EEET 4071 Advanced Control (2021)

Lecture 7: Kalman filter

Dr. Yee Wei Law ⟨yeewei.law@unisa.edu.au⟩

Contents

1 Introduction

In the previous lecture, we discussed observability and observer design. For observer design, we now know how to design a Luenberger observer, which is a full-order closed-loop observer. However, the Luenberger observer is susceptible to measurement noise. This is where the Kalman filter comes in.

The Kalman filter is an optimal observer, and is often referred to as a *linear quadratic estimator*, because it minimizes a *quadratic* function of the estimation error for a *linear* dynamic system with white measurement noise and white disturbance noise (the meaning of "white" will be clarified later). The "filter" in "Kalman filter" refers to the solution of an inverse problem, where we are given the values of the dependent variables to find the values of the independent variables. Figure 1 shows the distinction between prediction, filtering and smoothing. Whereas the Kalman filter is a discrete-time estimator, the Kalman-Bucy filter, invented after the Kalman filter, is a continuous-time filter. This lecture covers only the Kalman filter and its extension for nonlinear processes, the extended Kalman filter.

The Kalman filter is one of the greatest achievements in engineering in the 20th century. As a state estimation tool in control theory, the Kalman filter and its extensions are widely used in the predictive design of estimation and control systems — these include system identification, robotics, process control, power system control, and the tracking and navigation of all sorts of vehicles. In fact, the earliest applications of the Kalman filter were found in aerospace [GA15, p. 18], where it was used for the guidance and navigation of the Apollo spacecraft and the C-5A transport aircraft (see Figure 2). In short, it led humankind into the Space Age.

Quiz 1

What are ArduPilot and PX4? Do they use the extended Kalman filter? If yes, what do they use it for?

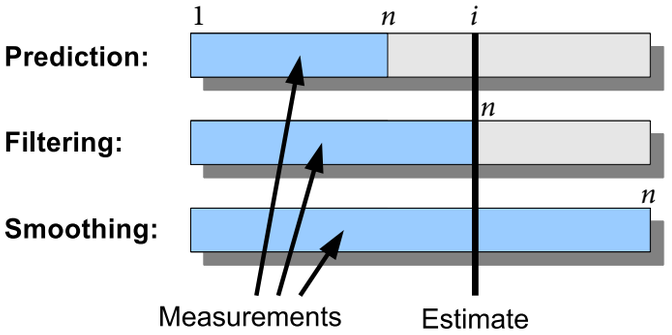


Figure 1: Given samples from $t = 1$ to $t = n$, "prediction" estimates the value of the independent variable at $t = i$, where $i > n$; "filtering" estimates the value of the independent variable at $t = n$; "smoothing" estimates the value of the independent variable at $t = i$, where $i < n$.

Figure 2: A C-5A military transport aircraft (U.S. Air Force photo by Brett Snow [Public domain], via Wikimedia Commons).

As a signal processing and time series analysis tool in estimation theory, which is used in nearly every branch of science or engineering, the Kalman filter and its extensions can be found in a diverse range of applications, including global positioning, radars, communications, voice recognition, speech enhancement, computer vision, video stabilization, surveying, econometrics, transportation planning, biomedical research, risk assessment, fault diagnosis, earthquake prediction, condition-based maintenance, performance (or health or product quality) monitoring, groundwater flow and contaminant transport modeling [Che12], [HRW12], [Gib11, p. 1], [DDS14, Sect. 6.1]. The list of applications seems inexhaustible. In one of the course practicals, we will learn how to apply Kalman filtering to attitude/orientation estimation.

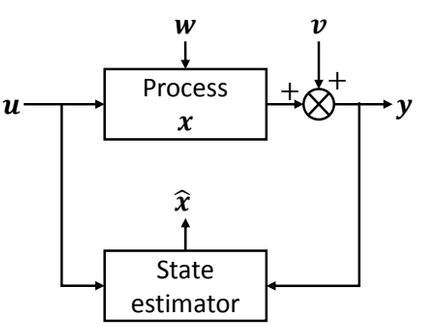


Figure 3: State estimation in the presence of process noise $\boldsymbol{w}$ and measurement noise $\boldsymbol{v}$.

The problem we are trying to solve in this lecture is that of *state estimation*. Figure 3 shows a process or plant taking input $\boldsymbol{u}$ and describable using states $\boldsymbol{x}$. The states are subjected to *process noise* $\boldsymbol{w}$, and the output measurements are subjected to *measurement noise* $\boldsymbol{v}$. This lecture addresses the problem of estimating $\boldsymbol{x}$ in discrete time for the case where the process, $\boldsymbol{w}$ and $\boldsymbol{v}$ satisfy the *linear Gaussian model*. The ensuing discussion will first cover the essential probabilistic and statistical concepts to the point where we can explain the linear Gaussian model, and then informally derive the Kalman filter algorithm. After that, we will introduce the steady-state Kalman filter and some sample applications. This lecture ends by introducing the extended Kalman filter.

2 Revision of essential probabilistic and statistical concepts

Noise is by definition probabilistic. The standard mathematical representation of a noise signal is a *random variable*.

Definition(s): Random variable [MC12, Definition 2.9]

A *random variable* is a real-valued function of the elements of a *sample space*. Given an experiment with sample space S, a random variable X maps each possible outcome $\xi \in S$ to a real number $X(\xi)$ as specified by some rule.

- If X maps to a finite number of values, then X is a *discrete random variable*.
- If X maps to an infinite number of values, then X is a *continuous random variable*.

In this lecture, we are only concerned with continuous random variables. X is written as $X(t)$ when it changes with time, so we can use $X(t)$ to represent a noise signal. An example of a continuous random variable can be defined for the random experiment of throwing a javelin in a 100 m-long field. The sample space is $S = [0, 100]$. Define outcome $\xi \in S$ as where the javelin lands, then we can define a random variable X such that $X(\xi) = x$, where $x = \xi$ (this is an example of "some rule" in the definition above). Thus, a simpler (non-rigorous) way to understand a random variable is a variable (X), rather than a function, that takes on a value that is *less than* a certain value (x) at a certain probability — this probability is called a *distribution function*.

Quiz 2

In the preceding discussion, why "takes on a value that is *less than* a certain value", rather than simply "takes on a certain value"?

Definition(s): Distribution function [GRG01, Sects. 2.1 and 2.3]

The *distribution function* of a random variable X is the function $F_X : \mathbb{R} \mapsto [0, 1]$ given by $F_X(x) = \Pr\{X \leq x\}$. If X is continuous, the function $p_X : \mathbb{R} \mapsto [0, \infty]$ whose integral gives the distribution function is called the *probability density function* (pdf), i.e.,

$$F_X(x) = \int_{-\infty}^{x} p_X(u)\,\mathrm{d}u, \quad x \in \mathbb{R}.$$

We write $X \sim p_X$ to mean X follows the pdf p_X.

In this lecture, we are only concerned with pdfs and only Gaussian/normal pdfs. The pdf of a random variable can be characterized by two parameters:

- *mean*: $\mathsf{E}\{X\}$
- *variance*: $\mathsf{Var}\{X\} \stackrel{\text{def}}{=} \mathsf{E}\left\{(X - \mathsf{E}\{X\})^2\right\} = \mathsf{E}\{X^2\} - \mathsf{E}\{X\}^2$.

While the mean captures the expected value of the random variable, the variance captures the "spread" of the pdf. If random variable X follows the Gaussian distribution with mean $\mu = \mathsf{E}\{X\}$ and variance $\sigma^2 = \mathsf{Var}\{X\}$, then we write $X \sim \mathcal{N}(\mu, \sigma^2)$, and X has the pdf

$$p_X(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-\frac{(x-\mu)^2}{2\sigma^2}\right]. \tag{1}$$

When there are multiple noise signals, such as the case with $\boldsymbol{w}$ and $\boldsymbol{v}$ in Figure 3, we need to use *random vectors*. A random vector is simply a vector of random variables. Whereas a random variable follows a *univariate* pdf, a random vector follows a *multivariate* pdf. The multivariate pdf of the random vector $\boldsymbol{X} = [X_1\ X_2\ \cdots\ X_n]^\top$ is given by the *joint pdf* of $X_1, X_2, \ldots, X_n$, denoted $p_{\boldsymbol{X}}$. The mean vector of $p_{\boldsymbol{X}}$ is given by the means of the elements of X, i.e.,

$$\mathsf{E}\{\boldsymbol{X}\} = \left[\mathsf{E}\{X_1\}\ \mathsf{E}\{X_2\}\ \cdots\ \mathsf{E}\{X_n\}\right]^\top.$$

However, for random vectors, the concept of variance is replaced by *covariance matrix*. If random vector $\boldsymbol{X}$ follows the n-variate Gaussian distribution with mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$, then we write $\boldsymbol{X} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, and $\boldsymbol{X}$ has the pdf

$$p_{\boldsymbol{X}}(\boldsymbol{x}) = \frac{1}{(2\pi)^{n/2}\det(\boldsymbol{\Sigma})^{1/2}} \exp\left[-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right]. \tag{2}$$

In Sect. 2.1, we will see that a valid covariance matrix $\boldsymbol{\Sigma}$ ensures $\det(\boldsymbol{\Sigma}) > 0$. The Gaussian distribution has some interesting and useful properties, the best known of which is that any affine combination of Gaussian random vectors is also a Gaussian random vector.

Theorem 1: [KC14, Theorem 3.6]

Let $\boldsymbol{X}_i \sim \mathcal{N}(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$, $i = 1, \ldots, N$, be *independent* n_i-dimensional random vectors. If $\boldsymbol{Y} = \sum_{i=1}^{N}(\mathbf{A}_i\boldsymbol{X}_i) + \boldsymbol{b}$, where $\mathbf{A}_i$ are $n \times n_i$ constant matrices and $\boldsymbol{b}$ is an n-dimensional constant vector, then

$$\boldsymbol{Y} \sim \mathcal{N}\left(\sum_{i=1}^{N}\mathbf{A}_i\boldsymbol{\mu}_i + \boldsymbol{b},\ \sum_{i=1}^{N}\mathbf{A}_i\boldsymbol{\Sigma}_i\mathbf{A}_i^\top\right).$$

Quiz 3

If random variable X follows the standard normal distribution, what is the variance of random variable $2X$?

Quiz 4

Suppose $\boldsymbol{X}_1 \sim \mathcal{N}(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1)$ and $\boldsymbol{X}_2 \sim \mathcal{N}(\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2)$ are mutually independent. If $\boldsymbol{Y} = \mathbf{H}_1\boldsymbol{X}_1 + \mathbf{H}_2\boldsymbol{X}_2$, where $\mathbf{H}_1$ and $\mathbf{H}_2$ are constant matrices of appropriate dimensions, then what is the pdf of $\boldsymbol{Y}$?

In the ensuing subsection, we shall look at the definition and properties of the covariance matrix in details.

2.1 Covariance matrix

Let us first define covariance before we define covariance matrix. The *covariance* between random variables X and Y is defined as

$$\mathsf{Cov}\{X, Y\} \overset{\text{def}}{=} \mathsf{E}\left\{(X - \mathsf{E}\{X\})(Y - \mathsf{E}\{Y\})\right\} = \mathsf{E}\{XY\} - \mathsf{E}\{X\}\,\mathsf{E}\{Y\}. \tag{3}$$

Between two random variables, covariance measures their correlation, i.e., how much they change together. Simplistically speaking, two variables are correlated if they wax and wane together.

