



- [Main page](#)
- [Contents](#)
- [Featured content](#)
- [Current events](#)
- [Random article](#)
- [Donate to Wikipedia](#)

- Interaction
  - [Help](#)
  - [About Wikipedia](#)
  - [Community portal](#)
  - [Recent changes](#)
  - [Contact Wikipedia](#)

- [Toolbox](#)
- [Print/export](#)

- Languages

- [Català](#)
- [Deutsch](#)
- [Eesti](#)
- [Español](#)
- [Esperanto](#)

- [Français](#)
- [Italiano](#)
- [Lietuvių](#)
- [Nederlands](#)
- [Polski](#)
- [Português](#)
- [Русский](#)
- [Suomi](#)
- [Svenska](#)
- [Türkçe](#)
- [Українська](#)

Article [Talk](#)

Read

[Edit](#)



# Kalman filter

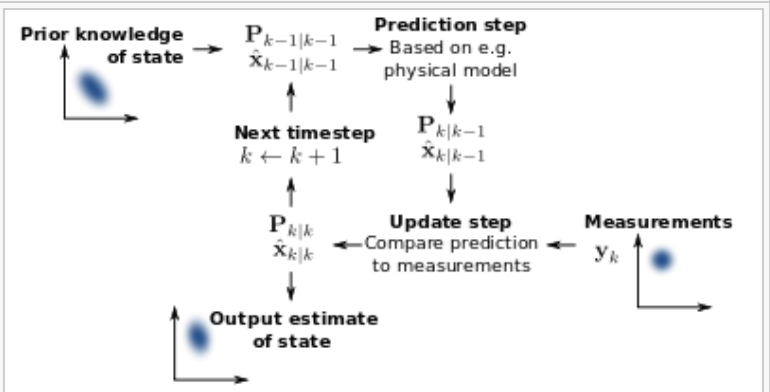
From Wikipedia, the free encyclopedia

The **Kalman filter**, also known as **linear quadratic estimation** (LQE), is an [algorithm](#) which uses a series of measurements observed over time, containing [noise](#) (random variations) and other inaccuracies, and produces estimates of unknown variables that tend to be more precise than those that would be based on a single measurement alone. More formally, the Kalman filter operates [recursively](#) on streams of noisy input data to produce a statistically optimal [estimate](#) of the underlying [system state](#). The filter is named for [Rudolf \(Rudy\) E. Kálmán](#), one of the primary developers of its theory.

The Kalman filter has numerous [applications](#) in technology. A common application is for [guidance, navigation and control](#) of vehicles, particularly aircraft and spacecraft.

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. Because of the algorithm's recursive nature, it can run in [real time](#) using only the present input measurements and the previously calculated state; no additional past information is required.

From a theoretical standpoint, the main assumption of the Kalman filter is that the underlying system is a [linear dynamical system](#) and that all error terms and measurements have a [Gaussian distribution](#) (often a [multivariate Gaussian distribution](#)). 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 [Bayesian model](#) similar to a [hidden Markov model](#) but where the state space of the [latent variables](#) is continuous and where all latent and observed variables have Gaussian distributions.



The Kalman filter keeps track of the estimated state of the system and the variance or uncertainty of the estimate. The estimate is updated using a state transition model and measurements.  $\hat{x}_{k|k-1}$  denotes the estimate of the system's state at time step  $k$  before the  $k$ -th measurement  $y_k$  has been taken into account;  $P_{k|k-1}$  is the corresponding uncertainty.

## Contents [hide]

- 1 Naming and historical development
- 2 Overview of the calculation
- 3 Example application
- 4 Technical description and context
- 5 Underlying dynamic system model
- 6 The Kalman filter
  - 6.1 Predict
  - 6.2 Update
  - 6.3 Invariants
  - 6.4 Estimation of the noise covariances  $Q_k$  and  $R_k$

6.5 Optimality and performance of the Kalman filter

7 Example application, technical

8 Derivations

8.1 Deriving the *a posteriori* estimate covariance matrix

8.2 Kalman gain derivation

8.3 Simplification of the *a posteriori* error covariance formula

9 Sensitivity analysis

10 Square root form

11 Relationship to recursive Bayesian estimation

12 Information filter

13 Fixed-lag smoother

14 Fixed-interval smoothers

14.1 Rauch–Tung–Striebel

14.2 Modified Bryson–Frazier smoother

15 Non-linear filters

15.1 Extended Kalman filter

15.2 Unscented Kalman filter

16 Kalman–Bucy filter

17 Hybrid Kalman filter

18 Kalman filter variants for the recovery of sparse signals

19 Applications

20 See also

21 References

22 Further reading

23 External links

## Naming and historical development

[\[edit\]](#)

The filter is named after Hungarian emigré [Rudolf E. Kálmán](#), though [Thorvald Nicolai Thiele](#)<sup>[1][2]</sup> and [Peter Swerling](#) developed a similar algorithm earlier. Richard S. Bucy of the [University of Southern California](#) contributed to the theory, leading to it often being called the Kalman-Bucy filter. [Stanley F. Schmidt](#) is generally credited with developing the first implementation of a Kalman filter. It was during a visit by Kalman to the [NASA Ames Research Center](#) that he saw the applicability of his ideas to the problem of trajectory estimation for the [Apollo program](#), leading to its incorporation in the Apollo navigation computer. This Kalman filter was first described and partially developed in technical papers by Swerling (1958), Kalman (1960) and Kalman and Bucy (1961).

Kalman filters have been vital in the implementation of the navigation systems of [U.S. Navy](#) nuclear [ballistic missile submarines](#), and in the guidance and navigation systems of cruise missiles such as the U.S. Navy's [Tomahawk missile](#) and the [U.S. Air Force's Air Launched Cruise Missile](#). It is also used in the guidance and navigation systems of the [NASA Space Shuttle](#) and the [attitude control](#) and navigation systems of the [International Space Station](#).

This digital filter is sometimes called the *Stratonovich–Kalman–Bucy filter* because it is a special case of a more general, non-linear filter developed somewhat earlier by the Soviet [mathematician Ruslan L. Stratonovich](#).<sup>[3][4][5][6]</sup> In fact, some of the special case linear filter's equations appeared in these papers by Stratonovich that were published before summer 1960, when Kalman met with Stratonovich during a conference in Moscow.

## Overview of the calculation

[\[edit\]](#)

The Kalman filter uses a system's dynamics model (e.g., physical laws of motion), known control inputs to that system, and multiple sequential measurements (such as from sensors) to form an estimate of the system's varying quantities (its [state](#)) that is better than the estimate obtained by using any one measurement alone. As such, it is a common [sensor fusion](#) and [data fusion](#) algorithm.

All measurements and calculations based on models are estimates to some degree. Noisy sensor data,

approximations in the equations that describe how a system changes, and external factors that are not accounted for introduce some uncertainty about the inferred values for a system's state. The Kalman filter averages a prediction of a system's state with a new measurement using a [weighted average](#). The purpose of the weights is that values with better (i.e., smaller) estimated uncertainty are "trusted" more. The weights are calculated from the [covariance](#), a measure of the estimated uncertainty of the prediction of the system's state. The result of the weighted average is a new state estimate that lies in between the predicted and measured state, and has a better estimated uncertainty than either alone. This process is repeated every time step, with the new estimate and its covariance informing the prediction used in the following iteration. This means that the Kalman filter works [recursively](#) and requires only the last "best guess" - not the entire history - of a system's state to calculate a new state.

Because the certainty of the measurements is often difficult to measure precisely, it is common to discuss the filter's behavior in terms of gain. The Kalman gain is a function of the relative certainty of the measurements and current state estimate, and can be "tuned" to achieve particular performance. With a high gain, the filter places more weight on the measurements, and thus follows them more closely. With a low gain, the filter follows the model predictions more closely, smoothing out noise but decreasing the responsiveness. At the extremes, a gain of one causes the filter to ignore the state estimate entirely, while a gain of zero causes the measurements to be ignored.

When performing the actual calculations for the filter (as discussed below), the state estimate and covariances are coded into [matrices](#) to handle the multiple dimensions involved in a single set of calculations. This allows for representation of linear relationships between different state variables (such as position, velocity, and acceleration) in any of the transition models or covariances.

## Example application

[\[edit\]](#)

As an example application, consider the problem of determining the precise location of a truck. The truck can be equipped with a [GPS](#) unit that provides an estimate of the position within a few meters. The GPS estimate is likely to be noisy; readings 'jump around' rapidly, though always remaining within a few meters of the real position. The truck's position can also be estimated by integrating its speed and direction over time, determined by keeping track of wheel revolutions and the angle of the steering wheel. This is a technique known as [dead reckoning](#). Typically, dead reckoning will provide a very smooth estimate of the truck's position, but it will [drift](#) over time as small errors accumulate. Additionally, the truck is expected to follow the laws of physics, so its position should be expected to change proportionally to its velocity.

In this example, the Kalman filter can be thought of as operating in two distinct phases: predict and update. In the prediction phase, the truck's old position will be modified according to the physical [laws of motion](#) (the dynamic or "state transition" model) plus any changes produced by the accelerator pedal and steering wheel. Not only will a new position estimate be calculated, but a new covariance will be calculated as well. Perhaps the covariance is proportional to the speed of the truck because we are more uncertain about the accuracy of the dead reckoning estimate at high speeds but very certain about the position when moving slowly. Next, in the update phase, a measurement of the truck's position is taken from the GPS unit. Along with this measurement comes some amount of uncertainty, and its covariance relative to that of the prediction from the previous phase determines how much the new measurement will affect the updated prediction. Ideally, if the dead reckoning estimates tend to drift away from the real position, the GPS measurement should pull the position estimate back towards the real position but not disturb it to the point of becoming rapidly changing and noisy.

## Technical description and context

[\[edit\]](#)

The Kalman filter is an efficient [recursive filter](#) that [estimates](#) the internal state of a [linear dynamic system](#) from a series of [noisy](#) measurements. It is used in a wide range of [engineering](#) and [econometric](#) applications from [radar](#) and [computer vision](#) to estimation of structural macroeconomic models,<sup>[7][8]</sup> and is an important topic in [control theory](#) and [control systems](#) engineering. Together with the [linear-quadratic regulator](#) (LQR), the Kalman filter solves the [linear-quadratic-Gaussian control](#) problem (LQG). The Kalman filter, the linear-quadratic regulator and the linear-quadratic-Gaussian controller are solutions to what probably are the most

fundamental problems in control theory.


