Multivariable control of tank-system

Computer exercises in Control Theory, TSRT06/TSRT09

12 november 2020

1 Introduction

The purpose of these exercises is to provide opportunities to study various aspects of multivariable dynamic systems and multivariable control systems. This is done by using a simulation model of a system consisting of four interconnected tanks. The system is described in detail in Section 2. The model is implemented in Simulink and features an animation that illustrates the behavior of the system. The simulation model is described in more detail in Section 3. . A brief review of useful MATLAB commands is available in Section 4.

2 System description

The system is a non-linear simulation model of a tank system consisting of four interconnected tanks. A schematic image of the tank system is shown in figure 2. The simulation model imitates a lab process[1] developed at Lund University of Technology.

Tank 1 and tank 2 are at the lower left and the lower right respectively, while tank 3 and 4 are the upper left and right respectively. The inputs to the system are the voltages to the two pumps, where pump 1 is the left one. How water from the pumps is distributed to the tanks is adjusted by the paramters γ_i.

By representing the levels in the tanks with the variables x_i and denote the voltages of the pumps with the variables u_i the system can be described by the equations

$$\begin{aligned}
\dot{x}_1 &= -\frac{a_1}{A_1}\sqrt{2gx_1} + \frac{a_3}{A_1}\sqrt{2gx_3} + \frac{\gamma_1 k_1}{A_1}u_1 \\
\dot{x}_2 &= -\frac{a_2}{A_2}\sqrt{2gx_2} + \frac{a_4}{A_2}\sqrt{2gx_4} + \frac{\gamma_2 k_2}{A_2}u_2 \\
\dot{x}_3 &= -\frac{a_3}{A_3}\sqrt{2gx_3} + \frac{(1-\gamma_2)k_2}{A_3}u_2 \\
\dot{x}_4 &= -\frac{a_4}{A_4}\sqrt{2gx_4} + \frac{(1-\gamma_1)k_1}{A_4}u_1
\end{aligned} \tag{1}$$

The maximal water-level in the tanks is $x_i = 50$.

[1] K-H Johansson. "The Quadruple Tank Process - A Multivariable Laboratory Process with an Adjustable Zero." IEEE Trans. on Control Systems Technology, 1999

In the equations above we use the following notation:

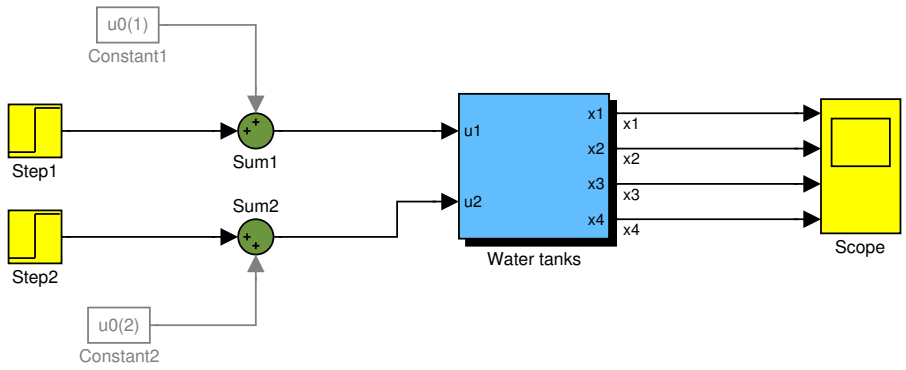
A_i	– surface area for tank i
a_i	– outlet area for tank i
k_i	– gains in pumps
γ_i	– valve setting for distribution

The objective is to control the levels of the lower tanks.

