

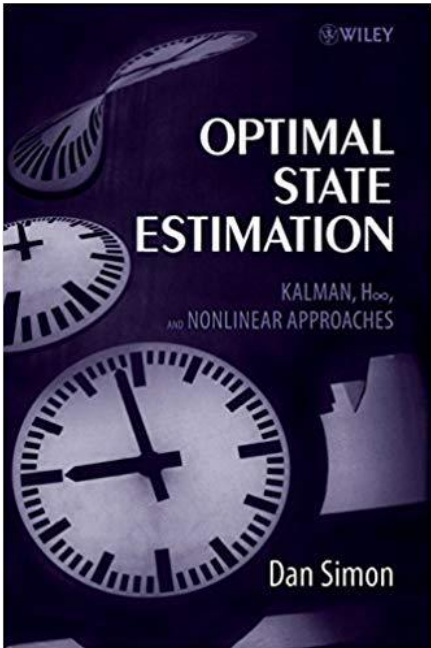
The Discrete-time Kalman Filter

Week 07
2022-04-11

Handong Global University
Smart Sensors and IoT Devices

0. References

- 1) **Optimal State Estimation: Kalman, H infinity, and nonlinear approaches**
Simon, Dan. Optimal state estimation: Kalman, H infinity, and nonlinear approaches. John Wiley & sons, 2006.
- 2) **An Introduction to the Kalman Filter**
Welch, Greg, and Gary Bishop. 'An introduction to the Kalman filter.' (1995): 41-95.
- 3) **Understanding the Basis of the Kalman Filter Via a Simple and Intuitive Derivation**
Faragher, Ramsey. 'Understanding the basis of the Kalman filter via a simple and intuitive derivation.' IEEE Signal processing magazine 29.5 (2012): 128-132.



An Introduction to the Kalman Filter

Greg Welch¹ and Gary Bishop²

TR 95-041
Department of Computer Science
University of North Carolina at Chapel Hill
Chapel Hill, NC 27599-3175

Updated: Monday, July 24, 2006

Abstract

In 1960, R.E. Kalman published his famous paper describing a recursive solution to the discrete-data linear filtering problem. Since that time, due in large part to advances in digital computing, the Kalman filter has been the subject of extensive research and application, particularly in the area of autonomous or assisted navigation.

The Kalman filter is a set of mathematical equations that provides an efficient computational (recursive) means to estimate the state of a process, in a way that minimizes the mean of the squared error. The filter is very powerful in several aspects: it supports estimations of past, present, and even future states, and it can do so even when the precise nature of the modeled system is unknown.

The purpose of this paper is to provide a practical introduction to the discrete Kalman filter. This introduction includes a description and some discussion of the basic discrete Kalman filter, a derivation, description and some discussion of the extended Kalman filter, and a relatively simple (tangible) example with real numbers & results.

¹ welch@cs.unc.edu, <http://www.cs.unc.edu/~welch>
² gbi@cs.unc.edu, <http://www.cs.unc.edu/~gb>

lecture NOTES

Ramsey Faragher

Understanding the Basis of the Kalman Filter Via a Simple and Intuitive Derivation

This article provides a simple and intuitive derivation of the Kalman filter with the aim of teaching this useful tool to students from disciplines that do not require a strong mathematical background. The most complicated level of mathematics required to understand this derivation is the ability to multiply two Gaussian functions together and reduce the result to a compact form.

The Kalman filter is over 50 years old but is still one of the most important and common data fusion algorithms in use today. Named after Rudolf E. Kalman, the great success of the Kalman filter is due to its small computational requirement, elegant recursive properties, and its status as the optimal estimator for one-dimensional linear systems with Gaussian error statistics [1]. Typical uses of the Kalman filter include smoothing noisy data and providing estimates of parameters of interest. Applications include global positioning system receivers, phase-locked loops in radio equipment, smoothing the output from laptop trackballs, and many more.

From a theoretical standpoint, the Kalman filter is an algorithm permitting exact inference in a linear dynamical system, which is a Bayesian model similar to a hidden Markov model but where the state space of the latent variables is continuous and where all latent and observed variables have a Gaussian distribution (often a multivariate Gaussian distribution). The aim of this lecture note is to permit people who find this description confusing or terrifying to understand the basis of the Kalman filter via a simple and intuitive derivation.

