Lecture 12: Markov and Hidden Markov Models

Prof. Krishna R. Pattipati
Dept. of Electrical and Computer Engineering
University of Connecticut
Contact: krishna@engr.uconn.edu (860) 486-2890

December 3, 2018

Lecture Outline

- **Introduction to HMMs**
  - Markov Models
  - Hidden Markov Models
  - Applications of HMMs
- **Inference in HMMs**
  - Filtering, Smoothing, Viterbi, Classification
- **Learning HMM Parameters**
  - EM algorithm (Baum-Welch algorithm)
- **Generalizations of HMMs**

Analysis of Sequential Data

- Sequential structure arises in a broad range of applications
  - Repeated measurements of a temporal process
  - Online decision making & control
  - Text, biological sequences, etc
- Standard machine learning methods are often difficult to directly apply
  - Do not exploit temporal correlations
  - Computation & storage requirements typically scale poorly to realistic applications

Sequential Processes

- Consider a system which can occupy one of N discrete *states* or *categories*

$$x_t \in \{1, 2, .., N\}; x_t = \text{state at time } t$$

- We are interested in *stochastic* systems, in which state evolution is random
- Any *joint* distribution can be factored into a series of *conditional* distributions:

$$P(x_0, x_1, x_2, ..., x_{t-1}, x_t, ..., x_{T-1}, x_T) = P(x_0)P(x_1 \mid x_0).....P(x_t \mid x_0, x_1, .., x_{t-1})....P(x_T \mid x_0, x_1, .., x_{T-1})$$

$$= P(x_0)\prod_{t=1}^{T} P(x_t \mid \{x_k\}_{k=0}^{t-1})$$

$$Example: P(A, B, C, D) = P(A)P(B \mid A)P(C \mid A, B)P(D \mid A, B, C)$$

Markov Processes

- For a *Markov* process, the next state depends only on the current state: ("**bigram**" model)

$$P(x_t \mid x_0, x_1, .., x_{t-1}) = P(x_t \mid x_{t-1})$$

- This property in turn implies that

$$P(x_0, x_1, x_2, ..., x_{t-1}, x_{t+1}, ..., x_{T-1}, x_T \mid x_t) = \frac{P(x_0, x_1, x_2, ..., x_{t-1}, x_t, ..., x_{T-1}, x_T)}{P(x_t)}$$

$$= \frac{P(x_0, x_1, x_2, ..., x_{t-1}, x_t) P(x_{t+1}, ..., x_{T-1}, x_T \mid x_0, x_1, x_2, ..., x_{t-1}, x_t)}{P(x_t)}$$

$$= P(x_0, x_1, x_2, ..., x_{t-1} \mid x_t) P(x_{t+1}, ..., x_{T-1}, x_T \mid x_t)$$

In particular, $x_{t+1} \perp x_{t-1} \mid x_t \Rightarrow x_t$ separates x_{t-1} and x_{t+1}

"Conditioned on the present, the past & future are independent"

State Transition Matrix

- A *stationary* Markov chain with N states is described by an N x N *discrete-time transition matrix:*

$$P = \begin{bmatrix} P_{11} & P_{12} & P_{13} \\ P_{21} & P_{22} & P_{23} \\ P_{31} & P_{32} & P_{33} \end{bmatrix}$$

$$P_{ij} \triangleq P(x_{t+1} = j \mid x_t = i)$$

- Constraints on valid transition matrices (**row sum=1**)

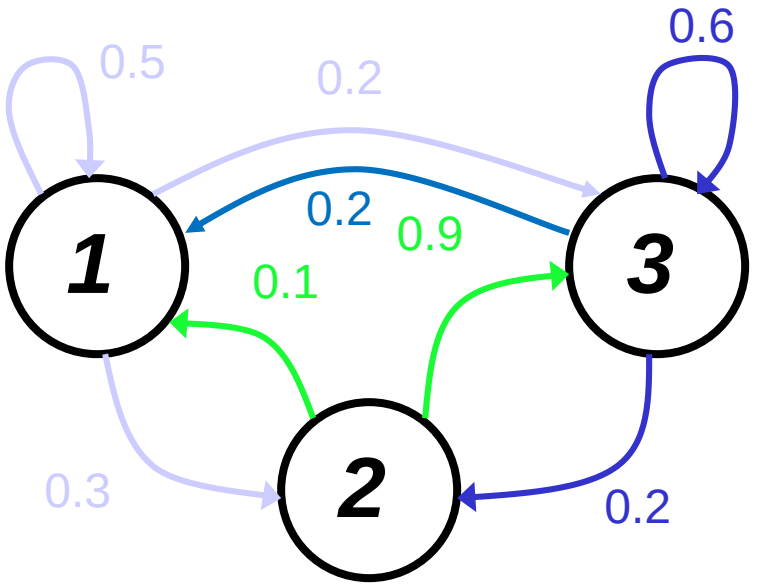
$$P_{ij} \geq 0 \quad \& \quad \sum_{j=1}^{N} P_{ij} = 1 \; \forall i = 1, 2, .., N \Rightarrow P\underline{e} = \underline{e}$$

$$\Rightarrow \lambda = 1 \; eigen \; value \; \& \; \underline{e} \; is \; an \; eigen \; vector$$

State Transition Diagrams

$$P_{ij} \triangleq P(x_{t+1} = j \mid x_t = i)$$

$$P = \begin{bmatrix} 0.5 & 0.3 & 0.2 \\ 0.1 & 0 & 0.9 \\ 0.2 & 0.2 & 0.6 \end{bmatrix}$$



- Think of a particle randomly following an arrow at each discrete time step
- Most useful when *N* is small, and *P* is *sparse*

Markov Models: Applications

• Statistical **language models (**probability distributions over sequences of words)

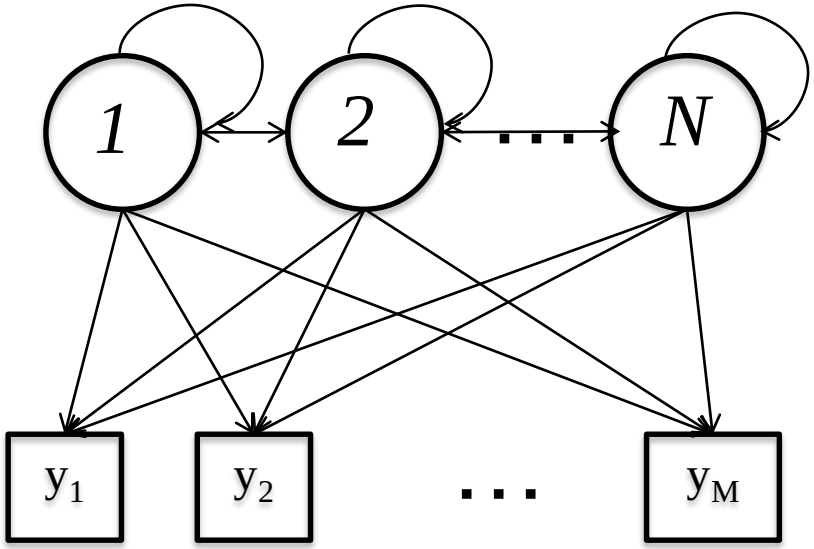
- The marginal probabilities $P(x_t = j)$ are called **unigram** statistics.
- First-order Markov model, then

  $P(x_t = j \mid x_{t-1} = i)$ is called a **bigram** model.
- Second-order Markov model, then

  $P(x_t = k \mid x_{t-1} = j, x_{t-2} = i)$ is called a **trigram** model.

- **Uses of Language models: superseded by LSTM and attention models**
  - Sentence completion: A language model can predict the next word given the previous words in a sentence (reduce the amount of typing required, particularly for disabled users)
  - Data compression
  - Text classification
  - Automatic essay writing