Definition(s): Covariance matrix

The *covariance matrix* between random vectors $\boldsymbol{X} = [X_1 \ \cdots \ X_n]^\top$ and $\boldsymbol{Y} = [Y_1 \ \cdots \ Y_n]^\top$ is defined as

$$\mathsf{Cov}\{\boldsymbol{X}, \boldsymbol{Y}\} \overset{\text{def}}{=} \mathsf{E}\left\{(\boldsymbol{X} - \mathsf{E}\{\boldsymbol{X}\})(\boldsymbol{Y} - \mathsf{E}\{\boldsymbol{Y}\})^\top\right\} = \mathsf{E}\{\boldsymbol{X}\boldsymbol{Y}^\top\} - \mathsf{E}\{\boldsymbol{X}\}\,\mathsf{E}\{\boldsymbol{Y}\}^\top, \tag{4}$$

that is, the (i, j)th element of $\mathsf{Cov}\{\boldsymbol{X}, \boldsymbol{Y}\}$ is $\mathsf{Cov}\{X_i, Y_j\}$.

The *covariance matrix* of a multivariate pdf $p_{\boldsymbol{X}}$, denoted $\boldsymbol{\Sigma}_{\boldsymbol{X}}$, is the covariance matrix

between the random vector $\boldsymbol{X}$ and itself, i.e.,

$$\boldsymbol{\Sigma}_{\boldsymbol{X}} = \mathsf{Var}\{\boldsymbol{X}\} \stackrel{\text{def}}{=} \mathsf{Cov}\{\boldsymbol{X}, \boldsymbol{X}\} = \mathsf{E}\{\boldsymbol{X}\boldsymbol{X}^\top\} - \mathsf{E}\{\boldsymbol{X}\}\,\mathsf{E}\{\boldsymbol{X}\}^\top. \tag{5}$$

Note:

- Whereas $\boldsymbol{X}^\top\boldsymbol{Y}$ is an inner product (giving a scalar), $\boldsymbol{X}\boldsymbol{Y}^\top$ is an outer product (giving a matrix).
- For *zero-mean* $\boldsymbol{X}$ and $\boldsymbol{Y}$, $\mathsf{Cov}\{\boldsymbol{X}, \boldsymbol{Y}\} = \mathsf{E}\{\boldsymbol{X}\boldsymbol{Y}^\top\}$, and when $\mathsf{E}\{\boldsymbol{X}\boldsymbol{Y}^\top\} = \mathbf{0}$, we say $\boldsymbol{X}$ and $\mathbf{Y}$ are mutually *orthogonal*. Therefore between two zero-mean Gaussian random vectors, orthogonality means zero covariance. The concept of orthogonality is essential for the derivation of the Kalman filter.

Attention: Orthogonality

To be absolutely clear, between two vectors, orthogonality means *zero inner product*, but between two *zero-mean, random* vectors, orthogonality means *zero outer product in expectation*.

A useful property of the covariance matrix has to do with the effect of affine transformation on a random vector:

Theorem 2: [MC12, Theorem 6.2]

Let $\boldsymbol{X}$ be a random vector, and $\boldsymbol{Y} = \mathbf{A}\boldsymbol{X} + \boldsymbol{b}$. Then,

$$\begin{aligned}\mathsf{E}\{\boldsymbol{Y}\} &= \mathbf{A}\,\mathsf{E}\{\boldsymbol{X}\} + \boldsymbol{b},\\ \mathsf{Var}\{\boldsymbol{Y}\} &= \mathbf{A}\,\mathsf{Var}\{\boldsymbol{X}\}\mathbf{A}^\top.\end{aligned}$$

Another useful and actually fundamental property of covariance matrices is that

Theorem 3: [MC12, Theorem 6.1]

Covariance matrices are *symmetric* and *positive definite*.

This is an important consideration for robust numerical computation of the Kalman filter, as we shall see later. For now, let us learn what positive definiteness means for a symmetric matrix. We start by stating the structure of a matrix determines the nature of its eigenvalues. For example, a symmetric matrix must have real eigenvalues. A symmetric matrix that is positive semidefinite must additionally have nonnegative eigenvalues. Positive definiteness is a stronger quality than positive semidefiniteness, in the sense that the eigenvalues are not just nonnegative, but positive.

Definition(s): Positive definiteness (see [Mey01, p. 559], [Dei82], [Nah69])

For Hermitian matrix $\mathbf{A}$, the following statements are equivalent, and any one can serve as the definition of $\mathbf{A}$ as *positive definite*:

- $\boldsymbol{x}^*\mathbf{A}\boldsymbol{x} > 0$ for all $\boldsymbol{x} \neq \mathbf{0}$. Note: $\boldsymbol{x}^*\mathbf{A}\boldsymbol{x}$ is called a *quadratic form*.
- The eigenvalues of $\mathbf{A}$ are all positive.
- $\mathbf{A}$ can be put in the form $\mathbf{A} = \mathbf{B}^*\mathbf{B}$ or $\mathbf{A} = \mathbf{B}\mathbf{B}^*$, where $\mathbf{B}$ is a nonsingular matrix.

Similarly, the following statements are equivalent, and any one can serve as the definition of $\mathbf{A}$ as *positive semidefinite*:

- $\boldsymbol{x}^*\mathbf{A}\boldsymbol{x} \geq 0$ for all $\boldsymbol{x} \neq \mathbf{0}$.

- The eigenvalues of $\mathbf{A}$ are all nonnegative.

It is exactly because the covariance matrix is positive definite that in Eq. (2) — the equation for the multivariate Gaussian pdf — we can always find some nonsingular $\mathbf{Q}$ such that $\mathbf{\Sigma} = \mathbf{Q}^*\mathbf{Q}$, and thus

$$\det(\mathbf{\Sigma}) = \det(\mathbf{Q}^*)\det(\mathbf{Q}) = \det(\mathbf{Q})^2 > 0. \tag{6}$$

The fact that the covariance matrix is positive definite can be used as a diagnostic mechanism to ensure the computational robustness of the Kalman filter, as we shall see in Sect. 4.2.

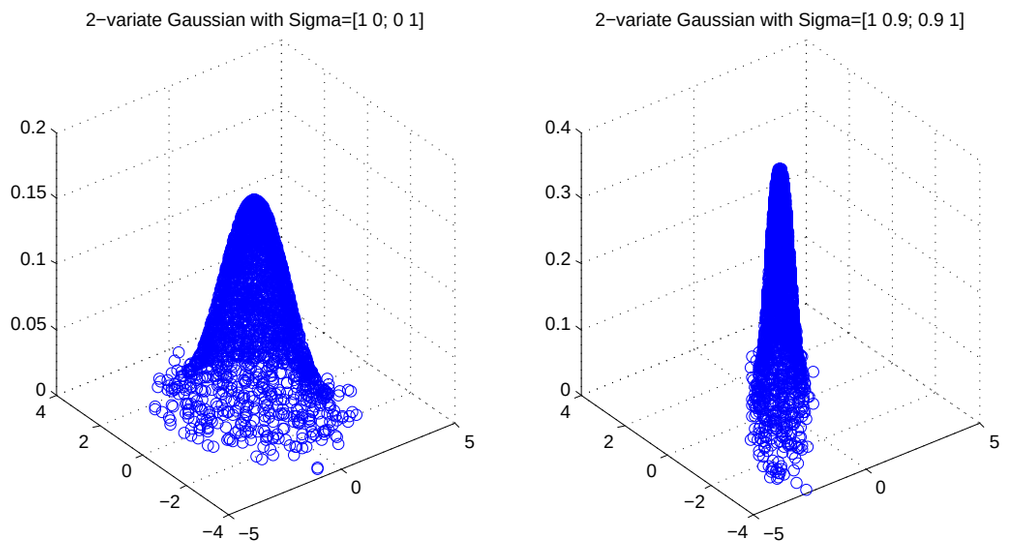


Figure 4: From Example 1: Sample 2-variate Gaussian distributions with different covariance matrices. On the left, the two variables are uncorrelated. On the right, the two variables are correlated by a covariance of 0.9. The more correlated the two variables are, the more linearly related they are.

Example 1

Figure 4, plotted using Listing 1, helps us visualize the effect of different covariance matrices on a 2-variate Gaussian distribution. The covariance matrices are

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \text{ and } \begin{bmatrix} 1 & 0.9 \\ 0.9 & 1 \end{bmatrix}.$$

Note that we cannot use

$$\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix},$$

which is positive semidefinite rather than positive definite as required for any covariance matrix.

Listing 1: Code for Example 1

```
Mu = [0 0];

figure; rotate3d on;

subplot(1,2,1)
Sigma = [1 0; 0 1];
data = mvnrnd(Mu, Sigma, 2500); p = mvnpdf(data, Mu, Sigma);
scatter3(data(:,1), data(:,2), p);
title('2-variate Gaussian with Sigma=[1 0; 0 1]');

subplot(1,2,2)
Sigma = [1 0.9; 0.9 1];
```

```
data = mvnrnd(Mu, Sigma, 2500); p = mvnpdf(data, Mu, Sigma);
scatter3(data(:,1), data(:,2), p);
title('2-variate Gaussian with Sigma=[1 0.9; 0.9 1]');
```

At this point, we conclude our revision of essential probabilistic and statistical concepts. In the next section, we will apply these concepts to an intuitive "derivation" of the Kalman filter. For detailed discussion of topics not adequately covered in this section, please refer to [MC12] or Khan Academy.

3 Intuitive derivation of the Kalman filter

This section is dedicated to an intuitive, nonrigorous "derivation" of the Kalman filter, adapted from Faragher's article [Far12]. This is by no means a replacement for a rigorous derivation, which can be found in Supplementary Lecture D, but already this intuitive "derivation" allows us to bring a few control and estimation concepts together.

Consider a train traveling in one dimension, as in Figure 5. By Newton's second law, ignoring friction, we can write down the train's equation of motion as

$$\ddot{x} = \frac{1}{m} f,$$

where x is the train's position, and f is the force produced by the train's engine. To make the equation more useful, let us introduce the state s for speed, and apply discretization using the forward rectangular version of Euler's method, to get

$$\begin{aligned} &\text{Acceleration:} && \frac{s_k - s_{k-1}}{T} = \frac{1}{m} f_{k-1} \implies s_k = s_{k-1} + \frac{T}{m} f_{k-1}, \\ &\text{Speed:} && s_{k-1} = \frac{x_k - x_{k-1}}{T} \implies x_k = x_{k-1} + T s_{k-1}. \end{aligned}$$

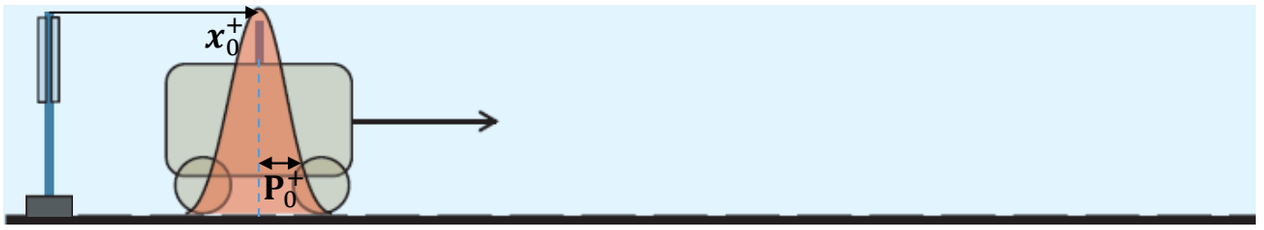
In the equations above, T is the sample period. Define the state vector as $\boldsymbol{x} = [x \; s]^\top$. Assuming s can be measured (for example by the Doppler effect), but x cannot, we define the output as the speed measurement. Then, we can form the discrete-time linear state-space equations

$$\begin{cases} \boldsymbol{x}_k = \begin{bmatrix} x_k \\ s_k \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix}}_{\mathbf{F}_{k-1}} \begin{bmatrix} x_{k-1} \\ s_{k-1} \end{bmatrix} + \underbrace{\begin{bmatrix} 0 \\ \frac{T}{m} \end{bmatrix}}_{\mathbf{G}_{k-1}} f_{k-1} + \boldsymbol{w}_{k-1}, & \text{(7a)} \\ \boldsymbol{y}_k = \mathbf{H}_k \boldsymbol{x}_k + \boldsymbol{v}_k. & \text{(7b)} \end{cases}$$

