SIMULATIONS USING THE KALMAN FILTER

Vincent G. Vascimini

A Thesis

Submitted to the Graduate College of Bowling Green
State University in partial fulfillment of
the requirements for the degree of

MASTER OF ARTS

May 2020

Committee:

Kit Chan, Advisor

So-Hsiang Chou

ABSTRACT

Kit Chan, Advisor

Control and estimation theory are branches of mathematics that involve using data and measurements to estimate the value of a parameter of interest, and how changing certain parameters effects this estimation. The Kalman filter is a fundamental result in control and estimation theory that was introduced by Rudolf E. Kalman in 1960. The Kalman filter is a set of equations that provides an optimal estimate of the state of a system in a least-squares sense. The filter is often sought for its recursive and noise-smoothing properties, and has been found useful across many disciplines and in real world systems. This thesis will contribute to the literature of control and estimation theory by providing an introduction to the principles of the filter. This introduction includes a brief history of the filter, a derivation of the filter equations, and simple examples of applications of the filter.

ACKNOWLEDGMENTS

Foremost, I would like to express my sincere gratitude to Dr. Kit Chan for being an excellent advisor. He provided me with a mathematically stimulating environment and guided me through each stage of this thesis. I am grateful to thank my committee member Dr. So-Hsiang Chou for providing me with valuable comments and insights. Additionally, I would like to thank Dr. Tong Sun for his interest in my work and for his intuition. It is a pleasure to thank my colleagues for the continued encouragement and enthusiasm over the last two years. Finally, I am deeply indebted to my friends and family for their unparalleled love, help, and support.

TABLE OF CONTENTS

LIST OF FIGURES

Figure Page

LIST OF TABLES

Table Page

CHAPTER 1 INTRODUCTION

1.1 A Brief History

Control theory is a branch of mathematics that involves the study of how changing parameters effects the behavior of a system. Parameters are often manipulated so that the system may produce an optimal outcome of interest. Estimation theory is a branch of statistics that involves using measurements and data to estimate the value of a parameter of interest. The Kalman filter has roots in both control theory and estimation theory. The Kalman filter is an algorithm that uses data and measurements, which are often noisy and imprecise, to produce an optimal estimate of the system's current state; for details, one may refer to the link from [17]. In 1960 Rudolf Emil Kalman introduced the world to the Kalman filter, and since then it has been considered one of the most pivotal results in modern control and estimation theory.

Rudolf Emil Kalman was a Hungarian mathematician born in Budapest, Hungary on May 19, 1930 and died on July 2, 2016. He received his bachelor's degree in 1953 and his master's degree in 1954 at the Massachusetts Institute of Technology in Cambridge, Massachusetts, where he studied electrical engineering. In 1957 he received his doctorate degree from Columbia University in New York, New York while under the supervision of John R. Ragazzini [13].

In [13], Stepanov states that Kalman was involved in fundamental research in systems analysis and control theory as a student. At the beginning of his career, Kalman studied many different areas in addition to mathematics, including electrical engineering, mechanics, and operations research. Of course, Kalman is most well-known for his contributions to filtering and estimation theory.

Two names that are quintessential in the advancement of estimation theory are that of Carl Friedrich Gauss (1777-1855), a German mathematician, and Adrien-Marie Legendre (1752-1833), a French mathematician [13]. Gauss and Legendre were both involved in the development of the least-squares method, which is widely known as the first method for forming an optimal estimate from noisy data by minimizing the sum of the squares of the offsets of points on a curve [6].

A. N. Kolmogorov (1903-1986) and N. Weiner (1894-1964) are also predecessors of Kalman. Kolmogorov was a Soviet mathematician and one of the founders of modern probability theory. Weiner was an American mathematician who established the science of cybernetics, which he defined as the science of control and communications in the animal and machine [3]. Gauss and Legendre worked on the estimation problem of time-invariant vector, and Kolmogorov and Weiner worked on the problems of time variant parameter estimation [13].

The Weiner filter solves the estimation problem for continuous stationary random processes. Wiener solved this problem on the basis of factorization of rational spectral densities, and derived an estimation algorithm in the form of convolution of observations with the weight function which satisfied the integral equation of Wiener and Hopf, as stated in [13]. However, Kalman was unsatisfied with Wiener's solution and did not agree with his assumptions, definitions, and methodology. Kalman also noted the massive computational power required to run the Wiener filter and its suggested algorithms, and sought a more efficient and less restrictive algorithm. Through his efforts, Kalman realized that linear systems described by a transfer function matrix are equivalent to linear vector differential equations (which are completely controllable and observable); this conclusion is described further in [13]. For more historical accounts, one may refer to the 2011 article by [13].

Kalman's first paper ”A New Approach to Linear Filtering and Prediction Problems” was published in 1960; see [7]. In that paper Kalman suggested an algorithm to solve Wiener's filtering problem. Kalman was able to remove the stationary requirements of the Weiner filter and presented a sequential solution to the time-varying linear filtering problem, as explained in [10]. In [13], Stepanov states that many of Kalman's proofs were based on the orthogonal projection theorem. Kalman's proofs suggested that his algorithm would also remain optimal for non-Gaussian sequences, and its effectiveness in solving applied problems was soon to follow. For this reason, among others, the publication of this paper is often considered the mark of a new age in filtering and estimation theory. After Kalman's first publication, his algorithm was applied mainly in aerospace. In fact, researchers at NASA Ames Research Center and Charles Stark Draper Laboratory quickly found opportunities to apply Kalman's algorithm. The algorithm Kalman proposed is now known as the Kalman filter.

1.2 The First Application of the Kalman Filter

The first publicly known application of the Kalman filter was developed at NASA Ames Research Center in the 1960s during studies for navigation and control of the Apollo space capsule. While the Apollo mission was a feat in itself, there were numerous studies and extensive research that made it possible. Two primary areas of research that were identified in being crucial for the mission were that of midcourse navigation and guidance for the circumlunar mission, and the autopilot design for large, flexible body liquid-fuel boosters [10]. Researchers in the Dynamics Analysis Branch at Ames Research Center aimed their focus at the midcourse navigation and guidance problem. They sought technology for a system that would live solely in the spacecraft and be capable of making large computations quickly and efficiently. According to the authors of [10], the researchers had some success using the Wiener filer for guidance and navigation of beam-rider and homing missiles, and so they considered using the Wiener filter for this problem as well. However, they encountered difficulties with the nonlinearity of lunar vehicle navigation and the irregular series of discrete measurements [10]. Incidentally, Dr. Schmidt at Ames Research Center and Dr. Kalman were acquaintances, and Kalman arranged a visit to discuss topics of mutual interest. During this visit Kalman presented his first paper detailing his improvement on the Wiener filter. The researchers at Ames Research Center found Kalman's results very promising and began examining them in greater detail. After extensive research, it was determined that Kalman's original formulation would have required an on-board crew to make a continuous sequence of empirical measurements equally spaced in time throughout the mission, which is highly unrealistic. Therefore, a modification to Kalman's formulation was made that decomposed the original formulation into a discrete-time update portion and a discrete-time optical measurement update portion, so that measurements could be processed at any time interval [10]. This modification to the Kalman filter would come to be known as the ”extended Kalman filter.” Briefly, the extended Kalman filter produces similar results to the Kalman filter, but can be applied to continuous systems as well, which the (discrete) Kalman filter is not capable of. Researchers at Ames Research Center had several technical breakthroughs which led to this first major application of the Kalman filter. Some of these breakthroughs include adapting Kalman's original formulation to nonlinear problems and demonstrating that the Kalman filter could be used on-board in the Apollo spacecraft guidance and control system. A more detailed account of the adaptation of the Kalman filter to the Apollo mission is given in [10].

The Kalman filter has proven to be a fundamental tool for analyzing and solving various estimator problems. In addition to the Apollo mission, the Kalman filter has proven to be useful in applications across many disciplines including, but certainly not limited to, GPS navigation, economic predictions, satellite tracking, weather predictions, and monitoring populations. In the chapters that follow we will derive the formulation of the Kalman filter and apply it in some examples.

CHAPTER 2 BASIC PRINCIPLES OF THE KALMAN FILTER

2.1 Introductory Definitions and Results

We begin with some necessary definitions and results.

Definition 2.1.1. Consider an experiment with sample space S. A real-valued *random variable* X is a function from S to $\mathbb{R}$. If X takes on finitely many or countably many values, then X is a *discrete random variable*.

Definition 2.1.2. Suppose X is a random variable on the sample space S. The *probability density function* (*pdf*), denoted f_X, is defined by $f_X = \mathbb{P}(X = k)$ for all k in the range of X. If k is not in the range of X, then $f_X(k) = 0$.

Definition 2.1.3. Let Y be a function from a sample space S to $\mathbb{R}$. The function Y is called a *continuous random variable* if there exists a function $f_Y(y)$ such that for any real numbers $a < b$,

$$\mathbb{P}(a \leq Y \leq b) = \int_a^b f_Y(y)dy.$$

The values of a random variable may be any convenient mathematical entity, such as real or complex numbers, vectors, etc. Sums, products, and functions of random variables are also random variables [7].

Definition 2.1.4. [7] A *random* (or *stochastic*) *process* is a sequence of random variables, which can be finite or infinite.

Definition 2.1.5. Let X and Y be random variables. The *expected value* of X, denoted $E[X]$, is the weighted average of possible values of X. If X is a discrete random variable, then

$$E[X] = \mu_X = \sum_k k\mathbb{P}(X = k) = \sum_k k f_x(k).$$

If Y is a continuous random variable, then

$$E[Y] = \mu_Y = \int_{-\infty}^{\infty} y f_Y(y) dy.$$

The expected values of X and Y examine the central tendency of X and Y, respectively, but are not always possible values of X or Y.

Theorem 2.1.6. Suppose X and Y are random variables and a and b are constants. Then

1. $E[X + Y] = E[X] + E[Y]$

2. $E[aX] = aE[X]$

3. $E[b] = b$

In some situations the mean may describe what is being measured, while the standard deviation may represent noise and other interference.

Definition 2.1.7. Let X be a random variable with expected value $E[X] = \mu$. We define the *variance of* X, denoted σ^2, as the expected value of its square deviations from μ,

$$\text{Var}(X) = \sigma^2 = E[(X - \mu)^2].$$

The *standard deviation* of X, denoted σ, is

$$\sigma = \sqrt{\text{Var}(X)}.$$

If X is a discrete random variable, then

$$\text{Var}(X) = E[(X - \mu)^2] = \sum_k (k - \mu)^2 \mathbb{P}(X = k) = \sum_k (k - \mu)^2 f_X(k).$$

If Y is a continuous random variable, then

$$\text{Var}(Y) = E[(Y-\mu)^2] = \int_{-\infty}^{\infty} (y-\mu)^2 f_Y(y)dy.$$

Definition 2.1.8. The *distribution* of a variable tell us the values the variable can take on and communicates, by way of its mean μ and standard deviation σ, some sense for how frequently those values occur.

Some common distributions are the binomial distribution with pdf $f_x(n,p,k) = \binom{n}{k} kp^k(1-p)^{n-k}$, geometric distribution with pdf $f_x(p,k) = p(1-p)^{k-1}$, exponential distribution with pdf $f_Y(y,\lambda) = \lambda e^{-\lambda y}$, and the normal distribution.

Definition 2.1.9. A *normal distribution* in Y is a statistical distribution with probability density function

$$f_Y(y) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(y-\mu)^2}{2\sigma^2}}$$

with mean μ and standard deviation σ. A random variable Y distributed normally with mean μ and variance σ^2 is often denoted $Y \sim \text{N}(\mu, \sigma^2)$. A normal distribution is often called a *Gaussian distribution*.

In [4], Gallager states that it is very common and beneficial to use zero-mean Gaussian random variables to model noise and other random phenomena because they serve as good approximations to sums of independent zero-mean random variables, they are easy to manipulate analytically, and are usually the ”most random” random variable for a given variance.

Normal distributions have the well-known bell-shaped curves. Two examples of a normal distribution can be seen in figures 2.1 and 2.2.

