Lund University
Centre for Mathematical Sciences
Mathematical Statistics

Computer exercise 3 in Stationary stochastic processes, HT 18.

The purpose with this computer exercise is to examine the connection between the covariance function and the spectral density of a stationary stochastic process and to study different AR- and MA-processes.

1 Preparations

- Read through the entire computer exercise.
- Study chapters 7.1-7.3 in the course book.
- **Answer the exercises in the question dictionary 1.1 below and E-mail your answers to FMSF10@matstat.lu.se at latest by Thursday October 11 at 8.00.** The answers can be written in english or swedish by hand (readable) or in any text editor program but the final attached file should be a pdf-file. Use the following in the subject row, "Assignment 3 by tpi07xyz" (one student) or "Assignment 3 by tfy08xyz and nat09xyz" (two students). Larger group sizes than two are not allowed. **Note that you might be asked to explain your answers during the exercise.**

1.1 Question dictionary

1) Define the covariance function and spectral density of an AR(1)-process. For which frequency does the spectral density reach its maximum for $a_1 < 0$ and for $a_1 > 0$, respectively?

2) Calculate the covariance function $r_X(\tau)$, for $\tau = 0, \pm 1, \pm 2, \pm 3$, of the AR(2)-process $X_t + X_{t-1} + 0.5X_{t-2} = e_t$, where e_t is white Gaussian noise with $E[e_t] = 0$ and $V[e_t] = 1$.

3) How do the locations of poles and zeros of an ARMA-process relate to the view of the spectral density? Define an example, explain and illustrate.

4) To ensure stationarity, what is required of the poles of an AR(p)-process?

5) How does covariance function of an MA(q)-process depend on the order q?

2 ARMA-models in MATLAB

This section is an introduction to how AR, MA and ARMA models are handled in MATLAB.

An ARMA(p, q)-model

$$X_t + a_1 X_{t-1} + \cdots + a_p X_{t-p} = e_t + c_1 e_{t-1} + \cdots + c_q e_{t-q}$$

can also be written as

$$A(z^{-1})X_t = C(z^{-1})e_t,$$

where

$$A(z^{-1}) = 1 + a_1 z^{-1} + \cdots + a_p z^{-p}$$
$$C(z^{-1}) = 1 + c_1 z^{-1} + \cdots + c_q z^{-q}$$

and where the operator z^{-1} is the unit time delay. If you choose $C(z) \equiv 1$, the process becomes an AR(p)-process, whereas if you choose $A(z) \equiv 1$, it becomes an MA(q)-process. In MATLAB, polynomials are represented as vectors; e.g. $z^p A(z^{-1}) = z^p + \cdots + a_{p-1}z + a_p$ is represented as `[1 a1 ... ap]`.

2.1 Study of AR(1)-processes

Create an AR(1)-model in MATLAB according to the previous section, i.e., $C(z) = 1$, and $A(z^{-1}) = 1 + a_1 z^{-1}$ represented as `C=1` and `A=[1 a1]` for your choice of the parameter. Remember to save your MATLAB code in scripts. One may compute the spectral density $R_X(f)$ for the process as

```
>> [H,w]=freqz(C,A);
>> R=abs(H).^2;
```

where `freqz` is a MATLAB function that calculates the frequency function corresponding to the polynomials `C` and `A` in a number of equally spaced frequency values (default 512) between $0 \leq \omega < \pi$, with $\omega = 2\pi f$. Plot the spectral density using

```
>> plot(w/2/pi,R)
```

To calculate the covariance function, the inverse Fourier transform of the spectral density is computed. The frequency range must be the whole period with the frequency range $0 \leq f < 1$ when the command `ifft` is used. The frequency function is computed for this range using `whole` in the function `freqz`,

```
>> H=freqz(C,A,512,'whole');
>> Rd=abs(H).^2;
>> r=ifft(Rd);
>> stem([0:49],r(1:50))
```

where the 50 first values of the covariance function are plotted. Observe that the vector element `r(1)` is equal to the covariance value at zero, i.e. $r_X(0) =$ `r(1)`, and more generally $r_X(k) =$ `r(k+1)` in MATLAB. Try a positive and a negative value of a_1, e.g. $a_1 = 0.5$ and $a_1 = -0.5$, and study the results.

