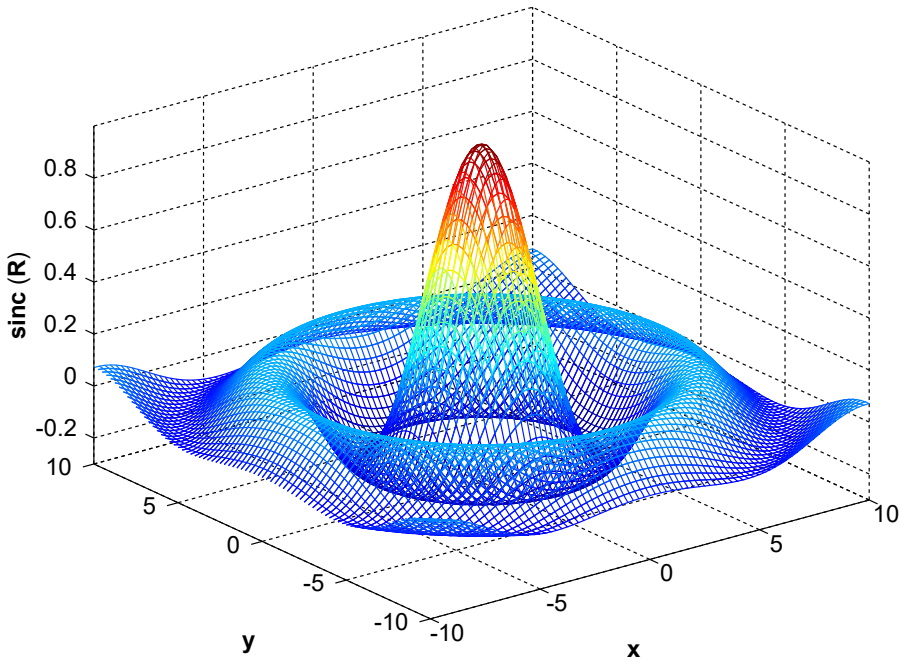
Depto. Matemática Aplicada y Ciencias de la Computación
E.T.S. Ingenieros Industriales y de Telecomunicación
Universidad de Cantabria

G404/G272: CÁLCULO I.
PRÁCTICAS DE LABORATORIO



Curso 2020/21

Índice general

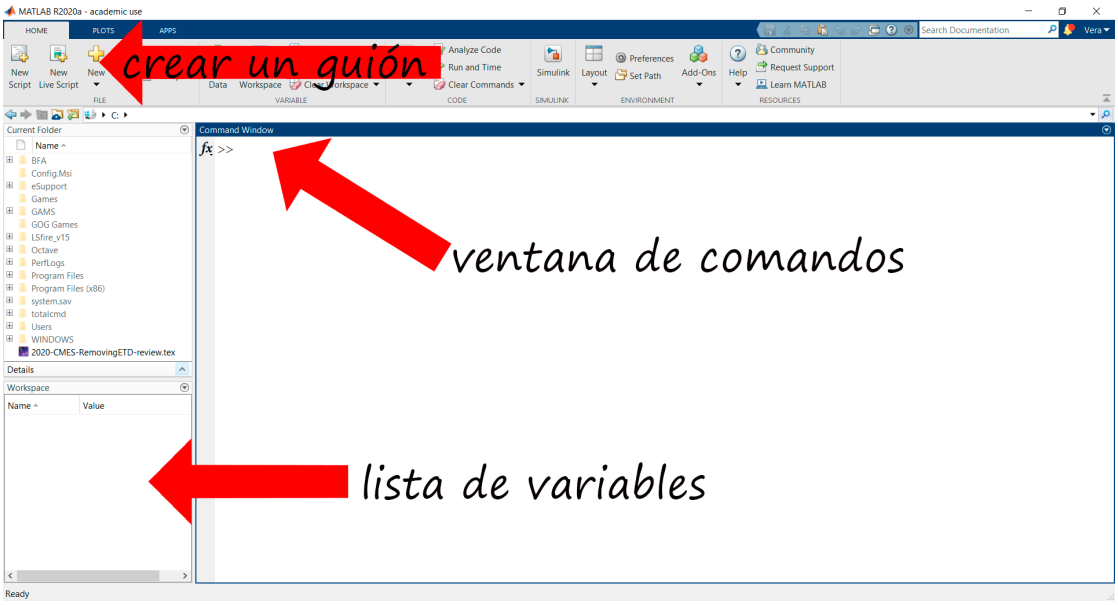
1 Introducción a MATLAB

1.1. Primer contacto con MATLAB

MATLAB es un potente paquete de software para computación científica orientado al cálculo numérico, las operaciones matriciales y muy especialmente a las aplicaciones en diversas disciplinas científicas y en la ingeniería. Puede ser utilizado como una simple calculadora, pero su interés principal radica en los cientos de funciones (tanto de propósito general como especializadas) que posee, así como en sus posibilidades para la visualización gráfica. Hoy en día MATLAB (acrónimo de MATrix LABoratory) es un software muy potente que dispone de un entorno de uso muy agradable, así como un lenguaje de programación de alto nivel.

La interfaz principal de MATLAB se divide en varias secciones. Por ahora, las más importantes para nosotros son las siguientes:

- En la parte superior se encuentra una barra de herramientas desde la que se pueden controlar todas las opciones disponibles.
- Las órdenes se escriben en la ventana de comandos, `Command Window`.
- La ventana `Workspace` proporciona información sobre las variables que están almacenadas en memoria.



1.2. Nociones básicas

1.2.1. Tipos de datos

Aunque hay más, los tipos de datos básicos en MATLAB para nosotros son los siguientes: números reales o complejos, *booleanos* (valores lógicos) y cadenas de caracteres (o *strings*). En cualquier caso, el objeto básico de trabajo en MATLAB son las matrices y los *arrays* (hipermatrices en tres o más dimensiones). Los escalares y los vectores son considerados casos particulares de matrices.

1.2.2. Uso de la ventana de comandos

Las instrucciones que terminan por punto y coma no producen salida de resultados por pantalla. Se pueden recuperar instrucciones anteriores, usando las teclas de flechas arriba y abajo.

1.2.3. Definición de variables

Para crear una nueva variable con la que trabajar, se especifica el nombre que se le quiere dar a dicha variable, seguido del signo igual `=` (operador de asignación) y el valor que se le desea asignar.

```
x = 5
y = 4;
x + y
x - y;
```

Hay ciertas reglas que han de seguirse al definir variables en MATLAB:

- El nombre de una variable puede tener como máximo 63 caracteres que pueden ser letras, números y el guión bajo.
- El primer carácter tiene que ser una letra. Por ejemplo, `matriz2` es un nombre válido, `2matriz` no lo es.
- Las mayúsculas y las minúsculas tienen valor distintivo. La variable `X` es distinta de la variable `x`.
- Dentro de un nombre de variable no puede haber espacios en blanco. `modulo1` es un nombre de variable válido, `modulo 1` no.
- Existen nombres de variable que deben evitarse porque tienen significado propio en MATLAB (son constantes predefinidas). Por ejemplo, `i, j` (unidad imaginaria), `pi` (número π), `Inf` (infinito), `NaN` (Not a Number), etc.

1.2.4. Operadores

- Operador de asignación: =
- Operadores aritméticos: +, -, *, /, ^ , .*, ./, .^
- Operadores de comparación: <, >, <=, >=, ==, ~=
- Operadores lógicos: &, |, ~

1.2.5. El editor

Para llevar a cabo una determinada tarea, en lugar de escribir las instrucciones una a una en la línea de comandos, se pueden escribir todas las órdenes (una detrás de otra) en un fichero de texto con extensión `.m` al que se denomina (**script**). Para escribir un nuevo script debemos acceder primero al editor de MATLAB. Para ello, basta con clicar el botón `New Script` en la barra de herramientas superior. Al guardar el script hay que tener en cuenta que el nombre que le demos no debe contener acentos, espacios en blanco y/o caracteres especiales. Para ejecutar un script que esté en el directorio de trabajo (desde MATLAB se puede navegar fácilmente por el árbol de directorios del equipo) basta escribir su nombre en la línea de comandos o pulsar el botón `Run`.

1.2.6. Ayuda

Algunos de los comandos más útiles de MATLAB son los que permiten obtener ayuda sobre otros comandos/funciones: `help`, `doc` y `lookfor`. La documentación online de MATLAB (accesible mediante `doc`) es realmente buena.

```
help plot
doc plot
lookfor plot
```

1.3. MATLAB como calculadora

Para utilizar MATLAB como una simple calculadora basta con escribir la expresión aritmética que queramos evaluar y pulsar `Intro`. El resultado de la operación queda asignado a una variable del sistema que se denomina **ans** (última variable guardada en memoria). Como ya se ha mencionado, si se quiere evitar que MATLAB muestre por pantalla el resultado de la operación bastará con escribir ";" al final de la expresión.

Algunas funciones matemáticas elementales son las siguientes:

`sqrt(x)`	raiz cuadrada	`sin(x)`	seno (radianes)
`abs(x)`	módulo	`cos(x)`	coseno (radianes)
`conj(z)`	complejo conjugado	`tan(z)`	tangente (radianes)
`real(z)`	parte real	`cotg(x)`	cotangente (radianes)
`imag(z)`	parte imaginaria	`asin(x)`	arcoseno
`exp(x)`	exponencial	`acos(x)`	arcocoseno
`log(x)`	logaritmo natural	`atan(x)`	arcotangente
`log10(x)`	logaritmo decimal	`cosh(x)`	cos. hiperbólico
`rat(x)`	aprox. racional	`sinh(x)`	seno hiperbólico
`mod(x,y)` `rem(x,y)`	resto de dividir x por y Iguales si $x, y > 0$. Ver help para definición exacta	`tanh(x)`	tangente hiperbólica
`fix(x)`	Redondeo hacia 0	`acosh(x)`	arcocoseno hiperb.
`ceil(x)`	Redondeo hacia $+\infty$	`asinh(x)`	arcoseno hiperb.
`floor(x)`	Redondeo hacia $-\infty$	`atanh(x)`	arcotangente hiperb.
`round(x)`	Redondeo al entero más próximo		

1.3.1. Formatos

Podemos cambiar la forma en la que MATLAB muestra por pantalla los números mediante el comando `format`.

```
format short, 1/3   % 4 decimales
format long, 1/3   % 15 decimales
format rat, 1/3   % fraccion
```

Sin embargo, hay que tener en cuenta que el cambio de formato sólo tiene efecto a nivel de visualización (el número con el que MATLAB trabaja en memoria seguirá siendo siempre el mismo).

1.4. Vectores y matrices

Se puede definir un vector fila de las siguiente formas:

- Introduciendo entre corchetes sus componentes, separadas por un espacio o una coma:

```
a = [1 4 9]
a = [1, 4, 9]
```

- Usando el símbolo ":", que permite crear secuencias regulares:

```
a = 1:2:10
```

- Usando el comando `linspace(a,b,n)`, que genera un vector fila de n componentes cuyo primer elemento es a y el último b, siendo todos sus elementos equidistantes:

```
a = linspace(-10, 15, 7)
```

Si lo que queremos es un vector columna podemos hacer lo siguiente:

- Separar las distintas componentes con ";"

```
b = [-1; 2; 3]
```

- Transponer un vector fila existente. Para ello se usa la comilla simple.

```
b = [1 2 3]'
```

Se pueden sumar y multiplicar todas las componentes de un mismo vector mediante los comandos `sum` y `prod`, respectivamente:

```
c = [-5 1 3];  % defino el vector c
sum(c)  % -5+1+3
prod(c)  % -5*1*3
```

Para crear una matriz combinamos la definición de vector fila y vector columna. Por ejemplo:

```
M = [1 2 3; 4 5 6]  % matriz de 2 filas y 3 columnas
N = [1 2 1; 2 4 3; 3 5 2]  % matriz de 3 filas y 3 columnas
```

Se pueden seleccionar elementos de un vector indicando, entre paréntesis, la posición (posiciones) que ocupa(n) la(s) componente(s) que no(s) interesa(n).

```
a = [-9 4 3 -7]; % defino el vector a
```

```
a(2)  % segunda componente del vector a
a([2 4])  % segunda y cuarta componentes del vector a
a(2) = pi  % modifico la segunda componente del vector a
```

En el caso de matrices, habría que indicar la posición de la fila y la columna que nos interese.

```
M = [1 2 3; 4 5 6]  % defino la matriz M
M(2, 2)  % elemento de la segunda fila y segunda columna
```

Se puede seleccionar el contenido de una fila (o una columna) entera mediante el símbolo ":".

```
M(1, :)  % primera fila de M
M(:, 1)  % primera columna de M
M(:, [1 3])  % primera y tercera columnas de M
```

Ejemplo 1.1

Crea la matriz $A = \begin{pmatrix} 1 & 3 & 2 & -1 \\ 0 & 0 & 2 & 5 \\ -2 & 0 & 0 & 3 \\ 1 & 1 & 1 & -1 \end{pmatrix}$ en MATLAB y haz las siguientes operaciones:

- Sustituye el elemento $a_{2,3}$ por un -3 y el $a_{1,2}$ por un 7, y llama B a la matriz resultante

```
A = [1 3 2 -1; 0 0 2 5; -2 0 0 3; 1 1 1 -1];   % ...
   defino la matriz A
A(2,3) = -3; A(1,2) = 7; B = A
```

- Extrae en un vector la primera fila de B, y en otro vector la segunda columna de B

```
B(1, :)   % primera fila de B
B(:, 2)   % segunda columna de B
```

- Extrae en una matriz C las filas 1 y 3 de B y en otra matriz D las columnas 2 y 4

```
C = B([1 3], :)   % filas 1 y 3 de B
D = B(:, [2 4])   % columnas 2 y 4 de B
```

