Table of Contents

EXAMPLES 1
0. Practiques 1
1.1. Bisection Method 1
1.2. Newton's Method 2
2. Polinomial interpolation 3
2.1. Interpolar funcions entre -1 i 1 (equispaced) 3
2.2. Interpolar funcions entre a i b (equispaced) 4
2.3. Interpolar funcions entre a i b (Chebyshev) 5
3. Numerical differentiation 6
3.1. Diffmat() 6
3.2 Chebdiff() 7
4. Numerical integration 8
4.1 Composite Trapezoidal Quadrature 8
4.2. Composite Simpson's Quadrature 9
4.3 Clenshaw - Curtis Quadrature 9
4.4 Qcot() Integrals impropies en [0,oo) 9
4.5 Qtanh() Integrals amb singularitats 10
Plantilles plots 10

EXAMPLES

```
% Exemples basics de diferents funcions de matlab
clear all;
format long;
```

0. Practiques

```
% Practica 4: Metodo de Newton, representacion de errores, orden de
% convergencia
% Practica 5: Polinomios cardinales (-1,1), funcion de Lebesgue, error
 en
% la interpolacion
% Practica 6: Interpolacion de chebychev (0,1)
% Practica 7: toeplitz, diferencias finitas
% Practica 8: Diferencias finitas, derivacion glogal (equispaced i
 chevy)
% Practica 9: polyfit
% Practica 10: (potencial en el eje), qclencurt, curvas
 equipotenciales
% Practica 11: qtanh
```

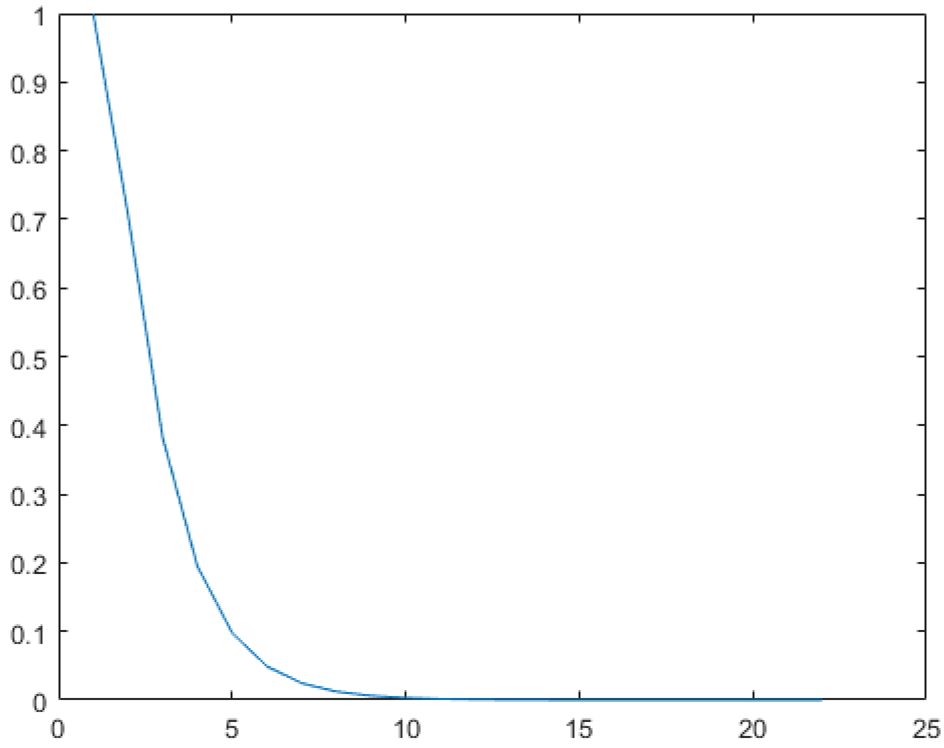
1.1. Bisection Method

```
% Funcio bisection:
% Input:
% 1. [a,b]: interval (it assumes that f(a)f(b) < 0)
% 2. tol: tolerance so that abs(x_k+1 - x_k) < tol
```

```
% 3. itmax: maximum number of iterations allowed
% 4. y: function’s name
% Output:
% 1. xk: resulting sequence
% 2. res: resulting residuals
% 3. it: number of required iterations
tol = 10^(-6);
itmax = 50;
y = @(x) sin(x);

[xk,res,it] = bisection(0,pi,tol,itmax,y);

xk(end)              % arrel
plot([1:it],res)     % residu en cada iteracio

%Exemple: Practica 4
```

