Computer Vision

CPS 296.1 Supplementary Lecture Notes
Carlo Tomasi – Duke University
Fall 2007

7. The Kalman Filter

[Last Modified on November 13, 2007]

Warning. These notes are being written as the course progresses. As errors are discovered, old notes will be corrected and the new version will be posted online.

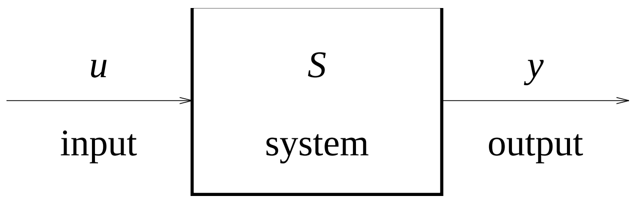

Figure 7.1: A general system.

7 The Kalman Filter

Perhaps the most important part of studying a problem in robotics or vision, as well as in most other sciences, is to determine a good model for the phenomena and events that are involved. For instance, studying manipulation requires defining models for how a robot arm can move and for how it interacts with the world. Analyzing image motion implies defining models for how points move in space and how this motion projects onto the image. When motion is involved, as is very often the case, models take on frequently the form of *dynamic systems*. A dynamic system is a mathematical description of a quantity that evolves over time. The theory of dynamic systems is both rich and fascinating. Although in this chapter we will barely scratch its surface, we will consider one of its most popular and useful aspects, the theory of state estimation, in the particular form of *Kalman filtering*. To this purpose, an informal definition of a dynamic system is given in the next section. The definition is then illustrated by setting up the dynamic system equations for a simple but realistic application, that of modeling the trajectory of an enemy mortar shell. In sections 7.3 through 7.5, we will develop the theory of the Kalman filter, and in section 7.6 we will see that the shell can be shot down before it hits us. As discussed in section 7.7, Kalman filtering has intimate connections with the theory of algebraic linear systems.

7.1 Dynamic Systems

In its most general meaning, the term *system* refers to some physical entity on which some action is performed by means of an input u. The system reacts to this input and produces an output y (see figure 7.1).

A *dynamic* system is a system whose phenomena occur over time. One often says that a system *evolves over time*. Simple examples of a dynamic system are the following:

- An electric circuit, whose input is the current in a given branch and whose output is a voltage across a pair of nodes.

- A chemical reactor, whose inputs are the external temperature, the temperature of the gas being supplied, and the supply rate of the gas. The output can be the temperature of the reaction product.

- A mass suspended from a spring. The input is the force applied to the mass and the output is the position of the mass.

In all these examples, what is input and what is output is a choice that depends on the application. Also, all the quantities in the examples vary continuously with time. In other cases, as for instance for switching networks and computers, it is more natural to consider time as a discrete variable. If time varies continuously, the system is said to be *continuous*; if time varies discretely, the system is said to be *discrete*.

7.1.1 State

Given a dynamic system, continuous or discrete, the modeling problem is to somehow correlate inputs (causes) with outputs (effects). The examples above suggest that the output at time t cannot be determined in general by the value assumed by the input quantity at the same point in time. Rather, the output is the result of the entire history of the system. An effort of abstraction is therefore required, which leads to postulating a new quantity, called the *state*, which summarizes information about the past and the present of the system. Specifically, the value $\mathbf{x}(t)$ taken by the state at time t must be sufficient to determine the output at the same point in time. Also, knowledge of both $\mathbf{x}(t_1)$ and $\mathbf{u}_{[t_1,t_2)}$, that is, of the state at time t_1 and the input over the interval $t_1 \leq t < t_2$, must allow computing the state (and hence the output) at time t_2. For the mass attached to a spring, for instance, the state could be the position and velocity of the mass. In fact, the laws of classical mechanics allow computing the new position and velocity of the mass at time t_2 given its position and velocity at time t_1 and the forces applied over the interval $[t_1, t_2)$. Furthermore, in this example, the output $\mathbf{y}$ of the system happens to coincide with one of the two state variables, and is therefore always deducible from the latter.

Thus, in a dynamic system the input affects the state, and the output is a function of the state. For a discrete system, the way that the input changes the state at time instant number k into the new state at time instant $k+1$ can be represented by a simple equation:

$$\mathbf{x}_{k+1} = f(\mathbf{x}_k, \mathbf{u}_k, k)$$

where f is some function that represents the change, and $\mathbf{u}_k$ is the input at time k. Similarly, the relation between state and output can be expressed by another function:

$$\mathbf{y}_k = h(\mathbf{x}_k, k) \ .$$

A *discrete dynamic system* is completely described by these two equations and an initial state $\mathbf{x}_0$. In general, all quantities are vectors.

For continuous systems, time does not come in quanta, so one cannot compute $\mathbf{x}_{k+1}$ as a function of $\mathbf{x}_k$, $\mathbf{u}_k$, and k, but rather compute $\mathbf{x}(t_2)$ as a functional ϕ of $\mathbf{x}(t_1)$ and the entire input $\mathbf{u}$ over the interval $[t_1, t_2)$:

$$\mathbf{x}(t_2) = \phi(\mathbf{x}(t_1), \mathbf{u}(\cdot), t_1, t_2)$$

where $\mathbf{u}(\cdot)$ represents the entire function $\mathbf{u}$, not just one of its values. A description of the system in terms of functions, rather than functionals, can be given in the case of a *regular system*, for which the functional ϕ is continuous, differentiable, and with continuous first derivative. In that case, one can show that there exists a function f such that the state $\mathbf{x}(t)$ of the system satisfies the differential equation

$$\dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t), t)$$

where the dot denotes differentiation with respect to time. The relation from state to output, on the other hand, is essentially the same as for the discrete case:

$$\mathbf{y}(t) = h(\mathbf{x}(t), t) \; .$$

Specifying the initial state $\mathbf{x}_0$ completes the definition of a continuous dynamic system.

7.1.2 Uncertainty

The systems defined in the previous section are called *deterministic*, since the evolution is exactly determined once the initial state $\mathbf{x}$ at time 0 is known. Determinism implies that both the evolution function f and the output function h are known exactly. This is, however, an unrealistic state of affairs. In practice, the laws that govern a given physical system are known up to some uncertainty. In fact, the equations themselves are simple abstractions of a complex reality. The coefficients that appear in the equations are known only approximately, and can change over time as a result of temperature changes, component wear, and so forth. A more realistic model then allows for some inherent, unresolvable uncertainty in both f and h. This uncertainty can be represented as *noise* that perturbs the equations we have presented so far. A discrete system then takes on the following form:

$$\begin{aligned} \mathbf{x}_{k+1} &= f(\mathbf{x}_k, \mathbf{u}_k, k) + \eta_k \\ \mathbf{y}_k &= h(\mathbf{x}_k, k) + \xi_k \end{aligned}$$

and for a continuous system

$$\begin{aligned} \dot{\mathbf{x}}(t) &= f(\mathbf{x}(t), \mathbf{u}(t), t) + \eta(t) \\ \mathbf{y}(t) &= h(\mathbf{x}(t), t) + \xi(t) \; . \end{aligned}$$

Without loss of generality, the noise distributions can be assumed to have zero mean, for otherwise the mean can be incorporated into the deterministic part, that is, in either f or h. The mean may not be known, but this is a different story: in general the parameters that enter into the definitions of f and h must be estimated by some method, and the mean perturbations are no different.

A common assumption, which is sometimes valid and always simplifies the mathematics, is that η and ξ are zero-mean Gaussian random variables with known covariance matrices Q and R, respectively.

7.1.3 Linearity

The mathematics becomes particularly simple when both the evolution function f and the output function h are linear. Then, the system equations become

$$\begin{aligned} \mathbf{x}_{k+1} &= F_k \mathbf{x}_k + G_k \mathbf{u}_k + \eta_k \\ \mathbf{y}_k &= H_k \mathbf{x}_k + \xi_k \end{aligned}$$

for the discrete case, and

$$\begin{aligned}\dot{\mathbf{x}}(t) &= F(t)\mathbf{x}(t) + G(t)\mathbf{u}(t) + \eta(t) \\ \mathbf{y}(t) &= H(t)\mathbf{x}(t) + \xi(t)\end{aligned}$$

for the continuous one. It is useful to specify the sizes of the matrices involved. We assume that the input $\mathbf{u}$ is a vector in $\mathcal{R}^p$, the state $\mathbf{x}$ is in $\mathcal{R}^n$, and the output $\mathbf{y}$ is in $\mathcal{R}^m$. Then, the *state propagation matrix* F is $n \times n$, the *input matrix* G is $n \times p$, and the *output matrix* H is $m \times n$. The covariance matrix Q of the *system noise* η is $n \times n$, and the covariance matrix of the output noise ξ is $m \times m$.

