THE KALMAN FILTER < HTTPS://THEKALMANFILTER.COM/>

we help students master the Kalman Filter

Kalman Filter Explained Simply

Most tutorials for the Kalman Filter are difficult to understand because they require advanced math skills to understand how the Kalman Filter is derived. If you have tried to read Rudolf E Kalman's **1960 Kalman Filter paper < http://www.cs.unc.edu/~welch/kalman/kalmanPaper.html>** , you know how confusing this concept can be. But do you need to understand how to derive the Kalman Filter in order to use it?

No. If you want to design and implement a Kalman Filter, you do not need to know how to derive it, you just need to understand how it works.

The truth is, anybody can understand the Kalman Filter if it is explained in small digestible chunks. This post simply explains the Kalman Filter and how it works to estimate the state of a system.

The big picture of the Kalman Filter

Lets look at the Kalman Filter as a black box. The Kalman Filter has inputs and outputs. The inputs are noisy and sometimes inaccurate measurements. The outputs are less noisy and sometimes more accurate estimates. The estimates can be system state parameters that were not measured or observed. This last sentence describes the super power of the Kalman Filter. Again, the Kalman Filter estimates system parameters that are not observed or measured.

In short, you can think of the Kalman Filter as an algorithm that can estimate observable and unobservable parameters with great accuracy in real-time. Estimates with high accuracy are used to make precise predictions and decisions. For these reasons, Kalman Filters are used in robotics and real-time systems that need reliable information.

What is the Kalman Filter?

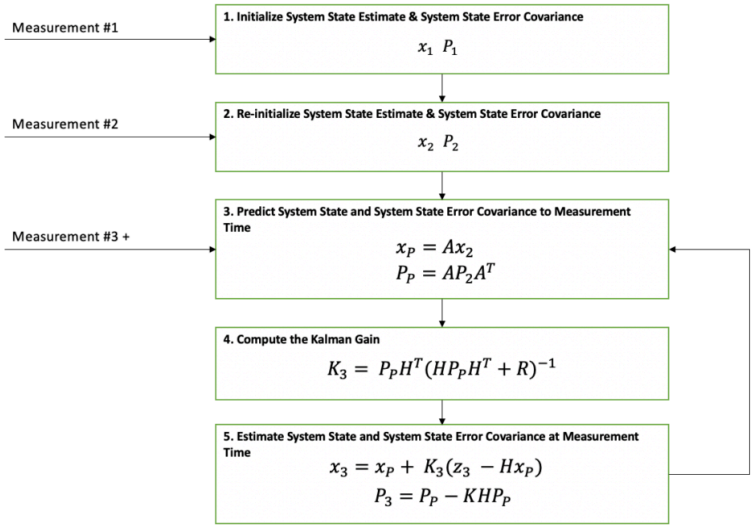
Simply put, the Kalman Filter is a generic algorithm that is used to estimate system parameters. It can use inaccurate or noisy measurements to estimate the state of that variable or another unobservable variable with greater accuracy. For example, Kalman Filtering is used to do the following:

- Object Tracking – Use the measured position of an object to more accurately estimate the position and velocity of that object.
- Body Weight Estimate on Digital Scale – Use the measured pressure on a surface to estimate the weight of object on that surface.

- Guidance, Navigation, and Control – Use Inertial Measurement Unit (IMU) sensors to estimate an objects location, velocity, and acceleration; and use those estimates to control the objects next moves.

The real power of the Kalman Filter is not smoothing measurements. It is the ability to estimate system parameters that can not be measured or observed with accuracy. Estimates with improved accuracy in systems that operate in real time, allow systems greater control and thus more capabilities.

Kalman Filter Algorithm Overview



The process diagram above shows the Kalman Filter algorithm step by step. I know those equations are intimidating but I assure you this will all make sense by the time you finish reading this article. Let's look at this process one step at a time. For your reference, here is a table of definitions that will be referred to throughout.

x	state variable	n x 1 column vector	Output
P	state covariance matrix	n x n matrix	Output
z	measurement	m x 1 column vector	Input
A	state transition matrix	n x n matrix	System Model
H	state-to-measurement matrix	m x n matrix	System Model
R	measurement covariance matrix	m x m matrix	Input
Q	process noise covariance matrix	n x n matrix	System Model
K	Kalman Gain	n x m	Internal

Kalman Filter Algorithm Reference Terms

The table above identifies the variables used in the algorithm. Each variable listed has a structure type and category. As this article continues, use the table as a reference.

