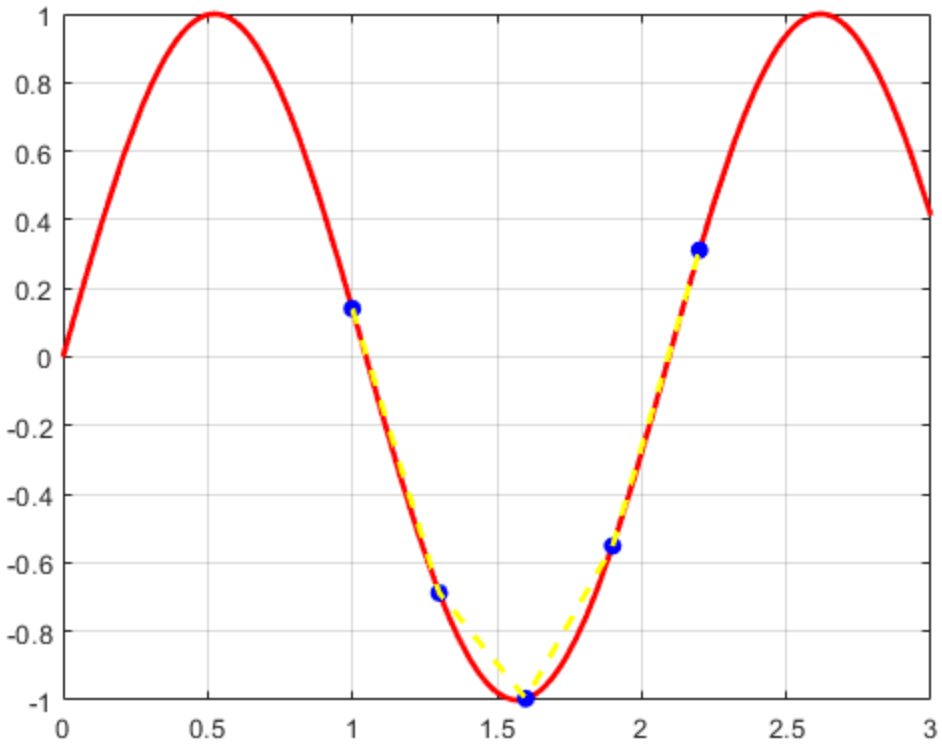
Table of Contents

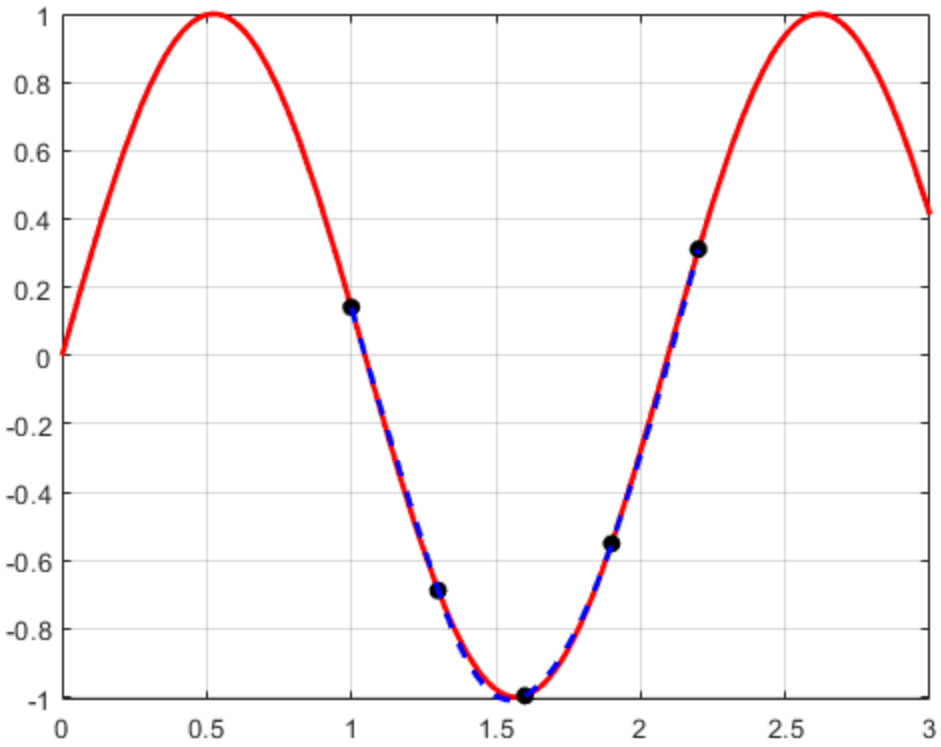
Ex1.

