THE LINEAR KALMAN FILTER

4.1: Introduction

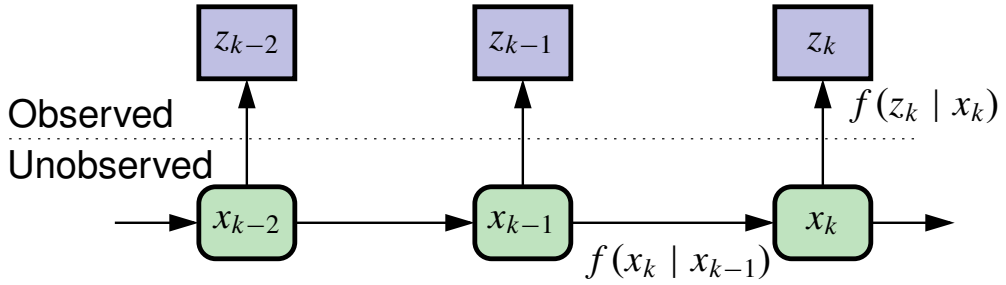
- The principal goal of this course is to learn how to estimate the present hidden state (vector) value of a dynamic system, using noisy measurements that are somehow related to that state (vector).
- We assume a general, possibly nonlinear, model

$$x_k = f_{k-1}(x_{k-1}, u_{k-1}, w_{k-1})$$

$$z_k = h_k(x_k, u_k, v_k),$$

where u_k is a known (deterministic/measured) input signal, w_k is a process-noise random input, and v_k is a sensor-noise random input.

SEQUENTIAL PROBABILISTIC INFERENCE: Estimate the present state x_k of a dynamic system using all measurements $\mathbb{Z}_k = \{z_0, z_1, \cdots, z_k\}$.



- This notes chapter provides a unified theoretic framework to develop a family of estimators for this task: particle filters, Kalman filters, extended Kalman filters, sigma-point (unscented) Kalman filters. . .

A smattering of estimation theory

- There are various approaches to "optimal estimation" of some unknown quantity x.
- One says that we would like to minimize the expected magnitude (length) of the error vector between x and the estimate $\hat{x}$.

$$\hat{x}^{\text{MME}} = \arg\min_{\hat{x}} \left(\mathbb{E}\left[\left\| x - \hat{x} \right\| \mid \mathbb{Z} \right] \right).$$

- This turns out to be the median of the *a posteriori* pdf $f(x \mid \mathbb{Z})$.
- A similar result, but easier to derive analytically minimizes the expected length squared of that error vector.
- This is the minimum mean square error (MMSE) estimator

$$\begin{aligned}
\hat{x}^{\text{MMSE}}(\mathbb{Z}) &= \arg\min_{\hat{x}} \left(\mathbb{E}\left[\left\| x - \hat{x} \right\|_2^2 \mid \mathbb{Z} \right] \right) \\
&= \arg\min_{\hat{x}} \left(\mathbb{E}\left[(x - \hat{x})^T (x - \hat{x}) \mid \mathbb{Z} \right] \right) \\
&= \arg\min_{\hat{x}} \left(\mathbb{E}\left[x^T x - 2x^T \hat{x} + \hat{x}^T \hat{x} \mid \mathbb{Z} \right] \right).
\end{aligned}$$

- We solve for $\hat{x}$ by differentiating the cost function and setting the result to zero

$$\begin{aligned}
0 &= \frac{\mathrm{d}}{\mathrm{d}\hat{x}} \mathbb{E}\left[x^T x - 2x^T \hat{x} + \hat{x}^T \hat{x} \mid \mathbb{Z} \right] \\
&= \mathbb{E}\left[-2(x - \hat{x}) \mid \mathbb{Z} \right] \qquad \text{(trust me)} \\
&= 2\hat{x} - 2\mathbb{E}\left[x \mid \mathbb{Z} \right] \\
\hat{x} &= \mathbb{E}\left[x \mid \mathbb{Z} \right].
\end{aligned}$$

- Another approach to estimation is to optimize a likelihood function

$$\hat{x}^{\text{ML}}(\mathbb{Z}) = \arg\max_{x} f(\mathbb{Z} \mid x).$$

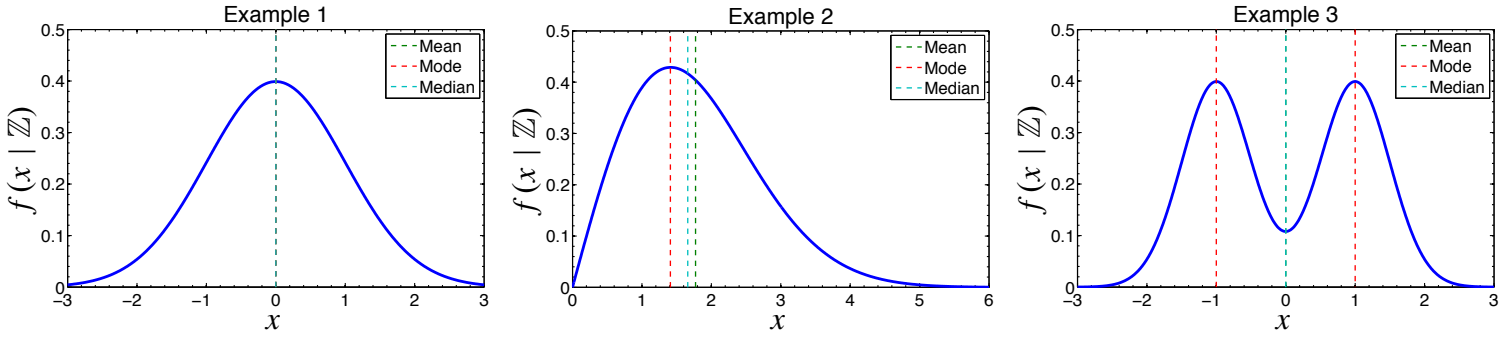
- Yet a fourth is the maximum *a posteriori* estimate

$$\hat{x}^{\text{MAP}}(\mathbb{Z}) = \arg\max_x f(x \mid \mathbb{Z}) = \arg\max_x f(\mathbb{Z} \mid x) f(x).$$

- In general, $\hat{x}^{\text{MME}} \neq \hat{x}^{\text{MMSE}} \neq \hat{x}^{\text{ML}} \neq \hat{x}^{\text{MAP}}$, so which is "best"?
- Answer: It probably depends on the application.
- The text gives some metrics for comparison: bias, MSE, etc.
- Here, we use $\hat{x}^{\text{MMSE}} = \mathbb{E}[x \mid \mathbb{Z}]$ because it "makes sense" and works well in a lot of applications and is mathematically tractable.
- Also, for the assumptions that we will make, $\hat{x}^{\text{MME}} = \hat{x}^{\text{MMSE}} = \hat{x}^{\text{MAP}}$ but $\hat{x}^{\text{MMSE}} \neq \hat{x}^{\text{ML}}$. $\hat{x}^{\text{MMSE}}$ is unbiased and has smaller MSE than $\hat{x}^{\text{ML}}$.

Some examples

- In example 1, mean, median, and mode are identical. Any of these statistics would make a good estimator of x.
- In example 2, mean, median, and mode are all different. Which to choose is not necessarily obvious.
- In example 3, the distribution is multi-modal. None of the estimates is likely to be satisfactory!



4.2: Developing the framework

- The Kalman filter applies the MMSE estimation criteria to a dynamic system. That is, our state estimate is the conditional mean

$$\mathbb{E}\big[x_k \mid \mathbb{Z}_k\big] = \int_{R_{x_k}} x_k f(x_k \mid \mathbb{Z}_k)\,\mathrm{d}x_k,$$

  where R_{x_k} is the set comprising the range of possible x_k.

- To make progress toward implementing this estimator, we must break $f(x_k \mid \mathbb{Z}_k)$ into simpler pieces.

- We first use Bayes' rule to write:

$$f(x_k \mid \mathbb{Z}_k) = \frac{f(\mathbb{Z}_k \mid x_k) f(x_k)}{f(\mathbb{Z}_k)}.$$

- We then break up $\mathbb{Z}_k$ into smaller constituent parts within the joint probabilities as $\mathbb{Z}_{k-1}$ and z_k

$$\frac{f(\mathbb{Z}_k \mid x_k) f(x_k)}{f(\mathbb{Z}_k)} = \frac{f(z_k, \mathbb{Z}_{k-1} \mid x_k) f(x_k)}{f(z_k, \mathbb{Z}_{k-1})}.$$

- Thirdly, we use the joint probability rule $f(a,b) = f(a \mid b) f(b)$ on the numerator and denominator terms

$$\frac{f(z_k, \mathbb{Z}_{k-1} \mid x_k) f(x_k)}{f(z_k, \mathbb{Z}_{k-1})} = \frac{f(z_k \mid \mathbb{Z}_{k-1}, x_k) f(\mathbb{Z}_{k-1} \mid x_k) f(x_k)}{f(z_k \mid \mathbb{Z}_{k-1}) f(\mathbb{Z}_{k-1})}.$$

- Next, we apply Bayes' rule once again in the terms within the [ ]

$$\frac{f(z_k \mid \mathbb{Z}_{k-1}, x_k) \big[f(\mathbb{Z}_{k-1} \mid x_k)\big] f(x_k)}{f(z_k \mid \mathbb{Z}_{k-1}) f(\mathbb{Z}_{k-1})}$$

$$= \frac{f(z_k \mid \mathbb{Z}_{k-1}, x_k) \big[f(x_k \mid \mathbb{Z}_{k-1}) f(\mathbb{Z}_{k-1})\big] f(x_k)}{f(z_k \mid \mathbb{Z}_{k-1}) f(\mathbb{Z}_{k-1}) \big[f(x_k)\big]}.$$

- We now cancel some terms from numerator and denominator

$$\frac{f(z_k \mid \mathbb{Z}_{k-1}, x_k) f(x_k \mid \mathbb{Z}_{k-1}) f(\mathbb{Z}_{k-1}) f(x_k)}{f(z_k \mid \mathbb{Z}_{k-1}) f(\mathbb{Z}_{k-1}) f(x_k)} = \frac{f(z_k \mid \mathbb{Z}_{k-1}, x_k) f(x_k \mid \mathbb{Z}_{k-1})}{f(z_k \mid \mathbb{Z}_{k-1})}.$$

- Finally, recognize that z_k is conditionally independent of $\mathbb{Z}_{k-1}$ given x_k

$$\frac{f(z_k \mid \mathbb{Z}_{k-1}, x_k) f(x_k \mid \mathbb{Z}_{k-1})}{f(z_k \mid \mathbb{Z}_{k-1})} = \frac{f(z_k \mid x_k) f(x_k \mid \mathbb{Z}_{k-1})}{f(z_k \mid \mathbb{Z}_{k-1})}.$$

- So, overall, we have shown that

$$f(x_k \mid \mathbb{Z}_k) = \frac{f(z_k \mid x_k) f(x_k \mid \mathbb{Z}_{k-1})}{f(z_k \mid \mathbb{Z}_{k-1})}.$$

KEY POINT #1: This shows that we can compute the desired density recursively with two steps per iteration:

- The first step computes probability densities for predicting x_k given all past observations

$$f(x_k \mid \mathbb{Z}_{k-1}) = \int_{R_{x_{k-1}}} f(x_k \mid x_{k-1}) f(x_{k-1} \mid \mathbb{Z}_{k-1}) \, \mathrm{d}x_{k-1}.$$

- The second step updates the prediction via

$$f(x_k \mid \mathbb{Z}_k) = \frac{f(z_k \mid x_k) f(x_k \mid \mathbb{Z}_{k-1})}{f(z_k \mid \mathbb{Z}_{k-1})}.$$

- Therefore, the general sequential inference solution breaks naturally into a prediction/update scenario.

- To proceed further using this approach, the relevant probability densities may be computed as

$$f(z_k \mid \mathbb{Z}_{k-1}) = \int_{R_{x_k}} f(z_k \mid x_k) f(x_k \mid \mathbb{Z}_{k-1}) \, \mathrm{d}x_k$$

$$f(x_k \mid x_{k-1}) = \int_{R_{w_k}} \delta(x_k - f(x_{k-1}, u_{k-1}, w_{k-1})) f(w_{k-1}) \, \mathrm{d}w_{k-1}$$

$$= \sum_{\{w_{k-1} \,:\, x_k = f(x_{k-1}, u_{k-1}, w_{k-1})\}} f(w_{k-1})$$

$$f(z_k \mid x_k) = \int_{R_{v_k}} \delta(z_k - h(x_k, u_k, v_k)) f(v_k) \, \mathrm{d}v_k$$
$$= \sum_{\{v_k \,:\, z_k = h(x_k, u_k, v_k)\}} f(v_k).$$

KEY POINT #2: Closed-form solutions to solving the multi-dimensional integrals is intractable for most real-world systems.

- For applications that justify the computational expense, the integrals may be approximated using Monte Carlo methods (particle filters).
- But, besides applications using particle filters, this approach appears to be a dead end.

KEY POINT #3: A simplified solution may be obtained if we are willing to make the *assumption* that all probability densities are *Gaussian*.

- This is the basis of the original Kalman filter, the extended Kalman filter, and the sigma-point (unscented) Kalman filters to be discussed.

4.3: Setting up The Gaussian assumption

- In the next two sections, our goal is to compute the MMSE estimator $\hat{x}_k^+ = \mathbb{E}\big[x_k \mid \mathbb{Z}_k\big]$ under the assumption that all densities are Gaussian.
  - We will find that the result is the same natural predict/update mechanism of sequential inference.
  - Note that although the math on the following pages gets a little rough in places, the end result is "quite simple."
- So, with malice aforethought, we define prediction error $\tilde{x}_k^- = x_k - \hat{x}_k^-$ where $\hat{x}_k^- = \mathbb{E}\big[x_k \mid \mathbb{Z}_{k-1}\big]$.
  - Error is always computed as "truth" minus "prediction," or as "truth" minus "estimate."
  - We can never compute the error in practice, since the truth value is not known.
  - However, we can still prove statistical results using this definition that give an algorithm for estimating the truth using known or measurable values.
- Also, define the measurement innovation (what is new or unexpected in the measurement) $\tilde{z}_k = z_k - \hat{z}_k$ where $\hat{z}_k = \mathbb{E}\big[z_k \mid \mathbb{Z}_{k-1}\big]$.
- Both $\tilde{x}_k^-$ and $\tilde{z}_k$ can be shown to be zero mean using the method of iterated expectation: $\mathbb{E}\big[\mathbb{E}\big[X \mid Z\big]\big] = \mathbb{E}\big[X\big]$.

$$\mathbb{E}\big[\tilde{x}_k^-\big] = \mathbb{E}\big[x_k\big] - \mathbb{E}\big[\mathbb{E}\big[x_k \mid \mathbb{Z}_{k-1}\big]\big] = \mathbb{E}\big[x_k\big] - \mathbb{E}\big[x_k\big] = 0$$

$$\mathbb{E}\big[\tilde{z}_k\big] = \mathbb{E}\big[z_k\big] - \mathbb{E}\big[\mathbb{E}\big[z_k \mid \mathbb{Z}_{k-1}\big]\big] = \mathbb{E}\big[z_k\big] - \mathbb{E}\big[z_k\big] = 0.$$

- Note also that $\tilde{x}_k^-$ is uncorrelated with past measurements as they have already been incorporated into $\hat{x}_k^-$

$$\mathbb{E}\big[\tilde{x}_k^- \mid \mathbb{Z}_{k-1}\big] = \mathbb{E}\big[x_k - \mathbb{E}\big[x_k \mid \mathbb{Z}_{k-1}\big] \mid \mathbb{Z}_{k-1}\big] = 0 = \mathbb{E}\big[\tilde{x}_k^-\big].$$

- Therefore, on one hand we can write the relationship

$$\mathbb{E}\big[\tilde{x}_k^- \mid \mathbb{Z}_k\big] = \underbrace{\mathbb{E}\big[x_k \mid \mathbb{Z}_k\big]}_{\hat{x}_k^+} - \underbrace{\mathbb{E}\big[\hat{x}_k^- \mid \mathbb{Z}_k\big]}_{\hat{x}_k^-}.$$

- On the other hand, we can write

$$\mathbb{E}\big[\tilde{x}_k^- \mid \mathbb{Z}_k\big] = \mathbb{E}\big[\tilde{x}_k^- \mid \mathbb{Z}_{k-1}, z_k\big] = \mathbb{E}\big[\tilde{x}_k^- \mid z_k\big].$$

- So,

$$\hat{x}_k^+ = \hat{x}_k^- + \mathbb{E}\big[\tilde{x}_k^- \mid z_k\big],$$

  which is a predict/correct sequence of steps, as promised.

