function wyniki = **metodaEulera**(dfdx, h, x0, y0, koniec)

%METODAEULERA Calkowanie numeryczne metoda Eulera

% dfdx - funkcja wartosci pochodnej y, h - krok, x0,y0 - warunki

% poczatkowe, koniec - ostatni punkt dla ktorego zosatnie policzony wynik

% Pocz?tkiem przedzia?u jest x0

liczbaWynikow = length(x0:h:koniec);

wyniki = zeros(2, liczbaWynikow);

wyniki(:, 1) = [x0; y0];

for i = 2:liczbaWynikow

xp = wyniki(1, i-1);

yp = wyniki(2, i-1);

xn = xp + h;

yn = yp + h\*dfdx(xp, yp);

wyniki(:, i) = [xn, yn];

end

end

function wyniki = **metodaEuleraDlaUkladow**(dxdt, dydt, h, x0, y0, t0, koniec)

%METODAEULERA Calkowanie numeryczne metoda Eulera

% dfdx - funkcja wartosci pochodnej y, h - krok, x0,y0 - warunki

% poczatkowe, koniec - ostatni punkt dla ktorego zosatnie policzony wynik

% Pocz?tkiem przedzia?u jest x0

liczbaWynikow = length(t0:h:koniec);

wyniki = zeros(3, liczbaWynikow);

wyniki(:, 1) = [x0; y0; t0];

for i = 2:liczbaWynikow

xp = wyniki(1, i-1);

yp = wyniki(2, i-1);

tp = wyniki(3, i-1);

xn = xp + h\*dxdt(xp, yp, tp);

yn = yp + h\*dydt(xp, yp, tp);

tn = tp + h;

wyniki(:, i) = [xn; yn; tn];

end

end

function wyniki = **metodaHeuna**(dfdx, h, x0, y0, koniec)

%METODAHEUNA Calkowanie numeryczne metoda Heuna

% dfdx - funkcja wartosci pochodnej y, h - krok, x0,y0 - warunki

% poczatkowe, koniec - ostatni punkt dla ktorego zosatnie policzony wynik

% Pocz?tkiem przedzia?u jest x0

liczbaWynikow = length(x0:h:koniec);

wyniki = zeros(2, liczbaWynikow);

wyniki(:, 1) = [x0; y0];

for i = 2:liczbaWynikow

xp = wyniki(1, i-1);

yp = wyniki(2, i-1);

xn = xp + h;

fi = (dfdx(xp, yp) + dfdx(xp+h, yp+h\*dfdx(xp, yp)))/2;

yn = yp + h\*fi;

wyniki(:, i) = [xn, yn];

end

end

function wyniki = metodaHeunaDlaUkladow(dxdt, dydt, h, x0, y0, t0, koniec)

%METODAHEUNA Calkowanie numeryczne metoda Heuna

% dfdx - funkcja wartosci pochodnej y, h - krok, x0,y0 - warunki

% poczatkowe, koniec - ostatni punkt dla ktorego zosatnie policzony wynik

% Pocz?tkiem przedzia?u jest x0

liczbaWynikow = length(t0:h:koniec);

wyniki = zeros(3, liczbaWynikow);

wyniki(:, 1) = [x0; y0; t0];

for i = 2:liczbaWynikow

xp = wyniki(1, i-1);

yp = wyniki(2, i-1);

tp = wyniki(3, i-1);

fi\_x = (dxdt(xp, yp, tp) + dxdt(xp+h, yp+h\*dxdt(xp, yp, tp)))/2;

fi\_y = (dydt(xp, yp, tp) + dydt(xp+h, yp+h\*dydt(xp, yp, tp)))/2;

xn = xp + h\*fi\_x;

yn = yp + h\*fi\_y;

tn = tp + h;

wyniki(:, i) = [xn; yn; tn];

end

end

function wyniki = **metodaAdamsaBashforthaDlalUkladow**(dxdt, dydt, h, x0, y0, t0, koniec, q)

%METODAADAMSABASHFORTHADLALUKLADOW Calkowanie numeryczne metoda

%Adamsa-Bashfortah

% dfdx - funkcja wartosci pochodnej y, h - krok, x0,y0 - warunki

% poczatkowe, koniec - ostatni punkt dla ktorego zosatnie policzony wynik

% q - rzad metody

% Poczatkiem przedzialu jest x0

if q < 0 || q > 4

wyniki = NaN;

return;

end

wspolczynniki = zeros(5, 5);

wspolczynniki(1, :) = [1 0 0 0 0];

wspolczynniki(2, :) = [3 -1 0 0 0] / 2;

wspolczynniki(3, :) = [23 -16 5 0 0] / 12;

wspolczynniki(4, :) = [55 -59 37 -9 0] / 24;

wspolczynniki(5, :) = [1901 -2774 2616 -1274 251] / 720;

liczbaWynikow = length(t0:h:koniec);

