Computer Exercise 2
Transfer function models and Prediction

In this computer exercise, you will work with input-output relations, as well as prediction in time series models. Firstly, you will be acquainted with time series having an exogenous input, having to analyze the impulse respons of such a system and from it build a suitable model. Secondly, this computer exercise deals with prediction, perhaps the most important application of time series modeling. You will be expected to make predictions of all models introduced in this course.

1 Preparations before the lab

Review chapters 3, 4, and carefully read chapter 6 in the course textbook. Make sure to read section 4.5 in particular, as it deals with transfer function models, as well as this entire computer exercise guide. Answers to some of the computer exercise will be graded using the course's *Mozquizto* page. Ensure that you can access the system before the exercise and answer the preparatory questions as well as (at least) three of numbered exercise questions below *before the exercise.*

You can find the *Mozquizto* system at

`https://quizms.maths.lth.se`

It should be stressed that a thorough understanding of the material in this exercise is important to be able to complete the course project, and we encourage you to discuss any questions you might have on the exercises with the teaching staff. This will save you a lot of time when you start working with the project!

You are allowed to solve the exercise in groups of two, but not more. Please respect this.

2 Lab Tasks

The computer program Matlab and the functions that belong to its System Identification Toolbox (SIT) will be used. In addition, some extra functions will also be used in this exercise. Make sure to download these functions and the required data files from the course homepage. You are free to use other programs, such as R or Python, but then need to find the appropriate functions to use on your own.

2.1 Modeling of an exogenous input signal

In this and in the next section, you will work with modeling of input-output relations, both using the ARMAX model and the transfer function model frameworks. As modeling of a signal which has an exogenous input (an input which is known, i.e., deterministic) is generally more complex than the common time series models encountered so far in this course, one must take care and proceed with caution. Often very simple models of a low order will suffice, while complex ones will only add variance, detrimental to the precision of predictions.

We start by creating a typical time series with a deterministic input signal, using a slight generalization of the ARMAX model, i.e., the Box-Jenkins (BJ) model, having the form of

$$y_t = \frac{B(z)z^{-d}}{A_2(z)} x_t + \frac{C_1(z)}{A_1(z)} e_t$$

where y_t is the output signal, e_t is a white noise, x_t is the input signal, and d is the time delay between input and output. Note that if $A_1(z) = A_2(z)$, we have the standard ARMAX model.

Begin by generating some data following the Box-Jenkins model:

```
rng(0)
n = 500;                                   % Number of samples
A3 = [1 .5];
C3 = [1 -.3 .2];
w =  sqrt(2) * randn(n + 100,1);
x = filter(C3,A3,w);                       % Create the input

A1 = [1 -.65];
A2 = [1 .90 .78];
C  = 1;
B  = [0 0 0 0 .4];
e  = sqrt(1.5) * randn(n + 100,1);
y  = filter(C,A1,e) + filter(B,A2,x);   % Create the output

x = x(101:end), y = y(101:end)         % Omit initial samples
clear A1, A2, C, B, e, w, A3, C3
```

Here, the known input x_t has been generated as an ARMA(1,2) process.

Remark: As discussed in the first computer exercise, we typically generate more data than needed when simulating a process to avoid initialisation effects. We here also clear the variables used to create the signals to avoid the risk of accidentally referring to these later in the code. One notable benefit of using simulated data in this way is that we know the true values we seek, so we can compare our results with these to see if our code works properly.

In order to now model y_t as a time series formed from x_t and e_t, several steps must be taken beyond regular ARMA modeling. We must first select the appropriate model orders for the polynomials in the model, then proceeding to estimate the parameters of these polynomials. This may be done in various ways; here, we will follow the steps outlined in Section 4.5 in the course textbook[1].

1. As a first step, we wish to determine the orders of the $B(z)$ and $A_2(z)$ polynomials. Using the transfer function framework, we denote the transfer function from x_t to y_t by $H(z) = B(z)z^{-d}/A_2(z)$. In order to estimate the order of the $B(z)$ and $A_2(z)$ polynomials, as well as determining the delay d, we need to form an estimate of the (possibly infinite) impulse response, and from it identify the appropriate models for these polynomials.

