Physiological Models and Computations

Exercises

Department of Automatic Control
Lund University, Faculty of Engineering

2014

0. Repetition of Linear Algebra, Differential equations and MATLAB

Solve the following exercises by hand. If you are unsure about how to solve the exercises, please go back to your Linear algebra and Analysis books and review the material needed.

0.1

a. Find the solution to the differential equation below when $x(0) = 1$,

$$\frac{dx}{dt} = c$$

b. Find the solution to the differential equation below when $x(0) = 1$,

$$\frac{dx}{dt} = cx$$

c. Find the solution to the differential equation below when $x(0) = 1$ and $x \neq 0$ for any t,

$$\frac{dx}{dt} = 2tx^2$$

d. Rewrite the differential equation into a system of first order differential equations.

$$\begin{aligned} \ddot{y} + 7\dot{y} - 3y &= 0 \\ y(0) &= 0 \\ \dot{y}(0) &= 1 \end{aligned}$$

Solve the following exercises using `MATLAB`. These exercises are inspired by or fully extracted from *EDA017: Föreläsningsanteckningar,* *OCTAVE/MATLAB* by Christian Söderberg.

0.2 In `MATLAB`, find the commands necessary to derive the following results for matrices A and B

$$A = \begin{pmatrix} 2 & 0 & 0 \\ 0 & 3 & 4 \\ 0 & 4 & 9 \end{pmatrix}, \quad B = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$$

a. calculate $A \cdot B$ and $B^T \cdot A$. What about $B \cdot A$?

b. Give the eigenvalues and eigenvectors of A.

c. Give the transpose and the determinant of A.

d. Give the inverse of A and review how the inverse is derived by hand for a 2-by-2 matrix.

0.3

a. Plot $y(x) = e^{-x/2}\cos(2\pi x)$ when $-6 \le x \le 3$ by using the function handle to create an anonymous function. Give your plot a title as well as labels on the axes. Useful commands: `fplot`, `xlabel`, `ylabel`, `title`.

b. Modify your code such that you only show values $-4.5 \le x \le -1$ and $-10 \le y \le 10$. Useful command: `axis`.

c. Integrate the function for $-4.5 \le x \le -1$.
Useful commands: `integral`, `quad`.

d. Find the solution to $f(x) = 0$ when $f(x) = x^3 + 2x - 1$. Comment on the answer. Useful command: `fsolve`.

0.4 Write a function which for every matrix A gives you the sum of the diagonal elements of that matrix. Useful commands: `diag`, `sum` and `size`.

0.5 Solve the differential equation

$$\begin{aligned} \ddot{y} + 7\dot{y} - 3y &= 0 \\ y(0) &= 0 \\ \dot{y}(0) &= 1 \end{aligned}$$

in the interval $0 \le t \le 5$ by using `MATLAB`s solver `ode45`.

Solve the following exercises using `SIMULINK` in `MATLAB`. These exercises are taken from *Exercises in* `MATLAB`*/*`Simulink`*, Signals and Systems* by Thomas Munther.

0.6 Investigate the bacterial growth in a jam pot. Assume that the number of born bacteria is increasing proportional to the existing number of bacteria x and the number dying is proportional to the existing number in square. This gives the following differential equation

$$\frac{dx}{dt} = bx - px^2$$

where $b = 1$ [1/hour] is the birth rate constant and
$p = 0.5$ [1/(bacteria·hour)] is the death rate constant.
Assume $x(0) = 100$ [bacteria]. Use `SIMULINK` to show what the solution to the differential equation looks like.

0.7 Some physiological systems are better described in discrete time which gives rise to difference equations. Show the behavior of y in the two following difference equations

a.

$$y_t = -0.5 \cdot y_{t-1} + x_t$$

b.

$$y_t = 0.5 \cdot y_{t-1} + x_t$$

where x is the input signal to the system, in shape of a step starting in $t = 0$ with amplitude 1 and $y_{-1} = 1$. y_t is the value of y in time step t.

0.8 Get familiar with some of the blocks that will be used in the course; `From Workspace`, `To Workspace`, `Constant`, `Scope`, `Step` and `Sine Wave`. Look at how `Step` and `Sine Wave` can be altered and how they look by the use of a `Scope`. Try to save the result to the workspace by `To Workspace` and plot it. Save the plots as an .eps-file. Create a document, write something nice about the plot, add the plot with a figure text, save the document as a .pdf-file.

Solutions to Chapter 0. Repetition of Linear Algebra, Differential equations and MATLAB

Solve the following exercises by hand. If you are unsure about how to solve the exercises, please go back to your Linear algebra and Analysis books and review the material needed.

