Códigos proyecto final en matlab

```
%Proyecto Final
%7-12-18

%biseccion
clear, clc
funcion=char(inputdlg('ingrese funcion')); %char para ingresar funcion
"inputdlg transformarla
f=inline(funcion); %iline crea funcion
x=-5:0.1:5;
n=length(x);
for i=1:n
    y(i)=f(x(i));
end
plot(x,y);
hold on;
plot([-5 5],[0 0],'r');
grid on;
plot ([0 0],[-5 5],'r');
j=-5;
for i=1:11
    text(j,0,num2str(j)); %str2double transforms un string en un numero
    j=j+1;
end
a=str2double(inputdlg('ingrese el valor de a'));
b=str2double(inputdlg('ingrese el valor de b'));


iteraciones=1;
j=1;
pmviejo=0;
while iteraciones<5
    pm=((a+b)/2);
    fa=f(a);
    fb=f(b);
    fpm=f(pm);
    aa(j)=a;
    bb(j)=b;
    ppm(j)=pm;
    errorf=abs(pm-pmviejo);
    err(j)=errorf;
    if fa*fpm<0
        b=pm;
    end
    if fb*fpm<0
        a=pm;
    end
    pmviejo=pm;
    j=j+1;
    iteraciones=iteraciones+1;
end
g=figure;
t=uitable(g,'data',[aa' ppm' bb' err']);
```

```
%metodo falsa posicion
clear, clc
e=input('ingrese la funcion a evaluar');
f=inline(e)
x=-5:0.1:5;
y=f(x);
plot(x,y);
grid on;

xa=input('ingrese el intervalo inferior:');
xb=input('ingrese el intervalo superior:');
T=input('ingrese la tolerancia:');
vc=0; ite=0.0000; error=100;
fprintf('\t iter \t\txa \t\t vc \t\t xb \terror\n')


while (error > T)
    vc=(xa*f(xb)-xb*f(xa))/(f(xb)-f(xa));
    disp([ite,xa,vc,xb,error]);
    if (f(xa)*f(vc)<0)
       xb=vc;
    else xa=vc;
    end
    error=abs(f(vc));
    ite=ite+1;
%err=-5:0.1:5;
%iteracion=ite;
%plot(err,iteracion);
%grid on;
end
fprintf('la raiz se encontro en el punto:\n\t%f\n',vc);
```

RIDDER

```
solve('x^3-13*x-12')
clc
x=3.5;
x^3-13*x-12
x=4.25
x^3-13*x-12
7.75/2
x=3.875
x^3-13*x-12
x=(3.875+4.25)/2
x^3-13*x-12
x=(3.875+4.06)/2
x^3-13*x-12
x=5-(48*3)/78
x^3-13*x-12
ans+48
x^3-13*x-12
48+ans
x^3-13*x-12
48-ans
5-x
x=5-(48*1.8462)/69.6295
x^3-13*x-12
48-ans
5-x
x=5-(48*1.2727)/56.6725
x^3-13*x-12
48-ans
5-x
x=5-(48*1.0779)/50.6555
x^3-13*x-12
48-ans
5-x
x=5-(48*1.0214)/48.7433
x^3-13*x-12
syms('xi', 'xm', 'xs')
[xi^2 xi 1; xm^2 xm 1; xs^2 xs 1]
inv([xi^2 xi 1; xm^2 xm 1; xs^2 xs 1])
xi=2
xs = 5
xm = (xi+xs)/2
xm = (xi+xs)/2
[xi*xm xi*xs xm*xs xi^2 xm^2 xs^2]
-30/4.5+14.625/2.25+48/4.5
(30*8.5)/4.5-14.625*7/2.25-48*5.5/4.5
-30*17.5/4.5-14.625*10/2.25-48*7/4.5
a2=-30/4.5+14.625/2.25+48/4.5
a1=(30*8.5)/4.5-14.625*7/2.25-48*5.5/4.5
a0=-30*17.5/4.5-14.625*10/2.25-48*7/4.5
(-a1+sqrt(a1^2-4*a0*a2))/(2*a2)
(-a1-sqrt(a1^2-4*a0*a2))/(2*a2)
a0
a0=-30*17.5/4.5+14.625*10/2.25-48*7/4.5
(-a1+sqrt(a1^2-4*a0*a2))/(2*a2)
a0=-30*17.5/4.5+14.625*10/2.25+48*7/4.5
```

```
(-a1-sqrt(a1^2-4*a0*a2))/(2*a2)
(-a1+sqrt(a1^2-4*a0*a2))/(2*a2)
x=(-a1+sqrt(a1^2-4*a0*a2))/(2*a2)
x^3-13*x-12
```