In most applications, the internal state is much larger (more [degrees of freedom](#)) than the few "observable" parameters which are measured. However, by combining a series of measurements, the Kalman filter can estimate the entire internal state.

In [Dempster-Shafer theory](#), each state equation or observation is considered a special case of a [Linear belief function](#) and the Kalman filter is a special case of combining linear belief functions on a join-tree or Markov tree.

A wide variety of Kalman filters have now been developed, from Kalman's original formulation, now called the "simple" Kalman filter, the [Kalman-Bucy filter](#), Schmidt's "extended" filter, the [information filter](#), and a variety of "square-root" filters that were developed by Bierman, Thornton and many others. Perhaps the most commonly used type of very simple Kalman filter is the [phase-locked loop](#), which is now ubiquitous in radios, especially [frequency modulation](#) (FM) radios, television sets, [satellite communications](#) receivers, outer space communications systems, and nearly any other [electronic](#) communications equipment.

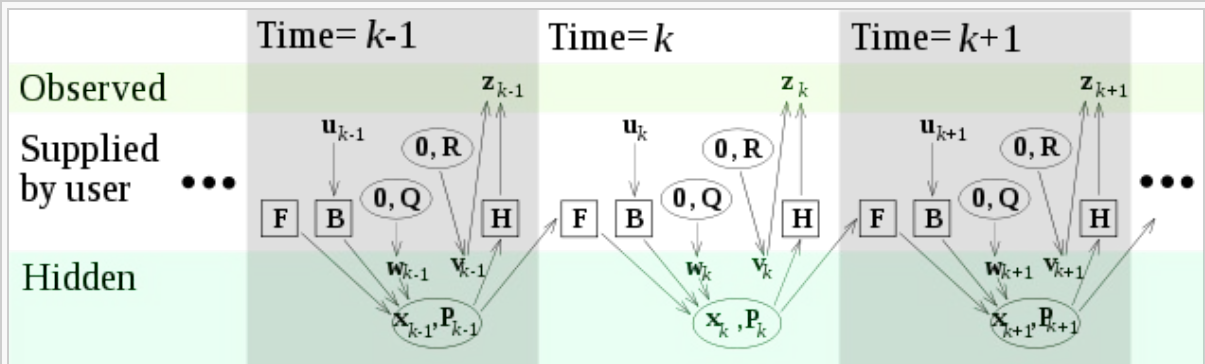
Underlying dynamic system model

[edit]

 This section requires [expansion](#).

Kalman filters are based on [linear dynamic systems](#) discretized in the time domain. They are modelled on a [Markov chain](#) built on [linear operators](#) perturbed by [Gaussian noise](#). The [state](#) of the system is represented as a [vector](#) of [real numbers](#). At each [discrete time](#) increment, a linear operator is applied to the state to generate the new state, with some noise mixed in, and optionally some information from the controls on the system if they are known. Then, another linear operator mixed with more noise generates the observed outputs from the true ("hidden") state. The Kalman filter may be regarded as analogous to the hidden Markov model, with the key difference that the hidden state variables take values in a continuous space (as opposed to a discrete state space as in the hidden Markov model). Additionally, the hidden Markov model can represent an arbitrary distribution for the next value of the state variables, in contrast to the Gaussian noise model that is used for the Kalman filter. There is a strong [duality](#) between the equations of the Kalman Filter and those of the hidden Markov model. A review of this and other models is given in Roweis and Ghahramani (1999)<sup>[9]</sup> and Hamilton (1994), Chapter 13.<sup>[10]</sup>

In order to use the Kalman filter to estimate the internal state of a process given only a sequence of noisy observations, one must model the process in accordance with the framework of the Kalman filter. This means specifying the following matrices:  $\mathbf{F}_k$ , the state-transition model;  $\mathbf{H}_k$ , the observation model;  $\mathbf{Q}_k$ , the covariance of the process noise;  $\mathbf{R}_k$ , the covariance of the observation noise; and sometimes  $\mathbf{B}_k$ , the control-input model, for each time-step,  $k$ , as described below.



Model underlying the Kalman filter. Squares represent matrices. Ellipses represent [multivariate normal distributions](#) (with the mean and covariance matrix enclosed). Unenclosed values are [vectors](#). In the simple case, the various matrices are constant with time, and thus the subscripts are dropped, but the Kalman filter allows any of them to change each time step.

The Kalman filter model assumes the true state at time  $k$  is evolved from the state at  $(k - 1)$  according to

$$\mathbf{x}_k = \mathbf{F}_k \mathbf{x}_{k-1} + \mathbf{B}_k \mathbf{u}_k + \mathbf{w}_k$$

where

- $\mathbf{F}_k$  is the state transition model which is applied to the previous state  $\mathbf{x}_{k-1}$ ;
- $\mathbf{B}_k$  is the control-input model which is applied to the control vector  $\mathbf{u}_k$ ;
- $\mathbf{w}_k$  is the process noise which is assumed to be drawn from a zero mean [multivariate normal distribution](#) with [covariance](#)  $\mathbf{Q}_k$ .

$$\mathbf{w}_k \sim N(0, \mathbf{Q}_k)$$

At time  $k$  an observation (or measurement)  $\mathbf{z}_k$  of the true state  $\mathbf{x}_k$  is made according to

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

where  $\mathbf{H}_k$  is the observation model which maps the true state space into the observed space and  $\mathbf{v}_k$  is the observation noise which is assumed to be zero mean Gaussian white noise with covariance  $\mathbf{R}_k$ .


$$\mathbf{v}_k \sim N(0, \mathbf{R}_k)$$

The initial state, and the noise vectors at each step  $\{\mathbf{x}_0, \mathbf{w}_1, \dots, \mathbf{w}_k, \mathbf{v}_1 \dots \mathbf{v}_k\}$  are all assumed to be mutually [independent](#).

Many real dynamical systems do not exactly fit this model. In fact, unmodelled dynamics can seriously degrade the filter performance, even when it was supposed to work with unknown stochastic signals as inputs. The reason for this is that the effect of unmodelled dynamics depends on the input, and, therefore, can bring the estimation algorithm to instability (it diverges). On the other hand, independent white noise signals will not make the algorithm diverge. The problem of separating between measurement noise and unmodelled dynamics is a difficult one and is treated in control theory under the framework of [robust control](#).

## The Kalman filter

[\[edit\]](#)



This section **needs additional citations for verification**. Please help [improve this article](#) by adding citations to [reliable sources](#). Unsourced material may be [challenged](#) and [removed](#). (December 2010)

The Kalman filter is a [recursive](#) estimator. This means that only the estimated state from the previous time step and the current measurement are needed to compute the estimate for the current state. In contrast to batch estimation techniques, no history of observations and/or estimates is required. In what follows, the notation  $\hat{\mathbf{x}}_{n|m}$  represents the estimate of  $\mathbf{x}$  at time  $n$  given observations up to, and including at time  $m$ .

The state of the filter is represented by two variables:

- $\hat{\mathbf{x}}_{k|k}$ , the [a posteriori](#) state estimate at time  $k$  given observations up to and including at time  $k$ ;
- $\mathbf{P}_{k|k}$ , the [a posteriori](#) error covariance matrix (a measure of the estimated [accuracy](#) of the state estimate).

The Kalman filter can be written as a single equation, however it is most often conceptualized as two distinct phases: "Predict" and "Update". The predict phase uses the state estimate from the previous timestep to produce an estimate of the state at the current timestep. This predicted state estimate is also known as the *a priori* state estimate because, although it is an estimate of the state at the current timestep, it does not include observation information from the current timestep. In the update phase, the current *a priori* prediction is combined with current observation information to refine the state estimate. This improved estimate is termed the *a posteriori* state estimate.

Typically, the two phases alternate, with the prediction advancing the state until the next scheduled observation, and the update incorporating the observation. However, this is not necessary; if an observation is unavailable for some reason, the update may be skipped and multiple prediction steps performed. Likewise, if multiple independent observations are available at the same time, multiple update steps may be performed (typically with different observation matrices  $\mathbf{H}_k$ ).

### Predict

[\[edit\]](#)

Predicted (*a priori*) state estimate  $\hat{\mathbf{x}}_{k|k-1} = \mathbf{F}_k \hat{\mathbf{x}}_{k-1|k-1} + \mathbf{B}_k \mathbf{u}_k$

Predicted (*a priori*) estimate covariance  $\mathbf{P}_{k|k-1} = \mathbf{F}_k \mathbf{P}_{k-1|k-1} \mathbf{F}_k^T + \mathbf{Q}_k$



Update

[edit]

Innovation or measurement residual	$\tilde{\mathbf{y}}_k = \mathbf{z}_k - \mathbf{H}_k \hat{\mathbf{x}}_{k k-1}$
Innovation (or residual) covariance	$\mathbf{S}_k = \mathbf{H}_k \mathbf{P}_{k k-1} \mathbf{H}_k^T + \mathbf{R}_k$
Optimal Kalman gain	$\mathbf{K}_k = \mathbf{P}_{k k-1} \mathbf{H}_k^T \mathbf{S}_k^{-1}$
Updated ( <i>a posteriori</i> ) state estimate	$\hat{\mathbf{x}}_{k k} = \hat{\mathbf{x}}_{k k-1} + \mathbf{K}_k \tilde{\mathbf{y}}_k$
Updated ( <i>a posteriori</i> ) estimate covariance	$\mathbf{P}_{k k} = (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k) \mathbf{P}_{k k-1}$

The formula for the updated estimate and covariance above is only valid for the optimal Kalman gain. Usage of other gain values require a more complex formula found in the [derivations](#) section.