- Crea una nueva matriz E de tamaño 4×8 que contenga en las columnas 1-4 D, y en las columnas 5-8 la matriz identidad de orden 4 (comando eye)

```
id = eye(4);  % matriz identidad de orden 4
E = [D id]
```

- Crea una nueva matriz F de tamaño 6×4 que contenga en las filas 1-4 D, y en las filas 5-6 una matriz de ceros de tamaño 2×4 (comando `zeros`)

```
ceros = zeros(2, 4);  % matriz 2x4 de ceros
F = [D; ceros]
```

El comando `length` nos devuelve la longitud de un vector. Para el caso de matrices, `size` nos devuelve su tamaño.

```
length(a)
size(M)
```

Si `M1` y `M2` son matrices y `r` es un escalar, la forma de indicar a MATLAB que realice las operaciones algebraicas habituales es mediante los operadores ya vistos de suma (+), producto (*) y exponenciación ($\hat{}$). Para poder realizar estos cálculos únicamente es necesario que los vectores y matrices tengan la dimensión adecuada para que la operación pueda realizarse.

```
M1*M2 + r*M1^2
```

Si quisiéramos realizar estas operaciones algebraicas "elemento a elemento" los operadores que debemos utilizar serían ".*", "./" y ".$\hat{}$"

```
[2 3] .* [2 4]  % multiplicacion termino a termino
[2 3] ./ [2 4]  % division termino a termino
[2 3] .^ 2  % potenciacion termino a termino
```

1.5. Ejercicios

1. Calcula:
   a) La suma de los primeros 100 números naturales.
   b) La suma de los cubos de los primeros 100 números naturales.
   c) La suma de todos los números pares de 3 cifras y la suma de sus cuadrados.
   d) El producto de las raíces cuadradas de los primeros 50 números naturales.
2. Realiza los siguientes cálculos:

   a) $\left(\sqrt{6}-3\right)\left(3+2^{1/2}\cdot 3^{0,5}\right)$
   b) $\cos(3\pi)+12\,\text{sen}\,\dfrac{3\pi}{2}$
   c) $e^{4-8^{2/3}}$
   d) $e^{3\pi}-\sqrt{2+3\cdot\tan\dfrac{\pi}{2}}$
   e) $\dfrac{1-13e^{2}}{1-\cos 2\pi}$
   f) $\dfrac{1}{3\ln 1}\,\text{sen}\,4\pi$

3. La velocidad v de un paracaidista que cae está dada por

$$v=\frac{gm}{c}\left(1-e^{(-c/m)t}\right)$$

   donde $g=9{,}8m/s^2$. Para un paracaidista con coeficiente de arrastre de $c=15kg/s$, calcular la velocidad en $t=9s$, si $m=60kg$.

2 Números complejos

2.1. Manejo básico de números complejos

MATLAB devolverá como resultado un número complejo si le pedimos que calcule la raíz cuadrada de un número negativo:

```
sqrt(-4)
```

Nota: Fíjate que sólo nos indica una de las dos raíces cuadradas.

La unidad imaginaria en MATLAB se representa mediante la variable `i` (o `j`). Por tanto, para definir el número complejo `2+3i` bastaría con hacer lo siguiente:

```
% todas estas definiciones son equivalentes
z = 2+3i
z = 2+3*i
z = 2+3j
```

También podemos trabajar con números complejos expresados en forma exponencial:

```
z = 2*exp(i*pi/6)   % forma exponencial
```

En cualquier caso, podemos llevar a cabo las operaciones básicas (suma, producto, potenciación) mediante los operadores habituales:

```
z1 = 5-8j
z2 = exp(i*pi/4)
z1 + z2   % suma
z1 - z2   % resta
z1*z2   % producto
z1^2   % potenciacion
```

El comando `real` devuelve la parte real del número complejo, y el comando `imag` la parte imaginaria:

```
z = -6-4i
real(z)  % parte real de z
imag(z)  % parte imaginaria de z
```

Por su parte, conj devuelve el conjugado:

```
z = -3-9i
conj(z)  % conjugado de z
```

El comando abs permite obtener el valor absoluto:

```
z = -5-2i
abs(z)  % valor absoluto de z
```

Ejemplo 2.1

omprueba, para $3 + 4i$, que el producto de un número complejo por su correspondiente conjugado es igual al cuadrado del módulo:

```
z = 3+4*i;
z_conj = conj(z);
z*z_conj - abs(z)^2  % comprobacion
```

El comando angle devuelve el argumento, expresado en radianes:

```
z = pi+i
angle(z)  % argumento de z
```

Para pasar de radianes a grados se puede usar el comando rad2deg:

```
z = exp(1)-7i
angle(z)  % argumento de z, en radianes
rad2deg(angle(z))  % argumento de z, en grados
```

Todos los comandos vistos hasta el momento operan no sólo a nivel de un único número complejo, sino también sobre vectores (o matrices)

Ejemplo 2.2

btén la parte real, la imaginaria, el conjugado, el módulo y el argumento de los siguientes números complejos: $2 + i$, $\frac{1}{3} - 7i$ y $-6 + 4i$

```
z1 = 2+i;
z2 = (1/3)-7i;
z3 = -6+4*j;

real([z1 z2 z3])  % parte reale
imag([z1 z2 z3])  % parte imaginaria
conj([z1 z2 z3])  % conjugado
abs([z1 z2 z3])  % modulo
angle([z1 z2 z3])  % argumento
```

2.2. Graficado de números complejos

El comando `compass` nos permite representar gráficamente un número complejo (o un vector de números complejos) en el plano complejo por medio de una flecha que tiene su origen en el punto $(0, 0)$:

```
compass(3+2j)
compass(3, 2)  % definicion equivalente a la anterior
compass([3+2j i])
```

Se puede cambiar el color de la flecha fácilmente:

```
compass(3+2j, 'r')  % rojo
compass(3+2j, 'g')  % verde
```

El comando `hold on` permite seguir dibujando en la última ventana gráfica que esté activa:

```
compass(5-i, 'b')  % azul
hold on
compass(3+2j, 'c')  % cyan
hold on
compass(2j, 'm')  % magenta
```

Para desactivar el efecto de `hold on` se utiliza el comando `hold off`.

Nota: El comando `close` cierra la última ventana gráfica que esté activa. `close all` cierran todas las ventanas gráficas.

2.3. Ejercicios

1. Dados $z = -1 + 2i$, $w = \sqrt{2} + 3i$ y $s = -i$, escribe en forma binómica los números complejos resutantes de las siguientes operaciones:
   - $c_1 = \frac{1}{2}z - \frac{3}{5}\overline{w}Im(s) + szw$
   - $c_2 = 2zw - s^{1023}z$
   - $c_3 = s^{1023} + s^{1024} + s^{1025} + s^{1026}$
   - $c_4 = \overline{zw} - \overline{z}\ \overline{w}$
   - $c_5 = \overline{z + w} - \overline{z} - \overline{w}$
2. Dados $z_1 = 2\sqrt{3} + 2i$ y $z_2 = 2e^{i\frac{\pi}{3}}$:
   a) Expresa z_1 en forma exponencial y comprueba que el valor absoluto y el argumento son los mismos que en la forma algebraica.
   b) Calcula el producto $z_1 z_2$ y expresa el resultado tanto en forma binómica como en forma exponencial.
   c) Representa en un mismo gráfico z_1 (en azul) y z_2 (en rojo) sobre el plano complejo.

3 Funciones de una variable

3.1. Funciones

Una **función** es un programa con una *interfaz* de comunicación con el exterior mediante argumentos de entrada y de salida. Las funciones deben guardarse en un fichero con el mismo nombre que la propia función y extensión **.m**. Las funciones en MATLAB tienen la siguiente estructura:

```
function [argumentos de salida] = nombre_funcion(argumentos ...
   de entrada)
% comentarios
instrucciones
end
```

Ejemplo 3.1

Función que halla el área (A) de un círculo:

```
function [A] = area_circ(r)
    A = pi*r^2;
end
```

Para que MATLAB reconozca la función que acabamos de crear tenemos dos opciones:

1. Situarnos en la misma ruta (directorio) en la que hayamos salvado la función. Esta opción sólo será válida mientras permanezcamos en el mismo directorio.
2. Utilizar la orden addpath para incluir en memoria la ruta (directorio) en la que hayamos guardado la función.

   ```
   addpath directorio_donde_has_salvado_la_funcion
   ```

   Una vez hecho esto, podremos volver a movernos libremente por el árbol de directorios del equipo.

A partir de este momento, podremos llamar a la función desde la línea de comandos:

```
area_circ(5)  % area de un circulo de radio = 5 unidades
```

A menudo, funciones tan sencillas como la que acabamos de crear (que devuelven el resultado de una única orden) se definen directamente en la línea de comandos o incluso en mitad de un script (**funciones anónimas** o *function handle*). La forma de hacerlo es la siguiente:

```
nombre_funcion = @(argumentos de entrada) instruccion
```

Ejemplo 3.2

Función que eleva un número dado al cuadrado:

```
eleva_cuad = @(x)  x^2;  % defino funcion
class(eleva_cuad)  % funcion tipo handle
eleva_cuad(4)  % elevo 4 al cuadrado
```

A partir de este momento ya podremos definir en MATLAB funciones de una variable real.

Ejemplo 3.3

Define la función $f(x) = x^2 + 2$:

```
f = @(x)  x^2 + 2;
```

Para evaluar dicha función en un determinado x sólo tendremos que darle el valor deseado al argumento de entrada x. Por ejemplo, para evaluar f en el punto $x = -pi/2$ bastaría con hacer lo siguiente:

```
f(-pi/2)
```

Si quisiéramos evaluar f en varios puntos a la vez bastaría con utilizar como argumento de entrada un vector (en lugar de un único número). Sin embargo, fíjate que para poder operar con vectores, debemos modificar ligeramente la función f antes:

```
f_vec = @(x)  x.^2 + 2;  % redefino la funcion para que ...
   opere con vectores
f_vec(-10:10)  % evaluo la funcion en {-10,-9,-8,...,8,9,10}
```

3.2. Gráficos

Llegados a este punto podemos ya *plotear* (representar gráficamente) funciones de una variable real. La función básica de dibujo en MATLAB es `plot`, que permite representar pares de puntos (x, y).

```
x = -10:10;
y = x;
plot(x, y)  % recta y=x en el intervalo x=[-10,10]
```

Si le pasamos un único argumento de entrada, la función `plot` entiende que nos referimos al eje de ordenadas, y asigna valores naturales (comenzando en 1) al eje de abscisas.

```
plot(y)
```

Nota: Recuerda que `close all` cierra todas las ventanas gráficas que estén activas. `figure` abre una nueva ventana gráfica. `figure(x)` abre una nueva ventana gráfica numerada con el índice "x".

Ejemplo 3.4

Representa en rojo la parábola $f(x) = x^2 - 50$ en el intervalo $[-20, 15]$. Etiqueta el eje de abscisas como "x", el de ordenadas como "y" y pon un título al gráfico. Dibuja una cuadrícula en el fondo.

```
f = @(x)  x.^2 -50;
plot(-20:15, f(-20:15), 'r')
xlabel('x')  % etiqueta eje x
ylabel('y')  % etiqueta eje y
title('Parabola')  % titulo figura
grid on  % cuadricula en el fondo
```

Como ye hemos visto, `hold on` se utiliza para seguir dibujando en la última ventana gráfica que esté activa. Esto nos permitirá superponer la gráfica de dos o más funciones en una única figura. Con el comando `legend` podremos además añadir una leyenda, que siempre resulta muy útil.

Ejemplo 3.5

Representa en azul la función $f(x) = \frac{x^2}{4}$ y en rojo la función $g(x) = \sqrt{x^2}$ en el intervalo $[-10, 10]$. Dibuja una leyenda que identifique apropiadamente ambas funciones. Etiqueta el eje de abscisas como "eje x", el de ordenadas como "eje y" y pon un título al gráfico. Por último, dibuja una cuadrícula en el fondo.

```
f = @(x)  x.^2/4;  % defino f
g = @(x)  sqrt(x.^2);  % defino g

vec_abscisas = -10:10;  % defino dominio

plot(vec_abscisas, f(vec_abscisas), 'b')
```

```
hold on  % permanezco en la misma ventana grafica
plot(vec_abscisas, g(vec_abscisas), 'r')

legend('y=x^2/4', 'y=sqrt(x^2)', 'Location', ...
   'northeast')  % defino leyenda

xlabel('eje x'), ylabel('eje y'), title('Dos funciones')
grid on  % rejilla de fondo
```

3.3. Variables (y funciones) simbólicas

MATLAB permite trabajar con variables simbólicas, es decir, variables genéricas que, en principio, no tienen porqué tomar un valor concreto (piensa en las incógnitas de cualquier ecuación)

```
syms x  % defino la variable x como simbolica
eq = x^2 + 5;  % ecuacion que depende de x
```

La forma de obligar a que una variable simbólica tome un determinado valor es la siguiente:

```
subs(eq, x, 3)  % forzamos a que x tome el valor 3 en la ...
   expresion 'eq'
```

Se pueden definir varias variables simbólicas a la vez, así como el tipo exacto de variable con la que queremos trabajar:

```
syms x y real  % forzamos a que x e y solo puedan tomar ...
   valores reales
```

De este modo resulta muy fácil definir, en modo simbólico, funciones de una variable real en MATLAB. Para representar gráficamente este tipo de funciones se utiliza el comando `ezplot`.

