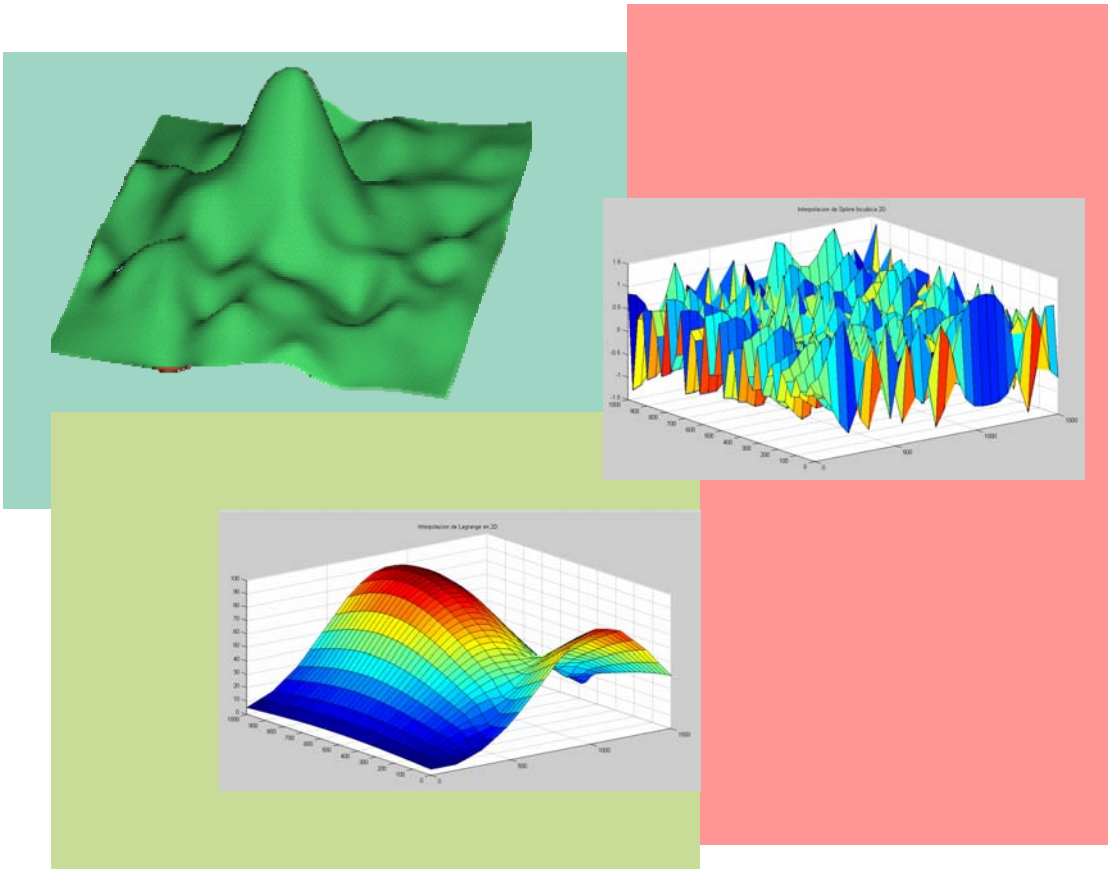
UNIVERSIDAD DE LAS PALMAS
DE GRAN CANARIA

TRABAJO DE LA ASIGNATURA

MÉTODOS NUMÉRICOS



ESCUELA TÉCNICA SUPERIOR DE INGENIEROS INDUSTRIALES

CARMELO J. MUÑOZ RUIZ
JACOB MORALES GARCÍA
MANUEL UCHE SORIA

4º ETSII

CURSO 2006-2007

ÍNDICE

0. ENUNCIADO DEL PROBLEMA

Realizar un programa en MATLAB que construya una interpolación por splines cúbicas en 2-D de una superficie definida digitalmente sobre una malla de rectángulos, tomando como soporte otra malla de cuadriláteros. El trabajo debe considerar los siguientes aspectos y requisitos:

a. Posibilidad de que la superficie esté definida de forma continua o forma discreta. En el primer caso, el programa deberá generar una cuadrícula donde venga definida la superficie de forma discreta, para luego operar como en el segundo caso.

b. Definir una cuadrícula más fina que la del soporte para representar gráficamente dicha superficie.

c. Definir una interpolación spline cúbica bidimensional. Localizar previamente, para cada punto de la malla final, en qué cuadrícula de la malla soporte cae. Tener en cuenta los casos particulares de coincidencia en una arista o en un vértice.

d. Comparar los resultados obtenidos mediante la spline con los de la interpolación de Lagrange.

1. INTRODUCCIÓN

▪ Superficies rectangulares

Una superficie paramétrica producto tensorial se ha definido como la que puede expresarse como combinación lineal de funciones que son el producto tensorial de dos bases univariadas:

$$S(u,v) = \sum_{i=0}^{n} \sum_{j=0}^{m} P_{ij} B_i(u) C_j(v)$$

Donde $P_{ij} = \left(x_{ij}, y_{ij}, z_{ji} \right)$ son puntos (coeficientes) distribuidos según una estructura rectangular y que poseerán mayor o menor significados geométrico dependiendo de las bases elegidas.

Dado que $B_i(u)$ y $C_j(v)$ son polinomios de Lagrange asociados a soportes $S_u = \{u_0, u_1 ... u_n\}$ y $S_v = \{v_0, v_1 ... v_n\}$, los coeficientes $P_{ij} = S(u_i, u_j)$ son puntos sobre la superficie, con lo cual la ecuación anterior se resuelve el problema de interpolación.

▪ Interpolación de Lagrange en dos dimensiones

Puesto que en la parte final de nuestro algoritmo hemos recurrido a la interpolación de Lagrange en 2D, merece la pena exponer brevemente las fórmulas que aparecen dicho algoritmo.

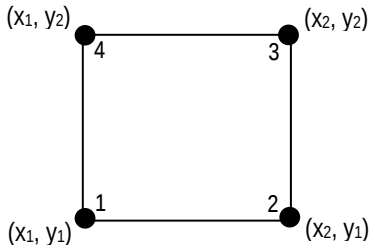
Dado un punto *(X,Y)* en coordenadas de la topografía original calcula la topografía en ese punto a través de una interpolación de Lagrange de cuatro puntos. Esta función sólo es válida para rectángulos de lados paralelos a los ejes, como en nuestro caso.

Entradas:

X: coordenada x del punto a interpolar.
Y: coordenada y del punto a interpolar.

Salidas:

Z: resultado del punto a interpolar.



La fórmula de interpolación de Lagrange en 2D será:

$$\boxed{Z = N_1 \cdot Z_1 + N_2 \cdot Z_2 + N_3 \cdot Z_3 + N_4 \cdot Z_4}$$

Siendo sus términos:

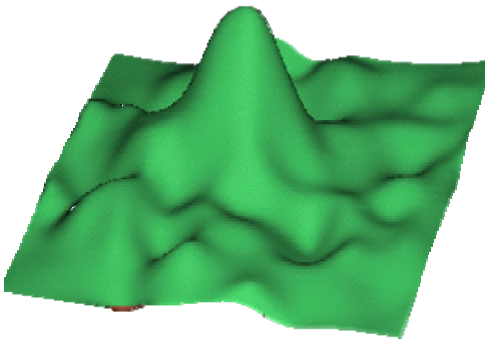
$$N_1 = \frac{(x_2 - x)(y_2 - y)}{D_1} \qquad N_2 = \frac{(x_1 - x)(y_2 - y)}{D_2}$$

$$N_3 = \frac{(x_1 - x)(y_1 - y)}{D_3} \qquad N_4 = \frac{(x - x_2)(y_1 - y)}{D_4}$$

Y también: $D_1 = (x_2 - x_1)(y_2 - y_1)$ $\quad D_2 = -D_1$ $\quad D_3 = D_1$ $\quad D_4 = D_1$

- **Interpolación con superficies B-Spline:**

El esquema para obtener una superficie B-Spline interpolando un conjunto de puntos Q_{ij} se logra de la siguiente forma.



En primer lugar, es conveniente que el número de puntos a interpolar y su estructura rectangular coincida con el número de funciones base. Aun así, todavía se tendrá libertad para elegir el grado para cada una de las variables, la que determinará el número de nodos de los correspondientes soportes de definición de B-Splines.

Por ejemplo, si Q_{ij} $0 \le i \le n$, $0 \le j \le m$ son los puntos a interpolar y buscamos una superficie B-Spline de grado p en u y q en v, entonces el soporte en u deberá contener $r + 1 = n + p + i$ nodos y el soporte en v, $s + 1 = m + q + 2$, es decir:

$$S_u = \{u_0, \ldots, u_{n+p+1}\} \qquad S_v = \{v_0, \ldots, v_{m+q+1}\}$$

Y la superficie será de la forma: $S(u,v) = P_{ij} B^p{}_i(u) B^q{}_j(v)$

En las condiciones planteadas queda precisado el número de nodo en cada soporte de definición de los B-Splines y que pueden existir diferentes alterativas para elegir los mismos, lo que significa operar con unas funciones B-Splines u otras.