If you are enjoying this post, check out my [book < https://thekalmanfilter.com/kalman-filter-made-easy-ebook>](https://thekalmanfilter.com/kalman-filter-made-easy-ebook) Kalman Filter Made Easy. You will learn: the first principles behind the Kalman Filter,

how to create simulations and perform analysis on Kalman Filters, and more.

Kalman Filter Radar Tracking Tutorial

This tutorial will go through the step by step process of a Kalman Filter being used to track airplanes and objects near airports. The output track states are used to display to the air traffic control operators monitoring the air space.

Kalman Filter Tutorial Notation

Radars are not built equally. Each one has different capabilities and therefore provides different types of information to its supporting systems. For this example, the radar will output its measurements in 2D cartesian coordinates, x and y. These measurements will be represented as a 2-by-1 column vector, **z**. The associated **variance-covariance matrix < https://thekalmanfilter.com/covariance-matrix-explained/>** for these measurements, **R**, will also be provided by the radar along with the time tag for when the measurement occurred, **t**. The subscript **m** denotes the measurement parameters. And the **k** subscript denotes the order of the measurement.

$$z_k = \begin{bmatrix} x_{m_k} \\ y_{m_k} \end{bmatrix} \qquad R_k = \begin{bmatrix} \sigma_{x_m}^2 & \sigma_{xy_m} \\ \sigma_{xy_m} & \sigma_{y_m}^2 \end{bmatrix} \qquad t_k = t_{m_k}$$

The Kalman Filter estimates the objects position and velocity based on the radar measurements. The estimate is represented by a 4-by-1 column vector, **x**.

It's associated **variance-covariance matrix < https://thekalmanfilter.com/covariance-matrix-explained/>** for the estimate is represented by a 4-by-4 matrix, **P**. Additionally, the state estimate has a time tag denoted as **T**.

$$x_k = \begin{bmatrix} x \\ y \\ \dot{x} \\ \dot{y} \end{bmatrix} \qquad P_k = \begin{bmatrix} \sigma_x^2 & \sigma_{xy} & \sigma_{x\dot{x}} & \sigma_{x\dot{y}} \\ \sigma_{xy} & \sigma_y^2 & \sigma_{y\dot{x}} & \sigma_{y\dot{y}} \\ \sigma_{x\dot{x}} & \sigma_{y\dot{x}} & \sigma_{\dot{x}}^2 & \sigma_{\dot{x}\dot{y}} \\ \sigma_{x\dot{y}} & \sigma_{y\dot{y}} & \sigma_{\dot{x}\dot{y}} & \sigma_{\dot{y}}^2 \end{bmatrix} \qquad T_k$$

Step 1: Initialize System State

Initializing the system state of a Kalman Filter varies across applications. In this tutorial, the Kalman Filter initializes the system state with the first measurement.

$\mathbf{x_k}$	Eqn. 1-1
$\mathbf{P_k}$	Eqn. 1-2

In this radar tracking example, the input measurements contain position only information. The output system state will contain the position and velocity of the object.

When the first measurement comes, the only information known about the object is the position at that point in time. The system state estimate will be set to the input position after the first estimate. The system state error covariance will be set to the first measurement's position accuracy.

Initialize System State in Equations

These equations show the input and output values for this Kalman Filter after receiving the first measurement.

$$z_1 = \begin{bmatrix} x_{m_1} \\ y_{m_1} \end{bmatrix} \qquad R_1 = \begin{bmatrix} \sigma_{x_m}^2 & \sigma_{xy_m} \\ \sigma_{xy_m} & \sigma_{y_m}^2 \end{bmatrix} \qquad t_1 = t_{m_1}$$

$$x_1 = \begin{bmatrix} x \\ y \\ \dot{x} \\ \dot{y} \end{bmatrix} = \begin{bmatrix} x_{m_1} \\ y_{m_1} \\ 0 \\ 0 \end{bmatrix} \qquad T_1 = t_{m_1}$$

$$P_1 = \begin{bmatrix} \sigma_x^2 & \sigma_{xy} & \sigma_{x\dot{x}} & \sigma_{x\dot{y}} \\ \sigma_{xy} & \sigma_y^2 & \sigma_{y\dot{x}} & \sigma_{y\dot{y}} \\ \sigma_{x\dot{x}} & \sigma_{y\dot{x}} & \sigma_{\dot{x}}^2 & \sigma_{\dot{x}\dot{y}} \\ \sigma_{x\dot{y}} & \sigma_{y\dot{y}} & \sigma_{\dot{x}\dot{y}} & \sigma_{\dot{y}}^2 \end{bmatrix} = \begin{bmatrix} \sigma_{x_m}^2 & \sigma_{xy_m} & 0 & 0 \\ \sigma_{xy_m} & \sigma_{y_m}^2 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Step 2: Reinitialize System State

The system state estimate is reinitialized because a velocity estimate needs a second position measurement for computation.

