# MATLAB: Tutorial de ayuda para usarlo vinculado a temáticas impartidas en el curso Matemáticas para Finanzas con aplicaciones (Parte 1)

**María Gulnara Baldoquin de la Peña**
**Universidad EAFIT**

## Índice

## Introducción

Qué es el MATLAB? MATLAB es el nombre abreviado de "MATrix LABoratory".
Constituye un sistema interactivo cuyo elemento de dato básico es una matriz que no requiere dimensionamiento. Integra cálculos, visualización y programación.

MATLAB realiza cálculos numéricos con vectores y matrices. Como caso particular puede trabajar con números escalares −tanto reales como complejos−, que son matrices de dimensión 1. También con cadenas de caracteres y con otras estructuras de información más complejas.

MATLAB es un excelente programa de cálculo técnico y científico, así como un lenguaje de programación.

El sistema consiste de 5 partes fundamentales:
1. Ambiente de desarrollo
2. Poderosa librería de funciones matemáticas
3. Lenguaje de programación que incluye las características de la programación orientada a objeto.
4. Gráficos, con funciones de alto nivel para visualización de datos en 2 y 3 dimensiones, procesamiento de imágenes, animación y presentación de gráficos.
5. API (The MATLAB Application Program Interface ), librería que permite interactuar con programas en C, Fortran, Java.

## Ventana del MATLAB

Una vez instalado el MATLAB y presionado el icono del MATLAB aparecerá la ventana del MATLAB en el *desktop*.
La ventana del MATLAB está subdividida en tres subventanas que se describen a continuación.
1. Ventana de comandos (*Command Window*), donde se ejecutan los comandos del MATLAB. En dicha ventana el símbolo >> indica que puede introducir comandos del MATLAB.
2. Ventana de Directorio de trabajo (*Current directory o Current Folder*), el cual contiene todos los archivos que se encuentran en el directorio en el cual se está trabajando. En la parte superior de la ventana del MATLAB puede visualizar el directorio en el cual se encuentra y puede cambiarlo. Al iniciar una sesión de MATLAB el que presenta por defecto es el directorio *work* del MATLAB.
3. Ventana de historia de los comandos (*Command History o Workspace*) que muestra los comandos dados en la sesión actual y en sesiones anteriores pudiendo mostrar las fechas de esas sesiones.

La Figura 1 muestra una ventana del MATLAB. Sus posiciones pueden variarse, y también depende de la versión del MATLAB.

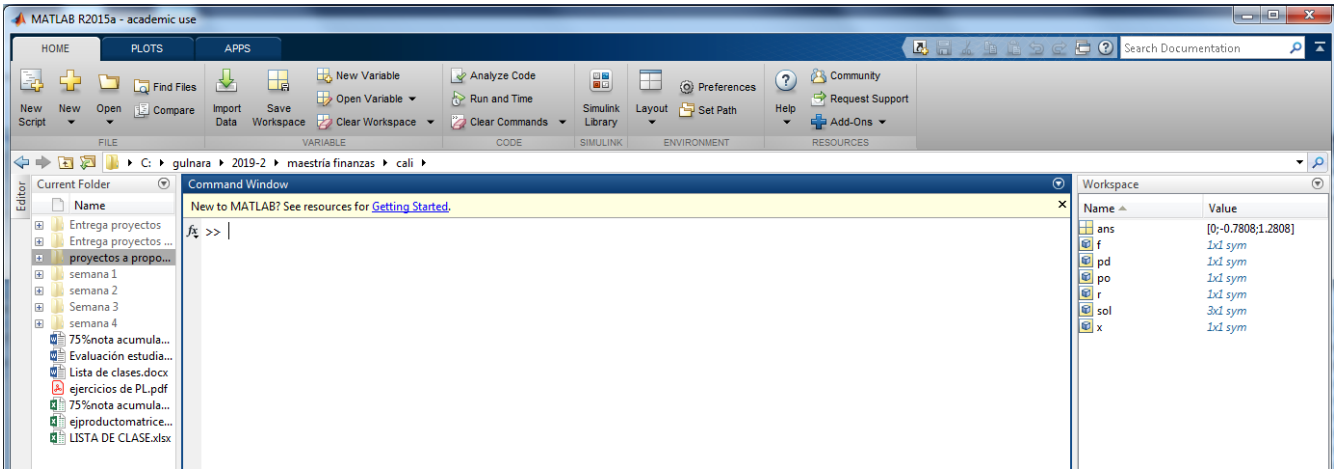



**Figura 1: Ventana del MATLAB**

En este contexto nos concentramos en la Ventana de comandos (*Command Window*), donde se ejecutan los comandos del MATLAB, y que en la Figura 1 aparece en el centro.

**Antes de iniciar explicación de comandos, es importante tener en cuenta algunos aspectos:**

1. Todos los comandos en MATLAB deben usarse en **minúscula**.
2. Las variables usadas en MATLAB pueden contener letras minúsculas, mayúsculas o mixtas, pero **el MATLAB es sensible a letras minúsculas y mayúsculas**. Lo anterior quiere decir que si colocó algo en la variable denotada A y luego la llama o usa esa misma como a no la entenderá. Igual serán variables diferentes por ejemplo, ra y Ra.
3. La potencia se expresa con el símbolo ^, por ejemplo, si queremos expresar el polinomio $x^2+1$ se debe poner x^2+1.
4. El MATLAB trabaja con números complejos, una extensión de los números reales y que puede descubrirlo si tiene en alguna respuesta un valor de la forma a+bi, donde a y b son números reales.
Ejemplos de números complejos son 3+2i, 0 +4i, etc.
Si le aparecen números donde el coeficiente de i no es cero (o sea que aparece i) no se tiene en cuenta en su respuesta.

Por ejemplo, al resolver la siguiente ecuación con el MATLAB $x^3-x^2+4x-4 = 0$ (luego se verá cómo se resuelve) la respuesta que da son 3 valores, pues una ecuación de tercer grado tiene 3 raíces.
**1.0000 + 0.0000i**
0.0000 - 2.0000i
0.0000 + 2.0000i

Observar que la única solución real es el valor 1 (pues el valor que acompaña a i es cero, no se tiene en cuenta) los otros dos valores que son números complejos no se consideran.

**Grupo de comandos generales no específicos con una temática en particular, muy importantes para quienes se inician en el MATLAB:**

**Comando *ver*:**
Es importante para que se conozca qué versión del MATLAB tiene instalado, cuáles toolboxs del MATLAb tiene instalado en su computadora (de ellos depende comandos que puede usar o no) y la versión que tiene de cada toolbox.

