

USO DE SOFTWARE ESPECIALIZADO

CÁLCULO NUMÉRICO

INTRODUCCIÓN

IMPORTANCIA DEL MATLAB, OCTAVE Y SCILAB
COMANDOS BÁSICOS Y PRINCIPALES

CONTENIDO

<i>Título</i>	Programación para los métodos numérico
<i>Duración</i>	240 minutos
<i>Información general</i>	Resaltar la importancia de la programación de los Métodos Numéricos para resolver problemas de la Ingeniería.
<i>Objetivo</i>	Conocer, aplicar los comandos básicos y de programación de Matlab, Octave y Scilab.

SUBMATRICES

Trabajemos con la matriz mágica A de orden (6x6)

```
Command Window
>> A=magic(6)

A =

    35     1     6    26    19    24
     3    32     7    21    23    25
    31     9     2    22    27    20
     8    28    33    17    10    15
    30     5    34    12    14    16
     4    36    29    13    18    11
```

A(2, 6) filas, columnas

* A2=A(2:5, 3:4)

Se quiere extraer de la matriz A la submatriz A1, que corresponden a las filas 3, 4 y todas las columnas.

Se puede generar de 3 formas:

```
>> A1=A([3,4], [1 2 3 4 5 6])  Cuando no son consecutivos

A1 =

    31     9     2    22    27    20
     8    28    33    17    10    15  FORMA 1

>> A1=A(3:4, 1:6)  Cuando son consecutivos

A1 =

    31     9     2    22    27    20
     8    28    33    17    10    15  FORMA 2

>> A1=A(3:4, :)

A1 =

    31     9     2    22    27    20
     8    28    33    17    10    15  FORMA 3
```

Matrices por Bloques

Las siguientes matrices se pueden considerar Bloques

```
Command Window
>> A11=[1 2;3 4];
>> A12=eye(2);
>> A21=[-1 0;0 -1];
>> A22=ones(2);
>>
```

Con estas matrices se puede formar La matriz por Bloques A

```
>> A=[A11 A12;A21 A22]

A =

     1     2     1     0
     3     4     0     1
    -1     0     1     1
     0    -1     1     1
```

Ejercicio

Resolver el sistema lineal de ecuaciones

$$\begin{cases} 5x_1 - 2x_2 + x_3 = 4 \\ 2x_1 + 4x_2 - 3x_3 = -6 \\ x_1 - x_2 + 2x_3 = 1 \end{cases}$$

Sistema en puesto matricial
 $Ax=B$
 multiplicamos por la inversa
 $\text{Inv}(A) \cdot A \cdot X = \text{Inv}(A) \cdot B$
 $X = \text{Inv}(A) \cdot B$

Generamos las matrices A de coeficientes del sistema y B de términos independientes, en Matlab y resolvemos el sistema:

```

Command Window
>> A=[5 -2 1;2 4 -3;1 -1 2];
>> B=[4 -6 1];
>>
>> x=inv(A)*B'      B transpuesta

x =

    0.1212
   -1.9697
   -0.5455

>> % Otra forma
>>
>> X=A\B'

X =

    0.1212
   -1.9697
   -0.5455

fx >>

```

Ejercicio

$x = [9 \ 8 \ 7; 4 \ 5 \ 6; 7 \ 8 \ 9]$

Dada la matriz $x = \begin{pmatrix} 9 & 8 & 7 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$. En no más de 2 líneas de ejecución generar la matriz aplicando submatrices

$$y = \begin{pmatrix} 9 & 8 & 7 \\ 8 & 5 & 8 \\ 7 & 8 & 9 \end{pmatrix}$$

$y=[x(1,:);x(:,2)';x(3,:)]$

Fila columna transpuesta

Otros Operadores Aritméticos

- **.*** Producto escalar de vectores
- **** $A \setminus B = \text{inv}(A) * B$, A y B matrices
- **./** $A ./ B = [B(i,j)/A(i,j)]$; siendo A, B vectores de R^n

Ejercicio.-

1. Dada una matriz M cuadrada aleatoria uniforme de orden 3, obtener su inversa, su M' transpuesta y su diagonal. Transformarla en una matriz triangular inferior y en otra superior y rotarla 90 grados. Obtener la suma de los elementos de la primera fila y la suma de los elementos de la diagonal. Extraer la submatriz cuya diagonal son los elementos a_{11} y a_{22} y extraer también la submatriz cuyos elementos de la diagonal son a_{11} y a_{33} .

Triangular inferior
 $T_{\text{inf}} = \text{tril}(M)$

Triangular Superior
 $T_{\text{sup}} = \text{triu}(M)$

Rotar la matriz 90 grados en sentido antihorario
 $M_{\text{rot90}} = \text{rot90}(M)$

Suma de los elementos de la primera fila
 $\text{sum}(M(1,:))$

Suma de la diagonal
 $\text{sum}(\text{diag}(M))$

Extraer la submatriz cuya diagonal son los elementos 11 y 22
 $M(1:2,1:2)$

extraer también la submatriz cuyos elementos de la diagonal son 11 y 33
 $M([1 \ 3], [1 \ 3])$

2. Dada la matriz $M = \begin{pmatrix} i & 2i & 3i \\ 4i & 5i & 6i \\ 7i & 8i & 9i \end{pmatrix}$, obtener su logaritmo neperiano elemento a elemento y realice las operaciones matrices e^M y $\ln(M)$.
3. Hallar la matriz diferencia entre una matriz aleatoria cuadrada de orden 4 y una matriz aleatoria normal de orden 4 (consultar para normal). Calcular la transpuesta y la inversa de la citada diferencia.

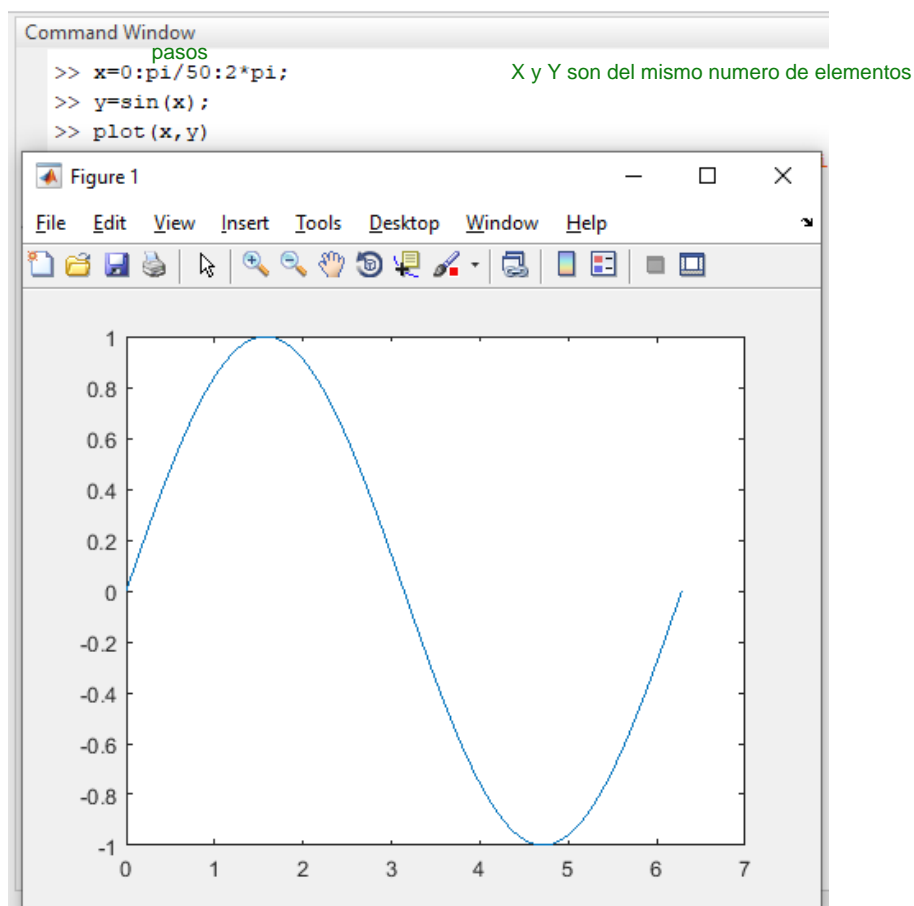
GRÁFICAS DE FUNCIONES EN R²

Matlab permite representar funciones de una y dos variables en:

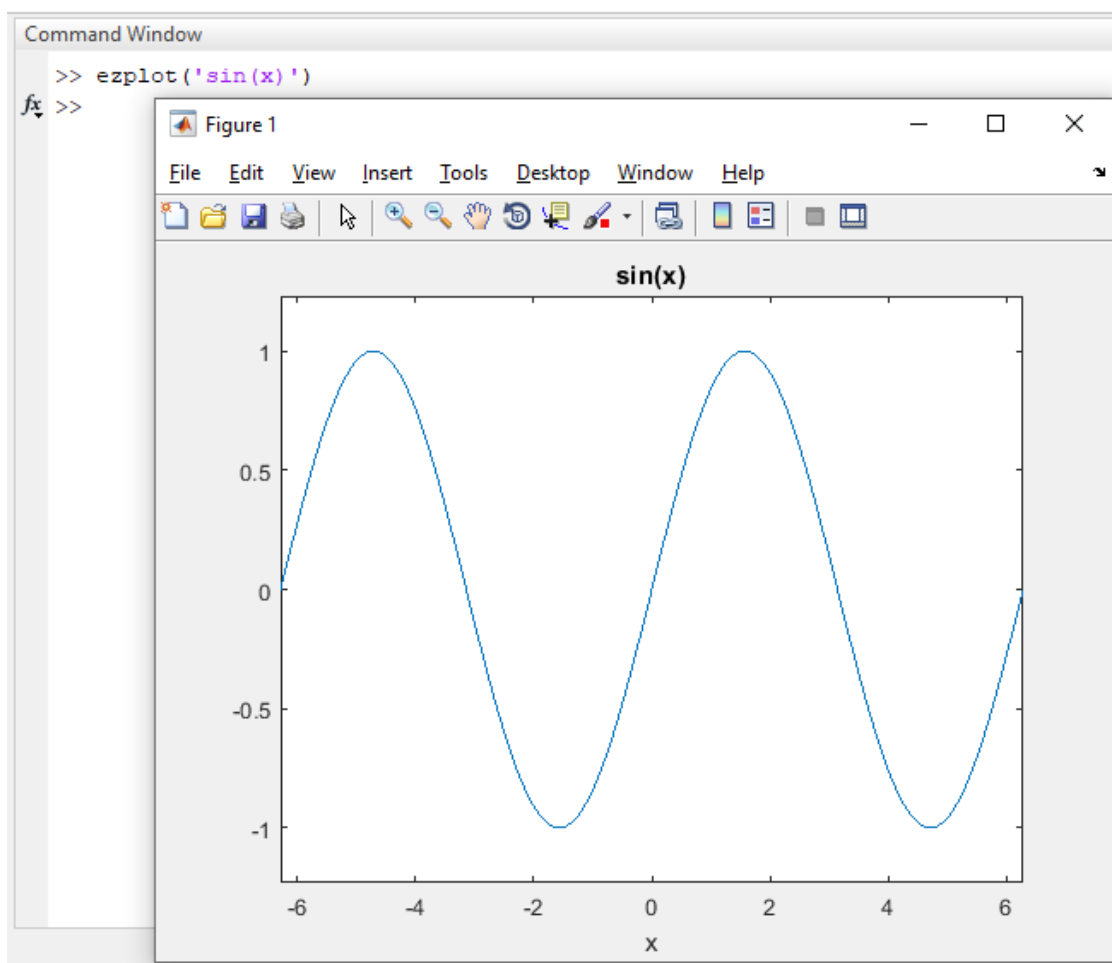
- Coordenadas cartesianas
- Coordenadas polares
- Ecuaciones paramétricas

Gráfica de la función $y = \sin x$ en el intervalo $[0, 2\pi]$

FORMA 1



FORMA 2



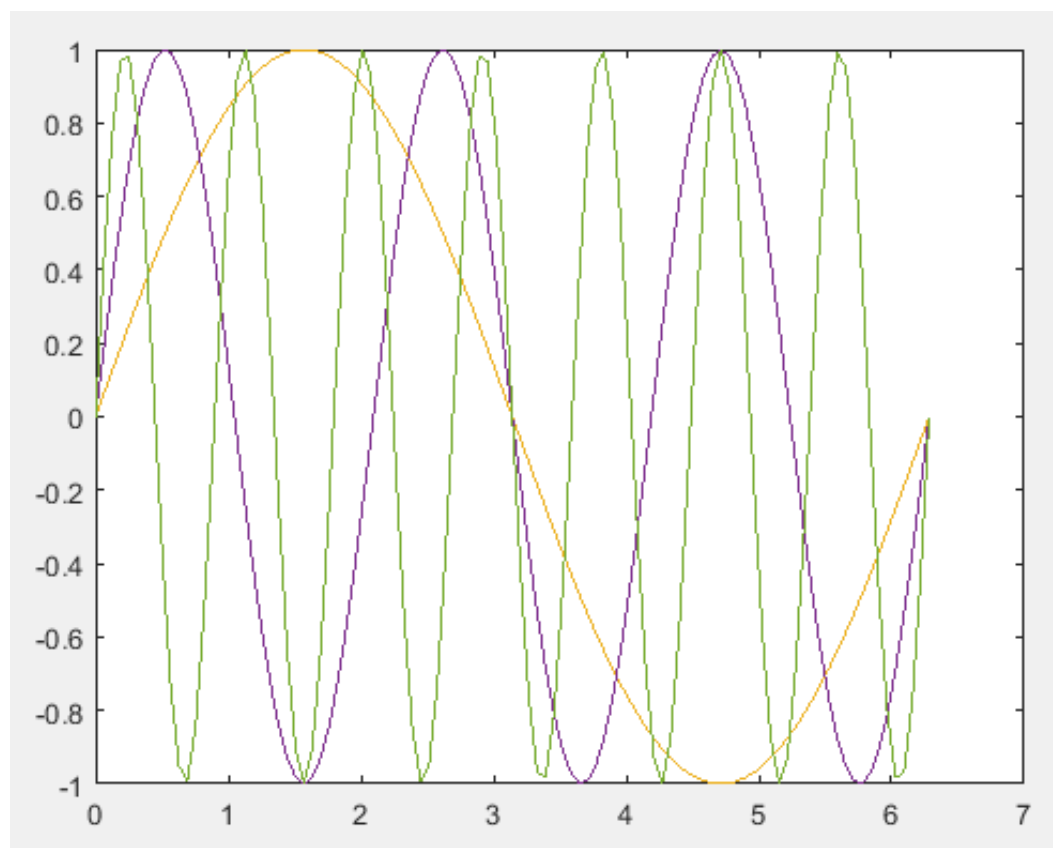
Añadir títulos y textos

- axis → Permite cambiar la escala del eje x y del eje y
- xlabel → Poner título en el eje x
- ylabel → Poner título en el eje y
- title → Coloca un título al gráfico
- grid → Produce un mallado en el plano
- text → Coloca un texto en una coordenada dada

En la pantalla gráfica de Matlab, que aparece cuando se hace un gráfico, tenemos numerosos comandos que permiten modificar el aspecto de la gráfica, salvarla, imprimirla

Varias funciones en la misma gráfica

```
Command Window
>> x = 0:pi/50:2*pi;
>> y = sin(x); z = sin(3*x); u = sin(7*x);
>> plot(x,y,'b',x,z,x,u)
>>
>> hold on -> La grafica se super pone en la otra grafica
>>
>> x = 0:pi/50:2*pi;
>> Y = [sin(x); sin(3*x); sin(7*x)];
>> plot(x,Y)
>>
>> hold off -> cierra la superposicion
>> |
```