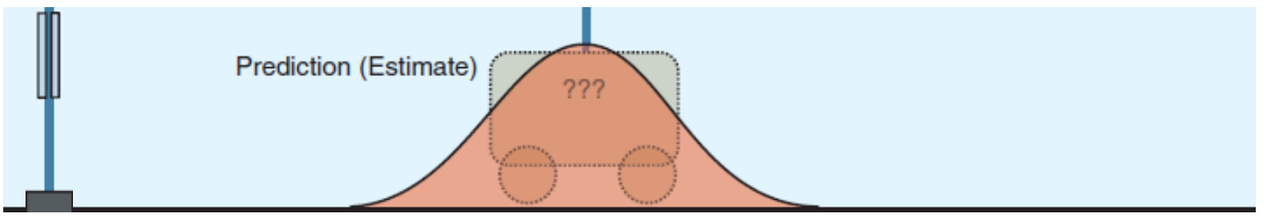
Above, $\mathbf{H}_k$ is meant to capture the physics relating the speed to speed measurement, but its exact content does not concern us here. The noise terms $\boldsymbol{w}_{k-1}$ and $\boldsymbol{v}_k$ capture the uncertain and noisy nature of the system: while $\boldsymbol{v}_k$ captures measurement noise, $\boldsymbol{w}_{k-1}$ captures uncertainties (e.g., unaccounted friction) in the model, and disturbances (e.g., track irregularities, wind) acting on the system.

The problem of state estimation is, at the kth sample period, to determine $\boldsymbol{x}_k$ based only on the system model in the form of Eq. (7), and knowledge of the input f_{k-1} and output measurement $\boldsymbol{y}_k$, in the presence of noise $\boldsymbol{w}_{k-1}$ and $\boldsymbol{v}_k$. In what follows, we outline a solution for this state estimation problem, based on the assumptions:

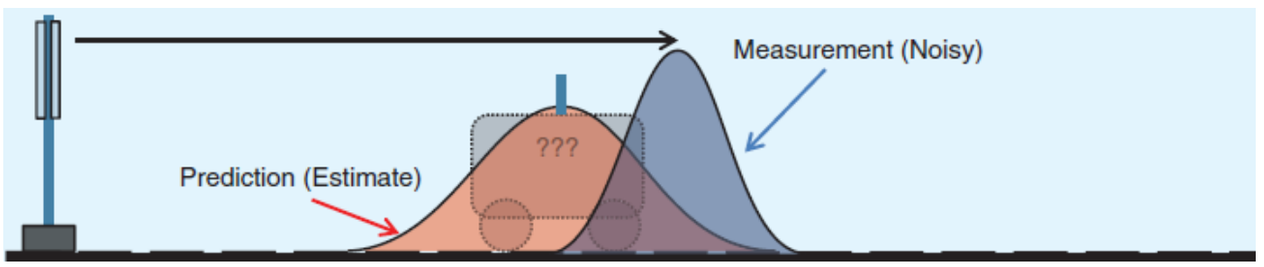
1. $\boldsymbol{w}$ and $\boldsymbol{v}$ are zero-mean Gaussian distributed with covariance matrices $\mathbf{Q}$ and $\mathbf{R}$ respectively, and are mutually independent.



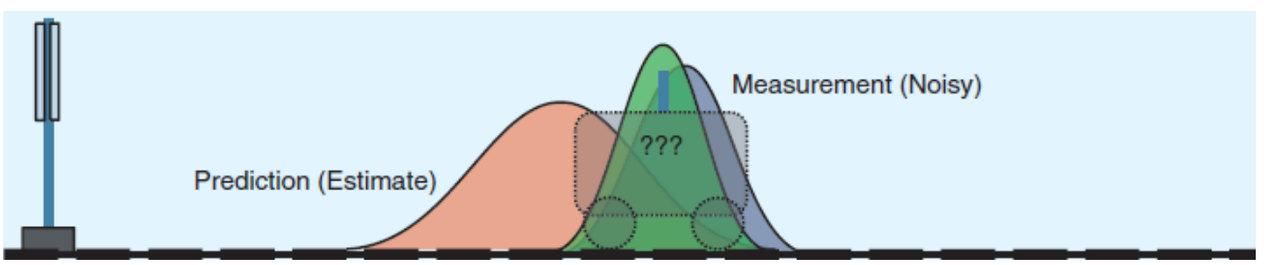
(a) $k = 0$



(b) $k = 1$, before acquiring measurement



(c) $k = 1$, after acquiring measurement



(d) $k = 1$, fusing probability density functions

Figure 5: A train of mass m traveling in one dimension described by state $\boldsymbol{x} = [x\ s]^\top$, where x represents position and s represents speed. Images adapted from [Far12].

2. $\boldsymbol{x}$ is Gaussian distributed.

Solution outline:

(a) At time $k = 0$ (see Figure 5(a)), suppose we have an *initial state estimate*, denoted $\hat{\boldsymbol{x}}_0^+$. The accent $\hat{\square}$ denotes "estimate" by convention. The superscript $+$ denotes "a posteriori" or "posterior", the meaning of which will become clearer through ensuing discussion. A sensible value for $\hat{\boldsymbol{x}}_0^+$ is the expected value of $\boldsymbol{x}_0$, i.e.,

$$\hat{\boldsymbol{x}}_0^+ = \mathsf{E}\{\boldsymbol{x}_0\}. \tag{8}$$

The uncertainty in $\hat{\boldsymbol{x}}_0^+$ can be captured by the *initial error covariance*:

$$\mathbf{P}_0^+ = \mathsf{E}\left\{(\boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0^+)(\boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0^+)^\top\right\}. \tag{9}$$

Detail: Prior and posterior

In the literature, it is common to use "posterior" and "prior" interchangeably with their Latin equivalents, which are "a posteriori" and "a priori" respectively. Alternating between prior and posterior estimates is a standard Bayesian filtering technique, as can be seen in Supplementary Lecture D.

(b) At time $k = 1$ (see Figure 5(b)), *before* we acquire measurement $\boldsymbol{y}_1$, we can make an "a priori" or "prior" state estimate, by essentially making a prediction based on the initial state estimate $\hat{\boldsymbol{x}}_0^+$ and the model in the form of the state equation (7):

$$\hat{\boldsymbol{x}}_1^- = \mathbf{F}_0 \hat{\boldsymbol{x}}_0^+ + \mathbf{G}_0 f_0. \tag{10}$$

Without measurement $\boldsymbol{y}_1$, it should be understandable that the uncertainty $\mathbf{P}_0^+$ would have grown over the duration of a sample period. In fact, we can estimate the current uncertainty $\mathbf{P}_1^-$ by applying the covariance operator to the difference between Eq. (7a) and Eq. (10):

$$\boldsymbol{x}_1 - \hat{\boldsymbol{x}}_1^- = \mathbf{F}_0(\boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0^+) + \boldsymbol{w}_0 \implies \boldsymbol{e}_1^- = \mathbf{F}_0 \boldsymbol{e}_0^+ + \boldsymbol{w}_0,$$

where $\boldsymbol{e}_k^- \overset{\text{def}}{=} \boldsymbol{x}_k - \hat{\boldsymbol{x}}_k^-$ and $\boldsymbol{e}_k^+ \overset{\text{def}}{=} \boldsymbol{x}_k - \hat{\boldsymbol{x}}_k^+$.

$$\begin{aligned}
& \mathsf{E}\left\{\boldsymbol{e}_1^-(\boldsymbol{e}_1^-)^\top\right\} = \mathsf{E}\left\{(\mathbf{F}_0\boldsymbol{e}_0^+ + \boldsymbol{w}_0)(\mathbf{F}_0\boldsymbol{e}_0^+ + \boldsymbol{w}_0)^\top\right\} \\
& \qquad\qquad \text{noticing } \boldsymbol{e}_0^+ \text{ and } \boldsymbol{w}_0 \text{ are independent} \\
\implies & \mathbf{P}_1^- = \mathsf{E}\left\{\mathbf{F}_0\boldsymbol{e}_0^+(\boldsymbol{e}_0^+)^\top\mathbf{F}_0^\top\right\} + \mathsf{E}\{\boldsymbol{w}_0\boldsymbol{w}_0^\top\} \\
\implies & \mathbf{P}_1^- = \mathbf{F}_0\mathbf{P}_0^+\mathbf{F}_0^\top + \mathbf{Q}_0,
\end{aligned} \tag{11}$$

where $\mathbf{P}_k^- \overset{\text{def}}{=} \mathsf{E}\left\{\boldsymbol{e}_k^-(\boldsymbol{e}_k^-)^\top\right\}$ and $\mathbf{P}_k^+ \overset{\text{def}}{=} \mathsf{E}\left\{\boldsymbol{e}_k^+(\boldsymbol{e}_k^+)^\top\right\}$.

(c) Still at time $k = 1$ (see Figure 5(c)), *after* we acquire measurement $\boldsymbol{y}_1$, we can improve our prior state estimate $\boldsymbol{x}_1^-$ using the additional information provided by $\boldsymbol{y}_1$. However, $\boldsymbol{y}_1$ is noisy — although the measurement noise $\boldsymbol{v}_1$ has a mean of zero, it has a "spread" represented by $\mathbf{R}_1$.

