Paweł Maśluch

[MNUM] Metody Numeryczne

Semestr 2021L

Projekt 4

Sprawozdanie

Treść.

Ruch punktu jest opisany równaniami:

$$x_1' = x_2 + x_1 \cdot (0{,}9 - x_1^2 - x_2^2)$$

$$x_2' = -x_1 + x_2 \cdot (0{,}9 - x_1^2 - x_2^2)$$

Należy obliczyć przebieg trajektorii ruchu na przedziale [0, 20] dla następujących warunków początkowych:

a) $x_1(0)=10$ $x_2(0)=8$; (zestaw nr 1)
b) $x_1(0)=0$ $x_2(0)=9$; (zestaw nr 2)
c) $x_1(0)=8$ $x_2(0)=0$; (zestaw nr 3)
d) $x_1(0)=0{,}001$ $x_2(0)=0{,}001$ (zestaw nr 4).

Do rozwiązania zadania należy użyć zaimplementowanych przez siebie w języku MATLAB (w formie solwerów) metod:

1. Rungego–Kutty czwartego rzędu (RK4) ze stałym krokiem. Proszę przy tym wykonać tyle prób (kilka – kilkanaście), ile będzie potrzebnych do znalezienia takiego kroku, którego zmniejszanie nie wpływa znacząco na rozwiązanie, podczas gdy zwiększanie – już wpływa;

2. Wielokrokowej predyktor–korektor Adamsa czwartego rzędu ze stałym krokiem, który należy dobrać w sposób podany dla metody z punktu 1;

3. Rungego–Kutty czwartego rzędu (RK4) ze zmiennym krokiem.

W każdym kroku należy szacować błąd aproksymacji.

Sprawozdanie (w formacie PDF) powinno zawierać:
a) krótki opis zastosowanych algorytmów;

b) dla metod stałokrokowych RK4 i predyktor–korektor Adamsa komentarze i wnioski dotyczące doboru długości kroku, zilustrowane wykresami rozwiązań otrzymanych przy wybranym i przy zbyt dużym kroku;

c) porównanie rozwiązań otrzymanych wszystkimi metodami z rozwiązaniami wyznaczonymi poleceniem ode45 programu MATLAB; przydadzą się rysunki (kilku) funkcji jednej zmiennej (czasu) oraz trajektorii w przestrzeni fazowej (x_1, x_2) ;

d) odpowiedź na pytanie, z powołaniem na wyniki eksperymentów, która metoda jest lepsza i dlaczego (proszę wziąć, w szczególności, pod uwagę liczbę iteracji/kroków, czas obliczeń);

e) opis sposobu szacowania błędu dla metody RK4 ze zmiennym krokiem i znajdowania nowego rozwiązania dla każdej z metod;

f) wydruk dobrze skomentowanych programów z implementacją metod.

Rozwiązanie.

Na początku załączam listing funkcji pomocniczej dla polecenia *ode45* języka MATLAB (right_sides.m).

```
function dy = right_sides(x,y)
% Funkcja obliczajaca funkcje prawych stron
% dla ukladu rownan rozniczkowych z zadania
% (na potrzeby metody ode45).

% x - zmienna niezalezna
% y - wektor zmiennych zaleznych

dy = zeros(2,1);
dy(1) =  y(2) + y(1)*(0.9 - y(1)^2 - y(2)^2);
dy(2) = -y(1) + y(2)*(0.9 - y(1)^2 - y(2)^2);

end
```

1) Metoda Rungego-Kutty czwartego rzędu (RK4) ze stałym krokiem

Metoda Rungego-Kutty czwartego rzędu (RK4) ze stałym krokiem jest jedną z metod służących do rozwiązywania równań i układów równań różniczkowych zwyczajnych.

Wersja metody dla pojedynczego równania różniczkowego

Dane jest równanie różniczkowe zwyczajne $y'(x) = f(x, y)$,
dla $x \in [a, b]$, przy warunku początkowym $y(a) = y_a$.

Metodę tę opisuje wzór $y_{n+1} = y_n + \frac{h}{6} \cdot (k_1 + 2 \cdot k_2 + 2 \cdot k_3 + k_4)$ [1],
gdzie:

- $k_1 = f(x_n, y_n)$
- $k_2 = f(x_n + \frac{h}{2}, y_n + \frac{h}{2} \cdot k_1)$
- $k_3 = f(x_n + \frac{h}{2}, y_n + \frac{h}{2} \cdot k_2)$
- $k_4 = f(x_n + h, y_n + h \cdot k_3)$.

Wersja metody dla układu równań różniczkowych

Metodę tę można również zastosować do rozwiązywania
układów równań różniczkowych zwyczajnych.

Niech $x \in \mathrm{R}$ będzie zmienną niezależną
oraz niech $\boldsymbol{y} = [y_1 \quad y_2 \quad \cdots \quad y_m]^T$ będzie wektorem zmiennych zależnych.

Dany jest układ równań różniczkowych

$$\left[\frac{dy_1}{dx} \quad \frac{dy_2}{dx} \quad \cdots \quad \frac{dy_m}{dx}\right]^T = [f_1(x, \boldsymbol{y}) \quad f_2(x, \boldsymbol{y}) \quad \cdots \quad f_m(x, \boldsymbol{y})]^T,$$

gdzie f_i: $\mathrm{R}^{m+1} \to \mathrm{R}$ (dla $i = 1,2, \ldots, m$) oraz $x \in [a, b]$.

Przyjmując oznaczenia $\boldsymbol{y}'(x) \stackrel{\text{def}}{=} \left[\frac{dy_1}{dx} \quad \frac{dy_2}{dx} \quad \dots \quad \frac{dy_m}{dx}\right]^T$
oraz $\boldsymbol{f} \stackrel{\text{def}}{=} [f_1(x, \boldsymbol{y}) \quad f_2(x, \boldsymbol{y}) \quad \cdots \quad f_m(x, \boldsymbol{y})]^T$, otrzymujemy zapis wektorowy układu równań różniczkowych zwyczajnych: $\boldsymbol{y}'(x) = \boldsymbol{f}(x, \boldsymbol{y})$,
gdzie $x \in [a, b]$ oraz wektor warunków początkowych $\boldsymbol{y}(a) = \boldsymbol{y}_a$.

Przyjmijmy dodatkowe oznaczenia:

- $y_i^{(j)}$ - wartość i-tej zmiennej zależnej w j-tej iteracji metody
- $\boldsymbol{y}^{(j)} \stackrel{\text{def}}{=} \left[y_1^{(j)} \; y_2^{(j)} \; \dots \; y_m^{(j)}\right]^T$ - wektor wartości zmiennych zależnych w j-tej iteracji metody.

Wówczas naszą metodę (dla jednej iteracji) opisuje wzór
$y_i^{(n+1)} = y_i^{(n)} + \frac{h}{6} \cdot (k_{1,i} + 2 \cdot k_{2,i} + 2 \cdot k_{3,i} + k_{4,i})$ $(i = 1,2, \dots, m)$,
gdzie:

- $k_{1,i} = f_i(x_n, y_1^{(n)}, y_2^{(n)}, \dots, y_m^{(n)})$
- $k_{2,i} = f_i(x_n + \frac{h}{2}, \quad y_1^{(n)} + \frac{h}{2} \cdot k_{1,1}, \quad y_2^{(n)} + \frac{h}{2} \cdot k_{1,2}, \quad \dots, \quad y_m^{(n)} + \frac{h}{2} \cdot k_{1,m})$
- $k_{3,i} = f_i(x_n + \frac{h}{2}, \quad y_1^{(n)} + \frac{h}{2} \cdot k_{2,1}, \quad y_2^{(n)} + \frac{h}{2} \cdot k_{2,2}, \quad \dots, \quad y_m^{(n)} + \frac{h}{2} \cdot k_{2,m})$
- $k_{4,i} = f_i(x_n + h, \quad y_1^{(n)} + h \cdot k_{3,1}, \quad y_2^{(n)} + h \cdot k_{3,2}, \quad \dots, \quad y_m^{(n)} + h \cdot k_{3,m})$

Oszacowanie błędu pojedynczego kroku o długości h, dla i-tej zmiennej zależnej $(i = 1,2, \dots, m)$ w n-tej iteracji, obliczamy ze wzoru $\delta_i^{(n)} = \frac{2^p}{2^p - 1} \cdot (y_i^{(n)(2)} - y_i^{(n)(1)})$ [1], gdzie:

- $y_i^{(n)(2)}$ – wartość i-tej zmiennej zależnej w n-tej iteracji, obliczona poprzez 2 iteracje pomocnicze metody z krokiem $\frac{h}{2}$ (startując z punktu x_{n-1} oraz $\boldsymbol{y}^{(n-1)}$)

- $y_i^{(n)(1)}$ – wartość i-tej zmiennej zależnej w n-tej iteracji, obliczona poprzez 1 iterację metody z krokiem h (startując z punktu x_{n-1} oraz $\boldsymbol{y}^{(n-1)}$)

- p – rząd metody (w naszym przypadku $p = 4$).

Zapis wektorowy dla oszacowania błędu pojedynczego kroku o długości h (w n-tej iteracji) wygląda następująco: $\boldsymbol{\delta}^{(n)} = \frac{2^p}{2^p-1} \cdot (\boldsymbol{y}^{(n)(2)} - \boldsymbol{y}^{(n)(1)})$,
gdzie:

- $\boldsymbol{\delta}^{(n)} = \left[\delta_1^{(n)} \;\; \delta_2^{(n)} \;\; \dots \;\; \delta_m^{(n)}\right]^T$,
- $\boldsymbol{y}^{(n)(1)} = \left[y_1^{(n)(1)}, y_2^{(n)(1)}, \dots, y_m^{(n)(1)}\right]^T$,
- $\boldsymbol{y}^{(n)(2)} = \left[y_1^{(n)(2)}, y_2^{(n)(2)}, \dots, y_m^{(n)(2)}\right]^T$.

Listing solwera (RK4_const_h.m):

```
function [y, delta_h] = RK4_const_h(f, m, a, b, h, y0)
% Metoda Rungego-Kutty RK4 ze stalym krokiem,
% sluzaca do rozwiazywania rownan
% i ukladow rownan rozniczkowych zwyczajnych.

% y - wartosci zmiennych zaleznych,
%     dla okreslonego przedzialu wartosci zmiennej niezaleznej
% delta_h - oszacowania bledu pojedynczego kroku o długości h
%           dla kazdej zmiennej zaleznej
% f - funkcje prawych stron (w tablicy komorkowej)
% m - liczba rownan (i zmiennych zaleznych zarazem)
% [a, b] - przedzial zmiennej niezaleznej,
%          dla ktorego poszukujemy rozwiazania
% y0 - wektor warunkow poczatkowych
% h - krok calkowania


% Rzad metody
p = 4;


% Wartosci zmiennej niezaleznej,
% dla ktorych obliczymy wartosci zmiennych zaleznych
x = a:h:b;


% Liczba krokow calkowania
n = length(x);
```

```
% Wartosci pomocnicze k, charakterystyczne dla naszej metody;
% wartosci te wyliczamy w ramach konkretnej iteracji
k = zeros(4, m);


% y - wartosci zmiennych zaleznych:
% wiersze - wartosci wszystkich zmiennych zaleznych
%           dla danej iteracji,
% kolumny - wartosci okreslonych zmiennych zaleznych
%           we wszystkich iteracjach;
% Inicjacja warunkami poczatkowymi
y(1, :) = y0;


% Oszacowania bledu pojedynczego kroku o długości h
% dla kazdej zmiennej zaleznej
delta_h(1,:) = zeros(m, 1);


% (Prawie) kazda iteracja
for nr=2:n

    % Wektor zmiennych zaleznych, obliczanych poprzez 2 iteracje
    % z krokiem h/2
    y_tmp = y(nr-1,:);

    % ---------------------------------

    % 2 iteracje z krokiem h/2
    % (celem oszacowania błędu pojedynczego kroku o długości h)
    for j=1:2

        % Wartosci k1 dla kazdej zmiennej zaleznej
        for i=1:m
            k(1,i) = feval( f{i}, x(nr-1), y_tmp );
        end


        % Wartosci k2 dla kazdej zmiennej zaleznej
        for i=1:m
            k(2,i) = feval( f{i}, x(nr-1) + j*h/4,   y_tmp + j*h/4 * k(1,:)
);
        end


        % Wartosci k3 dla kazdej zmiennej zaleznej
        for i=1:m
            k(3,i) = feval( f{i},   x(nr-1) + j*h/4,   y_tmp + j*h/4 *
k(2,:) );
        end


        % Wartosci k4 dla kazdej zmiennej zaleznej
        for i=1:m
            k(4,i) = feval( f{i},   x(nr-1) + j*h/2,   y_tmp + j*h/2 *
k(3,:) );
        end
```

```
        % Obliczenie wartosci zmiennych zaleznych w aktualnym kroku
        y_tmp = y_tmp + h/12 * ( k(1,:) + 2*k(2,:) + 2*k(3,:) + k(4,:) );
    end

    % ---------------------------------------


    % 1 iteracja z krokiem h

    % Wartosci k1 dla kazdej zmiennej zaleznej
    for i=1:m
        k(1,i) = feval( f{i}, x(nr-1), y(nr-1,:) );
    end


    % Wartosci k2 dla kazdej zmiennej zaleznej
    for i=1:m
        k(2,i) = feval( f{i}, x(nr-1) + h/2,   y(nr-1,:) + h/2 * k(1,:) );
    end


    % Wartosci k3 dla kazdej zmiennej zaleznej
    for i=1:m
        k(3,i) = feval( f{i},   x(nr-1) + h/2,   y(nr-1,:) + h/2 * k(2,:)
);
    end


    % Wartosci k4 dla kazdej zmiennej zaleznej
    for i=1:m
        k(4,i) = feval( f{i},   x(nr-1) + h,   y(nr-1,:) + h * k(3,:) );
    end


    % Obliczenie wartosci zmiennych zaleznych w aktualnym kroku
    y(nr,:) = y(nr-1,:) + h/6 * ( k(1,:) + 2*k(2,:) + 2*k(3,:) + k(4,:) );

    % --------------------------


    % Obliczenie oszacowania bledu pojedynczego kroku o dlugosci h
    delta_h(nr-1, :) = (2^p / (2^p - 1)) * ( y_tmp - y(nr,:) );

end


end
```

Listing skryptu wywołującego (Zad1.m):

```
% Zadanie 1.
% Metoda Rungego-Kutty RK4 ze stalym krokiem.


clear;
clc;


% Liczba rownan rozniczkowych (i zmiennych zaleznych zarazem)
m = 2;


% f - tablica komorkowa, zawierajaca uchwyty
% do prawych stron rownan rozniczkowych
f = cell(m,1);


% Wstawienie (do tablicy f) uchwytow
% do prawych stron rownan rozniczkowych
f{1} = @(x, y)( y(2) + y(1)*(0.9 - y(1)^2 - y(2)^2) );
f{2} = @(x, y)( -y(1) + y(2)*(0.9 - y(1)^2 - y(2)^2) );


% Przedzial [a,b] zmiennej niezaleznej,
% dla ktorego bedziemy calkowali
% uklad rownan rozniczkowych
a = 0;
b = 20;


% Warunki poczatkowe (kolejne wiersze)
y0 = [10    8;
       0    9;
       8    0;
    1e-3 1e-3];


% Kroki calkowania (odpowiadajace kolejnym warunkom poczatkowym)
h = [0.014 0.025 0.02 0.2];


% Zbyt duze kroki calkowania (odpowiadajace kolejnym warunkom poczatkowym)
h_big = [0.015 0.03 0.025 0.3];


% Liczba warunkow poczatkowych
L = 4;


% Numery zestawow warunkow poczatkowych
nr = 1:L;
```

```
% Czasy dzialania solwerow
T = zeros(L,1);


% Liczby iteracji
iter = zeros(L,1);


% Licznik wykresow
q = 1;


% Wyznaczenie rozwiazan
% dla roznych warunkow poczatkowych
for i=1:L

    % Obliczenie rozwiazan i bledow oszacowan dla "dobrego" kroku h
    tic;
    [y, delta_h] = RK4_const_h(f, m, a, b, h(i), y0(i,:));
    T(i) = toc;


    % Wykres fazowy dla "dobrego" kroku h
    figure(q);
    hold on;
    plot( y(:,1), y(:,2) );
    grid on;
    title( ['Rozwiazanie w przestrzeni fazowej - warunki poczatkowe nr ',
num2str(i), ', krok h=', num2str( h(i) )] );
    xlabel('x1');
    ylabel('x2');
    hold off;
    q = q+1;


    % Obliczenie rozwiazan i bledow oszacowan (zbyt duzy krok h)
    [y_big, delta_h_big] = RK4_const_h(f, m, a, b, h_big(i), y0(i,:));


    % Wykres fazowy dla zbyt "duzego" kroku h
    figure(q);
    hold on;
    plot( y_big(:,1), y_big(:,2) );
    grid on;
    title( ['Rozwiazanie w przestrzeni fazowej - warunki poczatkowe nr ',
num2str(i), ', zbyt "duzy" krok h=', num2str( h_big(i) )] );
    xlabel('x1');
    ylabel('x2');
    hold off;
    q = q+1;


    % Wyznaczenie wektora (osi poziomej) do wykresow bledow aproksymacji
    x = a:h(i):b;


    % Wyznaczenie liczby iteracji
    iter(i) = length(x);
```

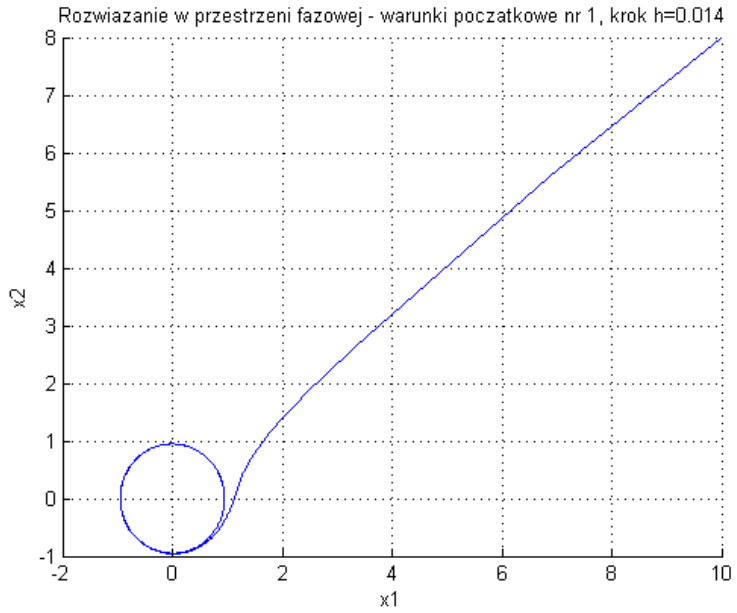
```
    % Wykres modulow bledow aproksymacji - zmienna zalezna x1 ("dobry" krok
h)
    figure(q);
    hold on;
    plot( x([2:length(x)]), abs( delta_h(:,1) ) );
    grid on;
    title( ['Oszacowanie modulow bledow aproksymacji - zmienna zalezna x1 -
warunki poczatkowe nr ', num2str(i), ', "dobry "krok h=', num2str( h(i) )]
);
    xlabel('t');
    ylabel('Oszacowanie bledu aproksymacji');
    hold off;
    q = q+1;


    % Wykres bledu aproksymacji - zmienna zalezna x2 ("dobry" krok h)
    figure(q);
    hold on;
    plot( x([2:length(x)]), abs( delta_h(:,2) ) );
    grid on;
    title( ['Oszacowanie modulow bledow aproksymacji - zmienna zalezna x2 -
warunki poczatkowe nr ', num2str(i), ', "dobry" krok h=', num2str( h(i) )]
);
    xlabel('t');
    ylabel('Oszacowanie bledu aproksymacji');
    hold off;
    q = q+1;


    % Wykres normy maksimum bledu aproksymacji ("dobry" krok h)
    delta_h_aggr = zeros( length(delta_h), 1 );
    for c=1:length(delta_h)
        delta_h_aggr(c) = norm( delta_h(c, :), Inf );
    end

    figure(q);
    hold on;
    plot( x([2:length(x)]), delta_h_aggr );
    grid on;
    title( ['Norma maksimum bledu aproksymacji - warunki poczatkowe nr ',
num2str(i), ', "dobry" krok h=', num2str( h(i) )] );
    xlabel('t');
    ylabel('Oszacowanie bledu aproksymacji');
    hold off;
    q = q+1;


    % Rozwiazanie przy pomocy polecenia ode45
    [x_ode, y_ode] = ode45(@right_sides,[a b], y0(i,:));


    % Wykresy porownawcze

    % Wykres fazowy
    figure(q);
    hold on;
    plot( y(:,1), y(:,2), y_ode(:,1), y_ode(:,2) );
    grid on;
```

```
    title( ['Porownanie rozwiazan - przestrzen fazowa - warunki poczatkowe
nr ', num2str(i), ', "dobry" krok h=', num2str( h(i) )] );
    xlabel('x1');
    ylabel('x2');
    legend('RK4 const h', 'ode45', 'Location', 'North');
    hold off;
    q = q+1;


    % Zmienna zalezna x1
    figure(q);
    hold on;
    plot( x, y(:,1), x_ode, y_ode(:,1) );
    grid on;
    title( ['Porownanie rozwiazan - zmienna zalezna x1 - warunki poczatkowe
nr ', num2str(i), ', "dobry" krok h=', num2str( h(i) )] );
    xlabel('t');
    ylabel('x1');
    legend('RK4 const h', 'ode45');
    hold off;
    q = q+1;


    % Zmienna zalezna x2
    figure(q);
    hold on;
    plot( x, y(:,2), x_ode, y_ode(:,2) );
    grid on;
    title( ['Porownanie rozwiazan - zmienna zalezna x2 - warunki poczatkowe
nr ', num2str(i), ', "dobry" krok h=', num2str( h(i) )] );
    xlabel('t');
    ylabel('x2');
    legend('RK4 const h', 'ode45');
    hold off;
    q = q+1;


end
```

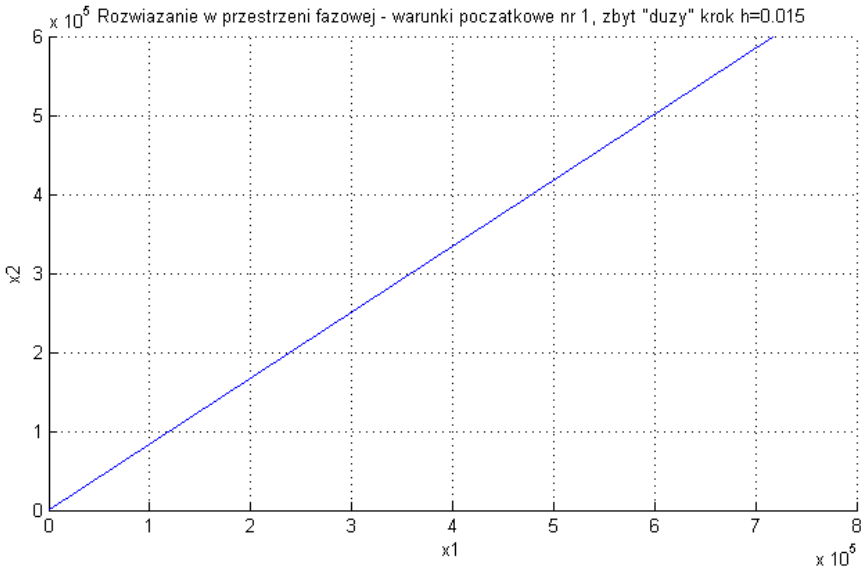
Wykresy:

Warunki początkowe a):

Dobrałem krok $h = 0.014$.
Jeśli zwiększymy tę wartość do $h = 0.015$, to zmienia się istotnie postać rozwiązania - zmienne zależne osiągają bardzo duże wartości oraz zmienia się przebieg rozwiązania (patrz wykresy 1. oraz 2.).

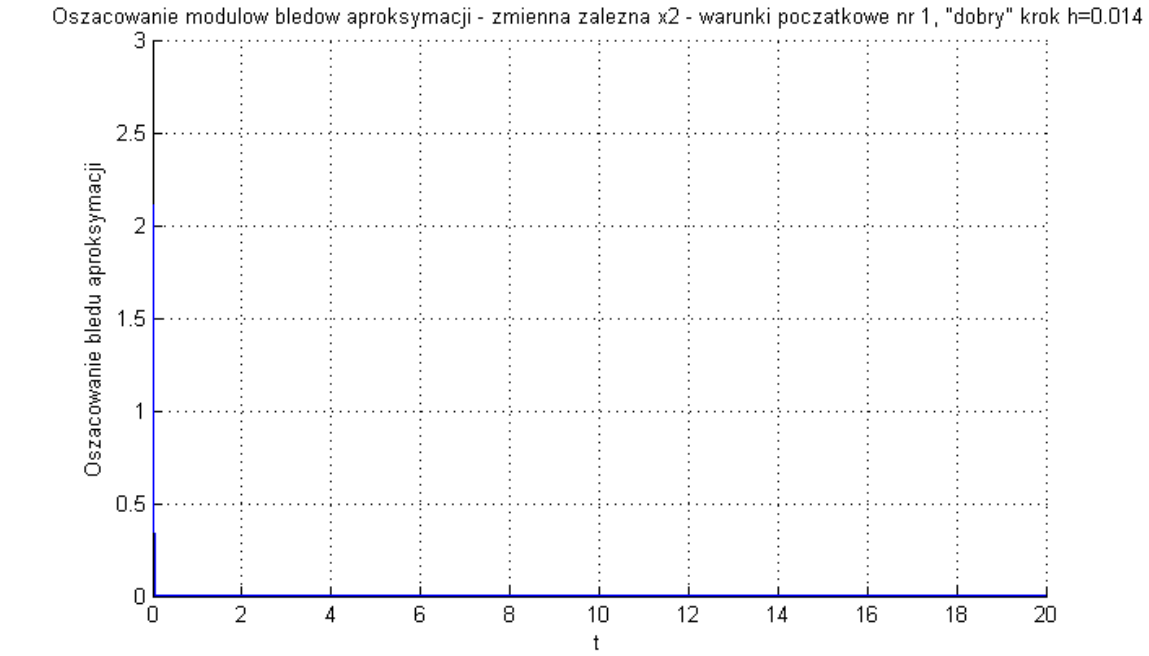


Wykres 1.