Formalization of a HMM

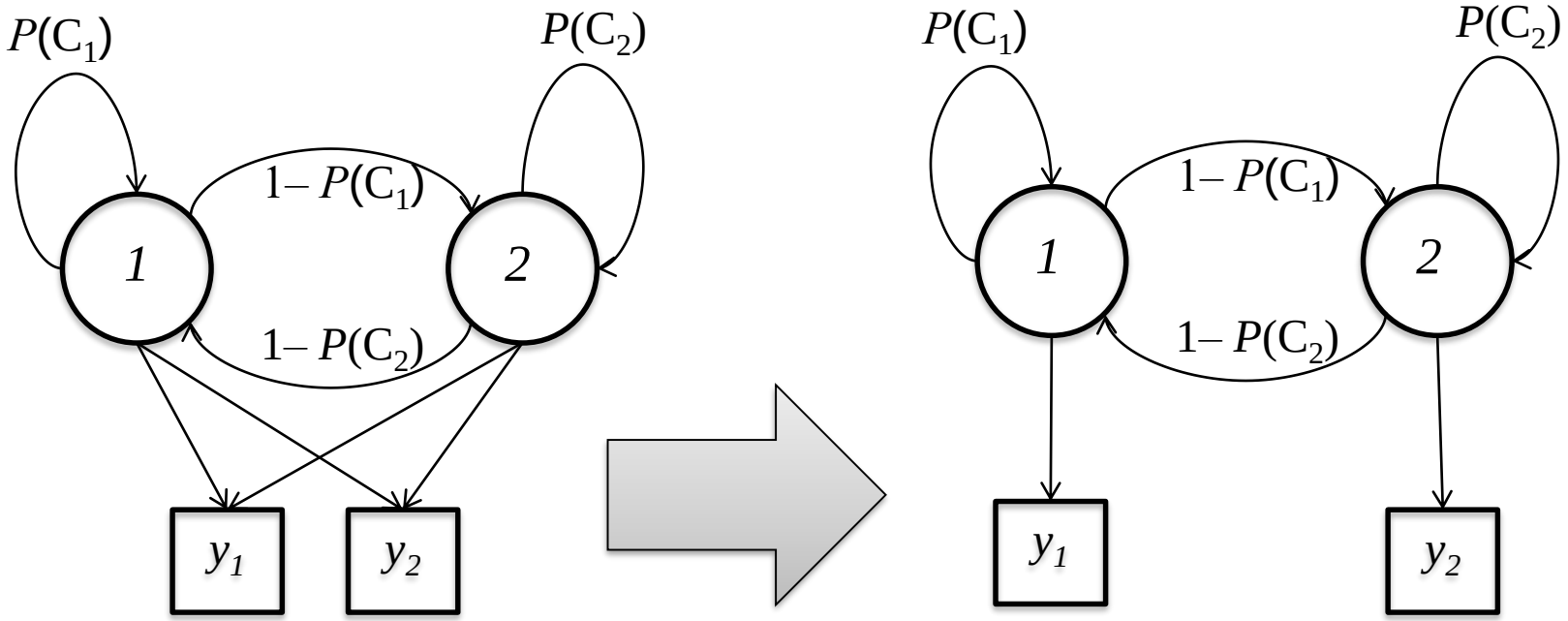


$$P_{ij} \triangleq P(x_{t+1} = j \mid x_t = i)$$

Elements of a discrete-time discrete-state HMM

1. Number of states, N
2. Number of observations, M
3. State transition probability matrix, P (NxN matrix)
4. Observation (emission) probability matrix, B (MxN matrix)
5. Prior probability vector, $\underline{\pi}$ (Nx1 vector)

MC ≡ Fully Observed HMM



Emission Probability Matrix

	1	2
$P(y_1)$	b_{11}	b_{12}
$P(y_2)$	$1\text{-}b_{11}$	$1\text{-}b_{12}$

Transition Probability Matrix

	1	2
1	P_{11}	P_{12}
2	P_{21}	P_{22}

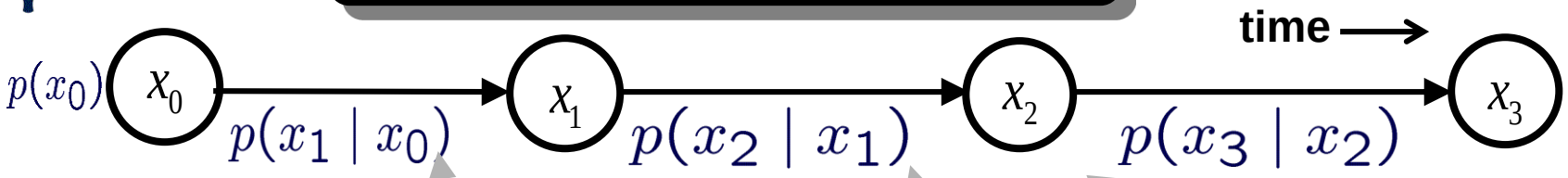
Number of states and number of outputs are always the same (i.e., the emission matrix is an identity matrix)

	1	2
$P(y_1)$	1	0
$P(y_2)$	0	1

	1	2
1	P_{11}	P_{12}
2	P_{21}	P_{22}

MC: Fully observed HMM

Markov Chain Statistics



$$\underline{\alpha}_t = \begin{bmatrix} P(x_t = 1) \\ P(x_t = 2) \\ . \\ P(x_t = N) \end{bmatrix}; \underline{\beta}_t = \begin{bmatrix} P(x_T \in \Omega \mid x_t = 1) \\ P(x_T \in \Omega \mid x_t = 2) \\ . \\ P(x_T \in \Omega \mid x_t = N) \end{bmatrix}$$

$$P = \begin{bmatrix} P_{11} & P_{12} & P_{13} \\ P_{21} & P_{22} & P_{23} \\ P_{31} & P_{32} & P_{33} \end{bmatrix}$$

$$P_{ij} \triangleq P(x_{t+1} = j \mid x_t = i)$$

$$Note: P(x_T \in \Omega) = \underline{\beta}_t^T \underline{\alpha}_t \forall t$$

- Forward and Backward Kolmogorov equations

$$\alpha_t(j) = \sum_{i=1}^{N} P_{ij} \alpha_{t-1}(i)$$

$$\underline{\alpha}_t = P^T \underline{\alpha}_{t-1} = \left(P^T\right)^t \underline{\alpha}_0 = \left(P^T\right)^t \underline{\pi}$$

$$\underline{\alpha}_\infty = \left(P^T\right)^\infty \underline{\alpha}_0 = P^T \underline{\alpha}_\infty$$

= eigen vector of P^T

corresponding to eigen value 1.

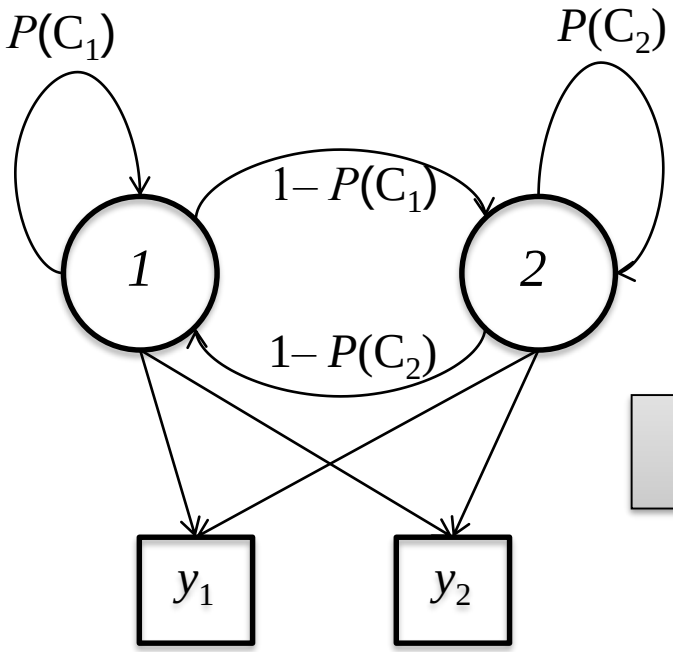
$$\beta_t(i) = \sum_{j=1}^{N} P(x_T \in \Omega, x_{t+1} = j \mid x_t = i) = \sum_{j=1}^{N} P_{ij} \beta_{t+1}(j)$$

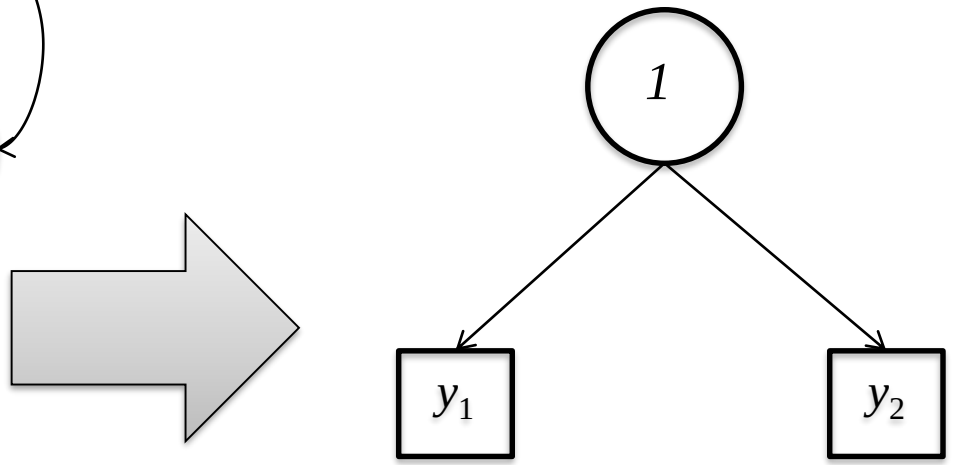
$$\underline{\beta}_t = P \underline{\beta}_{t+1}; \beta_T(i) = \begin{cases} 1 if\ i \in \Omega \\ 0\ if\ i \notin \Omega \end{cases}$$