(d) Still at time $k = 1$ (see Figure 5(d)), our strategy is to somehow "fuse" the uncertainties represented by $\mathbf{P}_1^-$ and $\mathbf{R}_1$. While Faragher [Far12] proposed multiplying the pdfs associated with $\mathbf{H}_1\hat{\boldsymbol{x}}_1^-$ and $\boldsymbol{y}_1$, the resultant function is Gaussian but not a pdf, because the unnormalized function does not integrate to 1 [Ahr05]. Instead of fusing the pdfs, the approach here is to express the posterior state estimate as

$$\hat{\boldsymbol{x}}_1^+ = \hat{\boldsymbol{x}}_1^- + \mathbf{K}_1(\boldsymbol{y}_1 - \mathbf{H}_1\hat{\boldsymbol{x}}_1^-), \tag{12}$$

where the second term on the right is a feedback term that is proportional to the difference between the measured output and predicted output, not unlike the feedback term of the Luenberger observer. Adding $\boldsymbol{x}_1$ to the negative of the preceding equation gives us

$$\begin{aligned}
& \boldsymbol{x}_1 - \hat{\boldsymbol{x}}_1^+ = \boldsymbol{x}_1 - \hat{\boldsymbol{x}}_1^- - \mathbf{K}_1[\mathbf{H}_1(\boldsymbol{x}_1 - \hat{\boldsymbol{x}}_1^-) + \boldsymbol{v}_1] \\
\implies & \boldsymbol{e}_1^+ = (\mathbf{I} - \mathbf{K}_1\mathbf{H}_1)\boldsymbol{e}_1^- - \mathbf{K}_1\boldsymbol{v}_1 \qquad (13) \\
\implies & \mathsf{E}\left\{\boldsymbol{e}_1^+(\boldsymbol{e}_1^+)^\top\right\} = \mathsf{E}\left\{\left[(\mathbf{I} - \mathbf{K}_1\mathbf{H}_1)\boldsymbol{e}_1^- - \mathbf{K}_1\boldsymbol{v}_1\right]\left[(\mathbf{I} - \mathbf{K}_1\mathbf{H}_1)\boldsymbol{e}_1^- - \mathbf{K}_1\boldsymbol{v}_1\right]^\top\right\} \\
& \qquad\qquad \text{noticing } \boldsymbol{e}_1^- \text{ and } \boldsymbol{v}_1 \text{ are independent} \\
\implies & \mathbf{P}_1^+ = (\mathbf{I} - \mathbf{K}_1\mathbf{H}_1)\mathbf{P}_1^-(\mathbf{I} - \mathbf{K}_1\mathbf{H}_1)^\top + \mathbf{K}_1\mathbf{R}_1\mathbf{K}_1^\top. \qquad (14)
\end{aligned}$$

The gain matrix $\mathbf{K}$ has yet to be determined, and for this, we need one last ingredient:

$$\mathsf{E}\left\{\boldsymbol{e}_k^+\left(\mathbf{H}_k\boldsymbol{e}_k^- + \boldsymbol{v}_k\right)^\top\right\} = \mathbf{0}, \tag{15}$$

which is a result of the *orthogonality principle*, as detailed in Supplementary Lecture D. Substituting Eq. (13) into Eq. (15) for $k = 1$, and after some algebra, we can get

$$\mathbf{K}_1 = \mathbf{P}_1^-\mathbf{H}_1^\top(\mathbf{H}_1\mathbf{P}_1^-\mathbf{H}_1^\top + \mathbf{R}_1)^{-1}. \tag{16}$$

At this point, we have all the ingredients we need to assemble the Kalman filter (generalizing subscript 1 to k):

- Eqs. (8) and (9) for initialization;
- Eqs. (10) and (11) for prior state estimation — this is called the *time update* or *prediction update* step;
- Eq. (16) for updating the *Kalman gain*;
- Eqs. (12) and (14) for posterior state estimation — this is called the *measurement update* step.

The prediction update and measurement update steps reflect the recursive nature of the algorithm. Except for the "magic" Eq. (15), we were able to derive all the equations using the statistical tools from the previous section.

4 The Kalman filter algorithm

The Kalman filter is based on the *linear Gaussian model* (see [Sim06, Sect. 5.1], [Hay09, Sect. 14.2], [Sär13, p. 37]):

- The discrete-time system is linear and stochastic:

$$\boldsymbol{x}_k = \mathbf{F}_{k-1}\boldsymbol{x}_{k-1} + \mathbf{G}_{k-1}\boldsymbol{u}_{k-1} + \boldsymbol{w}_{k-1}, \tag{17}$$
$$\boldsymbol{y}_k = \mathbf{H}_k\boldsymbol{x}_k + \boldsymbol{v}_k, \tag{18}$$

  where $\mathbf{F}$, $\mathbf{G}$ and $\mathbf{H}$ can be time-varying; $\boldsymbol{w}$ is the *process noise*; and $\boldsymbol{v}$ is the *measurement noise* or *observation noise*. This notation is slightly different from the notation we have been using for discrete-time systems so far, but this is more consistent with the notation used in the Kalman filter literature.

- The process noise and measurement noise are
  - additive, i.e., they corrupt the original signal by being added to (rather than multiplied with) the signal;
  - zero-mean Gaussian distributed;
  - uncorrelated forward and backward in time.

  In other words, they are *additive white Gaussian noise*. Note that we are dealing with discrete white noise here which has constant power spectral density over a finite frequency range, and a finite variance [GRG01, p. 384]; whereas continuous white noise has a constant power spectral density over an infinite frequency range, and an infinite variance [Pri81, p. 235]. Furthermore, we assume the process noise is uncorrelated with the measurement noise.

  In summary, for the process noise and measurement noise, we assume for all j, k,

$$\begin{aligned} &\boldsymbol{w}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_k), && \boldsymbol{v}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_k), \\ &\mathsf{Cov}\{\boldsymbol{w}_k, \boldsymbol{w}_j\} = \mathbf{Q}_k\delta_{k-j}, && \mathsf{Cov}\{\boldsymbol{v}_k, \boldsymbol{v}_j\} = \mathbf{R}_k\delta_{k-j}, \\ &\mathsf{Cov}\{\boldsymbol{v}_k, \boldsymbol{w}_j\} = \mathbf{0}, \end{aligned}$$

  where $\mathbf{Q}_k = \mathsf{E}\{\boldsymbol{w}_k\boldsymbol{w}_k^\top\}$ and $\mathbf{R}_k = \mathsf{E}\{\boldsymbol{v}_k\boldsymbol{v}_k^\top\}$ are covariance matrices; and δ_{k-j} is the Kronecker delta, i.e.,

$$\delta_{k-j} = \begin{cases} 1 & \text{if } k = j, \\ 0 & \text{if } k \neq j. \end{cases}$$

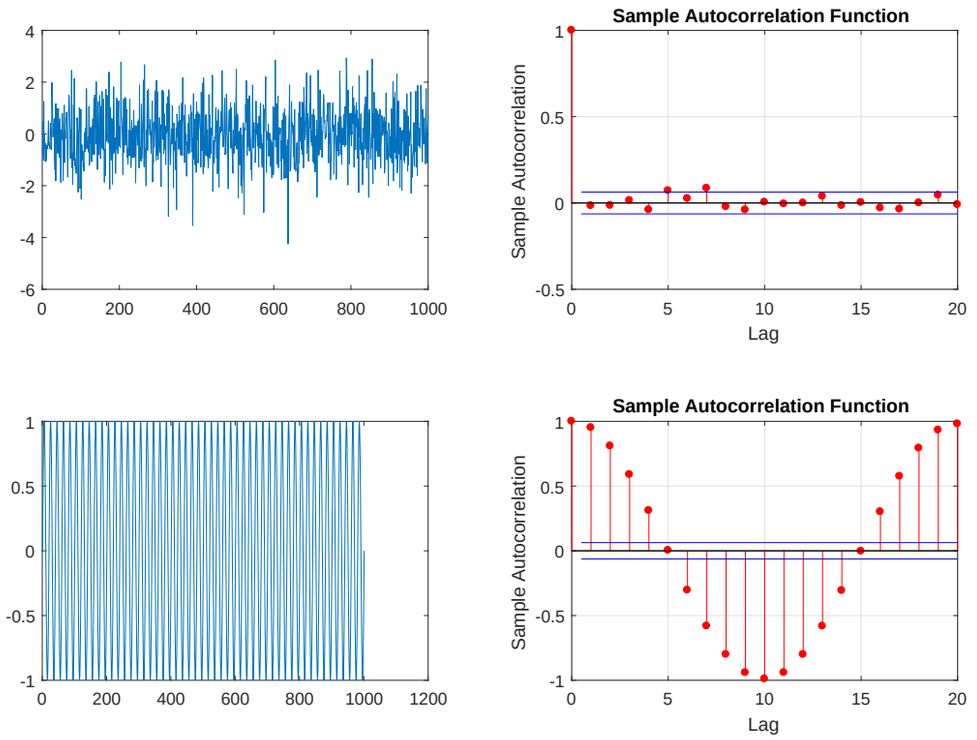


Figure 6: From Example 2: two signals and their sample autocorrelation plots.

Example 2

The "whiteness" of a noise signal can be checked using the *sample autocorrelation* ("auto" means "self") function. The function measures the correlation between samples separated by l sample periods, where l is called the number of lags. Suppose we generate two signals using Listing: v_1 being a Gaussian white noise, and v_2 being a sinusoid.

Listing 2: Code for Example 2

```
n = 1000;
v1 = randn(1, n);
v2 = sin(0:pi/10:n*pi/10);

figure;
subplot(2,2,1); plot(1:numel(v1), v1); subplot(2,2,2); autocorr(v1)
subplot(2,2,3); plot(1:numel(v2), v2); subplot(2,2,4); autocorr(v2)
```

Then as Figure 6 shows, the sample autocorrelation function of v_1 returns small values for nonzero lags, while the sample autocorrelation function of v_2 returns about 1 for lags that are some multiple of 20, because $\pi/10 \times 20 = 2\pi$.

The standard recursive form of the Kalman filter (see [Sim06, pp. 128-129], [CJ12, Table 3.1], [Hay01, Table 1.1], [Jek00, Figure 7.1]) is given below:

Kalman filter

Initialization ($k = 0$):

$$\hat{\boldsymbol{x}}_0^+ = \mathsf{E}\{\boldsymbol{x}_0\}, \quad \mathbf{P}_0^+ = \mathsf{E}\left\{(\boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0^+)(\boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0^+)^\top\right\}. \tag{19}$$

Time update / prediction update equations for propagating *a priori* estimates ($k = 1, 2, \ldots$):

$$\hat{\boldsymbol{x}}_k^- = \mathbf{F}_{k-1}\hat{\boldsymbol{x}}_{k-1}^+ + \mathbf{G}_{k-1}\boldsymbol{u}_{k-1}, \tag{20}$$

$$\mathbf{P}_k^- = \mathbf{F}_{k-1}\mathbf{P}_{k-1}^+\mathbf{F}_{k-1}^\top + \mathbf{Q}_{k-1}. \tag{21}$$

Kalman gain update:

$$\mathbf{K}_k = \mathbf{P}_k^-\mathbf{H}_k^\top(\mathbf{H}_k\mathbf{P}_k^-\mathbf{H}_k^\top + \mathbf{R}_k)^{-1}. \tag{22}$$

Measurement update equations for updating *a posteriori* estimates:

$$\hat{\boldsymbol{x}}_k^+ = \hat{\boldsymbol{x}}_k^- + \mathbf{K}_k(\boldsymbol{y}_k - \mathbf{H}_k\hat{\boldsymbol{x}}_k^-) = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\hat{\boldsymbol{x}}_k^- + \mathbf{K}_k\boldsymbol{y}_k, \tag{23}$$

Joseph stabilized version: $\mathbf{P}_k^+ = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}_k^-(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)^\top + \mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^\top.$ (24)

Standard form: $\mathbf{P}_k^+ = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}_k^-.$ (25)

Information filter form: $\mathbf{P}_k^+ = [(\mathbf{P}_k^-)^{-1} + \mathbf{H}_k^\top\mathbf{R}_k^{-1}\mathbf{H}_k]^{-1}.$ (26)

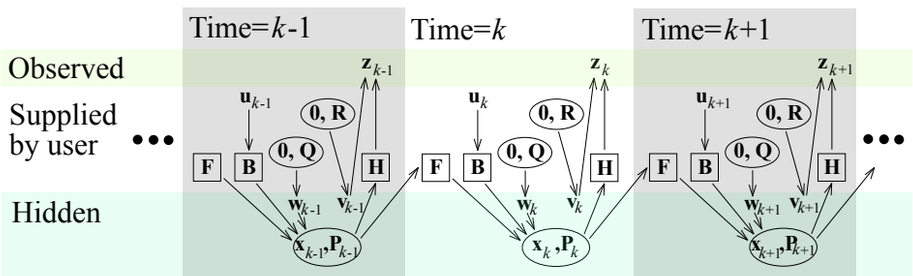


Figure 7: The Kalman filter illustrated. Squares represent matrices. Ellipses represent multivariate Gaussian distributions (with the mean and covariance matrix enclosed). Unenclosed values are vectors. Source: https://en.wikipedia.org/wiki/Kalman_filter.