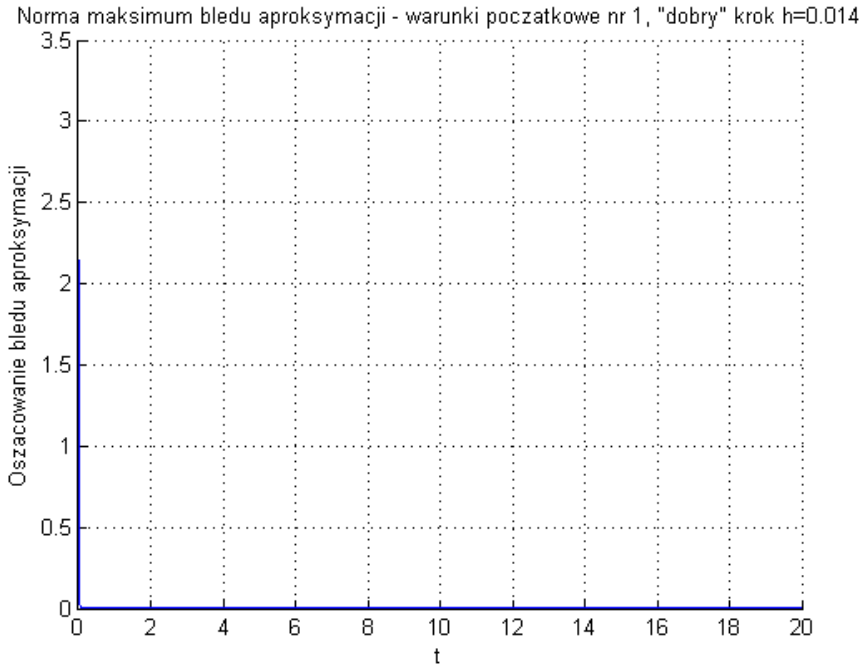


Wykres 2.

Wykres 3.

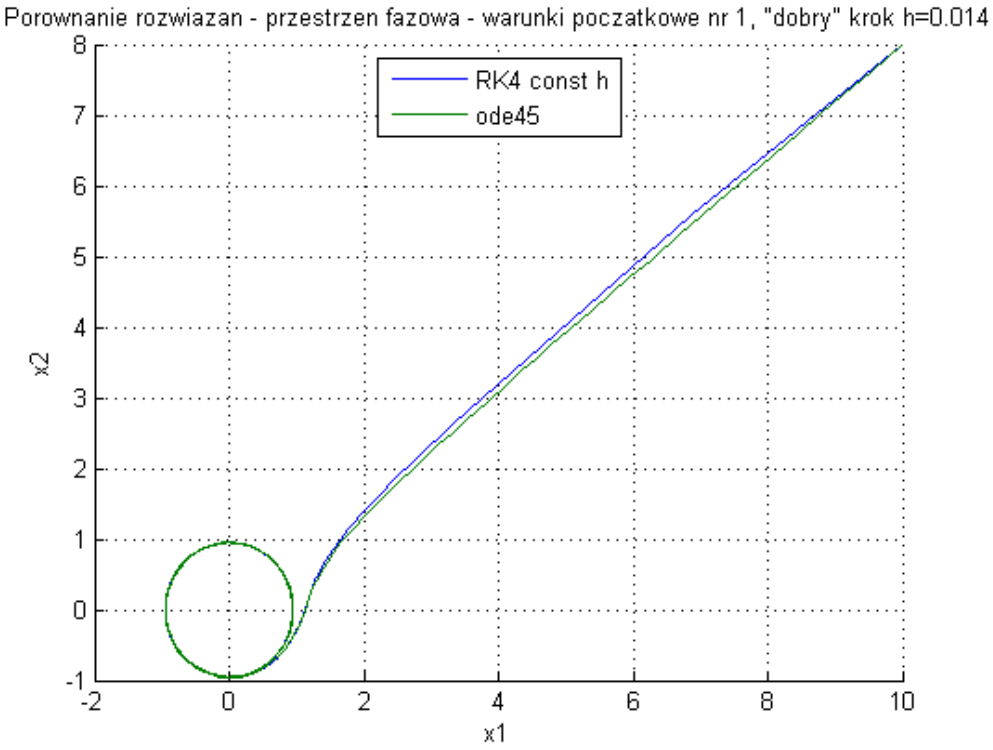


Wykres 4.

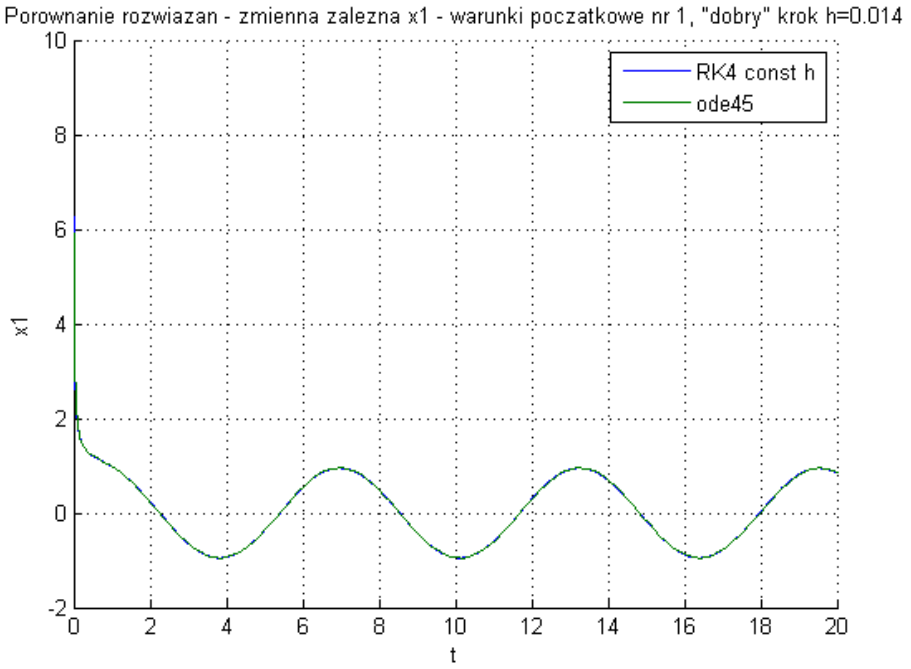


Wykres 5.

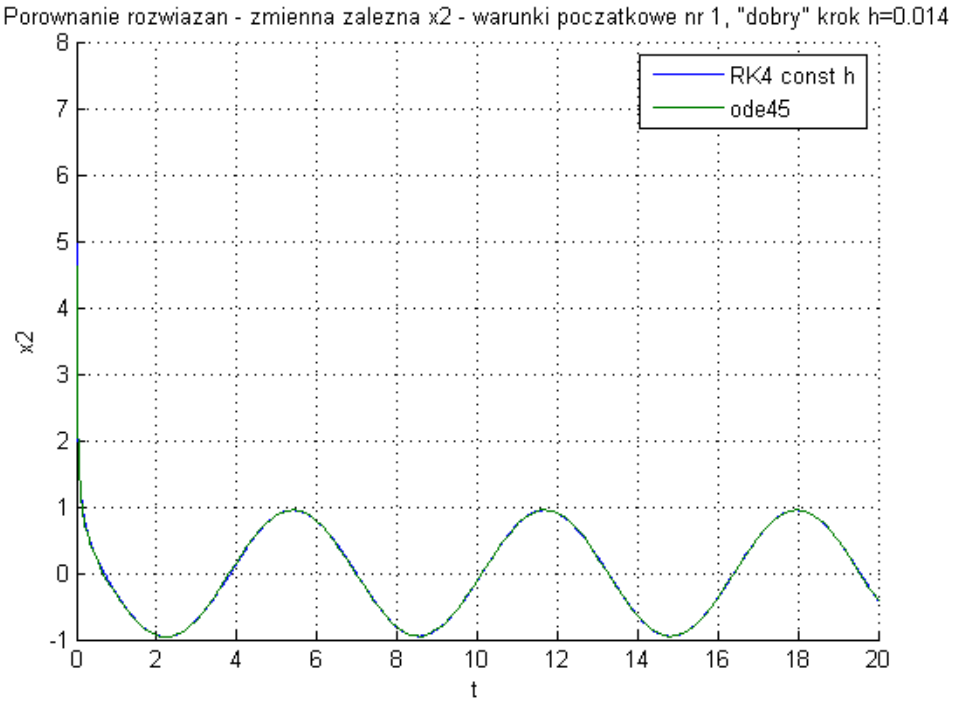
Zauważmy, że błędy aproksymacji bardzo szybko ustalają się na niskim poziomie (patrz wykresy 3., 4., 5.).



Wykres 6.



Wykres 7.

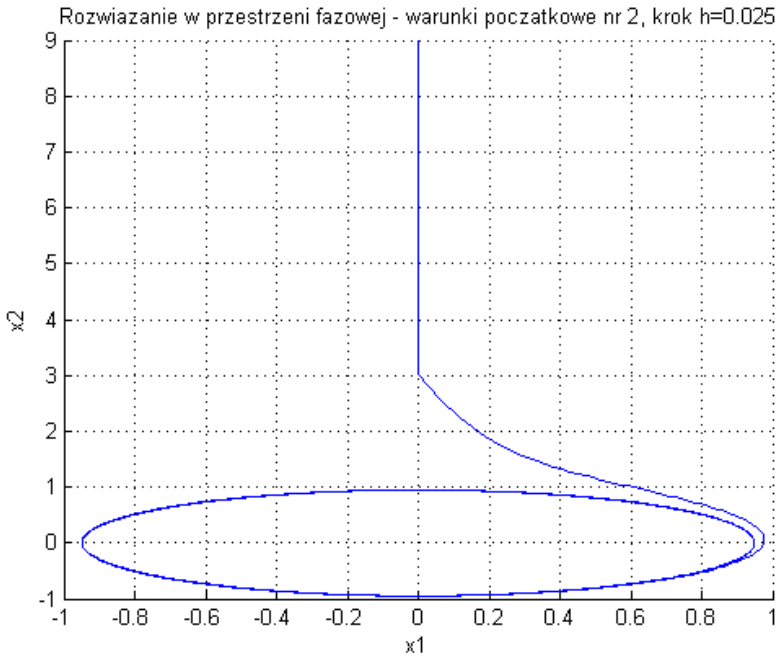


Wykres 8.

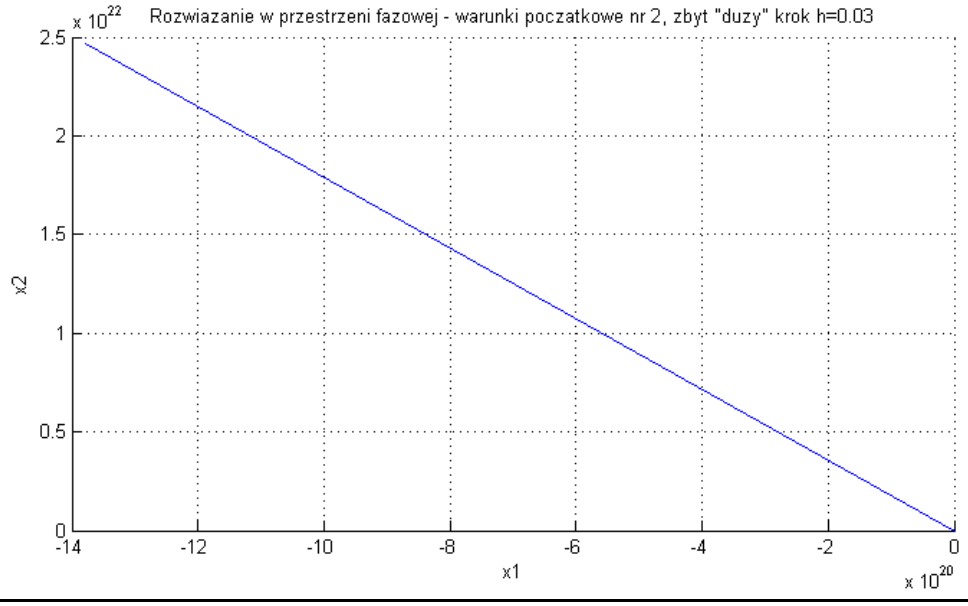
Uzyskane rozwiązanie (metodą RK4 ze stałym krokiem) dość dobrze (choć różnice są widoczne) oddaje rozwiązanie uzyskane dzięki poleceniu ode45 języka MATLAB (wykresy 6., 7., 8.).

Warunki początkowe b):

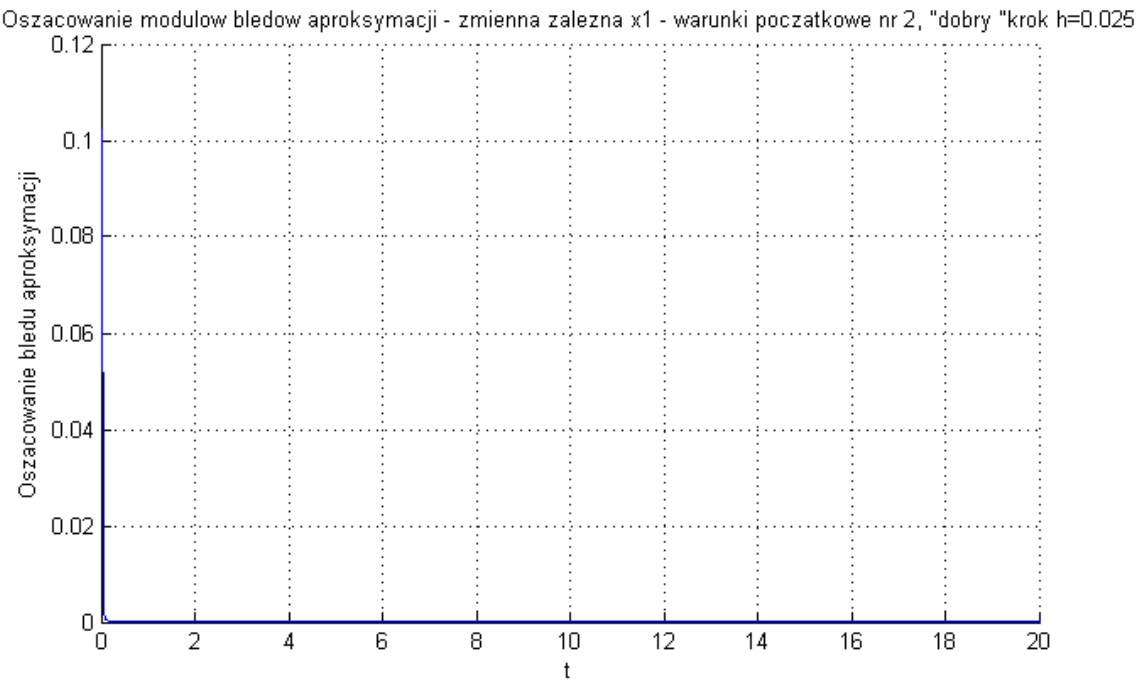
Dobrałem krok $h = 0.025$.
Jeśli zwiększymy tę wartość do $h = 0.03$, to zmienia się istotnie postać rozwiązania - zmienne zależne osiągają bardzo duże wartości oraz zmienia się przebieg rozwiązania (patrz wykresy 9. oraz 10.).



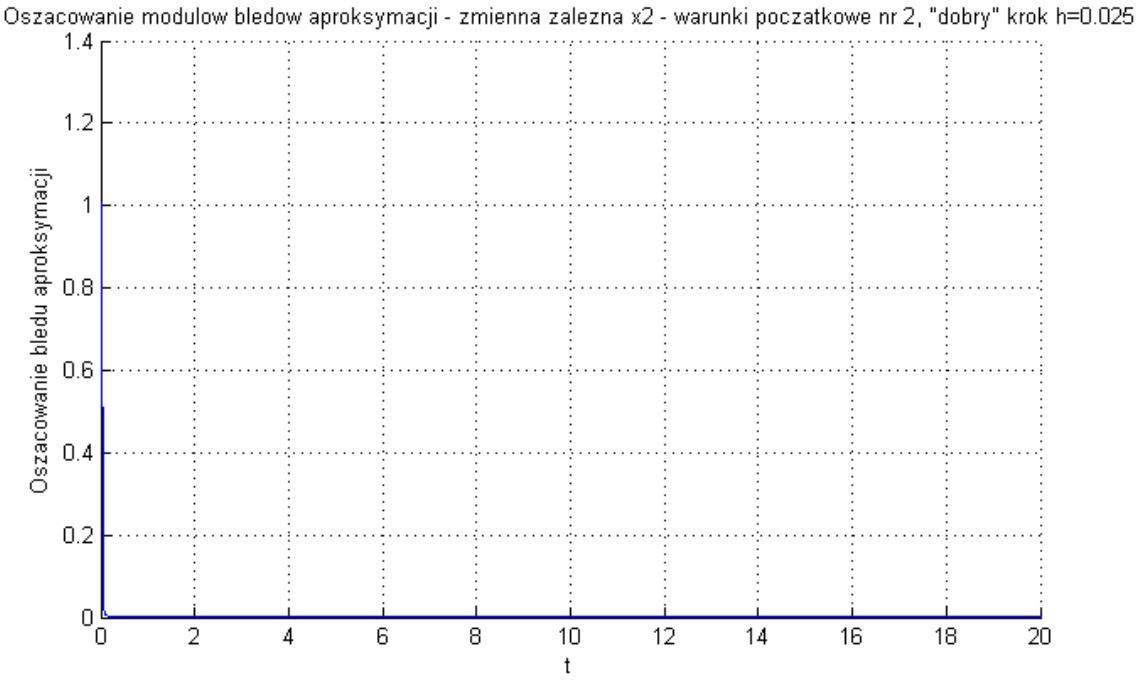
Wykres 9.



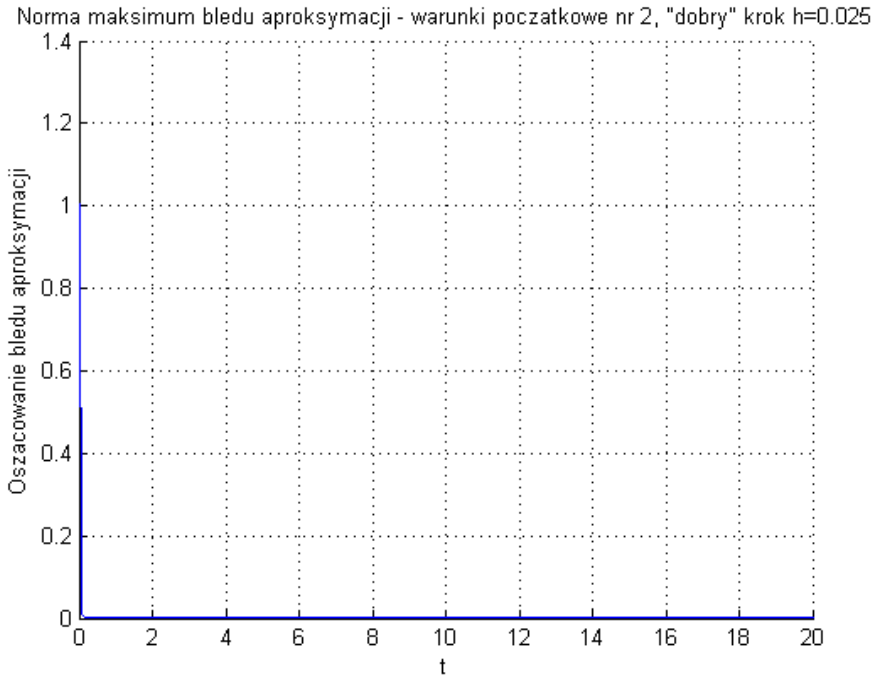
Wykres 10.



Wykres 11.

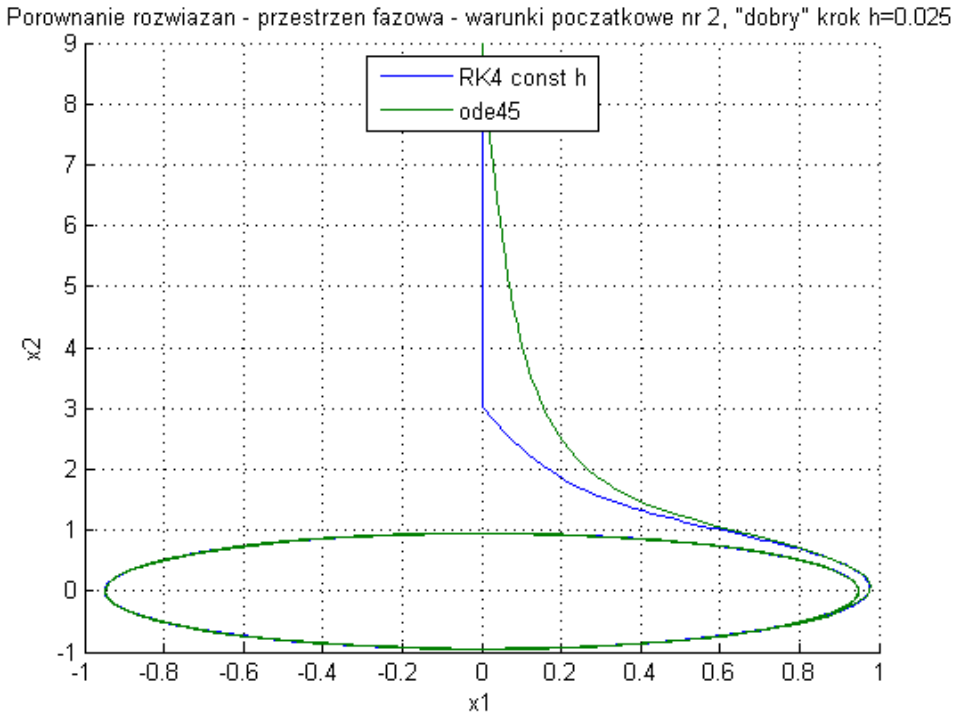


Wykres 12.

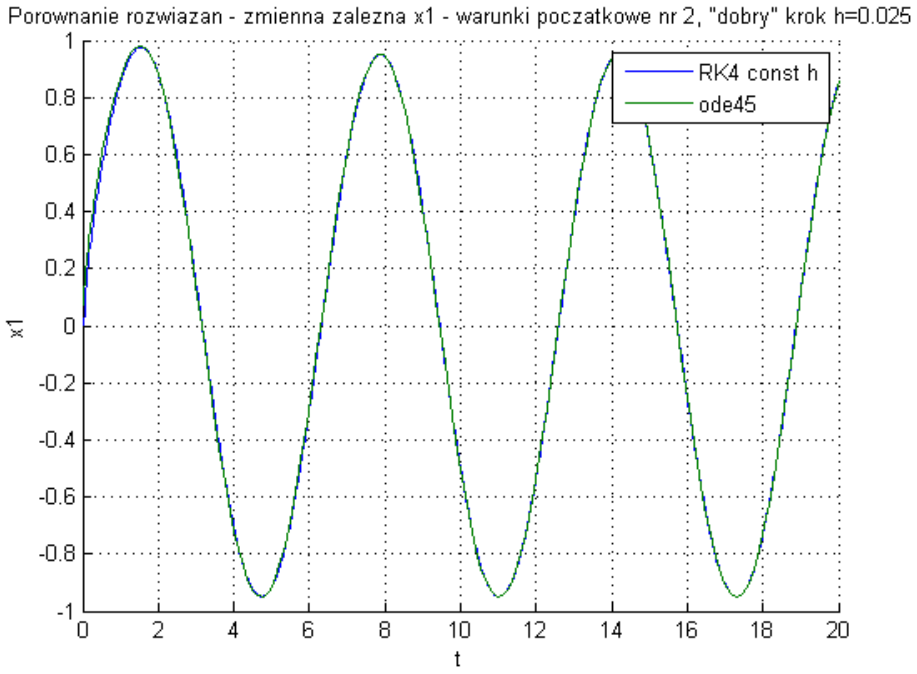


Wykres 13.

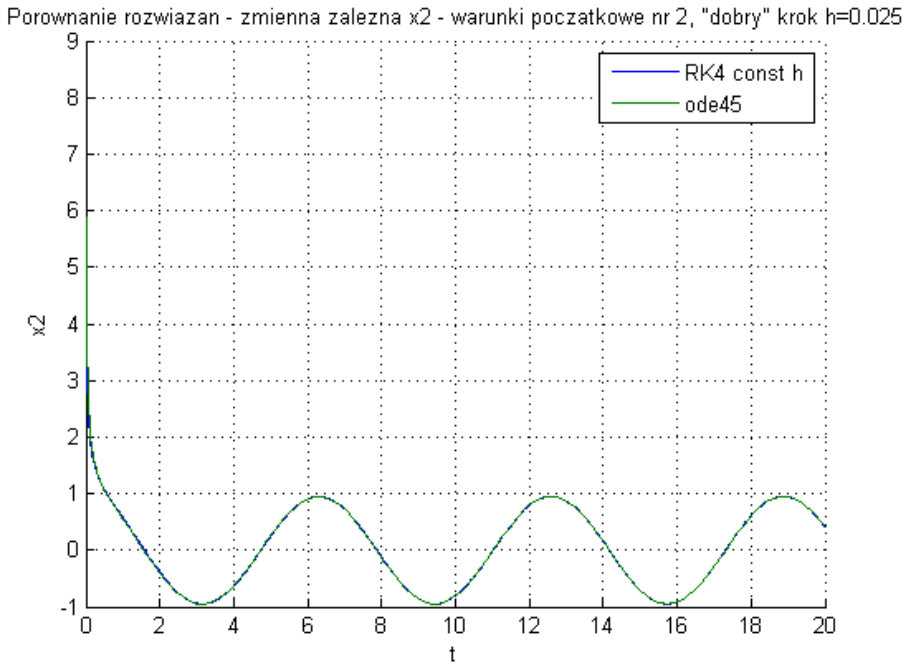
Zauważmy, że błędy aproksymacji bardzo szybko ustalają się na niskim poziomie (patrz wykresy 11., 12., 13.).