En este contexto son importantes los toolboxs de Matemática Simbólica y de Optimización.
En la Figura 2 aparece el resultado del comando ver, según lo instalado en una computadora.

```
>> ver
-----------------------------------------------------------------------------------------------------
MATLAB Version: 8.5.0.197613 (R2015a)
MATLAB License Number: 40686331
Operating System: Microsoft Windows 7 Professional  Version 6.1 (Build 7601: Service Pack 1)
Java Version: Java 1.7.0_60-b19 with Oracle Corporation Java HotSpot(TM) 64-Bit Server VM mixed mode
-----------------------------------------------------------------------------------------------------
MATLAB                                                Version 8.5        (R2015a)
Simulink                                              Version 8.5        (R2015a)
Control System Toolbox                                Version 9.9        (R2015a)
Curve Fitting Toolbox                                 Version 3.5.1      (R2015a)
Data Acquisition Toolbox                              Version 3.7        (R2015a)
Datafeed Toolbox                                      Version 5.1        (R2015a)
Econometrics Toolbox                                  Version 3.2        (R2015a)
Financial Instruments Toolbox                         Version 2.1        (R2015a)
Financial Toolbox                                     Version 5.5        (R2015a)
Fuzzy Logic Toolbox                                   Version 2.2.21     (R2015a)
Image Acquisition Toolbox                             Version 4.9        (R2015a)
Image Processing Toolbox                              Version 9.2        (R2015a)
Instrument Control Toolbox                            Version 3.7        (R2015a)
MATLAB Coder                                          Version 2.8        (R2015a)
MATLAB Compiler                                       Version 6.0        (R2015a)
MATLAB Compiler SDK                                   Version 6.0        (R2015a)
Mapping Toolbox                                       Version 4.1        (R2015a)
Neural Network Toolbox                                Version 8.3        (R2015a)
Optimization Toolbox                                  Version 7.2        (R2015a)
Robust Control Toolbox                                Version 5.3        (R2015a)
Signal Processing Toolbox                             Version 7.0        (R2015a)
Simulink 3D Animation                                 Version 7.3        (R2015a)
Simulink Coder                                        Version 8.8        (R2015a)
Simulink Control Design                               Version 4.2        (R2015a)
Simulink Design Optimization                          Version 2.7        (R2015a)
Simulink Desktop Real-Time                            Version 5.0        (R2015a)
Simulink Real-Time                                    Version 6.2        (R2015a)
Stateflow                                             Version 8.5        (R2015a)
Statistics and Machine Learning Toolbox               Version 10.0       (R2015a)
Symbolic Math Toolbox                                 Version 6.2        (R2015a)
System Identification Toolbox                         Version 9.2        (R2015a)
Wavelet Toolbox                                       Version 4.14.1     (R2015a)
```

**Figura 2: Versión de MATALB y toolboxes instalados, en una computadora**

**Comando *help* (ejemplificado con el comando *clc*):**
Si se conoce el nombre de un comando del MATLAB, pero no la sintaxis de cómo utilizarlo, o se quiere profundizar más sobre este comando una de las posibilidades es:
Con el formato *help* seguido de la palabra que identifica el comando

**Ejemplo 1*:***
>> help clc
clc - Clear Command Window
This MATLAB function clears all input and output from the Command Window display, giving you a clean screen.

La Figura 3 en la parte superior muestra la ventana de los comandos del MATLAB con lo trabajado hasta el momento.
Si donde aparece >> al final de la pantalla se coloca clc, aparecerá la pantalla "borrada" que se muestra en la parte inferior de la Figura 3.

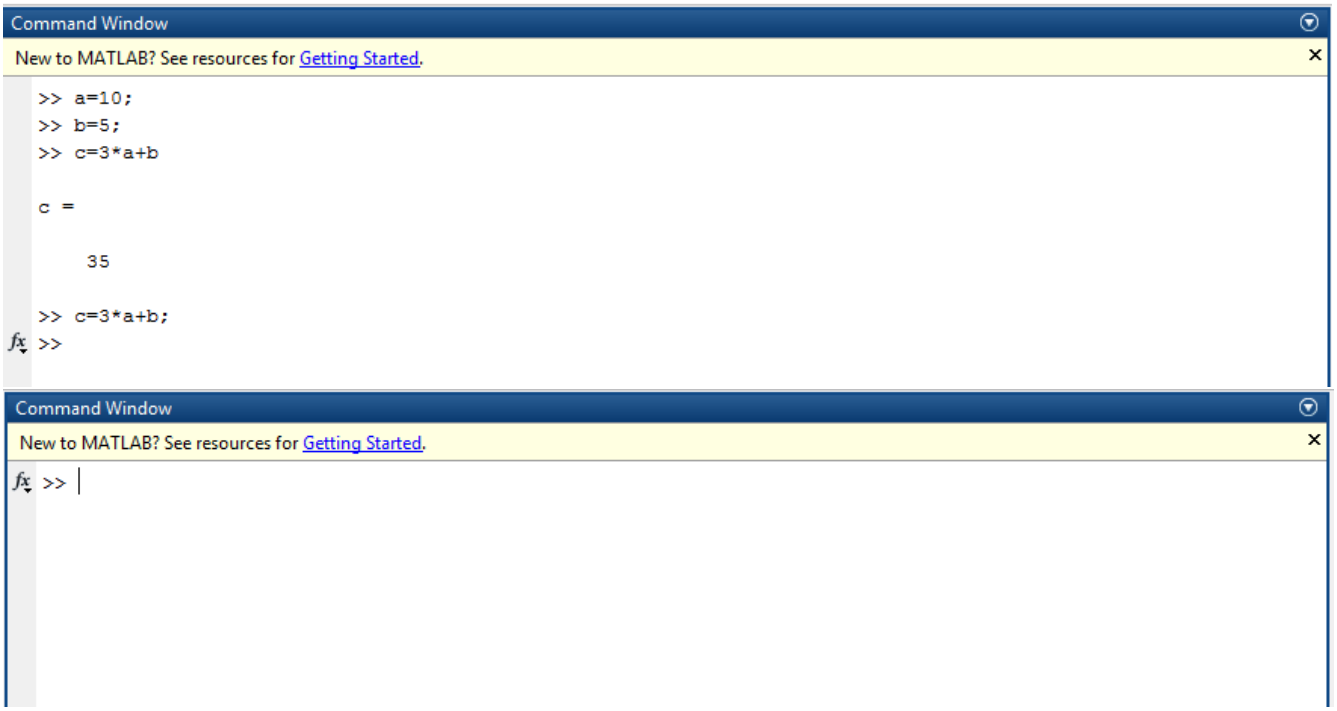



Figura 3: Uso del comando clc para limpiar ventana de trabajo

Cuando usa la función help el comando que pretende buscar información de él sí puede ponerlo en mayúscula, pero no usarlo en mayúscula. Respecto al Ejemplo 1 usando así el MATLAB:
>> help CLC
clc - Clear Command Window
This MATLAB function clears all input and output from the Command Window display, giving you a clean screen.

>> CLC
Undefined function or variable 'CLC'.
Did you mean:
>> clc

**Comando *ans*:**
La palabra *ans* (de *answer*, respuesta) es un nombre de variable del MATLAB por defecto para colocar el resultado de la última operación realizada que no ha sido asignada a ninguna otra variable.