Finalmente habrá que especificar un soporte de interpolación en cada variable. Una vez definidos los soportes de nodos para los B-Splines y los soportes de interpolación. Se comenzará, por ejemplo con la variable u, se realizan $m + 1$ interpolaciones, cada una de $n + 1$ puntos:

$$\begin{array}{l} Q_{00}, Q_{10}, \ldots, Q_{n0} \quad \text{interpola con} \quad C_0(u) = \sum_{i=0}^{n} R_{i0} B^p{}_i(u) \\ Q_{01}, Q_{11}, \ldots, Q_{n1} \quad \text{interpola con} \quad C_1(u) = \sum_{i=0}^{n} R_{i1} B^p{}_i(u) \\ \qquad\qquad \ldots \\ Q_{0m}, Q_{1m}, \ldots, Q_{nm} \quad \text{interpola con} \quad C_m(u) = \sum_{i=0}^{n} R_{im} B^p{}_i(u) \end{array}$$

A continuación., interpolamos en la variable v los coeficientes R_{ij} por columnas, es decir:

$$\begin{array}{c} R_{00}, R_{10},\ldots,R_{n0} \quad \text{interpola con} \quad D_0(v) = \sum_{j=0}^{n} P_{j0} B^q_j(v) \\ R_{01}, R_{11},\ldots,R_{n1} \quad \text{interpola con} \quad D_1(v) = \sum_{j=0}^{n} P_{j1} B^q_j(v) \\ \ldots \\ R_{0m}, R_{1m},\ldots,R_{nm} \text{ interpola con} \quad D_m(v) = \sum_{j=0}^{n} P_{jm} B^q_j(v) \end{array}$$

Los coeficientes P_{ij} obtenidos son los puntos de control de la superficie B-Spline de interpolación, si:

$$S(u,v) = \sum_{i=0}^{n} \sum_{j=0}^{m} P_{ij} B^p_i(u)\, B^q_j$$

Entonces, llamando $u_0,\ldots,u_n$ y $v_0,\ldots,v_m$ a los nodos de interpolación:

$$S(u_k,v_k) = \sum_{i=0}^{n} [\sum_{j=0}^{m} P_{ij}\, B^q_j(v_h)]\, B^p_i(u_k) = \sum_{i=0}^{n} D_i\,(v_h)\, B_i(u_k) = \sum_{i=0}^{n} R_{ih}\, B^p_i(u_k) = C_h(u_k) = Q_{kh}$$

Observemos que el proceso es simétrico en ambas variables, por lo que igualmente podría haberse comenzado con las interpo1aciones en v, obteniéndose al final los mismos puntos P_{ij}.

2. ALGORITMO

- ***Comentario del algoritmo***

Declaración de variables:

```
syms('F','OK','A','B','N','FLAG','FLAG1','NAME','OUP','E');
syms('M','FLAG0','C','X','s','D','NTOT');
syms('xx','yy','zz','xi','yi','zi','Zi');
syms('IP1','JP1','IP2','JP2','IP3','JP3','IP4','JP4');
syms('X1','YI','Z1','X2','Y2','Z2','X3','Y3','Z3','X4','Y4','Z4');
syms('D1','D2','D3','D4','N1','N2','N3','N4');
TRUE = 1;
FALSE = 0;
```

Selección tipo de entrada de datos

```
fprintf(1,'Formula de Interpolacion Spline bicubica en 2D\n');
 OK = FALSE;
 while OK == FALSE
    fprintf(1,'Elija la forma de entrada de datos:\n');
    fprintf(1,'1.  Generar datos usando una funcion F\n');
    fprintf(1,'2. Entrada desde un fichero de texto\n');
    fprintf(1,'Escriba 1 o 2 \n');
    FLAG = input(' ');
    if FLAG == 1 | FLAG == 2
        OK = TRUE;
    end
 end
```

Mientras OK sea falso, el programa nos pide la forma de entrada de los datos. El usuario puede elegir dos maneras. La primera es usando una función para generar los datos y la segunda forma sería la entrada de datos a través de un fichero de texto (.DTA, .doc,…) previamente creado por el usuario.

Tal y como se observa en el algoritmo, la variable FLAG tomará un 1 (si queremos la primera opción, tecleamos 1) o bien, un 2 (si queremos que la entrada de datos se produzca por fichero, tecleamos 2). Una vez terminada la elección la variable OK se convierte en verdadero si tecleamos 1 o 2, saltando a la siguiente fase del programa: Inserción de datos.

Inserción de datos y creación de la primera malla

➲ *Si el usuario elige usar una función para generar los datos, la variable FLAG se habrá puesto en 1 y por tanto; después de inicializar la variable OK a falso, el programa pedirá la longitud de X e Y, el número de puntos X e Y y la función F(x,y). Pero vamos a ir por partes.*

- *Cuando el usuario teclea las longitudes de X e Y el programa las almacena en A y B respectivamente.*
- *Cuando el usuario teclea el número de puntos en X y en Y se guardan en N y M respectivamente.*

Una vez se han introducido estos cuatro valores, el programa chequea si, tanto las longitudes como el número de puntos son mayores que cero. Si se verifica dicha condición, la variable OK pasará a verdadero. En caso contrario, seguirá pidiendo valores hasta que se cumpla la condición.

```
if FLAG == 1
 OK = FALSE;
 while OK == FALSE
    fprintf(1,'Escriba la longitud de X y de Y en lineas
separadas.\n');
    A = input(' ');
    B = input(' ');
    fprintf(1,'Escriba el numero de puntos en X y en Y en lineas
separadas.\n');
    N = input(' ');
    M = input(' ');
    if A > 0 & B > 0 & N > 0 & M > 0
       OK = TRUE;
    end
 end
```

El siguiente paso es calcular los incrementos (longitudes de los rectángulos que forman la malla sobre la que trabajaremos): el incremento de x (asociada a la longitud X almacenada en A y al número de puntos almacenado en N) y el incremento de y (asociado a la longitud Y almacenada en B y al número de puntos almacenado en M).

```
 C = A/(N-1);
 D = B/(M-1);
```

Llegamos a la creación la malla de puntos cuyas coordenadas x e y se almacenarán en las matrices xx e yy respectivamente. Para ello usamos la función meshgrid que se observa a continuación:

```
 [xx,yy] = meshgrid(0:C:A,0:D:B);
 zz = zeros(M,N);
```

Lo que hace esta función es crear una malla de puntos de dimensiones AxB, donde el paso de un punto a otro en x es C y el paso de un punto a otro en y es D, y guardar en las matrices xx e yy las coordenadas x e y de los puntos de la malla respectivamente. Inicializamos además una matriz de dimensiones MxN donde se guardará el valor de la función en cada punto de la malla:

Seguimos en la fase de inserción de datos, y es el turno de teclear la función a interpolar. Esta función la guardamos en F y barremos los valores de x e y pero por columnas, (véanse los bucles for).

```
    fprintf(1,'Escriba la funcion F(x,y) en terminos de x e y\n');
    fprintf(1,'Por ejemplo: y-x^2+1 \n');
    s = input(' ','s');
    F = inline(char(s),'x','y');
    for J = 1 : N
        for I = 1 : M
            zz(I,J) = F(xx(I,J),yy(I,J));
        end;
    end;
```

Respecto a la elección de la forma de salida de datos, el usuario puede elegir si prefiere por pantalla o, por el contrario, mediante fichero de texto. En el caso de elegir por pantalla se teclea 1 y la variable FLAG0 adoptará el valor 1, y en caso contrario, se teclea 2 y FLAG0 tomará el valor 2.

Si la elección es por fichero de texto, se pedirá al usuario que escriba el nombre del fichero y la dirección donde desea que se guarde dicho archivo de texto. El nombre del archivo en cuestión se guarda en la variable NAME.

```
    fprintf(1,'Elija la forma de salida de datos:\n');
    fprintf(1,'1. Pantalla\n');
    fprintf(1,'2. Fichero de texto\n');
    fprintf(1,'Escriba 1 o 2\n');
    FLAG0 = input(' ');
    if FLAG0 == 2
        fprintf(1,'Escriba el nombre del fichero de la forma -
disco:\\nombre.ext\n');
        fprintf(1,'Por ejemplo   A:\\OUTPUT.DTA\n');
        NAME = input(' ','s');
        OUP = fopen(NAME,'wt');
    else
        OUP = 1;
    end;
    fprintf(OUP, '%11.5f %11.5f %11.5f %11.5f\n', A,B,N,M);
    for J = 1 : N
        for I = 1 : M
            fprintf(OUP, '%11.5f %11.5f %11.5f\n',
xx(I,J),yy(I,J),zz(I,J));
        end;
    end;
    if OUP ~= 1
        fclose(OUP);
         fprintf(1,'Fichero de salida creado satisfactoriamente
\n',NAME);
    end;
 end;
```

➲ *Si, por el contrario, el usuario elige usar un fichero de texto deberá teclear 2 y la variable FLAG se habrá puesto en 2, y por tanto, ya no será necesario introducir el número de puntos de X e Y ni tampoco las longitudes en X e Y, pues estos datos estarán en el fichero que el usuario haya creado.*

De esta forma, el programa pregunta si se ha creado el fichero de texto con las características necesarias para su correcta lectura, esto es, en tres columnas organizadas de tal manera que la primera fila tenga cuatro columnas que contenga la longitud de X, la longitud de Y y el número de puntos de X e Y y el resto de las filas tenga en su primera columna la coordenada x del punto, la coordenada y en la segunda columna y el valor de la función en la tercera. Como se trabaja por columnas, primero se ha de poner para el primer valor de x todos los valores de y con sus respectivos valores de la función, luego el segundo valor de x y se repite la operación, y así sucesivamente. Si esto es así, E será igual a "Y" o "y", teniéndose que teclear el nombre del fichero en cuestión.

