Politechnika Wrocławska

# MATLAB i równania różniczkowe

Wojciech Myszka

31 października 2024

HR EXCELLENCE IN RESEARCH


# Najpierw symbolicznie

# Równanie pierwszego rzędu I

Właściwie równie proste jak każde inne równanie...

$$\frac{dy}{dt} = ty$$

Szukamy funkcji $y(t)$, która będzie rozwiązaniem równania. Zadeklarujemy ją jako obiekt symboliczny

```
syms y(t)
```

Do liczenia pochodnych służy funkcja `diff()`, więc nasze równanie można zapisać jako:

```
ode = diff(y,t) == y*t
```

# Równanie pierwszego rzędu II

$$\text{ode(t)} = \frac{\partial}{\partial t}\, y\,(t) = t\, y\,(t)$$

Do rozwiązywania równań różniczkowych służy funkcja `dsolve()`.

```
S = dsolve(ode)
```

$$\text{S} = C_1\, \mathrm{e}^{\frac{t^2}{2}}$$

# Nadanie wartości stałej C

```
var = subs(S,[sym("C1")],[1])
```

var =

$$e^{\frac{t^2}{2}}$$

# Warunki początkowe

Definiujemy warunek początkowy

```
cond = y(0) == 2;
ySol = dsolve(ode, cond)
```

$$\text{ySol} = 2\,\mathrm{e}^{\frac{t^2}{2}}$$

Warunek początkowy może wyglądać tak:

```
cond = y(1) == 2;
ySol = dsolve(ode, cond)
```

$$\text{ySol} = 2\,\mathrm{e}^{-\frac{1}{2}}\,\mathrm{e}^{\frac{t^2}{2}}$$

# Równanie różniczkowe drugiego stopnia I

Zacznijmy od równania $\frac{d^2y}{dt^2} = a^2 y$ z warunkami początkowymi $y(0) = b$ i $y'(0) = 1$

```
syms y(t) a b
eqn = diff(y,t,2) == a^2*y;
Dy = diff(y,t);
cond = [y(0)==b, Dy(0)==1];
ySol(t) = dsolve(eqn,cond)
```

ySol(t) =

$$\frac{e^{a\,t}\,(a\,b+1)}{2\,a} + \frac{e^{-a\,t}\,(a\,b-1)}{2\,a}$$

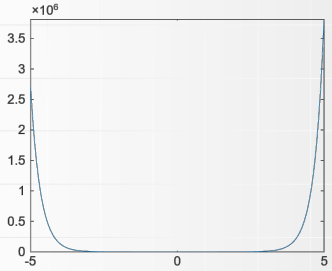
```
var = subs(ySol,[a,b],[3,2])
```

# Równanie różniczkowe drugiego stopnia II

$$\mathrm{var}(t) =$$

$$\frac{5\,e^{-3\,t}}{6} + \frac{7\,e^{3\,t}}{6}$$

```
fplot(var)
```



# Układ równań I

Mamy układ równań o postaci

$$\begin{aligned} \frac{dy}{dt} &= z \\ \frac{dz}{dt} &= -y. \end{aligned}$$

```
syms y(t) z(t)
eqns = [diff(y,t) == z, diff(z,t) == -y];
S = dsolve(eqns)
```

```
S = struct with fields:
    z: C2*cos(t) - C1*sin(t)
    y: C1*cos(t) + C2*sin(t)
```

Żeby wyodrębnić rozwiązania ze struktury

```
ySol(t) = S.y
```

# Układ równań II

$$\text{ysol(t)} = C_1 \cos(t) + C_2 \sin(t)$$

```
zSol(t) = S.z
```

$$\text{zSol(t)} = C_2 \cos(t) - C_1 \sin(t)$$

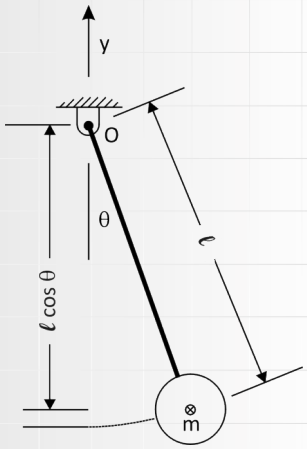
# Wahadło I

Ruch wahadła (matematycznego) może być opisany równaniem:

$$ml^2\ddot{\theta} + \mu\dot{\theta} + mgl\sin(\theta) = 0$$

gdzie:

- $\theta$ — kąt wychylenia;
- $m$ — masa wahadła,
- $l$ — długość wahadła,
- $\mu$ — współczynnik oporu powietrza (tarcia),
- $g$ — stałą grawitacyjna.




Wszystko wydaje się dosyć proste...

