For continuous-time:

\[ x_0, x_1, x_2, x_3, \ldots \]

or simply:

\[ x_0, x(t), x_{at}, x_{at}, \ldots \]

Today: Discrete-time

Define a random variable at time \( t \):

Random Process: \( X(t) \)

Reading: 3.5 and 3.6

Innovation Sequence and Discrete-time Kalman Filter.
Estimate scalar \( \bar{X} \) with \( \hat{Y} \):

\[
\hat{Y} = \begin{bmatrix} \bar{X} \\ \vdots \\ \bar{X} \end{bmatrix}
\]

\[
\text{cov}(\hat{Y}) = \frac{1}{N}
\]

\[
\text{cov}(\bar{X}|\hat{Y}) = \frac{1}{K} \text{diag}(\hat{Y})
\]

\[
\text{cov}(\bar{X}|\hat{Y}) = \frac{1}{K} \hat{Y}^T \hat{Y}
\]

Decorrelate

\[
\hat{Y} = \begin{bmatrix} \bar{X} \\ \vdots \\ \bar{X} \end{bmatrix}
\]

\[
\text{cov}(\hat{Y}) = \frac{1}{K} \hat{Y}^T \hat{Y}
\]

Linear Innovation Sequence

And if \( N \) is growing

\[
\text{cov}(\bar{X}|\hat{Y}) \rightarrow \text{diag}(\hat{Y})
\]

The complexity will become \( O(N^2) \).
"Online" algorithm - Gram-Schmidt orthogonalization.

And it does not solve the ground N problem.

But computing $K_Z$ is even harder than matrix inversion: $O(N^3)$. This should be obvious, right?

They are orthogonal.

\[
\sum_{i=1}^{n} E(\mathcal{X}_i) + \sum_{i=1}^{n} E(\mathcal{X}_i - EX_i) = 0. \]

If we can correlate...
Linear Stochastic Modeling = Linear System + Noise.

The incorporation of such domain knowledge allows us to keep track of the statistics of the process without looking up the memory.

...produced (as part of a `noisy` dynamic system).

In many occasions, we have knowledge about how the random process is - eventually it will blow up.

The problem with orthogonization is that the memory still grows linearly.
In stochastic linear system modeling

\[ x_t = C x_t + D u_t \]

\[ x_{t+1} = A x_t + B u_t \]

\( u \) = uncorrelated noise (state and measurement),

\( x \) = unobserved state that describes the underlying process,

\( \hat{x} \) = observed measurable sequence.

Remember the state-space approach.
If \( \mathbf{y}_1 \), \( \mathbf{y}_2 \), ..., \( \mathbf{y}_n \) are system realizations:

- System identification
- \( f_1, f_2, \ldots, f_n \)

- Prediction

Typical usage:

\[
\mathbf{v}_k = \mathbf{H}_k \mathbf{x}_k + \mathbf{w}_k
\]
\[
\mathbf{x}_k = \mathbf{F}_k \mathbf{x}_{k-1} + \mathbf{v}_k
\]

Specifically,

\[
\mathbf{v}_k = \mathbf{H}_k \mathbf{x}_k + \mathbf{w}_k
\]
\[
\mathbf{x}_k = \mathbf{F}_k \mathbf{x}_{k-1} + \mathbf{v}_k
\]
The partition term is due to the discretization process. \( \Lambda \) is non-zero in acceleration matrices.

\[
\begin{pmatrix}
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0
\end{pmatrix}
\]

\( \text{State} \ X(t) \)

\( \text{Car velocity} \)

\( \text{Car position} \)

\( \text{Ground} \)

\( \text{Light Ray} \)

\( \text{Particle Path} \)

\( \text{C} \)

\( \text{x} \)

\( \text{t} \)

\( \text{Example: Camera Tracking} \)

\( \text{Constant Velocity Model (Zero Acceleration Model)} \)

\( \text{\( X(t) = X(t) + W(t) \)} \)

\( \text{\( P(t) = P(t) + \frac{1}{2} F(t) \)} \)

\( \text{\( F(t) \)} \)
Camera measurement (assume camera plane parallel to ground)

\[ \mathbf{X}_t = \begin{pmatrix} M_t \\ N_t \end{pmatrix} = \begin{pmatrix} \alpha E_t \\ \alpha P_t \end{pmatrix} = \begin{pmatrix} \alpha E_t \\ 0 \times 0 \times 0 \end{pmatrix} \times \mathbf{X}_t \]
Observation of the Kalman filter is a time delay of the fixed amount of time that can track the mean and variance of the state. Formally, we assume $X_0 \sim N(\mu_0, P_0)$.

If $X_t$, $W_t$, and $V_t$ are jointly Gaussian, the filter is jointly Gaussian. Therefore, the Kalman filter $(\hat{\mathbf{x}}, \tilde{\mathbf{V}})$ is Gaussian.

