DEPARTMENT OF COMPUTER SCIENCE
UNIVERSITY OF COPENHAGEN

Sequential Inference by Filtering: Kalman filtering

Kim Steenstrup Pedersen

Plan for today

- The sequential inference / estimation problem solved with Bayesian filtering
- Kalman filtering

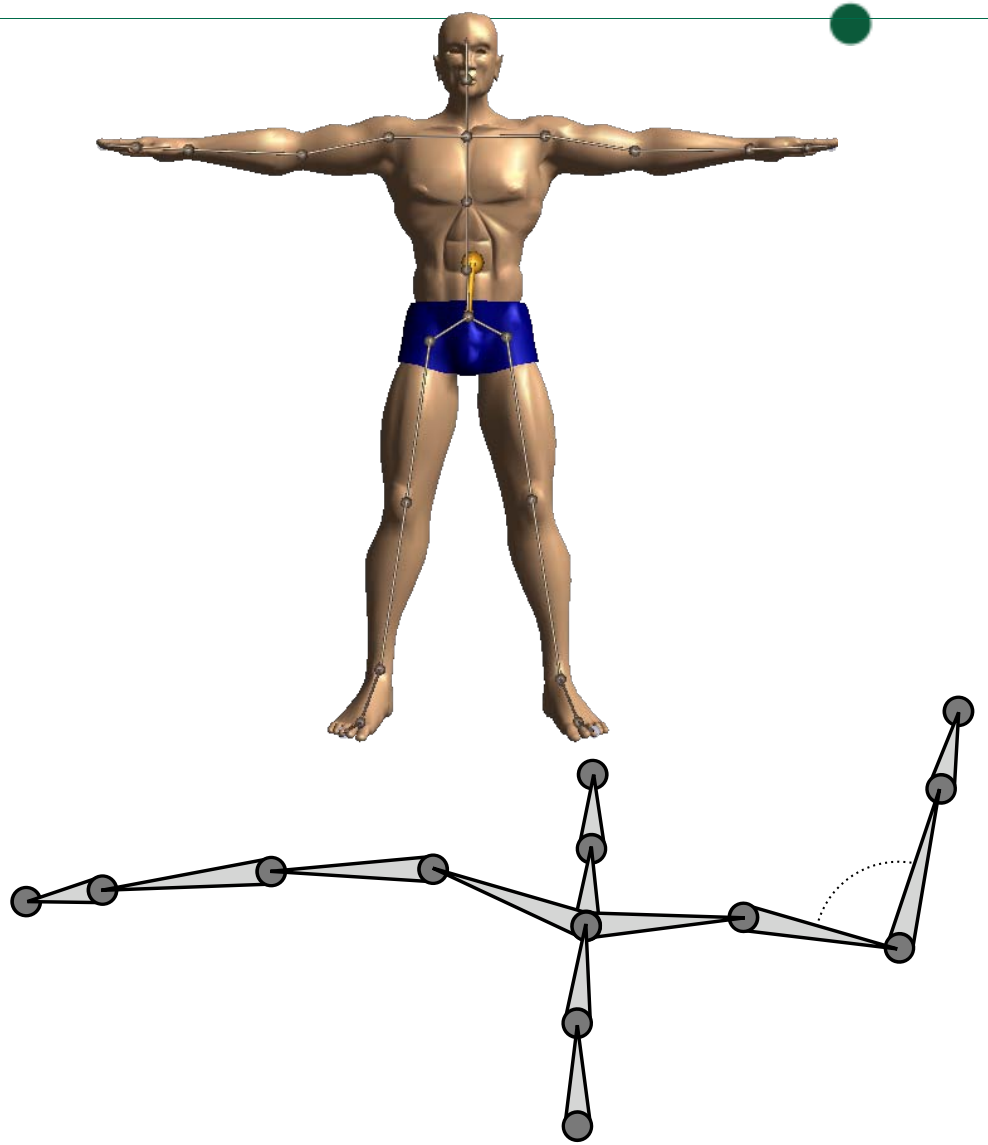
Next time:

- Particle filtering - a sequential Monte Carlo method

- COMMENT: I will use human motion tracking as motivating example, but remember these are general methods.

Visual human motion modelling and tracking

Visual human motion modelling and tracking
(Tracking = estimation of model from observations)



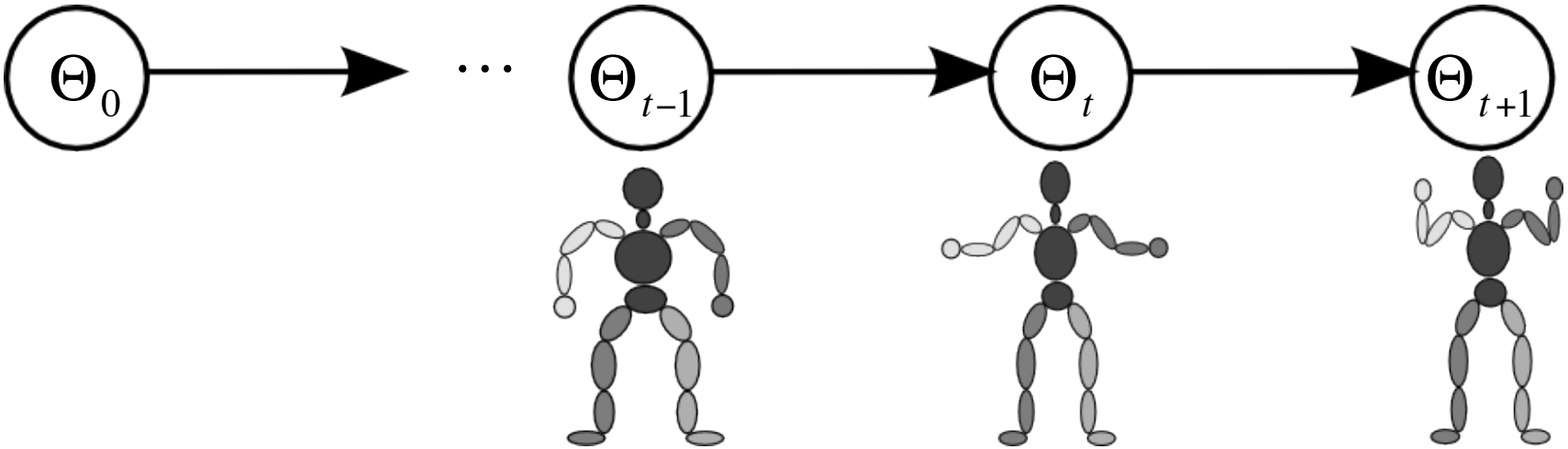
- **Model:** Stick figure representation of human body (rigid sticks connected by joints, no mass or volume).
  - Model parameters a.k.a. the state): Joint position vector

