From Classical Filters to Particle Flows

Report of JP Morgan MLCOE TSRL 2026 Internship

Question 2

Yaxin Feng

November 24, 2025

1 Warm-up.

For those who are not familiar with filtering, do spend some time learning these basics before the next part.

1.1 Linear-Gaussian SSM with Kalman Filter

a) Implement the Kalman filter for a multidimensional linear-Gaussian State Space Models (SSM).
Use synthetic data from a standard LGSSM, see examples 2 in [Doucet(09)] [9].

Example 2 - *Linear Gaussian model.* Here, $\mathcal{X} = \mathbb{R}^{n_x}$, $\mathcal{Y} = \mathbb{R}^{n_y}$, $X_1 \sim \mathcal{N}(0, \Sigma)$ and

$$X_n = AX_{n-1} + BV_n,$$
$$Y_n = CX_n + DW_n$$

where $V_n \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, I_{n_v})$, $W_n \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, I_{n_w})$ and A, B, C, D are matrices of appropriate dimensions. Note that $\mathcal{N}(m, \Sigma)$ denotes a Gaussian distribution of mean m and variance-covariance matrix Σ, whereas $\mathcal{N}(x; m, \Sigma)$ denotes the Gaussian density of argument x and similar statistics. In this case $\mu(x) = \mathcal{N}(x; 0, \Sigma)$, $f(x'|x) = \mathcal{N}(x'; Ax, BB^{\mathrm{T}})$ and $g(y|x) = \mathcal{N}(y; Cx, DD^{\mathrm{T}})$. As inference is analytically tractable for this model, it has been extremely widely used for problems such as target tracking and signal processing.

LGSSM. The LGSSM is defined by the State Equation and the Measurement Equation, assuming all noise is Gaussian.

I regard the constant velocity (CV) model as an example:

A. State Vector and Time Step

The state X_n is 4-dimensional, combining position (P) and velocity (V) in 2D, and Δt is the time step.

$$X_n = \begin{bmatrix} p_{x,n} \\ v_{x,n} \\ p_{y,n} \\ v_{y,n} \end{bmatrix} \in \mathbb{R}^4,$$

here are the recursion steps in mathematical form, repeating for $n = 1, 2, 3, \ldots, N$.

B. State Equation (Process Model)

This describes the object's motion (dynamics) and the effect of unmodeled acceleration noise.

$$X_n = AX_{n-1} + BV_n$$

State Transition Matrix (A) in the CV model is:

$$A = \begin{bmatrix} 1 & \Delta t & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \Delta t \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Process Noise Coupling Matrix (B): This matrix determines how the input noise V_n affects the state (it models piecewise constant acceleration):

$$B = \begin{bmatrix} \Delta t^2/2 & 0 \\ \Delta t & 0 \\ 0 & \Delta t^2/2 \\ 0 & \Delta t \end{bmatrix}$$

Process Noise (V_n): $V_n \sim \mathcal{N}(\mathbf{0}, I_2)$.

Process Noise Covariance (Q): Calculated as $Q = BB^T$

C. Measurement Equation (Observation Model)

This links the state X_n to the noisy sensor readings Y_n (position).

$$Y_n = CX_n + DW_n$$

Observation Matrix (C): Measures p_x and p_y only.

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Observation Noise (W_n): $W_n \sim \mathcal{N}(\mathbf{0}, I_2)$.

Observation Noise Coupling Matrix (D): an scaled identity matrix $D = I$ is used.

Observation Noise Covariance (R): Calculated as $R = DD^T$.

D. Initial State

The initial state X_1 is stochastic (random):

$$X_1 \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$$

The filter's starting values are the mean and covariance: $\hat{X}_{0|0} = \mathbf{0}$ and $P_{0|0} = \mathbf{\Sigma}$.

Kalman Filter. Kalman filter recursion steps is the process of iteratively calculating the optimal state estimate by alternating between two phases: Prediction (Time Update) and Update (Measurement Update).

1. Prediction Step (Time Update)

The Prediction step projects the posterior estimate from the previous time step, $\hat{\mathbf{X}}_{n-1|n-1}$ and $\mathbf{P}_{n-1|n-1}$, forward to the current time n. This results in a prior estimate, before the measurement $\mathbf{Y}_n$ is seen.