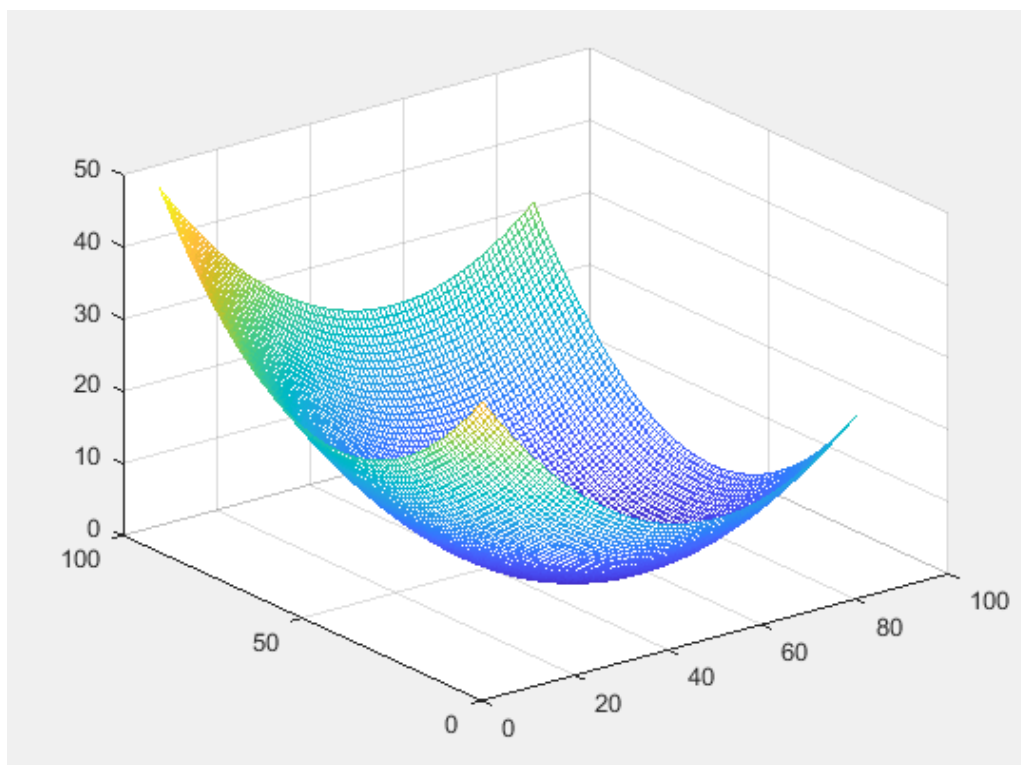
GRÁFICAS DE FUNCIONES EN R^3

`plot3` crea una gráfica lineal tridimensional

```
x=linspace(0,10*pi,1000);
>> y=cos(x);
>> z=sin(x);
>> plot3(x,y,z)
>> comet3(x,y,z)
```

```
Command Window
>> [X Y]=meshgrid(-5:0.1:3,-4:0.1:5);
>> z=X.^2+Y.^2;
>> mesh(z)
fx >> |
```

```
x=-10:0.01:10;
>> y=-10:0.01:10;
>> z=x.^2+y.^2;
>> plot3(x,y,z)
```



Otros comandos que se pueden utilizar:

```
>> surf(z)
>> surfc(z)
>> meshc(z)
>> meshz(z)
```

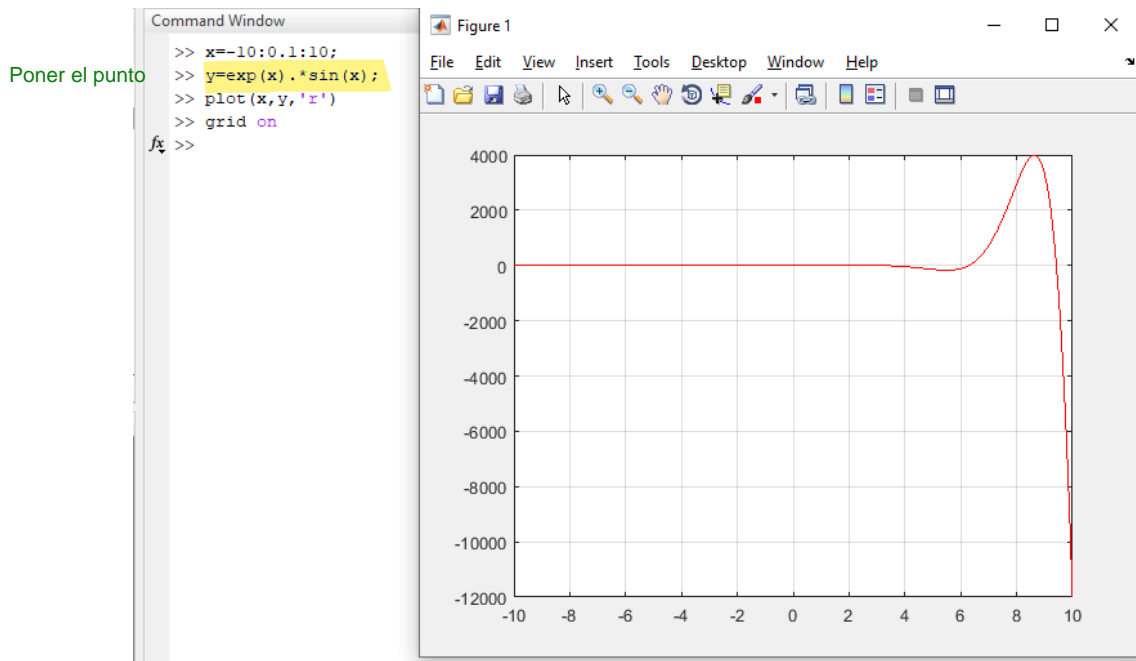
Ejercicio:
crear una gráfica de superficie para
 $z = x^2 + y^2$

```
[X Y]=meshgrid(-5:0.1:3,-4:0.1:5);
>> z=X.^2+Y.^2;
>> mesh(z)
```

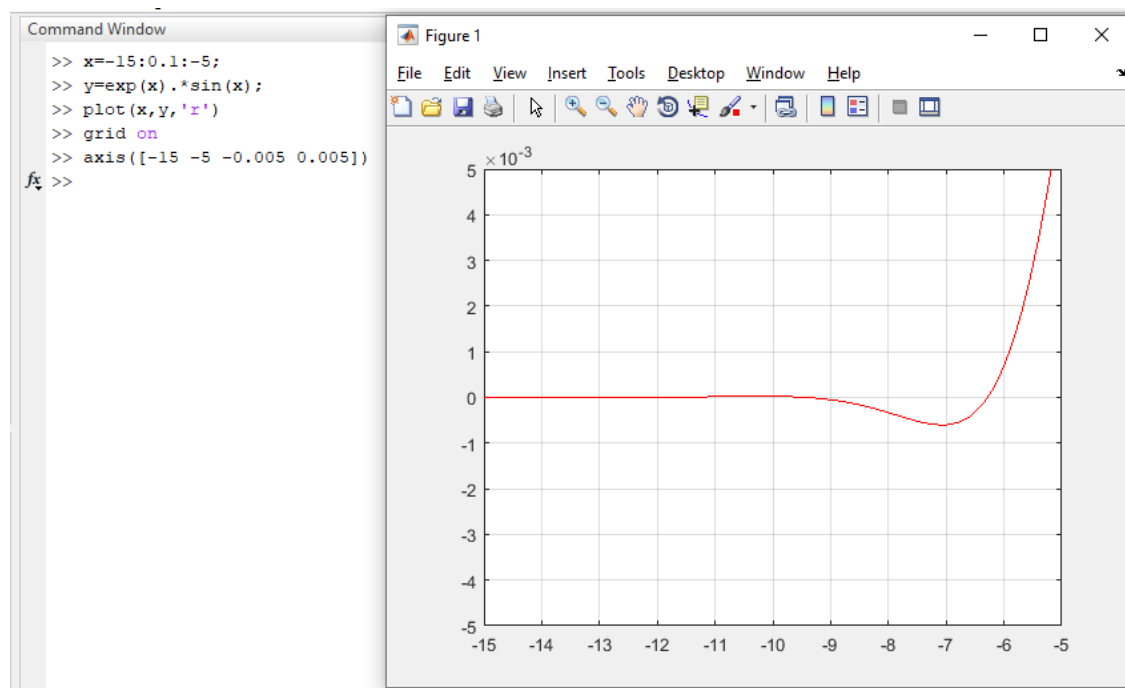
Ejercicio de clase

Construir la gráfica de la función $y = e^x \cdot \sin(x)$. Adicionalmente, intuir el valor del cero negativo de la función.

