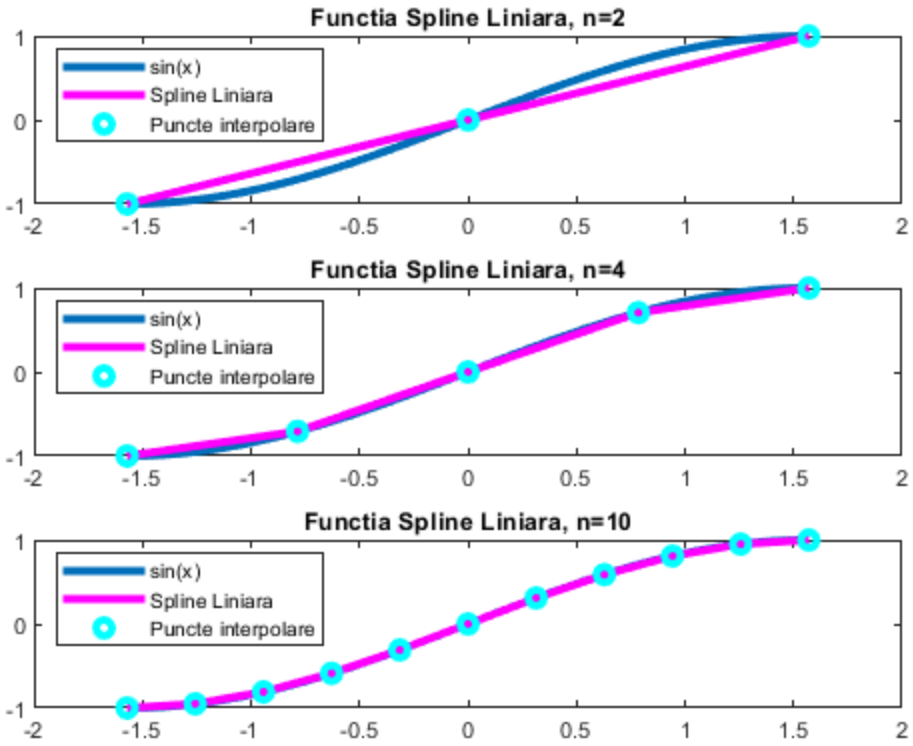
Tema lab06

Table of Contents

Problema 2

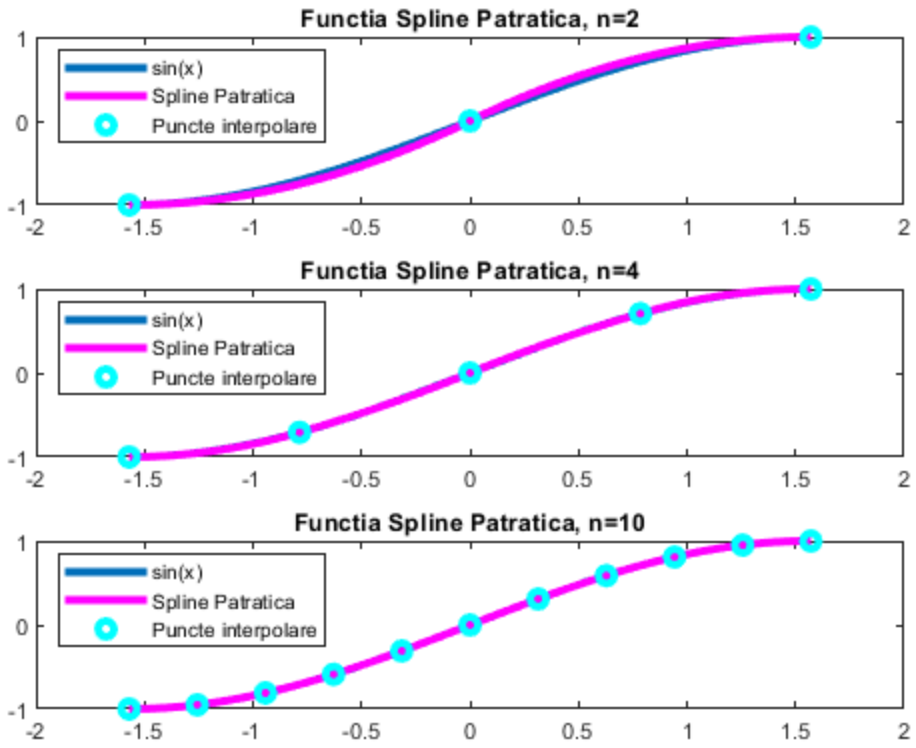
```
f = @(x)sin(x);
xmin = -pi/2;
xmax = pi/2;
x = linspace(xmin,xmax,100);

N = [2, 4, 10];

for index=1:3

    n = N(index);
    X = linspace(xmin,xmax,n+1);
    Y = f(X);

    figure(1);
    subplot(3,1,index);
    plot(x,f(x), 'LineWidth', 3);
    hold on;
    plot(x, SplineL(X, Y, x), 'm', 'LineWidth', 3); % reprezentarea
 grafica a functiei spline liniare
    plot(X, f(X), 'oc', 'LineWidth', 3); % punctele
    legend('sin(x)','Spline Liniara', 'Puncte
 interpolare', 'Location', 'northwest'); %plasarea legendei in coltul
 stanga sus
    title('Functia Spline Liniara, n='+string(n));

end
```

