Focus of this week

- In the final week of the first course (week 1.4), you learned how to implement a linear Kalman filter in Octave code.
- Some examples illustrated scenarios where the filter produced good state estimates and other examples illustrated cases where it failed.
- Last week, you learned how to derive the linear Kalman filter as preparation to understand why the Kalman filter might fail in certain circumstances.
- This week, we will leverage this new understanding in order to overcome the vulnerabilities of the Kalman filter to make it "bulletproof."
- Over the course of the week, you will learn how to:
  - Increase numeric robustness,
  - Improve estimates when our system model is not correct,
  - Modify the filter equations when the noises are not zero-mean and white,
  - And others.

Toward improving numeric robustness

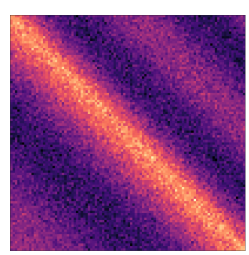
- Within KF, covariance matrices $\Sigma^-_{\tilde{x},k}$ and $\Sigma^+_{\tilde{x},k}$ must remain:
  1. Symmetric, and
  2. Positive definite (all eigenvalues strictly positive)

  at every time step.

- It is possible for both conditions to be violated due to numeric floating-point round-off errors in a computer implementation.
  - This will cause the filter's confidence bounds to be incorrect and can cause the Kalman filter to become unstable (estimates diverging from the truth).
- We wish to find ways to limit or eliminate these problems.

Searching for cause of loss of symmetry

- The cause of covariance matrices becoming asymmetric or nonpositive definite must be due either to the prediction-error time-update or estimation-error measurement-update equations of the filter (steps 1b or 2c).



- Consider first the prediction-error time-update equation:

$$\Sigma^-_{\tilde{x},k} = A\Sigma^+_{\tilde{x},k-1}A^T + \Sigma_{\widetilde{w}}.$$

  - Because we are adding two positive-definite quantities together, the result must be positive definite.
  - A "suitable implementation" of the products of the matrices will avoid loss of symmetry in the final result.

Dealing with loss of symmetry

- Now, consider estimation-error measurement-update equation:

$$\Sigma^+_{\tilde{x},k} = \Sigma^-_{\tilde{x},k} - L_k C_k \Sigma^-_{\tilde{x},k}.$$

- Theoretically, the result is positive definite.
  - But notice the subtraction of one positive-definite quantity from another positive-definite quantity.
  - If there are numeric floating-point round-off errors in an implementation, this subtraction can result in a nonpositive-definite solution.
- It is better to use the Joseph-form covariance update:

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

- Because the subtraction occurs in the "squared" term, the Joseph form guarantees a positive-definite result (two positive-definite terms are added, not subtracted).

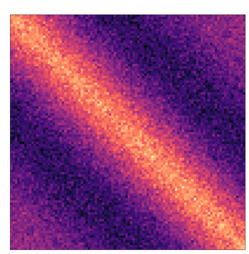
Proof of Joseph-form update

- This may be proven correct via:

$$\begin{aligned}
\Sigma^+_{\tilde{x},k} &= [I - L_k C_k]\, \Sigma^-_{\tilde{x},k}\, [I - L_k C_k]^T + L_k \Sigma_{\tilde{v}} L_k^T \\
&= \Sigma^-_{\tilde{x},k} - L_k C_k \Sigma^-_{\tilde{x},k} - \Sigma^-_{\tilde{x},k} C_k^T L_k^T + L_k C_k \Sigma^-_{\tilde{x},k} C_k^T L_k^T + L_k \Sigma_{\tilde{v}} L_k^T \\
&= \Sigma^-_{\tilde{x},k} - L_k C_k \Sigma^-_{\tilde{x},k} - \Sigma^-_{\tilde{x},k} C_k^T L_k^T + L_k \underbrace{\left(C_k \Sigma^-_{\tilde{x},k} C_k^T + \Sigma_{\tilde{v}}\right)}_{\Sigma_{\tilde{z},k}} L_k^T \\
&= \Sigma^-_{\tilde{x},k} - L_k C_k \Sigma^-_{\tilde{x},k} - \Sigma^-_{\tilde{x},k} C_k^T L_k^T + L_k \Sigma_{\tilde{z},k} L_k^T \\
&= \Sigma^-_{\tilde{x},k} - L_k C_k \Sigma^-_{\tilde{x},k} - \Sigma^-_{\tilde{x},k} C_k^T L_k^T + \underbrace{\left(\Sigma^-_{\tilde{x},k} C_k^T \Sigma^{-1}_{\tilde{z},k}\right)}_{L_k} \Sigma_{\tilde{z},k} L_k^T \\
&= \Sigma^-_{\tilde{x},k} - L_k C_k \Sigma^-_{\tilde{x},k} - \Sigma^-_{\tilde{x},k} C_k^T L_k^T + \Sigma^-_{\tilde{x},k} C_k^T L_k^T \\
&= \Sigma^-_{\tilde{x},k} - L_k C_k \Sigma^-_{\tilde{x},k}.
\end{aligned}$$

Enforcing nonnegative eigenvalues

- If we still compute nonpositive-definite matrix (numerics), can find "nearest" symmetric positive semidefinite matrix.[1]
- Omitting the details, the procedure is:
  - Calculate the singular-value decomposition of the covariance matrix: $\Sigma = USV^T$.
  - Compute $H = VSV^T$.
  - Replace Σ with $(\Sigma + \Sigma^T + H + H^T)/4$.
- In Octave code we modify previously computed $\Sigma^+_{\tilde{x},k}$ via:



```
% First compute SigmaX via step 2c, then:
[~,S,V] = svd(SigmaX);
H = V*S*V';
SigmaX = (SigmaX + SigmaX' + H + H')/4;
```

[1]Nicholas J. Higham, "Computing a Nearest Symmetric Positive Semidefinite Matrix," *Linear Algebra and its Applications*, 103, 103–118, 1988

Summary

- If covariance matrices become nonpositive-definite, KF will diverge and “confidence bounds” will become meaningless.
- Theoretically “impossible” but often happens in practice due to numeric “round-off” errors in floating-point operations.
- Can minimize likelihood of problem using Joseph-form covariance update.
- Can further improve using Higham’s method, guaranteeing at least positive semi-definite matrices.

Credits

Credits for photos in this lesson

- Mall on slide 1: Pixabay license (`https://pixabay.com/en/service/license/`), `https://pixabay.com/en/mall-black-and-white-escalators-1331514/`