$$\begin{aligned} y_1 &= x_1 \\ y_2 &= x_2 \end{aligned} \tag{2}$$

3 Description of the simulation environment

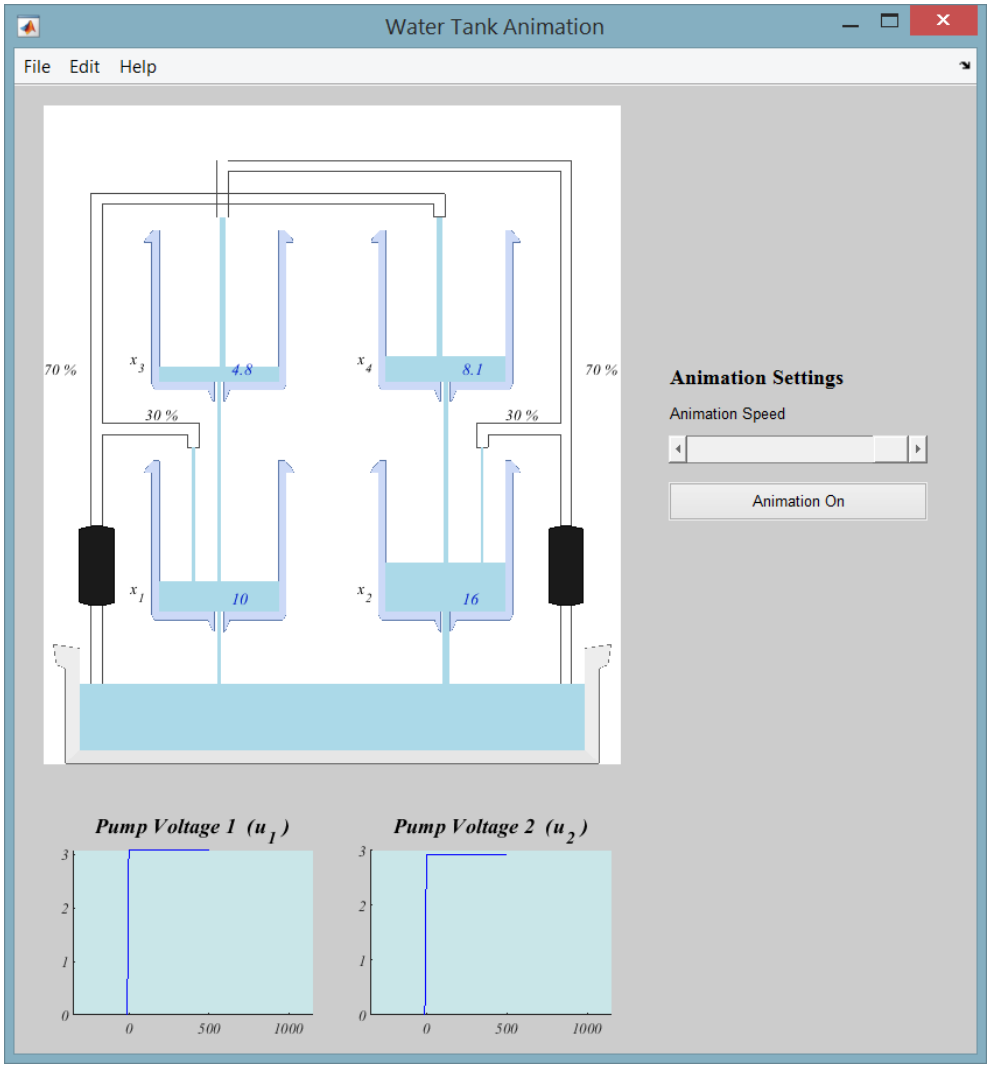
The non-linear tank-model is implemented in Simulink and the model is shown in Figure 1.



Figur 1: Simulink model

The input signals are formed by a sum of a constants (blocks `Constant`) and steps respectively (blocks `Step`). The constant inputs define the stationary point around which we model the system and perform small steps.

The simulation model also has an animation that illustrates the behavior of the tank system. The animation is shown in figure 2.



Figur 2: Animation of the tank-system

The numbers at the branch points and under "Valve Settings" indicate how the flow is distributed between going to the upper and lower tanks respectively. The distribution is determined by the variables γ_1 (left side) and γ_2 (right side). These variables are called `g1` and `g2` in the interface. You can also adjust the speed of the animation (number of updates of the graphics per time unit) as well as turn off the animation completely.

The simulation and animation is used as follows:

- Start MATLAB.
- Define `gamma`, `u0` and `x0` as required in the exercise.
- Write `tanks` to start the animation. When the animation starts, physical parameters in the model are defined (needed for the simulation).
- Write `watertanks` in the command-line of MATLAB to open the Simulink-model.
- With the slider `Animation Speed` it is possible to, within some limits, control the speed of the animation.
- With the button `Animation On/Off` you can turn on and off the animation and simply look at the Scopes in the Simulink model. Simulation is much faster if the animation is turned off.