RELEVANCE

The Kalman filter [2] (and its variants such as the extended Kalman filter [3] and unscented Kalman filter [4]) is one of the most celebrated and popular data fusion algorithms in the field of information processing. The most famous early use of the Kalman filter was in the Apollo navigation computer that took Neil Armstrong to the moon, and (most importantly) brought him back. Today, Kalman filters are at work in every satellite navigation device, every smart phone, and many computer games.

THE KALMAN FILTER IS OVER 50 YEARS OLD BUT IS STILL ONE OF THE MOST IMPORTANT AND COMMON DATA FUSION ALGORITHMS IN USE TODAY.

The Kalman filter is typically derived using vector algebra as a minimum mean squared estimator [5], an approach suitable for students confident in mathematics but not one that is easy to grasp for students in disciplines that do not require strong mathematics. The Kalman filter is derived here from first principles considering a simple physical example exploiting a key property of the Gaussian distribution—specifically the property that the product of two Gaussian distributions is another Gaussian distribution.

PREREQUISITES

This article is not designed to be a thorough tutorial for a brand-new student to the Kalman filter, in the interests of being concise, but instead aims to provide tutors with a simple method of teaching the concepts of the Kalman filter to students who are not strong mathematicians. The reader is expected to be familiar with vector notation and terminology associated with Kalman filtering such as the state vector and covariance matrix. This article is aimed at those who need to teach the Kalman filter to others in a simple and intuitive manner, or for those who already have some experience with the Kalman filter but may not fully understand its foundations. This article is not intended to be a thorough and standalone education tool for the complete novice, as that would require a chapter, rather than a few pages, to convey.

PROBLEM STATEMENT

The Kalman filter model assumes that the state of a system at a time t evolved from the prior state at time $t-1$ according to the equation

$$\mathbf{x}_t = \mathbf{F}_t \mathbf{x}_{t-1} + \mathbf{B}_t \mathbf{u}_t + \mathbf{w}_t, \quad (1)$$

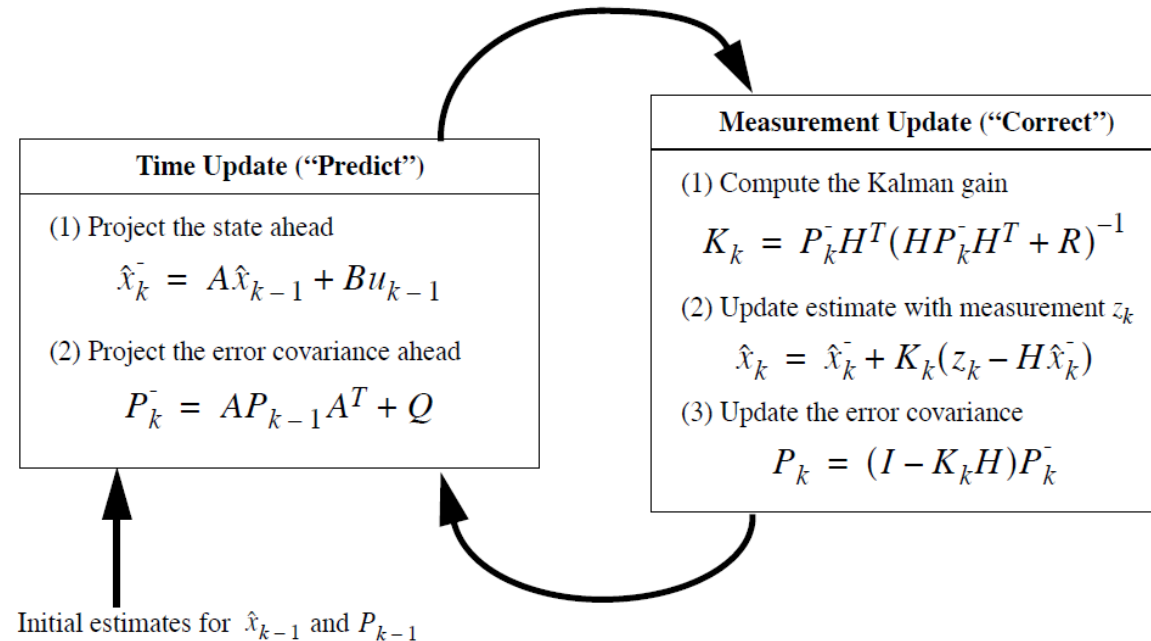
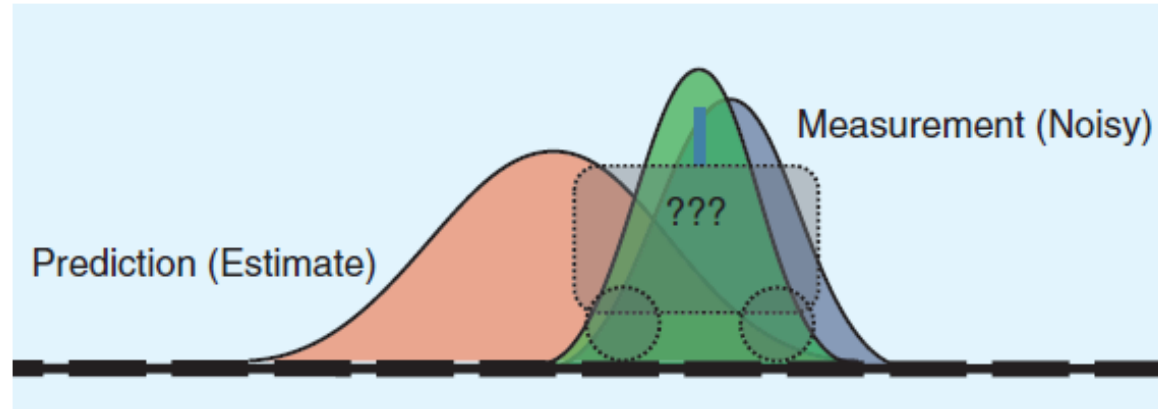
where:

- \mathbf{x}_t is the state vector containing the terms of interest for the system (e.g., position, velocity, heading) at time t
- \mathbf{F}_t is the vector containing any control inputs (steering angle, thrust, etc) affecting the system
- \mathbf{B}_t is the state transition matrix which applies the effect of each system state parameter at time $t-1$ on the system state at time t , e.g., the position and velocity at time $t-1$ both affect the position at time t
- \mathbf{u}_t is the control input matrix which applies the effect of each

IEEE SIGNAL PROCESSING MAGAZINE | 128 | SEPTEMBER 2012

1. Summary of the Kalman Filter

How to estimate the position of a car from an imperfect dynamic equation and a noisy measurement?



1. Summary of the Kalman Filter

2-1. Least Squares Method

- Weighted Cost Function with Covariance
- Recursive Expression

2-2. State-Space Equations of a Linear System

$$\begin{array}{ll} \text{➤ } CT: \begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ y(t) = Cx(t) + Du(t) \end{cases} & DT: \begin{cases} x[k+1] = Ax[k] + Bu[k] \\ y[k] = Cx[k] + Du[k] \end{cases} \end{array}$$

2-3. Predictor & Corrector Algorithm

- Runge-Kutta, K-means, Kalman Filter ...

2-1. Least Squares Method

- Equations of a constant vector, x

x : state variables(n elements)

v_k : random measurement noise @ k^{th} time step

y_k : noisy measurement @ k^{th} time step

$$\begin{cases} y_1 &= H_{11}x_1 + \cdots + H_{1n}x_n + v_1 \\ \vdots &= \vdots \\ y_k &= H_{k1}x_1 + \cdots + H_{kn}x_n + v_k \end{cases} \rightarrow y = Hx + v$$

- How to find an **optimized estimate of x, \hat{x}** ?

➤ Minimize the sum of squares of residuals: cost function, J

$$J = \varepsilon_{y_1}^2 + \varepsilon_{y_2}^2 + \cdots + \varepsilon_{y_k}^2 = \varepsilon_y^T \varepsilon_y = (y - H\hat{x})^T (y - H\hat{x})$$

$$\frac{\partial J}{\partial x} = -y^T H - y^T H + 2\hat{x}^T H^T H = 0$$

$$\therefore \hat{x} = (H^T H)^{-1} H^T y$$

2-1. Least Squares Method

Example

Voltage measurements with a noisy DMM

$$\begin{cases} y_1 &= x_1 + v_1 \\ \vdots &= \vdots \\ y_k &= x_1 + v_k \end{cases} \rightarrow y = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} x + v \quad H = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}$$

$$\hat{x} = (H^T H)^{-1} H^T y = \left([1 \quad \dots \quad 1] \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} \right)^{-1} [1 \quad \dots \quad 1] \begin{bmatrix} y_1 \\ \vdots \\ y_k \end{bmatrix}$$

$$\therefore \hat{x} = \frac{1}{k} (y_1 + \dots + y_k)$$

2-2. Weighted Least Squares Method

$$y = Hx + v$$

$$R = E(vv^T) = \begin{bmatrix} \sigma_1^2 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \sigma_k^2 \end{bmatrix}, \text{ covariance matrix}$$