Una primera intención sería:



Una segunda intención sería:



NOTA.-

Se puede seguir aproximando cada vez más

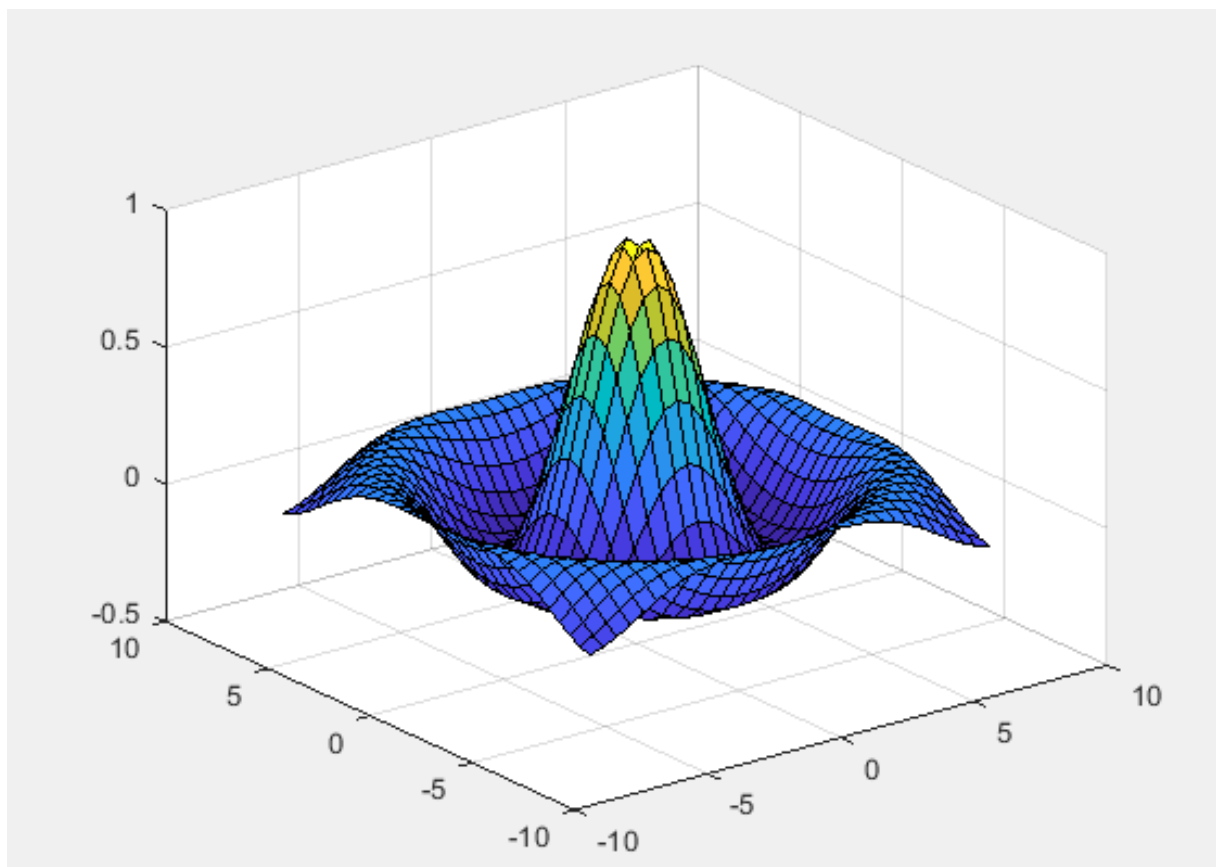
Ejercicio de clase

Realizar el gráfico de la función

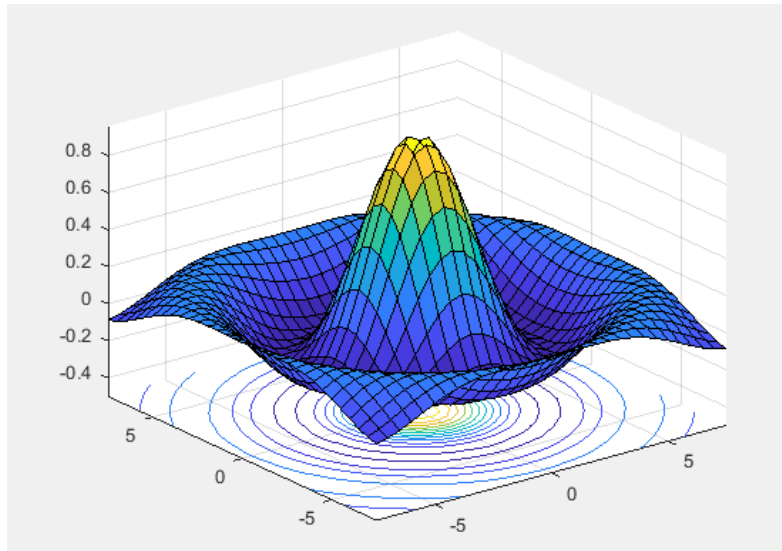
$$z = \frac{\text{Sen}(\sqrt{x^2 + y^2})}{\sqrt{x^2 + y^2}}$$

Command Window

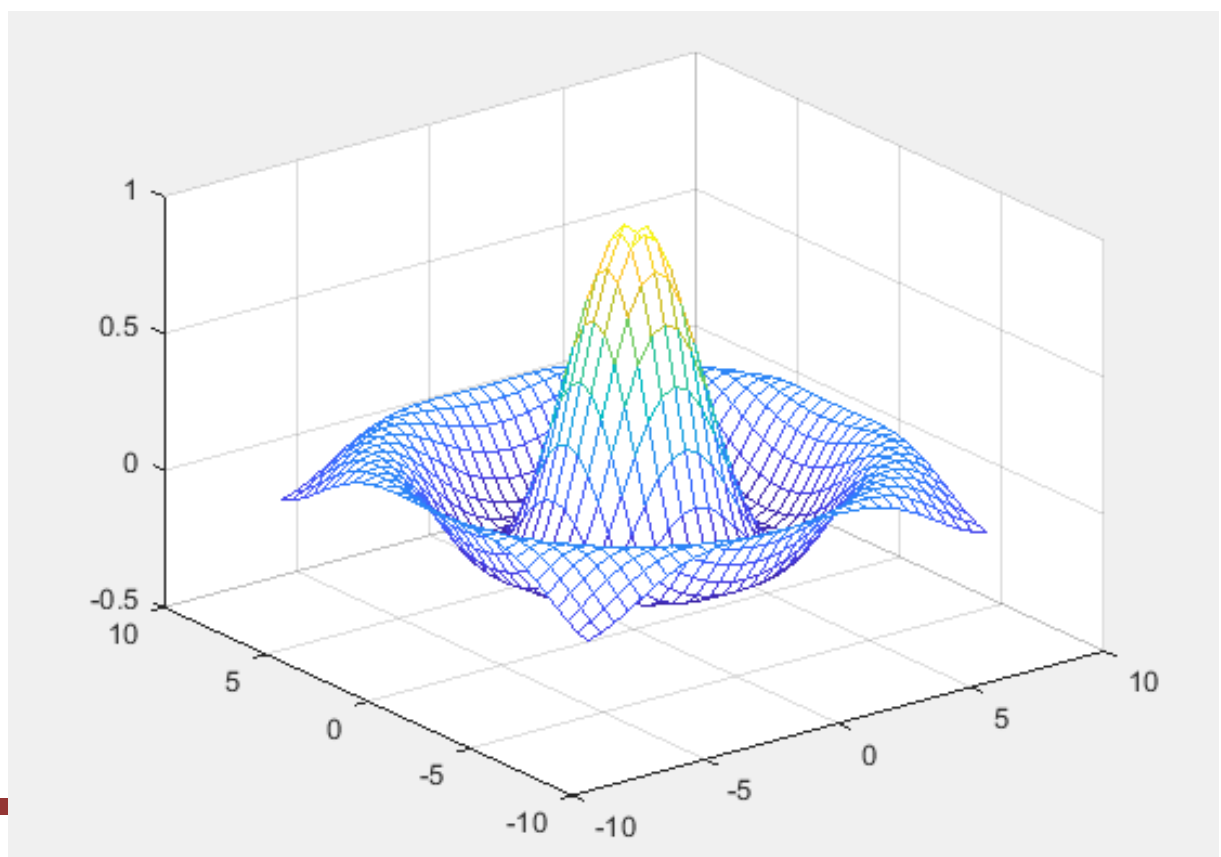
```
>> [x y]=meshgrid(-7.5:0.5:7.5);  
>> z=sin(sqrt(x.^2+y.^2))./sqrt(x.^2+y.^2);  
>> surf(x,y,z)
```