- To compute a stationary point of the tank-system, the MATLAB-function `statpoint` is provided. It can be used in two ways:

  1. Given the power to the pumps, u_1 and u_2, as well as the valve positions γ_1 and γ_2, compute the stationary water levels in the tanks.

     ```
     x = statpoint([u1 u2],[],[g1 g2])
     ```

  2. Given the water levels of tanks 1 and 2 and the valve-positions γ_1, γ_2, compute the stationary levels of all tanks and the power to the pumps (u_1 and u_2) that are required to maintain it.

     ```
     [x,u] = statpoint([],[x1 x2],[g1 g2])
     ```

  See also `help statpoint`.

- With the MATLAB-function `tanklin` a linearized model of the tank-system can be computed in a certain stationary point, given the levels of all tanks. This function is used as follows:

  ```
  [A,B,C,D]=tanklin(x,[g1 g2])
  ```

  `A, B, C` and `D` are the system matrices in the stationary point given by the variable `x`. See also `help tanklin`.

4 Useful MATLAB-commands

The following MATLAB-commands can be useful.

`eig`	Compute eigenvalues
`ss`	Construct LTI-object on state-space form
`tf`	Construct LTI-object as transfer function
`bode`	Draw bodediagram
`sigma`	Draw singular values
`pole`	Compute poles
`zero`	Compute zeros
`place`	Compute a state feedback that places the closed loop system poles at desired pos
`freqresp`	Compute $G(ix)$
`evalfr`	Compute $G(x)$

5 Impact of stationary point

Task:

Determine the stationary points of the system by simulation in a few different cases and investigate how the system's properties depend on the stationary point.

Hints:

- Create a variable `gamma` with $\gamma_1 = \gamma_2 = 0.3$ (note what that means physically in terms of water flow in the system)
- Check that final values in the `Step`-blocks are 0.
- Set the initial state `x0` to 0 (i.e. a vector of length 4 with zeros).
- Create a variable `u0`, first with input signals $u_1 = u_2 = 3$ and then $u_1 = u_2 = 5$.
- Simulate the system until stationarity, and note the stationary levels for the two cases (you can read these either in the animation plots, the scopes, or check the variable `x` which has been saved to the workspace)
- Determine the linearized system using the function `tanklin` and compute poles and zeros in the two cases and compare them.
- Confirm that the stationary points found by simulation coincides with analytic result you obtain when using `statpoint` with the selected input levels

```
[xs,us] = statpoint(u0,[],gamma)
```

- Useful MATLAB-commands: `eig, ss, zero`.

Summary of results and observations:

6 Effects of the valve positions

Task:

Investigate how the system properties depend on the setting of the valves. Be particulariy observant on zeros, and relate to the physical setup and resulting water flow and why some configurations intuitively should be hard to control.

Hints:

- Study the system around the stationary point given by input signal combination $u_1 = u_2 = 3$.

```
[xs,us] = statpoint(u0,[],gamma)
[A,B,C,D]=tanklin(xs,[gamma])
```

- Study for instance the cases $\gamma_1 = \gamma_2 = 0.3, \gamma_1 = \gamma_2 = 0.5$ and $\gamma_1 = \gamma_2 = 0.7$.
- Determine poles, zeros and singular values for models linearized in the stationary points defined by the different cases.
- Useful MATLAB-commands: `eig, zero, ss, sigma`.

Summary of results and observations:

7 Impact of the direction of the input signal vector

Task:

Examine how the system's step response depends on the direction of the input vector at different valve settings.

Hints:

- Use the computed stationary point (stationary point) and settings as in the previous task.
- Set the initial values of the tank levels using the variable `x0` so that the system is started in its stationary point (you will always do this from now on, so this must be run when you change anything)

```
[x0,u0] = statpoint(u0,[],gamma)
```

- Select the amplitude of the blocks `Step` to make steps of a few centimeters (i.e. not too large as that moves us way from the stationary point where our linearized model is valid)
- Evaluate the transfer matrix of the system for $s = 0$ and choose the input vector so as to achieve maximal resp. minimal gain. Use Exempel 3.4 on page 76 in the course book as help to choose the input signal. **Note!** Printing error in the book on page 77. In the MATLAB-code it should be
  `» [V,D]=eig(g0'*g0)`
- Useful MATLAB commands: `freqresp, eig, sigma`.

