Podstawy Sterowania Optymalnego
Labolatorium 8

Modelowanie układów nieliniowych

Prowadzący: mgr inż. Krzysztof Hałas
Wykonał: Ryszard Napierała

8 stycznia 2022

Zadanie 2

1. **Zaimportować biblioteki *numpy, scipy.integrate.odeint* oraz *matplotlib.pyplot*.**

```
from typing import List
import numpy as np
from scipy.integrate import odeint
import matplotlib.pyplot as plt
```

2. **Utworzyć model wyznaczający wartość $\frac{dy(t)}{dt}$, na podstawie równania.**

```
RESOLUTION = 100

def model(y: np.ndarray, t:float) -> List:
    return [t**2]
```

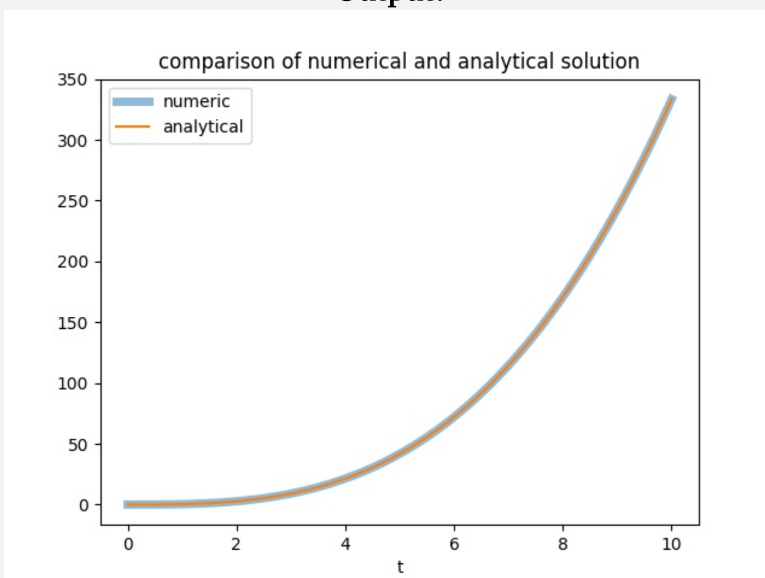
3. **Utworzyć wektor czasu $t \in (0, 10)s$. Dla tak opisanych chwili czasowych wyznaczyć numerycznie rozwiązanie równania, dla zerowych warunków początkowych ($c = 0$). Wykorzystać polecenie *odeint*.**

```
t = np.linspace(0, 10, RESOLUTION)

res_n = odeint(model, [0], t)
```

4. **Utworzyć zmienną, która odpowiada analitycznemu rozwiązaniu. Porównać na wykresie rozwiązanie równania uzyskane numerycznie (*odeint*) i analitycznie.**

```
res_a = t**3/3

plt.plot(t, res_n, linewidth=5, alpha=0.5)
plt.plot(t, res_a)
plt.title("comparison of numerical and analytical solution")
plt.legend(['numeric', 'analytical'])
plt.xlabel("t")
plt.show()
plt.close()
```

Output:



Czy rozwiązania się pokrywają? Jaki rodzaj numerycznego wyznaczania rozwiązania równania różniczkowego został użyty?
Wykresy się pokrywają
Zostało użyte całkowanie układu równań różniczkowych zwyczajnych

Zadanie 3

1. **Zaimportować biblioteki *numpy, scipy.integrate.odeint* oraz *matplotlib.pyplot*.**

```
from typing import Tuple
import numpy as np
from scipy.integrate import odeint
import matplotlib.pyplot as plt
```

2. **Przyjąć następujące wartości zmiennych $k_p = 2$, $\omega = 4$, $\zeta = 0.25$, $u(t) = \mathbb{1}(t)$.**

```
RESOLUTION = 100

kp = 2
omega = 4
zeta = 0.25
u = 1
```

3. **Utworzyć model wyznaczający wartość $\frac{d^2y(t)}{dt^2}$.**

```
def model(
    x:np.ndarray,
    t:float,
    kp:float,
    omega:float,
    zeta:float,
    u:float
) -> Tuple[float, float]:
    x1, x2 = x
    dxdt1 = x2
    dxdt2 = -(2*zeta*x2 + np.sqrt(x1))/omega + kp*u/omega**2
    return dxdt1, dxdt2
```

