The Kalman Filter

ELG 5218 - Uncertainty Evaluation in Engineering Measurements and Machine Learning

Miodrag Bolic

University of Ottawa

February 4, 2026

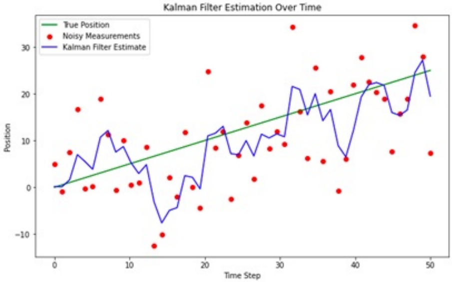
Outline

Notebook: ssm_examples.ipynb

The Filtering Problem

Scenario: A robot moving in 1D space

- **State:** position z_t (hidden, uncertain)
- **Observations:** noisy GPS measurements y_t
- **Goal:** Estimate current position from all measurements so far

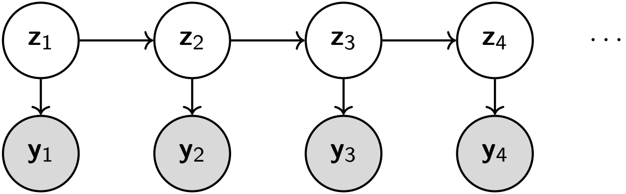


Challenge: How to optimally fuse noisy measurements with predictions from a motion model?

What is Filtering?

General Problem:

- Hidden state sequence: $\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_t$
- Observations: $\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_t$
- Compute posterior: $p(\mathbf{z}_t \mid \mathbf{y}_{1:t})$



Types of Inference:

- **Filtering:** $p(\mathbf{z}_t \mid \mathbf{y}_{1:t})$ — estimate current state
- **Smoothing:** $p(\mathbf{z}_t \mid \mathbf{y}_{1:T})$ — estimate past state (more data)
- **Prediction:** $p(\mathbf{z}_{t+k} \mid \mathbf{y}_{1:t})$ — forecast future

Why Kalman Filter?

The Special Case

Linear dynamics + Gaussian noise ⇒ Exact, closed-form solution!

Advantages:

1. **Optimal:** Minimum mean squared error (MMSE) estimator
2. **Recursive:** Online processing, constant memory $O(n_z^2)$
3. **Efficient:** No integrals, just matrix operations
4. **Interpretable:** Clear predict-update structure

Historical Impact:

- Apollo lunar missions (1960s)
- GPS navigation
- Autonomous vehicles
- Economics, signal processing, control systems

Probabilistic State Space Model

Transition Model (Dynamics)

$$\mathbf{z}_t = \mathbf{F}_t \mathbf{z}_{t-1} + \mathbf{B}_t \mathbf{u}_t + \mathbf{b}_t + \mathbf{q}_t, \quad \mathbf{q}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_t) \tag{1}$$

Observation Model (Measurement)

$$\mathbf{y}_t = \mathbf{H}_t \mathbf{z}_t + \mathbf{D}_t \mathbf{u}_t + \mathbf{d}_t + \mathbf{r}_t, \quad \mathbf{r}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_t) \tag{2}$$

Components:

- $\mathbf{z}_t \in \mathbb{R}^{n_z}$: hidden state (position, velocity, ...)
- $\mathbf{y}_t \in \mathbb{R}^{n_y}$: observation (sensor measurement)
- $\mathbf{u}_t \in \mathbb{R}^{n_u}$: control input (known)
- $\mathbf{F}_t$: state transition matrix, $\mathbf{H}_t$: observation matrix
- $\mathbf{Q}_t$: process noise covariance, $\mathbf{R}_t$: measurement noise covariance

Bayesian Filter: Recursive Structure

Goal: Compute $p(\mathbf{z}_t \mid \mathbf{y}_{1:t})$ recursively

Initialization

Start with prior: $p(\mathbf{z}_0) = \mathcal{N}(\mathbf{z}_0 \mid \boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0)$

Prediction Step (Chapman-Kolmogorov)

Marginalize over previous state:

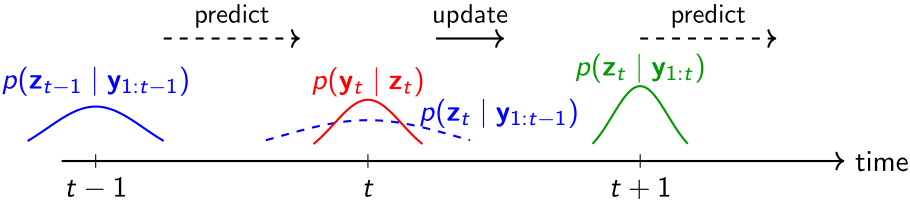
$$p(\mathbf{z}_t \mid \mathbf{y}_{1:t-1}) = \int p(\mathbf{z}_t \mid \mathbf{z}_{t-1}) p(\mathbf{z}_{t-1} \mid \mathbf{y}_{1:t-1}) d\mathbf{z}_{t-1} \tag{3}$$

Update Step (Bayes' Rule)

Incorporate new measurement:

$$p(\mathbf{z}_t \mid \mathbf{y}_{1:t}) = \frac{p(\mathbf{y}_t \mid \mathbf{z}_t) p(\mathbf{z}_t \mid \mathbf{y}_{1:t-1})}{p(\mathbf{y}_t \mid \mathbf{y}_{1:t-1})} \tag{4}$$

Bayesian Filter: Graphical View



Key Insight:

- **Prediction** propagates uncertainty (wider distribution)
- **Update** incorporates evidence (narrower distribution)

Bayesian Filter Derivation (1/3): Model Assumptions

Goal: Compute $p(\mathbf{z}_t \mid \mathbf{y}_{1:t})$ recursively.