$$\Theta = \left[\theta_1, \ldots, \theta_D\right]^T$$

- **Observations:** Video sequence of motion

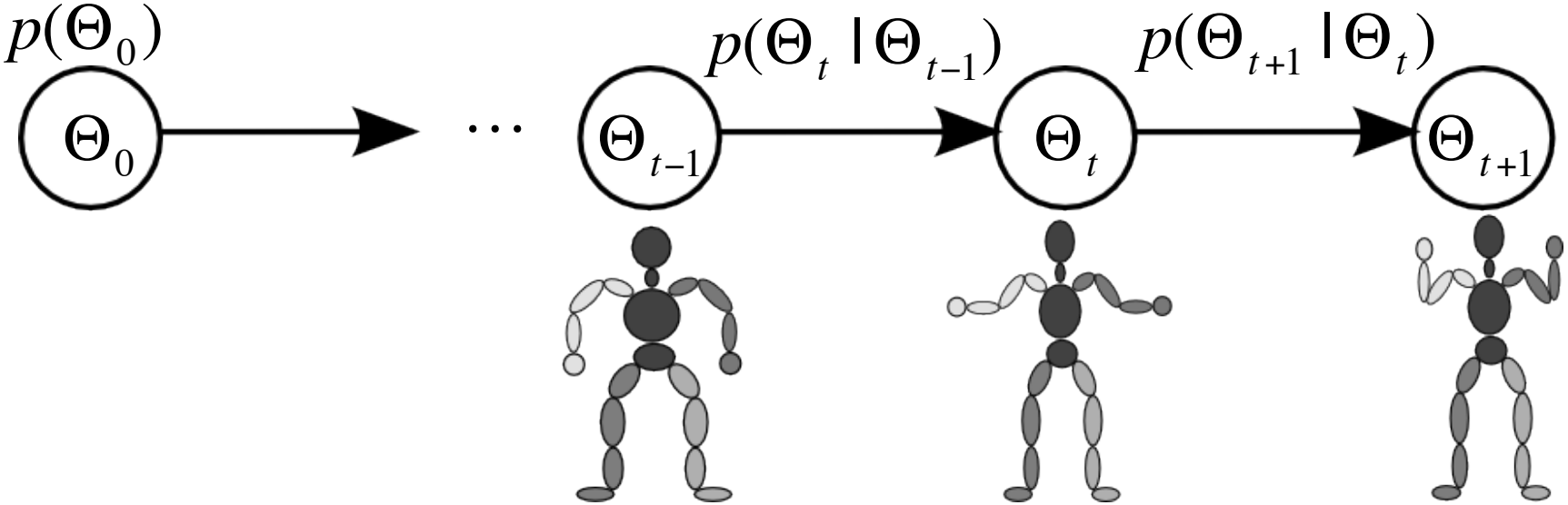
$$I_{1:t} = \left\{I_1, I_2, \ldots, I_t\right\}$$

How do we include dynamics into our model? We will use a first order Markov chain model



- Lets assume that the current state only depends on the immediate past state. We assume that somehow we can compute the new state given the old state!
- However this update of states is stochastic (uncertain) – we are going to estimate it from noisy observations.

First order Markov chains (discrete time)



- If we know the transition conditional probability distributions and the prior distribution on the initial state, we can compute the probability distribution of state sequences (of particular sequences of stick figures):

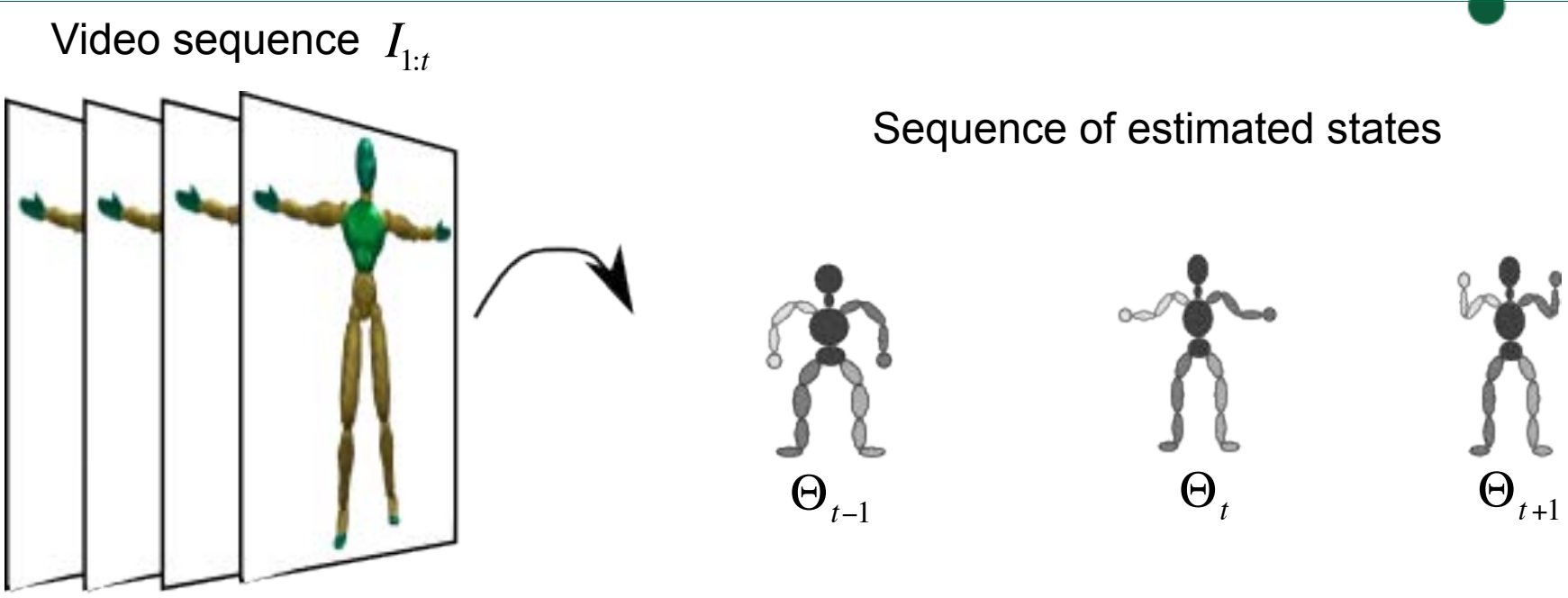
$$p(\Theta_0,\ldots,\Theta_t) = p(\Theta_{0:t}) = p(\Theta_0)\prod_{i=1}^{t} p(\Theta_i \mid \Theta_{i-1})$$

