Lund University
Mathematical Statistics
Centre for Mathematical Sciences

Time Series Analysis
Fall 2019

Computer Exercise 1
Estimation and Model Validation

This computer exercise treats identification, estimation, and model validation in ARMA- and SARIMA-processes. Your goal is to be proficient in simulating different kinds of stationary and non-stationary time series, and to master the 'craft' of identification and modeling of such. You will use the autocorrelation function (ACF), the partial autocorrelation function (PACF), probability plots and different parameter estimators as your basic toolkit. In this exercise, you will use these on both simulated and measured time series data.

1 Preparations before the lab

Read Chapter 3 and 4 in the course textbook as well as this guide to the computer exercise. Solve the problem exercises 3.6, 3.10, 4.3, 4.4, and 5.2, as well as answer the preparatory exercises in the following section. Before the lab starts these questions will be discussed. You are expected to be able to show that you have studied them in order to pass the computer exercise.

2 Preparatory exercises

1. Define
   (a) an ARIMA-process
   (b) a SARIMA-process
   (c) the ACF
   (d) the PACF
2. Study an AR-process

$$y_t + a_1 y_{t-1} + a_2 y_{t-2} = e_t,$$

   where $y_1, \ldots, y_{100}$ have been observed. In order to compute the LS-estimate of the unknown parameter a_1 and a_2 a particular function has to be minimized. Which function is this? Express your answer using a matrix formulation. A solution to the problem may be computed using the normal equations. Write down these equations. Remember to define your notation. How can $\sigma^2 = V(e_t)$ be estimated?

3. Consider an ARMA-process

$$y_t + a_1 y_{t-1} = e_t + c_1 e_{t-1},$$

where the parameters a_1 and c_1 shall be estimated using the ML-method. It is then necessary to compute the residuals $\epsilon(t; a, c)$ for different values of a and c. How do we compute these residuals? Which expression should be minimized in order to get the ML-estimates? Let $\hat{a}$ and $\hat{c}$ be these ML-estimates. Then, $\epsilon(t; \hat{a}, \hat{c})$ is is a reconstruction of a certain quantity. Which one?

4. Write down the Matlab code you would use to estimate the parameters a_1, a_3, c_1, c_{12} for the model:

$$\begin{aligned} A(z) &= 1 + a_1 z^{-1} + a_3 z^{-3} \\ S &= 5 \\ C(z) &= 1 + c_1 z^{-1} + c_{12} z^{-12} \end{aligned}$$

for the data y (See section (3.4)).

5. Describe FPE. How is this used and how does it compare to an ordinary residual variance estimate?

6. Analysis of the residuals is one possible way to validate the estimated model. What statistical properties are the residuals $\epsilon(t, \hat{\theta})$ expected to have when the model is reasonably correct? How may this be verified?

7. How can one estimate the covariance function $r(\tau)$? Mention some of the properties of the estimator. Give an expression for the variance of the estimator. What is the rule-of-thumb for a maximum value of τ?

3 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 the toolkit of functions and data from the course homepage.

Use a new empty script in matlab for inputing all your code. Save this in the same folder as the toolkit or use the `addpath()` command. You can easily divide your script into cells using *cell mode* in Matlaband then run these separately. Simply switch *cell mode* on and use `%%` on an empty line to divide your code is separate parts. To run a cell press *Run Section* in the toolbar.

Please make sure to answer all highlighted questions, i.e., 1-12, and summarize your results (on a sheet of paper, for instance), as to make presenting to the teaching assistants easier.

3.1 Working with time series in Matlab

Important: In the following, we will make use of various Matlab functions. Use the command `help` to learn more about these functions and their expected inputs and outputs. For example, type `help pzmap` in the command window to learn about the function `pzmap`.

