Rozwiązywanie układów równań różniczkowych zwyczajnych

Modelowanie matematyczne

Andrzej Kupiec

Listopad 2023

dr. hab. inż. Kajetana Snopek dr inż. Paweł Mazurek

Wydział Matematyki i Nauk Informacyjnych
Politechnika Warszawska

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

Spis treści

1. Lista symboli użytych w sprawozdaniu

URRZ - układ równań różniczkowych zwyczajnych:

$$\frac{dy_1(t)}{dt} = -7y_1(t) - 2y_2(t) + x(t)$$
$$\frac{dy_2(t)}{dt} = 2y_1(t) - 2y_2(t) + x(t)$$

dla $t \in [0, 8]$, gdzie $x(t) = \exp(-t) \cdot \sin(t)$

$y_1(t)$, $y_2(t)$ - dokładne rozwiązania URRZ
$y_{i,3}(t_n)$ - rozwiązanie URRZ metodą ode45 $y_{i,4}(t_n)$ - rozwiązanie URRZ metodą $y_n = y_{n-1} + hf(t_{n-1} + \frac{h}{2}, y_{n-1} + \frac{h}{2}f(t_{n-1}, y_{n-1})$
$y_{i,5}(t_n)$ - rozwiązanie URRZ metodą $(y_n = y_{n-1} + \frac{h}{12}(5f(t_n, y_n) + 8f(t_{n-1}, y_{n-1}) - f(t_{n-2}, y_{n-2}))$
$y_{i,6}(t_n)$ - rozwiązanie URRZ metodą $y_n = y_{n-1} + h\sum_{k=1}^{3} w_k f_k$,
gdzie $f_k = f(t_{n-1} + c_k h, y_{n-1} + h\sum_{k=1}^{3} a_{k,k} f_k)$
$i = 1, 2$ jest indeksem opisującym rozwiązanie $y_i(t)$, np. $y_{1,5}$ to rozwiązanie $y_1(t)$ metodą $y_5(t)$
$a_{k,k}, w_k, c_k$ - wartości z tabeli Butchera, opisane dokładniej w dalszej części sprawozdania
A - macierz reprezentująca współczynniki przy niewiadomych $y_1(t)$ oraz $y_2(t)$
b - wektor reprezentujący współczynniki przy $x(t)$
$y(t) = \begin{bmatrix} y_1(t) & y_2(t) \end{bmatrix}^T$ - wektor funkcji będących rozwiązaniem URRZ
$f(t_n, y_n)$ - funkcja określona przez URRZ i dana wzorem $\left.\frac{dy(t)}{dt}\right|_{t=t_n} = f(t_n)$, ponadto $f(t, y) = \mathbf{A}y(t) + bx(t)$
$\delta_1(h)$ - zagregowany błąd względny dla rozwiązania $y_1(t)$, opisany dokładniej w dalszej części sprawozdania
$\delta_2(h)$ - analogiczny błąd dla rozwiązania $y_2(t)$
I - macierz jednostkowa 2×2
t_0, T - czas początkowy i końcowy, równe kolejno 0s i 8s
$h = \frac{T - t_0}{N}$
C - stała liczba rzeczywista

2. Wstęp

Projekt poświęcony był porównaniem różnych metod numerycznych rozwiązywania układu równań różniczkowych zwyczajnych. Na początku warto uściślić kilka niezbędnych pojęć. Rzędem metody nazywa się potęga ostatniego wyrazu przy rozkładzie w szereg Taylora, któremu odpowiada rozwiązanie uzyskane przy jej pomocy. Bardziej formalnie definiuje się, że metoda ma rząd równy r wtedy i tylko wtedy, gdy

$$e(h) = Ch^{r+1} + o(h^{r+1}),\ C \neq 0,\ \lim_{h to 0} \frac{o(h^{r+1})}{h^{r+1}} = 0 \tag{2.1}$$

Najważniejszą informacją jest to, że im większy rząd, tym metoda dokładniejsza. Metody rozwiązywania URRZ dzielimy, ze względu na sposób postępowania numerycznego, na jedno- i wielokrokowe. Schematem jednokrokowym nazywamy dla stałej różnicy $h = \frac{T - t_0}{N}$ równanie różnicowe:

$$y_{n+1} = y_n + f(h, t_n, x_n, x_{n+1})\ dla\ n = 0, 1, ..., N \tag{2.2}$$

Dodatkowo, jeśli f nie zależy od y_{n+1} to schemat nazywamy otwartym, a w przeciwnym przypadku - zamkniętym. Schematem wielokrokowym nazywamy, przy tak samo zdefiniowanym h, równanie:

$$\sum_{j=0}^{k} a_k x_{n+j}^h = h \sum_{j=0}^{k} b_j f_{j+n}^h \; n \geqslant 0, \; a_k \neq 0 \; if_j^h = f(t_j, x_j^h) \; dla \; t_j = t_0 + jh \tag{2.3}$$