Short-hand notation:

$$\Theta_{0:t} = \{\Theta_0,\ldots,\Theta_t\}$$

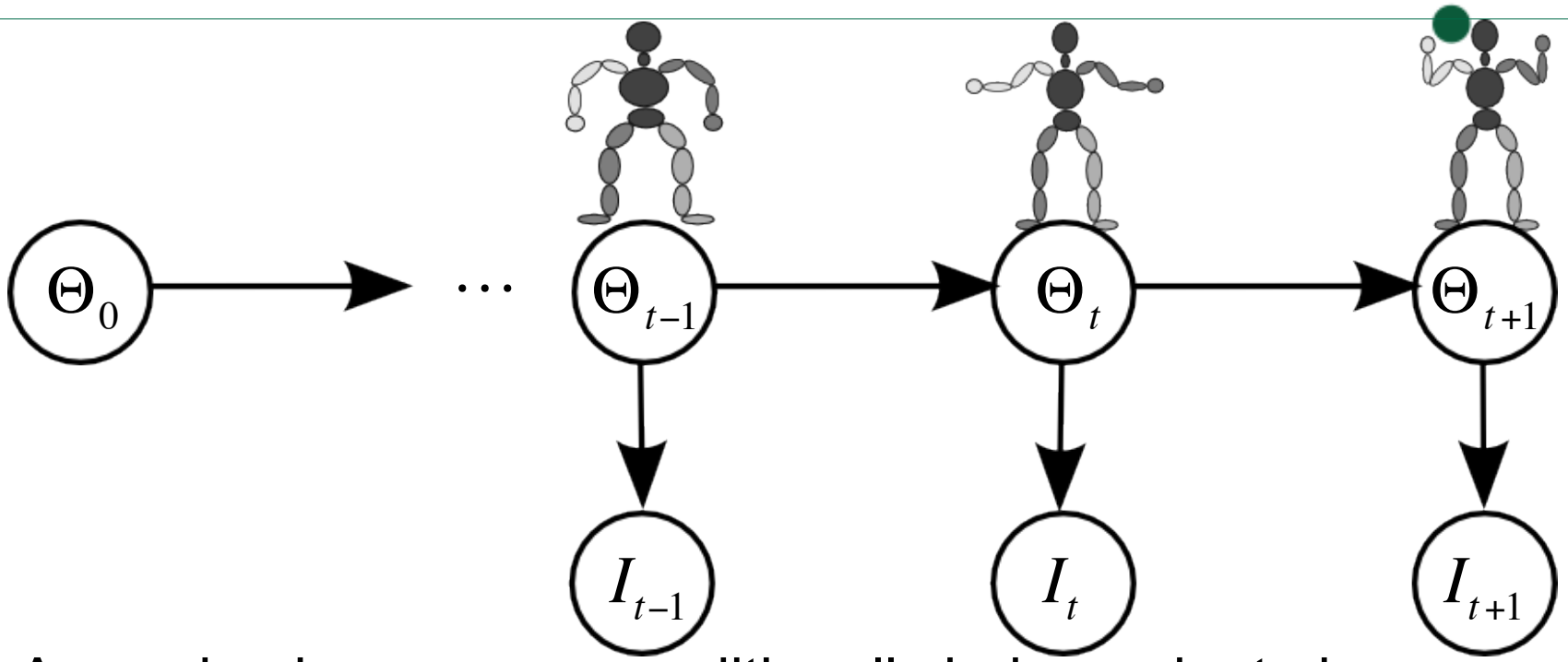
How to relate the model state with observations?

(Tracking = estimation of model from observations)



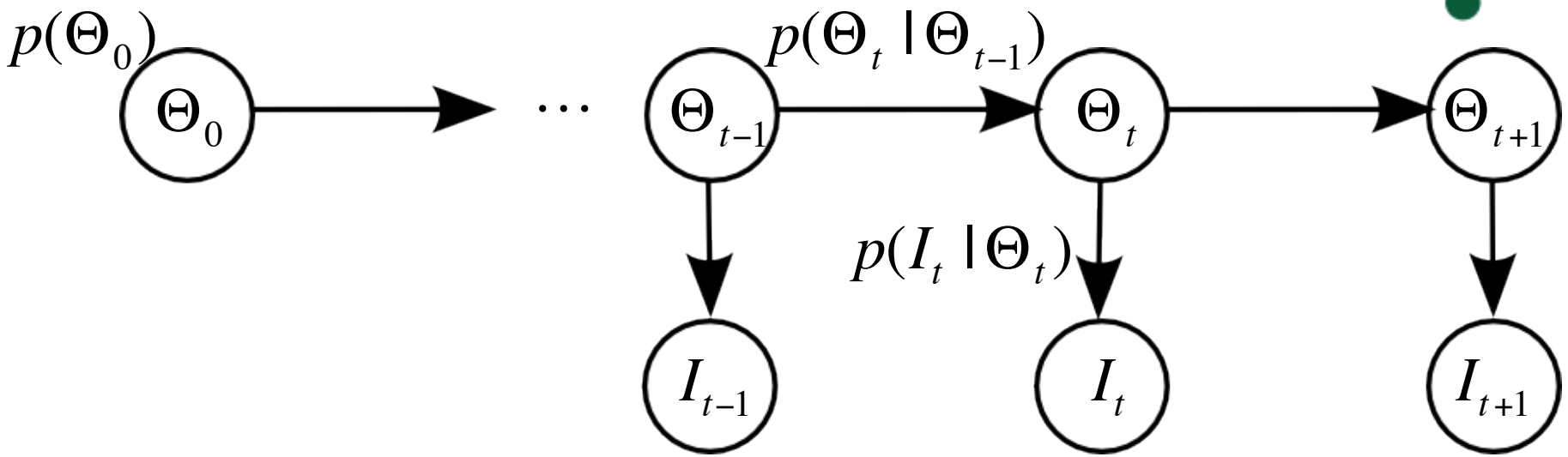
However our observations are noisy so we want a probabilistic model for this!

Adding observations to the model



- Assuming images are conditionally independent given state.
- This model states that we only observe the images directly and the states indirectly – they are hidden (latent variables).

We need a probabilistic observation model



How about a Gaussian observation model?

$$p(I_t \mid \Theta_t) = \frac{1}{Z}\exp\left(-\frac{\left\|I_t - F(\Theta_t)\right\|^2}{2\sigma^2}\right)$$

Or perhaps more useful, an exponential distribution

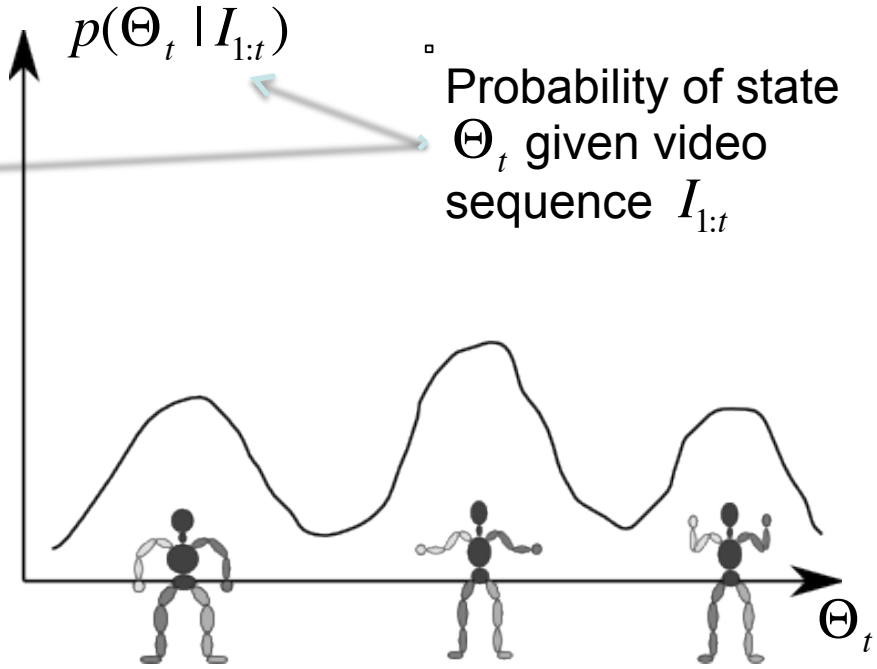
$$p(I_t \mid \Theta_t) = \frac{1}{Z}\exp\left(-H(I_t, \Theta_t)\right)$$

What about tracking?