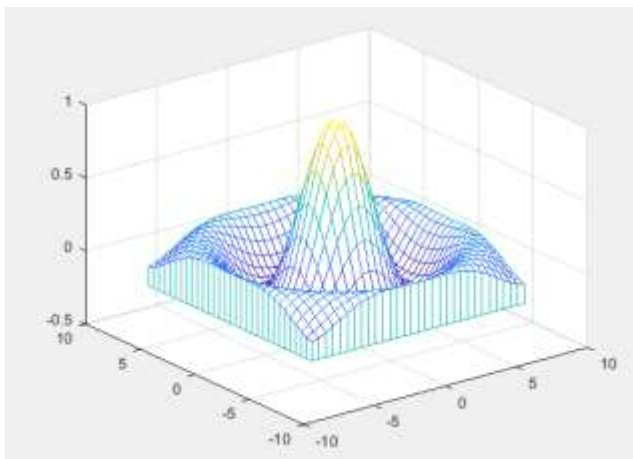
```
>> surfc(x,y,z)
fx >> |
```



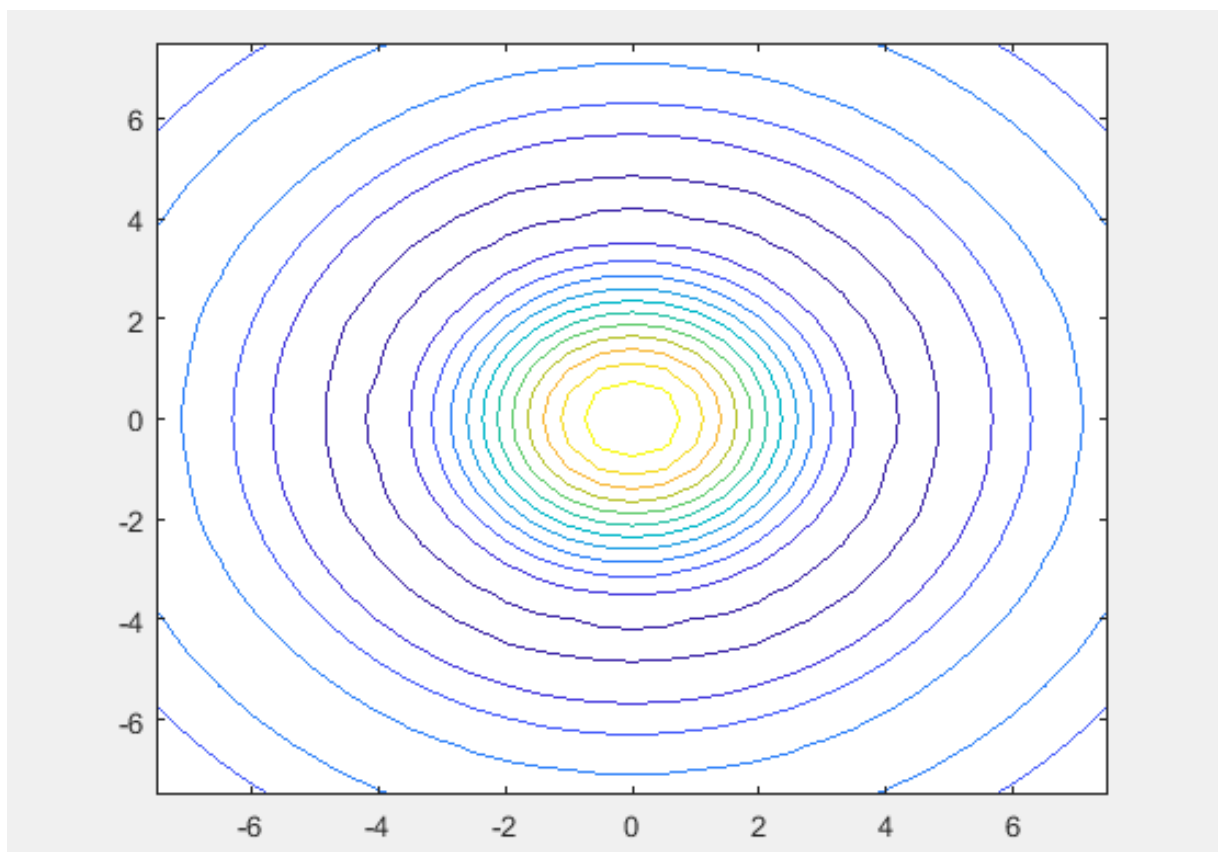
```
>> mesh(x,y,z) % gráfico de malla relativo a la superficie anterior
fx >> |
```



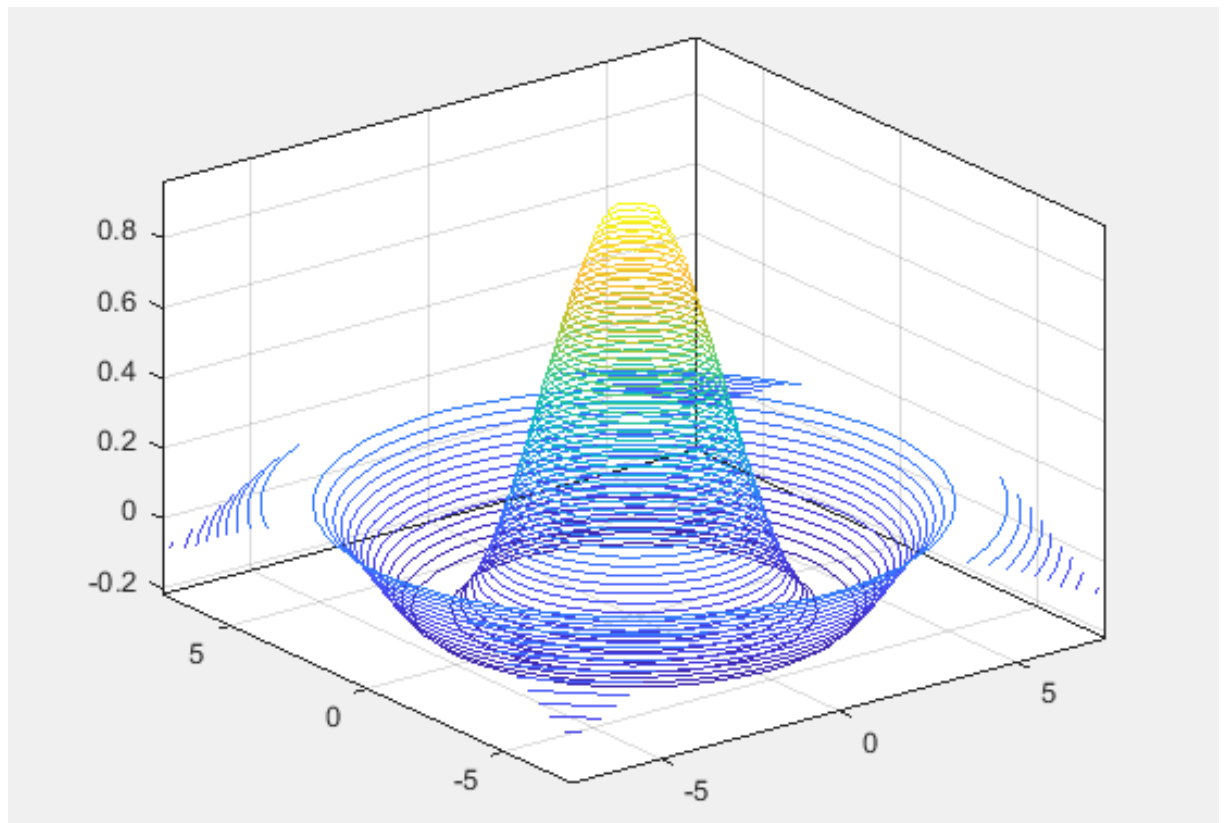
```
>> meshz(x,y,z) % gráfico de malla anterior con opción de cortina
fx >> |
```



```
>> contour(x,y,z) % curvas de nivel bidimensional para la superficie anterior
fx >> |
```



```
>> contour3(x,y,z,50) % curvas de nivel tridimensional para la superficie anterior
fx >> |
```

