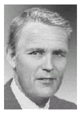
Course handouts

Data Fusion

Jindong Tan
Electrical and Computer Engineering
Michigan Technological University
jitan@mtu.edu

Rudolf Emil Kalman

- Born 1930 in Hungary
- BS and MS from MIT
- PhD 1957 from Columbia
- Filter developed in 1960-61
- Now retired



Applications

- Tracking missiles
- Tracking heads/drumsticks
- Extracting lip motion from video
- Fitting Bezier patches to point data
- Lots of computer vision applications
- Economics
- Navigation

Start from an example

$$z_1, \sigma_{z_1}^2$$
$$\hat{x}_1 = z_1$$
$$\hat{\sigma}_1^2 = \sigma_{z_1}^2$$

Example

$$z_2, \sigma_{z_2}^2$$
$$\hat{x}_2 = ?$$
$$\hat{\sigma}_2^2 = ?$$

Combine estimate

$$\hat{x}_2 = \hat{x}_1 + k_2(z_2 - \hat{x}_1)$$
$$k_2 = \frac{\sigma_1^2}{\sigma_1^2 + \sigma_{z2}^2}$$

Combine variance

$$\frac{1}{\sigma_2^2} = \frac{1}{\sigma_1^2} + \frac{1}{\sigma_{z2}^2}$$

Combined Estimates

$$\hat{x} = \hat{x}_2$$
$$\hat{\sigma}^2 = \sigma_2^2$$

Dynamic systems

- Not *all* the difference is error
- Some may be motion
- KF can include a motion model
- Estimate velocity and position

Process Model

- Describes how the *state* changes over time
- The *state* for the first example was scalar
- The *process model* was "nothing changes"

- A better model might be
  - State is a 2-vector [position, velocity]
  - $\text{position}_{n+1} = \text{position} + \text{velocity}_n * \text{time}$
  - $\text{velocity}_{n+1} = \text{velocity}_n$

Measurement Model

- "What you see from where you are"
- not
- "Where you are from what you see"

Predict – Correct cycle

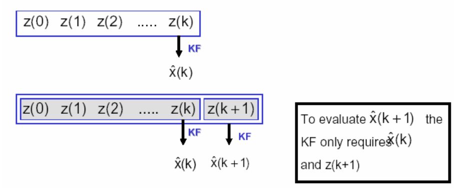
- **KF operates by**
  - **Predicting the new state and its uncertainty**
  - **Correcting with the new measurement**

The concept

- **Optimal** Recursive Data Processing Algorithm
  - x(k+1)=f(x(k), u(k), w(k))
  - z(k+1)=h(x(k),v(k))
  - x - state
  - f - system dynamics
  - h - measurement function
  - u - controls
  - w - system error sources
  - v - measurement error sources
  - z - observed measurements
- **Given**
  - f, h, noise characterization, initial conditions
  - z(0), z(1), z(2), …, z(k)
- **Obtain**
  - the **"best"** estimate of x(k)

The concept

- Optimal **Recursive** Data Processing Algorithm
  - the KF does not require all previous data to be kept in storage and reprocessed every time a new measurement is taken.



Example: Process Model

$$\begin{bmatrix} x_k \\ y_k \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_{k-1} \\ y_{k-1} \end{bmatrix} + \begin{bmatrix} \sim x_{k-1} \\ \sim y_{k-1} \end{bmatrix}$$

$$\overline{x}_k = A\overline{x}_{k-1} + \overline{w}_{k-1}$$

Example: Measurement Model

$$\begin{bmatrix} u_k \\ v_k \end{bmatrix} = \begin{bmatrix} H_x & 0 \\ 0 & H_y \end{bmatrix} \begin{bmatrix} x_k \\ y_k \end{bmatrix} + \begin{bmatrix} \sim u_k \\ \sim v_k \end{bmatrix}$$

$$\overline{z}_k = H\overline{x}_k + \overline{v}_k$$

Preparation

$$A = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$ State Transition

$$Q = E\{\overline{w} * \overline{w}^T\} = \begin{bmatrix} Q_{xx} & 0 \\ 0 & Q_{yy} \end{bmatrix}$$ Process Noise Covariance

$$R = E\{\overline{v} * \overline{v}^T\} = \begin{bmatrix} R_{xx} & 0 \\ 0 & R_{yy} \end{bmatrix}$$ Measurement Noise Covariance