# Wahadło II

```
syms m l mu g theta(t)
eqn = m*l^2*diff(theta,t,2) + mu*diff(theta,t,1) +
      m*g*l*sin(theta) == 0
```

$$\mathrm{eqn}(t) = m\,l^2\,\frac{\partial^2}{\partial t^2}\,\theta(t) + \mu\,\frac{\partial}{\partial t}\,\theta(t) + g\,m\,\sin(\theta(t))\,l = 0$$

```
dsolve(eqn)
```

```
Warning: Unable to find symbolic solution.
ans =
[ empty sym ]
```

Przyjrzymy się temu bliżej w demonstracji...


# Teraz numerycznie

# Konwersja równania II stopnia do układu równań I stopnia I

Zaczynamy od gotowego już równania symbolicznego

```
syms m l mu g theta(t)
eqn = m*l^2*diff(theta,t,2)+mu*diff(theta,t,1)+m*g*l*sin(theta) =
```

$$\text{eqn(t)} = m\,l^2\,\frac{\partial^2}{\partial t^2}\,\theta\,(t) + \mu\,\frac{\partial}{\partial t}\,\theta\,(t) + g\,m\,\sin\left(\theta\,(t)\right)\,l = 0$$

Uprośćmy równanie tak, żeby mieć po lewej stronioe drugą pochodną (użyjemy polecenia `isolate()`.

```
eqn = isolate(eqn,diff(theta,2))
```

# Konwersja równania II stopnia do układu równań I stopnia II

$$\text{eqn} = \frac{\partial^2}{\partial t^2}\,\theta(t) = -\frac{\mu\,\frac{\partial}{\partial t}\,\theta(t) + g\,l\,m\,\sin(\theta(t))}{l^2\,m}$$

Teraz musimy, metodą zmiany zmiennych, zamienić te równanie drugiego stopnia na układ dwu równań pierwszego stopnia.

Standardowe postępowanie sprowadza się do tego, że podstawiamy:

$$y_1(t) = \theta(t)$$

i

$$y_2(t) = \frac{\partial}{\partial t}\theta(t)$$

a nasze równani zamienia się w układ

# Konwersja równania II stopnia do układu równań I stopnia III

$$\frac{dy_2}{dt} = -\frac{1}{ml^2}\left(\mu y_2 + glm\sin(y_1)\right)$$

i

$$\frac{dy_1}{dt} = y_2$$

Najpierw nadajmy zmiennym wartości parametrów (żeby uprościć sobie życie)

```
eqn = subs(eqn,[m,l,g,mu],[1,1,9.81,0.1])
```

# Konwersja równania II stopnia do układu równań I stopnia IV

$$\text{eqn} = \frac{\partial^2}{\partial t^2}\,\theta(t) = -\frac{\frac{\partial}{\partial t}\,\theta(t)}{10} - \frac{981\,\sin(\theta(t))}{100}$$

Teraz skorzystamy z funkcji `odeToVectorField()`, która redukuje rząd równania różniczkowego do pierwszego stopnia. (Metody numeryczne najlepiej sobie radzą z równaniami pierwszego stopnia.)

```
[V] = odeToVectorField(eqn)
```

$$\mathrm{V} = \begin{pmatrix} Y_2 \\ -\frac{981\,\sin(Y_1)}{100} - \frac{Y_2}{10} \end{pmatrix}$$

Na podstawie tablicy `V` musimy wygenerować odpowiednie funkcje, żeby rozwiązać te równanie numerycznie. Procedura `matlabFunction()` konwertuje wyrażenie symboliczne do funkcji (anonimowej lub nie)

```
M = matlabFunction(V,'Vars',{'t','Y'})
```

# Konwersja równania II stopnia do układu równań I stopnia V

```
M = function_handle with value:
    @(t,Y)[Y(2);sin(Y(1)).*(-9.81e+2./1.0e+2)-Y(2)./1.0e+1]
```

Aby rozwiązać numerycznie uzywamy procedury `ode45()`. Parametry tej funkcji to:

1. Funkcja opisująca nasz układ równań
2. Zakres czasu dla którego interesuje nas rozwiązanie
3. Wartości początkowe (w tym przypadku wektor o dwu składowych: wychylenie początkowe i prędkość początkowa)

```
rozwiazanie = ode45(M, [0 20], [pi 0]);
```

Reszta w demonstracji

Politechnika Wrocławska

# Simulink

1. Wyobraźmy sobie, że mamy magiczne „coś”, które mając na wejściu funkcję, na wyjściu generuje całkę z niej.
   - gdy na wejściu pojawi się druga pochodna — na wyjściu będzie pierwsza
   - gdy na wejściu pojawi się pierwsza — na wyjściu będzie orygina