- But, what is $\mathbb{E}\big[\tilde{x}_k^- \mid z_k\big]$?

Conditional expectation of jointly-distributed Gaussian random variables

- To evaluate this expression, we consider very generically the problem of finding $f(x \mid z)$ if x and z are jointly Gaussian vectors.

- We combine x and z into an augmented vector X where

$$X = \begin{bmatrix} x \\ z \end{bmatrix} \qquad \text{and} \qquad \mathbb{E}\big[X\big] = \begin{bmatrix} \bar{x} \\ \bar{z} \end{bmatrix}.$$

  and x and z have joint covariance matrix

$$\Sigma_{\tilde{X}} = \begin{bmatrix} \Sigma_{\tilde{x}} & \Sigma_{\tilde{x}\tilde{z}} \\ \Sigma_{\tilde{z}\tilde{x}} & \Sigma_{\tilde{z}} \end{bmatrix}.$$

- We will then apply this generic result for $f(x \mid z)$ to our specific problem of determining $f(\tilde{x}_k^- \mid z_k)$ for the dynamic system we are considering.

- We can then use this conditional pdf to find $\mathbb{E}\left[\tilde{x}_k^- \mid z_k\right]$.
- To proceed, we first write

$$f(x \mid z) = \frac{f(x,z)}{f(z)}.$$

- Note that this is proportional to

$$\frac{f(x,z)}{f(z)} \propto \frac{\exp\left(-\frac{1}{2}\left(\begin{bmatrix} x \\ z \end{bmatrix} - \begin{bmatrix} \bar{x} \\ \bar{z} \end{bmatrix}\right)^T \Sigma_{\tilde{X}}^{-1} \left(\begin{bmatrix} x \\ z \end{bmatrix} - \begin{bmatrix} \bar{x} \\ \bar{z} \end{bmatrix}\right)\right)}{\exp\left(-\frac{1}{2}(z-\bar{z})^T \Sigma_{\tilde{z}}^{-1}(z-\bar{z})\right)},$$

  where the constant of proportionality is not important to subsequent operations.
- Combining the exponential terms, the term in the exponent becomes

$$-\frac{1}{2}\left(\begin{bmatrix} x \\ z \end{bmatrix} - \begin{bmatrix} \bar{x} \\ \bar{z} \end{bmatrix}\right)^T \Sigma_{\tilde{X}}^{-1} \left(\begin{bmatrix} x \\ z \end{bmatrix} - \begin{bmatrix} \bar{x} \\ \bar{z} \end{bmatrix}\right) + \frac{1}{2}(z-\bar{z})^T \Sigma_{\tilde{z}}^{-1}(z-\bar{z}).$$

- To condense notation somewhat, we define $\tilde{x} = x - \bar{x}$ and $\tilde{z} = z - \bar{z}$. Then, the terms in the exponent become,

$$-\frac{1}{2}\begin{bmatrix} \tilde{x} \\ \tilde{z} \end{bmatrix}^T \Sigma_{\tilde{X}}^{-1} \begin{bmatrix} \tilde{x} \\ \tilde{z} \end{bmatrix} + \frac{1}{2}\tilde{z}^T \Sigma_{\tilde{z}}^{-1} \tilde{z}.$$

- To proceed further, we need to be able to invert $\Sigma_{\tilde{X}}$. To do so, we substitute the following transformation (which you can verify)

$$\begin{bmatrix} \Sigma_{\tilde{x}} & \Sigma_{\tilde{x}\tilde{z}} \\ \Sigma_{\tilde{z}\tilde{x}} & \Sigma_{\tilde{z}} \end{bmatrix} = \begin{bmatrix} I & \Sigma_{\tilde{x}\tilde{z}}\Sigma_{\tilde{z}}^{-1} \\ 0 & I \end{bmatrix} \begin{bmatrix} \Sigma_{\tilde{x}} - \Sigma_{\tilde{x}\tilde{z}}\Sigma_{\tilde{z}}^{-1}\Sigma_{\tilde{z}\tilde{x}} & 0 \\ 0 & \Sigma_{\tilde{z}} \end{bmatrix} \begin{bmatrix} I & 0 \\ \Sigma_{\tilde{z}}^{-1}\Sigma_{\tilde{z}\tilde{x}} & I \end{bmatrix}.$$

- Then,

$$\begin{bmatrix} \Sigma_{\tilde{x}} & \Sigma_{\tilde{x}\tilde{z}} \\ \Sigma_{\tilde{z}\tilde{x}} & \Sigma_{\tilde{z}} \end{bmatrix}^{-1} = \begin{bmatrix} I & 0 \\ \Sigma_{\tilde{z}}^{-1}\Sigma_{\tilde{z}\tilde{x}} & I \end{bmatrix}^{-1} \begin{bmatrix} \Sigma_{\tilde{x}} - \Sigma_{\tilde{x}\tilde{z}}\Sigma_{\tilde{z}}^{-1}\Sigma_{\tilde{z}\tilde{x}} & 0 \\ 0 & \Sigma_{\tilde{z}} \end{bmatrix}^{-1} \begin{bmatrix} I & \Sigma_{\tilde{x}\tilde{z}}\Sigma_{\tilde{z}}^{-1} \\ 0 & I \end{bmatrix}^{-1}$$

$$= \begin{bmatrix} I & 0 \\ -\Sigma_{\tilde{z}}^{-1}\Sigma_{\tilde{z}\tilde{x}} & I \end{bmatrix} \begin{bmatrix} (\Sigma_{\tilde{x}} - \Sigma_{\tilde{x}\tilde{z}}\Sigma_{\tilde{z}}^{-1}\Sigma_{\tilde{z}\tilde{x}})^{-1} & 0 \\ 0 & \Sigma_{\tilde{z}}^{-1} \end{bmatrix} \begin{bmatrix} I & -\Sigma_{\tilde{x}\tilde{z}}\Sigma_{\tilde{z}}^{-1} \\ 0 & I \end{bmatrix}.$$

- Multiplying out all terms, we get (if we let $M = \Sigma_{\tilde{x}} - \Sigma_{\tilde{x}\tilde{z}}\Sigma_{\tilde{z}}^{-1}\Sigma_{\tilde{z}\tilde{x}}$),

$$\text{exponent} = -\frac{1}{2}\big(\tilde{x}^T M^{-1}\tilde{x} - \tilde{x}^T M^{-1}\Sigma_{\tilde{x}\tilde{z}}\Sigma_{\tilde{z}}^{-1}\tilde{z} - \tilde{z}^T\Sigma_{\tilde{z}}^{-1}\Sigma_{\tilde{z}\tilde{x}}M^{-1}\tilde{x} + \tilde{z}^T\Sigma_{\tilde{z}}^{-1}\Sigma_{\tilde{z}\tilde{x}}M^{-1}\Sigma_{\tilde{x}\tilde{z}}\Sigma_{\tilde{z}}^{-1}\tilde{z} + \tilde{z}^T\Sigma_{\tilde{z}}^{-1}\tilde{z} - \tilde{z}^T\Sigma_{\tilde{z}}^{-1}\tilde{z}\big).$$

  EDUCATIONAL NOTE: M is called the Schur complement of Σ.

- Notice that the last two terms cancel out, and that the remaining terms can be grouped as

$$\text{exponent} = -\frac{1}{2}\big(\tilde{x} - \Sigma_{\tilde{x}\tilde{z}}\Sigma_{\tilde{z}}^{-1}\tilde{z}\big)^T M^{-1}\big(\tilde{x} - \Sigma_{\tilde{x}\tilde{z}}\Sigma_{\tilde{z}}^{-1}\tilde{z}\big),$$

  or, in terms of only the original variables, the exponent is

$$-\frac{1}{2}\Big(x-\big(\bar{x}+\Sigma_{\tilde{x}\tilde{z}}\Sigma_{\tilde{z}}^{-1}(z-\bar{z})\big)\Big)^T\Big(\Sigma_{\tilde{x}}-\Sigma_{\tilde{x}\tilde{z}}\Sigma_{\tilde{z}}^{-1}\Sigma_{\tilde{z}\tilde{x}}\Big)^{-1}\Big(x-\big(\bar{x}+\Sigma_{\tilde{x}\tilde{z}}\Sigma_{\tilde{z}}^{-1}(z-\bar{z})\big)\Big).$$

- We infer from this that the mean of $f(x \mid z)$ must be $\bar{x} + \Sigma_{\tilde{x}\tilde{z}}\Sigma_{\tilde{z}}^{-1}(z - \bar{z})$, and that the covariance of $f(x \mid z)$ must be $\Sigma_{\tilde{x}} - \Sigma_{\tilde{x}\tilde{z}}\Sigma_{\tilde{z}}^{-1}\Sigma_{\tilde{z}\tilde{x}}$.

- So, we conclude that

$$f(x \mid z) \sim \mathcal{N}(\bar{x} + \Sigma_{\tilde{x}\tilde{z}}\Sigma_{\tilde{z}}^{-1}(z - \bar{z}),\ \Sigma_{\tilde{x}} - \Sigma_{\tilde{x}\tilde{z}}\Sigma_{\tilde{z}}^{-1}\Sigma_{\tilde{z}\tilde{x}}).$$

- Then, generically, when x and z are jointly Gaussian distributed,

$$\mathbb{E}\big[x \mid z\big] = \mathbb{E}\big[x\big] + \Sigma_{\tilde{x}\tilde{z}}\Sigma_{\tilde{z}}^{-1}\big(z - \mathbb{E}\big[z\big]\big).$$

4.4: Generic Gaussian probabilistic inference solution

- Now, we specifically want to find $\mathbb{E}\big[\tilde{x}_k^- \mid z_k\big]$. Note that $z_k = \tilde{z}_k + \hat{z}_k$.
- Then,

$$\begin{aligned}\mathbb{E}\big[\tilde{x}_k^- \mid z_k\big] &= \mathbb{E}\big[\tilde{x}_k^- \mid \tilde{z}_k + \hat{z}_k\big] \\ &= \mathbb{E}\big[\tilde{x}_k^-\big] + \Sigma_{\tilde{x}\tilde{z},k}^- \Sigma_{\tilde{z},k}^{-1}\big(\tilde{z}_k + \hat{z}_k - \mathbb{E}\big[\tilde{z}_k + \hat{z}_k\big]\big) \\ &= 0 + \Sigma_{\tilde{x}\tilde{z},k}^- \Sigma_{\tilde{z},k}^{-1}\big(\tilde{z}_k + \hat{z}_k - (0 + \hat{z}_k)\big) \\ &= \underbrace{\Sigma_{\tilde{x}\tilde{z},k}^- \Sigma_{\tilde{z},k}^{-1}}_{L_k}\tilde{z}_k.\end{aligned}$$

- Putting all of the pieces together, we get the general update equation:

$$\hat{x}_k^+ = \hat{x}_k^- + L_k\tilde{z}_k.$$

Covariance calculations

- Note that the covariance of $\tilde{x}_k^+$ is a function of L_k, and may be computed as

$$\begin{aligned}\Sigma_{\tilde{x},k}^+ &= \mathbb{E}\big[(x_k - \hat{x}_k^+)(x_k - \hat{x}_k^+)^T\big] \\ &= \mathbb{E}\big[\big\{(x_k - \hat{x}_k^-) - L_k\tilde{z}_k\big\}\big\{(x_k - \hat{x}_k^-) - L_k\tilde{z}_k\big\}^T\big] \\ &= \mathbb{E}\big[\big\{\tilde{x}_k^- - L_k\tilde{z}_k\big\}\big\{\tilde{x}_k^- - L_k\tilde{z}_k\big\}^T\big] \\ &= \Sigma_{\tilde{x},k}^- - L_k\underbrace{\mathbb{E}\big[\tilde{z}_k(\tilde{x}_k^-)^T\big]}_{\Sigma_{\tilde{z},k}L_k^T} - \underbrace{\mathbb{E}\big[\tilde{x}_k^-\tilde{z}_k^T\big]}_{L_k\Sigma_{\tilde{z},k}}L_k^T + L_k\Sigma_{\tilde{z},k}L_k^T \\ &= \Sigma_{\tilde{x},k}^- - L_k\Sigma_{\tilde{z},k}L_k^T.\end{aligned}$$

Forest and trees

- To be perfectly clear, the output of this process has two parts:

1. The state estimate. At the end of every iteration, we have computed our best guess of the present state value, which is $\hat{x}_k^+$.
2. The covariance estimate. Our state estimate is not perfect. The covariance matrix $\Sigma_{\tilde{x},k}^+$ gives the uncertainty of $\hat{x}_k^+$. In particular, it might be used to compute error bounds on the estimate.

- Summarizing, the generic Gaussian sequential probabilistic inference recursion becomes:

$$\hat{x}_k^+ = \hat{x}_k^- + L_k\big(z_k - \hat{z}_k\big) = \hat{x}_k^- + L_k\tilde{z}_k$$
$$\Sigma_{\tilde{x},k}^+ = \Sigma_{\tilde{x},k}^- - L_k\Sigma_{\tilde{z},k}L_k^T,$$

where

$$\hat{x}_k^- = \mathbb{E}\big[x_k \mid \mathbb{Z}_{k-1}\big] \qquad \Sigma_{\tilde{x},k}^- = \mathbb{E}\big[(x_k - \hat{x}_k^-)(x_k - \hat{x}_k^-)^T\big] = \mathbb{E}\big[(\tilde{x}_k^-)(\tilde{x}_k^-)^T\big]$$
$$\hat{x}_k^+ = \mathbb{E}\big[x_k \mid \mathbb{Z}_k\big] \qquad \Sigma_{\tilde{x},k}^+ = \mathbb{E}\big[(x_k - \hat{x}_k^+)(x_k - \hat{x}_k^+)^T\big] = \mathbb{E}\big[(\tilde{x}_k^+)(\tilde{x}_k^+)^T\big]$$
$$\hat{z}_k = \mathbb{E}\big[z_k \mid \mathbb{Z}_{k-1}\big] \qquad \Sigma_{\tilde{z},k} = \mathbb{E}\big[(z_k - \hat{z}_k)(z_k - \hat{z}_k)^T\big] = \mathbb{E}\big[(\tilde{z}_k)(\tilde{z}_k)^T\big]$$
$$L_k = \mathbb{E}\big[(x_k - \hat{x}_k^-)(z_k - \hat{z}_k)^T\big]\Sigma_{\tilde{z},k}^{-1} = \Sigma_{\tilde{x}\tilde{z},k}^-\Sigma_{\tilde{z},k}^{-1}.$$

- Note that this is a linear recursion, even if the system is nonlinear(!)