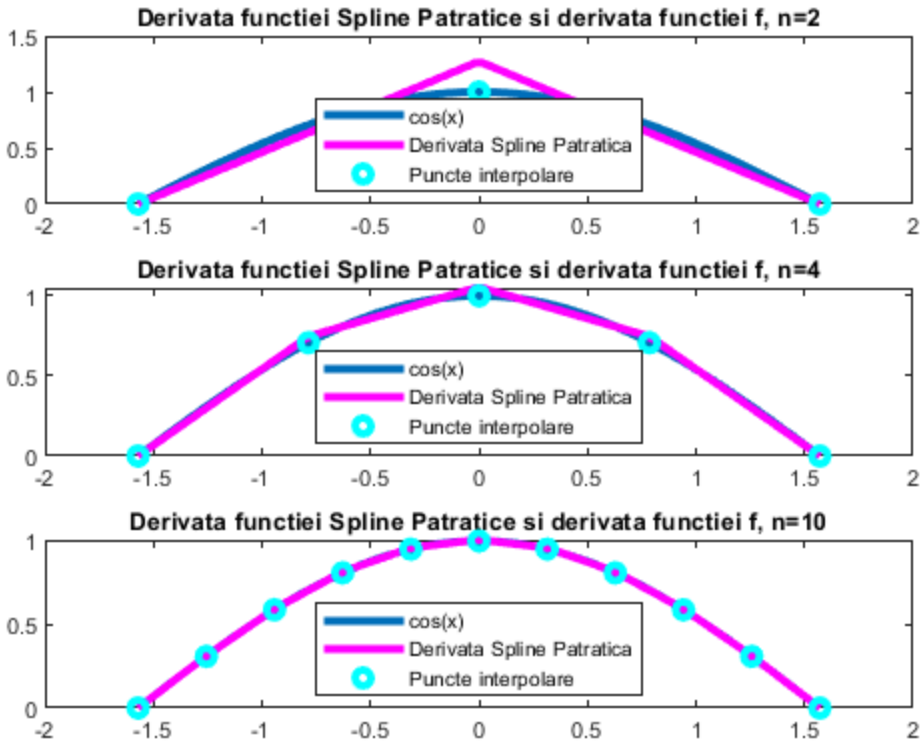


Problema 4

```
f = @(x)sin(x);
fp = @(x)cos(x);
xmin = -pi/2;
xmax = pi/2;
x = linspace(xmin,xmax,100);
fpa = fp(xmin);

N = [2, 4, 10];
for index=1:3

    n = N(index);
    X = linspace(xmin,xmax,n+1);
    Y = f(X);

    figure(2);

    subplot(3,1,index);
    plot(x,f(x), 'LineWidth', 3);
    hold on;

    [yP, zP] = SplineP(X, Y, fpa, x);

    plot(x, yP, 'm', 'LineWidth', 3);
```

```
    plot(X, f(X), 'oc', 'LineWidth', 3);
    legend('sin(x)','Spline Patratica', 'Puncte
 interpolare', 'Location', 'northwest'); % legenda stanga sus
    title('Functia Spline Patratica, n='+string(n));

    figure(3);

    subplot(3,1,index);
    plot(x, fp(x), 'LineWidth', 3);
    hold on;
    plot(x, zP, 'm', 'LineWidth', 3);
    plot(X, fp(X), 'oc', 'LineWidth', 3);
    legend('cos(x)','Derivata Spline Patratica','Puncte
 interpolare', 'Location', 'south');
    title('Derivata functiei Spline Patratice si derivata functiei f,
 n='+string(n));
end
```