Invariants

[edit]

If the model is accurate, and the values for  $\hat{\mathbf{x}}_{0|0}$  and  $\mathbf{P}_{0|0}$  accurately reflect the distribution of the initial state values, then the following invariants are preserved: (all estimates have a mean error of zero)

- $\mathbb{E}[\mathbf{x}_k - \hat{\mathbf{x}}_{k|k}] = \mathbb{E}[\mathbf{x}_k - \hat{\mathbf{x}}_{k|k-1}] = 0$
- $\mathbb{E}[\tilde{\mathbf{y}}_k] = 0$

where  $\mathbb{E}[\xi]$  is the [expected value](#) of  $\xi$ , and covariance matrices accurately reflect the covariance of estimates

- $\mathbf{P}_{k|k} = \text{cov}(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k})$
- $\mathbf{P}_{k|k-1} = \text{cov}(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k-1})$
- $\mathbf{S}_k = \text{cov}(\tilde{\mathbf{y}}_k)$

Estimation of the noise covariances  $\mathbf{Q}_k$  and  $\mathbf{R}_k$

[edit]

Practical implementation of the Kalman Filter is often difficult due to the inability in getting a good estimate of the noise covariance matrices  $\mathbf{Q}_k$  and  $\mathbf{R}_k$ . Extensive research has been done in this field to estimate these covariances from data. One of the more promising approaches to doing this is called the **Autocovariance Least-Squares (ALS)** technique that uses [autocovariances](#) of routine operating data to estimate the covariances.<sup>[11][12]</sup> The [GNU Octave](#) code used to calculate the noise covariance matrices using the **ALS** technique is available online under the [GNU General Public License](#) license.<sup>[13]</sup>

Optimality and performance of the Kalman filter

[edit]

It is known from the theory, that the Kalman filter is optimal in case that a) the model perfectly matches the real system, b) the entering noise is white and c) the covariances of the noise are exactly known. Several methods for the noise covariance estimation have been proposed during past decades. ALS is mentioned in the previous paragraph. After the covariances are identified, it is useful to evaluate the performance of the filter, i.e. whether it is possible to improve the state estimation quality or not. It is well known, that if the Kalman filter works optimally, the innovation sequence (the output prediction error) is a white noise. The whiteness property reflects the state estimation quality. For evaluation the filter performance it is necessary to inspect the whiteness property of the innovations. Several different methods can be used for this purpose. Three optimality tests with numerical examples are described in Matisko and Havlena (2012)<sup>[14]</sup>.

Example application, technical

[edit]

Consider a truck on perfectly frictionless, infinitely long straight rails. Initially the truck is stationary at position 0, but it is buffeted this way and that by random acceleration. We measure the position of the truck every  $\Delta t$  seconds, but these measurements are imprecise; we want to maintain a model of where the truck is and what its [velocity](#) is. We show here how we derive the model from which we create our Kalman filter.

Since **F**, **H**, **R** and **Q** are constant, their time indices are dropped.

The position and velocity of the truck are described by the linear state space

$$\mathbf{x}_k = \begin{bmatrix} x \\ \dot{x} \end{bmatrix}$$

where  $\dot{x}$  is the velocity, that is, the derivative of position with respect to time.

We assume that between the  $(k - 1)$  and  $k$  timestep the truck undergoes a constant acceleration of  $a_k$  that is [normally distributed](#), with mean 0 and standard deviation  $\sigma_a$ . From [Newton's laws of motion](#) we conclude that

$$\mathbf{x}_k = \mathbf{F}\mathbf{x}_{k-1} + \mathbf{G}a_k$$

(note that there is no  $\mathbf{B}u$  term since we have no known control inputs) where

$$\mathbf{F} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix}$$

and

$$\mathbf{G} = \begin{bmatrix} \frac{\Delta t^2}{2} \\ \Delta t \end{bmatrix}$$

so that

$$\mathbf{x}_k = \mathbf{F}\mathbf{x}_{k-1} + \mathbf{w}_k$$

where  $\mathbf{w}_k \sim N(0, \mathbf{Q})$  and

$$\mathbf{Q} = \mathbf{G}\mathbf{G}^T\sigma_a^2 = \begin{bmatrix} \frac{\Delta t^4}{4} & \frac{\Delta t^3}{2} \\ \frac{\Delta t^3}{2} & \Delta t^2 \end{bmatrix}\sigma_a^2.$$

At each time step, a noisy measurement of the true position of the truck is made. Let us suppose the measurement noise  $v_k$  is also normally distributed, with mean 0 and standard deviation  $\sigma_z$ .

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

where

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

and

$$\mathbf{R} = \mathbf{E}[\mathbf{v}_k\mathbf{v}_k^T] = \begin{bmatrix} \sigma_z^2 \end{bmatrix}$$

We know the initial starting state of the truck with perfect precision, so we initialize

$$\hat{\mathbf{x}}_{0|0} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

and to tell the filter that we know the exact position, we give it a zero covariance matrix:

$$\mathbf{P}_{0|0} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

If the initial position and velocity are not known perfectly the covariance matrix should be initialized with a suitably large number, say  $L$ , on its diagonal.

$$\mathbf{P}_{0|0} = \begin{bmatrix} L & 0 \\ 0 & L \end{bmatrix}$$

The filter will then prefer the information from the first measurements over the information already in the model.

## Derivations

[\[edit\]](#)

This section **needs additional citations for verification**. Please help [improve this article](#) by adding citations to [reliable sources](#). Unsourced



material may be [challenged](#) and [removed](#). (December 2010)

## Deriving the *a posteriori* estimate covariance matrix

[\[edit\]](#)

Starting with our invariant on the error covariance  $\mathbf{P}_{k|k}$  as above

$$\mathbf{P}_{k|k} = \text{cov}(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k})$$

substitute in the definition of  $\hat{\mathbf{x}}_{k|k}$

$$\mathbf{P}_{k|k} = \text{cov}(\mathbf{x}_k - (\hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k \tilde{\mathbf{y}}_k))$$

and substitute  $\tilde{\mathbf{y}}_k$

$$\mathbf{P}_{k|k} = \text{cov}(\mathbf{x}_k - (\hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k (\mathbf{z}_k - \mathbf{H}_k \hat{\mathbf{x}}_{k|k-1})))$$

and  $\mathbf{z}_k$

$$\mathbf{P}_{k|k} = \text{cov}(\mathbf{x}_k - (\hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k (\mathbf{H}_k \mathbf{x}_k + \mathbf{v}_k - \mathbf{H}_k \hat{\mathbf{x}}_{k|k-1})))$$

and by collecting the error vectors we get

$$\mathbf{P}_{k|k} = \text{cov}((\mathbf{I} - \mathbf{K}_k \mathbf{H}_k)(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k-1}) - \mathbf{K}_k \mathbf{v}_k)$$

Since the measurement error  $\mathbf{v}_k$  is uncorrelated with the other terms, this becomes

$$\mathbf{P}_{k|k} = \text{cov}((\mathbf{I} - \mathbf{K}_k \mathbf{H}_k)(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k-1})) + \text{cov}(\mathbf{K}_k \mathbf{v}_k)$$

by the properties of [vector covariance](#) this becomes

$$\mathbf{P}_{k|k} = (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k) \text{cov}(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k-1}) (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k)^T + \mathbf{K}_k \text{cov}(\mathbf{v}_k) \mathbf{K}_k^T$$

which, using our invariant on  $\mathbf{P}_{k|k-1}$  and the definition of  $\mathbf{R}_k$  becomes

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

This formula (sometimes known as the "**Joseph form**" of the covariance update equation) is valid for any value of  $\mathbf{K}_k$ . It turns out that if  $\mathbf{K}_k$  is the optimal Kalman gain, this can be simplified further as shown below.

## Kalman gain derivation

[\[edit\]](#)

The Kalman filter is a [minimum mean-square error](#) estimator. The error in the *a posteriori* state estimation is

$$\mathbf{x}_k - \hat{\mathbf{x}}_{k|k}$$

We seek to minimize the expected value of the square of the magnitude of this vector,  $E[|\mathbf{x}_k - \hat{\mathbf{x}}_{k|k}|^2]$ .

This is equivalent to minimizing the [trace](#) of the *a posteriori* estimate covariance matrix  $\mathbf{P}_{k|k}$ . By expanding out the terms in the equation above and collecting, we get:

$$\begin{aligned} \mathbf{P}_{k|k} &= \mathbf{P}_{k|k-1} - \mathbf{K}_k \mathbf{H}_k \mathbf{P}_{k|k-1} - \mathbf{P}_{k|k-1} \mathbf{H}_k^T \mathbf{K}_k^T + \mathbf{K}_k (\mathbf{H}_k \mathbf{P}_{k|k-1} \mathbf{H}_k^T + \mathbf{R}_k) \mathbf{K}_k^T \\ &= \mathbf{P}_{k|k-1} - \mathbf{K}_k \mathbf{H}_k \mathbf{P}_{k|k-1} - \mathbf{P}_{k|k-1} \mathbf{H}_k^T \mathbf{K}_k^T + \mathbf{K}_k \mathbf{S}_k \mathbf{K}_k^T \end{aligned}$$

The trace is minimized when the [matrix derivative](#) is zero:

$$\frac{\partial (\text{tr}(\mathbf{P}_{k|k}))}{\partial \mathbf{K}_k} = -2(\mathbf{H}_k \mathbf{P}_{k|k-1})^T + 2\mathbf{K}_k \mathbf{S}_k = 0$$

Solving this for  $\mathbf{K}_k$  yields the Kalman gain:

$$\begin{aligned} \mathbf{K}_k \mathbf{S}_k &= (\mathbf{H}_k \mathbf{P}_{k|k-1})^T = \mathbf{P}_{k|k-1} \mathbf{H}_k^T \\ \mathbf{K}_k &= \mathbf{P}_{k|k-1} \mathbf{H}_k^T \mathbf{S}_k^{-1} \end{aligned}$$

This gain, which is known as the *optimal Kalman gain*, is the one that yields [MMSE](#) estimates when used.