Assume white noise: $E(v_i) = 0$, $E(v_i^2) = \sigma_i^2$, $E(v_i v_j) = 0$

- How to find an optimized estimate of x , \hat{x} ?
 - Minimize the sum of **weighted** squares of residuals

$$J = \frac{\varepsilon_{y1}^2}{\sigma_1^2} + \frac{\varepsilon_{y2}^2}{\sigma_2^2} + \cdots + \frac{\varepsilon_{yk}^2}{\sigma_k^2} = \varepsilon_y^T R^{-1} \varepsilon_y = (y - H\hat{x})^T R^{-1} (y - H\hat{x})$$

$$\frac{\partial J}{\partial x} = -y^T R^{-1} H - y^T R^{-1} H + 2\hat{x}^T H^T R^{-1} H = 0$$

$$\therefore \hat{x} = (H^T R^{-1} H)^{-1} H^T R^{-1} y$$

2-3. Recursive Weighted Least Squares Method

$$y_k = H_k x + v_k$$

$$\hat{x}_k = \hat{x}_{k-1} + K_k(y_k - \hat{y}_k^-)$$

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

x : state variables

v_k : random measurement noise @ k^{th} time step

y_k : noisy measurement @ k^{th} time step

K_k : correction gain @ k^{th} time step

\hat{y}_k^- : priori measurement estimate @ k^{th} time step

- How to find an estimate of x, \hat{x} after each measurement update?

➤ A **recursive** expression of the weighted least squares estimation

$$J_k = E[(x_1 - \hat{x}_1)^2] + \dots + E[(x_k - \hat{x}_k)^2] = E[\varepsilon_{x,k} \varepsilon_{x,k}^T] = \text{Tr}(P_k)$$

$$P_k = (I - K_k H_k) P_{k-1} (I - K_k H_k)^T + K_k R_k K_k^T$$

$$\frac{\partial J_k}{\partial K_k} = 2(I - K_k H_k) P_{k-1} (-H_k^T) + 2K_k R_k = 0$$

$$\therefore K_k = P_{k-1} H_k^T (H_k P_{k-1} H_k^T + R_k)^{-1}$$

$$\text{Tr}(A) = \sum A_{ii}$$

Sum of diagonal elements

$$P_k = E[\varepsilon_{x,k} \varepsilon_{x,k}^T]$$

Covariance matrix of error

2-3. Recursive Weighted Least Squares Method

$$\begin{aligned}E(\varepsilon_{x,k}) &= E(x - \hat{x}_k) \\&= E(x - \hat{x}_{k-1} - K_k(y_k - H_k \hat{x}_{k-1})) \\&= E(\varepsilon_{x,k-1} - K_k(H_k x + v_k - H_k \hat{x}_{k-1})) \\&= E(\varepsilon_{x,k-1} - K_k H_k (x - \hat{x}_{k-1}) - K_k v_k) \\&= E(\varepsilon_{x,k-1} - K_k H_k (\varepsilon_{x,k-1}) - K_k v_k) \\&= (I - K_k H_k) E(\varepsilon_{x,k-1}) - K_k E(v_k)\end{aligned}$$

$$\begin{aligned}J_k &= E[(x_1 - \hat{x}_1)^2] + \cdots + E[(x_k - \hat{x}_k)^2] \\&= E[\varepsilon_{x_1,k}^2 + \cdots + \varepsilon_{x_k,k}^2] \\&= E[\varepsilon_{x,k} \varepsilon_{x,k}^T] \\&= \text{Tr}(P_k)\end{aligned}$$

$$y_k = H_k x + v_k$$

$$\begin{aligned}\hat{x}_k &= \hat{x}_{k-1} + K_k(y_k - \hat{y}_k^-) \\&= \hat{x}_{k-1} + K_k(y_k - H_k \hat{x}_{k-1})\end{aligned}$$