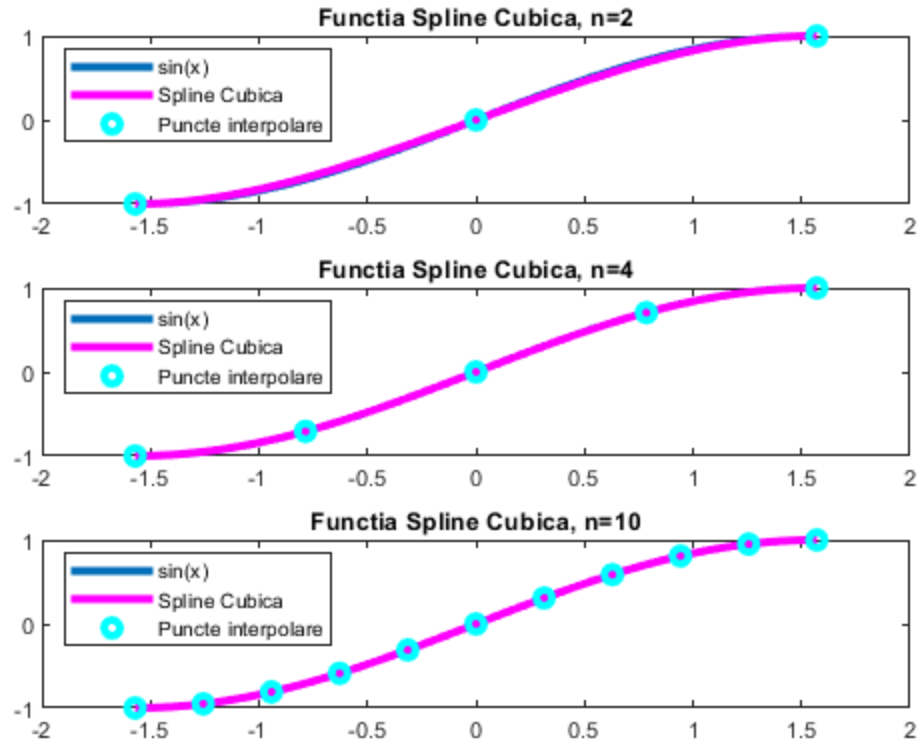


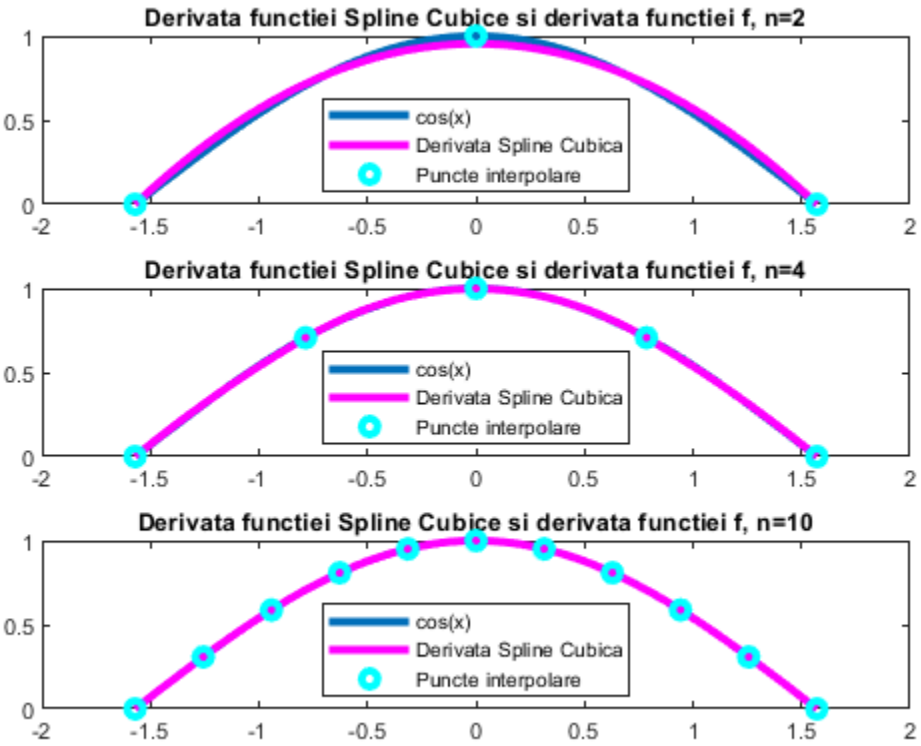


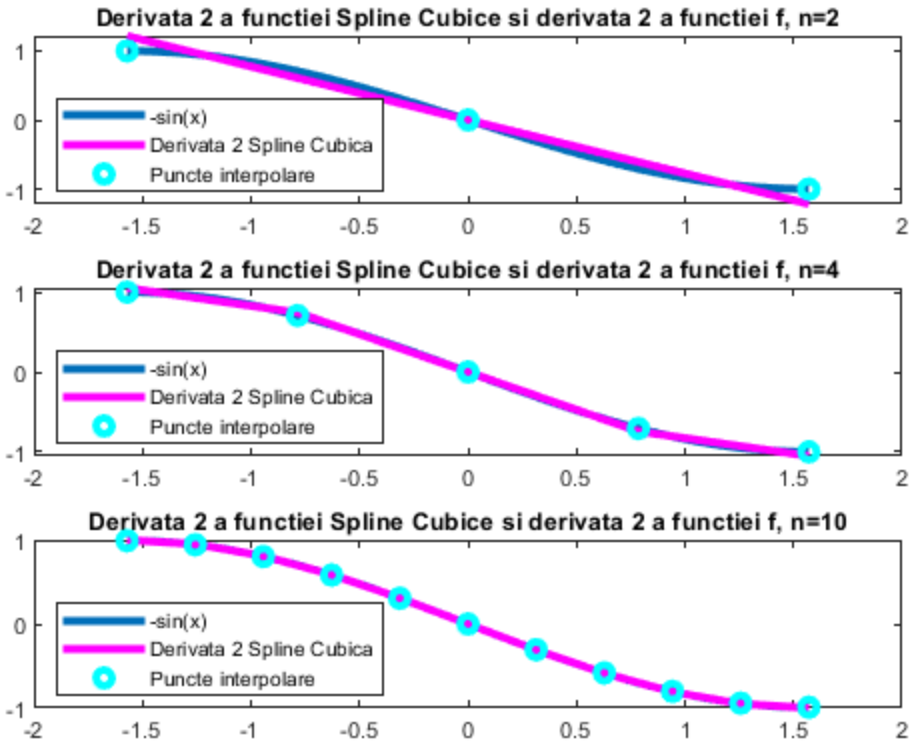
Problema 5

```
f = @(x)sin(x);
fp = @(x)cos(x);
fs = @(x)-sin(x);

xmin = -pi/2;
xmax = pi/2;
x = linspace(xmin,xmax,100);
fpa = fp(xmin);
fpb = fp(xmax);

N = [2, 4, 10];

for index=1:3
    n = N(index);
    X = linspace(xmin,xmax,n+1);
    Y = f(X);

    figure(4);

    subplot(3,1,index);
    plot(x,f(x), 'LineWidth', 3);
    hold on;
```

```
    [yCubic, zCubic, tCubic] = SplineCubic(X, Y, x, fpa, fpb);
    plot(x, yCubic, 'm', 'LineWidth', 3);
    plot(X, f(X), 'oc', 'LineWidth', 3);
    legend('sin(x)','Spline Cubica', 'Puncte
 interpolare', 'Location', 'northwest');
    title('Functia Spline Cubica, n='+string(n));

    figure(5);

    subplot(3,1,index);
    plot(x, fp(x), 'LineWidth', 3);
    hold on;
    plot(x, zCubic, 'm', 'LineWidth', 3);
    plot(X, fp(X), 'oc', 'LineWidth', 3);
    legend('cos(x)','Derivata Spline Cubica','Puncte
 interpolare', 'Location', 'south');
    title('Derivata functiei Spline Cubice si derivata functiei f,
 n='+string(n));

    figure(6);

    subplot(3,1,index);
    plot(x, fs(x), 'LineWidth', 3);
    hold on;
    plot(x, tCubic, 'm', 'LineWidth', 3);
    plot(X, fs(X), 'oc', 'LineWidth', 3);
    legend('-sin(x)','Derivata 2 Spline Cubica','Puncte
 interpolare', 'Location', 'southwest');
    title('Derivata 2 a functiei Spline Cubice si derivata 2 a
 functiei f, n='+string(n));
end
```


Functia Spline Cubica, n=2
sin(x)
Spline Cubica
Puncte interpolare
Functia Spline Cubica, n=4
sin(x)
Spline Cubica
Puncte interpolare
Functia Spline Cubica, n=10
sin(x)
Spline Cubica
Puncte interpolare