A. State Prediction (Prior Estimate)

The previous optimal state estimate is projected forward using the State Transition Matrix ($\mathbf{A}$).

$$\hat{\mathbf{X}}_{n|n-1} = \mathbf{A}\hat{\mathbf{X}}_{n-1|n-1}$$

B. Covariance Prediction

The previous error covariance is projected forward, and the uncertainty introduced by the Process Noise Covariance ($\mathbf{Q}$) is added.

$$\mathbf{P}_{n|n-1} = \mathbf{A}\mathbf{P}_{n-1|n-1}\mathbf{A}^T + \mathbf{Q}$$

2. Update Step (Measurement Update)

The Update step incorporates the new measurement, $\mathbf{y}_n$, to correct the prior estimate, resulting in the optimal posterior estimate.

A. Innovation Covariance

This step calculates the predicted covariance of the measurement residual ($\tilde{\mathbf{y}}_n$), using the Observation Matrix ($\mathbf{C}$) and the Observation Noise Covariance ($\mathbf{R}$).

$$\mathbf{S}_n = \mathbf{C}\mathbf{P}_{n|n-1}\mathbf{C}^T + \mathbf{R}$$

B. Kalman Gain

The Kalman Gain ($\mathbf{K}_n$) is calculated to determine how much the filter should trust the current measurement ($\mathbf{R}$) versus its own prediction ($\mathbf{P}_{n|n-1}$).

$$\mathbf{K}_n = \mathbf{P}_{n|n-1}\mathbf{C}^T\mathbf{S}_n^{-1}$$

C. State Update (Posterior Estimate)

The final state estimate is calculated by adding the weighted difference between the actual measurement ($\mathbf{Y}_n$) and the expected measurement ($\mathbf{C}\hat{\mathbf{X}}_{n|n-1}$) to the prior estimate. The difference ($\mathbf{Y}_n - \mathbf{C}\hat{\mathbf{X}}_{n|n-1}$) is known as the innovation or measurement residual.

$$\hat{\mathbf{X}}_{n|n} = \hat{\mathbf{X}}_{n|n-1} + \mathbf{K}_n(\mathbf{Y}_n - \mathbf{C}\hat{\mathbf{X}}_{n|n-1})$$

D. Covariance Update (Standard Version)

The error covariance is updated, reflecting the reduced uncertainty after the measurement is incorporated. ($\mathbf{I}$ is the identity matrix).

$$\mathbf{P}_{n|n} = (\mathbf{I} - \mathbf{K}_n\mathbf{C})\mathbf{P}_{n|n-1}$$

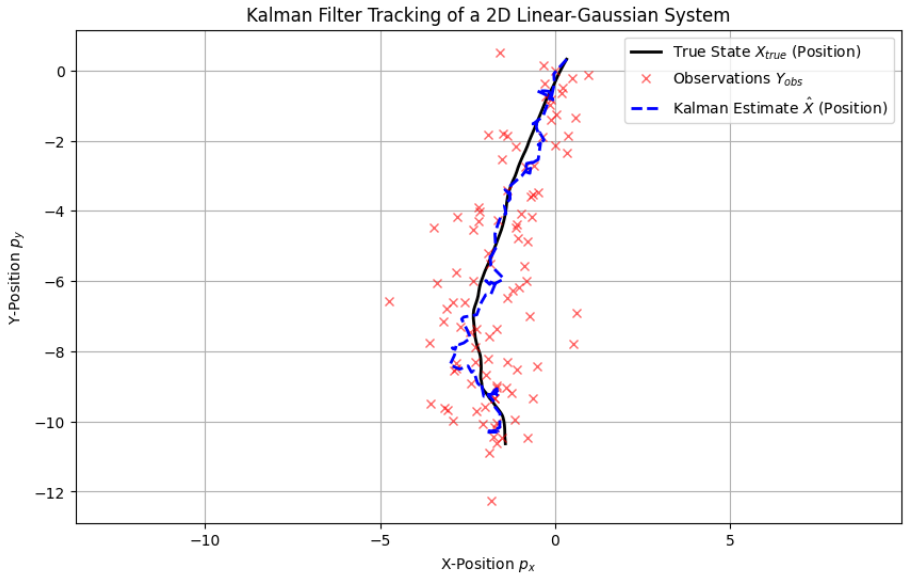

Figure 1: Experiment of Kalman Filter for CV model.

Experiment Result. After generating the synthetic data from the LGSSM, the Kalman Filter is applied for target tracking; see Fig. 1.

b) Analyze its filtering optimality and numerical stability: compare filtered means/covariances to the Kalman recursion; use Joseph stabilized covariance updates; discuss conditioning number.