   As noted in the course textbook, if x_t is a white noise, the (scaled) impulse response can be directly estimated using the cross correlation function (CCF) from x_t to y_t. However, if x_t is not white, we need to perform pre-whitening, i.e., we need to form a model for the input, such that it may be viewed as being driven by a white noise, and then inverse filter both input and output with this model. In order to do so, we form an ARMA model of the input

   $$A_3(z)x_t = C_3(z)w_t$$

   and then replace x_t with w_t, i.e.,

   $$y_t = \frac{B(z)z^{-d}}{A_2(z)}\frac{C_3(z)}{A_3(z)}w_t + \frac{C_1(z)}{A_1(z)}e_t$$

   The pre-whitening step, i.e., multiplying with $A_3(z)/C_3(z)$, yields

   $$\underbrace{\frac{A_3(z)}{C_3(z)}y_t}_{\epsilon_t} = \underbrace{\frac{B(z)z^{-d}}{A_2(z)}}_{H(z)}w_t + \underbrace{\frac{A_3(z)}{C_3(z)}\frac{C_1(z)}{A_1(z)}e_t}_{v_t}$$

   and the preferred transfer function model may thus be expressed as

   $$\epsilon_t = H(z)w_t + v_t$$

   Note that the pre-whitened ϵ_t is now the output of the transfer function model, having the preferred uncorrelated signal as its input, allowing $H(z)$ to be estimated using the CCF from w_t to ϵ_t.

[1] However, it should be noted that if you can select your model orders in another way, including simply guessing, this is fully acceptable - what counts is if your model actually works, not the intermediate steps used to designed it!

Task: Use the basic analysis (`acf`, `pacf`, and `normplot`) to create an ARMA model for the input signal x_t as a function of a white noise, w_t. Which model did you find most suitable for x_t? Is it reasonably close to the one you used to generate the input?

QUESTION 1 *In Mozquizto, answer question 1.*

We then pre-whiten y_t, creating ϵ_t. Next, we compute the CCF from w_t to ϵ_t by typing

```
M = 40; stem(-M:M, crosscorr(w_t  ,eps_t , M));
title('Cross correlation function'), xlabel('Lag')
hold on
plot(-M:M,  2/sqrt(n)*ones(1,2*M+1),'--')
plot(-M:M, -2/sqrt(n)*ones(1,2*M+1),'--')
hold off
```

As the estimated CCF now yields an estimate of the impulse response, $H(z)$, we can proceed to use this to determine suitable model orders for the delay, and the $B(z)$ and $A_2(z)$ polynomials using Table 4.7 in the textbook. Use `pem` to estimate your model, using

```
A2     = ... ;
B      = ... ;
Mi     = idpoly([1],[B],[],[],[A2]);
z      = iddata(y,x);
Mba2   = pem(z,Mi); present(Mba2)
etilde = resid(Mba2,z);
```

where the delay may be added to `B` by adding d zeros in the beginning of the vector. If the model orders are suitable, the CCF between the input, x_t, and the residual $\tilde{e}_t$ (defined below) should be uncorrelated.

Task: Analyze the CCF w_t to ϵ_t to find the model orders of the transfer function. Calculate the residual $\tilde{e}_t$ and verify that it is uncorrelated with x_t. Also, analyze the residual using the regular basic analysis. Can you conclude that $\tilde{e}_t$ is white noise? Should it be?

QUESTION 2 *In Mozquizto, answer question 2.*

2. We have now modeled y_t as a function of the input x_t, but have not yet formed a model of the ARMA-process in the BJ model, i.e., modeled the polynomials $C_1(z)$ and $A_1(z)$. Therefore, defining the ARMA-part as

$$\tilde{e}_t = \frac{C_1(z)}{A_1(z)} e_t$$

we use the estimated polynomials $B(z)$ and $A_2(z)$ and estimate $\tilde{e}_t$ as

$$\tilde{e}_t = y_t - \frac{\hat{B}(z) z^{-\hat{d}}}{\hat{A}_2(z)} x_t$$

By filtering out the input-dependent part of the process y_t, we may then estimate determining suitable orders for the polynomials $C_1(z)$ and $A_1(z)$ using the standard ARMA-modeling procedure.

Task: Use the estimates of the polynomials $B(z)$ and $A_2(z)$ obtained for the pre-whitened data and form $\tilde{e}_t$. Determine suitable model orders for $A_1(z)$ and $C_1(z)$. Was all dependence from x_t removed in $\tilde{e}_t$?

QUESTION 3 *In Mozquizto, answer question 3.*

3. Finally, now having determined all the polynomial orders in our model, we estimate all polynomials all together using `pem`.

```
A1 = ...
A2 = ...
B  = ...
C  = ...
Mi = idpoly(1,B,C,A1,A2);
z  = iddata(y,x);
MboxJ = pem(z,Mi);
present(MboxJ)
ehat = resid(MboxJ,z);
```

Here, `ehat` is the estimate of the noise process e_t; notice that this is not the same process as $\tilde{e}_t$ (which is the filtered version of e_t as shown above).

Task: Analyze the model residual, verifying that the CCF from x_t to e_t, as well as the basic analysis, shows it to be white. Are the parameter estimates significantly different from zero? Can you conclude that the residual is white noise, uncorrelated with the input signal? If not, can you twiddle with the model slightly to improve the residual?

