

Tareas de las unidades 1, 2 y 3

Ejercicio 1

Código:

```
n = input('Numero de terminos?: ');
a = [];
for k=0:n
    if k == 0
        anterior = 0;
    else
        anterior = a(k);
    end
    a(k+1) = anterior + ((-1).^k)/(2.*k + 1);
end
disp('Terminos:');
disp(a);

tolerancia = abs(input('Tolerancia?: '));
k = 0;
err = tolerancia+1;
a = 0;
while tolerancia < err
    a = a + ((-1).^k)/(2.*k + 1);
    err = abs(a - pi/4);
    k = k + 1;
end
disp(['Numero de terminos: ', num2str(k), ', Error: ', num2str(err), ', Valor: ', num2str(a)]);
```

En la primera parte pedimos el número de términos con el comando input y calculamos con un bucle for los términos en la posición k+1 del vector a. Utilizamos el valor anterior k para el cálculo del valor actual k+1 salvo en el primer caso, que es 0.

En la segunda parte pedimos la tolerancia con el comando input (nos aseguramos que no es negativo con abs) y utilizamos un bucle while para calcular los términos mientras el error sea mayor a la tolerancia pedida (por eso inicializamos el error como la tolerancia + 1).

Podemos ver dos ejemplos de este script a continuación:

```
>> ejercicio1
Numero de terminos?: 5
Terminos:
    1.0000    0.6667    0.8667    0.7238    0.8349    0.7440

Tolerancia?: 0.1
Numero de terminos: 3, Error: 0.081269, Valor: 0.86667

>> ejercicio1
Numero de terminos?: 10
Terminos:
    1.0000    0.6667    0.8667    0.7238    0.8349    0.7440    0.8209    0.7543    0.8131    0.7605    0.8081

Tolerancia?: 0.001
Numero de terminos: 250, Error: 0.001, Valor: 0.7844
```

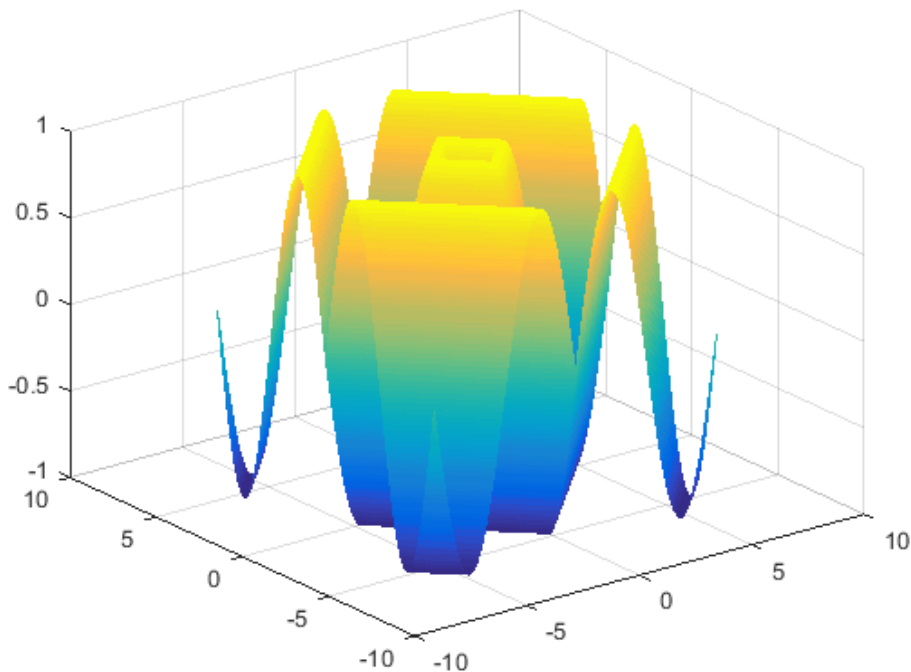
Ejercicio 2

Código:

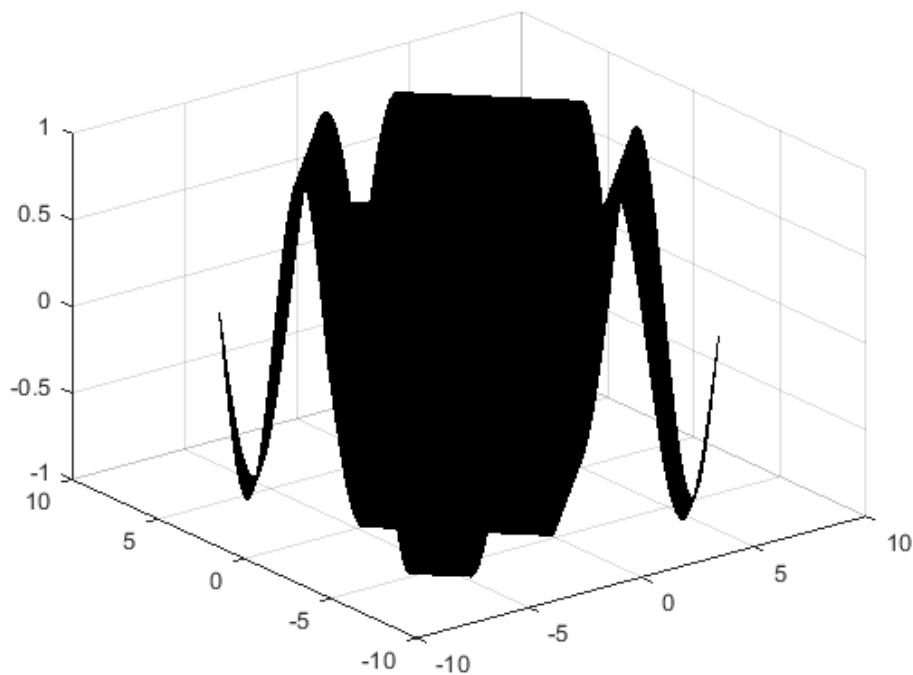
```
x = linspace(-2*pi,2*pi,400);  
y = x;  
[X,Y] = meshgrid(x,y);  
F = sin(abs(X)+abs(Y));  
mesh(X,Y,F);  
figure  
surf(X,Y,F);  
figure  
contour(X,Y,F, 'ShowText', 'on');
```

