# Pendulum Angle Estimation From Noisy Measurements

Emir Uncu
e273968@metu.edu.tr
Github repo: https://github.com/emiiruncu10/EE5506-Project.git

# Contents

# 1. Introduction

Accurate state estimation is a critical aspect of control and signal processing in nonlinear dynamical systems. In many practical systems, the true states of interest are not directly observable, and only noisy measurements can be obtained. Estimators such as the **Extended Kalman Filter (EKF), Unscented Kalman Filter (UKF),** and **Particle Filter (PF)** provide tools to infer these hidden states based on available data and knowledge of system dynamics.

This report investigates the performance of EKF, UKF, and PF in estimating the angular position of a nonlinear pendulum system, using simulated measurements. The pendulum serves as a representative benchmark due to its nonlinear dynamics and widespread usage in control theory and filtering literature. The aim is to evaluate and compare the estimation accuracy of these filters under identical simulation conditions using Monte Carlo analysis. By analyzing their **Root Mean Square Error (RMSE)** and variance across multiple trials, we asses the robustness and reliability of each method in handling nonlinear state-space models.

# 2. Problem Definition

In this study, we address the nonlinear state estimation problem of a single pendulum with noisy measurements. The objective is to estimate the pendulum's angular position over time using sequential observations corrupted by Gaussian noise, applying various Bayesian filtering methods including EKF, UKF and Particle Filter.

## 2.1. System Description

We consider a simple pendulum of unit length and mass, subject to gravity g = 9.8 m/$s^2$ The state vector is defined as:

$x_k = [\alpha_k; \dot{\alpha}_k]$

Where $\alpha k$ is the pendulum angle and $\dot{\alpha} k$ is the angular velocity at time step k. A simple pendulum consist of a point mass suspended by a massless rod of fixed length. The motion of the pendulum is governed by the nonlinear second-order ordinary differential equation as:

$$\frac{\mathrm{d}^2\alpha}{\mathrm{d}t^2} = -g\ \sin(\alpha) + w(t),$$

This model can be converted into the following state

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} x_2 \\ -g\ \sin(x_1) \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \end{pmatrix} w(t),$$

Where x1 = $\alpha_k$ and x2 = $\dot{\alpha}_k$. This form can be seen to be a special case of continuous-time non-linear dynamic models of the form:

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \mathbf{f}(\mathbf{x}) + \mathbf{L}\,\mathbf{w},$$

Where **f(x)** is a non-linear function. The physics of the problem is shown below, where $\alpha$ is the angle relative to vertical, and w(t) is a white noise process added to the angular velocity (a random acceleration term.)

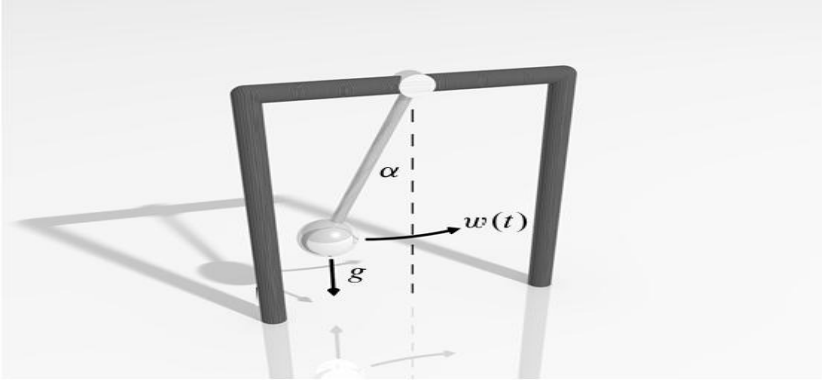


*Figure 1: Ullustration of pendulum*

## 2.2. Discretization via Euler Integration

To use this model in a digital filter, we discretize it using **Euler Integration,** a basic method for solving ODEs numerically. For small time step $\Delta t$, we approximate:

$$\alpha_k \approx \alpha_{k-1} + \dot{\alpha}_{k-1} \cdot \Delta t$$

$$\dot{\alpha}_k \approx \dot{\alpha}_{k-1} - g\sin(\alpha_{k-1}) \cdot \Delta t$$

Hence, the discrete-time state transition function f(.) becomes:

$$x_k = f(x_{k-1}) = [\alpha_{k-1} + \dot{\alpha}_{k-1} \cdot \Delta t; \dot{\alpha}_{k-1} - g\sin(\alpha_{k-1}) \cdot \Delta t]$$

To model real-world uncertainty, we add process noise:

$$x_k = f(x_{k-1}) + q_{k-1}, q_{k-1} \sim N(0, Q)$$

The process noise covariance $Q \in R^{2X2}$ models uncertainty in the dynamics and is typically derived from a white noise acceleration model:

$$Q = q^c \begin{pmatrix} \frac{\Delta^3}{3} & \frac{\Delta^2}{2} \\ \frac{\Delta^2}{2} & \Delta \end{pmatrix}$$

Where $q^c$ is the spectral density of the continuous-time noise process.

## 2.3. Measurement Model

In many practical settings, we cannot directly observe the full state vector $xk$. Instead, we receive nonlinear measurements that depend on the state. In this case the observed quantity is:

$$y_k = sin(\alpha_k) + r_k$$

Where $rk \sim N(0, R)$ is Gaussian measurement noise, and $R \in$ R is the measurement variance. This defines the observation function h(.):

$$y_k = h(x_k) + r_k = sin(\alpha_k) + rk$$

## 2.4. Summary: Nonlinear State-Space Model

The full discrete-time stochastic nonlinear system is:
State Transition (dynamics):

$$x_k = f(x_{k-1}) + q_{k-1}, q_{k-1} \sim N(0, Q)$$

Observation Model:

$$y_k = h(x_k) + r_k, r_k \sim N(0, R)$$

Where:

- f(.): nonlinear Euler-integrated dynamics.
- h(.) = sin(αk): nonlinear observation function.