0.1 **a.** $x(t) = ct + 1$.

b. $x(t) = e^{ct}$.

c. The differential equation is separable. Rewrite it as

$$\frac{1}{x^2}dx = 2tdt$$

$$\frac{1}{x^2}dx = 2tdt$$

Integrating on both sides gives

$$-1/x = t^2 + c$$

where c is a constant. Hence, $x(t) = -1/(c + t^2)$. $x(0) = -\frac{1}{c} = 1 \rightarrow c = -1$. The solution to the differential equation is therefore, $x(t) = \frac{1}{1-t^2}$.

d. Introduce $y_1(t) = y(t)$ and $y_2(t) = \dot{y}(t)$ in order to rewrite the initial second-order differential equation into two first-order differential equations as follows

$$\dot{y}_1 = y_2 \tag{0.1}$$
$$\dot{y}_2 = 3y_1 - 7y_2 \tag{0.2}$$

The initial conditions for $y_1(t)$ and $y_2(t)$ are

$$y_1(0) = y(0) = 0$$
$$y_2(0) = \dot{y}(0) = 1$$

(0.1) and (0.2) can be written together on matrix form as follows

$$\begin{pmatrix} \dot{y}_1 \\ \dot{y}_2 \end{pmatrix} = \begin{pmatrix} y_2 \\ 3y_1 - 7y_2 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 3 & -7 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \end{pmatrix}$$

Define $\mathbf{v} = \begin{pmatrix} y_1 \\ y_2 \end{pmatrix}$. Then, define f as the following function

$$f(t,\mathbf{v}) = f(t, \begin{pmatrix} y_1 \\ y_2 \end{pmatrix}) = \begin{pmatrix} \dot{y}_1 \\ \dot{y}_2 \end{pmatrix} = \begin{pmatrix} y_2 \\ 3y_1 - 7y_2 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 3 & -7 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \end{pmatrix}.$$

Solve the following exercises using `MATLAB`. These exercises are inspired by or fully extracted from *EDA017: Föreläsningsanteckningar,* *`OCTAVE/MATLAB`* by Christian Söderberg.

0.2 Use the help function and MathWorks webpage.

0.3 a. Create an anonymous function using the function handle. This function is only saved in your workspace until you close MATLAB (or clear you workspace by the `clear all` command). In case you would like to save your function as a file in your current folder (from where you can reach it at another time), use a function m-file (go to new → function).

```
y = @(x) exp(-x/2)*cos(2*pi*x);

figure
fplot(y,[-6 3])
title('My fancy plot')
xlabel('x')
ylabel('y')
```

`figure` is a command which is useful when you want to create several plots in the same script. Use the `help`-command whenever you need information about one of `MATLAB`s buildt-in functions. In this case you would write `help figure` in the command window and the description of the function should appear.

b.

```
axis([-4.5 -1 -10 10])
```

c.

```
% Rewrite y to be accepted by quad/integral (read in the
% description of quad/integral to understand why).
y = @(x) exp(-x/2).*cos(2*pi*x);

integral(y,-4.5,-1)
% or
quad(y,-4.5,-1)
```

d.

```
f = @(x) x^3+2*x-1; solution = fsolve(f,0)
```

The answer is 0.4534. Write `format long` in the command window (then use the `fsolve` command) to get more decimals in the answer. Due to it being numerically calculated $f(0.4534)$ is approximately zero.

0.4 Go to new → function. A file with a function-shell will appear. The function shell looks like:

```
function [ output_args ] = untitled( input_args )
%UNTITLED Summary of this function goes here
%   Detailed explanation goes here


end
```

Replace `untitled` with the name of your function, `input_args` with the input your function needs and `output_args` with the output your function will give. Between the `function`-row and the `end` you should write the code for the function.

For the particular function of this exercise, it will look as follows

```
function sumOfDiag = sumOfDiagonal(A)
[n,m] = size(A);

if n ≠ m
    error('A is not a square matrix')
end

sumOfDiag = sum(diag(A));
end
```

Where $\neq$ is written as ~= in MATLAB. Save your function as an m-file in your current folder, by the name of your function. In this case it would be "sumOfDiagonal.m". Now you can use your function directly from the command window or from a script which is saved in the same folder as your function.

To create a matrix in MATLAB use the following principle

```
my_matrix = [1 2; 3 4];
```

`[` and `]` begins and ends the matrix. Elements are separated by space (or comma) and rows are separated by `;`. The resulting matrix is

$$\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$$

0.5 Introduce $y_1(t) = y(t)$ and $y_2(t) = \dot{y}(t)$ in order to rewrite the initial second-order differential equation into two first-order differential equations as follows

$$\dot{y}_1 = y_2 \tag{0.3}$$
$$\dot{y}_2 = 3y_1 - 7y_2 \tag{0.4}$$

The initial conditions for $y_1(t)$ and $y_2(t)$ are

$$y_1(0) = y(0) = 0$$
$$y_2(0) = \dot{y}(0) = 1$$

(0.1) and (0.2) can be written together on matrix form as follows

$$\begin{pmatrix} \dot{y}_1 \\ \dot{y}_2 \end{pmatrix} = \begin{pmatrix} y_2 \\ 3y_1 - 7y_2 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 3 & -7 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \end{pmatrix}$$