The filtering optimality of the filtered means/covariances can be evaluated by the mean squared estimate error. The numerical stability can be analyzed by verifying the positive definiteness, examining the condition number of the estimation-error covariance matrix and providing limits on the numerical errors, according to [7]. In this brief report, the condition number of the $\mathbf{P}_{n|n}$ is discussed, which is a measure of how close it is to being singular (non-invertible). The higher the condition number, the worse the numerical stability.

The equation $\mathbf{P}_{n|n} = (\mathbf{I} - \mathbf{K}_n\mathbf{C})\mathbf{P}_{n|n-1}$ is the covariance update in standard Kalman Filter, while $\mathbf{P}_{n|n} = (\mathbf{I} - \mathbf{K}_n\mathbf{C})\mathbf{P}_{n|n-1}(\mathbf{I} - \mathbf{K}_n\mathbf{C})^T + \mathbf{K}_n\mathbf{R}\mathbf{K}_n^T$, is the Joseph Form. The Joseph Form is a better way to calculate $\mathbf{P}_{n|n}$ because it offers enhanced numerical stability and guaranties that the covariance matrix remains valid, which is crucial for the long-term operation of the filter.

As shown in Fig. 2, the mean and covariance between the standard and Joseph form of covariance update in the Kalman Filter. The resulting ellipse contour guarantees that the true state has a 95% probability of being found inside the boundary defined by this ellipse, based on the filter's estimate ($\hat{\mathbf{X}}_{n|n}$) and covariance ($\mathbf{P}_{n|n}$). The ellipses of these two forms look close, of which the size decreases during the recursion. This signifies a reduction in uncertainty regarding the filter's state estimate.

In the Fig. 3, the LGSSM with the state update equation $X_n = AX_{n-1} + BV_n, B = I$ with a log-scale y-axis, the condition number $\kappa(\mathbf{P}_{n|n})$ peaks at approximately 10^5, and stables at $\kappa(\mathbf{P}_{n|n})$

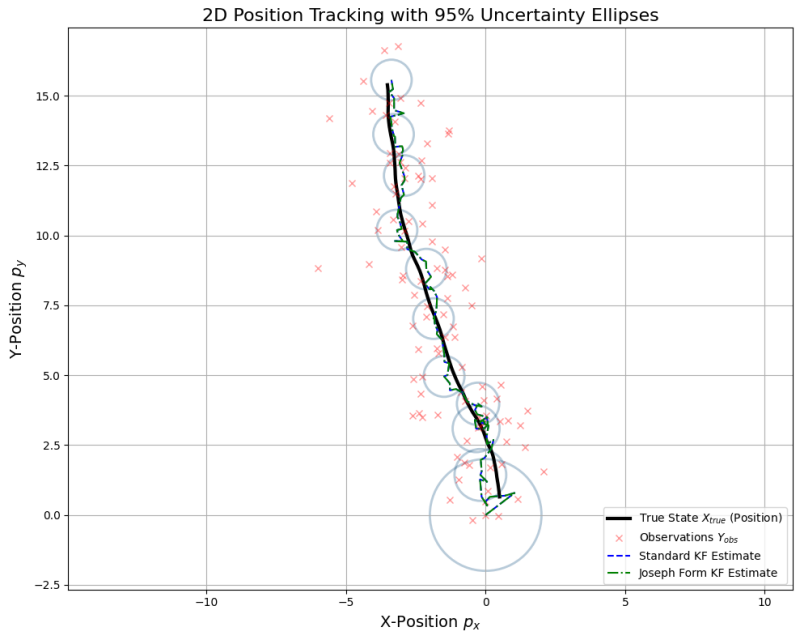

Figure 2: Comparison of the mean/covariance during the standard/Joseph Kalman Recursion.