7.2 An Example: the Mortar Shell

In this section, the example of the mortar shell will be discussed in order to see some of the technical issues involved in setting up the equations of a dynamic system. In particular, we consider discretization issues because the physical system is itself continuous, but we choose to model it as a discrete system for easier implementation on a computer.

In sections 7.3 through 7.5, we consider the *state estimation* problem: given observations of the output $\mathbf{y}$ over an interval of time, we want to determine the state $\mathbf{x}$ of the system. This is a very important task. For instance, in the case of the mortar shell, the state is the (initially unknown) position and velocity of the shell, while the output is a set of observations made by a tracking system. Estimating the state then leads to enough knowledge about the shell to allow driving an antiaircraft gun to shoot the shell down in mid-flight.

You spotted an enemy mortar installation about thirty kilometers away, on a hill that looks about 0.5 kilometers higher than your own position. You want to track incoming projectiles with a Kalman filter so you can aim your guns accurately. You do not know the initial velocity of the projectiles, so you just guess some values: 0.6 kilometers/second for the horizontal component, 0.1 kilometers/second for the vertical component. Thus, your estimate of the initial state of the projectile is

$$\hat{\mathbf{x}}_0 = \begin{bmatrix} \dot{d} \\ d \\ \dot{z} \\ z \end{bmatrix} = \begin{bmatrix} -0.6 \\ 30 \\ 0.1 \\ 0.5 \end{bmatrix}$$

where d is the horizontal coordinate, z is the vertical, you are at $(0, 0)$, and dots denote derivatives with respect to time.

From your high-school physics, you remember that the laws of motion for a ballistic trajectory are the following:

$$d(t) = d(0) + \dot{d}(0)t \qquad (1)$$

$$z(t) = z(0) + \dot{z}(0)t - \frac{1}{2}gt^2 \qquad (2)$$

where g is the gravitational acceleration, equal to 9.8×10^{-3} kilometers per second squared. Since you do not trust your physics much, and you have little time to get ready, you decide to ignore air drag. Because of this, you introduce a state update covariance matrix $Q = 0.1I_4$, where I_4 is the 4×4 identity matrix.

All you have to track the shells is a camera pointed at the mortar that will rotate so as to keep the projectile at the center of the image, where you see a blob that increases in size as the projectile gets closer. Thus, the aiming angle of the camera gives you elevation information about the projectile's position, and the size of the blob tells you something about the distance, given that you know the actual size of the projectiles used and all the camera parameters. The projectile's elevation is

$$e = 1000\frac{z}{d} \tag{3}$$

when the projectile is at (d, z). Similarly, the size of the blob in pixels is

$$s = \frac{1000}{\sqrt{d^2 + z^2}} \ . \tag{4}$$

You do not have very precise estimates of the noise that corrupts e and s, so you guess measurement covariances $R_e = R_s = 1000$, which you put along the diagonal of a 2×2 diagonal measurement covariance matrix R.

7.2.1 The Dynamic System Equation

Equations (1) and (2) are continuous. Since you are taking measurements every $dt = 0.2$ seconds, you want to discretize these equations. For the z component, equation (2) yields

$$\begin{aligned} z(t+dt) - z(t) &= z(0) + \dot{z}(0)(t+dt) - \frac{1}{2}g(t+dt)^2 - \left[z(0) + \dot{z}(0)t - \frac{1}{2}gt^2\right] \\ &= (\dot{z}(0) - gt)dt - \frac{1}{2}g(dt)^2 \\ &= \dot{z}(t)dt - \frac{1}{2}g(dt)^2 \ , \end{aligned}$$

since $\dot{z}(0) - gt = \dot{z}(t)$.

Consequently, if $t + dt$ is time instant $k + 1$ and t is time instant k, you have

$$z_{k+1} = z_k + \dot{z}_k dt - \frac{1}{2}g(dt)^2 \ . \tag{5}$$

The reasoning for the horizontal component d is the same, except that there is no acceleration:

$$d_{k+1} = d_k + \dot{d}_k dt \ . \tag{6}$$

Equations (5) and (6) can be rewritten as a single system update equation

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

where

$$\mathbf{x}_k = \begin{bmatrix} \dot{d}_k \\ d_k \\ \dot{z}_k \\ z_k \end{bmatrix}$$

is the state, the 4×4 matrix F depends on dt, the control scalar u is equal to $-g$, and the 4×1 control matrix G depends on dt. The two matrices F and G are as follows:

$$F = \begin{bmatrix} 1 & 0 & 0 & 0 \\ dt & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & dt & 1 \end{bmatrix} \qquad G = \begin{bmatrix} 0 \\ 0 \\ dt \\ \frac{-dt^2}{2} \end{bmatrix} .$$

7.2.2 The Measurement Equation

The two nonlinear equations (3) and (4) express the available measurements as a function of the true values of the projectile coordinates d and z. We want to replace these equations with linear approximations. To this end, we develop both equations as Taylor series around the current estimate and truncate them after the linear term. From the elevation equation (3), we have

$$e_k = 1000\frac{z_k}{d_k} \approx 1000\left[\frac{\hat{z}_k}{\hat{d}_k} + \frac{z_k - \hat{z}_k}{\hat{d}_k} - \frac{\hat{z}_k}{\hat{d}_k^2}(d_k - \hat{d}_k)\right] ,$$

so that after simplifying we can redefine the measurement to be the discrepancy from the estimated value:

$$e'_k = e_k - 1000\frac{\hat{z}_k}{\hat{d}_k} \approx 1000(\frac{z_k}{\hat{d}_k} - \frac{\hat{z}_k}{\hat{d}_k^2}d_k) . \tag{7}$$

We can proceed similarly for equation (4):

$$s_k = \frac{1000}{\sqrt{d_k^2 + z_k^2}} \approx \frac{1000}{\sqrt{\hat{d}_k^2 + \hat{z}_k^2}} - \frac{1000\hat{d}_k}{(\hat{d}_k^2 + \hat{z}_k^2)^{3/2}}(d_k - \hat{d}_k) - \frac{1000\hat{z}_k}{(\hat{d}_k^2 + \hat{z}_k^2)^{3/2}}(z_k - \hat{z}_k)$$

and after simplifying:

$$s'_k = s_k - \frac{2000}{\sqrt{\hat{d}^2 + \hat{z}^2}} \approx -1000\left[\frac{\hat{d}_k}{(\hat{d}_k^2 + \hat{z}_k^2)^{3/2}}d_k + \frac{\hat{z}_k}{(\hat{d}_k^2 + \hat{z}_k^2)^{3/2}}z_k\right] . \tag{8}$$

The two measurements s'_k and e'_k just defined can be collected into a single measurement vector

$$\mathbf{y}_k = \begin{bmatrix} s'_k \\ e'_k \end{bmatrix} ,$$

and the two approximate measurement equations (7) and (8) can be written in the matrix form

$$\mathbf{y}_k = H_k\mathbf{x}_k \tag{9}$$

where the measurement matrix H_k depends on the current state estimate $\hat{\mathbf{x}}_k$:

$$H_k = -1000 \begin{bmatrix} 0 & \frac{\hat{d}_k}{\left(\hat{d}_k^2+\hat{z}_k^2\right)^{3/2}} & 0 & \frac{\hat{z}_k}{\left(\hat{d}_k^2+\hat{z}_k^2\right)^{3/2}} \\ 0 & \frac{\hat{z}_k}{\hat{d}_k^2} & 0 & -\frac{1}{\hat{d}_k} \end{bmatrix}$$

As the shell approaches us, we frantically start studying state estimation, and in particular Kalman filtering, in the hope to build a system that lets us shoot down the shell before it hits us. The next few sections will be read under this impending threat.

Knowing the model for the mortar shell amounts to knowing the laws by which the object moves and those that relate the position of the projectile to our observations. So what else is there left to do? From the observations, we would like to know where the mortar shell is right now, and perhaps predict where it will be in a few seconds, so we can direct an antiaircraft gun to shoot down the target. In other words, we want to know $\mathbf{x}_k$, the state of the dynamic system. Clearly, knowing $\mathbf{x}_0$ instead is equivalent, at least when the dynamics of the system are known exactly (the system noise η_k is zero). In fact, from $\mathbf{x}_0$ we can simulate the system up until time t, thereby determining $\mathbf{x}_k$ as well. Most importantly, we do not want to have all the observations before we shoot: we would be dead by then. A scheme that refines an initial estimation of the state as new observations are acquired is called a *recursive*[1] *state estimation* system. The *Kalman filter* is one of the most versatile schemes for recursive state estimations. The original paper by Kalman (R. E. Kalman, "A new approach to linear filtering and prediction problems," *Transactions of the ASME Journal Basic Engineering*, 82:34–45, 1960) is still one of the most readable treatments of this subject from the point of view of stochastic estimation.

