Rozwiązywanie układów równań różniczkowych zwyczajnych

Autor: Wiktor Murawski

Przedmiot: Modelowanie matematyczne
Prowadzący: dr inż. Jakub Wagner

Politechnika Warszawska
Wydział Matematyki i Nauk Informacyjnych

Oświadczam, że niniejsza praca, stanowiąca podstawę do uznania osiągnięcia efektów uczenia się z przedmiotu Modelowanie matematyczne, została wykonana przeze mnie samodzielnie.

Warszawa
2 grudnia 2024

Spis treści

1 Lista Symboli i Akronimów

URRZ	układ równań różniczkowych zwyczajnych
t	zmienna skalarna, czas
$y_1(t), y_2(t), x(t)$	funkcje w dziedzinie czasu t
$\mathbf{A}$	macierz 2×2 współczynników URRZ
$\mathbf{b}$	pionowy wektor współczynników przy $x(t)$
$\mathbf{y}(t)$	pionowy wektor zawierający wartości $y_1(t)$ i $y_2(t)$, $\mathbf{y}(t) = \begin{bmatrix} y_1(t) \\ y_2(t) \end{bmatrix}$
h	wartość kroku całkowania
t_n	wartość czasu, $t_n = t_0 + (n-1)h$, $n \in \mathbb{Z}^+$
$\mathbf{y}_n$	$\mathbf{y}(t_n)$
$\mathbf{f}(t_n, \mathbf{y}_n)$	funkcja określona przez URRZ: $\left.\frac{d\mathbf{y}(t)}{dt}\right\|_{t=t_n} = \mathbf{f}(t_n, \mathbf{y}_n)$
$N(h)$	zależna od kroku całkowania liczba punktów rozwiązania
h_{min}	najmniejszy badany krok całkowania
h_{max}	największy badany krok całkowania
$\dot{y}_1(t), \dot{y}_2(t)$	dokładne rozwiązanie URRZ
$\hat{y}_1(t,h), \hat{y}_2(t,h)$	przybliżone rozwiązanie URRZ dla kroku całkowania h
$c_i, a_{i,j}, w_i$	współczynniki w tabeli Butchera; $i, j \in \{1, 2, 3\}$
$\delta_1(h), \delta_2(h)$	zagregowane błędy względne dla kroku całkowania h

2 Wprowadzenie

Dany jest następujący układ równań różniczkowych zwyczajnych (URRZ):

$$\left.\begin{aligned}\frac{dy_1(t)}{dt} &= -\frac{14}{3}y_1(t) - \frac{2}{3}y_2(t) + x(t)\\ \frac{dy_2(t)}{dt} &= \frac{2}{3}y_1(t) - \frac{19}{3}y_2(t) + x(t)\end{aligned}\right\} \text{ dla } t \in [0,8], \text{ w którym } x(t) = \exp(-t)\sin(t) \tag{1}$$

W celu rozwiązania URRZ przedstawionego w (1) dla zerowych warunków początkowych, tj. $y_1(0) = y_2(0) = 0$:

1. wyznaczono dokładne rozwiązanie URRZ za pomocą procedury `dsolve` (*MATLAB Symbolic Toolbox*)

2. zastosowano procedurę `ode45` (*MATLAB*), będącą zaawansowaną implementacją metody Rungego-Kutty czwartego rzędu z adaptacyjnym krokiem czasowym

3. zaimplementowano oraz zastosowano trzy inne metody dyskretne:

   - metodę 1. zdefiniowaną wzorem $\mathbf{y}_n = \mathbf{y}_{n-1} + h\mathbf{f}\left(t_{n-1} + \frac{h}{2}, \mathbf{y}_{n-1} + \frac{h}{2}\mathbf{f}\left(t_{n-1}, \mathbf{y}_{n-1}\right)\right)$
   - metodę 2. zdefiniowaną wzorem $\mathbf{y}_n = \mathbf{y}_{n-2} + h\Big[\mathbf{f}\left(t_n, \mathbf{y}_n\right) + \mathbf{f}\left(t_{n-2}, \mathbf{y}_{n-2}\right)\Big]$
   - metodę 3. zdefiniowaną wzorem $\mathbf{y}_n = \mathbf{y}_{n-1} + h\sum\limits_{k=1}^{3} w_k\mathbf{f}_k$

     gdzie $\mathbf{f}_k = \mathbf{f}\left(t_{n-1} + c_k h, \mathbf{y}_{n-1} + h\sum\limits_{\kappa=1}^{3} a_{k,\kappa}\mathbf{f}_\kappa\right)$

     a współczynniki przyjmują wartości przedstawione w poniższej tabeli Butchera:

$$\begin{array}{c|ccc} c_1 & a_{1,1} & a_{1,2} & a_{1,3}\\ c_2 & a_{2,1} & a_{2,2} & a_{2,3}\\ c_3 & a_{3,1} & a_{3,2} & a_{3,3}\\ \hline & w_1 & w_2 & w_3\end{array} \quad = \quad \begin{array}{c|ccc} 0 & \frac{1}{6} & -\frac{1}{6} & 0\\ \frac{1}{2} & \frac{1}{6} & \frac{1}{3} & 0\\ 1 & \frac{1}{6} & \frac{5}{6} & 0\\ \hline & \frac{1}{6} & \frac{2}{3} & \frac{1}{6}\end{array}$$