Several key points need to be highlighted about the algorithm given above:

- The Kalman filter is applicable to both time-invariant and time-varying problems, and gives good results in practice due to its optimal formulation. It is also incremental, in the sense that it does not need the entire data history.
- $\hat{\boldsymbol{x}}_k^+ \overset{\text{def}}{=} \hat{\boldsymbol{x}}_{k|k}$ is the posterior estimate. $\hat{\boldsymbol{x}}_k^- \overset{\text{def}}{=} \hat{\boldsymbol{x}}_{k|k-1}$ is the prior estimate. The new but standard-compliant notation is meant to alleviate notational clutter.

> **Quiz 5**
>
> In theory, is $\hat{\boldsymbol{x}}_k^-$ or $\hat{\boldsymbol{x}}_k^+$ more accurate?

- In essence, the Kalman filter simplifies the Bayesian filtering equations (see Supplementary Lecture D) by replacing the pdfs with conditional means, i.e.,

$$\hat{\boldsymbol{x}}_k^+ \overset{\text{def}}{=} \mathsf{E}\{\boldsymbol{x}_k|\boldsymbol{y}_{1:k}\}, \qquad \hat{\boldsymbol{x}}_0^+ \overset{\text{def}}{=} \mathsf{E}\{\boldsymbol{x}_0\} \text{ (user-specified);}$$
$$\hat{\boldsymbol{x}}_k^- \overset{\text{def}}{=} \mathsf{E}\{\boldsymbol{x}_k|\boldsymbol{y}_{1:(k-1)}\}, \qquad \hat{\boldsymbol{x}}_0^- \text{ is undefined.}$$

  The resultant conditional mean estimate is the MMSE estimate. The difference $(\boldsymbol{y}_k - \mathbf{H}_k\hat{\boldsymbol{x}}^-)$ is called the *innovation*, and can be understood as a measure of compensation required to improve/innovate the estimate. The measurement update equation (23) therefore makes intuitive sense by this reasoning: the estimate $\hat{\boldsymbol{x}}_k^+$ is compensated more if the innovation is more, and less if otherwise. At the same time updating the posterior and prior estimates of the state, the Kalman filter updates the posterior and prior covariances of the estimation error, defined as:

$$\mathbf{P}_k^+ \overset{\text{def}}{=} \mathsf{E}\{(\boldsymbol{x}_k - \hat{\boldsymbol{x}}_k^+)(\boldsymbol{x}_k - \hat{\boldsymbol{x}}_k^+)^\top\}, \qquad \mathbf{P}_0^+ \overset{\text{def}}{=} \mathsf{E}\{(\boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0^+)(\boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0^+)^\top\} \text{ (user-specified);}$$
$$\mathbf{P}_k^- \overset{\text{def}}{=} \mathsf{E}\{(\boldsymbol{x}_k - \hat{\boldsymbol{x}}_k^-)(\boldsymbol{x}_k - \hat{\boldsymbol{x}}_k^-)^\top\}, \qquad \mathbf{P}_0^- \text{ is undefined.}$$

- The *general* Kalman filter, as implemented in MATLAB, can deal with mutually correlated process and measurement noise, although the algorithm given above cannot. For the general algorithm, please refer to [Sim06, Sect. 7.1].

Example 3

This example is adapted from [GA15, Example 5.4]. Consider the first-order discrete-time system

$$x[k+1] = x[k] + w[k],$$
$$y[k] = x[k] + v[k].$$

It is known that $w \sim \mathcal{N}(0,1)$, $v \sim \mathcal{N}(0,2)$, $y[1] = 2$, $y[2] = 3$, $\hat{x}_0^+ = 1$, $P_0^+ = 10$. Find $\hat{x}_2^+$ and the steady-state covariance matrix P_∞^+ using Kalman filtering. Hint: In steady state, set $P_k^+ = P_{k-1}^+$.

Solution: Based on the given system model, we have

$$F = 1, \quad G = 0, \quad H = 1, \quad Q = 1, \quad R = 2.$$

From Eqs. (20) and (21), we get the time update equations:

$$\hat{x}_k^- = \hat{x}_{k-1}^+. \tag{27}$$
$$P_k^- = P_{k-1}^+ + 1. \tag{28}$$

From Eq. (22), we get the Kalman gain update equation:

$$K_k = P_k^-(P_k^- + 2)^{-1}. \tag{29}$$

From Eqs. (25), (29) and (28), we get the measurement update equations:

$$\hat{x}_k^+ = \hat{x}_k^- + K_k(y_k - \hat{x}_k^-) = (1 - K_k)\hat{x}_k^- + K_k y_k = \left(1 - \frac{P_k^-}{P_k^- + 2}\right)\hat{x}_k^- + \frac{P_k^- y_k}{P_k^- + 2}$$
$$= \frac{2\hat{x}_k^- + P_k^- y_k}{P_k^- + 2}. \tag{30}$$

$$P_k^+ = \frac{2P_k^-}{P_k^- + 2} = 2\frac{P_{k-1}^+ + 1}{P_{k-1}^+ + 3}. \tag{31}$$

Therefore,

$$\hat{x}_1^+ = \frac{2\hat{x}_1^- + P_1^- y_1}{P_1^- + 2} = \frac{2\hat{x}_0^+ + (P_0^+ + 1)y_1}{P_0^+ + 3} = \frac{2 + 11 \times 2}{13} = \frac{24}{13}.$$
$$P_1^+ = 2\frac{P_0^+ + 1}{P_0^+ + 3} = \frac{2 \times 11}{13} = \frac{22}{13}.$$
$$\hat{x}_2^+ = \frac{2\hat{x}_2^- + P_2^- y_2}{P_2^- + 2} = \frac{2\hat{x}_1^+ + (P_1^+ + 1)y_2}{P_1^+ + 3} = \frac{2(24/13) + (22/13 + 1)3}{22/13 + 3} = \frac{153}{61}.$$

$$P_\infty^+ = \frac{2(P_\infty^+ + 1)}{P_\infty^+ + 3} \implies (P_\infty^+)^2 + 3P_\infty^+ = 2P_\infty^+ + 2$$
$$\implies (P_\infty^+)^2 + P_\infty^+ - 2 = 0$$
$$\implies P_\infty^+ = 1 \qquad \text{(ruling out } P_\infty^+ = -2\text{)}.$$

Example 4

This example is adapted from [Sim06, Example 5.1]. Consider a system with constant acceleration:

$$\begin{aligned}\dot{r} &= v,\\ \dot{v} &= a,\\ \dot{a} &= 0,\end{aligned}$$

where r represents a 1D position, v represents velocity, and a represents acceleration. Assume no input, and output is r. Express $\mathbf{P}_k^-$ as

$$\mathbf{P}_k^- = \begin{bmatrix} P^-_{k(1,1)} & P^-_{k(1,2)} & P^-_{k(1,3)} \\ P^-_{k(2,1)} & P^-_{k(2,2)} & P^-_{k(2,3)} \\ P^-_{k(3,1)} & P^-_{k(3,2)} & P^-_{k(3,3)} \end{bmatrix}. \tag{32}$$

Using the standard-form update equation: $\mathbf{P}_k^+ = \mathbf{P}_k^- - \mathbf{K}_k\mathbf{H}_k\mathbf{P}_k^-$, show that

$$\mathrm{Tr}(\mathbf{P}_k^+) = \mathrm{Tr}(\mathbf{P}_k^-) - \frac{(P^-_{k(1,1)})^2 + (P^-_{k(1,2)})^2 + (P^-_{k(1,3)})^2}{P^-_{k(1,1)} + \sigma^2},$$

where Tr is the trace operator, which returns the sum of the diagonal elements of a matrix, and σ^2 is the variance of the observation noise. In other words, the posterior error covariance is always smaller than the prior error covariance for the same time interval in terms of their traces.

Solution: The system can be represented by the continuous-time state-space equations:

$$\begin{bmatrix}\dot{r}\\ \dot{v}\\ \dot{a}\end{bmatrix} = \begin{bmatrix}0 & 1 & 0\\ 0 & 0 & 1\\ 0 & 0 & 0\end{bmatrix}\begin{bmatrix}r\\ v\\ a\end{bmatrix}, \quad r = \begin{bmatrix}1 & 0 & 0\end{bmatrix}\begin{bmatrix}r\\ v\\ a\end{bmatrix}. \tag{33}$$

Discretizing the above with sampling interval T, we have

$$\boldsymbol{x}_{k+1} = \mathbf{F}\boldsymbol{x}_k, \quad y_k = \mathbf{H}\boldsymbol{x}_k, \tag{34}$$

where $\boldsymbol{x} = [r \quad v \quad a]^\top$, $\mathbf{F} = \exp(\mathbf{A}T)$, $y = r$, $\mathbf{H} = [1 \quad 0 \quad 0]$. Since $\mathbf{A}$ is a nilpotent matrix s.t. $\mathbf{A}^i = \mathbf{0}$ for $i \geq 3$, we can calculate the matrix exponential of $\mathbf{A}$, and hence $\mathbf{F}$, as

$$\mathbf{F} = \exp(\mathbf{A}T) = \mathbf{I} + \mathbf{A}T + \frac{1}{2!}\mathbf{A}^2T^2 = \begin{bmatrix}1 & T & \frac{T^2}{2}\\ 0 & 1 & T\\ 0 & 0 & 1\end{bmatrix}. \tag{35}$$

Assuming there is no process noise, and the measurement noise has variance $R_k = \sigma^2$, we can derive a Kalman filter for the system as follows.

From Eqs. (20) and (21), we get the time update equations:

$$\hat{\boldsymbol{x}}_k^- = \mathbf{F}\hat{\boldsymbol{x}}_{k-1}^+. \tag{36}$$

$$\mathbf{P}_k^- = \mathbf{F}\mathbf{P}_{k-1}^+\mathbf{F}^\top. \tag{37}$$

From Eq. (22), we get the Kalman gain update equation:

$$\begin{aligned}\mathbf{K}_k &= \mathbf{P}_k^- \mathbf{H}_k^\top (\mathbf{H}_k \mathbf{P}_k^- \mathbf{H}_k^\top + \mathbf{R}_k)^{-1} = \mathbf{P}_k^- \begin{bmatrix}1\\0\\0\end{bmatrix} \left([1 \quad 0 \quad 0] \mathbf{P}_k^- \begin{bmatrix}1\\0\\0\end{bmatrix} + \sigma^2 \right)^{-1} \\ &= \frac{1}{P_{k(1,1)}^- + \sigma^2} \begin{bmatrix} P_{k(1,1)}^- \\ P_{k(2,1)}^- \\ P_{k(3,1)}^- \end{bmatrix} = \frac{1}{P_{k(1,1)}^- + \sigma^2} \begin{bmatrix} P_{k(1,1)}^- \\ P_{k(1,2)}^- \\ P_{k(1,3)}^- \end{bmatrix}.\end{aligned} \tag{38}$$

Note that above, we have applied the fact that $\mathbf{P}_k^-$ is symmetric:

$$\mathbf{P}_k^- = \begin{bmatrix} P_{k(1,1)}^- & P_{k(1,2)}^- & P_{k(1,3)}^- \\ P_{k(2,1)}^- & P_{k(2,2)}^- & P_{k(2,3)}^- \\ P_{k(3,1)}^- & P_{k(3,2)}^- & P_{k(3,3)}^- \end{bmatrix} = \begin{bmatrix} P_{k(1,1)}^- & P_{k(1,2)}^- & P_{k(1,3)}^- \\ P_{k(1,2)}^- & P_{k(2,2)}^- & P_{k(2,3)}^- \\ P_{k(1,3)}^- & P_{k(2,3)}^- & P_{k(3,3)}^- \end{bmatrix}.$$

