CS 460/560
Introduction to Computational Robotics
Fall 2019, Rutgers University

Lecture 06
Kalman Filter Intro

Instructor: Jingjin Yu

Outline

Uncertainty

Model of dynamical systems

Bayesian filtering: the concept

An illustrative example

Applications of Kalman filters

Derivation of Kalman Filter

A 1D example

Uncertainty

An everyday experience

⇨Where **exactly** are we?

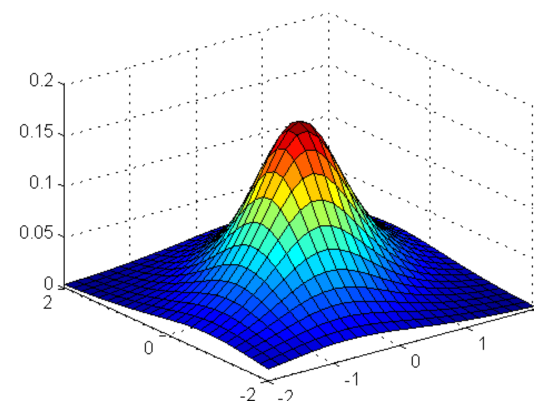
⇨E.g., in a classroom

⇨We do not know!

⇨Position is estimated

⇨We can get philosophical

A more accurate model



⇨A **dynamical system** with state x (a function of time)

⇨Positions (i.e., x) are estimates

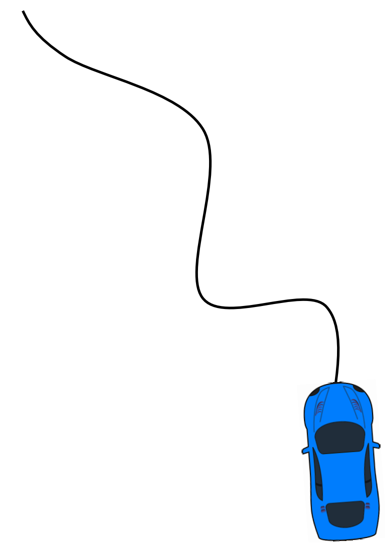
⇨Associate each location x with some probability

⇨This gives us a probability distribution $P(x)$

⇨For a robot, $P(x)$ changes as the robot moves around

⇨**Kalman filter** (and other **Bayesian filters**) tracks $P(x)$ as x changes over time

Modeling Dynamical Systems



A **dynamical system** (e.g., a car) is often modeled as

$$\dot{x} = f(x, u)$$

- ⇨ x: the **state** of the system,
  - ⇨ E.g., for a car, $x = (x_1, x_2, \theta)$
  - ⇨ $\dot{x} = \frac{dx}{dt}$ is the time derivative, i.e., the velocity of the system
  - ⇨ For a car, $\dot{x} = (\dot{x_1}, \dot{x_2}, \dot{\theta})$
- ⇨ u: the **control input**
  - ⇨ E.g., for a real car, $u = (\theta, v)$ (one possible control)
    - ⇨ θ is the front wheel bearing
    - ⇨ v is the forward speed (for a 2-wheel drive, assuming no slippery)
  - ⇨ u may be speed, acceleration, and so on…
- ⇨ f: system **evolution** function
  - ⇨ How do x, u determine $\dot{x}$

In **discrete** settings, often written as $x_t = f(x_{t-1}, u_{t-1})$

- ⇨ May view this as integration of the continuous model: $x_t = x_{t-1} + \int_{t-1}^{t} \dot{x} dt$
- ⇨ Often written as $x_k = f(x_{k-1}, u_{k-1})$