Be prepared to answer these questions when discussing with the examiner at the computer exercise!

2.2 Hairdryer data

In this section, we will try to construct a model for a set of measured data. In the file `tork.dat`, you will find 1000 observations from an input-output experiment. These measurements have been obtained from a laboratory process, which essentially is a hair dryer with measuring equipment, i.e., air is propelled by a fan through a pipe. The air is heated at the entrance of the pipe and its temperature is measured at the outlet. The input signal that is applied, stored in the second column of the data set, is the voltage over the heating coil and the output signal. The first column is the temperature of the airflow at the outlet. This physical systems can be reasonably well modeled using a simple linear model of the process. The sampling distance is 0.08 s.

Start by accessing the data material and subtract the mean values, create an `iddata` object, now having both an input and an output, and plot the first 300 points of the object using

```
load('tork.dat')
tork = tork - repmat(mean(tork),length(tork),1);
y = tork(:,1); x = tork(:,2);
z = iddata(y,x);
plot(z(1:300))
```

Task: Model this input-output relation using the Box-Jenkins model introduced above. Repeating the steps in section 2.1, use the basic analysis and the CCF to find suitable model orders. Finally, estimate the model in its entirety and plot the `acf`, `pacf`, `normplot`, and CCF from x_t to e_t.

How long is the delay from x_t to y_t in seconds? Can you conclude that the residual is white noise, uncorrelated with the input signal? Are the parameter estimates significantly different from zero? If not, can you twiddle with the model slightly to improve the residual?

Be prepared to answer these questions when discussing with the examiner at the computer exercise!

2.3 Prediction of ARMA-processes

In this section, we examine how to predict future values of a process, using temperature measurements from the Swedish city Svedala. The temperature data is sampled every hour during a period in April and May 1994, with its (estimated) mean value subtracted (11.35° C). Load the measurements with the command `load svedala`. Suitable model parameters for the data set are

```
A = [ 1 -1.79 0.84 ];
C = [ 1 -0.18 -0.11 ];
```

To make a k-step prediction, $\hat{y}_{t+k|t}$, one needs to solve the equation

$$C(z)\hat{y}_{t+k|t} = G_k(z)y_t$$

This can be done using the filter command[2]

```
yhat_k = filter( Gk, C, y );
```

where G_k is obtained from the Diophantine equation

$$C(z) = A(z)F_k(z) + z^{-k}G_k(z).$$

Here, we have included the desired prediction range, k, in the polynomials $G_k(z)$ and $F_k(z)$ to stress that you will need a different polynomial for each k. Thus, if you wish to predict two steps ahead into the future, forming both $\hat{y}_{t+1|t}$ and $\hat{y}_{t+2|t}$, you will need to use both $G_1(z)$ and $G_2(z)$ to construct these estimates.

[2]Remember to remove the initial samples after using the command `filter`.

To solve the Diophantine equation, you can use the provided function `polydiv`

```
[Fk, Gk] = polydiv( C, A, k );
```

The prediction error is formed as

$$y_{t+k} - \hat{y}_{t+k|t} = F_k(z) e_{t+k},$$

Note in particular that the prediction error will (for a perfect model) have the form of an MA($k-1$) process with the generating polynomial

$$F_k(z) = 1 + f_1 z^{-1} + \cdots + f_{k-1} z^{-(k-1)}.$$

Note also that if $k = 1$, then $F_1(z) = 1$, suggesting that the prediction error should be a white noise, and that, for this case, the prediction error thus allows for an estimate of the noise variance.

QUESTION 4 *In Mozquizto, answer question 4.*

Task: In the following questions, examine the `k`-step prediction using $k = 3$ and 26. Answer the following questions:

1. What is the estimated mean and the expectation of the prediction error for each of these cases?
2. Assuming that the estimated noise variance is the true one, what is the theoretical variance of the prediction error? Using the same noise variance, what is the estimated variance of the prediction error? Comment on the the differences in these variances.
3. For each of the cases, determine the theoretical 95% confidence interval of the prediction errors?
4. How large percentage of the prediction errors are outside the 95% confidence interval? A useful trick might be to use `sum(res>c)` to compute how many elements in `res` that are greater then `c`.
5. Plot the process and the predictions in the same plot, and in a separate figure, plot the residuals. Check if the sequence of residuals behaves as an MA($k-1$) process by, e.g., estimating its covariance function using `covf`. If it does not, try to explain why.

Be prepared to answer these questions when discussing with the examiner at the computer exercise!

2.4 Prediction of ARMAX-processes

When predicting ARMAX-processes, one needs to consider also the external input. We will now make use of an additional temperature measurement done at the airport Sturup. The Swedish Meteorological and Hydrological Institute (SMHI) has made a 3-step predictions of the temperature for Sturup, which may be used as an external signal to our temperature measurements in Svedala.

