Lund University
Mathematical Statistics
Centre for Mathematical Sciences

Time Series Analysis
Fall 2020

Computer Exercise 3
Recursive Estimation and
Models with Time-Varying Parameters

This computer exercise treats recursive parameter estimation using Kalman filtering and recursive least squares. We attempt to model dynamic systems of both the SARIMA-type, having time-varying A and C polynomials, as well as to allow for ARMAX processes which have a synthetic input signal and time-varying B polynomial.

1 Preparations before the lab

Read Chapter 8 in the course textbook as well as this guide to the computer exercise. Sketch an m-files that:

1. Simulates the process u_t in Section 2.3 below. Let u_t be a Markov chain that switches slowly between two states, using $p_{11} = p_{22} = 7/8$ and $p_{12} = p_{21} = 1/8$.
2. Express an AR(2) process on state space form and estimate the parameters of the process using a Kalman filter. Section 3 gives a rough outline of the required code.

You are expected to be able to answer detailed questions on your implementation. 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 Recursive Least Squares estimation

You will begin by implementing the recursive least squares (RLS) estimation of time-varying A-parameters in an AR model.

1. Load the data material `tar2.dat`, the data is an AR(2)-process with one time dependent parameter and the other one constant. The correct parameter trajectories are stored in the file `thx.dat`. Use `subplot` to plot the data and the parameter in the same figure.

2. Use the Matlab `rarx` function to estimate the A-parameters recursively. Set up a `model=[na,nb,nk]` (remember what `nb,nk` mean? If there is no input, only set `na`). Try different forgetting factors, using $\lambda = 1$, 0.99, 0.95 with the Matlab code

```
[Aest,yhat,covAest,yprev]=rarx(tar2,model,'ff',lambda)
```

   Here, `Aest` is the estimated parameters, `yhat` is the one step estimation of time step, whereas `covAest` and `yprev` are used for on-line estimation (which we will not use here). Plot the parameter estimates together with the true parameter.

3. To choose λ, one option is to use the least squares estimate.

```
n = 100;
lambda_line = linspace(0.85,1,n);
ls2         = zeros(n,1);
for i=1:length(lambda_line)
    [Aest,yhat,CovAest,trash]=...
    rarx(tar2,[2],'ff',lambda_line(i));
    ls2(i) = sum((tar2-yhat).^2);
end
plot(lambda_line,ls2)
```

What effects has the choice of λ for the parameter estimates? Explain what `ls2` vector contains? What does the plot show?

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

2.2 Kalman filtering of time series

A quite important drawback of the RLS estimate is that it should not be used to estimate MA parameters, and one should therefore not use it for MA or ARMA processes. We continue to again estimate the AR parameters from the previous section, but by using the Kalman filter that you've implemented in the preparatory exercises. Note that the Kalman implementation can be extended to also allow for MA coefficients.

Set the measurement variance (`Rw`) to 1.25. Plot the parameter estimate. Tune `Re` to improve the estimate. Try tuning `Rw` to improve the estimates.

How do you choose `Re` so that one of the parameter is assumed to vary with time whereas the other is constant? What effect has the choice of `Rw` and `Re` for the parameter estimates? Did you manage to improve the estimation by using Kalman filtering instead of RLS? What are the advantages and disadvantages of using Kalman filtering?

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

2.3 Quality control of a process

In the quality control division at a factory, one has found that the process which is to be followed shows a drift like

$$x_t = x_{t-1} + e_t.$$

However, it is not possible to measure the quality variable x_t exactly, and one instead is limited to the observations

$$y_t = x_t + bu_t + v_t,$$

where the processes e_t and v_t are two mutually uncorrelated sequences of white noise, with the variances σ_e^2 and σ_v^2. Furthermore, b is a parameter. Assume that the external signal u_t is known.

Use the m-files written in the preparatory exercises for the lab to simulate the process with the input signal u_t. Select $b = 20$, $\sigma_e^2 = 1$ and $\sigma_v^2 = 4$, but feel free to change these at will. Now consider x_t and b to be unknown, and use the Kalman filter you prepared and implement a filter that estimates b. Plot your estimates of the hidden states together with the originals.

How should the state space vector be chosen? How would you chose an initial value of `Re` and `Rw`? How can then proceed to fine-tune the filter?

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

2.4 Recursive temperature modeling

The file `svedala94.mat` contains temperature measurements from the Swedish city Svedala, taken every four hours throughout 1994. The temperature can be modeled using a SARIMA$(2,0,2) \times (0,1,0)_6$ process, i.e., $A(z)\nabla_6 y_t = C(z)e_t$, where the $A(z)$ and $C(z)$ polynomials are of order 2.

1. Plot the temperature, differentiate (use `filter` and remember those initial samples) and plot the differentiated temperature. To obtain months on the x-axis, you can use the commands

```
T = linspace(datenum(1994,1,1),datenum(1994,12,31),...
  length(svedala94));
plot(T,svedala94);
datetick('x');
```

2. Use `armax` to estimate an ARMA(2,2)-process for the differentiated data. Do this (i) for the entire year, (ii) for January-March, and (iii) for June-August. Compare the different estimated parameters.

```
th = armax(ydiff,[2 2]);
th_winter = armax(ydiff(1:540),[2 2]);
th_summer = armax(ydiff(907:1458),[2 2]);
```

