e

Time Series Analysis & Recurrent Neural Networks

Lecturer: Daniel Durstewitz
Tutors: Georgia Koppe, Manuel Brenner, Daniel Kramer, Leonard Bereska
WS2020/2021

Exercise 11

To be uploaded before the exercise group on 7th July, 2021.

Task 1: Laplace Approximation

$$\int e^{Mf(x)} = \sqrt{\frac{2\pi}{M|f''(x_0)|}} e^{Mf(x_0)} \text{ as } M \to \infty, \text{with } x_0 \text{ as the global maximum of } f. \tag{I}$$

Apply Laplace's method (Eq. I) to approximate the integral $N! = \int_0^\infty e^{-t} t^N dt$.

Task 2: Extended Kalman Filter

In file `par.pkl`, you are given matrices $\mathbf{A}$, $\mathbf{W}$, $\mathbf{B}$, Γ, Σ, and vector μ_0, which specify the parameters of the following non-linear recurrent neural network:

$$\mathbf{z}_t = f(\mathbf{z}_{t-1}, \eta_t) = \mathbf{A}\mathbf{z}_{t-1} + \mathbf{W}\phi(\mathbf{z}_{t-1}) + \mathbf{h} + \epsilon_t, \quad \epsilon_t \sim N(0, \Sigma) ,$$

where $\phi(\mathbf{z}_t) = \max(\mathbf{z}_t, 0)$ is an (element-wise) piece-wise linear transfer function which returns each element of $\mathbf{z}_t$ if it crosses the threshold 0, or 0 otherwise. We will assume that observations drawn from this (latent) dynamical system are just a linear transformation of the latent states $\mathbf{z}_t$:

$$\mathbf{x}_t = g(\mathbf{z}_t, \eta_t) = \mathbf{B}\mathbf{z}_t + \eta_t, \quad \eta_t \sim N(0, \Gamma) .$$

Both our evolution and observation processes are noisy (that is, the noise follows a normal distribution with 0 mean and covariance Σ or Γ, respectively). The initial state estimate is μ_0.

1. Assuming the latent state dimension to be equal to $M = 3$, and the observation dimension to be $N = 10$, let this system run for $T = 500$ time steps and create "true" latent states $\{\mathbf{z}_t\}_{1:T}$, and observations $\{\mathbf{x}_t\}_{1:T}$, based on this system and the given parameters.

2. Run the Kalman filter that we have previously implemented (exercise 5) to obtain latent state estimates based on your created observations. *

3. Implement the extended Kalman filter (EKF), as described in (9.46) of the script. Use it to obtain latent state estimates based on the created observations.

4. Compare the obtained estimate of $\{\mathbf{z}_t\}_{1:T}$ between the Kalman filter and the EKF by quantifying the mean squared error (MSE) between true and estimated latent states for both methods.

*Use `kalmanfilter.py` if you have not already implemented it in exercise 5.