## Simplification of the *a posteriori* error covariance formula

[\[edit\]](#)

The formula used to calculate the *a posteriori* error covariance can be simplified when the Kalman gain



equals the optimal value derived above. Multiplying both sides of our Kalman gain formula on the right by  $\mathbf{S}_k \mathbf{K}_k^T$ , it follows that

$$\mathbf{K}_k \mathbf{S}_k \mathbf{K}_k^T = \mathbf{P}_{k|k-1} \mathbf{H}_k^T \mathbf{K}_k^T$$

Referring back to our expanded formula for the *a posteriori* error covariance,

$$\mathbf{P}_{k|k} = \mathbf{P}_{k|k-1} - \mathbf{K}_k \mathbf{H}_k \mathbf{P}_{k|k-1} - \mathbf{P}_{k|k-1} \mathbf{H}_k^T \mathbf{K}_k^T + \mathbf{K}_k \mathbf{S}_k \mathbf{K}_k^T$$

we find the last two terms cancel out, giving

$$\mathbf{P}_{k|k} = \mathbf{P}_{k|k-1} - \mathbf{K}_k \mathbf{H}_k \mathbf{P}_{k|k-1} = (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k) \mathbf{P}_{k|k-1}.$$

This formula is computationally cheaper and thus nearly always used in practice, but is only correct for the optimal gain. If arithmetic precision is unusually low causing problems with [numerical stability](#), or if a non-optimal Kalman gain is deliberately used, this simplification cannot be applied; the *a posteriori* error covariance formula as derived above must be used.

## Sensitivity analysis

[\[edit\]](#)



This section **needs additional citations for verification**. Please help [improve this article](#) by adding citations to [reliable sources](#). Unsourced material may be [challenged](#) and [removed](#). (December 2010)

The Kalman filtering equations provide an estimate of the state  $\hat{\mathbf{x}}_{k|k}$  and its error covariance  $\mathbf{P}_{k|k}$  recursively. The estimate and its quality depend on the system parameters and the noise statistics fed as inputs to the estimator. This section analyzes the effect of uncertainties in the statistical inputs to the filter.<sup>[15]</sup> In the absence of reliable statistics or the true values of noise covariance matrices  $\mathbf{Q}_k$  and  $\mathbf{R}_k$ , the expression

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

no longer provides the actual error covariance. In other words,

$\mathbf{P}_{k|k} \neq E[(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k})(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k})^T]$ . In most real time applications the covariance matrices that are used in designing the Kalman filter are different from the actual noise covariances matrices.<sup>[citation needed]</sup> This sensitivity analysis describes the behavior of the estimation error covariance when the noise covariances as well as the system matrices  $\mathbf{F}_k$  and  $\mathbf{H}_k$  that are fed as inputs to the filter are incorrect. Thus, the sensitivity analysis describes the robustness (or sensitivity) of the estimator to misspecified statistical and parametric inputs to the estimator.

This discussion is limited to the error sensitivity analysis for the case of statistical uncertainties. Here the actual noise covariances are denoted by  $\mathbf{Q}_k^a$  and  $\mathbf{R}_k^a$  respectively, whereas the design values used in the estimator are  $\mathbf{Q}_k$  and  $\mathbf{R}_k$  respectively. The actual error covariance is denoted by  $\mathbf{P}_{k|k}^a$  and  $\mathbf{P}_{k|k}$  as computed by the Kalman filter is referred to as the Riccati variable. When  $\mathbf{Q}_k \equiv \mathbf{Q}_k^a$  and  $\mathbf{R}_k \equiv \mathbf{R}_k^a$ , this means that  $\mathbf{P}_{k|k} = \mathbf{P}_{k|k}^a$ . While computing the actual error covariance using

$\mathbf{P}_{k|k}^a = E[(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k})(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k})^T]$ , substituting for  $\hat{\mathbf{x}}_{k|k}$  and using the fact that  $E[\mathbf{w}_k \mathbf{w}_k^T] = \mathbf{Q}_k^a$  and  $E[\mathbf{v}_k \mathbf{v}_k^T] = \mathbf{R}_k^a$ , results in the following recursive equations for  $\mathbf{P}_{k|k}^a$ :

$$\mathbf{P}_{k|k-1}^a = \mathbf{F}_k \mathbf{P}_{k-1|k-1}^a \mathbf{F}_k^T + \mathbf{Q}_k^a$$

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

- While computing  $\mathbf{P}_{k|k}$ , by design the filter implicitly assumes that  $E[\mathbf{w}_k \mathbf{w}_k^T] = \mathbf{Q}_k$  and  $E[\mathbf{v}_k \mathbf{v}_k^T] = \mathbf{R}_k$ .
- The recursive expressions for  $\mathbf{P}_{k|k}^a$  and  $\mathbf{P}_{k|k}$  are identical except for the presence of  $\mathbf{Q}_k^a$  and  $\mathbf{R}_k^a$  in place of the design values  $\mathbf{Q}_k$  and  $\mathbf{R}_k$  respectively.

## Square root form

[\[edit\]](#)

One problem with the Kalman filter is its [numerical stability](#). If the process noise covariance  $\mathbf{Q}_k$  is small, round-off error often causes a small positive eigenvalue to be computed as a negative number. This renders the numerical representation of the state covariance matrix  $\mathbf{P}$  [indefinite](#), while its true form is [positive-definite](#).

Positive definite matrices have the property that they have a [triangular matrix square root](#)  $\mathbf{P} = \mathbf{S} \cdot \mathbf{S}^T$ . This can be computed efficiently using the [Cholesky factorization](#) algorithm, but more importantly if the covariance is kept in this form, it can never have a negative diagonal or become asymmetric. An equivalent form, which avoids many of the [square root](#) operations required by the matrix square root yet preserves the desirable numerical properties, is the U-D decomposition form,  $\mathbf{P} = \mathbf{U} \cdot \mathbf{D} \cdot \mathbf{U}^T$ , where  $\mathbf{U}$  is a [unit triangular matrix](#) (with unit diagonal), and  $\mathbf{D}$  is a diagonal matrix.

Between the two, the U-D factorization uses the same amount of storage, and somewhat less computation, and is the most commonly used square root form. (Early literature on the relative efficiency is somewhat misleading, as it assumed that square roots were much more time-consuming than divisions,<sup>[16]:69</sup> while on 21-st century computers they are only slightly more expensive.)

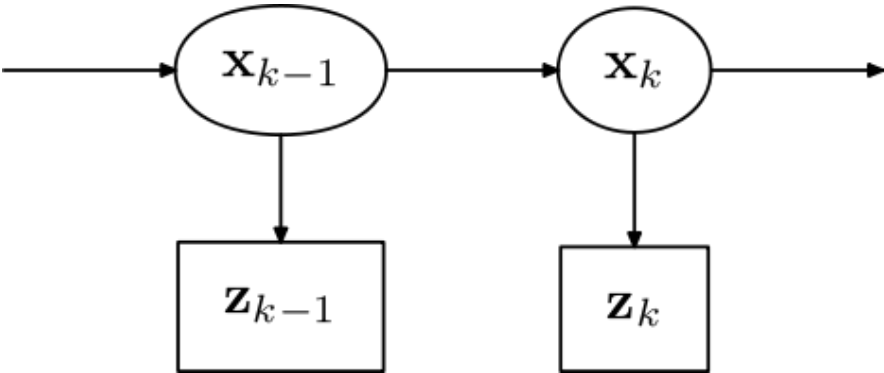
Efficient algorithms for the Kalman prediction and update steps in the square root form were developed by G. J. Bierman and C. L. Thornton.<sup>[16][17]</sup>

The [L·D·L<sup>T</sup> decomposition](#) of the innovation covariance matrix  $\mathbf{S}_k$  is the basis for another type of numerically efficient and robust square root filter.<sup>[18]</sup> The algorithm starts with the LU decomposition as implemented in the Linear Algebra PACKage ([LAPACK](#)). These results are further factored into the  $\mathbf{L} \cdot \mathbf{D} \cdot \mathbf{L}^T$  structure with methods given by Golub and Van Loan (algorithm 4.1.2) for a symmetric nonsingular matrix.<sup>[19]</sup> Any singular covariance matrix is [pivoted](#) so that the first diagonal partition is [nonsingular](#) and [well-conditioned](#). The pivoting algorithm must retain any portion of the innovation covariance matrix directly corresponding to observed state-variables  $\mathbf{H}_k \cdot \mathbf{x}_{k|k-1}$  that are associated with auxiliary observations in  $\mathbf{y}_k$ . The  $\mathbf{L} \cdot \mathbf{D} \cdot \mathbf{L}^T$  square-root filter requires [orthogonalization](#) of the observation vector.<sup>[17][18]</sup> This may be done with the inverse square-root of the covariance matrix for the auxiliary variables using Method 2 in Higham (2002, p. 263).<sup>[20]</sup>

### Relationship to recursive Bayesian estimation [\[edit\]](#)

The Kalman filter can be considered to be one of the most simple [dynamic Bayesian networks](#). The Kalman filter calculates estimates of the true values of measurements recursively over time using incoming measurements and a mathematical process model. Similarly, [recursive Bayesian estimation](#) calculates [estimates](#) of an unknown [probability density function](#) (PDF) recursively over time using incoming measurements and a mathematical process model.<sup>[21]</sup>

In recursive Bayesian estimation, the true state is assumed to be an unobserved [Markov process](#), and the measurements are the observed states of a hidden Markov model (HMM).



Because of the Markov assumption, the true state is conditionally independent of all earlier states given the immediately previous state.

$$p(\mathbf{x}_k | \mathbf{x}_0, \dots, \mathbf{x}_{k-1}) = p(\mathbf{x}_k | \mathbf{x}_{k-1})$$

Similarly the measurement at the  $k$ -th timestep is dependent only upon the current state and is conditionally independent of all other states given the current state.

$$p(\mathbf{z}_k | \mathbf{x}_0, \dots, \mathbf{x}_k) = p(\mathbf{z}_k | \mathbf{x}_k)$$

Using these assumptions the probability distribution over all states of the hidden Markov model can be written simply as:

$$p(\mathbf{x}_0, \dots, \mathbf{x}_k | \mathbf{z}_1, \dots, \mathbf{z}_k) = p(\mathbf{x}_0) \prod_{i=1}^k p(\mathbf{z}_i | \mathbf{x}_i) p(\mathbf{x}_i | \mathbf{x}_{i-1})$$

However, when the Kalman filter is used to estimate the state  $\mathbf{x}$ , the probability distribution of interest is that associated with the current states conditioned on the measurements up to the current timestep. This is achieved by marginalizing out the previous states and dividing by the probability of the measurement set.

