\hat{z}) \\ &= B_x\vec{A}\cdot\hat{x} + B_y\vec{A}\cdot\hat{y} + B_z\vec{A}\cdot\hat{z} \\ &= B_xA_x + B_yA_y + B_zA_z\ ,\end{aligned}$$

por lo tanto

$$\vec{A}\cdot\vec{B} \equiv \sum_i B_iA_i = \sum_i A_iB_i = \vec{B}\cdot\vec{A}\ . \tag{1.26}$$

Si $\vec{A} = \vec{B}$ en la ecuación (1.26), recuperamos la magnitud $A = (\sum A_i^2)^{1/2}$ de $\vec{A}$ en la ecuación (1.6) a partir de la ecuación (1.26).

Es obvio, a partir de la ecuación (1.26), que el producto escalar trata a $\vec{A}$ y $\vec{B}$ de la misma manera, i.e. es simétrico en $\vec{A}$ y $\vec{B}$, luego es conmutativo. Así, alternativamente y equivalentemente, podemos primero generalizar la ecuación (1.23) a la proyección A_B de un

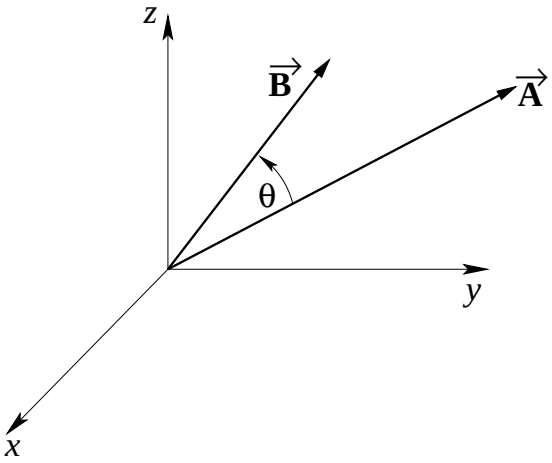


Figura 1.7: Producto escalar $\vec{A} \cdot \vec{B} = AB\cos\theta$.

vector $\vec{A}$ sobre la dirección de un vector $\vec{B} \neq 0$ como $A_B = A\cos\theta \equiv \vec{A} \cdot \hat{B}$, donde $\hat{B} = \vec{B}/B$ es el vector unitario en la dirección de $\vec{B}$ y θ es el ángulo entre $\vec{A}$ y $\vec{B}$ como muestra la figura 1.7. Similarmente, proyectamos $\vec{B}$ sobre $\vec{A}$ como $B_A = B\cos\theta \equiv \vec{B} \cdot \hat{A}$. Segundo, hacemos esta proyección simétrica en $\vec{A}$ y $\vec{B}$, la cual produce la definición

$$\vec{A} \cdot \vec{B} \equiv A_B B = AB_A = AB\cos\theta \ . \tag{1.27}$$

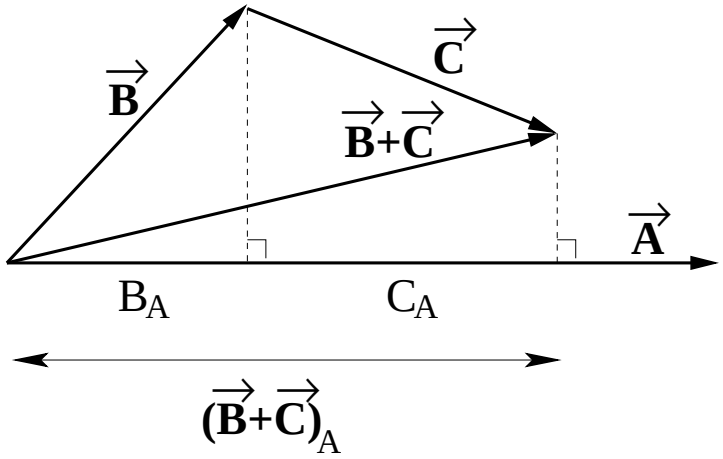


Figura 1.8: La ley distributiva $\vec{A} \cdot (\vec{B} + \vec{C}) = AB_A + AC_A = A(\vec{B} + \vec{C})_A$, ecuación (1.24).

La ley distributiva en la ecuación (1.24) es ilustrada en la figura 1.8, en la cual se muestra que la suma de las proyecciones de $\vec{B}$ y $\vec{C}$, $B_A + C_A$, es igual a la proyección de $\vec{B} + \vec{C}$ sobre $\vec{A}$, $(\vec{B} + \vec{C})_A$.

Se sigue a partir de las ecuaciones (1.23), (1.26) y (1.27) que los vectores unitarios coordenados satisfacen la relación

$$\hat{x} \cdot \hat{x} = \hat{y} \cdot \hat{y} = \hat{z} \cdot \hat{z} = 1 \ , \tag{1.28}$$

mientras

$$\hat{x} \cdot \hat{y} = \hat{x} \cdot \hat{z} = \hat{y} \cdot \hat{z} = 0 \ . \tag{1.29}$$

Si la definición de las componentes, ecuación (1.26), es etiquetada como una definición algebraica, entonces la ecuación (1.27) es una definición geométrica. Una de las más comunes aplicaciones del producto escalar en Física es el cálculo del trabajo ejercido por una fuerza constante = fuerza × desplazamiento × $\cos\theta$, lo cual es interpretado como el desplazamiento por la proyección de la fuerza en la dirección del desplazamiento, *i.e.*, el producto escalar de la fuerza y el desplazamiento, $W = \vec{F} \cdot \vec{S}$.

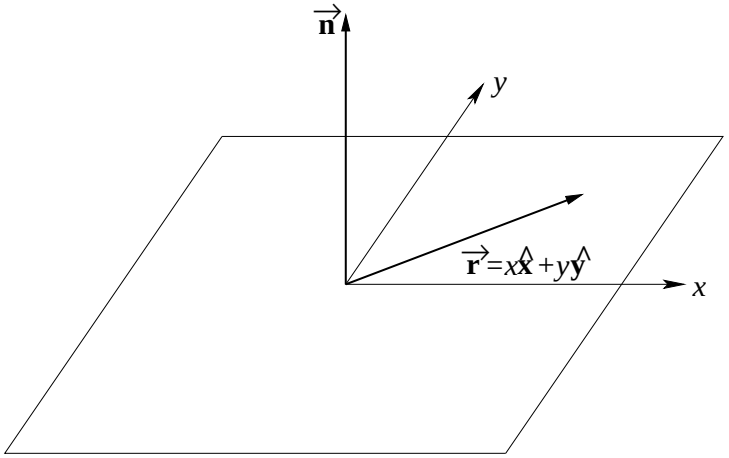


Figura 1.9: Un vector normal.

Si $\vec{A} \cdot \vec{B} = 0$ y sabemos que $\vec{A} \neq 0$ y $\vec{B} \neq 0$, entonces desde la ecuación (1.27), $\cos\theta = 0$ implica $\theta = \pi/2$ o $\theta = 3\pi/2$. Los vectores $\vec{A}$ y $\vec{B}$ deben ser perpendiculares. Alternativamente, podemos decir que son ortogonales. Los vectores unitarios $\hat{x}$, $\hat{y}$ y $\hat{z}$ son mutuamente ortogonales. Para desarrollar esta noción de ortogonalidad demos un paso más, supongamos que $\vec{n}$ es un vector unitario y $\vec{r}$ es un vector distinto de cero en el plano-xy; esto es $\vec{r} = \hat{x}x + \hat{y}y$ (figura 1.9). Si

$$\vec{n} \cdot \vec{r} = 0 \ ,$$

para todas las elecciones posibles de $\vec{r}$, entonces $\vec{n}$ debe ser perpendicular (ortogonal) al plano-xy.

A menudo es conveniente reemplazar $\hat{x}$, $\hat{y}$ y $\hat{z}$ por vectores unitarios con subíndice $\vec{e}_m$ con $m = 1, 2, 3$, con $\hat{x} = \vec{e}_1$ y así sucesivamente. Entonces las ecuaciones (1.28) y (1.29) llegan a ser

$$\vec{e}_m \cdot \vec{e}_n = \delta_{mn} \ . \tag{1.30}$$

Para $m \neq n$ los vectores unitarios $\vec{e}_m$ y $\vec{e}_n$ son ortogonales. Para $m = n$ cada vector es normalizado a la unidad, esto es, tiene magnitud uno. El conjunto de vectores $\vec{e}_m$ se dice que es *ortonormal*. La mayor ventaja de la ecuación (1.30) sobre las ecuaciones (1.28) y (1.29) es que la ecuación (1.30) puede ser fácilmente generalizada a un espacio de N dimensiones; $m, n = 1, 2, \ldots, N$. Finalmente, escogeremos un conjunto de vectores unitarios $\vec{e}_m$ que sean ortonormales por conveniencia.

1.3.1. Invariancia del producto escalar bajo rotaciones.

No hemos mostrado aún que la palabra escalar esté justificada o que el producto escalar sea realmente una cantidad escalar. Para hacer esto, investiguemos el comportamiento de $\vec{A}\cdot\vec{B}$ bajo una rotación del sistema de coordenadas. Usando la ecuación (1.15)

$$A'_xB'_x + A'_yB'_y + A'_zB'_z = \sum_i a_{xi}A_i \sum_j a_{xj}B_j + \sum_i a_{yi}A_i \sum_j a_{yj}B_j + \sum_i a_{zi}A_i \sum_j a_{zj}B_j \ . \tag{1.31}$$

Usando los índices k y l para sumar sobre x, y y z, obtenemos

$$\sum_k A'_kB'_k = \sum_l \sum_i \sum_j a_{li}A_i a_{lj}B_j \ , \tag{1.32}$$

y, arreglando términos en el lado derecho, tenemos

$$\sum_k A'_kB'_k = \sum_l \sum_i \sum_j (a_{li}a_{lj})A_iB_j = \sum_i \sum_j \delta_{ij}A_iB_j = \sum_i A_iB_i \ . \tag{1.33}$$

Los dos últimos pasos se siguen de la ecuación (1.19), la condición de ortogonalidad de los cosenos directores, y de la ecuación (1.21) la cual define la delta de Kronecker. El efecto de la delta de Kronecker es cancelar todos los términos en la suma sobre uno de sus dos índices excepto el término en el cual son iguales. En la ecuación (1.33) su efecto es fijar $j = i$ y eliminar la suma sobre j. Por supuesto, podríamos fijar $i = j$ y eliminar la suma sobre i. la ecuación (1.33) nos da

$$\sum_k A'_kB'_k = \sum_i A_iB_i \ , \tag{1.34}$$

la cual es justo nuestra definición de una cantidad escalar, una que permanece *invariante* ante rotaciones de los sistemas de coordenadas.

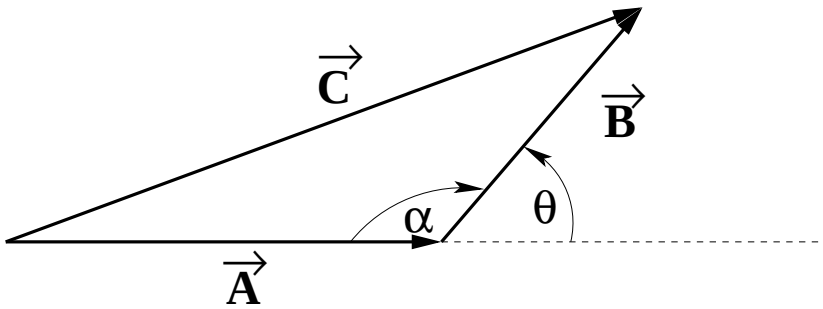


Figura 1.10: La ley de los cosenos.

En un acercamiento, similar el cual se aprovecha de este concepto de invarianza, tomamos $\vec{C} = \vec{A} + \vec{B}$ y hacemos producto punto consigo mismo.

$$\begin{aligned} \vec{C}\cdot\vec{C} &= (\vec{A}+\vec{B})\cdot(\vec{A}+\vec{B}) \\ &= \vec{A}\cdot\vec{A} + \vec{B}\cdot\vec{B} + 2\vec{A}\cdot\vec{B} \ . \end{aligned} \tag{1.35}$$

Ya que

$$\vec{C} \cdot \vec{C} = C^2 \ , \tag{1.36}$$

el cuadrado de la magnitud del vector $\vec{C}$ y por esto una cantidad invariante, veamos que

$$\vec{A} \cdot \vec{B} = \frac{C^2 - A^2 - B^2}{2} \ , \quad \text{es invariante.} \tag{1.37}$$

Ya que la mano derecha de la ecuación (1.37) es invariante –esto es, una cantidad escalar– el lado izquierdo, $\vec{A} \cdot \vec{B}$, también debiera ser invariante bajo una rotación del sistema de coordenadas. En consecuencia $\vec{A} \cdot \vec{B}$ es un escalar.

La ecuación (1.35) es realmente otra forma de escribir el teorema del coseno, el cual es

$$\begin{aligned} C^2 &= A^2 + B^2 + 2AB\cos\theta \ , \\ &= A^2 + B^2 + 2AB\cos(\pi - \alpha) \ , \\ &= A^2 + B^2 - 2AB\cos\alpha \ , \end{aligned} \tag{1.38}$$

Comparando las ecuaciones (1.35) y (1.38), tenemos otra verificación de la ecuación (1.27), o, si se prefiere, una derivación vectorial del teorema del coseno (figura 1.10).

El producto punto, dado por la ecuación (1.26), podría ser generalizado en dos formas. El espacio no necesita ser restringido a tres dimensiones. En un espacio n-dimensional, la ecuación se aplica con la suma corriendo desde 1 a n. Donde n puede ser infinito, con la suma como una serie convergente infinita. La otra generalización extiende el concepto de vector para abarcar las funciones. La funcional análoga a un producto punto o interno aparece más adelante.

1.4. Producto vectorial o producto cruz.

Una segunda forma de multiplicar vectores emplea el seno del ángulo sustentado en vez del coseno. Por ejemplo, el momento angular (figura 1.11) de un cuerpo está definido como

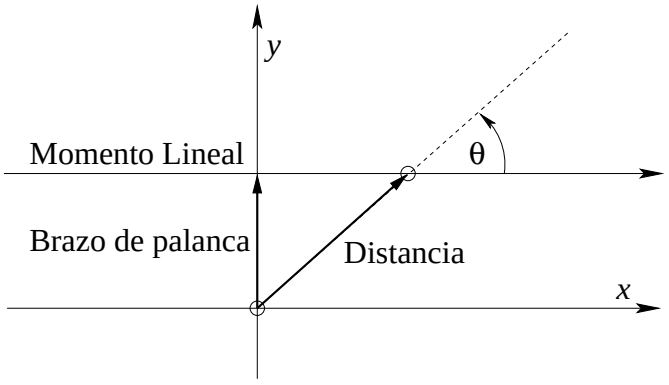


Figura 1.11: Momento angular.

$$\begin{aligned} \text{momento angular} &= \text{brazo de palanca} \times \text{momento lineal} \\ &= \text{distancia} \times \text{momento lineal} \times \operatorname{sen}\theta \ . \end{aligned}$$

Por conveniencia en el tratamiento de problemas relacionados con cantidades tales como momento angular, torque y velocidad angular, definiremos el producto vectorial o producto cruz como

$$\vec{C} = \vec{A} \times \vec{B} \ ,$$

con

$$C = AB \operatorname{sen} \theta \ . \tag{1.39}$$

A diferencia del caso anterior del producto escalar, $\vec{C}$ ahora es un vector, y le asignamos una dirección perpendicular al plano que contiene a $\vec{A}$ y $\vec{B}$ tal que $\vec{A}$, $\vec{B}$ y $\vec{C}$ forman un sistema diestro. Con esta elección de dirección tenemos

$$\vec{A} \times \vec{B} = -\vec{B} \times \vec{A} \ , \quad \text{anticonmutación.} \tag{1.40}$$

A partir de esta definición de producto cruz tenemos

$$\hat{x} \times \hat{x} = \hat{y} \times \hat{y} = \hat{z} \times \hat{z} = 0 \ , \tag{1.41}$$

mientras

$$\begin{array}{lll} \hat{x} \times \hat{y} = \ \ \hat{z} \ , & \hat{y} \times \hat{z} = \ \ \hat{x} \ , & \hat{z} \times \hat{x} = \ \ \hat{y} \ , \\ \hat{y} \times \hat{x} = -\hat{z} \ , & \hat{z} \times \hat{y} = -\hat{x} \ , & \hat{x} \times \hat{z} = -\hat{y} \ . \end{array} \tag{1.42}$$

Entre los ejemplos del producto cruz en Física Matemática están la relación entre el momento lineal $\vec{p}$ y el momento angular $\vec{L}$ (que define al momento angular),

$$\vec{L} = \vec{r} \times \vec{p} \ ,$$

y la relación entre velocidad lineal $\vec{v}$ y velocidad angular $\vec{\omega}$,

$$\vec{v} = \vec{\omega} \times \vec{r} \ .$$

Los vectores $\vec{v}$ y $\vec{p}$ describen propiedades de las partículas o un sistema físico. Sin embargo, la posición del vector $\vec{r}$ está determinado por la elección del origen de las coordenadas. Esto significa que $\vec{\omega}$ y $\vec{L}$ dependen de la elección del origen.

El campo magnético $\vec{B}$ usualmente está definido por el producto vectorial de la ecuación de fuerza de Lorentz[7]

$$\vec{F} = \frac{q}{c} \vec{v} \times \vec{B} \ , \quad (\text{CGS}).$$

Donde $\vec{v}$ es la velocidad de la carga eléctrica q, c es la velocidad de la luz en el vacío y $\vec{F}$ es la fuerza resultante sobre la carga en movimiento.

El producto cruz tiene una importante interpretación geométrica, la cual se usará en secciones futuras. En el paralelogramo definido por $\vec{A}$ y $\vec{B}$ (figura 1.12), $B \operatorname{sen} \theta$ es la altura si A es tomada como la longitud de la base. Luego $\left| (\vec{A} \times \vec{B}) \right| = AB \operatorname{sen} \theta$ es el área del paralelogramo. Como un vector, $\vec{A} \times \vec{B}$ es el área del paralelogramo definido por $\vec{A}$ y $\vec{B}$, con el vector normal al plano del paralelogramo. Esto sugiere que el área podría ser tratada como una cantidad vectorial.

[7]El campo eléctrico $\vec{E}$ lo hemos supuesto nulo.

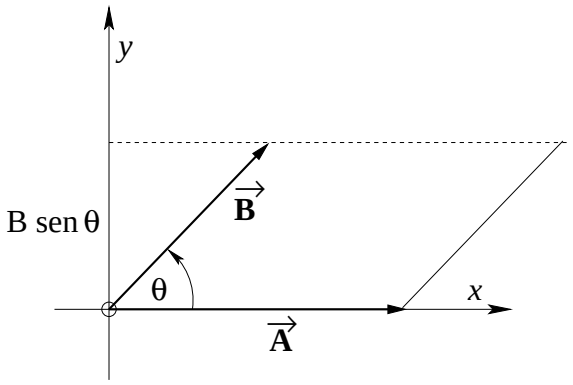


Figura 1.12: Paralelogramo que representa el producto cruz.

Entre paréntesis, podemos notar que la ecuación (1.42) y una ecuación modificada (1.41) forman el punto de partida para el desarrollo de los cuaterniones. La ecuación (1.41) es remplazada por $\hat{x} \times \hat{x} = \hat{y} \times \hat{y} = \hat{z} \times \hat{z} = -1$.

Una definición alternativa del producto vectorial puede ser derivada del caso especial de los vectores unitarios coordenados en las ecuaciones (1.40-1.42), en conjunto con la linealidad del producto cruz en ambos argumentos vectoriales, en analogía con la ecuación (1.24) y (1.25) para el producto punto,

$$\begin{aligned} \vec{A} \times (\vec{B} + \vec{C}) &= \vec{A} \times \vec{B} + \vec{A} \times \vec{C} \ , \\ (\vec{A} + \vec{B}) \times \vec{C} &= \vec{A} \times \vec{C} + \vec{B} \times \vec{C} \ , \end{aligned} \tag{1.43}$$

$$\vec{A} \times (y\vec{B}) = y\vec{A} \times \vec{B} = (y\vec{A}) \times \vec{B} \ , \tag{1.44}$$

donde y es un número. Usando la descomposición de $\vec{A}$ y $\vec{B}$ en sus componentes cartesianas de acuerdo a la ecuación (1.5), encontramos

$$\begin{aligned} \vec{A} \times \vec{B} \equiv \vec{C} &= (C_x, C_y, C_z) = (A_x\hat{x} + A_y\hat{y} + A_z\hat{z}) \times (B_x\hat{x} + B_y\hat{y} + B_z\hat{z}) \\ &= (A_xB_y - A_yB_x)\hat{x} \times \hat{y} + (A_xB_z - A_zB_x)\hat{x} \times \hat{z} + (A_yB_z - A_zB_y)\hat{y} \times \hat{z} \ , \end{aligned}$$

por aplicación de las ecuaciones (1.43),(1.44) y sustituyendo en las ecuaciones (1.40-1.42), tal que las componentes cartesianas de $\vec{A} \times \vec{B}$ sean

$$C_x = A_yB_z - A_zB_y \ , \quad C_y = A_zB_x - A_xB_z \ , \quad C_z = A_xB_y - A_yB_x \ , \tag{1.45}$$

o

$$C_i = A_jB_k - A_kB_j \ , \quad i, j, k \text{ todos diferentes}, \tag{1.46}$$

y con la permutación cíclica de los índices i, j, k. El producto vectorial $\vec{C}$ puede ser convenientemente representado por un determinante

$$\vec{C} = \det \begin{bmatrix} \hat{x} & \hat{y} & \hat{z} \\ A_x & A_y & A_z \\ B_x & B_y & B_z \end{bmatrix} \ . \tag{1.47}$$

La expansión del determinante por la fila superior reproduce las tres componentes de $\vec{C}$ listadas en la ecuación (1.45).

La ecuación (1.39) podría ser llamada una definición geométrica del producto vectorial. Luego la ecuación (1.45) podría ser una definición algebraica.

Para mostrar la equivalencia de la ecuación (1.39) y la definición de componente, ecuación (1.45), formemos $\vec{A}\cdot\vec{C}$ y $\vec{B}\cdot\vec{C}$, usando la ecuación (1.45). Tenemos

$$\begin{aligned}\vec{A}\cdot\vec{C} &= \vec{A}\cdot(\vec{A}\times\vec{B})\ ,\\ &= A_x(A_yB_z - A_zB_y) + A_y(A_zB_x - A_xB_z) + A_z(A_xB_y - A_yB_x)\ ,\\ &= 0\ .\end{aligned} \tag{1.48}$$

Similarmente,

$$\vec{B}\cdot\vec{C} = \vec{B}\cdot(\vec{A}\times\vec{B}) = 0\ . \tag{1.49}$$

Las ecuaciones (1.48) y (1.49) muestran que $\vec{C}$ es perpendicular tanto al vector $\vec{A}$ como al vector $\vec{B}$ ($\cos\theta = 0, \theta = \pm 90°$) y por lo tanto perpendicular al plano que ellos determinan. La dirección positiva está determinada considerando el caso especial de los vectores unitarios $\hat{x}\times\hat{y} = \hat{z}$ ($C_z = +A_xB_y$).

La magnitud es obtenida a partir de

$$\begin{aligned}(\vec{A}\times\vec{B})\cdot(\vec{A}\times\vec{B}) &= A^2B^2 - (\vec{A}\cdot\vec{B})^2\ ,\\ &= A^2B^2 - A^2B^2\cos^2\theta\ ,\\ &= A^2B^2\,\mathrm{sen}^2\,\theta\ .\end{aligned} \tag{1.50}$$

De donde

$$C = AB\,\mathrm{sen}\,\theta\ . \tag{1.51}$$

El gran primer paso en la ecuación (1.50) puede ser verificado expandiendo en componentes, usando la ecuación (1.45) para $\vec{A}\times\vec{B}$ y la ecuación (1.26) para el producto punto. A partir de las ecuaciones (1.48), (1.49) y (1.51) vemos la equivalencia de las ecuaciones (1.39) y (1.45), las dos definiciones del producto vectorial. Todavía permanece el problema de verificar que $\vec{C} = \vec{A}\times\vec{B}$ es por cierto un vector; esto es, que obedece la ecuación (1.15), la ley de transformación vectorial. Comencemos en un sistema rotado (sistema prima)

$$\begin{aligned}C_i' &= A_j'B_k' - A_k'B_j'\ , \quad i,\ j \text{ y } k \text{ en orden cíclico},\\ &= \sum_l a_{jl}A_l\sum_m a_{km}B_m - \sum_l a_{kl}A_l\sum_m a_{jm}B_m\ ,\\ &= \sum_{l,m}(a_{jl}a_{km} - a_{kl}a_{jm})A_lB_m\ .\end{aligned} \tag{1.52}$$

La combinación de cosenos directores en paréntesis desaparece para $m = l$. Por lo tanto tenemos que j y k toman valores fijos, dependiendo de la elección de i, y seis combinaciones de l y m. Si $i = 3$, luego $j = 1$, $k = 2$ (en orden cíclico), y tenemos la siguiente combinación

de cosenos directores[8]

$$\begin{aligned} a_{11}a_{22} - a_{21}a_{12} &= a_{33} \ , \\ a_{13}a_{21} - a_{23}a_{11} &= a_{32} \ , \\ a_{12}a_{23} - a_{22}a_{13} &= a_{31} \ , \end{aligned} \tag{1.53}$$

y sus negativos. Las ecuaciones (1.53) son identidades que satisfacen los cosenos directores. Ellas pueden ser verificadas con el uso de determinantes y matrices. Sustituyendo hacia atrás en la ecuación (1.52),

$$\begin{aligned} C_3' &= a_{33}A_1B_2 + a_{32}A_3B_1 + a_{31}A_2B_3 - a_{33}A_2B_1 - a_{32}A_1B_3 - a_{31}A_3B_2 \ , \\ &= a_{31}C_1 + a_{32}C_2 + a_{33}C_3 \ , \\ &= \sum_n a_{3n}C_n \ . \end{aligned} \tag{1.54}$$

Permutando los índices para levantar C_1' y C_2', vemos que la ecuación (1.15) se satisface y $\vec{C}$ es por cierto un vector. Podría mencionarse aquí que esta naturaleza vectorial del producto cruz es un accidente asociado con la naturaleza tridimensional del espacio ordinario[9]. Veremos en el capítulo siguiente que el producto cruz también puede ser tratado como un tensor asimétrico de rango dos.

Definimos un vector como un triplete ordenado de números (o funciones) como en la última parte de la sección 1.2, luego no hay problemas en identificar el producto cruz como un vector. La operación de producto cruz mapea los dos tripletes $\vec{A}$ y $\vec{B}$ en un tercer triplete $\vec{C}$, el cual por definición es un vector.

Ahora tenemos dos maneras de multiplicar vectores; una tercera forma aparece en el próximo capítulo . Pero ¿qué hay de la división por un vector? resulta ser que la razón $\vec{B}/\vec{A}$ no está unívocamente especificada a menos que $\vec{A}$ y $\vec{B}$ sean paralelos. Por lo tanto la división de un vector por otro no está bien definida.

1.5. Productos escalar triple y vectorial triple.

1.5.1. Producto escalar triple.

Las secciones 1.3 y 1.4 cubren los dos tipos de multiplicación de interés aquí. Sin embargo, hay combinaciones de tres vectores, $\vec{A}\cdot(\vec{B}\times\vec{C})$ y $\vec{A}\times(\vec{B}\times\vec{C})$, las cuales ocurren con la suficiente frecuencia para merecer una atención más amplia. La combinación

$$\vec{A}\cdot(\vec{B}\times\vec{C}) \ ,$$

es conocida como el producto escalar triple. El producto $\vec{B}\times\vec{C}$ produce un vector, el cual producto punto con $\vec{A}$, da un escalar. Notemos que $(\vec{A}\cdot\vec{B})\times\vec{C}$ representa un escalar producto

[8]La ecuación (1.53) se mantiene ante rotaciones porque ellas mantienen el volumen.

[9]Específicamente la ecuación (1.53) se mantiene sólo para un espacio tridimensional. Técnicamente, es posible definir un producto cruz en $\mathbb{R}^7$, el espacio de dimensión siete, pero el producto cruz resulta tener propiedades inaceptables (patológicas).

cruz con un vector, una operación que no está definida. Por lo tanto, si estamos de acuerdo en excluir estas interpretaciones no definidas, los paréntesis puede ser omitidos y el producto de escalar triple puede ser escrito como $\vec{A}\cdot\vec{B}\times\vec{C}$.

Usando la ecuación (1.45) para el producto cruz y la ecuación (1.26) para el producto punto, obtenemos

$$\begin{aligned}\vec{A}\cdot\vec{B}\times\vec{C} &= A_x(B_yC_z - B_zC_y) + A_y(B_zC_x - B_xC_z) + A_z(B_xC_y - B_yC_x)\ ,\\ &= \vec{B}\cdot\vec{C}\times\vec{A} = \vec{C}\cdot\vec{A}\times\vec{B}\ ,\\ &= -\vec{A}\cdot\vec{C}\times\vec{B} = -\vec{C}\cdot\vec{B}\times\vec{A} = -\vec{B}\cdot\vec{A}\times\vec{C}\ , \quad \text{y así sucesivamente.}\end{aligned} \tag{1.55}$$

Notemos el alto grado de simetría presente en la expansión en componentes. Cada término contiene los factores A_i, B_j y C_k. Si i, j, k están en un orden cíclico (x, y, z), el signo es positivo. Si el orden es anticíclico, el signo es negativo. Por lo tanto, el punto y la cruz pueden ser intercambiados,

$$\vec{A}\cdot\vec{B}\times\vec{C} = \vec{A}\times\vec{B}\cdot\vec{C}\ . \tag{1.56}$$

Una representación conveniente de la componente de expansión de la ecuación (1.55) está dada por el determinante

$$\vec{A}\cdot\vec{B}\times\vec{C} = \begin{vmatrix} A_x & A_y & A_z \\ B_x & B_y & B_z \\ C_x & C_y & C_z \end{vmatrix}\ . \tag{1.57}$$

La regla para intercambiar filas por columnas de un determinante provee una inmediata verificación de la permutación listada en la ecuación (1.55), mientras la simetría de $\vec{A}$, $\vec{B}$ y $\vec{C}$ en la forma determinante sugiere la relación dada en la ecuación (1.56).

El producto triple encontrado en la sección 1.4, en la cual se muestra que $\vec{A}\times\vec{B}$ era perpendicular a ambos, $\vec{A}$ y $\vec{B}$, es un caso especial del resultado general (ecuación (1.55)).

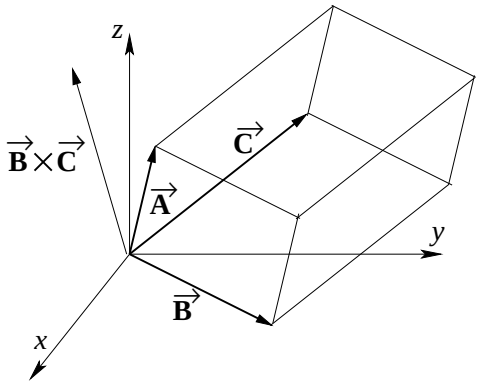


Figura 1.13: Paralelepípedo que representa el producto escalar triple.

El producto escalar triple tiene una interpretación geométrica directa. Los tres vectores $\vec{A}$, $\vec{B}$, y $\vec{C}$ pueden ser interpretados como la definición de un paralelepípedo (figura 1.13).

$$\begin{aligned}\left|\vec{B}\times\vec{C}\right| &= BC\,\mathrm{sen}\,\theta\ ,\\ &= \text{área de la base del paralelógramo.}\end{aligned} \tag{1.58}$$

La dirección, por supuesto, es normal a la base. Haciendo el producto punto con $\vec{A}$, esto significa multiplicar el área de la base, por la proyección de A sobre la normal, o la base tantas veces por la altura. Por lo tanto

$$\vec{A}\cdot\vec{B}\times\vec{C} = \text{volumen del paralelepípedo definido por } \vec{A},\ \vec{B} \text{ y } \vec{C}.$$

Este es el volumen del paralelepípedo definido por $\vec{A}$, $\vec{B}$ y $\vec{C}$. Debemos notar que $\vec{A}\cdot\vec{B}\times\vec{C}$ puede algunas veces volverse negativo. Este problema y su interpretación los consideraremos más adelante.

El producto escalar triple encuentra una interesante e importante aplicación en la construcción de una red recíproca cristalina. Sea $\vec{a}$, $\vec{b}$ y $\vec{c}$ (no necesariamente perpendiculares entre ellos) vectores que definen una red cristalina. La distancia desde un punto de la red a otro puede ser escrita

$$\vec{r} = n_a\vec{a} + n_b\vec{b} + n_c\vec{c}\ , \tag{1.59}$$

con n_a, n_b y n_c tomando los valores sobre los enteros. Con estos vectores podemos formar

$$\vec{a}\,' = \frac{\vec{b}\times\vec{c}}{\vec{a}\cdot\vec{b}\times\vec{c}}\ , \quad \vec{b}\,' = \frac{\vec{c}\times\vec{a}}{\vec{a}\cdot\vec{b}\times\vec{c}}\ , \quad \vec{c}\,' = \frac{\vec{a}\times\vec{b}}{\vec{a}\cdot\vec{b}\times\vec{c}}\ . \tag{1.60}$$

Vemos que $\vec{a}\,'$ es perpendicular al plano que contiene a $\vec{b}$ y a $\vec{c}$ y tiene una magnitud proporcional a a^{-1}. En efecto, podemos rápidamente mostrar que

$$\vec{a}\,'\cdot\vec{a} = \vec{b}\,'\cdot\vec{b} = \vec{c}\,'\cdot\vec{c} = 1\ , \tag{1.61}$$

mientras

$$\vec{a}'\cdot\vec{b} = \vec{a}'\cdot\vec{c} = \vec{b}'\cdot\vec{a} = \vec{b}'\cdot\vec{c} = \vec{c}'\cdot\vec{a} = \vec{c}'\cdot\vec{b} = 0\ . \tag{1.62}$$

Esto es a partir de las ecuaciones (1.61) y (1.62) derivamos el nombre de red recíproca. El espacio matemático en el cual esta red recíproca existe es llamado a veces espacio de Fourier. Esta red recíproca es útil en problemas que intervienen el *scattering* de ondas a partir de varios planos en un cristal. Más detalles pueden encontrarse en R.B. Leigthon's *Principles of Modern Physics*, pp. 440-448 [New York: McGraw-Hill (1995)].

1.5.2. Producto vectorial triple.

El segundo producto triple de interés es $\vec{A}\times(\vec{B}\times\vec{C})$, el cual es un vector. Aquí los paréntesis deben mantenerse, como puede verse del caso especial $(\hat{x}\times\hat{x})\times\hat{y} = 0$, mientras que si evaluamos $\hat{x}\times(\hat{x}\times\hat{y}) = \hat{x}\times\hat{z} = -\hat{y}$. El producto vectorial triple es perpendicular a $\vec{A}$ y a $\vec{B}\times\vec{C}$. El plano definido por $\vec{B}$ y $\vec{C}$ es perpendicular a $\vec{B}\times\vec{C}$ y así el producto triple yace en este plano (ver figura 1.14)

$$\vec{A}\times(\vec{B}\times\vec{C}) = x\vec{B} + y\vec{C}\ . \tag{1.63}$$

Multiplicando (1.63) por $\vec{A}$, lo que da cero para el lado izquierdo, tal que $x\vec{A}\cdot\vec{B} + y\vec{A}\cdot\vec{C} = 0$. De aquí que $x = z\vec{A}\cdot\vec{C}$ e $y = -z\vec{A}\cdot\vec{B}$ para un z apropiado. Sustituyendo estos valores en la ecuación (1.63) da

$$\vec{A}\times(\vec{B}\times\vec{C}) = z(\vec{B}\vec{A}\cdot\vec{C} - \vec{C}\vec{A}\cdot\vec{B}); \tag{1.64}$$

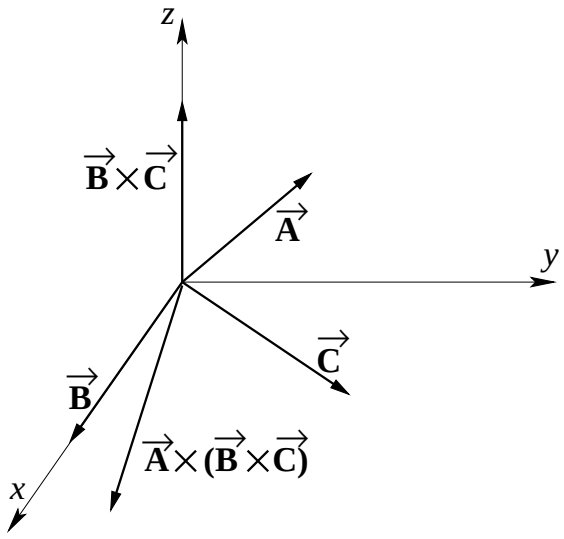


Figura 1.14: Los vectores $\vec{B}$ y $\vec{C}$ están en el plano xy. $\vec{B}\times\vec{C}$ es perpendicular al plano xy y es mostrado aquí a lo largo del eje z. Entonces $\vec{A}\times(\vec{B}\times\vec{C})$ es perpendicular al eje z y por lo tanto está de regreso en el plano xy.

deseamos mostrar que $z = 1$ en la ecuación (1.64), una importante relación algunas veces conocida como la regla $BAC - CAB$. Ya que la ecuación (1.64) es lineal en $\vec{A}$, $\vec{B}$ y $\vec{C}$, z es independiente de esas magnitudes. Entonces, sólo necesitamos mostrar que $z = 1$ para vectores unitarios $\hat{A}$, $\hat{B}$, $\hat{C}$. Denotemos $\hat{B}\cdot\hat{C} = \cos\alpha$, $\hat{C}\cdot\hat{A} = \cos\beta$, $\hat{A}\cdot\hat{B} = \cos\gamma$, y el cuadrado de la ecuación (1.64) para obtener

$$\begin{aligned}[\hat{A}\times(\hat{B}\times\hat{C})]^2 &= \hat{A}^2(\hat{B}\times\hat{C})^2 - [\hat{A}\cdot(\hat{B}\times\hat{C})]^2 = 1-\cos^2\alpha - [\hat{A}\cdot(\hat{B}\times\hat{C})]^2\ ,\\ &= z^2[(\hat{A}\cdot\hat{C})^2 + (\hat{A}\cdot\hat{B})^2 - 2\hat{A}\cdot\hat{B}\hat{A}\cdot\hat{C}\hat{B}\cdot\hat{C}]\ ,\\ &= z^2(\cos^2\beta + \cos^2\gamma - 2\cos\alpha\cos\beta\cos\gamma)\ ,\end{aligned} \tag{1.65}$$

usando $(\hat{v}\times\hat{w})^2 = \hat{v}^2\hat{w}^2 - (\hat{v}\cdot\hat{w})^2$ repetidas veces. Consecuentemente, el volumen abarcado por $\hat{A}$, $\hat{B}$ y $\hat{C}$ que aparece en la ecuación (1.65) puede ser escrito como

$$[\hat{A}\cdot(\hat{B}\times\hat{C})]^2 = 1-\cos^2\alpha - z^2(\cos^2\beta+\cos^2\gamma - 2\cos\alpha\cos\beta\cos\gamma)\ .$$

De aquí $z^2 = 1$, ya que este volumen es simétrico en α, β y γ. Esto es $z = \pm 1$ e independiente de $\hat{A}$, $\hat{B}$ y $\hat{C}$. Usando el caso especial $\hat{x}\times(\hat{x}\times\hat{y}) = -\hat{y}$ en la ecuación (1.64) finalmente da $z = 1$.

Una derivación alternativa usando el tensor de Levi-Civita ϵ_{ijk},

$$\epsilon_{ijk} = \begin{cases} 1 & \text{para } i,j,k{=}x,y,z \text{ o } y,z,x \text{ y } z,x,y,\\ -1 & \text{para } i,j,k{=}y,x,z \text{ o } x,z,y \text{ y } z,y,x,\\ 0 & \text{en otros casos.}\end{cases}$$

La veremos más adelante.

Podemos notar aquí que los vectores son independientes de las coordenadas tal que una ecuación vectorial es independiente de un particular sistema de coordenadas. El sistema de coordenadas sólo determina las componentes. Si la ecuación vectorial puede ser establecida en coordenadas cartesianas, ella se puede establecer y es válida en cualquier sistema de coordenadas que será introducido en el próximo capítulo.

1.6. Gradiente, $\vec{\nabla}$

Suponga que $\varphi(x, y, z)$ es una función de punto escalar, esto es, una función cuyo valor depende de los valores de las coordenadas (x, y, z). Como un escalar, debería tener el mismo valor en un punto fijo dado en el espacio, independiente de la rotación de nuestro sistema de coordenadas, o

$$\varphi'(x_1', x_2', x_3') = \varphi(x_1, x_2, x_3) \ . \tag{1.66}$$

Diferenciando con respecto a x_i' obtenemos

$$\frac{\partial\varphi'(x_1', x_2', x_3')}{\partial x_i'} = \frac{\partial\varphi(x_1, x_2, x_3)}{\partial x_i'} = \sum_j \frac{\partial\varphi}{\partial x_j}\frac{\partial x_j}{\partial x_i'} = \sum_j a_{ij}\frac{\partial\varphi}{\partial x_j} \ , \tag{1.67}$$

por las reglas de diferenciación parcial y las ecuaciones (1.16) y (1.17). Pero la comparación con la ecuación (1.18), la ley de transformación vectorial, ahora nos muestra que hemos construido un vector con componentes $\partial\varphi/\partial x_j$. Este vector lo etiquetamos como la gradiente de φ. Un simbolismo conveniente es

$$\vec{\nabla}\varphi = \hat{x}\frac{\partial\varphi}{\partial x} + \hat{y}\frac{\partial\varphi}{\partial y} + \hat{z}\frac{\partial\varphi}{\partial z} \tag{1.68}$$

o

$$\vec{\nabla} = \hat{x}\frac{\partial}{\partial x} + \hat{y}\frac{\partial}{\partial y} + \hat{z}\frac{\partial}{\partial z} \ . \tag{1.69}$$

$\vec{\nabla}\varphi$ (o delφ o gradφ) es nuestro gradiente del campo escalar φ, mientras $\vec{\nabla}$ es por sí mismo un operador diferencial vectorial (disponible para operar sobre o para diferenciar un campo escalar φ). Todas las relaciones para $\vec{\nabla}$ pueden ser derivadas a partir de la naturaleza híbrida del gradiente, por un lado la expresión en términos de derivadas parciales y por otra su naturaleza vectorial.

Ejemplo: el gradiente de una función de r.

Calculemos el gradiente de $f(r) = f(\sqrt{x^2 + y^2 + z^2})$.

$$\vec{\nabla}f(r) = \hat{x}\frac{\partial f(r)}{\partial x} + \hat{y}\frac{\partial f(r)}{\partial y} + \hat{z}\frac{\partial f(r)}{\partial z} \ .$$

La dependencia de $f(r)$ sobre x es a través de la dependencia de r sobre x. Por lo tanto[10]

$$\frac{\partial f(r)}{\partial x} = \frac{df(r)}{dr}\frac{\partial r}{\partial x} \ .$$

De r como función de x, y, z

$$\frac{\partial r}{\partial x} = \frac{\partial\sqrt{x^2+y^2+z^2}}{\partial x} = \frac{x}{\sqrt{x^2+y^2+z^2}} = \frac{x}{r} \ .$$

Por lo tanto

$$\frac{\partial f(r)}{\partial x} = \frac{df(r)}{dr}\frac{x}{r} \ .$$

Permutando las coordenadas obtenemos las otras derivadas, para que finalmente podamos escribir

$$\vec{\boldsymbol{\nabla}} f(r) = (\hat{x}x + \hat{y}y + \hat{z}z)\,\frac{1}{r}\frac{df}{dr} = \frac{\vec{r}}{r}\frac{df}{dr} = \hat{r}\frac{df}{dr} \ .$$

Aquí $\hat{r}$ es un vector unitario $\vec{r}/r$ en la dirección radial positiva. El gradiente de una función de r es un vector en la dirección radial.

1.6.1. Una interpretación geométrica

Una aplicación inmediata de $\vec{\boldsymbol{\nabla}}\varphi$ es hacer el producto punto contra un diferencial de camino o diferencial de longitud

$$d\vec{s} = \hat{x}dx + \hat{y}dy + \hat{z}dz \ . \tag{1.70}$$

Así obtenemos

$$(\vec{\boldsymbol{\nabla}}\varphi)\cdot d\vec{s} = \frac{\partial\varphi}{\partial x}dx + \frac{\partial\varphi}{\partial y}dy + \frac{\partial\varphi}{\partial z}dz = d\varphi \ , \tag{1.71}$$

el cambio en la función escalar φ corresponde al cambio de posición ds. Ahora consideremos P y Q para ser dos puntos sobre una superficie $\varphi(x,y,z) = C$, una constante. Esos puntos son escogidos tal que Q está a un distancia $d\vec{s}$ de P. Entonces moviéndose de P a Q, el cambio en $\varphi(x,y,z) = C$ está dado por

$$d\varphi = (\vec{\boldsymbol{\nabla}}\varphi)\cdot d\vec{s} = 0 \ , \tag{1.72}$$

ya que permanecemos sobre la superficie $\varphi(x,y,z) = C$. Esto muestra que $\vec{\boldsymbol{\nabla}}\varphi$ es perpendicular a $d\vec{s}$. Ya que $d\vec{r}$ puede tener cualquier dirección desde P mientras permanezca en la superficie φ, el punto Q está restringido a la superficie, pero teniendo una dirección arbitraria, $\vec{\boldsymbol{\nabla}}\varphi$ es normal a la superficie $\varphi =$ constante (figura 1.15).

[10]Este es un caso especial de la regla de la cadena para derivadas parciales:

$$\frac{\partial f(r,\theta,\varphi)}{\partial x} = \frac{\partial f}{\partial r}\frac{\partial r}{\partial x} + \frac{\partial f}{\partial\theta}\frac{\partial\theta}{\partial x} + \frac{\partial f}{\partial\varphi}\frac{\partial\varphi}{\partial x} \ .$$

Ya que $\partial f/\partial\theta = \partial f/\partial\varphi = 0$, $\partial f/\partial r \to df/dr$.

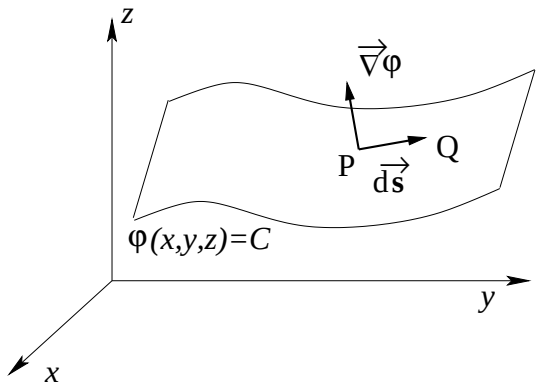


Figura 1.15: Requerimos que el diferencial de longitud $d\vec{s}$ permanezca sobre la superficie $\varphi = C$.

Si ahora permitimos que $d\vec{s}$ vaya desde la superficie $\varphi = C_1$ a una superficie adyacente $\varphi = C_2$ (figura 1.16),

$$d\varphi = C_2 - C_1 = \Delta C = (\vec{\boldsymbol{\nabla}}\varphi) \cdot d\vec{s}\ . \tag{1.73}$$

Para un $d\varphi$ dado, $|\,d\vec{s}\,|$ es un mínimo cuando se escoge paralelo a $\vec{\boldsymbol{\nabla}}\varphi\,(\cos\theta = 1)$; o, para un $|\,d\vec{r}\,|$ dado, el cambio en la función escalar φ es maximizado escogiendo $d\vec{s}$ paralelo a $\vec{\boldsymbol{\nabla}}\varphi$. *Esto identifica $\vec{\boldsymbol{\nabla}}\varphi$ como un vector que tiene la dirección de la máxima razón espacial de cambio de φ*, una identificación que será útil en el próximo capítulo cuando consideremos sistemas de coordenadas no cartesianos.

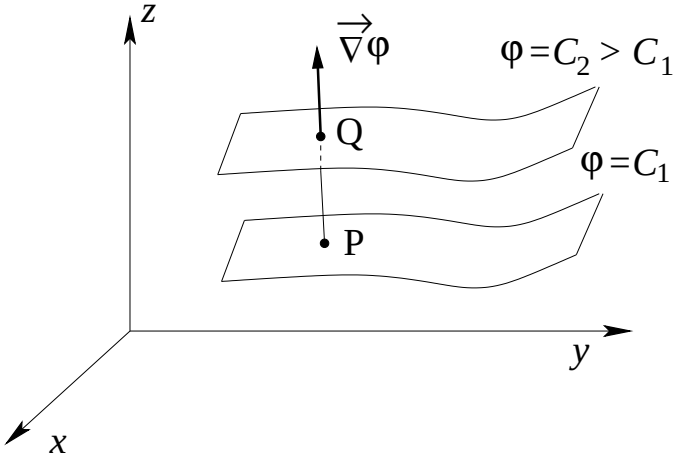


Figura 1.16: Gradiente.

Esta identificación de $\vec{\boldsymbol{\nabla}}\varphi$ también puede ser desarrollada usando el cálculo de variaciones sujeto a constricciones.

El gradiente de un campo escalar es de extrema importancia en Física al expresar la relación entre un campo de fuerza y un campo potencial.

$$\vec{F} = -\vec{\boldsymbol{\nabla}}(\text{energía potencial})\ . \tag{1.74}$$

Esto es ilustrado para ambos campos, gravitacional y electroestático, entre otros. Debemos notar que el signo menos en la ecuación (1.74) viene del hecho que el agua fluye cerro abajo y no cerro arriba.

1.7. Divergencia, $\vec{\nabla}\cdot$.

Diferenciar una función vectorial es una extensión simple de diferenciación de cantidades escalares. Supongamos que $\vec{r}(t)$ describe la posición de un satélite en algún tiempo t. Luego, por diferenciación con respecto al tiempo,

$$\begin{aligned}\frac{d\vec{r}(t)}{dt} &= \lim_{\Delta t\to 0}\frac{\vec{r}(t+\Delta t)-\vec{r}(t)}{\Delta t}\ ,\\ &= \vec{v}\ , \quad \text{la velocidad lineal.}\end{aligned}$$

Gráficamente, tenemos la pendiente de la curva, órbita, o trayectoria, como se muestra en la figura 1.17.

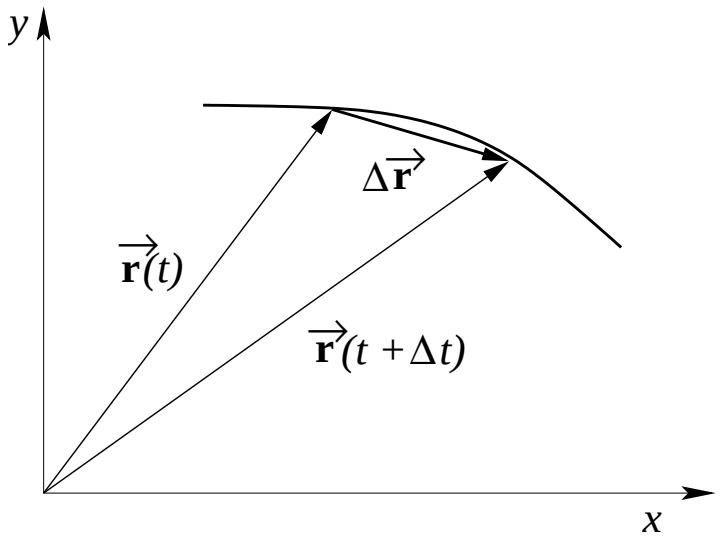


Figura 1.17: Diferenciación de un vector.

Si resolvemos $r(t)$ dentro de las componentes cartesianas, dr/dt siempre se reduce a una suma vectorial de no más que tres (para el espacio tridimensional) derivadas escalares. En otro sistema de coordenadas la situación es un poco más complicada, los vectores unitarios no son más constantes en dirección. La diferenciación con respecto al espacio de coordenadas se maneja de la misma manera como la diferenciación con respecto al tiempo, como veremos en los siguientes párrafos.

En la sección 1.6, $\vec{\nabla}$ fue definido como un operador vectorial. Ahora, poniendo cuidadosamente atención a ambas: sus propiedades vectoriales y sus propiedades diferenciales, operemos sobre un vector. Primero, como un vector le hacemos producto punto con un segundo vector para obtener

$$\vec{\nabla}\cdot\vec{V} = \frac{\partial V_x}{\partial x}+\frac{\partial V_y}{\partial y}+\frac{\partial V_z}{\partial z}\ , \tag{1.75}$$

lo que se conoce como divergencia de $\vec{V}$. Este es un escalar, como se discutió en la sección 1.3.

Ejemplo Calculemos $\vec{\nabla}\cdot\vec{r}$

$$\vec{\nabla}\cdot\vec{r}=\left(\hat{x}\frac{\partial}{\partial x}+\hat{y}\frac{\partial}{\partial y}+\hat{z}\frac{\partial}{\partial z}\right)\cdot(\hat{x}x+\hat{y}y+\hat{z}z)=\frac{\partial x}{\partial x}+\frac{\partial y}{\partial y}+\frac{\partial z}{\partial z}\ ,$$

$$\vec{\nabla}\cdot\vec{r}=3\ .$$

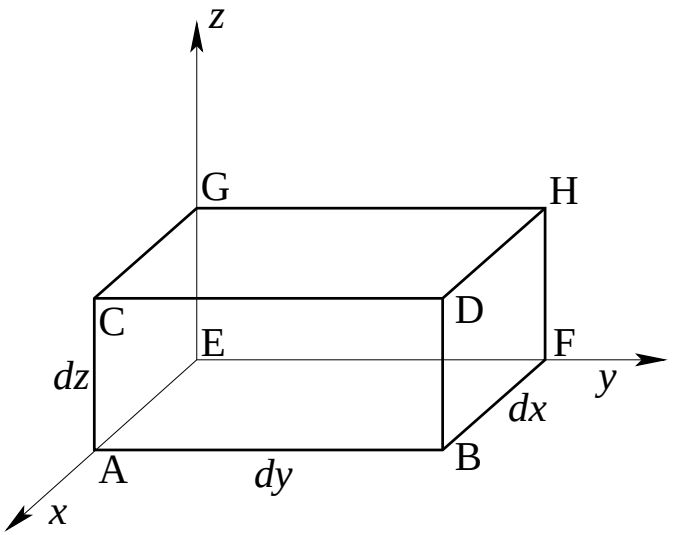


Figura 1.18: Diferencial paralelepípedo rectangular (en el primer octante).

1.7.1. Una interpretación física.

Para desarrollar una intuición del sentido físico de la divergencia, consideramos $\vec{\nabla}\cdot(\rho\vec{v})$ con $\vec{v}(x,y,z)$, la velocidad de un fluido compresible $\rho(x,y,z)$, su densidad en el punto (x,y,z). Si consideramos un pequeño volumen dx,dy,dz (figura 1.18), el fluido fluye dentro de su volumen por unidad de tiempo (dirección positiva de x) a través de la cara $EFGH$ es (masa de fluido que sale por unidad de tiempo)$_{EFGH}$ $=\rho v_x|_{x=0}\,dydz$. Los componentes del flujo, ρv_y y ρv_z, tangenciales a esta cara no contribuyen en nada al flujo en esa cara. La masa de fluido por unidad de tiempo que pasa (todavía en la dirección positiva de x) a través de la cara $ABCD$ es $\rho v_x|_{x=dx}dydz$. Para comparar estos flujos y para encontrar el flujo neto, expandimos este último resultado en una serie de Maclaurin. Este produce

$$\begin{aligned}\text{(La masa de fluido que sale por unidad de tiempo)}_{ABCD} &= \rho v_x|_{x=dx}\,dydz\ ,\\ &= \left[\rho v_x+\frac{\partial}{\partial x}(\rho v_x)dx\right]_{x=0}dydz\ .\end{aligned}$$

Aquí el término de la derivada es un primer término de corrección permitiendo la posibilidad de no uniformidad de la densidad o velocidad o ambas[11]. El término de orden cero $\rho v_x|_{x=0}$ (correspondiente al flujo uniforme) se cancela.

$$\text{(La masa neta de fluido que sale por unidad de tiempo)}_x = \frac{\partial}{\partial x}(\rho v_x)dxdydz \ .$$

Equivalentemente, podemos llegar a este resultado por

$$\lim_{\Delta x\to 0}\frac{\rho v_x(\Delta x,0,0)-\rho v_x(0,0,0)}{\Delta x}\equiv\left.\frac{\partial \rho v_x(x,y,z)}{\partial x}\right|_{0,0,0}$$

Ahora el eje x no requiere ningún tratamiento especial. El resultado precedente para las dos caras perpendiculares al eje x debe mantenerse para las dos caras perpendiculares al eje y, reemplazando x por y y los correspondientes cambios para y y z: $y\to z$, $z\to x$. Esta es una permutación cíclica de las coordenadas. Una ulterior permutación cíclica produce los resultados para las restantes dos caras de nuestro paralelepípedo. Sumando la velocidad de flujo neto para los tres pares de superficies de nuestro elemento de volumen, tenemos

$$\begin{aligned}\text{la masa neta de fluido que sale} &= \left[\frac{\partial}{\partial x}(\rho v_x)+\frac{\partial}{\partial y}(\rho v_y)+\frac{\partial}{\partial z}(\rho v_z)\right]dx\,dy\,dz\\ \text{(por unidad de tiempo)}\qquad &= \vec{\boldsymbol{\nabla}}\cdot(\rho\vec{v})\,dx\,dy\,dz \ .\end{aligned} \tag{1.76}$$

Por lo tanto, el flujo neto de nuestro fluido compresible del elemento volumen $dx\,dy\,dz$ por unidad de volumen por unidad de tiempo es $\vec{\boldsymbol{\nabla}}\cdot(\rho\vec{v})$. De aquí el nombre de *divergencia*. Una aplicación directa es en la ecuación de continuidad

$$\frac{\partial\rho}{\partial t}+\vec{\boldsymbol{\nabla}}\cdot(\rho\vec{v})=0 \ , \tag{1.77}$$

la cual simplemente establece que el flujo neto resulta en una disminución de la densidad dentro del volumen. Note que en la ecuación (1.77), ρ es considerada como una función posible del tiempo tanto como del espacio: $\rho(x,y,z,t)$. La divergencia aparece en una amplia variedad de problemas físicos, pasando desde una densidad de corriente de probabilidad en mecánica cuántica, a la pérdida de neutrones en un reactor nuclear.

La combinación $\vec{\boldsymbol{\nabla}}\cdot(f\vec{V})$, en la cual f es una función escalar y $\vec{V}$ una función vectorial, puede ser escrita

$$\begin{aligned}\vec{\boldsymbol{\nabla}}\cdot(f\vec{V}) &= \frac{\partial}{\partial x}(fV_x)+\frac{\partial}{\partial y}(fV_y)+\frac{\partial}{\partial z}(fV_z)\\ &= \frac{\partial f}{\partial x}V_x+f\frac{\partial V_x}{\partial x}+\frac{\partial f}{\partial y}V_y+f\frac{\partial V_y}{\partial y}+\frac{\partial f}{\partial z}V_z+f\frac{\partial V_z}{\partial z}\\ &= (\vec{\boldsymbol{\nabla}}f)\cdot\vec{V}+f\vec{\boldsymbol{\nabla}}\cdot\vec{V} \ ,\end{aligned} \tag{1.78}$$

[11]Estrictamente hablando, ρv_x es promediado sobre la cara $EFGH$ y la expresión $\rho v_x+\partial/\partial x(\rho v_x)dx$ es también promediada sobre la cara $ABCD$. Usando un diferencial de volumen arbitrariamente pequeño, encontramos que los promedios se reducen a los valores empleados aquí.

la cual es justo lo que uno debería esperar para la derivada de un producto. Notemos que $\vec{\boldsymbol{\nabla}}$ como un operador diferencial, diferencia a ambos, f y $\vec{V}$; como un vector, se le hace producto punto con $\vec{V}$ (en cada término).

Si tenemos el caso especial de la divergencia de un campo vectorial igual a cero,

$$\vec{\boldsymbol{\nabla}} \cdot \vec{B} = 0 \ , \tag{1.79}$$

el vector $\vec{B}$ se dice que es *solenoidal*, el término viene del ejemplo en cual $\vec{B}$ es el campo o inducción magnética y la ecuación (1.79) aparece como una de las ecuaciones de Maxwell. Cuando un vector es solenoidal puede ser escrito como el rotor de otro vector conocido como el vector potencial. Más adelante calcularemos tal vector potencial.

1.8. Rotor, $\vec{\boldsymbol{\nabla}}\times$

Otra posible operación con el operador vectorial $\vec{\boldsymbol{\nabla}}$ es hacerlo producto cruz contra otro vector. Obtenemos

$$\begin{aligned} \vec{\boldsymbol{\nabla}} \times \vec{V} &= \hat{x}\left(\frac{\partial}{\partial y}V_z - \frac{\partial}{\partial z}V_y\right) + \hat{y}\left(\frac{\partial}{\partial z}V_x - \frac{\partial}{\partial x}V_z\right) + \hat{z}\left(\frac{\partial}{\partial x}V_y - \frac{\partial}{\partial y}V_x\right) \\ &= \begin{vmatrix} \hat{x} & \hat{y} & \hat{z} \\ \dfrac{\partial}{\partial x} & \dfrac{\partial}{\partial y} & \dfrac{\partial}{\partial z} \\ V_x & V_y & V_z \end{vmatrix} \ , \end{aligned} \tag{1.80}$$

el cual es llamado el rotor de $\vec{V}$. En la expansión de este determinante debemos considerar la naturaleza diferencial de $\vec{\boldsymbol{\nabla}}$. Específicamente, $\vec{V} \times \vec{\boldsymbol{\nabla}}$ está definido sólo como un operador, como otro operador diferencial. Ciertamente no es igual, en general, a $-\vec{\boldsymbol{\nabla}} \times \vec{V}$[12]. En el caso de la ecuación (1.80) el determinante debe ser expandido desde arriba hacia abajo, tal que obtengamos las derivadas correctas. Si hacemos producto cruz contra el producto de un escalar por un vector, podemos ver que

$$\begin{aligned} \vec{\boldsymbol{\nabla}} \times (f\vec{V})\big|_x &= \left[\frac{\partial f V_z}{\partial y} - \frac{\partial f V_y}{\partial z}\right] \\ &= \left(f\frac{\partial V_z}{\partial y} + \frac{\partial f}{\partial y}V_z - f\frac{\partial V_y}{\partial z} + \frac{\partial f}{\partial z}V_y\right) \\ &= f\vec{\boldsymbol{\nabla}} \times (\vec{V})\big|_x + (\vec{\boldsymbol{\nabla}} f) \times \vec{V}\big|_x \ . \end{aligned} \tag{1.81}$$

Si permutamos las coordenadas tenemos

$$\vec{\boldsymbol{\nabla}} \times (f\vec{V}) = f\vec{\boldsymbol{\nabla}} \times (\vec{V}) + (\vec{\boldsymbol{\nabla}} f) \times \vec{V} \ , \tag{1.82}$$

la cual es el producto vectorial análogo a (1.78). De nuevo, como operador diferencial diferencia a ambos f y $\vec{V}$. Como vector hace producto cruz contra $\vec{V}$ en cada término.

[12]En este mismo espíritu, si $\vec{A}$ es un operador diferencial, no necesariamente es cierto que $\vec{A} \times \vec{A} = 0$. Específicamente, en mecánica cuántica el *operador* de momento angular, $\vec{L} = -i(\vec{r} \times \vec{\boldsymbol{\nabla}})$, encontramos que $\vec{L} \times \vec{L} = i\vec{L}$.

Ejemplo Calculemos $\vec{\boldsymbol{\nabla}} \times \vec{r} f(r)$. Por la ecuación (1.82),

$$\vec{\boldsymbol{\nabla}} \times \vec{r} f(r) = f(r) \vec{\boldsymbol{\nabla}} \times \vec{r} + [\vec{\boldsymbol{\nabla}} f(r)] \times \vec{r} \, . \tag{1.83}$$

Primero,

$$\vec{\boldsymbol{\nabla}} \times \vec{r} = \begin{vmatrix} \hat{x} & \hat{y} & \hat{z} \\ \dfrac{\partial}{\partial x} & \dfrac{\partial}{\partial y} & \dfrac{\partial}{\partial z} \\ x & y & z \end{vmatrix} = 0 \, . \tag{1.84}$$

Segundo, usando $\vec{\boldsymbol{\nabla}} f(r) = \hat{r}(df/dr)$, obtenemos

$$\vec{\boldsymbol{\nabla}} \times \vec{r} f(r) = \frac{df}{dr} \hat{r} \times \vec{r} \, . \tag{1.85}$$

El producto vectorial se anula, ya que $\vec{r} = \hat{r}\, r$ y $\hat{r} \times \hat{r} = 0$.

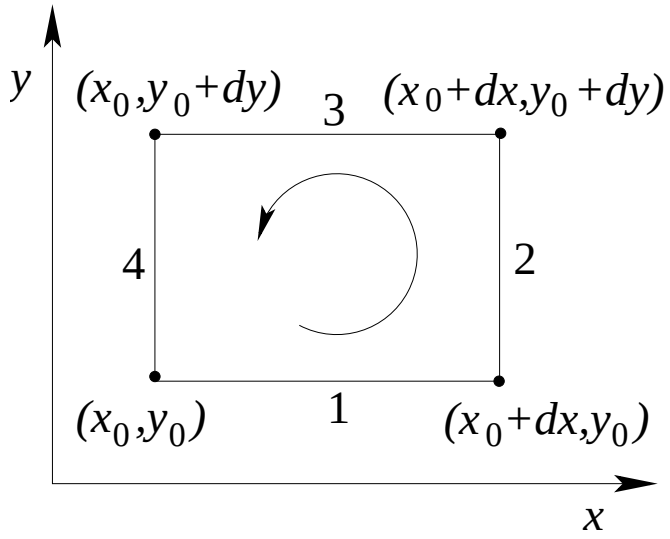


Figura 1.19: Circulación alrededor de un *loop* diferencial.

Para desarrollar un mejor sentido del significado físico del rotor, consideremos la circulación de un fluido alrededor de un *loop* diferencial en el plano xy, figura 1.19.

Aunque la circulación técnicamente está dada por la integral lineal de un vector $\int \vec{V} \cdot d\vec{\lambda}$ (sección 1.10), podemos establecer las integrales escalares equivalentes. Tomemos la circulación como

$$\text{circulación}_{1234} = \int_1 V_x(x,y) d\lambda_x + \int_2 V_y(x,y) d\lambda_y + \int_3 V_x(x,y) d\lambda_x + \int_4 V_y(x,y) d\lambda_y \, . \tag{1.86}$$

Los números 1, 2, 3 y 4 se refieren al segmento lineal enumerado en la figura 1.19. En la primera integral $d\lambda_x = +dx$, pero en la tercera integral $d\lambda_x = -dx$, ya que el tercer segmento de línea es recorrido en la dirección negativa de x. Similarmente, $d\lambda_y = +dy$, para la segunda integral y $-dy$ para la cuarta. Luego, los integrandos están referidos al punto (x_0, y_0) con una

expansión de Taylor[13] tomando en cuenta el desplazamiento del segmento de línea 3 desde 1 y 2 desde 4. Para nuestro segmento de línea diferencial esto tiende a

$$\begin{aligned}\text{ciculación}_{1234} &= V_x(x_0,y_0)dx + \left[V_y(x_0,y_0) + \frac{\partial V_y}{\partial x}dx\right]dy + \\ &+ \left[V_x(x_0,y_0) + \frac{\partial V_x}{\partial y}dy\right](-dx) + V_y(x_0,y_0)(-dy) \\ &= \left(\frac{\partial V_y}{\partial x} - \frac{\partial V_x}{\partial y}\right)dxdy\ .\end{aligned} \tag{1.87}$$

Dividiendo por $dxdy$, tenemos

$$\text{circulación por unidad de área} = \vec{\nabla}\times\vec{V}\big|_z. \tag{1.88}$$

La circulación[14] alrededor de nuestra área diferencial en el plano xy está dada por la componente z de $\vec{\nabla}\times\vec{V}$. En principio, el rotor, $\vec{\nabla}\times\vec{V}$ en (x_0,y_0), podría ser determinado insertando una rueda con paleta (diferencial) dentro del fluido en movimiento en el punto (x_0,y_0). La rotación de la pequeña rueda de paleta podría ser una medida del rotor, y su eje a lo largo de la dirección de $\vec{\nabla}\times\vec{V}$, la cual es perpendicular al plano de circulación.

Usaremos el resultado, ecuación (1.87), en la sección 1.13 para derivar el teorema de Stokes. Cada vez que el rotor de un vector $\vec{V}$ se anula,

$$\vec{\nabla}\times\vec{V} = 0\ , \tag{1.89}$$

$\vec{V}$ es llamado irrotacional. Los ejemplos físicos más importantes de vectores irrotacionales son las fuerzas gravitacional y electroestática. En cada caso

$$\vec{V} = C\frac{\hat{r}}{r^2} = C\frac{\vec{r}}{r^3}\ , \tag{1.90}$$

donde C es una constante y $\hat{r}$ es un vector unitario hacia afuera en la dirección radial. Para el caso gravitacional tenemos $C \equiv Gm_1m_2$, dado por la ley de Newton de la gravitación universal. Si $C = q_1q_2$, tenemos la ley de Coulomb de la electroestática (unidades cgs). La fuerza $\vec{V}$ dada en la ecuación (1.90) puede ser mostrada como irrotacional por expansión directa en las componentes cartesianas, como lo hicimos en el ejemplo. Otro acercamiento será desarrollado en el próximo capítulo, en el cual expresamos $\vec{\nabla}\times$, el rotor, en término de coordenadas polares esféricas. En la sección 1.13 veremos que cada vez que un vector es irrotacional, el vector puede ser escrito como la gradiente (negativa) de un potencial escalar. En la sección 1.15 probaremos que un campo vectorial puede ser resuelto en una parte irrotacional y una parte solenoidal (sujetos a condiciones en infinito). En términos del campo electromagnético esto corresponde a la resolución dentro de un campo eléctrico irrotacional y un campo magnético solenoidal.

[13] $V_y(x_0+dx,y_0) = V_y(x_0,y_0) + \left(\frac{\partial V_y}{\partial x}\right)_{x_0,y_0}dx + \cdots$ Los términos de órdenes más altos se van a cero en el límite $dx\to 0$. Un término de corrección por la variación de V_y con y es cancelado por el correspondiente término en la cuarta integral.

[14] En dinámica de fluido $\vec{\nabla}\times\vec{V}$ es llamada la vorticidad.

Para ondas en un medio elástico, si el desplazamiento $\vec{u}$ es irrotacional, $\vec{\nabla} \times \vec{u} = 0$, como ondas planas (u ondas esféricas en distancias grandes) estas llegan a ser longitudinales. Si $\vec{u}$ es solenoidal, $\vec{\nabla} \cdot \vec{u} = 0$, luego las ondas llegan a ser transversales. Una perturbación sísmica producirá un desplazamiento que puede ser resuelto en una parte solenoidal y una parte irrotacional (compare la sección 1.15). La parte irrotacional produce la longitudinal P (primaria) de las ondas de terremotos. La parte solenoidal da origen a las ondas transversales más lentas S (secundarias).

Usando la gradiente, divergencia y rotor y por su puesto la regla $BAC - CAB$, podemos construir o verificar un gran número de útiles identidades vectoriales. Para la verificación, una expansión completa en componentes cartesianas es siempre una posibilidad. Algunas veces si usamos agudeza de ingenio en vez de la pesada rutina de las componentes cartesianas, el proceso de verificación puede ser drásticamente acortado.

Recuerde que $\vec{\nabla}$ es un operador vectorial, un objeto híbrido que satisface dos conjuntos de reglas:

1. reglas vectoriales, y

2. reglas de diferenciación parcial incluyendo la de diferenciación de un producto.

Ejemplo Verifiquemos que

$$\vec{\nabla}(\vec{A} \cdot \vec{B}) = (\vec{B} \cdot \vec{\nabla})\vec{A} + (\vec{A} \cdot \vec{\nabla})\vec{B} + \vec{B} \times (\vec{\nabla} \times \vec{A}) + \vec{A} \times (\vec{\nabla} \times \vec{B}) \ . \tag{1.91}$$

En este particular ejemplo el factor más importante es reconocer que $\vec{\nabla}(\vec{A} \cdot \vec{B})$ es el tipo de términos que aparecen en la expansión $BAC-CAB$ de un producto triple vectorial, ecuación (1.64). Por ejemplo,

$$\vec{A} \times (\vec{\nabla} \times \vec{B}) = \vec{\nabla}(\vec{A} \cdot \vec{B}) - (\vec{A} \cdot \vec{\nabla})\vec{B} \ ,$$

con $\vec{\nabla}$ diferenciando sólo a $\vec{B}$, no a $\vec{A}$. De la conmutatividad de los factores en el producto escalar nosotros podemos intercambiar $\vec{A}$ con $\vec{B}$ y escribimos

$$\vec{B} \times (\vec{\nabla} \times \vec{A}) = \vec{\nabla}(\vec{A} \cdot \vec{B}) - (\vec{B} \cdot \vec{\nabla})\vec{A} \ ,$$

ahora con $\vec{\nabla}$ diferenciando sólo a $\vec{A}$, no a $\vec{B}$. Sumando estas ecuaciones, obtenemos $\vec{\nabla}$ diferenciando al producto $\vec{A} \cdot \vec{B}$ y la identidad (1.91).

Esta identidad es usada frecuentemente en teoría electromagnética avanzada.

1.9. Aplicaciones sucesivas de $\vec{\nabla}$.

Hemos definido gradiente, divergencia y rotor para obtener un vector, un escalar y una cantidad vectorial, respectivamente. Usando el operador $\vec{\nabla}$ sobre cada una de estas cantidades, obtenemos

$$\text{(a) } \vec{\nabla} \cdot \vec{\nabla}\varphi \qquad \text{(b) } \vec{\nabla} \times \vec{\nabla}\varphi \qquad \text{(c)} \vec{\nabla}\vec{\nabla} \cdot \vec{V}$$
$$\text{(d) } \vec{\nabla} \cdot \vec{\nabla} \times \vec{V} \qquad \text{(e) } \vec{\nabla} \times (\vec{\nabla} \times \vec{V}) \ ,$$

las cinco expresiones involucran una segunda derivada y las cinco aparecen en las ecuaciones diferenciales de segundo orden de la Física Matemática, particularmente en la teoría electromagnética.

La primera expresión, $\vec{\nabla}\cdot\vec{\nabla}\varphi$, la divergencia del gradiente, es llamada el Laplaciano de φ. Tenemos

$$\begin{aligned}\vec{\nabla}\cdot\vec{\nabla}\varphi &= \left(\hat{x}\frac{\partial}{\partial x}+\hat{y}\frac{\partial}{\partial y}+\hat{z}\frac{\partial}{\partial z}\right)\cdot\left(\hat{x}\frac{\partial\varphi}{\partial x}+\hat{y}\frac{\partial\varphi}{\partial y}+\hat{z}\frac{\partial\varphi}{\partial z}\right)\\ &= \frac{\partial^2\varphi}{\partial x^2}+\frac{\partial^2\varphi}{\partial y^2}+\frac{\partial^2\varphi}{\partial z^2}\ .\end{aligned} \tag{1.92}$$

Cuando φ es el potencial electroestático, tenemos

$$\vec{\nabla}\cdot\vec{\nabla}\varphi=\nabla^2\varphi=0\ . \tag{1.93}$$

la cual es la ecuación de Laplace de la electroestática. A menudo la combinación $\vec{\nabla}\cdot\vec{\nabla}$ es escrita como $\vec{\nabla}^2$.

la expresión (b) puede ser escrita

$$\vec{\nabla}\times\vec{\nabla}\varphi=\begin{vmatrix}\hat{x} & \hat{y} & \hat{z}\\ \dfrac{\partial}{\partial x} & \dfrac{\partial}{\partial y} & \dfrac{\partial}{\partial z}\\ \dfrac{\partial\varphi}{\partial x} & \dfrac{\partial\varphi}{\partial y} & \dfrac{\partial\varphi}{\partial z}\end{vmatrix}\ .$$

Expandiendo el determinante, obtenemos

$$\vec{\nabla}\times\vec{\nabla}\varphi=\hat{x}\left(\frac{\partial^2\varphi}{\partial y\partial z}-\frac{\partial^2\varphi}{\partial z\partial y}\right)+\hat{y}\left(\frac{\partial^2\varphi}{\partial z\partial x}-\frac{\partial^2\varphi}{\partial x\partial z}\right)+\hat{z}\left(\frac{\partial^2\varphi}{\partial x\partial y}-\frac{\partial^2\varphi}{\partial y\partial x}\right)=0\ , \tag{1.94}$$

suponiendo que el orden de las derivadas parciales puede ser intercambiado. Esto es cierto ya que estas segundas derivadas parciales de φ son funciones continuas.

Luego, de la ecuación (1.94), el rotor de un gradiente es idénticamente nulo. Todos los gradientes, por lo tanto, son irrotacionales. Note cuidadosamente que el cero en la ecuación (1.94) se vuelve una identidad matemática, independiente de cualquier Física. El cero en la ecuación (1.93) es una consecuencia de la Física.

La expresión (d) es un producto escalar triple el cual puede ser escrito

$$\vec{\nabla}\cdot\vec{\nabla}\times\vec{V}=\begin{vmatrix}\dfrac{\partial}{\partial x} & \dfrac{\partial}{\partial y} & \dfrac{\partial}{\partial z}\\ \dfrac{\partial}{\partial x} & \dfrac{\partial}{\partial y} & \dfrac{\partial}{\partial z}\\ V_x & V_y & V_z\end{vmatrix}\ . \tag{1.95}$$

De nuevo, suponiendo continuidad tal que el orden de la diferenciación es irrelevante, obtenemos

$$\vec{\nabla}\cdot\vec{\nabla}\times\vec{V}=0\ . \tag{1.96}$$

La divergencia de un rotor se anula siempre o todos los rotores son solenoidales. En la sección 1.15 veremos que los vectores pueden ser resueltos dentro de una parte solenoidal y otra irrotacional por el teorema de Helmholtz.

Las dos expresiones remanentes satisfacen una relación

$$\vec{\boldsymbol{\nabla}} \times (\vec{\boldsymbol{\nabla}} \times \vec{V}) = \vec{\boldsymbol{\nabla}}\vec{\boldsymbol{\nabla}} \cdot \vec{V} - \vec{\boldsymbol{\nabla}} \cdot \vec{\boldsymbol{\nabla}}\vec{V} \ . \tag{1.97}$$

Esto se deduce inmediatamente a partir de la ecuación (1.64), la regla $BAC - CAB$, la cual reescribimos tal que C aparece en el extremo derecho de cada término. El término $\vec{\boldsymbol{\nabla}} \cdot \vec{\boldsymbol{\nabla}}\vec{V}$ no fue incluido en nuestra lista, pero puede ser definido por la ecuación (1.97).

Ejemplo Una importante aplicación de estas relaciones vectoriales es la derivación de la ecuación de ondas electromagnéticas. En el vacío las ecuaciones de Maxwell llegan a ser

$$\vec{\boldsymbol{\nabla}} \cdot \vec{B} = 0 \ , \tag{1.98}$$

$$\vec{\boldsymbol{\nabla}} \cdot \vec{E} = 0 \ , \tag{1.99}$$

$$\vec{\boldsymbol{\nabla}} \times \vec{B} = \ \frac{1}{c}\frac{\partial \vec{E}}{\partial t} \ , \tag{1.100}$$

$$\vec{\boldsymbol{\nabla}} \times \vec{E} = -\frac{1}{c}\frac{\partial \vec{B}}{\partial t} \ , \tag{1.101}$$

Aquí $\vec{E}$ es el campo eléctrico, $\vec{B}$ es el campo magnético c la velocidad de la luz en el vacío (unidades cgs). Supongamos que eliminamos $\vec{B}$ de las ecuaciones (1.100) y (1.101). Podemos hacer esto tomando rotor a ambos lados de (1.101) y la derivada temporal a ambos lados de (1.100). Ya que las derivadas espaciales y temporales conmutan

$$\frac{\partial}{\partial t}\vec{\boldsymbol{\nabla}} \times \vec{B} = \vec{\boldsymbol{\nabla}} \times \frac{\partial \vec{B}}{\partial t} \ , \tag{1.102}$$

y obtenemos

$$\vec{\boldsymbol{\nabla}} \times (\vec{\boldsymbol{\nabla}} \times \vec{E}) = -\frac{1}{c^2}\frac{\partial^2 \vec{E}}{\partial t^2} \ . \tag{1.103}$$

Aplicando las ecuaciones (1.97) y (1.99) se produce

$$\nabla^2 \vec{E} = \frac{1}{c^2}\frac{\partial^2 \vec{E}}{\partial t^2} \ , \tag{1.104}$$

que es la ecuación de onda para el vector electromagnético. De nuevo, si $\vec{E}$ es expresado en coordenadas cartesianas, la ecuación (1.104) se separa en tres ecuaciones escalares.

1.10. Integración vectorial.

El siguiente paso después de la derivación vectorial es la integración. Comencemos con las integrales de línea y luego procederemos con las integrales de superficie y de volumen. En cada caso el método será reducir la integral vectorial a una o varias integrales escalares con las cuales supuestamente estamos más familiarizados.

1.10.1. Integrales lineales.

Usando un incremento de longitud $d\vec{s} = \hat{x}dx + \hat{y}dy + \hat{z}dz$, podemos encontrar las integrales de línea

$$\int_c \varphi \, d\vec{s} \, , \tag{1.105}$$

$$\int_c \vec{V} \cdot d\vec{s} \, , \tag{1.106}$$

$$\int_c \vec{V} \times d\vec{s} \, , \tag{1.107}$$

en cada una de las cuales la integral está sobre algún contorno C que puede ser abierto (con un punto inicial y un punto final separados) o cerrado (formando un *loop*). Dada la interpretación física que sigue, la segunda forma, ecuación (1.106), es lejos la más importante de las tres.

Con φ, un escalar, la primera integral se reduce a

$$\int_c \varphi \, d\vec{s} = \hat{x} \int_c \varphi(x,y,z)\, dx + \hat{y} \int_c \varphi(x,y,z)\, dy + \hat{z} \int_c \varphi(x,y,z)\, dz \, . \tag{1.108}$$

Esta separación ha empleado la relación

$$\int \hat{x}\, \varphi \, dx = \hat{x} \int \varphi \, dx \, , \tag{1.109}$$

la cual es permisible porque los vectores unitarios cartesianos $\hat{x}$, $\hat{y}$, $\hat{z}$ son constantes en magnitud y dirección. Quizás esta relación es obvia aquí, pero no será verdadera en los sistemas no cartesianos encontrados en el siguiente capítulo.

Las tres integrales sobre el lado derecho de la ecuación (1.108) son integrales escalares ordinarias. Note, sin embargo, que la integral con respecto a x no puede ser evaluada a menos que y y z sean conocidos en términos de x y similarmente para las integrales con respecto a y y z. Esto simplemente significa que el camino de integración C debe ser especificado. A menos que el integrando tenga propiedades especiales que lleve a la integral a depender solamente de los valores de los puntos extremos, el valor dependerá de la elección particular del contorno de C. Por ejemplo, si escogemos el caso muy especial de $\varphi = 1$, la ecuación (1.108) es sólo la distancia vectorial desde el comienzo del contorno C al punto final, en este caso es independiente de la elección del camino que conecta los puntos extremos. Con $d\vec{s} = \hat{x}dx + \hat{y}dy + \hat{z}dz$, las segunda y tercera formas también se reducen a una integral escalar y, como la ecuación (1.105), son dependientes, en general, de la elección del camino. La forma (ecuación (1.106)) es exactamente la misma, como la encontrada cuando calculamos el trabajo hecho por una fuerza que varía a lo largo del camino,

$$\begin{aligned} W &= \int \vec{F} \cdot d\vec{s} \\ &= \int F_x(x,y,z)dx + \int F_y(x,y,z)dy + \int F_z(x,y,z)dz \, . \end{aligned} \tag{1.110}$$

En esta expresión $\vec{F}$ es la fuerza ejercida sobre una partícula.

1.10.2. Integrales de superficie.

Las integrales de superficie aparecen en la misma forma que las integrales de línea, el elemento de área también es un vector, $d\vec{a}$. A menudo este elemento de área es escrito $\hat{n}da$ en el cual $\hat{n}$ es un vector unitario (normal) para indicar la dirección positiva. Hay dos convenciones para escoger la dirección positiva. Primero, si la superficie es una superficie cerrada, concordamos en tomar hacia afuera como positivo. Segundo, si la superficie es abierta, la normal positiva depende de la dirección en la cual el perímetro de la superficie abierta es recorrido. Si los dedos de la mano derecha están colocados en la dirección de viaje alrededor del perímetro, la normal positiva está indicada por el pulgar de la mano derecha. Como una ilustración, un círculo en el plano xy (figura 1.23) recorrido de x a y a, $-x$ a $-y$ y de regreso a x tendrá su normal positiva paralela al eje z positivo (por el sistema de coordenadas de la mano derecha). Si encontraran superficies de un lado, tal como las bandas de Moebius, se sugiere que se corten las bandas y formen superficies bien comportadas o las etiqueten de patológicas y las envíen al departamento de Matemáticas más cercano.

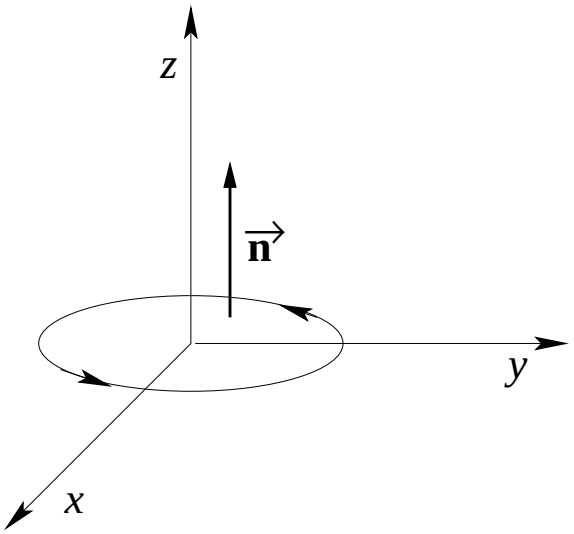


Figura 1.20: Regla de la mano derecha para la normal positiva.

Análogo a las integrales de línea, ecuaciones (1.105-1.107), las integrales de superficie pueden aparecer en las formas

$$\int \varphi \, d\vec{a} \, , \qquad \int \vec{V} \cdot d\vec{a} \, , \qquad \int \vec{V} \times d\vec{a} \, .$$

Nuevamente, el producto punto es lejos la forma más comúnmente encontrada.

La integral de superficie $\int \vec{V} \cdot d\vec{a}$ puede ser interpretada como un flujo a través de la superficie dada. Esto es realmente lo que hacemos en la sección 1.7 para obtener el significado del término divergencia. Esta identificación reaparece en la sección 1.11 como el teorema de Gauss. Note que en ambos, físicamente y desde el punto de vista del producto punto, la componente tangencial de la velocidad no contribuye en nada al flujo a través de la superficie.

1.10.3. Integrales de volumen.

Las integrales de volumen son algo muy simple, porque el elemento de volumen dv es una cantidad escalar[15]. Tenemos

$$\int_V \vec{V} dv = \hat{x} \int_V V_x \, dv + \hat{y} \int_V V_y \, dv + \hat{z} \int_V V_z \, dv \ , \tag{1.111}$$

de nuevo reduciendo la integral vectorial a una suma vectorial de integrales escalares.

1.10.4. Definiciones integrales de gradiente, divergencia y rotor.

Una interesante y significativa aplicación de nuestras integrales de volumen y de superficie es su uso en el desarrollo de definiciones alternativas de nuestras relaciones diferenciales. Encontramos

$$\vec{\boldsymbol{\nabla}} \varphi = \lim_{\int dv \to 0} \frac{\int \varphi \, d\vec{a}}{\int dv} \ , \tag{1.112}$$

$$\vec{\boldsymbol{\nabla}} \cdot \vec{V} = \lim_{\int dv \to 0} \frac{\int \vec{V} \cdot d\vec{a}}{\int dv} \ , \tag{1.113}$$

$$\vec{\boldsymbol{\nabla}} \times \vec{V} = \lim_{\int dv \to 0} \frac{\int d\vec{a} \times \vec{V}}{\int dv} \ . \tag{1.114}$$

En estas tres ecuaciones $\int dv$ es el volumen de una pequeña región del espacio y da es el elemento de área vectorial de ese volumen. La identificación de la ecuación (1.113) como la divergencia de $\vec{V}$ fue realizada en la sección 1.7. Aquí mostramos que la ecuación (1.112) es consistente con nuestra definición inicial de $\vec{\boldsymbol{\nabla}}\varphi$ (ecuación (1.68)). Por simplicidad escogemos dv como la diferencial de volumen $dxdydz$ (figura 1.21). Esta vez colocamos el origen en el centro geométrico de nuestro elemento de volumen. La integral de área tiende a seis integrales, una por cada una de las seis caras. Recordando que $d\vec{a}$ es hacia afuera, $d\vec{a}\hat{x} = -\,|\,d\vec{a}\,|$ para la superficie $EFHG$, y $+\,|\,d\vec{a}\,|$ para la superficie $ABCD$, tenemos

$$\begin{aligned}\int \varphi \, d\vec{a} = &-\hat{x} \int_{EFGH} \left(\varphi - \frac{\partial \varphi}{\partial x} \frac{dx}{2} \right) dydz + \hat{x} \int_{ABDC} \left(\varphi + \frac{\partial \varphi}{\partial x} \frac{dx}{2} \right) dydz - \\ &- \hat{y} \int_{AEGC} \left(\varphi - \frac{\partial \varphi}{\partial y} \frac{dy}{2} \right) dxdz + + \hat{y} \int_{BFHD} \left(\varphi - \frac{\partial \varphi}{\partial y} \frac{dy}{2} \right) dxdz - \\ &- \hat{z} \int_{ABFE} \left(\varphi - \frac{\partial \varphi}{\partial z} \frac{dz}{2} \right) dxdy + \hat{z} \int_{CDHG} \left(\varphi - \frac{\partial \varphi}{\partial z} \frac{dz}{2} \right) dxdy \ .\end{aligned}$$

Usando los dos primeros términos de la expansión de Maclaurin, evaluamos cada integrando en el origen con una corrección incluida para corregir por el desplazamiento ($\pm dx/2$, etc.) del centro de la cara desde el origen. Habiendo escogido el volumen total como el tamaño diferencial ($\int dv = dxdydz$), obtenemos

$$\int \varphi \, d\vec{a} = \left(\hat{x} \frac{\partial \varphi}{\partial x} + \hat{y} \frac{\partial \varphi}{\partial y} + \hat{z} \frac{\partial \varphi}{\partial z} \right) dxdydz \ . \tag{1.115}$$

[15] Frecuentemente se denota d^3r o d^3x al elemento de volumen.

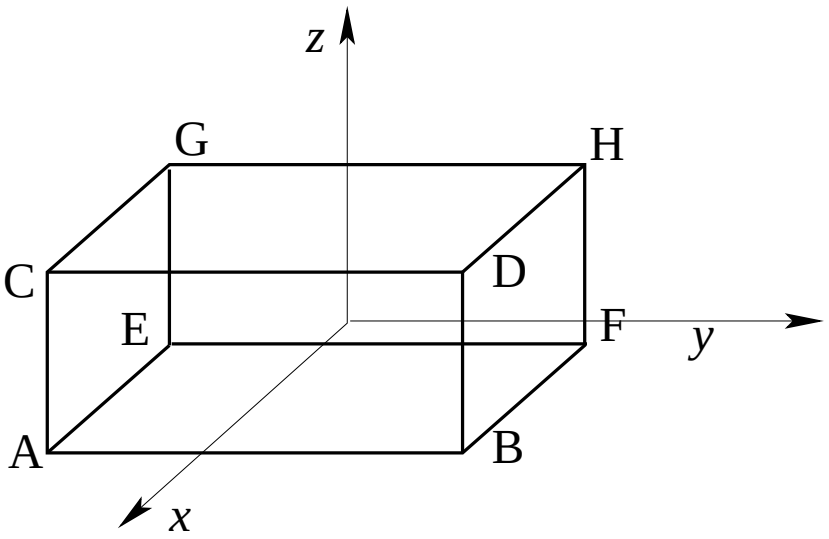


Figura 1.21: Paralelepípedo rectangular diferencial con el origen en el centro.

Dividiendo por

$$\int dv = dxdydz \ ,$$

verificamos la ecuación (1.112).

Esta verificación ha sido sobre simplificada al ignorar otros términos más allá de las primeras derivadas. Estos términos adicionales, los cuales son introducidos más adelante cuando la expansión de Taylor es desarrollada, se anulan en el límite

$$\int dv \to 0 \quad (dx \to 0 \ , \ dy \to 0 \ , \ dz \to 0) \ .$$

Esto, por supuesto, son las razones especificadas en las ecuaciones (1.112), (1.113) y (1.114), en las cuales este límite es tomado.

La verificación de la ecuación (1.114) sigue esta misma línea exactamente, usando un volumen diferencial $dxdydz$.

1.11. Teorema de Gauss.

Aquí derivamos una útil relación entre una integral de superficie de un vector y la integral de volumen de la divergencia de ese vector. Supongamos que el vector $\vec{V}$ y su primera derivada son continuas en la región de interés. Luego el teorema de Gauss establece que

$$\int_S \vec{V} \cdot d\vec{a} = \int_V \vec{\boldsymbol{\nabla}} \cdot \vec{V} \, dv \ . \tag{1.116}$$

En otras palabras, la integral de superficie de un vector sobre una superficie cerrada es igual a la integral de volumen de la divergencia del vector integrada sobre el volumen encerrado definido por la superficie.

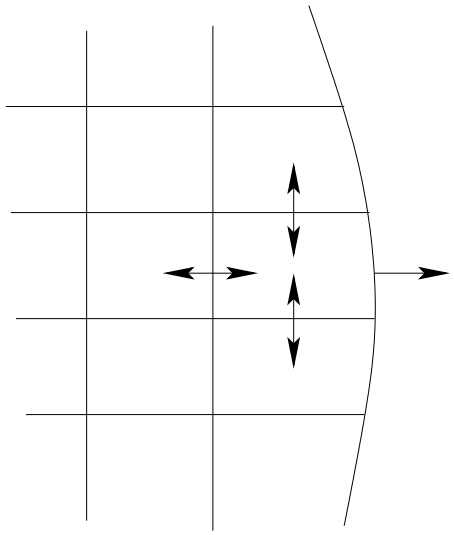


Figura 1.22: Cancelación exacta de los $d\vec{a}$ sobre las superficies interiores. Sobre la superficie externa no hay cancelación.

Imaginemos que el volumen V lo subdividimos en un gran número de pequeños paralelepípedos (diferenciales). Para cada uno de los paralelepípedos

$$\sum_{\text{seis superficies}} \vec{V} \cdot d\vec{a} = \vec{\boldsymbol{\nabla}} \cdot \vec{V}\, dv \ , \tag{1.117}$$

del análisis de la sección 1.7, la ecuación (1.76), con $\rho\vec{v}$ reemplazado por $\vec{V}$. La suma es sobre las seis caras del paralelepípedo. Sumando sobre todos los paralelepípedos, encontramos que el término $\vec{V} \cdot d\vec{a}$ se cancela para todas las caras internas; solamente las contribuciones de las superficies externas sobreviven (figura 1.22). Análoga a la definición de una integral de Riemann como el límite de una suma, tomamos el límite sobre el número de paralelepípedos tendiendo a infinito ($\to \infty$) y las dimensiones de cada uno tendiendo a cero ($\to 0$).

$$\begin{array}{ccc} \displaystyle\sum_{\text{superficies externas}} \vec{V} \cdot d\vec{a} = & & \displaystyle\sum_{\text{volúmenes}} \vec{\boldsymbol{\nabla}} \cdot \vec{V}\, dv \\ \downarrow & & \downarrow \\ \displaystyle\int_S \vec{V} \cdot d\vec{a} = & & \displaystyle\int_V \vec{\boldsymbol{\nabla}} \cdot \vec{V}\, dv \end{array}$$

El resultado es la ecuación (1.116), el teorema de Gauss.

A partir de un punto de vista físico la ecuación (1.76) ha establecido $\vec{\boldsymbol{\nabla}} \cdot \vec{V}$ como el flujo neto de fluido por unidad de volumen. Luego la integral de volumen da el flujo neto total. Pero la integral de superficie $\int \vec{V} \cdot d\vec{a}$ es sólo otra manera de expresar esta misma cantidad, igualando ambas obtenemos el teorema de Gauss.

1.11.1. Teorema de Green.

Un corolario útil del teorema de Gauss es una relación conocida como teorema de Green. Si u y v son dos funciones escalares, tenemos las identidades

$$\vec{\boldsymbol{\nabla}} \cdot (u\vec{\boldsymbol{\nabla}} v) = u\vec{\boldsymbol{\nabla}} \cdot \vec{\boldsymbol{\nabla}} v + (\vec{\boldsymbol{\nabla}} u) \cdot (\vec{\boldsymbol{\nabla}} v) \ , \tag{1.118}$$

$$\vec{\boldsymbol{\nabla}} \cdot (v\vec{\boldsymbol{\nabla}} u) = v\vec{\boldsymbol{\nabla}} \cdot \vec{\boldsymbol{\nabla}} u + (\vec{\boldsymbol{\nabla}} v) \cdot (\vec{\boldsymbol{\nabla}} u) \ . \tag{1.119}$$

Sustrayendo la ecuación (1.119) de la (1.118) e integrando sobre un volumen (suponiendo las derivadas de u y v continuas), y aplicando la ecuación (1.116) (teorema de Gauss), obtenemos

$$\int_V (u\vec{\boldsymbol{\nabla}} \cdot \vec{\boldsymbol{\nabla}} v - v\vec{\boldsymbol{\nabla}} \cdot \vec{\boldsymbol{\nabla}} u)\, dv = \int_S (u\vec{\boldsymbol{\nabla}} v - v\vec{\boldsymbol{\nabla}} u) \cdot d\vec{a} \ . \tag{1.120}$$

Este es el teorema de Green. Una forma alternativa del teorema de Green derivada de la ecuación (1.118) es

$$\int_S u\vec{\boldsymbol{\nabla}} v \cdot d\vec{a} = \int_V u\vec{\boldsymbol{\nabla}} \cdot \vec{\boldsymbol{\nabla}} v\, dv + \int_V \vec{\boldsymbol{\nabla}} u \cdot \vec{\boldsymbol{\nabla}} v\, dv \ . \tag{1.121}$$

Esta es otra forma del teorema de Green.

1.11.2. Forma alternativa del Teorema de Gauss.

Aunque la ecuación (1.116) involucra la divergencia, es con mucho la forma más usada del teorema de Gauss. Las integrales de volumen también podrían involucrar el gradiente o el rotor. Suponga

$$\vec{V}(x, y, z) = V(x, y, z)\, \vec{a}, \tag{1.122}$$

en el cual $\vec{a}$ es un vector con una magnitud constante y de dirección constante pero arbitraria. (se puede elegir la dirección, pero una vez que se escoge hay que mantenerla fija). La ecuación (1.116) se convierte en

$$\begin{aligned} \vec{a} \cdot \int_S V d\vec{a} &= \int_V \vec{\boldsymbol{\nabla}} \cdot \vec{a} V\, dv \\ &= \vec{a} \cdot \int_V \vec{\boldsymbol{\nabla}} V\, dv \end{aligned} \tag{1.123}$$

usando la ecuación (1.78). Esta puede ser reescrita

$$\vec{a} \cdot \left[\int_S V d\vec{a} - \int_V \vec{\boldsymbol{\nabla}} V\, dv \right] = 0 \ . \tag{1.124}$$

Ya que $|\,\vec{a}\,| \neq 0$ y su dirección es arbitraria, significa que el coseno del ángulo sustentado entre ambos vectores no siempre se anula, lo cual implica que el término en paréntesis debe ser nulo. El resultado es

$$\int_S V d\vec{a} = \int_V \vec{\boldsymbol{\nabla}} V\, dv \ . \tag{1.125}$$

En una manera similar, usando $\vec{V} = \vec{a} \times \vec{P}$ en el cual $\vec{a}$ es un vector constante, podemos mostrar

$$\int_S d\vec{a} \times \vec{P} = \int_V \vec{\boldsymbol{\nabla}} \times P \, dv \ . \tag{1.126}$$

Estas dos últimas formas del teorema de Gauss son usadas en la forma vectorial de la teoría de difracción de Kirchoff. También pueden ser usadas para verificar las ecuaciones (1.112) y (1.114).

1.12. El Teorema de Stokes.

El teorema de Gauss relaciona la integral de volumen de una derivada con una integral de una función sobre la superficie cerrada que rodea el volumen. Aquí consideramos una relación análoga entre la integral de superficie de una derivada de una función y la integral de línea de la función, el camino de integración es el perímetro que rodea la superficie. Tomemos la superficie y la dividimos en una red de rectángulos arbitrariamente pequeños. En la sección 1.8 mostramos que la circulación alrededor de tales rectángulos diferenciales (en el plano xy) es $\vec{\boldsymbol{\nabla}} \times \vec{V}|_2 dxdy$. De la ecuación (1.87) aplicada a un rectángulo diferencial

$$\sum \vec{V} \cdot d\vec{v} = \vec{\boldsymbol{\nabla}} \times \vec{V} \cdot d\vec{a} \ . \tag{1.127}$$

Sumamos sobre todos los pequeños rectángulos como en la definición de una integral de Riemann. Las contribuciones superficiales (lado derecho de la ecuación (1.127)) son sumadas juntas. Las integrales de línea (lado izquierdo de la ecuación (1.127)) sobre todos los segmentos internos se cancelan. Solamente la integral de línea alrededor del perímetro sobrevive (figura 1.23). Tomando el límite usual con el número de rectángulos tendiendo a infinito mientras $dx \to 0$, $dy \to 0$, tenemos

$$\begin{aligned} \sum_{\text{segmentos externos}} \vec{V} \cdot d\vec{s} &= \sum_{\text{rectángulos}} \vec{\boldsymbol{\nabla}} \times \vec{V} \cdot d\vec{a} \\ \downarrow \quad &= \quad \downarrow \\ \oint \vec{V} \cdot d\vec{s} &= \int_S \vec{\boldsymbol{\nabla}} \times \vec{V} \cdot d\vec{a} \ . \end{aligned} \tag{1.128}$$

Este es el teorema de Stokes. La integral de superficie de la derecha es sobre la superficie encerrada por el perímetro o contorno de la integral de línea de la izquierda. La dirección del vector que representa el área es hacia afuera del plano del papel si la dirección en que se recorre el contorno para la integral de línea es en el sentido matemático positivo como se muestra en la figura 1.23.

Esta demostración del teorema de Stokes está limitada por el hecho de que usamos una expansión de Maclaurin de $\vec{V}(x, y, z)$ para establecer la ecuación (1.87) en la sección 1.8. Realmente sólo necesitamos pedir que el rotor de $\vec{V}(x, y, z)$ exista y que sea integrable sobre la superficie.

El teorema de Stokes obviamente se aplica a una superficie abierta. Es posible considerar una superficie cerrada como un caso límite de una superficie abierta con la abertura (y por lo tanto el perímetro) contraída a cero.

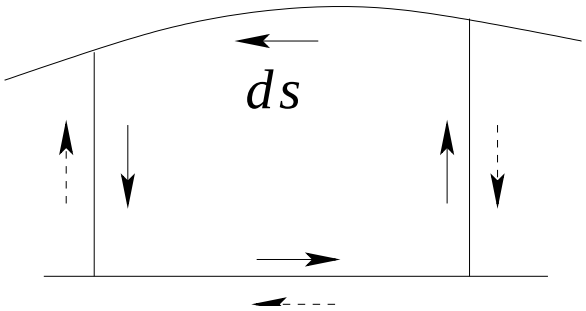


Figura 1.23: Cancelación exacta de los caminos interiores. Sobre el perímetro externo no hay cancelación.

1.12.1. Forma alternativa del Teorema de Stokes.

Como con el teorema de Gauss, otras relaciones entre superficies e integrales de línea son posibles. Encontramos

$$\int_S d\vec{a} \times \vec{\boldsymbol{\nabla}}\varphi = \oint \varphi \, d\vec{s} \tag{1.129}$$

y

$$\int_S (d\vec{a} \times \vec{\boldsymbol{\nabla}}) \times \vec{P} = \oint d\vec{s} \times \vec{P} \ . \tag{1.130}$$

La ecuación (1.129) puede ser verificada rápidamente sustituyendo $\vec{V} = \vec{a}\,\varphi$ en la cual $\vec{a}$ es un vector de magnitud constante y de dirección arbitraria, como en la sección 1.11. Sustituyendo dentro del teorema de Stokes, la ecuación (1.128)

$$\begin{aligned} \int_S (\vec{\boldsymbol{\nabla}} \times \vec{a}\varphi) \cdot d\vec{a} &= -\int_S \vec{a} \times \vec{\boldsymbol{\nabla}}\varphi \cdot d\vec{a} \\ &= -\vec{a} \cdot \int_S \vec{\boldsymbol{\nabla}}\varphi \times d\vec{a} \ . \end{aligned} \tag{1.131}$$

Para la integral de línea

$$\oint \vec{a}\varphi \cdot d\vec{s} = \vec{a} \cdot \oint \varphi \, d\vec{s} \ , \tag{1.132}$$

obtenemos

$$\vec{a} \cdot \left(\oint \vec{a}\varphi \cdot d\vec{s} + \int_S \vec{\boldsymbol{\nabla}}\varphi \times d\vec{a} \right) = 0 \ . \tag{1.133}$$

Ya que la elección de la dirección de $\vec{a}$ es arbitraria, la expresión entre paréntesis debe anularse, esto verifica la ecuación (1.129). La ecuación (1.130) puede ser derivada similarmente usando $\vec{V} = \vec{a} \times \vec{P}$, en el cual $\vec{a}$ nuevamente es un vector constante.

Ambos teoremas el de Stokes y de Gauss son de tremenda importancia en una variedad de problemas que involucran cálculo vectorial.

1.13. Teoría potencial.

1.13.1. Potencial escalar.

Si una fuerza sobre una región dada del espacio S puede ser expresada como el gradiente negativo de una función escalar φ,

$$\vec{F} = -\vec{\boldsymbol{\nabla}}\varphi \ , \tag{1.134}$$

llamamos φ a un potencial escalar el cual describe la fuerza por una función en vez de tres. Un potencial escalar está determinado salvo una constante aditiva la cual puede ser usada para ajustar su origen. La fuerza $\vec{F}$ presentada como el gradiente negativo de un potencial escalar, mono valuado, es etiquetada como una fuerza conservativa. Deseamos saber cuándo existe una función de potencial escalar. Para contestar esta pregunta establezcamos otras dos relaciones equivalentes a la ecuación (1.134). Estas son

$$\vec{\boldsymbol{\nabla}} \times \vec{F} = 0 \tag{1.135}$$

y

$$\oint \vec{F} \cdot d\vec{s} = 0 \ , \tag{1.136}$$

para todo camino cerrado en nuestra región S. Procedemos a mostrar que cada una de esas ecuaciones implica las otras dos.

Comencemos con

$$\vec{F} = -\vec{\boldsymbol{\nabla}}\varphi \ . \tag{1.137}$$

Entonces

$$\vec{\boldsymbol{\nabla}} \times \vec{F} = -\vec{\boldsymbol{\nabla}} \times \vec{\boldsymbol{\nabla}}\varphi = 0 \tag{1.138}$$

por las ecuaciones (1.94) o (1.18) implica la ecuación (1.135). Volviendo a la integral de línea, tenemos

$$\oint \vec{F} \cdot d\vec{s} = -\oint \vec{\boldsymbol{\nabla}}\varphi \cdot d\vec{s} = -\oint d\varphi \ , \tag{1.139}$$

usando la ecuación (1.71). Al integrar $d\varphi$ nos da φ. Ya que hemos especificado un *loop* cerrado, los puntos extremos coinciden y obtenemos cero para cada camino cerrado en nuestra región S, por lo cual la ecuación (1.134) se mantiene. Es importante notar aquí la restricción de que el potencial sea univaluado y que la ecuación (1.134) se mantenga para todos los puntos sobre S. Este problema puede presentarse al usar un potencial escalar magnético, un procedimiento perfectamente válido mientras el camino no rodeé corriente neta. Tan pronto como escojamos un camino que rodeé una corriente neta, el potencial magnético escalar cesa de ser mono valuado y nuestro análisis no tiene mayor aplicación.

Continuando esta demostración de equivalencia, supongamos que la ecuación (1.136) se mantiene. Si $\oint \vec{F} \cdot d\vec{s} = 0$ para todos los caminos en S, vemos que el valor de la integral cerrada para dos puntos distintos A y B es independiente del camino (figura 1.24). Nuestra premisa es que

$$\oint_{ACBDA} \vec{F} \cdot d\vec{s} = 0 \ . \tag{1.140}$$

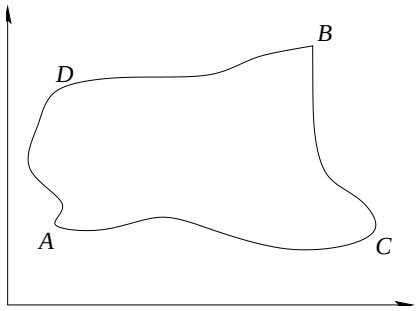


Figura 1.24: Posible camino para hacer el trabajo.

Por lo tanto

$$\int_{ACB} \vec{F} \cdot d\vec{s} = -\int_{BDA} \vec{F} \cdot d\vec{s} = \int_{ADB} \vec{F} \cdot d\vec{s} \,, \tag{1.141}$$

cambiando el signo al revertir la dirección de integración. Físicamente, esto significa que el trabajo hecho al ir de A a B es independiente del camino y que el trabajo hecho sobre un camino cerrado es cero. Esta es la razón para etiquetar como una fuerza conservativa: la energía es conservada.

Con el resultado mostrado en la ecuación (1.141), tenemos que el trabajo hecho depende solamente de los puntos extremos, A y B. Esto es,

$$\text{trabajo hecho por la fuerza} = \int_A^B \vec{F} \cdot d\vec{s} = \varphi(A) - \varphi(B) \,. \tag{1.142}$$

La ecuación (1.142) define un potencial escalar (estrictamente hablando, la diferencia de potencial entre los puntos A y B) y provee una manera de calcular el potencial. Si el punto B es tomado como una variable, digamos, (x, y, z), luego la diferenciación con respecto a x, y y z cubrirá la ecuación (1.134). La elección del signo de la parte derecha de esa ecuación es arbitraria. La hacemos para obtener concordancia con la ecuación (1.134) y asegurar que el agua correrá río abajo en vez que lo haga río arriba. Para los puntos A y B separados por una longitud $d\vec{s}$, la ecuación (1.142) se convierte

$$\begin{aligned} \vec{F} \cdot d\vec{s} &= -d\varphi \\ &= -\vec{\boldsymbol{\nabla}}\varphi \cdot d\vec{s} \,. \end{aligned} \tag{1.143}$$

Esto puede ser reescrito

$$(\vec{F} + \vec{\boldsymbol{\nabla}}\varphi) \cdot d\vec{s} = 0 \,, \tag{1.144}$$

y ya que $d\vec{s}$ es arbitrario se concluye la ecuación (1.134).

Si

$$\oint \vec{F} \cdot d\vec{s} = 0 \,, \tag{1.145}$$

podemos obtener la ecuación (1.135) usando el teorema de Stokes (ecuación (1.132)).

$$\oint \vec{F} \cdot d\vec{s} = \int \vec{\boldsymbol{\nabla}} \times \vec{F} \cdot d\vec{a} \,. \tag{1.146}$$

Si tomamos el camino de integración como el perímetro de una diferencial de área da arbitrario, el integrando en la integral de superficie debe ser nula. De ahí que la ecuación (1.136) implica la ecuación (1.135).

Finalmente, si $\vec{\boldsymbol{\nabla}} \times \vec{F} = 0$, necesitamos solamente revertir nuestra afirmación del teorema de Stokes (ecuación (1.146)) para derivar la ecuación (1.136). Luego, por las ecuaciones (1.142-1.144) la afirmación inicial $\vec{F} = -\vec{\boldsymbol{\nabla}}\varphi$ está derivada. La triple equivalencia está demostrada en la figura 1.25.

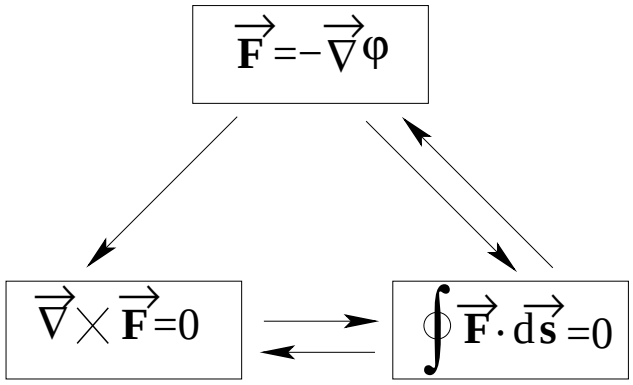


Figura 1.25: Formulaciones equivalentes para una fuerza conservativa.

Para resumir, una función potencial escalar univaluada φ existe si y sólo si $\vec{F}$ es irrotacional o el trabajo hecho en torno a *loops* cerrados es cero. Los campos de fuerza gravitacional y electroestáticos dados por la ecuación (1.91) son irrotacionales y por lo tanto, conservativos. Un potencial escalar gravitacional y electroestático existen.

1.13.2. Termodinámica, diferenciales exactas.

En termodinámica encontramos ecuaciones de la forma

$$df = P(x,y)dx + Q(x,y)dy \ . \tag{1.147}$$

El problema usual es determinar si $\int (P(x,y)dx + Q(x,y)dy)$ depende solamente de los puntos extremos, esto es, si df es realmente una diferencial exacta. La condición necesaria y suficiente es que

$$df = \frac{\partial f}{\partial x}dx + \frac{\partial f}{\partial y}dy \tag{1.148}$$

o que

$$\begin{aligned} P(x,y) &= \frac{\partial f}{\partial x} \ , \\ Q(x,y) &= \frac{\partial f}{\partial y} \ . \end{aligned} \tag{1.149}$$

Las ecuaciones (1.149) dependen de que la relación

$$\frac{\partial P(x,y)}{\partial y} = \frac{\partial Q(x,y)}{\partial x} \tag{1.150}$$

se satisfaga. Esto, sin embargo, es exactamente análogo a la ecuación (1.135), con el requerimiento de que $\vec{F}$ sea irrotacional. Por cierto, la componente z de la ecuación (1.135) es

$$\frac{\partial F_x}{\partial y} = \frac{\partial F_y}{\partial x} \ , \tag{1.151}$$

con

$$F_x = \frac{\partial f}{\partial x} \ , \quad F_y = \frac{\partial f}{\partial y} \ .$$

1.13.3. Potencial vectorial.

En algunas ramas de la Física, especialmente en la teoría electromagnética, es conveniente introducir un potencial vectorial $\vec{A}$, tal que un (fuerza) campo $\vec{B}$ está dado por

$$\vec{B} = \vec{\boldsymbol{\nabla}} \times \vec{A} \ . \tag{1.152}$$

Claramente, si la ecuación (1.152) se mantiene, $\vec{\boldsymbol{\nabla}} \cdot \vec{B} = 0$ por la ecuación (1.96) y $\vec{B}$ es solenoidal. Aquí queremos desarrollar el converso, mostrar que cuando $\vec{B}$ es solenoidal un potencial vectorial $\vec{A}$ existe. Demostramos la existencia de $\vec{A}$ calculándolo. Suponga que $\vec{B} = \hat{x}b_1 + \hat{y}b_2 + \hat{z}b_3$ y nuestra incógnita $\vec{A} = \hat{x}a_1 + \hat{y}a_2 + \hat{z}a_3$. Por la ecuación (1.152)

$$\frac{\partial a_3}{\partial y} - \frac{\partial a_2}{\partial z} = b_1 \ , \tag{1.153}$$

$$\frac{\partial a_1}{\partial z} - \frac{\partial a_3}{\partial x} = b_2 \ , \tag{1.154}$$

$$\frac{\partial a_2}{\partial x} - \frac{\partial a_1}{\partial y} = b_3 \ . \tag{1.155}$$

Supongamos que las coordenadas han sido escogidas tal que $\vec{A}$ es paralelo al plano yz; esto es, $a_1 = 0$[16]. Entonces

$$\begin{aligned} b_2 &= -\frac{\partial a_3}{\partial x} \\ b_3 &= \frac{\partial a_2}{\partial x} \ . \end{aligned} \tag{1.156}$$

Integrando, obtenemos

$$\begin{aligned} a_2 &= \int_{x_0}^{x} b_3 \, dx + f_2(y,z) \ , \\ a_3 &= -\int_{x_0}^{x} b_2 \, dx + f_3(y,z) \ , \end{aligned} \tag{1.157}$$

donde f_2 y f_3 son funciones arbitrarias de y y z pero no son funciones de x. Estas dos ecuaciones pueden ser comprobadas diferenciando y recobrando la ecuación (1.156). La ecuación

[16]Claramente esto puede hacerse en cualquier punto, y su justificación final será que obtendremos un potencial con dos componentes que satisfaga la ecuación 1.152.

(1.153) se convierte en

$$\begin{aligned}\frac{\partial a_3}{\partial y}-\frac{\partial a_2}{\partial z} &= -\int_{x_0}^{x}\left(\frac{\partial b_2}{\partial y}+\frac{\partial b_3}{\partial z}\right)dx+\frac{\partial f_3}{\partial y}-\frac{\partial f_2}{\partial z} \\ &= \int_{x_0}^{x}\frac{\partial b_1}{\partial x}dx+\frac{\partial f_3}{\partial y}-\frac{\partial f_2}{\partial z}\ ,\end{aligned} \tag{1.158}$$

usando $\vec{\boldsymbol{\nabla}}\cdot\vec{B}=0$. Integrando con respecto a x, tenemos

$$\frac{\partial a_3}{\partial y}-\frac{\partial a_2}{\partial z}=b_1(x,y,z)-b_1(x_0,y,z)+\frac{\partial f_3}{\partial y}-\frac{\partial f_2}{\partial z}\ . \tag{1.159}$$

Recordando que f_3 y f_2 son funciones arbitrarias de y y z, escogemos

$$\begin{aligned} f_2 &= 0\ , \\ f_3 &= \int_{y_0}^{y} b_1(x_0,y,z)dy\ ,\end{aligned} \tag{1.160}$$

tal que el lado derecho de la ecuación (1.159) se reduce a $b_1(x,y,z)$ en acuerdo con la ecuación (1.153). Con f_2 y f_3 dado por la ecuación (1.160), podemos construir $\vec{A}$.

$$\vec{A}=\hat{y}\int_{x_0}^{x}b_3(x,y,z)dx+\hat{z}\left[\int_{y_0}^{y}b_1(x_0,y,z)dy-\int_{x_0}^{x}b_2(x,y,z)dx\right]\ . \tag{1.161}$$

Esto no es muy completo. Podemos añadir una constante, ya que $\vec{B}$ es una derivada de $\vec{A}$. Y mucho más importante, podemos añadir un gradiente de una función escalar $\vec{\boldsymbol{\nabla}}\varphi$ sin afectar a $\vec{B}$. Finalmente, las funciones f_2 y f_3 no son únicas. Otras elecciones podrían haber sido hechas. En vez de ajustar $a_1=0$ para obtener las ecuaciones (1.153)-(1.155) cualquier permutación cíclica de 1, 2, 3, x, y, z, x_0, y_0, z_0 también podría funcionar.

Agregar un gradiente de una función escalar, digamos Λ, al vector potencial $\vec{A}$ no afecta a $\vec{B}$, por la ecuación (1.94), esta transformación es conocida como transformación de *gauge*.

$$\vec{A}\rightarrow\vec{A}'=\vec{A}+\vec{\boldsymbol{\nabla}}\Lambda\ . \tag{1.162}$$

1.14. Ley de Gauss y ecuación de Poisson.

Consideremos una carga eléctrica puntual q en el origen de nuestro sistema de coordenadas. Esta produce un campo eléctrico $\vec{E}$ dado por

$$\vec{E}=\frac{q}{r^2}\,\hat{r}\ ,\qquad (\text{CGS}). \tag{1.163}$$

Ahora derivemos la ley de Gauss, la cual establece que la integral de superficie en la figura 1.26 es $4\pi q$ si la superficie S incluye el origen (donde q está localizada) y cero si la superficie no incluye el origen. La superficie S es cualquier superficie cerrada; no necesariamente esférica.

Usando el teorema de Gauss, la ecuación (1.116), obtenemos

$$\int_S\frac{q\,\hat{r}\cdot d\vec{a}}{r^2}=q\int_v\vec{\boldsymbol{\nabla}}\cdot\frac{\hat{r}}{r^2}dv\ , \tag{1.164}$$

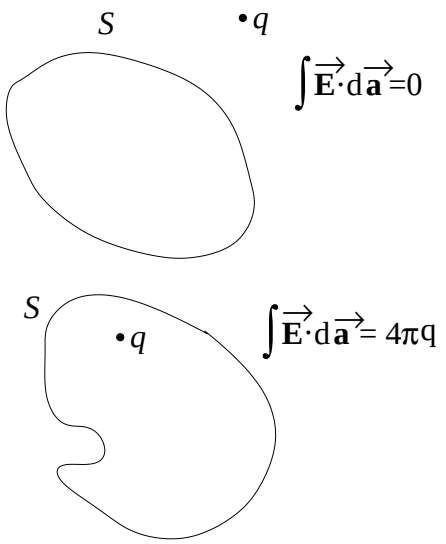


Figura 1.26: Ley de Gauss.

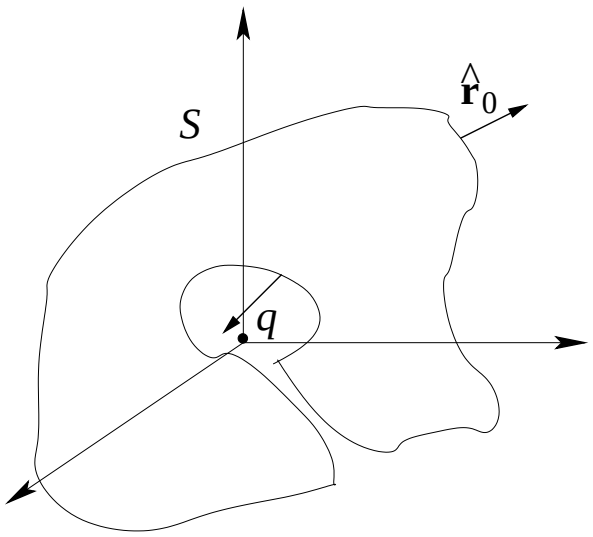


Figura 1.27: Exclusión del origen.

siempre que la superficie S no incluya el origen, donde el integrando no está bien definido. Esto prueba la segunda parte de la ley de Gauss.

La primera parte, en la cual la superficie S debería incluir el origen, puede ser tomada por los alrededores del origen con una pequeña esfera S' de radio δ (figura 1.27). De esa manera no habrá duda de que si está afuera o si está adentro, imaginemos que el volumen externo a la superficie más externa y el volumen del lado interno de la superficie $S'(r > \delta)$ se conectan por un pequeño espacio. Estas superficies unidas S y S', las combinamos en una superficie cerrada. Ya que el radio del espacio interior puede hacerse infinitamente pequeño, allí no hay contribución adicional de la integral de superficie. La superficie interna está deliberadamente escogida para que sea esférica tal que seamos capaces de integrar sobre ella. El teorema de

Gauss ahora se aplica sobre el volumen S y S' sin ninguna dificultad. Tenemos

$$\int_S \frac{q\,\hat{r}\cdot d\vec{a}}{r^2} + \int_{S'} \frac{q\,\hat{r}\cdot d\vec{a'}}{\delta^2} = 0 \ , \tag{1.165}$$

Podemos evaluar la segunda integral, para $d\vec{a}' = -\hat{r}\delta^2 d\Omega$, donde $d\Omega$ es el diferencial de ángulo sólido. El signo menos aparece porque concordamos en la sección 1.10 para tener la normal positiva $\hat{r}'$ fuera del volumen. En este caso $\hat{r}'$ externo está en la dirección radial, $\hat{r}' = -\hat{r}$. Integrando sobre todos los ángulos, tenemos

$$\int_{S'} \frac{q\,\hat{r}\cdot d\vec{a'}}{\delta^2} = -\int_{S'} \frac{q\,\hat{r}\cdot\hat{r}\delta^2 d\Omega}{\delta^2} = -4\pi q \ , \tag{1.166}$$

independiente del radio δ. Luego tenemos

$$\int_s \vec{E}\cdot d\vec{\sigma} = 4\pi q \ , \tag{1.167}$$

completando la prueba de la ley de Gauss. Note cuidadosamente que aunque la superficie S puede ser esférica, no necesariamente es esférica.

Avanzando un poco más, consideremos una carga distribuida tal que

$$q = \int_V \rho\, dv \ . \tag{1.168}$$

La ecuación (1.167) todavía se aplica, con q interpretada como la distribución de carga total encerrada por la superficie S.

$$\int_s \vec{E}\cdot d\vec{a} = 4\pi \int_V \rho\, dv \ . \tag{1.169}$$

Usando el teorema de Gauss, tenemos

$$\int_V \vec{\boldsymbol{\nabla}}\cdot\vec{E} dv = \int_V 4\pi\rho\, dv \ . \tag{1.170}$$

Ya que nuestro volumen es completamente arbitrario, los integrandos deben ser iguales o

$$\vec{\boldsymbol{\nabla}}\cdot\vec{E} = 4\pi\rho \ , \tag{1.171}$$

una de las ecuaciones de Maxwell. Si revertimos el argumento, la ley de Gauss se deduce inmediatamente a partir de la ecuación de Maxwell.

1.14.1. Ecuación de Poisson.

Reemplazando $\vec{E}$ por $-\vec{\boldsymbol{\nabla}}\varphi$, la ecuación (1.171) se convierte en

$$\vec{\boldsymbol{\nabla}}\cdot\vec{\boldsymbol{\nabla}}\varphi = \nabla^2\varphi = -4\pi\rho \ , \tag{1.172}$$

la cual es la ecuación de Poisson. Para la condición $\rho = 0$ ésta se reduce a la famosa ecuación,

$$\nabla^2\varphi = 0 \ , \tag{1.173}$$

de Laplace. Encontramos frecuentemente la ecuación de Laplace en distintos sistemas de coordenadas y las funciones especiales de la Física Matemática aparecerán como sus soluciones. La ecuación de Poisson será de gran valor en el desarrollo de la teoría de las funciones de Green.

