Identification and Control of a Quadruple Tank system

Lecture 2 – Linearization and control design

Lorenzo Nigro, Prof. Riccardo Scattolini

Slides and control schemes by Fabio Bonassi

Useful information

POLITECNICO MILANO 1863

Responsible of tutoring activities: Lorenzo Nigro (`lorenzo.nigro@polimi.it`)

Tentative Schedule:

Lecture 1. Modeling, identification, and validation	April 10, 16:15 – 18:15	Online (Webex)
Lecture 2. Design of a PI-based control architecture	April 12, 11:15 – 13:15	Online (Webex)
Lecture 3. Testing of students' LQ control architectures (3 groups)	April 24, 10:30 – 12:30	Lab (IRL)
Lecture 4. Testing of students' LQ control architectures (4 groups)	April 24, 16:00 – 19:00	Lab (IRL)
Lecture 5. Testing of students' LQ control architectures (4 groups)	April 30, 9:30 – 12:30	Lab (IRL)
Lecture 6. Testing of students' LQ control architectures (3 groups)	May 8, 10:30 – 12:30	Lab (IRL)
Lecture 7. Testing of students' LQ control architectures (4 groups)	May 8, 16:00 – 19:00	Lab (IRL)
Lecture 8. Testing of students' LQ control architectures (4 groups)	May 15, 16:00 – 19:00	Lab (IRL)

Please remember to fill in the attendance form!

Register your participation

POLITECNICO
MILANO 1863

tutorapp.polimi.it

10537535

Goals of the second lecture

POLITECNICO MILANO 1863

- Linearize the grey-box model of the Quadruple Tank
- Design a decentralized control system
- Design a decoupled control system

System linearization

Nonlinear model under consideration

$$\dot{h}_1 = -\frac{A_1}{S}\sqrt{2gh_1} + \frac{\kappa_2(V_2)\cdot\gamma_2(V_2)}{S}V_2$$

$$\dot{h}_2 = \frac{A_1}{S}\sqrt{2gh_1} - \frac{A_2}{S}\sqrt{2gh_2} + \frac{\kappa_1(V_1)\cdot(1-\gamma_1(V_1))}{S}V_1$$

$$\dot{h}_3 = -\frac{A_3}{S}\sqrt{2gh_3} + \frac{\kappa_1(V_1)\cdot\gamma_1(V_1)}{S}V_1$$

$$\dot{h}_4 = \frac{A_3}{S}\sqrt{2gh_3} - \frac{A_4}{S}\sqrt{2gh_4} + \frac{\kappa_2(V_2)\cdot(1-\gamma_2(V_2))}{S}V_2$$

System linearization

Nonlinear model under consideration

$$\dot{h}_1 = -\frac{\boxed{A_1}}{S}\sqrt{2gh_1} + \frac{\boxed{\kappa_2(V_2)}\cdot\boxed{\gamma_2(V_2)}}{S}V_2$$

$$\dot{h}_2 = \frac{\boxed{A_1}}{S}\sqrt{2gh_1} - \frac{\boxed{A_2}}{S}\sqrt{2gh_2} + \frac{\boxed{\kappa_1(V_1)}\cdot(1-\boxed{\gamma_1(V_1)})}{S}V_1$$

$$\dot{h}_3 = -\frac{\boxed{A_3}}{S}\sqrt{2gh_3} + \frac{\boxed{\kappa_1(V_1)}\cdot\boxed{\gamma_1(V_1)}}{S}V_1$$

$$\dot{h}_4 = \frac{\boxed{A_3}}{S}\sqrt{2gh_3} - \frac{\boxed{A_4}}{S}\sqrt{2gh_4} + \frac{\boxed{\kappa_2(V_2)}\cdot(1-\boxed{\gamma_2(V_2)})}{S}V_2$$

Estimated (nonlinear) map

Identified parameter

System linearization

Nonlinear model under consideration

$$\dot{h}_1 = -\frac{\boxed{A_1}}{S}\sqrt{2gh_1} + \frac{\boxed{\kappa_2(V_2)} \cdot \boxed{\gamma_2(V_2)}}{S} V_2$$

$$\dot{h}_2 = \frac{\boxed{A_1}}{S}\sqrt{2gh_1} - \frac{\boxed{A_2}}{S}\sqrt{2gh_2} + \frac{\boxed{\kappa_1(V_1)} \cdot (1 - \boxed{\gamma_1(V_1)})}{S} V_1$$

$$\dot{h}_3 = -\frac{\boxed{A_3}}{S}\sqrt{2gh_3} + \frac{\boxed{\kappa_1(V_1)} \cdot \boxed{\gamma_1(V_1)}}{S} V_1$$

$$\dot{h}_4 = \frac{\boxed{A_3}}{S}\sqrt{2gh_3} - \frac{\boxed{A_4}}{S}\sqrt{2gh_4} + \frac{\boxed{\kappa_2(V_2)} \cdot (1 - \boxed{\gamma_2(V_2)})}{S} V_2$$

Estimated (nonlinear) map

Identified parameter

In the following, for simplicity we will denote the grey-box model by

$$\dot{x}(t) = f(x(t), u(t))$$
$$y(t) = C\,x(t)$$

$$x = \begin{bmatrix} h_1 \\ h_2 \\ h_3 \\ h_4 \end{bmatrix} \qquad u = \begin{bmatrix} V_1 \\ V_2 \end{bmatrix} \qquad y = \begin{bmatrix} h_2 \\ h_4 \end{bmatrix}$$

System linearization

Nonlinear model under consideration

$$\dot{h}_1 = -\frac{\boxed{A_1}}{S}\sqrt{2gh_1} + \frac{\boxed{\kappa_2(V_2)} \cdot \boxed{\gamma_2(V_2)}}{S} V_2$$

$$\dot{h}_2 = \frac{\boxed{A_1}}{S}\sqrt{2gh_1} - \frac{\boxed{A_2}}{S}\sqrt{2gh_2} + \frac{\boxed{\kappa_1(V_1)} \cdot (1 - \boxed{\gamma_1(V_1)})}{S} V_1$$

$$\dot{h}_3 = -\frac{\boxed{A_3}}{S}\sqrt{2gh_3} + \frac{\boxed{\kappa_1(V_1)} \cdot \boxed{\gamma_1(V_1)}}{S} V_1$$

$$\dot{h}_4 = \frac{\boxed{A_3}}{S}\sqrt{2gh_3} - \frac{\boxed{A_4}}{S}\sqrt{2gh_4} + \frac{\boxed{\kappa_2(V_2)} \cdot (1 - \boxed{\gamma_2(V_2)})}{S} V_2$$

Estimated (nonlinear) map

Identified parameter

In the following, for simplicity we will denote the grey-box model by

$$\dot{x}(t) = f(x(t), u(t))$$
$$y(t) = C\, x(t)$$

$$x = \begin{bmatrix} h_1 \\ h_2 \\ h_3 \\ h_4 \end{bmatrix} \qquad u = \begin{bmatrix} V_1 \\ V_2 \end{bmatrix} \qquad y = \begin{bmatrix} h_2 \\ h_4 \end{bmatrix}$$

Problem: How do we compute the equilibrium $(\bar{x}, \bar{u}, \bar{y})$ corresponding to some $\bar{y}$?

