Zespół	Radosław Smoter Arkadiusz Halat
Numer grupy	LK3
Nazwa ćwiczenia	Rozwiązywanie równań różniczkowych
Numer ćwiczenia	1
Data oddania	11.04.2022
Prowadzący przedmiot	Mgr inż. Denys Gutenko
Ocena	

Modelowanie Układów Dynamicznych

Kraków 2022

Spis treści

Cel ćwiczenia

Rozwiązywanie układów równań różniczkowych na potrzeby modelowania układów dynamicznych, w sytuacjach, gdy „prawdziwe” problemy nie posiadają rozwiązań możliwych do osiągnięcia drogą analityczną.

Kody źródłowe

Zadanie 1.

Funkcja wspólna dla wszystkich przykładów.

```
function [dy] = funkcja(t,y)
  dy = [y(2); 4*sin(t) + 5*cos(2*t) + y(1)];
end
```

Przykład 1.

```
clear; clc; close all;

h = 0.001;          % Step.
time = 10;          % Time.
y = [-1; -2];       % Starting conditions.
t = 0:h:time;       % Full timestamp vector.

% Predeclare yres.
yres = zeros(2, length(t));

for i = 1:length(t)
  yres(:, i) = y + h.*funkcja(t(i), y);
  y = yres(:, i);
end

ydok = -2.*sin(t) - cos(2.*t);

figure(1);
plot( ...
  t, ydok, 'b',             ...
  t, yres(1, :), 'r',       ...
  t, yres(2, :), 'g--');
grid on;
legend('ydok', 'ynum', 'dynum');
```

Przykład 2.

```
clear; clc; close all;

h = 0.001;          % Step.
time = 10;          % Time.
y = [-1; -2];       % Starting conditions.
t = 0:h:time;       % Full timestamp vector.

% Predeclare yres.
yres = zeros(2, length(t));

for i = 1:length(t)
  k1 = h.*funkcja(t(i), y);
  k2 = h.*funkcja(t(i) + h, y + k1);
```

```
  yres(:, i) = y + 0.5.*(k1 + k2);
  y = yres(:, i);
end

ydok = -2.*sin(t) - cos(2.*t);
figure(1);
plot(...
  t, ydok, 'b',           ...
  t, yres(1, :), 'r--',     ...
  t, yres(2, :), 'g--');
grid on;
legend('ydok', 'ynum', 'dynum');
```

Zadanie 2.

Nie zadano konkretnej funkcji, więc nasza funkcja wygląda tak:

```
function [dy] = funkcja(t,y)
  dy = [y(2); 4*sin(t) + 5*cos(2*t) + y(1)];
end
```

Rozwiązanie:

```
clear; clc; close all;

h = 0.001;        % Step.
time = 2;         % Time.
y = [4; 5];       % Initial conditions.
t = 0:h:time;     % Full timestamp vector.

% Predeclare yres.
yres = zeros(2, length(t));

for i = 1:length(t)
  k1 = funkcja(t(i), y);
  k2 = funkcja(t(i) + h, y + h*k1);
  yres(:, i) = y + (0.5*h).*(k1 + k2);
  y = yres(:, i);
end
% Wypisz.
figure(1)
plot( ...
    t,yres(1,:), ...
    'r',t,yres(2,:),'g--');
grid on;
legend('ynum','dynum');
```

Zadanie 3.

Funkcja:

```
function [dy] = funkcja(t, y)
  dy = [y(2); sin(t/2) + cos(t/2) - y(1)];
end
```

Metoda 1. Metoda Eulera.

```
clear; clc; close all;

h = 0.001;          % Step.
time = 2;           % Time.
y = [-1; -2];       % Initial conditions.
t = 0:h:time;       % Full timestamp vector.

% Predeclare yres.
yres = zeros(2, length(t));

for i = 1:length(t)
  yres(:, i) = y + h.*funkcja(t(i), y);
  y = yres(:, i);
end

% Exact solution.
ydok = (4*sin(t/2) + 4*cos(t/2) - 8*sin(t) - 7*cos(t)) / 3;

figure(1)
plot( ...
  t, ydok, 'b', ...
  t,yres(1,:),'r--', ...
  t,yres(2,:),'g--');
grid on;
legend('ydok', 'ynum','dynum');
```