Ejemplo 3.6

Representa la función $f(x) = sen(x) + 3x + \frac{8}{x+1}$ en el intervalo $[-4, 4]$

```
syms x real  % defino 'x' como variable simbolica real
y = sin(x) + 3*x + 8/(x+1)  % defino f(x)
class(y)  % y es una variable simbolica
ezplot(y, [-4, 4])  % represento f(x) en el intervalo ...
   [-4, 4]
```

```
figure
ezplot('sin(x) + 3*x + 8/(x+1)', [-4, 4])  % equivalente ...
   a la sentencia anterior

figure
ezplot('sin(x) + 3*x + 8/(x+1)', [-4, 4, -20, 20])  % ...
   modifico el area mostrada: [-4,4]x[-20,20]

syms x y real  % defino 'x' e 'y' como variables ...
   simbolicas reales
figure
ezplot('y = sin(x) + 3*x + 8/(x+1)', [-4, 4, -20, 20])  ...
   % se pueden meter variable y dependiente en la ...
   definicion de la funcion a representar
```

Para modificar un gráfico creado con `ezplot` necesitamos la función `set`:

Ejemplo 3.7

Vuelve a representar la función $f(x) = sen(x) + 3x + \frac{8}{x+1}$ en el intervalo $[-4, 4]x[-30, 20]$, pero esta vez en rojo y con línea discontinua más gruesa.

```
syms x real
fig = ezplot('sin(x) + 3*x + 8/(x+1)', [-4, 4, -30, 20]) ...
    % guardo el grafico generado en una nueva variable
set(fig, 'Color', 'r', 'LineStyle', '--', 'LineWidth', 2)
```

Nota: Consulta la documentación de `set` para un mayor detalle: https://es.mathworks.com/help/matlab/ref/set.html.

3.4. Ejercicios

1. Define las siguientes funciones de una variable y halla el valor en el punto $x = \pi$.

   a) $f(x) = \frac{x}{1+|x|}$

   b) $f(x) = \frac{e^x+e^{-x}}{2}$

   c) $f(x) = \text{senh}(x)$

   d) $g_1(x) = f_1(f_2(x))$ y $g_2(x) = f_2(f_1(x))$, con $f_1(x) = x^2 + 1$ y $f_2(x) = \frac{2}{x}$

2. Halla, gráficamente, el punto en el que se cortan las funciones $f(x) = \frac{2}{x}$ y $g(x) = \sqrt{x^2 - 3}$. Para ello, representa en una única figura f (en negro) y g (en rojo). Pon una leyenda que identifique cada curva y un título a la figura.

3. Dibuja, en una misma gráfica y en color verde, las circunferencias $x^2 + y^2 - 4x = 1$ (con línea continua) y $x^2 + y^2 + 2y = 9$ (con línea discontinua). Pon una leyenda adecuada. Representa los puntos de corte entre ambas circunferencias con un asterisco azul.

4. A partir de los gráficos necesarios, estudia el dominio, imagen y el carácter par/impar de las siguientes funciones:

   a) $cos(x^2)$

   b) $cos^2(x)$

   c) $cos^2(x^2)$

   d) $\sqrt{x^3}$

4 Límites y derivación

El cálculo simbólico en MATLAB es muy potente. Entre otras operaciones, podremos calcular límites y derivadas muy fácilmente.

4.1. Límites

El comando `limit` permite calcular directamente el límite de una función, ya sea esta anónima (del tipo *function handle*) o simbólica.

Ejemplo 4.1

Calcula los siguientes límites:

- $\lim_{x\to\infty} \frac{x^4-x^3+5}{3x^4+x^2-4x}$

```
syms x real  % defino x como variable simbolica real
f = @(x)  (x^4 - x^3 + 5) / (3*x^4 + x^2 - 4*x);  % ...
   funcion anonima
class(f)
limit(f, x, inf)

f = (x^4 - x^3 + 5) / (3*x^4 + x^2 - 4*x);  % ...
   funcion simbolica
class(f)
limit(f, x, inf)
```

- $\lim_{x\to 0} \frac{(1-cosx)^2}{2sen^4x-sen^5x}$

```
f = @(x)  (1-cos(x))^2 / (2*sin(x)^4 - sin(x)^5);
limit(f, x, 0)
```

- $\lim_{x\to 0} \frac{sen|x|}{x}$

```
f = sin(abs(x)) / x;
limit(f, x, 0)  % este limite no existe
```

```
limit(f, x, 0, 'left')  % limite por la izquierda ...
   --> -1
limit(f, x, 0, 'right')  % limite por la derecha --> 1
```

Fíjate en que el resultado que devuelve `limit` es una expresión simbólica:

```
L = limit(x^4 / (2*sin(x)), x, pi/2)
class(L)
```

Para pasar a formato decimal podemos recurrir al comando `double`:

```
double(L)
```

A partir del cálculo de límites podemos estudiar la continuidad de una función. Para ello será conveniente conocer el comando `solve`, que permite resolver sistemas de ecuaciones. En el caso de una sola ecuación con una única incógnita, su sintaxis es la siguiente:

```
syms x real
solve(x^2 - 5*x - 6 == 0)  % raices ecuacion segundo grado ...
   --> x = {-1,6}
solve(x^2 - 5*x - 6)  % equivalente a la orden anterior
solve(x^2 == x + 2)  % la igualdad en una expresion ...
   simbolica se indica con el simbolo '==' --> x = {-1,2}
```

`solve` está diseñado para trabajar con expresiones simbólicas. Por tanto, en el caso de funciones anónimas, habría que convertirlas antes al tipo simbólico con ayuda del comando `sym`:

```
syms x real
f = @(x)  4*x^2 - x;
solve(f)  % no permitido
solve(sym(f))  % permitido --> x = {0, 1/4}
```

El resultado devuelto por `solve` también es una expresión simbólica:

```
syms x real
raices = solve(x^2 == x + 2)
class(raices)  % expresion simbolica
double(raices)  % paso a decimal
```

Ejemplo 4.2

Estudia la continuidad de la función $f(x) = \frac{x}{x^2-x}$

```
syms x real
f = @(x)  x / (x^2 - x);
raices = double(solve(x^2 - x));  % f no esta definida ...
   en los puntos en los que el denominador se anule

limit(f, x, raices(1))  % discontinuidad evitable en x = ...
   0 (existe limite)
limit(f, x, raices(2))  % discontinuidad no evitable en ...
   x = 1 (no existe limite)
```

4.2. Derivación

El cálculo de derivadas se lleva a cabo con la función `diff`. Al igual que `limit`, `diff` acepta tanto funciones anónimas como simbólicas:

Ejemplo 4.3

Calcula $f'(x)$ y $f''(x)$, siendo $f(x) = \frac{x^4-x^3+5}{3x^4+x^2-4x}$:

```
syms x real

f = @(x)  x^4 - x^3 + 5 / 3*x^4 + x^2 - 4*x;  % funcion ...
   anonima
f_prima = diff(f, x, 1)  % derivada de orden 1
f_prima2 = diff(f, x, 2)  % derivada de orden 2

f = x^4 - x^3 + 5 / 3*x^4 + x^2 - 4*x;  % funcion simbolica
f_prima = diff(f, x)  % derivada de orden 1
f_prima2 = diff(f, x, 2)  % derivada de orden 2
```

En ambos casos, el resultado devuelto por `diff` es una expresión simbólica. Sin embargo, como ya sabemos, el comando `subs` permite evaluar una expresión simbólica en un valor concreto (o valores concretos).

Ejemplo 4.4

Siendo $f(x) = \frac{sin(x)^4}{x^2+exp(x)}$, calcula $f'(\pi)$ y $f''(-4)$:

```
syms x real

f = @(x)  sin(x)^4 / (x^2 + exp(x));  % funcion anonima
f_prima = diff(f, x, 1);  % derivada de orden 1
f_prima2 = diff(f, x, 2);  % derivada de orden 2

f_prima_pi = double(subs(f_prima, pi))  % f'(x=pi)
f_prima2_4 = double(subs(f_prima2, -4))  % f''(x=-4)
```

Con lo que hemos visto ahora podríamos calcular (y dibujar) las rectas tangentes y normales a una curva dada:

Ejemplo 4.5

Calcula las tangentes a $f(x) = x^3 - 3x$ que sean paralelas a la recta $y = 6x + 10$. Representa en una misma figura $f(x)$ (en rojo) y las tangentes halladas (en negro y con línea discontinua) en el dominio $[-5, 5]$. Pon un título y una leyenda adecuada.

```
syms x real
f = x^3 - 3*x;  % funcion simbolica
f_prima = diff(f, x)  % derivada de orden 1
a = double(solve(f_prima == 6))  % para ser paralela a y ...
   = 6x + 10, la pendiente tiene que ser 6 --> a = ...
   abscisas para las que habra que calcular las rectas ...
   tangentes

rt1 = subs(f_prima, a(1))*(x-a(1)) + subs(f, a(1))  % ...
   eq. recta tangente en x = a(1)
rt2 = subs(f_prima, a(2))*(x-a(2)) + subs(f, a(2))  % ...
   eq. recta tangente en x = a(1)

fig_f = ezplot(f);  % dibujo f
hold on
fig_rt1 = ezplot(rt1);  % dibujo recta tangente en x = a(1)
hold on
fig_rt2 = ezplot(rt2, [-5, 5]);  % dibujo recta tangente ...
   en x = a(2)
```

```
set(fig_f, 'Color', 'r')
set(fig_rt1, 'Color', 'k', 'LineStyle', '--')
set(fig_rt2, 'Color', 'k', 'LineStyle', '--')

legend('f(x)', 'RT1', 'RT2')
xlabel('eje x'), ylabel('eje y')
title('Rectas tangentes')
grid on
```

Se puede demostrar que si una función viene definida en forma implícita tal que $F(x, y) = 0$ (siendo $y = y(x)$), entonces $y' = -\frac{F'_x}{F'_y}$:

Ejemplo 4.6

Dada la expresión $yx^2 + y^3 = 1$, calcula $y'(0)$ en un entorno del punto $(0, 1)$

```
syms x y real  % necesito definir como simbolicas tanto ...
   'x' como 'y'
F = y*x^2 + y^3 - 1  % funcion implicita F(x,y) = 0
F_prima = -diff(F, x) / diff(F, y)  % F'
double(subs(F_prima, [x y], [0 1]))  % 'subs' permite ...
   sustituir mas de un valor a la vez
```

4.3. Ejercicios

1. Utiliza la expresión $\frac{f(x+\Delta x)-f(x)}{\Delta x}$ para calcular un valor aproximado de $\frac{d}{dx}\left(\frac{log(1+x^2)}{sin^2 x}\right)$ en $x_0 = 1$ tomando $\Delta x = \{0{,}1; 0{,}05; 0{,}0025\}$. A continuación, calcula el valor exacto de la derivada usando los comandos `limit` y `diff`. Compara las aproximaciones obtenidas con el valor exacto en términos de error porcentual $\left(100|\frac{valor_{aprox}-valor_{exacto}}{valor_{exacto}}|\right)$.

2. Halla el punto P donde se cortan las funciones $f(x) = \frac{2}{x}$ y $g(x) = \sqrt{x^2-3}$. Representa en una misma figura f (en rojo) y g (en negro) en $[0, 5] \times [0, 3]$. Pon una leyenda que identifique cada curva y un título. Halla la ecuación de la tangente a cada curva en P, dibuja ambas rectas (en el mismo color que la curva correspondiente pero en línea discontinua) y comprueba que son perpendiculares.

3. Representa la gráfica de la circunferencia $x^2 + y^2 + 2y = 9$ en color negro y con una línea gruesa. Halla todas las rectas tangentes y normales a la circunferencia en el punto $x_0 = 2$ y represéntalas en línea discontinua en el dominio $[-6, 6] \times [-8, 8]$. Pon un título.

5 Polinomios de Taylor

5.1. Obtención del polinomio de Taylor

En la práctica, los polinomios de Taylor se utilizan para calcular el valor aproximado de una función en el entorno de un punto dado:

Definición 5.1

Sea $f(x)$ una función derivable n veces en el punto $x = a$. Se define el **polinomio de Taylor** de grado n correspondiente a la función f en $x = a$ como:

$$P_n(f,a) = f(a) + f'(a)(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \frac{f'''(a)}{3!}(x-a)^3 + \ldots + \frac{f^{(n)}(a)}{n!}(x-a)^n$$

Si $a = 0$, el polinomio se llama **de Maclaurin**.

En base a esta definición, y con lo que hemos visto hasta ahora, ya podríamos calcular polinomios de Taylor:

Ejemplo 5.2

Calcula el polinomio de Taylor de grado 3 de la función $f(x) = xsen(x+1)$ en el punto $x = 0$:

```
syms x real
f = x*sin(x+1);  % funcion simbolica
fprima1 = diff(f, x, 1);  % primera derivada de f
fprima2 = diff(f, x, 2);  % segunda derivada de f
fprima3 = diff(f, x, 3);  % tercera derivada de f

a = 0;  % punto de aplicacion del desarrollo
pt3 = subs(f, x, a) + subs(fprima1, x, a)*(x-a) + ...
   (subs(fprima2, x, a)/factorial(2))*(x-a)^2 + ...
   (subs(fprima3, x, a)/factorial(3))*(x-a)^3  % ...
   polinomio de Taylor de grado 3

fig_f = ezplot(f, [-1 1]);  set(fig_f, 'Color', 'k')  % ...
   dibujo f en un entorno de x=0
```

```
hold on
fig_pt3 = ezplot(pt3, [-1 1]);  set(fig_pt3, 'Color', ...
   'r', 'LineStyle', '--')  % dibujo tambien el ...
   polinomio de Taylor de grado 3
grid on
xlabel('x'), ylabel('y')
legend('f', 'P_3(f, 0)')
title('funcion y polinomio de Taylor')
```

Sin embargo, el comando `taylor` de MATLAB permite calcular directamente el polinomio de Tayor (de cualquier grado) para una cierta función (en cualquier punto). Dicha función puede estar definida como anónima (*function handle*) o como simbólica. El resultado que devuelve `taylor` es siempre una expresión simbólica.