3 Metodyka i Wyniki Doświadczeń

3.1 Procedura dsolve

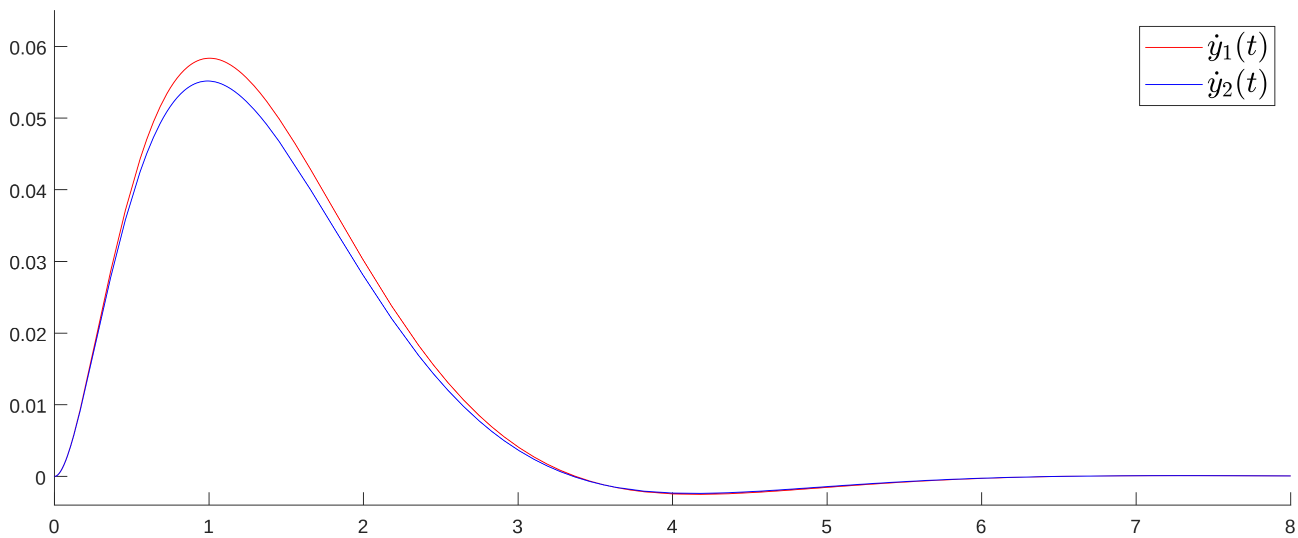
Za pomocą poniższego kodu korzystającego z procedury `dsolve` wyznaczono rozwiązanie (1)

```
function [y1_exact, y2_exact] = solve_using_dsolve(x)

syms t y1(t) y2(t)

eq1 = diff(y1,t,1) == -14/3*y1(t) - 2/3*y2(t) + x(t);
eq2 = diff(y2,t,1) == 2/3*y1(t) - 19/3*y2(t) + x(t);
eqns = [eq1; eq2];

cond1 = y1(0) == 0;
cond2 = y2(0) == 0;
conds = [cond1; cond2];

sol = dsolve(eqns,conds);
y1_exact = sol.y1;
y2_exact = sol.y2;

end % function
```

Uzyskano

$$\dot{y}_1(t) = -\frac{1}{2}\exp(-6t)\left(\frac{\exp(5t)\,(\cos(t) - 5\sin(t))}{39} - \frac{1}{39}\right) - 2\exp(-5t)\left(\frac{\exp(4t)\,(\cos(t) - 4\sin(t))}{51} - \frac{1}{51}\right)$$
$$\dot{y}_2(t) = -\exp(-6t)\left(\frac{\exp(5t)\,(\cos(t) - 5\sin(t))}{39} - \frac{1}{39}\right) - \exp(-5t)\left(\frac{\exp(4t)\,(\cos(t) - 4\sin(t))}{51} - \frac{1}{51}\right)$$

Wykresy otrzymanych funkcji przedstawione są na rysunku 1.



Rysunek 1: Wykresy $\dot{y}_1(t)$, $\dot{y}_2(t)$ wyznaczonych analitycznie procedurą `dsolve`

3.2 Przekształcenie URRZ do postaci macierzowej

Niech $\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}$, wtedy $\mathbf{y}' = \begin{bmatrix} \dfrac{dy_2}{dt} \\ \dfrac{dy_1}{dt} \end{bmatrix}$, oraz niech $\mathbf{f}\left(t, \mathbf{y}\right) = \mathbf{y}'$.

Wtedy URRZ (1) można przedstawić następująco:

$$\mathbf{f}\left(t, \mathbf{y}\right) = \mathbf{A}\mathbf{y} + \mathbf{b}x(t) \tag{2}$$

gdzie

$$\mathbf{A} = \begin{bmatrix} -\dfrac{14}{3} & -\dfrac{2}{3} \\ \dfrac{2}{3} & -\dfrac{19}{3} \end{bmatrix} \qquad \mathbf{b} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

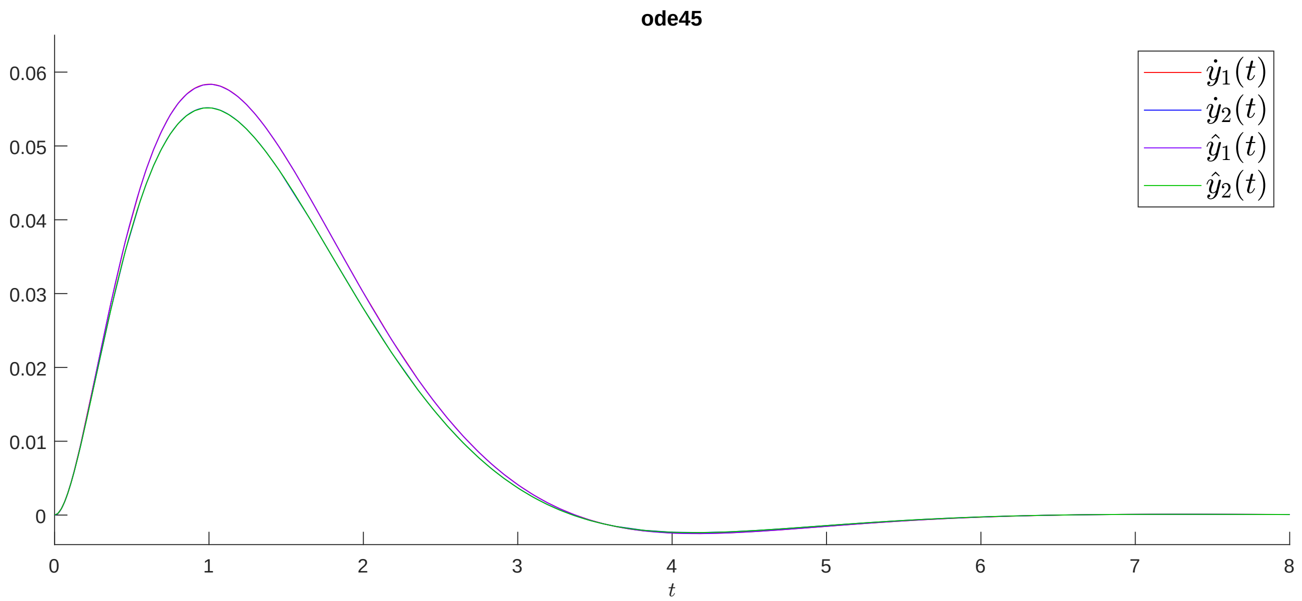
Funkcja $\mathbf{f}\left(t_n, \mathbf{y}_n\right)$ określona jest przez $\left.\dfrac{d\mathbf{y}(t)}{dt}\right|_{t=t_n} = \mathbf{f}(t_n, \mathbf{y}_n)$.