Tal y como se hizo antes, el nombre se guarda en la variable NAME, el primer dato que lee (longitud de X) se guarda en A, el siguiente (longitud de Y) se guarda en B, el siguiente (número de puntos en X) se almacena en N y el último dato de la primera fila (número de puntos en Y) se almacena en M.

Por último, y como ya se ha explicado, se calculan los incrementos en dirección x (C) y en dirección y (D).

```
if FLAG == 2
    fprintf(1,'¿Esta creado un fichero de texto con los datos en tres
columnas\n');
     fprintf(1,'donde la primera fila contiene la longitud de X, la
longitud de Y\n');
     fprintf(1,'y el numero de puntos en X e Y?\n');
    fprintf(1,'Escriba Y o N\n')
    E = input(' ','s');
    if E == 'Y' | E == 'y'
        fprintf(1,'Escriba el nombre del fichero de la forma - ');
        fprintf(1,'disco:\\nombre.ext\n');
        fprintf(1,'Por ejemplo:   A:\\DATA.DTA\n');
        NAME = input(' ','s');
        INP = fopen(NAME,'rt');
        A = fscanf(INP, '%f',1);
        B = fscanf(INP, '%f',1);
        N = fscanf(INP, '%f',1);
        M = fscanf(INP, '%f',1);
        C = A/(N-1);
        D = B/(M-1);
```

Inicializamos una nueva matriz X a cero de dimensiones M,N,3 (en realidad son tres matrices de MxN)para hacer la lectura de forma más ordenada y volvemos a utilizar la función meshgrid de la misma manera que se explicó anteriormente.

```
        X = zeros(M,N,3);
        [xx,yy] = meshgrid(0:C:A,0:D:B);
        zz = zeros(M,N);
        OK = FALSE;
        while OK == FALSE
            for J = 1 : N
                for I = 1 : M
                    for K = 1 : 3
                        X(I,J,K) = fscanf(INP, '%f',1);
                    end;
                end;
            end;
            OK = TRUE;
            fclose(INP);
        end
    end;
```

Guardamos los datos de X(I,J,3) en zz(I,J), por columnas y de manera ordenada.

```
    for J = 1 : N
        for I = 1 : M
            zz(I,J)=X(I,J,3);
        end;
    end;
end;
```

Creación de la segunda malla

```
OK = FALSE;
 while OK == FALSE
 fprintf(1,'Introduzca un nuevo numero de puntos para x y para y en
lineas separadas\n');
  fprintf(1,'Recuerde que deben ser mayores que los anteriores\n');
 NN = input(' ');
 MM = input(' ');
    if  NN > N & MM > M
        zi = zeros(MM,NN);
        OK = TRUE;
    end;
 end;
```

En la creación de la una segunda malla, el programa pide al usuario que introduzca un nuevo número de puntos para x y para y; valores que deberán ser mayores que los que hayan sido anteriormente introducidos (bien por fichero o por pantalla).

En la variable NN guarda el nuevo número de puntos de x y en MM guarda el nuevo número de puntos de Y, verificando posteriormente si estos nuevos valores son mayores que los anteriores. Si todo está bien, inicializa una nueva matriz zi de dimensiones MM y NN y pone OK en verdadero para salir del bucle.

```
 CC = A/(NN-1);
 DD = B/(MM-1);
 [xi,yi] = meshgrid(0:CC:A,0:DD:B);
```

A continuación se calculan los nuevos incrementos CC y DD correspondientes a los nuevos números de puntos de X e Y respectivamente. Y vuelve a hacer lo mismo con la función meshgrid, sólo que ahora cambian los incrementos.

Solución y representación gráfica

```
zi = interp2(xx,yy,zz,xi,yi,'spline');
```

Vamos a explicar, brevemente, el funcionamiento del comando interp2:

El comando interp2 interpola entre los puntos de referencia. Respecto a la función ***Zi = interp2(xx,yy,zz,xi,zi,'spline')****: lo que hace es interpolar la función definida para la malla de puntos cuyas coordenadas x e y están guardadas en las matrices xx e yy, y cuyo valor de la función en cada punto está guardado en zz, usando como soporte la malla más fina creada cuyas coordenadas x e y se encuentran guardadas respectivamente en las matrices xi e yi. Se pueden utilizar varios métodos:* ***nearest*** *("lo más cerca posible"): La interpolación vecina más cercana;* ***linear****: Interpolación lineal (por defecto);* ***spline****: interpolación cúbica de spline;* ***cubic****: interpolación cúbica.*

```
figure(1);
surf(xi,yi,zi);
title('Interpolacion de Spline bicubica 2D');

fprintf(1,'Elija si quiere hacer la interpolación de Lagrange:\n');
    fprintf(1,'1. No\n');
    fprintf(1,'2. Si\n');
    fprintf(1,'Escriba 1 o 2 \n');

FLAG1 = input(' ');
if FLAG1==1
    fprintf(1,'Fin del programa\n');
end
```

Por último, se crea la figura 1 que representamos mediante la función surf y el programa nos da la opción de comparar esta interpolación con la interpolación de Lagrange en 2D. En el caso de elegir no, FLAG1 se pondrá a 1 y se finalizará el programa. Si por el contrario quisiéramos compararla con lo que se obtendría al hacer la interpolación de Lagrange, el usuario tecleará un 2 y aparecería en pantalla la gráfica 2 con la interpolación de Lagrange en 2D.

Las fórmulas usadas para la realización del algoritmo de Lagrange en 2D son las que se encuentran en la introducción teórica del trabajo y el procedimiento de interpolación es el mismo que el descrito en el trabajo del año anterior sobre interpolación de Lagrange de funciones de 2 variables y que no es el objeto principal de este trabajo.

```
if FLAG1==2
    Zi = zeros(MM,NN);
    for J = 1 : NN
        for I = 1 : MM
            IP1 = fix(yi(I,J)/D)+1;
            JP1 = fix(xi(I,J)/C)+1;
        if I==MM
            IP1 = fix(yi(I,J)/D);
        end;
        if J==NN
            JP1 = fix(xi(I,J)/C);
        end;
        IP2 = IP1;
        JP2 = JP1+1;
        IP3 = IP1+1;
        JP3 = JP1+1;
        IP4 = IP1+1;
        JP4 = JP1;

        X1 = xx(IP1,JP1);
        Y1 = yy(IP1,JP1);
        X2 = xx(IP3,JP3);
        Y2 = yy(IP3,JP3);
        Z1 = zz(IP1,JP1);
        Z2 = zz(IP2,JP2);
        Z3 = zz(IP3,JP3);
        Z4 = zz(IP4,JP4);
```

```
        D1 = ((X2-X1)*(Y2-Y1));
        D2 = -D1;
        D3 = D1;
        D4 = D1;

        N1 = (((X2-xi(I,J))*(Y2-yi(I,J)))/D1);
        N2 = (((X1-xi(I,J))*(Y2-yi(I,J)))/D2);
        N3 = (((X1-xi(I,J))*(Y1-yi(I,J)))/D3);
        N4 = (((xi(I,J)-X2)*(Y1-yi(I,J)))/D4);

        Zi(I,J) = (N1*Z1+N2*Z2+N3*Z3+N4*Z4);
    end;
 end;
figure(2);
surf(xi,yi,Zi);
title('Interpolacion de Lagrange en 2D');
end
```