**Ejemplo 2**:
>> >> a=10;
>> b=5;
>> 3*a+b
ans = 35
>> c=3*a+b
c = 35

Observar que se asignaron los valores 10 y 5 a las variables a y b, respectivamente. Como la operación 3*a+b no se asignó a ninguna variable, la respuesta la colocó en la variable ans. Observar que después esa misma operación se asignó a la variable c.

**El uso de %**

El carácter tanto por ciento (%) indica comienzo de comentario.
Cuando aparece en una línea de comandos, el programa supone que todo lo que va desde ese carácter hasta el fin de la línea es un comentario.

**Ejemplo 3**

```
>> a=5;
>> b=4;
>> c=(a*b)/2    % En c se coloca el área de un triángulo rectángulo con base a y altura b
c= 10
```

Lo anterior es equivalente a haber puesto:

```
>> a=5;
>> b=4;
>> c=(a*b)/2
c= 10
```

Pues lo que aparece luego del signo % en esa misma línea (O sea, En c se coloca el área de un triángulo rectángulo con base a y altura b ) el MATLAB no lo procesa, es un comentario útil para el que va a leer y entender qué representa esa operación.

**El uso de punto y coma (;) y de coma(,)**

La terminación en una línea de comandos con el carácter punto y coma (;) indica que no se visualice en pantalla el vector o matriz generada, o el resultado calculado. La coma permite separar dos comandos en una misma línea.

**Ejemplo 4**

En el Ejemplo 3, si se coloca

```
>> c=(a*b)/2;
```

No se visualiza el resultado obtenido en c. O sea, se calcula, pero no se puede ver cúal es.
Siguiendo el mismo ejemplo, es equivalente poner

```
>> a=5;
>> b=4;
```

A colocar ambos comandos en la misma línea separados por coma, o sea,

```
>> a=5;, b=4;
```

**Comando *format*:**

Esta función o comando del MATLAB cambia el formato de visualización de salida en la ventana de comandos al formato especificado. Las opciones más usadas son:
format: Visualiza los números con 4 cifras decimales, en el caso que no sea enteros
format long: Visualiza los números con 15 cifras decimales, en el caso que no sea enteros
format rat: Visualiza el número (si no es entero) como el cociente de 2 números enteros

**Ejemplo 5**

```
>> a=1/3;  % Se coloca en a al número fraccionario 1/3
>> a
a = 0.3333  % Por defecto el formato que se tiene es format
>> format long    % Se pasa al formato con 15 cifras decimales
>> a

a = 0.333333333333333

>> format rat  % Se pasa al formato como número racional
>> a
a = 1/3
>> format
>> pi        % El número π, que en MATALB se escribe pi)
ans = 3.1416
>> format long
>> pi
ans = 3.141592653589793
>> format rat
>> pi
ans = 355/113
```

Una observación importante: Son dos cosas distintas la cantidad de decimales conque opera los números y otra el formato de visualización. Para realizar operaciones con números no enteros, usa una cantidad considerable de cifras decimales, independiente del formato de visualización de resultados obtenidos.

**Operadores aritméticos (aplicables a matrices y números)**

+: Adición
-: Substracción
*: Multiplicación
/: División
^: Potencia
‘: Matriz transpuesta.

**Cómo importar al MATLAB tablas de datos en Excel o exportar al Excel matrices creadas en MATLAB**

En muchas ocasiones se puede tener almacenado en una hoja de Excel una información que utilizará en MATLAB como una matriz que es de gran tamaño como para introducirla de manera manual. Similarmente, puede obtener una matriz en MATLAB por determinadas operaciones realizadas, y la quiere guardar en una hoja de Excel.
Para ello se usan los comandos xlsread (importar al MATLAB) y xlswrite (exportar al Excel)

Hay que tener en cuenta que para invocar con el MATLAB tablas de Excel las mismas deben estar en la misma carpeta donde está usando el MATLAB.

Las sintaxis son:
A=**xlsread**('ejemplo.xlsx')

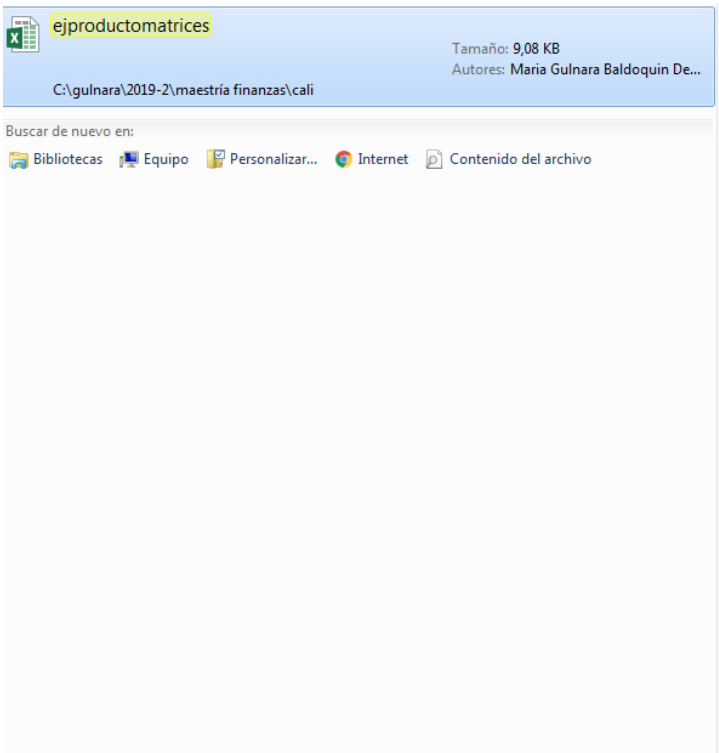
Esta función de MATLAB lee la primera hoja de trabajo en el libro de hojas de cálculo de Microsoft Excel llamado ejemplo y devuelve los datos numéricos en la matriz A.

A=**xlsread**('ejemplo.xlsx',p)
Esta función de MATLAB lee la hoja de trabajo especificada p (cuando no es la primera) en el libro de hojas de cálculo de Microsoft Excel llamado ejemplo y devuelve los datos numéricos en la matriz A.

respuesta=xlswrite(filename,A)
Esta función MATLAB escribe la matriz A en la primera hoja de trabajo con el nombre del archivo respuesta de la hoja de cálculo de Microsoft Excel que comienza en la celda A1.
respuesta=xlswrite(filename,A, p)
Esta función MATLAB escribe la matriz A en la hoja de trabajo especificada p con el nombre del archivo respuesta, de la hoja de cálculo de Microsoft Excel que comienza en la celda A1.

