Nonlinear Control (FRTN05)

Computer Exercise 4

Last updated: Spring of 2011

Introduction

Goal

The goal of the computer exercise is to simulate parts of a JAS 39 Gripen (a military aircraft) control system, and to use the describing function method to analyze Pilot-Induced Oscillation (PIO).

Contents

The exercise is performed in Matlab and Simulink, either at the department or at any other computer that has Matlab with the Control System Toolbox, and Simulink. You will need some files for the exercise. These are available at the course homepage. Copy the files into the directory where you will run. To find out if you have the control system toolbox write `help control`. (We will use the commands `ss, tf, bode, nyquist, evalfr`. Use `help` to find out how they work.)

The first part, Section 1, contains small examples of how Simulink works. The following part of the exercise can be hard, so do not spend too much time on the introduction to Simulink! The Simulink introduction can be skipped if you already are familiar with Simulink. Section 2 contains the main problem; to analyze PIO of JAS.

1. Introduction to Simulink

Simulink is a simulation program based upon Matlab. There are several ways to define a model. One can work graphically and connect block-diagrams with predefined blocks. Alternatively one can give the mathematical description in forms of differential equations in an m-file (the format for programs written in the Matlab programming language). Matlab/Simulink supports both these representations as well as combinations. Furthermore one can use descriptions that include a hierarchy of connected subsystems.

To understand how models are described and simulated using block diagrams, it is best to run small examples on a computer. The rest of Section 1 shows some examples. If you are familiar with Simulink you can go directly to Section 2.

1.1 How to Start Simulink

Start Matlab. Then give the command `simulink` in Matlab. This gives a window with blocks as in Figure 1. Each block represents a library that contains several building blocks.

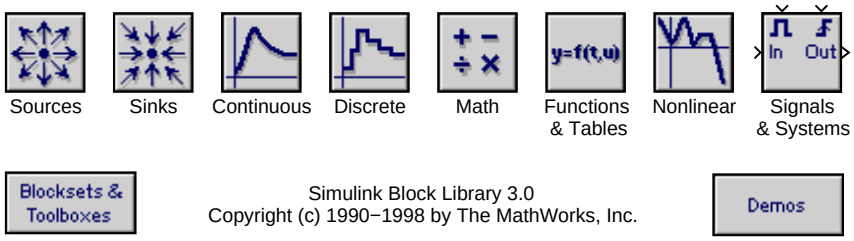


Figure 1 Available Simulink block diagram libraries

1.2 A Simple System

Click on `File` in the Simulink-window and choose `New->Model`. Click on the library `Continuous` and move a `Transfer Fcn` to the new window called "Untitled". Do the same with `Sources->Step Fcn` and `Sinks->Scope`. Draw arrows (left mouse button) and connect the ports on the block. You should now have a block diagram as in Figure 2.

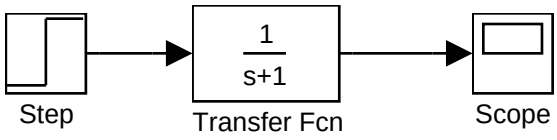


Figure 2 A simple Simulink system

Choose `Simulation->Parameters` in the window called "Untitled". Set `Stop time` to 5. Open a window for the `Scope` by double clicking on it. Start a simulation by `Simulation->Start` (or by pressing Ctrl-t in the window called "Untitled').

How to Change a System To change the system to

$$\frac{1}{s^2 + 0.5s + 2}$$

you double-click on the block `Transfer Fcn` and change `Denominator` to [1 0.5 2]. Simulate the new system (`Simulation->Start` or Ctrl-t). Change parameters in the `Simulation` menu and the scales in the block `Scope` until you are satisfied.

How to Change an Input Signal To change the input signal, start with removing the block `Step Fcn` by clicking on it and delete it by pressing Delete (or using `Edit->Cut` or pressing Ctrl-x). Replace it by a `Sources->Signal Generator` block. Double-click on `Signal Generator` and select a wave form, amplitude and frequency. Also change `Simulation->Parameters->Stop Time` to 99999 and press `Simulation->Start`. This gives an "infinite" simulation that can be stopped by pressing `Simulation->Stop` (or Ctrl-t). Can the amplitude of the input signal be changed during simulation? Also try to change the block `Transfer Fcn` during simulation.

