Kalman Filtering

What is state estimation?

- Given a “black box” component, we can try to use a linear or nonlinear system to model it (maybe based on physics, or data-driven)
- Model may posit that the plant has internal states, but we typically have access only to the outputs of the model (whatever we can measure using a sensor)
- May need internal states to implement controller: how do we estimate them?
- State estimation: Problem of determining internal states of the plant

Deterministic vs. Noisy case

Typically sensor measurements are noisy (manufacturing imperfections, environment uncertainty, errors introduced in signal processing, etc.)

In the absence of noise, the model is deterministic: for the same input you always get the same output

Can use a simpler form of state estimator called an observer (e.g. a Luenberger observer)

- $\frac{d\hat{\mathbf{x}}}{dt} = A\hat{\mathbf{x}} + B\mathbf{u} + L(\mathbf{y} - \hat{\mathbf{y}})$
- $\hat{\mathbf{y}} = C\hat{\mathbf{x}} + D\mathbf{u}$
- $\mathbf{u}(t) = -K_{\mathrm{lqr}}\hat{\mathbf{x}}(t),$

$\longleftrightarrow$ $\dot{e} = (A - LC)e$

In the presence of noise, we use a state estimator, such as a Kalman Filter

Kalman Filter is one of the most fundamental algorithm that you will see in autonomous systems, robotics, computer graphics, …

Random variables and statistics refresher

- For random variable w, $\mathbb{E}[w]$: expected value of w, also known as mean
- Suppose $\mathbb{E}[x] = \mu$: then $\mathrm{var}(\mathrm{w})$: variance of w, is $\mathbb{E}[(w - \mu)^2]$
- For random variables x and y, $\mathrm{cov}(x, y)$: covariance of x and y
  - $\mathrm{cov}(x, y) = \mathbb{E}[(x - \mathbb{E}(x))(y - \mathbb{E}(y))]$
- For random ***vector*** $\mathbf{x}$, $\mathbb{E}[\mathbf{x}]$is a vector
- For random vectors, $\mathbf{x} \in \mathbb{R}^m$ and $\mathbf{y} \in \mathbb{R}^n$, cross-covariance matrix is $m \times n$ matrix: $\mathrm{cov}(\mathbf{x}, \mathbf{y}) = \mathbb{E}[(\mathbf{x} - \mathbb{E}[\mathbf{x}])(\mathbf{y} - \mathbb{E}[\mathbf{y}])^{\mathrm{T}}]$
- $w \sim N(\mu, \sigma^2)$: w is a normally distributed variable with mean μ and variance σ