Las hojas de cálculo deben colocarse con su extensión y entre comillas.
Ilustraremos lo anterior con la función xlsread en el Ejemplo X

**Ejemplo 6**
En la Figura 4 aparece la búsqueda de la tabla de Excel llamada ejproductomatrices que tiene como se puede observar al lado información en las primeras 30 filas y 5 columnas. Notar que aparece en la carpeta C:\gulnara\2019-2\maestriafinanzas\cali, que es la misma carpeta donde se está trabajando con el MATLAB y que aparece declarada encima de la ventana de trabajo (Command Window) en la Figura 1




| | A | B | C | D | E | F |
|---|---|---|---|---|---|---|
| 1 | 4 | 3 | 2 | 2 | 4 | |
| 2 | 4 | 4 | 4 | 3 | 5 | |
| 3 | 3 | 3 | 3 | 4 | 2 | |
| 4 | 5 | 4 | 2 | 2 | 2 | |
| 5 | 5 | 2 | 3 | 4 | 4 | |
| 6 | 3 | 5 | 3 | 4 | 3 | |
| 7 | 3 | 5 | 2 | 2 | 2 | |
| 8 | 2 | 2 | 4 | 2 | 4 | |
| 9 | 5 | 4 | 3 | 5 | 3 | |
| 10 | 3 | 2 | 4 | 3 | 5 | |
| 11 | 5 | 2 | 5 | 3 | 3 | |
| 12 | 2 | 2 | 3 | 4 | 3 | |
| 13 | 2 | 3 | 5 | 4 | 2 | |
| 14 | 5 | 5 | 2 | 5 | 5 | |
| 15 | 3 | 3 | 5 | 3 | 2 | |
| 16 | 4 | 5 | 4 | 3 | 5 | |
| 17 | 3 | 3 | 4 | 4 | 4 | |
| 18 | 2 | 2 | 3 | 5 | 5 | |
| 19 | 3 | 5 | 3 | 5 | 5 | |
| 20 | 4 | 4 | 4 | 2 | 2 | |
| 21 | 3 | 3 | 2 | 4 | 5 | |
| 22 | 3 | 5 | 2 | 2 | 4 | |
| 23 | 2 | 4 | 2 | 3 | 4 | |
| 24 | 4 | 2 | 2 | 3 | 5 | |
| 25 | 5 | 5 | 3 | 5 | 4 | |
| 26 | 3 | 3 | 5 | 5 | 4 | |
| 27 | 3 | 3 | 5 | 3 | 3 | |
| 28 | 2 | 5 | 5 | 3 | 4 | |
| 29 | 5 | 5 | 3 | 3 | 3 | |
| 30 | 5 | 2 | 4 | 4 | 4 | |

Figura 4: Tabla de Excel llamada ejproductomatrices.xlsx

Se asigna a la matriz A los valores de dicha tabla, en igual orden.

>> A=xlsread('ejproductomatrices.xlsx')

**Como no se puso un punto y coma al final del comando, aparece en la ventana la matriz A que se tiene**

```
A =

     4     3     2     2     4
     4     4     4     3     5
     3     3     3     4     2
     5     4     2     2     2
     5     2     3     4     4
     3     5     3     4     3
     3     5     2     2     2
     2     2     4     2     4
     5     4     3     5     3
     3     2     4     3     5
     5     2     5     3     3
     2     2     3     4     3
     2     3     5     4     2
     5     5     2     5     5
     3     3     5     3     2
     4     5     4     3     5
     3     3     4     4     4
     2     2     3     5     5
     3     5     3     5     5
     4     4     4     2     2
     3     3     2     4     5
     3     5     2     2     4
     2     4     2     3     4
     4     2     2     3     5
     5     5     3     5     4
     3     3     5     5     4
     3     3     5     3     3
     2     5     5     3     4
     5     5     3     3     3
     5     2     4     4     4
```

Si se usa el comando size para saber el orden de una matriz (dos números, primero número de filas y luego número de columnas), se comprueba que tiene exactamente 30 filas y 5 columnas, según aparece en la tabla de Excel

>> t=size(A)
t = 30 5

## Algebra Lineal

### Creación de vectores en MATLAB

No hace falta declararlos o establecer de antemano su tamaño, el cual puede modificarse dinámicamente. Existen varias formas de definirlos, una de ellas:

1. Entre dos corchetes, los elementos separados por blancos o comas.

**Ejemplo 7**

Se crea el vector de longitud 4, v = [2 5 4 7]

```
>> v = [2 5 4 7]
v= 2 5 4 7
```

**Comando length**
La longitud de un vector v se determina con el comando *length*(v)
Para el Ejemplo 7:

```
>> length(v)
ans = 4
```

**Creación de matrices**
Igual que en el caso de los vectores, no hace falta declararlas o establecer de antemano su tamaño, el cual puede modificarse dinámicamente.
Las matrices se definen o introducen por filas; los elementos de una misma fila están separados por blancos o comas, mientras que las filas están separadas por caracteres punto y coma.

**Ejemplo** 8
Se crea una matriz de orden 3x3

```
>> A=[1 -2 3;0 7 -4;8 -1 9]
A =1   -2    3
    0    7   -4
    8   -1    9
```

**Tamaño (orden de una matriz)**
**Comando size**
Devuelve una matriz fila con dos valores: el primero es el número de filas de la matriz, siendo el segundo el número de columnas de la misma.
Sintaxis: size(A), siendo A la matriz declarada previamente.

**Ejemplo 9**

```
>> a=[2 3 -1 4;5 7 2 0;-1 8 6 10];
>> t=size(a)
t = 3     4
```

**Operaciones con matrices**

**Ejemplo 10**

```
>> a=[1 -1 3;2 1 0;-2 4 5],b=[-3 1 2;4 5 -1;6 0 -2]

a = 1   -1    3                b = -3    1    2
     2    1    0                      4    5   -1
    -2    4    5                      6    0   -2

>> c=2*a+b  % Dos operaciones juntas: Multiplicar una matriz por un número y su resultado sumado
                con otra matriz de igual tamaño
c =
   -1   -1    8
    8    7   -1
    2    8    8
```

```
>> c=a*b
c = 11    -4    -3
     -2     7     3
     52    18   -18
```

Observar en el Ejemplo 11 cómo si las matrices no tienen los tamaños adecuados para realizar las operaciones solicitadas, se envía un mensaje de error.

**Ejemplo 11**

```
>> b=[-3 1 2;4 5 -1;6 0 -2; 1 -1 4]
b =
    -3     1     2
     4     5    -1
     6     0    -2
     1    -1     4

>> a+b
Error using  +
Matrix dimensions must agree.
```

**Transpuesta de matrices**

En ocasiones es necesario trabajar con la matriz traspuesta de una dada, que puede ser obtenida, por ejemplo, de una tabla de Excel. Trasponer significa intercambiar sus filas con columnas. Para eso se utiliza el "apóstrofe" luego de la variable donde está la matriz.

