A Smooth Introduction to the Kalman Filter

Steven Bell

November 6, 2024

Abstract

The Kalman filter is a common and versatile solution for signal filtering and data fusion tasks. However, most literature discussing it is abstract and math-heavy, which is intimidating and confusing for many newcomers. This document attempts to explain the Kalman filter from the ground up, starting with the one-dimensional case and building up to the abstract vector case. Software examples are presented in MATLAB, Python, and Java. A knowledge of statistics and linear algebra is critical to fully understanding the algorithm, but the basic concepts presented should be accessible without these.

1 Introduction

Suppose you have a signal you want to measure: a voltage from a sensor, a radio transmission, an audio clip, or anything else. In an ideal world, you would have a device that would convert the signal to a number (or series of numbers) that you could use. Figure 1 shows an example signal that we will use throughout the rest of this document.

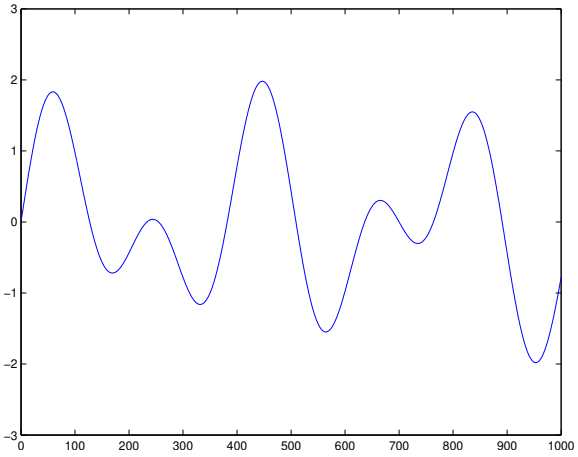

Figure 1: Example signal which will be used in this document.

Unfortunately, the real world is filled with noise, which inevitably corrupts the signal. Depending on the signal, noise could be caused by another radio transmission, currents in nearby wires, random collisions of electrons, or the quantum nature of photons. Even the process of converting the signal into digital values adds noise. The end result is a signal that looks like Figure 2.

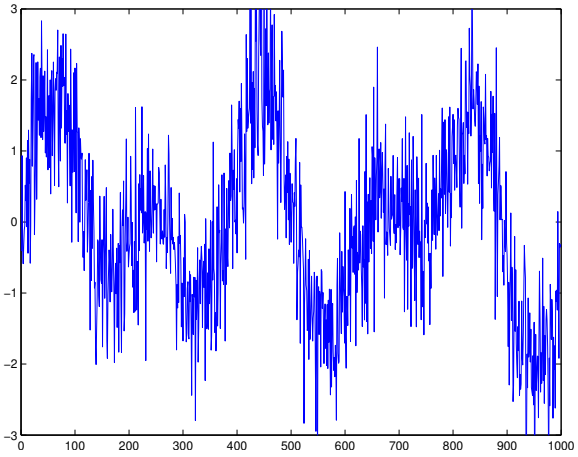

Figure 2: Example signal corrupted by noise.

The filter's job is to remove as much of the noise as possible, producing an accurate representation of the original signal.

Because measurements are never perfect, sometimes you want to combine several measurements together to get a single result that is more accurate than any one measurement. Ideally, you would combine them in a way that gives more weight to the measurements that are likely to be correct, and less weight to the measurements that are less reliable. The Kalman filter provides a way to do exactly that.

2 A little statistical background

The Kalman filter is rooted in statistics, so it's important to understand a little bit of statistical background first. Suppose we have a single measurement of our signal at a specific point in time, which we will designate X. This is called a random variable, because its value is random. Although we can't determine what X is prior to measuring it, we can describe it in terms of its mean (average) and its variance. The mean is also called the "expected value" of the random variable, and is written $\mathrm{E}[X]$. Given no other information, the best estimate for X is the value $\mathrm{E}[X]$.

Variance, as its name suggests, quantifies how much a signal varies. Specifically, it's the square of the standard deviation (often written as σ_x), so it it written as σ_x^2. Mathematically, variance is calculated as

$$\sigma_x^2 = \mathrm{E}\left[(X - \mathrm{E}[X])^2\right]$$

With a little algebra[1], this equation becomes

$$\sigma_x^2 = \mathrm{E}\left[X^2\right] - \mathrm{E}[X]^2$$

When working with a sensor, the value we care about is the difference between the sensor reading and the actual value. That is, we need to quantify the variance of the noise. Here we want the variance to be as small as possible. If the noise variance is zero, we can be certain that the sensor reading is exactly the physical value. If the variance is large, then the reading tells us very little about the physical measurement.

