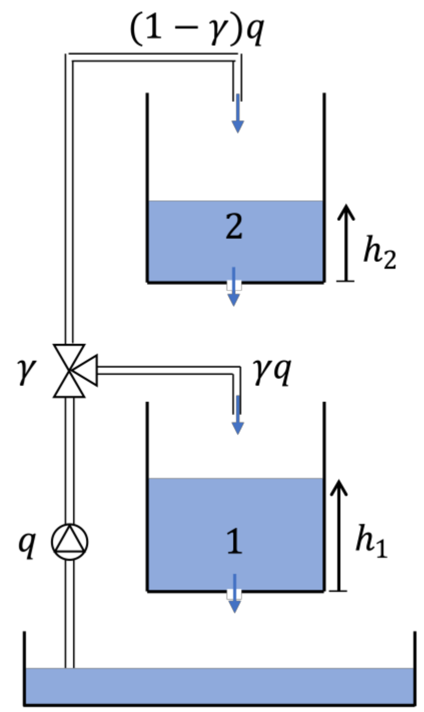
Model Predictive Control of a Two-Tank system



The Two-Tank system, depicted above, consists of two tanks fed by a volumetric water pump q. A valve splits the flow rate in γq, which feeds Tank 1, and in $(1-\gamma)q$, which feeds Tank 2. The dynamic model of the system is

$$\begin{cases} \dot{h}_1 = -\dfrac{a_1}{S_1}\sqrt{2gh_1} + \dfrac{a_2}{S_1}\sqrt{2gh_2} + \dfrac{\gamma}{S_1}q \\ \dot{h}_2 = -\dfrac{a_2}{S_2}\sqrt{2gh_2} + \dfrac{1-\gamma}{S_2}q \end{cases}$$

State: $x = [h_1, h_2]'$ Input: $u = q$ Output: $y = x$

Where the parameters are the following:

- $S_1 = S_2 = 0.06\, m^2$ are the Tank's cross-sections
- $a_1 = 1.31 \cdot 10^{-4}\, m^2$ and $a_2 = 9.57 \cdot 10^{-5}\, m^2$ are the cross-sections of the openings
- $\gamma = 0.4$ is the valve splitting coefficient

The system is subject to physical constraints

$$0.1\, m \le h_1 \le 1.3\, m$$
$$0.1\, m \le h_2 \le 1.2\, m$$

Also, the pump's flow rate is subject to saturation

$$10^{-4}\, {m^3}/{s} \le q \le 10^{-3}\, {m^3}/{s}$$

Goal of the control system: Track constant references for the level h_1 while respecting the system's and actuator's constraints.

Tasks – Part A

1. Compute the equilibrium $(\bar{h}_1, \bar{h}_2, \bar{q})$ corresponding to $\bar{h}_1 = 0.8$.
2. Linearize the system around such equilibrium and design an LQ control strategy.
   Then, test the closed-loop performances in Simulink, considering the following initial state and reference trajectory

$$h_1(0) = 0.5\ m$$
$$h_2(0) = 0.2\ m$$
$$h_1^{ref}(t) = \bar{h}_1 + 0.45 \cdot \text{step}(t - 1000)$$

   Are the system constraints respected?

Tasks – Part B

3. Design and implement a nonlinear Model Predictive Control law with zero terminal constraint. This task may be divided in the following sub-tasks:
   a. Write a MATLAB function `compute_equilibrium` which computes the equilibrium $\bar{x}, \bar{u}$ for an arbitrary $\bar{h}_1$

$$[\bar{x}, \bar{u}] = \text{compute\_equilibrium}(\bar{h}_1)$$

   b. Write a MATLAB function `model_step` which implements the system dynamics, discretized e.g. via Forward Euler with sampling time τ_s:

$$[x_{k+1}, y_k] = \text{model\_step}(x_k, u_k, \tau_s)$$

   c. Write a MATLAB function FHOCP which solves MPC's optimal control problem at the current time instant

$$u_k^\star = \text{FHOCP}(x_k, Q, R, N, \bar{x}, \bar{u}, \tau_s)$$

   d. Implement the MPC law in Simulink using a MATLAB System Block[1], and test the closed loop in the following scenario

$$h_1(0) = 0.5\ m$$
$$h_2(0) = 0.2\ m$$
$$h_1^{ref}(t) = 0.8 + 0.45 \cdot \text{step}(t - 1000)$$

   using different prediction horizons N, weight matrices Q and R, sampling frequencies τ_s.

4. Assume now that the actuator is affected by a maximum variation rate constraint, i.e.

$$|\,\dot{q}\,| \le 0.5 \cdot 10^{-5}\, {m^3}/{s^2}$$

   Modify the MPC law to ensure the satisfaction of the constraint.

[1] https://it.mathworks.com/help/simulink/ug/what-is-matlab-system-block.html

Summary of CasADi's main commands

Installation instructions: https://web.casadi.org/get/

- Import CasADi and instantiate an optimization problem

```
import casadi.*;
opti = casadi.Opti();
```

- Declare a $n \times N$ optimization variable

```
x = opti.variable(n, N);
```

- Declare an equality or inequality constraint

```
opti.subject_to(x(:, 1) == ones(n, 1));
opti.subject_to(0 <= x(:, 1) <= 2);   %  Inequality applied element-wise
```

- Declare the cost function

```
J = 0
for k=1:N
      J = J + x(:, k).' x(:, k)
end
opti.minimize(J)
```

- Declare the solver (default: ipopt)

```
opti.solver('ipopt', prob_opts, ip_opts);
```

- Fire the solution of the optimization problem

```
try
      sol = opti.solve()
catch EX:
      keyboard;          % Enter debug mode
end
```

- Extraction of the values of the optimal solution

```
if sol.stats.success == 1
      x_opt = sol.value(x);
end
```

Additional commands

- Set the initial guess of an optimization variable (when available)

```
opti.set_initial(x, x_initial_guess);
```