A partir de la directa comparación de la fuerza electroestática de Coulomb y la ley de Newton de la gravitación universal

$$\vec{F}_E = \frac{q_1 q_2}{r^2}\hat{r} \ , \quad \vec{F}_G = -G\frac{m_1 m_2}{r^2}\hat{r} \ .$$

Todas las teorías de potencial de esta sección se aplican igualmente bien a los potenciales gravitacionales. Por ejemplo, la ecuación gravitacional de Poisson es

$$\nabla^2\varphi = +4\pi G\rho \tag{1.174}$$

con ρ ahora una densidad de masa.

1.15. La delta de Dirac.

Del desarrollo de la ley de Gauss en la sección 1.14

$$\int \vec{\boldsymbol{\nabla}}\cdot\vec{\boldsymbol{\nabla}}\left(\frac{1}{r}\right)dv = -\int \vec{\boldsymbol{\nabla}}\cdot\left(\frac{\hat{r}}{r^2}\right)dv = \begin{cases} -4\pi \\ 0 \end{cases} \ , \tag{1.175}$$

dependiendo de si la integración incluye el origen $\vec{r} = 0$ o no. Este resultado puede ser convenientemente expresado introduciendo la delta de Dirac,

$$\nabla^2\left(\frac{1}{r}\right) = -4\pi\delta(r) = -4\pi\delta(x)\delta(y)\delta(z) \ . \tag{1.176}$$

Esta "función" delta de Dirac está definida por sus propiedades

$$\delta(x) = 0 \ , \quad x \neq 0 \tag{1.177}$$

$$\int f(x)\delta(x)dx = f(0) \ , \tag{1.178}$$

donde $f(x)$ es cualquiera función bien comportada y la integración incluye el origen. Un caso especial de la ecuación (1.178),

$$\int \delta(x)dx = 1 \ . \tag{1.179}$$

De la ecuación (1.178), $\delta(x)$ debe ser un pico infinitamente alto y delgado en $x = 0$, como en la descripción de un impulso de fuerza o la densidad de carga para una carga puntual. El problema es que tales funciones no existen en el sentido usual de una función. Sin embargo la propiedad crucial en la ecuación (1.178) puede ser desarrollada rigurosamente como el límite de una sucesión de funciones, una distribución. Por ejemplo, la función delta puede

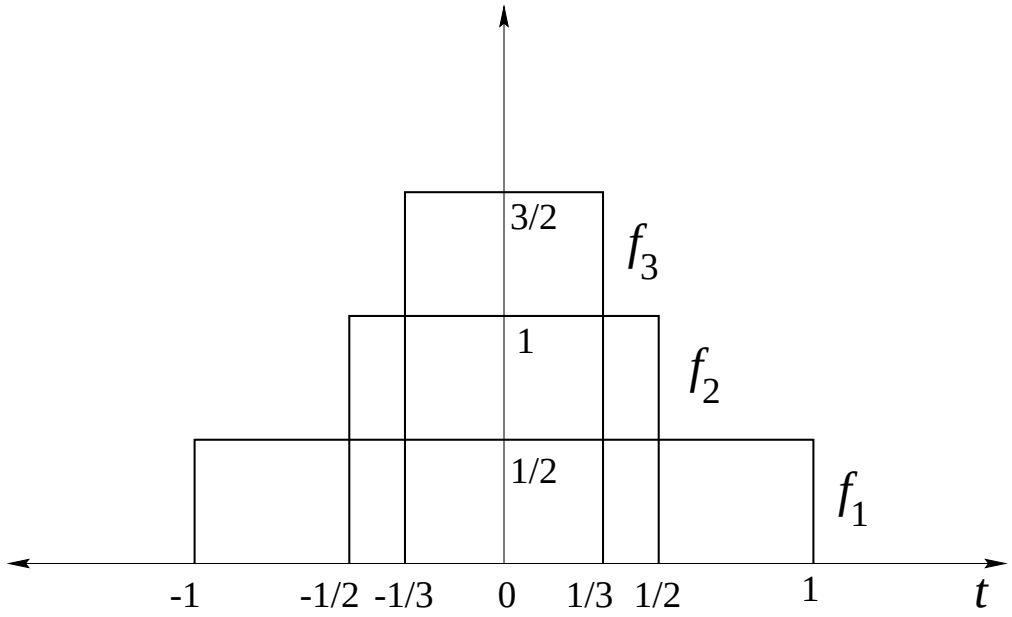


Figura 1.28: Sucesión a la delta.

ser aproximada por la sucesión de funciones, ecuaciones (1.180) a (1.182) y las figuras 1.28 a 1.30 :

$$\delta_n(x) = \begin{cases} 0\ , & \text{si } x < -1/n \\ n/2\ , & \text{si } -1/n < x < 1/n \\ 0\ , & \text{si } x > 1/n \end{cases} \tag{1.180}$$

$$\delta_n(x) = \frac{n}{\sqrt{\pi}} e^{-n^2x^2} \tag{1.181}$$

$$\delta_n(x) = \frac{\operatorname{sen}(nx)}{\pi x} = \frac{1}{2\pi}\int_{-n}^{n} e^{ixt}\,dt\ . \tag{1.182}$$

Estas aproximaciones tienen grados de utilidad variable. La ecuación (1.180) es útil en dar una derivación simple de la propiedad integral, la ecuación (1.178). La ecuación (1.181) es conveniente para diferenciar. Sus derivadas tienden a los polinomios de Hermite. La ecuación (1.182) es particularmente útil en el análisis de Fourier y en sus aplicaciones a la mecánica cuántica. En la teoría de las series de Fourier, la ecuación (1.182) a menudo aparece (modificada) como el kernel de Dirichlet:

$$\delta_n(x) = \frac{1}{2\pi}\frac{\operatorname{sen}\left[(n+\frac{1}{2})x\right]}{\frac{1}{2}x}\ . \tag{1.183}$$

Usando esta aproximación en la ecuación (1.178) y después, suponemos que $f(x)$ es bien comportada no ofrece problemas para x grandes. Para muchos propósitos físicos tales aproximaciones son muy adecuadas. Desde un punto de vista matemático la situación todavía es insatisfactoria: los límites

$$\lim_{n\to\infty} \delta_n(x)$$

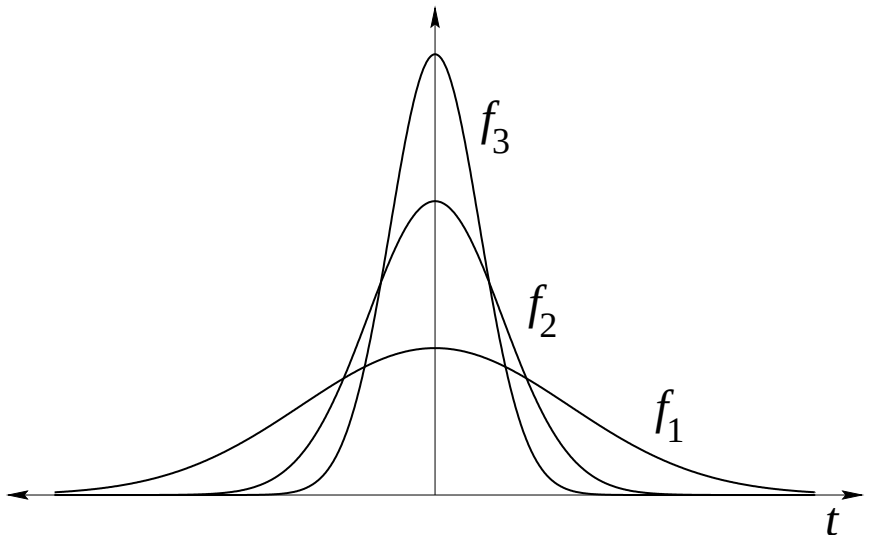


Figura 1.29: Sucesión a la delta.

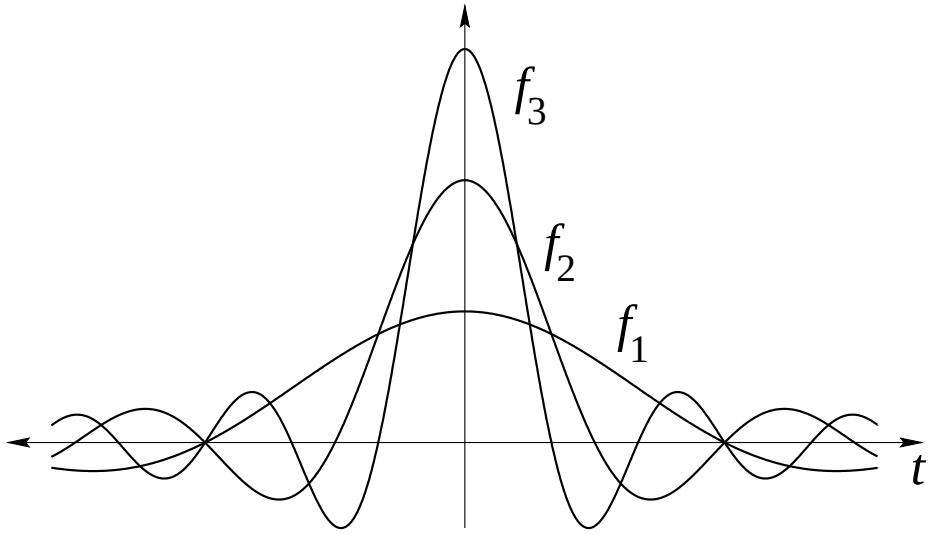


Figura 1.30: Sucesión a la delta.

no existen.

Una salida para esta dificultad está dada por la teoría de las distribuciones. Reconociendo que la ecuación (1.178) es la propiedad fundamental, enfocaremos nuestra atención sobre algo más que $\delta(x)$. Las ecuaciones (1.180)-(1.182) con $n = 1, 2, 3, \ldots$ pueden ser interpretadas como una sucesión de funciones normalizadas:

$$\int_{-\infty}^{\infty} \delta_n(x)\, dx = 1 \ . \tag{1.184}$$

La sucesión de integrales tiene el límite

$$\lim_{n\to\infty}\int_{-\infty}^{\infty}\delta_n(x)f(x)\,dx = f(0)\ . \tag{1.185}$$

Note que la ecuación (1.185) es el límite de una sucesión de integrales. Nuevamente , el límite $\delta_n(x)$, $n\to\infty$, no existe. (El límite para todas las formas de δ_n diverge en $x=0$.)

Podemos tratar $\delta(x)$ consistentemente en la forma

$$\int_{-\infty}^{\infty}\delta(x)f(x)\,dx = \lim_{n\to\infty}\int_{-\infty}^{\infty}\delta_n(x)f(x)\,dx\ . \tag{1.186}$$

$\delta(x)$ es etiquetado como una distribución (no una función) definida por la sucesión $\delta_n(x)$ como está indicado en la ecuación (1.186). Podríamos enfatizar que la integral sobre el lado izquierdo de la ecuación (1.186) no es una integral de Riemann. Es un límite.

Esta distribución $\delta(x)$ es solamente una de una infinidad de posibles distribuciones, pero es la única en que estamos interesados en la ecuación (1.178). A partir de estas sucesión de funciones vemos que la "función" delta de Dirac debe ser par en x, $\delta(-x)=\delta(x)$. La propiedad integral, la ecuación (1.178), es útil en casos donde el argumento de la función delta es una función $g(x)$ con ceros simples sobre el eje real, el cual tiende a las reglas

$$\delta(ax) = \frac{1}{a}\delta(x)\ , \quad a>0\ , \tag{1.187}$$

$$\delta(g(x)) = \sum_{a,g(a)=0,g'(a)\neq 0}\frac{\delta(x-a)}{\mid g'(a)\mid}\ . \tag{1.188}$$

Para obtener la ecuación (1.187) cambiamos la variable de integración en

$$\int_{-\infty}^{\infty}\delta(ax)f(x)\,dx = \frac{1}{a}\int_{-\infty}^{\infty}\delta(y)f(y/a)\,dy = \frac{1}{a}f(0)\ .$$

y aplicamos la ecuación (1.178). Para probar la ecuación (1.188) descomponemos la integral

$$\int_{-\infty}^{\infty}\delta(g(x))f(x)\,dx = \sum_a\int_{a-\epsilon}^{a+\epsilon}f(x)\delta(x-a)g'(a)\,dx \tag{1.189}$$

en una suma de integrales sobre pequeños intervalos que contiene los ceros de $g(x)$. En esos intervalos, $g(x)\approx g(a)+(x-a)g'(a)=(x-a)g'(a)$. Usando la ecuación (1.187) sobre el lado derecho de la ecuación (1.189) obtenemos la integral de la ecuación (1.188).

Usando integración por partes también podemos definir la derivada $\delta'(x)$ de la delta de Dirac por la relación

$$\int_{-\infty}^{\infty}\delta'(x-x_0)f(x)\,dx = -\int_{-\infty}^{\infty}f'(x)\delta(x-x_0)\,dx = -f'(x_0)\ . \tag{1.190}$$

Usamos $\delta(x)$ frecuentemente y la llamamos la delta de Dirac[17] por razones históricas. Recuerde que no es realmente una función. Es esencialmente una notación más taquigráfica,

[17]Dirac introdujo la delta para Mecánica Cuántica.

definida implícitamente como el límite de integrales en una sucesión, $\delta_n(x)$, de acuerdo a la ecuación (1.186). Podría entenderse que nuestra delta de Dirac es a menudo mirada como un operador. Un operador lineal: $\delta(x-x_0)$ opera sobre $f(x)$ y tiende a $f(x_0)$.

$$\mathcal{L}(x_0)f(x) \equiv \int_{-\infty}^{\infty} f(x)\delta(x-x_0)\,dx = f(x_0)\ . \tag{1.191}$$

También podría ser clasificada como un mapeo lineal o simplemente como una función generalizada. Desplazando nuestra singularidad al punto $x = x_0$, escribimos la delta de Dirac como $\delta(x-x_0)$. La ecuación (1.178) se convierte en

$$\int_{-\infty}^{\infty} f(x)\delta(x-x_0)\,dx = f(x_0) \tag{1.192}$$

Como una descripción de una singularidad en $x = x_0$, la delta de Dirac puede ser escrita como $\delta(x-x_0)$ o como $\delta(x_0-x)$. En tres dimensiones y usando coordenadas polares esféricas, obtenemos

$$\int_0^{2\pi}\int_0^{\pi}\int_0^{\infty} \delta(\vec{r})\,r^2 dr\,\mathrm{sen}\,\theta\,d\theta\,d\varphi = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \delta(x)\delta(y)\delta(z)\,dx\,dy\,dz = 1\ . \tag{1.193}$$

Esto corresponde a una singularidad (o fuente) en el origen. De nuevo, si la fuente está en $\vec{r} = \vec{r}_1$, la ecuación (1.193) se convierte en

$$\int_0^{2\pi}\int_0^{\pi}\int_0^{\infty} \delta(\vec{r}_2-\vec{r}_1)\,r_2^2 dr_2\ \mathrm{sen}\,\theta_2 d\theta_2\,d\varphi_2 = 1\ . \tag{1.194}$$

1.15.1. Representación de la delta por funciones ortogonales.

La función delta de Dirac puede ser expandida en términos de alguna base de funciones ortogonales reales $\varphi_n(x)$ con $n = 0, 1, 2, \ldots$. Tales funciones aparecerán más adelante como soluciones de ecuaciones diferenciales ordinarias de la forma Sturm-Liouville. Ellas satisfacen las relaciones de ortogonalidad

$$\int_a^b \varphi_m(x)\varphi_n(x)\,dx = \delta_{mn}\ , \tag{1.195}$$

donde el intervalo (a, b) puede ser infinito en uno de los lados o en ambos. [Por conveniencia suponemos que $\varphi_n(x)$ ha sido definida para incluir $(w(x))^{1/2}$ si las relaciones de ortogonalidad contienen una función de peso positivo adicional $w(x)$]. Usamos $\varphi_n(x)$ para expandir la delta como

$$\delta(x-t) = \sum_{n=1}^{\infty} a_n(t)\varphi_n(x)\ , \tag{1.196}$$

donde los coeficientes a_n son funciones de la variable t. Multiplicando por $\varphi_m(x)$ e integrando sobre el intervalo ortogonal (ecuación (1.195)), tenemos

$$a_m(t) = \int_a^b \delta(x-t)\varphi_m(x)\,dx = \varphi_m(t) \tag{1.197}$$

o

$$\delta(x-t) = \sum_{m=1}^{\infty} \varphi_m(t)\varphi_m(x) = \delta(t-x) \ . \tag{1.198}$$

Esta serie ciertamente no es uniformemente convergente, pero puede ser usada como parte de un integrando en el cual la integración resultante la hará convergente.

Suponga que forma la integral $\int F(t)\delta(t-x)dx$, donde se supone que $F(t)$ puede ser expandida en una serie de funciones ortogonales $\varphi_p(t)$, una propiedad llamada completitud. Luego obtenemos

$$\begin{aligned} \int F(t)\delta(t-x)dt &= \int \sum_{p=0}^{\infty} a_p\varphi_p(t) \sum_{n=0}^{\infty} \varphi_n(x)\varphi_n(t)\, dt \\ &= \sum_{p=0}^{\infty} a_p\varphi_p(x) = F(x) \ , \end{aligned} \tag{1.199}$$

los productos cruzado $\int \varphi_p\varphi_n dt$ con $(n \neq p)$ se anulan por ortogonalidad (ecuación (1.195)). Refiriéndonos a la anterior definición de la función delta de Dirac, ecuación (1.178), vemos que nuestra representación en serie, ecuación (1.198), satisface la propiedad de definición de la delta de Dirac que es llamada clausura. La suposición de completitud de un conjunto de funciones para la expansión de $\delta(x-t)$ produce la relación de clausura. Lo converso, que clausura implica completitud.

1.15.2. Representación integral para la delta.

Las transformaciones integrales, tal como la integral de Fourier

$$F(\omega) = \int_{-\infty}^{\infty} f(t) \exp(i\omega t)\, dt$$

conducen a representaciones integrales de la delta de Dirac. Por ejemplo,

$$\delta_n(t-x) = \frac{\operatorname{sen} n(t-x)}{\pi(t-x)} = \frac{1}{2\pi} \int_{-n}^{n} \exp(i\omega(t-x))\, d\omega \ , \tag{1.200}$$

usando la ecuación (1.182). Tenemos

$$f(x) = \lim_{n\to\infty} \int_{-\infty}^{\infty} f(t)\delta_n(t-x)\, dt \ , \tag{1.201}$$

donde $\delta_n(t-x)$ es la sucesión en la ecuación (1.200) definiendo la distribución $\delta(t-x)$. Note que la ecuación (1.201) supone que $f(t)$ es continua en $t = x$. Si sustituimos la ecuación (1.200) en la ecuación (1.201) obtenemos

$$f(x) = \lim_{n\to\infty} \frac{1}{2\pi} \int_{-\infty}^{\infty} f(t)\, dt \int_{-n}^{n} \exp(i\omega(t-x))\, d\omega \ . \tag{1.202}$$

Intercambiando el orden de integración y luego tomando el límite $n \to \infty$, tenemos el teorema de integral de Fourier.

Con el entendimiento de que pertenece bajo el signo de la integral como en la ecuación (1.201), la identificación

$$\delta(t-x) = \frac{1}{2\pi}\int_{-\infty}^{\infty} \exp(i\omega(t-x))\, d\omega \tag{1.203}$$

proporciona una muy útil representación integral de la delta.

Cuando la transformada de Laplace

$$L_\delta(s) = \int_0^\infty \exp(-st)\delta(t-t_0)\, dt = \exp(-st_0)\ , \quad t_0 > 0 \tag{1.204}$$

es invertida, obtenemos la representación compleja

$$\delta(t-t_0) = \frac{1}{2\pi i}\int_{\gamma-i\infty}^{\gamma+i\infty} \exp(s(t-t_0))\, ds\ , \tag{1.205}$$

la cual es esencialmente equivalente a la representación previa de Fourier de la delta de Dirac.

1.16. Teorema de Helmholtz.

En la sección 1.13 enfatizamos que la elección de un potencial vectorial magnético $\vec{A}$ no era único. La divergencia de $\vec{A}$ estaba aún indeterminada. En esta sección dos teoremas acerca de la divergencia y el rotor de un vector son desarrollados. El primer teorema es el siguiente:

Un vector está únicamente especificado dando su divergencia y su rotor dentro de una región y su componente normal sobre el contorno.

Tomemos

$$\begin{aligned} \vec{\boldsymbol{\nabla}} \cdot \vec{V}_1 &= s\ , \\ \vec{\boldsymbol{\nabla}} \times \vec{V}_1 &= \vec{c}\ , \end{aligned} \tag{1.206}$$

donde s puede ser interpretado como una fuente (densidad de carga) y $\vec{c}$ como una circulación (densidad de corriente). Suponiendo también que la componente normal V_{1n} sobre el límite está dada, deseamos mostrar que $\vec{V}_1$ es único. Hacemos esto suponiendo la existencia de un segundo vector $\vec{V}_2$, el cual satisface la ecuación (1.206) y tiene la misma componente normal sobre el borde, y luego mostrando que $\vec{V}_1 - \vec{V}_2 = 0$. Sea

$$\vec{W} = \vec{V}_1 - \vec{V}_2\ .$$

Luego

$$\vec{\boldsymbol{\nabla}} \cdot \vec{W} = 0 \tag{1.207}$$

y

$$\vec{\boldsymbol{\nabla}} \times \vec{W} = 0\ . \tag{1.208}$$

Ya que $\vec{W}$ es irrotacional podemos escribir (por la sección 1.13)

$$\vec{W} = -\vec{\boldsymbol{\nabla}}\varphi\ . \tag{1.209}$$

Sustituyendo esto en la ecuación (1.207), obtenemos

$$\vec{\boldsymbol{\nabla}} \cdot \vec{\boldsymbol{\nabla}}\varphi = 0, \tag{1.210}$$

la ecuación de la Laplace.

Ahora hagamos uso del teorema de Green en la forma dada en la ecuación (1.121, siendo u y v cada uno igual a φ. Ya que

$$W_n = V_{1n} - V_{2n} = 0 \tag{1.211}$$

sobre el borde, el teorema de Green se reduce a

$$\int_V (\vec{\boldsymbol{\nabla}}\varphi) \cdot (\vec{\boldsymbol{\nabla}}\varphi)\, dv = \int_V \vec{W} \cdot \vec{W}\, dv = 0 \ . \tag{1.212}$$

La cantidad $\vec{W} \cdot \vec{W} = W^2$ es no negativa y por eso tenemos

$$\vec{W} = \vec{V}_1 - \vec{V}_2 = 0 \ , \tag{1.213}$$

en todo lugar. Por lo tanto $\vec{V}_1$ es único, probando el teorema.

Para nuestro potencial vectorial magnético $\vec{A}$ la relación $\vec{B} = \vec{\boldsymbol{\nabla}} \times \vec{A}$ especifica el rotor de $\vec{A}$. A menudo por conveniencia ajustamos $\vec{\boldsymbol{\nabla}} \cdot \vec{A} = 0$. Luego (con las condiciones de borde) $\vec{A}$ está fijo.

Este teorema puede ser escrito como un teorema de unicidad para las soluciones de ecuaciones de Laplace. En esta forma, este teorema de unicidad es de gran importancia en la solución de los problemas de contorno de electroestática y otras de ecuaciones tipo Laplace con condiciones de borde. Si podemos encontrar una solución de las ecuación de Laplace que satisfaga las condiciones de borde necesarias, nuestra solución es la solución completa.

1.16.1. Teorema de Helmholtz.

El segundo teorema que probaremos es el teorema de Helmholtz.

Un vector $\vec{V}$ que satisface la ecuación (1.206) con ambas densidades de fuente y circulación que se anulan en infinito puede ser escrito como la suma de dos partes, una de la cuales es irrotacional y la otra es solenoidal.

El teorema de Helmholtz claramente será satisfecho si podemos escribir $\vec{V}$ como

$$\vec{V} = -\vec{\boldsymbol{\nabla}}\varphi + \vec{\boldsymbol{\nabla}} \times \vec{A} \ , \tag{1.214}$$

$-\vec{\boldsymbol{\nabla}}\varphi$ irrotacional y $\vec{\boldsymbol{\nabla}} \times \vec{A}$ solenoidal. Procederemos a justificar la ecuación (1.214).

Sea $\vec{V}$ un vector conocido. Le tomamos la divergencia y el rotor

$$\vec{\boldsymbol{\nabla}} \cdot \vec{V} = s(\vec{r}) \tag{1.215}$$

$$\vec{\boldsymbol{\nabla}} \times \vec{V} = \vec{c}(\vec{r}) \ . \tag{1.216}$$

con $s(\vec{r})$ y $\vec{c}(\vec{r})$ funciones conocidas de la posición. A partir de estas dos funciones construimos un potencial escalar $\varphi(\vec{r}_1)$,

$$\varphi(\vec{r}_1) = \frac{1}{4\pi} \int \frac{s(\vec{r}_2)}{r_{12}}\, dv_2 \ , \tag{1.217}$$

y un potencial vector $\vec{A}(\vec{r}_1)$,

$$\vec{A}(\vec{r}_1) = \frac{1}{4\pi}\int \frac{\vec{c}(\vec{r}_2)}{r_{12}}\, dv_2 \ . \tag{1.218}$$

Si $s = 0$, luego $\vec{V}$ es solenoidal y la ecuación (1.217) implica $\varphi = 0$. De la ecuación (1.214), $\vec{V} = \vec{\boldsymbol{\nabla}} \times \vec{A}$ con $\vec{A}$ como en la ecuación (1.161) es consistente con la sección 1.13. Además, si $\vec{c} = 0$, luego $\vec{V}$ es irrotacional y la ecuación (1.218) implica $\vec{A} = 0$, y la ecuación (1.214) implica $\vec{V} = -\vec{\boldsymbol{\nabla}}\varphi$, consistente con la teoría de potencial escalar de la sección 1.13.

Aquí, el argumento $\vec{r}_1$ indica que (x_1, y_1, z_1), es el campo puntual; $\vec{r}_2$, las coordenadas de la fuente puntual (x_2, y_2, z_2), mientras

$$r_{12} = [(x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2]^{1/2}. \tag{1.219}$$

Cuando una dirección está asociada con r_{12}, la dirección positiva se toma alejándose de la fuente. Vectorialmente, $\vec{r}_{12} = \vec{r}_1 - \vec{r}_2$, como se muestra en la figura 1.31. Por supuesto, s y $\vec{c}$ deben anularse suficientemente rápido en distancias grandes tal que existan las integrales. La expansión actual y la evaluación de las integrales tales como las ecuaciones (1.217) y (1.218) están tratadas más adelante,

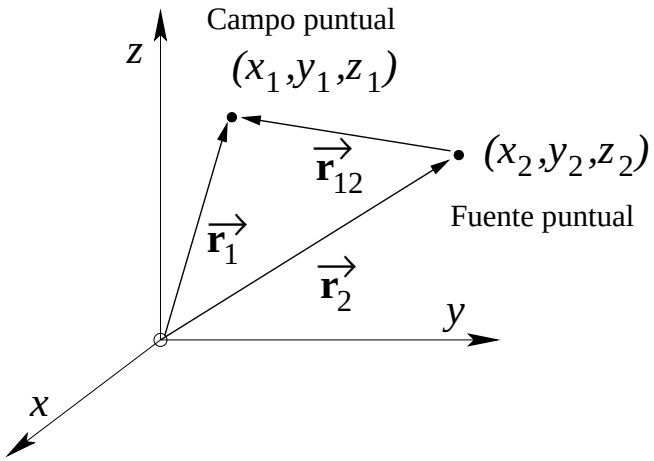


Figura 1.31: Campo y fuente puntual.

A partir del teorema de unicidad en los comienzos de esta sección, $\vec{V}$ es único al especificar su divergencia, s, y su rotor, $\vec{c}$ (y su valor sobre el contorno). Volviendo a la ecuación (1.214), tenemos

$$\vec{\boldsymbol{\nabla}} \cdot \vec{V} = -\vec{\boldsymbol{\nabla}} \cdot \vec{\boldsymbol{\nabla}}\varphi \ , \tag{1.220}$$

la divergencia del rotor se anula y

$$\vec{\boldsymbol{\nabla}} \times \vec{V} = \vec{\boldsymbol{\nabla}} \times \vec{\boldsymbol{\nabla}} \times \vec{A} \ , \tag{1.221}$$

el rotor del gradiente se anula. Si podemos mostrar que

$$-\vec{\boldsymbol{\nabla}} \cdot \vec{\boldsymbol{\nabla}}\varphi(\vec{r}_1) = s(\vec{r}_1) \tag{1.222}$$

y

$$\vec{\boldsymbol{\nabla}} \times \vec{\boldsymbol{\nabla}} \times \vec{A}(\vec{r}_1) = \vec{c}(\vec{r}_1), \tag{1.223}$$

luego $\vec{V}$ está dado en la ecuación (1.214) tendrá la divergencia y el rotor apropiado. Nuestra descripción será internamente consistente y la ecuación (1.214) estará justificada.

$$\vec{\boldsymbol{\nabla}} \cdot \vec{V} = -\vec{\boldsymbol{\nabla}} \cdot \vec{\boldsymbol{\nabla}} \varphi = -\frac{1}{4\pi} \vec{\boldsymbol{\nabla}} \cdot \vec{\boldsymbol{\nabla}} \int \frac{s(\vec{r}_2)}{r_{12}} \, dv_2 \ . \tag{1.224}$$

El operador Laplaciano, $\vec{\boldsymbol{\nabla}} \cdot \vec{\boldsymbol{\nabla}}$ ó ∇^2, opera sobre las coordenadas de campo (x_1, y_1, z_1) y por lo tanto conmuta con la integración con respecto a (x_2, y_2, z_2). Tenemos

$$\vec{\boldsymbol{\nabla}} \cdot \vec{V} = -\frac{1}{4\pi} \int s(\vec{r}_2) \nabla_1^2 \left(\frac{1}{r_{12}} \right) dv_2 \tag{1.225}$$

Debemos hacer dos modificaciones menores en la ecuación (1.175) antes de aplicarla.

Primero, nuestra fuente está en $\vec{r}_2$, no está en el origen. Esto significa que el 4π en la ley de Gauss aparece si y sólo si la superficie incluye el punto $\vec{r} = \vec{r}_2$. Para mostrar esto, reescribimos la ecuación (1.175):

$$\nabla_1^2 \left(\frac{1}{r_{12}} \right) = -4\pi \delta(\vec{r}_1 - \vec{r}_2) \ . \tag{1.226}$$

Este corrimiento de la fuente a $\vec{r}_2$ puede ser incorporado en las ecuaciones de definición (1.178) como

$$\delta(\vec{r}_1 - \vec{r}_2) = 0 \ , \qquad \vec{r}_1 \neq \vec{r}_2 \ , \tag{1.227}$$

$$\int f(\vec{r}_1) \delta(\vec{r}_1 - \vec{r}_2) \, dv_1 = f(\vec{r}_2) \ . \tag{1.228}$$

Segundo, notando que diferenciando dos veces r_{12}^{-1} con respecto a x_2, y_2, z_2 es la misma cuando diferenciamos dos veces con respecto a x_1, y_1, z_1, tenemos

$$\begin{aligned} \nabla_1^2 \left(\frac{1}{r_{12}} \right) = \nabla_2^2 \left(\frac{1}{r_{12}} \right) &= -4\pi \delta(\vec{r}_1 - \vec{r}_2) \\ &= -4\pi \delta(\vec{r}_2 - \vec{r}_1) \ . \end{aligned} \tag{1.229}$$

Reescribiendo la ecuación (1.225) y usando la delta de Dirac en la ecuación (1.229), podemos integrar para obtener

$$\begin{aligned} \vec{\boldsymbol{\nabla}} \cdot \vec{V} &= -\frac{1}{4\pi} \int s(\vec{r}_2) \nabla_1^2 \left(\frac{1}{r_{12}} \right) dv_2 \\ &= -\frac{1}{4\pi} \int s(\vec{r}_2)(-4\pi) \delta(\vec{r}_2 - \vec{r}_1) \, dv_2 \\ &= s(\vec{r}_1) \ . \end{aligned} \tag{1.230}$$

El paso final se sigue de la ecuación (1.228) con los subíndices 1 y 2 intercambiados. Nuestros resultados, ecuación (1.230), muestran que la forma supuesta de $\vec{V}$ y del potencial escalar φ están en concordancia con la divergencia dada (ecuación (1.215)). Para completar la prueba

del teorema de Helmholtz, necesitamos mostrar que nuestras suposiciones son consistentes con la ecuación (1.216), esto es, que el rotor de $\vec{V}$ es igual a $\vec{c}(\vec{r}_1)$. De la ecuación (1.214),

$$\vec{\boldsymbol{\nabla}} \times \vec{V} = \vec{\boldsymbol{\nabla}} \times \vec{\boldsymbol{\nabla}} \times \vec{A} = \vec{\boldsymbol{\nabla}}\vec{\boldsymbol{\nabla}} \cdot \vec{A} - \nabla^2 \vec{A} \, . \tag{1.231}$$

El primer término, $\vec{\boldsymbol{\nabla}}\vec{\boldsymbol{\nabla}} \cdot \vec{A}$ tiende a

$$4\pi \vec{\boldsymbol{\nabla}}\vec{\boldsymbol{\nabla}} \cdot \vec{A} = \int \vec{c}(\vec{r}_2) \cdot \boldsymbol{\nabla_1}\boldsymbol{\nabla_1} \left(\frac{1}{r_{12}} \right) dv \tag{1.232}$$

por la ecuación (1.218). Nuevamente reemplazando la segunda derivada con respecto a x_1, y_1, z_1 por segundas derivadas con respecto a x_2, y_2, z_2, integramos cada una de las componentes de la ecuación (1.141) por partes:

$$\begin{aligned} 4\pi \vec{\boldsymbol{\nabla}}\vec{\boldsymbol{\nabla}} \cdot \vec{A}\Big|_x &= \int \vec{c}(\vec{r}_2) \cdot \boldsymbol{\nabla_2} \frac{\partial}{\partial x_2} \left(\frac{1}{r_{12}} \right) dv_2 \\ &= \int \boldsymbol{\nabla_2} \cdot \left[\vec{c}(\vec{r}_2) \frac{\partial}{\partial x_2} \left(\frac{1}{r_{12}} \right) \right] dv_2 - \int [\boldsymbol{\nabla_2} \cdot \vec{c}(\vec{r}_2)] \frac{\partial}{\partial x_2} \left(\frac{1}{r_{12}} \right) dv_2 \, . \end{aligned} \tag{1.233}$$

La segunda integral se anula, ya que la densidad de circulación $\vec{c}$ es solenoidal. La primera integral puede ser transformada a una integral de superficie por el teorema de Gauss. Si $\vec{c}$ está enlazado en el espacio o se anula más rápido que $1/r$ para r grandes, de modo que la integral en la ecuación (1.218) existe, luego escogiendo una superficie suficientemente grande, la primera integral de la mano derecha de la ecuación (1.233) también se anula. Con $\vec{\boldsymbol{\nabla}}\vec{\boldsymbol{\nabla}} \cdot \vec{A} = 0$, la ecuación (1.231) se reduce a

$$\vec{\boldsymbol{\nabla}} \times \vec{V} = -\nabla^2 \vec{A} = -\frac{1}{4\pi} \int \vec{c}(\vec{r}_2) \nabla_1^2 \left(\frac{1}{r_{12}} \right) dv_2 \, . \tag{1.234}$$

Esto es exactamente como la ecuación (1.225) excepto que el escalar $s(\vec{r}_2)$ es reemplazado por la densidad de circulación vectorial $\vec{c}(\vec{r}_2)$. Introduciendo la delta de Dirac, como antes, como una manera conveniente de llevar a cabo la integración, definimos que la ecuación (1.234) se reduce a la ecuación (1.206). Vemos que nuestra forma supuesta de $\vec{V}$, dada por la ecuación (1.214), y del potencial vectorial $\vec{A}$, dado por la ecuación (1.218), están de acuerdo con la ecuación (1.206), específicamente con el rotor de $\vec{V}$.

Esto completa la prueba del teorema de Helmholtz, mostrando que un vector puede ser resuelto en una parte irrotacional y otra solenoidal. Aplicado al campo electromagnético, hemos resuelto nuestro campo vectorial $\vec{V}$ en un campo eléctrico irrotacional $\vec{E}$, derivado de un potencial escalar φ, y un campo de magnético solenoidal $\vec{B}$, derivado de un potencial vectorial $\vec{A}$. La densidad de fuente $s(\vec{r})$ puede ser interpretada como una densidad de carga eléctrica (dividida por la permitividad eléctrica ϵ), mientras que la densidad de circulación $\vec{c}(\vec{r})$ se convierte en la densidad de corriente eléctrica.

Capítulo 2

Análisis vectorial en coordenadas curvilíneas y tensores.

versión final 2.2-021008[1]

Anteriormente nos hemos restringido casi por completo a coordenadas cartesianas. Estas coordenadas ofrecen la ventaja de que los tres vectores unitarios, $\hat{x}$, $\hat{y}$, $\hat{z}$, son constantes tanto en dirección como en magnitud. Infortunadamente, no todos los problemas físicos se adaptan bien a una solución en coordenadas cartesianas. Por ejemplo, si tenemos un problema de fuerzas centrales, $\vec{F} = \hat{r}F(r)$, tal como una fuerza gravitacional o electroestática, las coordenadas cartesianas son particularmente poco apropiadas. Tales problemas llaman a usar un sistema de coordenadas en el cual la distancia radial es tomada como una de las coordenadas, es decir, coordenadas polares esféricas.

El sistema de coordenadas debe ser elegido apropiadamente para el problema, explotando cualquier restricción o simetría presente en él.

Naturalmente, hay un precio que pagar por usar un sistema no cartesiano de coordenadas. Debemos desarrollar nuevas expresiones para el gradiente, la divergencia, el rotor y el Laplaciano en estos nuevos sistemas de coordenadas. En este capítulo desarrollaremos tales expresiones de una manera muy general para luego particularizarla a los sistemas de coordenadas más usados: coordenadas circulares cilíndricas y coordenadas polares esféricas.

2.1. Coordenadas ortogonales.

En coordenadas cartesianas nosotros tratamos con tres familias de planos mutuamente perpendiculares: x = constante, y = constante, y z = constante. Imaginemos que imponemos sobre este sistema otras tres familias de superficies. Las superficies de una familia no necesariamente son paralelas unas con otras y ellas pueden no ser planos. Las tres nuevas familias de superficies no necesariamente son mutuamente perpendiculares, pero por simplicidad, rápidamente impondremos esta condición. Podemos describir cualquier punto (x, y, z) como la intersección de tres planos en coordenadas cartesianas o como la intersección de las tres superficies que forman nuestras nuevas coordenadas curvilíneas. Describiendo las superficies de las coordenadas curvilíneas por q_1 = constante, q_2 = constante, q_3 = constante, podemos identificar nuestro punto por (q_1, q_2, q_3), tanto como por (x, y, z). Esto significa que

[1]Este capítulo está basado en el segundo capítulo del libro: *Mathematical Methods for Physicists, fourth edition* de George B. Arfken & Hans J. Weber, editorial Academic Press.

en principio podemos escribir

$$\begin{array}{cc} \text{Coordenadas generales curvilíneas,} & \text{Coordenadas circulares cilíndricas} \\ q_1, q_2, q_3 & \rho, \varphi, z \\ x = x(q_1, q_2, q_3) & -\infty < x = \rho\, \cos\varphi < \infty \\ y = y(q_1, q_2, q_3) & -\infty < y = \rho\, \operatorname{sen}\varphi < \infty \\ z = z(q_1, q_2, q_3) & -\infty < z = z < \infty\ , \end{array} \tag{2.1}$$

especificando x,y,z en términos de los q_i y las relaciones inversas,

$$\begin{array}{cc} q_1 = q_1(x, y, z) & 0 \le \rho = (x^2 + y^2)^{1/2} < \infty \\ q_2 = q_2(x, y, z) & 0 \le \varphi = \arctan(y/x) < 2\pi \\ q_3 = q_3(x, y, z) & -\infty \le z = z < \infty\ . \end{array} \tag{2.2}$$

Como una ilustración específica de los abstractos (q_1, q_2, q_3) incluimos las ecuaciones de transformación para las coordenadas circulares cilíndricas. Para cada familia de superficies $q_i =$ constante, podemos asociar un vector unitario $\hat{e}_i$ normal a la superficie $q_i =$ constante y en la dirección de crecimiento de q_i. Entonces un vector $\vec{V}$ puede ser escrito

$$\vec{V} = \hat{e}_1 V_1 + \hat{e}_2 V_2 + \hat{e}_3 V_3\ . \tag{2.3}$$

Los $\hat{e}_i$ están normalizados por $\hat{e}_i^2 = 1$ y forman un sistema de coordenadas diestro con volumen $\hat{e}_1 \cdot (\hat{e}_2 \times \hat{e}_3) > 0$.

Diferenciando a x en (2.1) conduce a

$$dx = \frac{\partial x}{\partial q_1}\, dq_1 + \frac{\partial x}{\partial q_2}\, dq_2 + \frac{\partial x}{\partial q_3}\, dq_3\ , \tag{2.4}$$

de la misma manera diferenciando y y z. A partir del teorema de Pitágoras en coordenadas cartesianas, el cuadrado de la distancia entre dos puntos vecinos es

$$ds^2 = dx^2 + dy^2 + dz^2\ . \tag{2.5}$$

El cuadrado del elemento de distancia en nuestras coordenadas curvilíneas puede ser escrito

$$\begin{aligned} ds^2 =& g_{11}\, dq_1^2 + g_{12}\, dq_1 dq_2 + g_{13}\, dq_1 dq_3 + g_{21}\, dq_2 dq_1 + g_{22}\, dq_2^2 \\ & g_{23}\, dq_2 dq_3 + g_{31}\, dq_3 dq_1 + g_{32}\, dq_3 dq_2 + g_{33}\, dq_3^2 = \sum_{ij} g_{ij} dq_i dq_j\ , \end{aligned} \tag{2.6}$$

donde[2]

$$g_{ij} = \frac{\partial x}{\partial q_i}\frac{\partial x}{\partial q_j} + \frac{\partial y}{\partial q_i}\frac{\partial y}{\partial q_j} + \frac{\partial z}{\partial q_i}\frac{\partial z}{\partial q_j}\ . \tag{2.7}$$

Estos coeficientes g_{ij} pueden verse como especificando la naturaleza del sistema de coordenadas (q_1, q_2, q_3). Colectivamente estos coeficientes son referidos como la métrica. Mostraremos más adelante que forman un tensor simétrico de segundo rango. En Relatividad General los

[2]Los dq_i son arbitrarios. Por ejemplo, haciendo $dq_2 = dq_3 = 0$ aislamos g_{11}.

componentes son determinados por las propiedades de la materia. La Geometría se mezcla con la Física.

En este punto nos limitamos a sistemas de coordenadas ortogonales (*i.e.* superficies mutuamente perpendiculares)

$$g_{ij} = 0 \ , \qquad i \neq j \ , \tag{2.8}$$

y $\hat{e}_i \cdot \hat{e}_j = \delta_{ij}$. Veremos algo de sistemas no ortogonales en la sección de análisis tensorial. Para simplificar la notación escribimos $g_{ii} = h_i^2$ tal que

$$ds^2 = (h_1\, dq_1)^2 + (h_2\, dq_2)^2 + (h_3\, dq_3)^2 = \sum_i (h_i\, dq_i)^2 \ . \tag{2.9}$$

Los específicos sistemas de coordenadas ortogonales son descritos en las secciones siguientes, especificando estos factores de escala h_1, h_2 y h_3. Por otra parte, los factores de escala pueden ser identificados por la relación

$$ds_i = h_i\, dq_i \ , \tag{2.10}$$

para algún dq_i dado, manteniendo los otros q_i constantes. Notemos que las tres coordenadas curvilíneas (q_1, q_2, q_3) no necesariamente son una longitud. Los factores de escala h_i pueden depender de las q_i y pueden tener dimensiones. Los productos $h_i dq_i$ deben tener dimensiones de longitud y ser positivos. El vector de desplazamiento diferencial $d\vec{s}$ puede ser escrito

$$d\vec{s} = h_1 dq_1 \hat{e}_1 + h_2 dq_2 \hat{e}_2 + h_3 dq_3 \hat{e}_3 = \sum_i h_i dq_i \hat{e}_i \ .$$

Usando las componentes curvilíneas, encontramos que la integral de línea llega a ser

$$\int \vec{V} \cdot d\vec{s} = \sum_i \int V_i h_i dq_i \ .$$

De la ecuación (2.10) podemos inmediatamente desarrollar los elementos de área y de volumen

$$da_{ij} = ds_i ds_j = h_i h_j dq_i dq_j \tag{2.11}$$

y

$$dv = ds_1 ds_2 ds_3 = h_1 h_2 h_3 dq_1 dq_2 dq_3 \ . \tag{2.12}$$

Estas expresiones coinciden con los resultados obtenidos al usar las transformaciones explícitamente y el Jacobiano.

A partir de la ecuación (2.11) un elemento de área puede ser expandido:

$$\begin{aligned} d\vec{a} &= ds_2 ds_3 \hat{e}_1 + ds_3 ds_1 \hat{e}_2 + ds_1 ds_2 \hat{e}_3 \\ &= h_2 h_3 dq_2 dq_3 \hat{e}_1 + h_3 h_1 dq_3 dq_1 \hat{e}_2 + h_1 h_2 dq_1 dq_2 \hat{e}_3 \ . \end{aligned}$$

Una integral de superficie llega a ser

$$\int \vec{V} \cdot d\vec{a} = \int V_1 h_2 h_3 dq_2 dq_3 + \int V_2 h_3 h_1 dq_3 dq_1 + \int V_3 h_1 h_2 dq_1 dq_2 \ . \tag{2.13}$$

Debemos dejar claro que el álgebra vectorial es el mismo en coordenadas curvilíneas ortogonales que en coordenadas cartesianas. Específicamente, el producto punto

$$\vec{A} \cdot \vec{B} = A_1 B_1 + A_2 B_2 + A_3 B_3 \ , \tag{2.14}$$

donde los subíndices indican componentes curvilíneas. Para el producto cruz

$$\vec{A} \times \vec{B} = \begin{vmatrix} \hat{e}_1 & \hat{e}_2 & \hat{e}_3 \\ A_1 & A_2 & A_3 \\ B_1 & B_2 & B_3 \end{vmatrix} \tag{2.15}$$

2.2. Operadores diferenciales vectoriales.

2.2.1. Gradiente.

El punto de partida para desarrollar los operadores diferenciales: gradiente, divergencia y el rotor en coordenadas curvilíneas es nuestra interpretación del gradiente como un vector que tiene la magnitud y dirección de la razón máxima de crecimiento. A partir de esta interpretación las componentes de $\vec{\boldsymbol{\nabla}}\psi(q_1, q_2, q_3)$ en la dirección normal a la familia de superficies $q_1 =$ constante es dado por

$$\hat{e}_1 \cdot \vec{\boldsymbol{\nabla}}\psi = \vec{\boldsymbol{\nabla}}\psi\big|_1 = \frac{\partial \psi}{\partial s_1} = \frac{\partial \psi}{h_1 \partial q_1} \ , \tag{2.16}$$

ya que esta es la razón de cambio de ψ para variaciones de q_1, manteniendo q_2 y q_3 fijos. La cantidad ds_1 es un diferencial de longitud en la dirección de incremento de q_1. El vector unitario $\hat{e}_1$ indica la dirección. Si repetimos este cálculo para las otras componentes q_2 y q_3 y sumamos vectorialmente obtenemos la expresión del gradiente

$$\begin{aligned} \vec{\boldsymbol{\nabla}}\psi(q_1, q_2, q_3) &= \hat{e}_1 \frac{\partial \psi}{\partial s_1} + \hat{e}_2 \frac{\partial \psi}{\partial s_2} + \hat{e}_3 \frac{\partial \psi}{\partial s_3} \\ &= \hat{e}_1 \frac{\partial \psi}{h_1 \partial q_1} + \hat{e}_2 \frac{\partial \psi}{h_2 \partial q_2} + \hat{e}_3 \frac{\partial \psi}{h_3 \partial q_3} \\ &= \sum_i \hat{e}_i \frac{1}{h_i} \frac{\partial \psi}{\partial qi} \ . \end{aligned} \tag{2.17}$$

2.2.2. Divergencia.

El operador divergencia puede obtenerse de una de sus definiciones

$$\vec{\boldsymbol{\nabla}} \cdot \vec{V}(q_1, q_2, q_3) = \lim_{\int dv \to 0} \frac{\int \vec{V} \cdot d\vec{a}}{\int dv} \ , \tag{2.18}$$

con un volumen diferencial $h_1 h_2 h_3 \, dq_1 dq_2 dq_3$ (figura 2.1). Notemos que la dirección positiva ha sido escogida tal que (q_1, q_2, q_3) o $(\hat{e}_1, \hat{e}_2, \hat{e}_3)$ formen un sistema diestro, $\hat{e}_1 \times \hat{e}_2 = \hat{e}_3$.

La integral de área sobre las dos caras $q_1 =$ constante es dada por

$$\left[V_1 h_2 h_3 + \frac{\partial}{\partial q_1}(V_1 h_2 h_3) dq_1 \right] dq_2 dq_3 - V_1 h_2 h_3 dq_2 dq_3 = \frac{\partial}{\partial q_1}(V_1 h_2 h_3) dq_1 dq_2 dq_3 \ , \tag{2.19}$$

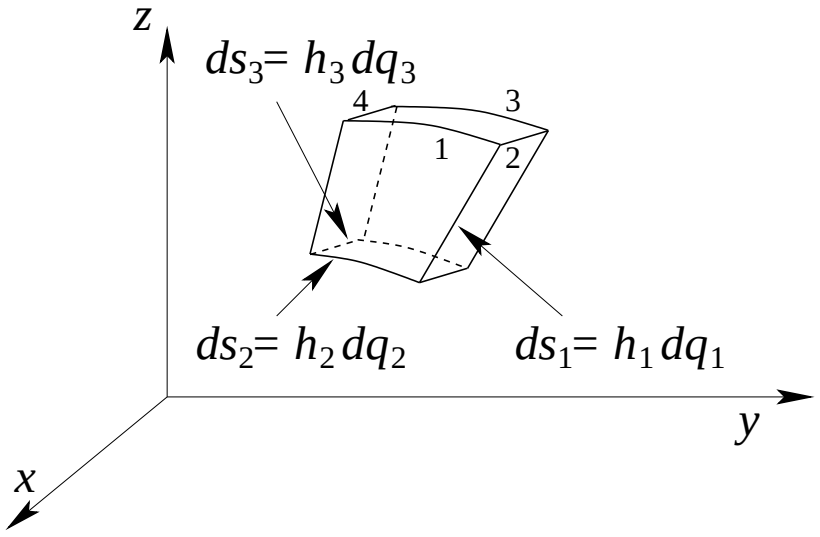


Figura 2.1: Elemento de volumen curvilíneo.

considerando las otras componentes

$$\int \vec{V}(q_1,q_2,q_3)\cdot d\vec{a} = \left[\frac{\partial}{\partial q_1}(V_1h_2h_3)+\frac{\partial}{\partial q_2}(V_2h_3h_1)+\frac{\partial}{\partial q_3}(V_3h_1h_2)\right]dq_1dq_2dq_3\ . \qquad (2.20)$$

Dividiendo por la diferencial de volumen obtenemos

$$\vec{\boldsymbol{\nabla}}\cdot\vec{V}(q_1,q_2,q_3) = \frac{1}{h_1h_2h_3}\left[\frac{\partial}{\partial q_1}(V_1h_2h_3)+\frac{\partial}{\partial q_2}(V_2h_3h_1)+\frac{\partial}{\partial q_3}(V_3h_1h_2)\right]\ . \qquad (2.21)$$

En la ecuación anterior, V_i es la componente de $\vec{V}$ en la dirección $\hat{e}_i$ en la que crece q_i. Esto es, $V_i = \hat{e}_i\cdot\vec{V}$, que es la proyección de $\vec{V}$ sobre la dirección $\hat{e}_i$.

Podemos obtener el Laplaciano combinando la ecuación (2.17) y la ecuación (2.21), y usando que $\vec{V} = \vec{\boldsymbol{\nabla}}\psi(q_1,q_2,q_3)$, obtenemos

$$\vec{\boldsymbol{\nabla}}\cdot\vec{\boldsymbol{\nabla}}\psi(q_1,q_2,q_3) = \nabla^2\psi = \frac{1}{h_1h_2h_3}\left[\frac{\partial}{\partial q_1}\left(\frac{h_2h_3}{h_1}\frac{\partial\psi}{\partial q_1}\right)+\frac{\partial}{\partial q_2}\left(\frac{h_3h_1}{h_2}\frac{\partial\psi}{\partial q_2}\right)+\frac{\partial}{\partial q_3}\left(\frac{h_1h_2}{h_3}\frac{\partial\psi}{\partial q_3}\right)\right]\ . \qquad (2.22)$$

2.2.3. Rotor.

Finalmente, para desarrollar $\vec{\boldsymbol{\nabla}}\times\vec{V}$, apliquemos el teorema de Stokes y, como con la divergencia, tomemos el límite en que el área de la superficie llega a ser despreciablemente pequeña. Trabajando sobre una componente a la vez, consideremos un diferencial de área en la superficie curvilínea q_1 = constante. A partir de

$$\int_S \vec{\boldsymbol{\nabla}}\times\vec{V}\cdot d\vec{\sigma} = \hat{e}_1\cdot(\vec{\boldsymbol{\nabla}}\times\vec{V})\,h_2h_3dq_2dq_3\ , \qquad (2.23)$$

el teorema de Stokes produce

$$\hat{e}_1 \cdot (\vec{\boldsymbol{\nabla}} \times \vec{V})\, h_2 h_3 dq_2 dq_3 = \oint \vec{V} \cdot d\vec{s}\ , \tag{2.24}$$

con la integral de línea sobre la superficie q_1 =constante. Siguiendo el *loop* (1, 2, 3, 4) en la figura 2.2,

$$\begin{aligned} \oint \vec{V}(q_1, q_2, q_3) \cdot d\vec{s} &= V_2 h_2 dq_2 + \left[V_3 h_3 + \frac{\partial}{\partial q_2}(V_3 h_3) dq_2 \right] dq_3 \\ &\quad - \left[V_2 h_2 + \frac{\partial}{\partial q_3}(V_2 h_2) dq_3 \right] dq_2 - V_3 h_3 dq_3 \\ &= \left[\frac{\partial}{\partial q_2}(V_3 h_3) - \frac{\partial}{\partial q_3}(V_2 h_2) \right] dq_2 dq_3\ . \end{aligned} \tag{2.25}$$

Tomamos el signo positivo cuando vamos en la dirección positiva sobre las partes 1 y 2 y negativa sobre las partes 3 y 4 porque aquí vamos en la dirección negativa. Los términos más altos en las expansiones de Taylor son omitidos. Ellos desaparecerán en el límite en que la superficie llega a ser despreciablemente pequeña ($dq_2 \to 0$, $dq_3 \to 0$).

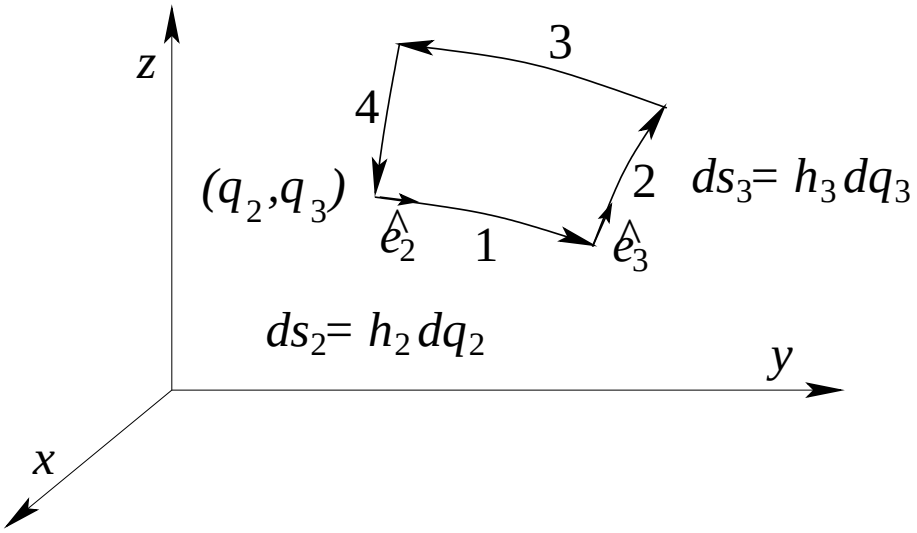


Figura 2.2: Elemento de área curvilíneo con q_1 =constante.

Desde la ecuación (2.24)

$$\vec{\boldsymbol{\nabla}} \times \vec{V}\big|_1 = \frac{1}{h_2 h_3} \left[\frac{\partial}{\partial q_2}(V_3 h_3) - \frac{\partial}{\partial q_3}(V_2 h_2) \right]\ . \tag{2.26}$$

Las dos componentes restantes pueden ser evaluadas tomando una permutación cíclica de

los índices. Finalmente, podemos escribir el rotor en forma de determinante

$$\vec{\boldsymbol{\nabla}} \times \vec{V} = \frac{1}{h_1 h_2 h_3} \begin{vmatrix} \hat{e}_1 h_1 & \hat{e}_2 h_2 & \hat{e}_3 h_3 \\ \dfrac{\partial}{\partial q_1} & \dfrac{\partial}{\partial q_2} & \dfrac{\partial}{\partial q_3} \\ h_1 V_1 & h_2 V_2 & h_3 V_3 \end{vmatrix} . \tag{2.27}$$

Notemos que esta ecuación no es idéntica a la forma del producto cruz de dos vectores, ya que $\vec{\boldsymbol{\nabla}}$ no es un vector ordinario, sino que es un vector operador.

2.3. Sistemas particulares de coordenadas.

Hay once sistemas de coordenadas en los cuales la ecuación tridimensional de Helmholtz puede separarse en tres ecuaciones diferenciales ordinarias[3]. Algunos de estos sistemas adquirieron importancia en el desarrollo histórico de la Mecánica Cuántica. Otros como el bipolar satisfacen necesidades específicas. Debido al desarrollo de computadores de alta velocidad y a la eficiencia de las técnicas de programación se ha reducido la necesidad de utilizar estos sistemas. Nos limitaremos a coordenadas cartesianas, coordenadas polares esféricas, y coordenadas circulares cilíndricas.

2.3.1. Coordenadas cartesianas rectangulares.

Este es el sistema más simple de todos:

$$\begin{aligned} h_1 &= h_x = 1 \ , \\ h_2 &= h_y = 1 \ , \\ h_3 &= h_z = 1 \ . \end{aligned} \tag{2.28}$$

Las familias de superficies coordenadas son tres conjuntos de planos paralelos: $x =$ constante, $y =$ constante, y $z =$ constante. Las coordenadas cartesianas es el único sistema en que todos los h_i son constantes. Notemos también que los vectores unitarios, $\hat{e}_1$, $\hat{e}_2$, $\hat{e}_3$ o $\hat{x}$, $\hat{y}$, $\hat{z}$ tienen direcciones fijas.

Los operadores vectoriales corresponden a

$$\vec{\boldsymbol{\nabla}}\psi = \hat{x}\frac{\partial \psi}{\partial x} + \hat{y}\frac{\partial \psi}{\partial y} + \hat{z}\frac{\partial \psi}{\partial z} \ , \tag{2.29}$$

$$\vec{\boldsymbol{\nabla}} \cdot \vec{V} = \frac{\partial V_x}{\partial x} + \frac{\partial V_y}{\partial y} + \frac{\partial V_z}{\partial z} \ , \tag{2.30}$$

$$\vec{\boldsymbol{\nabla}} \cdot \vec{\boldsymbol{\nabla}}\psi = \nabla^2 \psi = \frac{\partial^2 \psi}{\partial x^2} + \frac{\partial^2 \psi}{\partial y^2} + \frac{\partial^2 \psi}{\partial z^2} \ , \tag{2.31}$$

[3]Ver por ejemplo, Morse y Feshbach.

$$\vec{\boldsymbol{\nabla}} \times \vec{V} = \begin{vmatrix} \hat{x} & \hat{y} & \hat{z} \\ \dfrac{\partial}{\partial x} & \dfrac{\partial}{\partial y} & \dfrac{\partial}{\partial z} \\ V_x & V_y & V_z \end{vmatrix} . \tag{2.32}$$

2.4. Coordenadas circulares cilíndricas (ρ, φ, z).

En el sistema de coordenadas circulares cilíndricas las tres coordenadas curvilíneas (q_1, q_2, q_3) son reetiquetadas por (ρ, φ, z). Las superficies coordenadas mostradas en la figura 2.3 son:

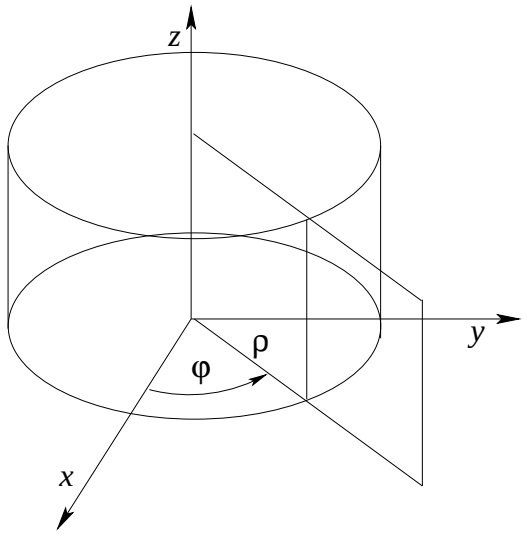


Figura 2.3: Coordenadas circulares cilíndricas.

1. Cilindros circulares derechos que tienen el eje-z como eje común,

$$\rho = \left(x^2 + y^2\right)^{1/2} = \text{constante.}$$

2. Semi planos a través del eje-z,

$$\varphi = \tan^{-1}\left(\frac{y}{x}\right) = \text{constante.}$$

3. Planos paralelos al plano xy como en el sistema cartesiano

$$z = \text{constante.}$$

Los límites para ρ, φ y z son

$$0 \leq \rho < \infty \ , \quad 0 \leq \varphi \leq 2\pi \ , \quad \text{y} \quad -\infty < z < \infty \ .$$

Notemos que estamos usando ρ como la distancia perpendicular al eje z. Las relaciones de transformación inversas

$$\begin{aligned} x &= \rho \cos\varphi \ , \\ y &= \rho \operatorname{sen}\varphi \ , \\ z &= z \ , \end{aligned} \tag{2.33}$$

de acuerdo a las ecuaciones anteriores los factores de escala son

$$\begin{aligned} h_1 &= h_\rho = 1 \ , \\ h_2 &= h_\varphi = \rho, \\ h_3 &= h_z = 1 \ . \end{aligned} \tag{2.34}$$

Los vectores unitarios $\hat{e}_1$, $\hat{e}_2$, $\hat{e}_3$ son reetiquetados $(\hat{\rho}, \hat{\varphi}, \hat{z})$, figura 2.4. El vector unitario $\hat{\rho}$ es normal a la superficie cilíndrica apuntando en la dirección de crecimiento del radio ρ. El vector unitario $\hat{\varphi}$ es tangencial a la superficie cilíndrica, perpendicular al semi plano $\varphi =$ constante y apuntando en la dirección de crecimiento del ángulo φ. El tercer vector unitario, $\hat{z}$, es el vector unitario cartesiano usual.

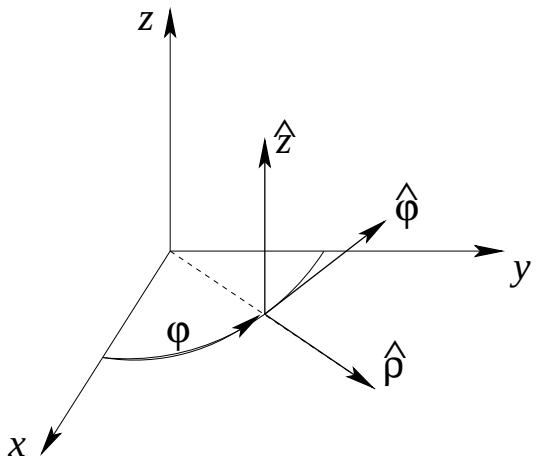


Figura 2.4: Vectores unitarios en coordenadas circulares cilíndricas.

Un vector diferencial de desplazamiento $d\vec{s}$ puede ser escrito

$$d\vec{s} = \hat{\rho} ds_\rho + \hat{\varphi} ds_\varphi + \hat{z} dz = \hat{\rho} d\rho + \hat{\varphi} \rho d\varphi + \hat{z} dz \ . \tag{2.35}$$

Los operadores diferenciales que involucran $\vec{\boldsymbol{\nabla}}$

$$\vec{\boldsymbol{\nabla}} \psi(\rho, \varphi, z) = \hat{\rho} \frac{\partial \psi}{\partial \rho} + \hat{\varphi} \frac{1}{\rho} \frac{\partial \psi}{\partial \varphi} + \hat{z} \frac{\partial \psi}{\partial z} \ , \tag{2.36}$$

$$\vec{\boldsymbol{\nabla}} \cdot \vec{V} = \frac{1}{\rho} \frac{\partial}{\partial \rho} (\rho V_\rho) + \frac{1}{\rho} \frac{\partial V_\varphi}{\partial \varphi} + \frac{\partial V_z}{\partial z} \ , \tag{2.37}$$

$$\nabla^2 \psi = \frac{1}{\rho} \frac{\partial}{\partial \rho} \left(\rho \frac{\partial \psi}{\partial \rho} \right) + \frac{1}{\rho^2} \frac{\partial^2 \psi}{\partial \varphi^2} + \frac{\partial^2 \psi}{\partial z^2} \ , \tag{2.38}$$

$$\vec{\boldsymbol{\nabla}} \times \vec{V} = \frac{1}{\rho} \begin{vmatrix} \hat{\rho} & \rho\hat{\varphi} & \hat{z} \\ \dfrac{\partial}{\partial\rho} & \dfrac{\partial}{\partial\varphi} & \dfrac{\partial}{\partial z} \\ V_\rho & \rho V_\varphi & V_z \end{vmatrix} . \tag{2.39}$$

2.5. Coordenadas polares esféricas (r, θ, φ).

Reetiquetando (q_1, q_2, q_3) como (r, θ, φ), vemos que el sistema de coordenadas polares esféricas consiste en lo siguiente:

1. Esferas concéntricas centradas en el origen,

$$r = \left(x^2 + y^2 + z^2\right)^{1/2} = \text{constante.}$$

2. Conos circulares centrados sobre el eje z y vértice en el origen

$$\theta = \arccos\left(\frac{z}{(x^2 + y^2 + z^2)^{1/2}}\right) = \text{constante.}$$

3. Semi planos a través del eje-z,

$$\varphi = \tan^{-1}\left(\frac{y}{x}\right) = \text{constante.}$$

Por nuestra elección arbitraria de la definición de θ, el ángulo polar, y φ, el ángulo azimutal, el eje z queda particularizado para un tratamiento especial. Las ecuaciones de transformación son

$$\begin{aligned} x &= r \operatorname{sen}\theta \cos\varphi \ , \\ y &= r \operatorname{sen}\theta \operatorname{sen}\varphi \ , \\ z &= r \cos\theta \ , \end{aligned} \tag{2.40}$$

midiendo θ desde el eje z positivo y φ en el plano xy desde el eje x positivo. Los intervalos de valores son $0 \leq r < \infty$, $0 \leq \theta \leq \pi$, y $0 \leq \varphi \leq 2\pi$. A partir de la ecuación (2.7)

$$\begin{aligned} h_1 &= h_r = 1 \ , \\ h_2 &= h_\theta = r, \\ h_3 &= h_\varphi = r \operatorname{sen}\theta \ . \end{aligned} \tag{2.41}$$

El elemento de arco

$$d\vec{s} = \hat{r}dr + \hat{\theta}rd\theta + \hat{\varphi}r \operatorname{sen}\theta d\varphi \ .$$

En este sistema de coordenadas esféricas el elemento de área (para r = constante) es

$$dA = da_{\theta\varphi} = r^2 \operatorname{sen}\theta d\theta d\varphi \ , \tag{2.42}$$

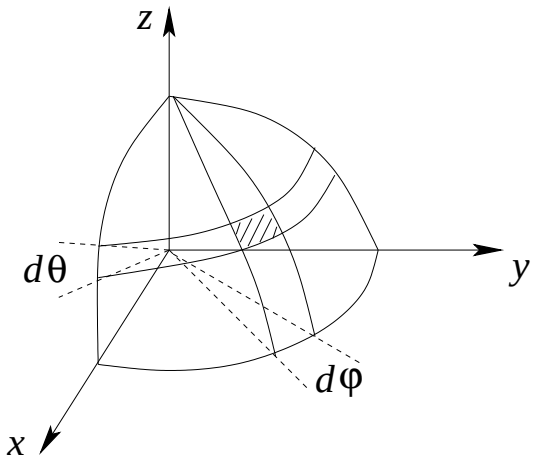


Figura 2.5: Elementos de área en coordenadas polares esféricas.

el área sombreada en la figura 2.5. Integrando sobre el azimutal φ, encontramos que el elemento de área llega a ser un anillo de ancho $d\theta$,

$$dA = 2\pi r^2 \operatorname{sen}\theta d\theta \ . \tag{2.43}$$

Esta forma aparece con frecuencia en problemas en coordenadas polares con simetría azimutal. Por definición el stereoradian, un elemento de ángulo sólido $d\Omega$, es dado por

$$d\Omega = \frac{dA}{r^2} = \operatorname{sen}\theta d\theta d\varphi \ . \tag{2.44}$$

Integrando sobre la superficie esférica completa, obtenemos

$$\int d\Omega = 4\pi \ .$$

A partir de la ecuación (2.12) el elemento de volumen es

$$dv = r^2 dr \operatorname{sen}\theta d\theta d\varphi = r^2 dr d\Omega \ . \tag{2.45}$$

Los vectores unitarios en coordenadas polares esféricas son mostrados en la figura 2.6.

Debemos enfatizar que los vectores unitarios $\hat{r}$, $\hat{\theta}$ y $\hat{\varphi}$ varían en dirección cuando los ángulos θ y φ varían. Cuando diferenciamos vectores en coordenadas polares esféricas (o en cualquier otro sistema no cartesiano) estas variaciones de los vectores unitarios con respecto a la posición no deben ser despreciadas. Escribamos los vectores unitarios $\hat{r}$, $\hat{\theta}$ y $\hat{\varphi}$ en término de los vectores unitarios cartesiano de dirección fija $\hat{x}$, $\hat{y}$, $\hat{z}$

$$\begin{aligned} \hat{r} &= \hat{x}\operatorname{sen}\theta\cos\varphi + \hat{y}\operatorname{sen}\theta\operatorname{sen}\varphi + \hat{z}\cos\theta \ , \\ \hat{\theta} &= \hat{x}\cos\theta\cos\varphi + \hat{y}\cos\theta\operatorname{sen}\varphi - \hat{z}\operatorname{sen}\theta \ , \\ \hat{\varphi} &= -\hat{x}\operatorname{sen}\varphi + \hat{y}\cos\varphi \ . \end{aligned} \tag{2.46}$$

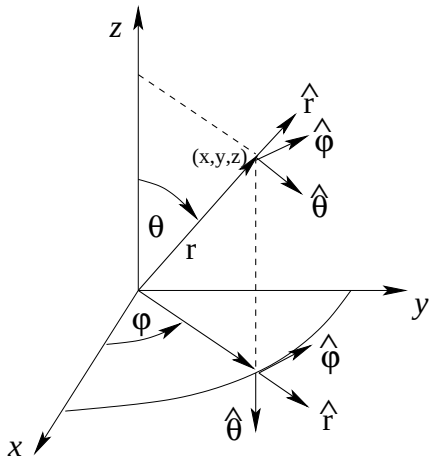


Figura 2.6: Coordenadas polares esféricas.

Notemos que un vector dado puede ser expresado en un número de diferentes (pero equivalentes) maneras. Por ejemplo, el vector posición $\vec{r}$ puede ser escrito

$$\begin{aligned}\vec{r} &= \hat{r}r \\ &= \hat{r}\left(x^2+y^2+z^2\right)^{1/2} \\ &= \hat{x}x+\hat{y}y+\hat{z}z \\ &= \hat{x}r\,\mathrm{sen}\,\theta\cos\varphi+\hat{y}r\,\mathrm{sen}\,\theta\,\mathrm{sen}\,\varphi+\hat{z}r\cos\theta\ . \end{aligned} \tag{2.47}$$

Podemos seleccionar la forma que nos es más útil para nuestro problema particular.

A partir de la sección 2.2 reetiquetando los vectores unitarios en coordenadas curvilíneas $\hat{e}_1$, $\hat{e}_2$, $\hat{e}_3$ como $\hat{r}$, $\hat{\theta}$, y $\hat{\varphi}$ tenemos

$$\vec{\boldsymbol{\nabla}}\psi(\rho,\varphi,z)=\hat{r}\frac{\partial\psi}{\partial r}+\hat{\theta}\frac{1}{r}\frac{\partial\psi}{\partial\theta}+\hat{\varphi}\frac{1}{r\,\mathrm{sen}\,\theta}\frac{\partial\psi}{\partial\varphi}\ , \tag{2.48}$$

$$\vec{\boldsymbol{\nabla}}\cdot\vec{V}=\frac{1}{r^2\,\mathrm{sen}\,\theta}\left[\mathrm{sen}\,\theta\frac{\partial}{\partial r}(r^2V_r)+r\frac{\partial}{\partial\theta}(\mathrm{sen}\,\theta V_\theta)+r\frac{\partial V_\varphi}{\partial\varphi}\right]\ , \tag{2.49}$$

$$\nabla^2\psi=\frac{1}{r^2\,\mathrm{sen}\,\theta}\left[\mathrm{sen}\,\theta\frac{\partial}{\partial r}\left(r^2\frac{\partial\psi}{\partial r}\right)+\frac{\partial}{\partial\theta}\left(\mathrm{sen}\,\theta\frac{\partial\psi}{\partial\theta}\right)+\frac{1}{\mathrm{sen}\,\theta}\frac{\partial^2\psi}{\partial\varphi^2}\right]\ , \tag{2.50}$$

$$\vec{\boldsymbol{\nabla}}\times\vec{V}=\frac{1}{r^2\,\mathrm{sen}\,\theta}\begin{vmatrix}\hat{r} & r\hat{\theta} & r\,\mathrm{sen}\,\theta\hat{\varphi}\\ \dfrac{\partial}{\partial r} & \dfrac{\partial}{\partial\theta} & \dfrac{\partial}{\partial\varphi}\\ V_r & rV_\theta & r\,\mathrm{sen}\,\theta V_\varphi\end{vmatrix}\ . \tag{2.51}$$

2.6. Análisis tensorial.

2.6.1. Introducción y definiciones.

Los tensores son importantes en muchas áreas de la Física, incluyendo relatividad y electrodinámica. Escalares y vectores son casos especiales de tensores. Como ya vimos, una cantidad que no cambia bajo rotaciones del sistema de coordenadas en un espacio tridimensional, un invariante, fue etiquetado como un escalar. Un escalar es especificado por un número real y es un *tensor de rango cero*. Una cantidad cuyas componentes transforman bajo rotaciones como las de la distancia de un punto a un origen elegido, fue llamado un vector. La transformación de las componentes del vector bajo una rotación de las coordenadas preserva el vector como una entidad geométrica (tal como una flecha en el espacio), independiente de la orientación del sistema de referencia. En un espacio tridimensional, un vector es especificado por $3 = 3^1$ números reales, por ejemplo, sus componentes, y es *un tensor de rango uno*. En un espacio N dimensional un tensor de rango n tiene N^n componentes, las cuales transforman de una manera definida.

Hay una posible ambigüedad en la ley de transformación de un vector

$$A'_i = \sum_j a_{ij} A_j \ , \tag{2.52}$$

en la cual a_{ij} es el coseno del ángulo entre el eje x'_i y el eje x_j.

Si partimos con un vector diferencial de distancia $d\vec{s}$, entonces, tomando dx'_i como una función de las variables sin primas,

$$dx'_i = \sum_j \frac{\partial x'_i}{\partial x_j} dx_j \ , \tag{2.53}$$

tomando la diferencial. Si fijamos

$$a_{ij} = \frac{\partial x'_i}{\partial x_j} \ , \tag{2.54}$$

las ecuaciones (2.52) y (2.53) son consistentes. Cualquier conjunto de cantidades A_j que transforman de acuerdo a

$$A'_i = \sum_j \frac{\partial x'_i}{\partial x_j} A_j \ , \tag{2.55}$$

es definido como un vector contravariante.

Sin embargo, hemos ya encontrado un tipo de transformación vectorial un poco diferente. El gradiente de un escalar ,$\vec{\boldsymbol{\nabla}}\varphi$, definido por

$$\vec{\boldsymbol{\nabla}}\varphi = \hat{x}\frac{\partial \varphi}{\partial x_1} + \hat{y}\frac{\partial \varphi}{\partial x_2} + \hat{z}\frac{\partial \varphi}{\partial x_3} \ , \tag{2.56}$$

usando (x_1, x_2, x_3) para (x, y, z), transforma como

$$\frac{\partial \varphi'}{\partial x'_i} = \sum_j \frac{\partial \varphi}{\partial x_j}\frac{\partial x_j}{\partial x'_i} \ , \tag{2.57}$$

usando $\varphi = \varphi(x, y, z) = \varphi(x', y', z') = \varphi'$, φ definida como una cantidad escalar. Notemos que esta difiere de la ecuación (2.55) en que tenemos $\partial x_j/\partial x'_i$ en vez de $\partial x'_i/\partial x_j$. Tomemos la ecuación (2.57) como la definición de un vector covariante y al gradiente como el prototipo.

En coordenadas cartesianas

$$\frac{\partial x_j}{\partial x'_i} = \frac{\partial x'_i}{\partial x_j} = a_{ij} \ , \tag{2.58}$$

y no hay diferencia entre transformaciones covariantes y contravariantes. En otros sistemas la ecuación (2.58) en general no se aplica, y la distinción entre covariante y contravariante es real y debe ser observada. Esto es de primera importancia en espacios curvos de Riemann, en relatividad general.

En lo que resta de esta sección las componentes de un vector contravariante son denotados por superíndices, A^i, mientras un subíndice es usado para las componentes de un vector covariante A_i.[4]

2.6.2. Definición de tensores de rango dos.

Ahora procedemos a definir *tensores de rango dos contravariantes, mixtos y covariantes* por las siguientes ecuaciones para sus componentes bajo transformación de coordenadas:

$$\begin{aligned} A'^{ij} &= \sum_{kl} \frac{\partial x'_i}{\partial x_k} \frac{\partial x'_j}{\partial x_l} A^{kl} \ , \\ B'^{i}_{\ j} &= \sum_{kl} \frac{\partial x'_i}{\partial x_k} \frac{\partial x_l}{\partial x'_j} B^k_l \ , \\ C'_{ij} &= \sum_{kl} \frac{\partial x_k}{\partial x'_i} \frac{\partial x_l}{\partial x'_j} C_{kl} \ . \end{aligned} \tag{2.59}$$

Claramente, el rango va como el número de derivadas parciales (o cosenos directores) en la definición: cero para un escalar, uno para un vector y dos para un tensor de segundo rango, y así sucesivamente. Cada índice (subíndice o superíndice) corre sobre el número de dimensiones del espacio. El número de índices (rango del tensor) es independiente de las dimensiones del espacio. Vemos que A^{kl} es contravariante con respecto a ambos índices, C_{kl} es covariante respecto a ambos índices, y B^k_l transforma contravariantemente con respecto al primero de los índices (k) y covariantemente con respecto al segundo índice (l). Si usamos coordenadas cartesianas, las tres formas de tensores de segundo rango, contravariante, mixto y covariante son la misma.

Como con las componentes de un vector, las leyes de transformación para las componentes de un tensor, ecuación (2.59), produce entidades (y propiedades) que son independientes de la elección de un sistema de referencia. Esto es lo que hace el análisis tensorial importante en Física. La independencia del sistema de referencia (invariancia) es ideal para expresar afirmaciones en Física.

[4]Esto significa que las coordenadas (x, y, z) deben ser escritas (x^1, x^2, x^3) ya que $\vec{r}$ transforma como un vector contravariante. Ya que nos restringiremos rápidamente a tensores cartesianos (donde la distinción entre contravariante y covariante desaparece) continuaremos usando subíndices para las coordenadas. Esto evita la ambigüedad de x^2 representa el cuadrado de x o y.

El tensor de segundo rango $\mathbf{A}$ (de componentes A^{kl}) puede ser convenientemente escrito por sus componentes en un arreglo cuadrado (de 3×3 si estamos en un espacio tridimensional),

$$\mathbf{A} = \begin{pmatrix} A^{11} & A^{12} & A^{13} \\ A^{21} & A^{22} & A^{23} \\ A^{31} & A^{32} & A^{33} \end{pmatrix} \ . \tag{2.60}$$

Esto no significa que cualquier arreglo de números o funciones formen un tensor. La condición esencial es que las componentes transformen de acuerdo a la ecuación (2.59).

2.6.3. Suma y resta de tensores.

La suma y resta de tensores está definida en términos de los elementos individuales de la misma manera que para vectores. Para sumar o restar dos tensores, debemos sumar o restar sus correspondientes elementos. Si

$$\mathbf{A} + \mathbf{B} = \mathbf{C} \ , \tag{2.61}$$

entonces

$$A^{ij} + B^{ij} = C^{ij} \ .$$

Naturalmente, $\mathbf{A}$ y $\mathbf{B}$ deben ser tensores del mismo rango y ambos expresados en un espacio del mismo número de dimensiones.

2.6.4. Convención de suma.

En análisis tensorial es costumbre adoptar una convención de suma para poner la ecuación (2.59) y las subsecuentes ecuaciones tensoriales en forma más compacta. Siempre y cuando distingamos entre covariante y contravariante, acordemos que cuando un índice aparezca a un lado de una ecuación, una vez como superíndice y una vez como subíndice (excepto para las coordenadas donde ambos son subíndices), automáticamente sumaremos sobre aquel índice. Entonces podemos escribir la segunda expresión en la ecuación (2.59) como

$$B'^{i}_{\ j} = \frac{\partial x'_i}{\partial x_k} \frac{\partial x_l}{\partial x'_j} B^k_l \ , \tag{2.62}$$

con las sumas sobre k y l implícitas en el lado derecho. Esta es la convención de suma.

Para ilustrar el uso de la convención de suma y algunas técnicas del análisis tensorial, mostremos que la familiar delta de Kronecker, δ_{kl}, es realmente un tensor mixto de rango dos, δ^k_l.[5] La pregunta es: ¿La δ^k_l transformará de acuerdo a la ecuación (2.59)? Este es nuestro criterio para llamarlo un tensor. Usando la convención de suma tenemos

$$\delta^k_l \frac{\partial x'_i}{\partial x_k} \frac{\partial x_l}{\partial x'_j} = \frac{\partial x'_i}{\partial x_k} \frac{\partial x_k}{\partial x'_j} \ , \tag{2.63}$$

[5]Es práctica común referirse a un tensor $\mathbf{A}$ especificando una componente típica A_{ij}.

por definición de la delta de Kronecker. Ahora

$$\frac{\partial x'_i}{\partial x_k}\frac{\partial x_k}{\partial x'_j} = \frac{\partial x'_i}{\partial x'_j} \ , \tag{2.64}$$

a partir de la regla de la cadena. Sin embargo, x'_i y x'_j son coordenadas independientes, y por tanto la variación de una respecto a la otra debe ser cero si ellas son diferentes y uno si ellas coinciden, esto es

$$\frac{\partial x'_i}{\partial x'_j} = \delta'^i_{\ j} \ . \tag{2.65}$$

Por tanto

$$\delta'^i_{\ j} = \frac{\partial x'_i}{\partial x_k}\frac{\partial x_l}{\partial x'_j}\delta^k_l \ ,$$

mostrando que δ^k_l son realmente las componentes de un tensor mixto de rango dos. Notemos que el resultado es independiente del número de dimensiones de nuestro espacio.

La delta de Kronecker posee además una interesante propiedad. Tiene las mismas componentes en todos los sistemas de coordenadas rotados y por lo tanto es llamada isotrópica. Tensores de primer rango (vectores) no isotrópicos existen.

2.6.5. Simetría–Antisimetría.

El orden en el cual los índices aparecen en nuestra descripción de un tensor es importante. En general, A^{mn} es independiente de A^{nm}, pero hay algunos casos de interés especial. Si, para todo m y n,

$$A^{mn} = A^{nm} \ , \tag{2.66}$$

lo llamaremos un tensor simétrico. Si, por otra parte,

$$A^{mn} = -A^{nm} \ , \tag{2.67}$$

el tensor lo llamaremos antisimétrico. Claramente, cada tensor (de segundo rango) puede ser resuelto en una parte simétrica y otra antisimétrica por la identidad

$$A^{mn} = \frac{1}{2}\left(A^{mn} + A^{nm}\right) + \frac{1}{2}\left(A^{mn} - A^{nm}\right) \ , \tag{2.68}$$

el primer término de la derecha es un tensor simétrico y el segundo, un tensor antisimétrico. Una similar resolución de las funciones en una parte simétrica y otra antisimétrica es de extrema importancia en Mecánica Cuántica.

2.6.6. Spinores.

Podríamos pensar que un sistema de escalares, vectores, tensores (de rango dos) y superiores forman un sistema matemático completo, uno que es adecuado para describir una Física independiente de nuestra elección de sistema de coordenadas. Pero el universo y la Física Matemática no es así de simple. En el reino de las partículas elementales, por ejemplo,

partículas de spin cero (mesones π, partículas α) pueden ser descritos con escalares; partículas de spin 1 (deuterones) por vectores, y partículas de spin 2 (gravitones), por tensores. Esta lista omite a las partículas más comunes: electrones, protones y neutrones, todas ellas con spin $\frac{1}{2}$. Estas partículas son propiamente descritas por spinores. Un spinor no es un escalar, vector o tensor.

2.7. Contracción y producto directo.

2.7.1. Contracción.

Cuando tratamos con vectores, formamos un producto escalar sumando el producto de las componentes correspondientes:

$$\vec{A} \cdot \vec{B} = A_i B_i \ . \tag{2.69}$$

La generalización de esta expresión en el análisis tensorial es un proceso conocido como contracción. Dos índices, uno covariante y el otro contravariante, se igualan uno con otro y luego sumamos sobre ese índice repetido. Por ejemplo, veamos la contracción del tensor mixto de segundo orden B^i_j.

$$B'^i_{\ j} \rightarrow B'^i_{\ i} = \frac{\partial x'_i}{\partial x_k}\frac{\partial x_l}{\partial x'_i} B^k_l = \frac{\partial x_l}{\partial x_k} B^k_l \ , \tag{2.70}$$

por la ecuación (2.64) y luego por la ecuación (2.65)

$$B'^i_{\ i} = \delta^l_k B^k_l = B^k_k \ . \tag{2.71}$$

Nuestro tensor contraído es invariante y por lo tanto un escalar. Esto es exactamente lo que obtuvimos anteriormente para el producto punto de dos vectores y para la divergencia de un vector. En general, la operación de contracción reduce el orden de un tensor en 2.

2.7.2. Producto Directo.

Las componentes de un vector covariante (tensor de rango uno) a_i y aquellos de un vector contravariante (tensor de rango uno) b^j puede ser multiplicado componente a componente para dar el término general $a_i b^j$. Esto, por la ecuación (2.59) es realmente un tensor de segundo orden, por

$$a'_i b'^j = \frac{\partial x_k}{\partial x'_i} a_k \frac{\partial x'_j}{\partial x_l} b^l = \frac{\partial x_k}{\partial x'_i}\frac{\partial x'_j}{\partial x_l}(a_k b^l) \ . \tag{2.72}$$

Contrayendo, obtenemos

$$a'_i b'^i = a_k b^k \ , \tag{2.73}$$

como en las ecuaciones (2.70) y (2.71) para dar el producto escalar regular.

La operación de asociar dos vectores a_i y b^j, como en el último párrafo, es conocido como el producto directo. Para el caso de dos vectores, el producto directo es un tensor de segundo rango. En ese sentido podemos atribuir significado a $\vec{\boldsymbol{\nabla}}\vec{E}$, el cual no estaba definido dentro

del esquema del análisis vectorial. En general, el producto directo de dos tensores es un tensor de rango igual a la suma de los dos rangos iniciales; esto es,

$$A_j^i B^{kl} = C_j^{i\,k\,l} \ , \tag{2.74}$$

donde $C_j^{i\,k\,l}$ es un tensor de cuarto rango. A partir de la ecuación (2.59)

$$C'^{\,i\,k\,l}_{\ j} = \frac{\partial x'_i}{\partial x_m}\frac{\partial x_n}{\partial x'_j}\frac{\partial x'_k}{\partial x_p}\frac{\partial x'_l}{\partial x_q} C_n^{m\,p\,q} \ . \tag{2.75}$$

El producto directo aparece en Física Matemática como una técnica para crear nuevos tensores de mayor rango.

Cuando $\mathbf{T}$ es un tensor cartesiano de n-ésimo rango, $\partial/\partial x_i T_{jkl}\ldots$, un elemento de $\vec{\boldsymbol{\nabla}}\mathbf{T}$, es un tensor *cartesiano* de rango $n+1$. Sin embargo, $\partial/\partial x_i T_{jkl}\ldots$ no es un tensor bajo transformaciones más generales. En sistemas no cartesianos $\partial/\partial x'_i$ actuará sobre las derivadas parciales $\partial x_p/\partial x'_q$ y destruirá la relación simple de transformación tensorial.

Hasta aquí la distinción entre una transformación covariante y una contravariante ha sido mantenida ya que existe en espacio no cartesiano y porque es de gran importancia en relatividad general. Ahora, sin embargo, nos restringimos al tensor cartesiano. Como se hizo notar en el caso cartesiano, la distinción entre contravariancia y covariancia desaparece y todos los índices están, a partir de ahora en adelante, mostrados como subíndices.

2.7.3. Convención de la suma.

Cuando un subíndice (letra, no número) aparece dos veces sobre un lado de una ecuación, implica que se suma con respecto a este subíndice.

2.7.4. Contracción.

La contracción consiste en fijar dos índices distintos (subíndices), igualar uno a otro y luego sumarlos según la convención de suma.

2.8. Regla del cociente.

Si A_i y B_j son vectores, podemos fácilmente mostrar que A_iB_j es un tensor de segundo rango. Aquí, estamos interesados en una variedad de relaciones inversas. Consideremos tales ecuaciones como

$$K_iA_i = B \ , \tag{2.76a}$$
$$K_{ij}A_j = B_i \ , \tag{2.76b}$$
$$K_{ij}A_{jk} = B_{ik} \ , \tag{2.76c}$$
$$K_{ijkl}A_{ij} = B_{kl} \ , \tag{2.76d}$$
$$K_{ij}A_k = B_{ijk} \ . \tag{2.76e}$$

En cada una de esas expresiones $\mathbf{A}$ y $\mathbf{B}$ son tensores conocidos cuyo rango está indicado por el número de índices y el tensor $\mathbf{A}$ es arbitrario. En cada caso K es una cantidad desconocida.

Deseamos establecer las propiedades de transformación de K. La regla del cociente afirma que si la ecuación de interés se mantiene en todos los sistemas (rotados) de coordenadas cartesianas, K es un tensor del rango indicado. La importancia en Física Teórica es que la regla del cociente puede establecer la naturaleza tensorial de las cantidades. La regla del cociente (ecuación 2.76b) muestra que la matriz de inercia que aparece en la ecuación de momento angular $\vec{L} = I\vec{\omega}$, es un tensor.

Para probar la regla del cociente, consideremos la ecuación (2.76b) como un caso típico. En nuestro sistema de coordenadas prima

$$K'_{ij}A'_j = B'_i = a_{ik}B_k \ , \tag{2.77}$$

usando las propiedades de transformación vectorial de $\vec{B}$. Ya que la ecuación se mantiene en todos los sistemas de coordenadas rotados,

$$a_{ik}B_k = a_{ik}(K_{kl}A_l) \ . \tag{2.78}$$

Ahora, transformando $\vec{A}$ regresamos al sistema de coordenadas prima (compare la ecuación (2.55)), tenemos

$$K'_{ij}A'_j = a_{ik}K_{kl}a_{jl}A'_j \ . \tag{2.79}$$

Reordenando, tenemos

$$(K'_{ij} - a_{ik}a_{jl}K_{kl})A'_j = 0 \ . \tag{2.80}$$

Esto debe mantenerse para cada valor del índice i y para cada sistema de coordenadas prima. Ya que A'_j es arbitrario, concluimos

$$K'_{ij} = a_{ik}a_{jl}K_{kl} \ , \tag{2.81}$$

la cual es nuestra definición de un tensor de segundo rango.

Las otras ecuaciones pueden ser tratadas en forma similar, dando origen a otras formas de la regla del cociente. Un peligro menor debería ser tomado en cuenta, la regla del cociente no se aplica necesariamente si $\vec{B}$ es igual a cero. Las propiedades de transformación de cero son indeterminadas.

2.9. Pseudo tensores y tensores duales.

Hasta aquí nuestras transformaciones de coordenadas han sido restringidas a rotaciones puras. Ahora consideramos el efecto de reflexiones o inversiones. Si tenemos coeficientes de transformaciones $a_{ij} = -\delta_{ij}$, entonces por la ecuación (2.53)

$$x_i = -x'_i \ , \tag{2.82}$$

la cual es una inversión o transformación de paridad. Notemos que esta transformación cambia nuestro sistema de coordenadas inicialmente diestro a un sistema de coordenadas siniestro.[6] Nuestro vector prototipo $\vec{r}$ con componentes (x_1, x_2, x_3) se transforma en el vector $\vec{r}\,' = (x'_1, x'_2, x'_3) = (-x_1, -x_2, -x_3)$. Este nuevo vector $\vec{r}\,'$ tiene componentes negativas, relativas

[6]Esta es una inversión del sistema de coordenadas, los objetos en el mundo físico permanecen fijos.

al nuevo conjunto de ejes transformados. Como se muestra en la figura 2.7, invirtiendo las direcciones de los ejes de coordenadas y cambiando los signos de las componentes da $\vec{r}\,' = \vec{r}$. El vector (una flecha en el espacio) permanece exactamente como estaba antes de que la transformación se llevara a cabo. El vector posición $\vec{r}$ y todos los otros vectores cuyas componentes se comporten de esta manera (cambiando el signo con una inversión de los ejes de coordenadas) son llamados vectores polares y tienen paridad impar.

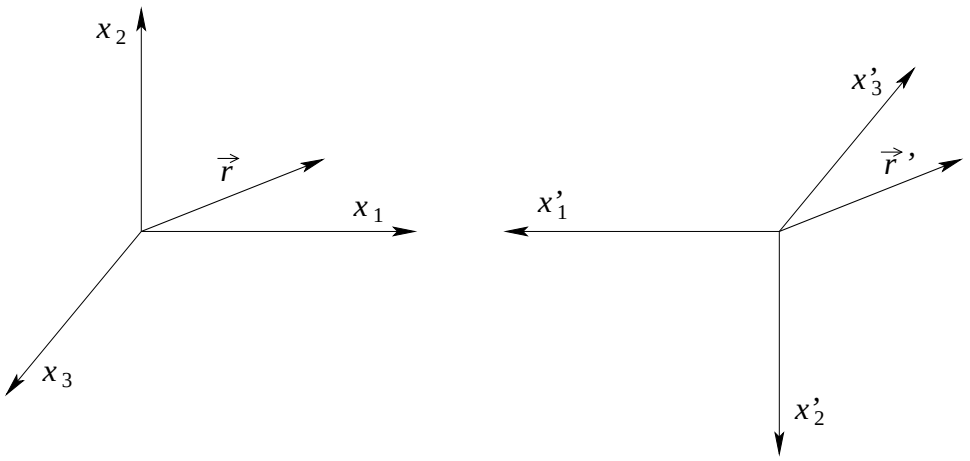


Figura 2.7: Inversión de coordenadas cartesianas, vector polar.

Una diferencia fundamental aparece cuando encontramos un vector definido como el producto cruz de dos vectores polares. Sea $\vec{C} = \vec{A} \times \vec{B}$, donde tanto $\vec{A}$ como $\vec{B}$ son vectores polares. Las componentes de $\vec{C}$ están dadas por

$$C_1 = A_2B_3 - A_3B_2 \ . \tag{2.83}$$

y así sucesivamente. Ahora cuando los ejes de coordenadas son invertidos, $A_i \to A_i'$, $B_j \to B_j'$, de su definición $C_k \to +C_k'$; esto es, nuestro vector producto cruz, vector $\vec{C}$, no se comporta como un vector polar bajo inversión. Para distinguir, lo etiquetamos como pseudo vector o vector axial (ver figura 2.8) que tiene paridad par.

Ejemplos son:

$$\begin{aligned} \text{velocidad angular,} \qquad & \vec{v} = \vec{\omega} \times \vec{r} \ , \\ \text{momento angular,} \qquad & \vec{L} = \vec{r} \times \vec{p} \ , \\ \text{torque,} \qquad & \vec{\tau} = \vec{r} \times \vec{F} \ , \\ \text{campo magnético,} \qquad & \frac{\partial \vec{B}}{\partial t} = -c\vec{\boldsymbol{\nabla}} \times \vec{E} \ . \end{aligned}$$

En $\vec{v} = \vec{\omega} \times \vec{r}$, el vector axial es la velocidad angular $\vec{\omega}$, y $\vec{r}$ y $\vec{v} = d\vec{r}/dt$ son vectores polares. Claramente, los vectores axiales ocurren frecuentemente en Física elemental, aunque este hecho usualmente no es recalcado. En un sistema de coordenadas de mano derecha un vector axial $\vec{C}$ tiene un sentido de rotación asociado con el dado por la regla de la mano derecha. En

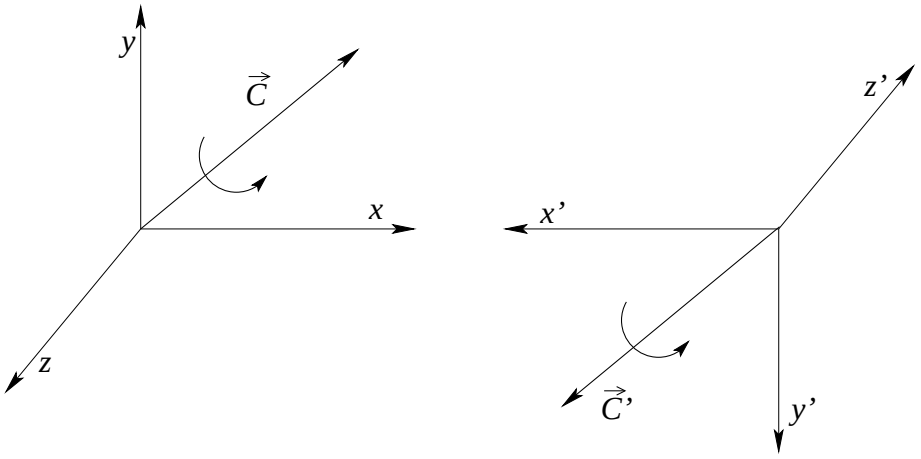


Figura 2.8: Inversión de coordenadas cartesianas, vector axial.

el sistema invertido de mano izquierda, el sentido de la rotación es una rotación de la mano izquierda. Esto es indicado por las flechas curvadas en la figura 2.8.

La distinción entre vectores polares y axiales también puede ser ilustrada por una reflexión. Un vector polar se refleja en un espejo como un flecha física real, figura 2.9a. En la figuras 2.7 y 2.8 las coordenadas están invertidas; el mundo físico permanece fijo. Aquí los ejes de coordenadas permanecen fijos; el mundo es reflejado como en un espejo en el plano xz. Específicamente, en esa representación mantenemos los ejes fijos y asociamos un cambio de signo con la componente del vector. Para un espejo en el plano xz, $P_y \to -P_y$. Tenemos

$$P = (P_x, P_y, P_z) \ ,$$
$$P' = (P_x, -P_y, P_z) \ , \quad \text{vector polar.}$$

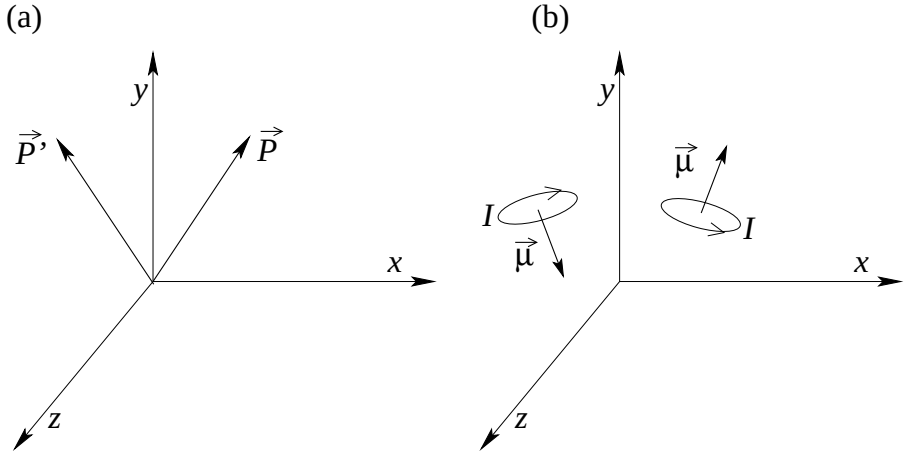


Figura 2.9: (a) Espejo en el plano xz; (b) Espejo en el plano xz.

Un vector axial tal como el momento magnético $\vec{\mu}$ (corriente $\times$ área de loop) o el campo magnético $\vec{B}$ se comporta muy diferentemente bajo reflexión.

Consideremos el campo magnético $\vec{B}$ y el momento magnético $\vec{\mu}$ como producidos por una carga eléctrica moviéndose circularmente. La reflexión revierte el sentido de la rotación de la carga. Los dos loops de corriente y el resultante de los momentos magnéticos son mostrados en la figura 2.9b. Tenemos

$$\begin{aligned} \vec{\mu} &= (\mu_x, \mu_y, \mu_z) \ , \\ \vec{\mu}\,' &= (-\mu_x, \mu_y, -\mu_z) \quad \text{vector axial.} \end{aligned}$$

Concordamos que el Universo no se preocupa si usamos sistemas de coordenadas de mano derecha o mano izquierda, luego no tiene sentido sumar un vector axial a un vector polar. En la ecuación vectorial $\vec{A} = \vec{B}$, ambos $\vec{A}$ y $\vec{B}$ son tanto vectores polares como vectores axiales[7]. Restricciones similares se aplican a los escalares y pseudo escalares y, en general, a los tensores y pseudo tensores.

Usualmente, los pseudo escalares, pseudo vectores, y pseudo tensores transforman como

$$\begin{aligned} S' &= |\,a\,|\,S \\ C'_i &= |\,a\,|\,a_{ij}C_j \ , \\ A'_{ij} &= |\,a\,|\,a_{ik}a_{jl}A_{kl} \ , \end{aligned} \tag{2.84}$$

donde $|\,a\,|$ es el determinante de un arreglo de coeficientes a_{mn}. En nuestra inversión el determinante es

$$|\,a\,| = \begin{vmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{vmatrix} = -1 \tag{2.85}$$

Para la reflexión de un eje, el eje x,

$$|\,a\,| = \begin{vmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{vmatrix} = -1 \tag{2.86}$$

y nuevamente el determinante es $|\,a\,| = -1$. Por otra parte, para todas las rotaciones puras el determinante $|\,a\,|$ es siempre $+1$. A menudo las cantidades que transforman de acuerdo a la ecuación (2.84) son conocidas como densidad tensorial. Ellos son tensores regulares en cuanto a rotaciones se refiere, diferenciándose de los tensores solamente en reflexiones o inversiones de las coordenadas, y luego la única diferencia es la aparición de un signo menos adicional del determinante $|\,a\,|$.

Anteriormente mostramos que el producto escalar triple $S = \vec{A} \times \vec{B} \cdot \vec{C}$ es un escalar (bajo rotaciones). Ahora considerando la transformación de paridad dada por la ecuación (2.82), vemos que $S \to -S$, probando que el producto escalar triple es un pseudo escalar. Este comportamiento fue presagiado por la analogía geométrica de un volumen. Si los tres parámetros del volumen, longitud, ancho, profundidad, cambian de distancias positivas a distancias negativas, el producto de los tres será negativo.

[7]La gran excepción a esto está en el decaimiento beta, interacciones débiles. Aquí el Universo distingue entre sistemas derechos y sistemas izquierdos.

2.9.1. Símbolo de Levi-Civita.

Para usos futuros es conveniente presentar el símbolo tridimensional de Levi-Civita ε_{ijk} definido por

$$\begin{aligned} \varepsilon_{123} = \varepsilon_{231} = \varepsilon_{312} = 1 \ , \\ \varepsilon_{132} = \varepsilon_{213} = \varepsilon_{321} = -1 \ , \\ \text{todos los otros } \varepsilon_{ijk} = 0 \ . \end{aligned} \tag{2.87}$$

Note que ε_{ijk} es totalmente antisimétrico con respecto a todo par de índices. Suponga que tenemos un pseudo tensor de tercer rango δ_{ijk}, el cual en un sistema de coordenadas particular es igual a ε_{ijk}. Luego

$$\delta'_{ijk} = \mid a \mid a_{ip} a_{jq} a_{kr} \, \varepsilon_{pqr} \ , \tag{2.88}$$

por definición de pseudo tensor. Ahora, podemos expresar el determinante de a como

$$\mid a \mid = \varepsilon_{pqr} \, a_{1p} a_{2q} a_{3r} \ , \tag{2.89}$$

por directa expansión del determinante, mostrando que $\delta'_{123} = \mid a \mid^2 = 1 = \varepsilon_{123}$. Considerando las otras posibilidades una a una, encontramos

$$\delta'_{ijk} = \varepsilon_{ijk} \ , \tag{2.90}$$

para rotaciones y reflexiones. Luego ε_{ijk} es un pseudo tensor. Además, se ve como un pseudo tensor isotrópico con las mismas componentes en todos los sistemas de coordenadas cartesianos rotados.

2.9.2. Tensores duales.

A cualquier tensor antisimétrico de segundo rango C_{jk} (en el espacio tridimensional) podemos asociarle un pseudo vector dual C_i definido por

$$C_i = \frac{1}{2} \varepsilon_{ijk} C_{jk} \ . \tag{2.91}$$

Aquí el antisimétrico C_{jk} puede ser escrito

$$C_{jk} = \begin{pmatrix} 0 & C_{12} & -C_{31} \\ -C_{12} & 0 & C_{23} \\ C_{31} & -C_{23} & 0 \end{pmatrix} \ . \tag{2.92}$$

Sabemos que C_i debe transformar como un vector bajo rotaciones a partir de la doble contracción del (pseudo) tensor de quinto rango $\varepsilon_{ijk} C_{mn}$, pero éste es realmente un pseudo vector desde la naturaleza "pseudo" de ε_{ijk}. Específicamente, las componentes de $\vec{C}$ están dadas por

$$(C_1, C_2, C_3) = (C_{23}, C_{31}, C_{12}) \ . \tag{2.93}$$

Note que el orden cíclico de los índices vienen del orden cíclico de las componentes de ε_{ijk}. Esta dualidad, dada por la ecuación (2.93), significa que nuestro producto vectorial tridimensional

puede, literalmente, ser tomado ya sea como un pseudo vector o como un tensor antisimétrico de segundo rango, dependiendo de como escojamos escribirlo.

Si tomamos tres vectores (polares) $\vec{A}$, $\vec{B}$, y $\vec{C}$, podemos definir

$$V_{ijk} = \begin{vmatrix} A_i & A_j & A_k \\ B_i & B_j & B_k \\ C_i & C_j & C_k \end{vmatrix} = A_i B_j C_k - A_i B_k C_j + \dots \ . \tag{2.94}$$

Por una extensión del análisis de la sección 2.6 cada término $A_p B_q C_r$ es visto como un tensor de tercer rango, haciendo V_{ijk} un tensor de rango tres. A partir de su definición como un determinante, V_{ijk} es totalmente antisimétrico, cambiando signo bajo el intercambio de cualquier par de índices, esto es, el intercambio de cualquier par de filas del determinante. La cantidad dual es

$$V = \frac{1}{3!} \varepsilon_{ijk} V_{ijk} \ , \tag{2.95}$$

claramente un pseudo escalar. Por expansión se ve que

$$V = \begin{vmatrix} A_1 & A_2 & A_3 \\ B_1 & B_2 & B_3 \\ C_1 & C_2 & C_3 \end{vmatrix} \ , \tag{2.96}$$

nuestro conocido producto escalar triple.

2.9.3. Tensores irreducibles.

Para algunas aplicaciones, particularmente en la teoría cuántica del momento angular, nuestros tensores cartesianos no son particularmente convenientes. En el lenguaje matemático nuestro tensor general de segundo orden A_{ij} es reducible, lo que significa que puede ser descompuesto en partes, varios tensores de menor orden. En efecto, ya hemos hecho esto. De la ecuación (2.71)

$$A = A_{ii} \ , \tag{2.97}$$

es una cantidad escalar, la traza de A_{ij}.

La parte antisimétrica

$$B_{ij} = \frac{1}{2}(A_{ij} - A_{ji}) \tag{2.98}$$

hemos mostrado que es equivalente a un (pseudo) vector, o

$$B_{ij} = C_k \ , \quad \text{permutación cíclica de } i,\ j,\ k. \tag{2.99}$$

Sustrayendo el escalar A y el vector C_k de nuestro tensor original, tenemos un tensor de segundo orden irreducible, simétrico y de traza nula, S_{ij}, en el cual

$$S_{ij} = \frac{1}{2}(A_{ij} + A_{ji}) - \frac{1}{3} A \, \delta_{ij} \ , \tag{2.100}$$

con cinco componentes independientes. Luego, nuestro tensor cartesiano original puede ser escrito

$$A_{ij} = \frac{1}{3} A \, \delta_{ij} + C_k + S_{ij} \ . \tag{2.101}$$

Las tres cantidades A, C_k, y S_{ij} forman el tensor esférico de orden 0, 1 y 2 respectivamente, transformando como los armónicos esféricos Y_L^M para L =0, 1 y 2.

Un ejemplo específico de la reducción anterior está dada por el tensor cuadrupolo eléctrico simétrico

$$Q_{ij} = \int (3x_i x_j - r^2\delta_{ij})\rho(x_1, x_2, x_3)\, d^3x \ .$$

El término $-r^2\delta_{ij}$ representa una resta de la traza escalar (los 3 términos $i = j$). El resultante Q_{ij} tiene traza cero.

2.10. Tensores no cartesianos, diferenciación covariante.

La distinción entre transformaciones contravariante y transformaciones covariante fueron establecidas en la sección 2.6. Luego, por conveniencia, nos restringimos a coordenadas cartesianas (en la cual la distinción desaparece). Ahora en estas dos secciones volvemos a las coordenadas no cartesianas y resurge la distinción entre contravariante y covariante. Como en la sección 2.6, un superíndice será usado para denotar la dependencia contravariante y un subíndice para diferenciar la covariante. El tensor métrico de la sección 2.1 será usado para relacionar los índices contravariante y covariante.

El énfasis en esta sección es sobre diferenciación, culminando en la construcción de una derivada covariante. Vimos en la sección 2.7 que la derivada de un campo vectorial es un tensor de segundo orden en coordenadas cartesianas. La derivada covariante de un campo vectorial es un tensor de segundo orden en sistemas de coordenadas no cartesianas.

2.10.1. Tensor métrico, subida y bajada de índices.

Empecemos con un conjunto de vectores base $\vec{\varepsilon}_i$ tal que un desplazamiento infinitesimal $d\vec{s}$ podría estar dado por

$$d\vec{s} = \vec{\varepsilon}_1\, dq^1 + \vec{\varepsilon}_2\, dq^2 + \vec{\varepsilon}_3\, dq^3 \ . \tag{2.102}$$

Por conveniencia tomamos $\vec{\varepsilon}_1$, $\vec{\varepsilon}_2$, $\vec{\varepsilon}_3$ formando un sistema diestro. Estos tres vectores no son necesariamente ortogonales. También, se requerirá una limitación al espacio tridimensional solamente para la discusión de los productos cruz y rotores. De lo contrario estos $\vec{\varepsilon}_i$ pueden estar en un espacio de N-dimensiones, incluyendo la cuarta dimensión espacio-tiempo de la relatividad especial y general. Los vectores base $\vec{\varepsilon}_i$ pueden ser expresados por

$$\vec{\varepsilon}_i = \frac{\partial \vec{s}}{\partial q^i} \ , \tag{2.103}$$

Note, sin embargo, que los $\vec{\varepsilon}_i$ no tienen necesariamente magnitud uno. Se puede probar que los vectores unitarios son

$$\hat{e}_i = \frac{1}{h_i}\frac{\partial \vec{s}}{\partial q^i} \ , \quad \text{(no hay suma)},$$

y por lo tanto

$$\vec{\varepsilon}_i = h_i \hat{e}_i \ , \quad \text{(no hay suma)}. \tag{2.104}$$

Los $\vec{\varepsilon}_i$ están relacionados a los vectores unitarios $\hat{e}_i$ por el factor de escala h_i de la sección 2.2. Los $\hat{e}_i$ no tienen dimensiones, los $\vec{\varepsilon}_i$ tienen dimensiones de h_i. En coordenadas polares esféricas, como un ejemplo específico,

$$\begin{aligned} \vec{\varepsilon}_r &= \hat{e}_r \ , \\ \vec{\varepsilon}_\theta &= r\,\hat{e}_\theta \ , \\ \vec{\varepsilon}_\phi &= r\ \text{sen}\,\theta\,\hat{e}_\phi \ . \end{aligned} \tag{2.105}$$

Como en la sección 2.1, construimos el cuadrado de un desplazamiento diferencial

$$(ds)^2 = d\vec{s}\cdot d\vec{s} = (\vec{\varepsilon}_i\,dq^i)^2 = \vec{\varepsilon}_i\cdot\vec{\varepsilon}_j\,dq^i dq^j \ . \tag{2.106}$$

Comparando esto con $(ds)^2$ de la sección 2.1, ecuación (2.6), definimos $\vec{\varepsilon}_i\cdot\vec{\varepsilon}_j$ como el tensor métrico covariante

$$\vec{\varepsilon}_i\cdot\vec{\varepsilon}_j = g_{ij} \ . \tag{2.107}$$

Claramente g_{ij} es simétrico. La naturaleza del tensor g_{ij} se deduce a partir de la regla del cociente. Tomemos la relación

$$g^{ik}g_{kj} = \delta^i_j \ , \tag{2.108}$$

para definir el correspondiente tensor contravariante g^{ik}. El contravariante g^{ik} entra como el inverso[8] del covariante g_{kj}. Usemos este contravariante g^{ik} para subir índices, convirtiendo un índice covariante en uno contravariante, como se muestra más adelante. Así mismo el covariante g_{kj} será usado para bajar índices. La elección de g^{ik} y g_{kj} para esta operación de subir y bajar es arbitraria. Cualquier tensor de segundo orden (y su inverso) podría hacerlo. Específicamente, tenemos

$$\begin{aligned} g^{ij}\vec{\varepsilon}_j = \vec{\varepsilon}^{\,i} \ , &\quad \text{relación de covariante y} \\ &\quad \text{contravariante de vectores bases} \\ g^{ij}\vec{F}_j = \vec{F}^{\,i} \ , &\quad \text{relación de covariante y} \\ &\quad \text{contravariante de componentes vectoriales.} \end{aligned} \tag{2.109}$$

Entonces

$$\begin{aligned} g_{ij}\vec{\varepsilon}^{\,j} = \vec{\varepsilon}_i \ , &\quad \text{son las correspondientes relaciones} \\ g_{ij}\vec{F}^{\,j} = \vec{F}_i \ , &\quad \text{de bajada de índices.} \end{aligned} \tag{2.110}$$

Como un ejemplo de esas transformaciones comenzaremos con la forma contravariante del vector

$$\vec{F} = F^i\vec{\varepsilon}_i \ . \tag{2.111}$$

De las ecuaciones (2.109) y (2.110)

$$\vec{F} = F_j g^{ji} g_{ik}\vec{\varepsilon}^{\,k} = F_j\vec{\varepsilon}^{\,j} \ , \tag{2.112}$$

la igualdad final viene de la ecuaciones (2.108). La ecuación (2.111) da la representación contravariante de $\vec{F}$. La ecuación (2.112) da la correspondiente representación covariante del mismo $\vec{F}$.

[8] Si el tensor g_{ij} es escrito como una matriz, el tensor g^{ij} está dado por la matriz inversa.

Deberíamos enfatizar de nuevo que $\vec{\varepsilon}_i$ y $\vec{\varepsilon}^j$ no son unitarios. Esto puede verse en las ecuaciones (2.105) y en el tensor métrico g_{ij} para coordenadas polares esféricas y su inverso g^{ij}:

$$(g_{ij}) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & r^2 & 0 \\ 0 & 0 & r^2 \operatorname{sen}^2\theta \end{pmatrix} \qquad (g^{ij}) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \dfrac{1}{r^2} & 0 \\ 0 & 0 & \dfrac{1}{r^2 \operatorname{sen}^2\theta} \end{pmatrix} .$$

2.10.2. Derivadas y símbolos de Christoffel.

Formemos la diferencial de un escalar

$$d\psi = \frac{\partial \psi}{\partial q^i}\, dq^i \ . \tag{2.113}$$

Ya que la dq^i son las componentes de un vector contravariante, las derivadas parciales, $\partial\psi/\partial q^i$ debieran ser un vector covariante, por la regla del cociente. El gradiente de un escalar llega a ser

$$\vec{\boldsymbol{\nabla}}\psi = \frac{\partial \psi}{\partial q^i}\, \vec{\varepsilon}^{\,i} \ . \tag{2.114}$$

Deberíamos notar que $\partial\psi/\partial q^i$ no son las componentes del gradiente de la sección 2.2, ya que $\vec{\varepsilon}^{\,i} \neq \hat{e}_i$ de la sección 2.2.

Continuando con las derivadas de un vector, encontramos que la situación es mucho más complicada porque los vectores bases $\vec{\varepsilon}_i$, en general, no son constantes. Recordemos que no estamos más restringidos a las coordenadas cartesianas con sus convenientes $\hat{x}$, $\hat{y}$, $\hat{z}$. La diferenciación directa es

$$\frac{\partial \vec{V}}{\partial q^j} = \frac{\partial V^i}{\partial q^j}\vec{\varepsilon}_i + V^i \frac{\partial \vec{\varepsilon}_i}{\partial q^j} \ . \tag{2.115}$$

Ahora $\partial\vec{\varepsilon}_i/\partial q^j$ será alguna combinación lineal de $\vec{\varepsilon}_k$ con coeficientes que dependen de los índices i y j de las derivadas parciales y el índice k del vector base. Escribimos

$$\frac{\partial \vec{\varepsilon}_i}{\partial q^j} = \Gamma^k_{ij}\vec{\varepsilon}_k \ . \tag{2.116}$$

Multiplicando por $\vec{\varepsilon}^{\,m}$ y usando $\vec{\varepsilon}^{\,m} \cdot \vec{\varepsilon}_k = \delta^m_k$ (pruébelo).

$$\Gamma^m_{ij} = \vec{\varepsilon}^{\,m} \cdot \frac{\partial \vec{\varepsilon}_i}{\partial q^j} \ . \tag{2.117}$$

El Γ^k_{ij} es un símbolo de Christoffel (del segundo tipo). También se le conoce como "coeficiente de conexión". Estos Γ^k_{ij} no son tensores de rango tres y las $\partial V^i/\partial q^j$ de la ecuación (2.115) no son tensores de segundo rango. En coordenadas cartesianas, $\Gamma^k_{ij} = 0$ para todos los valores de los índices i, j, k.

Usando la ecuación (2.103), obtenemos

$$\frac{\partial \vec{\varepsilon}_i}{\partial q^j} = \frac{\partial^2 \vec{s}}{\partial q^j \partial q^i} = \frac{\partial \vec{\varepsilon}_j}{\partial q^i} == \Gamma^k_{ji}\vec{\varepsilon}_k \ . \tag{2.118}$$

Luego estos símbolos de Christoffel son simétricos en los dos índices inferiores:

$$\Gamma^k_{ij} = \Gamma^k_{ji} \ . \tag{2.119}$$

2.10.3. Derivada covariante.

Con los símbolos de Christoffel, la ecuación (2.115) puede ser reescrita

$$\frac{\partial \vec{V}}{\partial q^j} = \frac{\partial V^i}{\partial q^j}\vec{\varepsilon}_i + V^i \Gamma^k_{ij}\vec{\varepsilon}_k \ . \tag{2.120}$$

Ahora i y k en el último término son índices mudos. Intercambiando i y k, tenemos

$$\frac{\partial \vec{V}}{\partial q^j} = \left(\frac{\partial V^i}{\partial q^j} + V^k \Gamma^i_{kj}\right)\vec{\varepsilon}_i \ . \tag{2.121}$$

La cantidad entre paréntesis es denominada **derivada covariante**, $V^i_{;j}$. Tenemos

$$V^i_{;j} \equiv \frac{\partial V^i}{\partial q^j} + V^k \Gamma^i_{kj} \ . \tag{2.122}$$

Los subíndices $;j$ indican que la diferenciación es con respecto a q^j. La diferencial $d\vec{V}$ se convierte en

$$d\vec{V} = \frac{\partial \vec{V}}{\partial q^j} dq^j = [V^i_{;j} dq^j]\vec{\varepsilon}_i \ . \tag{2.123}$$

Una comparación con las ecuaciones (2.102) o (2.111) muestra que la cantidad entre paréntesis cuadrados es la i-ésima componente de un vector contravariante. Ya que dq^j es la j-ésima componente de un vector contravariante, $V^i_{;j}$ debe ser la componente ij-ésima de un tensor mixto de segundo rango (regla del cociente). Las derivadas covariantes de las componentes contravariantes de un vector forma un tensor mixto de segundo rango, $V^i_{;j}$.

Ya que los símbolos de Christoffel desaparecen en coordenadas cartesianas, la derivada covariante y la derivada parcial ordinaria coinciden

$$\frac{\partial V^i}{\partial q^j} = V^i_{;j} \ , \qquad \text{(En coordenadas cartesianas).} \tag{2.124}$$

Se puede demostrar que la derivada covariante de un vector covariante V_i está dada por

$$V_{i;j} = \frac{\partial V_i}{\partial q^j} - V_k \Gamma^k_{ij} \ . \tag{2.125}$$

Como $V^i_{;j}$, el tensor $V_{i;j}$ es de rango dos.

La importancia física de la derivada covariante es:

> *Un reemplazo consistente de las derivadas parciales regulares por derivadas covariantes lleva las leyes de la Física (por componentes) desde un espacio tiempo plano al espacio tiempo curvo (Riemanniano) de la relatividad general. Realmente, esta sustitución puede ser tomada como una declaración matemática del principio de equivalencia.*[9]

[9] C.W. Misner, K.S. Thorne, and J.A. Wheeler, *Gravitation*. San Francisco: W. H. Freeman (1973), p 387.

2.10.4. Los símbolos de Christoffel como derivadas del tensor métrico.

A menudo es conveniente tener una expresión explícita para los símbolos de Christoffel en términos de derivadas del tensor métrico. Como un paso inicial, definimos el símbolo de Christoffel del primer tipo $[ij,k]$ por

$$[ij,k] \equiv g_{mk}\Gamma^m_{ij} \ . \tag{2.126}$$

Este $[ij,k]$ no es un tensor de rango tres. De la ecuación (2.117)

$$\begin{aligned} [ij,k] &= g_{mk}\vec{\varepsilon}^m \cdot \frac{\partial \vec{\varepsilon}_i}{\partial q^j} \ , \\ &= \vec{\varepsilon}_k \cdot \frac{\partial \vec{\varepsilon}_i}{\partial q^j} \ . \end{aligned} \tag{2.127}$$

Ahora diferenciamos $g_{ij} = \vec{\varepsilon}_i \cdot \vec{\varepsilon}_j$, ecuación (2.107):

$$\begin{aligned} \frac{\partial g_{ij}}{\partial q^k} &= \frac{\partial \vec{\varepsilon}_i}{\partial q^k} \cdot \vec{\varepsilon}_j + \vec{\varepsilon}_i \cdot \frac{\partial \vec{\varepsilon}_j}{\partial q^k} \ , \\ &= [ik,j] + [jk,i] \ , \end{aligned} \tag{2.128}$$

por la ecuación (2.127).

Luego

$$[ij,k] = \frac{1}{2}\left\{ \frac{\partial g_{ik}}{\partial q^j} + \frac{\partial g_{jk}}{\partial q^i} - \frac{\partial g_{ij}}{\partial q^k} \right\} \ , \tag{2.129}$$

y

$$\begin{aligned} \Gamma^s_{ij} &= g^{ks}[ij,k] \ , \\ &= \frac{1}{2} g^{ks}\left\{ \frac{\partial g_{ik}}{\partial q^j} + \frac{\partial g_{jk}}{\partial q^i} - \frac{\partial g_{ij}}{\partial q^k} \right\} \ . \end{aligned} \tag{2.130}$$

Estos símbolos de Christoffel y las derivadas covariantes son aplicadas en la próxima sección.

2.11. Operadores diferenciales de tensores.

En esta sección la derivada covariante de la sección 2.10 es aplicada para derivar las operaciones diferenciales vectoriales de la sección 2.2 en la forma tensorial general.

2.11.1. Divergencia.

Reemplazando las derivadas parciales por la derivada covariante, tomamos la divergencia como

$$\vec{\boldsymbol{\nabla}} \cdot \vec{V} = V^i_{;i} = \frac{\partial V^i}{\partial q^i} + V^k \Gamma^i_{ik} \ . \tag{2.131}$$

Expresando Γ^i_{ik} por la ecuación (2.130), tenemos

$$\Gamma^i_{ik} = \frac{1}{2} g^{im} \left\{ \frac{\partial g_{im}}{\partial q^k} + \frac{\partial g_{km}}{\partial q^i} - \frac{\partial g_{ik}}{\partial q^m} \right\} . \tag{2.132}$$

Cuando contraemos con g^{im} los dos últimos términos en el paréntesis de llave se cancelan, ya que

$$g^{im} \frac{\partial g_{km}}{\partial q^i} = g^{mi} \frac{\partial g_{ki}}{\partial q^m} = g^{im} \frac{\partial g_{ik}}{\partial q^m} . \tag{2.133}$$

Entonces

$$\Gamma^i_{ik} = \frac{1}{2} g^{im} \frac{\partial g_{im}}{\partial q^k} . \tag{2.134}$$

Por la teoría de determinantes

$$\frac{\partial g}{\partial q^k} = g g^{im} \frac{\partial g_{im}}{\partial q^k} , \tag{2.135}$$

donde g es el determinante de la métrica, $g = det(g_{ij})$. Sustituyendo este resultado en la ecuación (2.134), obtenemos

$$\Gamma^i_{ik} = \frac{1}{2g} \frac{\partial g}{\partial q^k} = \frac{1}{g^{1/2}} \frac{\partial g^{1/2}}{\partial q^k} . \tag{2.136}$$

Esto da

$$\vec{\boldsymbol{\nabla}} \cdot \vec{V} = V^i_{;i} = \frac{1}{g^{1/2}} \frac{\partial}{\partial q^k} (g^{1/2} V^k) . \tag{2.137}$$

Para comparar este resultado con la ecuación (2.21), note que $h_1 h_2 h_3 = g^{1/2}$ y $V^i = V_i / h_i$ (V^i es el coeficiente contravariante de $\vec{\varepsilon}_i$ y no hay suma), donde V_i es el coeficiente de $\hat{e}_i$.