Innymi słowy, metody jednokrokowe mogą być zależne wyłącznie od poprzedniego przybliżenia, czasu, kroku, a także obecnie obliczanego przybliżenia. Schematy wielokrokowe nie mają takich ograniczeń, mogą być zależne od większej liczby przybliżeń.

2.1. Omówienie algorytmów z treści zadania

Pierwsza metoda rozwiązania opierała się na wykorzystaniu MATLAB Symbolic Toolbox, pakietu umożliwiającego rozwiązywanie problemów matematycznych w sposób zbliżony do analitycznego, a nie numerycznego. Poprzez użycie procedury dsolve z tego pakietu możliwe jest uzyskanie dokładnego wyniku układu równań w postaci wzorru symbolicznego. W zadaniu wykorzystana została ponadto inna funkcja wbudowana programu MATLAB, *ode45*. Zgodnie z dokumentacją, implementuje ona jednokrokową metodę Runge-Kutty czwartego rzędu. Nadaje się ona do rozwiązywania równań różniczkowych zwyczajnych, czyli takich, gdzie występują pochodne po jednej zmiennej. W kolejnym punkcie dana została do implementacji, metoda zmodyfikowana metoda Eulera, natomiast w literaturze angielskiej zwana zazwyczaj midpoint method - metoda kroku środkowego. W dalszej części sprawozdania, ze względu na wygodę, nazywana będzie zmodyfikowaną metodą Eulera. Następnie dana do zaimplementowania była metoda Adamsa-Moultona rzędu trzeciego, która jest wielokrokowa. Warunki początkowe są niewystarczające do jej wyznaczenia, zatem trzeba skorzystać z innej metody do wyznaczenia drugiego przybliżenia, co dokładniej opisane zostało w odpowiednim podrozdziale raportu. Ostatnia funkcja do napisania także należy do jednokrokowych metod Runge-Kutty, której współczynniki przedstawiono w tablicy Butchera. Oczekiwane rozwiązanie URRZ dla trzech ostatnich metod powinno być dokładniejsze (poza funkcjami wbudowanymi) dla metody Adamsa-Moultona i Runge-Kutty wyrażoną tabelą Butchera.

3. Metodyka i wyniki doświadczeń

Przed rozpoczęciem implementacji części metod niezbędne było przeprowadzenie przekształceń. Dane były zerowe warunki początkowe, czyli, mówiąc ściślej $y_1(0) = 0, \; y_2(0) = 0$

3.1. Zmodyfikowana metoda Eulera

To równanie nie wymagało większej liczby przekształceń. Podstawiając $f(t, y) \; = \mathbf{A}y(t) + bx(t)$ do:

$$y_n = y_{n-1} + hf(t_{n-1} + \frac{h}{2}, y_{n-1} + \frac{h}{2} f(t_{n-1}, y_{n-1}))$$

dochodzimy do ostatecznej postaci równania do implementacji:

$$y_n = y_{n-1} + h\mathbf{A}(y_{n-1} + \frac{h}{2}\mathbf{A}y_{n-1} + bx(t_{n-1})) + hbx(t_{n-1} + \frac{h}{2}) \tag{3.1}$$

3.2. Metoda Adamsa-Moultona rzędu trzeciego

Pierwszą trudnością napotkaną przy tej metodzie jest jej wielokrokowość. Z tego powodu, do obliczeń niezbędne jest wyznaczenie pierwszego przybliżenia za pomocą innego algorytmu. Ze względu na większą stabilność nume-

ryczną, wykorzystuje się do tego zamkniętą (zwaną także niejawną) metodę Eulera, która dana jest równaniem:

$$y_n = y_{n-1} + hf(t_n, y_n)$$

które przekształcając do implementacji przybiera postać:

$$y_n = y_{n-1} + \frac{h}{12}[5f(t_n, y_n) + 8f(t_{n-1}, y_{n-1})] \tag{3.2}$$

Podobnie postępując z równaniem:

$$y_n = (\mathbf{I} - h\mathbf{A}^{-1}) \cdot (y_{n-1} + hbx(t_n))$$

które, po podstawieniu $f(t_n, y_n) = \mathbf{A}y_n + bx(t_n)$ i dokonaniu kolejnych przekształceń, wygląda w następujący sposób:

$$y_n = (1 - \frac{5}{12}h\mathbf{A})^{-1} \cdot (y_{n-1} + \frac{h}{12}(5bx(t_n) + 8\mathbf{A}y_{n-1} + 8bx(t_{n-1}) - \mathbf{A}y_{n-2} + bx(t_{n-2}))) \tag{3.3}$$