Starting point: joint probability factorization

For a state-space model (SSM), the joint distribution of states and observations up to time t factorizes as:

$$p(\mathbf{z}_{0:t}, \mathbf{y}_{1:t}) = p(\mathbf{z}_0) \prod_{k=1}^{t} p(\mathbf{z}_k \mid \mathbf{z}_{k-1})\, p(\mathbf{y}_k \mid \mathbf{z}_k).$$

Key assumptions

- **Markov property (state dynamics):**

$$p(\mathbf{z}_k \mid \mathbf{z}_{0:k-1}) = p(\mathbf{z}_k \mid \mathbf{z}_{k-1}).$$

- **Conditional independence (observations):**

$$p(\mathbf{y}_k \mid \mathbf{z}_{0:k}, \mathbf{y}_{1:k-1}) = p(\mathbf{y}_k \mid \mathbf{z}_k).$$

Bayesian Filter Derivation (2/3): Prediction

Prediction step (Chapman–Kolmogorov)

Start from the posterior at time $t-1$: $p(\mathbf{z}_{t-1} \mid \mathbf{y}_{1:t-1})$. To obtain the *prior* (predictive) at time t:

$$\begin{aligned}
p(\mathbf{z}_t \mid \mathbf{y}_{1:t-1}) &= \int p(\mathbf{z}_t, \mathbf{z}_{t-1} \mid \mathbf{y}_{1:t-1})\, d\mathbf{z}_{t-1} \\
&= \int p(\mathbf{z}_t \mid \mathbf{z}_{t-1}, \mathbf{y}_{1:t-1})\, p(\mathbf{z}_{t-1} \mid \mathbf{y}_{1:t-1})\, d\mathbf{z}_{t-1} \\
&\overset{\text{Markov}}{=} \int p(\mathbf{z}_t \mid \mathbf{z}_{t-1})\, p(\mathbf{z}_{t-1} \mid \mathbf{y}_{1:t-1})\, d\mathbf{z}_{t-1}.
\end{aligned}$$

Bayesian Filter Derivation (3/3): Update

Update step (Bayes' rule)

Start with Bayes' theorem (posterior at time t): $p(\mathbf{z}_t \mid \mathbf{y}_{1:t}) = \frac{p(\mathbf{y}_{1:t}|\mathbf{z}_t)\, p(\mathbf{z}_t)}{p(\mathbf{y}_{1:t})}$.

Decompose $\mathbf{y}_{1:t} = \{\mathbf{y}_{1:t-1}, \mathbf{y}_t\}$: $p(\mathbf{z}_t \mid \mathbf{y}_{1:t}) = \frac{p(\mathbf{y}_t, \mathbf{y}_{1:t-1}|\mathbf{z}_t)\, p(\mathbf{z}_t)}{p(\mathbf{y}_t, \mathbf{y}_{1:t-1})}$.

Use conditional probability: $p(\mathbf{y}_t, \mathbf{y}_{1:t-1} \mid \mathbf{z}_t) = p(\mathbf{y}_t \mid \mathbf{y}_{1:t-1}, \mathbf{z}_t)\, p(\mathbf{y}_{1:t-1} \mid \mathbf{z}_t)$.

Conditional independence assumption: $p(\mathbf{y}_t \mid \mathbf{y}_{1:t-1}, \mathbf{z}_t) = p(\mathbf{y}_t \mid \mathbf{z}_t)$.

Substitute into Bayes: $p(\mathbf{z}_t \mid \mathbf{y}_{1:t}) = \frac{p(\mathbf{y}_t|\mathbf{z}_t)\; p(\mathbf{y}_{1:t-1}|\mathbf{z}_t)\; p(\mathbf{z}_t)}{p(\mathbf{y}_t, \mathbf{y}_{1:t-1})}$.

Recognize two identities: $\frac{p(\mathbf{y}_{1:t-1}|\mathbf{z}_t)\, p(\mathbf{z}_t)}{p(\mathbf{y}_{1:t-1})} = p(\mathbf{z}_t \mid \mathbf{y}_{1:t-1})$, $\qquad \frac{p(\mathbf{y}_{1:t-1})}{p(\mathbf{y}_t, \mathbf{y}_{1:t-1})} = \frac{1}{p(\mathbf{y}_t|\mathbf{y}_{1:t-1})}$.

Therefore (Bayes update): $p(\mathbf{z}_t \mid \mathbf{y}_{1:t}) = \frac{p(\mathbf{y}_t|\mathbf{z}_t)\; p(\mathbf{z}_t|\mathbf{y}_{1:t-1})}{p(\mathbf{y}_t|\mathbf{y}_{1:t-1})}$.

- **Summary:** Predict (propagate uncertainty) $\rightarrow$ Update (incorporate measurement).
- **Kalman filter case:** for linear-Gaussian models, both integrals are closed form.

Kalman Filter: The Gaussian Case

Key Property: Linear transformations of Gaussians are Gaussian!

If $\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ and $\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{b} + \mathbf{w}$ where $\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \mathbf{W})$, then:

$$\mathbf{y} \sim \mathcal{N}(\mathbf{A}\boldsymbol{\mu} + \mathbf{b}, \mathbf{A}\boldsymbol{\Sigma}\mathbf{A}^T + \mathbf{W}) \tag{5}$$

Consequence for filtering:

- If $p(\mathbf{z}_{t-1} \mid \mathbf{y}_{1:t-1}) = \mathcal{N}(\boldsymbol{\mu}_{t-1|t-1}, \boldsymbol{\Sigma}_{t-1|t-1})$
- Then $p(\mathbf{z}_t \mid \mathbf{y}_{1:t-1})$ is also Gaussian!
- And $p(\mathbf{z}_t \mid \mathbf{y}_{1:t})$ is also Gaussian!

