Kalman Filter Research

Pengfei Nie[1]

[1] *V2X Software Engineer pfnie@yahoo.com https://github.com/pengfeinie*

Publication date: 20221006

Abstract— Most modern systems have numerous sensors that estimate hidden (unknown) states based on a series of measurements.the measurement uncertainty depends on many external factors.The Kalman Filter is one of the most important and common estimation algorithms. The Kalman Filter produces estimates of hidden variables based on inaccurate and uncertain measurements. Also, the Kalman Filter predicts the future system state based on past estimations.The filter is named after Rudolf E. Kálmán (May 19, 1930 – July 2, 2016). In 1960, Kálmán published his famous paper describing a recursive solution to the discrete-data linear filtering problem.

Keywords— Kalman Filter, Measurements, Estimation Algorithms , Prediction.

I. Introduction

In statistics and control theory, Kalman filtering, also known as linear quadratic estimation, is an algorithm that uses a series of measurements observed over time, containing statistical noise and other inaccuracies, and produces estimates of unknown variables that tend to be more accurate than those based on a single measurement alone, by estimating a joint probability distribution over the variables for each timeframe. The filter is named after Rudolf E. Kálmán, one of the primary developers of its theory.

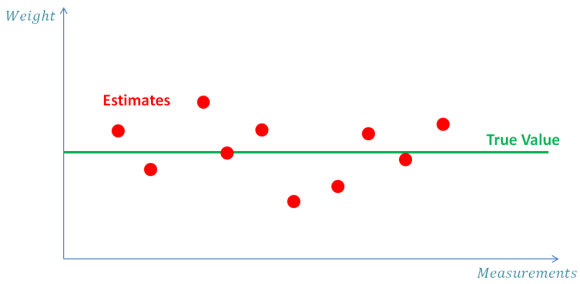
The algorithm works in a two-step process. In the prediction step, the Kalman filter produces estimates of the current state variables, along with their uncertainties. Once the outcome of the next measurement (necessarily corrupted with some amount of error, including random noise) is observed, these estimates are updated using a weighted average, with more weight being given to estimates with higher certainty. The algorithm is recursive. It can run in real time, using only the present input measurements and the previously calculated state and its uncertainty matrix; no additional past information is required.

Extensions and generalizations to the method have also been developed, such as the extended Kalman filter and the unscented Kalman filter which work on nonlinear systems. The underlying model is a hidden Markov model where the state space of the latent variables is continuous and all latent and observed variables have Gaussian distributions. Also, the Kalman filter has been successfully used in multi-sensor fusion, and distributed sensor networks to develop distributed or consensus Kalman filter.

II. Weight the Gold

Now we are ready for the first simple example. In this example, we estimate a static system's state. A static system is a system that doesn't change its state over a reasonable period. For instance, the static system could be a tower, and the state would be its height. In this example, we estimate a gold bar's weight. But the measurements do include random noise.The system is the gold bar, and the system's state is the weight of the gold bar. The system's dynamic model is constant since we assume that the weight doesn't change over short periods. In order to estimate the system's state (i.e., the weight value), we can make multiple measurements and average them.

a. Estimate The Weight of Gold



$$\hat{x}_{n,n} = \frac{1}{n}(z_1 + z_2 + \ldots + z_{n-1} + z_n) = \frac{1}{n}\sum_{i=1}^{n}(z_i)$$

Table 1: Example Notation

variables	description
x	is the true value of the weight.
z_n	is the measured value of the weight at time n.
$\hat{x}_{n,n}$	is the estimate of x at time n.
$\hat{x}_{n+1,n}$	is the predicted of the future state $(n+1)$ of x.
$\hat{x}_{n-1,n-1}$	is the estimate of x at time $n-1$.
$\hat{x}_{n,n-1}$	is the predicted of the future state n of x.

$$\hat{x}_{n,n} = \frac{1}{n}\sum_{i=1}^{n}(z_i)$$
$$= \frac{1}{n}\left(\sum_{i=1}^{n-1}(z_i) + z_n\right)$$
$$= \frac{1}{n}\sum_{i=1}^{n-1}(z_i) + \frac{1}{n}z_n$$
$$= \frac{1}{n}\frac{n-1}{n-1}\sum_{i=1}^{n-1}(z_i) + \frac{1}{n}z_n$$
$$= \frac{n-1}{n}\frac{1}{n-1}\sum_{i=1}^{n-1}(z_i) + \frac{1}{n}z_n$$
$$= \frac{n-1}{n}\hat{x}_{n-1,n-1} + \frac{1}{n}z_n$$
$$= \hat{x}_{n-1,n-1} - \frac{1}{n}\hat{x}_{n-1,n-1} + \frac{1}{n}z_n$$
$$= \hat{x}_{n-1,n-1} + \frac{1}{n}(z_n - \hat{x}_{n-1,n-1})$$

Since the dynamic model in this example is static, i.e., the weight of the gold bar doesn't change over time, the predicted state of x equals the estimated state of x :

$$\hat{x}_{n,n-1} = \hat{x}_{n-1,n-1}$$

Based on the above, the estimate of the current state $\hat{x}_{n,n}$, can be written as follows:

$$\hat{x}_{n,n} = \hat{x}_{n,n-1} + \frac{1}{n}(z_n - \hat{x}_{n,n-1})$$

The factor $\frac{1}{n}$ is specific for our example. We will discuss the vital role of this factor later.The subscript n indicates that the it can change with every iteration.