Covariance is a measure of how correlated two signals are. If we have two signals, X and Y, then the covariance between them is[2]

$$\mathrm{Cov}(X, Y) = \mathrm{E}[XY] - \mathrm{E}[X]\,\mathrm{E}[Y]$$

If two variables have a high covariance (whether positive or negative), then knowing one variable gives us information about another. Conversely, if two variables have a covariance of zero, then knowing one variable gives us no information about the other[3].

Mathematically, we will define a signal to be a series of random values indexed by time. That is, for each point in time, we have a unique measurement of the input, and the series of these measurements is our signal. If we designate our signal as $\mathbf{X}$, then we can describe the individual measurements as X_0, $X_1 \ldots X_n$ for time values $k = 1$, $k = 2$, to $k = n$. Each measurement is a random variable on its own, but it may also be correlated to other variables.

2.1 What *is* the Kalman filter?

So what exactly is the Kalman filter? Many have a hazy idea that it's the "right solution" for filtering and sensor fusion[4]. Others feel that it is simply statistical black magic or technological voodoo. While discussion of the Kalman filter is typically couched in academic terms and dense mathematics, the filter itself is beautifully simple. The Kalman filter is an iterative algorithm which produces an optimal estimate of a set of variables, given a set of noisy measurements.

By iterative, we mean that the algorithm works on the previous estimate and the current measurement only. The algorithm does not have to save all of the past measurements, or perform complex calculations into the future. It only has to update the current estimate with the new measurements.

[1] Ok, a little algebra and some statistics: $\sigma_x^2 = \mathrm{E}\left[(X - \mathrm{E}[X])^2\right] = \mathrm{E}\left[X^2 - 2X \cdot \mathrm{E}[X] + \mathrm{E}[X]^2\right]$. But since $\mathrm{E}[X]$ is a constant and expectation is a linear operator, $\sigma_x^2 = \mathrm{E}\left[X^2\right] - 2\mathrm{E}[X]\,\mathrm{E}[X] + \mathrm{E}[X]^2 = \mathrm{E}\left[X^2\right] - \mathrm{E}[X]^2$. Pick up a good statistics text to understand exactly why this works out.

[2] Note that the covariance of X with itself is exactly the variance of X.

[3] This is not precisely true: suppose we have two random variables. X equals 1 and -1 with probability 1/4, and 0 with probability 1/2. Y equals 0 if $X \neq 0$ and 1 or -1 if $X = 0$. In this case, the covariance is zero, but knowing that $X = 1$ tells us the value of Y exactly.

[4] I myself was in this category for many years.

By optimal, we mean that the estimate of the output minimizes the mean-squared error (MSE) of the estimate. That is, given the inputs, the Kalman filter produces the result that is statistically guaranteed to have the minimum error [5].

In academic terms, the "set of variables" is called the "state space" of the system - the variables we care about. This might be position and velocity, pressure, acidity, or anything else we care to measure. For the Kalman filter to work, we must have a model that describes how the state space changes over time. In the case of position, we can multiply velocity by time to get a new position. Conversely, the state may not change at all: we might expect the acidity of a chemical vat to stay constant, for example, until a measurement tells us otherwise.

3 One dimensional case

The first example we will consider is the one-dimensional case. A diagram showing how the variables are related is shown in Figure 3. Here, the unknown random variable is X, and our objective is to estimate X_i given Y_i and all past measurements of Y (that is, Y_{i-1}, Y_{i-2}, all the way back to Y_0). At each time step, the input U_i is added to X_i to get the new value for X. This is an input which changes the actual state of the system. On the other side, V_i is noise that gets added to the true value to in the measurement process. This does not change the state of the system; it merely corrupts the measurement. For now, we will assume that U and V are zero-mean independent white noise processes. That is, each U_i is unrelated to any other U_i and $\mathrm{E}[U_i] = 0$.

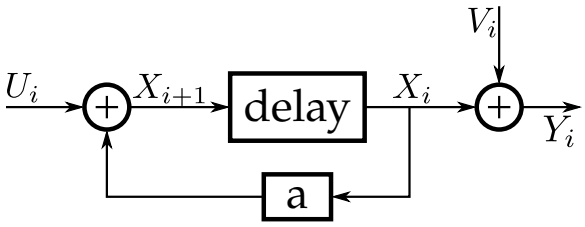

Figure 3: State diagram for a one-dimensional system. At each time increment, $X_{i+1} = aX_i + U_i$ and $Y_i = X_i + V_i$.

The Kalman filter consists of two basic steps: "predict", which predicts the next value based on the past measurements, and "update" which corrects the prediction based on the latest measurement. Along with updating the estimate at each step, the filter also updates the variance of the estimate. By keeping track of the variance, the filter can optimally combine the new measurements with the previous estimate to obtain the best result. These steps are repeated over and over again, continually updating as the estimate as new measurements come in.