Kalman Filter = Recursive Gaussian Propagation

We only need to track means $\boldsymbol{\mu}_{t|t}$ and covariances $\boldsymbol{\Sigma}_{t|t}$ at each step!

Notation: $t \mid t$ vs. $t \mid t-1$ (Prediction & Update)

How to read the notation

"$t \mid s$" $\implies$ time t given data up to time s.

State estimates (mean & covariance)

$\boldsymbol{\mu}_{t|t-1} = \mathbb{E}[\mathbf{z}_t \mid \mathbf{y}_{1:t-1}]$, $\quad \boldsymbol{\Sigma}_{t|t-1} = \mathrm{Cov}(\mathbf{z}_t \mid \mathbf{y}_{1:t-1})$

$\boldsymbol{\mu}_{t|t} = \mathbb{E}[\mathbf{z}_t \mid \mathbf{y}_{1:t}]$, $\quad \boldsymbol{\Sigma}_{t|t} = \mathrm{Cov}(\mathbf{z}_t \mid \mathbf{y}_{1:t})$

Predicted measurement

For linear observations $\mathbf{y}_t = \mathbf{H}_t \mathbf{z}_t + \mathbf{v}_t$:

$$\hat{\mathbf{y}}_t = \mathbb{E}[\mathbf{y}_t \mid \mathbf{y}_{1:t-1}] = \mathbf{H}_t\, \boldsymbol{\mu}_{t|t-1}$$

(Nonlinear: $\hat{\mathbf{y}}_t = \mathbb{E}[h(\mathbf{z}_t) \mid \mathbf{y}_{1:t-1}] \approx h(\boldsymbol{\mu}_{t|t-1})$.)

Mini example (1D)

Random walk + noisy measurement:
$z_t = z_{t-1} + q_t, \quad q_t \sim \mathcal{N}(0, 0.2^2)$,
$y_t = z_t + v_t, \quad v_t \sim \mathcal{N}(0, 0.5^2)$.
Assume at $t-1$: $\mu_{t-1|t-1} = 2.0$, $\quad \Sigma_{t-1|t-1} = 0.3^2 = 0.09$

Predict: $\mu_{t|t-1} = 2.0$, $\quad \Sigma_{t|t-1} = 0.09 + 0.04 = 0.13$, $\hat{y}_t = \mu_{t|t-1} = 2.0$

If measurement $y_t = 2.6$ arrives, innovation $= y_t - \hat{y}_t = 0.6$.

Update (Kalman-style):
$S = \Sigma_{t|t-1} + 0.5^2 = 0.13 + 0.25 = 0.38$
$K = \frac{\Sigma_{t|t-1}}{S} = \frac{0.13}{0.38} = 0.342$
$\mu_{t|t} = 2.0 + 0.342(0.6) = 2.205$,
$\Sigma_{t|t} = (1 - K)\, 0.13 = 0.0856$

Prediction Step

Goal: derive $p(\mathbf{z}_t \mid \mathbf{y}_{1:t-1})$.

Identify Theorem 3 (Affine-Gaussian forward model):
Let $\mathbf{x} = \mathbf{z}_{t-1}$ and $\mathbf{y} = \mathbf{z}_t$. Assume

$$p(\mathbf{x}) = \mathcal{N}(\boldsymbol{\mu}_{t-1|t-1}, \boldsymbol{\Sigma}_{t-1|t-1}), \qquad p(\mathbf{y} \mid \mathbf{x}) = \mathcal{N}(\mathbf{F}_t\mathbf{x} + \mathbf{B}_t\mathbf{u}_t + \mathbf{b}_t, \mathbf{Q}_t).$$

This matches the dynamics $\mathbf{z}_t = \mathbf{F}_t\mathbf{z}_{t-1} + \mathbf{B}_t\mathbf{u}_t + \mathbf{b}_t + \mathbf{q}_t$.

Apply Corollary 2 (marginal / predictive distribution):

$$p(\mathbf{z}_t \mid \mathbf{y}_{1:t-1}) = \int p(\mathbf{z}_t \mid \mathbf{z}_{t-1})\, p(\mathbf{z}_{t-1} \mid \mathbf{y}_{1:t-1})\, d\mathbf{z}_{t-1} = \mathcal{N}(\boldsymbol{\mu}_{t|t-1}, \boldsymbol{\Sigma}_{t|t-1})$$

with

$$\boldsymbol{\mu}_{t|t-1} = \mathbf{F}_t\boldsymbol{\mu}_{t-1|t-1} + \mathbf{B}_t\mathbf{u}_t + \mathbf{b}_t, \qquad \boldsymbol{\Sigma}_{t|t-1} = \mathbf{Q}_t + \mathbf{F}_t\boldsymbol{\Sigma}_{t-1|t-1}\mathbf{F}_t^\top.$$

Kalman Filter: Prediction (Time Update)

Goal: derive $p(\mathbf{z}_t \mid \mathbf{y}_{1:t-1})$.

Mapping: Gaussian formulas

Use Corollary 2: $p(\mathbf{y}) = \int p(\mathbf{y} \mid \mathbf{x}) p(\mathbf{x})\, d\mathbf{x}$.

Theorem symbol	Kalman symbol
$\mathbf{x}$	$\mathbf{z}_{t-1}$
$\mathbf{y}$	$\mathbf{z}_t$
A	$\mathbf{F}_t$
b	$\mathbf{B}_t\mathbf{u}_t + \mathbf{b}_t$
$\Sigma_{\mathbf{y}\mid\mathbf{x}}$	$\mathbf{Q}_t$
$\mu_{\mathbf{x}}$	$\boldsymbol{\mu}_{t-1\mid t-1}$
$\Sigma_{\mathbf{x}}$	$\boldsymbol{\Sigma}_{t-1\mid t-1}$

Intuition: Affine transformation shifts the mean by $\mathbf{F}_t(\cdot) + (\mathbf{B}_t\mathbf{u}_t + \mathbf{b}_t)$ and adds uncertainty: *propagated covariance* $+\ \mathbf{Q}_t$.

