MNUM-PROJEKT 3, zadanie 3.25

Wojciech Grunwald (311566)

01.06.2023

1 Wstęp

Celem projektu jest obliczenie przebiegu trajektorii ruchu punktu w danym przedziale dla pewnych warunków początkowych, korzystając ze środowiska *Matlab* dwoma różnymi metodami:

1. metodą Heuna przy zmiennym kroku z szacowaniem błędu metodą zdwajania kroku
2. solwerem ode45 (gotowy w Matlabie) - metoda średniego rzedu

Układ równań:

$$\frac{\mathrm{d}x_1}{\mathrm{d}t} = x_2 + x_1(0.3 - x_1^2 - x_2^2)$$
$$\frac{\mathrm{d}x_2}{\mathrm{d}t} = -x_1 + x_2(0.3 - x_1^2 - x_2^2)$$

w przedziale [0, 10] dla warunków początkowych:

$$x_1(0) = 9$$
$$x_2(0) = -7$$

Algorytmy zaimplementowałem w *Matlabie* w wersji R2023a. Obliczenia były wykonywane na procesorze Intel Core i5-9300H CPU 2.40Ghz.

W folderze *Kod źródłowy* załączonym do sprawozdania znajdują się odpowiednie skrypty rozwiązujące zadania:

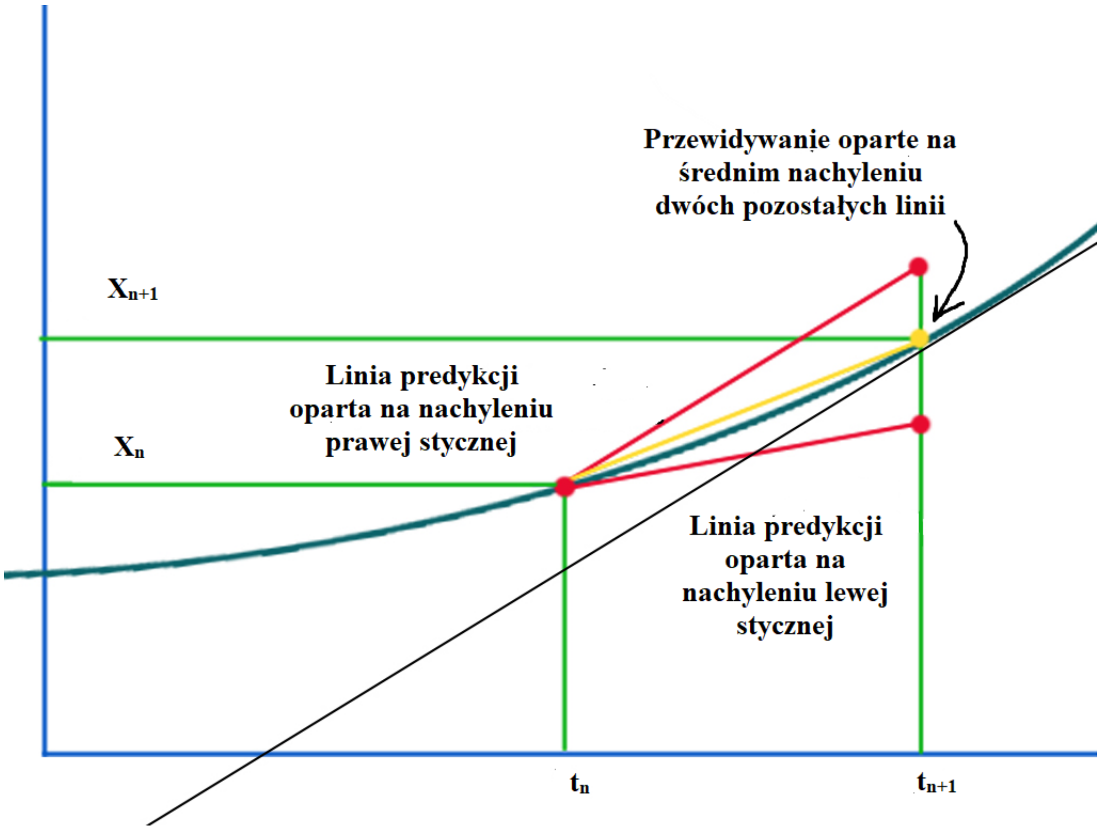
1. diffEq.m - skrypt z opisanym układem równań różniczkowych zwracający wartości funkcji na zmiennych
2. Heun.m - skrypt z zaimplementowaną metodą Heuna
3. diffEqVisualizerODE.m - skrypt uruchamiający rozwiązywanie równania metodą Heuna i wizualizujący zależności na wykresach

