CHEN 461 HW12

May 2, 2023

```
[ ]: from control import ss, tf, step_response, input_output_response
from matplotlib.pyplot import plot, xlabel, ylabel, title, legend, grid
from numpy import zeros, ones, linspace, array
from scipy.integrate import solve_ivp

from sympy import symbols, expand, simplify, exp, factor, latex
from sympy.abc import s, t, lamda, theta
from sympy.matrices import Matrix, eye
```

1 Problem 18.3

$A_1 \frac{dh_1}{dt} = F_{in,1} - \frac{h_1}{R_1}$

$A_2 \frac{dh_2}{dt} = F_{in,2} + \frac{h_1}{R_1} - \frac{h_2}{R_2}$

$F_{in,1} = k_{c,1}\left(h_{1,sp} - h_1\right)$

$F_{in,2} = k_{c,2}\left(h_{2,sp} - h_2\right)$

$A_1 \frac{dh_1}{dt} = k_{c,1}\left(h_{1,sp} - h_1\right) - \frac{h_1}{R_1}$

$A_2 \frac{dh_2}{dt} = k_{c,2}\left(h_{2,sp} - h_2\right) + \frac{h_1}{R_1} - \frac{h_2}{R_2}$

State space description:

$A_1 \frac{dh_1}{dt} = -\left(k_{c,1} + R_1^{-1}\right) h_1 + k_{c,1} h_{1,sp}$

$A_2 \frac{dh_2}{dt} = R_1^{-1} h_1 - \left(k_{c,2} + R_2^{-1}\right) h_2 + k_{c,2} h_{2,sp}$

Taking Laplace Transforms

$H_1 = \frac{k_{c,1}}{A_1 s + k_{c,1} + R_1^{-1}} H_{1,sp}$

$H_2 = \frac{k_{c,2}}{A_2 s + k_{c,2} + R_2^{-1}} H_{2,sp} + \frac{R_1^{-1} H_1}{A_2 s + k_{c,2} + R_2^{-1}}$

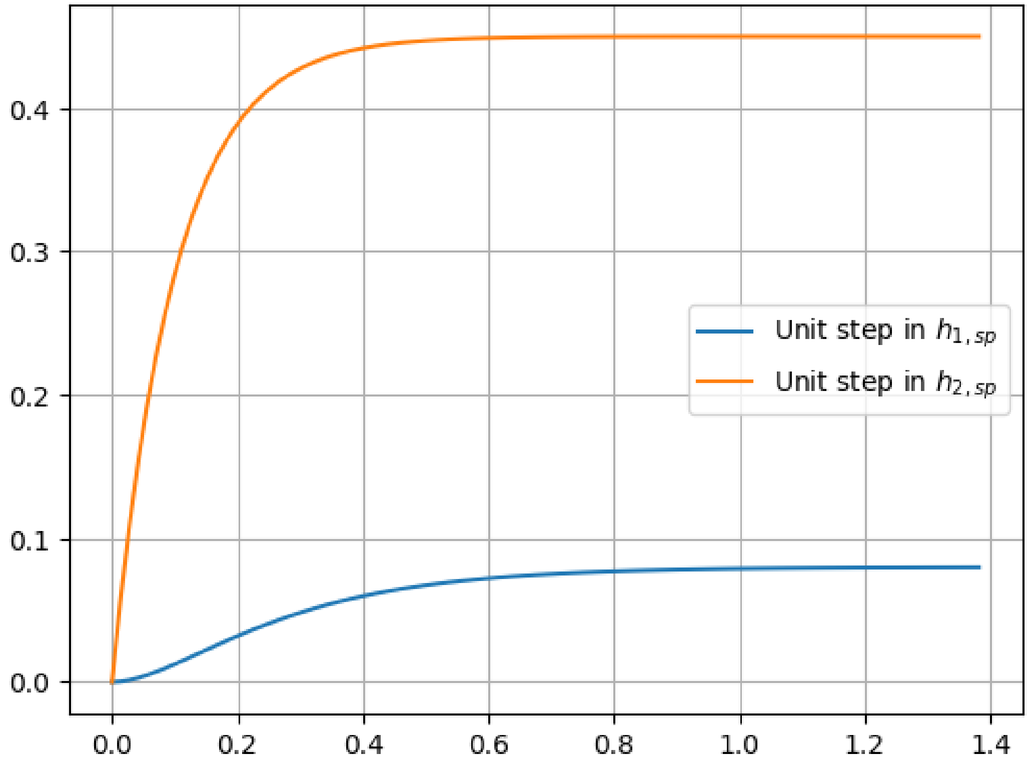
$H_2 = \frac{k_{c,2}}{A_2 s + k_{c,2} + R_2^{-1}} H_{2,sp} + \frac{R_1^{-1} k_{c,1}}{(A_2 s + k_{c,2} + R_2^{-1})(A_1 s + k_{c,1} + R_1^{-1})} H_{1,sp}$