ans =

3.141591904575737



1.2. Newton's Method

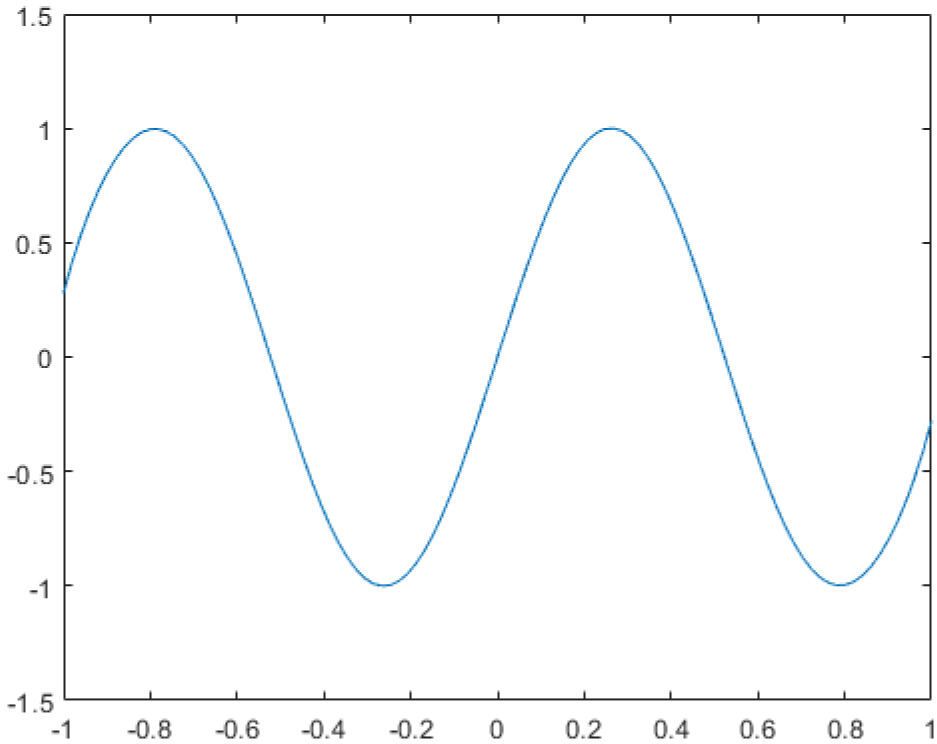
```
% Funcio minewton:
```

```
% Atencio: minewton utilitza la funcio derivative
% Inputs:
    % x1: incial guess
    % tol: tolerancia desitjada
    % itmax: nombre maxim d'iteracions permeses
    % y: funcio de la que volem trobar l'arrel
% Outputs:
    % xk: valors de x que anem trobant fins arribar a l'arrel
    % fk: residu de la funcio y en cada un dels xk
    % it: nombre d'iteracions necesaries

y = @(x) x.^2 -1;
tol = 10^(-6);
itmax = 50;
x1 = 0.5;

[xk,fk,it] = minewton(x1,tol,itmax,y);
```