The generic Gaussian sequential-probabilistic-inference solution

- We can now summarize the six basic steps of any Kalman-style filter.

General step 1a: State estimate time update.

- Each time step, compute an updated prediction of the present value of x_k, based on *a priori* information and the system model

$$\hat{x}_k^- = \mathbb{E}\big[x_k \mid \mathbb{Z}_{k-1}\big] = \mathbb{E}\big[f_{k-1}(x_{k-1}, u_{k-1}, w_{k-1}) \mid \mathbb{Z}_{k-1}\big].$$

General step 1b: Error covariance time update.

- Determine the predicted state-estimate error covariance matrix $\Sigma_{\tilde{x},k}^-$ based on *a priori* information and the system model.

- We compute $\Sigma_{\tilde{x},k}^{-} = \mathbb{E}\big[(\tilde{x}_k^{-})(\tilde{x}_k^{-})^T\big]$, knowing that $\tilde{x}_k^{-} = x_k - \hat{x}_k^{-}$.

General step 1c: Estimate system output z_k.

- Estimate the system's output using *a priori* information

$$\hat{z}_k = \mathbb{E}\big[z_k \mid \mathbb{Z}_{k-1}\big] = \mathbb{E}\big[h_k(x_k, u_k, v_k) \mid \mathbb{Z}_{k-1}\big].$$

General step 2a: Estimator gain matrix L_k.

- Compute the estimator gain matrix L_k by evaluating $L_k = \Sigma_{\tilde{x}\tilde{z},k}^{-}\Sigma_{\tilde{z},k}^{-1}$.

General step 2b: State estimate measurement update.

- Compute the *a posteriori* state estimate by updating the *a priori* estimate using the L_k and the output prediction error $z_k - \hat{z}_k$

$$\hat{x}_k^{+} = \hat{x}_k^{-} + L_k(z_k - \hat{z}_k).$$

General step 2c: Error covariance measurement update.

- Compute the *a posteriori* error covariance matrix

$$\begin{aligned}\Sigma_{\tilde{x},k}^{+} &= \mathbb{E}\big[(\tilde{x}_k^{+})(\tilde{x}_k^{+})^T\big] \\ &= \Sigma_{\tilde{x},k}^{-} - L_k\Sigma_{\tilde{z},k}L_k^T,\end{aligned}$$

  which is the result we found previously.

- The estimator output comprises the state estimate $\hat{x}_k^{+}$ and error covariance estimate $\Sigma_{\tilde{x},k}^{+}$.
- The estimator then waits until the next sample interval, updates k, and proceeds to step 1.

PERSPECTIVE: The rest of this course is all about learning how to implement these steps for different scenarios and doing so efficiently and robustly.

General Gaussian sequential probabilistic inference solution

General state-space model:

$$x_k = f_{k-1}(x_{k-1}, u_{k-1}, w_{k-1})$$
$$z_k = h_k(x_k, u_k, v_k),$$

where w_k and v_k are independent, Gaussian noise processes of covariance matrices $\Sigma_{\tilde{w}}$ and $\Sigma_{\tilde{v}}$, respectively.

Definitions: Let

$$\tilde{x}_k^- = x_k - \hat{x}_k^-, \qquad \tilde{z}_k = z_k - \hat{z}_k.$$

Initialization: For $k = 0$, set

$$\hat{x}_0^+ = \mathbb{E}\big[x_0\big]$$
$$\Sigma_{\tilde{x},0}^+ = \mathbb{E}\big[(x_0 - \hat{x}_0^+)(x_0 - \hat{x}_0^+)^T\big].$$

Computation: For $k = 1, 2, \ldots$ compute:

State estimate time update: $\hat{x}_k^- = \mathbb{E}\big[\, f_{k-1}(x_{k-1}, u_{k-1}, w_{k-1}) \mid \mathbb{Z}_{k-1}\big].$

Error covariance time update: $\Sigma_{\tilde{x},k}^- = \mathbb{E}\big[(\tilde{x}_k^-)(\tilde{x}_k^-)^T\big].$

Output prediction: $\hat{z}_k = \mathbb{E}\big[h_k(x_k, u_k, v_k) \mid \mathbb{Z}_{k-1}\big].$

Estimator gain matrix: $L_k = \mathbb{E}\big[(\tilde{x}_k^-)(\tilde{z}_k)^T\big]\Big(\mathbb{E}\big[(\tilde{z}_k)(\tilde{z}_k)^T\big]\Big)^{-1}.$

State estimate measurement update: $\hat{x}_k^+ = \hat{x}_k^- + L_k\big(z_k - \hat{z}_k\big).$

Error covariance measurement update: $\Sigma_{\tilde{x},k}^+ = \Sigma_{\tilde{x},k}^- - L_k \Sigma_{\tilde{z},k} L_k^T.$

4.5: Optimal application to linear systems: The Kalman filter

- In this section, we take the general probabilistic inference solution and apply it to the specific case where the system dynamics are linear.
- Linear systems have the desirable property that all pdfs do in fact remain Gaussian if the stochastic inputs are Gaussian, so the assumptions made in deriving the filter steps hold exactly.
- We will not prove it, but if the system dynamics are linear, then the Kalman filter is the optimal minimum-mean-squared-error and maximum-likelihood estimator.
- The linear Kalman filter assumes that the system being modeled can be represented in the "state-space" form

$$x_k = A_{k-1}x_{k-1} + B_{k-1}u_{k-1} + w_{k-1}$$

$$z_k = C_k x_k + D_k u_k + v_k.$$

- We assume that w_k and v_k are mutually uncorrelated white Gaussian random processes, with zero mean and covariance matrices with known value:

$$\mathbb{E}[w_n w_k^T] = \begin{cases} \Sigma_{\tilde{w}}, & n = k; \\ 0, & n \neq k. \end{cases} \qquad \mathbb{E}[v_n v_k^T] = \begin{cases} \Sigma_{\tilde{v}}, & n = k; \\ 0, & n \neq k, \end{cases}$$

  and $\mathbb{E}[w_k x_0^T] = 0$ for all k.
- The assumptions on the noise processes w_k and v_k and on the linearity of system dynamics are rarely (never) met in practice, but the consensus of the literature and practice is that the method still works very well.

KF step 1a: State estimate time update.

- Here, we compute

$$\begin{aligned}\hat{x}_k^- &= \mathbb{E}\big[f_{k-1}(x_{k-1}, u_{k-1}, w_{k-1}) \mid \mathbb{Z}_{k-1}\big] \\ &= \mathbb{E}\big[A_{k-1}x_{k-1} + B_{k-1}u_{k-1} + w_{k-1} \mid \mathbb{Z}_{k-1}\big] \\ &= \mathbb{E}\big[A_{k-1}x_{k-1} \mid \mathbb{Z}_{k-1}\big] + \mathbb{E}\big[B_{k-1}u_{k-1} \mid \mathbb{Z}_{k-1}\big] + \mathbb{E}\big[w_{k-1} \mid \mathbb{Z}_{k-1}\big] \\ &= A_{k-1}\hat{x}_{k-1}^+ + B_{k-1}u_{k-1},\end{aligned}$$

  by the linearity of expectation, noting that w_{k-1} is zero-mean.

  INTUITION: In this step, we are predicting the present state given only past measurements.

  - The best we can do is to use the most recent state estimate and the system model, propagating the mean forward in time.
  - This is exactly the same thing we did in the last chapter when we discussed propagating the mean state value forward in time without making measurements in that time interval.

KF step 1b: Error covariance time update.

- First, we note that the estimation error $\tilde{x}_k^- = x_k - \hat{x}_k^-$.
- We compute the error itself as

$$\begin{aligned}\tilde{x}_k^- &= x_k - \hat{x}_k^- \\ &= A_{k-1}x_{k-1} + B_{k-1}u_{k-1} + w_{k-1} - A_{k-1}\hat{x}_{k-1}^+ - B_{k-1}u_{k-1} \\ &= A_{k-1}\tilde{x}_{k-1}^+ + w_{k-1}.\end{aligned}$$

- Therefore, the covariance of the prediction error is

$$\begin{aligned}
\Sigma_{\tilde{x}_k}^- &= \mathbb{E}\big[(\tilde{x}_k^-)(\tilde{x}_k^-)^T\big] \\
&= \mathbb{E}\big[(A_{k-1}\tilde{x}_{k-1}^+ + w_{k-1})(A_{k-1}\tilde{x}_{k-1}^+ + w_{k-1})^T\big] \\
&= \mathbb{E}\big[A_{k-1}\tilde{x}_{k-1}^+(\tilde{x}_{k-1}^+)^T A_{k-1}^T + w_{k-1}(\tilde{x}_{k-1}^+)^T A_{k-1}^T \\
&\qquad + A_{k-1}\tilde{x}_{k-1}^+ w_{k-1}^T + w_{k-1}w_{k-1}^T\big] \\
&= A_{k-1}\Sigma_{\tilde{x},k-1}^+ A_{k-1}^T + \Sigma_{\widetilde{w}}.
\end{aligned}$$

- The cross terms drop out of the final result since the white process noise w_{k-1} is not correlated with the state at time $k-1$.

  INTUITION: When estimating error covariance of the state prediction,

  - The best we can do is to use the most recent covariance estimate and propagate it forward in time.
  - This is exactly what we did in the last chapter when we discussed propagating the covariance of the state forward in time, without making measurements during that time interval.
  - For stable systems, $A_{k-1}\Sigma_{\tilde{x},k-1}^+ A_{k-1}^T$ is contractive, meaning that the covariance gets "smaller." The state of stable systems always decays toward zero in the absence of input, or toward a known trajectory if $u_k \neq 0$. As time goes on, this term tells us that we tend to get more and more certain of the state estimate.
  - On the other hand, $\Sigma_{\widetilde{w}}$ adds to the covariance. Unmeasured inputs add uncertainty to our estimate because they perturb the trajectory away from the known trajectory based only on u_k.

KF step 1c: Predict system output.

- We estimate the system output as

$$\begin{aligned}\hat{z}_k &= \mathbb{E}\big[h_k(x_k, u_k, v_k) \mid \mathbb{Z}_{k-1}\big] \\ &= \mathbb{E}\big[C_k x_k + D_k u_k + v_k \mid \mathbb{Z}_{k-1}\big] \\ &= \mathbb{E}\big[C_k x_k \mid \mathbb{Z}_{k-1}\big] + \mathbb{E}\big[D_k u_k \mid \mathbb{Z}_{k-1}\big] + \mathbb{E}\big[v_k \mid \mathbb{Z}_{k-1}\big] \\ &= C_k \hat{x}_k^- + D_k u_k,\end{aligned}$$

since v_k is zero-mean.

INTUITION: $\hat{z}_k$ is our best guess of the system output, given only past measurements.

- The best we can do is to estimate the output given the output equation of the system model, and our best guess of the system state at the present time.

KF step 2a: Estimator (Kalman) gain matrix.

- To compute the Kalman gain matrix, we first need to compute several covariance matrices: $L_k = \Sigma_{\tilde{x}\tilde{z},k}^- \Sigma_{\tilde{z},k}^{-1}$.
- We first find $\Sigma_{\tilde{z},k}$.

$$\begin{aligned}\tilde{z}_k &= z_k - \hat{z}_k \\ &= C_k x_k + D_k u_k + v_k - C_k \hat{x}_k^- - D_k u_k \\ &= C_k \tilde{x}_k^- + v_k \\ \Sigma_{\tilde{z},k} &= \mathbb{E}\big[(C_k \tilde{x}_k^- + v_k)(C_k \tilde{x}_k^- + v_k)^T\big] \\ &= \mathbb{E}\big[C_k \tilde{x}_k^- (\tilde{x}_k^-)^T C_k^T + v_k (\tilde{x}_k^-)^T C_k^T + C_k \tilde{x}_k^- v_k^T + v_k v_k^T\big] \\ &= C_k \Sigma_{\tilde{x},k}^- C_k^T + \Sigma_{\tilde{v}}.\end{aligned}$$

- Again, the cross terms are zero since v_k is uncorrelated with $\tilde{x}_k^-$.

- Similarly,

$$\begin{aligned}\mathbb{E}[\tilde{x}_k^- \tilde{z}_k^T] &= \mathbb{E}\left[\tilde{x}_k^- (C_k \tilde{x}_k^- + v_k)^T\right] \\ &= \mathbb{E}\left[\tilde{x}_k^- (\tilde{x}_k^-)^T C_k^T + \tilde{x}_k^- v_k^T\right] \\ &= \Sigma_{\tilde{x},k}^- C_k^T.\end{aligned}$$

- Combining,

$$L_k = \Sigma_{\tilde{x},k}^- C_k^T [C_k \Sigma_{\tilde{x},k}^- C_k^T + \Sigma_{\tilde{v}}]^{-1}.$$

INTUITION: Note that the computation of L_k is the most critical aspect of Kalman filtering that distinguishes it from a number of other estimation methods.

- The whole reason for calculating covariance matrices is to be able to update L_k.
- L_k is time-varying. It adapts to give the best update to the state estimate based on present conditions.
- Recall that we use L_k in the equation $\hat{x}_k^+ = \hat{x}_k^- + L_k(z_k - \hat{z}_k)$.
- The first component to L_k, $\Sigma_{\tilde{x}\tilde{z},k}^-$, indicates relative need for correction to $\hat{x}_k$ and how well individual states within $\hat{x}_k$ are coupled to the measurements.
- We see this clearly in $\Sigma_{\tilde{x}\tilde{z},k}^- = \Sigma_{\tilde{x},k}^- C_k^T$.
- $\Sigma_{\tilde{x},k}^-$ tells us about state uncertainty at the present time, which we hope to reduce as much as possible.
  - A large entry in $\Sigma_{\tilde{x},k}^-$ means that the corresponding state is very uncertain and therefore would benefit from a large update.
  - A small entry in $\Sigma_{\tilde{x},k}^-$ means that the corresponding state is very well known already and does not need as large an update.

- The C_k^T term gives the coupling between state and output.
  - Entries that are zero indicate that a particular state has no direct influence on a particular output and therefore an output prediction error should not directly update that state.
  - Entries that are large indicate that a particular state is highly coupled to an output so has a large contribution to any measured output prediction error; therefore, that state would benefit from a large update.
- $\Sigma_{\tilde{z}}$ tells us how certain we are that the measurement is reliable.
  - If $\Sigma_{\tilde{z}}$ is "large," we want small, slow updates.
  - If $\Sigma_{\tilde{z}}$ is "small," we want big updates.
  - This explains why we divide the Kalman gain matrix by $\Sigma_{\tilde{z}}$.
- The form of $\Sigma_{\tilde{z}}$ can also be explained.
  - The $C_k \Sigma_{\tilde{x}}^- C_k^T$ part indicates how error in the state contributes to error in the output estimate.
  - The $\Sigma_{\tilde{v}}$ term indicates the uncertainty in the sensor reading due to sensor noise.
  - Since sensor noise is assumed independent of the state, the uncertainty in $\tilde{z}_k = z_k - \hat{z}_k$ adds the uncertainty in z_k to the uncertainty in $\hat{z}_k$.