Metoda 2. Metoda Eulera-Cauchy’ego.

```
clear; clc; close all;

h = 0.001;          % Step.
time = 2;           % Time.
y = [-1; -2];       % Initial conditions.
t = 0:h:time;       % Full timestamp vector.

% Predeclare yres.
yres = zeros(2, length(t));

for i = 1:length(t)
  k1 = funkcja(t(i), y);
  k2 = funkcja(t(i) + h, y + h*k1);
  yres(:, i) = y + (0.5*h).*(k1 + k2);
  y = yres(:, i);
end

% Exact solution.
ydok = (4*sin(t/2) + 4*cos(t/2) - 8*sin(t) - 7*cos(t)) / 3;

figure(1)
plot( ...
  t, ydok, 'b', ...
  t,yres(1,:),'r--', ...
```

```
  t,yres(2,:),'g--');
grid on;
legend('ydok', 'ynum','dynum');
```

Metoda 3. Metoda RK2.

```
clear; clc; close all;

h = 0.001;          % Step.
time = 2;           % Time.
y = [-1; -2];       % Initial conditions.
t = 0:h:time;       % Full timestamp vector.

% Predeclare yres.
yres = zeros(2, length(t));

for i = 1:length(t)
  k1 = h.*funkcja(t(i), y);
  k2 = h.*funkcja(t(i) + h, y + k1);
  yres(:, i) = y + 0.5.*(k1 + k2);
  y = yres(:, i);
end

% Exact solution.
ydok = (4*sin(t/2) + 4*cos(t/2) - 8*sin(t) - 7*cos(t)) / 3;

figure(1)
plot( ...
  t, ydok, 'b', ...
  t,yres(1,:),'r--', ...
  t,yres(2,:),'g--');
grid on;
legend('ydok', 'ynum','dynum');
```

Metoda 4. Metoda trapezów.

```
clear; clc; close all;

h = 0.001;          % Step.
time = 2;           % Time.
y = [-1; -2];       % Initial conditions.
t = 0:h:time;       % Full timestamp vector.

% Predeclare yres.
yres = zeros(2, length(t));

for i = 1:length(t)
  yres(:, i) = y + (h/2).*(funkcja(t(i), y) + funkcja(t(i) + h, y));
  y = yres(:, i);
end

% Exact solution.
ydok = (4*sin(t/2) + 4*cos(t/2) - 8*sin(t) - 7*cos(t)) / 3;
```

```
figure(1)
plot( ...
  t, ydok, 'b', ...
  t,yres(1,:),'r--', ...
  t,yres(2,:),'g--');
grid on;
legend('ydok', 'ynum','dynum');
```

Zadanie 4[1].

```
function [dy] = funkcja(t, y)
  dy = [-3*y(2)^2 / 3; sin(t)^2 + y(1)];
end
```

Metoda 1. Metoda Eulera.

```
clear; clc; close all;

h = 0.001;          % Step.
time = 2;           % Time.
y = [4; 5];         % Starting conditions.
t = 0:h:time;       % Full timestamp vector.

% Predeclare yres.
yres = zeros(2, length(t));

for i = 1:length(t)
  yres(:, i) = y + h.*funkcja(t(i), y);
  y = yres(:, i);
end

figure(1)
plot(t,yres(1,:),'r',t,yres(2,:),'g--');grid on;
legend('ynum','dynum');
```

Metoda 2. Metoda Eulera-Cauchy'ego.

```
clear; clc; close all;

h = 0.001;          % Step.
time = 2;           % Time.
y = [4; 5];         % Starting conditions.
t = 0:h:time;       % Full timestamp vector.

% Predeclare yres.
yres = zeros(2, length(t));

for i = 1:length(t)
  k1 = funkcja(t(i), y);
  k2 = funkcja(t(i) + h, y + h*k1);
  yres(:, i) = y + (0.5*h).*(k1 + k2);
  y = yres(:, i);
end

figure(1)
```

```
plot(t,yres(1,:),'r',t,yres(2,:),'g--');grid on;
legend('ynum','dynum');
```

Metoda 3. Metoda Runggego-Kutty RK2.