Definition 2.1.10. [7] A sequence of random variables (finite or infinite)

$$\{X(t)\} = \ldots, X(-1), X(0), X(1), \ldots \tag{2.1.11}$$

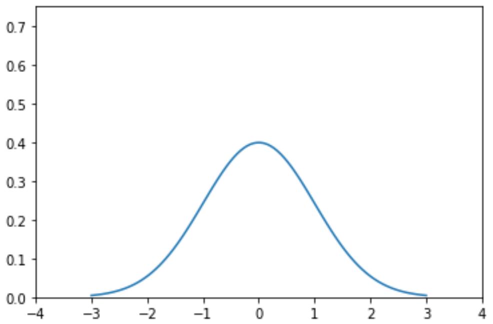

Figure 2.1 Normal distribution with $\mu = 0$ and $\sigma^2 = 1$.

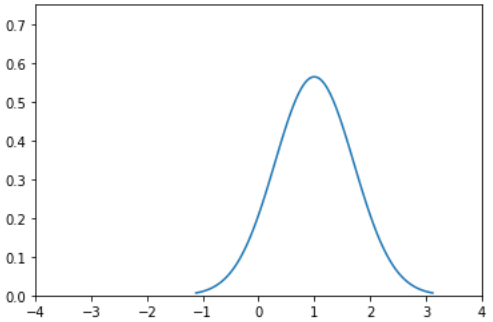

Figure 2.2 Normal distribution with $\mu = 1$ and $\sigma^2 = 0.5$.

is called a *discrete random* (or *stochastic*) *process*. One particular set of observed values

$$\dots f_X(-1), f_X(0), f_X(1), \dots$$

of the random process (2.1.11) is called a *realization* of the process.

Definition 2.1.12. Two random variables X and Y with probability distribution functions f_X and f_Y respectively are *independent random variables* if their joint probability $f_{X,Y}(x, y)$ is equal to the product of their individual probabilities,

$$f_{X,Y}(x, y) = f_X(x)f_Y(y).$$

Definition 2.1.13. A *random vector* is a vector whose entries are random variables. A *random matrix* is a matrix whose entries are random variables.

For example, if $X_1, \dots X_n$ are random variables, we can define the *random vector* ***X*** to be the column matrix

$$\mathbf{X} = \begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_n \end{bmatrix}$$

Similarly, for random variables X_{ij}, where $1 \leq i \leq m$ and $1 \leq j \leq n$, we can define the *random matrix*

$$\mathbf{M} = \begin{bmatrix} X_{11} & X_{12} & \dots & X_{1n} \\ X_{21} & X_{22} & \dots & X_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ X_{m1} & X_{m2} & \dots & X_{mn} \end{bmatrix}$$

Correspondingly, the expected value of **M** is the mean matrix

$$E[\mathbf{M}] = \begin{bmatrix} E[X_{11}] & E[X_{12}] & \dots & E[X_{1n}] \\ E[X_{21}] & E[X_{22}] & \dots & E[X_{2n}] \\ \vdots & \vdots & \ddots & \vdots \\ E[X_{m1}] & E[X_{m2}] & \dots & E[X_{mn}] \end{bmatrix}$$

Definition 2.1.14. [9] A random process $\{X(t)\}$ is *independent (in time)* or *white* if, for any choice of $t_1, \dots, t_n$, the random variables $X(t_1), \dots, X(t_n)$ are a set of independent random vectors.

Definition 2.1.15. A random process $\{X(t)\}$ is an *uncorrelated random process* if

$$E[X(s)X(t)] = E[X(s)]E[X(t)]$$

for any $s \neq t$.

Definition 2.1.16. Let X and Y be random variables with means μ_X and μ_Y. The *covariance* of X and Y is

$$\text{cov}(X, Y) = E[(X - \mu_x)(Y - \mu_Y)] \tag{2.1.17}$$

Definition 2.1.18. (Properties of covariance) Let X and Y be random variables with means μ_X and μ_Y. Then

1. $\text{cov}(aX + b, cY + d) = ac\,\text{cov}(X, Y)$, for constants a, b, c, d.

2. $\text{cov}(X_1 + X_2, Y) = \text{cov}(X_1, Y) + \text{cov}(X_2, Y)$

3. $\text{cov}(X, X) = \text{Var}(X)$

4. $\text{cov}(X, Y) = E[XY] - E[X]E[Y] = E[XY] - \mu_X\mu_Y$

5. If X and Y are independent, then $\text{cov}(X, Y) = 0$.

Property 4 is a convenient way to express and calculate the covariance of two random variables. As such, it is often taken as the definition for covariance. We will be using this expression shortly, so we will prove it.

Proof. Show $\text{cov}(X, Y) = E[XY] - \mu_X\mu_Y$. By definition 2.1.16,

$$\begin{aligned}
\text{cov}(X, X) &= E[(X - \mu_X)(Y - \mu_Y)] \\
&= E[XY - X\mu_Y - \mu_X Y + \mu_X\mu_Y] \\
&= E[XY] - E[X\mu_Y] - E[\mu_X Y] + E[\mu_X\mu_Y] \\
&= E[XY] - E[X]E[\mu_Y] - E[\mu_X]E[Y] + E[\mu_X]E[\mu_Y] \\
&= E[XY] - \mu_X\mu_Y - \mu_X\mu_Y + \mu_X\mu_Y \\
&= E[XY] - \mu_X\mu_Y.
\end{aligned}$$

□

Also note that the covariance of a random variable X with itself is its variance That is

$$\text{cov} = E[(X - \mu_X)(X - \mu_X)] = E[(X - \mu_X)^2] = \text{Var}(X). \tag{2.1.19}$$

Definition 2.1.20. Let **X** be a vector-valued random variable with components $X_1, \ldots, X_n$. The *covariance matrix for **X*** is

$$C_{\mathbf{X}} = E[(\mathbf{X} - E[\mathbf{X}])(\mathbf{X} - E[\mathbf{X}])^T]. \tag{2.1.21}$$

Variance measures variation of a single random variable; it measures how far the values are spread out from their mean. Covariance measures how two random variables vary together. If the random vector **X** has components $X_1, \ldots, X_n$, then its covariance matrix is

$$\begin{aligned} C_{\mathbf{X}} &= E[(\mathbf{X} - E[\mathbf{X}])(\mathbf{X} - E[\mathbf{X}])^T] \\ &= \begin{bmatrix} \text{var}(X_1) & \text{cov}(X_1, X_2) & \ldots & \text{cov}(X_1, X_n) \\ \text{cov}(X_2, X_1) & \text{var}(X_2) & \ldots & \text{cov}(X_2, X_n) \\ \vdots & \vdots & \ddots & \vdots \\ \text{cov}(X_n, X_1) & \text{cov}(X_n, X_2) & \ldots & \text{var}(X_n) \end{bmatrix} \end{aligned}$$

Example 2.1.22. We will compute a covariance matrix for a small data set. Suppose we have three random variables X_1, X_2, X_3 taking values $[14, 11, 20]$, $[137, 123, 106]$, and $[55, 42, 56]$ respectively. We then have the matrix

$$M = \begin{bmatrix} 14 & 137 & 55 \\ 11 & 123 & 42 \\ 20 & 106 & 56 \end{bmatrix} \tag{2.1.23}$$

The corresponding covariance matrix is

$$\text{cov}(M) = \begin{bmatrix} 21 & -\frac{99}{2} & \frac{57}{2} \\ -\frac{99}{2} & 241 & -\frac{29}{2} \\ \frac{57}{2} & -\frac{29}{2} & 61 \end{bmatrix} \tag{2.1.24}$$

Definition 2.1.25. X and Y are *uncorrelated random variables* if their covariance is zero,

$$\text{cov}(X, Y) = 0.$$

Equivalently, by Property 4 of definition 2.1.18, X and Y are *uncorrelated random variables* if

$$E[XY] = E[X]E[Y].$$

Lemma 2.1.26. If the random variables X and Y are independent, then they are uncorrelated.

Proof. If X and Y are independent, then $f_{X,Y} = f_X(x)f_Y(y)$ by definition 2.1.12. By definition 2.1.5,

$$\begin{aligned}
E[XY] &= \int\int xyf_{X,Y}(x,y)dydx \\
&= \int\int xyf_X(x)f_Y(y)dydx \\
&= \int xf_X(x)\Big(\int yf_Y(y)dy\Big)dx \\
&= \Big(\int xf_X(x)dx\Big)\Big(\int yf_Y(y)dy\Big) \\
&= E[X]E[Y]
\end{aligned}$$

Thus $\text{cov}(X, Y) = E[XY] - E[X]E[Y] = 0$, so X and Y are uncorrelated. □

Definition 2.1.27. *Noise* is an error that corrupts a true value. Noise can be random or systematic.

Definition 2.1.28. *White noise* is noise that has equal intensity at different frequencies, and is not correlated in time.

In other words, knowing the value of the white noise at time t provides no information about the value of the white noise at time $t + i$ for any i [9].

Definition 2.1.29. [2] *Gaussian noise* is defined as noise with a probability distribution function of a Gaussian (or normal) distribution.

Definition 2.1.30. *Additive noise* is noise that is added to the signal, or measurement, itself.

For example, the received signal r includes the original, noise free, signal s as well as the white Gaussian noise w [2],

$$r = s + w.$$

Therefore, *additive white Gaussian noise* is random white noise (uncorrelated in time) that follows a Gaussian distribution and is added directly to the received signal.

Definition 2.1.31. A *dynamical system* describes how one state of interest develops into another state over time.

2.2 The Kalman Filter

The Kalman filter addresses the problem of trying to estimate the state $x \in \mathbb{R}^n$ of a discrete-time controlled process given a time-varying sequence of noisy measurements [16]. The state x is a parameter of interest such as the height of a hot air balloon, the trajectory of an asteroid, or the gas level in the tank of a car. The Kalman filter assumes that 1) the underlying system is linear, 2) measurements are taken as linear combinations of the desired state, and 3) both system noise and measurement noise are white and Gaussian, and that these noises are independent [12].

The system model is governed by the equations

$$x_k = \Phi_{k-1}x_{k-1} + w_{k-1} \tag{2.2.1}$$

$$z_k = H_k x_k + v_k \tag{2.2.2}$$

where Φ_k are the state transition matrices that describe the state evolution at successive time steps. State transition matrices are matrices whose product with a state at an initial time gives the state at a later time. The system noise w_k is a zero-mean Gaussian white noise with covariance Q_k, following the normal distribution $N(0, Q_k)$. The vectors w_k are normally distributed random variables consisting of white noise corruptions to the states. The system covariance Q_k describes the relations between all state variables, and conveys a sense for how they vary compared to each other.

For example, height is proportional to weight, so the matrix Q_k would contain information about how strongly they are correlated in one of its entries. The vectors $z_k = [z_{1k}, z_{2k}, ..., z_{lk}]$ are empirical measurements. Since the Kalman filter applies to linear systems, the measurements are taken as linear combinations of the state variables. The matrices H_k describe how the measurements are related to the states, and converts those states into measurements by adjusting the dimension. The vectors v_k are zero-mean Gaussian white noises with covariances R_k, following the normal distribution $N(0, R_k)$. These vectors consist of white noise corruptions to the measurements. Notice from equation (2.2.1) that the Kalman filter is recursive.

Definition 2.2.3. [5] A *recursive filter* is a filter which re-uses one or more of its outputs as an input.

At each step the Kalman filter makes use of all available measurement data, plus prior knowledge about the system and measurement devices, to produce an estimate. The filter operates so that there is no need to store all past measurements, thus greatly reducing the computational power required. The recursivity of the Kalman filter is very advantageous since it often requires matrix computations which can become very expensive to execute.

Since the state and measurement noises are assumed to be independent, by lemma 2.1.26 we have

$$E[w_k v_j^T] = E[v_k w_j^T] = 0$$

for all j and k. We assume that w_k and v_k have zero mean and covariances Q_k and R_k respectively. Therefore by definition 2.1.20,

$$\begin{aligned} Q_k &= E[(w_k - E[w_k])(w_k - E[w_k])^T] \\ &= E[(w_k - \mu_{w_k})(w_k - \mu_{w_k})^T] \\ &= E[(w_k - 0)(w_k - 0)^T] \\ &= E[w_k w_k^T] \end{aligned} \tag{2.2.4}$$

and

$$\begin{aligned} R_k &= E[(v_k - E[v_k])(v_k - E[v_k])^T] \\ &= E[(v_k - \mu_{v_k})(v_k - \mu_{v_k})^T] \\ &= E[(v_k - 0)(v_k - 0)^T] \\ &= E[v_k v_k^T]. \end{aligned} \tag{2.2.5}$$

State estimate errors are found by calculating the difference between the true value and estimated value of the state. We define the *a priori* and *a posteriori* state estimate errors as

$$e_k(-) = \hat{x}_k(-) - x_k \iff \hat{x}_k(-) = e_k(-) + x_k \tag{2.2.6}$$

$$e_k(+) = \hat{x}_k(+) - x_k \iff \hat{x}_k(+) = e_k(+) + x_k \tag{2.2.7}$$

respectively.