2. Polinomial interpolation

2.1. Interpolar funcions entre -1 i 1 (equi-spaced)

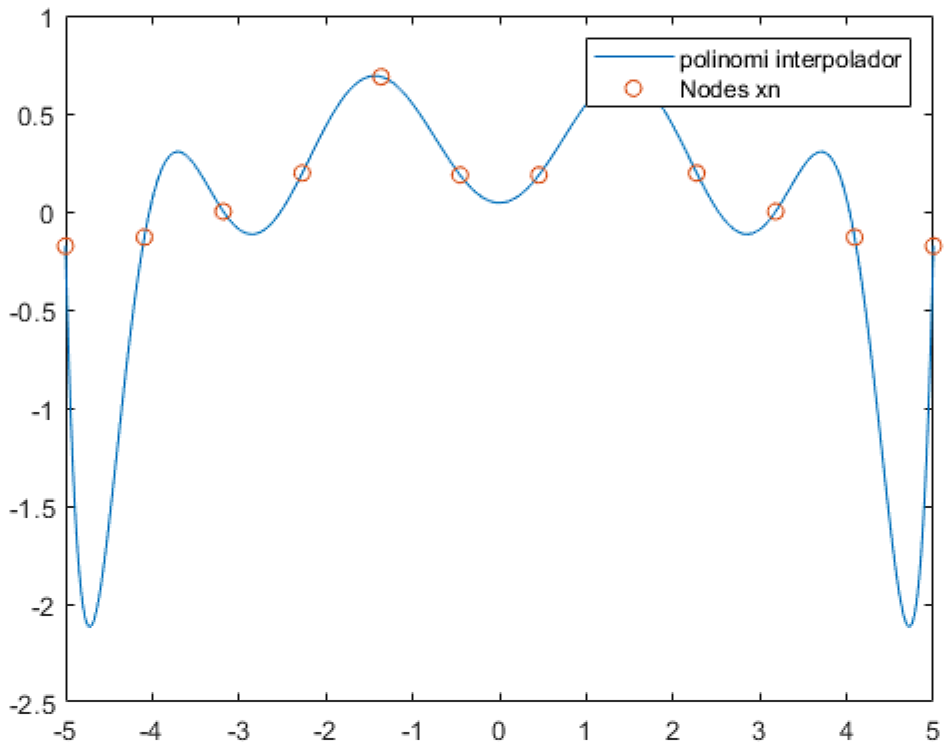
```
% Funcio interpol_f
% Atencio: utilitza la funcio interpol
% Input
    % m: nombre de punts de la malla (z)
    % n: nombre de punts equiespaiats (x)
    % f: funcio a interpolar
% Output
    % P: matriu (m+1) x (n+1) dels polinomis cardinals de Lagrange
    % evaluats en la malla z. P(i,j) = lambda_i(z_j)
    % pi: valor del polinomi interpolador en cada punt de la malla
    % e_n: error maxim comes en cada una de les interpolacions pi
    % z: malla
m = 101;
n = 10;
y = @(x) sin(6*x);
[P,pi,e_n,z] = interpol_f(m,n,y);

plot(z, pi)       %representacio del polinomi interpolador
```



2.2. Interpolar funcions entre a i b (equispaced)

```
% Funcio cardpolequi (no tinc funcio implementada que em doni
 directament
% el polinomi interpolador aixi que calen uns passos extres indicats)
% Input:    1. a & b: interval [a,b]
%           2. n: with x_0 = a and x_n = b (n+1 interp nodes)
%           3. m: number of points in dense grid
%
% Output:   1. P: matrix of card. polynomials.
%           2. xn: equispaced interpolation nodes
%           3. z: dense grid
f = @(x) (sin(x).^3)./x;
a = -5;
b = 5;
n = 11;         % al tanto, cert nombre de nodes fan que es demani
                % evaluar la funcio a xn = 0, i genera error (p.e n=
 10)
m = 500;
[P, xn, z] = cardpolequi(a, b, n, m);

Fn = f(xn);             % evaluem f en els nodes
Fz = P*Fn;              % matriu de polinomis * valor de f en els
 nodes
```

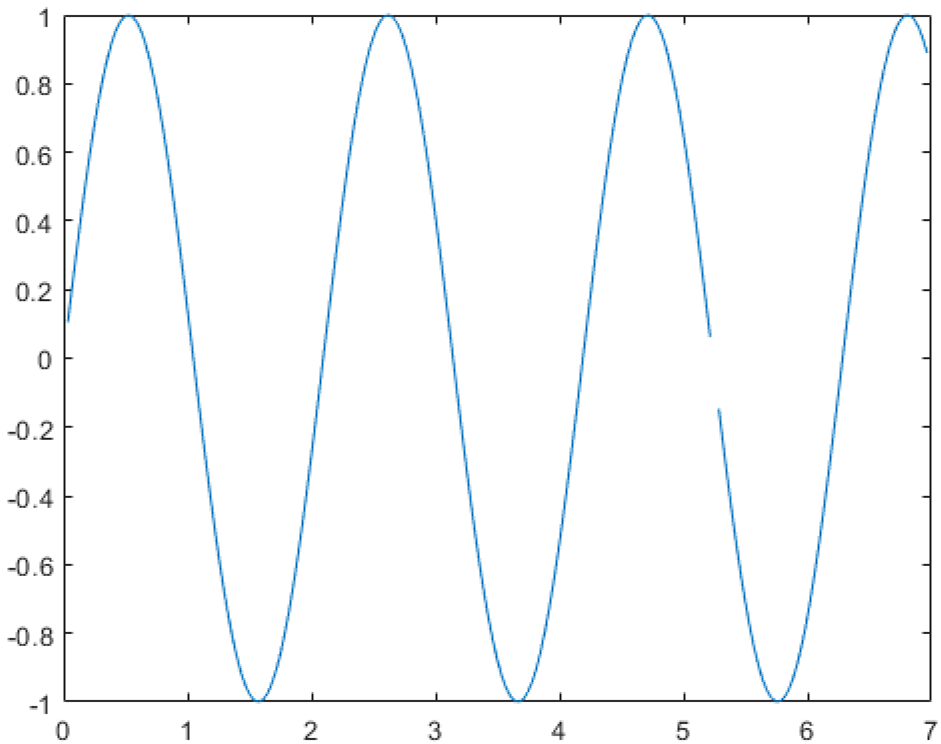
```
                       % Fz sera la derivada de F evaluada en la
 malla z

figure(1)
plot(z, Fz);
hold on;
plot(xn, f(xn), 'o');
legend ('polinomi interpolador', 'Nodes xn')
hold off
```