- ***Exposición del algoritmo completo***

```
 syms('F','OK','A','B','N','FLAG','FLAG1','NAME','OUP','E');
 syms('M','FLAG0','C','X','s','D','NTOT');
 syms('xx','yy','zz','xi','yi','zi','Zi');
 syms('IP1','JP1','IP2','JP2','IP3','JP3','IP4','JP4');
 syms('X1','YI','Z1','X2','Y2','Z2','X3','Y3','Z3','X4','Y4','Z4');
 syms('D1','D2','D3','D4','N1','N2','N3','N4');
 TRUE = 1;
 FALSE = 0;
 fprintf(1,'Formula de Interpolacion Spline bicubica en 2D\n');
 OK = FALSE;
 while OK == FALSE
    fprintf(1,'Elija la forma de entrada de datos:\n');
    fprintf(1,'1.  Generar datos usando una funcion F\n');
    fprintf(1,'2. Entrada desde un fichero de texto\n');
    fprintf(1,'Escriba 1 o 2 \n');
    FLAG = input(' ');
    if FLAG == 1 | FLAG == 2
        OK = TRUE;
    end
 end

if FLAG == 1
 OK = FALSE;
 while OK == FALSE
    fprintf(1,'Escriba la longitud de X y de Y en lineas
separadas.\n');
    A = input(' ');
    B = input(' ');
    fprintf(1,'Escriba el numero de puntos en X y en Y en lineas
separadas.\n');
    N = input(' ');
    M = input(' ');
    if A > 0 & B > 0 & N > 0 & M > 0
       OK = TRUE;
    end
 end
```

```
 C = A/(N-1);
 D = B/(M-1);
 [xx,yy] = meshgrid(0:C:A,0:D:B);
 zz = zeros(M,N);
    fprintf(1,'Escriba la funcion F(x,y) en terminos de x e y\n');
    fprintf(1,'Por ejemplo: y-x^2+1 \n');
    s = input(' ','s');
    F = inline(char(s),'x','y');
    for J = 1 : N
        for I = 1 : M
            zz(I,J) = F(xx(I,J),yy(I,J));
        end;
    end;
    fprintf(1,'Elija la forma de salida de datos:\n');
    fprintf(1,'1. Pantalla\n');
    fprintf(1,'2. Fichero de texto\n');
    fprintf(1,'Escriba 1 o 2\n');
    FLAG0 = input(' ');
    if FLAG0 == 2
        fprintf(1,'Escriba el nombre del fichero de la forma -
disco:\\nombre.ext\n');
        fprintf(1,'Por ejemplo   A:\\OUTPUT.DTA\n');
        NAME = input(' ','s');
        OUP = fopen(NAME,'wt');
    else
        OUP = 1;
    end;
    fprintf(OUP, '%11.5f %11.5f %11.5f %11.5f\n', A,B,N,M);
    for J = 1 : N
        for I = 1 : M
            fprintf(OUP, '%11.5f %11.5f %11.5f\n',
xx(I,J),yy(I,J),zz(I,J));
        end;
    end;
    if OUP ~= 1
        fclose(OUP);
         fprintf(1,'Fichero de salida creado satisfactoriamente
\n',NAME);
    end;
 end;

if FLAG == 2
    fprintf(1,'¿Esta creado un fichero de texto con los datos en tres
columnas\n');
     fprintf(1,'donde la primera fila contiene la longitud de X, la
longitud de Y\n');
     fprintf(1,'y el numero de puntos en X e Y?\n');
    fprintf(1,'Escriba Y o N\n')
    E = input(' ','s');
    if E == 'Y' | E == 'y'
        fprintf(1,'Escriba el nombre del fichero de la forma - ');
        fprintf(1,'disco:\\nombre.ext\n');
        fprintf(1,'Por ejemplo:   A:\\DATA.DTA\n');
        NAME = input(' ','s');
        INP = fopen(NAME,'rt');
        A = fscanf(INP, '%f',1);
        B = fscanf(INP, '%f',1);
        N = fscanf(INP, '%f',1);
        M = fscanf(INP, '%f',1);
        C = A/(N-1);
        D = B/(M-1);
```

```
        X = zeros(M,N,3);
        [xx,yy] = meshgrid(0:C:A,0:D:B);
        zz = zeros(M,N);
        OK = FALSE;

        while OK == FALSE
            for J = 1 : N
                for I = 1 : M
                    for K = 1 : 3
                        X(I,J,K) = fscanf(INP, '%f',1);
                    end;
                end;
            end;
            OK = TRUE;
            fclose(INP);
        end
    end;

    for J = 1 : N
        for I = 1 : M
            zz(I,J)=X(I,J,3);
        end;
    end;
 end;
  OK = FALSE;

 while OK == FALSE
 fprintf(1,'Introduzca un nuevo numero de puntos para x y para y en
lineas separadas\n');
  fprintf(1,'Recuerde que deben ser mayores que los anteriores\n');
 NN = input(' ');
 MM = input(' ');
    if  NN > N & MM > M
        zi = zeros(MM,NN);
        OK = TRUE;
    end;
 end;
 CC = A/(NN-1);
 DD = B/(MM-1);
 [xi,yi] = meshgrid(0:CC:A,0:DD:B);
 zi = interp2(xx,yy,zz,xi,yi,'spline');


figure(1);
surf(xi,yi,zi);
title('Interpolacion de Spline bicubica 2D');

fprintf(1,'Elija si quiere hacer la interpolación de Lagrange:\n');
    fprintf(1,'1. No\n');
    fprintf(1,'2. Si\n');
    fprintf(1,'Escriba 1 o 2 \n');
FLAG1 = input(' ');
if FLAG1==1
    fprintf(1,'Fin del programa\n');
end
```

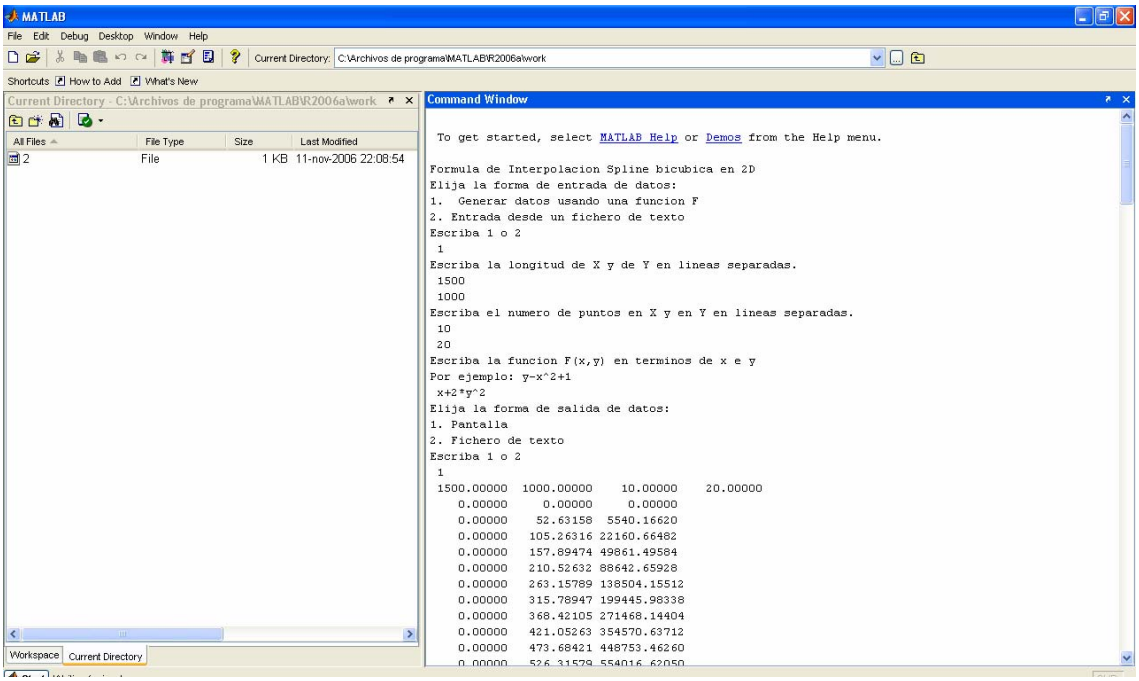
```
if FLAG1==2
    Zi = zeros(MM,NN);
    for J = 1 : NN
        for I = 1 : MM
            IP1 = fix(yi(I,J)/D)+1;
            JP1 = fix(xi(I,J)/C)+1;
        if I==MM
            IP1 = fix(yi(I,J)/D);
        end;
        if J==NN
            JP1 = fix(xi(I,J)/C);
        end;

        IP2 = IP1;
        JP2 = JP1+1;
        IP3 = IP1+1;
        JP3 = JP1+1;
        IP4 = IP1+1;
        JP4 = JP1;

        X1 = xx(IP1,JP1);
        Y1 = yy(IP1,JP1);
        X2 = xx(IP3,JP3);
        Y2 = yy(IP3,JP3);
        Z1 = zz(IP1,JP1);
        Z2 = zz(IP2,JP2);
        Z3 = zz(IP3,JP3);
        Z4 = zz(IP4,JP4);

        D1 = ((X2-X1)*(Y2-Y1));
        D2 = -D1;
        D3 = D1;
        D4 = D1;

        N1 = (((X2-xi(I,J))*(Y2-yi(I,J)))/D1);
        N2 = (((X1-xi(I,J))*(Y2-yi(I,J)))/D2);
        N3 = (((X1-xi(I,J))*(Y1-yi(I,J)))/D3);
        N4 = (((xi(I,J)-X2)*(Y1-yi(I,J)))/D4);

        Zi(I,J) = (N1*Z1+N2*Z2+N3*Z3+N4*Z4);

    end;
 end;
figure(2);
surf(xi,yi,Zi);
title('Interpolacion de Lagrange en 2D');
end
```