How to Use Matlab Variables in Blocks Note that variables defined in the Matlab environment can be used in Simulink. Define numerator and denominator by writing the following in the Matlab window.

```
num=[1 1]
den=[1 2 3 4]
```

Change `Transfer Fcn->Numerator` to `num` and `Transfer Fcn->Denominator` to `den`.

How to Save Results to Matlab Variables To save input and output, move two copies of the block `Sinks->To Workspace`. Connect these with the input and output to the block `Transfer Fcn`. Get a `Sources->Clock` and connect it to a `Sinks->To Workspace`. Double click on the "Workspace blocks" to be able to change the variable names to `u`,`y`, and `t` respectively. Also change `Save format` to the value `Array`. The window should look something like Fig. 3.

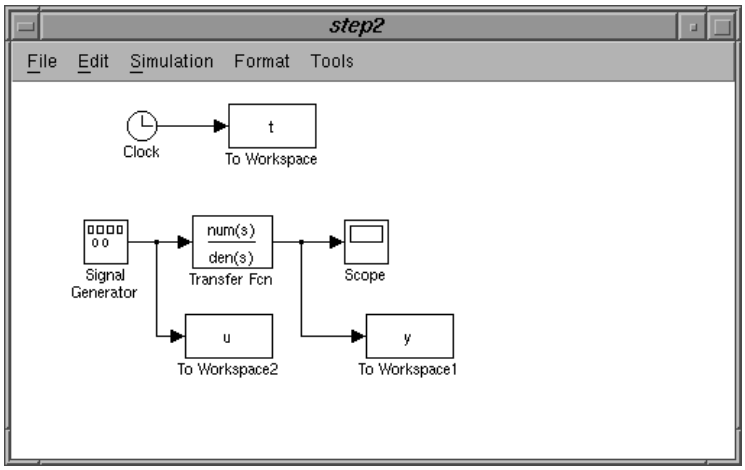


Figure 3 How to save results to Matlab variables in Simulink

How to Use Simulation Results in Matlab Calculations Let the input signal be a sinusoidal with frequency 0.1 rad/s and amplitude 1. Do a simulation that is long enough for the output to become stationary. Compute

```
n=length(y)
max(y(n/2:n))
```

and compare this with the theoretical value $|G(0.1i)|$.

```
>>g=tf(num,den)

Transfer function:
        s + 1
--------------------
s^3 + 2 s^2 + 3 s + 4

>> abs(evalfr(g,0.1*i))
ans =
    0.2518
```

How to Save Systems Use `File-Save As` or `File->Save`.

1.3 A Flow System

Consider a simple tank as in the basic control course

$$\begin{aligned} \dot{h} &= \frac{1}{A}(u-q) \\ q &= a\sqrt{2gh}. \end{aligned}$$

This can be implemented in Simulink as in Figure 4. The function $f(u)$

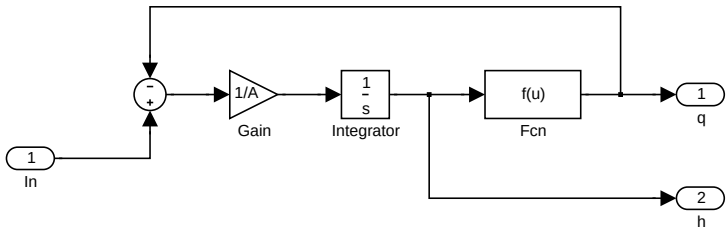


Figure 4 A tank system

has the value `a*sqrt(2*g*u[1])`. The summation block has been given two inputs with different signs by assigning the string "-+" to `Sum->List of Signs`. The summation and the `Gain` blocks are found in the `Math` library and the `Fcn` block is found in the `Functions & Tables` library. The small ellipses, that are contained in the `Signals & Systems` library, tell Simulink what should be considered inputs and outputs to this (sub)system. The block titles can be changed by clicking on them. Mark the entire system by holding the left mouse bottom pressed and drawing a rectangle around it. Then choose `Edit->Create Subsystem`. The result is that the system is represented by one block. Use `Edit->Copy` to create the following double-tank system. Use the commands `trim` and `linmod` to find a linearized model