2 Obliczenie przebiegu trajektorii ruchu metodą Heuna

2.1 Wprowadzenie teoretyczne

Metoda Heuna (metoda Rungego-Kutty 2 rzędu) jest ulepszoną wersją metody Eulera:

$$x_{i+1} = x_i + \frac{1}{2}h[f(t_i, x_i) + f(t_i + h, x_i + hf(t_i, x_i))] \tag{1}$$



Rysunek 1: Graficzne wyjaśnienie metody Heuna

2.1.1 Szacowanie wartości błędu według zasady podwójnego kroku

W celu szacowania błędu, oprócz kroku o długości h wykonujemy dodatkowo, równolegle (tzn. startując też z tego samego punktu t_n) i dokładnie tą samą metodą, dwa dodatkowe kroki o długości 0.5h każdy. Wprowadzając oznaczenia:

- y_n^1 - nowy punkt uzyskany w kroku o długości h
- y_n^2 - nowy punkt wyznaczony przez dwa kroki o długościach 0.5h

Można dowieść, że oszacowanie błędu uzyskanego z wykonania dwóch podkroków jest mniejsze niż przy jednym kroku, więc praktycznziejsze jest przyjmowanie po prostu:

$$y_{n+1} = y_n^2 \tag{2}$$

Dla układu k równań przymuje się współczynnik wyliczony dla najbardziej krytycznego równania (na najgorszy przypadek), w naszym przypadku z szacowaniem błędu według zasady podwójnego kroku, mamy:

$$\delta_n(h)_i = \frac{(y_i)_n^{(2)} - (y_i)_n^{(1)}}{2^p - 1} \qquad i = 1, 2, ..., k \tag{3}$$

$$\varepsilon_i = \left|(y_i)_n^{(2)}\right| \varepsilon_w + \varepsilon_b \tag{4}$$

$$\alpha = \min_{1 \leqslant i \leqslant k} \left(\frac{\varepsilon_i}{|\delta_n(h)_i|} \right)^{\frac{1}{p+1}} \tag{5}$$

Gdzie:

- h - krok metody

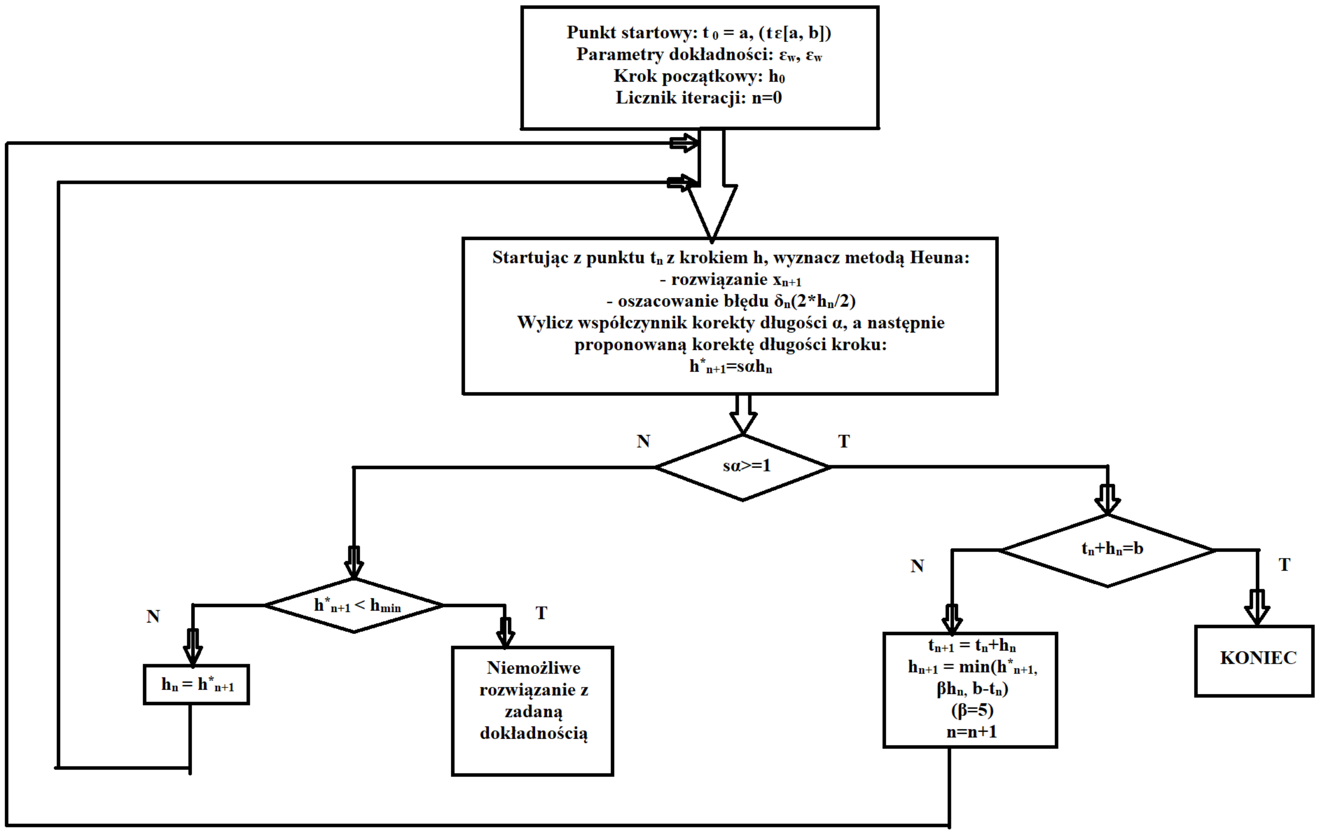
- p - rząd metody
- ϵ - parametr dokładności obliczeń
- ϵ_w - błąd względny
- ϵ_b - błąd bezwzględny
- α - współczynnik korekty długości kroku α
- $\delta_n(h)$ - oszacowanie błędu w n-tym kroku

W praktyce dla uwzględnienia niedokładności oszacowania błędu stosuje się jeszcze współczynnik bezpieczeństwa, stąd wyliczenie następnego kroku:

$$h_{n+1} = s\alpha h_n \tag{6}$$

Gdzie $s < 1$ (przyjęte jest s=0.9)

Sieć działań metod Rungego-Kutty:

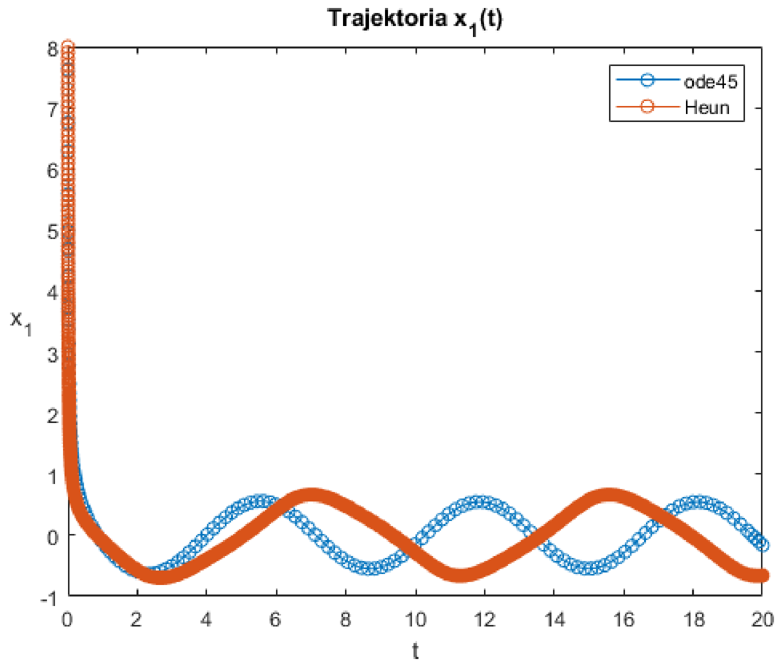


Rysunek 2: Sieć działań w podstawowych realizacjach metod RK i RKF

Warto zwrócic uwagę na zastosowane w tym schemacie heurystyczne ograniczenie maksymalnego wzrostu długości kroku h_n w jednej iteracji.

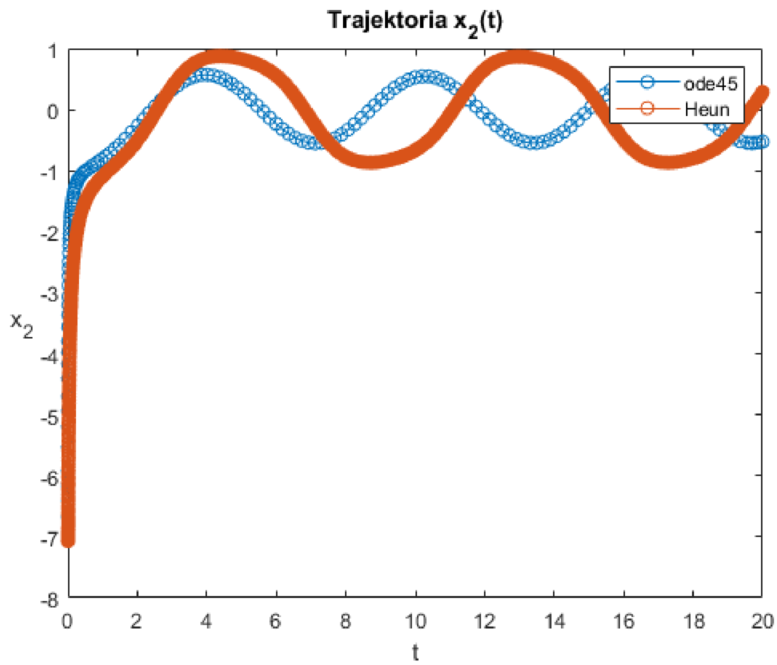
2.2 Testowanie własnego solwera i porównanie z ode45