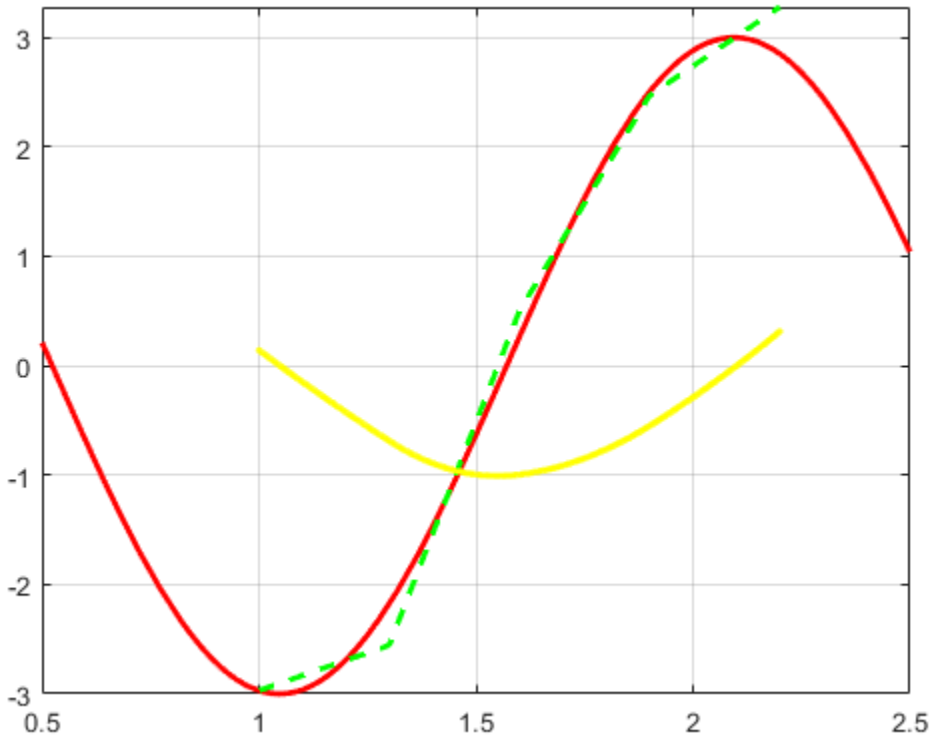
```
clear all
f=@(x)sin(3.*x);
n=4;
X=linspace(1,2.2,n+1);
Y=f(X);
fplot(f,[0,3],'-r','LineWidth',2);%Graficul Functiei
hold on;
grid on;
plot(X,Y,'.b','Markersize',23);%Graficul punctelor
xgraf=linspace(1,2.2,200);
for i=1:length(xgraf)
    [y(i)]=SplineL(X,Y,xgraf(i));
end
plot(xgraf,y,'--y','LineWidth',2);%Graficul Splineului
figure();
type ("SplineL");


function [y] = SplineL(X,Y,x)
    n=length(X)-1;
    for j=1:n
        a(j)=Y(j);
        b(j)=(Y(j+1)-Y(j))/(X(j+1)-X(j));
    end
    for j=1:n
        if (x>=X(j) & x<=X(j+1))
            S=a(j)+b(j)*(x-X(j));
            break
        end
    end
y=S;
end
```