Define $\mathbf{v} = \begin{pmatrix} y_1 \\ y_2 \end{pmatrix}$. Then, define f as the following function

$$f(t, \mathbf{v}) = f(t, \begin{pmatrix} y_1 \\ y_2 \end{pmatrix}) = \begin{pmatrix} \dot{y}_1 \\ \dot{y}_2 \end{pmatrix} = \begin{pmatrix} y_2 \\ 3y_1 - 7y_2 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 3 & -7 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \end{pmatrix}$$

In MATLAB this can be written as

```
f = @(t,v) [v(2); 3*v(1)-7*v(2)];
```

Or by matrix multiplication

```
f = @(t,v) [0 1; 3 -7]*v;
```

To solve the differential equation write the following code

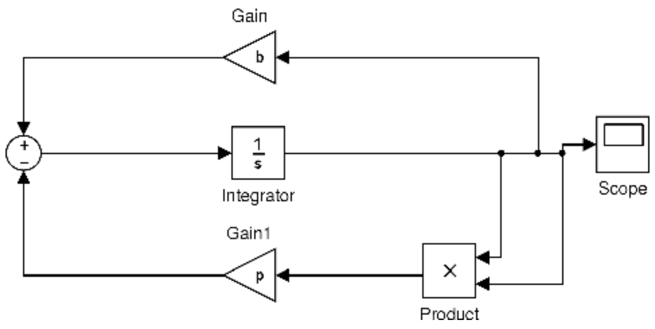
```
[t_ode V] = ode45(f,[0 5],[0 1]);
```

The first input to `ode45` is the right part of the differential equation, the second input is the time span of the solution while the third is the initial condition of the differential equation. `V` is a matrix with two columns, the first column corresponds to $y_1(t) = y(t)$ and the second column corresponds to $y_2(t) = \dot{y}(t)$. `t_ode` is the times between 0 and 5 at which `ode45` has calculated y_1 and y_2. Use the following code to plot $y(t)$ over $0 \leq t \leq 5$

```
plot(t_ode,V(:,1))
```

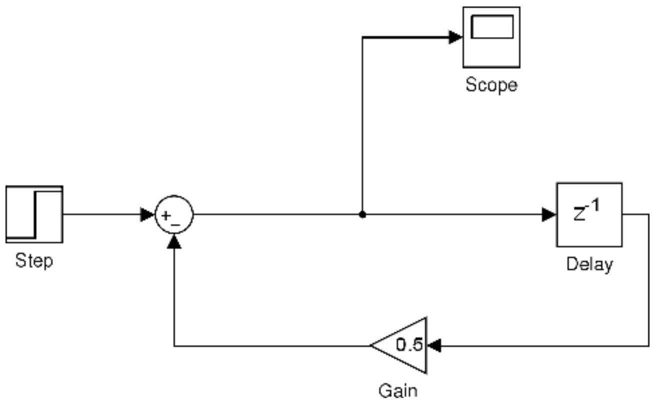
Solve the following exercises using SIMULINK in MATLAB. These exercises are taken from *Exercises in MATLAB/Simulink, Signals and Systems* by Thomas Munther.

0.6 Start SIMULINK by writing `simulink` in the MATLAB command window. This makes the SIMULINK Library Browser window pop up. Go to File → New → Model. In this window you can start to create your SIMULINK model. Use the Library Browser to find appropriate blocks and drag them into the model sheet. You can connect two blocks by their connection spots.

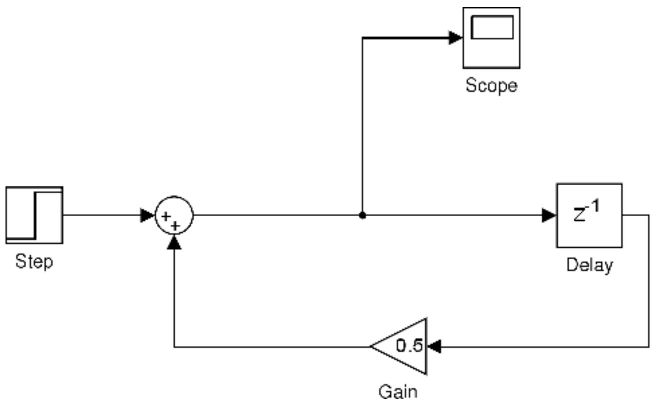


p and b can be defined in the current workspace. Go to display → blocks and check "Sorted Execution Order". This will numerate the blocks in the order in which they are first activated.

0.7 a. Before running the simulation go to Simulation → Configuration Parameters. In Solver Options choose Fixed-step and Solver → Discrete. Set the sample time in each block to 1 [sec].



b. The only difference from the previous model is that the minus sign in the sum-block is changed to a plus sign.



0.8 Just play around.