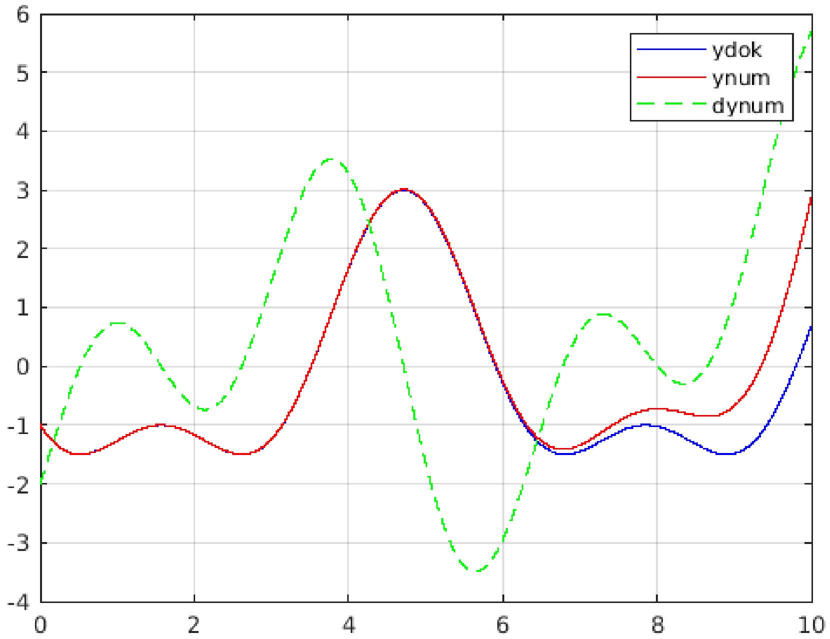
```
clear; clc; close all;

h = 0.001;          % Step.
time = 2;           % Time.
y = [4; 5];         % Starting conditions.
t = 0:h:time;       % Full timestamp vector.

% Predeclare yres.
yres = zeros(2, length(t));

for i = 1:length(t)
  k1 = h.*funkcja(t(i), y);
  k2 = h.*funkcja(t(i) + h, y + k1);
  yres(:, i) = y + 0.5.*(k1 + k2);
  y = yres(:, i);
end

figure(1)
plot(t,yres(1,:),'r',t,yres(2,:),'g--');grid on;
legend('ynum','dynum');
```

Metoda 4. Metoda trapezów.

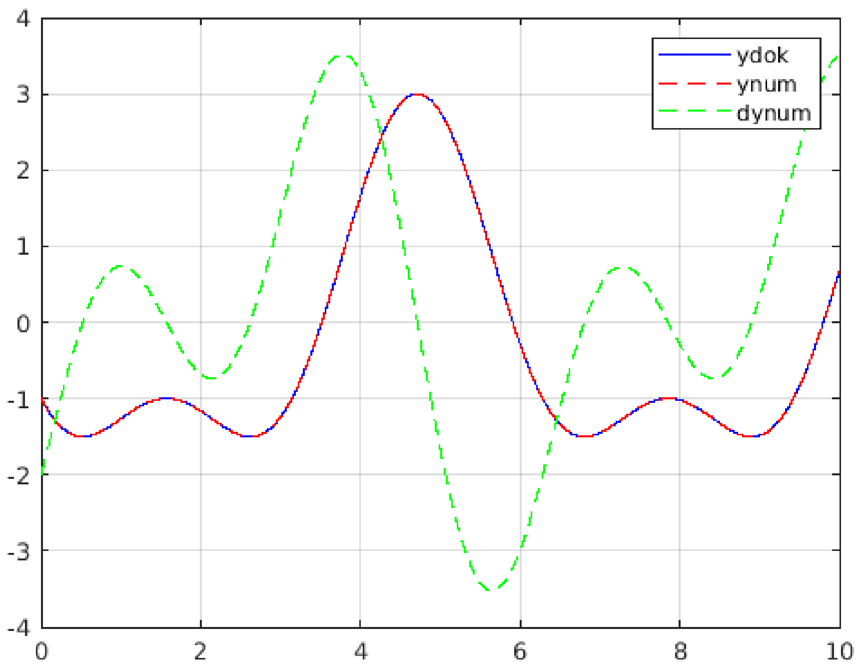
```
clear; clc; close all;

h = 0.001;          % Step.
time = 2;           % Time.
y = [4; 5];         % Starting conditions.
t = 0:h:time;       % Full timestamp vector.

% Predeclare yres.
yres = zeros(2, length(t));

for i = 1:length(t)
  yres(:, i) = y + (h/2).*(funkcja(t(i), y) + funkcja(t(i) + h, y));
  y = yres(:, i);
end

figure(1)
plot(t,yres(1,:),'r',t,yres(2,:),'g--');grid on;
legend('ynum','dynum');
```

Wyniki

Zadanie 1.