2.3. Interpolar funcions entre a i b (Chebyshev)

```
% Funcio barifun
% Input
    % a, b: limits inferior i superior del interval
    % n = nombre de punts equiespaiats (x)
    % m = nombre de punts de la malla (z)
    % f = funcio a interpolar
% Output
    % ff: interpolant evaluat a la malla z
    % xn: nodes de chevychev
    % z: malla
f =  @(x) sin(3*x);
a = 0;
b = 7;
n = 30;
```

```
m = 201;

[ff,xn,z] = barifun(a,b,n,m,f);
plot(z,ff)     %polinomi interpolador

%La practica 6 es un exemple complet i ens mostra com trobar l'error
```

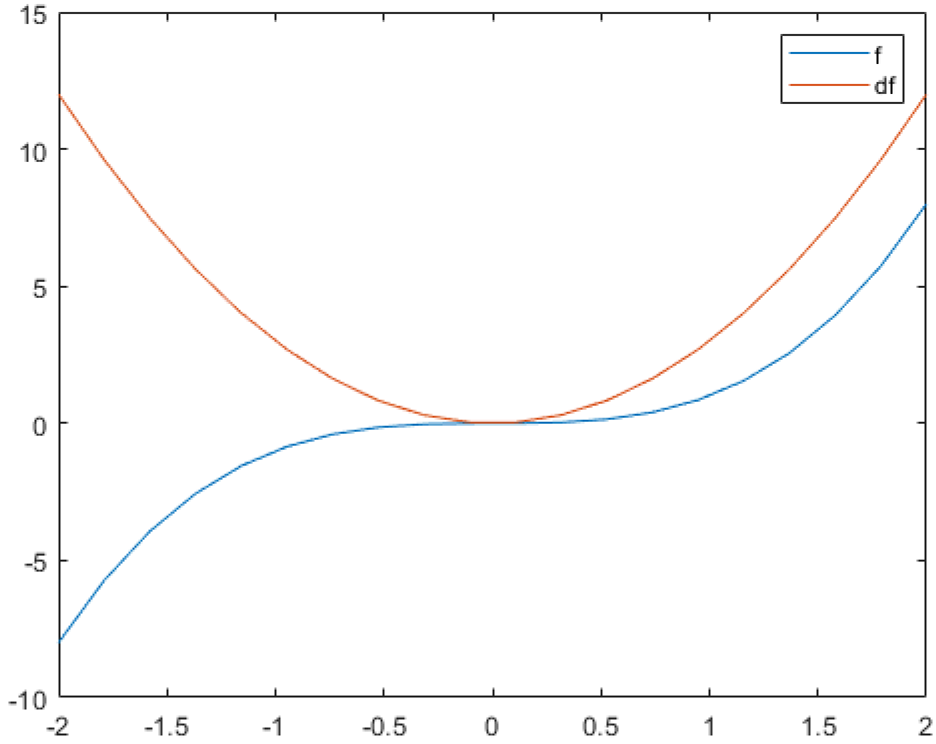


3. Numerical differentiation

3.1. Diffmat()

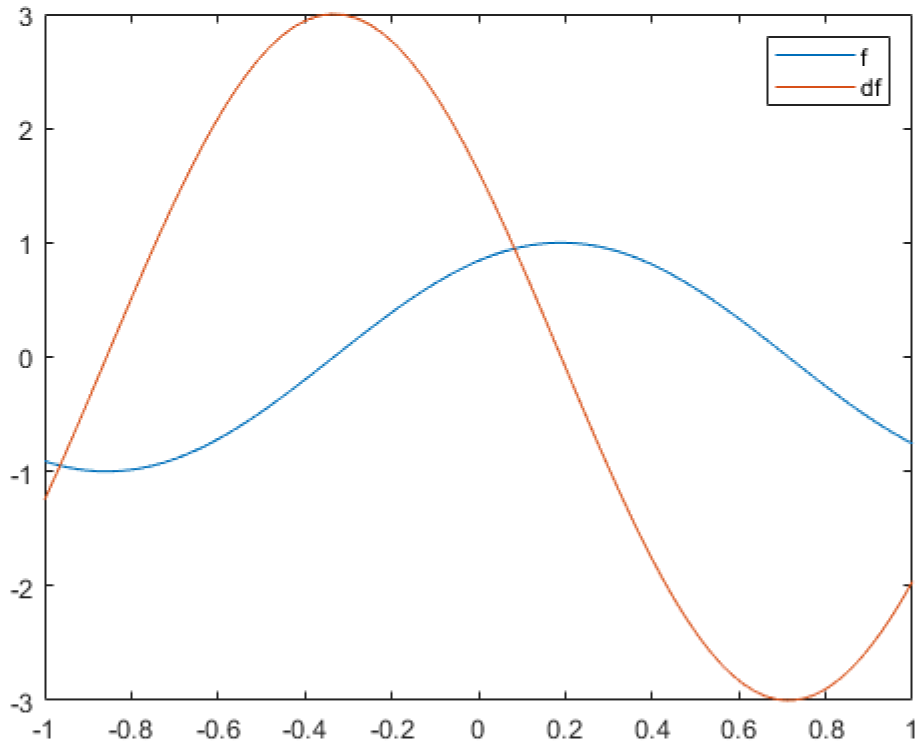
```
% Funcio diffmat
% Input: vector of nodes x= [x_0; x_1; . . .; x_n]
% Output: differentiation matrix D
f =  @(x) x.^3;
a = -2; b = 2;
n = 20;
h = 1/n;
x = linspace(a,b,n);
fx = f(x);
D = diffmat(x);
df = (D*fx');

figure(1)
plot(x,fx)
```

```
hold on
plot(x,df)
hold off
legend('f','df')
```



3.2 Chebdiff()

```
% Funcio chebdiff
% Code 5B: Chebyshev Differentiation matrix
% Input: n
% Output: differentiation matrix D and Chebyshev nodes
n = 100;
f = @(x) sin(3*x + 1);
[D,xj] = chebdiff(n);