There are several key variables in the equations which follow.[6]

- $\hat{X}$, the estimate of X, pronounced "X-hat". Accordingly, X_i is the value of $\mathbf{X}$ at time i, X_{i+1} the value at time $i+1$, and so forth. I will use the notation $\hat{X}_{i|i-1}$ to refer to the estimate of X given all of the measurements up to Y_{i-1}, and $\hat{X}_{i|i}$ to be the estimate given the measurements up to Y_i.
- σ_i^2, the variance of the estimate ($\hat{X}$) at time i. This would more properly be written $\sigma_{\hat{X}_i}$, but σ_i is notationally simpler and the other variances will be subscripted so there is no ambiguity.
- σ_x^2, the variance of X. In many practical cases, this value is an estimate rather than a known value. We will see later how this value affects the operation of the filter.

[5] More correctly, it produces the best *linear* estimate in the mean-squared-error sense.

[6] Numerous other symbols are used for these variables, so my selection is rather arbitrary. Wikipedia uses F and P, in place of A and σ_X^2. Wikipedia also includes B (control input) and H (observation matrix), which are discussed in section 6.

- a, a value which describes how the state changes from one time step to the next. This will take on more significance in higher-dimensional cases, where the value of one variable affects how another changes.
- $Q = \sigma_U^2$, the variance of the system input U.
- $N = \sigma_V^2$, the variance of the noise input V.

Before we look at the filter equations, let's make a few notes of intuition about how the system behaves at several edge cases:

- If $N = 0$, then V is always zero, and $Y_i = X_i + V_i$ collapses to $Y_i = X_i$. In English, when the noise is zero, our measurement is exactly the system state - which would make the Kalman filter superfluous!
- If $a = 0$, then the update equation $X_{i+1} = aX_i + U_i$ becomes $X_{i+1} = U_i$. Here, the current state does not depend on the past state at all; it only depends on the input. Intuitively, this means that none of the past measurements are useful; only the current measurement will tell us anything about the current state.
- If $a = 1$ and $Q = 0$, then $X_{i+1} = aX_i + U_i$ becomes $X_{i+1} = X_i$, which means that the state stays exactly the same forever. The task here is to estimate a single constant from a series of measurements, and here, every measurement is useful, regardless of how old it is.

Now that we have the terms out of the way, we can dive into the meat of the algorithm.

Prediction step

In the prediction step, a new estimate is calculated for time i using the previous estimate. At this point, we do not yet consider Y_i.

$$\hat{X}_{i|i-1} = a\hat{X}_{i-1} \tag{1}$$

$$\sigma_{i|i-1}^2 = a^2\sigma_{i-1}^2 + Q_i \tag{2}$$

Essentially what we're doing here is projecting the current state into the future, given what we know about how the state changes over time. We also update the variance correspondingly. Equation 1 should make sense; the reasoning behind Equation 2 will be shown in section 5.

Update step

In the update step, we use the new measurement Y_i to improve our previous prediction.

$$\hat{X}_{i|i} = \hat{X}_{i|i-1} + k \cdot \left(Y_i - \hat{X}_{i|i-1}\right) \tag{3}$$

$$\sigma_{i|i}^2 = (1 - k)\,\sigma_{i|i-1} \tag{4}$$

Where k, known as the "kalman gain" is given by

$$k = \frac{\sigma_{i|i-1}^2}{\sigma_{i|i-1}^2 + N} \tag{5}$$

Before we move on to an example, stop and understand what is going on here. Note that X, the actual signal, is not in the equations. That's because we have no way to know the value of X for sure; all we have are the measurements. Neither do we know U or V - we only know their variances.

Initialization

As mentioned before, the core of the Kalman filter is an iterative algorithm running in a loop. Each update depends on the past, so before the filter starts its loop, it knows nothing. Thus, we have to use our best "blind guess" for the initial values of $\hat{X}$ and σ^2:

$$\hat{X}_0 = \mathrm{E}[X] \tag{6}$$

$$\sigma_0^2 = \sigma_X^2 \tag{7}$$

Now we have all of the pieces, and it's time for an example!

Example: Estimating a constant

Suppose we want to measure the distance to an object using a series of noisy ultrasonic rangefinder measurements. Assuming that neither the sensor nor the object moves, the distance will be a constant. The measurements, however, may fluctuate significantly, so we want to filter the values. Since $X_{i+1} = aX_i + U_i = X_i$, $a = 1$ and $U_i = 0$ for all i, which means that $Q = 0$.