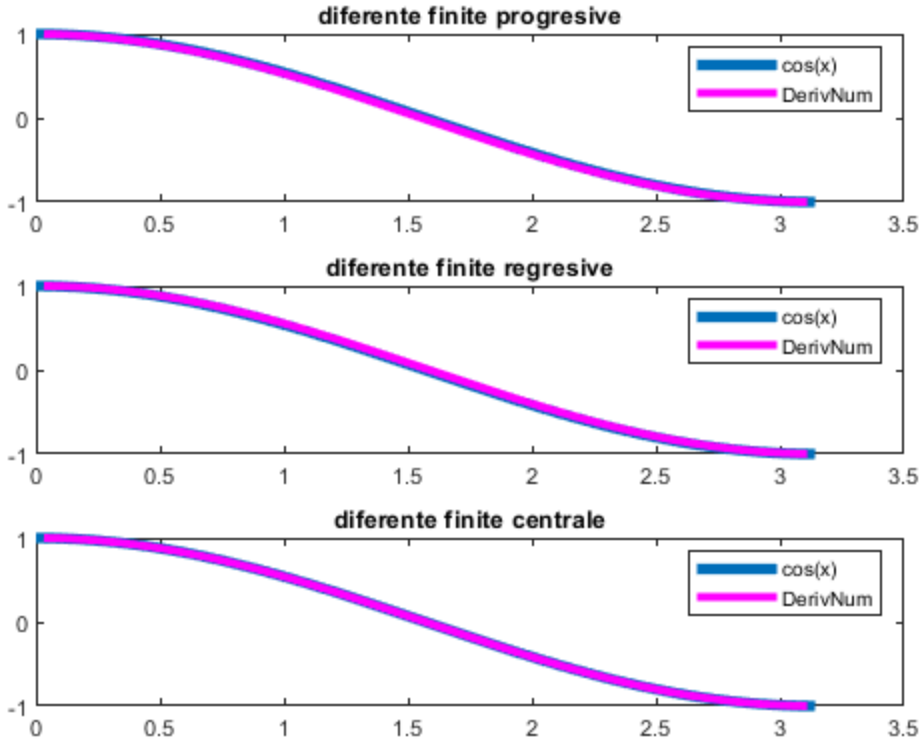

Derivata functiei Spline Cubice si derivata functiei f, n=2
cos(x)
Derivata Spline Cubica
Puncte interpolare
Derivata functiei Spline Cubice si derivata functiei f, n=4
cos(x)
Derivata Spline Cubica
Puncte interpolare
Derivata functiei Spline Cubice si derivata functiei f, n=10
cos(x)
Derivata Spline Cubica
Puncte interpolare