Equilibrium computation

POLITECNICO MILANO 1863

We want to solve numerically the following system of equations $\begin{cases} 0 = f(\bar{x}, \bar{u}) \\ \bar{y} = C\bar{x} \end{cases}$

Letting $z = \begin{bmatrix} \bar{x} \\ \bar{u} \end{bmatrix}$, computing the equilibrium amounts to finding $\underline{z} \leq z \leq \bar{z}$ such that $g(z) = 0$

Equilibrium computation

POLITECNICO MILANO 1863

We want to solve numerically the following system of equations $\begin{cases} 0 = f(\bar{x}, \bar{u}) \\ \bar{y} = C\bar{x} \end{cases}$

Letting $z = \begin{bmatrix} \bar{x} \\ \bar{u} \end{bmatrix}$, computing the equilibrium amounts to finding $\underline{z} \leq z \leq \bar{z}$ such that $g(z) = 0$

$$g(z) = [\ f(\bar{z});\ \tilde{C}\bar{z} - \bar{y}]$$

Possible solutions:

- **fsolve** – does not support bounds on z
- **Symbolic Toolbox** – does not support numeric maps used for $\kappa_j(V_j)$ and $\gamma_j(V_j)$
- **CasADi** – does not support numeric maps used for $\kappa_j(V_j)$ and $\gamma_j(V_j)$
- **<u>fmincon</u> –** supports bounds and numeric maps

Equilibrium computation – fmincon

POLITECNICO MILANO 1863

```
nlconstr = @(z) build_nonlinear_constraints(z, y_bar);  % Nonlinear constraints
z_lb = [ eps; eps; eps; eps; 0; 0 ]; % Lower bound on decision variables
z_ub = [ 30; 30; 30; 30; 18; 18 ]; % Upper bound on decision variables
z0 = [ 6; 6; 6; 6; 6; 6 ]; % Initial guess
```

Lower and Upper decision variable bounds

```
z_sol = fmincon(@(z) z(1)+z(3), z0, [], [], [], [], z_lb, z_ub, nlconstr);
```

```
function [ nlcnstr_le, nlcnstr_eq ] = build_nonlinear_constraints(z, y_bar)
        nlcnstr_le = [];

        nlcnstr_eq = [ dynamics_f(z(1:4), z(5:6));
                       y_bar - dynamics_g(z(1:4)) ];
end

function x_dot = dynamics_f(x, u)
        x_dot(1) = - A1 / S * sqrt(2 * g * x(1)) + k2(u(2)) * gamma2(u(2)) / S * u(2);
        x_dot(2) = A1 / S * sqrt(2 * g * x(1)) – A2 / S * sqrt(2 * g * x(2)) + k1(u(1)) * (1 - gamma1(u(1))) / S * u(1);
        x_dot(3) = - A3 / S * sqrt(2 * g * x(3)) + k1(u(1)) * gamma1(u(1)) / S * u(1);
        x_dot(4) = A3 / S * sqrt(2 * g * x(3)) – A4 / S * sqrt(2 * g * x(4)) + k2(u(2)) * (1 - gamma2(u(2))) / S * u(2);
end

function y = dynamics_g(x)
        y = [ x(2); x(4) ];
end
```

Equilibrium computation – fmincon

```
nlconstr = @(z) build_nonlinear_constraints(z, y_bar);  % Nonlinear constraints
z_lb = [ eps; eps; eps; eps; 0; 0 ]; % Lower bound on decision variables
z_ub = [ 30; 30; 30; 30; 18; 18 ]; % Upper bound on decision variables
z0 = [ 6; 6; 6; 6; 6; 6 ]; % Initial guess
```

Lower and Upper decision variable bounds

Random initial guess

```
z_sol = fmincon(@(z) z(1)+z(3), z0, [], [], [], [], z_lb, z_ub, nlconstr);
```

```
function [ nlcnstr_le, nlcnstr_eq ] = build_nonlinear_constraints(z, y_bar)
        nlcnstr_le = [];

        nlcnstr_eq = [ dynamics_f(z(1:4), z(5:6));
                y_bar - dynamics_g(z(1:4)) ];
end

function x_dot = dynamics_f(x, u)
        x_dot(1) = - A1 / S * sqrt(2 * g * x(1)) + k2(u(2)) * gamma2(u(2)) / S * u(2);
        x_dot(2) = A1 / S * sqrt(2 * g * x(1)) – A2 / S * sqrt(2 * g * x(2)) + k1(u(1)) * (1 - gamma1(u(1))) / S * u(1);
        x_dot(3) = - A3 / S * sqrt(2 * g * x(3)) + k1(u(1)) * gamma1(u(1)) / S * u(1);
        x_dot(4) = A3 / S * sqrt(2 * g * x(3)) – A4 / S * sqrt(2 * g * x(4)) + k2(u(2)) * (1 - gamma2(u(2))) / S * u(2);
end

function y = dynamics_g(x)
        y = [ x(2); x(4) ];
end
```

Equilibrium computation – fmincon

POLITECNICO MILANO 1863

```
nlconstr = @(z) build_nonlinear_constraints(z, y_bar);  % Nonlinear constraints
z_lb = [ eps; eps; eps; eps; 0; 0 ]; % Lower bound on decision variables
z_ub = [ 30; 30; 30; 30; 18; 18 ]; % Upper bound on decision variables
z0 = [ 6; 6; 6; 6; 6; 6 ]; % Initial guess

z_sol = fmincon(@(z) z(1)+z(3), z0, [], [], [], [], z_lb, z_ub, nlconstr);
```

Lower and Upper decision variable bounds

Random initial guess

Inequalities, equalities

```
function [ nlcnstr_le, nlcnstr_eq ] = build_nonlinear_constraints(z, y_bar)
        nlcnstr_le = [];

        nlcnstr_eq = [ dynamics_f(z(1:4), z(5:6));
                y_bar - dynamics_g(z(1:4)) ];
end

function x_dot = dynamics_f(x, u)
        x_dot(1) = - A1 / S * sqrt(2 * g * x(1)) + k2(u(2)) * gamma2(u(2)) / S * u(2);
        x_dot(2) = A1 / S * sqrt(2 * g * x(1)) – A2 / S * sqrt(2 * g * x(2)) + k1(u(1)) * (1 - gamma1(u(1))) / S * u(1);
        x_dot(3) = - A3 / S * sqrt(2 * g * x(3)) + k1(u(1)) * gamma1(u(1)) / S * u(1);
        x_dot(4) = A3 / S * sqrt(2 * g * x(3)) – A4 / S * sqrt(2 * g * x(4)) + k2(u(2)) * (1 - gamma2(u(2))) / S * u(2);
end


function y = dynamics_g(x)
        y = [ x(2); x(4) ];
end
```

$$g(z) = [\, f(\bar{z}); \\ \tilde{C}\bar{z} - \bar{y}]$$

Nonlinear constraints

Equilibrium computation – fmincon

$$\bar{y} = \begin{bmatrix} 10 \\ 10 \end{bmatrix} \rightarrow \quad \bar{x} = \begin{bmatrix} 24.66 \\ 10 \\ 3.04 \\ 10 \end{bmatrix}, \quad \bar{u} = \begin{bmatrix} 6.60 \\ 9.28 \end{bmatrix}$$

System linearization – cont'd

Two approaches:

1. Time-based linearization block using the Simulink model

System linearization – cont'd

POLITECNICO MILANO 1863

Two approaches:

1. Time-based linearization block using the Simulink model
2. **By hand**

$$\dot{\delta x} = \left.\frac{\partial f(x,u)}{\partial x}\right|_{\bar{x},\bar{u}} \delta x + \left.\frac{\partial f(x,u)}{\partial u}\right|_{\bar{x},\bar{u}} \delta u$$

System linearization – cont'd

POLITECNICO MILANO 1863

Two approaches:

1. Time-based linearization block using the Simulink model
2. **By hand**

$$\dot{\delta x} = \left.\frac{\partial f(x,u)}{\partial x}\right|_{\bar{x},\bar{u}} \delta x + \left.\frac{\partial f(x,u)}{\partial u}\right|_{\bar{x},\bar{u}} \delta u$$

$$\delta\dot{h}_1 = -\frac{A_1\sqrt{2g}}{S\sqrt{\bar{h}_1}}\delta h_1 + \left.\frac{\partial[k_2(V_2)\gamma_2(V_2)\,V_2]}{\partial V_2}\right|_{\bar{V}_2} \delta V_2$$

$$\delta\dot{h}_2 = \frac{A_1\sqrt{2g}}{S\sqrt{\bar{h}_1}}\delta h_1 - \frac{A_2\sqrt{2g}}{S\sqrt{\bar{h}_2}}\delta h_2 + \left.\frac{\partial\left[k_1(V_1)\left(1-\gamma_1(V_1)\right)V_1\right]}{\partial V_1}\right|_{\bar{V}_1} \delta V_1$$

$$\delta\dot{h}_3 = -\frac{A_3\sqrt{2g}}{S\sqrt{\bar{h}_3}}\delta h_3 + \left.\frac{\partial[k_1(V_1)\gamma_1(V_1)\,V_1]}{\partial V_1}\right|_{\bar{V}_1} \delta V_1$$

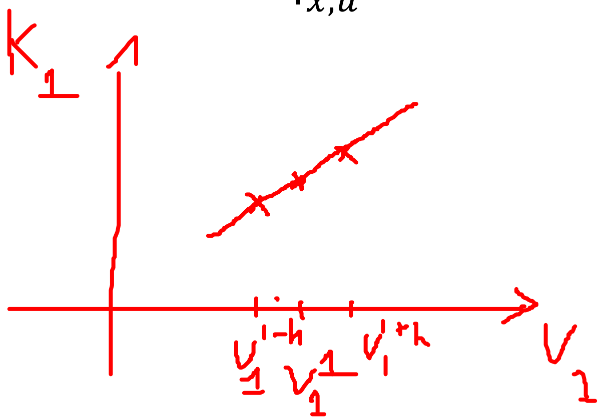
$$\delta\dot{h}_4 = \frac{A_3\sqrt{2g}}{S\sqrt{\bar{h}_3}}\delta h_3 - \frac{A_4\sqrt{2g}}{S\sqrt{\bar{h}_4}}\delta h_4 + \left.\frac{\partial\left[k_2(V_2)\left(1-\gamma_2(V_2)\right)V_2\right]}{\partial V_2}\right|_{\bar{V}_2} \delta V_2$$

System linearization – cont'd

POLITECNICO MILANO 1863

Two approaches:
1. Time-based linearization block using the Simulink model
2. **By hand**

$$\delta\dot{x} = \left.\frac{\partial f(x,u)}{\partial x}\right|_{\bar{x},\bar{u}} \delta x + \left.\frac{\partial f(x,u)}{\partial u}\right|_{\bar{x},\bar{u}} \delta u$$



$$\delta\dot{h}_1 = -\frac{A_1\sqrt{2g}}{S\sqrt{\bar{h}_1}}\delta h_1 + \boxed{\left.\frac{\partial[k_2(V_2)\gamma_2(V_2)\,V_2]}{\partial V_2}\right|_{\bar{V}_2}}\delta V_2$$

$$\delta\dot{h}_2 = \frac{A_1\sqrt{2g}}{S\sqrt{\bar{h}_1}}\delta h_1 - \frac{A_2\sqrt{2g}}{S\sqrt{\bar{h}_2}}\delta h_2 + \boxed{\left.\frac{\partial\left[k_1(V_1)\left(1-\gamma_1(V_1)\right)V_1\right]}{\partial V_1}\right|_{\bar{V}_1}}\delta V_1$$

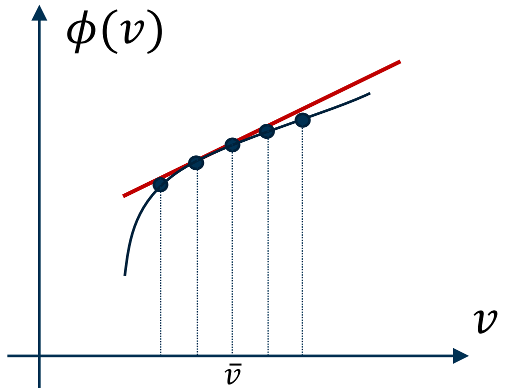
$$\delta\dot{h}_3 = -\frac{A_3\sqrt{2g}}{S\sqrt{\bar{h}_3}}\delta h_3 + \boxed{\left.\frac{\partial[k_1(V_1)\gamma_1(V_1)\,V_1]}{\partial V_1}\right|_{\bar{V}_1}}\delta V_1$$

$$\delta\dot{h}_4 = \frac{A_3\sqrt{2g}}{S\sqrt{\bar{h}_3}}\delta h_3 - \frac{A_4\sqrt{2g}}{S\sqrt{\bar{h}_4}}\delta h_4 + \boxed{\left.\frac{\partial\left[k_2(V_2)\left(1-\gamma_2(V_2)\right)V_2\right]}{\partial V_2}\right|_{\bar{V}_2}}\delta V_2$$

Since κ_j and γ_j are numerical maps, these terms should be **estimated numerically**

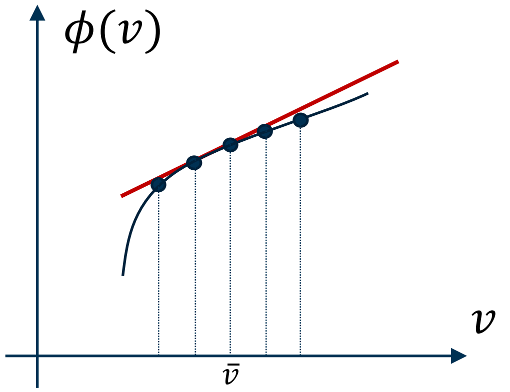
Five-point stencil derivative approximation

$$\left.\frac{d\phi(v)}{dv}\right|_{\bar{v}} \approx \frac{\phi(\bar{v}-2h) - 8\phi(\bar{v}-h) + 8\phi(\bar{v}+h) - \phi(\bar{v}+2h)}{12h}$$



This method can be applied to estimate the “missing derivatives”

Five-point stencil derivative approximation



$$\left.\frac{d\phi(v)}{dv}\right|_{\bar{v}} \approx \frac{\phi(\bar{v}-2h) - 8\phi(\bar{v}-h) + 8\phi(\bar{v}+h) - \phi(\bar{v}+2h)}{12h}$$

This method can be applied to estimate the “missing derivatives”

<u>Resulting linearized model</u>

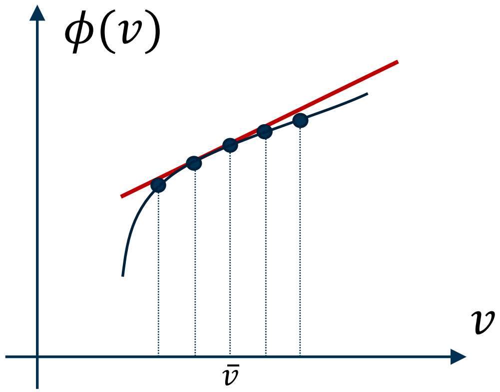
$$\delta\dot{x} = \begin{bmatrix} -0.01714 & 0 & 0 & 0 \\ 0.01714 & -0.09525 & 0 & 0 \\ 0 & 0 & -0.05238 & 0 \\ 0 & 0 & 0.05238 & 0.099 \end{bmatrix} \delta x + \begin{bmatrix} 0 & 0.09701 \\ 0.1207 & 0 \\ 0.1387 & 0 \\ 0 & 0.1595 \end{bmatrix} \delta u$$

$$\delta y = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \delta x$$

Poles: -0.0952, -0.0171, -0.0993, -0.0524
Zeros: -0.0654, -0.0041

Five-point stencil derivative approximation

$$\left.\frac{d\phi(v)}{dv}\right|_{\bar{v}} \approx \frac{\phi(\bar{v}-2h) - 8\phi(\bar{v}-h) + 8\phi(\bar{v}+h) - \phi(\bar{v}+2h)}{12h}$$



