Stochastic Systems 2

Lukas Deria Lewe Düzakin

May 2025

Introduction

This is the second mandatory assignment. This assignment consists only of simulation results. These simulations have been completed in Python.

Since the code for this assignment is divided into several files, a ZIP archive containing all scripts will be attached to the email. If, for any reason, this is not the desired format or the attachment does not come through correctly, please let me know.

Task 2

To discretise the system, we need to convert it from its continuous form:

$$\dot{x} = Fx + Lu + Gv, \quad v \sim \mathcal{N}(0, Q\delta(t-\tau))$$

to the discretised form:

$$x_{k+1} = \Phi x_k + \Lambda u_k + \Gamma v_k, \quad v_k \sim \mathcal{N}(0, \delta_{kl} Q)$$

This requires calculating the matrices Φ, Λ, Γ, and S. This is done using Van Loan's method, and the computations are performed in Python. The results are:

$$\Phi = \begin{bmatrix} 1.000 & 9.99 \cdot 10^{-3} & 9.96 \cdot 10^{-6} \\ 0.000 & 9.98 \cdot 10^{-1} & 1.99 \cdot 10^{-3} \\ 0.000 & 0.000 & 9.90 \cdot 10^{-1} \end{bmatrix} \quad \Lambda = \begin{bmatrix} 3.32 \cdot 10^{-8} \\ 9.96 \cdot 10^{-6} \\ 9.95 \cdot 10^{-3} \end{bmatrix}$$

$$\Gamma = \begin{bmatrix} 9.93 \cdot 10^{-9} & 4.07 \cdot 10^{-8} & 4.73 \cdot 10^{-8} \\ -1.22 \cdot 10^{-7} & 8.13 \cdot 10^{-6} & 1.42 \cdot 10^{-5} \\ 3.29 \cdot 10^{-7} & -4.24 \cdot 10^{-5} & 1.42 \cdot 10^{-2} \end{bmatrix}$$

$$S = \begin{bmatrix} 6.67 \cdot 10^{-16} & -3.33 \cdot 10^{-13} & 6.66 \cdot 10^{-10} \\ 3.33 \cdot 10^{-13} & -1.33 \cdot 10^{-10} & 2.00 \cdot 10^{-7} \\ 6.67 \cdot 10^{-10} & -2.00 \cdot 10^{-7} & 2.00 \cdot 10^{-4} \end{bmatrix}$$

Task 3

In this task and the following tasks, a simplification was made in the code. Instead of running the simulation multiple times, this task and Task 4 are solved using a single simulation. This simulation includes the deterministic part, the stochastic part (forming the ground truth for the Kalman filter), as well as the Kalman filter itself. This is evident from the code. This approach simplifies the code and makes it more modular.

The plot shows the evolution of the second state, i.e., the velocity of the system, when simulated using both the deterministic and the stochastic parts. The deterministic part refers to the system without the added modelling noise.

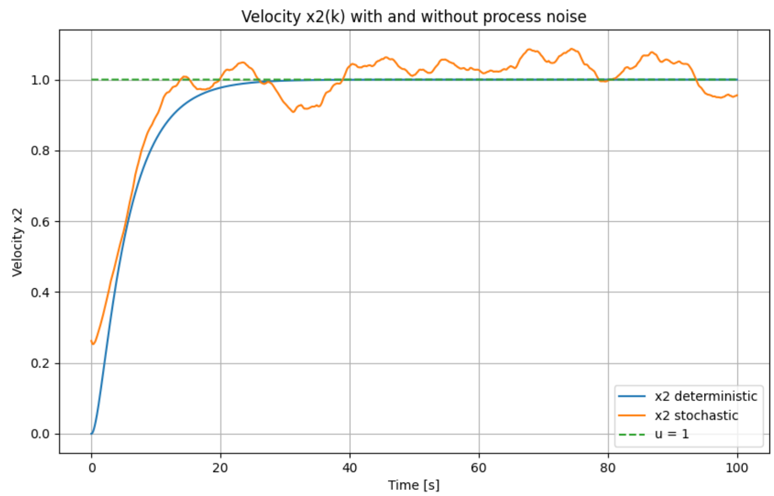

Figure 1: Plot of deterministic and stochastic simulation under the influence of a constant input.

It is evident from the plot that the added noise causes the system to approach the desired state but eventually diverge, which is not the case for the deterministic part.

Task 4

In this task, we reuse the stochastic part from the previous task as the foundation for the Kalman filter. To perform the filtering, we need measurements. These measurements must be sampled or generated. In the code, this is done by extracting the position at a given rate from the ground truth simulation. Measurement noise is then added to provide a more realistic representation.