## 2.5. Estimation Objective

Given a sequence of noisy observations $y1{:}\,k$, our objective is to estimate the hidden state $xk$, i.e., the posterior:

$$p(x_k \mid y_{1:k})$$

Due to the nonlinearity in both state evolution and observation, exact Bayesian filtering is intractable. Therefore, we use approximate Bayesian filters:

- Extended Kalman Filter (EKF): Linearizes the model using Jacobians.

- Unscented Kalman Filter (UKF): Uses deterministic sigma-point sampling to approximate nonlinear transformation of Gaussian variables.
- Particle Filter (PF): Uses importance sampling and resampling of particles to approximate the posterior.

The performance of each filter is assessed by comparing their estimated state trajectories $\alpha k\hat{}$ to the true values $\alpha k$, using RMSE and variance as metrics over multiple Monte Carlo simulations.

# 3. Description Of The Methods And Background

This section presents three sequential Bayesian estimation methods applied to nonlinear pendulum tracking problem: EKF, UKF and PF. Each method aims to estimate the pendulum angle and angular velocity using noisy nonlinear measurements over time. Each filtering method will be introduced with the necessary theoretical background, provide mathematical formulations, and describe how the method is applied in practice. Additionally, block diagrams are included to visually summarize the filtering pipeline.

## 3.1. Extended Kalman Filter (EKF)

The extended Kalman Filter (EKF) is designed to estimate the hidden states of a nonlinear dynamic system by linearizing the nonlinear functions around the current estimate. It assumes: Additive Gaussian process and measurement noise, Approximate Gaussian posterior at each step (i.e. state is summarized by a mean and covariance). EKF consists of two steps:

- Prediction: Forward propagation of state and covariance.
- Update: Correction based on observation and innovation.

### 3.1.1. Mathematical Model

We will consider a nonlinear discrete-time state-space model:

$$x_k = f(x_{k-1}) + q_{k-1},\ q_{k-1} \sim N(0, Q)$$

$$y_k = h(x_k) + r_k, r_k \sim N(0, R)$$

Where:

$x_k \in R^n$: $state\ vector\ (angle, angular\ velocity)$

$y_k \in R^m$: $measurement\ (sine\ of\ the\ angle)$

$f(.)$: $nonlinear\ state\ transition\ function$

$h(.)$: $nonlinear\ observation\ function$

For the pendulum system:

$$f(x) = [x_1 + x_2.\Delta t\,; x_2 - g.\sin(x_1).\Delta t]$$

$$h(x) = \sin(x_1)$$

### 3.1.2. Euler Integration

The continuous-time dynamics of a pendulum are given by:

$$d/dt[\alpha; \omega] = [\omega; -gsin(\alpha)]$$

The discretized system using the Euler integration:

$$\alpha_k = \alpha_{k-1} + \omega_{k-1} \cdot \Delta t$$

$$\omega_k = \omega_{k-1} - g.\sin(\alpha_{k-1}).\Delta t$$

This leads to the discrete dynamics used in the EKF.

### 3.1.3. EKF Algorithm

Let:

- $x\hat{}_{k|k-1}$ : $Predicted\ State\ mean$
- $P_{k|k-1}$ : $Predicted\ covariance$
- $x\hat{}_{k|k}$ : $Updated\ (filtered)\ State\ mean$
- $P_{k|k}$: $Updated\ covariance$

Prediction step:

$$x\hat{}_{k|k-1} = f(x\hat{}_{k-1|k-1})$$

$$P_{k|k-1} = F_K P_{K-1|K-1} {F_K}^T + Q$$

Where $F_K = df/dx|_{x\hat{}_{k-1|k-1}}$ is the jacobian of the dynamics.

Update step:

$$y_{k|k-1} = h(x\hat{}_{k|k-1})$$

$$H_k = dh/dt|_{x\hat{}_{k|k-1}}$$

$$S_k = H_k P_{k|k-1} {H_k}^T + R$$

$$K_k = P_{k|k-1} {H_k}^T {S_k}^{-1}$$

$$x\hat{}_{k|k} = x\hat{}_{k|k-1} + K_k(y_k - y_{k|k-1})$$

$$P_{k|k} = (I - K_k H_k) P_{k|k-1}$$

### 3.1.4. EKF Block Diagram

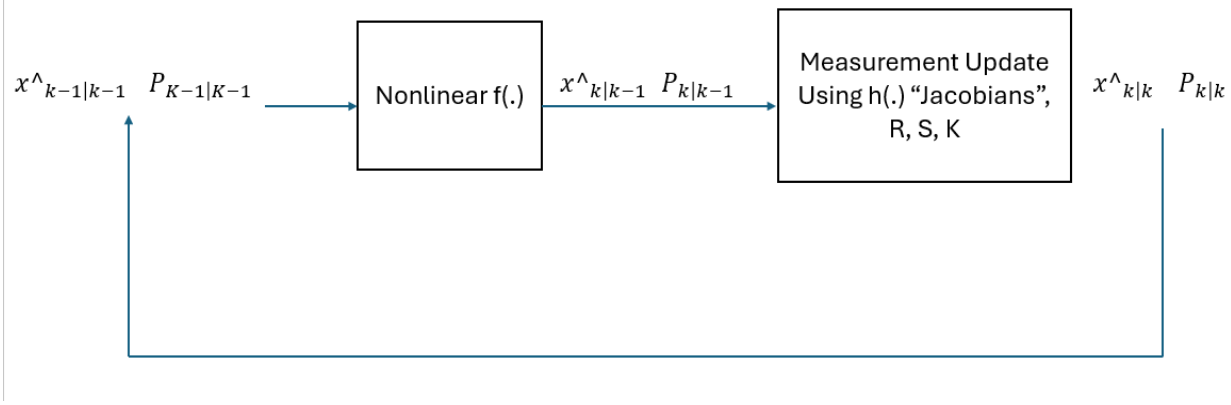


*Figure 2: EKF block diagram*

### 3.1.5. EKF in Implementation

In the pendulum angle estimation:
The State is: $x_k = [\alpha_k, \omega_k]^T$

The observation is: $y_k = \sin(\alpha_k) + r_k$

Jacobians are computed at each step:

- Dynamics Jacobian:

$$F = \begin{pmatrix} 1 & \Delta t \\ -g.\cos(\hat{\alpha}_{k-1})\Delta t & 1 \end{pmatrix}$$

- Observation Jacobian:

$$H = [\cos(\hat{\alpha}_{k|k-1}) \ 0]$$

The EKF is executed in MATLAB for 100 monte Carlo runs, and the root mean square error (RMSE) is computed for each run to evaluate performance. The EKF performs reasonably well when the nonlinearities are mild and the initial estimate is close to the true state.