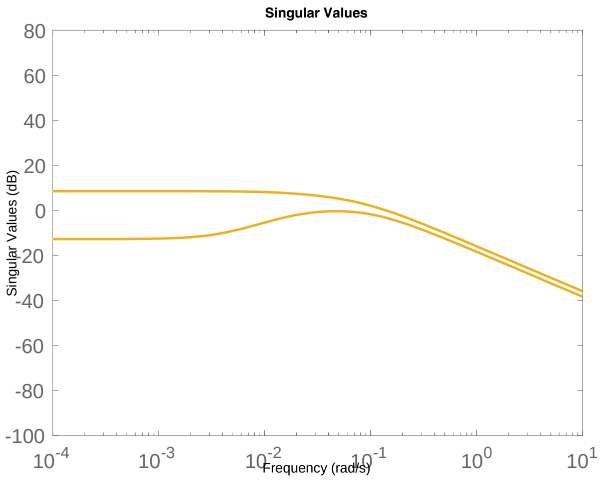
This method can be applied to estimate the "missing derivatives"

Resulting linearized model

$$\delta\dot{x} = \begin{bmatrix} -0.01714 & 0 & 0 & 0 \\ 0.01714 & -0.09525 & 0 & 0 \\ 0 & 0 & -0.05238 & 0 \\ 0 & 0 & 0.05238 & 0.099 \end{bmatrix} \delta x + \begin{bmatrix} 0 & 0.09701 \\ 0.1207 & 0 \\ 0.1387 & 0 \\ 0 & 0.1595 \end{bmatrix} \delta u$$

$$\delta y = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \delta x$$

Poles: -0.0952, -0.0171, -0.0993, -0.0524
Zeros: -0.0654, -0.0041



Design #1 – Decentralized PI

POLITECNICO MILANO 1863

Idea: Design two decentralized PI controllers: $R_1(s)$ to regulate $G_{11}(s)$, and $R_2(s)$ to regulate $G_{22}(s)$, or vice versa.

Design #1 – Decentralized PI

Idea: Design two decentralized PI controllers: $R_1(s)$ to regulate $G_{11}(s)$, and $R_2(s)$ to regulate $G_{22}(s)$, or vice versa.

The input output pairs are decided by inspecting the **Relative Gain Array (RGA)** matrix

$$RGA = G(0) \odot (G(0)^{-1})^T = \begin{bmatrix} \lambda & 1-\lambda \\ 1-\lambda & \lambda \end{bmatrix}$$

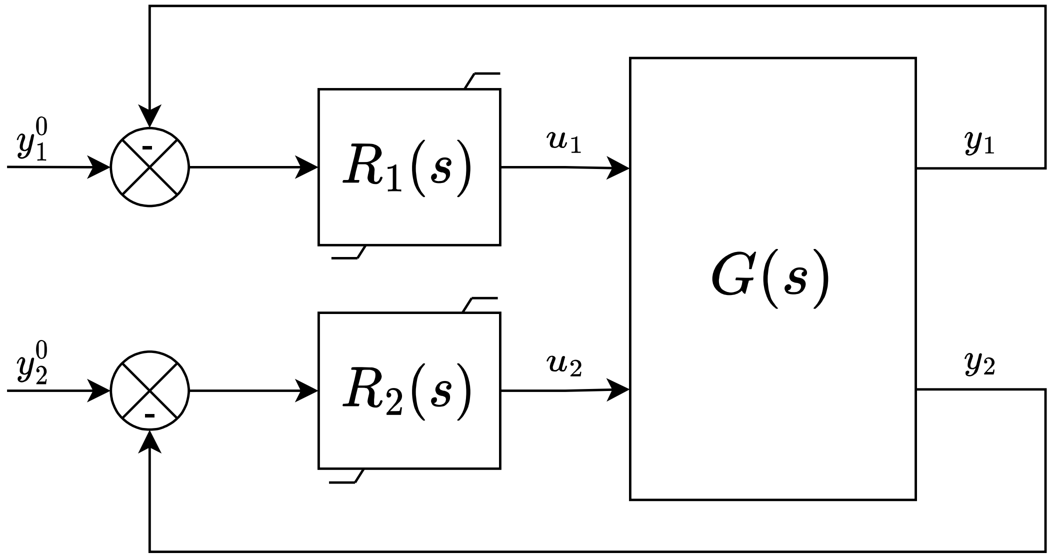
In our case, $\lambda \approx 3.3$, which indicates

- **Very strong coupling** between the variables
- Best pairs: (u_1, y_1) and (u_2, y_2)

Design #1 – Decentralized PI

POLITECNICO MILANO 1863

Idea: Design two decentralized PI controllers: $R_1(s)$ to regulate $G_{11}(s)$, and $R_2(s)$ to regulate $G_{22}(s)$, or vice versa.



The input output pairs are decided by inspecting the **Relative Gain Array (RGA)** matrix

$$RGA = G(0) \odot (G(0)^{-1})^T = \begin{bmatrix} \lambda & 1-\lambda \\ 1-\lambda & \lambda \end{bmatrix}$$

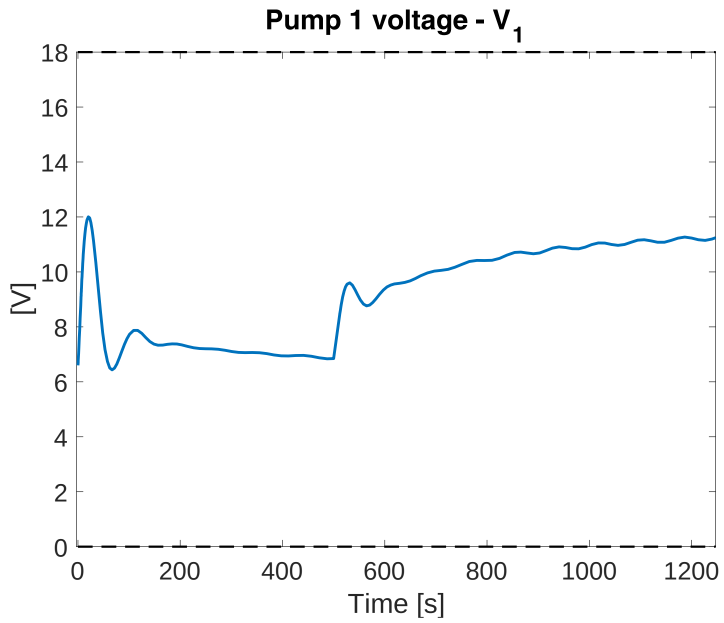
In our case, $\lambda \approx 3.3$, which indicates

