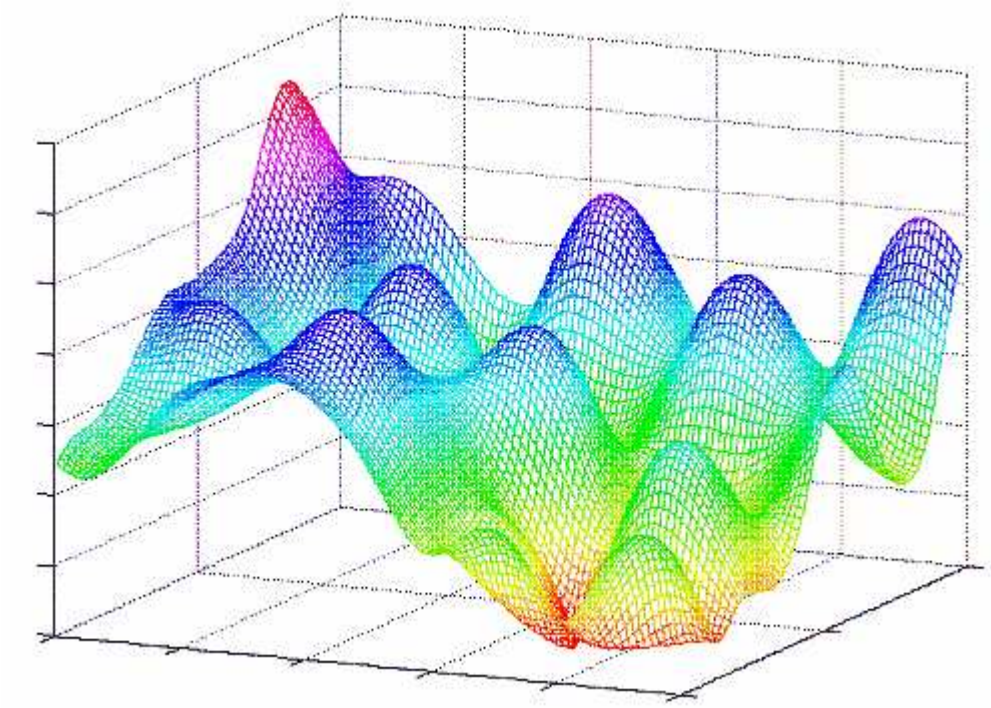
Representación de las diferentes formas de una superficie

Métodos Numéricos **Curso 05/06** **Grupo 13**



Magdalena Medina Rodríguez
Fernando García Torcelly
Miguel Hernández Jorge
Amanda Pérez Gil

INDICE

1. Enunciado

Realizar un programa en MATLAB que represente de diferentes formas una superficie. El trabajo debe considerar los siguientes aspectos y requisitos:

a. La superficie puede ser introducida mediante un fichero con una funcion explicitamente dada o con una cuadricula en cuyos vertices esta definida la funcion.

b. Representar la superficie con diferentes colores relativos a las diferentes cotas. Usar funciones de MatLab que permitan el movimiento tridimensional de las figuras. Ajustar los colores a los estandar de los plano topograficos. Señalar puntos, figuras geometricas, letreros, etc., dentro de la grafica.

c. Representar las lineas de nivel. Obtener un fichero con puntos que definan las lineas de nivel

d. Representar otro tipo de graficas que se considere oportuno.

2. Introducción

Para la realización de este trabajo hemos utilizado la versión 7 de Matlab. Antes de proceder a la realización del programa en terminos de pseudolenguaje, hemos de plantearnos cuales van a ser las pautas a seguir para un correcto desarrollo del mismo. Analizaremos los distintos puntos del enunciado con este fin.

En primer lugar debemos considerar las distintas opciones que puede tener el usuario a la hora de introducir los datos. Estas formas seran dos: Mediante un fichero con una función explícitamente dada, o mediante una cuadrícula en cuyos vértices estará definida la función.

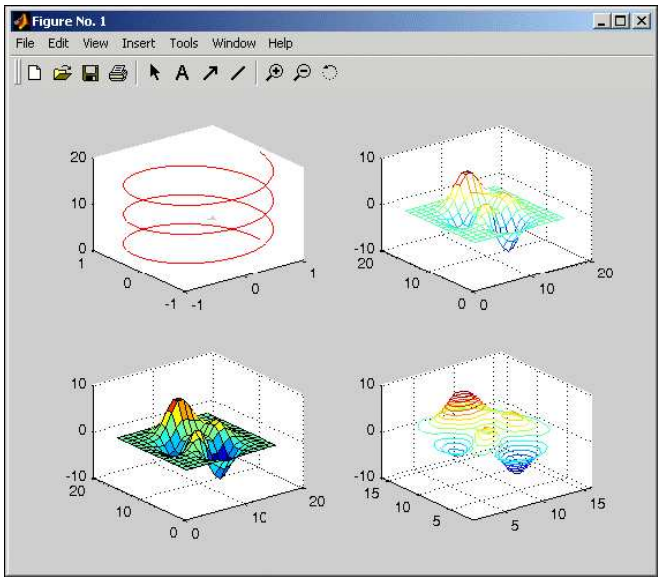
Para la representación de la superficie y todas sus posibilidades; ya sea la modificación de color relativos a las diferentes cotas, ajustando las mismas a los colores estandar de los planos topograficos, o bien para el movimiento tridimensional de la figura, señalización de puntos, implementación de figuras geometricas, letreros etc, hemos de realizar una búsqueda de los distintos comandos.

La búsqueda la realizaremos a traves de internet y a traves de la propia ayuda que nos facilita el programa, en concreto, los tutoriales de Matlab 7.0

3. Definicion de comandos

Tipos de funciones gráficas tridimensionales

MATLAB tiene posibilidades de realizar varios tipos de gráficos 3D. La primera forma de gráfico 3D es la función ***plot3***, que es el análogo tridimensional de la función ***plot***. Esta función dibuja puntos cuyas coordenadas están contenidas en 3 vectores, y a priori no creemos oportuna su utilizacion pues nuestro cometido se centra en la representacion grafica de superficies.



En la figura anterior aparece una buena muestra de algunas de las posibilidades gráficas tridimensionales de MATLAB.

DIBUJO DE MALLADOS: FUNCIONES *MESHGRID*, *MESH* Y *SURF*

Ahora se verá con detalle cómo se puede dibujar una función de dos variables (z=f(x,y)) sobre un dominio rectangular. Se verá que también se pueden dibujar los elementos de una matriz como función de los dos índices.
Sean **x** e **y** dos vectores que contienen las coordenadas en una y otra dirección de la retícula (*grid*) sobre la que se va a dibujar la función. Después hay que crear dos matrices **X** (cuyas filas son copias de **x**) e **Y** (cuyas columnas son copias de **y**).

Estas matrices se crean con la función ***meshgrid***. Estas matrices representan respectivamente las coordenadas *x* e *y* de todos los puntos de la retícula. La matriz de valores **Z** se calcula a partir de las matrices de coordenadas **X** e **Y**. Finalmente hay que dibujar esta matriz **Z** con la función ***mesh***, cuyos elementos son función elemento a elemento de los elementos de **X** e **Y**.

La función ***mesh*** dibuja *en perspectiva* una función en base a una retícula de líneas de colores, rodeando cuadriláteros del color de fondo, con eliminación de líneas ocultas. Más adelante se verá cómo controlar estos colores que aparecen. Por ahora el color dependerá del valor **z** de la función.

Ejecutando ahora el comando **surf(_)** tenemos que en vez de líneas aparece ahora una superficie *faceteada*, también con eliminación de líneas ocultas. El color de las facetas depende también del valor de la función.
En el caso de crear dos funciones paralelas utilizaremos la instrucción ***pause*** –que espera 5 segundos– para que se puedan ver las dos formas de representar la función.

DIBUJO DE LÍNEAS DE CONTORNO: FUNCIONES *CONTOUR* Y *CONTOUR3*

Una forma distinta de representar funciones tridimensionales es por medio de isolíneas o curvas de nivel.
A continuación se verá cómo se puede utilizar estas representaciones con matrices de datos , por ejemplo **Z** y **W**, calculadas previamente:

```
>> contour(Z,20)
>> contour3(Z,20)
>> contour(W,20)
>> contour3(W,20)
```

donde "20" representa el número de líneas de nivel. Si no se pone se utiliza un número por defecto. Otras posibles formas de estas funciones son las siguientes:

contour(Z, val) → siendo **val** un vector de valores para las isolíneas adibujar
contour(u,v,W,20) → se utilizan **u** y **v** para dar valores a los ejes de coordenadas
contour(Z,20,'r--') → se puede especificar el tipo de línea como en la función ***plot***
contourf(Z, val) → análoga a ***contour()***, pero rellenando el espacio entre líneas

Utilización del color en gráficos 3-D

En los dibujos realizados hasta ahora, se ha visto que el resultado adoptaba determinados colores, pero todavía no se ha explicado de dónde han salido. Ahora se verá qué sistema utiliza MATLAB para determinar los colores.

MAPAS DE COLORES

Un ***mapa de colores*** se define como una matriz de tres columnas, cada una de las cuales contiene un valor entre 0 y 1, que representa la intensidad de uno de los colores fundamentales: R (red o rojo), G (green o verde) y B (blue o azul).

La longitud por defecto de los mapas de colores de MATLAB es 64, es decir, cada mapa de color contiene 64 colores. Algunos mapas de colores están predefinidos en MATLAB. Buscando ***colormap*** en ***Help Desk*** se obtiene –entre otra información– la lista de los mapas de colores.

El colormap por defecto es ***jet***. Para visualizar estos mapas de colores se pueden utilizar los comandos:

```
>> colormap(hot)
>> pcolor([1:65;1:65]')
```

donde la función ***pcolor*** permite visualizar por medio de colores la magnitud de los elementos de una matriz (en realidad representa colores de "celdas", para lo que necesita que la matriz tenga una fila y columna más de las necesarias; ésa es la razón de que en el ejemplo anterior a la función ***pcolor*** se le pasen 65 filas y 2 columnas).
Si se desea imprimir una figura en una impresora láser en blanco y negro, puede utilizarse el mapa de color ***gray***.