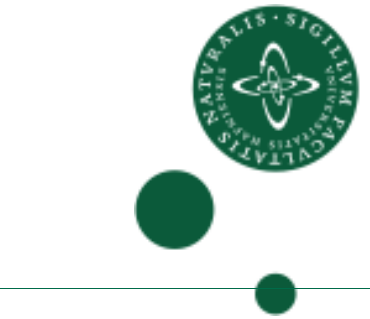
- The model gives us the joint distribution

$$p(I_{1:t}, \Theta_{0:t}) = p(\Theta_0) \prod_{i=1}^{t} p(I_i \mid \Theta_i) p(\Theta_i \mid \Theta_{i-1})$$

- If we want to do real-time tracking we need $p(\Theta_t \mid I_{1:t})$
- Just apply the sum and product rule to $p(I_{1:t}, \Theta_{0:t})$
- And then take averages to compute a prediction of the current state.

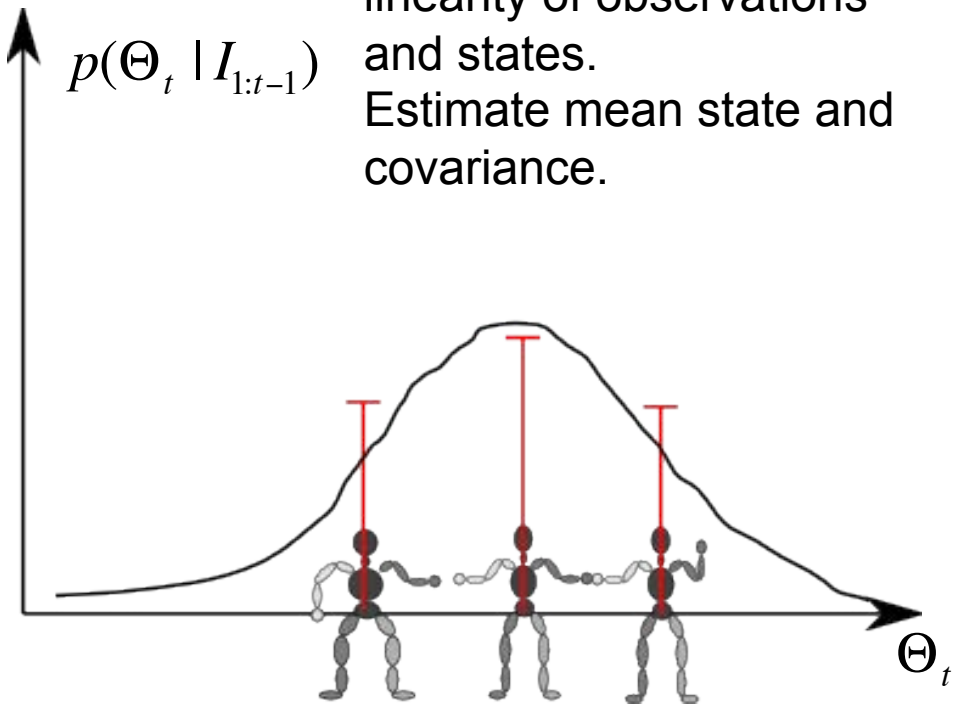


We need to sequentially estimate $p(\Theta_t \mid I_{1:t})$



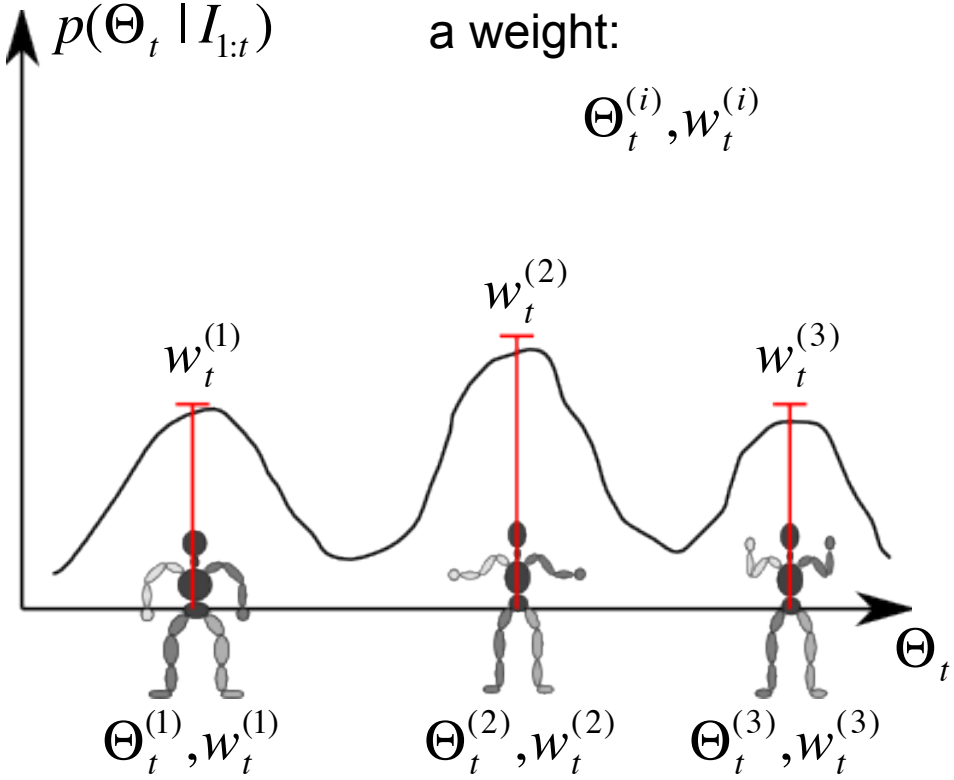
Kalman filtering

- Assume Gaussianity and linearity of observations and states.
  Estimate mean state and covariance.