3.3. Metoda Runge-Kutty z tabelą Butchera

Do rozwiązania tego podpunktu potrzebne jest wyznaczenie f_1, f_2 i f_3. Tworzymy zatem układ równań:

$$\begin{aligned} f_1 &= \mathbf{A}((y_{n-1} + ha_{1,1}f_1 + ha_{1,2}f_2 + ha_{1,3}f_3) + bx(t_{n-1} + c_1h) \\ f_2 &= \mathbf{A}((y_{n-1} + ha_{2,1}f_1 + ha_{2,2}f_2 + ha_{2,3}f_3) + bx(t_{n-1} + c_2h) \\ f_3 &= \mathbf{A}((y_{n-1} + ha_{3,1}f_1 + ha_{3,2}f_2 + ha_{3,3}f_3) + bx(t_{n-1} + c_3h) \end{aligned} \tag{3.4}$$

Dokonując odpowiednich przekształceń dochodzimy do: $\mathbf{L}g = p$, gdzie macierz lewej strony L jest równa:
$\begin{bmatrix} \mathbf{I} - ha_{1,1}\mathbf{A} & -ha_{1,2}\mathbf{A} & -ha_{1,3}\mathbf{A} \\ -ha_{2,1}\mathbf{A} & \mathbf{I} - ha_{2,2}\mathbf{A} & -ha_{2,3}\mathbf{A} \\ -ha_{3,1}\mathbf{A} & -ha_{3,2}\mathbf{A} & \mathbf{I} - h_{3,3}\mathbf{A} \end{bmatrix}$,wektor g: $\begin{bmatrix} f_1 \\ f_2 \\ f_3 \end{bmatrix}$ a macierz prawej strony p ma postać: $\begin{bmatrix} f(t_{n-1} + c_1h, y_{n-1}) \\ f(t_{n-1} + c_2h, y_{n-1}) \\ f(t_{n-1} + c_3h, y_{n-1}) \end{bmatrix}$

Wartości współczynników odczytujemy z tabeli:

c_1	$a_{1,1}$	$a_{1,2}$	$a_{1,3}$	0	$\frac{1}{30}$	$-\frac{1}{15}$	$\frac{1}{30}$
c_2	$a_{2,1}$	$a_{2,2}$	$a_{2,3}$	$\frac{1}{2}$	$\frac{5}{24}$	$\frac{1}{3}$	$-\frac{1}{24}$
c_3	$a_{3,1}$	$a_{3,2}$	$a_{3,3}$	1	$\frac{2}{15}$	$\frac{11}{15}$	$\frac{2}{15}$
	w_1	w_2	w_3		$\frac{1}{6}$	$\frac{2}{3}$	$\frac{1}{6}$

Po wyznaczeniu g z równania:

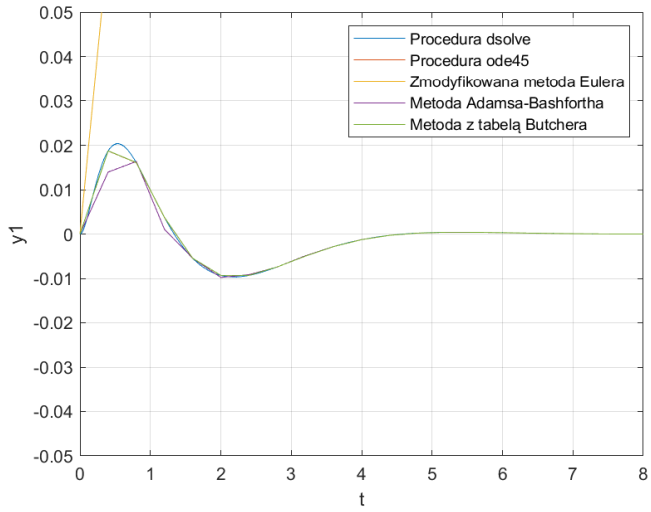
$$g = \mathbf{L}^{-1} \cdot p \tag{3.5}$$

wystarczy zaimplementować:

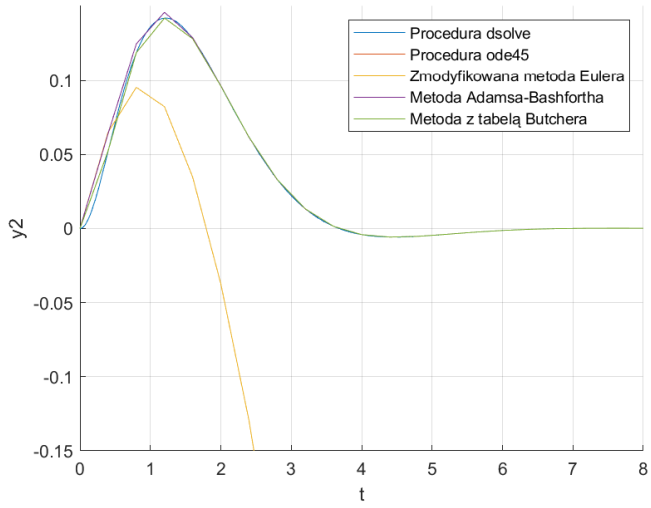
$$y_n = y_{n-1} + h\sum_{k=1}^{3} w_k f_k \tag{3.6}$$

4. Dyskusja wyników eksperymentów numerycznych

Porównanie najlepiej zacząć od kilku wykresów. Jak widać zarówno na Rysunku 1., jak i 2., już dla względnie

