UPPSALA UNIVERSITET
AVDELNINGEN FÖR SYSTEMTEKNIK
BC 0801, last rev 15 september 2014

Computer exercise 1 for the course:
Empirical Modelling

Introduction to empirical modelling/system identification

The main purpose of these computer exercises is to help you to learn system identification and to get you well prepared for the project work. Please be well prepared before the exercises and try to use the allocated time as effective as possible.

Preparation exercises:

1. Read the lab instruction carefully.
2. Do the preparations in Section 2.2
3. Read Ch 11 in the text book with emphasis on Ch 11.1 and the discussion around Figure 11.7.
4. Read about properties of ARX-modelling (lecture notes)
5. Read the text on Linear regression (used in Section 4 in the lab)

Name	(Assistant's comments)
Program Year of reg.	
Date	

Innehåll

1 Introduction

The aim of this computer laboratory is to give a first introduction to System identification. Non-parametric identification as well as the least squares method are illustrated. Also, some examples of linear regression are covered.

In this exercise you will use small pre-prepared m-files to solve the tasks. whereas in the next lab exercise you will use a graphical user interface which minimize the need for own coding (but not the understanding of the methods!). In the project work, you can in most of the tasks chose if you want to work from the graphical user interface or from the command window (like in this lab).
It is recommended that after this exercise, you should at least be familiar with the following Matlab commands

```
randn
filter
cra
spa
arx
polydata
polyfit
\ (left matrix divide)
```

Furthermore, you should always check if a function already is available for solving a specific task instead of trying to write your own function. For example, functions exist for calculating mean, variance and standard deviation:

```
mean(X) - calculates the mean of the elements in the vector X
var(X) - calculates the variance of the elements in the vector X
std(X) - calculates the standard deviation of the elements in the vector X
```

1.1 MATLAB startup

1. Log in with your UpUnet-S user name and password C.
2. Start MATLAB from the program-menu (neglect any error message).
3. The predefined macros and other files you will work with is under the catalogue G:/Program/Systemteknik/Enviroprocess. In order to run the macros you will have to change the MATLAB working directory to the one mentioned above. This is easiest done by pressing the little button with three dots, situated next to the "Current Directory"-field in the tool bar of MATLAB, and then search under My Computer → G:/ etc.
4. If you want to save (files, data etc.) this has to be done under H:/, which is your own students directory. To be able to execute these files while still in the G:/... working directory you have to add a *search path* to the files. This is easiest done under **File** and **Set Path**. Press the button **Add Folder**, and search your way to H:/<*directory*>, where *directory* is the path to the directory under H:/ where you saved your files. You do not have to save the new search path (*i.e.* you cannot). Just press Close and No on the question if you want to "save the current path". The new search path will now be stored in MATLABs working memory, which means it will be active for this session only and has to be set again if you restart MATLAB.

5. If you want to *modify* one of the predefined files under G:/ you have to save a copy of this file under your own account (H:/), preferably with a new file and function name to avoid mix-ups. If you have set the path to the directory as described above, this file can then be run in the same manner as the original file.

2 Non parametric methods

A typical example of a non-parametric method is to estimate a Bode diagram by frequency analysis. The input signal is then a sinusoid and the amplitude and phase shift of the output signal is measured. The theory of frequency analysis was presented in the Basic Automatic Control course (Reglerteknik) and is not illustrated here (please see the text book on p 41 if you need a repetition of frequency analysis).

We will here illustrate how the impulse response of a system can be found and how spectral analyses can be used. We will, however, first have a short look of the most basic method of all, namely the transient analysis.

2.1 Systems used to generate the data

The basic problem in system identification is to find the system dynamics (in parametric or non parametric form) from measurements of y and u. Of course, in practice the true system is unknown (otherwise it would be no need for system identification) and only the measurements are available. We are here using a system for (i) generating data and (ii) evaluate how well various system identification methods work.

We will consider estimation from data given by the following system[1]

$$\mathcal{S}_1: \quad A(q)y(t) \quad = \quad B(q)u(t) + e(t)$$

The disturbance $e(t)$ is white Gaussian noise, independent of the input $u(t)$, and of zero mean and variance λ.

The system can also be written as

$$y(t) \quad = \quad \frac{B(q)}{A(q)}u(t) + \frac{1}{A(q)}e(t)$$

Note that when simulating the system, the white noise $e(t)$ should be filtered by $\frac{1}{A(q)}$ before it is added to the other term.