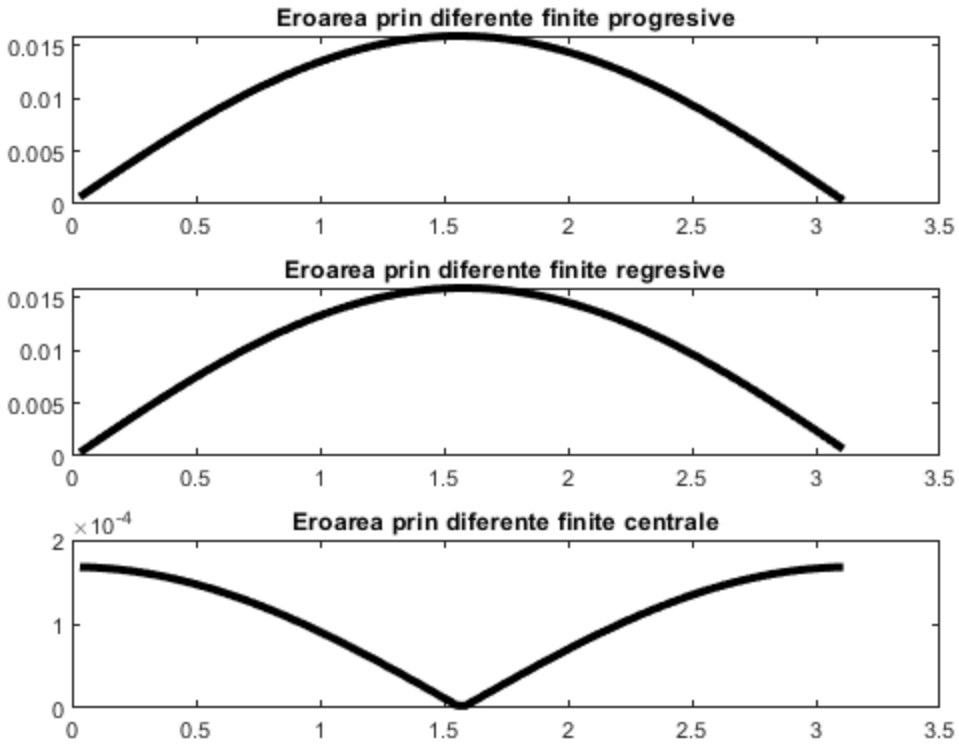


Problema 6

```
f = @(x)sin(x);
fp = @(x)cos(x);
a = 0;
b = pi;
m = 100;

x = linspace(a,b,m);
y = f(x);

metode = ["diferente finite progresive", "diferente finite
 regresive", "diferente finite centrale"];
for index = 1:3
    metoda = metode(index);
    dy = DerivNum(x,y,metoda);

    figure(7);
    subplot(3,1,index);
    plot(x,fp(x), 'LineWidth', 4);
    hold on;
    plot(x(2:length(x)-1),dy(2:length(dy)), 'm', 'LineWidth', 3);
    legend('cos(x)', 'DerivNum', 'Location', 'northeast');
    title(metoda);
```

```
    figure(8);
    subplot(3,1,index);
    plot(x(2:length(x)-1),abs(dy(2:length(dy)) -
 fp(x(2:length(x)-1))),'k', 'LineWidth', 3);
    title("Eroarea prin " + metoda);
end
```