KF step 2b: State estimate measurement update.

- The fifth step is to compute the *a posteriori* state estimate by updating the *a priori* estimate using the estimator gain and the output prediction error $z_k - \hat{z}_k$

$$\hat{x}_k^+ = \hat{x}_k^- + L_k(z_k - \hat{z}_k).$$

INTUITION: The variable $\hat{z}_k$ is what we expect the measurement to be, based on our state estimate at the moment.

- So, $z_k - \hat{z}_k$ is what is unexpected or new in the measurement.
- We call this term the innovation. The innovation can be due to a bad system model, state error, or sensor noise.
- So, we want to use this new information to update the state, but must be careful to weight it according to the value of the information it contains.
- L_k is the optimal blending factor, as we have already discussed.

KF step 2c: Error covariance measurement update.

- Finally, we update the error covariance matrix.

$$\begin{aligned}\Sigma^+_{\tilde{x},k} &= \Sigma^-_{\tilde{x},k} - L_k \Sigma_{\tilde{z},k} L_k^T \\ &= \Sigma^-_{\tilde{x},k} - L_k \Sigma_{\tilde{z},k} \Sigma^{-T}_{\tilde{z},k} \left(\Sigma^-_{\tilde{x}\tilde{z},k}\right)^T \\ &= \Sigma^-_{\tilde{x},k} - L_k C_k \Sigma^-_{\tilde{x},k} \\ &= (I - L_k C_k)\Sigma^-_{\tilde{x},k}.\end{aligned}$$

INTUITION: The measurement update has decreased our uncertainty in the state estimate.

- A covariance matrix is positive semi-definite, and $L_k \Sigma_{\tilde{z},k} L_k^T$ is also a positive-semi-definite form, and we are subtracting this from the predicted-state covariance matrix; therefore, the resulting covariance is "lower" than the pre-measurement covariance.

COMMENT: If a measurement is missed for some reason, then simply skip steps 4–6 for that iteration. That is, $L_k = 0$ and $\hat{x}^+_k = \hat{x}^-_k$ and $\Sigma^+_{\tilde{x},k} = \Sigma^-_{\tilde{x},k}$.

Summary of the linear Kalman filter

Linear state-space model:

$$x_k = A_{k-1}x_{k-1} + B_{k-1}u_{k-1} + w_{k-1}$$
$$z_k = C_k x_k + D_k u_k + v_k,$$

where w_k and v_k are independent, zero-mean, Gaussian noise processes of covariance matrices $\Sigma_{\tilde{w}}$ and $\Sigma_{\tilde{v}}$, respectively.

Initialization: For $k = 0$, set

$$\hat{x}_0^+ = \mathbb{E}[x_0]$$
$$\Sigma_{\tilde{x},0}^+ = \mathbb{E}\big[(x_0 - \hat{x}_0^+)(x_0 - \hat{x}_0^+)^T\big].$$

Computation: For $k = 1, 2, \ldots$ compute:

State estimate time update: $\hat{x}_k^- = A_{k-1}\hat{x}_{k-1}^+ + B_{k-1}u_{k-1}.$

Error covariance time update: $\Sigma_{\tilde{x},k}^- = A_{k-1}\Sigma_{\tilde{x},k-1}^+ A_{k-1}^T + \Sigma_{\tilde{w}}.$

Output prediction: $\hat{z}_k = C_k\hat{x}_k^- + D_k u_k.$

*Estimator gain matrix:** $L_k = \Sigma_{\tilde{x},k}^- C_k^T [C_k \Sigma_{\tilde{x},k}^- C_k^T + \Sigma_{\tilde{v}}]^{-1}.$

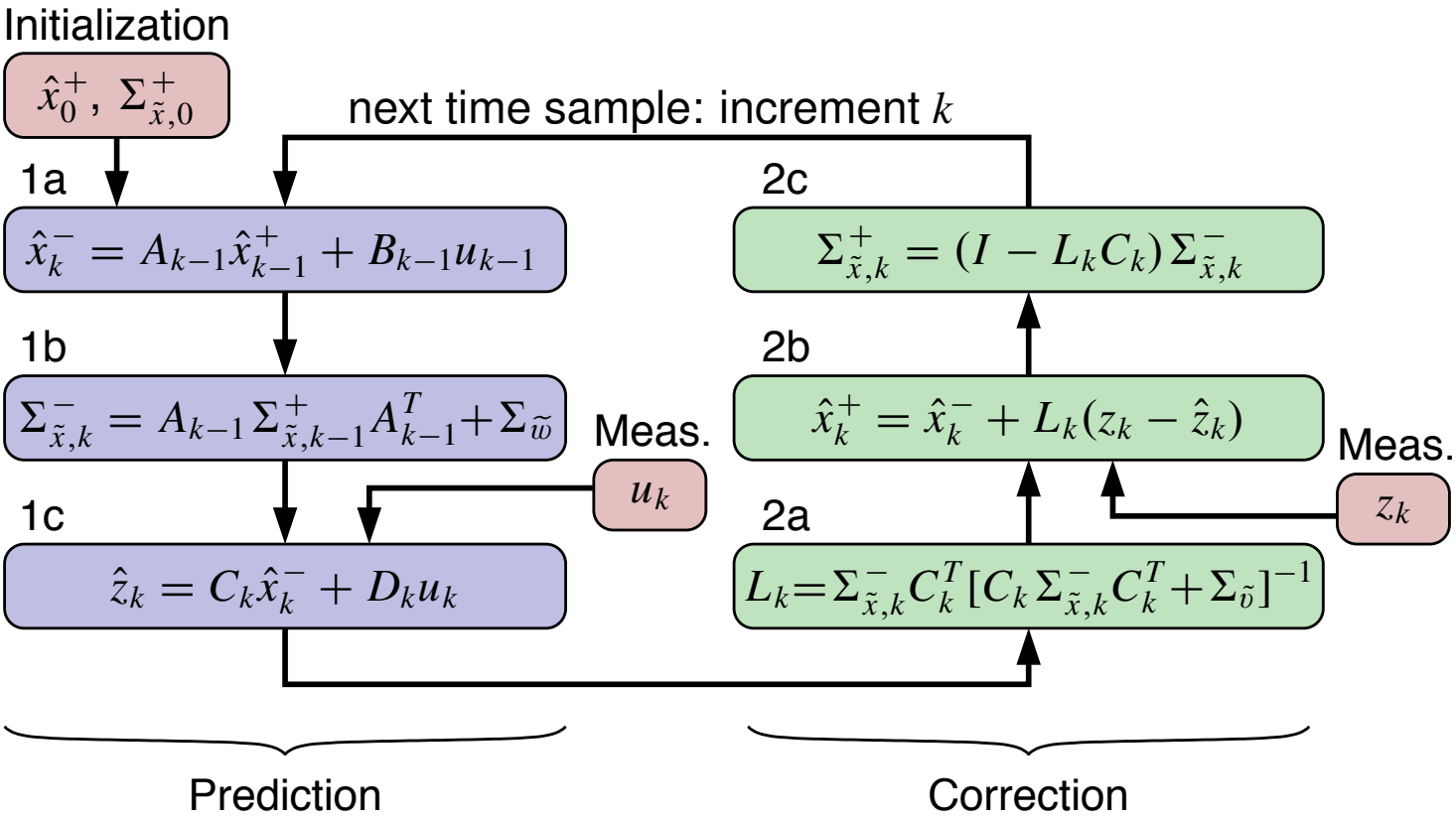
State estimate measurement update: $\hat{x}_k^+ = \hat{x}_k^- + L_k\big(z_k - \hat{z}_k\big).$

Error covariance measurement update: $\Sigma_{\tilde{x},k}^+ = (I - L_k C_k)\Sigma_{\tilde{x},k}^-.$

*If a measurement is missed for some reason, then simply skip the measurement update for that iteration. That is, $L_k = 0$ and $\hat{x}_k^+ = \hat{x}_k^-$ and $\Sigma_{\tilde{x},k}^+ = \Sigma_{\tilde{x},k}^-$.

4.6: Visualizing the Kalman filter

- The Kalman filter equations naturally form the following recursion

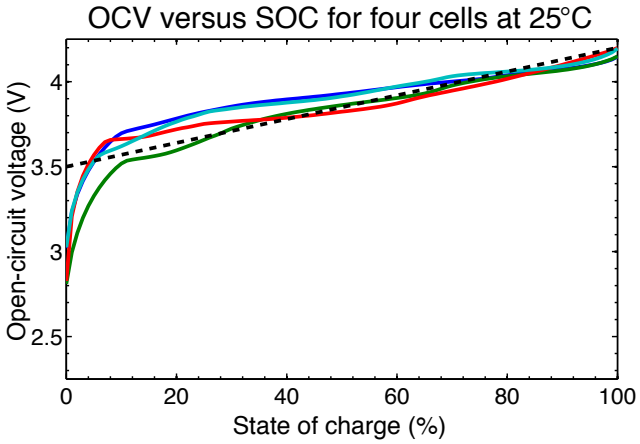


Working an example by hand to gain intuition

- We'll explore state estimation for a battery cell, whose model is nonlinear. We cannot apply the (linear) Kalman filter directly.
- To demonstrate the KF steps, we'll develop and use a linearized cell model

$$q_{k+1} = 1 \cdot q_k - \frac{1}{3600 \cdot Q} i_k$$

$$\text{volt}_k = 3.5 + 0.7 \times q_k - R_0 i_k.$$



- We have linearized the OCV function, and omitted some dynamic effects.
- This model still isn't linear because of the "3.5" in the output equation, so we debias the measurement via $z_k = \text{volt}_k - 3.5$ and use the model

$$q_{k+1} = 1 \cdot q - \frac{1}{3600 \cdot Q} i_k$$

$$z_k = 0.7 \times q_k - R_0 i_k.$$

- Define state $x_k \equiv q_k$ and input $u_k \equiv i_k$.
- For the sake of example, we will use $Q = 10000/3600$ and $R_0 = 0.01$.
- This yields a state-space description with $A = 1$, $B = -1 \times 10^{-4}$, $C = 0.7$, and $D = -0.01$. We also model $\Sigma_{\tilde{w}} = 10^{-5}$, and $\Sigma_{\tilde{v}} = 0.1$.
- We assume no initial uncertainty so $\hat{x}_0^+ = 0.5$ and $\Sigma_{\tilde{x},0}^+ = 0$.

Iteration 1: (let $i_0 = 1$, $i_1 = 0.5$ and $v_1 = 3.85$)

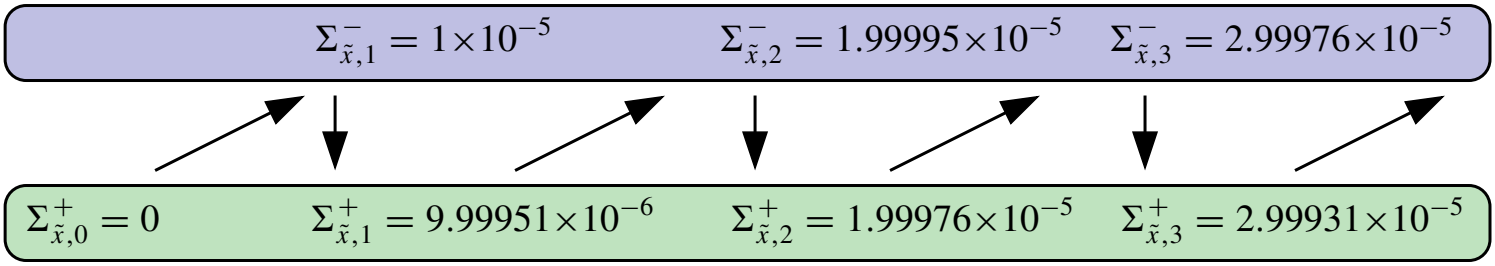
$\hat{x}_k^- = A_{k-1}\hat{x}_{k-1}^+ + B_{k-1}u_{k-1}$	$\hat{x}_1^- = 1 \times 0.5 - 10^{-4} \times 1 = 0.4999$
$\Sigma_{\tilde{x},k}^- = A_{k-1}\Sigma_{\tilde{x},k-1}^+ A_{k-1}^T + \Sigma_{\tilde{w}}$	$\Sigma_{\tilde{x},1}^- = 1 \times 0 \times 1 + 10^{-5} = 10^{-5}$
$\hat{z}_k = C_k\hat{x}_k^- + D_k u_k$	$\hat{z}_1 = 0.7 \times 0.4999 - 0.01 \times 0.5 = 0.34493$
$L_k = \Sigma_{\tilde{x},k}^- C_k^T [C_k \Sigma_{\tilde{x},k}^- C_k^T + \Sigma_{\tilde{v}}]^{-1}$	$L_1 = 10^{-5} \times 0.7[0.7^2 \times 10^{-5} + 0.1]^{-1}$ $= 6.99966 \times 10^{-5}$
$\hat{x}_k^+ = \hat{x}_k^- + L_k(z_k - \hat{z}_k)$ (where $z_k = 3.85 - 3.5$)	$\hat{x}_1^+ = 0.4999 + 6.99966 \times 10^{-5}(0.35 - 0.34493)$ $= 0.4999004$
$\Sigma_{\tilde{x},k}^+ = (I - L_k C_k)\Sigma_{\tilde{x},k}^-$	$\Sigma_{\tilde{x},1}^+ = (1 - 6.99966 \times 10^{-5} \times 0.7) \times 10^{-5}$ $= 9.9995 \times 10^{-6}$

- Output: $\hat{q} = 0.4999 \pm 3\sqrt{9.9995 \times 10^{-6}} = 0.4999 \pm 0.0094866$.

