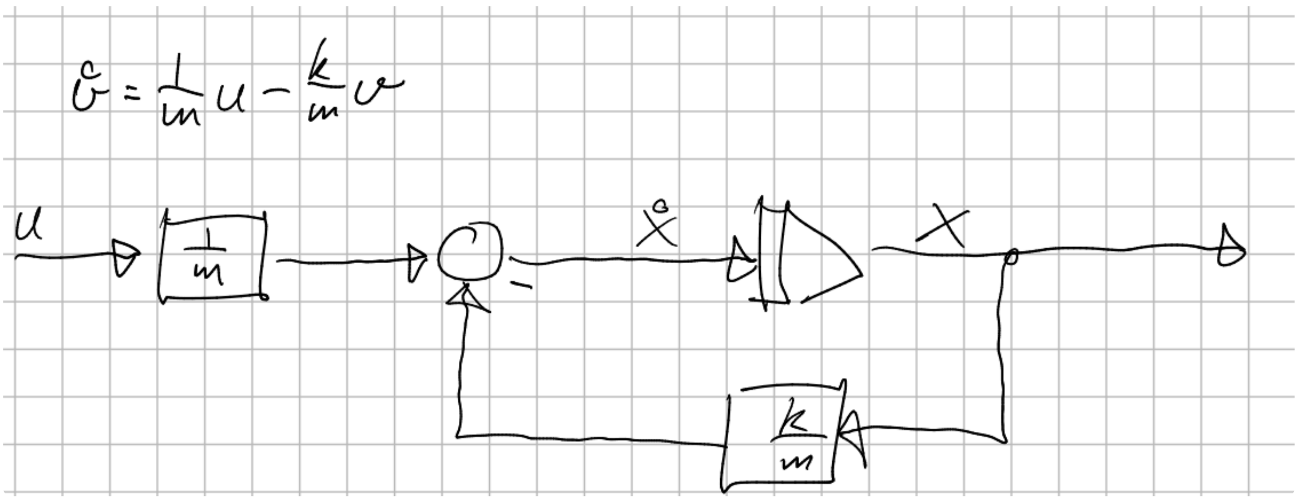
Kybintro øving 3 - Lars André Roda Jansen

Oppgave 1

a)



Ettersom at vi definerer u som konstant, så vil systemet være monovariabelt fordi den eneste variabelen er v. Det blir én tilbakekobling i systemet.

b)

Eulers metode er brukt for å kunne numerisk beregne / simulere en differensiallikning i tilfeller der en analytisk løsning på systemet ikke er lett tilgjengelig. Vi har en differensiallikning på formen:

$$\dot{x} = f(x)$$

Vi sier da at en eksakt løsning vil være på formen:

$$x(t)$$

For å numerisk beregne dette så bruker vi et tidsskritt h, slik at en t_n vil kunne defineres som tiden til det forrige steget addert med tidsskrittet:

$$t_{n+1} = t_n + h$$

VI sier da att resultatet av neste steg x_{n+1} vil da tilsvare:

$$x_{n+1} = x_n + hf(x_n)$$

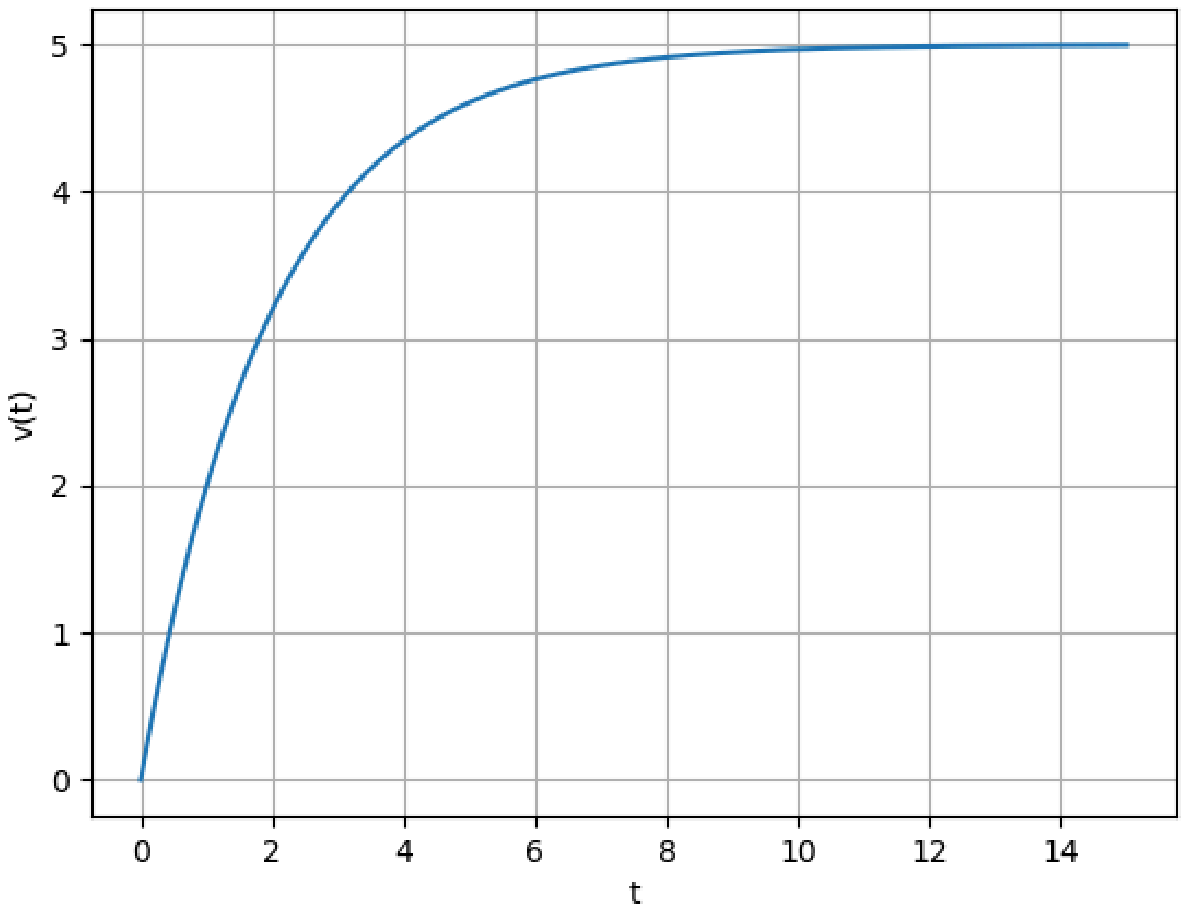
Om vi var til å simulere en differensiallikning ved et program så kunne man ha gjort dette med en for-løkke som beregner x_{n+1} ved å definere en funksjon for $f(x_n)$.

c)

Nøyaktigheten av løsningen ved bruk av Eulers metode blir bestemt av tidsskrittet h. Om h er for stor så vil resultatet bli veldig unøyaktig, men om h er for liten så kan det være kostbart i form av datakraft å beregne.

d)