$\mathbf{x_k}$	Eqn. 2-1
$\mathbf{P_k}$	Eqn. 2-2

Velocity is estimated with a linear approximation. As you most likely recall from high school physics, velocity is equal to the distance traveled divided by the time it took to travel that distance.

The updated system state estimate will be the second measurement's position and the computed velocity.

The updated system state error covariance will be the second measurement's position accuracy and an approximated velocity accuracy. Note that this velocity accuracy approximation is something that can be tuned and adjusted after running data through your filter. There are different ways of approximating these values so if this doesn't match your approximation, that's okay!

Reinitialize System State in Equations for the Kalman Filter

These equations show the input and output values for this Kalman Filter after receiving the second measurement. Note the velocity variance terms in the state **covariance matrix < https://thekalmanfilter.com/covariance-matrix-explained/>** . These values are being set to 10^4. In other words, this value indicates a large uncertainty for the velocity state values. In this example, the velocity units are meters per second.

$$z_2 = \begin{bmatrix} x_{m_2} \\ y_{m_2} \end{bmatrix} \qquad R_2 = \begin{bmatrix} \sigma_{x_m}^2 & \sigma_{xy_m} \\ \sigma_{xy_m} & \sigma_{y_m}^2 \end{bmatrix} \qquad t_2 = t_{m_2}$$

$$x_2 = \begin{bmatrix} x \\ y \\ \dot{x} \\ \dot{y} \end{bmatrix} = \begin{bmatrix} x_{m_2} \\ y_{m_2} \\ \frac{x_{m_2} - x_{m_1}}{\Delta T} \\ \frac{y_{m_2} - y_{m_1}}{\Delta T} \end{bmatrix} \qquad T_2 = t_{m_2} \quad \Delta T = T_2 - T_1$$

$$P_2 = \begin{bmatrix} \sigma_x^2 & \sigma_{xy} & \sigma_{x\dot{x}} & \sigma_{x\dot{y}} \\ \sigma_{xy} & \sigma_y^2 & \sigma_{y\dot{x}} & \sigma_{y\dot{y}} \\ \sigma_{x\dot{x}} & \sigma_{y\dot{x}} & \sigma_{\dot{x}}^2 & \sigma_{\dot{x}\dot{y}} \\ \sigma_{x\dot{y}} & \sigma_{y\dot{y}} & \sigma_{\dot{x}\dot{y}} & \sigma_{\dot{y}}^2 \end{bmatrix} = \begin{bmatrix} \sigma_{x_m}^2 & \sigma_{xy_m} & 0 & 0 \\ \sigma_{xy_m} & \sigma_{y_m}^2 & 0 & 0 \\ 0 & 0 & 10^4 & 0 \\ 0 & 0 & 0 & 10^4 \end{bmatrix}$$

Quick Note on Initialization

Steps 1 and 2 used the first couple measurements to initialize and re-initialize the system estimate. Each application of the Kalman Filter may do this differently but the goal is to have a system state estimate that can be updated for future measurement with the Kalman Filter equations.

Steps 3 through 6 demonstrate how measurements are filtered in and the state estimate is updated.

Step 3: Predict System State Estimate

When the third measurement is received, the system state estimate is propagated forward to time align it with the measurement. This alignment is done so that the measurement and state estimate can be combined.

$\mathbf{x_p = Ax_{k-1}}$	Eqn. 3-1
$\mathbf{P_p = AP_{k-1}A^T + Q}$	Eqn. 3-2

The system model is used to perform this prediction. In this example, a constant velocity linear motion model is used to approximate the objects position change over a time interval. Note that a constant velocity model does assume zero acceleration. Remember this because it will resurface later.

The constant velocity linear motion model is something you may also remember from your high school physics class. The equation states that the position of an object is equal to its initial position

plus its displacement over a specified time period assuming a constant velocity.