$H_2 = \frac{k_{c,2} R_2}{A_2 R_2 s + R_2 k_{c,2} + 1} H_{2,sp} + \frac{R_2 k_{c,1}}{(A_2 R_2 s + R_2 k_{c,2} + 1)(A_1 R_1 s + R_1 k_{c,1} + 1)} H_{1,sp}$

```
[ ]: h_1, h_2, A_1, A_2, R_1, R_2, h_1sp, h_2sp, k_c1, k_c2 = symbols("h_1, h_2,␣
↪A_1, A_2, R_1, R_2, h_1sp, h_2sp, k_c1, k_c2")
```

```python
F_in1 = k_c1 * (h_1sp - h_1)
F_in2 = k_c2 * (h_2sp - h_2)

f_1 = expand((F_in1 - h_1 / R_1) / A_1)
f_2 = expand((F_in2 + h_1 / R_1 - h_2 / R_2) / A_2)

A_sym = Matrix([
    [f_1.coeff(h_1), f_1.coeff(h_2)],
    [f_2.coeff(h_1), f_2.coeff(h_2)],
])

B_sym = Matrix([
    [f_1.coeff(h_1sp), f_1.coeff(h_2sp)],
    [f_2.coeff(h_1sp), f_2.coeff(h_2sp)],
])

c_sym = Matrix([[0, 1 / R_2]])

d = 0

sub_dict = {A_1: 1, A_2: 0.5, R_1: 1, R_2: 2, k_c1: 4, k_c2: 4.5}

A = array(A_sym.subs(sub_dict), dtype=float)
B = array(B_sym.subs(sub_dict), dtype=float)
c = array(c_sym.subs(sub_dict), dtype=float)

sys = ss(A, B, c, d)
t, y = step_response(sys)

plot(t, y[0, 0], label=r"Unit step in $h_{1,sp}$")
plot(t, y[0, 1], label=r"Unit step in $h_{2,sp}$")
grid()
legend(loc="right")
```

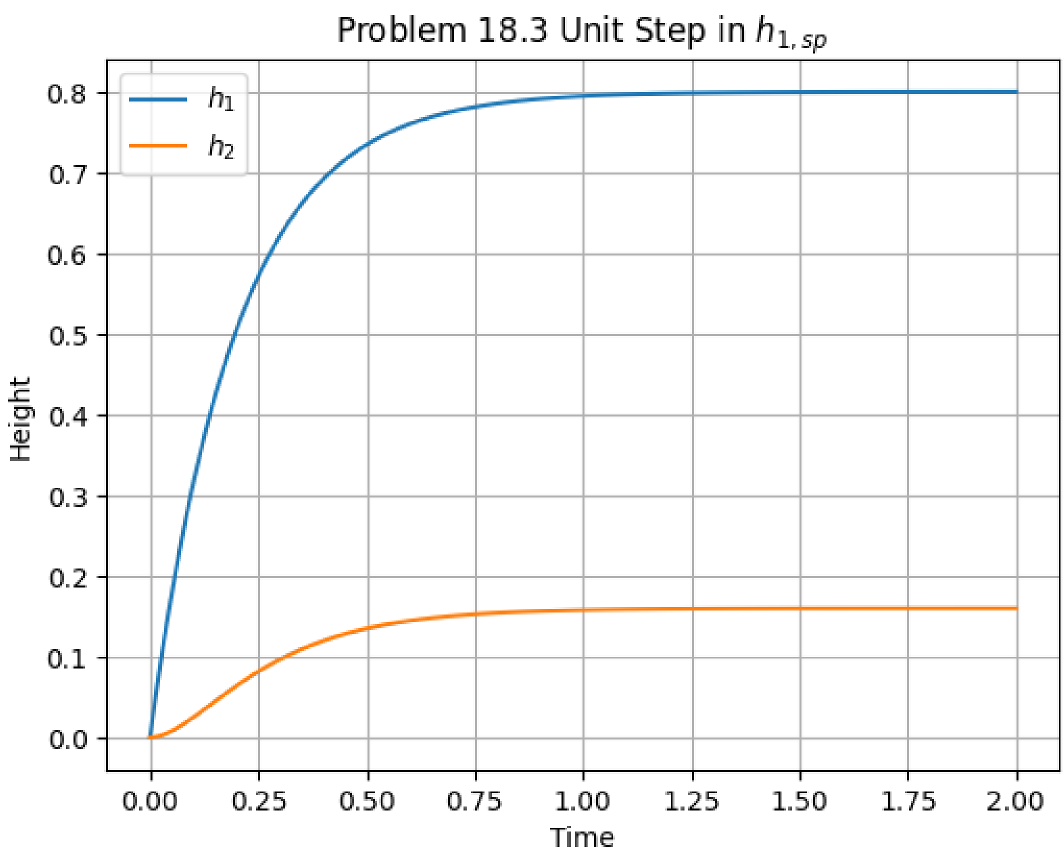
[ ]: <matplotlib.legend.Legend at 0x140d6bf0a50>