3.3 Procedura ode45

Za pomocą poniższego kodu korzystającego z procedury `ode45` wyznaczono rozwiązanie (1)

```
% Procedura ode45
% Funkcja f
f = @(t,y) A*y + b*x(t);
% Przedział czasowy
tspan = [0, 8];
% Warunki początkowe
y0 = [0; 0];
% Wywołanie procedury
[T_ode45,y_ode45] = ode45(f,tspan,y0);
```

Wykresy $\hat{y}_1(t)$ oraz $\hat{y}_2(t)$ otrzymanych procedurą `ode45` są, wraz z wynikami dokładnymi, przedstawione na rysunku 2.



Rysunek 2: Wykresy $\dot{y}_1(t)$, $\dot{y}_2(t)$ oraz $\hat{y}_1(t)$, $\hat{y}_2(t)$ otrzymanych procedurą `ode45`

3.4 Metoda 1

Metoda 1. zdefiniowana jest wzorem

$$\mathbf{y}_n = \mathbf{y}_{n-1} + h\mathbf{f}\left(t_{n-1} + \frac{h}{2}, \mathbf{y}_{n-1} + \frac{h}{2}\mathbf{f}\left(t_{n-1}, \mathbf{y}_{n-1}\right)\right) \tag{3}$$

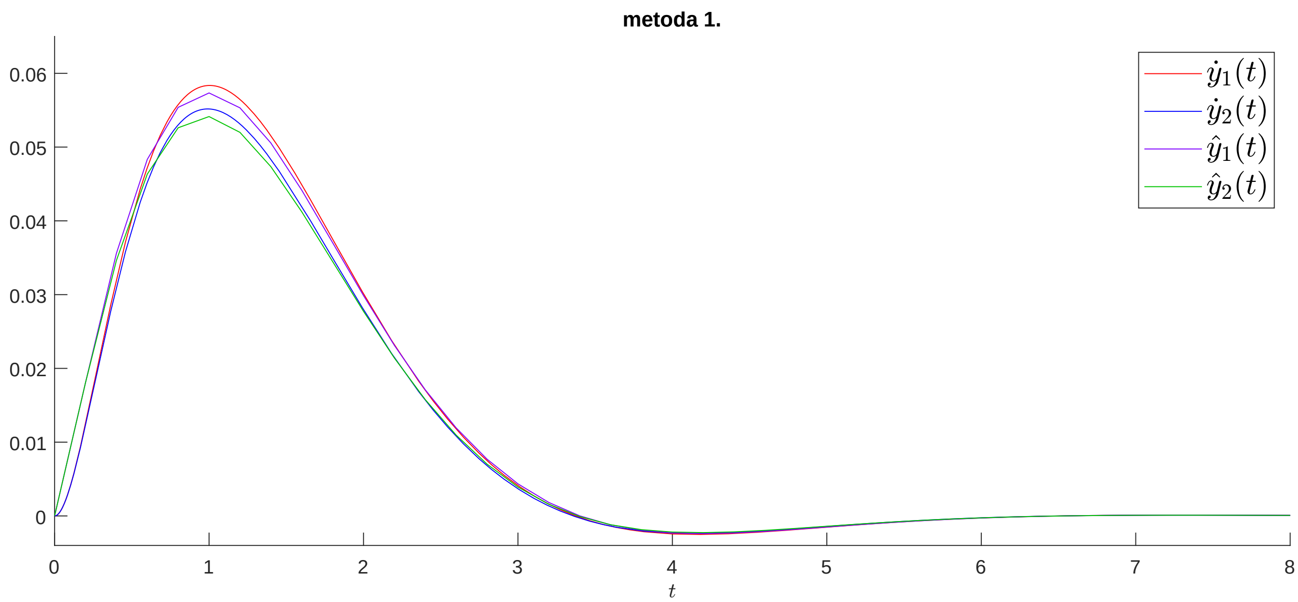
Podstawiając (2) do (3) otrzymano

$$\mathbf{y}_n = \mathbf{y}_{n-1} + h\left(A\left(\mathbf{y}_{n-1} + \frac{h}{2}\left(A\mathbf{y}_{n-1} + \mathbf{b}x(t_{n-1})\right)\right) + \mathbf{b}x\left(t_{n-1} + \frac{h}{2}\right)\right)$$

Kod zawierający implementację metody 1.:

```
function y = metoda1(A,b,x,h,N,t)

y = zeros(2,N);

for n = 2:N
  y(:,n) = y(:,n-1) + ...
    h*(A*(y(:,n-1) + h/2*(A*y(:,n-1) + b*x(t(n-1)))) + b*x(t(n-1) + h/2));
end % for n

end % function
```

Wykresy $\hat{y}_1(t)$ oraz $\hat{y}_2(t)$ otrzymanych metodą 1. są, wraz z wynikami dokładnymi, przedstawione na rysunku 3.



Rysunek 3: Wykresy $\dot{y}_1(t)$, $\dot{y}_2(t)$ oraz $\hat{y}_1(t)$, $\hat{y}_2(t)$ otrzymanych metodą 1. dla kroku $h = 0.2$

3.5 Metoda 2

Metoda 2. jest metodą niejawną zdefiniowaną wzorem

$$\mathbf{y}_n = \mathbf{y}_{n-2} + h\Big[\mathbf{f}\,(t_n, \mathbf{y}_n) + \mathbf{f}\,(t_{n-2}, \mathbf{y}_{n-2})\Big] \tag{4}$$

Podstawiając (2) do (4) otrzymano

$$\mathbf{y}_n = \mathbf{y}_{n-2} + h\Big[\big(\mathbf{A}\mathbf{y}_n + \mathbf{b}x(t_n)\big) + \big(\mathbf{A}\mathbf{y}_{n-2} + \mathbf{b}x(t_{n-2})\big)\Big]$$

Po przekształceniach

$$\mathbf{y}_n = \Big(\mathbf{I} - h\mathbf{A}\Big)^{-1}\Big(\mathbf{y}_{n-2} + h\mathbf{A}\mathbf{y}_{n-2} + h\mathbf{b}x(t_n) + h\mathbf{b}x(t_{n-2})\Big)$$