Newton Raphson

```
clear
clc
disp('metodo de Newton Rapshon ');
syms x
f=input('dame la funcion :');
pi=input('ingrese el valor del punto de inicio:');
err=input('dame el porcentaje de error:');
ezplot(f),grid on
d=diff(f);
d=inline(d);
f=inline(f);
ea=100;
j=0;
while ea>err
    xi=pi-(f(pi)/d(pi));
    ea=abs(((xi-pi)/xi)*100);
    pi=xi;
    j=j+1;
end
fprintf('\nRaiz=%12.3f en %4d iteraciones\n',pi,j);
```

SECANTE

```
clear
clc
f=input('ingrese a funcion:','s');
x1=input('ingrese el punto x1-1:');
x2=input('ingrese el punto xi :');
err=input('ingrese el porcentaje de error :');
ezplot(f),grid on
f=inline(f);
ea=100;
i=0;
while ea>err
xi=x2((f(x2)*(x1-x2))/(f(x1)-f(x2)));
ea =abs(((xi-x2)/xi)*100);
fprintf('   %f          %8.3f\n',i,xi);
x1=x2;
x2=xi;
i=i+1;
end
fprintf('\nRaiz e la funcion =%12.6f\nCalculada en%4d
iteraciones\n',xi,i);
```

BAIRSTOW

```
%datos de entrada
syms x;
f=x^5-3.5*x^4+2.75*x^3+2.125*x^2-3.875*x+1.25;
p=-1;
q=-1;
E=0.05;
E1=1;
E2=1;

%coeficientes de f
a=sym2poly(f);
a=a(length(a):-1:1);
n=length(a);

while(E1>E & E2>E)
    b(n+2)=0;
    b(n+1)=0;
    for i=n:-1:1
        b(i)=a(i)-p*b(i+1)-q*b(i+2);
    end
    c(n+2)=0;
    c(n+1)=0;
    for i=n:-1:1
       c(i)=b(i)-p*c(i+1)-q*c(i+2);
    end
    A=[c(2) c(3);c(3) c(4)];
    B=[b(1);b(2)];
    delta=inv(A)*B;
    p=p+delta(1);
    q=q+delta(2);
    E1=abs(delta(1)/p);
    E2=abs(delta(2)/q);
end
raiz1=(-p+sqrt(p^2-4*q))/2
raiz2=(-p-sqrt(p^2-4*q))/2
```

Muller

```
function [F]=muller()
f=input('introduzca la funcion: ');
x0=input('ingrese x0:');
x1=input('ingrese x1:');
x2=input('ingrese x2:');
e=input('ingrese ea:');
ezplot(f),grid on
%------------------------------------
fx=inline(f);
k=0;
xi=0;
sigue =1;
while(sigue)
        xi=x2;
        k=k+1;
        fprintf('iteracion:  ');
        fprintf('%f',k);

        h0=x1-x0;
        h1=x2-x1;
        d0=(fx(x1)-fx(x0))/h0;
        d1=(fx(x2)-fx(x1))/h1;
        a=(d1-d0)/(h1+h0);
        b=a*h1+d1;
        c=fx(x2);
        raizd=sqrt(b*b-4*a*c);
%-------------------------------
  if abs(b+raizd)>abs(b-raizd)
      den=b+raizd;
  else
      den=b-raizd;
  end
%--------------------------------
  dxr=-2*c/den;
  xr=x2+dxr;
  %validar si sigue el metodo
  sigue=abs(dxr)/xr>e||k<c||abs(fx(xr))>e;
  et=abs(xr-x2)/xr;
  ea=et*100;
  %hacer cambio de variables
  x0=x1;
  x1=x2;
  x2=xr;
  %--------------------------------
  fprintf('\nxi: ');
  fprintf('%f',xi);

  fprintf('\nea: ');
  fprintf('%f',ea);

  fprintf('\n               ');
end
```