2-3. Recursive Weighted Least Squares Method

$$\begin{aligned}P_k &= E[\varepsilon_{x,k} \varepsilon_{x,k}^T] \\&= E[(x_1 - \hat{x}_k)(x_1 - \hat{x}_k)^T] \\&= E\left[\{(I - K_k H_k) \varepsilon_{x,k-1} - K_k v_k\} \{(I - K_k H_k) \varepsilon_{x,k-1} - K_k v_k\}^T\right] \\&= (I - K_k H_k) E(\varepsilon_{x,k-1} \varepsilon_{x,k-1}^T) (I - K_k H_k)^T + K_k E(v_k v_k^T) K_k^T \\&= (I - K_k H_k) P_{k-1} (I - K_k H_k)^T + K_k R_k K_k^T\end{aligned}$$

$$\frac{\partial J_k}{\partial K_k} = \frac{\partial \text{Tr}(P_k)}{\partial K_k} = 2(I - K_k H_k) P_{k-1} (-H_k^T) + 2K_k R_k = 0$$

$$K_k = P_{k-1} H_k^T (H_k P_{k-1} H_k^T + R_k)^{-1}$$

$$P_k = E[\varepsilon_{x,k} \varepsilon_{x,k}^T]$$

Covariance matrix of error

K_k : correction gain

@ k^{th} time step

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

$$P_k = (I - K_k H_k) P_{k-1} (I - K_k H_k)^T + K_k R_k K_k^T$$

$$K_k = P_{k-1} H_k^T (H_k P_{k-1} H_k^T + R_k)^{-1}$$

2-3. Recursive Weighted Least Squares Method

Summary

1) $k = 0$

$$\hat{\mathbf{x}}_0 = E(\mathbf{x})$$

$$\mathbf{P}_0 = E[(\mathbf{x} - \hat{\mathbf{x}}_0)(\mathbf{x} - \hat{\mathbf{x}}_0)^T]$$

2) $k = 1, 2, \dots$

A. Measurement

$$\mathbf{y}_k = \mathbf{H}_k \mathbf{x} + \mathbf{v}_k$$

B. Estimation

$$\mathbf{K}_k = \mathbf{P}_{k-1} \mathbf{H}_k^T (\mathbf{H}_k \mathbf{P}_{k-1} \mathbf{H}_k^T + \mathbf{R}_k)^{-1}$$

$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_{k-1} + \mathbf{K}_k (\mathbf{y}_k - \mathbf{H}_k \hat{\mathbf{x}}_{k-1})$$

$$\mathbf{P}_k = (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k) \mathbf{P}_{k-1} (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k)^T + \mathbf{K}_k \mathbf{R}_k \mathbf{K}_k^T$$

\mathbf{x} : state variables

$\hat{\mathbf{x}}_k$: estimation of \mathbf{x} @ k^{th} time step

\mathbf{v}_k : measurement noise @ k^{th} time step

\mathbf{y}_k : noisy measurement @ k^{th} time step

white noise: $E(\mathbf{v}_i) = 0, E(\mathbf{v}_i^2) = \sigma_i^2, E(\mathbf{v}_i \mathbf{v}_j) = 0$

\mathbf{K}_k : correction gain @ k^{th} time step

\mathbf{R}_k : covariance matrix of measurement noise

\mathbf{P}_k : covariance matrix of estimation error

2-3. Recursive Weighted Least Squares Method

Example

Voltage measurement using a DMM with a variance of $(1V)^2$

1) $k = 0$ (assume the worst-case scenario)

$$\hat{x}_0 = E(x) = 0$$

$$P_0 = E[(x - \hat{x}_0)(x - \hat{x}_0)^T] = \infty$$

2) $k = 1$

$$y_1 = x + v_1 \quad (\rightarrow H = 1)$$

$$K_1 = P_0 H_1^T (H_1 P_0 H_1^T + R_1)^{-1} = P_0 (P_0 + R)^{-1} = \frac{\infty}{\infty + 1} = 1 \quad (\rightarrow R = 1^2)$$

$$\hat{x}_1 = \hat{x}_0 + K_1 (y_1 - \hat{x}_0) = 0 + 1(y_1 - 0) = y_1$$

$$P_1 = (I - K_1 H_1) P_0 (I - K_1 H_1)^T + K_1 R_1 K_1^T = (1 - 1)(\infty)(1 - 1)^T + 1 \cdot 1 \cdot 1^T = 0 + 1 = 1$$

2-3. Recursive Weighted Least Squares Method

Example

Voltage measurement using a DMM with a variance of $(1V)^2$

3) $k = 2$

$$y_2 = x + v_2$$

$$K_2 = P_1 H_2^T (H_2 P_1 H_2^T + R)^{-1} = P_1 (P_1 + R)^{-1} = \frac{1}{1+1} = \frac{1}{2}$$

$$\hat{x}_2 = \hat{x}_1 + K_2 (y_2 - \hat{x}_1) = y_1 + \frac{1}{2} (y_2 - y_1) = \frac{1}{2} (y_1 + y_2)$$

$$P_2 = (I - K_2 H_2) P_1 (I - K_2 H_2)^T + K_2 R K_2^T = \left(1 - \frac{1}{2}\right) (1) \left(1 - \frac{1}{2}\right)^T + \frac{1}{2} \cdot 1 \cdot \frac{1}{2}^T = \frac{1}{4} + \frac{1}{4} = \frac{1}{2}$$

