Ćwiczenie 13 – całkowanie równań różniczkowych metodą Eulera i Runge-Kutty

Treść zajęć:
metoda Eulera i Rungego-Kutty: przyjęcie poziomu dyskretyzacji, zastosowanie metody strzału sterowanej bisekcja

Cel zajęć:
poznanie metod całkowania numerycznego, zapis operatorów różnicowych

Wzory, algorytm:
równanie liny

$$\frac{dy'}{\sqrt{1+y'^2}} = \frac{q}{H}dx$$

z warunkami brzegowymi:

$$y_0 = 0, \;\; y_i = f$$

po podstawieniu: $y' \equiv t$ można zastąpić układem dwóch równań pierwszego rzędu

$$t' = \frac{q}{H}\sqrt{1+t^2}, \;\; y' = t$$

z warunkami początkowymi:

$$y_0 = 0, \qquad t_0 = x$$

gdzie *x* należy tak dobrać, aby $y_i = f$.
<u>Metoda Eulera:</u>
Po dyskretyzacji równania zapiszą się:

$$y_{i+1} = y_i + ht_i$$
$$t_{i+1} = t_i + h\frac{q}{H}\sqrt{1+t_i^2}$$

<u>Metoda Runge-Kutty:</u>
Po dyskretyzacji równania zapiszą się:

$$k_1 = \frac{hq}{H}\sqrt{1+t_n^2}, \;\; k_2 = \frac{hq}{H}\sqrt{1+(t_n+0.5k_1)^2}, \;\; k_3 = \frac{hq}{H}\sqrt{1+(t_n+0.5k_2)^2}, \;\; k_4 = \frac{hq}{H}\sqrt{1+(t_n+k_3)^2}$$

$$t_{n+1} = t_n + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

$$l_1 = ht_n, \;\; l_2 = \frac{h}{2}(t_n + t_{n+1}), \;\; l_3 = \frac{h}{2}(t_n + t_{n+1}), \;\; l_4 = ht_{n+1}$$

$$y_{n+1} = y_n + \frac{1}{6}(l_1 + 2l_2 + 2l_3 + l_4)$$

Problem:
Obliczyć równanie zwisu liny dla *l* = 25 [m], *f* = 7 [m], *q* = 42 [N/m], *H* = 420 [N].
Rozwiązanie
(kod matlaba)

```
% Adam Zaborski, ćw. 14 - całkowanie metodą Eulera + metoda strzału sterowana bisekcją
clc
clear all
format compact, format long
l = 25; f = 7; q = 42; H = 420;
H_q = H / q;
np = 100;       % przyjęty poziom dyskretyzacji
```

