Time Series Analysis 2025
Lund University

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. Answers to some of the computer exercise will be graded using the course's *Mozquizto* page, available at `https://quizms.maths.lth.se`. 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.* These questions aim at allowing to check your implementation.

Before the computer exercise:

1. Express an AR(2) process on state space form and estimate the parameters of the process using a Kalman filter as specified in Sections 2.2 and 2.3.
2. Write a Matlab script that simulates the process u_t in Section 2.4 below. Let u_t be a Markov chain that switches slowly between two states, using $p_{11} = p_{22} = 7/8$ and $p_{12} = p_{21} = 1/8$. Hint: This is easy to do using a loop where you at each time instance change state according to the specified probabilities.

Note that 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 examining the recursive least squares (RLS) estimate of a time-varying AR process.

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 object `recursiveAR` to estimate the $A(z)$ polynomial recursively with the Matlab code

```
X = recursiveAR(2);
X.ForgettingFactor = 1;
X.InitialA = [1 0 0];
for kk=1:length(tar2)
    [Aest(kk,:),yhat(kk)] = step(X, tar2(kk) );
end
```

   Here, `Aest` is the estimated parameters, `yhat` is the estimate of y_t based on the estimated $A(z)$ polynomial and past values of y_t. Try different forgetting factors, using $\lambda = 1$, 0.95, 0.9. Plot the parameter estimates together with the true parameter. What effect does the value of λ have?

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);
yhat  = zeros(n,1);
for i=1:length(lambda_line)
     reset(X);
     X.ForgettingFactor = lambda_line(i);
     X.InitialA = [1 0 0];
     for kk=1:length(tar2)
          [~,yhat(kk)] = step(X, tar2(kk) );
     end
     ls2(i)=sum((tar2-yhat).^2);
end
plot(lambda_line,ls2)
```

QUESTION 1 *In Mozquizto, answer question 1.*

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, making it unsuitable for, e.g., ARMA processes. We continue to again estimate the AR parameters from the previous section, but by using the Kalman filter that you have implemented in the preparatory exercises. Note that the Kalman implementation can be extended to also allow for MA coefficients.

We here make use of the example code given in Section 3. At first, ignore the part of the example code for the 2-step prediction. This code uses data up to time $t-1$ to predict the state value $\hat{x}_{t|t-1}$, stored in the variable `x_t1`, and then use this value to predict (the one-step prediction) $\hat{y}_{t|t-1}$. The prediction error (also often termed the prediction residual) between $\hat{y}_{t|t-1}$ and y_t, i.e.,

$$\epsilon_{t|t} = y_t - \hat{y}_{t|t-1}$$

is then used to update the Kalman filter. Here, the prediction residual is stored in the variable `ehat`.

Proceed to complete the missing part of the code (ignore the part for the 2-step prediction). Use the tar2 data as y_t. Then, to be able to check your implementation, set

```
Re     = [ 0.004  0;  0  0  ];
Rw     = 1.25;
Rx_t1  = 10*eye(2);
```

Also, set the initial value of the state vector to zero values. This will set the covariance matrix of the observation (`Re`) and measurement (`Rw`) noises, as well as the initial state vector and its covariance matrix. The large initial value for `Rx_t1`, i.e., $R^{x,x}_{t+1|t}$, indicates that we have little confidence in the initial states. Notice in particular that the way `Re` is selected reflects the assumption that the first parameter varies, whereas the second does not. Plot the resulting parameter estimates and notice the difference in convergence between the two parameters.

QUESTION 2 *In Mozquizto, answer question 2.*

This question strives to check that your implementation is correct. As you will use this as the basis for the following steps, as well as in the project, it is important that you get this to work properly. Ask the teaching staff to help you if you do not get the correct answer!

QUESTION 3 *In Mozquizto, answer question 3.*

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? Can you reduce the sum of the squared residual (`ehat`) by tuning `Re` and `Rw`?

2.3 Using the Kalman filter for prediction

We now proceed to form a 2-step prediction using the Kalman filter. To do this, you need to complete the latter part of the example code. As it can be difficult to verify that the implementation is correct, we will use simulated data for this. To be able to check your implementation, use

```
rng(0);
N  = 10000;
ee = 0.1*randn(N,1);
A0 = [1 -.8 .2];
y  = filter(1, A0, ee);
Re = [1e-6 0; 0 1e-6];
Rw = 0.1;
```

We here select `Re` small to ensure that we get states that converge close to the true values (which is also why we select N so large).

In order to form $\hat{y}_{t+2|t}$, you first need to form $\hat{C}_{t+2|t}$, as

$$\hat{y}_{t+2|t} = \hat{C}_{t+2|t}\hat{x}_{t+2|t} = \hat{C}_{t+2|t}\hat{x}_{t|t}$$

Note that, in general, $\hat{x}_{t+2|t} \neq \hat{x}_{t|t}$. Why does this equality hold here?

Proceed to write down an expression for $\hat{C}_{t+2|t}$ and note that this will depend on $\hat{y}_{t+1|t}$. As a result, you will first need to estimate $\hat{y}_{t+1|t}$, then use this value to form $\hat{C}_{t+2|t}$, and then finally use this to form $\hat{y}_{t+2|t}$. Update the example code with the missing lines.

To examine the predictions when the filter has converged, use the following code to plot the last 100 samples of y_t, $\hat{y}_{t|t-1}$, and $\hat{y}_{t+2|t}$.