Even without noise, a single observation $\mathbf{y}_k$ may not be sufficient to determine the state $\mathbf{x}_k$ (in the example, one observation happens to be sufficient). This is a very interesting aspect of state estimation. It is really the ensemble of all observations that let one estimate the state, and yet observations are processed one at a time, as they become available. A classical example of this situation in computer vision is the reconstruction of three-dimensional shape from a sequence of images. A single image is two-dimensional, so by itself it conveys no three-dimensional information. Kalman filters exist that recover shape information from a sequence of images. See for instance L. Matthies, T. Kanade, and R. Szeliski, "Kalman filter-based algorithms for estimating depth from image sequences," *International Journal of Computer Vision*, 3(3):209-236, September 1989; and T.J. Broida, S. Chandrashekhar, and R. Chellappa, "Recursive 3-D motion estimation from a monocular image sequence," *IEEE Transactions on Aerospace and Electronic Systems*, 26(4):639–656, July 1990.

Here, we introduce the Kalman filter from the simpler point of view of least squares estimation, since we have developed all the necessary tools in the first part of this course. The next section defines the state estimation problem for a discrete dynamic system in more detail. Then, section 7.4 defines the essential notions of estimation theory that are necessary to understand the quantitative aspects of Kalman filtering. Section 7.5 develops the equation of the Kalman filter, and section 7.6

[1]The term "recursive" in the systems theory literature corresponds loosely to "incremental" or "iterative" in computer science.

reconsiders the example of the mortar shell. Finally, section 7.7 establishes a connection between the Kalman filter and the solution of a linear system.

7.3 State Estimation

In this section, the estimation problem is defined in some more detail. Given a discrete dynamic system

$$\begin{aligned}\mathbf{x}_{k+1} &= F_k\mathbf{x}_k + G_k\mathbf{u}_k + \eta_k &(10)\\ \mathbf{y}_k &= H_k\mathbf{x}_k + \xi_k &(11)\end{aligned}$$

where the system noise η_k and the measurement noise ξ_k are Gaussian variables,

$$\begin{aligned}\eta_k &\sim \mathcal{N}(0, Q_k)\\ \xi_k &\sim \mathcal{N}(0, R_k)\ ,\end{aligned}$$

as well as a (possibly completely wrong) estimate $\hat{\mathbf{x}}_0$ of the initial state and an initial covariance matrix P_0 of the estimate $\hat{\mathbf{x}}_0$, the Kalman filter computes the optimal estimate $\hat{\mathbf{x}}_{k|k}$ at time k given the measurements $\mathbf{y}_0, \ldots, \mathbf{y}_k$. The filter also computes an estimate $P_{k|k}$ of the covariance of $\hat{\mathbf{x}}_{k|k}$ given those measurements. In these expressions, the hat means that the quantity is an estimate. Also, the first k in the subscript refers to which variable is being estimated, the second to which measurements are being used for the estimate. Thus, in general, $\hat{\mathbf{x}}_{i|j}$ is the estimate of the value that $\mathbf{x}$ assumes at time i given the first $j+1$ measurements $\mathbf{y}_0, \ldots, \mathbf{y}_j$.

7.3.1 Update

The covariance matrix $P_{k|k}$ must be computed in order to keep the Kalman filter running, in the following sense. At time k, just before the new measurement $\mathbf{y}_k$ comes in, we have an estimate $\hat{\mathbf{x}}_{k|k-1}$ of the state vector $\mathbf{x}_k$ based on the previous measurements $\mathbf{y}_0, \ldots, \mathbf{y}_{k-1}$. Now we face the problem of incorporating the new measurement $\mathbf{y}_k$ into our estimate, that is, of transforming $\hat{\mathbf{x}}_{k|k-1}$ into $\hat{\mathbf{x}}_{k|k}$. If $\hat{\mathbf{x}}_{k|k-1}$ were exact, we could *compute* the new measurement $\mathbf{y}_k$ without even looking at it, through the measurement equation (11). Even if $\hat{\mathbf{x}}_{k|k-1}$ is not exact, the estimate

$$\hat{\mathbf{y}}_{k|k-1} = H_k\hat{\mathbf{x}}_{k|k-1}$$

is still our best bet. Now $\mathbf{y}_k$ becomes available, and we can consider the *residue*

$$\mathbf{r}_k = \mathbf{y}_k - \hat{\mathbf{y}}_{k|k-1} = \mathbf{y}_k - H_k\hat{\mathbf{x}}_{k|k-1}\ .$$

If this residue is nonzero, we probably need to correct our estimate of the state $\mathbf{x}_k$, so that the new prediction

$$\hat{\mathbf{y}}_{k|k} = H_k\hat{\mathbf{x}}_{k|k}$$

of the measurement value is closer to the measurement $\mathbf{y}_k$ than the old prediction

$$\hat{\mathbf{y}}_{k|k-1} = H_k\hat{\mathbf{x}}_{k|k-1}$$

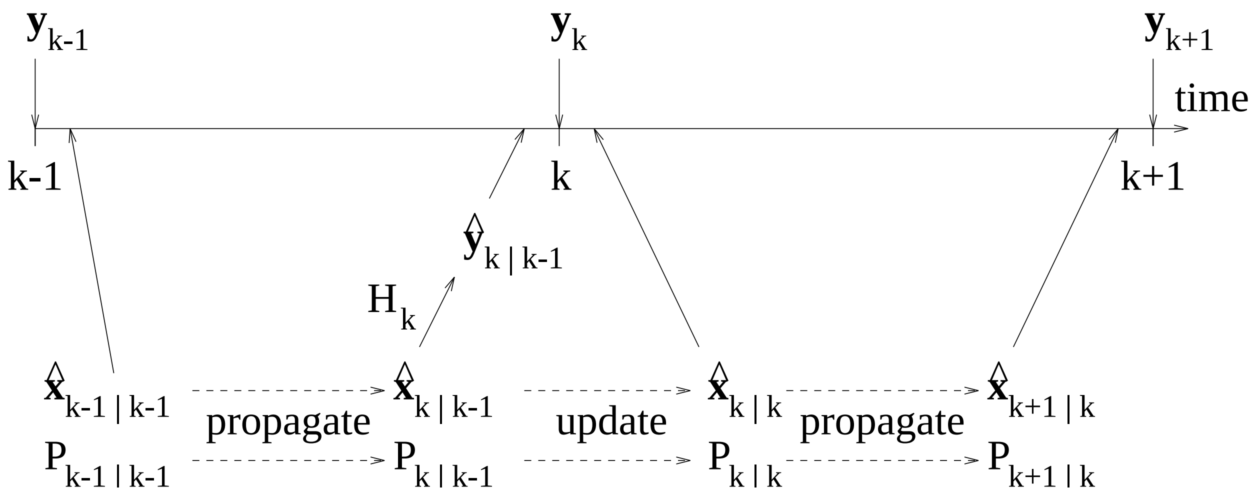

Figure 7.2: The update stage of the Kalman filter changes the estimate of the current system state $\mathbf{x}_k$ to make the prediction of the measurement closer to the actual measurement $\mathbf{y}_k$. Propagation then accounts for the evolution of the system state, as well as the consequent growing uncertainty.

that we made just before the new measurement $\mathbf{y}_k$ was available.

The question however is, by how much should we correct our estimate of the state? We do not want to make $\hat{\mathbf{y}}_{k|k}$ *coincide* with $\mathbf{y}_k$. That would mean that we trust the new measurement completely, but that we do not trust our state estimate $\hat{\mathbf{x}}_{k|k-1}$ at all, even if the latter was obtained through a large number of previous measurements. Thus, we need some criterion for comparing the quality of the new measurement $\mathbf{y}_k$ with that of our old estimate $\hat{\mathbf{x}}_{k|k-1}$ of the state. The uncertainty about the former is R_k, the covariance of the observation error. The uncertainty about the state just before the new measurement $\mathbf{y}_k$ becomes available is $P_{k|k-1}$. The *update* stage of the Kalman filter uses R_k and $P_{k|k-1}$ to weigh past evidence ($\hat{\mathbf{x}}_{k|k-1}$) and new observations ($\mathbf{y}_k$). This stage is represented graphically in the middle of figure 7.2. At the same time, also the uncertainty measure $P_{k|k-1}$ must be updated, so that it becomes available for the next step. Because a new measurement has been read, this uncertainty becomes usually smaller: $P_{k|k} < P_{k|k-1}$.

