Kalman Filter

Richard Dirauf, M.Sc.
Machine Learning and Data Analytics (MaD) Lab
Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU)
MLTS Exercise, 15.12.2022

Kalman Filter Overview

MLTS Exercise 06

Kalman filters (KF) are used to estimate the state of a dynamical system

- Trajectory estimation
- Indirect measurements (Rocket thruster temperature)

Assumptions:

- Linear Gaussian system with additive noise
  - Optimal estimator

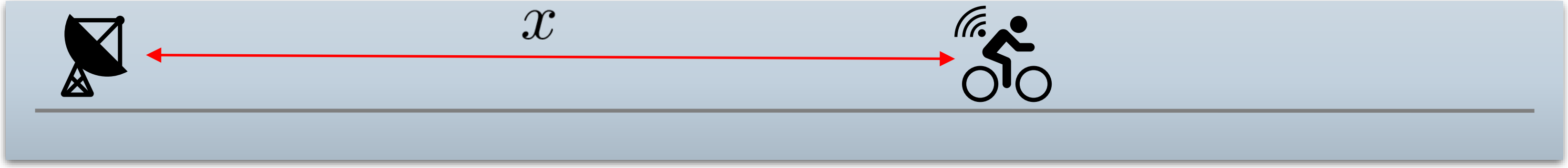
Kalman Filter Overview

Two important steps:

1. Prediction
   - Given an initial state, we leverage our knowledge of the process to produce an estimate of the current state – along with its uncertainty

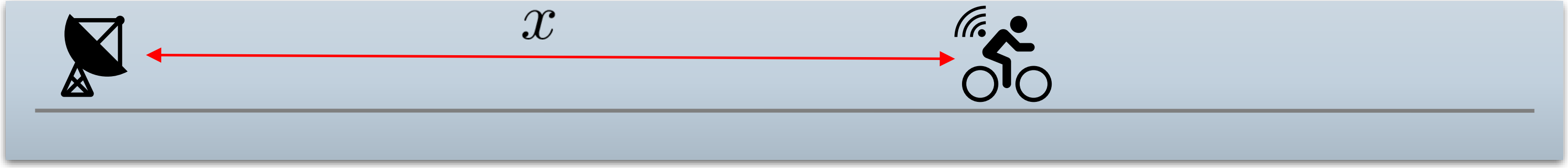
2. Correction (or Filtering)
   - We update our belief based on new sensory information

Handy example - Prediction



x is the position of a rider
$\dot{x}$ is the rider‘s velocity

Handy example - Prediction

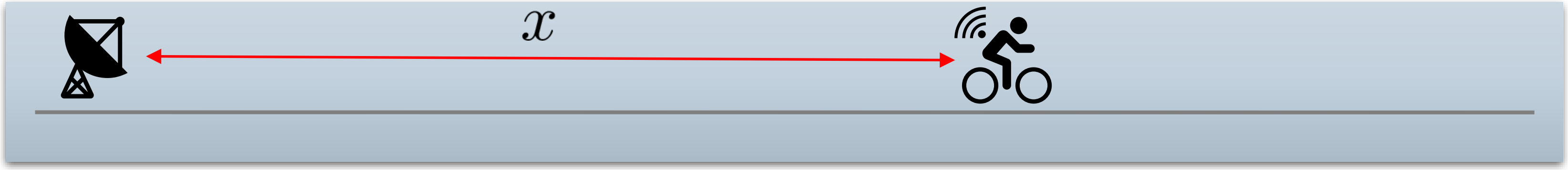


x is the position of a rider

$\dot{x}$ is the rider‘s velocity

$z = \begin{bmatrix} x \\ \dot{x} \end{bmatrix}$ is the system‘s state

Handy example - Prediction



x is the position of a rider

$\dot{x}$ is the rider's velocity

$z = \begin{bmatrix} x \\ \dot{x} \end{bmatrix}$ is the system's state

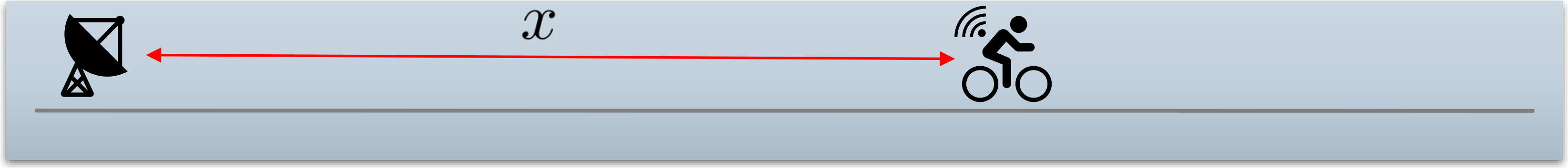
Process model:

We can write down the Newtonian laws of motion for the system

$$x_n = x_{n-1} + \dot{x}_{n-1}\Delta t + \frac{1}{2}\frac{f}{m}(\Delta t)^2 + r_{1_n}$$

$$\dot{x}_n = \dot{x}_{n-1} + \frac{f}{m}\Delta t + r_{2_n}$$

Handy example - Prediction



x is the position of a rider

$\dot{x}$ is the rider's velocity

$z = \begin{bmatrix} x \\ \dot{x} \end{bmatrix}$ is the system's state

Process model:

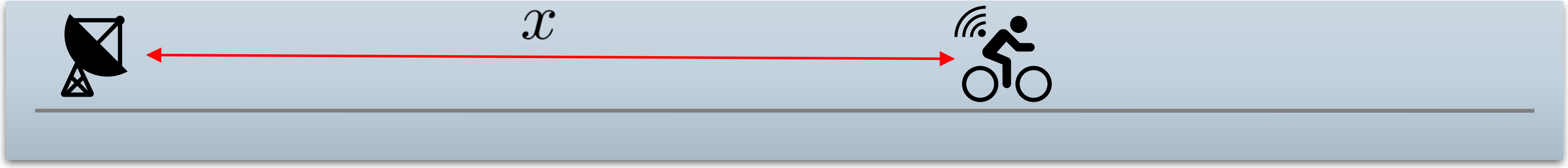
We can write down the Newtonian laws of motion for the system

$$x_n = x_{n-1} + \dot{x}_{n-1}\Delta t + \frac{1}{2}\frac{f}{m}(\Delta t)^2 + r_{1_n}$$

$$\dot{x}_n = \dot{x}_{n-1} + \frac{f}{m}\Delta t + r_{2_n}$$

$$\begin{bmatrix} x_n \\ \dot{x}_n \end{bmatrix} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_{n-1} \\ \dot{x}_{n-1} \end{bmatrix} + \begin{bmatrix} \frac{(\Delta t)^2}{2m} \\ \frac{\Delta t}{m} \end{bmatrix} f + \begin{bmatrix} r_{1_n} \\ r_{2_n} \end{bmatrix}$$

Handy example - Prediction



x is the position of a rider
$\dot{x}$ is the rider‘s velocity

$z = \begin{bmatrix} x \\ \dot{x} \end{bmatrix}$ is the system‘s state

Process model:

We can write down the Newtonian laws of motion for the system

$$x_n = x_{n-1} + \dot{x}_{n-1}\Delta t + \frac{1}{2}\frac{f}{m}(\Delta t)^2 + r_{1_n}$$

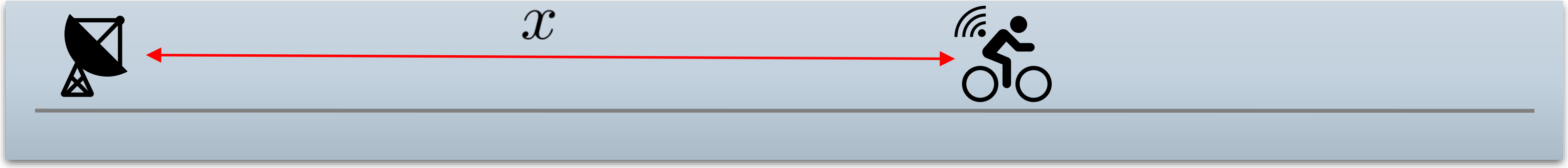
$$\dot{x}_n = \dot{x}_{n-1} + \frac{f}{m}\Delta t + r_{2_n}$$

$$\begin{bmatrix} x_n \\ \dot{x}_n \end{bmatrix} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_{n-1} \\ \dot{x}_{n-1} \end{bmatrix} + \begin{bmatrix} \frac{(\Delta t)^2}{2m} \\ \frac{\Delta t}{m} \end{bmatrix} f + \begin{bmatrix} r_{1_n} \\ r_{2_n} \end{bmatrix}$$

$$z_n = F z_{n-1} + B u_n + r_n$$

r_n Gaussian Noise
F State Transition Matrix
B Additional Information

Handy example - Filtering



x is the position of a rider

$\dot{x}$ is the rider‘s velocity

$z = \begin{bmatrix} x \\ \dot{x} \end{bmatrix}$ is the system‘s state

Measurement model:

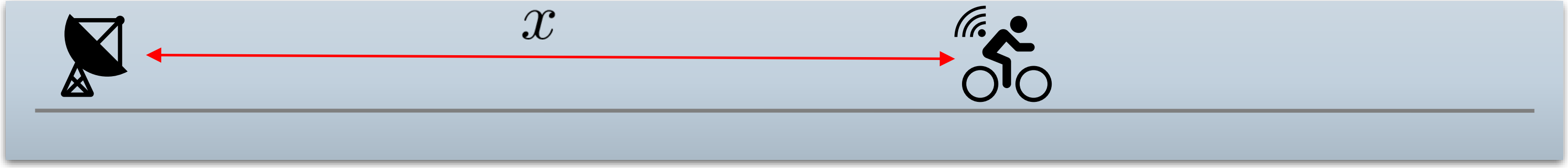
Let's assume that we measure the position directly and that this measurement is subject to random noise

$$y_n = x_n + q_n$$

Kalman Filter Example

MLTS Exercise 06

Handy example - Filtering



x is the position of a rider
$\dot{x}$ is the rider's velocity

$z = \begin{bmatrix} x \\ \dot{x} \end{bmatrix}$ is the system's state

Measurement model:

Let's assume that we measure the position directly and that this measurement is subject to random noise

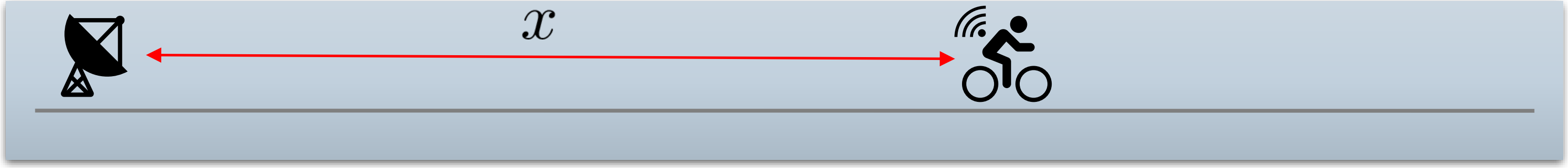
$$y_n = x_n + q_n$$

$$y_n = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_n \\ \dot{x}_n \end{bmatrix} + \begin{bmatrix} q_{1_n} \\ q_{2_n} \end{bmatrix}$$

Kalman Filter Example

MLTS Exercise 06

Handy example - Filtering



x is the position of a rider

$\dot{x}$ is the rider's velocity

$$z = \begin{bmatrix} x \\ \dot{x} \end{bmatrix}$$ is the system's state

Measurement model:

Let's assume that we measure the position directly and that this measurement is subject to random noise

$$y_n = x_n + q_n$$

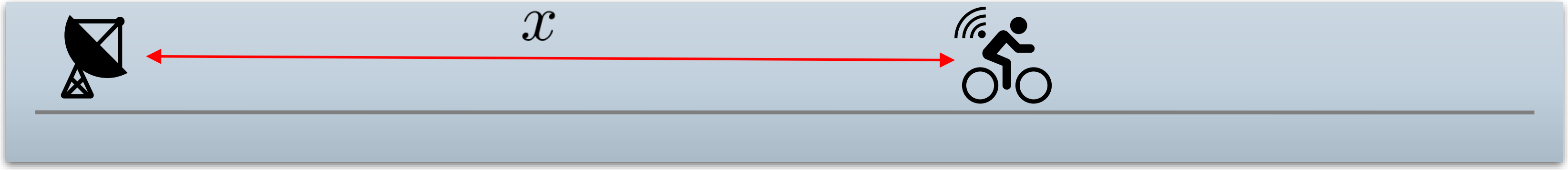
$$y_n = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_n \\ \dot{x}_n \end{bmatrix} + \begin{bmatrix} q_{1_n} \\ q_{2_n} \end{bmatrix}$$

$$y_n = H z_n + q_n$$

Kalman Filter Example

MLTS Exercise 06

Handy example - Initialization



x is the position of a rider
$\dot{x}$ is the rider's velocity

$z = \begin{bmatrix} x \\ \dot{x} \end{bmatrix}$ is the system's state

Initial conditions:

We need to define initial conditions for the state and the error covariance

$$z_0 = \begin{bmatrix} x_0 \\ \dot{x}_0 \end{bmatrix} = \begin{bmatrix} 0 \\ 5 \end{bmatrix}$$

$$\Sigma_0 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Notice: *We also need to define covariance matrices associated with r and q.*