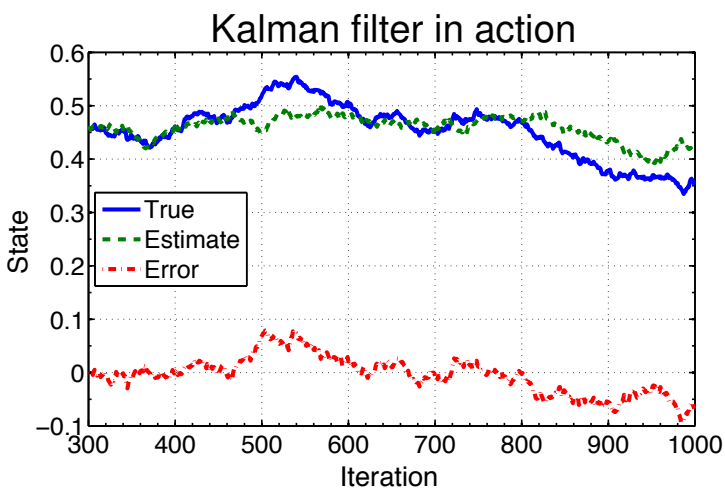
Iteration 2: (let $i_1 = 0.5$, $i_2 = 0.25$, and $v_2 = 3.84$)

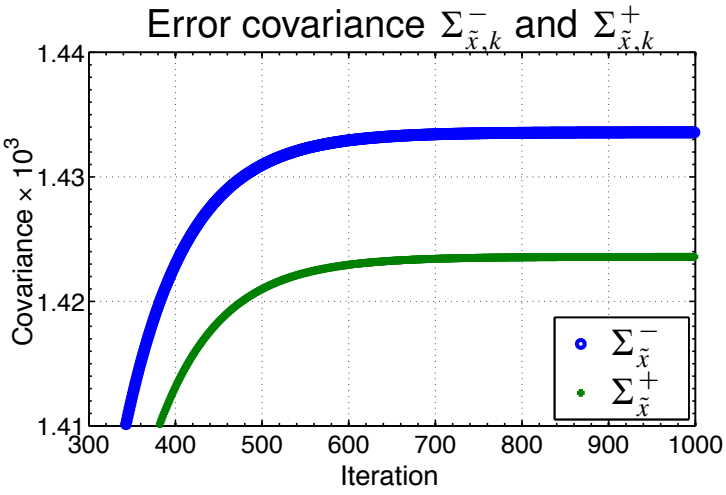
$\hat{x}_k^- = A_{k-1}\hat{x}_{k-1}^+ + B_{k-1}u_{k-1}$	$\hat{x}_1^- = 0.4999004 - 10^{-4} \times 0.5 = 0.49985$
$\Sigma_{\tilde{x},k}^- = A_{k-1}\Sigma_{\tilde{x},k-1}^+ A_{k-1}^T + \Sigma_{\tilde{w}}$	$\Sigma_{\tilde{x},2}^- = 9.9995\times10^{-6} + 10^{-5} = 1.99995\times10^{-5}$
$\hat{z}_k = C_k\hat{x}_k^- + D_k u_k$	$\hat{z}_2 = 0.7\times0.49985 - 0.01 \times 0.25 = 0.347395$
$L_k = \Sigma_{\tilde{x},k}^- C_k^T [C_k \Sigma_{\tilde{x},k}^- C_k^T + \Sigma_{\tilde{v}}]^{-1}$	$L_2 = 1.99995\times10^{-5} \times 0.7[1.99995\times10^{-5} \times 0.7^2 + 0.1]^{-1}$ $= 0.00013998$
$\hat{x}_k^+ = \hat{x}_k^- + L_k(z_k - \hat{z}_k)$ (where $z_k = 3.84 - 3.5$)	$\hat{x}_2^+ = 0.49985 + 0.00013998(0.34 - 0.347395)$ $= 0.499849$
$\Sigma_{\tilde{x},k}^+ = (I - L_k C_k)\Sigma_{\tilde{x},k}^-$	$\Sigma_{\tilde{x},2}^+ = (1 - 0.00013998 \times 0.7) \times 1.99995\times10^{-5}$ $= 1.99976\times10^{-5}$

- Output: $\hat{q} = 0.4998 \pm 3\sqrt{1.99976 \times 10^{-5}} = 0.4998 \pm 0.013416.$
- Note that covariance (uncertainty) converges, but it can take time



- Covariance increases during propagation and is then reduced by each measurement.
- Covariance still oscillates at steady state between $\Sigma_{\tilde{x},ss}^-$ and $\Sigma_{\tilde{x},ss}^+$.
- Estimation error bounds are $\pm 3\sqrt{\Sigma_{\tilde{x},k}^+}$ for 99^+ % assurance of accuracy of estimate.
- The plots below show a sample of the Kalman filter operating. We shall look at how to write code to evaluate this example shortly.





- Note that Kalman filter does not perform especially well since $\Sigma_{\tilde{v}}$ is quite large.

Steady state

- Note in the prior example that the estimate closely follows the true state, the estimation error never converges to zero (because of the driving process noise) but stays within steady-state bounds.
- Note the two different steady-state values for state covariance: one pre-measurement and one post-measurement.
- Often find that the gains of our filter L_k have an initial transient, and then settle to constant values very quickly, when A, B, C constant.
- Of course, there are two steady-state solutions: $\Sigma^-_{\tilde{x},ss}$ and $\Sigma^+_{\tilde{x},ss}$, and furthermore $L_{ss} = \Sigma^+_{\tilde{x},ss} C^T \Sigma^{-1}_{\tilde{v}}$.
- Consider filter loop as $k \to \infty$

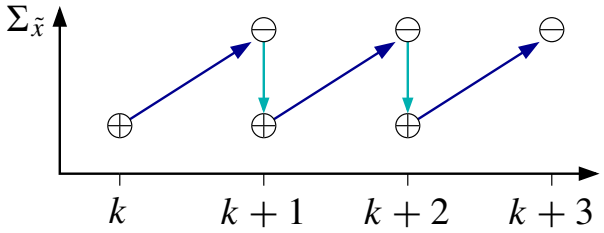
$$\Sigma^-_{\tilde{x},ss} = A \Sigma^+_{\tilde{x},ss} A^T + \Sigma_{\tilde{w}}$$

$$\Sigma^+_{\tilde{x},ss} = \Sigma^-_{\tilde{x},ss} - \Sigma^-_{\tilde{x},ss} C^T \left[C \Sigma^-_{\tilde{x},ss} C^T + \Sigma_{\tilde{v}} \right]^{-1} C \Sigma^-_{\tilde{x},ss}.$$

- Combine these two to get

$$\Sigma_{\tilde{x},ss}^{-} = \Sigma_{\tilde{w}} + A\Sigma_{\tilde{x},ss}^{-}A^T - A\Sigma_{\tilde{x},ss}^{-}C^T\left[C\Sigma_{\tilde{x},ss}^{-}C^T + \Sigma_{\tilde{v}}\right]^{-1}C\Sigma_{\tilde{x},ss}^{-}A^T.$$

- One matrix equation, one matrix unknown ➠ `dlqe.m`
- We will look at this more deeply later in this chapter.



EXAMPLE: $x_{k+1} = x_k + w_k$. $z_k = x_k + v_k$. Steady-state covariance?

- Let $\mathbb{E}[w_k] = \mathbb{E}[v_k] = 0$, $\Sigma_{\tilde{w}} = 1$, $\Sigma_{\tilde{v}} = 2$.
- $\mathbb{E}[x_0] = \hat{x}_0$ and $\mathbb{E}[(x_0 - \hat{x}_0)(x_0 - \hat{x}_0)] = \Sigma_{\tilde{x},0}$.
- Notice $A = 1$, $C = 1$, so

$$\Sigma_{\tilde{x},ss}^{-} = \Sigma_{\tilde{w}} + A\Sigma_{\tilde{x},ss}^{-}A - \frac{\left(A\Sigma_{\tilde{x},ss}^{-}C\right)\left(C\Sigma_{\tilde{x},ss}^{-}A\right)}{C\Sigma_{\tilde{x},ss}^{-}C + \Sigma_{\tilde{v}}}$$

$$= 1 + \Sigma_{\tilde{x},ss}^{-} - \frac{\left(\Sigma_{\tilde{x},ss}^{-}\right)^2}{\Sigma_{\tilde{x},ss}^{-} + 2}$$

$$\Sigma_{\tilde{x},ss}^{-} + 2 = \left(\Sigma_{\tilde{x},ss}^{-}\right)^2,$$

  which leads to the solutions $\Sigma_{\tilde{x},ss}^{-} = -1$ or $\Sigma_{\tilde{x},ss}^{-} = 2$.

- We must chose the positive-definite soln., so $\Sigma_{\tilde{x},ss}^{-} = 2$ and $\Sigma_{\tilde{x},ss}^{+} = 1$.

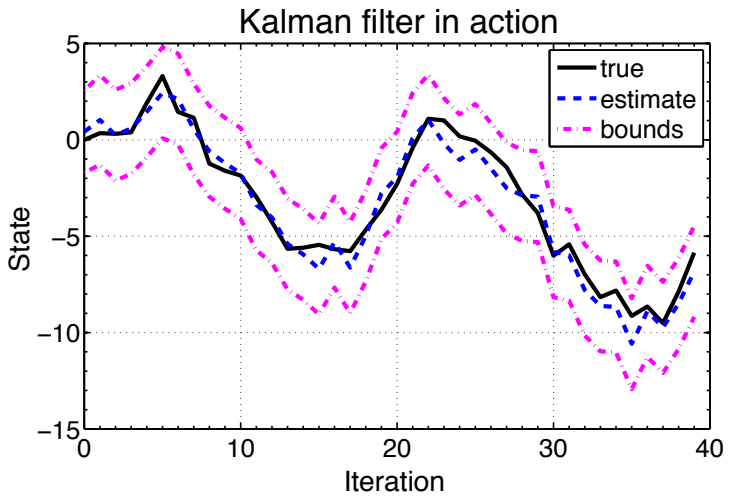
$$\begin{array}{ccccccccc} \Sigma_{\tilde{x},0}^{-} = \infty & & \Sigma_{\tilde{x},1}^{-} = 3 & & \Sigma_{\tilde{x},2}^{-} = 11/5 & & \Sigma_{\tilde{x},3}^{-} = 43/21 & & \\ \downarrow & \nearrow & \downarrow & \nearrow & \downarrow & \nearrow & \downarrow & \nearrow & \\ \Sigma_{\tilde{x},0}^{+} = 2 & & \Sigma_{\tilde{x},1}^{+} = 6/5 & & \Sigma_{\tilde{x},2}^{+} = 22/21 & & \Sigma_{\tilde{x},3}^{+} = 86/85 & & \end{array}$$

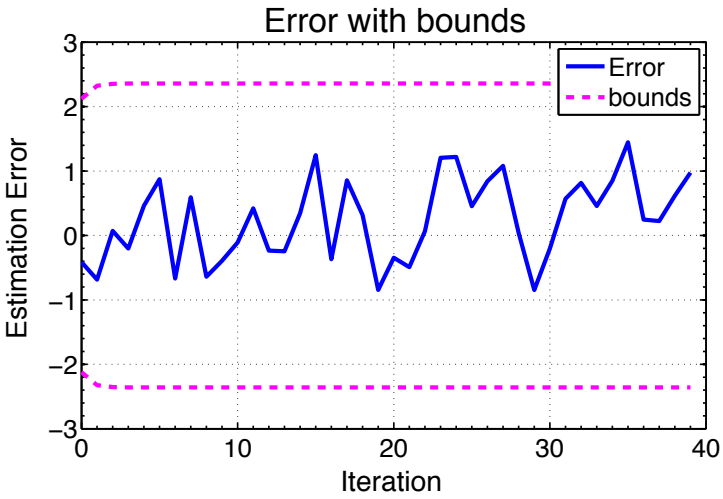
4.7: MATLAB code for the Kalman filter steps

- It is straightforward to convert the Kalman filter steps to MATLAB.
  - Great care must be taken to ensure that all "k" and "$k+1$" indices (etc) are kept synchronized.
- Coding a KF in another language (*e.g.*, "C") is no harder, except that you will need to write (or find) code to do the matrix manipulations.

```
% Initialize simulation variables
SigmaW = 1; SigmaV = 1; % Process and sensor noise covariance
A = 1; B = 1; C = 1; D = 0; % Plant definition matrices
maxIter = 40;

xtrue = 0; xhat = 0; % Initialize true system and KF initial state
SigmaX = 0; % Initialize Kalman filter covariance
u = 0;      % Unknown initial driving input: assume zero

% Reserve storage for variables we  want to plot/evaluate
xstore = zeros(length(xtrue),maxIter+1); xstore(:,1) = xtrue;
xhatstore = zeros(length(xhat),maxIter);
SigmaXstore = zeros(length(xhat)^2,maxIter);

for k = 1:maxIter,
  % KF Step 1a: State estimate time update
  xhat = A*xhat + B*u; % use prior value of "u"

  % KF Step 1b: Error covariance time update
  SigmaX = A*SigmaX*A' + SigmaW;

  % [Implied operation of system in background, with
  % input signal u, and output signal z]
  u = 0.5*randn(1) + cos(k/pi); % for example...
  w = chol(SigmaW)'*randn(length(xtrue));
  v = chol(SigmaV)'*randn(length(C*xtrue));
  ztrue = C*xtrue + D*u + v;  % z is based on present x and u
  xtrue = A*xtrue + B*u + w;  % future x is based on present u

  % KF Step 1c: Estimate system output
  zhat = C*xhat + D*u;
```

```
  % KF Step 2a: Compute Kalman gain matrix
  L = SigmaX*C'/(C*SigmaX*C' + SigmaV);

  % KF Step 2b: State estimate measurement update
  xhat = xhat + L*(ztrue - zhat);

  % KF Step 2c: Error covariance measurement update
  SigmaX = SigmaX - L*C*SigmaX;

  % [Store information for evaluation/plotting purposes]
  xstore(:,k+1) = xtrue; xhatstore(:,k) = xhat;
  SigmaXstore(:,k) = SigmaX(:);
end

figure(1); clf;
plot(0:maxIter-1,xstore(1:maxIter)','k-',0:maxIter-1,xhatstore','b--', ...
     0:maxIter-1,xhatstore'+3*sqrt(SigmaXstore'),'m-.',...
     0:maxIter-1,xhatstore'-3*sqrt(SigmaXstore'),'m-.'); grid;
legend('true','estimate','bounds');
title('Kalman filter in action'); xlabel('Iteration'); ylabel('State');

figure(2); clf;
plot(0:maxIter-1,xstore(1:maxIter)'-xhatstore','b-',...
     0:maxIter-1,3*sqrt(SigmaXstore'),'m--',...
     0:maxIter-1,-3*sqrt(SigmaXstore'),'m--'); grid;
legend('Error','bounds',0);
title('Error w/ bounds'); xlabel('Iteration'); ylabel('Estimation Error');
```





4.8: Steady state: Deriving the Hamiltonian

- As we saw in some earlier examples, it is possible for the Kalman-filter recursion to converge to a unique steady-state solution.
- The technical requirements for this to be true are:
  1. The linear system has A, B, C, D, $\Sigma_{\tilde{w}}$, and $\Sigma_{\tilde{v}}$ matrices that are *not* time-varying
  2. The pair $\{A, C\}$ is completely observable
  3. The pair $\{A, \sqrt{\Sigma_{\tilde{w}}}\}$, where $\sqrt{\Sigma_{\tilde{w}}}$ is the Cholesky factor of $\Sigma_{\tilde{w}}$ such that $\Sigma_{\tilde{w}} = \sqrt{\Sigma_{\tilde{w}}}\sqrt{\Sigma_{\tilde{w}}}^T$, is controllable.
- The first criterion is needed for there to be a possibility of a non time-varying steady state solution
- The second criterion is needed to guarantee a "steady flow" of information about *each* state component, preventing the uncertainty from becoming unbounded
- The third criterion is needed to ensure that the process noise affects *each* state, preventing the convergence of the covariance of any state to zero. It forces the covariance to be positive definite.
- When these criteria are satisfied, there is a unique steady-state solution: $\Sigma^+_{\tilde{x},k} \rightarrow \Sigma^+_{\tilde{x},ss}$, $\Sigma^-_{\tilde{x},k} \rightarrow \Sigma^-_{\tilde{x},ss}$, and $L_k \rightarrow L$.
- The importance of this result is that we can give up a small degree of optimality by using the time-constant L instead of the time-varying L_k, but have a vastly simpler implementation.
- The reason is that if we can pre-compute L, then we no longer need to recursively compute $\Sigma^{\pm}_{\tilde{x},k}$. The filter equations become:

$$\hat{x}_k^- = A\hat{x}_{k-1}^+ + Bu_{k-1}$$

$$\hat{x}_k^+ = \hat{x}_k^- + L(z_k - C\hat{x}_k^- - Du_k).$$

- This could, of course, be combined to form a single equation defining a recursion for $\hat{x}_k$.
- One approach is to directly optimize the coefficients of the vector L for a specific model.
  - This results in the popular α-β and α-β-γ filters.
  - The α-β assumes an NCV model. α and β are unitless parameters, and the steady-state gain has the form: $L = \left[\begin{array}{cc} \alpha, & \beta/T \end{array}\right]^T$.
  - The α-β-γ filter assumes an NCA model. γ is similarly unitless, and the steady-state gain has the form: $L = \left[\begin{array}{ccc} \alpha, & \beta/T, & \gamma/T^2 \end{array}\right]^T$.
- We will investigate a more general approach here, using a derivation based on a “Hamiltonian” approach.
- The details of the derivation are involved, but the final result is a method for determining L that is “quite simple.”
- The generality of the method is important to ensure that we don’t have to re-derive an approach to obtaining a steady-state solution for every situation that we encounter.