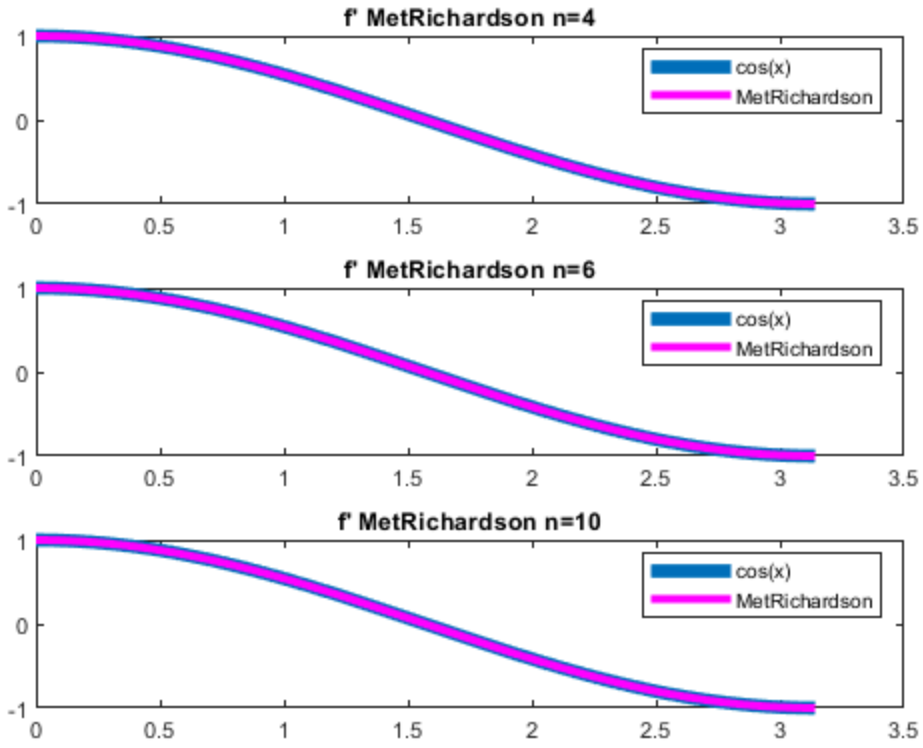


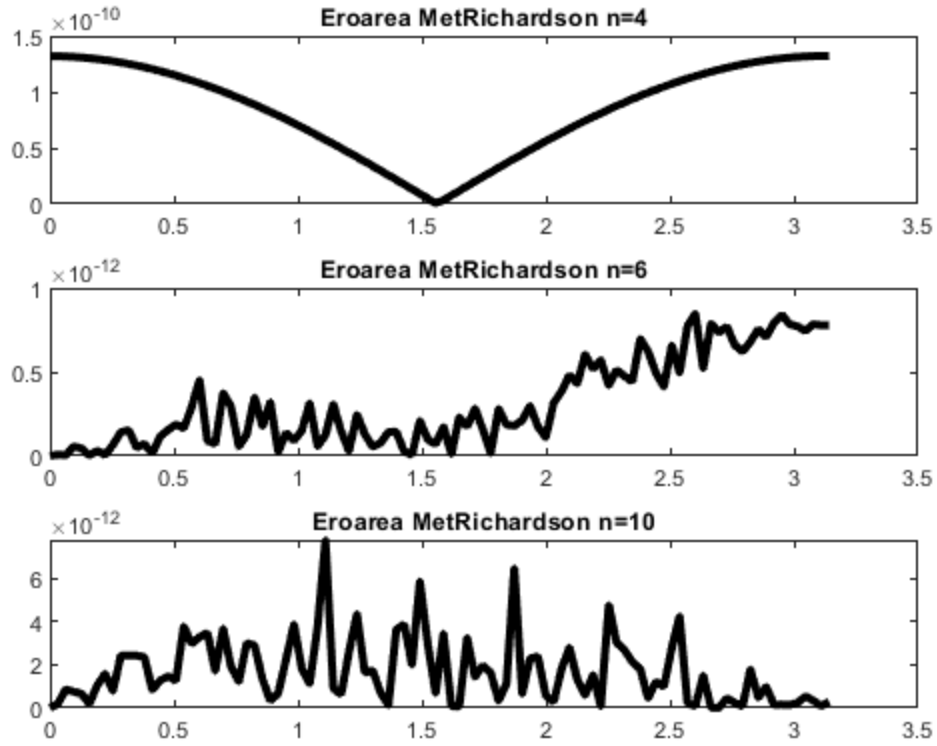


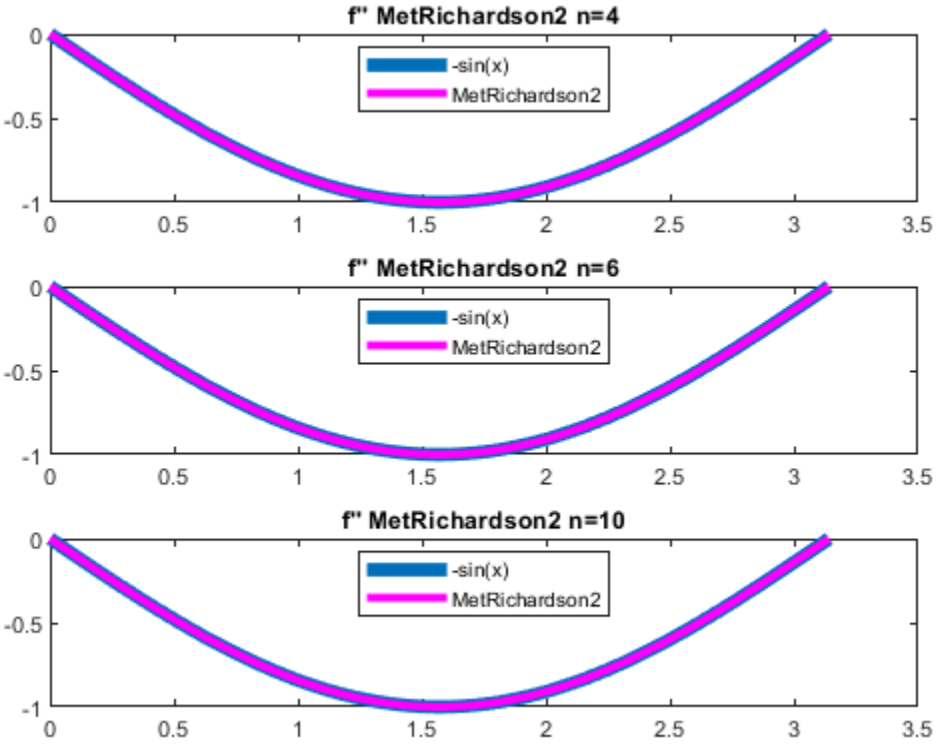
Problema 7

```
f = @(x)sin(x);
fp = @(x)cos(x);
fs = @(x)-sin(x);
a = 0;
b = pi;
N = [4, 6, 10];
x=linspace(a,b,100);
h=x(2)-x(1);

for index = 1:3
   n = N(index);

   figure(9);
   subplot(3,1,index);
   df = MetRichardson(f, x, h, n);
   plot(x,fp(x), 'LineWidth', 5);
   hold on;
   plot(x, df, 'm', 'LineWidth', 3);
   legend('cos(x)', 'MetRichardson', 'Location', 'northeast');
   title("f' MetRichardson n="+string(n));

   figure(10);
   subplot(3,1,index);
```

```
    plot(x, abs(df - fp(x)),'k', 'LineWidth', 3);
    title("Eroarea MetRichardson n="+string(n));

    figure(11);
    subplot(3,1,index);
    d2f = MetRichardson2(f, x, h, n-1);
    plot(x,fs(x), 'LineWidth', 5);
    hold on;
    plot(x, d2f, 'm', 'LineWidth', 3);
    legend('-sin(x)', 'MetRichardson2', 'Location', 'north');
    title("f'' MetRichardson2 n="+string(n));
end
```




Eroarea MetRichardson n=4
×10^-10
Eroarea MetRichardson n=6
×10^-12
Eroarea MetRichardson n=10
×10^-12


f'' MetRichardson2 n=4
f'' MetRichardson2 n=6
f'' MetRichardson2 n=10
-sin(x)
MetRichardson2