Summary of results and observations:

8 The meaning of a zero(*)

Task:

Verify equation (6) in Appendix A on the tank-system.

Hints:

- Choose $\gamma_1 = \gamma_2 = 0.3$, which gives a zero in the right half plane.
- Choose a stationary point that correspond to water levels roughly in the middle of span for the upper tanks.
- Compute the linearized system and the zero z that is in the right half plane. Compute a control input u^0 that satisfies equation (5).
- Of course, one cannot apply an exponentially growing control signal particularly long without the upper tanks either flood or dry out. If necessary, scale down u^0 so that it can simulate at least in $500-600$ time units.
- Simulate the tank-system with the input $u^0 e^{zt}$. Verify (6).
- The formula (6) assumes a linear system. The equation is no longer valid when non-linear effects are too large. What non-linear effects can be observed?
- Useful MATLAB-commands: `zero, evalfr, eig`.

9 RGA and single-variable control

Task:

Use RGA to pair each input with an output. Use a PID controller for each such control circuit.

Remember that the PID-controllers should act on the deviations away from the stationary point and the references are steps away from the stationary point. Hence, when defining the control error you must not forget to substract the stationary point from the measurement, and the input to the system should be the signal computed by the controller plus the stationary point input.

Hints:

- Test and comment on the differences between the cases $\gamma_1 = \gamma_2 = 0.7$, $\gamma_1 = \gamma_2 = 0.3$ and $\gamma_1 = \gamma_2 = 0.5$. It may be necessary to linearize around different stationary points in these cases for the levels to become sensible.
- Do not forget to set initial values. If they are set correctly, nothing should happen until the steps occur.
- First, check the static RGA (i.e. at $\omega = 0$), but try other ω values as well.
- MATLAB code for calculating RGA can be found on page 244 in the textbook.
- When you make steps, make sure they appear at different times (Step Time in the block), so you can separate the effects on the different tanks.
- Useful MATLAB-commands: `freqresp`.

Summary of results and observations:

10 Decoupling (frikoppling)

Task: Investigate how couplings can be reduced using a static decoupling $\bar{F}$ where $\bar{F}$ is a constant matrix which is used to distribute the output from a simple multivariable PID controller. In other words

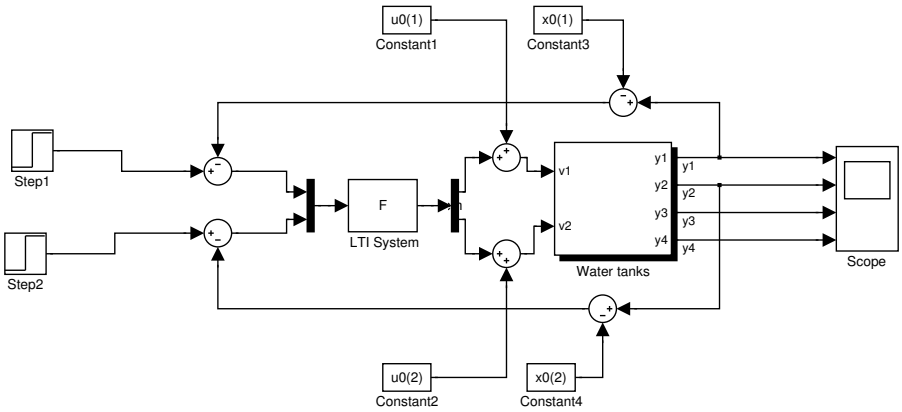
$$U(s) = U_0(s) + \bar{F}F_{diag}(s)(R(s) - (Y(s) - Y_0(s)))$$

You can probably reuse your previous PID controllers.

Hints:

- Introduce, e.g., the controller using a single block of the type `LTI System` which can be found in the library `Blocksets&Toolboxes`. Alternatively build it using a series connection of a matrix-vector multiplication and the PID controllers.
- Use blocks of the type `Mux` och `Demux` to create vectors from several scalars and vice versa. These blocks can be found in the library `Signals&Systems`.
- The constant input signal defining the stationary point is added **efter** the decoupling.
- Repeat experiments with the different valve settings.
- Note how the voltages behave in the different settings when you perform steps on the different tanks.