2.11.2. Laplaciano.

En la sección 2.2 reemplazamos el vector $\vec{V}$ en $\vec{\boldsymbol{\nabla}} \cdot \vec{V}$ por $\vec{\boldsymbol{\nabla}} \psi$ para obtener al Laplaciano $\vec{\boldsymbol{\nabla}} \cdot \vec{\boldsymbol{\nabla}} \psi$. Aquí tenemos un V^i contravariante. Usando el tensor métrico para crear un contravariante $\vec{\boldsymbol{\nabla}} \psi$, hacemos la sustitución

$$V^i \to g^{ik} \frac{\partial \psi}{\partial q^k} . \tag{2.138}$$

Entonces el Laplaciano $\vec{\boldsymbol{\nabla}} \cdot \vec{\boldsymbol{\nabla}} \psi$ se convierte

$$\vec{\boldsymbol{\nabla}} \cdot \vec{\boldsymbol{\nabla}} \psi = \frac{1}{g^{1/2}} \frac{\partial}{\partial q^i} (g^{1/2} g^{ik} \frac{\partial \psi}{\partial q^k}) . \tag{2.139}$$

Para los sistemas *ortogonales* de la sección 2.2 el tensor métrico es diagonal y el contravariante g^{ii} (no hay suma) llega a ser

$$g^{ii} = (h_i)^{-2} .$$

La ecuación (2.139) se reduce a

$$\vec{\boldsymbol{\nabla}} \cdot \vec{\boldsymbol{\nabla}} \psi = \frac{1}{h_1 h_2 h_3} \frac{\partial}{\partial q^i} \left(\frac{h_1 h_2 h_3}{h_i^2} \frac{\partial \psi}{\partial q^k} \right) .$$

en concordancia con la ecuación (2.22).

2.11.3. Rotor.

La diferencia de derivadas que aparece en el rotor (ecuación (2.26)) será escrita

$$\frac{\partial V_i}{\partial q^j} - \frac{\partial V_j}{\partial q^i} \ .$$

Nuevamente, recordemos que las componentes de V_i aquí son los coeficientes contravariantes de los vector no unitarios base $\vec{\varepsilon}^{\,i}$. Los V_i de la sección 2.2 son coeficientes de los vectores unitarios $\hat{e}_i$. Usando

$$\Gamma^k_{ij} = \Gamma^k_{ji} \ ,$$

obtenemos

$$\begin{aligned} \frac{\partial V_i}{\partial q^j} - \frac{\partial V_j}{\partial q^i} &= \frac{\partial V_i}{\partial q^j} - V_k\Gamma^k_{ij} - \frac{\partial V_j}{\partial q^i} + V_k\Gamma^k_{ji} \\ &= V_{i;j} - V_{j;i} \ . \end{aligned} \tag{2.140}$$

Las diferencias características de derivadas de un rotor llegan a ser diferencias en derivadas covariantes y por lo tanto es un tensor de segundo orden (covariante en ambos índices). Como enfatizamos en la sección 2.9, la forma vectorial especial del rotor existe solamente en el espacio tridimensional.

De la ecuación (2.130) es claro que todos los símbolos de Christoffel de tres índices se anulan en el espacio de Minkowski y en el espacio-tiempo real de la relatividad especial con

$$g_{\lambda\mu} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix} \ . \tag{2.141}$$

Aquí

$$x_0 = ct \ , \quad x_1 = x \ , \quad x_2 = y \ , \quad \text{y} \quad x_3 = z \ .$$

Esto completa el desarrollo de los operadores diferenciales en la forma tensorial general. En adición a los campos elásticos y electromagnéticos, estas formas diferenciales encuentran aplicación en mecánica (mecánica Lagrangiana, Hamiltoniana, y las ecuaciones de Euler para la rotación de cuerpos rígidos); mecánica de fluidos, y quizás mucho más importante que todas, en el espacio-tiempo curvado de la teoría moderna gravitacional.

Capítulo 3

Determinantes y matrices.

versión final 2.20-021015[1]

3.1. Determinantes.

Comenzamos el estudio de matrices resolviendo ecuaciones lineales las cuales nos llevan a determinantes y matrices. El concepto de determinante y su notación fueron introducidos por Leibniz.

3.1.1. Ecuaciones lineales homogéneas.

Una de las mayores aplicaciones de los determinantes está en el establecimiento de una condición para la existencia de una solución no trivial de un conjunto de ecuaciones algebraicas lineales homogéneas. Supongamos que tenemos tres incógnitas x_1, x_2, x_3 (o n ecuaciones con n incógnitas).

$$\begin{aligned} a_1x_1 + a_2x_2 + a_3x_3 &= 0 \ , \\ b_1x_1 + b_2x_2 + b_3x_3 &= 0 \ , \\ c_1x_1 + c_2x_2 + c_3x_3 &= 0 \ . \end{aligned} \tag{3.1}$$

El problema es: ¿en qué condiciones hay alguna solución, aparte de la solución trivial $x_1 = 0$, $x_2 = 0$, $x_3 = 0$? Si usamos notación vectorial $\vec{x} = (x_1, x_2, x_3)$ para la solución y tres filas $\vec{a} = (a_1, a_2, a_3)$, $\vec{b} = (b_1, b_2, b_3)$, $\vec{c} = (c_1, c_2, c_3)$ para los coeficientes, tenemos que las tres ecuaciones, ecuación (3.1), se convierten en

$$\vec{a} \cdot \vec{x} = 0 \ , \quad \vec{b} \cdot \vec{x} = 0 \ , \quad \vec{c} \cdot \vec{x} = 0 \ . \tag{3.2}$$

Estas tres ecuaciones vectoriales tienen la interpretación geométrica obvia que $\vec{x}$ es ortogonal a $\vec{a}$, $\vec{b}$, $\vec{c}$. Si el volumen sustentado por $\vec{a}$, $\vec{b}$, $\vec{c}$ dado por el determinante (o el producto escalar triple)

$$D_3 = (\vec{a} \times \vec{b}) \cdot \vec{c} = \det(\vec{a}, \vec{b}, \vec{c}) = \begin{vmatrix} a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \\ c_1 & c_2 & c_3 \end{vmatrix} \ , \tag{3.3}$$

[1]Este capítulo está basado en el tercer capítulo del libro: *Mathematical Methods for Physicists, fourth edition* de George B. Arfken & Hans J. Weber, editorial Academic Press.

no es cero, claramente sólo existe la solución trivial $\vec{x} = 0$.

Vice-versa, si el anterior determinante de coeficientes se anula, uno de los vectores columna es una combinación lineal de otros dos. Supongamos que $\vec{c}$ está en el plano que sustenta $\vec{a}$, $\vec{b}$, *i.e.*, la tercera ecuación es una combinación lineal de las primeras dos y no es independiente. Luego $\vec{x}$ es ortogonal a ese plano tal que $\vec{x} \sim \vec{a} \times \vec{b}$. Ya que las ecuaciones homogéneas pueden ser multiplicadas por números arbitrarios, solamente las relaciones de x_i son relevantes, para lo cual obtenemos razones de determinantes de 2×2

$$\begin{aligned} \frac{x_1}{x_3} &= \frac{(a_2 b_3 - a_3 b_2)}{(a_1 b_2 - a_2 b_1)} \ , \\ \frac{x_2}{x_3} &= -\frac{(a_1 b_3 - a_3 b_1)}{(a_1 b_2 - a_2 b_1)} \ , \end{aligned} \tag{3.4}$$

a partir de los componentes del producto cruz $\vec{a} \times \vec{b}$.

3.1.2. Ecuaciones lineales no homogéneas.

El caso más simple es de dos ecuaciones con dos incógnitas

$$\begin{aligned} a_1 x_1 + a_2 x_2 &= a_3 \ , \\ b_1 x_1 + b_2 x_2 &= b_3 \ , \end{aligned} \tag{3.5}$$

puede ser reducido al caso previo embebiéndolo en un espacio tridimensional con una solución vectorial $\vec{x} = (x_1, x_2, -1)$ y el vector fila $\vec{a} = (a_1, a_2, a_3)$, $\vec{b} = (b_1, b_2, b_3)$. Como antes, ecuación (3.5) en notación vectorial, $\vec{a} \cdot \vec{x} = 0$ y $\vec{b} \cdot \vec{x} = 0$, implica que $\vec{x} \sim \vec{a} \times \vec{b}$ tal que el análogo de la ecuación (3.4) se mantiene. Para que esto se aplique, la tercera componente de $\vec{a} \times \vec{b}$ debiera ser distinta de cero, *i.e.*, $a_1 b_2 - a_2 b_1 \neq 0$, ya que la tercera componente de $\vec{x}$ es $-1 \neq 0$. Esto produce que los x_i tengan la forma

$$x_1 = \frac{(a_3 b_2 - a_2 b_3)}{(a_1 b_2 - a_2 b_1)} = \frac{\begin{vmatrix} a_3 & a_2 \\ b_3 & b_2 \end{vmatrix}}{\begin{vmatrix} a_1 & a_2 \\ b_1 & b_2 \end{vmatrix}} \tag{3.6a}$$

$$x_2 = \frac{(a_1 b_3 - a_3 b_1)}{(a_1 b_2 - a_2 b_1)} = \frac{\begin{vmatrix} a_1 & a_3 \\ b_1 & b_3 \end{vmatrix}}{\begin{vmatrix} a_1 & a_2 \\ b_1 & b_2 \end{vmatrix}} \ . \tag{3.6b}$$

El determinante en el numerador de $x_1 (x_2)$ es obtenido a partir del determinante de los coeficientes $\begin{vmatrix} a_1 & a_2 \\ b_1 & b_2 \end{vmatrix}$ reemplazando el primer vector columna (segundo) por el vector $\begin{pmatrix} a_3 \\ b_3 \end{pmatrix}$ del lado inhomogéneo de la ecuación (3.5).

Estas soluciones de ecuación lineal en términos de determinantes pueden ser generalizados a n dimensiones. El determinante es un arreglo cuadrado

$$D_n = \begin{vmatrix} a_1 & a_2 & \dots & a_n \\ b_1 & b_2 & \dots & b_n \\ c_1 & c_2 & \dots & c_n \\ \cdot & \cdot & \dots & \cdot \end{vmatrix} \ , \tag{3.7}$$

de números (o funciones), los coeficientes de n ecuaciones lineales en nuestro caso. El número n de columnas (y de filas) en el arreglo es llamado algunas veces como el orden del determinante. La generalización de la expansión del producto escalar triple (de vectores fila de las tres ecuaciones lineales) tiende al siguiente valor del determinante D_n en n dimensiones,

$$D_n = \sum_{i,j,k,\ldots} \varepsilon_{ijk\ldots} a_i b_j c_k \ldots \ , \tag{3.8}$$

donde $\varepsilon_{ijk}\ldots$, análogo al símbolo de Levi-Civita de la sección (2.2), es $+1$ para permutaciones pares $(ijk\ldots)$ de $(123\ldots n)$, -1 para permutaciones impares, y cero si algún índice es repetido.

Específicamente, para el determinante de orden tres D_3 de las ecuaciones (3.3) y (3.8) tenemos

$$D_3 = +a_1b_2c_3 - a_1b_3c_2 - a_2b_1c_3 + a_2b_3c_1 + a_3b_1c_2 - a_3b_2c_1 \ . \tag{3.9}$$

El determinante de orden tres, entonces, es esta particular combinación lineal de productos. Cada producto contiene uno y sólo un elemento de cada fila y de cada columna. Cada producto es sumado si las columnas (los índices) representan una permutación par de (123) y restando si corresponde a una permutación impar. La ecuación (3.3) puede ser considerada la notación abreviada de la ecuación (3.9). El número de términos en la suma (ecuación (3.8)) es 24 para un determinante de cuarto orden, en general, $n!$ para un determinante de orden n. A causa de la aparición de signos negativos en la ecuación (3.9) puede haber cancelaciones. Debido a esto es muy posible que un determinante de elementos grandes tenga un valor pequeño.

Algunas propiedades útiles de los determinantes de n-ésimo orden siguen de la ecuación (3.8). De nuevo, para ser específico, la ecuación (3.9) para determinantes de orden tres es usada para ilustrar estas propiedades.

3.1.3. Desarrollo Laplaciano por las menores.

La ecuación (3.9) puede ser reescrita

$$\begin{aligned} D_3 &= a_1(b_2c_3 - b_3c_2) - a_2(b_1c_3 - b_3c_1) + a_3(b_1c_2 - b_2c_1) \\ &= a_1 \begin{vmatrix} b_2 & b_3 \\ c_2 & c_3 \end{vmatrix} - a_2 \begin{vmatrix} b_1 & b_3 \\ c_1 & c_3 \end{vmatrix} + a_3 \begin{vmatrix} b_1 & b_2 \\ c_1 & c_2 \end{vmatrix} \ . \end{aligned} \tag{3.10}$$

En general, el determinante de orden n-ésimo puede ser expandido como una combinación lineal de productos de elementos de alguna fila (o columna) por determinantes de orden $(n-1)$ formados suprimiendo la fila y la columna del determinante original en el cual aparece el elemento. Este arreglo reducido (2×2 en el ejemplo específico) es llamado una menor. Si el elemento está en la i-ésima fila y en la j-ésima columna, el signo asociado con el producto es $(-1)^{i+j}$. La menor con este signo es llamada el cofactor. Si M_{ij} es usado para designar la menor formada omitiendo la fila i y la columna j y c_{ij} es el cofactor correspondiente, la ecuación (3.10) se convierte en

$$D_3 = \sum_{j=1}^{3} (-1)^{j+1} a_j M_{1j} = \sum_{j=1}^{3} a_j c_{1j} \ . \tag{3.11}$$

En este caso, expandiendo a lo largo de la primera fila, tenemos $i = 1$ y la suma es sobre j, las columnas.

Esta expansión de Laplace puede ser usada para sacar ventaja en la evaluación de determinantes de alto orden en la cual muchos de los elementos son nulos. Por ejemplo, para encontrar el valor del determinante

$$D = \begin{vmatrix} 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -1 & 0 \end{vmatrix} , \tag{3.12}$$

expandimos a través de la fila superior para obtener

$$D = (-1)^{1+2} \cdot (1) \begin{vmatrix} -1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{vmatrix} . \tag{3.13}$$

Nuevamente, expandimos a través de la fila superior para obtener

$$\begin{aligned} D &= (-1) \cdot (-1)^{1+1} \cdot (-1) \begin{vmatrix} 0 & 1 \\ -1 & 0 \end{vmatrix} \\ &= \begin{vmatrix} 0 & 1 \\ -1 & 0 \end{vmatrix} = 1 . \end{aligned} \tag{3.14}$$

Este determinante D (ecuación (3.12)) está formado de una de las matrices de Dirac que aparecen en la teoría relativista del electrón de Dirac.

3.1.4. Antisimetría.

El determinante cambia de signo si cualquier par de filas son intercambiadas o si cualquier par de columnas son intercambiadas. Esto deriva del carácter par-impar del Levi-Civita ε en la ecuación (3.8) o explícitamente de la forma de las ecuaciones (3.9) y (3.10).

Esta propiedad fue usada en la sección 2.9 para desarrollar una combinación lineal totalmente antisimétrica. Esto es también frecuentemente usado en Mecánica Cuántica en la construcción de una función de onda de muchas partículas que, en concordancia con el principio de exclusión de Pauli, será antisimétrica bajo el intercambio de cualquier par de partículas idénticas con spin 1/2 (electrones, protones, neutrones, etc).

Como un caso especial de antisimetría, cualquier determinante con dos filas iguales o dos columnas iguales es nulo.

Si cada elemento en una fila o de una columna es cero el determinante completo es nulo.

Si cada elemento en una fila o de una columna es multiplicado por una constante, el determinante completo es multiplicado por esa constante.

El valor de un determinante es inalterado si un múltiplo de una fila es añadido (columna por columna) a otra fila o si un múltiplo de una columna es añadido (fila por fila) a otra columna. Tenemos

$$\begin{vmatrix} a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \\ c_1 & c_2 & c_3 \end{vmatrix} = \begin{vmatrix} a_1 + ka_2 & a_2 & a_3 \\ b_1 + kb_2 & b_2 & b_3 \\ c_1 + kc_2 & c_2 & c_3 \end{vmatrix} . \tag{3.15}$$

Usando el desarrollo de Laplace sobre el lado derecho, obtenemos

$$\begin{vmatrix} a_1 + ka_2 & a_2 & a_3 \\ b_1 + kb_2 & b_2 & b_3 \\ c_1 + kc_2 & c_2 & c_3 \end{vmatrix} = \begin{vmatrix} a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \\ c_1 & c_2 & c_3 \end{vmatrix} + k \begin{vmatrix} a_2 & a_2 & a_3 \\ b_2 & b_2 & b_3 \\ c_2 & c_2 & c_3 \end{vmatrix} , \tag{3.16}$$

entonces por la propiedad de antisimetría, el segundo determinante del lado derecho se anula, verificando la ecuación (3.15).

En un caso especial, un determinante es igual a cero si cualquier par de filas o columnas son proporcionales.

Volviendo a las ecuaciones homogéneas (3.1) y multiplicando el determinante de los coeficientes por x_1, y luego sumando x_2 veces la segunda columna y x_3 veces la tercera columna, podemos establecer directamente la condición para la presencia de una solución no trivial para la ecuación (3.1):

$$x_1 \begin{vmatrix} a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \\ c_1 & c_2 & c_3 \end{vmatrix} = \begin{vmatrix} x_1 a_1 & a_2 & a_3 \\ x_1 b_1 & b_2 & b_3 \\ x_1 c_1 & c_2 & c_3 \end{vmatrix} = \begin{vmatrix} a_1 x_1 + a_2 x_2 + a_3 x_3 & a_2 & a_3 \\ b_1 x_1 + b_2 x_2 + b_3 x_3 & b_2 & b_3 \\ c_1 x_1 + c_2 x_2 + c_3 x_3 & c_2 & c_3 \end{vmatrix} = \begin{vmatrix} 0 & a_2 & a_3 \\ 0 & b_2 & b_3 \\ 0 & c_2 & c_3 \end{vmatrix} = 0 \ . \tag{3.17}$$

Por lo tanto x_1 (x_2 y x_3) deberían ser cero a menos que el determinante de los coeficientes sea nulo. Podemos mostrar que si el determinante de los coeficientes es nulo, existe realmente una solución no trivial.

Si nuestras ecuaciones lineales son inhomogéneas, esto es, como en la ecuación (3.5) o si los ceros en el lado derecho de la ecuación (3.1) fueran reemplazados por a_4, b_4, c_4, respectivamente, y con de la ecuación (3.17) obtenemos,

$$x_1 = \frac{\begin{vmatrix} a_4 & a_2 & a_3 \\ b_4 & b_2 & b_3 \\ c_4 & c_2 & c_3 \end{vmatrix}}{\begin{vmatrix} a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \\ c_1 & c_2 & c_3 \end{vmatrix}} , \tag{3.18}$$

la cual generaliza la ecuación (3.6a) a la dimensión $n = 3$. Si el determinante de los coeficientes se anula, el conjunto de ecuaciones no homogéneas no tiene solución a menos que el numerador también se anule. En este caso las soluciones pueden existir pero ellas no son únicas.

Para el trabajo numérico, esta solución del determinante, ecuación (3.18), es enormemente difícil de manejar. El determinante puede involucrar grandes números con signos alternados, y en la resta de dos números grandes el error relativo podría remontarse al punto que hace que el resultado no tenga valor. También, aunque el método del determinante es ilustrado aquí con tres ecuaciones y tres incógnitas, podríamos fácilmente tener 200 ecuaciones con 200 incógnitas las cuales, involucran sobre 200! términos por determinante, lo que pone un desafío muy alto a la velocidad computacional. Debería haber una mejor manera. En efecto, hay una mejor manera. Una de las mejores es un proceso a menudo llamado eliminación de Gauss. Para ilustrar esta técnica, consideremos el siguiente conjunto de ecuaciones.

Resolvamos

$$\begin{aligned} 3x + 2y + z &= 11 \\ 2x + 3y + z &= 13 \\ x + y + 4z &= 12 \ . \end{aligned} \tag{3.19}$$

El determinante de la ecuación lineal no homogénea ecuación (3.19) es 18, por lo tanto existe una solución.

Por conveniencia y para una óptima precisión numérica, las ecuaciones son reordenadas tal que los coeficientes mayores corran a lo largo de la diagonal principal (superior izquierda a inferior derecha). Esto ha sido hecho en el conjunto anterior.

La técnica de Gauss es usar la primera ecuación para eliminar la primera incógnita x de las ecuaciones restantes. Entonces la (nueva) segunda ecuación es usada para eliminar y de la última ecuación. En general, descendemos poco a poco a través del conjunto de ecuaciones, y luego, con una incógnita determinada, avanzamos gradualmente para resolver cada una de las otras incógnitas en sucesión.

Dividiendo cada fila por su coeficiente inicial, vemos que las ecuaciones (3.19) se convierten en

$$\begin{aligned} x + \frac{2}{3}y + \frac{1}{3}z &= \frac{11}{3} \\ x + \frac{3}{2}y + \frac{1}{2}z &= \frac{13}{2} \\ x + y + 4z &= 12 \ . \end{aligned} \tag{3.20}$$

Ahora, usando la primera ecuación, eliminamos x de la segunda y la tercera:

$$\begin{aligned} x + \frac{2}{3}y + \frac{1}{3}z &= \frac{11}{3} \\ \frac{5}{6}y + \frac{1}{6}z &= \frac{17}{6} \\ \frac{1}{3}y + \frac{11}{3}z &= \frac{25}{3} \ , \end{aligned} \tag{3.21}$$

y

$$\begin{aligned} x + \frac{2}{3}y + \frac{1}{3}z &= \frac{11}{3} \\ y + \frac{1}{5}z &= \frac{17}{5} \\ y + 11z &= 25 \ . \end{aligned} \tag{3.22}$$

Repitiendo la técnica, usamos la segunda ecuación para eliminar y a partir de la tercera ecuación:

$$\begin{aligned} x + \frac{2}{3}y + \frac{1}{3}z &= \frac{11}{3} \\ y + \frac{1}{5}z &= \frac{17}{5} \\ 54z &= 108 \ , \end{aligned} \tag{3.23}$$

o

$$z = 2 \ .$$

Finalmente, al reemplazar obtenemos

$$y + \frac{1}{5} \times 2 = \frac{17}{5} \ ,$$

o

$$y = 3 \ .$$

Luego con z e y determinados,

$$x + \frac{2}{3} \times 3 + \frac{1}{3} \times 2 = \frac{11}{3} \ ,$$

y

$$x = 1 \ .$$

La técnica podría parecer no tan elegante como la ecuación (3.17), pero está bien adaptada a los computadores modernos y es más rápida que el tiempo gastado con los determinantes.

Esta técnica de Gauss puede ser usada para convertir un determinante en una forma triangular:

$$D = \begin{vmatrix} a_1 & b_1 & c_1 \\ 0 & b_2 & c_2 \\ 0 & 0 & c_3 \end{vmatrix} \ ,$$

para un determinante de tercer orden cuyos elementos no deben ser confundidos con aquellos en la ecuación (3.3). De esta forma $D = a_1 b_2 c_3$. Para un determinante de n-ésimo orden la evaluación de una forma triangular requiere solamente $n - 1$ multiplicaciones comparadas con las $n!$ requeridas para el caso general.

Una variación de esta eliminación progresiva es conocida como eliminación de Gauss-Jordan. Comenzamos como si fuera el procedimiento de Gauss, pero cada nueva ecuación considerada es usada para eliminar una variable de todas las "otras" ecuaciones, no sólo de aquellas bajo ella. Si hemos usado esta eliminación de Gauss-Jordan, la ecuación (3.23) llegaría a ser

$$\begin{aligned} x + \frac{1}{5} z &= \frac{7}{5} \\ y + \frac{1}{5} z &= \frac{17}{5} \\ z &= 2 \ , \end{aligned} \tag{3.24}$$

usando la segunda ecuación de la ecuación (3.22) para eliminar y de ambas, la primera y tercera ecuaciones. Entonces la tercera ecuación de la ecuación (3.24) es usada para eliminar z de la primera y segunda ecuaciones, dando

$$\begin{aligned} x &= 1 \\ y &= 3 \\ z &= 2 \ , \end{aligned} \tag{3.25}$$

Volveremos a la técnica de Gauss-Jordan cuando invertimos matrices.

Otra técnica disponible para el uso computacional es la técnica de Gauss-Seidel. Cada técnica tiene sus ventajas y desventajas. Los métodos de Gauss y Gauss-Jordan pueden tener problemas de precisión para un determinante grande. Esto también es un problema para la inversión de matrices. El método de Gauss-Seidel, como un método iterativo, puede tener problemas de convergencia.

3.2. Matrices.

El análisis matricial pertenece al álgebra lineal ya que las matrices son operadores o mapas lineales tales como rotaciones. Supongamos, por ejemplo, que rotamos las coordenadas cartesianas de un espacio bidimensional tal que, en notación vectorial,

$$\begin{pmatrix} x_1' \\ x_2' \end{pmatrix} = \begin{pmatrix} x_1 \cos\varphi + x_2 \operatorname{sen}\varphi \\ -x_1 \operatorname{sen}\varphi + x_2 \cos\varphi \end{pmatrix} = \left(\textstyle\sum_j a_{ij} x_j\right) \ . \tag{3.26}$$

Etiquetamos el arreglo de elementos a_{ij} por la matriz A de 2×2 consistente de dos filas y dos columnas, además, consideramos los vectores x, x' como matrices de 2×1. Tomemos la suma de productos de la ecuación (3.26) como una definición de la multiplicación matricial que involucra el producto escalar de cada uno de los vectores fila de A con el vector columna x. Así en notación matricial la ecuación (3.26) se convierte en

$$x' = \mathsf{A}x \ . \tag{3.27}$$

Para extender esta definición de multiplicación de una matriz por un vector columna a el producto de dos matrices de 2×2, consideremos la rotación de coordenada seguida por una segunda rotación dada por la matriz B tal que

$$x'' = \mathsf{B}x' \ . \tag{3.28}$$

Por componentes

$$x_i'' = \sum_j b_{ij} x_j' = \sum_j b_{ij} \sum_k a_{jk} x_k = \sum_k \left(\sum_j b_{ij} a_{jk} \right) x_k \ . \tag{3.29}$$

La suma sobre j es la multiplicación matricial definiendo una matriz $\mathsf{C} = \mathsf{BA}$ tal que

$$x_i'' = \sum_k c_{ik} x_k \ , \tag{3.30}$$

o $x'' = \mathsf{C}x$ en notación matricial. Nuevamente, esta definición involucra el producto escalar de vectores filas de B con vectores columnas de A. Esta definición de multiplicación matricial se puede generalizar a matrices de $m \times n$ y es útil, realmente "su utilidad es la justificación de su existencia". La interpretación física es que el producto matricial de dos matrices, BA, es la rotación que conduce del sistema sin prima directamente al sistema de coordenadas con doble prima. Antes de pasar a la definición formal, podemos notar que el operador A está descrito

por sus efectos sobre las coordenadas o vectores base. Los elementos de matriz a_{ij} constituyen una representación del operador, una representación que depende de la elección de una base.

El caso especial donde una matriz tiene una columna y n filas es llamada un vector columna, $|x\rangle$, con componentes x_i, $i = 1, 2, \ldots, n$. Si A es una matriz de $n \times n$, $|x\rangle$ es un vector columna de n componentes, $\mathsf{A}|x\rangle$ está definida como en la ecuación (3.27) y (3.26). Similarmente, si una matriz tiene una fila y n columnas, es llamada un vector fila, $\langle x|$ con componentes x_i, $i = 1, 2, \ldots n$. Claramente, $\langle x|$ resulta de $|x\rangle$ por el intercambio de filas y columnas, una operación matricial llamada *transposición*, y para cualquier matriz A, $\tilde{\mathsf{A}}$ es llamada [2] la transpuesta de A con elementos de matriz $(\tilde{\mathsf{A}})_{ik} = A_{ki}$. Transponiendo un producto de matrices AB se invierte el orden y da BA; similarmente, $\mathsf{A}|x\rangle$ se transpone como $\langle x|\mathsf{A}$. El producto escalar toma la forma $\langle x|y\rangle$.

3.2.1. Definiciones básicas.

Una matriz puede ser definida como una arreglo cuadrado o rectangular de números o funciones que obedecen ciertas leyes. Esto es una extensión perfectamente lógica de los conceptos matemáticos familiares. En aritmética tratamos con números simples. En la teoría de variable compleja tratamos con pares ordenados de números, $(1, 2) = 1 + 2i$, en el cual el orden es importante. Ahora consideremos números (o funciones) ordenados en un arreglo cuadrado o rectangular. Por conveniencia en el trabajo posterior los números son distinguidos por dos subíndices, el primero indica la fila (horizontal) y el segundo indica la columna (vertical) en la cual aparecen los números. Por ejemplo, a_{13} es el elemento de matriz en la primera fila y tercera columna. De este modo, si A es una matriz con m filas y n columnas,

$$\mathsf{A} = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix} . \tag{3.31}$$

Quizás el hecho más importante a notar es que los elementos a_{ij} no están combinados unos con otros. Una matriz no es un determinante. Es un arreglo ordenado de números, no un simple número.

La matriz A hasta ahora es sólo un arreglo de números que tiene las propiedades que le asignamos. Literalmente, esto significa construir una nueva forma de matemáticas. Postulamos que las matrices A, B y C, con elementos a_{ij}, b_{ij} y c_{ij}, respectivamente, combinan de acuerdo a las siguientes reglas.

3.2.2. Igualdad.

Matriz A = Matriz B si y sólo si $a_{ij} = b_{ij}$ para todos los valores de i y j. Esto, por supuesto, requiere que A y B sean cada uno arreglos de $m \times n$ (m filas y n columnas).

[2] Algunos textos denotan A transpuesta por A^T.

3.2.3. Suma.

$\mathsf{A}+\mathsf{B}=\mathsf{C}$ si y sólo si $a_{ij}+b_{ij}=c_{ij}$ para todos los valores de i y j, los elementos se combinan de acuerdo a las leyes del álgebra lineal (o aritmética si hay números simples). Esto significa que $\mathsf{A}+\mathsf{B}=\mathsf{B}+\mathsf{A}$, la conmutación. También, se satisface la ley de asociatividad $(\mathsf{A}+\mathsf{B})+\mathsf{C}=\mathsf{A}+(\mathsf{B}+\mathsf{C})$. Si todos los elementos son cero, la matriz es llamada matriz nula y se denota por $\mathsf{0}$. Para todo A,

$$\mathsf{A}+\mathsf{0}=\mathsf{0}+\mathsf{A}=\mathsf{A}\ ,$$

con

$$\mathsf{0}=\begin{pmatrix}0&0&\cdots&0\\0&0&\cdots&0\\\vdots&\vdots&\ddots&\vdots\\0&0&\cdots&0\end{pmatrix}\ . \tag{3.32}$$

Tal que las matrices de $m\times n$ forman un espacio lineal con respecto a la suma y la resta.

3.2.4. Multiplicación (por un escalar).

La multiplicación de la matriz A por una cantidad escalar α está definida como

$$\alpha\mathsf{A}=(\alpha\mathsf{A})\ , \tag{3.33}$$

en la cual los elementos de $\alpha\mathsf{A}$ son αa_{ij}; esto es, cada elemento de la matriz A es multiplicado por el factor escalar. Esto contrasta con el comportamiento de los determinantes en el cual el factor α multiplica solamente una columna o una fila y no cada elemento del determinante. Una consecuencia de esta multiplicación por escalar es que

$$\alpha\mathsf{A}=\mathsf{A}\alpha\ ,\quad \text{conmutación.} \tag{3.34}$$

3.2.5. Multiplicación (multiplicación matricial) producto interno.

$$\mathsf{AB}=\mathsf{C}\quad \text{si y sólo si}\quad c_{ij}=\sum_k a_{ik}b_{kj}\ . \tag{3.35}$$

Los elementos i y j de C están formados como un producto escalar de la i-ésima fila de A con el j-ésima columna de B (el cual demanda que A tenga el mismo número de columnas como B tiene de filas). El índice mudo k toma los valores $1,2,\ldots,n$ en sucesión, esto es,

$$c_{ij}=a_{i1}b_{1j}+a_{i2}b_{2j}+a_{i3}b_{3j}\ , \tag{3.36}$$

para $n=3$. Obviamente, el índice mudo k puede ser reemplazado por algún otro símbolo que no esté en uso sin alterar la ecuación (3.35). Quizás la situación puede ser aclarada afirmando que la ecuación (3.35) defina el método de combinar ciertas matrices. Este método de combinación, es llamado multiplicación matricial. Para ilustrar, consideremos dos matrices (matrices de Pauli)

$$\sigma_1=\begin{pmatrix}0&1\\1&0\end{pmatrix}\quad \text{y}\quad \sigma_3=\begin{pmatrix}1&0\\0&-1\end{pmatrix}\ . \tag{3.37}$$

El elemento $_{11}$ del producto, $(\sigma_1\sigma_3)_{11}$ está dado por la suma de productos de elementos de la primera fila de σ_1 con el correspondiente elemento de la primera columna de σ_3:

$$(\sigma_1\sigma_3)_{ij} = \sigma_{1i1}\sigma_{31j} + \sigma_{1i2}\sigma_{32j} \ .$$

Una aplicación directa de la multiplicación de matrices muestra que

$$\sigma_3\sigma_1 = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} \tag{3.38}$$

y por la ecuación (3.35)

$$\sigma_1\sigma_3 = -\sigma_3\sigma_1 \ . \tag{3.39}$$

Excepto en casos especiales, la multiplicación de matrices no es conmutativa.[3]

$$\mathsf{AB} \neq \mathsf{BA} \ . \tag{3.40}$$

Sin embargo, de la definición de multiplicación de matrices podemos mostrar que se mantiene una ley de asociatividad, $(\mathsf{AB})\mathsf{C} = \mathsf{A}(\mathsf{BC})$. También se satisface una ley de distributividad, $\mathsf{A}(\mathsf{B}+\mathsf{C}) = \mathsf{AB} + \mathsf{AC}$. La matriz unidad tiene elementos δ_{ij}, la delta de Kronecker, y la propiedad de que $\mathsf{1A} = \mathsf{A1} = \mathsf{A}$ para toda A,

$$\mathsf{1} = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix} \ . \tag{3.41}$$

Notamos que es posible que el producto de dos matrices sea una matriz nula sin ser ninguna de ellas una matriz nula. Por ejemplo, si

$$\mathsf{A} = \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix} \quad \text{y} \quad \mathsf{B} = \begin{pmatrix} 1 & 0 \\ -1 & 0 \end{pmatrix} \ .$$

$\mathsf{AB} = \mathsf{0}$. Esto difiere de la multiplicación de números reales o complejos los cuales forman un *campo*, mientras que las estructura aditiva y multiplicativa de las matrices es llamada *anillo* por los matemáticos.

Si A es una matriz de $n \times n$ con determinante $|\mathsf{A}| \neq 0$, tiene una única inversa A^{-1}, tal que $\mathsf{A}\mathsf{A}^{-1} = \mathsf{A}^{-1}\mathsf{A} = \mathsf{1}$. Si B es también una matriz de $n \times n$ con inversa B^{-1}, luego el producto de AB tiene la inversa

$$(\mathsf{AB})^{-1} = \mathsf{B}^{-1}\mathsf{A}^{-1} \ , \tag{3.42}$$

ya que $\mathsf{ABB}^{-1}\mathsf{A}^{-1} = \mathsf{1} = \mathsf{B}^{-1}\mathsf{A}^{-1}\mathsf{AB}$.

El *teorema del producto* dice que el determinante de un producto, $|\mathsf{AB}|$, de dos matrices de $n \times n$, A y B, es igual al producto de los determinantes, $|\mathsf{A}||\mathsf{B}|$, uniendo matrices con determinantes. El anterior teorema puede ser fácilmente probado.

[3]La pérdida de la propiedad conmutativa es descrita por el conmutador $[\mathsf{A},\mathsf{B}] = \mathsf{AB} - \mathsf{BA}$. La no conmutatividad se expresa por $[\mathsf{A},\mathsf{B}] \neq 0$.

3.2.6. Producto directo.

Un segundo procedimiento para multiplicar matrices es conocido como el tensor producto directo o de Kronecker. Si A es una matriz de $m \times m$ y B una matriz de $n \times n$, el producto directo es

$$\mathsf{A} \otimes \mathsf{B} = \mathsf{C} \ . \tag{3.43}$$

C es una matriz de $mn \times mn$ con elementos

$$C_{\alpha\beta} = A_{ij} B_{kl} \ , \tag{3.44}$$

con

$$\alpha = n(i-1) + k \ , \qquad \beta = n(j-1) + l \ .$$

Por ejemplo, si A y B ambas son matrices de 2×2,

$$\begin{aligned} \mathsf{A} \otimes \mathsf{B} &= \begin{pmatrix} a_{11}\mathsf{B} & a_{12}\mathsf{B} \\ a_{21}\mathsf{B} & a_{22}\mathsf{B} \end{pmatrix} \\ &= \begin{pmatrix} a_{11}b_{11} & a_{11}b_{12} & a_{12}b_{11} & a_{12}b_{12} \\ a_{11}b_{21} & a_{11}b_{22} & a_{12}b_{21} & a_{12}b_{22} \\ a_{21}b_{11} & a_{21}b_{12} & a_{22}b_{11} & a_{22}b_{12} \\ a_{21}b_{21} & a_{21}b_{22} & a_{22}b_{21} & a_{22}b_{22} \end{pmatrix} \ . \end{aligned} \tag{3.45}$$

El producto directo es asociativo, pero no conmutativo. Como un ejemplo de producto directo, están las matrices de Dirac las que pueden ser desarrolladas como productos directos de las matrices de Pauli y de la matriz unidad. Otros ejemplos aparecen en la construcción de grupos, en teoría de grupos y en espacios de Hilbert, en teoría cuántica.

El producto directo definido aquí es algunas veces llamado la forma *standard* y es denotado por $\otimes$. Otros tres tipos de producto directo de matrices existen como posibilidades o curiosidades matemáticas pero tienen muy poca o ninguna aplicación en Física Matemática.

3.2.7. Matrices diagonales.

Un tipo especial muy importante de matrices es la matriz cuadrada, en la cual todos los elementos no diagonales son cero. Específicamente, si una matriz A de 3×3 es diagonal,

$$\mathsf{A} = \begin{pmatrix} a_{11} & 0 & 0 \\ 0 & a_{22} & 0 \\ 0 & 0 & a_{33} \end{pmatrix} \ .$$

Una interpretación física de tales matrices diagonales y el método de reducir matrices a esta forma diagonal son considerados en la sección 3.5. Aquí nos limitamos a notar la importante propiedad de que la multiplicación de matrices es conmutativa, $\mathsf{AB} = \mathsf{BA}$, si A y B son cada una diagonales.

3.2.8. Traza.

En cualquier matriz cuadrada la suma de los elementos diagonales es llamada la *traza*. Claramente la traza es una operación lineal:

$$\text{traza}(\mathsf{A}-\mathsf{B}) = \text{traza}(\mathsf{A}) - \text{traza}(\mathsf{B}) \ .$$

Una de sus interesantes y útiles propiedades es que la traza de un producto de dos matrices A y B es independiente del orden de la multiplicación:

$$\begin{aligned} \text{traza}(\mathsf{AB}) &= \sum_i (\mathsf{AB})_{ii} = \sum_i \sum_j a_{ij} b_{ji} \\ &= \sum_i \sum_j b_{ji} a_{ij} = \sum_j (\mathsf{BA})_{jj} \\ &= \text{traza}(\mathsf{BA}) \ . \end{aligned} \tag{3.46}$$

Esto se mantiene aún cuando $\mathsf{AB} \neq \mathsf{BA}$. La ecuación (3.46) significa que la traza de cualquier conmutador, $[\mathsf{A},\mathsf{B}] = \mathsf{AB} - \mathsf{BA}$, es cero. De la ecuación (3.46) obtenemos

$$\text{traza}(\mathsf{ABC}) = \text{traza}(\mathsf{BCA}) = \text{traza}(\mathsf{CAB}) \ ,$$

lo cual muestra que la traza es invariante bajo permutaciones cíclicas de la matriz en un producto.

Para una matriz simétrica o para una matriz Hermítica compleja, la traza es la suma, y el determinante es el producto, de sus autovalores, y ambos son coeficientes del polinomio característico. La traza tendrá una función similar para las matrices, como la tiene la ortogonalidad para los vectores y funciones.

En términos de tensores, la traza es una contracción y como el tensor de segundo orden contraído es un escalar (invariante), esta también lo es.

Las matrices son usadas ampliamente para representar los elementos de grupos. La traza de las matrices representando los elementos de grupo es conocido, en teoría de grupos, como el *carácter*. La razón de este nombre especial y especial atención es que mientras las matrices pueden variar la traza o carácter, este se mantiene invariante.

3.2.9. Inversión de matriz.

Al comienzo de esta sección la matriz A fue presentada como la representación de un operador que (linealmente) transforma los ejes de coordenadas. Una rotación podría ser un ejemplo de tal transformación lineal. Ahora buscaremos la transformación inversa A^{-1} que restablecerá los ejes de coordenadas originales. Esto significa, ya sea como una ecuación matricial o de operador[4],

$$\mathsf{A}\mathsf{A}^{-1} = \mathsf{A}^{-1}\mathsf{A} = 1 \ . \tag{3.47}$$

Podemos probar (ejercicio) que

$$a_{ij}^{-1} = \frac{C_{ji}}{|\,\mathsf{A}\,|} \ , \tag{3.48}$$

[4]Aquí y a través de todo el capítulo nuestras matrices tienen rango finito.

con la suposición que el determinante de A ($|\mathsf{A}|$) $\neq 0$. Si es cero, etiquetaremos a A como singular. No existe la inversa. Como fue explicado en la sección 3.1, esta forma con determinante es *totalmente inapropiado para el trabajo numérico* con grandes matrices.

Hay una amplia variedad de técnicas alternativas. Una de las mejores y más comúnmente usada es la técnica de inversión de matrices de Gauss-Jordan. La teoría está basada en los resultados que muestran que existen matrices M_L tales que el producto $\mathsf{M}_L\mathsf{A}$ será A, pero con:

a. una fila multiplicada por una constante, o

b. una fila reemplazada por la fila original menos un múltiplo de otra fila, o

c. filas intercambiadas.

Otras matrices M_R operando sobre la derecha de ($\mathsf{A}\mathsf{M}_R$) pueden llevar a las mismas operaciones sobre las columnas de A.

Esto significa que las filas y las columnas de la matriz pueden ser alteradas (por multiplicación de matrices) como si estuviéramos tratando con determinantes, así podemos aplicar las técnicas de eliminación de Gauss-Jordan a los elementos de matriz. Por tanto existe una matriz M_L (o M_R) tal que[5]

$$\mathsf{M}_L\mathsf{A} = \mathsf{1} \ . \tag{3.49}$$

La $\mathsf{M}_L = \mathsf{A}^{-1}$. Determinamos M_L realizando las operaciones de eliminación idénticas sobre la matriz unidad. Luego

$$\mathsf{M}_L\mathsf{1} = \mathsf{M}_L \ . \tag{3.50}$$

Para clarificar ésto consideremos un ejemplo específico.

Deseamos invertir la matriz

$$\mathsf{A} = \begin{pmatrix} 3 & 2 & 1 \\ 2 & 3 & 1 \\ 1 & 1 & 4 \end{pmatrix} \ . \tag{3.51}$$

Por conveniencia escribimos A y $\mathsf{1}$ lado a lado realizando operaciones idénticas sobre cada una de ellas

$$\begin{pmatrix} 3 & 2 & 1 \\ 2 & 3 & 1 \\ 1 & 1 & 4 \end{pmatrix} \qquad \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \ . \tag{3.52}$$

Para ser sistemáticos, multiplicamos cada fila para obtener $a_{k1} = 1$,

$$\begin{pmatrix} 1 & \frac{2}{3} & \frac{1}{3} \\ 1 & \frac{3}{2} & \frac{1}{2} \\ 1 & 1 & 4 \end{pmatrix} \qquad \begin{pmatrix} \frac{1}{3} & 0 & 0 \\ 0 & \frac{1}{2} & 0 \\ 0 & 0 & 1 \end{pmatrix} \ . \tag{3.53}$$

Restando la primera fila de la segunda y tercera, obtenemos

$$\begin{pmatrix} 1 & \frac{2}{3} & \frac{1}{3} \\ 0 & \frac{5}{6} & \frac{1}{6} \\ 0 & \frac{1}{3} & \frac{11}{3} \end{pmatrix} \qquad \begin{pmatrix} \frac{1}{3} & 0 & 0 \\ -\frac{1}{3} & \frac{1}{2} & 0 \\ -\frac{1}{3} & 0 & 1 \end{pmatrix} \ . \tag{3.54}$$

[5]Recordemos que $\det(\mathsf{A}) \neq 0$.

Entonces dividimos la segunda fila (de ambas matrices) por 5/6 y sustrayéndola 2/3 veces de la primera fila, y 1/3 veces de la tercera fila. Los resultados para ambas matrices son

$$\begin{pmatrix} 1 & 0 & \frac{1}{5} \\ 0 & 1 & \frac{1}{5} \\ 0 & 0 & \frac{18}{5} \end{pmatrix} \qquad \begin{pmatrix} \frac{3}{5} & -\frac{2}{5} & 0 \\ -\frac{2}{5} & \frac{3}{5} & 0 \\ -\frac{1}{5} & -\frac{1}{5} & 1 \end{pmatrix} . \tag{3.55}$$

Dividimos la tercera fila (de ambas matrices) por 18/5. Luego como último paso 1/5 veces la tercera fila es sustraída de cada una de las dos primeras filas (de ambas matrices). Nuestro par final es

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \qquad \begin{pmatrix} \frac{11}{8} & -\frac{7}{18} & -\frac{1}{18} \\ -\frac{7}{18} & \frac{11}{18} & -\frac{1}{18} \\ -\frac{1}{18} & -\frac{1}{18} & \frac{5}{18} \end{pmatrix} . \tag{3.56}$$

El chequeo es multiplicar la original A por la calculada A^{-1} para ver si realmente obtuvimos la matriz unidad $\mathsf{1}$.

Como con la solución de Gauss-Jordan de ecuaciones algebraicas simultáneas, esta técnica está bien adaptada para computadores.

3.3. Matrices ortogonales.

El espacio de tres dimensiones ordinario puede ser descrito con las coordenadas cartesianas (x_1, x_2, x_3). Consideremos un segundo conjunto de coordenadas cartesianas (x'_1, x'_2, x'_3) cuyo origen y sentido coinciden con el primero pero su orientación es diferente (figura 3.1).

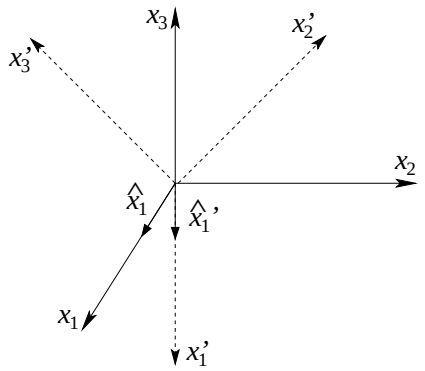


Figura 3.1: Sistemas de coordenadas cartesianos.

Podemos decir que el sistema de ejes prima ha sido rotado respecto al inicial sistema de coordenadas sin prima. Ya que esta rotación es una operación lineal, esperamos una ecuación matricial que relacione la base con primas con la sin primas.

3.3.1. Cosenos directores.

Un vector unitario a lo largo del eje x_1' ($\hat{x}_1'$) puede ser resuelto en sus componentes a lo largo de los ejes x_1, x_2 y x_3 por las usuales técnicas de proyección.

$$\hat{x}_1' = \hat{x}_1 \cos(x_1', x_1) + \hat{x}_2 \cos(x_2', x_2) + \hat{x}_3 \cos(x_3', x_3) \ . \tag{3.57}$$

Por conveniencia estos cosenos, los cuales son los cosenos directores, son etiquetados

$$\begin{aligned} \cos(x_1', x_1) &= \hat{x}_1' \cdot \hat{x}_1 = a_{11} \ , \\ \cos(x_1', x_2) &= \hat{x}_1' \cdot \hat{x}_2 = a_{12} \ , \\ \cos(x_1', x_3) &= \hat{x}_1' \cdot \hat{x}_3 = a_{13} \ . \end{aligned} \tag{3.58}$$

Continuando, tenemos

$$\begin{aligned} \cos(x_2', x_1) &= \hat{x}_2' \cdot \hat{x}_1 = a_{21} \ , \qquad (a_{21} \neq a_{12}) \ , \\ \cos(x_2', x_2) &= \hat{x}_2' \cdot \hat{x}_2 = a_{22} \ , \qquad \text{y así sucesivamente.} \end{aligned} \tag{3.59}$$

Ahora la ecuación (3.57) puede ser reescrita como

$$\hat{x}_1' = \hat{x}_1 a_{11} + \hat{x}_2 a_{12} + \hat{x}_3 a_{13}$$

y también

$$\begin{aligned} \hat{x}_2' &= \hat{x}_1 a_{21} + \hat{x}_2 a_{22} + \hat{x}_3 a_{23} \\ \hat{x}_3' &= \hat{x}_1 a_{31} + \hat{x}_2 a_{32} + \hat{x}_3 a_{33} \ . \end{aligned} \tag{3.60}$$

También podemos ir de la otra manera resolviendo $\hat{x}_1$, $\hat{x}_2$ y $\hat{x}_3$ en sus componentes en el sistema con primas. Entonces

$$\begin{aligned} \hat{x}_1 &= \hat{x}_1' a_{11} + \hat{x}_2' a_{21} + \hat{x}_3' a_{31} \\ \hat{x}_2 &= \hat{x}_1' a_{12} + \hat{x}_2' a_{22} + \hat{x}_3' a_{32} \\ \hat{x}_3 &= \hat{x}_1' a_{13} + \hat{x}_2' a_{23} + \hat{x}_3' a_{33} \ . \end{aligned} \tag{3.61}$$

3.3.2. Aplicaciones a vectores.

Si consideramos un vector cuyas componentes son funciones de la posición, entonces

$$\begin{aligned} \vec{V}(x_1, x_2, x_3) &= \hat{x}_1 V_1 + \hat{x}_2 V_2 + \hat{x}_3 V_3 \\ \vec{V}'(x_1', x_2', x_3') &= \hat{x}_1' V_1' + \hat{x}_2' V_2' + \hat{x}_3' V_3' \ , \end{aligned} \tag{3.62}$$

ya que el punto puede ser dado en cualquiera de los dos sistema de coordenadas (x_1, x_2, x_3) o (x_1', x_2', x_3'). Notemos que $\vec{V} = \vec{V}'$, es decir, son geométricamente el mismo vector (pero con diferentes componentes). Si los ejes de coordenadas son rotados, el vector se mantiene fijo. Usando la ecuación (3.60) para eliminar $\hat{x}_1$, $\hat{x}_2$, $\hat{x}_3$, podemos separar la ecuación (3.62) en tres ecuaciones escalares

$$\begin{aligned} V_1' &= a_{11} V_1 + a_{12} V_2 + a_{13} V_3 \\ V_2' &= a_{21} V_1 + a_{22} V_2 + a_{23} V_3 \\ V_3' &= a_{31} V_1 + a_{32} V_2 + a_{33} V_3 \ . \end{aligned} \tag{3.63}$$

En particular, estas relaciones se mantendrán para las coordenadas de un punto (x_1, x_2, x_3) y (x'_1, x'_2, x'_3), dando

$$\begin{aligned} x'_1 &= a_{11}x_1 + a_{12}x_2 + a_{13}x_3 \\ x'_2 &= a_{21}x_1 + a_{22}x_2 + a_{23}x_3 \\ x'_3 &= a_{31}x_1 + a_{32}x_2 + a_{33}x_3 \ , \end{aligned} \tag{3.64}$$

y similarmente para las coordenadas primas. En esta notación el conjunto de tres ecuaciones (3.64) pueden ser escritas como

$$x'_i = \sum_{j=1}^{3} a_{ij}x_j \ , \tag{3.65}$$

donde i toma los valores 1, 2 y 3 y el resultado son tres ecuaciones separadas.

De la ecuación anterior podemos derivar interesante información sobre los a_{ij}, los cuales describen la orientación del sistema de coordenadas (x'_1, x'_2, x'_3) relativa al sistema (x_1, x_2, x_3). La distancia respecto al origen es la misma en ambos sistemas. Elevando al cuadrado,

$$\begin{aligned} \sum_i x_i^2 &= \sum_i {x'_i}^2 \\ &= \sum_i \left(\sum_j a_{ij}x_j \right) \left(\sum_k a_{ik}x_k \right) \\ &= \sum_{j,k} x_j x_k \sum_i a_{ij}a_{ik} \ . \end{aligned} \tag{3.66}$$

Esto sólo puede ser cierto para todos los puntos si y sólo si

$$\sum_i a_{ij}a_{ik} = \delta_{jk} \ , \qquad j,k = 1,2,3 \ . \tag{3.67}$$

La ecuación (3.67) es una consecuencia de requerir que la longitud permanezca constante (invariante) bajo rotaciones del sistema de coordenadas, y es llamada *condición de ortogonalidad*. Los a_{ij} escritos como una matriz A, forman una matriz ortogonal. Notemos que la ecuación (3.67) no es una multiplicación matricial.

En notación matricial la ecuación (3.65) llega a ser

$$|x'\rangle = \mathsf{A}|x\rangle \ . \tag{3.68}$$

3.3.3. Condiciones de ortogonalidad, caso bidimensional.

Podemos ganar un mejor entendimiento de los a_{ij} y de la condición de ortogonalidad considerando con detalle rotaciones en dos dimensiones. Esto lo podemos pensar como un sistema tridimensional con los ejes x_1 y x_2 rotados respecto a x_3. De la figura 3.2,

$$\begin{aligned} x'_1 &= x_1 \cos\varphi + x_2 \operatorname{sen}\varphi \ , \\ x'_2 &= -x_1 \operatorname{sen}\varphi + x_2 \cos\varphi \ . \end{aligned} \tag{3.69}$$

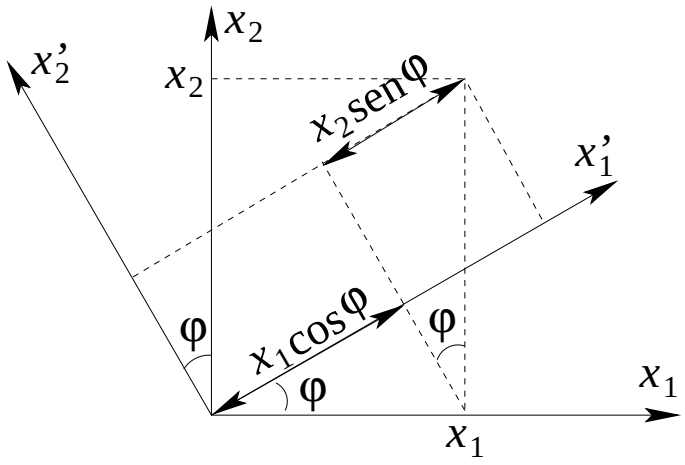


Figura 3.2: Sistemas de coordenadas rotados en dos dimensiones.

Por lo tanto por la ecuación (3.68)

$$\mathsf{A} = \begin{pmatrix} \cos\varphi & \operatorname{sen}\varphi \\ -\operatorname{sen}\varphi & \cos\varphi \end{pmatrix} \ . \tag{3.70}$$

Notemos que A se reduce a la matriz unidad para $\varphi = 0$. La rotación cero significa que nada ha cambiado. Es claro a partir de la figura 3.2 que

$$\begin{aligned} a_{11} &= \cos\varphi = \cos(x_1', x_1) \ , \\ a_{12} &= \operatorname{sen}\varphi = \cos\left(\frac{\pi}{2} - \varphi\right) = \cos(x_1', x_2) \ , \quad \text{y así sucesivamente,} \end{aligned} \tag{3.71}$$

de este modo identificamos los elementos de matriz a_{ij} con los cosenos directores. La ecuación (3.67), la condición de ortogonalidad, llega a ser

$$\begin{aligned} \operatorname{sen}^2\varphi + \cos^2\varphi = 1 \ , \\ \operatorname{sen}\varphi\cos\varphi - \operatorname{sen}\varphi\cos\varphi = 0 \ . \end{aligned} \tag{3.72}$$

La extensión a tres dimensiones (rotación de las coordenadas a lo largo del eje z en un ángulo φ en el sentido contrario a los punteros del reloj) es simplemente

$$\mathsf{A} = \begin{pmatrix} \cos\varphi & \operatorname{sen}\varphi & 0 \\ -\operatorname{sen}\varphi & \cos\varphi & 0 \\ 0 & 0 & 1 \end{pmatrix} \ . \tag{3.73}$$

El $a_{33} = 1$ expresa el hecho que $x_3' = x_3$, ya que la rotación ha sido en torno al eje x_3. Los ceros garantizan que x_1' y x_2' no dependen de x_3 y que x_3' no depende de x_1 y x_2. En un lenguaje más sofisticado, x_1 y x_2 se extienden sobre un subespacio invariante, mientras que x_3 forma un subespacio invariante por sí solo. La forma de A es reducible. La ecuación (3.73) da una posible descomposición.

3.3.4. Matriz inversa A^{-1}.

Volviendo a la matriz de transformación general A, la matriz inversa A^{-1} es definida tal que

$$|x\rangle = \mathsf{A}^{-1}|x'\rangle \ . \tag{3.74}$$

Esto es, A^{-1} describe el inverso de la rotación dada por A y retorna el sistema de coordenadas a su posición original. Simbólicamente, las ecuaciones (3.68) y (3.74) combinadas dan

$$|x\rangle = \mathsf{A}^{-1}\mathsf{A}|x\rangle \ , \tag{3.75}$$

y ya que $|x\rangle$ es arbitrario,

$$\mathsf{A}^{-1}\mathsf{A} = 1 \ , \tag{3.76}$$

la matriz unidad. Similarmente,

$$\mathsf{A}\mathsf{A}^{-1} = 1 \ . \tag{3.77}$$

usando las ecuaciones (3.68) y (3.74) y eliminando $|x'\rangle$ en vez de $|x\rangle$.

3.3.5. Matriz transpuesta, $\tilde{\mathsf{A}}$.

Podemos determinar los elementos de nuestra postulada matriz inversa A^{-1} empleando la condición de ortogonalidad. La ecuación (3.67), la condición de ortogonalidad, no está de acuerdo con nuestra definición de multiplicación matricial, pero la podemos definir de acuerdo a una nueva matriz $\tilde{\mathsf{A}}$ tal que

$$\tilde{a}_{ji} = a_{ij} \ . \tag{3.78}$$

La ecuación (3.67) llega a ser

$$\tilde{\mathsf{A}}\mathsf{A} = 1 \ . \tag{3.79}$$

Ésta es una reformulación de la condición de ortogonalidad y puede ser tomada como una definición de ortogonalidad. Multiplicando (3.79) por A^{-1} por la derecha y usando la ecuación (3.77), tenemos

$$\tilde{\mathsf{A}} = \mathsf{A}^{-1} \ . \tag{3.80}$$

Este importante resultado, que la inversa es igual a la transpuesta, se mantiene sólo para matrices ortogonales y puede ser tomado como una reformulación de la condición de ortogonalidad.

Multiplicando la ecuación (3.80) por A por la izquierda, obtenemos

$$\mathsf{A}\tilde{\mathsf{A}} = 1 \ , \tag{3.81}$$

o

$$\sum_i a_{ji}a_{ki} = \delta_{jk} \ , \tag{3.82}$$

lo cual es otra forma más de la condición de ortogonalidad.

Resumiendo, la condición de ortogonalidad puede ser enunciada de varias maneras equivalentes:

$$\sum_i a_{ij}a_{ik} = \delta_{jk} \tag{3.83a}$$

$$\sum_i a_{ji}a_{ki} = \delta_{jk} \tag{3.83b}$$

$$\tilde{\mathsf{A}}\mathsf{A} = \mathsf{A}\tilde{\mathsf{A}} = 1 \tag{3.83c}$$

$$\tilde{\mathsf{A}} = \mathsf{A}^{-1} \ . \tag{3.83d}$$

Cualquiera de estas relaciones es condición necesaria y suficiente para que A sea ortogonal.

Es posible ahora ver y entender porqué el nombre *ortogonal* es apropiado para estas matrices. Tenemos la forma general

$$\mathsf{A} = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \ ,$$

de una matriz de cosenos directores en la cual a_{ij} es el coseno del ángulo entre x_i' y x_j. Por lo tanto, a_{11}, a_{12}, a_{13}, son los cosenos directores de x_1' relativo a x_1, x_2, x_3. Estos tres elementos de A definen una unidad de longitud a lo largo de x_1', esto es, un vector unitario $\hat{x}_1'$,

$$\hat{x}_1' = \hat{x}_1 a_{11} + \hat{x}_2 a_{12} + \hat{x}_3 a_{13} \ .$$

La relación de ortogonalidad (ecuación (3.82)) es simplemente una declaración que los vectores unitarios $\hat{x}_1'$, $\hat{x}_2'$, y $\hat{x}_3'$ son mutuamente perpendiculares u ortogonales. Nuestra matriz de transformación ortogonal, A, transforma un sistema ortogonal en un segundo sistema ortogonal de coordenadas por rotación y/o reflexión.

3.3.6. Ángulos de Euler.

Nuestra matriz de transformación A contiene nueve cosenos directores. Claramente, sólo tres de ellos son independientes, la ecuación (3.67) provee seis restricciones. De otra manera, uno puede decir que necesita dos parámetros (θ y φ en coordenadas polares esféricas) para fijar el eje de rotación, más uno adicional para describir la magnitud de la rotación en torno a ese eje. En la formulación Lagrangiana de la mecánica es necesario describir A usando algún conjunto de tres parámetros independientes más que los redundantes cosenos directores. La elección usual de estos parámetros es la de los ángulos de Euler.[6]

El objetivo es describir la orientación de un sistema final rotado (x_1''', x_2''', x_3''') relativo a algún sistema de coordenadas inicial (x_1, x_2, x_3). El sistema final es desarrollado en tres pasos, cada paso involucra una rotación descrita por un ángulo de Euler (figura 3.3):

1. Los ejes x_1', x_2', y x_3' son rotados respecto al eje x_3 en un ángulo α en el sentido horario relativo a x_1, x_2 y x_3. (Los ejes x_3 y x_3' coinciden.)

[6]No hay una única manera de definir los ángulos de Euler. Usamos la elección usual en Mecánica Cuántica de momento angular.

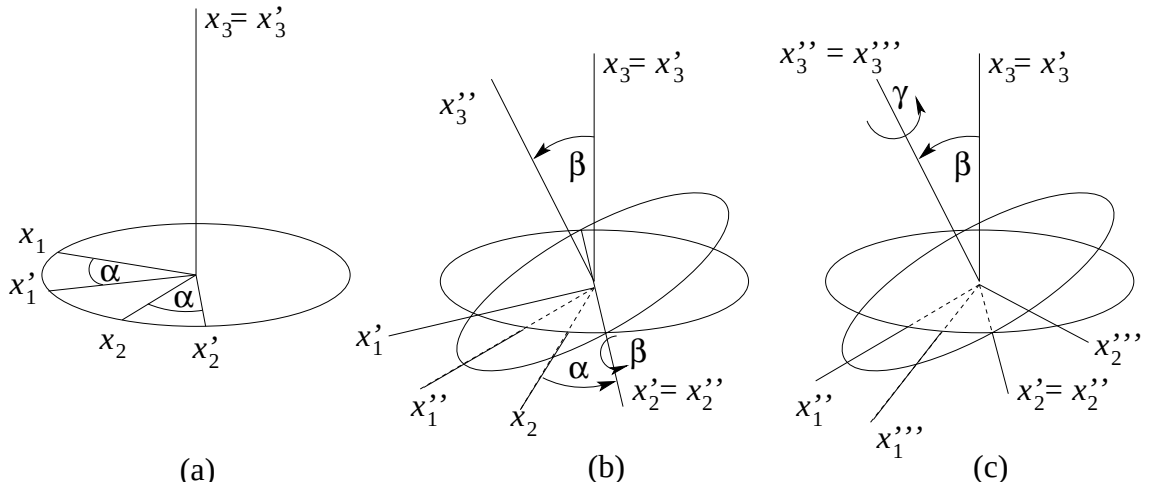


Figura 3.3: (a) Rotación respecto al eje x_3 en un ángulo α; (b) Rotación respecto a un eje x_2' en un ángulo β; (c) Rotación respecto a un eje x_3'' en un ángulo γ.

2. los ejes x_1'', x_2'', y x_3'' son rotados respecto al eje x_2' en un ángulo β en el sentido horario relativo a x_1', x_2' y x_3'. (Los ejes x_2' y x_2'' coinciden.)

3. la tercera y final rotación es en un ángulo γ en sentido horario respecto al eje x_3'' produciendo el sistema (x_1''', x_2''', x_3'''). (Los ejes x_3'' y x_3''' coinciden.)

Las tres matrices que describen estas rotaciones son:

$$\mathsf{R}_z(\alpha) = \begin{pmatrix} \cos\alpha & \operatorname{sen}\alpha & 0 \\ -\operatorname{sen}\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix} , \tag{3.84}$$

exactamente como en la ecuación (3.73),

$$\mathsf{R}_y(\beta) = \begin{pmatrix} \cos\beta & 0 & -\operatorname{sen}\beta \\ 0 & 1 & 0 \\ \operatorname{sen}\beta & 0 & \cos\beta \end{pmatrix} \tag{3.85}$$

y

$$\mathsf{R}_z(\gamma) = \begin{pmatrix} \cos\gamma & \operatorname{sen}\gamma & 0 \\ -\operatorname{sen}\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{pmatrix} . \tag{3.86}$$

La rotación total es descrita por el producto matricial triple,

$$\mathsf{A}(\alpha,\beta,\gamma) = \mathsf{R}_z(\gamma)\mathsf{R}_y(\beta)\mathsf{R}_z(\alpha) . \tag{3.87}$$

Notemos el orden: $\mathsf{R}_z(\alpha)$ opera primero, luego $\mathsf{R}_y(\beta)$, y finalmente $\mathsf{R}_z(\gamma)$. La multiplicación da

$$\mathsf{A} = \begin{pmatrix} \cos\gamma\cos\beta\cos\alpha - \operatorname{sen}\gamma\operatorname{sen}\alpha & \cos\gamma\cos\beta\operatorname{sen}\alpha - \operatorname{sen}\gamma\cos\alpha & -\cos\gamma\operatorname{sen}\beta \\ -\operatorname{sen}\gamma\cos\beta\cos\alpha - \cos\gamma\operatorname{sen}\alpha & -\operatorname{sen}\gamma\cos\beta\operatorname{sen}\alpha + \cos\gamma\cos\alpha & \operatorname{sen}\gamma\operatorname{sen}\beta \\ \operatorname{sen}\beta\cos\alpha & \operatorname{sen}\beta\operatorname{sen}\alpha & \cos\beta \end{pmatrix} . \tag{3.88}$$

Comparando $\mathsf{A}(a_{ij})$ con $\mathsf{A}(\alpha,\beta,\gamma)$, elemento por elemento, nos produce los cosenos directores en términos de los ángulos de Euler.

3.3.7. Propiedades de simetría.

Nuestra descripción matricial conduce al grupo de rotaciones $SO(3)$ en el espacio tridimensional $\mathbb{R}^3$, y la descripción en términos de ángulos de Euler de las rotaciones forman una base para desarrollar el grupo de rotaciones. Las rotaciones pueden también ser descritas por el grupo unitario $SU(2)$ en el espacio bidimensional $\mathbb{C}^2$.

La matriz transpuesta es útil en la discusión de las propiedades de simetría. Si

$$\mathsf{A} = \tilde{\mathsf{A}} \ , \qquad a_{ij} = a_{ji} \ , \tag{3.89}$$

la matriz es llamada *simétrica*, mientras que si

$$\mathsf{A} = -\tilde{\mathsf{A}} \ , \qquad a_{ij} = -a_{ji} \ , \tag{3.90}$$

es llamada *antisimétrica*. Los elementos de la diagonal son nulos. Es fácil mostrar que cualquier matriz cuadrada puede ser escrita como la suma de una matriz simétrica y una antisimétrica. Consideremos la identidad

$$\mathsf{A} = \frac{1}{2}\left[\mathsf{A} + \tilde{\mathsf{A}}\right] + \frac{1}{2}\left[\mathsf{A} - \tilde{\mathsf{A}}\right] \ . \tag{3.91}$$

$\mathsf{A} + \tilde{\mathsf{A}}$ es claramente simétrica, mientras que $\mathsf{A} - \tilde{\mathsf{A}}$ es claramente antisimétrica. Esta es la análoga matricial a la ecuación tensorial (2.68).

Hasta ahora hemos interpretado las matrices ortogonales como rotaciones del sistema de coordenadas. Éstas cambian las componentes de un vector fijo. Sin embargo, una matriz ortogonal A puede ser interpretada igualmente bien como una rotación del vector en la dirección opuesta (figura 3.4).

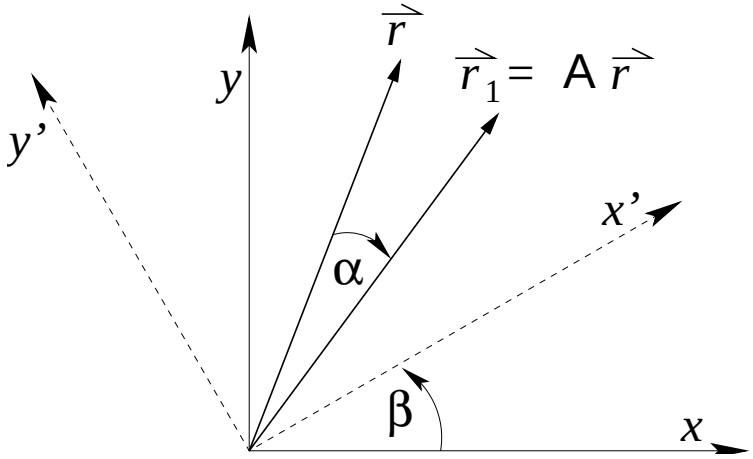


Figura 3.4: Vector fijo con coordenadas rotadas.

Estas dos posibilidades, (1) rotar el vector manteniendo la base fija y (2) rotar la base (en el sentido opuesto) manteniendo el vector fijo.

Supongamos que interpretamos la matriz A como rotar un vector $\vec{r}$ en una nueva posición $\vec{r}_1$, *i.e.*, en un particular sistema de coordenadas tenemos la relación

$$\vec{r}_1 = \mathsf{A}\vec{r} \ . \tag{3.92}$$

Ahora rotemos las coordenadas aplicando una matriz B, la cual rota (x, y, z) en (x', y', z'),

$$\begin{aligned} \vec{r}\,'_1 &= \mathsf{B}\vec{r}_1 = \mathsf{BA}\vec{r} = (\mathsf{A}\vec{r})' \\ &= \mathsf{BA}(\mathsf{B}^{-1}\mathsf{B})\vec{r} \\ &= (\mathsf{BAB}^{-1})\mathsf{B}\vec{r} = (\mathsf{BAB}^{-1})\vec{r}\,' \ . \end{aligned} \tag{3.93}$$

$\mathsf{B}\vec{r}_1$ es justo $\vec{r}_1\,'$ en el nuevo sistema de coordenadas con una interpretación similar se mantiene para $\mathsf{B}\vec{r}$. Ya que en este nuevo sistema $(\mathsf{B}\vec{r})$ es rotado a la posición $(\mathsf{B}\vec{r}_1)$ por la matriz BAB^{-1}.

$$\begin{array}{ccccc} \mathsf{B}\vec{r}_1 & = & (\mathsf{BAB}^{-1}) & \mathsf{B}\vec{r} \\ \downarrow & & \downarrow & \downarrow \\ \vec{r}\,'_1 & = & \mathsf{A}' & \vec{r}\,' \ . \end{array}$$

En el nuevo sistema las coordenadas han sido rotadas por la matriz B, A tiene la forma A', en la cual

$$\mathsf{A}' = \mathsf{BAB}^{-1} \ . \tag{3.94}$$

A' opera en el espacio x', y', z' como A opera en el espacio x, y, z.

La transformación definida por la ecuación (3.94) con B cualquier matriz, no necesariamente ortogonal, es conocida como trasformación de similaridad. Por componentes la ecuación (3.94) llega a ser

$$a'_{ij} = \sum_{k,l} b_{ik} a_{kl} (\mathsf{B}^{-1})_{lj} \ . \tag{3.95}$$

Ahora si B es ortogonal,

$$(\mathsf{B}^{-1})_{lj} = (\tilde{\mathsf{B}})_{lj} = b_{jl} \ , \tag{3.96}$$

y tenemos

$$a'_{ij} = \sum_{k,l} b_{ik} b_{jl} a_{kl} \ . \tag{3.97}$$

La matriz A es la representación de un mapeo lineal en un sistema de coordenadas dado o base. Pero hay direcciones asociadas con A, ejes cristalinos, ejes de simetría en un sólido rotando y etc. tal que la representación depende de la base. La transformación de similaridad muestra exactamente como la representación cambia con un cambio de base.

3.4. Matrices Hermíticas, matrices unitarias.

3.4.1. Definiciones.

Hasta aquí hemos generalmente supuesto que nuestro espacio vectorial es un espacio real y que los elementos de las matrices (la representación de los operadores lineales) son reales. Para muchos cálculos en Física Clásica los elementos de matriz reales serán suficientes. Sin

embargo, en Mecánica Cuántica las variables complejas son inevitables por la forma de las reglas de conmutación básicas (o la ecuación tiempo dependiente de Schödinger). Con esto en mente, generalizamos al caso de matrices con elementos complejos. Para manejar estos elementos, definamos algunas propiedades.

1. Compleja conjugada, A^*, formada por tomar los complejos conjugados ($i \to -i$) de cada elemento, donde $i = \sqrt{-1}$.

2. Adjunta, $\mathsf{A}^\dagger$, formada por transponer A^*,

$$\mathsf{A}^\dagger = \widetilde{\mathsf{A}^*} = \tilde{\mathsf{A}}^* \ . \tag{3.98}$$

3. Matriz hermítica: La matriz es etiquetada como hermítica (o autoadjunta) si

$$\mathsf{A} = \mathsf{A}^\dagger \ . \tag{3.99}$$

   Si A es real, entonces $\mathsf{A}^\dagger = \tilde{\mathsf{A}}$, y las matrices hermíticas reales son matrices reales y simétricas. En Mecánica Cuántica las matrices son hermíticas o unitarias.

4. Matriz unitaria: La matriz U es etiquetada como unitaria si

$$\mathsf{U}^\dagger = \mathsf{U}^{-1} \ . \tag{3.100}$$

   Si U es real, entonces $\mathsf{U}^{-1} = \tilde{\mathsf{U}}$, tal que las matrices reales unitarias son matrices ortogonales. Éste representa una generalización del concepto de matriz ortogonal.

5. $(\mathsf{AB})^* = \mathsf{B}^*\mathsf{A}^*$, $(\mathsf{AB})^\dagger = \mathsf{B}^\dagger\mathsf{A}^\dagger$.

Si los elementos son complejos, a la Física casi siempre le interesan las matrices adjuntas, hermíticas y unitarias. Las matrices unitarias son especialmente importantes en Mecánica Cuántica porque ellas dejan el largo de un vector (complejo) inalterado, análoga a la operación de una matriz ortogonal sobre un vector real. Una importante excepción a este interés en las matrices unitarias es el grupo de matrices de Lorentz.

En un espacio n-dimensional complejo el cuadrado del largo de un punto $\tilde{x} = (x_1, x_2, \ldots, x_n)$, o el cuadrado de su distancia al origen, es definido como $x^\dagger x = \sum_i x_i^* x_i = \sum_i |\, x_i \,|^2$. Si una trasformación de coordenadas $y = \mathsf{U}x$ deja la distancia inalterada, entonces $x^\dagger x = y^\dagger y = (\mathsf{U}x)^\dagger \mathsf{U}x = x^\dagger \mathsf{U}^\dagger \mathsf{U}x$. Ya que x es arbitrario concluimos que $\mathsf{U}^\dagger\mathsf{U} = \mathsf{1}_n$, *i.e.*, U es una matriz unitaria de $n \times n$. Si $x' = \mathsf{A}x$ es un mapeo lineal, entonces su matriz en las nuevas coordenadas llega a ser una transformación unitaria (análogo de una de similaridad)

$$\mathsf{A}' = \mathsf{U}\mathsf{A}\mathsf{U}^\dagger \ ,$$

porque $\mathsf{U}x' = y' = \mathsf{U}\mathsf{A}x = \mathsf{U}\mathsf{A}\mathsf{U}^{-1}y = \mathsf{U}\mathsf{A}\mathsf{U}^\dagger y$.

3.4.2. Matrices de Pauli y de Dirac.

El conjunto de tres matrices de Pauli de 2×2 σ_i

$$\sigma_1 = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \ , \quad \sigma_2 = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \ , \quad \sigma_3 = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \ , \tag{3.101}$$

fueron introducidas por W. Pauli para describir una partícula de spin $\frac{1}{2}$ en Mecánica Cuántica no relativista. Se puede demostrar que las σ satisfacen

$$\sigma_i\sigma_j + \sigma_j\sigma_i = 2\delta_{ij}1_2 \ , \quad \text{anticonmutación} \tag{3.102}$$

$$\sigma_i\sigma_j = i\sigma_k \ , \quad \text{permutación cíclica de los índices} \tag{3.103}$$

$$(\sigma_i)^2 = 1_2 \ , \tag{3.104}$$

donde 1_2 es la matriz unidad de 2×2. El vector $\vec{\sigma}/2$, con componentes σ_i, satisface las mismas reglas de conmutación

$$[\sigma_i, \sigma_j] \equiv \sigma_i\sigma_j - \sigma_j\sigma_i = 2i\varepsilon_{ijk}\sigma_k \ , \tag{3.105}$$

que el momento angular $\vec{L}$.

Las tres matrices de Pauli $\vec{\sigma}$ y la matriz unitaria forman un conjunto completo tal que cualquier matriz de 2×2 M puede ser expandida como

$$\mathsf{M} = m_0 1 + m_1\sigma_1 + m_2\sigma_2 + m_3\sigma_3 = m_0 1 + \vec{m}\cdot\vec{\sigma} \ , \tag{3.106}$$

donde los m_i son constantes. Usando $\sigma_i^2 = 1$ y $\mathrm{tr}(\sigma_i) = 0$ nosotros obtenemos a partir de la ecuación (3.106) los coeficientes m_i de la expansión formando las trazas,

$$2m_0 = \mathrm{tr}(\mathsf{M}) \ , \quad 2m_i = \mathrm{tr}(\mathsf{M}\,\sigma_i) \ , \quad i = 1,\, 2,\, 3 \ . \tag{3.107}$$

En 1927 P.A.M. Dirac extendió este formalismo para partículas de spin $\frac{1}{2}$ moviéndose a velocidades cercana a la de la luz tales como electrones. Para incluir la relatividad especial su punto de partida es la ecuación de Einstein para la energía $E^2 = \vec{p}^{\,2}c^2 + m^2c^4$ en vez de la energía cinética y potencial no relativista $E = \vec{p}^{\,2}/2m + V$. La clave para la ecuación de Dirac es factorizar

$$E^2 - \vec{p}^{\,2}c^2 = E^2 - (c\vec{\sigma}\cdot\vec{p})^2 = (E - c\vec{\sigma}\cdot\vec{p})(E + c\vec{\sigma}\cdot\vec{p}) = m^2c^4 \ , \tag{3.108}$$

usando la identidad matricial en 2×2

$$(c\vec{\sigma}\cdot\vec{p})^2 = \vec{p}^{\,2}1_2 \ . \tag{3.109}$$

La matriz unidad de 2×2 1_2 no es escrita explícitamente en la ecuación (3.108) y (3.109). Podemos presentar las matrices γ_0 y γ para factorizar $E^2 - \vec{p}^{\,2}c^2$ directamente,

$$(\gamma_0 E - \gamma c\vec{\sigma}\cdot\vec{p})^2 = \gamma_0^2E^2 + \gamma^2c^2(\vec{\sigma}\cdot\vec{p})^2 - Ec\vec{\sigma}\cdot\vec{p}(\gamma_0\gamma + \gamma\gamma_0) = E^2 - \vec{p}^{\,2}c^2 = m^2c^4 \ . \tag{3.110}$$

Si reconocemos

$$\gamma_0 E - \gamma c\vec{\sigma}\cdot\vec{p} = \gamma\cdot p = (\gamma_0, \gamma\vec{\sigma})\cdot(E, c\vec{p}) \ , \tag{3.111}$$

como el producto escalar de dos cuadrivectores γ^μ y p^μ, entonces la ecuación (3.110) con $p^2 = p \cdot p = E^2 - \vec{p}^2 c^2$ puede ser vista como una generalización cuadrivectorial de la ecuación (3.109). Claramente, para que la ecuación (3.110) mantenga las condiciones

$$\gamma_0^2 = 1 = -\gamma^2 \ , \quad \gamma_0\gamma + \gamma\gamma_0 = 0 \ , \tag{3.112}$$

debe satisfacerse que las cuatro matrices γ^μ anticonmuten, justo como las tres matrices de Pauli. Ya que estas últimas son un conjunto completo de matrices anticonmutantes de 2×2, la condición (3.112) no puede satisfacerse para matrices de 2×2, pero ella puede ser satisfecha para matrices de 4×4

$$\begin{aligned} \gamma_0 = \gamma^0 &= \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix} = \begin{pmatrix} 1_2 & 0 \\ 0 & -1_2 \end{pmatrix} \ , \\ \gamma &= \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 \\ -1 & 0 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 1_2 \\ -1_2 & 0 \end{pmatrix} \ . \end{aligned} \tag{3.113}$$

Alternativamente, el vector de matrices de 4×4

$$\gamma = \begin{pmatrix} 0 & \vec{\sigma} \\ -\vec{\sigma} & 0 \end{pmatrix} = \gamma\vec{\sigma} = \sigma_1 \times \vec{\sigma} \ , \tag{3.114}$$

puede obtenerse como el producto directo en el mismo sentido de la sección 3.2 de las matrices de 2×2 de Pauli. De la misma manera, $\gamma_0 = \sigma_3 \times 1_2$ y $1_4 = 1_2 \times 1_2$.

Resumiendo el tratamiento no relativista de una partícula de spin $\frac{1}{2}$, produce matrices de 4×4, mientras que las partículas no relativistas de spin $\frac{1}{2}$ son descritas por las matrices de Pauli σ de 2×2.

3.5. Diagonalización de matrices.

3.5.1. Momento de la matriz de inercia.

En muchos problemas en Física que involucran matrices reales simétricas o complejas hermíticas es deseable llevar a cabo una real transformación de similaridad ortogonal o una transformación unitaria (correspondiente a una rotación del sistema de coordenadas) para reducir la matriz a una forma diagonal, con todos los elementos no diagonales nulos. Un ejemplo particularmente directo de ésto es la matriz del momento de inercia I de un cuerpo rígido. A partir de la definición del momento angular $\vec{L}$ tenemos

$$\vec{L} = \mathsf{I}\vec{\omega} \ , \tag{3.115}$$

donde $\vec{\omega}$ viene a ser la velocidad angular. La matriz de inercia I tiene elementos diagonales

$$I_{xx} = \sum_i m_i(r_i^2 - x_i^2) \ \text{, y así sucesivamante,} \tag{3.116}$$

el subíndice i referencia la masa m_i localizada en $\vec{r}_i = (x_i, y_i, z_i)$. Para las componentes no diagonales tenemos

$$I_{xy} = -\sum_i m_i x_i y_i = I_{yx} \ . \tag{3.117}$$

Por inspección la matriz I es simétrica. También, ya que I aparece en una ecuación física de la forma (3.115), la cual se mantiene para todas las orientaciones del sistema de coordenadas, ésta puede ser considerada un tensor (regla del cociente).

La clave ahora es la orientación de los ejes (a lo largo de un cuerpo fijo) tal que I_{xy} y los otros elementos no diagonales desaparezcan. Como una consecuencia de esta orientación y una indicación de ella, si la velocidad angular está a lo largo de tales realineados ejes, la velocidad angular y el momento angular serán paralelos.

3.5.2. Autovectores y autovalores (eigenvector y eigenvalues).

Es quizás instructivo considerar un cuadro geométrico asociado a este problema. Si la matriz de inercia I es multiplicada a cada lado por un vector unitario cuya dirección es variable, $\hat{n} = (\alpha, \beta, \gamma)$, entonces en notación de Dirac

$$\langle \hat{n}|\mathsf{I}|\hat{n}\rangle = I \ , \tag{3.118}$$

donde I es el momento de inercia respecto a la dirección $\hat{n}$ y es un número positivo (escalar). Llevando a cabo la multiplicación, obtenemos

$$I = I_{xx}\alpha^2 + I_{yy}\beta^2 + I_{zz}\gamma^2 + 2I_{xy}\alpha\beta + 2I_{xz}\alpha\gamma + 2I_{yz}\beta\gamma \ . \tag{3.119}$$

Si introducimos

$$\vec{n} = \frac{\hat{n}}{\sqrt{I}} = (n_1, n_2, n_3) \ , \tag{3.120}$$

la cual es variable en dirección y magnitud, entonces la ecuación (3.119) llega a ser

$$1 = I_{xx}n_1^2 + I_{yy}n_2^2 + I_{zz}n_3^2 + 2I_{xy}n_1n_2 + 2I_{xz}n_1n_3 + 2I_{yz}n_2n_3 \ , \tag{3.121}$$

una forma cuadrática positiva, la cual debe ser un elipsoide (ver figura 3.5).

A partir de la geometría analítica es sabido que los ejes de coordenadas pueden ser rotados para coincidir con los ejes de nuestro elipsoide. En muchos casos elementales, especialmente cuando hay simetría, estos nuevos ejes, llamados ejes principales, pueden ser encontrados por inspección. Ahora nosotros procederemos a desarrollar un método general de hallazgo de los elementos diagonales y los ejes principales.

Si $\mathsf{R}^{-1} = \tilde{\mathsf{R}}$ es la correspondiente matriz ortogonal real tal que $\vec{n}' = \mathsf{R}\vec{n}$, o $|n'\rangle = \mathsf{R}|n\rangle$, en la notación de Dirac son las nuevas coordenadas; luego, obtenemos usando $\langle n'|\mathsf{R} = \langle n|$ en la ecuación (3.121)

$$\langle n|\mathsf{I}|n\rangle = \langle n'|\mathsf{R}\mathsf{I}\tilde{\mathsf{R}}|n'\rangle = I_1'{n_1'}^2 + I_2'{n_2'}^2 + I_3'{n_3'}^2 \ , \tag{3.122}$$

donde los $I_i' > 0$ son los momentos de inercia principales. La matriz de inercia I' en la ecuación (3.122) es diagonal en las nuevas coordenadas,

$$\mathsf{I}' = \mathsf{R}\mathsf{1}\tilde{\mathsf{R}} = \begin{pmatrix} I_1' & 0 & 0 \\ 0 & I_2' & 0 \\ 0 & 0 & I_3' \end{pmatrix} \ . \tag{3.123}$$

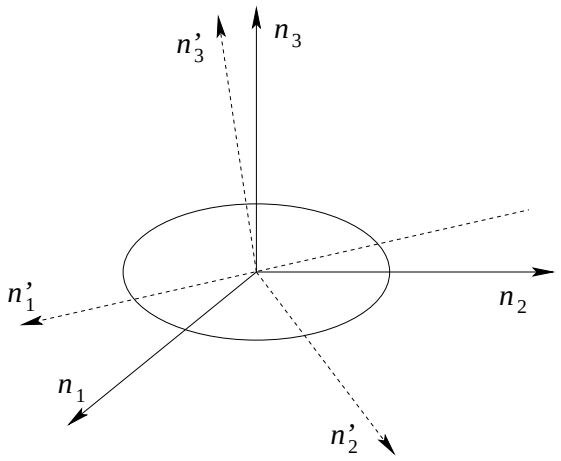


Figura 3.5: Elipsoide del momento de inercia.

Si reescribimos la ecuación (3.123) usando $\mathsf{R}^{-1} = \tilde{\mathsf{R}}$

$$\tilde{\mathsf{R}}\mathsf{I}' = \mathsf{I}\tilde{\mathsf{R}} \ , \tag{3.124}$$

y tomando $\tilde{\mathsf{R}} = (\vec{v}_1, \vec{v}_2, \vec{v}_3)$ compuesto de tres vectores columnas, entonces la ecuación (3.124) se separa en tres ecuaciones de autovalores

$$\mathsf{I}\vec{v}_i = I_i'\vec{v}_i \ , \quad i = 1, 2, 3 \ , \tag{3.125}$$

con autovalores I_i' y autovectores $\vec{v}_i$. Como estas ecuaciones son lineales y homogéneas (para un i fijo), por la sección 3.1 los determinantes tienen que anularse:

$$\begin{vmatrix} I_{11} - I_1' & I_{12} & I_{13} \\ I_{21} & I_{22} - I_2' & I_{23} \\ I_{31} & I_{32} & I_{33} - I_3' \end{vmatrix} = 0 \ . \tag{3.126}$$

Reemplazando los autovalores I_i' por una variable λ veces la matriz unidad $\mathsf{1}$, podríamos reescribir la ecuación (3.125) como

$$(\mathsf{I} - \lambda\mathsf{1})|v\rangle = 0 \ , \tag{3.127}$$

cuyo determinante

$$|\mathsf{I} - \lambda\mathsf{1}| = 0 \ , \tag{3.128}$$

es un polinomio cúbico en λ; sus tres raíces, por supuesto, son los I_i'. Sustituyendo una raíz de regreso en la ecuación (3.125), podemos encontrar los correspondientes autovectores. La ecuación (3.126) (o la (3.128)) es conocida como la *ecuación secular*. El mismo tratamiento se aplica a una matriz simétrica real I, excepto que sus autovalores no necesitan ser todos positivos. También, la condición de ortogonalidad en la ecuación (3.83a-3.83d) para R dice que, en términos geométricos, los autovectores $\vec{v}_i$ son vectores mutuamente ortogonales unitarios. Por cierto ellos forman las nuevas coordenadas del sistema. El hecho que cualquier par de

autovectores $\vec{v}_i$, $\vec{v}_j$ son ortogonales si $I'_i \neq I'_j$ se deduce de la ecuación (3.125) en conjunción con la simetría de I multiplicando con $\vec{v}_i$ y $\vec{v}_j$, respectivamente,

$$\langle v_j|\mathsf{I}|v_i\rangle = I'_i\langle v_j|v_i\rangle = \langle v_i|\mathsf{I}|v_j\rangle = I'_j\langle v_j|v_i\rangle \ . \tag{3.129}$$

Ya que $I'_i \neq I'_j$ y la ecuación (3.129) implica que $(I'_i - I'_j)\,\vec{v}_i \cdot \vec{v}_j = 0$, por lo tanto $\vec{v}_i \cdot \vec{v}_j = 0$.

3.5.3. Matrices hermíticas.

Para espacios vectoriales complejos las matrices unitarias y hermíticas juegan el mismo rol como las matrices ortogonales y simétricas sobre los espacios vectoriales reales, respectivamente. Primero, generalicemos el importante teorema acerca de los elementos diagonales y los ejes principales para la ecuación de autovalores

$$\mathsf{A}|r\rangle = \lambda|r\rangle \ . \tag{3.130}$$

Ahora mostramos que si A es una matriz hermítica, sus autovalores son reales y sus autovectores ortogonales.

Sean λ_i y λ_j dos autovalores y $|r_i\rangle$ y $|r_j\rangle$, los correspondientes autovectores de A, una matriz hermítica. Entonces

$$\mathsf{A}|r_i\rangle = \lambda_i|r_i\rangle \tag{3.131}$$
$$\mathsf{A}|r_j\rangle = \lambda_j|r_j\rangle \ . \tag{3.132}$$

La ecuación (3.131) es multiplicada por $|r_j\rangle$

$$\langle r_j|\mathsf{A}|r_i\rangle = \lambda_i \langle r_j|r_i\rangle \ . \tag{3.133}$$

La ecuación (3.132) es multiplicada por $|r_i\rangle$ para dar

$$\langle r_i|\mathsf{A}|r_j\rangle = \lambda_j \langle r_i|r_j\rangle \ . \tag{3.134}$$

Tomando la adjunta conjugada de esta ecuación, tenemos

$$\langle r_j|\mathsf{A}^\dagger|r_i\rangle = \lambda_j^* \langle r_j|r_i\rangle \tag{3.135}$$

o

$$\langle r_j|\mathsf{A}|r_i\rangle = \lambda_j^* \langle r_j|r_i\rangle \ , \tag{3.136}$$

ya que A es hermítica. Sustrayendo la ecuación (3.136) de la ecuación (3.133), obtenemos

$$(\lambda_i - \lambda_j^*)\langle r_j|r_i\rangle = 0 \ . \tag{3.137}$$

Este es un resultado general para todas las combinaciones posibles de i y j. Primero, sea $j = i$. Luego la ecuación (3.137) se convierte en

$$(\lambda_i - \lambda_i^*)\,\langle r_i|r_i\rangle = 0 \ . \tag{3.138}$$

Ya que $\langle r_i|r_i\rangle = 0$ sería una solución trivial de la ecuación (3.138), concluimos que

$$\lambda_i = \lambda_i^* \ , \tag{3.139}$$

es decir, λ_i es real, para todo i.

Segundo, para $i \neq j$ y $\lambda_i \neq \lambda_j$,

$$(\lambda_i - \lambda_j)\langle r_i|r_j\rangle = 0 \tag{3.140}$$

o

$$\langle r_i|r_j\rangle = 0 \tag{3.141}$$

lo cual significa que los autovectores de distintos autovalores son ortogonales, la ecuación (3.141) es la generalización de ortogonalidad en este espacio complejo.

Si $\lambda_i = \lambda_j$ (caso degenerado), $\langle r_i|$ no es automáticamente ortogonal a $|r_j\rangle$, pero podría hacerse ortogonal. Consideremos el problema físico de la matriz del momento de inercia nuevamente. Si x_i es un eje de simetría rotacional, entonces encontraremos que $\lambda_2 = \lambda_3$. Los autovectores $|r_2\rangle$ y $|r_3\rangle$ son cada uno perpendiculares al eje de simetría, $|r_1\rangle$, pero ellos yacen en alguna parte en el plano perpendicular a $|r_1\rangle$; esto es, alguna combinación lineal de $|r_2\rangle$ y $|r_3\rangle$ es también un autovector. Considere $(a_2|r_2\rangle + a_3|r_3\rangle)$ con a_2 y a_3 constantes. Entonces

$$\begin{aligned} \mathsf{A}(a_2|r_2\rangle + a_3|r_3\rangle) &= a_2\lambda_2|r_2\rangle + a_3\lambda_3|r_3\rangle \\ &= \lambda_2(a_2|r_2\rangle + a_3|r_3\rangle) \ , \end{aligned} \tag{3.142}$$

como es esperado, para x_1 un eje de simetría rotacional. Por lo tanto, si $|r_1\rangle$ y $|r_2\rangle$ son fijos, $|r_3\rangle$, puede simplemente escogerse yaciendo en el plano perpendicular a $|r_1\rangle$ y también perpendicular a $|r_2\rangle$. Un método general para ortogonalizar soluciones conocido como proceso de Gram-Schmidt, es aplicado a funciones más adelante.

El conjunto de n autovectores ortogonales de nuestra matriz hermítica de $n \times n$ forma un conjunto completo, generando el espacio de n dimensiones complejo. Este hecho es útil en un cálculo variacional de los autovalores. Los autovalores y los autovectores no están limitados a las matrices hermíticas. Todas las matrices tienen autovalores y autovectores. Por ejemplo, la matriz T de población estocástica satisface una ecuación de autovalores

$$\mathsf{T}\vec{P}_{\text{equilibrio}} = \lambda\vec{P}_{\text{equilibrio}} \ ,$$

con $\lambda = 1$. Sin embargo, solamente las matrices hermíticas tienen todos los autovectores ortogonales y todos sus autovalores reales.

3.5.4. Matrices antihermíticas.

Ocasionalmente, en Mecánica Cuántica encontramos matrices antihermíticas:

$$\mathsf{A}^\dagger = -\mathsf{A} \ .$$

Siguiendo el análisis de la primera porción de esta sección, podemos mostrar que

a. Los autovalores son imaginarios puros (o cero).

b. Los autovectores correspondientes a autovalores distintos son ortogonales.

La matriz R formada de los autovectores normalizados es unitaria. Esta propiedad antihermítica es preservada bajo transformaciones unitarias.

Ejemplo: Autovalores y autovectores de una matriz real simétrica.

Sea

$$\mathsf{A} = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \ . \tag{3.143}$$

La ecuación secular es

$$\begin{vmatrix} -\lambda & 1 & 0 \\ 1 & -\lambda & 0 \\ 0 & 0 & -\lambda \end{vmatrix} = 0 \ , \tag{3.144}$$

o

$$-\lambda(\lambda^2 - 1) = 0 \ , \tag{3.145}$$

expandiéndolo por las menores. Las raíces son $\lambda = -1, 0, 1$. Para encontrar el autovector correspondiente a $\lambda = -1$, sustituimos este valor de vuelta en la ecuación de autovalores, ecuación (3.130),

$$\begin{pmatrix} -\lambda & 1 & 0 \\ 1 & -\lambda & 0 \\ 0 & 0 & -\lambda \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \ . \tag{3.146}$$

Con $\lambda = -1$, esto produce

$$x + y = 0 \ , \qquad z = 0 \ . \tag{3.147}$$

Dentro de un factor de escala arbitrario, y un signo arbitrario (factor de fase), $\langle r_1| = (1, -1, 0)$. Notemos que (para el real $|r\rangle$ en el espacio ordinario) el autovector define una línea en el espacio. El sentido positivo o negativo no está determinado. Esta indeterminación puede ser entendida si notamos que la ecuación (3.130) es homogénea en $|r\rangle$. Por conveniencia requeriremos que los autovectores estén normalizados a la unidad, $\langle r_1|r_1\rangle = 1$. Con esta elección de signo

$$\langle r_1| = \vec{r_1} = \left(\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}, 0 \right) \ , \tag{3.148}$$

está fijo. Para $\lambda = 0$, la ecuación (3.130) produce

$$y = 0 \ , \qquad x = 0 \ , \tag{3.149}$$

$\langle r_2|$ o $\vec{r}_2 = (0, 0, 1)$ es un autovector aceptable. Finalmente, para $\lambda = 1$, tenemos

$$\begin{aligned} -x + y &= 0 \ , \\ z &= 0 \ , \end{aligned} \tag{3.150}$$

o

$$\langle r_3| = \vec{r_3} = \left(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}, 0 \right) \ . \tag{3.151}$$

La ortogonalidad de $\vec{r}_1$, $\vec{r}_2$ y $\vec{r}_3$, correspondientes a los tres autovalores distintos, puede ser fácilmente verificada.

Ejemplo: Autovalores degenerados.

Consideremos

$$\mathsf{A} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} \ . \tag{3.152}$$

La ecuación secular es

$$\begin{vmatrix} 1-\lambda & 0 & 0 \\ 0 & -\lambda & 1 \\ 0 & 1 & -\lambda \end{vmatrix} = 0 \ , \tag{3.153}$$

o

$$(1-\lambda)(\lambda^2 - 1) = 0 \ , \qquad \lambda = -1, 1, 1 \ , \tag{3.154}$$

un caso degenerado. Si $\lambda = -1$, la ecuación de autovalores (3.130) produce

$$2x = 0 \ , \qquad y + z = 0 \ . \tag{3.155}$$

Un autovector normalizado adecuado es

$$\langle r_1| = \vec{r_1} = \left(0, \frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}\right) \ . \tag{3.156}$$

para $\lambda = 1$, tenemos

$$-y + z = 0 \ . \tag{3.157}$$

Cualquier autovector que satisface la ecuación (3.157) es perpendicular a $\vec{r}_1$. Tenemos infinito número de opciones. Tomemos una elección posible tomando

$$\langle r_2| = \vec{r_2} = \left(0, \frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right) \ , \tag{3.158}$$

la cual claramente satisface la ecuación (3.157). Entonces $\vec{r}_3$ debe ser perpendicular a $\vec{r}_1$ y puede ser escogido perpendicular a $\vec{r}_2$ por[7]

$$\vec{r_3} = \vec{r_1} \times \vec{r_2} = (1, 0, 0) \ . \tag{3.159}$$

3.5.5. Funciones de matrices.

Polinomios con uno o más argumentos matriciales están bien definidos y ocurren a menudo. Series de potencias de una matriz también pueden estar definidas para dar la convergencia de la serie para cada uno de los elementos de matriz. Por ejemplo, si A es cualquier matriz de $n \times n$, entonces la serie de potencia

$$\exp(\mathsf{A}) = \sum_{i=0}^{\infty} \frac{\mathsf{A}^i}{i!} \ , \tag{3.160a}$$

$$\text{sen}(\mathsf{A}) = \sum_{i=0}^{\infty} (-1)^i \frac{\mathsf{A}^{2i+1}}{(2i+1)!} \ , \tag{3.160b}$$

$$\cos(\mathsf{A}) = \sum_{i=0}^{\infty} (-1)^i \frac{\mathsf{A}^{2i}}{(2i)!} \ , \tag{3.160c}$$

[7]El uso del producto cruz es limitado a tres dimensiones.

son matrices de $n \times n$ bien definidas. Para todas las matrices de Pauli σ_k la identidad de Euler para θ real y k =1, 2 ó 3

$$\exp(i\sigma_k\theta) = \mathsf{1}_2 \cos(\theta) + i\sigma_k \operatorname{sen}(\theta) \ , \tag{3.161}$$

sale a partir de colectar las potencias pares e impares en series separadas usando $\sigma_k^2 = \mathsf{1}$. Para las matrices de Dirac σ^{ij} de 4×4 con $(\sigma^{ij})^2 = 1$, si $j \neq k = 1, 2$ ó 3, obtenemos de manera similar (sin escribir las obvias matrices $\mathsf{1}_4$ nunca más)

$$\exp(i\sigma^{jk}\theta) = \cos(\theta) + i\sigma^{jk} \operatorname{sen}(\theta) \ , \tag{3.162}$$

mientras

$$\exp(i\sigma^{0k}\zeta) = \cosh(\zeta) + i\sigma^{0k} \operatorname{senh}(\zeta) \ , \tag{3.163}$$

manteniendo ζ real porque $(i\sigma^{0k})^2 = 1$ para $k = 1$, 2 ó 3.

Para una matriz hermítica A hay una matriz unitaria U que la diagonaliza, es decir, $\mathsf{UAU}^\dagger = [a_1, a_2, \ldots, a_n]$. Entonces la *fórmula de la traza*

$$\det(\exp(\mathsf{A})) = \exp(\operatorname{tr}(\mathsf{A})) \ . \tag{3.164}$$

Puede ser fácilmente demostrado.

Otra importante relación es la de *fórmula de Baker-Hausdorff*

$$\exp(i\mathsf{G})\mathsf{H}\exp(-i\mathsf{G}) = \mathsf{H} + [i\mathsf{G}, \mathsf{H}] + \frac{[i\mathsf{G}, [i\mathsf{G}, \mathsf{H}]]}{2!} + \cdots \tag{3.165}$$

lo cual resulta de multiplicar la serie de potencias para $\exp(i\mathsf{G})$ y recolectar los términos de la misma potencia en $i\mathsf{G}$. Aquí definimos

$$[\mathsf{G}, \mathsf{H}] = \mathsf{GH} - \mathsf{HG}$$

como el conmutador de G con H.

3.6. Matrices normales.

En la sección 3.5 nos concentramos principalmente en matrices hermíticas o reales simétricas y en el proceso de encontrar autovalores y autovectores. En esta sección generalizaremos a matrices normales con matrices hermíticas y unitarias como casos especiales. Consideramos los casos físicamente importantes como el problema de los modos de vibraciones y el problema numérico importante de matrices patológicas.

Una matriz normal es una matriz que conmuta con su adjunta,

$$[\mathsf{A}, \mathsf{A}^\dagger] = 0 \ .$$

Ejemplos obvios e importantes son las matrices hermíticas y unitarias. Mostraremos que las matrices normales tienen autovectores ortogonales (ver tabla 3.1). Procederemos en dos pasos:

Matriz	Autovalores	Autovectores (para diferentes autovalores)
Hermítica	Real	Ortogonal
Antihermítica	Imaginaria puro (o cero)	Ortogonal
Unitaria	Magnitud uno	Ortogonal
Normal	Si A tiene autovalor λ	Ortogonal
	$\mathsf{A}^\dagger$ tiene autovalor λ^*	A y $\mathsf{A}^\dagger$ tienen los mismos autovectores

Cuadro 3.1:

I. Sea $|x\rangle$ un autovector de A con correspondiente autovalor λ. Entonces

$$\mathsf{A}|x\rangle = \lambda|x\rangle \tag{3.166}$$

o

$$(\mathsf{A} - \lambda\mathsf{1})|x\rangle = 0 \ . \tag{3.167}$$

Por conveniencia la combinación $\mathsf{A}-\lambda\mathsf{1}$ la etiquetamos B. Tomando la adjunta de la ecuación (3.167), obtenemos

$$\langle x|(\mathsf{A} - \lambda\mathsf{1})^\dagger = 0 = \langle x|\mathsf{B}^\dagger \ . \tag{3.168}$$

Porque

$$[(\mathsf{A} - \lambda\mathsf{1}), (\mathsf{A} - \lambda\mathsf{1})^\dagger] = [\mathsf{A}, \mathsf{A}^\dagger] = 0 \ ,$$

tenemos

$$[\mathsf{B}, \mathsf{B}^\dagger] = 0 \ . \tag{3.169}$$

La matriz B es también normal.

A partir de las ecuaciones (3.167) y (3.168) formamos

$$\langle x|\mathsf{B}^\dagger\mathsf{B}|x\rangle = 0 \ . \tag{3.170}$$

Usando (3.169)

$$\langle x|\mathsf{B}\mathsf{B}^\dagger|x\rangle = 0 \ . \tag{3.171}$$

Ahora la ecuación (3.171) puede ser rescrita como

$$(\mathsf{B}^\dagger|x\rangle)^\dagger(\mathsf{B}^\dagger|x\rangle) = 0 \ . \tag{3.172}$$

Así

$$\mathsf{B}^\dagger|x\rangle = (\mathsf{A}^\dagger - \lambda^*\mathsf{1})|x\rangle = 0 \ . \tag{3.173}$$

Vemos que para matrices normales, $\mathsf{A}^\dagger$ tiene los mismos autovectores que A pero los autovalores son los complejos conjugados.

II. Ahora, consideremos más de un autovector-autovalor, tenemos

$$\mathsf{A}|x_i\rangle = \lambda_i|x_i\rangle \ , \tag{3.174}$$

$$\mathsf{A}|x_j\rangle = \lambda_j|x_j\rangle \ . \tag{3.175}$$

Multiplicando la ecuación (3.175) por la izquierda por $\langle x_i|$ produce

$$\langle x_i|\mathsf{A}|x_j\rangle = \lambda_j\langle x_i|x_j\rangle \ . \tag{3.176}$$

Operando sobre el lado izquierdo de la ecuación (3.176), obtenemos

$$\langle x_i|\mathsf{A} = (\mathsf{A}^\dagger|x_i\rangle)^\dagger \ . \tag{3.177}$$

A partir de la ecuación (3.173) sabemos que $\mathsf{A}^\dagger$ tiene los mismos autovectores que A pero con los complejos conjugados de los autovalores

$$(\mathsf{A}^\dagger|x_i\rangle)^\dagger = (\lambda_i^*|x_i\rangle)^\dagger = \lambda_i\langle x_i| \ . \tag{3.178}$$

Sustituyendo en la ecuación (3.176) tenemos

$$\lambda_i\langle x_i|x_j\rangle = \lambda_j\langle x_i|x_j\rangle \tag{3.179}$$

o

$$(\lambda_i - \lambda_j)\langle x_i|x_j\rangle = 0 \ . \tag{3.180}$$

Ésta es la misma que la ecuación (3.140).

Para $\lambda_i \neq \lambda_j$

$$\langle x_i|x_j\rangle = 0 \ .$$

Los autovectores correspondientes a diferentes autovalores de una matriz normal son *ortogonales*. Esto significa que una matriz normal puede ser diagonalizada por una transformación unitaria. La matriz unitaria requerida puede ser construida a partir de los vectores ortonormales como se mostró en la sección anterior.

El converso también es válido. Si A puede ser diagonalizada por una transformación unitaria, entonces A es normal.

3.6.1. Modos normales de vibración.

Consideremos las vibraciones de un modelo clásico de la molécula de CO_2. Ésta es una ilustración de la aplicación de las técnicas matriciales a un problema que no parte como un problema de matrices. También provee un ejemplo de autovalores y autovectores de una matriz real asimétrica.

Ejemplo: Modos Normales.

Consideremos tres masas sobre el eje x unidas por resortes como muestra la figura 3.6. Las fuerzas de los resortes se suponen lineales (para pequeños desplazamientos, ley de Hooke) y las masas se restringen a mantenerse sobre el eje x.

Usando una coordenada diferente para cada masa, la segunda ley de Newton produce el conjunto de ecuaciones

$$\begin{aligned}
\ddot{x}_1 &= -\frac{k}{M}(x_1 - x_2) \\
\ddot{x}_2 &= -\frac{k}{M}(x_2 - x_1) - \frac{k}{m}(x_2 - x_3) \\
\ddot{x}_3 &= -\frac{k}{M}(x_3 - x_2) \ .
\end{aligned} \tag{3.181}$$

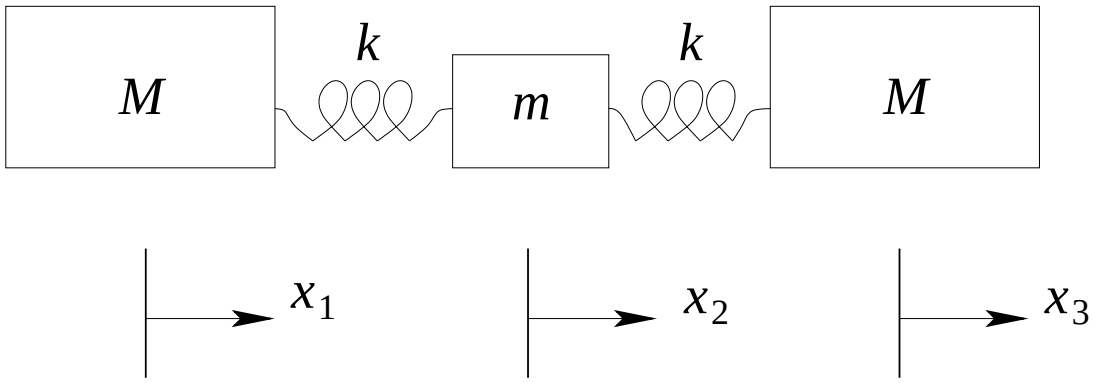


Figura 3.6: Masa con resortes.

El sistema de masas está vibrando. Buscamos las frecuencias comunes, ω tal que todas las masas vibren en esta misma frecuencia. Éstos son los modos normales. Sea

$$x_i = x_{i0}e^{i\omega t} \ , \qquad i = 1, 2, 3.$$

Substituyendo en la ecuación (3.181), podemos escribir este conjunto como

$$\begin{pmatrix} \frac{k}{M} & -\frac{k}{M} & 0 \\ -\frac{k}{m} & \frac{2k}{m} & -\frac{k}{m} \\ 0 & -\frac{k}{M} & \frac{k}{M} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \omega^2 \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \ , \tag{3.182}$$

dividiendo por el factor común $e^{i\omega t}$. Tenemos una ecuación matricial de autovalores con la matriz asimétrica. La ecuación secular es

$$\begin{vmatrix} \frac{k}{M} - \omega^2 & -\frac{k}{M} & 0 \\ -\frac{k}{m} & \frac{2k}{m} - \omega^2 & -\frac{k}{m} \\ 0 & -\frac{k}{M} & \frac{k}{M} - \omega^2 \end{vmatrix} = 0 \ . \tag{3.183}$$

Ésto conduce a

$$\omega^2 \left(\frac{k}{M} - \omega^2 \right) \left(\omega^2 - \frac{2k}{m} - \frac{k}{M} \right) = 0$$

Los autovalores son

$$\omega^2 = 0 \ , \quad \frac{k}{M} \ , \quad \text{y} \quad \frac{k}{M} + \frac{2k}{m} \ ,$$

todas reales.

Los correspondientes autovectores son determinados sustituyendo los autovalores de regreso en la ecuación (3.182) un autovalor a la vez. Para $\omega^2 = 0$, ecuación (3.182) produce

$$\begin{aligned} x_1 - x_2 &= 0 \\ -x_1 + 2x_2 - x_3 &= 0 \\ -x2 + x_3 &= 0 \ . \end{aligned}$$

Entonces, tenemos

$$x_1 = x_2 = x_3 \ .$$

Ésto describe una translación pura sin movimiento relativo de las masas y sin vibración.

Para $\omega^2 = \dfrac{k}{M}$, la ecuación (3.182) produce

$$x_1 = -x_3 \ , \qquad x_2 = 0 \ . \tag{3.184}$$

Las masas exteriores se mueven en direcciones opuestas. El masa del centro está estacionaria.

Para $\omega^2 = \dfrac{k}{M} + \dfrac{2k}{M}$, las componentes de los autovectores son

$$x_1 = x_3 \ , \qquad x_2 = -\frac{2M}{m} x_1 \ .$$

Las dos masas exteriores se están moviendo juntas. La masa del centro se está moviendo opuesta a las otras dos. El momento neto es cero.

Cualquier desplazamiento de estas tres masas a lo largo del eje x puede ser descrito como una combinación lineal de estos tres tipos de movimiento: translación más dos formas de vibración.

3.6.2. Sistemas con condiciones patológicas.

Un sistema lineal de ecuaciones puede ser escrito como

$$\mathsf{A}|x\rangle = |y\rangle \quad \text{o} \quad \mathsf{A}^{-1}|y\rangle = |x\rangle \ , \tag{3.185}$$

con A y $|y\rangle$ conocido y $|x\rangle$ desconocido. Podemos encontrar ejemplos en los cuales un pequeño error en $|y\rangle$ resulta en un gran error en $|x\rangle$. En este caso la matriz A es llamada de condición patológica. Si $|\delta x\rangle$ es el error en $|x\rangle$ y $|\delta y\rangle$ es el error en $|y\rangle$, entonces los errores relativos pueden ser escritos como

$$\left[\frac{\langle \delta x|\delta x\rangle}{\langle x|x\rangle}\right]^{1/2} \leq K(\mathsf{A}) \left[\frac{\langle \delta y|\delta y\rangle}{\langle y|y\rangle}\right]^{1/2} \ . \tag{3.186}$$

Aquí $K(\mathsf{A})$, una propiedad de la matriz A, es llamada la *condición de número*. Para A hermítica una forma de la condición de número es dada por

$$K(\mathsf{A}) = \frac{|\,\lambda\,|_{\max}}{|\,\lambda\,|_{\min}} \ . \tag{3.187}$$

Una forma aproximada debido a Turing es

$$K(\mathsf{A}) = n[A_{ij}]_{\max}[A_{ij}^{-1}]_{\max} \ , \tag{3.188}$$

en la cual n es el orden de la matriz y $[A_{ij}]_{\max}$ es el máximo elemento en A.

Ejemplo: Una matriz patológica.

Un ejemplo común de una matriz con condición patológica es la matriz de Hilbert, la matriz de Hilbert de orden 4 es $H_{ij} = (i + j - 1)^{-1}$,

$$\mathsf{H}_4 = \begin{pmatrix} 1 & \frac{1}{2} & \frac{1}{3} & \frac{1}{4} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} \\ \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} \end{pmatrix} \ . \tag{3.189}$$

Los elementos de la matriz inversa (orden n) son dados por

$$(\mathsf{H}_n^{-1})_{ij} = \frac{(-1)^{i+j}}{i+j\pm 1} \cdot \frac{(n+i-1)!(n+j-1)!}{[(i-1)!(j-1)!]^2(n-i)!(n-j)!} \ . \tag{3.190}$$

Para $n = 4$

$$\mathsf{H}_4^{-1} = \begin{pmatrix} 16 & -120 & 240 & -140 \\ -120 & 1200 & -2700 & 1680 \\ 240 & -2700 & 6480 & -4200 \\ -140 & 1680 & -4200 & 2800 \end{pmatrix} \ . \tag{3.191}$$

A partir de la ecuación (3.188) la estimación de Turing de la condición de número para H_4 llega a ser

$$\begin{aligned} K_{\text{Turing}} &= 4 \times 1 \times 6480 \\ &= 2.59 \times 10^4 \ . \end{aligned}$$

Ésto es una advertencia de que un error en la entrada puede ser multiplicado por 26000 en el cálculo del resultado de salida. Ésto indica que H_4 tiene condición patológica. Si usted encuentra un sistema altamente patológico tiene un par de alternativas (además de abandonar el problema).

a. Tratar un ataque matemático diferente.

b. Hacer arreglos para llevar más cifras significativas y, a costa de fuerza bruta, empujar de principio a fin.

Capítulo 4

Teoría de grupo.

versión final 2.31-021030[1]

Disciplined judgment about what is neat and simmetrical and elegant has time and time again proved an excellent guide to how nature work.

Murray Gell-Mann

4.1. Introducción.

En mecánica clásica la *simetría* de un sistema físico conduce a una *ley de conservación*. La conservación del momento angular es una consecuencia directa de la simetría rotacional, lo cual significa *invariancia* bajo rotaciones espaciales. A principios del siglo pasado, Wigner y otros comprendieron que la invariancia era el concepto clave en el entendimiento de los nuevos fenómenos y en el desarrollo de teorías apropiadas. Así, en mecánica cuántica los conceptos de momento angular y spin han llegado a ser aún más centrales. Sus generalizaciones, el *isospin*, en física nuclear y la simetría de *sabor* en física de partículas, son herramientas indispensables en la construcción teórica y en sus soluciones. Las generalizaciones del concepto de invariancia de *gauge* de la electrodinámica clásica para la simetría del isospin conduce a la teoría de gauge electro-débil.

En cada caso el conjunto de estas operaciones de simetría forman un *grupo*. La teoría de grupo es la herramienta matemática para tratar las invariancias y las simetrías. Ella trae consigo unificación y formalización de principios tales como reflexión espacial, paridad, momento angular y geometría, que son ampliamente usados por los físicos.

En geometría el rol fundamental de la teoría de grupo fue reconocido hace mucho tiempo por los matemáticos. En geometría euclidiana, la distancia entre dos puntos y el producto escalar de dos vectores, o métrica, no cambia, bajo rotaciones o translaciones. Estas simetrías son características de esta geometría. En relatividad especial la métrica, o producto escalar de cuadrivectores, difiere del de la geometría euclidiana en que ya no es más positivo definido y es invariante ante transformaciones de Lorentz.

[1]Este capítulo está basado en el cuarto capítulo del libro: *Mathematical Methods for Physicists, fourth edition* de George B. Arfken & Hans J. Weber, editorial Academic Press.

Para un cristal el grupo de simetría contiene sólo un número finito de rotaciones en valores discretos del ángulo y reflexiones. La teoría de tales grupos discretos o finitos, desarrollada inicialmente como una rama de las matemáticas pura, ahora es una útil herramienta para el desarrollo de la cristalografía y la física de la materia condensada. Haremos una breve introducción a ellos. Cuando las rotaciones dependen de un ángulo continuo el grupo de rotaciones tiene un número infinito de elementos. Estudiaremos tales grupos continuos o de Lie.

4.1.1. Definición de grupo.

Un grupo G puede ser definido como un conjunto de objetos u operaciones, llamados los elementos de G, que pueden ser combinados o "multiplicados" para formar un producto bien definido en G, el cual satisface las siguientes cuatro condiciones.

1. Si a y b son cualquier par de elementos de G, entonces el producto ab es también elemento de G; o $(a, b) \to ab$ mapea $G \times G$ sobre G.

2. Esta multiplicación es asociativa, $(ab)c = a(bc)$.

3. Hay un elemento unidad o neutro I en G tal que $Ia = aI = a$ para cada elemento a de G.[2]

4. Debe haber un inverso o recíproco de cada elemento a de G, etiquetado a^{-1}, tal que $aa^{-1} = a^{-1}a = I$.

Un ejemplo para un grupo es el conjunto de rotaciones de coordenadas en el sentido contrario al del puntero del reloj,

$$\mathsf{R}(\varphi) = \begin{pmatrix} \cos\varphi & \operatorname{sen}\varphi \\ -\operatorname{sen}\varphi & \cos\varphi \end{pmatrix} \tag{4.1}$$

en un ángulo φ del sistema de coordenadas xy a una nueva orientación. El producto de dos rotaciones $\mathsf{R}(\varphi_1)\mathsf{R}(\varphi_2)$ es definida como una rotación, primero en un ángulo φ_2 y entonces en un ángulo φ_1. De acuerdo a la ecuación (3.29), esto corresponde al producto de las dos matrices ortogonales de 2×2

$$\begin{pmatrix} \cos\varphi_1 & \operatorname{sen}\varphi_1 \\ -\operatorname{sen}\varphi_1 & \cos\varphi_1 \end{pmatrix} \begin{pmatrix} \cos\varphi_2 & \operatorname{sen}\varphi_2 \\ -\operatorname{sen}\varphi_2 & \cos\varphi_2 \end{pmatrix} = \begin{pmatrix} \cos(\varphi_1+\varphi_2) & \operatorname{sen}(\varphi_1+\varphi_2) \\ -\operatorname{sen}(\varphi_1+\varphi_2) & \cos(\varphi_1+\varphi_2) \end{pmatrix} , \tag{4.2}$$

usando las fórmulas de adición de funciones trigonométricas. El producto es claramente una rotación representada por una matriz ortogonal con un ángulo $\varphi_1 + \varphi_2$. El producto es la multiplicación asociativa de matrices. Es *conmutativo* o *abeliano* porque el orden en el cual estas rotaciones son realizadas no importa. El inverso de la rotación con ángulo φ es una rotación con ángulo $-\varphi$. La unidad o neutro corresponde al ángulo $\varphi = 0$. El nombre del grupo es SO(2), si el ángulo varía continuamente desde 0 a 2π. Claramente, SO(2) tiene infinitos elementos. La unidad con ángulo $\varphi = 0$ y la rotación con $\varphi = \pi$ forman un *subgrupo*

[2]También etiquetan al elemento unidad como E.

finito. Un subgrupo G' de un grupo G consiste de elementos de G tal que el producto de cualquiera de sus elementos está de nuevo en el subgrupo G', *i.e.*, G' es cerrado bajo la multiplicación de G. Si $gg'g^{-1}$ es un elemento de G' para cualquier g de G y g' de G', entonces G' es llamado un *subgrupo invariante* de G.

Las matrices ortogonales $n \times n$ forman el grupo O(n), y también SO(n) si sus determinantes son +1 (S por eSpecial). Si $\tilde{\mathsf{O}}_i = \mathsf{O}_i^{-1}$ para $i = 1$ y 2, entonces el producto

$$\widetilde{\mathsf{O}_1\mathsf{O}_2} = \tilde{\mathsf{O}}_2\tilde{\mathsf{O}}_1 = \mathsf{O}_1^{-1}\mathsf{O}_2^{-1} = (\mathsf{O}_1\mathsf{O}_2)^{-1}$$

es también una matriz ortogonal en SO(n). La inversa es la matriz (ortogonal) transpuesta. La unidad del grupo es $\mathsf{1}_n$. Una matriz real ortogonal de $n \times n$ tiene $n(n-1)/2$ parámetros independientes. Para $n = 2$ hay sólo un parámetro: un ángulo en la ecuación (4.1). Para $n = 3$, hay tres parámetros independientes: los tres ángulos de Euler de la sección 3.3.

De la misma manera, las matrices unitarias de $n \times n$ forman el grupo U(n), y también SU(n) si sus determinantes son +1. Si $\mathsf{U}_i^\dagger = \mathsf{U}_i^{-1}$, entonces

$$(\mathsf{U}_1\mathsf{U}_2)^\dagger = \mathsf{U}_2^\dagger\mathsf{U}_1^\dagger = \mathsf{U}_2^{-1}\mathsf{U}_1^{-1} = (\mathsf{U}_1\mathsf{U}_2)^{-1} \ ,$$

tal que el producto es unitario y un elemento de SU(n). Cada matriz unitaria tiene una inversa, la cual es también unitaria.

4.1.2. Homomorfismo, isomorfismo.

Puede haber una correspondencia entre los elementos de dos grupos (o entre dos representaciones), uno a uno, dos a uno, o muchos a uno. Si esta correspondencia preserva la multiplicación del grupo, diremos que los dos grupos son *homomórficos*. Una de las más importantes correspondencias homomórficas entre el grupo de rotaciones SO(3) y el grupo de matrices unitarias SU(2) será desarrollado en la próxima sección. Si la correspondencia es uno a uno, y aún preserva la multiplicación del grupo,[3] entonces los grupos son *isomórficos*. Un ejemplo es las rotaciones de coordenadas a través de un ángulo finito φ en el sentido horario respecto al eje z en el espacio tridimensional descrito por

$$\mathsf{R}_z(\varphi) = \begin{pmatrix} \cos\varphi & \operatorname{sen}\varphi & 0 \\ -\operatorname{sen}\varphi & \cos\varphi & 0 \\ 0 & 0 & 1 \end{pmatrix} \ . \tag{4.3}$$

El grupo de rotaciones R_z es isomórfico al grupo de rotaciones en la ecuación (4.1).

4.1.3. Representaciones matriciales, reducibles e irreducibles.

La representación de los elementos de un grupo por matrices es una técnica muy poderosa y ha sido casi universalmente adoptada por los físicos. El uso de matrices no impone restricciones significativas. Puede mostrarse que los elementos de cualquier grupo finito y de grupos

[3]Supongamos que los elementos del primer grupo son etiquetados por g_i y los elementos del segundo grupo por h_i. Entonces $g_i \leftrightarrow h_i$ en una correspondencia uno a uno para todos los valores de i. Si $g_ig_j = g_k$ y $h_ih_j = h_k$, entonces g_k y h_k deben ser los elementos correspondientes del grupo.

continuos pueden ser representados por matrices. Ejemplos son las rotaciones descritas en la ecuación (4.1) y (4.3).

