Assignment 4: Kalman filter and Parameter Estimation

Instructions: The assignment report is to be handed in via DTU Learn "FeedbackFruits" latest at May 23rd at 23:59. You are allowed to hand in in groups of 1 to 4 persons. You must hand in a single pdf file presenting the results using text, math, tables and plots, do not include code in the report! Shortened software result output is ok to include, to save some time formatting in tables. Arrange the report in sections and subsections according to the questions in this document. Please indicate your student numbers on the report.

NOTE, that this time there is no peer-review. The teachers evaluate the reports and the assessment will count for the grade.

NOTE, that the report should not be too long. Include one or two (ok, with more figures for the longer questions) (i.e. one figure can have multiple plots) per question and make the text concise! Long and unprecise reports are not good!

All the additional material needed is provided in the `assignment4.zip` file.

1 Parameter estimation in state-space model

This exercise concerns parameter estimation in state space models and how to use the Kalman filter for this.

Let the process $\{X_t\}$ be an 1-dimensional process given by the state space model

$$X_t = AX_{t-1} + Bu_{t-1} + e_t$$

where will will deal with the version:

$$X_t = aX_{1-1} + b + e_t$$

where $\{e_t\}$ is a unit variance white noise process.

Note on Notation:

In this part of assignment, we work with a simplified version of the linear state-space model. You have seen the standard formulation in the slides:

$$X_t = AX_{t-1} + Bu_{t-1} + e_{1,t}$$
$$Y_t = CX_t + e_{2,t},$$

where:

- X_t is the state at time t,
- u_{t-1} is the input (here assumed constant),
- $e_{1,t} \sim \mathcal{N}(0, \Sigma_1)$ is system noise,
- Y_t is the observation,
- $e_{2,t} \sim \mathcal{N}(0, \Sigma_2)$ is observation noise.

In this part of the assignment, the input is constant and absorbed into a bias term, resulting in a scalar system of the form:

$$X_t = AX_{t-1} + B + e_{1,t},$$

where:

- A is the scalar state transition coefficient (previously written as `b` in code),
- B is a scalar bias term (previously `a` in code),
- $e_{1,t} \sim \mathcal{N}(0, \sigma_1^2)$ is standard Gaussian noise.

Likewise, the observation model used in this part is:

$$Y_t = X_t + e_{2,t}, \quad \text{which corresponds to } C = 1.$$

To make the 1-dimensionality of this process explicit, we replace the capitol letter notation of (usally matrixes) A, B, C, with the lower case notation a, b and c respectively.

Answer the following:

1.1. Simulate 5 independent realizations of the process up to $n = 100$ time steps, using the parameter values $a = 0.9$, $b = 1$, $\sigma_1 = 1$, and an initial value $X_0 = 5$.

Implement:

- Plot all 5 trajectories in a single figure. Use different colors or line types to distinguish them.
- Remember to set a random seed to ensure reproducibility.

1.2. Simulate a single realization of the process described below:

$$X_t = aX_{t-1} + b + e_{1,t}, \quad e_{1,t} \sim \mathcal{N}(0, \sigma_1^2),$$
$$Y_t = X_t + e_{2,t}, \quad e_{2,t} \sim \mathcal{N}(0, \sigma_2^2 = 1),$$

using the same parameters as in the previous task.

Implement:

- Plot the latent state trajectory X_t along with the noisy observations Y_t in a single figure.
- Clearly distinguish between the true state and the observations (e.g., color, line style).

Discuss:

- Briefly comment on the plot.
- *Hint:* How do the noisy observations compare to the true state?

In many real-world applications, the true state of a system is not directly observed and only noisy measurements are available. This complicates parameter estimation, as classical methods like ordinary least squares are no longer directly applicable.

In the following tasks, you will explore how to estimate parameters in such partially observed models using the Kalman filter and the maximum likelihood framework.

1.3. In Chapter 10 of the course book, and as introduced in the lecture slides, we estimate parameters in a state-space model by maximizing the (log-)likelihood of the observations. This approach requires computing the predicted density of each observation Y_t, based on past observations and the model parameters.