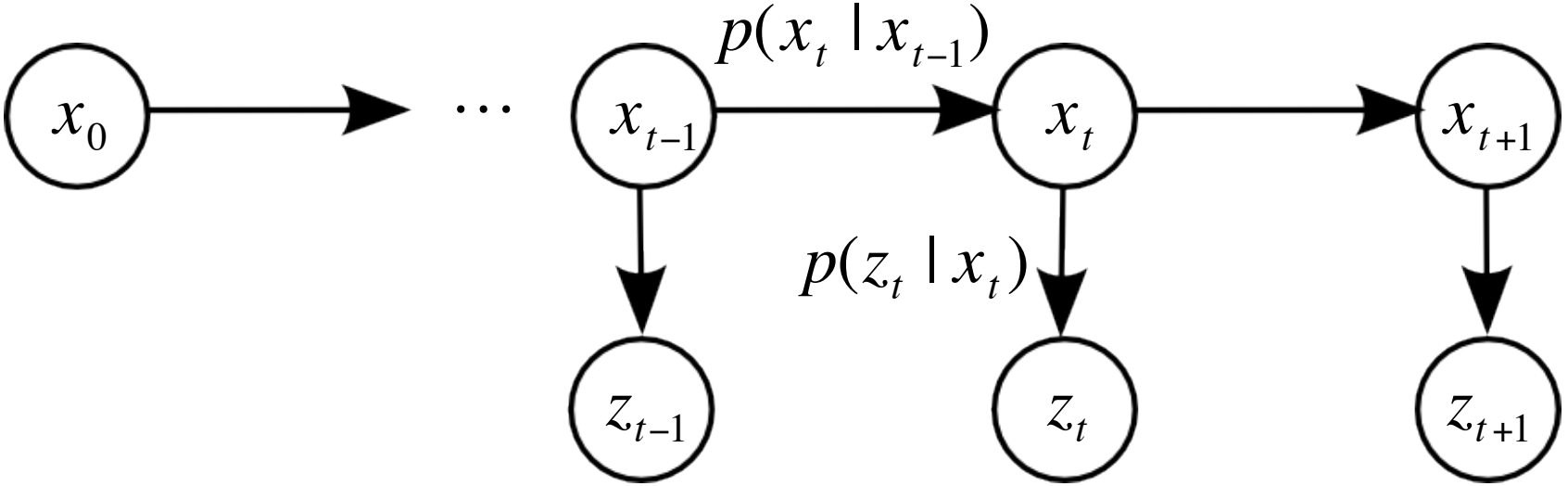
Particle filtering

- Particles: A state and a weight:
  $\Theta_t^{(i)}, w_t^{(i)}$



Sequential estimation by Bayesian filtering (Lets generalize the notation a bit)

- Given observations $z_t \in R^m$ infer the hidden state $x_t \in R^n$ where *t* denotes discrete time.
- x_t and z_t are random variables (i.e. are uncertain / noisy).
- Lets assume that the states x_t form a 1st order Markov chain and the observations z_t are independent conditioned on x_t.



Sequential estimation by Bayesian filtering

Bayesian filtering are governed by two sets of equations:

- Dynamical model:

$$x_t = f(x_{t-1}) + w_t$$ (Stochastic difference equation)

$$x_t \sim p(x_t \mid x_{t-1})$$ (Distribution of the state)

- Observation model:

$$z_t = g(x_t) + v_t$$

$$z_t \sim p(z_t \mid x_t)$$ (Distribution of observations given state)

We also need to know the distribution on the initial state $p(x_0)$

Bayesian filtering

- Let $x_{0:t} \equiv \{x_0, \ldots, x_t\}$ and $z_{1:t} \equiv \{z_1, \ldots, z_t\}$
- We want to recursively estimate either the:
  - Posterior distribution

$$p(x_{0:t} \mid z_{1:t})$$

  - Filtering distribution (marginal of posterior)

$$p(x_t \mid z_{1:t}) = \int p(x_{0:t} \mid z_{1:t}) dx_{0:t-1}$$

  - Expectation with respect to posterior

$$E_{p(x_{0:t}|z_{1:t})}[h_t(x_{0:t})] = \int h_t(x_{0:t}) p(x_{0:t} \mid z_{1:t}) dx_{0:t}$$

  - Expectation with respect to filtering distribution, etc.

Bayesian filtering: Recursive posterior

- Let $x_{0:t} \equiv \{x_0, \ldots, x_t\}$ and $z_{1:t} \equiv \{z_1, \ldots, z_t\}$
- Bayes' theorem gives the posterior

$$p(x_{0:t} \mid z_{1:t}) = \frac{p(z_{1:t} \mid x_{0:t}) p(x_{0:t})}{\int p(z_{1:t} \mid x_{0:t}) p(x_{0:t}) dx_{0:t}}$$

$$\text{Posterior} = \frac{\text{Likelihood} \times \text{Prior}}{\text{Evidence}}$$

- Recursive estimate of posterior

$$p(x_{0:t+1} \mid z_{1:t+1}) = p(x_{0:t} \mid z_{1:t}) \frac{p(z_{t+1} \mid x_{t+1}) p(x_{t+1} \mid x_t)}{p(z_{t+1})}$$

Bayesian filtering: Recursive marginal

- Relation between posterior and filtering distribution:

$$p(x_t \mid z_{1:t}) = \int p(x_{0:t} \mid z_{1:t}) dx_{0:t-1}$$

General filtering steps:

- Prediction:

$$p(x_t \mid z_{1:t-1}) = \int \underbrace{p(x_t \mid x_{t-1}) p(x_{t-1} \mid z_{1:t-1})}_{= p(x_t, x_{t-1} \mid z_{1:t-1})} dx_{t-1}$$

- Correction (update):

$$p(x_t \mid z_{1:t}) = \frac{p(z_t \mid x_t) p(x_t \mid z_{1:t-1})}{\int p(z_t \mid x_t) p(x_t \mid z_{1:t-1}) dx_t}$$

So what's the problem?

- In general we cannot evaluate the posterior distribution (and the filtering distribution) in closed form!

- Only in the case of:
  - Discrete hidden Markov models (discrete state space).
  - If the dynamical model is linear and the noises are Gaussian we may apply the Kalman filter.

- In all other cases, we need to introduce some form of approximation like the particle filter or extended Kalman filter.

The Kalman filter

(Was developed for tracking rockets for NASA for the 1960's Apollo missions.)

- Dynamical model: Can be ignored!
  $x_t = A_t x_{t-1} + B_t u_t + w_t$
- Observation model:
  $z_t = H_t x_t + v_t$
- $x \in R^n$, $z \in R^m$, $A_t \in R^{n \times n}$ and $H_t \in R^{m \times n}$
- $u_t \in R^l$ denotes control parameters and $B_t \in R^{n \times l}$
- Both $w_t \in R^n$ and $v_t \in R^m$ are Gaussian distributed noises: $w_t \sim N(w_t | 0, Q_t)$ and $v_t \sim N(v_t | 0, R_t)$

The Kalman filter

- From the Gaussianity of the noises it follows that the transition probability is

$$p(x_t \mid x_{t-1}) = N(x_t \mid A_t x_{t-1} + B_t u_t, Q_t)$$

- and the observation probability is

$$p(z_t \mid x_t) = N(z_t \mid H_t x_t, R_t)$$

- The filtering distribution is also Gaussian

$$p(x_t \mid z_{1:t}) = N(x_t \mid x'_t, P_t)$$

- Hence we want to track the mean and covariance of the filtering distribution!

The Kalman filter intuition

- The dynamical model makes a prediction of the new state with Gaussian uncertainty.
- New variance is the sum of prediction error variance and prior variance (or convolution of two Gaussians).
- As a result, our uncertainty (the variance) will grow.
- When we include an observation our uncertainty (the variance) of the state should decrease and our state estimate should improve.
- The new state estimate should weigh our uncertainty in the measurement against the uncertainty in our prediction.