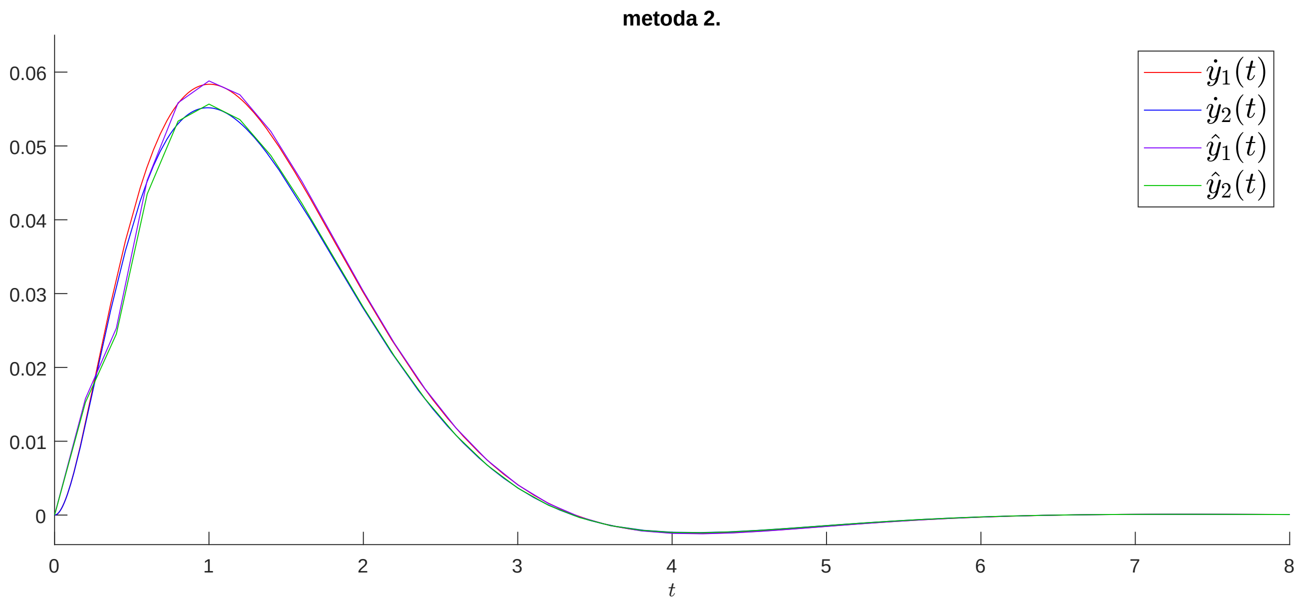
W celu wyznaczenia $\mathbf{y}_2$ zastosowano niejawną metodę Eulera ($\mathbf{y}_1$ to zerowy warunek początkowy)

$$\mathbf{y}_2 = \Big(\mathbf{I} - h\mathbf{A}\Big)^{-1}\Big(\mathbf{y}_1 + h\mathbf{b}x(t_2)\Big)$$

Kod zawierający implementację metody 2.:

```
function y = metoda2(A,b,x,h,N,t)

I = eye(2);
y = zeros(2,N);

% Niejawna metoda Eulera
y(:,2) = (I - h*A) \ (y(:,1) + h*b*x(t(2)));

for n = 3:N
  y(:,n) = (I - h*A) \ ...
      (y(:,n-2) + h*A*y(:,n-2) + h*b*x(t(n)) + h*b*x(t(n-2)));
end % for n

end % function
```

Wykresy $\hat{y}_1(t)$ oraz $\hat{y}_2(t)$ otrzymanych metodą 2. są, wraz z wynikami dokładnymi, przedstawione na rysunku 4.



Rysunek 4: Wykresy $\dot{y}_1(t)$, $\dot{y}_2(t)$ oraz $\hat{y}_1(t)$, $\hat{y}_2(t)$ otrzymanych metodą 2. dla kroku $h = 0.2$

3.6 Metoda 3

Metoda 3. zdefiniowana jest wzorem

$$\mathbf{y}_n = \mathbf{y}_{n-1} + h \sum_{k=1}^{3} w_k \mathbf{f}_k \tag{5}$$

Po rozpisaniu sumy

$$\mathbf{y}_n = \mathbf{y}_{n-1} + h\Big(w_1\mathbf{f}_1 + w_2\mathbf{f}_2 + w_3\mathbf{f}_3\Big) \tag{6}$$

gdzie

$$\mathbf{f}_k = \mathbf{f}\left(t_{n-1} + c_k h, \mathbf{y}_{n-1} + h \sum_{\kappa=1}^{3} a_{k,\kappa}\mathbf{f}_\kappa\right) \tag{7}$$

a współczynniki przyjmują wartości przedstawione w poniższej tabeli Butchera:

$$\begin{array}{c|ccc} c_1 & a_{1,1} & a_{1,2} & a_{1,3} \\ c_2 & a_{2,1} & a_{2,2} & a_{2,3} \\ c_3 & a_{3,1} & a_{3,2} & a_{3,3} \\ \hline & w_1 & w_2 & w_3 \end{array} \quad = \quad \begin{array}{c|ccc} 0 & \frac{1}{6} & -\frac{1}{6} & 0 \\ \dfrac{1}{2} & \frac{1}{6} & \frac{1}{3} & 0 \\ 1 & \frac{1}{6} & \frac{5}{6} & 0 \\ \hline & \frac{1}{6} & \frac{2}{3} & \frac{1}{6} \end{array}$$

Z (7) otrzymano

$$\begin{cases} \mathbf{f}_1 = \mathbf{A}\big(\mathbf{y}_{n-1} + ha_{1,1}\mathbf{f}_1 + ha_{1,2}\mathbf{f}_2 + ha_{1,3}\mathbf{f}_3\big) + \mathbf{b}x\big(t_{n-1} + c_1 h\big) \\ \mathbf{f}_2 = \mathbf{A}\big(\mathbf{y}_{n-1} + ha_{2,1}\mathbf{f}_1 + ha_{2,2}\mathbf{f}_2 + ha_{2,3}\mathbf{f}_3\big) + \mathbf{b}x\big(t_{n-1} + c_2 h\big) \\ \mathbf{f}_3 = \mathbf{A}\big(\mathbf{y}_{n-1} + ha_{3,1}\mathbf{f}_1 + ha_{3,2}\mathbf{f}_2 + ha_{3,3}\mathbf{f}_3\big) + \mathbf{b}x\big(t_{n-1} + c_3 h\big) \end{cases} \tag{8}$$