Algoritmi functii

```
function [y,z] = SplineP(X, Y, fpa, x)

  n = length(X)-1;

  for j=1:n
      a(j) = Y(j);
  end

  b(1) = fpa;
  h = X(2)-X(1);

  for j=1:n-1
      b(j+1) = (2*(Y(j+1) - Y(j)) / h) - b(j);
  end

  for j=1:n
      c(j) = (Y(j+1) - Y(j) - h*b(j))/(h*h);
  end

  for i=1:length(x)
    for j=1:n
      if x(i) >= X(j) && x(i) <= X(j+1)
          y(i) = a(j) + b(j)*(x(i)-X(j)) + c(j)*(x(i)-X(j))^2;
          z(i) = b(j) + 2*c(j)*(x(i)-X(j));
      end
    end
  end

end


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 i=1:length(x)
    for j=1:n
      if x(i) >= X(j) && x(i) <= X(j+1)
          y(i) = a(j) + b(j)*(x(i)-X(j));
      end
    end
  end

end
```

```
function [y,z,t] = SplineCubic(X,Y,x,fpa,fpb)

  n = length(X)-1;

  for j=1:n
      a(j) = Y(j);
  end

  B(1,1) = 1;
  B(n+1,n+1) = 1;

  for j=2:n
      B(j,j) = 4; %initializarea matricei
      B(j,j-1) = 1;
      B(j,j+1) = 1;
  end

  w(1) = fpa;
  w(n+1) = fpb;
  h = X(2)-X(1);

  for j=2:n
      w(j) = 3*(Y(j+1)-Y(j-1))/h;
  end

  w = w';
  b = B\w;

  for j=1:n
      d(j) = -2*(Y(j+1)-Y(j))/(h*h*h) + (b(j+1)+b(j))/(h*h);
      c(j) = 3*(Y(j+1)-Y(j))/(h*h) - (b(j+1)+2*b(j))/h;
  end

  for i=1:length(x)
      for j=1:n
          if x(i)<=X(j+1) && x(i)>=X(j)
              S = a(j) + b(j)*(x(i)-X(j)) + c(j)*(x(i)-X(j))*(x(i)-
X(j)) + d(j)*(x(i)-X(j))*(x(i)-X(j))*(x(i)-X(j));
              Sp = b(j) + 2*c(j)*(x(i)-X(j)) + 3*d(j)*(x(i)-
X(j))*(x(i)-X(j));
              Ss = 2*c(j) + 6*d(j)*(x(i)-X(j));
              break;
          end
      end

      y(i) = S;
      z(i) = Sp;
      t(i) = Ss;
  end

end


function [df] = MetRichardson(f, x, h, n)
```

```
  phi = @(x,h)(f(x+h)-f(x))/h;

  for k=1:length(x)

      for i=1:n
          Q(i,1) = phi(x(k),h/(2^(i-1)));
      end

      for i=2:n
          for j=2:i
              Q(i,j) = Q(i,j-1) + (Q(i,j-1)-Q(i-1,j-1))/(2^(j-1)-1);
          end
      end

      df(k) = Q(n,n);
  end

end


function [d2f] = MetRichardson2(f, x, h, n)

  phi = @(x,h)(f(x+h)-2*f(x)+f(x-h))/(h*h);

  for k=1:length(x)
      for i=1:n
          Q(i,1) = phi(x(k),h/(2^(i-1)));
      end

      for i=2:n
          for j=2:i
              Q(i,j) = Q(i,j-1) + (Q(i,j-1)-Q(i-1,j-1))/(2^(j-1)-1);
          end
      end

      d2f(k) = Q(n,n);
  end

end


function [y] = MetLagrange(X, Y, x)

  syms Q;
  n = length(X);
  Pn = 0;

  for k=1:n

      Lnk = 1;

      for i=1:n
          if i==k
```

```
                continue
            end

            Lnk = Lnk * (Q-X(i)) / (X(k)-X(i));
        end

        Pn = Pn + Lnk*Y(k);
    end

    Pn = matlabFunction(Pn, 'vars', {Q});
    vectorize(Pn);
    y = Pn(x);
end


function [dy] = DerivNum(x,y,metoda)

    m = length(x)-1;

      switch metoda % pentru fiecare tip de diferenta facem operatia
 necesara

        case 'diferente finite progresive'
            for i=2:m
                dy(i) = (y(i+1)-y(i)) / (x(i+1)-x(i));
            end

        case 'diferente finite regresive'
            for i=2:m
                dy(i) = (y(i)-y(i-1)) / (x(i)-x(i-1));
            end

        case 'diferente finite centrale'
            for i=2:m
                dy(i) = (y(i+1)-y(i-1)) / (x(i+1)-x(i-1));
            end
      end
end
```

Published with MATLAB® R2018b