Analysis and Design of Model Based Fault Diagnosis Systems for Large Scale Models
Exercises to the interactive session

Mattias Krysander and Erik Frisk

July 8, 2017

The exercises in the interactive session are based on a two-tank system. The system is presented in Section 1 and a model of the system is given in Section 2. Then Section 3 includes the exercises. To be able to solve the exercises the fault diagnosis toolbox at `faultdiagnosistoolbox.github.io` need to be downloaded and installed. The installation procedure is simple; download the zip-archive, unpack, and add the folder to your Matlab search path. Furthermore the following three Matlab-files are needed

- `skeleton.m` - Skeleton file for your solution
- `WaterTankModel.m` - Model definition
- `SimulateWaterTank.m` - Function to simulate the water tanks (do not need to be modified)

Some code is given in `skeleton.m` and you are supposed to complete the solutions in this file. In `WaterTankModel.m` the model of the water tank system is defined. The file `SimulateWaterTank.m` includes code to simulate the water tanks and do not need to be modified in the exercises.

1 The water tank process

The water tank system is illustrated in Figure 1, with an input signal u and four outputs $y_1, \ldots, y_4$. Water is pumped into the upper tank (tank 1) and the control signal u is controlled with respect to the water level in the upper tank. The tanks have level measurements y_1 and y_2, and flow measurements y_3 and y_4. Table 1 summarizes the faults that are considered in the system.

Table 1: Possible faults in the water tank system.

F_a	Actuator fault in the pump.
F_{h1}	Fault in sensor 1 measuring the water level h_1 in the upper tank, tank 1.
F_{h2}	Fault in sensor 2 measuring the water level h_2 in the lower tank, tank 2.
F_{f1}	Fault in sensor 3 measuring the flow W_1 between tank 1 and tank 2.
F_{l2}	Leakage between sensor 3 and tank 2.
F_{l3}	Leakage between tank 2 and sensor 4.
F_{c1}	Partial obstruction (clogging) in the pipe between tank 1 and tank 2.

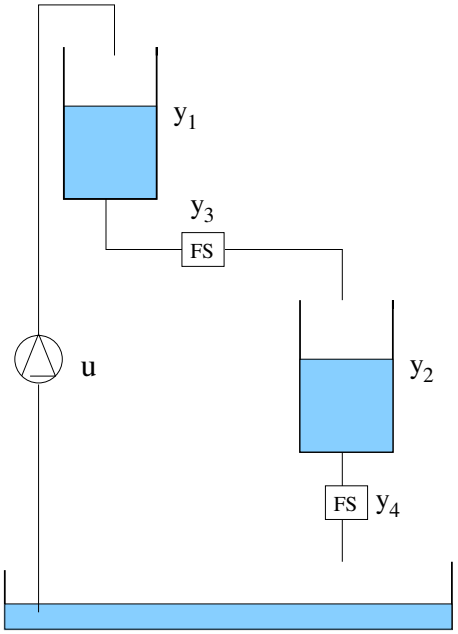

Figure 1: A schematic setup of two coupled water tanks. Each tank also has a level sensor (y_1 and y_2). The flow out of each tank is measured with a flow sensor (FS) (y_3 and y_4).

2 A model of the water tank system

A model for the water tank system is obtained from first principles as

$$A_1 \cdot dh_1 = W_{in} - W_1 \tag{1}$$
$$A_2 \cdot dh_2 = (1 - f_{l2})W_1 - W_2 \tag{2}$$
$$W_{in} = c_3 u + f_a \tag{3}$$
$$W_1 = (1 - f_{c1})c_1\sqrt{h_1} \tag{4}$$
$$W_2 = c_2\sqrt{h_2} \tag{5}$$
$$W_s = (1 - f_{l3})W_2 \tag{6}$$
$$y_1 = h_1 + f_{h1} \tag{7}$$
$$y_2 = h_2 + f_{h2} \tag{8}$$
$$y_3 = W_1 + f_{f1} \tag{9}$$
$$y_4 = W_s \tag{10}$$
$$dh_1 = \frac{d}{dt}h_1 \tag{11}$$
$$dh_2 = \frac{d}{dt}h_2 \tag{12}$$

where Table 2 describes the variables and parameters in the model. The different fault modes introduced in the system (see Table 1) is modeled as signals or deviations in constant parameters, e.g., F_a denotes fault mode and f_a denotes the modeled fault where $f_i = 0$ means that the corresponding fault is not present.

Table 2: Variables and parameters in the water tank model.

Parameters	
A_i	Bottom area of tank i.
c_i	Geometrical constants.
Unknown variables	
h_i	Water level in tank i.
W_{in}	Water flow into tank 1.
W_i	Water flow out from tank i.
W_s	Water flow through sensor y_4.