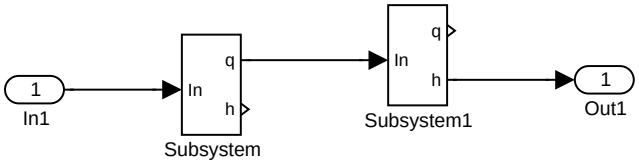


Figure 5 Two tanks and some connections

of the double tank around $h_1^0 = h_2^0 = 0.1$. Use the parameters $A_1 = A_2 = 2.7 \times 10^{-3}$, $a_1 = a_2 = 7.0 \times 10^{-6}$, $g = 9.8$. Plot the Nyquist curve using the command `nyquist`.

```
>> A=2.7e-3;a=7e-6;g=9.8;
>> [x0,u0,y0]=trim('flow',[0.1 0.1]',[],0.1)
 Warning: Output port 2 of block 'twotank/Subsystem' is not connected.
 Warning: Output port 1 of block 'twotank/Subsystem1' is not connected.
```

```
x0 =
    0.1000
    0.1000
u0 =
   9.7995e-06
y0 =
    0.1000
>> [aa,bb,cc,dd]=linmod('flow',x0,u0);
>> sys=ss(aa,bb,cc,dd);
>> bode(sys)
```

Alternative: Linearization in Simulink;
By right-clicking on a signal connector in a Simulink model you can add "Linearization points" (inputs and/or outputs). Use this for the two water tanks.
Start a "Control and Estimation Tool Manager" by
`Tools -> Control Design ->Linear analysis ...`, see Fig. 6. Here you can set the desired operating points, export linearized model to Workspace (Model-> Export to Workspace) and much more.

Repeat the linearization of the system at the same equilibrium point as above.

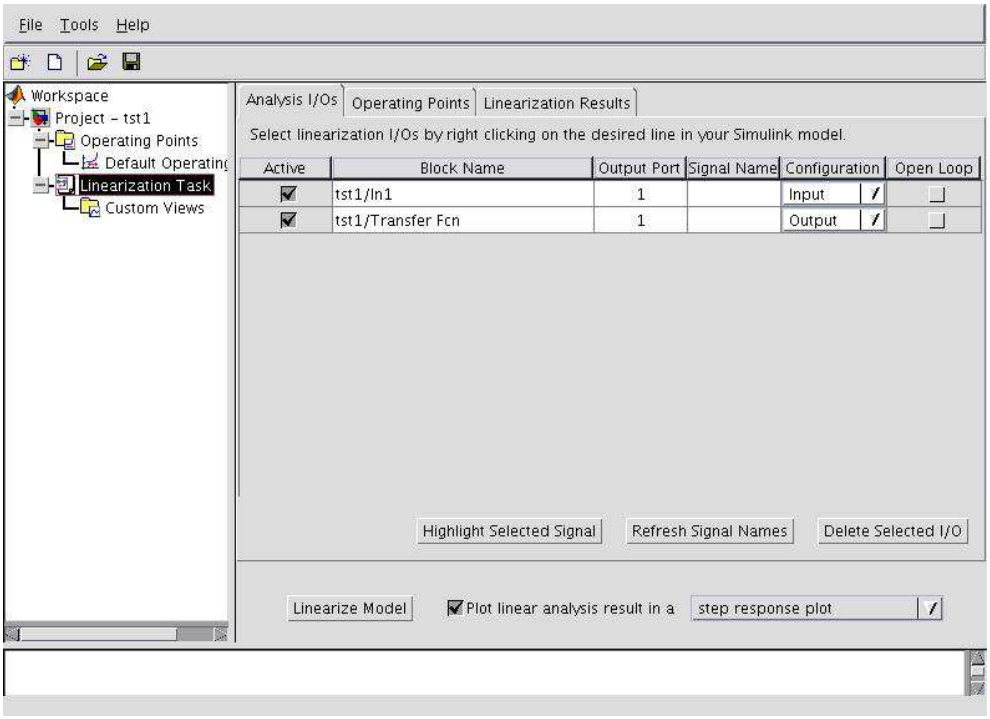


Figure 6 View of "Control and Estimation Tool Manager" in Matlab/Simulink.