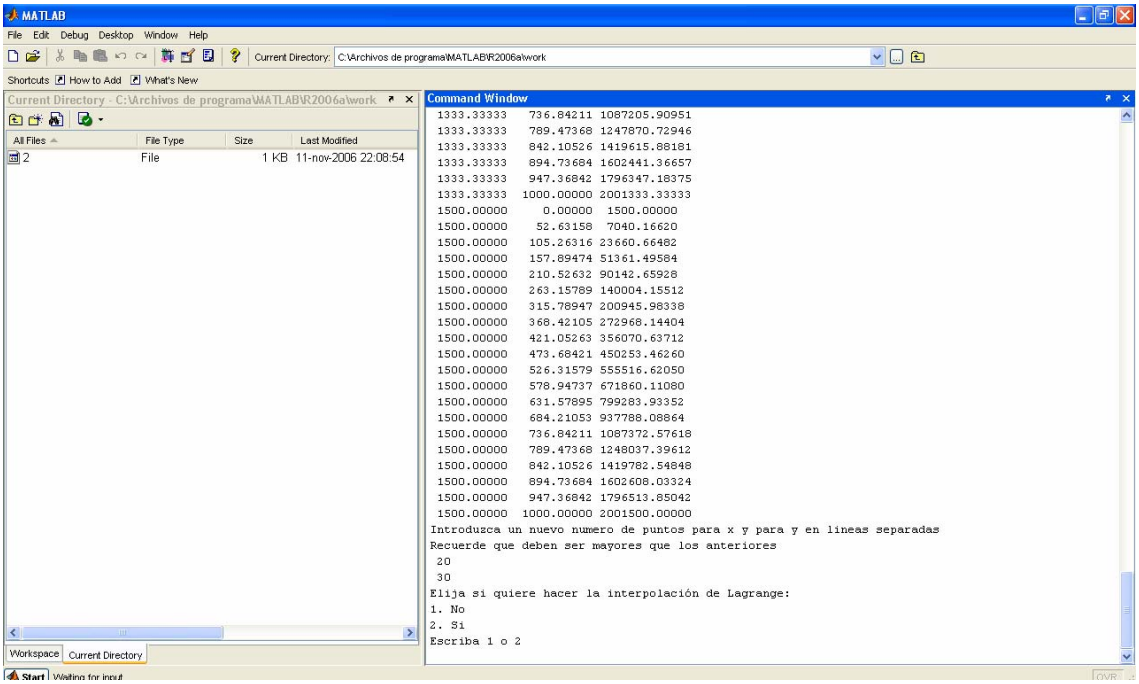
3. EJEMPLOS

Generación de datos mediante función F

- **Salida de datos por pantalla**

Para la función: $z = x + 2y^2$

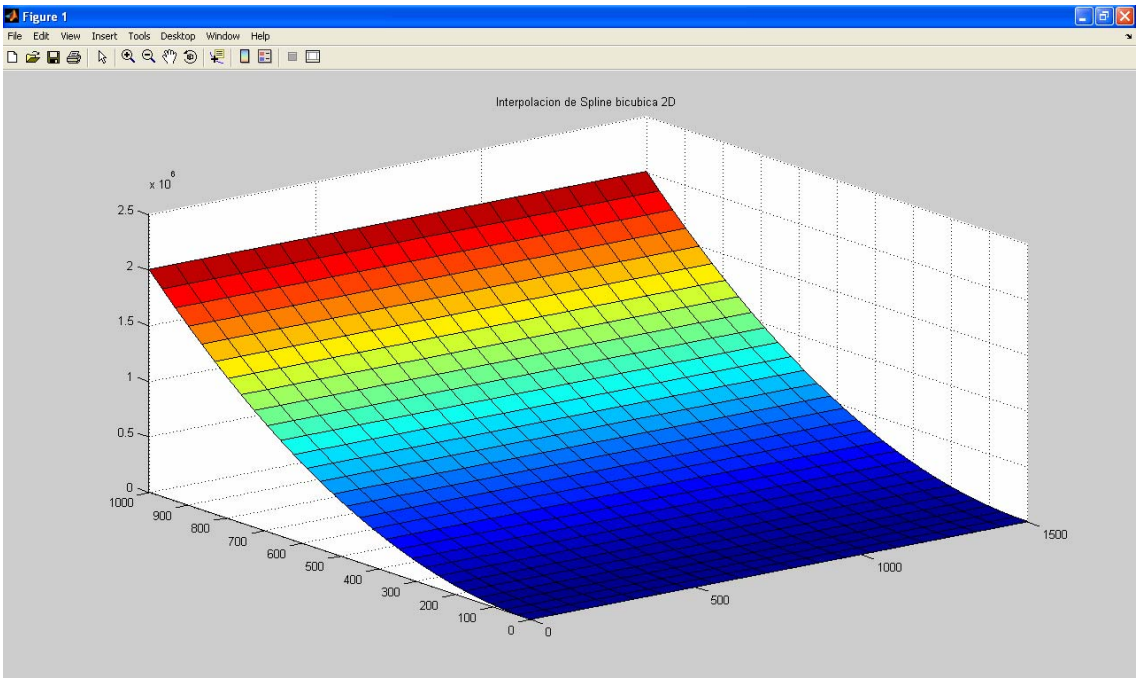


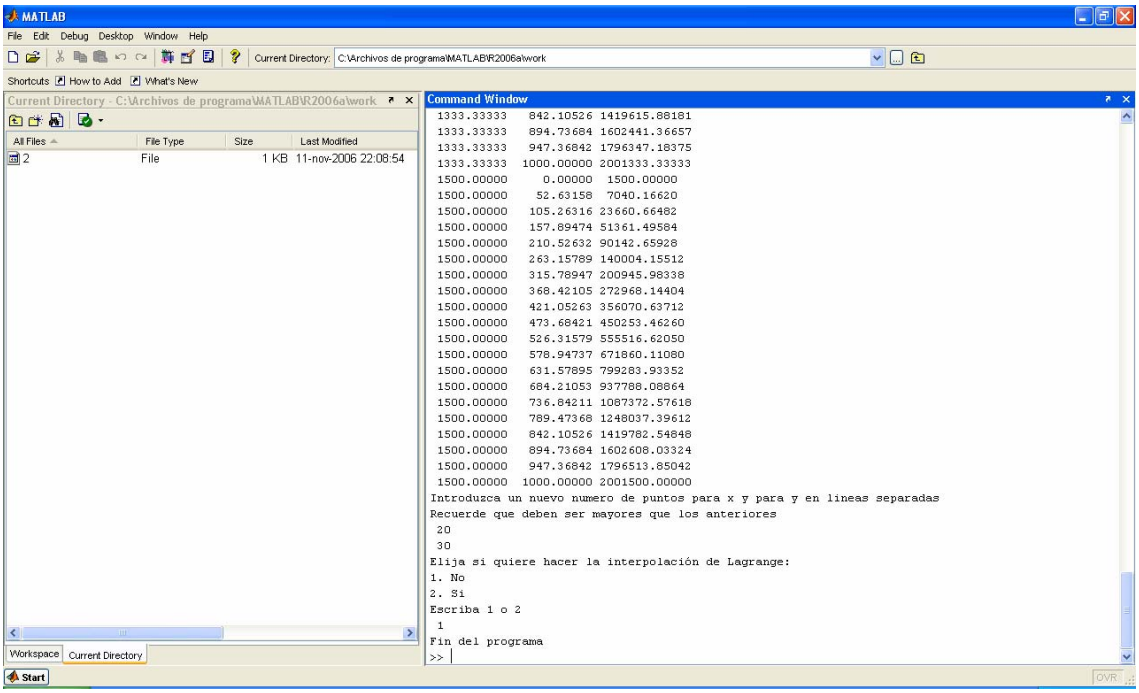


El usuario elige generar los datos mediante una función F, entonces, tecleamos la longitud de ***x*** y de ***y***, así como el número de puntos en ***x*** y en ***y***.

Por otra parte, elegimos la forma de salida por pantalla y se nos muestran todos los valores, y si nos fijamos, en la primera fila tenemos la longitud de ***x***, ***y*** así como los puntos en ***x*** e ***y***.

El programa nos pide el nuevo número de puntos para construir una malla más fina, de esa manera insertamos los valores y nos sale la figura de abajo. En este caso, hemos elegido que el programa no compare con Lagrange 2D.