The system parameters are given by

$$A(q) = 1 + a_1 q^{-1} = 1 - 0.8q^{-1} \qquad B(q) = b_1 q^{-1} = 1.0q^{-1}$$

With the parameters above, the system becomes

$$y(t) = 0.8y(t-1) + 1.0u(t-1) + e(t) \tag{1}$$

The system has a time delay of one (sample) since the right hand side starts with $u(t-1)$.

[1] In some texts the polynomials are written as $A(q^{-1})$ etc.

2.2 Preparation exercises

2.2.1 Static gain

The static gain, denoted y_s below, could easily be calculated from (1). Note that if u is a step with amplitude 1, the static gain is simply the steady state value of y. During steady state, with the input u(t)=1 and $e(t) = 0$ (we only consider the deterministic part of the system) we have $y_s = y(t) = y(t-1)$ which gives

$$y_s = 0.8y_s + 1.0 \times 1$$

In general we can calculate the static gain[2] as

$$y_s = \frac{B(1)}{A(1)}$$

Calculate the static gain for the system (you may also want to test that the two ways above to calculate the static gain give the same result)

Answer: _____

_____

How could the static gain be found in a bode plot?

Answer: _____

_____

2.2.2 Impulse response

The impulse response of a linear discrete time system is given by

$$y(t) = \sum_{k=0}^{\infty} g(k)u(t-k) = \sum_{k=0}^{\infty} g(k)q^{-k}u(t)$$

In order to calculate the impulse response of the system (1) we form

$$G(q) = \frac{B(q)}{A(q)} = \frac{b_1 q^{-1}}{1 + aq^{-1}} = b_1 q^{-1}\frac{1}{1 + aq^{-1}}$$

Using the power series expansion $\frac{1}{1+c} = \sum_{j=0}^{\infty}(-c)^j$ with $c = a * q^{-1}$ we then have

$$\begin{aligned} G(q) &= b_1 q^{-1}[(-aq^{-1})^0 + (-aq^{-1})^1 + (-aq^{-1})^2 + (-aq^{-1})^3 + ...] \\ &= 0 + b_1 q^{-1} - b_1 a q^{-2} + b_1 a^2 q^{-3} ... \end{aligned}$$

Calculate $\{g(k)\}_{k=0}^{2}$ for the system, that is for the case $b_1 = 1$, and $a_1 = -0.8$. Hint: The coefficients are given in the right hand above.

Answer: _____

_____

[2]Note that for a *continuous* time system with transfer function $G(s)$, the static gain is given by $G(0)$, see the basic control course!

Note that the step response (the input jumps from 0 to 1 at $t = 0$) is easy to calculate if the impulse response is known:

$$\begin{aligned} y(0) &= g(0) \\ y(1) &= g(0) + g(1) \\ y(2) &= g(0) + g(1) + g(2) \\ \vdots &= \vdots \end{aligned}$$

2.3 Transient analysis

In transient analysis, the input signal u is a step or impulse. If the system was noise free this would give enough information to determine a model exactly. However, even for noisy data, transient analysis is very useful. It can, for example give a rough estimate of the time delay (dödtid"), dominating time constant, and static gain
We begin with illustrating transient analysis. The system $\mathcal{S}_1$ with the input $u(t)$ being 0 for $t \leq 9$ and being 1 for $10 \leq t \leq 100$ is simulated and the response is plotted. The task can be compiled using the Matlab code below. The code is available as file `EM1a`.

```
%file EM1a.m  (1 is the number one)
%Computer Excercise 1
%Empirical Modelling
%BC 080114
%last rev
%
a=-0.8,b=1,A=[1 a],B=[0 b],
s=input('Give variance lambda of the noise: ');
e=randn(100,1)*sqrt(s);  %If a signal is multiplied with k the variance is incr
u=[zeros(9,1);ones(91,1)];
yNoNoise=filter(B,A,u);
Noise=filter(1,A,e);
y=yNoNoise+Noise; %Note how y is constructed!
%
subplot 211
plot(u)
axis([0 100 -1 2])
grid
title('Step responses')
title('Input signal u')
subplot 212
plot([yNoNoise,y])
title('Noisfree output (blue) and Measured output') %For Matlab 7
%legend('Noisfree output Measured output','Location','Best' ) %For Matlab 7
%legend('Noisfree output','Measured output',0 ) %For Matlab 6
grid
```

Try the above function for at least two different noise variances (for example 0.1 and 1) and notice how an increase in the noise level deteriorates the possibility to see the system dynamics (like the static gain). If you write `figure` after you have run the macro, a new plot will be used the next time you run the macro (this makes comparisons easier).