The idea is that as time goes by the uncertainty on the state decreases, while that about the measurements may remain the same. Then, measurements count less and less as the estimate approaches its true value.

7.3.2 Propagation

Just after arrival of the measurement $\mathbf{y}_k$, both state estimate and state covariance matrix have been updated as described above. But between time k and time $k+1$ both state and covariance may change. The state changes according to the system equation (10), so our estimate $\hat{\mathbf{x}}_{k+1|k}$ of $\mathbf{x}_{k+1}$ given $\mathbf{y}_0, \ldots, \mathbf{y}_k$ should reflect this change as well. Similarly, because of the system noise η_k, our uncertainty about this estimate may be somewhat greater than one time epoch ago. The system equation (10) essentially "dead reckons" the new state from the old, and inaccuracies in our model of how this happens lead to greater uncertainty. This increase in uncertainty depends on the system noise covariance Q_k. Thus, both state estimate and covariance must be *propagated* to the new time $k+1$ to yield the new state estimate $\hat{\mathbf{x}}_{k+1|k}$ and the new covariance $P_{k+1|k}$. Both these changes are shown on the right in figure 7.2.

In summary, just as the state vector $\mathbf{x}_k$ represents all the information necessary to describe the evolution of a deterministic system, the covariance matrix $P_{k|k}$ contains all the necessary information about the probabilistic part of the system, that is, about how both the system noise η_k and the measurement noise ξ_k corrupt the quality of the state estimate $\hat{\mathbf{x}}_{k|k}$.

Hopefully, this intuitive introduction to Kalman filtering gives you an idea of *what* the filter does, and what information it needs to keep working. To turn these concepts into a quantitative algorithm we need some preliminaries on optimal estimation, which are discussed in the next section. The Kalman filter itself is derived in section 7.5.

7.4 BLUE Estimators

In what sense does the Kalman filter use covariance information to produce better estimates of the state? As we will se later, the Kalman filter computes the *Best Linear Unbiased Estimate* (*BLUE*) of the state. In this section, we see what this means, starting with the definition of a linear estimation problem, and then considering the attributes "best" and "unbiased" in turn.

7.4.1 Linear Estimation

Given a quantity $\mathbf{y}$ (the *observation*) that is a known function of another (deterministic but unknown) quantity $\mathbf{x}$ (the *state*) plus some amount of noise,

$$\mathbf{y} = h(\mathbf{x}) + \mathbf{n} \; , \tag{12}$$

the estimation problem amounts to finding a function

$$\hat{\mathbf{x}} = \mathcal{L}(\mathbf{y})$$

such that $\hat{\mathbf{x}}$ is as close as possible to $\mathbf{x}$. The function $\mathcal{L}$ is called an *estimator*, and its value $\hat{\mathbf{x}}$ given the observations $\mathbf{y}$ is called an *estimate*. Inverting a function is an example of estimation. If the function h is invertible and the noise term $\mathbf{n}$ is zero, then $\mathcal{L}$ is the inverse of h, no matter how the phrase "as close as possible" is interpreted. In fact, in that case $\hat{\mathbf{x}}$ is equal to $\mathbf{x}$, and any distance between $\hat{\mathbf{x}}$ and $\mathbf{x}$ must be zero. In particular, solving a square, nonsingular system

$$\mathbf{y} = H\mathbf{x} \tag{13}$$

is, in this somewhat trivial sense, a problem of estimation. The optimal estimator is then represented by the matrix

$$L = H^{-1}$$

and the optimal estimate is

$$\hat{\mathbf{x}} = L\mathbf{y} \; .$$

A less trivial example occurs, for a linear observation function, when the matrix H has more rows than columns, so that the system (13) is overconstrained. In this case, there is usually no inverse to H, and again one must say in what sense $\hat{\mathbf{x}}$ is required to be "as close as possible" to $\mathbf{x}$. For linear systems, we have so far considered the criterion that prefers a particular $\hat{\mathbf{x}}$ if it makes the Euclidean norm of the vector $\mathbf{y} - H\mathbf{x}$ as small as possible. This is the *(unweighted) least squares* criterion. In section 7.4.2, we will see that in a very precise sense ordinary least squares solve a particular type of estimation problem, namely, the estimation problem for the observation equation (12) with h a linear function and $\mathbf{n}$ Gaussian zero-mean noise with the indentity matrix for covariance.

An estimator is said to be *linear* if the function $\mathcal{L}$ is linear. Notice that the observation function h can still be nonlinear. If $\mathcal{L}$ is required to be linear but h is not, we will probably have an estimator that produces a worse estimate than a nonlinear one. However, it still makes sense to look for the best possible linear estimator. The best estimator for a linear observation function happens to be a linear estimator.

7.4.2 Best

In order to define what is meant by a "best" estimator, one needs to define a measure of goodness of an estimate. In the least squares approach to solving a linear system like (13), this distance is defined as the Euclidean norm of the residue vector

$$\mathbf{y} - H\hat{\mathbf{x}}$$

between the left and the right-hand sides of equation (13), evaluated at the solution $\hat{\mathbf{x}}$. Replacing (13) by a "noisy equation",

$$\mathbf{y} = H\mathbf{x} + \mathbf{n} \tag{14}$$

does not change the nature of the problem. Even equation (13) has no exact solution when there are more independent equations than unknowns, so requiring equality is hopeless. What the least squares approach is really saying is that even at the solution $\hat{\mathbf{x}}$ there is some residue

$$\mathbf{n} = \mathbf{y} - H\hat{\mathbf{x}} \tag{15}$$

and we would like to make that residue as small as possible in the sense of the Euclidean norm. Thus, an overconstrained system of the form (13) and its "noisy" version (14) are really the same problem. In fact, (14) is the correct version, if the equality sign is to be taken literally.

The noise term, however, can be used to generalize the problem. In fact, the Euclidean norm of the residue (15) treats all components (all equations in (14)) equally. In other words, each equation counts the same when computing the norm of the residue. However, different equations can have noise terms of different variance. This amounts to saying that we have reasons to prefer the quality of some equations over others or, alternatively, that we want to enforce different equations to different degrees. From the point of view of least squares, this can be enforced by some scaling of the entries of $\mathbf{n}$ or, even, by some linear transformation of them:

$$\mathbf{n} \to W\mathbf{n}$$

so instead of minimizing $\|\mathbf{n}\|^2 = \mathbf{n}^T\mathbf{n}$ (the square is of course irrelevant when it comes to minimization), we now minimize

$$\|W\mathbf{n}\|^2 = \mathbf{n}^T R^{-1}\mathbf{n}$$

where

$$R^{-1} = W^T W$$

is a symmetric, nonnegative-definite matrix. This minimization problem, called *weighted least squares*, is only slightly different from its unweighted version. In fact, we have

$$W\mathbf{n} = W(\mathbf{y} - H\mathbf{x}) = W\mathbf{y} - WH\mathbf{x}$$

so we are simply solving the system

$$W\mathbf{y} = WH\mathbf{x}$$

in the traditional, "unweighted" sense. We know the solution from normal equations:

$$\hat{\mathbf{x}} = ((WH)^T WH)^{-1}(WH)^T W\mathbf{y} = (H^T R^{-1} H)^{-1} H^T R^{-1}\mathbf{y} \; .$$

Interestingly, this same solution is obtained from a completely different criterion of goodness of a solution $\hat{\mathbf{x}}$. This criterion is a probabilistic one. We consider this different approach because it will let us show that the Kalman filter is optimal in a very useful sense.

The new criterion is the so-called *minimum-covariance* criterion. The estimate $\hat{\mathbf{x}}$ of $\mathbf{x}$ is some function of the measurements $\mathbf{y}$, which in turn are corrupted by noise. Thus, $\hat{\mathbf{x}}$ is a function of a random vector (noise), and is therefore a random vector itself. Intuitively, if we estimate the same quantity many times, from measurements corrupted by different noise samples from the same distribution, we obtain different estimates. In this sense, the estimates are random.