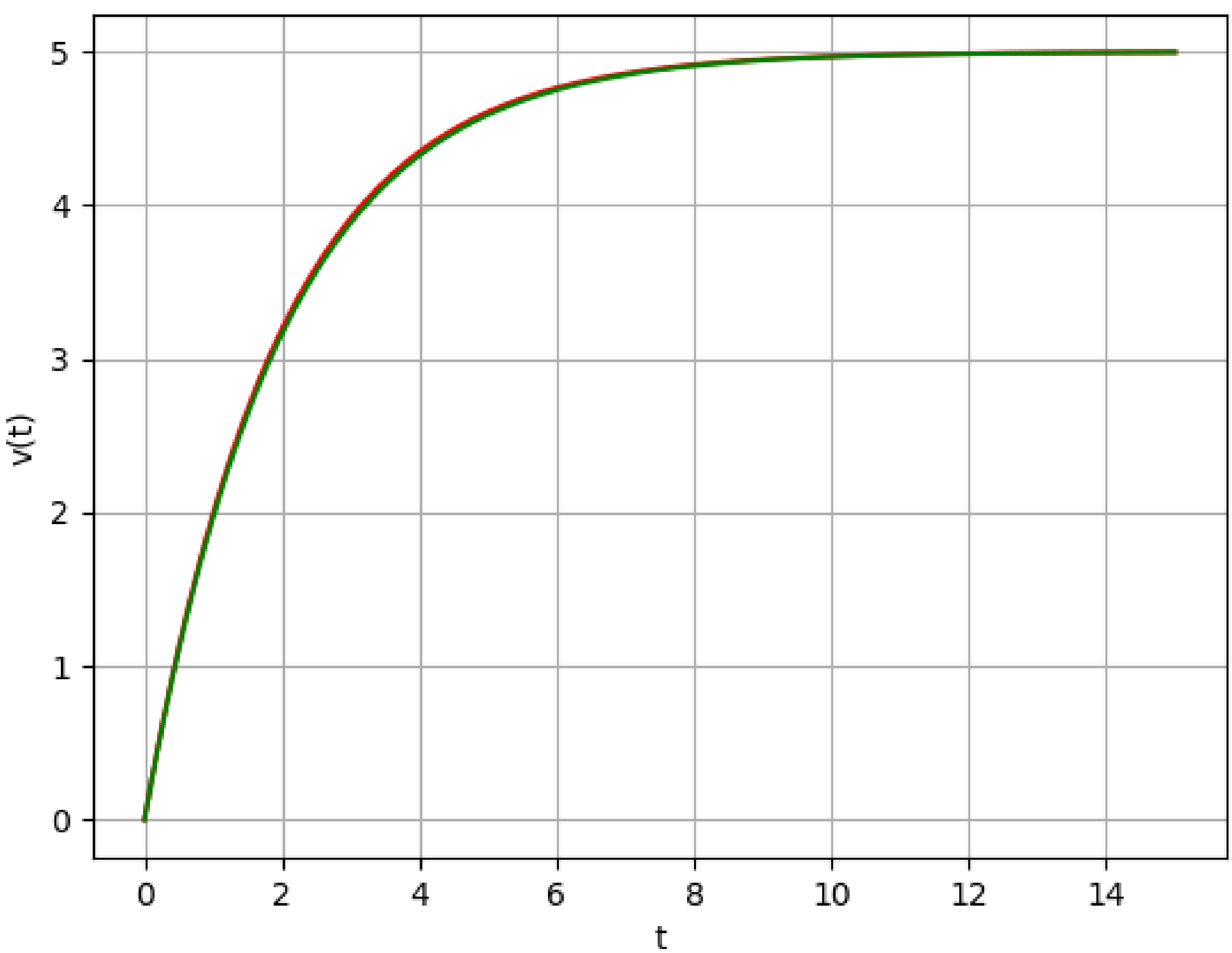
```
import numpy as np

import matplotlib.pyplot as plt



def f(v, u, m, k):
        return (1 / m) * u - (k / m) * v

def main()
        u = 500
        m = 200
        k = 100

        h = 0.1 #Tidsskritt
        t_start = 0.0 #Tidsintervall
        t_slutt = 15.0

        steps = int((t_slutt - t_start) / h)

        t = np.linspace(t_start, t_slutt, steps)

        print(t)

        v = np.zeros(steps)

        v[0] = 0



        for i in range(1, steps):
                v[i] = v[i - 1] + h * f(v[i - 1], u, m, k)


        print(v)

        plt.plot(t, v)
        plt.xlabel("t")
        plt.ylavel("x(t)")
        plt.grid()
        plt.show()


main()
```