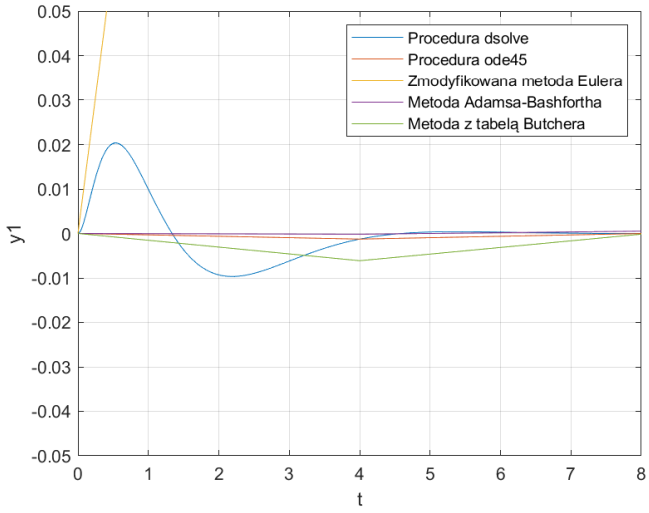


Rysunek 1. Porównanie metod dla y1 i kroku h = 0.4

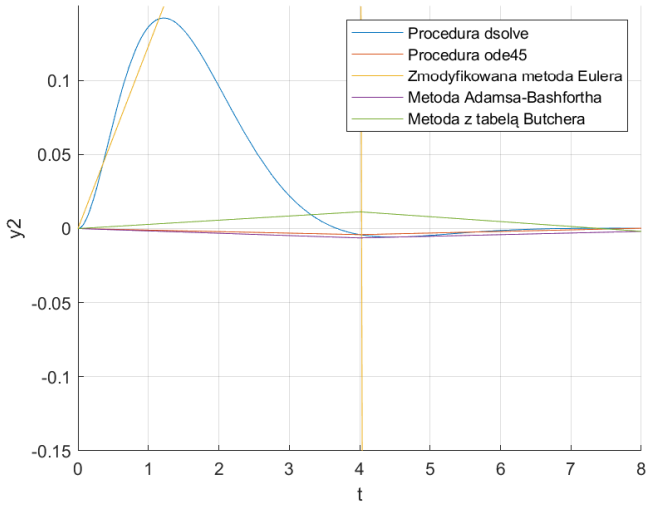


Rysunek 2. Porównanie metod dla y2 i kroku h = 0.4

małego kroku metody mają bardzo różną stabilność numeryczną. Najmniejszą dokładność możemy zaobserwować dla zmodyfikowanej procedury Eulera, której wynik znacząco odbiega od pozostałych. Najbardziej zbliżoną z samodzielnie zaimplementowanych metod do dokładnego wyniku okazuje się metoda wykorzystująca tabelę Butchera. Porównując jednak czasy wykonywania programów dla kroku $h = \begin{bmatrix} 0.01 & 0.1 & 0.2 & 0.4 & 0.8 & 2.0 & 4.0 & 8.0 \end{bmatrix}$ wynika, że z dokładnością zwiększa się także czas wykonywania programu. O ile w tym przypadku nie ma to aż tak dużego znaczenia, jednak wybierając odpowiednią metodę do rozwiązywania układów równań różniczkowych, należy wziąć ten aspekt pod uwagę. Niewykluczone, że w bardziej skomplikowanych przypadkach program będzie całkowicie nieefektywny. Ponadto w metodzie z tabelą Butchera tworzona jest macierz L o wymiarach 6 ×6, w większych układach równań może to wiązać się z większą złożonością pamieciową, a co za tym idzie, potencjalną nieefektywnością programu. Kolejnym wnioskiem, być może oczywistym, jest to, że tak kluczowym dla dokładnego rozwiązania układu równań jest także odpowiedni wybór długości kroku. Z Rysunków 3. i 4.

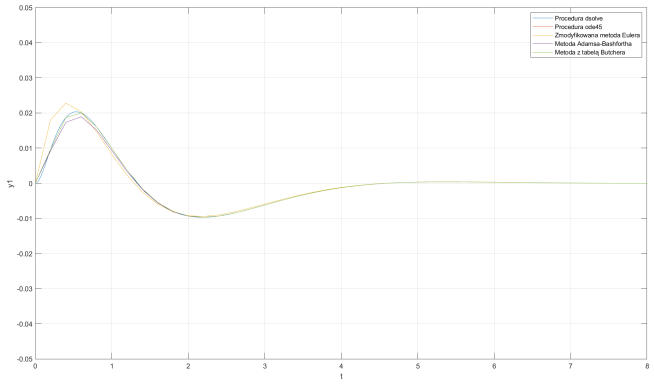


Rysunek 3. Porównanie metod dla y1 i h = 4

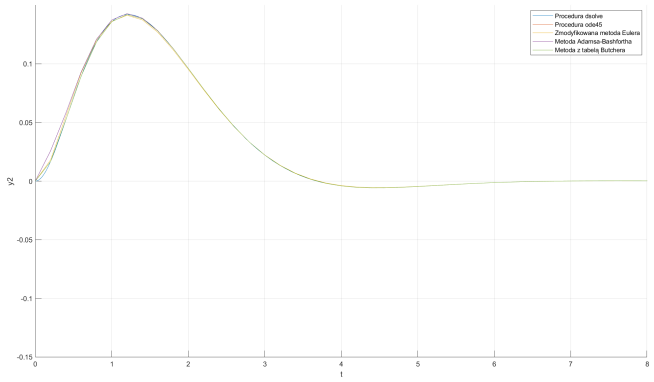


Rysunek 4. Porównanie metod dla y2 i h = 4

widać, że dla odpowiednio dużej wartości h żadna z metod nie pozostaje zbytnio stabilna. Jeżeli zatem dla zadanego problemu nieznana byłaby wartość optymalnego kroku, należy rozważyć rozwiązanie symboliczne, które jednak prawdopodobnie może być czasochłonne. Należy się także liczyć z tym, że rozwiązanie analityczne może w ogóle nie istnieć. O ile duży błąd na Rysunkach 3. i 4. nie powinien dziwić, to jednak wynik na 5. i 6. może budzić pewne zaskoczenie. Dobierając wciąż względnie mały krok 0.2 zmodyfikowana metoda Eulera już zaczyna odbiegać od dokładnego wyniku, podczas gdy wszystkie pozostałe metody wciąż są poprawne. Warto także wspomnieć, że dzieje się tak przede wszystkim dla y_1, co dodatkowo sugeruje, że dobór odpowiedniej metody w dużej mierze zależy od warunków początkowych i oczekiwań użytkownika. Może zdarzyć się sytuacja, w której jedno z rozwiązań powinno być dokładniejsze, a wyjątkowo ważny jest czas jak i złożoność pamięciowa. Jednak w klasycznej sytuacji, kiedy przywiązywana jest duża waga do dokładności, najlepiej nie rozważać zmodyfikowanej metody Eulera.