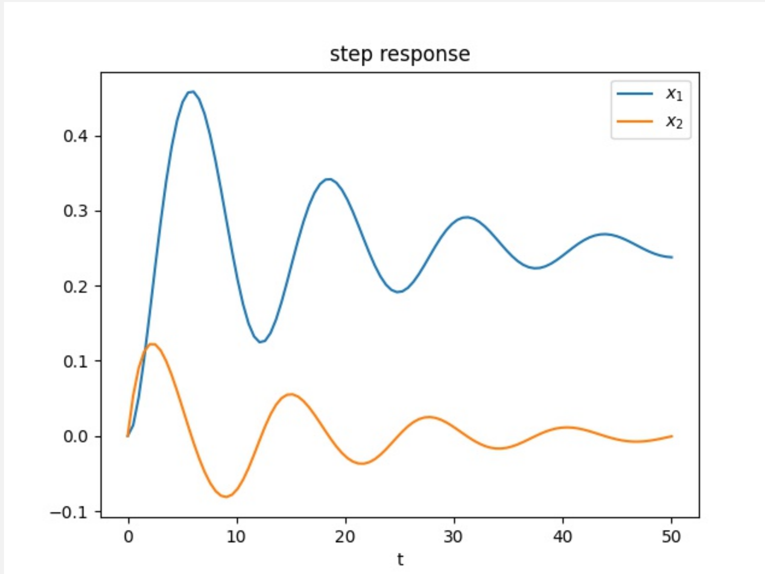
4. **Utworzyć wektor czasu $t \in (0, 50)s$. Dla tak opisanych chwili czasowych wyznaczyć numerycznie rozwiązanie równania wykorzystując polecenie *odeint*.**

```
t = np.linspace(0, 50, RESOLUTION)

res = odeint(model, (0, 0), t, args=(kp, omega, zeta, u))
```

5. **Wyświetlić rozwiązanie równania w funkcji czasu.**

```
plt.plot(t, res)
plt.title('step response')
plt.legend(['$x_1$', '$x_2$'])
plt.xlabel("t")
plt.show()
plt.close()
```

Output:



Jaki jest charakter odpowiedzi układu na wymuszenie skokowe?
Odpowiedź układu ma charakter liniowych gasnących oscylacji

Zadanie 4

1. **Przyjąć następujące wartości zmiennych** $k_p = 2$, $T = 2$, $k_o b = 4$, $x(t) = \mathbb{1}(t)$.

```
RESOLUTION = 100
kp = 2
T = 2
kob = 4
x = 1
```

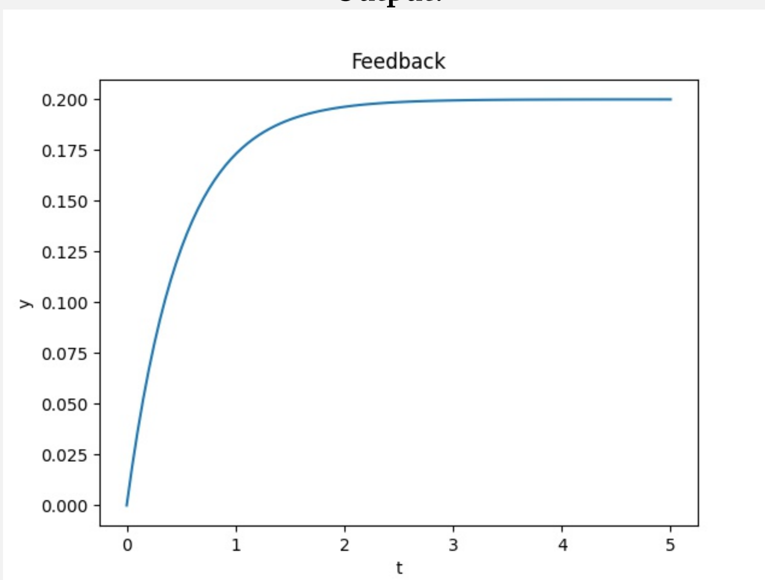
2. **Utworzyć funkcję *feedback(t,y)* opisującą rozważany schemat blokowy.**

```
def feedback(
    y:np.ndarray,
    t:float,
    kp:float,
    T:float,
    kob:float,
    x:float
) -> np.ndarray:
    u = kp*(x - y)
    u_clip = np.clip(u, -0.1, 0.1)
    dy = kob*u_clip - T*y
    return dy
```