From Eqs. (25) and (38), we get the measurement update equations:

$$\begin{aligned}\hat{\boldsymbol{x}}_k^+ &= \hat{\boldsymbol{x}}_k^- + \mathbf{K}_k(\boldsymbol{y}_k - \mathbf{H}\hat{\boldsymbol{x}}_k^-) = (\mathbf{I} - \mathbf{K}_k\mathbf{H})\hat{\boldsymbol{x}}_k^- + \mathbf{K}_k\boldsymbol{y}_k \\ &= \begin{bmatrix} 1 - \frac{P_{k(1,1)}^-}{P_{k(1,1)}^- + \sigma^2} & 0 & 0 \\ -\frac{P_{k(1,2)}^-}{P_{k(1,1)}^- + \sigma^2} & 1 & 0 \\ -\frac{P_{k(1,3)}^-}{P_{k(1,1)}^- + \sigma^2} & 0 & 1 \end{bmatrix} \hat{\boldsymbol{x}}_k^- + \frac{1}{P_{k(1,1)}^- + \sigma^2} \begin{bmatrix} P_{k(1,1)}^- \\ P_{k(1,2)}^- \\ P_{k(1,3)}^- \end{bmatrix} \boldsymbol{y}_k;\end{aligned} \tag{39}$$

$$\begin{aligned}\mathbf{P}_k^+ &= \mathbf{P}_k^- - \mathbf{K}_k\mathbf{H}_k\mathbf{P}_k^- = \mathbf{P}_k^- - \frac{1}{P_{k(1,1)}^- + \sigma^2} \begin{bmatrix} P_{k(1,1)}^- & 0 & 0 \\ P_{k(1,2)}^- & 0 & 0 \\ P_{k(1,3)}^- & 0 & 0 \end{bmatrix} \mathbf{P}_k^- \\ &= \mathbf{P}_k^- - \frac{1}{P_{k(1,1)}^- + \sigma^2} \begin{bmatrix} (P_{k(1,1)}^-)^2 & P_{k(1,1)}^- P_{k(1,2)}^- & P_{k(1,1)}^- P_{k(1,3)}^- \\ P_{k(1,2)}^- P_{k(1,1)}^- & (P_{k(1,2)}^-)^2 & P_{k(1,2)}^- P_{k(1,3)}^- \\ P_{k(1,3)}^- P_{k(1,1)}^- & P_{k(1,3)}^- P_{k(1,2)}^- & (P_{k(1,3)}^-)^2 \end{bmatrix}.\end{aligned} \tag{40}$$

Thus far, we have derived a Kalman filter for tracking the 1D position of an object moving with constant acceleration. Applying the trace operator to both sides of Eq. (40) gives us

$$\mathrm{Tr}(\mathbf{P}_k^+) = \mathrm{Tr}(\mathbf{P}_k^-) - \frac{(P_{k(1,1)}^-)^2 + (P_{k(1,2)}^-)^2 + (P_{k(1,3)}^-)^2}{P_{k(1,1)}^- + \sigma^2}$$

as required. Eqs. (37), (38) and (40) from Example 4 reflect a general and useful fact: the Kalman gain and error covariances depend only on the system parameters and not the observations. This means the Kalman gain and error covariances can be calculated offline and stored in memory for efficient computations. For a resource-constrained embedded system, this can make the difference between whether or not the system can perform Kalman filtering in real time.

4.1 Steady-state Kalman filter

Most Kalman filter implementations exist in embedded systems for which memory is a premium. If the observed process is time-invariant, and the process and measurement noise are stationary, then a *steady-state Kalman filter* can be used [Sim06, Sect. 7.3]. A steady-state Kalman

filter has a constant Kalman gain. Using a constant gain instead of a dynamic gain calculated with Eq. (22) is not optimal, but the constant gain approaches optimality as $k \to \infty$. In fact, for many applications, the performance of the steady-state filter matches that of the time-varying filter. Nevertheless, the accuracy and performance of a steady-state filter should be assessed empirically before it is adopted for any application.

To determine an expression for the steady-state Kalman gain, observe in the Kalman update equation (Eq. (22)), the Kalman gain depends on the prior error covariance. In turn, the prior error covariance depends on the posterior error covariance in the prediction update equation (Eq. (21)):

$$\begin{aligned}\mathbf{P}_k^- &= \mathbf{F}\mathbf{P}_{k-1}^+\mathbf{F}^\top + \mathbf{Q} = \mathbf{F}\left[(\mathbf{I}-\mathbf{K}_k\mathbf{H})\mathbf{P}_{k-1}^-\right]\mathbf{F}^\top + \mathbf{Q} = \mathbf{F}\mathbf{P}_{k-1}^-\mathbf{F}^\top - \mathbf{F}\mathbf{K}_k\mathbf{H}\mathbf{P}_{k-1}^-\mathbf{F}^\top + \mathbf{Q} \\ &= \mathbf{F}\mathbf{P}_{k-1}^-\mathbf{F}^\top - \mathbf{F}\mathbf{P}_{k-1}^-\mathbf{H}^\top(\mathbf{H}\mathbf{P}_{k-1}^-\mathbf{H}^\top + \mathbf{R})^{-1}\mathbf{H}\mathbf{P}_{k-1}^-\mathbf{F}^\top + \mathbf{Q}.\end{aligned}$$

Suppose $\mathbf{P}_k^-$ and $\mathbf{K}_k$ converge to $\mathbf{P}_\infty$ and $\mathbf{K}_\infty$ respectively, then

$$\mathbf{F}\mathbf{P}_\infty\mathbf{F}^\top - \mathbf{P}_\infty + \mathbf{Q} - \mathbf{F}\mathbf{P}_\infty\mathbf{H}^\top(\mathbf{H}\mathbf{P}_\infty\mathbf{H}^\top + \mathbf{R})^{-1}\mathbf{H}\mathbf{P}_\infty\mathbf{F}^\top = \mathbf{0}. \tag{41}$$

Eq. (41) is called a *discrete-time algebraic Riccati equation* (DARE), and it can be solved using the MATLAB function `idare`, which implements an algorithm similar to the Python function `scipy.linalg.solve_discrete_are`[1]. In terms of solving DAREs by hand, we limit ourselves to solving first-order DAREs here. In Lecture 8, we will learn how to solve higher-order DAREs by hand.

Detail: Riccati equation

A Riccati equation is a first-order ordinary differential equation of the form

$$\dot{x}(t) = a_2(t)x^2(t) + a_1(t)x(t) + a_0(t),$$

where $a_2(t)$ and $a_0(t)$ are nonzero. A DARE is a matrix-valued Riccati equation with the derivative replaced by a difference. We will encounter algebraic Riccati equations again when we discuss optimal control later in the course.

Upon solving the DARE that is Eq. (41) for $\mathbf{P}_\infty$, the Kalman gain can be calculated through Eq. (22) as

$$\mathbf{K}_\infty = \mathbf{P}_\infty\mathbf{H}^\top(\mathbf{H}\mathbf{P}_\infty\mathbf{H}^\top + \mathbf{R})^{-1}. \tag{42}$$

$\mathbf{K}_\infty$ is called a *steady-state Kalman gain*. Substituting Eqs. (42) and (20) into Eq. (23), we get

$$\hat{\boldsymbol{x}}_k^+ = (\mathbf{I}-\mathbf{K}_\infty\mathbf{H})\hat{\boldsymbol{x}}_k^- + \mathbf{K}_\infty\boldsymbol{y}_k = (\mathbf{I}-\mathbf{K}_\infty\mathbf{H})(\mathbf{F}\hat{\boldsymbol{x}}_{k-1}^+ + \mathbf{G}\boldsymbol{u}_{k-1}) + \mathbf{K}_\infty\boldsymbol{y}_k.$$

$$\therefore \hat{\boldsymbol{x}}_k^+ = (\mathbf{I}-\mathbf{K}_\infty\mathbf{H})\mathbf{F}\hat{\boldsymbol{x}}_{k-1}^+ + (\mathbf{I}-\mathbf{K}_\infty\mathbf{H})\mathbf{G}\boldsymbol{u}_{k-1} + \mathbf{K}_\infty\boldsymbol{y}_k. \tag{43}$$

For the steady-state Kalman filter to be asymptotically stable, the matrix $(\mathbf{I}-\mathbf{K}_\infty\mathbf{H})\mathbf{F}$ must be *convergent* / *discrete-time Hurwitz*, i.e., having eigenvalues whose magnitude is less than 1. The *first* problem is not every system has a solvable DARE, but:

[1] https://docs.scipy.org/doc/scipy/reference/generated/scipy.linalg.solve_discrete_are.html

Theorem 4: [Sim06, Theorem 26]

If $(\mathbf{F}, \mathbf{H})$ is detectable, then the DARE has at least one positive semidefinite solution. Furthermore, at least one such solution results in a marginally stable steady-state Kalman filter.

Even if a solution exists, it may lead to a different $\mathbf{K}_\infty$ than that of Eq. (42), depending on $\mathbf{P}_0$. Even if a solution exists and it leads to the same $\mathbf{K}_\infty$ as that of Eq. (42) regardless of $\mathbf{P}_0$, it may lead to an unstable Kalman filter. Thus, the detectability of $(\mathbf{F}, \mathbf{H})$ is not a sufficient condition. The following theorem is crucial.

Theorem 5: [Sim06, Theorem 23]

If $(\mathbf{F}, \mathbf{H})$ is detectable, and $(\mathbf{F}, \mathbf{G}_q)$ is stabilizable, where $\mathbf{G}_q\mathbf{G}_q^\top = \mathbf{Q}$, then the DARE has a unique positive semidefinite solution, and $(\mathbf{I} - \mathbf{K}_\infty\mathbf{H})\mathbf{F}$ is convergent.

For checking the condition of the theorem above, $\mathbf{G}_q$ can be obtained from $\mathbf{Q}$ by *Cholesky factorization* (MATLAB function `chol`). For positive definite $\mathbf{Q}$, $\mathbf{G}_q$ is a unique lower triangular matrix with positive diagonal entries [Ber09, Fact 8.9.37]. If $\mathbf{Q} = \text{diag}(q_1, \ldots, q_n)$, then $\mathbf{G}_q = \text{diag}(\sqrt{q_1}, \ldots, \sqrt{q_n})$.

Example 5

Consider the first-order discrete-time system of the form

$$\begin{aligned} x[k+1] &= Fx[k] + Gu[k] + w[k], \\ y[k] &= Hx[k] + v[k], \end{aligned}$$

where $H \neq 0$, $Q = \mathsf{Var}\{w\} \neq 0$, $R = \mathsf{Var}\{v\} \neq 0$, determine the constraint on F for the system to have a convergent steady-state Kalman filter.

Solution: Let us determine whether the conditions of Theorem 5 are satisfied.

- Detectability: The observability "matrix" of the first-order system (F, H) is H, so as long as $H \neq 0$, the system is detectable.
- Stabilizability: The controllability "matrix" of the first-order system $(F, \sqrt{Q})$ is $\sqrt{Q}$, so as long as $Q > 0$, the system is stabilizable.

Both conditions above are satisfied by the given system, hence the system has a convergent steady-state Kalman filter regardless of the value of F.