Ejemplo 5.3

Utiliza el comando `taylor` para calcular el polinomio de Taylor de grado 3 de la función $f(x) = xsen(x+1)$ en el punto $x = 0$. A continuación, utiliza dicho polinomio para obtener un valor aproximado de $0{,}15 \cdot sen(1{,}15)$. ¿Qúe error porcentual se comete en la aproximación?

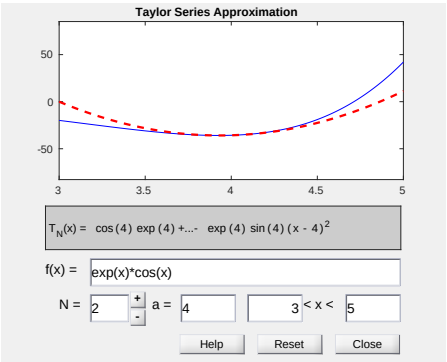
```
syms x real
f = @(x)  x*sin(x+1);  % funcion anonima
class(f)
pt3 = taylor(f, x, 'ExpansionPoint', 0, 'Order', 4)  % ...
   polinomio de Taylor de orden 3 de f en el punto 0
class(pt3)  % funcion simbolica

f = x*sin(x+1);  % funcion simbolica
class(f)
pt3 = taylor(f, x, 'ExpansionPoint', 0, 'Order', 4)  % ...
   polinomio de Taylor de orden 3 de f en el punto 0
class(pt3)  % funcion simbolica

ve = 0.15*sin(1.15)  % valor exacto: 0.1369
va = double(subs(pt3, x, 0.15))  % valor aproximado ...
   (necesitaremos que x=0.15): 0.1370
err = 100*abs((va-ve)/ve)  % error: 0.03%
```

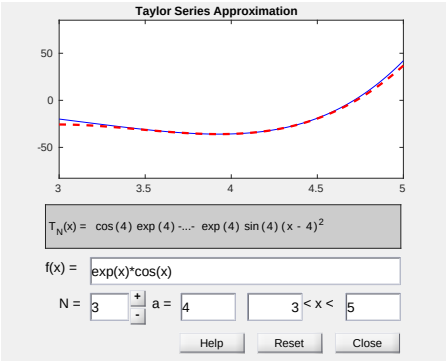
Nota: Fíjate en que los argumentos de entrada `ExpansionPoint` y `Order` son opcionales. Si estos argumentos no se especifican, `taylor` calcula por defecto el polinomio de Taylor de grado 5 centrado en el punto $x = 0$. Consulta la documentación (`doc taylor`) si quieres obtener más información.

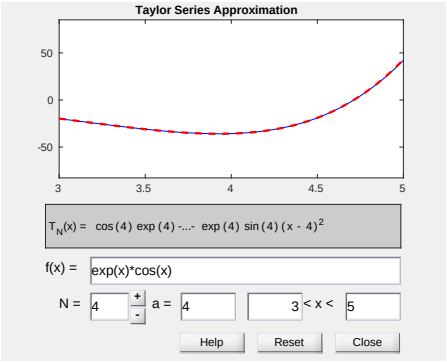
Además del comando `taylor`, MATLAB cuenta con una herramienta gráfica para el cálculo (y visualización) de polinomios de Taylor. Para activarla, bastará con teclear `taylortool` en la ventana de comandos. Esto abrirá una ventana gráfica en la que se puede introducir por pantalla la función a aproximar, el grado que se desee para el polinomio de Taylor y el intervalo en el que se quieren representar la función y el polinomio aproximador.

Ejemplo 5.4

Comprueba, para la función $f(x) = e^x cosx$, en un entorno del punto $x = 4$, que a medida que el grado del polinomio de Taylor crece, éste aproxima mejor a f y sobre un intervalo mayor. Utiliza para ello la herramienta `taylortool`:







5.1.1. El bucle for

Llegados a este punto, conviene conocer la estructura de control más utilizada en programación, el bucle `for`, que nos permitirá automatizar la ejecución repetida de una cierta secuencia de órdenes. La estructura de un bucle (o *loop*) en MATLAB es la siguiente:

```
for k=n1:n2
ordenes a ejecutar
end
```

Donde k es el contador que toma valores desde $n1$ hasta $n2$ (en este caso con incremento unidad). La última iteración del bucle se produce cuando k llega al último valor de la secuencia a recorrer (en este caso $n2$).

Ejemplo 5.5

Diseña un bucle que muestre por pantalla el inverso de todos los número pares entre el 1 y el 20:

```
for i = 2:2:20    % valores a recorrer en el bucle
    disp(1/i)  % 'disp' muestra por pantalla
end
```

Ejemplo 5.6

Diseña un bucle que guarde en un vector la raíz cuadrada de los 20 primeros números naturales (excluyendo el 0):

```
vec = [];  % defino vector vacio, que se ira rellenando ...
   en el bucle
for i = 1:20  % valores a recorrer en el bucle
    vec(i) = sqrt(i)  % voy rellenando el vector
end
```

5.2. Ejercicios

1. Considera la función $f(x) = x^4 - 5x^3 + 5x^2 + x + 2$

   a) Calcula el polinomio de Taylor de grado 1 de f en $x = 2$. Comprueba que se trata de la recta tangente a f en dicho punto.

   b) Calcula los polinomios de Taylor de grado 2, 3, 4, 5 y 10. ¿Qué ocurre a partir de grado 4?

2. Considera la función $f(x) = log(\frac{x+2}{2x-2})$

   a) Dibuja en una misma gráfica f y sus polinomios de Taylor de orden 1, 2 y 3 en torno al punto $x = 4$.

   b) ¿Cuál es el grado del polinomio de Taylor en $x = 4$ que habría que considerar para llegar a aproximar $log(\frac{6,1}{6,2})$ con un error porcentual inferior al 0,05 %?

3. Considera la función $f(x) = \sqrt{1 - \frac{x}{2}}$

   a) Obtén el polinomio de Mclaurin de orden 3 para f y utilízalo para dar un valor aproximado de $\sqrt{0,5}$. ¿Qué error porcentual se comete?

   b) Diseña un bucle que guarde en un vector el error porcentual cometido en la aproximación del apartado anterior para polinomios de Taylor de grado 3, 5, 8, 10 y 12. Dibuja ese error en función del grado del polinomio aproximador. ¿Qué conclusión sacas?

   c) Repite el experimento propuesto en el apartado anterior pero con el objetivo de encontrar valores aproximados para $\sqrt{0,75}$. ¿Cómo influye el hecho de acercarse al punto en el que se hace el desarrollo ($x = 0$ en este caso)?

6 Series

6.1. Sumas (parciales y totales) de una serie

Hemos visto que, salvo para el caso de series geométricas y telescópicas, el cálculo de la suma de cualquier otro tipo de serie suele resultar una tarea poco menos que imposible. En cambio, con MATLAB resulta muy fácil calcular tanto cualquier suma parcial de una serie como su suma total. Para ello se utiliza el comando `symsum`, que opera sobre expresiones simbólicas.

Ejemplo 6.1

Considera la serie $\sum_{n=1}^{\infty} \frac{1}{n}$.

1. Calcula su suma parcial 56-ésima:

```
syms n integer  % defino 'n' como variable simbolica ...
   (entera)
a_n = 1/n;  % termino general de la serie

S_56 = symsum(a_n, n, 1, 56)  % suma parcial 56-esima
S_56 = symsum(a_n, 1, 56)  % suma parcial 56-esima
class(S_56)  % expresion simbolica
double(S_56)  % paso a decimal
```

2. Calcula su suma total:

```
symsum(a_n, n, 1, inf)  % suma total (serie divergente)
```

Como ya sabíamos, la serie $\sum_{n=1}^{\infty} \frac{1}{n}$ es divergente. Otra forma de verlo sería a partir de la expresión (simbólica) de su suma parcial enésima:

```
syms n integer
```

```
a_n = 1/n;

S_n = symsum(a_n, n, 1, n);  % expresion simbolica de la ...
   suma parcial enesima
S_3 = subs(S_n, n, 3)  % suma parcial 3-esima: 1/1 + 1/2 + 1/3
S = limit(S_n, n, inf)  % suma total (serie divergente)
```

Fíjate en que resulta muy fácil analizar el carácter (convergente o divergente) de cualquier serie con MATLAB.

Ejemplo 6.2

Analiza el carácter de la siguientes series:

1. $\sum_{n=1}^{\infty} \frac{1}{n(n+1)}$ (serie telescópica $\Rightarrow$ convergente)

```
syms n integer
a_n = 1/(n*(n+1));  % termino general
S = double(symsum(a_n, 1, inf))  % convergente (su ...
   suma es 1)
```

2. $\sum_{n=1}^{\infty} \frac{3n-1}{(\sqrt{2})^n}$ (criterio del cociente $\Rightarrow$ convergente)

```
syms n integer
a_n = (3*n - 1)/(sqrt(2)^n);  % termino general
S = double(symsum(a_n, 1, inf))  % convergente (su ...
   suma es 22.3137)
```

3. $\sum_{n=1}^{\infty} \frac{(2n)!}{(n!)^2}$ (criterio del cociente $\Rightarrow$ divergente)

```
syms n integer
a_n = factorial(2*n)/(factorial(n))^2;  % termino ...
   general
S = double(symsum(a_n, 1, inf))   % divergente ...
   (MATLAB devuelve NaN/Inf)
```

Otra cosa que resulta prácticamente inmediata con MATLAB es calcular el término general de una serie, a_n, conocida la expresión de su suma parcial enésima ($a_n = S_n - S_{n-1}$).

Ejemplo 6.3

Calcula el término general de la serie cuya suma parcial enésima es $S_n = \frac{n+1}{n+10}$

```
syms n integer
S_n = (n+1)/(n+10);  % suma parcial enesima
S_nmenos1 = subs(S_n, n, n-1);  % suma parcial (n-1)-esima
a_n = S_n - S_nmenos1  % termino general
class(a_n)  % expresion simbolica
```

El comando `simplify` permite simplificar (si se puede) las expresiones simbólicas que MATLAB muestra por pantalla:

```
simplify(a_n)
```

Es importante tener claro que las sucesiones $\{a_n\}$ y $\{S_n\}$ no tienen porqué tener el mismo carácter. Resulta evidente para el caso de la serie $\sum_{n=1}^{\infty} \frac{1}{n}$:

```
syms n integer
a_n = 1/n;  % termino general
limit(a_n, n, inf)  % sucesion convergente

S_n = symsum(a_n, 1, n);  % suma parcial enesima
limit(S_n, n, inf)  % sucesion divergente
```

6.1.1. Series de potencias

Definición 6.4

- La expresión $\sum_{n=0}^{\infty} a_n(x-a)^n$ responde a una **serie de potencias centrada en el punto** $x = a$, con a constante. Este tipo de series pueden interpretarse como una función de x:

$$f(x) = \sum_{n=0}^{\infty} a_n(x-a)^n$$

- El **radio de convergencia** de la serie $\sum_{n=0}^{\infty} a_n(x-a)^n$ viene dado por:

$$R = \lim_{n\to\infty} \left| \frac{a_n}{a_{n+1}} \right|$$

  Dentro del **intervalo de convergencia** $(a-R, a+R)$ la serie converge. En los extremos, podrá converger o diverger, por lo que habrá que estudiar específicamente qué sucede en los puntos $x = a+R$ y $x = a-R$.

- El dominio de la función $f(x) = \sum_{n=0}^{\infty} a_n(x-a)^n$ es el conjunto de valores x para los que la serie converge.

Basándonos en esta definición, resulta inmediato calcular en MATLAB el intervalo de convergencia para una serie de potencias centrada en el punto a. Para ello, habrá que calcular R y estudiar el carácter de la serie en los puntos $(x = a-R)$ y $(x = a+R)$.

Ejemplo 6.5

Calcula el intervalo de convergencia para la serie $\sum_{n=1}^{\infty} \frac{(-1)^{n+1}}{n5^n}(x-5)^n$.

```
syms n integer
a_n = ((-1)^(n+1)) / (n*5^n);
a_nmas1 = subs(a_n, n, n+1);
R = limit(abs(a_n/a_nmas1), n, inf)  % radio de ...
   convergencia alrededor de a=5

syms x real  % necesito una nueva variable simbolica ...
   (real): 'x'
tgc = a_n*((x-5)^n); % termino general (completo) de la ...
   serie
symsum(subs(tgc, x, 5-R), 1, inf)  % serie divergente en ...
   x=5-5
symsum(subs(tgc, x, 5+R), 1, inf)  % serie convergente ...
   en x=5+5
% El intervalo de covergencia sera (0, 10]
```

6.2. Ejercicios

1. Halla el término general y el carácter de las series cuya suma parcial enésima es:
   a) $S_n = \frac{3^{n+1}-1}{2}$, $n \in \mathbb{N}$
   b) $S_n = \frac{2^n-1}{2 \cdot 3^n}$, $n \in \mathbb{N}$
2. Analiza la convergencia y halla la suma total de las siguientes series geométricas:
   a) $\sum_{n=1}^{\infty} \frac{2^{n-1}}{4^{2n+1}}$
   b) $\sum_{n=1}^{\infty} \frac{2^{n-2}}{3^n}$

   A continuación, representa en un mismo gráfico los 50 primeros términos de las sucesiones $\{S_n\}$ (sumas parciales) y $\{a_n\}$ (término general).
3. Considera el siguiente desarrollo en serie de potencias centrado en $x = 0$:
   $f(x) = -log(1+x) = \sum_{n=1}^{\infty} \frac{(-1)^n x^n}{n}$
   a) Averigua cuál es el dominio de f (conjunto de valores x para los que la serie converge).
   b) Representa en una misma gráfica f y la suma de los 20 primeros términos de la serie en el dominio $[-1, 1]$.