El comando ***colormap*** actúa sobre la figura activa, cambiando sus colores. Si no hay ninguna figura activa, sustituye al mapa de color anterior para las siguientes figuras que se vayan a dibujar.

IMÁGENES Y GRÁFICOS EN *PSEUDOCOLOR*. FUNCIÓN *CAXIS*

Cuando se desea dibujar una figura con un determinado mapa de colores se establece una correspondencia (o un *mapping*) entre los valores de la función y los colores del mapa de colores. Esto hace que los valores pequeños se dibujen con los colores *bajos* del mapa, mientras que los valores grandes se dibujan con los colores *altos*.

La función ***pcolor*** es –en cierta forma– equivalente a la función ***surf*** con el punto de vista situado perpendicularmente al dibujo. Un ejemplo interesante de uso de la función ***pcolor*** es el siguiente:

Se genera una matriz **A** de tamaño 100x100 con valores aleatorios entre 0 y 1. La función ***pcolor(A)*** dibuja en color los elementos de la matriz **A**, mientras que la función ***pcolor(inv(A))*** dibuja los colores correspondientes a los elementos de la

matriz inversa. Se puede observar que los colores de la matriz inversa son mucho más uniformes que los de la matriz original. Los comandos son los siguientes:

```
>> A=rand(100,100); colormap(hot); pcolor(A); pause(5), pcolor(inv(A));
```

donde el comando ***pause(5)*** simplemente introduce un pausa de 5 seg en la ejecución. Al ejecutar todos los comandos en la misma línea es necesario poner ***pause*** pues si no dibuja directamente la inversa sin pasar por la matriz inicial.
Si todavía se conservan las matrices **Z** y **W** que se han definido previamente, se pueden hacer algunas pruebas cambiando el mapa de colores.
La función ***caxis*** permite ajustar manualmente la escala de colores. Su forma general es:

```
caxis([cmin, cmax])
```

donde ***cmin*** y ***cmax*** son los valores numéricos a los que se desea ajustar el mínimo y el máximo valor de la escala de colores.

DIBUJO DE SUPERFICIES FACETEADAS

La función ***surf*** tiene diversas posibilidades referentes a la forma en que son representadas las facetas o polígonos coloreados. Las tres posibilidades son las siguientes:

shading flat determina sombreado con color constante para cada polígono. Este sombreado se llama plano o *flat*.

shading interp establece que el sombreado se calculará por interpolación de colores
entre los vértices de cada faceta. Se llama también sombreado de Gouraud

shading faceted consiste en sombreado constante con líneas negras superpuestas. Esta es la opción por defecto.

OTRAS FORMAS DE LAS FUNCIONES *MESH* Y *SURF*

Por defecto, las funciones ***mesh*** y ***surf*** atribuyen color a los bordes y facetas en función de los valores de la función, es decir en función de los valores de la matriz **Z**. Ésta no es sin embargo la única posibilidad. En las siguientes funciones, las dos matrices argumento **Z** y **C** tienen el mismo tamaño:

```
mesh(Z,C)
surf(Z,C)
```

En las figuras resultantes, mientras se dibujan los valores de **Z**, los colores se obtienen de **C**. Un caso típico es aquél en el que se quiere que los colores dependan de la curvatura de la superficie (y no de su valor).
MATLAB dispone de la función ***del2***, que aproxima la curvatura por diferencias finitas con el promedio de los 4 elementos contiguos, resultando así una matriz proporcional a la

curvatura. Obsérvese el efecto de esta forma de la función ***surf*** en el siguiente ejemplo :
Mediante una matriz (Z) ;

```
>> C=del2(Z);
>> close, surf(Z,C)
```

FORMAS PARAMÉTRICAS DE LAS FUNCIONES *MESH*, *SURF* Y *PCOLOR*

Existen unas formas más generales de las funciones ***mesh***, ***surf*** y ***pcolor***. Son las siguientes (se presentan principalmente con la funciones ***mesh*** y ***surf***). La función:

```
mesh(x,y,Z,C)
```

dibuja una superficie cuyos puntos tienen como coordenadas (x(j), y(i), Z(i,j)) y como color C(i,j). Obsérvese que **x** varía con el índice de columnas e **y** con el de filas. Análogamente, la función:

```
mesh(X,Y,Z,C)
```

dibuja una superficie cuyos puntos tienen como coordenadas (X(i,j), Y(i,j), Z(i,j)) y como color C(i,j). Las cuatro matrices deben ser del mismo tamaño. Si todavía están disponibles las matrices calculadas con un fichero creado, ejecútese el siguiente comando y obsérvese que se obtiene el mismo resultado que anteriormente:

```
>> close, surf(X,Y,Z), pause(5), mesh(X,Y,Z)
```

¿Cuál es la ventaja de estas nuevas formas de las funciones ya conocidas? La principal es que admiten más variedad en la forma de representar la cuadrícula en el plano (x-y). La primera forma admite vectores **x** e **y** con puntos desigualmente espaciados, y la segunda admite conjuntos de puntos muy generales, incluso los provenientes de coordenadas *cilíndricas* y *esféricas*.

OTRAS FUNCIONES GRÁFICAS 3D

Las siguientes funciones se derivan directamente de las anteriores, pero añaden algún pequeño detalle y/o funcionalidad:

surfc combinación de ***surf***, y ***contour*** en z=0

meshz ***mesh*** con plano de referencia en el valor mínimo y una especie de "cortina" en los bordes del dominio de la función

surfl para controlar la iluminación determinando la posición e intensidad de un foco
de luz.

light crea un foco de luz en los ejes actuales capaz de actuar sobre superficies 3-D. Se le deben pasar como argumentos el color, el estilo (luz local o en el infinito) y la posición. Son muy importantes las propiedades de los objetos iluminados ***patch*** y ***surface.***

Elementos generales: ejes, puntos de vista, lineas ocultas

Las funciones ***surf*** y ***mesh*** dibujan funciones tridimensionales en perspectiva. La localización del punto de vista o dirección de observación se puede hacer mediante la función ***view***, que tiene la siguiente forma:

```
view(azimut, elev)
```

donde ***azimut*** es el ángulo de rotación de un plano horizontal, medido sobre el eje **z** a partir del eje **x** en sentido antihorario, y ***elev*** es el ángulo de elevación respecto al plano (x-y). Ambos ángulos se miden ***en grados***, y pueden tomar valores positivos y negativos (sus valores por defecto son -37.5 y 30). También se puede definir la dirección del punto de vista mediante las tres coordenadas cartesianas de un vector (sólo se tiene en cuenta la dirección):

```
view([xd,yd,zd])
```

En los gráficos tridimensionales existen funciones para controlar los ejes, por ejemplo:

```
axis([xmin,xmax,ymin,ymax,zmin,zmax])
```

También se pueden utilizar las funciones siguientes: ***xlabel***, ***ylabel***, ***zlabel***, ***axis('auto')***, ***axis(axis)***, etc.

Las funciones ***mesh*** y ***surf*** disponen de un algoritmo de *eliminación de líneas ocultas* (los polígonos o facetas, no

dejan ver las líneas que están detrás). El comando ***hidden*** activa y desactiva la eliminación de líneas ocultas. En el dibujo de funciones tridimensionales, a veces también son útiles los *NaNs*. Cuando una
parte de los elementos de la matriz de valores **Z** son *NaNs*, esa parte de la superficie no se dibuja, permitiendo ver el resto de la superficie.

4. Planteamiento del algoritmo

El primer paso será definir los dos tipos de entrada de datos solicitados en el enunciado. La superficie deberá ser introducida mediante un fichero con una funcion explicitamente dada o con una cuadricula en cuyos vertices estará definida la funcion. Y antes de empezar con la inicialización de variables, es conveniente crear la traza que nos facilite el planteamiento, ademas de servirnos de guia para la elavoracion del programa.

Queremos que la entrada de datos, por tanto, no sea unica. Para ello definimos dos vias de acceso para su representación. Tendremos en cuenta que por matriz necesitaremos de un fichero .m donde escribiremos los datos de nuestra cuadricula. Mientras que con la función tendremos la posibilidad de introducirla mediante teclado y fichero, donde en este ultimo escribiremos la ecuacion en funcion de las variables X e Y.

Si nuestra entrada es a traves de una funcion, durante el desarrollo del programa hemos de definir los puntos inciales de la cuadricula donde posteriormente se vera representada nuestra superficie. En coordenadas de X e Y representaremos tambien los puntos finales, y por ultimo el programa nos pedirá la distancia entre puntos con la utilidad intrinseca de aumentar la posibilidad de definicion de nuestra superficie.

Si nuestra entrada es a traves de una matriz ubicada en un fichero .m tendremos que darle la ubicación de dicho fichero ademas del numero de ecuaciones antes de disponer de la representacion grafica de nuestra superficie. Es importante especificar que la matriz de puntos debe escribirse en el fichero a traves de puntos con coordenadas x crecientes y, aquellos que tengan misma coordenada x, se escriben con orden de coordenada y creciente.

