Time Series Models: From Statistics to AI

lecturer: Daniel Durstewitz
tutors: **Lukas Eisenmann**, Christoph Hemmer, Alena Brändle, Florian Hess, Elias Weber, Florian Götz
SS2025

Exercise Sheet 4

Regulations

Please submit your solutions via Moodle in teams of **2 students**, before the exercise group on **Wednesday, May 21st, 2025**. Each submission must include **exactly** two files:

- A `.pdf` file containing both your Jupyter notebook and solutions to analytical exercises. The Jupyter notebook can be exported to pdf by selecting **File → Download as → pdf** in JupyterLab. If this method does not work, you may print the notebook as a pdf instead. Your analytical solutions can be either scanned handwritten solutions or created using LaTeX.
- A `.ipynb` file containing your code as Jupyter notebook.

Both files must follow the naming convention:
`Lastname1-Lastname2-sheet04.pdf`
`Lastname1-Lastname2-sheet04.ipynb`

Task 1. Kalman filter-smoother
In this exercise sheet we will implement and apply the Kalman filter-smoother algorithm. The exercise contains references to ”Supplemental Script Ch7”, which you can download on the lecture Moodle page.

Attached to the exercise sheet you will find the zip-file `data.zip` containing variables 'z', '$\boldsymbol{x}$', '$\boldsymbol{A}$', '$\boldsymbol{B}$', '$\boldsymbol{C}$', '$\boldsymbol{u}$', '$\boldsymbol{\Sigma}$' and '$\boldsymbol{\Gamma}$' as `.csv`-files, which specify a model as given in (supplemental script Ch7, eq. 7.53) with an additional control variable term, as described below.

The variables represent a linear dynamical system, where the state of the system (in this case a drone) is contained in the multivariate vector variable z of size $p \times T$. Here $p = 2$ specifies the number of latent states and $T = 100$ the number of time points. The states correspond to the position (height) and velocity of the drone. In addition, the drone is controlled by externally (i.e. through a controller) contained in variable u. The controlled linear state space model is given by

$$
\begin{aligned}
z_t &= \boldsymbol{A} z_{t-1} + \boldsymbol{C} u_t + \boldsymbol{\epsilon}_t & \boldsymbol{\epsilon}_t &\sim \mathcal{N}(0, \boldsymbol{\Sigma}) \\
\boldsymbol{x}_t &= \boldsymbol{B} z_t + \boldsymbol{\eta}_t & \boldsymbol{\eta}_t &\sim \mathcal{N}(0, \boldsymbol{\Gamma})
\end{aligned}
$$

Here A is the transition matrix, C marks the influence of the control variables $\{u_t\}$ on the state, and $\boldsymbol{\Sigma}$ is the state noise covariance matrix (e.g. air current). Assume the states are directly observable, but instead only noise and mixed measurements from the position and the velocity of the drone are available, contained in vector variable $\boldsymbol{x}$ (with rows 1–2 corresponding to a mixture of latent states z).

Note: To read the `.csv` into Python you can use numpy's `np.genfromtxt("FILENAME.csv", delimiter=",")`.

1. Retrieve an estimate for the latent states z purely from the observations $\boldsymbol{x}$ by implementing the Kalman filter recursion. To this end, use equation (7.63) to implement the 'filter' going forward in time from $t = 1 \ldots T$. Make sure to include the extracted control variable term by changing the top row of (7.63) to:

   $$\boldsymbol{\mu}_t = \boldsymbol{A}\boldsymbol{\mu}_{t-1} + \boldsymbol{C}u_t + \boldsymbol{K}_t(\boldsymbol{x}_t - \boldsymbol{B}(\boldsymbol{A}\boldsymbol{\mu}_{t-1} + \boldsymbol{C}u_t))$$

   For $t = 1$, use $\boldsymbol{\mu}_0$ and $\boldsymbol{L}_0$ — also provided in the files — as initial conditions for latent states and state covariance matrix. Use that $\boldsymbol{L}_t = \boldsymbol{A}\boldsymbol{V}_{t-1}\boldsymbol{A}^T + \boldsymbol{\Sigma}$. And plug the expression in the other equations from (7.63) from the script.

2. Plot the obtained predicted latent states against the true latent states (in variable z). How well can you recover the true latent states? What could the drone be doing?

3. Examine the parameter values of $\boldsymbol{A}$ and $\boldsymbol{C}$. Why are they chosen the way they are?

4. Now implement the Kalman smoother.
   Use eq. (7.70) to implement the Kalman smoother going backwards from $t = T$, to $t = 1$. Note that $\boldsymbol{\mu}_T = \tilde{\boldsymbol{\mu}}_T$ and $\boldsymbol{V}_T = \tilde{\boldsymbol{V}}_T$ for $t = T$.
   The Kalman smoother equations for this model are given by:

   $$\tilde{\boldsymbol{V}}_t = \boldsymbol{V}_t + \boldsymbol{V}_t\boldsymbol{A}^T\boldsymbol{L}_t^{-1}(\tilde{\boldsymbol{V}}_{t+1} - \boldsymbol{L}_t)\boldsymbol{L}_t^{-T}\boldsymbol{A}\boldsymbol{V}_t^T$$
   $$\tilde{\boldsymbol{\mu}}_t = \boldsymbol{\mu}_t + \boldsymbol{V}_t\boldsymbol{A}^T\boldsymbol{L}_t^{-1}(\tilde{\boldsymbol{\mu}}_{t+1} - (\boldsymbol{A}\boldsymbol{\mu}_t + \boldsymbol{C}u_t))$$

5. Apply the Kalman filter-smoother to obtain estimates for your latent state path. Plot the obtained predicted latent states against the true latent states (variable `'z'`), and against the latent states obtained by only using the filter. Quantify these results by computing the mean squared error (MSE) between the true and recovered latent states by both Kalman filter and Kalman filter-smoother. How do they differ?