Rysunek 5. Porównanie metod dla y1 i kroku h=0.2

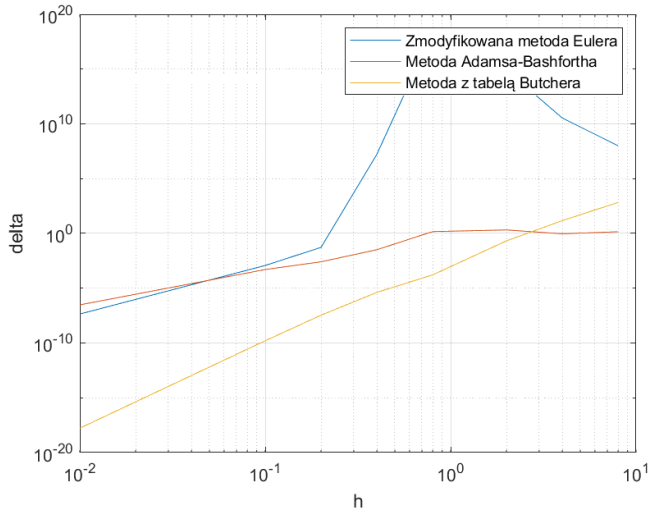


Rysunek 6. Porównanie metod dla y2 i kroku h=0.2

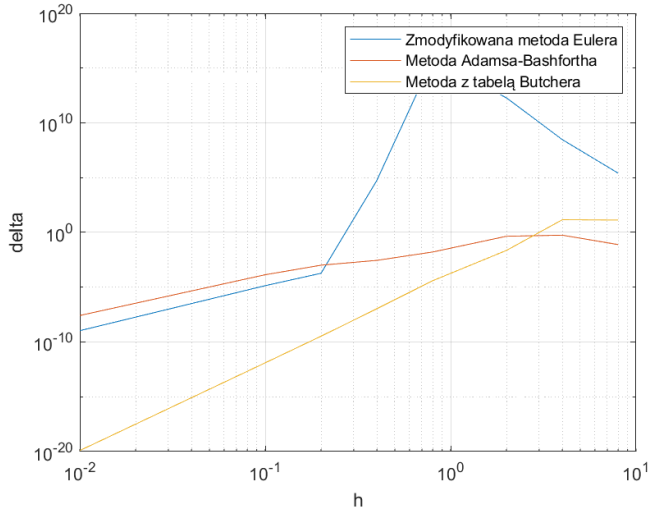
Do dokładniejszej analizy błędu metod potrzeba kolejnej definicji. Niech zagregowany błąd względny użyty w dalszych rozważaniach, będzie wyrażony następującym wzorem:

$$\delta_1(h) = \frac{\sum_{n=1}^{N(h)} (y_1(t_n, h) - y_{i,1}(t_n, h))^2}{\sum_{n=1}^{N(h)} (y_1(t_n, h))^2}, \ \delta_2(h) = \frac{\sum_{n=1}^{N(h)} (y_2(t_n, h) - y_{i,2}(t_n, h))^2}{\sum_{n=1}^{N(h)} (y_2(t_n, h))^2} \tag{4.1}$$

gdzie $i = 3, 4, 5, 6$, jak na początku sprawozdania Na podstawie Rysunku 7. i 8. dojść do kilku konkluzji. Widać, że zagregowany błąd dla metody wykorzystującej tabelę Butchera, początkowo najdokładniejszej, od pewnego h staje się większy od błędu metody Adamsa-Bashfortha. Zatem algorytm teoretycznie najlepszy, z czasem może przestać być aż tak stabilny numerycznie. Zaobserwować też można znaczący wzrost niedokładności metody Eulera dla stosunkowo małej wartości kroku. Potwierdza to dokładnie oczekiwania, metoda ta ma najmniejszy rząd. Warto też wspomnieć, że metody Adamsa-Bashfortha i wykorzystujące tabelę Butchera dla wszystkich wartości zachowują się względnie stabilnie, błąd osiągnął maksimum ≈ 10.



Rysunek 7. Zależność błędu wyznaczenia wartości y1 od kroku h



Rysunek 8. Zależność błędu wyznaczenia wartości y2 od kroku h

5. Wnioski

Na podstawie przeprowadzonych doświadczeń możemy wysunąć kilka wniosków. Wystarczy pobieżnie spojrzeć na wykres błędu, by zobaczyć, że metody Adamsa-Moultona oraz Runge-Kutty są znacznie bardziej stabilne numeryczne i lepsze do wyznaczania rozwiązań zbliżonych do rzeczywistości od zmodyfikowanej metody Eulera. Wiążą się one z ewentualnie trochę wydłużonym działaniem programu, dodatkowym ograniczeniem tutaj może być ewentualna trudność w implementacji. Oczywiście wszystkie parametry mierzone były wyłącznie dla podanego układu równań, trudno przewidzieć, jak zachowałyby się dla innych, być może bardziej obciążających możliwości obliczeniowe komputera. W przypadku, gdy istnieje potrzeba szybkich obliczeń, pakiet symboliczny, pomimo wielu swoich zalet, może okazać się niewystarczający. Funkcja wbudowana *ode45*, oraz wspomniane metody Adamsa i z tabelą Butchera radzą sobie wystarczająco dobrze. Natomiast zmodyfikowana metoda Eulera sprawdzi się najlepiej, gdy przedział będzie dość mały, przez co wybór małego kroku nie obciąży możliwości pamięciowych komputera. Wtedy rozwiązanie powinno być wystarczająco dokładne oraz przypuszczalnie także szybsze i prostsze do implementacji od pozostałych metod.