Implement:

- Download and complete the function `myKalmanFilter(y, theta, R, X0, P0)`. It should implement the Kalman filter and return the following:
  - Predicted state mean $\hat{X}_{t+1|t}$ and variance $\hat{\Sigma}^{xx}_{t+1|t}$,
  - Innovation $Y_t - C\hat{X}_{t+1|t}$ and its variance $\hat{\Sigma}^{yy}_{t+1|t}$,
  - Filtered state mean $\hat{X}_{t|t}$ and variance $\hat{\Sigma}^{xx}_{t|t}$.
  - *Note:* In this exercise we are dealing with 1-d variables, so don't implement matrix operations.
- Apply your Kalman filter implementation to the time series simulated in previous question.
- Plot the following in a single figure:
  - The latent state X_t (ground truth),
  - The observation Y_t,
  - The predicted state $\hat{X}_{t+1|t}$,
  - A 95% confidence interval around $\hat{X}_{t+1|t}$.

Discuss:

- Comment on your findings.
- *Hint:* Do the predicted distributions match the simulated data well?
- *Hint:* Does the filter successfully track the hidden state despite the noise?

1.4. You are now ready to estimate parameters from noisy observations using the maximum likelihood framework.

Implement:

- Download and complete the function `myLogLikFun(theta, Y, R, X0, P0)`, which:
  - Takes as input: parameter vector $\theta = (a, b, \sigma_1)$, observations Y, observation noise variance Σ_2 called (called `R` in code), and initial values X_0, $\Sigma_{xx,0}$ (called `P0` in code),
  - Runs the Kalman filter from previous question,
  - Returns the negative log-likelihood of the observed data under the model.
- Use this function in combination with an optimization routine (e.g., `optim` in R) to estimate the parameters by minimizing the negative log-likelihood.
- Simulate 100 independent realizations of the system for $n = 100$ time steps, using the parameters $a = 1$, $b = 0.9$, and $\sigma_1 = 1$.
- Estimate parameters from each realization and summarize the results using appropriate plots, such as boxplots of the estimated values.
- Repeat the estimation for the following parameter combinations:
  - $a = 5$, $b = 0.9$, $\sigma_1 = 1$
  - $a = 1$, $b = 0.9$, $\sigma_1 = 5$

Discuss:

- Compare the resulting estimates across cases.
- *Hint:* Do the estimated parameters align with the true values?
- *Hint:* Are some parameters harder to estimate than others?

1.5. So far, you have assumed that both system and observation noise are Gaussian. In this exercise, you will challenge this assumption by simulating system noise from a Student's t-distribution instead.

Implement:

- Simulate the following model 100 times, each for $n = 100$ time steps:

$$X_t = aX_{t-1} + b + \sigma_1 \cdot \lambda_t, \quad \lambda_t \sim t(\nu),$$

$$Y_t = X_t + e_{2,t}, \quad e_{2,t} \sim \mathcal{N}(0, \sigma_2 = 1),$$

  using the parameter values $a = 1$, $b = 0.9$, $\sigma_1 = 1$ (note: in this case, we use σ_1 to scale the noise). Try several values of ν (degrees of freedom): $\nu = 100, 5, 2, 1$.
- Plot the density of the t-distribution for each ν alongside the standard normal distribution.

Disucss:

- Comment on the differences in the tails.
- *Hint:* How might this affect the Kalman filtering results?

Implement some more:

- For each value of ν, apply your previous estimation pipeline (Kalman filter + likelihood optimization) to the simulated data — even though the model assumes Gaussian noise.
- Summarize the estimated parameters for each case using plots (e.g., boxplots). Compare to the estimates from the Gaussian case.

Discuss some more:

- Comment on the differences in the estimates.
- *Hint:* How do the parameter estimates change as the tails become heavier (i.e. as ν decreases)?
- *Hint:* How does it relate to estimated variances of either student-t or Gaussian distribution?
- *Hint:* Does the Kalman filter still perform well?
- *Hint:* What does this imply about robustness of your model assumptions in practice?

2 Parameter estimation in state-space model

In this exercise, we shall develop models to predict the temperature evolution in a transformer station in a DSO (distribution system operator, distributing electricity from high-voltage grid out to the end consumer) grid. Transformer stations are vital to the electricity transportation system, and having the operate optimally is advantagous.

A brief introduction to a transformer station. Roughly speaking, a transformer station receives electricity at a high voltage and sends out electricity at a lower voltage. In this process, some energy is lost in the transformer and converted into heat. DSO transformer stations typically use oil for cooling (like water cooling, just with oil). The highest temperature inside the transformer station needs to be below a given threshold in order to limit the speed of degradation of the transformer station and to avoid it catching fire. In this data, a good approximation to this temperature is measured. We would now like a model that is able to predict this temperature.

This is useful because we will be able to predict dangerous temperatures before they happen, and because we will be able to relate the input (electricity load) to the output (temperature).