7 Integración

7.1. Cálculo de integrales indefinidas

La función `int` de MATLAB permite calcular directamente integrales indefinidas en las que el integrando sea una expresión simbólica.

Ejemplo 7.1

Calcula las siguientes integrales:

1. $\int cos^2(\frac{x}{2})dx$
   Nota: Esta integral se resuelve fácilmente usando la propiedad $cos^2(x) = \frac{1+cos(2x)}{2}$

```
syms x real  % defino 'x' como variable simbolica real
f1 = cos(x/2)^2;  % funcion a integrar
F1 = int(f1)  % primitiva (sin constante de ...
   integracion)
```

2. $\int \frac{1}{1+\sqrt{x}}dx$
   Nota: Esta integral se puede resolver utilizando el cambio de variable $t = \sqrt{x}$

```
syms x real  % defino 'x' como variable simbolica real
f2 = 1/(1+sqrt(x));  % funcion a integrar
F2 = int(f2)  % primitiva (sin constante de ...
   integracion)
```

3. $\int xe^x dx$
   Nota: Esta integral se resuelve fácilmente por partes ($u = x$, $dv = e^x dx$)

```
syms x real  % defino 'x' como variable simbolica real
f3 = x*exp(x);  % funcion a integrar
F3 = int(f3)  % primitiva (sin constante de ...
   integracion)
```

Podemos comprobar fácilmente que las primitivas que hemos obtenido son correctas. Para ello, bastaría con ver que su derivada coincide con el integrando:

```
simplify(diff(F1) - f1)  % F1 es una primitiva correcta
simplify(diff(F2) - f2)  % F2 es una primitiva correcta
simplify(diff(F3) - f3)  % F3 es una primitiva correcta
```

Evidentemente, si sumamos cualquier constante de integración a las primitivas que hemos obtenido, éstas seguirán siendo igual de válidas:

```
simplify(diff(F1 + 5) - f1)  % F1+5 es una primitiva correcta
simplify(diff(F2 - pi) - f2)  % F2-pi es una primitiva correcta
simplify(diff(F3 + sqrt(2)) - f3)  % F3+sqrt(2) es una ...
   primitiva correcta
```

7.2. Cálculo de integrales definidas

El comando `int` también permite calcular integrales definidas. Para ello, sólo tendremos que especificar mediante argumentos de entrada los límites del intervalo de integración de interés.

Ejemplo 7.2

Calcula las siguientes integrales (continuación del ejemplo anterior):

1. $\int_{-2}^{\frac{\pi}{2}} cos^2(\frac{x}{2})dx$

```
f1_def = double(int(f1, -2, pi/2))  % valor ...
   (numerico) de la integral definida
```

2. $\int_0^1 \frac{1}{1+\sqrt{x}}dx$

```
f2_def = double(int(f2, 0, 1))  % valor (numerico) ...
   de la integral definida
```

3. $\int_0^e xe^x dx$

```
f3_def = double(int(f3, 0, exp(1)))  % valor ...
   (numerico) de la integral definida
```

Podríamos recurrir a la regla de Barrow para comprobar que los resultados que acabamos de obtener son correctos:

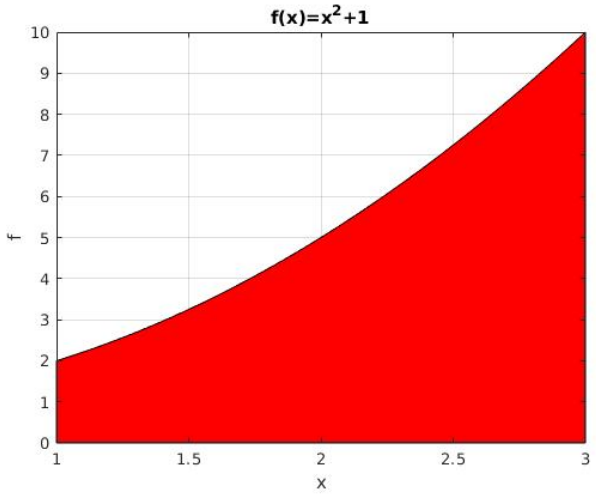
```
f1_def - (double(subs(F1, x, pi/2) - subs(F1, x, -2)))  % ...
   comprobacion: regla de Barrow
f2_def - (double(subs(F2, x, 1) - subs(F2, x, 0)))  % ...
   comprobacion: regla de Barrow
f3_def - (double(subs(F3, x, exp(1)) - subs(F3, x, 0)))  % ...
   comprobacion: regla de Barrow
```

Como sabemos, el valor de una integral definida se corresponde con el área encerrada entre la curva definida por el integrando y el eje X, confinada lateralmente por los extremos del intervalo de integración (hay que tener en cuenta que las porciones que queden por debajo del eje X serían negativas). La función `area` de MATLAB permite dibujar dicho área, por lo que resulta útil en la interpretación gráfica de los resultados obtenidos al calcular integrales definidas. Ten en cuenta que esta función sólo acepta valores numéricos (nunca expresiones simbólicas) como argumentos de entrada.

Ejemplo 7.3

Representa gráficamente la función $f(x) = x^2 + 1$ en el intervalo $[1, 3]$ y sombrea, en rojo, la región del plano cuya área viene dada por $\int_1^3 \left(x^2 + 1\right) dx$:

```
x = linspace(1, 3);  % secuencia de 100 numeros ...
   (equiespaciados) entre el 1 y el 3
y = x.^2+1;  % valor (numerico) de la funcion 'f' en ...
   esos 100 puntos
area(x, y, 'FaceColor','r')  % sombreo el area encerrada ...
   entre 'f' y el eje 'X' en rojo (por defecto se ...
   utiliza el color azul)
xlabel('x'); ylabel('f'); title('f(x)=x^2+1')
grid on
```



Cuando se trabaja con intervalos de integración centrados en torno a 0 y de igual extensión por la derecha y por la izquierda, puede resultar útil estudiar la paridad del integrando:

Ejemplo 7.4

Calcula las siguientes integrales definidas:

1. $\int_{\frac{-\pi}{2}}^{\frac{\pi}{2}} sen^2(x)cos(x)dx$

```
%% OPCION 1 %%
syms x real  % defino 'x' como variable simbolica
f = sin(x)^2*cos(x);  % integrando (expresion ...
   simbolica)
int(f, -pi/2, pi/2)  % calculo la integral definida ...
   pedida

%% OPCION 2 %%
simplify(f - subs(f, x, -x))  % funcion PAR: f(x)=f(-x)
2*int(f, 0, pi/2)  % utilizo propiedades de simetria ...
   para calcular la integral definida pedida

% visualizacion %
x = linspace(-pi/2, pi/2);  % secuencia de 100 ...
   numeros (equiespaciados) entre -pi/2 y pi/2
y = sin(x).^2.*cos(x);  % valores (numericos) que ...
   toma el integrando para esa secuencia de 'x'
area(x, y)  % area encerrada entre la grafica ...
   definida por el integrando y el eje X
```

```
grid on
```

2. $\int_{-2}^{2} x(x^2+1)^3 dx$

```
%% OPCION 1 %%
syms x real  % defino 'x' como variable simbolica
f = x*(x^2+1)^3;  % integrando (expresion simbolica)
int(f, -2, 2)  % calculo la integral definida pedida

%% OPCION 2 %%
simplify(f + subs(f, x, -x))  % funcion IMPAR: ...
   f(x)=-f(-x). Por tanto, la integral pedida es 0

% visualizacion %
x = linspace(-2, 2);  % secuencia de 100 numeros ...
   (equiespaciados) entre -2 y 2
y = x.*(x.^2+1).^3;  % valores (numericos) que toma ...
   el integrando para esa secuencia de 'x'
area(x, y)   % area encerrada entre la grafica ...
   definida por el integrando y el eje X
grid on
```

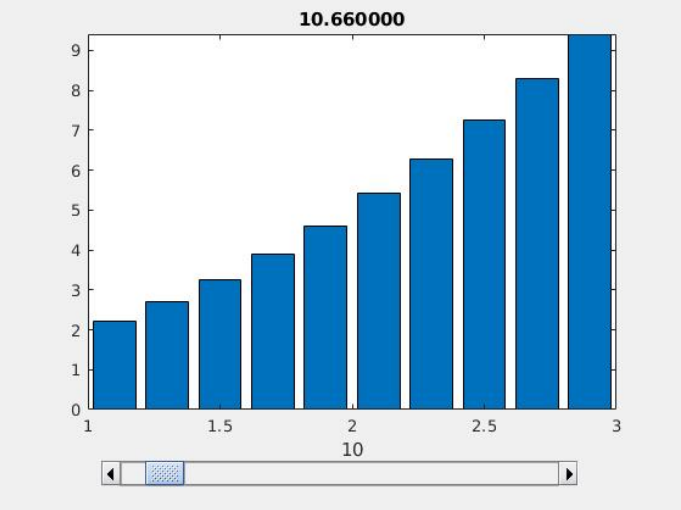
7.2.1. Aproximación mediante sumas de Riemann

La función `rsums` permite aproximar el valor de una integral definida mediante sumas de Riemann. Para ello, se abre una ventana gráfica desde la cual es posible ajustar el número de rectángulos considerados en la aproximación (el área correspondiente aparecerá en la parte superior de la figura). Ten en cuenta que `rsums` requiere como entrada una función simbólica.

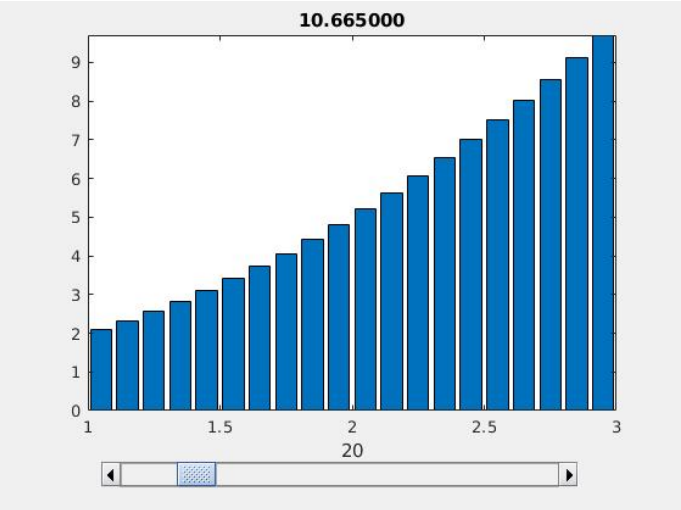
Ejemplo 7.5

Aproxima el valor de $\int_1^3 \left(x^2+1\right) dx$ mediante sumas de Riemann con 10, 20 y 30 rectángulos. ¿Qué error porcentual (con respecto al valor exacto) se comete en cada una de estas aproximaciones?

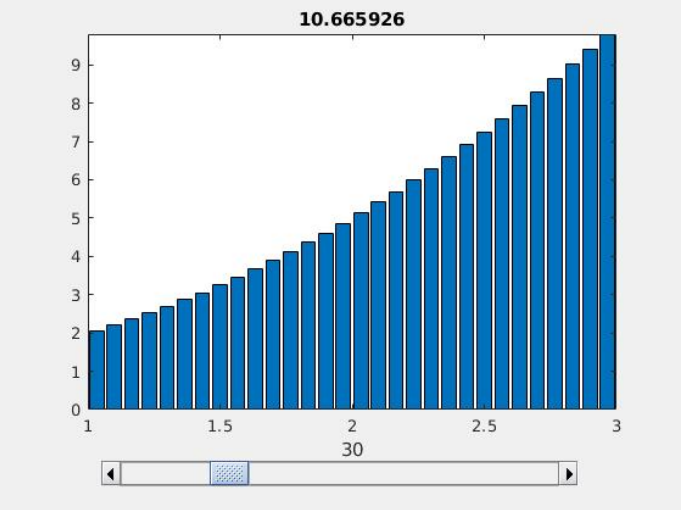
```
syms x real  % defino 'x' como variable simbolica
rsums(x^2+1, 1, 3)  % aproximacion de la integral pedida ...
   con 10 rectangulos (valor por defecto que usa 'rsums')
```




```
va10 = 10.66;  % valor aproximado de la integral pedida ...
   usando 10 rectangulos
```




```
va20 = 10.665;  % valor aproximado de la integral pedida ...
   usando 20 rectangulos
```




```
va30 = 10.665926;  % valor aproximado de la integral ...
   pedida usando 30 rectangulos
```

```
ve = int(x^2+1, 1, 3);  % valor exacto de la integral pedida
100*abs(([va10 va20 va30] - ve) / ve)  % errores (en %) ...
   cometidos por las aproximaciones con 10, 20 y 30 ...
   rectangulos
```

Como se puede observar, la aproximación mediante sumas de Riemann mejora a medida que aumenta el número de rectángulos considerados.

7.3. Ejercicios

1. Dada la función $f(x) = ln^2(x)$, ¿cuánto valdría $F(\sqrt{\pi})$, siendo F una primitiva de f con constante de integración igual a 5?
2. Representa en una misma figura las funciones $f(x) = (x-1)^2$ y $g(x) = \sqrt{\frac{\pi}{3}}x$ y halla el área que queda encerrada entre ambas.
3. ¿Qué error porcentual se comete al aproximar mediante sumas de Riemann con 5, 15 y 25 rectángulos la integral $\int_{-1}^{\pi} e^x sen(2x)dx$?
4. Comprueba, estudiando su paridad, que las integrales $\int_{-5}^{5} \left(\frac{arctan(x)}{\sqrt{3}}\right) dx$ y $\int_{-1}^{1} \left(x^3 - x\right)^5 dx$ son nulas. A continuación, ratifica tus resultados gráficamente utilizando la función `area`.

