Examples up to now have been discrete (binary) random variables.

Kalman ‘filtering’ can be seen as a special case of a temporal inference with continuous random variables.

Everything is continuous...

\[ \mathbf{x} \quad \mathbf{e} \quad P(\mathbf{x}_0) \quad P(e|\mathbf{x}) \quad P(\mathbf{x}_t|\mathbf{x}_{t-1}) \]

probability distributions are no longer tables but functions
Making the connection to the ‘filtering’ equations

(Discrete) Filtering

\[ P(X_{t+1}|e_{1:t+1}) \propto P(e_{t+1}|X_{t+1}) \sum_{X_t} P(X_{t+1}|X_t)P(X_t|e_{1:t}) \]

Kalman Filtering

\[ P(X_{t+1}|e_{1:t+1}) \propto P(e_{t+1}|X_{t+1}) \int_{x_t} P(X_{t+1}|x_t)P(x_t|e_{1:t})dx_t \]

integral because continuous PDFs
Simple, 1D example…

\( x \)
\[ x_t = x_{t-1} + s + r_t \]

\[ r_t \sim \mathcal{N}(0, \sigma_R) \]

'Sampled from'

System (motion) model
How do you represent the motion model?

\[ x_t = x_{t-1} + s + r_t \]

\[ r_t \sim \mathcal{N}(0, \sigma_R) \]

\[ P(x_t | x_{t-1}) \]
How do you represent the motion model?

A linear Gaussian (continuous) transition model

\[ P(x_t \mid x_{t-1}) = \mathcal{N}(x_t; x_{t-1} + s, \sigma_r) \]

How can you visualize this distribution?
A linear Gaussian (continuous) transition model

$$P(x_t | x_{t-1}) = \mathcal{N}(x_t; x_{t-1} + s, \sigma_r)$$

Why don’t we just use a table as before?
The observation (measurement) model is given by:

\[ z_t = x_t + q_t \]

where

\[ q_t \sim \mathcal{N}(0, \sigma_Q) \]

is sampled from a Gaussian distribution.
How do you represent the observation (measurement) model?

\[ z_t = x_t + q_t \]

\[ q_t \sim \mathcal{N}(0, \sigma_Q) \]
How do you represent the observation (measurement) model?

Also a linear Gaussian model

\[ P(z_t | x_t) = \mathcal{N}(z_t; x_t, \sigma_Q) \]
How do you represent the observation (measurement) model?

Also a linear Gaussian model

\[ P(z_t | x_t) = \mathcal{N}(z_t; x_t, \sigma_Q) \]
Prior (initial) State

$x_0$ true position

$\hat{x}_0$ initial estimate

initial estimate uncertainty $\sigma_0$
How do you represent the prior state probability?
How do you represent the prior state probability?

Also a linear Gaussian model!

\[ P(\hat{x}_0) = \mathcal{N}(\hat{x}_0; x_0, \sigma_0) \]
How do you represent the prior state probability?

Also a linear Gaussian model

\[ P(\hat{x}_0) = \mathcal{N}(\hat{x}_0; x_0, \sigma_0) \]
Inference

So how do you do temporal filtering with the KL?
Recall: the first step of filtering was the ‘prediction step’

\[ P(X_{t+1}|e_{1:t+1}) \propto P(e_{t+1}|X_{t+1}) \int_{x_t} P(X_{t+1}|x_t)P(x_t|e_{1:t})dx_t \]

prediction step

compute this! It’s just another Gaussian

need to compute the ‘prediction’ mean and variance…
Prediction
(Using the motion model)

How would you predict $\hat{x}_1$ given $\hat{x}_0$?

using this ‘cap’ notation to denote ‘estimate’

\[
\hat{x}_1 = \hat{x}_0 + s \quad \text{(This is the mean)}
\]

\[
\sigma_1^2 = \sigma_0^2 + \sigma_r^2 \quad \text{(This is the variance)}
\]
\[ P(X_{t+1}|e_{1:t+1}) \propto P(e_{t+1}|X_{t+1}) \int_{x_t} P(X_{t+1}|x_t) P(x_t|e_{1:t}) dx_t \]

the second step after prediction is …
... update step!

\[ P(X_{t+1}|e_{1:t+1}) \propto P(e_{t+1}|X_{t+1}) \int_{x_t} P(X_{t+1}|x_t)P(x_t|e_{1:t})dx_t \]

compute this
(using results of the prediction step)
In the **update step**, the **sensor measurement** corrects the system **prediction**

Which estimate is correct? Is there a way to know?

*Is there a way to merge this information?*
*Intuitively*, the smaller variance mean less uncertainty.