Modeling Dynamical Systems, Continued

Examples

⇨ A car going at fixed speed along x_1-axis: $\dot{x}_1 = 1$

⇨ In this case, $f(x, u) = 1$ is a constant

⇨ An accelerating car along x_1-axis with acceleration a: $\dot{x}_1 = at$

⇨ $u = a$, the acceleration, $f(x, u) = at$, does not depend on x

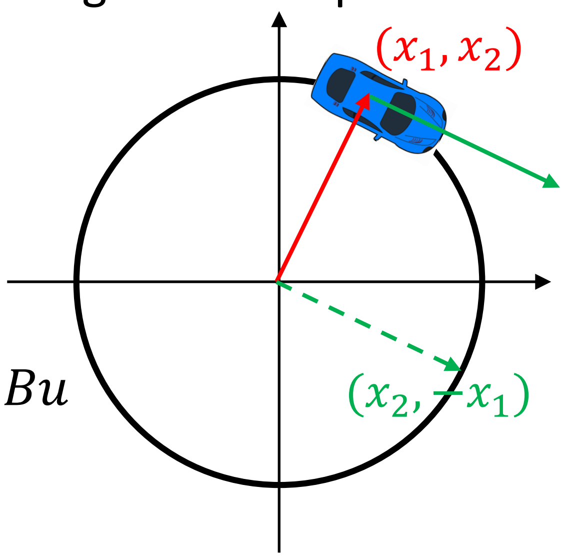
⇨ A car going clockwise along the unit circle around the origin at unit speed

⇨ $\dot{x} = (\dot{x}_1, \dot{x}_2, \dot{\theta}) = (x_2, -x_1, -1)$

⇨ Initial condition: $x_1 = 1, x_2 = 0$

⇨ The car will keep circling the unit circle at unit speed

⇨ So it takes 2π time to go one round



Linear and non-linear systems

⇨ Linear systems: f is a linear function, e.g., $\dot{x} = Ax + Bu$

⇨ Non-linear systems: f is non-linear

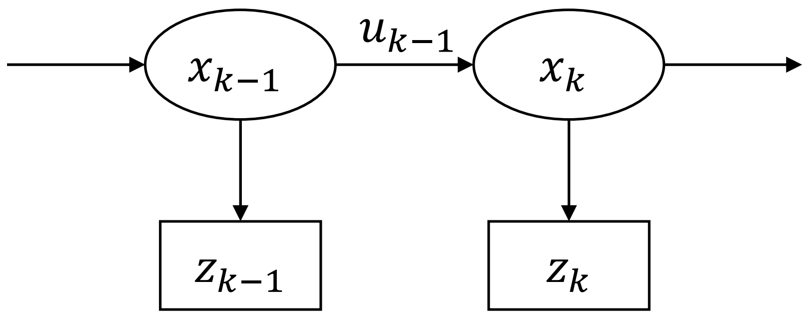
What to grasp from the last two slides?

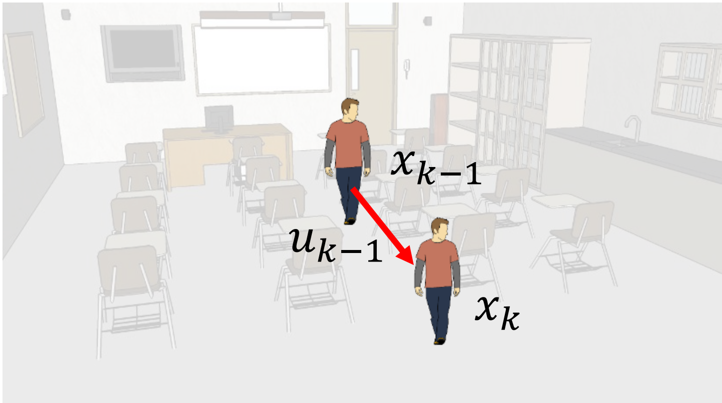
⇨ Dynamical systems may be modeled as what we have described

⇨ In particular, given x_{k-1}, u_{k-1}, and $f(x, u)$, we can **predict** x_k

Kalman Filter as a Bayesian Filter

Kalman filter is a type of Bayesian filters over a Hidden Markov model





- ⇨ x_is are **hidden (actual)** system states
- ⇨ They cannot be known exactly

Other examples: temperature of a room, population

- ⇨ We can **only observe** x_i using sensors to get z_i

Thermometer, census

- ⇨ The (discrete) process is modeled as a two-step iterative one
  - ⇨ Noisy state change: $\boxed{x_k = f(x_{k-1}, u_{k-1})} + \omega_{k-1}$
  - ⇨ Noisy measurement after state change: $z_k = h(x_k) + v_k$
  - ⇨ More details coming up

"white noises"