8 Graficado de funciones de dos variables

Definición 8.1

Una **función real de dos variables** f es una correspondencia que asigna a cada par de números reales (x, y) otro número real único, $z = f(x, y)$.

El **dominio** D es el conjunto de pares (x, y) para los cuales la función f está definida.

El **rango** (imagen) de f es el conjunto de números reales z tales que $(x, y) \in D$.

8.1. Superficies en coordenadas cartesianas

Si quisiéramos representar la función $z = f(x, y) = x^2 + y^2$ en $x = [-2, 2] \times y = [-1, 1]$, tendríamos que generar primero una malla sobre el plano XY que cubra el dominio de interés, para lo cual recurriremos a la función `meshgrid`.

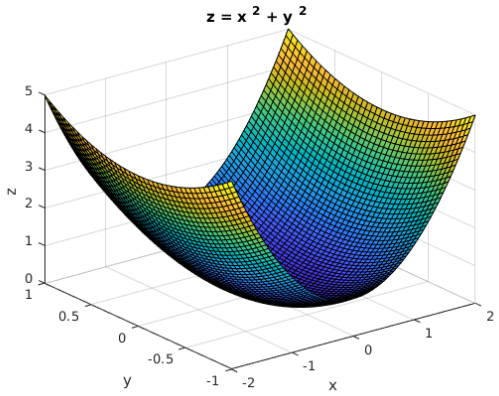
```
x = linspace(-2, 2);  % coordenadas x del dominio: vector de ...
   100 numeros (equiespaciados)
y = -1:0.05:1  % coordenadas y del dominio: vector con 41 ...
   numeros
[X Y] = meshgrid(x, y);  % malla que cubre el dominio de ...
   interes

length(x), length(y)  % vectores
size(X), size(Y)  % matrices
```

Fíjate que mientras las entradas de `meshgrid`, x e y, son vectores de longitud 100 y 41, respectivamente, las salidas X y Y son matrices de tamaño 41×100.

Una vez hemos creado la malla que cubre el dominio deseado, sólo tendremos que definir $f(x, y)$ y utilizar la función `surf` para representarla.

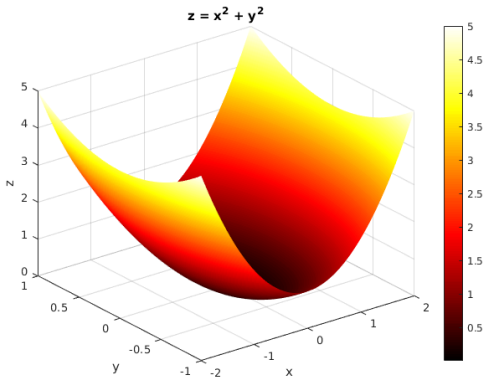
Nota: X e Y son matrices numéricas, por lo que Z se calculará punto a punto (es decir, habrá que utilizar las operaciones .*, ./, .^, etc.).

```
Z = X.^2 + Y.^2;  % funcion a representar: matriz
surf(X, Y, Z)  % represento la superficie
xlabel('x'), ylabel('y'), zlabel('z')
title('z = x^2 + y^2')
```



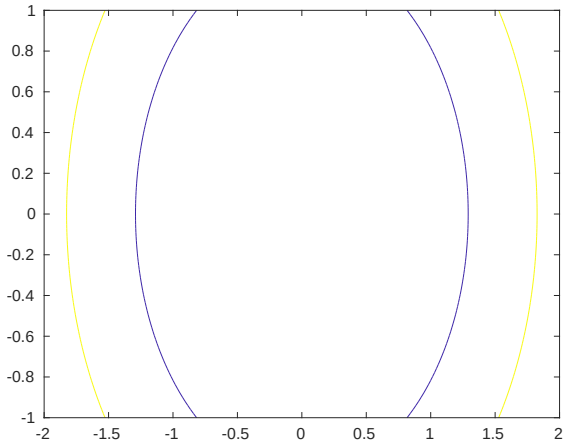
Podemos cambiar la paleta de colores fácilmente, así como incluir una barra de colores y dar transparencia a la figura.

```
surf(X, Y, Z)  % represento la superficie
xlabel('x'), ylabel('y'), zlabel('z')
title('z = x^2 + y^2')

colormap(hot)  % paleta de colores 'hot'
colorbar  % barra de colores (escala)
shading interp  % interpolo lineas y colores (genera un ...
   efecto similar a introducir una transparencia)
```

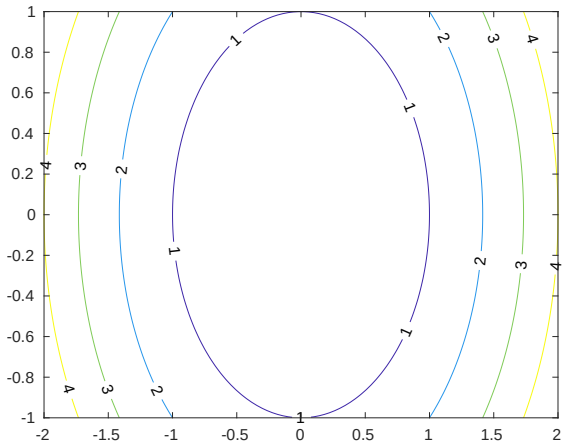


Para una función $z = f(x, y)$, la **curva de nivel** de valor k es el conjunto de todos los pares $(x, y) \in D$ tales que su imagen por f es el valor k. La función `contour` pemite dibujar las curvas de nivel de cualquier función $z = f(x, y)$.

```
contour(X, Y, Z, 2)  % se representan 2 curvas de nivel
```




```
contour(X, Y, Z, 1:4, 'ShowText', 'on')  % se representan ...
   las curvas de nivel correspondientes a k = {1,2,3,4}
% el argumento opcional 'ShowTest' permite visualizar el 'k' ...
   correspondiente a cada curva de nivel
```



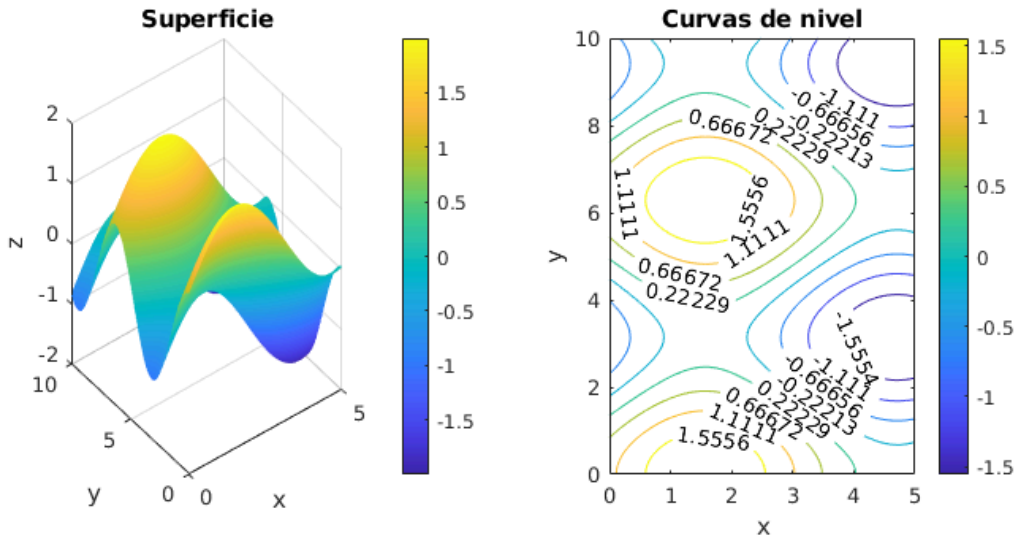
Es posible combinar distintos gráficos en una única figura haciendo uso del comando `subplot(m,n,p)`, que divide la ventana gráfica en una matriz con $m \times n$ sistemas de referencia (cada uno con sus respectivos ejes) y selecciona el que ocupa la posición p para dibujar sobre él.

Nota: p crece de izquierda a derecha y de arriba a abajo.

Ejemplo 8.2

Abre una nueva ventana gráfica y divídela en dos mitades. En la mitad de la derecha dibuja la superficie $z = sen(x) + cos(y)$ en el dominio $[0, 5] \times [0, 10]$. En la de la izquierda, dibuja 8 curvas de nivel.

```
[X, Y] = meshgrid(linspace(0, 5), linspace(0, 10));
Z = sin(X) + cos(Y);

figure
subplot(1, 2, 1)  % mitad izquierda de la figura
surf(X, Y, Z)
xlabel('x'), ylabel('y'), zlabel('z')
colorbar
title('Superficie')
shading interp

subplot(1, 2, 2)  % mitad derecha de la figura
contour(X, Y, Z, 8, 'ShowText', 'on')
xlabel('x'), ylabel('y')
colorbar
title('Curvas de nivel')
```



8.2. Superficies en coordenadas polares

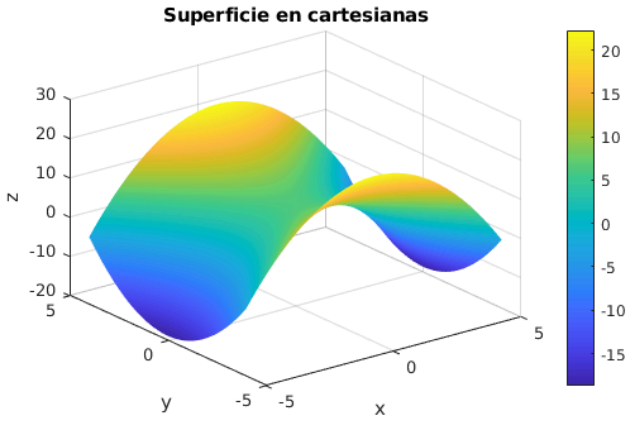
Una superficie $z = f(x, y)$ será simétrica respecto al eje OZ si $f(x, y) = f(-x, -y)$. Para representar este tipo de superficies es más adecuado utilizar como dominio un círculo centrado en $(0, 0)$ en lugar de un rectángulo. Para ello, tendríamos que cambiar nuestro sistema de referencia de coordenadas cartesianas (x, y) a polares (r, θ), donde $r > 0$ sería la distancia al centro del círculo de un punto proyectado en el plano XY y $0 \leq \theta < 2\pi$ el ángulo polar. Como sabes, la conversión de cartesianas a polares es $x = rcos(\theta)$ e $y = rsen(\theta)$.

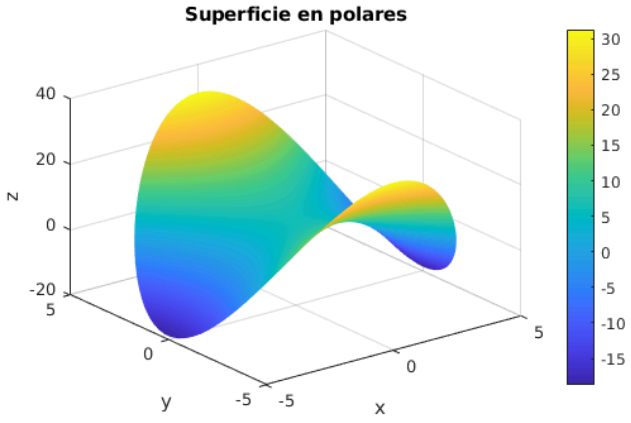
Ejemplo 8.3

Abre una nueva ventana gráfica y divídela en dos mitades. En la mitad superior dibuja la superficie $z = 2\pi - x^2 + y^2$ en el dominio $x = [-5, 5] \times y = [-4, 4]$. Comprueba que z presenta simetría con respecto al eje OZ. A continuación, convierte esa superficie a coordenadas polares y represéntala, en la mitad inferior, en el dominio $r = [0, 5] \times \theta = [0, 2\pi]$.

```
%% Superficie en cartesianas %%
[X, Y] = meshgrid(linspace(-5, 5), linspace(-4, 4));
Z = 2*pi - X.^2 + Y.^2;

figure
subplot(2, 1, 1)  % mitad superior: superficie en ...
   cartesianas
surf(X, Y, Z)
xlabel('x'), ylabel('y'), zlabel('z')
colorbar
title('Superficie en cartesianas')
shading interp


%% Compruebo simetria con respecto a eje OZ %%
syms x y real
z = 2*pi - x^2 + y^2;  % defino funcion simbolica
simplify(z- subs(z, [x, y], [-x, -y]))  % f(x,y) = ...
   f(-x,-y), por lo que la f es simetrica con respecto ...
   al eje OZ


%% Superficie en polares %%
[R, T] = meshgrid(linspace(0, 5), linspace(0, 2*pi));
X = R.*cos(T);
Y = R.*sin(T);
```

```
Z = 2*pi - X.^2 + Y.^2;

subplot(2, 1, 2)
surf(X, Y, Z)  % mitad inferior: superficie en polares
xlabel('x'), ylabel('y'), zlabel('z')
colorbar
title('Superficie en polares')
shading interp
```





8.3. Ejercicios

1. Representa las siguientes funciones de dos variables:

   a) $z = \frac{x}{2}^2 + \frac{y}{3}^2$ en $[-3, 3] \times [-3, 3]$

   b) $z = \frac{sen(x^2+y^2)}{x^2+y^2}$ en $[-3, 3] \times [-3, 3]$

   c) $z = \frac{log(x^2+y^2)}{x^2+y^2-1}$ en $[-1,5, 1,5] \times [-1,5, 1,5]$

   ¿Cuál es su dominio? ¿Son continuas en el rectángulo escogido para la representación?