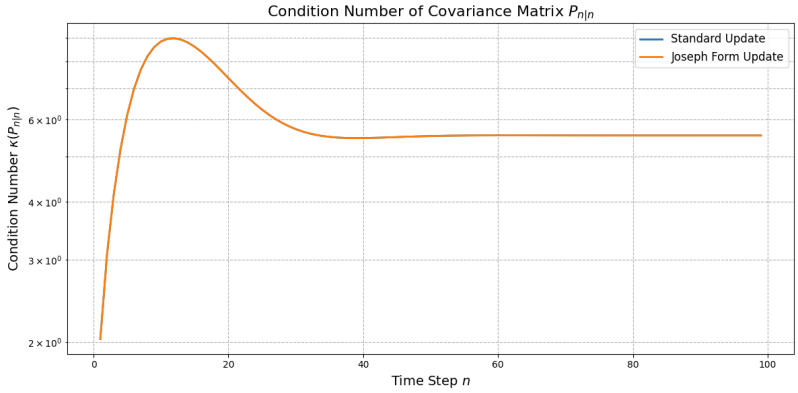

Figure 3: Comparison of condition number of $\mathbf{P}_{n|n}$ during the standard/Joseph Kalman Recursion with different q_{factor}.

stabilizes at approximately 10^4. This shows that the joseph form of covariance update is similar to standard form in this example.

In the Fig. 4, the LGSSM with a scaled matrix $\tilde{B} = \sqrt{q_{\text{factor}}} I, \tilde{Q} = q_{\text{factor}} Q$ is used. In this situation, the condition number explodes. The reason the condition number explodes (increases dramatically) when the q factor drops from 1 to 0.0001 is due to the severe imbalance in the estimated variances of the state variables, specifically the ratio between position uncertainty and velocity uncertainty.

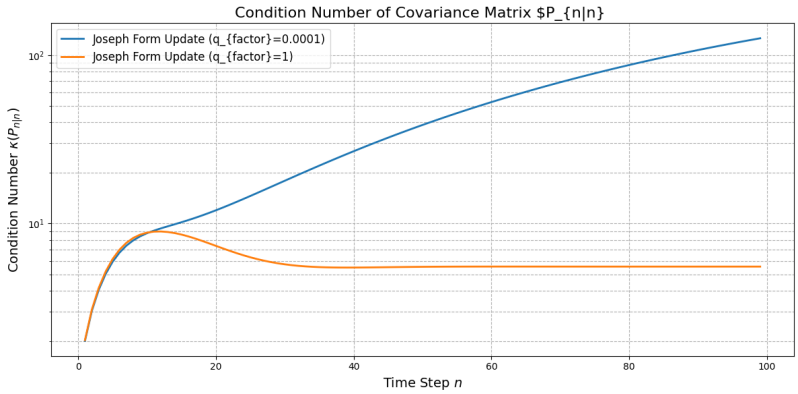

Figure 4: Comparison of condition number of $\mathbf{P}_{n|n}$ during the Joseph Kalman Recursion with different q_{factor}.

Remark: Later, I will do some more comprehensive experiments to discuss different LGSSMs, and the numerical stability and filter accuracy on the LGSSM.

1.2 Nonlinear/Non-Gaussian SSM with EKF/UKF and Particle Filter

a) Design a nonlinear and non-Gaussian SSM. A stochastic volatility model (SV model) is used; see example 4 in [Doucet(09)] [9].

Example 4 - *Stochastic Volatility model.* We have $\mathcal{X} = \mathcal{Y} = \mathbb{R}$, $X_1 \sim \mathcal{N}\left(0, \frac{\sigma^2}{1-\alpha^2}\right)$ and

$$X_n = \alpha X_{n-1} + \sigma V_n,$$
$$Y_n = \beta \exp\left(X_n/2\right) W_n$$

where $V_n \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0,1)$ and $W_n \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0,1)$. In this case we have $\mu(x) = \mathcal{N}\left(x; 0, \frac{\sigma^2}{1-\alpha^2}\right)$, $f(x'|x) = \mathcal{N}\left(x'; \alpha x, \sigma^2\right)$ and $g(y|x) = \mathcal{N}\left(y; 0, \beta^2 \exp(x)\right)$. Note that this choice of initial distribution ensures that the marginal distribution of X_n is also $\mu(x)$ for all n. This type of model, and its generalisations, have been very widely used in various areas of economics and mathematical finance: inferring and predicting underlying volatility from observed price or rate data is an important problem. Figure 1 shows a short section of data simulated from such a model with parameter values $\alpha = 0.91, \sigma = 1.0$ and $\beta = 0.5$ which will be used below to illustrate the behaviour of some simple algorithms.

Experiment Result. The replicated result can be seen in the Fig. 5.