Wykres 14.



Wykres 15.

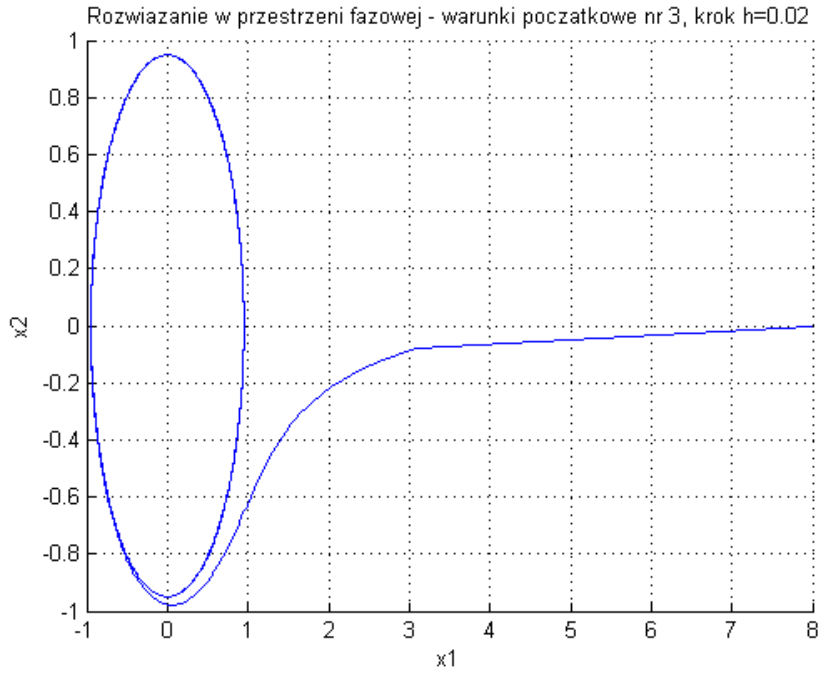


Wykres 16.

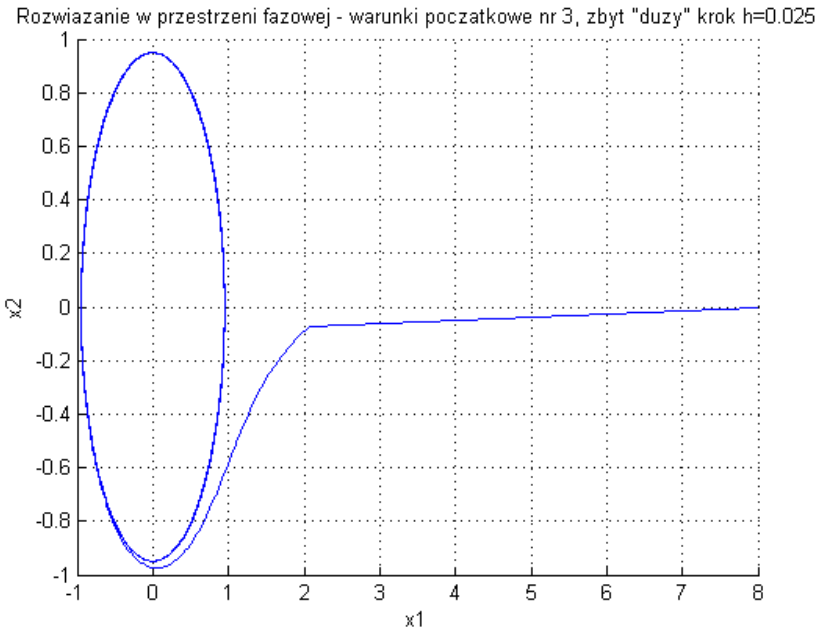
Uzyskane rozwiązanie (metodą RK4 ze stałym krokiem) dobrze oddaje rozwiązanie uzyskane dzięki poleceniu ode45 (języka MATLAB) w niektórych fragmentach, a w niektórych już nie (wykresy 14., 15., 16.).

Warunki początkowe c):

Dobrałem krok $h = 0.02$.
Jeśli zwiększymy tę wartość do $h = 0.025$, to rozwiązanie przestaje być „gładkie” (patrz wykresy 17. oraz 18.).

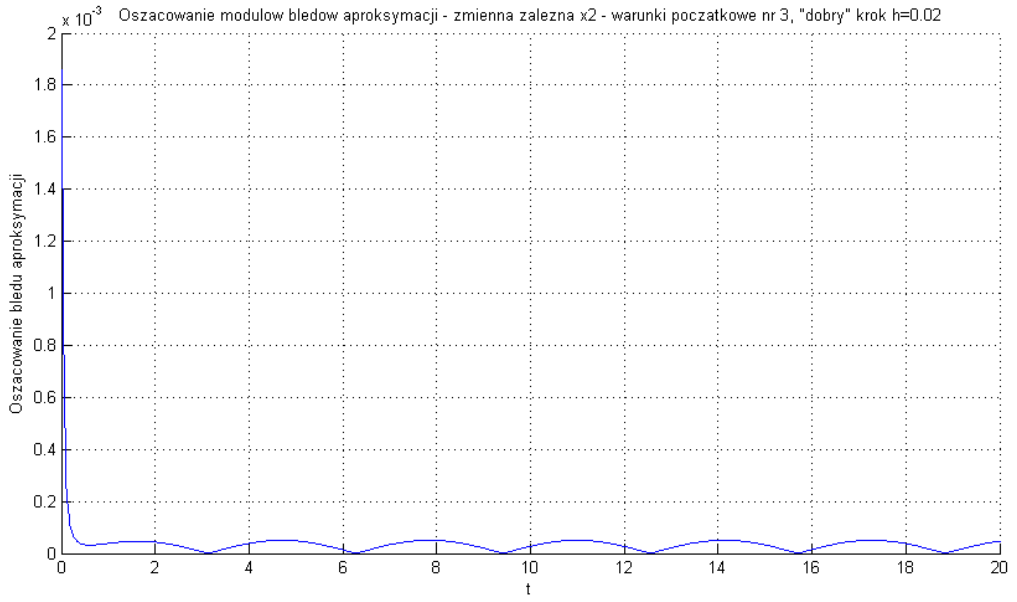


Wykres 17.

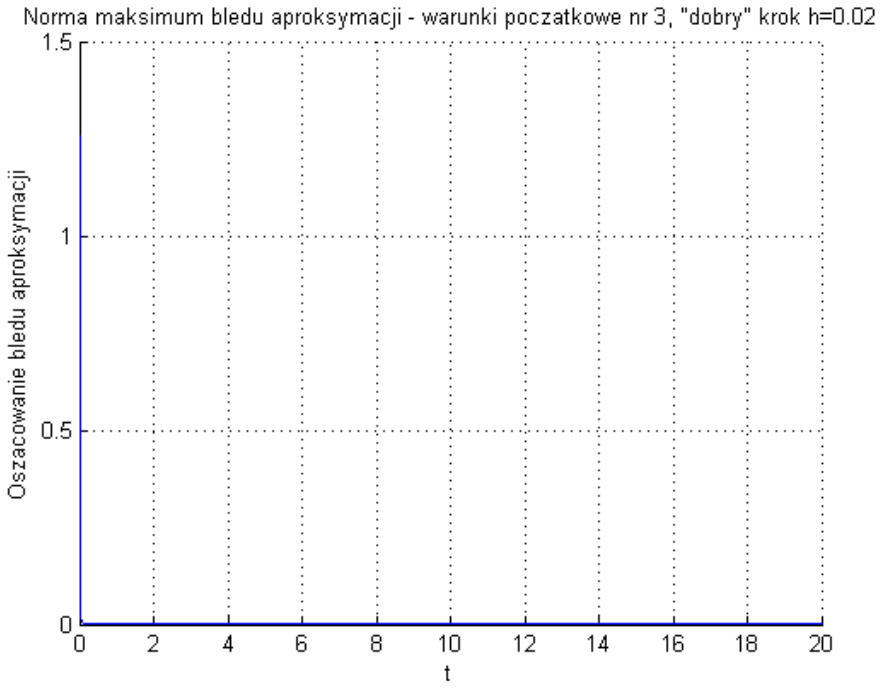


Wykres 18.

Wykres 19.

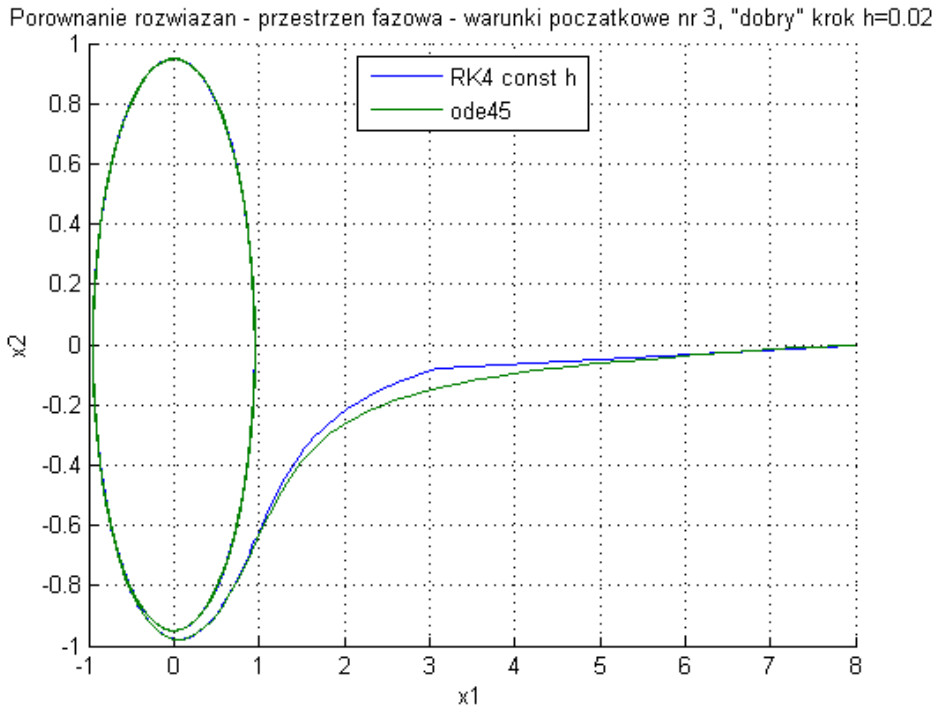


Wykres 20.

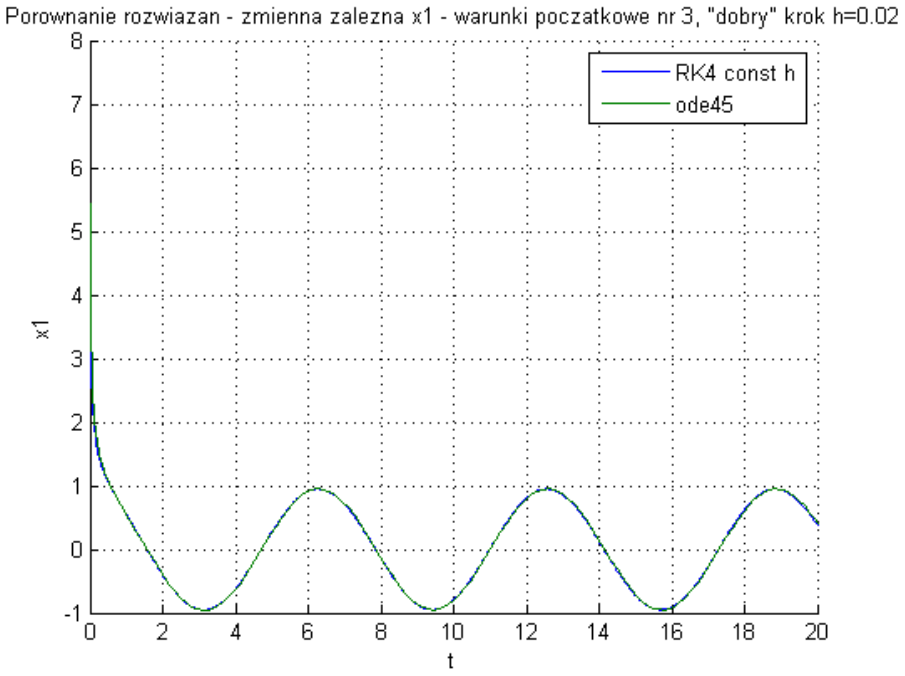


Wykres 21.

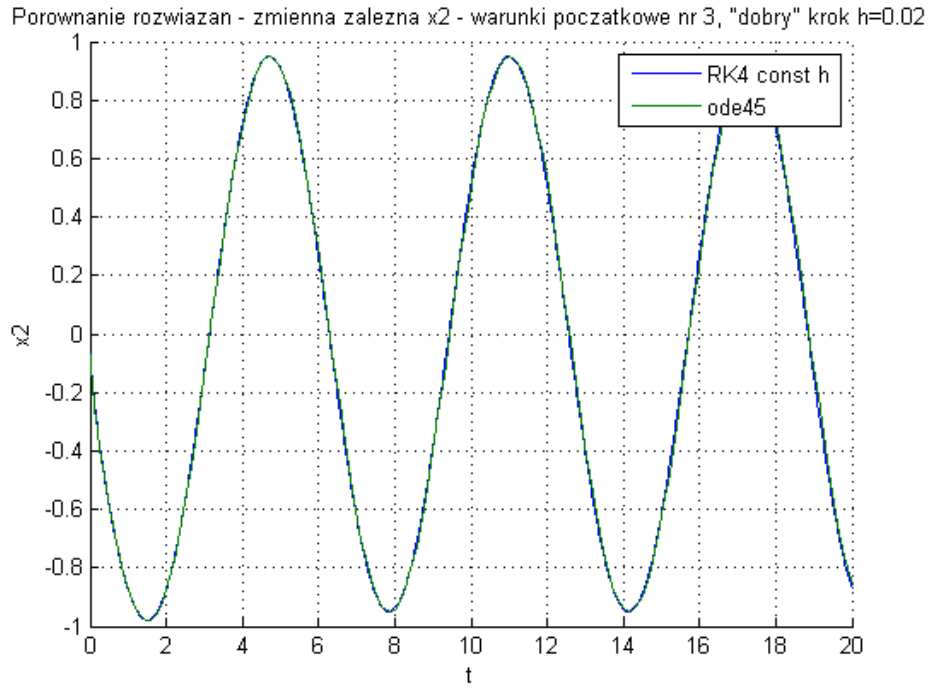
Zauważmy, że błędy aproksymacji bardzo szybko ustalają się/oscylują na niskim poziomie (patrz wykresy 19., 20., 21.).



Wykres 22.



Wykres 23.

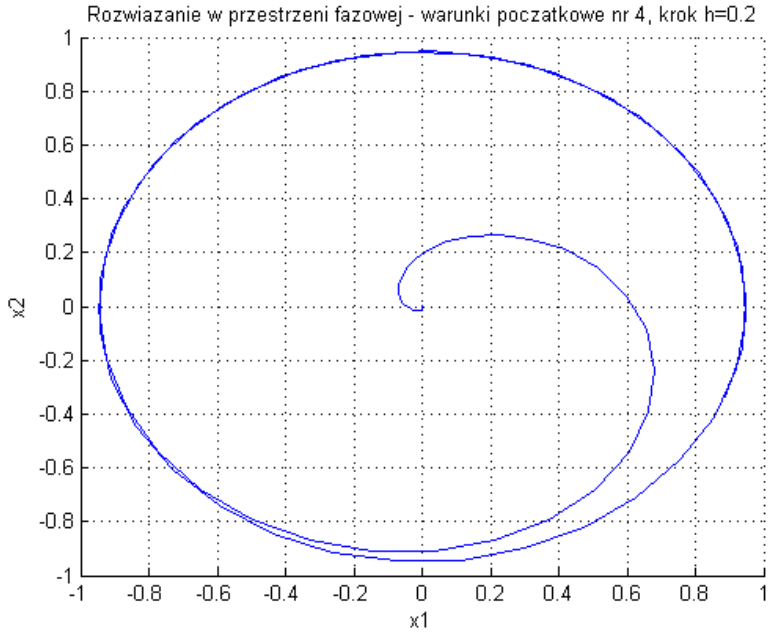


Wykres 24.

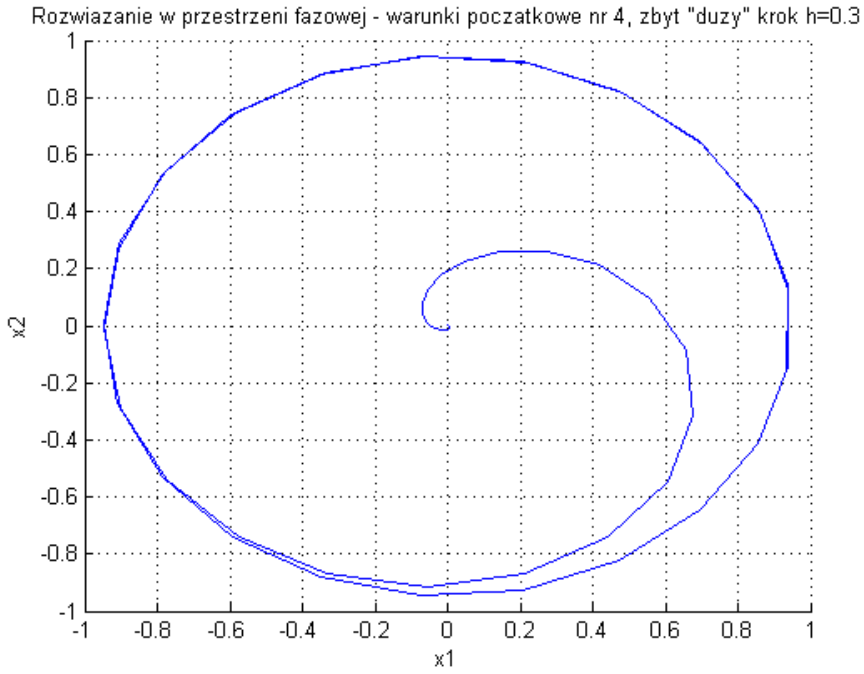
Uzyskane rozwiązanie (metodą RK4 ze stałym krokiem) dobrze oddaje rozwiązanie uzyskane dzięki poleceniu ode45 (języka MATLAB) w niektórych fragmentach, a w niektórych już nie (wykresy 22., 23., 24.).

Warunki początkowe d):

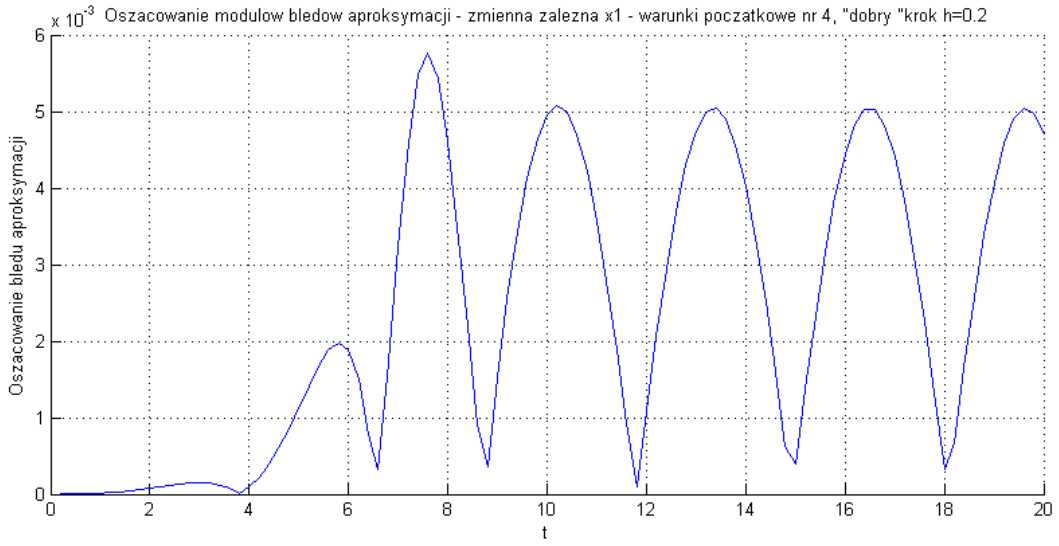
Dobrałem krok $h = 0.2$.
Jeśli zwiększymy tę wartość do $h = 0.3$, to rozwiązanie przestaje być „gładkie” (patrz wykresy 25. oraz 26.).



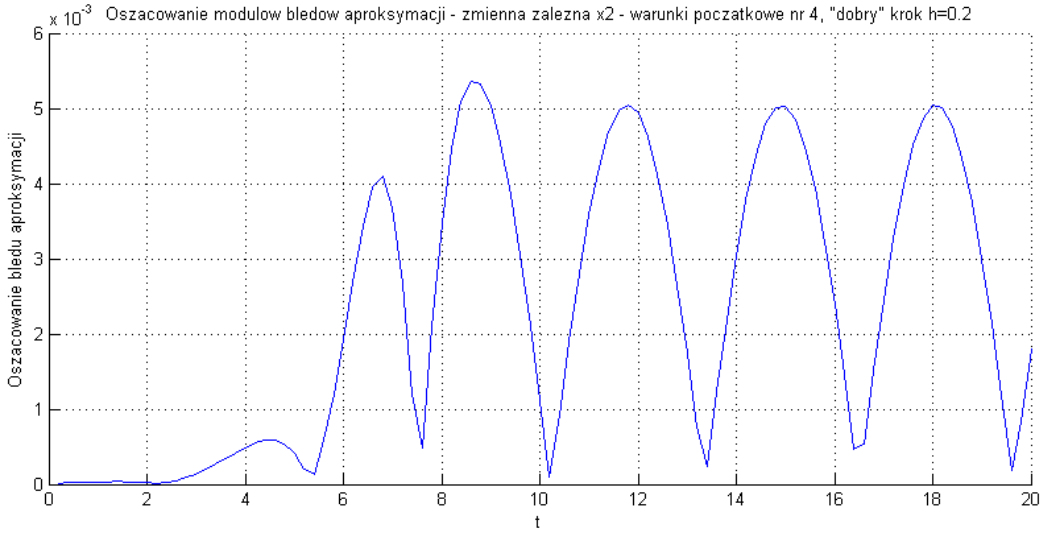
Wykres 25.



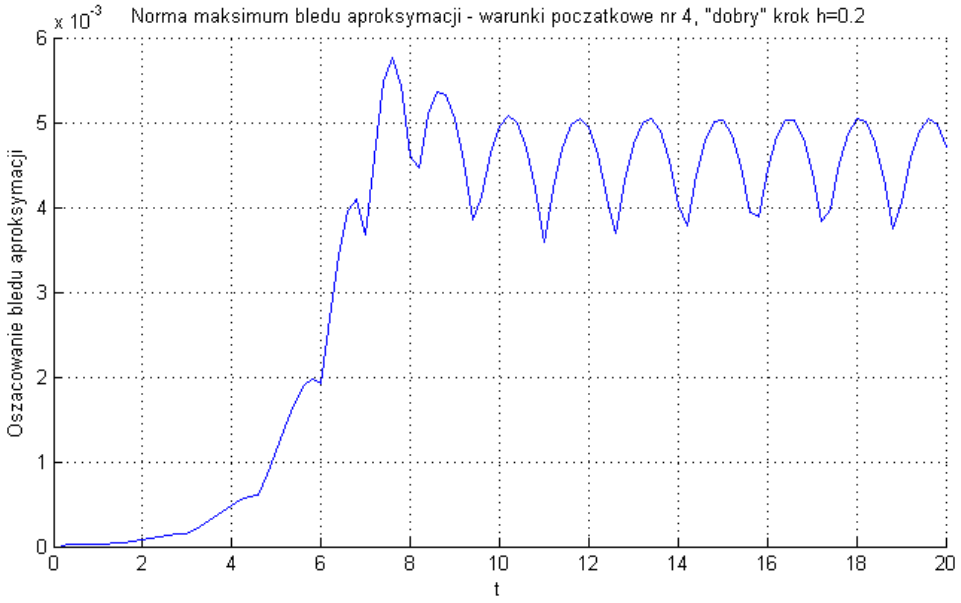
Wykres 26.



Wykres 27.

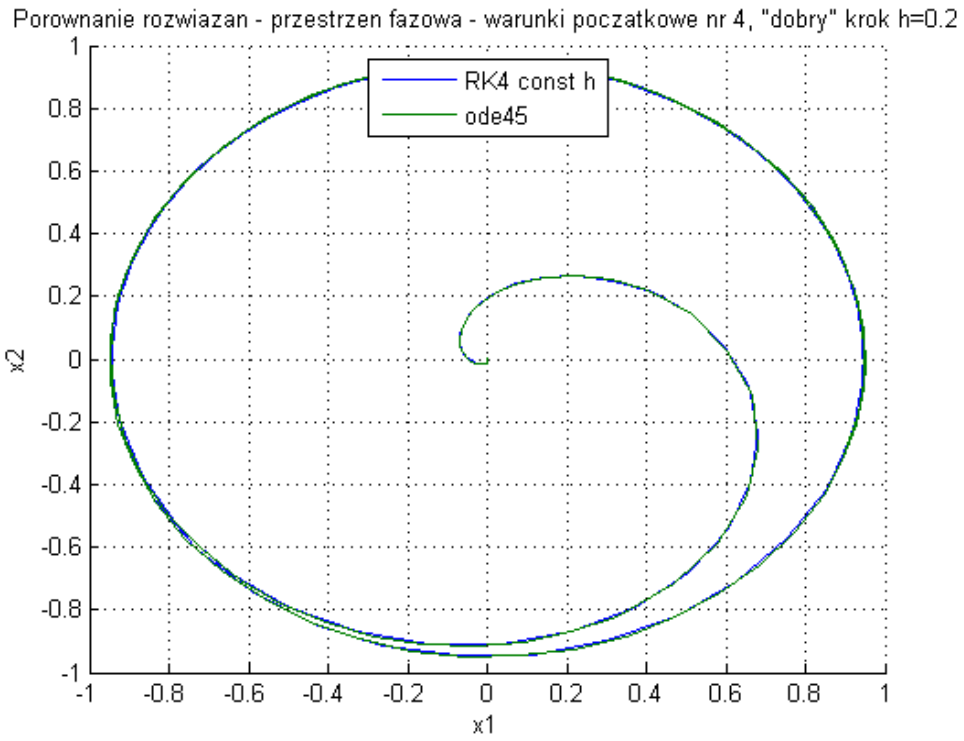


Wykres 28.