1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
0.5
1
1.5
2
2.5
3

Ex2.

```
clear all;
f=@(x)sin(3.*x);
n=4;
X=linspace(1,2.2,n+1);
Y=f(X);
fplot(f,[0,3],'-r','LineWidth',2);%Graficul functiei
hold on;
grid on;
plot(X,Y,'.k','Markersize',23);%Graficul punctelor de interpolare
fp = @(x)3*cos(3.*x);
a=1;
fpa=fp(a);
xgraf=linspace(1,2.2,200);
for i=1:length(xgraf)
    [y(i),z(i)]=SplineP(X,Y,xgraf(i),fpa);
end
plot(xgraf,y,'--b','LineWidth',2);%Graficul Splineului
figure();
fplot(fp,[0.5,2.5],'-r','LineWidth',2);%Graficul derivatei functiei
hold on;
grid on;
plot(xgraf,z,'--g','LineWidth',2);%Graficul derivatei splineului
plot(xgraf,y,'.y','LineWidth',2);%Graficul splineului
type ("SplineP");
figure();


function [y,z] = SplineP(X,Y,x,fpa)
    n=length(X)-1;
    b(1)=fpa;
    h=X(2)-X(1);
    for j=1:n-1
        b(j+1)=(2/h)*(Y(j+1)-Y(j))-b(j);
    end
    for j=1:n
        a(j)=Y(j);
        c(j)=(1/h^2)*(Y(j+1)-Y(j)-h*b(j));
    end
    for j=1:n
        if(X(j)<=x && X(j+1)>=x)
            S=a(j)+b(j)*(x-X(j))+c(j)*((x-X(j))^2);
            ds=b(j)+2*c(j)*(x-X(j));
            break
        end
    end
    y=S;
    z=ds;
end
```