Load the SMHI predictions into Matlab, with `load sturup`, and set the model parameters to be

```
A = [ 1  -1.49  0.57  ];
B = [ 0 0 0  0.28  -0.26  ];
C = [ 1 ];
```

How large is the delay in this temperature model? How do you know? Form the k-step predictor of the temperature at Svedala using $k = 3$ and 26 as

$$C(z)\hat{y}_{t+k|t} = B(z)F_k(z)x_t + G_k(z)y_t,$$

where $F_k(z)$ and $G_k(z)$ are computed as indicated above. The k-step prediction is then formed as

$$\hat{y}_{t+k|t} = \hat{F}_k(z)\hat{x}_{t+k|t} + \frac{\hat{G}_k(z)}{C(z)}x_t + \frac{G_k(z)}{C(z)}y_t$$

where $\hat{x}_{t+k}$ denotes the predicted future inputs, and the polynomials $\hat{F}_k(z)$ and $\hat{G}_k(z)$ are given by the Diophantine equation

$$B(z)F_k(z) = C(z)\hat{F}_k(z) + z^{-k}\hat{G}_k(z)$$

In the prediction, the two first terms represents the contribution of the input signal, whereas the third is from the ARMA part of the process.

QUESTION 5 *In Mozquizto, answer question 5.*

Important: A common error is that one forgets to add the term $\hat{F}_k(z)\hat{x}_{t+k|t}$ when forming the prediction $\hat{y}_{t+k|t}$. Note that it is *only* in cases when the input cannot be predicted, i.e., when x_t is a white process, that one omits the $\hat{F}_k(z)\hat{x}_{t+k|t}$ term from the prediction. Otherwise, when x_t has any form of structure, it may be predicted, and then the term *should* be included. To avoid making this error, it is recommended that you *always* include the term; when predicting $\hat{x}_{t+k|t}$ in the (rare) white noise case, this will of course be zero, so you will just add a zero sequence, which will not corrupt your results, but then you will not forget to add it, which will certainly cause problematic results (this typically appears as predictions that seem to have the correct pattern, but with a too low amplitude).

Important: Another common error is that one gets removes a different number of initial samples when creating $\frac{\hat{G}_k(z)}{C(z)}x_t$ and $\hat{F}_k(z)\hat{x}_{t+k|t}$; as discussed before, one needs to remove the same number of samples as the order of the denominator polynomial to avoid the problem of the initialization of the filter. However, to avoid the sequences to get out of sync with each other one should remove the same number of samples from both sequences, so that one removes the maximum of the number of samples that are required to be removed from either sequence[3].

Task: Using $k = 3$, what is the variance of the prediction errors? Plot the process, the prediction and the prediction errors.

A very common error when making predictions of ARMAX and BJ processes is to forget to add the $\hat{F}_k(z)\hat{x}_{t+k|t}$ term. Plot this erroneous prediction and the corresponding prediction errors. Can you see how this error appears in your prediction?

Be prepared to answer these questions when discussing with the examiner at the computer exercise!

2.5 Prediction of SARIMA-processes

The temperature measurements from Svedala are very periodical and can therefore likely be well modeled as a SARIMA-process. Load the data and find the suitable period. What is the period for this data?

In Matlab, it is easier to write the SARIMA-processes as an ARMA process by including the differentiation in the $A(z)$ polynomial. Let

$$\nabla_S y_t = (1 - z^{-S})y_t$$

which, in Matlab, is the same as filtering y_t with the polynomial

```
AS = [ 1 zeros(1, S-1) -1 ];
```

Remember those initial samples! Forming predictions for the SARIMA-model

$$A(z)\nabla_S y_{t+k|t} = C(z)e_{t+k},$$

may be done by seeing it as a non-stable ARMA-model (recall that that polynomial multiplication may be computed using `conv`), and performing predictions for such a model.

[3] A common problem is that one gets prediction that are out of sync with the reference data; this may be due to that one has made the above error, but can also be due to the shifting resulting from the group delay (see also the lecture code). Often, it is easiest to find this form of errors using a simulated data sequence, forming a somewhat longer (say $k = 7$) prediction with the *true* $A(z)$ and $C(z)$ polynomials. Then, the resulting shift is easy to see, and you can easier find the reason for it; it is also often helpful to first plot the prediction without the predicted input and see if you are still out of sync, and, if not, then add the input and see where the shift occurs.

Task: After removing the season, form an appropriate model for the Svedala data. Which model did you find? Compute the estimated prediction error variance for $k = 3$ and 26, and compare them with the variance obtained from the ARMA model. Are they any better?

Be prepared to answer these questions when discussing with the examiner at the computer exercise!