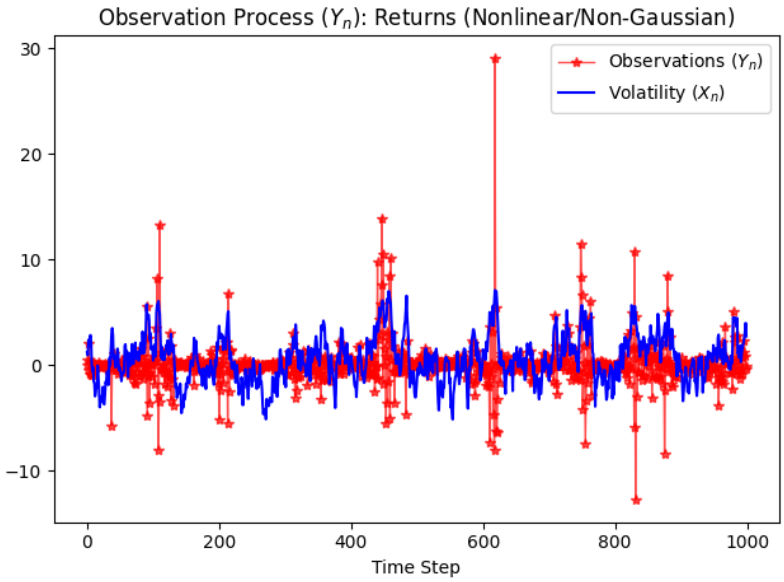

Figure 5: Simulation of the SV model of example 4 in [9].

b) Implement the Extended Kalman filter (EKF) and Unsent Kalman Filter(UKF), discuss linearization accuracy limits and sigma point failures under strong nonlinearity.

Experiment Result. The result of Extended Kalman Filter (EKF) and Unscented Kalman Filter (UKF) are shown in the Fig. 6.

EKF is a direct extension of the linear Kalman Filter [12]. It addresses nonlinear dynamical systems by employing analytical linearization via a first-order Taylor series expansion (jacobian). In non-linear systems, this introduces a linearization bias. The true mean $E[h(\mathbf{X})]$ is generally not equal to the function evaluated at the mean $h(E[\mathbf{X}])$ [2].

Linearization Accuracy Limits: The main accuracy limitation of the Extended Kalman Filter (EKF) is its reliance on a first-order Taylor series approximation, which introduces errors in highly nonlinear systems.

In the SV model: While the transformed observation $\mathbf{Z}_n$ is linearly dependent on $\mathbf{X}_n$, the initial non-linear step ($\mathbf{Y}_n^2 \to \ln(\mathbf{Y}_n^2)$) forces you to use Gaussian approximations for the noise ν (the mean b_ν and variance R_{cov}). Any error in this initial approximation (e.g., $E[\ln(\mathbf{W}_n^2)] \neq \mu_\nu$) contributes directly to the EKF's ultimate bias and limits its accuracy. Besides, the EKF's calculate covariance ($\mathbf{P}_{k|k}$) only captures the uncertainty derived from the slope (Jacobian) at the mean, leading to an inaccurate representation of the true uncertainty (it usually underestimates the covariance).

UKF addresses nonlinearity by performing a statistical linear regression using a set of representative sample points. This method is called the Unscented Transform (UT) [10]. The UKF's accuracy depends heavily on how well the small set of sigma points can capture the behavior of the non-linear function over the spread defined by the current covariance $\mathbf{P}$.

Sigma Point Failures under Strong Nonlinearity: If the non-linear function is highly curved (high second or third derivatives) within the region spanned by the sigma points (e.g., if

$\mathbf{P}_{k|k-1}$ is very large), the UKF's weighted sum of the transformed points may still fail to accurately capture the true mean and covariance of the transformed density [11].

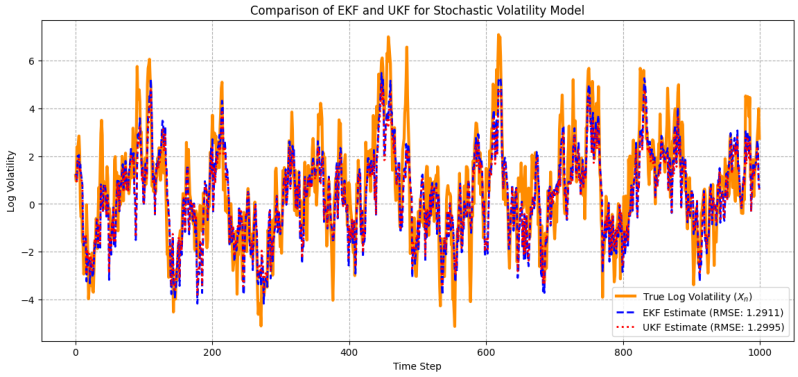

Figure 6: Comparison of the SV model in [9] by EKF and UKF.