fj = f(xj);          % Evaluación de f en los nodos
df = D*fj;           % Aproximación de la función derivada

figure(2)
plot(xj,fj)
hold on
plot(xj,df)
hold off
legend('f','df')
```



4. Numerical integration

4.1 Composite Trapezoidal Quadrature

```
% Funcio ctrap
% Code 6: Composite Trapezoidal Quadrature
% Input:
    % a-b (low-up lim.);
    % m (# intervals);
    % f (func. name)
% Output: I_{1,m}(f)

f = @(x) x.^2;
a = 1; b = 2;
m = 100;              %number of interval divisions
T = ctrap(a,b,m,f)


T =

   2.333350000000000
```

4.2. Composite Simpson's Quadrature

```
% Funcio csimp
% Code 7: Composite Simpson Quadrature
% Input:
    % a-b (low-up lim.);
    % m (# intervals);
    % fun (func. name)
% Output: I_{1,m}(f)

f = @(x) x.^2;
a = 1; b = 2;
m = 100;              %number of interval divisions
S = csimp(a,b,m,f)


S =

   2.333333333333333
```

4.3 Clenshaw - Curtis Quadrature

```
% Funcio qclencurt
% Input: a-b (low-up lim.)
%        n (# nodes-1)
%        fun (func. name)
% Output: I_n(f)
f = @(x) x.^2;
a = 1; b = 2;
m = 100;              %number of interval divisions
Icc = qclencurt(a,b,m,f)    %valor numeric de l'integral


Icc =

   2.333333333333333
```