Una vez procesados todos estos datos de entrada, el programa ya estara en disposicion de representar nuestra figura, utilizando las funciones mesh y surf anteriormente vistas; las posibilidades para la representación seran:

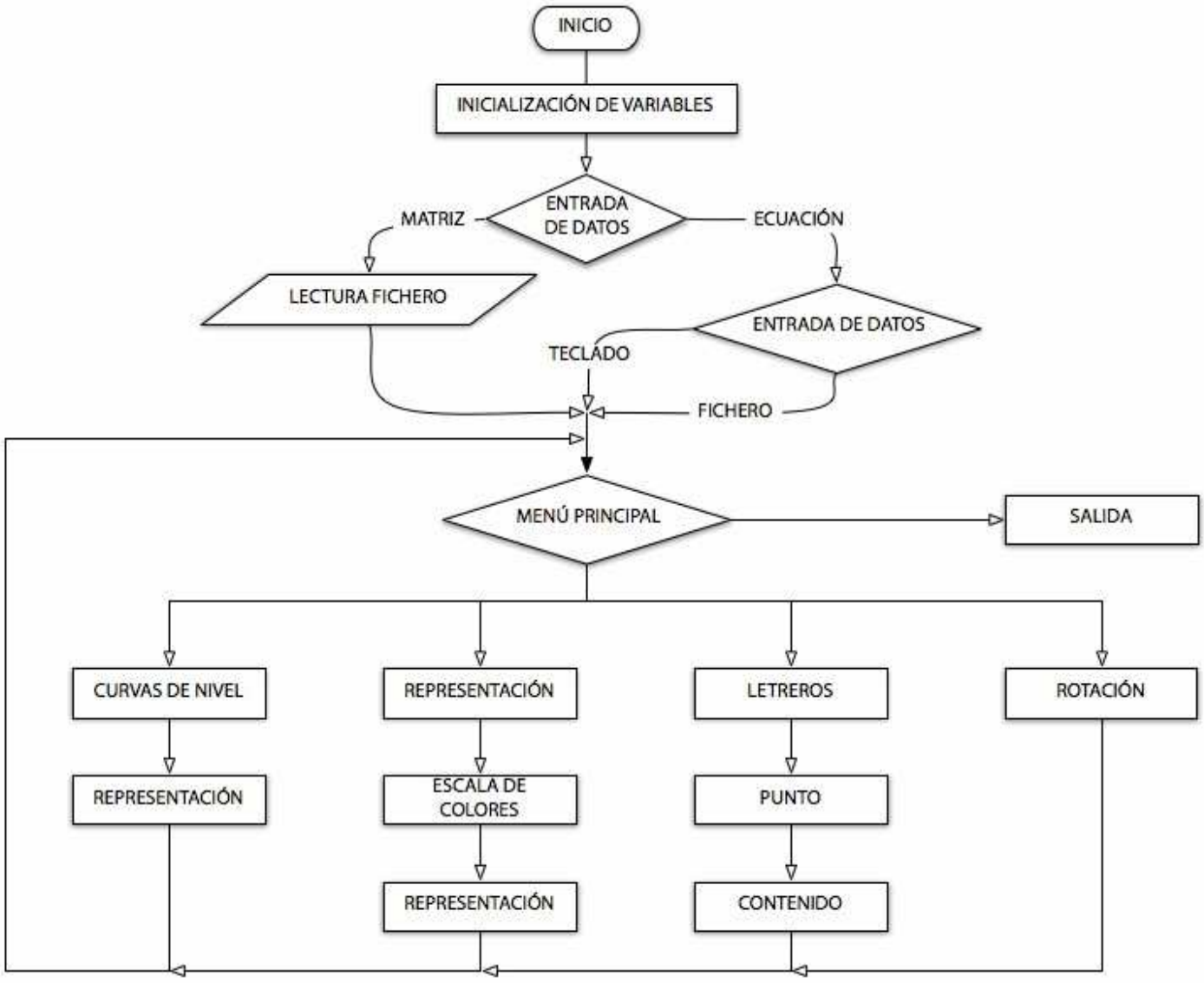
1. Clasica
2. Clasica solida
3. Sombreada

Una vez elegida la opcion deseada, automaticamente aparecera una ventana donde podremos visualizar graficamente la superficie deseada. A continuación se ejecutará un menu, donde podremos elegir las distintas opciones pedidas en el enunciado. Estas son:

• Movimiento de rotacion graficamente sobre la superficie. (rotate3d)
• Representacion de las curvas de nivel. (contour(Z))

• Posicionamiento y ubicación de letreros.

Teniendo la posibilidad final de poder representar de nuevo la grafica, ademas de un quinto punto de salida automatica del programa.
A continuacion, se describe a modo de esquema, la traza de nuestro programa, señalando los distintos pasos, junto al recorrido y sus posibilidades :



5. Desarrollo

El codigo fuente del programa creado para la representacion grafica de superficies es el siguiente:

```
                    REPRESENTACION GRAFICA DE SUPERFICIES

 Representar una superficie dada.

ENTRADA: Fichero con una funcion explicitamente dado o con una cuadricula en cuyos
          vertices esta definida la funcion.
SALIDA:  Grafica con la representacion de los datos introducidos.

DECLARACION DE VARIABLES

syms('INP','NAME','OK','OPC',  'XI','YI','XF','YF','XD','YD','N',  'I',  'J',  'P',
'TOL', 'NN', 'X','ZZ', 'g');
syms('FLAG','FLAG2','FLAG3',   'FUNC',   'OUP',   'K',   'A',   'R',   's',   'ss',
'fs','F','Mx','Mx1','Mn','Mn1','Dif');
syms('K1',    'I1',    'Z1',    'IR1',    'IA1',    'J1',    'C1',    'L1',    'JA1',
'EP','XT','YT','ZT','T');

INICIALIZACION DE VARIABLES

      TRUE = 1;
      FALSE = 0;
      X=0;
      y=0;
      z=0;

TITULO

 fprintf(1,'Representacion grafica de superficies.\n\n');
 OK = FALSE;

TIPO DE INFORMACION DEL FICHERO

   fprintf(1,'Indique como encuentra estructurada la informacion de la superficie a
   representar.\n');
   fprintf(1,'\t1. Ecuacion en (X,Y).\n');
   fprintf(1,'\t2. Matriz con coordenadas X,Y,Z.\n');
   FLAG = input(' ');

ECUACION XY
        if FLAG == 1
            fprintf(1,'El fichero debe contener la funcion en terminos de x,y (en
             minuscula).\n');
            fprintf(1,'La funcion puede ser leidas desde:\n');
            fprintf(1,'\t1. Teclado.\n');
            fprintf(1,'\t2. Fichero.\n');
           FLAG2 = input(' ');
                if FLAG2 == 1
                    fprintf(1,'Introduzca la ecuacion en (x,y):\n');
                    s = input(' ','s');
                    F = inline(s,'x','y');
                else
                    fprintf(1,'Entrada de la funcion en terminos de X,Y.\n');
                    fprintf(1,'Escriba el nombre del fichero en la forma -
                   disco:\\nombre.ext\n');
                    fprintf(1,'Por ejemplo:   A:\\DATA.DTA\n');
```

```
                NAME = input(' ','s');
                INP = fopen(NAME,'rt');
                s = fscanf(INP,'%s',1);
                F = inline(s,'x','y');
            end;

          OK = FALSE;


            while OK == FALSE
                fprintf(1,'Indique el punto inicial de la cuadricula
                escribiendo las coordenadas X,Y en lineas separadas.\n');
                XI = input(' ');
                YI = input(' ');
                fprintf(1,'Indique el punto final de la cuadricula escribiendo
                las coordenadas X,Y en lineas separadas.\n');
                XF = input(' ');
                YF = input(' ');
                fprintf(1,'Indique la distancia entre puntos en el eje X.\n');
                XD = input(' ');
                fprintf(1,'Indique la distancia entre puntos en el eje Y.\n');
                YD = input(' ');
                [x,y] = meshgrid(XI:XD:XF,YI:YD:YF);
                z=F(x,y);
                OK = TRUE;
            end;
        end;
```

MATRIZ DE PUNTOS

```
    if FLAG == 2
        fprintf(1,'Entrada de la funcion en terminos de X,Y.\n');
        fprintf(1,'Escriba el nombre del fichero en la forma -
        disco:\\nombre.ext\n');
        fprintf(1,'Por ejemplo:   A:\\DATA.DTA\n');
        NAME = input(' ','s');
        INP = fopen(NAME,'rt');
        fprintf(1,'Escriba el numero de ecuaciones.\n');
        N = input(' ');
        IND = N/3;
        I = 1;
        J = 1;
            while J <= IND
                while I <= 3
                    x(I,J) = fscanf(INP,'%f',1);
                    y(I,J) = fscanf(INP,'%f',1);
                    z(I,J) = fscanf(INP,'%f',1);
                    I=I+1;
                end;
            J=J+1;
            I=1;
            end;
    end;
```

TIPO DE REPRESENTACION

```
OKP = FALSE;
    while OKP == FALSE;
        fprintf(1,'Indique el tipo de representacion.\n');
        fprintf(1,'\t1. Clasica.\n');
        fprintf(1,'\t2. Clasica Solida.\n');
```

```
        fprintf(1,'\t3. Sombreada.\n');
        FLAG2 = input(' ');
        OK = FALSE;
        while OK == FALSE


            if (FLAG2 == 1) || (FLAG2 == 2) || (FLAG2 == 3)
                OK = TRUE;
            else
                fprintf(1,'Introduzca 1, 2 o 3.\n');
            end;
        end;
```

EJECUCION REPRESENTACION

```
        A=[79/255   168/255   206/255;102/255   187/255   215/255;130/255   196/255
        222/255;154/255  217/255  234/255;192/255  230/255  241/255;203/255  235/255
        230/255;230/255  246/255  236/255;205/255  233/255  148/255;178/255  212/255
        125/255;154/255  201/255  123/255;241/255  222/255  163/255;241/255  206/255
        112/255;240/255  175/255  109/255;193/255  156/255  103/255;199/255  130/255
        61/255;199/255 90/255 52/255];
         colormap(A);
         V=[-6000,6000];
         caxis(V);

          CLASICA

          if (FLAG2 == 1)
              mesh(z);
          end;

          RELLENO

          if (FLAG2 == 2)
              surf(z);
          end;

          SOMBREADO

          if (FLAG2 == 3)
              surf(z);
              shading interp;
          end;
          xlabel = ('Eje X');
          ylabel = ('Eje Y');
          zlabel = ('Eje Z');
          title = (s);
```