INTERPOLACION

```
clc;
close all;
clear all;
x1=load('Datos_2_2.txt');
y1=load('Datos_2_1.txt');
inc=0.1;
np = length(y1);
y=y1(1,:);
fx1=y1(2,:);
for i1=1:np
    xp = x1(1,i1);
    x=interpn(y1,xp,1);
   interli(1,i1) = x;
end
ea=(x1(2,:)-interli).^2/np;
one=ones(np,1);
et=ea*one;
figure(1);
plot(x1(1,:),x1(2,:),'r.',y1(1,:),y1(2,:),'b.',x1(1,1:end),interli,'g.');
grid on;
xlabel('x');
ylabel('y');
title('Interpolacion lineal');
print('-f1', '-djpeg90', '-r300', 'GraficaI1.jpg');
fx = zeros(1,np);
for i1=1:np
    xp = x1(1,i1);
    x=interpn(y1,xp,2);
   interli2(1,i1) = x;
end
ea=(x1(2,1:end)-interli2).^2/np;
one=ones(np,1);
et2=ea*one;
figure(2);
plot(x1(1,:),x1(2,:),'r.',y1(1,:),y1(2,:),'b.',x1(1,1:end),interli2,'g.')
;
grid on;
xlabel('x');
ylabel('y');
title('Interpolacion Cuadrada');
print('-f2', '-djpeg90', '-r300', 'GraficaI2.jpg');

for i1=1:np
    xp = x1(1,i1);
    x=interpn(y1,xp,3);
   interli3(1,i1) = x;
end
ea=(x1(2,1:end)-interli3).^2/np;
one=ones(np,1);
et3=ea*one;
figure(3);
```

```
plot(x1(1,:),x1(2,:),'r.',y1(1,:),y1(2,:),'b.',x1(1,1:end),interli3,'g.')
;
grid on;
xlabel('x');
ylabel('y');
title('Interpolacion Cubica');
print('-f3', '-djpeg90', '-r300', 'GraficaI3.jpg');


for i1=1:np
    xp = x1(1,i1);
    x=interpn(y1,xp,4);
   interli4(1,i1) = x;
end
ea=(x1(2,1:end)-interli4).^2/np;
one=ones(np,1);
et4=ea*one;
figure(4);
plot(x1(1,:),x1(2,:),'r.',y1(1,:),y1(2,:),'b.',x1(1,1:end),interli3,'g.')
;
grid on;
xlabel('x');
ylabel('y');
title('Interpolacion Cuarta');
print('-f4', '-djpeg90', '-r300', 'GraficaI4.jpg');
```

INTERPOLACION 2D

```
clc;
close all;
clear all;

x1=load('Datos_3_2.txt');
y1=load('Datos_3_1.txt');
[n np]=size(x1);
figure(1)
mesh(y1(1,:), y1(2,:), y1(3:end,:));
grid on;
xlabel('x');
ylabel('y');
title('3D');
print('-f1', '-djpeg90', '-r300', 'GraficaID1.jpg');

figure(2)
mesh(y1(1,:), y1(2,:), y1(3:end,:));
hold on;
for i=1:np
    x(i,:)=x1(1,:);
    x2(i,:)=x1(2,:);
end
xi=x1(1,:);
xi2=x1(2,:);
yi=y1(1,:);
yi2= y1(2,:);
    plot3(x, x2', x1(3:end,:),'b.');
    mesh(y1(1,:), y1(2,:), y1(3:end,:));
grid on;
xlabel('x');
ylabel('y');
title('3D');
print('-f2', '-djpeg90', '-r300', 'GraficaID2.jpg');

%%
xi=x1(1,:);
xi2=x1(2,:);
yi=y1(1,:);
yi2= y1(2,:);
fxy=y1(3:end,:);
fxy2=x1(3:end,:);
incx=yi(2)-yi(1);
incy=yi2(2)-yi2(1);
nm=1;
for i=1:np
    for j=1:np
        yp=xi2(j);
        xp=xi(i);
       [it,xt]=pibotes(yi,xp,incx,nm);
       [it2,yt]=pibotes(yi2,yp,incy,nm);
        n=length(xt);
        for k=1:n
            p(2,k)=interpn([yi;fxy(:,it2(k))'],xp,nm);
        end
        p(1,:)=yi(it2);
        fxyp(i,j)=interpn(p,yp,nm);
    end
end
for i=1:n
    ea(i,:)=(fxyp(i,:)-fxy2(i,:)).^2;
end
et=0;
for i=1:n
    et=et+sum(ea(i,:))/(n*n);
end
    figure(3)
    plot3(x, x2', fxyp,'g.');
    hold on;
    plot3(x, x2', x1(3:end,:),'b.');
    mesh(y1(1,:), y1(2,:), y1(3:end,:));
    xlabel('x');
    ylabel('y');
    title('3D');
    grid on;
    print('-f3', '-djpeg90', '-r300', 'GraficaID3.jpg');

    nm=2;
for i=1:np
    for j=1:np
        yp=xi2(j);
        xp=xi(i);
       [it,xt]=pibotes(yi,xp,incx,nm);
       [it2,yt]=pibotes(yi2,yp,incy,nm);
        n=length(xt);
        for k=1:n
            p(2,k)=interpn([yi;fxy(:,it2(k))'],xp,nm);
        end
        p(1,:)=yi(it2);
        fxyp2(i,j)=interpn(p,yp,nm);
    end
end

for i=1:n
    ea(i,:)=(fxyp2(i,:)-fxy2(i,:)).^2;
end
et2=0;
for i=1:n
    et2=et2+sum(ea(i,:))/(n*n);
end
    figure(4)
    plot3(x, x2', fxyp2,'g.');
    hold on;
    plot3(x, x2', x1(3:end,:),'b.');
    mesh(y1(1,:), y1(2,:), y1(3:end,:));
    xlabel('x');
    ylabel('y');
    title('3D');
    grid on;
    print('-f4', '-djpeg90', '-r300', 'GraficaID4.jpg');

    nm=3;
for i=1:np
    for j=1:np
        yp=xi2(j);
        xp=xi(i);
       [it,xt]=pibotes(yi,xp,incx,nm);
       [it2,yt]=pibotes(yi2,yp,incy,nm);
        n=length(xt);
        for k=1:n
            p(2,k)=interpn([yi;fxy(:,it2(k))'],xp,nm);
        end
        p(1,:)=yi(it2);
        fxyp3(i,j)=interpn(p,yp,nm);
    end
end
for i=1:n
    ea(i,:)=(fxyp3(i,:)-fxy2(i,:)).^2;
end
et3=0;
for i=1:n
    et3=et3+sum(ea(i,:))/(n*n);
end
    figure(5)
    plot3(x, x2', fxyp3,'g.');
    hold on;
    plot3(x, x2', x1(3:end,:),'b.');
    mesh(y1(1,:), y1(2,:), y1(3:end,:));
    xlabel('x');
    ylabel('y');
    title('3D');
    grid on;
    print('-f5', '-djpeg90', '-r300', 'GraficaID5.jpg');
```