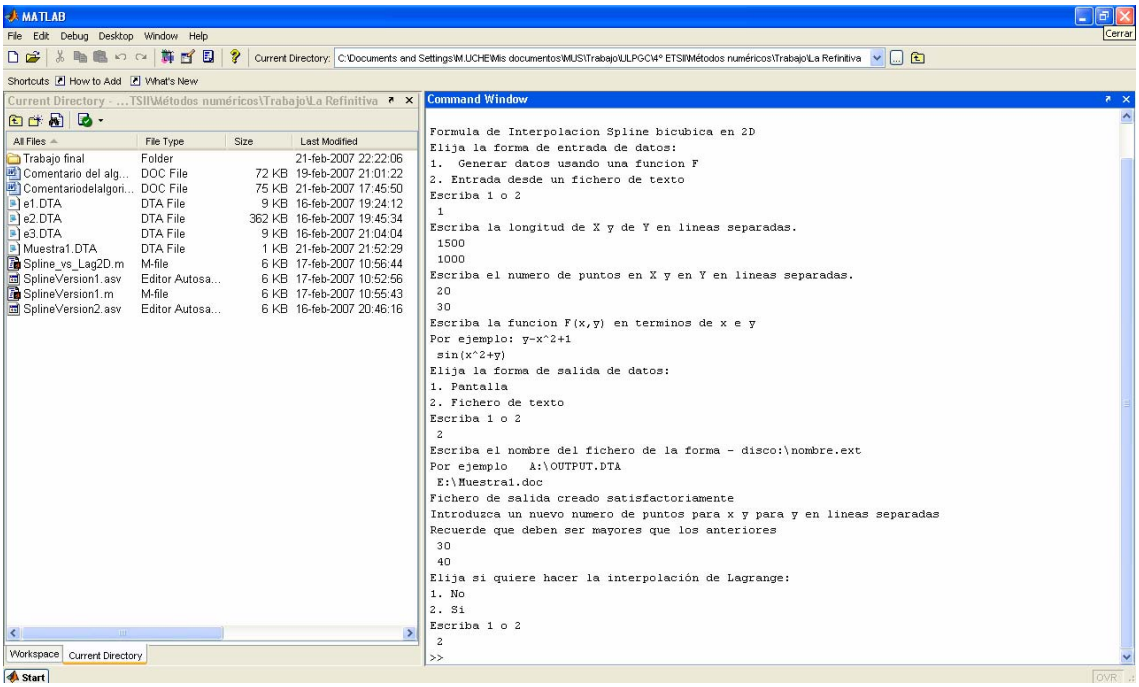




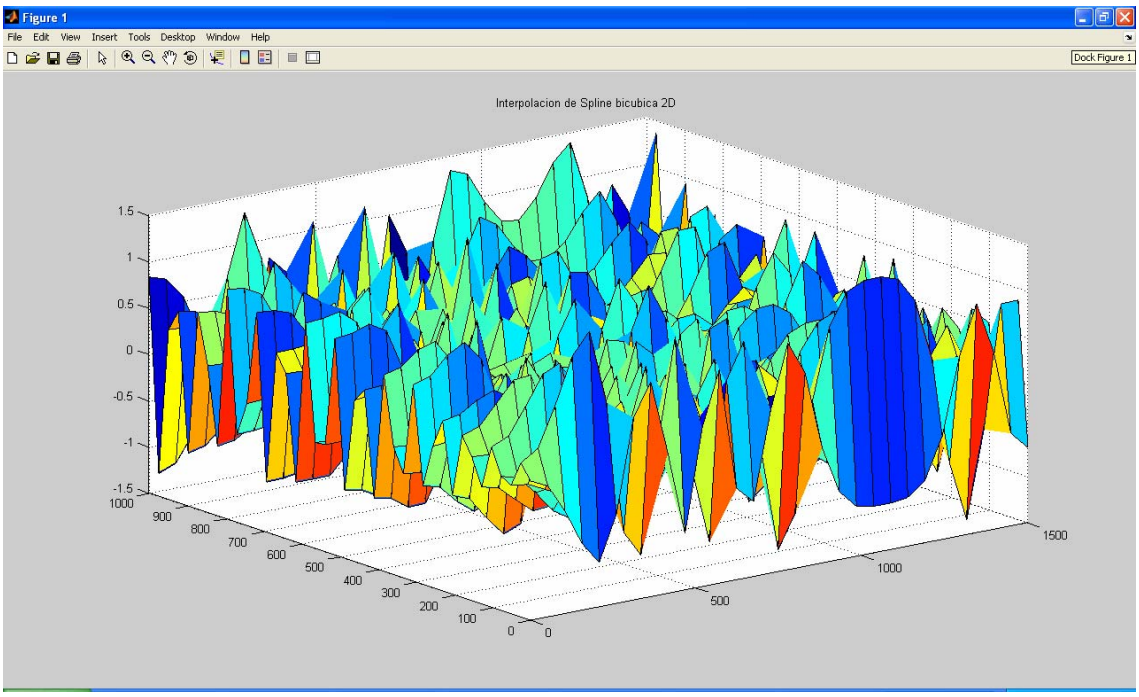
- **Salida de datos por fichero de texto**

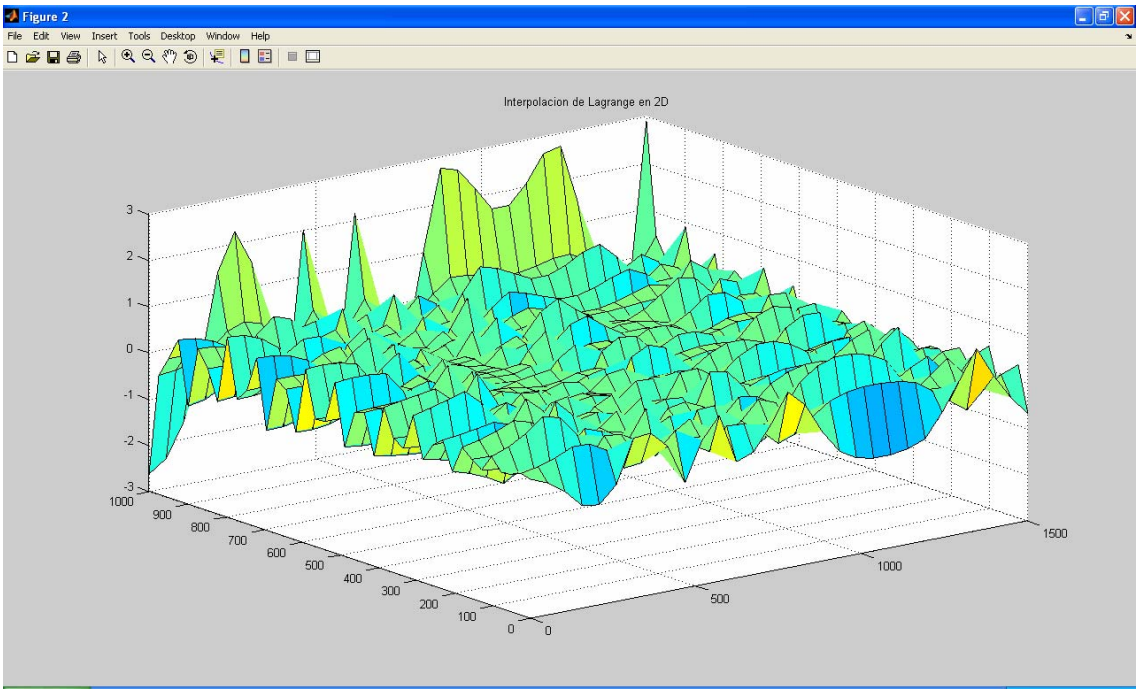
Para la función: $z = sen\left(x^2 + y\right)$

En este caso lo que hacemos es elegir la salida de datos mediante fichero.



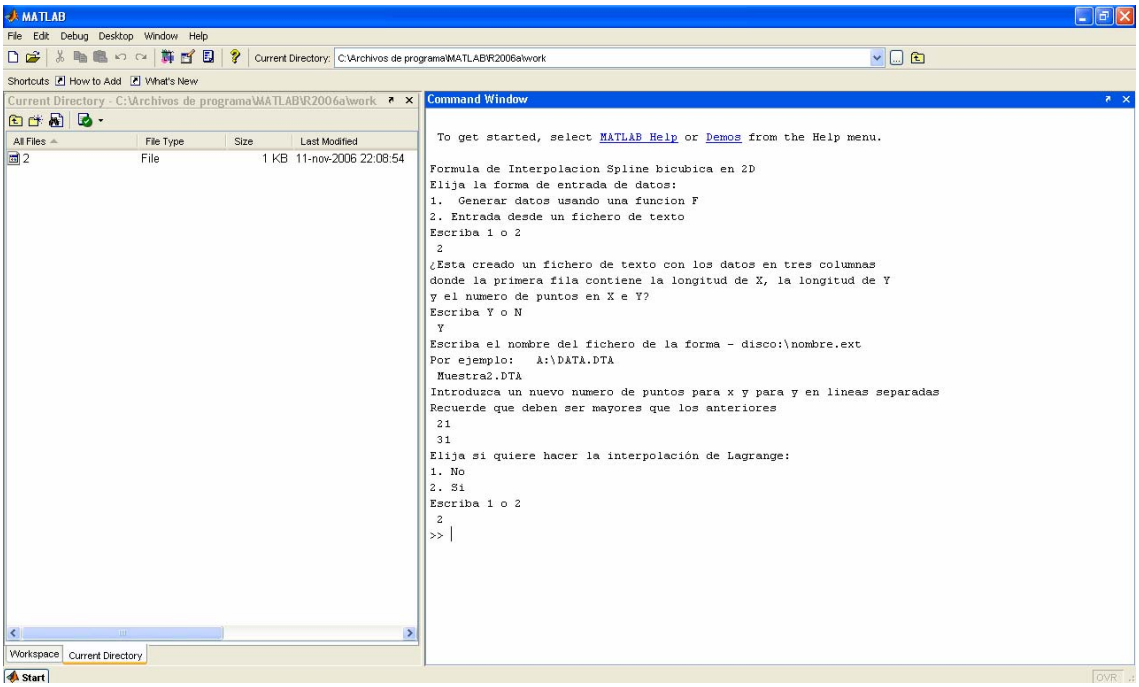
El resultado de la interpolación es la que se muestra. Además en esta ocasión elegimos compararla con la interpolación de Lagrange en 2D.