The Kalman Filter is a predict-correct algorithm consisting of five equations: two prediction equations and three update equations.

<u>Predict Equations:</u>

$$\begin{aligned} \text{First step: } & \hat{x}_k(-) = \Phi_{k-1}\hat{x}_{k-1}(+) \\ \text{Second step: } & P_k(-) = \Phi_{k-1}P_{k-1}(+)\Phi_{k-1}^T + Q_{k-1} \end{aligned}$$

<u>Update Equations:</u>

$$\begin{aligned} \text{First step: } & K_k = P_k(-)H_k^T[H_k P_k(-)H_k^T + R_k]^{-1} \\ \text{Second step: } & \hat{x}_k(+) = \hat{x}_k(-) + K_k[z_k - H_k\hat{x}_k(-)] \\ \text{Third step: } & P_k(+) = [I - K_k H_k]P_k(-) \end{aligned}$$

Throughout the text, $(-)$ indicates that a computation is a *prediction* and is occurring in the *state* *space*. Likewise, $(+)$ indicates that a computation is an *update*, or correction, and is occuring in the *measurement space*. Here $\hat{x}_k(-)$ is the *a priori* (predicted) state estimate, $\hat{x}_k(+)$ is the *a posteriori* updated state estimate, $P_k(-)$ is the *a priori* predicted state estimate error covariance, and $P_k(+)$ is the *a posteriori* updated state estimate error covariance. The system noise Q_k and measurement noise R_k are to be calculated before running the filter using equations (2.2.4) and (2.2.5) [2]. The covariance matrices relate the measurement inputs to the state predictions. Figure 2.3 helps to visualize the algorithm. To initialize the filter we require

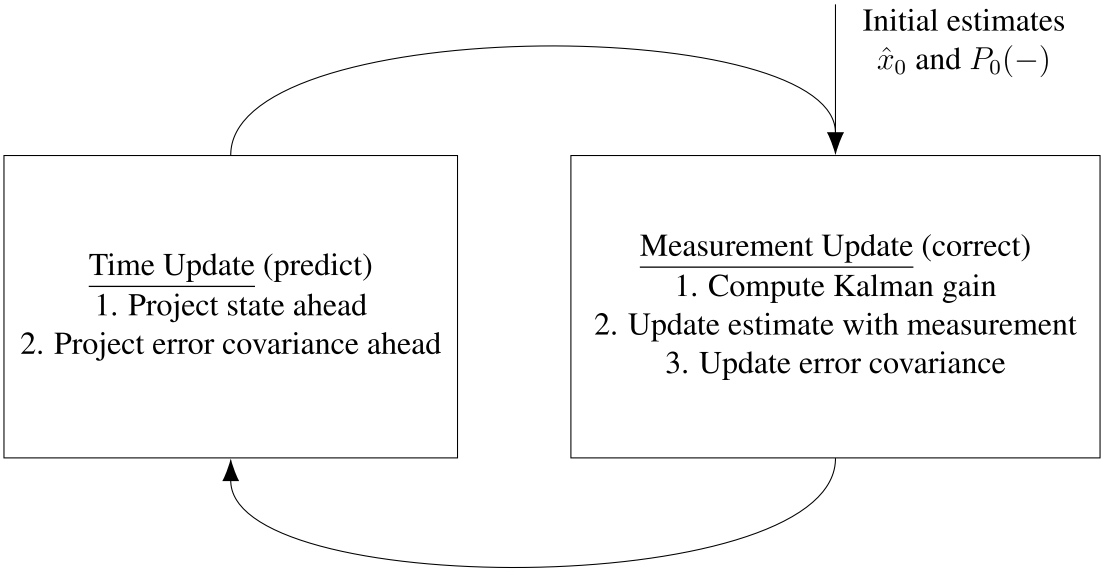
Initial estimates
$\hat{x}_0$ and $P_0(-)$
Time Update (predict)
1. Project state ahead
2. Project error covariance ahead
Measurement Update (correct)
1. Compute Kalman gain
2. Update estimate with measurement
3. Update error covariance

Figure 2.3 Kalman Filter flow diagram.

$$E[x_0] = \hat{x}_0 \tag{2.2.8}$$

and

$$E[e_0(-)e_0(-)^T] = P_0. \tag{2.2.9}$$

A note to the reader: the derivations that follow are nontrivial, so great detail is included for the sake of completeness and precision.

2.3 Formulation of State Estimate Update $\hat{x}_k(+)$

Our goal in deriving $\hat{x}_k(+)$ is to update our estimate $\hat{x}_k(-)$ using our measurement z_k. Naturally, we expect $\hat{x}_k(+)$ to be an improvement of $\hat{x}_k(-)$. The *a priori* state estimate and the measurement at each iteration both contribute to the computation of the *a posteriori* state estimate $\hat{x}_k(+)$. Therefore, we start by expressing the state update $\hat{x}_k(+)$ in a linear and recursive form as in [5]

$$\hat{x}_k(+) = K'_k \hat{x}_k(-) + K_k z_k \tag{2.3.1}$$

where K' and K are two distinct weighting matrices, both of which are yet to be defined.

Using the measurement from equation (2.2.2) and the estimation error relations from equations (2.2.6) and (2.2.7), we are able to express $e_k(+)$ as

$$\begin{aligned}
\hat{x}_k(+) &= K'_k \hat{x}_k(-) + K_k z_k \\
x_k + e_k(+) &= K'_k[x_k + e_k(-)] + K_k[H_k x_k + v_k] \\
e_k(+) &= K'_k x_k + K'_k e_k(-) + K_k H_k x_k + K_k v_k - x_k \\
e_k(+) &= [K'_k + K_k H_k - 1]x_k + K'_k e_k(-) + K_k v_k. \qquad (2.3.2)
\end{aligned}$$

Definition 2.3.3. The *bias* of an estimator $\hat{\theta}$, denoted $B(\hat{\theta})$, is the difference between the estimator's expected value and the true value of the parameter being estimated,

$$B(\hat{\theta}) = E[\hat{\theta}] - \theta.$$

An estimator $\hat{\theta}$ is *unbiased for* θ if $B(\hat{\theta}) = 0$, that is if $E[\hat{\theta}] = \theta$.

By design, v_k is a zero-mean Gaussian white noise, therefore $E[v_k] = \mu_{v_k} = 0$. Ideally, we want

$$E[e_k(+)] = E[\hat{x}_k(+) - x_k] = E[\hat{x_k}(+)] - E[x_k] = x_k - x_k = 0$$

and

$$E[e_k(-)] = E[\hat{x}_k(-) - x_k] = E[\hat{x}_k(-)] - E[x_k] = x_k - x_k = 0.$$

Intuitively, this is to say that we want the *a priori* and *a posteriori* state estimate errors as small as possible. Therefore, we require $\hat{x}_k(+)$ and $\hat{x}_k(-)$ to be unbiased estimators. Now using equation (2.3.2) we can determine K'. We start by taking expected values on both sides

$$\begin{aligned} E[e_k(+)] &= E\big[[K'_k + K_k H_k - 1]x_k + K'_k e_k(-) + K_k v_k\big] \\ 0 &= E[[K'_k + K_k H_k - 1]x_k] + E[K'_k e_k(-)] + E[K_k v_k] \\ 0 &= E[K'_k + K_k H_k - 1]E[x_k] + E[K'_k]E[e_k(-)] + E[K_k]E[v_k] \\ 0 &= [K'_k + K_k H_k - 1]E[x_k] + E[K'_k] \cdot 0 + E[K_k] \cdot 0 \\ 0 &= [K'_k + K_k H_k - 1]E[x_k]. \end{aligned}$$

Therefore we require

$$\begin{aligned} 0 &= K'_k + K_k H_k - 1 \\ K'_k &= 1 - K_k H_k. \end{aligned}$$

Now equation (2.3.1) becomes

$$\begin{aligned} \hat{x}_k(+) &= K'_k \hat{x}_k(-) + K_k z_k \\ &= [1 - K_k H_k]\hat{x}_k(-) + K_k z_k \\ &= \hat{x}_k(-) - K_k H_k \hat{x}_k(-) + K_k z_k \\ &= \hat{x}_k(-) + K_k[z_k - H_k \hat{x}_k(-)] \qquad (2.3.4) \end{aligned}$$

which is our desired formulation.

2.4 Formulation of Error Covariance Update $P_k(+)$

As explained above, we denote the *a priori* state estimate error covariance as $P_k(-)$, and the *a posteriori* state estimate error covariance as $P_k(+)$. These covariances correspond to the *a priori* and *a posteriori* state estimate errors $e_k(-)$ and $e_k(+)$, respectively. By the preceding arguments and definition 2.1.20, we have

$$\begin{aligned} P_k(-) &= E\Big[\big(e_k(-) - E[e_k(-)]\big)\big(e_k(-) - E[e_k(-)]\big)^T\Big] \\ &= E\Big[\big(\hat{x}_k(-) - x_k - E[\hat{x}_k(-) - x_k]\big)\big(\hat{x}_k(-) - x_k - E[\hat{x}_k(-) - x_k]\big)^T\Big] \\ &= E\Big[\big(\hat{x}_k(-) - x_k - (x_k - x_k)\big)\big(\hat{x}_k(-) - x_k - (x_k - x_k)\big)^T\Big] \\ &= E\Big[\big(\hat{x}_k(-) - x_k\big)\big(\hat{x}_k(-) - x_k\big)^T\Big] \\ &= E[e_k(-)e_k(-)^T] \end{aligned} \tag{2.4.1}$$

and

$$\begin{aligned} P_k(+) &= E\Big[\big(e_k(+) - E[e_k(+)]\big)\big(e_k(+) - E[e_k(+)]\big)^T\Big] \\ &= E\Big[\big(\hat{x}_k(+) - x_k - E[\hat{x}_k(+) - x_k]\big)\big(\hat{x}_k(+) - x_k - E[\hat{x}_k(+) - x_k]\big)^T\Big] \\ &= E\Big[\big(\hat{x}_k(+) - x_k - (x_k - x_k)\big)\big(\hat{x}_k(+) - x_k - (x_k - x_k)\big)^T\Big] \\ &= E\Big[\big(\hat{x}_k(+) - x_k\big)\big(\hat{x}_k(+) - x_k\big)^T\Big] \\ &= E[e_k(+)e_k(+)^T] \end{aligned} \tag{2.4.2}$$

which we take as definitions for $P_k(-)$ and $P_k(+)$. To formulate $P_k(+)$ we need an updated expression for $e_k(+)$. Using equations (2.2.2), (2.2.6), (2.2.7), and (2.3.4), we have

$$\begin{aligned}
e_k(+) &= \hat{x}_k(+) - x_k \\
&= \hat{x}_k(-) + K_k[z_k - H_k\hat{x}_k(-)] - x_k \\
&= \hat{x}_k(-) + K_kz_k - K_kH_k\hat{x}_k(-) - x_k \\
&= \hat{x}_k(-) + K_k(H_kx_k + v_k) - K_kH_k\hat{x}_k(-) - x_k \\
&= \hat{x}_k(-) - x_k + K_kH_kx_k + K_kv_k - K_kH_k\hat{x}_k(-) \\
&= \hat{x}_k(-) - x_k + K_kH_k(x_k - \hat{x}_k(-)) + K_kv_k \\
&= e_k(-) + K_kH_ke_k(-) + K_kv_k \\
&= (I - K_kH_k)e_k(-) + K_kv_k. \qquad (2.4.3)
\end{aligned}$$

To simplify the expression for $P_k(+)$ we will need to know the value of $E[e_k(-)v_k^T]$. Recall that $E[v_k] = 0$. Since the state estimate errors $e_k(-)$ and measurement noise v_k are uncorrelated [5], by definition (2.1.15) we have

$$E[e_k(-)v_k^T] = E[e_k(-)]E[v_k^T] = E[v_k^T]E[e_k(-)] = E[v_ke_k(-)^T] = 0. \qquad (2.4.4)$$