DESCOMPOSICION LU E INVERSION DE MATRICES

```
function x=LU(A,b)
[N, M] = size(A);
L = eye(N, M);
U = A;
Ld = b;
D=det(A);
for i=1:N
    j=i;
    while j<=N
        if U(i,j)<0.00000000000001
            j=j+1;
        else
            break;
        end
     U([i j],:)=U([j i],:);
     Ld([i j])=Ld([j i]);
    end
end

for i1=1:(M-1)
    for i2=(i1+1):N
        fct=U(i2,i1)/U(i1,i1);
        U(i2,:)=U(i2,:)-U(i1,:)*fct;
        Ld(i2)=Ld(i2)-Ld(i1)*fct;
        L(i2,i1)=fct;
    end
end

x = U\Ld;
```

MINIMOS CUADRADOS

```
function [a,f1]=Mcuadraos(x,y,ni)
nt=ni;
n=nt+1;
nx=nt*2;
nm = length(x);
for i=2:(nx)
    x(i,:)=x(1,:).^i;
end

for i=2:n
    y(i,:)=y(1,:).*x(i-1,:);
end
one=ones(nm,1);
xs=zeros(nx,1);
ys=zeros(n,1);
for i=1:nx
    xs(i,1)=x(i,:)*one;
end
for i=1:n
    ys(i,1)=y(i,:)*one;
end
M=ones(n);
M(1,1)=nm;
M(1,2:end)=xs(1:n-1);
j=n-1;
for i=2:n
    j=j+1;
    M(i,1:end)=[M(i-1,2:end) xs(j)];
end

a=LU(M,ys);
f=poly2sym(fliplr(a'));
f1=inline(f);
end
```

PUNTO FIJO

```
function punto fijo
gx=input('ingrese la funcion igualada a x: ','s');

g=inline(gx);
x=input('ingrese el x inicial:');
maxiter=input('ingrese el maximo de iteraciones a alcanzar :');
error=input('ingrese el error tolerado:');
i=1
fprintf('i \t x \t \t error \n');
fprintf('0 \t %f n',x);

while (i<=maxiter)
    a=x;
    x=g(a);
    e=abs(x-a);
    fprintf('%d \t %1.f \t \t %f \n',i,x,e);
    if(e<error)
        i=maxiter+1;
    end
    i=i+1;
end
fprint('la aproximacion es: %1.7f con un error de: %f \n',i,x,e);
end
```