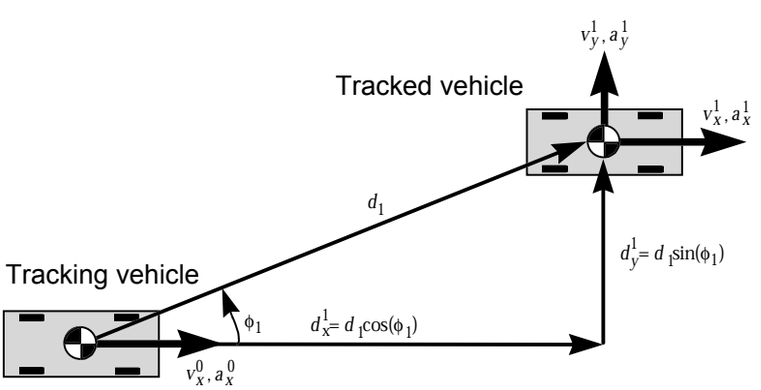


Figure 8: Parameters of Adaptive Cruise Control: v's represent velocities, a's represent accelerations. v_x^o is measured using the car's own speedometer. d_1 is measured using radar or LIDAR. Image from [VN99, Fig. 2].

Example 6

We mentioned in the very beginning that Kalman filters are used in a broad range of industries, therefore it should not come as a surprise that it is widely used in the automotive industry. For this example, we refer to an article published by some BMW engineers in 1999 [VN99], which describes the applications of Kalman filtering to Driver Assistance Systems. The goal of a Driver Assistance System is to reduce the driver's workload, and the chance of accidents. An important part of this system is Adaptive Cruise Control, which is responsible for longitudinal control (with inputs being throttle and brake), depending on the distance behind the preceding vehicle or obstacle.

Referring to Figure 8, we can write down the following continuous-time state-space equations:

$$\begin{bmatrix} \dot{v}_x^o \\ \dot{a}_x^o \\ \dot{d}_x^1 \\ \dot{v}_x^1 \\ \dot{a}_x^1 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} v_x^o \\ a_x^o \\ d_x^1 \\ v_x^1 \\ a_x^1 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} u + \boldsymbol{w}, \tag{44}$$

$$y = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} v_x^o \\ a_x^o \\ d_x^1 \\ v_x^1 \\ a_x^1 \end{bmatrix} + \boldsymbol{\nu}, \tag{45}$$

where

- the input u is the vehicle acceleration which can be determined from known engine maps and brake characteristics;
- the process noise $\boldsymbol{w}$ and measurement noise $\boldsymbol{\nu}$ are assumed to be additive white Gaussian noise.

While $\boldsymbol{\nu}$ can be determined from the speedometer's and range sensor's characteristics, $\boldsymbol{w}$ is assumed to affect only $\dot{a}_x^o$ and $\dot{a}_x^1$ and can be determined experimentally. Using the MATLAB code in Listing 3, we can discretize the continuous-time model into

$$\left[\begin{array}{ccccc|c} 1 & T & 0 & 0 & 0 & T \\ 0 & 1 & 0 & 0 & 0 & 0 \\ -T & -T^2/2 & 1 & T & T^2/2 & -T^2/2 \\ 0 & 0 & 0 & 1 & T & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ \hline 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{array}\right]. \tag{46}$$

Using the same code,

- We can determine the system satisfies the condition of Theorem 5, and hence there is a unique positive semidefinite solution to the associated DARE, and the resultant steady-state Kalman filter is convergent.
- We can calculate the steady-state Kalman gain as

$$\mathbf{K}_\infty = \begin{bmatrix} 0.3617 & -0.0062 \\ 7.9883 & -0.0760 \\ -0.0062 & 0.1815 \\ 0.0247 & 1.8123 \\ 0.1353 & 9.0468 \end{bmatrix}.$$

- Looking at the norm of the eigenvalues of $(\mathbf{I} - \mathbf{K}_\infty \mathbf{H})\mathbf{F}$, we can verify the steady-state Kalman filter is convergent.

Listing 3: MATLAB code for Example 6

```
A = [ 0 1 0 0 0;
      0 0 0 0 0;
     -1 0 0 1 0;
      0 0 0 0 1;
      0 0 0 0 0];
B = [1; 0; 0; 0; 0];
C = [1 0 0 0 0;
     0 0 1 0 0];
syms T t;
F = expm(A*T), G = int(expm(A*t)*B, 0, T), H = C;
F = double(subs(F, T, 0.01));
Q = diag([0 1 0 0 1]); R = diag([0.01 0.01]); % arbitrary
Gq = sqrt(Q); % because Q is diagonal
fprintf('(F, H) has rank %d\n', rank(obsv(F, H)));
fprintf('(F, Gq) has rank %d\n', rank(ctrb(F, Gq)));
[P, ~, eigs] = idare(F', H', Q, R)
K = P*H'/(H*P*H' + R)
```

Attention: MATLAB command `idare`

Note the syntax of `idare` that is used in Example 6: `[X,G,L] = idare(A,B,Q,R)`. Since we are designing an estimator rather than a controller, we apply the duality theorem and substitute `A` with `F'`, and `B` with `H'`.

The second output `G` is not used in Example 6 but it is worth mentioning what it means. `G` is defined[a] as

$$\mathbf{G}^\top = \underbrace{(\mathbf{H}\mathbf{P}\mathbf{H}^\top + \mathbf{R})^{-1}\mathbf{H}\mathbf{P}}_{\mathbf{K}_\infty^\top}\mathbf{F}^\top = \mathbf{K}_\infty^\top \mathbf{F}^\top,$$

where $\mathbf{K}_\infty$ is the steady-state Kalman gain as per Eq. (42). The reason $\mathbf{G}$ is defined as above (by MathWorks) is that

$$\underbrace{\mathbf{F}^\top}_{\text{A}} - \underbrace{\mathbf{H}^\top}_{\text{B}}\mathbf{G}^\top$$

takes the form of A$-$B$\times$(Gain), and

$$\mathbf{F}^\top - \mathbf{H}^\top\mathbf{G}^\top = \mathbf{F}^\top - \mathbf{H}^\top\mathbf{K}_\infty^\top\mathbf{F}^\top = (\mathbf{I} - \mathbf{K}_\infty\mathbf{H})^\top\mathbf{F}^\top$$

has the same eigenvalues as $(\mathbf{I} - \mathbf{K}_\infty\mathbf{H})\,\mathbf{F}$. Note that given any two square matrices $\mathbf{M}_1$ and $\mathbf{M}_2$, we can show through Silvester's determinant theorem (see Lecture 4) that $\mathbf{M}_1\mathbf{M}_2$ and $\mathbf{M}_1^\top\mathbf{M}_2^\top$ have the same eigenvalues.

[a] https://www.mathworks.com/help/control/ref/idare.html

4.2 Practical aspects

Kalman filter implementations are so ubiquitous that it is easy to take the implementation challenges for granted. Important considerations for practical implementations include the following:

- **Offline pre-computation and online lookup**: As demonstrated in Example 4, the Kalman gain and error covariances depend only on the system and noise models, and not on the measurements $\boldsymbol{y}_k$, so their computation can be done offline, and their values looked up at

run time; this can make the difference to whether a resource-constrained system can perform Kalman filtering in real time.

- **Self-diagnostics**: The Kalman filter is also capable of self-diagnostics: the innovation $(\boldsymbol{y}_k - \mathbf{H}_k\hat{\boldsymbol{x}}_k^-)$ is expected to be zero-mean and white with covariance $(\mathbf{H}_k\mathbf{P}_k^-\mathbf{H}_k^\top + \mathbf{R}_k)$, so any run-time observation that significantly contradicts this expectation suggests potential error in the system model, noise model, or implementation [Sim06, p. 130].

- **Square root filtering**: Computing hardware has limited precision. Due to the finite precision of a microcontroller, the error covariance, which is supposed to be positive semidefinite, may become indefinite (having both positive and negative eigenvalues) or even nonsymmetric (remember only symmetric matrices and certain square matrices can be positive semidefinite). This was especially a problem in the 1960s, when computers had short word lengths. *Square root filtering* is a way to mathematically increase the precision of the Kalman filter when hardware precision is challenged; this technique was in fact developed for Apollo [Sim06, Sect. 6.3]. The idea of square root filtering can be summarized as the observations below:

  - The positive-semidefinite error covariance $\mathbf{P}$ can be expressed as $\mathbf{P} = \mathbf{S}\mathbf{S}^\top$, where $\mathbf{S}$ can be obtained using Cholesky factorization. Since $\mathbf{P}$ is positive semidefinite, $\mathbf{S}$ is a nonunique lower triangular matrix with nonnegative diagonal entries [Ber09, Fact 8.9.37].

  - The *condition number* of $\mathbf{S}$ equals the square root of the condition number of $\mathbf{P}$ [Sim06, Eq. (6.48)]. In linear algebra, the condition number is a measure of how well or badly conditioned a matrix equation is. It is the ratio of the largest singular value to the smallest singular value of $\mathbf{A}$ in matrix equation $\mathbf{A}\boldsymbol{x} = \square$. The base-$b$ logarithm of the condition number provides a worst-case estimate of the number of base-b digits that are lost in solving the matrix equation[2]. The smaller the condition number, the wider range of numbers (higher precision) can be accommodated.

  - In the Kalman filter algorithm, every instance of $\mathbf{P}$ is replaced with $\mathbf{S}\mathbf{S}^\top$; and instead of $\mathbf{P}$, $\mathbf{S}$ is stored.

- **Information filtering**: The Kalman filter suffers from poor performance when $\mathbf{P}_0^+$ is large. If we define the *information matrix*, $\boldsymbol{\mathcal{I}}$, as the inverse of the error covariance, i.e.,

$$\boldsymbol{\mathcal{I}} = \mathbf{P}^{-1}, \tag{47}$$

  then $\boldsymbol{\mathcal{I}}_0^+ = \mathbf{0}$ corresponds to $\mathbf{P}_0^+ = \boldsymbol{\infty}$, i.e., the case where there is absolute uncertainty with $\boldsymbol{x}_0$ [Sim06, Sect. 6.2]. Furthermore, Eqs. (21), (22) and (26) can be respectively written as

$$\boldsymbol{\mathcal{I}}_k^- = \mathbf{Q}_{k-1}^{-1} - \mathbf{Q}_{k-1}^{-1}\mathbf{F}_{k-1}(\boldsymbol{\mathcal{I}}_{k-1}^+ + \mathbf{F}_{k-1}^\top\mathbf{Q}_{k-1}^{-1}\mathbf{F}_{k-1})^{-1}\mathbf{F}_{k-1}^\top\mathbf{Q}_{k-1}^{-1}, \tag{48}$$

$$\mathbf{K}_k = (\boldsymbol{\mathcal{I}}_k^+)^{-1}\mathbf{H}_k^\top\mathbf{R}_k^{-1}, \tag{49}$$

$$\boldsymbol{\mathcal{I}}_k^+ = \boldsymbol{\mathcal{I}}_k^- + \mathbf{H}_k^\top\mathbf{R}_k^{-1}\mathbf{H}_k. \tag{50}$$

  Replacing $\mathbf{P}_k^-, \mathbf{P}_k^+$ with $\boldsymbol{\mathcal{I}}_k^-, \boldsymbol{\mathcal{I}}_k^+$, and replacing the corresponding update equations in the Kalman filter with Eqs. (48)–(49), gives us the *information filter*. If $\mathbf{H}_k^\top\mathbf{R}_k^{-1}\mathbf{H}_k$ is time-invariant, then it can be pre-computed offline, and Eq. (50) is just a summation operation with a constant time complexity; this is a well-known advantage of the information filter that is widely leveraged for simultaneous localization and mapping (SLAM) [TLK+04].