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

$$Q = \begin{bmatrix} 3 & 0 \\ 0 & 3 \end{bmatrix}$$

Algorithmic view

- **Prediction (Time update)**

$$\bar{z}_n = F_n z_{n-1} + B_n u_n$$

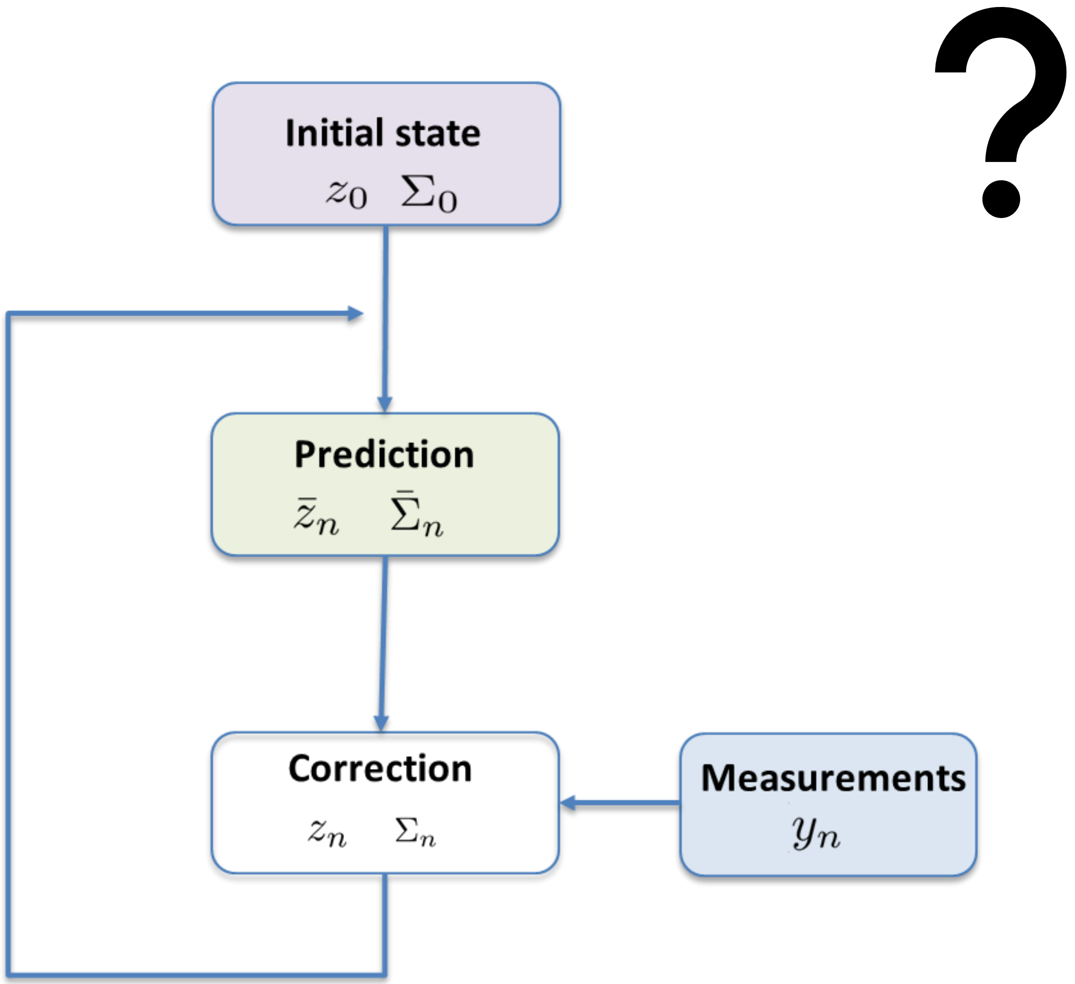
$$\bar{\Sigma}_n = F_n \Sigma_{n-1} F_n^T + R_n$$

- **Correction (Measurements update)**

$$K_n = \bar{\Sigma}_n H_n^T \left(H_n \bar{\Sigma}_n H_n^T + Q_n \right)^{-1}$$

$$z_n = \bar{z}_n + K_n \left(y_n - H_n \bar{z}_n \right)$$

$$\Sigma_n = (1 - K_n H_n) \bar{\Sigma}_n$$



Algorithmic view

- **Prediction (Time update)**

$$\bar{z}_n = F_n z_{n-1} + B_n u_n$$

$$\bar{\Sigma}_n = F_n \Sigma_{n-1} F_n^T + R_n$$

- **Correction (Measurements update)**

$$K_n = \bar{\Sigma}_n H_n^T \left(H_n \bar{\Sigma}_n H_n^T + Q_n \right)^{-1}$$

$$z_n = \bar{z}_n + K_n \left(y_n - H_n \bar{z}_n \right)$$

$$\Sigma_n = \left(1 - K_n H_n \right) \bar{\Sigma}_n$$

$\bar{z}$: a priori state
$\bar{\Sigma}$: a priori covariance
F: state transition matrix
B: user input transformation matrix
u: user input
R: process noise

z: a posteriori state
Σ: a posteriori covariance
K: Kalman gain (intermediate variable)
H: measurement matrix
Q: measurement noise
y: measurements

Kalman Filter Limitations

MLTS Exercise 06

Remember KF assumptions!