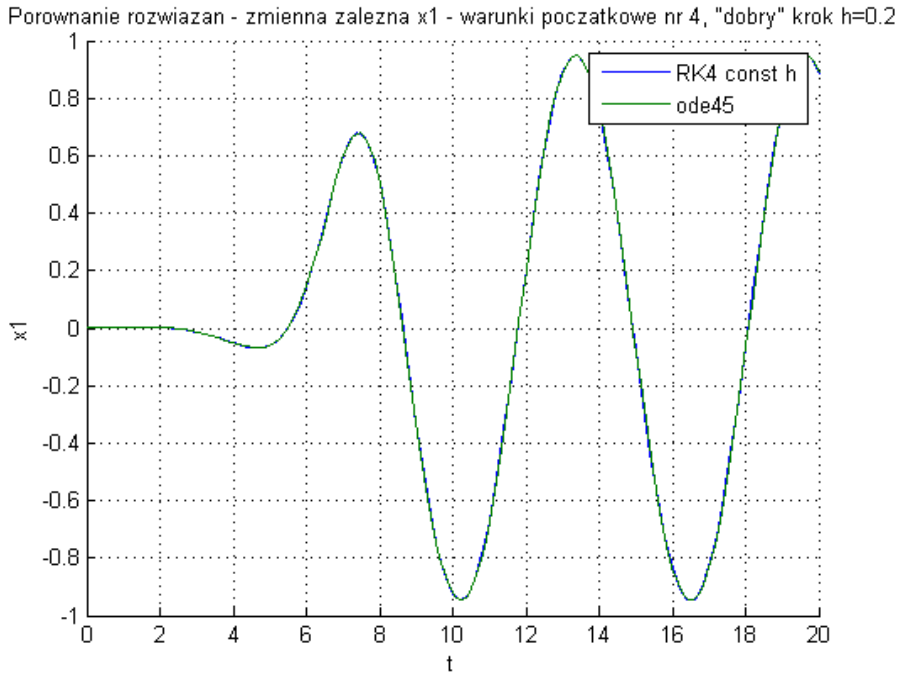


Wykres 29.

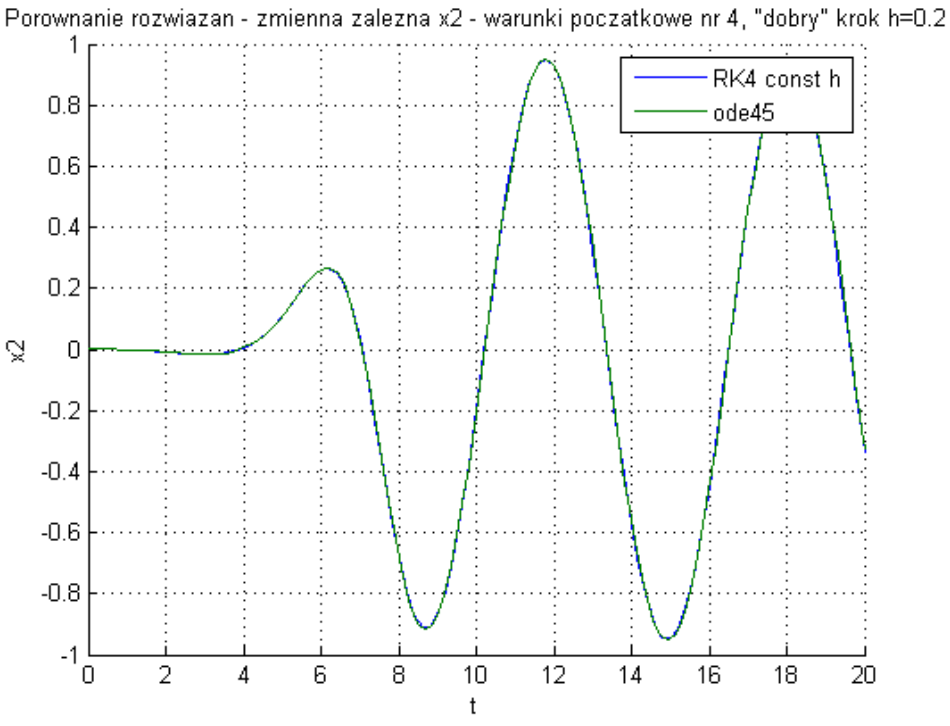
Zauważmy, że błędy aproksymacji szybko zaczynają oscylować na dość niskim poziomie rzędu $4 \div 6 \cdot 10^{-3}$ (patrz wykresy 27., 28., 29.).



Wykres 30.



Wykres 31.



Wykres 32.

Uzyskane rozwiązanie (metodą RK4 ze stałym krokiem) dobrze oddaje rozwiązanie uzyskane dzięki poleceniu ode45 języka MATLAB (wykresy 30., 31., 32.).

2) Metoda wielokrokowa predyktor-korektor Adamsa czwartego rzędu ze stałym krokiem

Metoda wielokrokowa predyktor-korektor Adamsa czwartego rzędu ze stałym krokiem jest jedną z metod służących do rozwiązywania równań i układów równań różniczkowych zwyczajnych.

Niech $x \in \mathrm{R}$ będzie zmienną niezależną
oraz niech $\boldsymbol{y} = [y_1 \quad y_2 \quad \cdots \quad y_m]^T$ będzie wektorem zmiennych zależnych.

Dany jest układ równań różniczkowych

$$\left[\frac{dy_1}{dx} \quad \frac{dy_2}{dx} \quad \cdots \quad \frac{dy_m}{dx}\right]^T = [f_1(x, \boldsymbol{y}) \quad f_2(x, \boldsymbol{y}) \quad \cdots \quad f_m(x, \boldsymbol{y})]^T,$$

gdzie f_i: $\mathrm{R}^{m+1} \to \mathrm{R}$ (dla $i = 1,2, \ldots, m$) oraz $x \in [a, b]$.

Przyjmując oznaczenia $\boldsymbol{y}'(x) \stackrel{\text{def}}{=} \left[\frac{dy_1}{dx} \quad \frac{dy_2}{dx} \quad \cdots \quad \frac{dy_m}{dx}\right]^T$
oraz $\boldsymbol{f} \stackrel{\text{def}}{=} [f_1(x, \boldsymbol{y}) \quad f_2(x, \boldsymbol{y}) \quad \cdots \quad f_m(x, \boldsymbol{y})]^T$, otrzymujemy zapis wektorowy układu równań różniczkowych zwyczajnych: $\boldsymbol{y}'(x) = \boldsymbol{f}(x, \boldsymbol{y})$,
gdzie $x \in [a, b]$ oraz wektor warunków początkowych $\boldsymbol{y}(a) = \boldsymbol{y}_a$.

W naszej metodzie, w fazie predykcji (P), będziemy używali 4-krokowej metody Adamsa-Bashfortha (jest to metoda jawna, 4. rzędu). Z kolei w fazie korekcji (K) będziemy używali 3-krokowej metody Adamsa-Moultona (jest to metoda jawna, 4. rzędu).

Schemat naszej metody oznaczamy przez $P_4\ E\ K_3\ E$, co oznacza, że zastosujemy 4-krokowy predyktor (P_4), następnie dokonamy ewaluacji (E),
w kolejnej fazie zastosujemy 3-krokowy korektor (K_3), a na końcu dokonamy ewaluacji (E).

Oznaczenia:

- $y_{n,i}$ – wartość i-tej zmiennej zależnej w n-tej iteracji
- $\boldsymbol{y}_n \stackrel{\text{def}}{=} \left[y_{n,1}\ y_{n,2}\ \ldots y_{n,m}\right]^T$ – wektor wartości zmiennych zależnych w n-tej iteracji
- $f_{n,i} \stackrel{\text{def}}{=} f_i(x_n, \boldsymbol{y}_n)$ – wartość i-tej funkcji prawych stron dla wartości zmiennej niezależnej w n-tej iteracji (x_n) oraz dla wektora wartości zmiennych zależnych w n-tej iteracji ($\boldsymbol{y}_n$)
- $k_{pred} = 4$ – liczba kroków predyktora
- $k_{corr} = 3$ – liczba kroków korektora

Schemat metody (dla pojedynczej iteracji):

1) Dla $i = 1,2,\dots,m$ (tj. dla każdej zmiennej zależnej) wykonuj:
   1.1) P: $y_{n,i}^{[0]} = y_{n-1,i} + h \cdot \sum_{j=1}^{k_{pred}} \beta_j \cdot f_{n-j,i}$ (faza predykcji)

2) E: $\boldsymbol{f}_n^{[0]} = \boldsymbol{f}(x_n,\ \boldsymbol{y}_n^{[0]})$, gdzie $\boldsymbol{y}_n^{[0]} = \left[y_{n,1}^{[0]}\ y_{n,2}^{[0]}\ \dots y_{n,m}^{[0]}\right]^T$ (faza ewaluacji)
3) Dla $i = 1,2,\dots,m$ (tj. dla każdej zmiennej zależnej) wykonuj:
   3.1) K: $y_{n,i} = y_{n-1,i} + h \cdot \beta_0^* \cdot f_{n,i}^{[0]} + h \cdot \sum_{j=1}^{k_{corr}} \beta_j^* \cdot f_{n-j,i}$ (faza korekcji)
4) E: $\boldsymbol{f}_n = \boldsymbol{f}(x_n, \boldsymbol{y}_n)$ (faza ewaluacji)

Wektor $\boldsymbol{\beta} = [\beta_1\ \beta_2\ \beta_3\ \beta_4]^T$ jest wektorem współczynników metody w fazie predykcji i wynosi $\boldsymbol{\beta} = \frac{1}{24} \cdot [55\ \ -59\ \ 37\ \ -9]^T$ [1].

Wektor $\boldsymbol{\beta}^* = [\beta_0^*\ \beta_1^*\ \beta_2^*\ \beta_3^*]^T$ jest wektorem współczynników metody w fazie korekcji i wynosi $\boldsymbol{\beta}^* = \frac{1}{24} \cdot [9\ \ 19\ \ -5\ \ 1]^T$ [1].

Wektor estymat błędów aproksymacji będziemy obliczali ze wzoru $\boldsymbol{\delta}_n(h) = \frac{-19}{270} \cdot (\boldsymbol{y}_n^{[0]} - \boldsymbol{y}_n)$ [1].

W zaimplementowanym rozwiązaniu nie wykonuję fazy 4)
(z racji na to, że nie używam wyników tych obliczeń gdziekolwiek indziej).

Listing solwera (Adams4_const_h.m):

```
function [ y, delta ] = Adams4_const_h( f, m, a, b, h, y0 )
% Wielokrokowa metoda predyktor-korektor Adamsa 4. rzędu ze stałym krokiem,
% sluzaca do rozwiazywania rownan
% i ukladow rownan rozniczkowych zwyczajnych.
% Metoda predyktora: metoda (jawna)   4-krokowa Adamsa-Bashfortha (4.
rzedu)
% Metoda korektora: metoda (niejawna) 3-krokowa Adamsa-Moultona (4. rzedu)
% Schemat metody: P4 E K3 E

% y - wartosci zmiennych zaleznych,
%     dla okreslonego przedzialu wartosci zmiennej niezaleznej
% delta - oszacowania bledu pojedynczego kroku o długości h
%           dla kazdej zmiennej zaleznej
% f - funkcje prawych stron (w tablicy komorkowej)
% m - liczba rownan (i zmiennych zaleznych zarazem)
% [a, b] - przedzial zmiennej niezaleznej,
%          dla ktorego poszukujemy rozwiazania
% y0 - wektor warunkow poczatkowych
% h - krok całkowania
```

```
% Liczba krokow predyktora
k_pred = 4;


% Liczba krokow korektora
k_corr = 3;


% Wspolczynniki dla metody predyktora
% (tj. metody (jawnej) 4-krokowej Adamsa-Bashfortha)
beta_pred = 1/24 * [ 55 -59 37 -9 ];


% Wspolczynniki dla metody korektora
% (tj. metody (niejawnej) 3-krokowej Adamsa-Moultona)
beta_corr = 1/24 * [ 9 19 -5 1 ];


% Wartosci zmiennej niezaleznej,
% dla ktorych obliczymy wartosci zmiennych zaleznych
x = a:h:b;


% Liczba krokow calkowania
n = length(x);


% y - wartosci zmiennych zaleznych:
% wiersze - wartosci wszystkich zmiennych zaleznych
%           dla danej iteracji,
% kolumny - wartosci okreslonych zmiennych zaleznych
%           we wszystkich iteracjach;
% Inicjacja warunkami poczatkowymi
y(1, :) = y0;


% Oszacowania bledu pojedynczego kroku o dlugosci h
% dla kazdej zmiennej zaleznej
delta(1,:) = zeros(m, 1);


% (Prawie) kazda iteracja
for nr=2:n

    % 1) Predykcja

    % Wektor wartosci predykcji
    y_pred = zeros(1,m);

    % Kazda zmienna zalezna
    for i=1:m

        % Obliczenie sumy pomocniczej
        sum = 0;
        for j=1:k_pred
            if nr - j >= 1
```

```
                sum = sum + beta_pred(j) * feval( f{i}, x(nr-j), y(nr-j, :)
);
            end
        end

        % Obliczenie wartosci predykcji i-tej zmiennej zaleznej
        % w aktualnej iteracji
        y_pred(i) = y(nr-1, i) + h * sum;
    end


    % 2) Ewaluacja

    % Wektor wartosci funkcji prawych stron
    % dla wartosci predykcji
    f_eval = zeros(1,m);

    for i=1:m
        f_eval(i) = feval( f{i}, x(nr), y_pred );
    end


    % 3) Korekcja

    % Kazda zmienna zalezna
    for i=1:m

        % Obliczenie sumy pomocniczej
        sum = 0;
        for j=2:k_corr+1
            if nr - j >= 1
                sum = sum + beta_corr(j) * feval( f{i}, x(nr-j+1), y(nr-
j+1, :) );
            end
        end

        % Obliczenie wartosci i-tej zmiennej zaleznej
        % w aktualnej iteracji
        y(nr, i)    =    y(nr-1, i)    +    h*sum    +    h * beta_corr(1) *
f_eval(i);
    end


    % 4) Ewaluacja - nie jest konieczna


    % Obliczenie wektora estymat bledow aproksymacji
    delta(nr-1, :) = -19/270 * ( y_pred - y(nr, :) );

end



end
```

Listing skryptu wywołującego (Zad2.m):

```
% Zadanie 2.
% Wielokrokowa metoda predyktor-korektor Adamsa 4. rzędu ze stałym krokiem.


clear;
clc;


% Liczba rownan rozniczkowych (i zmiennych zaleznych zarazem)
m = 2;


% f - tablica komorkowa, zawierajaca uchwyty
% do prawych stron rownan rozniczkowych
f = cell(m,1);


% Wstawienie (do tablicy f) uchwytow
% do prawych stron rownan rozniczkowych
f{1} = @(x, y)( y(2) + y(1)*(0.9 - y(1)^2 - y(2)^2) );
f{2} = @(x, y)( -y(1) + y(2)*(0.9 - y(1)^2 - y(2)^2) );


% Przedzial [a,b] zmiennej niezaleznej,
% dla ktorego bedziemy calkowali
% uklad rownan rozniczkowych
a = 0;
b = 20;


% Warunki poczatkowe (kolejne wiersze)
y0 = [10    8;
       0    9;
       8    0;
    1e-3 1e-3];


% Kroki calkowania (odpowiadajace kolejnym warunkom poczatkowym)
h = [0.004 0.004 0.005 0.2];


% Zbyt duze kroki calkowania (odpowiadajace kolejnym warunkom poczatkowym)
h_big = [0.005 0.008 0.01 0.3];


% Liczba warunkow poczatkowych
L = length(h);


% Numery zestawow warunkow poczatkowych
nr = 1:L;
```

```
% Czasy dzialania solwerow
T = zeros(L,1);


% Liczby iteracji
iter = zeros(L,1);


% Licznik wykresow
q = 1;


% Wyznaczenie rozwiazan
% dla roznych warunkow poczatkowych
for i=1:L

    % Obliczenie rozwiazan i bledow oszacowan dla "dobrego" kroku h
    tic;
    [y, delta_h] = Adams4_const_h(f, m, a, b, h(i), y0(i,:));
    T(i) = toc;


    % Wykres fazowy dla "dobrego" kroku h
    figure(q);
    hold on;
    plot( y(:,1), y(:,2) );
    grid on;
    title( ['Rozwiazanie w przestrzeni fazowej - warunki poczatkowe nr ',
num2str(i), ', krok h=', num2str( h(i) )] );
    xlabel('x1');
    ylabel('x2');
    hold off;
    q = q+1;


    % Obliczenie rozwiazan i bledow oszacowan (zbyt duzy krok h)
    [y_big, delta_h_big] = Adams4_const_h(f, m, a, b, h_big(i), y0(i,:));


    % Wykres fazowy dla zbyt "duzego" kroku h
    figure(q);
    hold on;
    plot( y_big(:,1), y_big(:,2) );
    grid on;
    title( ['Rozwiazanie w przestrzeni fazowej - warunki poczatkowe nr ',
num2str(i), ', zbyt "duzy" krok h=', num2str( h_big(i) )] );
    xlabel('x1');
    ylabel('x2');
    hold off;
    q = q+1;


    % Wyznaczenie wektora (osi poziomej) do wykresow bledow aproksymacji
    x = a:h(i):b;


    % Wyznaczenie liczby iteracji
    iter(i) = length(x);
```

```
    % Wykres modulow bledow aproksymacji - zmienna zalezna x1 ("dobry" krok
h)
    figure(q);
    hold on;
    plot( x([2:length(x)]), abs( delta_h(:,1) ) );
    grid on;
    title( ['Oszacowanie modulow bledow aproksymacji - zmienna zalezna x1 -
warunki poczatkowe nr ', num2str(i), ', "dobry "krok h=', num2str( h(i) )]
);
    xlabel('t');
    ylabel('Oszacowanie bledu aproksymacji');
    hold off;
    q = q+1;


    % Wykres bledu aproksymacji - zmienna zalezna x2 ("dobry" krok h)
    figure(q);
    hold on;
    plot( x([2:length(x)]), abs( delta_h(:,2) ) );
    grid on;
    title( ['Oszacowanie modulow bledow aproksymacji - zmienna zalezna x2 -
warunki poczatkowe nr ', num2str(i), ', "dobry" krok h=', num2str( h(i) )]
);
    xlabel('t');
    ylabel('Oszacowanie bledu aproksymacji');
    hold off;
    q = q+1;


    % Wykres normy maksimum bledu aproksymacji ("dobry" krok h)
    delta_h_aggr = zeros( length(delta_h), 1 );
    for c=1:length(delta_h)
        delta_h_aggr(c) = norm( delta_h(c, :), Inf );
    end

    figure(q);
    hold on;
    plot( x([2:length(x)]), delta_h_aggr );
    grid on;
    title( ['Norma maksimum bledu aproksymacji - warunki poczatkowe nr ',
num2str(i), ', "dobry" krok h=', num2str( h(i) )] );
    xlabel('t');
    ylabel('Oszacowanie bledu aproksymacji');
    hold off;
    q = q+1;


    % Rozwiazanie przy pomocy polecenia ode45
    [x_ode, y_ode] = ode45(@right_sides,[a b], y0(i,:));


    % Wykresy porownawcze

    % Wykres fazowy
    figure(q);
    hold on;
    plot( y(:,1), y(:,2), y_ode(:,1), y_ode(:,2) );
```

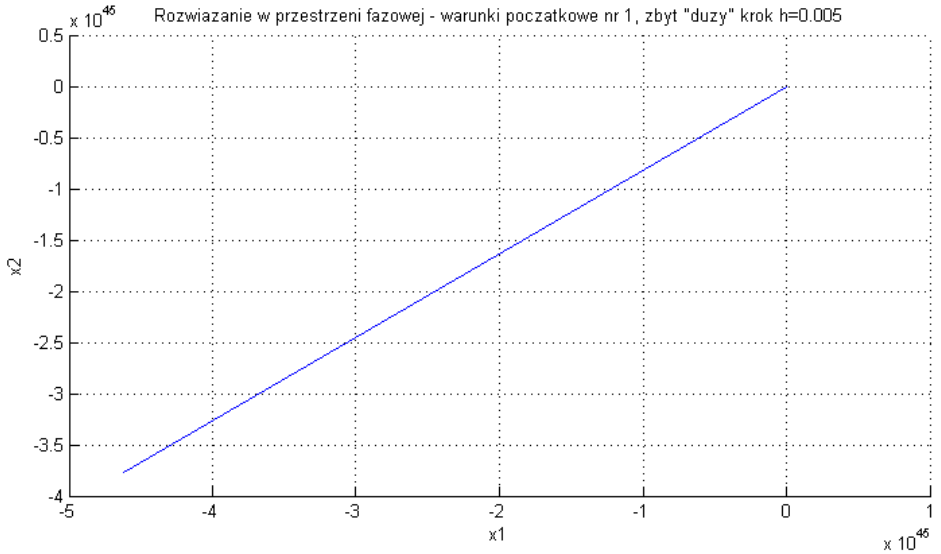
```
    grid on;
    title( ['Porownanie rozwiazan - przestrzen fazowa - warunki poczatkowe
nr ', num2str(i), ', "dobry" krok h=', num2str( h(i) )] );
    xlabel('x1');
    ylabel('x2');
    legend('Adams4 const h', 'ode45', 'Location', 'North');
    hold off;
    q = q+1;


    % Zmienna zalezna x1
    figure(q);
    hold on;
    plot( x, y(:,1), x_ode, y_ode(:,1) );
    grid on;
    title( ['Porownanie rozwiazan - zmienna zalezna x1 - warunki poczatkowe
nr ', num2str(i), ', "dobry" krok h=', num2str( h(i) )] );
    xlabel('t');
    ylabel('x1');
    legend('Adams4 const h', 'ode45');
    hold off;
    q = q+1;


    % Zmienna zalezna x2
    figure(q);
    hold on;
    plot( x, y(:,2), x_ode, y_ode(:,2) );
    grid on;
    title( ['Porownanie rozwiazan - zmienna zalezna x2 - warunki poczatkowe
nr ', num2str(i), ', "dobry" krok h=', num2str( h(i) )] );
    xlabel('t');
    ylabel('x2');
    legend('Adams4 const h', 'ode45');
    hold off;
    q = q+1;

end
```

Wykresy:

Warunki początkowe a):

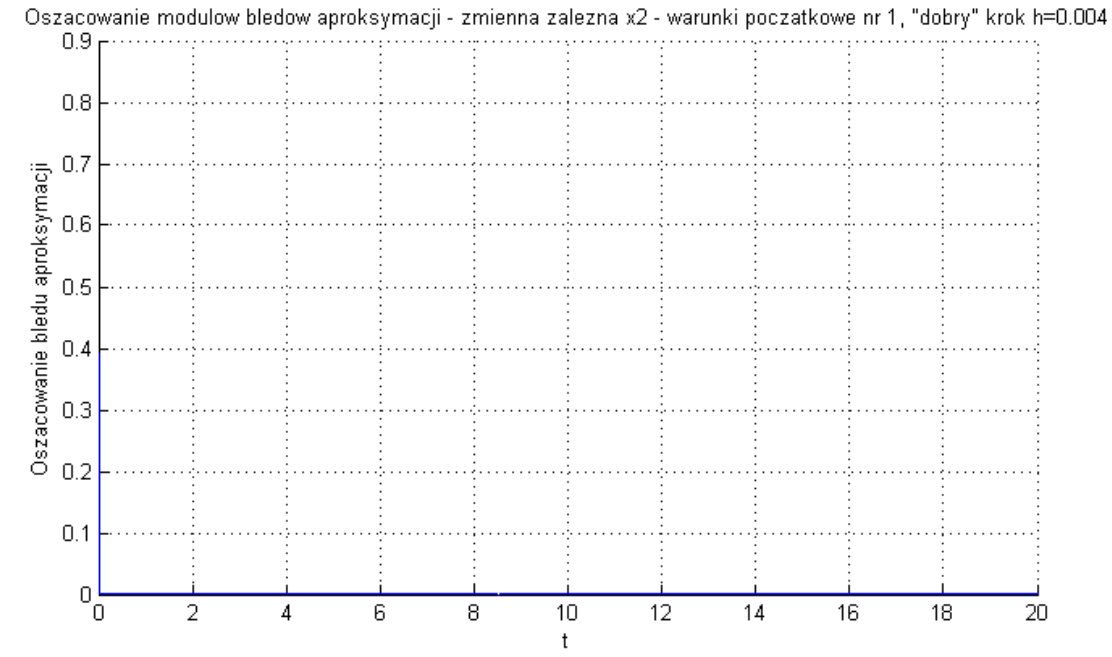
Dobrałem krok $h = 0.004$.
Jeśli zwiększymy tę wartość do $h = 0.005$, to zmienia się istotnie postać rozwiązania – zmienne zależne osiągają bardzo duże wartości oraz zmienia się przebieg rozwiązania (patrz wykresy 33. oraz 34.).