A state transition matrix represents these equations. This matrix is used to propagate the state estimate and state error covariance matrix appropriately. You may be wondering why the state error covariance matrix is propagated. The reason for this is because when a state estimate is propagated in time, the uncertainty about its state at this future time step is inherently uncertain, so the error covariance grows.

On the Q Matrix

The Q matrix represents process noise for the system model. The system model is an approximation. Throughout the life of a system state, that system model fluctuates in its accuracy. Therefore, the Q matrix is used to represent this uncertainty and adds to the existing noise on the state. For this example, the systems actual accelerations and decelerations contribute to this error.

On the H Matrix

The Kalman Filter uses the state-to-measurement matrix, H, to convert the system state estimate from the state space to the measurement space. For some applications, this is a matrix of zeros and ones. For other applications that use the Extended Kalman Filter, the H matrix is populated with differential equations. To learn more about Extended Kalman Filters, check out my article on them **here <**

https://thekalmanfilter.com/extended-kalman-filter-python-example/> .

In this tutorial, the H matrix is a simple matrix that is set up to reduce the state estimate and error covariance to position only values rather than position and velocity.

Predict System State in Equations

$$z_3 = \begin{bmatrix} x_{m_3} \\ y_{m_3} \end{bmatrix} \quad R_2 = \begin{bmatrix} \sigma_{x_m}^2 & \sigma_{xy_m} \\ \sigma_{xy_m} & \sigma_{y_m}^2 \end{bmatrix} \quad t_3 = t_{m_3}$$

$$T_3 = t_{m_3} \quad \Delta T = T_3 - T_2$$

$$A = \begin{bmatrix} 1 & 0 & \Delta T & 0 \\ 0 & 1 & 0 & \Delta T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad Q = \begin{bmatrix} 10 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 \\ 0 & 0 & 25 & 0 \\ 0 & 0 & 0 & 25 \end{bmatrix}$$

$$x_p = Ax_2 = \begin{bmatrix} x + \dot{x}\Delta T \\ y + \dot{y}\Delta T \\ \dot{x} \\ \dot{y} \end{bmatrix}$$

$$P_P = AP_2A^T + Q = \begin{bmatrix} \sigma_x^2 + 10000 \cdot \Delta T^2 + 10 & \sigma_{xy} & 10000 \cdot \Delta T & 0 \\ \sigma_{xy} & \sigma_y^2 + 10000 \cdot \Delta T^2 + 10 & 0 & 10000 \cdot \Delta T \\ 10000 \cdot \Delta T & 0 & 10025 & 0 \\ 0 & 10000 \cdot \Delta T & 0 & 10025 \end{bmatrix}$$

Step 4: Compute the Kalman Gain

The Kalman Filter computes a Kalman Gain for each new measurement that determines how much the input measurement will influence the system state estimate. In other words, when a really noisy measurement comes in to update the system state, the Kalman Gain will trust its current state estimate more than this new inaccurate information.

This concept is the root of the Kalman Filter algorithm and why it works. It can recognize how to properly weight its current estimate and the new measurement information to form an optimal estimate.

$K = P_P H^T (HP_P H^T + R)^{-1}$	Eqn. 4-1

Step 5: Estimate System State and System State Error Covariance Matrix

The Kalman Filter uses the Kalman Gain to estimate the system state and error covariance matrix for the time of the input measurement. After the Kalman Gain is computed, it is used to weight the measurement appropriately in two computations.

The first computation is the new system state estimate. The second computation is the system state error covariance.

$x_k = x_p + K(z_k - Hx_p)$	Eqn. 5-1
$P_k = P_P - KHP_P$	Eqn. 5-2

Kalman Filter Estimation Equations

The state estimate computed above is the only state history the Kalman Filter retains. As a result, Kalman Filters can be implemented on machines with low memory restrictions.

If you enjoyed reading this post, check out my eBook < https://thekalmanfilter.com/kalman-

filter-made-easy-ebook/> Kalman Filter Made Easy and my Unscented Kalman Filter book < https://thekalmanfilter.com/unscented-kalman-filter-made-easy/> . You will learn: the first principles behind the Kalman Filter, how to create simulations and perform analysis on Kalman Filters, how the Extended Kalman Filter and Unscented Kalman Filter work, and more!