Sammanfattning av resultat och observationer:



Figur 3: Typical Simulink model with a multivariable controller working with deviations from a working-point.

11 Removed (but kept for numbering)

12 Control without feedforward

Task: In the previous controllers we have used a feedforward term with the working-point input. Remove these and get your controller to work anyway

$$U(s) = F(s)(R(s) - (Y(s) - Y_0(s)))$$

Tips:

- Remove the feedforward terms.
- Start with a pure P-controller setting and $\gamma_1 = \gamma_2 = 0.7$ and then try to improve the controller to remove stationary errors.
- Test the developed controller on the system when you change the water tank to have $\gamma_1 = \gamma_2 = 0.3$.
- Try to achieve similiar performance also when you remove the decoupling.

13 State feedback

Task: Assume that all states in the system can be measured and introduce state feedback on the form

$$u(t) - u_0 = -L(x(t) - x_0) + L_0 r(t)$$

Study how the choice of poles for the feedback system affects the speed of the system and the size of the control signals.

Hints:

- Consider that the state-feedback is derived for the linearized system

  $$\frac{d}{dt}(x - x_0) = A(x - x_0) + B(u - u_0),\ u - u_0 = -L(x - x_0) + L_0 r(t)$$

  where x_0, u_0 correspond to the the stationary point.

- Test for instance the cases $\gamma_1 = \gamma_2 = 0.3$ and $\gamma_1 = \gamma_2 = 0.7$.

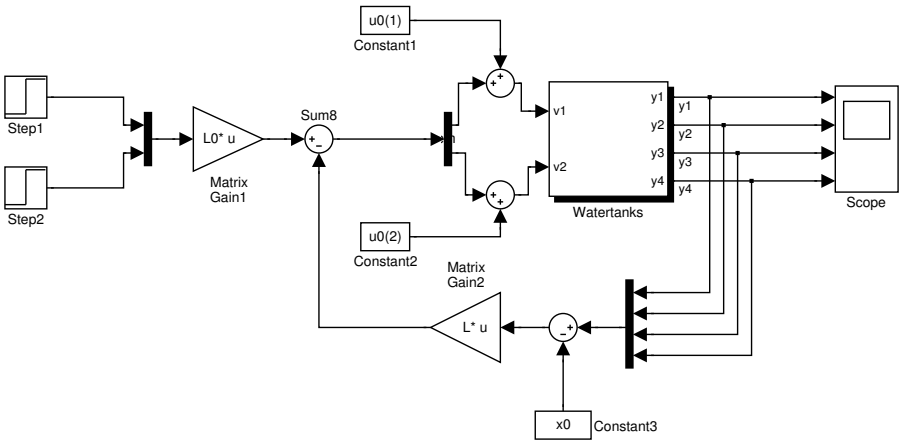
- Pole placement state feedback for multivariable systems can be performed with the `place` function. Note however that in this case a maximum of two poles can be placed in the exact same location (computational limitation in the command).

- Begin by making the feedback system slightly faster than the open loop system and then make it progressively faster.

- An example of a Simulink model with state feedback is given in the figure below. The reference signals consist of steps around stationary point `u0`, `x0`.

- Take steps in the respective reference signals individually as well as combinations of them combinations of steps. Preferably, choose different directions of the reference signal.

- Select the matrix L_0 so that the linearized feedback system gets the property $G_c(0) = I$. Since the simulation model is non-linear, this will however not result in that the output signals become exactly the same as the reference signals in steady state.



Figur 4: State feedback around stationary point

Summary of results and observations:

14 LQ state feedback

Task:

Design a controller based on linear quadratic theory. Investigate the case that all tank levels are measured, i.e. that pure state feedback is possible.

Hints:

- Test and comment on the differences between the cases $\gamma_1 = \gamma_2 = 0.7$, $\gamma_1 = \gamma_2 = 0.3$ och $\gamma_1 = \gamma_2 = 0.5$. It may be necessary to linearize around different equilibrium positions in these cases for the levels to become sensible.
- The feed-forward L_0 from the reference signal should be chosen such that $y - y_0 = r$ in steady state.
- The command `lqr` is used.
- Investigate how different weights in the matrices Q_1 and Q_2 affect control performance.
- Because we work with nonlinear system, one might obtain different behaviors for different amplitudes on r. Do some experiments to investigate this.