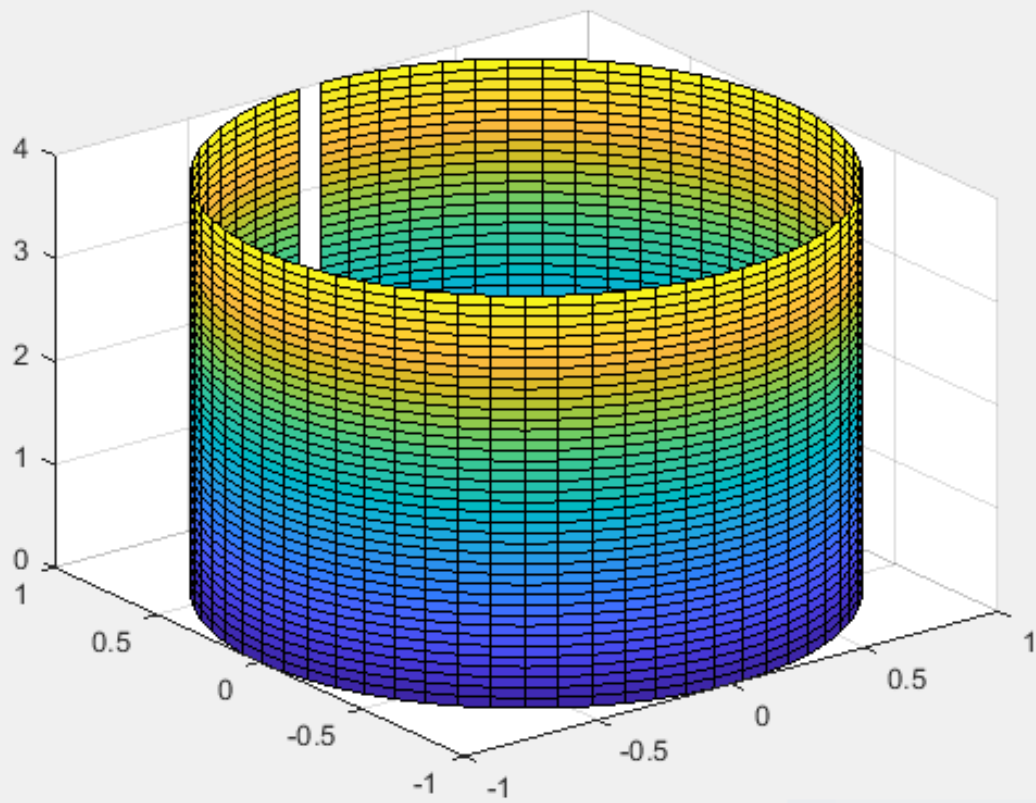


Graficar un cilindro dado en coordenadas paramétricas

$$\begin{cases} x(t) = t \\ y(t) = \text{Sen}(t) \\ z(t) = u \end{cases} \quad \begin{matrix} 0 \leq t \leq 2\pi \\ 0 \leq u \leq 4 \end{matrix}$$

x(t) = sen(t)
y(t) = cos(t)
z(t) =

```
>> t=(0:0.1:2*pi)';
>> r=(0:0.1:4);
>> x=sin(t)*ones(size(r));
>> y=cos(t)*ones(size(r));
>> z=ones(1,length(t))'*r;
>> surf(x,y,z)
fx >> |
```



PROGRAMACIÓN

TIPOS DE *m. files*

1. **ARCHIVOS DE INSTRUCCIONES (script).**- Secuencia de instrucciones dentro de *m. files* que persiguen una solución específica y única.

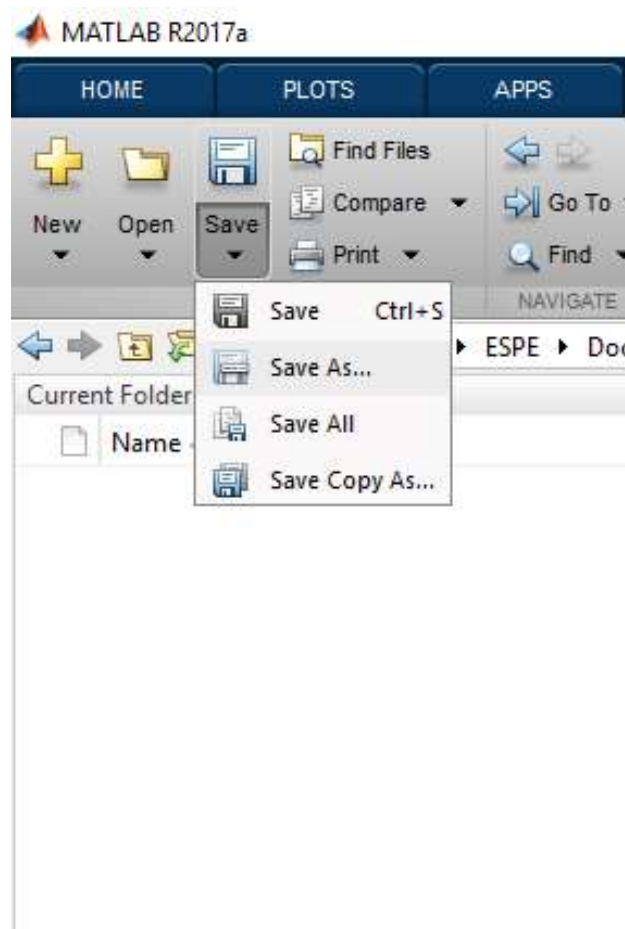
Se ejecuta con la flechita verde

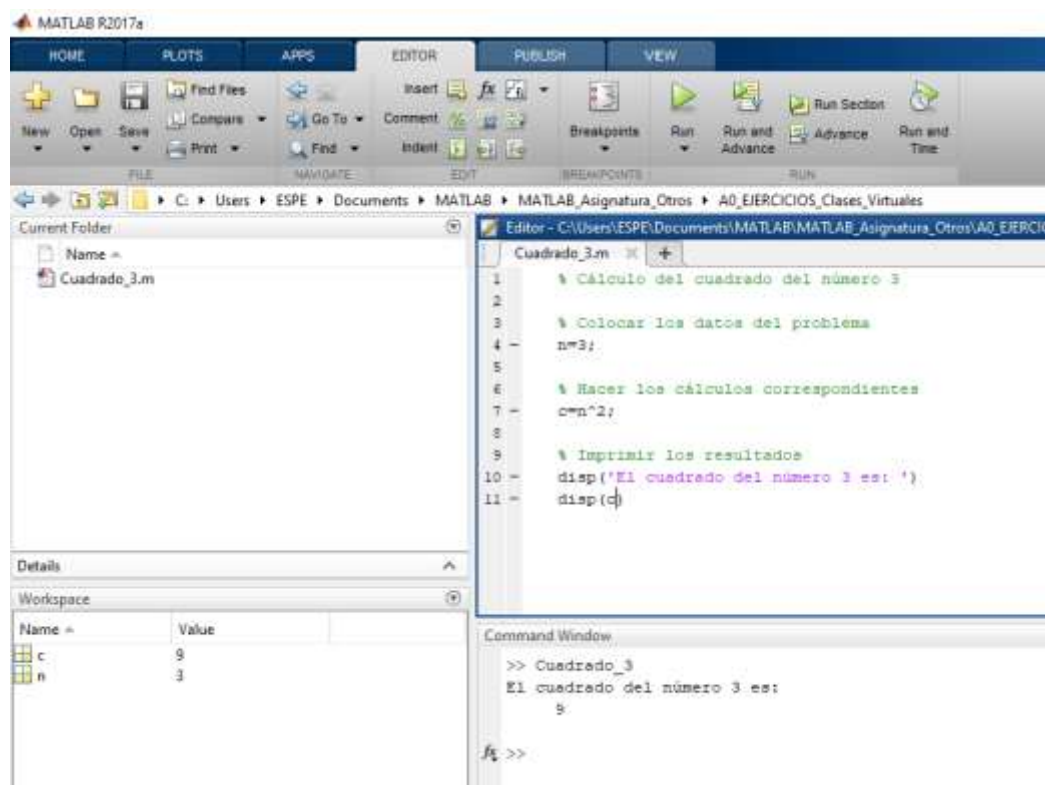
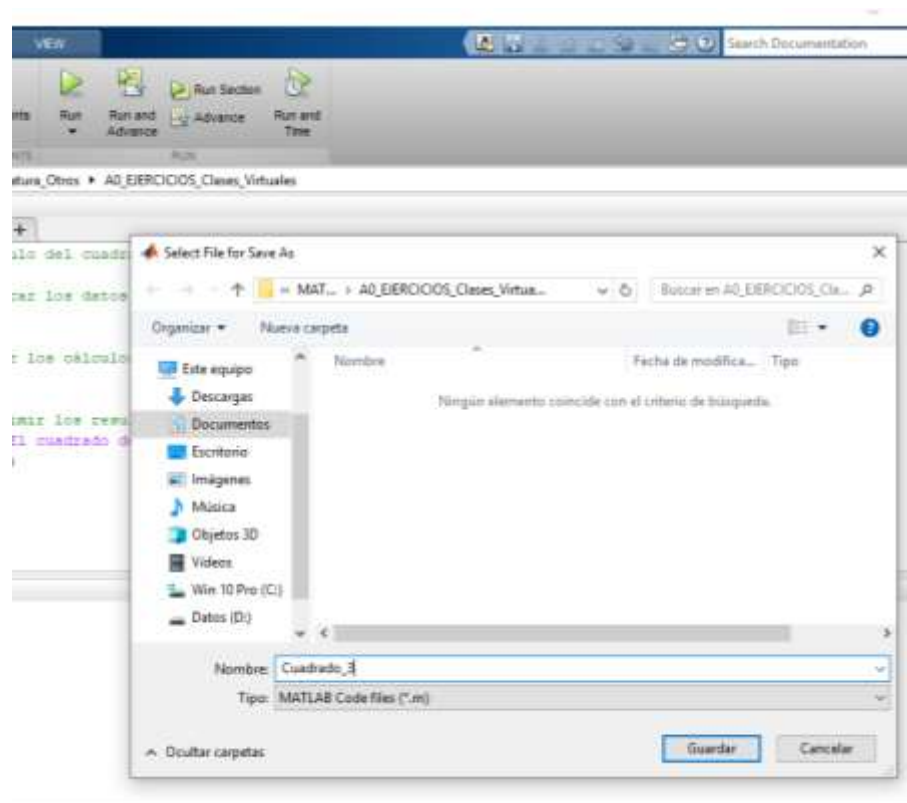
2. **ARCHIVO DE FUNCIONES (*function*).**- Conjunto de instrucciones que se pueden ejecutar como una función intrínseca de Matlab. Son nuevas funciones definidas por el usuario.