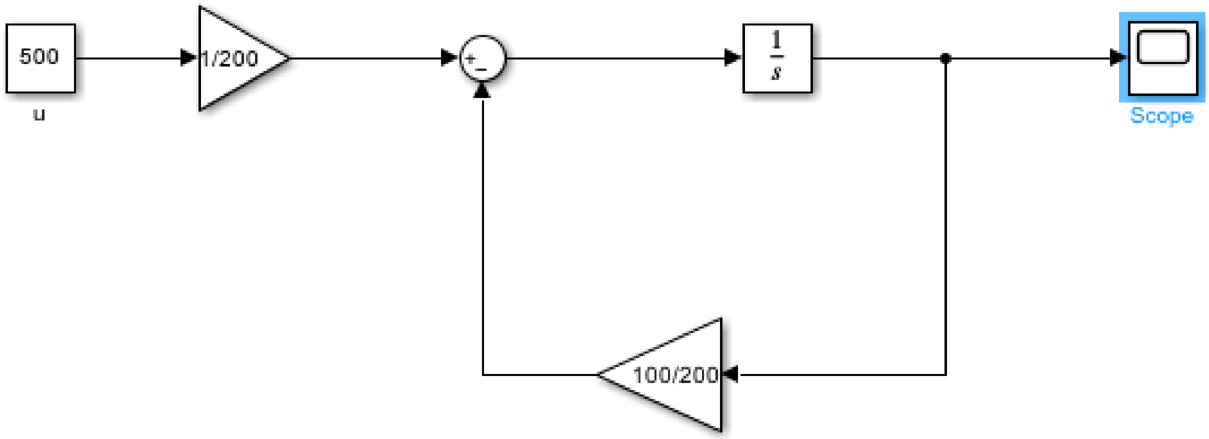
e)

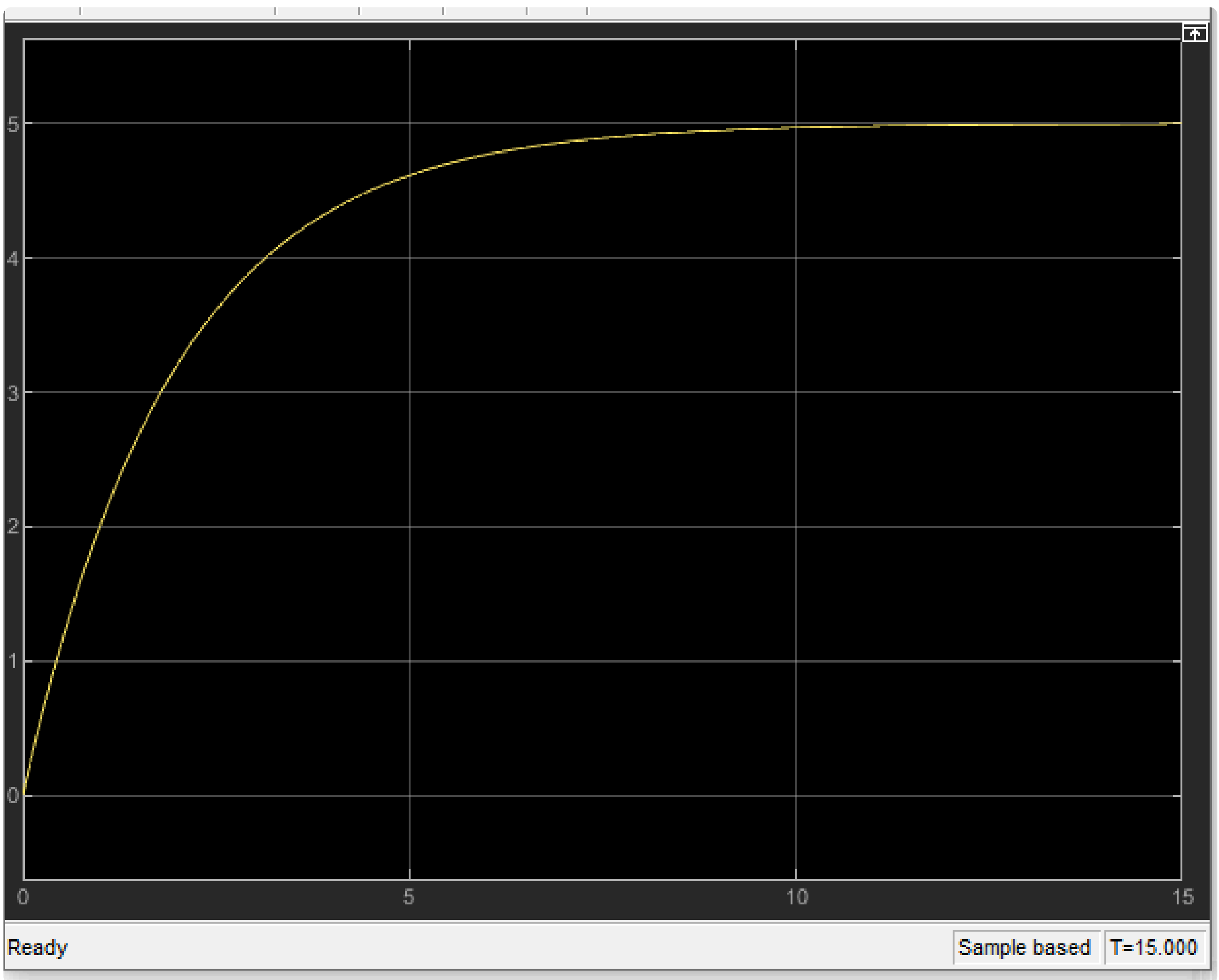
```
import numpy as np
import matplotlib.pyplot as plt



def f(v, u, m, k):
return (1 / m) * u - (k / m) * v


def v_t(v0, u, k, m, t):
return np.exp(- (k / m) * t) * (v0 - (u / k)) + (u / k)



def main():
        v0 = 0
        u = 500
        m = 200
        k = 100


        h = 0.1 #Tidsskritt
        t_start = 0.0 #Tidsintervall
        t_slutt = 15.0
```

```
        steps = int((t_slutt - t_start) / h)
        t = np.linspace(t_start, t_slutt, steps)

        print(t)

        v_euler = np.zeros(steps)
        v_analytical = np.zeros(steps)

        v_euler[0] = v0

        for i in range(1, steps):
                v_euler[i] = v_euler[i - 1] + h * f(v_euler[i - 1], u, m, k)


        for i in range(0, steps):
                v_analytical[i] = v_t(v0, u, k, m, t[i])

        plt.plot(t, v_euler, "r")
        plt.plot(t, v_analytical, "g")
        plt.xlabel("t")
        plt.ylabel("v(t)")
        plt.grid()
        plt.show()

main()
```