wyniki = zeros(3, liczbaWynikow);

koniecEulera = t0 + h \* q;

punktyStartowe = metodaEuleraDlaUkladow(dxdt, dydt, h, x0, y0, t0, koniecEulera);

wyniki(:, 1:q+1) = punktyStartowe;

for i = q+2:liczbaWynikow

p = i-1;

xn = 0;

for j = 0:q

xn = xn + wspolczynniki(q+1, j+1) \* dxdt(wyniki(1, p-j), wyniki(2, p-j), wyniki(3, p-j));

end

xn = xn \* h + wyniki(1, p);

yn = 0;

for j = 0:q

yn = yn + wspolczynniki(q+1, j+1) \* dydt(wyniki(1, p-j), wyniki(2, p-j), wyniki(3, p-j));

end

yn = yn \* h + wyniki(2, p);

tn = wyniki(3, p) + h;

wyniki(:, i) = [xn, yn, tn];

end

end

%% **Przykladowe zadanie 1**

% Dana funkcja

% y' + y = -2\*pi\*e^(-x)\*sin(2\*pi\*x)

% y' = -2\*pi\*e^(-x)\*sin(2\*pi\*x) - y

% Warunki poczatkowe

% x0 = 0, y0 = y(x0) = 1

% Dodatkowe dane

% Policzy? w przedziale <0, 10>

% h = 0,001

% Wynik analityczny = y(x)=e^(-x)\*cos(2\*pi\*x)

clc;

clear;

dydx = @(x, y) (-2\*pi\*exp(-x)\*sin(2\*pi\*x) - y);

h = 0.001;

wynikEulera = metodaEulera(dydx, h, 0, 1, 10);

wynikHeuna = metodaHeuna(dydx, h, 0, 1, 10);

% Wynik analityczny

f = @(x) (exp(-x)\*cos(2\*pi\*x));

x = 0:h:10;

y = arrayfun(f, x);

plot(x, y, '-', 'DisplayName','Wynik analityczny');

title("y = e\^(-x)\*cos(2\*pi\*x) oraz wyniki rozwiazan metodami numerycznymi");

xlabel("x");

ylabel("y");

hold on;

grid on;

plot(wynikEulera(1, :), wynikEulera(2, :), 'o', 'DisplayName','Rozwiazania metoda Eulera');

plot(wynikHeuna(1, :), wynikHeuna(2, :), 'x', 'DisplayName','Rozwiazania metoda Heuena');

legend();

bledyEulera = abs(y - wynikEulera(2, :));

maksBladEulera = max(bledyEulera)

bledyHeuna = abs(y - wynikHeuna(2, :));

maksBladHeuna = max(bledyHeuna)

%% Zadanie 2

clc;

clear;

dxdt = @(x, y, t) (y);

dydt = @(x, y, t) (-x);

x0 = 0;

y0 = 1;

t0 = 0;

t\_max = 10;

h = [0.1; 0.01; 0.001];

% Wynik analityczny

f = @(t) (sin(t));

for i = 1:length(h)

t = t0:h(i):t\_max;

x = arrayfun(f, t);

wynikiEulera = metodaEuleraDlaUkladow(dxdt, dydt, h(i), x0, y0, t0, t\_max);

wynikiHeuna = metodaHeunaDlaUkladow(dxdt, dydt, h(i), x0, y0, t0, t\_max);

bledyEulera = abs(x - wynikiEulera(1, :));

maksBladEulera = max(bledyEulera)

bledyHeuna = abs(x - wynikiHeuna(1, :));

maksBladHeuna = max(bledyHeuna)

end

plot(t, x, '-', 'DisplayName','Wynik analityczny');

hold on;

plot(wynikiEulera(3, :), wynikiEulera(1, :), 'x', 'DisplayName','Rozwiazania metoda Eulera');

plot(wynikiHeuna(3, :), wynikiHeuna(1, :), 'o', 'DisplayName','Rozwiazania metoda Heuna');

%plot(wynikiEulera(3, :), wynikiEulera(2, :), '-', 'DisplayName','Rozwiazania metoda Eulera');

%% Zadanie 3

clc;

clear;

dxdt = @(x, y, t) (y);

dydt = @(x, y, t) (-x);

x0 = 0;

y0 = 1;

t0 = 0;

t\_max = 10;

h = 0.001;

q = 1;

wynikiDwukrokowe = metodaAdamsaBashforthaDlalUkladow(dxdt, dydt, h, x0, y0, t0, t\_max, 1);

wynikiTrzykrokowe = metodaAdamsaBashforthaDlalUkladow(dxdt, dydt, h, x0, y0, t0, t\_max, 2);

% Wyniki analityczne

T = 0:h:t\_max;

X = sin(T);

bladDwykrokowy = max(abs(wynikiDwukrokowe(1, :) - X))

bladTrzykrokowy = max(abs(wynikiTrzykrokowe(1, :) - X))