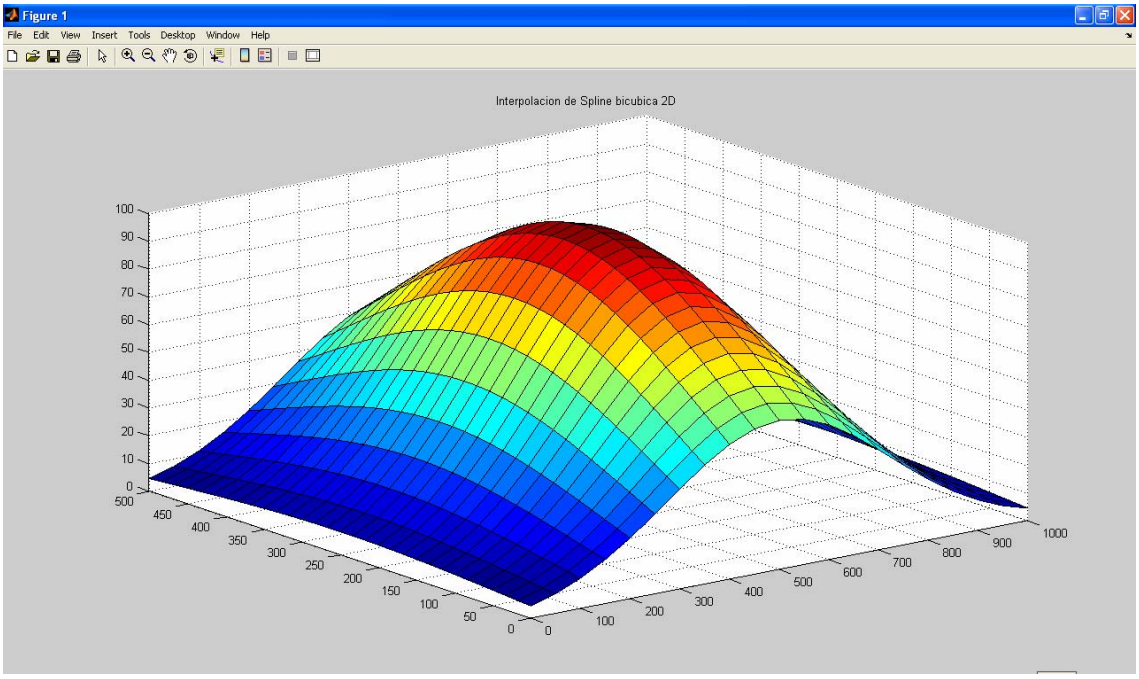


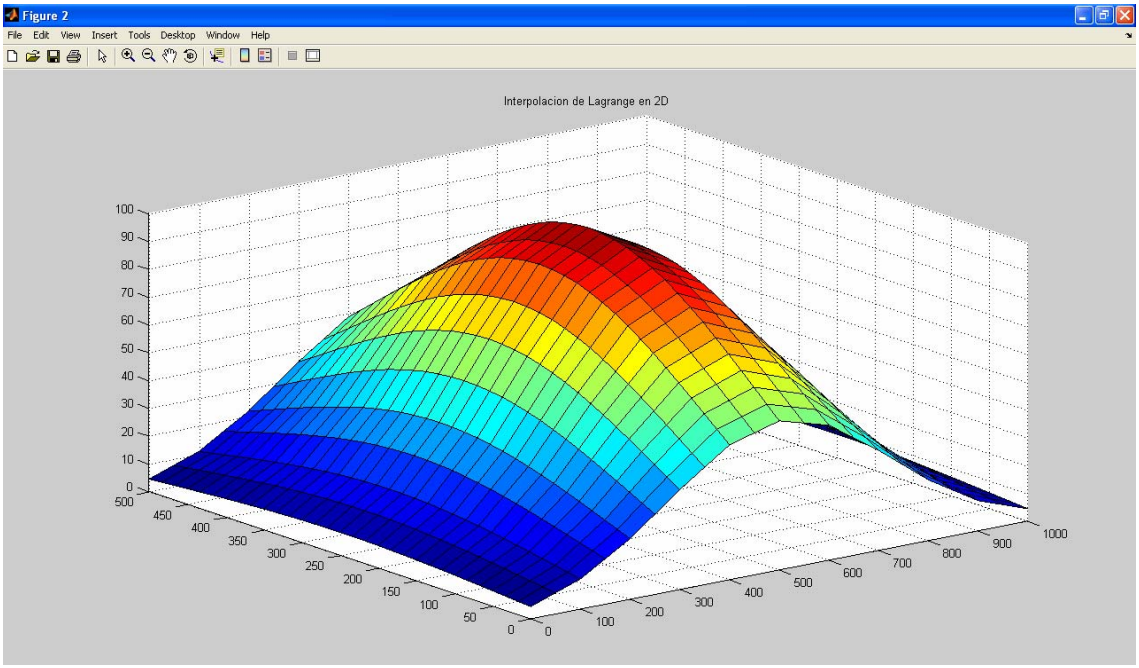
Entrada de datos mediante un fichero de texto

En esta ocasión, el usuario teclea la opción de entrada de datos por fichero, entonces, se supone que hay un fichero creado donde en la primera fila se especifica lo que pide (longitud de ***x***, de ***y*** y el número de puntos en ***x*** e ***y***):



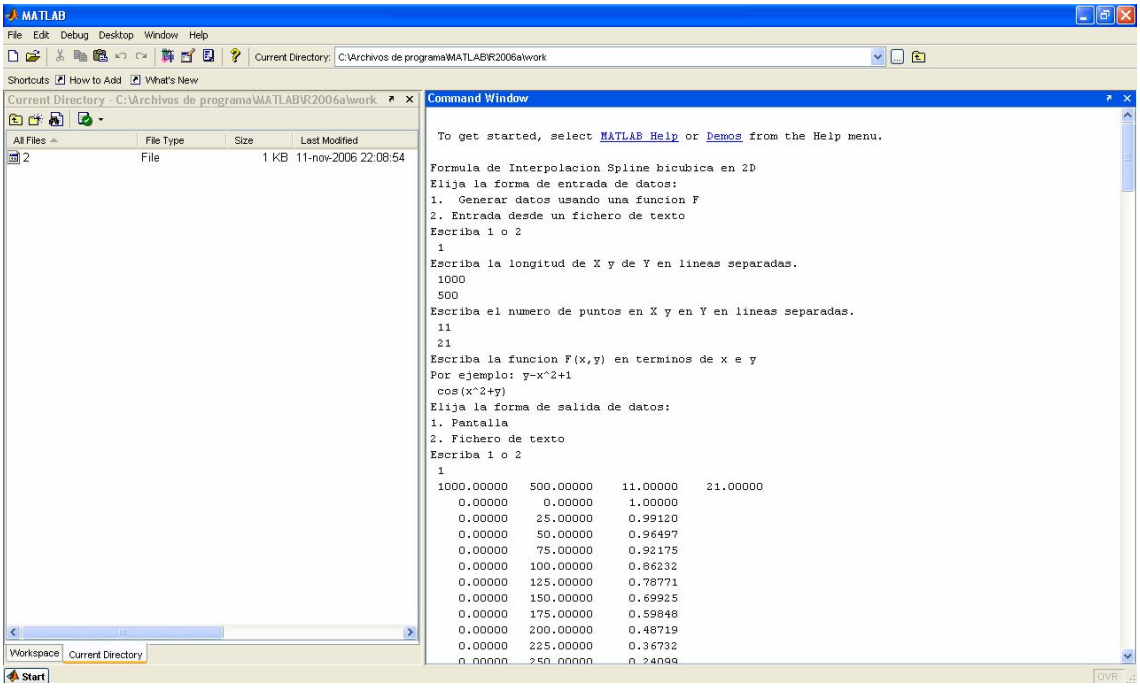
Donde se muestra la comparación de las dos interpolaciones, Spline bicúbica 2D y Lagrange 2D.