Substituting (2.4.3) into (2.4.2) and using (2.4.4), we have

$$\begin{aligned}
P_k(+) &= E[e_k(+)e_k(+)^T] \\
&= E\Big[\Big((I - K_kH_k)e_k(-) + K_kv_k\Big)\Big((I - K_kH_k)e_k(-) + K_kv_k\Big)^T\Big], \text{ by 2.4.3} \\
&= E\Big[\Big((I - K_kH_k)e_k(-) + K_kv_k\Big)\Big(((I - K_kH_k)e_k(-))^T + (K_kv_k)^T\Big)\Big] \\
&= E\Big[\Big((I - K_kH_k)e_k(-) + K_kv_k\Big)\Big(e_k(-)^T(I - K_kH_k)^T + v_k^TK_k^T\Big)\Big] \\
&= E\Big[(I - K_kH_k)e_k(-)\Big(e_k(-)^T(I - K_kH_k)^T + v_k^TK_k^T\Big) \\
&\qquad + K_kv_k\Big(e_k(-)^T(I - K_kH_k)^T + v_k^TK_k^T\Big)\Big] \\
&= E\Big[(I - K_kH_k)e_k(-)e_k(-)^T(I - K_kH_k)^T + (I - K_kH_k)e_k(-)v_k^TK_k^T \\
&\qquad + K_kv_ke_k(-)^T(I - K_kH_k)^T + K_kv_kv_k^TK_k^T\Big] \\
&= E\Big[(I - K_kH_k)e_k(-)e_k(-)^T(I - K_kH_k)^T\Big] + E\Big[(I - K_kH_k)e_k(-)v_k^TK_k^T\Big] \\
&\qquad + E\Big[K_kv_ke_k(-)^T(I - K_kH_k)^T\Big] + E\Big[K_kv_kv_k^TK_k^T\Big] \\
&= E\Big[(I - K_kH_k)\Big]E\Big[e_k(-)e_k(-)^T\Big]E\Big[(I - K_kH_k)^T\Big] \\
&\qquad + E\Big[(I - K_kH_k)\Big]E\Big[e_k(-)v_k^T\Big]E\Big[K_k^T\Big] + E\Big[K_k\Big]E\Big[v_ke_k(-)^T\Big]E\Big[(I - K_kH_k)^T\Big] \\
&\qquad + E\Big[K_k\Big]E\Big[v_kv_k^T\Big]E\Big[K_k^T\Big] \\
&= (I - K_kH_k)P_k(-)(I - K_kH_k)^T + (I - K_kH_k)\cdot 0 \cdot K_k^T \\
&\qquad + K_k^T \cdot 0 \cdot (I - K_kH_k)^T + K_kR_kK_k^T, \text{ by equ. 2.4.1, equ. 2.4.4, def. 2.2.4} \\
&= (I - K_kH_k)P_k(-)(I - K_kH_k)^T + K_kR_kK_k^T.
\end{aligned}$$

Thus we obtain

$$P_k(+) = (I - K_kH_k)P_k(-)(I - K_kH_k)^T + K_kR_kK_k^T. \tag{2.4.5}$$

This formula for $P_k(+)$ is known as the ”Joseph form” of the covariance update equation; it was derived by P.D. Joseph in 1968. More information on the Joseph form equation can be found in [1]. We will update this expression for $P_k(+)$ once we formulate the Kalman gain matrix.

2.5 Formulation of the Kalman gain K_k

The Kalman filter operates by providing a least-squares best estimate of the desired state, which is accomplished by minimizing the variance of the estimation error.

Definition 2.5.1. [9] *Least-squares estimation* is a classical technique used to minimize the sum of squares of the differences between the actual measurement data and the proposed estimate.

The Kalman gain matrix K_k is chosen to minimize a weighted scalar sum of the diagonal elements of the error covariance matrix $P_k(+)$, which provides the least-squares aspect to the filter [5].

Definition 2.5.2. The *trace* of an $n \times n$ matrix A is the scalar sum of its diagonal entries

$$\text{trace}(A) = \sum_{i=1}^{n} a_{i,i}.$$

Therefore, K_k is chosen to minimize trace$[P_k(+)]$. This derivation will require the following lemma.

Lemma 2.5.3. [5] Let A and B be two matrices with B symmetric. Then $\frac{\partial}{\partial A}\text{trace}[ABA^T] = 2AB$.

Proof. For scalars x_p and x_q, we note that

$$\frac{\partial x_p}{\partial x_q} = \begin{cases} 1, & p = q \\ 0, & p \neq q \end{cases} \tag{2.5.4}$$

A typical term in ABA^T is $\sum_j a_{ij} b_{jk} (A^T)_{ki} = \sum_j a_{ij} b_{jk} a_{ik}$. Therefore, the (m, n) entry in

$$\frac{\partial}{\partial A}\text{trace}(ABA^T)$$

is of the form

$$\begin{aligned}
\frac{\partial}{\partial a_{mn}} \sum_i \sum_j a_{ij} b_{jk} a_{ki} &= \frac{\partial}{\partial a_{mn}} \sum_{ijk} a_{ij} b_{jk} a_{ik} \\
&= \sum_{ijk} \frac{\partial}{\partial a_{mn}} (a_{ij} b_{jk} a_{ik}) \\
&= \sum_{ijk} \frac{\partial a_{ij}}{\partial a_{mn}} (b_{jk} a_{ik}) + \sum_{ijk} (a_{ij}) b_{jk} \frac{\partial a_{ik}}{\partial a_{mn}} \text{ by the product rule} \\
&= \sum_{ijk} \delta_{im} \delta_{jn} (b_{jk} a_{ik}) + \sum_{ijk} (a_{ij} b_{jk}) \delta_{im} \delta_{kn} \text{ by (2.5.4)} \\
&= \sum_k b_{nk} a_{mk} + \sum_j a_{mj} b_{jn} \text{ since } m \text{ and } n \text{ are fixed} \\
&= \sum_k a_{mk} b_{nk} + \sum_j a_{mj} b_{jn} \\
&= \sum_k a_{mk} (B^T)_{kn} + \sum_j a_{mj} b_{jn} \\
&= (AB^T)_{mn} + (AB)_{mn} \\
&= (AB)_{mn} + (AB)_{mn} \text{ since } B = B^T \\
&= 2(AB)_{mn}.
\end{aligned}$$

Note here that B *must* be symmetric in order to achieve our result. If not, then we are simply left with an expression of the form $\frac{\partial}{\partial A}\text{trace}[ABA^T] = (AB^T)_{mn} + (AB)_{mn}$ which does not aid in our formulation of K_k.

□

To find K_k that provides a minimum, we need to solve

$$\frac{\partial}{\partial K_k}\text{trace}[P_k(+)] = 0 \qquad (2.5.5)$$

for K_k. After substituting equation (2.4.5), we must solve

$$\begin{aligned}
0 &= \frac{\partial}{\partial K_k}\text{trace}[P_k(+)] \\
0 &= \frac{\partial}{\partial K_k}\text{trace}[(I - K_k H_k)P_k(-)(I - K_k H_k)^T + K_k R_k K_k^T] \\
0 &= \frac{\partial}{\partial K_k}\Big[\text{trace}[(I - K_k H_k)P_k(-)(I - K_k H_k)^T] + \text{trace}[K_k R_k K_k^T]\Big] \\
0 &= \frac{\partial}{\partial K_k}\Big[\text{trace}[(I - K_k H_k)P_k(-)(I - K_k H_k)^T]\Big] + \frac{\partial}{\partial K_k}\Big[\text{trace}[K_k R_k K_k^T]\Big]. \qquad (2.5.6)
\end{aligned}$$

First we will compute $\frac{\partial}{\partial K_k}\Big[\text{trace}[(I - K_k H_k)P_k(-)(I - K_k H_k)^T]\Big]$. To avoid over complications with subscripts in the following computations, we will consider a fixed time step k and simply write $A_{i,j}$ to denote the i, j entry in the matrix A_k. That is, we take $A_{i,j} = A_{k_{i,j}}$. We proceed with a change of variables; let

$$J = I - KH = I - K_k H_k = J_k.$$

A typical entry in J is of the form

$$\begin{aligned}
J_{i,j} &= (I - KH)_{i,j} \\
&= \delta_{i,j} - (KH)_{i,j} \\
&= \delta_{i,j} - \sum_l K_{i,l} H_{l,j}
\end{aligned}$$

where

$$\delta_{i,j} = \begin{cases} 1, & i = j \\ 0 & i \neq j \end{cases}$$

is the Kronecker delta function. A typical entry in $\frac{\partial J}{\partial K}$ is of the form

$$\begin{aligned}\frac{\partial J_{i,j}}{\partial K_{m,n}} &= \frac{\partial}{\partial K_{m,n}}\Big(\partial_{i,j} - \sum_l K_{i,l}H_{l,j}\Big) \\ &= 0 - \frac{\partial}{\partial K_{m,n}}\big(\sum_l K_{i,l}H_{l,j}\big) \\ &= -\sum_l \frac{\partial}{\partial K_{m,n}}(K_{i,l}H_{l,j}) \\ &= -H_{n,j}, \text{where } i = m \text{ and } l = n.\end{aligned}$$

Let S be some matrix dependent on J. Using the chain rule for derivatives we have

$$\begin{aligned}\frac{\partial S}{\partial K_{m,n}} &= \sum_{i,j} \frac{\partial S}{\partial J_{i,j}} \frac{\partial J_{i,j}}{\partial K_{m,n}} \\ &= \sum_{i,j} \frac{\partial S}{\partial J_{i,j}}(-H_{n,j}), \text{where } i = m \text{ and } l = n \\ &= -\sum_j \frac{\partial S}{\partial J_{m,j}} H_{n,j} \\ &= -\sum_j \frac{\partial S}{\partial J_{m,j}} (H^T)_{j,n}.\end{aligned}$$

Therefore

$$\frac{\partial S}{\partial K} = \frac{\partial S}{\partial J}\frac{\partial J}{\partial K} = -\frac{\partial S}{\partial J}H^T. \tag{2.5.7}$$

Now let

$$\begin{aligned}S &= \text{trace}[(I - K_kH_k)P_k(-)(I - K_kH_k)^T] \\ &= \text{trace}[J_kP_k(-)J_k^T] \\ &= \text{trace}[JPJ^T].\end{aligned}$$

Note that both $P_k(-)$ and R_k are symmetric by definition 2.1.20 and equation 2.2.5, respectively.

Using equation (2.5.7) we compute

$$\begin{aligned}
\frac{\partial S}{\partial K} &= \frac{\partial S}{\partial J}\frac{\partial J}{\partial K} \\
&= \Big(- \frac{\partial S}{\partial J}\Big) H^T \\
&= \Big(- \frac{\partial}{\partial J}\text{trace}[JP(-)J^T]\Big) H^T \\
&= \big(- 2JP(-)\big) H^T \text{ by lemma 2.5.3} \\
&= -2(I - KH)P(-)H^T. \qquad (2.5.8)
\end{aligned}$$

Thus $\frac{\partial}{\partial K_k}\Big[\text{trace}[(I - K_kH_k)P_k(-)(I - K_kH_k)^T]\Big] = \frac{\partial}{\partial K_k}S = -2(I - K_kH_k)P_k(-)H_k^T$.

To compute $\frac{\partial}{\partial K_k}\Big[\text{trace}[K_kR_kK_k^T]\Big]$ we simply apply lemma 2.5.3

$$\frac{\partial}{\partial K_k}\Big[\text{trace}[K_kR_kK_k^T]\Big] = 2K_kR_k. \qquad (2.5.9)$$

Finally by substituting (2.5.8) and (2.5.9) into (2.5.6), we obtain

$$\begin{aligned}
0 &= \frac{\partial}{\partial K_k}\Big[\text{trace}[(I - K_kH_k)P_k(-)(I - K_kH_k)^T]\Big] + \frac{\partial}{\partial K_k}\Big[\text{trace}[K_kR_kK_k^T]\Big] \\
0 &= -2(I - K_kH_k)P_k(-)H_k^T + 2K_kR_k.
\end{aligned}$$

Solving this for K_k, we have

$$\begin{aligned}
0 &= -2(I - K_kH_k)P_k(-)H_k^T + 2K_kR_k \\
0 &= -2IP_k(-)H_k^T + 2K_kH_kP_k(-)H_k^T + 2K_kR_k \\
2IP_k(-)H_k^T &= K_k(2H_kP_k(-)H_k^T + 2R_k) \\
\frac{2IP_k(-)H_k^T}{2H_kP_k(-)H_k^T + 2R_k} &= K_k \\
\frac{P_k(-)H_k^T}{H_kP_k(-)H_k^T + R_k} &= K_k \\
K_k &= P_k(-)H_k^T[H_kP_k(-)H_k^T + R_k]^{-1} \qquad (2.5.10)
\end{aligned}$$

as desired.