Next Steps

I hope this post allowed you to see how amazing the Kalman Filter is. And when it is broken up into parts, it is not that intimidating.

In conclusion, the Kalman Filter is a generic process for optimal state estimation. It is used in a variety of applications that require accurate estimation. So now that you know what it is and how it works, go out and use it in your projects!

If you enjoyed reading this post, please share it with your friends on your favorite social network! Thanks for reading!

19 comments

John

March 21, 2021 at 12:59 pm

As I beginner, I found this a very understandable explanation of the Filter. Thanks!

Rod

March 27, 2021 at 11:01 am

Appreciate the post. It really helped me better understand the Kalman filter and its applications when broken down into digestibles steps. Thank you!

Carl Thomsen

April 1, 2021 at 12:26 pm

I don’t see how you got the Pp matrix. My math came out different.

William Franklin < https://thekalmanfilter.com>

April 1, 2021 at 4:26 pm

Hi Carl, that was the symbolic representation of Pp. I replaced the equation so that it matches the example values. Thanks!

Tp

April 2, 2021 at 7:37 pm

Hello, thanks for the useful resource. Could be that Eqn. 5-2 has a “+” where it should show a “-“?

William Franklin <https://thekalmanfilter.com>

April 2, 2021 at 11:44 pm

Hi Tp, You are right, that should be a minus side. I updated the post to match. Thanks!

Flaviu

June 27, 2021 at 2:36 pm

Hello,
At the beginning you mentioned this example will output the position and velocity, yet when you arrive at the H matrix, you mention its reduced to position only.
Sorry for my bad English.
Can you elaborate?
I am really interested in how the output would be if it would include also the velocity.
Thanks

William Franklin <https://thekalmanfilter.com>

June 29, 2021 at 6:31 am

Hi Flaviu,

The H matrix is the state-to-measurement matrix. This matrix shows the relationship between the output state estimate and the input measurements. In this

example, and other applications, the output state estimate will contain variables that are not being measured or observed. In order to compare an input measurement with the state estimate for that time tag, we need to reduce the state to match the measurement structure. This is done to compute the Kalman Gain and the output state estimate. If you take another look, you can see that this matrix will not reduce the output state estimate to position only. The output for this example is still position and velocity.

Flaviu

July 3, 2021 at 10:22 am

Thank you for the explanation! It makes more sense now

rock-on

December 28, 2021 at 10:35 pm

The Reference Terms table really helps me understand the algorithm in matrix with very clear input and output. Thanks a lot ^^!

William Franklin <https://thekalmanfilter.com>

December 31, 2021 at 1:09 pm

Awesome! Glad to hear it helped!

James Oteng Danquah

January 7, 2022 at 9:40 am

can i be helped with how zk was calculated because i needed it to help me calculate for the update mean

William Franklin < https://thekalmanfilter.com>

January 7, 2022 at 10:00 am

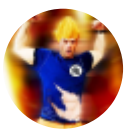
Hi James, you can see an example of computing measurements in my other **post with python code < https://thekalmanfilter.com/kalman-filter-python-example/>**

Otherwise, my ebook has additional examples of this as well. You can order the ebook **here < https://thekalmanfilter.com/kalman-filter-made-easy-ebook/>** .

prajjwal

February 26, 2022 at 6:17 am

Nice Work Man



Ryan

August 13, 2022 at 10:26 pm

Finally a clear explanation! Thank you!

Ayenew

December 13, 2023 at 9:48 pm

This is by far the best explanation I have ever seen on the subject. Eye opening for me, thanks!

Rathinam Chandramohan

December 25, 2023 at 9:09 am

Great Effort to simply explain Kalman filter with illustration. Thank you

Andy Wilson

February 12, 2024 at 4:18 pm

If Q and R are constant (as is usually the case), the gain quickly converges, such that the Kalman filter is just an exponential filter with a prediction step. For many people this is a lot easier to understand, and even matches how it is typically used, where Q and R are manually tuned until it “looks good” and never changed again. Moreover, there is just one gain to manually tune instead of multiple quantities Q and R.

James McCombe

February 12, 2024 at 4:47 pm

This article taken in combination with your article visualising the meaning of the co-variance matrix

really brings a lot of clarity to the Kalman filter application! Thank you for taking the time

---

THE KALMAN FILTER < HTTPS://THEKALMANFILTER.COM/>

Proudly powered by **WordPress < https://wordpress.org/>** .