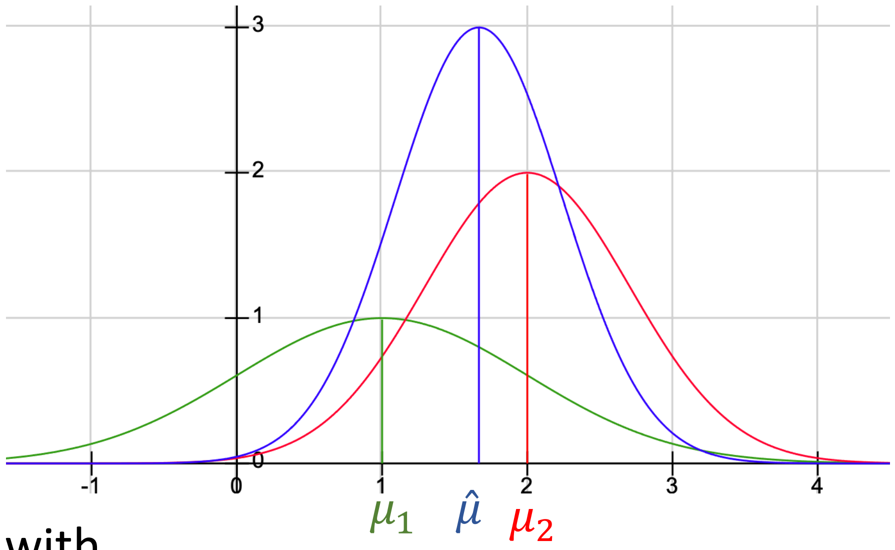
Data fusion example

- Using radar and a camera to estimate the distance to the lead car:
  - Measurement is never free of noise
  - Actual distance: x
  - Measurement with radar: $z_1 = x + v_1$ ($v_1 \sim N(\mu_1, \sigma_1^2)$ is radar noise)
  - With camera: $z_2 = x + v_2$ ($v_2 \sim N(\mu_2, \sigma_2^2)$ is camera noise)
  - How do you combine the two estimates?
- Use a weighted average of the two estimates, prioritize more likely measurement
  - $\hat{\mu} = \frac{(z_1/\sigma_1^2)+(z_2/\sigma_2^2)}{(1/\sigma_1^2)+(1/\sigma_2^2)} = kz_1 + (1-k)z_2$, where $k = \frac{\sigma_2^2}{\sigma_1^2+\sigma_2^2}$
  - $\hat{\sigma}^2 = \frac{\sigma_1^2\sigma_2^2}{\sigma_1^2+\sigma_2^2}$
- Observe: uncertainty reduced, and mean is closer to measurement with lower uncertainty

$\mu_1 = 1, \sigma_1^2 = 1$

$\mu_2 = 2, \sigma_2^2 = 0.5$

$\hat{\mu} = 1.67, \sigma_2^2 = 0.33$



Multi-variate sensor fusion

- Instead of estimating one quantity, we want to estimate n quantities, then:
- Actual value is some vector $\mathbf{x}$
- Measurement noise for i^{th} sensor is $v_i \sim N(\boldsymbol{\mu}_i, \Sigma_i)$, where $\boldsymbol{\mu}_i$ is the mean vector, and Σ_i is the covariance matrix
- $\Lambda = \Sigma^{-1}$ is the **information matrix**
- For the two-sensor case:
  - $\hat{\mathbf{x}} = (\Lambda_1 + \Lambda_2)^{-1}(\Lambda_1 \mathbf{z}_1 + \Lambda_2 \mathbf{z}_2)$, and $\hat{\Sigma} = (\Lambda_1 + \Lambda_2)^{-1}$

Motion makes things interesting

- What if we have one sensor and making repeated measurements of a moving object?
- Measurement differences are not all because of sensor noise, some of it is because of object motion
- Kalman filter is a tool that can include a motion model (or in general a dynamical model) to account for changes in internal state of the system
- Combines idea of ***prediction*** using the system dynamics with ***correction*** using weighted average (Bayesian inference)

Stochastic Difference Equation Models

- We assume that the plant (whose state we are trying to estimate) is a stochastic discrete dynamical process with the following dynamics:

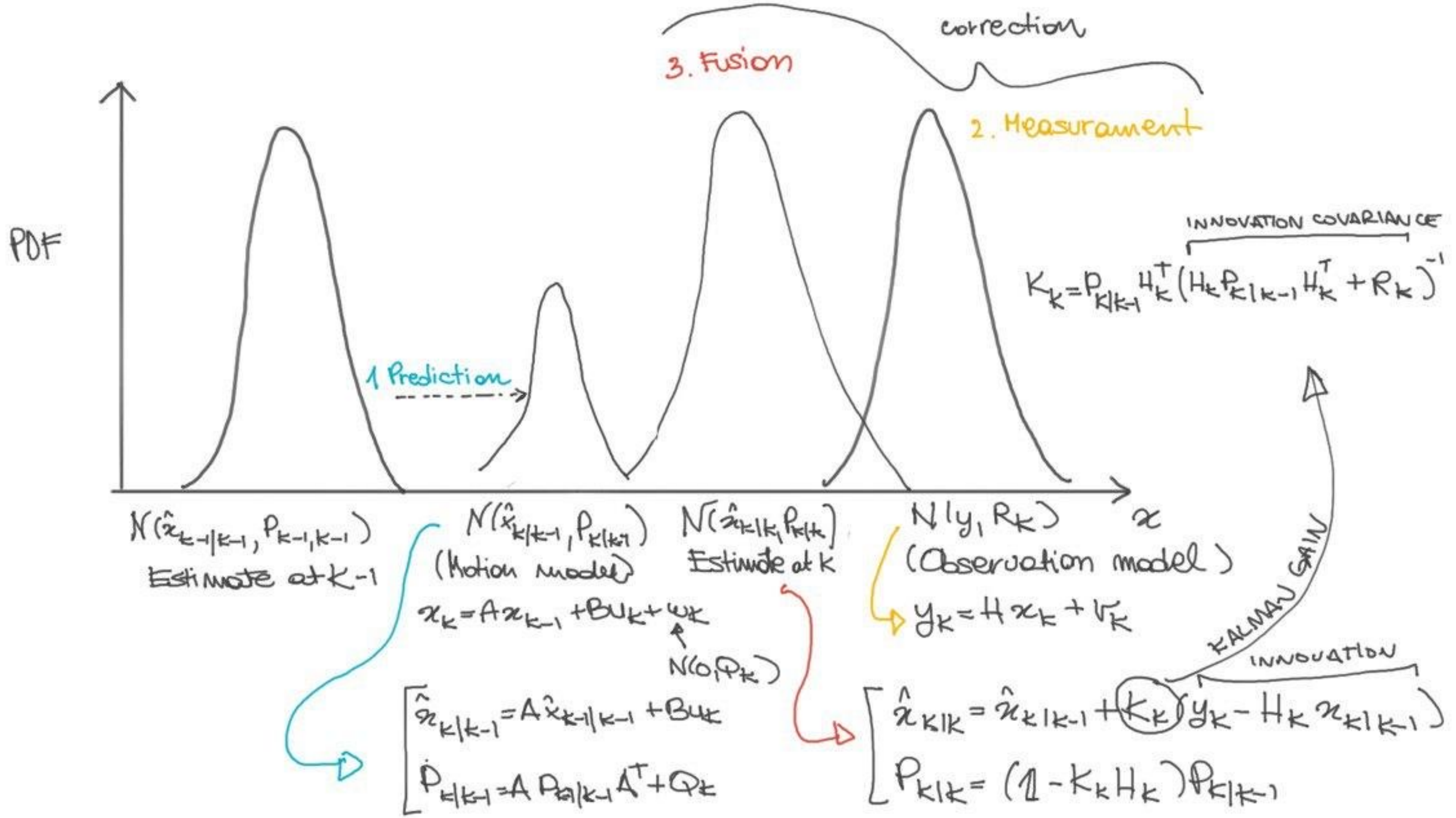
$$\mathbf{x}_k = A\mathbf{x}_{k-1} + B\mathbf{u}_k + \mathbf{w}_k \text{ (Process Model)}$$

$$\boldsymbol{y}_k = H\mathbf{x}_k + \mathbf{v}_k \text{ (Measurement Model)}$$

$\mathbf{x}_k, \mathbf{x}_{k-1}$	State at time k,$k-1$
$\mathbf{u}_k$	Input at time k
$\mathbf{w}_k$	Random vector representing noise in the plant, $\mathbf{w} \sim N(\mathbf{0}, Q_k)$
$\mathbf{v}_k$	Random vector representing sensor noise, $\mathbf{v} \sim N(\mathbf{0}, R_k)$
$\mathbf{z}_k$	Output at time k

n	Number of states
m	Number of inputs
p	Number of outputs
A	$n \times n$ matrix
B	$n \times m$ matrix
H	$p \times n$ matrix

Kalman Filter



Step I: Prediction

- We assume an estimate of $\mathbf{x}$ at time $k-1$, fusing information obtained by measurements till time $k-1$: this is denoted $\hat{\mathbf{x}}_{k-1|k-1}$
- We also assume that the error between the estimate $\hat{\mathbf{x}}_{k-1|k-1}$ and the actual $\mathbf{x}_{k-1}$ has 0 mean, and covariance $P_{k-1|k-1}$
- Now, we use these values and the state dynamics to predict the value of $\mathbf{x}_k$
- Because we are using measurements only up to time $k-1$, we can denote this predicted value as $\hat{\mathbf{x}}_{k|k-1}$, and compute it as follows:

$$\hat{\mathbf{x}}_{k|k-1} \coloneqq A\hat{\mathbf{x}}_{k-1|k-1} + B\mathbf{u}_k$$