The following plots show the same stochastic simulation as in the previous task, but also include the filtered velocity of the system. Additionally, the second and third subplots show the difference between the true state and the filtered and predicted velocity, respectively.

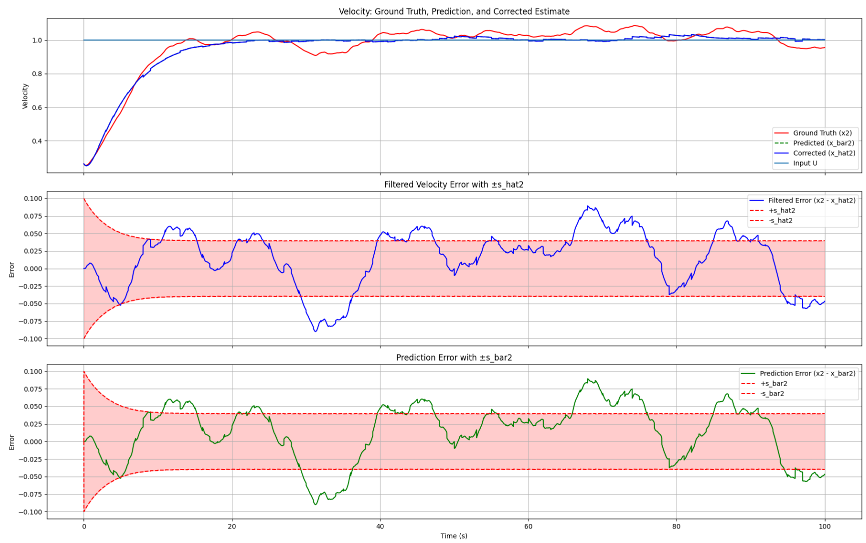

Figure 2:

Due to the high model update rate and the low frequency of sensor readings, the filtered and predicted outputs are identical in 99% of the time steps. This is expected, as the filtered output depends on both the predicted state and the measurements. However, when no measurement is available, as is the case 99 out of 100 time steps, the filtered output is equal to the predicted state. As a result, the two curves appear to overlap to the naked eye in the plot. Nevertheless, if the simulation is examined closely, a small difference is observable at the time instants when new measurements are available.

This observation raises the question of whether the system truly benefits from the addition of a sensor or if the increased complexity of integrating a sensor outweighs the potential advantages.

Task 5

For Task 5, the same simulation, system, and parameters as before are used in a new setting. The system is simulated N times, each run producing slightly different results due to the randomness in the process and measurement noise. This allows us to evaluate how the Kalman filter performs over multiple realisations.