Theorem 3 + Corollary 2

If $p(\mathbf{x}) = \mathcal{N}(\mathbf{x}; \mu_{\mathbf{x}}, \Sigma_{\mathbf{x}})$ and

$p(\mathbf{y} \mid \mathbf{x}) = \mathcal{N}(\mathbf{y}; A\mathbf{x} + b, \Sigma_{\mathbf{y}\mid\mathbf{x}})$,

then the marginal (predictive) distribution is

$$p(\mathbf{y}) = \mathcal{N}\big(\mathbf{y};\ A\mu_{\mathbf{x}} + b,\ \Sigma_{\mathbf{y}\mid\mathbf{x}} + A\Sigma_{\mathbf{x}}A^\top\big).$$

Apply mapping $\Rightarrow$ Kalman prediction

$$p(\mathbf{z}_t \mid \mathbf{y}_{1:t-1}) = \mathcal{N}\big(\mathbf{z}_t; \boldsymbol{\mu}_{t\mid t-1}, \boldsymbol{\Sigma}_{t\mid t-1}\big)$$

$$\boldsymbol{\mu}_{t\mid t-1} = \mathbf{F}_t\boldsymbol{\mu}_{t-1\mid t-1} + \mathbf{B}_t\mathbf{u}_t + \mathbf{b}_t$$

$$\boldsymbol{\Sigma}_{t\mid t-1} = \mathbf{Q}_t + \mathbf{F}_t\boldsymbol{\Sigma}_{t-1\mid t-1}\mathbf{F}_t^\top$$

Kalman Filter: Update (1/2) – Build a Joint Gaussian

Given (from prediction): $p(\mathbf{z}_t \mid \mathbf{y}_{1:t-1}) = \mathcal{N}\big(\mathbf{z}_t; \boldsymbol{\mu}_{t|t-1}, \boldsymbol{\Sigma}_{t|t-1}\big)$.

Mapping to Theorem 3

Theorem symbol	Kalman symbol
$\mathbf{x}$	$\mathbf{z}_t$
$\mathbf{y}$	$\mathbf{y}_t$
A	$\mathbf{H}_t$
b	$\mathbf{D}_t\mathbf{u}_t + \mathbf{d}_t$
$\Sigma_{\mathbf{y}\mid\mathbf{x}}$	$\mathbf{R}_t$
$\mu_{\mathbf{x}}$	$\boldsymbol{\mu}_{t\mid t-1}$
$\Sigma_{\mathbf{x}}$	$\boldsymbol{\Sigma}_{t\mid t-1}$

Observation model (Affine-Gaussian)

$$p(\mathbf{y}_t \mid \mathbf{z}_t) = \mathcal{N}\big(\mathbf{y}_t; \mathbf{H}_t\mathbf{z}_t + \mathbf{D}_t\mathbf{u}_t + \mathbf{d}_t, \mathbf{R}_t\big).$$

Why we do this Once we have the **joint Gaussian** of $(\mathbf{z}_t, \mathbf{y}_t)$, Theorem 2 gives the **conditional** $p(\mathbf{z}_t \mid \mathbf{y}_t, \mathbf{y}_{1:t-1})$.

Theorem 3 (Affine-Gaussian

If $p(\mathbf{x}) = \mathcal{N}(\mu_{\mathbf{x}}, \Sigma_{\mathbf{x}})$ and $p(\mathbf{y} \mid \mathbf{x}) = \mathcal{N}(A\mathbf{x} + b, \Sigma_{\mathbf{y}\mid\mathbf{x}})$, then $\begin{bmatrix}\mathbf{x}\\ \mathbf{y}\end{bmatrix}$ is jointly Gaussian with block covariance structure.

Apply mapping $\Rightarrow$ joint of $(\mathbf{z}_t, \mathbf{y}_t) \mid \mathbf{y}_{1:t-1}$

$$\begin{bmatrix}\mathbf{z}_t\\ \mathbf{y}_t\end{bmatrix}\Bigg|\mathbf{y}_{1:t-1} \sim \mathcal{N}\left(\begin{bmatrix}\boldsymbol{\mu}_{t|t-1}\\ \hat{\mathbf{y}}_t\end{bmatrix}, \begin{bmatrix}\boldsymbol{\Sigma}_{t|t-1} & \boldsymbol{\Sigma}_{t|t-1}\mathbf{H}_t^\top\\ \mathbf{H}_t\boldsymbol{\Sigma}_{t|t-1} & \mathbf{S}_t\end{bmatrix}\right)$$

$$\hat{\mathbf{y}}_t = [\mathbf{y}_t \mid \mathbf{y}_{1:t-1}] = \mathbf{H}_t\boldsymbol{\mu}_{t|t-1} + \mathbf{D}_t\mathbf{u}_t + \mathbf{d}_t$$

$$\mathbf{S}_t = (\mathbf{y}_t \mid \mathbf{y}_{1:t-1}) = \mathbf{H}_t\boldsymbol{\Sigma}_{t|t-1}\mathbf{H}_t^\top + \mathbf{R}_t$$

Kalman Filter: Update (2/2) – Condition the Joint

Goal: Compute the filtering posterior: $p(\mathbf{z}_t \mid \mathbf{y}_{1:t}) \;=\; p(\mathbf{z}_t \mid \mathbf{y}_t, \mathbf{y}_{1:t-1})$.