Para ilustrar cómo las representaciones matriciales surgen a partir de una simetría, consideremos la ecuación estacionaria de Schrödinger (o alguna otra ecuación de autovalores tal como $\mathsf{I}\, v_i = I_i v_i$ para los momentos principales de inercia de un cuerpo rígido en mecánica clásica)

$$H\psi = E\psi \ . \tag{4.4}$$

Supongamos que la ecuación (4.4) se mantiene invariante bajo la acción de un grupo G de transformaciones R en G (rotaciones de coordenadas, por ejemplo, para un potencial central $V(r)$ en el Hamiltoniano H), *i.e.*,

$$H_R = \mathsf{R}H\mathsf{R}^{-1} = H \ . \tag{4.5}$$

Ahora tomamos una solución ψ de la ecuación (4.4) y la "rotamos": $\psi \to \mathsf{R}\psi$. Entonces $\mathsf{R}\psi$ tiene la misma energía E porque multiplicando la ecuación (4.4) por R y usando (4.5) produce

$$\mathsf{R}H\psi = E(\mathsf{R}\psi) = (\mathsf{R}H\mathsf{R}^{-1})\mathsf{R}\psi = H(\mathsf{R}\psi) \ . \tag{4.6}$$

En otras palabras, todas las soluciones rotadas $\mathsf{R}\psi$ son *degeneradas* en energía o forman lo que los físicos llaman un *multiplete*. Supongamos que este espacio vectorial V_ψ de soluciones transformadas tiene una dimensión finita n. Sean $\psi_1, \psi_2, \ldots, \psi_n$ una base. Ya que $\mathsf{R}\psi_j$ es un miembro del multiplete, podemos expandirlo en términos de esta base

$$\mathsf{R}\psi_j = \sum_k r_{jk}\psi_k \ . \tag{4.7}$$

Así, cada R en G puede ser asociado a una matriz (r_{jk}), y este mapeo $\mathsf{R} \to (r_{jk})$ es llamada una representación de G. Si podemos tomar cualquier elemento de V_ψ y por rotaciones con todos los elementos de R de G transforman en todos los otros elementos de V_ψ entonces la representación es *irreducible*. Si todos los elementos de V_ψ no son alcanzados, entonces V_ψ se separa en una suma directa de dos o más subespacios vectoriales, $V_\psi = V_1 \oplus V_2 \oplus \ldots$, los cuales son mapeados dentro de ellos mismos por rotación de sus elementos. En este caso la representación es llamada *reducible*. Entonces podemos encontrar una base en V_ψ (*i.e.*, hay una matriz unitaria U) tal que

$$\mathsf{U}(r_{jk})\mathsf{U}^\dagger = \begin{pmatrix} \mathsf{r}_1 & \mathsf{0} & \ldots \\ \mathsf{0} & \mathsf{r}_2 & \ldots \\ \vdots & \vdots & \ddots \end{pmatrix} \tag{4.8}$$

para todos los R de G y todas las matrices (r_{jk}). Aquí r_1, r_2, ..., son matrices de menor dimensión que (r_{jk}), las que están alineadas a lo largo de la diagonal y los $\mathsf{0}$ son matrices de ceros. Podemos decir que R ha sido descompuesta en $\mathsf{r}_1 + \mathsf{r}_2 + \ldots$ en paralelo con $V_\psi = V_1 \oplus V_2 \oplus \ldots$.

Las representaciones irreducibles juegan un rol en teoría de grupo que es aproximadamente análogo a los vectores unitarios en el análisis vectorial. Ellas son las representaciones más simples, toda otra puede ser construida desde ellas.

4.2. Generadores de grupos continuos.

Un característica de los grupos continuos, conocidos como grupos de Lie, es que los parámetros de un producto de elementos son funciones analíticas de los parámetros de los factores. La naturaleza analítica de las funciones nos permite desarrollar el concepto de *generador* y reduce el estudio del grupo completo a un estudio de los elementos del grupo en la vecindad del elemento identidad.

La idea esencial de Lie fue el estudio de elementos R en un grupo G que estén infinitesimalmente cercanos a la unidad de G. Consideremos el grupo SO(2) como un ejemplo simple. Las matrices de rotación de 2×2 en la ecuación (4.1) puede ser escrita en forma exponencial usando la identidad de Euler ecuación (3.161) como

$$\mathsf{R}(\varphi)=\begin{pmatrix}\cos\varphi & \operatorname{sen}\varphi\\ -\operatorname{sen}\varphi & \cos\varphi\end{pmatrix}=\mathsf{1}_2\cos\varphi+i\sigma_2\operatorname{sen}\varphi=\exp(i\sigma_2\varphi)\ . \tag{4.9}$$

A partir de la forma exponencial es obvio que la multiplicación de estas matrices es equivalente a la suma de los argumentos

$$\mathsf{R}(\varphi_2)\mathsf{R}(\varphi_1)=\exp(i\sigma_2\varphi_2)\exp(i\sigma_2\varphi_1)=\exp(i\sigma_2(\varphi_1+\varphi_2))=\mathsf{R}(\varphi_1+\varphi_2)\ .$$

Por supuesto las rotaciones cercanas a $\mathsf{1}$ tienen un ángulo pequeño $\varphi\sim 0$.

Esto sugiere que busquemos una representación exponencial

$$\mathsf{R}=\exp(i\varepsilon\mathsf{S})\ ,\qquad \varepsilon\to 0\ , \tag{4.10}$$

para elementos del grupo $\mathsf{R}\in G$ cercanos a la $\mathsf{1}$. Las transformaciones infinitesimales S son llamadas los *generadores* de G. Ellos forman un espacio lineal cuya dimensión es el *orden* de G porque la multiplicación de los elementos R del grupo se traduce en la suma de los generadores S.

Si R no cambia el elemento de volumen, *i.e.*, $\det(\mathsf{R})=1$, nosotros usamos la ecuación (3.164) para ver que

$$\det(\mathsf{R})=\exp(\operatorname{tr}(\ln\mathsf{R}))=\exp(i\varepsilon\operatorname{tr}(\mathsf{S}))=1$$

implica que los generadores son de traza nula,

$$\operatorname{tr}(\mathsf{S})=0\ . \tag{4.11}$$

Este es el caso para el grupo de rotaciones SO(n) y el grupo unitario SU(n), como veremos más adelante.

Si R de G en la ecuación (4.1) es unitario, entonces $\mathsf{S}^\dagger=\mathsf{S}$ es hermítica, lo cual también es el caso para SO(n) y SU(n). Ya que hay un i extra en la ecuación (4.10).

Expandamos los elementos del grupo

$$\begin{aligned}\mathsf{R}_i&=\exp(i\varepsilon_i\mathsf{S}_i)=\mathsf{1}+i\varepsilon_i\mathsf{S}_i-\frac{1}{2}\varepsilon_i^2\mathsf{S}_i^2+\dots\ ,\\ \mathsf{R}_i^{-1}&=\exp(-i\varepsilon_i\mathsf{S}_i)=\mathsf{1}-i\varepsilon_i\mathsf{S}_i-\frac{1}{2}\varepsilon_i^2\mathsf{S}_i^2+\dots\ ,\end{aligned} \tag{4.12}$$

a segundo orden en el pequeño parámetro del grupo ε_i porque los términos lineales y varios términos cuadráticos se cancelan en el producto (figura 4.1)

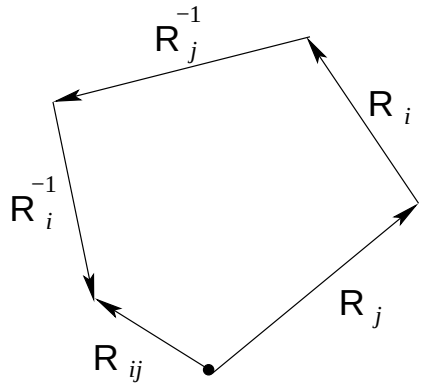


Figura 4.1: Ilustración de la ecuación (4.13).

$$\begin{aligned} \mathsf{R}_i^{-1}\mathsf{R}_j^{-1}\mathsf{R}_i\mathsf{R}_j &= 1 + \varepsilon_i\varepsilon_j[\mathsf{S}_j,\mathsf{S}_i] + \dots \ , \\ &= 1 + \varepsilon_i\varepsilon_j \sum_k c_{ji}^k \mathsf{S}_k + \dots \ , \end{aligned} \tag{4.13}$$

cuando las ecuaciones (4.12) son sustituidas dentro de la ecuación (4.13). La última línea es debido a que el producto en la ecuación (4.13) es nuevamente un elemento, R_{ij}, cercano a la unidad en el grupo G. Por tanto su exponente debe ser una combinación lineal de los generadores S_k y sus parámetros infinitesimales del grupo tienen que ser proporcionales al producto $\varepsilon_i\varepsilon_j$. Comparando ambas líneas (4.13) encontramos la relación de *clausura* de los generadores del grupo de Lie G,

$$[\mathsf{S}_i,\mathsf{S}_j] = \sum_k c_{ij}^k \mathsf{S}_k \tag{4.14}$$

Los coeficientes c_{ij}^k son las *constantes de estructura* del grupo G. Ya que el conmutador en la ecuación (4.14) es antisimétrico en i y en j, también lo son las constantes de estructura en los índices inferiores,

$$c_{ij}^k = -c_{ji}^k \ . \tag{4.15}$$

Si el conmutador en la ecuación (4.14) es tomado como *la regla de multiplicación de los generadores*, vemos que el espacio vectorial de los generadores llega a ser un álgebra, el álgebra de Lie G del grupo G. Para $\mathrm{SU}(l+1)$ el álgebra de Lie es llamada A_l, para $\mathrm{SO}(2l+1)$ es B_l y para $\mathrm{SO}(2l)$ es D_l, donde $l = 1, 2, \dots$ es un entero positivo, esto será llamado el *rango* de grupo de Lie G o de su álgebra G.

Finalmente, la *identidad de Jacobi* se satisface para los dobles conmutadores

$$[[\mathsf{S}_i,\mathsf{S}_j],\mathsf{S}_k] + [[\mathsf{S}_j,\mathsf{S}_k],\mathsf{S}_i] + [[\mathsf{S}_k,\mathsf{S}_i],\mathsf{S}_j] = 0 \ , \tag{4.16}$$

lo cual es fácilmente verificable usando la definición de conmutador. Cuando la ecuación (4.14) es substituida en (4.16) encontramos otra restricción sobre las constantes de estructura,

$$\sum_m \left\{ c_{ij}^m[\mathsf{S}_m,\mathsf{S}_k] + c_{jk}^m[\mathsf{S}_m,\mathsf{S}_i] + c_{ki}^m[\mathsf{S}_m,\mathsf{S}_j] \right\} = 0 \ . \tag{4.17}$$

Usando de nuevo la ecuación (4.14) en la ecuación (4.17) implica que

$$\sum_{mn} \left\{ c_{ij}^m c_{mk}^n \mathsf{S}_n + c_{jk}^m c_{mi}^n \mathsf{S}_n + c_{ki}^m c_{mj}^n \mathsf{S}_n \right\} = 0 \ , \tag{4.18}$$

donde el factor común S_n (y la suma sobre n) pueden eliminarse porque los generadores son linealmente independientes. Por tanto

$$\sum_{m} \left\{ c_{ij}^m c_{mk}^n + c_{jk}^m c_{mi}^n + c_{ki}^m c_{mj}^n \right\} = 0 \ . \tag{4.19}$$

Las relaciones (4.14), (4.15) y (4.19) forman la base de las álgebras de Lie desde la cual los elementos finitos del grupo de Lie cerca de su unidad puede ser reconstruido.

Volviendo a la ecuación (4.5), el inverso de R es exactamente $\mathsf{R}^{-1} = \exp(-i\varepsilon \mathsf{S})$. Expandimos H_R de acuerdo a la fórmula de Baker-Haudorff, ecuación (3.17),

$$\mathsf{H} = \mathsf{H}_R = \exp(i\varepsilon \mathsf{S}) \mathsf{H} \exp(-i\varepsilon \mathsf{S}) = \mathsf{H} + i\varepsilon [\mathsf{S}, \mathsf{H}] - \varepsilon^2 \frac{[\mathsf{S}, [i\mathsf{S}, \mathsf{H}]]}{2!} + \cdots \ . \tag{4.20}$$

Al simplificar H de la ecuación (4.20), dividiendo por ε y haciendo $\varepsilon \to 0$. Entonces la ecuación (4.20) implica que para cualquier rotación cercana a $\mathsf{1}$ en G el conmutador

$$[\mathsf{S}, \mathsf{H}] = 0 \ . \tag{4.21}$$

Si S y H son matrices hermíticas, la ecuación (4.21) dice que S y H pueden ser simultáneamente diagonalizados. Si S y H son operadores diferenciales como el Hamiltoniano y el momento angular orbital en mecánica cuántica, entonces la ecuación (4.21) dice que S y H tienen autofunciones en común y que los autovalores degenerados de H pueden ser distinguidos por los autovalores de los generadores S. Esta, es con mucho, la más importante aplicación de teoría de grupos a mecánica cuántica.

A continuación, estudiaremos los grupos ortogonales y unitarios como ejemplos.

4.2.1. Grupos de rotaciones SO(2) y SO(3).

Para SO(2) definido por la ecuación (4.1) hay sólo un generador linealmente independiente, σ_2 y el orden de SO(2) es 1. Obtenemos σ_2 a partir de diferenciar la ecuación (4.9) y evaluarla en cero,

$$-i \frac{d\mathsf{R}(\varphi)}{d\varphi}\bigg|_{\varphi=0} = -i \begin{pmatrix} -\operatorname{sen}\varphi & \cos\varphi \\ -\cos\varphi & -\operatorname{sen}\varphi \end{pmatrix} \bigg|_{\varphi=0} = -i \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} = \sigma_2 \ . \tag{4.22}$$

Para las rotaciones $\mathsf{R}_z(\varphi)$ sobre el eje z descritas por la ecuación (4.3), el generador es dado por

$$-i \frac{d\mathsf{R}(\varphi)}{d\varphi}\bigg|_{\varphi=0} = \mathsf{S}_z = \begin{pmatrix} 0 & -i & 0 \\ i & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \ , \tag{4.23}$$

donde el factor extra i es insertado para hacer S_z hermítica. La rotación $\mathsf{R}_z(\delta\varphi)$ en un ángulo infinitesimal $\delta\varphi$ puede ser escrita como

$$\mathsf{R}_z(\delta\varphi) = \mathsf{1}_3 + i\delta\varphi \mathsf{S}_z \ , \tag{4.24}$$

Una expansión de Maclaurin-Taylor de R_z cerca de la unidad, $\varphi = 0$, con términos hasta orden $(\delta\varphi)^2$ y los superiores son despreciados. Una rotación finita puede ser compuesta por sucesivas rotaciones infinitesimales

$$\mathsf{R}_z(\delta\varphi_1 + \delta\varphi_2) = (1_3 + i\delta\varphi_1\mathsf{S}_z)(1_3 + i\delta\varphi_2\mathsf{S}_z) \ . \tag{4.25}$$

Sea $\delta\varphi = \varphi/N$ para N rotaciones, con $N \to \infty$. Entonces,

$$\mathsf{R}_z(\varphi) = \lim_{N\to\infty}\left[1_3 + i\frac{\varphi}{N}\mathsf{S}_z\right]^N = \exp(i\mathsf{S}_z) \ . \tag{4.26}$$

Esta forma identifica S_z como el generador del grupo R_z, un subgrupo abeliano de SO(3), el grupo de rotaciones en tres dimensiones con determinante +1. Cada matriz de 3×3 $\mathsf{R}_z(\varphi)$ es ortogonal, por lo tanto unitaria, y la $\mathrm{tr}(\mathsf{S}_z) = 0$ de acuerdo con la ecuación (4.11).

Por diferenciación de las rotaciones de coordenadas

$$\mathsf{R}_x(\psi) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\psi & \operatorname{sen}\psi \\ 0 & -\operatorname{sen}\psi & \cos\psi \end{pmatrix} \ , \quad \mathsf{R}_y(\theta) = \begin{pmatrix} \cos\theta & 0 & -\operatorname{sen}\theta \\ 0 & 1 & 0 \\ \operatorname{sen}\theta & 0 & \cos\theta \end{pmatrix} \ , \tag{4.27}$$

obtenemos los generadores

$$\mathsf{S}_x = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & -i \\ 0 & i & 0 \end{pmatrix} \ , \quad \mathsf{S}_y = \begin{pmatrix} 0 & 0 & -i \\ 0 & 0 & 0 \\ i & 0 & 0 \end{pmatrix} \ , \tag{4.28}$$

de R_x y R_y, los subgrupos de rotaciones en torno a los ejes x e y respectivamente.

4.2.2. Rotaciones de funciones y momento angular orbital.

En la discusión precedente los elementos del grupo son matrices que rotan las coordenadas. Cualquier sistema físico que esta siendo descrito se mantiene fijo. Ahora mantengamos fijas las coordenadas y rotemos una función $\psi(x, y, z)$ relativa a nuestras coordenadas fijas. Con R para rotar las coordenadas,

$$\vec{x}\,' = \mathsf{R}\vec{x} \ , \tag{4.29}$$

definimos R por

$$\mathsf{R}\psi(x, y, z) = \psi'(x, y, z) \to \psi(\vec{x}\,') \ . \tag{4.30}$$

En palabras, la matriz R opera sobre la función ψ, creando una *nueva función* ψ' que es numéricamente igual a $\psi(\vec{x}\,')$, donde $\vec{x}\,'$ son las coordenadas rotadas por R. Si R rota las coordenadas en el sentido antihorario, el efecto de la matriz R es rotar el modelo de la función ψ en el sentido antihorario.

Volviendo a las ecuaciones (4.3) y (4.28), consideremos una rotación infinitesimal, $\varphi \to \delta\varphi$. Luego, usando R_z, ecuación (4.3), obtenemos

$$\mathsf{R}_z(\delta\varphi)\psi(x, y, z) = \psi(x + y\delta\varphi, y - x\delta\varphi, z) \ . \tag{4.31}$$

El lado derecho puede ser expandido como una serie de Taylor de primer orden en $\delta\varphi$ para dar

$$\begin{aligned}\mathsf{R}_z(\delta\varphi)\psi(x,y,z) &= \psi(x,y,z) - \delta\varphi\left\{x\frac{\partial\psi}{\partial y} - y\frac{\partial\psi}{\partial x}\right\} + \mathcal{O}(\delta\varphi)^2 \\ &= (1 - i\delta\varphi L_z)\psi(x,y,z)\ ,\end{aligned} \tag{4.32}$$

la expresión diferencial en el paréntesis de llave es iL_z. Ya que una rotación primero en φ y luego en $\delta\varphi$ alrededor del eje z está dado por

$$\mathsf{R}_z(\varphi+\delta\varphi)\psi(x,y,z) = \mathsf{R}_z(\delta\varphi)\mathsf{R}_z(\varphi)\psi(x,y,z) = (1 - i\delta\varphi L_z)\mathsf{R}_z(\varphi)\psi(x,y,z)\ , \tag{4.33}$$

tenemos (como una ecuación de operadores)

$$\frac{\mathsf{R}_z(\varphi+\delta\varphi) - \mathsf{R}_z(\varphi)}{\delta\varphi} = -iL_z\mathsf{R}_z(\varphi)\ . \tag{4.34}$$

El lado izquierdo es justo $d\mathsf{R}_z(\varphi)/\delta\varphi$ (para $\delta\varphi \to 0$). En esta forma la ecuación (4.34) se integra inmediatamente a

$$\mathsf{R}_z(\varphi) = \exp(-i\varphi L_z)\ . \tag{4.35}$$

Note cuidadosamente que $\mathsf{R}_z(\varphi)$ rota funciones (en el sentido antihorario) relativa a las coordenadas fijadas y que L_z es la componente z del momento angular orbital $\vec{L}$. La constante de integración está fijada por la condición de borde $\mathsf{R}_z(0) = 1$.

Si reconocemos que los elementos de matriz

$$L_z = (x,y,z)\mathsf{S}_z\begin{pmatrix}\dfrac{\partial}{\partial x}\\ \dfrac{\partial}{\partial y}\\ \dfrac{\partial}{\partial z}\end{pmatrix}\ , \tag{4.36}$$

claramente L_x, L_y, L_z satisface la misma relación de conmutación

$$[L_i, L_j] = i\varepsilon_{ijk}L_k \tag{4.37}$$

que S_x, S_y, S_z y tienen a la misma constantes de estructura $i\varepsilon_{ijk}$ de SO(3).

4.2.3. Homomorfismo SU(2)-SO(3).

El grupo *unitario especial* SU(2) de matrices unitarias de 2×2 con determinante $+1$ tiene las tres matrices de Pauli, σ_i, como generadores. Por lo tanto SU(2) es de orden 3 y depende de tres parámetros continuos reales ξ, η y ζ los cuales a menudo son llamados los parámetros de *Cayley-Klein*. Sus elementos generales tienen la forma

$$\mathsf{U}_2(\xi,\eta,\zeta) = \begin{pmatrix} e^{i\xi}\cos\eta & e^{i\zeta}\,\mathrm{sen}\,\eta \\ -e^{-i\zeta}\,\mathrm{sen}\,\eta & e^{-i\xi}\cos\eta\end{pmatrix} = \begin{pmatrix} a & b\\ -b^* & a^*\end{pmatrix}\ . \tag{4.38}$$

Es fácil chequear que el determinante $\det(\mathsf{U}_2) = 1$ y que se satisface $\mathsf{U}_2^\dagger \mathsf{U}_2 = \mathsf{1} = \mathsf{U}_2 \mathsf{U}_2^\dagger$.

Para obtener los generadores diferenciamos

$$\begin{aligned} -i\frac{\partial \mathsf{U}_2}{\partial \xi}\bigg|_{\xi=0,\eta=0} &= \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} = \sigma_3 \ , \\ \frac{-i}{\operatorname{sen}\eta}\,\frac{\partial \mathsf{U}_2}{\partial \zeta}\bigg|_{\zeta=0} &= \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} = \sigma_1 \ , \\ -i\frac{\partial \mathsf{U}_2}{\partial \eta}\bigg|_{\zeta=0,\eta=0} &= \begin{pmatrix} 0 & i \\ -i & 0 \end{pmatrix} = \sigma_2 \ . \end{aligned} \tag{4.39}$$

Por supuesto, las matrices de Pauli son todas de traza nula y hermíticas.

Con las matrices de Pauli como generadores, los elementos (U_1, U_2, U_3) de SU(2) pueden ser generados por

$$U_1 = \exp(ia_1\sigma_1/2) \ , \quad U_2 = \exp(ia_2\sigma_2/2) \ , \quad U_3 = \exp(ia_3\sigma_3/2) \ . \tag{4.40}$$

Los tres parámetros a_i son reales. El factor extra $1/2$ está presente en los exponentes ya que $s_i = \sigma_i/2$ satisface las mismas relaciones de conmutación [4]

$$[\mathsf{s}_i, \mathsf{s}_j] = i\varepsilon_{ijk}\mathsf{s}_k \tag{4.41}$$

como el momento angular en la ecuación (4.37).

La ecuación (4.3) da un operador de rotación para rotar las coordenadas cartesianas en el espacio tridimensional. Usando la matriz de momento angular s_3, tenemos el correspondiente operador de rotación en el espacio de dos dimensiones (complejo) $R_z(\varphi) = \exp(i\varphi\sigma_3/2)$.

Para rotar una función de onda vectorial de dos componentes (spinor) o una partícula de spin $1/2$ relativa a coordenadas fijas, el operador de rotación es $R_z(\varphi) = \exp(-i\varphi\sigma_3/2)$ de acuerdo a la ecuación (4.35).

Usando la ecuación (4.40) la identidad de Euler, la ecuación (3.161), obtenemos

$$\mathsf{U}_j = \cos\left(\frac{a_j}{2}\right) + i\sigma_j \operatorname{sen}\left(\frac{a_j}{2}\right) \ .$$

Aquí el parámetro a_j aparece como un ángulo, el coeficiente de una matriz tipo momento angular φ en la ecuación (4.26). Con esta identificación de los exponenciales, la forma general de la matriz SU(2) (para rotar funciones más que coordenadas) podría ser escrita como

$$\mathsf{U}(\alpha, \beta, \gamma) = \exp(-i\gamma\sigma_3/2)\exp(-i\beta\sigma_2/2)\exp(-i\alpha\sigma_1/2) \ .$$

Como vimos, los elementos de SU(2) describen rotaciones en un espacio complejo bidimensional que deja invariante a $|z_1|^2 + |z_2|^2$. El determinante es $+1$. Hay tres parámetros independientes. Nuestro grupo ortogonal real SO(3) de determinante $+1$, claramente describe rotaciones comunes en el espacio tridimensional con la importante característica de dejar invariante a $x^2 + y^2 + z^2$. También hay tres parámetros independientes. Las interpretaciones de rotación y la igualdad de números de parámetros sugiere la existencia de alguna clase de correspondencia entre los grupos SU(2) y SO(3). Aquí desarrollamos esta correspondencia.

[4] Las constantes de estructuras $(i\varepsilon_{ijk})$ conducen a las representaciones de SU(2) de dimensión $2J = 1$ para generadores de dimensión $2j + 1$, con $J = 0, 1/2, 1, \ldots$. Los casos con J entero también conducen a las representaciones de SO(3).

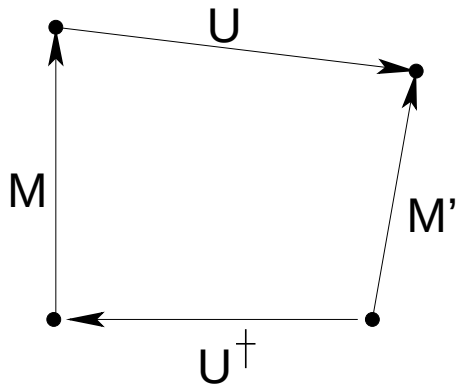


Figura 4.2: Ilustración de $\mathsf{M}' = \mathsf{UMU}^\dagger$ ecuación (4.42).

La operación SU(2) sobre una matriz está dada por una transformación unitaria, la ecuación (4.5), con $\mathsf{R} = \mathsf{U}$ y la figura (4.2)

$$\mathsf{M}' = \mathsf{UMU}^\dagger \ . \tag{4.42}$$

Tomando M como una matriz de 2×2, notemos que cualquier matriz de 2×2 puede ser escrita como una combinación lineal de la matriz unidad y las tres matrices de Pauli. Sea M la matriz de traza cero,

$$\mathsf{M} = x\sigma_1 + y\sigma_2 + z\sigma_3 = \begin{pmatrix} z & x - iy \\ x + iy & -z \end{pmatrix} \ , \tag{4.43}$$

la matriz unidad no entra. Ya que la traza es invariante bajo transformaciones unitarias, M' debería tener la misma forma,

$$\mathsf{M}' = x'\sigma_1 + y'\sigma_2 + z'\sigma_3 = \begin{pmatrix} z' & x' - iy' \\ x' + iy' & -z' \end{pmatrix} \ . \tag{4.44}$$

El determinante también es invariante bajo una transformación unitaria. Por lo tanto

$$-(x^2 + y^2 + z^2) = -(x'^2 + y'^2 + z'^2) \ , \tag{4.45}$$

o $(x^2 + y^2 + z^2)$ es invariante bajo esta operación de SU(2), como con SO(3). SU(2) debe, por lo tanto, describir una rotación. Esto sugiere que SU(2) y SO(3) pueden ser isomórficos o homomórficos.

Aproximemos el problema de qué rotación describe SU(2), considerando casos especiales. Retomando la ecuación (4.38) sea $a = e^{i\xi}$ y $b = 0$, o

$$\mathsf{U}_z = \begin{pmatrix} e^{i\xi} & 0 \\ 0 & e^{-i\xi} \end{pmatrix} \ . \tag{4.46}$$

En anticipación de la ecuación (4.50), esta U le está dado un subíndice z.

Realizando una transformación unitaria sobre cada una de las tres matrices de Pauli, tenemos

$$\begin{aligned} \mathsf{U}_z\sigma_1\mathsf{U}_z^\dagger &= \begin{pmatrix} e^{i\xi} & 0 \\ 0 & e^{-i\xi} \end{pmatrix}\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}\begin{pmatrix} e^{-i\xi} & 0 \\ 0 & e^{i\xi} \end{pmatrix} \\ &= \begin{pmatrix} 0 & e^{2i\xi} \\ e^{-2i\xi} & 0 \end{pmatrix} . \end{aligned} \tag{4.47}$$

Reexpresamos este resultado en términos de las matrices de Pauli para obtener

$$\mathsf{U}_z x\sigma_1\mathsf{U}_z^\dagger = x\cos 2\xi\sigma_1 - x\,\mathrm{sen}\,2\xi\sigma_2 . \tag{4.48}$$

Similarmente,

$$\begin{aligned} \mathsf{U}_z y\sigma_2\mathsf{U}_z^\dagger &= y\,\mathrm{sen}\,2\xi\sigma_1 - y\cos 2\xi\sigma_2 , \\ \mathsf{U}_z z\sigma_3\mathsf{U}_z^\dagger &= z\sigma_3 . \end{aligned} \tag{4.49}$$

A partir de esta expresión de doble ángulo vemos que podríamos comenzar con el ángulo medio: $\xi = \alpha/2$. Entonces, de las ecuaciones (4.42)–(4.44), (4.48) y (4.49),

$$\begin{aligned} x' &= \quad x\cos\alpha + y\,\mathrm{sen}\,\alpha \\ y' &= -x\,\mathrm{sen}\,\alpha + y\cos\alpha \\ z' &= z . \end{aligned} \tag{4.50}$$

La transformación unitaria de 2×2 usando $U_z(\alpha/2)$ es equivalente al operador de rotación $R(\alpha)$ de la ecuación (4.3).

El establecimiento de la correspondencia de

$$\mathsf{U}_y(\beta/2) = \begin{pmatrix} \cos\beta/2 & \mathrm{sen}\,\beta/2 \\ -\,\mathrm{sen}\,\beta/2 & \cos\beta/2 \end{pmatrix} \tag{4.51}$$

y $R_y(\beta)$ y de

$$\mathsf{U}_x(\varphi/2) = \begin{pmatrix} \cos\varphi/2 & i\,\mathrm{sen}\,\varphi/2 \\ i\,\mathrm{sen}\,\varphi/2 & \cos\varphi/2 \end{pmatrix} \tag{4.52}$$

y $R_x(\varphi)$ pueden calcularse como ejercicio. Podemos notar que $\mathsf{U}_k(\psi/2)$ tiene la forma general

$$\mathsf{U}_k(\psi/2) = \mathsf{1}\cos\psi/2 + i\sigma_k\,\mathrm{sen}\,\psi/2 , \tag{4.53}$$

donde $k = x, y, z$.

La correspondencia

$$\mathsf{U}_z\left(\frac{\alpha}{2}\right) = \begin{pmatrix} e^{i\alpha/2} & 0 \\ 0 & e^{-i\alpha/2} \end{pmatrix} \leftrightarrow \begin{pmatrix} \cos\alpha & \mathrm{sen}\,\alpha & 0 \\ -\sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix} = \mathsf{R}_z(\alpha) , \tag{4.54}$$

no es una simple correspondencia uno a uno. Específicamente, como α en R_z recorre desde 0 a 2π, el parámetro U_z, $\alpha/2$, recorre desde 0 a π . Encontramos que

$$\begin{aligned} \mathsf{R}_z(\alpha + 2\pi) &= \mathsf{R}_z(\alpha) \\ \mathsf{U}_z(\alpha/2 + \pi) &= \begin{pmatrix} -e^{i\alpha/2} & 0 \\ 0 & -e^{-i\alpha/2} \end{pmatrix} = -\mathsf{U}_z(\alpha/2) . \end{aligned} \tag{4.55}$$

Por lo tanto *ambos* $U_z(\alpha/2)$ y $U_z(\alpha/2+\pi) = -U_z(\alpha/2)$ corresponde a $R_z(\alpha)$. La correspondencia es de 2 a 1, o SU(2) y SO(3) son homomórficos. Este establecimiento de la correspondencia entre las representaciones de SU(2) y de aquella SO(3) significa que las representaciones conocidas de SU(2) automáticamente nos proporciona de las representaciones de SO(3).

Combinando las rotaciones, encontramos que una transformación unitaria usada

$$\mathsf{U}(\alpha,\beta,\gamma) = \mathsf{U}_z(\gamma/2)\mathsf{U}_y(\beta/2)\mathsf{U}_z(\alpha/2) \ , \tag{4.56}$$

corresponde a la rotación general de Euler $R_z(\gamma)R_y(\beta)R_z(\alpha)$. Por multiplicación directa,

$$\begin{aligned}\mathsf{U}(\alpha,\beta,\gamma) &= \begin{pmatrix} e^{i\gamma/2} & 0 \\ 0 & e^{-i\gamma/2}\end{pmatrix}\begin{pmatrix} \cos\beta/2 & \operatorname{sen}\beta/2 \\ -\operatorname{sen}\beta/2 & \cos\beta/2\end{pmatrix}\begin{pmatrix} e^{i\alpha/2} & 0 \\ 0 & e^{-i\alpha/2}\end{pmatrix} \\ &= \begin{pmatrix} e^{i(\gamma+\alpha)/2}\cos\beta/2 & e^{i(\gamma-\alpha)/2}\operatorname{sen}\beta/2 \\ -e^{-i(\gamma-\alpha)/2}\operatorname{sen}\beta/2 & e^{-i(\gamma+\alpha)/2}\cos\beta/2\end{pmatrix} \ .\end{aligned} \tag{4.57}$$

Esta es nuestra forma general alternativa, la ecuación (4.38), con

$$\xi = \frac{(\gamma+\alpha)}{2} \ , \quad \eta = \frac{\beta}{2} \ , \quad \zeta = \frac{(\gamma-\alpha)}{2} \ . \tag{4.58}$$

De la ecuación (4.57) podemos identificar los parámetros de la ecuación (4.38) como

$$\begin{aligned} a &= e^{i(\gamma+\alpha)/2}\cos\beta/2 \\ b &= e^{i(\gamma-\alpha)/2}\operatorname{sen}\beta/2 \end{aligned} \tag{4.59}$$

4.2.4. SU(2) isospin y el octeto SU(3).

En el tratamiento de las partículas con interacciones fuertes de Física nuclear y de altas energías que conducen al grupo de SU(2) de isospin y la simetría de sabor SU(3), podríamos mirar el momento angular y el grupo de rotación SO(3) por una analogía. Supongamos que tenemos un electrón en un potencial atractivo esféricamente simétrico de algún núcleo atómico. La función de onda para el electrón puede ser caracterizada por tres números cuánticos n, l, m, que están relacionados con los autovalores de los operadores conservados H, L^2, L_z. La energía,[5] sin embargo, es $2l+1$ veces degenerada, dependiendo solamente de n y l. La razón para esta degeneración puede ser expresada de dos maneras equivalentes:

1. El potencial es simétricamente esférico, independiente de θ, φ.
2. El Hamiltoniano de Schrödinger $-(\hbar^2/2m_e)\nabla^2 + V(r)$ es invariante bajo rotaciones espaciales ordinarias SO(3).

Como una consecuencia de la simetría esférica del potencial $V(r)$, el momento angular orbital $\vec{L}$ es conservado. En la sección 4.2 las componentes cartesianas de $\vec{L}$ están identificadas como los generadores del grupo de rotación SO(3). En vez de representar L_x, L_y, L_z por operadores, usaremos matrices. Las matrices L_i son matrices $(2l+1)\times(2l+1)$ con la misma

[5]Para un potencial de Coulomb puro la energía depende sólo de n.

dimensión del número de estados degenerados. La dimensión $2l+1$ está identificada con los estados degenerados $2l+1$.

Esta degeneración es removida por un campo magnético $\vec{B}$, hecho conocido como el efecto Zeeman. Esta interacción magnética añade un término al Hamiltoniano que *no* es invariante bajo SO(3). Este es un término que quiebra la simetría.

En el caso de partículas con interacción fuerte (protones, neutrones, etc.) no podemos seguir la analogía directamente, ya que todavía no entendemos completamente las interacciones nucleares. La fuerza fuerte está descrita por la teoría gauge de Yang-Mills, basada sobre la simetría de color SU(3) llamada cromo-dinámica cuántica o abreviada QCD. Sin embargo, QCD es una teoría no lineal y por lo tanto, es complicada a grandes distancias y baja energía, por lo que permanece no resuelta. Por lo tanto, no conocemos el Hamiltoniano, en vez de esto, volveremos a la analogía.

En los años 1930, después del descubrimiento del neutrón, Heinsenberg propuso que las fuerzas nucleares eran cargas independientes. Los neutrones difieren en masa de los protones solamente en un 1.6 %. Si esta pequeña diferencia es ignorada, el neutrón y el protón podrían ser considerados como dos estados de cargas (o isospin) de un doblete, llamado nucleón. El isospin I tiene proyección en el eje z $I_3 = 1/2$ para el protón $I_3 = -1/2$, para el neutrón. El isospin no tiene nada que ver con el spin (el momento angular intrínseco de una partícula), pero las dos componentes del estado de isospin obedecen las mismas relaciones matemáticas que el estado de spin 1/2. Para el nucleón, $\mathsf{I} = \tau/2$, son las matrices usuales de Pauli además los estados de isospin $(\pm 1/2)$ son autovectores de la matriz de Pauli $\tau_3 = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$. Similarmente, los tres estados de carga del pión π^+, π^0 , π^- forman un triplete. El pión es la partícula más liviana con interacción fuerte y es la mediadora de la fuerza nuclear a distancia, como el fotón es partícula que media la fuerza electromagnética. La interacción fuerte trata igualmente a miembros de esa familia de partículas, o multipletes y conserva el isospin. La simetría es el grupo isospin SU(2).

El octuplete mostrado en la tabla 4.1 llama la atención[6]. Los números cuánticos conservados que son análogos y generalizaciones de L_z y L^2 de SO(3) son I_3 e I^2 para el isospin, e Y para *hipercarga*. Las partículas pueden ser agrupadas dentro de multipletes de carga o de isospin. Entonces, la hipercarga puede ser tomada como dos veces el promedio de carga del multiplete. Para el nucleón, *i.e.*, el doblete neutrón–protón, $Y = 2 \cdot \frac{1}{2}(0+1) = 1$. Los valores de la hipercarga y los del isospin son listados en la tabla 4.1 para bariones como el nucleón y sus compañeros (aproximadamente degenerados). Ellos forman un octeto como lo muestra la figura 4.3. En 1961 Gell-Mann, e independientemente Ne'eman, sugirieron que la interacción fuerte debe ser (aproximadamente) invariante bajo un grupo espacial tridimensional unitario, SU(3), esto es, tienen *simetría de sabor* SU(3).

La elección de SU(3) estuvo basada primero sobre los dos números cuánticos conservados e independientes $H_1 = I_3$ y $H_2 = Y$ (*i.e.*, generados con $[I_3, Y] = 0$), que llaman para un grupo de rango 2. Segundo, el grupo ha tenido una representación de ocho dimensiones para tener en cuenta a los cercanamente degenerados bariones y cuatro octetos similares para los mesones. En un sentido, SU(3) es la generalización más simple del isospin SU(2). Tres de sus generadores son matrices hermíticas de 3×3 de traza nula que contienen las matrices de

[6]Todas las masas están dadas en unidades de energía.

		Masa [MeV]	Y	I	I_3
	Ξ^-	1321.32			$-\frac{1}{2}$
Ξ			-1	$\frac{1}{2}$	
	Ξ^0	1314.9			$+\frac{1}{2}$
	Σ^-	1197.43			-1
Σ	Σ^0	1192.55	0	1	0
	Σ^+	1189.37			+1
Λ	Λ	1115.63	0	0	0
	n	939.566			$-\frac{1}{2}$
N			1	$\frac{1}{2}$	
	p	938.272			$+\frac{1}{2}$

Cuadro 4.1: Bariones con spin $\frac{1}{2}$ y paridad par

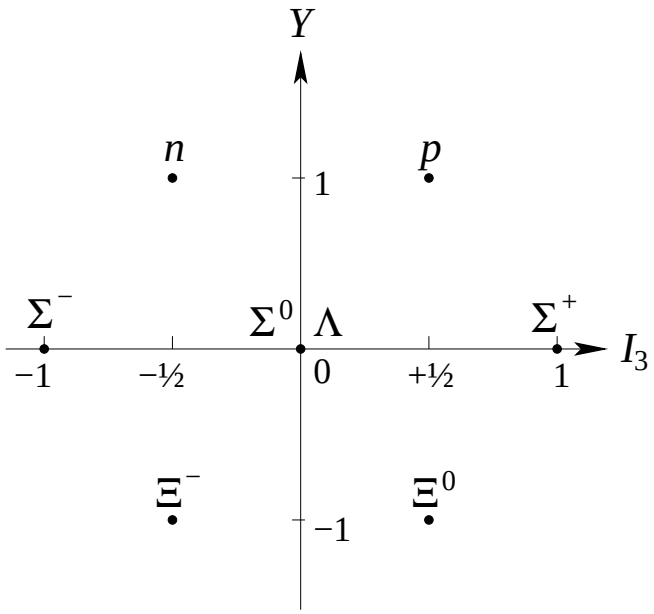


Figura 4.3: Octeto bariónico diagrama de peso para SU(3).

Pauli de 2×2 para los isospin τ_i en la esquina superior izquierda.

$$\lambda_i = \begin{pmatrix} \tau_i & & 0 \\ & & 0 \\ 0 & 0 & 0 \end{pmatrix} \quad , \quad i = 1, 2, 3 \ . \tag{4.60}$$

De este modo, el grupo del isospin SU(2) es un subgrupo de SU(3) con $I_3 = \lambda_3/2$. Otros cuatro generadores tienen los no diagonales 1 de τ_1 e i, $-i$ de τ_2 en todas las otras posibles ubicaciones para formar las matrices hermíticas 3×3 de traza nula.

$$\begin{aligned} \lambda_4 &= \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix} , \qquad \lambda_5 = \begin{pmatrix} 0 & 0 & -i \\ 0 & 0 & 0 \\ i & 0 & 0 \end{pmatrix} , \\ \lambda_6 &= \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} , \qquad \lambda_7 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & -i \\ 0 & i & 0 \end{pmatrix} . \end{aligned} \tag{4.61}$$

El segundo generador diagonal tiene la matriz unidad bidimensional $\mathbf{1}_2$ en la esquina superior izquierda, la cual la hace claramente independiente del subgrupo SU(2) isospin ya que su traza no nula en el subespacio, y -2 en el lugar de la tercera diagonal la hace traza nula,

$$\lambda_8 = \frac{1}{\sqrt{3}} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -2 \end{pmatrix} . \tag{4.62}$$

Generalmente hay $3^2 - 1 = 8$ generadores para SU(3) el cual tiene orden 8. De los conmutadores de esos generadores pueden obtenerse fácilmente las constantes de estructura de SU(3).

Volviendo a la simetría de sabor SU(3), imaginemos que el Hamiltoniano para nuestro octeto de bariones están compuesto de tres partes

$$H = H_{\text{fuerte}} + H_{\text{medio}} + H_{\text{electromagnético}} . \tag{4.63}$$

La primera parte, H_{fuerte}, tiene la simetría SU(3) y conduce a la degeneración ocho. La introducción del término de quiebre de simetría, H_{medio}, remueve parte de la degeneración dando los cuatro multipletes del isospin (Ξ^-, Ξ^0), $(\Sigma^-, \Sigma^0, \Sigma^+)$, Λ, y $N = (p, n)$, con diferentes masas. Estos aún son multipletes ya que H_{medio} tiene la simetría del isospin SU(2). Finalmente, la presencia de fuerzas dependientes de la carga separan los multipletes de isospin y remueve la última degeneración. Esta secuencia se muestra en la figura 4.4

Aplicando teoría de perturbación de primer orden de Mecánica Cuántica, relaciones simples de masas de bariónicas pueden ser calculadas. Quizás el suceso más espectacular de este modelo SU(3) ha sido su predicción de nuevas partículas. En 1961 cuatro mesones K y tres π (todos pseudo escalares; spin 0, paridad impar) sugieren otro octeto, similar al del octeto bariónico. SU(3) predice un octavo mesón η, de masa 563 MeV. El mesón η, con una masa determinada experimentalmente de 548 MeV, fue encontrado poco después. Agrupamientos de nueve de los bariones más pesados (todos con spin 3/2, con paridad par) sugirió un multiplete de 10 miembros o un decaplete de SU(3). El décimo barión faltante fue predicho con una masa cercana a 1680 MeV y una carga negativa. En 1964 Ω^- cargada negativamente con masa (1675±12) MeV fue descubierta.

La representación de octeto no es la más simple para SU(3). La representación más simple son triangulares, como se muestra en la figura 4.5, a partir de las cuales todas las otras pueden ser generadas por acoplamiento del momento angular generalizado. La representación

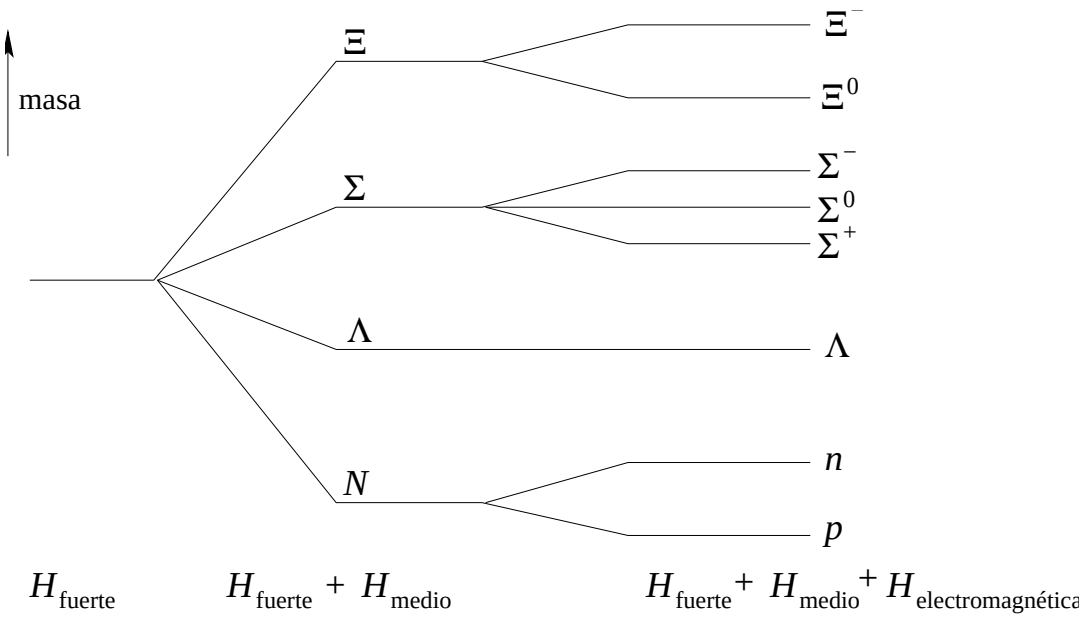


Figura 4.4: Separación de masa bariónica.

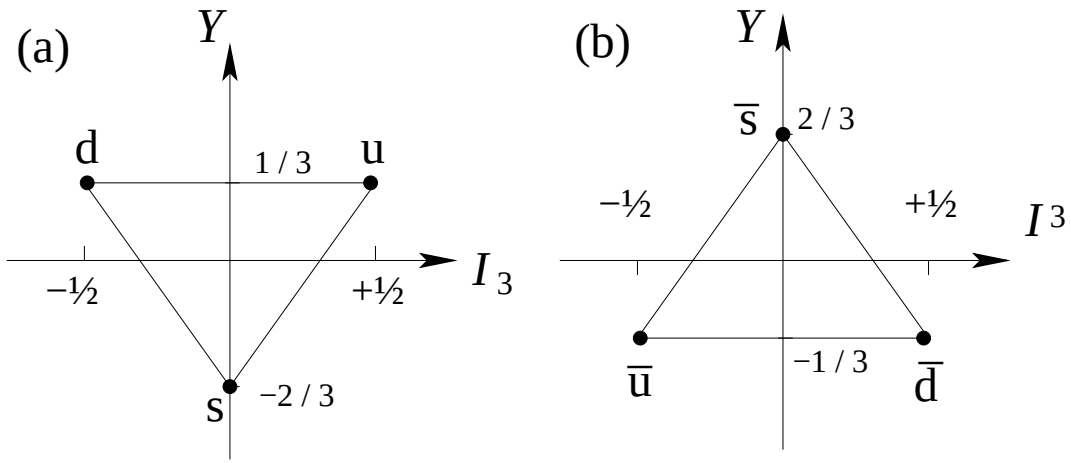


Figura 4.5: (a) Representación fundamental de SU(3), el diagrama de peso para los quark u, d, s; (b) diagrama de peso para los antiquark $\bar{u}$, $\bar{d}$, $\bar{s}$.

fundamental en la figura 4.5 (a) contiene los quark u (arriba) y d (abajo) y el s (extrañeza), y figura 4.5 (b) los correspondientes antiquarks. Los octetos de mesones pueden ser obtenidos a partir de la representación de quark como pares $q\bar{q}$, $3^2 = 8+1$, esto sugiere que los mesones contienen quarks (y antiquarks) como sus constituyentes. El modelo de quarks resultante da una exitosa descripción de la espectroscopia hadrónica. La solución de sus problemas con el principio de exclusión de Pauli, eventualmente, conduce a la teoría de gauge de SU(3)-color de las interacciones fuertes, llamada cromo-dinámica cuántica o QCD.

Para mantener la teoría de grupo en su real perspectiva, podríamos enfatizar que la teoría de grupo identifica y formaliza las simetrías. Ella clasifica partículas (y algunas veces predice).

Pero a parte de decir que una parte del Hamiltoniano tiene simetría SU(2) y otra parte tiene simetría SU(3), la teoría de grupo no dice nada acerca de la interacción de las partículas. Recuerde que la afirmación de que el potencial atómico es esféricamente simétrico no nos dice nada acerca de la dependencia radial del potencial o de su función de onda. En contraste, en una teoría de gauge la interacción es mediada por bosones vectoriales (como el fotón media en la electrodinámica cuántica) y determinado únicamente por la derivada covariante de gauge.

4.3. Momento angular orbital.

El concepto clásico de momento angular $\vec{L}_{\text{clásico}} = \vec{r} \times \vec{p}$ es mostrado en el capítulo de vectores para presentar el producto cruz. Siguiendo con la representación usual de Schrödinger de la Mecánica Cuántica, el momento lineal clásico $\vec{p}$ es reemplazado por el operador $-i\vec{\nabla}$. El operador de momento angular en la mecánica cuántica se convierte en[7]

$$\vec{L}_{QM} = -i\vec{r} \times \vec{\nabla} \ . \tag{4.64}$$

Las componentes del momento angular satisfacen las relaciones de conmutación

$$[L_i, L_j] = i\varepsilon_{ijk} L_k \ . \tag{4.65}$$

El ε_{ijk} es el símbolo de Levi-Civita. Una suma sobre el índice k es sobreentendida.

El operador diferencial correspondiente al cuadrado del momento angular

$$\vec{L}^{\,2} = \vec{L} \cdot \vec{L} = L_x^2 + L_y^2 + L_z^2 \ , \tag{4.66}$$

puede ser determinado a partir de

$$\vec{L} \cdot \vec{L} = (\vec{r} \times \vec{p}) \cdot (\vec{r} \times \vec{p}) \ , \tag{4.67}$$

la cual puede verificarse como ejercicio. Ya que $\vec{L}^{\,2}$ es un escalar rotacional, $[\vec{L}^{\,2}, L_i] = 0$, el cual también puede ser verificado directamente.

La ecuación (4.65) presenta las relaciones de conmutación básicas de los componentes del momento angular en Mecánica Cuántica. Por cierto, dentro del marco de la Mecánica Cuántica y la teoría de grupo, estas relaciones de conmutación definen un operador de momento angular.

4.3.1. Operadores de subida y bajada.

Comencemos con una aproximación más general, donde el momento angular $\vec{J}$ lo consideramos que puede representar un momento angular orbital $\vec{L}$, un spin $\sigma/2$, o un momento angular total $\vec{L} + \sigma/2$, etc. Supongamos que

1. J es un operador hermítico cuyas componentes satisfacen las relaciones de conmutación

$$[J_i, J_j] = i\varepsilon_{ijk} J_k \ , \qquad [\vec{J}^{\,2}, J_i] = 0 \ . \tag{4.68}$$

[7]Por simplicidad, $\hbar = 1$. Esto significa que el momento angular es medido en unidades de $\hbar$.

2. $|\lambda M\rangle$ es simultáneamente una autofunción normalizada (o autovector) de J_z con autovalor M y una autofunción de $\vec{J}^2$,

$$J_z|\lambda M\rangle = M|\lambda M\rangle \ , \qquad \vec{J}^2|\lambda M\rangle = \lambda|\lambda M\rangle \ . \tag{4.69}$$

Mostraremos ahora que $\lambda = J(J+1)$. Este tratamiento ilustrará la generalidad y potencia de las técnicas de operadores particularmente el uso de operadores de subida y bajada.

Un operador de subida o bajada se define como

$$J_+ = J_x + iJ_y \ , \qquad J_- = J_x - iJ_y \ . \tag{4.70}$$

En términos de ese operador $\vec{J}^2$ puede ser reescrito como

$$\vec{J}^2 = \frac{1}{2}(J_+J_- + J_-J_+) + J_z^2 \ . \tag{4.71}$$

A partir de las relaciones de conmutación, encontramos que

$$[J_z, J_+] = +J_+ \ , \quad [J_z, J_-] = -J_- \ , \quad [J_+, J_-] = 2J_z \ . \tag{4.72}$$

Ya que J_+ conmuta con $\vec{J}^2$ (hágalo como ejercicio)

$$\vec{J}^2(J_+|\lambda M\rangle) = J_+(\vec{J}^2|\lambda M\rangle) = \lambda(J_+|\lambda M\rangle) \ . \tag{4.73}$$

Por lo tanto, $J_+|\lambda M\rangle$ todavía es una autofunción de $\vec{J}^2$ con autovalores λ, y similarmente para $J_-|\lambda M\rangle$. Pero de la ecuación (4.72)

$$J_zJ_+ = J_+(J_z+1) \ , \tag{4.74}$$

o

$$J_z(J_+|\lambda M\rangle) = J_+(J_z+1)|\lambda M\rangle = (M+1)(J_+|\lambda M\rangle) \ . \tag{4.75}$$

Por lo tanto, $J_+|\lambda M\rangle$ todavía es una autofunción de J_z con autovalores $M+1$. J_+ ha elevado el autovalor en 1 y por eso es llamado *operador de subida*. Similarmente, J_- baja los autovalores en 1 y a menudo es llamado *operador de bajada*.

Tomando los valores esperados y usando $J_x^\dagger = J_x$, $J_y^\dagger = J_y$,

$$\langle\lambda M|\vec{J}^2 - J_z^2|\lambda M\rangle = \langle\lambda M|J_x^2 + J_y^2|\lambda M\rangle = |\ J_x|\lambda M\rangle\ |^2 + |\ J_y|\lambda M\rangle\ |^2 \ ,$$

vemos que $\lambda - M^2 \geq 0$, tal que M es ligado. Sea J el más grande valor de M. Luego $J_+|\lambda J\rangle = 0$, lo cual implica que $J_-J_+|\lambda J\rangle = 0$. Luego combinando las ecuaciones (4.71) y (4.72) obtenemos

$$\vec{J}^2 = J_-J_+ + J_z(J_z+1) \ , \tag{4.76}$$

encontramos que a partir de la ecuación (4.76)

$$0 = J_-J_+|\lambda M = J\rangle = (\vec{J}^2 - J_z^2 - J_z)|\lambda M = J\rangle = (\lambda - J^2 - J)|\lambda M = J\rangle \ .$$

Por lo tanto

$$\lambda = J(J+1) \geq 0; \tag{4.77}$$

con un J no negativo. Ahora reetiquetaremos los estados $|\lambda M\rangle = |JM\rangle$. Similarmente, sea J' el más pequeño de los M. Entonces $J_-|JJ'\rangle = 0$. A partir de

$$\vec{J}^2 = J_+J_- - J_z(J_z+1) \ , \tag{4.78}$$

vemos que

$$0 = J_+J_-|JJ'\rangle = (\vec{J}^2 + J_z - J_z^2)|JJ'\rangle = (\lambda + J' - J'^2)|JJ'\rangle \ . \tag{4.79}$$

De manera que

$$\lambda = J(J+1) = J'(J'-1) = (-J)(-J-1) \ .$$

Así $J' = -J$, y M corre en pasos enteros desde $-J$ a $+J$,

$$-J \leq M \leq +J \ . \tag{4.80}$$

Comenzando desde $|JJ\rangle$ y aplicando J_- repetidas veces, alcanzaremos todos los otros estados $|JM\rangle$. De manera que $|JM\rangle$ forma una representación irreductible; M varía y J está fijo.

Entonces usando las ecuaciones (4.68), (4.76) y (4.78) obtenemos

$$\begin{aligned} J_-J_+|JM\rangle &= [J(J+1) - M(M+1)]|JM\rangle = (J-M)(J+M+1)|JM\rangle \ , \\ J_+J_-|JM\rangle &= [J(J+1) - M(M-1)]|JM\rangle = (J+M)(J-M+1)|JM\rangle \ . \end{aligned} \tag{4.81}$$

Como J_+ y J_- son hermíticos conjugados,

$$J_+^\dagger = J_- \ , \qquad J_-^\dagger = J_+ \ , \tag{4.82}$$

los autovalores o valores esperados en la ecuación (4.81) deberían ser positivos o cero.

Ya que J_+ aumenta el autovalor de M a $M+1$, reetiquetaremos la autofunción resultante $|JM+1\rangle$. La normalización está dada por la ecuación (4.81) como

$$J_+|JM\rangle = \sqrt{(J-M)(J+M+1)}|JM+1\rangle \ , \tag{4.83}$$

tomando la raíz cuadrada positiva y no introduciendo ningún factor de fase. Por los mismos argumentos

$$J_-|JM\rangle = \sqrt{(J+M)(J-M+1)}|JM-1\rangle \ . \tag{4.84}$$

Finalmente, ya que M va desde $-J$ a $+J$ en pasos unitarios, $2J$ *debería ser un número entero*. J es por lo tanto un entero o la mitad de un entero impar. Como hemos visto, el momento angular orbital está descrito con J entero. A partir de los spin de algunas partículas fundamentales y de algunos núcleos, tenemos que $J = 1/2, 3/2, 5/2, \ldots$ Nuestro momento angular está cuántizado esencialmente como un resultado de relaciones de conmutaciones. En coordenadas polares esféricas θ, φ las funciones $\langle\theta,\varphi|lm\rangle = Y_l^m(\theta,\varphi)$ son armónicos esféricos.

4.3.2. Resumen de grupos y álgebras de Lie.

Las relaciones de conmutaciones generales, ecuación (4.14) en la sección 4.2, para un grupo de Lie clásico [SO(n) y SU(n) en particular] pueden ser simplificadas para verse más como la ecuación (4.72) para SO(3) y SU(2) en la sección 4.3.

Álgebra de Lie	A_l	B_l	D_l
Grupo de Lie	SU(l+1)	SO(2l+1)	SO(2l)
rango	l	l	l
orden	l(l+2)	l(2l+1)	l(2l-1)

Cuadro 4.2: Rango y orden de grupos rotacionales y unitarios.

Primero escogemos generadores H_i que sean linealmente independientes y que conmuten entre sí, estos son generalizaciones de J_z de SO(3) y SU(2). Sea l el número máximo de tales H_i con

$$[H_i, H_k] = 0 \ . \tag{4.85}$$

Entonces l es llamado el *rango* del grupo Lie G o de su álgebra. El rango y la dimensión u orden de algunos grupos Lie son dados en la tabla 4.2. Todos los otros generadores E_α son operadores de subida y bajada con respecto a todos los H_i, tal que

$$[H_i, E_\alpha] = \alpha_i E_\alpha \ . \tag{4.86}$$

El conjunto de los $(\alpha_1, \alpha_2, \ldots, \alpha_l)$ son llamados los *vectores raíces*.

Ya que los H_i conmutan entre sí, ellos pueden ser diagonalizados simultáneamente. Ellos nos dan un conjunto de autovalores $m_1, m_2, \ldots, m_l$. Al conjunto $(m_1, m_2, \ldots..m_l)$ se les llama vectores de peso de una representación irreductible. Hay l operadores invariantes C_i, llamados *operadores de Casimir*, los cuales conmutan con todos los generadores y son generalizaciones de J^2,

$$[C_i, H_j] = 0 \ , \qquad [C_i, E_\alpha] = 0 \ . \tag{4.87}$$

El primero, C_1, es una función cuadrática de los generadores, los otros son más complicados. Ya que C_i conmuta con todos los H_j, ellos pueden ser diagonalizados simultáneamente con los H_j. Sus autovalores $c_1, c_2, \ldots, c_l$ caracterizan las representaciones irreductibles y permanecen constantes mientras los vectores de peso varían sobre una representación irreductible particular. Por lo tanto, la autofunción general puede ser escrita como

$$|(c_1, c_2, \ldots, c_l)m_1, m_2, \ldots, m_l\rangle \ , \tag{4.88}$$

generalizando $|JM\rangle$ de SO(3) y SU(2). Sus ecuaciones de autovalores son

$$H_i|(c_1, c_2, \ldots, c_l)m_1, m_2, \ldots, m_l\rangle = m_i|(c_1, c_2, \ldots, c_l)m_1, m_2, \ldots, m_l\rangle \ , \tag{4.89a}$$
$$C_i|(c_1, c_2, \ldots, c_l)m_1, m_2, \ldots, m_l\rangle = c_i|(c_1, c_2, \ldots, c_l)m_1, m_2, \ldots, m_l\rangle \ . \tag{4.89b}$$

Ahora podemos mostrar que $E_\alpha|(c_1, c_2, \ldots, c_l)m_1, m_2, \ldots, m_l\rangle$ tiene los vector peso $(m_1 + \alpha_1, m_2+\alpha_2, \ldots, m_l+\alpha_l)$ usando las relaciones de conmutación, la ecuación (4.86), en conjunto con las ecuaciones (4.89a) y (4.89b),

$$\begin{aligned} H_i E_\alpha|(c_1, c_2, \ldots, c_l)m_1, m_2, \ldots, m_l\rangle &= (E_\alpha H_i + [H_i, E_\alpha])|(c_1, c_2, \ldots, c_l)m_1, m_2, \ldots, m_l\rangle \\ &= (m_i + \alpha_i)E_\alpha|(c_1, c_2, \ldots, c_l)m_1, m_2, \ldots, m_l\rangle \ . \end{aligned} \tag{4.90}$$

Por lo tanto

$$E_\alpha|(c_1, c_2, \ldots, c_l)m_1, m_2, \ldots, m_l\rangle \sim |(c_1, c_2, \ldots, c_l)m_1 + \alpha_1, m_2 + \alpha_2, \ldots, m_l + \alpha_l\rangle$$

estas son las generalizaciones de las ecuaciones (4.83) y (4.84) a partir de SO(3). Esos cambios de autovalores por el operador E_α son llamados sus reglas de selección en mecánica cuántica.

4.4. Grupo homogéneo de Lorentz.

En relatividad especial requerimos que nuestras leyes físicas sean covariantes[8] bajo

a. translaciones en el tiempo y en el espacio,

b. rotaciones en el espacio real tridimensional, y

c. transformaciones de Lorentz.

El requerimiento para la covarianza bajo translaciones está basada en la homogeneidad del espacio y el tiempo. Covarianza bajo rotaciones es una afirmación de la isotropía del espacio. El requerimiento de la covarianza de Lorentz viene de la relatividad especial. Todas estas tres transformaciones en conjunto forman el grupo inhomogéneo de Lorentz o el grupo de Poincaré. Aquí excluimos las translaciones. Las rotaciones espaciales y las transformaciones de Lorentz forman un grupo, el grupo homogéneo de Lorentz.

Primero generamos un subgrupo, las transformaciones de Lorentz en el cual la velocidad relativa $\vec{v}$ está a lo largo del eje $x = x_1$. El generador puede ser determinado considerando un marco de referencia espacio-temporal moviéndose con una velocidad relativa infinitesimal δv. Las relaciones son similares a aquellas para rotaciones en el espacio real, excepto que aquí el ángulo de rotación es imaginario puro.

Las transformaciones de Lorentz son lineales no sólo en el espacio de coordenadas x_i sino que también en el tiempo t. Ellas se originan a partir de las ecuaciones de Maxwell de la electrodinámica, las cuales son invariantes bajo la transformaciones de Lorentz, como veremos luego. Las transformaciones de Lorentz dejan invariante la forma cuadrática siguiente $c^2t^2 - x_1^2 - x_2^2 - x_3^2 = x_0^2 - x_1^2 - x_2^2 - x_3^2$ donde $x_0 = ct$. Vemos esto si encendemos una fuente de luz en el origen del sistema de coordenadas. En tiempo t la luz ha viajado una distancia $ct = \sqrt{\sum x_i^2}$, tal que $c^2t^2 - x_1^2 - x_2^2 - x_3^2 = 0$. La relatividad especial requiere esto en todos los sistemas (inerciales) que se mueven con velocidad $v \leq c$ en cualquier dirección relativa al sistema x_i y que tengan el mismo origen a tiempo $t = 0$, se mantenga también que $c^2t'^2 - x_1'^2 - x_2'^2 - x_3'^2 = 0$. El espacio cuadridimensional con la métrica $x_0^2 - x_1^2 - x_2^2 - x_3^2$ es llamado espacio de Minkowski con el producto escalar de dos cuadrivectores definido como $a \cdot b = a_0b_0 - \vec{a} \cdot \vec{b}$. Usando el tensor métrico

$$(g_{\mu\nu}) = (g^{\mu\nu}) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix} , \tag{4.91}$$

[8]Ser covariante significa que tienen la misma forma en diferentes sistemas de coordenadas tal que no hay un sistema de referencia privilegiado.

podemos subir y bajar índices de un cuadrivector tal como de las coordenadas $x^\mu = (x_0, \vec{x})$ es decir $x_\mu = g_{\mu\nu}x^\nu = (x_0, -\vec{x})$ y $x^\mu g_{\mu\nu} x^\nu = x_0^2 - \vec{x}^{\,2}$, la convención de suma de Einstein se da por entendida. Para el gradiente $\partial^\mu = (\partial/\partial x_0, -\vec{\nabla}) = \partial/\partial x_\mu$ y $\partial_\mu = (\partial/\partial x_0, \vec{\nabla})$ tal que $\partial^2 = \partial^2/\partial x_0^2 - \nabla^2$ es un escalar de Lorentz, al igual que la métrica $x_0^2 - \vec{x}^{\,2}$.

Para $v \ll c$, en el límite no relativista, las transformaciones de Lorentz deben ser transformaciones de Galileo. Por lo tanto, para derivar la forma de una transformación de Lorentz a lo largo del eje x_1, partimos con una transformación Galileana para una velocidad relativa infinitesimal δv:

$$x_1' = x_1 - \delta v t = x_1 - x_0 \delta\beta \ . \tag{4.92}$$

Como es usual $\beta = \dfrac{v}{c}$. Por simetría también podemos escribir

$$x_0' = x_0 - a x_1 \delta\beta \ , \tag{4.93}$$

donde a es un parámetro a fijar una vez que se imponga que $x_0^2 - x_1^2$ deba ser invariante,

$$x_0'^{\,2} - x_1'^{\,2} = x_0^2 - x_1^2 \ . \tag{4.94}$$

Recordemos que $x = (x_0; x_1, x_2, x_3)$ es el prototipo de cuadrivector en el espacio de Minkowski. Así la ecuación (4.94) es simplemente una afirmación de la invariancia del cuadrado de la magnitud del vector distancia bajo rotaciones en el espacio de Minkowski. Aquí es donde la relatividad especial compromete nuestra transformación. Elevando al cuadrado y restando las ecuaciones (4.92) y (4.93) y descartando términos del orden de $\delta\beta^2$, encontramos $a = -1$. Las ecuaciones (4.92) y (4.93) pueden ser combinadas como una ecuación matricial

$$\begin{pmatrix} x_0' \\ x_1' \end{pmatrix} = (1 - \delta\beta\sigma_1) \begin{pmatrix} x_0 \\ x_1 \end{pmatrix} \ , \tag{4.95}$$

σ_1 es la matriz de Pauli, y el parámetro $\delta\beta$ representa un cambio infinitesimal. Repetimos la transformación N veces para desarrollar una transformación finita con el parámetro velocidad $\rho = N\delta\beta$. entonces

$$\begin{pmatrix} x_0' \\ x_1' \end{pmatrix} = \Big(\mathsf{1} - \frac{\rho\sigma_1}{N}\Big)^N \begin{pmatrix} x_0 \\ x_1 \end{pmatrix} \ . \tag{4.96}$$

En el límite $N \to \infty$

$$\lim_{N\to\infty} \Big(\mathsf{1} - \frac{\rho\sigma_1}{N}\Big)^N = \exp(-\rho\sigma_1) \ . \tag{4.97}$$

Interpretamos la exponencial como una serie de Maclaurin

$$\exp(-\rho\sigma_1) = \mathsf{1} - \rho\sigma_1 + \frac{(\rho\sigma_1)^2}{2!} - \frac{(\rho\sigma_1)^3}{3!} + \cdots \ . \tag{4.98}$$

Notando que $\sigma^2 = \mathsf{1}$,

$$\exp(-\rho\sigma_1) = \mathsf{1}\cosh\rho + \sigma_1 \operatorname{senh}\rho \ . \tag{4.99}$$

Por lo tanto nuestra transformación de Lorentz finita es

$$\begin{pmatrix} x_0' \\ x_1' \end{pmatrix} = \begin{pmatrix} \cosh\rho & -\operatorname{senh}\rho \\ -\operatorname{senh}\rho & \cosh\rho \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \end{pmatrix} \ . \tag{4.100}$$

σ_1 ha generado las representaciones de esta especial transformación de Lorentz.

El $\cosh\rho$ y el $\operatorname{senh}\rho$ pueden ser identificados considerando el origen del sistema de coordenadas primas, $x_1' = 0$, o $x_1 = vt$. Sustituyendo en la ecuación (4.100), tenemos

$$0 = x_1 \cosh\rho - x_0 \operatorname{senh}\rho \ . \tag{4.101}$$

Con $x_1 = vt$ y $x_0 = ct$.

$$\tanh\rho = \beta = \frac{v}{c} \ .$$

Note que la rapidez $\rho \neq \dfrac{v}{c}$, excepto en el límite $v \to 0$.

Usando $1 - \tanh^2\rho = (\cosh^2\rho)^{-1}$,

$$\cosh\rho = (1-\beta^2)^{-1/2} \equiv \gamma \ , \qquad \operatorname{senh}\rho = \beta\gamma \ . \tag{4.102}$$

El anterior caso especial en que la velocidad es paralela a uno de los ejes espaciales es simple, pero ilustra la velocidad infinitesimal, la técnica de la exponenciación y el generador. Ahora esta técnica puede ser aplicada para derivar las transformaciones de Lorentz para una velocidad relativa $\vec{v}$ no paralela a ningún eje. Las matrices dadas por la ecuación (4.100) para el caso $\vec{v} = \hat{x}v_x$ forman un subgrupo. Las matrices en el caso general no lo hacen. El producto de dos matrices de transformaciones de Lorentz, $\mathsf{L}(\vec{v}_1)$ y $\mathsf{L}(\vec{v}_2)$, producen una tercera matriz de transformación $\mathsf{L}(\vec{v}_3)$, si las dos velocidades $\vec{v}_1$ y $\vec{v}_2$ son paralelas. La velocidad resultante $\vec{v}_3$ está relacionada con $\vec{v}_1$ y con $\vec{v}_2$ mediante la regla de adición de velocidades de Einstein. Si $\vec{v}_1$ y $\vec{v}_2$ no son paralelas, no existe entonces una relación simple.

4.5. Covarianza de las Ecuaciones de Maxwell.

Si una ley física se mantiene para todas las orientaciones de nuestro (real) espacial sistema de coordenadas (*i.e.* es invariante ante rotaciones), los términos de la ecuación deben ser covariantes bajo rotaciones. Esto significa que escribimos las leyes físicas en la forma matemática escalar=escalar, vector=vector, tensor de segundo rango=tensor de segundo rango, y así sucesivamente. Similarmente, si una ley física se mantiene para todos los sistemas inerciales, los términos de la ecuación deben ser covariantes bajo transformaciones de Lorentz.

Usando el espacio de Minkowski ($x = x_1$, $y = x_2$, $z = x_3$, $ct = x_0$) tenemos un espacio cuadridimensional cartesiano con métrica $g_{\mu\nu}$. Las transformaciones de Lorentz son lineales en el espacio y en el tiempo en este espacio real de cuatro dimensiones.

Consideremos las ecuaciones de Maxwell

$$\vec{\nabla} \times \vec{E} = -\frac{\partial \vec{B}}{\partial t} \ , \tag{4.103a}$$

$$\vec{\nabla} \times \vec{H} = \frac{\partial \vec{D}}{\partial t} + \rho\vec{v} \ , \tag{4.103b}$$

$$\vec{\nabla} \cdot \vec{D} = \rho \ , \tag{4.103c}$$

$$\vec{\nabla} \cdot \vec{B} = 0 \ , \tag{4.103d}$$

y las relaciones

$$\vec{D} = \varepsilon_0 \vec{E} \ , \qquad \vec{B} = \mu_0 \vec{H} \ . \tag{4.104}$$

Todos los símbolos tienen sus significados usuales y hemos supuesto el vacío por simplicidad.

Supongamos que las ecuaciones de Maxwell se mantienen en todos los sistemas inerciales; esto es , las ecuaciones de Maxwell son consistentes con la relatividad especial. (La covariancia de las ecuaciones de Maxwell bajo transformaciones de Lorentz fue realmente mostrada por Lorentz y Poincaré antes de que Einstein propusiera su teoría de la relatividad especial). Nuestro objetivo inmediato es reescribir las ecuaciones de Maxwell como ecuaciones tensoriales en el espacio de Minkowski. Esto hará la covariancia de Lorentz explícita.

En términos de los potenciales escalar y vectorial, podemos escribir

$$\begin{aligned} \vec{B} &= \vec{\nabla} \times \vec{A} \ , \\ \vec{E} &= -\frac{\partial \vec{A}}{\partial t} - \vec{\nabla}\varphi \ . \end{aligned} \tag{4.105}$$

La ecuación anterior especifica el rotor de $\vec{A}$; la divergencia de $\vec{A}$ no está definida. Podemos, y por futuras conveniencias lo hacemos, imponer la siguiente relación sobre el vector potencial

$$\vec{\nabla} \cdot \vec{A} + \varepsilon_0 \mu_0 \frac{\partial \varphi}{\partial t} = 0 \ . \tag{4.106}$$

Este es conocido como el gauge de Lorentz. Servirá a nuestros propósitos de desacoplar las ecuaciones diferenciales para $\vec{A}$ y para φ .

Ahora reescribimos las ecuaciones de Maxwell en términos de los potenciales. A partir de la ecuación (4.103c) para $\vec{\nabla} \cdot \vec{D}$ y (4.105)

$$\nabla^2 \varphi + \vec{\nabla} \cdot \frac{\partial \vec{A}}{\partial t} = -\frac{\rho}{\varepsilon_0} \ , \tag{4.107}$$

considerando que la ecuación (4.103b) para $\vec{\nabla} \times \vec{H}$ y (4.105) y la identidad vectorial para el rotor del rotor produce

$$\frac{\partial^2 \vec{A}}{\partial t^2} + + \vec{\nabla} \frac{\partial \varphi}{\partial t} + \frac{1}{\varepsilon_0 \mu_0} \left[\vec{\nabla} \vec{\nabla} \cdot \vec{A} - \nabla^2 \vec{A} \right] = \frac{\rho \vec{v}}{\varepsilon_0} \ . \tag{4.108}$$

Usando el gauge de Lorentz, la ecuación (4.106), y la relación $\varepsilon_0 \mu_0 = 1/c^2$, obtenemos

$$\begin{aligned} \left[\nabla^2 - \frac{1}{c^2} \frac{\partial^2}{\partial t^2} \right] \vec{A} &= -\mu_0 \rho \vec{v} \ , \\ \left[\nabla^2 - \frac{1}{c^2} \frac{\partial^2}{\partial t^2} \right] \varphi &= -\frac{\rho}{\varepsilon_0} \ . \end{aligned} \tag{4.109}$$

Ahora el operador diferencial

$$\nabla^2 - \frac{1}{c^2} \frac{\partial^2}{\partial t^2} = \partial^2 = -\partial^\mu \partial_\mu \ ,$$

es un Laplaciano cuadridimensional. Usualmente este operador es llamado el d'Alembertiano y denotado por $\Box^2$. Puede probarse que es un escalar.

Por conveniencia definimos

$$\begin{aligned} A^1 &\equiv \frac{A_x}{\mu_0 c} = c\varepsilon_0 A_x \ , \qquad A^3 \equiv \frac{A_z}{\mu_0 c} = c\varepsilon_0 A_z \ , \\ A^2 &\equiv \frac{A_y}{\mu_0 c} = c\varepsilon_0 A_y \ , \qquad A_0 \equiv \varepsilon_0 \varphi = A^0 \ . \end{aligned} \tag{4.110}$$

Si ponemos además

$$\frac{\rho v_x}{c} \equiv i^1 \ , \quad \frac{\rho v_y}{c} \equiv i^2 \ , \quad \frac{\rho v_z}{c} \equiv i^3 \ , \quad \rho \equiv i_0 = i^0 \ , \tag{4.111}$$

entonces la ecuación (4.109) puede ser escrita de la forma

$$\partial^2 A^\mu = i^\mu \ . \tag{4.112}$$

La ecuación anterior parece una ecuación tensorial, pero eso no basta. Para probar que es una ecuación tensorial, partimos investigando las propiedades de transformación de la corriente generalizada i^μ.

Ya que un elemento de carga de es una cantidad invariante, tenemos

$$de = \rho dx_1 dx_2 dx_3 \ , \quad \text{invariante.} \tag{4.113}$$

Vimos que el elemento de volumen cuadridimensional es también un invariante, $dx_1 dx_2 dx_3 dx_0$, comparando estos resultados vemos que la densidad de carga ρ debe transformar de la misma manera que x_0. Ponemos $\rho = i^0$ con i^0 establecida como la componente cero de un cuadrivector. Las otras partes de la ecuación (4.111) pueden ser expandidas como

$$\begin{aligned} i^1 &= \frac{\rho v_x}{c} = \frac{\rho}{c}\frac{dx_1}{dt} \\ &= i^0 \frac{dx_1}{dt} \ . \end{aligned} \tag{4.114}$$

Ya que justo mostramos que i_0 transforma como dx_0, esto significa que i_1 transforma como dx_1. Con resultados similares para i_2 e i_3. tenemos que i^λ transforma como dx^λ, probando de esta manera que i^λ es un vector, un vector del espacio cuadridimensional de Minkowski.

La ecuación (4.112), la cual deriva directamente de las ecuaciones de Maxwell, suponemos que se mantiene en todos los sistemas cartesianos. Entonces, por la regla del cociente A_μ es también un vector y (4.112) es una legítima ecuación tensorial.

Ahora, devolviéndonos, la ecuación (4.105) puede ser escrita

$$\begin{aligned} \varepsilon_0 E_j &= -\frac{\partial A^j}{\partial x_0} + \frac{\partial A^0}{\partial x_j} \ , \quad j = 1,2,3, \\ \frac{1}{\mu c} B_i &= \frac{\partial A^k}{\partial x_j} - \frac{\partial A^j}{\partial x_k} \ , \quad (i,j,k) = (1,2,3) \ , \end{aligned} \tag{4.115}$$

y permutaciones cíclicas.

Definimos un nuevo tensor

$$\partial^\mu A^\nu - \partial^\nu A^\mu = \frac{\partial A^\nu}{\partial x_\mu} - \frac{\partial A^\mu}{\partial x_\nu} \equiv F^{\mu\nu} = -F^{\nu\mu} \quad (\mu, \nu = 0,1,2,3)$$

un tensor antisimétrico de segundo rango, ya que A^μ es un vector. Lo escribimos explícitamente

$$F_{\mu\nu} = \varepsilon_0 \begin{pmatrix} 0 & E_x & E_y & E_z \\ -E_x & 0 & -cB_z & cB_y \\ -E_y & cB_z & 0 & -cB_x \\ -E_z & -cB_y & cB_x & 0 \end{pmatrix} , \quad F^{\mu\nu} = \varepsilon_0 \begin{pmatrix} 0 & -E_x & -E_y & -E_z \\ E_x & 0 & -cB_z & cB_y \\ E_y & cB_z & 0 & -cB_x \\ E_z & -cB_y & cB_x & 0 \end{pmatrix} . \tag{4.116}$$

Notemos que en nuestro espacio de Minkowski $\vec{E}$ y $\vec{B}$ no son más vectores, sino que juntos forman un tensor de segundo rango. Con este tensor podemos escribir las dos ecuaciones de Maxwell no homogéneas (4.103b) y (4.103c) y combinándolas como una ecuación tensorial

$$\frac{\partial F_{\mu\nu}}{\partial x_\nu} = i_\mu \ . \tag{4.117}$$

El lado izquierdo es una divergencia cuadridimensional de un tensor y por lo tanto un vector. Esto es, por supuesto, equivalente a contraer un tensor de tercer rango $\dfrac{\partial F^{\mu\nu}}{\partial x_\lambda}$. Las ecuaciones de Maxwell (4.103a) para $\vec{\nabla} \times \vec{E}$ y la ecuación (4.103d) para $\vec{\nabla} \cdot B$ pueden ser expresadas en forma tensorial

$$\frac{\partial F_{23}}{\partial x_1} + \frac{\partial F_{31}}{\partial x_2} + \frac{\partial F_{12}}{\partial x_3} = 0 \ , \tag{4.118}$$

para (4.103d) y tres ecuaciones de la forma

$$-\frac{\partial F_{30}}{\partial x_2} - \frac{\partial F_{02}}{\partial x_3}\frac{\partial F_{23}}{\partial x_0} = 0 \ , \tag{4.119}$$

para (4.103a). Una segunda ecuación, permutando 120 y una tercera, permutando 130.

Ya que

$$\partial^\lambda F^{\mu\nu} = \frac{\partial F^{\mu\nu}}{\partial x_\lambda} \equiv t^{\lambda\mu\nu} \ ,$$

es un tensor de tercer rango, las ecuaciones (4.117) y (4.119) pueden ser expresadas por la ecuación tensorial

$$t^{\lambda\mu\nu} + t^{\nu\lambda\mu} + t^{\mu\nu\lambda} = 0 \ . \tag{4.120}$$

En todos los casos anteriores los índices μ, ν y λ se suponen diferentes.

4.5.1. Transformaciones de Lorentz de $\vec{E}$ y $\vec{B}$.

La construcción de las ecuaciones tensoriales (4.118) y (4.120) completan nuestro objetivo inicial de reescribir las ecuaciones de Maxwell en forma tensorial. Ahora explotamos las propiedades tensoriales de nuestros cuadrivectores y del tensor $F^{\mu\nu}$.

Para las transformaciones de Lorentz que corresponden a movimientos a lo largo del eje $z(x_3)$ con velocidad v, los "cosenos directores" están dados por

$$\begin{aligned} x'_0 &= \gamma(x_0 - \beta x_3) \\ x'_3 &= \gamma(x_3 - \beta x_1) \ , \end{aligned} \tag{4.121}$$

donde

$$\beta = \frac{v}{c}$$

y

$$\gamma = \left(1-\beta^2\right)^{-1/2} \ . \tag{4.122}$$

Usando las propiedades de transformación tensorial, podemos calcular los campos eléctrico y magnético en el sistema en movimiento en términos de los valores en el marco de referencias original. A partir de las ecuaciones (2.59), (4.116) y (4.121) obtenemos

$$\begin{aligned} E'_x &= \frac{1}{\sqrt{1-\beta^2}}\left(E_x - \frac{v}{c^2}B_y\right) \ , \\ E'_y &= \frac{1}{\sqrt{1-\beta^2}}\left(E_y + \frac{v}{c^2}B_x\right) \ , \\ E'_z &= E_z \ , \end{aligned} \tag{4.123}$$

y

$$\begin{aligned} B'_x &= \frac{1}{\sqrt{1-\beta^2}}\left(B_x + \frac{v}{c^2}E_y\right) \ , \\ B'_y &= \frac{1}{\sqrt{1-\beta^2}}\left(B_y - \frac{v}{c^2}E_x\right) \ , \\ B'_z &= B_z \ . \end{aligned} \tag{4.124}$$

Este acoplamiento de $\vec{E}$ y $\vec{B}$ es esperado. Consideremos, por ejemplo, el caso de campo eléctrico nulo en el sistema sin prima

$$E_x = E_y = E_z = 0 \ .$$

Claramente, no habrá fuerza sobre una partícula de carga estacionaria. Cuando la partícula está en movimiento con una velocidad pequeña $\vec{v}$ a lo largo del eje z un observador sobre la partícula ve campos (ejerciendo una fuerza sobre la partícula cargada) dados por

$$\begin{aligned} E'_x &= -vB_y \ , \\ E'_y &= vB_x \ , \end{aligned}$$

donde $\vec{B}$ es un campo magnético en el sistema sin primas. Estas ecuaciones pueden ser puestas en forma vectorial

$$\vec{E}\,' = \vec{v} \times \vec{B} \ , \qquad \text{o bien,} \qquad \vec{F} = q\vec{v} \times \vec{B} \ , \tag{4.125}$$

la cual es usualmente tomada como la definición operacional del campo magnético $\vec{B}$.

4.5.2. Invariantes electromagnéticas.

Finalmente, las propiedades tensoriales (o vectoriales) nos permiten construir una multitud de cantidades invariantes. Una de las importantes es el producto escalar de los cuadrivectores A_λ y i_λ. Tenemos

$$\begin{aligned} A^\lambda i_\lambda &= -c\varepsilon_0 A_x \frac{\rho v_x}{c} - c\varepsilon_0 A_y \frac{\rho v_y}{c} - c\varepsilon_0 A_z \frac{\rho v_z}{c} + \varepsilon_0 \varphi \rho \\ &= \varepsilon_0(\rho\varphi - \vec{A}\cdot\vec{J}) \ , \quad \text{invariante,} \end{aligned} \tag{4.126}$$

con $\vec{A}$ el usual potencial vector y $\vec{J}$ la densidad de corriente ordinaria. El primer término $\rho\varphi$ es el ordinario acoplamiento electroestático con dimensiones de energía por unidad de volumen. En consecuencia nuestro recién construido invariante escalar es un densidad de energía. La interacción dinámica del campo y corriente es dado por el producto $\vec{A} \cdot \vec{J}$. Este invariante $A^\lambda i_\lambda$ aparece en los Lagrangianos electromagnéticos.

4.6. Grupos discretos.

En esta sección regresamos a grupos con un número finito de elementos. En Física, los grupos usualmente aparecen como un conjunto de operaciones que dejan un sistema sin cambios, *i.e.* invariante. Esta es una expresión de simetría. Realmente una simetría puede ser definida como la invariancia del Hamiltoniano de un sistema bajo un grupo de transformaciones. La simetría en este sentido es importante en Mecánica Clásica, pero llega a ser aún más importante y más profunda en Mecánica Cuántica. En esta sección investigaremos las propiedades de simetría de un conjunto de objetos (átomos en una molécula o cristal). Esto proveerá ilustración adicional de los conceptos de grupos de la sección 4.1 y conduce directamente a los grupos dihédricos. Los grupos dihédricos a su vez nos abren el estudio de los 32 grupos puntuales y 230 grupos espaciales que son de suma importancia en cristalografía y Física del Sólido. Notemos que fue a través del estudio de las simetrías cristalinas que los conceptos de simetría y teoría de grupos entraron a la Física. En Física, la condición de grupo abstracto a menudo asume un significado físico directo en términos de transformaciones de vectores, spinores y tensores.

Como un muy simple, pero no trivial, ejemplo de un grupo finito, consideremos el conjunto 1, a, b, c que combinamos de acuerdo a la tabla de multiplicación del grupo.[9]

	1	a	b	c
1	1	a	b	c
a	a	b	c	1
b	b	c	1	a
c	c	1	a	b

Claramente, las cuatro condiciones de la definición de grupo son satisfechas. Los elementos a, b, c y 1 son entidades matemáticas abstractas, completamente sin restricciones, excepto por la anterior tabla de multiplicación.

Ahora, para una específica *representación* de estos elementos del grupo, tomemos

$$1 \to 1 \ , \quad a \to i \ , \quad b \to -1 \ , \quad c \to -i \ , \tag{4.127}$$

combinados con la multiplicación ordinaria. Nuevamente, las cuatro condiciones de grupo son satisfechas y estos cuatro elementos forman un grupo. Etiquetamos este grupo por C_4. Ya que la multiplicación de los elementos del grupo es conmutativa, el grupo es llamado *conmutativo* o *abeliano*. Nuestro grupo es también un *grupo cíclico*, en que los elementos pueden ser escritos como potencias sucesivas de un elemento, en este caso i^n, $n = 0, 1, 2, 3$. Notemos que al

[9] El orden de los factores es fila-columna: $ab = c$

	I	V_1	V_2	V_2
I	I	V_1	V_2	V_3
V_1	V_1	I	V_3	V_2
V_2	V_2	V_3	I	V_1
V_3	V_3	V_2	V_1	I

Cuadro 4.3: Tabla de producto del *vierergruppe*.

escribir explícitamente la ecuación (4.127) hemos seleccionado una *representación* específica para este grupo de cuatro objetos, C_4.

Reconocemos que los elementos del grupo 1, i, -1, $-i$ pueden ser interpretados como sucesivas rotaciones en 90° en el plano complejo. Entonces a partir de la ecuación (3.70) creamos un conjunto de cuatro matrices de 2×2 (reemplazando φ por $-\varphi$ en la ecuación (3.70) para rotar un vector más que rotar las coordenadas.)

$$\mathsf{R}(\varphi) = \begin{pmatrix} \cos\varphi & -\operatorname{sen}\varphi \\ \operatorname{sen}\varphi & \cos\varphi \end{pmatrix} \ ,$$

y para $\varphi = 0$, $\pi/2$, π y $3\pi/2$ tenemos

$$\begin{aligned} \mathsf{1} &= \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \qquad \mathsf{A} = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} \\ \mathsf{B} &= \begin{pmatrix} -1 & 0 \\ 0 & -1 \end{pmatrix} \qquad \mathsf{C} = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} \ . \end{aligned} \tag{4.128}$$

Este conjunto de cuatro matrices forman un grupo donde la ley de combinación es la multiplicación de matrices. Aquí hay una segunda representación, ahora en términos de matrices. Es fácil verificar que esta representación es también abeliana o cíclica. Claramente hay una correspondencia uno-a-uno entre las dos representaciones.

$$1 \leftrightarrow 1 \leftrightarrow \mathsf{1} \quad a \leftrightarrow i \leftrightarrow \mathsf{A} \quad b \leftrightarrow -1 \leftrightarrow B \quad c \leftrightarrow -i \leftrightarrow C \ . \tag{4.129}$$

En el grupo C_4 las dos representaciones $(1, i, -1, -i)$ y $(\mathsf{1}, \mathsf{A}, \mathsf{B}, \mathsf{C})$ son isomórficas.

En contraste con esto, no hay tal correspondencia entre cualquiera de esta representaciones del grupo C_4 y otro grupo de cuatro objetos, el *vierergruppe*. El *vierergruppe* tiene una tabla de multiplicación distinta (tabla 4.3). Confirmando la falta de correspondencia entre el grupo representado por $(1, i, -1, -i)$ o por las matrices $(\mathsf{1}, \mathsf{A}, \mathsf{B}, \mathsf{C})$ con el *vierergruppe*, notemos que aunque el *vierergruppe* es abeliano, no es cíclico. El grupo cíclico C_4 y el *vierergruppe* no son isomórficos.

4.6.1. Clase y caracteres.

Considere un elemento del grupo x transformado en un elemento del grupo y mediante una transformación de similaridad con respecto a g_i, un elemento del grupo

$$g_i x g_i^{-1} = y \ . \tag{4.130}$$

El elemento del grupo y es el conjugado de x. Una *clase* es un conjunto de elementos del grupo mutuamente conjugados. En general, este conjunto de elementos que forma una clase no satisface los postulados de un grupo y no es un grupo. De hecho, el elemento unidad I el cual está siempre en una clase con sí mismo, es la única clase que es también un subgrupo. Todos los miembros de una clase dada son equivalentes en el sentido en que cualquier elemento es una transformación de similaridad de otro miembro de la clase. Claramente si un grupo es abeliano, cada elemento del grupo es una clase por sí mismo. Encontramos que

1. Cada elemento del grupo original pertenece a una y sólo a una clase.

2. El número de elementos de una clase es factor del orden del grupo.

Obtenemos una posible interpretación física del concepto de clases si notamos que y es una transformada de similaridad de x. Si g_i representa una rotación del sistema de coordenadas, y es la misma operación que x pero relativa al nuevo sistema de coordenadas.

En la sección 3.3 vimos que una matriz real transforma bajo rotaciones de coordenadas como una transformación de similaridad ortogonal. Dependiendo de la elección del sistema de referencia, esencialmente la misma matriz puede tomar una infinidad de diferentes formas. Así mismo, nuestras representaciones de grupo puede ser puesta por una infinidad de formas diferentes utilizando una transformación unitaria. Pero, cada representación es isomórfica a la original. Se puede probar que la traza de cada elemento (cada matriz de nuestra representación) es invariante ante transformaciones unitarias. Es por esto, que la traza (reetiquetado *caracter*) asume un rol de alguna importancia en la teoría de grupo, en particular en la aplicación a la Física del estado sólido. Claramente, todos los miembros de una clase dada (en una representación dada) tienen el mismo caracter. Elementos de clases diferentes pueden tener el mismo caracter pero elementos con distintos caracteres no pueden estar en la misma clase.

El concepto de clase es importante (1) por la traza o caracter y (2) porque *el número de representaciones no equivalentes irreducibles de un grupo de igual número de clases.*

4.6.2. Subgrupos y cosetos.

Frecuentemente un subconjunto de elementos del grupo (incluyendo el elemento unidad I) va a satisfacer los cuatro requerimientos para ser grupo, por lo tanto es un grupo. Dicho conjunto es llamado un *subgrupo*. Cada grupo tiene dos subgrupos triviales: el elemento unidad solo y el grupo en si mismo. Los elementos 1 y b del grupo de cuatro elementos C_4 discutido anteriormente forman un subgrupo no trivial. En la sección 4.1 consideramos $SO(3)$, el grupo (continuo) de todas las rotaciones en el espacio ordinario. Las rotaciones con respecto a un solo eje forman un subgrupo de $SO(3)$. Muchos otros ejemplos de subgrupos aparecen en la siguientes secciones.

Considere un subgrupo H con elementos h_i y un elemento del grupo x que no está en H. Entonces $h_i x$ y $x h_i$ no están en el subgrupo H. Los conjuntos generados por

$$x h_i \ , \ i = 1, 2, 3 \ldots \quad \text{y} \quad h_i x \ , i = 1, 2, 3, \ldots$$

son llamados *cosetos*, el coseto izquierdo y derecho del subgrupo H con respecto a x, respectivamente. Puede ser demostrado (suponiendo lo contrario y probando una contradicción) que

el coseto de un subgrupo tiene el mismo número de elementos que el subgrupo. Extendiendo este resultado podemos expresar el grupo original G como la suma de H y cosetos:

$$G = H + x_1H + x_2H + \cdots \ .$$

Entonces el orden de un subgrupo es un divisor del orden del grupo. Es este resultado es el que hace significativo el concepto de coseto. En la próxima sección el grupo de seis elementos D_3 (de orden 6) tiene subgrupos de orden 1, 2 y 3. D_3 no puede (y no tiene) subgrupos de orden 4 o 5.

La transformación de similaridad de un subgrupo H por un elemento fijo x que no pertenece a H, xHx^{-1} produce un subgrupo. Si este nuevo grupo es idéntico a H para todo x,

$$xHx^{-1} = H \ ,$$

entonces H es llamado un *subgrupo invariante, normal o auto conjugado.* Tales subgrupos están involucrados en el análisis de múltiples atómicos, espectros nucleares y en las partículas discutidas en la sección 4.2. Todos los subgrupos de un grupo conmutativo (abeliano) son automáticamente invariantes.

4.6.3. Dos objetos–Doble eje de simetría.

Consideremos primero el sistema bidimensional de dos átomos idénticos en el plano xy en (1,0) y (-1,0), figura (4.6). ¿Qué rotaciones[10] pueden ser llevadas a cabo (manteniendo ambos átomos en el plano xy) tal de dejar al sistema invariante? El primer candidato es, por supuesto, el operador unitario 1. Una rotación en π radianes respecto el eje z completa la lista. De esta manera tenemos un muy poco interesante grupo de 2 elementos (1, -1). El eje z es llamado un eje doble de simetría que corresponde a los dos ángulos de rotación, 0 y π que dejan el sistema invariante.

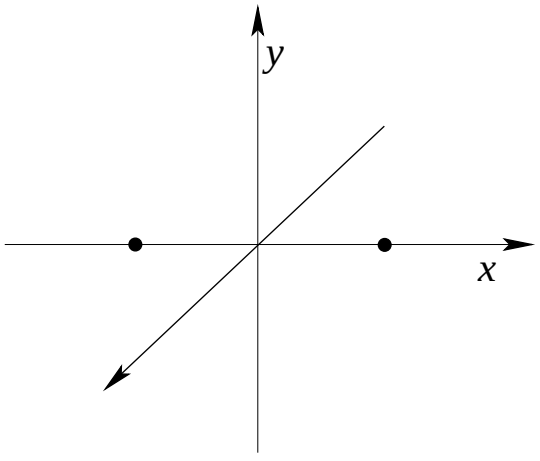


Figura 4.6: Molécula Diatómica.

[10] Aquí excluimos deliberadamente reflexiones e inversiones. Ellas deben ser introducidas para desarrollar el conjunto completo de 32 grupos puntuales.

Nuestro sistema llega a ser más interesante en tres dimensiones. Ahora imaginemos una molécula (o parte de un cristal) con átomos de elemento X en $\pm a$ en el eje x, átomos de elemento Y en $\pm b$ en el eje y, y átomos de elementos Z en $\pm c$ en el eje z como muestra la figura (4.7). Claramente, cada eje es ahora un doble eje de simetría. Usando $\mathsf{R}_x(\pi)$ para designar una rotación en π radianes respecto al eje x, podemos establecer una representación matricial de las rotaciones como en la sección 3.3:

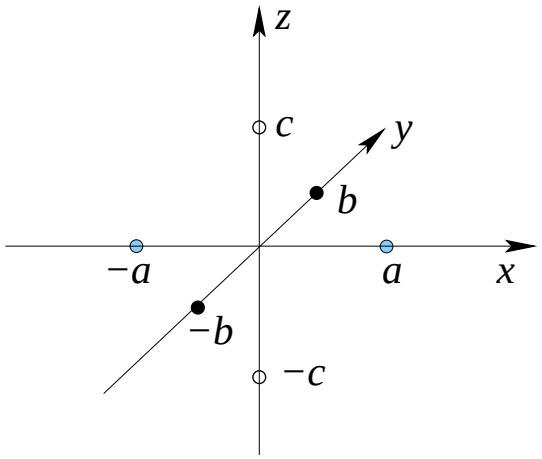


Figura 4.7: Simetría D_2.

$$\mathsf{R}_x(\pi) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{pmatrix} \qquad \mathsf{R}_y(\pi) = \begin{pmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{pmatrix}$$
$$\mathsf{R}_z(\pi) = \begin{pmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \qquad \mathsf{1} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{4.131}$$

Estos cuatro elementos $[\mathsf{1}, \mathsf{R}_x(\pi), \mathsf{R}_y(\pi), \mathsf{R}_z(\pi)]$ forman un grupo abeliano con la tabla de multiplicación mostrada en la tabla 4.4.

Los productos mostrados en la tabla 4.4 se pueden obtener de dos diferentes maneras: (1) Podemos analizar las operaciones sobre ellos mismos, una rotación en π respecto del eje x seguida de una rotación en π respecto al eje y es equivalente a una rotación en π respecto al eje z: $\mathsf{R}_y(\pi)\mathsf{R}_x(\pi) = \mathsf{R}_z(\pi)$. (2) Alternativamente, una vez que la representación matricial es establecida, podemos obtener los productos por multiplicación matricial. Esto es donde se muestra la potencia de las matemáticas cuando el sistema es demasiado complejo para una interpretación física directa.

Fácilmente podemos darnos cuenta que este grupo es el *vierergruppe*. También, ellas son obviamente reducibles, siendo diagonal. Los subgrupos son $(\mathsf{1}, \mathsf{R}_x)$, $(\mathsf{1}, \mathsf{R}_y)$ y $(\mathsf{1}, \mathsf{R}_z)$. Ellos son invariantes. Debemos notar que una rotación en π respecto al eje y y una rotación en π respecto al eje z es equivalente a un rotación en π respecto al eje x: $\mathsf{R}_z(\pi)\mathsf{R}_y(\pi) = \mathsf{R}_x(\pi)$. En

	1	$\mathsf{R}_x(\pi)$	$\mathsf{R}_y(\pi)$	$\mathsf{R}_z(\pi)$
1	1	$\mathsf{R}_x(\pi)$	$\mathsf{R}_y(\pi)$	$\mathsf{R}_z(\pi)$
$\mathsf{R}_x(\pi)$	$\mathsf{R}_x(\pi)$	1	$\mathsf{R}_z(\pi)$	$\mathsf{R}_y(\pi)$
$\mathsf{R}_y(\pi)$	$\mathsf{R}_y(\pi)$	$\mathsf{R}_z(\pi)$	1	$\mathsf{R}_x(\pi)$
$\mathsf{R}_z(\pi)$	$\mathsf{R}_z(\pi)$	$\mathsf{R}_y(\pi)$	$\mathsf{R}_x(\pi)$	1

Cuadro 4.4: Tabla de producto.

términos de simetría, si y y z son dos ejes doble de simetría, x es automáticamente un eje doble de simetría.

Este grupo de simetría,[11] el *vierergruppe*, es a menudo llamado D_2, la D significa un grupo dihédrico y el subíndice 2 significa un eje de simetría doble (y sin ejes de simetrías más altos).

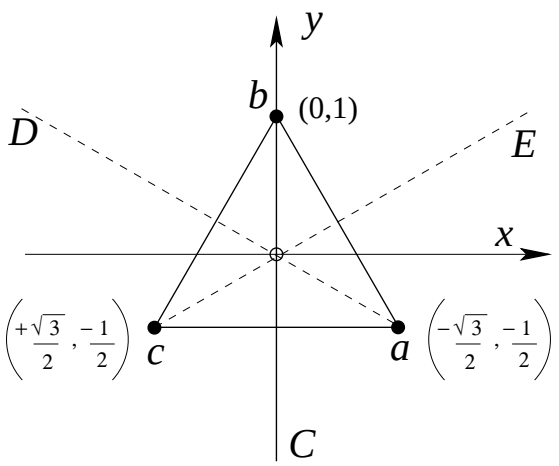


Figura 4.8: Operaciones de simetría en un triángulo equilátero.

4.6.4. Tres objetos–Triple eje de simetría.

Consideremos ahora tres átomos idénticos en los vértices de un triángulo equilátero, figura(4.8). Rotaciones del *triángulo* de 0, $2\pi/3$, y $4\pi/3$ dejando el triángulo invariante. En

[11] Un grupo de simetría es un grupo de operaciones que simetría-preservantes, tales como, rotaciones, reflexiones e inversiones. Un grupo simétrico es el grupo de las permutaciones de n objetos distintos de orden $n!$.

forma matricial, tenemos[12]

$$\begin{aligned}
\mathsf{1} &= \mathsf{R}_z(0) = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \\
\mathsf{A} &= \mathsf{R}_z(2\pi/3) = \begin{pmatrix} \cos 2\pi/3 & -\operatorname{sen} 2\pi/3 \\ \operatorname{sen} 2\pi/3 & \cos 2\pi/3 \end{pmatrix} = \begin{pmatrix} -1/2 & -\sqrt{3}/2 \\ \sqrt{3}/2 & -1/2 \end{pmatrix} \\
\mathsf{B} &= \mathsf{R}_z(4\pi/3) = \begin{pmatrix} -1/2 & \sqrt{3}/2 \\ -\sqrt{3}/2 & -1/2 \end{pmatrix} .
\end{aligned} \tag{4.132}$$

El eje z es un eje de simetría triple. $(\mathsf{1}, \mathsf{A}, \mathsf{B})$ forman un grupo cíclico, un subgrupo del grupo completo de seis elementos que forman.

En el plano xy hay tres ejes de simetría adicionales, cada átomo (vértice) y el centro geométrico definen un eje. Cada uno de éstos son un eje de simetría doble. Éstas rotaciones pueden ser descritas más fácilmente dentro de nuestro esquema bidimensional introduciendo reflexiones. La rotación en π respecto el eje C o al eje y, lo que significa el intercambio de los átomos a y c, es sólo una reflexión de el eje x:

$$\mathsf{C} = \mathsf{R}_C(\pi) = \begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix} . \tag{4.133}$$

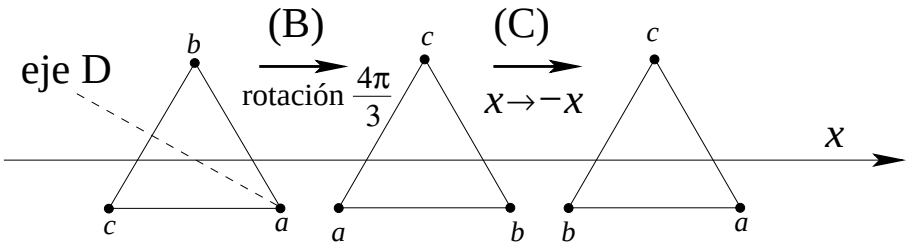


Figura 4.9: El triángulo a l derecha es el triángulo a la izquierda rotado en 180° respecto al eje D. $\mathsf{D} = \mathsf{CB}$.

Podemos reemplazar la rotación respecto al eje D por una rotación en $4\pi/3$ (respecto a nuestro eje z) seguido por una reflexión del eje x ($x \to -x$) (figura 4.9):

$$\begin{aligned}
\mathsf{D} &= \mathsf{R}_D(\pi) = \mathsf{CB} \\
&= \begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} -1/2 & \sqrt{3}/2 \\ -\sqrt{3}/2 & -1/2 \end{pmatrix} \\
&= \begin{pmatrix} 1/2 & -\sqrt{3}/2 \\ -\sqrt{3}/2 & -1/2 \end{pmatrix} .
\end{aligned} \tag{4.134}$$

De manera similar; la rotación en π respecto al eje E que intercambia a con b es reemplazada

[12]Note que estamos rotando el triángulo en sentido antihorario relativo a las coordenadas fijadas.

por una rotación en $2\pi/3$ (A) y luego una reflexión[13] del eje x ($x \to -x$):

$$\begin{aligned}\mathsf{E} &= \mathsf{R}_E(\pi) = \mathsf{CA} \\ &= \begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} -1/2 & -\sqrt{3}/2 \\ \sqrt{3}/2 & -1/2 \end{pmatrix} \\ &= \begin{pmatrix} 1/2 & \sqrt{3}/2 \\ \sqrt{3}/2 & -1/2 \end{pmatrix} .\end{aligned} \tag{4.135}$$

La tabla de multiplicación completa del grupo es

	1	A	B	C	D	E
1	1	A	B	C	D	E
A	A	B	1	D	E	C
B	B	1	A	E	C	D
C	C	E	D	1	B	A
D	D	C	E	A	1	B
E	E	D	C	B	A	1

Notemos que cada elemento del grupo aparece sólo una vez en cada fila y en cada columna.

También, a partir de la tabla de multiplicación el grupo no es abeliano. Hemos construido un grupo de seis elementos y una representación en matriz irreducible de 2×2. El único otro grupo distinto de seis elementos es el grupo cíclico $[1, \mathsf{R}, \mathsf{R}^2, \mathsf{R}^3, \mathsf{R}^4, \mathsf{R}^5]$ con

$$\mathsf{R} = \begin{pmatrix} \cos\pi/3 & -\operatorname{sen}\pi/3 \\ \operatorname{sen}\pi/3 & \cos\pi/3 \end{pmatrix} = \begin{pmatrix} 1/2 & -\sqrt{3}/2 \\ \sqrt{3}/2 & 1/2 \end{pmatrix} . \tag{4.136}$$

Nuestro grupo $[1, \mathsf{A}, \mathsf{B}, \mathsf{C}, \mathsf{D}, \mathsf{E}]$ es llamado D_3 en cristalografía, el grupo diedro con un eje de simetría triple. Los tres ejes (C, D, y E) en el plano xy llegan a ser automáticamente ejes de simetría dobles. Como una consecuencia, $(1, \mathsf{C})$, $(1, \mathsf{D})$ y $(1, \mathsf{E})$ todos forman subgrupos de dos elementos. Ninguno de estos subgrupos de dos elementos de D_3 es invariante.

Hay otras dos representaciones irreducibles del grupo de simetría del triángulo equilátero: (1) la trivial (1, 1, 1, 1, 1, 1), y (2) el casi trivial (1, 1, 1, -1, -1, -1), los signos positivos corresponden a rotaciones apropiadas y los signos negativos a rotaciones impropias (aquellas que involucran una reflexión). Ambos representaciones son homomórficas con D_3.

Un resultado general e importante para grupos finitos de h elementos es que

$$\sum_i n_i^2 = h \ , \tag{4.137}$$

donde n_i es la dimensión de las matrices de la i-ésima representación irreducible. Ésta igualdad, a veces llamada teorema de dimensionalidad, es muy útil para establecer las representaciones irreducible de un grupo. Aquí para D_3 tenemos $1^2 + 1^2 + 2^2 = 6$ para nuestras tres representaciones. No existe otra representación del grupo de simetría de tres objetos.

[13]Note que, como una consecuencia de esas reflexiones, $\det(\mathsf{C}) = \det(\mathsf{D}) = \det(\mathsf{E}) = -1$. Las rotaciones A y B tienen un determinante igual a $+1$.

4.6.5. Grupos dihédricos, D_n

Un grupo dihédrico D_n con un eje de simetría de orden n implica n ejes con separación angular $2\pi/n$ radianes. n es un entero positivo, pero sin más restricciones. Si nosotros aplicamos los argumentos de simetría a *redes cristalinas*, entonces n está limitado a 1, 2, 3, 4 y 6. Los requerimientos de invariancia del la red cristalina bajo translaciones en el plano perpendicular al eje de orden n excluye $n = 5$, 7, y valores más altos. Trate de cubrir un plano completamente con pentágonos regulares idénticos y sin sobreponerse. Para moléculas individuales, esta restricción no existe, aunque los ejemplos con $n > 6$ son raros. Sin embargo, $n = 5$ es una posibilidad real. Como un ejemplo, el grupo de simetría del rutenoceno, $(C_5H_5)_2Ru$, ilustrado en la figura 4.10:, es D_5.

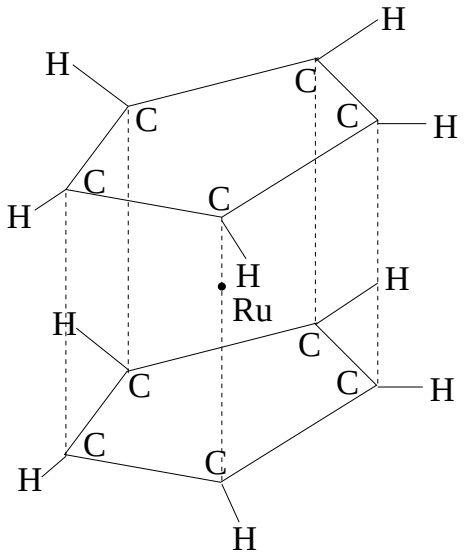


Figura 4.10: Rutenoceno, $(C_5H_5)_2Ru$.

4.6.6. Grupos espaciales y cristalográficos puntuales.

Los grupos dihédricos recién considerados son ejemplos de grupos cristalográficos puntuales. Un grupo puntual está compuesto de combinaciones de rotaciones y reflexiones (incluyendo inversiones) que dejarán alguna red cristalina sin cambios. Limitando las operaciones a rotaciones y reflexiones (incluyendo inversiones) significa que un punto, el origen, permanece fijo, de aquí el término *grupo puntual*. Incluyendo los grupos cíclicos, los dos grupos cúbicos (simetrías tetrahédricas y octahédricas) y las formas impropias (que incluyen reflexiones), tenemos un total de 32 grupos puntuales.

Si, a las operaciones de rotación y reflexión que producen los grupos puntuales, agregamos la posibilidad de translación y todavía demandamos que alguna red cristalina se mantenga invariante, llegamos a los grupos espaciales. Hay 230 distintos grupos espaciales, un número que espanta, excepto, posiblemente, a los especialistas en el campo. Para detalles hay que ver literatura especializada.

Parte II

Ecuaciones diferenciales ordinarias.

Capítulo 5

Ecuaciones diferenciales de primer orden.

versión final 2.2-021029[1]

5.1. Introducción.

Al estudiar los fenómenos físicos, con frecuencia no es posible hallar de inmediato las expresiones explícitas de las magnitudes que caracterizan dichos fenómenos. Sin embargo, suele ser más fácil establecer la dependencia entre esas magnitudes y sus derivadas o sus diferenciales. Así, obtenemos ecuaciones que contienen las funciones desconocidas, escalares o vectoriales, bajo el signo de derivadas o de diferenciales.

Las ecuaciones en las cuales la función desconocida, escalar o vectorial, se encuentra bajo el signo de derivada o de diferencial, se llaman *ecuaciones diferenciales*. Veamos algunos ejemplos de ecuaciones diferenciales:

1) $\dfrac{dx}{dt} = -kx$ es la ecuación de la desintegración radiactiva. (k es la constante de desintegración; x es la cantidad de sustancia no desintegrada en el tiempo t. La velocidad de desintegración $\dfrac{dx}{dt}$ es proporcional a la cantidad de sustancia que se desintegra).

2) $m\dfrac{d^2\vec{r}}{dt^2} = \vec{F}\left(t, \vec{r}, \dfrac{d\vec{r}}{dt}\right)$ es la ecuación del movimiento de un punto de masa m, bajo la influencia de una fuerza $\vec{F}$ dependiente del tiempo, de la posición, $\vec{r}$, y de su velocidad $\dfrac{d\vec{r}}{dt}$. La fuerza es igual al producto de la masa por la aceleración.

3) $\dfrac{\partial^2 u}{\partial x^2} + \dfrac{\partial^2 u}{\partial y^2} + \dfrac{\partial^2 u}{\partial z^2} = -4\pi\rho(x, y, z)$ es la ecuación de Poisson, la cual satisface el potencial electroestático $u(x, y, z)$ en presencia de la densidad de carga $\rho(x, y, z)$.

Si se indican los métodos para hallar las funciones incógnitas, determinadas por las ecuaciones diferenciales, se habrá hallado así la dependencia entre las magnitudes indicadas. La

[1]Este capítulo está basado en el primer capítulo del libro: *Ecuaciones diferenciales y cálculo variacional* de L. Elsgoltz, editorial MIR

búsqueda de las funciones desconocidas, determinadas por las ecuaciones diferenciales, es precisamente el problema fundamental de la teoría de ecuaciones diferenciales.

Si en una ecuación diferencial la función desconocida, escalar o vectorial, en función de una sola variable, la ecuación diferencial es llamada *ordinaria* (los dos primeros casos en el ejemplo anterior). Si, en cambio, la función desconocida es función de dos o más variables independientes, la ecuación diferencial se llama *ecuación en derivadas parciales* (el tercer caso en el ejemplo anterior).

Se denomina *orden* de la ecuación diferencial al grado de la derivada (o diferencial) máxima de la función desconocida, que figura en la ecuación.

Se llama *solución* de la ecuación diferencial a una función que, al ser sustituida en la ecuación diferencial, la convierte en una identidad.

Por ejemplo, la ecuación de la desintegración radiactiva

$$\frac{dx}{dt} = -kx \ , \tag{5.1}$$

tiene la solución

$$x(t) = ce^{-kt} \ , \tag{5.2}$$

donde c es una constante arbitraria.

Es evidente que la ecuación diferencial (5.1) aún no determina por completo la ley de desintegración $x = x(t)$. Para su completa determinación hay que conocer la cantidad de sustancia x_0 en un momento inicial t_0. Si x_0 es conocida, entonces tomando en cuenta la condición $x(t_0) = x_0$, de (5.2) hallamos la ley de desintegración radiactiva:

$$x(t) = x_0 e^{-k(t-t_0)} \ .$$

El proceso de determinar las soluciones de una ecuación diferencial se llama *integración* de la misma. En el ejemplo anterior hallamos fácilmente la solución exacta, pero, en casos más complejos, con frecuencia es necesario utilizar métodos aproximados de integración de dichas ecuaciones. Este tema lo abordaremos en la parte IV de estos apuntes.

Veamos más detalladamente el problema más complejo mencionado antes, de la determinación de la trayectoria $\vec{r} = \vec{r}(t)$ de un punto material de masa m, bajo la acción de una fuerza dada $\vec{F}(t, \vec{r}, \dot{\vec{r}})$. Según la ecuación de Newton,

$$m\ddot{\vec{r}} = \vec{F}(t, \vec{r}, \dot{\vec{r}}) \ . \tag{5.3}$$

Por lo tanto, el problema se reduce a la integración de esta ecuación diferencial. Es evidente que la ley de movimiento aún no queda determinada por completo si se dan la masa m y la fuerza $\vec{F}$; hay que conocer también la posición inicial del punto

$$\vec{r}(t_0) = \vec{r}_0 \tag{5.4}$$

y su velocidad inicial

$$\dot{\vec{r}}(t_0) = \dot{\vec{r}}_0 \tag{5.5}$$

Obsérvese que la ecuación vectorial (5.3) de segundo orden puede sustituirse por un sistema equivalente de dos ecuaciones vectoriales de primer orden, si consideramos la velocidad $\vec{v}$ como una segunda función vectorial desconocida:

$$\frac{d\vec{r}}{dt} = \vec{v} \ , \qquad m\frac{d\vec{v}}{dt} = \vec{F}(t, \vec{r}, \vec{v}) \ . \tag{5.6}$$

Cada ecuación vectorial en el espacio tridimensional puede ser sustituida, proyectando sobre los ejes de coordenadas, por tres ecuaciones escalares. Por lo tanto, la ecuación (5.3) es equivalente a un sistema de tres ecuaciones escalares de segundo orden, y el sistema (5.6), a un sistema de seis ecuaciones escalares de primer orden.

Finalmente, podemos sustituir una ecuación vectorial (5.3) de segundo orden en el espacio tridimensional por una ecuación vectorial de primer orden en el espacio de seis dimensiones cuyas coordenadas son las tres componentes del vector posición r_x, r_y y r_z, más las tres componentes del vector velocidad v_x, v_y y v_z. Usualmente este espacio es llamado *espacio de fase*. El vector $\vec{R}(t)$ en dicho espacio tiene coordenadas $(r_x, r_y, r_z, v_x, v_y, v_z)$. En esta notación el sistema (5.6) toma la forma:

$$\frac{d\vec{R}}{dt} = \Phi(t, \vec{R}(t)) \ , \tag{5.7}$$

las proyecciones del vector Φ en el espacio de seis dimensiones son las correspondientes proyecciones de los segundos miembros del sistema (5.6) en el espacio tridimensional.

Bajo esta interpretación, las condiciones iniciales (5.4) y (5.5) se sustituyen por la condición

$$\vec{R}(t_0) = \vec{R}_0 \ .$$

La solución de la ecuación (5.7) $\vec{R} = \vec{R}(t)$ será una trayectoria en el espacio de fase, en la que cada uno de sus puntos corresponde a cierto estado instantáneo del sistema.

5.2. Ecuaciones de primer orden.

Una ecuación diferencial ordinaria de primer orden, puede escribirse en la forma

$$\frac{dy}{dx} = f(x, y) \ .$$

Un ejemplo simple de tal ecuación,

$$\frac{dy}{dx} = f(x) \ ,$$

se analiza en el curso de cálculo integral. En este caso simple, la solución

$$y = \int f(x)\, dx + c \ ,$$

contiene una constante arbitraria , que puede determinarse si se conoce el valor de $y(x_0) = y_0$; entonces

$$y = y_0 + \int_{x_0}^{x} f(x')\, dx' \ .$$

Más adelante se mostrará que, bajo algunas limitaciones establecidas sobre la función $f(x, y)$, la ecuación

$$\frac{dy}{dx} = f(x, y) \ ,$$

tiene también una solución única, que satisface la condición $y(x_0) = y_0$, y su *solución general*, es decir, el conjunto de soluciones que contienen, sin excepción, todas las soluciones, depende de una constante arbitraria.

La ecuación diferencial $\dfrac{dy}{dx} = f(x, y)$ establece una dependencia entre las coordenadas de un punto y el coeficiente angular de la tangente $\dfrac{dy}{dx}$ a la gráfica de la solución en ese punto. Conociendo x e y, se puede calcular $\dfrac{dy}{dx}$. Por consiguiente, la ecuación diferencial de la forma considerada determina un campo de direcciones (figura 5.1), y el problema de la integración de la ecuación diferencial se reduce a hallar las llamadas *curvas integrales*, para las cuales la dirección de las tangentes a éstas coincide en cada punto con la dirección del campo.

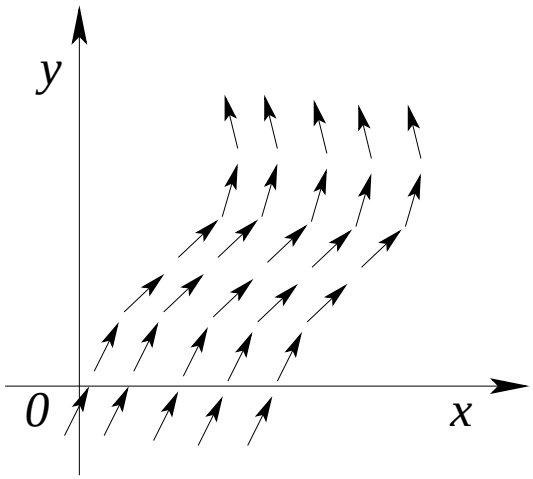


Figura 5.1: Curvas integrales.

Ejemplo Consideremos la siguiente ecuación

$$\frac{dy}{dx} = \frac{y}{x} \ .$$

En cada punto, diferente del punto (0,0), el coeficiente angular de la tangente a la curva integral buscada es igual a la razón $\frac{y}{x}$, o sea, coincide con el coeficiente angular de la recta dirigida desde el origen de coordenadas al mismo punto (x, y). En la figura (5.2) está representado con flechas el campo de direcciones determinado por la ecuación estudiada. Evidentemente, en este caso las curvas integrales serán las rectas $y = cx$, ya que las direcciones de estas rectas coinciden en todas partes con la dirección del campo.

En muchos problemas, en particular en casi todos los problemas de carácter geométricos, las variables x e y son equivalentes. Por ello, en dichos problemas, si éstos se reducen a la resolución de la ecuación diferencial

$$\frac{dy}{dx} = f(x, y) \ , \tag{5.8}$$

es natural considerar, conjuntamente con (5.8), también

$$\frac{dx}{dy} = \frac{1}{f(x, y)} \ . \tag{5.9}$$

Si ambas ecuaciones tienen sentido, entonces son equivalentes, ya que si la función $y = y(x)$ es solución de la ecuación (5.8), la función inversa $x = x(y)$ es solución de (5.9) y, por lo tanto, ambas ecuaciones poseen curvas integrales comunes.

Si, en cambio, en algunos puntos una de las ecuaciones (5.8) o (5.9) pierde sentido, entonces en esos puntos es natural sustituirla por la otra ecuación.

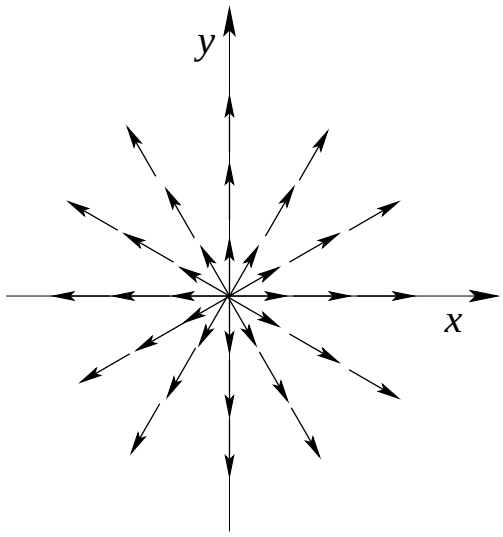


Figura 5.2: Curvas integrales $y = cx$.

5.3. Ecuaciones con variables separables.

Las ecuaciones diferenciales del tipo

$$f_2(y)\,dy = f_1(x)\,dx\ , \tag{5.10}$$

se llaman *ecuaciones con variables separadas*. Consideremos que las funciones f_1 y f_2 son continuas.

Supongamos que $y(x)$ es solución de esta ecuación; entonces al sustituir $y(x)$ en la ecuación (5.10), obtenemos una identidad que, al ser integrada, da

$$\int f_2(y)\,dy = \int f_1(x)\,dx + c\ , \tag{5.11}$$

donde c es una constante arbitraria.

Hemos obtenido una ecuación en que no figuran ni derivadas ni diferenciales, ecuaciones así se denominan *ecuaciones finitas*, satisfechas por todas las soluciones de la ecuación (5.10). Además, cada solución de la ecuación (5.11) es solución de (5.10), ya que si una función $y(x)$, al ser sustituida en la ecuación (5.11), la transforma en una identidad, entonces derivando dicha identidad obtenemos que $y(x)$ satisface (5.10).

La ecuación finita $\Phi(x, y) = 0$ que determina la solución $y(x)$ de la ecuación diferencial como función implícita de x, se llama *integral* de la ecuación diferencial considerada.

Si esta ecuación finita determina sin excepción todas las soluciones de la ecuación diferencial dada, entonces se llama *integral general* de dicha ecuación diferencial. Por consiguiente, la ecuación (5.11) es integral general de la ecuación (5.10). Para que (5.11) determine y como función implícita de x, es suficiente exigir que $f_2(y) \neq 0$.

Es posible que en algunos problemas no sea posible expresar las integrales indefinidas $\int f_1(x)\,dx$ y $\int f_2(y)\,dy$ en funciones elementales; pero, a pesar de esto, consideramos resuelto también en este caso el problema de la integración de la ecuación diferencial (5.10), en el

sentido de que lo hemos reducido a un problema simple del cálculo de integrales indefinidas. Como el término integral en teoría de ecuaciones diferenciales se utiliza frecuentemente en el sentido de integral de la ecuación diferencial, entonces para referirnos a integrales de funciones, $\int f(x)\,dx$, generalmente se utiliza el término "cuadratura".

Si hay que obtener la solución particular que satisface la condición $y(x_0) = y_0$, ésta evidentemente se determina por la ecuación

$$\int_{y_0}^{y} f_2(y)\,dy = \int_{x_0}^{x} f_1(x)\,dx \ ,$$

la cual se obtiene de

$$\int_{y_0}^{y} f_2(y)\,dy = \int_{x_0}^{x} f_1(x)\,dx + c \ ,$$

utilizando las condiciones iniciales $y(x_0) = y_0$.