2. *a*) Considera la función $z = \frac{x}{2}^2 + \frac{y}{3}^2$. Comprueba si es simétrica respecto al eje OZ. En caso de serlo, abre una ventana gráfica y divídela en dos mitades. En la mitad izquierda, representa z en coordenadas polares sobre el dominio $r = [0, 4] \times \theta = [0, 2\pi]$. En la derecha, dibuja 10 curvas de nivel.

   b) Repite el ejercicio del apartado anterior para la función $z = \frac{sen(x^2+y^2)}{x^2+y^2}$. En este caso, dibuja las curvas de nivel para $k = \{0; 0,1; 0,2; 0,3; 0,4; 0,5; 0,6; 0,7; 0,8; 0,9; 1\}$. ¿Qué ocurre para $k = 1$?

9 Derivación parcial: Gradiente

9.1. Cálculo de derivadas parciales

Para el cálculo de derivadas parciales utilizaremos la función `diff`, en la que habrá que especificar la variable con respecto a la cual se quiera derivar y el orden de la derivada de interés.

Ejemplo 9.1

Dada la función $f(x,y) = \frac{e^x}{sen(y)}$, calcula $\frac{\partial f}{\partial x}$, $\frac{\partial f}{\partial y}$, $\frac{\partial^2 f}{\partial x^2}$, $\frac{\partial^2 f}{\partial y^2}$ y $\frac{\partial^2 f}{\partial x \partial y}$ en el punto $P(-3, \frac{\pi}{2})$. Comprueba que $\frac{\partial^2 f}{\partial x \partial y} = \frac{\partial^2 f}{\partial y \partial x}$.

```
syms x y real; f = exp(x)/sin(y);  % funcion a derivar

fx = diff(f, 1, x);  % primera derivada parcial con ...
   respecto a 'x'
subs(fx, [x, y], [-3, pi/2])

fy = diff(f, y);  % primera derivada parcial con ...
   respecto a 'y'
subs(fy, [x, y], [-3, pi/2])

fxx = diff(f, 2, x);  % segunda derivada parcial con ...
   respecto a 'x'
subs(fxx, [x, y], [-3, pi/2])

fyy = diff(f, 2, y);  % segunda derivada parcial con ...
   respecto a 'y'
subs(fyy, [x, y], [-3, pi/2])

fxy = diff(diff(f, y), x);  % derivada cruzada
subs(fxy, [x, y], [-3, pi/2])

fyx = diff(diff(f, x), y);
simplify(fxy -fyx)  % igualdad derivadas cruzadas
```

Ejemplo 9.2

Dada la función $w = ze^{(x+y)}$, siendo $x = s$, $y = t^2$ y $z = s^2t$, calcula $\frac{\partial w}{\partial s}$ y $\frac{\partial w}{\partial t}$.

```
syms s t x y z real
x = s; y = t^2; z = s^2*t;
w = z*exp(x+y);  % funcion a derivar

ws = diff(w, 1, s)  % derivada parcial de 'w' con ...
   respecto a 's'
simplify(ws)
wt = diff(w, 1, t)  % derivada parcial de 'w' con ...
   respecto a 't'
simplify(wt)
```

Ejemplo 9.3

Dada la función $F(x, y, z) = x^2 - y^2 + 2z - 4 = 0$, calcula $\frac{\partial z}{\partial x}$ y $\frac{\partial z}{\partial y}$. Utiliza la derivación implícita.

```
syms x y z real
F = x^2-y^2+2*z-4;  % funcion a derivar

zx = -diff(F, x)/diff(F, z)  % derivacion implicita
zy = -diff(F, y)/diff(F, z)  % derivacion implicita
```

9.2. Cálculo y representación del campo de vectores gradiente

A partir de aquí, el cálculo del gradiente de una función $z = f(x, y)$ sería inmediato.

Ejemplo 9.4

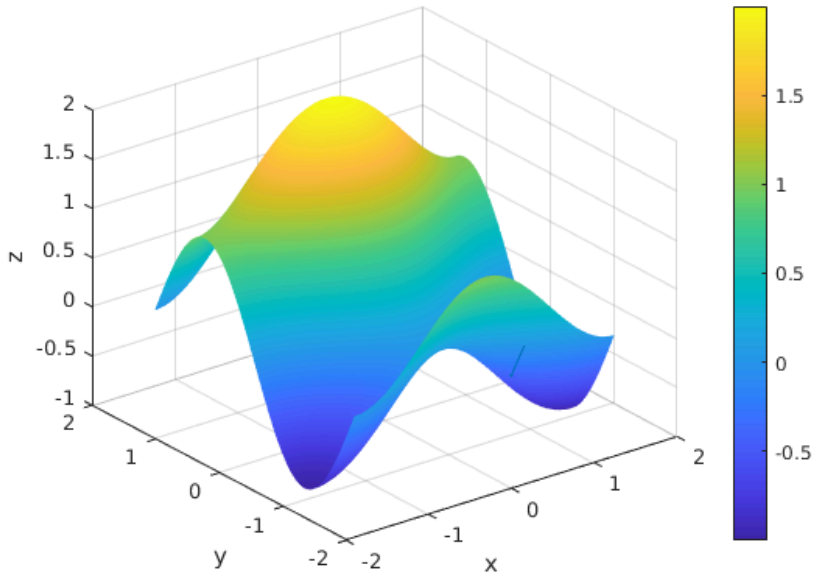
Dada la función $f(x, y) = cos^2(x) + sen(2y)$, calcula $\vec{\nabla} f$ en el punto $P(1, -1)$.

```
syms x y real
f = cos(x)^2 + sin(2*y);
```

```
fx = diff(f, 1, x);  % derivada parcial con respecto a 'x'
fy = diff(f, 1, y);  % derivada parcial con respecto a 'y'
gradfP = double([subs(fx, [x,y], [1,-1]) , subs(fy, ...
   [x,y], [1,-1])])  % evaluo el gradiente en el punto 'P'
```

Podríamos dibujar la superficie definida por f (por ejemplo en el dominio $x = [-\frac{\pi}{2}, \frac{\pi}{2}] \times y = [-\frac{\pi}{2}, \frac{\pi}{2}]$), junto con el vector gradiente a f en el punto P. Para dibujar vectores en MATLAB se usa el comando `quiver`:

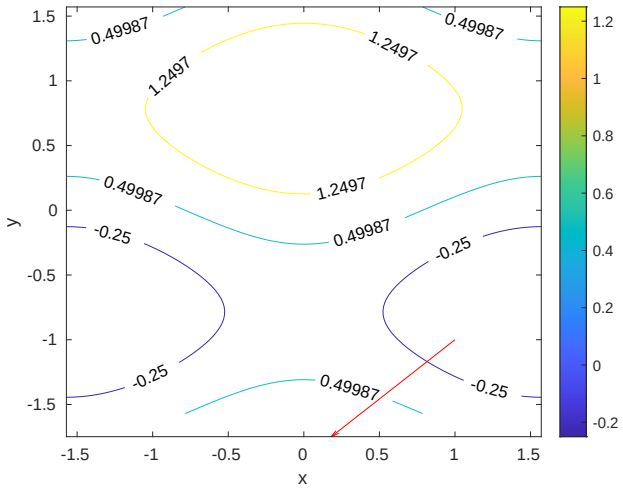
```
[X, Y] = meshgrid(linspace(-pi/2, pi/2), linspace(-pi/2, ...
   pi/2));  % genero una rejilla de puntos sobre el dominio ...
   de interes
Z = cos(X).^2 + sin(2*Y);  % defino superficie
surf(X,Y,Z)  % dibujo superficie
shading interp
colorbar, xlabel('x'), ylabel('y'), zlabel('z')
hold on
quiver(1, -1, gradfP(1), gradfP(2))  % dibujo un vector que ...
   tiene por origen el punto (1,-1), y por componentes ...
   (gradfP(1), gradfP(2))
```



Como sabes, el vector gradiente se sitúa sobre el plano XY, por lo que la representación más natural de nuestro problema sería en 2-dimensiones, sobre dicho plano. Para ello, podríamos

proyectar algunas curvas de nivel de f (por ejemplo, 3) junto con el vector gradiente en P:

```
figure
contour(X, Y, Z, 3, 'ShowText', 'on')  % dibujo 3 curvas de ...
   nivel sobre el plano XY
colorbar, xlabel('x'), ylabel('y')
hold on
quiver(1, -1, gradfP(1), gradfP(2), 'Color', 'r')  % vector ...
   gradiente, en rojo
```



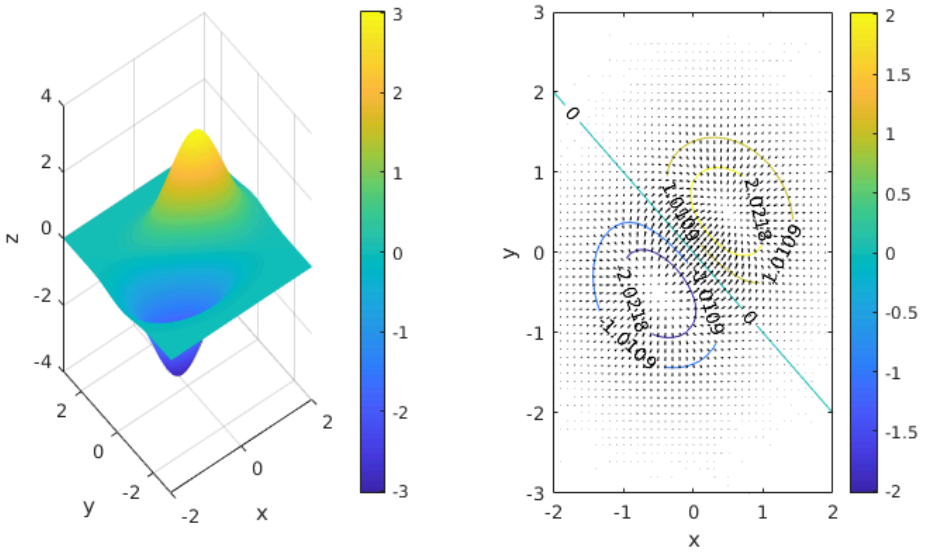
La función `gradient` permite calcular directamente el valor (numérico) que tomarían las componentes de los vectores gradiente a una función $f(x, y)$ sobre cada uno de los puntos del dominio $x = [x_1, x_2] \times y = [y_1, y_2]$ en el que se define (que se especificará como una malla de puntos). Una vez calculadas dichas componentes, se podría utilizar la función `quiver` para representar todo el campo de vectores $\vec{\nabla} f$.

Ejemplo 9.5

Abre una nueva ventana gráfica con 1×2 subventanas. En la mitad izquierda dibuja la superficie $f(x, y) = 5(x + y)e^{(-x^2-y^2)}$, definida sobre el dominio $x = [-2, 2] \times y = [-3, 3]$. En la izquierda, dibuja 5 curvas de nivel junto con el campo de vectores $\vec{\nabla} f$.

```
x = -2:0.1:2;
y = -3:0.1:3;
[X, Y] = meshgrid(x, y);  % malla que cubre el dominio ...
   de 'f'
Z = 5*(X+Y) .* exp(-X.^2-Y.^2); % superficie: 'X' e 'Y' ...
   son matrices numericas, por lo que 'Z' se calcula ...
```

```
   punto a punto
[Zx, Zy] = gradient(Z);  % componentes del vector ...
   gradiente a 'f' en cada punto de la malla

figure  % abro una nueva ventana grafica
subplot(1,2,1)  % matriz con 1x2 plots (activo el ...
   elemento (1,1))
surf(X, Y, Z)  % dibujo superficie
colorbar, xlabel('x'), ylabel('y'), zlabel('z')
shading interp

subplot(1,2,2)  % matriz con 1x2 plots (activo el ...
   elemento (1,2))
contour(X, Y, Z, 5, 'ShowText', 'on')  % dibujo 5 curvas ...
   de nivel
colorbar, xlabel('x'), ylabel('y')
hold on
quiver(X, Y, Zx, Zy, 'Color', 'k')  % campo de vectores ...
   gradiente, en color negro
```

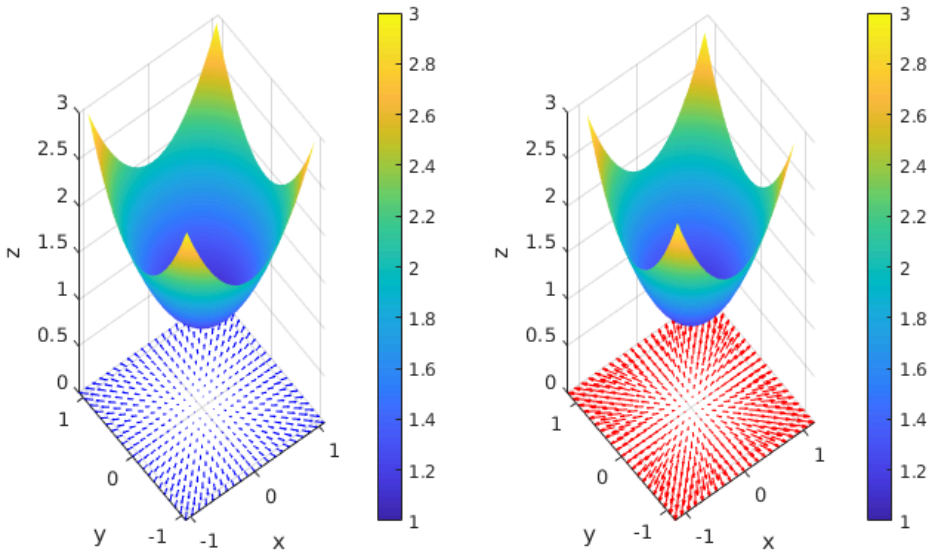


Puedes ver como los vectores gradiente son siempre perpendiculares a las curvas de nivel. Además, el flujo de vectores sale de un mínimo de f (fuente) y llega a un máximo (sumidero).