f)





Oppgave 2

a)

$$\sum M_t = J\ddot{\theta}$$

$$M_d = k_2\dot{\theta}$$

$$M_T = k_3\theta$$

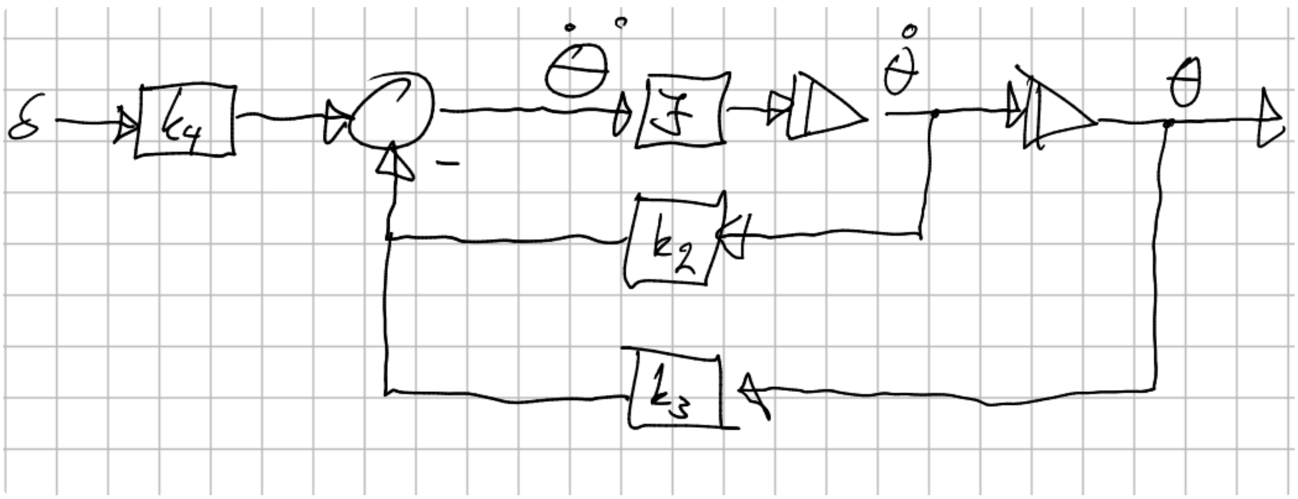
$$u = k_4\delta$$

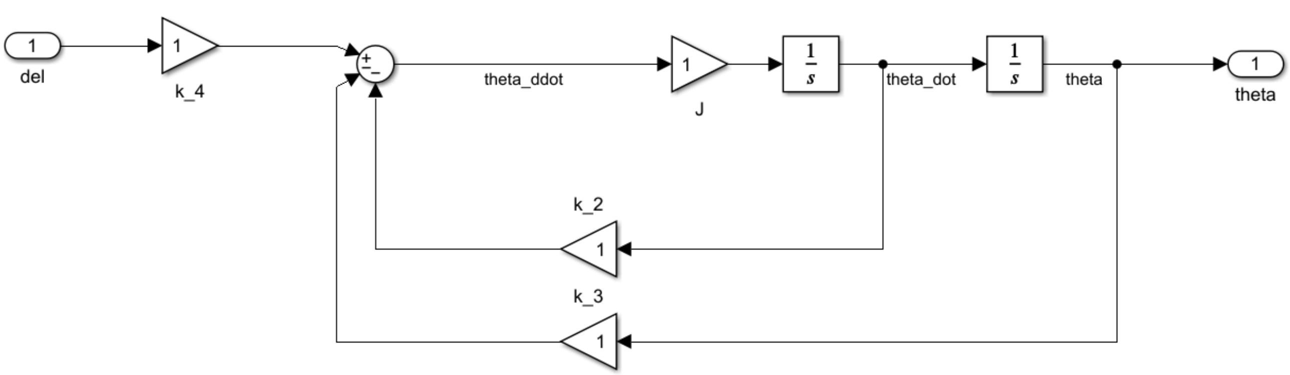
$$\sum M_t = -k_2\dot{\theta} - k_3\theta - k_4\delta$$