2.2.1 Trajektoria $x_1(t)$ metodą Heuna i przy użyciu solwera ode45



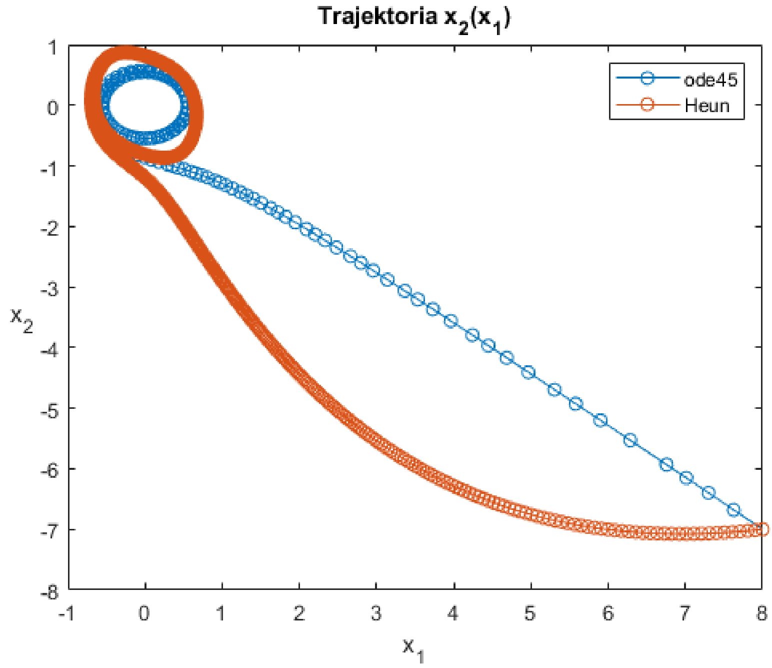
Rysunek 3: Trajektoria $x_1(t)$

2.2.2 Trajektoria $x_2(t)$ metodą Heuna i przy użyciu solwera ode45



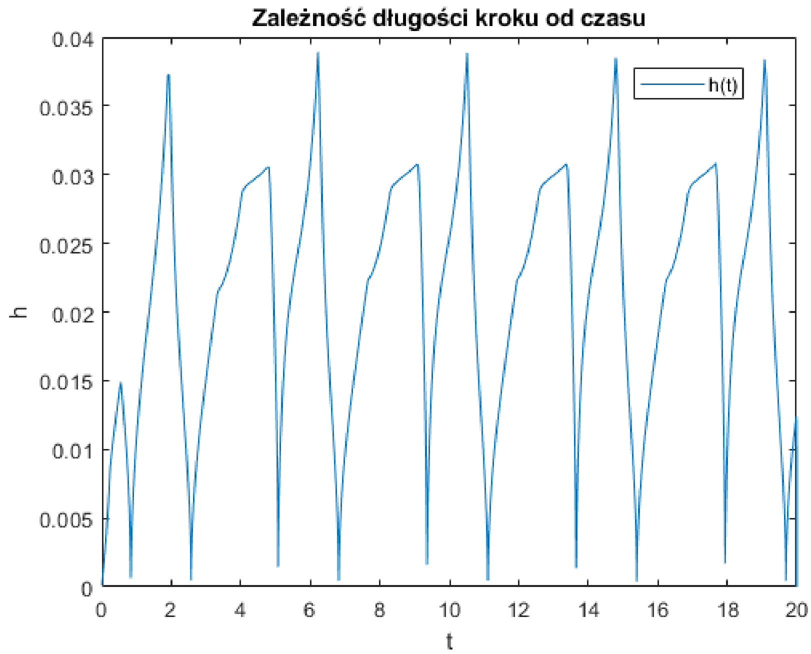
Rysunek 4: Trajektoria $x_2(t)$

2.2.3 Trajektoria (x_1, x_2) metodą Heuna i przy użyciu solwera ode45



Rysunek 5: Trajektoria w przestrzeni fazowej

2.2.4 Zależność długości kroku od czasu dla metody Heuna



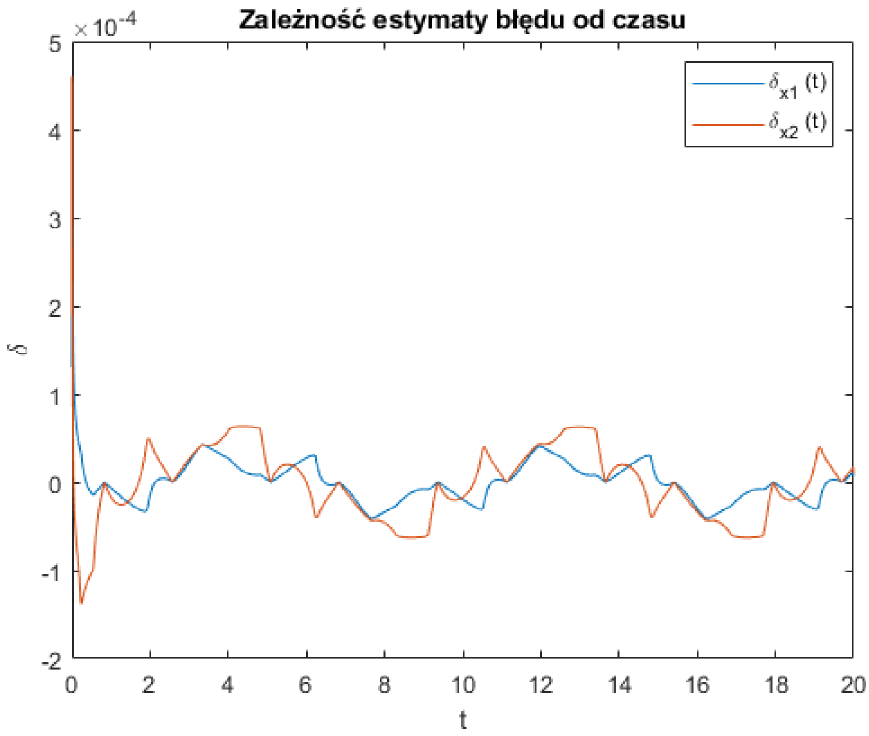
Rysunek 6: Zależność długości kroku h od czasu t dla metody Heuna