Przekształcając (8) otrzymano

$$\begin{cases} \big(\mathbf{I} - ha_{1,1}\mathbf{A}\big)\mathbf{f}_1 - ha_{1,2}\mathbf{A}\mathbf{f}_2 - ha_{1,3}\mathbf{A}\mathbf{f}_3 = \mathbf{A}\mathbf{y}_{n-1} + \mathbf{b}x\big(t_{n-1} + c_1 h\big) \\ -ha_{2,1}\mathbf{A}\mathbf{f}_1 + \big(\mathbf{I} - ha_{2,2}\mathbf{A}\big)\mathbf{f}_2 - ha_{1,3}\mathbf{A}\mathbf{f}_3 = \mathbf{A}\mathbf{y}_{n-1} + \mathbf{b}x\big(t_{n-1} + c_2 h\big) \\ -ha_{3,1}\mathbf{A}\mathbf{f}_1 - ha_{3,2}\mathbf{A}\mathbf{f}_2 + \big(\mathbf{I} - ha_{3,3}\mathbf{A}\big)\mathbf{f}_3 = \mathbf{A}\mathbf{y}_{n-1} + \mathbf{b}x\big(t_{n-1} + c_3 h\big) \end{cases} \tag{9}$$

Układ (9) można przedstawić w następującej postaci:

$$\mathbf{L}\mathbf{g} = \mathbf{p} \tag{10}$$

gdzie

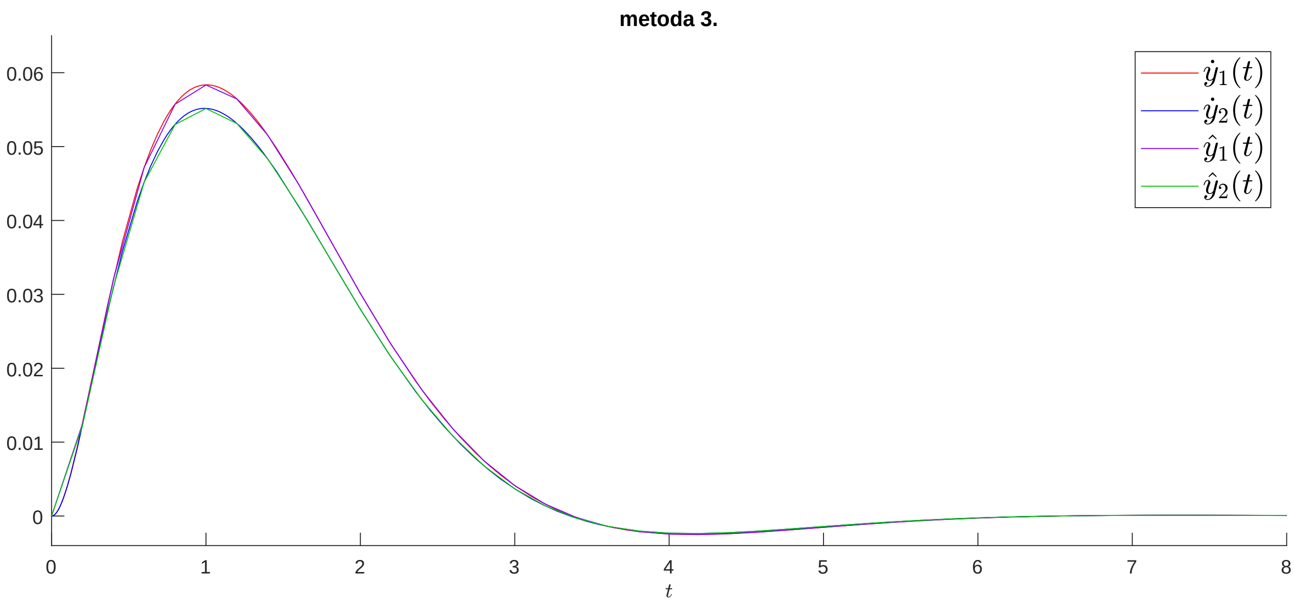
$$\mathbf{g} = \begin{bmatrix} \mathbf{f}_1 \\ \mathbf{f}_2 \\ \mathbf{f}_3 \end{bmatrix} \qquad \mathbf{L} = \begin{bmatrix} \mathbf{I} - ha_{1,1}\mathbf{A} & -ha_{1,2}\mathbf{A} & -ha_{1,3}\mathbf{A} \\ -ha_{2,1}\mathbf{A} & \mathbf{I} - ha_{2,2}\mathbf{A} & -ha_{2,3}\mathbf{A} \\ -ha_{3,1}\mathbf{A} & -ha_{3,2}\mathbf{A} & \mathbf{I} - ha_{3,3}\mathbf{A} \end{bmatrix} \qquad \mathbf{p} = \begin{bmatrix} \mathbf{A}\mathbf{y}_{n-1} + \mathbf{b}x\big(t_{n-1} + c_1 h\big) \\ \mathbf{A}\mathbf{y}_{n-1} + \mathbf{b}x\big(t_{n-1} + c_2 h\big) \\ \mathbf{A}\mathbf{y}_{n-1} + \mathbf{b}x\big(t_{n-1} + c_3 h\big) \end{bmatrix}$$

Rozwiązując układ (10) wyznaczono $\mathbf{f}_1$, $\mathbf{f}_2$ oraz $\mathbf{f}_3$. Następnie podstawiono otrzymane $\mathbf{f}_i$ do (6) w celu wyliczenia $\mathbf{y}_n$ dla $n \geqslant 2$.

Kod zawierający implementację metody 3.:

```
function y = metoda3(A,b,x,h,N,t)

% Współczynniki metody 3
c = [  0,   1/2,    1];
w = [1/6,   2/3,  1/6];
a = [1/6,  -1/6,   0;
     1/6,   1/3,   0;
     1/6,   5/6,   0;];

I = eye(2);
y = zeros(2,N);

L = [ I-h*a(1,1)*A,   -h*a(1,2)*A,   -h*a(1,3)*A;
       -h*a(2,1)*A, I-h*a(2,2)*A,   -h*a(2,3)*A;
       -h*a(3,1)*A,   -h*a(3,2)*A, I-h*a(3,3)*A;];

for n = 2:N
  p = [ A*y(:,n-1) + b*x(t(n-1) + c(1)*h);
        A*y(:,n-1) + b*x(t(n-1) + c(2)*h);
        A*y(:,n-1) + b*x(t(n-1) + c(3)*h);];

  g = L \ p;

  f1 = g(1:2);
  f2 = g(3:4);
  f3 = g(5:6);

  y(:,n) = y(:,n-1) + h*(w(1)*f1 + w(2)*f2 + w(3)*f3);
end % for n

end % function
```

Wykresy $\hat{y}_1(t)$ oraz $\hat{y}_2(t)$ otrzymanych metodą 3. są, wraz z wynikami dokładnymi, przedstawione na rysunku 5.