2. Example – JAS 39 Gripen

2.1 Background–Model

We will study simulation and control of the pitch dynamics of a JAS 39 Gripen. The dynamics of an airplane is highly nonlinear. The control system uses gain scheduling on speed and altitude to compensate for parts of the nonlinearities. Linear models have been obtained for approximately 50 different operating conditions. The models are given in state-space form and are the result of extensive wind tunnel experiments and calculations at SAAB. A controller is designed for each linear model, and a switching mechanism is used to switch between the different controllers. Many of the parameters in the models vary considerably within normal operation of the aircraft. Two of the extreme cases are "heavily loaded aircraft at low speed" and "unloaded aircraft at high speed". The aircraft is also very sensitive when the speed is close to Mach 1 and the dynamics change from unstable to stable. The Mach number gives the speed of the aircraft, and is given by $M = v/a$, where a is the velocity of sound.

We will work with a linear model with five states for normal load at $M = 0.6$ and altitude 1 km. The model is taken from a master thesis at Chalmers, see [1]. The state space model is defined in the file `jasdata.m`. Write

```
>>jasdata
```

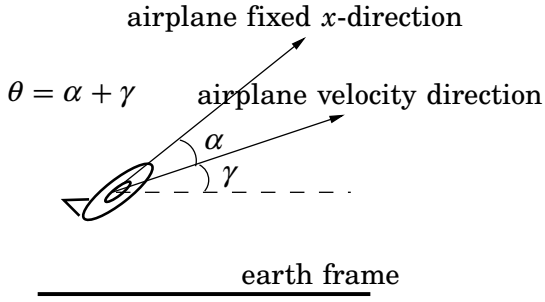
The model of the aircraft is given by

$$\dot{x} = Ax + Bu,$$

where the state vector x consists of the seven variables

α	angle of attack
q	pitch rate
θ	pitch angle
δ_e	elevator angle
δ_s	spoiler angle
x_e	internal elevator state
x_s	internal spoiler state

The inputs in u are given by u_e, that is the elevator command, and u_s, that is the spoiler command.



The rudder servos are controlled in inner loops that give the rudder dynamics

$$\delta = \frac{1}{(0.05s+1)(0.008s+1)}u.$$

This means that each rudder has two states, the rudder angle (δ_e and δ_s) and the internal rudder state (x_e and x_s).

The plane is controlled by linear state feedback $u = -Lx$. The feedback loop stabilizes the states q and α. The pitch angle θ is not stabilized. The control of this mode is left for the pilot (more about this later). The L-vector has been designed using linear quadratic control theory. A reduced model is used in the design, with the fast rudder dynamics neglected (corresponding to the pole in $s = -1/0.008$). For this reason, the internal rudder states are not used in the feedback loop. In reality there are no measurements of rudder angles δ_e and δ_s. These are instead estimated with a Kalman filter. The total control signal is given by

$$u(t) = -Lx(t) + K_f u^f_{pilot}(t),$$

where K_f is a constant which gives the correct closed loop steady state gain, and u^f_{pilot} is the filtered pilot command signal

$$u^f_{pilot} = \frac{1}{T_f s + 1}u_{pilot}$$

To see the Simulink model of the aircraft write

```
>>planemodel
```

This gives the block diagram in Figure 7.

2.2 The Pilot – Normal Function

The pilot is modeled as a PD-controller with time delay of $0.3s$. The pilot transfer function is given by

$$u_{pilot} = K_p \frac{1 + T_d s}{1 + T_d/Ns} e^{-0.3s} \cdot (\theta_{ref} - \theta),$$

where $K_p = 0.2$, $T_d = 0.5$, and $N = 10$. Modelling a pilot as a PD-controller can be motivated by that it is natural that a pilot gives control commands

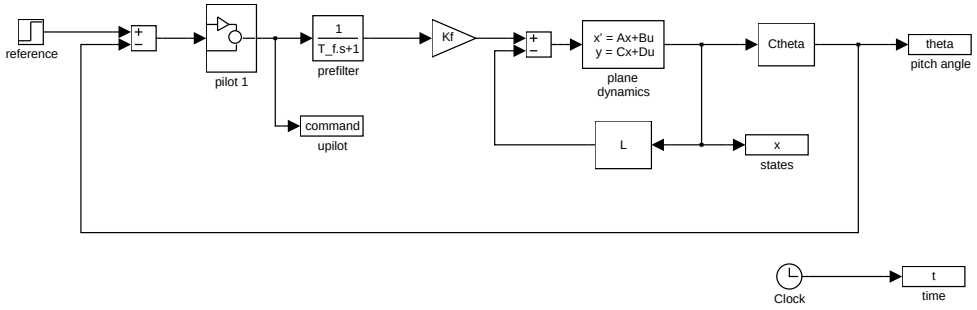


Figure 7 Simulink model of plane and pilot.

proportional to the control error, but is a little more careful when the pitch angle is changing rapidly.