Theorem 2

For a joint Gaussian $\begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix}$,

$$\boldsymbol{\mu}_{\mathbf{x}|\mathbf{y}} = \boldsymbol{\mu}_{\mathbf{x}} + \boldsymbol{\Sigma}_{\mathbf{xy}}\boldsymbol{\Sigma}_{\mathbf{yy}}^{-1}(\mathbf{y} - \boldsymbol{\mu}_{\mathbf{y}}),$$

$$\boldsymbol{\Sigma}_{\mathbf{x}|\mathbf{y}} = \boldsymbol{\Sigma}_{\mathbf{xx}} - \boldsymbol{\Sigma}_{\mathbf{xy}}\boldsymbol{\Sigma}_{\mathbf{yy}}^{-1}\boldsymbol{\Sigma}_{\mathbf{yx}}.$$

Mapping to joint from Update

Theorem 2 symbol	Kalman symbol
$\mathbf{x}$	$\mathbf{z}_t$
$\mathbf{y}$	$\mathbf{y}_t$
$\boldsymbol{\mu}_{\mathbf{x}}$	$\boldsymbol{\mu}_{t\|t-1}$
$\boldsymbol{\mu}_{\mathbf{y}}$	$\hat{\mathbf{y}}_t$
$\boldsymbol{\Sigma}_{\mathbf{xy}}$	$\boldsymbol{\Sigma}_{t\|t-1}\mathbf{H}_t^\top$
$\boldsymbol{\Sigma}_{\mathbf{yy}}$	$\mathbf{S}_t$

Kalman update = Theorem 2

$$\mathbf{K}_t = \boldsymbol{\Sigma}_{t|t-1}\mathbf{H}_t^\top \mathbf{S}_t^{-1}$$

$$\mathbf{S}_t = \mathbf{H}_t\boldsymbol{\Sigma}_{t|t-1}\mathbf{H}_t^\top + \mathbf{R}_t$$

Define the **innovation** (prediction error):

$$\tilde{\mathbf{y}}_t = \mathbf{y}_t - \hat{\mathbf{y}}_t$$

$$(\hat{\mathbf{y}}_t = \mathbf{H}_t\boldsymbol{\mu}_{t|t-1} + \mathbf{D}_t\mathbf{u}_t + \mathbf{d}_t)$$

$$\boldsymbol{\mu}_{t|t} = \boldsymbol{\mu}_{t|t-1} + \mathbf{K}_t\,\tilde{\mathbf{y}}_t$$

$$\boldsymbol{\Sigma}_{t|t} = \boldsymbol{\Sigma}_{t|t-1} - \mathbf{K}_t\,\mathbf{S}_t\,\mathbf{K}_t^\top$$

Kalman Filter: Complete Algorithm

Initialization

$\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0$ (prior mean and covariance)

For $t = 1, 2, \ldots, T$:

Prediction Step

$$\boldsymbol{\mu}_{t|t-1} = \mathbf{F}_t \boldsymbol{\mu}_{t-1|t-1} + \mathbf{B}_t \mathbf{u}_t + \mathbf{b}_t \tag{6}$$

$$\boldsymbol{\Sigma}_{t|t-1} = \mathbf{F}_t \boldsymbol{\Sigma}_{t-1|t-1} \mathbf{F}_t^T + \mathbf{Q}_t \tag{7}$$

Update Step

$$\hat{\mathbf{y}}_t = \mathbf{H}_t \boldsymbol{\mu}_{t|t-1} + \mathbf{D}_t \mathbf{u}_t + \mathbf{d}_t \tag{8}$$

$$\mathbf{S}_t = \mathbf{H}_t \boldsymbol{\Sigma}_{t|t-1} \mathbf{H}_t^T + \mathbf{R}_t \tag{9}$$

$$\mathbf{K}_t = \boldsymbol{\Sigma}_{t|t-1} \mathbf{H}_t^T \mathbf{S}_t^{-1} \tag{10}$$

$$\boldsymbol{\mu}_{t|t} = \boldsymbol{\mu}_{t|t-1} + \mathbf{K}_t (\mathbf{y}_t - \hat{\mathbf{y}}_t) \tag{11}$$

$$\boldsymbol{\Sigma}_{t|t} = (\mathbf{I} - \mathbf{K}_t \mathbf{H}_t) \boldsymbol{\Sigma}_{t|t-1} \tag{12}$$

Kalman Filter Operations (One Time Step)

Model + process noise
$\mathbf{F}_t,\ \mathbf{B}_t\mathbf{u}_t,\ \mathbf{b}_t,\ \mathbf{Q}_t$

Measurement + sensor noise
$\mathbf{y}_t,\ \mathbf{H}_t,\ \mathbf{D}_t\mathbf{u}_t,\ \mathbf{d}_t,\ \mathbf{R}_t$

Previous posterior
$\boldsymbol{\mu}_{t-1|t-1},\ \boldsymbol{\Sigma}_{t-1|t-1}$

Prediction (time update)

$$\boldsymbol{\mu}_{t|t-1} = \mathbf{F}_t\boldsymbol{\mu}_{t-1|t-1} + \mathbf{B}_t\mathbf{u}_t + \mathbf{b}_t$$
$$\boldsymbol{\Sigma}_{t|t-1} = \mathbf{F}_t\boldsymbol{\Sigma}_{t-1|t-1}\mathbf{F}_t^\top + \mathbf{Q}_t$$