[2] http://mathworld.wolfram.com/ConditionNumber.html

5 Extended Kalman filter

The Kalman filter was designed for linear systems expressible by Eqs. (17)–(18). Given a nonlinear model, the Kalman filter can be used with the linearized version of the model, but for more accurate state estimation, the original nonlinear model should be used, and to estimate the state of the nonlinear model, the extended Kalman filter can be used.

For discrete-time nonlinear systems of the form:

$$\boldsymbol{x}_k = \boldsymbol{f}_{k-1}(\boldsymbol{x}_{k-1}, \boldsymbol{u}_{k-1}) + \boldsymbol{w}_{k-1}, \quad \boldsymbol{w}_k \sim \mathcal{N}(\boldsymbol{0}, \mathbf{Q}_k), \tag{51}$$

$$\boldsymbol{y}_k = \boldsymbol{h}_k(\boldsymbol{x}_k) + \boldsymbol{v}_k, \quad \boldsymbol{v}_k \sim \mathcal{N}(\boldsymbol{0}, \mathbf{R}_k), \tag{52}$$

where $\boldsymbol{f}_{k-1}$ and $\boldsymbol{h}_k$ are differentiable nonlinear functions, we can use the extended Kalman filter (EKF), which is essentially the Kalman filter for the first-order linearized version of Eqs. (51)–(52) based on Taylor series expansion. The EKF, although not "optimal", has been shown to be effective for a wide range of nonlinear systems, and by rule of thumb, it should be the first tool to consider whenever a nonlinear filtering problem is encountered. The EKF for system (51)–(52) is similar to the Kalman filter except the $\mathbf{F}$ and $\mathbf{H}$ matrices are now Jacobians [Sim06, p. 409]:

Extended Kalman filter

Initialization ($k = 0$):

$$\hat{\boldsymbol{x}}_0^+ = \mathsf{E}\{\boldsymbol{x}_0\}, \quad \mathbf{P}_0^+ = \mathsf{E}\left\{(\boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0^+)(\boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0^+)^\top\right\}. \tag{53}$$

Time update / prediction update equations for propagating *a priori* estimates ($k = 1, 2, \ldots$):

$$\hat{\boldsymbol{x}}_k^- = \boldsymbol{f}_{k-1}(\hat{\boldsymbol{x}}_{k-1}^+, \boldsymbol{u}_{k-1}), \tag{54}$$

$$\mathbf{P}_k^- = \mathbf{F}_{k-1}\mathbf{P}_{k-1}^+\mathbf{F}_{k-1}^\top + \mathbf{Q}_{k-1}, \text{ where } \mathbf{F}_{k-1} = \left.\frac{\partial \boldsymbol{f}_{k-1}}{\partial \boldsymbol{x}}\right|_{\substack{\boldsymbol{x}=\hat{\boldsymbol{x}}_{k-1}^+ \\ \boldsymbol{u}=\boldsymbol{u}_{k-1}}}. \tag{55}$$

Kalman gain update:

$$\mathbf{K}_k = \mathbf{P}_k^-\mathbf{H}_k^\top(\mathbf{H}_k\mathbf{P}_k^-\mathbf{H}_k^\top + \mathbf{R}_k)^{-1}, \text{ where } \mathbf{H}_k = \left.\frac{\partial \boldsymbol{h}_k}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}=\hat{\boldsymbol{x}}_k^-}. \tag{56}$$

Measurement update equations for updating *a posteriori* estimates:

$$\hat{\boldsymbol{x}}_k^+ = \hat{\boldsymbol{x}}_k^- + \mathbf{K}_k(\boldsymbol{y}_k - \boldsymbol{h}_k(\hat{\boldsymbol{x}}_k^-)), \tag{57}$$

Joseph stabilized version: $\mathbf{P}_k^+ = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}_k^-(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)^\top + \mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^\top$. (58)

Standard form: $\mathbf{P}_k^+ = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}_k^-$. (59)

Information filter form: $\mathbf{P}_k^+ = [(\mathbf{P}_k^-)^{-1} + \mathbf{H}_k^\top\mathbf{R}_k^{-1}\mathbf{H}_k]^{-1}$. (60)

The algorithm above is readily applicable to discrete-time nonlinear systems of the form in Eqs. (51)–(52). However, many practical problems are governed by continuous-time dynamics, even if the measurements are obtained at discrete time instants. To continuous-time nonlinear systems with discrete-time measurements:

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t)) + \boldsymbol{w}(t), \quad \boldsymbol{w}(t) \sim \mathcal{N}(\boldsymbol{0}, \mathbf{Q}(t)), \tag{61}$$

$$\boldsymbol{y}_k = \boldsymbol{h}_k(\boldsymbol{x}_k) + \boldsymbol{v}_k, \quad \boldsymbol{v}_k \sim \mathcal{N}(\boldsymbol{0}, \mathbf{R}_k), \tag{62}$$

where $\boldsymbol{f}$ and $\boldsymbol{h}_k$ are differentiable, we can apply the hybrid EKF [Sim06, p. 405], [CJ12, Table 3.7]:

Hybrid extended Kalman filter

Initialization ($k = 0$):

$$\hat{\boldsymbol{x}}_0^+ = \mathsf{E}\{\boldsymbol{x}_0\}, \quad \mathbf{P}_0^+ = \mathsf{E}\left\{(\boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0^+)(\boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0^+)^\top\right\}. \tag{63}$$

Time update / prediction update equations for propagating *a priori* estimates ($k = 1, 2, \ldots$):

$$\hat{\boldsymbol{x}}_k^- = \hat{\boldsymbol{x}}(t_k), \tag{64}$$

where $\hat{\boldsymbol{x}}(t)$ is the solution to $\dot{\hat{\boldsymbol{x}}}(t) = \boldsymbol{f}(\hat{\boldsymbol{x}}(t), \boldsymbol{u}(t))$, with boundary condition $\hat{\boldsymbol{x}}(t_{k-1}) = \hat{\boldsymbol{x}}_{k-1}^+$.

$$\mathbf{P}_k^- = \mathbf{P}(t_k), \tag{65}$$

where $\mathbf{P}(t)$ is the solution to $\dot{\mathbf{P}}(t) = \mathbf{F}(t)\mathbf{P}(t) + \mathbf{P}(t)\mathbf{F}(t)^\top + \mathbf{Q}(t)$, with boundary condition $\mathbf{P}(t_{k-1}) = \mathbf{P}_{k-1}^+$, where

$$\mathbf{F}(t) = \left.\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}\right|_{\substack{\boldsymbol{x}=\hat{\boldsymbol{x}}(t) \\ \boldsymbol{u}=\boldsymbol{u}(t)}}. \tag{66}$$

Kalman gain update:

$$\mathbf{K}_k = \mathbf{P}_k^-\mathbf{H}_k^\top(\mathbf{H}_k\mathbf{P}_k^-\mathbf{H}_k^\top + \mathbf{R}_k)^{-1}, \text{ where } \mathbf{H}_k = \left.\frac{\partial \boldsymbol{h}_k}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}=\hat{\boldsymbol{x}}_k^-}. \tag{67}$$

Measurement update equations for updating *a posteriori* estimates:

$$\hat{\boldsymbol{x}}_k^+ = \hat{\boldsymbol{x}}_k^- + \mathbf{K}_k(\boldsymbol{y}_k - \boldsymbol{h}_k(\hat{\boldsymbol{x}}_k^-)), \tag{68}$$

Joseph stabilized version: $\mathbf{P}_k^+ = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}_k^-(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)^\top + \mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^\top$. (69)

Standard form: $\mathbf{P}_k^+ = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}_k^-$. (70)

Information filter form: $\mathbf{P}_k^+ = [(\mathbf{P}_k^-)^{-1} + \mathbf{H}_k^\top\mathbf{R}_k^{-1}\mathbf{H}_k]^{-1}$. (71)

The EKF performs well if the first-order linearization error is small. Otherwise, the unscented Kalman filter ("unscented" because it does not "stink") or the particle filter should be considered.

References

[Ahr05] P. Ahrendt, The multivariate gaussian probability distribution, https://www.academia.edu/download/49874923/The_Multivariate_Gaussian_Probability_Di20161026-27105-77g7a0.pdf, 2005.

[Ber09] D. R. Bernstein, *Matrix Mathematics: Theory, Facts, and Formulas*, 2nd ed., Princeton University Press, 2009.

[Che12] S. Y. Chen, Kalman filter for robot vision: A survey, *IEEE Transactions on Industrial Electronics* **59** no. 11 (2012), 4409–4420. https://doi.org/10.1109/TIE.2011.2162714.

[CJ12] J. L. Crassidis and J. L. Junkins, *Optimal Estimation of Dynamic Systems*, 2nd ed., CRC Press, 2012.

[Dei82] A. S. Deif, *Advanced matrix theory for scientists and engineers*, Abacus Press, 1982.

[DDS14] R. Doraiswami, C. Diduch, and M. Stevenson, *Identification of Physical Systems: Applications to Condition Monitoring, Fault Diagnosis, Soft Sensor and Controller Design*, Wiley, 2014.

[Far12] R. Faragher, Understanding the basis of the kalman filter via a simple and intuitive derivation [lecture notes], *IEEE Signal Processing Magazine* **29** no. 5 (2012), 128–132. https://doi.org/10.1109/MSP.2012.2203621.

[GRG01] D. R. S. Geoffrey R. Grimmett, *Probability and Random Processes*, 3rd ed., Oxford University Press, 2001.

[Gib11] B. Gibbs, *Advanced Kalman Filtering, Least-Squares and Modeling: A Practical Handbook*, Wiley, 2011.

[GA15] M. S. Grewal and A. P. Andrews, *Kalman Filtering: Theory and Practice Using MATLAB®*, 4th ed., John Wiley & Sons, Inc., 2015.

[Hay01] S. Haykin (ed.), *Kalman Filtering and Neural Networks*, Wiley-Interscience, 2001.

[Hay09] S. S. Haykin, *Neural networks and learning machines*, 3rd ed., Pearson Education, Inc., 2009.

[HRW12] J. Humpherys, P. Redd, and J. West, A fresh look at the Kalman filter, *SIAM Review* **54** no. 4 (2012), 801–823. https://doi.org/10.1137/100799666.

[Jek00] C. Jekeli, *Inertial Navigation Systems with Geodetic Applications*, De Gruyter, 2000.

[KC14] D. P. Kroese and J. C. Chan, *Statistical Modeling and Computation*, Springer, 2014.

[Mey01] C. Meyer, *Matrix Analysis and Applied Linear Algebra*, SIAM, 2001.

[MC12] S. L. Miller and D. Childers, *Probability and Random Processes With Applications to Signal Processing and Communications*, 2nd ed., Academic Press, 2012.

[Nah69] N. E. Nahi, *Estimation theory and applications*, John Wiley & Sons, Inc., 1969.

[Pri81] M. Priestley, *Spectral Analysis and Time Series: Volume 1: Univariate Series*, Academic Press, 1981.

[Sär13] S. Särkkä, *Bayesian Filtering and Smoothing*, Cambridge University Press, 2013.

[Sim06] D. Simon, *Optimal State Estimation: Kalman, H_∞, and Nonlinear Approaches*, John Wiley & Sons, Inc., 2006.

[TLK+04] S. Thrun, Y. Liu, D. Koller, A. Y. Ng, Z. Ghahramani, and H. Durrant-Whyte, Simultaneous localization and mapping with sparse extended information filters, *The International Journal of Robotics Research* **23** no. 7-8 (2004), 693–716.

[VN99] P. J. Venhovens and K. Naab, Vehicle dynamics estimation using kalman filters, *Vehicle System Dynamics* **32** no. 2-3 (1999), 171–184. https://doi.org/10.1076/vesd.32.2.171.2088.