```
indV = N-101:N-2;
plot( indV, [y(indV) yhat1(indV) yhat2(indV)] )
legend('y_t', 'y_{t|t-1}', 'y_{t+2|t}')
title( 'Data vs predictions')
xlabel('Time')
xlim([ indV(1) indV(end) ])
```

Notice that the plot stops at $N-2$. Why is that? Plot the estimated states and check that they converge properly.

QUESTION 4 *In Mozquizto, answer question 4.*

Again, it is important as you get your code to work properly as you will use this code in the project. Ask the teaching staff if you have problems!

Compute the sum of the squared prediction residual for the last 100 samples (why not all?). Can you improve the estimate, i.e., lower the sum of the squared prediction residual by tuning the choice of `Rw` and `Re`?

Show the plot of y_t, $\hat{y}_{t|t-1}$, and $\hat{y}_{t+2|t}$, as well of the predicted states, to the teaching staff. Why is it that, in general, $\hat{x}_{t+2|t} \neq \hat{x}_{t|t}$?

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

It is worth noting that in case you are using the Kalman filter to predict an MA or an ARMA process, your `C` vector will contain earlier noise values, i.e., e_t, e_{t-1}, etc. These noise values are obviously not known, so one then use the corresponding one-step prediction errors in place of these, i.e., $\epsilon_{t|t}$, $\epsilon_{t-1|t-1}$, etc. These values are here stored in the variable `ehat`.

Important: Note that when using the Kalman filter for prediction, you *should not* use the polynomial division techniques discussed in the earlier computer exercise, as presented in Chapter 6 in the course textbook. This will not yield the correct estimates! Instead, predictions needs to be made by updating the states as indicated in Chapter 8.

2.4 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. For simplicity, we assume that the external signal u_t is known.

Use the m-file written in the preparatory exercise for the computer exercise 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 true values. Plot the one-step prediction $\hat{y}_{t|t-1}$ as compared to the measured signal y_t.

How should the state space vector be chosen? How would you choose 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!

2.5 Recursive temperature modeling

The file svedala94.mat contains temperature measurements from the Swedish city Svedala, taken every four hours throughout 1994.
One potential model to describe the temperature with can be 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 the process to form $\nabla_6 y_t$ (use filter and remember those initial samples) and plot the differentiated temperature. To obtain months on the x-axis, use

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

2. Determine your own model for the first 540 samples. One potential model could be an ARMA(3,6), formed as

$$\nabla_6(1 + a_1 z^{-1} + a_2 z^{-2} + a_3 z^{-3})y_t = (1 + c_6 z^{-6})e_t$$

   Did you get a similar model? To allow comparisons, we will now use the above ARMA(3,6) model. Estimate the parameters for this model for a) the winter (say, January to March, i.e., samples 1 to 540), b) summer (say, June to August, i.e., samples 907 to 1458), and c) for the entire year. Compare the different estimated parameters. Do they seem to change? Does the model seem to work reasonably throughout the year?

3. Use the winter model to form a one-step prediction of the temperature for the validation data, here the last 400 samples, i.e., samples 1790 to 2190. To do so, add in the differentiated season in the model before computing the $G(z)$ polynomial to ensure that the prediction is formed in the correct domain. Apply the resulting filter to the *entire* signal. Then, cut out the part of the prediction that forms the validation data (this way you avoid any initialization problems). Plot the ACF of the resulting prediction residual for the validation data and compute its variance.

4. Use the Kalman filter to estimate the parameters of the ARMA(3,6) model throughout the year. To do so requires some care. To begin with, do not remove the initial samples when forming $z_t = \nabla_6 y_t = y_t - y_{t-6}$. Clearly, these initial samples will be corrupted, as always, but we will

instead remove these later to simplify the alignment of the vectors. Then, form the Kalman filter predict $\hat{z}_{t|t-1}$ using the ARMA structure above using the appropriate `C` vector. As

$$\hat{y}_{t|t-1} = \hat{z}_{t|t-1} + y_{t-6}$$

you can reconstruct $\hat{y}_{t|t-1}$ using