Update (measurement update)

$$\hat{\mathbf{y}}_t = \mathbf{H}_t\boldsymbol{\mu}_{t|t-1} + \mathbf{D}_t\mathbf{u}_t + \mathbf{d}_t$$
$$\mathbf{S}_t = \mathbf{H}_t\boldsymbol{\Sigma}_{t|t-1}\mathbf{H}_t^\top + \mathbf{R}_t$$
$$\mathbf{K}_t = \boldsymbol{\Sigma}_{t|t-1}\mathbf{H}_t^\top\mathbf{S}_t^{-1}$$
$$\boldsymbol{\mu}_{t|t} = \boldsymbol{\mu}_{t|t-1} + \mathbf{K}_t(\mathbf{y}_t - \hat{\mathbf{y}}_t)$$
$$\boldsymbol{\Sigma}_{t|t} = (\mathbf{I} - \mathbf{K}_t\mathbf{H}_t)\boldsymbol{\Sigma}_{t|t-1}$$

innovation (residual)
$\mathbf{v}_t = \mathbf{y}_t - \hat{\mathbf{y}}_t$

New posterior
$\boldsymbol{\mu}_{t|t},\ \boldsymbol{\Sigma}_{t|t}$

next time step

Computational Complexity

Per time step:

- **Prediction:**
  - Mean: $O(n_z^2)$ (matrix-vector multiply)
  - Covariance: $O(n_z^3)$ (matrix-matrix multiply)
- **Update:**
  - Innovation covariance $\mathbf{S}_t$: $O(n_z^2 n_y)$
  - Matrix inverse: $O(n_y^3)$ or $O(n_z^3)$ depending on form
  - Kalman gain: $O(n_z^2 n_y)$
  - Mean and covariance update: $O(n_z^2 n_y)$

Total: $O(n_z^3)$ per time step

Memory

Constant: $O(n_z^2)$ — only need to store current $\boldsymbol{\mu}_{t|t}$ and $\boldsymbol{\Sigma}_{t|t}$

Compare to batch inference: $O(T^3 n_z^3)$ *complexity!*

The Kalman Gain: Optimal Weighting

$$\mathbf{K}_t = \mathbf{\Sigma}_{t|t-1}\mathbf{H}_t^T(\mathbf{H}_t\mathbf{\Sigma}_{t|t-1}\mathbf{H}_t^T + \mathbf{R}_t)^{-1}$$

Interpretation:

$$\boldsymbol{\mu}_{t|t} = \boldsymbol{\mu}_{t|t-1} + \mathbf{K}_t \underbrace{(\mathbf{y}_t - \hat{\mathbf{y}}_t)}_{\text{innovation}}$$

- $\mathbf{K}_t$ determines how much we trust the new measurement vs. prediction
- Innovation $\mathbf{v}_t = \mathbf{y}_t - \hat{\mathbf{y}}_t$: "surprise" in the measurement
- Large $\mathbf{K}_t \rightarrow$ trust measurement more
- Small $\mathbf{K}_t \rightarrow$ trust prediction more

Extreme Cases

- If $\mathbf{R}_t \rightarrow 0$ (perfect measurements): $\mathbf{K}_t \rightarrow \mathbf{H}_t^{-1}$, trust measurement fully
- If $\mathbf{Q}_t \rightarrow 0$ (perfect model): $\mathbf{K}_t \rightarrow \mathbf{0}$, ignore measurements

Kalman Filter: 1D Scalar Case

Simplified state-space model

$$z_t = z_{t-1} + q_t, \qquad q_t \sim \mathcal{N}(0,\ q_t)$$

$$y_t = z_t + r_t, \qquad r_t \sim \mathcal{N}(0,\ r_t)$$

What was replaced (from the general matrix KF)

- $\boldsymbol{\mu}_{t|t} \to \mu_{t|t}$, $\quad \boldsymbol{\Sigma}_{t|t} \to \sigma^2_{t|t}$
- $F_t = 1$ so $F_t^2\sigma^2 \to \sigma^2$
- $H_t = 1$ so $S_t = H_t^2\sigma^2 + r_t \to \sigma^2 + r_t$
- $u_t = b_t = d_t = 0$ so $\hat{y}_t = H_t\mu_{t|t-1} \to \mu_{t|t-1}$
- $\mathbf{K}_t = \Sigma H^\top S^{-1} \to K_t = \sigma^2_{t|t-1}/(\sigma^2_{t|t-1} + r_t)$

Intuition: q_t makes uncertainty grow over time (prediction), r_t controls how much we trust the measurement (update).

Complete algorithm

Initialization: $\quad z_0 \sim \mathcal{N}(\mu_0, \sigma_0^2)$

For $t = 1, \dots, T$**:**

Prediction (time update):

$$\mu_{t|t-1} = \mu_{t-1|t-1} \qquad \sigma^2_{t|t-1} = \sigma^2_{t-1|t-1} + q_t$$

Update (measurement update):

$$\hat{y}_t = \mu_{t|t-1} \qquad S_t = \sigma^2_{t|t-1} + r_t$$

$$K_t = \frac{\sigma^2_{t|t-1}}{\sigma^2_{t|t-1} + r_t}$$

$$\mu_{t|t} = \mu_{t|t-1} + K_t\,(y_t - \hat{y}_t)$$

$$\sigma^2_{t|t} = (1 - K_t)\,\sigma^2_{t|t-1}$$

Kalman Gain: 1D Scalar Case

For scalar z_t, y_t, the Kalman gain simplifies to:

$$K_t = \frac{\sigma^2_{t|t-1}}{\sigma^2_{t|t-1} + r_t}$$

Special cases:

- If $\sigma^2_{t|t-1} \gg r_t$ (uncertain prediction, accurate measurement):

$$K_t \approx 1 \quad \Rightarrow \quad \mu_{t|t} \approx y_t$$

- If $\sigma^2_{t|t-1} \ll r_t$ (confident prediction, noisy measurement):

$$K_t \approx 0 \quad \Rightarrow \quad \mu_{t|t} \approx \mu_{t|t-1}$$

- If $\sigma^2_{t|t-1} = r_t$ (equal uncertainty):

$$K_t = 0.5 \quad \Rightarrow \quad \mu_{t|t} = 0.5\mu_{t|t-1} + 0.5y_t$$

The Kalman gain automatically balances prediction and measurement!