The term $(z_n - \hat{x}_{n,n-1})$ is the "measurement residual".In this example, $\frac{1}{n}$ decreases as n increases. In the beginning, we don't have enough information about the current state; thus, the first estimation is based on the first measurement $\frac{1}{n}\big|_{n=1} = 1$. As we continue, each successive measurement has less weight in the estimation process, since $\frac{1}{n}$ decreases. Let's continue with the example. Before we make the first measurement, we can guess (or rough estimate) the gold bar weight simply by reading the stamp on the gold bar. It is called the Initial Guess, and it is our first estimate. The Kalman Filter requires the initial guess as a preset, which can be very rough.

1. Zero Iteration

Initialization
Our initial guess of the gold bar weight is 1000 grams. The initial guess is used only once for the filter initiation. Thus, it won't be required for success iterations.

$$\hat{x}_{0,0} = 1000g$$

Prediction
The weight of the gold bar is not supposed to change. Therefore, the dynamic model of the system is static. Our next state estimate (prediction) equals the initialization:

$$\hat{x}_{1,0} = \hat{x}_{0,0} = 1000g$$

2. First Iteration

Step 1
Making the weight measurement with the scales:

$$z_1 = 1030g$$

Step 2
Calculating the factor,thus:

$$\frac{1}{n} = \frac{1}{1} = 1$$

Calculating the current estimate State:

$$\hat{x}_{1,1} = \hat{x}_{1,0} + \frac{1}{1}(z_1 - \hat{x}_{1,0}) = 1000 + 1(1030 - 1000) = 1030g$$

Step 3
The dynamic model of the system is static; thus, the weight of the gold bar is not supposed to change. Our next state prediction equals to current state estimate:

$$\hat{x}_{2,1} = \hat{x}_{1,1} = 1030g$$

3. Second Iteration

After a unit time delay, the predicted estimate from the previous iteration becomes the previous estimate in the current iteration:

$$\hat{x}_{2,1} = 1030g$$

Step 1
Making the second measurement of the weight:

$$z_1 = 989g$$

Step 2
Calculating the factor,thus:

$$\frac{1}{n} = \frac{1}{2}$$

Calculating the current estimate State:

$$\hat{x}_{2,2} = \hat{x}_{2,1} + \frac{1}{2}(z_2 - \hat{x}_{2,1}) = 1030 + \frac{1}{2}(989 - 1030) = 1009.5g$$

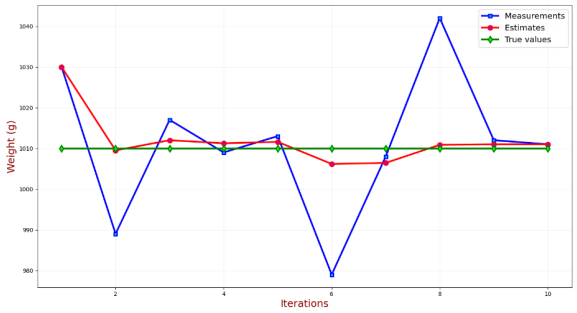
Step 3
The dynamic model of the system is static; thus, the weight of the gold bar is not supposed to change. Our next state prediction equals to current state estimate:

$$\hat{x}_{3,2} = \hat{x}_{2,2} = 1009.5g$$

We can stop here. The gain decreases with each measurement. Therefore, the contribution of each successive measurement is lower than the contribution of the previous measurement. We get pretty close to the true weight, which is 1010g. If we were making more measurements, we would get closer to the true value.

The following table summarizes our measurements and estimates, and the chart compares the measured values, the es-

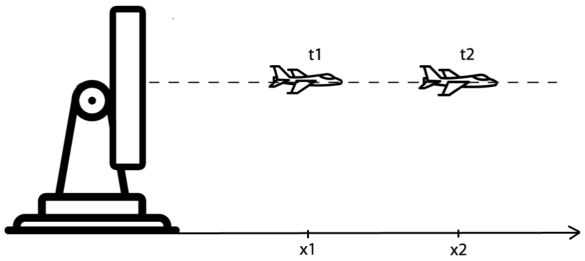
timates, and the true value.



III. Estimate The State of Aircraft

a. Tracking Constant Velocity Air in One Dimension

It is time to examine a dynamic system that changes its state over time.Let us assume a one-dimensional world. We assume an aircraft that is moving radially away from the radar. In the one-dimensional world, the angle to the radar is constant, and the aircraft's altitude is constant, as shown in the following figure.



x_n represents the range to the aircraft at time n. The aircraft velocity can be approximated using the range differentiation method - the change in the measured range with time. Thus, the velocity is a derivative of the range:

$$v = \frac{dx}{dt}$$

The radar sends a track beam in the direction of the target at a constant rate. The track-to-track interval is Δt. Two equations of motion can describe the system's dynamic model for constant velocity motion:

$$x_{n+1} = x_n + \Delta t * v$$

$$v_{n+1} = v_n$$

According to these equations, the aircraft range at the next track cycle equals the range at the current track cycle plus the target velocity multiplied by the track-to-track interval. Since we assume constant velocity in this example, the velocity at the next cycle equals the velocity at the current cycle. This system of equations extrapolates the current state to the next state.The Equations depend on the system dynamics and differ from example to example.