So we want a weighted state estimate correction

\[
\hat{x}_1^+ = \frac{\sigma_q^2}{\sigma_1^2 + \sigma_q^2} \hat{x}_1 + \frac{\sigma_1^2}{\sigma_1^2 + \sigma_q^2} \hat{z}_1
\]

*This happens naturally in the Bayesian filtering (with Gaussians) framework!*
Recall the filtering equation:

\[ P(X_{t+1}|e_{1:t+1}) \propto P(e_{t+1}|X_{t+1}) \int_{x_t} P(X_{t+1}|x_t) P(x_t|e_{1:t}) dx_t \]

**Observation**

**One step motion prediction**

What is the product of two Gaussians?
When we multiply the prediction (Gaussian) with the observation model (Gaussian) we get ...

... a product of two Gaussians

\[ \mu = \frac{\mu_1 \sigma_2^2 + \mu_2 \sigma_1^2}{\sigma_2^2 + \sigma_1^2} \]

\[ \sigma = \frac{\sigma_1^2 \sigma_2^2}{\sigma_1^2 + \sigma_2^2} \]

applied to the filtering equation...
\[ P(X_{t+1}|e_{1:t+1}) \propto P(e_{t+1}|X_{t+1}) \int_{x_t} P(X_{t+1}|x_t)P(x_t|e_{1:t})dx_t \]

mean: \( z_1 \)
variance: \( \sigma_q \)

mean: \( \hat{x}_1 \)
variance: \( \sigma_1 \)

new mean: \[ \hat{x}_1^+ = \frac{\hat{x}_1 \sigma_q^2 + z_1 \sigma_1^2}{\sigma_q^2 + \sigma_1^2} \]
new variance: \[ \hat{\sigma}_1^+ = \frac{\sigma_q^2 \sigma_1^2}{\sigma_q^2 + \sigma_1^2} \]

'plus' sign means post 'update' estimate
With a little algebra…

\[
\hat{x}_1^+ = \frac{\hat{x}_1 \sigma_q^2 + z_1 \sigma_1^2}{\sigma_q^2 + \sigma_1^2} = \hat{x}_1 \frac{\sigma_q^2}{\sigma_q^2 + \sigma_1^2} + z_1 \frac{\sigma_1^2}{\sigma_q^2 + \sigma_1^2}
\]

We get a **weighted state estimate correction**!
Kalman gain notation

With a little algebra...

$$\hat{x}_1^+ = \hat{x}_1 + \frac{\sigma_1^2}{\sigma_q^2 + \sigma_1^2} (z_1 - \hat{x}_1) = \hat{x}_1 + K (z_1 - \hat{x}_1)$$

'Kalman gain'  'Innovation'

With a little algebra...

$$\sigma_1^+ = \frac{\sigma_1^2 \sigma_q^2}{\sigma_1^2 + \sigma_q^2} = \left(1 - \frac{\sigma_1^2}{\sigma_1^2 + \sigma_q^2}\right) \sigma_1^2 = (1 - K) \sigma_1^2$$
Summary (1D Kalman Filtering)

To solve this...

\[
P(X_{t+1|e_1:t+1}) \propto P(e_{t+1|X_{t+1}}) \int_{x_t} P(X_{t+1|x_t})P(x_t|e_1:t)dx_t
\]

Compute this...

\[
\hat{x}_1^+ = \hat{x}_1 + \frac{\sigma_1^2}{\sigma_1^2 + \sigma_q^2} (z_1 - \hat{x}_1) \quad \sigma_1^{2+} = \sigma_1^2 - \frac{\sigma_1^2}{\sigma_1^2 + \sigma_q^2} \sigma_1^2
\]

\[K = \frac{\sigma_1^2}{\sigma_1^2 + \sigma_q^2}\]

‘Kalman gain’

\[
\hat{x}_1^+ = \hat{x}_1 + K(z_1 - \hat{x}_1) \quad \sigma_1^{2+} = \sigma_1^2 - K \sigma_1^2
\]

mean of the new Gaussian

variance of the new Gaussian
Simple 1D Implementation

\[
\begin{bmatrix} x & p \end{bmatrix} = \text{KF}(x, v, z)
\]

\[
x = x + s;
\]
\[
v = v + q;
\]

\[
K = \frac{v}{v + r};
\]

\[
x = x + K \times (z - x);
\]
\[
p = v - K \times v;
\]