4) $k = 3$

$$y_3 = x + v_3$$

$$K_3 = P_2 H_3^T (H_3 P_2 H_3^T + R)^{-1} = P_2 (P_2 + R)^{-1} = \frac{\frac{1}{2}}{\frac{1}{2} + 1} = \frac{1}{3}$$

$$\hat{x}_3 = \hat{x}_2 + K_3 (y_3 - \hat{x}_2) = \frac{1}{2} (y_1 + y_2) + \frac{1}{3} \left(y_3 - \frac{1}{2} (y_1 + y_2) \right) = \frac{1}{3} (y_1 + y_2 + y_3)$$

$$P_3 = (I - K_3 H_3) P_2 (I - K_3 H_3)^T + K_3 R K_3^T = \left(1 - \frac{1}{3}\right) \left(\frac{1}{2}\right) \left(1 - \frac{1}{3}\right)^T + \frac{1}{3} \cdot 1 \cdot \frac{1}{3}^T = \frac{2}{9} + \frac{1}{9} = \frac{1}{3}$$

2-3. Recursive Weighted Least Squares Method

Example

Voltage measurement using a DMM with a variance of $(1V)^2$

$k = k$ 일때

$$K_k = \frac{1}{k}$$

$$\hat{x}_k = \frac{1}{k} (y_1 + \cdots + y_k)$$

$$P_k = \frac{1}{k}$$

가장 최적화된 전압 추정 값은 계측 값의 평균

오차의 공분산은 $1/k$ 이며, 계측을 무한히 반복하면 0에 수렴하여 추정 값도 참값에 수렴

3. State-Space Representation of a linear System

- **State: 상태**

$t = t_0$ 의 초기값과 $t \geq t_0$ 의 입력이 주어졌을 때, 시스템 거동을 완전히 설명하는 변수의 최소 집합

- **State variables: 상태 변수**

시스템 거동을 완전히 표현하기 위한 최소 개수의 변수들
초기 값을 가지는 변수들이 대부분 상태변수

- **State space: 상태 공간**

상태 변수로 이루어진 n 차원의 공간

3. State-Space Representation of a linear System

- State-Space Equations

$$CT: \begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ y(t) = Cx(t) + Du(t) \end{cases}, \quad DT: \begin{cases} x[k+1] = Ax[k] + Bu[k] \\ y[k] = Cx[k] + Du[k] \end{cases}$$

미분방정식을 행렬형태로 표현

MIMO, Non-linear, Time-variant system을 취급하기에 유용

컴퓨터 시뮬레이션이 쉽다

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \text{ 상태 벡터}$$

$$u = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_r \end{bmatrix}, \text{ 입력 벡터} \quad \text{or} \quad \text{제어 벡터} \quad y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}, \text{ 출력 벡터}$$

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix}$$

상태 행렬, $n \times n$
(state matrix)

$$B = \begin{bmatrix} b_{11} & b_{12} & \cdots & b_{1r} \\ b_{21} & b_{22} & \cdots & b_{2r} \\ \vdots & \vdots & & \vdots \\ b_{n1} & b_{n2} & \cdots & b_{nr} \end{bmatrix}$$

입력 행렬, $n \times r$ or 제어 행렬
(input matrix)

$$C = \begin{bmatrix} c_{11} & c_{12} & \cdots & c_{1n} \\ c_{21} & c_{22} & \cdots & c_{2n} \\ \vdots & \vdots & & \vdots \\ c_{m1} & c_{m2} & \cdots & c_{mn} \end{bmatrix}$$

출력 행렬, $m \times n$
(output matrix)

$$D = \begin{bmatrix} d_{11} & d_{12} & \cdots & d_{1r} \\ d_{21} & d_{22} & \cdots & d_{2r} \\ \vdots & \vdots & & \vdots \\ d_{m1} & d_{m2} & \cdots & d_{mr} \end{bmatrix}$$

직접전송 행렬 or 피드포워드 행렬
(direct transmission matrix)