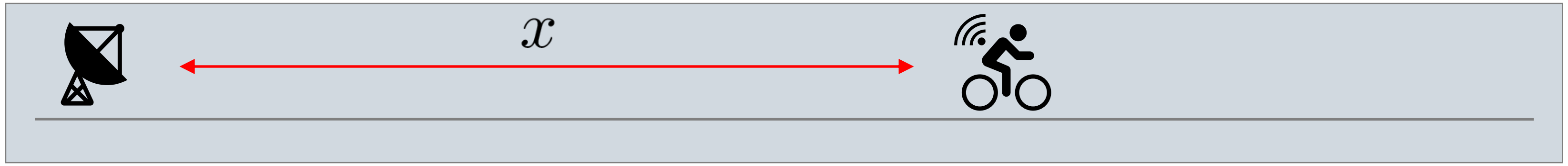
- Linear state transition model
- Linear measurement model
- Gaussian noise

If this assumptions do not hold, we apply other methods:

- Extended Kalman Filter (non-linear models, Gaussian noise)
- Particle Filtering (non-linear models, any noise distribution)

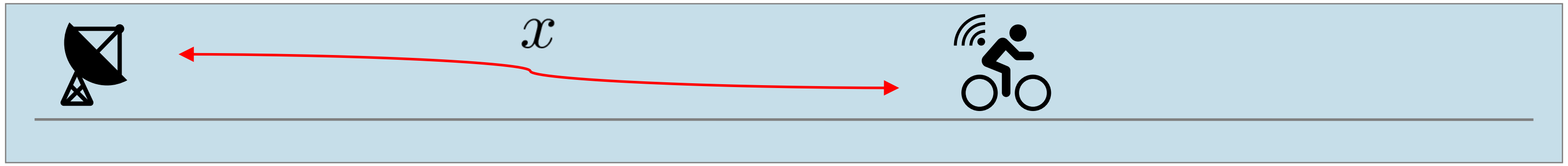
Extended Kalman Filter (EKF)

Extended Kalman Filter Example



$$z_n = F z_{n-1} + B u_n + r_n$$

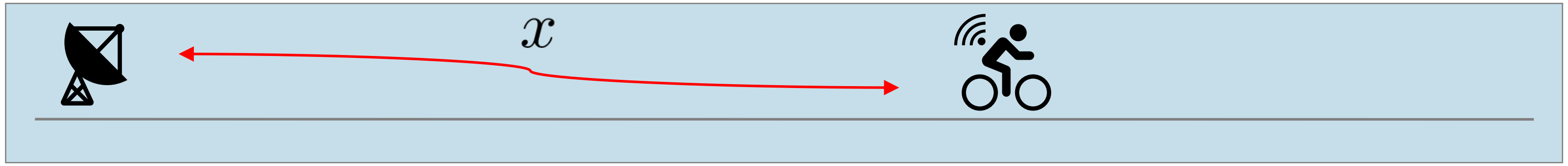
$$y_n = H z_n + q_n$$



$$z_n = f(z_{n-1}, u_n) + r_n$$

$$y_n = h(z_n) + q_n$$

Extended Kalman Filter Example

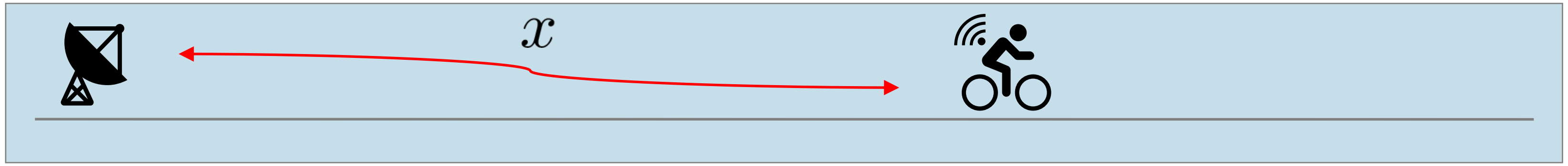


$$z_n = f(z_{n-1}, u_n) + r_n$$

$$y_n = h(z_n) + q_n$$

Assumption: Non-linear but differentiable State-transition model and/or Measurement model