1. The $\alpha - \beta$ Filter

Let the radar track-to-track (Δt period be 5 seconds. Assume that at time $n-1$ the estimated range of the aircraft is 30,000m, and the estimated aircraft velocity is 40m/s.

Using the above Equations, we can predict the target position at time n :

$$\hat{x}_{n,n-1} = \hat{x}_{n-1,n-1} + \Delta t * \hat{v}_{n-1,n-1} = 30000 + 5 * 40 = 30200m$$

The target velocity prediction for time n:

$$\hat{v}_{n,n-1} = \hat{v}_{n-1,n-1} = 40m/s$$

However, at time n, the radar measures range (z_n) of 30,110m and not 30,200m as expected. There is a 90m gap between the predicted range and the measured range. There are two possible reasons for this gap:

- The radar measurements are not precise.
- The aircraft velocity has changed. The new aircraft velocity is: $\frac{30{,}110-30{,}000}{5}$ = 22m/s

Which of the two statements is true? Let us write down the Equation for the velocity:

$$\hat{v}_{n,n} = \hat{v}_{n,n-1} + \beta(\frac{z_n - \hat{x}_{n,n-1}}{\Delta t})$$

The value of the factor β depends on the precision level of the radar. Suppose that the 1σ precision of the radar is 20m. The 90 meters gap between the predicted and measured ranges most likely results from a change in the aircraft velocity. We should set the β factor to a high value in this case. If we set β = 0.9, then the estimated velocity would be:

$$\hat{v}_{n,n} = \hat{v}_{n,n-1} + \beta(\frac{z_n - \hat{x}_{n,n-1}}{\Delta t})$$

$$= 40 + 0.9(\frac{30{,}110 - 30{,}200}{5}) = 23.8m/s$$

On the other hand, suppose that the 1σ precision of the radar is 150m. Then the 90 meters gap probably results from the radar measurement error. We should set the β factor to a low value in this case. If we set β = 0.1, then the estimated velocity would be:

$$\hat{v}_{n,n} = \hat{v}_{n,n-1} + \beta(\frac{z_n - \hat{x}_{n,n-1}}{\Delta t})$$

$$= 40 + 0.1(\frac{30{,}110 - 30{,}200}{5}) = 38.2m/s$$

If the gap has been caused by measurement error, then the successive measurements would be in front or behind the predicted positions. Thus on average, the target velocity would not change.

The Equation for the aircraft position:

$$\hat{x}_{n,n} = \hat{x}_{n,n-1} + \alpha(z_n - \hat{x}_{n,n-1})$$

The magnitude of the α factor depends on the radar measurement precision. For high precision radar, we should choose high α, giving high weight to the measurements.
If α=1, then the estimated range equals the measured range:

$$\hat{x}_{n,n} = \hat{x}_{n,n-1} + 1(z_n - \hat{x}_{n,n-1}) = z_n$$

If α=0 , then the measurement has no meaning:

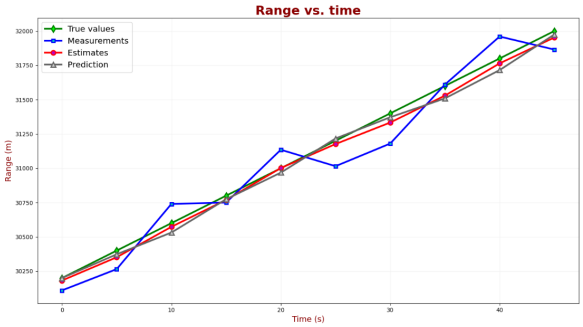
$$\hat{x}_{n,n} = \hat{x}_{n,n-1} + 0(z_n - \hat{x}_{n,n-1}) = \hat{x}_{n,n-1}$$

So, we have derived a system of equations that composes the Equation for the radar tracker.

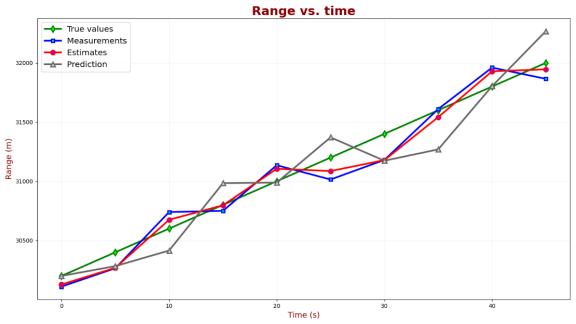
$$\hat{x}_{n,n} = \hat{x}_{n,n-1} + \alpha(z_n - \hat{x}_{n,n-1})$$

$$\hat{v}_{n,n} = \hat{v}_{n,n-1} + \beta(\frac{z_n - \hat{x}_{n,n-1}}{\Delta t})$$

The following chart depicts the true values, measured values, and estimates for $\alpha = 0.2$ and $\beta = 0.1$.



The following chart depicts the true values, measured values, and estimates for $\alpha = 0.8$ and $\beta = 0.5$.



The "smoothing" degree of this filter is much lower. The "current estimate" is very close to the measured values, and predicted estimate errors are pretty high.