3 Exercises

Different versions of the model (1)-(12) will be used. In Exercises 1-4 we will assume that sensor y_1 is not available, in Exercise 5 no sensors is assumed to be installed, and in Exercise 6 that sensor y_3 is not available. All versions of the model are loaded by running the script `WaterTankModel.m`.

Exercise 1.

a) Run `WaterTankModel.m` that loads the different versions of the DAE-model (1)-(12). Here the model `model` should be used where sensor y_1 has been removed.

b) Use the command `Lint` to see if the model is well defined. What is the degree of redundancy of the model?

c) Plot the structure of the model with unknown variables, known variables, and faults using the command `PlotModel`.

d) Plot the structure of the unknown variables with the equivalence classes and faults using the command `PlotDM`.

e) Compute the structural isolability of the model with mixed and integral causality using the command `IsolabilityAnalysis`.

Exercise 2. Rewrite the model used in Exercise 1 in state-space form and compute the structural isolability of this model. The model without faults can be written as

$$\dot{h}_1 = \frac{1}{A_1}\left(c_3 u - c1\sqrt{h_1}\right) \tag{13}$$

$$\dot{h}_2 = \frac{1}{A_2}\left(c_1\sqrt{h_1} - c_2\sqrt{h_2}\right) \tag{14}$$

$$y_2 = h_2 \tag{15}$$

$$y_3 = c_1\sqrt{h_1} \tag{16}$$

$$y_4 = c_2\sqrt{h_2} \tag{17}$$

a) The state-space model for the fault free case is given in the script `WaterTankModel.m`. Extend this model with the faults and create a new DiagnosisModel object `model_statespace`.

b) Compare the structural isolability of the two analytically equivalent models and discuss the results.

Exercise 3. Continue with the model in Exercise 1.

a) Compute all MSO sets and all MTES sets in the model using `MSO` and `MTES` respectively. Compare the difference.

b) Check observability and index of the MSO sets with the commands `IsObservable` and `IsLowIndex` respectively.

c) Select a minimum cardinality set of low-index MSO sets or MTES sets to achieve maximum single fault isolation under integral causality by using `TestSelection`.

d) Compute the fault signature matrix of the selected tests with `FSM` and compute the isolability of these tests with the command `IsolabilityAnalysisArrs`.

Exercise 4.

a) Generate code for the selected tests. The tests should either be algebraic or sequential residual generators with integral causality.

As a help consider first the following example. The code for generating a sequential residual generator for the MSO set $\{(2), (5), (6), (8), (9), (11)\}$ is

```
mso1 = [2 5 6 8 9 11];
model.MSOCausalitySweep( mso1 )
red1 = mso1(5);
M01 = setdiff(mso1,red1);
Gamma1 = model.Matching( M01 );
model.SeqResGen( Gamma1, red1, 'ResGen1' );
```

Here the 5:th equation in the MSO set, i.e., equation (9), is selected as the redundant equation and the rest of the equations, in the code stored in `M01`, are used to solve for all unknown variables. To choose redundant equation consider the following properties:

i) The causality obtained by looking at the result of `MSOCausalitySweep`.

ii) Try to avoid selecting a differential constraint as the redundant equation.

The resulting code of the residual generator is stored in `ResGen2.m`.

b) Run and evaluate the response of the residuals for different fault scenarios using the provided simulation environment. Try at least the modes NF, F_{c1}, F_a, and F_{h2}, which are prepared in the code, and compare the response of the residuals with the fault signature matrix computed in Exercise 3d).

Exercise 5. Consider the model `model_nosens` in this exercise. It describes the water tank system without any sensors.

a) Check that the model is exactly determined.

b) Specify that h_1, h_2, W_{in}, W_1, and W_s are the possible sensor locations by using `PossibleSensorLocations`.

c) Compute all minimal sensor sets with maximum possible single fault isolability given that added sensors cannot fail with `SensorPlacementIsolability`.

d) Assume that all sensors have faults by using `SensorLocationsWithFaults`.

e) Compute all minimal sensor sets with maximum possible single fault isolability given sensors can fail. Compare the results with the solutions obtained in Exercise 5c).

Exercise 6. In this exercise we consider the water tank system without sensor y_3. The model is defined in `WaterTankModel.m` with the name `model_no_y3`.

a) Compute the structural isolability of the model with mixed and integral causality.

b) Design a diagnostic system with minimum number of sequential residual generators providing maximum possible isolability under integral causality.

c) To the diagnostic system developed in Exercies 6b) add a minimum number of sequential residual generators with arbitrary differential causality in order to obtain maximum single fault isolability.

d) Evaluate the diagnostic system by comparing the fault response in simulations with the fault signature matrix.