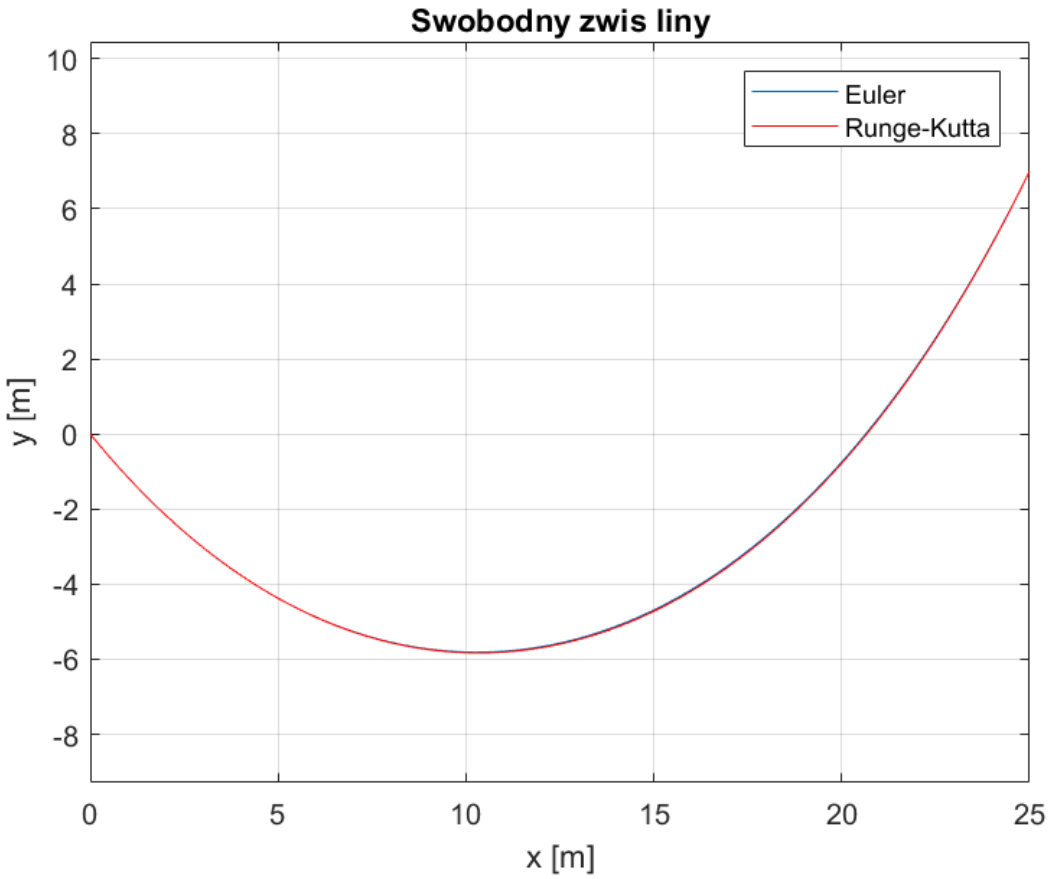
```
krok = l / np;
y(1) = 0;
t(1) = 0;
x_1 = t(1);
for i = 1: 1: np
    y(i+1) = y(i) + krok * t(i);
    t(i+1) = t(i) + krok / H_q * (1 + (t(i))^2)^0.5;
end
a = y(np+1);
y(1) = 0;
t(1) = -1.5;
x_2 = t(1);
for i = 1: 1: np
    y(i+1) = y(i) + krok * t(i);
    t(i+1) = t(i) + krok / H_q * (1 + (t(i))^2)^0.5;
end
b = y(np+1);
for j = 1: 1: 20
    y(1) = 0;
    t(1) = 0.5 * (x_1 + x_2);
    x = t(1);
    for i = 1: 1: np
        y(i+1) = y(i) + krok * t(i);
        t(i+1) = t(i) + krok / H_q * (1 + (t(i))^2)^0.5;
    end
    c = y(np+1);
    if c - f > 0
        x_1 = x;
    else
        x_2 = x;
    end
end
c
for i = 1: 1: np+1
    xx(i) = krok * (i-1);
end
plot( xx, y )
hold on
xlabel('x [m]')
ylabel('y [m]')
title('Swobodny zwis liny')
axis equal


% ---------------------------------------------------------------------------
% Adam Zaborski, ćw. 14 - całkowanie metodą Runge-Kutty + metoda strzału
% sterowana bisekcją
clear all
format compact, format long
l = 25; f = 7; q = 42; H = 420;
```

```
H_q = H / q;
np = 100;       % przyjęty poziom dyskretyzacji
krok = l / np;
y(1) = 0;
t(1) = 0;
x_1 = t(1);
for i = 1: 1: np
   k1 = krok * sqrt( 1 + ( t(i) )^2 ) / H_q;
   k2 = krok * sqrt( 1 + (t(i) + k1 / 2 )^2 ) / H_q;
   k3 = krok * sqrt( 1 + (t(i) + k2 / 2 )^2 ) / H_q;
   k4 = krok * sqrt( 1 + (t(i) + k3 )^2 ) / H_q;
   t(i+1) = t(i) + k1 / 6 + k2 / 3 + k3 / 3 + k4 / 6;
   m1 = krok * t(i);
   m2 = krok * ( ( t(i) + t(i+1) ) / 2 );
   m3 = krok * ( ( t(i) + t(i+1) ) / 2 );
   m4 = krok * ( t(i+1) );
   y(i+1) = y(i) + m1 / 6 + m2 / 3 + m3 / 3 + m4 / 6;
end
a = y(np+1);
y(1) = 0;
t(1) = -1.5;
x_2 = t(1);
for i = 1: 1: np
   k1 = krok * sqrt( 1 + ( t(i) )^2 ) / H_q;
   k2 = krok * sqrt( 1 + (t(i) + k1 / 2 )^2 ) / H_q;
   k3 = krok * sqrt( 1 + (t(i) + k2 / 2 )^2 ) / H_q;
   k4 = krok * sqrt( 1 + (t(i) + k3 )^2 ) / H_q;
   t(i+1) = t(i) + k1 / 6 + k2 / 3 + k3 / 3 + k4 / 6;
   m1 = krok * t(i);
   m2 = krok * ( ( t(i) + t(i+1) ) / 2 );
   m3 = krok * ( ( t(i) + t(i+1) ) / 2 );
   m4 = krok * ( t(i+1) );
   y(i+1) = y(i) + m1 / 6 + m2 / 3 + m3 / 3 + m4 / 6;
end
b = y(np+1);
for j = 1: 1: 20
   y(1) = 0;
   t(1) = 0.5 * (x_1 + x_2);
   x = t(1);
   for i = 1: 1: np
   k1 = krok * sqrt( 1 + ( t(i) )^2 ) / H_q;
   k2 = krok * sqrt( 1 + (t(i) + k1 / 2 )^2 ) / H_q;
   k3 = krok * sqrt( 1 + (t(i) + k2 / 2 )^2 ) / H_q;
   k4 = krok * sqrt( 1 + (t(i) + k3 )^2 ) / H_q;
   t(i+1) = t(i) + k1 / 6 + k2 / 3 + k3 / 3 + k4 / 6;
   m1 = krok * t(i);
   m2 = krok * ( ( t(i) + t(i+1) ) / 2 );
   m3 = krok * ( ( t(i) + t(i+1) ) / 2 );
   m4 = krok * ( t(i+1) );
   y(i+1) = y(i) + m1 / 6 + m2 / 3 + m3 / 3 + m4 / 6;
```

```
    end
    c = y(np+1);
    if c - f > 0
        x_1 = x;
    else
        x_2 = x;
    end
end
c
for i = 1: 1: np+1
    xx(i) = krok * (i-1);
end
plot( xx, y, 'red' )
legend Euler Runge-Kutta
axis equal
grid on
```

Wyniki



Jak widać, wyniki z obu metod praktycznie pokrywają się.