So, shall we always choose low values for α and β ?

The answer is NO. The value of α and β should depend on the measurement precision. If we use high precision equipment, like laser radar, we would prefer a high α and β that follow measurements. In this case, the filter would quickly respond to a velocity change of the target. On the other hand, if measurement precision is low, we prefer low α and β. In this case, the filter smoothes the uncertainty (errors) in the measurements. However, the filter reaction to target velocity changes would be much slower.

b. Tracking Accelerating Air in One Dimension

1. The $\alpha - \beta - \gamma$ Filter

Thus, the State Extrapolation Equations become:

$$\hat{x}_{n,n-1} = \hat{x}_{n-1,n-1} + \hat{v}_{n-1,n-1}\Delta t + \hat{a}_{n-1,n-1}\frac{\Delta t^2}{2}$$

$$\hat{v}_{n,n-1} = \hat{v}_{n-1,n-1} + \hat{a}_{n-1,n-1}\Delta t$$

$$\hat{a}_{n,n-1} = \hat{a}_{n-1,n-1}$$

The State Update Equations become:

$$\hat{x}_{n,n} = \hat{x}_{n,n-1} + \alpha(z_n - \hat{x}_{n,n-1})$$

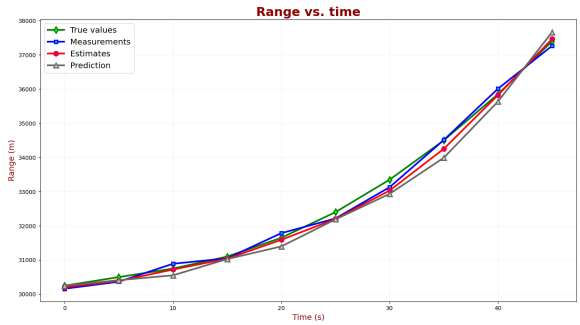
$$\hat{v}_{n,n} = \hat{v}_{n,n-1} + \beta(\frac{z_n - \hat{x}_{n,n-1}}{\Delta t})$$

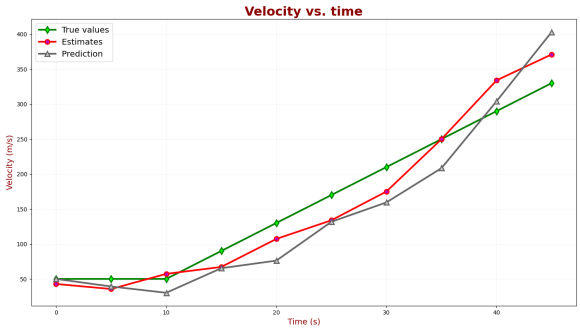
$$\hat{a}_{n,n} = \hat{a}_{n,n-1} + \gamma(\frac{z_n - \hat{x}_{n,n-1}}{0.5\Delta t^2})$$

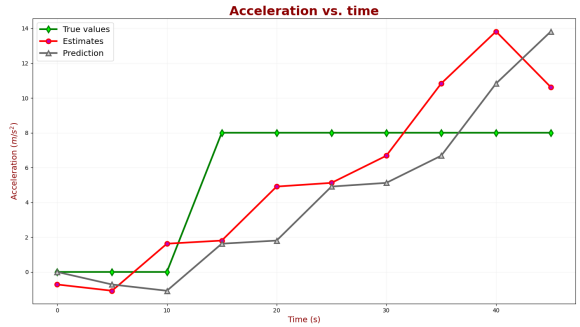
Let's take the scenario from the example: an aircraft that moves with a constant velocity of 50m/s for 15 seconds and then accelerates with a constant acceleration of 8m/s2 for another 35 seconds. The $\alpha - \beta - \gamma$ filter parameters are:

- $\alpha = 0.5$
- $\beta = 0.4$
- $\gamma = 0.1$

The track-to-track interval is 5 seconds.







As you can see, the $\alpha - \beta - \gamma$ filter with dynamic model equations that include acceleration can track the target with constant acceleration.

But what happens in the case of a maneuvering target? The target can suddenly change the flight direction by making a maneuver. The target's dynamic model can also include a changing acceleration. In such cases, the $\alpha - \beta - \gamma$ filter with constant $\alpha - \beta - \gamma$ coefficients produces estimation errors and, in some cases, loses the target track.

The Kalman filter can handle uncertainty in the dynamic model.

IV. KALMAN FILTER

a. State Extrapolation Equation

Until now, we've dealt with one dimensional processes, like estimating the liquid temperature. But many dynamic processes have two, three, or even more dimensions.