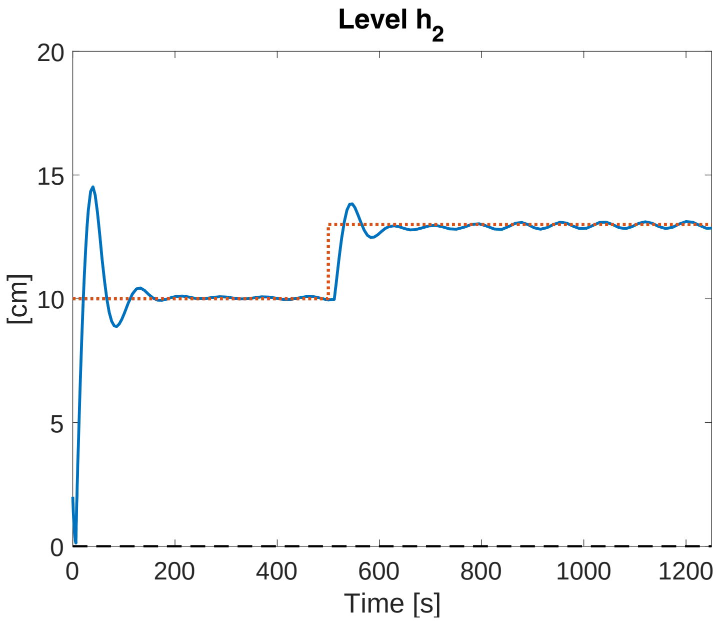
- **Very strong coupling** between the variables
- Best pairs: (u_1, y_1) and (u_2, y_2)

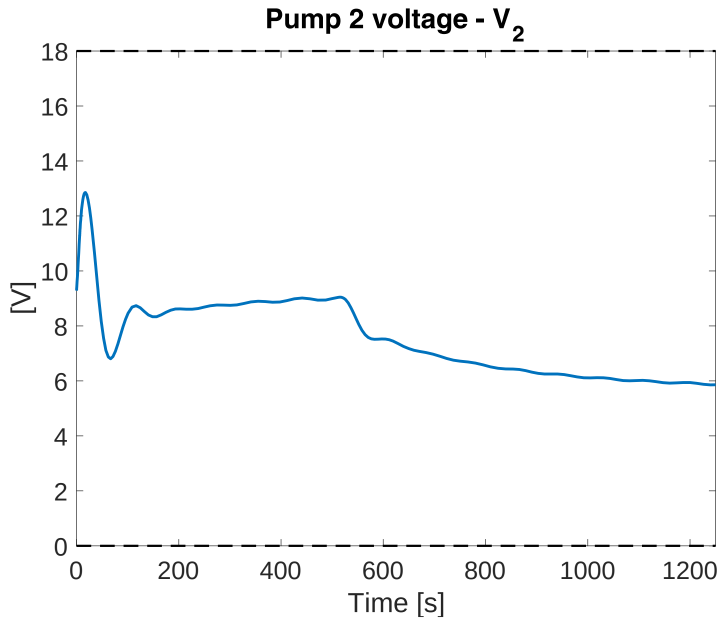
```
R1 = pidtune(Gs(1, 1), 'PI', 0.05);
R2 = pidtune(Gs(2, 2), 'PI', 0.05);
```

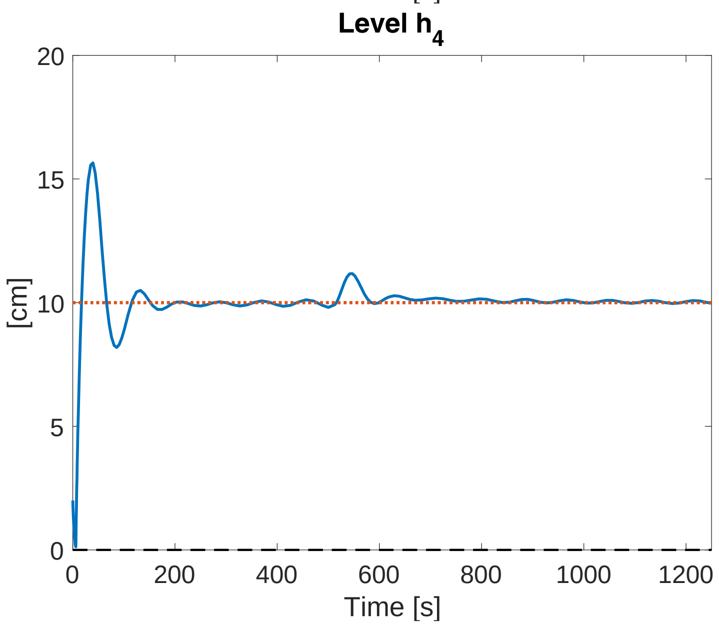
Remark: Don’t forget to equip the regulator with some **anti-windup** action!

Design #1 – Performances









Design #1 – PROs and CONs of decentralized PI

POLITECNICO MILANO 1863

- Very **easy to design**, it's just 1 line of code for the RGA and 1 line for the regulator design
- Allows to easily include control saturation and anti-windup strategies

- In cases where there are strong cross-couplings ($\lambda > 1$), may not work well.
- Naïve scheme: in general, it may **lead to instability**!

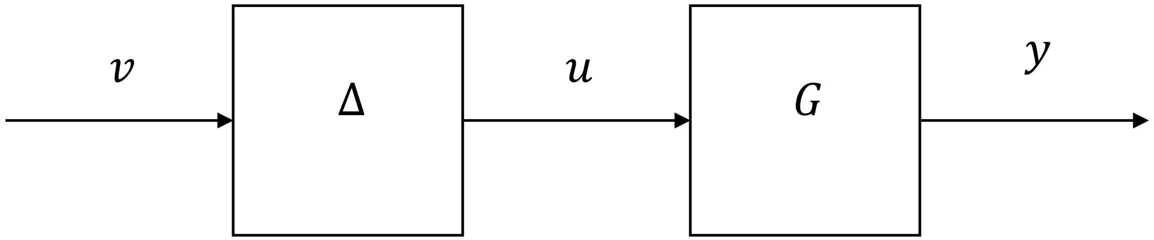
We can do something better, resorting to a "decoupler"!

Design #2 – Backward-decoupled PI

POLITECNICO MILANO 1863

Idea: Design $\Gamma_{12}(s)$ and $\Gamma_{21}(s)$ so that

$$G(s)\Delta(s) = \begin{bmatrix} G_{11}(s) & 0 \\ 0 & G_{22}(s) \end{bmatrix}$$



Design #2 – Backward-decoupled PI

POLITECNICO MILANO 1863

Idea: Design $\Gamma_{12}(s)$ and $\Gamma_{21}(s)$ so that

$$G(s)\Delta(s) = \begin{bmatrix} G_{11}(s) & 0 \\ 0 & G_{22}(s) \end{bmatrix}$$

If $G(s)$ is asymptotically stable and does not have unstable zeros, one can use

$$\Delta(s) = \left(I - \begin{bmatrix} 0 & \Gamma_{12}(s) \\ \Gamma_{21}(s) & 0 \end{bmatrix}\right)^{-1}$$

$$\left.\begin{aligned} \Gamma_{12}(s) &= -\frac{G_{12}(s)}{G_{11}(s)} = -\frac{0.0138}{s + 0.017} \\ \Gamma_{21}(s) &= -\frac{G_{21}(s)}{G_{22}(s)} = -\frac{0.0456}{s + 0.052} \end{aligned}\right\}$$