Rysunek 5: Wykresy $\dot{y}_1(t)$, $\dot{y}_2(t)$ oraz $\hat{y}_1(t)$, $\hat{y}_2(t)$ otrzymanych metodą 3. dla kroku $h = 0.2$

4 Dyskusja Wyników Eksperymentów Numerycznych

W ramach przeprowadzonego rozwiązywania układu równań różniczkowych zwyczajnych (URRZ) (1) uzyskano wartości funkcji $y_1(t)$ oraz $y_2(t)$ dla pięciu różnych podejść: analitycznego, wykorzystującego procedurę `dsolve`, oraz czterech metod numerycznych, w tym procedury `ode45` oraz trzech innych metod dyskretnych określonych jako metoda 1, metoda 2 i metoda 3.

Wyniki uzyskane przy pomocy procedury ode45 wykazały niemal dokładną zgodność z rozwiązaniem analitycznym. Jest to zgodne z teoretycznym założeniem, że ode45, jako metoda adaptacyjna oparta na algorytmie Rungego-Kutty czwartego i piątego rzędu, charakteryzuje się wysoką precyzją w przypadkach, gdzie funkcje są wystarczająco gładkie. Wykresy $\dot{y}_1(t)$ i $\hat{y}_1(t)$ oraz $\dot{y}_2(t)$ i $\hat{y}_2(t)$ nałożone na siebie w tych przypadkach niemalże się pokrywają.

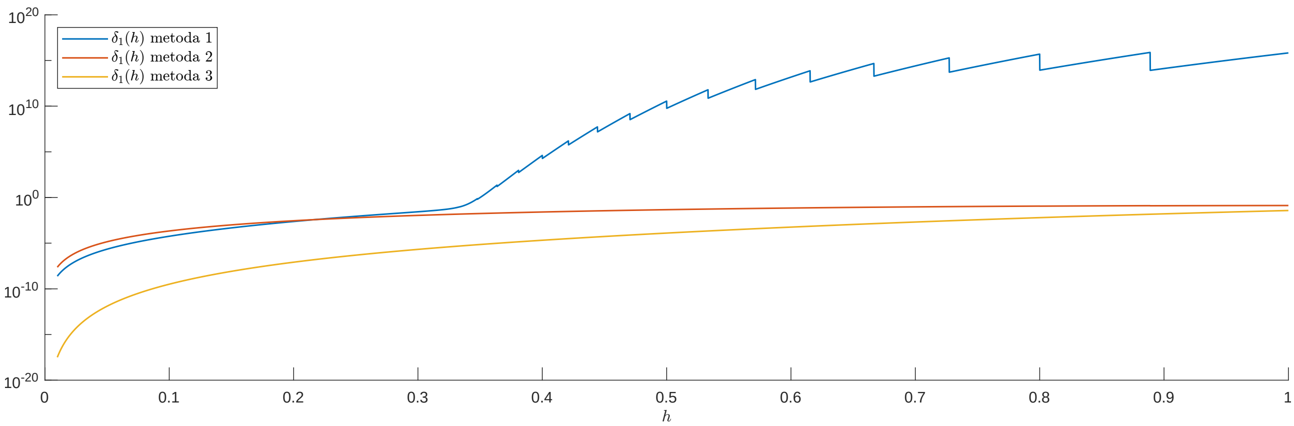
Trzy ostatnie metody okazały się zauważalnie mniej dokładne, z czego metoda 1. jest widocznie mniej dokładna niż metody 2. i 3., a metoda 3. wydaje się być dokładniejsza od metody 2., gdyż na wykresach przedstawionych na rysunku 5. wyliczone punkty pokrywają się z rozwiązaniem dokładnym częściej niż w przypadku metody 2. na rysunku 4.

W celu jednoznacznego ustalenia, która metoda daje najdokładniejsze rozwiązania, zbadano zależność dokładności rozwiązań numerycznych uzyskanych za ich pomocą od długości kroku całkowania h, dla $h \in \left[0.01, 1.00\right]$. Jako kryterium dokładności rozwiązań przyjęto zagregowane błędy względne zdefiniowane następująco:

$$\delta_1(h) = \frac{\sum_{n=1}^{N(h)} \left(\hat{y}_1\left(t_n, h\right) - \dot{y}_1\left(t_n\right)\right)^2}{\sum_{n=1}^{N(h)} \left(\dot{y}_1\left(t_n\right)\right)^2} \quad \text{i} \quad \delta_2(h) = \frac{\sum_{n=1}^{N(h)} \left(\hat{y}_2\left(t_n, h\right) - \dot{y}_2\left(t_n\right)\right)^2}{\sum_{n=1}^{N(h)} \left(\dot{y}_2\left(t_n\right)\right)^2}$$

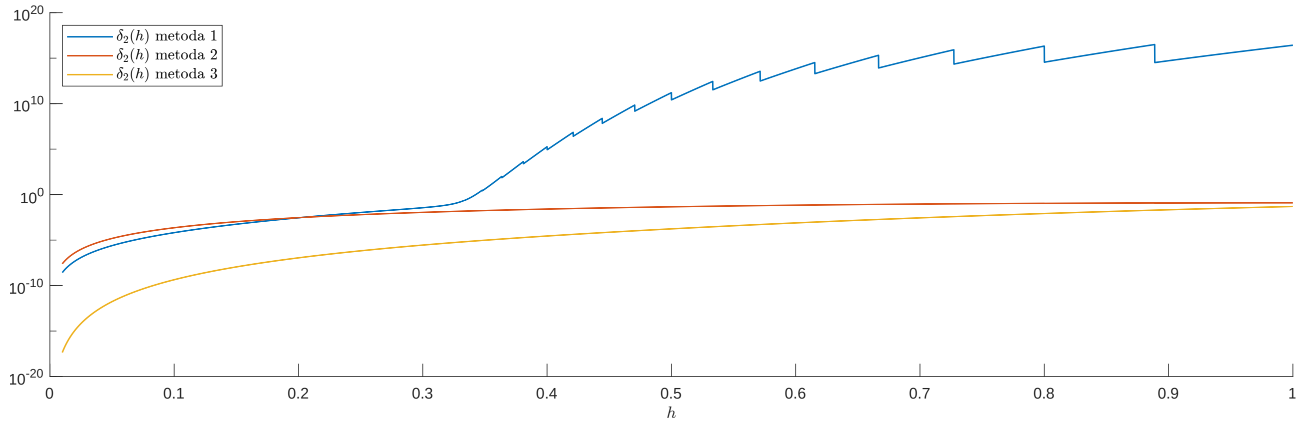
gdzie $\dot{y}_1(t_n)$ i $\dot{y}_2(t_n)$ to wartości funkcji uzyskanych analitycznie za pomocą procedury `dsolve`, a $\hat{y}_1(t_n, h)$ i $\hat{y}_2(t_n, h)$ to ich estymaty uzyskane dla kroku całkowania h numeryczną metodą dyskretną. $N(h)$ oznacza zależną od kroku całkowania liczbę punktów rozwiązania.