Using these values, the prediction equations simplify to

$$\hat{X}_{i|i-1} = \hat{X}_{i-1}$$

$$\sigma_{i|i-1}^2 = \sigma_{i-1}^2$$

The update equations remain the same. Figure 4 shows the results of using this filter.

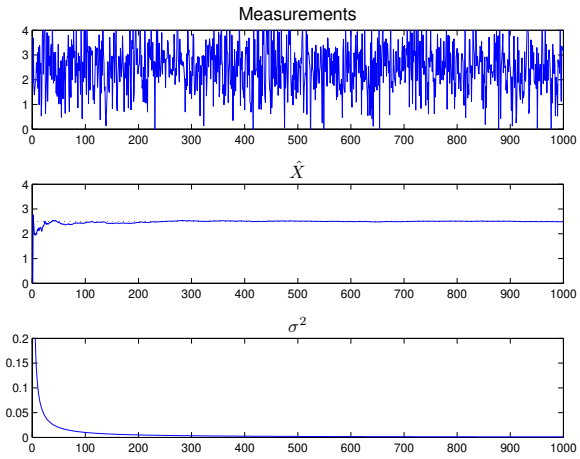

Figure 4: Constant estimated with the Kalman filter. Here, $N = 1$, $\mathrm{E}[X] = 0$, and $\sigma_X^2 = 3$. As time progresses, the estimate becomes closer to the actual value and the variance of the estimate approaches zero.

Example: Estimating a changing signal

Now, let's try to estimate the changing signal introduced at the beginning of the document. Here we use the prediction and update equations in their full forms. First, we have to define values for Q, N, and a. The easiest value for a is 1, since without a model for how the signal behaves, the best we can do is to guess that it remains constant unless a measurement tells us otherwise. For now, we will let $Q = 0.01$ and $N = 0.5$. Running the filter, we get the result shown in Figure 5. Most of the noise is filtered out, and the result

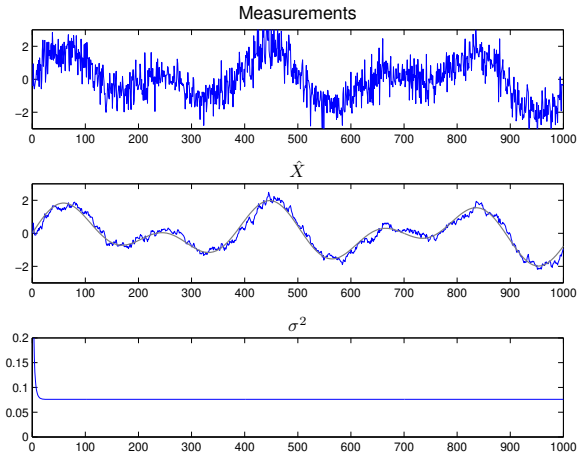

Figure 5: Varying signal estimated with the Kalman filter. For this run, $Q = 0.01$, $N = 0.5$, $\mathrm{E}[X] = 0$, and $\sigma_X^2 = 1$. The original signal is overlaid in gray on the filtered signal for comparison.

closely resembles the original signal. Note that unlike the previous example, the variance of the estimate never goes to zero. Instead, it approaches a value related to Q and N[7].

MATLAB, Python, and Java code examples are presented in appendices A through C. Now that we've seen the Kalman filter in action, let's explore how the different parameters affect its operation.

- Graphs of the basic example
  - What happens when you change the terms? Why does this make sense?

4 Two-measurement case

There's no reason that $\boldsymbol{A}$ can't change from one update to the next, provided you use it consistently across each update iteration.

5 Full derivation

In this section, we'll go through a complete derivation of the Kalman filter.

The variance of a sum of two uncorrelated variables is the sum of the variances,

$$\mathrm{Var}(X + U) = \mathrm{Var}(X) + \mathrm{Var}(U)$$

Once again, $\boldsymbol{X}$ represents our state. Here, we will let $\boldsymbol{X}$ be a vector of any length.

5.1 Limit

To derive the limit that it the variance approaches, we can use the variance update equation:

$$\sigma_i^2 = \sigma_{i-1}^2$$

[7]Specifically, it approaches $\frac{1}{2}\left(Q + \sqrt{Q^2 + 4QN}\right)$. The derivation of this formula is presented in section

6 A few extensions

For completeness
$\boldsymbol{H}$ is used as the matrix to map the measurements to the state
Variance could change every timestep

7 Useful links and references

http://www.cs.unc.edu/~welch/kalman/kalmanIntro.html Corresponding Python code: http://www.scipy.org/Cookbook/Ka

A MATLAB/Octave code

B Python code

C Java code