6. Programy

Poniżej znajduje się lista wszystkich programów użytych do wyznaczenia rozwiązania

```
function dydt = myOde(t,y)
% MYODE funkcja reprezentujaca uklad rownan rozniczkowych zwyczajnych
% z z zadania

    x = sin(t) * exp(-t);
    dydt = zeros(2,1);
    dydt(1) = -7*y(1)-2*y(2)+x;
    dydt(2) = 2*y(1)-2*y(2)+x;
end
```

```
function [y1, y2] = zad1()
% ZAD1 funkcja rozwiazujaca uklad rownan rozniczkowych zwyczajnych:
%
%   [ dy1(t)/dt = -7*y1(t)-2y2(t)+x(t);
%     dy2(t)/dt = 2y1(t)-2y2(t)+x(t); ],
% dla 0 <= t <= 8, x(t) = exp(-t)+sin(T)
% z wykorzystaniem pakietu symbolicznego. Nie przyjmuje ona zadnych
% argument w wejsciowych.
%
% Wyjscie:
% y1, y2 - zmienne symboliczne b d ce rozwi zaniem ukladu r wnan


syms y1(t) y2(t) t
tspan = [0 8];
dy1 = diff(y1,t);
dy2 = diff(y2,t);
eqn1 = dy1 == -7*y1(t) - 2*y2(t) + exp(-t) .* sin(t);
eqn2 = dy2 == 2*y1(t) - 2*y2(t) + exp(-t) .* sin(t);
cond = [y1(0) == 0 y2(0) == 0];
[y1(t), y2(t)] = dsolve([eqn1 eqn2], cond);

end % function
```

```
function [y] = zad2_1(h)
% ZAD2_1 funkcja rozwiazujaca uklad rownan rozniczkowych zwyczajnych za
% pomoca metody danej wzorem:
% y(n) = y(n-1)+h*f(t(n-1)+h/2, y(n-1)+h/2*f(t(n-1), y(n-1))
% Nie przyjmuje zadnych argumentww wejsciowych.
%
% Wyjscie:
%  y - macierz, w ktorej pierwsza kolumna odpowiada wartosciom y1(t).
%      a drugi y2(t)

y0 = [0; 0];
tspan = 0:h:8;
[t, y] = ode45(@myOde, tspan, y0);

end % function
```

```
function [y] = zad2_2(h, y)
% ZAD2_2 funkcja rozwi zuj ca uklad rownan rozniczkowych zwyczajnych za
% pomoca metody danej wzorem:
% y(n) = y(n-1)+h*f(t(n-1)+h/2, y(n-1)+h/2*f(t(n-1), y(n-1))
```

```
%
% Wejscie:
%  h - krok
%  y - warto    y(1)
% Wyj cie:
%  y - macierz, w ktorej pierwsza kolumna odpowiada wartosciom y1(t).
%      a druga y2(t)

tspan = [0:h:8];
n = length(tspan);
x = @(t) exp(-t) .* sin(t);
A=[-7 -2; 2 -2];

for i = 2:n
  y_mid = y(i - 1, :)' + h / 2 * (A * y(i - 1, :)' + x(tspan(i - 1)));
  y(i, :) = y(i - 1, :) + h * (A * y_mid + x(tspan(i - 1) + h / 2))';
end

end % function
```

```
function [y] = zad2_3(h, y)
% ZAD2_3 funkcja rozwiazujaca uklad rownan rozniczkowych zwyczajnych za
% pomoca metody danej wzorem:
% y(n) = y(n-1)+h/12*(5*f(t(n), y(n))+8f(t(n-1), y(n-1))-5*f(t(n-2),
% y(n-2)))
%
% Wejscie:
%  h - krok
%  y - wektor wartosci [y(1) y(2)]
% Wyjscie:
%  y - macierz, w ktorej pierwsza kolumna odpowiada wartosciom y1(t).
%      a druga y2(t)


tspan = [0:h:8];
n=length(tspan);
y = zeros(n,2);
x = @(t) sin(t) * exp(-t);
A = [-7 -2; 2 -2];
y(2,:) = inv(eye(2,2) - h*A) * ((y(1,:) + h*x(tspan(2))))';

for i=3:n
    y(i,:) = inv(eye(2,2)-5/12*h*A) * (y(i-1,:)' + h/12*(5*x(tspan(i))+8*A*y(i-1,:)'+8*x(tspan(i-1))-...
        A*y(i-2,:)'-x(tspan(i-2))));
end

end % function
```

```
function [err] = err(y1, y2, y_approx, h)
% ERR funkcja liczaca zagregowany blad wzgledny
%
% Wejscie:
% y1, y2 - dokladne rozwiazania URRZ
% y_approx - macierz z przyblizonymi rozwiazaniami
% h - krok

tspan = [0:h:8];
err(1) = double(sum((y_approx(:,1)' - y1(tspan)).^2) / (sum(y1(tspan) .^2)));
err(2) = double(sum((y_approx(:,2)' - y2(tspan)) .^2) / (sum(y2(tspan) .^2)));
```

```

end
```