Z wykresu wynika, że krok jest zmieniany w sposób pierodyczny: pętla progrmau rozchodzi się na gałąź zarówno zmniejszania kroku, jak i jego zwiększania w zależności od wyliczonego współczynnika korekty długości kroku α. Jak widać - metoda jest tutaj niestabilna: istnieją pewne układy równań różniczkowych, dla których metoda Heuna może generować niestabilne rozwiązania lub oscylacje wokół prawidłowego rozwiązania. **Warto rozważyć zastosowanie bardziej zaawansowanych metod numerycznych, takich jak metoda Rungego-Kutty czwartego rzędu.**

Również czas działania metody Heuna był znacznie większy niż ODE45;

- $t_Heun = 0.0216$ s
- $t_{ODE45} = $ 1e-4 s

2.2.5 Zależność estymaty błędu od czasu dla metody Heuna



Rysunek 7: Zależność estymaty błędu od czasu dla metody Heuna

Jak widać na załączonym wykresie - estymata błędu dla obu zmienncyh na początku jest duża, a następnie stabilizuje się wokół pewnej wartości średniej.

2.2.6 Minimalny krok, dokładność względna i bezwzględna

h_{min}	ϵ_w	ϵ_b
0.0001	1e-4	1e-8

Wymienione wartości parametrów: kroku h_{min}, dokładności względnej ϵ_w i bezwzględnej ϵ_b dobrałem eksperymentalnie tak, żeby możliwe było obliczenie kolejnych kroków.

3 Listingi funkcji

3.1 diffEq.m

```
function dxdt = diffEq(t, x)
    dxdt = [x(2)+x(1)*(0.3-x(1)^2-x(2)^2); -x(1)+x(2)*(0.3-x(1)^2-x(2)^2)];
end
```