Assignment 1: *Regard the state space model of the aircraft (given by the A-, B-, C-, and D-matrices). Find the poles of the system (the eigenvalues of the A-matrix). Is the open loop system unstable? How many inputs and outputs does the model have? Go through the interconnections in the A-matrix to see if you can find "sub-systems" and how these relate to the open-loop poles (use the state vector information on p.6).*

Choose a nominal design for the state feedback by typing

```
>>design1
```

Look at the L-matrix. Are the correct states used in the feedback loop? What are the eigenvalues of $A - BL$*? Explain the pole close to the origin (i.e., which state does it correspond to). Why has this pole placement been chosen?*

Simulate the closed-loop system including the pilot and check that it works properly. Check that the rudder angles are within ± 0.5 *radians (use* `plot(t,x(:,[4 5]))`*). Why is it important that the rudder angles are not too large?*

2.3 Pilot Induced Oscillations (PIO)

The pilot model used in Assignment 1 can be seen as a pilot who works rationally and always does the right thing. In an emergency situation the pilot may panic, and try to compensate the error with maximal command signals. Imagine yourself in any balancing act. When you are in control, you can keep the balance using very small motions, but as soon as you are a little out of control, your movements tends to be very large and eventually you will fall. This is typical for systems with relatively slow dynamics, which cannot react quick enough to the control signals. If you start to fall and try to get back up again, because of the slow dynamics you will use too much efforts to come up so that once your up again you cannot stop, but will immediately start falling in the other direction.

The phenomena got considerable attention after the crash in Stockholm in 1993, and was given the name pilot induced oscillations, PIO. In the lab we will make a simplified analysis using a relay model of the pilot in PIO mode. The pilot gives maximal joystick commands based on the sign of θ.

Assignment 2: *A "relay pilot" can be found in the Simulink pilot model library. Write*

```
>>pilotlib
```

Plot the Nyquist curve of the linear system from u_{pilot} to θ. This can be done by deleting the feedback path from θ, connecting input and output connections at appropriate places, saving the system to a new file and using the `linmod` *and* `nyquist` *commands. Use* `help` *to see what the commands do. The describing function for a relay is*

$$N(A) = \frac{4D}{\pi A},$$

What is the amplitude D of the "relay pilot"? Use describing function analysis to estimate the amplitude and frequency of a possible limit cycle by reading out the intersection between $-1/N(A)$ and the nyquist plot. (You can easily zoom and mark points on the Nyquist plot to read out e.g., frequencies).

Change pilot in the plane model by deleting the first pilot and instead choose the "relay pilot". Simulate the system. How good is the prediction of the limit cycle with the simulation results?

Assignment 3: *As you can see, the amplitude is relatively moderate. This is because the flight condition is high speed and high altitude. Let us anyway discuss two possible ways to reduce PIO:*

- *Use the command* `design2` *to change L and K_f to a faster design. Is the PIO amplitude decreased?*
- *Use the command* `design3` *to make the filter faster by reducing the filter constant to $T_f = 0.03$. Is the PIO amplitude decreased?*

Compare the Nyquist curves from u_{pilot} to θ for the different designs (design 1-3). In the light of the describing function method; which design reduces PIO amplitude? Can you find some drawback with this method? Hint: Simulate the system with the normal pilot from assignment 1 using the different designs.

PhD Assignment: *There are no rate limitations on rudders in the model. Rate limitations were also part of the source of control problems on the JAS. Introduce rate limitations, for example as in the article by Rundquist et al., and investigate what happens to the limit cycle. Does it become unstable? Try to understand the idea of the patented nonlinear filter.*

References

[1] Lars Axelsson, Reglerstudier av Back-up regulator för JAS 39 Gripen, examensarbete CTH/Saab flygdivisionen, 1992.
[2] Rundquist et al., Rate Limiters with Phase Compensation in JAS 39 Gripen, European Control Conference 1997.