It makes therefore sense to measure the quality of an estimator by requiring that its variance be as small as possible: the fluctuations of the estimate $\hat{\mathbf{x}}$ with respect to the true (unknown) value $\mathbf{x}$ from one estimation experiment to the next should be as small as possible. Formally, we want to choose a linear estimator L such that the estimates $\hat{\mathbf{x}} = L\mathbf{y}$ it produces minimize the following *covariance* matrix:

$$P = E[(\mathbf{x} - \hat{\mathbf{x}})(\mathbf{x} - \hat{\mathbf{x}})^T] \; .$$

Minimizing a matrix, however, requires a notion of "size" for matrices: how large is P? Fortunately, most interesting matrix norms are equivalent, in the sense that given two different definitions $\|P\|_1$ and $\|P\|_2$ of matrix norm there exist two positive scalars α, β such that

$$\alpha\|P\|_1 < \|P\|_2 < \beta\|P\|_1 \; .$$

Thus, we can pick any norm we like. In fact, in the derivations that follow, we only use properties shared by all norms, so which norm we actually use is irrelevant.

7.4.3 Unbiased

In additionto requiring our estimator to be linear and with minimum covariance, we also want it to be *unbiased*, in the sense that if repeat the same estimation experiment many times we neither consistently overestimate nor consistently underestimate $\mathbf{x}$. Mathematically, this translates into the following requirement:

$$E[\mathbf{x} - \hat{\mathbf{x}}] = 0 \quad \text{and} \quad E[\hat{\mathbf{x}}] = E[\mathbf{x}] \; .$$

7.4.4 The BLUE

We now address the problem of finding the Best Linear Unbiased Estimator (BLUE)

$$\hat{\mathbf{x}} = L\mathbf{y}$$

of $\mathbf{x}$ given that $\mathbf{y}$ depends on $\mathbf{x}$ according to the model (14), which is repeated here for convenience:

$$\mathbf{y} = H\mathbf{x} + \mathbf{n} \; . \tag{16}$$

First, we give a necessary and sufficient condition for L to be unbiased.

Lemma 7.1 *Let $\mathbf{n}$ in equation (16) be zero mean. Then the linear estimator L is unbiased if an only if*

$$LH = I \; ,$$

the identity matrix.

Proof.

$$\begin{aligned} E[\mathbf{x} - \hat{\mathbf{x}}] &= E[\mathbf{x} - L\mathbf{y}] = E[\mathbf{x} - L(H\mathbf{x} + \mathbf{n})] \\ &= E[(I - LH)\mathbf{x}] - E[L\mathbf{n}] = (I - HL)E[\mathbf{x}] \end{aligned}$$

since $E[L\mathbf{n}] = L\,E[\mathbf{n}]$ and $E[\mathbf{n}] = 0$. For this to hold for all $\mathbf{x}$ we need $I - LH = 0$. Δ

And now the main result.

Theorem 7.2 *The Best Linear Unbiased Estimator (BLUE)*

$$\hat{\mathbf{x}} = L\mathbf{y}$$

for the measurement model

$$\mathbf{y} = H\mathbf{x} + \mathbf{n}$$

where the noise vector $\mathbf{n}$ has zero mean and covariance R is given by

$$L = (H^T R^{-1} H)^{-1} H^T R^{-1}$$

and the covariance of the estimate $\hat{\mathbf{x}}$ is

$$P = E[(\mathbf{x} - \hat{\mathbf{x}})(\mathbf{x} - \hat{\mathbf{x}})^T] = (H^T R^{-1} H)^{-1} \; . \tag{17}$$

Proof. We can write

$$\begin{aligned} P &= E[(\mathbf{x}-\hat{\mathbf{x}})(\mathbf{x}-\hat{\mathbf{x}})^T] = E[(\mathbf{x}-L\mathbf{y})(\mathbf{x}-L\mathbf{y})^T] \\ &= E[(\mathbf{x}-LH\mathbf{x}-L\mathbf{n})(\mathbf{x}-LH\mathbf{x}-L\mathbf{n})^T] = E[((I-LH)\mathbf{x}-L\mathbf{n})((I-LH)\mathbf{x}-L\mathbf{n})^T] \\ &= E[L\mathbf{n}\mathbf{n}^T L^T] = L\,E[\mathbf{n}\mathbf{n}^T]\,L^T = LRL^T \end{aligned}$$

because L is unbiased, so that $LH = I$.

To show that

$$L_0 = (H^T R^{-1} H)^{-1} H^T R^{-1} \tag{18}$$

is the best choice, let L be any (other) linear unbiased estimator. We can trivially write

$$L = L_0 + (L - L_0)$$

and

$$\begin{aligned} P &= LRL^T = [L_0 + (L-L_0)]R[L_0 + (L-L_0)]^T \\ &= L_0 R L_0^T + (L-L_0)RL_0^T + L_0 R(L-L_0)^T + (L-L_0)R(L-L_0)^T\ . \end{aligned}$$

From (18) we obtain

$$RL_0^T = RR^{-1}H(H^TR^{-1}H)^{-1} = H(H^TR^{-1}H)^{-1}$$

so that

$$(L-L_0)RL_0^T = (L-L_0)H(H^TR^{-1}H)^{-1} = (LH - L_0H)(H^TR^{-1}H)^{-1}\ .$$

But L and L_0 are unbiased, so $LH = L_0H = I$, and

$$(L-L_0)RL_0^T = 0\ .$$

The term $L_0R(L-L_0)^T$ is the transpose of this, so it is zero as well. In conclusion,

$$P = L_0RL_0^T + (L-L_0)R(L-L_0)^T\ ,$$

the sum of two positive definite or at least semidefinite matrices. For such matrices, the norm of the sum is greater or equal to either norm, so this expression is minimized when the second term vanishes, that is, when $L = L_0$.

This proves that the estimator given by (18) is the best, that is, that it has minimum covariance. To prove that the covariance P of $\hat{\mathbf{x}}$ is given by equation (17), we simply substitute L_0 for L in $P = LRL^T$:

$$\begin{aligned} P &= L_0RL_0^T = (H^TR^{-1}H)^{-1}H^TR^{-1}RR^{-1}H(H^TR^{-1}H)^{-1} \\ &= (H^TR^{-1}H)^{-1}H^TR^{-1}H(H^TR^{-1}H)^{-1} = (H^TR^{-1}H)^{-1} \end{aligned}$$

as promised. Δ

7.5 The Kalman Filter: Derivation

We now have all the components necessary to write the equations for the Kalman filter. To summarize, given a linear measurement equation

$$\mathbf{y} = H\mathbf{x} + \mathbf{n}$$

where $\mathbf{n}$ is a Gaussian random vector with zero mean and covariance matrix R,

$$\mathbf{n} \sim \mathcal{N}(0, R) \ ,$$

the best linear unbiased estimate $\hat{\mathbf{x}}$ of $\mathbf{x}$ is

$$\hat{\mathbf{x}} = PH^T R^{-1}\mathbf{y}$$

where the matrix

$$P \stackrel{\Delta}{=} E[(\hat{\mathbf{x}} - \mathbf{x})(\hat{\mathbf{x}} - \mathbf{x})^T] = (H^T R^{-1} H)^{-1}$$

is the covariance of the estimation error.

Given a dynamic system with system and measurement equations

$$\begin{aligned} \mathbf{x}_{k+1} &= F_k \mathbf{x}_k + G_k \mathbf{u}_k + \eta_k \\ \mathbf{y}_k &= H_k \mathbf{x}_k + \xi_k \end{aligned} \tag{19}$$

where the system noise η_k and the measurement noise ξ_k are Gaussian random vectors,

$$\begin{aligned} \eta_k &\sim \mathcal{N}(0, Q_k) \\ \xi_k &\sim \mathcal{N}(0, R_k) \ , \end{aligned}$$

as well as the best, linear, unbiased estimate $\hat{\mathbf{x}}_0$ of the initial state with an error covariance matrix P_0, the Kalman filter computes the best, linear, unbiased estimate $\hat{\mathbf{x}}_{k|k}$ at time k given the measurements $\mathbf{y}_0, \ldots, \mathbf{y}_k$. The filter also computes the covariance $P_{k|k}$ of the error $\hat{\mathbf{x}}_{k|k} - \mathbf{x}_k$ given those measurements. Computation occurs according to the phases of update and propagation illustrated in figure 7.2. We now apply the results from optimal estimation to the problem of updating and propagating the state estimates and their error covariances.