**Ejemplo 12**

Suponga que tiene la matriz introducida de la siguiente manera

```
>> a=[3 5 2;4 1 6]
a =
     3     5     2
     4     1     6
```

Como se observa a es una matriz de orden 2x3, pero se necesita su matriz traspuesta 3x2. En lugar de volverla a introducir basta obtenerla de la siguiente manera:

```
>> at=a'
at =
     3     4
     5     1
     2     6
```

**Solución de sistemas de ecuaciones lineales (SEL)**
**Comando *rank***
**rank(a) calcula el rango de la matriz a**

Es útil por ejemplo, para clasificar el SEL y saber si tiene solución única o infinitas soluciones, para lo cual se calculan el rango de la matriz y matriz ampliada del sistema con dicho comando.

**Ejemplo 13**
El dueño de un bar compró gaseosas, cerveza y vino por un total de $500, sin considerar impuestos. El valor del vino fue de $60 menos que el total de lo invertido en gaseosa y cerveza. Se conoce que las gaseosas, cerveza y vino pagan un impuesto de 6%, 12% y 30% respectivamente. Si la factura total con impuesto fue de $592.40 calcule la cantidad invertida en cada tipo de bebida.

Modelo matemático:
Variables:
x,y,z: $ invertidos en gaseosas, cerveza y vino, respectivamente.

x + y+ z = 500  Dinero invertido sin impuesto
z = x+y – 60  Vino $60 menos que otras bebidas
1.06x+1.12y+1.3z=592.4  Factura total con impuesto

Reorganizando el sistema:
x + y+ z = 500  Dinero invertido sin impuesto
x+y – z =60  Vino $60 menos que otras bebidas
1.06x+1.12y+1.3z=592.4  Factura total con impuesto

Matrices del sistema en el MATLAB:
```
>> a=[1 1 1;1 1 -1;1.06 1.12 1.3];, b=[500;60;592.4];
>> ab=[a b]; % ab es la matriz ampliada, basta colocar al lado de a la columna b
>> rank(a)
ans = 3
>> rank(ab)
ans = 3
```

Como el rango de la matriz del sistema es igual al de la ampliada tiene soluciones, y como coincide con el número de variables sabemos que el sistema tiene solución única.

**Sistemas con una única solución**
a. En este caso el comando más adecuado será X=a\b, donde X es un vector columna donde aparecerán los valores de las soluciones para las variables del sistema de ecuaciones (en forma matricial) aX=b

**Ejemplo 14**
Resolviendo el Ejemplo 13:
```
>> X=a\b

X =
  120.0000
  160.0000
  220.0000
```

De la respuesta, tenemos que x=120, y=160, z=220, o sea, invirtió (sin considerar impuestos) $120 en gaseosas, $160 en vino y $220 en cerveza.

b. **Comando *rref***

Otra forma de resolver el sistema es usando el comando rref a la matriz ampliada. Con ello ya da despejado los valores de x,y,z, pues se obtiene la matríz idéntica con la última columna el valor de las soluciones, donde cada columna corresponde a una variables con exactamente un valor 1 y el resto cero. Siguiendo el mismo Ejemplo 13:

```
>> rref(ab)
ans =
    1    0    0  120
    0    1    0  160
    0    0    1  220
```

O sea, x=120, y=160, z=220

**Sistemas con infinitas soluciones**

En este caso la mejor forma es usar el comando solve para resolver ecuaciones o sistemas de ecuaciones, pudiendo definir cuál (cuáles) serán las variables libres que deseamos.

Cuando hay infinitas soluciones, el comando X=a\b, le da una sola de las infinitas, no siendo quizás la que uno desearía escoger.

**Ejemplo 15**

Resolver el SEL

$$x - y + 3z = 4$$
$$2x - y - z = 6$$
$$3x - 2y + 2z = 10$$

Primero comprobamos que es un sistema indeterminado calculando los rangos de la matriz del sistema y el de la ampliada.

```
>> a=[1 -1 3;2 -1 -1;3 -2 2];, b=[4;6;10];
>> ab=[a b];
>> rank(a)
ans = 2
>> rank(ab)
ans = 2
```

Como el rango de la matriz del sistema  es igual al de la ampliada el sistema tiene soluciones. Pero al ser el rango 2 y el número de variables 3, sabemos que el sistema tiene infinitas soluciones con una variable libre (n-r(a)=3-2=1).

Para resolverlo con MATLAB asumamos que z es la variable libre.

```
>> [x y]=solve(x-y+3*z==4,2*x-y-z==6,3*x-2*y+2*z==10,[x y])
x = 4*z + 2
y = 7*z - 2
```

Observar en la solución general que los valores de x,y dependen del valor que se le dé a la variable libre z.

Notar el resultado si hubiéramos usado el comando X=a\b
>> X=a\b
Warning: Matrix is close to singular or badly scaled. Results may be inaccurate. RCOND = 6.167906e-18.
X =
-1.0000
-7.2500
-0.7500

El Warning en la respuesta le dice que esta no es la mejor opción.
Le da una única solución, la que corresponde al valor de z = – 0.75, probablemente no le interese soluciones con valores negativos.

**Determinante de una matriz**
El determinante es un número asociado a matrices que son cuadradas. Tiene múltiples aplicaciones, y en el contexto de los temas que abordamos se utiliza cuando se quiere determinar si una función es cóncava o convexa, por ejemplo.

El comando a usar es det(A), donde A es la matriz a buscar su determinante.

**Ejemplo 16**
>>A=[2 1 -1;3 1 0;1 4 2];
>> d=det(A)
d = -13

**Creación de funciones matemáticas en MATLAB**

**Funciones de una variable**
**Comando syms**
Genera variables simbólicas, una o más, colocándolas luego de syms separadas por espacios en blanco

**Ejemplo 17**
Plantee en MATLAB la función **p(x)=10-2x que expresa el precio (p) de un producto, en función de la demanda (x).**

>> *syms* x % Define a x como variable simbólica
>> p=10-2*x; % Asigna a p la expresión 10-2*x

Observar que no es necesario colocar
>> p(x)=10-2*x;
Pues al declararse la variable x previamente y aparecer en la expresión de p, se asume que p depende de x, aunque no da error si lo hace.

**Comando *subs*:**

Es útil para evaluar una función para valores de las variables de la que depende (válido para funciones de una o varias variables).