Initialization

$$\overline{x}_0 = Hz_0$$

$$P_0 = \begin{bmatrix} \varepsilon & 0 \\ 0 & \varepsilon \end{bmatrix}$$

Predict

$$\overline{x}_k^- = A\overline{x}_{k-1}$$
$$P_k^- = AP_{k-1}A^T + Q$$

Correct

$$\overline{x}_k = \overline{x}_k^- + K\left(\underbrace{\overline{z}_k}_{\text{actual}} - \underbrace{H\overline{x}_k^-}_{\text{predicted}}\right)$$
$$P_k = (I - KH)P_k^-$$

$$K = P_k^- H^T \left(HP_k^- H^T + R\right)^{-1}$$

Summary- two steps

$$\overline{x}_k^- = A\overline{x}_{k-1}$$
$$P_k^- = AP_{k-1}A^T + Q$$

$$K = P_k^- H^T \left(HP_k^- H^T + R\right)^{-1}$$
$$\overline{x}_k = \overline{x}_k^- + K\left(\overline{z}_k - H\overline{x}_k^-\right)$$
$$P_k = (I - KH)P_k^-$$

What does a Kalman Filter do, anyway?

Given the linear dynamical system:

$$x(k+1) = F(k)x(k) + G(k)u(k) + v(k)$$
$$y(k) = H(k)x(k) + w(k)$$

$x(k)$ is the n-dimensional state vector (unknown)
$u(k)$ is the m-dimensional input vector (known)
$y(k)$ is the p-dimensional output vector (known, measured)
$F(k), G(k), H(k)$ are appropriately dimensioned system matrices (known)
$v(k), w(k)$ are zero-mean, white Gaussian noise with (known) covariance matrices $Q(k), R(k)$

the Kalman Filter is a recursion that provides the "best" estimate of the state vector *x*.

What's so great about that?

$$x(k+1) = F(k)x(k) + G(k)u(k) + v(k)$$
$$y(k) = H(k)x(k) + w(k)$$

- noise smoothing (improve noisy measurements)
- state estimation (for state feedback)
- recursive (computes next estimate using only most recent measurement)

How does it work?

$$x(k+1) = F(k)x(k) + G(k)u(k) + v(k)$$
$$y(k) = H(k)x(k) + w(k)$$

1. prediction based on last estimate:

$$\hat{x}(k+1|k) = F(k)\hat{x}(k|k) + G(k)u(k)$$
$$\hat{y}(k) = H(k)\hat{x}(k+1|k)$$

2. calculate correction based on prediction and current measurement:

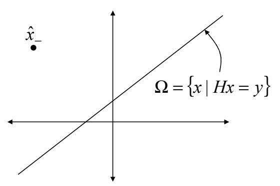
$$\Delta x = f\left(y(k+1), \hat{x}(k+1|k)\right)$$

3. update prediction: $\hat{x}(k+1|k+1) = \hat{x}(k+1|k) + \Delta x$

Finding the correction (no noise!)

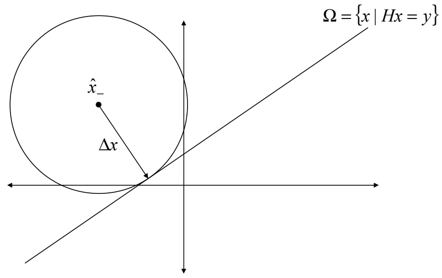
$$y = Hx$$

Given prediction $\hat{x}_-$ and output y, find Δx so that $\hat{x} = \hat{x}_- + \Delta x$ is the "best" estimate of x.



$$\boxed{\Delta x = H^T \left(HH^T\right)^{-1} v}$$

A Geometric Interpretation



LTI Kalman Filter Summary

System:
$$x(k+1) = Fx(k) + Gu(k) + v(k)$$
$$y(k) = Hx(k) + w(k)$$

Kalman Filter

1. Predict

$$\boxed{\begin{aligned} \hat{x}(k+1 \mid k) &= F\hat{x}(k \mid k) + Gu(k) \\ P(k+1 \mid k) &= FP(k \mid k)F^T + Q \end{aligned}}$$

2. Correction

$$\boxed{\begin{aligned} S &= HP(k+1 \mid k)H^T + \mathrm{R} \\ W &= P(k+1 \mid k)HS^{-1} \\ \Delta x &= W\left(y(k+1) - H\hat{x}(k+1 \mid k)\right) \end{aligned}}$$

3. Update

$$\boxed{\begin{aligned} \hat{x}(k+1 \mid k+1) &= \hat{x}(k+1 \mid k) + Wv \\ P(k+1 \mid k+1) &= P(k+1 \mid k) - WSW^T \end{aligned}}$$