Airplane - No Control Input

For instance, the state vector that describes the airplane's position in space is one-dimensional. The state vector that describes the airplane position and velocity is two-dimensional. The state vector that describes the airplane position, velocity, and acceleration is three-dimensional.

$$\begin{bmatrix} x \\ \dot{x} \\ \ddot{x} \end{bmatrix}$$

Assuming a constant acceleration dynamic model, we can describe the extrapolated airplane state at time n by nine motion equations:

$$\begin{cases} \hat{x}_{n,n-1} = \hat{x}_{n-1,n-1} + \hat{\dot{x}}_{n-1,n-1}\Delta t + \frac{1}{2}\hat{\ddot{x}}_{n-1,n-1}\Delta t^2 \\ \hat{\dot{x}}_{n,n-1} = \hat{\dot{x}}_{n-1,n-1} + \hat{\ddot{x}}_{n-1,n-1}\Delta t \\ \hat{\ddot{x}}_{n,n-1} = \hat{\ddot{x}}_{n-1,n-1} \end{cases}$$

$$\begin{bmatrix} \hat{x}_{n,n-1} \\ \hat{\dot{x}}_{n,n-1} \\ \hat{\ddot{x}}_{n,n-1} \end{bmatrix} = \begin{bmatrix} 1 & \Delta t & 0.5\Delta t^2 \\ 0 & 1 & \Delta t \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \hat{x}_{n-1,n-1} \\ \hat{\dot{x}}_{n-1,n-1} \\ \hat{\ddot{x}}_{n-1,n-1} \end{bmatrix}$$

It is common practice to describe a multidimensional process with a single equation in matrix form.First, it is very exhausting to write all these equations; representing them in matrix notation is much shorter and more elegant.Second, computers are highly efficient at matrix calculations. Implementing the Kalman Filter in matrix form yields faster computation run time.

The following chapters describe the Kalman Filter equations in matrix form. And, of course, the theoretical part is followed by fully solved numerical examples.

The state extrapolation equation is:

$$\hat{x}_{n,n-1} = F\hat{x}_{n-1,n-1}$$

Airplane - With Control Input

This example is similar to the previous example, but now we have a sensor connected to the pilot's controls, so we have additional information about the airplane acceleration based on the pilot's commands.

$$\begin{bmatrix} x \\ \hat{\dot{x}} \end{bmatrix}$$

The control vector that describes the measured airplane acceleration in a cartesian coordinate system is:

$$\begin{bmatrix} \hat{\ddot{x}} \end{bmatrix}$$

$$\begin{cases} \hat{x}_{n,n-1} = \hat{x}_{n-1,n-1} + \hat{\dot{x}}_{n-1,n-1}\Delta t + \frac{1}{2}\ddot{x}_{n-1,n-1}\Delta t^2 \\ \hat{\dot{x}}_{n,n-1} = \hat{\dot{x}}_{n-1,n-1} + \ddot{x}_{n-1,n-1}\Delta t \end{cases}$$

$$\begin{bmatrix} \hat{x}_{n,n-1} \\ \hat{\dot{x}}_{n,n-1} \end{bmatrix} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \hat{x}_{n-1,n-1} \\ \hat{\dot{x}}_{n-1,n-1} \end{bmatrix} + \begin{bmatrix} \frac{1}{2}\Delta t^2 \\ \Delta t \end{bmatrix} \begin{bmatrix} \ddot{x}_{n-1,n-1} \end{bmatrix}$$

The state extrapolation equation is:

$$\hat{x}_{n,n-1} = F\hat{x}_{n-1,n-1} + Gu_{n-1,n-1}$$

Falling Object - With Control Input

Consider a free-falling object. The state vector includes the altitude h and the object's velocity v:

$$\begin{bmatrix} h \\ v \end{bmatrix}$$

The equation results in:

$$\begin{cases} \hat{h}_{n,n-1} = \hat{h}_{n-1,n-1} + \Delta t\hat{v}_{n-1,n-1} + \frac{1}{2}\Delta t^2 g \\ \hat{v}_{n,n-1} = \hat{v}_{n-1,n-1} + \Delta t g \end{cases}$$

Where g is the gravitational acceleration. We don't have a sensor that measures acceleration, but we know that for a falling object, acceleration equals g.The state extrapolation equation looks as follows:

$$\begin{bmatrix} \hat{h}_{n,n-1} \\ \hat{v}_{n,n-1} \end{bmatrix} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \hat{h}_{n-1,n-1} \\ \hat{v}_{n-1,n-1} \end{bmatrix} + \begin{bmatrix} 0.5\Delta t^2 \\ \Delta t \end{bmatrix} [g]$$

The state extrapolation equation is:

$$\hat{x}_{n,n-1} = F\hat{x}_{n-1,n-1} + Gu_{n-1,n-1}$$

Summary

Using the state extrapolation equation, we can predict the next system state based on the knowledge of the current state. It extrapolates the state vector from the present (time step $n-1$) to the future (time step n). Extrapolation is the method of predicting the value of the independent variable for a certain value of the dependent variable outside of the available data set using suitable mathematical tools and models.

$$\hat{x}_{n,n-1} = F\hat{x}_{n-1,n-1} + Gu_{n-1,n-1} + w_{n-1,n-1} \qquad (1)$$

The following table specifies the matrix dimensions of the state extrapolation equation variables:

TABLE 2: EXAMPLE NOTATION

variables	description
$\hat{x}_{n,n-1}$	is a predicted system state vector at time step n.
$\hat{x}_{n-1,n-1}$	is a estimated system state vector at time step $(n-1)$.
$u_{n-1,n-1}$	is a control variable at time $(n-1)$.
$w_{n-1,n-1}$	is a process noise $(n-1)$.
F	is a state transition matrix.
G	is a control matrix.