Si queremos obtener f(a) la sintaxis sería subs(f,a)

**Ejemplo 18**
Usando la función del Ejemplo 17 (p(x)=10-2x) evaluar precio si la demanda es de 3.
Se debe obtener p(3)

```
>> syms x    % Se define la variable
>> p = 10-2*x;    % Se define la función
>> r=subs(p,3)   % Se coloca en r el valor de p(3)
r = 4
```

**Comandos para graficar funciones**
Existe una variada cantidad, se describen 3 de ellos que puede utilizar de manera fácil y rápida

- ✓ **Comando ezplot**

Grafica una función simbólica en el intervalo deseado.
Si se quiere graficar la función f(x) en el intervalo [a,b], luego de definirla se usa la sintaxis del comando ezplot(f,a,b)
Si utilizara la sintaxis ezplot(f), por defecto el MATLAB grafica a f en el intervalo $[-2\pi, 2\pi]$

- ✓ **Comando grid**

Se combina con el comando ezplot para hacer un "enrejado" del área del gráfico con líneas horizontales y verticales que permitan distinguir mejor coordenadas de puntos de la curva

Su sintaxis es simplemente grid
Si se quiere seguir haciendo gráficos sin dicho enrejado se coloca grid off

**Ejemplo 19**
Graficar la función $f(x)=x^3-x^2-2x$ en el intervalo [-2,3]

```
>> syms x
>> f=x^3-x^2-2*x;
>> ezplot(f,-2,3)   % Puede demorar algunos segundos en desplegar el gráfico (Figura 5)
>> grid
```

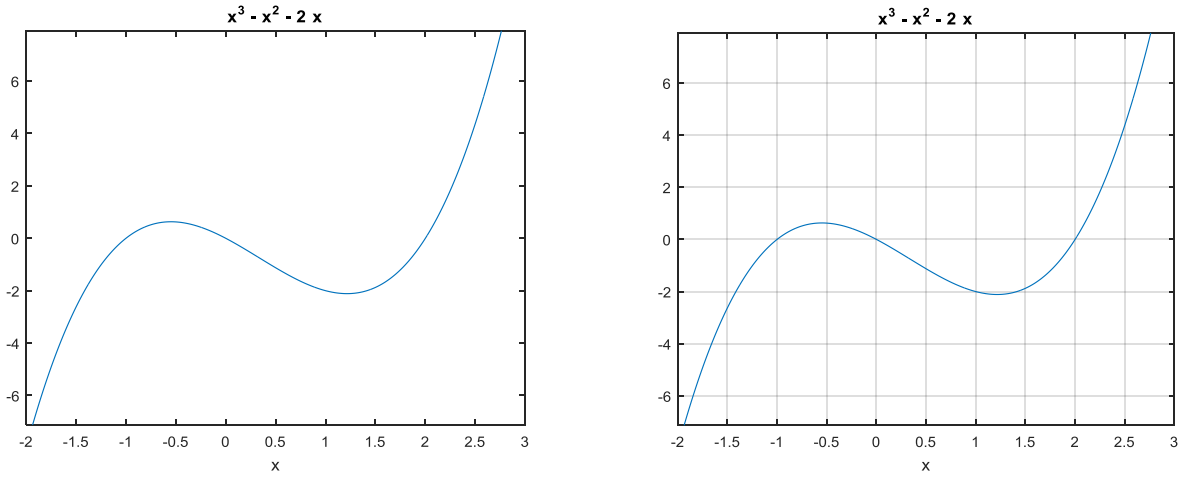



**Figura 5: Gráfico de una función sin enrejado (izquierda) y con enrejado (derecha)**

✓ **Comando hold on**

Este comando es adecuado cunado se quiere ver el resultado del gráfico de más de una función en el mismo gráfico. Es útil por ejemplo, para visualizar el punto de equilibrio (ocurre en un precio cuando la cantidad demandada es igual a la cantidad ofrecida) teniendo en cuenta las funciones de oferta y de demanda.

Basta colocar hold on luego de haber graficado la primera función, una única vez, sea que grafique dos o más funciones y quiere visualizarlas en el mismo gráfico.

Para seguir trabajando haciendo gráficos de funciones una función en cada gráfico coloca el comando hold off

**Ejemplo 20**

Determine de manera gráfica la cantidad de equilibrio de las leyes de la oferta y la demanda siguientes:

$$pd = 25-2x$$
$$po = 3x+5$$

```
>> pd= 25-2*x;
>> ezplot(pd,0,10)
>> hold on
>> po=3*x+5;
>> ezplot(po,0,10)
>> grid
```

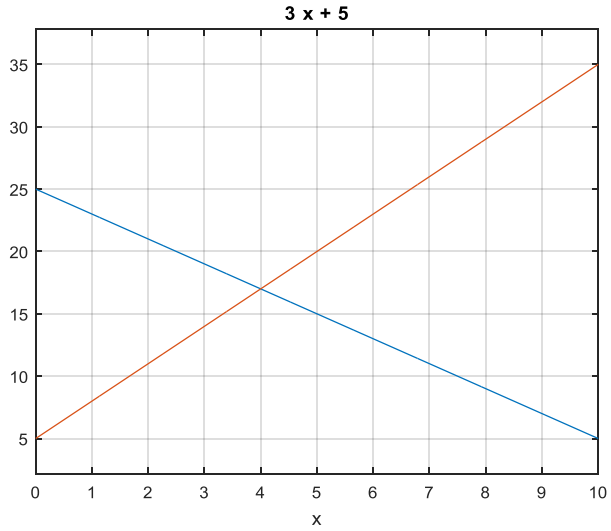



**Figura 6: Gráfico de dos funciones**

En la Figura 6 en rojo aparece graficada la función oferta y en azul la función demanda. En la parte de arriba aparece la primera función dibujada, que hay opciones en la pantalla del gráfico para cambiarlo, borrarlo, etc. De manera gráfica se vé que la cantidad de equilibrio es cuando x= 4. El precio de equilibrio gráficamente se observa está entre 15 y 20, y sustituyendo x=4 en cualquiera de las ecuaciones se puede verificar que es exactamente 17.

**Comando plot**

Se utiliza para representar una función no de manera simbólica, sino explicitando un conjunto de puntos (x,y) de la función que se van uniendo entre ellos. Se recomienda para su mejor visualización que dos puntos x consecutivos estén muy cerca entre sí.

Sintaxis:

x=vi:p:vf; donde vi es el valor inicial del intervalo del dominio de la función donde se quiere graficar la misma, p es la "distancia" entre dos puntos consecutivos, vf el valor final de la variable donde se quiere graficar.

**Ejemplo**:

Graficar $y=x^2$, en el intervalo [-1,1]

Lo ejemplificamos con el gráfico de las retenciones según salario (expresados ambos en número de UVT) y que aparece en la Tabla 1:

| Rangos en UVT | | Tarifa marginal | Impuesto |
|---|---|---|---|
| Desde | Hasta | | |
| > 0 | 87 | 0% | 0 |
| > 87 | 145 | 19% | (Ingreso laboral gravado expresado en UVT menos 87 UVT) X 19% |
| > 145 | 335 | 28% | (Ingreso laboral gravado expresado en UVT menos 145 UVT) X 28% más 11 UVT |
| > 335 | 640 | 33% | (Ingreso laboral gravado expresado en UVT menos 335 UVT) X 33% más 64 UVT |
| > 640 | 945 | 35% | (Ingreso laboral gravado expresado en UVT menos 640 UVT) X 35% más 165 UVT |
| > 945 | 2.300 | 37% | (Ingreso laboral gravado expresado en UVT menos 945 UVT) X 37% más 272 UVT |
| > 2.300 | En adelante | 39% | (Ingreso laboral gravado expresado en UVT menos 2.300 UVT) X 39% más 773 UVT |