cuando es una función se ejecuta desde ventana de comandos con el nombre de esa función.

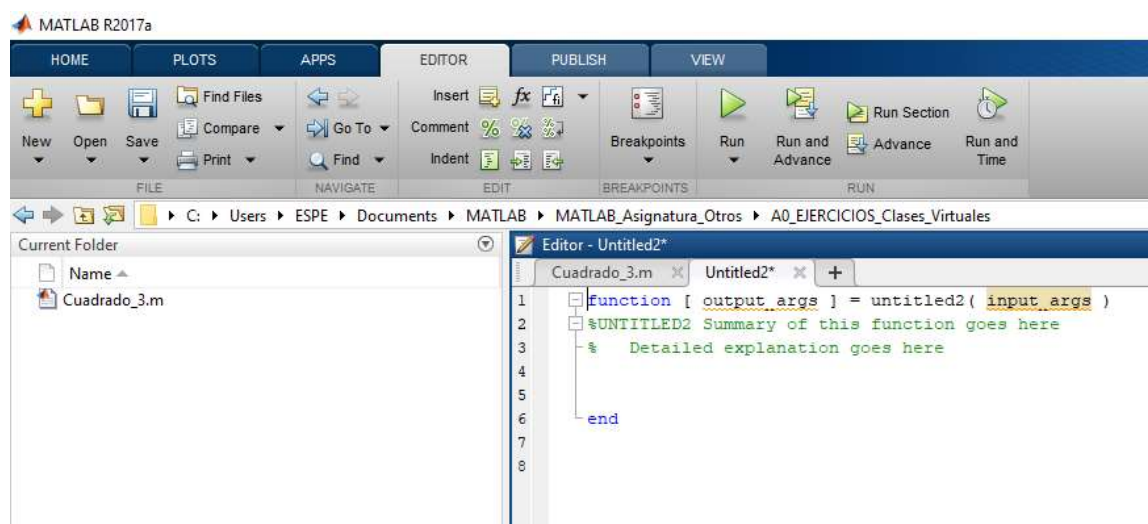
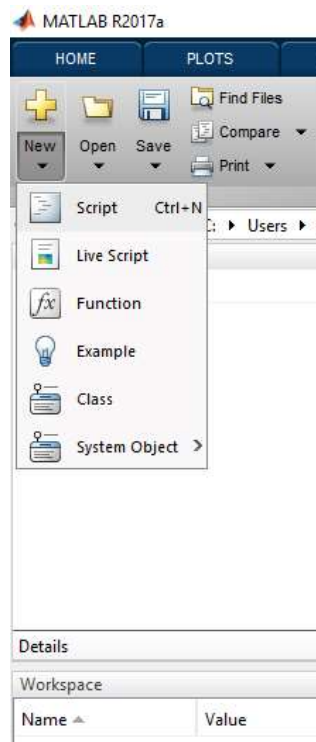
Ejercicio