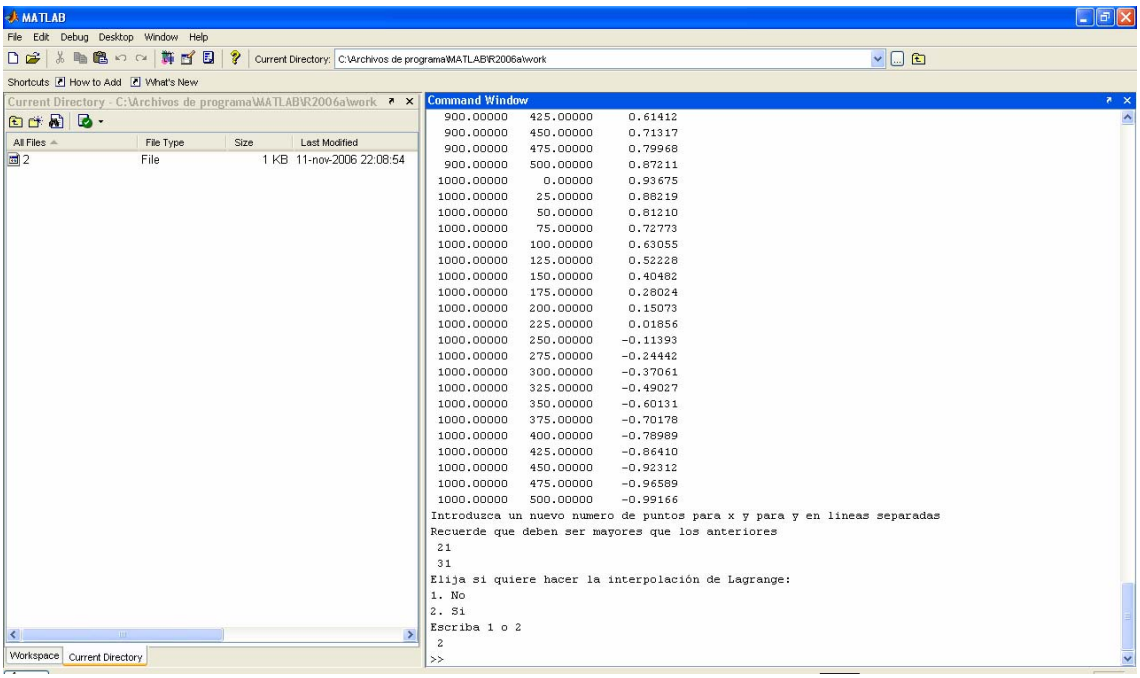


Otros ejemplos

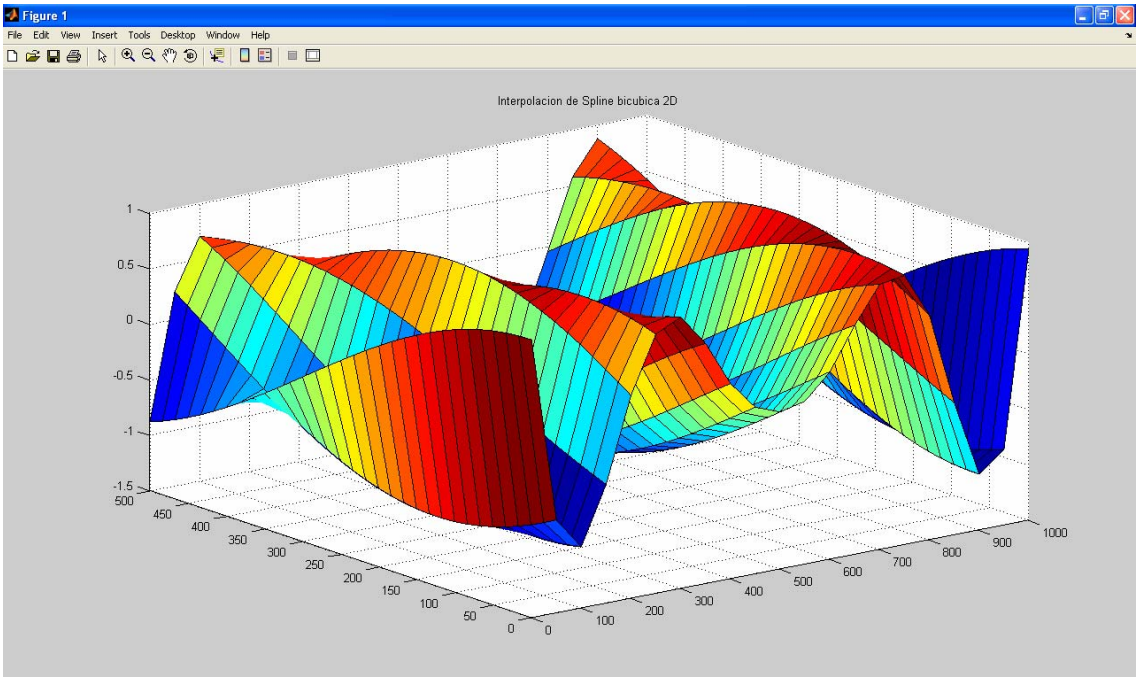
Para la función: $z = \cos\left(x^2 + y\right)$

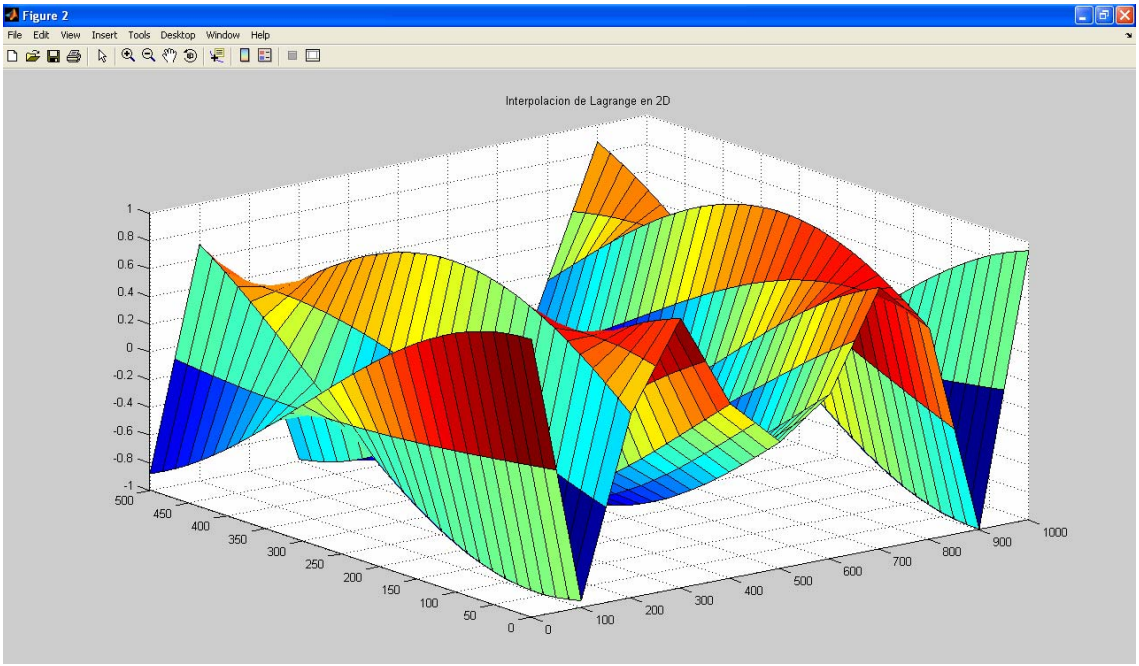


En este caso el usuarios elige generar datos por función e imprimir por pantalla, además desea la comparación con la interpolación de Lagrange en 2D.



Véanse las dos gráficas correspondientes a la función, la primera con interpolación Spline bicúbica2D y la segunda con interpolación de Lagrange 2D.





4. CONCLUSIONES

De forma muy breve, tras la comparación de gráficas se puede observar que la interpolación por Spline es más suave que en el caso Lagrange 2D. Esto, naturalmente, se debe a que la primera tiene carácter bicúbico y, sin embargo, la segunda tiene un carácter bilineal.

5. BIBLIOGRAFÍA

Libros:

- ✓ *Curvas y superficies para modelado geométrico. Juan M. Cordero Valle. Ed. RA-MA.*
- ✓ *Spline functions Basic theory. Schumaker, Larry L. Ed. Malabar, Fl: Krieger, 1993.*

Online:

- ✓ http://debin.etsin.upm.es/~leonardo/tema5.pdf
- ✓ http://www.uv.es/%7Ediaz/mn/node40.html
- ✓ http://www.pdipas.us.es/e/esplebrue/ampcap5a_0506.pdf
- ✓ http://en.wikipedia.org/wiki/B-spline
- ✓ http://wgpi.tsc.uvigo.es/~xulio/Web-TMM/Clase8Splines/c5%20sup%20sp2.pdf
- ✓ http://debin.etsin.upm.es/~leonardo/tema5.pdf