[ ]:
```
def p1_ode(t, y, h_1sp, h_2sp):
    f = y * 0
    h_1 = y[0]
    h_2 = y[1]

    A_1 = 1
    A_2 = 0.5
    R_1 = 1
    R_2 = 2
    k_c1 = 4
    k_c2 = 4.5

    f[0] = (k_c1 * (h_1sp - h_1) - h_1 / R_1) / A_1
    f[1] = (k_c2 * (h_2sp - h_2) + h_1 / R_1 - h_2 / R_2) / A_2

    return f

ode_args = (p1_ode, [0, 2], [0, 0])

ode_kwargs = {
    'method': "Radau",
    'atol': 1e-8,
```

```
    'rtol': 1e-8,
}

p1_sol_1 = solve_ivp(*ode_args, **ode_kwargs, args=(1, 0))

plot(p1_sol_1.t, p1_sol_1.y[0], label=r"$h_1$")
plot(p1_sol_1.t, p1_sol_1.y[1], label=r"$h_2$")
grid()
xlabel("Time")
ylabel("Height")
title(r"Problem 18.3 Unit Step in $h_{1,sp}$")
legend()
```

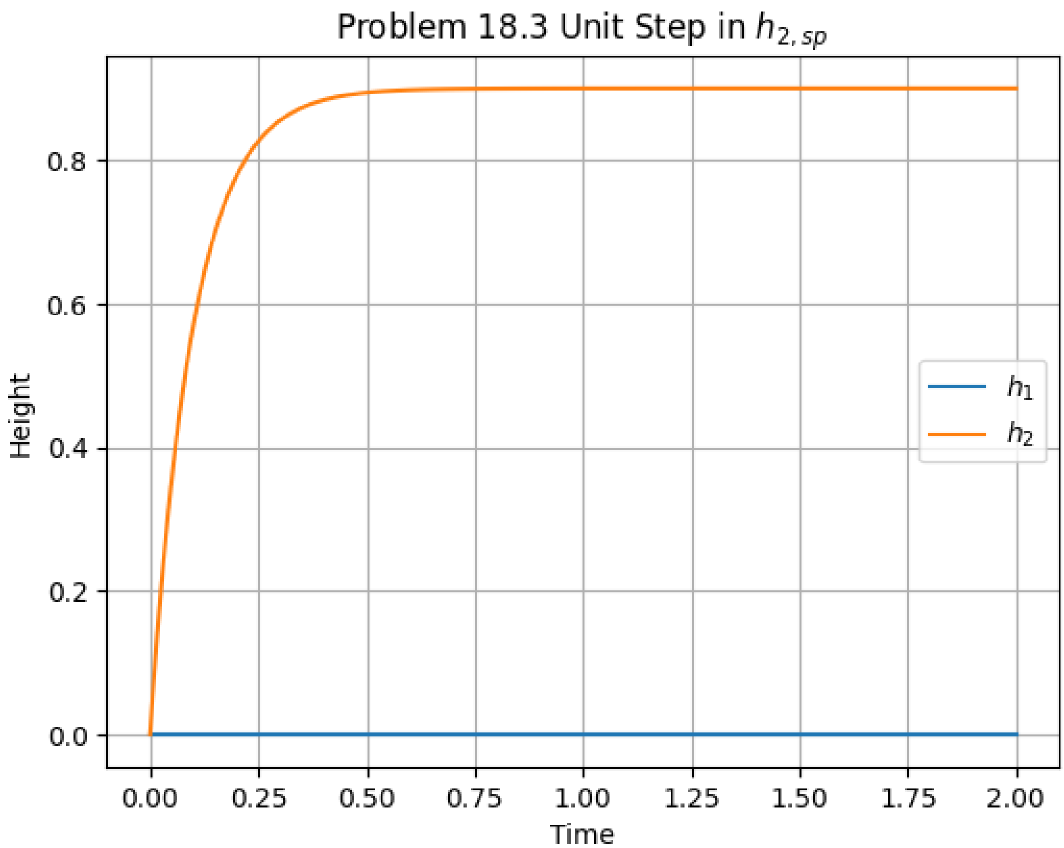
[ ]: <matplotlib.legend.Legend at 0x140daedfad0>