The main accuracy limitation of the Extended Kalman Filter (EKF) is its reliance on a first-order Taylor series approximation, which introduces errors in highly nonlinear systems.

c) Implement a standard particle filter for your model. Visualize and discuss issues such as particle degeneracy.

Experiment Result. The result of Particle Filter (PF) is shown in the Fig. 7. The Root Mean Square Error (RMSE) of the result by PF is lower than those by the EKF and UKF methods.

Degeneracy Problem: In a standard Particle Filter (PF), after multiple cycles of prediction and update, the distribution of weights often becomes heavily skewed. Most of the probability mass concentrates on a single particle (or a very small group).

The Effective Sample Size (ESS) is the primary tool used to detect weight degeneracy. ESS is calculated using the normalized importance weights ($\tilde{w}_k^{(i)}$) of all N particles at time k:

$$\text{ESS} = \frac{1}{\sum_{i=1}^{N} \left(\tilde{w}_k^{(i)}\right)^2}$$

Where $\tilde{w}_k^{(i)}$ is the normalized weight of the i-th particle, such that $\sum_{i=1}^{N} \tilde{w}_k^{(i)} = 1$.

The ESS ranges from 1 to N, and its value provides immediate feedback on the state of the particle set. When ESS $\approx N$, it means excellent health. All particles have roughly equal weights and contribute equally. When ESS ≈ 1 , it means critical health (Weight Degeneracy). Only one or a few particles have significant weight; all others are negligible.

The main purpose of calculating the ESS is to decide when to resample. It can play the role of resampling threshold: particle filters typically set a threshold (e.g., $\text{ESS}_{\text{threshold}} = N/2$, or a fixed value like 50). If the calculated ESS drops below this threshold, the algorithm triggers a resampling step.

In summary, an average ESS of 577.55 suggests the Particle Filter is well-tuned, and the proposal distribution is generating particles efficiently.

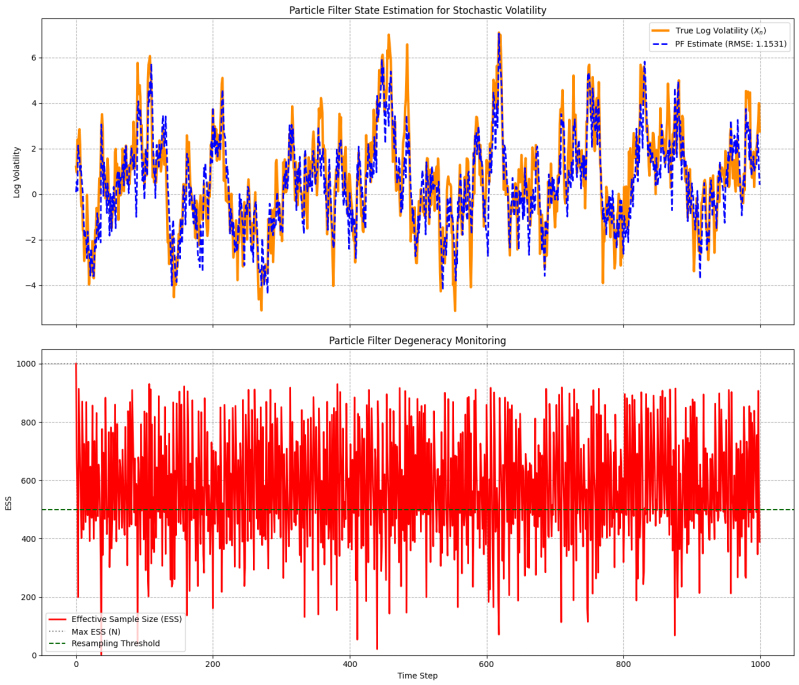

Figure 7: Comparison of the SV model in [9] by PF.

d) Compare PF and EKF/UKF performance. How to evaluate your SSMs? What metrics can you use? In practice, we care about the runtime and memory, could you also compare the runtime and peak memory(CPU/GPU RAM) for each SSM?

The evaluation metric of the SSMs is root mean square error (RMSE), which are 1.2911 (EKF), 1.2995 (UKF) and 1.1531 (PF), respectively that are shown in the Fig. 6 and 7. The RMSE of PF is the lowest.