No observations

Histogram ≡ a Special HMM

- Histogram/Discrete Distribution: A Special Case of HMM





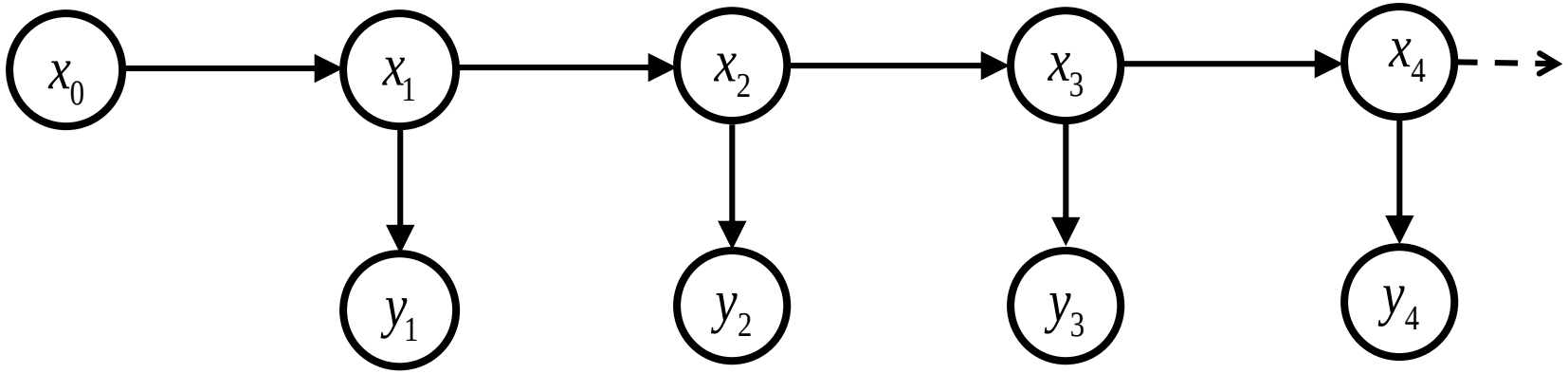
	1	2
P(y1)	b_{11}	b_{12}
P(y2)	$1\text{-}b_{11}$	$1\text{-}b_{12}$

	1	2
1	P_{11}	P_{12}
2	P_{21}	P_{22}

	1
P(y1)	b_{11}
P(y2)	$1\text{-}b_{11}$

	1
1	1

Graphical Model View of HMM

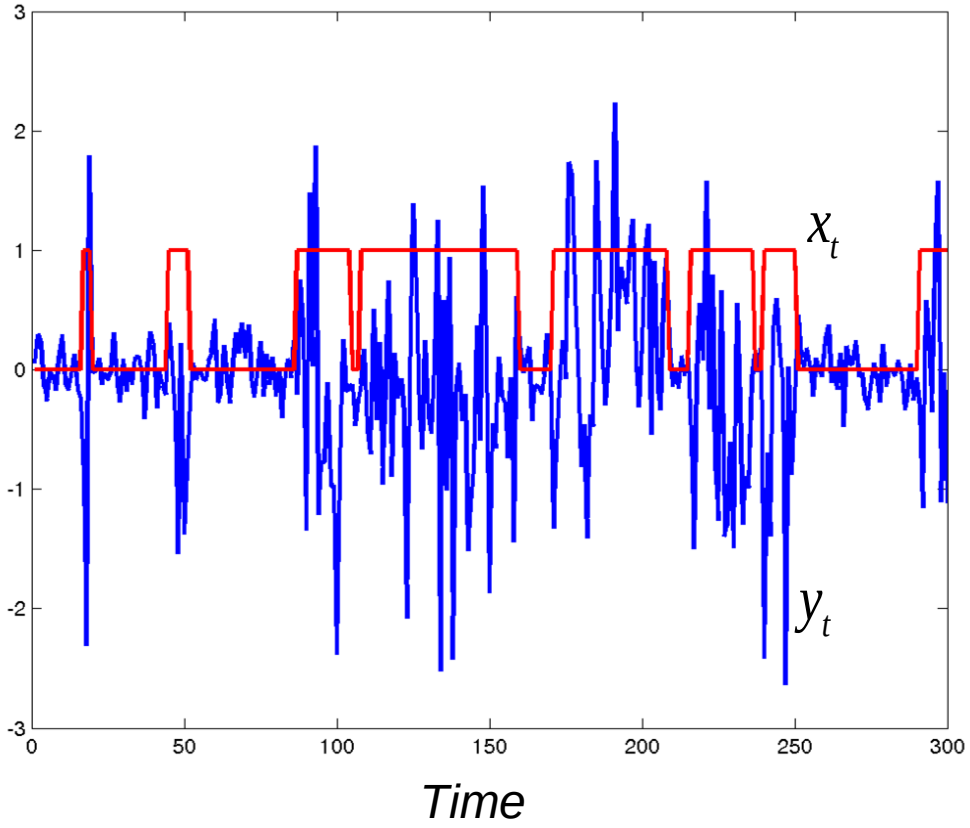


- Observation (emission) model:

  $P(y_t \mid x_t)$ *for discrete* y_t; $p(y_t \mid x_t)$ *for continuous* y_t

- Transition model: $P(x_{t+1} \mid x_t)$
- Initial state distribution: $P(x_0)$

Gilbert-Elliott Channel Model



Hidden State:

$$x_t \in \{0,1\}$$

$$P = \begin{bmatrix} 1-\varepsilon & \varepsilon \\ \varepsilon & 1-\varepsilon \end{bmatrix}$$

Observations:

$$y_t \sim N\{0, \sigma^2_{x_t}\}$$

$$\sigma_0^2 \sim \text{small}$$

$$\sigma_1^2 \sim \text{large}$$

Simple model for correlated, but bursty noise

(Elliott, 1963)

HMM Applications

- **Automatic speech recognition**: Here y_t represents features extracted from the speech signal, and x_t represents the word that is being spoken. The transition model $p(x_t|x_{t-1})$ represents the language model, and the observation model $p(y_t|x_t)$ represents the acoustic model. (Jelinek 1997; Jurafsky and Martin 2008).
- **Activity recognition**: Here y_t represents features extracted from a video frame, and x_t is the class of activity the person is engaged in (e.g., running, walking, sitting, etc (Szeliski 2010).
- **Part of speech tagging**: Here y_t represents a word, and x_t represents its part of speech (noun, verb, adjective, etc.)
- **Gene finding**: Here y_t represents the DNA nucleotides (A,C,G,T), and x_t represents whether we are inside a gene-coding region or not. (Schweikerta et al. 2009).
- **Protein sequence alignment**: see (Durbin et al. 1998) for details on profile HMMs.
- **Time series Prediction, as a black-box model of sequences,…**

Filtering, Smoothing & Prediction

- For online data analysis, we seek *filtered* state estimates given observations so far

$$P(x_t \mid y_1, y_2, ..., y_t) \qquad t = 1, 2, ...,$$

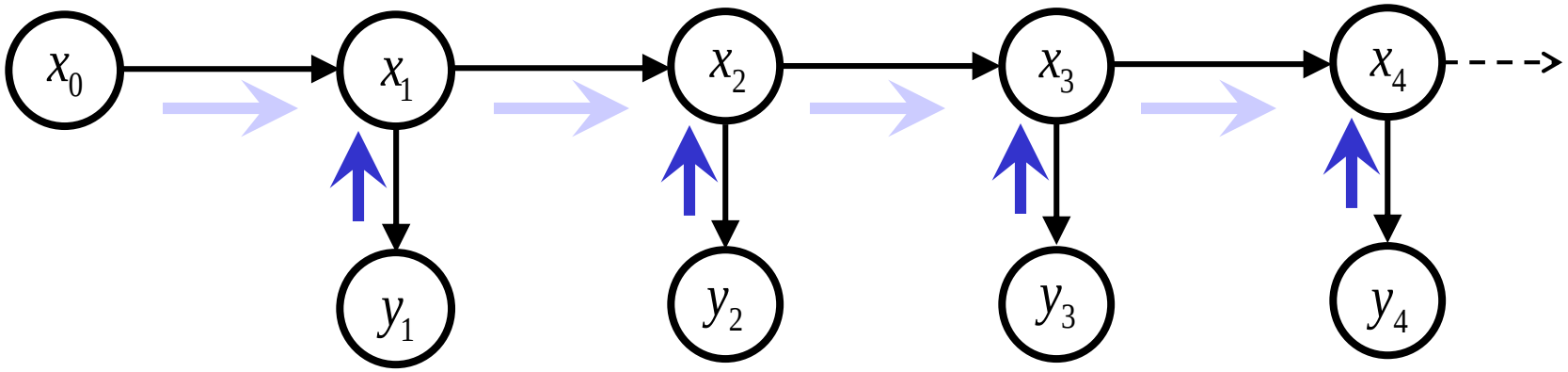
- In other cases, find *smoothed* estimates given earlier and later observations

$$P(x_t \mid y_1, y_2, ..., y_T) \qquad t = 1, 2, ...,$$

- Lots of other alternatives, including *fixed-lag smoothing* & *fixed-lag prediction*

$$P(x_t \mid y_1, y_2, ..., y_{t+L}) \qquad P(x_t \mid y_1, y_2, ..., y_{t-L})$$

Discrete HMMs: Filtering



$$\delta_t(x_t) \triangleq \overbrace{P(x_t \mid y_1, y_2, .., y_{t-1}, y_t)}^{Update} = \frac{\overbrace{P(x_t, y_1, y_2, .., y_{t-1}, y_t)}^{\alpha_t(x_t)}}{P(y_1, y_2, ..., y_t)} = \frac{P(y_t \mid x_t)\sum_{x_{t-1}} P(x_t \mid x_{t-1}) P(x_{t-1} \mid y_1, y_2, ..., y_{t-1})}{P(y_t \mid y_1, y_2, .., y_{t-1})}$$

$$= \frac{1}{Z_t} P(y_t \mid x_t) \sum_{x_{t-1}} P(x_t \mid x_{t-1}) \delta_{t-1}(x_{t-1}); \delta_0(x_0) = P(x_0)$$