## 3.2. Unscented Kalman Filter (UKF)

The unscented Kalman Filter (UKF) is an advanced nonlinear state estimation method that avoids linearizing the nonlinear functions as done in EKF. Instead, UKF approximates the distribution of the state using a set of deterministically chosen sample points called **sigma points** and propagates them through the actual nonlinear functions. This makes the UKF more accurate than EKF, especially in systems with highly nonlinear dynamics or

observations. UKF is based on the Unscented Transform, which is a method for computing the mean and covariance of a random variable undergoing a nonlinear transformation.

### 3.2.1. Mathematical Model

Again considering a nonlinear discrete-time state-space model:

$$x_k = f(x_{k-1}) + q_{k-1}, q_{k-1} \sim N(0, Q)$$

$$yk = h(x_k) + r_k, r_k \sim N(0, R)$$

But instead of approximating $f(.)$ and $h(.)$ With Jacobians, we approximate the probability distributions directly using sigma points.

### 3.2.2. Unscented Transform

Given a state vector X $\in R^n$ with mean μ and covariance $P$, the UKF generates 2n + 1sigma points:

$$X_0 = \mu$$

$$X_i = \mu + (\sqrt{(n + \lambda)P})_{i,}\ i = 1, \dots\dots\dots, n$$

$$X_{i+n} = \mu - (\sqrt{(n + \lambda)P})_{i,}\ i = 1, \dots\dots\dots, n$$

Where $\lambda = \alpha^2(n + \kappa) - n$ is a scaling parameter, and $\sqrt{(n + \lambda)P}$ denotes the matrix square root (typically Cholesky decomposition). Each sigma point is weighted

- For the mean:

$$W_0^{(m)} = \frac{\lambda}{n + \lambda}, \qquad W_i^{(m)} = \frac{1}{2(n + \lambda)}$$

- For the covariance:

$$W_0^{(c)} = \frac{\lambda}{n + \lambda} + (1 - \alpha^2 + \beta), \qquad W_i^{(c)} = \frac{1}{2(n + \lambda)}$$

- Recommended parameters:

$$\alpha = 10^{-3}, \kappa = 0, \beta = 2$$

### 3.2.3. UKF Algorithm

Prediction Step:

- Generate sigma points from $x\hat{}_{k-1|k-1}$ , $P_{K-1|K-1}$
- Propagate each sigma point through the dynamics:

$$x_{k|k-1}{}^{(i)} = f\ (x_{k-1}{}^{(i)})$$

- Estimate predicted mean and covariance:

$$x\hat{}_{k|k-1} = \sum_i W_i^{(m)} x_{k|k-1}{}^{(i)}$$

$$P_{K|K-1=} \sum_i W_i^{(c)} (x_{k|k-1}{}^{(i)} - x\hat{}_{k|k-1})(x_{k|k-1}{}^{(i)} - x\hat{}_{k|k-1})^T + Q$$

Update Step:

- Transform sigma points through observation model:

$$\gamma_k^{(i)} = h(x_{k|k-1}{}^{(i)})$$

- Estimate predicted observation mean and innovation covariance:

$$\hat{y_k} = \sum_i W_i^{(m)} \gamma_k^{(i)}$$

$$S_k = \sum_i W_i^{(c)} (\gamma_k^{(i)} - \hat{y_k})^2 + R$$

- Compute cross-covariance:

$$P_{xy} = \sum_i W_i^{(c)} (x_{k|k-1}{}^{(i)} - \hat{x}_{k|k-1})(\gamma_k^{(i)} - \hat{y_k})^T$$

- Kalman gain, update state and covariance:

$$K_k = P_{xy} S_k{}^{-1}$$

$$\hat{x_{k|k}} = \hat{x_{k-1}} + K_k(y_k - \hat{y_k})$$

$$P_{k|k} = P_{k|k-1} - K_k S_k K_k{}^T$$

### 3.2.4. UKF Block Diagram

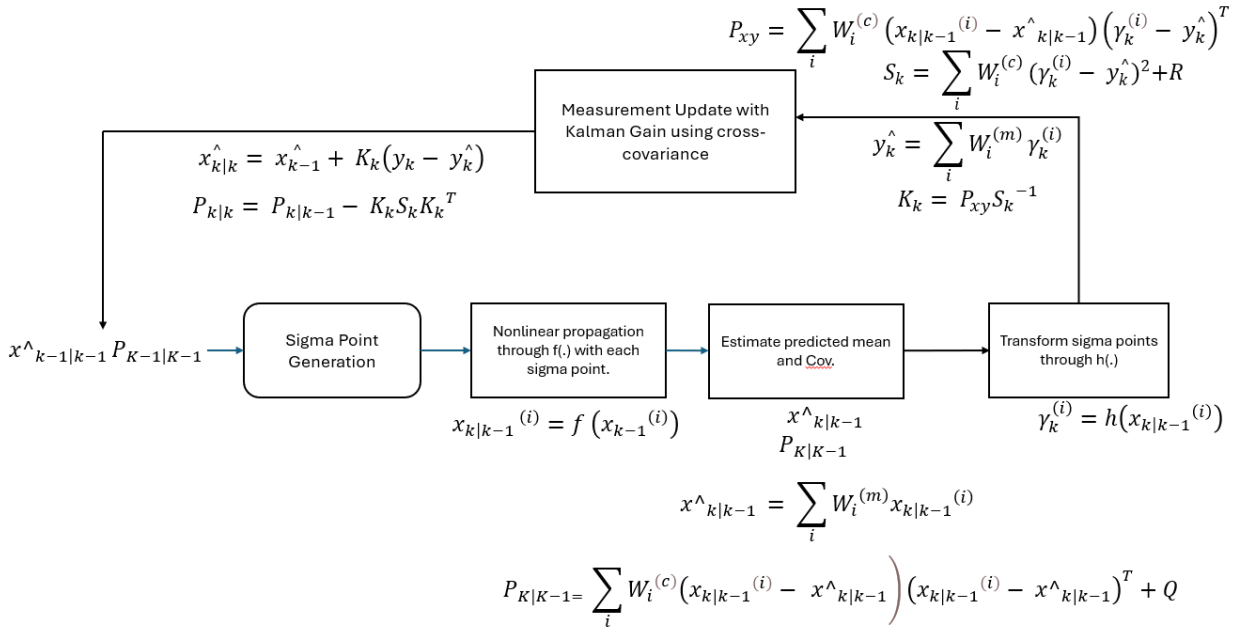


*Figure 3: UKF Block Diagram*

### 3.2.5. UKF in Implementation

Within the pendulum angle estimation problem:

- State: $x_k = [\alpha_k,\ \omega_k]^T$
- Observation: $y_k = \sin(\alpha_k) + r_k$
- No Jacobian computations are needed.
- Sigma points are deterministically chosen at each iteration.
- System dynamics and observation models are the same as EKF but are directly used on each sigma point.

The MATLAB implementation includes Monte Carlo runs to average the RMSE across runs. UKF usually performs better than EKF when the system is highly nonlinear, as it captures higher-order effects.

## 3.3. Particle Filter (PF)

The particle Filter (PF), also known as Sequential Monte Carlo (SMC), is a non-parametric Bayesian filtering method used to estimate the posterior distribution of the system state when the state-space model is nonlinear and/or the noise is non-Gaussian. Instead of approximating the distribution with a Gaussian (as in EKF or UKF), PF represents the posterior using a set

of random samples (particles) and associated importance weights. It is especially powerful in highly nonlinear and non-Gaussian systems where Kalman-type filters perform poorly.

### 3.3.1. Mathematical Model

As before, the system is modeled using a nonlinear discrete-time state-space model:

$$x_k = f(x_{K-1}) + q_{K-1}, q_{K-1} \sim p(q)$$

$$y_k = h(x_k) + r_K, r_K \sim p(r)$$

The goal is to estimate the posterior distribution $p(x_k/y_{1:k})$ where $y_{1:k}$ is the sequence of measurements up to time k.

### 3.3.2. Particle Filtering Algorithm

The PF approximates the posterior using N particles $\{x_k{}^{(i)}\}$ i = 1……N with weights $\{w_k{}^{(i)}\}$ i = 1…….N such that:

$$p\left(\frac{x_k}{y_{1:k}}\right) \approx \sum_{i=1}^{N} w_k{}^{(i)} \delta(x_k - x_k{}^{(i)})$$

Where $\delta(\cdot)$ is the Dirac delta function.

Steps:

- Initialization
  - Sample initial particles $x_0{}^{(i)} \sim p(x_0)$
  - Set weights $w_0{}^{(i)} = 1/N$
- Recursive Steps (for each time step k)
  - Prediction:

$$x_k{}^{(i)} \sim p(x_k/x_{K-1}{}^{(i)})$$

  - Weight Update:

$$w_k{}^{(i)} \propto w_{k-1}{}^{(i)} p(y_k/\, x_k{}^{(i)})$$

    Normalize $\sum_{i=1}^{N} w_k{}^{(i)} = 1$

  - Resampling (optional but often necessary):
    If the effective sample size Neff = $\frac{1}{\sum_i (w_k{}^{(i)})^2}$ is to small, perform resampling to avoid degeneracy (i.e. all weight concentrated on a few particles).

### 3.3.3. Residual Resampling

In our implementation, we use residual Resampling, which reduces variance compared to naïve multinomial resampling. It works by:

- Assigning particles deterministically based on the integer parts of N. $w_k{}^{(i)}$
- Filling remaining slots by sampling from the residual fractional weights.

This helps maintain diversity in the particle set while avoiding redundant duplication.

### 3.3.4. PF Block Diagram

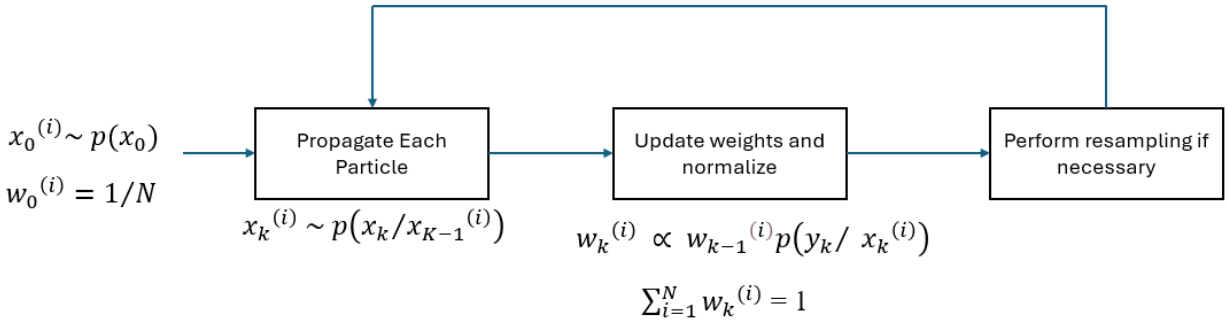


*Figure 4: PF Block Diagram*

### 3.3.5. PF in Implementation

In the pendulum estimation setup:

- State: $x_k = [\alpha_k,\ \omega_k]^T$
- Observation: $y_k = \sin(\alpha_k) + r_k$
- The dynamics and emission functions are nonlinear.
- Gaussian process and measurement noise are assumed.
- Each particle is updated through:

$$x_k{}^{(i)} = f\left(x_{k-1}{}^{(i)}\right) + noise$$

- Likelihood is calculated as:

$$p(y_k/x_k{}^{(i)}) \ \propto \ \exp\left(-\frac{1}{2R}(y_k - \sin\left(\alpha_k{}^{(i)}\right))^2\right)$$

Each particle is reweighted according to this likelihood, and residual resampling ensures effective representation.