```
    zt = C*x_t1;                     % \hat{z}_{t|t-1}
    yhat(t) = zt + y(t-6);           % \hat{y}_{t|t-1}
```

Next, you can initiate the model at the beginning of the data set using the parameters you have found from the modeling data. To make the plots a bit nicer, we do this also for the states before the predictions:

```
xt = zeros(p0+q0,N);
for k=1:6
    xt(:,k) = [winterModel.A(2:end)'; winterModel.C(end)];
end
```

Here, the parameters `p0` and `q0` indicate the number of unknown coefficients in the $A(z)$ and $C(z)$-polynomials, respectively. As this choice of initial states should be fairly accurate, you trust your initial estimates to be reasonably good and can thus select a low initial $R^{x,x}_{1|0}$, for instance using

```
Rx_t1  = 1e-5*eye(p0+q0);
Re     = 1e-5*eye(p0+q0);
Rw     = 2;
```

where we have also selected rather low values for R_w and R_e; try using some different values to get a feel for how these choices affect the resulting estimates and if you can improve the quality of the estimates[1].

Compare the resulting prediction with the result you got above when using the winter model as well as with the naive predictor. As you can see, the winter model is actually rather stable and actually works quite well even for the validation data, even without allowing the parameters to vary. Can you suggest any reasons why this might be?

5. Proceed to plot the 2-step and 3-step predictions of the temperature data. It is worth noting that the 3-step prediction will need to use the 2-step prediction. Compute the variance of the corresponding prediction residuals.

Be prepared to show your parameter estimates and predictions when discussing with the examiner at the computer exercise!

[1]Note that you can (and maybe should?) have different parameter variances.

2.6 (optional) Examine the project data

Using the model for the input signal you developed in the second computer exercise, form a one-step prediction of this signal using a Kalman filter. Select the state covariances using the variance of the estimated prediction residual and the standard deviation of the parameter estimates you obtained in the second computer exercise. Initiate your parameters with the values you obtain in the second exercise.

Start the filtering at the beginning of the entire data set, to avoid any initialization and convergence effects, and then extract the predictions for the validation data. Is the prediction residual white? Compute the variance of the prediction residual. Plot the estimated parameters. Are these similar to the ones you had with the fixed model?[2] Are some parameters very small in comparison to their standard deviations? Are they significant? Can you remove some parameters without increasing the variance of the prediction residual more than marginally? Do you get better estimates if you fix some of the parameters so that these are not allowed to vary, instead using the parameter value you had from the second exercise?

Form a k-step prediction of the input using the Kalman filter. Recall that you need to update the `Ct` vector at each step. How should you handle the future noise estimates? Examine the ACF of the prediction residual as well as its variance. Do you get a lower variance than you had for your fixed-parameter model?[3]

Hint: You will typically perform these task as part of your project, so the time you spend on this now will be time saved later on...

[2]Often, they are not, so this is not a cause for concern in itself.

[3]This is not given, in case the ”true” parameters do not really vary, a fixed-parameter model may well perform better than one that allows the parameters to vary!

3 Kalman Filter Outline

```
% Example of Kalman filter

% Simulate N samples of a process to test your code.
y = ?                          % Simulated data

% Define the state space equations.
A   = ?;
Re  = ?;                       % State covariance matrix
Rw  = ?;                       % Observation variance

% Set some initial values
Rx_t1 = ? * eye(?);            % R_{1|0}^{x,x}
h_et  = zeros(?,1);            % Estimated prediction error.
xt    = zeros(?,?);            % Estimated states.
yhat1 = zeros(?,1);            % Estimated 1-step prediction.
yhat2 = zeros(?,1);            % Estimated 2-step prediction.

% Where should the loop start and stop?
for t= ? : ?

   % Update the predicted state
    x_t1 = A*xt(:,t-1);                      % x_{t|t-1}
    Ct = ?;                                  % C_{t|t-1}

    % Update the parameter estimates.
    Ry = ?;                                  % R_{t|t-1}^{y,y}
    Kt = ?;                                  % K_t
    yhat = Ct*x_t1;                          % \hat{y}_{t|t-1}.
    h_et(t) = y(t)-yhat;                     % Prediction error,
    xt(:,t) = x_t1 + Kt*( h_et(t) );         % x_{t|t}

    % Update the covariance matrix estimates.
    Rx_t  = ?;                               % R^{x,x}_{t|t}
    Rx_t1 = ?;                               % R^{x,x}_{t+1|t}

    % Form \hat{y}_{t+1|t}.
    Ct1 = ?;                                 % C_{t+1|t}
    yhat1(t+1) = Ct1*xt(:,t);                % \hat{y}_{t+1|t}

    % Form \hat{y}_{t+2|t}.
    Ct2 = ?;                                 % C_{t+2|t}
    yhat2(t+2) = Ct2*xt(:,t);                % \hat{y}_{t+2|t}
end
```