This leads to the *predict* and *update* steps of the Kalman filter written probabilistically. The probability distribution associated with the predicted state is the sum (integral) of the products of the probability distribution associated with the transition from the  $(k - 1)$ -th timestep to the  $k$ -th and the probability distribution associated with the previous state, over all possible  $\mathbf{x}_{k-1}$ .

$$p(\mathbf{x}_k | \mathbf{Z}_{k-1}) = \int p(\mathbf{x}_k | \mathbf{x}_{k-1}) p(\mathbf{x}_{k-1} | \mathbf{Z}_{k-1}) d\mathbf{x}_{k-1}$$

The measurement set up to time  $t$  is

$$\mathbf{Z}_t = \{\mathbf{z}_1, \dots, \mathbf{z}_t\}$$

The probability distribution of the update is proportional to the product of the measurement likelihood and the predicted state.

$$p(\mathbf{x}_k | \mathbf{Z}_k) = \frac{p(\mathbf{z}_k | \mathbf{x}_k) p(\mathbf{x}_k | \mathbf{Z}_{k-1})}{p(\mathbf{z}_k | \mathbf{Z}_{k-1})}$$

The denominator

$$p(\mathbf{z}_k | \mathbf{Z}_{k-1}) = \int p(\mathbf{z}_k | \mathbf{x}_k) p(\mathbf{x}_k | \mathbf{Z}_{k-1}) d\mathbf{x}_k$$

is a normalization term.

The remaining probability density functions are

$$\begin{aligned} p(\mathbf{x}_k | \mathbf{x}_{k-1}) &= \mathcal{N}(\mathbf{F}_k \mathbf{x}_{k-1}, \mathbf{Q}_k) \\ p(\mathbf{z}_k | \mathbf{x}_k) &= \mathcal{N}(\mathbf{H}_k \mathbf{x}_k, \mathbf{R}_k) \\ p(\mathbf{x}_{k-1} | \mathbf{Z}_{k-1}) &= \mathcal{N}(\hat{\mathbf{x}}_{k-1}, \mathbf{P}_{k-1}) \end{aligned}$$

Note that the PDF at the previous timestep is inductively assumed to be the estimated state and covariance. This is justified because, as an optimal estimator, the Kalman filter makes best use of the measurements, therefore the PDF for  $\mathbf{x}_k$  given the measurements  $\mathbf{Z}_k$  is the Kalman filter estimate.

## Information filter [edit]

In the information filter, or inverse covariance filter, the estimated covariance and estimated state are replaced by the [information matrix](#) and [information](#) vector respectively. These are defined as:

$$\begin{aligned} \mathbf{Y}_{k|k} &= \mathbf{P}_{k|k}^{-1} \\ \hat{\mathbf{y}}_{k|k} &= \mathbf{P}_{k|k}^{-1} \hat{\mathbf{x}}_{k|k} \end{aligned}$$

Similarly the predicted covariance and state have equivalent information forms, defined as:

$$\begin{aligned} \mathbf{Y}_{k|k-1} &= \mathbf{P}_{k|k-1}^{-1} \\ \hat{\mathbf{y}}_{k|k-1} &= \mathbf{P}_{k|k-1}^{-1} \hat{\mathbf{x}}_{k|k-1} \end{aligned}$$

as have the measurement covariance and measurement vector, which are defined as:

$$\begin{aligned} \mathbf{I}_k &= \mathbf{H}_k^T \mathbf{R}_k^{-1} \mathbf{H}_k \\ \mathbf{i}_k &= \mathbf{H}_k^T \mathbf{R}_k^{-1} \mathbf{z}_k \end{aligned}$$

The information update now becomes a trivial sum.

$$\begin{aligned} \mathbf{Y}_{k|k} &= \mathbf{Y}_{k|k-1} + \mathbf{I}_k \\ \hat{\mathbf{y}}_{k|k} &= \hat{\mathbf{y}}_{k|k-1} + \mathbf{i}_k \end{aligned}$$

The main advantage of the information filter is that  $N$  measurements can be filtered at each timestep simply by summing their information matrices and vectors.

$$\begin{aligned} \mathbf{Y}_{k|k} &= \mathbf{Y}_{k|k-1} + \sum_{j=1}^N \mathbf{I}_{k,j} \\ \hat{\mathbf{y}}_{k|k} &= \hat{\mathbf{y}}_{k|k-1} + \sum_{j=1}^N \mathbf{i}_{k,j} \end{aligned}$$


To predict the information filter the information matrix and vector can be converted back to their state space equivalents, or alternatively the information space prediction can be used.

$$\begin{aligned} \mathbf{M}_k &= [\mathbf{F}_k^{-1}]^T \mathbf{Y}_{k-1|k-1} \mathbf{F}_k^{-1} \\ \mathbf{C}_k &= \mathbf{M}_k [\mathbf{M}_k + \mathbf{Q}_k^{-1}]^{-1} \\ \mathbf{L}_k &= \mathbf{I} - \mathbf{C}_k \\ \mathbf{Y}_{k|k-1} &= \mathbf{L}_k \mathbf{M}_k \mathbf{L}_k^T + \mathbf{C}_k \mathbf{Q}_k^{-1} \mathbf{C}_k^T \\ \hat{\mathbf{y}}_{k|k-1} &= \mathbf{L}_k [\mathbf{F}_k^{-1}]^T \hat{\mathbf{y}}_{k-1|k-1} \end{aligned}$$

Note that if  $F$  and  $Q$  are time invariant these values can be cached. Note also that  $F$  and  $Q$  need to be invertible.

### Fixed-lag smoother

[\[edit\]](#)



This section **needs additional citations for verification**. Please help [improve this article](#) by adding citations to [reliable sources](#). Unsourced material may be [challenged](#) and [removed](#). *(December 2010)*

The optimal fixed-lag smoother provides the optimal estimate of  $\hat{\mathbf{x}}_{k-N|k}$  for a given fixed-lag  $N$  using the measurements from  $\mathbf{Z}_1$  to  $\mathbf{Z}_k$ . It can be derived using the previous theory via an augmented state, and the main equation of the filter is the following:

$$\begin{bmatrix} \hat{\mathbf{x}}_{t|t} \\ \hat{\mathbf{x}}_{t-1|t} \\ \vdots \\ \hat{\mathbf{x}}_{t-N+1|t} \end{bmatrix} = \begin{bmatrix} \mathbf{I} \\ 0 \\ \vdots \\ 0 \end{bmatrix} \hat{\mathbf{x}}_{t|t-1} + \begin{bmatrix} 0 & \dots & 0 \\ \mathbf{I} & 0 & \vdots \\ \vdots & \ddots & \vdots \\ 0 & \dots & \mathbf{I} \end{bmatrix} \begin{bmatrix} \hat{\mathbf{x}}_{t-1|t-1} \\ \hat{\mathbf{x}}_{t-2|t-1} \\ \vdots \\ \hat{\mathbf{x}}_{t-N+1|t-1} \end{bmatrix} + \begin{bmatrix} K^{(0)} \\ K^{(1)} \\ \vdots \\ K^{(N-1)} \end{bmatrix} y_{t|t-1}$$

where:

- $\hat{\mathbf{x}}_{t|t-1}$  is estimated via a standard Kalman filter;
- $y_{t|t-1} = z(t) - H\hat{\mathbf{x}}_{t|t-1}$  is the innovation produced considering the estimate of the standard Kalman filter;
- the various  $\hat{\mathbf{x}}_{t-i|t}$  with  $i = 0, \dots, N$  are new variables, i.e. they do not appear in the standard Kalman filter;
- the gains are computed via the following scheme:

$$K^{(i)} = P^{(i)} H^T [H P H^T + R]^{-1}$$

and

$$P^{(i)} = P [F - K H]^T{}^i$$

where  $P$  and  $K$  are the prediction error covariance and the gains of the standard Kalman filter (i.e.,  $P_{t|t-1}$ ).

If the estimation error covariance is defined so that

$$P_i := E \left[ \left( \mathbf{x}_{t-i} - \hat{\mathbf{x}}_{t-i|t} \right)^* \left( \mathbf{x}_{t-i} - \hat{\mathbf{x}}_{t-i|t} \right) \mid z_1 \dots z_t \right],$$

then we have that the improvement on the estimation of  $\mathbf{x}_{t-i}$  is given by:

$$P - P_i = \sum_{j=0}^i \left[ P^{(j)} H^T \left[ H P H^T + R \right]^{-1} H \left( P^{(i)} \right)^T \right]$$

### Fixed-interval smoothers [edit]

The optimal fixed-interval smoother provides the optimal estimate of  $\hat{\mathbf{x}}_{k|n}$  ( $k < n$ ) using the measurements from a fixed interval  $\mathbf{z}_1$  to  $\mathbf{z}_n$ . This is also called "Kalman Smoothing". There are several smoothing algorithms in common use.

### Rauch–Tung–Striebel [edit]

The Rauch–Tung–Striebel (RTS) smoother<sup>[22]</sup> is an efficient two-pass algorithm for fixed interval smoothing. The main equations of the smoother are the following (assuming  $\mathbf{B}_k = \mathbf{0}$ ):

- forward pass: regular Kalman filter algorithm
- backward pass:

$\hat{\mathbf{x}}_{k|n} = \tilde{F}_k \hat{\mathbf{x}}_{k+1|n} + \tilde{K}_k \hat{\mathbf{x}}_{k+1|k}$ , where

- $\tilde{F}_k = \mathbf{F}_k^{-1} (\mathbf{I} - \mathbf{Q}_k \mathbf{P}_{k+1|k}^{-1})$
- $\tilde{K}_k = \mathbf{F}_k^{-1} \mathbf{Q}_k \mathbf{P}_{k+1|k}^{-1}$

### Modified Bryson–Frazier smoother [edit]

An alternative to the RTS algorithm is the modified Bryson–Frazier (MBF) fixed interval smoother developed by Bierman.<sup>[17]</sup> This also uses a backward pass that processes data saved from the Kalman filter forward pass. The equations for the backward pass involve the recursive computation of data which are used at each observation time to compute the smoothed state and covariance.