Ejemplo Considere la siguiente ecuación

$$xdx + ydy = 0 \ .$$

Las variables están separadas, ya que el coeficiente de dx es función sólo de x, y el coeficiente de dy, sólo de y. Integrando, obtenemos

$$\int x\,dx + \int y\,dy = c \quad \text{o bien } x^2 + y^2 = c_1^2 \ ,$$

que es una familia de circunferencias con centro en el origen de coordenadas.

Las ecuaciones del tipo

$$\phi_1(x)\psi_1(y)\,dx = \phi_2(x)\psi_2(y)\,dy \ ,$$

en las cuales los coeficientes de las diferenciales se descomponen en factores dependientes sólo de x o de y, se llaman *ecuaciones diferenciales con variables separables*, ya que dividiendo entre $\psi_1(y)\phi_2(x)$, éstas se reducen a una ecuación de variables separadas:

$$\frac{\phi_1(x)}{\phi_2(x)}\,dx = \frac{\psi_2(y)}{\psi_1(y)}\,dy \ ,$$

Obsérvese que la división entre $\psi_1(y)\phi_2(x)$ puede conducir a la pérdida de soluciones particulares, que reducen a cero el producto $\psi_1(y)\phi_2(x)$; si las funciones $\psi_1(y)$ y $\phi_2(x)$ pueden ser discontinuas, es posible la aparición de soluciones superfluas, que reducen a cero el factor

$$\frac{1}{\psi_1(y)\phi_2(x)} \ .$$

Ejemplo Como ya fue mencionado, se ha establecido que la velocidad de desintegración radiactiva es proporcional a la cantidad x de sustancia aún no desintegrada. Hallar la dependencia de x respecto al tiempo t, si en el momento inicial para $t = t_0$, era $x = x_0$.

Supondremos conocido el coeficiente de proporcionalidad k, llamado constante de desintegración. La ecuación diferencial que gobierna el proceso tendrá la forma

$$\frac{dx}{dt} = -kx \ ,$$

El signo menos en el término derecho de la ecuación indica que x decrece cuando t aumenta, $k > 0$. Separando variables e integrando, se obtiene

$$\frac{dx}{x} = -k\,dt \ ; \quad \ln(|\,x\,|) - \ln(|\,x_0\,|) = -k\,(t - t_0) \ ,$$

de donde

$$x = x_0\, e^{-k(t-t_0)} \ .$$

Determinemos también el período de desintegración τ (o sea, el tiempo durante el cual se desintegra $x_0/2$). Haciendo $t - t_0 = \tau$, obtenemos $x_0/2 = x_0 e^{-k\tau}$, de donde $\tau = \ln 2/k$.

La ecuación

$$\frac{dx}{dt} = kx \ , \quad k > 0 \ ,$$

se diferencia sólo en el signo del segundo miembro de la ecuación anterior, sin embargo, describe procesos de "reproducción" completamente diferentes, por ejemplo: la variación de la cantidad de neutrones en las reacciones en nucleares en cadena, o la reproducción de una colonia de bacterias en condiciones ideales. La solución de esta ecuación que satisface la condición inicial $x(t_0) = x_0$ tiene la forma

$$x = x_0\, e^{k(t-t_0)} \ ,$$

y, a diferencia de las soluciones anteriores, $x(t)$ no disminuye, sino que crece exponencialmente con el incremento de t.

5.4. Ecuaciones que se reducen a ecuaciones de variables separables

Muchas ecuaciones diferenciales pueden ser reducidas a ecuaciones con variables separables mediante una sustitución de variables. A dicho grupo pertenecen, por ejemplo, las ecuaciones de la forma

$$\frac{dy}{dx} = f(ax + by) \ ,$$

donde a y b son magnitudes constantes, las cuales se transforman en ecuaciones con variables separables por medio de la sustitución $z = ax + by$. Efectivamente, pasando a las nuevas variables x y z, tendremos

$$\frac{dz}{dx} = a + b\frac{dy}{dx} \ , \quad \frac{dz}{dx} = a + bf(z) \ ,$$

o bien

$$\frac{dz}{a + bf(z)} = dx \ ,$$

con lo que hemos separado las variables. Integrando, obtenemos

$$x = \int \frac{dz}{a + bf(z)} + c \ .$$

Ejemplo Consideremos la siguiente ecuación diferencial

$$\frac{dy}{dx} = \frac{1}{x - y} + 1 \ .$$

Haciendo $x - y = z$, obtenemos

$$\frac{dy}{dx} = 1 - \frac{dz}{dx} \quad 1 - \frac{dz}{dx} = \frac{1}{z} + 1 \ ;$$

$$\frac{dz}{dx} = -\frac{1}{z} \ , \quad z dz = -dx \ , \quad z^2 = -2x + c \ , \quad (x - y)^2 = -2x + c \ .$$

5.4.1. Ecuaciones homogéneas.

Las *ecuaciones diferenciales homogéneas de primer orden*, que tienen la forma

$$\frac{dy}{dx} = f\left(\frac{y}{x}\right) \ ,$$

pueden reducirse a ecuaciones con variables separables. En efecto, después de la sustitución $z = \dfrac{y}{x}$, o bien $y = xz$, obtenemos

$$\frac{dy}{dx} = x\frac{dz}{dx} + z \ , \quad x\frac{dz}{dx} + z = f(z) \ , \quad \frac{dz}{f(z) - z} = \frac{dx}{x} \ ,$$

$$\int \frac{dz}{f(z) - z} = \ln |\, x \,| + \ln c \ , \quad x = c\, e^{\int \frac{dz}{f(z) - z}} \ .$$

Obsérvese que el segundo miembro de la ecuación homogénea es un función homogénea en las variables x e y de grado nulo[2] de homogeneidad; por eso la ecuación del tipo

$$M(x, y)\, dx + N(x, y)\, dy = 0 \ ,$$

será homogénea si $M(x, y)$ y $N(x, y)$ son funciones homogéneas de x e y, del mismo grado de homogeneidad, puesto que en este caso

$$\frac{dy}{dx} = -\frac{M(x, y)}{N(x, y)} = f\left(\frac{y}{x}\right) \ .$$

[2]Una función $f(u, v)$ es homogénea de grado n si al multiplicar las variables por λ, la función resultante es λ^n veces la función original.

Ejemplo Considere la siguiente ecuación

$$\frac{dy}{dx} = \frac{y}{x} + \tan\frac{y}{x} \ .$$

Haciendo $y = xz$, $\dfrac{dy}{dx} = x\dfrac{dz}{dx} + z$ y sustituyendo en la ecuación inicial, obtenemos

$$x\frac{dz}{dx} + z = z + \tan z \ , \quad \frac{\cos z\, dz}{\operatorname{sen} z} = \frac{dx}{x} \ ,$$

$$\ln|\operatorname{sen} z| = \ln|x| + \ln c \ , \quad \operatorname{sen} z = cx \quad \operatorname{sen}\frac{y}{x} = cx \ .$$

5.4.2. Ecuaciones que se reducen a homogéneas.

Las ecuaciones del tipo

$$\frac{dy}{dx} = f\left(\frac{a_1x + b_1y + c_1}{a_2x + b_2y + c_2}\right) \ , \tag{5.12}$$

pueden reducirse a ecuaciones homogéneas, si trasladamos el origen de coordenadas al punto de intersección (x_1, y_1) de las rectas

$$a_1x + b_1y + c_1 = 0 \ , \qquad a_2x + b_2y + c_2 = 0 \ .$$

Efectivamente, el miembro independiente en las ecuaciones de estas rectas en las nuevas coordenadas $X = x - x_1$, $Y = y - y_1$, será igual a cero; los coeficientes de las coordenadas permanecen invariables; $\dfrac{dy}{dx} = \dfrac{dY}{dX}$, y la ecuación (5.12) toma la forma

$$\frac{dY}{dX} = f\left(\frac{a_1X + b_1Y}{a_2X + b_2Y}\right) \ ,$$

o bien

$$\frac{dY}{dX} = f\left(\frac{a_1 + b_1\frac{Y}{X}}{a_2 + b_2\frac{Y}{X}}\right) = \varphi\left(\frac{Y}{X}\right) \ ,$$

que ya es una ecuación homogénea.

Este método no se puede aplicar sólo en el caso en que haya paralelismo entre las rectas $a_1x + b_1y + c_1 = 0$ y $a_2x + b_2y + c_2 = 0$. Pero en este caso los coeficientes de las coordenadas son proporcionales: $\dfrac{a_2}{a_1} = \dfrac{b_2}{b_1} = k$, y la ecuación (5.12) se puede escribir en la forma

$$\frac{dy}{dx} = f\left(\frac{a_1x + b_1y + c_1}{k(a_1x + b_1y) + c_2}\right) = F(a_1x + b_1y) \ ,$$

como ya sabemos, el cambio de variable $z = a_1x + b_1y$ transforma la ecuación considerada en una ecuación con variables separables.

Ejemplo Considere la siguiente ecuación

$$\frac{dy}{dx} = \frac{x-y+1}{x+y-3} \ .$$

Resolviendo el sistema de ecuaciones $x-y+1=0$, $x+y-3=0$, obtenemos $x_1=1$, $y_1=2$. Haciendo $x=X+1$, $y=Y+2$, tendremos

$$\frac{dY}{dX} = f\left(\frac{X-Y}{X+Y}\right) \ .$$

El cambio de variable $Y=zX$ conduce a una ecuación de variables separables:

$$z + X\frac{dz}{dX} = \frac{1-z}{1+z} \ ,$$

$$\frac{(1+z)\,dz}{1-2z+z^2} = \frac{dX}{X} \ ,$$

$$-\frac{1}{2}\ln\left|\,1-2z-z^2\,\right| = \ln|\,X\,| - \frac{1}{2}\ln c \ ,$$

$$(1-2z-z^2)X^2 = c \ ,$$

$$X^2 - 2XY - Y^2 = c \ ,$$

$$x^2 - 2xy - y^2 + 2x + 6y = c_1 \ .$$

5.5. Ecuaciones lineales de primer orden.

Se llama *ecuación diferencial de primer orden* a una ecuación lineal con respecto a la función desconocida y a su derivada. La ecuación lineal tiene la forma

$$\frac{dy}{dx} + p(x)y = f(x) \ , \tag{5.13}$$

donde $p(x)$ y $f(x)$ se considerarán en lo sucesivo funciones continuas de x en la región en que se exige integrar la ecuación (5.13).

Si $f(x) \equiv 0$, la ecuación (5.13) se llama *lineal homogénea*. En la ecuación lineal homogénea las variables se separan

$$\frac{dy}{dx} + p(x)y = 0 \ , \quad \text{de donde } \frac{dy}{y} = -p(x)\,dx \ ,$$

e integrando, obtenemos

$$\ln|\,y\,| = -\int p(x)\,dx + \ln c_1 \ , \quad c_1 > 0 \ ,$$

$$y = c_1\, e^{-\int p(x)\,dx} \ , \quad c_1 \neq 0 \ . \tag{5.14}$$

Al dividir entre y se perdió la solución $y=0$; sin embargo, ésta puede ser incluida en la familia de soluciones halladas (5.14), si se considera que c puede tomar el valor 0.

5.5.1. Ecuaciones lineales no homogéneas.

Para integrar la ecuación lineal no homogénea

$$\frac{dy}{dx} + p(x)y = f(x) \ . \tag{5.13}$$

Así puede ser aplicado el llamado *método de variación de la constante*. Al aplicar dicho método, primeramente se integra la ecuación homogénea correspondiente (o sea, la que tiene el mismo primer miembro):

$$\frac{dy}{dx} + p(x)y = 0 \ ,$$

cuya solución general, como fue indicado anteriormente, tiene la forma

$$y = c\, e^{-\int p(x')\,dx'} \ .$$

Cuando c es constante, la función $c\, e^{-\int p(x')\,dx'}$ es la solución de la ecuación homogénea. Probemos ahora satisfacer la ecuación no homogénea considerando c como función de x, o sea, realizando en esencia la sustitución de variables

$$y = c(x)\, e^{-\int p(x')\,dx'} \ ,$$

donde $c(x)$ es una función desconocida de x.

Calculando la derivada

$$\frac{dy}{dx} = \frac{dc}{dx}\, e^{-\int p(x')\,dx'} - c(x)p(x)\, e^{-\int p(x')\,dx'} \ ,$$

y sustituyéndola en la ecuación no homogénea inicial (5.13), se obtiene

$$\frac{dc}{dx}\, e^{-\int p(x')\,dx'} - c(x)p(x)\, e^{-\int p(x')\,dx'} + p(x)c(x)\, e^{-\int p(x')\,dx'} = f(x) \ ,$$

o bien

$$\frac{dc}{dx} = f(x)\, e^{\int p(x')\,dx'} \ ,$$

de donde, integrando, se halla

$$c(x) = \int f(x'')e^{\int p(x')\,dx'}\,dx'' + c_1 \ ,$$

y, por consiguiente,

$$\boxed{y = c(x)\, e^{-\int p(x')\,dx'} = c_1\, e^{-\int p(x')\,dx'} + e^{-\int p(x')\,dx'} \int f(x'')e^{\int p(x')\,dx'}\,dx'' \ .} \tag{5.15}$$

De este modo, la solución general de la ecuación lineal no homogénea es igual a la suma de la solución general de la ecuación homogénea correspondiente

$$c_1\, e^{-\int p(x')\,dx'} \ ,$$

y de la solución particular de la ecuación no homogénea

$$e^{-\int p(x')\,dx'} \int f(x'')e^{\int p(x')\,dx'}\,dx'' \ ,$$

que se obtiene de (5.15) si $c_1 = 0$.

Obsérvese que en ejemplos concretos no es conveniente utilizar la fórmula (5.15) , compleja y difícil de recordar; es más sencillo repetir cada vez todas las operaciones expuestas más arriba.

Ejemplo En un circuito eléctrico con auto inducción tiene lugar el paso de corriente alterna. La tensión U es un función dada del tiempo, $U = U(t)$, la resistencia R y la auto inducción L son constantes; la corriente inicial es dada, $I(0) = I_0$. Hallar la dependencia de la intensidad de la corriente $I = I(t)$ respecto al tiempo. Aplicando la ley de Ohm para el circuito obtenemos

$$U - L\frac{dI}{dt} = RI \ .$$

La solución de esta ecuación lineal que satisface la condición inicial $I(0) = I_0$, de acuerdo con (5.15), tiene la forma

$$I = e^{-\frac{R}{L}t}\left[I_0 + \frac{1}{L}\int_0^t U(t')e^{\frac{R}{L}t'}\,dt'\right] \ . \tag{5.16}$$

Para una tensión constante $U = U_0$, obtenemos

$$I = \frac{U_0}{R} + \left(I_0 - \frac{U_0}{R}\right)e^{-\frac{R}{L}t} \ .$$

Es interesante el caso de tensión alterna sinusoidal: $U = A\,\mathrm{sen}\,\omega t$. En este caso, según (5.16), obtenemos

$$I = e^{-\frac{R}{L}t}\left[I_0 + \frac{A}{L}\int_0^t \mathrm{sen}\,\omega t' e^{\frac{R}{L}t'}\,dt'\right] \ .$$

La integral del segundo miembro se toma fácilmente.

5.5.2. Ecuaciones de Bernoulli y Riccati.

Muchas ecuaciones diferenciales pueden ser reducidas a ecuaciones lineales mediante un cambio de variables. Por ejemplo, la *ecuación de Bernouilli*, que tiene la forma

$$\frac{dy}{dx} + p(x)y = f(x)y^n \ , \quad n \neq 1 \ ,$$

o bien

$$y^{-n}\frac{dy}{dx} + p(x)y^{1-n} = f(x) \ , \tag{5.17}$$

con el cambio de variable $y^{1-n} = z$, se reduce a una ecuación lineal. Efectivamente, derivando $y^{1-n} = z$, hallamos

$$(1-n)y^{-n}\frac{dy}{dx} = \frac{dz}{dx} \ ,$$

y sustituyendo en (5.17), obtenemos la ecuación lineal

$$\frac{1}{1-n}\,\frac{dz}{dx} + p(x)\,z = f(x)\ .$$

Ejemplo Consideremos la ecuación

$$\frac{dy}{dx} = \frac{y}{2x} + \frac{x^2}{2y}\ ,$$

o bien

$$2y\frac{dy}{dx} = \frac{y^2}{x} + x^2\ ,$$

con el cambio de variable $y^2 = z$ tenemos $2y\dfrac{dy}{dx} = \dfrac{dz}{dx}$, con lo que la ecuación nos queda

$$\frac{dz}{dx} = \frac{z}{x} + x^2\ ,$$

que es una ecuación diferencial lineal de primer orden no homogénea.

La ecuación

$$\frac{dy}{dx} + p(x)y + q(x)y^2 = f(x)\ ,$$

llamada *ecuación de Riccati*, en general no se integra en cuadraturas, pero por sustitución de variables puede ser transformada en una ecuación de Bernoulli, si se conoce una solución particular $y_1(x)$ de esta ecuación. Efectivamente, haciendo $y = y_1 + z$, se obtiene

$$y_1' + z' + p(x)(y_1 + z) + q(x)(y_1 + z)^2 = f(x)\ ,$$

o, como $y_1' + p(x)y_1 + q(x)y_1^2 \equiv f(x)$, tendremos la ecuación de Bernoulli

$$z' + [p(x) + 2q(x)y_1]z + q(x)z^2 = 0\ .$$

Ejemplo Consideremos la ecuación

$$\frac{dy}{dx} = y^2 - \frac{2}{x^2}\ .$$

Aquí no es difícil hallar la solución particular $y_1 = \frac{1}{x}$. Haciendo $y = z + \frac{1}{x}$, obtenemos

$$y' = z' + \frac{1}{x^2}\ , \quad z' - \frac{1}{x^2} = \left(z - \frac{1}{x2}\right)^2 - \frac{2}{x^2}\ ,$$

o bien

$$z' = z^2 + 2\frac{z}{x}\ ,$$

que es una ecuación de Bernoulli.

$$\frac{z'}{z^2} = \frac{2}{xz} + 1\ , \quad u = \frac{1}{z}\ , \quad \frac{du}{dx} = -\frac{z'}{z^2}\ ,$$

$$\frac{du}{dx} = -\frac{2u}{x} - 1 \ , \quad \frac{du}{u} = -\frac{2dx}{x} \ ,$$

$$\ln|u| = -2\ln|x| + \ln c \ , \quad u = \frac{c}{x^2} \ , \quad u = \frac{c(x)}{x^2} \ ,$$

$$\frac{c'(x)}{x^2} = -1 \ , \quad c(x) = -\frac{x^3}{3} + c_1 \ ,$$

$$u = \frac{c_1}{x^2} - \frac{x}{3} \ , \quad \frac{1}{z} = \frac{c_1}{x^2} - \frac{x}{3} \ , \quad \frac{1}{y - \frac{1}{x}} = \frac{c_1}{x^2} - \frac{x}{3} \ ,$$

$$y = \frac{1}{x} + \frac{3x^2}{c_2 - x^3} \ .$$

5.6. Ecuaciones en diferenciales totales.

Puede suceder que el primer miembro de la ecuación diferencial

$$M(x,y)\,dx + N(x,y)\,dy = 0 \ , \tag{5.18}$$

sea la diferencial total de cierta función $u(x,y)$:

$$du(x,y) = M(x,y)\,dx + N(x,y)\,dy \ ,$$

y que por consiguiente, la ecuación (5.18) tome la forma

$$du(x,y) = 0 \ .$$

Si la función $y(x)$ es solución de la ecuación (5.18), entonces

$$u(x, y(x)) = c \ , \tag{5.19}$$

donde c es una constante. recíprocamente, si cierta función $y(x)$ convierte en identidad la ecuación finita (5.19), entonces, derivando la identidad obtenida, tendremos $du(x, y(x)) = 0$ y, por consiguiente, $u(x,y) = c$, donde c es una constante arbitraria, es integral general de la ecuación inicial.

Si las condiciones iniciales $y(x_0) = y_0$ están dadas, la constante c se determina de (5.19): $c = u(x_0, y_0)$ y

$$u(x, y(x)) = u(x_0, y_0) \ , \tag{5.20}$$

es la integral particular buscada. Si $\dfrac{\partial u}{\partial y} = N(x,y) \neq 0$ en el punto (x_0, y_0), entonces la ecuación (5.20) determina y como función implícita de x.

Para que el primer miembro de la ecuación (5.18)

$$M(x,y)\,dx + N(x,y)\,dy \ ,$$

sea un diferencial total de cierta función $u(x, y)$, como se sabe, es necesario y suficiente que

$$\frac{\partial M(x,y)}{\partial y} \equiv \frac{\partial N(x,y)}{\partial x} \ . \tag{5.21}$$

Si esta condición, señalada inicialmente por Euler, se cumple, entonces (5.18) se integra fácilmente. En efecto

$$du = M\,dx + N\,dy \ .$$

Por otra parte,

$$du = \frac{\partial u}{\partial x}\,dx + \frac{\partial u}{\partial y}\,dy \ .$$

Por consiguiente,

$$\frac{\partial u}{\partial x} = M(x,y) \ ; \qquad \frac{\partial u}{\partial y} = N(x,y) \ ,$$

de donde

$$u(x,y) = \int M(x,y)\,dx + c(y) \ .$$

Al calcular la integral de la expresión anterior, la magnitud y se considera constante; por eso, $c(y)$ es una función arbitraria de y.

Para determinar la función $c(y)$, derivamos la función hallada $u(x, y)$ respecto a y y, como $\dfrac{\partial u}{\partial y} = N(x, y)$, obtenemos

$$\frac{\partial}{\partial y}\left(\int M(x,y)\,dx\right) + c'(y) = N(x,y) \ .$$

De esta ecuación se determina $c'(y)$, e integrando se halla $c(y)$.

Se puede determinar aún más fácilmente la función $u(x, y)$ por su diferencial total $du = M(x, y)\,dx + N(x, y)\,dy$, tomando la integral curvilínea de $M(x, y)\,dx + N(x, y)\,dy$ desde cierto punto fijo (x_0, y_0) hasta un punto con coordenadas variables (x, y), por cualquier camino:

$$u(x,y) = \int_{(x_0,y_0)}^{(x,y)} M(x,y)\,dx + N(x,y)\,dy \ .$$

Con frecuencia, es cómodo tomar una línea quebrada, compuesta por dos segmentos paralelos a los ejes de coordenadas (figura 5.3); en este caso

$$\int_{(x_0,y_0)}^{(x,y)} M\,dx + N\,dy = \int_{(x_0,y_0)}^{(x,y_0)} M\,dx + \int_{(x,y_0)}^{(x,y)} N\,dy \ ,$$

o bien

$$\int_{(x_0,y_0)}^{(x,y)} M\,dx + N\,dy = \int_{(x_0,y_0)}^{(x_0,y)} N\,dy + \int_{(x_0,y_0)}^{(x,y)} N\,dy \ .$$

Ejemplo Consideremos la siguiente ecuación

$$(x + y + 1)\,dx + (x - y^2 + 3)\,dy = 0 \ .$$

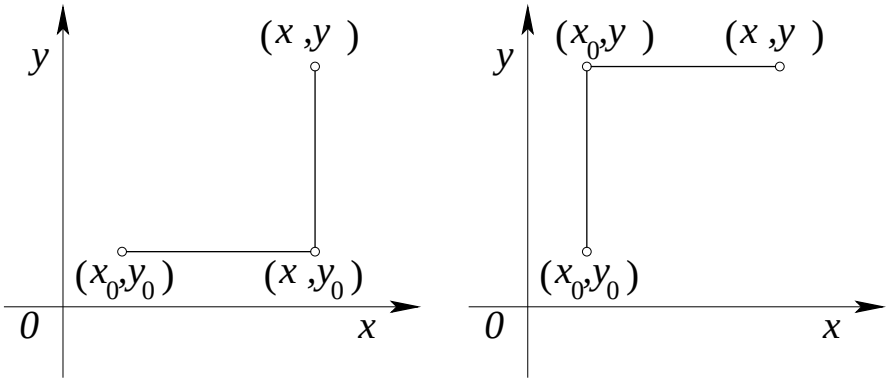


Figura 5.3: Caminos de integración posibles.

El primer miembro de la ecuación es la diferencial total de cierta función $u(x, y)$, puesto que

$$\frac{\partial(x+y+1)}{\partial y} \equiv \frac{\partial(x-y^2+3)}{\partial x} \ ,$$

$$\frac{\partial u}{\partial x} = x+y+1 \quad u = \frac{x^2}{2} + xy + x + c(y) \ ,$$

$$\frac{\partial u}{\partial y} = x + c'(y) \ , \quad x + c'(y) = x - y^2 + 3 \ ,$$

$$c'(y) = -y^2 + 3 \ , \quad c(y) = -\frac{y^3}{3} + 3y + c_1 \ .$$

Por lo tanto, la integral general tiene la forma

$$3x^2 + 6xy + 6x - 2y^3 + 18y = c_2 \tag{5.22}$$

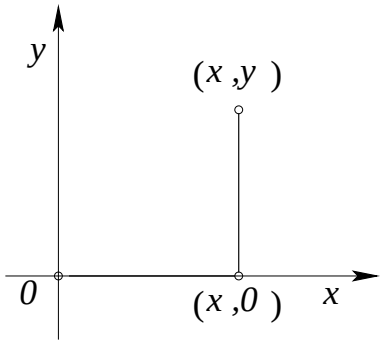


Figura 5.4: Camino de integración.

Se puede utilizar también el otro método de determinación de la función $u(x, y)$:

$$u(x,y) = \int_{(x_0,y_0)}^{(x,y)} (x+y+1)\,dx + (x-y^2+3)\,dy \ .$$

Como punto inicial (x_0, y_0) escogemos por ejemplo el origen de coordenadas, y como camino de integración el mostrado en la figura (5.4). Entonces

$$u(x,y) = \int_{(0,0)}^{(x,0)} (x+1)\,dx + \int_{(x,0)}^{(x,y)} (x-y^2+3)\,dy \ ,$$

$$u(x,y) = \frac{x^2}{2} + x + xy - \frac{y^3}{3} + 3y \ ,$$

y la integral general tiene la forma

$$\frac{x^2}{2} + x + xy - \frac{y^3}{3} + 3y = c \ ,$$

o bien como en (5.22).

5.6.1. Factor integrante.

En algunos casos, si el primer miembro de la ecuación

$$M(x,y)\,dx + N(x,y)\,dy = 0\ , \tag{5.18}$$

no es una diferencial total, resulta fácil escoger una función $\mu(x,y)$, tal que el producto de está y el miembro izquierdo de (5.18), se transforma en una diferencial total:

$$du = \mu M\,dx + \mu N\,dy\ .$$

Esta función μ se llama *factor integrante*. Obsérvese que la multiplicación por el factor integrante $\mu(x,y)$ puede conducir a que aparezcan soluciones particulares superfluas, que reducen este factor a cero.

Ejemplo Consideremos la siguiente ecuación

$$x\,dx + y\,dy + (x^2+y^2)x^2\,dx = 0\ .$$

es fácil comprobar que después de multiplicar por el factor $\mu = 1/(x^2+y^2)$, el primer miembro se transforma en diferencial total. Efectivamente, luego del producto por $\mu = 1/(x^2+y^2)$, obtenemos

$$\frac{x\,dx + y\,dy}{x^2+y^2} + x^2\,dx = 0\ ,$$

o integrando:

$$\frac{1}{2}\ln(x^2+y^2) + \frac{x^3}{3} = \ln c_1\ .$$

multiplicando por 2 y potenciando, tendremos

$$(x^2+y^2)\,e^{2x^3/3} = c\ .$$

Es claro que no siempre el factor integrante se escoge tan fácilmente. En general, para hallar dicho factor es necesario escoger por lo menos una solución particular no idénticamente nula de la ecuación en derivadas parciales

$$\frac{\partial \mu M}{\partial y} = \frac{\partial \mu N}{\partial x}\ ,$$

o, en forma desarrollada,

$$\frac{\partial \mu}{\partial y}M + \mu\frac{\partial M}{\partial y} = \frac{\partial \mu}{\partial x}N + \mu\frac{\partial N}{\partial x}\ ,$$

la cual, después de dividir entre μ y de cambiar de miembro algunos términos, se reduce a

$$\frac{\partial \ln\mu}{\partial y}M - \frac{\partial \ln\mu}{\partial x}N = \frac{\partial N}{\partial x} - \frac{\partial M}{\partial y}\ . \tag{5.23}$$

En general, la integración de esta ecuación en derivadas parciales no es un problema más simple que la integración de la ecuación inicial. Sin embargo, a veces la elección de la solución particular de (5.23) no presenta dificultades.

Aparte de ello, considerando que el factor integrante es función de un solo argumento (por ejemplo, es función sólo de $x+y$ o de x^2+y^2, o función sólo de x o de y, etc.), se puede integrar ya sin dificultad la ecuación (5.23) e indicar las condiciones bajo las cuales existe un factor integrante del tipo considerado. Con esto se obtienen clases de ecuaciones para las cuales el factor integrante puede ser hallado fácilmente.

Por ejemplo, encontremos las condiciones bajo las cuales la ecuación $M\,dx + N\,dy = 0$ tiene factor integrante que depende sólo de x, $\mu = \mu(x)$. En este caso, la ecuación (5.23) se simplifica y toma la forma

$$-\frac{d\ln\mu}{dx}N = \frac{\partial N}{\partial x} - \frac{\partial M}{\partial y}\ ,$$

de donde, considerando $\dfrac{\frac{\partial M}{\partial y} - \frac{\partial N}{\partial x}}{N}$ función continua de x, obtenemos

$$\ln\mu = \int \frac{\frac{\partial M}{\partial y} - \frac{\partial N}{\partial x}}{N}\,dx + \ln c\ ,$$

$$\mu = c\,\exp\left(\int \frac{\frac{\partial M}{\partial y} - \frac{\partial N}{\partial x}}{N}\,dx\right)\ . \tag{5.24}$$

Se puede considerar $c = 1$, ya que es suficiente tener sólo un factor integrante.

Si $\dfrac{\frac{\partial M}{\partial y} - \frac{\partial N}{\partial x}}{N}$ es función sólo de x, entonces existe un factor integrante que sólo depende de x, y es igual a (5.24). En caso contrario, no existe ningún factor de la forma $\mu(x)$.

La condición de existencia de un factor integrante que depende sólo de x se cumple, por ejemplo, para la ecuación lineal

$$\frac{dy}{dx} + p(x)y = f(x)\ , \qquad \text{o bien } \ [p(x)y - f(x)]\,dx + dy = 0\ .$$

Efectivamente, $\dfrac{\frac{\partial M}{\partial y} - \frac{\partial N}{\partial x}}{N} = p(x)$ y, por lo tanto, $\mu = e^{\int p(x)\,dx}$. De manera análoga se pueden hallar las condiciones de existencia de factores integrantes de la forma

$$\mu(y)\ ,\ \mu(x \pm y)\ ,\ \mu(x^2 \pm y^2)\ ,\ \mu(xy)\ ,\ \mu\left(\frac{x}{y}\right) \text{ etc.}$$

Ejemplo ¿Tiene la ecuación

$$x\,dx + y\,dy + x\,dy - y\,dx = 0\ , \tag{5.25}$$

un factor integrante de la forma $\mu = \mu(x^2+y^2)$?

Designemos $x^2+y^2 = z$. La ecuación (5.23) para $\mu = \mu(x^2+y^2) = \mu(z)$ toma la forma

$$2(My - Nx)\frac{\ln\mu}{dz} = \frac{\partial N}{\partial x} - \frac{\partial M}{\partial y}\ ,$$

de donde

$$\ln|\,\mu\,| = \frac{1}{2}\int \varphi(z)\,dz + \ln c\ ,$$

o bien

$$\mu = c\,\exp\left(\frac{1}{2}\int \varphi(z)\,dz\right)\ , \tag{5.26}$$

donde

$$\varphi(z) = \frac{\dfrac{\partial N}{\partial x} - \dfrac{\partial M}{\partial y}}{My - Nx}\ .$$

Para la existencia de un factor integrante del tipo dado, es necesario (y en caso que $\varphi(z)$ sea continua, suficiente) que φ sea función sólo de $x^2 + y^2$. En nuestro caso,

$$\frac{\dfrac{\partial N}{\partial x} - \dfrac{\partial M}{\partial y}}{My - Nx} = -\frac{2}{x^2 + y^2}\ ;$$

por lo tanto, el factor integrante $\mu = \mu(x^2 + y^2)$ existe y es igual a (5.26). Para $c = 1$, obtenemos:

$$\mu = \exp\left(-\int \frac{dz}{z}\right) = \frac{1}{z} = \frac{1}{x^2 + y^2}\ .$$

Multiplicando la ecuación (5.25) por el μ que determinamos, la reducimos a la forma

$$\frac{x\,dx + y\,dy}{x^2 + y^2} + \frac{x\,dy - y\,dx}{x^2 + y^2} = 0\ ,$$

o bien

$$\frac{\dfrac{1}{2}d(x^2 + y^2)}{x^2 + y^2} + \frac{\left(d\dfrac{y}{x}\right)}{1 + \left(\dfrac{y}{x}\right)^2} = 0\ , \quad \frac{1}{2}d\ln(x^2 + y^2) + d\arctan\frac{y}{x} = 0\ .$$

Integrando, obtenemos

$$\ln\sqrt{x^2 + y^2} = -\arctan\frac{y}{x} + \ln c\ , \quad \sqrt{x^2 + y^2} = c\,\exp\left(-\arctan\frac{y}{x}\right)\ ,$$

o bien en coordenadas polares $\rho = c\,e^{-\varphi}$, que es una familia de espirales logarítmicas.

5.7. Teoremas.

Teorema 5.1 Existencia y unicidad de la solución.
Si en la ecuación

$$\frac{dy}{dx} = f(x, y)\ , \tag{5.27}$$

la función $f(x, y)$ es continua en el rectángulo D:

$$x_0 - a \le x \le x_0 + a\ , \quad y_0 - b \le y \le y_0 + b\ , \tag{5.28}$$

y satisface en D la condición de Lipschitz:

$$|\,f(x, y_1) - f(x, y_2)\,| \le N\,|\,y_1 - y_2\,|\ ,$$

donde N es una constante, entonces existe una solución única $y = y(x)$, $x_0 - H \leq x \leq x_0 + H$ de la ecuación (5.27), que satisface la condición $y(x_0) = y_0$, donde

$$H < \text{mín}\left(a\ ,\ \frac{b}{M}\ ,\ \frac{1}{N}\right)$$
$$M = \text{máx}\,|\,f(x,y)\,|\ \text{ en } D.$$

Teorema 5.2 Sobre la dependencia continua de la solución con respecto al parámetro y a los valores iniciales.

Si el segundo miembro de la ecuación diferencial

$$\frac{dy}{dx} = f(x, y, \mu)\ , \tag{5.29}$$

es continuo en μ para $\mu_0 \leq \mu \leq \mu_1$ y satisface la condiciones del teorema de existencia y unicidad, y la constante de Lipschitz N no depende de μ, entonces la solución $y(x, y)$ de la ecuación considerada que satisface la condición $y(x_0) = y_0$ depende en forma continua de μ.

Teorema 5.3 Sobre la derivabilidad de las soluciones.

Si en un entorno del punto (x_0, y_0) la función $f(x, y)$ tiene derivadas continuas hasta k-ésimo orden inclusive, la solución $y(x)$ de la ecuación

$$\frac{dy}{dx} = f(x, y)\ ,$$

que satisface la condición inicial $y(x_0) = y_0$, tendrá derivadas continuas hasta $k + 1$-ésimo orden, inclusive en cierto entorno del punto (x_0, y_0).

Capítulo 6

Ecuaciones diferenciales de orden mayor que uno.

versión final 2.3-021107[1]

6.1. Teorema de existencia y unicidad para la ecuación diferencial de n-ésimo orden.

Las ecuaciones diferenciales de n-ésimo orden tienen la forma

$$y^{(n)} = f(x, y, y', \ldots, y^{(n-1)}) \ , \tag{6.1}$$

o bien, si no están resueltas con respecto a la derivada de orden mayor:

$$F(x, y, y', \ldots, y^{(n)}) = 0 \ .$$

El teorema de existencia y unicidad para ecuación de n-ésimo orden se puede obtener fácilmente, llevándola a un sistema de ecuaciones.

Teorema 6.1 Si en un entorno de las condiciones iniciales $(x_0, y_0, y_0', \ldots, y_0^{(n-1)})$ la función f es continua en todos sus argumentos y satisface la condición de Lipschitz respecto a todos los argumentos a partir del segundo, existe una solución única de la ecuación diferencial de n-ésimo orden $y^{(n)} = f(x, y, y', ..., y^{(n-1)})$ que satisface las condiciones

$$y(x_0) = y_0 \ , \quad y'(x_0) = y_0' \ , \quad y''(x_0) = y_0'' \ , \quad \ldots, \quad y^{(n-1)}(x_0) = y_0^{(n-1)} \ .$$

6.1.1. Solución general.

Se llama solución general de la ecuación diferencial de n-ésimo orden al conjunto de soluciones formado por todas las soluciones particulares, sin excepción. Si el segundo miembro de la ecuación

$$y^{(n)} = f(x, y, y', \ldots, y^{(n-1)}) \ , \tag{6.1}$$

[1]Este capítulo está basado en el segundo capítulo del libro: *Ecuaciones diferenciales y cálculo variacional* de L. Elsgoltz, editorial MIR.

satisface, en cierta región de variación de sus argumentos, las condiciones del teorema de existencia y unicidad, entonces la solución general de la ecuación (6.1) depende de n parámetros, en calidad de los cuales se pueden tomar, por ejemplo, las condiciones iniciales de la función buscada y de sus derivadas y_0, y_0', y_0'',..., $y_0^{(n-1)}$.

En particular, la solución general de la ecuación de segundo grado $y'' = f(x, y, y')$ depende de dos parámetros, por ejemplo, de y_0 y de y_0'. Si fijamos y_0 e y_0', o sea, damos el punto (x_0, y_0), y la dirección de la tangente a la curva integral buscada en dicho punto, entonces, si se cumplen las condiciones del teorema de existencia y unicidad, se determinará una sola curva integral, mediante estas condiciones.

Por ejemplo, la ecuación del movimiento rectilíneo de un punto material de masa m bajo la acción de la fuerza $f(t, x, \dot{x})$:

$$m\ddot{x} = f(t, x, \dot{x}) \ ,$$

la posición inicial del punto $x(t_0) = x_0$ y la velocidad inicial $\dot{x}(t_0) = \dot{x_0}$ determinan una solución única, una trayectoria única $x = x(t)$ si, por supuesto, la función f satisface las condiciones del teorema de existencia y unicidad.

6.2. Casos simples de reducción del orden.

En ciertos casos el orden de la ecuación diferencial puede ser reducido, lo que a menudo facilita su integración.

Señalemos algunas clases de ecuaciones que se encuentran con mayor frecuencia y que pueden reducir su orden.

1. La ecuación no contiene la función buscada y sus derivadas hasta el orden $k-1$ inclusive:

$$F(x, y^{(k)}, y^{(k+1)}, \ldots, y^{(n)}) = 0 \ . \tag{6.2}$$

   En este caso el orden de la ecuación puede ser reducido a $n - k$ mediante el cambio de variables $y^{(k)} = p$.

   En efecto, luego del cambio de variables, la ecuación (6.2) toma la forma

$$F(x, p, p', \ldots, p^{(n-k)}) = 0 \ . \tag{6.3}$$

   De esta ecuación se determina $p = p(x, c_1, c_2, \ldots, c_{n-k})$, e y se halla de la ecuación $y^{(k)} = p(x, c_1, c_2, \ldots, c_{n-k})$, integrando k veces. En particular, si la ecuación de segundo orden no contiene a y, la sustitución de variables $y' = p$ conduce a una ecuación de primer orden.

   Ejemplo Consideremos la siguiente ecuación

$$\frac{d^5y}{dx^5} - \frac{1}{x}\frac{d^4y}{dx^4} = 0 \ .$$

Haciendo $\dfrac{d^4y}{dx^4} = p$ obtenemos $\dfrac{dp}{dx} - \dfrac{1}{x}p = 0$; separando variables e integrando, tendremos: $\ln|p| = \ln|x| + \ln c$, o bien $p = cx$, $\dfrac{d^4y}{dx^4} = cx$, de donde

$$y = c_1x^5 + c_2x^3 + c_3x^2 + c_4x + c_5 \ .$$

Ejemplo Hallar la trayectoria de un cuerpo que cae sin velocidad inicial en la atmósfera, considerando la resistencia del aire proporcional al cuadrado de la velocidad.

La ecuación de movimiento tiene la forma

$$m\frac{d^2s}{dt^2} = mg - k\left(\frac{ds}{dt}\right)^2 \ ,$$

donde s es el espacio recorrido por el cuerpo; m, la masa del mismo; t, el tiempo. Para $t = 0$, se tiene $s = 0$ y $\dfrac{ds}{dt} = 0$.

La ecuación no contiene explícitamente a la función incógnita s; por lo tanto, se puede reducir el orden de la misma considerando $\dfrac{ds}{dt} = v$. Entonces la ecuación de movimiento toma la forma

$$m\frac{dv}{dt} = mg - kv^2 \ .$$

Separando variables e integrando, se obtiene

$$\frac{m\,dv}{mg - kv^2} = dt \ ; \quad t = m\int_0^v \frac{dv}{mg - kv^2} = \frac{1}{k\sqrt{g}}\mathrm{Arctanh}\frac{kv}{\sqrt{g}} \ ,$$

de donde $v = \dfrac{\sqrt{g}}{k}\tanh(k\sqrt{g}\,t)$; multiplicando por dt e integrando nuevamente, hallamos la trayectoria del movimiento:

$$s = \frac{1}{k^2}\ln\cosh(k\sqrt{g}\,t) \ .$$

2. La ecuación no contiene a la variable independiente:

$$F(y, y', y'', \ldots, y^{(n)}) = 0 \ .$$

En este caso el orden de la ecuación se puede reducir en una unidad por medio de la sustitución $y' = p$; además, p se considera nueva función desconocida de y, $p = p(y)$ y, por lo tanto, todas las derivadas $\dfrac{d^k}{dx^k}$ deben expresarse por medio de las derivadas de la nueva función desconocida $p(y)$ respecto a y:

$$\begin{aligned}
&\frac{dy}{dx} = p \ , \\
&\frac{d^2y}{dx^2} = \frac{dp}{dx} = \frac{dp}{dy}\frac{dy}{dx} = \frac{dp}{dy}\,p \ , \\
&\frac{d^3y}{dx^3} = \frac{d}{dx}\left(\frac{dp}{dy}\,p\right) = \frac{d}{dy}\left(\frac{dp}{dy}\,p\right)\frac{dy}{dx} = \frac{d^2p}{dy^2}\,p^2 + \left(\frac{dp}{dy}\right)^2 p \ .
\end{aligned}$$

y análogamente para las derivadas de orden superior. Además, es evidente que la derivada $\dfrac{d^k y}{dx^k}$ se expresa mediante las derivadas de p respecto a y de orden no superior a $k-1$, lo cual precisamente conduce a la disminución del orden en una unidad.

En particular, si la ecuación de segundo orden no contiene a la variable independiente, entonces la sustitución de variables señalada conduce a una ecuación de primer orden.

Ejemplo Integrar la ecuación del péndulo matemático $\ddot{x} + a^2 \operatorname{sen} x = 0$ con condiciones iniciales $x(0) = x_0$, $\dot{x}(0) = \dot{x}_0$.

Reducimos el orden, haciendo

$$\dot{x} = v \ , \quad \ddot{x} = v\frac{dv}{dx} \ , \quad v\,dv = -a^2 \operatorname{sen} x\,dx \ ,$$

$$\frac{v^2}{2} = a^2\,(\cos x - \cos x_0) \ , \quad v = \pm a\sqrt{2(\cos x - \cos x_0)} \ ,$$

$$\frac{dx}{dt} = \pm a\sqrt{2(\cos x - \cos x_0)} \ , \quad t = \pm\frac{1}{a\sqrt{2}} \int_{x_0}^{x} \frac{dx'}{\sqrt{\cos x' - \cos x_0}} \ .$$

La integral del segundo miembro no se resuelve en funciones elementales, pero se reduce fácilmente a funciones elípticas.

3. El primer miembro de la ecuación

$$F(x, y, y', \ldots, y^{(n)}) = 0 \ . \tag{6.4}$$

es la derivada de cierta expresión diferencial $\Phi(x, y, y', \ldots, y^{(n-1)})$ de orden $n-1$.

En este caso se halla fácilmente la llamada *primera integral*, o sea, una ecuación diferencial de orden $n-1$, que contiene una constante arbitraria, y que es equivalente a la ecuación dada de n-ésimo orden, con lo cual reducimos el orden de la ecuación en una unidad. Efectivamente, la ecuación (6.4) puede escribirse en la forma

$$\frac{d}{dx}\Phi(x, y, y', \ldots, y^{(n-1)}) = 0 \tag{6.5}$$

Si $y(x)$ es solución de la ecuación (6.5), entonces la derivada de $\Phi(x, y, y', \ldots, y^{(n-1)})$ es idénticamente nula. Por lo tanto, la función $\Phi(x, y, y', \ldots, y^{(n-1)})$ es igual a una constante, con lo que se obtiene la primera integral

$$\Phi(x, y, y', \ldots, y^{(n-1)}) = c$$

Ejemplo Consideremos la siguiente ecuación

$$yy'' + (y')^2 = 0 \ .$$

Esta ecuación se puede escribir en la forma $d(yy') = 0$, de donde $yy' = c_1$, o bien $y\,dy = c_1\,dx$. Por lo tanto, la integral general será $y^2 = c_1 x + c_2$.

A veces el primer miembro de la ecuación $F(x, y, y', \ldots, y^{(n)}) = 0$ se convierte en derivada de la diferencial $\Phi(x, y, y', \ldots, y^{(n-1)})$ de orden $n-1$ sólo después de multiplicarlo por un factor, $\mu(x, y, y', \ldots, y^{(n-1)})$.

Ejemplo Consideremos la siguiente ecuación:

$$yy'' - (y')^2 = 0 \ . \tag{6.6}$$

Multiplicando por el factor $\mu = \dfrac{1}{y^2}$, se obtiene $[yy'' - (y')^2]/y^2 = 0$, o bien $\dfrac{d}{dx}\left(\dfrac{y'}{y}\right) = 0$, de donde $\dfrac{y'}{y} = c_1$, ó $\dfrac{d}{dx} \ln |\, y \,| = c_1$. Por lo tanto, $\ln |\, y \,| = c_1 x + \ln c_2$, $c_2 > 0$, de donde $y = c_2 e^{c_1 x}$, $c_2 \neq 0$.

Observación: Al multiplicar por el factor $\mu(x, y, y', \ldots, y^{(n-1)})$ se pueden introducir soluciones superfluas, que reducen dicho factor a cero. Si μ, es discontinuo, pueden también perderse soluciones. En el ejemplo anterior, al multiplicar por $\mu = 1/y^2$ se perdió la solución $y = 0$; sin embargo, puede incluirse en la solución obtenida, si se considera que c_2 puede tomar el valor 0.

4. La ecuación $F(x, y, y', \ldots, y^{(n)}) = 0$ es homogénea con respecto a los argumentos $y, y', \ldots, y^{(n)}$.

   El orden de la ecuación homogénea respecto a $y, y', \ldots, y^{(n)}$

   $$F(x, y, y', \ldots, y^{(n)}) = 0 \tag{6.7}$$

   es decir, de la ecuación para la cual se cumple la identidad

   $$F(x, ky, ky', \ldots, ky^{(n)}) = k^p \, F(x, y, y', \ldots, y^{(n)}) \ ,$$

   puede ser reducido en una unidad por medio de la sustitución $y = e^{\int z \, dx}$, donde z es una nueva función desconocida. En efecto, derivando, se obtiene

   $$\begin{aligned} y' &= e^{\int z \, dx} \, z \ , \\ y'' &= e^{\int z \, dx} \, (z^2 + z') \ , \\ y''' &= e^{\int z \, dx} \, (z^3 + 3zz' + z'') \ , \\ &\vdots \\ y^{(k)} &= e^{\int z \, dx} \, \Phi(z, z', z'', \ldots, z^{(k-1)}) \ , \end{aligned}$$

   se puede comprobar la veracidad de esta igualdad mediante el método de inducción completa.

   Sustituyendo en (6.7) y observando que en base a la homogeneidad el factor $e^{p \int z \, dx}$ se puede sacar de la función F, reordenamos en función de las derivadas de z obteniendo

   $$e^{p \int z \, dx} \, f(x, z, z', \ldots, z^{(n-1)}) = 0 \ ,$$

o bien, dividiendo entre $e^{p\int z\,dx}$, tendremos para la nueva función f,

$$f(x, z, z', \ldots, z^{(n-1)}) = 0 \ .$$

Ejemplo Consideremos la ecuación:

$$yy'' - (y')^2 = 6xy^2 \ .$$

Haciendo $y = e^{\int z\,dx}$, obtenemos la ecuación para z, $z' = 6x$, con solución $z = 3x^2 + c_1$. Recuperando la función original $y = e^{\int (3x^2+c_1)\,dx}$, o bien $y = c_2 e^{(x^3+c_1 x)}$.

6.2.1. Ecuaciones de segundo orden y representación paramétrica.

En las aplicaciones se encuentran con particular frecuencia ecuaciones diferenciales de segundo orden que pueden reducir su orden.

1.
$$F(x, y'') = 0 \ . \tag{6.8}$$

En esta ecuación se puede disminuir el orden mediante la sustitución $y' = p$, y reducirla a la ecuación $F\left(x, \frac{dp}{dx}\right) = 0$.

La ecuación (6.8) se puede resolver con respecto al segundo argumento, $y'' = f(x)$, e integrar dos veces, o introducir un parámetro y sustituir la ecuación (6.8) por su representación paramétrica

$$\frac{d^2y}{dx^2} = \varphi(t) \ , \quad x = \psi(t) \ ,$$

de donde

$$dy' = y''\,dx = \varphi(t)\psi'(t)\,dt \ , \quad y' = \int \varphi(t)\psi'(t)\,dt + c_1 \ ,$$

$$dy = y'\,dx \ , \quad y = \int \left[\int \varphi(t)\psi'(t)\,dt \ + c_1\right] \psi'(t')\,dt' + c_2 \ .$$

2.
$$F(y', y'') = 0 \ , \tag{6.9}$$

la ecuación (6.9) en forma paramétrica:

$$y'_x = \varphi(t) \ , \quad y''_{xx} = \psi(t) \ ,$$

de donde

$$dx = \frac{dy'}{y''} = \frac{\varphi'(t)\,dt}{\psi(t)} \ , \quad x = \int \frac{\varphi'(t)\,dt}{\psi(t)} + c_1 \ ,$$

luego de lo cual y se determina por cuadratura:

$$dy = y'\,dx = \varphi(t)\frac{\varphi'(t)}{\psi(t)}\,dt \ , \quad y = \int \frac{\varphi(t)\varphi'(t)}{\psi(t)}\,dt + c_2 \ .$$

3.

$$F(y, y'') = 0 \ . \tag{6.10}$$

Se puede reducir el orden haciendo

$$\frac{dy}{dx} = p \quad \frac{d^2y}{dx^2} = \frac{dp}{dy}\frac{dy}{dx} = p\frac{dp}{dy} \ .$$

Si la ecuación (6.10) es posible resolver fácilmente con respecto al segundo argumento, $y'' = f(y)$, entonces, multiplicando esta ecuación por la igualdad $2y'\,dx = 2\,dy$, obtenemos $d(y')^2 = 2f(y)\,dy$, de donde

$$\frac{dy}{dx} = \pm\sqrt{2\int f(y)\,dy + c_1} \ , \qquad \pm\frac{dy}{\sqrt{2\int f(y)\,dy + c_1}} = dx \ ,$$

$$x + c_2 = \pm\int \frac{dy}{\sqrt{2\int f(y)\,dy + c_1}}$$

La ecuación (6.10) se puede sustituir por su representación paramétrica $y = \varphi(t)$, $y'' = \psi(t)$; entonces de $dy' = y''dx$ y de $dy = y'dx$ se obtiene $y'dy' = y''dy$, o bien

$$\frac{1}{2}d(y')^2 = \psi(t)\varphi'(t)\,dt \ ,$$

$$(y')^2 = 2\int \psi(t)\varphi'(t)\,dt + c_1 \ ,$$

$$y' = \pm\sqrt{2\int \psi(t)\varphi'(t)\,dt + c_1} \ ,$$

luego de lo cual, de $dy = y'dx$ se halla dx, y después x:

$$dx = \frac{dy}{y'} = \pm\frac{\varphi'(t)\,dt}{\sqrt{2\int \psi(t)\varphi'(t)\,dt + c_1}} \ ,$$

$$x = \pm\int \frac{\varphi'(t)\,dt}{\sqrt{2\int \psi(t)\varphi'(t)\,dt + c_1}} + c_2 \ .$$

Las ecuaciones anteriores e $y = \varphi(t)$ determinan en forma paramétrica a la familia de curvas integrales.

Ejemplo Consideremos la siguiente ecuación

$$y'' = 2y^3 \ , \quad y(0) = 1 \ , \quad y'(0) = 1 \ .$$

Multiplicando ambos miembros de esta ecuación por $2y'dx$, se obtiene $d(y')^2 = 4y^3\,dy$, de donde $(y')^2 = y^4 + c_1$. Teniendo en cuenta las condiciones iniciales, se halla que $c_1 = 0$ e $y' = y^2$. Por lo tanto, $\frac{dy}{y^2} = dx$, $-\frac{1}{y} = x + c_2$, $c_2 = -1$, $y = \frac{1}{1-x}$.

6.3. Ecuaciones diferenciales lineales de n-ésimo orden.

Se llama *ecuación diferencial lineal de n-ésimo orden* una ecuación lineal con respecto a la función desconocida y a sus derivadas, y que, por lo tanto, tiene la forma

$$a_0(x)y^{(n)} + a_1(x)y^{(n-1)} + \ldots + a_{n-1}(x)y' + a_n(x)y = \varphi(x) \ . \tag{6.11}$$

Si el segundo miembro $\varphi(x) = 0$, entonces la ecuación se llama lineal homogénea, puesto que es homogénea con respecto a la función desconocida y y a sus derivadas.

Si el coeficiente $a_0(x)$ es diferente de cero en todos los puntos de cierto intervalo $a \le x \le b$, entonces, dividiendo entre $a_0(x)$, reducimos la ecuación lineal homogénea (si x varía en dicho intervalo) a la forma:

$$y^{(n)} + p_1(x)y^{(n-1)} + \ldots + p_{n-1}(x)y' + p_n(x)y = 0 \ , \tag{6.12}$$

o bien

$$y^{(n)} = -\sum_{i=1}^{n} p_i(x)y^{(n-i)} \ . \tag{6.13}$$

Si los coeficientes $p(x)$ son continuos en el intervalo $a \le x \le b$, entonces en un entorno de condiciones iniciales arbitrarias

$$y(x_0) = y_0 \ , \quad y'(x_0) = y_0' \ , \quad y''(x_0) = y_0'' \ , \quad \ldots \quad y^{(n-1)}(x_0) = y_0^{(n-1)} \ ,$$

donde x_0 es cualquier punto del intervalo $a \le x \le b$, se satisfacen las condiciones del teorema de existencia y unicidad.

6.3.1. Conservación de la linealidad y la homogeneidad.

Obsérvese que la linealidad y la homogeneidad de la ecuación se conservan en cualquier transformación de la variable independiente $x = \varphi(t)$, donde $\varphi(t)$ es una función arbitraria derivable n veces, cuya derivada $\varphi(t) \neq 0$ en el segmento de variación de t considerado.

En efecto,

$$\begin{aligned} \frac{dy}{dx} &= \frac{dy}{dt}\frac{1}{\varphi'(t)} \ , \\ \frac{d^2y}{dx^2} &= \frac{d^2y}{dt^2}\frac{1}{[\varphi'(t)]^2} - \frac{dy}{dt}\frac{\varphi''}{[\varphi'(t)]^3} \ , \\ &\vdots \end{aligned}$$

La derivada de cualquier orden $\dfrac{d^ky}{dx^k}$ es función lineal homogénea de $\dfrac{dy}{dt}, \dfrac{d^2y}{dt^2}, \ldots, \dfrac{d^ky}{dt^k}$ y, por lo tanto, al sustituir en la ecuación (6.12) su linealidad y su homogeneidad se conservan.

La linealidad y la homogeneidad se conservan también al efectuarse una transformación lineal homogénea de la función desconocida: $y(x) = \alpha(x)z(x)$. En efecto, por la fórmula de derivada de un producto,

$$y^{(k)} = \alpha(x)z^{(k)} + k\alpha'(x)z^{(k-1)} + \frac{k(k-1)}{2!}\alpha''(x)z^{(k-2)} + \ldots + \alpha^{(k)}(x)z \ ,$$

es decir, la derivada $y^{(k)}$ es función lineal homogénea de z, z', z'',..., $z^{(k)}$. En consecuencia, el primer miembro de la ecuación lineal homogénea

$$a_0(x)y^{(n)} + a_1(x)y^{(n-1)} + \ldots + a_{n-1}(x)y' + a_n(x)y = 0 \ .$$

luego de sustituir las variables, será función lineal homogénea de z, z', z'', ..., $z^{(n)}$.

6.3.2. Operador diferencial lineal.

Escribamos la ecuación lineal homogénea

$$y^{(n)} + p_1(x)y^{(n-1)} + \ldots + p_{n-1}(x)y' + p_n(x)y = 0 \ ,$$

en forma compacta:

$$L[y] = 0 \ ,$$

donde

$$L[y] = y^{(n)} + p_1(x)y^{(n-1)} + \ldots + p_{n-1}(x)y' + p_n(x)y \ .$$

Llamaremos a $L[y]$ *operador diferencial lineal.*

El operador diferencial lineal posee las dos propiedades fundamentales siguientes:

1. Un factor constante puede sacarse del símbolo del operador:

$$L[cy] \equiv cL[y] \ .$$

En efecto,

$$\begin{aligned}(cy)^{(n)} + p_1(x)(cy)^{(n-1)} + \ldots + p_{n-1}(x)(cy)' + p_n(x)(cy) \\ = c[y^{(n)} + p_1(x)y^{(n-1)} + \ldots + p_{n-1}(x)y' + p_n(x)y] \ .\end{aligned}$$

2. El operador diferencial lineal, aplicado a la suma de dos funciones es igual a la suma de los resultados de la aplicación del mismo a cada función por separado:

$$L[y_1 + y_2] \equiv L[y_1] + L[y_2] \ .$$

en efecto,

$$\begin{aligned}(y_1 + y_2)^{(n)} + p_1(x)(y_1 + y_2)^{(n-1)} + \ldots + p_{n-1}(x)(y_1 + y_2)' + p_n(x)(y_1 + y_2) \equiv \\ [y_1^{(n)} + p_1(x)y_1^{(n-1)} + \ldots + p_{n-1}(x)y_1' + p_n(x)y_1] + \\ [y_2^{(n)} + p_1(x)y_2^{(n-1)} + \ldots + p_{n-1}(x)y_2' + p_n(x)y_2].\end{aligned}$$

Como consecuencia de las propiedades anteriores, resulta

$$L\left[\sum_{i=1}^{m} c_i y_i\right] \equiv \sum_{i=1}^{m} c_i L[y_i] \ ,$$

donde los c_i son constantes.

Basándonos en las propiedades del operador lineal L, se puede demostrar una serie de teoremas sobre las soluciones de la ecuación lineal homogénea.

Teorema 6.2 Si y_1 es solución de la ecuación lineal homogénea $L[y] = 0$, entonces cy_1, donde c es una constante arbitraria, también es solución de ésta.

Teorema 6.3 La suma $y_1 + y_2$ de dos soluciones y_1 e y_2 de la ecuación lineal homogénea $L[y] = 0$ es solución de dicha ecuación.

Corolario de los dos teoremas anteriores. La combinación lineal con coeficientes arbitrarios constantes $\sum_{i=1}^{m} c_i y_i$ de las soluciones y_1, y_2, ..., y_m de la ecuación lineal homogénea $L[y] = 0$ es solución de dicha ecuación.

Teorema 6.4 Si la ecuación lineal homogénea $L[y] = 0$ con coeficientes reales $p_i(x)$ tiene solución compleja $y(x) = u(x) + iv(x)$, entonces la parte real $u(x)$ de esta solución y su parte imaginaria $v(x)$ son, por separado, soluciones de dicha ecuación homogénea.

6.3.3. Dependencia lineal.

Las funciones y_1, y_2, ..., y_n se llaman *linealmente dependientes* en cierto intervalo de x, $a \leq x \leq b$, si existen constantes α_1, α_2, ..., α_n, en dicho intervalo tal que

$$\alpha_1 y_1 + \alpha_2 y_2 + \ldots + \alpha_n y_n \equiv 0 \ , \tag{6.14}$$

con, por lo menos, un $\alpha_i \neq 0$. Si la identidad (6.14) se verifica sólo para el caso en que $\alpha_1 = \alpha_2 =, \ldots, = \alpha_n = 0$, las funciones y_1, y_2, ..., y_n se llaman *linealmente independientes* en el intervalo $a \leq x \leq b$.

Teorema 6.5 Si las funciones y_1, y_2, ..., y_n son linealmente dependientes en el intervalo $a \leq x \leq b$, entonces en dicho intervalo el determinante

$$W(x) = W[y_1, y_2, \ldots, y_n] = \begin{vmatrix} y_1 & y_2 & \cdots & y_n \\ y_1' & y_2' & \cdots & y_n' \\ y_1'' & y_2'' & \cdots & y_n'' \\ \vdots & \vdots & & \vdots \\ y_1^{(n-1)} & y_2^{(n-1)} & \cdots & y_n^{(n-1)} \end{vmatrix}$$

llamado *wronskiano*, es idénticamente nulo.

Teorema 6.6 Si las funciones linealmente independientes y_1, y_2, ..., y_n son soluciones de la ecuación lineal homogénea.

$$y^{(n)} + p_1(x) y^{(n-1)} + \ldots + p_{n-1}(x) y' + p_n(x) y = 0 \ , \tag{6.15}$$

con coeficientes continuos $p_i(x)$ en el intervalo $a \leq x \leq b$, entonces el wronskiano

$$W(x) = \begin{vmatrix} y_1 & y_2 & \cdots & y_n \\ y_1' & y_2' & \cdots & y_n' \\ y_1'' & y_2'' & \cdots & y_n'' \\ \vdots & \vdots & & \vdots \\ y_1^{(n-1)} & y_2^{(n-1)} & \cdots & y_n^{(n-1)} \end{vmatrix}$$

es diferente de cero en todos los puntos del intervalo $a \leq x \leq b$.

Teorema 6.7 La combinación lineal $\sum_{i=1}^{n} c_i y_i$ con coeficientes constantes arbitrarios de n soluciones particulares linealmente independientes, y_i $(i = 1, 2, \ldots, n)$, en el intervalo $a \leq x \leq b$ es solución general, para $a \leq x \leq b$, de la ecuación lineal homogénea

$$y^{(n)} + p_1(x)y^{(n-1)} + \ldots + p_{n-1}(x)y' + p_n(x)y = 0 \ , \tag{6.15}$$

con coeficientes $p_i(x)$ continuos en dicho intervalo $(i = 1, 2, \ldots, n)$.

Corolario del teorema anterior. El número máximo de soluciones linealmente independientes de una ecuación lineal homogénea es igual a su orden.
Observación: Se llama *sistema fundamental de soluciones* de una ecuación lineal homogénea de n-ésimo orden al conjunto de soluciones particulares cualquieras linealmente independiente. Para cada ecuación lineal homogénea (6.15) existe un sistema fundamental de soluciones. Para la construcción de un sistema fundamental de soluciones, se dan n^2 cifras arbitrarias

$$y_i^{(k)}(x_0) \quad \{i = 1, 2, \ldots, n;\ k = 0, 1, \ldots, n-1\} \ ,$$

sometiendo su elección exclusivamente a la condición

$$\begin{vmatrix} y_1(x_0) & y_2(x_0) & \ldots & y_n(x_0) \\ y_1'(x_0) & y_2'(x_0) & \ldots & y_n'(x_0) \\ y_1''(x_0) & y_2''(x_0) & \ldots & y_n''(x_0) \\ \vdots & \vdots & & \vdots \\ y_1^{(n-1)}(x_0) & y_2^{(n-1)}(x_0) & \ldots & y_n^{(n-1)}(x_0) \end{vmatrix} \neq 0 \ ,$$

donde x_0 es un punto cualquiera del intervalo $a \leq x \leq b$. Entonces las soluciones $y_i(x)$, determinadas por los valores iniciales $y_i^{(k)}(x_0)$ con $\{i = 1, 2, \ldots, n;\ k = 0, 1, \ldots, n-1\}$, forman un sistema fundamental, puesto que su wronskiano $W(x)$ en el punto $x = x_0$ es diferente de cero y, por lo tanto, en virtud del teorema 6.5 y del teorema 6.6, las soluciones y_1, y_2, $\ldots$, y_n son linealmente independientes.

6.3.4. Reducción de orden.

Conociendo una solución particular no trivial, y_1, de la ecuación lineal homogénea

$$y^{(n)} + p_1(x)y^{(n-1)} + \ldots + p_{n-1}(x)y' + p_n(x)y = 0 \ , \tag{6.15}$$

se puede, por medio de la sustitución $y = y_1 \int u\, dx$, reducir su orden manteniendo la linealidad y la homogeneidad.

En efecto, la sustitución $y = y_1 \int u\, dx$ se puede reemplazar por dos sustituciones $y = y_1 z$ y $z' = u$. La transformación lineal homogénea

$$y = y_1 z \tag{6.16}$$

conserva la linealidad y la homogeneidad de la ecuación; por lo tanto, la ecuación (6.15) se reduce en este caso a la forma

$$a_0(x)\, z^{(n)} + a_1(x)\, z^{(n-1)} + \cdots + a_n(x)\, z = 0 \ . \tag{6.17}$$

Además, a la solución $y = y_1$ de la ecuación (6.15) le corresponde, en virtud de (6.16), la solución $z \equiv 1$, de la ecuación (6.17). Sustituyendo $z \equiv 1$ en la ecuación (6.17), se obtiene $a_n = 0$. Por consiguiente, la ecuación (6.17) tiene la forma

$$a_0(x)\, z^{(n)} + a_1(x)\, z^{(n-1)} + \cdots + a_{n-1}(x)\, z' = 0 \ ,$$

y la sustitución $z' = u$ reduce su orden en una unidad:

$$a_0(x)\, u^{(n-1)} + a_1(x)\, u^{(n-2)} + \cdots + a_{n-1}(x)\, u = 0 \ .$$

Obsérvese que la misma sustitución $y = y_1 \int u\, dx$, donde y_1 es solución de la ecuación $L[y] = 0$, reduce en una unidad el orden de la ecuación lineal no homogénea $L[y] = f(x)$, puesto que dicha sustitución no altera el segundo miembro de la ecuación.

Conociendo k soluciones linealmente independientes $y_1, \ldots, y_k$ en el intervalo $a \le x \le b$, de la ecuación lineal homogénea, se puede reducir su orden hasta $n-k$, en el mismo intervalo $a \le x \le b$.

6.3.5. Fórmulas de Ostrogradski-Liouville.

Lema. Dos ecuaciones de la forma

$$y^{(n)} + p_1(x)y^{(n-1)} + \ldots + p_{n-1}(x)y' + p_n(x)y = 0 \ , \tag{6.18}$$

$$y^{(n)} + q_1(x)y^{(n-1)} + \ldots + q_{n-1}(x)y' + q_n(x)y = 0 \ , \tag{6.19}$$

donde las funciones $p_i(x)$ y $q_i(x)$ $(i = 1, 2, \cdots, n)$ son continuas en el intervalo $a \le x \le b$, y poseen un sistema fundamental común de soluciones $y_1, y_2, \ldots, y_n$, coinciden, es decir, que $p_i(x) \equiv q_i(x)$ $(i = 1, 2, \cdots, n)$ en el intervalo $a \le x \le b$.

De este modo, el sistema fundamental de soluciones $y_1, y_2, \ldots, y_n$ determina por completo la ecuación lineal homogénea

$$y^{(n)} + p_1(x)y^{(n-1)} + \ldots + p_{n-1}(x)y' + p_n(x)y = 0 \tag{6.18}$$

y, por consiguiente, se puede plantear el problema de hallar la ecuación (6.18) que posea el sistema fundamental de soluciones

$$y_1, \ y_2, \ \ldots, \ y_n \ .$$

Como cualquier solución y de la ecuación buscada (6.18) debe ser linealmente dependiente de las soluciones $y_1, y_2, \ldots, y_n$, entonces el wronskiano $W[y_1, y_2, \ldots, y_n] = 0$. Escribamos esta ecuación en forma desarrollada

$$\begin{vmatrix} y_1 & y_2 & \cdots & y_n & y \\ y_1' & y_2' & \cdots & y_n' & y' \\ y_1'' & y_2'' & \cdots & y_n'' & y'' \\ \vdots & \vdots & & \vdots & \vdots \\ y_1^{(n)} & y_2^{(n)} & \cdots & y_n^{(n)} & y^{(n)} \end{vmatrix} = 0 \ ,$$

o bien, descomponiéndola por los elementos de la última columna,

$$W[y_1, y_2, \ldots, y_n]\, y^{(n)} - \begin{vmatrix} y_1 & y_2 & \ldots & y_n \\ y_1' & y_2' & \ldots & y_n' \\ \vdots & \vdots & & \vdots \\ y_1^{(n-2)} & y_2^{(n-2)} & \ldots & y_n^{(n-2)} \\ y_1^{(n)} & y_2^{(n)} & \ldots & y_n^{(n)} \end{vmatrix} y^{(n-1)} + \ldots = 0 \ . \tag{6.20}$$

La ecuación obtenida (6.20) es la ecuación lineal homogénea buscada, que posee el sistema dado de soluciones y_1, y_2,...,y_n. Dividiendo ambos miembros de la ecuación entre el wronskiano de la derivada de mayor grado, diferente de cero, la reducimos a la forma (6.18).

De aquí se deduce que, en particular,

$$p_1(x) = -\frac{\begin{vmatrix} y_1 & y_2 & \ldots & y_n \\ y_1' & y_2' & \ldots & y_n' \\ \vdots & \vdots & & \vdots \\ y_1^{(n-2)} & y_2^{(n-2)} & \ldots & y_n^{(n-2)} \\ y_1^{(n)} & y_2^{(n)} & \ldots & y_n^{(n)} \end{vmatrix}}{W[y_1, y_2, \ldots, y_n]} \ .$$

Obsérvese que el determinante

$$\begin{vmatrix} y_1 & y_2 & \ldots & y_n \\ y_1' & y_2' & \ldots & y_n' \\ \vdots & \vdots & & \vdots \\ y_1^{(n-2)} & y_2^{(n-2)} & \ldots & y_n^{(n-2)} \\ y_1^{(n)} & y_2^{(n)} & \ldots & y_n^{(n)} \end{vmatrix} , \tag{6.21}$$

es igual a la derivada del wronskiano $W[y_1, y_2, \ldots, y_n]$. En efecto, según la regla de derivación de un determinante, la derivada

$$\frac{d}{dx} \begin{vmatrix} y_1 & y_2 & \ldots & y_n \\ y_1' & y_2' & \ldots & y_n' \\ \vdots & \vdots & & \vdots \\ y_1^{(n-2)} & y_2^{(n-2)} & \ldots & y_n^{(n-2)} \\ y_1^{(n-1)} & y_2^{(n-1)} & \ldots & y_n^{(n-1)} \end{vmatrix} ,$$

es igual a la suma sobre i desde 1 hasta n de determinantes que se diferencian del wronskiano en que se han derivado los elementos de la i-ésima fila, y las filas restantes se dejan sin variación. En esta suma solamente el último determinante, para $i = n$, que coincide con el determinante (6.21) puede ser diferente de cero. Los restantes son iguales a cero, ya que sus filas i e $i + 1$ coinciden.

Por lo tanto, $p_1(x) = \dfrac{W'}{W}$, de donde, multiplicando por dx e integrando, se obtiene

$$\log |\, W \,| = -\int p_1(x)\, dx + \log c \ , \quad W = c\, e^{-\int p_1(x)\, dx} \ ,$$

o bien

$$W = c\, e^{-\int_{x_o}^{x} p_1(x')\,dx'} \ . \tag{6.22}$$

Para $x = x_o$ se obtiene $c = W(x_o)$, de donde

$$W = W(x_o)\, e^{-\int_{x_o}^{x} p_1(x')\,dx'} \ . \tag{6.23}$$

La fórmulas anteriores son conocidas como *fórmulas de Ostrogradski-Liouville*. Estas fórmulas pueden aplicarse a la integración de la ecuación lineal homogénea de segundo orden

$$y'' + p_1(x)\, y' + p_2(x)\, y = 0 \ , \tag{6.24}$$

si es conocida una solución no trivial y_1 de la misma. Según la fórmula (6.23), cualquier solución de la ecuación (6.24) debe ser también solución de la ecuación

$$\begin{vmatrix} y_1 & y \\ y_1' & y' \end{vmatrix} = c_1\, e^{-\int p_1(x)\,dx} \ ,$$

o bien

$$y_1 y' - y y_1' = c_1\, e^{-\int p_1(x)\,dx} \ .$$

Para integrar esta ecuación lineal de primer orden, lo más fácil es aplicar el método del factor integrante.

Multiplicando por $\mu = \dfrac{1}{y_1^2}$, se obtiene

$$\frac{d}{dx}\left(\frac{y}{y_1}\right) = \frac{c_1}{y_1^2}\, e^{-\int p_1(x)\,dx} \ ,$$

de donde

$$\frac{y}{y_1} = \int \frac{c_1 e^{-\int p_1(x')\,dx'}}{y_1^2}\, dx + c_2 \ ,$$

o bien

$$y = c_2 y_1 + c_1 y_1 \int \frac{e^{-\int p_1(x')\,dx'}}{y_1^2}\, dx \ .$$

6.4. Ecuaciones lineales homogéneas con coeficientes constantes.

Si en la ecuación lineal homogénea

$$a_0\, y^{(n)} + a_1\, y^{(n-1)} + \ldots + a_n y = 0 \ , \tag{6.25}$$

todos los coeficientes a_i son constantes, entonces sus soluciones particulares pueden ser halladas en la forma $y = e^{kx}$, donde k es una constante. En efecto sustituyendo en (6.25) $y = e^{kx}$ e $y^{(p)} = k^p e^{kx}$, con $p = 1, 2, \cdots, n$, tendremos:

$$a_0 k^n e^{kx} + a_1 k^{n-1} e^{kx} + \ldots + a_n e^{kx} = 0 \ .$$

Dividiendo entre el factor e^{kx}, diferente de cero, se obtiene la llamada ecuación característica

$$a_0k^n + a_1k^{n-1} + \ldots + a_n = 0 \ . \tag{6.26}$$

Esta ecuación de n-ésimo grado determina los valores de k para los cuales $y = e^{kx}$ es solución de la ecuación lineal homogénea inicial con coeficientes constantes (6.25). Si todas las raíces k_1, k_2, ..., k_n de la ecuación característica son diferentes, entonces de esta forma se hallan n soluciones linealmente independientes e^{k_1x}, e^{k_2x}, ..., e^{k_nx} de la ecuación (6.25). Por consiguiente,

$$y = c_1e^{k_1x} + c_2e^{k_2x} + \ldots + c_ne^{k_nx} \ ,$$

donde c_i son constantes arbitrarias, es solución general de la ecuación inicial (6.25). Este método de integración de las ecuaciones lineales con coeficientes constantes fue aplicado por primera vez por Euler.

Ejemplo Consideremos la ecuación

$$y'' - 3y' + 2y = 0 \ .$$

la ecuación característica tiene la forma $k^2 - 3k + 2 = 0$; sus raíces son $k_1 = 1$, $k_2 = 2$. Por lo tanto, la solución general de la ecuación inicial tiene la forma $y = c_1e^x + c_2e^{2x}$.

6.4.1. Raíces complejas.

Puesto que los coeficientes de la ecuación (6.25) se presuponen reales, las raíces complejas de la ecuación característica pueden aparecer sólo en pares conjugados. Las soluciones complejas $e^{(\alpha+i\beta)x}$ y $e^{(\alpha-i\beta)x}$, correspondientes al par de raíces complejas conjugadas

$$k_1 = \alpha + i\beta \quad \text{y} \quad k_2 = \alpha - i\beta \ ,$$

pueden ser sustituidas por dos soluciones reales: por las partes reales e imaginarias de una de las soluciones.

$$e^{(\alpha\pm i\beta)x} = e^{\alpha x}\left(\cos\beta x \pm i \operatorname{sen}\beta x\right) \ ,$$

De esta manera, al par de raíces complejas conjugadas $k_{1,2} = \alpha \pm i\beta$ le corresponden dos soluciones reales: $e^{\alpha x}\cos\beta x$ y $e^{\alpha x}\operatorname{sen}\beta x$.

6.4.2. Raíces múltiples.

Si entre las raíces de la ecuación característica hay raíces múltiples, entonces la cantidad de soluciones distintas del tipo e^{kx} es menor que n y, por lo tanto, las soluciones linealmente independientes que faltan deben ser buscadas de otra forma.

Se puede demostrar que si la ecuación característica tiene una raíz k_i de multiplicidad α_i, entonces no sólo e^{k_jx} será solución de la ecuación inicial, sino también xe^{k_jx}, $x^2e^{k_jx}$, ..., $x^{\alpha_i-1}e^{k_jx}$. Por lo tanto, la solución general de la ecuación (6.25) tiene la forma

$$y = \sum_{i=1}^{m}\left(c_{0i} + c_{1i}x + c_{2i}x^2 + \ldots + c_{\alpha_i-1,i}x^{\alpha_i-1}\right)e^{k_ix} \ ,$$

donde c_{si} son constantes arbitrarias.

Ejemplo Consideremos la ecuación

$$y''' - 3y'' + 3y' - y = 0 \ .$$

La ecuación característica $k^3 - 3k^2 + 3k - 1 = 0$, o bien $(k-1)^3 = 0$, posee la raíz triple $k_{1,2,3} = 1$. Por consiguiente, la solución general tiene la forma

$$y = (c_1 + c_2 x + c_3 x^2) e^x \ .$$

6.4.3. Raíces complejas con multiplicidad.

Si la ecuación característica tiene una raíz múltiple compleja $p + iq$, de multiplicidad α, entonces sus soluciones correspondientes

$$e^{(p+iq)x} = e^{px} \left(\cos qx + i \operatorname{sen} qx\right)$$

y, separando las partes real e imaginaria, obtenemos 2α soluciones reales:

$$\begin{aligned} &e^{px}\cos qx \ , \ xe^{px}\cos qx \ , \ x^2 e^{px}\cos qx \ , \ \ldots \ , \ x^{\alpha-1}e^{px}\cos qx \ , \\ &e^{px}\operatorname{sen} qx \ , \ xe^{px}\operatorname{sen} qx \ , \ x^2 e^{px}\operatorname{sen} qx \ , \ \ldots \ , \ x^{\alpha-1}e^{px}\operatorname{sen} qx \ . \end{aligned} \tag{6.27}$$

Tomando las partes reales e imaginarias de las soluciones correspondientes a la raíz conjugada $p - iq$ de la ecuación característica, no se obtienen nuevas soluciones linealmente independientes. De esta manera, al par de raíces complejas conjugadas $p \pm iq$ de multiplicidad α le corresponde 2α soluciones reales linealmente independientes (6.27).

6.5. Ecuación de Euler.

Las ecuaciones de la forma

$$a_0 x^n \, y^{(n)} + a_1 x^{n-1} \, y^{(n-1)} + \ldots + a_{n-1} x y' + a_n y = 0 \ , \tag{6.28}$$

donde todas las a_i son constantes, se llaman *ecuaciones de Euler*. La ecuación de Euler se reduce, mediante la sustitución de la variable independiente $x = e^t$, a una ecuación lineal homogénea con coeficientes constantes.

En efecto, como fue señalado anteriormente, la linealidad y la homogeneidad de la ecuación se conservan en la transformación de la variable independiente, y los coeficientes se vuelven constantes, puesto que

$$\begin{aligned} \frac{dy}{dx} &= \frac{dy}{dt} e^{-t} \ , \\ \frac{d^2y}{dx^2} &= e^{-2t}\left(\frac{d^2y}{dt^2} - \frac{dy}{dt}\right) \ , \\ &\vdots \\ \frac{d^k y}{dx^k} &= e^{-kt}\left(\beta_1 \frac{dy}{dt} + \beta_2 \frac{d^2y}{dt^2} + \ldots + \beta_k \frac{d^k y}{dt^k}\right) \ , \end{aligned} \tag{6.29}$$

donde todas las β_i son constantes, y al sustituir en la ecuación (6.28) los factores e^{-kt} se simplifican con los factores $x^k = e^{kt}$.

La validez de la igualdad (6.29) puede ser demostrada por el método de inducción. Por lo tanto, los productos

$$x^k \frac{d^k y}{dx^k} = \beta_1 \frac{dy}{dt} + \beta_2 \frac{d^2 y}{dt^2} + \ldots + \beta_k \frac{d^k y}{dt^k} \ ,$$

que entran en la forma lineal con coeficientes constantes en la ecuación de Euler

$$\sum_{k=0}^{n} a_{n-k} x^k \frac{d^k y}{dx^k} = 0 \ , \tag{6.30}$$

se expresan en forma lineal (y con coeficientes constantes) mediante las derivadas de la función y respecto a la nueva variable t. De aquí se deduce que la ecuación transformada será una ecuación lineal homogénea con coeficientes constantes:

$$b_0 \frac{d^n y}{dt^n} + b_1 \frac{d^{n-1} y}{dt^{n-1}} + \ldots + b_{n-1} \frac{dy}{dt} + b_n y = 0 \ . \tag{6.31}$$

En lugar de transformar la ecuación de Euler en una ecuación lineal con coeficientes constantes, cuyas soluciones particulares tienen la forma $y = e^{kt}$, se puede buscar directamente la solución de la ecuación inicial en la forma $y = x^k$, ya que

$$e^{kt} = x^k \ .$$

La ecuación obtenida después de simplificar por x^k

$$a_0 k(k-1)\ldots(k-n+1) + a_1 k(k-1)\ldots(k-n+2) + \ldots + a_n = 0 \ , \tag{6.32}$$

para la determinación de k, debe coincidir con la ecuación característica para la ecuación transformada (6.31). En consecuencia, a las raíces k_i de la ecuación (6.32), de multiplicidad α_i, les corresponden las soluciones

$$e^{k_i t} \ , \ t e^{k_i t} \ , \ t^2 e^{k_i t} \ , \ \ldots \ , \ t^{\alpha_i - 1} e^{k_i t} \ .$$

de la ecuación transformada, o bien las

$$x^{k_i} \ , \ x^{k_i} \log(n) \ , \ x^{k_i} \log^2(n) \ , \ \ldots \ , \ x^{k_i} \log^{\alpha_i - 1}(n) \ ,$$

de la ecuación inicial. A las raíces complejas conjugadas $p \pm iq$ de la ecuación (6.32) de multiplicidad α le corresponden las soluciones

$$\begin{aligned} &e^{pt}\cos qt \ , \ t e^{pt}\cos qt \ , \ \ldots \ , \ t^{\alpha-1} e^{pt}\cos qt \ , \\ &e^{pt}\operatorname{sen} qt \ , \ t e^{pt}\operatorname{sen} qt \ , \ \ldots \ , \ t^{\alpha-1} e^{pt}\operatorname{sen} qt \ , \end{aligned}$$

de la ecuación trasformada, o las

$$\begin{aligned} &x^p \cos(q\log x) \ , \ x^p \log x \cos(q\log x) \ , \ \ldots \ , x^p \log^{\alpha_i - 1} x \cos(q\log x) \ , \\ &x^p \operatorname{sen}(q\log x) \ , \ x^p \log x \operatorname{sen}(q\log x) \ , \ \ldots \ , x^p \log^{\alpha_i - 1} x \operatorname{sen}(q\log x) \ , \end{aligned}$$

de la ecuación inicial de Euler.

Ejemplo Consideremos la ecuación

$$x^2y'' - xy' + y = 0 \ .$$

Buscamos la solución en la forma $y = x^k$; $k(k-1) - k + 1 = 0$, o bien $(k-1)^2 = 0$, $k_{1,2} = 1$. Por consiguiente, la solución general para $x > 0$ será

$$y = (c_1 + c_2 \log x)x \ .$$

Las ecuaciones de la forma

$$a_0(ax+b)^n\, y^{(n)} + a_1(ax+b)^{n-1}\, y^{(n-1)} + \ldots + a_{n-1}(ax+b)y' + a_n y = 0 \ , \tag{6.33}$$

se denominan también *ecuaciones de Euler*, y se reducen a la ecuación (6.28) por medio de la sustitución de la variable independiente $(ax+b) = x_1$. Por lo tanto, las soluciones particulares de esta ecuación se pueden buscar en la forma $y = (ax+b)^k$, o transformar la ecuación (6.33) a una ecuación lineal homogénea con coeficientes constantes, mediante la sustitución de las variables $ax + b = e^t$.

6.6. Ecuaciones lineales no homogéneas.

La ecuación *lineal no homogénea* tiene la forma

$$a_0(x)\, y^{(n)} + a_1(x)\, y^{(n-1)} + \ldots + a_{n-1}(x)y' + a_n(x)y = \varphi(x) \ ,$$

Si $a_0(x) \neq 0$ en el intervalo considerado de variación de x, entonces dividiendo entre $a_0(x)$ se obtiene

$$y^{(n)} + p_1(x)y^{(n-1)} + \ldots + p_{n-1}(x)y' + p_n(x)y = f(x) \ . \tag{6.34}$$

Esta ecuación, conservando las notaciones anteriores, la escribimos en forma compacta:

$$L[y] = f(x) \ .$$

Si para $a \leq x \leq b$, en la ecuación (6.34) todos los coeficientes p_i y el segundo miembro $f(x)$ son continuos, entonces ella posee una solución única que satisface las condiciones

$$y^{(k)}(x_0) = y_0^{(k)} \ , \quad \text{con } k = 0, 1, \ldots, n-1 \ ,$$

donde $y_0^{(k)}$ son números reales cualesquiera, y x_0 un punto arbitrario del intervalo $a \leq x \leq b$.

De las dos propiedades fundamentales del operador lineal

$$L[cy] \equiv cL[y] \ ,$$
$$L[y_1 + y_2] \equiv L[y_1] + L[y_2] \ ,$$

donde c es una constante, se deduce directamente que:

(I) La suma $\tilde{y} + y_1$ de una solución $\tilde{y}$ de la ecuación no homogénea

$$L[y] = f(x) \tag{6.35}$$

y de una solución y_1 de la ecuación homogénea correspondiente $L[y] = 0$, es solución de la ecuación no homogénea (6.35).

(II) Si y_i es solución de la ecuación $L[y] = f_i(x)$ con $i = 1, 2, \ldots, m$ entonces $y = \sum_{i=1}^{m} \alpha_i y_i$ es solución de la ecuación

$$L[y] = \sum_{i=1}^{m} \alpha_i f_i(x) \ ,$$

donde las α_i son constantes.

Esta propiedad, denominada *principio de superposición*, conserva evidentemente su validez también para $m \to \infty$, si la serie $\sum_{i=1}^{\infty} \alpha_i y_i$ converge y puede ser derivada término a término n veces.

(III) Si la ecuación $L[y] = U(x) + iV(x)$, donde todos los coeficientes $p_i(x)$ y las funciones $U(x)$ y $V(x)$ son reales, tiene la solución $y = u(x) + iv(x)$, entonces la parte real $u(x)$ y la parte imaginaria $v(x)$ son respectivamente soluciones de las ecuaciones

$$L[y] = U(x) \quad \text{y} \quad L[y] = V(x) \ .$$

Teorema 6.8 La solución general en el intervalo $a \leq x \leq b$ de la ecuación $L[y] = f(x)$ con coeficientes p_i y la función del miembro derecho $f(x)$ continuos en dicho intervalo, es igual a la suma de la solución general $\sum_{i=1}^{n} c_i y_i$ de la ecuación homogénea correspondiente y de cualquier solución particular $\tilde{y}$ de la ecuación no homogénea.

Ejemplo Consideremos la ecuación

$$y'' + y = x \ ,$$

Una solución particular de esta ecuación $y = x$ es inmediata; la solución general de la ecuación homogénea correspondiente tiene la forma

$$y = c_1 \cos x + c_2 \operatorname{sen} x \ .$$

Por lo tanto, la solución general de la ecuación no homogénea inicial es

$$y = c_1 \cos x + c_2 \operatorname{sen} x + x \ .$$

Si la elección de una solución particular de la ecuación no homogénea es difícil, pero la solución general de la ecuación homogénea correspondiente $y = \sum_{i=1}^{n} c_i y_i$ ya fue hallada, entonces se puede integrar la ecuación lineal no homogénea por el método de variación de las constantes.

Al aplicar este método, la solución de la ecuación no homogénea se busca en la forma $y = \sum_{i=1}^{n} c_i(x) y_i$, o sea que en esencia, en lugar de la función incógnita y introducimos n

funciones desconocidas $c_i(x)$. Puesto que escogiendo las funciones $c_i(x)$ con $i = 1, 2, \ldots, n$ hay que satisfacer solamente una ecuación

$$y^{(n)} + p_1(x)y^{(n-1)} + \ldots + p_{n-1}(x)y' + p_n(x)y = f(x) \ , \tag{6.34}$$

se puede exigir que estas n funciones $c_i(x)$ satisfagan otras $n-1$ ecuaciones, las cuales se escogen de manera que las derivadas de la función $y = \sum_{i=1}^{n} c_i(x)y_i$ tengan en lo posible la misma forma que tiene cuando las c_i son constantes. Escojamos $c_i(x)$ de manera tal que la segunda suma de

$$y' = \sum_{i=1}^{n} c_i(x)y_i'(x) + \sum_{i=1}^{n} c_i'(x)y_i(x) \ ,$$

sea igual a cero,

$$\sum_{i=1}^{n} c_i'(x)y_i(x) = 0$$

y, por lo tanto,

$$y' = \sum_{i=1}^{n} c_i(x)y_i'(x) \ ,$$

es decir, que y' tiene la misma forma que cuando las c_i son constantes. De la misma manera, en la derivada segunda

$$y'' = \sum_{i=1}^{n} c_i(x)y_i''(x) + \sum_{i=1}^{n} c_i'(x)y_i'(x) \ ,$$

exigimos que la segunda suma sea igual a cero, con lo cual se somete $c_i(x)$ a la segunda condición

$$\sum_{i=1}^{n} c_i'(x)y_i'(x) = 0 \ .$$

Continuamos calculando las derivadas de la función $y = \sum_{i=1}^{n} c_i(x)y_i$ hasta el orden $n-1$ inclusive, e igualando cada vez a cero la suma $\sum_{i=1}^{n} c_i'(x)y_i^{(k)}(x)$:

$$\sum_{i=1}^{n} c_i'(x)y_i^{(k)}(x) = 0 \quad \text{para } k = 0, 1, 2, \ldots, n-2 \ , \tag{6.36}$$

obtenemos

$$\begin{aligned} y &= \sum_{i=1}^{n} c_i(x) y_i \ , \\ y' &= \sum_{i=1}^{n} c_i(x) y_i'(x) \ , \\ y'' &= \sum_{i=1}^{n} c_i(x) y_i''(x) \ , \\ &\vdots \\ y^{(n-1)} &= \sum_{i=1}^{n} c_i(x) y_i^{(n-1)}(x) \ , \\ y^{(n)} &= \sum_{i=1}^{n} c_i(x) y_i^{(n)}(x) + \sum_{i=1}^{n} c_i'(x) y_i^{(n-1)}(x) \ . \end{aligned} \tag{6.37}$$

En la última igualdad no podemos exigir que $\sum_{i=1}^{n} c_i'(x) y_i^{(n-1)}(x) = 0$, puesto que las funciones $c_i(x)$ ya están sometidas a las $n-1$ condiciones (6.36), y hay aún que satisfacer la ecuación inicial (6.34). Sustituyendo y, y', $\ldots$,$y^{(n)}$ de (6.37) en la ecuación

$$y^{(n)} + p_1(x) y^{(n-1)} + \ldots + p_{n-1}(x) y' + p_n(x) y = f(x) \ , \tag{6.34}$$

se obtiene la ecuación que falta para la determinación de $c_i(x)$ con $i = 1, 2, \ldots, n$. Es evidente que en el primer miembro de (6.34) queda sólo la suma $\sum_{i=1}^{n} c_i'(x) y_i^{(n-1)}(x)$, ya que todos los términos restantes tienen la misma forma que cuando las c_i son constantes, y cuando éstas son constantes, la función $y = \sum_{i=1}^{n} c_i(x) y_i$ satisface la ecuación homogénea correspondiente.

De esta manera, las funciones $c_i(x)$ con $i = 1, 2, \ldots, n$ se determinan del sistema de n ecuaciones lineales

$$\begin{aligned} \sum_{i=1}^{n} c_i'(x) y_i &= 0 \ , \\ \sum_{i=1}^{n} c_i'(x) y_i' &= 0 \ , \\ \sum_{i=1}^{n} c_i'(x) y_i'' &= 0 \ , \\ &\vdots \\ \sum_{i=1}^{n} c_i'(x) y_i^{(n-2)} &= 0 \ , \\ \sum_{i=1}^{n} c_i'(x) y_i^{(n-1)} &= f(x) \ . \end{aligned} \tag{6.38}$$

cuyo determinante es diferente de cero, debido a que éste

$$\begin{vmatrix} y_1 & y_2 & \dots & y_n \\ y_1' & y_2' & \dots & y_n' \\ \vdots & \vdots & & \vdots \\ y_1^{(n-2)} & y_2^{(n-2)} & \dots & y_n^{(n-2)} \\ y_1^{(n-1)} & y_2^{(n-1)} & \dots & y_n^{(n-1)} \end{vmatrix} ,$$

es el wronskiano de las soluciones linealmente independientes de la ecuación homogénea correspondiente. Al determinar de (6.38) todas las $c_i'(x) = \varphi_i(x)$ por cuadraturas, hallamos

$$c_i(x) = \int \varphi_i(x')\, dx' + \tilde{c}_i \ .$$

Ejemplo Consideremos la ecuación

$$y'' + y = \frac{1}{\cos x} \ .$$

La solución general de la ecuación homogénea correspondiente es $y = c_1 \cos(x) + c_2 \operatorname{sen} x$. Variemos c_1 y c_2:

$$y = c_1(x) \cos x + c_2(x) \operatorname{sen} x \ .$$

$c_1(x)$ y $c_2(x)$ se determinan del sistema (6.38):

$$\begin{aligned} c_1'(x) \cos x + c_2'(x) \operatorname{sen} x &= 0 \ , \\ -c_1'(x) \operatorname{sen} x + c_2'(x) \cos x &= \frac{1}{\cos x} \ , \end{aligned}$$

de donde

$$\begin{aligned} c_1'(x) = -\frac{\operatorname{sen} x}{\cos x} \ , \quad c_1(x) &= \log|\cos x| + \tilde{c}_1 \ ; \\ c_2'(x) = 1 \ , \quad c_2(x) &= x + \tilde{c}_2 \ . \end{aligned}$$

La solución general de la ecuación inicial es:

$$y = \tilde{c}_1 \cos x + \tilde{c}_2 \operatorname{sen} x + \cos x \log|\cos x| + x \operatorname{sen} x \ .$$

6.6.1. Reducción de orden.

De este modo, si se conocen n soluciones particulares linealmente independientes de la ecuación homogénea correspondiente, se puede, por el método de variación de constantes, integrar la ecuación no homogénea

$$L[y] = f(x) \ .$$

Si se conoce, en cambio, solamente k (donde $k < n$) soluciones linealmente independientes y_1, y_2,...,y_k de la ecuación homogénea correspondiente, entonces, como ya fue señalado, el

cambio de variables permite reducir su orden hasta $n-k$, conservando su linealidad. Obsérvese que si $k = n - 1$, el orden de la ecuación se reduce a 1, y la ecuación lineal de primer orden siempre se puede integrar en cuadraturas.

Análogamente se pueden utilizar k soluciones de la ecuación no homogénea $\tilde{y}_1$, $\tilde{y}_2$,...,$\tilde{y}_k$, puesto que sus diferencias son ya soluciones de la ecuación homogénea correspondiente. En efecto,

$$L[\tilde{y}_j] \equiv f(x) \ , \qquad L[\tilde{y}_p] \equiv f(x) \ ;$$

por lo tanto,

$$L[\tilde{y}_j - \tilde{y}_p] \equiv L[\tilde{y}_j] - L[\tilde{y}_p] \equiv f(x) - f(x) \equiv 0 \ .$$

Si las soluciones particulares de la ecuación homogénea correspondiente

$$(\tilde{y}_1 - \tilde{y}_k) \ , (\tilde{y}_2 - \tilde{y}_k) \ , \ \ldots \ , (\tilde{y}_{k-1} - \tilde{y}_k) \ , \tag{6.39}$$

son linealmente independientes, entonces el orden de la ecuación $L[y] = f(x)$ puede ser reducido hasta $n - (k - 1)$. Es evidente que las otras diferencias $\tilde{y}_j - \tilde{y}_k$ son combinaciones lineales de las soluciones (6.39):

$$\tilde{y}_j - \tilde{y}_p = (\tilde{y}_j - \tilde{y}_k) - (\tilde{y}_p - \tilde{y}_k) \ ,$$

y, por consiguiente, no pueden ser utilizadas para la reducción ulterior del orden.

6.6.2. Método de Cauchy.

Señalemos otro método, el *método de Cauchy*, para hallar la solución particular de la ecuación lineal no homogénea

$$L[y(x)] = f(x) \ . \tag{6.40}$$

En este método se supone conocida la solución $K(x, s)$, que depende de un parámetro, de la ecuación homogénea correspondiente $L[y(x)] = 0$, y que satisface las condiciones

$$K(s, s) = K'(s, s) = \ldots = K^{(n-2)}(s, s) = 0 \tag{6.41}$$

$$K^{(n-1)}(s, s) = 1 \ . \tag{6.42}$$

No es difícil comprobar que en este caso

$$y(x) = \int_{x_0}^{x} K(x, s) f(s) \, ds \ , \tag{6.43}$$

será solución particular de la ecuación (6.40), que satisface las condiciones iniciales nulas

$$y(x_0) = y'(x_0) = \ldots = y^{(n-1)}(x_0) = 0 \ .$$

En efecto, derivando[2] (6.43) y teniendo en cuenta las condiciones (6.41) y (6.42), se obtiene

$$\begin{aligned} y'(x) &= \int_{x_0}^{x} K'_x(x,s) f(s)\, ds \ , \\ y''(x) &= \int_{x_0}^{x} K''_x(x,s) f(s)\, ds \ , \\ &\vdots \\ y^{(n-1)}(x) &= \int_{x_0}^{x} K_x^{(n-1)}(x,s) f(s)\, ds \ , \\ y^{(n)}(x) &= \int_{x_0}^{x} K_x^{(n)}(x,s) f(s)\, ds + f(x) \ . \end{aligned} \tag{6.44}$$

El subíndice x en la función K indican que las derivadas son tomadas respecto a esa variable. Sustituyendo (6.43) y (6.44) en la ecuación (6.40), obtenemos

$$\int_{x_0}^{x} L[K(x,s)] f(s)\, ds + f(x) \equiv f(x) \ ,$$

y puesto que $K(x,s)$ es solución de la ecuación homogénea correspondiente tenemos que $L[K(x,s)] \equiv 0$ lo cual demuestra que la función $y(x) = \int_{x_0}^{x} K(x,s) f(s)\, ds$ es solución de $L[y(x)] = f(x)$.

La solución $K(x,s)$ puede ser tomada de la solución general $y = \sum_{i=1}^{n} c_i y_i$ de la ecuación homogénea, si se escogen las constantes arbitrarias c_i de manera que se cumplan las condiciones (6.41) y (6.42).

Ejemplo Para la ecuación

$$y'' + \alpha^2 y = f(x) \ , \tag{6.45}$$

la solución general es $y = c_1 \cos ax + c_2 \operatorname{sen} ax$. Las condiciones (6.41) y (6.42) conducen a las siguientes ecuaciones:

$$\begin{aligned} c_1 \cos as + c_2 \operatorname{sen} as &= 0 \ , \\ -ac_1 \operatorname{sen} as + ac_2 \cos as &= 1 \ . \end{aligned}$$

Por lo tanto,

$$c_1 = -\frac{\operatorname{sen} as}{a} \ , \qquad c_2 = -\frac{\cos as}{a} \ ,$$

y la solución buscada $K(x,s)$ tiene la forma

$$K(x,s) = \frac{1}{a} \operatorname{sen} a(x-s) \ .$$

[2]Debemos de considerar la regla de diferenciación de integrales:

$$\frac{d}{dx}\left[\int_{\phi_1(x)}^{\phi_2(x)} F(x,s)\, ds\right] = \int_{\phi_1(x)}^{\phi_2(x)} \frac{\partial F(x,s)}{\partial x}\, ds + F(\phi_1(x),x)\frac{d\phi_1}{dx} - F(\phi_2(x),x)\frac{d\phi_2}{dx} \ .$$

La solución de la ecuación (6.45) que satisface las condiciones iniciales nulas, según (6.43), se puede representar en la forma

$$y(x) = \frac{1}{a}\int_{x_0}^{x} \operatorname{sen} a(x' - s) f(s)\, ds \ .$$

6.6.3. Función de Green.

Se puede dar una interpretación física a la función $K(x, s)$ y a la solución de la ecuación lineal con segundo miembro en la forma (6.43). Aquí será más cómodo designar la variable independiente por la letra t.

En muchos problemas, la solución $y(t)$ de la ecuación

$$y^{(n)} + p_1(t)y^{(n-1)} + \ldots + p_n(t)y = f(t) \ , \tag{6.46}$$

describe el desplazamiento de cierto sistema, y la función $f(t)$ es la fuerza que actúa en este sistema; t es el tiempo.

Supongamos primeramente que, para $t < s$, el sistema se encuentra en estado de reposo, y que su desplazamiento se efectúa debido a la fuerza $f_\varepsilon(t)$, diferente de cero sólo en el intervalo $s < t < s + \varepsilon$, y cuyo impulso es igual a 1:

$$\int_{s}^{s+\varepsilon} f_\varepsilon(\tau)\, d\tau = 1 \ . \tag{6.47}$$

Designemos por $y_\varepsilon(t)$ la solución de la ecuación

$$y^{(n)} + p_1(t)y^{(n-1)} + \ldots + p_n(t)y = f_\varepsilon(t) \ .$$

Se comprueba fácilmente la existencia del límite $y_\varepsilon(t)$ cuando $\varepsilon \to 0$, el cual no depende de la función $f_\varepsilon(t)$, si suponemos que ésta no cambia su signo. En efecto,

$$y_\varepsilon(x) = \int_{t_0}^{t} K(t, s) f_\varepsilon(s)\, ds \ .$$

Aplicando el teorema del valor medio para $t > s + \varepsilon$, obtenemos

$$y_\varepsilon(x) = K(t, s + \varepsilon^*) \int_{s}^{s+\varepsilon} f_\varepsilon(\tau)\, d\tau = K(t, s + \varepsilon^*) \ ,$$

donde $0 < \varepsilon^* - \varepsilon$; por lo tanto,

$$\lim_{\varepsilon \to 0} y_\varepsilon(t) = K(t, s) \ .$$

Por ello, es natural llamar a la función $K(t, s)$ *función de influencia* del impulso instantáneo en el momento $t = s$. También se le conoce como la *función de Green* del sistema.

Dividiendo el intervalo (t_0, t) mediante los puntos s_i con $i = 0, 1, 2, \ldots, m$ en m partes iguales de longitud $\Delta s = (t - t_0)/m$, representamos la función $f(t)$ en (6.46) como una suma

de las funciones $f_i(t)$, donde $f_i(t)$ es diferente de cero sólo en el i-ésimo intervalo $s_{i-1} < t < s_i$. En éste, $f_i(t)$ coincide con la función $f(t)$:

$$f(t) = \sum_{i=1}^{m} f_i(t) \ .$$

Debido al principio de superposición, la solución de la ecuación (6.46) tiene la forma

$$y(t) = \sum_{i=1}^{m} y_i(t) \ ,$$

donde y_i son soluciones de la ecuación

$$y^{(n)} + p_1(t)y^{(n-1)} + \ldots + p_n(t)y = f_i(t) \ ,$$

con condiciones iniciales nulas. Si m es suficientemente grande, la solución $y_i(t)$ se puede considerar como función de influencia del impulso instantáneo de intensidad $f_i(s_i)\Delta s$. Por consiguiente,

$$y(t) \approx \sum_{i=1}^{m} K(t, s_i) f(s_i) \Delta s \ .$$

Pasando al límite cuando $m \to \infty$, se obtiene la solución de la ecuación (6.46) con condiciones iniciales nulas, en la forma

$$y(t) = \int_{t_0}^{t} K(t, s) f(s) \, ds \ ,$$

la cual demuestra que la influencia de la fuerza de acción continua se puede considerar como superposición de las influencias de impulsos separados.

6.7. Ecuaciones lineales no homogéneas con coeficientes constantes.

Al resolver ecuaciones lineales no homogéneas con coeficientes constantes, en muchos casos es posible escoger una solución particular sin dificultad, reduciendo así el problema a la integración de la ecuación homogénea correspondiente.

Supongamos, por ejemplo, que el segundo miembro es un polinomio de grado s y que , por lo tanto, la ecuación tiene la forma

$$a_0 \, y^{(n)} + a_1 \, y^{(n-1)} + \ldots + a_{n-1} y' + a_n y = A_0 \, x^s + A_1 \, x^{s-1} + \ldots + A_s \ , \tag{6.48}$$

donde todas las a_j y las A_i son constantes.

Si $a_n \neq 0$, entonces existe una solución particular de la ecuación (6.48) que tiene también la forma de polinomio de grado s. En efecto, sustituyendo

$$y = B_0 \, x^s + B_1 \, x^{s-1} + \ldots + B_s \ ,$$

en la ecuación (6.48) y comparando coeficientes de iguales potencias de x en ambos miembros, se obtiene un sistema de ecuaciones lineales para la determinación de los coeficientes B_i, que es siempre resoluble si $a_n \neq 0$:

$$a_n B_o = A_0 \ , \quad B_0 = \frac{A_0}{a_n} \ ,$$

$$a_n B_1 + s a_{n-1} B_0 = A_1 \ ,$$

de donde se determina B_1

$$a_n B_2 + (s-1) a_{n-1} B_1 + s(s-1) a_{n-2} B_0 = A_2 \ ,$$

de donde se determina B_2, y seguimos así hasta

$$a_n B_s + \ldots = A_s \ ,$$

de donde se determina B_s.

De esta manera, si $a_n \neq 0$ existe una solución particular que tiene la forma de polinomio cuyo grado es igual al grado del polinomio del segundo miembro.

Supongamos ahora que el coeficiente $a_n = 0$ y, para mayor generalidad, escogemos que también $a_{n-1} = a_{n-2} = \ldots = a_{n-\alpha+1} = 0$, pero $a_{n-\alpha} \neq 0$, o sea, que $k = 0$ es raíz de multiplicidad α de la ecuación característica; además, el caso $\alpha = 1$ no se excluye. Entonces, la ecuación (6.48) toma la forma

$$a_0 \, y^{(n)} + a_1 \, y^{(n-1)} + \ldots + a_{n-\alpha} y^{(\alpha)} = A_0 \, x^s + A_1 \, x^{s-1} + \ldots + A_s \ . \tag{6.49}$$

Haciendo $y^{(\alpha)} = z$, llegamos al caso anterior y, en consecuencia, hay una solución particular de la ecuación (6.49), para la cual

$$y^{(\alpha)} = B_0 \, x^s + B_1 \, x^{s-1} + \ldots + B_s \ .$$

Esto significa que y es un polinomio de grado $s + \alpha$; además, los términos de grado menor o igual a $\alpha - 1$ de dicho polinomio tendrán coeficientes constantes arbitrarios, que pueden ser, en particular, escogidos iguales a cero. Entonces, la solución particular de la forma:

$$y = x^\alpha \left(B_0 \, x^s + B_1 \, x^{s-1} + \ldots + B_s \right) \ .$$

Ejemplo Consideremos la ecuación

$$y'' + y = x^2 + x \ . \tag{6.50}$$

La solución particular tiene la forma

$$y = B_0 x^2 + B_1 x + B_2 \ .$$

Sustituyendo en la ecuación (6.50) e igualando los coeficientes de los términos de igual grado respecto a x, obtenemos

$$B_0 = 1 \ , \quad B_1 = 1 \ , \quad B_2 = -2 \ , \quad \tilde{y} = x^2 + x - 2 \ .$$

La solución general es

$$y = c_1 \cos x + c_2 \operatorname{sen} x + x^2 + x - 2 \ .$$

Consideremos ahora la ecuación lineal no homogénea de la forma

$$a_0\, y^{(n)} + a_1\, y^{(n-1)} + \ldots + a_n y = e^{px} \left(A_0\, x^s + A_1\, x^{s-1} + \ldots + A_s \right) \ , \tag{6.51}$$

donde todas las a_j las A_i son constantes. Como fue indicado anteriormente, el cambio de variables $y = e^{px} z$ reduce la ecuación (6.51) a la forma

$$e^{px} \left[b_0\, z^{(n)} + b_1\, z^{(n-1)} + \ldots + b_n z \right] = e^{px} \left(A_0\, x^s + A_1\, x^{s-1} + \ldots + A_s \right) \ ,$$

o bien

$$b_0\, z^{(n)} + b_1\, z^{(n-1)} + \ldots + b_n z = A_0\, x^s + A_1\, x^{s-1} + \ldots + A_s \ , \tag{6.52}$$

donde todas las b_i son constantes.

La solución particular de la ecuación (6.52), si $b_n \neq 0$ tiene la forma

$$\tilde{z} = \left(B_0\, x^s + B_1\, x^{s-1} + \ldots + B_s \right) \ ;$$

por lo tanto, la solución particular de la ecuación (6.51) será

$$\tilde{y} = e^{px} \left(B_0\, x^s + B_1\, x^{s-1} + \ldots + B_s \right) \ .$$

La condición $b_n \neq 0$ significa que $\tilde{k} = 0$ no es raíz de la ecuación característica

$$b_0\, \tilde{k}^n + b_1\, \tilde{k}^{n-1} + \ldots + b_n = 0 \ . \tag{6.53}$$

Por consiguiente, $k = p$ no es raíz de la ecuación característica

$$a_0\, k^n + a_1\, k^{n-1} + \ldots + a_n = 0 \ , \tag{6.54}$$

puesto que las raíces de estas ecuaciones están ligadas por la dependencia $k = \tilde{k} + p$.

Si $\tilde{k} = 0$, en cambio, es raíz de multiplicidad α de la ecuación característica (6.53) o, en otras palabras, $k = p$ es raíz de la misma multiplicidad α de la ecuación característica (6.54), entonces las soluciones particulares de las ecuaciones (6.52) y (6.51) tienen respectivamente las formas

$$\begin{aligned} \tilde{z} &= x^\alpha \left(B_0\, x^s + B_1\, x^{s-1} + \ldots + B_s \right) \ , \\ \tilde{y} &= x^\alpha e^{px} \left(B_0\, x^s + B_1\, x^{s-1} + \ldots + B_s \right) \ . \end{aligned}$$

De esta manera, si el segundo miembro de la ecuación diferencial lineal con coeficientes constantes tiene la forma

$$e^{px} \left(A_0\, x^s + A_1\, x^{s-1} + \ldots + A_s \right) \ ,$$

y si p no es raíz de la ecuación característica, la solución particular debe buscarse en la misma forma

$$\tilde{y} = e^{px} \left(B_0\, x^s + B_1\, x^{s-1} + \ldots + B_s \right) \ .$$

Si, en cambio, p es raíz de multiplicidad α de la ecuación característica (este caso se denomina *singular* o *resonante*), la solución particular debe ser buscada en la forma

$$\tilde{y} = x^{\alpha} e^{px} \left(B_0\, x^s + B_1\, x^{s-1} + \ldots + B_s \right) \ .$$

Ejemplo Consideremos la ecuación

$$y'' - y = e^{3x}(x-2) \ .$$

La solución particular debe ser buscada en la forma

$$\tilde{y} = x e^{3x}(B_0\, x + B_1) \ .$$

Obsérvese que nuestros razonamientos son válidos también si las p son complejas; por eso, si el segundo miembro de la ecuación diferencial lineal tiene la forma

$$e^{px} \left[P_s(x) \cos qx + Q_s(x) \operatorname{sen} qx \right] \ , \tag{6.55}$$

donde uno de los dos polinomios $P_s(x)$ o $Q_s(x)$ tiene grado s y el otro, no mayor que s, entonces reduciendo según las fórmulas de Euler las funciones trigonométricas a la forma exponencial, obtenemos en el segundo miembro

$$e^{(p+iq)x} R_s(x) + e^{(p-iq)x} T_s(x) \ , \tag{6.56}$$

donde R_s y T_s son polinomios de grado s.

A cada sumando del segundo miembro se le puede aplicar la regla anteriormente indicada, es decir, si $p \pm iq$ no son raíces de la ecuación característica, la solución particular se puede buscar en la misma forma que el segundo miembro (6.56); si, en cambio, $p \pm iq$ son raíces de multiplicidad α de la ecuación característica, la solución particular debe multiplicarse además por x^{α}.

Si volvemos a las funciones trigonométricas, esta regla se puede formular así:

a) Si $p \pm iq$ no son raíces de la ecuación característica, la solución particular debe buscarse en la forma

$$\tilde{y} = e^{px} \left[\tilde{P}_s(x) \cos(qx) + \tilde{Q}_s(x) \operatorname{sen}(qx) \right] \ ,$$

donde $\tilde{P}_s(x)$ y $\tilde{Q}_s(x)$ son polinomios de grado s con coeficientes indeterminados.

Obsérvese que si uno de los polinomios de grado $P_s(x)$ ó $Q_s(x)$ tienen un grado menor que s, e incluso, en particular, es idénticamente nulo, de todos modos ambos polinomios $\tilde{P}_s(x)$ y $\tilde{Q}_s(x)$ tendrán, en general, grado s.

b) Si $p \pm iq$ son raíces de multiplicidad α de la ecuación característica, la solución particular debe ser buscada en la forma

$$\tilde{y} = x^{\alpha} e^{px} \left[\tilde{P}_s(x) \cos(qx) + \tilde{Q}_s(x) \operatorname{sen}(qx) \right] \ ,$$

Ejemplo Consideremos la ecuación

$$y'' + 4y' + 4y = \cos(2x) \ .$$

Como los números $\pm 2i$ no son raíces de la ecuación característica, buscamos la solución particular en la forma

$$\tilde{y} = A\cos 2x + B\,\mathrm{sen}\,2x \ .$$

Ejemplo Consideremos la ecuación

$$y'' + 4y = \cos(2x) \ .$$

Como los números $\pm 2i$ son raíces simples de la ecuación característica, buscamos la solución particular en la forma

$$\tilde{y} = x\left[A\cos 2x + B\,\mathrm{sen}\,2x\right] \ .$$

Ejemplo Consideremos la ecuación

$$y'''' + 2y'' + y = \mathrm{sen}(x) \ .$$

Puesto que los números $\pm i$ son raíces dobles de la ecuación característica, la solución particular se busca en la forma

$$\tilde{y} = x^2\left[A\cos x + B\,\mathrm{sen}\,x\right] \ .$$

Ejemplo Consideremos la ecuación

$$y'' + 2y'' + 2y = e^{-x}\left(x\cos x + 3\,\mathrm{sen}\,x\right) \ .$$

Debido a que los números $-1 \pm i$ son raíces simples de la ecuación característica, la solución particular debe buscarse en la forma

$$\tilde{y} = xe^{-x}\left[(A_0x + A_1)\cos x + (B_0x + B_1)\,\mathrm{sen}\,x\right] \ .$$

En muchos casos, al buscar soluciones particulares de ecuaciones lineales con coeficientes constantes con segundos miembros de la forma (6.55), es conveniente pasar a las funciones exponenciales.

Por ejemplo, en la ecuación

$$y'' - 2y' + y = \cos x \ ,$$

se puede transformar $\cos x$ por la fórmula de Euler, o de modo más sencillo, considerar la ecuación

$$y'' - 2y' + y = e^{ix} \ , \tag{6.57}$$

la parte real de cuya solución debe satisfacer la ecuación original.

La solución particular de la ecuación (6.57) se puede buscar en la forma

$$y = Ae^{ix} \ .$$

Entonces

$$A = \frac{i}{2} \ , \qquad y = \frac{i}{2}\left(\cos x + i \operatorname{sen} x\right) \ .$$

La solución particular de la ecuación original es

$$\tilde{y_1} = \operatorname{Re} y = -\frac{1}{2} \operatorname{sen} x \ .$$

6.8. Integración de las ecuaciones diferenciales por medio de series.

El problema de la integración de ecuaciones lineales homogéneas de n-ésimo orden

$$p_0(x)y^{(n)} + p_1(x)y^{(n-1)} + \ldots + p_n(x)y = 0 \ , \tag{6.58}$$

se reduce a elegir n, o por lo menos $n-1$ soluciones linealmente independientes. Sin embargo, las soluciones particulares se escogen con facilidad sólo en casos excepcionales. En casos más complejos las soluciones particulares son buscadas en forma de suma de una serie $\sum_{i=1}^{\infty} a_i\varphi_i(x)$, sobre todo en forma de suma de una serie de potencias o de una serie generalizada de potencias.

Las condiciones bajo las cuales existen soluciones en forma de suma de una serie de potencia o de una serie generalizada de potencias, se establecen comúnmente por métodos de la teoría de funciones de variables complejas, la que suponemos desconocida por el lector. Los teoremas fundamentales se darán sin demostración y aplicados a las ecuaciones de segundo orden, las cuales se encuentran con mayor frecuencia en la práctica.

Teorema 6.9 Sobre la propiedad analítica de la solución
Si p_0, $p_1(x)$ y $p_2(x)$ son funciones analíticas de x en un entorno del punto $x = x_0$ y $p_0(x_0) \neq 0$, entonces las soluciones de la ecuación

$$p_0(x)y'' + p_1(x)y' + p_2(x)y = 0 \tag{6.59}$$

son también funciones analíticas en cierto entorno del mismo punto; por lo tanto, la solución de la ecuación (6.59) se puede buscar de la forma

$$y = a_0 + a_1(x - x_0) + a_2(x - x_0)^2 + \ldots + a_n(x - x_n)^n + \ldots \ .$$

Teorema 6.10 Sobre el desarrollo de la solución en una serie generalizada de potencias.
Si la ecuación (6.59) satisface las condiciones del teorema anterior, pero $x = x_0$ es un cero de orden finito s de la función $p_0(x)$, cero de orden $s-1$ o superior de la función $p_1(x)$ $(s > 1)$ y cero de orden no inferior $s-2$ del coeficiente $p_2(x)$ (si $s > 2$), entonces existe por lo menos

una solución no trivial de la ecuación (6.59) en forma de suma de una serie generalizada de potencias

$$y = a_0(x - x_0)^k + a_1(x - x_0)^{k+1} + a_2(x - x_0)^{k+2} + \ldots + a_n(x - x_n)^{k+n} + \ldots \ . \qquad (6.60)$$

donde k es un número real que puede ser entero y fraccionario, positivo o negativo.

La segunda solución linealmente independiente de (6.60) por regla general, tiene también la forma de suma de una serie generalizada de potencias, pero a veces puede contener además el producto de una serie generalizada de potencia por $\log(x - x_0)$.

En problemas concretos se puede proceder sin los dos teoremas formulados más arriba, sobre todo porque en el enunciado de éstos no se establecen las regiones de convergencia de las series consideradas. Con mayor frecuencia en problemas concretos se escoge una serie de potencias o una serie generalizada de potencias que satisfaga formalmente la ecuación diferencial, o sea, que al sustituirla en la ecuación considerada de orden n (6.58) la transforme en una identidad, si suponemos la convergencia de la serie y la posibilidad de su derivación término a término n veces. Al obtener formalmente la solución en forma de serie, se investiga su convergencia y la posibilidad de su derivación término a término n veces. En la región donde la serie converge y permite su derivación término a término n veces, la misma no solamente satisface formalmente la ecuación, sino que su suma es en realidad la solución buscada.

Ejemplo Consideremos la ecuación

$$y'' - xy = 0 \ . \qquad (6.61)$$

Busquemos la solución en forma de una serie de potencias

$$y = \sum_{n=0}^{\infty} a_n x^n \ .$$

Basándonos en los teorema anteriores, o derivando esta serie formalmente término a término dos veces y sustituyendo en la ecuación (6.61), obtenemos

$$\sum_{n=0}^{\infty} a_n n(n-1)x^{n-2} - x\sum_{n=0}^{\infty} a_n x^n \equiv 0 \ .$$

Igualando los coeficientes de iguales potencias de x en ambos miembros de la identidad, obtenemos $a_2 = 0$, $3 \cdot 2a_3 - a_0 = 0$, de donde $a_3 = a_0/(2 \cdot 3)$; $4 \cdot 3a_4 - a_1 = 0$, de donde $a_4 = a_1/(3{\cdot}4)$; $5{\cdot}4a_5 - a_2 = 0$, de donde $a_5 = a_2/(4{\cdot}5), \ldots$ y en forma general $n(n-1)a_n - a_{n-3} = 0$, de donde $a_n = a_{n-3}/(n-1)n, \ldots$. Por consiguiente,

$$a_{3n-1} = 0 \ , \qquad a_{3n} = \frac{a_0}{2 \cdot 3 \cdot 5 \cdot 6 \ldots (3n-1)3n} \ ,$$

$$a_{3n+1} = \frac{a_1}{3 \cdot 4 \cdot 6 \cdot 7 \ldots 3n(3n+1)} \ , \qquad \text{con } n = 1, 2, 3, \ldots,$$

a_0 y a_1 permanecen arbitrarios. De esta manera,

$$y = a_0\left[1+\frac{x^3}{2\cdot 3}+\frac{x^6}{2\cdot 3\cdot 5\cdot 6}+\ldots+\frac{x^{3n}}{2\cdot 3\cdot 5\cdot 6\ldots(3n-1)3n}+\ldots\right]+ \\ +a_1\left[x+\frac{x^4}{3\cdot 4}+\frac{x^7}{3\cdot 4\cdot 6\cdot 7}+\ldots+\frac{x^{3n+1}}{3\cdot 4\cdot 6\cdot 7\ldots 3n(3n+1)}+\ldots\right]. \tag{6.62}$$

El radio de convergencia de esta serie de potencias es infinito. Por consiguiente, la suma de la serie (6.62) para valores cualesquiera de x es solución de la ecuación considerada.

Capítulo 7

Sistemas de ecuaciones diferenciales.

versión final 2.1-021107[1]

7.1. Conceptos generales.

La ecuación de movimiento de una partícula de masa m bajo la acción de la fuerza $\vec{F}(t,\vec{r},\dot{\vec{r}})$, es

$$m\frac{d^2\vec{r}}{dt^2} = \vec{F}(t,\vec{r},\dot{\vec{r}}) \ ;$$

proyectando sobre los ejes de coordenadas, ésta puede ser sustituida por un sistema de tres ecuaciones escalares de segundo orden:

$$\begin{aligned} m\frac{d^2x}{dt^2} &= F_x(t,x,y,z,\dot{x},\dot{y},\dot{z}) \ , \\ m\frac{d^2y}{dt^2} &= F_y(t,x,y,z,\dot{x},\dot{y},\dot{z}) \ , \\ m\frac{d^2z}{dt^2} &= F_z(t,x,y,z,\dot{x},\dot{y},\dot{z}) \ , \end{aligned}$$

o por un sistema de seis ecuaciones de primer orden; si consideramos como funciones desconocidas no sólo las coordenadas x, y, z de la partícula, sino también las proyecciones $\dot{x}$, $\dot{y}$, $\dot{z}$ de su velocidad

$$\begin{aligned} \dot{x} &= u \ , \\ \dot{y} &= v \ , \\ \dot{z} &= w \ ; \\ m\dot{u} &= F_x(t,x,y,z,u,v,w) \ , \\ m\dot{v} &= F_y(t,x,y,z,u,v,w) \ , \\ m\dot{w} &= F_z(t,x,y,z,u,v,w) \ . \end{aligned}$$

En este caso, por lo general, se dan la posición inicial del punto $x(t_0) = x_0$, $y(t_0) = y_0$, $z(t_0) = z_0$, y la velocidad inicial $u(t_0) = u_0$, $v(t_0) = v_0$, $w(t_0) = w_0$.

[1]Este capítulo está basado en el tercer capítulo del libro: *Ecuaciones diferenciales y cálculo variacional* de L. Elsgoltz, editorial MIR

Se puede demostrar un teorema sobre existencia y unicidad de la solución del sistema de ecuaciones diferenciales

$$\begin{aligned} \frac{dx_1}{dt} &= f_1(t, x_1, x_2, \ldots, x_n) \ , \\ \frac{dx_2}{dt} &= f_2(t, x_1, x_2, \ldots, x_n) \ , \\ &\vdots \\ \frac{dx_n}{dt} &= f_n(t, x_1, x_2, \ldots, x_n) \ , \end{aligned} \tag{7.1}$$

que satisfacen las condiciones iniciales

$$x_i(t_0) = x_{i0} \ , \quad (i = 1, 2, \ldots, n) \ . \tag{7.2}$$

Enumeremos las condiciones suficientes para la existencia y unicidad de la solución del sistema (7.1) con las condiciones iniciales (7.2):

i) Continuidad de todas las funciones f_i en un entorno de las condiciones iniciales.

ii) Cumplimiento de la condición de Lipschitz para todas las funciones f_i en todos sus argumentos, a partir del segundo, en dicho entorno.

La segunda condición se puede cambiar por una más grosera: la existencia de las derivadas parciales

$$\frac{\partial f_i}{\partial x_j} \ , \quad (i = 1, 2, \ldots, n) \ ,$$

acotadas en valor absoluto.

La solución $x_1(t)$, $x_2(t)$, ..., $x_n(t)$ del sistema de ecuaciones diferenciales es una función vectorial n-dimensional, que denotaremos abreviadamente por $X(t)$. Con esta notación, el sistema (7.1) se puede escribir en la forma

$$\frac{dX}{dt} = F(t, X) \ ,$$

donde F es una función vectorial con coordenadas $(f_1, f_2, \ldots, f_n)$ y las condiciones iniciales, en la forma $X(t_0) = X_0$, donde X_0 es un vector de n-dimensiones, con coordenadas $(x_{10}, x_{20}, \ldots, x_{n0})$. La solución

$$x_1 = x_1(t) \ , \quad x_2 = x_2(t) \ , \quad \ldots \quad x_n = x_n(t) \ ,$$

o, más compactamente, $X = X(t)$, del sistema de ecuaciones, determina en el espacio euclidiano de coordenadas t, x_1, x_2, ...,x_n, cierta curva llamada *curva integral*. Cuando se cumplen las condiciones i) y ii) del teorema de existencia y unicidad, por cada punto de dicho espacio pasa una sola curva integral, y el conjunto de éstas forma una familia dependiente de n parámetros. Como parámetros de esta familia se pueden tomar, por ejemplo, los valores iniciales x_{10}, x_{20}, ...,x_{n0}.