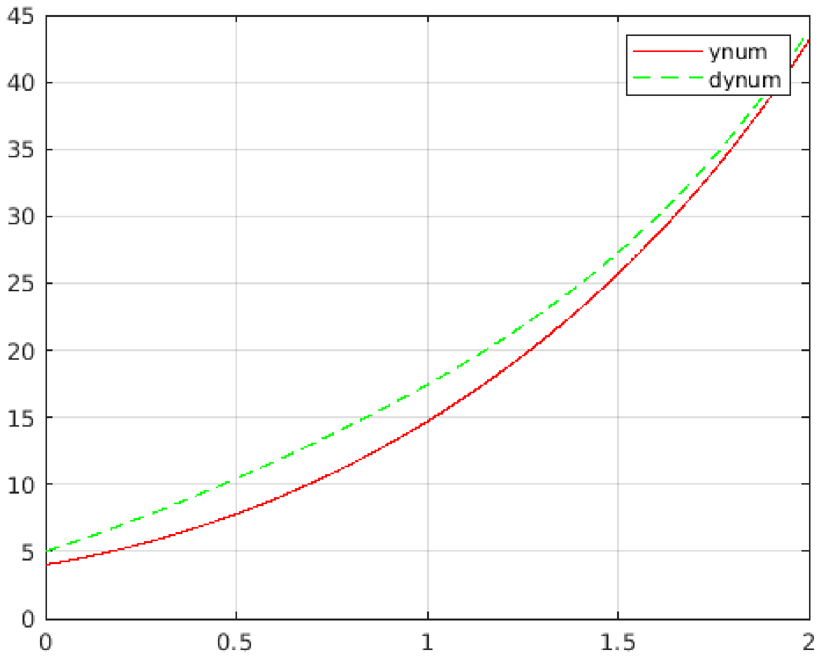


Wykres 1: Zadanie 1. Przykład 1.



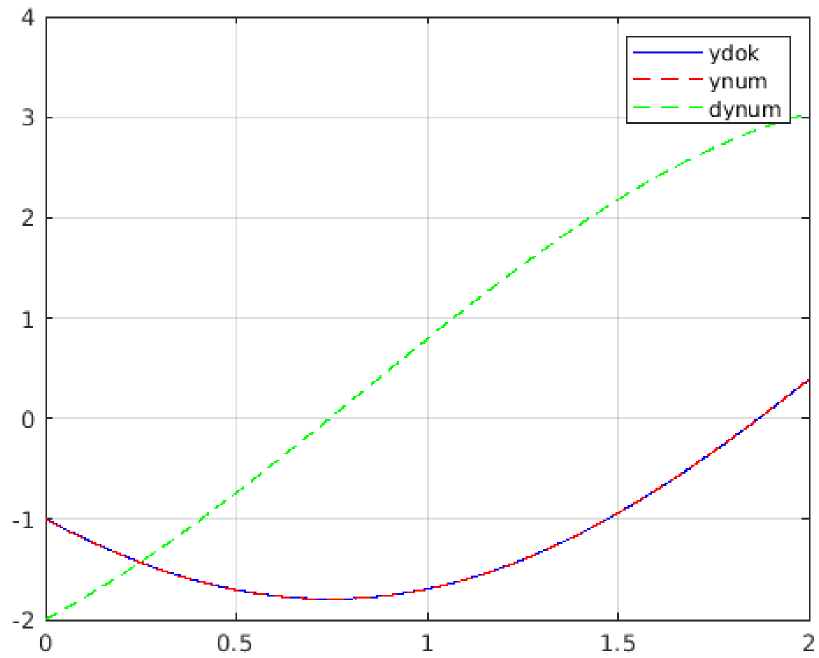
Wykres 2: Zadanie 2. Przykład 2.

Zadanie 2.