The recursive equations are

$$\begin{aligned} \tilde{\Lambda}_k &= \mathbf{H}_k^T \mathbf{S}_k^{-1} \mathbf{H}_k + \mathbf{C}_k^T \hat{\Lambda}_k \mathbf{C}_k \\ \hat{\Lambda}_{k-1} &= \mathbf{F}_k^T \tilde{\Lambda}_k \mathbf{F}_k \\ \hat{\Lambda}_n &= \mathbf{0} \\ \tilde{\lambda}_k &= -\mathbf{H}_k^T \mathbf{S}_k^{-1} \mathbf{z}_k + \mathbf{C}_k^T \hat{\lambda}_k \\ \hat{\lambda}_{k-1} &= \mathbf{F}_k^T \tilde{\lambda}_k \\ \hat{\lambda}_n &= \mathbf{0} \end{aligned}$$

where  $\mathbf{S}_k$  is the residual covariance and  $\mathbf{C}_k = \mathbf{I} - \mathbf{K}_k \mathbf{H}_k$ . The smoothed state and covariance can then be found by substitution in the equations

$$\begin{aligned} \mathbf{P}_{k|n} &= \mathbf{P}_{k|k} - \mathbf{P}_{k|k} \hat{\Lambda}_k \mathbf{P}_{k|k} \\ \mathbf{x}_{k|n} &= \mathbf{x}_{k|k} - \mathbf{P}_{k|k} \hat{\lambda}_k \end{aligned}$$

or

$$\begin{aligned} \mathbf{P}_{k|n} &= \mathbf{P}_{k|k-1} - \mathbf{P}_{k|k-1} \tilde{\Lambda}_k \mathbf{P}_{k|k-1} \\ \mathbf{x}_{k|n} &= \mathbf{x}_{k|k-1} - \mathbf{P}_{k|k-1} \tilde{\lambda}_k. \end{aligned}$$

An important advantage of the MBF is that it does not require finding the inverse of the covariance matrix.

### Non-linear filters [edit]

The basic Kalman filter is limited to a linear assumption. More complex systems, however, can be nonlinear. The non-linearity can be associated either with the process model or with the observation model or with both.

Extended Kalman filter

[edit]

Main article: *Extended Kalman filter*

In the extended Kalman filter (EKF), the state transition and observation models need not be linear functions of the state but may instead be non-linear functions. These functions are of [differentiable](#) type.

$$\begin{aligned}\mathbf{x}_k &= f(\mathbf{x}_{k-1}, \mathbf{u}_k) + \mathbf{w}_k \\ \mathbf{z}_k &= h(\mathbf{x}_k) + \mathbf{v}_k\end{aligned}$$

The function *f* can be used to compute the predicted state from the previous estimate and similarly the function *h* can be used to compute the predicted measurement from the predicted state. However, *f* and *h* cannot be applied to the covariance directly. Instead a matrix of partial derivatives (the [Jacobian](#)) is computed.

At each timestep the Jacobian is evaluated with current predicted states. These matrices can be used in the Kalman filter equations. This process essentially linearizes the non-linear function around the current estimate.

Unscented Kalman filter

[edit]

When the state transition and observation models – that is, the predict and update functions *f* and *h* (see above) – are highly non-linear, the extended Kalman filter can give particularly poor performance.<sup>[23]</sup> This is because the covariance is propagated through linearization of the underlying non-linear model. The unscented Kalman filter (UKF)<sup>[23]</sup> uses a deterministic sampling technique known as the [unscented transform](#) to pick a minimal set of sample points (called sigma points) around the mean. These sigma points are then propagated through the non-linear functions, from which the mean and covariance of the estimate are then recovered. The result is a filter which more accurately captures the true mean and covariance. (This can be verified using [Monte Carlo sampling](#) or through a [Taylor series](#) expansion of the posterior statistics.) In addition, this technique removes the requirement to explicitly calculate Jacobians, which for complex functions can be a difficult task in itself (i.e., requiring complicated derivatives if done analytically or being computationally costly if done numerically).

Predict

As with the EKF, the UKF prediction can be used independently from the UKF update, in combination with a linear (or indeed EKF) update, or vice versa.

The estimated state and covariance are augmented with the mean and covariance of the process noise.

$$\begin{aligned}\mathbf{x}_{k-1|k-1}^a &= [\hat{\mathbf{x}}_{k-1|k-1}^T \quad E[\mathbf{w}_k^T] ]^T \\ \mathbf{P}_{k-1|k-1}^a &= \begin{bmatrix} \mathbf{P}_{k-1|k-1} & 0 \\ 0 & \mathbf{Q}_k \end{bmatrix}\end{aligned}$$

A set of 2*L*+1 sigma points is derived from the augmented state and covariance where *L* is the dimension of the augmented state.

$$\begin{aligned}\chi_{k-1|k-1}^0 &= \mathbf{x}_{k-1|k-1}^a \\ \chi_{k-1|k-1}^i &= \mathbf{x}_{k-1|k-1}^a + \left( \sqrt{(L + \lambda) \mathbf{P}_{k-1|k-1}^a} \right)_i \quad i = 1..L \\ \chi_{k-1|k-1}^i &= \mathbf{x}_{k-1|k-1}^a - \left( \sqrt{(L + \lambda) \mathbf{P}_{k-1|k-1}^a} \right)_{i-L} \quad i = L + 1, \dots 2L\end{aligned}$$

where

$$\left( \sqrt{(L + \lambda) \mathbf{P}_{k-1|k-1}^a} \right)_i$$



is the  $i$ th column of the matrix square root of

$$(L + \lambda)\mathbf{P}_{k-1|k-1}^a$$

using the definition: square root  $A$  of matrix  $B$  satisfies

$$B \triangleq AA^T.$$

The matrix square root should be calculated using numerically efficient and stable methods such as the [Cholesky decomposition](#).

The sigma points are propagated through the transition function  $f$ .

$$\chi_{k|k-1}^i = f(\chi_{k-1|k-1}^i) \quad i = 0..2L$$

where  $f : R^L \rightarrow R^{|\mathbf{x}|}$ . The weighted sigma points are recombined to produce the predicted state and covariance.

$$\begin{aligned} \hat{\mathbf{x}}_{k|k-1} &= \sum_{i=0}^{2L} W_s^i \chi_{k|k-1}^i \\ \mathbf{P}_{k|k-1} &= \sum_{i=0}^{2L} W_c^i [\chi_{k|k-1}^i - \hat{\mathbf{x}}_{k|k-1}][\chi_{k|k-1}^i - \hat{\mathbf{x}}_{k|k-1}]^T \end{aligned}$$

where the weights for the state and covariance are given by:

$$\begin{aligned} W_s^0 &= \frac{\lambda}{L + \lambda} \\ W_c^0 &= \frac{\lambda}{L + \lambda} + (1 - \alpha^2 + \beta) \\ W_s^i &= W_c^i = \frac{1}{2(L + \lambda)} \\ \lambda &= \alpha^2(L + \kappa) \end{aligned}$$

$\alpha$  and  $\kappa$  control the spread of the sigma points.  $\beta$  is related to the distribution of  $\mathbf{x}$ . Normal values are  $\alpha = 10^{-3}, \kappa = 0$  and  $\beta = 2$ . If the true distribution of  $\mathbf{x}$  is Gaussian,  $\beta = 2$  is optimal.<sup>[24]</sup>

**Update**

The predicted state and covariance are augmented as before, except now with the mean and covariance of the measurement noise.

$$\begin{aligned} \mathbf{x}_{k|k-1}^a &= [\hat{\mathbf{x}}_{k|k-1}^T \quad E[\mathbf{v}_k^T]]^T \\ \mathbf{P}_{k|k-1}^a &= \begin{bmatrix} \mathbf{P}_{k|k-1} & 0 \\ 0 & \mathbf{R}_k \end{bmatrix} \end{aligned}$$

As before, a set of  $2L + 1$  sigma points is derived from the augmented state and covariance where  $L$  is the dimension of the augmented state.

$$\begin{aligned} \chi_{k|k-1}^0 &= \mathbf{x}_{k|k-1}^a \\ \chi_{k|k-1}^i &= \mathbf{x}_{k|k-1}^a + \left( \sqrt{(L + \lambda)\mathbf{P}_{k|k-1}^a} \right)_i \quad i = 1..L \\ \chi_{k|k-1}^i &= \mathbf{x}_{k|k-1}^a - \left( \sqrt{(L + \lambda)\mathbf{P}_{k|k-1}^a} \right)_{i-L} \quad i = L + 1, \dots, 2L \end{aligned}$$

Alternatively if the UKF prediction has been used the sigma points themselves can be augmented along the following lines

$$\chi_{k|k-1} := [\chi_{k|k-1}^T \quad E[\mathbf{v}_k^T]]^T \pm \sqrt{(L + \lambda)\mathbf{R}_k^a}$$

where

$$\mathbf{R}_k^a = \begin{bmatrix} 0 & 0 \\ 0 & \mathbf{R}_k \end{bmatrix}$$

The sigma points are projected through the observation function  $h$ .

$$\gamma_k^i = h(\chi_{k|k-1}^i) \quad i = 0..2L$$

The weighted sigma points are recombined to produce the predicted measurement and predicted measurement covariance.

$$\hat{\mathbf{z}}_k = \sum_{i=0}^{2L} W_s^i \gamma_k^i$$
$$\mathbf{P}_{z_k z_k} = \sum_{i=0}^{2L} W_c^i [\gamma_k^i - \hat{\mathbf{z}}_k][\gamma_k^i - \hat{\mathbf{z}}_k]^T$$

The state-measurement cross-covariance matrix,

$$\mathbf{P}_{x_k z_k} = \sum_{i=0}^{2L} W_c^i [\chi_{k|k-1}^i - \hat{\mathbf{x}}_{k|k-1}][\gamma_k^i - \hat{\mathbf{z}}_k]^T$$

is used to compute the UKF Kalman gain.

$$\mathbf{K}_k = \mathbf{P}_{x_k z_k} \mathbf{P}_{z_k z_k}^{-1}$$

As with the Kalman filter, the updated state is the predicted state plus the innovation weighted by the Kalman gain,

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

And the updated covariance is the predicted covariance, minus the predicted measurement covariance, weighted by the Kalman gain.

$$\mathbf{P}_{k|k} = \mathbf{P}_{k|k-1} - \mathbf{K}_k \mathbf{P}_{z_k z_k} \mathbf{K}_k^T$$

## Kalman–Bucy filter [\[edit\]](#)

The Kalman–Bucy filter (named after Richard Snowden Bucy) is a continuous time version of the Kalman filter.<sup>[25][26]</sup>

It is based on the state space model

$$\frac{d}{dt}\mathbf{x}(t) = \mathbf{F}(t)\mathbf{x}(t) + \mathbf{w}(t)$$
$$\mathbf{z}(t) = \mathbf{H}(t)\mathbf{x}(t) + \mathbf{v}(t)$$

where the covariances of the noise terms  $\mathbf{w}(t)$  and  $\mathbf{v}(t)$  are given by  $\mathbf{Q}(t)$  and  $\mathbf{R}(t)$ , respectively.