Compare the steady state value of the noise free system with the calculated static gain done in the preparation exercise.

Answer: _____

How would you estimate the static gain if you only had noisy data from a step response (as in the figure from the macro above)?

Answer: _____

Remark. As an alternative, simulation of the system could be done as follows

```
m=idpoly([1 a],[0 b]); % See help idpoly
y = sim(m,[u e]);  % See help idmodel/sim
```

A short overview of the commands in the System Identification Toolbox is obtained by `help ident`. We recommend that you remember this help function! You may also quickly browse through the list obtained from the help-function.

2.4 Correlation analysis

Next we consider correlation analysis (see the text book on p 251).

The input $u(t)$ is white binary noise of length $N = 1000$, taking the values ± 1. Correlation analysis is used to estimate the impulse response of the system for lags $k = 0, \ldots 20$. The task can be done using the following Matlab code, that is available as file `EM1b`.

```
%file EM1b.m
%Computer Excercise 1
%Empirical Modelling
%BC 080114
%last rev
%
a=-0.8,b=1,A=[1 a],B=[0  b],lambda=1, N=1000
e=randn(N,1)*sqrt(lambda);
u=sign(randn(N,1));
yNoNoise=filter(B,A,u);
y=yNoNoise+filter(1,A,e);
% Estimation of the impulse response, basically
%the algorithm 11.9 in Ljung Glad is used:
gest=cra([y u]);
```

Task Run the macro above. Notice the plot of the estimated impulse response.

Compare the estimated impulse response $\{gest(k)\}_{k=0}^{2}$ with the true values calculated in the Preparation excercise.

Answer: _____

Which one of the following system properties may be directly visible in the impulse response: Static gain, Number of poles or Time delay?

Answer: _____

_____

2.5 Spectral analysis

Next spectral analysis will be applied, see the text book Ch 11.3-11.4. The following input signal will be used:

$$u_2(t) = \frac{1}{1 - 0.8q^{-1}} u(t)$$

This gives an input with a more low-frequency character than white noise.
The estimated model from the spectral analysis, in form of a Bode plot, are compared with the Bode plot of the true system. The spectral analysis command `spa` is implemented with a Hamming lag window of length M. Do `help spa` to see an description of that command.
Note that the window length is denoted γ in Ljung Glad, Ch 11 (for example on p 266).

Try various values of M by using the file `EM1c`. Typically, you only need to generate data the first time you are using the function.

```
%file EM1c.m
%Computer Excercise 1
%Empirical Modelling
%BC 080114
%last rev
%
ND=input('Generate new data [y/n] (default n) ','s')
if strcmp(ND, 'y')==1
 a=-0.8,b=1,A=[1 a],B=[0 b],lambda=1, N=500
 e=randn(N,1)*lambda;
 u0=sign(randn(N,1));
 %Low pass filtering of u
 aa=-.8;
 u=filter(1,[1 aa],u0);
 %Generate data
 y=filter(B,A,u)+filter(1,A,e);
 z=[y u];
end

%Define true system as an idpoly object:
Gtrue=idpoly([1 a],[0 b]);
%
M=input('Give M (window size): ')
Gest=spa(z,M);
%Plot true and estimated model in bode diagrams:
bode(Gtrue,Gest)
```

Discuss how M affects the estimate. What value on M do you find reasonable[3]?

[3]Do not spend too long time on this task, the important thing is to see how M influences the estimate

Hint 1: You can plot various estimated models in one diagram with `bode(Gtrue,Gest10,Gest20,Gest100)` if you after each time you have run the macro give a unique name of your estimated model like `Gest20=Gest;`. If you want to see which curve corresponds to which model write `legend('Gtrue',Gest10','Gest20','Gest100')`. The legend commando creates a box in the plot where specified descriptions of the plots are shown.
Hint 2: If you write `figure` after you have run the macro, a new plot will be used the next time you run the macro (this should be seen as an alternative to Hint 1).