3. **Wyznaczyć odpowiedź układu zamkniętego przez zastosowanie funkcji odeint. Wykreślić jej przebieg w funkcji czasu.**

```
t = np.linspace(0, 5, RESOLUTION)
res = odeint(feedback, [0], t, args=(kp, T, kob, x))

plt.plot(t, res)
plt.title("Feedback")
plt.xlabel("t")
plt.ylabel("y")
plt.show()
plt.close()
```

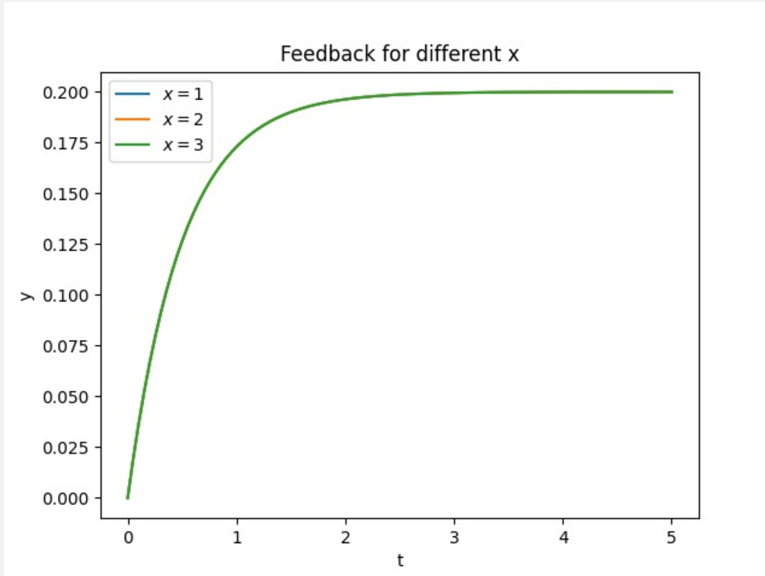
Output:



4. **Zmieniać wartość sygnału zadanego $x(t) = 1, 2, 3 * \mathbb{1}(t)$ i obserwować stany ustalone odpowiedzi układu.**

```
for x in (1, 2, 3):
    res = odeint(feedback, [0], t, args=(kp, T, kob, x))
    plt.plot(t, res)

plt.title("Feedback for different x")
plt.xlabel("t")
plt.ylabel("y")
plt.legend(["$x=1$", "$x=2$", "$x=3$"])
plt.show()
plt.close()
```

Output:



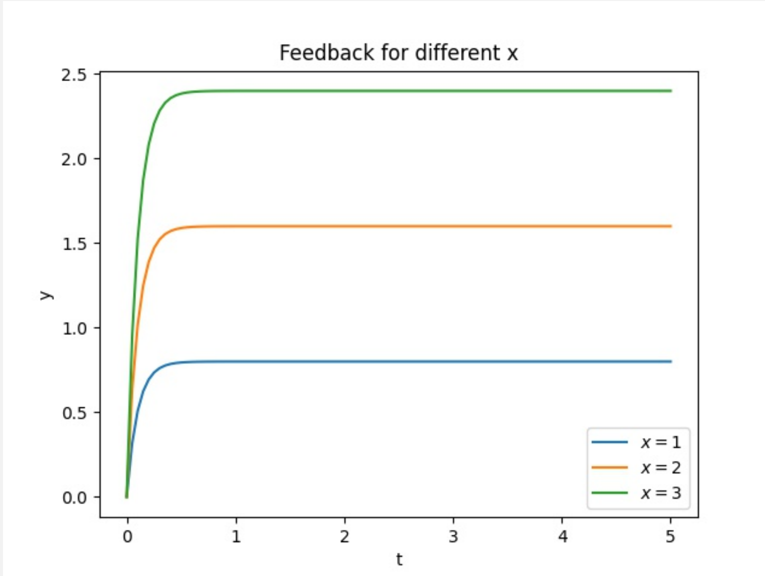
Czy zachowana jest zasada superpozycji i skalowania?
Zasada superpozycji i skalowania została zachowana.