b. Covariance Extrapolation Equation

Assume vector x with k elements:

$$x = \begin{bmatrix} x_1 \\ x_2 \\ . \\ . \\ . \\ x_k \end{bmatrix}$$

The covariance matrix of the vector x is given by:

$$Cov(x) = E((x-\mu_x)(x-\mu_x)^T)$$

$$M = \begin{bmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1p} \\ a_{21} & a_{22} & a_{23} & \cdots & a_{2p} \\ a_{31} & a_{32} & a_{33} & \cdots & a_{3p} \\ \vdots & & & & \\ a_{n1} & a_{n2} & a_{n3} & \cdots & a_{np} \end{bmatrix}$$

$$\mu_1 = \frac{a_{11}+a_{21}+a_{31}+...+a_{n1}}{n}$$

$$\mu_2 = \frac{a_{12}+a_{22}+a_{32}+...+a_{n2}}{n}$$

$$\mu_3 = \frac{a_{13}+a_{23}+a_{33}+...+a_{n3}}{n}$$

$$\mu_p = \frac{a_{1p}+a_{2p}+a_{3p}+...+a_{np}}{n}$$

$$\mu = \begin{bmatrix} \mu_1 & \mu_2 & \mu_3 & \cdots & \mu_p \\ \mu_1 & \mu_2 & \mu_3 & \cdots & \mu_p \\ \mu_1 & \mu_2 & \mu_3 & \cdots & \mu_p \\ \vdots & & & & \\ \mu_1 & \mu_2 & \mu_3 & \cdots & \mu_p \end{bmatrix}$$

$$Cov = E((M-\mu)(M-\mu)^T)$$

$$Cov = \frac{(M-\mu)(M-\mu)^T}{n}$$

$$M = \begin{bmatrix} 1 & 2 & 3 \\ 3 & 1 & 1 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

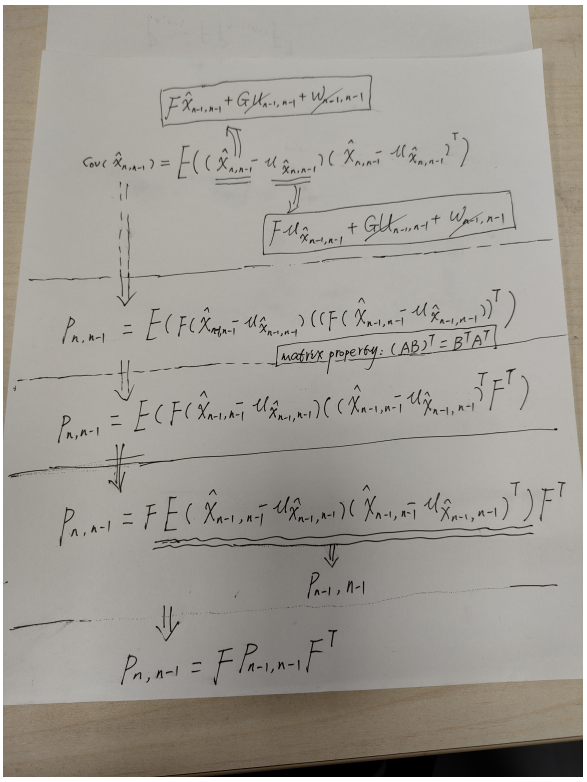
$$\mu_1 = \frac{1+3+4+7}{4} = \frac{15}{4} = 3.75$$

$$\mu_2 = \frac{2+1+5+8}{4} = \frac{16}{4} = 4$$

$$\mu_3 = \frac{3+1+6+9}{4} = \frac{19}{4} = 4.75$$

$$\mu = \begin{bmatrix} 3.75 & 4 & 4.75 \\ 3.75 & 4 & 4.75 \\ 3.75 & 4 & 4.75 \\ 3.75 & 4 & 4.75 \end{bmatrix}$$

$$\begin{cases} \hat{x}_{n,n-1} = F\hat{x}_{n-1,n-1} + Gu_{n-1,n-1} + w_{n-1,n-1} \\ Cov(\hat{x}_{n-1,n-1}) \\ = E((\hat{x}_{n-1,n-1} - \mu_{\hat{x}_{n-1,n-1}})(\hat{x}_{n-1,n-1} - \mu_{\hat{x}_{n-1,n-1}})^T) \end{cases}$$

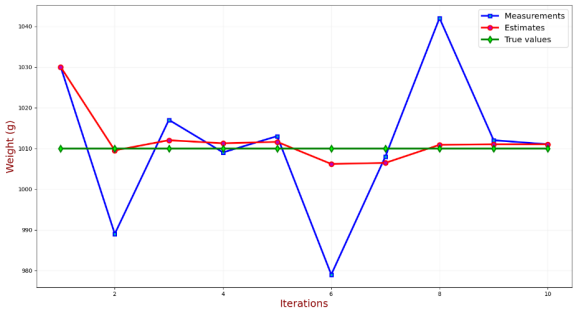


The Covariance Extrapolation Equation is

$$P_{n,n-1} = FP_{n-1,n-1}F^T + Q \tag{2}$$

c. Kalman Gain Equation

Let's recall our first example (gold bar weight measurement); We made multiple measurements and computed the estimate by averaging.We obtained the following result:



On the above plot, you can see the true values, the estimated values, and measurements vs. the number of measurements.