Se puede dar otra interpretación de las soluciones

$$x_1 = x_1(t) \ , \quad x_2 = x_2(t) \ , \quad \dots \ , x_n = x_n(t) \ ,$$

o, en la forma más compacta, $X = X(t)$, que es particularmente cómoda si los segundos miembros del sistema (7.1) no dependen explícitamente del tiempo.

En el espacio euclidiano con coordenadas rectangulares x_1, x_2, ...,x_n, la solución $x_1 = x_1(t)$, $x_2 = x_2(t)$, ..., $x_n = x_n(t)$ determina la ley de movimiento por cierta trayectoria según la variación del parámetro t, el cual en esta interpretación se considera como el tiempo. Así, la derivada dX/dt será la velocidad del movimiento de la partícula, y dx_1/dt, dx_2/dt, ..., dx_n/dt, las coordenadas de la velocidad en ese punto. En esta interpretación, muy natural y cómoda en ciertos problemas físicos y mecánicos, el sistema

$$\frac{dx_i}{dt} = f_i(x_1, x_2, \dots, x_n) \ , \qquad (i = 1, 2, \dots, n) \ , \tag{7.1}$$

o bien

$$\frac{dX}{dt} = F(X) \ ,$$

se llama generalmente *dinámico*; el espacio de coordenadas x_1, x_2, ...,x_n, *espacio de fase*, y la curva $X = X(t)$, *trayectoria de fases*.

El sistema dinámico (7.1) determina en un momento dado t en el espacio x_1, x_2, ...,x_n un campo de velocidades. Si la función vectorial F depende explícitamente de t, entonces el campo de velocidades cambia con el tiempo, y las trayectorias de fases pueden intersectarse. Si la función vectorial F o, lo que es lo mismo, todas las funciones f_i no dependen explícitamente de t, el campo de velocidades es estacionario, es decir, no cambia en el tiempo, y el movimiento será permanente.

En este último caso, si las condiciones del teorema de existencia y unicidad se cumplen, por cada punto del espacio de fase $(x_1, x_2, \dots, x_n)$ pasará una sola trayectoria. En efecto, en este caso por cada trayectoria $X = X(t)$ se realizan infinitos movimiento diferentes, $X = X(t + c)$, donde c es una constante arbitraria. Es fácil comprobar esto realizando la sustitución de variables $t_1 = t + c$, con lo cual el sistema dinámico no cambia su forma:

$$\frac{dX}{dt_1} = F(X) \ ,$$

Por consiguiente, $X = X(t_1)$ será su solución que, expresada en las variables anteriores, será $X = X(t + c)$.

Si por un punto X_0 del espacio de fase, en el caso considerado, pasaran dos trayectorias

$$X = X_1(t) \quad \text{y} \quad X = X_2(t) \ , \quad X_1(\tilde{t}_0) = X_2(\tilde{\tilde{t}}_0) = X_0 \ ,$$

entonces, tomando en cada una de ellas el movimiento para el cual el punto X_0 se alcanza en el momento $t = t_0$, o sea, considerando las soluciones

$$X = X_1(t - t_0 + \tilde{t}_0) \quad \text{y} \quad X = X_2(t - t_0 + \tilde{\tilde{t}}_0) \ ,$$

se obtendría una contradicción con el teorema de existencia y unicidad, puesto que las dos soluciones diferentes $X_1(t - t_0 + \tilde{t}_0)$ y $X_2(t - t_0 + \tilde{\tilde{t}}_0)$ satisfacerían a la misma condición inicial $X(t_0) = X_0$.

7.2. Integración de un sistema de ecuaciones diferenciales por reducción a una sola ecuación de mayor orden.

Uno de los métodos fundamentales de integración de sistemas de ecuaciones diferenciales consiste en lo siguiente: de las ecuaciones del sistema (7.1) y de las ecuaciones obtenidas derivando éstas, se excluyen todas las funciones desconocidas, excepto una, para cuya determinación se obtiene una ecuación diferencial de orden mayor. Integrando dicha ecuación, se halla una de las funciones desconocidas. Las funciones desconocidas restantes se determinan, en lo posible sin integración partiendo de las ecuaciones originales y de las obtenidas por derivación. Veamos algunos ejemplos:

Ejemplo Consideremos el sistema de ecuaciones

$$\frac{dx}{dt} = y \ , \qquad \frac{dy}{dt} = x \ .$$

Derivemos una de las ecuaciones, por ejemplo, la primera, $\dfrac{d^2x}{dt^2} = \dfrac{dy}{dt}$; eliminando $\dfrac{dy}{dt}$ mediante la segunda ecuación, se obtiene $\dfrac{d^2x}{dt^2} - x = 0$, de donde $x = c_1e^t + c_2e^{-t}$. Utilizando la primera ecuación, obtenemos $y = \dfrac{dx}{dt} = c_1e^t - c_2e^{-t}$.

Hemos determinado y sin integrar, mediante la primera ecuación. Si hubiéramos determinado y de la segunda ecuación,

$$\frac{dy}{dt} = x = c_1e^t + c_2e^{-t} \ , \quad y = c_1e^t - c_2e^{-t} + c_3 \ ,$$

entonces habríamos introducido soluciones superfluas, puesto que la sustitución directa en el sistema original muestra que las funciones $x = c_1e^t + c_2e^{-t}$ e $y = c_1e^t - c_2e^{-t} + c_3$ satisfacen al sistema, no para c_3 cualesquiera, sino para $c_3 = 0$.

Ejemplo Consideremos el sistema de ecuaciones

$$\frac{dx}{dt} = 3x - 2y \ , \tag{7.3a}$$

$$\frac{dy}{dt} = 2x - y \ . \tag{7.3b}$$

Derivemos la segunda ecuación:

$$\frac{d^2y}{dt^2} = 2\frac{dx}{dt} - \frac{dy}{dt} \ . \tag{7.4}$$

De las ecuaciones (7.3b) y (7.4) se determina x y $\dfrac{dx}{dt}$:

$$x = \frac{1}{2}\left(\frac{dy}{dt} + y\right) \ , \tag{7.5}$$

$$\frac{dx}{dt} = \frac{1}{2}\left(\frac{d^2y}{dt^2} + \frac{dy}{dt}\right) \ .$$

Sustituyendo en (7.3a), obtenemos

$$\frac{d^2y}{dt^2} - 2\frac{dy}{dt} + y = 0 \ .$$

Integrando la ecuación lineal homogénea con coeficientes constantes $y = e^t(c_1 + c_2t)$ y sustituyendo en (7.5) se halla $x(t)$:

$$x = \frac{1}{2}e^t(2c_1 + c_2 + 2c_2t) \ .$$

Si aplicamos el proceso de eliminación de funciones desconocidas al sistema

$$\frac{dx_i}{dt} = \sum_{j=1}^{n} a_{ij}(t)x_j \ , \quad (i = 1, 2, \ldots, n) \ ,$$

llamado lineal homogéneo, entonces, como es fácil comprobar, la ecuación de n-ésimo orden

$$\frac{d^nx_1}{dt^n} = \phi\left(t, x_1, \frac{dx_1}{dt}, \ldots, \frac{d^{n-1}x_1}{dt^{n-1}}\right) \ , \tag{7.6}$$

también será lineal homogénea. Además, si todos los coeficientes a_{ij} son constantes, la ecuación (7.6) será también lineal homogénea con coeficientes constantes. Una observación análoga se cumple también para el sistema lineal no homogéneo

$$\frac{dx_i}{dt} = \sum_{j=1}^{n} a_{ij}(t)x_j + f_i(t) \ , \quad (i = 1, 2, \ldots, n) \ ,$$

para el cual la ecuación (7.6) será una ecuación lineal no homogénea de n-ésimo orden.

7.3. Sistema de ecuaciones diferenciales lineales.

Un sistema de ecuaciones diferenciales se llama *lineal*, si es lineal respecto a todas las funciones desconocidas y a sus derivadas. El sistema de n ecuaciones lineales de primer orden, escrito en la forma normal, es

$$\frac{dx_i}{dt} = \sum_{j=1}^{n} a_{ij}(t)x_j + f_i(t) \ , \quad (i = 1, 2, \ldots, n) \ , \tag{7.7}$$

o, en forma vectorial,

$$\frac{dX}{dt} = AX + F \ , \tag{7.8}$$

donde X es un vector n-dimensional de coordenadas $x_1(t)$, $x_2(t)$, ..., $x_n(t)$; F es un vector n-dimensional de coordenadas $f_1(t)$, $f_2(t)$,...,$f_n(t)$. Es conveniente en lo sucesivo representar dichos vectores como matrices de una columna:

$$X = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \quad , \quad F = \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_n \end{bmatrix} \quad ,$$

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \dots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{bmatrix} \quad , \quad \frac{dX}{dt} = \begin{bmatrix} \dfrac{dx_1}{dt} \\ \dfrac{dx_2}{dt} \\ \vdots \\ \dfrac{dx_n}{dt} \end{bmatrix} \quad .$$

Según la regla del producto de matrices, las filas del primer factor deben multiplicarse por la columna del segundo: por tanto,

$$AX = \begin{bmatrix} \sum_{j=1}^{n} a_{1j}x_j \\ \sum_{j=1}^{n} a_{2j}x_j \\ \vdots \\ \sum_{j=1}^{n} a_{nj}x_j \end{bmatrix} \quad , \quad AX + F = \begin{bmatrix} \sum_{j=1}^{n} a_{1j}x_j + f_1 \\ \sum_{j=1}^{n} a_{2j}x_j + f_2 \\ \vdots \\ \sum_{j=1}^{n} a_{nj}x_j + f_n \end{bmatrix} \quad .$$

La igualdad de matrices significa la igualdad de todos sus elementos, por lo cual una sola ecuación matricial (7.8). o bien

$$\begin{bmatrix} \dfrac{dx_1}{dt} \\ \dfrac{dx_2}{dt} \\ \vdots \\ \dfrac{dx_n}{dt} \end{bmatrix} = \begin{bmatrix} \sum_{j=1}^{n} a_{1j}x_j + f_1 \\ \sum_{j=1}^{n} a_{2j}x_j + f_2 \\ \vdots \\ \sum_{j=1}^{n} a_{nj}x_j + f_n \end{bmatrix} \quad .$$

es equivalente al sistema (7.7).

Si todas las funciones $a_{ij}(t)$ y $f_i(t)$ en (7.7) son continuas en el intervalo $a \leq t \leq b$, entonces en un entorno suficientemente pequeño de cada punto $(t_0, x_{10}, x_{20}, \dots, x_{n0})$, donde $a \leq t_0 \leq b$ se cumplen las condiciones del teorema de existencia y unicidad y, en consecuencia, por cada punto con estas propiedades pasa una sola curva integral del sistema (7.7).

En efecto, en el caso considerado los segundos miembros del sistema (7.7) son continuos, y sus derivadas parciales con respecto a cualquier x_j son acotadas, puesto que dichas derivadas son iguales a los coeficientes $a_{ij}(t)$, continuos en el intervalo $a \leq t \leq b$.

Definamos el *operador lineal* L por la igualdad

$$L[X] = \frac{dX}{dt} - AX \ ;$$

entonces la ecuación (7.8) puede escribirse en la forma aún más compacta

$$L[X] = F \ . \tag{7.9}$$

Si todas las $f_i(t) \equiv 0$ con $i = 1, 2, \ldots, n$ o, lo que es lo mismo, la matriz $F = 0$, el sistema (7.7) se llama *lineal homogéneo*. En forma compacta, el sistema lineal homogéneo tiene la forma

$$L[X] = 0 \ . \tag{7.10}$$

El operador L posee las dos propiedades siguientes:

(I) $L[cX] \equiv cL[X]$, donde c es una constante arbitraria.

(II) $L[X_1 + X_2] \equiv L[X_1] + L[X_2]$.

Un corolario de (i) y (ii) es

$$L\left[\sum_{i=1}^{m} c_i X_i\right] \equiv \sum_{i=1}^{m} c_i L[X_i] \ ,$$

donde las c_i son constantes arbitrarias.

Teorema 7.1 Si X es solución del sistema lineal homogéneo $L[X] = 0$, entonces cX, donde c es una constante arbitraria, es también solución de dicho sistema.

Teorema 7.2 La suma $X_1 + X_2$ de dos soluciones X_1 y X_2 del sistema de ecuaciones lineal homogéneo es solución de dicho sistema.

Corolario de los teoremas anteriores. La combinación lineal $\sum_{i=1}^{m} c_i X_i$ de las soluciones X_1, X_2,...,X_m del sistema $L[X] \equiv 0$ con coeficientes constantes arbitrarios es solución de dicho sistema.

Teorema 7.3 Si el sistema lineal homogéneo (7.10) con coeficientes reales a_{ij} tiene una solución compleja $X = U + iV$, las partes real e imaginaria

$$U = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{bmatrix} \ , \quad V = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix} \ ,$$

son por separado soluciones de dicho sistema.

Los vectores X_1, X_2,..., X_n, donde

$$X_i = \begin{bmatrix} x_{1i}(t) \\ x_{2i}(t) \\ \vdots \\ x_{ni}(t) \end{bmatrix} ,$$

se llaman *linealmente dependientes* en el intervalo $a \leq t \leq b$, si existen α_1, α_2,..., α_n constantes tales que

$$\alpha_1 X_1 + \alpha_2 X_2 + \ldots + \alpha_n X_n \equiv 0 \tag{7.11}$$

cuando $a \leq t \leq b$, y existe al menos un $\alpha_i \neq 0$. Si, en cambio, la identidad (7.11) se cumple sólo cuando $\alpha_1 = \alpha_2 = \ldots = \alpha_n = 0$, entonces los vectores X_1, X_2,..., X_n se llaman *linealmente independientes*.

Obsérvese que la identidad vectorial (7.11) es equivalente a las n identidades:

$$\begin{aligned} \sum_{i=1}^{n} \alpha_i x_{1i}(t) &\equiv 0 \ , \\ \sum_{i=1}^{n} \alpha_i x_{2i}(t) &\equiv 0 \ , \\ &\vdots \\ \sum_{i=1}^{n} \alpha_i x_{ni}(t) &\equiv 0 \ . \end{aligned} \tag{7.12}$$

Si los vectores X_i $(i = 1, 2, \ldots, n)$ son linealmente dependientes y, por lo tanto, existe un sistema no trivial de α_i (es decir, no todas las α_i son iguales a cero) que satisface al sistema (7.12) de n ecuaciones lineales homogéneas con respecto a α_i, entonces el determinante del sistema (7.12)

$$W = \begin{vmatrix} x_{11} & x_{12} & \ldots & x_{1n} \\ x_{21} & x_{22} & \ldots & x_{2n} \\ \vdots & \vdots & \ldots & \vdots \\ x_{n1} & x_{n2} & \ldots & x_{nn} \end{vmatrix} ,$$

debe ser igual a cero para todos los valores de t del intervalo $a \leq t \leq b$. Este determinante se llama *wronskiano* del sistema de vectores X_1, X_2,..., X_n.

Teorema 7.4 Si el wronskiano W de las soluciones X_1, X_2,..., X_n del sistema de ecuaciones lineales homogéneo (7.10) con coeficientes continuos $a_{ij}(t)$ en el intervalo $a \leq t \leq b$, es igual a cero por lo menos en un punto $t = t_0$ de dicho intervalo, entonces el conjunto de soluciones X_1, X_2,..., X_n son linealmente dependientes en el intervalo mencionado y, por consiguiente, $W = 0$ en dicho intervalo.

Observación. Este teorema no se extiende a vectores arbitrarios X_1, X_2,..., X_n, que no son soluciones de un sistema (7.10) con coeficientes continuos.

Teorema 7.5 La combinación lineal $\sum_{i=1}^{n} c_i X_i$ de n soluciones linealmente independientes X_1, X_2,..., X_n del sistema lineal homogéneo (7.10) con coeficientes continuos $a_{ij}(t)$ en el intervalo $a \leq t \leq b$, es solución general de este sistema en dicho intervalo.

Teorema 7.6 Si $\tilde{X}$ es solución del sistema lineal no homogéneo

$$L[X] = F \ , \tag{7.9}$$

y X_1 es solución del sistema homogéneo correspondiente $L[X] = 0$, entonces la suma $X_1 + \tilde{X}$ es también solución del sistema no homogéneo $L[X] = F$.

Teorema 7.7 La solución general en el intervalo $a \leq t \leq b$ del sistema no homogéneo (7.9) con coeficientes $a_{ij}(t)$ y segundo miembro $f_i(t)$ continuos en dicho intervalo, es igual a la suma de la solución general $\sum_{i=1}^{n} c_i X_i$ del sistema homogéneo correspondiente y de una solución particular $\tilde{X}$ del sistema no homogéneo considerado.

Teorema 7.8 principio de superposición.
La solución del sistema de ecuaciones lineales

$$L[X] = \sum_{i=1}^{m} F_i \ , \quad F_i = \begin{bmatrix} f_{1i}(t) \\ f_{2i}(t) \\ \vdots \\ f_{ni}(t) \end{bmatrix} \ , \tag{7.13}$$

es la suma $\sum_{i=1}^{m} X_i$ de las soluciones X_i de las ecuaciones

$$L[X_i] = F_i \ , (i = 1, 2, \ldots, m) \ . \tag{7.14}$$

Observación. El teorema anterior puede extenderse también al caso cuando $m \to \infty$, si la serie $\sum_{i=1}^{\infty} X_i$ converge y puede ser derivada término a término.

Teorema 7.9 Si el sistema de ecuaciones lineales

$$L[X] = U + iV \ ,$$

donde

$$U = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{bmatrix} \ , \quad V = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix} \ ,$$

con funciones reales $a_{ij}(t)$, $u_i(t)$, $v_i(t)$, $(i,\ j = 1, 2, \ldots, n)$, tiene la solución

$$X = \tilde{U} + i\tilde{V} \ , \quad \tilde{U} = \begin{bmatrix} \tilde{u}_1 \\ \tilde{u}_2 \\ \vdots \\ \tilde{u}_n \end{bmatrix} \ , \quad \tilde{V} = \begin{bmatrix} \tilde{v}_1 \\ \tilde{v}_2 \\ \vdots \\ \tilde{v}_n \end{bmatrix} \ ,$$

entonces la parte real de $\tilde{U}$ de la solución y su parte imaginaria $\tilde{V}$ son, respectivamente, soluciones de las ecuaciones

$$L[X] = U \quad \text{y} \quad L[X] = V \ .$$

Parte III

Computación.

Capítulo 8

Elementos del sistema operativo UNIX.

versión final 3.4-021128

8.1. Introducción.

En este capítulo se intentará dar los elementos básicos para poder trabajar en un ambiente UNIX. Sin pretender cubrir todos los aspectos del mismo, nuestro interés se centra en dar las herramientas al lector para que pueda realizar los trabajos del curso bajo este sistema operativo. Como comentario adicional, conscientemente se ha evitado la traducción de gran parte de la terminología técnica teniendo en mente que documentación disponible se encuentre, por lo general, en inglés y nos interesa que el lector sea capaz de reconocer los términos.

El sistema operativo UNIX es el más usado en investigación científica, tiene una larga historia y muchas de sus ideas y métodos se encuentran presentes en otros sistemas operativos. Algunas de las características relevantes del UNIX moderno son:

- Memoria grande, lineal y virtual: Un programa en una máquina de 32 Bits puede acceder y usar direcciones hasta los 4 GB en un máquina de sólo 4 MB de RAM. El sistema sólo asigna memoria auténtica cuando le hace falta, en caso de falta de memoria de RAM, se utiliza el disco duro (*swap*).
- Multitarea (*Multitasking*): Cada programa tiene asignado su propio "espacio" de memoria. Es **imposible** que un programa afecte a otro sin usar los servicios del sistema operativo. Si dos programas escriben en la misma dirección de memoria cada uno mantiene su propia "idea" de su contenido.
- Multiusuario: Más de una persona puede usar la máquina al mismo tiempo. Programas de otros usuarios continúan ejecutándose a pesar de que un nuevo usuario entre a la máquina.
- Casi todo tipo de dispositivo puede ser accedido como un archivo.
- Existen muchos aplicaciones diseñadas para trabajar desde la línea de comandos. Además, la mayoría de las aplicaciones permiten que la salida de una pueda ser la entrada de la otra.
- Permite compartir dispositivos (como disco duro) entre una red de máquinas.

Por su naturaleza de multiusuario, **nunca** se debe apagar una máquina UNIX[1], ya que una máquina apagada sin razón puede matar trabajos de días, perder los últimos cambios de tus archivos e ir degradando el sistema de archivos en dispositivos como el disco duro.

Entre los sistemas operativos UNIX actuales cabe destacar:

- Linux: esta disponible para: Intel x86 e IA-64; Motorola 68k, en particular, para las estaciones Sun3, computadores personales Apple Macintosh, Atari y Amiga; Sun SPARC; Alpha; Motorola/IBM PowerPC; ARM, máquinas NetWinder; Sun UltraSPARC; MIPS CPUs, máquinas SGI y estaciones Digital; HP PA-RISC; S/390, servidores IBM S/390 y SuperH procesadores Hitachi SuperH.
- SunOS[2]: disponible para la familia 68K así como para la familia SPARC de estaciones de trabajo SUN
- Solaris[3] : disponible para la familia SPARC de SUN así como para la familia x86.
- OSF1[4]: disponible para Alpha.
- Ultrix: disponible para VAX de Digital
- SYSVR4[5]: disponible para la familia x86, vax.
- IRIX: disponible para MIPS.
- AIX[6]: disponible para RS6000 de IBM y PowerPC.

8.2. Ingresando al sistema.

En esta sección comentaremos las operaciones de comienzo y fin de una sesión en UNIX así como la modificación de la contraseña (que a menudo no es la deseada por el usuario, y que por lo tanto puede olvidar con facilidad).

8.2.1. Terminales.

Para iniciar una sesión es necesario poder acceder a un terminal. Pueden destacarse dos tipos de terminales:

- Terminal de texto: consta de una pantalla y de un teclado. Como indica su nombre, en la pantalla sólo es posible imprimir caracteres de texto.
- Terminal gráfico: Consta de pantalla gráfica, teclado y *mouse*. Dicha pantalla suele ser de alta resolución. En este modo se pueden emplear ventanas que emulan el comportamiento de un terminal de texto (`xterm` o `gnome-terminal`).

[1] Incluyendo el caso en que la máquina es un PC normal corriendo Linux u otra versión de UNIX.
[2] SunOS 4.1.x también se conoce como Solaris 1.
[3] También conocido como SunOS 5.x, solaris 2 o Slowaris :-).
[4] También conocido como Dec Unix.
[5] También conocido como Unixware y Novell-Unix.
[6] También conocido como Aches.

8.2.2. Login.

El primer paso es encontrar un terminal libre donde aparezca el *login prompt* del sistema:

```
Debian GNU/Linux 3.0 hostname tty2

hostname login:
```

También pueden ocurrir un par de cosas:

- La pantalla no muestra nada.
  - Comprobar que la pantalla esté encendida.
  - Pulsar alguna tecla o mover el *mouse* para desactivar el protector de pantalla.
- Otra persona ha dejado una sesión abierta. En este caso existe la posibilidad de intentar en otra máquina o bien finalizar la sesión de dicha persona (si ésta no se halla en las proximidades).

  Una vez que se haya superado el paso anterior de encontrar el *login prompt* se procede con la introducción del *Username* al *prompt* de *login* y después la contraseña (*password*) adecuada.

8.2.3. Passwords.

El *password* puede ser cualquier secuencia de caracteres a elección. Deben seguirse las siguientes pautas:

- Debe ser fácil de recordar por uno mismo. Si se olvida, deberá pasarse un mal rato diciéndole al administrador de sistema que uno lo ha olvidado.
- Para evitar que alguna persona no deseada obtenga el *password* y tenga libre acceso a los archivos de tu cuenta:
  - Las mayúsculas y minúsculas no son equivalentes sin embargo se recomienda que se cambie de una a otra.
  - Los caracteres numéricos y no alfabéticos también ayudan. Debe tenerse sin embargo la precaución de usar caracteres alfanuméricos que se puedan encontrar en todos los terminales desde los que se pretenda acceder.
  - Las palabras de diccionario deben ser evitadas.
- Debe cambiarlo si crees que su *password* es conocido por otras personas, o descubre que algún intruso[7] está usando su cuenta.
- El *password* debe ser cambiado con regularidad.

[7]Intruso es cualquier persona que no sea el usuario.

La orden para cambiar el password en UNIX es `passwd`. A menudo cuando existen varias máquinas que comparten recursos (disco duro, impresora, correo electrónico, ...), para facilitar la administración de dicho sistema se unifican los recursos de red (entre los que se hayan los usuarios de dicho sistema) en una base de datos común. Dicho sistema se conoce como NIS (*Network Information Service*)[8]. Si el sistema empleado dispone de este servicio, la modificación de la contraseña en una máquina supone la modificación en todas las máquinas que constituyan el dominio NIS.

8.2.4. Cerrando la sesión.

Es importante que nunca se deje abierta una sesión, pues algún "intruso" podría tener libre acceso a archivos de tu propiedad y manipularlos de forma indeseable para ti. Para evitar todo esto basta teclear `logout` ó `exit` y habrá acabado tu sesión de UNIX en dicha máquina[9].

8.3. Archivos y directorios.

Aunque haya diferentes distribuciones y cada una traiga sus programas, la estructura básica de directorios y archivos es más o menos la misma en todas:

```
/-|--> bin
  |--> boot
  |--> cdrom
  |--> dev
  |--> etc
  |--> floppy
  |--> home
  |--> lib
  |--> mnt
  |--> proc
  |--> root
  |--> sbin
  |--> tmp
  |--> usr -|--> X11
  |         |--> bin
  |         |--> include
  |         |--> lib
  |         |--> local -|--> bin
  |         |           |--> lib
  |         |--> man
  |         |--> src --> linux
  |         |--> doc
  |--> var --> adm
```

[8] Antiguamente se conocía como YP (*Yellow Pages*), pero debido a un problema de marca registrada de *United Kingdom of British Telecomunications* se adoptaron las siglas NIS.

[9] En caso que se estuviera trabajando bajo X-Windows debes cerrar la sesión con `Log out of Gnome`.

El árbol que observamos muestra un típico árbol de directorios en Linux. Pueden variar, sin embargo, algunos de los nombres dependiendo de la distribución o versión de Linux que se esté usando. Algunos directorios destacados son:

- `/home` - Espacio reservado para las cuentas de los usuarios.
- `/bin`, `/usr/bin` - Binarios (ejecutables) básicos de UNIX.
- `/etc`, aquí se encuentran los archivos de configuración de todo el software de la máquina.
- `/proc`, es un sistema de archivo virtual. Contiene archivos que residen en memoria pero no en el disco duro. Hace referencia a los programas que se están corriendo en el momento en el sistema.
- `/dev` (*device*) (dispositivo). Aquí se guardan los controladores de dispositivos. Se usan para acceder a los dispositivos físicos del sistema y recursos como discos duros, *modems*, memoria, *mouse*, etc. Algunos dispositivos:
  - `hd`: `hda1` será el disco duro IDE, primario (`a`), y la primera partición (`1`).
  - `fd`: los archivos que empiecen con las letras `fd` se referirán a los controladores de las disketteras: `fd0` sería la primera diskettera, `fd1` sería la segunda y así sucesivamente.
  - `ttyS`: se usan para acceder a los puertos seriales como por ejemplo, `ttyS0` es el puerto conocido como `com1`.
  - `sd`: son los dispositivos SCSI. Su uso es muy similar al del `hd`.
  - `lp`: son los puertos paralelos. `lp0` es el puerto conocido como `LPT1`.
  - `null`: éste es usado como un agujero negro, ya que todo lo que se dirige allí desaparece.
  - `tty`: hacen referencia a cada una de las consolas virtuales. Como es de suponer, `tty1` será la primera consola virtual, `tty2` la segunda, etc.
- `/usr/local` - Zona con las aplicaciones no comunes a todos los sistemas UNIX, pero no por ello menos utilizadas. En `/usr/share/doc` se puede encontrar información relacionada con dicha aplicación (en forma de páginas de manual, texto, html o bien archivos dvi, Postscript o pdf). También encontramos archivos de ejemplo, tutoriales, *HOWTO*, etc.

8.4. Órdenes básicas.

Para ejecutar un comando, basta con teclear su nombre (también debes tener permiso para hacerlo). Las opciones o modificadores empiezan normalmente con el carácter `-` (p. ej. `ls -l`). Para especificar más de una opción, se pueden agrupar en una sola cadena de caracteres (`ls -l -h` es equivalente a `ls -lh`). Algunos comandos aceptan también opciones dadas por palabras completas, en cuyo caso usualmente comienzan con `--` (`ls --color=auto`).

8.4.1. Archivos y directorios.

En un sistema computacional la información se encuentra en archivos que la contienen (tabla de datos, texto ASCII, fuente en lenguaje C, Fortran o C++, ejecutable, imagen, mp3, figura, resultados de simulación, ...). Para organizar toda la información se dispone de una entidad denominada directorio, que permite el almacenamiento en su interior tanto de archivos como de otros directorios[10]. Se dice que la estructura de directorios en UNIX es jerárquica o arborescente, debido a que todos los directorios nacen en un mismo punto (denominado directorio raíz). De hecho, la zona donde uno trabaja es un nodo de esa estructura de directorios, pudiendo uno a su vez generar una estructura por debajo de ese punto. Un archivo se encuentra situado siempre en un directorio y su acceso se realiza empleando el camino que conduce a él en el Árbol de Directorios del Sistema. Este camino es conocido como el *path*. El acceso a un archivo se puede realizar empleando:

- Path Absoluto, aquel que empieza con `/`

  Por ejemplo : `/etc/printcap`

- Path Relativo, aquel que NO empieza con `/`

  Por ejemplo : `../examples/rc.dir.01`

- Nombres de archivos y directorios pueden usar un máximo de 255 caracteres, cualquier combinación de letras y símbolos (el caracter `/` no se permite).

Los caracteres comodín pueden ser empleados para acceder a un conjunto de archivos con características comunes. El signo `*` puede sustituir cualquier conjunto de caracteres[11] y el signo `?` a cualquier caracter individual. Por ejemplo:[12]

```
bash$ ls
f2c.1          flexdoc.1    rcmd.1    rptp.1   zforce.1
face.update.1  ftptool.1    rlab.1    rxvt.1   zip.1
faces.1        funzip.1     robot.1   zcat.1   zipinfo.1
flea.1         fvwm.1       rplay.1   zcmp.1   zmore.1
flex.1         rasttoppm.1  rplayd.1  zdiff.1  znew.1
bash$ ls rp*
rplay.1        rplayd.1     rptp.1
bash$ ls *e??
face.update.1  zforce.1     zmore.1
```

Los archivos cuyo nombre comiencen por `.` se denominan **ocultos**, así por ejemplo en el directorio de partida de un usuario.

```
bash$ ls -a  user
.                .alias       .fvwmrc  .login    .xinitrc
..               .cshrc       .joverc  .profile
.Xdefaults       .enviroment  .kshrc   .tcshrc
```

[10]Normalmente se acude a la imagen de una carpeta que puede contener informes, documentos o bien otras carpetas, y así sucesivamente.

[11]Incluido el punto '.', UNIX no es DOS.

[12]`bash$` es el *prompt* en todos los ejemplos.

Algunos caracteres especiales para el acceso a archivos son:

.	Directorio actual
..	Directorio superior en el árbol
~	Directorio `$HOME`
~user	Directorio `$HOME` del usuario user

8.4.2. Órdenes relacionadas con directorios.

`ls` (LiSt)
Este comando permite listar los archivos de un determinado directorio. Si no se le suministra argumento, lista los archivos y directorios en el directorio actual. Si se añade el nombre de un directorio el listado es del directorio suministrado. Existen varias opciones que modifican su funcionamiento entre las que destacan:

- `-l` (Long listing) proporciona un listado extenso, que consta de los permisos[13] de cada archivo, el usuario el tamaño del archivo, ...
- `-a` (list All) lista también los archivos ocultos.
- `-R` (Recursive) lista recursivamente el contenido de todos los directorios que se encuentre.
- `-t` ordena los archivos por tiempo de modificación.
- `-S` ordena los archivos por tamaño.
- `-r` invierte el sentido de un orden.
- `-p` agrega un caracter al final de cada nombre de archivo, indicando el tipo de archivo (por ejemplo, los directorios son identificados con un `/` al final).

`pwd` (Print Working Directory)
Este comando proporciona el nombre del directorio actual.
`cd` (Change Directory)
Permite moverse a través de la estructura de directorios. Si no se le proporciona argumento se provoca un salto al directorio `$HOME`. El argumento puede ser un nombre absoluto o relativo de un directorio. `cd -` vuelve al último directorio visitado.
`mkdir` (MaKe DIRectory)
Crea un directorio con el nombre (absoluto o relativo) proporcionado.
`rmdir` (ReMove DIRectory)
Elimina un directorio con el nombre (absoluto o relativo) suministrado. Dicho directorio debe de estar vacío.

[13] Se comentará posteriormente este concepto.

8.4.3. Visitando archivos.

Este conjunto de órdenes permite visualizar el contenido de un archivo sin modificar su contenido.

`cat`
Muestra por pantalla el contenido de un archivo que se suministra como argumento.

`more`
Este comando es análogo al anterior, pero permite la paginación.

`less`
Es una versión mejorada del anterior. Permite moverse en ambas direcciones. Otra ventaja es que no lee el archivo entero antes de arrancar.

8.4.4. Copiando, moviendo y borrando archivos.

`cp` (CoPy)
copia un archivo(s) con otro nombre y/o a otro directorio. Veamos algunas opciones:

- `-i` (interactive), impide que la copia provoque una pérdida del archivo destino si éste existe[14].
- `-R` (recursive), copia un directorio y toda la estructura que cuelga de él.

`mv` (MoVe)
Mover un archivo(s) a otro nombre y/o a otro directorio. Dispone de opciones análogas al caso anterior.

`rm` (ReMove)
Borrar un archivo(s). En caso de que el argumento sea un directorio y se haya suministrado la opción `-r`, es posible borrar el directorio y todo su contenido. La opción `-i` pregunta antes de borrar.

8.4.5. Espacio de disco.

El recurso de almacenamiento en el disco es siempre limitado, a continuación se comentan un par de comandos relacionados con la ocupación de este recurso:

`du` (Disk Usage)
Permite ver el espacio de disco ocupado (en bloques de disco[15]) por el archivo o directorio suministrado como argumento. La opción `-s` impide que cuando se aplique recursividad en un directorio se muestren los subtotales. La opción `-h` imprime los tamaños en un formato fácil de leer (Human readable).

`df` (Disk Free)
Muestra los sistemas de archivos que están montados en el sistema, con las cantidades totales, usadas y disponibles para cada uno. `df -h` muestra los tamaños en formato fácil de leer.

[14] Muchos sistemas tienen esta opción habilitada a través de un alias, para evitar equivocaciones.
[15] 1 bloque normalmente es 1 Kbyte.

8.4.6. Links.

`ln` (LiNk)
Permite realizar un enlace (link) entre dos archivos o directorios. Un enlace puede ser:

- *hard link*: se puede realizar sólo entre archivos del mismo sistema de archivos. El archivo enlazado apunta a la zona de disco donde se halla el archivo original. Por tanto, si se elimina el archivo original, el enlace sigue teniendo acceso a dicha información. Es el enlace por omisión.
- *symbolic link*: permite enlazar archivos/directorios[16] de diferentes sistemas de archivos. El archivo enlazado apunta al nombre del original. Así si se elimina el archivo original el enlace apunta hacia un nombre sin información asociada. Para realizar este tipo de enlace debe emplearse la opción `-s`.

Un enlace permite el uso de un archivo en otro directorio distinto del original sin necesidad de copiarlo, con el consiguiente ahorro de espacio.

8.4.7. Protección de archivos.

Dado que el sistema de archivos UNIX es compartido por un conjunto de usuarios, surge el problema de la necesidad de privacidad. Sin embargo, dado que existen conjuntos de personas que trabajan en común, es necesario la posibilidad de que un conjunto de usuarios puedan tener acceso a una serie de archivos (que puede estar limitado para el resto de los usuarios). Cada archivo y directorio del sistema dispone de un propietario, un grupo al que pertenece y unos **permisos**. Existen tres tipos fundamentales de permisos:

- **lectura** (**r**-*Read*): en el caso de un archivo, significa poder examinar el contenido del mismo; en el caso de un directorio significa poder entrar en dicho directorio.
- **escritura** (**w**-*Write*): en el caso de un archivo significa poder modificar su contenido; en el caso de un directorio es crear un archivo o directorio en su interior.
- **ejecución** (**x**-*eXecute*): en el caso de un archivo significa que ese archivo se pueda ejecutar (binario o archivo de procedimientos); en el caso de un directorio es poder ejecutar alguna orden dentro de él.

Se distinguen tres grupos de personas sobre las que especificar permisos:

- **user**: el usuario propietario del archivo.
- **group**: el grupo propietario del archivo (excepto el usuario). Como ya se ha comentado, cada usuario puede pertenecer a uno o varios grupos y el archivo generado pertenece a uno de los mismos.
- **other**: el resto de los usuarios (excepto el usuario y los usuarios que pertenezcan al grupo)

[16] Debe hacerse notar que los directorios sólo pueden ser enlazados simbólicamente.

También se puede emplear *all* que es la unión de todos los anteriores. Para visualizar las protecciones de un archivo o directorio se emplea la orden `ls -l`, cuya salida es de la forma:

```
-rw-r--r-- ...otra información... nombre
```

Los 10 primeros caracteres muestran las protecciones de dicho archivo:

- El primer caracter indica el tipo de archivo de que se trata:
  - archivo
  - `d` directorio
  - `l` enlace (*link*)
  - `c` dispositivo de caracteres (p.e. puerta serial)
  - `b` dispositivo de bloques (p.e. disco duro)
  - `s` socket (conexión de red)
- Los caracteres 2, 3, 4 son los permisos de usuario
- Los caracteres 5, 6, 7 son los permisos del grupo
- Los caracteres 8, 9, 10 son los permisos del resto de usuarios

  Así en el ejemplo anterior `-rw-r--r--` se trata de un archivo donde el usuario puede leer y escribir, mientras que el grupo y el resto de usuarios sólo pueden leer. Estos suelen ser los permisos por omisión para un archivo creado por un usuario. Para un directorio los permisos por omisión suelen ser: `drwxr-xr-x`, donde se permite al usuario "entrar" en el directorio y ejecutar órdenes desde él.

  `chmod` (CHange MODe)
  Esta orden permite modificar los permisos de un archivo. Con opción `-R` es recursiva.
  `chmod` *permisos files*
  Existen dos modos de especificar los permisos:

- Modo absoluto o modo numérico. Se realiza empleando un número que resulta de la `OR` binario de los siguientes modos:

400	lectura por el propietario.
200	escritura por el propietario.
100	ejecución (búsqueda) por el propietario.
040	lectura por el grupo.
020	escritura por el grupo.
010	ejecución (búsqueda) por el grupo.
004	lectura por el resto.
002	escritura por el resto.
001	ejecución (búsqueda) por el resto.
4000	Set User ID, cuando se ejecuta el proceso corre con los permisos del dueño del archivo.

  Por ejemplo:

```
chmod 640 *.txt
```

Permite la lectura y escritura por el usuario, lectura para el grupo y ningún permiso para el resto, de un conjunto de archivos que acaban en `.txt`

- Modo simbólico o literal. Se realiza empleando una cadena (o cadenas separadas por comas) para especificar los permisos. Esta cadena se compone de los siguientes tres elementos: `who operation permission`

  - `who` : es una combinación de:
    - `u` : user
    - `g` : group
    - `o` : others
    - `a` : all (equivalente a `ugo`)

    Si se omite este campo se supone `a`, con la restricción de no ir en contra de la máscara de creación (`umask`).
  - `operation`: es una de las siguientes operaciones:
    - `+` : añadir permiso.
    - `-` : eliminar permiso.
    - `=` : asignar permiso, el resto de permisos de la misma categoría se anulan.
  - `permission`: es una combinación de los caracteres:
    - `r` : *read*.
    - `w` : *write*.
    - `x` : *execute*.
    - `s` : en ejecución fija el usuario o el grupo.

  Por ejemplo:
  `chmod u+x tarea`
  Permite la ejecución por parte del usuario[17] del archivo `tarea`.

  `chmod u=rx, go=r *.txt`
  Permite la lectura y ejecución del usuario, y sólo la lectura por parte del grupo y el resto de usuarios.

`umask`
Esta es una orden intrínseca del Shell que permite asignar los permisos que se desea tengan los archivos y directorios por omisión. El argumento que acompaña a la orden es un número octal que aplicará una XOR sobre los permisos por omisión (`rw-rw-rw-`) para archivos y (`rwxrwxrwx`) para directorios. El valor por omisión de la máscara es `022` que habilita al usuario para lectura-escritura, al grupo y al resto para lectura. Sin argumentos muestra el valor de la máscara.

[17] Un error muy frecuente es la creación de un archivo de órdenes (*script file*) y olvidar permitir la ejecución del mismo.

`chgrp` (CHange GRouP)
Cambia el grupo propietario de una serie de archivos/directorios
`chgrp` *grupo files*
El usuario que efectúa esta orden debe pertenecer al grupo mencionado.
`chown` (CHange OWNer)
Cambia el propietario y el grupo de una serie de archivos/directorios
`chown` *user:group files*
La opción `-r` hace que la orden se efectúe recursivamente.
`id`
Muestra la identificación del usuario[18], así como el conjunto de grupos a los que el usuario pertenece.

```
user@hostname:~$ id
uid=1000(user) gid=1000(group) groups=1000(group),25(floppy),29(audio)
user@hostname:~$
```

8.4.8. Filtros.

Existe un conjunto de órdenes en UNIX que permiten el procesamiento de archivos de texto. Se denominan **filtros** (*Unix Filters*), porque normalmente se trabaja empleando redirección recibiendo datos por su `stdin`[19] y retornándolos modificados por su `stdout`[20].

Para facilitar la comprensión de los ejemplos siguientes supondremos que existen dos archivo llamado `mylist.txt` y `yourlist.txt` que tienen en su interior:

```
mylist.txt       yourlist.txt

1 190            1 190
2 280            2 281
3 370            3 370
```

`echo`
Este no es propiamente un filtro, pero nos será muy útil más adelante. Despliega sobre la pantalla un mensaje

```
user@hostname:~$ echo Hola Mundo
Hola Mundo
user@hostname:~$
```

`cat`
Es el filtro más básico, copia la entrada a la salida.

```
user@hostname:~$ cat  mylist.txt
1 190
```

[18] A pesar de que el usuario se identifica por una cadena denominada *username*, también existe un número denominado UID que es un identificativo numérico de dicho usuario.
[19] Entrada estándar.
[20] Salida estándar.

```
2 280
3 370
user@hostname:~$
```

`cut`

Para un archivo compuesto por columnas de datos, permite escribir sobre la salida cierto intervalo de columnas. La opción `-b N-M` permite indicar el intervalo en bytes que se escribirán en la salida.

```
user@hostname:~$ cut -b 3-4 mylist.txt
19
28
37
user@hostname:~$
```

`paste`

Mezcla líneas de distintos archivos. Escribe líneas en el `stdout` pegando secuencialmente las líneas correspondientes de cada unos de los archivo separadas por tab. Ejemplo, supongamos que tenemos nuestros archivos `mylist.txt` y `mylist.txt` y damos el comando

```
user@hostname:~$ paste mylist.txt yourlist.txt
1 190   1 190
2 280   2 281
3 370   3 370
user@hostname:~$
```

`sed`

Es un editor de flujo. Veamos algunos ejemplos

```
user@hostname:~$ sed = mylist.txt
1
1 190
2
2 280
3
3 370
user@hostname:~$
```

Numera las líneas.

```
user@hostname:~$ sed -n '3p' mylist.txt
3 370
user@hostname:~$
```

Sólo muestra la línea 3. El modificador `-n` suprime la impresión de todas las líneas excepto aquellas especificadas por `p`. Separando por coma damos un rango en el número de líneas.

```
user@hostname:~$ sed -e '2q' mylist.txt
1 190
2 280
user@hostname:~$
```

Muestra hasta la línea 2 y luego se sale de `sed`. El modificador `-e` corre un *script*, secuencia de comandos.

```
user@hostname:~$ sed -e 's/0/a/g' mylist.txt
1 19a
2 28a
3 37a
user@hostname:~$
```

Reemplaza todos los 0 del archivo por la letra a. Este es uno de los usos más comunes.

```
user@hostname:~$ sed -e '/2 2/s/0/a/g' mylist.txt
1 190
2 28a
3 370
user@hostname:~$
```

Busca las líneas con la secuencia `2 2` y en ellas reemplaza todos los 0 por la letra a.

```
user@hostname:~$ sed -e s/1/XX/2 mylist.txt
1 XX90
2 280
3 370
user@hostname:~$
```

Reemplaza la segunda aparición de un 1 en una línea por los caracteres XX.

`diff`

Permite comparar el contenido de dos archivos

```
user@hostname:~$ diff mylist.txt yourlist.txt
2c2
< 2 280
---
> 2 281
user@hostname:~$
```

Hay una diferencia entre los archivos en la segunda fila.

`sort`

Permite ordenar alfabéticamente

```
user@hostname:~$ sort -n -r mylist.txt
3 370
2 280
1 190
user@hostname:~$
```

La opción `-n` considera los valores numéricos y la opción `-r` invierte el orden.

`find`

Permite la búsqueda de un archivo en la estructura de directorios

`find . -name file.dat -print`

Comenzando en el directorio actual recorre la estructura de directorios buscando el archivo `file.dat`, cuando lo encuentre imprime el path al mismo.

`find . -name '*~' -exec rm '{}' \;`

Busca en la estructura de directorios un archivo que acabe en `~` y lo borra. `xargs` ordena repetir orden para cada argumento que se lea desde *stdin*. Permite el uso muy eficiente de `find`.

`find . -name '*.dat' -print | xargs mv ../data \;`

Busca en la estructura de directorios todos los archivos que acaben en `.dat`, y los mueve al directorio `../data`.

`grep`

Permite la búsqueda de una cadena de caracteres en uno o varios archivos, imprimiendo el nombre del archivo y la línea en que se encuentra la cadena.

```
user@hostname:~$ grep 1 *list.txt
mylist.txt:1 190
yourlist.txt:1 190
yourlist.txt:2 281
user@hostname:~$
```

Algunas opciones útiles

- `-c` Elimina la salida normal y sólo cuenta el número de apariciones de la cadena en cada archivo.
- `-i` Ignora para la comparación entre la cadena dada y el archivo, si la cadena está en mayúsculas o minúsculas.
- `-n` Incluye el número de líneas en que aparece la cadena en la salida normal.
- `-r` Hace la búsqueda recursiva.
- `-v` Invierte la búsqueda mostrando todas las líneas donde no aparece la cadena pedida.

`head`

Muestra las primeras diez líneas de un archivo.

`head -30 file` Muestra las 30 primeras líneas de *file*.

```
user@hostname:~$ head -1  mylist.txt
1 190
user@hostname:~$
```

`tail`
Muestra las diez últimas líneas de un archivo.
`tail -30 file` Muestra las 30 últimas líneas de *file*.
`tail +30 file` Muestra desde la línea 30 en adelante de *file*.

```
user@hostname:~$ tail -1  mylist.txt
3 370
user@hostname:~$
```

`awk`
Es un procesador de archivos de texto que permite la manipulación de las líneas de forma tal que tome decisiones en función del contenido de la misma. Ejemplo, supongamos que tenemos nuestro archivo `mylist.txt` con sus dos columnas

```
user@hostname:~$ awk '{print $2, $1 }'  mylist.txt
190 1
280 2
370 3
user@hostname:~$
```

Imprime esas dos columnas en orden inverso.

```
user@hostname:~$ awk '{print ``a'', 8*$1, $2-1 }'  mylist.txt
a 8 189
a 16 279
a 24 369
user@hostname:~$
```

Permite operar sobre las columnas.

```
user@hostname:~$ awk '{print }'  mylist.txt
1 190
2 280
3 370
user@hostname:~$
```

Funciona como el comando `cat`

```
user@hostname:~$ awk '{ if (NR>1 && NR < 3) print}' mylist.txt
2 280
user@hostname:~$
```

Sólo imprime la línea 2.
`tar`
Este comando permite la creación/extracción de archivos contenidos en un único archivo

denominado `tarfile` (o `tarball`). Este `tarfile` suele ser luego comprimido con `gzip`, la versión de compresión **gnu**[21] o bien con `bzip2`.

La acción a realizar viene controlada por el primer argumento:

- `c` (Create) creación
- `x` (eXtract) extracción
- `t` (lisT) mostrar contenido
- `r` añadir al final
- `u` (Update) añadir aquellos archivos que no se hallen en el tarfile o que hayan sido modificados con posterioridad a la versión que aparece.

A continuación se colocan algunas de las opciones:

- `v` Verbose (indica qué archivos son agregados a medida que son procesados)
- `z` Comprimir o descomprimir el contenido con `gzip`.
- `j` Comprimir o descomprimir el contenido con `bzip2`.
- `f` File: permite especificar el archivo para el tarfile.

Veamos algunos ejemplos:

```
tar cvf simul.tar *.dat
```

Genera un archivo `simul.tar` que contiene todos los archivos que terminen en `.dat` del directorio actual. A medida que se va realizando indica el tamaño en bloques de cada archivo añadido modo *verbose*.

```
tar czvf simul.tgz *.dat
```

Igual que en el caso anterior, pero el archivo generado simul.tgz ha sido comprimido empleando gzip.

```
tar tvf simul.tar
```

Muestra los archivos contenidos en el tarfile `simul.tar`.

```
tar xvf simul.tar
```

Extrae todos los archivos contenidos en el tarfile `simul.tar`.

`wc` (*Word Count*) Contabiliza el número de líneas, palabras y caracteres de un archivo.

```
user@hostname:~$ wc  mylist.txt
      3       6      18 mylist.txt
user@hostname:~$
```

El archivo tiene 3 líneas, 6 palabras, considerando cada número como una palabra *i.e.* 1 es la primera palabra y 190 la segunda, y finalmente 18 caracteres. ¿Cuáles son los 18 caracteres?

[21] **gnu** es un acrónimo recursivo, significa: **gnu**'s Not UNIX! **gnu** es el nombre del producto de la *Free Software Foundation*, una organización dedicada a la creación de programas compatibles con UNIX (y mejorado respecto a los estándars) y de libre distribución. La distribución de Linux **gnu** es **debian**.

8.4.9. Otros usuarios y máquinas

`users` `who` `w`
Para ver quién está conectado en la máquina.

`ping`
Verifica si una máquina está conectada a la red y si el camino de Internet hasta la misma funciona correctamente.

`finger`
`finger user`, muestra información[22] sobre el usuario `user` en la máquina local.
`finger user@hostname`, muestra información sobre un usuario llamado `user` en una máquina `hostname`.
`finger @hostname`, muestra los usuarios conectados de la máquina `hostname`.

8.4.10. Fecha

`cal`
Muestra el calendario del mes actual. Con la opción `-y` y el año presenta el calendario del año completo.

`date`
Muestra el día y la hora actual.

8.4.11. Transferencia a diskettes.

La filosofía de diferentes unidades (A:, B:,...) difiere de la estructura única del sistema de archivos que existe en UNIX. Son varias las alternativas que existen para la transferencia de información a diskette.

- Una posibilidad es disponer de una máquina WIN9X con ftp instalado y acceso a red. Empleando dicha aplicación se pueden intercambiar archivos entre un sistema y el otro.

- Existe un conjunto de comandos llamados `mtools` disponible en multitud plataformas, que permiten el acceso a diskettes en formato WIN9X de una forma muy eficiente.

  `mdir a:` Muestra el contenido de un diskette en `a:`.

  `mcopy file a:` Copia el archivo `file` del sistema de archivos UNIX en un diskette en `a:`.

  `mcopy a:file file` Copia el archivo `a:file` del diskette en el sistema de archivos UNIX con el nombre `file`.

  `mdel a:file` Borra el archivo `a:file` del diskette.

[22] La información proporcionada es el nombre de completo del usuario, las últimas sesiones en dicha máquina, si ha leído o no su correo y el contenido de los archivos `.project` y `.plan` del usuario.

Con `a:` nos referimos a la primera diskettera `/dev/fd0` y luego al archivo que se encuentra en el diskette. Su nombre se compone de `a:filename`. Si se desea emplear el caracter comodín para un conjunto de archivos del diskette, estos deben rodearse de dobles comillas para evitar la actuación del *shell* (p.e. `mcopy ‘‘a:*.dat’’`). La opción `-t` realiza la conversión necesaria entre UNIX y WIN9X, que se debe realizar **sólo** en archivos de texto.

- Una alternativa final es montar el dispositivo `/dev/fd0` en algún directorio, típicamente `/floppy`, considerando el tipo especial de sistema de archivos que posee `vfat` y luego copiar y borrar usando comandos UNIX. Esta forma suele estar restringida sólo a `root`, el comando: `mount -t vfat /dev/fd0 /floppy` no puede ser dado por un usuario. Sin embargo, el sistema aceptará el comando `mount /floppy` de parte del usuario. Una vez terminado el trabajo con el floppy éste debe ser desmontado, antes de sacarlo, mediante el comando: `umount /floppy`.

8.4.12. Diferencias entre los sistemas.

Cuando se transfieren archivos de texto entre DOS y UNIX sin las precauciones adecuadas pueden aparecer los siguientes problemas:

- En DOS los nombres de los archivos pueden tener un máximo de 8 caracteres y una extensión de 3 caracteres. En UNIX no existe restricción respecto a la longitud del nombre, y aunque pueden llevar extensión, no es obligatorio. También pueden tener más de una extensión `algo.v01.tar.gz`, esto complica mucho a otros sistemas que tienen limitaciones en los nombres.

- El cambio de línea en DOS se compone de *Carriage Return* y *Line Feed*. Sin embargo, en UNIX sólo existe el *Carriage Return*. Así un archivo de UNIX visto desde DOS parece una única línea. El caso inverso es la aparición del caracter `^M` al final de cada línea. Además, el fin de archivo en DOS es `^Z` y en UNIX es `^D`.

- La presencia de caracteres con código ASCII por encima del 127 (ASCII extendido) suele plantear problemas. Debido a que en DOS dicho código depende de la asignación hecha, que a su vez depende del país.

Usando el comando `tr` se puede transformar un archivo con cambios de líneas para dos en uno para UNIX. Sabiendo que `^M` es ASCII 13 decimal, pero 15 en octal:

```
tr -d '\015' < datafile > TEMPFILE
mv -f TEMPFILE datafile
```

En Debian, instalando el paquete `sysutils`, queda instalado el comando `dos2unix` que también lo hace.

8.5. Shells.

El sistema operativo UNIX soporta varios intérpretes de comandos o *shells*, que ayudan a que la interacción con el sistema sea lo más cómoda y amigable posible. La elección de cuál es el *shell* más cómoda es algo personal; en este punto sólo indicaremos las cuatro más significativas y populares:

- **sh** : Bourne SHell, el *shell* básico, no pensado para uso interactivo.
- **csh** : C-SHell, *shell* con sintaxis como el lenguaje "C". El archivo de configuración es `.cshrc` (en el directorio `$HOME`).
- **tcsh** : alTernative C-Shell (Tenex-CSHell), con editor de línea de comando. El archivo de configuración es `.tcshrc`, o en caso de no existir, `.cshrc` (en el directorio `$HOME`).
- **bash** : Bourne-Again Shell, con lo mejor de sh, ksh y tcsh. El archivo de configuración es `.bash_profile` cuando se entra a la cuenta por primera vez, y después el archivo de configuración es `.bashrc` siempre en el directorio `$HOME`. La línea de comando puede ser editada usando comandos (secuencias de teclas) del editor `emacs`. Es el *shell* por defecto de Linux.

Si queremos cambiar de *shell* en un momento dado, sólo será necesario que tecleemos el nombre del mismo y estaremos usando dicho *shell*. Si queremos usar de forma permanente otro *shell* del que tenemos asignado por omisión[23] podemos emplear la orden `chsh` que permite realizar esta acción.

En los archivos de configuración se encuentran las definiciones de las variables de entorno (*enviroment variables*) como camino de búsqueda `PATH`, los aliases y otras configuraciones personales. Veamos unos caracteres con especial significado para el Shell:

- ` ‘ ` [24] permite que el output de un comando reemplace al nombre del comando. Por ejemplo: `echo ‘pwd‘` imprime por pantalla el nombre del directorio actual.

```
user@hostname:~$ echo ‘pwd‘
/home/user
user@hostname:~$
```

- ` ’ ` [25] preserva el significado literal de cada uno de los caracteres de la cadena que delimita.

```
user@hostname:~$ echo ’Estoy en ‘pwd‘’
Estoy en ‘pwd‘
user@hostname:~$
```

[23] Por omisión se asigna bash.
[24] Acento agudo o inclinado hacia atrás, *backquote*.
[25] Acento usual o inclinado hacia adelante, *single quote*.

- [”] [26] preserva el significado literal de todos los caracteres de la cadena que delimita, salvo $, ‘, \.

```
user@hostname:~$ echo "Estoy en ‘pwd‘"
Estoy en /home/user
user@hostname:~$
```

- [;] permite la ejecución de más de una orden en una sola línea de comando.

```
user@hostname:~$ mkdir mydir; cd mydir; cp *.txt . ; cd ..
user@hostname:~$
```

8.5.1. Variables de entorno.

Las variables de entorno permiten la configuración, por defecto, de muchos programas cuando ellos buscan datos o preferencias. Se encuentran definidas en los archivos de configuración anteriormente mencionados. Para referenciar a las variables se debe poner el símbolo `$` delante, por ejemplo, para mostrar el camino al directorio por defecto del usuario `user`:

```
user@hostname:~$ echo $HOME
/home/user
user@hostname:~$
```

Las variables de entorno más importantes son:

- `HOME` - El directorio por defecto del usuario.
- `PATH` - El camino de búsqueda, una lista de directorios separado con ‘:’ para buscar programas.
- `EDITOR` - El editor por defecto del usuario.
- `DISPLAY` - Bajo el sistema de X windows, el nombre de máquina y pantalla que está usando. Si esta variable toma el valor `:0` el despliegue es local.
- `TERM` - El tipo de terminal. En la mayoría de los casos bajo el sistema X windows se trata de `xterm` y en la consola en Linux es `linux`. En otros sistemas puede ser `vt100`.
- `SHELL` - La *shell* por defecto.
- `MANPATH` - Camino para buscar páginas de manuales.
- `PAGER` - Programa de paginación de texto (`less` o `more`).
- `TMPDIR` - Directorio para archivos temporales.

[26] *double quote.*

8.5.2. Redirección.

Cuando un programa espera que se teclee algo, aquello que el usuario teclea se conoce como el *Standard Input*: `stdin`. Los caracteres que el programa retorna por pantalla es lo que se conoce como *Standard Output*: `stdout` (o *Standard Error*: `stderr`[27]). El signo `<` permite que un programa reciba el `stdin` desde un archivo en vez de la interacción con el usuario. Por ejemplo: `mail root < file`, invoca el comando `mail` con argumento (destinatario del mail) `root`, siendo el contenido del mensaje el contenido del archivo `file` en vez del texto que usualmente teclea el usuario. Más a menudo aparece la necesidad de almacenar en un archivo la salida de un comando. Para ello se emplea el signo `>`. Por ejemplo, `man bash > file`, invoca el comando `man` con argumento (información deseada) `bash` pero indicando que la información debe ser almacenada en el archivo `file` en vez de ser mostrada por pantalla.

En otras ocasiones uno desea que la salida de un programa sea la entrada de otro. Esto se logra empleando los denominados *pipes*, para ello se usa el signo `|`. Este signo permite que el `stdout` de un programa sea el `stdin` del siguiente. Por ejemplo:

```
zcat  manual.gz | more
```

Invoca la orden de descompresión de `zcat` y conduce el **flujo** de caracteres hacia el paginador `more`, de forma que podamos ver página a página el archivo descomprimido. A parte de los símbolos mencionados existen otros que permiten acciones tales como:

- `>>` Añadir el `stdout` al final del archivo indicado (*append*).[28]
- `>&` o `&>` (sólo `csh`, `tcsh` y `bash`) Redireccionar el `stdout` y `stderr`. Con `2>` redireccionó sólo el `stderr`.
- `>>&` Igual que `>&` pero en modo *append*.
- `>>!` Igual que `>>`, pero con la adición que funciona también cuando el archivo no existe.

8.5.3. Ejecución de comandos.

- Si el comando introducido es propio del *shell* (*built-in*), se ejecuta directamente.
- En caso contrario:
  - Si el comando contiene `/`, el *shell* lo considera un `PATH` e intenta resolverlo (entrar en cada directorio especificado para encontrar el comando).
  - En caso contrario el *shell* busca en una tabla *hash table* que contiene los nombres de los comandos que se han encontrado en los directorios especificados en la variable `PATH`, cuando ha arrancado el *shell*.

8.5.4. Aliases.

Para facilitar la entrada de algunas órdenes o realizar operaciones complejas, los *shells* interactivos permiten el uso de aliases. La orden `alias` permite ver qué aliases hay definidos

[27] Si estos mensajes son de error.

[28] En `bash`, si el archivo no existe, es creado.

y también definir nuevos. Es corriente definir el alias `rm ='rm -i'`, de esta forma la orden siempre pide confirmación para borrar un archivo. Si alguna vez quieres usar `rm` sin alias, sólo hace falta poner delante el símbolo \, denominado *backslash* . Por ejemplo `\rm` elimina los alias aplicados a `rm`. Otro ejemplo, bastante frecuente (en `tcsh/csh`) podría ser (debido a la complejidad de la orden): `alias ffind 'find . -name \!* -print'`. Para emplearlo: `ffind tema.txt`, el resultado es la búsqueda recursiva a partir del directorio actual de un archivo que se llame `tema.txt`, mostrando el camino hasta el mismo.

8.5.5. Las shells csh y tcsh.

Son dos de los Shells interactivos más empleados. Una de las principales ventajas de `tcsh` es que permite la edición de la línea de comandos, y el acceso a la historia de órdenes usando las teclas de cursores.[29]

Comandos propios.

Los comandos propios o intrínsecos, *Built-In Commands*, son aquéllos que proporciona el propio *shell* [30].

`alias name def`
Asigna el nombre `name` al comando `def`.

```
foreach var (wordlist)
   commands
end
```

La variable `var` se asigna sucesivamente a los valores de cadena `wordlist`, y se ejecuta el conjunto de comandos. El contenido de dicha variable puede ser empleado en los comandos: `$var`.

`history`
Muestra las últimas órdenes introducidas en el *shell*. Algunos comandos relacionados con el *Command history* son:

- `!!`
  Repite la última orden.
- `!n`
  Repite la orden n-ésima.
- `!string`
  Repite la orden más reciente que empiece por la cadena `string`.

[29] `bash` también lo permite.
[30] A diferencia de los comandos que provienen de un ejecutable situado en alguno de los directorios de la variable `PATH`.

- `!?string`
  Repite la orden más reciente que contenga la cadena `string`.

- `^str1^str2` o `!!:s/str1/str2/`
  (*substitute*) Repite la última orden reemplanzando la primera ocurrencia de la cadena `str1` por la cadena `str2`.

- `!!:gs/str1/str2/`
  (*global substitute*) Repite la última orden reemplazando todas las ocurrencias de la cadena `str1` por la cadena `str2`.

- `!$`
  Es el último argumento de la orden anterior que se haya tecleado.

`pushd`
Cambia de directorio, recordando el directorio actual.

`popd`
Retorna al directorio desde donde se hizo `pushd` la última vez.

`repeat count command`
Repite `count` veces el comando `command`.

`rehash`
Rehace la tabla de comandos (*hash table*).

`set variable = VALUE`
Asigna el valor de una variable del *shell*.

`set variable`
Muestra el valor de la variable

`setenv VARIABLE VALUE`
Permite asignar el valor de una variable de entorno.

`source file`
Ejecuta las órdenes del fichero `file` en el *shell* actual.

`unset variable`
Desasigna el valor de una variable del *shell*.

`unsetenv VARIABLE VALUE`
Permite desasignar el valor de una variable de entorno.

`umask value`
Asigna la máscara para los permisos por omisión.

`unalias name`
Elimina un alias asignado.

Variables propias del shell.

Existe un conjunto de variables denominadas *shell variables*, que permiten modificar el funcionamiento del *shell*.

`filec` (*FILE Completion*)

Es una variable *toggle* que permite que el *shell* complete automáticamente el nombre de un archivo o un directorio[31]. Para ello, si el usuario introduce sólo unos cuantos caracteres de un archivo y pulsa el `TAB`, el *shell* completa dicho nombre. Si sólo existe una posibilidad, el completado es total y el *shell* deja un espacio tras el nombre. En caso contrario hace sonar un pitido. Pulsando `Ctrl-D` el *shell* muestra las formas existentes para completar.

`prompt`

Es una variable de cadena que contiene el texto que aparece al principio de la línea de comandos.

`savehist`

Permite definir el número de órdenes que se desea almacenar al abandonar el *shell*. Esto permite recordar las órdenes que se ejecutaron en la sesión anterior.

8.5.6. Las shell sh y bash.

Sólo `bash` puede considerarse un *shell* interactivo, permitiendo la edición de la línea de comandos, y el acceso a la historia de órdenes (*readline*). En uso normal (historia y editor de línea de comandos) `BASH` es compatible con `TCSH` y `KSH`. El modo de completado (*file completion*) es automático (usando `TAB` sólo) si el *shell* es interactivo.

Comandos propios del shell.

Los comandos `umask`, `source`, `pushd`, `popd`, `history`, `unalias`, `hash` [32], funcionan igual que en la *shell* `TCSH`.

`help`

Ayuda interna sobre los comandos del *shell*.

`VARIABLE=VALUE`

Permite asignar el valor de una variable de entorno. Para que dicha variable sea "heredada" es necesario emplear: `export VARIABLE` o bien combinarlas: `export VARIABLE=VALUE`.

`for var in wordlist do comandos done`

La variable `var`, que puede llamarse de cualquier modo, se le asignan sucesivamente los valores de la cadena `wordlist`, y se ejecuta el conjunto de comandos. El contenido de dicha variable puede ser empleado en los comandos: `$var`. Ejemplo `{for i in 1 2 tres 4; do echo $i; done}` 1 2 tres 4

`alias`

[31] `bash` permite no sólo completar ficheros/directorios sino también comandos.

[32] En `bash/sh` la *hash table* se va generando dinámicamente a medida que el usuario va empleando las órdenes. Así el arranque del *shell* es más rápido, y el uso de orden equivalente `hash -r` casi nunca hace falta.

En `bash`, `alias` sólo sirve para substitución simple de una cadena por otra. Por ejemplo: `alias ls='ls -F'`. Para crear alias con argumentos se usan funciones, ver la documentación.

8.5.7. Archivos de *script*.

Un archivo de *script* es una sucesión de comandos de la *shell* que se ejecutan secuencialmente. Veamos un ejemplo simple:

```
#!/bin/bash
variable=''/home/yo''
cp $1 /tmp/$2
rm $1
cd $variable
# Hecho por mi
```

La primera línea declara la *shell* específica que se quiere usar. En la segunda línea hay una declaración de una variable interna. La tercera contiene los dos primeros argumentos con que fue llamado el *script*. Por ejemplo, si el anterior *script* está en un archivo llamado `ejemplo`, el comando `ejemplo file1 file2` asocia `$1` a `file1` y `$2` a `file2`. La línea 5 hace uso de la variable interna dentro de un comando. La última línea por comenzar con un `#` corresponde a un comentario. Notemos que la primera también es un comentario, pero la combinación `#!` en la primera línea fuerza a que se ejecute esa *shell*.

Esto sólo es una mínima pincelada de una herramienta muy poderosa y útil. Los comandos disponibles en la *shell* conforman un verdadero lenguaje de programación en sí, y los *scripts* pueden diseñarse para realizar tareas monótonas y complejas. Este es un tema que le será útil profundizar.

8.6. Ayuda y documentación.

Para obtener ayuda sobre comandos de UNIX, se puede emplear la ayuda *on-line*, en la forma de páginas de manual. Así `man comando` proporciona la ayuda sobre el `comando` deseado. Por ejemplo, para leer el manual de los shells, puedes entrar: `man sh csh tcsh bash` la orden formatea las páginas y te permite leer los manuales en el orden pedido. En el caso de `bash` se puede usar el comando `help`, por ejemplo, `help alias`. Además, para muchos comandos y programas se puede obtener información tipeando `info comando`. Finalmente, algunos comandos tienen una opción de ayuda (`--help`), para recordar rápidamente las opciones más comunes disponibles (`ls --help`).

8.7. Procesos.

En una máquina existen una multitud de procesos que pueden estar ejecutándose simultáneamente. La mayoría de ellos no corresponden a ninguna acción realizada por el usuario y no merecen que se les preste mayor atención. Estos procesos corresponden a programas ejecutados en el arranque del sistema y tienen que ver con el funcionamiento global del servidor. En general, los programas suelen tener uno de estos dos modos de ejecución:

- **foreground**: Son aquellos procesos que requieren de la interacción y/o atención del usuario mientras se están ejecutando, o bien en una de sus fases de ejecución (*i.e.* introducción de datos). Así por ejemplo, la consulta de una página de manual es un proceso que debe ejecutarse claramente en *foreground*.
- **background**: Son aquellos procesos que no requieren de la interacción con el usuario para su ejecución. Si bien el usuario desearía estar informado cuando este proceso termine. Un ejemplo de este caso sería la impresión de un archivo.

Sin embargo, esta división que a primera vista pueda parecer tan clara y concisa, a menudo en la práctica aparece la necesidad de conmutar de un modo al otro, detención de tareas indeseadas, etc. Así por ejemplo, puede darse el caso de que estemos leyendo una página de manual y de repente necesitemos ejecutar otra tarea. Un proceso viene caracterizado por:

- *process number*
- *job number*

Veamos algunas de las órdenes más frecuentes para la manipulación de procesos:

- `comando &` Ejecución de un comando en el *background*. [33]
- `Ctrl-Z` Detiene el proceso que estuviera ejecutándose en el *foreground* y lo coloca detenido en el *background*.
- `Ctrl-C` Termina un proceso que estaba ejecutándose en *foreground*.
- `Ctrl-\` Termina de forma definitiva un proceso que estaba ejecutándose en *foreground*.
- `ps x` Lista todos los procesos que pertenezcan al usuario, incluyendo los que no están asociados a un terminal.
- `jobs` Lista los procesos que se hayan ejecutado desde el *shell* actual, mostrando el *job number*.
- `fg (job number)` Pasa a ejecución en *foreground* un proceso que se hallase en *background*.
- `bg (job number)` Pasa a ejecución en *background* un proceso que se hallase detenido con `Ctrl-Z`.
- `kill (process number)` Envía una señal[34] a un proceso UNIX. En particular para enviar la señal de término a un programa, damos el comando `kill -KILL`, pero no hace falta al ser la señal por defecto.

Cuando se intenta abandonar una sesión con algún proceso aún detenido en el *background* del *shell*, se informa de ello con un mensaje del tipo: `There are stopped jobs` si no importa, el usuario puede intentar abandonar de nuevo el *shell* y éste matará los *jobs*, o puedes utilizar `fg` para traerlos al *foreground* y ahí terminar el mismo.

[33] Por omisión un comando se ejecuta siempre en el *foreground*.

[34] Para ver las señales disponibles entra la orden `kill -l` (l por *list*).

8.8. Editores.

Un editor es un programa que permite crear y/o modificar un archivo. Existen multitud de editores diferentes, y al igual que ocurre con los *shells*, cada usuario tiene alguno de su predilección. Mencionaremos algunos de los más conocidos:

- **vi** - El editor standard de UNIX.
- **emacs (xemacs)** - Editor muy configurable escrito en lenguaje Lisp. Existen multitud de modos para este editor (lector de mail, news, www,...) que lo convierten en un verdadero *shell* para multitud de usuarios. Las últimas versiones del mismo permiten la ejecución desde X-windows o terminal indistintamente con el mismo binario. Posee un tutorial en línea, comando `C-H t` dentro del editor. El archivo de configuración personalizada es: `$HOME/.emacs`.
- **jove** - Basado en Emacs, (Jonathan's Own Version of Emacs). Posee tutorial en una utilidad asociada: `teachjove`. El archivo de configuración personalizada es: `$HOME/.joverc`.
- **jed** - Editor configurable escrito en S-Lang. Permite la emulación de editores como emacs y Wordstar. Posee una ayuda en línea `C-H C-H`. El archivo de configuración personalizada es: `$HOME/.jedrc`.
- **gedit** - Editor por defecto de gnome.
- **xjed** - Versión de jed para el X-windows system. Presenta como ventaja que es capaz de funcionar en muchos modos: lenguaje C, Fortran, TeX, etc, reconociendo palabras clave y signos de puntuación, empleando un colorido distinto para ellos. El archivo de configuración personalizada es el mismo que el de de jed.

Dado que los editor del tipo de **gedit** disponen de menús auto explicativos, daremos a continuación unas ligeras nociones sólo de `vi` y `emacs`.

8.8.1. El editor vi.

El `vi` es un editor de texto muy poderoso pero un poco difícil de usar. Lo importante de este editor es que se puede encontrar en cualquier sistema UNIX y sólo hay unas pocas diferencias entre un sistema y otro. Explicaremos lo básico solamente. Comencemos con el comando para invocarlo:

```
localhost:/# vi
```

```
~
~
~
/tmp/vi.9Xdrxi: new file: line 1
```

La sintaxis para editar un archivo es:
`localhost:/# vi nombre.de.archivo`

```
~
~
~
nombre.de.archivo: new file: line 1
```

Insertar y borrar texto en vi.

Cuando se inicia el `vi`, editando un archivo, o no, se entra en un modo de órdenes, es decir, que no se puede empezar a escribir directamente. Si se quiere entrar en modo de inserción de texto se debe presionar la tecla `i`. Entrando en el modo de inserción, se puede empezar a escribir. Para salir del modo de inserción de texto y volver al modo de órdenes se aprieta `ESC`.

```
Aqui ya estamos escribiendo porque apretamos
la tecla 'i' al estar en modo ordenes.
~
~
```

La tecla `a` en el modo de órdenes también entra en modo de inserción de texto, pero en vez de comenzar a escribir en la posición del cursor, empieza un espacio después.

La tecla `o` en el modo de órdenes inserta texto pero desde la línea que sigue a la línea donde se está ubicado.

Para borrar texto, hay que salir al modo órdenes, y presionar la tecla `x` que borrará el texto que se encuentre sobre el cursor. Si se quiere borrar las líneas enteras, entonces se debe presionar dos veces la tecla `d` sobre la línea que deseo eliminar. Si se presionan las teclas `dw` se borra la palabra sobre la que se está ubicado.

La letra `R` sobre una palabra se puede escribir encima de ella. Esto es una especie de modo de inserción de texto pero sólo se podrá modificar la palabra sobre la que se está situado. La tecla `~` cambia de mayúscula a minúscula la letra sobre la que se está situado.

Moverse dentro de vi.

Estando en modo ordenes podemos movernos por el archivo que se está editando usando las flechas hacia la izquierda, derecha, abajo o arriba. Con la tecla `0` nos movemos al comienzo de la línea y con la tecla `$` nos movemos al final de la misma.

Con las teclas `w` y `b` nos movemos al comienzo de la siguiente palabra o al de la palabra anterior respectivamente. Para moverme hacia la pantalla siguiente la combinación de teclas `CTRL F` y para volver a la pantalla anterior `CTRL B`. Para ir hasta el principio del archivo se presiona la tecla `G`.

Opciones de comandos.

Para entrar al menú de comandos se debe presionar la tecla : en el modo de órdenes. Aparecerán los dos puntos (:). Aquí se pueden ingresar ordenes para guardar, salir, cambiar de archivo entre otras cosas. Veamos algunos ejemplos:

- `:w` Guardar los cambios.
- `:w otherfile.txt` Guardar con el nuevo nombre `otherfile.txt`
- `:wq` Guardar los cambios y salir.
- `:q!` Salir del archivo sin guardar los cambios.
- `:e file1.txt` Si deseo editar otro archivo al que se le pondrá por nombre `file1.txt`.
- `:r file.txt` Si se quiere insertar un archivo que ya existente, por ejemplo `file.txt`.
- `:r! comando` Si se quiere ejecutar algún comando del *shell* y que su salida aparezca en el archivo que se está editando.

8.8.2. Editores modo emacs.

El editor `GNU Emacs`, escrito por Richard Stallman de la *Free Software Foundation*, es uno de los que tienen mayor aceptación entre los usuarios de UNIX, estando disponible bajo licencia **GNU GPL**[35] para una gran cantidad de arquitecturas. También existe otra versión de emacs llamada `XEmacs` totalmente compatible con la anterior pero presentando mejoras significativas respecto al `GNU Emacs`. Dentro de los "inconvenientes" que presenta es que no viene por defecto incluido en la mayoría de los sistemas UNIX. Las actuales distribuciones de Linux y en particular Debian GNU/Linux contienen ambas versiones de emacs, tanto `GNU Emacs` como `XEmacs`, como también versiones de `jove`, `jed`, `xjed` y muchos otros editores.

Los editores tipo emacs se parecen mucho y en su mayoría sus comandos son los mismos. Para ejemplificar este tipo de editores nos centraremos en `XEmacs`, pero los comandos y descripciones se aplican casi por igual a todos ellos. Los editores tipo emacs constan de tres zonas:

- La zona de edición: donde aparece el texto que está siendo editado y que ocupa la mayor parte de la pantalla.
- La zona de información: es una barra que esta situada en la penúltima línea de la pantalla.
- La zona de introducción de datos: es la última línea de la pantalla.

[35] La licencia de GNU, da el permiso de libre uso de los programas con su fuentes, pero los autores mantienen el *Copyright* y no es permitido distribuir los binarios sin acceso a sus fuentes, los programas derivados de dichos fuentes heredan la licencia GNU.

Emacs es un editor que permite la edición visual de un archivo (en contraste con el modo de edición de `vi`). El texto se agrega o modifica en la zona de edición, usando las teclas disponibles en el teclado.

Además, existen una serie de comandos disponibles para asistir en esta tarea.

La mayoría de los comandos de emacs se realizan empleando la tecla de `CONTROL` o la tecla `META`[36]. Emplearemos la nomenclatura: `C-key` para indicar que la tecla `key` debe de ser pulsada junto con `CONTROL` y `M-key` para indicar que la tecla `META` debe de ser pulsada junto a `key`. En este último caso NO es necesario pulsar simultáneamente las teclas `ESC` y `key`, pudiendo pulsarse secuencialmente `ESC` y luego `key`, sin embargo, si se usa `ALT` como `META` deben ser pulsadas simultáneamente. Observemos que en un teclado normal hay unos 50 caracteres (letras y números). Usando `SHIFT` se agregan otros 50. Así, usando `CONTROL` y `META`, hay unos $50 \cdot 4 = 200$ comandos disponibles. Además, existen comandos especiales llamados *prefijos*, que modifican el comando siguiente. Por ejemplo, `C-x` es un prefijo, y si `C-s` es un comando (de búsqueda en este caso), `C-x C-s` es otro (grabar archivo). Así, a través de un prefijo, se duplican el número de comandos disponibles sólo con el teclado, hasta llegar a unos $200 \cdot 2 = 400$ comandos en total.

Aparte de estos comandos accesibles por teclas, algunos de los cuales comentaremos a continuación, existen comandos que es posible ejecutar por nombre, haciendo así el número de comandos disponibles virtualmente infinito.

Revisemos los comandos más usuales, ordenados por tópico.

Abortar y deshacer

En cualquier momento, es posible abortar la operación en curso, o deshacer un comando indeseado:

C-g	abortar
C-x u	deshacer

Archivos

C-x C-f	cargar archivo
C-x i	insertar archivo
C-x C-s	grabar archivo
C-x C-w	grabar con nombre
C-x C-c	salir

Ventanas

`Emacs` permite dividir la pantalla en varias ventanas. En cada ventana se puede editar texto e ingresar comandos independientemente. Esto es útil en dos situaciones: a) si necesitamos editar un solo archivo, pero necesitamos ver su contenido en dos posiciones distintas

[36]Dado que la mayoría de los teclados actuales no poseen la tecla `META` se emplea ya sea `ESC` o `ALT`.

(por ejemplo, el comienzo y el final de archivos muy grandes); y b) si necesitamos editar o ver varios archivos simultáneamente. Naturalmente, aunque son independientes, sólo es posible editar un archivo a la vez. A la ventana en la cual se encuentra el cursor en un momento dado le llamamos la "ventana actual".

C-x 2	dividir ventana actual en 2 partes, con línea horizontal
C-x 3	dividir ventana actual en 2 partes, con línea vertical
C-x 1	sólo 1 ventana (la ventana actual, eliminando las otras)
C-x 0	elimina sólo la ventana actual
C-x o	cambia el cursor a la siguiente ventana

El cambio del cursor a una ventana cualquiera se puede hacer también rápidamente a través del *mouse*.

Comandos de movimiento

Algunos de estos comandos tienen dos teclas asociadas, como se indica a continuación.

C-b o ←	izquierda un carácter	**C-f** o →	derecha un carácter
C-p o ↑	arriba una línea	**C-n** o ↓	abajo una línea
C-a o `Home`	principio de la línea	**C-e** o `End`	fin de la línea
M-< o **C-**`Home`	principio del documento	**M->** o **C-**`End`	fin del documento
M-f o **M-**→	avanza una palabra	**M-b** o **M-**←	retrocede una palabra
C-v o `Page Up`	avanza una página	**M-v** o `Page Down`	retrocede una página
M-g (número)	salta a la línea (número)	**C-l**	refresca la pantalla

Comandos de inserción y borrado

Al ser un editor en modo visual, las modificaciones se pueden hacer en el texto sin necesidad de entrar en ningún modo especial.

C-d o `Delete`	borra un carácter después del cursor
`Backspace`	borra un carácter antes del cursor
C-k	borra desde la posición del cursor hasta el fin de línea (no incluye el cambio de línea)
M-d	borra desde el cursor hacia adelante, hasta que termina una palabra
M-`Backspace`	borra desde el cursor hacia atrás, hasta que comienza una palabra
C-o	Inserta una línea en la posición del cursor

Mayúsculas y minúsculas

M-u	Cambia a mayúscula desde la posición del cursor hasta el fin de la palabra
M-l	Cambia a minúscula desde la posición del cursor hasta el fin de la palabra
M-c	Cambia a mayúscula el carácter en la posición del cursor y a minúscula hasta el fin de la palabra

Por ejemplo, veamos el efecto de cada uno de estos comandos sobre la palabra `EmAcS`, si el cursor está sobre la letra `E` (¡el efecto es distinto si está sobre cualquier otra letra!):

M-u : `EmAcS` ⟶ `EMACS`
M-l : `EmAcS` ⟶ `emacs`
M-c : `EmAcS` ⟶ `Emacs`

Transposición

Los siguientes comandos toman como referencia la posición actual del cursor. Por ejemplo, **C-t** intercambia el carácter justo antes del cursor con el carácter justo después.

C-t	Transpone dos caracteres
M-t	Transpone dos palabras
C-x C-t	Transpone dos líneas

Búsqueda y reemplazo

C-s	Búsqueda hacia el fin del texto
C-r	Búsqueda hacia el inicio del texto
M- %	Búsqueda y sustitución (pide confirmación cada vez)
M-&	Búsqueda y sustitución (sin confirmación)

Definición de regiones y reemplazo

Uno de los conceptos importantes en `emacs` es el de región. Para ello, necesitamos dos conceptos auxiliares: el *punto* y la *marca*. El punto es simplemente el cursor. Específicamente, es el punto donde *comienza* el cursor. Así, si el cursor se encuentra sobre la letra `c` en `emacs`, el punto está entre la `a` y la `c`. La marca, por su parte, es una señal que se coloca en algún punto del archivo con los comandos apropiados. La *región* es el espacio comprendido entre el punto y la marca.

Para colocar una marca basta ubicar el cursor en el lugar deseado, y teclear **C-**`Space` o **C-@**. Esto coloca la marca donde está el punto (en el ejemplo del párrafo anterior, quedaría entre las letras `a` y `c`. Una vez colocada la marca, podemos mover el cursor a cualquier otro lugar del archivo (hacia atrás o hacia adelante respecto a la marca). Esto define una cierta ubicación para el punto, y, por tanto, queda definida la región automáticamente.

La región es una porción del archivo que se puede manipular como un todo. Una región se puede borrar, copiar, pegar en otro punto del archivo o incluso en otro archivo; una región se puede imprimir, grabar como un archivo distinto; etc. Así, muchas operaciones importantes se pueden efectuar sobre un bloque del archivo.

Por ejemplo, si queremos duplicar una región, basta con definir la región deseada (poniendo la marca y el punto donde corresponda) y teclear **M-w**. Esto copia la región a un buffer temporal (llamado *kill buffer*). Luego movemos el cursor al lugar donde queremos insertar el texto duplicado, y hacemos **C-y**. Este comando toma el contenido del *kill buffer* y lo inserta en el archivo. El resultado final es que hemos duplicado una cierta porción del texto.

Si la intención era mover dicha porción, el procedimiento es el mismo, pero con el comando **C-w** en vez de **M-w**. **C-w** también copia la región a un *kill buffer*, pero borra el texto de la pantalla.

Resumiendo:

C-`Space` o **C-@**	Comienzo de región
M-w	Copia región
C-w	Corta región
C-y	Pega región

El concepto de *kill buffer* es mucho más poderoso que lo explicado recién. En realidad, muchos comandos, no sólo **M-w** y **C-w**, copian texto en un *kill buffer*. En general, cualquier comando que borre más de un carácter a la vez, lo hace. Por ejemplo, **C-k** borra una línea. Lo que hace no es sólo borrarla, sino además copiarla en un *kill buffer*. Lo mismo ocurre con los comandos que borran palabras completas (**M-d**, **M-**`Backspace`), y muchos otros. Lo interesante es que **C-y** funciona también en todos esos casos: **C-y** lo único que hace es tomar el último texto colocado en un *kill buffer* (resultado de la última operación que borró más de un carácter a la vez), y lo coloca en el archivo. Por lo tanto, no sólo podemos copiar o mover "regiones", sino también palabras o líneas. Más aún, el *kill buffer* no es borrado con el **C-y**, así que ese mismo texto puede ser duplicado muchas veces. Continuará disponible con **C-y** mientras no se ponga un nuevo texto en el *kill buffer*.

Además, `emacs` dispone no de uno sino de muchos *kill buffers*. Esto permite recuperar texto borrado hace mucho rato. En efecto, cada vez que se borra más de un carácter de una vez, se una un nuevo *kill buffer*. Por ejemplo, consideremos el texto:

```
La primera linea del texto,
la segunda linea,
y finalmente la tercera.
```

Si en este párrafo borramos la primera línea (con **C-k**), después borramos la primera palabra de la segunda (con **M-d**, por ejemplo), y luego la segunda palabra de la última, entonces habrá tres *kill buffers* ocupados:

buffer 1 : `La primera linea del texto,`
buffer 2 : `la`
buffer 3 : `finalmente`

Al colocar el cursor después del punto final, **C-y** toma el contenido del último *kill buffer* y lo coloca en el texto:

```
 segunda linea,
y  la tercera. finalmente
```

Si se teclea ahora **M-y**, el último texto recuperado, `finalmente`, es reemplazado por el penúltimo texto borrado, y que está en el *kill buffer* anterior:

```
 segunda linea,
y  la tercera. la
```

Además, la posición de los *kill buffers* se rota:

```
buffer 1  :  finalmente
buffer 2  :  La primera linea del texto,
buffer 3  :  la
```

Sucesivas aplicaciones de **M-y** después de un **C-y** rotan sobre todos los *kill buffers* (que pueden ser muchos). El editor, así, conserva un conjunto de las últimas zonas borradas durante la edición, pudiendo recuperarse una antigua a pesar de haber seleccionado una nueva zona, o borrado una nueva palabra o línea. Toda la información en los *kill buffers* se pierde al salir de `emacs` (**C-c**).

Resumimos entonces los comandos para manejo de los *kill buffers*:

C-y	Copia el contenido del último *kill buffer* ocupado
M-y	Rota los *kill buffers* ocupados

Definición de macros

La clave de la configurabilidad de `emacs` está en la posibilidad de definir nuevos comandos que modifiquen su comportamiento o agreguen nuevas funciones de acuerdo a nuestras necesidades. Un modo de hacerlo es a través del archivo de configuración `$HOME/.emacs`, para lo cual se sugiere leer la documentación disponible en la distribución instalada. Sin embargo, si sólo necesitamos un nuevo comando en la sesión de trabajo actual, un modo más simple es definir una *macro*, un conjunto de órdenes que son ejecutados como un solo comando. Los comandos relevantes son:

C-x (	Comienza la definición de una macro
C-x)	Termina la definición de una macro
C-x e	Ejecuta una macro definida

Todas las sucesiones de teclas y comandos dados entre **C-x (** y **C-x)** son recordados por `emacs`, y después pueden ser ejecutados de una vez con **C-x e**.

Como ejemplo, consideremos el siguiente texto, con los cinco primeros lugares del ránking ATP (sistema de entrada) al 26 de marzo de 2002:

```
1 hewitt, lleyton (Aus)
2 kuerten, gustavo (Bra)
3 ferrero, juan (Esp)
4 kafelnikov, yevgeny (Rus)
5 haas, tommy (Ger)
```

Supongamos que queremos: (a) poner los nombres y apellidos con mayúscula (como debería ser); (b) poner las siglas de países sólo en mayúsculas.

Para definir una macro, colocamos el cursor al comienzo de la primera línea, en el `1`, y damos **C-x (**. Ahora realizamos todos los comandos necesarios para hacer las tres tareas solicitadas para el primer jugador solamente: **M-f** (avanza una palabra, hasta el espacio antes de `hewitt`; **M-c M-c** (cambia a `Hewitt, Lleyton`); **M-u** (cambia a `AUS`); `Home` (vuelve el cursor al comienzo de la línea); ↓ (coloca el cursor al comienzo de la línea siguiente, en el `2`). Los dos últimos pasos son importantes, porque dejan el cursor en la posición correcta para ejecutar el comando nuevamente. Ahora terminamos la definición con **C-x)**. Listo. Si ahora ejecutamos la macro, con **C-x e**, veremos que la segunda línea queda modificada igual que la primera, y así podemos continuar hasta el final:

```
1 Hewitt, Lleyton (AUS)
2 Kuerten, Gustavo (BRA)
3 Ferrero, Juan (ESP)
4 Kafelnikov, Yevgeny (RUS)
5 Haas, Tommy (GER)
```

Comandos por nombre

Aparte de los ya comentados existen muchas otras órdenes que no tienen necesariamente una tecla asociada (*bindkey*) asociada. Para su ejecución debe de teclearse previamente:

`M-x`

y a continuación en la zona inferior de la pantalla se introduce el comando deseado. Empleando el `TAB` se puede completar dicho comando (igual que en `bash`).

De hecho, esto sirve para cualquier comando, incluso si tiene tecla asociada. Por ejemplo, ya sabemos **M-g** `n` va a la línea n del documento. Pero esto no es sino el comando `goto-line`, y se puede también ejecutar tecleando: `M-x goto-line n`.

Repetición

Todos los comandos de `emacs`, tanto los que tienen una tecla asociada como los que se ejecutan con nombre, se pueden ejecutar más de una vez, anteponiéndoles un argumento numérico con

```
M-(number)
```

Por ejemplo, si deseamos escribir 20 letras `e`, basta teclear `M-20 e`. Esto es particularmente útil con las macros definidos por el usuario. En el ejemplo anterior, con el ránking ATP, después de definir la macro quedamos en la línea 2, y en vez de ejecutar **C-x e** 4 veces, podemos teclear `M-4 C-x e`, con el mismo resultado, pero en mucho menos tiempo.

Para terminar la discusión de este editor, diremos que es conveniente conocer las secuencias de control básico de `emacs`:

`C-a, C-e, C-k, C-y, C-w, C-t, C-d`, etc.,

porque funcionan para editar la línea de comandos en el *shell*, como también en muchos programas de texto y en ventanas de diálogo de las aplicaciones *X Windows*. A su vez, los editores `jed`, `xjed`, `jove` también usan por defecto estas combinaciones.

8.9. El sistema X Windows.

El *X Windows system* es el sistema estándar de ventanas en las estaciones de trabajo. Es corriente que el sistema de ventanas sea arrancando automáticamente cuando la máquina parte. En caso contrario, la orden para arrancarlo es `startx`. En el sistema *X Windows* deben distinguirse dos conceptos:

- **server** : Es un programa que se encarga de escribir en el dispositivo de vídeo y de capturar las entradas (por teclado, ratón, etc). Asimismo se encarga de mantener los recursos y preferencias de las aplicaciones. Sólo puede existir un server para cada pantalla.
- **client** : Es cualquier aplicación que se ejecute en el sistema *X Windows*. No hay límite (en principio) en el número de clientes que pueden estarse ejecutando simultáneamente. Los clientes pueden ser locales o remotos.

.
Window Manager (WM) Es un cliente con "privilegios especiales": controla el comportamiento (forma, tamaño,...) del resto de clientes. Existen varios, destacando:

- **fvwm** : *F* Virtual Window Manager*, el instalado por defecto.
- **icewm** : *Ice Window Manager*, uno de los *window managers* gnome compatible.
- **sawfish** : *Window managers* gnome compatible, altamente configurable y muy integrado al gnome *desktop*.
- **Metacity** : *Window managers* gnome 2 compatible.

El *look and feel* (o GUI) de *X Windows* es extremadamente configurable, y puede parecer que dos máquinas son muy distintas, pero esto se debe al `WM` que se esté usando y no a que las aplicaciones sean distintas.

Para configurar tu sesión es necesario saber qué programas estás usando y ver las páginas de manual. Los archivos principales son:

- `.xinitrc` ó `.xsession` archivo leído al arrancar *X Windows*. Aquí se pueden definir los programas que aparecen al inicio de tu sesión.
- `.fvwmrc` archivo de configuración del `fvwm`. Ver las páginas del manual de `fvwm`.
- `.olwmrc` archivo de configuración del `olwm`. Ver las páginas del manual de `olwm`.
- `.Xdefaults` Configuración general de las aplicaciones de X Windows. Aquí puedes definir los *resources* que encontrarás en los manuales de las aplicaciones de X.

En caso de que tengas que correr una aplicación de X que no esté disponible en la máquina que estás usando, eso no representa ningún problema. Las órdenes necesarias son (por ejemplo, para arrancar un `gnome-terminal` remoto):

```
userA@hostname1:~$ xhost +hostname2
hostname2 being added to access control list
user@hostname1:~$ ssh userB@hostname2
userB@hostname2's password:
userB@hostname2:~$ export DISPLAY=hostname1:0
userB@hostname2:~$ gnome-terminal &
```

Si todo está previamente configurado, es posible que no haga falta dar el *password*.

Cuando quieres salir, normalmente puedes encontrar un icono con la opción `Log out`, en un menú o panel de la pantalla.

8.10. Uso del ratón.

El ratón es un dispositivo esencial en el uso de programas X, sin embargo, la función que realiza en cada uno de ellos no está normalizada.

Comentaremos la pauta seguida por la mayoría de las aplicaciones, pero debe tenerse presente que es muy frecuente encontrar aplicaciones que no las respetan.[37]

- **Botón izquierdo** (LB): Seleccionar. Comienza el bloque de selección.
- **Botón central** (MB): Pegar. Copia la selección en la posición del cursor.
- **Botón derecho** (RB): Habitualmente ofrece un menú para partir aplicaciones.

Existen dos modos para determinar cuál es la **ventana activa**, aquella que recibe las entradas de teclado:

- *Focus Follows Mouse*: La ventana que contenga al ratón es la que es activa. No usado por defecto actualmente.
- *Click To Focus*: La ventana seleccionada es la activa. El modo que esté activo depende de la configuración del *Window Manager*.

8.11. Internet.

En esta sección denominaremos `unix1` a la máquina local (desde donde ejecutamos la orden) y `unix2` a la máquina remota (con la que interaccionamos). Ambos son los `hostnames` de las respectivas máquinas. Existen algunos conceptos que previamente debemos comentar:

- `IP-number`: es un conjunto de 4 números separados por puntos (p.e. 146.83.57.34) que se asocia a cada máquina. No puede haber dos máquinas conectadas en la misma red con el mismo número.

[37] Las aplicaciones que son conscientes de un uso anormal y están realizadas por programadores inteligentes, muestran en pantalla la función de cada botón cuando son posibles varias alternativas.

- `hostname`: es el nombre que tiene asociada la máquina (p.e. `macul`). A este nombre se le suelen añadir una serie de sufijos separados por puntos que constituye el denominado dominio (p.e. `macul.ciencias.uchile.cl`). Una máquina por tanto puede tener más de un nombre reconocido (se habla en este caso de alias). Se denomina resolución a la identificación entre un `hostname` y el `IP-number` correspondiente. La consulta se realiza inicialmente en el archivo `/etc/hosts`, donde normalmente se guardan las identificaciones de las máquinas más comúnmente empleadas. En caso de que no se logre se accede al servicio `DNS` (*Domain Name Service*), que permite la identificación (resolución) entre un `hostname` y un `IP-number`.

- `mail-address`: es el nombre que se emplea para enviar correo electrónico. Este nombre puede coincidir con el nombre de una máquina, pero se suele definir como un alias, con objeto de que la dirección no deba de cambiarse si la máquina se estropea o se cambia por otra.

8.11.1. Acceso a la red.

Existen muchos programas para la conexión de la red, los más usados son:

- `telnet unix2`, hace un login en la máquina `unix2`, debe ingresarse el usuario y su respectiva `passwd`. Además, permite especificar el puerto en conexión en la máquina remota.

- `ssh nombre@unix2`, muy similar a `telnet` pero se puede especificar el usuario, si no se especifica se usa el nombre de la cuenta local. Además, el `passwd` pasa encriptado a través de la red. `ssh nombre@unix2 comando`, muy similar a `rsh`, el `passwd` pasa encriptado y ejecuta el comando en la máquina remota, mostrando el resultado en la máquina local.

- `scp file1 usuario2@unix2:path/file`, copia el archivo `file1`, del usuario1, que se encuentra en el directorio local en la máquina `unix1` en la cuenta del `usuario2` en la máquina `unix2` en `$HOME/path/file`. Si no se especifica el nombre del usuario se usa el nombre de la cuenta local. Si se quiere copiar el archivo `file2` del `usuario3` en `unix2` en la cuenta actual de `unix1` el comando sería: `scp usuario3@unix2:file2 .`. Antes de realizar cualquiera de las copias el sistema preguntará por el `passwd` del usuario en cuestión en la máquina `unix2`. Nuevamente, el `passwd` pasa encriptada a través de la red.

- `talk usuario1@unix2`, intenta hacer una conexión para hablar con el `usuario1` en la máquina `unix2`. Existen varias versiones de `talk` en los diferentes sistemas operativos, de forma que no siempre es posible establecer una comunicación entre máquinas con sistemas operativos diferentes.

- `ftp unix2`, (file transfer protocol) aplicación para copiar archivos entre máquinas de una red. `ftp` exige un nombre de cuenta y password para la máquina remota. Algunas de las opciones más empleadas (una vez establecida la conexión) son:

- `bin`: Establece el modo de comunicación binario. Es decir, transfiere una imagen exacta del archivo.
- `asc`: Establece el modo de comunicación `ascii`. Realiza las conversiones necesarias entre las dos máquinas en comunicación. Es el modo por defecto.
- `cd`: Cambia directorio en la máquina remota.
- `lcd`: Cambia directorio en la máquina local.
- `ls`: Lista el directorio remoto.
- `!ls`: Lista el directorio local.
- `prompt` : No pide confirmación para transferencia múltiple de archivos.
- `get rfile [lfile]`: transfiere el archivo `rfile` de la máquina remota a la máquina local denominándolo `lfile`. En caso de no suministrarse el segundo argumento supone igual nombre en ambas máquinas.
- `put lfile [rfile]` : transfiere el archivo `lfile` de la máquina local a la máquina remota denominándolo `rfile`. En caso de no suministrarse el segundo argumento supone igual nombre en ambas máquinas. También puede usarse `send`.
- `mget rfile` : igual que `get`, pero con más de un archivo (`rfile` puede contener caracteres comodines).
- `mput lfile` : igual que `put`, pero con más de un archivo (`lfile` puede contener caracteres comodines).

Existen versiones mejoradas de `ftp` con muchas más posibilidades, por ejemplo, `ncftp`. También existen versiones gráficas de clientes `ftp` donde la elección de archivo, el sentido de la transferencia y el modo de ésta, se elige con el *mouse* (p.e. `wxftp`).

- `rlogin -l nombre unix2`, (*remote login*), hace un `login` a la máquina `unix2` como el usuario `nombre` por defecto, sin los argumentos `-l nombre` `rlogin` usa el nombre de la cuenta local. Normalmente `rlogin` pide el *password* de la cuenta remota, pero con el uso del archivo `.rhosts` o `/etc/hosts.equiv` esto no es siempre necesario.
- `rsh -l nombre unix2 orden`, (*remote shell*), ejecuta la orden en la máquina `unix2` como usuario `nombre`. Es necesario que pueda entrar en la máquina remota sin *password* para ejecutar una orden remota. Sin especificar orden actúa como `rlogin`.

8.11.2. El correo electrónico.

El correo electrónico (`e-mail`) es un servicio para el envío de mensajes entre usuarios, tanto de la misma máquina como de diferentes máquinas.

Direcciones de correo electrónico.

Para mandar un `e-mail` es necesario conocer la dirección del destinatario. Esta dirección consta de dos campos que se combinan intercalando entre ellos el @ (*at*): `user@domain`

- `user` : es la identificación del usuario (*i.e.* `login`) en la máquina remota.
- `domain` : es la máquina donde recibe correo el destinatario. A menudo, es frecuente que si una persona tiene acceso a un conjunto de máquinas, su dirección de correo no corresponda con una máquina sino que corresponda a un alias que se resolverá en un nombre específico de máquina en forma oculta para el que envía.

Si el usuario es local no es necesario colocar el campo `domain` (ni tampoco el @).

Nomenclatura.

Veamos algunos conceptos relacionados con el correo electrónico:

- **Subject** : Es una parte de un mensaje que piden los programas al comienzo y sirve como título para el mensaje.
- **Cc** (Carbon Copy) : Permite el envío de copias del mensaje que está siendo editado a terceras personas.
- **Reply** : Cuando se envía un mensaje en respuesta a otro se suele añadir el comienzo del *subject:* `Re:`, con objeto de orientar al destinatario sobre el tema que se responde. Es frecuente que se incluya el mensaje al que se responde para facilitar al destinatario la comprensión de la respuesta.
- **Forward** : Permite el envío de un mensaje (con modificaciones o sin ellas) a una tercera persona.
- **Forwarding Mail** : Permite a un usuario que disponga de cuentas en varias máquinas no relacionadas, de concentrar su correo en una cuenta única[38]. Para ello basta con tener un archivo `$HOME/.forward` que contenga la dirección donde desea centralizar su correo.
- **Mail group** : Un grupo de correo es un conjunto de usuarios que reciben el correo dirigido a su grupo. Existen órdenes para responder a un determinado correo recibido por esa vía de forma que el resto del grupo sepa lo que ha respondido un miembro del mismo.
- **In-Box** : Es el archivo donde se almacena el correo que todavía no ha sido leído por el usuario. Suele estar localizado en `/var/spool/mail/user`.
- **Mailer-Daemon** : Cuando existe un problema en la transmisión de un mensaje se recibe un mensaje proveniente del *Mailer-Daemon* que indica el problema que se ha presentado.

[38]Este comando debe usarse con conocimiento pues en caso contrario podría provocar un *loop* indefinido y no recibir nunca correo.

Aplicación mail.

Es posiblemente la aplicación más simple. Para la lectura de mail teclear simplemente: `mail` y a continuación aparece un índice con los diferentes mensajes recibidos. Cada mensaje tiene una línea de identificación con número. Para leer un mensaje basta teclear su número y a continuación `return`. Para enviar un mensaje: `mail (address)` se pregunta por el `Subject:` y a continuación se introduce el mensaje. Para acabar se teclea sólo un punto en una línea o bien `Ctr-D`. Por último, se pregunta por `Cc:`. Es posible personalizar el funcionamiento mediante el archivo `$HOME/.mailrc`. Para enviar un archivo de texto a través del correo se suele emplear la redirección de entrada: `mail (address) < file`.

8.11.3. Ftp anonymous.

Existen servidores que permiten el acceso por `ftp` a usuarios que no disponen de cuenta en dichas máquinas. Para ello se emplea como `login` de entrada el usuario `anonymous` y como `passwd` la dirección de *e-mail* personal. Existen servidores que no aceptan conexiones desde máquinas que no están declaradas correctamente en el servicio de nombre (*dns*), así como algunas que no permiten la entrada a usuarios que no se identifican correctamente. Dada la sobrecarga que existe, muchos de los servidores tienen limitado el número de usuarios que pueden acceder simultáneamente.

8.11.4. WWW.

WWW son las siglas de *World-Wide Web*. Este servicio permite el acceso a información entrelazada (dispone de un texto donde un término puede conducir a otro texto): *hyperlinks*. Los archivos están realizados en un lenguaje denominado *html*. Para acceder a este servicio es necesario disponer de un lector de dicho lenguaje conocido como *browser* o navegador. Destacan actualmente: Netscape, Mozilla, Opera y el simple pero muy rápido Lynx.

8.12. Impresión.

Cuando se quiere obtener una copia impresa de un archivo se emplea el comando `lpr`.

`lpr file` - Envía el archivo `file` a la cola de impresión por defecto. Si la cola está activada, la impresora lista y ningún trabajo por encima del enviado, nuestro trabajo será procesado de forma automática.

A menudo existen varias posibles impresoras a las que poder enviar los trabajos. Para seleccionar una impresora en concreto (en vez de la por defecto) se emplea el modificador: `lpr -Pimpresora`, siendo `impresora` el nombre lógico asignado a esta otra impresora. Para recibir una lista de las posibles impresoras de un sistema, así como su estado, se puede emplear el comando `/usr/sbin/lpc status`. La lista de impresoras y su configuración también está disponible en el archivo `/etc/printcap`.

Otras órdenes para la manipulación de la cola de impresión son:

- `lpq [-Pprinter]`, permite examinar el estado de una determinada cola (para ver la cantidad de trabajos sin procesar de ésta, por ejemplo).

- `lprm [-Pprinter] jobnumber`, permite eliminar un trabajo de la cola de impresión.

Uno de los lenguajes de impresión gráfica más extendidos en la actualidad es *PostScript*. La extensión de los archivos *PostScript* empleada es `.ps`. Un archivo *PostScript* puede ser visualizado e impreso mediante los programas: `gv`, `gnome-gv` o `ghostview`. Por ello muchas de las impresoras actuales sólo admiten la impresión en dicho formato.

En caso de desear imprimir un archivo `ascii` deberá previamente realizarse la conversión a *PostScript* empleando la orden `a2ps`: `a2ps file.txt` Esta orden envía a la impresora el archivo `ascii file.txt` formateado a 2 páginas por hoja. Otro programa que permite convertir un archivo `ascii` en postscript es `enscript`.

Otro tipo de archivos ampliamente difundido y que habitualmente se necesita imprimir es el conocido como *Portable Document Format*. Este tipo de archivo posee una extensión `.pdf` y pueden ser visualizados e impresos usando aplicaciones tales como: `acroread, gv` o `xpdf`.

8.13. Compresión.

A menudo necesitamos comprimir un archivo para disminuir su tamaño, o bien crear un respaldo (*backup*) de una determinada estructura de directorios. Se comentan a continuación una serie de comandos que permiten ejecutar dichas acciones:

El compresor `compress` esta relativamente fuera de uso, pero es la estándar de UNIX.

- `compress file` : comprime el archivo, creando el archivo `file.Z`, destruye el archivo original.
- `uncompress file.Z` : descomprime el archivo, creando el archivo `file`, destruye el archivo original.
- `zcat file.Z` : muestra por el `stdout` el contenido descomprimido del archivo (sin destruir el original).

Otra alternativa de compresor mucho más usada es `gzip` el compresor de `GNU` que posee una mayor razón de compresión que `compress`. Veamos los comandos:

- `gzip file` : comprime el archivo, creando el archivo `file.gz`, destruye el archivo original.
- `gunzip file.gz` : descomprime el archivo, creando el archivo `file`, destruye el archivo original.
- `zless file.gz` : muestra por el `stdout` el contenido descomprimido del archivo paginado por `less`.

La extensión empleada en los archivos comprimidos con `gzip` suele ser `.gz` pero a veces se usa `.gzip`. Adicionalmente el programa `gunzip` también puede descomprimir archivos creados con `compress`.

La opción con mayor tasa de compresión que `gzip` es `bzip2` y su descompresor `bunzip2`. La extensión usada en este caso es `.bz2`. El *kernel* de Linux se distribuye en formato `bzip2`.

Existen también versiones de los compresores compatibles con otros sistemas operativos: `zip`, `unzip`, `unarj`, `lha`, `rar` y `zoo`.

En caso que se desee crear un archivo comprimido con una estructura de directorios debe ejecutarse la orden:

```
tar cvzf  nombre.tgz directorio
```

o bien

```
tar cvjf  nombre.tbz directorio
```

En el primer caso comprime con `gzip` y en el segundo con `bzip2`. Para descomprimir y restablecer la estructura de directorio almacenada se usan los comandos:

```
tar xvzf  nombre.tgz directorio
```

si se realizó la compresión con `gzip` o bien

```
tar xvjf  nombre.tbz directorio
```

si se realizó la compresión con `bzip2`.

Capítulo 9

Una breve introducción a C++.

versión final 3.6-021212

En este capítulo se intentará dar los elementos básicos del lenguaje de programación C++. No se pretende más que satisfacer las mínimas necesidades del curso, sirviendo como un ayuda de memoria de los tópicos abordados, para futura referencia. Se debe consignar que no se consideran todas las posibilidades del lenguaje y las explicaciones están reducidas al mínimo.

9.1. Estructura básica de un programa en C++.

9.1.1. El programa más simple.

El primer ejemplo de todo manual es el que permite escribir "Hola" en la pantalla.

```
//
// Los comentarios comienzan con //
//
#include <iostream>
int main()
{
    cout << "Hola." << endl;
    return 0 ;
}
```

Las tres primeras líneas corresponden a comentarios, todo lo que está a la derecha de los caracteres `//` son comentarios y no serán considerados en la compilación. En la línea siguiente se incluye un archivo de cabecera, o *header*, con la instrucción de preprocesador `#include`. El nombre del archivo se puede escribir como `<nombre>` o bien `"nombre.h"`. En el primer caso el archivo `nombre` será buscado en el *path* por defecto para los `include`, típicamente `/usr/include` o `/usr/include/g++-3/` en el caso de *headers* propios de C++; en el segundo caso la búsqueda se hace en el directorio local. También podríamos incluir un *path* completo cuando se ocupan las comillas. En nuestro ejemplo se incluye el archivo `iostream`, en el cual se hacen las definiciones adecuadas para el manejo de la entrada y salida en C++. Este archivo es necesario para enviar luego un mensaje a pantalla.

La función `int main` es donde comienza a ejecutarse el programa; siempre debe haber una función `main` en nuestro programa. Debido a imposiciones del sistema operativo la función

`main` devuelve un entero y por tanto debe ser declarada `int`. Los paréntesis vacíos `()` indican que el `main` no tiene argumentos de entrada (más adelante se verá que puede tenerlos). Lo que está encerrado entre llaves `{}` corresponde al cuerpo de la función `main`. Cada una de las líneas termina con el carácter `;`. El identificador predefinido `cout` representa la salida a pantalla. El operador `<<` permite que lo que está a su derecha se le dé salida por el dispositivo que está a su izquierda, en este caso `cout`. Si se quiere enviar más de un objeto al dispositivo que está al inicio de la línea agregamos otro operador `<<`, y en este caso lo que está a la derecha del operador se agregará a lo que está a la izquierda y todo junto será enviado al dispositivo. En nuestro caso se ha enviado `endl`, un objeto predefinido en el archivo `iostream.h` que corresponde a un cambio de línea, el cual será agregado al final del mensaje. La línea final contiene la instrucción de retorno del entero cero, `return 0`.

Si escribimos nuestro primer programa en el editor `xemacs` con el nombre de `primero.cc` las instrucciones para editarlo, compilarlo y correrlo serán:

```
jrogan@pucon:~/tmp$ xemacs primero.cc
jrogan@pucon:~/tmp$ g++ -Wall -o primero primero.cc
jrogan@pucon:~/tmp$ ./primero
Hola.
jrogan@pucon:~/tmp$
```

Luego de la compilación, un archivo ejecutable llamado `primero` es creado en el directorio actual. Si el directorio actual no está en el `PATH`, nuestro programa debe ser ejecutado anteponiendo `./`. Si está en el `PATH`, para ejecutarlo basta escribir `primero`. (Para agregar el directorio local al `PATH` basta editar el archivo `~/.bashrc` agregarle una línea como `PATH="${PATH}:."` y ejecutar en la línea de comando `source ~/.bashrc` para que los cambios tengan efecto.)

9.1.2. Definición de funciones.

Las funciones en C++ son muy importantes, pues permiten aislar parte del código en una entidad separada. Esto es un primer paso a la *modularización* de nuestro programa, es decir, a la posibilidad de escribirlo en partes que puedan ser editadas de modo lo más independiente posible. Ello facilita enormemente la creación de código complicado, pues simplifica su modificación y la localización de errores. Nos encontraremos frecuentemente con este concepto.

Aprovecharemos de introducir las funciones modificando el primer programa de manera que se delegue la impresión del mensaje anterior a una función independiente:

```
//
// Segunda version incluye funcion adicional
//
#include <iostream>

void PrintHola()
{
    cout << "Hola." << endl;
```

```
}

int main()
{
    PrintHola();
    return 0;
}
```

La función debe estar definida antes de que sea ocupada, por eso va primero en el código fuente. Como ya se dijo antes, la ejecución del programa comienza en la función `main` a pesar de que no está primera en el código fuente. Los paréntesis vacíos indican que la función `PrintHola` no tiene argumentos y la palabra delante del nombre de la función indica el tipo de dato que devuelve. En nuestro caso la palabra `void` indica que no devuelve nada a la función `main`.

Una alternativa al código anterior es la siguiente:

```
#include <iostream>

void PrintHola();

int main()
{
    PrintHola();
    return 0 ;
}

void PrintHola()
{
    cout << "Hola." << endl;
}
```

En esta versión se ha separado la *declaración* de la función de su *implementación*. En la declaración se establece el nombre de la función, los argumentos que recibe, y el tipo de variable que entrega como resultado. En la implementación se da explícitamente el código que corresponde a la función. Habíamos dicho que una función debe estar definida antes que sea ocupada. En verdad, basta conque la función esté declarada. La implementación puede ir después (como en el ejemplo anterior), o incluso en un archivo distinto, como veremos más adelante. La separación de declaración e implementación es otro paso hacia la modularización de nuestro programa.

9.1.3. Nombres de variables.

Nuestros datos en los programas serán almacenados en objetos llamados variables. Para referirnos a ellas usamos un nombre que debe estar de acuerdo a las siguientes reglas:

– Deben comenzar con una letra (mayúsculas y minúsculas son distintas).

- Pueden contener números, pero no comenzar por uno.
- Pueden contener el símbolo `_` (*underscore*).
- Longitud arbitraria.
- No pueden corresponder a una de las palabras reservadas de C++[1]:

```
asm       delete  if         return    try
auto      do      inline     short     typedef
break     double  int        signed    union
case      else    long       sizeof    unsigned
catch     enum    new        static    virtual
char      extern  operator   struct    void
class     float   private    switch    volatile
const     for     protected  template  while
continue  friend  public     this
default   goto    register   throw
```

9.1.4. Tipos de variables.

Todas las variables a usar deben ser *declaradas* de acuerdo a su tipo. Por ejemplo, si usamos una variable `i` que sea un número entero, debemos, antes de usarla, declararla, y sólo entonces podemos asignarle un valor:

```
int i;
i=10;
```

Esta necesidad de declarar cada variable a usar se relaciona con la característica de C++ de ser fuertemente "tipeado"[2]. Algunos de los errores más habituales en programación se deben al intento de asignar a variables valores que no corresponden a sus tipos originales. Si bien esto puede no ser muy grave en ciertos contextos, a medida que los programas se vuelven más complejos puede convertirse en un verdadero problema. El compilador de C++ es capaz de detectar los usos indebidos de las variables pues conoce sus tipos, y de este modo nuestro código se vuelve más seguro.

Es posible reunir las acciones de declaración e inicialización en una misma línea:

```
int i=10;
```

o declarar más de una variable del mismo tipo simultáneamente, e inicializar algunas en la misma línea:

```
int r1, r2, r3 = 10;
```

[1] A esta tabla hay que agregar algunas palabras adicionales, presentes en versiones más recientes de C++, como `namespace` y `using`

[2] Una traducción libre del término inglés *strongly typed*.

A veces se requiere que una variable no varíe una vez que se le asigna un valor. Por ejemplo, podríamos necesitar definir el valor de $\pi = 3{,}14159...$, y naturalmente no nos gustaría que, por un descuido, a esa variable se le asignara otro valor en alguna parte del programa. Para asegurarnos de que ello no ocurra, basta agregar el modificador `const` a la variable:

```
const float pi = 3.14159;
```

Para números reales se puede usar la notación exponencial. Por ejemplo, `1.5e-3` representa el número $1{,}5 \cdot 10^{-3}$.

Una variable puede ser declarada sólo una vez, pero naturalmente se le pueden asignar valores en un número arbitrario de ocasiones.

Los tipos de variables disponibles son[3]:

`bool`	Booleanas *true* y *false*.
`int`	Enteros entre $-2^{15} = -32768$ y $2^{15} - 1 = 32767$
	o entre $-2^{31} = -2147483648$ y $2^{31} - 1 = 2147483647$
`short int`	
`short`	Enteros entre -2^{15} y $2^{15} - 1$.
`long int`	
`long`	Enteros entre -2^{31} y $2^{31} - 1$.
`unsigned int`	Enteros entre 0 y $2^{16} - 1$ o entre 0 y $2^{32} - 1$.
`unsigned short`	Enteros entre 0 y $2^{16} - 1$.
`unsigned long`	Enteros entre 0 y $2^{32} - 1$.
`char`	Caracteres.
`float`	Reales en los intervalos $[-1{,}7 \cdot 10^{38}, -0{,}29 \cdot 10^{-38}]$,
	$[0{,}29 \cdot 10^{-38}, 1{,}7 \cdot 10^{38}]$
	(Precisión de unos 7 dígitos decimales.)
`double`	Reales en los mismos intervalos que `float`,
	pero con precisión de 16 decimales,
	o en los intervalos $[-0{,}9 \cdot 10^{308}, -0{,}86 \cdot 10^{-308}]$
	y $[0{,}86 \cdot 10^{-308}, 0{,}9 \cdot 10^{308}]$, con precisión de 15 decimales.

Las variables tipo `char` alojan caracteres, debiendo inicializarse en la forma:

```
char c = 'a';
```

Además de las letras mayúsculas y minúsculas, y símbolos como `&`, `(`, `:`, etc., hay una serie de caracteres especiales (*escape codes*) que es posible asignar a una variable `char`. Ellos son:

[3]Los valores de los rangos indicados son simplemente representativos y dependen de la máquina utilizada. Además, estos valores no corresponden exactamente a las versiones más recientes de C++.

newline	`\n`
horizontal tab	`\t`
vertical tab	`\v`
backspace	`\b`
carriage return	`\r`
form feed	`\f`
alert (bell)	`\a`
backslash	`\\`
single quote	`\'`
double quote	`\"`

Por ejemplo, la línea:

```
cout << "Primera columna\t Segunda columna\n
          Segunda linea" << endl;
```

corresponde al *outpput*

```
Primera columna         Segunda columna
Segunda linea
```

9.1.5. Ingreso de datos desde el teclado.

El *header* `iostream` define un objeto especial llamado `cin` que está asociado al teclado o `stdin`. Con el operador `>>` asignamos la entrada en el dispositivo de la izquierda a la variable de la derecha; una segunda entrada requiere de otro operador `>>` y de otra variable. En el siguiente ejemplo veremos una declaración simultánea de dos variables del mismo tipo `i` y `j`, un mensaje a pantalla con las instrucciones a seguir, el ingreso de dos variables desde el teclado y luego su escritura en la pantalla.

```
#include <iostream>
int main()
{
    int i, j ;
    cout << "Ingrese dos numeros enteros: " ;
    cin >> i >> j ;
    cout << "Los dos numeros ingresados fueron: " << i <<" "<< j << endl ;
    return 0;
}
```

9.1.6. Operadores aritméticos.

Existen operadores binarios (*i.e.*, que actúan sobre dos variables, una a cada lado del operador) para la suma, la resta, la multiplicación y la división:

```
+   -   *   /
```

9.1.7. Operadores relacionales.

Los símbolos para los operadores relacionales de igualdad, desigualdad, menor, menor o igual, mayor y mayor o igual son:

```
==    !=    <    <=    >    >=
```

Para las relaciones lógicas `AND`, `OR` y `NOT`:

```
&&    ||    !
```

9.1.8. Asignaciones.

a) Asignación simple. Podemos asignar a una variable un valor explícito, o el valor de otra variable:

```
i = 1;
j = k;
```

Una práctica habitual en programación es iterar porciones del código. La iteración puede estar determinada por una variable cuyo valor aumenta (disminuye) cada vez, hasta alcanzar cierto valor máximo (mínimo), momento en el cual la iteración se detiene. Para que una variable `x` aumente su valor en `2`, por ejemplo, basta escribir:

```
x = x + 2;
```

Si `x` fuera una variable matemática normal, esta expresión no tendría sentido. Esta expresión es posible porque el compilador interpreta a `x` de modo distinto a cada lado del signo igual: a la derecha del signo igual se usa el valor contenido en la variable `x` (por ejemplo, 10); a la izquierda del signo igual se usa la dirección de memoria en la cual está alojada la variable `x`. De este modo, la asignación anterior tiene el efecto de colocar en la dirección de memoria que contiene a `x`, el valor que tiene `x` más `2`. En general, todas las variables tienen un *rvalue* y un *lvalue*: el primero es el valor usado a la derecha (*right*) del signo igual en una asignación, y el segundo es el valor usado a la izquierda (*left*), es decir, su dirección de memoria.

b) Asignación compuesta.

La expresión `x = x + 2` se puede reemplazar por `x += 2`.

Existen los operadores `+=` `-=` `*=` `/=`

c) Operadores de incremento y decremento.

La expresión `x = x + 1` se puede reescribir `x += 1` o bien `x++`.

Análogamente, existe el operador `--`. Ambos operadores unarios, `++` y `--` pueden ocuparse como prefijos o sufijos sobre una variable y su acción difiere en ambos casos. Como prefijo la operación de incremento o decremento se aplica antes de que el valor

de la variable sea usado en la evaluación de la expresión. Como sufijo el valor de la variable es usado en la evaluación de la expresión antes que la operación de incremento o decremento. Por ejemplo, supongamos que inicialmente $x = 3$. Entonces la instrucción `y=x++` hace que $y = 3$, $x = 4$; por su parte, `y=++x` hace que $y = 4$, $x = 4$.

Con estas consideraciones, deberíamos poder convencernos de que la salida del siguiente programa es `3 2 2-1 1 1` :

```
// Ejemplo de operadores unarios ++ y --.
#include <iostream>
int main()
{
  int y ; int x = (y = 1) ;
  int w = ++x + y++;
  cout << w <<" " << x << " " << y << "-" ;
  w = x-- - --y;
  cout <<  w << " " << x << " " << y << endl ;
  return 0;
}
```

Los operadores para asignación compuesta, y los de incremento y decremento, no son sólo abreviaciones. En realidad hay que preferirlas porque implican optimizaciones en el ejecutable resultante.

9.1.9. Conversión de tipos.

Una consecuencia de que C++ sea fuertemente "tipeado" es que no se pueden hacer operaciones binarias con objetos de tipos distintos. En la siguiente expresión,

```
int i = 3;
float x = 43.8;
cout << "Suma = " << x + i << endl;
```

el computador debe sumar dos variables de tipos distintos, y en principio la operación es imposible. La estrategia para resolver este problema es convertir ambas variables a un tipo común antes de efectuar la suma (en inglés, decimos que hacemos un *cast* de un tipo a otro. Existen dos modos de proceder:

a) Conversión explícita.

   Si `i` es un `int`, por ejemplo, entonces `float(i)` la convierte en `float`. Así, el programa anterior se puede reescribir:

```
int i = 3;
float x = 43.8;
cout << "Suma = " << x + float(i) << endl;
```

Ahora la suma es claramente entre dos variables `float`, y se puede realizar. Sin embargo, esto es bastante tedioso, por cuanto el programador debe realizar el trabajo de conversión personalmente cada vez que en su código se desee sumar un real con un número entero.

b) Conversión implícita.

En este caso, el compilador realiza las conversiones de modo automático, prefiriendo siempre la conversión desde un tipo de variable de menor precisión a uno de mayor precisión (de `int` a `double`, de `short` a `int`, etc.). Así, a pesar de lo que dijimos, el código anterior habría funcionado en su forma original. Evidentemente esto es muy cómodo, porque no necesitamos hacer una conversión explícita cada vez que sumamos un entero con un real. Sin embargo, debemos estar conscientes de que esta comodidad sólo es posible porque ocurren varias cosas: primero, el compilador detecta el intento de operar sobre dos variables que no son del mismo tipo; segundo, el compilador detecta, en sus reglas internas, la posibilidad de cambiar uno de los tipos (`int` en este caso) al otro (`float`); tercero, el compilador realiza la conversión, y finalmente la operación se puede llevar a cabo. Entender este proceso nos permitirá aprovechar las posibilidades de la conversión implícita de tipos cuando nuestro código involucre tipos de variables más complicados, y entender varios mensajes de error del compilador.

Es interesante notar cómo las conversiones implícitas de tipos pueden tener consecuencias insospechadas. Consideremos las tres expresiones:

i) `x = (1/2) * (x + a/x) ;`

ii) `x = (0.5) * (x + a/x) ;`

iii) `x = (x + a/x)/2 ;`

Si inicialmente `x=0.5` y `a=0.5`, por ejemplo, i) entrega el valor `x=0`, mientras ii) y iii) entregan el valor `x=1.5`. Lo que ocurre es que `1` y `2` son enteros, de modo que `1/2 = 0`. De acuerdo a lo que dijimos, uno esperaría que en i), como conviven números reales con enteros, los números enteros fueran convertidos a reales y, por tanto, la expresión tuviera el resultado esperado, 1.5. El problema es la *prioridad* de las operaciones. No todas las operaciones tienen igual prioridad (las multiplicaciones y divisiones se realizan antes que las sumas y restas, por ejemplo), y esto permite al compilador decidir cuál operación efectuar primero. Cuando se encuentra con operaciones de igual prioridad (dos multiplicaciones, por ejemplo), se procede a efectuarlas de izquierda a derecha.