Hacer un programa que calcule el cuadrado del número 3

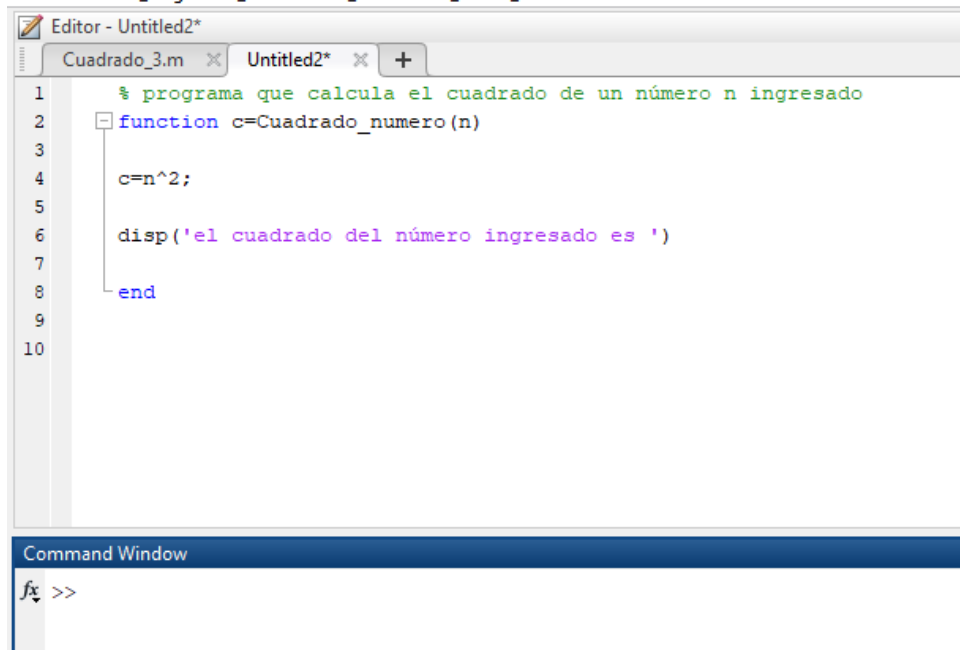




Archivo de función



AB ▸ MATLAB_Asignatura_Otros ▸ A0_EJERCICIOS_Clas_Virtuales




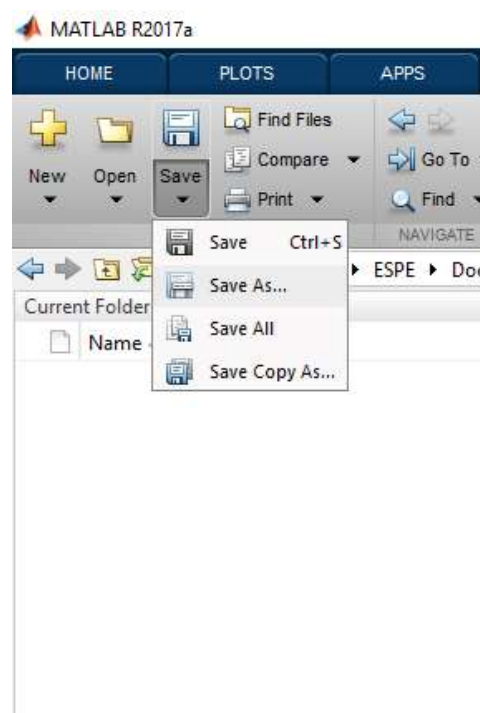
Editor - Untitled2*

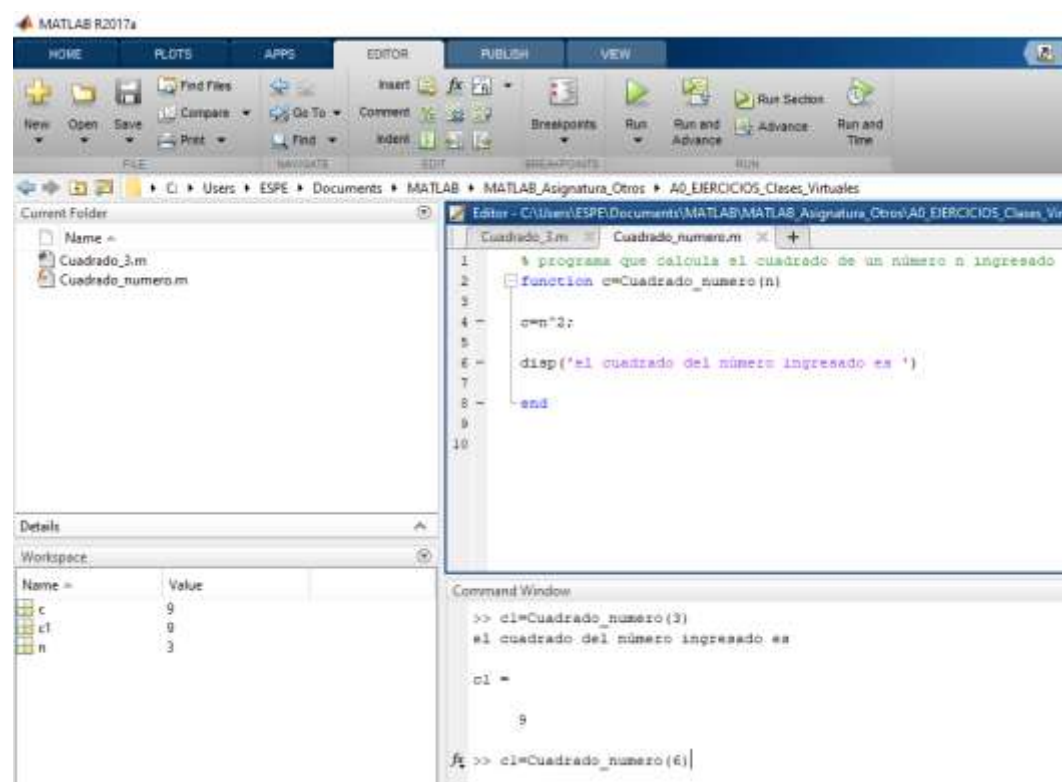
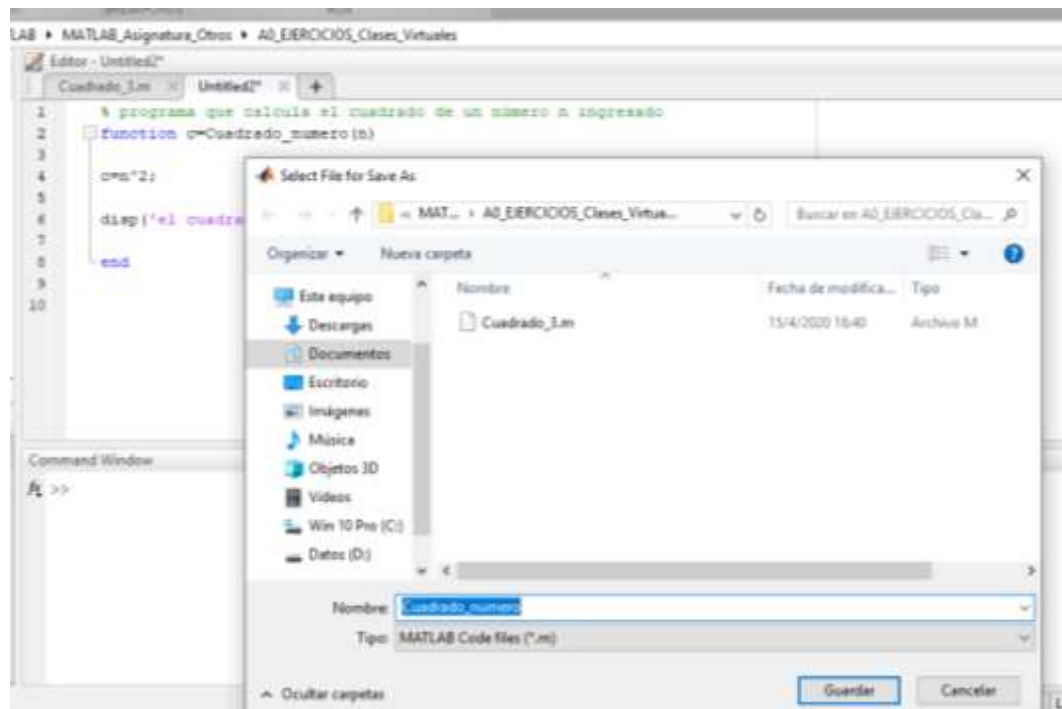
Cuadrado_3.m x Untitled2* x +

```
1 % programa que calcula el cuadrado de un número n ingresado
2 function c=Cuadrado_numero(n)
3
4     c=n^2;
5
6     disp('el cuadrado del número ingresado es ')
7
8 end
9
10
```

Command Window

 >>





ESTRUCTURAS DE CONTROL

Bucles *for*.- Repite un conjunto de instrucciones un determinado número de veces. Su estructura es

cuando SI sabemos
el número de iteraciones

↑
for ^{i,j,k,l}*index* = *n1:n2*
Lineas de ejecución
↓
end

↑
If condición
Lineas de ejecución
Else
Lineas de ejecución
↓
end

Bucle *while*.- Son similares a los bucles *for*; pero estos continúan hasta que se satisface algún criterio. Su estructura es:

Cuando NO conocemos el número de iteraciones

↑
while criterio
Lineas de ejecución
↓
end

Comando *break*.- Se usa para terminar un bloque prematuramente (sale del bucle). Su estructura es:

Sale totalmente

↑
if condición
Mensaje de salida
↓
break
end

Comando *continue*.- Similar al comando *break*. En lugar de salir del bucle, el programa salta al siguiente paso. Su estructura es: Otra oportunidad

```

↑ if condición
    Mensaje de Continuar
↓ continue
end

```

Comando *return*:
finaliza la secuencia actual de comandos y devuelve el control a la función que llamo a dicha función

```

if condición
    mensaje
return
end

```

Comando *switch case*.- Se utiliza cuando existe una serie de rutas de programación, para una variable dada, dependiendo de su valor. Permite elegir entre múltiples salidas. Su estructura es:

```

N=input(' opción A=1, opción B=2, opción C=3')
switch N
    case 1
        Lineas de ejecución
    case 2
        Lineas de ejecución
    case 3
        Lineas de ejecución
    otherwise
        disp('no existe este caso')
end

```

COMANDOS ADICIONALES IMPORTANTES

<i>fix(x)</i>	Elimina la parte decimal de x
<i>rem(x,y)</i>	Calcula el resto de dividir x para y
<i>round(x)</i>	Toma la parte entera de x (aproximando)
<i>floor(x)</i>	Toma el valor entero más próximo a la izquierda del número (no aproxima).
<i>ceil(x)</i>	Toma el valor entero más próximo a la derecha del número (no aproxima).
<i>sign(x)</i>	Determina el valor del signo de x. (-1 si es negativo, 1 si es positivo, 0 si x=0)

Ejercicios para la clase. -

1. Calcular la suma de los elementos menores a 90 e indicar el número de elementos mayores o iguales a 90, en el siguiente vector:

$$v = (60, 80, 95, 38, 90, 96, 48, 102)$$

2. Construir un archivo de funciones que proporcione una matriz $A = (a_{ij})$ de $(m \times n)$, m , n son parámetros de entrada; siendo

$$a_{ij} = \frac{1}{i + j - 1}.$$

Corresponde ésta a la matriz de Hilbert, mal condicionada.

3. Escribir la palabra 'HOLA', 10 veces, aplicando solo el bucle *while*.
4. Hacer un programa que calcule el logaritmo natural de un número positivo a , ingresado por pantalla; 10 ingresos en total (Usar comando *break*).
5. Hacer un programa que calcule el logaritmo natural de un número positivo a , ingresado por pantalla; 10 ingresos en total (Usar comando *continue*).

```
%WHILE
i=0 %inicializamos
while i<10
    disp('Hola')
    i=i+1
endwhile
```

codigos

codigos

REFERENCIAS BIBLIOGRAFICAS

1. Sánchez Juan Miguel, Problemas de Cálculo Numérico para ingenieros con aplicaciones Matlab, McGraw-Hill, Primera edición, 2005.
2. A. Quarteroni, F. Saleri, Cálculo Científico con Matlab y Octave. Springer-Verlag Italia, milano 2006.
3. César Pérez López, MATLAB a través de ejemplos. IBERGARCETA PUBLICACIONES, S.L., Madrid 2011.