Wykres 33.

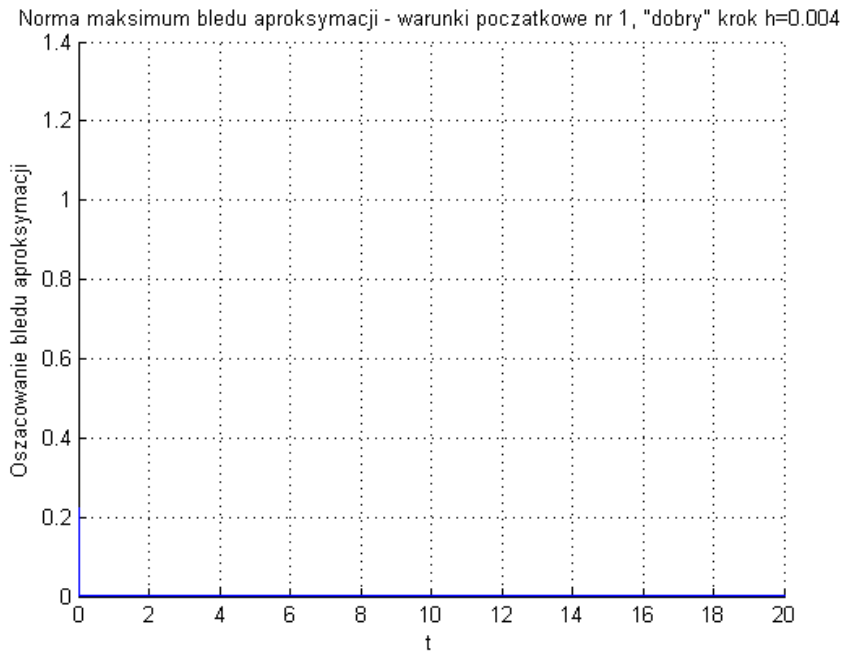


Wykres 34.

Wykres 35.

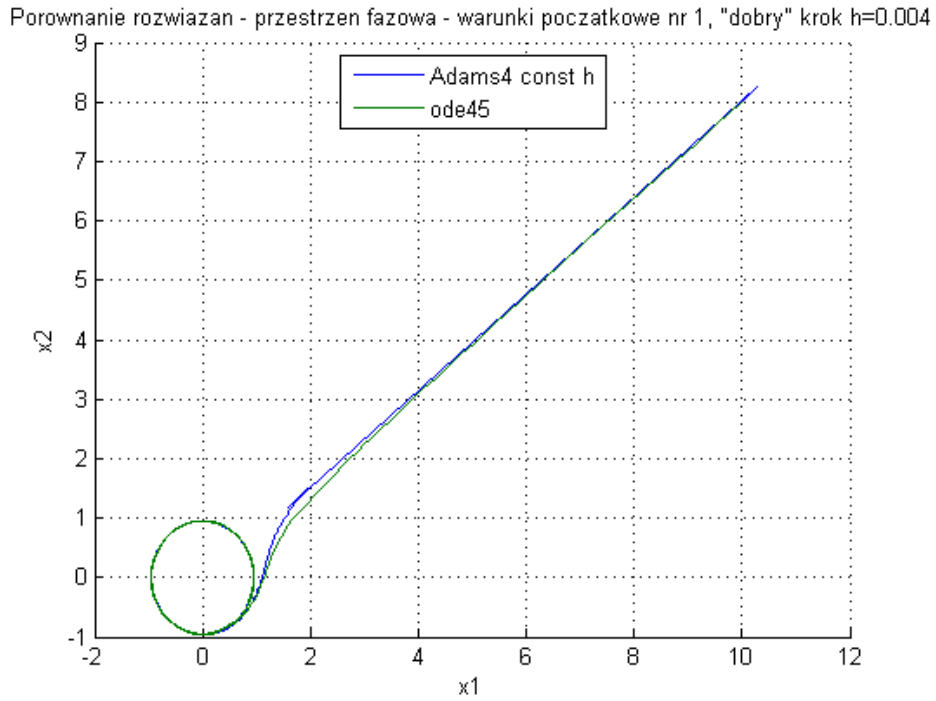


Wykres 36.

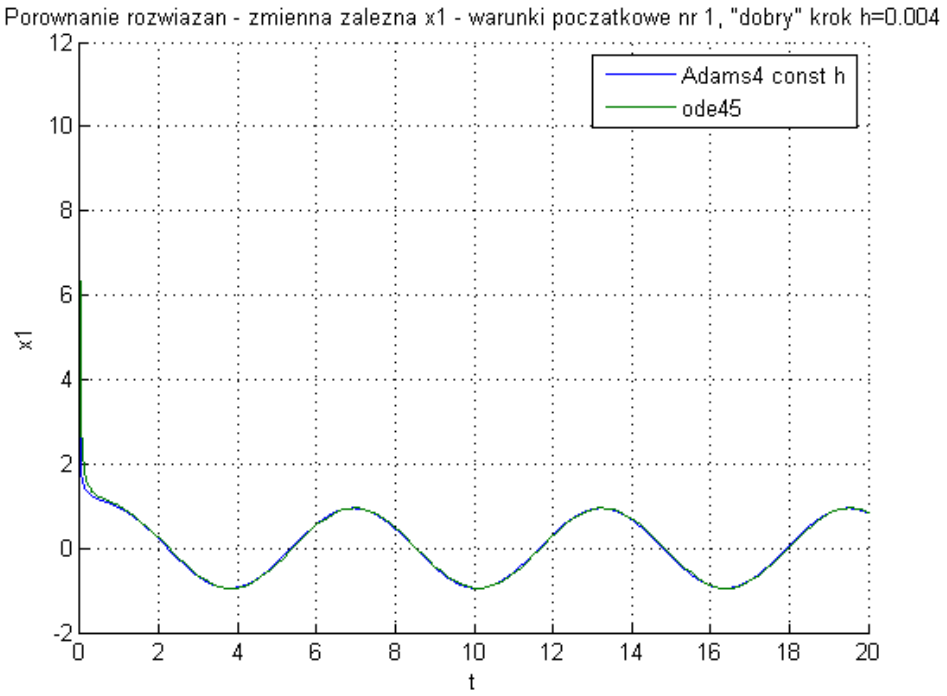


Wykres 37.

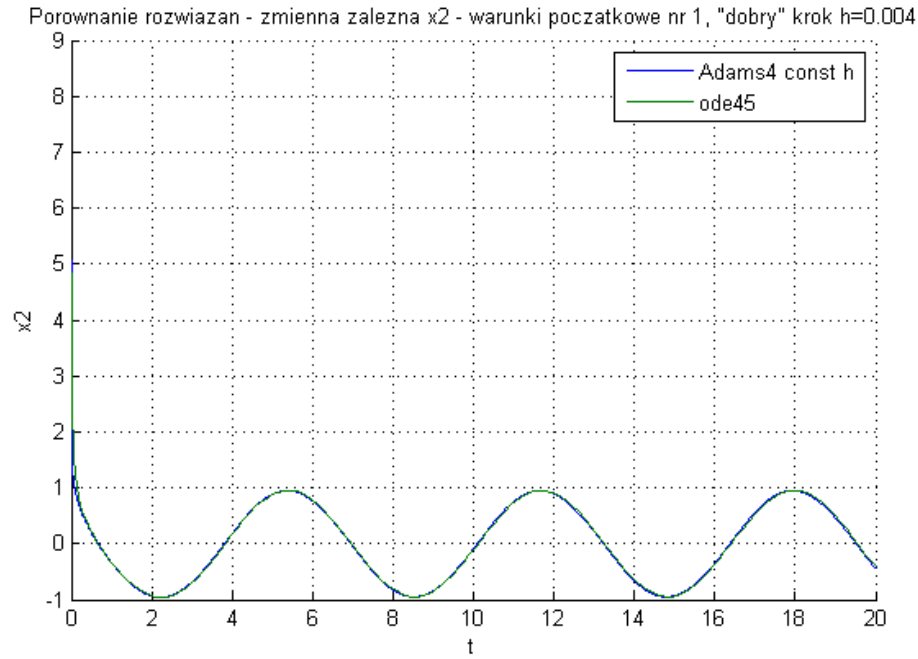
Zauważmy, że błędy aproksymacji bardzo szybko ustalają się na niskim poziomie (patrz wykresy 35., 36., 37.).



Wykres 38.



Wykres 39.

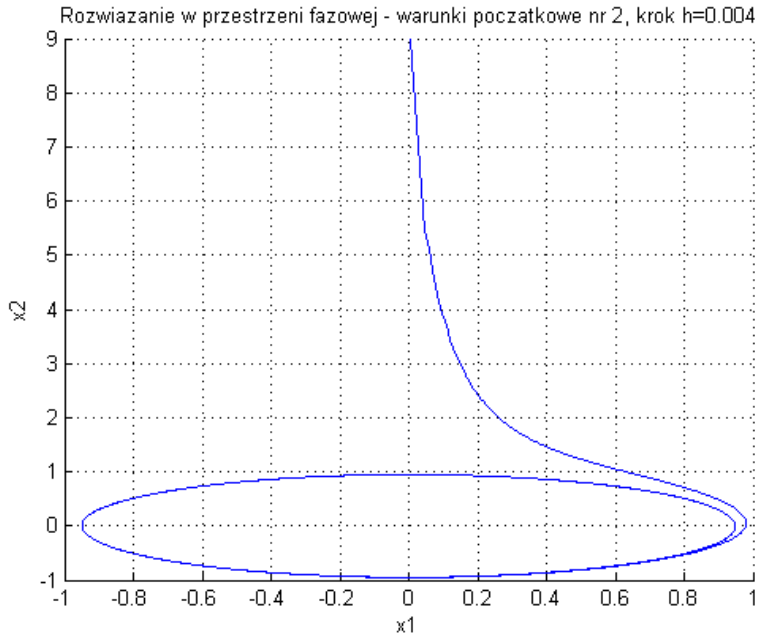


Wykres 40.

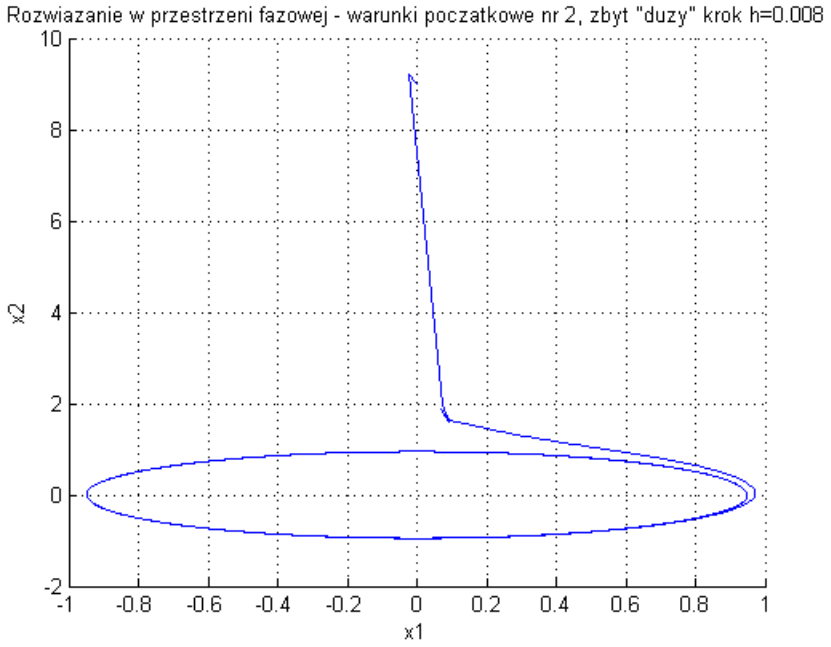
Uzyskane rozwiązanie (metodą predyktor-korektor Adamsa ze stałym krokiem) dobrze oddaje rozwiązanie uzyskane dzięki poleceniu ode45 (języka MATLAB) w niektórych fragmentach, a w niektórych już gorzej (wykresy 38., 39., 40.).

Warunki początkowe b):

Dobrałem krok $h = 0.004$.
Jeśli zwiększymy tę wartość do $h = 0.008$, to zmienia się istotnie postać rozwiązania – zmienne zależne osiągają bardzo duże wartości oraz zmienia się przebieg rozwiązania (patrz wykresy 41. oraz 42.).



Wykres 41.



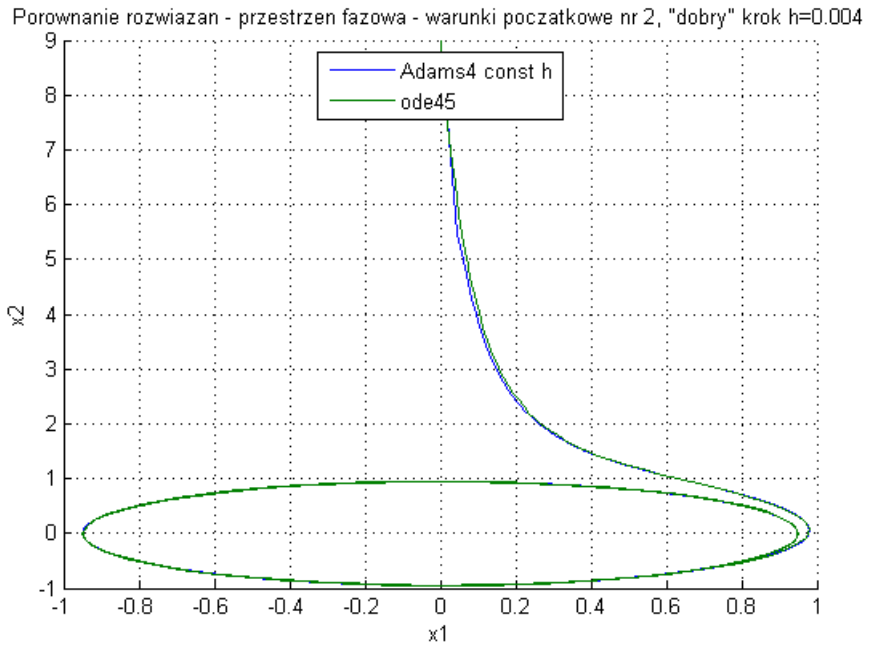
Wykres 42.

Wykres 43.

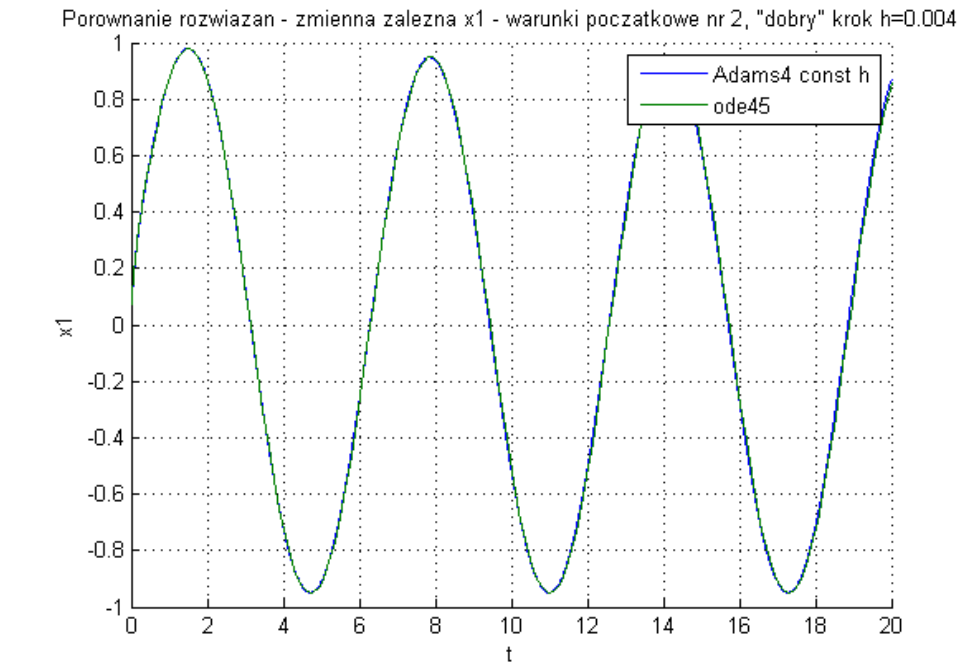
Wykres 44.

Wykres 45.

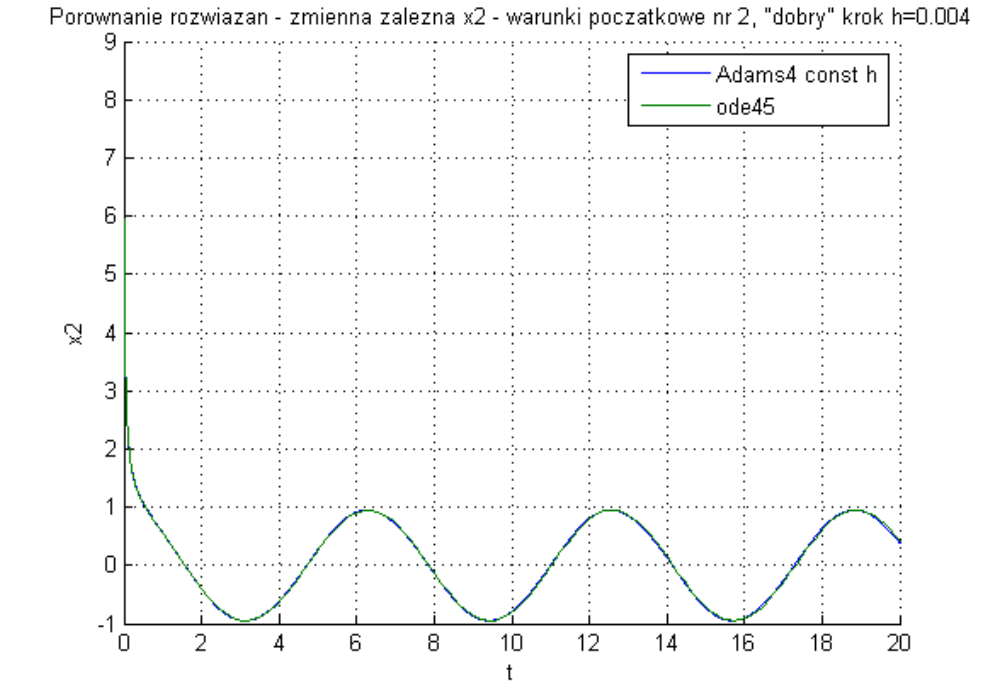
Zauważmy, że błędy aproksymacji bardzo szybko ustalają się na niskim poziomie (patrz wykresy 43., 44., 45.).



Wykres 46.



Wykres 47.

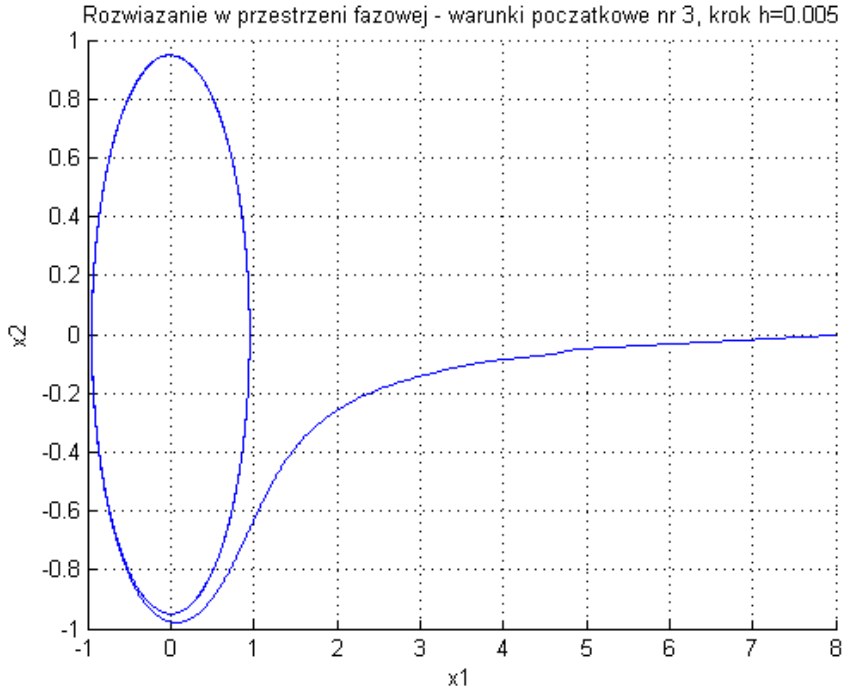


Wykres 48.

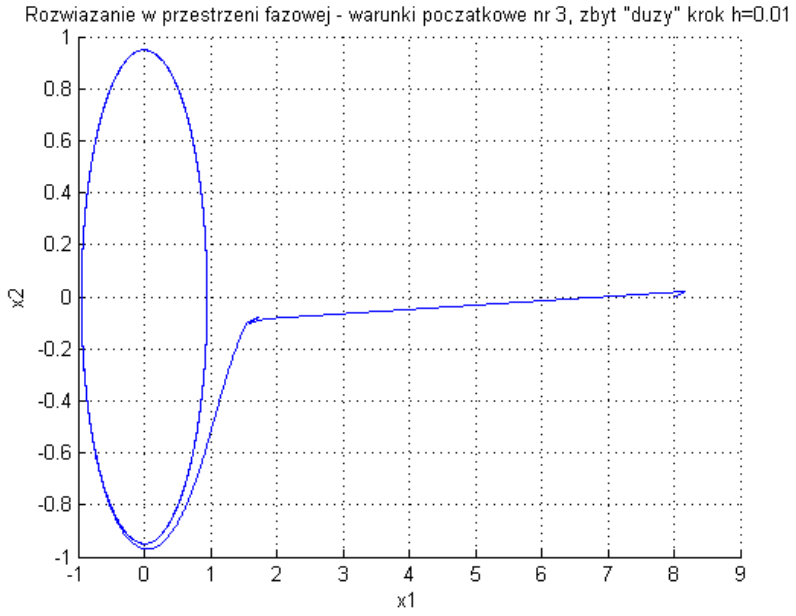
Uzyskane rozwiązanie (metodą predyktor-korektor Adamsa ze stałym krokiem) dość dobrze oddaje rozwiązanie uzyskane dzięki poleceniu ode45 języka MATLAB (wykresy 46., 47., 48.).

Warunki początkowe c):

Dobrałem krok $h = 0.005$.
Jeśli zwiększymy tę wartość do $h = 0.01$, to rozwiązanie przestaje być „gładkie” (patrz wykresy 49. oraz 50.).

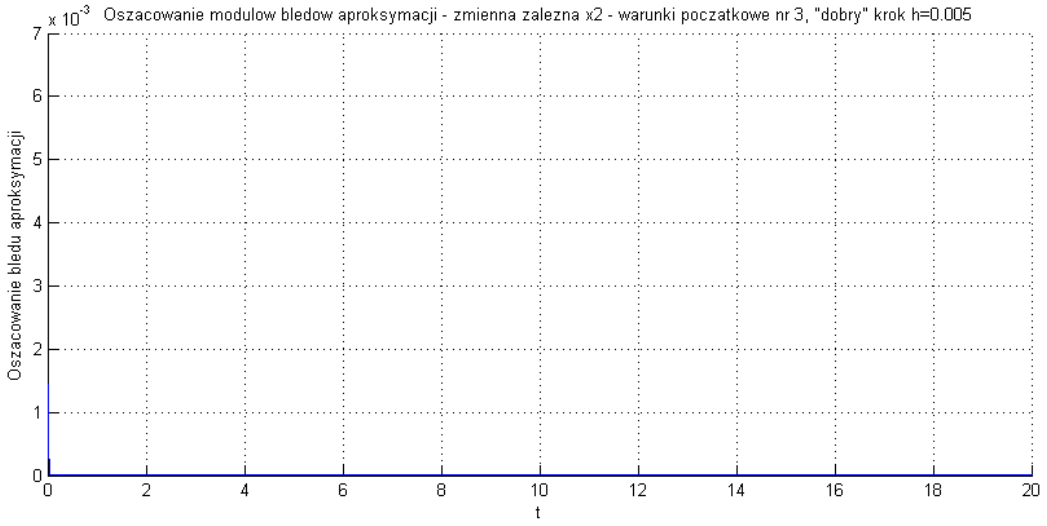


Wykres 49.



Wykres 50.

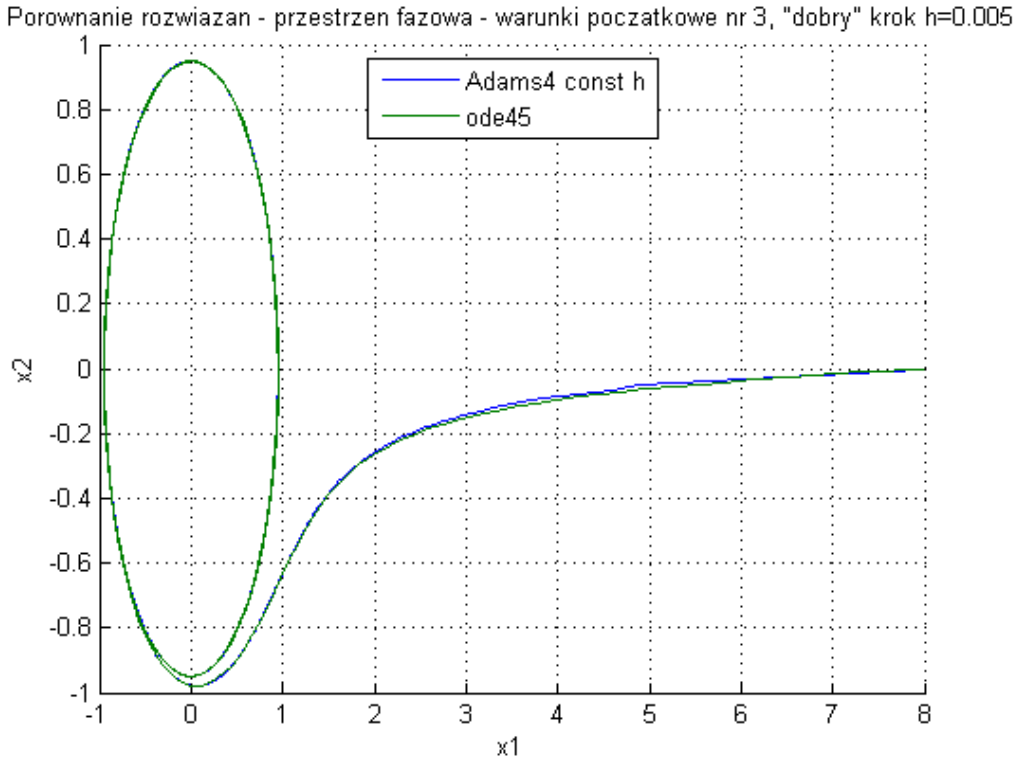
Wykres 51.