3.2 Heun.m

```
function [t, x, deltas, H] = Heun(func, k, range, cond, hmin, ew, eb)
    % func - funkcja obliczaj ca warto  funkcji po prawej stronie r wnania
    % r  niczkowego
    % k - ilo   r wna
    % range - zakres oblicze
    % cond - warunki pocz tkowe, cond(1/2) - warunek dla zmiennej x1/2
    % hmin - minimalny krok (pocz tkowy)
    % ew - blad wzgledny
    % eb - blad bezwzgledny

    deltas = [] % tablica estymaty b edu
    H = [hmin] % tablica d ugo ci krok w
    p=2;
    beta = 1.5; % heurystyka
    s=0.7; % wsp  czynnik bezpiecze stwa
    x=[cond]; %wektor zmiennych x1, ..., xn
    t=[0];
    h=hmin;
    n = 0; %licznik iteracji

    while 1
        %h wyznaczenie zwykla metoda Heuna potrzebne do oszacowania bledu
        f_value = func(0, x(end, :));
        f_nested_value = func(0, x(end, :)+h*f_value);

        x_next = [];
        for i=1:k
            x_next = [x_next, x(end, i) + 1/2*h*(f_value(i)+f_nested_value(i))];
        end

        h_half = h/2;

        %0.5h wyznaczenie nastepnej wartosci z podwojnym krokiem

        f_value = func(0, x(end, :));
        f_nested_value = func(0, x(end, :)+h_half*f_value);

        x_next_half_h = [];
        for i=1:k
```

```
        x_next_half_h = [x_next_half_h , x(end, i) + 1/2*h_half*(f_value(i)+f_nested_value(i))];
    end

    f_value = func(0, x_next_half_h);
    f_nested_value = func(0, x_next_half_h+h_half*f_value);

    x_full_half_h = [];
    for i=1:k
        x_full_half_h = [x_full_half_h , x_next_half_h(i) + 1/2*h_half*(f_value(i)+f_nested_value(i))];
    end

    % oszacowanie b  du
    DELTA = [];
    ERROR = [];
    for i=1:k
        DELTA = [DELTA, (x_full_half_h(i)-x_next(i))/(2^p-1)];
        ERROR = [ERROR, abs(x_full_half_h(i))*ew+eb];
    end
    deltas = [deltas; DELTA];

    % wyliczenie wsp  czynnika korekty d ugo ci kroku
    % 0.00000001,  eby  nie dzieli  przez zero w ostatnim kroku
    alpha = min((ERROR./(abs(DELTA)+0.00000001)).^(1/(p+1)));

    h_star = s*alpha*h;

    if s*alpha>=1
        if t(end)+h==range(2)
            disp("KONIEC, OSI GNIETY KRANIEC PRZEDZIA U")
            break
        end
        ogr = beta*h;
        h=min([h_star, ogr, range(2)-t(end)]);

        t_next = t(end)+h;
        t = [t, t_next];
        n=n+1;
    else
        if h_star<hmin
            disp("Niemozliwe rozwiazanie z zadana dokladnoscia")
            break;
        else
            h=h_star;
            t_next = t(end)+h;
            t = [t, t_next];
        end
    end
```

```
            H = [H, h];
            x = [x; x_full_half_h];
            n=n+1;
        end

end
```

3.3 diffEqVisualizerODE.m

```
tic;
[tODE, xODE] = ode45(@diffEq, [0, 20], [8; -7]);
timerode45=toc
tic;
[t, x, deltas, H] = Heun(@diffEq, k, [0, 20], [8, -7], 0.0001, 1e-4, 1e-8);
timerHeun=toc

figure(1)
plot(tODE, xODE(:, 1), '-o', 'DisplayName', 'ode45')
title('Trajektoria x_1(t)')
hold on
plot(t, x(:, 1), '-o', 'DisplayName', 'Heun')
hold off
xlabel('t')
ylabel('x_1')
set(get(gca,'ylabel'),'rotation', 0)
legend show

figure(2)
plot(tODE, xODE(:, 2), '-o', 'DisplayName', 'ode45')
title('Trajektoria x_2(t)')
hold on
plot(t, x(:, 2), '-o', 'DisplayName', 'Heun')
hold off
xlabel('t')
ylabel('x_2')
set(get(gca,'ylabel'),'rotation', 0)
legend show

figure(3)
plot(xODE(:, 1), xODE(:, 2), '-o', 'DisplayName', 'ode45')
title('Trajektoria x_2(x_1)')
hold on
plot(x(:, 1), x(:, 2), '-o', 'DisplayName', 'Heun')
hold off
xlabel('x_1')
ylabel('x_2')
set(get(gca,'ylabel'),'rotation', 0)
legend show

deltas(end, :) = []

```

```
figure(4)
plot(t(1:end-1), deltas(:, 1), 'DisplayName', '\delta_x_1 (t)')
title('Zale no   estymaty b  du od czasu')
hold on
plot(t(1:end-1), deltas(:, 2), 'DisplayName', '\delta_x_2 (t)')
hold off
xlabel('t')
ylabel('\delta')
legend show

length(H)

figure(5)
plot(t, H, 'DisplayName', 'h(t)')
title('Zale no   d ugo ci kroku od czasu')
xlabel('t')
ylabel('h')
legend show
```