Answer: _____

Remark. One important thing concerning spa (and some other function in System Identification Toolbox). The function spa returns the frequency response as an IDFRD (Identified Frequency Response Data model) object. This is not an ordinary vector which could be plotted or inspected directly If we for example want to have the magnitude and phase as 'normal' vectors, the following code illustrates how to proceed:

```
gest=spa(z,20);
[tmp1 tmp2]=bode(gest);
Magnitude_est=squeeze(tmp1);
Phase_est=squeeze(tmp2);
```

3 Parametric methods

3.1 The least squares method for system $\mathcal{S}_1$

3.1.1 Background

We now turn to the least squares method, which in contrast to the previous methods gives a parametric model. The LS method is applicable to models of the form

$$A(q)y(t) = B(q)u(t) + \varepsilon(t) \tag{2}$$

which equivalently can be expressed as the linear regression model

$$y(t) = \varphi^T(t)\theta + \varepsilon(t) \tag{3}$$

where

$$\varphi^T(t) = [-y(t-1)\cdots - y(t-n_a) \quad u(t-1)\cdots u(t-n_b)] \tag{4}$$

$$\theta = [a_1 \cdots a_{n_a} \quad b_1 \cdots b_{n_b}] \tag{5}$$

The least squares estimate is defined as

$$\hat{\theta} = \big[\frac{1}{N}\sum_{t=1}^{N}\varphi(t)\varphi^T(t)\big]^{-1}\frac{1}{N}\sum_{t=1}^{N}\varphi(t)y(t) \tag{6}$$

The least squares (LS) estimate of an ARX model is computed with the command `arx`. Write `help arx` and inspect the syntax carefully! This is one of the most useful routines in the whole course.

3.1.2 Exercises

We first use the same system (system $\mathcal{S}_1$) as before, and the ARX model structure

$$(1 + aq^{-1})y(t) = bu(t-1) + \varepsilon(t)$$

$$\theta = (a \quad b)^T$$

Compute the LS estimate and compare with the true parameter values. Also illustrate the obtained model by computing and plotting the frequency function (Bode diagram). The task can be performed using the file `EM1d`, see the print out below. In this exercise just type return as answers to the two last questions from the macro.
Try a few different number of data points (for example 10, 100 and 1000). Does the estimated model seem to converge to the true system as the number of data points increases (check both parameter values and Bode plots)?

Answer:

Note also that (an estimation of the) confidence interval can be presented in the Bode plots. See `help idmodel/bode`. For example, you may try the following command: `bode(th,'sd',3,'fill')`.

```
%file EM1d.m
%Computer Laboratory 1
%System Identification
%BC 080105
%last rev BC 080129

% The Least squares method

clear
a=-0.8;,b=1;,A=[1 a];,B=[0 b];,lambda=1;
sys0=idpoly(A,B);
N=input('Give numer of data points to be used: ');
e=randn(N,1)*lambda;
u=sign(randn(N,1));
LP=input('Low pass filter the input signal [y/n] (default n) ','s');
if strcmp(LP, 'y')==1
aa=-.8;
u=filter(1,[1 aa],u)/sqrt(2.77); %u with variance 1
end
SYS=input('System S1 or S2 [S1/S2] (default S1)','s');
if strcmp(SYS, 'S2')==1
y=filter(B,A,u)+e;
else
y=filter(B,A,u)+filter(1,A,e);
end
z=[y,u];
  th=arx(z,[1 1 1]);
```

```
bode(sys0,th)
[aest,best]=polydata(th)
subplot(211)
title('blue=true system, green=estim. system');
%legend('red=true system','blue=estim. system',1);
```

3.2 The least squares method and non white noise

Next we will repeat the previous task but when the data is collected from the following system:

$$\begin{aligned} \mathcal{S}_2: \quad A(q)x(t) &= B(q)u(t) \\ y(t) &= x(t) + e(t) \end{aligned}$$

The system $\mathcal{S}_2$ can also be written as

$$A(q)y(t) = B(q)u(t) + A(q)e(t)$$

or

$$y(t) = \frac{B(q)}{A(q)}u(t) + e(t)$$

Note that the noise enters differently in the the system $\mathcal{S}_2$ than in the previously used system $\mathcal{S}_1$. In $\mathcal{S}_1$ it enters as white noise in the difference equation, while in $\mathcal{S}_2$ it appears as white noise added to the output (usually called white measurement noise). The only difference between the two systems is how the noise enters.

For system $\mathcal{S}_2$ as well as for $\mathcal{S}_1$, the parameters are given by

$$A(q) = 1 - 0.8q^{-1} \qquad B(q) = 1.0q^{-1} \qquad \lambda = 1$$

Use the macro above and enter S2 as answer to the last question. Is it possible to accurately estimate the system $\mathcal{S}_2$ with the least squares estimate of the ARX model? In this exercise, use many data points, for example $N = 1000$ (or higher!).