4.4 Qcot() Integrals impropies en [0,oo)

```
% Funcio qcot
% Code 9: Cot-Map for 1st kind improper integrals [0,+inf)
% Input:
    % n: #abscissas;
    % L: scaling factor
% Output: I_n(f)
t = @(z) 1./(z.^3);
L = 10;
n = 50;
In = qcot(n,L,t)
```

```
In =

     2.693592296296307e+04
```

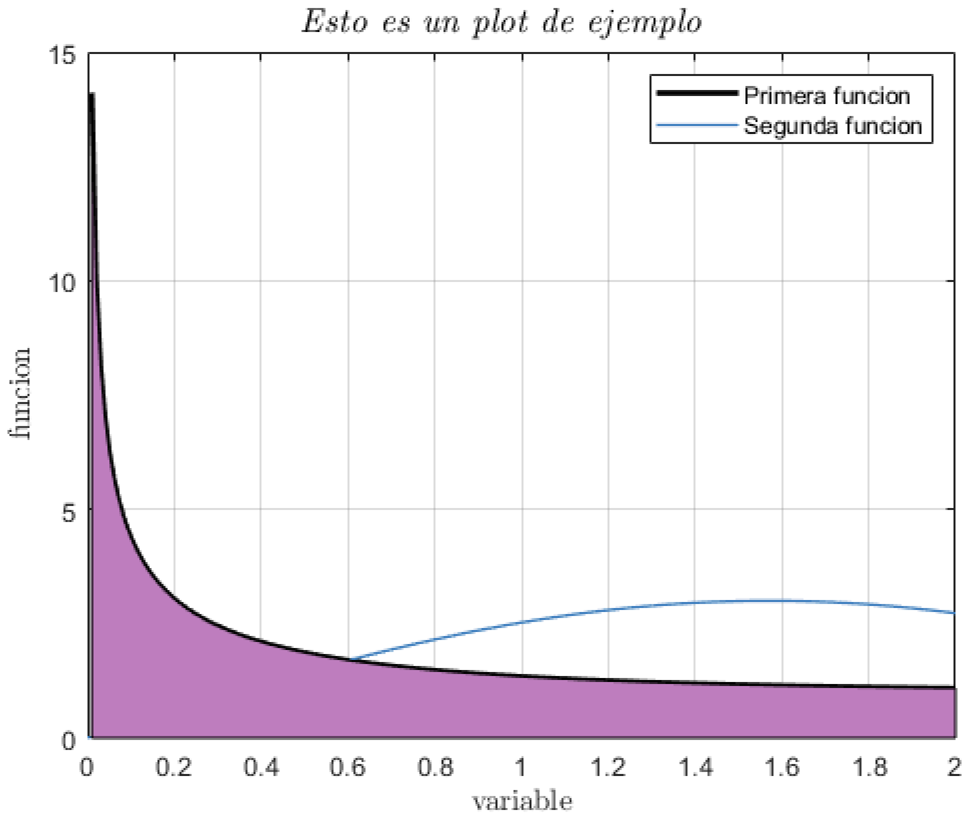
4.5 Qtanh() Integrals amb singularitats

```
% Funcio qtan
% Code 10: tanh-rule for 2nd kind improper integrals (-1, +1)
% Input:     n (# abscissas)
%            a-b (integration domain)
%            c (tanh scaling factor) usually 1-10
% Output:    I_n(f)
t = @(z) 1./(z.^3);
a = 0; b = 2;                       % limites de integracion
c = 5;                              % scaling factor
n = 50;                             % numero de nodos
I_tan = qtanh(n, a, b, c, t)
```

```
I_tan =

     1.187874522247161e+30
```

Plantilles plots

```
tz = @(z) sqrt((1+exp(-2.*z))./(1-exp(-z)));
sinus = @(z) 3*sin(z);
figure(3)
plot([0:0.01:2],tz(0:0.01:2),'LineWidth',2, 'Color', 'k') % r,b,k
hold on
plot([0:0.01:2],sinus(0:0.01:2),'LineWidth',1, 'Color', [0.25 0.5
 0.75])
area([0:0.01:2], tz(0:0.01:2),'FaceColor',[0.75 0.5 0.75]);
sgtitle('\textit{Esto es un plot de ejemplo}','Interpreter','Latex');
xlim([0 2]);
xlabel('variable','Interpreter','Latex')
ylabel('funcion','Interpreter','Latex')
legend('Primera funcion', 'Segunda funcion');
grid on
```



Published with MATLAB® R2020b