Prediction: $P(x_t \mid y_1, y_2, .., y_{t-1})$

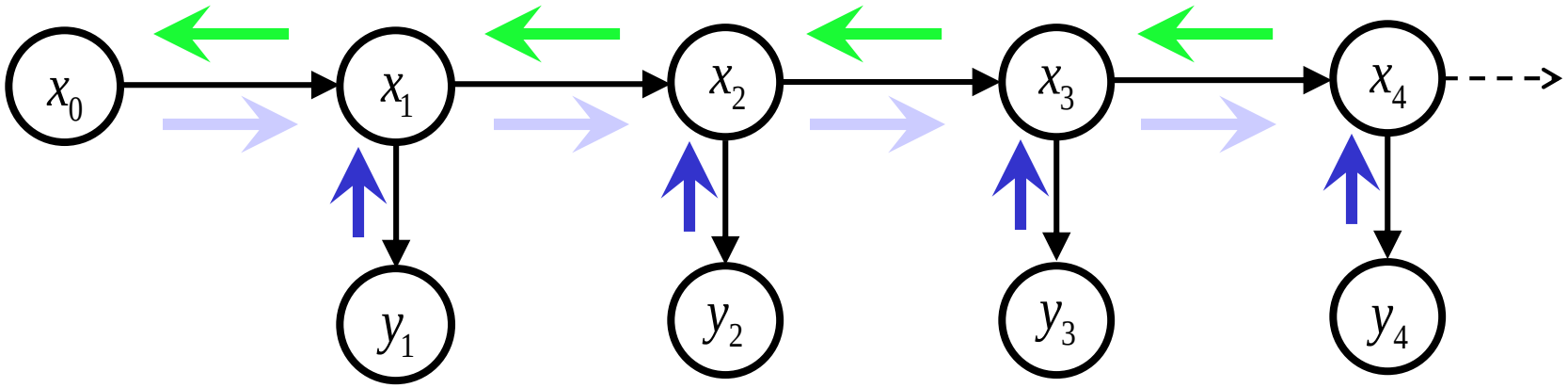
Likelihood: $P(y_t \mid x_t)$

Normalization constant, $P(y_t | \{y_\tau : \tau =$

$$\delta_t(j) = \frac{b_{y_t j} \sum_{i=1}^{N} P_{ij} \delta_{t-1}(i)}{\sum_{j=1}^{N} \sum_{i=1}^{N} b_{y_t j} P_{ij} \delta_{t-1}(i)}$$

Incorporates T observations in $\mathcal{O}(TN^2)$ *operations*

Discrete HMMs: Smoothing



$$\gamma_t(x_t) = P(x_t \mid y_1, y_2, \ldots, y_t, y_{t+1}, \ldots, y_T) = \frac{P(x_t, y_1, y_2, \ldots, y_t, y_{t+1}, \ldots, y_T)}{P(y_1, y_2, \ldots, y_t, y_{t+1}, \ldots, y_T)}$$

$$= \frac{P(y_1, y_2, \ldots, y_t, x_t) P(y_{t+1}, \ldots, y_T \mid x_t)}{P(y_1, y_2, \ldots, y_t, y_{t+1}, \ldots, y_T)}$$

$$= \frac{\alpha_t(x_t)\beta_t(x_t)}{\sum_{x_t} \alpha_t(x_t)\beta_t(x_t)}$$

$\alpha_t(x_t)$ = joint probability of observing all of the data upto time t and the value of x_t

$\beta_t(x_t)$ = conditional probability of all future data from time $(t+1)$ to T given the value of x_t

Forward-Backward Recursions

- **Forward recursion**

$$\alpha_t(x_t) = P(y_1, y_2, ..., y_{t-1}, y_t, x_t) = P(y_t \mid x_t) P(y_1, y_2, ..., y_{t-1}, x_t)$$

$$= P(y_t \mid x_t) \sum_{x_{t-1}} P(x_t \mid x_{t-1}) P(y_1, y_2, ..., y_{t-1}, x_{t-1})$$

$$= P(y_t \mid x_t) \sum_{x_{t-1}} P(x_t \mid x_{t-1}) \alpha_{t-1}(x_{t-1}); \alpha_0(x_0) = P(x_0) = \pi(x_0)$$

$$\delta_t(x_t) = \frac{P(y_1, y_2, ..., y_{t-1}, y_t, x_t)}{P(y_1, y_2, ..., y_{t-1}, y_t)} = \frac{\alpha_t(x_t)}{\sum_{\tilde{x}_t} \alpha_t(\tilde{x}_t)}$$

- **Backward Recursion**

$$\beta_t(x_t) = P(y_{t+1}, y_{t+2}, ..., y_T \mid x_t) = \sum_{x_{t+1}} P(x_{t+1}, y_{t+1}, y_{t+2}, ..., y_T \mid x_t)$$

$$= \sum_{x_{t+1}} P(y_{t+2}, y_{t+3}, ..., y_T \mid x_{t+1}) P(y_{t+1} \mid x_{t+1}) P(x_{t+1} \mid x_t)$$

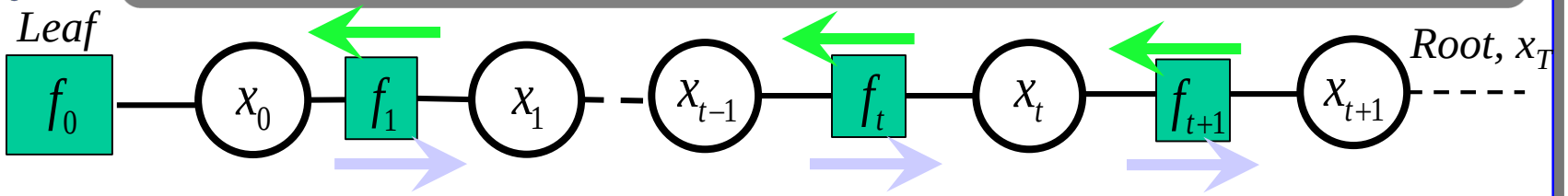
$$= \sum_{x_{t+1}} \beta_{t+1}(x_{t+1}) P(y_{t+1} \mid x_{t+1}) P(x_{t+1} \mid x_t); \beta_T(x_T) = 1 \forall x_T$$

α-β algorithm (or) Kolmogorov equations for HMM

$$Why: \gamma_T(x_T) = \delta_T(x_T) = P(x_T \mid y_1, y_2, ..., y_T) = \frac{P(x_T, y_1, y_2, ..., y_T)}{P(y_1, y_2, ..., y_T)} = = \frac{\alpha_T(x_T)}{\sum_{\tilde{x}_T} \alpha_T(\tilde{x}_T)} \Rightarrow \beta_T(x_T) = 1 \forall x_T$$

$$Also, P(y_1, y_2, ..., y_T) = \sum_{x_t} \alpha_t(x_t) \beta_t(x_t) \ \forall t. \text{ Note that } P(y_1, y_2, ..., y_T) = \sum_{x_T} \alpha_T(x_T) = \sum_{x_0} \alpha_0(x_0) \beta_0(x_0)$$

Forward-Backward = Sum-Product Algorithm



- Absorb emission probabilities into the factors

$$f_0(x_0) = P(x_0)$$

$$f(x_{t-1}, x_t) = P(x_t \mid x_{t-1}) P(y_t \mid x_t)$$

Sum-product algorithm works for tree structured graphs (including junction trees for Bayesian networks)

- Leaf $\rightarrow$ Root forward propagation of messages ($\rightarrow$)

$$\mu_{x_{t-1} \to f_t} = \mu_{f_{t-1} \to x_{t-1}} = \alpha(x_{t-1})$$

$$\mu_{f_t \to x_t} = \alpha(x_t) = \sum_{x_{t-1}} f_t(x_{t-1}, x_t) \mu_{x_{t-1} \to f_t} = \sum_{x_{t-1}} f_t(x_{t-1}, x_t) \mu_{f_{t-1} \to x_{t-1}} = P(y_t \mid x_t) \sum_{x_{t-1}} P(x_t \mid x_{t-1}) \alpha(x_{t-1})$$

Initial condition: $\mu_{f_0 \to x_0} = \mu_{x_0 \to f_1} = \alpha(x_0) = P(x_0)$

- Root $\rightarrow$ Leaf backward propagation of messages ($\leftarrow$)

$$\mu_{x_{t+1} \to f_{t+1}} = \mu_{f_{t+2} \to x_{t+1}} = \beta(x_{t+1})$$

$$\mu_{f_{t+1} \to x_t} = \beta(x_t) = \sum_{x_{t+1}} f_{t+1}(x_t, x_{t+1}) \mu_{x_{t+1} \to f_{t+1}} = \sum_{x_{t+1}} f_{t+1}(x_t, x_{t+1}) \mu_{f_{t+2} \to x_{t+1}} = \sum_{x_{t-1}} P(y_{t+1} \mid x_{t+1}) P(x_{t+1} \mid x_t) \beta(x_{t+1})$$