We use this value of K_k to optimize and rewrite our updated estimation error covariance $P_k(+)$. Expanding equation (2.4.5) gives

$$\begin{aligned} P_k(+) &= (I - K_kH_k)P_k(-)(I - K_kH_k)^T + K_kR_kK_k^T \\ &= (P_k(-) - K_kH_kP_k(-))(I - (K_kH_k)^T) + K_kR_kK_k^T \\ &= P_k(-) - P_k(-)(K_kH_k)^T - K_kH_kP_k(-) + K_kH_kP_k(-)(K_kH_k)^T + K_kR_kK_k^T \\ &= P_k(-) - K_kH_kP_k(-) - P_k(-)H_k^TK_k^T + K_kH_kP_k(-)H_k^TK_k^T + K_kR_kK_k^T \\ &= P_k(-) - K_kH_kP_k(-) - P_k(-)H_k^TK_k^T + K_k(H_kP_k(-)H_k^T + R_k)K_k^T. \end{aligned}$$

Substituting in equation (2.5.10) yields our desired formulation

$$\begin{aligned} P_k(+) &= P_k(-) - K_kH_kP_k(-) - P_k(-)H_k^T\left(\frac{P_k(-)H_k^T}{H_kP_k(-)H_k^T + R_k}\right)^T \\ &\quad + \left(\frac{P_k(-)H_k^T}{H_kP_k(-)H_k^T + R_k}\right)(H_kP_k(-)H_k^T + R_k)\left(\frac{P_k(-)H_k^T}{H_kP_k(-)H_k^T + R_k}\right)^T \\ &= P_k(-) - K_kH_kP_k(-) - \frac{P_k(-)H_k^T(P_k(-)H_k^T)^T}{(H_kP_k(-)H_k^T + R_k)^T} + \frac{P_k(-)H_k^T(P_k(-)H_k^T)^T}{(H_kP_k(-)H_k^T + R_k)^T} \\ &= (1 - K_kH_k)P_k(-). \end{aligned}$$

2.6 Formulation of State Estimate and Covariance Projections $\hat{x}(-)$ and $P_k(-)$

By the nature of our system model, the state estimate projection is given by

$$\hat{x}_k(-) = \Phi_{k-1}\hat{x}_{k-1}(+). \tag{2.6.1}$$

To complete the recursion it is necessary to find an equation which projects the *a priori* error covariance matrix $P_k(-)$ into the next time interval. First we find an expression for the *a priori* state estimate error. Using equations (2.2.1), (2.2.6), and (2.6.1) we have

$$\begin{aligned}
\hat{x}_k(-) &= \Phi_{k-1}\hat{x}_{k-1}(+) \\
\hat{x}_k(-) - x_k &= \Phi_{k-1}\hat{x}_{k-1}(+) - x_k \\
\hat{x}_k(-) - x_k &= \Phi_{k-1}\hat{x}_{k-1}(+) - (\Phi_{k-1}x_{k-1} + w_{k-1}) \text{ by (2.2.1)} \\
\hat{x}_k(-) - x_k &= \Phi_{k-1}(\hat{x}_{k-1}(+) - x_{k-1}) - w_{k-1} \\
e_k(-) &= \Phi_{k-1}e_{k-1}(+) - w_{k-1} \text{ by (2.2.6) and (2.2.7).}
\end{aligned}$$

Substituting this into (2.4.1) and using (2.4.4), we have

$$\begin{aligned}
P_k(-) &= E[e_k(-)e_k(-)^T] \\
&= E[(\Phi_{k-1}e_{k-1}(+) - w_{k-1})(\Phi_{k-1}e_{k-1}(+) - w_{k-1})^T] \\
&= E[(\Phi_{k-1}e_{k-1}(+) - w_{k-1})((\Phi_{k-1}e_{k-1}(+))^T - w_{k-1}^T)] \\
&= E[(\Phi_{k-1}e_{k-1}(+))(\Phi_{k-1}e_{k-1}(+))^T \\
&\qquad - (\Phi_{k-1}e_{k-1}(+))w_{k-1}^T - w_{k-1}(\Phi_{k-1}e_{k-1}(+))^T + w_{k-1}w_{k-1}^T] \\
&= E[(\Phi_{k-1}e_{k-1}(+))(\Phi_{k-1}e_{k-1}(+))^T] - E[(\Phi_{k-1}e_{k-1}(+))w_{k-1}^T] \\
&\qquad - E[w_{k-1}(\Phi_{k-1}e_{k-1}(+))^T] + E[w_{k-1}w_{k-1}^T] \\
&= E[\Phi_{k-1}e_{k-1}e_{k-1}(+)^T\Phi_{k-1}^T] - E[\Phi_{k-1}e_{k-1}(+)w_{k-1}^T] - \\
&\qquad E[w_{k-1}e_{k-1}(+)^T\Phi_{k-1}^T] + E[w_{k-1}w_{k-1}^T] \\
&= \Phi_{k-1}E[e_{k-1}e_{k-1}(+)^T]\Phi_{k-1}^T - \Phi_{k-1}E[e_{k-1}(+)w_{k-1}^T] - \\
&\qquad E[w_{k-1}e_{k-1}(+)^T]\Phi_{k-1}^T + E[w_{k-1}w_{k-1}^T] \\
&= \Phi_{k-1}P_{k-1}(+)\Phi_{k-1}^T - \Phi_{k-1}\cdot 0 - 0\cdot\Phi_{k-1}^T + Q_{k-1} \\
&= \Phi_{k-1}P_{k-1}(+)\Phi_{k-1}^T + Q_{k-1}
\end{aligned}$$

as desired.

CHAPTER 3 EXAMPLES AND SIMULATIONS

Example 3.0.1. This example is taken from [15]. Suppose we have a tank filled with some amount of water and we want to estimate the height of the water inside the tank. In this case the "state" we are interested in is the water level inside the tank. We are not able to see the inside of the tank, but a floating sensor inside provides a measurement reading of the height of the water every second. For this basic static model we assume that the true level of the water is constant, $L = c$, because we do not expect the water level to change. Since the water level is completely unknown, our initial state estimate is arbitrary: $\hat{x}_0 = 0$. For the same reason, our initial estimate for the state error covariance is fairly high: $P_0 = 1000$. If we were absolutely certain that our initial state estimate $\hat{x}_0 = 0$ is correct, then we would choose $P_0 = 0$. In this case the state transition matrix is $\Phi_k = 1$ for all k, and the measurement transition matrix is $H_k = 1$ for all k. Suppose we are fairly confident in the accuracy of our system model, so we choose the process noise to be rather small, say $Q_k = 0.0001$ for all k. Assume the measurement noise is normally distributed with zero mean and standard deviation $\sqrt{0.1}$. Therefore the measurement noise covariance is $R_k = (\sqrt{0.1})^2 = 0.1$ for all k. Finally, the measurement readings from the sensor are

$$z_k = \{0.9, 0.8, 1.1, 1, 0.95, 1.05, 1.2, 0.9, 0.85, 1.15\}.$$

We will compute the predict and update equations for the first time step of the Kalman filter directly. Note that $\hat{x}_0 = 0$ and $P_0 = 1000$. For our prediction equations we compute our first state estimate $\hat{x}_1(-)$, then we compute our first state error covariance estimate $P_1(-)$.

$$\hat{x}_1(-) = \Phi_0 \hat{x}_0 = 1 \cdot 0 = 0$$
$$P_1(-) = \Phi_0 P_0 \Phi_0^T + Q_0 = 1 \cdot 1000 \cdot 1 + 0.0001 = 1000.0001$$

Now we update, or correct, these two values with the update equations. For this, we compute the

Kalman Gain K_k, use that value and $\hat{x}_1(-)$ to update our state estimate, then use K_k and $\hat{x}_1(+)$ to update our error covariance estimate.

$$K_1 = P_1(-)H_1^T[H_1P_1(-)H_1^T + R_1]^{-1} = 1000.0001 \cdot 1(1 \cdot 1000.0001 \cdot 1 + 0.1)^{-1} = 0.9999$$

$$\hat{x}_1(+) = \hat{x}_1(-) + K_1[z_1 - H_1\hat{x}_1(-)] = 0 + 0.9999(0.9 - 1 \cdot 0) = 0.8999$$

$$P_1(+) = [I - K_1H_1]P_1(-) = (1 - 0.9999 \cdot 1)(1000.0001) = 0.1$$

We proceed with the computations for our second time step. For our prediction equations we have

$$\hat{x}_2(-) = \Phi_1\hat{x}_1(+) = 1 \cdot 0.8999 = 0.8999$$

$$P_2(-) = \Phi_1P_1(+)\Phi_1^T + Q_1 = 1 \cdot 0.1 \cdot 1 + 0.0001 = 0.1001,$$

and for our update equations we have

$$K_2 = P_k(-)H_k^T[H_kP_k(-)H_k^T + R_k]^{-1} = 0.1001 \cdot 1(1 \cdot 0.1001 \cdot 1 + 0.1)^{-1} = 0.5002$$

$$\hat{x}_2(+) = \hat{x}_2(-) + K_2[z_2 - H_2\hat{x}_2(-)] = 0.8999 + 0.5002(0.8 - 1 \cdot 0.8999) = 0.8499$$

$$P_2(+) = [I - K_2H_2]P_2(-) = (1 - 0.5002 \cdot 1)(0.1001) = 0.05.$$

We proceed in a similar manner to compute the remaining time steps. These computations are summarized in table 3.1. To help visualize this process we can examine the graph in figure 3.1.

The state estimations seem to be approaching $L = 1$, and taking this as the true value of the water level is a reasonable conclusion. Compared to the true value, the measurements ranging from 0.8 to 1.2 are noisy. In this case, the state estimations are within 0.05 of the true value after just four iterations.

To illustrate the smoothing properties of the Kalman filter, we can form another estimate with

	Predict		Update			
t	$\hat{x}_k(-)$	$P_k(-)$	z_k	K_k	$\hat{x}_k(+)$	$P_k(+)$
0	-	-	-	-	0	1000.0001
1	0	1000.0001	0.9	0.9999	0.8999	0.1
2	0.8999	0.1001	0.8	0.5002	0.8499	0.05
3	0.8499	0.0501	1.1	0.3339	0.9334	0.0334
4	0.9334	0.0335	1	0.2509	0.9501	0.0251
5	0.9501	0.0252	0.95	0.2012	0.9501	0.0201
6	0.9501	0.0202	1.05	0.1682	0.9669	0.0168
7	0.9669	0.0169	1.2	0.1447	1.0006	0.0145
8	1.0006	0.0146	0.9	0.1272	0.9878	0.0127
9	0.9878	0.0128	0.85	0.1136	0.9722	0.0114
10	0.9722	0.0115	1.15	0.1028	0.9905	0.0103

Table 3.1 Computations for measurements z_k.
$\hat{x}_0 = 0$, $Q_k = 0.0001$, and $R_k = (\sqrt{0.1})^2 = 0.1$

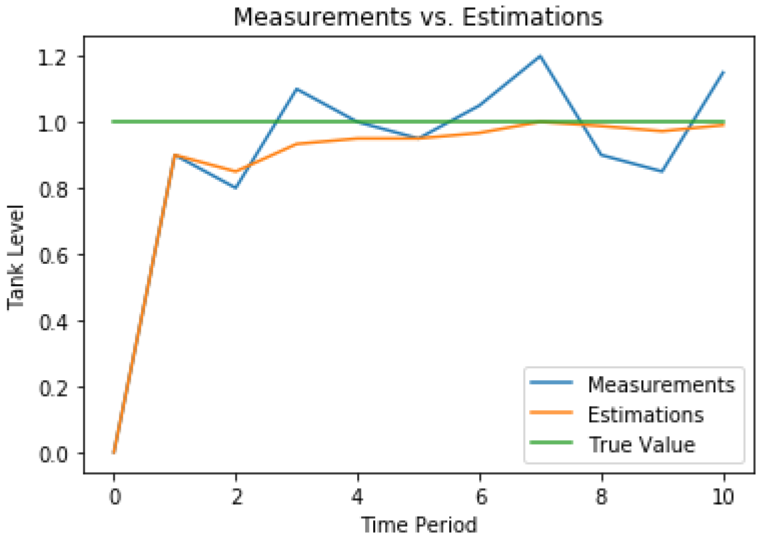

Figure 3.1 Static water level: measurements z_k
$\hat{x}_0 = 0$, $Q_k = 0.0001$, and $R_k = (\sqrt{0.1})^2 = 0.1$

a new set of noisier measurements. Consider new sensor readings

$$\tilde{z}_k = \{0.9772, 1.7265, 0.3919, 1.1781, 1.0465, 0.6284, 1.9333, 0.4858, 1.0296, 0.8379\}.$$

Table 3.2 and figure 3.2 summarize the computations with measurements $\tilde{z}_k$. The noisiness of the measurements becomes clear by examining the graph; these measurements range from 0.3919 to 1.9333. In this case, the state estimations are within 0.05 of the true value after eight iterations.