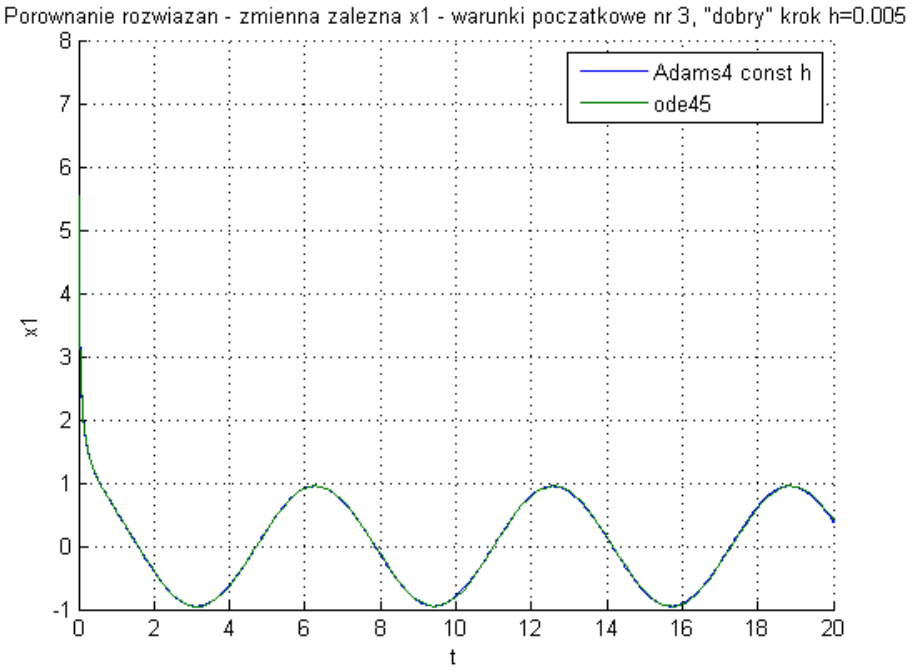
Wykres 52.

Wykres 53.

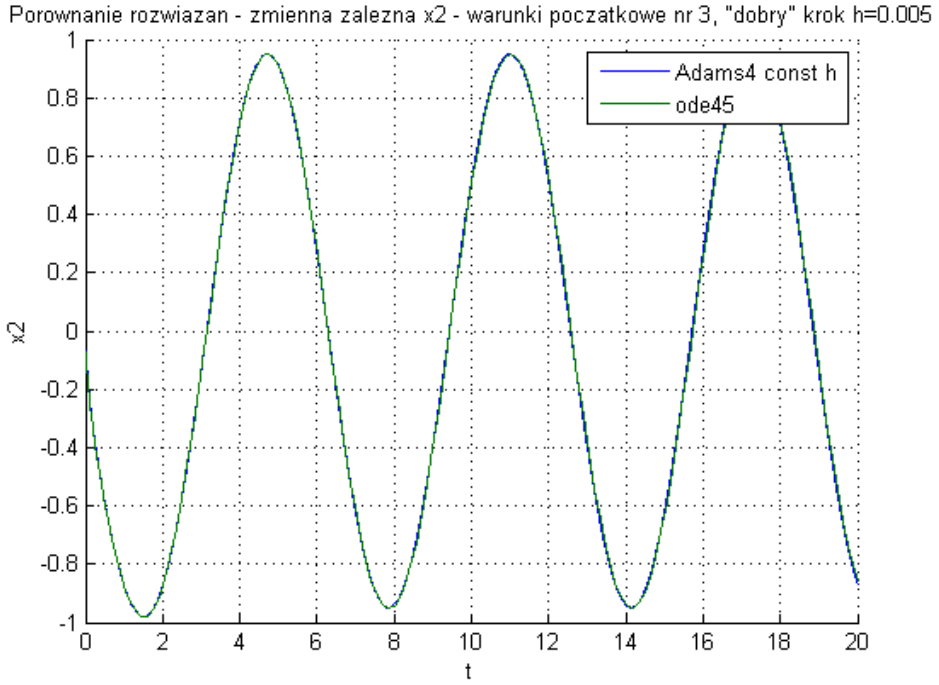
Zauważmy, że błędy aproksymacji bardzo szybko ustalają się na niskim poziomie (patrz wykresy 51., 52., 53.).



Wykres 54.



Wykres 55.

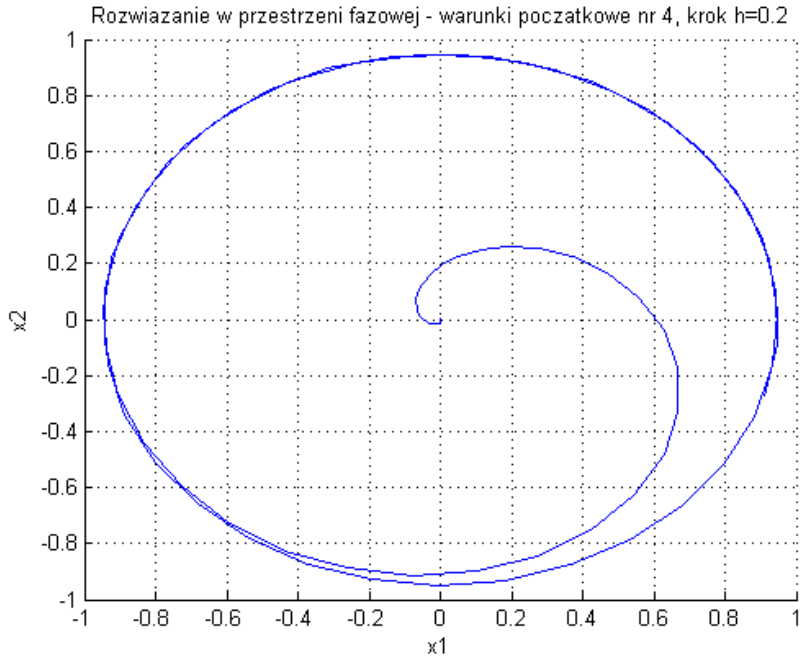


Wykres 56.

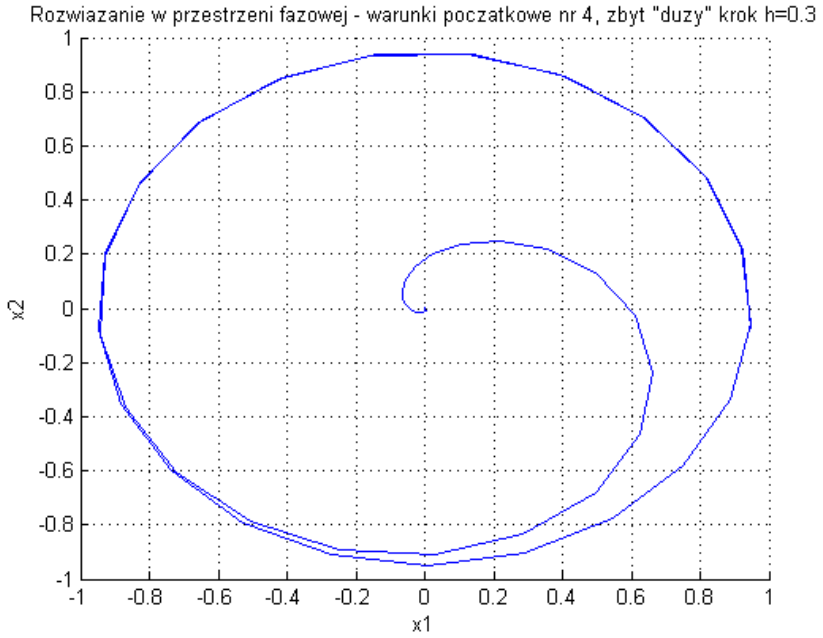
Uzyskane rozwiązanie (metodą RK4 ze stałym krokiem) dość dobrze oddaje rozwiązanie uzyskane dzięki poleceniu ode45 języka MATLAB (wykresy 54., 55., 56.).

Warunki początkowe d):

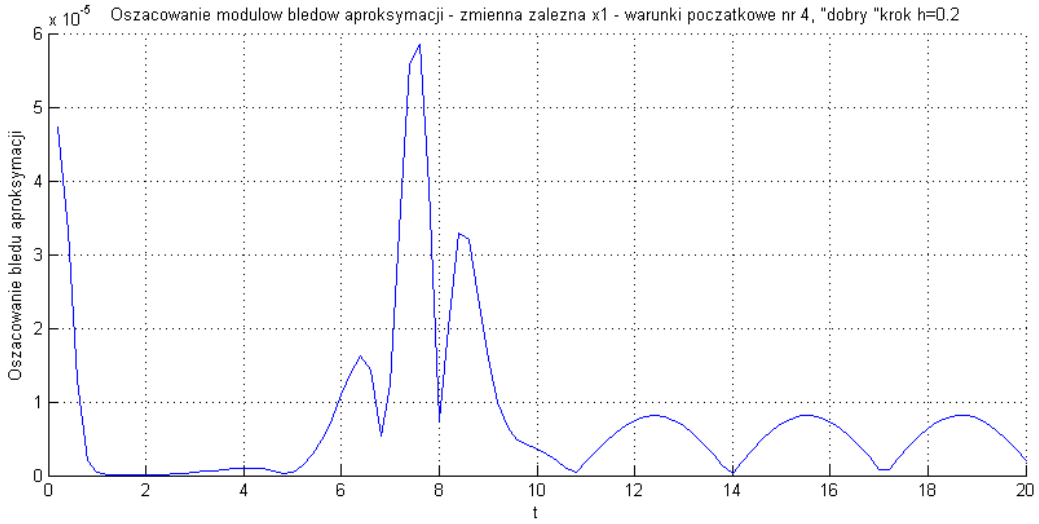
Dobrałem krok $h = 0.2$.
Jeśli zwiększymy tę wartość do $h = 0.3$, to rozwiązanie przestaje być „gładkie” (patrz wykresy 57. oraz 58.).



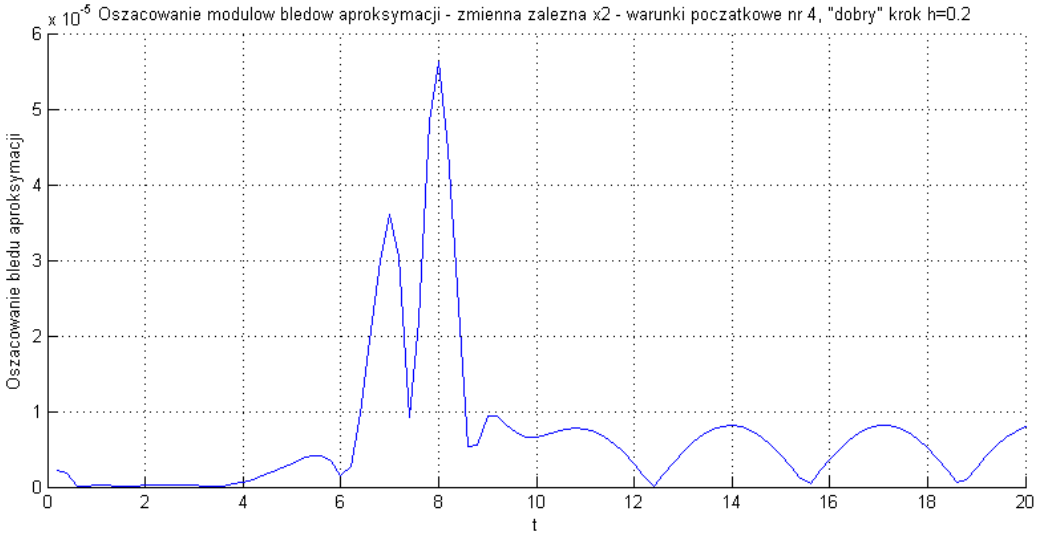
Wykres 57.



Wykres 58.



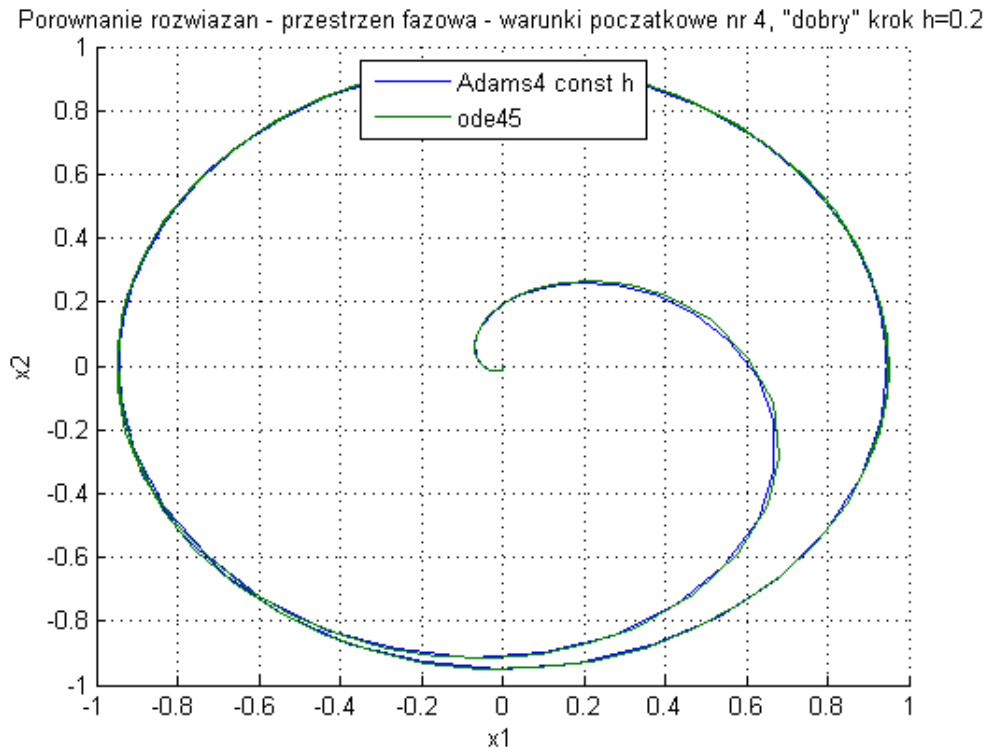
Wykres 59.



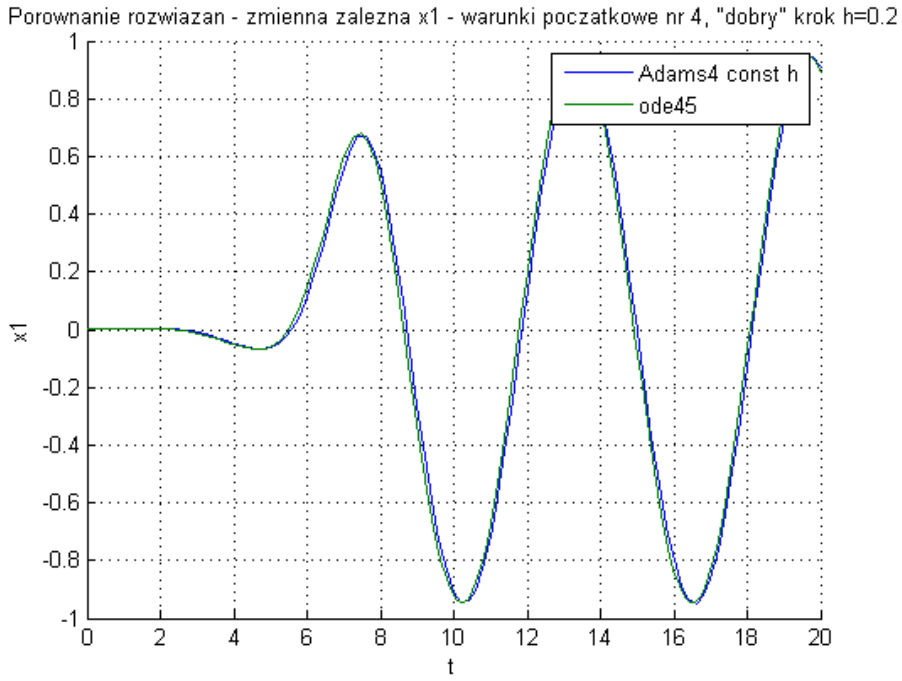
Wykres 60.

Wykres 61.

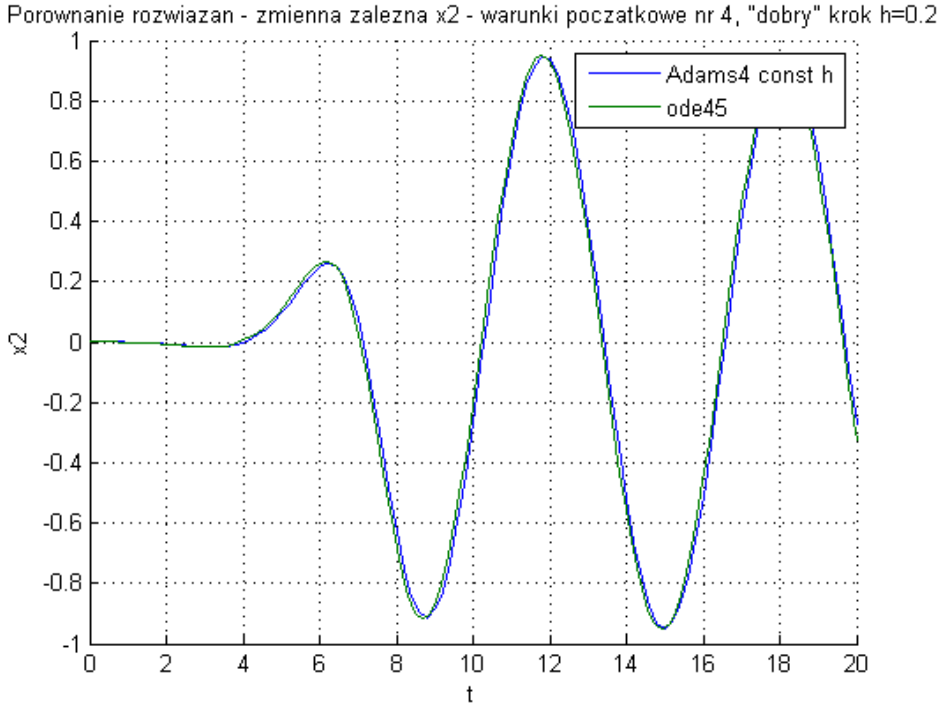
Zauważmy, że błędy aproksymacji są na niskim poziomie – nie przekraczają $6 \cdot 10^{-5}$. Ponadto, od pewnego miejsca nie przekraczają 10^{-5} (patrz wykresy 59., 60., 61.).



Wykres 62.



Wykres 63.



Wykres 64.

Uzyskane rozwiązanie (metodą predyktor-korektor Adamsa ze stałym krokiem) dość dobrze (choć można dostrzec różnicę) oddaje rozwiązanie uzyskane dzięki poleceniu ode45 języka MATLAB (wykresy 62., 63., 64.).

3) Metoda Rungego–Kutty czwartego rzędu (RK4) ze zmiennym krokiem

Metoda Rungego-Kutty czwartego rzędu (RK4) ze zmiennym krokiem jest jedną z metod służących do rozwiązywania równań i układów równań różniczkowych zwyczajnych.

Niech $x \in \mathrm{R}$ będzie zmienną niezależną
oraz niech $\boldsymbol{y} = [y_1 \quad y_2 \quad \cdots \quad y_m]^T$ będzie wektorem zmiennych zależnych.

Dany jest układ równań różniczkowych

$$\left[\frac{dy_1}{dx} \quad \frac{dy_2}{dx} \quad \cdots \quad \frac{dy_m}{dx}\right]^T = [f_1(x,\boldsymbol{y}) \quad f_2(x,\boldsymbol{y}) \quad \cdots \quad f_m(x,\boldsymbol{y})]^T,$$

gdzie f_i: $\mathrm{R}^{m+1} \to \mathrm{R}$ (dla $i = 1,2,\dots,m$) oraz $x \in [a,b]$.

Przyjmując oznaczenia $\boldsymbol{y}'(x) \stackrel{\text{def}}{=} \left[\frac{dy_1}{dx} \quad \frac{dy_2}{dx} \quad \cdots \quad \frac{dy_m}{dx}\right]^T$
oraz $\boldsymbol{f} \stackrel{\text{def}}{=} [f_1(x,\boldsymbol{y}) \quad f_2(x,\boldsymbol{y}) \quad \cdots \quad f_m(x,\boldsymbol{y})]^T$, otrzymujemy zapis wektorowy układu równań różniczkowych zwyczajnych: $\boldsymbol{y}'(x) = \boldsymbol{f}(x,\boldsymbol{y})$,
gdzie $x \in [a,b]$ oraz wektor warunków początkowych $\boldsymbol{y}(a) = \boldsymbol{y}_a$.

Schemat metody [1]:

1) Przyjmij dokładność względną ε_w, dokładność bezwzględną ε_b, krok początkowy h_0 oraz minimalną długość kroku h_{min}.
2) Ustaw $x_0 \coloneqq a$, $\boldsymbol{y}_0 = \boldsymbol{y}_a$ oraz licznik iteracji $n \coloneqq 0$.
3) Za pomocą metody Rungego-Kutty RK4 wyznacz $\boldsymbol{y}_{n+1}$ oraz oblicz oszacowanie błędu $\boldsymbol{\delta}^{(n)}$ (patrz: opis metody RK4 ze stały krokiem), wektor $\boldsymbol{\varepsilon} = |\boldsymbol{y}_n| \cdot \varepsilon_w + \varepsilon_b$ oraz współczynnik korekty długości kroku
   $\alpha = \min\limits_{1 \le i \le m} \sqrt[p+1]{\frac{\varepsilon_i}{|\delta_i^{(n)}|}}$. Wylicz proponowaną korektę długości kroku $h_{n+1}^* = s \cdot \alpha \cdot h_n$
   (s – współczynnik bezpieczeństwa; przykładowo, $s = 0.9$).
4) Jeśli $s \cdot \alpha \ge 1$, to wykonuj:
   4.1) Jeśli $x_n + h_n \ge b$, to zakończ proces całkowania.
   4.2) Jeśli nie zachodzi warunek z punktu 4.1), to wykonuj:
      4.2.1) $x_{n+1} \coloneqq x_n + h_n$
      4.2.2) $h_{n+1} = \min(h_{n+1}^*,\ \beta \cdot h_n,\ b - x_n)$, ($\beta$ - heurystyczny współczynnik dla ograniczenia maksymalnego wzrostu długości kroku h_n w jednej iteracji; przykładowo, $\beta = 5$)
      4.2.3) $n \coloneqq n + 1$
      4.2.4) Idź do kroku 3)
5) Jeśli nie zachodzi warunek z punktu 4), to wykonuj:
   5.1) Jeśli $h_{n+1}^* < h_{min}$, to $h_n \coloneqq h_{n+1}^*$ oraz idź do kroku 3).
   5.2) Jeśli nie zachodzi warunek z punktu 5.2), to sygnalizujemy, że nie jest możliwe rozwiązanie zadania z zadaną dokładnością.