Terminal Condition: $\mu_{x_T \to f_T} = \mu_{f_{T+1} \to x_T} = \beta(x_T) = 1$

An Interesting Conditional Probability

- **Joint Conditional probability of (x_t, x_{t-1})**

$$\xi_t(x_{t-1}, x_t) = P(x_{t-1}, x_t \mid y_1, y_2, ..., y_{t-1}, y_t, ..., y_T)$$

$$= \frac{P(y_1, y_2, ..., y_{t-1}, y_t, ..., y_T, x_{t-1}, x_t)}{P(y_1, y_2, ..., y_{t-1}, y_t, ..., y_T)}$$

$$= \frac{P(y_1, y_2, ..., y_{t-1}, x_{t-1}) P(x_t, y_t, y_{t+1}, ..., y_T \mid x_{t-1})}{P(y_1, y_2, ..., y_{t-1}, y_t, ..., y_T)}$$

$$= \frac{P(y_1, y_2, ..., y_{t-1}, x_{t-1}) P(y_t, y_{t+1}, ..., y_T \mid x_t, x_{t-1}) P(x_t \mid x_{t-1})}{P(y_1, y_2, ..., y_{t-1}, y_t, ..., y_T)}$$

$$= \frac{P(y_1, y_2, ..., y_{t-1}, x_{t-1}) P(y_t, y_{t+1}, ..., y_T \mid x_t) P(x_t \mid x_{t-1})}{P(y_1, y_2, ..., y_{t-1}, y_t, ..., y_T)}$$

$$= \frac{P(y_1, y_2, ..., y_{t-1}, x_{t-1}) P(x_t \mid x_{t-1}) P(y_t \mid x_t) P(y_{t+1}, ..., y_T \mid x_t)}{P(y_1, y_2, ..., y_{t-1}, y_t, ..., y_T)}$$

$$= \frac{\alpha_{t-1}(x_{t-1}) P_{x_{t-1}x_t} b_{y_t x_t} \beta_t(x_t)}{\sum_{x_t} \alpha_t(x_t) \beta_t(x_t)} = \frac{\alpha_{t-1}(x_{t-1}) P_{x_{t-1}x_t} b_{y_t x_t} \beta_t(x_t)}{\sum_{x_{t-1}} \sum_{x_t} \alpha_{t-1}(x_{t-1}) b_{y_t x_t} P_{x_{t-1}x_t} \beta_t(x_t)}$$

Evidently,

$$\gamma_t(x_t) = P(x_t \mid y_1, y_2, ..., y_t, y_{t+1}, ..., y_T) = \sum_{x_{t-1}} \xi_t(x_{t-1}, x_t)$$

HMM Parameter Learning

$$\hat{\pi}_i = \text{expected freq. in state } i \text{ at time } (t=0) = \gamma_0(i)$$

$$\hat{P}_{ij} = \frac{\text{Expected no. of transitions from } i \text{ to } j}{\text{Expected no. of transitions from } i}$$

$$= \frac{\sum_{t=1}^{T} \xi_t(i,j)}{\sum_{t=0}^{T-1} \gamma_t(i)}$$

$$\gamma_t(x_t) = \frac{\alpha_t(x_t)\beta_t(x_t)}{\sum_{x_t} \alpha_t(x_t)\beta_t(x_t)}$$

$$\xi_t(x_{t-1}, x_t) = \frac{\alpha_{t-1}(x_{t-1}) b_{y_t x_t} P_{x_{t-1}x_t} \beta_t(x_t)}{\sum_{x_t} \alpha_t(x_t)\beta_t(x_t)}$$

$$\hat{b}_{kj} = \frac{\text{exptd. no. of times in state } j \text{ and observing symbol } k}{\text{expected number of times in state } j} = \frac{\sum_{t=1}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)}$$

$$\text{s.t. } y_t = k$$

To avoid small $\{\alpha, \beta\}$, normalize *both* using the same normalization factor $Z(t) = \sum_{i=1}^{N} \alpha_i(t)$

Called Baum-Welch Algorithm … EM. For given parameters, evaluate α_t, β_t, γ_t, ξ_t. Update parameters and iterate.

Viterbi Algorithm for the Most Probable Sequence

- **Problem: Find the most probable state sequence (MAP estimate) given data**

$$\underline{x}^T = \arg \max_{x_0, x_1, .., x_T} P(x_0, x_1, ..., x_t, ..., x_T \mid y_1, y_2, ..., y_t, .., y_T)$$

$$= \arg \max_{x_0, x_1, .., x_T} P(y_1, y_2, ..., y_t, .., y_T, x_0, x_1, ..., x_t, ..., x_T)$$

$$= \arg \max_{x_0, x_1, .., x_T} P(x_0) \prod_{t=1}^{T} \left[P(y_t \mid x_t) P(x_t \mid x_{t-1}) \right]$$

$$= \arg \max_{x_0, x_1, .., x_T} \left[\ln P(x_0) + \sum_{t=1}^{T} \left\{ \ln P(y_t \mid x_t) + \ln P(x_t \mid x_{t-1}) \right\} \right]$$

- **Use forward *dynamic programming* (DP) to recursively find the probability of the most likely state sequence**

$$\omega(x_0) = \ln P(x_0)$$

$$\omega(x_1) = \ln P(y_1 \mid x_1) + \max_{x_0} \left[\ln P(x_1 \mid x_0) + \underbrace{\ln P(x_0)}_{\omega(x_0)} \right]$$

Viterbi Algorithm for the Most Probable Sequence

- **DP recursion**

In general,

$$\omega(x_t) = \max_{x_0, x_1, .., x_{t-1}} P(y_1, y_2, ..., y_t, x_0, x_1, ..., x_t)$$

$$= \max_{x_0, x_1, .., x_{t-1}} \left[\ln P(x_0) + \sum_{n=1}^{t} \left\{ \ln P(y_n \mid x_n) + \ln P(x_n \mid x_{n-1}) \right\} \right]$$

$$= \ln P(y_t \mid x_t) + \max_{x_0, x_1, .., x_{t-1}} \left[\ln P(x_0) + \sum_{n=1}^{t-1} \ln P(y_n \mid x_n) + \sum_{n=1}^{t} \ln P(x_n \mid x_{n-1}) \right]$$

$$= \ln P(y_t \mid x_t) + \max_{x_{t-1}} \left[\ln P(x_t \mid x_{t-1}) + \omega(x_{t-1}) \right]; \omega(x_0) = \ln P(x_0); t = 1, 2, .., T$$

Keep a record of the values of x_{t-1} that correspond to the maxima for each of the N values of x_t.

Store this in a list function $\psi_t(x_t) = \arg\max_{x_{t-1}} \left[\ln P(x_t \mid x_{t-1}) + \omega(x_{t-1}) \right]$

- **Backtrack to get the best sequence**

$$x_T^* = \arg\max_{x_T} \omega(x_T)$$

$$x_t^* = \psi_{t+1}(x_{t+1}^*); t = T-1, T-2, .., 1, 0$$

Poor Man's Viterbi:

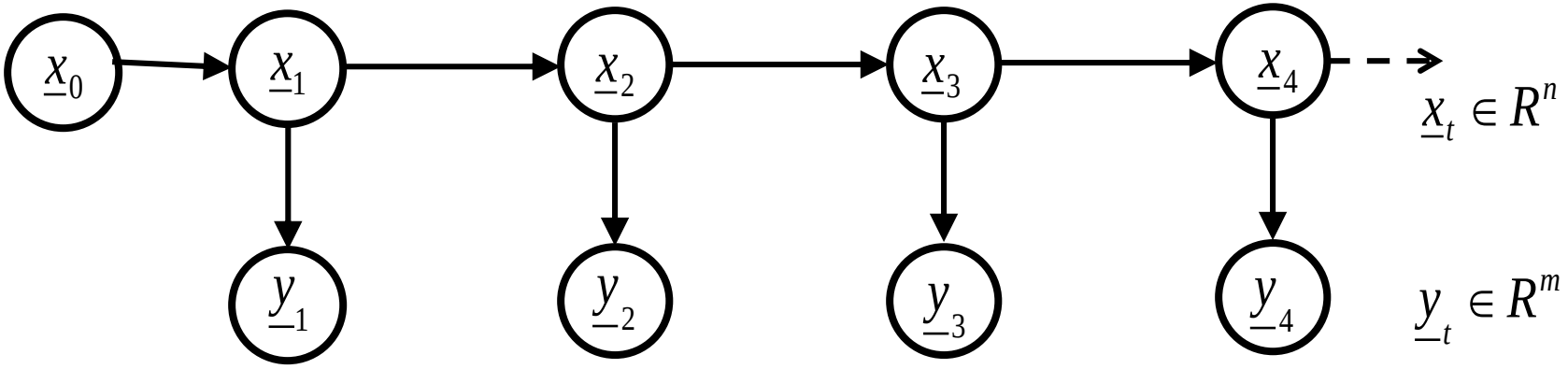
$$x_t^* = \arg\max_{x_t} \delta_t(x_t)$$

$$\delta_t(x_t) \triangleq P(x_t \mid y_1, y_2, .., y_{t-1}, y_t)$$

Generalizations of HMMs

- Higher order HMMs
- Factorial HMMs. See Tbishirani and some of my papers on diagnosis.
- Coupled HMMs. See some of my papers on diagnosis and refs there.
- Semi-Markov HMMs
- Hierarchical HMMs
- I/O HMMs
- Auto-regressive HMMs
- Buried HMMs
- **State space HMMs or State Space Models (SSMs)**
- Dynamic Bayesian networks (DBNs)