# 4. Implementation and Results

In this section, the practical implementation of the three nonlinear Bayesian filtering methods: EKF, UKF and PF. All filters are applied to the same pendulum angle estimation problem using a common system model, observation model, and noise characteristics. A monte carlo simulation with 100 independent trials is performed to statistically evaluate and compare the accuracy and robustness of each filtering method.

## 4.1. Common System Setup

The simulation environment is based on discretized pendulum model with the following dynamics.

**State Vector:** $x_k = [\alpha_k; \dot{\alpha}_k]$

Where $\alpha_k$ is the pendulum angle and $\dot{\alpha}_k$ is the angular velocity.

**Dynamics model (nonlinear):**

$$\alpha_{k+1} = \alpha_k + \dot{\alpha}_k . \Delta t$$

$$\dot{\alpha}_{k+1} = \dot{\alpha}_k - g sin(\alpha_k) . \Delta t$$

Implemented via Euler integration with time step $\Delta t$ = 0.0125

**Observation model:**

$$y_k = sin(\alpha_k) + v_k$$

Where $v_k \sim N(0, R)$ is Gaussian noise.

**Noise characteristics:**

Process noise covariance Q is constructed based on continuous white noise acceleration assumptions. Observation noise variance $R = 0.3^2$

In each trial a ground truth trajectory is generated by simulating the nonlinear dynamics with process noise. Noisy observations are generated from the ground truth using the observation model. All filtering methods estimate the angle trajectory $\alpha_k$ using only noise observations.

## 4.2. Extended Kalman Filter (EKF) Implementation

The EKF is implemented by linearizing the nonlinear state transition and observation models around the current state estimate:

**Prediction Step:**

Linearized with the Jacobian of the dynamics:

$$F_k = \frac{df}{dx} = \begin{bmatrix} 1 & \Delta t \\ -g\cos(\alpha_k).\Delta t & 1 \end{bmatrix}$$

State and covariance prediction using standard EKF equations.

**Update Step:**

Observation model linearized via:

$$H_k = \frac{dh}{dx} = [\cos(\alpha_k)\ 0]$$

Kalman gain and measurement update performed using the linearized model.

Initial state and covariance are set as:

$$\hat{x_0} = \left[\frac{\pi}{4}; 0\right], P_0 = 0.1.I$$

## 4.3. Unscented Kalman Filter (UKF) Implementation

The UKF avoids linearization by using the Unscented Transform to propagate a set of sigma points through the nonlinear functions.:

**Sigma point generation:**

It is performed using the sigma_points.m function, which computes: 2n + 1 sigma points based on the current mean and covariance. Weights $W_m$ and $W_c$ for mean and covariance reconstruction.

**Prediction Step:**

Each sigma point is passed through the nonlinear dynamics. Predicted mean and covariance are computed using weighted sums.

**Update Step:**

Transformed sigma points are passed through the observation model, Innovation covariance $P_{yy}$, cross-covariance $P_{xy}$ and Kalman gain $K_k$ are computed and used for state update. Tuning parameters: α = $10^{-3}$ , $\beta = 2$, $\kappa = 0$, the UKF implementation relies more on accurate approximation of nonlinear effects and provides better performance than EKF in highly nonlinear systems.

## 4.4. Particle Filter (PF) Implementation

The PF is implemented using a bootstrap particle filter with residual resampling, where particles are propagated and reweighted at each time step:

**Initialization:**

1000 particles are sampled from a Gaussian distribution around the initial state. All particles are initialized with equal weights.

**Prediction Step:**

Each particle is propagated through the nonlinear dynamics with process noise.

**Update Step:**

Particle weights are updated using the likelihood function:

$$w_k{}^{(i)} \propto \exp\left(-\frac{1}{2R}(y_k - \sin(\alpha_k{}^{(i)}))^2\right)$$

**Residual Resampling:**

Implemented via the custom residual_resample() function. Ensures particles with higher weights are sampled more often. Avoids degeneracy by maintaining particle diversity.

**Estimation:**

The filtered state estimate is computed as the mean of the particles.

This implementation of PF allows for approximation of highly nonlinear and non-Gaussian posterior distributions at the cost of computational complexity.

## 4.5. Monte Carlo Simulation

To ensure statistically significant comparisons, the entire experiment (state generation, observation generation, and filtering) is repeated for 100 trials. Each trial uses newly sampled process and measurement noise. For each method, the Root Mean Squared Error (RMSE) of the estimated angle $\hat{\alpha_k}$, against the true angle $\alpha_k$ is recorded:

$$\text{RMSE} = \sqrt{\frac{1}{T}\sum_{k=1}^{T}(\alpha_k - \hat{\alpha_k})^2}$$

At the end of 100 trials, the average RMSE and RMSE variance are computed for EKF, UKF and PF.

## 4.6. Results

To evaluate and compare the performance of three nonlinear state estimation techniques, EKF, UKF and PF, a pendulum angle tracking simulation using a common system model, observation model and noise characteristics are implemented. Each filter estimates the pendulum's angle over a fixed time horizon, using the same true states and noisy observations generated for each Monte Carlo run. The simulation is repeated over 100 trials to account for randomness due to noise.