MENU PRINCIPAL

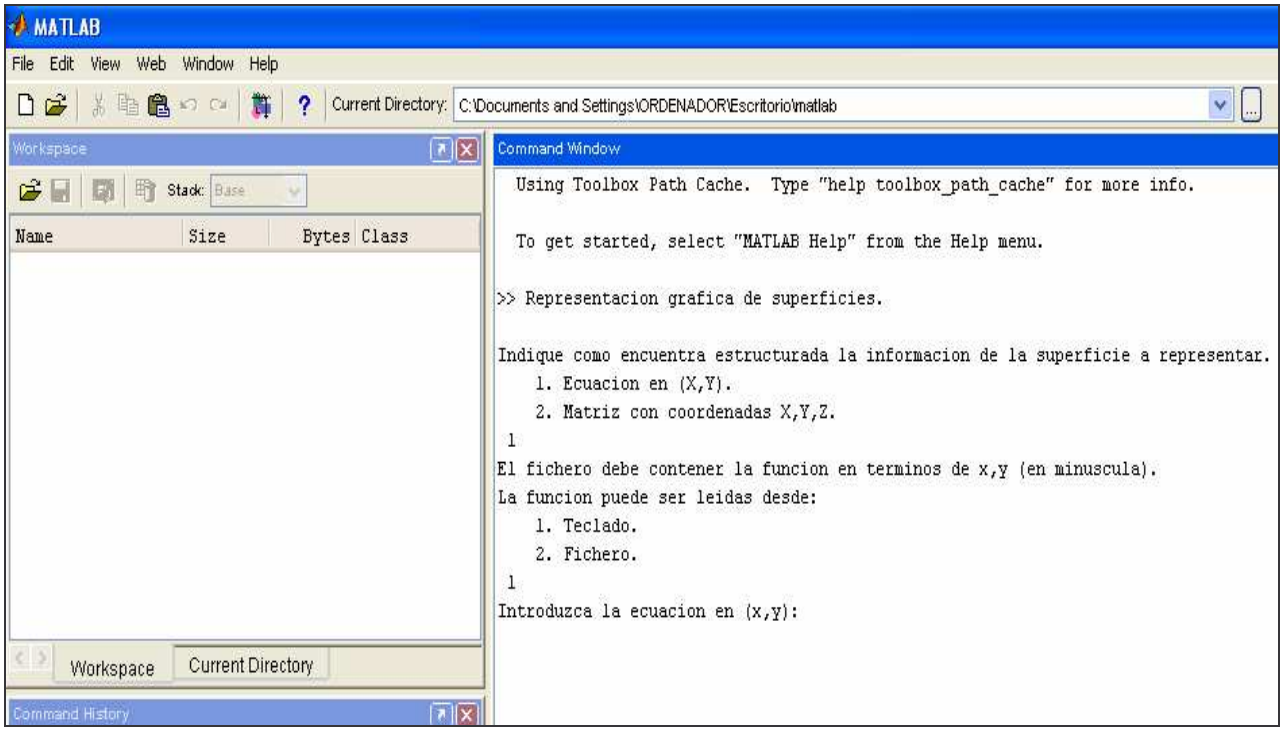
```
        OPC = TRUE;
        while (OPC == TRUE)
            fprintf(1,'Seleccione la opcion que prefiera:.\n');
            fprintf(1,'\t1.Representar de nuevo la grafica.\n');
            fprintf(1,'\t2.Mover grafica en 3D.\n');
            fprintf(1,'\t3.Representar curvas de nivel.\n');
            fprintf(1,'\t4.Colocar letrero.\n');
            fprintf(1,'\t5.Salir.\n');
            FLAG3 = input(' ');
```

```
        if (FLAG3 == 1)
            OKP = FALSE;
            OPC = FALSE;
        end;


        MOVIMIENTO DE CURVA

        if (FLAG3 == 2)
            rotate3d;
            fprintf(1,'Deje presionado el boton izquierdo del raton en la
            grafica y muevalo.\n\n\n');
        end;

        CURVAS DE NIVEL

        if (FLAG3 == 3)
            contour(Z)
        end;

        LETREROS

        if (FLAG3 == 4)
            fprintf(1,'Indique el punto x,y en el que quiera introducir el
            letrero.\n');
            XT = input(' ');
            YT = input(' ');
            Mx1=max(Z);
            Mx=max(Mx1);
            Mn1=min(Z);
            Mn=min(Mn1);
            Dif=(Mx-Mn)/10;
            H=F(XT,YT);
            ZT=H+Dif;
            fprintf(1,'Indique el texto que debe contener el letrero.\n');
            T = input(' ','s');

         Text(XT,YT,ZT,T,'FontSize',11,'VerticalAlignment','bottom','Horizontal
         Alignment', 'right','BackgroundColor',[236/256 241/256
         248/256],'EdgeColor',[60/256 60/256 60/256],'LineWidth',1);

        end;

        if (FLAG3 == 5)
            OPC = FALSE;
            OKP = TRUE;
        end;
    end;
end;
```

7. Ejemplos y Conclusiones

EJEMPLO 1 : Representación de una superficie a traves de una función introducida por teclado.

Ejecutamos el programa, y la primera eleccion que debemos realizar es la de introducir la superficie. Nos aparecera de la siguiente manera:



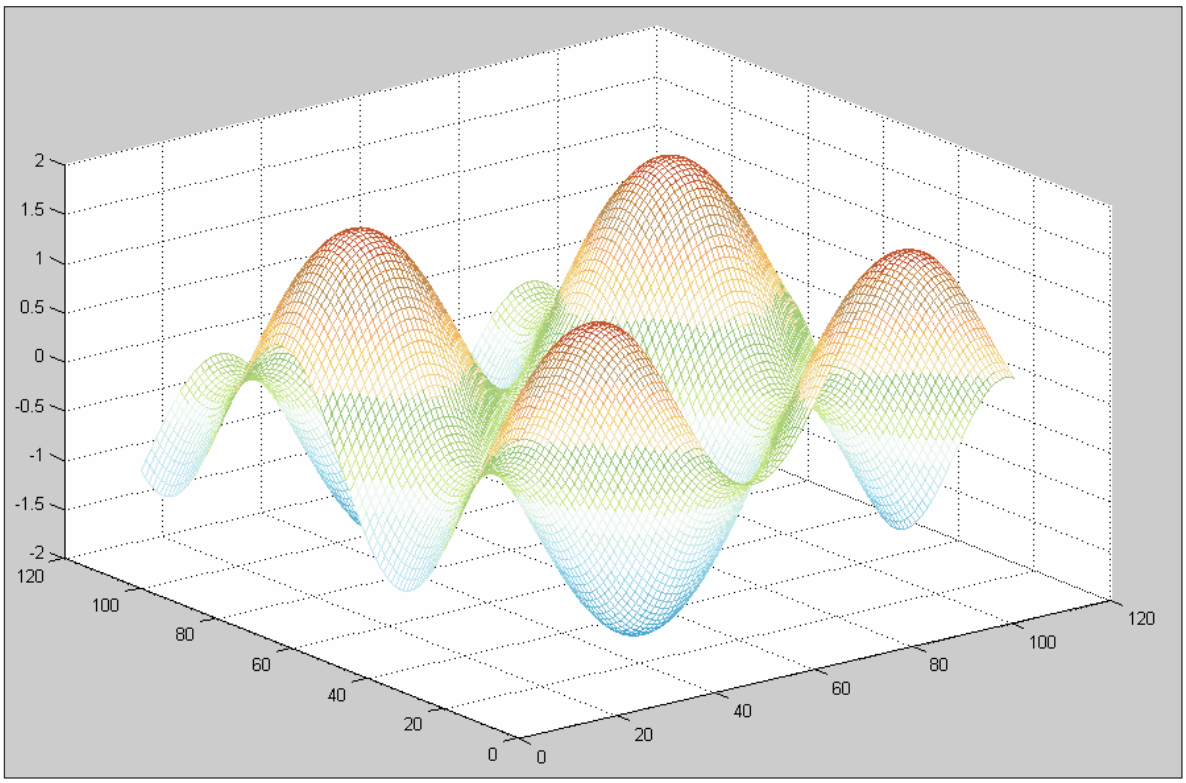
Introducimos por ejemplo la siguiente ecuacion :

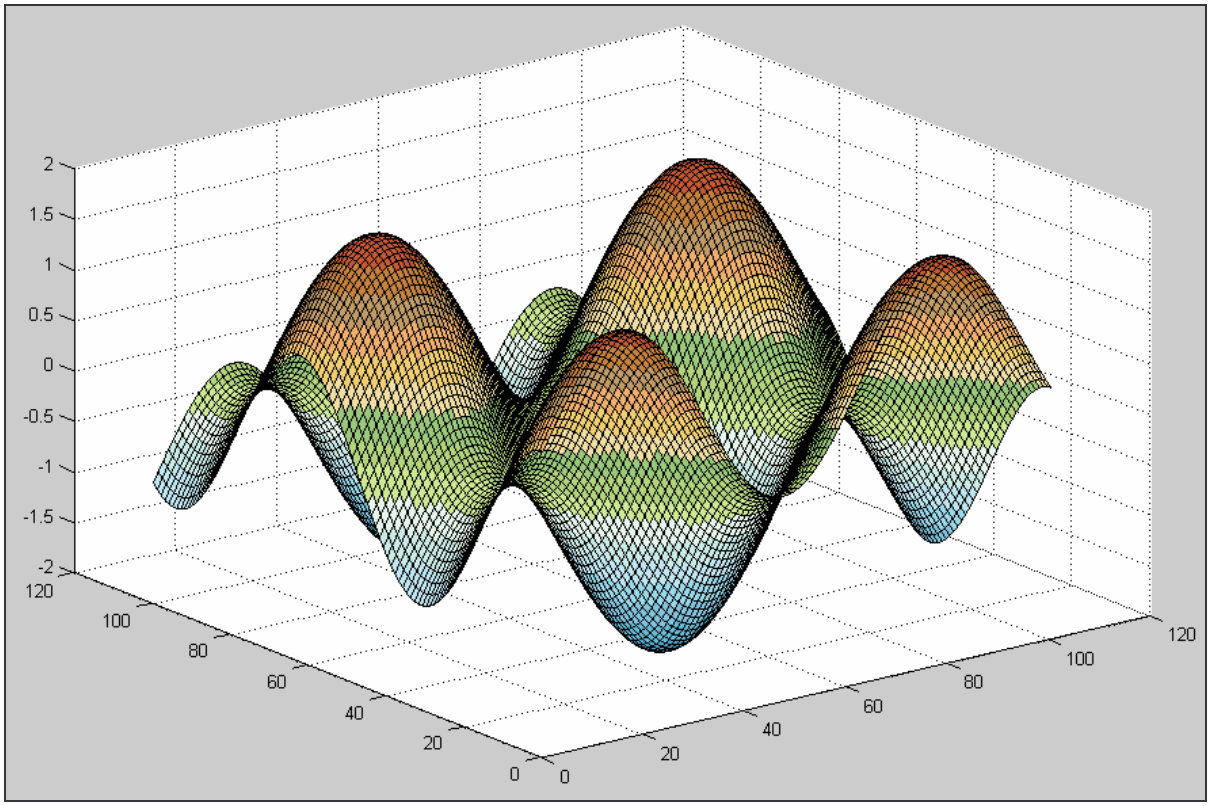
sen(x)+cos(y)