In this first section, we will work with two sets of A- and C-polynomials, the first set being:

```
A1 = [ 1 -1.79 0.84 ];
C1 = [ 1 -0.18 -0.11 ];
```

and the second set:

```
A2 = [ 1 -1.79 ];
C2 = [ 1 -0.18 -0.11 ];
```

These polynomials builds an ARMA-process of the form

$$A(z)Y(t) = C(z)e(t).$$

Create the model structure objects for the two sets of polynomials using the command:

```
ARMA_poly = idpoly( A, [], C );
```

The resulting object `ARMA_poly` will contain the specified polynomials, making it easier to pass all the information about the process to the different relevant Matlab functions (remember to use the `help` command to learn more about the function `idpoly`). Of course, you may name your objects as you see fit, for example `arma1` and `arma2`, or something shorter. You may directly access the contained polynomials by access the member variables of the object. For example, to access the A-polynomial, type

```
ARMA_poly.a
```

If you do not know the member variables of an object, you may view these by typing

```
ARMA_poly.
```

and then press tab. To view the poles and zeros of the ARMA-process, you can use the function `pzmap` by writing

```
pzmap( ARMA_poly )
```

An easy way to simulate an ARMA-process

$$Y(t) = \frac{C(z)}{A(z)} e(t)$$

in Matlab, is to use the function `filter`. The function needs the A and C polynomials as well as the driving noise process to generate a simulation. You can easily generate normal distributed noise with

```
e = sqrt(sigma2) * randn( N, 1 );
```

where `sigma2` is the variance of the noise process `e`, and `N` is the length of the resulting vector. After creating the noise vector, you may simulate the process with

```
y = filter( ARMA_poly.c, ARMA_poly.a, e );
```

Proceed to simulate the `y1` and `y2` processes, setting `sigma2=1.5` and `N=200`. Plot the two process using the subplot command.

```
subplot(211)
plot(y1)
subplot(212)
plot(y2)
```

QUESTION 1 *Explain why one of the processes diverges?*

The covariance function $r_Y(\tau)$ for an ARMA-process can be computed using the Yule-Walker equations. The function `rtheo = kovarians(C,A,m)` computes the theoretical covariance function using the specified polynomials, returning a column matrix `r` with the covariance function values for $\tau = 0, \ldots, \mathtt{m}$. The function assumes that the driving noise process has unit variance, i.e., $V(e(t)) = \sigma^2 = 1$. To instead compute the estimated covariance from a given data set, use the function `rest = covf(y,m)`.

Plot to compare the theoretical versus the estimated covariance for `y1` by (set `m=20`) by inputing

```
rtheo = kovarians( ARMA_poly.c, ARMA_poly.a, m );
stem( 0:m, rtheo*sigma2 )
hold on
rest = covf( y1, m+1 )
stem( 0:m, rest ,'r')
```

QUESTION 2

1. *Why are the estimated and theoretical covariance functions not identical?*
2. *How many lags of estimated covariance function are trust-worthy? How does this compare to general rule of thumb?*

Your task is now to re-estimate your model using the data vector `y1`. Do some basic analysis by plotting the `acf pacf`, and `normplot` of your data (for instance together in a figure with three subplots). For estimation, make your data an iddata object (Matlab time domain object), i.e.,

```
data = iddata(y1);
```

Estimate the model you wish to validate using the LS-based method `arx` for an AR(na) by typing

```
ar_model = arx( y1, [na]);
```

where `na` is the model order, or using the ML-based method `armax` for an ARMA(na,nc) by typing

```
arma_model = armax( y1, [na nc]);
```

To get an overview of the estimated parameters, their standard deviation, and the model FPE, use `present`. To calculate the error residual of your estimated model, you can do this with `filter` by running your data through the inverse filter of your model. Do so by typing

```
e_hat = filter( ARMA_poly.a, ARMA_poly.c, y );
```

Find an appropriate model model for the process by adding one extra parameter at a time, and then analyze the residual using `acf pacf`, and `normplot`.

QUESTION 3 *What is your 'best' model according to* `FPE`*? Could you consider another model order?*

3.2 Model order estimation of an AR-process

In this subsection you will perform model order estimation upon another AR(2)-process beyond the basic analysis. The aim is to get a more statistical feel for the distribution of the estimated parameters and how this affects the choice of model order. Choose the parameters of the AR-process, simulate `n=500` data points using the function `filter`, discard the first 100 values (why?) and then plot by inputing

```
n      = 500;
A      = [1 -1.35  0.43];
sigma2 = 4;
noise = sqrt(sigma2) * randn(n + 100,1);
y      = filter(1,A,noise);
y      = y(101:end);
subplot(211)
plot(y)
subplot(212)
plot(noise)
```

When choosing model order with Matlab you can use `arxstruc` which takes two `iddata` objects, one for estimation and one for validation, it also takes a matrix which explains which models to compare.