Q. Draw rough sketches of the covariance functions for $a_1 < 0$ ***and*** $a_1 > 0$***. How do they differ?***

Q. Sketch the spectral densities. At what frequency does the spectral density reaches the maximum value for $a_1 < 0$ ***and*** $a_1 > 0$***, respectively?***

To simulate a realisation of the AR-process, we generate a sequence of independent and Gaussian[1] distributed stochastic variables, $\{e_t, t = 1, \ldots, n\}$. This can be made as

```
>> m = 0;
>> sigma = 1;
>> e = normrnd(m, sigma, 1, n);
```

or, alternatively,

```
>> e = randn(1,n);
```

Each call to the above function will generate a row-vector `e` with `n` Gaussian distributed random numbers; here, choose `n`≈ 400. The simulation of the AR(1)-process is made using

[1] We note that, in general, e_t does not need to be Gaussian distributed.

```
>> x = filter(C, A, e);
```

which result in the vector `x` with the simulated time-series as the result, where `A` and `C` are defined as earlier. Create one realization for the process where $a_1 < 0$ and one for $a_1 > 0$.

Q. Sketch the view of the realizations. Express X_t in X_{t-1} and e_t for the two cases and explain the differences in how the realizations evolve with time.

2.2 Study of an ARMA(2,2)-process

Specify an ARMA(2,2)-process using `A=[1 -1 0.5]` and `C=[1 1 0.5]`. It is often convenient to specify the roots as poles and zeros instead of the coefficients of the A- and C-polynomials. Compute and plot the poles and zeros of the ARMA(2,2)-process using

```
>> P = roots(A)
>> Z = roots(C)
```

and

```
>> zplane(Z,P)
```

Note: When using the function `zplane`, the poles must be located in column vectors. If row vectors are used as input, `zplane` interprets this as polynomial vectors and plots `roots(A)`. Verify this by using the commands `zplane(Z,P)` and `zplane(C,A)`.

If the poles `P` and zeros `Z` are specified instead and you want to calculate the polynomials `A` and `C`, one may use the command `poly`

```
>> A=poly(P)
>> C=poly(Z)
```

Sketch the spectral density just from the view of the pole-zero plot. Verify by computing and plotting the spectral density according to the previous subsection.

Q. Which parts of the spectral density have high power and which has low power respectively in reference to the location of poles and zeros?

3 Study of ARMA-processes using armagui

Use the function `armagui` to investigate the connections between poles, zeros, covariance functions and spectral densities for different ARMA-models. Type

```
>> help armagui
```

to get information how to use it, and start the function by typing

```
>> armagui
```

The help text is also included at the end of this exercise. Use this function in the continuation.

3.1 Study of AR(2)-processes

Use `armagui` to produce an AR(2)-model with a complex conjugated pair of poles. Restart by removing the poles with the button "Remove all". Add a complex pole, (gives you the pole as well as the conjugate). Move the poles (easily done with the mouse button). Change the angle frequency from the real axis and keep quite close to the unit circle. Study the spectral density and the covariance function. Change the distances of the poles from the unit circle and study the covariance function and the spectral density.

Q. How do the covariance function and spectral density change when you increase the angle? How do they change when you increase the distance from the unit circle?

Move your poles very close to the origin.

Q. Sketch the spectral density and covariance function. Explain the views from the expression of the AR-process in this case.

To see what happens when the generating AR-filter is unstable, i.e., if the roots to the characteristic polynomial are outside the unit circle, de-select the button "Demand stability" in `armagui` and move one pole/pole pair outside the unit circle. Zoom in at the beginning of the realization and compare with the amplitude for larger values of t.