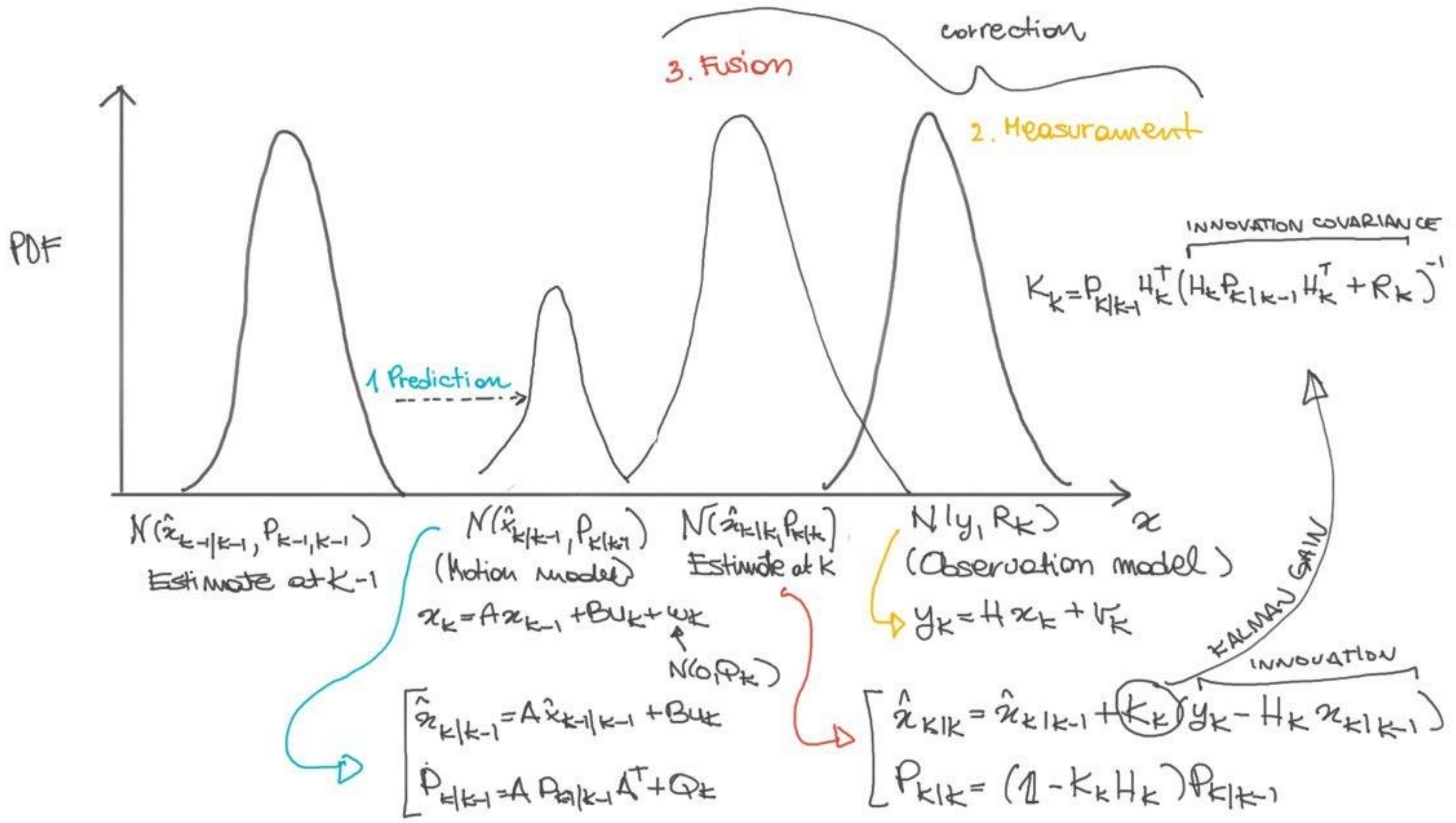
Step I: Prediction

$$
\begin{aligned}
P_{k|k-1} &= \mathrm{cov}\left(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k-1}\right) = \mathrm{cov}\left(A\mathbf{x}_{k-1} + B\mathbf{u}_k + w_k - A\hat{\mathbf{x}}_{k-1|k-1} - B\mathbf{u}_k\right) \\
&= A\mathrm{cov}\left(\mathbf{x}_{k-1} - \hat{\mathbf{x}}_{k-1|k-1}\right)A^T + cov(w_k) \\
&= AP_{k-1|k-1}A^T + Q_k
\end{aligned}
$$

- Thus, the state and error covariance prediction are:

$$
\begin{aligned}
\hat{\mathbf{x}}_{k|k-1} &:= A\hat{\mathbf{x}}_{k-1|k-1} + B\mathbf{u}_k \\
P_{k|k-1} &:= AP_{k-1|k-1}A^T + Q_k
\end{aligned}
$$

Kalman Filter



Step II: Correction

- This is where we basically do data fusion between new measurement and old prediction to obtain new estimate
- Note that data fusion is not straightforward like before because we don't really observe/measure $\mathbf{x}_k$ directly, but we get measurement $\boldsymbol{y}_k$, for an observable output!
- Idea remains similar: Do a weighted average of the prediction $\hat{\mathbf{x}}_{k|k-1}$ and new information
- We integrate new information by using the difference between the predicted output and the observation

Step II: Correction

- Predicted output: $\hat{\boldsymbol{y}}_k = H_k \hat{\mathbf{x}}_{k|k-1}$
- We denote the error in predicted output as the *innovation*

$$\mathbf{z}_k := \mathbf{y}_k - H_k \hat{\mathbf{x}}_{k|k-1}$$

- Covariance of innovation

$$S_k = \mathrm{cov}\,(\mathbf{z}_k) = cov(H_k \mathbf{x}_k + \mathbf{v}_k - H_k \hat{\mathbf{x}}_{k|k-1}) = R_k + H_k P_{k|k-1} H_k^T$$

- Then to do data fusion is given by:

$$\hat{\boldsymbol{x}}_{k|k} := \hat{\boldsymbol{x}}_{k|k-1} + K_k z_k$$

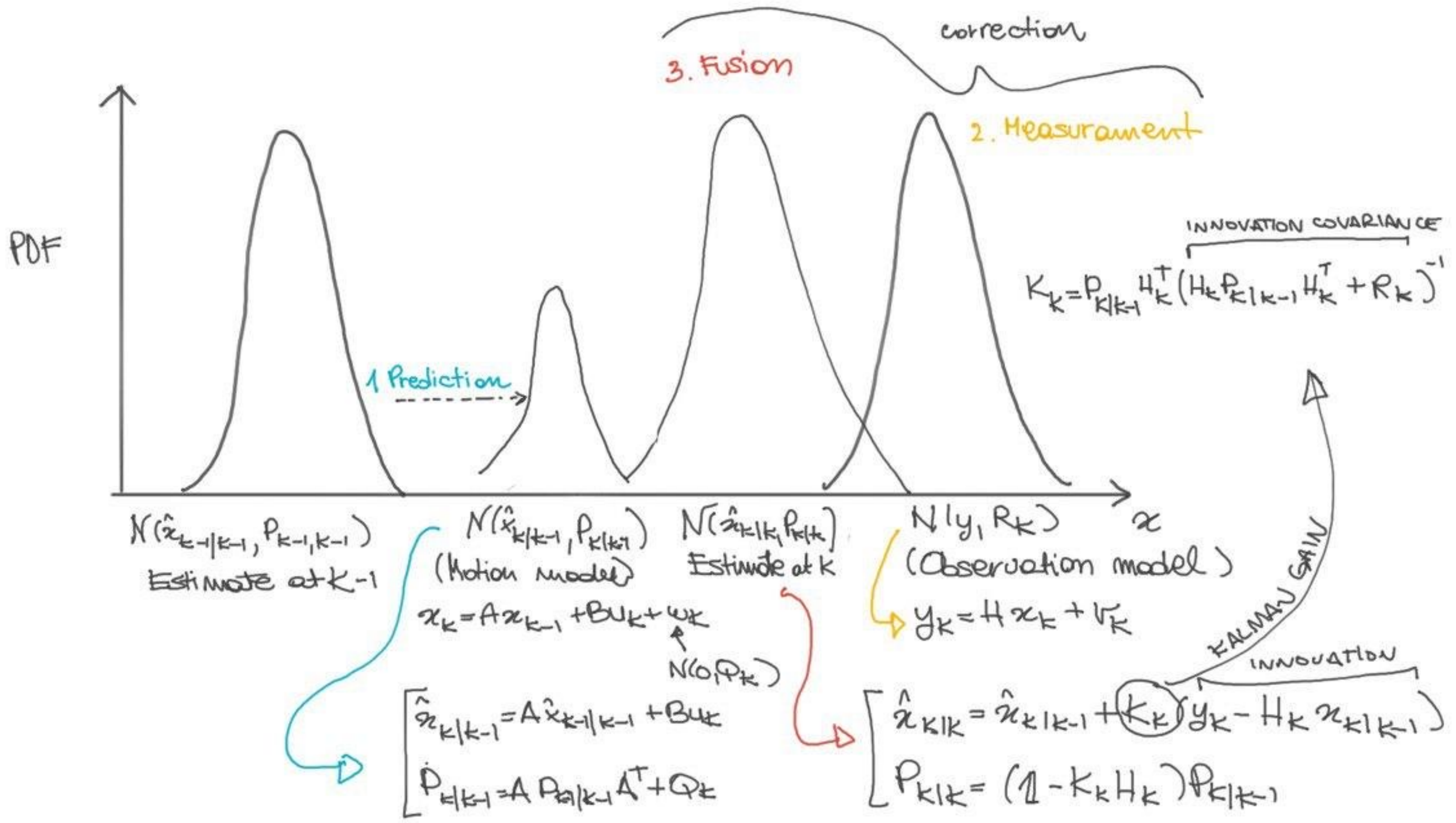
- Where, $K_k = P_{k|k-1} H_k^T S_k^{-1}$ is the (optimal) Kalman gain. It minimizes the least square error
- Finally, the updated error covariance estimate is given by:

$$P_{k|k} := (I - K_k H_k)\, P_{k|k-1}$$

Step II: Correction

Innovation	$\mathbf{z}_k := \mathbf{y}_k - H_k \hat{\mathbf{x}}_{k\|k-1}$
Innovation Covariance	$S_k := R_k + H_k P_{k\|k-1} H_k^T$
Optimal Kalman Gain	$K_k := P_{k\|k-1} H_k^T S_k^{-1}$
State estimate at time k	$\hat{\boldsymbol{x}}_{k\|k} := \hat{\boldsymbol{x}}_{k\|k-1} + K_k \mathbf{z}_k$
Covariance estimate at time k	$P_{k\|k} := P_{k\|k-1}(I - K_k H_k)$

Kalman Filter



one-dimensional example

- Let's take a simple one-dimensional example
- Kalman filter prediction equations become:
  - $\hat{x}_{k|k-1} := a\hat{x}_{k-1|k-1} + bu$; $\sigma^2_{k|k-1} := \underbrace{a^2\sigma^2_{k-1|k-1}}_{\text{prior uncertainty in estimate}} + \underbrace{\sigma^2_q}_{\text{uncertainty in process}}$
- Also, the correction equations become:
  - Innovation: $z_k := y_k - \hat{x}_{k|k-1}$, $\mathrm{S_k} = \sigma^2_r + \sigma^2_{k|k-1}$
  - Optimal gain: $k = \sigma^2_{k|k-1}/(\sigma^2_r + \sigma^2_{k|k-1})$,
  - Updated state estimate: $\hat{x}_{k|k} := \hat{x}_{k|k-1} + k(y_k - \hat{x}_{k|k-1})$
  - I.e. updated state estimate: $\hat{x}_{k|k} := (1-k)\ \hat{x}_{k|k-1} + ky_k$ (Weighted average!)