Extended Kalman Filter Example



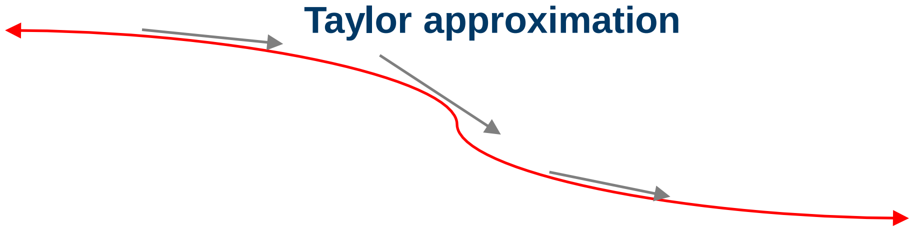
$$z_n = f(z_{n-1}, u_n) + r_n$$

$$y_n = h(z_n) + q_n$$

Linearization with 1st order Taylor Expansion around the current estimates:

$$J_{n-1}^{f} = \nabla f|_{z_{n-1},u_n}$$

$$J_{n}^{h} = \nabla h|_{z_n}$$



Algorithmic view

Prediction (Time update)

$$\bar{z}_n = f(z_{n-1}, u_n)$$

$$\bar{\Sigma}_n = J_n^f \Sigma_{n-1} {J_n^f}^T + R_n$$

Correction (Measurements update)

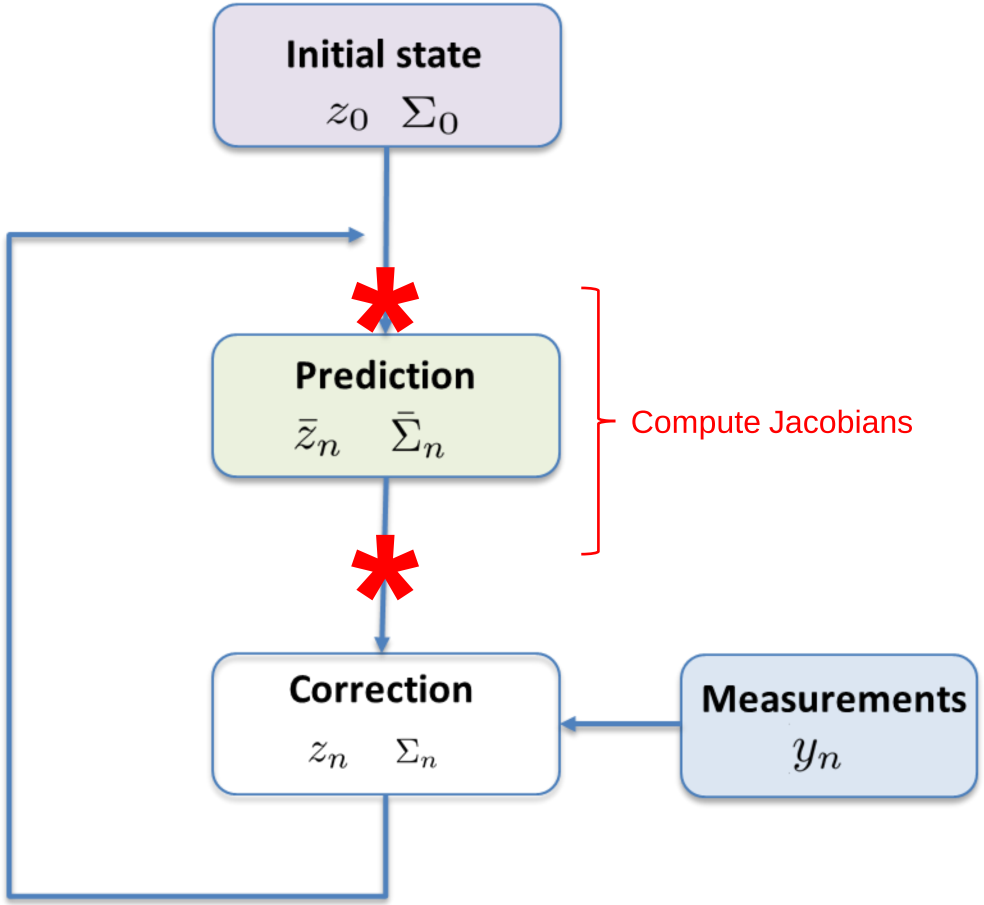
$$K_n = \bar{\Sigma}_n {J_n^h}^T \left(J_n^h \bar{\Sigma}_n {J_n^h}^T + Q_n \right)^{-1}$$

$$z_n = \bar{z}_n + K_n \left(y_n - h(\bar{z}_n) \right)$$

$$\Sigma_n = \left(1 - K_n J_n^h\right) \bar{\Sigma}_n$$

$\bar{z}$:	a priori state
$\bar{\Sigma}$:	a priori covariance
f:	non-linear state transition function
J^f:	Jacobian of f
R:	process noise
z:	a posteriori state
Σ:	a posteriori covariance
K:	Kalman gain (intermediate variable)
h:	non-linear measurement function
J^h:	Jacobian of h
Q:	measurement noise
y:	measurements

Algorithmic view



Extended Kalman Filter

Advantages:

- Deals with non-linear state-transition model and/or measurement model

Disadvantages:

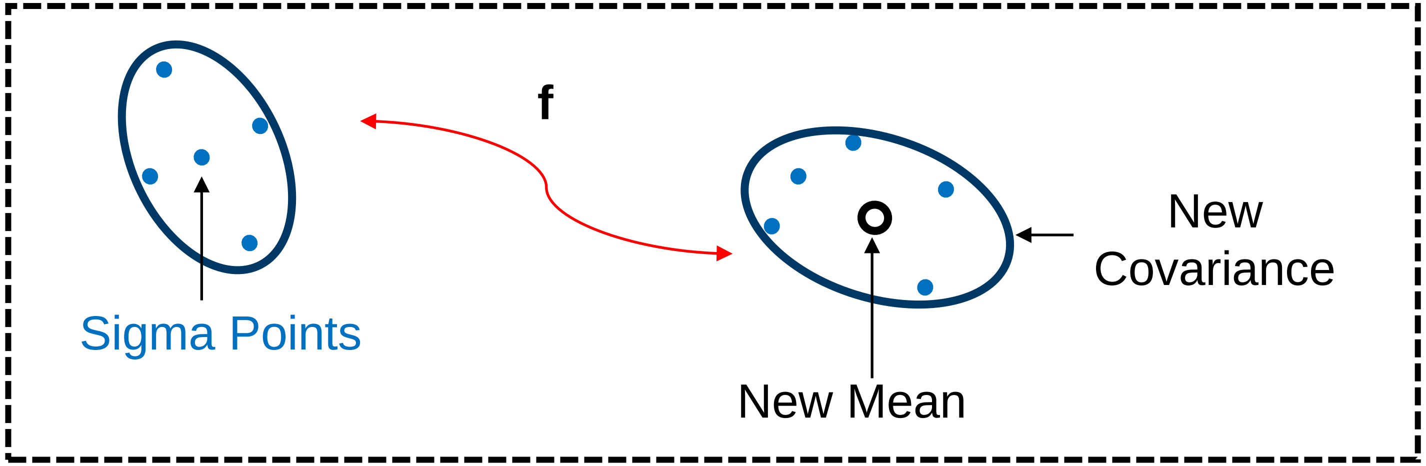
- Higher computational cost
- Non-optimal estimator (optimal only in the linear case, as for the KF)
- If the initial estimate is wrong, the system may diverge soon
- <u>It does not work if the models are highly non-linear</u>

Unscented Kalman Filter (UKF)

Unscented Kalman Filter

1. Makes use of deterministic sampling technique, namely **unscented transformation**
   - Pick up minimal sample of **sigma points**
2. Sigma points are **propagated** through a non-linear function **f**
   - New mean and covariance estimates

Unscented Transformation



Sigma points (simplest choice)

$$\{s^0, ..., s^{2D}\}_{n-1}$$

$$s_{n-1}^0 = z_{n-1}$$

$$s_{n-1}^i = z_{n-1} + \sqrt{\frac{D}{1-w_0}}A_i, i = 1, ..., D$$

$$s_{n-1}^{D+i} = z_{n-1} - \sqrt{\frac{D}{1-w_0}}A_i, i = 1, ..., D$$

$$w_i = \frac{1-w_0}{2D}, i = 1, ..., 2D$$

$$AA^T = \Sigma_{n-1}$$

Algorithmic view

Prediction (Time update)

$$\{s^0, \ldots, s^{2D}\}_{n-1} \qquad \bar{z}_n = \sum_{i=0}^{2D} w_i f(s^i_{n-1}) \qquad \bar{\Sigma}_n = \sum_{i=0}^{2D} w_i \left(f(s^i_{n-1}) - \bar{z}_n\right) \left(f(s^i_{n-1}) - \bar{z}_n\right)^T + R_n$$

Correction (Measurements update)