The data in this exercise contains:

- t: Time in hours (h)
- Y_t: The transformer station temperature (°C).
- $T_{a,t}$: The outdoor air temperature (°C).
- $\Phi_{s,t}$: The horizontal global solar radiation (W/m^2).
- $\Phi_{I,t}$: The load on the transformer station (kA).

Answer the following:

2.1. **Exploratory analysis of the data.**

Implement:

- Load the provided dataset.
- Plot all four variables (Y_t, $T_{a,t}$, $\Phi_{s,t}$, $\Phi_{I,t}$) together in a single figure, using different colors or panels.
- Use appropriate labels and time axes for clarity.

Discuss:

- Briefly comment on patterns, trends, or correlations you observe.
- *Hint:* Look for relationships between temperature, outdoor conditions, and load.

2.2. **Estimate a 1-dimensional state-space model.**

We now aim to fit a simple 1-dimensional state-space model of the form:

$$X_{t+1} = AX_t + Bu_t + e_{1,t},$$

$$Y_t = CX_t + e_{2,t},$$

where $u_t = [T_{a,t}, \Phi_{s,t}, \Phi_{I,t}]^T$ is the input vector.

You will use the provided R functions in `functions_exercise2.R`, but you must fill in missing parts.

Implement:

- Complete the functions `kf_logLik_dt` and `estimate_dt`.
- Map the parameter vector to the appropriate matrices A, B, C, Σ_1, Σ_2.
- Estimate the model parameters by maximizing the likelihood (use `optim`).
- *Hint:* You need to set starting values (`start_par`) and lower and upper bounds (`lower` and `upper`), for the estimation. Starting values between -1 and 1 should work for the parameters, the initial a state value around 20 (try with different values and observe if the fit converge and if the estimates are very different).
- *Hint:* Remember that variance parameters must be positive, thus setting the lower bound close to zero is necessary.
- *Hint:* If the observation variance hits a lower bound very close to zero, it's ok here!

Discuss:

- Present the estimated parameters and model performance (plots of residuals, ACF/PACF, QQ-plot, AIC and BIC values).
- Comment on the model fit:
  - Does the model capture the temperature dynamics satisfactorily?
  - Are there periods where the model performs poorly (e.g., daytime peaks)?
  - What do the estimated parameters tell you about the influence of load, solar radiation, and outdoor temperature?
- *Hint:* Consider the physical meaning of the parameters (e.g., effect of load on temperature rise).

2.3. **Expand to a 2-dimensional state-space model.**

Next, extend your model to allow for two latent states. The system equations are the same, but X_t is now a 2-dimensional state vector and the dimensions of the matrices increase.

Implement:

- Modify your Kalman filter and likelihood functions to work with 2-dimensional states and inputs.
- Make sure all matrices have correct dimensions (e.g., $A \in R^{2\times 2}$, $B \in R^{2\times 3}$).
- Estimate the model parameters using maximum likelihood optimization (`optim`).
- Adjust parameter starting values and bounds as needed.(*Hint:* you should either estimate the intial condition X_0 or insert a qualified guess in the Kalman filter.)
- *Hint:* The time running the estimation might be high (many iterations). This is ok, since often, with many parameters the optimizer can spend a long time in the last steps of fine tuning the parameters. Sometimes, a trick is to run the optimization, and if the max iterations are reached, then take the values and use them as starting values for a new run.
- Perform residual analysis:
  - Plot residuals,
  - Plot ACF and PACF,
  - Create QQ-plot,
  - Report AIC and BIC.

Discuss:

- Compare the 2D model's performance to the 1D model.
- *Hint:* Has the residual autocorrelation decreased? Does the model track temperature spikes better? (look at the 1-step predictions to get a feeling of this)
- Reflect on the interpretability of the two latent states. Can you guess what each state might represent physically?
- *Hint:* Consider whether one state could act as a "buffer" for temperature dynamics.

2.4. **Interpret the two reconstructed states.**

Having estimated a 2-dimensional model, you will now explore the meaning of the two latent states.

Implement:

- Plot the two reconstructed state trajectories over time in a single figure.
- Additionally, plot the input variables ($T_{a,t}$, $\Phi_{s,t}$, $\Phi_{I,t}$) alongside the states to help interpret their influence.
- Analyze the estimated A and B matrices to assist with interpretation.

Discuss:

- Propose a physical interpretation for each state.
- *Hint:* Could one state represent the core transformer temperature and the other a cooling or buffer effect?
- Discuss how the inputs affect each state according to the estimated parameters.
- Reflect on whether the model structure makes physical sense based on your plots and the parameter signs/magnitudes.