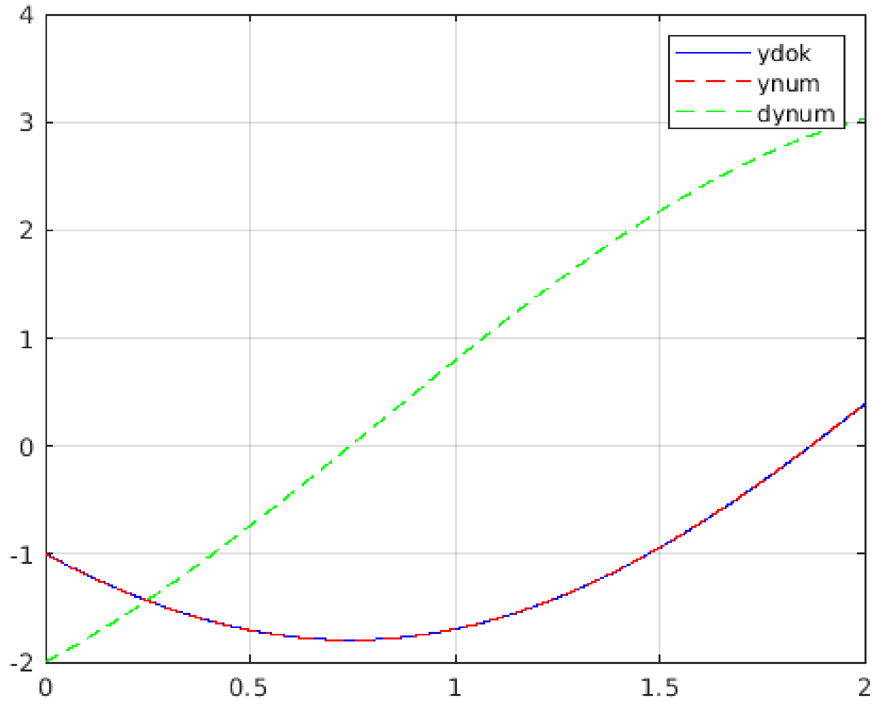


Wykres 3: Zadanie 2.

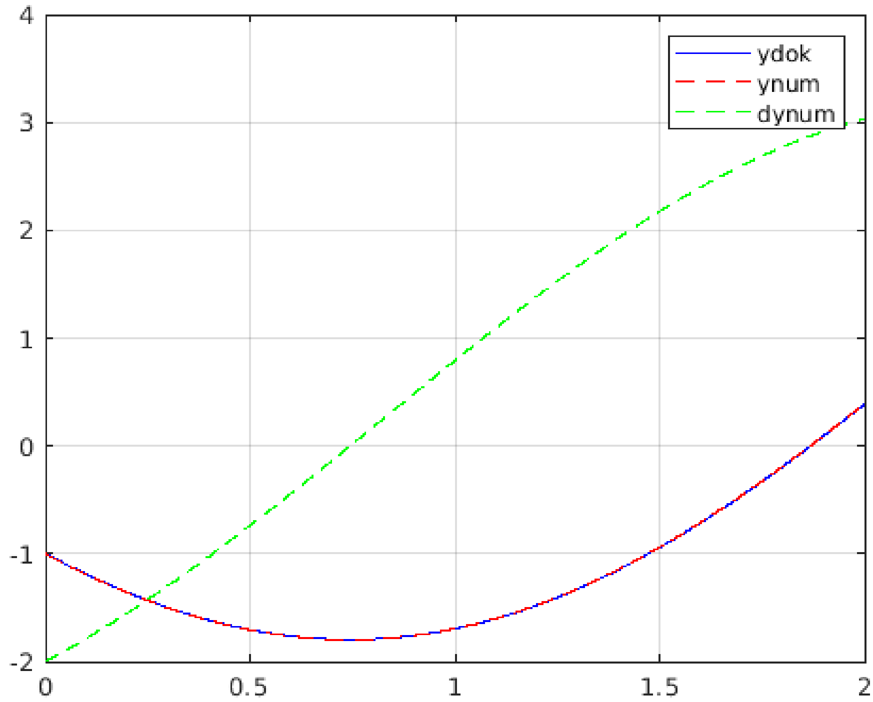
Zadanie 3.