The metric for runtime (latency) can be Floating Point Operations (FLOPs), and the metric for Peak Memory is primarily the size of the covariance matrix $\mathbf{P}$ and the number of particles N.

According to [11], both EKF and UKF have an $\mathcal{O}(n_x^3)$ complexity. [1] and [9] discusses the PF's memory bottleneck, noting that as the state dimension n_x increases, the number of particles N often needs to increase exponentially to maintain accuracy (the curse of dimensionality), exacerbating the memory and runtime cost $\mathcal{O}(N \cdot n_x)$, while the former one confirms the linear dependence on particle count for PF runtime ($\mathcal{O}(N)$).

Remark: This part will be experimented in the future full report.

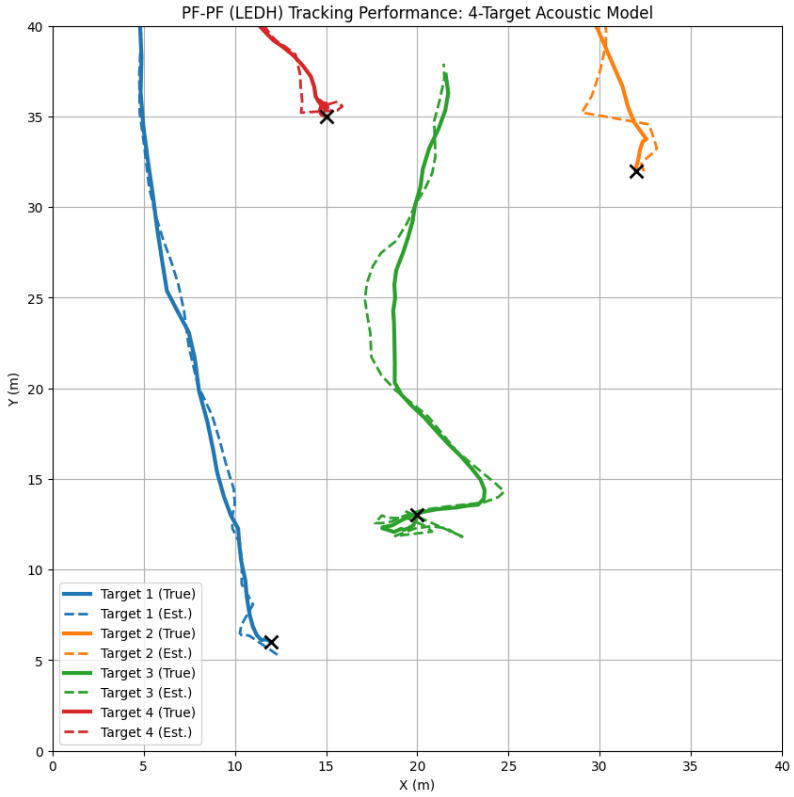

Figure 8: Result of the acoustic tracking model in [13] by PF.

2 Deterministic and Kernel Flows

2.1 invertible particle flow particle filter (PF-PF)

a) Study the Exact Daum-Huang (EDH) flow and Local Exact Daum-Huang (LEDH) flow, (see [6] and [5]), and the invertible particle flow particle filter (PF-PF) framework (see [13]).

Experiment Result. The replicated result of the Localized Exact Daum and Huang filter - particle flow particle filter (LEDH PF-PF) is shown in the Fig. 8.

The exact Daum and Huang filter (EDH) [6, 5] is a theoretically derived particle flow filter designed to address the challenges of sequential state estimation, particularly the problem of weight degeneracy in high-dimensional or highly informative measurement scenarios. It achieves this by modeling a deterministic flow that transports particles from the prior distribution to the desired posterior distribution.

The EDH flow is often incorporated into an encompassing particle filter framework [13]. **PF-PF (EDH):** In this method, the EDH flow is used as a highly accurate proposal distribution. Because the flow parameters ($\mathbf{A}$ and $\mathbf{b}$) are calculated using the global mean ($\overline{\eta}$), the Jacobian determinant term in the weight update is common to all particles and cancels out during normalization. This

makes the PF-PF (EDH) computationally efficient.

The Localized Exact Daum and Huang (LEDH) filter is an advanced particle flow technique that is a variation of the Exact Daum and Huang (EDH) filter. It is designed to address the challenges of nonlinear filtering problems, particularly by making the flow locally adaptive to the posterior density. It needs to track the local flow field of every particles. By performing local linearization, the LEDH constructs a more accurate and robust proposal distribution than the EDH, which is why the PF-PF (LEDH) often exhibits lower tracking error than the PF-PF (EDH) in highly challenging nonlinear scenarios.