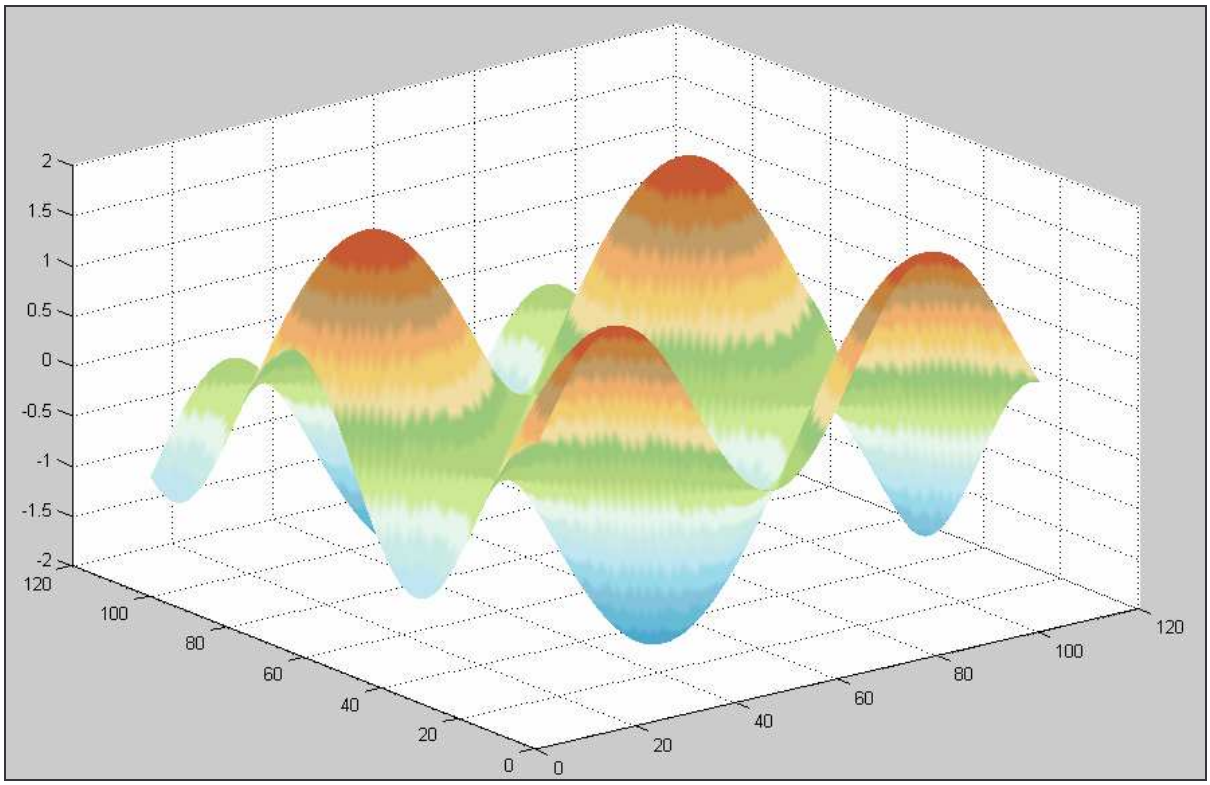
y los puntos limite de la cuadricula, siendo el minimo el 0 y el maximo el 10, con una distancia entre puntos de 0,1.-

```
>> Representacion grafica de superficies.

Indique como encuentra estructurada la informacion de la superficie a representar.
    1. Ecuacion en (X,Y).
    2. Matriz con coordenadas X,Y,Z.
 1
El fichero debe contener la funcion en terminos de x,y (en minuscula).
La funcion puede ser leidas desde:
    1. Teclado.
    2. Fichero.
 1
Introduzca la ecuacion en (x,y):
 sin(x)+cos(y)
Indique el punto inicial de la cuadricula escribiendo las coordenadas X,Y en lineas separadas.
 0
 0
Indique el punto final de la cuadricula escribiendo las coordenadas X,Y en lineas separadas.
 10
 10
Indique la distancia entre puntos en el eje X.
 0.1
Indique la distancia entre puntos en el eje Y.
 0.1
Indique el tipo de representacion.
    1. Clasica.
    2. Clasica Solida.
    3. Sombreada.
```

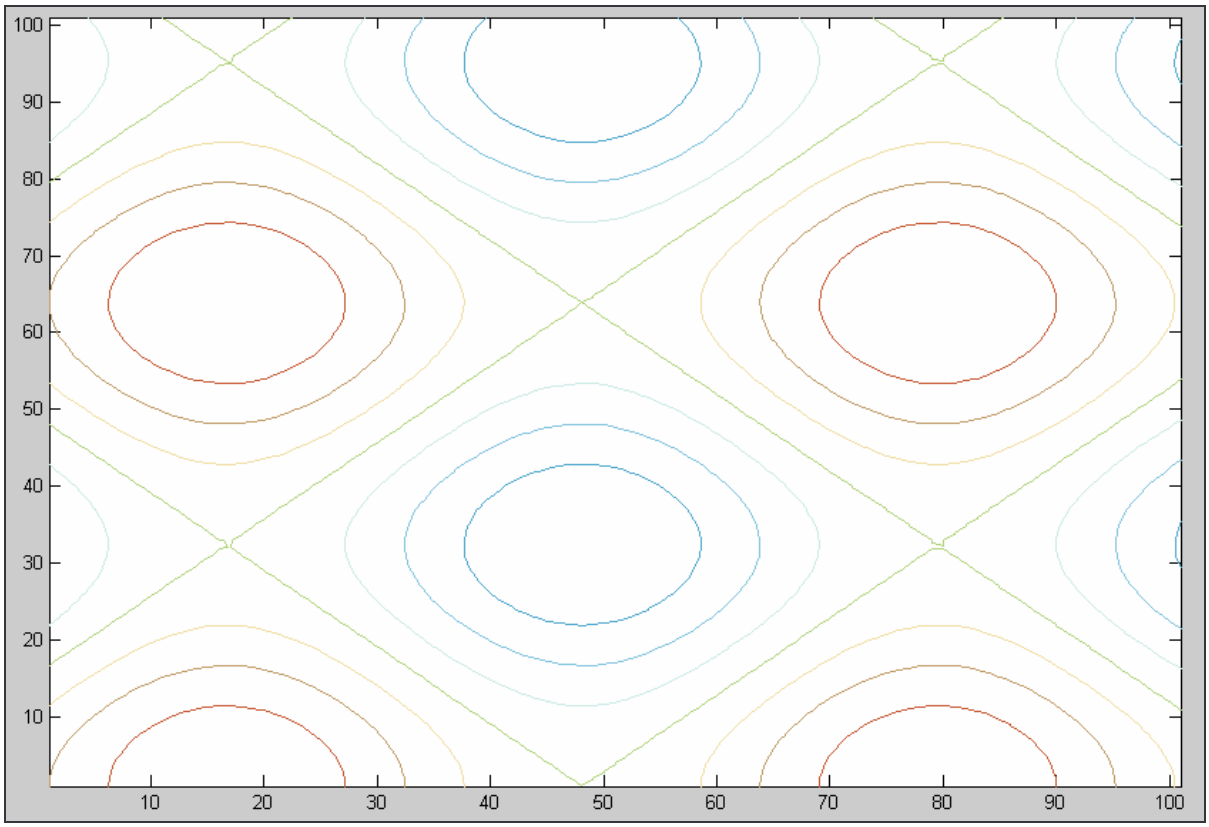
Hacemos uso de los distintos tipos de representación. Representado por orden: la clasica, la clasica solida y la sombreada respectivamente.

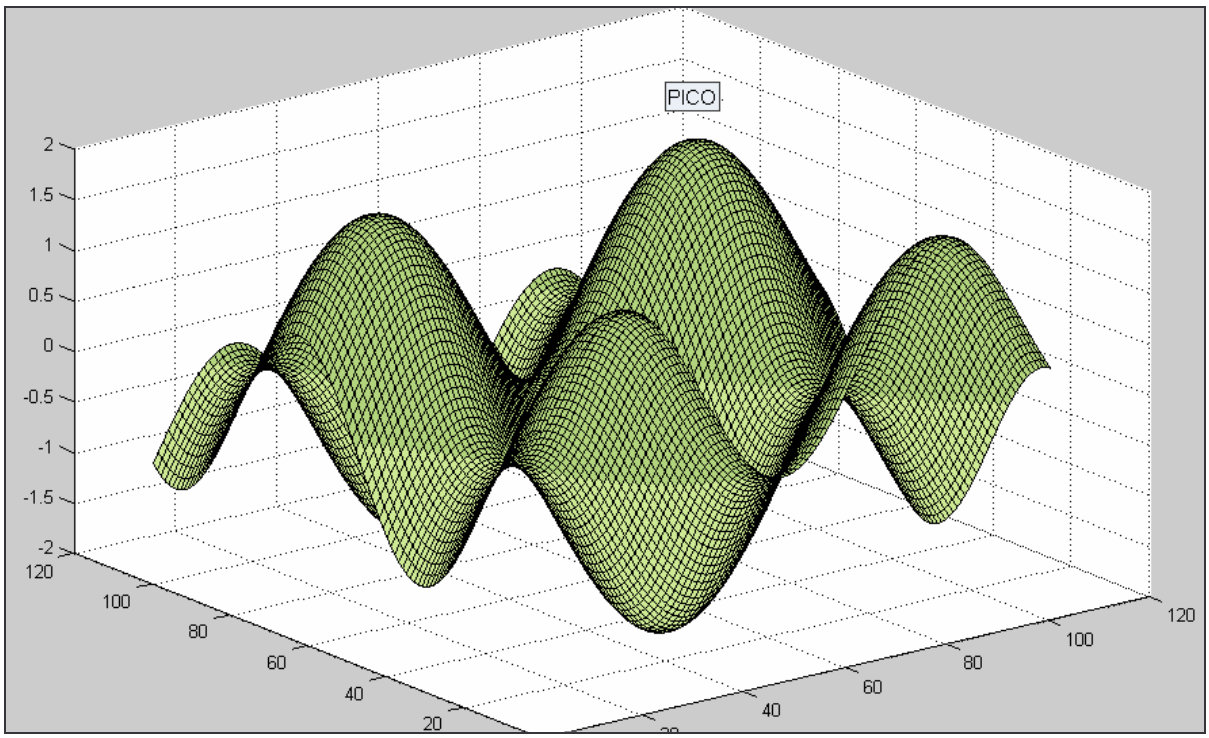



2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
120
100
80
60
40
20
0
0
20
40
60
80
100
120


2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
120
100
80
60
40
20
0
0
20
40
60
80
100
120

Seleccionando el numero 3 , tendremos las curvas de nivel, y en el caso de querer introducir algun letrero, usaremos la opcion 4.