Por defecto, el módulo (longitud) de los vectores que dibuja `quiver` no es el real, sino que la propia función hace un reescalado para que no se solapen unos vectores con otros y la representación gráfica mantenga un aspecto 'limpio'. Si nos interesase visualizar el módulo real

de los vectores, habría que fijar el parámetro opcional `AutoScale` a `off` en `quiver`. Veámoslo para la superficie $f(x, y) = x^2 + y^2 + 1$, definida sobre el dominio $x = [-1, 1] \times y = [-1, 1]$.

```
[X, Y] = meshgrid(-1:0.1:1, -1:0.1:1);  % genero rejilla ...
   cubriendo el dominio
Z = X.^2 + Y.^2 + 1;  % defino superficie
[Zx, Zy] = gradient(Z);  % calculo las componentes del ...
   vector gradiente a 'f' en cada punto de la rejilla

subplot(1,2,1)  % matriz con 1x2 plots (activo el elemento ...
   (1,1))
surf(X, Y, Z)  % dibujo superficie
colorbar, xlabel('x'), ylabel('y'), zlabel('z'), shading interp
hold on
quiver(X, Y, Zx, Zy, 'Color', 'b')  % campo de vectores ...
   gradiente (en negro, modulo reescalado)

subplot(1,2,2)  % matriz con 1x2 plots (activo el elemento ...
   (1,2))
surf(X, Y, Z)  % dibujo superficie
colorbar, xlabel('x'), ylabel('y'), zlabel('z'), shading interp
hold on
quiver(X, Y, Zx, Zy, 'Color', 'r', 'AutoScale', 'off')  % ...
   campo de vectores gradiente (en rojo, modulo real)
```



9.3. Ejercicios

1. Calcula las siguientes derivadas parciales:
   a) $\frac{\partial f}{\partial x}$, $\frac{\partial f}{\partial y}$ y $\frac{\partial f}{\partial z}$ en el punto $P(1,-1,1)$, siendo $f = \frac{x}{yz}$.
   b) $\frac{\partial z}{\partial x}$ y $\frac{\partial z}{\partial y}$ en el punto $P(0,1,-2)$, siendo $F = e^x sen(y+z) - z = 0$. Utiliza la derivación implícita.
   c) $\frac{df}{dt}$ en $t = \pi$, siendo $f = xyz$, con $x = t^2$, $y = 2t$ y $z = e^{-t}$.
   d) $\frac{\partial w}{\partial s}$ y $\frac{\partial w}{\partial t}$, siendo $w = xcos(yz)$, con $x = s^2$, $y = t^2$ y $z = t$.
2. Considera la función $f(x,y) = e^x cos(y) + e^y cos(x)$. Abre una nueva ventana gráfica con 1×2 subventanas y:
   a) En la mitad izquierda, representa z en el dominio $x = [-1,7] \times y = [-1,7]$.
   b) En la mitad derecha, dibuja las curvas de nivel correspondientes a los valores $k = \{-1000, -500, -250, 0, 250, 500, 1000\}$ y el campo de vectores gradiente.
3. Considera la superficie $f(x,y) = x^2 + cos(y^2)$. Abre una nueva ventana gráfica con 1×2 subventanas y:
   a) En la mitad izquierda, representa f en el dominio $x = [-1,1] \times y = [-1,1]$, junto con el vector gradiente a f en cada punto del dominio, en color rojo.
   b) En la mitad derecha, dibuja el vector gradiente a f (con su módulo real) en el punto $P(0{,}75; -0{,}5)$.

10 Derivación parcial: Plano tangente, puntos críticos

10.1. Plano tangente a una superficie

Definición 10.1: Plano tangente

Sea S una superficie diferenciable en el punto $P(x_0, y_0, z_0)$. La ecuación del plano tangente a S en P es:

1. $\boxed{F_x(x_0, y_0, z_0)(x - x_0) + F_y(x_0, y_0, z_0)(y - y_0) + F_z(x_0, y_0, z_0)(z - z_0) = 0}$, si S viene dada implícitamente como $S = F(x, y, z) = 0$
2. $\boxed{z - z_0 = f_x|_P(x_0, y_0)(x - x_0) + f_y(x_0, y_0)(y - y_0)}$, si S viene dada explícitamente como $S = z = f(x, y)$

Por tanto, con las funciones `diff` y `surf` podremos calcular y representar el plano tangente a una superficie del tipo $z = f(x, y)$ en un punto P.

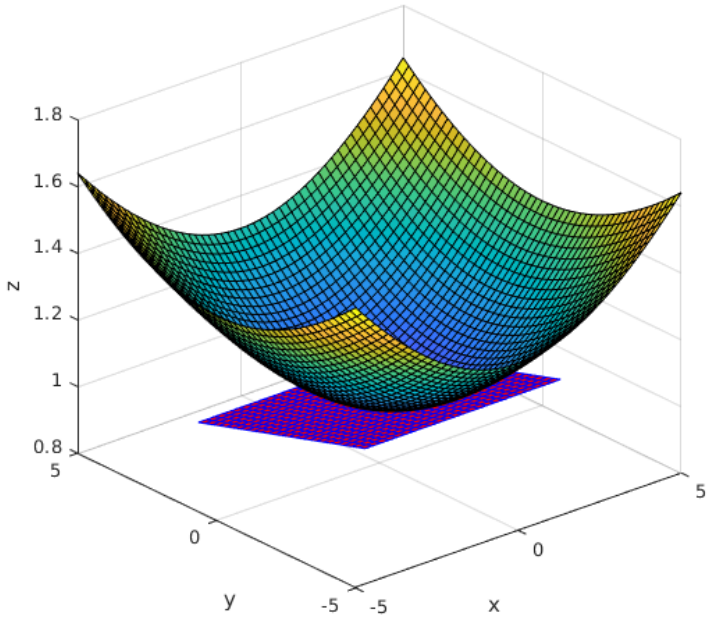
Ejemplo 10.2

Dada la función $z = f(x, y) = \sqrt{1 + (\frac{x}{5})^2 + (\frac{y}{6})^2}$:

1. Representa z en el dominio $x = [-5, 5] \times y = [-5, 5]$ y su plano tangente en $P(0, -1)$ en $x = [-3, 3] \times y = [-3, 3]$.

```
[X, Y] = meshgrid(-5:0.2:5, -5:0.2:5);  % malla ...
   sobre el dominio de 'z'
Z = sqrt(1 + (X./5).^2 + (Y./6).^2);  % superficie ...
   (numerica)
figure  % nueva ventana grafica
surf(X, Y, Z)  % dibujo superficie
xlabel('x'), ylabel('y'), zlabel('z')

syms x y real  % variables simbolicas
z = sqrt(1 + (x/5)^2 + (y/6)^2);  % superficie ...
```

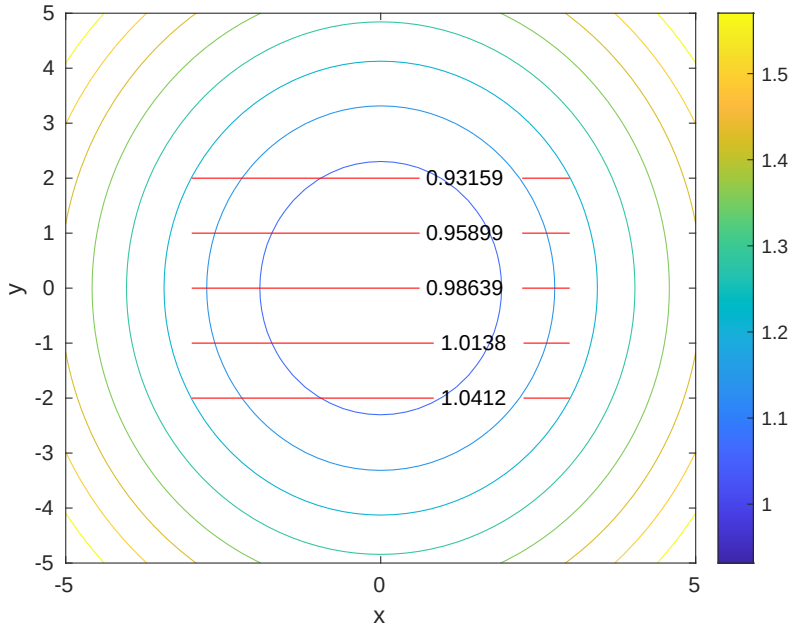
```
   (simbolica)
zx = diff(z, x);  % parcial de 'z' con respecto a 'x'
zy = diff(z, y);  % parcial de 'z' con respecto a 'y'
P = [0 -1];  % punto P
z0 = double(subs(z, [x y], P));  % valor que toma la ...
   superficie en P
[Xpt, Ypt] = meshgrid(-3:0.25:3, -3:0.25:3);  % ...
   malla sobre el dominio del plano tangente a 'z'
Zpt = double(subs(zx, [x y], P))*(Xpt - x0) + ...
   double(subs(zy, [x y], P))*(Ypt - y0) + z0;  % ...
   plano tangente a 'z' en P (numerico)
hold on
pt = surf(Xpt, Ypt, Zpt);  % dibujo plano tangente
set(pt, 'FaceColor','red','EdgeColor','b')  % plano ...
   en color rojo, rejilla en azul
```



2. Dibuja 8 curvas de nivel de z y otras 5 del plano tangente en el recinto $x = [-5, 5] \times y = [-5, 5]$.

```
figure  % nueva ventana grafica
contour(X,Y,Z,8), colorbar  % curvas de nivel de Z
```

```
hold on
contour(Xpt,Ypt,Zpt,5,'Color','red','ShowText','on') ...
    % curvas de nivel del plano tangente, en rojo
xlabel('x'), ylabel('y')
```



10.2. Clasificación de puntos críticos

Para hallar los puntos críticos de una función $z = f(x, y)$, hay que resolver el sistema de ecuaciones

$$\begin{cases} f_x(x, y) = 0 \\ f_y(x, y) = 0 \end{cases}$$

Para ello emplearemos la función `solve`. A continuación, en cada punto crítico (a, b) calcularemos el determinante de la matriz Hessiana de f:

$$|H| = \begin{vmatrix} f_{xx}(a, b) & f_{xy}(a, b) \\ f_{yx}(a, b) & f_{yy}(a, b) \end{vmatrix}$$

- Si $|H| > 0$ y $f_{xx}(a, b) > 0 \Rightarrow (a, b)$ es un mínimo relativo
- Si $|H| > 0$ y $f_{xx}(a, b) < 0 \Rightarrow (a, b)$ es un máximo relativo
- Si $|H| < 0 \Rightarrow (a, b)$ es un punto silla
- Si $|H| = 0$, no hay certeza sobre qué tipo de punto es (a, b)

Ejemplo 10.3

Halla los puntos críticos de la función $z = f(x,y) = \sqrt{1+(\frac{x}{5})^2+(\frac{y}{6})^2}$ en el dominio $x = [-5,5] \times y = [-5,5]$ y clasifícalos.

```
syms x y real  % variables simbolicas
z = sqrt(1 + (x/5)^2 + (y/6)^2);  % superficie (simbolica)
zx = diff(z, x);  % parcial de 'z' con respecto a 'x'
zy = diff(z, y);  % parcial de 'z' con respecto a 'y'

pc = solve(zx==0, zy==0, [x y])  % resuelvo sistema para ...
   hallar los puntos criticos
PC = double([pc.x, pc.y]) % (0,0) es el unico punto ...
   critico de 'z' en [-5,5]x[-5,5]

zxPC = double(subs(zx, [x y], PC));  % 'zx' en (0,0)
zyPC = double(subs(zy, [x y], PC));  % 'zy' en (0,0)

zxy = diff(zy, x);  % parcial cruzada de 'z'
zxyPC = double(subs(zxy, [x y], PC));  % 'zxy' en (0,0)

zxx = diff(z, 2, x);  % parcial segunda de 'z' con ...
   respecto a 'x', en (0,0)
zxxPC = double(subs(zxx, [x y], PC));  % 'zxx' en (0,0)

zyy = diff(z, 2, y);  % parcial segunda de 'z' con ...
   respecto a 'y', en (0,0)
zyyPC = double(subs(zyy, [x y], PC));  % 'zyy' en (0,0)

H = [zxxPC zxyPC; zxyPC zyyPC];  % matriz Hessiana
det(H), zxxPC  % det(H)>0 y zxxPC>0, por lo que el punto ...
   (0,0) es un minimo relativo
```

10.3. Ejercicios

1. Representa la superficie $f(x,y) = cos(x)sen(x+y)$ en el dominio $x = [-1,1] \times y = [-1,1]$, junto con su plano tangente en el punto $P(0{,}75; 0{,}5)$, este último sobre el dominio $x = [0; 1{,}5] \times y = [0; 1{,}5]$.
2. Considera la función $f(x,y) = -(x+3)^2 - y^2 + 3$. Abre una nueva ventana gráfica con 1×2 subventanas y:
   - En la mitad izquierda, representa f en el dominio $x = [-8,4] \times y = [-8,4]$, junto con su plano tangente en el punto $P(-1; 0{,}5)$, este último sobre el dominio $x = [-6,2] \times y = [-6,2]$.
   - En la mitad derecha, dibuja 10 curvas de nivel f, junto con otras 5 curvas de nivel del plano tangente (estas últimas en rojo).
3. Considera la función $f(x,y) = arctan(\frac{x}{y})$. Calcula el plano tangente a f en el punto $P(-1{,}25; 0{,}5)$ ¿Qué valor toma este plano en el punto $P'(1,0)$?
4. Halla el/los punto/s crítico/s de las siguientes funciones en el dominio $x = [-1,1] \times y = [-1,1]$ y clasifícalo/s de acuerdo con el criterio del Hessiano.
   a) $f(x,y) = x^2 - y^2$
   b) $f(x,y) = x - x^2 - \sqrt{5}y^3$