The differences between the measurements (blue samples) and the true values (green line) are measurement errors. Since the measurement errors are random, we can describe them by variance (σ^2). The variance of the measurement errors could be provided by the scale vendor or derived by a calibration procedure. The variance of the measurement errors is the measurement uncertainty.

We denote the measurement uncertainty by r.

The difference between the estimates (the red line) and the true values (the green line) is the estimate error. As you can see, the estimate error becomes smaller and smaller as we make additional measurements, and it converges towards

zero, while the estimated value converges towards the true value. We don't know the estimate error, but we can estimate the uncertainty in estimate.

We denote the estimate uncertainty by p. I would like to present the intuitive derivation of the Kalman Gain Equation – the third Kalman Filter equation. The mathematical derivation will be shown in the following chapters.

The Kalman Gain (denoted by K_n) is the weight given to the current state estimate and the measurements. The Kalman Gain is calculated dynamically for each filter iteration.

The Kalman Gain Equation is the following:

$$K_n = \frac{\text{Uncertainty in Prediction}}{\text{Uncertainty in Prediction + Uncertainty in Measure}}$$

The Kalman Gain is a number between zero and one: $0 <= K_n <= 1$. Let's rewrite the state update equation:

$$\hat{x}_{n,n} = \hat{x}_{n,n-1} + K_n(z_n - \hat{x}_{n,n-1}) = (1-K_n)\hat{x}_{n,n-1} + K_n z_n$$

As you can see, the Kalman Gain (K_n) is the measurement weight, and the $(1-K_n)$ term is the weight of the current state estimate.

When the measurement uncertainty is large and the estimate uncertainty is low, the Kalman Gain is close to zero. Hence we give significant weight to the estimate and a small weight to the measurement.

On the other hand, when the measurement uncertainty is low and the estimate uncertainty is large, the Kalman Gain is close to one. Hence we give a low weight to the estimate and a significant weight to the measurement.

If the measurement uncertainty equals the estimate uncertainty, then the Kalman gain equals 0.5. The Kalman gain tells how much the measurement changes the estimate.

The Kalman Filter is optimal since the Kalman Gain minimizes the estimation uncertainty.

Kalman Gain Mathematical Derivation

There are several ways to derive the one-dimensional Kalman Gain equation. I will present the simplest one.

Given the measurement z_n and the prior estimate $x_{n,n-1}$, we are interested to find an optimum combined estimate $x_{n,n}$ based on the measurement and the prior estimate.

The optimum combined estimate is actually a weighted mean of the prior estimate and the measurement:

$$\hat{x}_{n,n} = k_1 z_n + k_2 \hat{x}_{n,n-1}$$

Where k_1 and k_2 are the weights of the measurement and the prior estimate.

$$k_1 + k_2 = 1$$

$$\hat{x}_{n,n} = k_1 z_n + (1-k_1)\hat{x}_{n,n-1}$$

Since we are looking for an optimum estimate, we are interested to minimize $Var(\hat{x}_{n,n})$. The relation between variances is given by:

$$Var(\hat{x}_{n,n}) = Var(k_1 z_n + (1-k_1)\hat{x}_{n,n-1})$$
$$P_{n,n} = Var(k_1 z_n) + Var((1-k_1)\hat{x}_{n,n-1})$$
$$P_{n,n} = {k^2}_1 R_n + (1-k_1)^2 P_{n,n-1}$$

where:

- $P_{n,n}$ is the variance of optimum combined estimate $\hat{x}_{n,n}$.
- $P_{n,n-1}$ is the variance of the prior estimate $\hat{x}_{n,n-1}$.
- R_n is the variance of the measurement z_n.

Since we are looking for an optimum estimate, we are interested to minimize $p_{n,n}$.In order to find k_1 that minimizes $p_{n,n}$, we differentiate $p_{n,n}$ with respect to k_1 and set the result to zero.

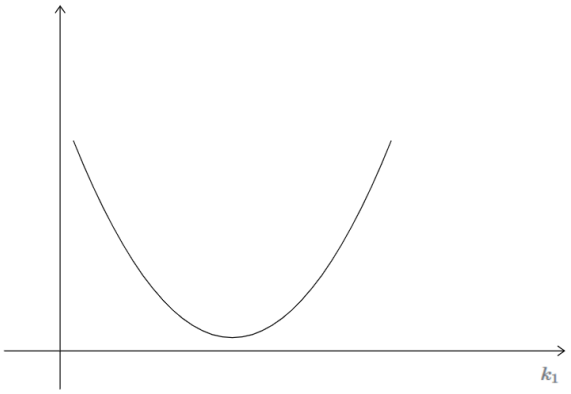
$$\frac{dP_{n,n}}{dk_1} = 2k_1 R_n - 2(1-k_1)P_{n,n-1}$$
$$k_1 R_n = P_{n,n-1} - k_1 P_{n,n-1}$$
$$k_1 P_{n,n-1} + k_1 R_n = P_{n,n-1}$$
$$k_1 = \frac{P_{n,n-1}}{P_{n,n-1}+R_n}$$

We have derived the Kalman Gain! Since the Kalman Gain yields the minimum variance estimate, the Kalman Filter is also called an optimal filter.