Q. What happens with the process realization? Why?

3.2 Study of MA(q)-processes

MA-processes behave a bit different from AR-processes and we will study MA-processes of some different orders. Restart by removing the poles with the button "Remove all". Add a complex zero pair, generating an MA(2)-process. Add more zeros to form MA(3)-, MA(4)- and MA(5)-processes.

Q. For which values of τ do the covariance functions of the different processes above become zero? How is this value of τ related to the order?

Examine the appearance of the spectrum from the view of the zeros placement. Study especially what happens when you have complex zeros close to the unit circle.

Q. Sketch two spectral densities, one with complex zeros selected close to the unit circle and second with complex zeros selected close to the origin. Explain the views from the locations of the zeros.

4 Speech modeling in cell phones

An early method for modeling speech is that it locally can be assumed to be an AR(20)-process. A refined version of this model is actually still used when transmitting sound between cellphones (or mobile phones if you prefer British English). When you are speaking in a cellphone, the sound is divided into 20 millisecond long time frames. The parameters of the model are estimated; these parameters of the AR-model are then sent to the other persons phone. By running white noise through the model, the speech is reconstructed.

4.1 Parameter estimation

Open the file `lab3_extra.m`. The code loads an audio file where someone is saying *"Why were you away a year Roy?"*. You can listen the file, once it's loaded, by typing

```
>>sound(x,Fs);
```

Plot the process and check that the process is zero mean! The file is then divided into sections of 20 ms; for each section we want to estimate the parameters of the AR(20)-model. Your task is the write that code into the for-loop. It's a very useful exercise to write this code yourself; for help see pages 180-181 in the book. However if you are short on time, there are built-in functions in MATLAB that you can look up. The function `arcov` solves exactly this problem and you can also use this function to verify that your programming is correct.

Once the parameters are estimated, one can see the sizes of audio file `x` and the AR parameters in `ar_param` by writing:

```
>> whos x ar_param
```

This will show you the sizes of the audio-file and the size of the parameters in bytes.

Q. How much storage is saved by sending the AR-parameters instead of the audio itself?

4.2 Evaluation

The second part of the code in `lab3_extra.m` will estimate the spectrum of one section of the audio file and compare it with the spectrum of the estimated AR(20)-model. If your code work they should show some similarities, although they are not exactly the same.

The third section of the code will reconstruct the audio using your estimated model parameters. The original audio and the reconstructed audio will be plotted in figure one. NOTE: Before playing the reconstructed audio, check the plotted reconstruction and make sure it is stable!

If the reconstruction looks stable and close to the original audio you can listen to it by typing

```
>> sound(x_rec,Fs)
```

As you (hopefully) can hear, the reconstruction is quite noisy and sounds somewhat inhuman. This is, as mentioned, an early model; if you want to know how some easy changes can be made, reducing the order to *AR(10)* with improved sound quality, you can learn this in the followup course *Time series analysis.*

5 MATLAB–functions

armagui

```
function armagui(action)
% ARMAGUI
%
% armagui  opens a window for  ARMA(p,q)-modeling.
%
% Use the buttons to add poles and zeros. Move them by using the mouse.
% Delete poles and zeros with the right mouse button.
%
% If the button "Demand stability" is marked, poles and zeros
% are not allowed outside the unit circle.
%
% The button "Simulate" gives a realization of the process.
%
% A- and C-polynomial can be loaded into ARMAGUI by creating, e.g.,
%
% >> modell.A=[1 -0.5];
% >> modell.C=[1 0.4 0.7];
%
% and write the name (i.e.
% "model") in the "Import"-box and push the button.
%
% Covariance function, spectral density and realizations can be
% exported to Matlab with the "Export"-button.
% Then a variable with the name written in the "Export"-box is created,
% which includes poles, zeros etc..
```