7.5.1 Update

At time k, two pieces of data are available. One is the estimate $\hat{\mathbf{x}}_{k|k-1}$ of the state $\mathbf{x}_k$ given measurements up to but not including $\mathbf{y}_k$. This estimate comes with its covariance matrix $P_{k|k-1}$. Another way of saying this is that the estimate $\hat{\mathbf{x}}_{k|k-1}$ differs from the true state $\mathbf{x}_k$ by an error term $\mathbf{e}_k$ whose covariance is $P_{k|k-1}$:

$$\hat{\mathbf{x}}_{k|k-1} = \mathbf{x}_k + \mathbf{e}_k \tag{20}$$

with

$$E[\mathbf{e}_k \mathbf{e}_k^T] = P_{k|k-1} \ .$$

The other piece of data is the new measurement $\mathbf{y}_k$ itself, which is related to the state $\mathbf{x}_k$ by the equation

$$\mathbf{y}_k = H_k \mathbf{x}_k + \xi_k \tag{21}$$

with error covariance

$$E[\xi_k \xi_k^T] = R_k \; .$$

We can summarize this available information by grouping equations 20 and 21 into one, and packaging the error covariances into a single, block-diagonal matrix. Thus, we have

$$\mathbf{y} = H \mathbf{x}_k + \mathbf{n}$$

where

$$\mathbf{y} = \begin{bmatrix} \hat{\mathbf{x}}_{k|k-1} \\ \mathbf{y}_k \end{bmatrix} , \qquad H = \begin{bmatrix} I \\ H_k \end{bmatrix} , \qquad \mathbf{n} = \begin{bmatrix} \mathbf{e}_k \\ \xi_k \end{bmatrix} ,$$

and where $\mathbf{n}$ has covariance

$$R = \begin{bmatrix} P_{k|k-1} & 0 \\ 0 & R_k \end{bmatrix} .$$

As we know, the solution to this classical estimation problem is

$$\begin{aligned} \hat{\mathbf{x}}_{k|k} &= P_{k|k} H^T R^{-1} \mathbf{y} \\ P_{k|k} &= (H^T R^{-1} H)^{-1} \; . \end{aligned}$$

This pair of equations represents the update stage of the Kalman filter. These expressions are somewhat wasteful, because the matrices H and R contain many zeros. For this reason, these two update equations are now rewritten in a more efficient and more familiar form. We have

$$\begin{aligned} P_{k|k}^{-1} &= H^T R^{-1} H \\ &= \begin{bmatrix} I & H_k^T \end{bmatrix} \begin{bmatrix} P_{k|k-1}^{-1} & 0 \\ 0 & R_k^{-1} \end{bmatrix} \begin{bmatrix} I \\ H_k \end{bmatrix} \\ &= P_{k|k-1}^{-1} + H_k^T R_k^{-1} H_k \end{aligned}$$

and

$$\begin{aligned} \hat{\mathbf{x}}_{k|k} &= P_{k|k} H^T R^{-1} \mathbf{y} \\ &= P_{k|k} \begin{bmatrix} P_{k|k-1}^{-1} & H_k^T R_k^{-1} \end{bmatrix} \begin{bmatrix} \hat{\mathbf{x}}_{k|k-1} \\ \mathbf{y}_k \end{bmatrix} \\ &= P_{k|k} (P_{k|k-1}^{-1} \hat{\mathbf{x}}_{k|k-1} + H_k^T R_k^{-1} \mathbf{y}_k) \\ &= P_{k|k} ((P_{k|k}^{-1} - H_k^T R_k^{-1} H_k) \hat{\mathbf{x}}_{k|k-1} + H_k^T R_k^{-1} \mathbf{y}_k) \\ &= \hat{\mathbf{x}}_{k|k-1} + P_{k|k} H_k^T R_k^{-1} (\mathbf{y}_k - H_k \hat{\mathbf{x}}_{k|k-1}) \; . \end{aligned}$$

In the last line, the difference

$$\mathbf{r}_k \stackrel{\Delta}{=} \mathbf{y}_k - H_k \hat{\mathbf{x}}_{k|k-1}$$

is the *residue* between the actual measurement $\mathbf{y}_k$ and its best estimate based on $\hat{\mathbf{x}}_{k|k-1}$, and the matrix

$$K_k \triangleq P_{k|k} H_k^T R_k^{-1}$$

is usually referred to as the *Kalman gain* matrix, because it specifies the amount by which the residue must be multiplied (or amplified) to obtain the correction term that transforms the old estimate $\hat{\mathbf{x}}_{k|k-1}$ of the state $\mathbf{x}_k$ into its new estimate $\hat{\mathbf{x}}_{k|k}$.

7.5.2 Propagation

Propagation is even simpler. Since the new state is related to the old through the system equation 19, and the noise term η_k is zero mean, unbiasedness requires

$$\hat{\mathbf{x}}_{k+1|k} = F_k \hat{\mathbf{x}}_{k|k} + G_k \mathbf{u}_k \ ,$$

which is the state estimate propagation equation of the Kalman filter. The error covariance matrix is easily propagated thanks to the linearity of the expectation operator:

$$\begin{aligned} P_{k+1|k} &= E[(\hat{\mathbf{x}}_{k+1|k} - \mathbf{x}_{k+1})(\hat{\mathbf{x}}_{k+1|k} - \mathbf{x}_{k+1})^T] \\ &= E[(F_k(\hat{\mathbf{x}}_{k|k} - \mathbf{x}_k) - \eta_k)(F_k(\hat{\mathbf{x}}_{k|k} - \mathbf{x}_k) - \eta_k)^T] \\ &= F_k E[(\hat{\mathbf{x}}_{k|k} - \mathbf{x}_k)(\hat{\mathbf{x}}_{k|k} - \mathbf{x}_k)^T] F_k^T + E[\eta_k \eta_k^T] \\ &= F_k P_{k|k} F_k^T + Q_k \end{aligned}$$

where the system noise η_k and the previous estimation error $\hat{\mathbf{x}}_{k|k} - \mathbf{x}_k$ were assumed to be uncorrelated.

7.5.3 Kalman Filter Equations

In summary, the Kalman filter evolves an initial estimate and an initial error covariance matrix,

$$\hat{\mathbf{x}}_{0|-1} \triangleq \hat{\mathbf{x}}_0 \quad \text{and } P_{0|-1} \triangleq P_0 \ ,$$

both assumed to be given, by the update equations

$$\begin{aligned} \hat{\mathbf{x}}_{k|k} &= \hat{\mathbf{x}}_{k|k-1} + K_k(\mathbf{y}_k - H_k \hat{\mathbf{x}}_{k|k-1}) \\ P_{k|k}^{-1} &= P_{k|k-1}^{-1} + H_k^T R_k^{-1} H_k \end{aligned}$$

where the Kalman gain is defined as

$$K_k = P_{k|k} H_k^T R_k^{-1}$$

and by the propagation equations

$$\begin{aligned} \hat{\mathbf{x}}_{k+1|k} &= F_k \hat{\mathbf{x}}_{k|k} + G_k \mathbf{u}_k \\ P_{k+1|k} &= F_k P_{k|k} F_k^T + Q_k \ . \end{aligned}$$