(strictly) proper

Design #2 – Backward-decoupled PI

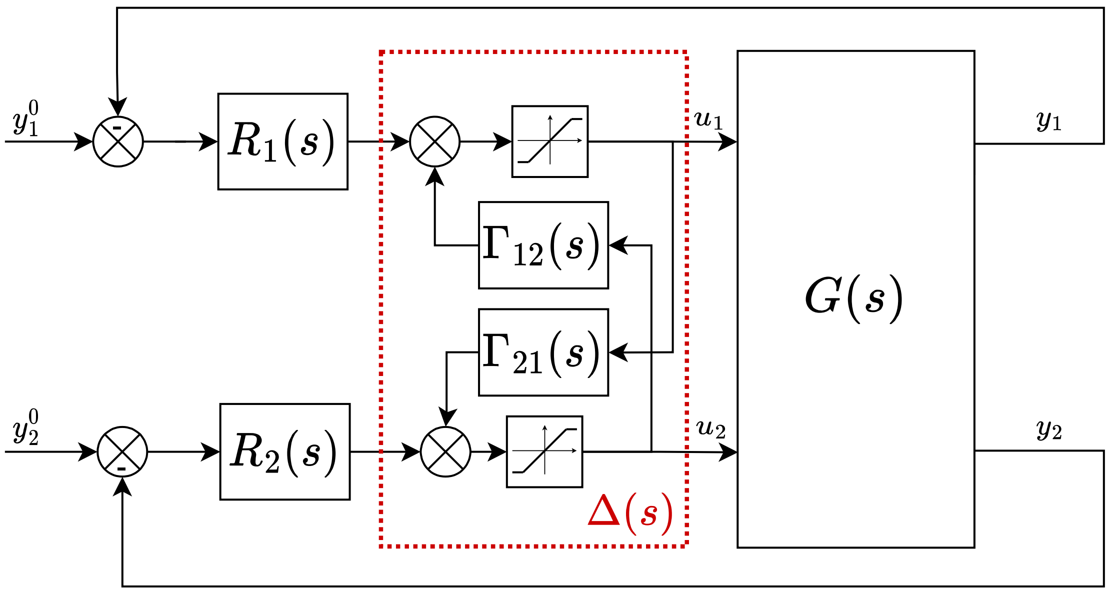
Idea: Design $\Gamma_{12}(s)$ and $\Gamma_{21}(s)$ so that

$$G(s)\Delta(s) = \begin{bmatrix} G_{11}(s) & 0 \\ 0 & G_{22}(s) \end{bmatrix}$$

If $G(s)$ is asymptotically stable and does not have unstable zeros, one can use

$$\Delta(s) = \left(I - \begin{bmatrix} 0 & \Gamma_{12}(s) \\ \Gamma_{21}(s) & 0 \end{bmatrix}\right)^{-1}$$

$$\left.\begin{aligned} \Gamma_{12}(s) &= -\frac{G_{12}(s)}{G_{11}(s)} = -\frac{0.0138}{s + 0.017} \\ \Gamma_{21}(s) &= -\frac{G_{21}(s)}{G_{22}(s)} = -\frac{0.0456}{s + 0.052} \end{aligned}\right\} \textbf{(strictly) proper}$$