Example: 1D position tracking

- Dynamical model:
  $x_t = x_{t-1} + 2 + w_t \quad , \quad w_t \sim N(w_t \mid 0, 0.5^2)$
- Observation model:
  $z_t = x_t + v_t \quad , \; v_t \sim N(v_t \mid 0, 0.5^2)$

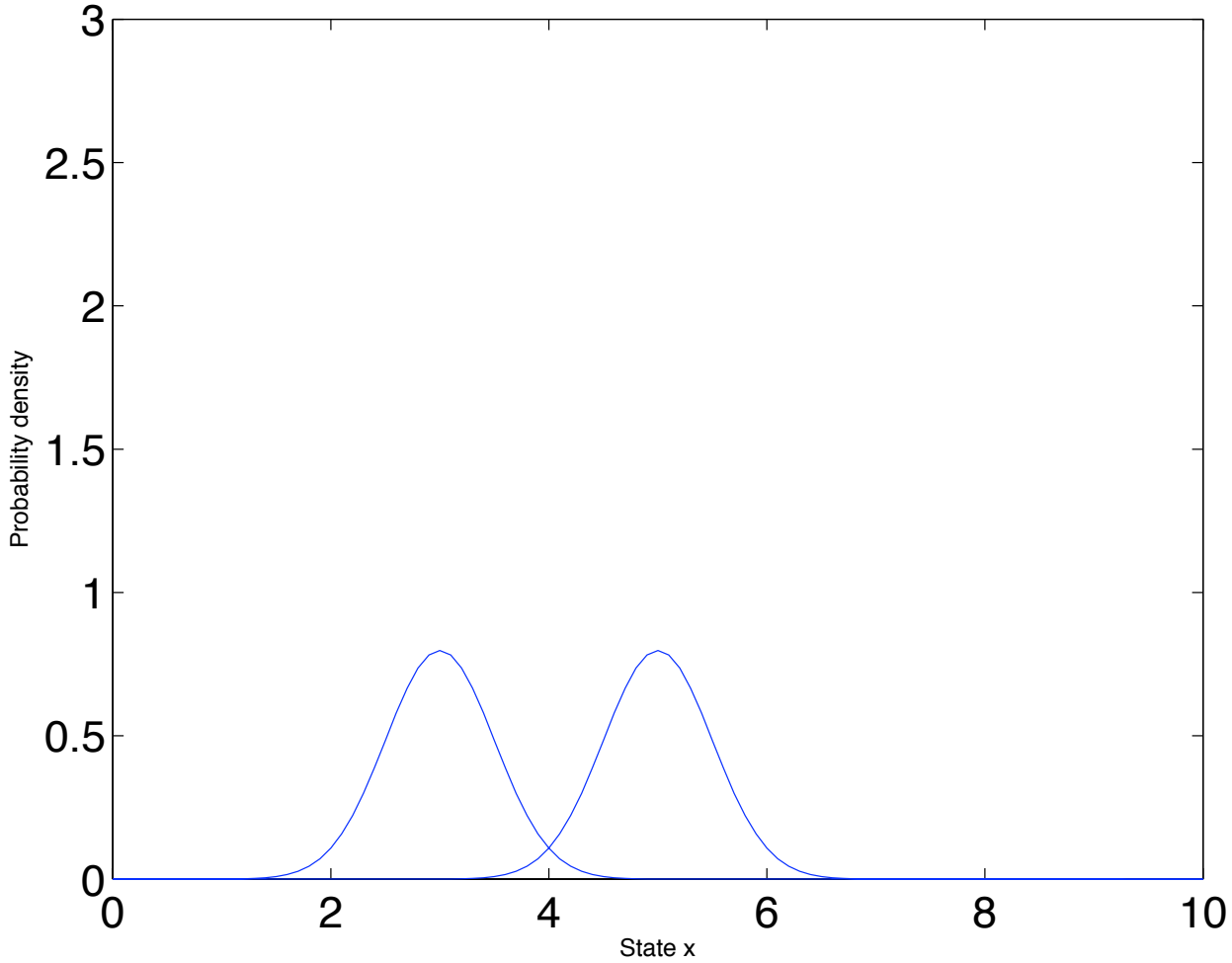
$$p(x_t \mid x_{t-1}) = N(x_t \mid x_{t-1} + 2, 0.5^2) \quad , \; p(z_t \mid x_t) = N(z_t \mid x_t, 0.5^2)$$

- Current state mean and uncertainty (variance):
  $x_{t-1} = 3 \quad , \quad P_{t-1} = 0.5^2$

  $p(x_{t-1} \mid z_{1:t-1}) = N(x_{t-1} \mid 3, 0.5^2)$

- Lets make a prediction of the new state:
  We use the dynamical model!