Focus on the prediction problem: find $\mathbf{x}_{t+1}$, $\tilde{\mathbf{V}}_{t+1}$.

1. $\mathbf{F} = H^T X_t + W_t$
2. $\tilde{\mathbf{V}}_t = \frac{1}{\mathbf{I}} \mathbf{x}_t + \mathbf{W}_t$

Back to the ground model.
Key (from last lecture): \( \mathbf{X} \sim \mathcal{N}(\mathbf{E}(\mathbf{X}), \text{cov}(\mathbf{X}, \mathbf{X}^T) \text{cov}^{-1}(\mathbf{X}, \mathbf{Y}) \text{cov}(\mathbf{Y})) \)

Thus, KF recursively computes \( \mathbf{X} \mid \mathbf{y}_0 = \mathbf{E}(\mathbf{X} \mid \mathbf{y}_0, \mathbf{y}_1, \ldots, \mathbf{y}_n) \) and \( \mathbf{Z} \mid \mathbf{y}_0 = \text{cov}(\mathbf{X} \mid \mathbf{y}_0) \)

Define \( R_{\mathbf{y}_0} = \mathbf{E}(\mathbf{X} \mid \mathbf{y}_0, \mathbf{y}_1) \)

\( T = \) time update step

\( P = \) prediction

\( \sigma = \) information update step

\( \ddot{x}_0 \) kalman filter
\[ X_{\theta} = \sqrt{\frac{P_{\theta} - P_{\theta} - P_{\theta}}{P_{\theta} + P_{\theta} + P_{\theta}}} \]

\[ R + H_{\theta}X_{\theta} = \begin{cases} 0 \text{ (if } H_{\theta}X_{\theta} = 0) \end{cases} \]

\[ \text{Suppose } (X_{\theta}, Y_{\theta}) = (0, 0) \]

\[ H_{\theta}X_{\theta} = \begin{cases} 0 \text{ (if } H_{\theta}X_{\theta} = 0) \end{cases} \]

\[ E(\theta) = (X_{\theta}, Y_{\theta}) \]

\[ \text{First Step: Update Information} \]

\[ X_{\theta} \leftarrow X_{\theta} \]
\( \mathbf{X}_{1|0} = \mathbf{P} \mathbf{x}_{1|0} \)

\( \mathbf{P} = \mathbf{P}_{10} + \mathbf{F} \mathbf{Q}_{10} \mathbf{F}^T \)

\( \mathbf{Q}_{1|0} = \mathbf{P} \mathbf{x}_{1|0} \mathbf{C}_{\mathbf{x}_{10} \mathbf{y}_{10}} \mathbf{C}_{\mathbf{x}_{10} \mathbf{y}_{10}}^T \mathbf{P}_{10} + \mathbf{R} \)

\( \mathbf{C}_{\mathbf{x}_{10} \mathbf{y}_{10}} = \mathbf{C}_{\mathbf{x}_{10} \mathbf{y}_{10}}^T \)

**Step 2**

**Time update**

\( \mathbf{x}_{1|0} = \mathbf{x}_{1|0} \)

\( \mathbf{x}_{1|0} = \mathbf{x}_{1|0} \)

To improve the balanced algorithm, \( \mathbf{R} \mathbf{H}^{-1} \mathbf{R}^T + \mathbf{P} \) is called Kalman gain – weighting factor of the new observation.
\[ R + H^{\perp \perp} \mathcal{R} = H \] 
\[ = H^{\perp \perp} \mathcal{R} + H \] 
\[ = H^{\perp \perp} \mathcal{R} + \text{cov}(V) + \text{cov}(V) \] 
\[ = \text{cov}(H^{\perp \perp} \mathcal{R} + V) \] 
\[ = \text{cov}(V) \] 
\[ \Rightarrow E(V \mid X) = E(V \mid X) \] 

Reduction Step 3
In summary, information update:

\[ X^{t+1} = X^t + Z^{t+1} (H^{t+1} X^{t+1} + R) \]

And

\[ Z_{t+1} = Z_t + (H_t Z_t + R) (X^t - H^t X^t) \]

Thus, Step 1:

\[ X_t = X_0 \]

\[ Z_t = Z_0 \]
\[ \text{Particle Filter} \quad \leftrightarrow \quad \text{Nonlinear + Non-Gaussian Noise} \]

\[ \text{Extended Kalman Filter} \quad \leftrightarrow \quad \text{Nonlinear + Gaussian Noise} \quad \leftrightarrow \quad \text{Unscented Kalman Filter} \]

System identification is forward recursion \( F_k \), \( x_k \), \( y_k \), \( \ldots \), \( x_0 \) and backward recursion (backward).