Zależność $\delta_1(h)$ od h dla trzech ostatnich metod została przedstawiona na rysunku 6.



Rysunek 6: Wykresy $\delta_1(h)$ dla metod 1.,2.,3.

Zależność $\delta_2(h)$ od h dla trzech ostatnich metod została przedstawiona na rysunku 7.



Rysunek 7: Wykresy $\delta_2(h)$ dla metod 1.,2.,3.

Na podstawie rysunku 6. oraz rysunku 7. zaobserwowano, że metoda 3. jest najdokładniejsza spośród trzech testowanych metod, a metoda 2. jest dokładniejsza od metody 1 dla h większych niż $h \approx 0.2$. Dla metody 1. zaobserwowano zjawisko niestabilności numerycznej. Dla h większych niż $h \approx \frac{1}{3}$ tempo wzrostu $\delta_1(h)$ i $\delta_2(h)$ znacznie rośnie, dokładność metody 1. znacznie się pogarsza.

Bibliografia

1. Dokumentacja MATLAB: `https://www.mathworks.com/help/matlab`.

Listing Programów

plik `Projekt1.m`

```
% Przedział t
tmin = 0;
tmax = 8;

% Współczynniki URRZ
A = [-14/3,-2/3; 2/3,-19/3];
b = [1;1];

% Funkcja x
x = @(t) exp(-t)*sin(t);


%% Zadanie 1

[y1_exact, y2_exact] = solve_using_dsolve(x);
fprintf("y_1(t) = \n");
pretty(y1_exact);
fprintf("y_2(t) = \n");
pretty(y2_exact);
plot_exact(y1_exact,y2_exact,tspan)

%% Zadanie 2

% Wartość kroku
h = 0.2;

% Wyznaczenie N(h) oraz wektora t
N = floor((tmax - tmin)/h) + 1;
t = tmin:h:tmax;

% Procedura ode45
% Funkcja f
f = @(t,y) A*y + b*x(t);
% Przedział czasowy
tspan = [0, 8];
% Warunki początkowe
y0 = [0; 0];
% Wywołanie procedury
[T_ode45,y_ode45] = ode45(f,tspan,y0);

y_ode45 = y_ode45';

% Metoda 1
y_metoda1 = metoda1(A,b,x,h,N,t);

% Metoda 2
y_metoda2 = metoda2(A,b,x,h,N,t);

% Metoda 3
y_metoda3 = metoda3(A,b,x,h,N,t);

plot_results(tspan,y1_exact,y2_exact,...
  T_ode45,y_ode45,t,y_metoda1,y_metoda2,y_metoda3);

```

```
%% Zadanie 3

y1_dot = matlabFunction(y1_exact);
y2_dot = matlabFunction(y2_exact);

n = 1e4;
hmin = 0.01;
hmax = 1.00;
h_range = linspace(hmin,hmax,n);

delta11 = zeros(n,1);
delta12 = zeros(n,1);
delta21 = zeros(n,1);
delta22 = zeros(n,1);
delta31 = zeros(n,1);
delta32 = zeros(n,1);

for i = 1:n
  h = h_range(i);
  N = floor((tmax - tmin)/h) + 1;
  t = tmin:h:tmax;

  y1 = metoda1(A,b,x,h,N,t);
  y2 = metoda2(A,b,x,h,N,t);
  y3 = metoda3(A,b,x,h,N,t);

  [delta11(i), delta12(i)] = calculate_error(y1,y1_dot,y2_dot,t,N);
  [delta21(i), delta22(i)] = calculate_error(y2,y1_dot,y2_dot,t,N);
  [delta31(i), delta32(i)] = calculate_error(y3,y1_dot,y2_dot,t,N);
end % for h

% Wykresy błędów zagregowanych w zależności od h
figure(3);clf;
subplot(2,1,1);
hold on;
yscale('log');
xlabel('$h$','Interpreter','latex');
plot(h_range,delta11,'DisplayName','$\delta_1(h)$ metoda 1','LineWidth',1);
plot(h_range,delta21,'DisplayName','$\delta_1(h)$ metoda 2','LineWidth',1);
plot(h_range,delta31,'DisplayName','$\delta_1(h)$ metoda 3','LineWidth',1);
lgd = legend('show','Interpreter', 'latex', 'Location', 'northwest');
set(lgd, 'FontSize', 10);

subplot(2,1,2);
hold on;
yscale('log');
xlabel('$h$','Interpreter','latex');
plot(h_range,delta12,'DisplayName','$\delta_2(h)$ metoda 1','LineWidth',1);
plot(h_range,delta22,'DisplayName','$\delta_2(h)$ metoda 2','LineWidth',1);
plot(h_range,delta32,'DisplayName','$\delta_2(h)$ metoda 3','LineWidth',1);
lgd = legend('show','Interpreter', 'latex', 'Location', 'northwest');
set(lgd, 'FontSize', 10);
```

plik solve_using_dsolve.m

```
function [y1_exact, y2_exact] = solve_using_dsolve(x)

syms t y1(t) y2(t)

eq1 = diff(y1,t,1) == -14/3*y1(t) - 2/3*y2(t) + x(t);
eq2 = diff(y2,t,1) == 2/3*y1(t) - 19/3*y2(t) + x(t);
eqns = [eq1; eq2];

cond1 = y1(0) == 0;
cond2 = y2(0) == 0;
conds = [cond1; cond2];

sol = dsolve(eqns,conds);
y1_exact = sol.y1;
y2_exact = sol.y2;

end % function
```

plik plot_exact.m

```
function [] = plot_exact(y1_exact,y2_exact,tspan)
y_lim = [-0.004,0.065];
latex_y1_dot = '$\dot{y}_1(t)$';
latex_y2_dot = '$\dot{y}_2(t)$';
figure(1);clf;
hold on;
ylim(y_lim);
xlabel('$t$','Interpreter','latex');
fplot(y1_exact,tspan,'DisplayName',latex_y1_dot,'Color','r');
fplot(y2_exact,tspan,'DisplayName',latex_y2_dot,'Color','b');
lgd = legend('show', 'Interpreter', 'latex');
set(lgd, 'FontSize', 16);
end
```

plik metoda1.m

```
function y = metoda1(A,b,x,h,N,t)

y = zeros(2,N);

for n = 2:N
  y(:,n) = y(:,n-1) + ...
    h*(A*(y(:,n-1) + h/2*(A*y(:,n-1) + b*x(t(n-1)))) + b*x(t(n-1) + h/2));
end % for n

end % function
```