Listing solwera (RK4_var_h.m):

```
function [x, y, delta_h, error_code] = RK4_var_h(f, m, a, b, h, h_min, y0,
eps_w, eps_b)
% Metoda Rungego-Kutty RK4 ze zmiennym krokiem,
% sluzaca do rozwiazywania rownan
% i ukladow rownan rozniczkowych zwyczajnych.

% x - wektor wartosci zmiennej niezaleznej
% y - wartosci zmiennych zaleznych,
%     dla okreslonego przedzialu wartosci zmiennej niezaleznej
% delta_h - oszacowania bledu pojedynczego kroku o długości h
%           dla kazdej zmiennej zaleznej
% error_code - kod bledu (0 - brak bledow,
%                         1 - niemożliwe rozwiazanie z zadana dokladnoscia
h_min)
% f - funkcje prawych stron (w tablicy komorkowej)
% m - liczba rownan (i zmiennych zaleznych zarazem)
% [a, b] - przedzial zmiennej niezaleznej,
%          dla ktorego poszukujemy rozwiazania
% y0 - wektor warunkow poczatkowych
% h - poczatkowy krok calkowania
% h_min - minimalny krok calkowania
% eps_w - dokladnosc wzgledna
% eps_b - dokladnosc bezwzgledna


% Poczatkowo zakladamy, ze nie bedzie bledow
error_code = 0;


% Wspolczynnik bezpieczenstwa
% (do wyznaczania nowego kroku)
s = 0.9;


% Heurystyczny wspolczynnik maksymalnego wzrostu kolejnego kroku
beta = 5;


% Rzad metody
p = 4;


% Wartosci pomocnicze k, charakterystyczne dla naszej metody;
% wartosci te wyliczamy w ramach konkretnej iteracji
k = zeros(4, m);


% Wartosci zmiennej niezaleznej,
% dla ktorych obliczymy wartosci zmiennych zaleznych
x(1) = a;


% y - wartosci zmiennych zaleznych:
% wiersze - wartosci wszystkich zmiennych zaleznych
%           dla danej iteracji,
```

```
% kolumny - wartosci okreslonych zmiennych zaleznych
%           we wszystkich iteracjach;
% Inicjacja warunkami poczatkowymi
y(1, :) = y0;


% Oszacowania bledu pojedynczego kroku o długości h
% dla kazdej zmiennej zaleznej
delta_h(1,:) = zeros(m, 1);


% Licznik iteracji calkowania
nr = 2;


% Pomocnicza flaga - czy zakonczyc caly proces calkowania (0-nie, 1-tak)
koniec = 0;


% Dopoki iterujemy, dopoty nie osiagnelismy konca przedzialu
% zmiennej niezaleznej
while koniec == 0

    % Wektor zmiennych zaleznych, obliczanych poprzez 2 iteracje
    % z krokiem h/2
    y_tmp = y(nr-1,:);


    % 2 iteracje z krokiem h/2
    % (celem oszacowania błędu pojedynczego kroku o długości h)
    for j=1:2

        % Wartosci k1 dla kazdej zmiennej zaleznej
        for i=1:m
            k(1,i) = feval( f{i}, x(nr-1), y_tmp );
        end


        % Wartosci k2 dla kazdej zmiennej zaleznej
        for i=1:m
            k(2,i) = feval( f{i}, x(nr-1) + j*h/4,   y_tmp + j*h/4 * k(1,:)
);
        end


        % Wartosci k3 dla kazdej zmiennej zaleznej
        for i=1:m
            k(3,i) = feval( f{i},   x(nr-1) + j*h/4,   y_tmp + j*h/4 *
k(2,:) );
        end


        % Wartosci k4 dla kazdej zmiennej zaleznej
        for i=1:m
            k(4,i) = feval( f{i},   x(nr-1) + j*h/2,   y_tmp + j*h/2 *
k(3,:) );
        end
```

```
        % Obliczenie wartosci zmiennych zaleznych w aktualnym kroku
        y_tmp = y_tmp + h/12 * ( k(1,:) + 2*k(2,:) + 2*k(3,:) + k(4,:) );
    end




    % 1 iteracja z krokiem h

    % Wartosci k1 dla kazdej zmiennej zaleznej
    for i=1:m
        k(1,i) = feval( f{i}, x(nr-1), y(nr-1,:) );
    end


    % Wartosci k2 dla kazdej zmiennej zaleznej
    for i=1:m
        k(2,i) = feval( f{i}, x(nr-1) + h/2,   y(nr-1,:) + h/2 * k(1,:) );
    end


    % Wartosci k3 dla kazdej zmiennej zaleznej
    for i=1:m
        k(3,i) = feval( f{i},   x(nr-1) + h/2,   y(nr-1,:) + h/2 * k(2,:)
);
    end


    % Wartosci k4 dla kazdej zmiennej zaleznej
    for i=1:m
        k(4,i) = feval( f{i},   x(nr-1) + h,   y(nr-1,:) + h * k(3,:) );
    end


    % Obliczenie wartosci zmiennych zaleznych w aktualnym kroku
    y(nr,:) = y(nr-1,:) + h/6 * ( k(1,:) + 2*k(2,:) + 2*k(3,:) + k(4,:) );


    % Obliczenie oszacowania bledu pojedynczego kroku o dlugosci h
    delta_h(nr-1, :) = (2^p / (2^p - 1)) * ( y_tmp - y(nr,:) );


    % Wektor parametrow dokladnosci obliczen
    eps = abs( y(nr, :) ) * eps_w   +   eps_b;


    % Obliczenie wspolczynnika modyfikacji kroku
    alfa = (   eps ./ abs( delta_h(nr-1, :) )   ) .^ ( 1/(p+1) );
    alfa = min(alfa);


    % Proponowana korekta dlugosci kroku
    h_tmp = s*alfa*h;
```

```
    if s*alfa >= 1 % jesli h_tmp >= h
        if x(nr-1) + h > b
            y = y( [1:length(y)-1], : ); % pozbywamy sie ostatniego
(nadmiarowego) wiersza
            koniec = 1; % koniec calkowania
        else
            x(nr) = x(nr-1) + h; % nowa wartosc zmiennej niezaleznej
            h = min( [h_tmp   beta*h   b - x(nr-1)] ); % kolejny krok
            nr = nr + 1; % kolejna iteracja
        end
    else
        if h_tmp < h_min
            x = [];
            y = [];
            delta_h = [];
            error_code = 1; % niemożliwe rozwiazanie z zadana dokladnoscia
h_min
            koniec = 1; % koniec calkowania
        else
            h = h_tmp;
        end
    end

end

end
```

Listing skryptu wywołującego (Zad3.m):

```
% Zadanie 3.
% Metoda Rungego-Kutty RK4 ze zmiennym krokiem.


clear;
clc;


% Liczba rownan rozniczkowych (i zmiennych zaleznych zarazem)
m = 2;


% f - tablica komorkowa, zawierajaca uchwyty
% do prawych stron rownan rozniczkowych
f = cell(m,1);


% Wstawienie (do tablicy f) uchwytow
% do prawych stron rownan rozniczkowych
f{1} = @(x, y)( y(2) + y(1)*(0.9 - y(1)^2 - y(2)^2) );
f{2} = @(x, y)( -y(1) + y(2)*(0.9 - y(1)^2 - y(2)^2) );
```

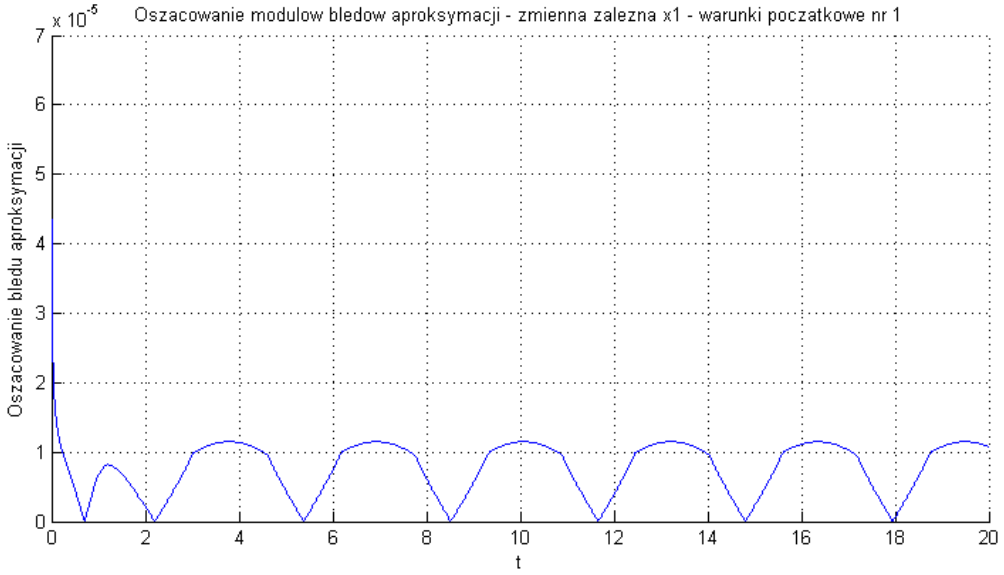
```
% Przedzial [a,b] zmiennej niezaleznej,
% dla ktorego bedziemy calkowali
% uklad rownan rozniczkowych
a = 0;
b = 20;


% Warunki poczatkowe (kolejne wiersze)
y0 = [10    8;
       0    9;
       8    0;
    1e-3 1e-3];


% Poczatkoe kroki calkowania (odpowiadajace kolejnym warunkom poczatkowym)
h0 = [0.01465 0.025 0.02 0.2];


% Minimalne kroki calkowania
h_min = [1e-5 5e-5 5e-5 5e-5];


% Dokladnosc wzgledna
eps_w = 1e-5;


% Dokladnosc bezwzgledna
eps_b = 1e-5;


% Liczba warunkow poczatkowych
L = 4;


% Numery zestawow warunkow poczatkowych
nr = 1:L;


% Czasy dzialania solwerow
T = zeros(L,1);


% Liczby iteracji
iter = zeros(L,1);


% Licznik wykresow
q = 1;


% Wyznaczenie rozwiazan
% dla roznych warunkow poczatkowych
for i=1:L

    % Obliczenie rozwiazan i bledow oszacowan
    tic;
```

```
    [x, y, delta_h, error_code] = RK4_var_h(f, m, a, b, h0(i), h_min(i),
y0(i,:), eps_w, eps_b);
    T(i) = toc;


    % Wyznaczenie liczby iteracji
    iter(i) = length(x);


    if error_code == 0 % pomyslne wykonanie metody

        % Wykres modulow bledow aproksymacji - zmienna zalezna x1 ("dobry"
krok h)
        figure(q);
        hold on;
        plot( x, abs( delta_h(:,1) ) );
        grid on;
        title( ['Oszacowanie modulow bledow aproksymacji - zmienna zalezna
x1 - warunki poczatkowe nr ', num2str(i)] );
        xlabel('t');
        ylabel('Oszacowanie bledu aproksymacji');
        hold off;
        q = q+1;


        % Wykres bledu aproksymacji - zmienna zalezna x2 ("dobry" krok h)
        figure(q);
        hold on;
        plot( x, abs( delta_h(:,2) ) );
        grid on;
        title( ['Oszacowanie modulow bledow aproksymacji - zmienna zalezna
x2 - warunki poczatkowe nr ', num2str(i)] );
        xlabel('t');
        ylabel('Oszacowanie bledu aproksymacji');
        hold off;
        q = q+1;


        % Wykres normy maksimum bledu aproksymacji ("dobry" krok h)
        delta_h_aggr = zeros( length(delta_h), 1 );
        for c=1:length(delta_h)
            delta_h_aggr(c) = norm( delta_h(c, :), Inf );
        end

        figure(q);
        hold on;
        plot( x, delta_h_aggr );
        grid on;
        title( ['Norma maksimum bledu aproksymacji - warunki poczatkowe nr
', num2str(i)] );
        xlabel('t');
        ylabel('Oszacowanie bledu aproksymacji');
        hold off;
        q = q+1;


        % Rozwiazanie przy pomocy polecenia ode45
        [x_ode, y_ode] = ode45(@right_sides,[a b], y0(i,:));
```

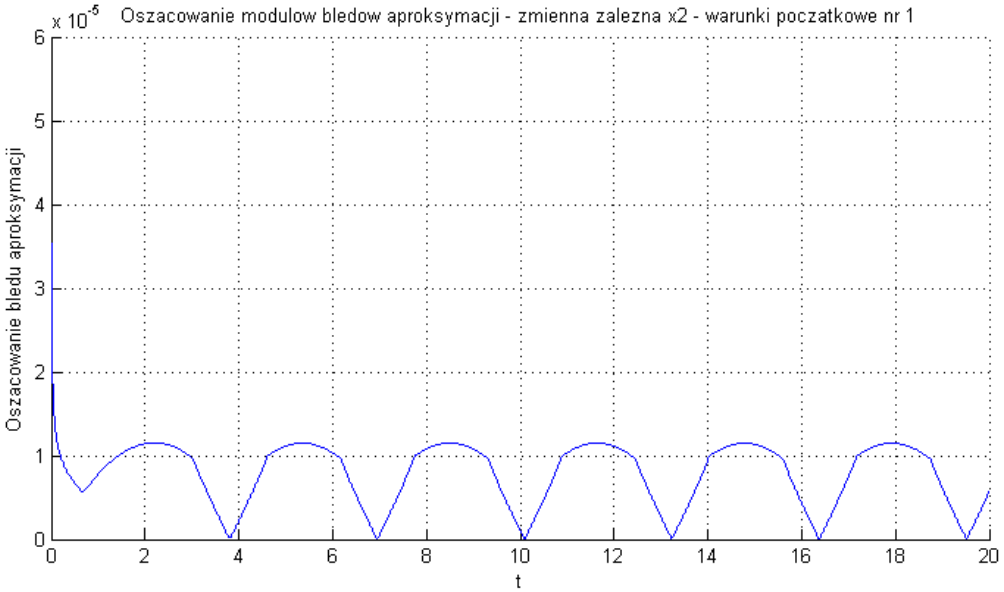
```
        % Wykresy porownawcze

        % Wykres fazowy
        figure(q);
        hold on;
        plot( y(:,1), y(:,2), y_ode(:,1), y_ode(:,2) );
        grid on;
        title( ['Porownanie rozwiazan - przestrzen fazowa - warunki
poczatkowe nr ', num2str(i)] );
        xlabel('x1');
        ylabel('x2');
        legend('RK4 var h', 'ode45', 'Location', 'North');
        hold off;
        q = q+1;


        % Zmienna zalezna x1
        figure(q);
        hold on;
        plot( x, y(:,1), x_ode, y_ode(:,1) );
        grid on;
        title( ['Porownanie rozwiazan - zmienna zalezna x1 - warunki
poczatkowe nr ', num2str(i)] );
        xlabel('t');
        ylabel('x1');
        legend('RK4 var h', 'ode45');
        hold off;
        q = q+1;


        % Zmienna zalezna x2
        figure(q);
        hold on;
        plot( x, y(:,2), x_ode, y_ode(:,2) );
        grid on;
        title( ['Porownanie rozwiazan - zmienna zalezna x2 - warunki
poczatkowe nr ', num2str(i)] );
        xlabel('t');
        ylabel('x2');
        legend('RK4 var h', 'ode45');
        hold off;
        q = q+1;

    else % metoda nie zakonczyla sie pomyslnie

        fprintf( 'Dla warunkoow poczatkowych nr %s, metoda zakonczyla sie
niepowodzeniem.\n', num2str(i) );
    end

end
```

Wykresy:

Warunki początkowe a):



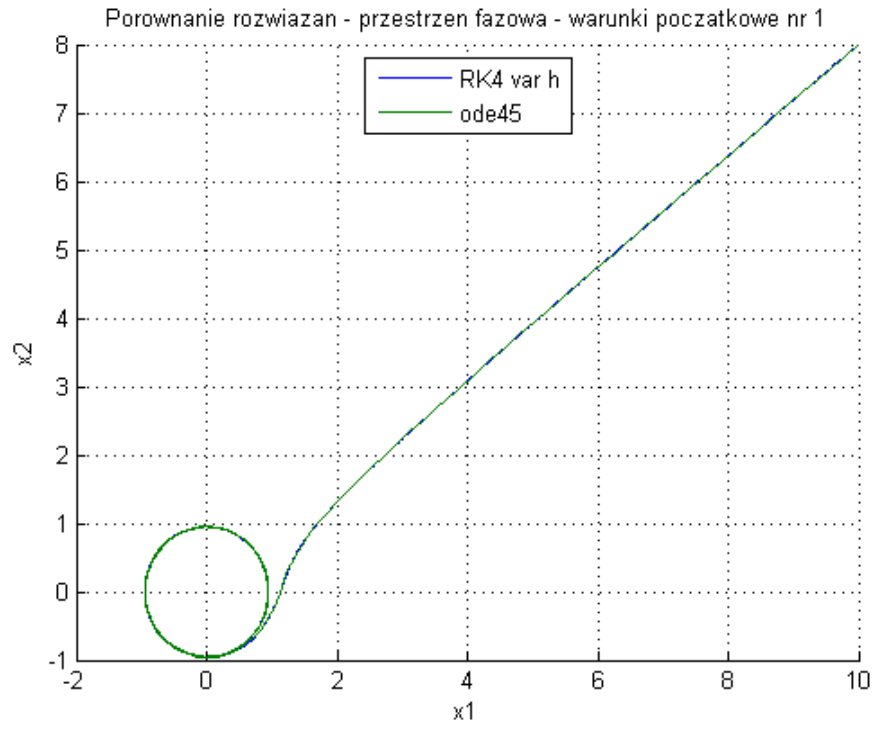
Wykres 65.



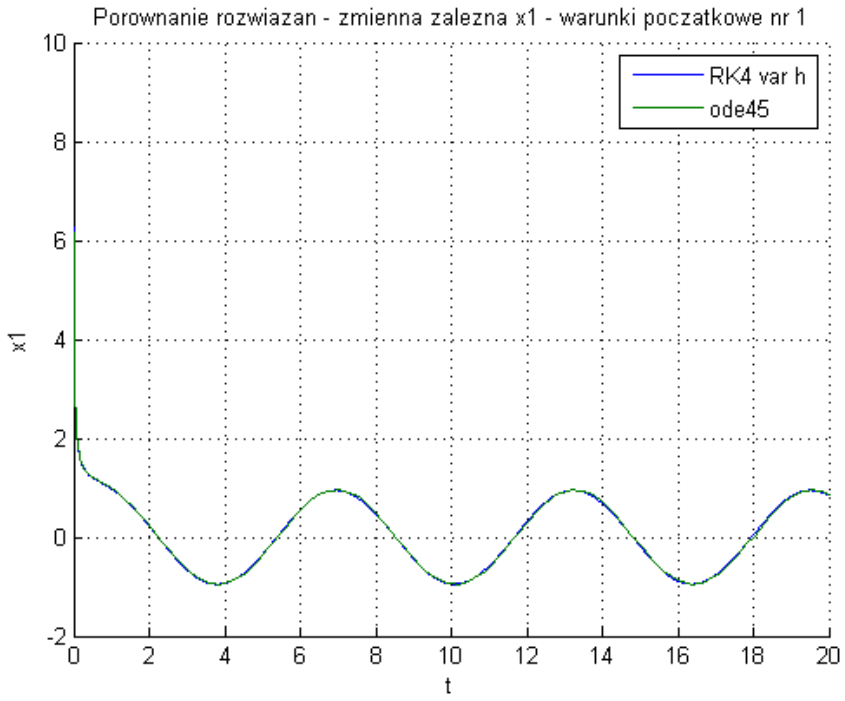
Wykres 66.

Wykres 67.

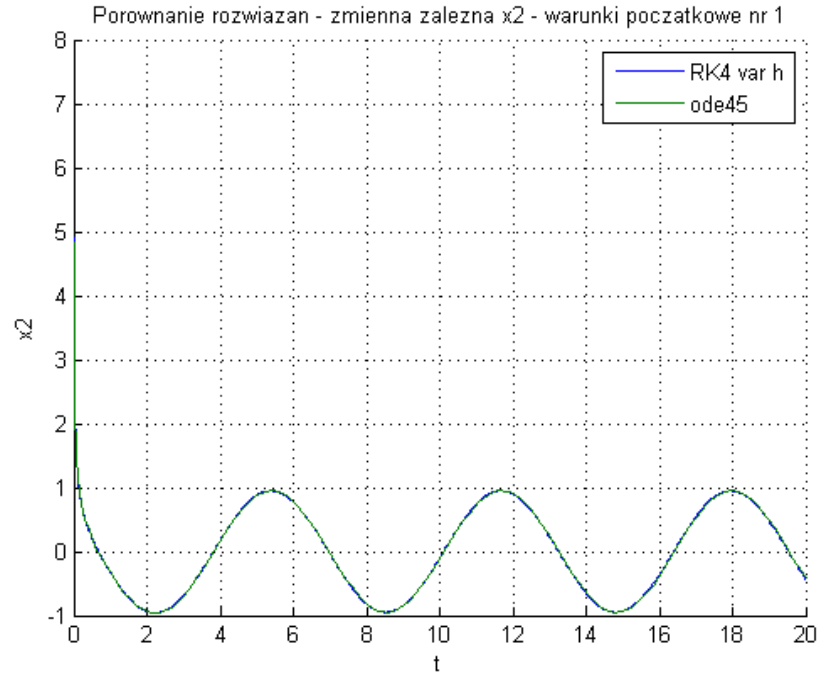
Zauważmy, że błędy aproksymacji są na niskim poziomie – nie przekraczają $5 \cdot 10^{-5}$. Ponadto, dość szybko zaczynają nie przekraczać $1.5 \cdot 10^{-5}$ (patrz wykresy 65., 66., 67.).



Wykres 68.



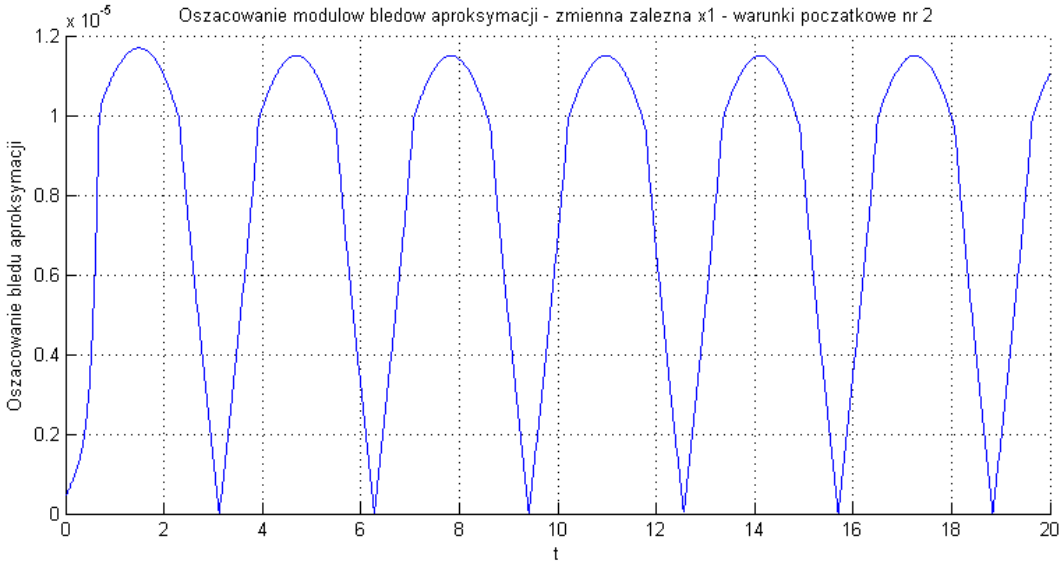
Wykres 69.



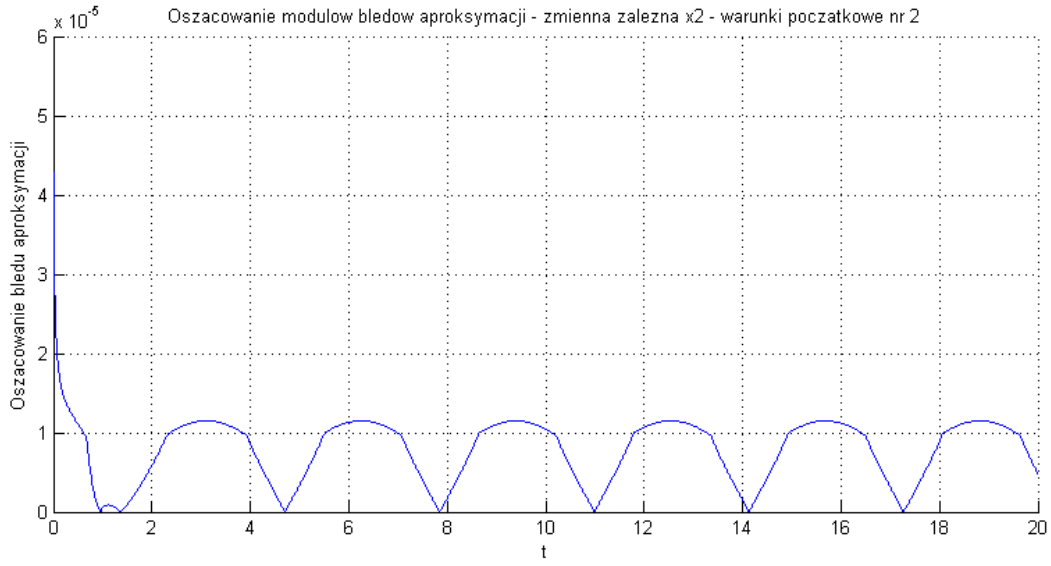
Wykres 70.

Uzyskane rozwiązanie (metodą RK4 ze zmiennym krokiem) bardzo dobrze oddaje rozwiązanie uzyskane dzięki poleceniu ode45 języka MATLAB (wykresy 68., 69., 70.).

Warunki początkowe b):



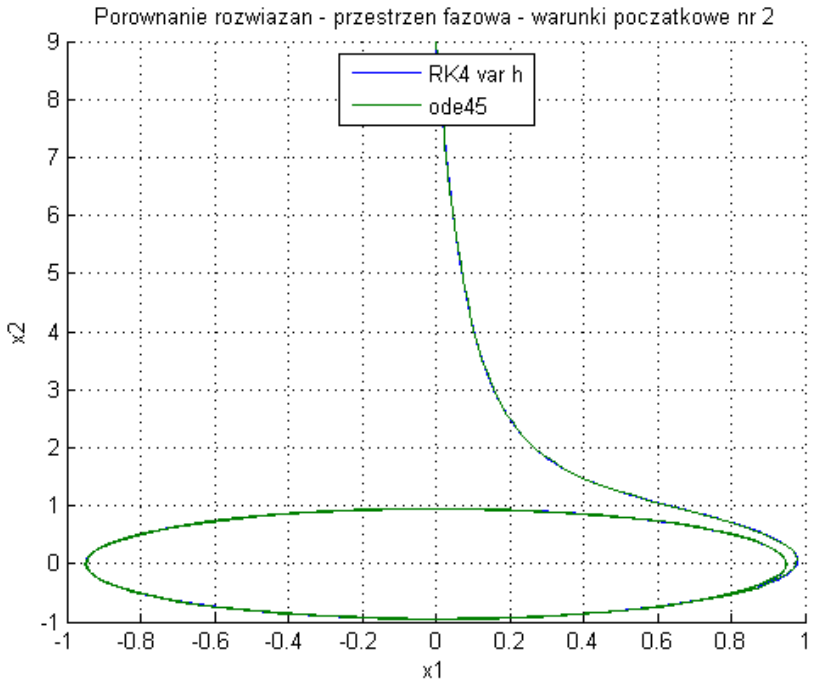
Wykres 71.