$R_1(s)$ and $R_2(s)$ can be then designed based on $G_{11}(s)$ and $G_{22}(s)$

Design #2 – Backward-decoupled PI w/ Anti-windup

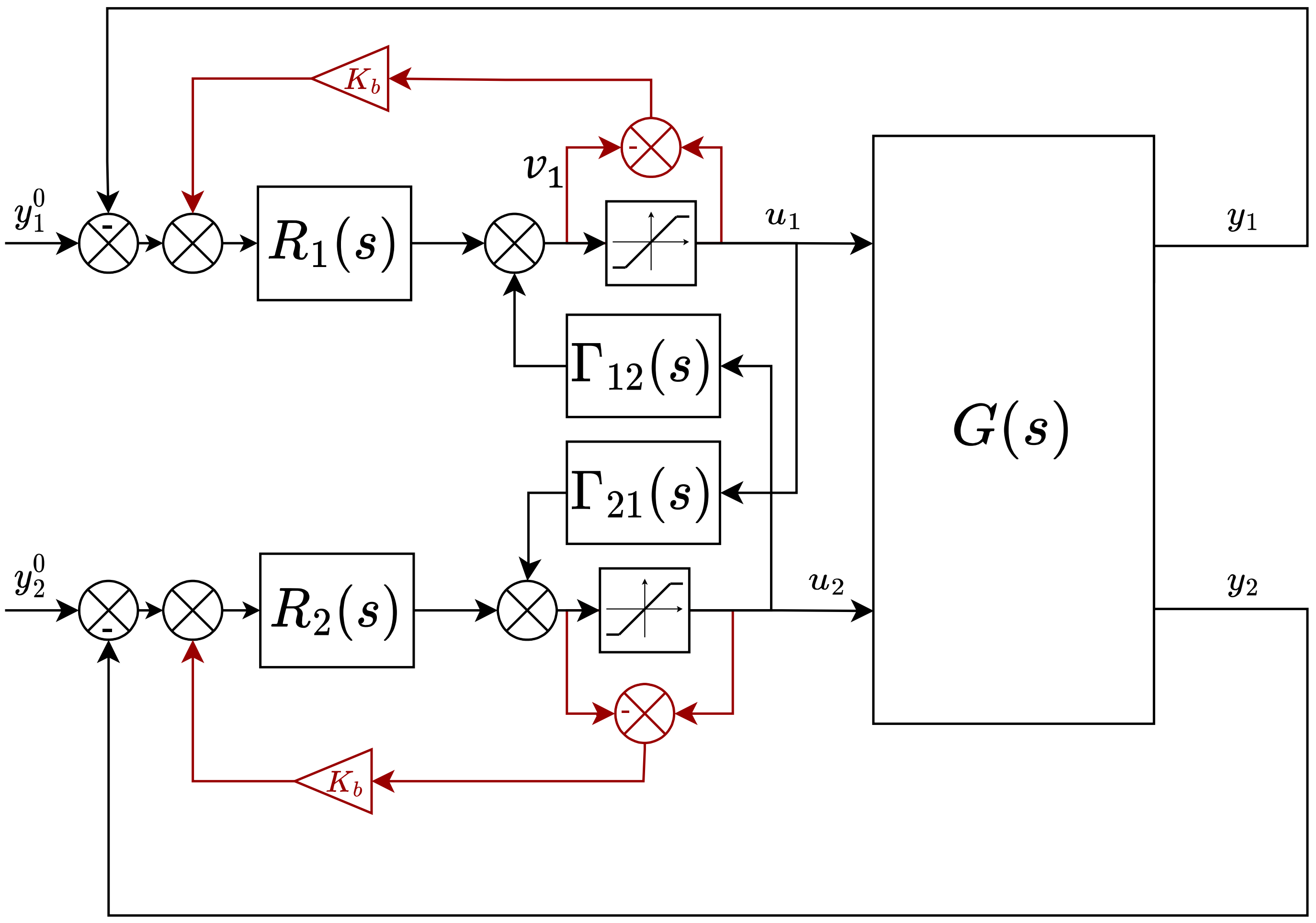
If

$$v_1 > u_{max}$$

Then

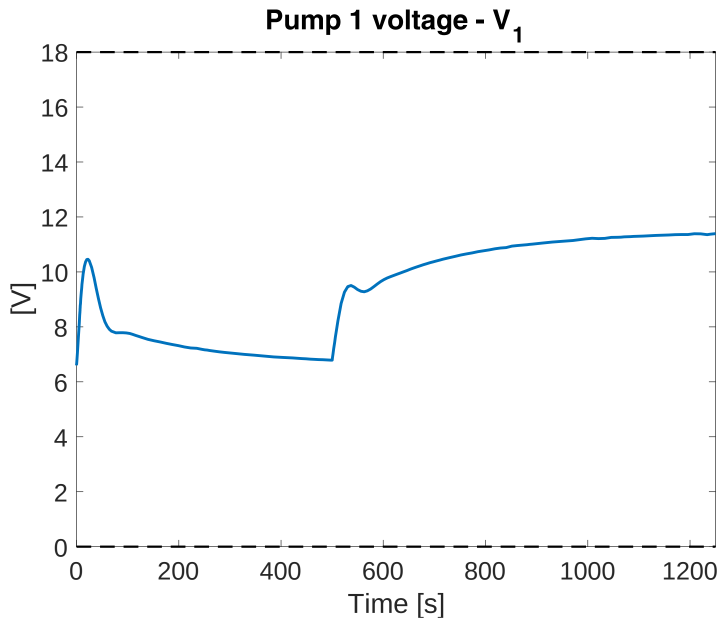
$$-v_1 + u_1 < 0$$

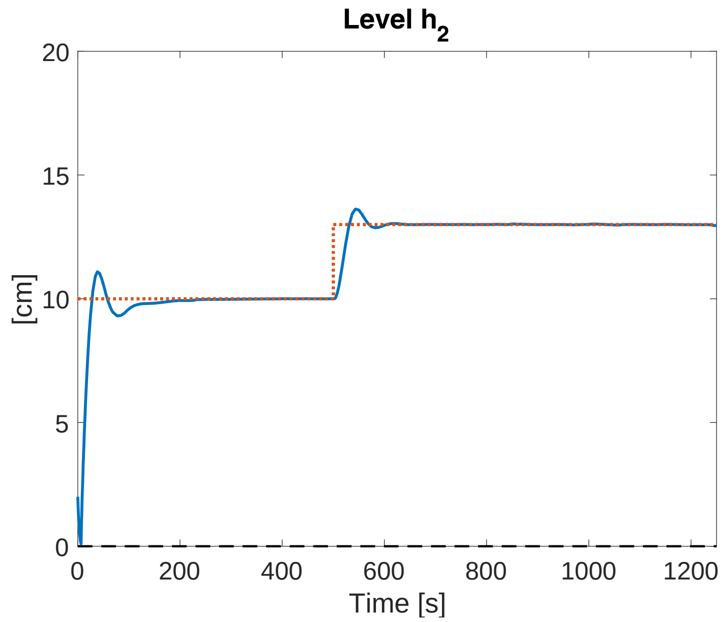
Reducing the input of the regulator

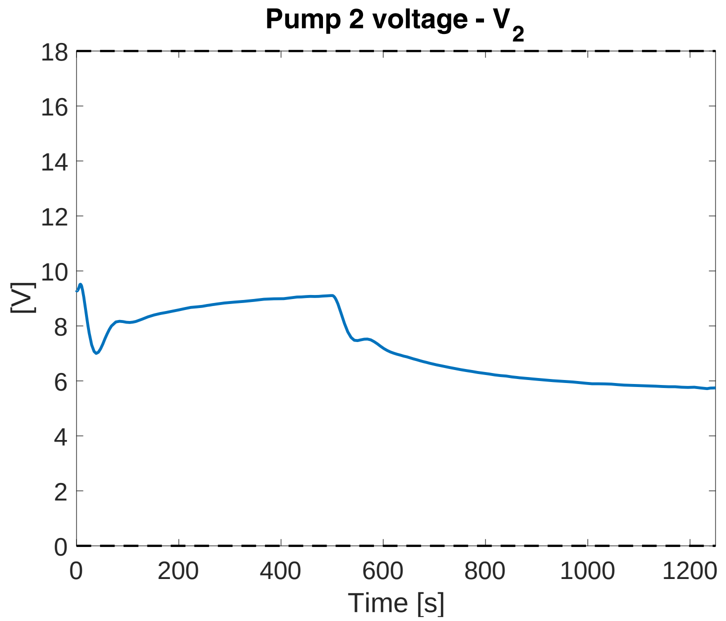


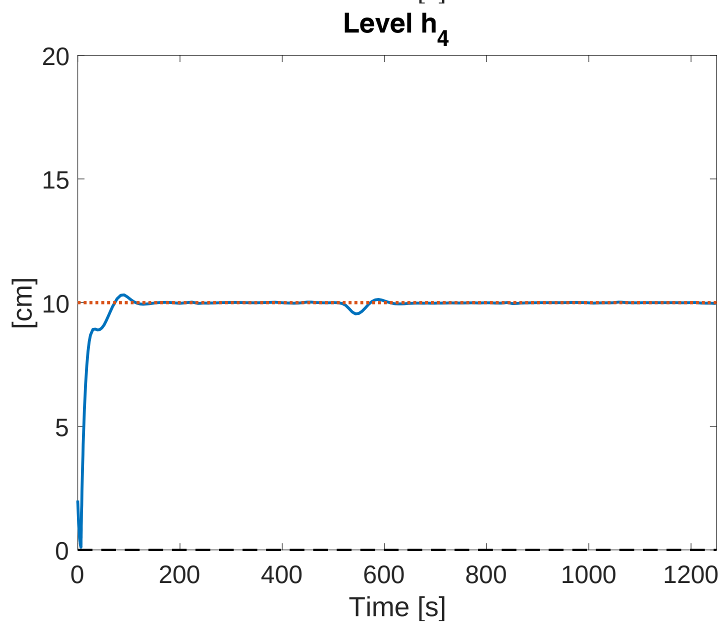
K_b is the backcalculation gain

Design #2 – Performances







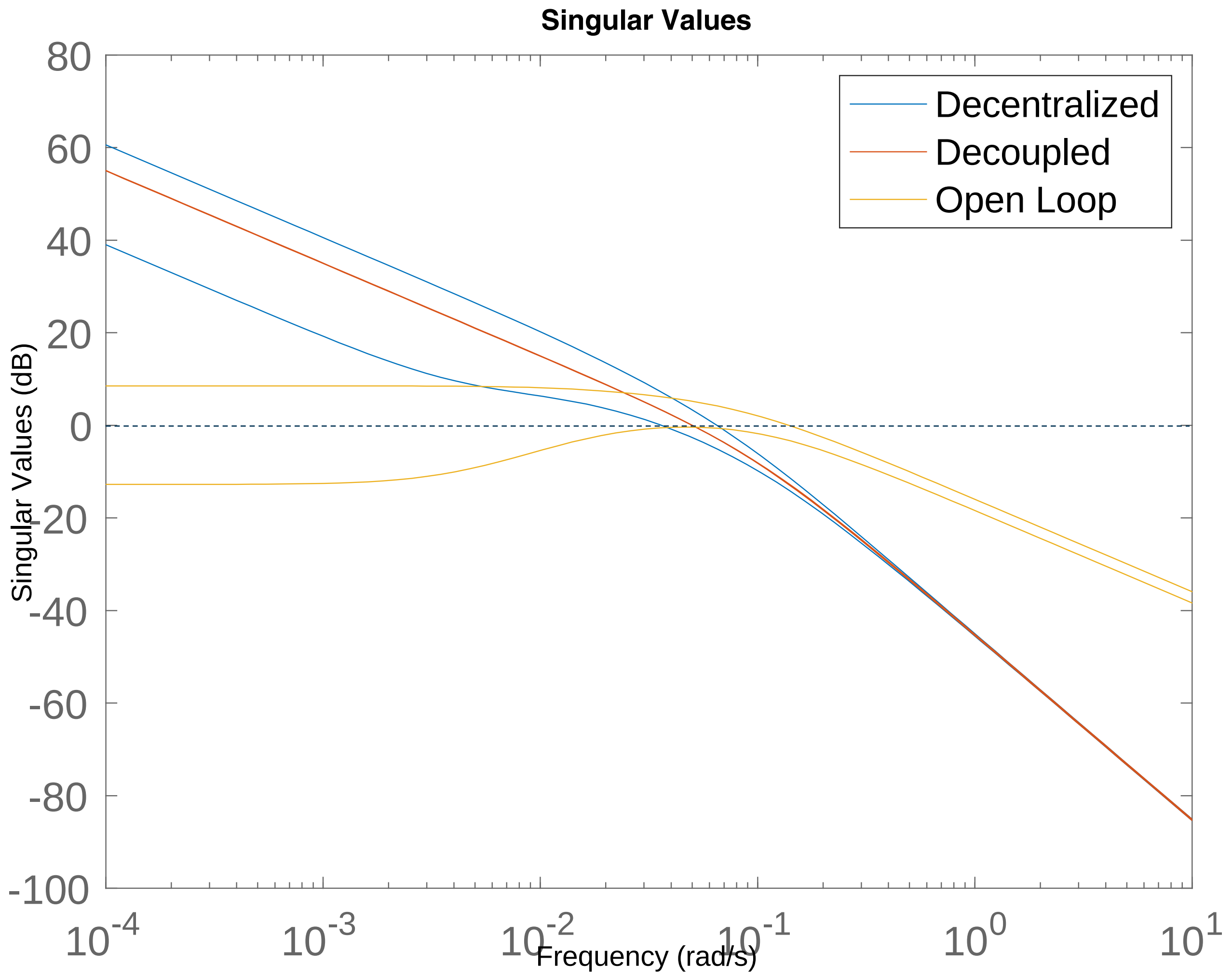


Design #2 – PROs and CONs of backward-decoupled PI

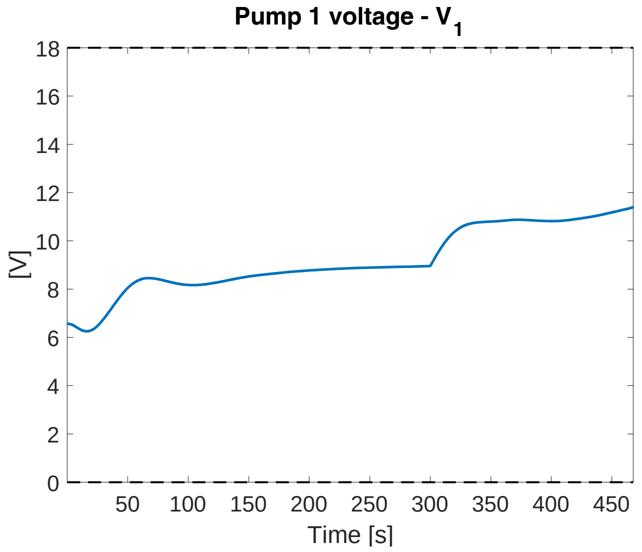
- Stability and performance guarantees even for strongly coupled systems

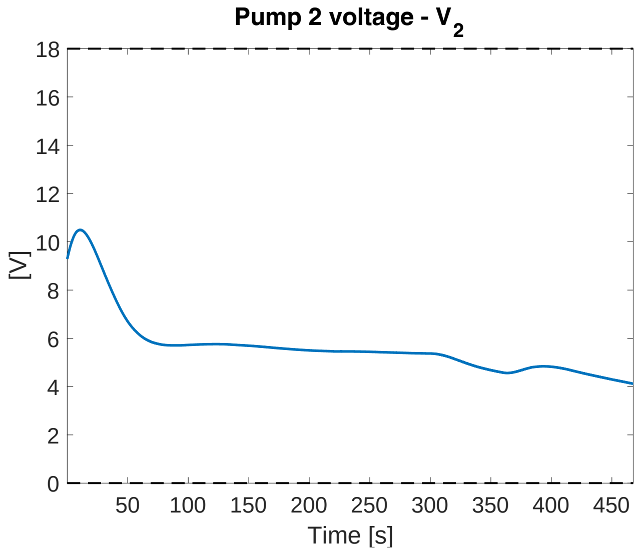
- Slightly harder design, anti-windup not trivial
- Model-sensitive: inaccurate models could lead to worse performances than decentralized PIs

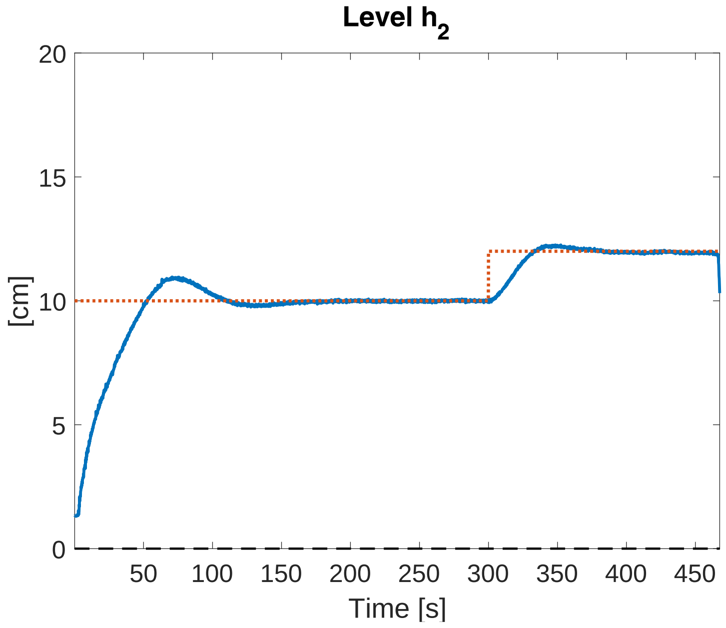