3. State-Space Representation of a linear System

Discrete-time System

x_k : state of a system

u_k : known input

w_k : Gaussian zero mean white noise with covariance Q_k

$$x_k = F_{k-1}x_{k-1} + G_{k-1}u_{k-1} + w_{k-1}$$

$$x_1 = F_0x_0 + G_0u_0 + w_0$$

$$x_2 = F_1x_1 + G_1u_1 + w_1 = F_1F_0x_0 + (F_1G_0u_0 + G_1u_1) + (F_1w_0 + w_1)$$

$$x_3 = F_2x_2 + G_2u_2 + w_2 = F_2F_1F_0x_0 + (F_2F_1G_0u_0 + F_2G_1u_1 + G_2u_2) + (F_2F_1w_0 + F_2w_1 + w_2)$$

$$x_k = F_{k,0}x_0 + \sum_{i=0}^{k-1}(F_{k,i+1}G_iu_i + F_{k,i+1}w_i)$$

$$\text{Where, } F_{k,i} = \begin{cases} F_{k-1}F_{k-2} \dots F_i & , k > i \\ I & , k = i \\ 0 & , k < i \end{cases}$$

2-3. Recursive Weighted Least Squares Method

$$\begin{aligned}\bar{x}_k &= E(x_k) = E(F_{k-1}x_{k-1} + G_{k-1}u_{k-1} + w_{k-1}) \\ &= F_{k-1}\bar{x}_{k-1} + G_{k-1}u_{k-1} + 0\end{aligned}$$

$$\begin{aligned}P_k &= E[(x - \hat{x}_k)(x - \hat{x}_k)^T] \\ &= F_{k-1}P_{k-1}F_{k-1}^T + Q_{k-1}\end{aligned}$$

If steady state,

$$P_k = P_{k-1} \rightarrow \therefore P = FPF^T + Q$$

If the system is stable, we get a unique solution

$$\bar{x} = F\bar{x} + Gu \rightarrow \bar{x} = (I - F)^{-1}Gu$$

$$P = \sum_{i=0}^{\infty} (F^i)Q(F^T)^i$$

3. State-Space Representation of a linear System

Example

- 포식자는 매일 전체 포식자 인구의 20%가 경쟁에서 살아남으며, 전체 먹이 인구의 40%만큼 증가
- 먹이는 매일 전체 포식자 인구의 40%만큼 감소하며, 외부에서 1만큼의 유입이 있음
- 초기 포식자 인구는 10마리, 초기 먹이 인구는 20마리로 설정
- 각 모델은 완벽하지 않아 각각 1,2 만큼의 분산이 있음

상태 공간 방정식을 통하여 steady-state 상태일 때의 인구 변화를 계산하라.

3. State-Space Representation of a linear System

Population of a predator:

$$x_1[k + 1] = x_1[k] - 0.8x_1[k] + 0.4x_2[k] + w_1[k]$$

Population of a prey:

$$x_2[k + 1] = x_2[k] - 0.4x_1[k] + u[k] + w_2[k]$$

Initial Condition

$$x_1[0] = 10, x_2[0] = 10, u[k] = 1, w \sim (0, Q), Q = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}$$

State-Space Equation

$$x[k + 1] = \begin{bmatrix} x_1[k + 1] \\ x_2[k + 1] \end{bmatrix} = \begin{bmatrix} 0.2 & 0.4 \\ -0.4 & 1 \end{bmatrix} \begin{bmatrix} x_1[k] \\ x_2[k] \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} 1 + \begin{bmatrix} w_1[k] \\ w_2[k] \end{bmatrix}$$

3. State-Space Representation of a linear System

Since the system is stable:

$$\bar{x}_{\infty} = (I - F)^{-1}Gu = \left(\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} - \begin{bmatrix} 0.2 & 0.4 \\ -0.4 & 1 \end{bmatrix} \right)^{-1} \begin{bmatrix} 0 \\ 1 \end{bmatrix} 1 = \begin{bmatrix} 2.5 \\ 5 \end{bmatrix}$$

$$P = \sum_{i=0}^{\infty} (F^i)Q(F^T)^i = \begin{bmatrix} 2.881 & 3.076 \\ 3.076 & 7.959 \end{bmatrix}$$