Summary of results and observations:

15 Observers

Task:

Design an observer that estimates the tank levels from the measured output y. (Since the output signal measures the levels of the lower thoughts, it is the estimates of the upper tank levels that are interesting.)

Hints:

- For example, take $\gamma_1 = \gamma_2 = 0.7$ and select x_3 and x_4 so that the equilibrium levels in all the tanks is in the interval [1030] when `statpoint` is used.
- If we have a linearized model

$$\frac{d}{dt}(x - x_0) = A(x - x_0) + B(u - u_0), \quad y - y_0 = C(x - x_0) \quad (3)$$

(where x_0, u_0, y_0 are equilibrium levels) it is natural to write the observer on the form

$$\frac{d}{dt}\hat{x} = A\hat{x} + B(u - u_0) + K(y - y_0 - C\hat{x}) \quad (4)$$

where $\hat{x}$ is the estimate of $x - x_0$.

- The observer becomes easier to understand if you use the blocks in Simulink with vector-valued signals as inputs/outputs.
- Select K in the observer, for example by placing the eigenvalues to $A - KC$. In MATLAB, this is done with the command `K = place(A', C', p)'` where `p` contains the desired poles. (Why do we need to take the transpose of the matrices?)
- Test different initial values in the observer. Let u be some transient, e.g. a step.
- Add noise to the measured signals, for example, by utilizing Simulink-block "Band-limited white noise". What placements of the eigenvalues of $A - KC$ are particularly sensitive to this?
- Add noise to the inflows. What placements of the eigenvalues of $A - KC$ are particularly sensitive to this? (Note that the observer should not get to "know" the noise. The u that goes into the observer is supposed to be without noise.)

Summary of results and observations:

16 Kalman-filter

Task:

Make the observer in the previous task a Kalman-filter by computing the optimal K.

Hints:

- Apply white noise to measured signals and inflows. Vary the size of these noise terms. The size determines the elements of the matrices R_1 and R_2.
- The Kalman filter can be calculated by the `lqe` command in MATLAB. Note that you can directly get the eigenvalues of $A - KC$ as an output of this function.
- Look in particular at the cases of large measurement noise – small inflow noise, versus small measurement noise – large inflow noise.
- Test some case where the actual noise differs from the values of R_1 and R_2 used to compute K.

Summary of results and observations:

17 LQG

Task:

Design an LQG controller for the case that only the levels of the lower tanks are measured. Make use of the results on LQ state feedback and the Kalman filter. Can you observe any performance limitation when there is a zero in the right half plane?

Hints:

- Test the differences between the cases $\gamma_1 = \gamma_2 = 0.7$, $\gamma_1 = \gamma_2 = 0.3$ and $\gamma_1 = \gamma_2 = 0.5$. It may be necessary to linearize around different stationary points in these cases for the levels to make sense.
- The commands `lqr` and `lqe` in MATLAB solves the algebraic Riccatiekvationerna.
- Also investigate how the control performance is affected by noise on both measurements and inflows.

Summary of results and observations:

A What is a zero?

For a SISO system with transfer function $G(s)$ it holds that if the input signal u equals $e^{\lambda t}$, the output y becomes

$$y(t) = G(\lambda)e^{\lambda t} + \text{ transients}$$

If the system has all poles strictly in the left half plane, the transients die out. The same formula applies in the multivariable case, (where we for simplicity's sake assume that u and y are vectors of the same dimension $= m$) if the applied control signal components u_i are given by $u_i(t) = u_i^0 e^{\lambda t}$. The result is

$$y(t) = G(\lambda)u^0 e^{\lambda t} + \text{ transients}$$

where u^0 is a vector with components u_i^0. In this case G is an $m \times m$-matrix. Now, assume that the system haz a zero in z. According to the textbook, p. 66, it holds (since G is square) $\det G(z) = 0$. Then there exist a vector u^0 such that

$$G(z)u^0 = 0 \tag{5}$$

If we apply the control input signal $u(t) = u^0 e^{zt}$ the output becomes

$$y(t) = G(z)u^0 e^{zt} + \text{ transienter} = 0 + \text{ transients} \tag{6}$$

This is particularly striking in case the system has a zero in the right half plane: the output is zero although an exponentially increasing input.