**Main.m**

**%% Lagrange interpolation with evenly spaced nodes**

clc;

clf;

clear;

N = [5 9 11 15];

from = -1;

to = 1;

x = linspace(from, to, 1111);

analiticResults = arrayfun(@f, x);

hold on;

for n = N

nodes = generateEvenlySpacedNodes(@f, n, from, to);

y = LagrangeInterpolation(x, nodes);

plot(x, y, 'DisplayName', n + " wezlow interoplacji")

absoluteErrors = abs(y - analiticResults);

n

maxError = max(absoluteErrors)

meanError = mean(absoluteErrors)

end

grid on;

title("Interpolacja metoda Lagrangea z rownoodleglymi wezlami");

xlabel("x");

ylabel("f(x)");

legend();

**%% Newton interpolation with evenly spaced nodes**

clc;

clf;

clear;

N = [5 9 11 15];

from = -1;

to = 1;

x = linspace(from, to, 1111);

analiticResults = arrayfun(@f, x);

hold on;

for n = N

nodes = generateEvenlySpacedNodes(@f, n, from, to);

y = NewtonInterpolation(x, nodes);

plot(x, y, 'DisplayName', n + " wezlow interoplacji")

absoluteErrors = abs(y - analiticResults);

n

maxError = max(absoluteErrors)

meanError = mean(absoluteErrors)

end

grid on;

title("Interpolacja metoda Newtona z rownoodleglymi wezlami");

xlabel("x");

ylabel("f(x)");

legend();

**%% Lagrange interpolation with Czebyszew nodes**

clc;

clf;

clear;

N = [5 9 11 15];

from = -1;

to = 1;

x = linspace(from, to, 1111);

analiticResults = arrayfun(@f, x);

hold on;

for n = N

nodes = generateCzebyszewNodes(@f, n);

y = LagrangeInterpolation(x, nodes);

plot(x, y, 'DisplayName', n + " wezlow interoplacji")

absoluteErrors = abs(y - analiticResults);

n

maxError = max(absoluteErrors)

meanError = mean(absoluteErrors)

end

grid on;

title("Interpolacja metoda Lagrangea z wezlami Czebyszewa");

xlabel("x");

ylabel("f(x)");

legend();

**%% Newton interpolation with Czebyszew nodes**

clc;

clf;

clear;

N = [5 9 11 15];

from = -1;

to = 1;

x = linspace(from, to, 1111);

analiticResults = arrayfun(@f, x);

hold on;

for n = N

nodes = generateCzebyszewNodes(@f, n);

y = NewtonInterpolation(x, nodes);

plot(x, y, 'DisplayName', n + " wezlow interoplacji")

absoluteErrors = abs(y - analiticResults);

n

maxError = max(absoluteErrors)

meanError = mean(absoluteErrors)

end

grid on;

title("Interpolacja metoda Newtona z wezlami Czebyszewa");

xlabel("x");

ylabel("f(x)");

legend();

**function nodes = generateEvenlySpacedNodes(f, numberOfPoints, from, to)**

%GENERUJROWNOODDZIELONEWEZLY Generate equaly spaced interpolation nodes

nodes = zeros(2, numberOfPoints);

x = linspace(from, to, numberOfPoints);

y = zeros(1, numberOfPoints);

for i = 1:numberOfPoints

y(1, i) = f(x(1, i));

end

nodes(1, :) = x;

nodes(2, :) = y;

end

**function nodes = generateCzebyszewNodes(f, numberOfPoints)**

%CZEBYSZEWNODES Generate Czebyszew interpolation nodes in range <-1, 1>

nodes = zeros(2, numberOfPoints);

x = zeros(1, numberOfPoints);

for k = 1:numberOfPoints

x(k) = cos(pi \* (2\*k -1) / (2\*numberOfPoints));

end

y = zeros(1, numberOfPoints);

for i = 1:numberOfPoints

y(i) = f(x(i));

end

nodes(1, :) = x;

nodes(2, :) = y;

end

**function y = LagrangeInterpolation(x, interpolationNodes)**

%LAGRANGEINTERPOLATION Generate points values using Lagrangea Interpolation

numberOfNodes = size(interpolationNodes, 2);

L = ones(numberOfNodes, length(x));

for i = 1:numberOfNodes

for j = 1:numberOfNodes

if (i ~= j)

L(i, :) = L(i, :).\*((x - interpolationNodes(1, j)) / (interpolationNodes(1, i) - interpolationNodes(1, j)));

end

end

end

y = zeros(1, length(x));

for i = 1:numberOfNodes

y = y + interpolationNodes(2, i) \* L(i, :);

end

end

**function y = NewtonInterpolation(x, interpolationNodes)**

%NEWTONINTERPOLATION Generate points values using Newton interpolation

X = interpolationNodes(1, :);

Y = interpolationNodes(2, :);

numberOfNodes = size(interpolationNodes, 2);

newtonMatrix = zeros(numberOfNodes, numberOfNodes);

newtonMatrix(:, 1) = Y';

for j = 2:numberOfNodes

for i = j:numberOfNodes

newtonMatrix(i, j) = (newtonMatrix(i, j-1) - newtonMatrix(i-1, j-1)) / (X(i) - X(i-j+1));

end

end

a = diag(newtonMatrix)';

p = a(numberOfNodes);

for i=numberOfNodes-1:-1:1

p = [p a(i)] - [0 p\*X(i)];

end

y = polyval(p,x);

end