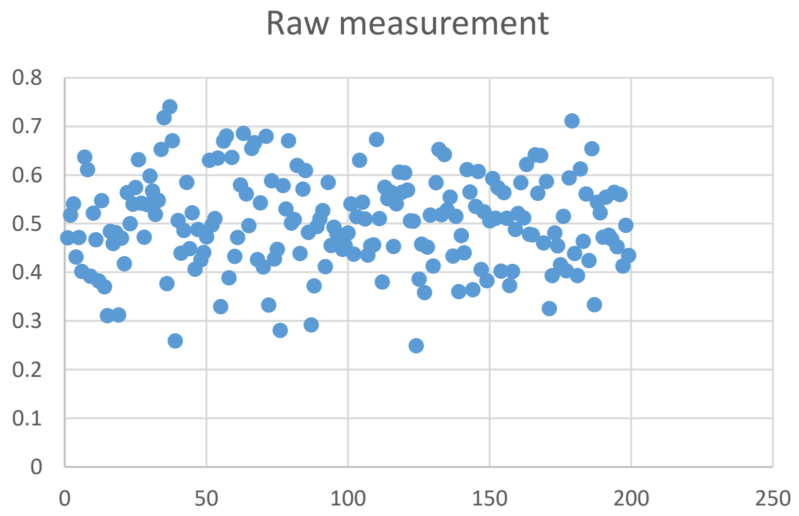
A dynamical system

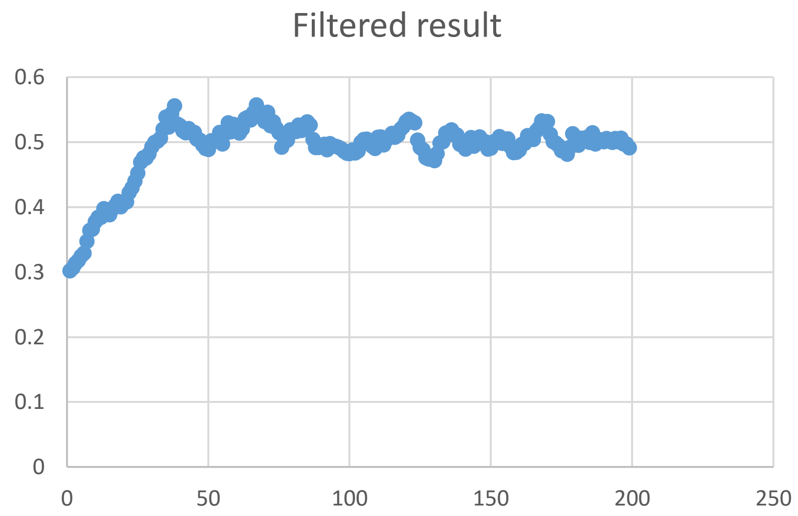
- ⇨ The "data" that we get are $u_0, z_1, u_1, z_2, u_2, z_3, \ldots$
- ⇨ We want to provide $\hat{x}_k$ as an accurate estimate of x_k
- ⇨ Yields **Kalman filters**, particle filters, and so on

An Example

A hypothetical measurement of a variable x

- ⇨Mean: 0.5
- ⇨Variance: 0.01
- ⇨200 sequential measurements
- ⇨Note: only the mean is shown in the second figure, not the variance





Important: Kalman filter is not simple averaging!

- ⇨It has (limited) predictive power

Applications

Numerous applications

- ⇨GPS
  - ⇨ Minimizes error in tracking the position in altitude, latitude, longitude
  - ⇨ Reduces error from ~ 30 meters to less than 5 meters
- ⇨Aircraft autopilot
  - ⇨ Internal inertial guidance system generates errors over time
  - ⇨ Minimize with a 6D Kalman filter: yaw, pitch, roll, altitude, latitude, longitude
- ⇨Many, many other similar applications
  - ⇨ Missile guidance
  - ⇨ Radar
  - ⇨ Economic signals, e.g., stock time series
  - ⇨ Weather forecasting
  - ⇨ …

Kalman Filter in More Detail

Kalman filter is a minimum mean square estimator (MMSE) for estimating the state $x \in \mathbb{R}^n$ of a discrete-time controlled process with a linear system equation and a linear observer under "white noise".