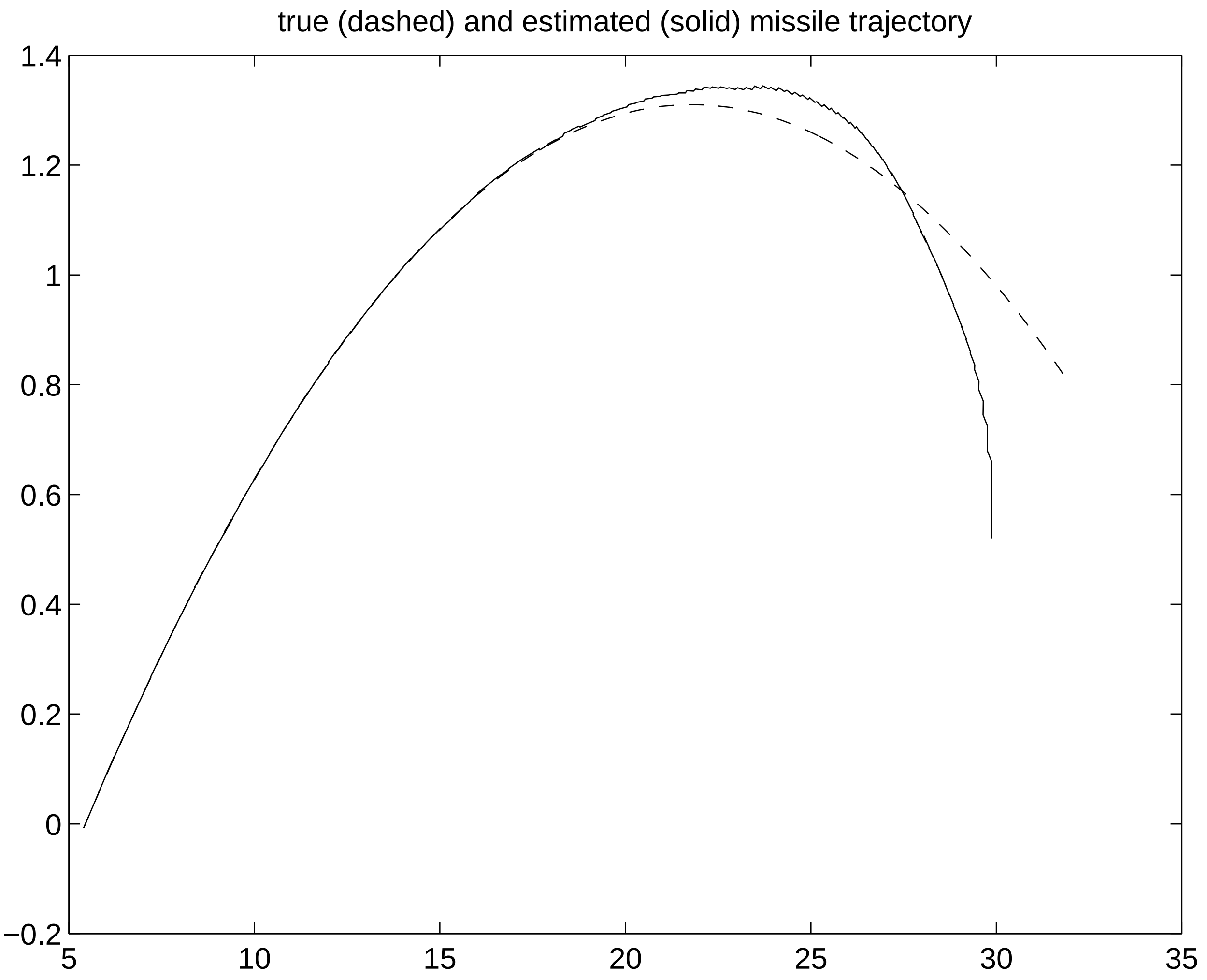

Figure 7.3: The true and estimated trajectories get closer to one another. Trajectories start on the right.

7.6 Results of the Mortar Shell Experiment

In section 7.2, the dynamic system equations for a mortar shell were set up. Matlab routines available through the class Web page implement a Kalman filter (with naive numerics) to estimate the state of that system from simulated observations. Figure 7.3 shows the true and estimated trajectories. Notice that coincidence of the trajectories does not imply that the state estimate is up-to-date. For this it is also necessary that any given point of the trajectory is reached by the estimate at the same time instant. Figure 7.4 shows that the distance between estimated and true target position does indeed converge to zero, and this occurs in time for the shell to be shot down. Figure 7.5 shows the 2-norm of the covariance matrix over time. Notice that the covariance goes to zero only asymptotically.

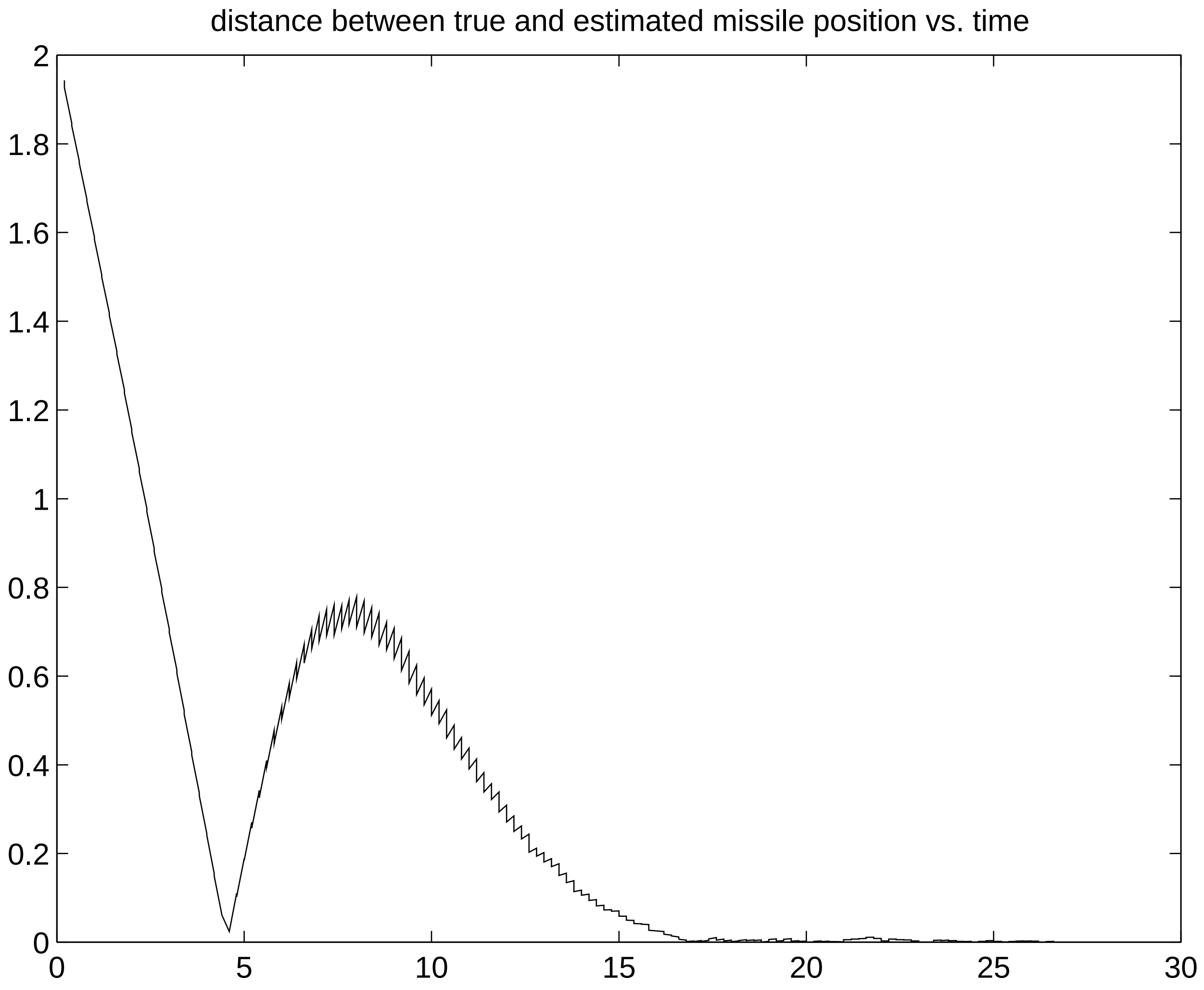

Figure 7.4: The estimate actually closes in towards the target.