$$J\ddot{\theta} + k_2\dot{\theta} + k_3\theta = k_4\delta$$

Systemet er av andre orden.

b)





Oppgave 3

a)

$$x = \cos\theta * \int v\,dt$$

$$\dot{x} = -\sin\theta * v$$

b)

$$z = -\sin\theta * \int v\,dt$$

$$\dot{z} = -\cos\theta * v$$

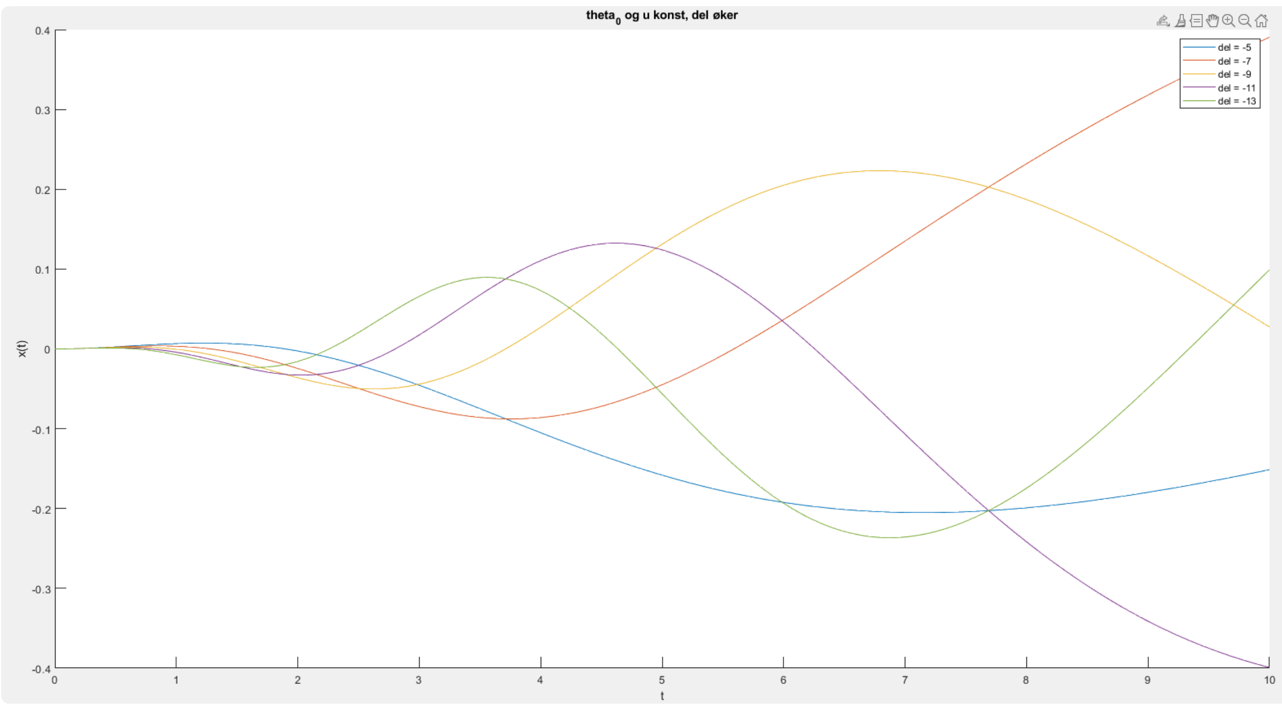
c)