Tabla 1: Impuesto según salarios (en UVT) 2019

```
>> x1=0:0.1:87;    %  x1=[0 0.1 0.2 0.3 0.4 0.5    ….   87]  x1 matriz  fila con 871 elementos
>> yx1=0*x1;       %  yx1=[0  0  0  0  0  ….   0]  x1 matriz  fila con 871 elementos
>> x2=87.1:0.1:145;
>> yx2=(x2-87)*0.19;
>> x3=145.1:0.1:335;
>> yx3=(x3-145)*0.28+11;
>> x4=335.1:0.1:640;
>> yx4=(x4-335)*0.33+64;
>> plot(x1,yx1,'b')
>> hold on
>> plot(x2,yx2,'g')
>> plot(x3,yx3,'r')
>> plot(x4,yx4,'c')
>> grid
```

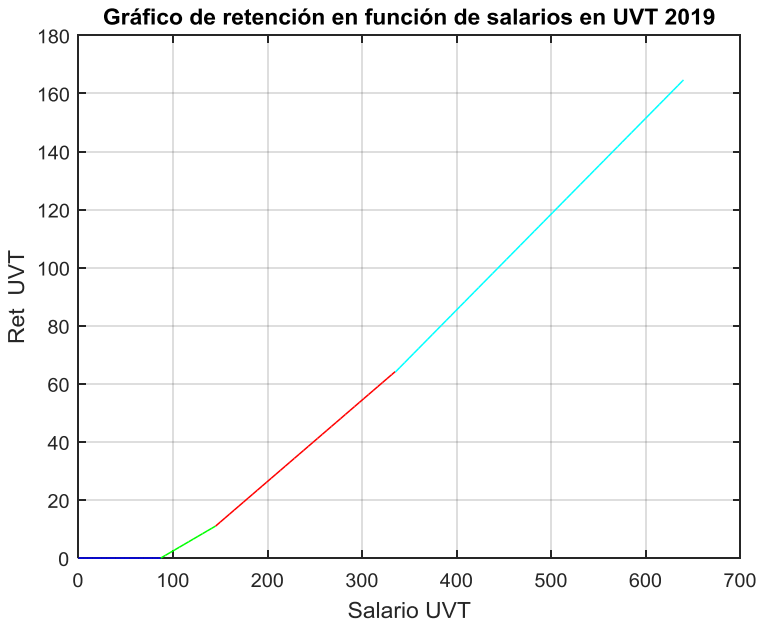



Figura 7: Representación del comportamiento de retención en fuente función "lineal a trozos"

**Comando *solve*:**

Uno de sus usos es resolver ecuaciones, en particular si se coloca solve(f), obtiene los valores de la variable donde f(x)=0, que gráficamente sería los puntos de la curva que representa a f que cortan al eje x.

**Ejemplo 21**

Determinar los puntos  donde la curva de la función del Ejemplo 19 ($f(x)=x^3-x^2-2x$) corta al eje x.

Lo que hay que obtener son los valores de x tal que $f(x)=x^3-x^2-2x = 0$

```
>> syms x
>> f=x^3-x^2-2*x;
>> sol=solve(f)
sol =
 -1
```

0
2

Observar que la ecuación es de tercer grado, luego tiene 3 raíces, que en este caso son -1, 0 y 2.
Puede comprobar en el gráfico de la Figura 4 cómo la curva corta al eje x justo en estos puntos.

**Comando *double*:**
Muchos comandos del MATLAB devuelven constantes de manera simbólica, de forma que no es fácil identificar el número real que representa. En esos casos se coloca el comando double y entre paréntesis el comando que quiere utilizar.

Por ejemplo, usemos el comando solve para resolver una ecuación donde las raíces (soluciones) no son números enteros.

**Ejemplo 22**
Resolver la ecuación

```
>> f=2*x^3-x^2-2*x;
>> sol=solve(f)
sol =                0
 1/4 - 17^(1/2)/4
 17^(1/2)/4 + 1/4
```

Observar que dos de las raíces no es fácil identificar qué números reales son, pues viene en función de operaciones aritméticas, incluidas raíces cuadradas de números.

Colocando previo al comando solve(f) a double:

```
>> sol=double(solve(f))
sol =
        0
   -0.7808
    1.2808
```

Ya se puede ver claramente cuáles son las 3 raíces numéricas de la ecuación.

**Comando diff**
Al utilizar la sintaxis diff(f), se devuelve la función derivada de la función f, previamente definida.

**Ejemplo 23**
Dada la función $f(x) = 2x^3-x^2-2x$, calcular $f\,'(x)$

```
>> f=2*x^3-x^2-2*x;
>> df=diff(f)
df = 6*x^2 - 2*x - 2
```

La función derivada de f se colocó en df y es $f\,'(x) = 6x^2-2x-2$

**Comando int**
Permite calcular integrales definidas e indefinidas, dependiendo de la sintaxis
F=int(f)

Coloca en F la función integral de f, previamente definida
S=int(f,a,b)
Coloca en S un número, el valor de la integral definida de f en el intervalo [a,b]

**Ejemplo 24**
Calcular
a. $F = \int (3x^3 - x^2 + x - 1)dx$
b. $S = \int_1^3 (3x^3 - x^2 + x - 1)dx$

```
>> f=3*x^3-x^2+x-1;
>> F=int(f)
F = (3*x^4)/4 - x^3/3 + x^2/2 - x
>> S=int(f,1,3)
S =
160/3
>> S=double(int(f,1,3))
S =
53.3333
```

**Funciones de varias variables**
**Comando syms**
Genera una o más variables simbólicas, luego en el caso de funciones de varias variables se coloca detrás de syms todas las variables de las que depende la función separadas por un espacio en blanco.

**Ejemplo 25**
(Función de costo) Una empresa elabora dos productos, A y B. El costo de los materiales y de la mano de obra es de $4 por cada unidad del producto A y de $7 por cada unidad de B. Los costos fijos son de $1500 por semana. Exprese el costo semanal C en términos de las unidades de A y B producidas cada semana.

Si se denotan x,y como las unidades del producto A y y unidades del producto B que se elaboran cada semana, respectivamente, entonces los costos de mano de obra y materiales para los dos tipos de productos son 4x y 7y dólares, respectivamente. Así que el costo C (en dólares) está dado por
C(x,y)= Costos de mano de obra y materiales +Costos fijos = 4x + 7y + 1500.

Expresado en MATLB:

```
>> syms x y
>> C= 4*x+7*y+1500
```

**Comando subs**
En este caso como la función depende de varias variables, hay que especificar el conjunto de variables independientes que se evalúan, seguido del conjunto de valores que se le debe dar a cada una, en igual orden.
Sintaxis: subs(f,{var1,var2,…,varp},{valor1,valor2,…,valorp})

**Ejemplo 26**
Sea la función del Ejemplo 25 y se quiere saber cuál es el costo incurrido cuando x=5 y y=8

```
>> V=subs(C,{x,y},{5,8})
V = 1576
```

**Comando diff**
En este caso como la función depende de varias variables, hay que especificar respecto a qué variable se quiere derivar, dando lugar a la derivada parcial correspondiente.