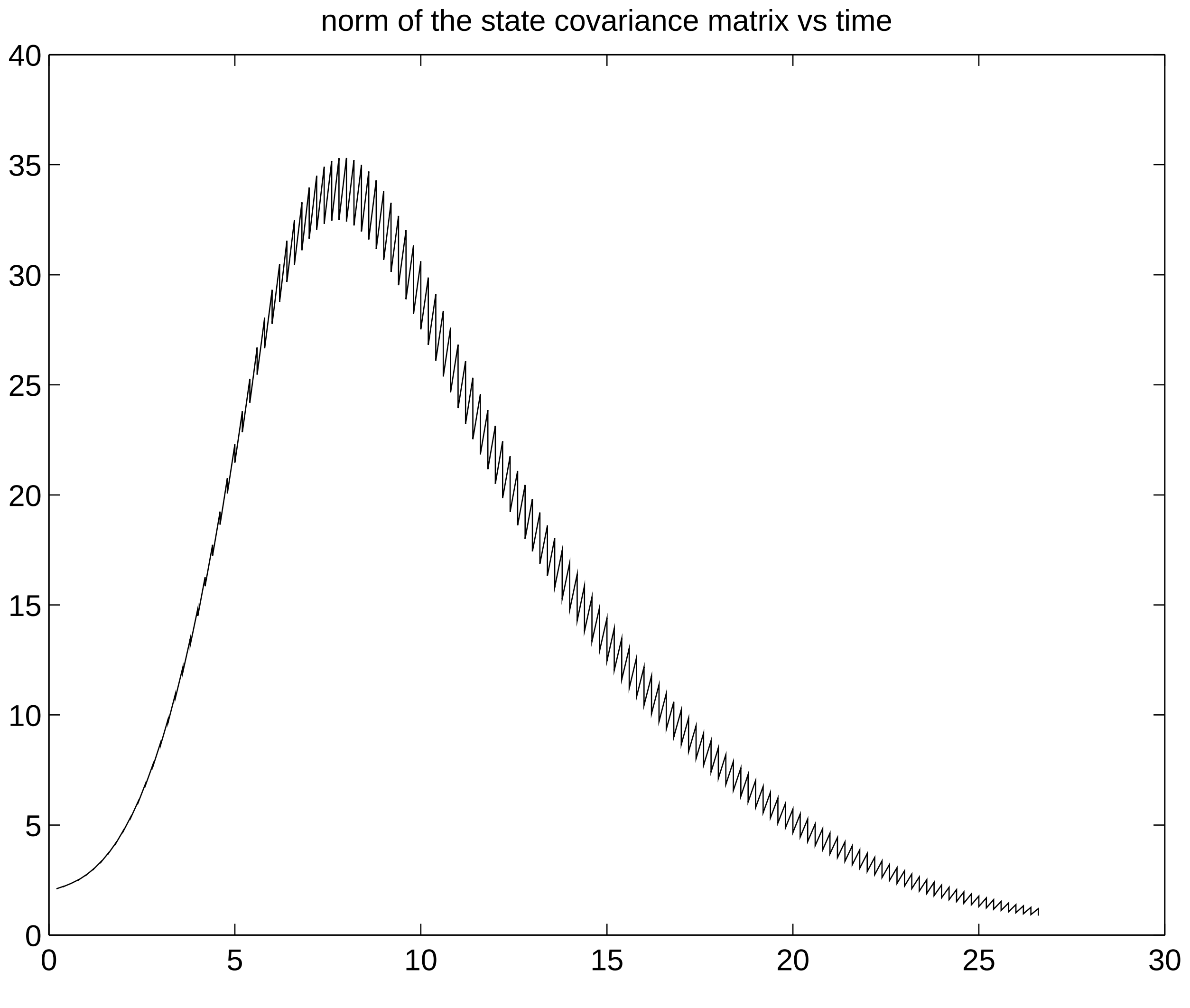

Figure 7.5: After an initial increase in uncertainty, the norm of the state covariance matrix converges to zero. Upwards segments correspond to state propagation, downwards ones to state update.

7.7 Linear Systems and the Kalman Filter

In order to connect the theory of state estimation with what we have learned so far about linear systems, we now show that estimating the initial state $\mathbf{x}_0$ from the first $k+1$ measurements, that is, obtaining $\hat{\mathbf{x}}_{0|k}$, amounts to solving a linear system of equations with suitable weights for its rows.

The basic recurrence equations (10) and (11) can be expanded as follows:

$$\begin{aligned}
\mathbf{y}_k &= H_k\mathbf{x}_k + \xi_k = H_k(F_{k-1}\mathbf{x}_{k-1} + G_{k-1}\mathbf{u}_{k-1} + \eta_{k-1}) + \xi_k \\
&= H_kF_{k-1}\mathbf{x}_{k-1} + H_kG_{k-1}\mathbf{u}_{k-1} + H_k\eta_{k-1} + \xi_k \\
&= H_kF_{k-1}(F_{k-2}\mathbf{x}_{k-2} + G_{k-2}\mathbf{u}_{k-2} + \eta_{k-2}) + H_kG_{k-1}\mathbf{u}_{k-1} + H_k\eta_{k-1} + \xi_k \\
&= H_kF_{k-1}F_{k-2}\mathbf{x}_{k-2} + H_k(F_{k-1}G_{k-2}\mathbf{u}_{k-2} + G_{k-1}\mathbf{u}_{k-1}) + \\
&\quad\; H_k(F_{k-1}\eta_{k-2} + \eta_{k-1}) + \xi_k \\
&\;\vdots \\
&= H_kF_{k-1}\ldots F_0\mathbf{x}_0 + H_k(F_{k-1}\ldots F_1G_0\mathbf{u}_0 + \ldots + G_{k-1}\mathbf{u}_{k-1}) + \\
&\quad\; H_k(F_{k-1}\ldots F_1\eta_0 + \ldots + \eta_{k-1}) + \xi_k
\end{aligned}$$

or in a more compact form,

$$\mathbf{y}_k = H_k\Phi(k-1,0)\mathbf{x}_0 + H_k\sum_{j=1}^{k}\Phi(k-1,j)G_{j-1}\mathbf{u}_{j-1} + \nu_k \tag{22}$$

where

$$\Phi(l,j) = \begin{cases} F_l\ldots F_j & \text{for } l \geq j \\ 1 & \text{for } l < j \end{cases}$$

and the term

$$\nu_k = H_k\sum_{j=1}^{k}\Phi(k-1,j)\eta_{j-1} + \xi_k$$

is noise.

The key thing to notice about this somewhat intimidating expression is that for any k it is a linear system in $\mathbf{x}_0$, the initial state of the system. We can write one system like the one in equation (22) for every value of $k = 0,\ldots,K$, where K is the last time instant considered, and we obtain a large system of the form

$$\mathbf{z}_K = \Psi_K\mathbf{x}_0 + \mathbf{g}_K + \mathbf{n}_K \tag{23}$$

where

$$
\begin{aligned}
\mathbf{z}_K &= \begin{bmatrix} \mathbf{y}_0 \\ \vdots \\ \mathbf{y}_K \end{bmatrix} \\
\Psi_K &= \begin{bmatrix} H_0 \\ H_1 F_0 \\ \vdots \\ H_K \Phi(K-1,0) \end{bmatrix} \\
\mathbf{g}_K &= \begin{bmatrix} 0 \\ H_1 G_0 \mathbf{u}_0 \\ \vdots \\ H_K(\Phi(K-1,1)G_0\mathbf{u}_0 + \ldots + \Phi(K-1,K)G_{K-1}\mathbf{u}_{K-1}) \end{bmatrix} \\
\mathbf{n}_K &= \begin{bmatrix} \nu_0 \\ \vdots \\ \nu_K \end{bmatrix} .
\end{aligned}
$$

Without knowing anything about the statistics of the noise vector $\mathbf{n}_K$ in equation (23), the best we can do is to solve the system

$$\mathbf{z}_K = \Psi_K \mathbf{x}_0 + \mathbf{g}_K$$

in the sense of least squares, to obtain an estimate of $\mathbf{x}_0$ from the measurements $\mathbf{y}_0, \ldots, \mathbf{y}_K$:

$$\hat{\mathbf{x}}_{0|K} = \Psi_K^{\dagger}(\mathbf{z}_K - \mathbf{g}_K)$$

where $\Psi_K^{\dagger}$ is the pseudoinverse of Ψ_K. We know that if Ψ_K has full rank, the result with the pseudoinverse is the same as we would obtain by solving the normal equations, so that

$$\Psi_K^{\dagger} = (\Psi_K^T \Psi_K)^{-1} \Psi_K^T .$$

The least square solution to system (23) minimizes the residue between the left and the right-hand side under the assumption that all equations are to be treated the same way. This is equivalent to assuming that all the noise terms in $\mathbf{n}_K$ are equally important. However, we know the covariance matrices of all these noise terms, so we ought to be able to do better, and weigh each equation to keep these covariances into account. Intuitively, a small covariance means that we believe in that measurement, and therefore in that equation, which should consequently be weighed more heavily than others. The quantitative embodiment of this intuitive idea is at the core of the Kalman filter.

In summary, the Kalman filter for a linear system has been shown to be equivalent to a linear equation solver, under the assumption that the noise that affects each of the equations has the same probability distribution, that is, that all the noise terms in $\mathbf{n}_K$ in equation 23 are equally important. However, the Kalman filter differs from a linear solver in the following important respects:

1. The noise terms in $\mathbf{n}_K$ in equation 23 are *not* equally important. Measurements come with covariance matrices, and the Kalman filter makes optimal use of this information for a proper weighting of each of the scalar equations in (23). Better information ought to yield more accurate results, and this is in fact the case.

2. The system (23) is not solved all at once. Rather, an initial solution is refined over time as new measurements become available. The final solution can be proven to be exactly equal to solving system (23) all at once. However, having better and better approximations to the solution as new data come in is much preferable in a dynamic setting, where one cannot in general wait for all the data to be collected. In some applications, data my never stop arriving.

3. A solution for the estimate $\hat{\mathbf{x}}_{k|k}$ of the current state is given, and not only for the estimate $\hat{\mathbf{x}}_{0|k}$ of the initial state. As time goes by, knowledge of the initial state may obsolesce and become less and less useful. The Kalman filter computes up-to-date information about the current state.