Deriving the Hamiltonian matrix

- In the optimal *control* problem, a matrix known as the “Hamiltonian matrix” comes about very naturally in one step of the solution.
- This matrix has certain desirable properties that aid finding solutions to the problem.

- Since control and estimation are "dual" problems, researchers have looked for–and found–certain ways of arranging estimation problems in a Hamiltonian format.
- Formulating an estimation problem in a Hamiltonian format is a little contrived, but the payoff is that it lends itself to a "simple" solution once you have reached that point.
- So, here goes... We start with the covariance update equations:

$$\begin{aligned}\Sigma_{\tilde{x},k}^{-} &= A\,\Sigma_{\tilde{x},k-1}^{+}A^T + \Sigma_{\tilde{w}} \\ \Sigma_{\tilde{x},k}^{+} &= \Sigma_{\tilde{x},k}^{-} - LC\,\Sigma_{\tilde{x},k}^{-} \\ &= \Sigma_{\tilde{x},k}^{-} - \Sigma_{\tilde{x},k}^{-}C^T\left(C\,\Sigma_{\tilde{x},k}^{-}C^T + \Sigma_{\tilde{v}}\right)^{-1} C\,\Sigma_{\tilde{x},k}^{-}.\end{aligned}$$

- We insert the second equation into the first, to get:

$$\Sigma_{\tilde{x},k+1}^{-} = A\,\Sigma_{\tilde{x},k}^{-}A^T - A\,\Sigma_{\tilde{x},k}^{-}C^T\left(C\,\Sigma_{\tilde{x},k}^{-}C^T + \Sigma_{\tilde{v}}\right)^{-1} C\,\Sigma_{\tilde{x},k}^{-}A^T + \Sigma_{\tilde{w}}.$$

- For the next pages, we will be interested in manipulating this equation solely for the purpose of solving for $\Sigma_{\tilde{x},ss}^{-}$. For ease of notation, we drop the superscript "$-$" and the subscript $\tilde{x}$. Rewriting,

$$\Sigma_{k+1} = A\,\Sigma_k A^T - A\,\Sigma_k C^T\left(C\,\Sigma_k C^T + \Sigma_{\tilde{v}}\right)^{-1} C\,\Sigma_k A^T + \Sigma_{\tilde{w}}.$$

- We can use the "matrix inversion lemma" (see appendix) to write

$$(C\,\Sigma_k C^T + \Sigma_{\tilde{v}})^{-1} = \Sigma_{\tilde{v}}^{-1} - \Sigma_{\tilde{v}}^{-1}C(C^T\Sigma_{\tilde{v}}^{-1}C + \Sigma_k^{-1})^{-1}C^T\Sigma_{\tilde{v}}^{-1}.$$

- Substituting this expression in the former equation, we get,

$$\begin{aligned}\Sigma_{k+1} = {} & A\,\Sigma_k A^T - A\,\Sigma_k C^T\Sigma_{\tilde{v}}^{-1}C\,\Sigma_k A^T + \\ & A\,\Sigma_k C^T\Sigma_{\tilde{v}}^{-1}C(C^T\Sigma_{\tilde{v}}^{-1}C + \Sigma_k^{-1})^{-1}C^T\Sigma_{\tilde{v}}^{-1}C\,\Sigma_k A^T + \Sigma_{\tilde{w}}.\end{aligned}$$

- Factoring out A and A^T from the beginning and end of the first three terms on the right hand side gives

$$\Sigma_{k+1} = A\big(\Sigma_k - \Sigma_k C^T \Sigma_{\tilde{\upsilon}}^{-1} C \Sigma_k + \Sigma_k C^T \Sigma_{\tilde{\upsilon}}^{-1} C (C^T \Sigma_{\tilde{\upsilon}}^{-1} C + \Sigma_k^{-1})^{-1} C^T \Sigma_{\tilde{\upsilon}}^{-1} C \Sigma_k\big) A^T + \Sigma_{\tilde{w}}.$$

- Factoring out $\Sigma_k C^T \Sigma_{\tilde{\upsilon}}^{-1} C$ from the early terms then gives

$$\Sigma_{k+1} = A\big(\Sigma_k - \Sigma_k C^T \Sigma_{\tilde{\upsilon}}^{-1} C \left[\Sigma_k - (C^T \Sigma_{\tilde{\upsilon}}^{-1} C + \Sigma_k^{-1})^{-1} C^T \Sigma_{\tilde{\upsilon}}^{-1} C \Sigma_k\right]\big) A^T + \Sigma_{\tilde{w}}.$$

- Additionally, factoring out $(C^T \Sigma_{\tilde{\upsilon}}^{-1} C + \Sigma_k^{-1})^{-1}$ gives

$$\Sigma_{k+1} = A\big(\Sigma_k - \Sigma_k C^T \Sigma_{\tilde{\upsilon}}^{-1} C (C^T \Sigma_{\tilde{\upsilon}}^{-1} C + \Sigma_k^{-1})^{-1} \times \left[(C^T \Sigma_{\tilde{\upsilon}}^{-1} C + \Sigma_k^{-1}) \Sigma_k - C^T \Sigma_{\tilde{\upsilon}}^{-1} C \Sigma_k\right]\big) A^T + \Sigma_{\tilde{w}}.$$

- Collapsing the content of the square brackets down to I:

$$\Sigma_{k+1} = A\big(\Sigma_k - \Sigma_k C^T \Sigma_{\tilde{\upsilon}}^{-1} C (C^T \Sigma_{\tilde{\upsilon}}^{-1} C + \Sigma_k^{-1})^{-1}\big) A^T + \Sigma_{\tilde{w}}.$$

- Factoring out an Σ_k on the left,

$$\Sigma_{k+1} = A \Sigma_k \big(I - C^T \Sigma_{\tilde{\upsilon}}^{-1} C (C^T \Sigma_{\tilde{\upsilon}}^{-1} C + \Sigma_k^{-1})^{-1}\big) A^T + \Sigma_{\tilde{w}}.$$

- Factoring out $(C^T \Sigma_{\tilde{\upsilon}}^{-1} C + \Sigma_k^{-1})^{-1}$ on the right,

$$\Sigma_{k+1} = A \Sigma_k \big[(C^T \Sigma_{\tilde{\upsilon}}^{-1} C + \Sigma_k^{-1}) - C^T \Sigma_{\tilde{\upsilon}}^{-1} C\big] (C^T \Sigma_{\tilde{\upsilon}}^{-1} C + \Sigma_k^{-1})^{-1} A^T + \Sigma_{\tilde{w}}$$

- Condensing the contents within the square brackets to Σ_k^{-1} and combining with Σ_k,

$$\Sigma_{k+1} = A (C^T \Sigma_{\tilde{\upsilon}}^{-1} C + \Sigma_k^{-1})^{-1} A^T + \Sigma_{\tilde{w}}.$$

- Factoring out a Σ_k and multiplying $\Sigma_{\tilde{w}}$ by I,

$$\Sigma_{k+1} = A \Sigma_k (C^T \Sigma_{\tilde{\upsilon}}^{-1} C \Sigma_k + I)^{-1} A^T + \Sigma_{\tilde{w}} A^{-T} A^T.$$

- Writing as a product of two terms

$$\Sigma_{k+1} = \left[A\Sigma_k + \Sigma_{\widetilde{w}}A^{-T}(C^T\Sigma_{\tilde{v}}^{-1}C\Sigma_k + I)\right](C^T\Sigma_{\tilde{v}}^{-1}C\Sigma_k + I)^{-1}A^T.$$

- Rewriting slightly,

$$\Sigma_{k+1} = \left[(A + \Sigma_{\widetilde{w}}A^{-T}C^T\Sigma_{\tilde{v}}^{-1}C)\Sigma_k + \Sigma_{\widetilde{w}}A^{-T}\right](C^T\Sigma_{\tilde{v}}^{-1}C\Sigma_k + I)^{-1}A^T.$$

- Now, suppose that Σ_k can be factored as

$$\Sigma_k = S_k Z_k^{-1}$$

  where S_k and Z_k both have the same dimensions as Σ_k.

- Making this substitution gives

$$\begin{aligned}
\Sigma_{k+1} &= \left[(A + \Sigma_{\widetilde{w}}A^{-T}C^T\Sigma_{\tilde{v}}^{-1}C)S_k + \Sigma_{\widetilde{w}}A^{-T}Z_k\right]Z_k^{-1} \times \\
&\qquad (C^T\Sigma_{\tilde{v}}^{-1}CS_kZ_k^{-1} + I)^{-1}A^T \\
&= \left[(A + \Sigma_{\widetilde{w}}A^{-T}C^T\Sigma_{\tilde{v}}^{-1}C)S_k + \Sigma_{\widetilde{w}}A^{-T}Z_k\right](C^T\Sigma_{\tilde{v}}^{-1}CS_k + Z_k)^{-1}A^T \\
&= \left[(A + \Sigma_{\widetilde{w}}A^{-T}C^T\Sigma_{\tilde{v}}^{-1}C)S_k + \Sigma_{\widetilde{w}}A^{-T}Z_k\right] \times \\
&\qquad (A^{-T}C^T\Sigma_{\tilde{v}}^{-1}CS_k + A^{-T}Z_k)^{-1} \\
&= S_{k+1}Z_{k+1}^{-1}.
\end{aligned}$$

- This shows that

$$\begin{aligned}
S_{k+1} &= (A + \Sigma_{\widetilde{w}}A^{-T}C^T\Sigma_{\tilde{v}}^{-1}C)S_k + \Sigma_{\widetilde{w}}A^{-T}Z_k \\
Z_{k+1} &= A^{-T}C^T\Sigma_{\tilde{v}}^{-1}CS_k + A^{-T}Z_k.
\end{aligned}$$

- These equations for S_{k+1} and Z_{k+1} can be written as the following single equation

$$\begin{bmatrix} Z_{k+1} \\ S_{k+1} \end{bmatrix} = \begin{bmatrix} A^{-T} & A^{-T}C^T\Sigma_{\tilde{v}}^{-1}C \\ \Sigma_{\widetilde{w}}A^{-T} & A + \Sigma_{\widetilde{w}}A^{-T}C^T\Sigma_{\tilde{v}}^{-1}C \end{bmatrix} \begin{bmatrix} Z_k \\ S_k \end{bmatrix} = \mathcal{H}\begin{bmatrix} Z_k \\ S_k \end{bmatrix}.$$

- If the covariance matrix Σ_k is an $n \times n$ matrix, then $\mathcal{H}$ will be $2n \times 2n$.
- The matrix $\mathcal{H}$ is called the Hamiltonian matrix, and is a symplectic matrix; that is, it satisfies the equation

$$J^{-1}\mathcal{H}^T J = \mathcal{H}^{-1} \quad \text{where } J = \begin{bmatrix} 0 & I \\ -I & 0 \end{bmatrix}.$$

- Symplectic matrices have the following properties:
  - None of the eigenvalues of a symplectic matrix are equal to zero.
  - If λ is an eigenvalue of a symplectic matrix, then so too is $1/\lambda$.
  - The determinant of a symplectic matrix is equal to ± 1.

4.9: Steady state: Solving for $\Sigma^{\pm}_{\tilde{x},ss}$ using the Hamiltonian matrix

- Given the known system description via the matrices A, C, $\Sigma_{\tilde{w}}$, $\Sigma_{\tilde{v}}$, it is a straightforward matter to compute the Hamiltonian matrix $\mathcal{H}$. But, what to do with it?
- We'll see that the coupled recursion on S_k and Z_k can be solved for steady state values of S_{ss} and Z_{ss}, allowing us to compute $\Sigma^{\pm}_{\tilde{x},ss}$.
- Assuming that $\mathcal{H}$ has no eigenvalues with magnitude exactly equal to one, then half of its eigenvalues will be inside the unit circle (stable) and half will be outside the unit circle (unstable).
- We define Λ as the diagonal matrix containing all unstable eigenvalues of $\mathcal{H}$. Then, $\mathcal{H}$ can be written as

$$\mathcal{H} = \Psi \begin{bmatrix} \Lambda^{-1} & 0 \\ 0 & \Lambda \end{bmatrix} \Psi^{-1} = \Psi D \Psi^{-1}.$$

- We partition the eigenvector matrix Ψ into four $n \times n$ blocks

$$\Psi = \begin{bmatrix} \Psi_{11} & \Psi_{12} \\ \Psi_{21} & \Psi_{22} \end{bmatrix}.$$

  - The left $2n \times n$ submatrix of Ψ contains the eigenvectors of $\mathcal{H}$ that correspond to the stable eigenvalues.
  - The right $2n \times n$ submatrix of Ψ contains the eigenvectors of $\mathcal{H}$ that correspond to the unstable eigenvalues.
- We can now write

$$\begin{bmatrix} Z_{k+1} \\ S_{k+1} \end{bmatrix} = \underbrace{\Psi D \Psi^{-1}}_{\mathcal{H}} \begin{bmatrix} Z_k \\ S_k \end{bmatrix}$$

$$\Psi^{-1} \begin{bmatrix} Z_{k+1} \\ S_{k+1} \end{bmatrix} = D \Psi^{-1} \begin{bmatrix} Z_k \\ S_k \end{bmatrix}.$$

- Now, define the $n \times n$ matrices $Y_{1,k}$ and $Y_{2,k}$ and the $2n \times n$ matrix Y_k:

$$Y_k = \begin{bmatrix} Y_{1,k} \\ Y_{2,k} \end{bmatrix} = \Psi^{-1} \begin{bmatrix} Z_k \\ S_k \end{bmatrix}.$$

- We now have a simple recursion

$$Y_{k+1} = DY_k$$

$$\begin{bmatrix} Y_{1,k+1} \\ Y_{2,k+1} \end{bmatrix} = \begin{bmatrix} \Lambda^{-1} & 0 \\ 0 & \Lambda \end{bmatrix} \begin{bmatrix} Y_{1,k} \\ Y_{2,k} \end{bmatrix}.$$

- From these equations, we see

$$Y_{1,k+1} = \Lambda^{-1} Y_{1,k} \qquad \text{and} \qquad Y_{2,k+1} = \Lambda Y_{2,k}$$

- So, taking k steps after time zero,

$$Y_{1,k} = \Lambda^{-k} Y_{1,0} \qquad \text{and} \qquad Y_{2,k} = \Lambda^{k} Y_{2,0}.$$

- Using this solution for $Y_{1,k}$ and $Y_{2,k}$ we can go back and solve:

$$\begin{bmatrix} Y_{1,k} \\ Y_{2,k} \end{bmatrix} = \Psi^{-1} \begin{bmatrix} Z_k \\ S_k \end{bmatrix}$$

$$\begin{bmatrix} Z_k \\ S_k \end{bmatrix} = \begin{bmatrix} \Psi_{11} & \Psi_{12} \\ \Psi_{21} & \Psi_{22} \end{bmatrix} \begin{bmatrix} Y_{1,k} \\ Y_{2,k} \end{bmatrix}$$

$$= \begin{bmatrix} \Psi_{11} & \Psi_{12} \\ \Psi_{21} & \Psi_{22} \end{bmatrix} \begin{bmatrix} \Lambda^{-k} Y_{1,0} \\ \Lambda^{k} Y_{2,0} \end{bmatrix}.$$