⇨Linear stochastic system

$$\underbrace{x_k = Ax_{k-1} + Bu_{k-1}}_{\text{Linear}} + \omega_{k-1}, \qquad \omega_{k-1} \sim \underbrace{N(0, Q)}_{\text{Gaussian}} \qquad (1)$$

⇨With a linear observer (sensor)

$$\underbrace{z_k = Hx_k}_{\text{Linear}} + \nu_k, \qquad \nu_k \sim \underbrace{N(0, R)}_{\text{Gaussian}} \qquad (2)$$

⇨ω and ν are unknown but independent (i.e., Q and R are unknown)

⇨Kalman filter tries to provide estimates of true x_k through the minimization of the estimation error based on (1) and (2)

⇨It does the minimization using the MMSE

Kalman Filter in More Detail

Kalman filter is a minimum mean square estimator (MMSE) for estimating the state $x \in \mathbb{R}^n$ of a discrete-time controlled process with a linear system equation and a linear observer under "white noise".

⇨Linear stochastic system

A: State transition model

Q: Covariance of process noise

$$\underbrace{x_k = Ax_{k-1} + Bu_{k-1}} + \omega_{k-1}, \qquad \omega_{k-1} \sim \underbrace{N(0,Q)}_{\text{Gaussian}} \qquad (1)$$

B: Control-input model

⇨With a linear observer (sensor)

H: Observation model

R: Covariance of observation noise

$$\underbrace{z_k = Hx_k}_{\text{Linear}} + \nu_k, \qquad \nu_k \sim \underbrace{N(0,R)}_{\text{Gaussian}} \qquad (2)$$

⇨ω and ν are unknown but independent (i.e., Q and R are unknown)

⇨Kalman filter tries to provide estimates of true x_k through the minimization of the estimation error based on (1) and (2)

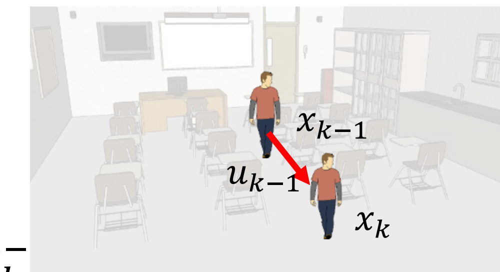
⇨It does the minimization using the MMSE

Input and Output of a Kalman Filter

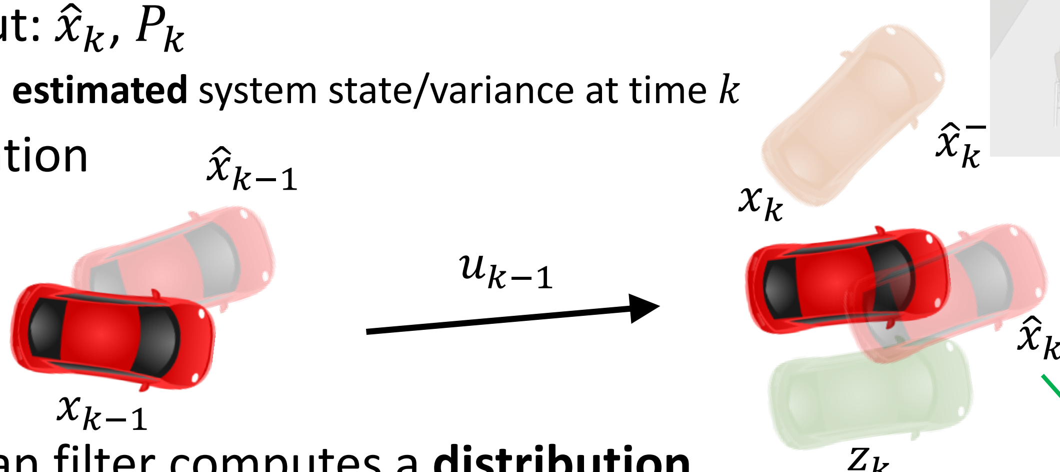
A Kalman filter provides an estimate of x_k under uncertainty

This is an **iterative** process

- ⇨In each iteration, the input: $\hat{x}_{k-1}$, P_{k-1}, u_{k-1}, z_k,A, B, H, Q, R (Q, R esti.)
  - ⇨ $\hat{x}_{k-1}$, P_{k-1}: **estimated** system state/variance at time $k-1$; $\hat{x}_0$, P_0 are guessed
  - ⇨ u_{k-1}: system input at time $k-1$, e.g., how hard the gas pedal is pressed
  - ⇨ $f(x_{k-1}, u_{k-1}) = Ax_{k-1} + Bu_{k-1}$, A and B are known
  - ⇨ z_k=Hx_k: the observation of x_k, H is known