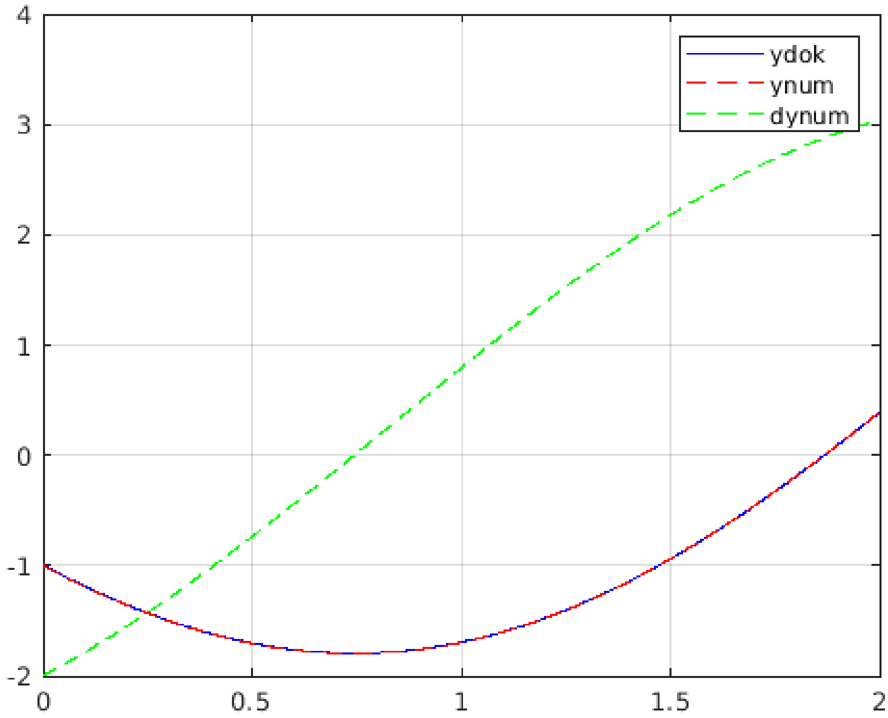
Wykres 4: Zadanie 3. Metoda Eulera.



Wykres 5: Zadanie 3. Metoda Eulera-Cauchy'ego.

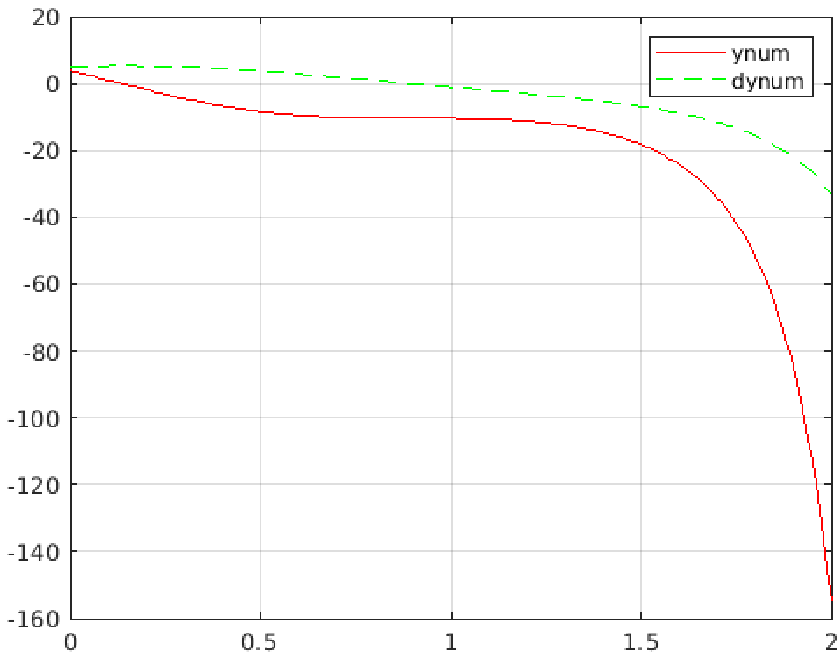


Wykres 6: Zadanie 3. Metoda Rungego-Kutty.