The filter consists of two differential equations, one for the state estimate and one for the covariance:

$$\frac{d}{dt}\hat{\mathbf{x}}(t) = \mathbf{F}(t)\hat{\mathbf{x}}(t) + \mathbf{K}(t)(\mathbf{z}(t) - \mathbf{H}(t)\hat{\mathbf{x}}(t))$$
$$\frac{d}{dt}\mathbf{P}(t) = \mathbf{F}(t)\mathbf{P}(t) + \mathbf{P}(t)\mathbf{F}^T(t) + \mathbf{Q}(t) - \mathbf{K}(t)\mathbf{R}(t)\mathbf{K}^T(t)$$

where the Kalman gain is given by

$$\mathbf{K}(t) = \mathbf{P}(t)\mathbf{H}^T(t)\mathbf{R}^{-1}(t)$$

Note that in this expression for  $\mathbf{K}(t)$  the covariance of the observation noise  $\mathbf{R}(t)$  represents at the same time the covariance of the prediction error (or *innovation*)  $\tilde{\mathbf{y}}(t) = \mathbf{z}(t) - \mathbf{H}(t)\hat{\mathbf{x}}(t)$ ; these covariances are equal only in the case of continuous time.<sup>[27]</sup>

The distinction between the prediction and update steps of discrete-time Kalman filtering does not exist in

continuous time.

The second differential equation, for the covariance, is an example of a [Riccati equation](#).

## Hybrid Kalman filter

[\[edit\]](#)

Most physical systems are represented as continuous-time models while discrete-time measurements are frequently taken for state estimation via a digital processor. Therefore, the system model and measurement model are given by

$$\begin{aligned}\dot{\mathbf{x}}(t) &= \mathbf{F}(t)\mathbf{x}(t) + \mathbf{B}(t)\mathbf{u}(t) + \mathbf{w}(t), & \mathbf{w}(t) &\sim N(\mathbf{0}, \mathbf{Q}(t)) \\ \mathbf{z}_k &= \mathbf{H}_k\mathbf{x}_k + \mathbf{v}_k, & \mathbf{v}_k &\sim N(\mathbf{0}, \mathbf{R}_k)\end{aligned}$$

where

$$\mathbf{x}_k = \mathbf{x}(t_k).$$

**Initialize**

$$\hat{\mathbf{x}}_{0|0} = E[\mathbf{x}(t_0)], \mathbf{P}_{0|0} = Var[\mathbf{x}(t_0)]$$

**Predict**

$$\begin{aligned}\dot{\hat{\mathbf{x}}}(t) &= \mathbf{F}(t)\hat{\mathbf{x}}(t) + \mathbf{B}(t)\mathbf{u}(t), \text{ with } \hat{\mathbf{x}}(t_{k-1}) = \hat{\mathbf{x}}_{k-1|k-1} \\ \Rightarrow \hat{\mathbf{x}}_{k|k-1} &= \hat{\mathbf{x}}(t_k)\end{aligned}$$

$$\begin{aligned}\dot{\mathbf{P}}(t) &= \mathbf{F}(t)\mathbf{P}(t) + \mathbf{P}(t)\mathbf{F}(t)^T + \mathbf{Q}(t), \text{ with } \mathbf{P}(t_{k-1}) = \mathbf{P}_{k-1|k-1} \\ \Rightarrow \mathbf{P}_{k|k-1} &= \mathbf{P}(t_k)\end{aligned}$$

The prediction equations are derived from those of continuous-time Kalman filter without update from measurements, i.e.,  $\mathbf{K}(t) = \mathbf{0}$ . The predicted state and covariance are calculated respectively by solving a set of differential equations with the initial value equal to the estimate at the previous step.

**Update**

$$\begin{aligned}\mathbf{K}_k &= \mathbf{P}_{k|k-1}\mathbf{H}_k^T(\mathbf{H}_k\mathbf{P}_{k|k-1}\mathbf{H}_k^T + \mathbf{R}_k)^{-1} \\ \hat{\mathbf{x}}_{k|k} &= \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k(\mathbf{z}_k - \mathbf{H}_k\hat{\mathbf{x}}_{k|k-1}) \\ \mathbf{P}_{k|k} &= (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}_{k|k-1}\end{aligned}$$

The update equations are identical to those of discrete-time Kalman filter.

## Kalman filter variants for the recovery of sparse signals

[\[edit\]](#)

Recently the traditional Kalman filter has been employed for the recovery of sparse, possibly dynamic, signals from noisy observations. Both works [\[28\]](#) and [\[29\]](#) utilize notions from the theory of [compressed sensing](#)/sampling, such as the restricted isometry property and related probabilistic recovery arguments, for sequentially estimating the sparse state in intrinsically low-dimensional systems.

## Applications

[\[edit\]](#)

- [Attitude and Heading Reference Systems](#)
- [Autopilot](#)
- [Battery state of charge \(SoC\) estimation](#) [\[1\]](#) [\[2\]](#) [\[3\]](#)
- [Brain-computer interface](#)
- [Chaotic signals](#)
- [Tracking of objects in computer vision](#)
- [Dynamic positioning](#)
- [Economics](#), in particular [macroeconomics](#), [time series](#), and [econometrics](#)
- [Inertial guidance system](#)
- [Orbit Determination](#)
- [Radar tracker](#)

- [Satellite navigation systems](#)
- [Seismology](#) [3] 
- [Simultaneous localization and mapping](#)
- [Speech enhancement](#)
- [Weather forecasting](#)
- [Navigation system](#)
- [3D modeling](#)
- [Structural health monitoring](#)








## See also

[\[edit\]](#)

- [Alpha beta filter](#)
- [Covariance intersection](#)
- [Data assimilation](#)
- [Ensemble Kalman filter](#)
- [Extended Kalman filter](#)
- [Invariant extended Kalman filter](#)
- [Fast Kalman filter](#)
- Compare with: [Wiener filter](#), and the multimodal [Particle filter](#) estimator.
- [Filtering problem \(stochastic processes\)](#)
- [Kernel adaptive filter](#)
- [Non-linear filter](#)
- [Predictor corrector](#)
- [Recursive least squares](#)
- [Sliding mode control](#) – describes a sliding mode observer that has similar noise performance to the Kalman filter
- [Separation principle](#)
- [Zakai equation](#)
- [Stochastic differential equations](#)
- [Volterra series](#)

## References

[\[edit\]](#)

- ↑  **Steffen L. Lauritzen** . "Time series analysis in 1880. A discussion of contributions made by T.N. Thiele". *International Statistical Review* **49**, 1981, 319-333.
- ↑  **Steffen L. Lauritzen**, *Thiele: Pioneer in Statistics* , Oxford University Press, 2002. ISBN 0-19-850972-3.
- ↑  **Stratonovich, R.L.** (1959). *Optimum nonlinear systems which bring about a separation of a signal with constant parameters from noise*. Radiofizika, 2:6, pp. 892–901.
- ↑  **Stratonovich, R.L.** (1959). *On the theory of optimal non-linear filtering of random functions*. Theory of Probability and its Applications, 4, pp. 223–225.
- ↑  **Stratonovich, R.L.** (1960) *Application of the Markov processes theory to optimal filtering*. Radio Engineering and Electronic Physics, 5:11, pp. 1–19.
- ↑  **Stratonovich, R.L.** (1960). *Conditional Markov Processes*. Theory of Probability and its Applications, 5, pp. 156–178.
- ↑  Ingvar Strid; Karl Walentin (April 2009). "Block Kalman Filtering for Large-Scale DSGE Models" . *Computational Economics* (Springer) **33** (3): 277–304. doi:10.1007/s10614-008-9160-4 
- ↑  Martin Møller Andreasen (2008). "Non-linear DSGE Models, The Central Difference Kalman Filter, and The Mean Shifted Particle Filter" 
- ↑  Roweis, S. and Ghahramani, Z., *A unifying review of linear Gaussian models* , Neural Comput. Vol. 11, No. 2, (February 1999), pp. 305–345.
- ↑  Hamilton, J. (1994), *Time Series Analysis*, Princeton University Press. Chapter 13, 'The Kalman Filter'.
- ↑  "Murali Rajamani, PhD Thesis"  Data-based Techniques to Improve State Estimation in Model Predictive Control, University of Wisconsin-Madison, October 2007
- ↑  Murali R. Rajamani and James B. Rawlings. Estimation of the disturbance structure from data using semidefinite programming and optimal weighting. Automatica, 45:142-148, 2009.