[ ]:
```
p1_sol_2 = solve_ivp(*ode_args, **ode_kwargs, args=(0, 1))

plot(p1_sol_2.t, p1_sol_2.y[0], label=r"$h_1$")
plot(p1_sol_2.t, p1_sol_2.y[1], label=r"$h_2$")
```

```
grid()
xlabel("Time")
ylabel("Height")
title(r"Problem 18.3 Unit Step in $h_{2,sp}$")
legend()
```

[ ]: <matplotlib.legend.Legend at 0x140d3daa690>



The set point of h_1 affects h_2 because tank 2 is downstream of tank 1. The set point of h_2 does not affect h_1 because the set point of h_2 does not affect the inlet flow rate.

2 Problem 19.1

2.1 Part A

[ ]:
```
k, tau_0, tau_1, tau_2, tau_3 = symbols("k, tau_0, tau_1, tau_2, tau_3")

Gpp = 1
Gpm = k * (1 + tau_0 * s) / (1 + tau_1 * s) / (1 + tau_2 * s) / (1 + tau_3 * s)
r = 2
```

```
G_c = factor(simplify(1 / ((lamda * s + 1)**r - Gpp) / Gpm))

G_c
```

[ ]: $$\frac{(s\tau_1 + 1)(s\tau_2 + 1)(s\tau_3 + 1)}{k\lambda s(\lambda s + 2)(s\tau_0 + 1)}$$

$G_c = \frac{(s\tau_1+1)(s\tau_2+1)(s\tau_3+1)}{k\lambda s(\lambda s+2)(s\tau_0+1)}$

Not a PID. There are three zeros and three poles.

2.2 Part B

[ ]:
```
Gpp = (1 - tau_0 * s) / (1 + tau_0 * s)

G_c = factor(simplify(1 / ((lamda * s + 1)**r - Gpp) / Gpm))

G_c
```

[ ]: $$\frac{(s\tau_1 + 1)(s\tau_2 + 1)(s\tau_3 + 1)}{ks(\lambda^2 s^2 \tau_0 + \lambda^2 s + 2\lambda s\tau_0 + 2\lambda + 2\tau_0)}$$

$G_c = \frac{(s\tau_1+1)(s\tau_2+1)(s\tau_3+1)}{ks(\lambda^2 s^2\tau_0+\lambda^2 s+2\lambda s\tau_0+2\lambda+2\tau_0)}$

Not a PID. There are three zeros.

2.3 Part C

[ ]:
```
Gpp = exp(-theta * s)
Gpm = k / (1 + tau_1 * s) / (1 + tau_2 * s)
r = 2
G_c = factor(simplify(1 / ((lamda * s + 1)**r - Gpp) / Gpm))

G_c
```

[ ]: $$\frac{(s\tau_1 + 1)(s\tau_2 + 1)e^{s\theta}}{k(\lambda^2 s^2 e^{s\theta} + 2\lambda s e^{s\theta} + e^{s\theta} - 1)}$$

$G_c = \frac{(s\tau_1+1)(s\tau_2+1)e^{s\theta}}{k(\lambda^2 s^2 e^{s\theta}+2\lambda s e^{s\theta}+e^{s\theta}-1)}$

Not a PID. There is an exponential.

2.3.1 Pade

[ ]:
```
pade_1 = (1 - theta * s / 2) / (1 + theta * s / 2)
Gpp = pade_1
Gpm = k / (1 + tau_1 * s) / (1 + tau_2 * s) * pade_1
r = 2
G_c = factor(simplify(1 / ((lamda * s + 1)**r - Gpp) / Gpm))

G_c
```

[ ]:

$$-\frac{\left(s\tau_1+1\right)\left(s\tau_2+1\right)\left(s\theta+2\right)^2}{ks\left(s\theta-2\right)\left(\lambda^2s^2\theta+2\lambda^2s+2\lambda s\theta+4\lambda+2\theta\right)}$$

$G_c = -\frac{(s\tau_1+1)(s\tau_2+1)(s\theta+2)^2}{ks(s\theta-2)(\lambda^2s^2\theta+2\lambda^2s+2\lambda s\theta+4\lambda+2\theta)}$

Not a PID. There are four zeros.