Comparison of HMM and SSM



$$\underline{x}_{t+1} = A\underline{x}_t + \underline{w}_t; \underline{w}_t \sim N(\underline{0}, Q); \underline{x}_0 \sim N(\bar{\underline{x}}_0, \Sigma_0)$$

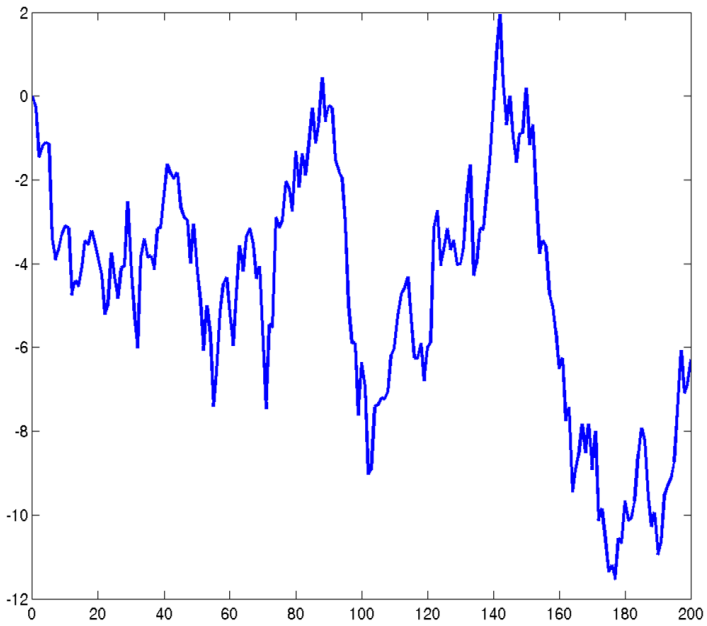
$$\underline{y}_t = C\underline{x}_t + \underline{v}_t; \underline{v}_t \sim N(\underline{0}, R)$$

- State Space Models (SMM) are continuous state HMMs
- States & observations are jointly Gaussian

Canonical Problems	HMM	SSM
Inference: Given (1) the output sequence $\{y_1, y_2, \ldots, y_T\}$ and (2) a model, how to compute the most likely or MMSE sate sequences $x_1, x_2, \ldots, x_T$	Viterbi algorithm (optimal) Forward algorithm (probabilistic)	Filtering & Smoothing • Kalman Filter /Smoother • Extended KF/KS • Unscented KF/KS • Particle Filter/Smoother
Learning: Given the observations $y_1, y_2, \ldots, y_T\}$, how do we fit an appropriate model to these observations?	Baum Welch Algorithm (EM algorithm)	EM algorithm/ML Identification

Simple Linear Dynamics

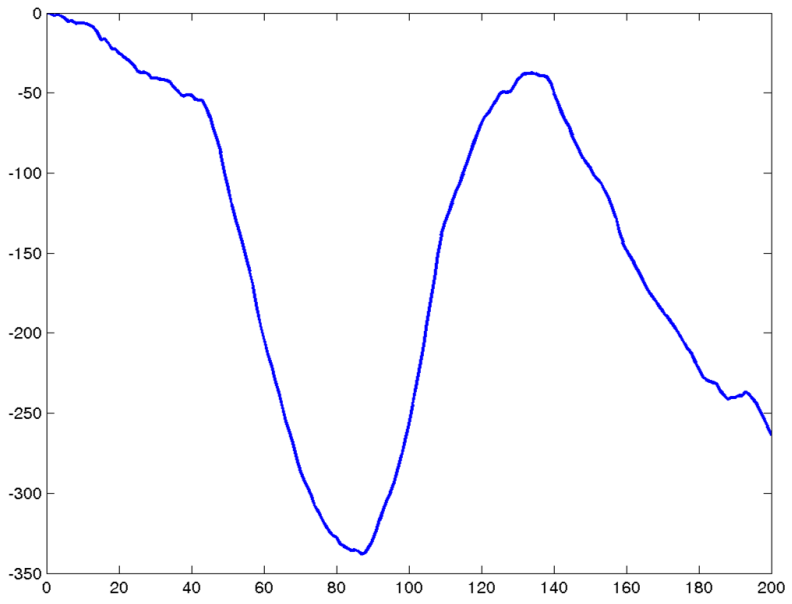
Brownian Motion



Time

$$X_{t+1} = X_t + W_t$$

Constant Velocity



Time

$$\begin{bmatrix} x_{t+1} \\ v_{t+1} \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_t \\ v_t \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} w_t$$

Kalman Filter

- Model

$$\underline{x}_{t+1} = A\underline{x}_t + \underline{w}_t; \underline{w}_t \sim N(\underline{0}, Q); \underline{x}_0 \sim N(\bar{\underline{x}}_0, \Sigma_0)$$

$$\underline{y}_t = C\underline{x}_t + \underline{v}_t; \underline{v}_t \sim N(\underline{0}, R)$$

- Represent Gaussians by *mean* & *covariance*

$$p(\underline{x}_t \mid \underline{y}_1, \underline{y}_2, .., \underline{y}_{t-1}) = N(\hat{\underline{x}}_{t|t-1}, \Sigma_{t|t-1})$$

$$p(\underline{x}_t \mid \underline{y}_1, \underline{y}_2, .., \underline{y}_t) = N(\hat{\underline{x}}_{t|t}, \Sigma_{t|t})$$

Recall $p(\underline{x}_t \mid \underline{y}_1, \underline{y}_2, .., \underline{y}_{t-1}) = \int_{\underline{x}_{t-1}} p(\underline{x}_t \mid \underline{x}_{t-1}) p(\underline{x}_{t-1} \mid \underline{y}_1, \underline{y}_2, ..., \underline{y}_{t-1}) d\underline{x}_{t-1}$ prediction

$$\Rightarrow \text{Prediction}: \hat{\underline{x}}_{t|t-1} = A\hat{\underline{x}}_{t-1|t-1}; \Sigma_{t|t-1} = A\Sigma_{t-1|t-1}A^T + Q$$

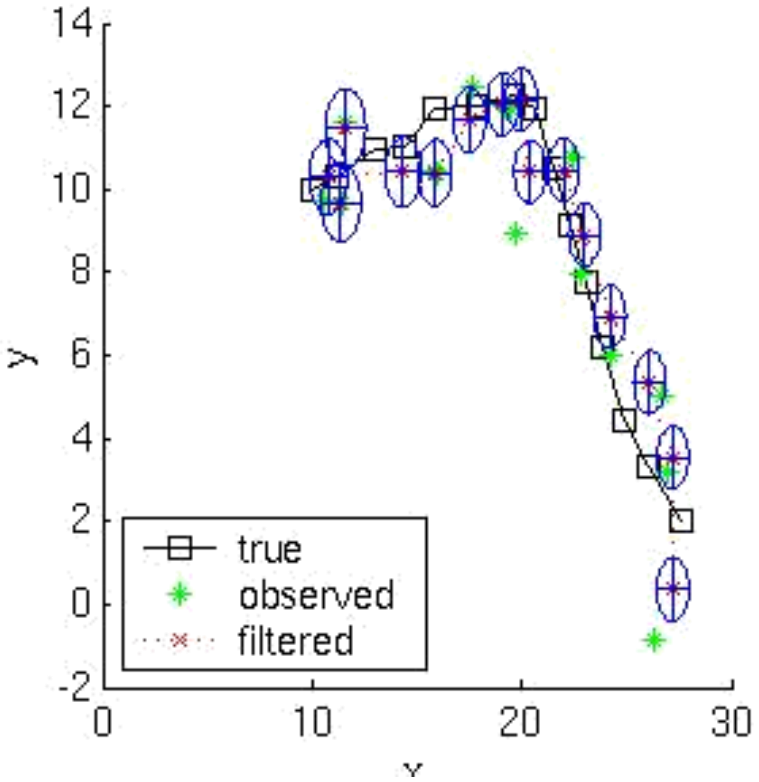
Also, $p(\underline{x}_t \mid \underline{y}_1, \underline{y}_2, .., \underline{y}_t) = \dfrac{p(\underline{y}_t \mid \underline{x}_t) p(\underline{x}_t \mid \underline{y}_1, \underline{y}_2, .., \underline{y}_{t-1})}{p(\underline{y}_t \mid \underline{y}_1, \underline{y}_2, .., \underline{y}_{t-1})}$ update

$$\Rightarrow \text{ Kalman Gain: } K_t = \Sigma_{t|t-1}C^T(C\Sigma_{t|t-1}C^T + R)^{-1} = \Sigma_{t|t}C^T R^{-1}$$