13. <sup>^</sup> <http://jbrwww.che.wisc.edu/software/als/>
14. <sup>^</sup> Matisko P. and V. Havlena (2012). Optimality tests and adaptive Kalman filter. Proceedings of IFAC System Identification Symposium, Belgium.
15. <sup>^</sup> Anderson, Brian D. O.; Moore, John B. (1979). *Optimal Filtering*. New York: [Prentice Hall](#). pp.129–133. ISBN 0-13-638122-7.
16. <sup>^</sup> <sup>a</sup> <sup>b</sup> Thornton, Catherine L. (15 October 1976). *Triangular Covariance Factorizations for Kalman Filtering* (PhD thesis). NASA. NASA Technical Memorandum 33-798
17. <sup>^</sup> <sup>a</sup> <sup>b</sup> <sup>c</sup> Bierman, G.J. (1977). *Factorization Methods for Discrete Sequential Estimation*. Academic Press
18. <sup>^</sup> <sup>a</sup> <sup>b</sup> Bar-Shalom, Yaakov; Li, X. Rong; Kirubarajan, Thiagalingam (July 2001). *Estimation with Applications to Tracking and Navigation*. New York: [John Wiley & Sons](#). pp.308–317. ISBN 978-0-471-41655-5.
19. <sup>^</sup> Golub, Gene H.; Van Loan, Charles F. (1996). *Matrix Computations*. Johns Hopkins Studies in the Mathematical Sciences (Third ed.). Baltimore, Maryland: [Johns Hopkins University](#). p.139. ISBN 978-0-8018-5414-9.
20. <sup>^</sup> Higham, Nicholas J. (2002). *Accuracy and Stability of Numerical Algorithms* (Second ed.). Philadelphia, PA: [Society for Industrial and Applied Mathematics](#). pp.680. ISBN 978-0-89871-521-7.
21. <sup>^</sup> [C. Johan Masreliez](#), R D Martin (1977); *Robust Bayesian estimation for the linear model and robustifying the Kalman filter* , IEEE Trans. Automatic Control
22. <sup>^</sup> Rauch, H.E.; Tung, F.; Striebel, C. T. (August 1965). "Maximum likelihood estimates of linear dynamic systems" . *AIAA J* **3** (8): 1445–1450. doi:10.2514/3.3166
23. <sup>^</sup> <sup>a</sup> <sup>b</sup> Julier, S.J.; Uhlmann, J.K. (1997). "A new extension of the Kalman filter to nonlinear systems" . *Int. Symp. Aerospace/Defense Sensing, Simul. and Controls* **3**. Retrieved 2008-05-03.
24. <sup>^</sup> Wan, Eric A. and van der Merwe, Rudolph "The Unscented Kalman Filter for Nonlinear Estimation"
25. <sup>^</sup> Bucy, R.S. and Joseph, P.D., *Filtering for Stochastic Processes with Applications to Guidance*, John Wiley & Sons, 1968; 2nd Edition, AMS Chelsea Publ., 2005. ISBN 0-8218-3782-6
26. <sup>^</sup> Jazwinski, Andrew H., *Stochastic processes and filtering theory*, Academic Press, New York, 1970. ISBN 0-12-381550-9
27. <sup>^</sup> Kailath, Thomas, "An innovation approach to least-squares estimation Part I: Linear filtering in additive white noise", *IEEE Transactions on Automatic Control*, 13(6), 646-655, 1968
28. <sup>^</sup> Carmi, A. and Gurfil, P. and Kanevsky, D. , "Methods for sparse signal recovery using Kalman filtering with embedded pseudo-measurement norms and quasi-norms", *IEEE Transactions on Signal Processing*, 58(4), 2405–2409, 2010
29. <sup>^</sup> Vaswani, N. , "Kalman Filtered Compressed Sensing", *15th International Conference on Image Processing*, 2008

## Further reading

[\[edit\]](#)

- Gelb, A. (1974). *Applied Optimal Estimation*. MIT Press.
- Kalman, R.E. (1960). "A new approach to linear filtering and prediction problems" . *Journal of Basic Engineering* **82** (1): 35–45. Retrieved 2008-05-03.
- Kalman, R.E.; Bucy, R.S. (1961). *New Results in Linear Filtering and Prediction Theory* . Retrieved 2008-05-03.
- Harvey, A.C. (1990). *Forecasting, Structural Time Series Models and the Kalman Filter*. Cambridge University Press.
- Roweis, S.; Ghahramani, Z. (1999). "A Unifying Review of Linear Gaussian Models". *Neural Computation* **11** (2): 305–345. doi:10.1162/089976699300016674 . PMID 9950734 .
- Simon, D. (2006). *Optimal State Estimation: Kalman, H Infinity, and Nonlinear Approaches* . Wiley-Interscience.
- Stengel, R.F. (1994). *Optimal Control and Estimation* . Dover Publications. ISBN 0-486-68200-5.
- Warwick, K. (1987). "Optimal observers for ARMA models" . *International Journal of Control* **46** (5): 1493–1503. doi:10.1080/00207178708933989 . Retrieved 2008-05-03.
- Bierman, G.J. (1977). *Factorization Methods for Discrete Sequential Estimation*. **128**. Mineola, N.Y.: Dover Publications. ISBN 978-0-486-44981-4.
- Bozic, S.M. (1994). *Digital and Kalman filtering*. Butterworth-Heinemann.
- Haykin, S. (2002). *Adaptive Filter Theory*. Prentice Hall.
- Liu, W.; Principe, J.C. and Haykin, S. (2010). *Kernel Adaptive Filtering: A Comprehensive Introduction*. John Wiley.
- Manolakis, D.G. (1999). *Statistical and Adaptive signal processing*. Artech House.
- Welch, Greg; Bishop, Gary (1997). *SCAAT: Incremental Tracking with Incomplete Information* . ACM Press/Addison-Wesley Publishing Co. pp. 333–344. doi:10.1145/258734.258876 . ISBN 0-89791-896-7.

- Jazwinski, Andrew H. (1970). *Stochastic Processes and Filtering*. Mathematics in Science and Engineering. New York: [Academic Press](#). pp.376. [ISBN 0-12-381550-9](#).
- Maybeck, Peter S. (1979). *Stochastic Models, Estimation, and Control*. Mathematics in Science and Engineering. **141-1**. New York: [Academic Press](#). pp.423. [ISBN 0-12-480701-1](#).
- Moriya, N. (2011). *Primer to Kalman Filtering: A Physicist Perspective*. New York: [Nova Science Publishers, Inc.](#) [ISBN 978-1-61668-311-5](#).
- Dunik, J.; Simandl M., Straka O. (2009). "Methods for estimating state and measurement noise covariance matrices: Aspects and comparisons.". *Proceedings of 15th IFAC Symposium on System Identification* (France): 372–377.
- Chui, Charles K.; Chen, Guanrong (2009). *Kalman Filtering with Real-Time Applications*. Springer Series in Information Sciences. **17** (4th ed.). New York: [Springer](#). pp.229. [ISBN 978-3-540-87848-3](#).
- Spivey, Ben; Hedengren, J. D. and Edgar, T. F. (2010). "Constrained Nonlinear Estimation for Industrial Process Fouling" [↗](#). *Industrial & Engineering Chemistry Research* **49** (17): 7824–7831. [doi:10.1021/ie9018116](#) [↗](#).
- Thomas Kailath, Ali H. Sayed, and Babak Hassibi, Linear Estimation, Prentice-Hall, NJ, 2000, [ISBN 978-0-13-022464-4](#).
- Ali H. Sayed, Adaptive Filters, Wiley, NJ, 2008, [ISBN 978-0-470-25388-5](#).

External links

[edit]

- [A New Approach to Linear Filtering and Prediction Problems](#) [↗](#), by R. E. Kalman, 1960
- [Kalman–Bucy Filter](#) [↗](#), a good derivation of the Kalman–Bucy Filter
- [MIT Video Lecture on the Kalman filter](#) [↗](#)
- [An Introduction to the Kalman Filter](#) [↗](#), SIGGRAPH 2001 Course, Greg Welch and Gary Bishop
- [Kalman filtering chapter](#) [↗](#) from *Stochastic Models, Estimation, and Control*, vol. 1, by Peter S. Maybeck
- [Kalman Filter](#) [↗](#) webpage, with lots of links
- [Kalman Filtering](#) [↗](#)
- [Kalman Filters](#), thorough introduction to several types, together with applications to *Robot Localization* [↗](#)
- [Kalman filters used in Weather models](#) [↗](#), SIAM News, Volume 36, Number 8, October 2003.
- [Critical Evaluation of Extended Kalman Filtering and Moving-Horizon Estimation](#) [↗](#), Ind. Eng. Chem. Res., 44 (8), 2451–2460, 2005.
- [Source code for the propeller microprocessor](#) [↗](#): Well documented source code written for the Parallax propeller processor.
- [Gerald J. Bierman's Estimation Subroutine Library](#) [↗](#): Corresponds to the code in the research monograph "Factorization Methods for Discrete Sequential Estimation" originally published by Academic Press in 1977. Republished by Dover.
- [Matlab Toolbox implementing parts of Gerald J. Bierman's Estimation Subroutine Library](#) [↗](#): UD / UDU' and LD / LDL' factorization with associated time and measurement updates making up the Kalman filter.
- [Matlab Toolbox of Kalman Filtering applied to Simultaneous Localization and Mapping](#) [↗](#): Vehicle moving in 1D, 2D and 3D
- [Derivation of a 6D EKF solution to Simultaneous Localization and Mapping](#) [↗](#) (In old version PDF [↗](#)). See also the tutorial on [implementing a Kalman Filter](#) [↗](#) with the [MRPT C++ libraries](#).
- [The Kalman Filter Explained](#) [↗](#) A very simple tutorial.
- [The Kalman Filter in Reproducing Kernel Hilbert Spaces](#) [↗](#) A comprehensive introduction.
- [Matlab code to estimate Cox–Ingersoll–Ross interest rate model with Kalman Filter](#) [↗](#): Corresponds to the paper "estimating and testing exponential-affine term structure models by kalman filter" published by Review of Quantitative Finance and Accounting in 1999.
- [Extended Kalman Filters](#) [↗](#) explained in the context of Simulation, Estimation, Control, and Optimization

Rate this page

What's this?

Trustworthy

★ ★ ★ ★ ★

Objective

★ ★ ★ ★ ★

Complete

★ ★ ★ ★ ★

Well-written

★ ★ ★ ★ ★

View page ratings



I am highly knowledgeable about this topic (optional)

Submit ratings

Categories: [Control theory](#) | [Nonlinear filters](#) | [Linear filters](#) | [Signal estimation](#) | [Stochastic differential equations](#) | [Robot control](#) | [Markov models](#) | [Hungarian inventions](#)

This page was last modified on 26 March 2012 at 18:57.

Text is available under the [Creative Commons Attribution-ShareAlike License](#); additional terms may apply. See [Terms of use](#) for details.  
Wikipedia® is a registered trademark of the [Wikimedia Foundation, Inc.](#), a non-profit organization.

[Contact us](#)

[Privacy policy](#) [About Wikipedia](#) [Disclaimers](#) [Mobile view](#)

