Kalman Filters

Part I: intro

Contents

- Why Kalman?
- Introduction via a hypothetical 1D case
- Matrix formulation: the linear case
- Next weekS: extensions
- Tuning
- Other issues/special cases
  - Partially observed controls
  - Intermittents updates
  - Continuous case
  - Matrix inverse approximation/update

Why Kalman filters?

Sensor Fusion

Kalman filters, grossly speaking, update distributional belief given uncertain measure updates: this includes but is not limited to estimating one parameter given, say 2 or 3 separate measures...

Base, robust case

Being relatively simple, Kalman filters are quite robust in practice and good benchmarks to evaluate/debug more sophisticated approaches

Important recipe

Sensor fusion means:

- Embedded (often)

Which means:

- Standalone C++ (often)

Having some basic recipes will help you!

Hypothetical 1D case...

System dynamics → Measure with noise → System dynamics

Linear discrete ODE with noise

$$x_{k+1} = A_k x_k + G_k d_k + w_k$$

state, input, noise (Q)

Uncertain measure

$$y_k = C_k x_k + v_k$$

observation, noise (R)

Linear discrete ODE with...

/REPEAT/

Linear Kalman filter equations (equations will be reviewed in detail this week)

$$\begin{aligned} x_{k+1} &= A_k x_k + G_k d_k + w_k \\ y_k &= C_k x_k + v_k \end{aligned}$$

(same state/measure update as before)

$$\begin{aligned} \hat{x}_k^- &= A_k \hat{x}_{k-1} + G_k d_k \\ P_k^- &= A_k P_{k-1} A_k^t + Q_{k-1} \end{aligned}$$

Estimate x as if there was no noise...
But add noise to P: error estimate (variance)

$$K_k = P_k^- C_k^t \left(C_k P_k^- C_k^t + R_k\right)^{-1}$$

This is ‘Kalman gain’ (will appear again, in different forms)

$$\begin{aligned} \hat{x}_k &= \hat{x}_k^- + K_k(y_k - C_k x_k) \\ P_k &= (I - K_k C_k) P_k^- \end{aligned}$$

Update x via measure ‘error’ / innovation
Update state error estimate (variance)

Linear Kalman filter equations
A simple interpretation (1D!)

$$
\begin{aligned}
x_{k+1} &= A_k x_k + G_k d_k + w_k \\
y_k &= C_k x_k + v_k
\end{aligned}
$$

$$
\begin{aligned}
\hat{x}_k^- &= A_k \hat{x}_{k-1} + G_k d_k \\
P_k^- &= A_k P_{k-1} A_k^t + Q_{k-1}
\end{aligned}
$$

$$
K_k = P_k^- C_k^t \left(C_k P_k^- C_k^t + R_k\right)^{-1}
$$

$$
\begin{aligned}
\hat{x}_k &= \hat{x}_k^- + K_k(y_k - C_k x_k) \\
P_k &= (I - K_k C_k) P_k^-
\end{aligned}
$$

$$
\begin{aligned}
e_k^- &= x_k - \hat{x}_k^- \\
e_k &= x_k - \hat{x}_k
\end{aligned}
$$

$$
\begin{aligned}
P_k^- &= E(e_k^- e_k^{-\,t}) \\
P_k = &= E(e_k e_k^t)
\end{aligned}
$$

Define error....

$$
\begin{aligned}
e_k &= e_k^- - K_k(v_k + C_k e_k^-) \\
P_k &= P_k^- + K_k^2(R_k + C_k^2 P_k^-) - 2K_k C_k P_k^- \\
argminP(K_k) &= \frac{P_k^- C_k}{C_k^2 P_k^- + R_k}
\end{aligned}
$$

Write P as function of K, optimize/minimize variance

Linear Kalman filter equations
A simple interpretation (1D!)

$$
\begin{aligned}
x_{k+1} &= A_k x_k + G_k d_k + w_k \\
y_k &= C_k x_k + v_k
\end{aligned}
$$

$$
\begin{aligned}
\hat{x}_k^- &= A_k \hat{x}_{k-1} + G_k d_k \\
P_k^- &= A_k P_{k-1} A_k^t + Q_{k-1}
\end{aligned}
$$

$$
K_k = P_k^- C_k^t (C_k P_k^- C_k^t + R_k)^{-1}
$$

$$
\begin{aligned}
\hat{x}_k &= \hat{x}_k^- + K_k (y_k - C_k x_k) \\
P_k &= (I - K_k C_k) P_k^-
\end{aligned}
$$

$$
\begin{aligned}
e_k^- &= x_k - \hat{x}_k^- \\
e_k &= x_k - \hat{x}_k
\end{aligned}
$$

$$
\begin{aligned}
P_k^- &= E(e_k^- e_k^{-t}) \\
P_k = &= E(e_k e_k^t)
\end{aligned}
$$

$$
\begin{aligned}
e_k &= e_k^- - K_k (v_k + C_k e_k^-) \\
P_k &= P_k^- + K_k^2 (R_k + C_k^2 P_k^-) - 2 K_k C_k P_k^-
\end{aligned}
$$

$$
argmin P(K_k) = \frac{P_k^- C_k}{C_k^2 P_k^- + R_k}
$$

SAME!