Steady-state 일 때,

포식자는 2.5마리, 먹이는 5마리에서 수렴

4. Discrete Kalman Filter

$$x_k = F_{k-1}x_{k-1} + G_{k-1}u_{k-1} + w_{k-1}$$

$$y_k = H_k x_k + v_k$$

1) $k = 0$

$$\hat{x}_0^+ = E(x_0)$$

$$P_0^+ = E[(x - \hat{x}_0)(x - \hat{x}_0)^T]$$

2) $k = 1, 2, \dots$

A. A priori(predictor)

$$\hat{x}_k^- = F_{k-1}\hat{x}_{k-1}^+ + G_{k-1}u_{k-1}$$

$$P_k^- = F_{k-1}P_{k-1}^+F_{k-1}^T + Q_{k-1}$$

B. A posteriori(corrector)

$$K_k = P_k^- H_k^T (H_k P_k^- H_k^T + R_k)^{-1}$$

$$\hat{x}_k = \hat{x}_k^- + K_k(y_k - H_k \hat{x}_k^-)$$

$$P_k = (I - K_k H_k) \mathbf{P}_k^- (I - K_k H_k)^T + K_k \mathbf{R}_k K_k^T$$

x : state variables

u : system input

w : process noise

white noise: $E(v_i) = 0, E(v_i^2) = \sigma_i^2, E(v_i v_j) = 0$

Q_k : covariance matrix of process noise

y : noisy measurement

v : measurement noise

white noise: $E(v_i) = 0, E(v_i^2) = \sigma_i^2, E(v_i v_j) = 0$

R_k : covariance matrix of measurement noise

\hat{x}_k^- : a priori estimate @ k^{th} time step

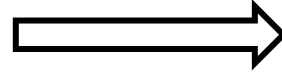
\hat{x}_k^+ : a posteriori estimate @ k^{th} time step

K_k : correction gain @ k^{th} time step

P_k : covariance matrix of estimation error

5. Digital gyroscope & accelerometer application

$$\begin{aligned}x_k &= F_{k-1}x_{k-1} + G_{k-1}u_{k-1} + w_{k-1} \\ y_k &= H_k x_k + v_k\end{aligned}$$



$$\begin{aligned}x_{k+1} &= Fx_k + w_k \\ y_k &= Hx_k + v_k\end{aligned}$$

$$x_{k+1} = Fx_k + w_k$$

$$\Rightarrow \begin{bmatrix} \theta_{k+1} \\ \dot{\theta}_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \theta_k \\ \dot{\theta}_{gyro} \end{bmatrix} + \begin{bmatrix} w_{process} \\ w_{gyro} \end{bmatrix}$$

$$y_k = Hx_k + w_k$$

$$\Rightarrow [\theta_k] = [1 \quad 0] \begin{bmatrix} \theta_{acc} \\ \dot{\theta}_k \end{bmatrix} + [v_{acc}]$$

Assumptions:

- 1) No input(ignore random, impulse-like inputs by your finger)
- 2) State equation: ω from gyroscope & Euler method
- 3) Measurement equation: θ from accelerometer
- 4) $Q = \begin{bmatrix} \sigma_{process}^2 & 0 \\ 0 & \sigma_{gyro}^2 \end{bmatrix}, R = [\sigma_{acc}^2]$
 - Uncertainties for the two sensors and Euler method

5. Digital gyroscope & accelerometer application

1) $k = 0$

$$\hat{x}_0^+ = E(x_0)$$

$$P_0^+ = E[(x - \hat{x}_0)(x - \hat{x}_0)^T]$$

2) $k = 1, 2, \dots$

A. A priori(predictor): gyroscope

$$\begin{bmatrix} \theta_k^- \\ \dot{\theta}_k^- \end{bmatrix} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \theta_{k+1}^+ \\ \dot{\theta}_{gyro} \end{bmatrix}$$

$$P_k^- = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} P_{k-1}^+ \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix}^T + \begin{bmatrix} \sigma_{process}^2 & 0 \\ 0 & \sigma_{gyro}^2 \end{bmatrix}$$

B. A posteriori(corrector): accelerometer fusion

$$K_k = P_k^- [1 \ 0]^T ([1 \ 0] P_k^- [1 \ 0]^T + [\sigma_{acc}^2])^{-1}$$

$$\begin{bmatrix} \theta_k^+ \\ \dot{\theta}_k^+ \end{bmatrix} = \begin{bmatrix} \theta_k^- \\ \dot{\theta}_k^- \end{bmatrix} + K_k (\theta_{acc} - [1 \ 0] \begin{bmatrix} \theta_k^- \\ \dot{\theta}_k^- \end{bmatrix})$$

$$P_k^+ = \left(\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} - K_k [1 \ 0] \right) P_k^- \left(\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} - K_k [1 \ 0] \right)^T + K_k [\sigma_{acc}^2] K_k^T$$