Just 5 lines of code!
or just 2 lines

\[
\begin{align*}
[x \ P] &= KF(x, v, z) \\
x &= (x+s) + (v+q) / ((v+q)+r) * (z-(x+s)) ; \\
p &= (v+q) - (v+q) / ((v+q)+r) * v ;
\end{align*}
\]
Bare computations (algorithm) of Bayesian filtering:

\[
\text{KalmanFilter}(\mu_{t-1}, \Sigma_{t-1}, u_t, z_t)
\]

\[
\bar{\mu}_t = A_t \mu_{t-1} + B u_t \quad \text{‘old’ mean}
\]

\[
\bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^\top + R \quad \text{Gaussian noise}
\]

\[
K_t = \bar{\Sigma}_t C_t^\top (C_t \bar{\Sigma}_t C_t^\top + Q_t)^{-1}
\]

\[
\mu_t = \bar{\mu}_t + K_t (z_t - C_t \bar{\mu}_t) \quad \text{observation model}
\]

\[
\Sigma_t = (I - K_t C_t) \bar{\Sigma}_t \quad \text{Update}
\]
Simple Multi-dimensional Implementation
(also 5 lines of code!)

\[
[x \ P] = KF(x, P, z)
\]

\[
x = A*x;
\]
\[
P = A*P*A' + Q;
\]

\[
K = P*C' / (C*P*C' + R);
\]

\[
x = x + K*(z - C*x);
\]
\[
P = (eye(size(K,1))-K*C)*P;
\]
2D Example
x

state

measurement

\[ x = \begin{bmatrix} x \\ y \end{bmatrix} \quad z = \begin{bmatrix} x \\ y \end{bmatrix} \]

Constant position Motion Model

\[ x_t = Ax_{t-1} + Bu_t + \epsilon_t \]
Constant position Motion Model

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

Constant position

\[ A = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad Bu = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \quad R = \begin{bmatrix} \sigma_r^2 & 0 \\ 0 & \sigma_r^2 \end{bmatrix} \]

System noise

\[ \epsilon_t \sim \mathcal{N}(0, R) \]
\[ x = \begin{bmatrix} x \\ y \end{bmatrix} \quad z = \begin{bmatrix} x \\ y \end{bmatrix} \]

Measurement Model

\[ z_t = C_t x_t + \delta_t \]
state measurement

\[ \mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix} \quad \mathbf{z} = \begin{bmatrix} x \\ y \end{bmatrix} \]

Measurement Model

\[ z_t = C_t x_t + \delta_t \]

zero-mean measurement noise

\[ C = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad \delta_t \sim \mathcal{N}(0, Q) \quad Q = \begin{bmatrix} \sigma^2_q & 0 \\ 0 & \sigma^2_q \end{bmatrix} \]
Algorithm for the 2D object tracking example

General Case

\[ \tilde{\mu}_t = A_t \mu_{t-1} + B u_t \]
\[ \tilde{\Sigma}_t = A_t \Sigma_{t-1} A_t^\top + R \]
\[ K_t = \tilde{\Sigma}_t C_t^\top (C_t \tilde{\Sigma}_t C_t^\top + Q_t)^{-1} \]
\[ \mu_t = \tilde{\mu}_t + K_t (z_t - C_t \tilde{\mu}_t) \]
\[ \Sigma_t = (I - K_t C_t) \tilde{\Sigma}_t \]

Constant position Model

\[ \bar{x}_t = x_{t-1} \]
\[ \bar{\Sigma}_t = \Sigma_{t-1} + R \]
\[ K_t = \bar{\Sigma}_t (\bar{\Sigma}_t + Q)^{-1} \]
\[ x_t = \bar{x}_t + K_t (z_t - \bar{x}_t) \]
\[ \Sigma_t = (I - K_t) \bar{\Sigma}_t \]

\[ A = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad C = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \]

motion model
observation model
Just 4 lines of code

\[
\begin{align*}
[x \ P] &= \text{KF\_constPos}(x, P, z) \\
P &= P + Q; \\
K &= \frac{P}{P + R}; \\
x &= x + K \times (z - x); \\
P &= (\text{eye(size(K,1))} - K) \times P;
\end{align*}
\]

Where did the 5th line go?
General Case

\[\hat{\mu}_t = A_t \mu_{t-1} + B u_t\]
\[\bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + R\]
\[K_t = \bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + Q_t)^{-1}\]
\[\mu_t = \hat{\mu}_t + K_t (z_t - C_t \mu_t)\]
\[\Sigma_t = (I - K_t C_t) \bar{\Sigma}_t\]

Constant position Model

\[\bar{x}_t = x_{t-1}\]
\[\bar{\Sigma}_t = \Sigma_{t-1} + R\]
\[K_t = \bar{\Sigma}_t (\bar{\Sigma}_t + Q)^{-1}\]
\[x_t = \bar{x}_t + K_t (z_t - \bar{x}_t)\]
\[\Sigma_t = (I - K_t) \bar{\Sigma}_t\]