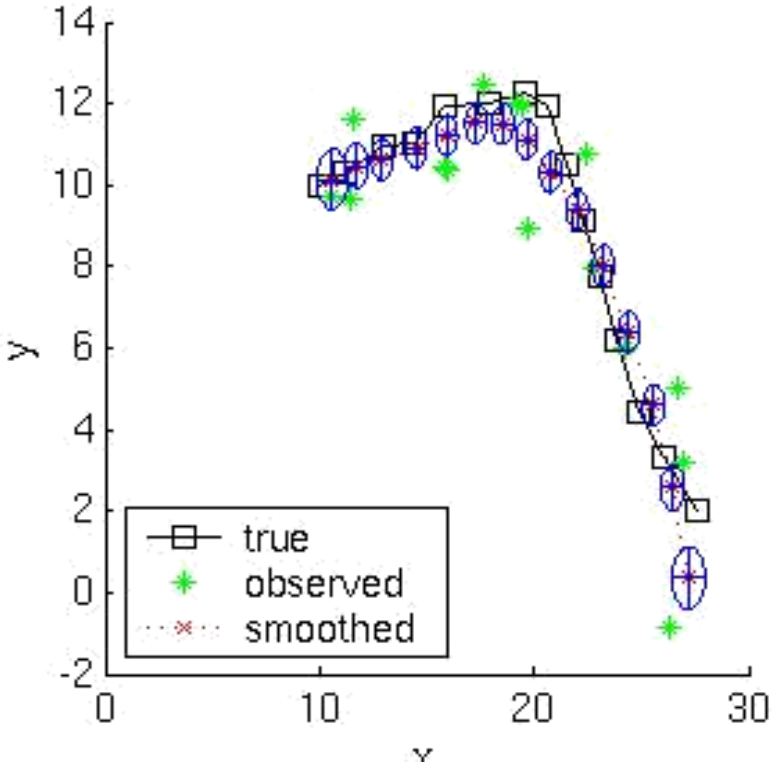
$$\Rightarrow \text{Update: } \hat{\underline{x}}_{t|t} = \hat{\underline{x}}_{t|t-1} + K_t\left(\underline{y}_t - C\hat{\underline{x}}_{t|t-1}\right); \Sigma_{t|t} = (I_n - K_t C)\Sigma_{t|t-1}$$

Constant Velocity Tracking

Kalman Filter

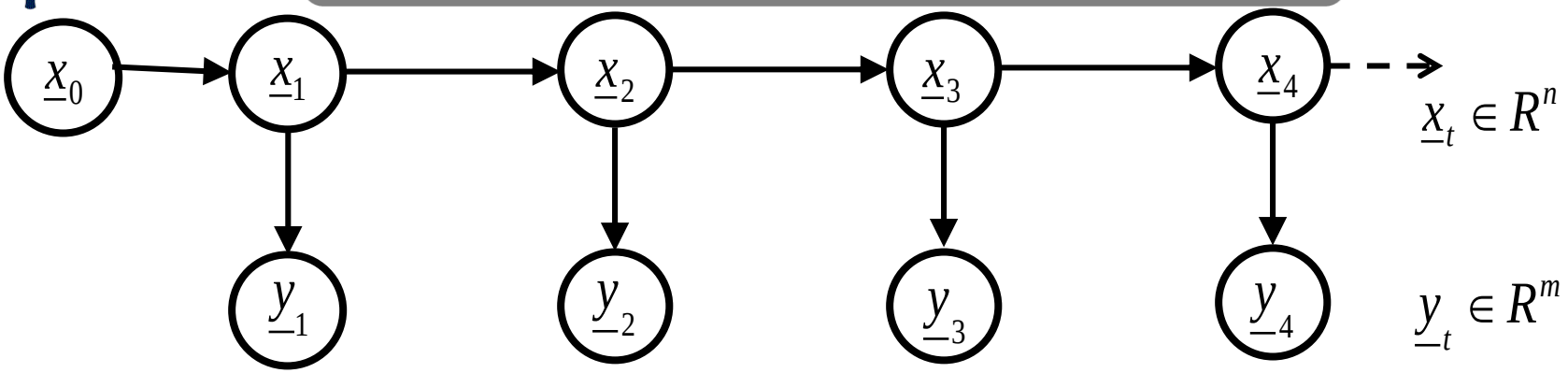


Kalman Smoother



(K. Murphy, 1998)

Nonlinear State Space Models



$$\underline{x}_{t+1} = f(\underline{x}_t, \underline{w}_t); \underline{w}_t \sim b; \underline{x}_0 \sim h$$

$$\underline{y}_t = g(\underline{x}_t, \underline{v}_t); \underline{v}_t \sim g$$

$$p(\underline{x}_t \mid \underline{y}_1, \underline{y}_2, .., \underline{y}_{t-1}) = \tilde{q}_t(\underline{x}_t)$$

$$p(\underline{x}_t \mid \underline{y}_1, \underline{y}_2, .., \underline{y}_t) = q_t(\underline{x}_t)$$

- State dynamics and measurements given by potentially complex *nonlinear functions*
- Initial state and noises sampled from *non-Gaussian* densities *h*, *b*, *g*

Prediction: $p(\underline{x}_t \mid \underline{y}_1, \underline{y}_2, .., \underline{y}_{t-1}) = \tilde{q}_t(\underline{x}_t) = \int_{\underline{x}_{t-1}} p(\underline{x}_t \mid \underline{x}_{t-1}) q_{t-1}(\underline{x}_{t-1}) \, d\underline{x}_{t-1}$

Update: $p(\underline{x}_t \mid \underline{y}_1, \underline{y}_2, .., \underline{y}_t) = q_t(\underline{x}_t) = \dfrac{p(\underline{y}_t \mid \underline{x}_t) \, \tilde{q}_t(\underline{x}_t)}{p(\underline{y}_t \mid \underline{y}_1, \underline{y}_2, .., \underline{y}_{t-1})} = \dfrac{p(\underline{y}_t \mid \underline{x}_t) \, \tilde{q}_t(\underline{x}_t)}{\int_{\underline{x}_t} p(\underline{y}_t \mid \underline{x}_t) \, \tilde{q}_t(\underline{x}_t) d\underline{x}_t}$

Approximate Nonlinear Filters

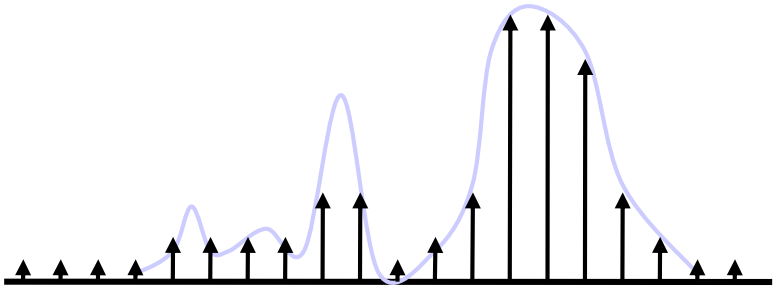
Basic Equation of Nonlinear Filtering: $q_t(\underline{x}_t) \propto p(\underline{y}_t \mid \underline{x}_t) \int_{\underline{x}_{t-1}} p(\underline{x}_t \mid \underline{x}_{t-1}) q_{t-1}(\underline{x}_{t-1})\, d\,\underline{x}_{t-1}$

- Typically cannot directly *represent* these continuous functions, or determine a closed form for the prediction *integral (unless components are Gaussian)*

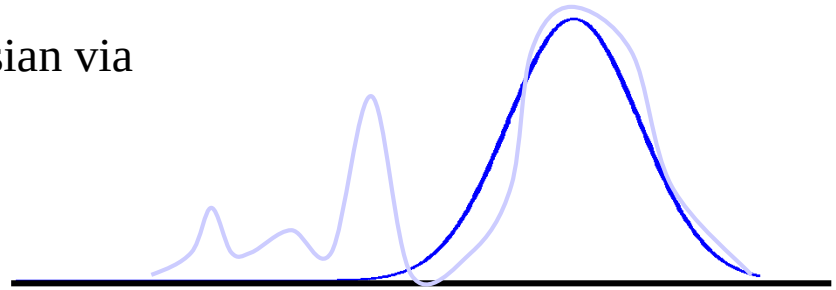
- A wide range of approximate nonlinear filters have been proposed, including
  - *Histogram filters*
  - *Extended & unscented Kalman filters*
  - *Particle filters*
  - *Assumed density filtering (ADF): Multiple Models, Interacting Multiple Models,…*

Nonlinear Filtering Taxonomy

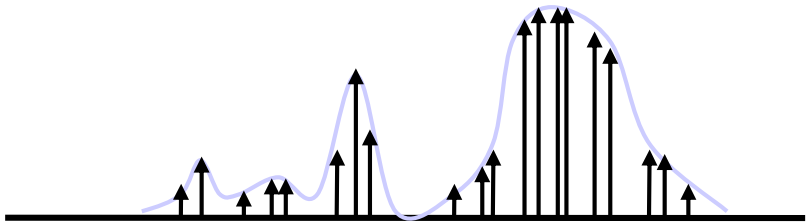
- **Histogram Filter**
  - Evaluate on a fixed discrete grid
  - Only feasible in low dimensions
  - Expensive or inaccurate



- **Extended Kalman Filter**
  - Approximate posterior as Gaussian via linearization, quadrature, …
  - Inaccurate for multimodal posterior distributions