1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
0.5
1
1.5
2
2.5
3


3
2
1
0
-1
-2
-3
0.5
1
1.5
2
2.5

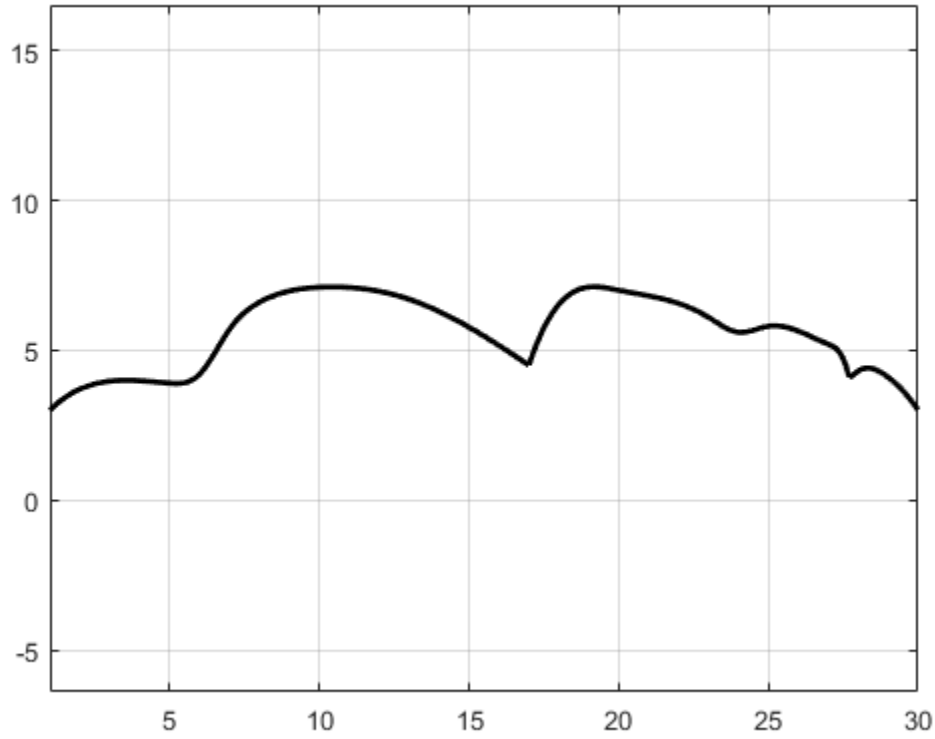
Ex3.

```
clear all;
X1=[1 2 5 6 7 8 10 13 17];
Y1=[3 3.7 3.9 4.2 5.7 6.6 7.1 6.7 4.5];
X2=[17 20 23 24 25 27 27.7];
Y2=[4.5 7 6.1 5.6 5.8 5.2 4.1];
X3=[27.7 28 29 30];
Y3=[4.1 4.3 4.1 3.0];
fpa1(1)=1;
fpa1(2)=-0.67;
fpa2(1)=3;
fpa2(2)=-4;
fpa3(1)=0.33;
fpa3(2)=-1.5;
xgr1=linspace(1,17,200);
xgr2=linspace(17,27.7,200);
xgr3=linspace(27.7,30,200);
for i=1:length(xgr1)
    [S1(i)]=SplinceC(X1,Y1,fpa1(1),fpa1(2),xgr1(i));
end
for i=1:length(xgr2)
    [S2(i)]=SplinceC(X2,Y2,fpa2(1),fpa2(2),xgr2(i));
end
for i=1:length(xgr3)
```

```
    [S3(i)]=SplinceC(X3,Y3,fpa3(1),fpa3(2),xgr3(i));
end
plot(xgr1,S1,'-k','LineWidth',2);
hold on;
grid on;
axis equal;
plot(xgr2,S2,'-k','LineWidth',2);
plot(xgr3,S3,'-k','LineWidth',2);
type ("SplinceC");
type ("GaussPivPart")


function S = Splinec(X, Y, sp1, sp2, x)
n = length(X) - 1;

% Aflam coeficientii aj
for j = 1:n
    h(j) = X(j+1) - X(j);
    a(j) = Y(j);
end

%Definim matricea asociata sistemului
% Aflam coeficientii bj

B(1, 1) = 1;
B(n+1, n+1) = 1;
for j = 2:n
    B(j,j-1) = 1/h(j-1);
    B(j, j) = (2/h(j) + 2/h(j-1));
    B(j,j+1) = 1/(h(j));
end

V(1) = sp1;
V(n+1) = sp2;
for j = 2:n
    V(j) = -3/h(j-1)^2 * Y(j-1) + (3/h(j-1)^2 - 3/h(j)^2) * Y(j) + 3/
h(j)^2 * Y(j+1);
end
b = GaussPivPart(B,V);

for j = 1:n
    d(j) = -2/h(j)^3 * (Y(j+1) - Y(j)) + 1/h(j)^2 * (b(j+1)+ b(j));
    c(j) = 3/h(j)^2 * (Y(j+1) - Y(j)) - (b(j+1) + 2 * b(j))/h(j);
end
for i = 1:length(x)
    for j = 1:n
        if x(i) >= X(j) && x(i) <= X(j+1)
           break;
        end
    end

S(i) = a(j) + b(j) * (x(i) - X(j)) + c(j) * (x(i) - X(j))^2 + d(j) *
 (x(i) - X(j))^3;
end
```

```
end

function [x] = GaussPivPart(A, b)
%GaussPivPart rezolva sisteme patratice de ecuatii liniare

%Synopsis: x = GaussPivPart(A, b)
%Input: A = matrice patratica
%       b = vectrul termanilor liberi
%Output: x = vectorul solutie

[m, n] = size(A);
%Se verifica daca matricea A este patratica
if m ~= n
    x = [];
    error('A nu este patratica')
end
%Verificare compatibilitate dimensiuni
if length(b) ~= n
    error('Incompatibilitate intre dimensiuni')
    x = [];
    return
end

Ae = A;
for i = 1:n
    Ae(i, n + 1) = b(i);
end
for k =1:n-1
    maxaux = abs(Ae(k, k));
    p = k;
    for j = k + 1:n
        if abs(Ae(j, k)) > maxaux
            maxaux = abs(Ae(j, k));
            p = j;
        end
    end
    if maxaux == 0
        error('Sistemul nu admite solutie unica');
        x = [];
        return
    end
    if p ~= k
    %    Ae([p, k], :)= Ae([k, p], :);
        for i = 1:n + 1
            linaux(i) = Ae(p, i);
            Ae(p, i) = Ae(k, i);
            Ae(k, i) = linaux(i);

        end
    end
    for i = k+1:n
        mik = Ae(i,k)/Ae(k,k);
        Ae(i,k)=0;
        Ae(i,k+1:n+1) = Ae(i, k+1:n+1) - mik * Ae(k, k+1:n+1);
```

```
    end
end
if Ae(n,n) == 0
    error('Sistemul este incomptaibil sau compatibil nedeterminat')
    x = [];
    return
end
A = Ae(1:n,1:n);
b = Ae(:,n+1);
x = SubsDesc(A,b);
end
```



Published with MATLAB® R2019a