Extended Kalman Filter

- We skipped derivations of equations of the Kalman filter, but a fundamental property assumed is that the process model and measurement model are both linear.
- Under linear models and Gaussian process/measurement noise, a Kalman filter is an *optimal* state estimator (minimizes mean square error between estimate and actual state)
- In an EKF, state transitions and observations need not be linear functions of the state, but can be any differentiable functions
- I.e., the process and measurement models are as follows:

$$\mathbf{x}_k = f(x_{k-1}, u_k) + w_k$$
$$y_k = h(x_k) + v_k$$

EKF updates

- Functions f and h can be used directly to compute state-prediction, and predicted measurement, but cannot be directly used to update covariances
- So, we instead use the Jacobian of the dynamics at the predicted state
- This linearizes the non-linear dynamics around the current estimate
- Prediction updates:

$$\hat{\mathbf{x}}_{k|k-1} := f(\hat{\mathbf{x}}_{k-1|k-1}, \mathbf{u}_k)$$
$$P_{k|k-1} := F_k P_{k-1|k-1} F_k^T + Q_k$$

$$F_k := \left. \frac{\partial f}{\partial \mathbf{x}} \right|_{\mathbf{x}=\hat{\mathbf{x}}_{k|k-1}, \mathbf{u}=\mathbf{u}_k}$$

EKF updates

- Correction updates:

$$H_k := \left.\frac{\partial h}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}_{k|k-1}}$$

Innovation $\mathbf{z}_k := \mathbf{y}_k - h(\hat{\mathbf{x}}_{k|k-1})$

Innovation Covariance $S_k := R_k + H_k P_{k|k-1} H_k^T$

Near-Optimal Kalman Gain $K_k := P_{k|k-1} H_k^T S_k^{-1}$

A posteriori state estimate $\hat{\boldsymbol{x}}_{k|k} := \hat{\boldsymbol{x}}_{k|k-1} + K_k \mathbf{y}_k$

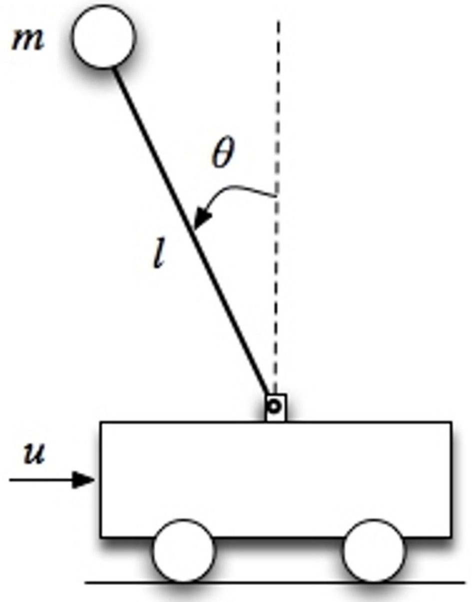
A posteriori error covariance estimate $P_{k|k} := P_{k|k-1}(I - K_k H_k)$

Simulink Example - Cartpole

$$\begin{cases} \ddot{p} & = \frac{u+m\,l\,\dot{\theta}^2\;\sin\theta - m\,g\;\cos\theta\sin\theta}{M+m\sin\theta^2} \\ \ddot{\theta} & = \frac{g\;\sin\theta - \cos\theta\ddot{p}}{l} \end{cases}$$

$$x = \left[p,\ \dot{p},\ \theta,\ \dot{\theta}\right]^T$$

$$y = [p, \theta]$$



- Full-state estimation (Luenberger, Kalman)
- Optimal Control