Linear formulation

Matrix formulation

$$
\begin{aligned}
x_{k+1} &= A_k x_k + G_k d_k + w_k \\
y_k &= C_k x_k + v_k
\end{aligned}
$$

$$
\begin{aligned}
\hat{x}_k^- &= A_k \hat{x}_{k-1} + G_k d_k \\
P_k^- &= A_k P_{k-1} A_k^t + Q_{k-1}
\end{aligned}
$$

$$
K_k = P_k^- C_k^t \left(C_k P_k^- C_k^t + R_k \right)^{-1}
$$

$$
\begin{aligned}
\hat{x}_k &= \hat{x}_k^- + K_k (y_k - C_k x_k) \\
P_k &= (I - K_k C_k) P_k^-
\end{aligned}
$$

Almost... exactly the same!

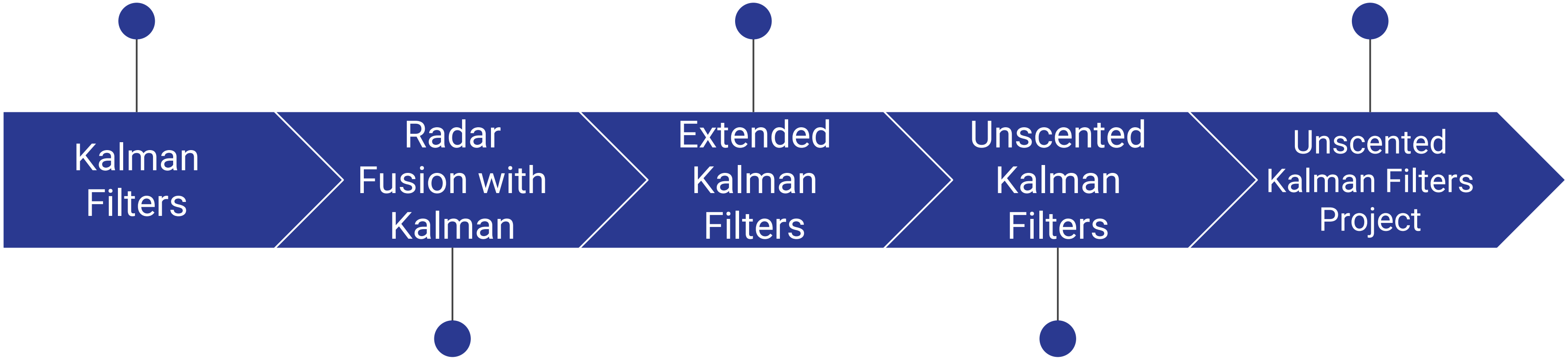
Preview: next weekS

Week 1, Introduction

Week 3.

Non-linear case (1)

Final project.



Week 2.

Coding a Kalman filter in C++

Week 4.

Non-linear case (2)

Important issue: Tuning

Linear Kalman filter : do you know Q and R? (equations will be reviewed in detail this week)

$$\begin{aligned} x_{k+1} &= A_k x_k + G_k d_k + w_k \\ y_k &= C_k x_k + v_k \end{aligned}$$

(same state/measure update as before)

$$\begin{aligned} \hat{x}_k^- &= A_k \hat{x}_{k-1} + G_k d_k \\ P_k^- &= A_k P_{k-1} A_k^t + Q_{k-1} \end{aligned}$$

Estimate x as if there was no noise...
But add noise to P: error estimate (variance)

$$K_k = P_k^- C_k^t \left(C_k P_k^- C_k^t + R_k \right)^{-1}$$

This is 'Kalman gain' (will appear again, in different forms)

$$\begin{aligned} \hat{x}_k &= \hat{x}_k^- + K_k (y_k - C_k x_k) \\ P_k &= (I - K_k C_k) P_k^- \end{aligned}$$

Update x via measure 'error' / innovation
Update state error estimate (variance)

Linear Kalman filter : do you know Q and R? (equations will be reviewed in detail this week)

One possible way to estimate: Maximum Likelihood (Zagrobelny & al. 2014)

$$
\begin{aligned}
x_{k+1} &= A x_k w_k \\
y &= C x_k + v \\
\begin{pmatrix} w \\ v \end{pmatrix} &\sim N\left(0, \begin{pmatrix} Q & 0 \\ 0 & R \end{pmatrix}\right)
\end{aligned}
$$

$$[some\, long\, derivation...]$$

$$\min_{R,Q} ln\, det P_{R,Q} + Y' P_{R,Q}^{-1} Y$$

Sparse! And likely to be amenable to 'approximations' In a few applications

$$s.t.\, Q, R \geq 0$$

$$P_{R,Q} = \sum_{i=1}^{N+K-1} \mathbb{O}_i Q \mathbb{O}_i' + \sum_{j=1}^{N} \mathbb{I}_j R \mathbb{I}_j'$$

(will require some matrix algorithm know how as the matrices are sparse, great rewards but 'pain and gain')

Other issues/special cases…

Other issues

More about it during the following weeks!

- Partially observed controls
  - Are updates always happening on a full vector?
- Intermittents updates
  - Is it true that time step is always constant?
- Continuous case
  - What about treating time as a real number?
- Matrix inverse approximation & update
  - Can we use approximations?

Contact

renoir42@yahoo.com

https://github.com/zeta1999/TeachingDemoKalman