plik `metoda2.m`

```
function y = metoda2(A,b,x,h,N,t)

I = eye(2);
y = zeros(2,N);

% Niejawna metoda Eulera
y(:,2) = (I - h*A) \ (y(:,1) + h*b*x(t(2)));

for n = 3:N
  y(:,n) = (I - h*A) \ ...
      (y(:,n-2) + h*A*y(:,n-2) + h*b*x(t(n)) + h*b*x(t(n-2)));
end % for n

end % function
```

plik `metoda3.m`

```
function y = metoda3(A,b,x,h,N,t)

% Współczynniki metody 3
c = [  0,  1/2,   1];
w = [1/6,  2/3, 1/6];
a = [1/6, -1/6,  0;
     1/6,  1/3,  0;
     1/6,  5/6,  0;];

I = eye(2);
y = zeros(2,N);

L = [ I-h*a(1,1)*A,  -h*a(1,2)*A,  -h*a(1,3)*A;
       -h*a(2,1)*A, I-h*a(2,2)*A,  -h*a(2,3)*A;
       -h*a(3,1)*A,  -h*a(3,2)*A, I-h*a(3,3)*A;];

for n = 2:N
  p = [ A*y(:,n-1) + b*x(t(n-1) + c(1)*h);
        A*y(:,n-1) + b*x(t(n-1) + c(2)*h);
        A*y(:,n-1) + b*x(t(n-1) + c(3)*h);];

  g = L \ p;

  f1 = g(1:2);
  f2 = g(3:4);
  f3 = g(5:6);

  y(:,n) = y(:,n-1) + h*(w(1)*f1 + w(2)*f2 + w(3)*f3);
end % for n

end % function
```

plik `plot_results.m`

```
function [] = plot_results(tspan,y1_exact,y2_exact,...
  T_ode45,y_ode45,t,y_m1,y_m2,y_m3)

y1_ode45 = y_ode45(1,:);
y2_ode45 = y_ode45(2,:);
y1_m1 = y_m1(1,:);
y2_m1 = y_m1(2,:);
y1_m2 = y_m2(1,:);
y2_m2 = y_m2(2,:);
y1_m3 = y_m3(1,:);
y2_m3 = y_m3(2,:);

y_lim = [-0.004,0.065];
colors{1} = [255 0 0];
colors{2} = [0 0 255];
colors{3} = [128 0 255];
colors{4} = [0 196 0];
for i=1:4
  colors{i} = colors{i} / 255;
end
latex_y1_dot = '$\dot{y}_1(t)$';
latex_y2_dot = '$\dot{y}_2(t)$';
latex_y1_hat = '$\hat{y}_1(t)$';
latex_y2_hat = '$\hat{y}_2(t)$';

figure(2); clf;
subplot(2,2,1);
hold on;
ylim(y_lim);
xlabel('$t$','Interpreter','latex');
title("ode45");
fplot(y1_exact,tspan,...
  'DisplayName',latex_y1_dot,'LineStyle','-','Color',colors{1});
fplot(y2_exact,tspan,...
  'DisplayName',latex_y2_dot,'LineStyle','-','Color',colors{2});
plot(T_ode45,y1_ode45,...
  'DisplayName',latex_y1_hat,'LineStyle','-','Color',colors{3});
plot(T_ode45,y2_ode45,...
  'DisplayName',latex_y2_hat,'LineStyle','-','Color',colors{4});
lgd = legend('show', 'Interpreter', 'latex');
set(lgd, 'FontSize', 16);

subplot(2,2,2);
hold on;
ylim(y_lim);
xlabel('$t$','Interpreter','latex');
title("metoda 1.");
fplot(y1_exact,tspan,...
'DisplayName',latex_y1_dot,'LineStyle','-','Color',colors{1});
fplot(y2_exact,tspan,...
'DisplayName',latex_y2_dot,'LineStyle','-','Color',colors{2});
plot(t,y1_m1,...
'DisplayName',latex_y1_hat,'LineStyle','-','Color',colors{3});
plot(t,y2_m1,...
'DisplayName',latex_y2_hat,'LineStyle','-','Color',colors{4});
lgd = legend('show', 'Interpreter', 'latex');
```

```
set(lgd, 'FontSize', 16);

subplot(2,2,3);
hold on;
ylim(y_lim);
xlabel('$t$','Interpreter','latex');
title("metoda 2.");
fplot(y1_exact,tspan,...
'DisplayName',latex_y1_dot,'LineStyle','-','Color',colors{1});
fplot(y2_exact,tspan,...
'DisplayName',latex_y2_dot,'LineStyle','-','Color',colors{2});
plot(t,y1_m2,...
'DisplayName',latex_y1_hat,'LineStyle','-','Color',colors{3});
plot(t,y2_m2,...
'DisplayName',latex_y2_hat,'LineStyle','-','Color',colors{4});
lgd = legend('show', 'Interpreter', 'latex');
set(lgd, 'FontSize', 16);

subplot(2,2,4);
hold on;
ylim(y_lim);
xlabel('$t$','Interpreter','latex');
title("metoda 3.");
fplot(y1_exact,tspan,...
'DisplayName',latex_y1_dot,'LineStyle','-','Color',colors{1});
fplot(y2_exact,tspan,...
'DisplayName',latex_y2_dot,'LineStyle','-','Color',colors{2});
plot(t,y1_m3,...
'DisplayName',latex_y1_hat','LineStyle','-','Color',colors{3});
plot(t,y2_m3,...
'DisplayName',latex_y2_hat,'LineStyle','-','Color',colors{4});
lgd = legend('show', 'Interpreter', 'latex');
set(lgd, 'FontSize', 16);
```

plik calculate_error.m

```
function [delta1, delta2] = calculate_error(y_hat,y1_dot,y2_dot,t,N)
  y1_hat = y_hat(1,:);
  y2_hat = y_hat(2,:);

  top_sum1 = 0;
  top_sum2 = 0;
  bottom_sum1 = 0;
  bottom_sum2 = 0;

  for n = 1:N
    top_sum1 = top_sum1 + (y1_hat(n) - y1_dot(t(n)))^2;
    bottom_sum1 = bottom_sum1 + (y1_dot(t(n)))^2;
    top_sum2 = top_sum2 + (y2_hat(n) - y2_dot(t(n)))^2;
    bottom_sum2 = bottom_sum2 + (y2_dot(t(n)))^2;
  end % for n
  delta1 = top_sum1/bottom_sum1;
  delta2 = top_sum2/bottom_sum2;

end % function
```