	Predict		Update			
t	$\hat{x}_k(-)$	$P_k(-)$	z_k	K_k	$\hat{x}_k(+)$	$P_k(+)$
0	-	-	-	-	0	1000.0001
1	0	1000.0001	0.9772	0.9999	0.9771	0.1
2	0.9771	0.1001	1.7265	0.5002	1.3520	0.05
3	1.3520	0.0501	0.3919	0.3339	1.0314	0.0334
4	1.0314	0.0335	1.1781	0.2509	1.0682	0.0251
5	1.0682	0.0252	1.0465	0.2012	1.0639	0.0201
6	1.0639	0.0202	0.6284	0.1682	0.9906	0.0168
7	0.9906	0.0169	1.9333	0.1447	1.1270	0.0145
8	1.1270	0.0146	0.4858	0.1272	1.0455	0.0127
9	1.0455	0.0128	1.0296	0.1136	1.0437	0.0114
10	1.0437	0.0115	0.8379	0.1028	1.0225	0.0103

Table 3.2 Computations for measurements $\tilde{z}_k$.
$\hat{x}_0 = 0$, $Q_k = 0.0001$, and $R_k = (\sqrt{0.1})^2 = 0.1$

For the next example we will need the following lemma.

Lemma 3.0.2. [15] Given continuous time state transition matrix A and continuous time state process matrix Q, the discrete time process noise matrix can be calculated as

$$Q(\delta t) = \int_0^{\delta t} e^{A\tau} Q e^{A^T \tau} d\tau$$

Example 3.0.3. This example is taken from [15]. Suppose instead that the tank in example 3.0.1 is being filled with water at a constant rate. If the level of the tank at time t is $L(t) = c$, then the rate at which the level is changing is $\frac{d}{dt}L(t) = r$, for some $r > 0$. Let the current water level estimate be $\hat{x}_L = L$ and the current water fill rate estimate be $\hat{x}_r = \frac{d\hat{x}_L}{dt}$. Now the overall state has two

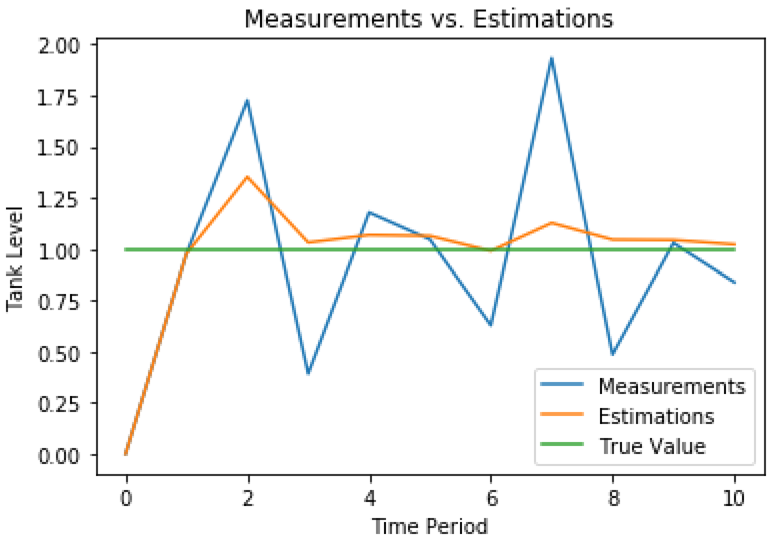

Figure 3.2 Static water level: measurements $\tilde{z}_k$
$\hat{x}_0 = 0$, $Q_k = 0.0001$, and $R_k = (\sqrt{0.1})^2 = 0.1$

components

$$\hat{x} = \begin{bmatrix} \hat{x}_L \\ \hat{x}_r \end{bmatrix}$$

In this case, the continuous time state transition matrix is

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

and the continuous time process noise matrix is

$$Q = \begin{bmatrix} 0 & 0 \\ 0 & q_f \end{bmatrix}$$

where q_f is the state process noise covariance. By lemma 3.0.2, the discrete time state transition matrix is

$$\Phi(\delta t) = \begin{bmatrix} 1 & \delta t \\ 0 & 1 \end{bmatrix}$$

and the discrete time process noise matrix is

$$Q(\delta t) = \begin{bmatrix} \frac{q_f \delta t^3}{3} & \frac{q_f \delta t^2}{2} \\ \frac{q_f \delta t^2}{2} & q_f \delta t \end{bmatrix}$$

Now we can express the filter update equations as the usual linear Kalman filter equations,

$$\begin{aligned} \hat{x}(+)(t+\delta t) &= \Phi(\delta T)\hat{x}(t) \\ P(+)(t+\delta t) &= \Phi(\delta T)P(+)\Phi^T(\delta T) + Q(\delta t). \end{aligned}$$

The floating sensor returns a noisy measurement z which represents the height of the float. In this case, the height of the float is proportional to the level of the water. That is,

$$L = \frac{z}{k_l}$$

where k_l is some *a priori* known constant scale factor. This relation tell us

$$z = Lk_l.$$

Using this form we can model the measurement estimates in general as a linear function of the system state $\hat{x}$,

$$\hat{z} = H\hat{x},$$

where H is the measurement matrix. The order of H will change depending on the dimension of our system. Assume that $\hat{x}_L$ is the first entry in the vector $\hat{x}$. If $\hat{x}$ is one-dimensional, then

$$H = k_l.$$

If $\hat{x}$ is two-dimensional, then

$$H = \begin{bmatrix} k_l & 0 \end{bmatrix}.$$

If $\hat{x}$ is three-dimensional, then

$$H = \begin{bmatrix} k_l & 0 & 0 \end{bmatrix}.$$

We can define H in a similar matter for states of higher dimension. The corresponding H will be used to calculate the Kalman gain K_k when implementing the actual Kalman filter equations.

Example 3.0.4. This example is adapted from [5]. Suppose we have a constant scalar x. We want to estimate the value of x given discrete measurements corrupted by an uncorrelated Gaussian noise with zero mean and covariance r. In this case, $\Phi_k = 1$ and $H_k = I = 1$. The scalar equations describing this system are

$$\begin{aligned} x_{k+1} &= x_k \\ z_k &= x_k + v_k \end{aligned}$$

where v_k is normally distributed with zero mean and covariance r. For this problem there is no system noise, hence $Q_k = 0$. We are interested in the effect the measurements have on the estimations of x. To investigate this we will deduce the equations of the Kalman filter and examine $\hat{x}_k(+)$. First we deduce the Kalman gain K_k

$$\begin{aligned} K_k &= \frac{P_k(-)H_k^T}{H_k P_k(-)H_k^T + R_k} \\ &= \frac{P_k(-) \cdot 1}{1 \cdot P_k(-) \cdot 1 + r} \\ &= \frac{P_k(-)}{P_k(-) + r} \end{aligned} \tag{3.0.5}$$

and the *a priori* state estimate error covariance $P_{k+1}(-)$

$$\begin{aligned} P_{k+1}(-) &= \Phi_k P_k(+)\Phi_k^T + Q_k \\ &= 1 \cdot P_k(+) \cdot 1 + 0 \\ &= P_k(+). \end{aligned} \tag{3.0.6}$$

We find the *a posteriori* state estimate error covariance using equations (3.0.5) and (3.0.6),

$$\begin{aligned} P_{k+1}(+) &= [I - K_{k+1}H_{k+1}]P_{k+1}(-) \\ &= \left[1 - \left(\frac{P_k(-)}{P_k(-) + r}\right) \cdot 1\right] P_{k+1}(-) \\ &= \left[\frac{P_k(+) + r - P_k(+)}{P_k(+) + r}\right] P_k(+) \\ &= \frac{rP_k(+)}{P_k(+) + r} \\ &= \frac{P_k(+)}{1 + \frac{P_k(+)}{r}} \end{aligned} \tag{3.0.7}$$

Equation (3.0.7) defines $P_{k+1}(+)$ in terms of it's previous time step $P_k(+)$, and therefore is a recurrence relation to which we can find a closed form solution [12].

Definition 3.0.8. [11] A *recurrence relation* is an equation that defines a sequence based on a rule that gives the next term as a function of the previous term(s).

Let $P_0(+) = P_0$. Then

$$P_1(+) = \frac{P_0(+)}{1 + \frac{P_0(+)}{r}} = \frac{P_0}{1 + \frac{P_0}{r}}$$

$$P_2(+) = \frac{P_1(+)}{1 + \frac{P_1(+)}{r}} = \frac{\frac{P_0}{1+\frac{P_0}{r}}}{1 + \frac{\frac{P_0}{1+\frac{P_0}{r}}}{r}} = \frac{rP_0}{r + 2P_0} = \frac{P_0}{1 + \frac{2P_0}{r}}$$

$$P_3(+) = \frac{P_2(+)}{1 + \frac{P_2(+)}{r}} = \frac{\frac{P_0}{1+\frac{2P_0}{r}}}{1 + \frac{P_0}{1+\frac{2P_0}{r}}} = \frac{rP_0}{r + 3P_0} = \frac{P_0}{1 + \frac{3P_0}{r}}$$

Continuing in a similar manner, we see

$$P_k(+) = \frac{P_0}{r + k\frac{P_0}{r}} \tag{3.0.9}$$

Now using equations (3.0.6) and (3.0.9), we can modify the Kalman gain from equation (3.0.5).

$$\begin{aligned} K_k &= \frac{P_k(-)}{P_k(-) + r} \\ &= \frac{P_{k-1}(+)}{P_{k-1}(+) + r} \\ &= \frac{\frac{P_0}{r+(k-1)\frac{P_0}{r}}}{\frac{P_0}{r+(k-1)\frac{P_0}{r}} + r} \\ &= \frac{P_0}{P_0 + r[1 + (k-1)\frac{P_0}{r}]} \\ &= \frac{P_0}{r + kP_0} \end{aligned} \tag{3.0.10}$$

Finally we can use equation (3.0.10) to express our *a posteriori* state estimate,

$$\begin{aligned} \hat{x}_k(+) &= \hat{x}_k(-) + K_k[z_k - H_k\hat{x}_k(-)] \\ &= \hat{x}_k(-) + \frac{P_0}{r + kP_0}[z_k - H_k\hat{x}_k(-)]. \end{aligned}$$

Naturally, we would like to have as many measurements as possible to produce an optimal state estimate. However, as $k \to \infty$ we see that $\frac{P_0}{r+kP_0}[z_k - H_k\hat{x}_k(-)] \to 0$. In this case, new measurements provide less and less information to the filter for making state estimates.

To illustrate this result we will estimate the value of a constant. First we choose a random scalar to estimate, $x = 0.7396307$. Note that we define this scalar as x and not $\hat{x}$ because it represents the true value we are trying to estimate. Assume that there is white noise of 0.1 corrupting the measurements, therefore the measurement noise is normally distributed about zero with a standard deviation of 0.1. With this standard deviation, from equation 2.1.19 we know that the ”true value” of our measurement noise covariance is $R = (0.1)^2 = 0.01$. Also assume that there is small process noise of $Q = 0.00001$. To initialize the filter we take our initial state estimate to be $x_0 = 0$ and our initial state estimate error covariance to be $P_0 = 1$. It is of note that the initial value of P_0 is not critical, as the Kalman filter will eventually converge with almost any non-zero choice [16]. Finally we simulate 50 distinct measurements z_k that have error normally distributed about zero with a standard deviation of 0.1; $z_k =$

{0.766277668, 0.814560689, 0.777369077, 0.702827878, 0.569886491,
0.652940024, 0.651269652, 0.614125042, 0.724566483, 0.880809683,
0.705239055, 0.738293019, 0.695943666, 0.741503529, 0.617636413,
0.606370548, 0.559460063, 0.695252835, 0.558329503, 0.741241213,
0.582532983, 0.792685733, 0.754692421, 0.665315396, 0.794927598,
0.712268683, 0.765142941, 0.825795361, 0.772658366, 0.794996244,
0.734940256, 0.850036573, 1.001966428, 0.905208076, 0.866572238,
0.636799795, 0.864499827, 0.606781973, 0.743481128, 0.610750397,
0.811586045, 0.779925983, 0.581047569, 0.750578152, 0.823447661,
0.751239373, 0.60483951, 0.547677686, 0.871661913, 0.551203349}

The results of the filter are shown in figure 3.3. The corresponding error covariances are graphed in figure 3.4. By the 50th iteration the covariances settle on the value of 0.0003.