Example: 1D position tracking
Prediction with the dynamical model

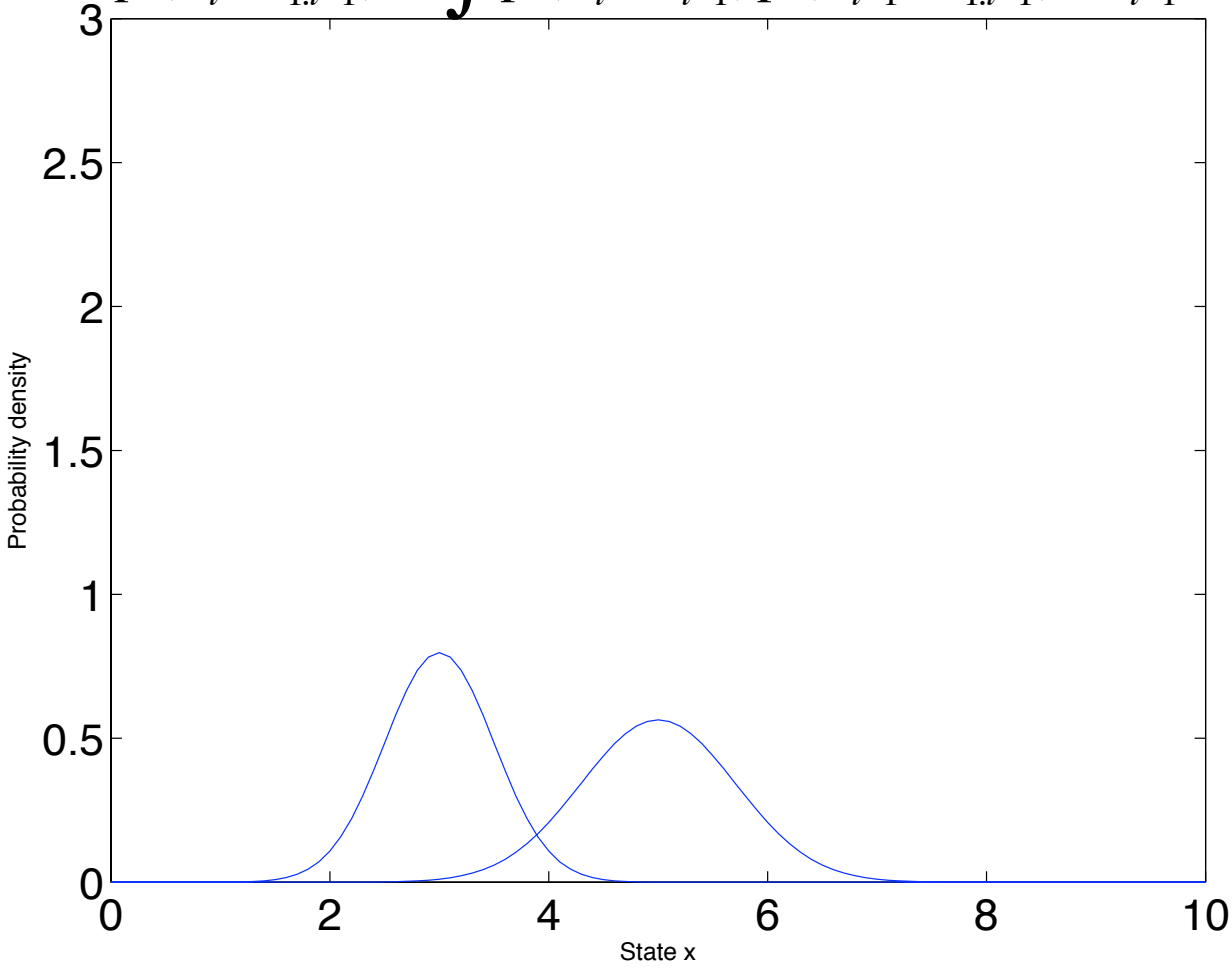


Prediction: $\hat{x}_t = x_{t-1} + 2 \quad , \quad x_{t-1} \sim N(x_{t-1} | 3, 0.5^2)$

Example: 1D position tracking

Include prediction noise:

$$p(x_t \mid z_{1:t-1}) = \int p(x_t \mid x_{t-1}) p(x_{t-1} \mid z_{1:t-1}) dx_{t-1}$$



$$\hat{P}_t = 0.7^2$$

Include prediction noise $\hat{x}_t \sim N(\hat{x}_t \mid 5, 0.5^2 + 0.5^2) = N(\hat{x}_t \mid 5, 0.7^2)$

Example: 1D position tracking

- How can we use observations to improve our estimate?
- Lets make the posterior prediction such that, we correct the prediction with the residual prediction error:

$$x_t = \hat{x}_t + K_t(z_t - H_t\hat{x}_t)$$

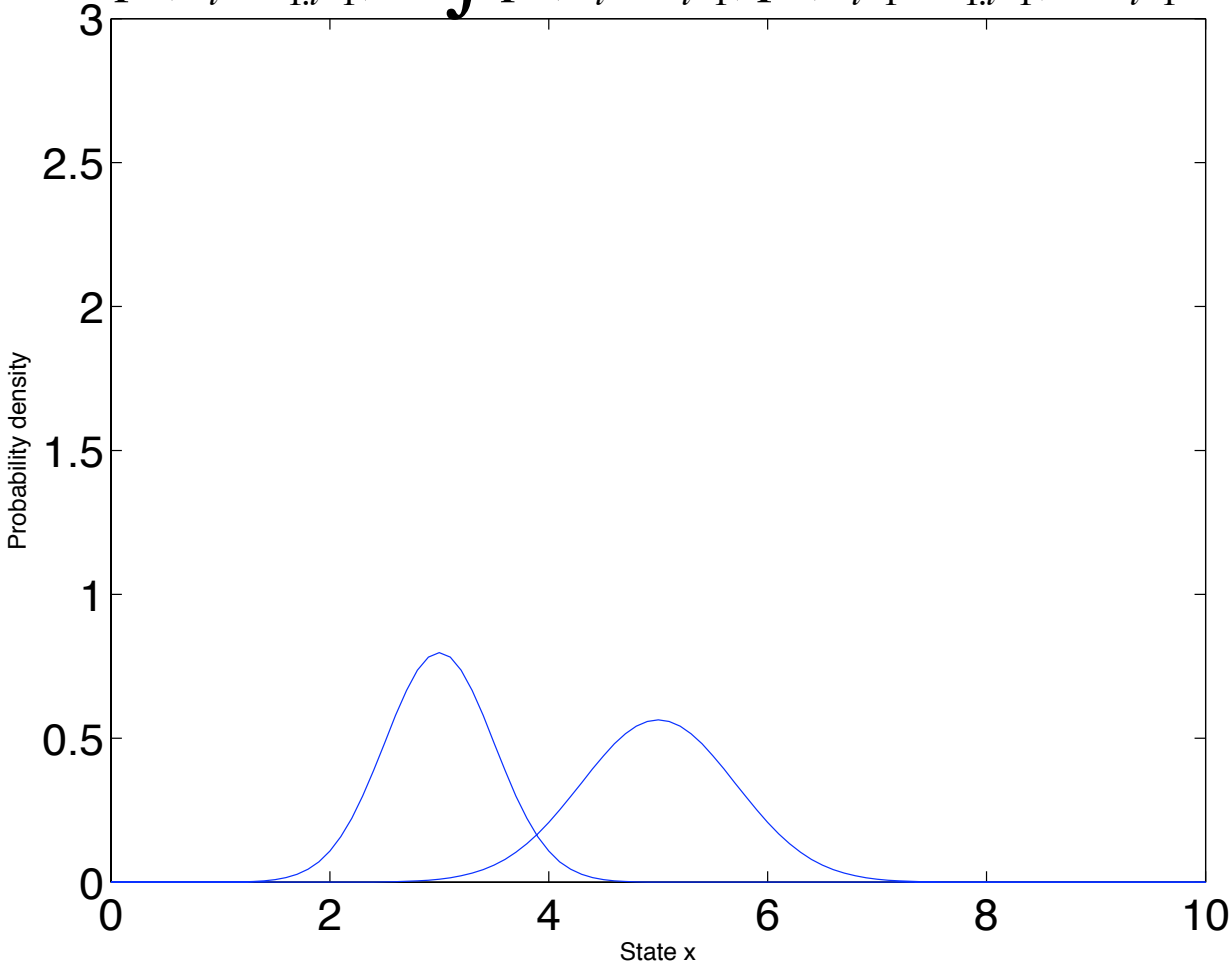
- Choose Kalman gain *K* that minimizes the posterior state covariance P_t (the uncertainty after including the measurement) and is a function of the prediction error covariance:

$$K_t = \hat{P}_t H_t^T (H_t \hat{P}_t H_t^T + R_t)^{-1}$$

Example: 1D position tracking

Include prediction noise:

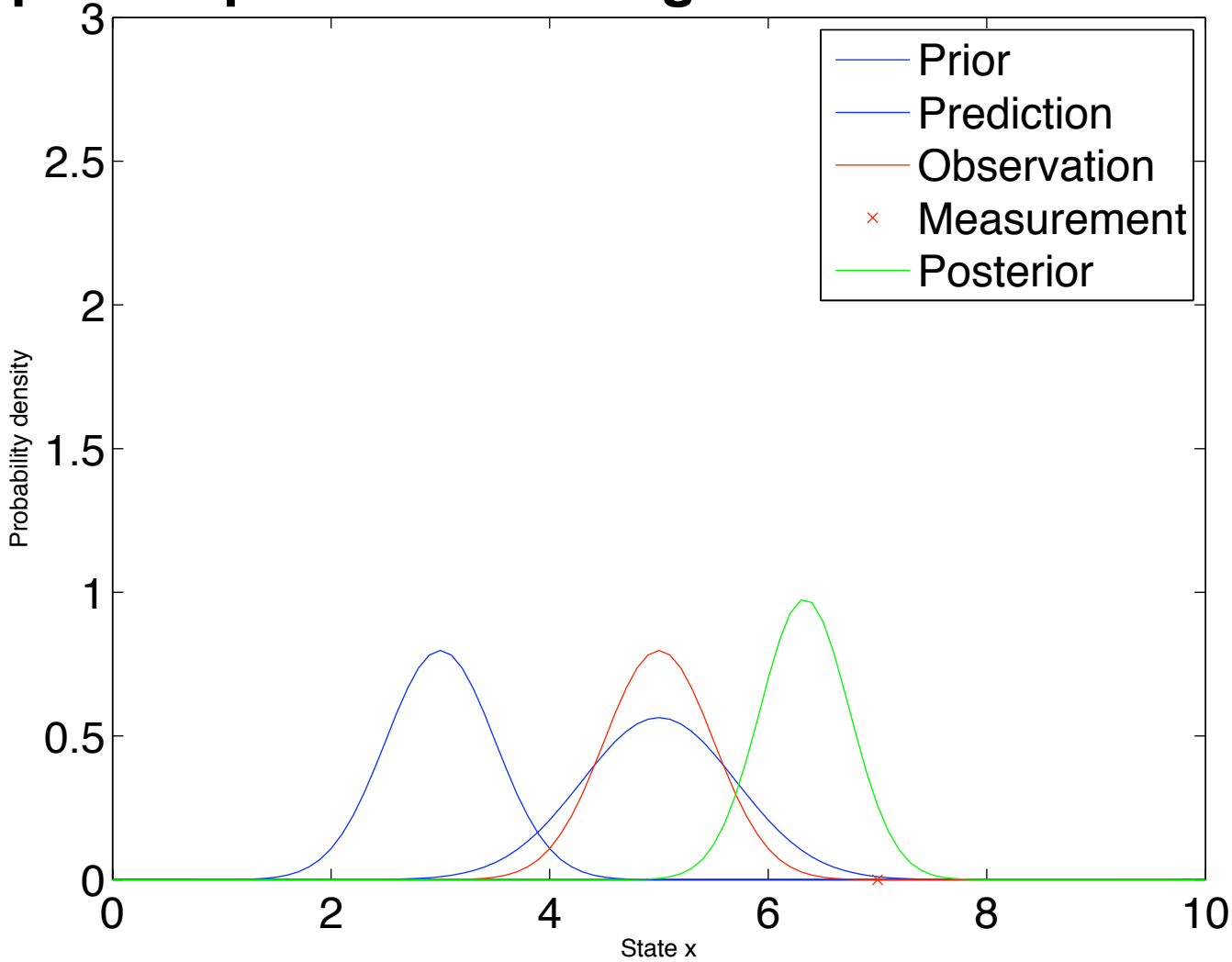
$$p(x_t \mid z_{1:t-1}) = \int p(x_t \mid x_{t-1}) p(x_{t-1} \mid z_{1:t-1}) dx_{t-1}$$



$$\hat{P}_t = 0.7^2$$

Include prediction noise $\hat{x}_t \sim N(\hat{x}_t \mid 5, 0.5^2 + 0.5^2) = N(\hat{x}_t \mid 5, 0.7^2)$

Example: 1D position tracking



$$p(z_t \mid \hat{x}_t) = N(z_t \mid 5, 0.5^2) \text{ and } x_t = \hat{x}_t + K_t(7 - \hat{x}_t) = 6.33 \ , P_t = 0.41^2$$

$$p(x_t \mid z_{1:t}) = N(x_t \mid 6.33, 0.41^2)$$

The Kalman filtering algorithm

Initialize estimates x_0 and P_0 and start with prediction step.

Iterate through these steps to compute estimates:

- Prediction:

$$\hat{x}_t = A_t x_{t-1} + B_t u_t \qquad \text{(Predict new state using dynamical model)}$$

$$\hat{P}_t = A_t P_{t-1} A_t^T + Q_t \qquad \text{(Prediction error covariance)}$$

- Correction:

$$K_t = \hat{P}_t H_t^T (H_t \hat{P}_t H_t^T + R_t)^{-1} \qquad \text{(Kalman gain)}$$

$$x_t = \hat{x}_t + K_t (z_t - H_t \hat{x}_t) \qquad \text{(Update state with observation)}$$

$$P_t = (I - K_k H_t) \hat{P}_t \qquad \text{(Update state covariance)}$$

What does Kalman gain do?

- Kalman gain: $K_t = \hat{P}_t H_t^T (H_t \hat{P}_t H_t^T + R_t)^{-1}$
- Update: $x_t = \hat{x}_t + K_t(z_t - H_t \hat{x}_t)$
- Observation error covariance:

$$\lim_{R_t \to 0} K_t = \lim_{R_t \to 0} \hat{P}_t H_t^T (H_t \hat{P}_t H_t^T + R_t)^{-1} = \hat{P}_t H_t^T (H_t \hat{P}_t H_t^T)^{-1}$$

$$= \hat{P}_t H_t^T (H_t^T)^{-1} \hat{P}_t^{-1} H_t^{-1} = H_t^{-1}$$

  - Intuition: Small observation error covariance means our observations are quite precise / good. Hence trust observations and update with full residual error!
- Prediction error covariance:

$$\lim_{\hat{P}_t \to 0} K_t = \lim_{\hat{P}_t \to 0} \hat{P}_t H_t^T (H_t \hat{P}_t H_t^T + R_t)^{-1} = 0$$

  - Intuition: Small prediction error covariance means our prediction is quite precise / good. Hence trust predictions!

Summary of Kalman filtering

The assumptions are:

- The dynamical and observation models are linear.
- Both the dynamical and observation noises are Gaussian distributed.

The Kalman filter tracks an estimate of a Gaussian filtering distribution, i.e. by tracking the mean and covariance.

Are these assumptions always valid?

Probably not – depends on the nature of the tracking problem.

Extensions of the Kalman filter exists

Examples:

- Extended Kalman filter:
  - Linearize both the dynamics and observation models
  - Apply the Kalman filter to this linear approximation

- Unscented Kalman filter:
  - Represent the state covariance and mean with samples / particles
  - Apply the (non-linear) dynamic and observation models to the samples
  - Reestimate mean and covariance from the samples

Literature

- Bishop Ch. 13.3
- Welch & Bishop: An Introduction to the Kalman Filter. SIGGRAPH course notes. 2001.
- S. Thrun, W. Burgard, D. Fox: Probabilistic Robotics. The MIT Press, 2005. Ch. 1 – 4.

(All references available in Absalon under the menu item Course material)

Assignment 3: Tracking with Bayesian Filtering
(Last assignment)

- Noisy pendulum:
  - Implement a Kalman and a particle filter
  - Apply them to a noisy pendulum data set and do experiments

- Optional question:
  - Apply your filters to a real tracking data set of a person standing on one leg. See if you can handle missing observations.