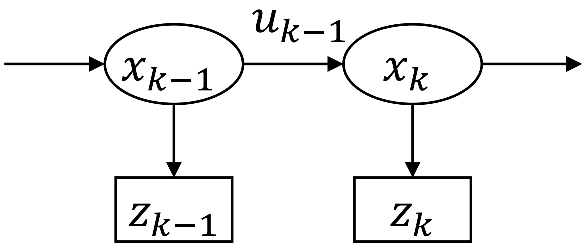
- ⇨The output: $\hat{x}_k$, P_k
  - ⇨ $\hat{x}_k$, P_k: **estimated** system state/variance at time k
- ⇨An illustration



These are the "mean" part

- ⇨The Kalman filter computes a **distribution**
  - ⇨ The estimate is not a single value! For 1D, two values: mean + variance
  - ⇨ For dimension n, an n-vector and an $n \times n$ covariance matrix
  - ⇨ Same applies to other variables

Deriving the MMSE (I)



The goal is to find true state x_k

⇨ But recall this is not possible because x_k is a hidden state

A **trick**: x, P, A, B, u, z… are high dimensional, but can treat as 1D

In each iteration, a Kalman filter does two updates

⇨ Time update: $\hat{x}_k^- = A\hat{x}_{k-1} + Bu_{k-1}$

⇨ Error of this step $e_k^- \equiv x_k - \hat{x}_k^-$

⇨ (a priori) Variance is $P_k^- = E[e_k^- e_k^{-\mathrm{T}}]$

⇨ Measurement update: $\hat{x}_k = \hat{x}_k^- + K_k(z_k - H\hat{x}_k^-)$

⇨ The term $(z_k - H\hat{x}_k^-)$ is called the measurement innovation

⇨ It gives us the "new information" from z_k that is not already in $\hat{x}_k^-$

⇨ Error of this step $e_k \equiv x_k - \hat{x}_k$

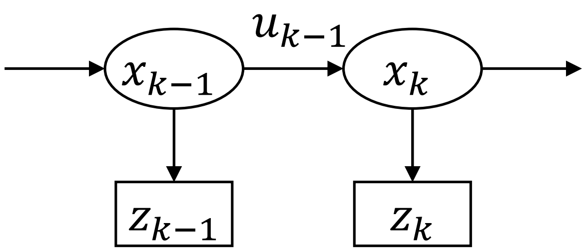
⇨ (a posteriori) Variance is $P_k = E[e_k e_k^{\mathrm{T}}]$

Kalman filter seeks the best K_k to minimize $E[\|e_k\|^2]$

⇨ This is the same as minimizing the **trace** of P_k

⇨ $e_k \equiv x_k - \hat{x}_k = x_k - \hat{x}_k^- - K_k(z_k - H\hat{x}_k^-)$

Deriving the MMSE (II)



$$e_k \equiv x_k - \hat{x}_k = x_k - \hat{x}_k^- - K_k(z_k - H\hat{x}_k^-)$$
$$= x_k - \hat{x}_k^- - K_k(z_k - Hx_k + Hx_k - H\hat{x}_k^-)$$
$$= x_k - \hat{x}_k^- - K_k H(x_k - \hat{x}_k^-) - K(z_k - Hx_k)$$
$$= (I - K_k H)(x_k - \hat{x}_k^-) - K_k(z_k - Hx_k)$$
$$= (I - K_k H)e_k^- - K_k \nu_k$$

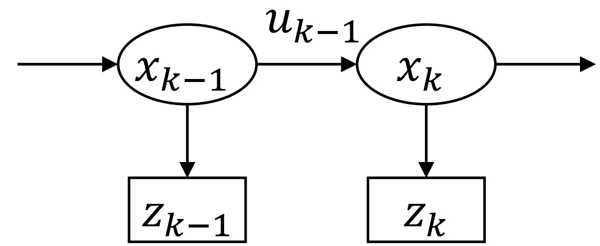
e_k^- and ν_k have zero covariance

$$P_k = E\left[e_k e_k^T\right]$$
$$= (I - K_k H)E\left[e_k^- e_k^{-\mathrm{T}}\right](I - K_k H)^T + E[K_k \nu_k \nu_k^T K_k^T]$$
$$= (I - K_k H)P_K^-(I - K_k H)^T + K_k R K_k^T$$