First, we create the estimation and validation data sets. Here, we use the first two third of the data to estimate the models and use the rest of the data to validate the models.

```
n_est = floor(2/3 * n);
y_est = iddata(y(1:n_est));
y_val = iddata(y(n_est + 1:end));
```

To compare two models, for instance, one sets up the matrix

```
NN=[na1 nb1 nk1;
    na2 nb2 nk2 ]
```

where `na,nb,nk` are the order of the A and B polynomials and the delay of the B polynomial, respectively. Since you only compare AR models without input signal set, use

```
NN = [1:10]';
```

to only assign different `na`:s, and then, setting it all together,

```
V       = arxstruc(y_est,y_val,NN);
n_order = selstruc(V,0);
n_aic   = selstruc(V,'aic');
```

where the function `selstruc` chooses which model order is the best fit (0 is to use a least squares model order criterion). Now, repeat the model selection 100 times using a loop and store `n_order,n_aic` for each iteration in the loop, i.e.,

```
for i= 1:100
        noise       =
        y           =
        y_est       =
        y_val       =
        V           =
        n_order(i)  =
        n_aic(i)    =
end
```

QUESTION 4

1. *Plot the histograms of* `n_order,n_aic`*. What can you explain from the histogram? Why and how does the two histograms vary?*

2. *Repeat the same loop as above, but double the amount of data points, i.e.* `n=1000`*. How and why does the histogram change ?*

To get and view the model that is estimated last use

```
ar_model = arx(y,n_order(end));
ar_model.NoiseVariance
ar_model.CovarianceMatrix
present(ar_model)
```

3.3 Model order estimation of an ARMA-process

We will now extend the parameter space to include MA-parameters. In the file `data.dat` you will find 200 observations of an ARMA(1,1)-process y_t. Begin working with this data by trying to estimate AR(p)-models (for $p = 1, 2, \ldots$) using `arx`. Instead of using `filter`, an alternative way of performing residuals analysis is to use the Matlab SIT toolbox, inputing, e.g.,

```
data=iddata(data);
rar1=resid(ar1_model,data);
rar2=resid(ar2_model,data);
```

with, for instance, the first two model orders. You may then, for comparison plot the residuals (available as `rar1.y` for the AR(1)-model) together with the original noise that was used for generating `data` (can be found in `noise.dat`). Also try typing `resid(ar1_model,data)`, i.e., without assigning an output variable.

When examining which model order to use, type `present(arp_model)` (for model order `p`) to see the parameter estimates and some statistics. Then try modeling with some different orders `p`.

QUESTION 5

1. *Does the estimate of a_1, for $p = 1$ and $p = 2$ coincide with the true value of the parameter ($a_1 = -0.6, c_1 = 0.8$)? Can you find an explanation?*
2. *Which model order is best according to FPE (taking the statistical significance of the parameters into account), and which is best according to theory (i.e., which AR(p) model approximates an ARMA(1,1) model best in theory)?*

Finally, estimate ARMA(p, p)–models for $p = 1, 2, \ldots$ on the data which truly is from an ARMA-process, by the ML-method,

```
am11_model = armax(data,[1 1])
am22_model = armax(data,[2 2])
```

QUESTION 6 *Which ARMA-model is best according to FPE (taking the statistical significance of the parameters into account)?*

3.4 Estimation of a SARIMA-process

In this subsection, you will work with a special type of non-stationary time series models, that is the seasonal ARMA models. Your task is to, first generate data from the model yourself using white Gaussian noise, and then use the ML-based estimator `pem` to re-estimate the parameters. So, when working with SARIMA-models, you often find model structures of the type

$$A(z)\nabla_s y_t = C(z)e_t,$$

where you may find, when performing your basic analysis and building the model, that you want to fix some parameters to zero. In the `pem` you can do this by indicating which of the parameters that should be estimated and which should not.