Czy układ ma charakter liniowy?
Układ ma charakter liniowy

5. **Zmienić postać funkcji *feedback(t,y)* tak, aby sygnał sterujący nie był ograniczony. Wyświetlić odpowiedzi układu dla różnych stałowartościowych sygnałów zadanych $x(t)$.**

```
def feedback(
    y:np.ndarray,
    t:float,
    kp:float,
    T:float,
    kob:float,
    x:float
) -> np.ndarray:
    u = kp*(x - y)
    dy = kob*u - T*y
    return dy

for x in (1, 2, 3):
    res = odeint(feedback, [0], t, args=(kp, T, kob, x))
    plt.plot(t, res)

plt.title("Feedback for different x")
plt.xlabel("t")
plt.ylabel("y")
plt.legend(["$x=1$", "$x=2$", "$x=3$"])
plt.show()
plt.close()
```

Output:



Czy układ ma charakter liniowy?
Układ ma charakter liniowy

Zadanie 5*

1. **Przedstawić równanie opisujące wahadło w postaci dwóch równań pierwszego rzędu.**

$$\ddot{\Theta} = \frac{1}{J}\tau_m - \frac{d}{J}\dot{\Theta} - \frac{mgR}{J}\sin(\Theta)$$

$$x_1 = \Theta$$

$$\dot{x_1} = x_2$$

$$\dot{x_2} = \frac{1}{J}\tau_m - \frac{d}{J}x_2 - \frac{mgR}{J}\sin(x_1)$$

2. **Utworzyć funkcję wyznaczającą $\frac{d^2}{dt^2}\Theta(t)$. Jako sygnał wejściowy przyjąć $\tau_m = A\cos(\omega t)$, gdzie $A = 1.5$ a $\omega = 0.65$. Wartość współczynnika tłumienia ustawić jako $d = 0.5$.**

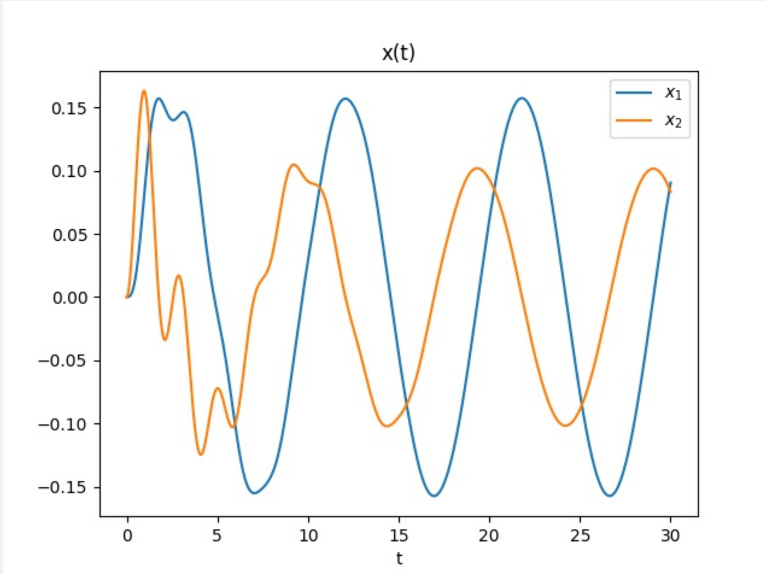
```
RESOLUTION = 1000
A = 1.5
J = 1
R = 1
omega = 0.65
d = 0.5
m = 1
g = 10

def tau_m(t:float, omega:float, A:float) -> float:
    return A*np.sin(omega*t)
```

```python

def model(
    x: np.ndarray,
    t: float,
    d: float,
    m: float,
    g: float,
    omega: float,
    A: float,
    J: float,
    R: float,
    tau_m: Callable[[float, float, float], float]
) -> np.ndarray:
    x1, x2 = x
    dx1 = x2
    dx2 = (tau_m(t, omega, A) - d*x2 - m*g*R*np.sin(x1))/J
    return np.array([dx1, dx2])
```

3. **Wyznaczyć i wykreślić rozwiązanie równania różniczkowego.**

```python
t = np.linspace(0, 30, RESOLUTION)

res = odeint(
    model,
    np.array([0, 0]),
    t,
    (d, m, g, omega, A, J, R, tau_m)
)

plt.plot(t, res)
plt.xlabel('t')
plt.title('x(t)')
plt.legend(['$x_1$', '$x_2$'])
plt.show()
plt.close()
```

Output:



Jaki jest charakter odpowiedzi układu na sygnał zadany równy funkcji trygonometrycznej?
Odpowiedź ma charakter liniowy oscylacyjny.