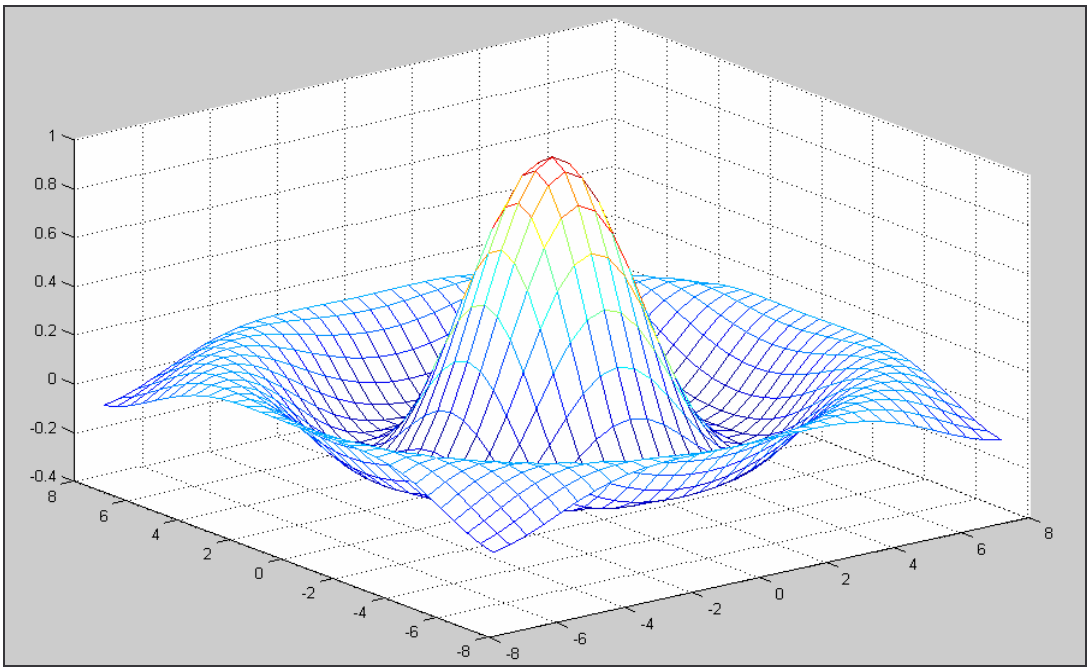




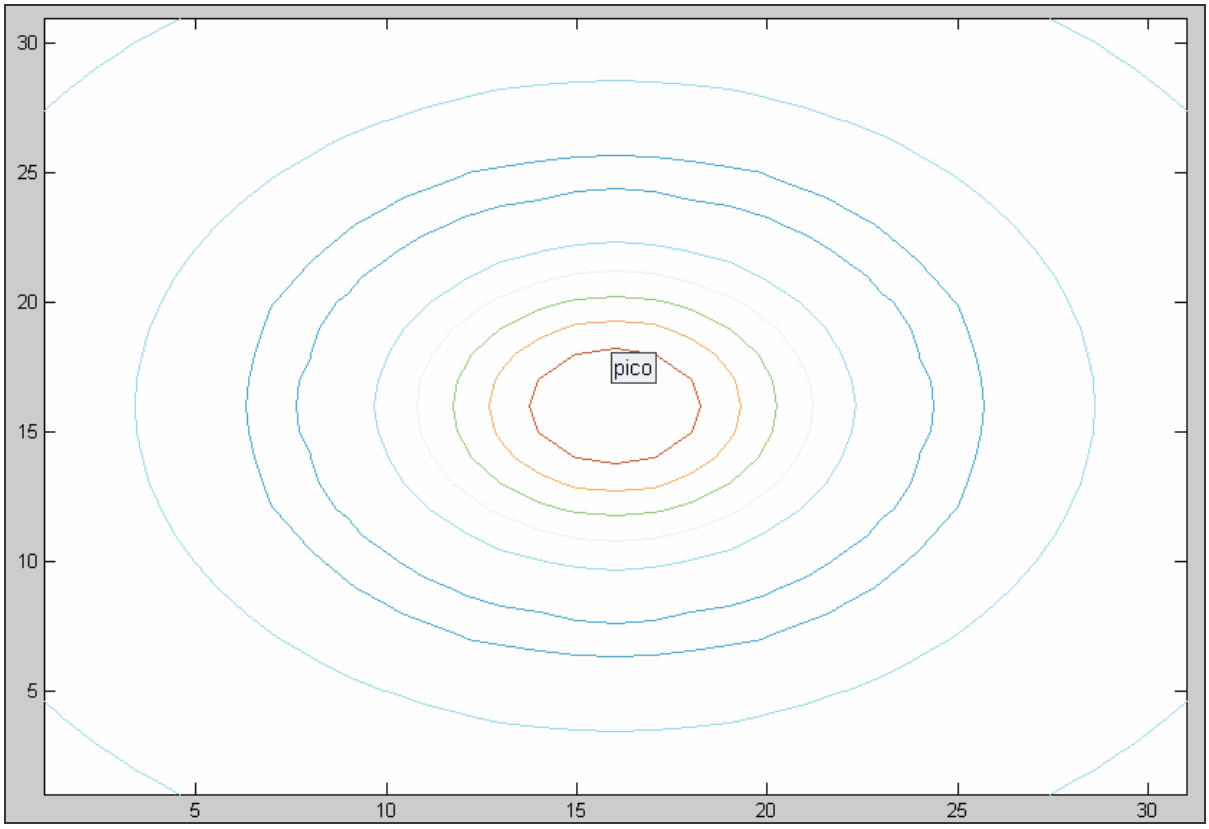
EJEMPLO 2 : Representación de una superficie a traves de una función introducida por fichero.

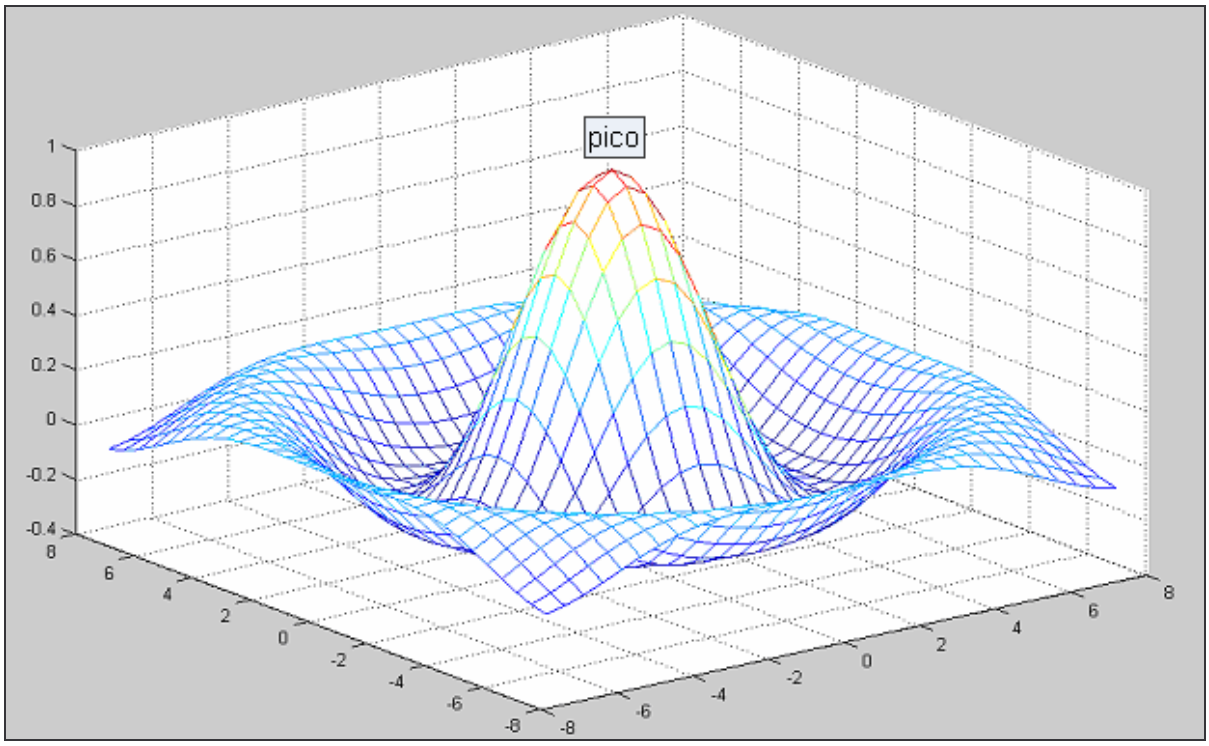
Supongamos ahora dibujo de la función ***sen(r)/r*** (siendo *r=sqrt(x2+y2);* para evitar dividir por 0 se suma al denominador el número pequeño ***eps***). Esta función con forma de sombrero mejicano la introduciremos a traves de un fichero .m que luego ejecutaremos desde nuestro programa eligiendo por tanto la segunda opcion de entrada de datos.

```
>> Representacion grafica de superficies.

Indique como encuentra estructurada la informacion de la superficie a representar.
    1. Ecuacion en (X,Y).
    2. Matriz con coordenadas X,Y,Z.
 1
El fichero debe contener la funcion en terminos de x,y (en minuscula).
La funcion puede ser leidas desde:
    1. Teclado.
    2. Fichero.
 2
Entrada de la funcion en terminos de X,Y.
Escriba el nombre del fichero en la forma - disco:\nombre.ext
Por ejemplo:   A:\DATA.DTA
 funcion.m
Indique el punto inicial de la cuadricula escribiendo las coordenadas X,Y en lineas separadas.
 -8
 -8
Indique el punto final de la cuadricula escribiendo las coordenadas X,Y en lineas separadas.
 8
 8
Indique la distancia entre puntos en el eje X.
 0.1
Indique la distancia entre puntos en el eje Y.
 0.1
```