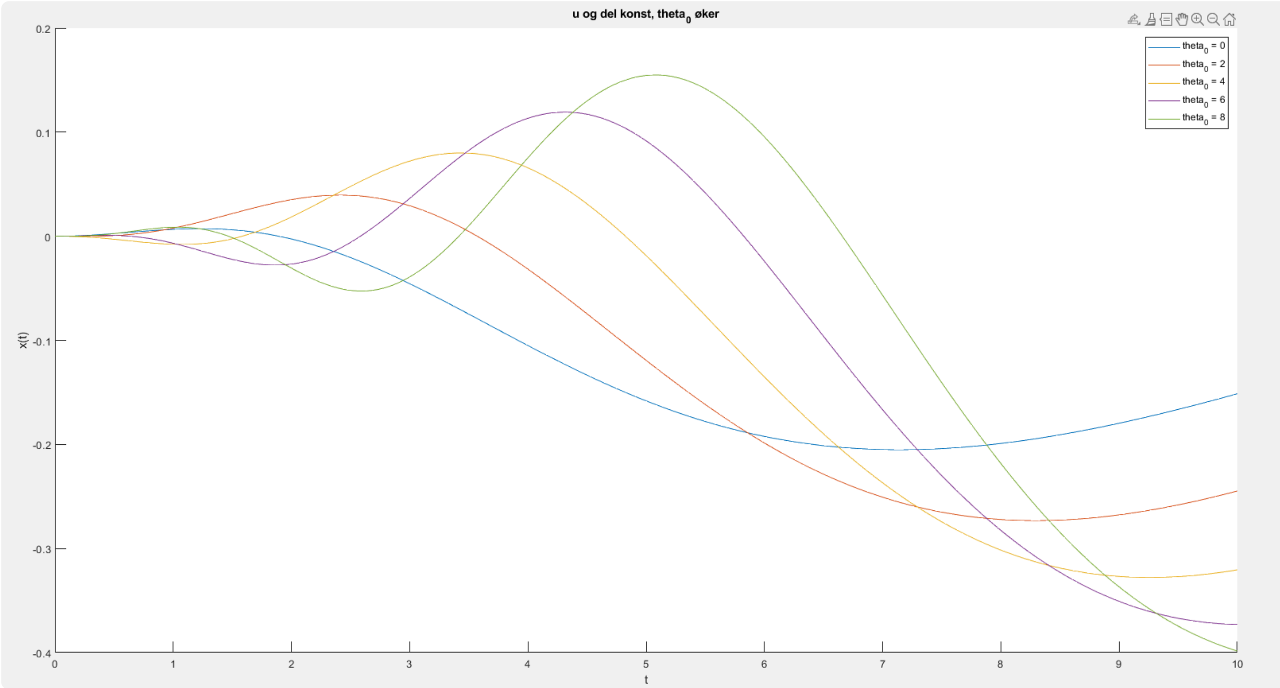
$$J\ddot{\theta} + k_2\dot{\theta} + k_3\theta = k_4\delta$$