Answer: _____

This exercise is optional (do it only if you have time left). In this exercise you should repeat the previous task but with an input signal where the signal energy is dominating for low frequencies. (this is obtained by answering y to the second question in the macro). In this exercise, use a high number of data points, for example $N = 1000$ (and system S2).

Is the difference between the true and estimated system ('the bias') at low frequencies smaller or larger when the input is low pass filtered compared to when the input signal is white noise? Does this make sense?

Answer:

3.3 Concluding remarks

Least squares estimation of ARX models work well when the noise is white and enters as in the system $\mathcal{S}_1$. If, for example, the noise enters as in $\mathcal{S}_2$ the least squares estimate of the ARX parameters will be biased. That is, even if the number of data points goes to infinity, the correct parameters will not be found. The bias will also depend on the input signal property.

Later in the course, you will encounter methods that can handle more general noise characters and also methods that will be able to give a god model in a certain frequency band.

4 Examples of linear regression

In this section we give some exercises of linear regression models. Do them during the lab occasion if you have time left or later at your own.

4.1 A simple example

Consider the following polynomial model:

$$y(t) = \theta_o + \theta_1 u(t) + \theta_2 u(t)^2$$

Let the available data be:
$u(1) = 1,\ \ u(2) = 2,\ \ u(3) = 3,\ \ u(4) = 4$
$y(1) = 6,\ \ y(2) = 17,\ \ y(3) = 34,\ \ y(4) = 57$

Form the regressor matrix:

$$\Phi = \begin{pmatrix} 1 & u(1) & u(1)^2 \\ 1 & u(2) & u(2)^2 \\ 1 & u(3) & u(3)^2 \\ 1 & u(4) & u(4)^2 \end{pmatrix}$$

and

$$Y = \begin{pmatrix} 6 \\ 17 \\ 34 \\ 57 \end{pmatrix}$$

Calculate the least squares estimate

$$\hat{\theta} = (\Phi^T \Phi)^{-1} \Phi^T Y \tag{7}$$

What estimate do you get?

Answer: _____

Also try the following simple Matlab command (where Φ is called Phi) for the least squares estimate

```
thetaHat= Phi\Y
```

Verify that you (basically) get the same estimate[4]
Note, that mldivide has better numerical properties than (7)

4.2 Prediction of the word population by a polynomial model

In the file popdata.mat, the (approximate) population (unit billions) in the world is tabulated. The data cover the years 1950-1998, where year=0 corresponds to the year 1950. It may be of interest to predict the population in the world 2020. The example below illustrate how to proceed. A third order polynomial trend is estimated and then used to predict the population in 2020.

[4]See *help mldivide* for a description on the backslash command

```
load popdata
[P,S]=polyfit(year,pop,3)
pop2020=polyval(P,70)
plot([year; 70],[pop;pop2020])
```

Repeat the prediction for different polynomial orders (you may paste the code above into an m-file and then use the polynomial order as a variable) and note that an arbitrary crazy estimate can be obtained by choosing[5] the model order sufficiently high (despite that the errors in the model fit decreases). What model order do you think gives the most realistic prediction?

Answer: _____

4.3 Bad data

Consider the following simple model:

$$y(t) = \theta_1 u_1(t) + \theta_2 u_2(t)$$

This could serve as very simple model how two environmental factors are influencing an environmental index. Now assume that $u_1(t) \approx u_2(t)$. This could be the case if two correlated measurement series are used. Let for example
$u_1(1) = 1,\ \ u_1(2) = -1,\ \ u_1(3) = 1,\ \ u_4(4) = -1$
$u_2(1) = 1 + e_1,\ \ u_2(2) = -1 + e_2,\ \ u_2(3) = 1 + e_3,\ \ u_2(4) = -1 + e_4$
where e_i are small numbers. In this case,

$$\Phi = \begin{pmatrix} u_1(1) & u_2(1) \\ u_1(2) & u_2(2) \\ u_1(3) & u_2(3) \\ u_1(4) & u_2(4) \end{pmatrix}$$

Check the elements of $(\Phi^T\Phi)^{-1}$ for some small values of e_i. Example of code:

```
ee=0.0001; Phi=[1 1+ee;-1 -1+ee; 1 1+ee;-1 -1+ee];
inv(Phi'*Phi),
```

Do the elements become large if e_i is small? If so, the variance of the estimated parameters will be large and the standard least squares method is not useful!

Answer: _____

Remark. A good measure of the sensitivity of a matrix is given by the condition number (function *cond* in Matlab). A large condition number indicates a nearly singular matrix.

[5] Later in the course, we will present methods for how to select the model order.