To minimize the trace of P_k, take $\frac{\partial tr(P_k)}{\partial K_k} = 0$

Yields $K_k = \frac{P_k^- H^T}{HP_k^- H^T + R}$, the optimal Kalman gain

Interpreting the Kalman Gain



Recall the Kalman gain $K_k = \frac{P_k^- H^T}{HP_k^- H^T + R}$ is used for computing $\hat{x}_k$

$$\hat{x}_k = \hat{x}_k^- + K_k(z_k - H\hat{x}_k^-)$$

As $R \to 0$, measurement becomes more accurate, $K_k \to H^{-1}$

$$\hat{x}_k = \hat{x}_k^- + K_k(z_k - H\hat{x}_k^-) \to H^{-1}z_k$$

As $P_k^- \to 0$, state propagation becomes more accurate, $K_k \to 0$

$$\hat{x}_k = \hat{x}_k^- + K_k(z_k - H\hat{x}_k^-) \to \hat{x}_k^-$$

Deriving The Iterative Update Algorithm

Recall - in each step, two updates

Time update: $\hat{x}_k^- = A\hat{x}_{k-1} + Bu_{k-1}$ ⇨Error of this step $e_k^- \equiv x_k - \hat{x}_k^-$ ⇨(a priori) Variance is $P_k^- = E[e_k^- {e_k^-}^{\mathrm{T}}]$	Measurement update: $\hat{x}_k = \hat{x}_k^- + K_k(z_k - H\hat{x}_k^-)$ ⇨Error of this step $e_k \equiv x_k - \hat{x}_k$ ⇨(a posteriori) Variance is $P_k = E[e_k e_k^{\mathrm{T}}]$

For time update $\hat{x}_k^- = A\hat{x}_{k-1} + Bu_{k-1}$

⇨$e_k^- \equiv x_k - \hat{x}_k^- = Ax_{k-1} + Bu_{k-1} + \omega_{k-1} - \hat{x}_k^-$

$= A(x_{k-1} - \hat{x}_{k-1}) + \omega_{k-1}$

$= Ae_{k-1} + \omega_{k-1}$

⇨$P_k^- = E\left[e_k^- {e_k^-}^{\mathrm{T}}\right]$

$= AE\left[e_{k-1}e_{k-1}^T\right]A^T + Q$

$= AP_{k-1}A^T + Q$

e_{k-1} and ω_{k-1} have zero covariance

For measurement update $\hat{x}_k = \hat{x}_k^- + K_k(z_k - H\hat{x}_k^-)$

⇨Already computed

⇨$K_k = \frac{P_k^- H^T}{HP_k^- H^T + R}$

⇨$P_k = (I - K_k H)P_k^-(I - K_k H)^T + K_k R K_k^T = (I - K_k H)P_k^-$

Tuning Parameters and Running the Filter

We have the iterative update algorithm

Time update

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

Measurement update

$$\hat{x}_k = \hat{x}_k^- + K_k(z_k - H\hat{x}_k^-)$$
$$K_k = \frac{P_k^- H^T}{HP_k^- H^T + R}$$
$$P_k = (I - K_k H)P_k^-$$

To run the algorithm

- ⇨First estimate Q and R offline
- ⇨Starting from some estimate and then tuning
- ⇨This is known as **system identification**
- ⇨Then start filter with some initial $\hat{x}_0$ and P_0
- ⇨Usually P_k and K_k will quickly converge

Example: Estimating a Random Constant

Suppose we are measuring a random constant, e.g., temperature of a light bulb

- ⇨ System: $x_k = x_{k-1} + \omega_{k-1}, \quad x_k, \omega_k \in \mathbb{R}$
- ⇨ Observation: $z_k = x_k + \nu_k, \quad z_k, \nu_k \in \mathbb{R}$

Filter update equations

- ⇨ Time update
  - ⇨ $\hat{x}_k^- = \hat{x}_{k-1}$
  - ⇨ $P_k^- = P_{k-1} + Q$
- ⇨ Measurement update
  - ⇨ $\hat{x}_k = \hat{x}_k^- + K_k(z_k - H\hat{x}_k^-)$
  - ⇨ $K_k = P_k^-(P_k^- + R)^{-1}$
  - ⇨ $P_k = (1 - K_k)P_k^-$

Running the example

- ⇨ If both (real, not our estimated) Q and R are large, it's hopeless
- ⇨ If P_0 is small, it trusts x_0 a lot
  - ⇨ If x_0 is bad, then it takes long time to converge with small P_0