Utilizamos linspace para crear x e y con 400 términos equidistantes entre -2π y 2π . Con meshgrid creamos la malla de puntos para evaluarlo y guardar el resultado en F. A continuación utilizamos los comandos mesh, surf y contour para visualizar el resultado (intercalando el comando figure para crear nuevas figuras cada vez).

Mesh:

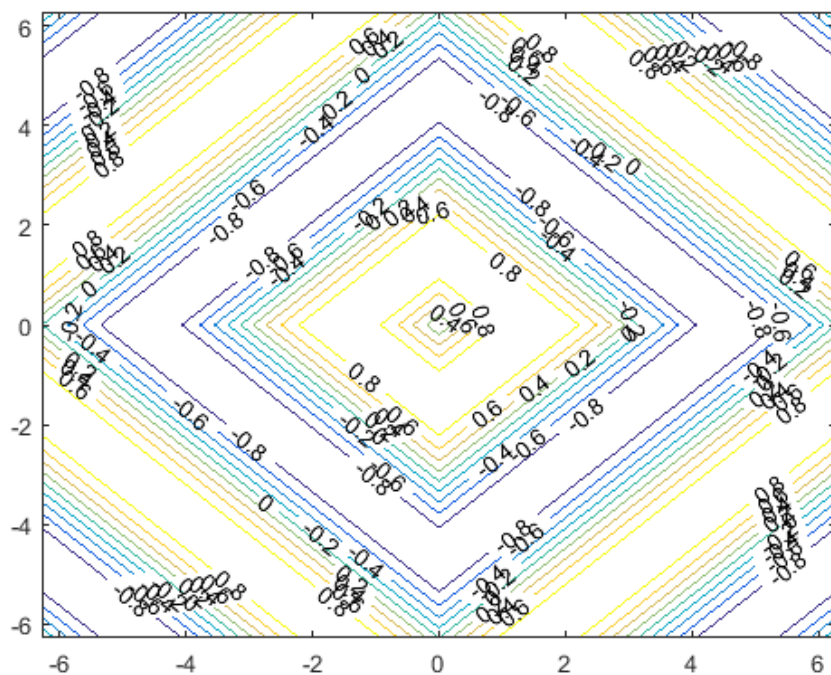


Surf:



La figura aparece de color negro porque el comando surf pinta los puntos de la malla de color negro, en este caso, el tamaño de la figura es demasiado pequeño y la malla tiene muchos puntos, por lo que no se aprecian las zonas coloreadas.

Contour:



Ejercicio 3

Código:

```
A=[-8, 5, 7, -3;  
    9, 2, 9, -9;  
    5, 6, -3, -7;  
    -6, 3, 3, 5];
```

```
B = [-90, -7, 15, -30]';
```

```
X = A\B;
```

```
[L,U,P]=lu(A);
```

```
X2 = U\ (L\ (P*B));
```

Para resolver el sistema guardamos en A los términos de la matriz de coeficientes y en B el vector de términos independientes (transpuesto). En la variable x guardamos el resultado de esta ecuación mediante el método de división y en x2 el resultado mediante la factorización LU.

Podemos ver el resultado de ejecutar este script a continuación:

```
>> ejercicio3
```

```
>> X
```

```
X =
```

```
    6.7348  
   -0.0030  
   -3.3927  
    4.1192
```

```
>> L
```

```
L =
```

```
    1.0000    0    0    0  
   -0.8889    1.0000    0    0  
    0.5556    0.7213    1.0000    0  
   -0.6667    0.6393    0.0314    1.0000
```

```
>> U
```

```
U =
```

```
    9.0000    2.0000    9.0000   -9.0000  
    0    6.7778   15.0000  -11.0000  
    0    0  -18.8197    5.9344  
    0    0    0    5.8467
```

```
>> P
```

```
P =
```

```
    0    1    0    0  
    1    0    0    0  
    0    0    1    0  
    0    0    0    1
```

```
>> X2
```

```
X2 =
```

```
    6.7348  
   -0.0030  
   -3.3927  
    4.1192
```

Podemos comprobar que era necesario multiplicar P*B para hacer la permuta pues la matriz de permutación es distinta a la matriz identidad.