Example 1: 1D Position Tracking

Setup:

- State: z_t = position (scalar)
- Dynamics: random walk with drift
- Observation: noisy position measurement

Model:

$$z_t = z_{t-1} + v\Delta t + q_t, \quad q_t \sim \mathcal{N}(0, q) \tag{13}$$
$$y_t = z_t + r_t, \quad r_t \sim \mathcal{N}(0, r) \tag{14}$$

State space form:

- $F_t = 1,\ b_t = v\Delta t,\ Q_t = q$
- $H_t = 1,\ R_t = r$

See Python notebook for implementation and visualization

Example 2: Constant Velocity Model (2D Tracking)

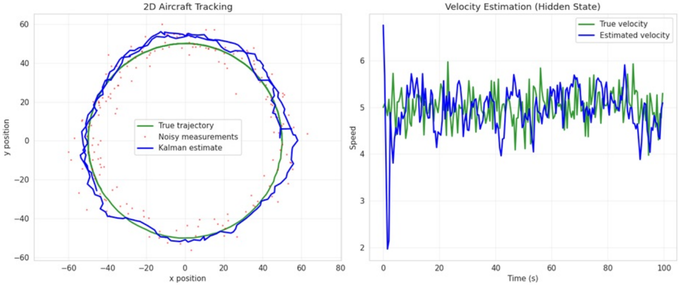
State: $\mathbf{z}_t = [x_t, \dot{x}_t, y_t, \dot{y}_t]^T$ (position + velocity in 2D)
Dynamics: Newton's laws with random acceleration

$$\mathbf{F}_t = \begin{bmatrix} 1 & \Delta t & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \Delta t \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{Q}_t = q \begin{bmatrix} \frac{\Delta t^4}{4} & \frac{\Delta t^3}{2} & 0 & 0 \\ \frac{\Delta t^3}{2} & \Delta t^2 & 0 & 0 \\ 0 & 0 & \frac{\Delta t^4}{4} & \frac{\Delta t^3}{2} \\ 0 & 0 & \frac{\Delta t^3}{2} & \Delta t^2 \end{bmatrix} \tag{15}$$

Observation: Only position is measured

$$\mathbf{H}_t = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \quad \mathbf{R}_t = \begin{bmatrix} \sigma_x^2 & 0 \\ 0 & \sigma_y^2 \end{bmatrix} \tag{16}$$

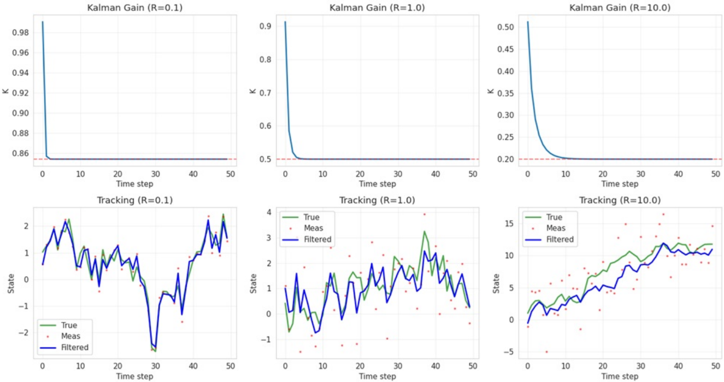
Example 2: Results



Key observations:

- Position estimate tracks ground truth well despite noisy measurements
- **Velocity is estimated despite not being measured!**
- Uncertainty (covariance) decreases as measurements accumulate
- Filter automatically adapts gain based on noise levels

Example 2: Kalman Gain Evolution



Key observations:

- Kalman gain K_t converges to steady-state value
- Initially: large gain (uncertain prior)
- After convergence: constant gain (balance prediction vs measurement)
- Variance decreases and stabilizes

Steady-State Kalman Filter: For time-invariant systems, we can pre-compute K_∞ by solving the **Discrete Algebraic Riccati Equation (DARE)**

Key Properties of Kalman Filter

1. **Optimality:**
   - Minimum Mean Squared Error (MMSE) estimator
   - Best Linear Unbiased Estimator (BLUE)
   - Maximum A Posteriori (MAP) estimator for Gaussians
2. **Consistency:**
   - Innovation $\mathbf{v}_t$ should be white noise: $\mathbf{v}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{S}_t)$
   - Covariance $\boldsymbol{\Sigma}_{t|t}$ represents true uncertainty (if model correct)
3. **Separability:**
   - Estimation and control can be designed independently
   - Basis for Linear Quadratic Gaussian (LQG) control

Limitations and Extensions

Kalman Filter Assumptions:

- Linear dynamics and observations
- Gaussian noise
- Known model parameters $(\mathbf{F}_t, \mathbf{H}_t, \mathbf{Q}_t, \mathbf{R}_t)$

What if assumptions are violated?

1. **Nonlinear systems:**
   - Extended Kalman Filter (EKF): linearize via Jacobians
   - Unscented Kalman Filter (UKF): sigma points
   - Particle Filter: Monte Carlo sampling
2. **Non-Gaussian noise:**
   - Mixture Kalman filters
   - Particle filters
3. **Unknown parameters:**
   - Expectation-Maximization (EM) algorithm
   - Adaptive Kalman filtering

Extended Kalman Filter (EKF)

Nonlinear system:

$$\mathbf{z}_t = f(\mathbf{z}_{t-1}, \mathbf{u}_t) + \mathbf{q}_t \tag{17}$$

$$\mathbf{y}_t = h(\mathbf{z}_t) + \mathbf{r}_t \tag{18}$$

Idea: Linearize around current estimate

Jacobians:

$$\mathbf{F}_t = \left.\frac{\partial f}{\partial \mathbf{z}}\right|_{\mathbf{z}=\boldsymbol{\mu}_{t-1|t-1}} \tag{19}$$

$$\mathbf{H}_t = \left.\frac{\partial h}{\partial \mathbf{z}}\right|_{\mathbf{z}=\boldsymbol{\mu}_{t|t-1}} \tag{20}$$