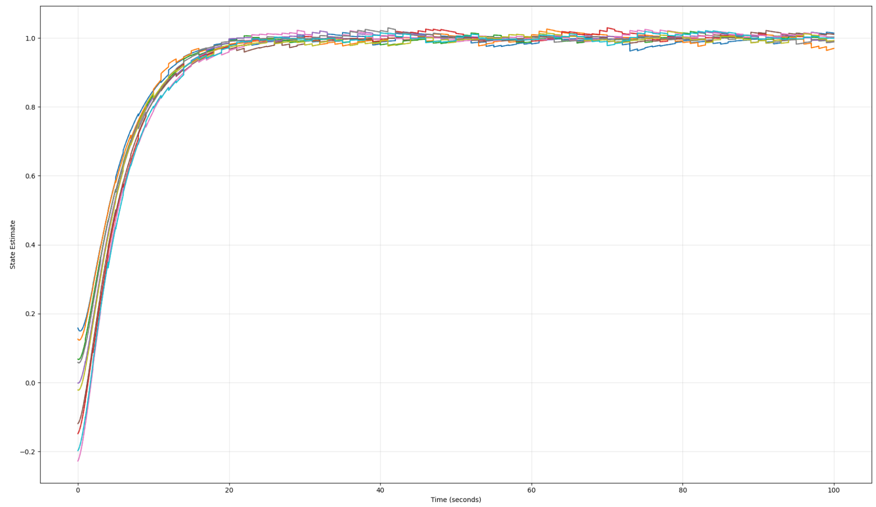

Figure 3: Total of 10 realisations

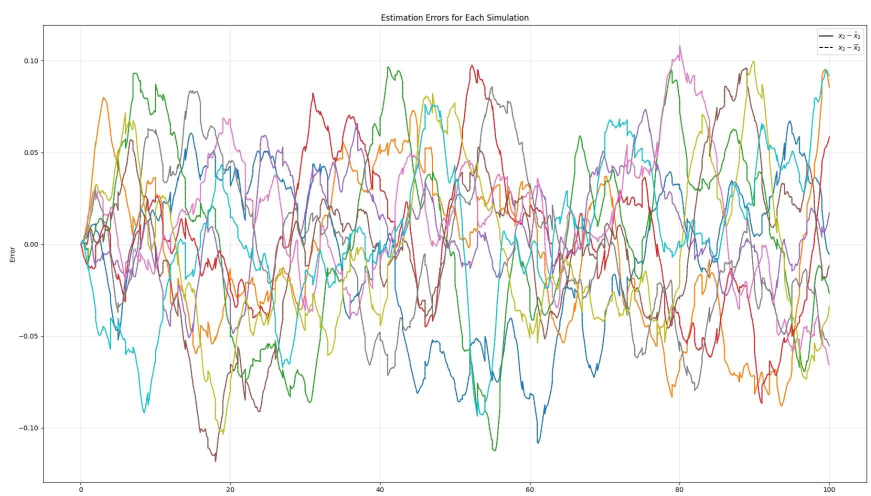

Figure 4: Plot of the errors for each realisation

The two plots above show several realisations as well as the error between the ground truth and the filtered output and the predicted output. It is again evident that the differences between the predicted and filtered outputs are small and that all realisations are close to each other.

Figure 5: Velocity error statistic for 10 realisations

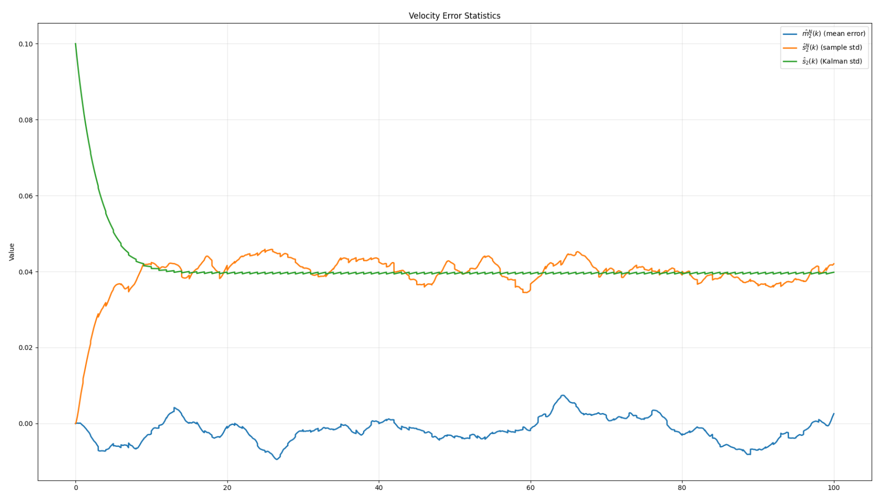

Figure 6: Velocity error statistic for 100 realisations

the next two plots show the mean velocity estimation error $\hat{m}_2^N(k)$, the sample standard deviation $\hat{s}_2^N(k)$, and the Kalman filter's predicted standard deviation $\hat{s}_2(k)$ for $N = 10$ and $N = 100$, respectively. In both cases, the Kalman standard deviation starts high and quickly converges to a steady-state value. For $N = 10$, the sample standard deviation exhibits noticeable fluctuations due to the limited number of Monte Carlo runs but still aligns well with the Kalman prediction. The mean error oscillates around zero, indicating the filter is approximately unbiased.

In contrast, for $N = 100$, the sample standard deviation becomes much smoother and closely matches the Kalman prediction throughout, demonstrating the consistency of the Kalman filter. The mean error remains small and centred around zero, further confirming the unbiased nature of the estimator and the reliability of the predicted covariance.

Task 6

In this task, we evaluate the error sources affecting each state, i.e., how randomness and noise impact the different states individually.