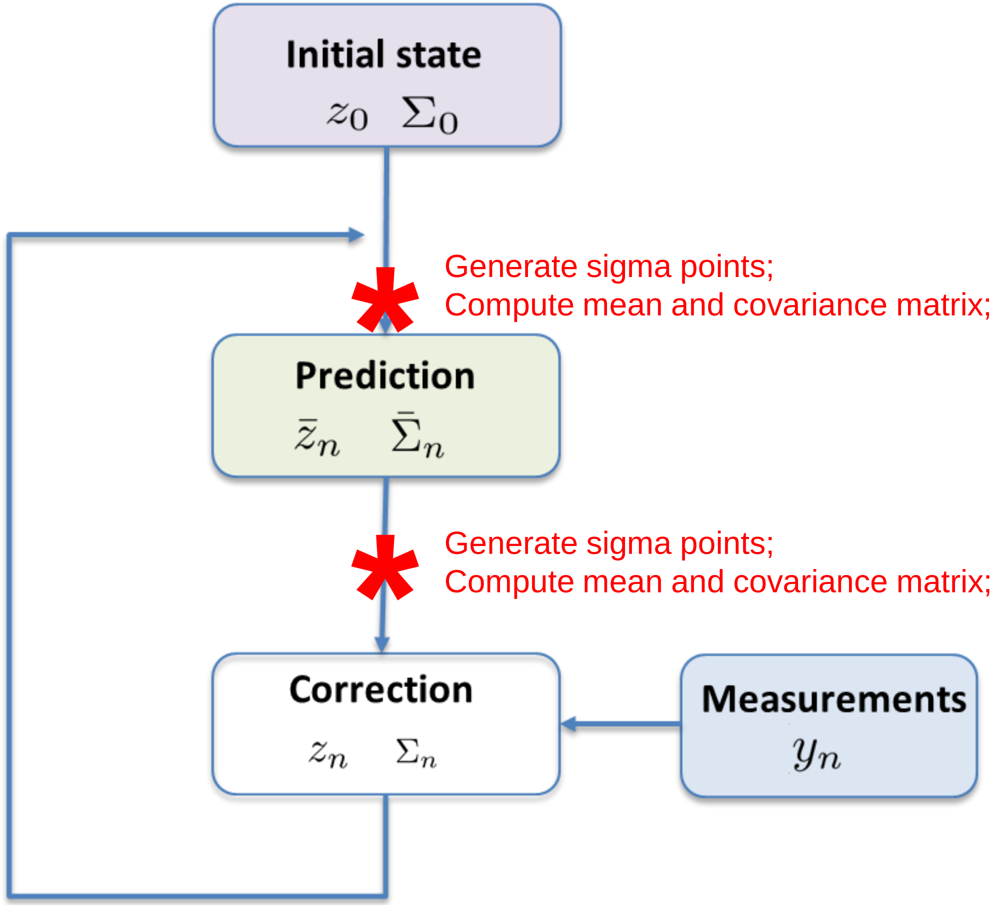
$$\{\bar{s}^0, \ldots, \bar{s}^{2D}\}_{n-1} \qquad \bar{y}_n = \sum_{i=0}^{2D} w_i h(\bar{s}^i_{n-1}) \qquad \bar{S}_n = \sum_{i=0}^{2D} w_i \left(h(\bar{s}^i_{n-1}) - \bar{y}_n\right) \left(h(\bar{s}^i_{n-1}) - \bar{y}_n\right)^T + Q_n$$

$$\bar{\Sigma}^{z,y}_n = \sum_{i=0}^{2D} w_i \left(\bar{s}^i_{n-1} - \bar{z}_n\right) \left(h(\bar{s}^i) - \bar{y}_n\right)^T$$

$$K_n = \bar{\Sigma}^{z,y}_n \bar{S}^{-1}_n$$

$$z_n = \bar{z}_n + K_n \left(y_n - \bar{y}_n\right) \qquad \Sigma_n = \bar{\Sigma}_n - K_n \bar{S}_n K_n^T$$

Algorithmic view



Unscented Kalman Filter

Advantages:

- For certain systems, UKF provides better estimates of mean and covariance
- No need to compute Jacobians
  (which is expansive or even not possible for difficult non-linear functions)

Disadvantages:

- Increased computational cost

Critical comparison

Critical comparison

?

Estimator	State-transition / Measurement models assumptions	Assumed noise distribution	Computational cost
Kalman Filter	Linear	Gaussian	Low
Extended Kalman Filter	Non-linear (but locally linear)	Gaussian	Low / Medium (depending on the difficulty of computing the Jacobian)
Unscented Kalman Filter	Non-linear	Gaussian	Medium

Critical comparison

Estimator	State-transition / Measurement models assumptions	Assumed noise distribution	Computational cost
Kalman Filter	Linear	Gaussian	Low
Extended Kalman Filter	Non-linear (but locally linear)	Gaussian	Low / Medium (depending on the difficulty of computing the Jacobian)
Unscented Kalman Filter	Non-linear	Gaussian	Medium
Particle Filter	**Non-linear**	**Non-Gaussian**	

Any questions?

Practice Questions

MLTS Exercise 06

- **Does the prediction step of the Kalman Filter increases or decreases the uncertainty of the state estimate?**
  - It decreases the uncertainty.
  - It increases the uncertainty.
- **Does the correction step of the Kalman Filter increases or decreases the uncertainty of the state estimate?**
  - It decreases the uncertainty.
  - It increases the uncertainty.

Practice Questions

MLTS Exercise 06

- **With a linear function, the extended Kalman filter is just a Kalman filter?**
  - Yes
  - No

- **Unscented and Extended Kalman filter are the same and only differ in their formulations?**
  - Yes
  - No