EKF equations: Same as Kalman filter, but use:

- $\boldsymbol{\mu}_{t|t-1} = f(\boldsymbol{\mu}_{t-1|t-1}, \mathbf{u}_t)$ instead of $\mathbf{F}_t \boldsymbol{\mu}_{t-1|t-1}$
- $\hat{\mathbf{y}}_t = h(\boldsymbol{\mu}_{t|t-1})$ instead of $\mathbf{H}_t \boldsymbol{\mu}_{t|t-1}$
- $\mathbf{F}_t, \mathbf{H}_t$ computed via Jacobians

Works well for mildly nonlinear systems; can diverge if highly nonlinear.

Deep Kalman Filters

Idea: Combine Kalman filtering with deep learning

Approach:

- Parameterize $f(\cdot), h(\cdot)$ with neural networks
- Learn from data using gradient descent
- Use Kalman filter for inference given learned model

Architectures:

1. **Variational RNN (VRNN):**
   - $p(\mathbf{z}_t \mid \mathbf{z}_{t-1})$ and $p(\mathbf{y}_t \mid \mathbf{z}_t)$ are neural networks
   - Approximate inference with amortized variational inference
2. **Neural ODE + Kalman:**
   - Continuous-time dynamics: $\frac{d\mathbf{z}}{dt} = f_\theta(\mathbf{z}, t)$
   - Use Kalman filter for discrete observations

Active research area: combining model-based and learning-based approaches

Modern multi-object visual trackers

Tracker	Year	State estimator / motion model	Tracking highlights
SORT	2016	Linear KF (CV)	Fast baseline; short-term linear motion
DeepSORT	2017	Linear KF (CV)	Adds deep ReID; longer occlusion tolerance
Tracktor++	2019	Detector bbox regression	Uses detector regression as motion model; +CMC/ReID
CenterTrack	2020	Learned offsets (no KF)	Tracks centers via regressed inter-frame offsets
ByteTrack	2021	Linear KF (CV)	Improves continuity by using low-score detections
BoT-SORT	2022	KF + camera motion comp.	Improved box state + CMC for stable motion
StrongSORT	2022	KF + motion/camera corr.	Upgraded DeepSORT baseline; optional smoothing/link
OC-SORT	2023	KF + ORU re-update	Observation-centric re-update reduces KF drift in occlusion
TrackFormer	2022	Track queries (Transformer)	End-to-end tracking-by-attention; no explicit KF

Summary: Why Kalman Filter Matters

Theoretical Elegance

- Optimal Bayesian solution for linear-Gaussian systems
- Closed-form recursive equations
- Beautiful mathematical structure

Practical Impact

- Real-time state estimation with constant memory
- Navigation: GPS, inertial guidance, autonomous vehicles
- Control: LQG controllers, Model Predictive Control
- Signal processing: noise reduction, prediction

Conceptual Foundation

- Template for all filtering algorithms (EKF, UKF, particle filters)
- Connects to modern ML: RNNs, VAEs, S4/Mamba models
- Embodiment of Bayesian recursive inference

Key Takeaways

1. **Predict-Update Cycle:**
    - Prediction: propagate state forward using dynamics
    - Update: incorporate new measurement via Bayes' rule
2. **Kalman Gain:**
    - Automatically balances model vs. measurement
    - Proportional to confidence in each source
3. **Optimality:**
    - MMSE estimator for linear-Gaussian systems
    - Recursive with $O(n_z^3)$ per step, $O(n_z^2)$ memory
4. **Extensions:**
    - EKF/UKF for nonlinear systems
    - RTS smoother for offline problems
    - Deep learning connections

Further Reading

Classic References:

- R. E. Kalman (1960), "A New Approach to Linear Filtering and Prediction Problems," *ASME Journal of Basic Engineering*. (doi:10.1115/1.3662552)
- H. E. Rauch, F. Tung, T. C. Striebel (1965), "Maximum Likelihood Estimates of Linear Dynamic Systems," *AIAA Journal*. (RTS smoother; doi:10.2514/3.3166)

Textbooks:

- K. P. Murphy, *Probabilistic Machine Learning: Advanced Topics* (MIT Press, 2023), Ch. 8 (Gaussian filtering & smoothing).
- S. Särkkä & L. Svensson, *Bayesian Filtering and Smoothing*, 2nd ed. (Cambridge Univ. Press, 2023).
- D. Simon, *Optimal State Estimation: Kalman, H_∞, and Nonlinear Approaches* (Wiley, 2006).

Tutorials:

- G. Welch & G. Bishop, "An Introduction to the Kalman Filter," UNC TR 95-041 (1995; updated 2006).
- Y. Pei, S. Biswas, D. S. Fussell, K. Pingali, "An Elementary Introduction to Kalman Filtering," arXiv:1710.04055 (2017). (Also appears as an ACM tutorial article, 2019.)

Modern Extensions (SSM-based sequence models):

- A. Gu, K. Goel, C. Ré, "Efficiently Modeling Long Sequences with Structured State Spaces (S4)," ICLR 2022 (arXiv:2111.00396).
- A. Gu & T. Dao, "Mamba: Linear-Time Sequence Modeling with Selective State Spaces," arXiv:2312.00752 (2023).

Acknowledgements/ Document preparation

This document was developed and converted into LaTeX slides and formatted with assistance from ChatGPT (OpenAI) and CoPilot (Microsoft). The instructor modified the source text and verified the final structure and wording.