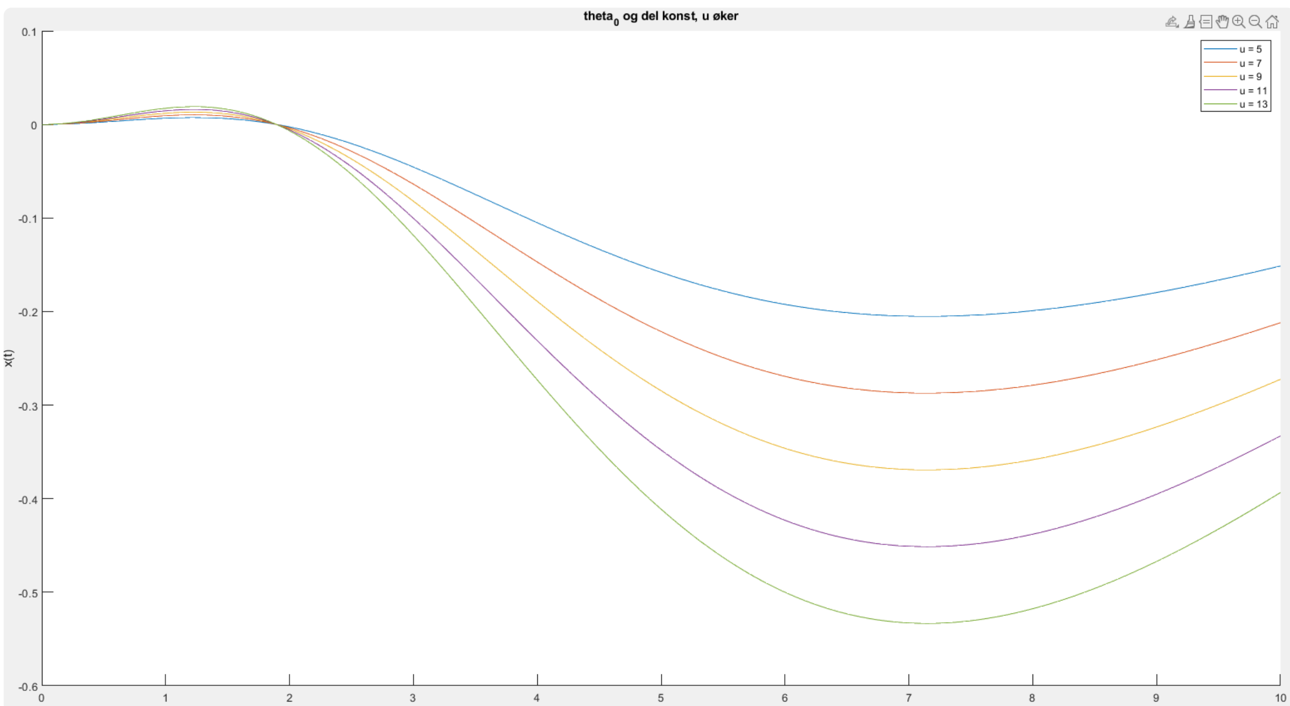
$$\begin{bmatrix} x \\ z \end{bmatrix} = \begin{bmatrix} \cos\theta * \int v\,dt \\ \\ -\sin\theta * \int v\,dt \end{bmatrix}$$

$$v = e^{-\frac{k}{m}t}(v_0 - \frac{u}{k}) + \frac{u}{k}$$

d)




u og del konst, theta$_0$ øker
theta$_0$ = 0
theta$_0$ = 2
theta$_0$ = 4
theta$_0$ = 6
theta$_0$ = 8
0.2
0.1
0
-0.1
-0.2
-0.3
-0.4
x(t)
0
1
2
3
4
5
6
7
8
9
10
t


theta$_0$ og del konst, u øker
u = 5
u = 7
u = 9
u = 11
u = 13
0.1
0
-0.1
-0.2
-0.3
-0.4
-0.5
-0.6
x(t)
0
1
2
3
4
5
6
7
8
9
10