3. Use `rarmax` to recursively estimate the A- and C-parameters for the differentiated process. The computation may take some time. It is possible to use the parameters estimated for January-March as initial values for the recursive estimation. Try some different forgetting factors. Compare with the non-recursive estimates obtained above. Plot the results and explain why and when the parameters vary.

```
th0 = [th_winter.A(2:end) th_winter.C(2:end)];
[thr,yhat] = rarmax(ydiff,[2 2],'ff',0.99,th0);

subplot(311)
plot(T,svedala94);
datetick('x')

subplot(312)
plot(thr(:,1:2))
hold on
plot(repmat(th_winter.A(2:end),[length(thr) 1]),'b:');
plot(repmat(th_summer.A(2:end),[length(thr) 1]),'r:');
axis tight
hold off
```

```
subplot(313)
plot(thr(:,3:end))
hold on
plot(repmat(th_winter.C(2:end),[length(thr) 1]),'b:');
plot(repmat(th_summer.C(2:end),[length(thr) 1]),'r:');
axis tight
hold off
```

Is it likely that the `th_winter` and `th_summer` are different processes? Does the recursive process coincide with the non-recursive winter and summer processes?

2.5 Recursive temperature modeling, again

An alternative to the seasonal operator and recursive ARMA-models is to use a reasonable input signal that models both the mean value and the seasonal variation of the process. To illustrate this, we will once more examine the temperature in Svedala, now using an ARMAX-model.

1. Load the Svedala data, `load svedala94`, extract a suitable period of data, e.g. `y = svedala94(850:1100)`, and subtract the mean value.

2. We want to use a sinusoidal signal as input in an ARMAX model to avoid seasonal filtering. However, since the phase of the oscillation is unknown, we will use a combination of a sine and a cosine signals. Construct the external signal, and estimate the parameters by first forming the model

```
t = (1:length(y))';
U = [sin(2*pi*t/6) cos(2*pi*t/6)];
Z = iddata(y,U);
model = [3 [1 1] 4 [0 0]];
       %[na [nb_1 nb_2] nc [nk_1 nk_2]];
```

   Then, use `thx = armax(Z,model)` to estimate the coefficients for the sines and cosines. The coefficients are stored in `thx.b`. Plot `y`, together with the seasonal function `U*cell2mat(thx.b)`.

   What happens if you change the season? Why? Is the season well modeled using this method? How can it be improved?

3. A varying mean can be estimated by inclusion of a constant external signal, $u_t = 1$, if the parameters are estimated recursively. Using Kalman filter estimation allows the estimated coefficients to be expressed in state from

$$x_t = Ax_{t-1} + e$$

   where $e \sim N(0, R_e)$ and x_t is the estimated coefficients at time t. Assuming that the parameter estimates are independent, the state covariance matrix R_e is diagonal, with diagonal elements corresponding to the noise variance for each of the states. Thus, if the ℓth diagonal index of R_e is non-zero, it is assumed to vary over time, otherwise not.

Hint: Order the states such that it contains the A coefficients, the B coefficients, and the C coefficients, in that order. Implement the code below and add the diagonal components of `Re` so that the only time dependent coefficient is the mean (choose this value to be 1).

```
U = [sin(2*pi*t/6) cos(2*pi*t/6) ones(size(t))];
Z = iddata(y,U);
m0 = [thx.A(2:end) thx.B' 0 thx.C(2:end)];
Re = diag([? ? ? ? ? ? ? ? ? ?] );
model=[3 [1 1 1] 4 0 [0 0 0] [1 1 1]];
[thr,yhat] = rpem(Z,model,'kf',Re,m0);
```

The `model` vector in `rpem` indicates the various model orders in the form

$$\Big[\text{na } [\text{nb}_1 \cdots] \text{ nc nd } [\text{nf}_1 \cdots] \ [\text{nk}_1 \cdots]\Big]$$

Here, the time delay for the input signals is necessary due to limitations in the command `rpem`.

4. The parts of the temperature that is due to the varying mean and the sine/cosine signals can now be reconstructed as

```
m = thr(:,?);
a = thr(end,4);
b = thr(end,5);
y_mean = m + a*U(:,1)+b*U(:,2);
y_mean = [0;y_mean(1:end-1)];
```

In the same plot, display `y` and `y_mean`. The values `a` and `b` are taken as the last values in `thr` as these are the final estimate of the constant coefficients.

5. We proceed to study the data from the entire year, recalling your earlier results from section 2.4. Due to stability issues the first values of the process should be close to zero.

```
y = svedala94;
y = y-y(1);
```

As before, estimate the process, letting only the mean value vary. Try finding a reasonable value for R_e.

Compare `y_mean` and `y`. Are these similar? If not, how and why do they differ? Is it a good estimate? If not, explain why and give a solution to make it better.

3 Kalman Filter Outline

```
% Example of Kalman filter

% Simulate process
y = ?;

% Data length
N = length(y);

% Define the state space equations
A   = [? ?; ? ?];
Re  = [? ?; ? ?]; % Hidden state noise covariance matrix
Rw  = ?; % Observation variance

%usually C should be set here to,
%but in this case C is a function of time.

% Set some initial values
Rxx_1 = ? * eye(2); % Initial variance
xtt_1 = [? ?]';     % Initial state

% Vector to store values in
xsave=zeros(2,N);

% Kalman filter. Start from k=3,
% since we need old values of y.
for k=3:N,
  % C is, in our case, a function of time.
  C = [? ?];

  % Update
  Ryy   = ?;
  Kt    = ?;
  xtt   = ?:
  Rxx   = ?;

  % Save
  xsave(:,k) = ?;

  % Predict
  Rxx_1 = ?;
  xtt_1 = ?;

end;
```