Sintaxis:
diff(f,x): Da la función $\partial f/\partial x$
diff(f,2,x): Da la función $\partial^2 f/\partial x^2$ (Segunda derivada parcial de f respecto a x)

**Ejemplo 27**
La función de costos conjuntos de una compañía que elabora dos tipos de productos x, y está dada por $C(x,y)= 0.1x^2+0.5y^2+4xy+2000$

Encuentre usando el MATLAB los costos marginales cuando se producen 500 unidades del producto x y 100 del producto y.

```
>> syms x y
>> C=0.1*x^2+0.5*y^2+4*x*y+2000;
>> Cx=diff(C,x)   % En Cx aparece la función derivada parcial ∂C/∂x
 Cx =  x/5 + 4*y

>> Cy=diff(C,y)   % En Cy aparece la función derivada parcial ∂C/∂y
Cy = 4*x + y

>> v1=subs(Cx,{x,y},{500,100})  % En v1 aparece derivada parcial ∂C/∂x evaluada en x=500, y=100
v1 = 500

>> v2=subs(Cy,{x,y},{500,100})  % En v2 aparece derivada parcial ∂C/∂y evaluada en x=500, y=100
v2 = 2100
```

**Comando hessian**
Calcula la matriz hessiana de una función de varias variables.
Sintaxis: hessian(f)

**Ejemplo 28**
Determine usando comandos del MATLAB adecuados si la función $f(x,y) = 2x^2 +2xy +3y^2$ es cóncava o convexa

```
>> syms x y
>> f=2*x^2+2*x*y+3*y^2;
>> H=hessian(f)   % Se calcula la matriz hessiana de f asignándola a H
H =
[ 4, 2]
[ 2, 6]

>> d=det(H)   % Se calcula el determinante de H (matriz hessiana de f)
```

d = 20

Como los elementos de la diagonal principal son positivos (no importa los valores de las variables x,y) y el determinante del hessiano también siempre es postivo (una constante positiva) entonces la función es convexa.

Observación:
Note que por ejemplo, según la matriz hessiana, $\partial^2 f/\partial x^2 = 4$, $\partial^2 f/\partial y^2 = 6$

Comprobemos lo anterior usando el comando diff para calcular esas derivadas parciales de orden 2:
>> d2x=diff(f,2,x)
d2x = 4

>> d2y=diff(f,2,y)
d2y = 6

**Ejemplos de por qué los software no deben ser una "caja negra"**

**1. La modelación de los problemas no las hacen los software**

Las empresas plantean problemas "verbalmente", ninguna le dice explícitamente el modelo de Optimización a resolver, o la integral, la ecuación, sistema de ecuaciones que debe resolver. Los software, al menos hasta el momento, NO modelan, hay que darle explícitamente los modelos a resolver.

Por lo anterior, la competencia de modelar es necesaria para resolver un problema donde la herramienta computacional a usar le exige el modelo que lo representa.

**2. En la solución de problemas de Optimización**

Salvo que sean modelos de Optimización Lineal los software (como el MATLAB) solo se comprometen a dar una solución que sea un óptimo "local", y dependen en general del punto inicial de búsqueda.

¿Cómo podemos saber si el óptimo es global o no, por ejemplo?

La teoría dada respecto a condiciones necesarias y suficientes de óptimos globales ayuda a responder la pregunta, y para ello el MATLAB cuenta con comandos que debemos conocer, a partir de esa teoría.

Para funciones de una variable: segunda derivada de la función a optimizar (diff(f,2))

Para funciones a optimizar de varias variables no lineales: el hessiano de la función a optimizar o función lagrangeana (hessian(f)) así como el determinante de submatrices de la matriz hessiana usando el comando det.

**3. En la solución de sistemas de ecuaciones lineales**

Hay sistemas de ecuaciones lineales cuya solución es muy sensible a mínimos cambios en los datos, y debemos saber cuándo eso puede suceder.

Consideramos el siguiente sistema de 2 ecuaciones y 2 incógnitas.
2.3x+1.2y=0.99
4.4x+2.3y=1.89

Resolviendo con MATLAB:
```
>> A=[2.3 1.2 ; 4.4 2.3];,b=[0.99; 1.89];
>> X=A\b
X =
    0.9000
   -0.9000
```

Veamos de qué manera un pequeño cambio en el término independiente afecta a la solución de este sistema. Modifiquemos levemente el sistema sustituyendo los términos independientes 0.99 por 1 y 1.89 por 1.9, o sea, una diferencia de 0.01 en ambos. Nuevo sistema:

2.3x+1.2y=1
4.4x+2.3y=1.9

Resolviendo nuevamente con MATLAB:
```
>> A=[2.3 1.2 ; 4.4 2.3];,b=[1; 1.9];
>> X=A\b
X =
    2.0000
   -3.0000
```

La diferencia en solución es muy grande comparada con cambios hechos.

El sistema planteado se clasifica como **sistema mal condicionado**.
Problema mal condicionado: Pequeñas modificaciones en x producen grandes modificaciones en su respuesta.
Problema bien condicionado: pequeñas modificaciones en x producen pequeñas modificaciones en su respuesta.

¿Cómo detectar un sistema mal condicionado?

Hallando el número de condición de la matriz A, que se define como la norma de la matriz A por la norma de su matriz inversa, o sea, si se denota por $\kappa(A)$, entonces:
$\kappa(A) = ||A||. ||A^{-1}||$ siendo $||A||$ la norma de la matriz A.

Se cumple que $\kappa(A) \geq 1$, y si $\kappa(A)$ es un número próximo a 1 se dice que A es una matriz bien condicionada; y si es mucho mayor que 1, que es mal condicionada.

Busquemos el número de condición de la matriz con el comando apropiado del MATLAB:

Sintaxis:
cond(A)  Respecto a la norma 2

cond(A,p)  Respecto a la norma p (p=1,2,’inf’, 'fro'. la norma inf es la norma infinito y la norma 'fro' es la de Frobenius)

Para el ejemplo visto:
>> cond(A)
ans = 3.1380e+03

O sea, el número de condición de la matriz es 3138, muchísimo mayor que 1, luego el sistema está mal condicionado.

El número de condición de una matriz mide la sensibilidad de la solución de un sistema de ecuaciones lineales a errores en los datos.

El número de condición de una matriz depende de la norma usada, pero si un sistema está mal condicionado, por ejemplo, con cualquier norma usada será un número alto. En el ejemplo visto:

```
>> cond(A,1)
ans = 4.4890e+03   % O sea, cond(A,1) = 4489

>> cond(A,inf)
ans = 4.4890e+03    % O sea, cond(A,inf) = 4489

>> cond(A,'fro')
ans = 3.1380e+03    O sea, cond(A, 'fro') = 3138
```