One might be interested in the effect a higher covariance might have. Let us increase R from $R = 0.01$ to $R = 1$. The results are shown in figure 3.5, and the covariances in figure 3.6. The filter is slower to believe the measurements, and the covariances settle on a value of 0.0198, higher in comparison to when $R = 0.01$.

Now let us decrease R to $R = 0.0001$. These results are shown in figure 3.7. The filter is very quick to believe the measurements and ”trusts” them more through each iteration.

Instead of varying the error covariances R, let us now change our initial state estimate x_0. The previous examples have shown the results when the initial state estimate is lower than the true value. Let us now use an initial state estimate higher than the true value, $x_0 = 1$. As shown in

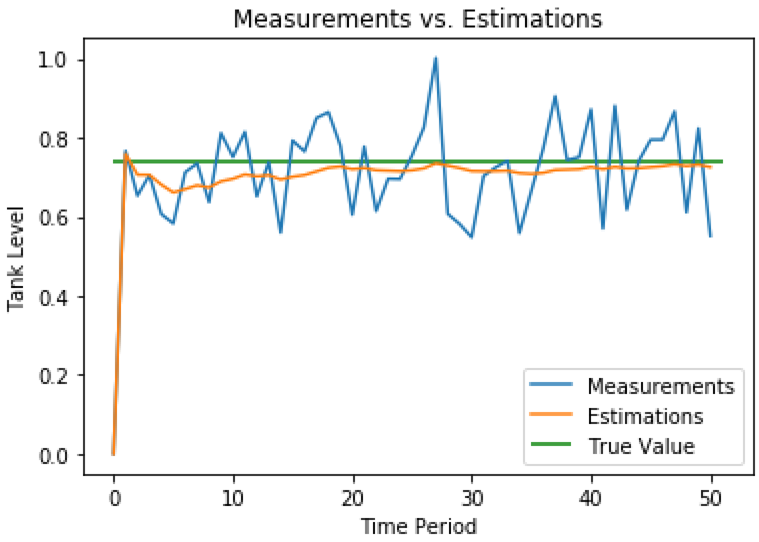

Figure 3.3 Estimations of a constant
$x_0 = 0$ and $R = (0.1)^2 = 0.01$

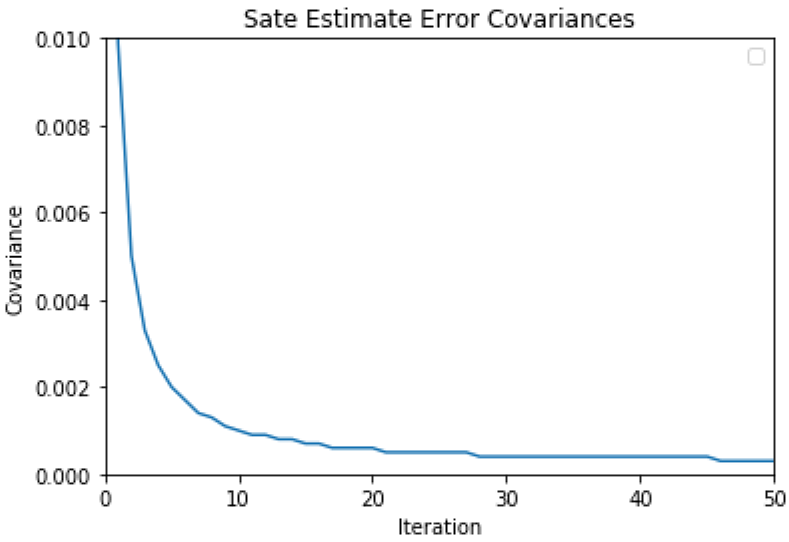

Figure 3.4 Covariances
$x_0 = 0$ and $R = (0.1)^2 = 0.01$

figure 3.8. In this case changing the initial state estimate has very little effect on the filter. Even further, the computations are nearly identical to our first simulation when $x_0 = 0$ and $R = 0.01$.

Example 3.0.11. Let us try to estimate the point $(1, 1)$ on the curve $f(x) = x^2$, ergo take the true value of the state to be $z = (1, 1)$. Suppose our initial state estimate is the point $z_0 = (x_0, y_0) = (0, 0)$, and our initial state estimate error covariance is $P_0 = 1$. Assume a small process

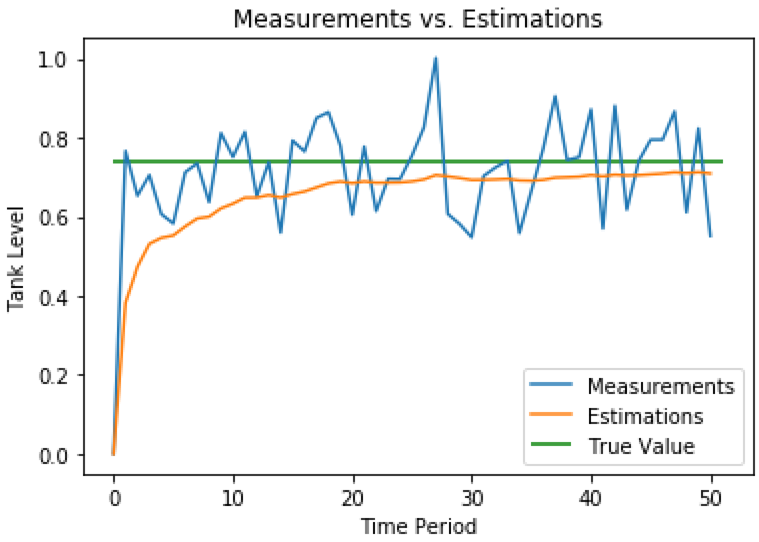

Figure 3.5 Estimations of a constant
$x_0 = 0$ and $R = 1^2 = 1$

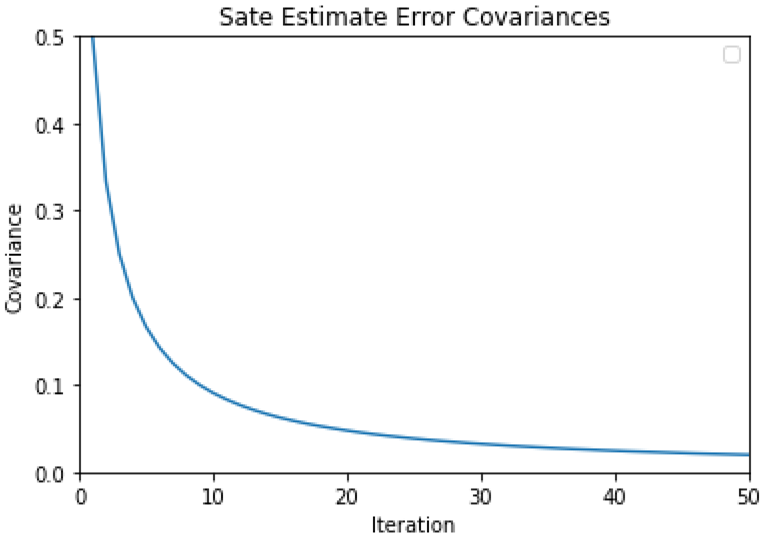

Figure 3.6 Covariances
$x_0 = 0$ and $R = 1^2 = 1$

noise of $Q_k = 0.0001$ for all k. In this case the state transition matrix is $\Phi_k = 1$ for all k, and the measurement transition matrix is $H_k = 1$ for all k. Assume that the measurement noise is normally distributed about zero with a standard deviation (white noise) of 0.1, so the measurement noise covariance is $R = (0.1)^2 = 0.01$. Now we simulate 10 measurements of points on the curve

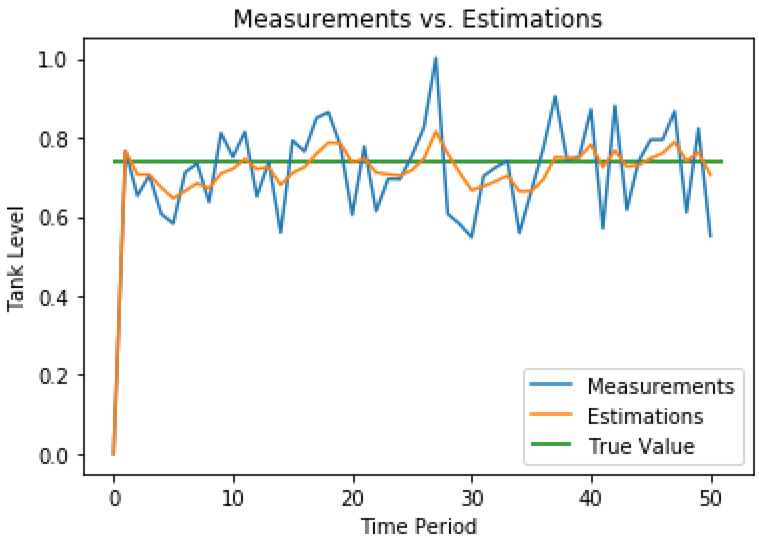

Figure 3.7 Estimations of a constant
$x_0 = 0$ and $R = (0.01)^2 = 0.0001$

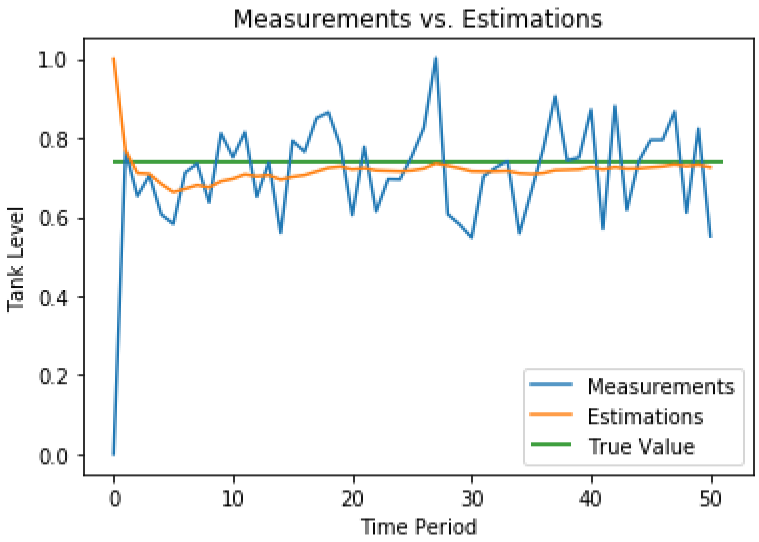

Figure 3.8 Estimations of a constant
$x_0 = 1$ and $R = (0.1)^2 = 0.01$

$f(x) = x^2$ on the interval $[0, 1]$.

$$z_k = \{(0.1055, -0.0672), (0.1315, 0.2028), (0.2261, 0.1184), (0.4077, 0.2106), (0.5952, 0.2953),$$
$$(0.6605, 0.3704), (0.7388, 0.6451), (0.9024, 0.4510), (0.9420, 0.9841), (1.0298, 1.0729)\}$$

Under these circumstances the Kalman filter produces the results in figure 3.9. Based off of these results, one might take the point $(0.8, 0.7)$ to be the optimal state estimate. Intuitively we know this is not accurate, so some adjustment must be made to better reflect our system. Generally, if the system model does not accurately model reality, then we can expect the Kalman filter to produce poor results. To combat this we can increase the process noise covariance which allows the filter to weigh the measurements more heavily throughout its iterations. If we increase Q to $Q = 0.1$, we obtain the results in figure 3.10. The optimal estimation produced here is much closer to the true value of $(1, 1)$, hence this system model should be preferred over the former.