- As k increases, the Λ^{-k} matrix approaches zero (because it is a diagonal matrix whose elements are all less than one in magnitude).
- Therefore, for large k we obtain

$$\begin{bmatrix} Z_k \\ S_k \end{bmatrix} = \begin{bmatrix} \Psi_{11} & \Psi_{12} \\ \Psi_{21} & \Psi_{22} \end{bmatrix} \begin{bmatrix} 0 \\ \Lambda^{k} Y_{2,0} \end{bmatrix}$$

$$Z_k = \Psi_{12} Y_{2,k}$$

$$S_k = \Psi_{22} Y_{2,k}.$$

- Solving for S_k in these last two equations gives $S_k = \Psi_{22}\Psi_{12}^{-1} Z_k$, but we know that $S_k = \Sigma_{\tilde{x},k}^{-} Z_k$, so

$$\lim_{k\to\infty} \Sigma_{\tilde{x},k}^{-} = \Psi_{22}\Psi_{12}^{-1}.$$

Summarizing the Hamiltonian method:

- Form the $2n \times 2n$ Hamiltonian matrix

$$\mathcal{H} = \begin{bmatrix} A^{-T} & A^{-T} C^T \Sigma_{\tilde{v}}^{-1} C \\ \Sigma_{\tilde{w}} A^{-T} & A + \Sigma_{\tilde{w}} A^{-T} C^T \Sigma_{\tilde{v}}^{-1} C \end{bmatrix}.$$

- Compute the eigensystem of $\mathcal{H}$.
- Put the n eigenvectors corresponding to the n unstable eigenvalues in a matrix partitioned as

$$\begin{bmatrix} \Psi_{12} \\ \Psi_{22} \end{bmatrix}.$$

- Compute the steady-state solution to the prediction covariance as

$$\Sigma_{\tilde{x}}^{-} = \Psi_{22}\Psi_{12}^{-1}.$$

- Compute the steady-state Kalman gain vector as

$$L = \Sigma_{\tilde{x}}^{-} C^T \left(C \Sigma_{\tilde{x}}^{-} C^T + \Sigma_{\tilde{v}} \right)^{-1}.$$

- Compute the steady-state solution to the estimation covariance as

$$\Sigma_{\tilde{x}}^{+} = \Sigma_{\tilde{x}}^{-} - LC\Sigma_{\tilde{x}}^{-}.$$

Solving for steady-state solution in MATLAB

- If you really want to cheat, `dkalman`, `dlqe`, or `dare`.

Example from before leading to steady-state solution

- Recall the system description from before, with $x_{k+1} = x_k + w_k$. $z_k = x_k + v_k$, $\mathbb{E}[w_k] = \mathbb{E}[v_k] = 0$, $\Sigma_{\tilde{w}} = 1$, and $\Sigma_{\tilde{v}} = 2$.
- Therefore, $A = 1$ and $C = 1$.
- We first form the $2n \times 2n$ Hamiltonian matrix

$$\mathcal{H} = \begin{bmatrix} A^{-T} & A^{-T} C^T \Sigma_{\tilde{v}}^{-1} C \\ \Sigma_{\tilde{w}} A^{-T} & A + \Sigma_{\tilde{w}} A^{-T} C^T \Sigma_{\tilde{v}}^{-1} C \end{bmatrix} = \begin{bmatrix} 1 & 0.5 \\ 1 & 1.5 \end{bmatrix}.$$

- Compute the eigensystem of $\mathcal{H}$:

$$\Psi = \begin{bmatrix} -1/\sqrt{2} & -1/\sqrt{5} \\ 1/\sqrt{2} & -2/\sqrt{5} \end{bmatrix} \quad \text{and} \quad \Lambda = \begin{bmatrix} 0.5 & 0 \\ 0 & 2 \end{bmatrix}.$$

- Put the n eigenvectors corresponding to the n unstable eigenvalues in a matrix partitioned as

$$\begin{bmatrix} \Psi_{12} \\ \Psi_{22} \end{bmatrix} = \begin{bmatrix} -1/\sqrt{5} \\ -2/\sqrt{5} \end{bmatrix}.$$

- Compute the steady-state solution to the prediction covariance as

$$\Sigma_{\tilde{x}}^{-} = \Psi_{22} \Psi_{12}^{-1} = -2/\sqrt{5} \times \sqrt{5}/-1 = 2.$$

- Compute the steady-state Kalman gain vector as

$$L = \Sigma_{\tilde{x}}^{-} C^T \left(C \Sigma_{\tilde{x}}^{-} C^T + \Sigma_{\tilde{v}} \right)^{-1} = 2/4 = 1/2.$$

- Compute the steady-state solution to the estimation covariance as

$$\Sigma_{\tilde{x}}^{+} = \Sigma_{\tilde{x}}^{-} - LC \Sigma_{\tilde{x}}^{-} = 2 - 2/2 = 1.$$

- These results agree with our example from before.

4.10*: Initializing the filter

- There are several practical steps that we have thus far glossed over, but need to be addressed in an implementation.
- The first is, "how do we initialize the filter before entering the processing loop?"[1]
  - That is, what values to use for $\hat{x}_0^+$, $\Sigma_{\tilde{x},0}^+$?
  - The "correct" answer is,

$$\hat{x}_0^+ = \mathbb{E}[x_0] \quad \text{and} \quad \Sigma_{\tilde{x},0}^+ = \mathbb{E}[(x_0 - \hat{x}_0)(x_0 - \hat{x}_0)^T]$$

    but, often we don't know what these expectations are in practice.
  - So, we'll settle for "good values."
- Assume that we have available a set of measured system outputs $\{z_q, \ldots, z_p\}$ where $q \leq p$ and $q \leq 0$ and that we similarly have a set of measured inputs u_k, $k \in \{q, \ldots, 0\}$.
- We can write, for general k,

$$x_k = \Phi_k x_0 + \Phi_{u,k} + \Phi_{w,k},$$

  - Φ_k is the "state transition matrix" from time 0 to time k,
  - $\Phi_{u,k}$ is the accumulated effect of the input signal on the state between time 0 and k, and
  - $\Phi_{w,k}$ is the accumulated effect of the process noise signal on the state between time 0 and k (more on how to compute these later).

[1] This discussion is simplified from: X. Rong Li and Chen He, *Optimal Initialization of Linear Recursive Filters,* Proceedings of the 37th IEEE Conference on Decision and Control, Tampa, FL, Dec. 1998, pp. 2335–2340. For additional generality and rigor, see the original paper.

- We substitute this expression into the system output equation and get

$$\begin{aligned} z_k &= C_k\Phi_k x_0 + C_k\Phi_{u,k} + C_k\Phi_{w,k} + D_k u_k + v_k \\ &= C_k\Phi_k x_0 + C_k\Phi_{u,k} + C_k\Phi_{\bar{w},k} + C_k\Phi_{\tilde{w},k} + D_k u_k + \bar{v}_k + \tilde{v}_k, \end{aligned}$$

  where we have broken the process noise and sensor noise contributions into a deterministic part (based on the noise mean) and a random part (based on the variation).

- We can augment all measured data (stacking equations into matrices and vectors)

$$\begin{bmatrix} z_p \\ \vdots \\ z_q \end{bmatrix} = \begin{bmatrix} C_p\Phi_p \\ \vdots \\ C_q\Phi_q \end{bmatrix} x_0 + \begin{bmatrix} C_p(\Phi_{u,p} + \Phi_{\bar{w},p}) + D_p u_p + \bar{v}_p \\ \vdots \\ C_q(\Phi_{u,q} + \Phi_{\bar{w},q}) + D_q u_q + \bar{v}_q \end{bmatrix} + \begin{bmatrix} C_p\Phi_{\tilde{w},p} + \tilde{v}_p \\ \vdots \\ C_q\Phi_{\tilde{w},q} + \tilde{v}_q \end{bmatrix}.$$

- Re-arranging,

$$\underbrace{\begin{bmatrix} z_p \\ \vdots \\ z_q \end{bmatrix} - \begin{bmatrix} C_p(\Phi_{u,p} + \Phi_{\bar{w},p}) + D_p u_p + \bar{v}_p \\ \vdots \\ C_q(\Phi_{u,q} + \Phi_{\bar{w},q}) + D_q u_q + \bar{v}_q \end{bmatrix}}_{\text{known}} = \underbrace{\begin{bmatrix} C_p\Phi_p \\ \vdots \\ C_q\Phi_q \end{bmatrix}}_{\text{known}} x_0 + \underbrace{\begin{bmatrix} C_p\Phi_{\tilde{w},p} + \tilde{v}_p \\ \vdots \\ C_q\Phi_{\tilde{w},q} + \tilde{v}_q \end{bmatrix}}_{\text{zero-mean, random}}$$

$$\widetilde{Z} = \widetilde{C}x_0 + \widetilde{V}.$$

- We would like to use the measured information and the statistics of the noise processes to come up with a good estimate of x_0.
- Intuitively, we would like to place more emphasis on measurements having the least noise, and less emphasis on measurements having greater noise.
- A logical way to do this is to weigh the value of measurements according to the inverse of the covariance matrix of $\widetilde{V}$.
  - A "weighted least squares" optimization approach to do so is,

$$\begin{aligned}\hat{x}_0 &= \arg\min_{x_0} J(x_0) \\ &= \arg\min_{x_0} \left(\widetilde{Z} - \widetilde{C}x_0\right)^T \Sigma_{\widetilde{V}}^{-1} \left(\widetilde{Z} - \widetilde{C}x_0\right) \\ &= \arg\min_{x_0} \widetilde{Z}^T \Sigma_{\widetilde{V}}^{-1} \widetilde{Z} - x_0^T \widetilde{C}^T \Sigma_{\widetilde{V}}^{-1} \widetilde{Z} - \widetilde{Z}^T \Sigma_{\widetilde{V}}^{-1} \widetilde{C} x_0 + x_0^T \widetilde{C}^T \Sigma_{\widetilde{V}}^{-1} \widetilde{C} x_0.\end{aligned}$$

- We find x_0 by taking the vector derivative of the cost function $J(x_0)$ with respect to x_0, and setting the result to zero.

NOTE: The following are identities from vector calculus,

$$\frac{\mathrm{d}}{\mathrm{d}X} Y^T X = Y, \qquad \frac{\mathrm{d}}{\mathrm{d}X} X^T Y = Y, \quad \text{and} \quad \frac{\mathrm{d}}{\mathrm{d}X} X^T A X = (A + A^T)X.$$

- Therefore,

$$\frac{\mathrm{d}J(x_0)}{\mathrm{d}x_0} = -\widetilde{C}^T \Sigma_{\widetilde{V}}^{-1} \widetilde{Z} - \widetilde{C}^T \Sigma_{\widetilde{V}}^{-1} \widetilde{Z} + 2\widetilde{C}^T \Sigma_{\widetilde{V}}^{-1} \widetilde{C} x_0 = 0$$

$$\hat{x}_0 = \left(\widetilde{C}^T \Sigma_{\widetilde{V}}^{-1} \widetilde{C}\right)^{-1} \left(\widetilde{C}^T \Sigma_{\widetilde{V}}^{-1}\right) \widetilde{Z}.$$

- So, assuming that we can compute $\widetilde{C}$, $\widetilde{Z}$, and $\Sigma_{\widetilde{V}}^{-1}$, we have solved for the initial state. We still need the initial covariance.

- Start with the definition for the estimate error: $\tilde{x}_0 = x_0 - \hat{x}_0$,

$$\begin{aligned}
\tilde{x}_0 &= x_0 - \hat{x}_0 \\
&= x_0 - \left(\widetilde{C}^T \Sigma_{\widetilde{V}}^{-1} \widetilde{C}\right)^{-1} \left(\widetilde{C}^T \Sigma_{\widetilde{V}}^{-1}\right) \left(\widetilde{C} x_0 + \widetilde{V}\right) \\
&= x_0 - \underbrace{\left(\widetilde{C}^T \Sigma_{\widetilde{V}}^{-1} \widetilde{C}\right)^{-1} \left(\widetilde{C}^T \Sigma_{\widetilde{V}}^{-1} \widetilde{C}\right)}_{I} x_0 - \left(\widetilde{C}^T \Sigma_{\widetilde{V}}^{-1} \widetilde{C}\right)^{-1} \left(\widetilde{C}^T \Sigma_{\widetilde{V}}^{-1}\right) \widetilde{V} \\
&= - \left(\widetilde{C}^T \Sigma_{\widetilde{V}}^{-1} \widetilde{C}\right)^{-1} \left(\widetilde{C}^T \Sigma_{\widetilde{V}}^{-1}\right) \widetilde{V}.
\end{aligned}$$

- Next,

$$\begin{aligned}
\Sigma_{\tilde{x},0} &= \mathbb{E}\left[\tilde{x}_0 \tilde{x}_0^T\right] \\
&= \left(\widetilde{C}^T \Sigma_{\widetilde{V}}^{-1} \widetilde{C}\right)^{-1} \left(\widetilde{C}^T \Sigma_{\widetilde{V}}^{-1}\right) \underbrace{\mathbb{E}\left[\widetilde{V} \widetilde{V}^T\right]}_{\Sigma_{\widetilde{V}}} \Sigma_{\widetilde{V}}^{-1} \widetilde{C} \left(\widetilde{C}^T \Sigma_{\widetilde{V}}^{-1} \widetilde{C}\right)^{-1}. \\
&= \left(\widetilde{C}^T \Sigma_{\widetilde{V}}^{-1} \widetilde{C}\right)^{-1}.
\end{aligned}$$

Summary to date: Using data measured from the system being observed, and statistics known about the noises, we can form $\widetilde{C}, \widetilde{Z}$, and $\Sigma_{\widetilde{V}}^{-1}$, and then initialize the Kalman filter with

$$\begin{aligned}
\Sigma_{\tilde{x},0} &= \left(\widetilde{C}^T \Sigma_{\widetilde{V}}^{-1} \widetilde{C}\right)^{-1} \\
\hat{x}_0 &= \Sigma_{\tilde{x},0} \widetilde{C}^T \Sigma_{\widetilde{V}}^{-1} \widetilde{Z}.
\end{aligned}$$

Computing $\widetilde{C}$ and $\widetilde{Z}$

- The model matrices $C_p \ldots C_q$ are assumed known, so all we need to compute are Φ_k, $\Phi_{\bar{w},k}$, and $\Phi_{u,k}$.
- Starting with $x_{k+1} = A_k x_k + B_k u_k + w_k$ it can be shown that, for $m \geq k$,[2]

[2] Note: The way we are using products and summations here is somewhat non-standard. If the upper limit of a product is less than the lower limit, the product is the identity matrix; if the upper limit of a sum is less than the lower limit, the sum is zero.

$$x_m = \left(\prod_{j=0}^{m-k-1} A_{m-1-j} \right) x_k + \sum_{i=k}^{m-1} \left(\prod_{j=0}^{m-i-2} A_{m-1-j} \right) (B_i u_i + w_i) .$$

- So, for $m = 0$ and $k \leq 0$,

$$x_0 = \left(\prod_{j=0}^{-k-1} A_{-1-j} \right) x_k + \sum_{i=k}^{-1} \left(\prod_{j=0}^{-i-2} A_{-1-j} \right) (B_i u_i + w_i) .$$

- Assuming that A_k^{-1}exists for all $k \leq 0$, we get

$$x_k = \underbrace{\left(\prod_{j=0}^{-k-1} A_{-1-j} \right)^{-1}}_{\Phi_k} x_0 + \underbrace{\left(-\sum_{i=k}^{-1} \left(\prod_{j=0}^{-k-1} A_{-1-j} \right)^{-1} \left(\prod_{j=0}^{-i-2} A_{-1-j} \right) B_i u_i \right)}_{\Phi_{u,k}}$$

$$+ \underbrace{\left(-\sum_{i=k}^{-1} \left(\prod_{j=0}^{-k-1} A_{-1-j} \right)^{-1} \left(\prod_{j=0}^{-i-2} A_{-1-j} \right) \bar{w}_i \right)}_{\Phi_{\bar{w},k}}$$

$$+ \underbrace{\left(-\sum_{i=k}^{-1} \left(\prod_{j=0}^{-k-1} A_{-1-j} \right)^{-1} \left(\prod_{j=0}^{-i-2} A_{-1-j} \right) \widetilde{w}_i \right)}_{\Phi_{\widetilde{w},k}}$$

- So, having Φ_k, $\Phi_{u,k}$, and $\Phi_{\bar{w},k}$, we can now compute $\widetilde{C}$ and $\widetilde{Z}$.