Główny skrypt obsługujący funkcje:

```
clear
[y1, y2] = zad1();
% n = flip([1 2 4 10 20 40 80 800]); % Odkomentowac do wykresu err(h)
% h = 8 ./ n;                        % jw.
% h = 0.2;   % Odkomentowac do wykresu porownuj cego rozwi zania,
             % w zaleznosci od h otrzymana zostanie otrzymana inna dokladnosc


close all
figure
j = 0;

for i=1:length(h)
    y3 = zad2_1(h(i));
    y = zeros(8/h(i)+1,2);
    % pkt.2
    y4 = zad2_2(h(i),y);
    % pkt.3
    y5 = zad2_3(h(i),y);
    % pkt.4
    y6 = zad2_4(h(i),y);
    er(1,i+j:i+j+1) = err(y1,y2,y4,h(i));
    er(2,i+j:i+j+1) = err(y1,y2,y5,h(i));
    er(3,i+j:i+j+1) = err(y1,y2,y6,h(i));
    j = j + 1;
end

loglog(h, er(:,1:2:end));
grid on
legend("Zmodyfikowana metoda Eulera", "Metoda Adamsa-Bashfortha", ...
    "Metoda z tabela Butchera");
xlabel("h");
ylabel("delta");
figure
loglog(h, er(:,2:2:end));
grid on
legend("Zmodyfikowana metoda Eulera", "Metoda Adamsa-Bashfortha", ...
    "Metoda z tabela Butchera");
xlabel("h");
ylabel("delta");
pause
close all
figure
t = [0:h:8];
fplot(y1);
hold on
plot(t, y3(:,1), t, y4(:,1), t, y5(:,1), t, y6(:,1));
grid on
axis([0 8 -0.05 0.05]);
legend("Procedura dsolve", "Procedura ode45", ...
    "Zmodyfikowana metoda Eulera", "Metoda Adamsa-Bashfortha", ...
    "Metoda z tabela Butchera");
xlabel("t");
ylabel("y1");
figure
```

```
hold on
grid on
fplot(y2);
plot(t, y3(:,2),t, y4(:,2), t, y5(:,2), t, y6(:,2));
axis([0 8 -0.15 0.15]);
legend("Procedura dsolve", "Procedura ode45", ...
    "Zmodyfikowana metoda Eulera", "Metoda Adamsa-Bashfortha", ...
    "Metoda z tabela Butchera");
xlabel("t");
ylabel("y2");
```

Bibliografia

[1] Marcinkowski, L. (n.d.). Numeryczne rozwiązywanie równań różniczkowych. Matematyka stosowana. Retrieved from https://mst.mimuw.edu.pl/wyklady/nrr/wyklad.pdf

[2] University of Manitoba. (2016, March 21). Multistep methods - University of Manitoba. Retrieved from https://home.cc.umanitoba.ca/ farhadi/Math2120/Multistep%20Methods.pdf

[3] Karcz-Dulęba, I. (n.d.). Metody numeryczne rozwiązywania równań różniczkowych zwyczajnych. Lecture, Wrocław. Retrieved December 1, 2023, from http://anna.czemplik.staff.iiar.pwr.wroc.pl/images/Danaliza/wykla_met_num.pdf.