Pues bien, en i), la primera operación es `1/2`, una división entre enteros, *i.e.* cero. En ii) no hay problema, porque todas son operaciones entre reales. Y en iii) la primera operación es el paréntesis, que es una operación entre reales. Al dividir por 2 éste es convertido a real antes de calcular el resultado.

i) aún podría utilizarse, cambiando el prefactor del paréntesis a `1.0/2.0`, una práctica que sería conveniente adoptar como *standard* cuando queremos utilizar enteros dentro de expresiones reales, para evitar errores que pueden llegar a ser muy difíciles de detectar.

9.2. Control de flujo.

9.2.1. `if`, `if... else`, `if... else if`.

Las construcciones siguientes permiten controlar el flujo del programa en base a si una expresión lógica es verdadera o falsa.

a) En el caso de la sentencia `if` se evaluará la expresión `(a==b)`, si ella es cierta ejecutará la o las líneas entre los paréntesis de llave y si la expresión es falsa el programa se salta esa parte del código.

```
if (a==b) {
    cout << "a es igual a b" << endl;
}
```

En este y en muchos de los ejemplos que siguen, los paréntesis cursivos son opcionales. Ellos indican simplemente un grupo de instrucciones que debe ser tratado como una sola instrucción. En el ejemplo anterior, los paréntesis cursivos después del `if` (o después de un `while`, `for`, etc. más adelante) indican el conjunto de instrucciones que deben o no ejecutarse dependiendo de si cierta proposición es verdadera o falsa. Si ese conjunto de instrucciones es una sola, se pueden omitir los paréntesis:

```
if (a==b) cout << "a es igual a b" << endl;
```

b) En el caso `if... else` hay dos acciones mutuamente excluyentes. La sentencia `if (c!=b)` evaluará la expresión `(c!=b)`. Si ella es cierta ejecutará la o las líneas entre los paréntesis de llave que le siguen, saltándose la o las líneas entre los paréntesis de llave que siguen a la palabra clave `else`. Si la expresión es falsa el programa se salta la primera parte del código y sólo ejecuta la o las líneas entre los paréntesis de llave que siguen a `else`.

```
if (c!=d) {
    cout << "c es distinto de d" << endl;
}
else {
    cout << "c es igual a d" << endl;
}
```

c) En el último caso se evaluará la expresión que acompaña al `if` y si ella es cierta se ejecutará la o las líneas entre los paréntesis de llave que le siguen, saltándose todo el resto de las líneas entre los paréntesis de llave que siguen a las palabras claves `else if` y `else`. Si la primera expresión es falsa el programa se salta la primera parte del código y evalúa la expresión que acompaña al primer `else if` y si ella es cierta ejecutará la o las líneas entre los paréntesis de llave que le siguen, saltándose todo el resto de las líneas entre los paréntesis que siguen a otros eventuales `else if` o al `else`. Si ninguna de las expresiones lógicas resulta cierta se ejecutará la o las líneas entre los paréntesis que siguen al `else`.

```
if (e > f) {
    cout << "e es mayor que f" << endl;
}
else if (e == f) {
    cout << "e es igual a f" << endl;
}
else {
    cout << "e es menor que f" << endl;
}
```

Para C++, una expresión verdadera es igual a `1`, y una falsa es igual a `0`. Esto es, cuando escribimos `if(e>f)`, y `e>f` es falsa, en realidad estamos diciendo `if(0)`. A la inversa, `0` es considerada una expresión falsa, y cualquier valor no nulo es considerado una expresión verdadera. Así, podríamos hacer que una porción del código siempre se ejecute (o nunca) poniendo directamente `if(1)` o `if(0)`, respectivamente.

Naturalmente, lo anterior no tiene mucho sentido, pero un error habitual (y particularmente difícil de detectar) es escribir `a = b` en vez de `a == b` en una expresión lógica. Esto normalmente trae consecuencias indeseadas, pues la asignación `a = b` es una función que se evalúa siempre al nuevo valor de `a`. En efecto, una expresión como `a=3` siempre equivale a verdadero, y `a=0` siempre equivale a falso. Por ejemplo, en el siguiente programa:

```
#include <iostream>
int main(){
  int k=3;
  if (k==3){
    cout << "k es igual a 3" << endl;
  }
  k=4;
  if (k=3){
    cout << "k es igual a 3" << endl;
  }
  return 0;
}
```

la salida siempre es:

```
k es igual a 3
k es igual a 3
```

aunque entre los dos `if` el valor de `k` cambia.

9.2.2. Expresión condicional.

Una construcción `if else` simple, que sólo asigna un valor distinto a una misma variable según si una proposición es verdadera o falsa, es muy común en programación. Por ejemplo:

```
if (a==b) {
    c = 1;
} else {
    c = 0;
}
```

Existen dos maneras de compactar este código. Éste se puede reemplazar por

```
if (a==b) c = 1;
else c = 0;
```

Sin embargo, esto no es recomendable por razones de claridad al leer el código. Una expresión más compacta y clara, se consigue usando el operador ternario `? :`

```
c = (a==b) ? 1 : 0;
```

Como en el caso de los operadores de incremento y decremento, el uso del operador `?` es preferible para optimizar el ejecutable resultante.

9.2.3. `switch`.

La instrucción `switch` permite elegir múltiples opciones a partir del valor de una variable entera. En el ejemplo siguiente tenemos que si `i==1` la ejecución continuará a partir del caso `case 1:`, si `i==2` la ejecución continuará a partir del caso `case 2:` y así sucesivamente. Si `i` toma un valor que no está enumerado en ningún `case` y existe la etiqueta `default`, la ejecución continuará a partir de ahí. Si no existe `default`, la ejecución continúa luego del último paréntesis cursivo.

```
switch (i)
{
case 1:
     {
         cout << "Caso 1." << endl;
     }
     break;
case 2:
     {
         cout << "Caso 2." << endl;
     }
     break;
default:
     {
         cout << "Otro caso." << endl;
     }
     break;
}
```

La instrucción `break` permite que la ejecución del programa salte a la línea siguiente después de la serie de instrucciones asociadas a `switch`. De esta manera sólo se ejecutarán las líneas correspondientes al `case` elegido y no el resto. Por ejemplo, si `i==1` veríamos en pantalla sólo la línea `Caso 1.` En el otro caso, si no existieran los `break`, y también `i==1`, entonces veríamos en pantalla las líneas `Caso 1.`, `Caso 2.` y `Otro caso.` La instrucción `default` es opcional.

9.2.4. `for.`

Una instrucción que permite repetir un bloque de instrucciones un número definido de veces es el `for`. Su sintaxis comienza con una o varias inicializaciones, luego una condición lógica de continuación mientras sea verdadera, y finalmente una o más expresiones que se evalúan vuelta por vuelta no incluyendo la primera vez. Siguiendo al `for(...)` viene una instrucción o un bloque de ellas encerradas entre paréntesis de llave. En el ejemplo siguiente la variable entera `i` es inicializada al valor 1, luego se verifica que la condición lógica sea cierta y se ejecuta el bloque de instrucciones. A la vuelta siguiente se evalúa la expresión a la extrema derecha (suele ser uno o más incrementadores), se verifica que la condición lógica se mantenga cierta y se ejecuta nuevamente el bloque de instrucciones. Cuando la condición lógica es falsa se termina el *loop*, saltando la ejecución a la línea siguiente al paréntesis que indica el fin del bloque de instrucciones del `for`. En este ejemplo, cuando `i=4` la condición de continuación será falsa y terminará la ejecución del `for`.

```
for (int i = 1; i < 4; i++) {
    cout << "Valor del indice: " << i << endl;
}
```

El *output* correspondiente es:

```
Valor del indice: 1
Valor del indice: 2
Valor del indice: 3
```

Cualquier variable declarada en el primer argumento del `for` es local al *loop*. En este caso, la variable `i` es local, y no interfiere con otras posibles variables `i` que existan en nuestro código.

`for` es una instrucción particularmente flexible. En el primer y tercer argumento del `for` se puede colocar más de una instrucción, separadas por comas. Esto permite, por ejemplo, involucrar más de una variable en el ciclo. El código:

```
for (int i=0, k=20; (i<10) && (k<50); i++, k+=6) {
    cout << "i + k = " << i + k << endl;
}
```

resulta en el *output*:

```
i + k = 20
i + k = 27
i + k = 34
```

```
i + k = 41
i + k = 48
```

Además, la condición de continuación (segundo argumento del `for`), no tiene por qué depender de las variables inicializadas en el primer argumento. Y el tercer argumento no tiene por qué ser un incremento o decremento de las variables del *loop*; puede ser cualquier expresión que queramos ejecutar cada vez que un ciclo termina. En el siguiente ejemplo, además de incrementar los contadores en cada ciclo, se envía un mensaje a pantalla:

```
for (int i=1, k=2;k<20 && i<20;k++, i+=2, cout << "Fin iteracion" << endl){
   cout << " i = " << i << endl;
   cout << " k = " << k << endl;
}
```

El resultado:

```
 i = 1, k = 2
Fin iteracion
 i = 3, k = 3
Fin iteracion
 i = 5, k = 4
```

Todos los argumentos del `for` son opcionales (no los `;`), por lo cual se puede tener un `for` que carezca de inicialización y/o de condición de continuación y/o de una expresión que se evalúe en cada iteración.

Un caso típico en que se aprovecha la opcionalidad de los argumentos del `for` es para tener un *loop* infinito, que puede servir para dejar el programa en pausa indefinida. Para salir del *loop* (y en general, para detener cualquier programa en C++), hay que presionar `^C`:

```
for (; ; ) cout << "Este es un loop infinito, ^C para detenerlo"<< endl;
```

Se puede además, salir abruptamente del *loop* con `break`. El código:

```
  for(int indice=0; indice<10; indice++) {
    int cuadrado = indice*indice ;
    cout << indice << " " ;
    if(cuadrado > 10 ) break ;
  }
  cout << endl;
```

da la salida a pantalla:

```
0 1 2 3 4
```

aun cuando la condición de continuación permite que `indice` llegue hasta `9`.

Finalmente, las variables involucradas en el `for` pueden ser modificadas dentro del ciclo. Por ejemplo, modifiquemos uno de los ejemplos anteriores, cambiando la variable `k` en medio del ciclo:

```
for (int i=1, k=2;k<5 && i<8;k++, i+=2, cout << "Fin iteracion" << endl){
  cout << " i = " << i  << ", k = " << k << endl;
  k = k+5;
}
```

El resultado es:

```
 i = 1, k = 2
Fin iteracion
```

En vez de pasar por el ciclo tres veces, como ocurría originalmente, el programa sale del *loop*, al cabo del primer ciclo, $k = 2 + 5 = 7 > 5$.

En general no es una buena práctica modificar las variables internas del ciclo en medio de él, porque no es muy ordenado, y el desorden normalmente conduce a los errores en programación, pero ocasionalmente puede ser útil hacer uso de esta libertad que proporciona el lenguaje. Los ciclos `for` pueden anidarse, tal que uno contenga a otro completamente.

9.2.5. `while`.

La instrucción `while` permite repetir un bloque de instrucciones encerradas entre paréntesis de llave mientras la condición lógica que acompaña al `while` se mantenga cierta. La condición es evaluada antes de que comience la primera iteración; si es falsa en ésta o en una posterior evaluación no se ejecuta el bloque de instrucciones que le siguen y se continúa la ejecución en la línea siguiente al paréntesis que indica el fin del bloque asociado al `while`. Hay que notar que la instrucción `while` podría no ejecutarse ni una sola vez si la condición no se cumple inicialmente. Un ejemplo simple:

```
int i=1;
while (i < 3) {
    cout << i++ << " ";
}
```

que da por resultado: `1 2 3`.

En el siguiente *loop*, la salida será: `5 4 3 2 1` (¿Por qué?)

```
int k=5 ;
while(k) {
    cout << k-- <<" ";
}
```

9.2.6. `do... while`.

La instrucción `do... while` es análoga a `while`, salvo que la condición lógica es evaluada después de la primera iteración. Por tanto, el bloque se ejecuta al menos una vez, siempre. Un ejemplo simple:

```
do {
    cout << i++ << endl;
} while (i<=20);
```

Podemos construir de otra manera un *loop* infinito usando `do while`

```
do {
    cout << "Este es un segundo loop infinito, ^C para detenerlo"<< endl;
} while (1);
```

9.2.7. `goto`.

Existe también en C++ una instrucción `goto` que permite saltar de un punto a otro del programa (`goto salto;` permite saltar a la línea que contiene la instrucción `salto:`). Sin embargo, se considera una mala técnica de programación usar `goto`, y siempre se puede diseñar un programa evitándolo. Es altamente no recomendable, pero si su utilización simplifica el código se puede usar.

9.3. Funciones.

Las funciones nos permiten programar partes del procedimiento por separado. Un ejemplo simple de ellas lo vimos en la subsección 9.1.2.

9.3.1. Funciones tipo `void`.

Un caso especial de funciones es aquél en que el programa que llama la función no espera que ésta le entregue ningún valor al terminar. Por ejemplo, en la subsección 9.1.2, la función `PrintHola` simplemente imprime un mensaje en pantalla. El resto del programa no necesita de ningún resultado parcial proveniente de la ejecución de dicha función. La definición de estas funciones debe ir precedida de la palabra `void`, como en el ejemplo citado.

9.3.2. `return`.

Si deseamos definir una función que calcule una raíz cuadrada, evidentemente esperamos que la función nos entregue un resultado: el valor de la raíz cuadrada. En este caso hay que traspasar el valor de una variable desde la función al programa que la llamó. Esto se consigue con `return`. Veamos un ejemplo muy simple:

```
int numero(){
    int i = 3;
    return i;
}

int main(){
    cout << "Llamamos a la funcion" << endl;
    cout << "El numero es: " << numero() << endl;
    int i = 5;
    i = i + numero();
    cout << "El numero mas 5 es: " << i << endl;
```

```
    return 0;
}
```

En este caso, la función simplemente entrega el valor de la variable interna `i`, es decir 3, el cual puede ser usado para salida en pantalla o dentro de operaciones matemáticas corrientes.

Separando declaración e implementación de la función, el ejemplo anterior se escribe:

```
int numero();

int main(){ ... }

int numero(){
   int i = 3;
   return i;
}
```

Dos observaciones útiles:

a) La declaración de la función lleva antepuesto el tipo de variable que la función entrega. En el ejemplo, la variable entregada es un entero, `i`, y la declaración debe ser, por tanto, `int numero()`. Podemos tener funciones tipo `double`, `char`, `long`, etc., de acuerdo al tipo de variable que corresponde a `return`.

b) La variable `i` que se usa dentro de `main()` y la que se usa dentro de `numero()` son distintas. A pesar de que tienen el mismo nombre, se pueden usar independientemente como si se llamaran distinto. Se dice que `i` es una variable *local*.

Después de `return` debe haber una expresión que se evalúe a una variable del tipo correspondiente, ya sea explícitamente o a través de un *cast* implícito. Las siguientes funciones devuelven un `double` al programa:

```
double f1(){
  double l = 3.0;
  return l;
}

double f2(){
  double l = 3.0, m = 8e10;
  return l*m;
}

double f3(){
  int l = 3;
  return l;
}
```

Sin embargo, la siguiente función hará que el compilador emita una advertencia, pues se está tratando de devolver un `double` donde debería ser un `int`, y la conversión implica una pérdida de precisión:

```
int f4(){
  double l=3.0;
  return l;
}
```

Naturalmente, podemos modificar la función anterior haciendo una conversión explícita antes de devolver el valor: `return int(l)`.

9.3.3. Funciones con parámetros.

Volviendo al ejemplo de la raíz cuadrada, nos gustaría llamar a esta función con un parámetro (el número al cual se le va a calcular la raíz cuadrada). Consideremos por ejemplo una función que necesita un solo parámetro, de tipo `int`, y cuyo resultado es otro `int`:

```
int funcion(int i){
    i+=4;
    return i;
}

int main(){
    int i = 3;
    cout << "El valor de la funcion es " << funcion(i)
         << endl;
    cout << "El valor del parametro es " << i << endl;
    return 0 ;
}
```

El resultado en pantalla es:

```
El valor de la funcion es 7
El valor del parametro es 3
```

La función `funcion` entrega el valor del parámetro más 4. Usamos el mismo nombre (`i`) para las variables en `main` y `funcion`, pero son variables locales, así que no interfieren. Lo importante es notar que cuando se llama a la función, la reasignación del valor de `i` (`i+=4`) ocurre sólo para la variable local en `funcion`; el parámetro externo mantiene su valor.

Separando declaración e implementación el ejemplo anterior se escribe:

```
int funcion(int);

int main(){...}

int funcion(int i){
   i+=4;
   return i;
}
```

Si nuestra función necesita más parámetros, basta separarlos con comas, indicando para cada uno su tipo:

```
int funcion2(int,double);
void funcion3(double,int,float);
double funcion4(float);
```

El compilador verifica cuidadosamente que cada función sea llamada con el número de parámetros adecuados, y que cada parámetro corresponda al tipo especificado. En los ejemplos anteriores, `funcion2` debe ser llamada siempre con dos argumentos, el primero de los cuales es `int` y el segundo `double`. Como siempre, puede ser necesario un *cast* implícito (si se llama `funcion2` con el segundo argumento `int`, por ejemplo), pero si no existe una regla de conversión automática (llamando a `funcion2` con el primer argumento `double`, por ejemplo), el compilador enviará una advertencia. Además, el compilador verifica que el valor de retorno de la función sea usado como corresponde. Por ejemplo, en las dos líneas:

```
double m = funcion2(2,1e-3);
int k = funcion4(0.4);
```

la primera compilará exitosamente (pero hay un *cast* implícito), y la segunda dará una advertencia.

Existen dos modos de transferir parámetros a una función:

a) Por valor. Se le pasan los parámetros para que la función que es llamada copie sus valores en sus propias variables locales, las cuales desaparecerán cuando la función termine y no tienen nada que ver con las variables originales.

   Hasta ahora, en todos los ejemplos de esta subsección el traspaso de parámetros ha sido por valor. En la función `int funcion(int)`, en el código de la página 300, lo que ha ocurrido es que la función copia el *valor* de la variable externa `i` en una nueva variable (que también se llama `i`, pero está en otra dirección de memoria). El valor con el que trabaja la función es la copia, manteniendo inalterada la variable original.

b) Por referencia. Se le pasa la dirección de memoria de los parámetros. La función llamada puede modificar el valor de tales variables.

   La misma función de la página 300 puede ser modificada para que el paso de parámetros sea por referencia, modificando la declaración:

```
int funcion(int &);

int main()
{
    int i = 3;
    cout << "El valor de la funcion es " << funcion(i)
         << endl;
    cout << "El valor del parametro es " << i << endl;
    return 0;
```

```
}
int funcion(int & i)
{
    i+=4;
    return i;
}
```

En vez de traspasarle a `funcion` el valor del parámetro, se le entrega la *dirección* de memoria de dicha variable. Debido a ello, `funcion` puede modificar el valor de la variable. El resultado en pantalla del último programa será:

```
El valor de la funcion es 7
El valor del parametro es 7
```

Debido a que las variables dejan de ser locales, el paso de parámetros por referencia debe ser usado con sabiduría. De hecho el ejemplo presentado es poco recomendable. Peor aún, el problema es no sólo que las variables dejan de ser locales, sino que *es imposible saber que no lo son* desde el `main`. En efecto, el `main` en ambas versiones de `funcion` es el mismo. Lo único que cambió es la declaración de la función. Puesto que un usuario normal usualmente no conoce la declaración e implementación de cada función que desea usar (pues pueden haber sido hechas por otros programadores), dejamos al usuario en la indefensión.

Por otro lado, hay al menos dos situaciones en que el paso de referencia es la única opción viable para entregar los parámetros. Un caso es cuando hay que cuidar el uso de la memoria. Supongamos que una función necesita un parámetros que es una matriz de 10 millones de filas por 10 millones de columnas. Seguramente estaremos llevando al límite los recursos de nuestra máquina, y sería una torpeza pasarle la matriz por valor: ello involucraría, primero, duplicar la memoria utilizada, con el consiguiente riesgo de que nuestro programa se interrumpa; y segundo, haría el programa más lento, porque la función necesitaría llenar su versión local de la matriz elemento por elemento. Es decir, nada de eficiente. En esta situación, el paso por referencia es lo adecuado.

Un segundo caso en que el paso por referencia es recomendable es cuando efectivamente nuestra intención es cambiar el valor de las variables. El ejemplo típico es el intercambio de dos variables entre sí, digamos `a1=1` y `a2=3`. Luego de ejecutar la función queremos que `a1=3` y `a1=1`. El siguiente código muestra la definición y el uso de una función para esta tarea, y por cierto requiere el paso de parámetros por referencia:

```
#include <iostream>
void  swap(int &,int &);

int main(){

  int i = 3, k=10;
  swap(i,k);
```

```
  cout <<  "Primer argumento: " << i << endl;
  cout <<  "Segundo argumento: " << k << endl;
  return 0 ;
}

void swap(int & j,int & p){
  int temp = j;
  j = p;
  p = temp;
}
```

El *output* es:

```
Primer argumento: 10
Segundo argumento: 3
```

En el ejemplo de la matriz anterior, sería interesante poder pasar el parámetro por referencia, para ahorrar memoria y tiempo de ejecución, pero sin correr el riesgo de que nuestra matriz gigantesca sea modificada por accidente. Afortunadamente existe el modo de hacerlo, usando una palabra que ya hemos visto antes: `const`. En el siguiente código:

```
int f5(const int &);
int main(){...}
int f5(const int & i){...};
```

`f5` recibirá su único argumento por referencia, pero, debido a la presencia del modificador `const`, el compilador avisará si se intenta modificar el argumento en medio del código de la función.

9.3.4. Parámetros por defecto.

C++ permite que omitamos algunos parámetros de la función llamada, la cual reemplaza los valores omitidos por otros predeterminados. Tomemos por ejemplo la función `int funcion(int);` de la subsección 9.3.3, y modifiquémosla de modo que si no le entregamos parámetros, asuma que el número entregado fue 5:

```
int funcion(int i = 5){
    i+=4;
    return i;
}

int main(){
    cout << "El resultado default es " << funcion() << endl;
    int i = 3;
    cout << "Cuando el parametro vale " << i <<
            " el resultado es " << funcion(i) << endl;
    return 0;
}
```

El *output* correspondiente es:

```
El resultado default es 9
Cuando el parametro vale 3 el resultado es 7
```

Separando declaración e implementación:

```
int funcion(int = 5);
main(){...}
int funcion(int i){
    i+=4;
    return i;
}
```

Si una función tiene n argumentos, puede tener $m \leq n$ argumentos opcionales. La única restricción es que, en la declaración e implementación de la función, los parámetros opcionales ocupen los últimos m lugares:

```
void f1(int,int = 4);
int f2(double,int = 4, double = 8.2);
double f3(int = 3,double = 0.0, int = 0);
```

En este caso, `f1(2)`, `f1(2,8)`, `f2(2.3,5)`, `f3(3)`, `f3()`, y muchas otras, son todas llamadas válidas de estas funciones. Cada vez, los parámetros no especificados son reemplazados por sus valores predeterminados.

9.3.5. Ejemplos de funciones: raíz cuadrada y factorial.

Raíz cuadrada.

Con lo visto hasta ahora, ya podemos escribir un programa que calcule la raíz cuadrada de una función. Para escribir una función, debemos tener claro qué se espera de ella: cuántos y de qué tipo son los argumentos que recibirá, qué tipo de valor de retorno deberá tener, y, por cierto, un nombre adecuado. Para la raíz cuadrada, es claro que el argumento es un número. Pero ese número podría ser un entero o un real, y eso al compilador no le da lo mismo. En este punto nos aprovechamos del *cast* implícito: en realidad, basta definir la raíz cuadrada con argumento `double`; de este modo, si se llama la función con un argumento `int`, el compilador convertirá automáticamente el `int` en `double` y nada fallará. En cambio, si la definiéramos para `int` y la llamamos con argumento `double`, el compilador se quejaría de que no sabe efectuar la conversión. Si el argumento es `double`, evidentemente esperamos que el valor de retorno de la función sea también un `double`. Llamando a la función `raiz`, tenemos la declaración:

```
double raiz(double);
```

Debido a la naturaleza de la función raíz cuadrada, `raiz()` no tendría sentido, y por tanto no corresponde declararla con un valor *default*.

Ahora debemos pensar en cómo calcular la raíz cuadrada. Usando una variante del método de Newton-Raphson, se obtiene que la secuencia

$$x_{n+1} = \frac{1}{2}\left(x_n + \frac{a}{x_n}\right)$$

converge a $\sqrt{a}$ cuando $n \to \infty$. Por tanto, podemos calcular la raíz cuadrada con aproximaciones sucesivas. El cálculo terminará en el paso N, cuando la diferencia entre el cuadrado de la aproximación actual, x_N, y el valor de a, sea menor que un cierto número pequeño: $|\, x_N^2 - a \,| < \epsilon \ll 1$. El valor de ϵ determinará la precisión de nuestro cálculo. Un ejemplo de código lo encontramos a continuación:

```
#include <iostream>
#include <cmath>

double raiz(double);

int main(){

  double r;

  cout.precision(20);
  cout << "Ingrese un numero: " << endl;
  cin >> r;
  cout << raiz(r) << endl;
  return 0 ;
}

double raiz(double a){
  double x, dx = 1e3, epsilon = 1e-8;

  while (fabs(dx)>epsilon){
    x = (x + a/x)/2;
    dx = x*x - a;
    cout << "x = " << x << ", precision = " << dx << endl;
   }
  return x;
}
```

Luego de la declaración de la función `raiz`, está `main`, y al final la implementación de `raiz`. En `main` se pide al usuario que ingrese un número, el cual se aloja en la variable `r`, y se muestra en pantalla el valor de su raíz cuadrada. La instrucción `cout.precision(20)` permite que la salida a pantalla muestre el resultado con 20 cifras significativas.

En la implementación de la función hay varios aspectos que observar. Se ha llamado `x` a la variable que contendrá las sucesivas aproximaciones a la raíz. Al final del ciclo, `x` contendrá el valor (aproximado) de la raíz cuadrada. `dx` contiene la diferencia entre el cuadrado de `x` y el

valor de `a`, `epsilon` es el número (pequeño) que determina si la aproximación es satisfactoria o no.

El ciclo está dado por una instrucción `while`, y se ejecuta mientras `dx>epsilon`, es decir, termina cuando `dx` es suficientemente pequeño. El valor absoluto del real `dx` se obtiene con la función matemática `fabs`, disponible en el *header* `cmath` incluido al comienzo del programa. Observar que inicialmente `dx=1e3`, esto es un valor muy grande; esto permite que la condición del `while` sea siempre verdadera, y el ciclo se ejecuta al menos una vez.

Dentro del ciclo, se calcula la nueva aproximación, y se envía a pantalla un mensaje con la aproximación actual y la precisión alcanzada (dada por `dx`). Eventualmente, cuando la aproximación es suficientemente buena, se sale del ciclo y la función entrega a `main` el valor de `x` actual, que es la última aproximación calculada.

Factorial.

Otro ejemplo útil es el cálculo del factorial, definido para números naturales:

$$n! = n \cdot (n-1) \cdots 2 \cdot 1 \ , \quad 0! \equiv 1 \ .$$

La estrategia natural es utilizar un ciclo `for`, determinado por una variable entera i, que va desde 1 a n, guardando los resultados en una variable auxiliar que contiene el producto de todos los números naturales desde 1 hasta i:

```
#include <iostream>

int factorial(int);

int main(){
  int n=5 ;
  cout << "El factorial de " << n << " es: " << factorial(n) << endl;
  return 0 ;
}

int factorial(int i)
{
  int f =1;
  for (int j=1;j<=i;j++){
    f = f*j;
  }
  return f;
}
```

Observar que la variable auxiliar `f`, que contiene el producto de los primeros i números naturales, debe ser inicializada a 1. Si se inicializara a 0, `factorial(n)` sería 0 para todo `n`.

Esta función no considera el caso `n=0`, pero al menos para el resto de los naturales funcionará bien.

9.3.6. Alcance, visibilidad, tiempo de vida.

Con el concepto de función hemos apreciado que es posible que coexistan variables con el mismo nombre en puntos distintos del programa, y que signifiquen cosas distintas. Conviene entonces tener en claro tres conceptos que están ligados a esta propiedad:

Alcance (*scope*) La sección del código durante la cual el nombre de una variable puede ser usado. Comprende desde la declaración de la variable hasta el final del cuerpo de la función donde es declarada.

Si la variable es declarada dentro de una función es *local*. Si es definida fuera de todas las funciones (incluso fuera de `main`), la variable es *global*.

Visibilidad Indica cuáles de las variables, actualmente al alcance, pueden ser accesadas. En nuestros ejemplos (subsección 9.3.3), la variable `i` en `main` aún está al alcance dentro de la función `funcion`, pero no es visible, y por eso es posible reutilizar el nombre.

Tiempo de vida Indica cuándo las variables son creadas y cuándo destruidas. En general este concepto coincide con el alcance (las variables son creadas cuando son declaradas y destruidas cuando la función dentro de la cual fueron declaradas termina), salvo porque es posible definir: (a) variables *dinámicas*, que no tienen alcance, sino sólo tiempo de vida; (b) variables *estáticas*, que conservan su valor entre llamadas sucesivas de una función (estas variables tienen tiempo de vida mayor que su alcance). Para declarar estas últimas se usa un modificador `static`.

El efecto del modificador `static` se aprecia en el siguiente ejemplo:

```
#include <iostream>

int f();

int main(){

  cout << f() << endl;
  cout << f() << endl;
  return 0;
}

int f(){
  int x=0;
  x++;
  return x;
}
```

La función `f` simplemente toma el valor inicial de `x` y le suma 1. Como cada vez que la función es llamada la variable local `x` es creada e inicializada, el resultado de este programa es siempre un 1 en pantalla:

```
1
1
```

Ahora modifiquemos la función, haciendo que `x` sea una variable estática:

```
#include <iostream>

int f();

int main(){

  cout << f() << endl;
  cout << f() << endl;
  return 0 ;
}

int f(){
  static int x=0;
  x++;
  return x;
}
```

Ahora, al llamar a `f` por primera vez, la variable `x` es creada e inicializada, pero no destruida cuando la función termina, de modo que conserva su valor cuando es llamada por segunda vez:

```
1
2
```

Veamos un ejemplo de una variable estática en el cálculo del factorial:

```
int factorial2(int i=1){
    static int fac = 1;
    fac*=i;
    return fac ;
}
int main (){
  int n=5;
  int m=n;
  while(n>0)  factorial2(n--);
  cout << "El factorial de "<< m << " es = " << factorial2() << endl;
  return 0 ;
}
```

La idea, si se desea calcular el factorial de 5, por ejemplo, es llamar a la función `factorial2` una vez, con argumento $n = 5$, y después disminuir n en 1. Dentro de la función, una variable estática (`fac`) aloja el valor $1 * 5 = 5$. Luego se llama nuevamente con $n = 4$, con lo cual `fac=1*5*4`, y así sucesivamente, hasta llegar a $n = 1$, momento en el cual `fac=1*5*4*3*2*1`.

Al disminuir n en 1 una vez más, la condición del `while` es falsa y se sale del ciclo. Al llamar una vez más a `factorial2`, esta vez sin argumentos, el programa asume que el argumento tiene el valor predeterminado 1, y así el resultado es `1*5*4*3*2*1*1`, es decir 5!.

Observemos el uso del operador de decremento en este programa: `factorial2(n--)` llama a la función con argumento `n` y *después* disminuye `n` en 1. Ésto es porque el operador de decremento está actuando como sufijo, y es equivalente a las dos instrucciones:

```
factorial2(n);
n--;
```

Si fuera un prefijo [`factorial2(n--)`], primero disminuiría `n` en 1, y llamaría luego a `factorial2` con el nuevo valor de `n`

Este ejemplo de cálculo del factorial ilustra el uso de una variable estática, que aloja los productos parciales de los números enteros, pero no es un buen ejemplo de una función que calcule el factorial, porque de hecho esta función no lo calcula: es `main` quien, a través de sucesivas llamadas a `factorial2`, calcula el factorial, pero la función en sí no.

9.3.7. Recursión.

C++ soporta un tipo especial de técnica de programación, la recursión, que permite que una función se llame a sí misma (esto es no trivial, por cuanto si definimos, digamos, una función `f`, dentro del cuerpo de la implementación no hay ninguna declaración a una función `f`, y por tanto en principio no se podría usar `f` porque dicho nombre no estaría en *scope*; C++ permite soslayar este hecho). La recursión permite definir de modo muy compacto una función que calcule el factorial de un número entero `n`.

```
int factorial3(int n){
    return (n<2) ? 1: n * factorial3(n-1);
}

int main(){
  int n=5;
  cout << "El factorial de "<< n << " es = " << factorial3(n) << endl;
  return 0;
}
```

En este tercer ejemplo, el factorial de n es definido en función del factorial de $n-1$. Se ha usado la expresión condicional (operador `?`) para compactar aún más el código. Por ejemplo, al pedir el factorial de 5 la función se pregunta si $5 < 2$. Esto es falso, luego, la función devuelve a `main` el valor `5*factorial3(4)`. A su vez, `factorial3(4)` se pregunta si $4 < 2$; siendo falso, devuelve *a la función que la llamó* (es decir, a `factorial3(5)`), el valor `4*factorial3(3)`. El proceso sigue hasta que `factorial(2)` llama a `factorial3(1)`. En ese momento, $1 < 2$, y la función `factorial3(1)`, en vez de llamar nuevamente al factorial, devuelve a la función que la llamó el valor 1. No hay más llamadas a `factorial3`, y el proceso de recursión se detiene. El resultado final es que `main` recibe el valor `factorial3(5)` = `5*factorial3(4)` = $\cdots$ = `5*4*3*2*factorial3(1)` = `5*4*3*2*1`= 120.

Este tercer código para el cálculo del factorial sí considera el caso $n = 0$, y además es más eficiente, al ser más compacto.

La recursión debe ser empleada con cuidado. Es importante asegurarse de que existe una condición para la cual la recursión se detenga, de otro modo, caeríamos en una recursión infinita que haría inútil nuestro programa. En el caso del factorial, pudimos verificar que dicha condición existe, por tanto el programa es finito. En situaciones más complicadas puede no ser tan evidente, y es responsabilidad del programador —como siempre— revisar que todo esté bajo control.

9.3.8. Funciones internas.

Existen muchas funciones previamente implementadas en C++ almacenadas en distintas bibliotecas. Una de las bibliotecas importante es la matemática. Para usarla uno debe incluir el archivo de *header* `<cmath>` y luego al compilar agregar al final del comando de compilación `-lm`:

```
g++ -Wall -o <salida> <fuente>.cc -lm
```

si se desea crear un ejecutable `<salida>` a partir del código en `<fuente>.cc`.

Veamos algunas de estas funciones:

`pow(x,y)`	Eleva a potencia, x^y
`fabs(x)`	Valor absoluto
`sqrt(x)`	Raíz cuadrada
`sin(x) cos(x)`	Seno y coseno
`tan(x)`	Tangente
`atan(x)`	Arcotangente de x en $[-\pi, \pi]$
`atan2(y, x)`	Arcotangente de y/x en $[-\pi, \pi]$
`exp(x)`	Exponencial
`log(x) log10(x)`	Logaritmo natural y logaritmo en base 10
`floor(x)`	Entero más cercano hacia abajo (e.g. `floor(3.2)=3`)
`ceil(x)`	Entero más cercano hacia arriba (e.g. `ceil(3.2)=4`)
`fmod(x,y)`	El resto de x/y (e.g. `fmod(7.3, 2)=1.3`)

Para elevar a potencias enteras, es más conveniente usar la forma explícita en vez de la función `pow`, *i.e.* calcular `x^3` como `x*x*x` es más eficiente computacionalmente que `pow(x,3)`, debido a los algoritmos que usa `pow` para calcular potencias. Éstos son más convenientes cuando las potencias no son enteras, en cuyo caso no existe una forma explícita en términos de productos.

9.4. Punteros.

Una de las ventajas de C++ es permitir el acceso directo del programador a zonas de memoria, ya sea para crearlas, asignarles un valor o destruirlas. Para ello, además de los tipos de variables ya conocidos (`int`, `double`, etc.), C++ proporciona un nuevo tipo: el *puntero*.

El puntero no contiene el valor de una variable, sino la dirección de memoria en la cual dicha variable se encuentra.

Un pequeño ejemplo nos permite ver la diferencia entre un puntero y la variable a la cual ese puntero "apunta":

```
int main(){
    int i = 42;
    int * p = &i;
    cout << "El valor del puntero es: " << p << endl;
    cout << "Y apunta a la variable: " << *p << endl;
    return 0;
}
```

En este programa definimos una variable `i` entera. Al crear esta variable, el programa reservó un espacio adecuado en algún sector de la memoria. Luego pusimos, en esa dirección de memoria, el valor `42`. En la siguiente línea creamos un puntero a `i`, que en este caso denominamos `p`. Los punteros no son punteros a cualquier cosa, sino punteros a un tipo particular de variable. Ello es manifiesto en la forma de la declaración: `int * p`. En la misma línea asignamos a este puntero un valor. Ese valor debe ser también una dirección de memoria, y para eso usamos `&i`, que es la dirección de memoria donde está `i`. Ya hemos visto antes el uso de `&` para entregar una dirección de memoria, al estudiar paso de parámetros a funciones por referencia (9.3.3).

Al ejecutar este programa vemos en pantalla los mensajes:

```
El valor del puntero es: 0xbffff9d8
Y apunta a la variable: 42
```

Primero obtenemos un número hexadecimal imposible de determinar *a priori*, y que corresponde a la dirección de memoria donde quedó ubicada la variable `i`. La segunda línea nos da el valor de la variable que está en esa dirección de memoria: 42. Puesto que `*` aplicado a un puntero entrega el contenido de esa dirección de memoria, se le denomina *operador de dereferenciación*.

En este ejemplo, hemos creado un puntero que contiene la dirección de memoria de una variable preexistente: declaramos una variable, esa variable queda en alguna dirección de memoria, y después asignamos esa dirección de memoria a un puntero. En este caso, podemos referirnos a la variable tanto por su nombre (`i`) como por su puntero asociado (`p_i`).

También es posible crear directamente una dirección de memoria, sin necesidad de crear una variable antes. En este caso, la única forma de manipular este objeto es a través de su puntero, porque no existe ninguna variable y por tanto ningún nombre asociado a él. Esto se hace con el operador `new`. El mismo ejemplo anterior puede ser reescrito usando sólo punteros:

```
int  main(){
    int * p = new int;
    *p = 42;
    cout << "El valor del puntero es: " << p << endl;
    cout << "Y apunta a la variable: " << *p << endl;
    delete p;
```

```
    return 0;
}
```

La primera línea crea un nuevo puntero a `int` llamado `p`. `new` verifica que haya suficiente memoria para alojar un nuevo `int`, y si es así reserva ese espacio de memoria. En `p` queda la dirección de la memoria reservada. Esto es equivalente a la declaración `int i;` del programa anterior, salvo que ahora la única manera de accesar esa dirección de memoria es a través del puntero `p`. A continuación se coloca *dentro* de esa dirección (observar la presencia del operador de dereferenciación `*`) el número `42`. El programa manda a pantalla la misma información que la versión anterior, salvo que seguramente el valor de `p` será distinto.

Finalmente, ya que el puntero no volverá a ser usado, la dirección de memoria debe ser liberada para que nuestro u otros programas puedan utilizarla. Ello se realiza con el operador `delete`. Todo puntero creado con `new` debe ser, cuando ya no se utilice, borrado con `delete`. Ello evitará desagradables problemas en nuestro programa debido a fuga de memoria (*memory leak*).

Los punteros tienen gran importancia cuando de manejar datos dinámicos se trata, es decir, objetos que son creados durante la ejecución del programa, en número imposible de predecir al momento de compilar. Por ejemplo, una aplicación `X-windows` normal que crea una, dos, tres, etc. ventanas a medida que uno abre archivos. En este caso, cada ventana es un objeto dinámico, creado durante la ejecución, y la única forma de manejarlo es a través de un puntero a ese objeto, creado con `new` cuando la ventana es creada, y destruido con `delete` cuando la ventana es cerrada.

9.5. Matrices o arreglos.

9.5.1. Declaración e inicialización.

Podemos declarar (e inicializar inmediatamente) matrices de enteros, reales de doble precisión, caracteres, etc., según nuestras necesidades.

```
int a[5];
double r[3] = {3.5, 4.1, -10.8};
char palabra[5];
```

Una vez declarada la matriz (digamos `a[5]`), los valores individuales se accesan con `a[i]`, con `i` desde 0 a 4. Por ejemplo, podemos inicializar los elementos de la matriz así:

```
a[0] = 3;
a[3] = 5; ...
```

o si queremos ingresarlos desde el teclado:

```
for (i = 0; i < 5; i++){
    cin >> a[i];
}
```

Y si deseamos escribirlos en pantalla:

```
for (i = 0; i < 5; i++){
    cout << a[i];
}
```

9.5.2. Matrices como parámetros de funciones.

Si deseamos, por ejemplo, diseñar una función que mande los elementos de una matriz a pantalla, necesitamos entregarle como parámetro la matriz que va a utilizar. Para ello se agrega `[]` luego del nombre de la variable, para indicar que se trata de una matriz:

```
void PrintMatriz(int, double []);

int main(){
     double matriz[5] = {3.5, 5.2, 2.4, -0.9, -10.8};
     PrintMatriz(5, matriz);
     return 0;
}

void PrintMatriz(int i, double a[]){
     for (int j = 0; j < i; j++){
          cout << "Elemento " << j << " = " << a[j] << endl;
     }
}
```

Observemos que la función debe recibir dos parámetros, uno de los cuales es la dimensión de la matriz. Esto se debe a que cuando las matrices son usadas como parámetros la información de su dimensión no es traspasada, y debe ser comunicada independientemente. Una ligera optimización al programa anterior es modificar `main` a:

```
int main()
{
     const int dim = 5;
     double matriz[dim] = {3.5, 5.2, 2.4, -0.9, -10.8};
     PrintMatriz(dim, matriz);
     return 0;
}
```

De este modo, si eventualmente cambiamos de opinión y deseamos trabajar con matrices de longitud distinta, sólo hay que modificar una línea de código (la primera) en todo el programa, el cual puede llegar a ser bastante largo por cierto. (En el ejemplo, también habría que cambiar la línea de inicialización de la matriz, porque asume que la matriz requiere sólo 5 elementos, pero de todos modos debería ser clara la enorme conveniencia.) Podemos reescribir este programa con un comando de preprocesador para hacer la definición de la dimensión:

```
#include <iostream>
#define DIM 5
```

```
int main(){
     double matriz[DIM] = {3.5, 5.2, 2.4, -0.9, -10.8};
     PrintMatriz(DIM, matriz);
     return 0;
}
```

Sin embargo, ninguna de estas alternativas resuelve el problema de que el compilador espera que la dimensión de una matriz sea un entero constante, determinado en el momento de la compilación (no de la ejecución).

9.5.3. Asignación dinámica.

La reserva de memoria para la matriz podemos hacerla en forma dinámica ocupando el operador `new` que pedirá al sistema la memoria necesaria, si está disponible el sistema se la asignará. Como con cualquier puntero, una vez desocupado el arreglo debemos liberar la memoria con el comando `delete`.

```
#include <iostream>
int main()
{
     cout<<"Ingrese la dimension deseada :" ;
     int dim ;
     cin >> dim ;
     double * matriz = new double[dim] ; // Reserva la memoria
     for(int i=0; i < dim; i++) {
        cout << "Ingrese elemento "<< i <<" : ";
        cin >> matriz[i] ;
     }

     for (int i=0;i<dim;i++){
        cout << matriz[i] << ", ";
     }
     cout << endl;

     delete [] matriz       // Libera la memoria reservada
     return 0;
}
```

Este ejemplo permite apreciar una gran ventaja del uso de punteros, al permitirnos liberarnos de definir la dimensión de una matriz como una constante. Aquí, `dim` es simplemente un `int`. La asignación dinámica permite definir matrices cuya dimensión se determina recién durante la ejecución.

Observemos finalmente que la liberación de memoria, en el caso de arreglos, se hace con el operador `delete []`, no `delete` como en los punteros usuales.

9.5.4. Matrices multidimensionales.

Es fácil declarar e inicializar matrices de más de una dimensión:

```
double array[10][8];
int array[2][3] = {{1, 2, 3},
                   {4, 5, 6}};
```

Una operación usual es definir primero las dimensiones de la matriz, y luego llenar sus elementos uno por uno (o desplegarlos en pantalla), recorriendo la matriz ya sea por filas o por columnas. Hay que tener cuidado del orden en el cual uno realiza las operaciones. En el siguiente código, definimos una matriz de 10 filas y 3 columnas, la llenamos con ceros elemento por elemento, y luego inicializamos tres de sus elementos a números distintos de cero. Finalmente desplegamos la matriz resultante en pantalla:

```
#include <iostream>

int main(){
  const int dimx=3, dimy=10;

  double a[dimy][dimx];

   for (int i=0;i<dimy;i++){
     for (int j=0;j<dimx;j++){
        a[i][j]=0;
     }
     cout << endl;
   }

  a[0][0]=1;
  a[3][2]=2;
  a[9][2]=3;

   for (int i=0;i<dimy;i++){
     for (int j=0;j<dimx;j++){
       cout << a[i][j] << ", ";
     }
     cout << endl;
   }
   return 0;
}
```

Inicializar los elementos a cero inicialmente es particularmente relevante. Si no, la matriz se llenaría con elementos aleatorios.

También es posible definir arreglos bidimensionales dinámicamente. En el siguiente ejemplo, se define una matriz de 200 filas y 400 columnas, inicializándose sus elementos a cero, y finalmente se borra:

```
int main()
{

  int width;
  int height;

  width = 200;
  height = 400;


  double ** matriz = new double * [width];

  for (int i=0;i<width;i++){
    matriz[i] = new double[height];
  }

  for (int i=0;i<width;i++){
    for (int j=0;j<height;j++){
      matriz[i][j] = 0;
    }
  }

  for (int i=0;i<width;i++){
    delete [] matriz[i];
  }
  delete [] matriz;
  return 0;
}
```

Primero se crea, con `new`, un puntero (`matriz`) de dimensión 200, que representará las filas. Cada uno de sus elementos (`matriz[i]`), a su vez, será un nuevo puntero, de dimensión 400, representando cada columna. Por tanto, `matriz` debe ser un puntero a puntero (de dobles, en este caso), y así es definido inicialmente (`double ** ...`). Esto puede parecer extraño a primera vista, pero recordemos que los punteros pueden ser punteros a cualquier objeto, en particular a otro puntero. Luego se crean los punteros de cada columna (primer ciclo `for`). A continuación se llenan los elementos con ceros (segundo ciclo `for`). Finalmente, se libera la memoria, en orden inverso a como fue asignada: primero se libera la memoria de cada columna de la matriz (`delete [] matriz[i]`, tercer ciclo `for`), y por último se libera la memoria del puntero a estos punteros (`delete [] matriz`).

9.5.5. Matrices de caracteres: cadenas (strings).

Una palabra, frase o texto más largo es representado internamente por C++ como una matriz de `char`s. A esto se le llama "cadena" (string). Sin embargo, esto ocasiona un problema, pues las matrices deben ser definidas con dimensión constante (a menos que sean definidas dinámicamente), y las palabras pueden tener longitud arbitraria. La convención de C++ para

resolver el problema es aceptar que una cadena tiene longitud arbitraria, pero debe indicar dónde termina. Esto se hace con el `char` nulo: `'\0'`. Así, para asignar a la variable `palabra` el valor "Hola", debe definirse como una matriz de dimensión 5 (una más que el número de letras):

```
char palabra[5] = {'H', 'o', 'l', 'a', '\0'};
```

Para escribir "Hola" en pantalla basta recorrer los elementos de `palabra` uno a uno:

```
for (i = 0; i < 5; i++)
{
    cout << palabra[i];
}
```

Si tuviéramos que hacer esto cada vez que queremos escribir algo a pantalla no sería muy cómodo. Por ello, también podemos escribir "Hola" en pantalla simplemente con `cout << "Hola"`, y de hecho ése fue el primer ejemplo de este capítulo. De hecho, la declaración de `palabra` podría haberse escrito:

```
char palabra[5] = "Hola";
```

Esto ya es bastante más cómodo, aunque persiste la inconsistencia de definir `palabra` con dimensión 5, cuando en realidad al lado derecho de la asignación hay un objeto con sólo 4 elementos (visibles).

Éste y otros problemas asociados con el manejo convencional de cadenas en C++ se resuelven incluyendo el *header* `string`.

string.

El código anterior se puede reescribir:

```
#include <iostream>
#include <string>

int main(){
  string palabra = "Hola";
  cout << palabra << endl;
  return 0;
}
```

Observar que la línea a incluir es `#include <string>`, *sin la extensión ".h"*. Al incluir `string`, las cadenas pueden ser declaradas como objetos tipo `string` en vez de arreglos de `char`. El hecho de que ya no tengamos que definir a priori la dimensión de la cadena es una gran ventaja. De hecho, permite ingresar palabras desde el teclado trivialmente, sin preocuparse de que el *input* del usuario sea demasiado grande (tal que supere la dimensión del arreglo que podamos haber declarado inicialmente) o demasiado corto (tal que se traduzca en un despilfarro de memoria por reservar más memoria para el arreglo de la que realmente se necesita):

```
#include <iostream>
#include <string>

int main(){
  string palabra;
  cin >> palabra;
  return 0;
}
```

Además, este nuevo tipo `string` permite acceder a un sinnúmero de funciones adicionales que facilitan enormemente el manejo de cadenas. Por ejemplo, las cadenas se pueden sumar, donde la suma de cadenas `a` y `b` está definida (siguiendo la intuición) como la cadena que resulta de poner `b` a continuación de `a`:

```
#include <iostream>
#include <string>

int main(){
  string texto1 = "Primera palabra";
  string texto2 = "Segunda palabra";
  cout << texto1 << endl << texto2 << endl;
  cout << texto1 + ", " + texto2  << endl;
  // La ultima linea es equivalente a:
  // string texto3 = texto1 + ", " + texto2;
  // cout << texto3 << endl;
  return 0 ;
}
```

El *output* de este programa será: `Primera palabra, Segunda palabra`.

Dijimos que es muy fácil ingresar una cadena desde el teclado, pues no es necesario definir la dimensión desde el comienzo. Sin embargo, el código anterior, usando `cin`, no es muy general, porque el *input* termina cuando el usuario ingresa el primer cambio de línea o el primer espacio. Esto es muy cómodo cuando queremos ingresar una serie de valores (por ejemplo, para llenar un arreglo), pues podemos ingresarlos ya sea en la forma: `1<Enter> 2<Enter> 3<Enter>`, etc., o `1 2 3`, etc, pero no es óptimo cuando deseamos ingresar texto, que podría constar de más de una palabra y, por tanto, necesariamente incluiría espacios (por ejemplo, al ingresar el nombre y apellido de una persona). Sin explicar demasiado por qué, digamos que la solución a este problema es utilizar una función asociada a `cin` llamada `gets`, y que espera *input* desde el teclado hasta que el usuario dé el primer cambio de línea. Un ejemplo simple lo encontramos en el siguiente código:

```
#include <iostream>
#include <string>

int main(){
  string texto1 = "El resultado es: " ;
  char * texto2;
```

```
  cin.gets(&texto2);
  cout << texto1 + string(texto2) << endl;
  return 0;
}
```

Observamos que `gets` acepta en realidad un argumento que es un puntero a puntero de caracteres (`texto2` fue declarado como un puntero a `char`, y `gets` es llamado con el argumento `&texto2`, que es la dirección de memoria asociada a `texto2`, *i.e.* el puntero que apunta a `texto2`.)

De este modo, `gets` espera *input* desde el teclado hasta el primer cambio de línea, y asigna la cadena ingresada a `texto2`. Sin embargo, si después queremos utilizar `texto2` en conjunto con otras cadenas (definidas como `string`), será necesario convertirla explícitamente a `string` (ver sección 9.1.9). En nuestro código, deseábamos sumar `texto1` con `texto2` y enviar el resultado a pantalla.

9.6. Manejo de archivos.

Una operación usual en todo tipo de programas es la interacción con archivos. Ya sea que el programa necesite conocer ciertos parámetros de configuración, hacer análisis estadístico sobre un gran número de datos generados por otro programa, entregar las coordenadas de los puntos de una trayectoria para graficarlos posteriormente, etc., lo que se requiere es un modo de ingresar datos desde, o poner datos en, archivos. En C++ ello se efectúa incluyendo el *header* `fstream`.

9.6.1. Archivos de salida.

Observemos el siguiente programa:

```
#include <iostream>
#include <fstream>

int main(){
  ofstream nombre_logico("nombre_fisico.dat");
  int i = 3, j;
  cout << i << endl;
  nombre_logico << i << endl;
  cout << "Ingrese un numero entero: ";
  cin >> j;
  cout << i << endl;
  nombre_logico << j << endl;
  nombre_logico.close();
  return 0;
}
```

La primera línea de `main` define un objeto de tipo `ofstream` (*output file stream*). Esto corresponde a un archivo de salida. Dentro de `main` este archivo será identificado por una variable llamada `nombre_logico`, y corresponderá a un archivo en el disco duro llamado `nombre_fisico.dat`. Naturalmente, el identificador `nombre_logico` puede ser cualquier nombre de variable válido para C++, y `nombre_fisico.dat` puede ser cualquier nombre de archivo válido para el sistema operativo. En particular, se pueden también dar nombres que incluyan *paths* absolutos o relativos:

```
ofstream nombre_logico_1("/home/vmunoz/temp/nombre_fisico.dat");
ofstream nombre_logico_2("../nombre_fisico.dat");
```

Las líneas tercera y sexta de `main` envían a `nombre_logico` (es decir, escribe en `nombre_fisico.dat`), las variables `i` y `j`. Observar la analogía que existe entre estas operaciones y las que envían la misma información a pantalla.[4] Si ejecutamos el programa y en el teclado ingresamos el número `8`, al finalizar la ejecución el archivo `nombre_fisico.dat` tendrá los dos números escritos:

```
3
8
```

Finalmente, el archivo creado debe ser cerrado (`nombre_logico.close()`). Si esta última operación se omite en el código, no habrá errores de compilación, y el programa se encargará de cerrar por sí solo los archivos abiertos durante su ejecución, pero un buen programador debiera tener cuidado de cerrarlos explícitamente. Por ejemplo, un mismo programa podría desear utilizar un mismo archivo más de una vez, o varios programas podrían querer acceder al mismo archivo, y si no se ha insertado un `close` en el punto adecuado esto podría provocar problemas.

El archivo indicado al declarar la variable de tipo `ofstream` tiene modo de escritura, para permitir la salida de datos hacia él. Si no existe un archivo llamado `nombre_fisico.dat` es creado; si existe, los contenidos antiguos se pierden y son reemplazados por los nuevos. No siempre deseamos este comportamiento. A veces deseamos agregar la salida de un programa a un archivo de texto ya existente. En ese caso la declaración del archivo es diferente, para crear el archivo en modo *"append"*:

```
#include <iostream>
#include <fstream>

int main(){

  ofstream nombre_logico("nombre_fisico.dat",ios::app);
  int i = 3;

  nombre_logico << i << endl;
  nombre_logico.close();
```

[4]Esta analogía no es casual y se entiende con el concepto de *clases* (Sec. 9.8). `fstream` e `iostream` definen clases que heredan sus propiedades de un objeto abstracto base, común a ambas, y que en el caso de `iostream` se concreta en la salida estándar —pantalla—, y en el de `fstream` en un archivo.

```
  return 0;
}
```

Si ejecutamos este programa y el archivo `nombre_fisico.dat` no existe, será creado. El resultado será un archivo con el número `3` en él. Al ejecutarlo por segunda vez, los datos se ponen a continuación de los ya existentes, resultando el archivo con el contenido:

```
3
3
```

La línea del tipo `ofstream a("b")` es equivalente a una del tipo `int i=3`, declarando una variable (`a`/`i`) de un cierto tipo (`ofstream`/`int`) y asignándole un valor simultáneamente `"b"`/`3`. Como para los tipos de variables predefinidos de C++, es posible separar declaración y asignación para una variable de tipo `ofstream`:

```
ofstream a;
a.open("b");
```

es equivalente a `ofstream a("b")`. Esto tiene la ventaja de que podríamos usar el mismo nombre lógico para identificar dos archivos físicos distintos, usados en distintos momentos del programa:

```
ofstream a;
a.open("archivo1.txt");
// Codigo en que "archivo1.txt" es utilizado
a.close();
a.open("archivo2.txt");
// Ahora "archivo2.txt" es utilizado
a.close();
```

Observar la necesidad del primer `close`, que permitirá liberar la asociación de `a` a un nombre físico dado, y reutilizar la variable lógica en otro momento.

En los ejemplos hemos escrito solamente variables de tipo `int` en los archivos. Esto por cierto no es restrictivo. Cualquiera de los tipos de variables de C++ —`float`, `double`, `char`, etc.— se puede enviar a un archivo del mismo modo. Dicho esto, en el resto de esta sección seguiremos usando como ejemplo el uso de `int`.

9.6.2. Archivos de entrada.

Ya sabemos que enviar datos a un archivo es tan fácil como enviarlos a pantalla. ¿Cómo hacemos ahora la operación inversa, de leer datos desde un archivo? Como es de esperar, es tan fácil como leerlos desde el teclado. Para crear un archivo en modo de lectura, basta declararlo de tipo `ifstream` (*input file stream*). Por ejemplo, si en `nombre_logico.dat` tenemos los siguientes datos:

```
3
6
9
12
```

el siguiente programa,

```
#include <iostream>
#include <fstream>

int main(){

  ifstream nombre_logico("nombre_fisico.dat");
  int i, j,k,l;

  nombre_logico >> i >> j >> k >> l;

  cout << i << "," << j << "," << k << "," << l << endl;

  nombre_logico.close();
  return 0;
}
```

será equivalente a asignar `i=3`, `j=6`, `k=9`, `l=12`, y luego enviar los datos a pantalla. Observar que la sintaxis para ingresar datos desde un archivo, `nombre_logico >> i`, es idéntica a `cin >> i`, para hacerlo desde el teclado. Al igual que `cin`, espacios en blanco son equivalentes a cambios de línea, de modo que el archivo podría haber sido también:

```
3 6 9 12
```

Por cierto, el ingreso de datos desde un archivo se puede hacer con cualquier técnica, por ejemplo, usando un `for`:

```
  ifstream nombre_logico("nombre_fisico.dat");
  int i;
  for (int j=0;j<10;j++){
    nombre_logico >> i;
    cout << i << ",";
  }
  nombre_logico.close();
}
```

Como con `ofstream`, es posible separar declaración e implementación:

```
ifstream a;
a.open("b");
a.close();
```

9.6.3. Archivos de entrada y salida.

Ocasionalmente nos encontraremos con la necesidad de usar un mismo archivo, en el mismo programa, a veces para escribir datos, y otras veces para leer datos. Por ejemplo, podríamos tener una secuencia de datos en un archivo, leerlos, y de acuerdo al análisis de

esos datos agregar más datos a continuación del mismo archivo, o reemplazar los datos ya existentes con otros. Necesitamos entonces un tipo de variable flexible, que pueda ser usado como entrada y salida. Ese tipo es `fstream`. Todo lo que hemos dicho para `ofstream` y `ifstream` por separado es cierto simultáneamente para `fstream`.[5] Para especificar si el archivo debe ser abierto en modo de escritura o lectura, `open` contiene el argumento `ios::out` o `ios::in`, respectivamente. Por ejemplo, el siguiente código escribe el número `4` en un archivo, y luego lo lee desde el mismo archivo:

```
#include <iostream>
#include <fstream>

int main(){
  fstream nombre_logico;
  nombre_logico.open("nombre_fisico.dat",ios::out);
  int i = 4,j;

  nombre_logico << i << endl;
  nombre_logico.close();

  nombre_logico.open("nombre_fisico.dat",ios::in);

  nombre_logico >> j;
  j = j++;
  cout << j << endl;
  nombre_logico.close();
  return 0;
}
```

Las dos primeras líneas de `main` separan declaración y asignación, y son equivalentes a `fstream nombre_logico("nombre_fisico.dat",ios::out);`, pero lo hemos escrito así para hacer evidente la simetría entre el uso del archivo como salida primero y como entrada después.

De lo anterior, se deduce que:

```
fstream archivo_salida("salida.dat",ios::out);
fstream archivo_entrada("entrada.dat",ios::in);
```

es equivalente a

```
ofstream archivo_salida("salida.dat");
ifstream archivo_entrada("entrada.dat");
```

[5]Nuevamente, este hecho se debe al concepto de clases que subyace a las definiciones de estos tres tipos de variables; `fstream` es una clase derivada a la vez de `ofstream` y de `ifstream`, heredando las propiedades de ambas.

9.7. `main` como función.

Para ejecutar un programa compilado en C++, escribimos su nombre en el *prompt*:

```
user@host:~/$ programa
```

Si el mismo usuario desea ejecutar alguno de los comandos del sistema operativo, debe hacer lo mismo:

```
user@host:~/$ ls
```

Sin embargo, `ls` es en realidad el nombre de un archivo ejecutable en el directorio `/bin`, de modo que en realidad no hay diferencias entre nuestro programa y un comando del sistema operativo en ese sentido. Sin embargo, éstos pueden recibir argumentos y opciones. Por ejemplo, para ver todos los archivos que comienzan con `l` en el directorio local basta con darle a `ls` el argumento `l*`: `ls l*`. Si queremos ordenar los archivos en orden inverso de modificación, basta dar otro argumento, en forma de opción: `ls -tr l*`. Se ve entonces que los argumentos de un archivo ejecutable permiten modificar el comportamiento del programa de modos específicos.

¿Es posible hacer lo mismo con archivos ejecutables hechos por el usuario? La respuesta es sí, y para eso se usan los argumentos del `main`. Recordemos que `main` es una función, pero hasta el momento no hemos aprovechado esa característica. Simplemente sabemos que el programa empieza a ejecutarse en la línea donde está la función `main`. Además, siempre hemos escrito esa línea como `main()`. Sin embargo, `main`, como cualquier función, es capaz de aceptar argumentos. Específicamente, acepta dos argumentos, el primero es un entero (que cuenta el número de argumentos que `main` recibió), y el segundo es un puntero a un arreglo de caracteres (que contiene los distintos argumentos, en forma de cadenas de caracteres, que se le entregaron).

Por ejemplo:

```
#include <iostream>
int main( int argc, char * argv[])
{
     for(int i = 0; i < argc; i++) cout << argv[i] << endl ;
     return 0;
}
```

Si llamamos a este programa `argumentos`, obtenemos distintas salidas al llamarlo con distintos argumentos:

```
user@host:~/$ argumentos
argumentos
user@host:~/$ argumentos ap k
argumentos
ap
k
user@host:~/$ argumentos -t -s arg1
argumentos
```

```
-t
-s
arg1
```

Observar que el primer argumento del programa es siempre el nombre del propio programa. Naturalmente, éste es un ejemplo muy simple. Es tarea del programador decidir cómo manejar cada una de las opciones o argumentos que se le entregan al programa desde la línea de comandos, escribiendo el código correspondiente.

Un segundo aspecto con el cual no hemos sido sistemáticos es que `main`, como toda función, tiene un tipo de retorno. En el caso de `main`, ese tipo debe ser `int`. Este `int` es entregado al sistema operativo, y puede servir para determinar si el programa se ejecutó con normalidad o si ocurrió algo anormal. Podríamos hacer ese valor de retorno igual a 0 o 1, respectivamente. Así, la siguiente estructura es correcta:

```
int main(){
   // Codigo

   return 0;
}
```

En este caso, el programa entrega siempre el valor 0 al sistema operativo.

Los códigos del tipo:

```
main(){
   // Codigo
}
```

o

```
void main(){
   // Codigo
}
```

también compilan, pero el compilador emite una advertencia si es llamado con la opción `-Wall` (*Warning all*). En el primer caso, la advertencia es:

```
warning: ANSI C++ forbids declaration ‘main’ with no type
```

En el segundo:

```
return type for ‘main’ changed to ‘int’
```

En general, siempre es conveniente compilar con la opción `-Wall`, para lograr que nuestro código esté realmente correcto (`g++ -Wall <archivo>.cc -o <archivo>`).

9.8. Clases.

C++ dispone de una serie de tipos de variables con las cuales nos está permitido operar: `int`, `double`, `char`, etc. Creamos variables de estos tipos y luego podemos operar con ellas:

```
int x, y;
x = 3;
y = 6;
int z = x + y;
```

No hay, sin embargo, en C++, una estructura predefinida que corresponda a números complejos, vectores de dimensión n o matrices, por ejemplo. Y sin embargo, nos agradaría disponer de números complejos que pudiéramos definir como

```
z = (3,5);
w = (6,8);
```

y que tuvieran sentido las expresiones

```
a = z + w;
b = z * w;
c = z / w;
d = z + 3;
e = modulo(z);
f = sqrt(z);
```

Todas estas expresiones son completamente naturales desde el punto de vista matemático, y sería bueno que el lenguaje las entendiera. Esto es imposible en el estado actual, pues, por ejemplo, el signo `+` es un operador que espera a ambos lados suyos un número. Sumar cualquier cosa con cualquier cosa no significa nada necesariamente, así que sólo está permitido operar con números. Pero los humanos sabemos que los complejos son números. ¿Cómo decírselo al computador? ¿Cómo convencerlo de que sumar vectores o matrices es también posible matemáticamente, y que el mismo signo `+` debería servir para todas estas operaciones?

La respuesta es: a través del concepto de *clases*. Lo que debemos hacer es definir una clase de números complejos. Llamémosla `Complejo`. Una vez definida correctamente, `Complejo` será un tipo más de variable que el compilador reconocerá, igual que `int`, `double`, `char`, etc. Y será tan fácil operar con los `Complejos` como con todos los tipos de variables preexistentes. Esta facilidad es la base de la extensibilidad de que es capaz C++, y por tanto de todas las propiedades que lo convierten en un lenguaje muy poderoso.

Las clases responden a la necesidad del programador de construir objetos o tipos de datos que respondan a sus necesidades. Si necesitamos trabajar con vectores de 5 coordenadas, será natural definir una clase que corresponda a vectores con 5 coordenadas; si se trata de un programa de administración de personal, la clase puede corresponder a un empleado, con sus datos personales como elementos.

Si bien es cierto uno puede trabajar con clases en el contexto de orientación al procedimiento, las clases muestran con mayor propiedad su potencial con la orientación al objeto, donde cada objeto corresponde a una clase. Por ejemplo, para efectuar una aplicación para `X-windows`, la ventana principal, las ventanas de los archivos abiertos, la barra de menú, las cajas de diálogo, los botones, etc., cada uno de estos objetos estará asociado a una clase.

9.8.1. Definición.

Digamos que queremos una clase para representar a los empleados de una empresa. Llamémosla `Persona`. La convención aceptada es que los nombres de las clases comiencen con mayúscula. Esto es porque las clases, recordemos, corresponderán a tipos de variables tan válidos como los internos de C++ (`int`, `char`, etc.). Al usar nombres con mayúscula distinguimos visualmente los nombres de un tipo de variable interno y uno definido por el usuario.

La estructura mínima de la definición de la clase `Persona` es:

```
class Persona
{

};
```

Todas las características de la clase se definen entre los paréntesis cursivos.

9.8.2. Miembros.

Se denomina *miembros* de una clase a todas las variables y funciones declaradas dentro de una clase. Por ejemplo, para personas, es natural caracterizarlas por su nombre y su edad. Y si se trata de empleados de una empresa, es natural también tener una función que entregue su sueldo:

```
class Persona
{
     string nombre;
     fecha nacimiento;
     int rut;
     double edad();
};
```

Los miembros de una clase pueden tener cualquier nombre, excepto el nombre de la propia clase dentro de la cual se definen, ese nombre está reservado.

9.8.3. Miembros públicos y privados.

Una clase distingue información (datos o funciones) privada (accesible sólo a otros miembros de la misma clase) y pública (accesible a funciones externas a la clase). La parte privada corresponde a la estructura interna de la clase, y la parte pública a la implementación (típicamente funciones), que permite la interacción de la clase con el exterior.

Consideremos ahora nuestro deseo de tener una clase que represente números complejos. Un número complejo tiene dos números reales (parte real e imaginaria), y ésos son elementos privados, es decir, parte de su estructura interna. Sin embargo, nos gustaría poder modificar y conocer esas cantidades. Eso sólo puede hacerse a través de funciones públicas.

```
class Complejo
{
private:
    double real, imaginaria;
public:
    void setreal(double);
    void setimag(double);
    double getreal();
    double getimag();
};
```

En este ejemplo, los miembros privados son sólo variables, y los miembros públicos son sólo funciones. Éste es el caso típico, pero puede haber variables y funciones de ambos tipos.

9.8.4. Operador de selección (.).

Hemos definido una clase de números complejos y funciones que nos permiten conocer y modificar las partes real e imaginaria. ¿Cómo se usan estos elementos? Consideremos el siguiente programa de ejemplo:

```
class Complejo
{
private:
    double real, imaginaria;
public:
    void setreal(double);
    void setimag(double);
    double getreal();
    double getimag();
};

int main()
{
    Complejo z, w;

    z.setreal(3);
    z.setimag(2.8);
    w.setreal(1.5);
    w.setimag(5);
    cout << "El primer numero complejo es: " << z.getreal()
         << " + i*" << z.getimag() << endl;
    cout << "El segundo es: " << w.getreal() << " + i*"
         << z.getimag() << endl;
    return 0;
}
```

Vemos en la primera línea de `main` cómo la clase `Complejo` se usa del mismo modo que usaríamos `int` o `double`. Ahora `Complejo` es un tipo de variable tan válido como los tipos predefinidos por C++. Una vez definida la variable, el operador de selección (`.`) permite acceder a las funciones públicas correspondientes a la clase `Complejo`, aplicadas a la variable particular que nos interesa: `z.setreal(3)` pone en la parte real del `Complejo z` el número 3, y `w.setreal(1.5)` hace lo propio con `w`.

9.8.5. Implementación de funciones miembros.

Ya sabemos cómo declarar funciones miembros en el interior de la clase y cómo usarlas. Ahora veamos cómo se implementan.

```
void Complejo::setreal(double x)
{
    real = x;
}

void Complejo::setimag(double x)
{
    imaginaria = x;
}

double Complejo::getreal()
{
    return real;
}

double Complejo::getimag()
{
    return imaginaria;
}
```

Como toda función, primero va el tipo de la función (`void` o `double` en los ejemplos), luego el nombre de la función y los argumentos. Finalmente la implementación. Lo diferente es que el nombre va precedido del nombre de la clase y el operador "`::`" .

9.8.6. Constructor.

Al declarar una variable, el programa crea el espacio de memoria suficiente para alojarla. Cuando se trata de variables de tipos predefinidos en C++ esto no es problema, pero cuando son tipos definidos por el usuario, C++ debe saber cómo construir ese espacio. La función que realiza esa tarea se denomina *constructor*.

El constructor es una función pública de la clase, que tiene el mismo nombre que ella. Agreguemos un constructor a la clase `Complejo`:

```
class Complejo
```

```
{
private:
    double real,imaginaria;
public:
    Complejo(double,double);
    void setreal(double);
    void setimag(double);
    double getreal();
    double getimag();
};

Complejo::Complejo (double x, double y)
: real(x), imaginaria(y)
{}
```

Definir el constructor de esta manera nos permite crear en nuestro programa variables de tipo `Complejo` y asignarles valores sin usar `setreal()` o `setimag()`:

```
Complejo z (2, 3.8);
Complejo w = Complejo(6.8, -3);
```

En el constructor se inicializan las variables internas que nos interesa inicializar al momento de crear un objeto de esta clase.

Si una de las variables internas a inicializar es una cadena de caracteres, hay que inicializarla de modo un poco distinto. Por ejemplo, si estamos haciendo una clase `OtraPersona` que sólo tenga el nombre de una persona, entonces podemos definir la clase y su constructor en la forma:

```
class OtraPersona
{
private:
    char nombre[20];
public:
    Persona(char []);
};
Persona::Persona(char a[])
{
   strcpy(nombre,a);
}
```

Si uno no especifica el constructor de una clase C++ crea uno *default*, pero en general será insuficiente para cualquier aplicación realmente práctica. Es una mala costumbre ser descuidado y dejar estas decisiones al computador.

9.8.7. Destructor.

Así como es necesario crear espacio de memoria al definir una variable, hay que deshacerse de ese espacio cuando la variable deja de ser necesaria. En otras palabras, la clase necesita

también un *destructor*. Si la clase es `Complejo`, el destructor es una función pública de ella, llamada `~Complejo`.

```
class Complejo
{
private:
    double real, imaginaria;
public:
    Complejo(double,double);
    ~Complejo();
    void setreal(double);
    void setimag(double);
    double getreal();
    double getimag();
};

Complejo::Complejo (double x, double y): real(x), imaginaria(y)
{
}

Complejo::~Complejo()
{
}
```

Como con los constructores, al omitir un destructor C++ genera un *default*, pero es una mala costumbre..., etc.

9.8.8. Arreglos de clases.

Una clase es un tipo de variable como cualquier otra de las predefinidas en C++. Es posible construir matrices con ellas, del mismo modo que uno tiene matrices de enteros o caracteres. La única diferencia con las matrices usuales es que no se pueden sólo declarar, sino que hay que inicializarlas simultáneamente. Por ejemplo, si queremos crear una matriz que contenga 2 números complejos, la línea

```
Complejo z[2];
```

es incorrecta, pero sí es aceptable la línea

```
Complejo z[2] = {Complejo(3.5,-0.8), Complejo(-2,4)};
```

9.9. Sobrecarga.

Para que la definición de nuevos objetos sea realmente útil, hay que ser capaz de hacer con ellos muchas acciones que nos serían naturales. Como ya comentamos al introducir el concepto de clase, nos gustaría sumar números complejos, y que esa suma utilizara el mismo

signo `+` de la suma usual. O extraerles la raíz cuadrada, y que la operación sea tan fácil como escribir `sqrt(z)`. Lo que estamos pidiendo es que el operador `+` o la función `sqrt()` sean *polimórficos*, es decir, que actúen de distinto modo según el tipo de argumento que se entregue. Si `z` es un real, `sqrt(z)` calculará la raíz de un número real; si es complejo, calculará la raíz de un número complejo.

La técnica de programación mediante la cual podemos definir funciones polimórficas se llama *sobrecarga*.

9.9.1. Sobrecarga de funciones.

Digamos que la raíz cuadrada de un número complejo $a + ib$ es $(a/2) + i(b/2)$. (Es más complicado en realidad, pero no queremos escribir las fórmulas ahora.)

Para sobrecargar la función `sqrt()` de modo que acepte números complejos basta definirla así:

```
Complejo sqrt(Complejo z)
{
     return Complejo (z.getreal()/2, z.getimag()/2);
}
```

Observemos que definimos una función `sqrt` que acepta argumentos de tipo `Complejo`, y que entrega un número del mismo tipo. Cuando pidamos la raíz de un número, el computador se preguntará si el número en cuestión es un `int`, `double`, `float` o `Complejo`, y según eso escogerá la versión de `sqrt` que corresponda.

Con la definición anterior podemos obtener la raíz cuadrada de un número complejo simplemente con las instrucciones:

```
Complejo z(1,3);
Complejo raiz = sqrt(z);
```

9.9.2. Sobrecarga de operadores.

¿Cómo le decimos al computador que el signo `+` también puede aceptar números complejos? La respuesta es fácil, porque para C++ un operador no es sino una función, y la acción de sobrecargar que ya vimos sirve en este caso también. La sintaxis es:

```
Complejo operator + (Complejo z, Complejo w)
{
    return Complejo (z.getreal() + w.getreal(),
                     z.getimag() + w.getimag());
}
```

9.9.3. Coerción.

Sabemos definir $a+b$, con a y b complejos. Pero ¿qué pasa si a o b son enteros? ¿O reales? Pareciera que tendríamos que definir no sólo

```
Complejo operator + (Complejo a, Complejo b);
```

sino también todas las combinaciones restantes:

```
Complejo operator + (Complejo a, int b);
Complejo operator + (Complejo a, float b);
Complejo operator + (int a, Complejo b);
```

etcétera.

En realidad esto no es necesario. Por cierto, un número real es un número complejo con parte imaginaria nula, y es posible hacerle saber esto a C++, usando la posibilidad de definir funciones con parámetros *default*. Basta declarar (en el interior de la clase) el constructor de los números complejos como

```
Complejo (double, double = 0);
```

Esto permite definir un número complejo con la instrucción:

```
Complejo c = Complejo(3.5);
```

resultando el número complejo $3{,}5 + i \cdot 0$. Y si tenemos una línea del tipo:

```
Complejo c = Complejo(3,2.8) + 5;
```

el computador convertirá implícitamente el entero 5 a `Complejo` (sabe cómo hacerlo porque el constructor de números complejos acepta también un solo argumento en vez de dos), y luego realizará la suma entre dos complejos, que es entonces la única que es necesario definir.

9.10. Herencia.

Herencia es el mecanismo mediante el cual es posible definir clases a partir de otras, preservando parte de las propiedades de la primera y agregando o modificando otras.

Por ejemplo, si definimos la clase `Persona`, toda `Persona` tendrá una variable miembro que sea su `nombre`. Si definimos una clase `Hombre`, también será `Persona`, y por tanto debería tener `nombre`. Pero además puede tener `esposa`. Y ciertamente no toda `Persona` tiene `esposa`. Sólo un `Hombre`.

C++ provee mecanismos para implementar estas relaciones lógicas y poder definir una clase `Hombre` a partir de `Persona`. Lo vemos en el siguiente ejemplo:

```
class Persona
{
private:
    string nombre;
public:
    Persona(string = "");
    ~Persona();
    string getname();
}
```

```
class Hombre : public Persona
{
private:
    string esposa;
public:
    Hombre(string) : Persona(a)
    { };
    string getwife();
    void setwife(string);
}
```

Primero definimos una clase `Persona` que tiene `nombre`. Luego definimos una clase `Hombre` *a partir de* `Persona` (con la línea `class Hombre : public Persona`). Esto permite de modo automático que `Hombre` tenga también una variable `nombre`. Y finalmente, dentro de la clase `Hombre`, se definen todas aquellas características adicionales que una `Persona` no tiene pero un `Hombre` sí: `esposa`, y funciones miembros para modificar y obtener el nombre de ella.

Un ejemplo de uso de estas dos clases:

```
Persona cocinera("Maria");
Hombre panadero("Claudio");
panadero.setwife("Estela");

cout << cocinera.getname() << endl;
cout << panadero.getname() << endl;
cout << panadero.getwife() << endl;
```

Observemos que `panadero` también tiene una función `getname()`, a pesar de que la clase `Hombre` no la define explícitamente. Esta función se ha *heredado* de la clase de la cual `Hombre` se ha derivado, `Persona`.

9.11. Compilación y debugging.

9.11.1. Compiladores.

El comando para usar el compilador de lenguaje C es `gcc`, para usar el compilador de C++ es `g++` y para usar el compilador de fortran 77 es `g77`. Centrémosnos en el compilador de C++, los demás funcionan en forma muy similar. Su uso más elemental es:
`g++ filename.cc`
Esto compila el archivo `filename.cc` y crea un archivo ejecutable que se denomina `a.out` por omisión. Existen diversas opciones para el compilador, sólo comentaremos una pocas.

- `-c` realiza sólo la compilación pero no el *link*:
  `g++ -c filename.cc`
  genera el archivo `filename.o` que es código objeto.

- `-o exename` define el nombre del ejecutable creado, en lugar del por defecto `a.out`.
  `g++ -o outputfile filename.cc`
- `-lxxx` incluye la librería `/usr/lib/libxxx.a` en la compilación.
  `g++ filename.cc -lm`
  En este caso se compila con la biblioteca matemática `libm.a`.
- `-g` permite el uso de un debugger posteriormente.
- `-On` optimización de grado `n` que puede tomar valores de 1 (por defecto) a 3. El objetivo inicial del compilador es reducir el tiempo de la compilación. Con `-On`, el compilador trata de reducir el tamaño del ejecutable y el tiempo de ejecución, con `n` se aumenta el grado de optimización.
- `-Wall` notifica todos los posibles *warnings* en el código que está siendo compilado.
- `-L/path1 -I/path2/include` incluye en el camino de búsqueda `/path1/` para las bibliotecas y `/path2/include` para los archivos de cabecera (*headers*).

El compilador `gcc` (*the GNU C compiler*) es compatible ANSI.

9.12. make & Makefile.

Frecuentemente los programas están compuestos por diferentes subprogramas que se hayan contenidos en diferentes archivos. La orden de compilación necesaria puede ser muy engorrosa, y a menudo no es necesario volver a compilar todos los archivos, sino sólo aquellos que hayan sido modificados. UNIX dispone de una orden denominada `make` que evita los problemas antes mencionados y permite el mantenimiento de una biblioteca personal de programas. Este comando analiza qué archivos fuentes han sido modificados después de la última compilación y evita recompilaciones innecesarias.

En su uso más simple sólo es necesario suministrar una lista de dependencias y/o instrucciones a la orden `make` en un archivo denominado `Makefile`. Una dependencia es la relación entre dos archivos de forma que un archivo se considera actualizado siempre que el otro tenga una fecha de modificación inferior a éste. Por ejemplo, si el archivo `file.cc` incluye el archivo `file.h`, no se puede considerar actualizado el archivo `file.o` si el archivo `file.cc` o el archivo `file.h` ha sido modificado después de la última compilación. Se dice que el archivo `file.o` depende de `file.cc` y el archivo `file.cc` depende del archivo `file.h`. El Makefile se puede crear con un editor de texto y tiene el siguiente aspecto para establecer una dependencia:

```
# Esto es un ejemplo de Makefile.
# Se pueden poner comentarios tras un caracter hash (#).

FILE1: DEP1 DEP2
        comandos para generar FILE1
ETIQUETA1: FILE2
FILE2: DEP3 DEP4
```

```
        comandos para generar FILE2
ETIQUETA2:
        comandos
```

Se comienza con un destino, seguido de dos puntos (`:`) y los prerrequisitos o dependencias necesarios. También puede ponerse una etiqueta y como dependencia un destino, o bien una etiqueta y uno o más comandos. Si existen muchos prerrequisitos, se puede finalizar la línea con un *backslash* (\) y continuar en la siguiente línea.

En la(s) línea(s) siguiente(s) se escriben uno o más comandos. Cada línea se considera como un comando independiente. Si se desea utilizar múltiples líneas para un comando, se debería poner un *backslash* (\) al final de cada línea del comando. El comando `make` conectará las líneas como si hubieran sido escritas en una única línea. En esta situación, se deben separar los comandos con un punto y coma (`;`) para prevenir errores en la ejecución de el *shell*. **Los comandos deben ser indentados con un tabulador, no con 8 espacios**

Make lee el `Makefile` y determina para cada archivo destino (empezando por el primero) si los comandos deben ser ejecutados. Cada destino, junto con los prerrequisitos o dependencias, es denominado una regla.

Si `make` se ejecuta sin argumentos, sólo se ejecutará el primer destino. Veamos un ejemplo:

```
file.o: file.cc file.h
        g++ -c file.cc
```

En este caso se comprueban las fechas de las última modificaciones de los archivos `file.cc` y `file.h`; si esta fecha es más reciente que las del archivo `file.o` se procede a la compilación.

El comando `make` se puede suministrar con un argumento, que indica la etiqueta situada a la izquierda de los dos puntos. Así en el ejemplo anterior podría invocarse `make file.o`..

Gracias a las variables, un `Makefile` se puede simplificar significativamente. Las variables se definen de la siguiente manera:

```
VARIABLE1=valor1
VARIABLE2=valor2
```

Una variable puede ser utilizada en el resto del `Makefile` refiriéndonos a ella con la expresión `$(VARIABLE)`. Por defecto, `make` sabe las órdenes y dependencias (reglas implícitas) para compilar un archivo `*.cc` y producir un archivo `*.o`, entonces basta especificar solamente las dependencias que make no puede deducir a partir de los nombres de los archivos, por ejemplo:

```
OUTPUTFILE = prog
OBJS = prog.o misc.o aux.o
INCLUDESMISC = misc.h aux.h
INCLUDESFILE = foo.h $(INCLUDESMISC)
LIBS = -lmylib -lg++ -lm

prog.o: $(INCLUDESFILE)
misc.o: $(INCLUDESMISC)
aux.o: aux.h

$(OUTPUTFILE): $(OBJS)
```

```
        gcc $(OBJS) -o $(OUTPUTFILE) $(LIBS)
```

Las reglas patrones son reglas en las cuales se especifican múltiples destinos y construye el nombre de las dependencias para cada blanco basada en el nombre del blanco. La sintaxis de una regla patrón:

```
destinos ... : destino patron:dependencias patrones
        comandos
```

La lista de destinos especifica aquellos sobre los que se aplicará la regla. El destino patrón y las dependencias patrones dicen cómo calcular las dependencias para cada destino. Veamos un ejemplo:

```
objects = foo.o \
          bar.o

all: $(objects)

$(objects): %.o: %.cc
        $(CXX) -c $(CFLAGS) $< -o $@
```

Cada uno de los destinos (`foo.o bar.o`) es comparado con el destino patrón `%.o` para extraer parte de su nombre. La parte que se extrae es conocida como el tronco o *stem*, `foo` y `bar` en este caso. A partir del tronco y la regla patrón de las dependencias `%.cc` `make` construye los nombres completos de ellas (`foo.cc bar.cc`). Además, en el comando del ejemplo anterior aparecen un tipo de variables especiales conocidas como automáticas. La variable `$<` mantiene el nombre de la dependencia actual y la variable `$@` mantiene el nombre del destino actual. Finalmente un ejemplo completo:

```
#
# Makefile para el programa eapp
#_____________________________________________________________
CXX = g++
CXXFLAGS = -Wall -O3 -mcpu=i686 -march=i686
#CXXFLAGS = -Wall -g
LIBS = -lm
BIN = eapp

OBJECTS = eapp.o md.o atoms.o vectores.o metalesEA.o Monte_Carlo.o\
        string_fns.o nlista.o rij2.o velver2.o cbc2.o nforce.o \
        temperature.o controles.o inparamfile.o \
        azar.o velver3.o funciones.o observables.o

$(BIN): $(OBJECTS)

        $(CXX) $(OBJECTS) -o $(BIN) $(LIBS)
```

```
$(OBJECTS): %.o:%.cc
	$(CXX)  -c $(CXXFLAGS) $< -o $@
clean:
	rm -fr  $(OBJECTS) $(BIN)
#End
```

Capítulo 10

Gráfica.

versión final 2.2-021217

En este capítulo queremos mostrar algunas de las posibilidades gráficas presentes en Linux. Cubriendo temas como la visualización, conversión, captura y creación de archivos gráficos. Sólo mencionaremos las aplicaciones principales en cada caso centrándonos en sus posibilidades más que en su utilización específica, ya que la mayoría posee una interfase sencilla de manejar y con amplia documentación.

10.1. Visualización de archivos gráficos.

Si disponemos de un archivo gráfico conteniendo algún tipo de imagen lo primero que es importante determinar es en qué tipo de formato gráfico está codificada. Existe un número realmente grande de diferentes tipos de codificaciones de imágenes, cada una de ellas se considera un formato gráfico. Por razones de reconocimiento inmediato del tipo de formato gráfico se suelen incluir en el nombre del archivo, que contiene la imagen, un trío de letras finales, conocidas como la extensión, que representan el formato. Por ejemplo: bmp, tiff, jpg, ps, eps, fig, gif entre muchas otras.

¿De qué herramientas disponemos en Linux para visualizar estas imágenes? La respuesta es que en Linux se dispone de variadas herramientas para este efecto.

Si se trata de archivos de tipo *PostScript* o *Encapsulated PostScript*, identificados por la extensión `ps` o `eps`, existen las aplicaciones `gv`, `gnome-gv` o `kghostview`, todos programas que nos permitirán visualizar e imprimir este tipo de archivos. Si los archivos son tipo *Portable Document Format*, con extensión `pdf`, tenemos las aplicaciones `gv`, `acroread` o `xpdf`, Con todas ellas podemos ver e imprimir dicho formato. Una mención especial requieren los archivos *DeVice Independent* con extensión `dvi` ya que son el resultado de la compilación de un documento TEX o LATEX, para este tipo de archivo existen las aplicaciones `xdvi` y `kdvi` entre otras. La primera aplicación sólo permite visualizar estos archivos y no imprimirlos, para hacer esto, los archivos se transforman a `ps` y se imprime como cualquier otro *Postscript*.

Para la gran mayoría de formatos gráficos más conocidos y usualmente usados para almacenar fotos existen otra serie se programas especializados en visualización que son capaces de entender la mayoría de los formatos más usados. Entre estos programas podemos mencionar: *Eye of Gnome* (`eog`), *Electric Eyes* (`eeyes`), `kview` o `display`. Podemos mencionar que aplicaciones como `display` entienden sobre ochenta formatos gráficos distintos entre los que se encuentran `ps`, `eps`, `pdf`, `fig`, `html`, entre muchos otros.

10.2. Modificando imágenes

Si queremos modificaciones como rotaciones, ampliaciones, cortes, cambios de paleta de colores, filtros o efectos sencillos, `display` es la herramienta precisa. Pero si se desea intervenir la imagen en forma profesional, el programa `gimp` es el indicado. El nombre `gimp` viene de *GNU Image Manipulation Program*. Se puede usar esta aplicación para editar y manipular imágenes. Pudiendo cargar y salvar en una variedad de formatos, lo que permite usarlo para convertir entre ellos. Gimp puede también ser usado como programa de pintar, de hecho posee una gran variedad de herramientas en este sentido, tales como brocha de aire, lápiz clonador, tijeras inteligentes, curvas bezier, etc. Además, permite incluir *plugins* que realizan gran variedad de manipulaciones de imagen. Como hecho anecdótico podemos mencionar que la imagen oficial de Tux, el pingüino mascota de Linux, fue creada en `gimp`.

10.3. Conversión entre formatos gráficos.

El problema de transformar de un formato a otro es una situación usual en el trabajo con archivos gráficos. Muchos *softwares* tienen salidas muy restringidas en formato o con formatos arcaicos (`gif`) y se presenta la necesidad de convertir estos archivos de salida en otros formatos que nos sean más manejables o prácticos. Como ya se mencionó, `gimp` puede ser usado para convertir entre formatos gráficos. También `display` permite este hecho. Sin embargo, en ambos casos la conversión es vía menús, lo cual lo hace engorroso para un gran número de conversiones e imposible para conversiones de tipo automático. Existe un programa llamado `convert` que realiza conversiones desde la línea de comando. Este programa junto con `display`, `import` y varios otros forman la *suite* gráfica *ImageMagick*, una de las más importantes en UNIX, en general, y en especial en Linux y que ya ha sido migrada a otras plataformas. Además, de la clara ventaja de automatización que proporciona `convert`, posee otro aspecto interesante, puede convertir un grupo de imágenes asociadas en una secuencia de animación o película. Veamos la sintaxis para este programa:

```
user@host:~/imagenes$convert cockatoo.tiff cockatoo.jpg

user@host:~/secuencias$convert -delay 20 dna.* dna.gif
```

En el primer caso convierte el archivo `cockatoo` de formato `tiff` a formato `jpg`. En el segundo, a partir de un conjunto de archivos `gif` numerados correlativamente, crea una secuencia animada con imágenes que persisten por 20 centésimas de segundos en un formato conocido como `gif` animado, muy usado en sitios en Internet.

10.4. Captura de pantalla.

A menudo se necesita guardar imágenes que sólo se pueden generar a tiempo de ejecución, es decir, mientras corre nuestro programa genera la imagen pero no tiene un mecanismo propio para exportarla o salvarla como imagen. En este caso necesitamos capturar la pantalla y poderla almacenar en un archivo para el cual podamos elegir el formato. Para estos efectos

existe un programa, miembro también de la *suite ImageMagick*, llamado `import` que permite hacer el trabajo. La sintaxis es

```
import figure.eps
```

```
import -window root root.jpg
```

En el primer caso uno da el comando en un terminal y queda esperando hasta que uno toque alguna de las ventanas, la cual es guardada en este caso en un archivo `figure.eps` en formato *PostScript*. La extensión le indica al programa qué formato usar para almacenar la imagen. En el segundo caso uno captura la pantalla completa en un archivo `root.jpeg`. Este comando puede ser dado desde la consola de texto para capturar la imagen completa en la pantalla gráfica.

10.5. Creando imágenes.

Para imágenes artísticas sin duda la alternativa es `gimp`, todo le que se dijo respecto a sus posibilidades para modificar imágenes se aplica también en el caso de crearlas. En el caso de necesitar imágenes más bien técnicas como esquemas o diagramas o una ilustración para aclarar un problema la alternativa de `xfig` es muy poderosa. El programa `xfig` es una herramienta manejada por menús que permite dibujar y manipular objetos interactivamente. Las imágenes pueden ser salvadas, en formato `xfig`, y posteriormente editadas. La documentación del programa está en `html` y es fácil de accesar y muy completa. La gran ventaja de `xfig` es que trabaja con objetos y no con bitmaps. Además, puede exportar las imágenes a una gran cantidad de formatos: LaTeX, *Metafont*, *PostScript* o *Encapsulated PostScript* o bien `gif`, `jpeg` y muchos otros.

Habitualmente los dibujos necesarios para ilustrar problemas en Física en tareas, pruebas y apuntes son realizados con este software, exportados a *PostScript* e incluidos en los respectivos archivos LaTeX. También existe una herramienta extremadamente útil que permite convertir un archivo *PostScript*, generado de cualquier manera, a un archivo `fig` que puede ser editado y modificado. Esta aplicación que transforma se llama `pstoedit` y puede llegar a ser realmente práctica.

Una aparente limitación de `xfig` es que se podría pensar que no podemos incluir curvas analíticas, es decir, si necesitamos ilustrar una función gaussiana no podemos pretender "dibujarla" con las herramientas de que dispone `xfig` ¿cómo resolver este problema?, simple veremos que un *software* que grafica funciones analíticas como `gnuplot` permite exportar en formato `fig` luego `xfig` puede leer el archivo y editarlo. Además, `xfig` permite importar e incluir imágenes del tipo *bitmap*, agregando riqueza a los diagramas que puede generar.

Una característica destacable del programa es que trabaja por capas, las cuales son tratadas independientemente, uno puede poner un objeto sobre otro o por debajo de otro logrando diferentes efectos. Algunos programas de presentación gráficos basados en LaTeX y `pdf` están utilizando esta capacidad para lograr animaciones de imágenes.

Finalmente este programa permite construir una biblioteca de objetos reutilizables ahorrando mucho trabajo. Por ejemplo, si uno dibuja los elementos de un circuito eléctrico y los almacena en el lugar de las bibliotecas de imágenes podrá incluir estos objetos en futuros trabajos. El programa viene con varias bibliotecas de objetos listas para usar.

10.6. Graficando funciones y datos.

Existen varias aplicaciones que permiten graficar datos de un archivo, entre las más populares están: `gnuplot`, `xmgrace` y `SciGraphica`. La primera está basada en la línea de comando y permite gráficos en 2 y 3 dimensiones, pudiendo además, graficar funciones directamente sin pasar por un archivo de datos. Las otras dos son aplicaciones basadas en menús que permiten un resultado final de mucha calidad y con múltiples variantes. La debilidad en el caso de `xmgrace` es que sólo hace gráficos bidimensionales.

El programa `gnuplot` se invoca de la línea de comando y da un *prompt* en el mismo terminal desde el cual se puede trabajar, veamos una sesión de `gnuplot`:

```
jrogan@huelen:~$ gnuplot

        G N U P L O T
        Version 3.7 patchlevel 2
        last modified Sat Jan 19 15:23:37 GMT 2002
        System: Linux 2.4.19

        Copyright(C) 1986 - 1993, 1998 - 2002
        Thomas Williams, Colin Kelley and many others

        Type ‘help‘ to access the on-line reference manual
        The gnuplot FAQ is available from
        http://www.gnuplot.info/gnuplot-faq.html

        Send comments and requests for help to <info-gnuplot@dartmouth.edu>
        Send bugs, suggestions and mods to <bug-gnuplot@dartmouth.edu>


Terminal type set to 'x11'
gnuplot> plot sqrt(x)
gnuplot> set xrange[0:5]
gnuplot> set xlabel" eje de las x"
gnuplot> replot
gnuplot> set terminal postscript
Terminal type set to 'postscript'
Options are 'landscape noenhanced monochrome dashed defaultplex "Helvetica" 14'
gnuplot> set output "mygraph.ps"
gnuplot> replot
gnuplot> set terminal X
Terminal type set to 'X11'
Options are '0'
gnuplot> set xrange[-2:2]
gnuplot> set yrange[-2:2]
gnuplot> splot exp(-x*x-y*y)
gnuplot> plot "myfile.dat" w l
```

```
gnuplot> exit
jrogan@huelen:~$
```

En el caso de `xmgrace` y `SciGraphica` mucho más directo manejarlo ya que está basado en menús. Además, existe abundante documentación de ambos *softwares*. El *software* `SciGraphica` es una aplicación de visualización y análisis de data científica que permite el despliegue de gráficos en 2 y 3 dimensiones, además, exporta los resultados a formato *PostScript*. Realmente esta aplicación nació como un intento de clonar el programa comercial `origen` no disponible para Linux.

10.7. Graficando desde nuestros programas.

Finalmente para poder graficar desde nuestro propio programa necesitamos alguna biblioteca gráfica, en nuestro caso usaremos la biblioteca `iglu`, hecha completamente en casa. El comando de compilación incluido en un *script*, que llamaremos `iglu_compila`, y que contiene las siguientes líneas:

```
#!/bin/bash
g++ -Wall -O3 -o $1 $1.cc -L. -L/usr/X11R6/lib/ -liglu -lX11 -lm
```

Veamos algunos ejemplos:

```
/* Ejemplo: sen(x) */
#include <cmath>

#include ‘‘iglu.h’’

int main()
{
   IgluDibujo v;
   const int N=100;
   double x[N], y[N];
   v.map_coordinates(0,2*M_PI,-1.2,1.2);
   double dx = 2*M_PI/(N-1);
   for (int i=0;i<N;i++){
     x[i] = i*dx;
     y[i] = sin(x[i]);
   }

   v.plot_line(x,y,N);
   v.flush();
   v.wait();
   return 0;
}
```

Este programa grafica la función seno con un número de puntos dado.

Otro caso, una primitiva animación

```
/* Ejemplo  sen(x-vt) */
#include <cmath>

#include "iglu.h"

int main(){
  IgluDibujo v;
  const int N=100, Nt=100;
  double x[N], y[N];
  v.map_coordinates(0,2*M_PI,-1.2,1.2);
  double dx = 2*M_PI/(N-1), dt = .1, t=0;
  for (int j=0;j<Nt;j++){
    v.clean();
    t += dt*j;
    for (int i=0;i<N;i++){
      x[i] = i*dx;
      y[i] = sin(x[i]-.1*t);
    }
    v.plot_line(x,y,N);
    v.wait(1);
    v.flush();
  }
  v.wait();
  return 0;
}
```

Capítulo 11

Una breve introducción a Octave/Matlab

versión 2.0-021217

11.1. Introducción

Octave es un poderoso software para análisis numérico y visualización. Muchos de sus comandos son compatibles con Matlab. En estos apuntes revisaremos algunas características de estos programas. En realidad, el autor de este capítulo ha sido usuario durante algunos años de Matlab, de modo que estos apuntes se han basado en ese conocimiento, considerando los comandos que le son más familiares de Matlab. En la mayoría de las ocasiones he verificado que los comandos descritos son también compatibles con Octave, pero ocasionalmente se puede haber omitido algo....

Matlab es una abreviación de *Matrix Laboratory*. Los elementos básicos con los que se trabaja con matrices. Todos los otros tipos de variables (vectores, texto, polinomios, etc.), son tratados como matrices. Esto permite escribir rutinas optimizadas para el trabajo con matrices, y extender su uso a todos los otros tipos de variables fácilmente.

11.2. Interfase con el programa

Con Octave/Matlab se puede interactuar de dos modos: un modo interactivo, o a través de *scripts*. Al llamar a Octave/Matlab (escribiendo `octave` en el prompt, por ejemplo), se nos presenta un prompt. Si escribimos `a=1`, el programa responderá `a=1`. Alternativamente, podemos escribir `a=3;` (con punto y coma al final), y el programa no responderá (elimina el eco), pero almacena el nuevo valor de `a`. Si a continuación escribimos `a`, el programa responderá `a=3`. Hasta este punto, hemos usado el modo interactivo.

Alternativamente, podemos introducir las instrucciones anteriores en un archivo, llamado, por ejemplo, `prueba.m`. En el prompt, al escribir `prueba`, y si nuestro archivo está en el path de búsqueda del programa, las líneas de `prueba.m` serán ejecutadas una a una. Por ejemplo, si el archivo consta de las siguientes cuatro líneas:

```
a=3;
a
```

```
a=5
a
```

el programa responderá con

```
a=3
a=5
a=5
```

`prueba.m` corresponde a un *script*. Todas las instrucciones de Octave/Matlab pueden ejecutarse tanto en modo interactivo como desde un *script*. En Linux se puede ejecutar un archivo de comandos Octave de modo *stand-alone* incluyendo en la primera línea:
`#!/usr/bin/octave -q`.

11.3. Tipos de variables

11.3.1. Escalares

A pesar de que éstos son sólo un tipo especial de matrices (ver subsección siguiente), conviene mencionar algunas características específicas.

- Un número sin punto decimal es tratado como un entero exacto. Un número con punto decimal es tratado como un número en doble precisión. Esto puede no ser evidente en el output. Por *default*, `8.4` es escrito en pantalla como `8.4000`. Tras la instrucción `format long`, sin embargo, es escrito como `8.40000000000000`. Para volver al formato original, basta la instrucción `format`.
- Octave/Matlab acepta números reales y complejos. La unidad imaginaria es `i`: `8i` y `8*i` definen el mismo número complejo. Como `i` es una varible habitualmente usada en iteraciones, también está disponible `j` como un sinónimo. Octave/Matlab distinguen entre mayúsculas y minúsculas.
- Octave/Matlab representa de manera especial los infinitos y cantidades que no son números. `inf` es infinito, y `NaN` es un no-número (Not-a-Number). Por ejemplo, escribir `a=1/0` no arroja un error, sino un mensaje de advertencia, y asigna a `a` el valor `inf`. Análogamente, `a=0/0` asigna a `a` el valor `NaN`.

11.3.2. Matrices

Este tipo de variable corresponde a escalares, vectores fila o columna, y matrices convencionales.

Construcción

Las instrucciones:

```
a = [1 2 ; 3 4]
```

ó

```
a = [1, 2; 3, 4]
```

definen la matriz $\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$. Las comas (opcionales) separan elementos de columnas distintas, y los punto y coma separan elementos de filas distintas. El vector fila $(1 \quad 2)$ es

```
b = [1 2]
```

y el vector columna $\begin{pmatrix} 1 \\ 2 \end{pmatrix}$ es

```
c = [1;2]
```

Un número se define simplemente como `d = [3]` ó `d = 3`.

Nota importante: Muchas funciones de Octave/Matlab en las páginas siguientes aceptan indistintamente escalares, vectores filas, vectores columnas, o matrices, y su output es un escalar, vector o matriz, respectivamente. Por ejemplo, `log(a)` es un vector fila si `a` es un vector fila (donde cada elemento es el logaritmo natural del elemento correspondiente en `a`), y un vector columna si `a` es un vector columna. En el resto de este manual *no se advertira* este hecho, y se pondrán ejemplos con un solo tipo de variable, en el entendido que el lector está conciente de esta nota.

Acceso y modificación de elementos individuales

Accesamos los elementos de cada matriz usando los índices de filas y columnas, que parten de uno. Usando la matriz `a` antes definida, `a(1,2)` es 2. Para modificar un elemento, basta escribir, por ejemplo, `a(2,2) = 5`. Esto convierte a la matriz en $\begin{pmatrix} 1 & 2 \\ 3 & 5 \end{pmatrix}$. En el caso especial de vectores filas o columnas, basta un índice. (En los ejemplos anteriores, `b(2)` $= 2$ `c(2)` $= 2$.)

Una característica muy importante del programa es que toda matriz es redimensionada automáticamente cuando se intenta modificar un elemento que sobrepasa las dimensiones actuales de la matriz, llenando con ceros los lugares necesarios. Por ejemplo, si `b = [1 2]`, y en seguida intentamos la asignación `b(5) = 8`, `b` es automáticamente convertido al vector fila de 5 elementos `[1 2 0 0 8]`.

Concatenación de matrices

Si $a = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$, $b = \begin{pmatrix} 5 & 6 \end{pmatrix}$, $c = \begin{pmatrix} 7 \\ 8 \end{pmatrix}$, entonces

```
d = [a c]
```

$d = \begin{pmatrix} 1 & 2 & 7 \\ 3 & 4 & 8 \end{pmatrix}$

```
d = [a; b]
```

$d = \begin{pmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{pmatrix}$

```
d = [a [0; 0] c]
```

$$d = \begin{pmatrix} 1 & 2 & 0 & 7 \\ 3 & 4 & 0 & 8 \end{pmatrix}$$

11.3.3. Strings

Las cadenas de texto son casos particulares de vectores fila, y se construyen y modifican de modo idéntico.

Construcción

Las instrucciones

```
t = ['un buen texto']
t = ["un buen texto"]
t = 'un buen texto'
t = "un buen texto"
```

definen el mismo string `t`.

Acceso y modificación de elementos individuales

```
   r = t(4)

   r = 'b'
t(9) = 's'

texto = 'un buen sexto'
```

Concatenación

```
   t = 'un buen texto';
   t1 = [t ' es necesario']

   t1 = 'un buen texto es necesario'
```

11.3.4. Estructuras

Las estructuras son extensiones de los tipos de variables anteriores. Una estructura consta de distintos campos, y cada campo puede ser una matriz (es decir, un escalar, un vector o una matriz), o una string.

Construcción

Las líneas

```
persona.nombre = 'Eduardo'
persona.edad = 30
persona.matriz_favorita = [2 8;10 15];
```

definen una estructura con tres campos, uno de los cuales es un string, otro un escalar, y otro una matriz:

```
persona =
{
nombre = 'Eduardo';
edad = 30;
matriz_favorita = [2 8; 10 15];
}
```

Acceso y modificación de elementos individuales

```
s = persona.nombre
```

```
s = 'Eduardo'
persona.nombre = 'Claudio'
persona.matriz_favorita(2,1) = 8

persona =
{
nombre = 'Claudio';
edad = 30;
matriz_favorita = [2 8; 8 15];
}
```

11.4. Operadores básicos

11.4.1. Operadores aritméticos

Los operadores `+`, `-`, `*` corresponden a la suma, resta y multiplicación convencional de matrices. Ambas matrices deben tener la misma dimensión, a menos que una sea un escalar. Un escalar puede ser sumado, restado o multiplicado de una matriz de cualquier dimensión.

`.*` y `./` permiten multiplicar y dividir elemento por elemento. Por ejemplo, si

$$a = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} \qquad b = \begin{pmatrix} 5 & 6 \\ 7 & 8 \end{pmatrix}$$

entonces

```
c = a.*b
```

$$c = \begin{pmatrix} 5 & 12 \\ 21 & 32 \end{pmatrix}$$

```
c = a./b
```

$$c = \begin{pmatrix} 0{,}2 & 0{,}3333 \\ 0{,}42857 & 0{,}5 \end{pmatrix}$$

Si `b` es un escalar, `a.*b` y `a./b` equivalen a `a*b` y `a/b`.
`a^b` es `a` elevado a `b`, si `b` es un escalar. `a.^b` eleva cada elemento de `a` a `b`.
`a'` es la matriz $\mathtt{a}^\dagger$ (traspuesta y conjugada)
`a.'` es la matriz traspuesta de `a`.

11.4.2. Operadores relacionales

Los siguientes operadores están disponibles:
`<  <=  >  >=  ==  ~=`
El resultado de estas operaciones es 1 (verdadero) ó 0 (falso). Si uno de los operandos es una matriz y el otro un escalar, se compara el escalar con cada elemento de la matriz. Si ambos operandos son matrices, el test se realiza elemento por elemento; en este caso, las matrices deben ser de igual dimensión. Por ejemplo,

```
a = [1 2 3];
b = [4 2 1];
c = (a<3);
d = (a>=b);
```

$c = (1, 1, 0)$
$d = (0, 1, 1)$

11.4.3. Operadores lógicos

Los siguientes símbolos corresponden a los operadores `AND`, `OR` y `NOT`:
`&  |  ~`
El resultado de estas operaciones es 1 (verdadero) ó 0 (falso).

11.4.4. El operador :

Es uno de los operadores fundamentales. Permite crear vectores y extraer submatrices.

`:` crea vectores de acuerdo a las siguientes reglas:

`j:k`	es lo mismo que `[j,j+1,...,k]`, si `j<=k`.
`j:i:k`	es lo mismo que `[j,j+i,j+2*i,...,k]`, si `i>0` y `j<k`, o si `i<0` y `j>k`.

`:` extrae submatrices de acuerdo a las siguientes reglas:

`A(:,j)`	es la `j`-ésima columna de `A`.
`A(i,:)`	es la `i`-ésima fila de `A`.
`A(:,:)`	es `A`.
`A(:,j:k)`	es `A(:,j)`, `A(:,j+1)`, ..., `A(:,k)`.
`A(:)`	son todos los elementos de `A`, agrupados en una única columna.

11.4.5. Operadores de aparición preferente en scripts

Los siguientes operadores es más probable que aparezcan durante la escritura de un *script* que en modo interactivo.

`%` : Comentario. El resto de la línea es ignorado.
`...` : Continuación de línea. Si una línea es muy larga y no cabe en la pantalla, o por alguna otra razón se desea dividir una línea, se puede usar el operador `...` . Por ejemplo,

```
m = [1 2 3 ...
4 5 6];
```

es equivalente a

```
m = [1 2 3 4 5 6];
```

11.5. Comandos matriciales básicos

Antes de revisar una a una diversas familias de comandos disponibles, y puesto que las matrices son el elemento fundamental en Octave/Matlab, en esta sección reuniremos algunas de las funciones más frecuentes sobre matrices, y cómo se realizan en Octave/Matlab.

Op. aritmética	`+`, `-`, `*`, `.*`, `/`	(ver subsección 11.4.1)
Conjugar	`conj(a)`	
Trasponer	`a.'`	
Trasponer y conjugar	`a'`	
Invertir	`inv(a)`	
Autovalores, autovectores	`[v,d]=eig(a)`	(ver subsección 11.6.5)
Determinante	`det(a)`	
Extraer elementos	`:`	(ver subsección 11.4.4)
Traza	`trace(a)`	
Dimensiones	`size(a)`	
Exponencial	`exp(a)`	(elemento por elemento)
	`expm(a)`	(exponencial matricial)

11.6. Comandos

En esta sección revisaremos diversos comandos de uso frecuente en Octave/Matlab. Esta lista no pretende ser exhaustiva (se puede consultar la documentación para mayores detalles), y está determinada por mi propio uso del programa y lo que yo considero más frecuente debido a esa experiencia. Insistimos en que ni la lista de comandos es exhaustiva, ni la lista de ejemplos o usos de cada comando lo es. Esto pretende ser sólo una descripción de los aspectos que me parecen más importantes o de uso más recurrente.

11.6.1. Comandos generales

`clear` Borra variables y funciones de la memoria

`clear`	Borra todas las variables en memoria
`clear a`	Borra la variable `a`

`disp` Presenta matrices o texto

`disp(a)` presenta en pantalla los contenidos de una matriz, sin imprimir el nombre de la matriz. `a` puede ser una string.

```
disp('     c1       c2');
disp([.3 .4]);
```

$c1$	$c2$
0,30000	0,40000

`load, save` Carga/Guarda variables desde el disco

`save fname a b`	Guarda las variables `a` y `b` en el archivo `fname`
`load fname`	Lee el archivo `fname`, cargando las definiciones de variables en él definidas.

`size,length` Dimensiones de una matriz/largo de un vector

Si a es una matrix de $n \times m$:

`d = size(a)`	`d = [m,n]`
`[m,n] = size(a)`	Aloja en `m` el número de filas, y en `n` el de columnas

Si b es un vector de n elementos, `length(b)` es n.

`who` Lista de variables en memoria

`quit` Termina Octave/Matlab

11.6.2. Como lenguaje de programación

Control de flujo

`for`

```
n=3;                                    a=[1 4 9]
for i=1:n
    a(i)=i^2;
end
```

Para Octave el vector resultante es columna en vez de fila.

Observar el uso del operador `:` para generar el vector `[1 2 3]`. Cualquier vector se puede utilizar en su lugar: `for i=[2 8 9 -3]`, `for i=10:-2:1` (equivalente a `[10 8 6 4 2]`), etc. son válidas. El ciclo `for` anterior se podría haber escrito en una sola línea así:

```
for i=1:n, a(i)=i^2; end
```

`if, elseif, else`

Ejemplos:

a) `if a~=b, disp(a); end`

b)
```
if a==[3 8 9 10]
   b = a(1:3);
end
```

c)
```
if a>3
   clear a;
elseif a<0
   save a;
else
   disp('Valor de a no considerado');
end
```

Naturalmente, `elseif` y `else` son opcionales. En vez de las expresiones condicionales indicadas en el ejemplo pueden aparecer cualquier función que dé valores `1` (verdadero) ó `0` (falso).

`while`

```
while s
    comandos
end
```

Mientras `s` es `1`, se ejecutan los `comandos` entre `while` y `end`. `s` puede ser cualquier expresión que dé por resultado `1` (verdadero) ó `0` (falso).

`break`

Interrumpe ejecución de ciclos `for` o `while`. En *loops* anidados, `break` sale del más interno solamente.

Funciones lógicas

Además de expresiones construidas con los operadores relacionales `==`, `<=`, etc., y los operadores lógicos `&`, `|` y `~`, los comandos de control de flujo anteriores admiten cualquier función cuyo resultado sea `1` (verdadero) ó `0` (falso). Particularmente útiles son funciones como las siguientes:

`all(a)`	`1` si todos los elementos de `a` son no nulos, y `0` si alguno es cero
`any(a)`	`1` si alguno de los elementos de `a` es no nulo
`isempty(a)`	`1` si `a` es matriz vacía (`a=[]`)

Otras funciones entregan matrices de la misma dimensión que el argumento, con unos o ceros en los lugares en que la condición es verdadera o falsa, respectivamente:

`finite(a)`	`1` donde `a` es finito (no `inf` ni `NaN`)
`isinf(a)`	`1` donde `a` es infinito
`isnan(a)`	`1` donde `a` es un `NaN`

Por ejemplo, luego de ejecutar las líneas

```
x = [-2 -1 0 1 2];
y = 1./x;
a = finite(y);
b = isinf(y);
c = isnan(y);
```

se tiene

```
a = [1 1 0 1 1]
b = [0 0 1 0 0]
c = [0 0 0 0 0]
```

Otra función lógica muy importante es `find`:

`find(a)` Encuentra los índices de los elementos no nulos de `a`.

Por ejemplo, si ejecutamos las líneas

```
x=[11 0 33 0 55];
z1=find(x);
z2=find(x>0 & x<40);
```

obtendremos

```
z1 = [1 3 5]
z2 = [1 3]
```

`find` también puede dar dos resultados de salida simultáneamente (más sobre esta posibilidad en la sección 11.6.2), en cuyo caso el resultado son los pares de índices (índices de fila y columna) para cada elemento no nulo de una matriz

```
y=[1 2 3 4 5;6 7 8 9 10];
[z3,z4]=find(y>8);
```

da como resultado

```
z3 = [2;2];
z4 = [4;5];
```

`z3` contiene los índice de fila y `z4` los de columna para los elementos no nulos de la matriz `y>8`. Esto permite construir, por ejemplo, la matriz `z5=[z3 z4]` $= \begin{pmatrix} 2 & 4 \\ 2 & 5 \end{pmatrix}$, en la cual cada fila es la posición de `y` tal que la condición `y>8` es verdadera (en este caso, es verdadera para los elementos `y(2,4)` e `y(2,5)`).

Funciones definidas por el usuario

Octave/Matlab puede ser fácilmente extendido por el usuario definiendo nuevas funciones que le acomoden a sus propósitos. Esto se hace a través del comando `function`.

Podemos definir (en modo interactivo o dentro de un *script*), una función en la forma

```
function nombre (argumentos)
   comandos
endfunction
```

`argumentos` es una lista de argumentos separados por comas, y `comandos` es la sucesión de comandos que serán ejecutados al llamar a `nombre`. La lista de argumentos es opcional, en cuyo caso los paréntesis redondos se pueden omitir.

A mediano y largo plazo, puede ser mucho más conveniente definir las funciones en archivos especiales, listos para ser llamados en el futuro desde modo interactivo o desde cualquier *script*. Esto se hace escribiendo la definición de una función en un *script* con extensión `.m`. Cuando Octave/Matlab debe ejecutar un comando o función que no conoce, por ejemplo, `suma(x,y)`,busca en los archivos accesibles en su path de búsqueda un archivo llamado `suma.m`, lo carga y ejecuta la definición contenida en ese archivo.

Por ejemplo, si escribimos en el *script* `suma.m` las líneas

```
function s=suma(x,y)
s = x+y;
```

el resultado de `suma(2,3)` será `5`.

Las funciones así definidas pueden entregar más de un argumento si es necesario (ya hemos visto algunos ejemplos con `find` y `size`). Por ejemplo, definimos una función que efectúe un análisis estadístico básico en `stat.m`:

```
function [mean,stdev] = stat(x)
n = length(x);
mean = sum(x)/n;
stdev = sqrt(sum((x-mean).^2/n));
```

Al llamarla en la forma `[m,s] = stat(x)`, si `x` es un vector fila o columna, en `m` quedará el promedio de los elementos de `x`, y en `s` la desviación estándard.

Todas las variables dentro de un *script* que define una función son locales, a menos que se indique lo contrario con `global`. Por ejemplo, si un *script* `x.m` llama a una función `f`, y dentro de `f.m` se usa una variable `a` que queremos sea global, ella se debe declarar en la forma `global a` tanto en `f.m` como en el *script* que la llamó, `x.m`, y en todo otro *script* que pretenda usar esa variable global.

11.6.3. Matrices y variables elementales

Matrices constantes importantes

Las siguientes son matrices que se emplean habitualmente en distintos contextos, y que es útil tener muy presente:

`eye(n)`	Matriz identidad de $n \times n$
`ones(m,n)`	Matriz de $m \times n$, con todos los elementos igual a 1.
`rand(m,n)`	Matriz de $m \times n$ de números al azar, distribuidos uniformemente.
`randn(m,n)`	Igual que `rand`, pero con distribución normal (Gaussiana).
`zeros(m,n)`	Igual que `ones`, pero con todos los elementos 0.

Matrices útiles para construir ejes o mallas para graficar

`v = linspace(min,max,n)`	Vector cuyo primer elemento es `min`, su último elemento es `max`, y tiene `n` elementos equiespaciados.
`v = logspace(min,max,n)`	Análogo a `linspace`, pero los `n` elementos están espaciados logarítmicamente.
`[X,Y] = meshgrid(x,y)`	Construye una malla del plano x-y. Las filas de `X` son copias del vector `x`, y las columnas de `Y` son copias del vector `y`.

Por ejemplo:

```
x = [1 2 3];
y = [4 5];
[X,Y] = meshgrid(x,y);
```

da

$$X = \begin{pmatrix} 1 & 2 & 3 \\ 1 & 2 & 3 \end{pmatrix} , \qquad Y = \begin{pmatrix} 4 & 4 & 4 \\ 5 & 5 & 5 \end{pmatrix} .$$

Notemos que al tomar sucesivamente los pares ordenados (`X(1,1),Y(1,1)`), (`X(1,2),Y(1,2)`), (`X(1,3),Y(1,3)`), etc., se obtienen todos los pares ordenados posibles tales que el primer elemento está en `x` y el segundo está en `y`. Esta característica hace particularmente útil el comando `meshgrid` en el contexto de gráficos de funciones de dos variables (ver secciones 11.6.7, 11.6.7).

Constantes especiales

Octave/Matlab proporciona algunos números especiales, algunos de los cuales ya mencionamos en la sección 11.3.1.

`i`, `j`	Unidad imaginaria ($\sqrt{-1}$)
`inf`	Infinito
`NaN`	Not-A-Number
`pi`	El número π (= 3,1415926535897...)

Funciones elementales

Desde luego, Octave/Matlab proporciona todas las funciones matemáticas básicas. Por ejemplo:

a) Funciones sobre números reales/complejos

`abs`	Valor absoluto de números reales, o módulo de números imaginarios
`angle`	Ángulo de fase de un número imaginario
`conj`	Complejo conjugado
`real`	Parte real
`imag`	Parte imaginaria
`sign`	Signo
`sqrt`	Raíz cuadrada

b) Exponencial y funciones asociadas

`cos`, `sin`, etc.	Funciones trigonométricas
`cosh`, `sinh`, etc.	Funciones hiperbólicas
`exp`	Exponencial
`log`	Logaritmo

c) Redondeo

`ceil`	Redondear hacia $+\infty$
`fix`	Redondear hacia cero
`floor`	Redondear hacia $-\infty$
`round`	Redondear hacia el entero más cercano

Funciones especiales

Además, Octave/Matlab proporciona diversas funciones matemáticas especiales. Algunos ejemplos:

`bessel`	Función de Bessel
`besselh`	Función de Hankel
`beta`	Función beta
`ellipke`	Función elíptica
`erf`	Función error
`gamma`	Función gamma

Así, por ejemplo, `bessel(alpha,X)` evalúa la función de Bessel de orden `alpha`, $J_\alpha(x)$, para cada elemento de la matriz X.

11.6.4. Polinomios

Octave/Matlab representa los polinomios como vectores fila. El polinomio

$$p = c_n x^n + \cdots + c_1 x + c_0$$

es representado en Octave/Matlab en la forma

```
p = [c_n, ..., c1, c0]
```

Podemos efectuar una serie de operaciones con los polinomios así representados.

`poly(x)`	Polinomio cuyas raíces son los elementos de `x`.
`polyval(p,x)`	Evalúa el polinomio `p` en `x` (en los elementos de `x` si éste es un vector)
`roots(p)`	Raíces del polinomio `p`

11.6.5. Álgebra lineal (matrices cuadradas)

Unos pocos ejemplos, entre los comandos de uso más habitual:

`det`	Determinante
`rank`	Número de filas o columnas linealmente independientes
`trace`	Traza
`inv`	Matriz inversa
`eig`	Autovalores y autovectores
`poly`	Polinomio característico

Notar que `poly` es la misma función de la sección 11.6.4 que construye un polinomio de raíces dadas. En el fondo, construir el polinomio característico de una matriz es lo mismo, y por tanto tiene sentido asignarles la misma función. Y no hay confusión, pues una opera sobre vectores y la otra sobre matrices cuadradas.

El uso de todos estos comandos son autoexplicativos, salvo `eig`, que se puede emplear de dos modos:

```
d = eig(a)
[V,D] = eig(a)
```

La primera forma deja en `d` un vector con los autovalores de `a`. La segunda, deja en `D` una matriz diagonal con los autovalores, y en `V` una matiz cuyas columnas son los autovalores, de modo que `A*V = V*D`. Por ejemplo, si `a =[1 2; 3 4]`, entonces

$$d = \begin{pmatrix} 5{,}37228 \\ -0{,}37228 \end{pmatrix}$$

y

$$D = \begin{pmatrix} 5{,}37228\ldots & 0 \\ 0 & -0{,}37228\ldots \end{pmatrix}, \qquad V = \begin{pmatrix} 0{,}41597\ldots & -0{,}82456\ldots \\ 0{,}90938\ldots & 0{,}56577\ldots \end{pmatrix}.$$

La primera columna de `V` es el autovector de `a` asociado al primer autovalor, $5{,}37228\ldots$.

11.6.6. Análisis de datos y transformada de Fourier

En Octave/Matlab están disponibles diversas herramientas para el análisis de series de datos (estadística, correlaciones, convolución, etc.). Algunas de las operaciones básicas son:

a) Máximos y mínimos
   Si `a` es un vector, `max(a)` es el mayor elemento de `a`. Si es una matriz, `max(a)` es un vector fila, que contiene el máximo elemento para cada columna.

```
a = [1 6 7; 2 8 3; 0 4 1]
b = max(a)
```
$b = (2 \quad 8 \quad 7)$

Se sigue que el mayor elemento de la matriz se obtiene con `max(max(a))`.

`min` opera de modo análogo, entregando los mínimos.

b) Estadística básica
Las siguientes funciones, como `min` y `max`, operan sobre vectores del modo usual, y sobre matrices entregando vectores fila, con cada elemento representando a cada columna de la matriz.

`mean`	Valor promedio
`median`	Mediana
`std`	Desviación standard
`prod`	Producto de los elementos
`sum`	Suma de los elementos

c) Orden
`sort(a)` ordena los elementos de `a` en orden ascendente si `a` es un vector. Si es una matriz, ordena cada columna.

```
b = sort([1 3 9; 8 2 1; 4 -3 0]);
```
$$b = \begin{pmatrix} 1 & -3 & 0 \\ 4 & 2 & 1 \\ 8 & 3 & 9 \end{pmatrix}$$

d) Transformada de Fourier

Por último, es posible efectuar transformadas de Fourier directas e inversas, en una o dos dimensiones. Por ejemplo, `fft` y `ifft` dan la transformada de Fourier y la transformada inversa de `x`, usando un algoritmo de *fast Fourier transform* (FFT). Específicamente, si `X=fft(x)` y `x=ifft(X)`, y los vectores son de largo `N`:

$$X(k) = \sum_{j=1}^{N} x(j)\omega_N^{(j-1)(k-1)} \ ,$$
$$x(j) = \frac{1}{N}\sum_{k=1}^{N} X(k)\omega_N^{-(j-1)(k-1)} \ ,$$

donde $\omega_N = e^{-2\pi i/N}$.

11.6.7. Gráficos

Una de las características más importantes de Matlab son sus amplias posibilidades gráficas. Algunas de esas características se encuentran también en Octave. En esta sección revisaremos el caso de gráficos en dos dimensiones, en la siguiente el caso de tres dimensiones, y luego examinaremos algunas posibilidades de manipulación de gráficos.

Gráficos bidimensionales

Para graficar en dos dimensiones se usa el comando `plot`. `plot(x,y)` grafica la ordenada `y` versus la abscisa `x`. `plot(y)` asume abscisa `[1,2,...n]`, donde `n` es la longitud de `y`.

Ejemplo: Si `x=[2 8 9]`, `y=[6 3 2]`, entonces
`plot(x,y)`

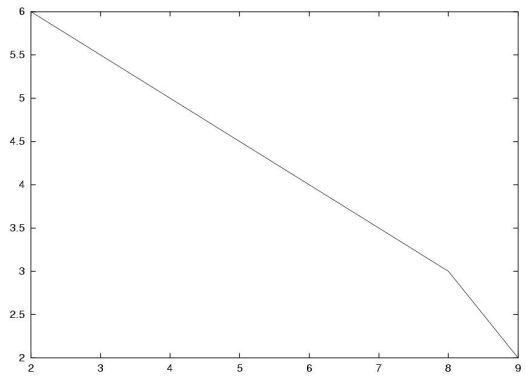


Figura 11.1: Gráfico simple.