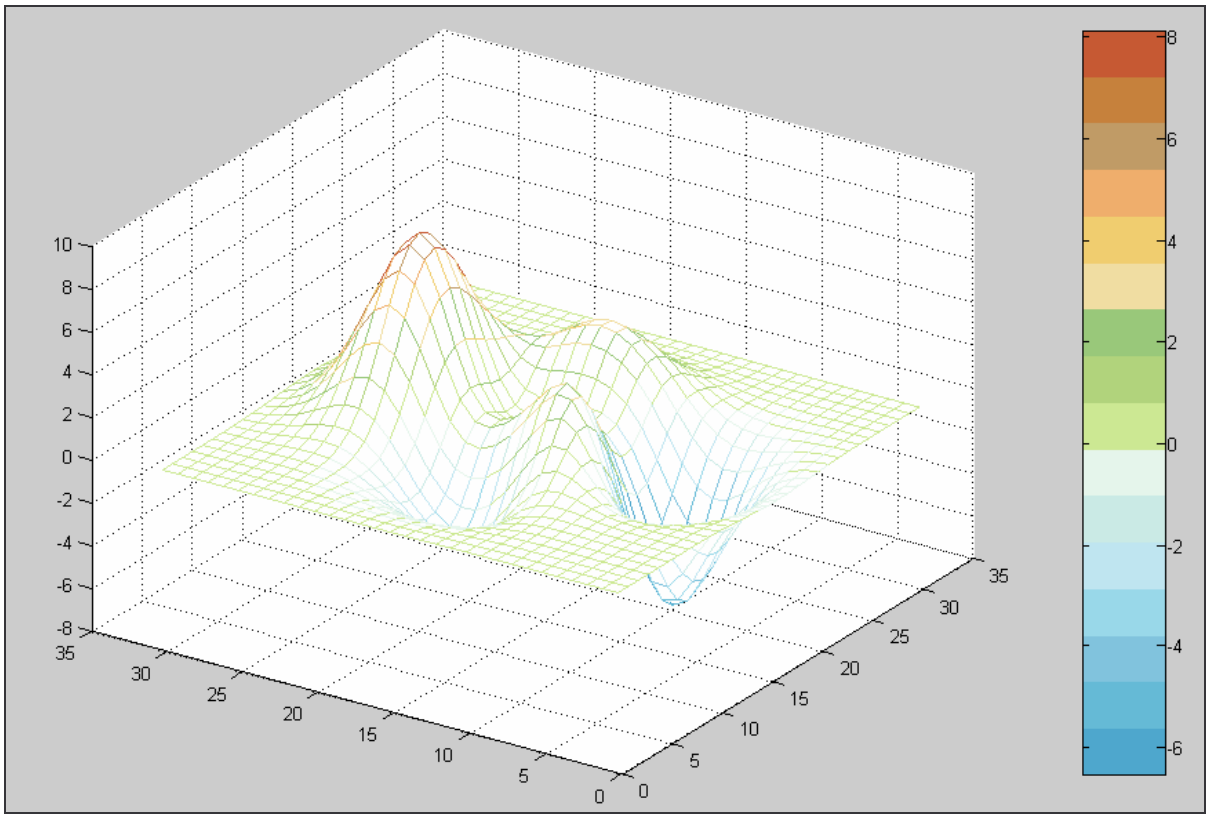
Las curvas de nivel y los letreros nos quedan de la siguiente forma:

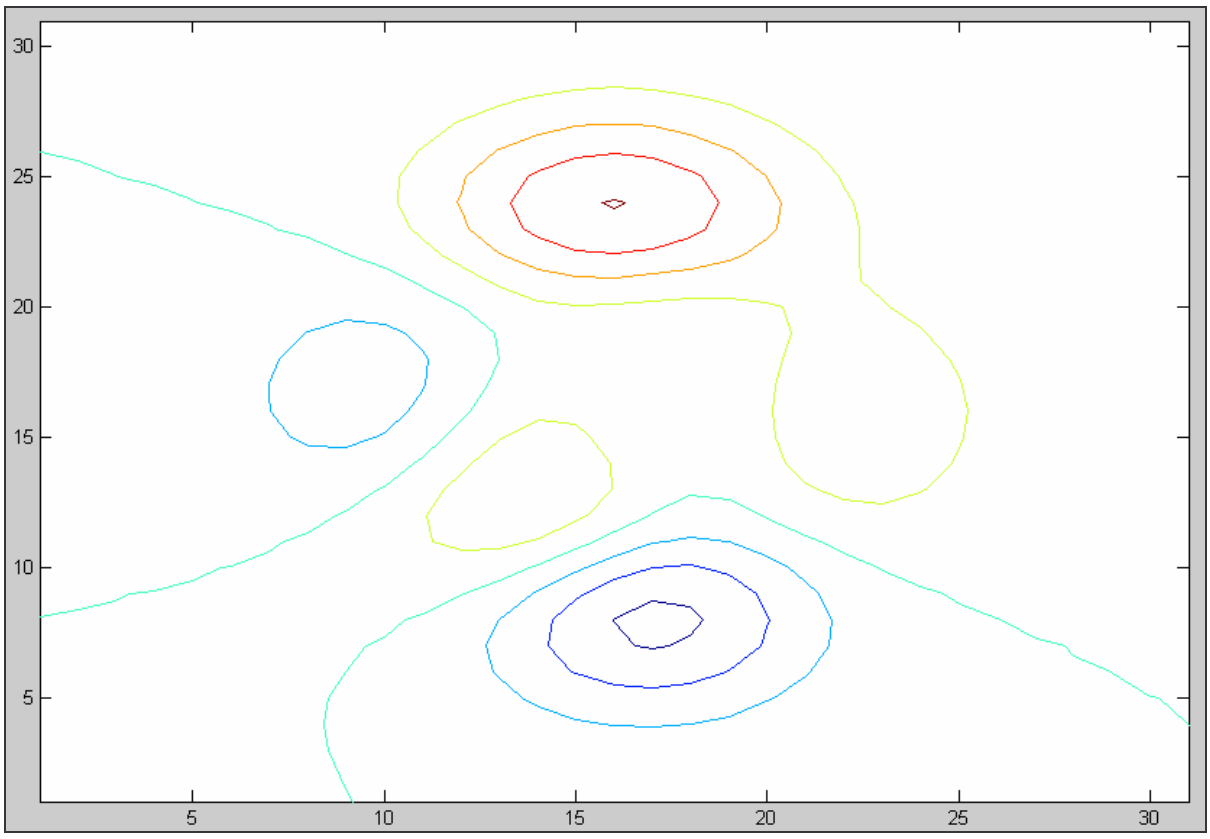


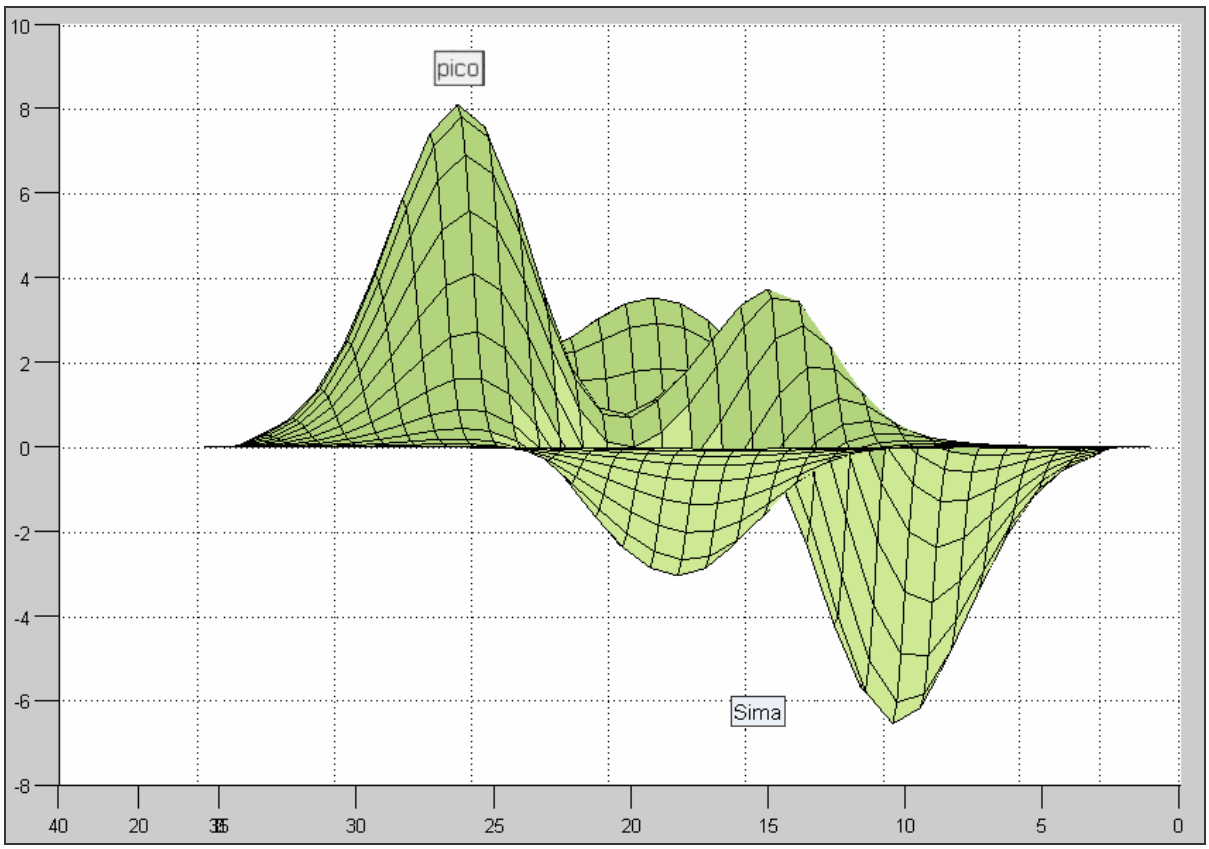


EJEMPLO 3 : Representación de una superficie a traves de una función introducida por teclado. Gráfica de una función con varios extremos (máximos y mínimos)

```
El fichero debe contener la funcion en terminos de x,y (en minuscula).
La funcion puede ser leidas desde:
    1. Teclado.
    2. Fichero.
 1
Introduzca la ecuacion en (x,y):
 3*(1-x).^2.*exp(-(x.^2)-(y+1).^2)-10*(x/5-x.^3-y.^5).*exp(-x.^2-y.^2)-1/3*exp(-(x+1).^2-y.^2)
Indique el punto inicial de la cuadricula escribiendo las coordenadas X,Y en lineas separadas.
 -3
 -3
Indique el punto final de la cuadricula escribiendo las coordenadas X,Y en lineas separadas.
 3
 3
Indique la distancia entre puntos en el eje X.
 0.2
Indique la distancia entre puntos en el eje Y.
 0.2
Indique el tipo de representacion.
    1. Clasica.
    2. Clasica Solida.
    3. Sombreada.
```




30
25
20
15
10
5
5
10
15
20
25
30


pico
Sima
10
8
6
4
2
0
-2
-4
-6
-8
40
20
30
25
20
15
10
5
0

EJEMPLO 4 : Representación de una superficie a traves de una función introducida por matriz.

Por último, vemos el ejemplo de la representacion grafica de una superficie introducida por matriz, y la posterior utilizacion del movimento tridimensional.

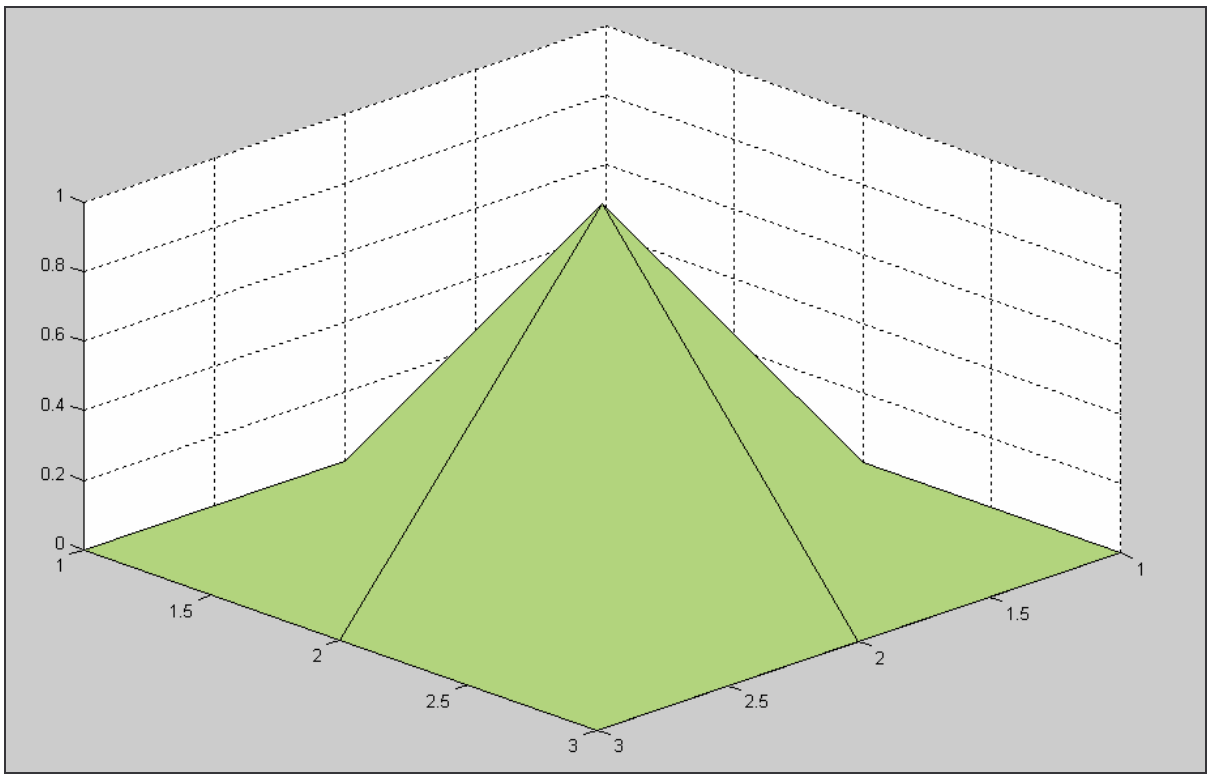
```
>> Representacion grafica de superficies.

Indique como encuentra estructurada la informacion de la superficie a representar.
    1. Ecuacion en (X,Y).
    2. Matriz con coordenadas X,Y,Z.
 2
Entrada de la funcion en terminos de X,Y.
Escriba el nombre del fichero en la forma - disco:\nombre.ext
Por ejemplo:   A:\DATA.DTA
 matriz.m
Escriba el numero de ecuaciones.
 10
Indique el tipo de representacion.
    1. Clasica.
    2. Clasica Solida.
    3. Sombreada.
 1
Seleccione la opcion que prefiera:.
    1.Representar de nuevo la grafica.
    2.Mover grafica en 3D.
    3.Representar curvas de nivel.
    4.Colocar letrero.
    5.Salir.
 3
Seleccione la opcion que prefiera:.
    1.Representar de nuevo la grafica.
    2.Mover grafica en 3D.
    3.Representar curvas de nivel.
    4.Colocar letrero.
    5.Salir.
 1
```

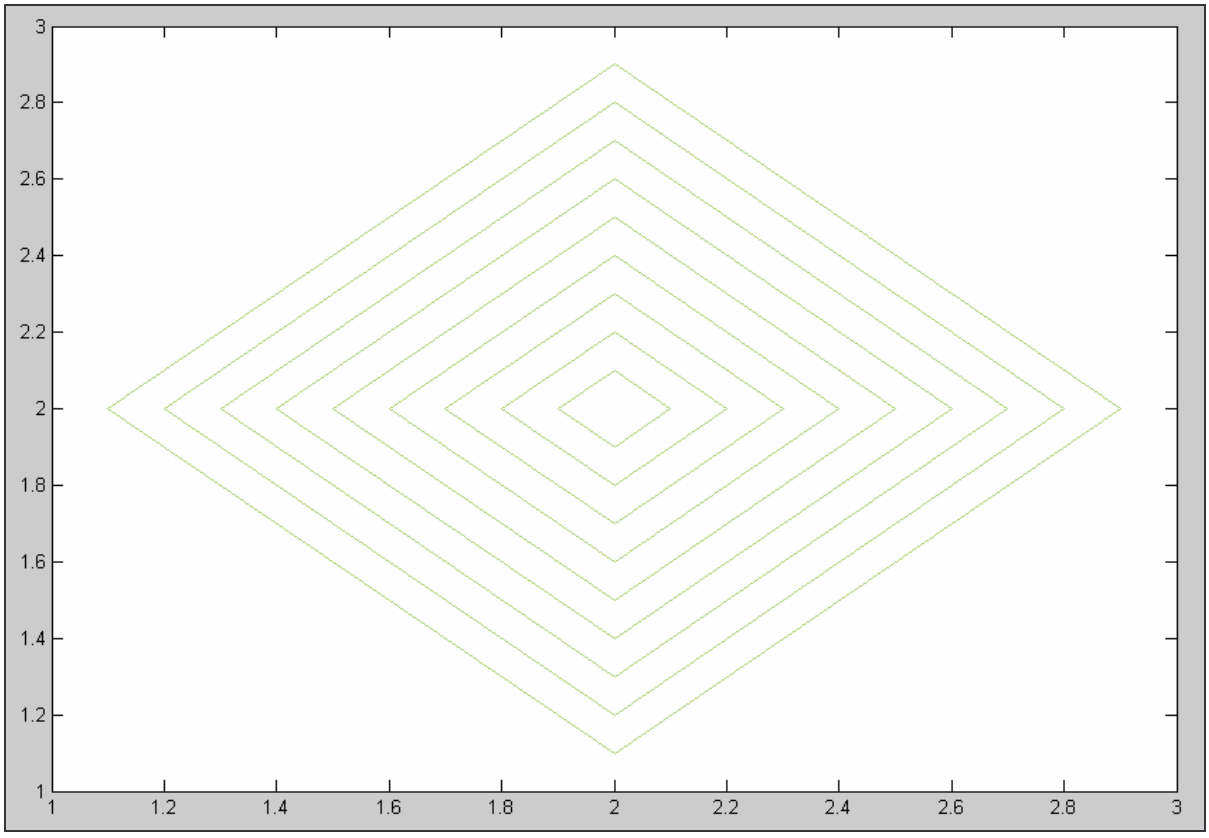
C:\Documents ...

File Edit View Text Debug
Breakpoints Web Window Help

1	0	0	0
2	0	1	0
3	0	2	0
4	1	0	0
5	1	1	1
6	1	2	0
7	2	0	0
8	2	1	0
9	2	2	0
10	3	0	0

Ln 1 Col 1




3
2.8
2.6
2.4
2.2
2
1.8
1.6
1.4
1.2
1
1
1.2
1.4
1.6
1.8
2
2.2
2.4
2.6
2.8
3