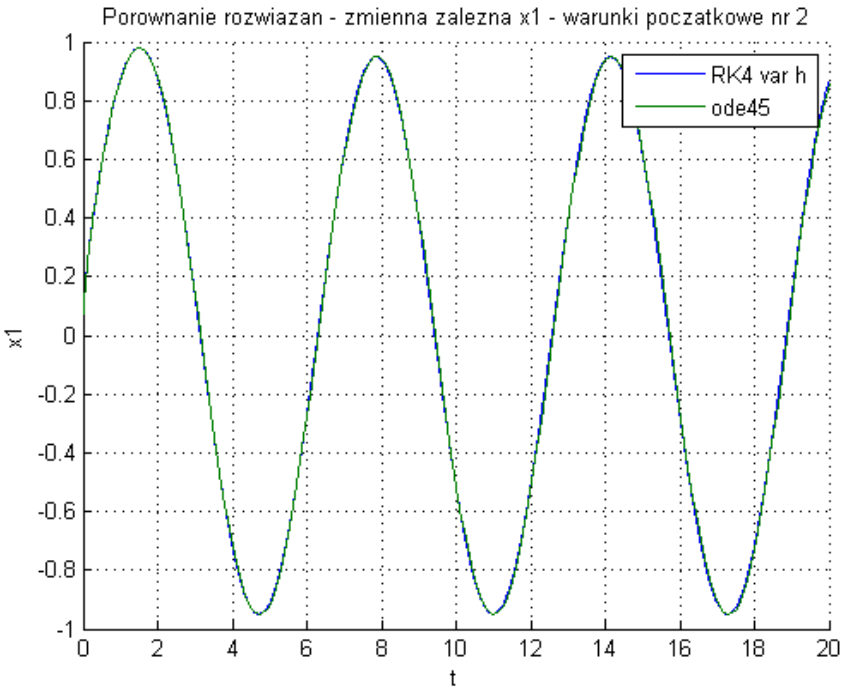
Wykres 72.

Wykres 73.

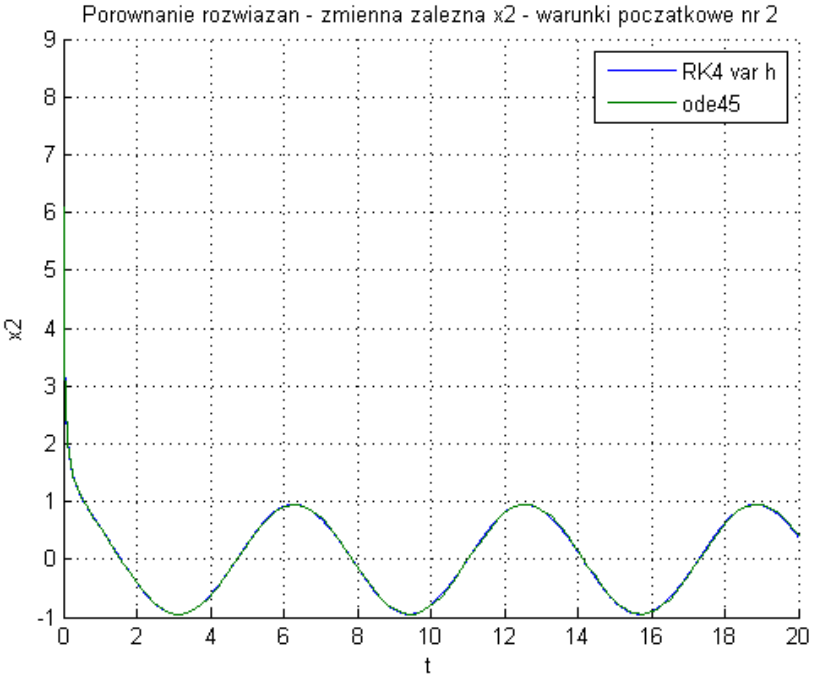
Zauważmy, że błędy aproksymacji są na niskim poziomie – nie przekraczają $5 \cdot 10^{-5}$. Ponadto, dość szybko zaczynają nie przekraczać $1.5 \cdot 10^{-5}$ (patrz wykresy 71., 72., 73.).



Wykres 74.



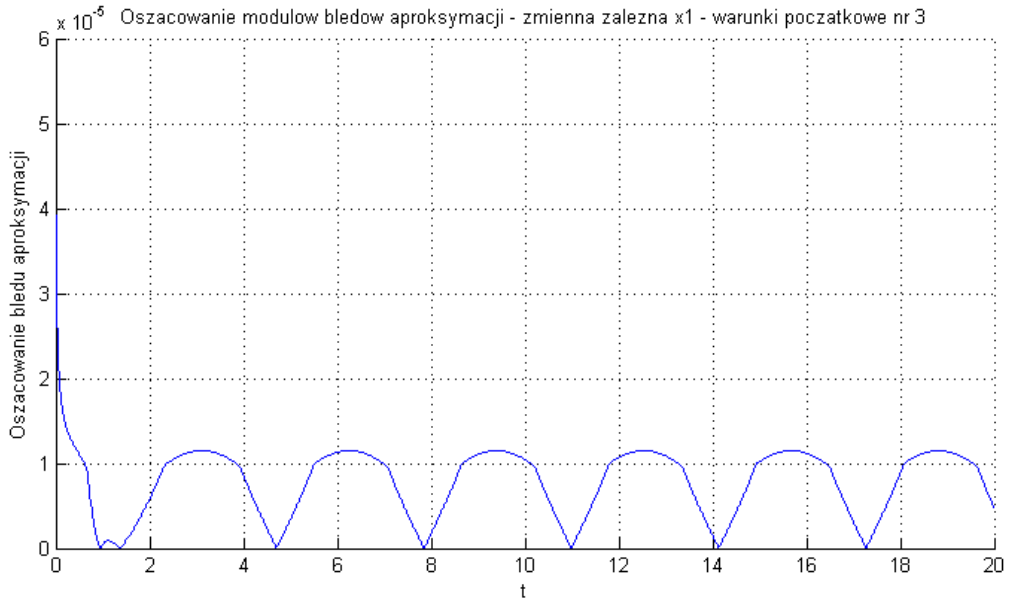
Wykres 75.



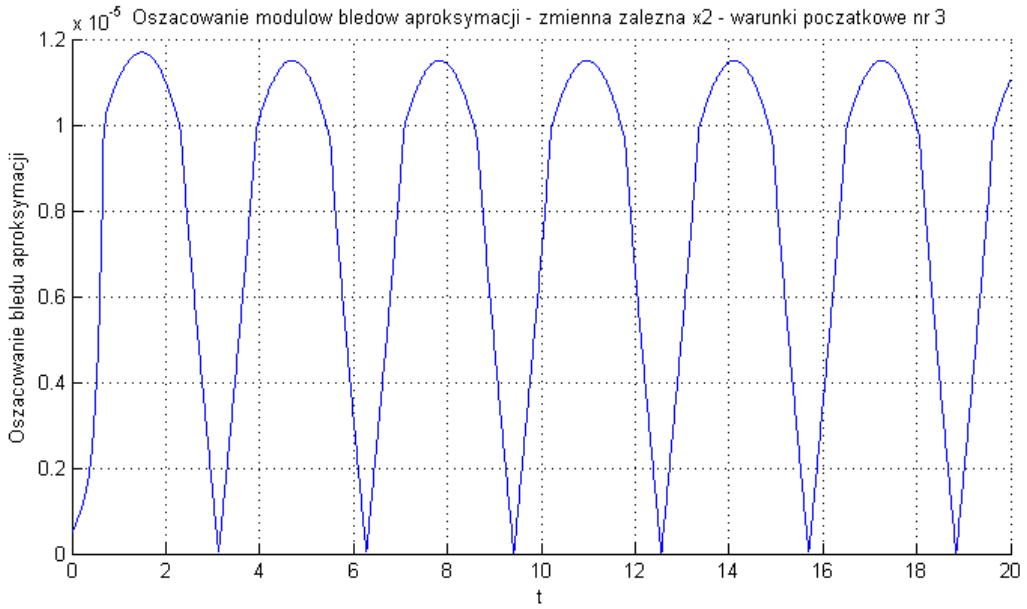
Wykres 76.

Uzyskane rozwiązanie (metodą RK4 ze zmiennym krokiem) bardzo dobrze oddaje rozwiązanie uzyskane dzięki poleceniu ode45 języka MATLAB (wykresy 74., 75., 76.).

Warunki początkowe c):



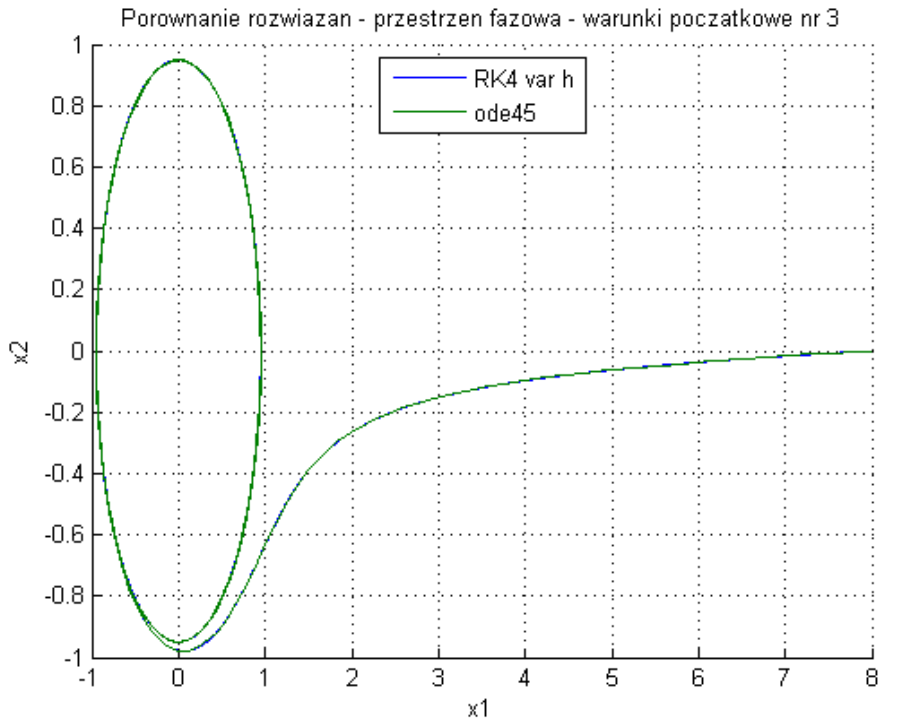
Wykres 77.



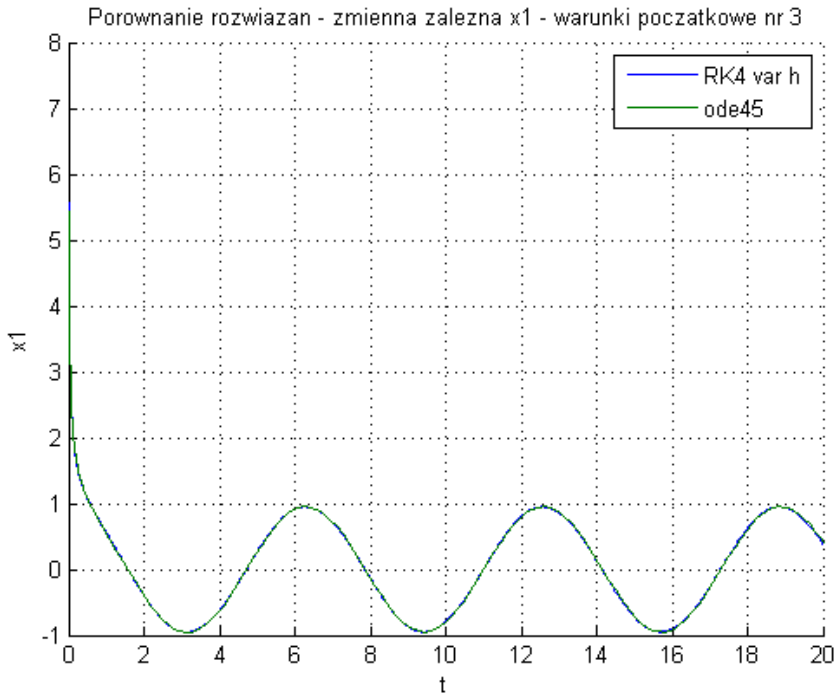
Wykres 78.

Wykres 79.

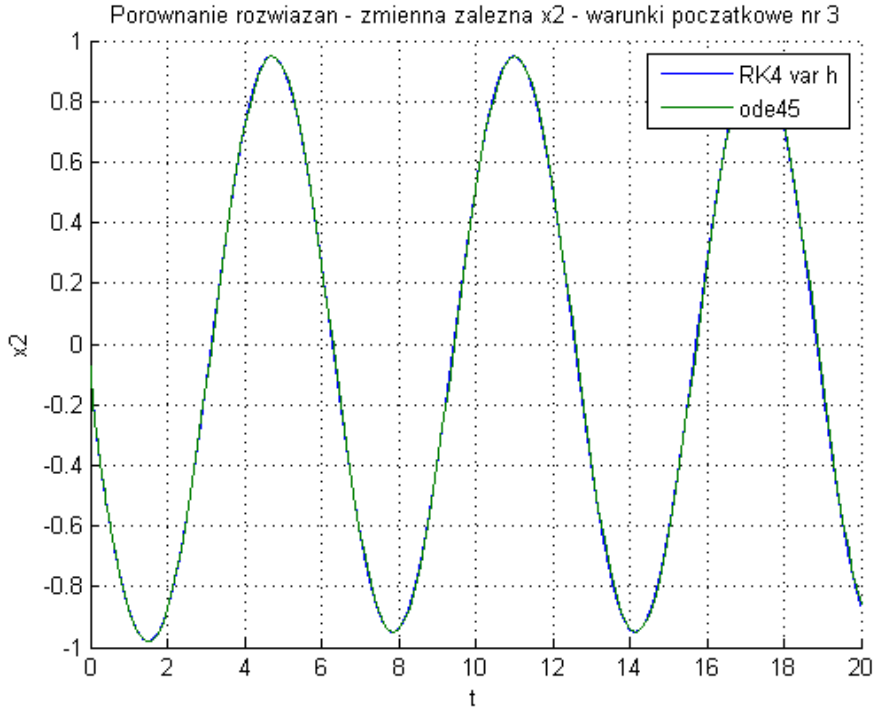
Zauważmy, że błędy aproksymacji są na niskim poziomie – nie przekraczają $4 \cdot 10^{-5}$. Ponadto, dość szybko zaczynają nie przekraczać $1.5 \cdot 10^{-5}$ (patrz wykresy 77., 78., 79.).



Wykres 80.



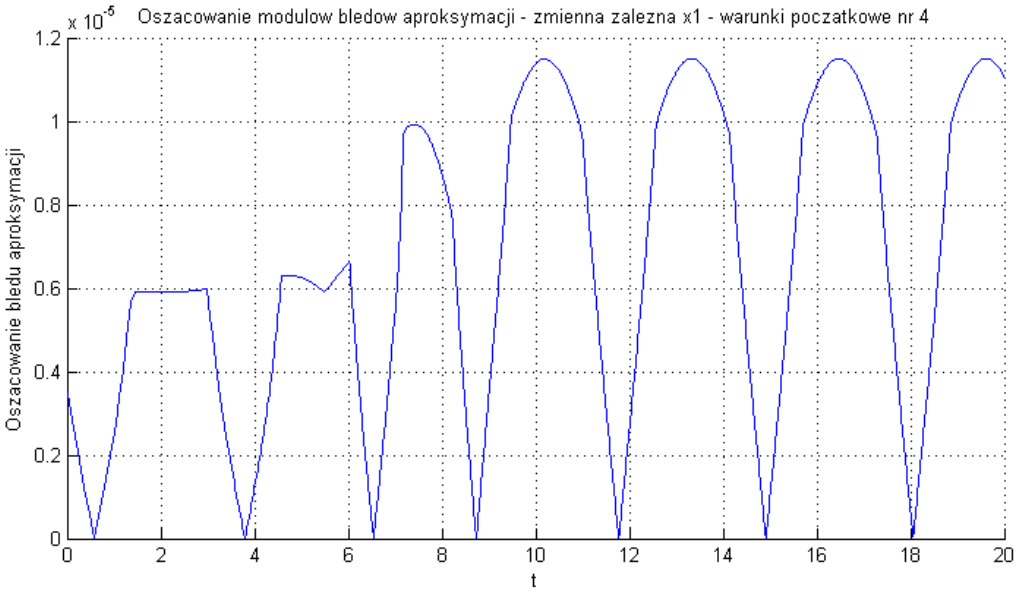
Wykres 81.



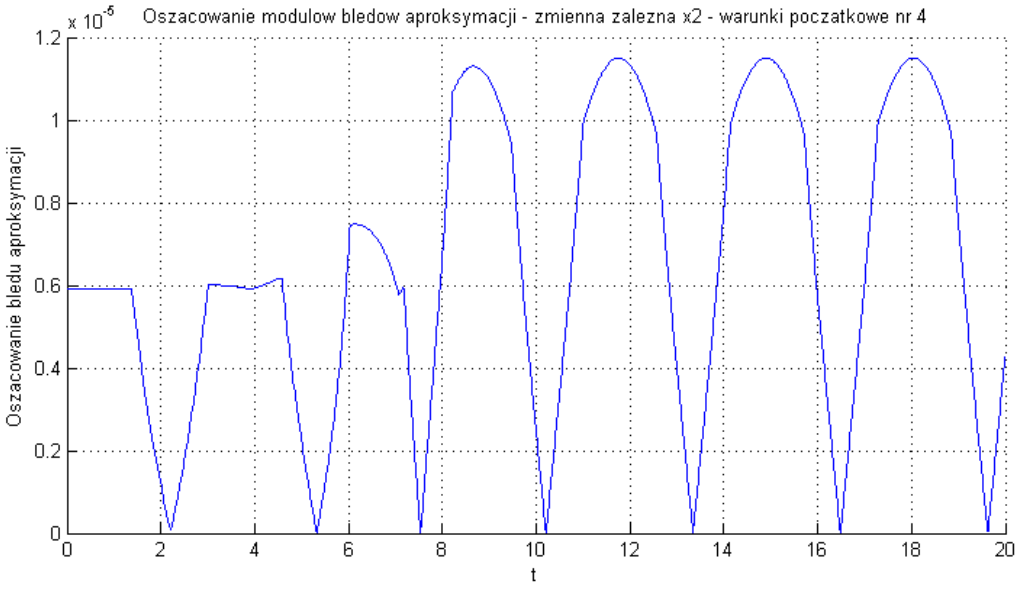
Wykres 82.

Uzyskane rozwiązanie (metodą RK4 ze zmiennym krokiem) bardzo dobrze oddaje rozwiązanie uzyskane dzięki poleceniu ode45 języka MATLAB (wykresy 80., 81., 82.).

Warunki początkowe d):



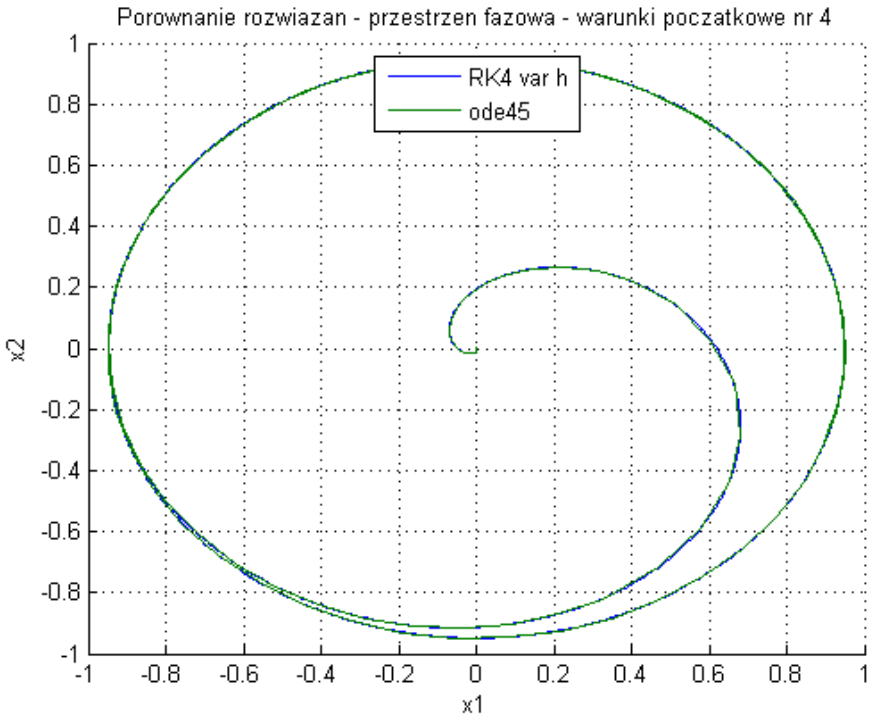
Wykres 83.



Wykres 84.

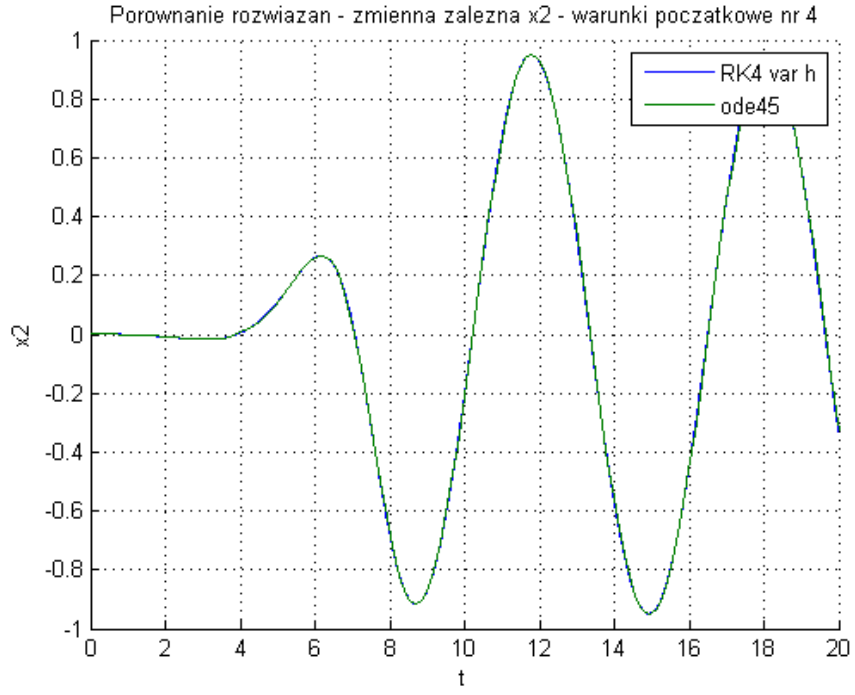
Wykres 85.

Zauważmy, że błędy aproksymacji są na niskim poziomie i nie przekraczają $1.2 \cdot 10^{-5}$; ponadto, od pewnego miejsca, błędy te zmieniają się okresowo (patrz wykresy 83., 84., 85.).



Wykres 86.

Wykres 87.



Wykres 88.

Uzyskane rozwiązanie (metodą RK4 ze zmiennym krokiem) bardzo dobrze oddaje rozwiązanie uzyskane dzięki poleceniu ode45 języka MATLAB (wykresy 86., 87., 88.).

Tabele porównawcze metod:

Czasy działania solwerów (w sekundach):

Zestaw/Metoda	RK4 (stały krok)	Adamsa 4. rzędu (stały krok)	RK4 (zmienny krok)
1	0,5867414	0,8405182	11,3257969
2	0,2226538	0,7271874	12,0634473
3	0,3005169	0,5686891	11,497943
4	0,0260178	0,0145989	11,067372

Tabela 1.

Liczby iteracji solwerów:

Zestaw/Metoda	RK4 (stały krok)	Adamsa 4. rzędu (stały krok)	RK4 (zmienny krok)
1	1429	5001	2565
2	801	5001	2465
3	1001	4001	2436
4	101	101	1677

Tabela 2.

Dobrane długości kroku dla metod stałokrokowych:

Zestaw/Metoda	RK4 (stały krok)	Adamsa 4. rzędu (stały krok)
1	0,014	0,004
2	0,025	0,004
3	0,02	0,005
4	0,2	0,2

Tabela 3.

Komentarz końcowy:

W zależności od zestawu warunków początkowych,
uzyskiwaliśmy inne przebiegi rozwiązania.

Jeśli chodzi o metody ze stałym krokiem, to mniejsze (z reguły) kroki stosowaliśmy w przypadku metody predyktor-korektor Adamsa.
Oczywiście, skutkowało to większą liczbą iteracji oraz dłuższym czasem obliczeń.

Jeśli zależy nam na bardzo dobrej jakości rozwiązania (tj. bardzo małych błędach aproksymacji), to należy wybrać metodę RK4 ze zmiennym krokiem (na tle rozwiązania uzyskanego za pomocą polecenia *ode45*, uzyskane przez nas rozwiązania prezentowały się znakomicie). Niestety, czas obliczeń jest dużo większy niż dla pozostałych metod (każda iteracja wiąże się z potencjalnie sporą liczbą modyfikacji kroku, a co za tym idzie, wzrasta nakład obliczeń).

Jeśli natomiast chcemy uzyskać rozwiązanie stosunkowo dobrej jakości w krótkim czasie, to dobrym wyborem jest metoda RK4 ze stałym krokiem – uzyskujemy sensowne rozwiązania przy większych krokach i mniejszej liczbie iteracji.

Warto zauważyć, że metoda predyktor-korektor Adamsa dawała dokładniejsze rezultaty niż w metodzie RK4 ze stałym krokiem, niemniej iteracji było więcej i czas obliczeń był (z reguły) dłuższy. Ze względu na to, że czasy obliczeń były mimo wszystko dość krótkie, to kierując się kryterium jakości do czasu, wybrałbym metodę predyktor-korektor Adamsa czwartego rzędu ze stałym krokiem.

Bibliografia:

[1] Piotr Tatjewski „Metody Numeryczne”, Oficyna Wydawnicza Politechniki Warszawskiej, Warszawa 2013