**RMSE** will be used as primary performance metric. For each method and trial, we compute the RMSE between the estimated angle trajectory and the true angle trajectory. RMSE values are averaged over 100 Monte carlo runs for each method to assess the overall accuracy. **Variance** of the RMSE values are measured for consistency and robustness of the filter. Lower variances implies more reliable performance across runs.

**Monte Carlo** is used because each run has randomly generated noise in the state evolution and observation processes. By averaging over many runs, statistically meaningful performance metrics that are not sensitive to one-off random outcomes will be obtained.

Here are the first and last trial plot results for estimating an angle using EKF, UKF, PF

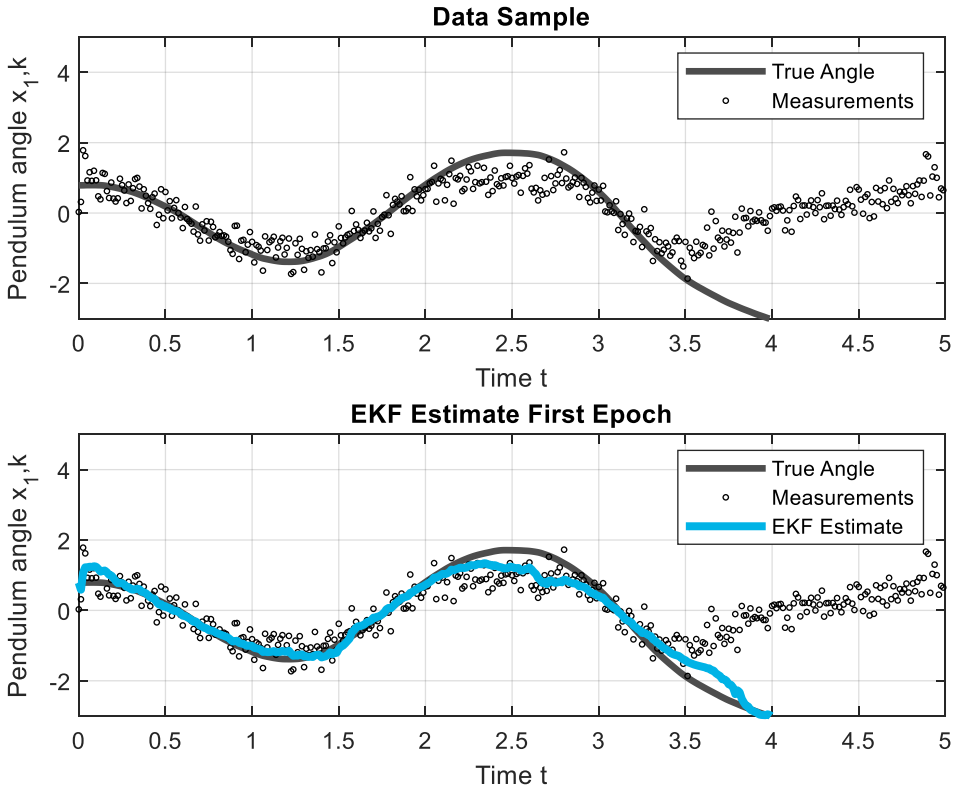


*Figure 5: EKF estimate first epoch*

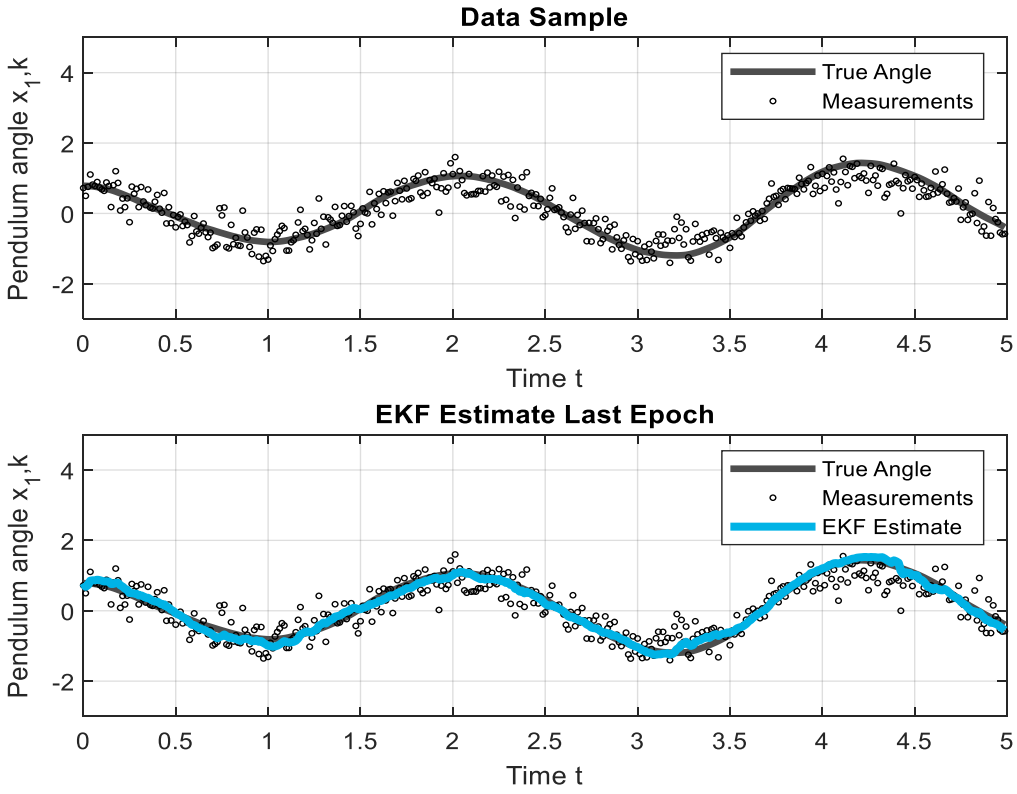