And we have other approach to derivd the Kalman Gain.

$$Var(\hat{x}_{n,n}) = Var(k_1 z_n + (1-k_1)\hat{x}_{n,n-1})$$
$$P_{n,n} = Var(k_1 z_n) + Var((1-k_1)\hat{x}_{n,n-1})$$
$$P_{n,n} = {k^2}_1 R_n + (1-k_1)^2 P_{n,n-1}$$

$$f(k_1) = {k^2}_1 R_n + (1-k_1)^2 P_{n,n-1} - P_{n,n}$$
$$f(k_1) = (R_n + P_{n,n-1}){k^2}_1 - 2k_1 P_{n,n-1} - P_{n,n} + P_{n,n-1}$$



Since we are looking for an optimum estimate, we are interested to minimize $f(k_1)$.we differentiate k_1 and set the result to zero.

$$\frac{df(k_1)}{dk_1} = 0$$

$$K_n = \frac{P_{n,n-1}}{P_{n,n-1}+R_n}$$

we denoted the measurement by z_n.

The measurement value represents a true system state in addition to the random measurement noise v_n, caused by the measurement device.

The measurement noise variance r_n can be constant for each measurement - for example, if we have scales that have a precision of 0.5kg (standard deviation). On the other hand, the measurement noise variance rn can be different for each

measurement - for example, if we have a thermometer that has precision of 0.5% (standard deviation). in the latter case, the noise variance depends on the measured temperature.

The generalized measurement equation in matrix form is given by:

$$z_n = Hx_n + v_n$$

Where:

- z_n is a measurement vector.
- x_n is a true system state (hidden state).
- v_n is a random noise vector.
- v_n is an observation matrix.

In many cases the measured value is not the desired system state. For example, a digital electric thermometer measures an electric current, while the system state is the temperature. There is a need for a transformation of the system state (input) to the measurement (output).

The purpose of the observation matrix H is to convert system state into outputs using linear transformations. The following chapters include examples of the observation matrix usage.

State Selection

Sometimes certain states are measured while others are not. For example, the first, third and fifth states of a five-dimensional state vector are measurable, while second and fourth states are not measurable:

$$z_n = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} + v_n = \begin{bmatrix} x_1 \\ x_3 \\ x_5 \end{bmatrix} + v_n$$

$$z_n = Hx_n + v_n$$

The measurement error is given by:

$$e_n = (z_n - H\hat{x}_{n,n-1})$$

The measurement uncertainty is given by:

$$r_n = E(e_n e_n^T)$$

The Kalman Gain Equation is the following:

$$K_n = \frac{P_{n,n-1}}{P_{n,n-1} + R_n} \tag{3}$$

d. State Update Equation

The State Update Equation in the matrix form is given by:

$$\hat{x}_{n,n} = \hat{x}_{n,n-1} + K_n(z_n - H\hat{x}_{n,n-1}) \tag{4}$$

You should pay attention on the dimensions. If, for instance, the state vector has 5 dimensions, while only 3 dimensions are measurable (the first, third and fifth states):

$$x_n = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix}$$

$$z_n = \begin{bmatrix} z_1 \\ z_2 \\ z_3 \end{bmatrix}$$

The observation matrix would be 3×5 matrix:

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

The innovation $(z_n - H\hat{x}_{n,n-1})$ yields:

$$\left(z_n - H\hat{x}_{n,n-\mathbf{1}}\right) = \begin{bmatrix} z_1 \\ z_3 \\ z_5 \end{bmatrix} - \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \hat{x}_1 \\ \hat{x}_2 \\ \hat{x}_3 \\ \hat{x}_4 \\ \hat{x}_5 \end{bmatrix} = \begin{bmatrix} (z_1 - \hat{x}_1) \\ (z_3 - \hat{x}_3) \\ (z_5 - \hat{x}_5) \end{bmatrix}$$

The Kalman Gain dimensions shall be 5 * 3.

e. Covariance Update Equation

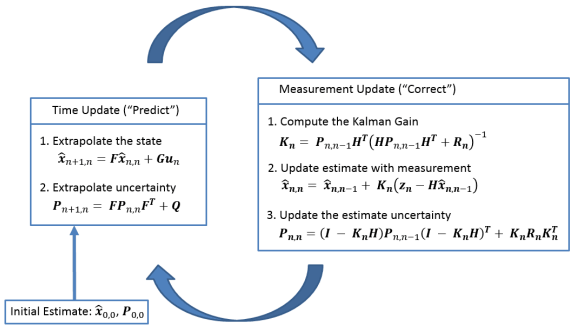
The Covariance Update Equation is given by:

$$P_{n,n} = (I - K_n H) P_{n,n-1} (I - K_n H)^T + K_n R_n K_n^T \tag{5}$$

Where:

- $P_{n,n}$ is the estimate uncertainty(covariance).
- $P_{n,n-1}$ is the prediction uncertainty(covariance).
- K_n is the Kalman Gain.
- H is the observation matrix.
- $R_{n,n-1}$ is the measurement uncertainty(covariance).
- I is an Identity matrix.

V. SUMMARY



VI. README

GitHub: https://github.com/pengfeinie

Home: https://pengfeinie.github.io/

Blog: https://blog.csdn.net/pfnie

REFERENCES

1. https://www.kalmanfilter.net/default.aspx