Computing $\Sigma_{\widetilde{V}}$

- Note that, by construction $\widetilde{V}$ has zero mean, so

$$\Sigma_{\widetilde{V}} = \mathbb{E}[\widetilde{V}\widetilde{V}^T]$$

$$= \mathbb{E}\left[\left(\begin{bmatrix} C_p\Phi_{\widetilde{w},p} \\ \vdots \\ C_q\Phi_{\widetilde{w},q} \end{bmatrix} + \begin{bmatrix} \tilde{v}_p \\ \vdots \\ \tilde{v}_q \end{bmatrix}\right)\left(\begin{bmatrix} C_p\Phi_{\widetilde{w},p} \\ \vdots \\ C_q\Phi_{\widetilde{w},q} \end{bmatrix} + \begin{bmatrix} \tilde{v}_p \\ \vdots \\ \tilde{v}_q \end{bmatrix}\right)^T\right]$$

$$= \begin{bmatrix} C_p\mathbb{E}[\Phi_{\widetilde{w},p}\Phi_{\widetilde{w},p}^T]C_p^T & \cdots & C_p\mathbb{E}[\Phi_{\widetilde{w},p}\Phi_{\widetilde{w},q}^T]C_q^T \\ \vdots & & \vdots \\ C_q\mathbb{E}[\Phi_{\widetilde{w},q}\Phi_{\widetilde{w},p}^T]C_p^T & \cdots & C_q\mathbb{E}[\Phi_{\widetilde{w},q}\Phi_{\widetilde{w},q}^T]C_q^T \end{bmatrix} +$$

$$\begin{bmatrix} \Sigma_{\tilde{v},p} & & 0 \\ & \ddots & \\ 0 & & \Sigma_{\tilde{v},q} \end{bmatrix}.$$

Some simplifications when $A_k = A$, $B_k = B$, etc

- When the system model is time invariant, many simplifications may be made.
- The state transition matrix becomes,

$$\Phi_k = \left(\prod_{j=0}^{-k-1} A\right)^{-1} = \begin{cases} (A^{-k})^{-1}, & k < 0 \\ I, & \text{else.} \end{cases}$$

- Within the calculations for $\Phi_{u,k}$ and $\Phi_{w,k}$ we must compute

$$B_{k,i} = -\left(\prod_{j=0}^{-k-1} A\right)^{-1}\left(\prod_{j=0}^{-i-2} A\right) = -\prod_{j=-i-1}^{-k-1} A^{-1} = \begin{cases} -(A^{i-k+1})^{-1}, & k \le i \\ I, & \text{else.} \end{cases}$$

- Therefore,

$$\Phi_{u,k} = \sum_{i=k}^{-1} B_{k,i}Bu_i, \qquad \Phi_{\bar{w},k} = \sum_{i=k}^{-1} B_{k,i}\bar{w}_i, \qquad \Phi_{\widetilde{w},k} = \sum_{i=k}^{-1} B_{k,i}\widetilde{w}_i.$$

- From this we can find,

$$\widetilde{C} = \begin{bmatrix} C\Phi_p \\ \vdots \\ C\Phi_q \end{bmatrix} = \begin{bmatrix} C(A^{-p})^{-1} \\ \vdots \\ C(A^{-q})^{-1} \end{bmatrix}$$

$$\widetilde{Z} = \begin{bmatrix} z_p - C\left(\Phi_{u,p} + \Phi_{\bar{w},p}\right) \\ \vdots \\ z_q - C\left(\Phi_{u,q} + \Phi_{\bar{w},q}\right) \end{bmatrix} = \begin{bmatrix} z_p + C\sum_{i=p}^{-1}\left(A^{i-p+1}\right)^{-1}(Bu_i + \bar{w}) \\ \vdots \\ z_q + C\sum_{i=q}^{-1}\left(A^{i-q+1}\right)^{-1}(Bu_i + \bar{w}) \end{bmatrix}.$$

Example for NCV model

- Let's try to make some sense out of all this abstract math by coming up with a concrete initialization method for a simple NCV model:

$$x_{k+1} = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} x_k + \begin{bmatrix} T^2/2 \\ T \end{bmatrix} u_k + w_k$$

$$z_k = \begin{bmatrix} 1 & 0 \end{bmatrix} x_k + v_k.$$

- For simplicity, we assume that $u_k = 0$ and that both w_k and v_k are zero mean. $\Sigma_{\tilde{v}} = r$ and

$$\Sigma_{\widetilde{w}} = q \begin{bmatrix} T^4/4, & T^3/2 \\ T^3/2, & T^2 \end{bmatrix}.$$

NOTE: We arrived at $\Sigma_{\widetilde{w}}$ by assuming that scalar white noise having covariance q is passed through an input matrix $B_w = B$. Then, $\Sigma_{\widetilde{w}} = BqB^T$.

- We make two measurements, z_{-1} and z_0, and wish to use these to initialize $\hat{x}_0$ and $\Sigma_{\tilde{x},0}$.

- To compute $\widetilde{C}$, we must first determine Φ_0 and Φ_{-1}:

$$\Phi_0 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \qquad \Phi_{-1} = A^{-1} = \begin{bmatrix} 1 & -T \\ 0 & 1 \end{bmatrix},$$

$$\widetilde{C} = \begin{bmatrix} C\Phi_0 \\ C\Phi_{-1} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 1 & -T \end{bmatrix}.$$

- To compute $\widetilde{Z}$, we must also determine $\Phi_{u,0}$, $\Phi_{u,-1}$, $\Phi_{\bar{w},0}$, and $\Phi_{\bar{w},-1}$.
- Since the input is assumed to be zero, and the process noise is assumed to have zero mean, these four quantities are all zero.
- Therefore,

$$\widetilde{Z} = \begin{bmatrix} z_0 \\ z_{-1} \end{bmatrix}.$$

- To compute $\Sigma_{\widetilde{V}}$, we must first compute $\Phi_{\widetilde{w},0}$, and $\Phi_{\widetilde{w},-1}$.

$$\Phi_{\widetilde{w},0} = \sum_{i=0}^{-1} B_{0,i}\widetilde{w}_0 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$\Phi_{\widetilde{w},-1} = \sum_{i=-1}^{-1} B_{-1,i}\widetilde{w}_{-1} = -A^{-1}\widetilde{w}_{-1}.$$

- Continuing with the evaluation of $\Sigma_{\widetilde{V}}$,

$$\begin{aligned}
\Sigma_{\widetilde{V}} &= \begin{bmatrix} C\mathbb{E}\left[\Phi_{\widetilde{w},0}\Phi_{\widetilde{w},0}^T\right]C^T, & C\mathbb{E}\left[\Phi_{\widetilde{w},0}\Phi_{\widetilde{w},-1}^T\right]C^T \\ C\mathbb{E}\left[\Phi_{\widetilde{w},-1}\Phi_{\widetilde{w},0}^T\right]C^T, & C\mathbb{E}\left[\Phi_{\widetilde{w},-1}\Phi_{\widetilde{w},-1}^T\right]C^T \end{bmatrix} + \begin{bmatrix} r & 0 \\ 0 & r \end{bmatrix} \\
&= \begin{bmatrix} 0 & 0 \\ 0 & s \end{bmatrix} + \begin{bmatrix} r & 0 \\ 0 & r \end{bmatrix} \\
&= \begin{bmatrix} r & 0 \\ 0 & r+s \end{bmatrix},
\end{aligned}$$

where

$$\begin{aligned} s &= CA^{-1}\Sigma_{\widetilde{w}}A^{-T}C^T \\ &= q\begin{bmatrix} 1 & 0 \end{bmatrix}\begin{bmatrix} 1 & -T \\ 0 & 1 \end{bmatrix}\begin{bmatrix} T^4/4, & T^3/2 \\ T^3/2, & T^2 \end{bmatrix}\begin{bmatrix} 1 & 0 \\ -T & 1 \end{bmatrix}\begin{bmatrix} 1 \\ 0 \end{bmatrix} \\ &= qT^4/4. \end{aligned}$$

- Therefore,

$$\Sigma_{\widetilde{V}} = \begin{bmatrix} r & 0 \\ 0 & r + qT^4/4 \end{bmatrix}.$$

- Now that we have computed all the basic quantities, we can proceed to evaluate $\Sigma^+_{\tilde{x},0}$ and $\hat{x}_0$.

$$\begin{aligned} \Sigma^+_{\tilde{x},0} &= \left(\widetilde{C}^T\Sigma^{-1}_{\widetilde{V}}\widetilde{C}\right)^{-1} = \widetilde{C}^{-1}\Sigma_{\widetilde{V}}\widetilde{C}^{-T} \\ &= \begin{bmatrix} 1 & 0 \\ 1/T, & -1/T \end{bmatrix}\begin{bmatrix} r & 0 \\ 0 & r + qT^4/4 \end{bmatrix}\begin{bmatrix} 1 & 1/T \\ 0 & -1/T \end{bmatrix} \\ &= \begin{bmatrix} r & \dfrac{r}{T} \\ \dfrac{r}{T} & \dfrac{2r + qT^4/4}{T^2} \end{bmatrix} \\ \hat{x}_0 &= \Sigma^+_{\tilde{x},0}\widetilde{C}^T\Sigma^{-1}_{\widetilde{V}}\widetilde{Z} \\ &= \begin{bmatrix} r & \dfrac{r}{T} \\ \dfrac{r}{T} & \dfrac{2r + qT^4/4}{T^2} \end{bmatrix}\begin{bmatrix} 1 & 1 \\ 0 & -T \end{bmatrix}\begin{bmatrix} \dfrac{1}{r} & 0 \\ 0 & \dfrac{1}{r + qT^4/4} \end{bmatrix}\begin{bmatrix} z_0 \\ z_{-1} \end{bmatrix} \\ &= \begin{bmatrix} r & 0 \\ \dfrac{r}{T} & \dfrac{-(r + qT^4/4)}{T} \end{bmatrix}\begin{bmatrix} \dfrac{1}{r} & 0 \\ 0 & \dfrac{1}{r + qT^4/4} \end{bmatrix}\begin{bmatrix} z_0 \\ z_{-1} \end{bmatrix} \end{aligned}$$

$$= \begin{bmatrix} 1 & 0 \\ \dfrac{1}{T} & \dfrac{-1}{T} \end{bmatrix} \begin{bmatrix} z_0 \\ z_{-1} \end{bmatrix} = \begin{bmatrix} z_0 \\ \dfrac{z_0 - z_{-1}}{T} \end{bmatrix}.$$

- Notice that the position state is initialized to the last measured position, and the velocity state is initialized to a discrete approximation of the velocity. Makes sense.
- Notice also that the position covariance is simply the sensor-noise covariance; the velocity covariance has contribution due to sensor noise plus process noise (to take into account possible accelerations).
- The algebra required to arrive at these initializations was involved, but notice that $\Sigma^{+}_{\tilde{x},0}$ may be initialized with known constant values without needing measurements, and $\hat{x}_0$ is trivial to compute based on the two position measurements that are made.

Appendix: Matrix inversion lemma

- Given that $\bar{A}$, $\bar{C}$, and $(\bar{A} + \bar{B}\bar{C}\bar{D})$ are invertible, then

$$\left(\bar{A} + \bar{B}\bar{C}\bar{D}\right)^{-1} = \bar{A}^{-1} - \bar{A}^{-1}\bar{B}\left(\bar{D}\bar{A}^{-1}\bar{B} + \bar{C}^{-1}\right)^{-1}\bar{D}\bar{A}^{-1}.$$

- Proof: Taking the product of the matrix and its (supposed) inverse, we have,

$$\begin{aligned}
&\left(\bar{A} + \bar{B}\bar{C}\bar{D}\right)^{-1}\left\{\bar{A}^{-1} - \bar{A}^{-1}\bar{B}\left(\bar{D}\bar{A}^{-1}\bar{B} + \bar{C}^{-1}\right)^{-1}\bar{D}\bar{A}^{-1}\right\} \\
&= I + \bar{B}\bar{C}\bar{D}\bar{A}^{-1} - \bar{B}\left(\bar{D}\bar{A}^{-1}\bar{B} + \bar{C}^{-1}\right)^{-1}\bar{D}\bar{A}^{-1} \\
&\quad -\bar{B}\bar{C}\bar{D}\bar{A}^{-1}\bar{B}\left(\bar{D}\bar{A}^{-1}\bar{B} + \bar{C}^{-1}\right)^{-1}\bar{D}\bar{A}^{-1} \\
&= I + \bar{B}\bar{C}\bar{D}\bar{A}^{-1} \\
&\quad -\bar{B}\left(I + \bar{C}\bar{D}\bar{A}^{-1}\bar{B}\right)\left(\bar{D}\bar{A}^{-1}\bar{B} + \bar{C}^{-1}\right)^{-1}\bar{D}\bar{A}^{-1} \\
&= I + \bar{B}\bar{C}\bar{D}\bar{A}^{-1} \\
&\quad -\bar{B}\bar{C}\left(\bar{C}^{-1} + \bar{D}\bar{A}^{-1}\bar{B}\right)\left(\bar{D}\bar{A}^{-1}\bar{B} + \bar{C}^{-1}\right)^{-1}\bar{D}\bar{A}^{-1} \\
&= I.
\end{aligned}$$

Appendix: Plett notation versus textbook notation

- For predicted and estimated state, I use $\hat{x}_k^-$ and $\hat{x}_k^+$, whereas Bar-Shalom uses $\hat{x}(k \mid k-1)$ and $\hat{x}(k \mid k)$.
- For predicted and estimated state covariance, I use $\Sigma_{\tilde{x},k}^-$ and $\Sigma_{\tilde{x},k}^+$, whereas Simon uses P_k^- and P_k^+ and Bar-Shalom uses $P(k \mid k-1)$ and $P(k \mid k)$.
- For measurement prediction covariance, I use $\Sigma_{\tilde{z},k}$, whereas Simon uses S_k and Bar-Shalom uses $S(k)$.