Por *default*, Octave utiliza `gnuplot` para los gráficos. Por *default*, los puntos se conectan con una línea roja en este caso. El aspecto de la línea o de los puntos puede ser modificado. Por ejemplo, `plot(x,y,'ob')` hace que los puntos sean indicados con círculos (`'o'`) azules (`'b'`, *blue*). Otros modificadores posibles son:

`-`	línea (*default*)	`r`	red
`.`	puntos	`g`	green
`@`	otro estilo de puntos	`b`	blue
`+`	signo más	`m`	magenta
`*`	asteriscos	`c`	cyan
`o`	círculos	`w`	white
`x`	cruces		

Dos o más gráficos se pueden incluir en el mismo output agregando más argumentos a `plot`. Por ejemplo: `plot(x1,y1,'x',x2,y2,'og',x3,y3,'.c')`.

Los mapas de contorno son un tipo especial de gráfico. Dada una función $z = f(x, y)$, nos interesa graficar los puntos (x, y) tales que $f = c$, con c alguna constante. Por ejemplo, consideremos

$$z = xe^{-x^2-y^2} \ , \quad x \in [-2, 2], \ y \in [-2, 3] \ .$$

Para obtener el gráfico de contorno de z, mostrando los niveles $z = -$,3, $z = -$,1, $z = 0$, $z =$,1 y $z =$,3, podemos usar las instrucciones:

```
x = -2:.2:2;
y = -2:.2:3;
[X,Y] = meshgrid(x,y);
```

```
Z = X.*exp(-X.^2-Y.^2);
contour(Z.',[-.3 -.1 0 .1 .3],x,y);   # Octave por default (gnuplot)
contour(x, y, Z.',[-.3 -.1 0 .1 .3]); # Octave con plplot y Matlab
```

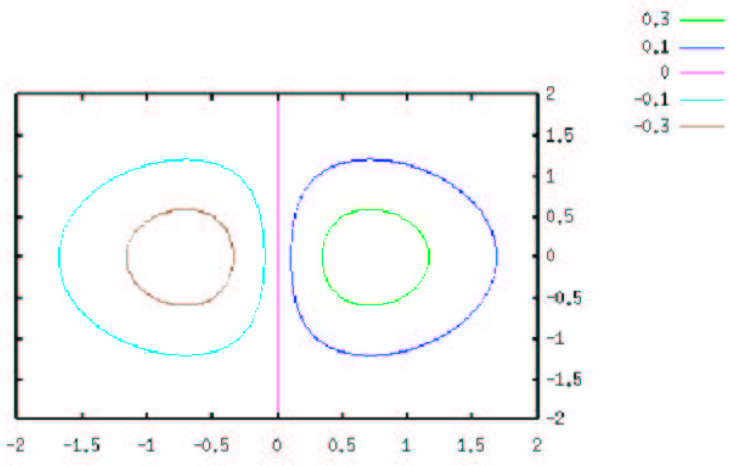


Figura 11.2: Curvas de contorno.

Las dos primeras líneas definen los puntos sobre los ejes x e y en los cuales la función será evaluada. En este caso, escojimos una grilla en que puntos contiguos están separados por `.2`. Para un mapa de contorno, necesitamos evaluar la función en todos los pares ordenados (x, y) posibles al escoger x en `x` e y en `y`. Para eso usamos `meshgrid` (introducida sin mayores explicaciones en la sección 11.6.3). Luego evaluamos la función [`Z` es una matriz, donde cada elemento es el valor de la función en un par ordenado (x, y)], y finalmente construimos el mapa de contorno para los niveles deseados.

Gráficos tridimensionales

También es posible realizar gráficos tridimensionales. Por ejemplo, la misma doble gaussiana de la sección anterior se puede graficar en tres dimensiones, para mostrarla como una superficie $z(x, y)$. Basta reemplazar la última instrucción, que llama a `contour`, por la siguiente:

```
mesh(X,Y,Z)
```

Observar que, mientras `contour` acepta argumentos dos de los cuales son vectores, y el tercero una matriz, en `mesh` los tres argumentos son matrices de la misma dimensión (usamos `X`, `Y`, en vez de `x`, `y`).

Nota importante: Otro modo de hacer gráficos bi y tridimensionales es con `gplot` y `gsplot` (instrucciones asociadas realmente no a Octave sino a `gnuplot`, y por tanto no equivalentes a instrucciones en Matlab). Se recomienda consultar la documentación de Octave para los detalles.

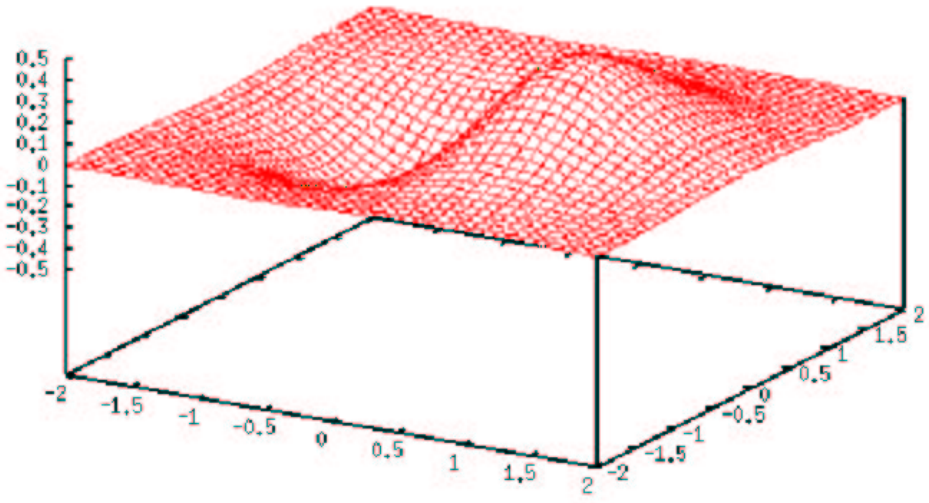


Figura 11.3: Curvas de contorno.

Manipulación de gráficos

Los siguientes comandos están disponibles para modificar gráficos construidos con Octave/Matlab:

a) Ejes

`axis([x1 y1 x2 y2])` Cambia el eje x al rango $(x1, x2)$, y el eje y al rango $(y1, y2)$.

b) Títulos

`title(s)`	Título (`s` es un string)
`xlabel(s)`	Título del eje x, y, z.
`ylabel(s)`	
`zlabel(s)`	

c) Grillas

`grid` Incluye o borra una grilla de referencia en un gráfico bidimensional. `grid ''on''` coloca la grilla y `grid ''off''` la saca. `grid` equivale a `grid ''on''`.

Al usar `gnuplot`, el gráfico mostrado en pantalla no es actualizado automáticamente. Para actualizarlo y ver las modificaciones efectuadas, hay que dar la instrucción `replot`.

Los siguientes comandos permiten manipular las ventanas gráficas:

`hold`	Permite "congelar" la figura actual, de modo que sucesivos comandos gráficos se superponen sobre dicha figura (normalmente la figura anterior es reemplazada por la nueva). `hold on` activa este "congelamiento", y `hold off` lo desactiva. `hold` cambia alternativamente entre el estado `on` y `off`.
`closeplot`	Cierra la ventana actual.

Finalmente, si se desea guardar un gráfico en un archivo, se puede proceder del siguiente modo si Octave está generando los gráficos con `gnuplot` y se trabaja en un terminal con XWindows. Si se desea guardar un gráfico de la función $y = x^3$, por ejemplo:

```
x = linspace(1,10,30);
y = x.^3;
plot(x,y);
gset term postscript color
gset output ‘‘xcubo.ps’’
replot
gset term x11
```

Las tres primeras líneas son los comandos de Octave/Matlab convencionales para graficar. Luego se resetea el terminal a un terminal postscript en colores (`gset term postscript` si no deseamos los colores), para que el output sucesivo vaya en formato postscript y no a la pantalla. La siguiente línea indica que la salida es al archivo `xcubo.ps`. Finalmente, se redibuja el gráfico (con lo cual el archivo `xcubo.ps` es realmente generado), y se vuelve al terminal XWindows para continuar trabajando con salida a la pantalla.

Debemos hacer notar que no necesariamente el gráfico exportado a *Postscript* se verá igual al resultado que `gnuplot` muestra en pantalla. Durante la preparación de este manual, nos dimos cuenta de ello al intentar cambiar los estilos de línea de `plot`. Queda entonces advertido el lector.

11.6.8. Strings

Para manipular una cadena de texto, disponemos de los siguientes comandos:

`lower`	Convierte a minúsculas
`upper`	Convierte a mayúsculas

Así, `lower('Texto')` da `'texto'`, y `upper('Texto')` da `'TEXTO'`.

Para comparar dos matrices entre sí, usamos `strcmp`:

`strcmp(a,b)`	1 si `a` y `b` son idénticas, 0 en caso contrario

Podemos convertir números enteros o reales en strings, y strings en números, con los comandos:

`int2str`	Convierte entero en string
`num2str`	Convierte número en string
`str2num`	Convierte string en número

Por ejemplo, podemos usar esto para construir un título para un gráfico:

```
s = ['Intensidad transmitida vs. frecuencia, n = ', num2str(1.5)];
title(s);
```

Esto pondrá un título en el gráfico con el texto:
`Intensidad transmitida vs. frecuencia, n = 1.5`.

11.6.9. Manejo de archivos

Ocasionalmente nos interesará grabar el resultado de nuestros cálculos en archivos, o utilizar datos de archivos para nuevos cálculos. El primer paso es abrir un archivo:

```
archivo = fopen('archivo.dat','w');
```

Esto abre el archivo `archivo.dat` para escritura (`'w'`), y le asigna a este archivo un número que queda alojado en la variable `archivo` para futura referencia.

Los modos de apertura posibles son:

`r`	Abre para lectura
`w`	Abre para escritura, descartando contenidos anteriores si los hay
`a`	Abre o crea archivo para escritura, agregando datos al final del archivo si ya existe
`r+`	Abre para lectura y escritura
`w+`	Crea archivo para lectura y escritura
`a+`	Abre o crea archivo para lectura y escritura, agregando datos al final del archivo si ya existe

En un archivo se puede escribir en modo binario:

`fread`	Lee datos binarios
`fwrite`	Escribe datos binarios

o en modo texto

`fgetl`	Lee una línea del archivo, descarta cambio de línea
`fgets`	Lee una línea del archivo, preserva cambio de l'ınea
`fprintf`	Escribe datos siguiendo un formato
`fscanf`	Lee datos siguiendo un formato

Referimos al lector a la ayuda que proporciona Octave/Matlab para interiorizarse del uso de estos comandos. Sólo expondremos el uso de `fprintf`, pues el formato es algo que habitualmente se necesita tanto para escribir en archivos como en pantalla, y `fprintf` se puede usar en ambos casos.

La instrucción

```
fprintf(archivo,'formato',A,B,...)
```

imprime en el archivo asociado con el identificador `archivo` (asociado al mismo al usar `fopen`, ver más arriba), las variables `A`, `B`, etc., usando el formato `'formato'`. `archivo=1` corresponde a la pantalla; si `archivo` se omite, el *default* es `1`, es decir, `fprintf` imprime en pantalla si `archivo=1` o si se omite el primer argumento.

`'formato'` es una string, que puede contener caracters normales, caracteres de escape o especificadores de conversión. Los caracteres de escape son:

`\n`	New line
`\t`	Horizontal tab
`\b`	Backspace
`\r`	Carriage return
`\f`	Form feed
`\\`	Backslash
`\'`	Single quote

Por ejemplo, la línea

```
fprintf('Una tabulacion\t y un {\'o}riginal\'\' cambio de linea\n aqui\n')
```

da como resultado

```
Una tabulacion       y un ''original'' cambio de linea
aqui
```

Es importante notar que por *default*, el cambio de línea al final de un `fprintf` no existe, de modo que, si queremos evitar salidas a pantalla o a archivo poco estéticas, siempre hay que terminar con un `\n`.

Los especificadores de conversión permiten dar formato adecuado a las variables numéricas `A`, `B`, etc. que se desean imprimir. Constan del caracter `%`, seguido de indicadores de ancho (opcionales), y caracteres de conversión. Por ejemplo, si deseamos imprimir el número π con 5 decimales, la instrucción es:

```
fprintf('Numero pi = %.5f\n',pi)
```

El resultado:

```
Numero pi = 3.14159
```

Los caracteres de conversión pueden ser

`%e`	Notación exponencial (Ej.: `2.4e-5`)
`%f`	Notación con punto decimal fijo (Ej.: `0.000024`)
`%g`	`%e` o `%f`, dependiendo de cuál sea más corto (los ceros no significativos no se imprimen)

Entre `%` y `e`, `f`, o `g` según corresponda, se pueden agregar uno o más de los siguientes caracteres, en este orden:

- Un signo menos (`-`), para especificar alineamiento a la izquierda (a la derecha es el *default*).

- Un número entero para especificar un ancho mínimo del campo.
- Un punto para separar el número anterior del siguiente número.
- Un número indicando la precisión (número de dígitos a la derecha del punto decimal).

En el siguiente ejemplo veremos distintos casos posibles. El output fue generado con las siguientes instrucciones, contenidas en un *script*:

```
a = .04395;
fprintf('123456789012345\n');
fprintf('a = %.3f.\n',a);
fprintf('a = %10.2f.\n',a);
fprintf('a = %-10.2f.\n',a);
fprintf('a = %4f.\n',a);
fprintf('a = %5.3e.\n',a);
fprintf('a = %f.\n',a);
fprintf('a = %e.\n',a);
fprintf('a = %g.\n',a);
```

El resultado:

```
12345678901234567890
a = 0.044.
a =       0.04.
a = 0.04      .
a = 0.043950.
a = 4.395e-02.
a = 0.043950.
a = 4.395000e-02.
a = 0.04395.
```

En la primera línea, se imprimen tres decimales. En la segunda, dos, pero el ancho mínimo es 10 caracteres, de modo que se alínea a la derecha el output y se completa con blancos. En la tercera línea es lo mismo, pero alineado a la izquierda. En la cuarta línea se ha especificado un ancho mínimo de 4 caracteres; como el tamaño del número es mayor, esto no tiene efecto y se imprime el número completo. En la quinta línea se usa notación exponencial, con tres decimal (nuevamente, el ancho mínimo especificado, 5, es menor que el ancho del output, luego no tiene efecto). Las últimas tres líneas comparan el output de `%f`, `%e` y `%g`, sin otras especificaciones.

Si se desean imprimir más de un número, basta agregar las conversiones adecuadas y los argumentos en `fprintf`. Así, la línea

```
fprintf('Dos numeros arbitrarios: %g y %g.\n',pi,exp(4));
```

da por resultado

```
Dos numeros arbitrarios: 3.14159 y 54.5982.
```

Si los argumentos numéricos de `fprintf` son matrices, el formato es aplicado a cada columna hasta terminar la matriz. Por ejemplo, el *script*

```
x = 1:5;
y1 = exp(x);
y2 = log(x);
a = [x; y1; y2];
fprintf = ('%g  %8g %8.3f\n',a);
```

da el output

```
1   2.71828    0.000
2   7.38906    0.693
3   20.0855    1.099
4   54.5982    1.386
5   148.413    1.609
```

Capítulo 12

El sistema de preparación de documentos TEX .

versión 4.0 021217

12.1. Introducción.

TEX es un procesador de texto o, mejor dicho, un avanzado sistema de preparación de documentos, creado por Donald Knuth, que permite el diseño de documentos de gran calidad, conteniendo textos y fórmulas matemáticas. Años después, LATEX fue desarrollado por Leslie Lamport, facilitando la preparación de documentos en TEX, gracias a la definición de "macros" o conjuntos de comandos de fácil uso.

LATEX tuvo diversas versiones hasta la 2.09. Actualmente, LATEX ha recibido importantes modificaciones, siendo la distribución actualmente en uso y desarrollo LATEX 2_ε, una versión transitoria en espera de que algún día se llegue a la nueva versión definitiva de LATEX, LATEX3. En estas páginas cuando digamos LATEX nos referiremos a la versión actual, LATEX 2_ε. Cuando queramos hacer referencia a la versión anterior, que debería quedar progresivamente en desuso, diremos explícitamente LATEX 2.09.

12.2. Archivos.

El proceso de preparación de un documento LATEX consta de tres pasos:

1. Creación de un archivo con extensión `tex` con algún editor.

2. Compilación del archivo `tex`, con un comando del tipo `latex <archivo>.tex` o `latex <archivo>`. Esto da por resultado tres archivos adicionales, con el mismo nombre del archivo original, pero con extensiones distintas:

   a) `dvi`. Es el archivo procesado que podemos ver en pantalla o imprimir. Una vez compilado, este archivo puede ser enviado a otro computador, para imprimir en otra impresora, o verlo en otro monitor, independiente de la máquina (de donde su extensión `dvi`, *device independent*).

b) `log`. Aquí se encuentran todos los mensajes producto de la compilación, para consulta si es necesario (errores encontrados, memoria utilizada, mensajes de advertencia, etc.).

c) `aux`. Contiene información adicional que por el momento no nos interesa.

3. Visión en pantalla e impresión del archivo procesado a través de un programa anexo (`xdvi` o `dvips`, por ejemplo), capaz de leer el `dvi`.

12.3. Input básico.

12.3.1. Estructura de un archivo.

En un archivo no pueden faltar las siguientes líneas:

```
\documentclass[12pt]{article}

\begin{document}

\end{document}
```

Haremos algunas precisiones respecto a la primera línea más tarde. Lo importante es que una línea de esta forma debe ser la primera de nuestro archivo. Todo lo que se encuentra antes de `\begin{document}` se denomina *preámbulo*. El texto que queramos escribir va entre `\begin{document}` y `\end{document}`. Todo lo que se encuentre después de `\end{document}` es ignorado.

12.3.2. Caracteres.

Pueden aparecer en nuestro texto todos los caracteres del código ASCII no extendido (teclado inglés usual): letras, números y los signos de puntuación:

```
.  :  ;  ,  ?  !  ‘  ’  (  )  [  ]  -  /  * @
```

Los caracteres especiales:

```
#  $  %  &  ~  _  ^  \  {  }
```

tienen un significado específico para LaTeX. Algunos de ellos se pueden obtener anteponiéndoles un *backslash*:

```
#  \#     $  \$     %  \%     &  \&     {  \{     }  \}
```

Los caracteres

```
+  =  |  <  >
```

generalmente aparecen en fórmulas matemáticas, aunque pueden aparecer en texto normal. Finalmente, las comillas dobles (") casi nunca se usan.

Los espacios en blanco y el fin de línea son también caracteres (invisibles), que LaTeX considera como un mismo carácter, que llamaremos espacio, y que simbolizaremos ocasionalmente como ␣ .

Para escribir en castellano requeriremos además algunos signos y caracteres especiales:

ñ `\~n` á `\'a` í `\'{\i}` ü `\"u` ¡ `!‘` ¿ `?‘`

12.3.3. Comandos.

Todos los comandos comienzan con un backslash, y se extienden hasta encontrar el primer carácter que no sea una letra (es decir, un espacio, un número, un signo de puntuación o matemático, etc.).

12.3.4. Algunos conceptos de estilo.

LaTeX es consciente de muchas convenciones estilísticas que quizás no apreciamos cuando leemos textos bien diseñados, pero las cuales es bueno conocer para aprovecharlas.

a) Observemos la siguiente palabra: fino. Esta palabra fue generada escribiendo simplemente `fino`, pero observemos que las letras ‘f’ e ‘i’ no están separadas, sino que unidas artísticamente. Esto es una *ligadura*, y es considerada una práctica estéticamente preferible. LaTeX sabe esto e inserta este pequeño efecto tipográfico sin que nos demos cuenta.

b) Las comillas de apertura y de cierre son distintas. Por ejemplo: ‘insigne’ (comillas simples) o “insigne” (comillas dobles). Las comillas de apertura se hacen con uno o con dos acentos graves (‘), para comillas simples o dobles, respectivamente, y las de cierre con acentos agudos (’): `‘insigne’`, `‘‘insigne’’`. No es correcto entonces utilizar las comillas dobles del teclado e intentar escribir `"insigne"` (el resultado de esto es el poco estético ”insigne”).

c) Existen tres tipos de guiones:

Corto	Saint-Exupéry	`-`	(entre palabras, corte en sílabas al final de la línea)
Medio	páginas 1–2	`--`	(rango de números)
Largo	un ejemplo —como éste	`---`	(puntuación, paréntesis)

d) LaTeX inserta después de un punto seguido un pequeño espacio adicional respecto al espacio normal entre palabras, para separar sutilmente frases. Pero, ¿cómo saber que un punto termina una frase? El criterio que utiliza es que todo punto termina una frase cuando va precedido de una minúscula. Esto es cierto en la mayoría de los casos, así como es cierto que generalmente cuando un punto viene después de una mayúscula no hay fin de frase:

> China y U.R.S.S. estuvieron de acuerdo. Sin embargo...

Pero hay excepciones:

> En la pág. 11 encontraremos noticias desde la U.R.S.S. Éstas fueron entregadas...

Cuando estas excepciones se producen, nosotros, humanos, tenemos que ayudarle al computador, diciéndole que, aunque hay un punto después de la "g", no hay un fin de frase, y que el punto después de la última "S" sí termina frase. Esto se consigue así:

```
En la p\'ag.\ 11 encontraremos noticias desde la
U.R.S.S\@. \'Estas fueron entregadas...
```

d) Énfasis de texto:

Éste es un texto *enfatizado*.	`\'Este es un texto {\em enfatizado}.`
Otro texto *enfatizado*.	`Otro texto \emph{enfatizado}.`

Al enfatizar, pasamos temporalmente a un tipo de letra distinto, la *itálica*. Esta letra es ligeramente inclinada hacia adelante, lo cual puede afectar el correcto espaciado entre palabras. Comparemos, por ejemplo:

Quiero *hoy* mi recompensa.	`Quiero {\em hoy} mi recompensa.`
Quiero *hoy* mi recompensa.	`Quiero {\em hoy\/} mi recompensa.`
Quiero *hoy* mi recompensa.	`Quiero \emph{hoy} mi recompensa.`

La segunda y tercera frase tienen un pequeño espacio adicional después de "hoy", para compensar el espacio entre palabras perdido por la inclinación de la itálica. Este pequeño espacio se denomina *corrección itálica*, y se consigue usando `\emph`, o, si se usa `\em`, agregando `\/` antes de cerrar el paréntesis cursivo. La corrección itálica es innecesaria cuando después del texto enfatizado viene un punto o una coma. LaTeX advierte esto y omite el espacio adicional aunque uno lo haya sugerido.

12.3.5. Notas a pie de página.

```
Insertemos una nota a pie de p\'agina.\footnote{Como \'esta.}
```

LaTeX colocará una nota a pie de página[1] en el lugar apropiado.

[1] Como ésta.

12.3.6. Fórmulas matemáticas.

LaTeX distingue dos modos de escritura: un modo de texto, en el cual se escriben los textos usuales como los ya mencionados, y un modo matemático, dentro del cual se escriben las fórmulas. Cualquier fórmula *debe* ser escrita dentro de un modo matemático, y si algún símbolo matemático aparece fuera del modo matemático el compilador acusará un error.

Hay tres formas principales para acceder al modo matemático:

a) `$x+y=3$`

b) `$$xy=8$$`

c)
```
\begin{equation}
x/y=5
\end{equation}
```

Estas tres opciones generan, respectivamente, una ecuación en el texto: $x + y = 3$, una ecuación separada del texto, centrada en la página:

$$xy = 8$$

y una ecuación separada del texto, numerada:

$$x/y = 5 \tag{12.1}$$

Es importante notar que al referirnos a una variable matemática en el texto debemos escribirla en modo matemático:

Decir que la incógnita es x es incorrecto. No: la incógnita es x.

```
Decir que la inc{\'o}gnita es
x es incorrecto. No: la
inc{\'o}gnita es $x$.
```

12.3.7. Comentarios.

Uno puede hacer que el compilador ignore parte del archivo usando `%`. Todo el texto desde este carácter hasta el fin de la línea correspondiente será ignorado (incluyendo el fin de línea).

Un pequeño comentario.

```
Un peque{\~n}o co% Texto ignorado
mentario.
```

12.3.8. Estilo del documento.

Las características generales del documento están definidas en el preámbulo. Lo más importante es la elección del *estilo*, que determina una serie de parámetros que al usuario normal pueden no importarle, pero que son básicas para una correcta presentación del texto: ¿Qué márgenes dejar en la página? ¿Cuánto dejar de sangría? ¿Tipo de letra? ¿Distancia entre líneas? ¿Dónde poner los números de página? Y un largo etcétera.

Todas estas decisiones se encuentran en un *archivo de estilo* (extensión `cls`). Los archivos standard son: `article`, `report`, `book` y `letter`, cada uno adecuado para escribir artículos cortos (sin capítulos) o más largos (con capítulos), libros y cartas, respectivamente.

La elección del estilo global se hace en la primera línea del archivo:[2]

```
\documentclass{article}
```

Esta línea será aceptada por el compilador, pero nos entregará un documento con un tamaño de letra pequeño, técnicamente llamado de 10 puntos ó 10pt (1pt = 1/72 pulgadas). Existen tres tamaños de letra disponibles: 10, 11 y 12 pt. Si queremos un tamaño de letra más grande, como el que tenemos en este documento, se lo debemos indicar en la primera línea del archivo:

```
\documentclass[12pt]{article}
```

Todas las decisiones de estilo contenidas dentro del archivo `cls` son modificables, existiendo tres modos de hacerlo:

a) Modificando el archivo `cls` directamente. Esto es poco recomendable, porque dicha modificación (por ejemplo, un cambio de los márgenes) se haría extensible a todos los archivos compilados en nuestro computador, y esto puede no ser agradable, ya sea que nosotros seamos los únicos usuarios o debamos compartirlo. Por supuesto, podemos deshacer los cambios cuando terminemos de trabajar, pero esto es tedioso.

b) Introduciendo comandos adecuados en el preámbulo. Ésta es la opción más recomendable y la más usada. Nos permite dominar decisiones específicas de estilo válidas sólo para el archivo que nos interesa.

c) Creando un nuevo archivo `cls`. Esto es muy recomendable cuando las modificaciones de estilo son abundantes, profundas y deseen ser reaprovechadas. Se requiere un poco de experiencia en LaTeX para hacerlo, pero a veces puede ser la única solución razonable.

En todo caso, la opción a usar en la gran mayoría de los casos es la b) (Sec. 12.9).

12.3.9. Argumentos de comandos.

Hemos visto ya algunos comandos que requieren argumentos. Por ejemplo: `\begin{equation}`, `\documentclass[12pt]{article}`, `\footnote{Nota}`. Existen dos tipos de argumentos:

1. **Argumentos obligatorios.** Van encerrados en paréntesis cursivos: `\footnote{Nota}`, por ejemplo. Es obligatorio que después de estos comandos aparezcan los paréntesis. A veces es posible dejar el interior de los paréntesis vacío, pero en otros casos el compilador reclamará incluso eso (`\footnote{}` no genera problemas, pero `\documentclass{}` sí es un gran problema).

   Una propiedad muy general de los comandos de LaTeX es que las llaves de los argumentos obligatorios se pueden omitir cuando dichos argumentos tienen sólo un carácter. Por ejemplo, `\~n` es equivalente a `\~{n}`. Esto permite escribir más fácilmente muchas expresiones, particularmente matemáticas, como veremos más adelante.

[2] En LaTeX 2.09 esta primera línea debe ser `\documentstyle[12pt]article`, y el archivo de estilo tiene extensión `sty`. Intentar compilar con LaTeX 2.09 un archivo que comienza con `\documentclass` da un error. Por el contrario, la compilación con LaTeX 2ε de un archivo que comienza con `\documentstyle` no genera un error, y LaTeX entra en un *modo de compatibilidad*. Sin embargo, interesantes novedades de LaTeX 2ε respecto a LaTeX 2.09 se pierden.

2. **Argumentos opcionales.** Van encerrados en paréntesis cuadrados. Estos argumentos son omitibles, `\documentclass[12pt]...` . Ya dijimos que `\documentclass{article}` es aceptable, y que genera un tamaño de letra de 10pt. Un argumento en paréntesis cuadrados es una opción que modifica la decisión default del compilador (en este caso, lo obliga a usar 12pt en vez de sus instintivos 10pt).

12.3.10. Título.

Un título se genera con:

```
\title{Una breve introducci\'on}
\author{V\'{\i}ctor Mu\~noz}
\date{30 de Junio de 1998}
\maketitle
```

`\title`, `\author` y `\date` pueden ir en cualquier parte (incluyendo el preámbulo) antes de `\maketitle`. `\maketitle` debe estar después de `\begin{document}`. Dependiendo de nuestras necesidades, tenemos las siguientes alternativas:

a) Sin título:

```
\title{}
```

b) Sin autor:

```
\author{}
```

c) Sin fecha:

```
\date{}
```

d) Fecha actual (en inglés): omitir `\date`.

e) Más de un autor:

```
\author{Autor_1 \and Autor_2 \and Autor_3}
```

Para artículos cortos, LaTeX coloca el título en la parte superior de la primera página del texto. Para artículos largos, en una página separada.

12.3.11. Secciones.

Los títulos de las distintas secciones y subsecciones de un documento (numerados adecuadamente, en negrita, como en este texto) se generan con comandos de la forma:

```
\section{Una secci\'on}
\subsection{Una subsecci\'on}
```

Los comandos disponibles son (en orden decreciente de importancia):

`\part`	`\subsection`	`\paragraph`
`\chapter`	`\subsubsection`	`\subparagraph`
`\section`		

Los más usados son `\chapter`, `\section`, `\subsection` y `\subsubsection`. `\chapter` sólo está disponible en los estilos `report` y `book`.

12.3.12. Listas.

Los dos modos usuales de generar listas:

a) Listas numeradas (ambiente `enumerate`):

1. Nivel 1, ítem 1.
2. Nivel 1, ítem 2.
   a) Nivel 2, ítem 1.
      1) Nivel 3, ítem 1.
3. Nivel 1, ítem 3.

```
\begin{enumerate}
\item Nivel 1, \'{\i}tem 1.
\item Nivel 1, \'{\i}tem 2.
\begin{enumerate}
\item Nivel 2, \'{\i}tem 1.
\begin{enumerate}
\item Nivel 3, \'{\i}tem 1.
\end{enumerate}
\end{enumerate}
\item Nivel 1, \'{\i}tem 3.
\end{enumerate}
```

b) Listas no numeradas (ambiente `itemize`):

- Nivel 1, ítem 1.
- Nivel 1, ítem 2.
  - Nivel 2, ítem 1.
    - Nivel 3, ítem 1.
- Nivel 1, ítem 3.

```
\begin{itemize}
\item Nivel 1, {\'\i}tem 1.
\item Nivel 1, {\'\i}tem 2.
\begin{itemize}
\item Nivel 2, {\'\i}tem 1.
\begin{itemize}
\item Nivel 3, {\'\i}tem 1.
\end{itemize}
\end{itemize}
\item Nivel 1, {\'\i}tem 3.
\end{itemize}
```

Es posible anidar hasta tres niveles de listas. Cada uno usa tipos distintos de rótulos, según el ambiente usado: números arábes, letras y números romanos para `enumerate`, y puntos, guiones y asteriscos para `itemize`. Los rótulos son generados automáticamente por cada `\item`, pero es posible modificarlos agregando un parámetro opcional:

a) Nivel 1, ítem 1.
b) Nivel 1, ítem 2.

```
\begin{enumerate}
\item[a)] Nivel 1, \'{\i}tem 1.
\item[b)] Nivel 1, \'{\i}tem 2.
\end{enumerate}
```

`\item` es lo primero que debe aparecer después de un `\begin{enumerate}` o `\begin{itemize}`.

12.3.13. Tipos de letras.

Fonts.

Los fonts disponibles por default en LaTeX son:

roman	*italic*	Small Caps
boldface	*slanted*	`typewriter`
sans serif		

Los siguientes modos de cambiar fonts son equivalentes:

texto	`{\rm texto}`	`\textrm{texto}`
texto	`{\bf texto}`	`\textbf{texto}`
texto	`{\sf texto}`	`\textsf{texto}`
texto	`{\it texto}`	`\textit{texto}`
texto	`{\sl texto}`	`\textsl{texto}`
Texto	`{\sc Texto}`	`\textsc{texto}`
`texto`	`{\tt texto}`	`\texttt{texto}`

`\rm` es el default para texto normal; `\it` es el default para texto enfatizado; `\bf` es el default para títulos de capítulos, secciones, subsecciones, etc.

`\textrm`, `\textbf`, etc., sólo permiten cambiar porciones definidas del texto, contenido entre los paréntesis cursivos. Con `\rm`, `\bf`, etc. podemos, omitiendo los paréntesis, cambiar el font en todo el texto posterior:

Un cambio local de fonts *y uno global, interminable e infinito...*

```
Un cambio {\sf local} de fonts
\sl y uno global, interminable
e infinito...
```

También es posible tener combinaciones de estos fonts, por ejemplo, ***bold italic***, pero no sirven los comandos anteriores, sino versiones modificadas de `\rm`, `\bf`, etc.:

```
\rmfamily
\sffamily
\ttfamily
\mdseries
```

```
\bfseries
\upshape
\itshape
\slshape
\scshape
```

Por ejemplo:

texto	`{\bfseries\itshape texto}`
texto	`{\bfseries\upshape texto}` (= `{\bf texto}`)
TEXTO	`{\ttfamily\scshape texto}`
texto	`{\sffamily\bfseries texto}`
texto	`{\sffamily\mdseries texto}` (= `{\sf texto}`)

Para entender el uso de estos comandos hay que considerar que un font tiene tres *atributos*: `family` (que distingue entre `rm`, `sf` y `tt`), `series` (que distingue entre `md` y `bf`), y `shape` (que distingue entre `up`, `it`, `sl` y `sc`). Cada uno de los comandos `\rmfamily`, `\bfseries`, etc., cambia sólo uno de estos atributos. Ello permite tener versiones mixtas de los fonts, como un *slanted sans serif*, imposible de obtener usando los comandos `\sl` y `\sf`. Los defaults para el texto usual son: `\rmfamily`, `\mdseries` y `\upshape`.

Tamaño.

Los tamaños de letras disponibles son:

texto	`\tiny`	texto	`\normalsize`	texto	`\LARGE`
texto	`\scriptsize`	texto	`\large`	texto	`\huge`
texto	`\footnotesize`	texto	`\Large`	texto	`\Huge`
texto	`\small`				

Se usan igual que los comandos de cambio de font `\rm`, `\sf`, etc., de la sección 12.3.13.

`\normalsize` es el default para texto normal; `\scriptsize` para sub o supraíndices; `\footnotesize` para notas a pie de página.

12.3.14. Acentos y símbolos.

LaTeX provee diversos tipos de acentos, que se muestran en la Tabla 12.1 (como ejemplo consideramos la letra "o", pero cualquiera es posible, por supuesto). (Hemos usado acá el hecho de que cuando el argumento de un comando consta de un carácter, las llaves son omitibles.)

Otros símbolos especiales y caracteres no ingleses disponibles se encuentran en la Tabla 12.2.

ó	\'o	õ	\~o	ǒ	\v o	o̧	\c o
ò	\`o	ō	\=o	ő	\H o	ọ	\d o
ô	\^o	ȯ	\. o	o͡o	\t{oo}	o̲	\b o
ö	\"o	ŏ	\u o	o̊	\r o		

Cuadro 12.1: Acentos.

†	\dag	œ	\oe	ł	\l
‡	\ddag	Œ	\OE	Ł	\L
§	\S	æ	\ae	ß	\ss
¶	\P	Æ	\AE	SS	\SS
©	\copyright	å	\aa	¿	?`
ⓐ	\textcircled a	Å	\AA	¡	!`
␣	\textvisiblespace	ø	\o		
£	\pounds	Ø	\O		

Cuadro 12.2: Símbolos especiales y caracteres no ingleses.

12.3.15. Escritura de textos en castellano.

LaTeX emplea sólo los caracteres ASCII básicos, que no contienen símbolos castellanos como ¿, ¡, ñ, etc. Ya hemos visto que existen comandos que permiten imprimir estos caracteres, y por tanto es posible escribir cualquier texto en castellano (y otros idiomas, de hecho).

Sin embargo, esto no resuelve todo el problema, porque en inglés y castellano las palabras se cortan en "sílabas" de acuerdo a reglas distintas, y esto es relevante cuando se debe cortar el texto en líneas. LaTeX tiene incorporados algoritmos para cortar palabras en inglés y, si se ha hecho una instalación especial de LaTeX en nuestro computador, también en castellano u otros idiomas (a través del programa **babel**, que es parte de la distribución standard de LaTeX 2_ε). En un computador con babel instalado y configurado para cortar en castellano basta incluir el comando `\usepackage[spanish]{babel}` en el preámbulo para poder escribir en castellano cortando las palabras en sílabas correctamente.[3]

Sin embargo, ocasionalmente LaTeX se encuentra con una palabra que no sabe cortar, en cuyo caso no lo intenta y permite que ella se salga del margen derecho del texto, o bien toma decisiones no óptimas. La solución es sugerirle a LaTeX la silabación de la palabra. Por ejemplo, si la palabra conflictiva es `matem\'aticas` (generalmente hay problemas con las palabras acentuadas), entonces basta con reescribirla en la forma: `ma\-te\-m\'a\-ti\-cas`. Con esto, le indicamos a LaTeX en qué puntos es posible cortar la palabra. El comando `\-` no tiene ningún otro efecto, de modo que si la palabra en cuestión no queda al final de la línea, LaTeX por supuesto ignora nuestra sugerencia y no la corta.

Consideremos el siguiente ejemplo:

[3]Esto resuelve también otro problema: los encabezados de capítulos o índices, por ejemplo, son escritos "Capítulo" e "Índice", en vez de "Chapter" e "Index", y cuando se usa el comando `\date`, la fecha aparece en castellano.

Podemos escribir matemáticas. O matemáticas.

```
Podemos escribir matem\'aticas.
O matem\'aticas.
```

Podemos escribir matemáticas. O matemáticas.

```
Podemos escribir
ma\-te\-m\'a\-ti\-cas.
O ma\-te\-m\'a\-ti\-cas.
```

En el primer caso, LaTeX decidió por sí mismo dónde cortar "matemáticas". Como es una palabra acentuada tuvo problemas y no lo hizo muy bien, pues quedó demasiado espacio entre palabras en esa línea. En el segundo párrafo le sugerimos la silabación y LaTeX pudo tomar una decisión más satisfactoria. En el mismo párrafo, la segunda palabra "matemáticas" también tiene sugerencias de corte, pero como no quedó al final de línea no fueron tomadas en cuenta.

12.4. Fórmulas matemáticas.

Hemos mencionado tres formas de ingresar al modo matemático: `$...$` (fórmulas dentro del texto), `$$...$$` (fórmulas separadas del texto, no numeradas) y `\begin{equation} ... \end{equation}` (fórmulas separadas del texto, numeradas). Los comandos que revisaremos en esta sección sólo pueden aparecer dentro del modo matemático.

12.4.1. Sub y supraíndices.

x^{2y} `x^{2y}` x^{y^2} `x^{y^{2}}` (ó `x^{y^2}`) x_1^y `x^y_1` (ó `x_1^y`)
x_{2y} `x_{2y}` x^{y_1} `x^{y_{1}}` (ó `x^{y_1}`)

`\textsuperscript` permite obtener supraíndices fuera del modo matemático:

La 3ª es la vencida.

```
La 3\textsuperscript{a}
es la vencida.
```

12.4.2. Fracciones.

a) Horizontales

$n/2$ `n/2`

b) Verticales

$\frac{1}{2}$ `\frac{1}{2}`, `\frac 1{2}`, `\frac{1}2` ó `\frac 12`

$x = \frac{y + z/2}{y^2 + 1}$

```
x = \frac{y + z/2}{y^2+1}
```

$\frac{x+y}{1 + \frac{y}{z+1}}$

```
\frac{x+y}{1 + \frac y{z+1}}
```

La forma a) es más adecuada y la preferida para fracciones dentro del texto, y la segunda para fórmulas separadas. `\frac` puede aparecer en fórmulas dentro del texto ($\frac{1}{2}$ con `$\frac 12$`), pero esto es inusual y poco recomendable estéticamente, salvo estricta necesidad.

12.4.3. Raíces.

$\sqrt{n}$	`\sqrt{n}` ó `\sqrt n`
$\sqrt{a^2 + b^2}$	`\sqrt{a^2 + b^2}`
$\sqrt[n]{2}$	`\sqrt[n]{2}`

12.4.4. Puntos suspensivos.

a) $\ldots$ `\ldots`

Para fórmulas como

$$a_1 a_2 \ldots a_n \qquad \texttt{a\_1 a\_2 \textbackslash ldots a\_n}$$

b) $\cdots$ `\cdots`

Entre símbolos como $+$, $-$, $=$:

$$x_1 + \cdots + x_n \qquad \texttt{x\_1 + \textbackslash cdots + x\_n}$$

c) $\vdots$ `\vdots`

$$\begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}$$

d) $\ddots$ `\ddots`

$$I_{n\times n} = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \ldots & 1 \end{pmatrix}$$

`\ldots` puede ser usado también en el texto usual:

Arturo quiso salir...pero se detuvo.	`Arturo quiso salir\ldots` `pero se detuvo.`

No corresponde usar tres puntos seguidos (...), pues el espaciado entre puntos es incorrecto.

Minúsculas

α	\alpha	θ	\theta	o	o	τ	\tau
β	\beta	ϑ	\vartheta	π	\pi	υ	\upsilon
γ	\gamma	ι	\iota	ϖ	\varpi	ϕ	\phi
δ	\delta	κ	\kappa	ρ	\rho	φ	\varphi
ϵ	\epsilon	λ	\lambda	ϱ	\varrho	χ	\chi
ε	\varepsilon	μ	\mu	σ	\sigma	ψ	\psi
ζ	\zeta	ν	\nu	ς	\varsigma	ω	\omega
η	\eta	ξ	\xi				

Mayúsculas

Γ	\Gamma	Λ	\Lambda	Σ	\Sigma	Ψ	\Psi
Δ	\Delta	Ξ	\Xi	Υ	\Upsilon	Ω	\Omega
Θ	\Theta	Π	\Pi	Φ	\Phi		

Cuadro 12.3: Letras griegas.

12.4.5. Letras griegas.

Las letras griegas se obtienen simplemente escribiendo el nombre de dicha letra (en inglés): `\gamma`. Para la mayúscula correspondiente se escribe la primera letra con mayúscula: `\Gamma`. La lista completa se encuentra en la Tabla 12.3.

No existen símbolos para α, β, η, etc. mayúsculas, pues corresponden a letras romanas (A, B, E, etc.).

12.4.6. Letras caligráficas.

Letras caligráficas mayúsculas $\mathcal{A}$, $\mathcal{B}$, ..., $\mathcal{Z}$ se obtienen con `\cal`. `\cal` se usa igual que los otros comandos de cambio de font (`\rm`, `\it`, etc.).

Sea $\mathcal{F}$ una función con $\mathcal{F}(x) > 0$.	`Sea $\cal F$ una funci\'on con ${\cal F}(x) > 0$.`

No son necesarios los paréntesis cursivos la primera vez que se usan en este ejemplo, porque el efecto de `\cal` está delimitado por los `$`.

12.4.7. Símbolos matemáticos.

LaTeX proporciona una gran variedad de símbolos matemáticos (Tablas 12.4, 12.5, 12.6, 12.7).

La negación de cualquier símbolo matemático se obtiene con `\not`:

$x \not< y$	`x \not < y`
$a \notin \mathcal{M}$	`a \not \in {\cal M}`

[Notemos, sí, en la Tabla 12.5, que existe el símbolo $\neq$ (`\neq`).]

$\pm$	\pm	$\cap$	\cap	$\diamond$	\diamond	$\oplus$	\oplus
$\mp$	\mp	$\cup$	\cup	$\triangle$	\bigtriangleup	$\ominus$	\ominus
$\times$	\times	$\uplus$	\uplus	$\triangledown$	\bigtriangledown	$\otimes$	\otimes
$\div$	\div	$\sqcap$	\sqcap	$\triangleleft$	\triangleleft	$\oslash$	\oslash
$*$	\ast	$\sqcup$	\sqcup	$\triangleright$	\triangleright	$\odot$	\odot
$\star$	\star	$\vee$	\lor	$\bigcirc$	\bigcirc		
$\circ$	\circ	$\wedge$	\land	$\dagger$	\dagger		
$\bullet$	\bullet	$\setminus$	\setminus	$\ddagger$	\ddagger		
$\cdot$	\cdot	$\wr$	\wr	$\amalg$	\amalg		

Cuadro 12.4: Símbolos de operaciones binarias.

$\leq$	\leq	$\geq$	\geq	$\equiv$	\equiv	$\models$	\models
$\prec$	\prec	$\succ$	\succ	$\sim$	\sim	$\perp$	\perp
$\preceq$	\preceq	$\succeq$	\succeq	$\simeq$	\simeq	$\mid$	\mid
$\ll$	\ll	$\gg$	\gg	$\asymp$	\asymp	$\parallel$	\parallel
$\subset$	\subset	$\supset$	\supset	$\approx$	\approx	$\bowtie$	\bowtie
$\subseteq$	\subseteq	$\supseteq$	\supseteq	$\cong$	\cong	$\neq$	\neq
$\smile$	\smile	$\sqsubseteq$	\sqsubseteq	$\sqsupseteq$	\sqsupseteq	$\doteq$	\doteq
$\frown$	\frown	$\in$	\in	$\ni$	\ni	$\propto$	\propto
$\vdash$	\vdash	$\dashv$	\dashv				

Cuadro 12.5: Símbolos relacionales.

$\leftarrow$	\gets	$\longleftarrow$	\longleftarrow	$\uparrow$	\uparrow
$\Leftarrow$	\Leftarrow	$\Longleftarrow$	\Longleftarrow	$\Uparrow$	\Uparrow
$\rightarrow$	\to	$\longrightarrow$	\longrightarrow	$\downarrow$	\downarrow
$\Rightarrow$	\Rightarrow	$\Longrightarrow$	\Longrightarrow	$\Downarrow$	\Downarrow
$\leftrightarrow$	\Leftrightarrow	$\longleftrightarrow$	\Longleftrightarrow	$\updownarrow$	\Updownarrow
$\mapsto$	\mapsto	$\longmapsto$	\longmapsto	$\nearrow$	\nearrow
$\hookleftarrow$	\hookleftarrow	$\hookrightarrow$	\hookrightarrow	$\searrow$	\searrow
$\leftharpoonup$	\leftharpoonup	$\rightharpoonup$	\rightharpoonup	$\swarrow$	\swarrow
$\leftharpoondown$	\leftharpoondown	$\rightharpoondown$	\rightharpoondown	$\nwarrow$	\nwarrow
$\rightleftharpoons$	\rightleftharpoons				

Cuadro 12.6: Flechas

$\aleph$	\aleph	$\prime$	\prime	$\forall$	\forall	∞	\infty
$\hbar$	\hbar	$\emptyset$	\emptyset	$\exists$	\exists	$\triangle$	\triangle
$\imath$	\imath	∇	\nabla	$\neg$	\lnot	$\clubsuit$	\clubsuit
$\jmath$	\jmath	$\surd$	\surd	$\flat$	\flat	$\diamondsuit$	\diamondsuit
ℓ	\ell	$\top$	\top	$\natural$	\natural	$\heartsuit$	\heartsuit
$\wp$	\wp	$\bot$	\bot	$\sharp$	\sharp	$\spadesuit$	\spadesuit
$\Re$	\Re	$\|$	|	$\backslash$	\backslash		
$\Im$	\Im	$\angle$	\angle	∂	\partial		

Cuadro 12.7: Símbolos varios.

$\sum$	$\sum$	\sum	$\bigcap$	$\bigcap$	\bigcap	$\bigodot$	$\bigodot$	\bigodot
$\prod$	$\prod$	\prod	$\bigcup$	$\bigcup$	\bigcup	$\bigotimes$	$\bigotimes$	\bigotimes
$\coprod$	$\coprod$	\coprod	$\bigsqcup$	$\bigsqcup$	\bigsqcup	$\bigoplus$	$\bigoplus$	\bigoplus
$\int$	$\int$	\int	$\bigvee$	$\bigvee$	\bigvee	$\bigoplus$	$\bigoplus$	\bigoplus
$\oint$	$\oint$	\oint	$\bigwedge$	$\bigwedge$	\bigwedge			

Cuadro 12.8: Símbolos de tamaño variable.

Algunos símbolos tienen tamaño variable, según aparezcan en el texto o en fórmulas separadas del texto. Se muestran en la Tabla 12.8.

Estos símbolos pueden tener índices que se escriben como sub o supraíndices. Nuevamente, la ubicación de estos índices depende de si la fórmula está dentro del texto o separada de él:

$$\sum_{i=1}^{n} x_i = \int_0^1 f$$ `$$\sum_{i=1}^n x_i = \int_0^1 f $$`

$\sum_{i=1}^{n} x_i = \int_0^1 f$ `$\sum_{i=1}^n x_i = \int_0^1 f $`

12.4.8. Funciones tipo logaritmo.

Observemos la diferencia entre estas dos expresiones:

$x = logy$ `$x  = log y$`

$x = \log y$ `$x  = \log y$`

En el primer caso LaTeX escribe el producto de cuatro cantidades, l, o, g e y. En el segundo, representa correctamente nuestro deseo: el logaritmo de y. Todos los comandos de la Tabla 12.9 generan el nombre de la función correspondiente, en letras romanas.

Algunas de estas funciones pueden tener índices:

$$\lim_{n\to\infty} x_n = 0$$ `$$\lim_{n\to\infty} x_n = 0 $$`

$\lim_{n\to\infty} x_n = 0$ `$\lim_{n\to\infty} x_n = 0 $`

\arccos	\cos	\csc	\exp	\ker	\limsup	\min	\sinh
\arcsin	\cosh	\deg	\gcd	\lg	\ln	\Pr	\sup
\arctan	\cot	\det	\hom	\lim	\log	\sec	\tan
\arg	\coth	\dim	\inf	\liminf	\max	\sin	\tanh

Cuadro 12.9: Funciones tipo logaritmo

(	(	)	)	↑	\uparrow
[	[	]	]	↓	\downarrow
{	\{	}	\}	↕	\updownarrow
⌊	\lfloor	⌋	\rfloor	⇑	\Uparrow
⌈	\lceil	⌉	\rceil	⇓	\Downarrow
⟨	\langle	⟩	\rangle	⇕	\Updownarrow
/	/	\	\backslash		
|	|	‖	\\|		

Cuadro 12.10: Delimitadores

12.4.9. Matrices.

Ambiente `array`.

Se construyen con el ambiente `array`. Consideremos, por ejemplo:

$$\begin{array}{clr} a+b+c & uv & 27 \\ a+b & u+v & 134 \\ a & 3u+vw & 2{,}978 \end{array}$$

La primera columna está alineada al centro (`c`, center); la segunda, a la izquierda (`l`, left); la tercera, a la derecha (`r`, right). `array` tiene un argumento obligatorio, que consta de tantas letras como columnas tenga la matriz, letras que pueden ser `c`, `l` o `r` según la alineación que queramos obtener. Elementos consecutivos de la misma línea se separan con `&` y líneas consecutivas se separan con `\\`. Así, el ejemplo anterior se obtiene con:

```
\begin{array}{clr}
a+b+c & uv & 27 \\
a+b & u + v & 134 \\
a & 3u+vw & 2.978
\end{array}
```

Delimitadores.

Un delimitador es cualquier símbolo que actúe como un paréntesis, encerrando una expresión, apareciendo a la izquierda y a la derecha de ella. La Tabla 12.10 muestra todos los delimitadores posibles.

Para que los delimitadores tengan el tamaño correcto para encerrar la expresión correspondiente hay que anteponerles `\left` y `\right`. Podemos obtener así expresiones matriciales:

$$\left(\begin{array}{cc} a & b \\ c & d \end{array} \right)$$

```
\left(\begin{array}{cc}
a&b\\
c&d
\end{array}\right)
```

$$v = \left(\begin{array}{c} 1 \\ 2 \\ 3 \end{array} \right)$$

```
v = \left(\begin{array}{c}
1\\
2\\
3
\end{array}\right)
```

$$\Delta = \left| \begin{array}{cc} a_{11} & a_{12} \\ a_{21} & a_{22} \end{array} \right|$$

```
\Delta = \left|\begin{array}{cc}
a_{11} & a_{12}\\
a_{21} & a_{22}
\end{array}\right|
```

`\left` y `\right` deben ir de a pares, pero los delimitadores no tienen por qué ser los mismos:

$$\left(\begin{array}{c} a \\ b \end{array} \right[$$

```
\left(\begin{array}{c}
a\\
b
\end{array}\right[
```

Tampoco es necesario que los delimitadores encierren matrices. Comparemos, por ejemplo:

$$(\vec{A} + \vec{B}) = (\frac{d\vec{F}}{dx})_{x=a}$$

```
(\vec A + \vec B) =
( \frac{d \vec F}{dx} )_{x=a}
```

$$\left(\vec{A} + \vec{B}\right) = \left(\frac{d\vec{F}}{dx}\right)_{x=a}$$

```
\left(\vec A + \vec B\right) =
\left( \frac{d \vec F}{dx} \right)_{x=a}
```

El segundo ejemplo es mucho más adecuado estéticamente.

Algunas expresiones requieren sólo un delimitador, a la izquierda o a la derecha. Un punto (`.`) representa un delimitador invisible. Los siguientes ejemplos son típicos:

$$\int_a^b dx \frac{df}{dx} = f(x) \Big|_a^b$$

```
\left. \int_a^b  dx \frac{df}{dx} =
f(x) \right |_a^b
```

$$f(x) = \left\{ \begin{array}{cl} 0 & x < 0 \\ 1 & x > 0 \end{array} \right.$$

```
f(x) = \left\{ \begin{array}{cl}
0 & x<0 \\
1 & x>0
\end{array} \right.
```

Fórmulas de más de una línea.

`eqnarray` ofrece una manera de ingresar a modo matemático (en reemplazo de `$`, `$$` o `equation`) equivalente a un `array` con argumentos `{rcl}`:

$\hat{a}$	`\hat a`	$\acute{a}$	`\acute a`	$\bar{a}$	`\bar a`	$\dot{a}$	`\dot a`
$\check{a}$	`\check a`	$\grave{a}$	`\grave a`	$\vec{a}$	`\vec a`	$\ddot{a}$	`\ddot a`
$\breve{a}$	`\breve a`	$\tilde{a}$	`\tilde a`				

Cuadro 12.11: Acentos matemáticos

$$\begin{aligned} x &= a+b+c+ \\ &\quad d+e \end{aligned}$$

```
\begin{eqnarray*}
x& = & a + b + c +\\
&& d + e
\end{eqnarray*}
```

El asterisco impide que aparezcan números en las ecuaciones. Si deseamos que numere cada línea como una ecuación independiente, basta omitir el asterisco:

$$\begin{aligned} x &= 5 \quad (12.2) \\ a+b &= 60 \quad (12.3) \end{aligned}$$

```
\begin{eqnarray}
x& = & 5 \\
a + b&= & 60
\end{eqnarray}
```

Si queremos que solamente algunas líneas aparezcan numeradas, usamos `\nonumber`:

$$\begin{aligned} x &= a+b+c+ \\ &\quad d+e \quad (12.4) \end{aligned}$$

```
\begin{eqnarray}
x& = & a + b + c + \nonumber\\
&& d + e
\end{eqnarray}
```

El comando `\eqnarray` es suficiente para necesidades sencillas, pero cuando se requiere escribir matemática de modo intensivo sus limitaciones comienzan a ser evidentes. Al agregar al preámbulo de nuestro documento la línea `\usepackage{amsmath}` quedan disponibles muchos comandos mucho más útiles para textos matemáticos más serios, como el ambiente `equation*`, `\split`, `\multline` o `\intertext`. En la sección 12.8.2 se encuentra una descripción de estos y otros comandos.

12.4.10. Acentos.

Dentro de una fórmula pueden aparecer una serie de "acentos", análogos a los de texto usual (Tabla 12.11).

Las letras i y j deben perder el punto cuando son acentuadas: $\vec{i}$ es incorrecto. Debe ser $\vec{\imath}$. `\imath` y `\jmath` generan las versiones sin punto de estas letras:

$\vec{\imath} + \hat{\jmath}$

```
\vec \imath + \hat \jmath
```

12.4.11. Texto en modo matemático.

Para insertar texto dentro de modo matemático empleamos `\mbox`:

$V_{\text{crítico}}$ `V_{\mbox{\scriptsize cr\'{\i}tico}}`

Bastante más óptimo es utilizar el comando `\text`, disponible a través de `amsmath` (sección 12.8.2).

12.4.12. Espaciado en modo matemático.

TEX ignora los espacios que uno escribe en las fórmulas y los determina de acuerdo a sus propios criterios. A veces es necesario ayudarlo para hacer ajustes finos. Hay cuatro comandos que agregan pequeños espacios dentro de modo matemático:

`\,`	espacio pequeño	`\:`	espacio medio
`\!`	espacio pequeño (negativo)	`\;`	espacio grueso

Algunos ejemplos de su uso:

$\sqrt{2}\,x$	`\sqrt 2 \, x`	en vez de	$\sqrt{2}x$
$n/\log n$	`n / \!\log n`	en vez de	$n/\log n$
$\int f\,dx$	`\int f \, dx`	en vez de	$\int f dx$

El último caso es quizás el más frecuente, por cuanto la no inserción del pequeño espacio adicional entre f y dx hace aparecer el integrando como el producto de tres variables, f, d y x, que no es la idea.

12.4.13. Fonts.

Análogamente a los comandos para texto usual (Sec. 12.3.13), es posible cambiar los fonts dentro del modo matemático:

(A, x)	`\mathrm{(A,x)}`
(A, x)	`\mathnormal{(A,x)}`
$(\mathcal{A}, \mathcal{B})$	`\mathcal{(A,B)}`
$(\mathbf{A}, \mathbf{x})$	`\mathbf{(A,x)}`
(A, x)	`\mathsf{(A,x)}`
$(\mathtt{A}, \mathtt{x})$	`\mathtt{(A,x)}`
(A, x)	`\mathit{(A,x)}`

(Recordemos que la letras tipo `\cal` sólo existen en mayúsculas.)

Las declaraciones anteriores permiten cambiar los fonts de letras, dígitos y acentos, pero no de los otros símbolos matemáticos:

$\mathbf{\tilde{A} \times 1}$ `\mathbf{\tilde A \times 1}`

Como en todo ambiente matemático, los espacios entre caracteres son ignorados:

Hola `\mathrm{H o l a}`

Finalmente, observemos que `\mathit` corresponde al font itálico, en tanto que `\mathnormal` al font matemático usual, que es también itálico... o casi:

$different$	`$different$`
$different$	`$\mathnormal{different}$`

different	`$\mathit{different}$`
different	`\textit{different}`

12.5. Tablas.

`array` nos permitió construir matrices en modo matemático. Para tablas de texto existe `tabular`, que funciona de la misma manera. Puede ser usado tanto en modo matemático como fuera de él.

Nombre	:	Juan Pérez
Edad	:	26
Profesión	:	Estudiante

```
\begin{tabular}{lcl}
Nombre&:&Juan P\'erez\\
Edad&:&26\\
Profesi\'on&:&Estudiante
\end{tabular}
```

Si deseamos agregar líneas verticales y horizontales para ayudar a la lectura, lo hacemos insertando `|` en los puntos apropiados del argumento de `tabular`, y `\hline` al final de cada línea de la tabla:

Item	Gastos
Vasos	$ 500
Botellas	$ 1300
Platos	$ 500
Total	$ 2300

```
\begin{tabular}{|l|r|}\hline
Item&Gastos\\ \hline
Vasos& \$ 500 \\
Botellas & \$ 1300 \\
Platos & \$ 500 \\ \hline
Total& \$ 2300 \\ \hline
\end{tabular}
```

12.6. Referencias cruzadas.

Ecuaciones, secciones, capítulos y páginas son entidades que van numeradas y a las cuales podemos querer referirnos en el texto. Evidentemente no es óptimo escribir explícitamente el número correspondiente, pues la inserción de una nueva ecuación, capítulo, etc., su eliminación o cambio de orden del texto podría alterar la numeración, obligándonos a modificar estos números dispersos en el texto. Mucho mejor es referirse a ellos de modo simbólico y dejar que TEX inserte por nosotros los números. Lo hacemos con `\label` y `\ref`.

La ecuación de Euler

$$e^{i\pi} + 1 = 0 \qquad (12.5)$$

reúne los números más importantes. La ecuación (12.5) es famosa.

```
La ecuaci\'on de Euler
\begin{equation}
\label{euler}
 e^{i\pi} + 1 = 0
\end{equation}
re\'une los n\'umeros
m\'as importantes.
La ecuaci\'on (\ref{euler})
es famosa.
```

El argumento de `\label` (reiterado luego en `\ref`) es una etiqueta simbólica. Ella puede ser cualquier secuencia de letras, dígitos o signos de puntuación. Letras mayúsculas y minúsculas son diferentes. Así, `euler`, `eq:euler`, `euler_1`, `euler1`, `Euler`, etc., son etiquetas válidas y distintas. Podemos usar `\label` dentro de `equation`, `eqnarray` y `enumerate`.

También podemos referenciar páginas con `\pageref`:

Ver página 390 para más detalles.

[Texto en pág. 390]

El significado de la vida...

```
Ver p\'agina
\pageref{significado}
para m\'as detalles.
...
El significado
\label{significado}
de la vida...
```

LaTeX puede dar cuenta de las referencias cruzadas gracias al archivo `aux` (auxiliar) generado durante la compilación.

Al compilar por primera vez el archivo, en el archivo `aux` es escrita la información de los `\label` encontrados. Al compilar por segunda vez, LaTeX lee el archivo `aux` e incorpora esa información al `dvi`. (En realidad, también lo hizo la primera vez que se compiló el archivo, pero el `aux` no existía entonces o no tenía información útil.)

Por tanto, para obtener las referencias correctas hay que compilar dos veces, una para generar el `aux` correcto, otra para poner la información en el `dvi`. Toda modificación en la numeración tendrá efecto sólo después de compilar dos veces más. Por cierto, no es necesario preocuparse de estos detalles a cada momento. Seguramente compilaremos muchas veces el archivo antes de tener la versión final. En todo caso, LaTeX avisa, tras cada compilación, si hay referencias inexistentes u otras que pudieron haber cambiado, y sugiere compilar de nuevo para obtener las referencias correctas. (Ver Sec. 12.14.2.)

12.7. Texto centrado o alineado a un costado.

Los ambientes `center`, `flushleft` y `flushright` permiten forzar la ubicación del texto respecto a los márgenes. Líneas consecutivas se separan con `\\`:

Una línea centrada,
otra
y otra más.

Ahora el texto continúa

alineado a la izquierda

y finalmente

dos líneas
alineadas a la derecha.

```
\begin{center}
Una l\'{\i}nea centrada,\\
otra\\
y otra m\'as.
\end{center}
Ahora el texto contin\'ua
\begin{flushleft}
alineado a la izquierda
\end{flushleft}
y finalmente
\begin{flushright}
dos l\'{\i}neas\\
alineadas a la derecha.
\end{flushright}
```

12.8. Algunas herramientas importantes

Hasta ahora hemos mencionado escencialmente comandos disponibles en LaTeX standard. Sin embargo, éstos, junto con el resto de los comandos básicos de LaTeX, se vuelven insuficientes cuando

se trata de ciertas aplicaciones demasiado específicas, pero no inimaginables: si queremos escribir un texto de alta matemática, o usar LaTeX para escribir partituras, o para escribir un archivo `.tex` en un teclado croata.... Es posible que con los comandos usuales LaTeX responda a las necesidades, pero seguramente ello será a un costo grande de esfuerzo por parte del autor del texto. Por esta razón, las distribuciones modernas de LaTeX incorporan una serie de extensiones que hacen la vida un poco más fácil a los eventuales autores. En esta sección mencionaremos algunas extensiones muy útiles. Muchas otras no están cubiertas, y se sugiere al lector consultar la documentación de su distribución para saber qué otros paquetes se encuentran disponibles.

En general, las extensiones a LaTeX vienen contenidas en *paquetes* (*"packages"*, en inglés), en archivos `.sty`. Así, cuando mencionemos el paquete `amsmath`, nos referimos a características disponibles en el archivo `amsmath.sty`. Para que los comandos de un paquete `<package>.sty` estén disponibles, deben ser cargados durante la compilación, incluyendo en el preámbulo del documento la línea:

```
\usepackage{<package>}
```

Si se requiere cargar más de un paquete adicional, se puede hacer de dos formas:

```
\usepackage{<package1>,<package2>}
```

o

```
\usepackage{<package1>}
\usepackage{<package2>}
```

Algunos paquetes aceptan opciones adicionales (del mismo modo que la clase `article` acepta la opción `12pt`):

```
\usepackage[option1,option2]{<package1>}
```

Revisemos ahora algunos paquetes útiles.

12.8.1. `babel`

Permite el procesamiento de textos en idiomas distintos del inglés. Esto significa, entre otras cosas, que se incorporan los patrones de silabación correctos para dicho idioma, para cortar adecuadamente las palabras al final de cada línea. Además, palabras claves como "Chapter", "Index", "List of Figures", etc., y la fecha dada por `\date`, son cambiadas a sus equivalentes en el idioma escogido. La variedad de idiomas disponibles es enorme, pero cada instalación de LaTeX tiene sólo algunos de ellos incorporados. (Ésta es una decisión que toma el administrador del sistema, de acuerdo a las necesidades de los usuarios. Una configuración usual puede ser habilitar la compilación en inglés, castellano, alemán y francés.)

Ya sabemos como usar `babel` para escribir en castellano: basta incluir en el preámbulo la línea

```
\usepackage[spanish]{babel}
```

12.8.2. $\mathcal{AMS}$-LaTeX

El paquete `amsmath` permite agregar comandos para escritura de textos matemáticos profesionales, desarrollados originalmente por la American Mathematical Society. Si un texto contiene abundante matemática, entonces seguramente incluir la línea correspondiente en el preámbulo:

```
\usepackage{amsmath}
```

aliviará mucho la tarea. He aquí algunas de las características adicionales disponibles con $\mathcal{AMS}$-LaTeX.

Ambientes para ecuaciones

Con `equation*` generamos una ecuación separada del texto, no numerada:

$$x = 2y - 3$$

```
\begin{equation*}
x = 2y - 3
\end{equation*}
```

`multline` permite dividir una ecuación muy larga en varias líneas, de modo que la primera línea quede alineada con el margen izquierdo, y la última con el margen derecho:

$$\sum_{i=1}^{15} = 1 + 2 + 3 + 4 + 5+ \\ 6 + 7 + 8 + 9 + 10+ \\ 11 + 12 + 13 + 14 + 15 \quad (12.6)$$

```
\begin{multline}
\sum_{i=1}^{15} = 1 +2+3+4+5+\\
6+7+8+9+10+\\
11+12+13+14+15
\end{multline}
```

`align` permite reunir un grupo de ecuaciones consecutivas alineándolas (usando `&`, igual que la alineación vertical de `tabular` y `array`). `gather` hace lo mismo, pero centrando cada ecuación en la página independientemente.

$$a_1 = b_1 + c_1 \quad (12.7)$$
$$a_2 = b_2 + c_2 - d_2 + e_2 \quad (12.8)$$

```
\begin{align}
a_1 &= b_1 + c_1 \\
a_2 &= b_2 + c_2 - d_2 + e_2
\end{align}
```

$$a_1 = b_1 + c_1 \quad (12.9)$$
$$a_2 = b_2 + c_2 - d_2 + e_2 \quad (12.10)$$

```
\begin{gather}
a_1 = b_1 + c_1 \\
a_2 = b_2 + c_2 - d_2 + e_2
\end{gather}
```

Con `multline*`, `align*` y `gather*` se obtienen los mismos resultados, pero con ecuaciones no numeradas.

`split` permite escribir una sola ecuación separada en líneas (como `multline`), pero permite alinear las líneas con `&` (como `align`). `split` debe ser usado dentro de un ambiente como `equation`, `align` o `gather` (o sus equivalentes con asterisco):

$$\begin{aligned} a_1 &= b_1 + c_1 \\ &= b_2 + c_2 - d_2 + e_2 \end{aligned} \tag{12.11}$$

```
\begin{equation}
\begin{split}
a_1& = b_1 + c_1 \\
   & =  b_2 + c_2 - d_2 + e_2
\end{split}
\end{equation}
```

Espacio horizontal

`\quad` y `\qquad` insertan espacio horizontal en ecuaciones:

$$x > y \ , \quad \forall\, x \in A$$
$$x \leq z \ , \qquad \forall\, z \in B$$

```
\begin{gather*}
x > y \ , \quad \forall\, x \in A \\
x \leq z \ , \qquad \forall\, z \in B
\end{gather*}
```

Texto en ecuaciones

Para agregar texto a una ecuación, usamos `\text`:

$$x = 2^n - 1 \ , \quad \text{con } n \text{ entero}$$

```
\begin{equation*}
x = 2^n - 1 \ , \quad \text{con $n$ entero}
\end{equation*}
```

`\text` se comporta como un buen objeto matemático, y por tanto se pueden agregar subíndices textuales más fácilmente que con `\mbox` (ver sección 12.4.11):

$V_{\text{crítico}}$

```
$V_{\text{cr\'{\i}tico}}$
```

Referencia a ecuaciones

`\eqref` es equivalente a `\ref`, salvo que agrega los paréntesis automáticamente:

La ecuación (12.5) era la de Euler.

```
La ecuaci\'on \eqref{euler} era la de Euler.
```

Ecuaciones con casos

Ésta es una construcción usual en matemáticas:

$$f(x) = \begin{cases} 1 & \text{si } x < 0 \\ 0 & \text{si } x > 0 \end{cases}$$

```
f(x)=
\begin{cases}
1&\text{si $x<0$} \\
0&\text{si $x>0$}
\end{cases}
```

Notar cómo es más simple que el ejemplo con los comandos convencionales en la sección 12.4.9.

Texto insertado entre ecuaciones alineadas

Otra situación usual es insertar texto entre ecuaciones alineadas, preservando la alineación:

$$x_1 = a + b + c \ ,$$
$$x_2 = d + e \ ,$$

y por otra parte

$$x_3 = f + g + h \ .$$

```
\begin{align*}
x_1 &=  a + b + c  \ , \\
x_2 &=  d + e \ , \\
\intertext{y por otra parte}
x_3 &=  f + g + h \ .
\end{align*}
```

Matrices y coeficientes binomiales

La complicada construcción de matrices usando `array` (sección 12.4.9), se puede reemplazar con ambientes como `pmatrix` y `vmatrix`, y comandos como `\binom`.

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

```
\begin{pmatrix}
a&b\\
c&d
\end{pmatrix}
```

$$\Delta = \begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix}$$

```
\Delta = \begin{vmatrix}
a_{11} & a_{12}\\
a_{21} & a_{22}
\end{vmatrix}
```

$$v = \binom{k}{2}$$

```
v = \binom{k}{2}
```

Podemos observar que el espaciado entre los paréntesis y el resto de la fórmula es más adecuado que el de los ejemplos en la sección 12.4.9.

Flechas extensibles

Las flechas en la tabla 12.6 vienen en ciertos tamaños predefinidos. `amsmath` proporciona flechas extensibles `\xleftarrow` y `\xrightarrow`, para ajustar sub o superíndices demasiado anchos. Además, tienen un argumento opcional y uno obligatorio, para colocar material sobre o bajo ellas:

$$A \xleftarrow{n+\mu-1} B \xrightarrow[T]{n\pm i-1} C \underset{U}{\longrightarrow} D$$

```
A \xleftarrow{n+\mu-1} B \xrightarrow[T]{n\pm i-1} C \xrightarrow[U]{} D
```

12.8.3. fontenc

Ocasionalmente, LaTeX tiene problemas al separar una palabra en sílabas. Típicamente, eso ocurre con palabras acentuadas, pues, debido a la estructura interna del programa, un carácter como la "á" en "matemáticas" no es tratado igual que los otros. Para solucionar el problema, y poder cortar en sílabas palabras que contengan letras acentuadas (además de acceder a algunos caracteres adicionales), basta incluir el paquete `fontenc`:

```
\usepackage[T1]{fontenc}
```

Técnicamente, lo que ocurre es que la codificación antigua para fonts es la `OT1`, que no contiene fonts acentuados, y que por lo tanto es útil sólo para textos en inglés. La codificación `T1` aumenta los fonts disponibles, permitiendo que los caracteres acentuados sean tratados en igual pie que cualquier otro.

12.8.4. enumerate

`enumerate.sty` define una muy conveniente extensión al ambiente `enumerate` de LaTeX. El comando se usa igual que siempre (ver sección 12.3.12), con un argumento opcional que determina el tipo de etiqueta que se usará para la lista. Por ejemplo, si queremos que en vez de números se usen letras mayúsculas, basta usar `\begin{enumerate}[A]`:

A Primer ítem.

B Segundo ítem.

Si queremos etiquetas de la forma "1.-", `\begin{enumerate}[1.-]`:

1.- Primer ítem.

2.- Segundo ítem.

Si deseamos insertar un texto que no cambie de una etiqueta a otra, hay que encerrarlo entre paréntesis cursivos (`\begin{enumerate}[{Caso} A:]`):

Caso A: Primer ítem.

Caso B: Segundo ítem.

12.8.5. Color.

A través de PostScript es posible introducir color en documentos LaTeX. Para ello, incluimos en el preámbulo el paquete `color.sty`:

```
\usepackage{color}
```

De este modo, está disponible el comando `\color`, que permite especificar un color, ya sea por nombre (en el caso de algunos colores predefinidos), por su código `rgb` (red-green-blue) o código `cmyk` (cian-magenta-yellow-black). Por ejemplo:

Un texto en azul
Un texto en un segundo color
Un texto en un tercer color

```
Un texto en {\color{blue} azul}

Un texto en un
{\color[rgb]{1,0,1} segundo color}

Un texto en un
{\color[cmyk]{.3,.5,.75,0} tercer color}
```

Los colores más frecuentes (azul, amarillo, rojo, etc.) se pueden dar por nombre, como en este ejemplo. Si se da el código `rgb`, se deben especificar tres números entre 0 y 1, que indican la cantidad

de rojo, verde y azul que constituyen el color deseado. En el ejemplo, le dimos máxima cantidad de rojo y azul, y nada de verde, con lo cual conseguimos un color violeta. Si se trata del código `cmyk` los números a especificar son cuatro, indicando la cantidad de cian, magenta, amarillo y negro. En el ejemplo anterior pusimos una cantidad arbitraria de cada color, y resultó un color café. Es evidente que el uso de los códigos `rgb` y `cmyk` permite explorar infinidad de colores.

Observar que `\color` funciona de modo análogo a los comandos de cambio de font de la sección 12.3.13, de modo que si se desea restringir el efecto a una porción del texto, hay que encerrar dicho texto entre paréntesis cursivos. Análogamente al caso de los fonts, existe el comando `\textcolor`, que permite dar el texto a colorear como argumento:

Un texto en azul
Un texto en un segundo color
Un texto en un tercer color

```
Un texto en \textcolor{blue}{azul}

Un texto en un
\textcolor[rgb]{1,0,1}{segundo color}

Un texto en un
\textcolor[cmyk]{.3,.5,.75,0}{tercer color}
```

12.9. Modificando el estilo de la página.

TEX toma una serie de decisiones por nosotros. Ocasionalmente nos puede interesar alterar el comportamiento normal. Disponemos de una serie de comandos para ello, los cuales revisaremos a continuación. Todos deben aparecer en el preámbulo, salvo en los casos que se indique.

12.9.1. Estilos de página.

a) Números de página.

Si se desea que los números de página sean arábicos (1, 2, 3...):

```
\pagenumbering{arabic}
```

Para números romanos (i, ii, iii,...):

```
\pagenumbering{roman}
```

`arabic` es el default.

b) Estilo de página.

El comando `\pagestyle` determina dónde queremos que vayan los números de página:

`\pagestyle{plain}`	Números de página en el extremo inferior, al centro de la página. (Default para estilos `article`, `report`.)
`\pagestyle{headings}`	Números de página y otra información (título de sección, etc.) en la parte superior de la página. (Default para estilo `book`.)
`\pagestyle{empty}`	Sin números de página.

12.9.2. Corte de páginas y líneas.

TEX tiene modos internos de decidir cuándo cortar una página o una línea. Al preparar la versión final de nuestro documento, podemos desear coartar sus decisiones. En todo caso, no hay que hacer esto antes de preparar la versión verdaderamente final, porque agregar, modificar o quitar texto puede alterar los puntos de corte de líneas y páginas, y los cortes inconvenientes pueden resolverse solos.

Los comandos de esta sección no van en el preámbulo, sino en el interior del texto.

Corte de líneas.

En la página 379 ya vimos un ejemplo de inducción de un corte de línea en un punto deseado del texto, al dividir una palabra en sílabas.

Cuando el problema no tiene relación con sílabas disponemos de dos comandos:

`\newline`	Corta la línea y pasa a la siguiente en el punto indicado.
`\linebreak`	Lo mismo, pero justificando la línea para adecuarla a los márgenes.

Un corte de línea
no justificado a los márgenes
en curso.

```
Un corte de l\'{\i}nea\newline
no justificado a los m\'argenes
en curso.
```

Un corte de línea
justificado a los márgenes
en curso.

```
Un corte de l\'{\i}nea\linebreak
justificado a los m\'argenes
en curso.
```

Observemos cómo en el segundo caso, en que se usa `\linebreak`, la separación entre palabras es alterada para permitir que el texto respete los márgenes establecidos.

Corte de páginas.

Como para cortar líneas, existe un modo violento y uno sutil:

`\newpage`	Cambia de página en el punto indicado. Análogo a `\newline`.
`\clearpage`	Lo mismo, pero ajustando los espacios verticales en el texto para llenar del mejor modo posible la página.

`\clearpage`, sin embargo, no siempre tiene efectos visibles. Dependiendo de la cantidad y tipo de texto que quede en la página, los espacios verticales pueden o no ser ajustados, y si no lo son, el resultado termina siendo equivalente a un `\newpage`. TEX decide en última instancia qué es lo óptimo.

Adicionalmente, tenemos el comando:

`\enlargethispage{<longitud>}`	Cambia el tamaño de la página actual en la cantidad `<longitud>`.

(Las unidades de longitud que maneja TEX se revisan a continuación.)

Unidades de longitud y espacios.

a) Unidades.

TEX reconoce las siguientes unidades de longitud:

`cm`	centímetro
`mm`	milímetro
`in`	pulgada
`pt`	punto (1/72 pulgadas)
`em`	ancho de una "M" en el font actual
`ex`	altura de una "x" en el font actual

Las cuatro primeras unidades son absolutas; las últimas dos, relativas, dependiendo del tamaño del font actualmente en uso.

Las longitudes pueden ser números enteros o decimales, positivos o negativos:

```
1cm     1.6in     .58pt     -3ex
```

b) Cambio de longitudes.

TEX almacena los valores de las longitudes relevantes al texto en comandos especiales:

`\parindent`	Sangría.
`\textwidth`	Ancho del texto.
`\textheight`	Altura del texto.
`\oddsidemargin`	Margen izquierdo menos 1 pulgada.
`\topmargin`	Margen superior menos 1 pulgada.
`\baselineskip`	Distancia entre la base de dos líneas de texto consecutivas.
`\parskip`	Distancia entre párrafos.

Todas estas variables son modificables con los comandos `\setlength`, que le da a una variable un valor dado, y `\addtolength`, que le suma a una variable la longitud especificada. Por ejemplo:

`\setlength{\parindent}{0.3em}` (`\parindent` = 0.3 cm.)

`\addtolength{\parskip}{1.5cm}` (`\parskip` = `\parskip` + 1.5 cm.)

Por default, el ancho y altura del texto, y los márgenes izquierdo y superior, están definidos de modo que quede un espacio de una pulgada ($\simeq$ 2,56 cm) entre el borde del texto y el borde de la página.

Un problema típico es querer que el texto llene un mayor porcentaje de la página. Por ejemplo, para que el margen del texto en los cuatro costados sea la mitad del default, debemos introducir los comandos:

```
\addtolength{\textwidth}{1in}
\addtolength{\textheight}{1in}
\addtolength{\oddsidemargin}{-.5in}
\addtolength{\topmargin}{-.5in}
```

Las dos primeras líneas aumentan el tamaño horizontal y vertical del texto en 1 pulgada. Si luego restamos media pulgada del margen izquierdo y el margen superior, es claro que la distancia entre el texto y los bordes de la página sera de media pulgada, como deseábamos.

c) Espacios verticales y horizontales.

Se insertan con `\vspace` y `\hspace`:

`\vspace{3cm}`	Espacio vertical de 3 cm.
`\hspace{3cm}`	Espacio horizontal de 3 cm.

Algunos ejemplos:

Un primer párrafo de un pequeño texto.

Y un segundo párrafo separado del otro.

Tres palabras separadas del resto.

```
Un primer p\'arrafo de un
peque\~no texto.

\vspace{1cm}
Y un segundo p\'arrafo
separado del otro.

Tres\hspace{.5cm}palabras
\hspace{.5cm}separadas
del resto.
```

Si por casualidad el espacio vertical impuesto por `\vspace` debiese ser colocado al comienzo de una página, TEX lo ignora. Sería molesto visualmente que en algunas páginas el texto comenzara algunos centímetros más abajo que en el resto. Lo mismo puede ocurrir si el espacio horizontal de un `\hspace` queda al comienzo de una línea.

Los comandos `\vspace*{<longitud>}` y `\hspace*{<longitud>}` permiten que el espacio en blanco de la `<longitud>` especificada no sea ignorado. Ello es útil cuando invariablemente queremos ese espacio vertical u horizontal, aunque sea al comienzo de una página o una línea —por ejemplo, para insertar una figura.

12.10. Figuras.

Lo primero que hay que decir en esta sección es que LATEX es un excelente procesador de texto, tanto convencional como matemático. Las figuras, sin embargo, son un problema aparte.

LATEX provee un ambiente `picture` que permite realizar dibujos simples. Dentro de la estructura `\begin{picture}` y `\end{picture}` se pueden colocar una serie de comandos para dibujar líneas, círculos, óvalos y flechas, así como para posicionar texto. Infortunadamente, el proceso de

ejecutar dibujos sobre un cierto umbral de complejidad puede ser muy tedioso para generarlo directamente. Existe software (por ejemplo, `xfig`) que permite superar este problema, pudiéndose dibujar con el mouse, exportando el resultado al formato `picture` de LaTeX. Sin embargo, `picture` tiene limitaciones (no se pueden hacer líneas de pendiente arbitraria), y por tanto no es una solución óptima.

Para obtener figuras de buena calidad es imprescindible recurrir a lenguajes gráficos externos, y LaTeX da la posibilidad de incluir esos formatos gráficos en un documento. De este modo, tanto el texto como las figuras serán de la más alta calidad. Las dos mejores soluciones son utilizar Metafont o PostScript. Metafont es un programa con un lenguaje de programación gráfico propio. De hecho, los propios fonts de LaTeX fueron creados usando Metafont, y sus capacidades permiten hacer dibujos de complejidad arbitraria. Sin embargo, los dibujos resultantes no son trivialmente reescalables, y exige aprender un lenguaje de programación específico.

Una solución mucho más versátil, y adoptada como el estándar en la comunidad de usuarios de LaTeX, es el uso de PostScript. Como se mencionó brevemente en la sección 8.12, al imprimir, una máquina UNIX convierte el archivo a formato PostScript, y luego lo envía a la impresora. Pero PostScript sirve más que para imprimir, siendo un lenguaje de programación gráfico completo, con el cual podemos generar imágenes de gran calidad, y reescalables sin pérdida de resolución. Además, muchos programas gráficos permiten exportar sus resultados en formato PostScript. Por lo tanto, podemos generar nuestras figuras en alguno de estos programas (`xfig` es un excelente software, que satisface la mayor parte de nuestras necesidades de dibujos simples; `octave` o `gnuplot` pueden ser usados para generar figuras provenientes de cálculos científicos, etc.), lo cual creará un archivo con extensión `.ps` (PostScript) o `.eps` (PostScript encapsulado).[4] Luego introducimos la figura en el documento LaTeX, a través del paquete `graphicx`.

12.10.1. graphicx.sty

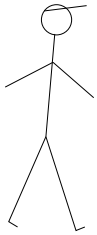
Si nuestra figura está en un archivo `figura.eps`, la instrucción a utilizar es:

```
\documentclass[12pt]{article}
\usepackage{graphicx}
\begin{document}
... Texto ...
\includegraphics[width=w, height=h]{figura.eps}
...
\end{document}
```

Los parámetros `width` y `height` son opcionales y puede omitirse uno para que el sistema escale de acuerdo al parámetro dado. Es posible variar la escala completa de la figura o rotarla usando comandos disponibles en `graphicx`.

[4] `eps` es el formato preferido, pues contiene información sobre las dimensiones de la figura, información que es utilizada por LaTeX para insertar ésta adecuadamente en el texto.

Una figura aquí:



puede hacer más agradable el
texto.

```
Una figura aqu\'{\i}:

\begin{center}
\includegraphics[height=3cm]{figura.eps}
\end{center}

puede hacer m\'as agradable
el texto.
```

En este ejemplo, indicamos sólo la altura de la figura (3cm). El ancho fue determinado de modo que las proporciones de la figura no fueran alteradas. Si no se especifica ni la altura ni el ancho, la figura es insertada con su tamaño natural.

Observemos también que pusimos la figura en un ambiente `center`. Esto no es necesario, pero normalmente uno desea que las figuras estén centradas en el texto.

12.10.2. Ambiente `figure`.

Insertar una figura es una cosa. Integrarla dentro del texto es otra. Para ello está el ambiente `figure`, que permite: (a) posicionar la figura automáticamente en un lugar predeterminado o especificado por el usuario; (b) numerar las figuras; y (c) agregar un breve texto explicativo junto a la figura.

Coloquemos la misma figura de la sección anterior dentro de un ambiente `figure`. El input:

```
\begin{figure}[h]
\begin{center}
\includegraphics[height=3cm]{figura.eps}
\end{center}
\caption{Un sujeto caminando.}
\label{caminando}
\end{figure}
```

da como resultado:

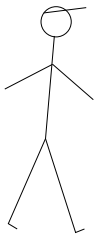


Figura 12.1: Un sujeto caminando.

`figure` delimita lo que en TEX se denomina un *objeto flotante*, es decir, un objeto cuya posición no está determinada *a priori*, y se ajusta para obtener los mejores resultados posibles. TEX considera (de acuerdo con la tradición), que la mejor posición para colocar una figura es al principio o al final

de la página. Además, lo ideal es que cada página tenga un cierto número máximo de figuras, que ninguna figura aparezca en el texto antes de que sea mencionada por primera vez, y que, por supuesto, las figuras aparezcan en el orden en que son mencionadas. Éstas y otras condiciones determinan la posición que un objeto flotante tenga al final de la compilación. Uno puede forzar la decisión de LaTeX con el argumento opcional de `figure`:

`t`	(*top)*	extremo superior de la página
`b`	(*bottom*)	extremo inferior de la página
`h`	(*here*)	aquí, en el punto donde está el comando
`p`	(*page of floats*)	en una página separada al final del texto

El argumento adicional `!` suprime, para ese objeto flotante específico, cualquier restricción que exista sobre el número máximo de objetos flotantes en una página y el porcentaje de texto mínimo que debe haber en una página.

Varios de estos argumentos se pueden colocar simultánemente, su orden dictando la prioridad. Por ejemplo,

```
\begin{figure}[htbp]
...
\end{figure}
```

indica que la figura se debe colocar como primera prioridad aquí mismo; si ello no es posible, al comienzo de página (ésta o la siguiente, dependiendo de los detalles de la compilación), y así sucesivamente.

Además, `figure` numera automáticamente la figura, colocando el texto "Figura N:", y `\caption` permite colocar una leyenda, centrada en el texto, a la figura. Puesto que la numeración es automática, las figuras pueden ser referidas simbólicamente con `\label` y `\ref` (sección 12.6). Para que la referencia sea correcta, `\label` debe estar dentro del argumento de `\caption`, o después, como aparece en el ejemplo de la Figura 12.1 (`\ref{caminando}`!).

Finalmente, notemos que la figura debió ser centrada explícitamente con `center`. `figure` no hace nada más que tratar la figura como un objeto flotante, proporcionar numeración y leyenda. El resto es responsabilidad del autor.

12.11. Cartas.

Para escribir cartas debemos emplear el estilo `letter` en vez del que hemos utilizado hasta ahora, `article`. Comandos especiales permiten escribir una carta, poniendo en lugares adecuados la dirección del remitente, la fecha, la firma, etc.

A modo de ejemplo, consideremos el siguiente input:

```
\documentclass[12pt]{letter}

\usepackage[spanish]{babel}

\begin{document}

\address{Las Palmeras 3425\\
\~Nu\~noa, Santiago}
\date{9 de Julio de 1998}
```

```
\signature{Pedro P\'erez \\ Secretario}

\begin{letter}{Dr.\ Juan P\'erez \\ Las Palmeras 3425 \\
\~Nu\~noa, Santiago}
\opening{Estimado Juan}

A\'un no tenemos novedades.

Parece incre\'{\i}ble, pero los recientes acontecimientos nos han superado,
a pesar de nuestros esfuerzos.  Esperamos que mejores tiempos nos
aguarden.

\closing{Saludos,}
\cc{Arturo Prat \\ Luis Barrios}

\end{letter}
\end{document}
```

El resultado se encuentra en la próxima página.

Las Palmeras 3425
Ñuñoa, Santiago

9 de Julio de 1998

Dr. Juan Pérez
Las Palmeras 3425
Ñuñoa, Santiago

Estimado Juan

Aún no tenemos novedades.

Parece increíble, pero los recientes acontecimientos nos han superado, a pesar de nuestros esfuerzos. Esperamos que mejores tiempos nos aguarden.

Saludos,

Pedro Pérez
Secretario

Copia a: Arturo Prat
Luis Barrios

Observemos que el texto de la carta está dentro de un ambiente `letter`, el cual tiene un argumento obligatorio, donde aparece el destinatario de la carta (con su dirección opcionalmente).

Los comandos disponibles son:

`\address{<direccion>}`	`<direccion>` del remitente.
`\signature{<firma>}`	`<firma>` del remitente.
`\opening{<apertura>}`	Fórmula de `<apertura>`.
`\closing{<despedida>}`	Fórmula de `<despedida>`.
`\cc{<copias>}`	Receptores de `<copias>` (si los hubiera).

Uno puede hacer más de una carta con distintos ambientes `letter` en un mismo archivo. Cada una tomará el mismo remitente y firma dados por `\address` y `\signature`. Si deseamos que `\address` o `\signature` valgan sólo para una carta particular, basta poner dichos comandos entre el `\begin{letter}` y el `\opening` correspondiente.

Por ejemplo, la siguiente estructura:

```
\documentclass[12pt]{letter}
\begin{document}
\address{<direccion remitente>}
\date{<fecha>}
\signature{<firma>}

\begin{letter}{<destinatario 1>}
\opening<apertura 1>
...
\end{letter}

\begin{letter}{<destinatario 2>}
\address{<direccion remitente 2>}
\signature{<firma 2>}
\opening<apertura 2>
...
\end{letter}

\begin{letter}{<destinatario 3>}
\opening<apertura 3>
...
\end{letter}
\end{document}
```

dará origen a tres cartas con la misma dirección de remitente y firma, salvo la segunda.

En todos estos comandos, líneas sucesivas son indicadas con `\\`.

12.12. LaTeX y el formato pdf.

Junto con PostScript, otro formato ampliamente difundido para la transmisión de archivos, especialmente a través de Internet, es el formato `pdf` (Portable Document Format). Para generar un

archivo `pdf` con LaTeX es necesario compilarlo con `pdflatex`. Así, `pdflatex <archivo>` generará un archivo `<archivo>.pdf` en vez del `<archivo>.dvi` generado por el compilador usual.

Si nuestro documento tiene figuras, sólo es posible incluirlas en el documento si están también en formato `pdf`. Por tanto, si tenemos un documento con figuras en PostScript, debemos introducir dos modificaciones antes de compilar con `pdflatex`:

a) Cambiar el argumento de `\includegraphics` (sección 12.10) de `<archivo_figura>.eps` a `<archivo_figura>.pdf`.

b) Convertir las figuras PostScript a `pdf` (con `epstopdf`, por ejemplo). Si tenemos una figura en el archivo `<archivo_figura>.eps`, entonces `epstopdf <archivo_figura>.eps` genera el archivo correspondiente `<archivo_figura>.pdf`.

Observar que el mismo paquete `graphicx` descrito en la sección 12.10 para incluir figuras PostScript permite, sin modificaciones, incluir figuras en `pdf`.

12.13. Modificando LaTeX.

Esta sección se puede considerar "avanzada". Normalmente uno se puede sentir satisfecho con el desempeño de LaTeX, y no es necesaria mayor intervención. A veces, dependiendo de la aplicación y del autor, nos gustaría modificar el comportamiento default. Una alternativa es definir nuevos comandos que sean útiles para nosotros. Si esos nuevos comandos son abundantes, o queremos reutilizarlos frecuentemente en otros documentos, lo conveniente es considerar crear un nuevo paquete o incluso una nueva clase. Examinaremos a continuación los elementos básicos de estas modificaciones.

12.13.1. Definición de nuevos comandos.

El comando `\newcommand`

Un nuevo comando se crea con:

```
\newcommand{<comando>}{<accion>}
```

El caso más sencillo es cuando una estructura se repite frecuentemente en nuestro documento. Por ejemplo, digamos que un sujeto llamado Cristóbal no quiere escribir su nombre cada vez que aparece en su documento:

Mi nombre es Cristóbal.
Sí, como oyes, Cristóbal.
Cristóbal Loyola.

```
\newcommand{\nombre}{Crist\'obal}
...
\begin{document}
...
Mi nombre es \nombre. S\'{\i}, como oyes,
\nombre. \nombre\ Loyola.
```

Un `\newcommand` puede aparecer en cualquier parte del documento, pero lo mejor es que esté en el preámbulo, de modo que sea evidente qué nuevos comandos están disponibles en el presente documento. Observemos además que la definición de un comando puede contener otros comandos (en este caso, `\'`). Finalmente, notamos que ha sido necesario agregar un espacio explícito con `\ `, al escribir "Cristóbal Loyola": recordemos que un comando comienza con un backslash y termina

con el primer carácter que no es letra. Por tanto, `\nombre Loyola` ignora el espacio al final de `\nombre`, y el output sería "CristóbalLoyola".

También es posible definir comandos que funcionen en modo matemático:

Sea $\dot{x}$ la velocidad, de modo que $\dot{x}(t) > 0$ si $t < 0$.

```
\newcommand{\vel}{\dot x}

Sea $\vel$ la velocidad, de modo que
$ \vel(t)> 0$ si $t<0$.
```

Como `\vel` contiene un comando matemático (`\dot`), `\vel` sólo puede aparecer en modo matemático.

Podemos también incluir la apertura de modo matemático en la definición de `\vel`: `\newcommand{\vel}{$\dot x$}`. De este modo, `\vel` (no `$\vel$`) da como output directamente $\dot{x}$. Sin embargo, esta solución no es óptima, porque la siguiente ocurrencia de `\vel` da un error. En efecto, si `\vel` = `$\dot x$`, entonces `$ \vel(t)>0$` = `$ $\dot x$> 0$`. En tal caso, LaTeX ve que un modo matemático se ha abierto y cerrado inmediatamente, conteniendo sólo un espacio entremedio, y luego, *en modo texto*, viene el comando `\dot`, que es matemático: LaTeX acusa un error y la compilación se detiene.

La solución a este problema es utilizar el comando `\ensuremath`, que asegura que haya modo matemático, pero si ya hay uno abierto, no intenta volverlo a abrir:

Sea $\dot{x}$ la velocidad, de modo que $\dot{x}(t) > 0$ si $t < 0$.

```
\newcommand{\vel}{\ensuremath{\dot x}}

Sea \vel\ la velocidad, de modo que
$ \vel(t)> 0$ si $t<0$.
```

Un caso especial de comando matemático es el de operadores tipo logaritmo (ver Tabla 12.9). Si queremos definir una traducción al castellano de `\sin`, debemos usar el comando `\DeclareMathOperator` disponible via `amsmath`:

Ahora podemos escribir en castellano, $\sen x$.

```
\usepackage{amsmath}
\DeclareMathOperator{\sen}{sen}
...
Ahora podemos escribir en castellano, $\sen x$.
```

A diferencia de `\newcommand`, `\DeclareMathOperator` sólo puede aparecer en el preámbulo del documento.

Un nuevo comando puede también ser usado para ahorrar tiempo de escritura, reemplazando comandos largos de LaTeX:

1. El primer caso.
2. Ahora el segundo.
3. Y el tercero.

```
\newcommand{\be}{\begin{enumerate}}
\newcommand{\ee}{\end{enumerate}}

\be
\item El primer caso.
\item Ahora el segundo.
\item Y el tercero.
\ee
```

Nuevos comandos con argumentos

Podemos también definir comandos que acepten argumentos. Si el sujeto anterior, Cristóbal, desea escribir cualquier nombre precedido de "Nombre:" en itálica, entonces puede crear el siguiente comando:

Nombre: Cristóbal
Nombre: Violeta

```
\newcommand{\nombre}[1]{\textit{Nombre:} #1}

\nombre{Crist\'obal}

\nombre{Violeta}
```

Observemos que `\newcommand` tiene un argumento opcional, que indica el número de argumentos que el nuevo comando va a aceptar. Esos argumentos se indican, dentro de la definición del comando, con `#1`, `#2`, etc. Por ejemplo, consideremos un comando que acepta dos argumentos:

$f(x, y) + f(x_3, y*) = 0$.

```
\newcommand{\fn}[2]{f(#1,#2)}

$$ \fn{x}{y} + \fn{x_3}{y*} = 0 \ . $$
```

En los casos anteriores, todos los argumentos son obligatorios. LaTeX permite definir comandos con un (sólo un) argumento opcional. Si el comando acepta n argumentos, el argumento opcional es el `#1`, y se debe indicar, en un segundo paréntesis cuadrado, su valor default. Así, podemos modificar el comando `\fn` del ejemplo anterior para que el primer argumento sea opcional, con valor default x:

$f(x, y) + f(x_3, y*) = 0$.

```
\newcommand{\fn}[2][x]{f(#1,#2)}

$$ \fn{y} + \fn[x_3]{y*} = 0 \ . $$
```

Redefinición de comandos

Ocasionalmente no nos interesa definir un nuevo comando, sino redefinir la acción de un comando preexistente. Esto se hace con `\renewcommand`:

La antigua versión de `ldots`: La antigua versi\'on de {\tt ldots}: \ldots
...
La nueva versión de `ldots`: •••

```
\renewcommand{\ldots}{\textbullet \textbullet
\textbullet}

La nueva versi\'on de {\tt ldots}: \ldots
```

Párrafos y cambios de línea dentro de comandos

En el segundo argumento de `\newcommand` o `\renewcommand` puede aparecer cualquier comando de LaTeX, pero ocasionalmente la aparición de líneas en blanco (para forzar un cambio de párrafo) puede provocar problemas. Si ello ocurre, podemos usar `\par`, que hace exactamente lo mismo. Además, la definición del comando queda más compacta:

```
\newcommand{\comandolargo}{\par Un nuevo comando que incluye un cambio de
  p\'arrafo, porque deseamos incluir bastante texto.\par \'Este es el
  nuevo p\'arrafo.\par}

Observemos en acci\'on el comando: \comandolargo Listo.
```

da como resultado:

Observemos en acción el comando:
Un nuevo comando que incluye un cambio de párrafo, porque deseamos incluir bastante texto.
Éste es el nuevo párrafo.
Listo.

Un ejemplo más útil ocurre cuando queremos asegurar un cambio de párrafo, por ejemplo, para colocar un título de sección:

Observemos en acción el comando:

Sección: Ejemplo

Listo.

```
\newcommand{\seccion}[1]{\par\vspace{.5cm}
{\bf Secci\'on: #1}\par\vspace{.5cm}}

Observemos en acci\'on el comando:
\seccion{Ejemplo} Listo.
```

Además de las líneas en blanco, los cambios de línea pueden causar problemas dentro de la definición de un nuevo comando. El ejemplo anterior, con el comando `\seccion`, es un buen ejemplo: notemos que cuando se definió, pusimos un cambio de línea después de `\vspace{.5cm}`. Ese cambio de línea es interpretado (como todos los cambios de línea) como un espacio en blanco, y es posible que, bajo ciertas circunstancias, ese espacio en blanco produzca un output no deseado. Para ello basta utilizar sabiamente el carácter `%`, que permite ignorar todo el resto de la línea, *incluyendo el cambio de línea*. Ilustremos lo anterior con los siguientes tres comandos, que subrayan (comando `\underline`) una palabra, y difieren sólo en el uso de `%` para borrar cambios de línea:

Notar la diferencia entre:
Un texto de prueba , Un texto de prueba , y Un texto de prueba.

```
\newcommand{\texto}{
Un texto de prueba
}
\newcommand{\textodos}{%
Un texto de prueba
}
\newcommand{\textotres}{%
Un texto de prueba%
}

Notar la diferencia entre:

\underline{\texto},
\underline{\textodos},
y
\underline{\textotres}.
```

`\texto` conserva espacios en blanco antes y después del texto, `\textodos` sólo el espacio en

blanco después del texto, y `\textotres` no tiene espacios en blanco alrededor del texto.

Nuevos ambientes

Nuevos ambientes en LaTeX se definen con `\newenvironment`:

```
\newenvironment{<ambiente>}{<comienzo ambiente>}{<final ambiente>}
```

define un ambiente `<ambiente>`, tal que `\begin{ambiente}` ejecuta los comandos `<comienzo ambiente>`, y `\end{ambiente}` ejecuta los comandos `<final ambiente>`.

Definamos un ambiente que, al comenzar, cambia el font a itálica, pone una línea horizontal (`\hrule`) y deja un espacio vertical de .3cm, y que al terminar cambia de párrafo, coloca `XXX` en sans serif, deja un nuevo espacio vertical de .3cm, y vuelve al font roman:

```
\newenvironment{na}{\it \hrule \vspace{.3cm}}{\par\sf XXX \vspace{.3cm}\rm}
```

Entonces, con

```
\begin{na}
  Hola a todos. Es un placer saludarlos en este d\'{\i}a tan especial.

Nunca esper\'e una recepci\'on tan calurosa.
\end{na}
```

obtenemos:

Hola a todos. Es un placer saludarlos en este día tan especial.
Nunca esperé una recepción tan calurosa.
XXX

Los nuevos ambientes también pueden ser definidos de modo que acepten argumentos. Como con `\newcommand`, basta agregar como argumento opcional a `\newenvironment` un número que indique cuántos argumentos se van a aceptar:

```
\newenvironment{<ambiente>}[n]{<comienzo ambiente>}{<final ambiente>}
```

Dentro de `<comienzo ambiente>`, se alude a cada argumento como `#1`, `#2`, etc. Los argumentos no pueden ser usados en los comandos de cierre del ambiente (`<final ambiente>`). Por ejemplo, modifiquemos el ambiente `na` anterior, de modo que en vez de colocar una línea horizontal al comienzo, coloque lo que le indiquemos en el argumento:

```
\newenvironment{na}[1]{\it #1 \vspace{.3cm}}{\par\sf XXX\hrule\vspace{.3cm}\rm}
```

Ahora usémoslo dos veces, cada una con un argumento distinto:

El mismo ejemplo anterior, ahora es

Hola a todos...
XXX

Pero podemos ahora cambiar el comienzo:
XXX Hola a todos...

XXX

```
El mismo ejemplo anterior, ahora es

\begin{na}{\hrule}
  Hola a todos...
\end{na}

Pero podemos ahora cambiar el comienzo:

\begin{na}{\it XXX}
  Hola a todos...
\end{na}
```

12.13.2. Creación de nuevos paquetes y clases

Si la cantidad de nuevos comandos y/o ambientes que necesitamos en nuestro documento es suficientemente grande, debemos considerar crear un nuevo paquete o una nueva clase. Para ello hay que tener clara la diferencia entre uno y otro. En general, se puede decir que si nuestros comandos involucran alterar la apariencia general del documento, entonces corresponde crear una nueva clase (`.cls`). Si, por el contrario, deseamos que nuestros comandos funcionen en un amplio rango de circunstancias, para diversas apariencias del documento, entonces lo adecuado es un paquete (`.sty`).

Consideremos por ejemplo la experiencia de los autores de estos apuntes. Para crear estos apuntes necesitamos básicamente la clase `book`, con ciertas modificaciones: márgenes más pequeños, inclusión automática de los paquetes `amsmath`, `babel` y `graphicx`, entre otros, y definición de ciertos ambientes específicos. Todo ello afecta la apariencia de este documento, cambiándola de manera apreciable, pero a la vez de un modo que en general no deseamos en otro tipo de documento. Por ello lo hemos compilado usando una clase adecuada, llamada `mfm2.cls`.

Por otro lado, uno de los autores ha necesitado escribir muchas tareas, pruebas y controles de ayudantía en su vida, y se ha convencido de que su trabajo es más fácil creando una clase `tarea.cls`, que sirve para esos tres propósitos, definiendo comandos que le permiten especificar fácilmente la fecha de entrega de la tarea, o el tiempo disponible para una prueba, los nombres del profesor y el ayudante, etc., una serie de comandos específicos para sus necesidades.

Sin embargo, tanto en este documento que usa `mfm2.cls`, como en las tareas y pruebas que usan `tarea.cls`, se utilizan algunos comandos matemáticos que no vienen con LaTeX, pero que son recurrentes, como `\sen` (la función seno en castellano), `\modulo` (el módulo de un vector), o `\TLaplace` (la transformada de Laplace). Para que estos comandos estén disponibles en cualquier tipo de documento, necesitamos reunirlos en un paquete, en este caso `addmath.sty`. De este modo, `mfm2.cls`, `tarea.cls` o cualquier otra clase pueden llamar a este paquete y utilizar sus comandos.

Estructura básica.

La estructura básica de un paquete o una clase es:

a) Identificación: Información general (nombre del paquete, fecha de creación, etc.). (Obligatoria.)

b) Declaraciones preliminares: Opcionales, dependiendo del paquete o clase en cuestión.

c) Opciones: Comandos relacionados con el manejo de las opciones con las cuales el paquete o clase pueden ser invocados. (Opcional.)

d) Más declaraciones: Aquí van los comandos que constituyen el cuerpo de la clase o paquete. (Obligatoria: si no hay ninguna declaración, el paquete o clase no hace nada, naturalmente.)

La identificación está consituida por las siguientes dos líneas, que deben ir al comienzo del archivo:

```
\NeedsTeXFormat{LaTeX2e}
\ProvidesPackage{<paquete>}[<fecha> <otra informacion>]
```

La primera línea indica a LaTeX que éste es un archivo para LaTeX 2_ε. La segunda línea especifica que se trata de un paquete, indicando el nombre del mismo (es decir, el nombre del archivo sin extensión) y, opcionalmente, la fecha (en formato `YYYY/MM/DD`) y otra información relevante. Por ejemplo, nuestro paquete `addmath.sty` comienza con las líneas:

```
\NeedsTeXFormat{LaTeX2e}
\ProvidesPackage{addmath}[1998/09/30 Macros matematicos adicionales (VM)]
```

Si lo que estamos definiendo es una clase, usamos el comando `\ProvidesClass`. Para nuestra clase `mfm2.cls`:

```
\NeedsTeXFormat{LaTeX2e}
\ProvidesClass{mfm2}[2002/03/25 Estilo para apuntes MFM II (VM)]
```

A continuación de la identificación vienen los comandos que se desean incorporar a través de este paquete o clase.

Como hemos dicho, `addmath.sty` contiene muchos nuevos comandos matemáticos que consideramos necesario definir mientras escribíamos estos apuntes. Veamos los contenidos de una versión simplificada de dicho paquete:

```
\NeedsTeXFormat{LaTeX2e}
\ProvidesPackage{addmath}[1998/09/30 Macros matematicos adicionales (VM)]

\newcommand{\prodInt}[2]{\ensuremath \left(\, #1\, |\, #2\, \right ) }
\newcommand{\promedio}[1]{\langle #1 \rangle}
\newcommand{\intii}{\int_{-\infty}^{\infty}}
\newcommand{\grados}{\ensuremath{^\circ}}
\newcommand{\Hipergeometrica}[4]{{}_2F_1\left (#1, #2, #3\, ; #4\right )}
...
```

De este modo, incluyendo en nuestro documento el paquete con `\usepackage{addmath}`, varios nuevos comandos están disponibles:

$(x \mid y)$	`\prodInt{x}{y}`
$\langle x \rangle$	`\promedio{x}`
$\displaystyle\int_{-\infty}^{\infty} dz\, f(z)$	`\intii dz\, f(z)`
$\angle ABC = 90^\circ$	`\angle\, ABC = 90\grados`
${}_2F_1\,(a, b, c\,; d)$	`\Hipergeometrica{a}{b}{c}{d}`

Incluyendo otros paquetes y clases

Los comandos `\RequirePackage` y `\LoadClass` permiten cargar un paquete o una clase, respectivamente.[5]. Esto es de gran utilidad, pues permite construir un nuevo paquete o clase aprovechando la funcionalidad de otros ya existentes.

Así, nuestro paquete `addmath.sty` define bastantes comandos, pero nos gustaría definir varios más que sólo pueden ser creados con las herramientas de $\mathcal{AMS}$-LaTeX. Cargamos entonces en `addmath.sty` el paquete `amsmath` y otros relacionados, y estamos en condiciones de crear más comandos:

```
\NeedsTeXFormat{LaTeX2e}
\ProvidesPackage{addmath}[1998/09/30 Macros matematicos adicionales (VM)]

\RequirePackage{amsmath}
\RequirePackage{amssymb}
\RequirePackage{euscript}
...
\newcommand{\norma}[1]{\ensuremath \left\lVert\, #1 \,\right\rVert}
\newcommand{\intC}{{\sideset{^*}{}\int}}
\DeclareMathOperator{\senh}{senh}
...
```

Por ejemplo:

$\| x \|$	`\norma{x}`
${}^*\!\int dz\, f(z)$	`\intC dz \, f(z)`
$\operatorname{senh}(2y)$	`\senh (2y)`

La posibilidad de basar un archivo `.sty` o `.cls` en otro es particularmente importante para una clase, ya que contiene una gran cantidad de comandos y definiciones necesarias para compilar el documento exitosamente. Sin embargo, un usuario normal, aun cuando desee definir una nueva clase, estará interesado en modificar sólo parte del comportamiento. Con `\LoadClass`, dicho usuario puede cargar la clase sobre la cual se desea basar, y luego introducir las modificaciones necesarias, facilitando enormemente la tarea.

Por ejemplo, al preparar este documento fue claro desde el comienzo que se necesitaba esencialmente la clase `book`, ya que sería un texto muy extenso, pero también era claro que se requerían ciertas modificaciones. Entonces, en nuestra clase `mfm2.cls` lo primero que hacemos es cargar la clase `book`, más algunos paquetes necesarios (incluyendo nuestro `addmath`), y luego procedemos a modificar o añadir comandos:

```
\NeedsTeXFormat{LaTeX2e}
\ProvidesClass{mfm2}[2002/03/25 Estilo para apuntes MFM II (VM)]

\LoadClass[12pt]{book}

\RequirePackage[spanish]{babel}
```

[5]Estos comandos sólo se pueden usar en un archivo `.sty` o `.cls` Para documentos normales, la manera de cargar un paquete es `\usepackage`, y para cargar una clase es `\documentclass`.

```
\RequirePackage{enumerate}
\RequirePackage{addmath}
```

En un archivo `.sty` o un `.cls` se pueden cargar varios paquetes con `\RequirePackage`. `\LoadClass`, en cambio, sólo puede aparecer en un `.cls`, y sólo es posible usarlo una vez (ya que normalmente clases distintas son incompatibles entre sí).

Manejo de opciones

En el último ejemplo anterior, la clase `mfm2` carga la clase `book` con la opción `12pt`. Esto significa que si nuestro documento comienza con `\documentclass{mfm2}`, será compilado de acuerdo a la clase `book`, en 12 puntos. No es posible cambiar esto desde nuestro documento. Sería mejor que pudiéramos especificar el tamaño de letra fuera de la clase, de modo que `\documentclass{mfm2}` dé un documento en 10 puntos, y `\documentclass[12pt]{mfm2}` uno en 12 puntos. Para lograr esto hay que poder pasar opciones desde la clase `mfm2` a `book`.

El modo más simple de hacerlo es con `\LoadClassWithOptions`. Si `mfm2.cls` ha sido llamada con opciones `<opcion1>,<opcion2>, etc.`, entonces `book` será llamada con las mismas opciones. Por tanto, basta modificar en `mfm2.cls` la línea `\LoadClass[12pt]{book}` por:

```
\LoadClassWithOptions{book}
```

`\RequirePackageWithOptions` es el comando análogo para paquetes. Si una clase o un paquete llaman a un paquete `<paquete_base>` y desean pasarle todas las opciones con las cuales han sido invocados, basta indicarlo con:

```
\RequirePackageWithOptions{<paquete_base>}
```

El ejemplo anterior puede ser suficiente en muchas ocasiones, pero en general uno podría llamar a nuestra nueva clase, `mfm2`, con opciones que no tienen nada que ver con `book`. Por ejemplo, podríamos llamarla con opciones `spanish,12pt`. En tal caso, debería pasarle `spanish` a babel, y `12pt` a book. Más aún, podríamos necesitar definir una *nueva* opción, que no existe en ninguna de las clases o paquetes cargados por `book`, para modificar el comportamiento de `mfm2.cls` de cierta manera específica no prevista. Estas dos tareas, discriminar entre opciones antes de pasarla a algún paquete determinado, y crear nuevas opciones, constituyen un manejo más avanzado de opciones. A continuación revisaremos un ejemplo combinado de ambas tareas, extraido de la clase con la cual compilamos este texto, `mfm2.cls`.

La idea es poder llamar a `mfm2` con una opción adicional `keys`, que permita agregar al `dvi` información sobre las etiquetas (dadas con `\label`) de ecuaciones, figuras, etc., que aparezcan en el documento (veremos la utilidad y un ejemplo de esto más adelante). Lo primero es *declarar* una nueva opción, con:

```
\DeclareOption{<opcion>}{<comando>}
```

`<opcion>` es el nombre de la nueva opción a declarar, y `<comando>` es la serie de comandos que se ejecutan cuando dicha opción es especificada.

Así, nuestro archivo `mfm2.cls` debe ser modificado:

```
\NeedsTeXFormat{LaTeX2e}
\ProvidesClass{mfm2}[2002/03/25 Estilo para apuntes MFM II (VM)]
...
\DeclareOption{keys}{...}
```

```
...
\ProcessOptions\relax
...
```

Observamos que después de declarar la o las opciones (en este caso `keys`), hay que *procesarlas*, con `\ProcessOptions`.[6]

Las líneas anteriores permiten que `\documentclass{mfm2}` y `\documentclass[keys]{mfm2}` sean ambas válidas, ejecutándose o no ciertos comandos dependiendo de la forma utilizada.

Si ahora queremos que `\documentclass[keys,12pt]{mfm2}` sea una línea válida, debemos procesar `keys` dentro de `mfm2.cls`, y pasarle a `book.cls` las opciones restantes. El siguiente es el código definitivo:

```
\NeedsTeXFormat{LaTeX2e}
\ProvidesClass{mfm2}[2002/03/25 Estilo para apuntes MFM II (VM)]
\newif\ifkeys\keysfalse
\DeclareOption{keys}{\keystrue}
\DeclareOption*{\PassOptionsToClass{\CurrentOption}{book}}
\ProcessOptions\relax
\LoadClass{book}

\RequirePackage[spanish]{babel}
\RequirePackage{amsmath}
\RequirePackage{theorem}
\RequirePackage{epsfig}
\RequirePackage{ifthen}
\RequirePackage{enumerate}
\RequirePackage{addmath}
\ifkeys\RequirePackage[notref,notcite]{showkeys}\fi

<nuevos comandos de la clase mfm2.cls>
```

Sin entrar en demasiados detalles, digamos que la opción `keys` tiene el efecto de hacer que una cierta variable lógica `\ifkeys`, sea verdadera (cuarta línea del código). La siguiente línea (`\DeclareOption*...`) hace que todas las opciones que no han sido procesadas (`12pt`, por ejemplo) se pasen a la clase `book`. A continuación se procesan las opciones con `\ProcessOptions`, y finalmente se carga la clase `book`.

Las líneas siguientes cargan todos los paquetes necesarios, y finalmente se encuentran todos los nuevos comandos y definiciones que queremos incluir en `mfm2.cls`.

Observemos que la forma particular en que se carga el paquete `showkeys`. Ésa es precisamente la función de la opción `keys` que definimos: `showkeys.sty` se carga con ciertas opciones sólo si se da la opción `keys`.

¿Cuál es su efecto? Consideremos el siguiente texto de ejemplo, en que `mfm2` ha sido llamada sin la opción `keys`:

[6] `\relax` es un comando de TeX que, esencialmente, no hace nada, ni siquiera introduce un espacio en blanco, y es útil incluirlo en puntos críticos de un documento, como en este ejemplo.

```
\documentclass[12pt]{mfm2}
\begin{document}
La opci\'on \verb+keys+ resulta muy \'util cuando tengo objetos numerados
autom\'aticamente, como una ecuaci\'on:
\begin{equation}
  \label{newton}
  \vec F = m \vec a \ .
\end{equation}
y luego quiero referirme a ella: Ec.\  \eqref{newton}.
```

En el primer caso, se ha compilado sin la opción `keys`, y en el segundo con ella. El efecto es que, si se usa un `\label` en cualquier parte del documento, aparece en el margen derecho una caja con el nombre de dicha etiqueta (en este caso, `newton`). Esto es útil para cualquier tipo de documentos, pero lo es especialmente en textos como estos apuntes, muy extensos y con abundantes referencias. En tal caso, tener un modo visual, rápido, de saber los nombres de las ecuaciones sin tener que revisar trabajosamente el archivo fuente es una gran ayuda. Así, versiones preliminares pueden ser compiladas con la opción `keys`, y la versión final sin ella, para no confesar al lector nuestra mala memoria o nuestra comodidad.

Caso 1: `\documentclass[12pt]{mfm2}`

La opción `keys` resulta muy útil cuando tengo objetos numerados automáticamente, como una ecuación:

$$\vec{F} = m\vec{a} \ . \tag{1}$$

y luego quiero referirme a ella: Ec. (1).

Caso 2: `\documentclass[keys,12pt]{mfm2}`

La opción `keys` resulta muy útil cuando tengo objetos numerados automáticamente, como una ecuación:

$$\vec{F} = m\vec{a} \ . \tag{1}$$ `newton`

y luego quiero referirme a ella: Ec. (1).

12.14. Errores y advertencias.

12.14.1. Errores.

Un mensaje de error típico tiene la forma:

```
LaTeX error. See LaTeX manual for explanation.
             Type H <return> for immediate help.
! Environment itemie undefined.
\@latexerr ...or immediate help.}\errmessage {#1}
                                                  \endgroup
l.140 \begin{itemie}

?
```

La primera línea nos comunica que LaTeX ha encontrado un error. A veces los errores tienen que ver con procesos más internos, y son encontrados por TeX. Esta línea nos informa quién encontró el error.

La tercera línea comienza con un signo de exclamación. Éste es el indicador del error. Nos dice de qué error se trata.

Las dos líneas siguientes describen el error en términos de comandos de bajo nivel.

La línea 6 nos dice dónde ocurrió el error: la línea 140 en este caso. Además nos informa del texto conflictivo: `\begin{itemie}`.

En realidad, el mensaje nos indica dónde LaTeX advirtió el error por primera vez, que no es necesariamente el punto donde el error se cometió. Pero la gran mayoría de las veces la indicación es precisa. De hecho, es fácil darse cuenta, con la tercera línea
(`Environment itemie undefined`)
y la sexta (`\begin{itemie}`) que el error consistió en escribir `itemie` en vez de `itemize`. La información de LaTeX es clara en este caso y nos dice correctamente qué ocurrió y dónde.

Luego viene un `?`. LaTeX está esperando una respuesta de nosotros. Tenemos varias alternativas. Comentaremos sólo cuatro, típicamente usadas:

(a) `h <Enter>`

Solicitamos ayuda. TeX nos explica brevemente en qué cree él que consiste el error y/o nos da alguna recomendación.

(b) `x <Enter>`

Abortamos la compilación. Deberemos volver al editor y corregir el texto. Es la opción más típica cuando uno tiene ya cierta experiencia, pues el mensaje basta para reconocer el error.

(c) `<Enter>`

Ignoramos el error y continuamos la compilación. TeX hace lo que puede. En algunos casos esto no tiene consecuencias graves y podremos llegar hasta el final del archivo sin mayores problemas. En otros casos, ignorar el error puede provocar que ulteriores comandos —perfectamente válidos en principio— no sean reconocidos y, así, acumular

muchos errores más. Podemos continuar con `<Enter>` sucesivos hasta llegar al final de la compilación.

(d) `q <Enter>`

La acción descrita en el punto anterior puede llegar a ser tediosa o infinita. `q` hace ingresar a TEX en `batchmode`, modo en el cual la compilación prosigue ignorando todos los errores hasta el final del archivo, sin enviar mensajes a pantalla y por ende sin que debamos darle infinitos `<Enter>`.

Las opciones (c) y (d) son útiles cuando no entendemos los mensajes de error. Como TEX seguirá compilando haciendo lo mejor posible, al mirar el `dvi` puede que veamos más claramente dónde comenzaron a ir mal las cosas y, por tanto, por qué.

Como dijimos, LATEX indica exactamente dónde encontró el error, de modo que hemos de ponerle atención. Por ejemplo, si tenemos en nuestro documento la línea:

```
... un error inesperado\fotnote{En cualquier punto.}
puede decidir...
```

generaría el mensaje de error:

```
! Undefined control sequence.

l.249 ...un error inesperado\fotnote
                                    {En cualquier punto.}
?
```

En la línea de localización, LATEX ha cortado el texto justo después del comando inexistente. LATEX no sólo indica la línea en la cual detectó el error, sino el punto de ella donde ello ocurrió. (En realidad, hizo lo mismo —cortar la línea para hacer resaltar el problema— en el caso expuesto en la pág. 418, pero ello ocurrió en medio de comandos de bajo nivel, así que no era muy informativo de todos modos.)

Errores más comunes.

Los errores más comunes son:

a) Comando mal escrito.

b) Paréntesis cursivos no apareados.

c) Uso de uno de los caracteres especiales `#`, `$`, `%`, `&`, `_`, `{`, `}`, `~`, `^`, `\` como texto ordinario.

d) Modo matemático abierto de una manera y cerrado de otra, o no cerrado.

e) Ambiente abierto con `\begin...` y cerrado con un `\end...` distinto.

f) Uso de un comando matemático fuera de modo matemático.

g) Ausencia de argumento en un comando que lo espera.

h) Línea en blanco en ambiente matemático.

Algunos mensajes de error.

A continuación, una pequeña lista de errores (de LaTeX y TeX) en orden alfabético, y sus posibles causas.

`*`

Falta `\end{document}`. (Dar `Ctrl-C` o escribir `\end{document}` para salir de la compilación.)

`! \begin{...} ended by \end{...}`

Error e) de la Sec. 12.14.1. El nombre del ambiente en `\end{...}` puede estar mal escrito, sobra un `\begin` o falta un `\end`.

`! Double superscript` (o `subscript`).

Una expresión como `x^2^3` o `x_2_3`. Si se desea obtener x^{2^3} (x_{23}), escribir `{x^2}^3` (`{x_2}_3`).

`! Environment ... undefined.`

`\begin{...}` con un argumento que corresponde a un ambiente no definido.

`! Extra alignment tab has been changed.`

En un `tabular` o `array` sobra un `&`, falta un `\\`, o falta una `c`, `l` ó `r` en el argumento obligatorio.

`! Misplaced alignment tab character &.`

Un `&` aparece fuera de un `tabular` o `array`.

`! Missing $ inserted.`

Errores c), d), f), h) de la Sec. 12.14.1.

`! Missing {` (o `}`) `inserted.`

Paréntesis cursivos no apareados.

`! Missing \begin{document}.`

Falta `\begin{document}` o hay algo incorrecto en el preámbulo.

`! Missing number, treated as zero.`

Falta un número donde LaTeX lo espera: `\hspace{}`, `\vspace cm`, `\setlength{\textwidth}{a}`, etc.

`! Something's wrong -- perhaps a missing \item.`

Posiblemente la primera palabra después de un `\begin{enumerate}` o `\begin{itemize}` no es `\item`.

`! Undefined control sequence.`

Aparece una secuencia `\<palabra>`, donde `<palabra>` no es un comando.

12.14.2. Advertencias.

La estructura de una advertencia de LaTeX es:

`LaTeX warning. <mensaje>.`

Algunos ejemplos:

`Label '...' multiply defined.`

Dos `\label` tienen el mismo argumento.

`Label(s) may have changed. Rerun to get cross-references right.`

Los números impresos por `\ref` y `\pageref` pueden ser incorrectos, pues los valores correspondientes cambiaron respecto al contenido del `aux` generado en la compilación anterior.

`Reference '...' on page ... undefined.`

El argumento de un `\ref` o un `\pageref` no fue definido por un `\label`.

TeX también envía advertencias. Se reconocen porque no comienzan con `TeX warning`. Algunos ejemplos.

`Overfull \hbox ...`

TeX no encontró un buen lugar para cortar una línea, y puso más texto en ella que lo conveniente.

`Overfull \vbox ...`

TeX no encontró un buen lugar para cortar una página, y puso más texto en ella que lo conveniente.

`Underfull \hbox ...`

TeX construyó una línea con muy poco material, de modo que el espacio entre palabras puede ser excesivo.

`Underfull \vbox ...`

TeX construyó una página con muy poco material, de modo que los espacios verticales (entre párrafos) pueden ser excesivos.

Las advertencias de LaTeX siempre deben ser atendidas. Una referencia doblemente definida, o no compilar por segunda vez cuando LaTeX lo sugiere, generará un resultado incorrecto en el `dvi`. Una referencia no definida, por su parte, hace aparecer un signo **??** en el texto final. Todos resultados no deseados, por cierto.

Las advertencias de TeX son menos decisivas. Un overfull o underfull puede redundar en que alguna palabra se salga del margen derecho del texto, que el espaciado entre palabras en una línea sea excesivo, o que el espacio vertical entre párrafos sea demasiado. Los estándares de calidad de TeX son altos, y por eso envía advertencias frecuentemente. Pero generalmente los defectos en el resultado final son imperceptibles a simple vista, o por lo menos no son suficientes para molestarnos realmente. A veces sí, por supuesto, y hay que estar atentos. Siempre conviene revisar el texto y prestar atención a estos detalles, aunque ello sólo tiene sentido al preparar la versión definitiva del documento.