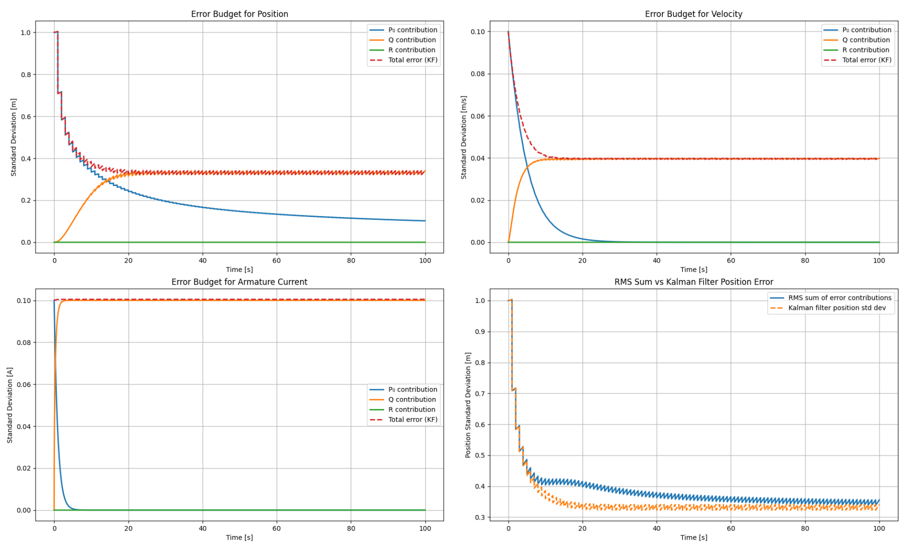

Figure 7:

The error budget plots illustrate how the initial uncertainty P_0 decays over time, while the process noise Q and measurement noise R dominate the steady-state behaviour. Each state shows a different balance of these contributions, with current being mostly influenced by R. In the lower right plot, the RMS sum of the individual error contributions slightly overestimates the Kalman filter's position standard deviation, which may indicate numerical artefacts or approximation errors in the decomposition.

Task 7

In the last task, we are exploring the effects of a reduced system model on the uncertainty and performance of the Kalman filter's ability to estimate the correct state. We are given a system where the current is approximated as noise.

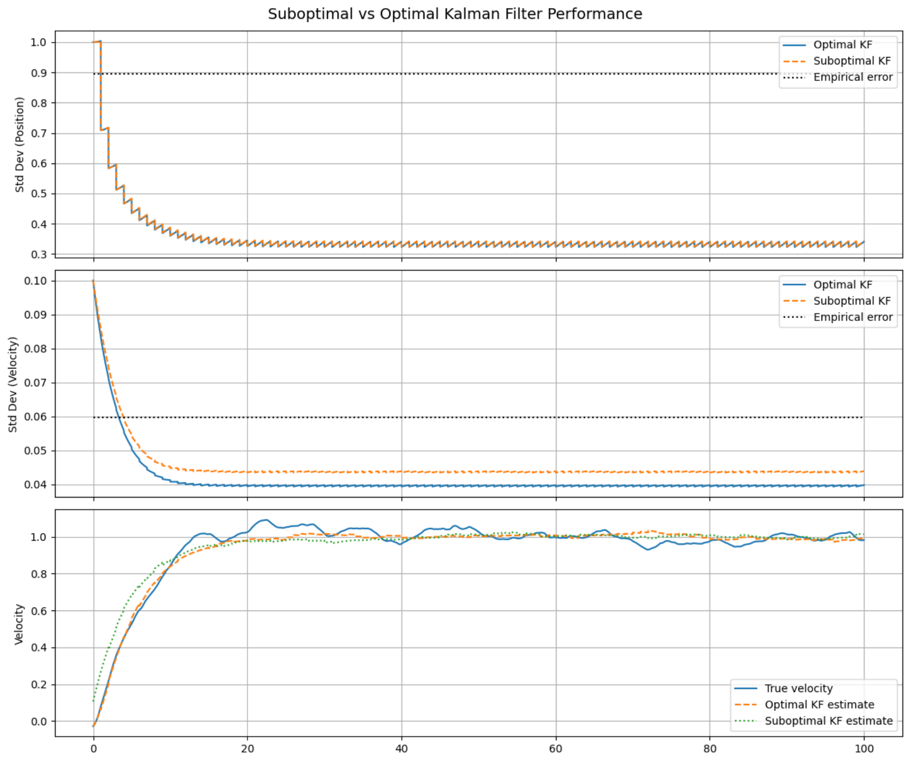

Figure 8:

In the figure above, we observe that the standard deviation, representing the uncertainty of the measured state (position), does not vary between the reduced and original systems. This is expected, as the position is the measured state in both systems.

However, what becomes more evident is the increased uncertainty in the velocity for the reduced system. This aligns with the fact that system reduction inherently results in a loss of information. Specifically, since the armature current is treated as noise in the reduced system, we lose a dimension, which leads to a higher uncertainty in the velocity estimate.

The third plot demonstrates that despite this increased uncertainty, the Kalman filter still performs on par with the optimal filter, at least for this system.

Note:
After completing the assignment, I realised that I did not use the correct augmented matrices for the suboptimal Kalman filter. Due to time constraints related to the national day, I was not able to redo the task. However, I believe it is reasonable to assume that the results — even if based on an incorrect formulation — still reflect the relative performance of the Kalman filters.