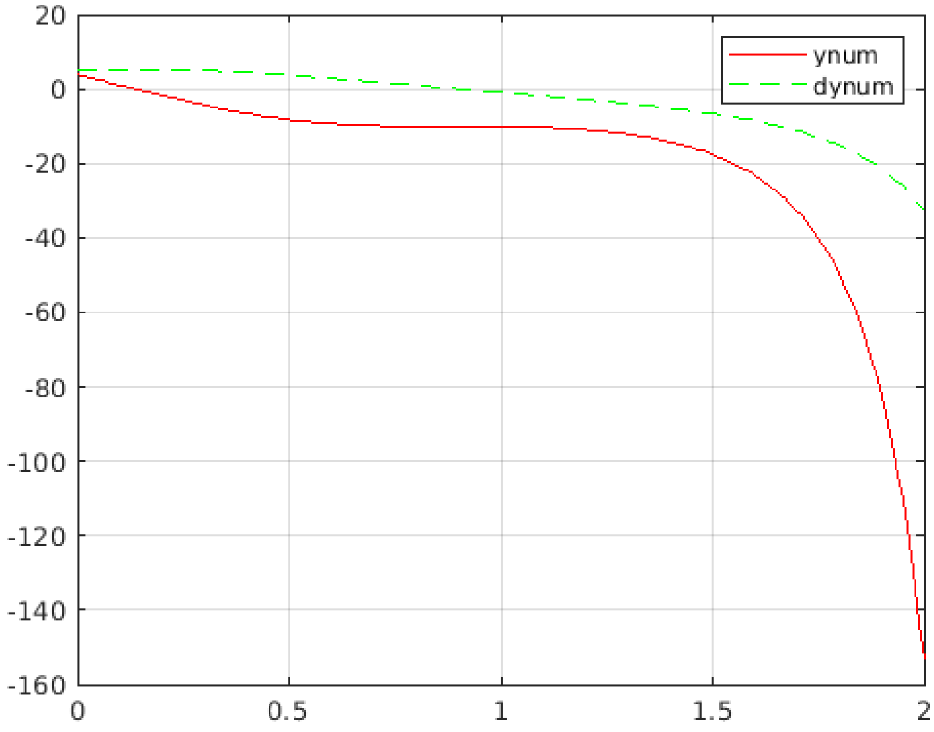


Wykres 7: Zadanie 3. Metoda trapezów.

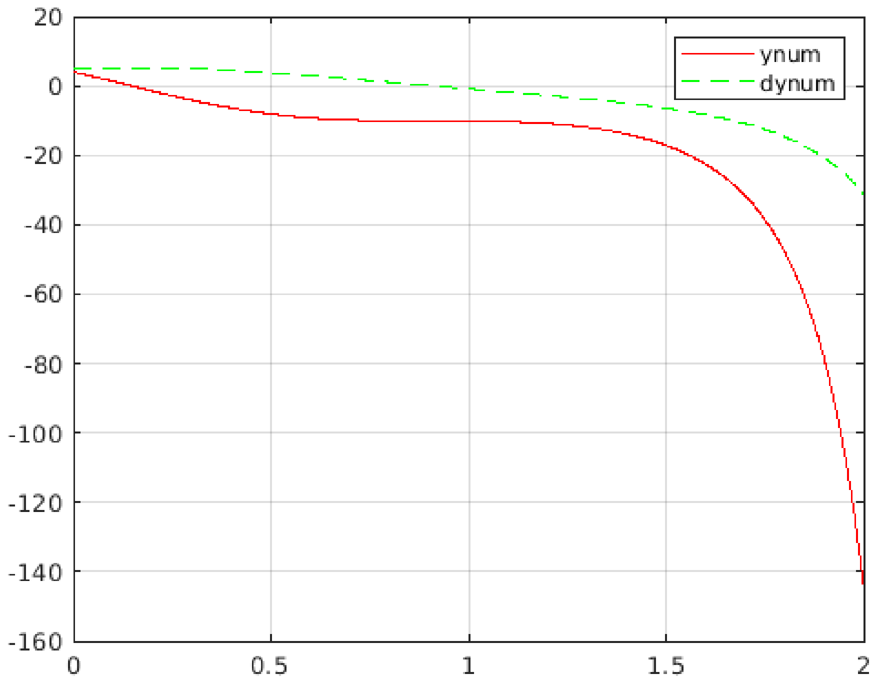
Zadanie 4.