Comparison of the singular values

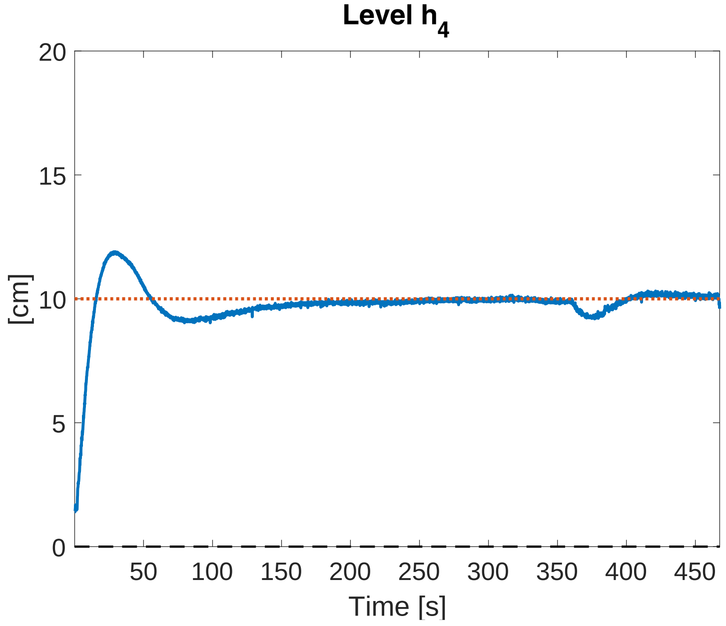


Validation on the real apparatus









Instructions for the lab experiments

POLITECNICO MILANO 1863

1. **Make sure your group is selected** (see WeBeep) TODO

2. **Fill your availabilities:** https://doodle.com/meeting/participate/id/b6z6Oy9b

Instructions for the lab experiments

1. **Make sure your group is selected** (see WeBeep) TODO

2. **Fill your availabilities:** https://doodle.com/meeting/participate/id/b6z6Oy9b

3. I’ll communicate the schedules on next Wednesday

4. Prepare:
    - A .mat file with all and only the necessary parameters (gains, variables, equilibria, ecc)
    - A Simulink scheme containing a subsystem having
        - $y^0(t)$ and $x(t)/y(t)$ as inputs
        - $u(t)$ as output
        - All the components must be discrete-time blocks having sampling time $\tau_s = 0.01s$

Thanks for your attention!