Simulate a SARIMA-process using a Gaussian noise as input, by inputing

```
A = [1 -1.5 0.7];
C = [1 zeros(1,11) -0.5];
A12 = [1 zeros(1,11) -1];
A_star = conv(A,A12);
e = randn(600,1);
y = filter(C,A_star,e);
y = y(100:end);
plot(y)
```

QUESTION 7 *Perform basic analysis (*`ACF`*,* `PACF`*, and* `normplot`*) of the process. What characteristics does it have?*

The ringning behavior you see in both the `ACF` and the `PACF` indicates strong seasonality. Therefore, start by removing the season and then create an `iddata` object

```
y_s=filter(A12,1,y);
data=iddata(y_s);
```

QUESTION 8 *Perform basic analysis (*`ACF`*,* `PACF`*, and* `normplot`*) of the deseasonalized process. What characteristics does it have?*

Then, set up an initial model with the Matlab object `idpoly`.

```
model_init = idpoly(A,B,C);
```

This object will set the maximum order and initial values of each of the polynomials. If one does not use for instance the B-polynomial, then one should set `B=[]`. Building a model to this data, one parameter at a time, could for instance mean starting with an AR(1), as an initial guess, redo the basic analysis and then proceed to an AR(2). Thus, use `pem` to estimate a_1 and a_2 by inputing

```
model_init = idpoly([1 0 0],[],[]);
model_armax=pem(data,model_init)
```

and then use the inverse of this filter to obtain the model residual.

QUESTION 9 *Perform basic analysis (*`ACF`*,* `PACF`*, and* `normplot`*) of the residual. What characteristics does it have?*

To add a single parameter of a higher order, i.e., to set some parameters fixed to a value, then set this value in the initial polynomial, i.e., when creating `model_init` and then specify which parameters to be free in this polynomial (`idpoly` structure) by:

```
model_init.Structure.c.Free = [zeros(1,12) 1]
```

Where a zero sets the parameter as fixed and a one sets the parameter as free. For the C-polynomial, we have thus made all parameters fixed except for c_{12}. Include this parameter in the model and estimate a_1, a_2, and c_{12} by inputing

```
model_init = idpoly([1 0 0],[],[1 zeros(1,12)]);
model_init.Structure.c.Free = [zeros(1,12) 1];
model_armax = pem(data,model_init)
```

QUESTION 10 *Perform basic analysis (*`ACF`*,* `PACF`*, and* `normplot`*) of the residual. Is it white noise? What are the parameter estimates?*

To summarize this computer exercise so far, we might then see the modeling of time series as working in accordance with the following steps:

1. Is there a trend? Remove it.
2. Is there any seasonality? Remove it.
3. Iterate between
   (a) Which is the lowest AR- or MA-parameter that is strong? Remove it.
   (b) Is the residual white noise? If not, go to (a). Is it Gaussian? If not, what are the consequences?
4. Are all parameter estimates statistically significant? If not, redo analysis and use a smaller model (order).

3.5 Estimation on real data

We now proceed to use our knowledge on estimation of SARIMA-models on real data. Load the temperature measurements from Svedala, using the command `load(svedala)`. Using the working order stated above, create a suitable model for this data.

QUESTION 11

- *What parameter estimates did you get?*
- *Can you improve the model by not differentiating, i.e., removing the season of, the data? Why or why not?*

4 MATLAB–routines

General MATLAB–routines

`armarx`	estimates the parameters of an ARMA-process an with exogenous input
`arx`	estimates the parameters of an AR-process an with exogenous input
`arxstruc`	creates a Matlab-structure consisting of many AR-models.
`covf`	estimates the covariance function for a data series with mean value equal to zero
`filter`	generates an ARMA-process from an input signal
`iddata`	creates an input/output data structure
`idpoly`	creates a polynomial ARMAX structure
`mean`	computes the mean value of data
`normplot`	makes a normal probability plot of the data
`pem`	compute the prediction error estimate of a linear or nonlinear model
`present`	presents a parametric model on the screen
`randn`	generates Gaussian-distributed data
`std`	computes the standard deviation of data
`selstruc`	determines the best model in an arxstruc-object according to some criterion

MATLAB–routines particularly derived for the lab

`acf`	estimates the Auto Correlation Function (ACF) for some dataset
`kovarians`	computes the theoretical covariance function for an ARMA-process
`pacf`	estimates the Partial Auto Correlation Function (PACF) for some dataset
`spekgui`	graphical user interface for spectral estimates.
`spekplot`	plots the spectrum etc. on a lin-lin scale
`tspekt`	computes the theoretical spectral density for an ARMA-process