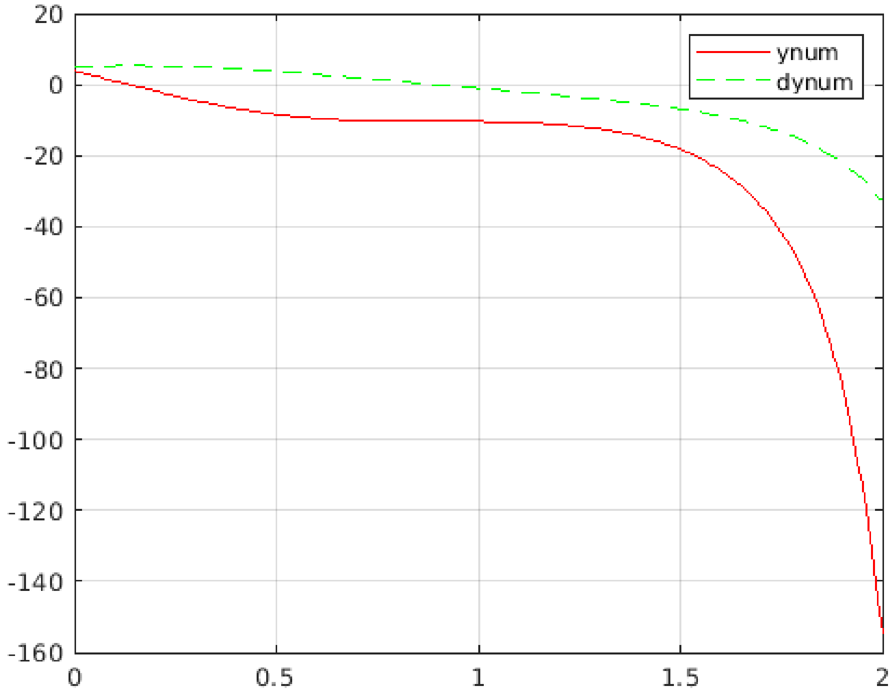
Wykres 8: Zadanie 4. Metoda Eulera.



Wykres 9: Zadanie 4. Metoda Eulera-Cauchy'ego.



Wykres 10: Zadanie 4. Metoda RK2.



Wykres 11: Zadanie 4. Metoda trapezów.

Opis działania programu

Metoda rozwiązywania równań różniczkowych metodami numerycznymi jest podobna dla wszystkich sposobów. W pliku **`funkcja.m`** znajduje się reprezentacja zadanego równania różniczkowego w postaci pionowego wektora. W każdym kolejnym elemencie tego wektora znajduje się kolejna, niższa, całka. Jako argumenty, ta funkcja przyjmuje czas **`t`** oraz **`y`**, co jest wynikiem poprzedniej iteracji tejże funkcji (a przy pierwszej iteracji – warunkami początkowymi), zwracanymi przez **`[dy]`**.
Następnie każdy program zawiera warunki początkowe, tj. **`y`**; oraz krok iteracji **`h`**, czas **`time`**, wektor czasu **`t`**.
Dla szybkości wykonywania obliczeń predeklarowany jest wektor **`yres`**, który stanowi rozwiązanie równania różniczkowego, które zawarte jest w pętli, odbywającej się dla całego czasu **`t`**. W niej zawarte jest rozwiązanie z użyciem odpowiednich wzorów (Eulera, Eulera-Cauchy'ego, RK2, trapezów).

Metoda Eulera:

$$x_{k+1}=x_k+h\cdot f(x_k,t_k)$$

Metoda trapezów:

$$x_{k+1}=x_k+\frac{h}{2}\cdot[f(x_k,t_k)+f(x_k,t_{k+1})]$$

Metoda RK2:

$$x_{k+1}=x_k+\frac{1}{2}(k_1+k_2)$$
$$k_1=h\cdot f(t_k,x_k)$$
$$k_2=h\cdot f(t_k+h,x_k+k_1)$$

Metoda Eulera-Cauchy'ego:

$$x_{k+1}=x_k+k_2$$
$$k_1=h\cdot f(t_k,x_k)$$
$$k_2=h\cdot f\left(t_k+\frac{1}{2}h,x_k+\frac{1}{2}k_1\right)$$

Wnioski

Istnieje wiele metod rozwiązywania równań różniczkowych. Znajdują one swoje zastosowanie w sytuacjach, gdy dokładne rozwiązywanie równań jest niemożliwe. Wiele z tych metod osiąga podobne wyniki, gdy skala jest mała, jednak różnią się nieznacznie nie tylko między sobą, ale też rozwiązaniem dokładnym. Stąd, znajomość i umiejętność zastosowania odpowiedniej metody, której złożoność obliczeniowa jest adekwatna do zadanego problemu oraz uzyskuje przybliżenie o akceptowalnej niedokładności, jest istotne w pracy inżyniera.

1.Pierwotnie, na zajęciech Pan Gutenko zarządził, że do sprawozdania należy dołączyć rozwiązania oraz opracowania zadania pierwszego, przepracowane przykłady z instrukcji, oraz rozwiązania wszystkimi metodami jednej funkcji. Jest to zawarte jako Zadanie 4. Zadanie 2 oraz zadanie 3 są tutaj uzupełnieniem wcześniej nie dołączonym do sprawozdania.