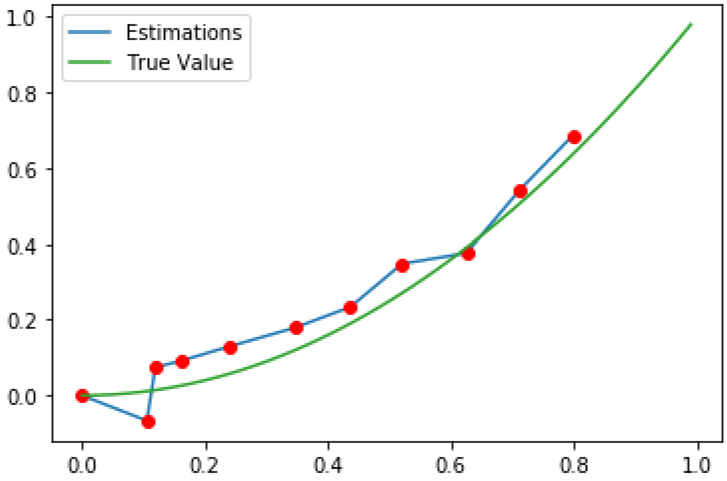

Figure 3.9 Estimations on $f(x) = x^2$ over $[0, 1]$
$R = (0.1)^2 = 0.01$ and $Q = 0.0001$

Let us now try to estimate the point $z = (5, 25)$ on the curve $f(x) = x^2$. In a similar manner as above, we take $z_0 = (x_0, y_0) = (0, 0)$, $P_0 = 1$, $Q_k = 0.0001$, $\Phi_k = 1$, $H_k = 1$, and $R = (0.1)^2 = 0.01$. Then we simulate 10 measurements of points on $f(x) = x^2$ on the interval $[0, 5]$,

$$\begin{aligned} z_k =&\{(0.4568, 0.3037), (0.9837, 0.9723), (1.5236, 2.2388), (1.7743, 3.9692), (2.3969, 6.3135), \\ &(2.8639, 8.8571), (3.5383, 12.1410), (3.9153, 16.0495), (4.3885, 20.1760), (5.0255, 24.9299)\}. \end{aligned}$$

The results under these circumstances is shown in figure 3.11. Based off of this one might take the optimal state estimate to be $(3.75, 16)$. Again, this is unrealistic. Increasing Q to $Q = 0.1$ so that

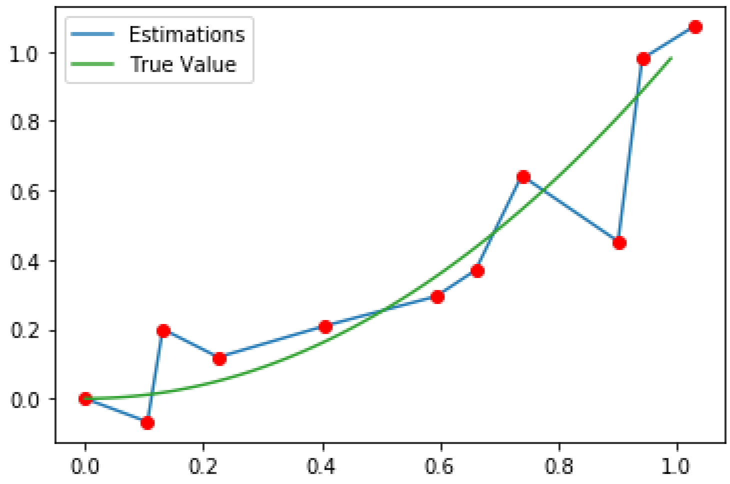

Figure 3.10 Estimations on $f(x) = x^2$ over $[0, 1]$
$R = (0.1)^2 = 0.01$ and $Q = 0.1$

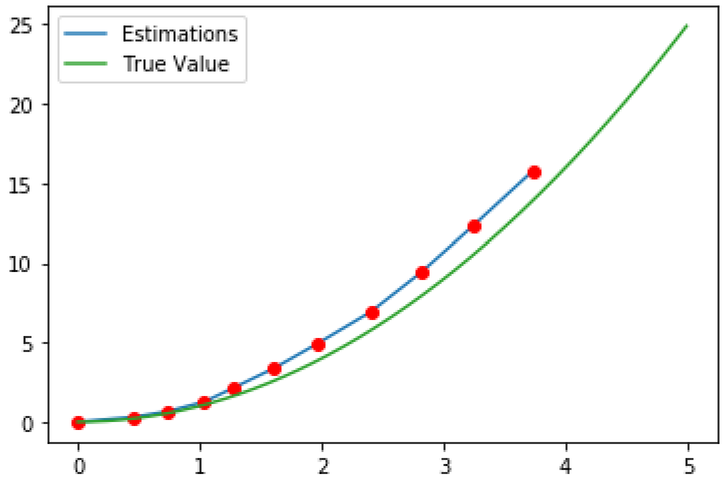

Figure 3.11 Estimations on $f(x) = x^2$ over $[0, 5]$
$R = (0.1)^2 = 0.01$ and $Q = 0.0001$

the filter places more trust the in measurements produces the results in figure 3.12. Clearly this model produces a more accurate optimal state estimate.

In the previous examples we saw how changing certain values effects the calculation of the optimal state estimate $\hat{x}_k$. The Kalman gain determines how heavily the measurements and *a priori* state estimates contribute to this calculation. If the measurement noise is small, then the measurements are trusted more than the *a priori* state estimates. If the *a priori* state estimate error covariance is small, then the *a priori* state estimates are trusted more than the measurements.

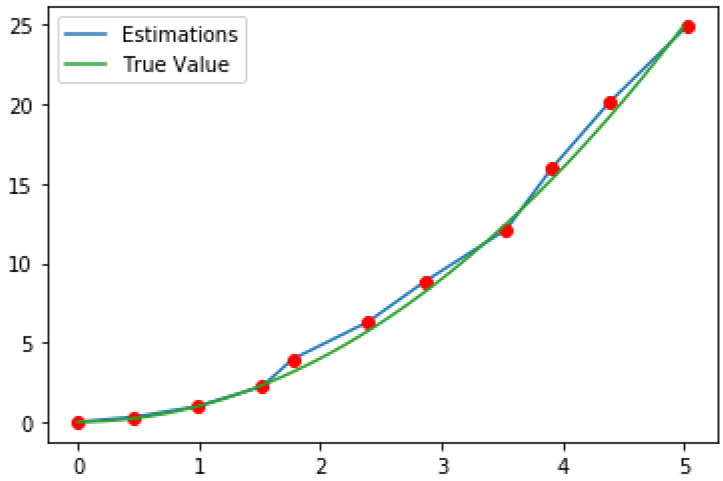

Figure 3.12 Estimations on $f(x) = x^2$ over $[0, 5]$
$R = (0.1)^2 = 0.01$ and $Q = 0.1$

We can demonstrate this analytically by considering two extreme cases: when $R_k = 0$ and when $P_k(-) = 0$. To produce the *a posteriori* state estimate $\hat{x}_k(+)$, the *a priori* state estimate $\hat{x}(-)$ is updated with the Kalman gain K_k. Therefore in each case we will examine the effect K_k has on $\hat{x}_k(+)$. In the former case, we have

$$\begin{aligned}\lim_{R_k \to 0} K_k &= \lim_{R_k \to 0} \frac{P_k(-)H_k^T}{H_k P_k(-) H_k^T + R_k} \\ &= \frac{P_k(-)H_k^T}{H_k P_k(-) H_k^T + 0} \\ &= \frac{1}{H_k}.\end{aligned}$$

Using this value of K_k to correct $\hat{x}_k(-)$ gives the *a posteriori* state estimate

$$\begin{aligned}\hat{x}_k(+) &= \hat{x}_k(-) + K_k[z_k - H_k \hat{x}_k(-)] \\ &= \hat{x}_k(-) + \frac{1}{H_k}[z_k - H_k \hat{x}_k(-)] \\ &= \hat{x}_k(-) + \frac{z_k}{H_k} - \hat{x}_k(-) \\ &= \frac{z_k}{H_k}.\end{aligned}$$

In this case we see that the calculation of $\hat{x}_k(+)$ comes mainly from measurement. In the latter case, we have

$$\begin{aligned}\lim_{P_k(-)\to 0} K_k &= \lim_{P_k(-)\to 0} \frac{P_k(-)H_k^T}{H_k P_k(-) H_k^T + R_k} \\ &= \frac{0 \cdot H_k^T}{H_k \cdot 0 \cdot H_k^T + R_k} \\ &= 0.\end{aligned}$$

Using this value of K_k gives

$$\begin{aligned}\hat{x}_k(+) &= \hat{x}_k(-) + K_k[z_k - H_k\hat{x}_k(-)] \\ &= \hat{x}_k(-) + 0 \cdot [z_k - H_k\hat{x}_k(-)] \\ &= \hat{x}_k(-).\end{aligned}$$

In this case the calculation of $\hat{x}_k(+)$ comes from the *a priori* state estimate. Of course it would be naive to expect all real life systems to fall so nicely into one of these extreme cases. Fortunately this weighting principle is embedded in the filter by design; the Kalman gain weighs the measurements and *a priori* state estimate errors appropriately to produce an optimal state estimate.

BIBLIOGRAPHY

[1] R.D Bucy and P.D. Joseph, *Filtering for Stochastic Processes with Applications to Guidance*, John Wiley & Sons, New York, 1968.

[2] Burkart, R., A., *Adaptable Slope Estimation Module and its Application in a Coolant Monitoring System for Predictive Observation*, Georgia Institute of Technology, August 2018.

[3] The Editors of Encyclopædia Britannica, *Cybernetics*, Encyclopædia Britannica, inc., https://www.britannica.com/science/cybernetics [accessed November 12, 2019].

[4] Gallager, R., Course Materials for 6.450 Principles of Digital Communications I, Fall 2006. MIT OpenCourseWare (http://ocw.mit.edu/), Massachusetts Institute of Technology. Downloaded on [27 November 2019].

[5] Gelb, A., Kasper, J.F., Nash, R.A., Price, C.F., and Sutherland, A.A., *Applied Optimal Estimation*, MIT Press, Cambridge, Massachusetts, 1974.

[6] Grewal, M. S., Andrews, A. P., *Kalman filtering: Theory and Practice using MATLAB®*, John Wiley & Sons, Inc., 3rd edition, 2008. authors: Mohinder S. Grewal and Angus P. Andrews title: Kalman Filtering Theory and Practice Using MATLAB®publisher: John Wiley & Sons, Inc. year: 2008 address: Hoboken, NJ

[7] Kalman, R., *A New Approach to Linear Filtering and Prediction Problems*" ASME- Journal of Basic Engineering, vol. 82, pp. 35–45, 1960.

[8] Lewis, F., Xie, L., Popa, D. (2008). *Optimal and Robust Estimation*. Boca Raton: CRC Press, https://doi.org/10.1201/9781315221656

[9] Maybeck, Peter S. 1979. *Stochastic Models, Estimation, and Control*, Volume 1, Academic Press, Inc.

[10] McGee, L. A. and Schmidt, S. F., *Discovery fo the Kalman Filter as a Practical Tool for Aerospace and Industry*, Technical Memorandum NASA-TM-86847, NASA - Ames Research Center, November 1985.

[11] Nykamp DQ, *Recurrence relation definition*. From Math Insight. http://mathinsight.org/definition/recurrence_relation

[12] Sonka, M., Hlava, V., and Boyle, R., *Image Processing, Analysis, and Machine Vision 4th Edition*, Cengage Learning, 200 First Stamford Place, 4th Floor Stamford, CT 06902, 2014.

[13] Stepanov, O.A., *Kalman Filtering: Past and Present. An outlook from Russia. (On the occasion of the 80th birthday of Rudolf Emil Kalman)*, Gyroscopy Navig. (2011) 2: 99. https://doi.org/10.1134/S2075108711020076

[14] Weisstein, Eric W. *Covariance Matrix*. From MathWorld–A Wolfram Web Resource. http://mathworld.wolfram.com/CovarianceMatrix.html

[15] Welch, G., *Team 18: The Kalman Filter Learning Tool Dynamic and Measurement Models*, http://www.cs.unc.edu/~welch/kalman/media/pdf/kftool_models.pdf (updated February 17, 2003).

[16] G. Welch and G. Bishop, *An Introduction to the Kalman filter*, Chapel Hill, NC, USA, Tech. Rep., 1995.

[17] Wolfram|Alpha, *Kalman Filter*, https://www.wolframalpha.com/input/?i=kalman%20filter&wal=header [accessed November 17, 2019].