2.2 Implement the kernel-embedded particle flow filter in an RKHS following [8]

The motivation of the study of kernel-embedded particle flow filter (kernel PFF) is to investigate how to apply the PFF to high-dimensional nonlinear problems. The scalar kernel version is insufficient in high-dimensional sparsely observed settings. The paper proposes the matrix-valued kernels version of PFF. This is also a kind of recently developed Monte Carlo filter based on a deterministic flow that avoids weight degeneracy problem and has high potential on high-dimensional complicated situations. When the particle flow is assumed to be embedded in RKHS, the analytical expression of the particle flow that minimizes the Kullback-Leibler divergence starting from the prior and ending to the posterior probability distribution function can be derived.

I have not reproduced this method yet, and plan to complete it before the final deadline in January, 2026.

3 Future Plan

In the part 2 of question 2, the stochastic particle flow and differentiable PF will need to be studied. Optimal transport (OT) resampling is a way to do differentiable particle filter, because the matrix operations are differentiable [3, 4].

OT theory provides the mathematical framework for finding the most cost-effective way to transform one probability distribution into another.

In resampling, the two distributions are: **the weighted prior distribution:** the set of particles $\{\mathbf{x}_k^i\}$ with non-uniform weights $\{w_k^i\}$. **The unweighted posterior distribution:** the new set of particles $\{\tilde{\mathbf{x}}_k^i\}$ with uniform weights $\{1/N\}$. The OT resampling methods seek to find a transport map that moves the particles from the weighted distribution to the uniform distribution while minimizing a predefined 'cost function'.

The purpose of defining the differentiable particle filter is that, it can leverage the recent technology of neural network as a dynamic surrogate and achieve much more efficient sampling process. I will try to do the bonus questions as many as possible and submit the final report before deadline in 2026.

References

[1] M Sanjeev Arulampalam et al. "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking". In: *IEEE Transactions on signal processing* 50.2 (2002), pp. 174–188.

[2] Yaakov Bar-Shalom, X Rong Li, and Thiagalingam Kirubarajan. *Estimation with applications to tracking and navigation: theory algorithms and software*. John Wiley & Sons, 2001.

[3] Xiongjie Chen and Yunpeng Li. "An overview of differentiable particle filters for data-adaptive sequential Bayesian inference". In: *arXiv preprint arXiv:2302.09639* (2023).

[4] Adrien Corenflos et al. "Differentiable particle filtering via entropy-regularized optimal transport". In: *International Conference on Machine Learning*. PMLR. 2021, pp. 2100–2111.

[5] Fred Daum and Jim Huang. "Particle degeneracy: root cause and solution". In: *Signal Processing, Sensor Fusion, and Target Recognition XX*. Vol. 8050. SPIE. 2011, pp. 367–377.

[6] Fred Daum, Jim Huang, and Arjang Noushin. "Exact particle flow for nonlinear filters". In: *Signal processing, sensor fusion, and target recognition XIX*. Vol. 7697. SPIE. 2010, pp. 92–110.

[7] Alexandros Evangelidis and David Parker. "Quantitative verification of numerical stability for Kalman filters". In: *International Symposium on Formal Methods*. Springer. 2019, pp. 425–441.

[8] Chih-Chi Hu and Peter Jan Van Leeuwen. "A particle flow filter for high-dimensional system applications". In: *Quarterly Journal of the Royal Meteorological Society* 147.737 (2021), pp. 2352–2374.

[9] Adam Johansen. "A tutorial on particle filtering and smoothing: Fifteen years later". In: (2009).

[10] Simon J Julier and Jeffrey K Uhlmann. "New extension of the Kalman filter to nonlinear systems". In: *Signal processing, sensor fusion, and target recognition VI*. Vol. 3068. Spie. 1997, pp. 182–193.

[11] Simon J Julier and Jeffrey K Uhlmann. "Unscented filtering and nonlinear estimation". In: *Proceedings of the IEEE* 92.3 (2004), pp. 401–422.

[12] Rudolph Emil Kalman. "A new approach to linear filtering and prediction problems". In: (1960).

[13] Yunpeng Li and Mark Coates. "Particle filtering with invertible particle flow". In: *IEEE Transactions on Signal Processing* 65.15 (2017), pp. 4102–4116.