*Figure 6: EKF estimate last epoch*

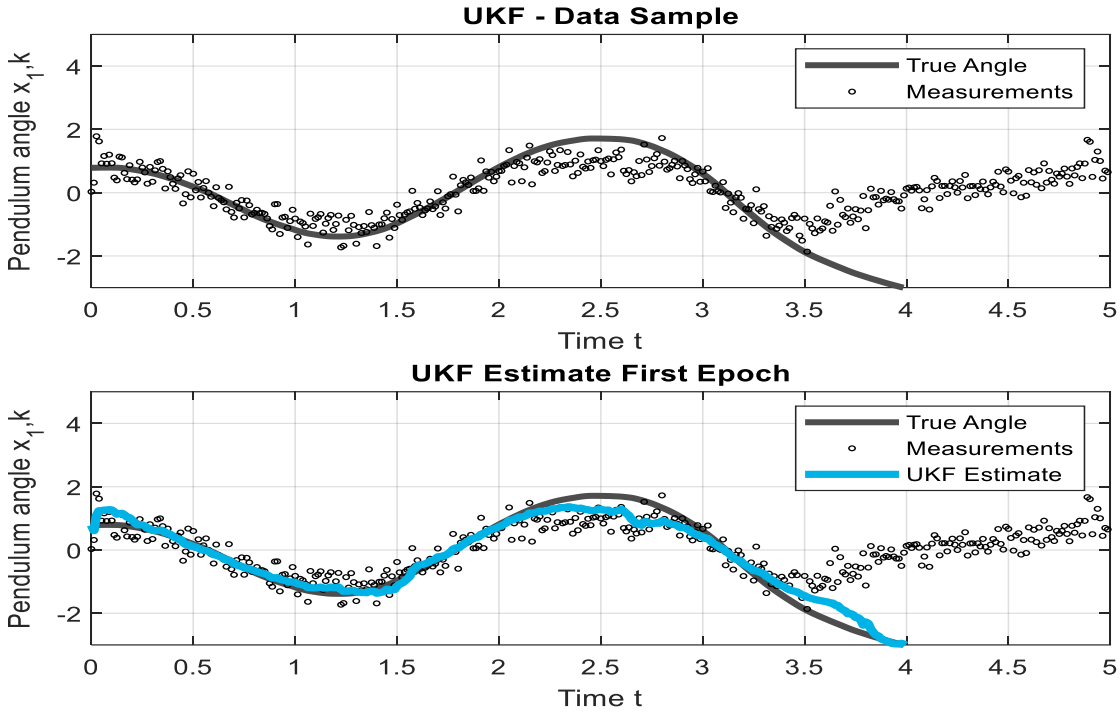


*Figure 7: UKF Estimate First Epoch*

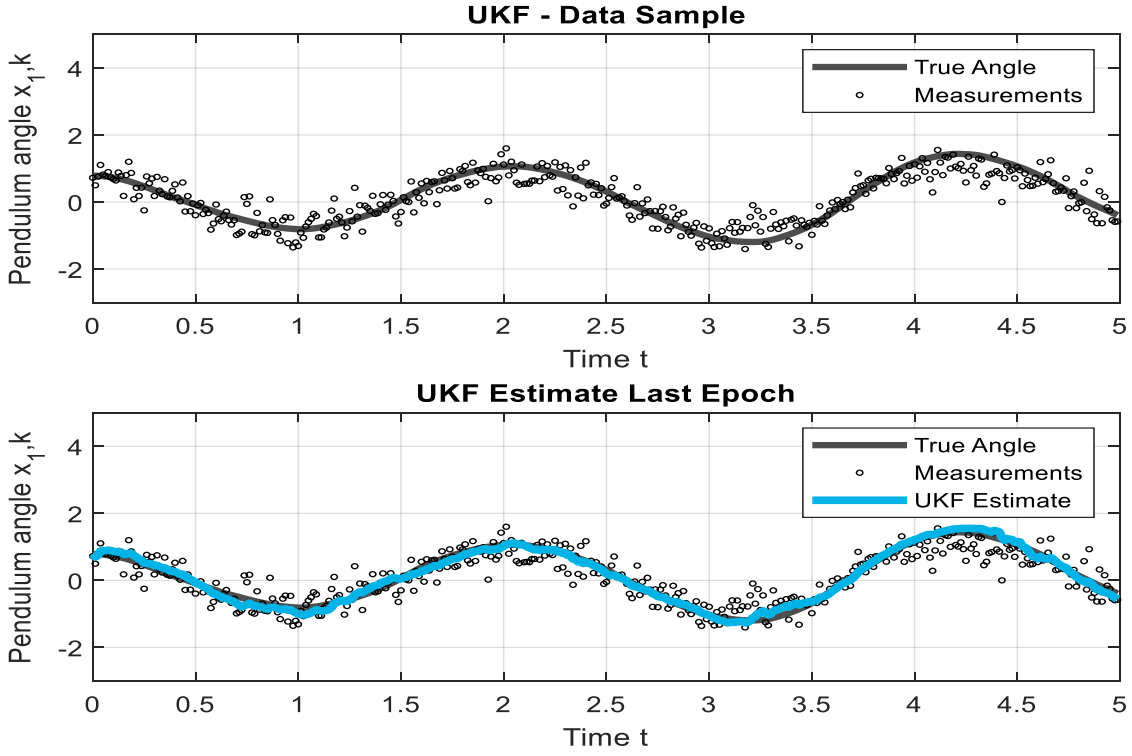


*Figure 8: UKF Estimate Last Epoch*

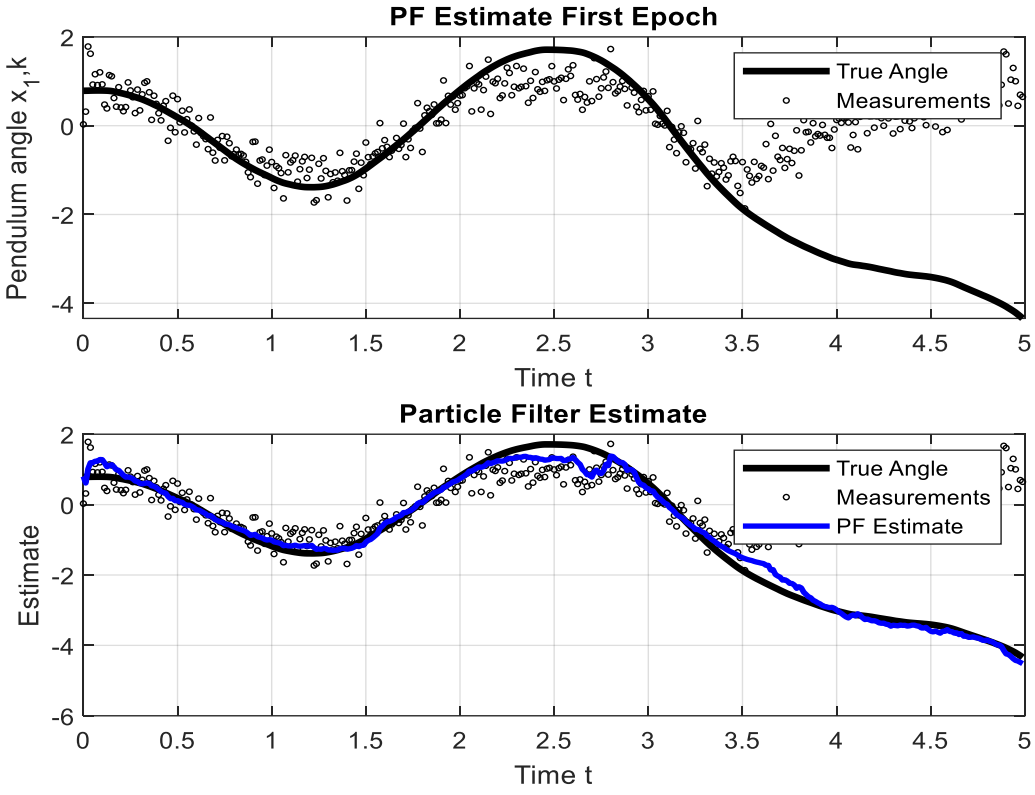


*Figure 9: Particle Filter First Epoch*

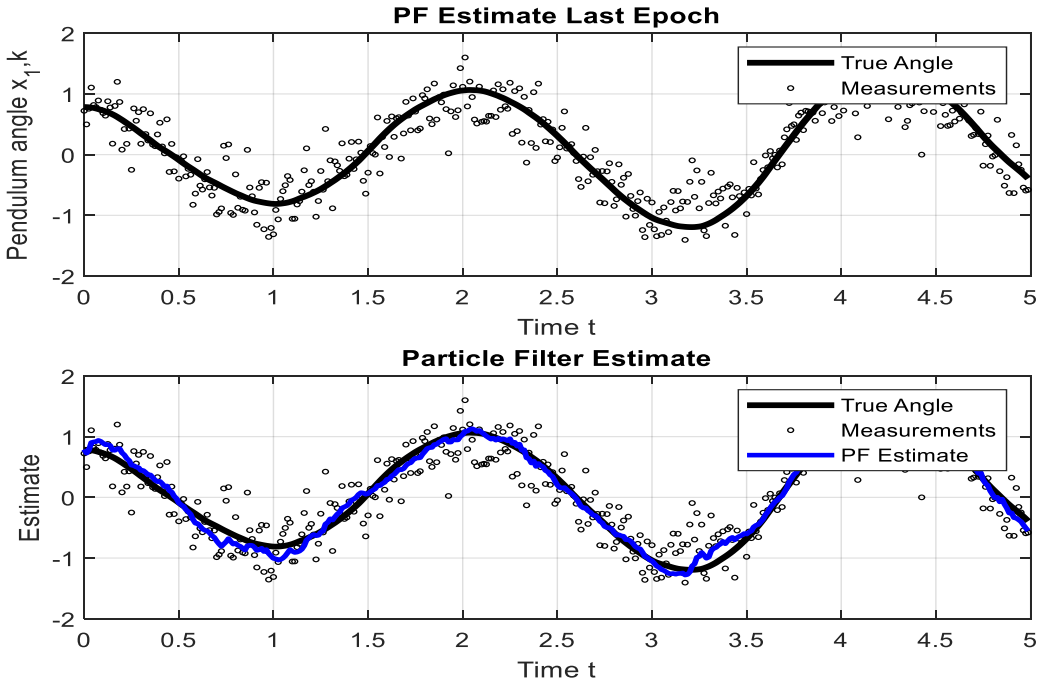


*Figure 10: Particle Filter Last Epoch*

Within first epochs: All three methods (EKF, UKF , PF) starts from the same initial conditions and are exposed to identical observations. The visual comparison shows that each method tracks the pendulum angle with reasonable accuracy even from the early stages. PF appears to match the ground truth more closely in regions with strong nonlinearity due to its sample-based approach.

Within last epochs: PF consistently aligns better with the ground truth trajectory, especially during highly dynamic transitions. EKF and UKF still follow the general trend but show slight divergence in fast-changing regions or larger amplitudes.

Calculated Metrics:

| Method | Average RMSE | RMSE Variance |
|---|---|---|
| **EKF** | 0.1721 | 0.1722 |
| **UKF** | 0.1723 | 0.1720 |
| **PF** | 0.1241 | 0.0019 |

The PF outperforms both EKF and UKF in terms of accuracy (lower RMSE) and consistency (lowest variance). EKF and UKF perform almost identically, which is expected since the degree of nonlinearity in the pendulum model is moderate and the system remains observable. The closeness of RMSE values between EKF and UKF indicates that the nonlinearity is not severe enough to cause divergence or major approximation errors in Jacobian or sigma point methods.

To better illustrate the strengths and limitations of each method especially the PF:

- Choose a system with higher nonlinearity or non-Gaussian noise.