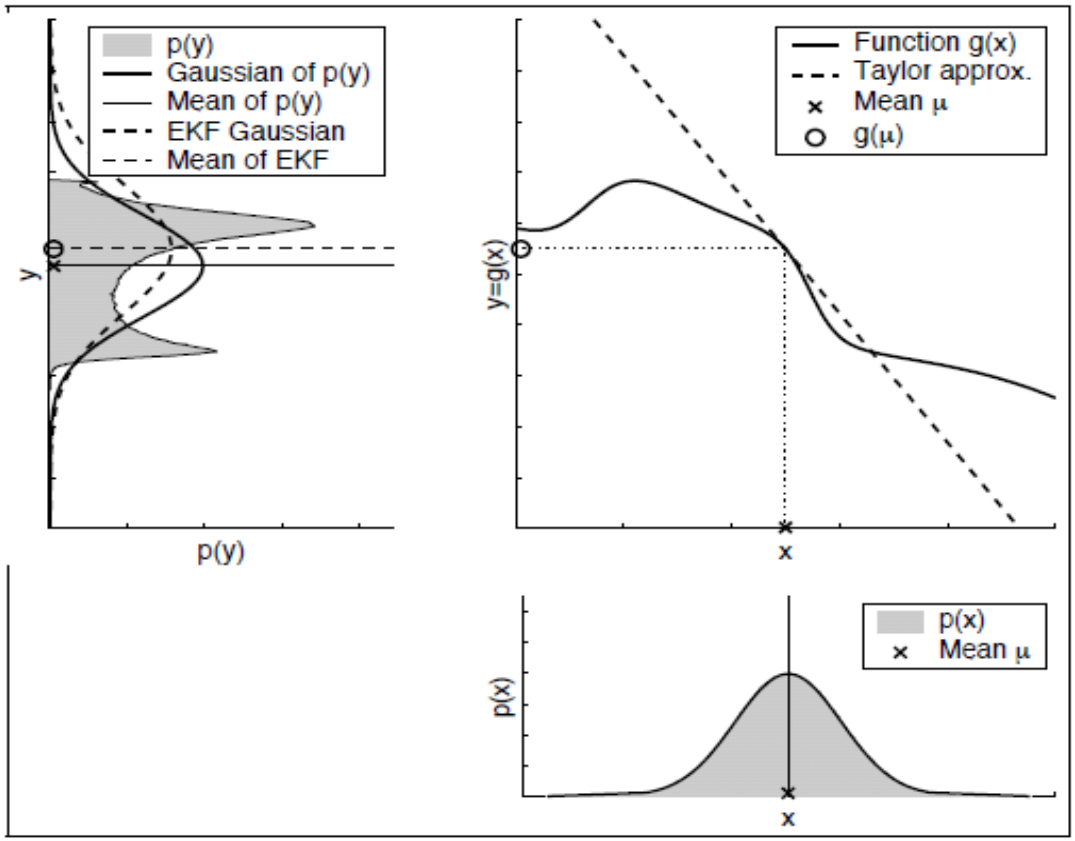
- **Particle/Unscented Kalman Filter**
  - Dynamically sample states with highest probability
  - Monte Carlo approximation



Nonlinear Filtering (EKF)

Extended Kalman filter (EKF)

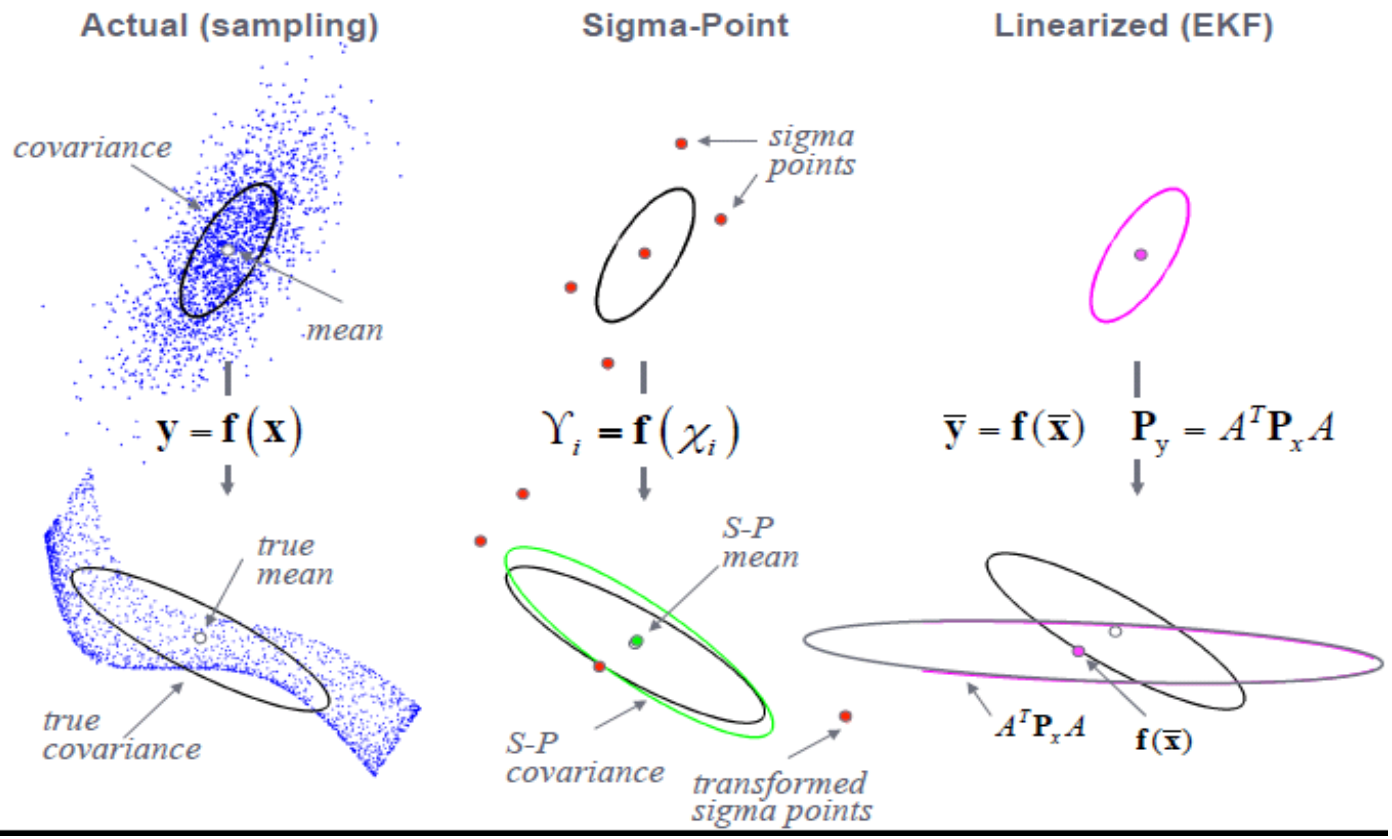
Source: Sebastian Thrun.



Nonlinear Filtering (UKF)

Unscented Kalman filter (UKF)

Source: Eric Wan.



Importance Sampling: Key Idea

$p(\underline{x})$ → true distribution (difficult to sample from)
assume may be evaluated *up to normalization constant*

$q(\underline{x})$ → proposal distribution (easy to sample from)

$$E(f) = \frac{1}{Z_p}\int f(x)\tilde{p}(x)dx = \frac{1}{Z_p}\int f(x)\frac{\tilde{p}(x)}{q(x)}q(x)dx$$

$$\approx \frac{1}{NZ_p}\sum_{i=1}^{N} f(x_i)w_i; x_i \sim q(x); w_i = \frac{\tilde{p}(\underline{x}_i)}{q(\underline{x}_i)}$$

$$Z_p = \int \tilde{p}(x)dx = \int \frac{\tilde{p}(x)}{q(x)}q(x)dx \approx \frac{1}{N}\sum_{i=1}^{N} w_i$$

$$So, E(f) = \sum_{i=1}^{N} f(x_i)\overline{w}_i; \overline{w}_i = \frac{w_i}{\sum_{k=1}^{N} w_k}$$

Importance Sampling

$p(\underline{x})$ ⟶ true distribution (difficult to sample from)
assume may be evaluated *up to normalization constant*

$q(\underline{x})$ ⟶ proposal distribution (easy to sample from)

- Draw N *weighted* samples from proposal distribution

$$\underline{x}_i \sim q(\underline{x}) \qquad w_i = \frac{\tilde{p}(\underline{x}_i)}{q(\underline{x}_i)}; \quad \overline{w}_i = \frac{w_i}{\sum_{k=1}^{N} w_k}$$

- Approximate the target distribution via a weighted mixture of delta functions

$$\hat{p}(\underline{x}) = \sum_{i=1}^{N} \overline{w}_i \delta(\underline{x} - \underline{x}_i) \quad (\text{Note: } \int_{\underline{x}} \hat{p}(\underline{x}) d\underline{x} = 1)$$

- Approaches true distribution as $N \to \infty$. Biased estimate for finite N because the weights are ratios.
- Also, depends on how close $q(\underline{x})$ is to unnormalized $p(\underline{x})$

Resampling

- **Multinomial resampling**
  - To avoid problems with importance sampling, draw samples from a multinomial distribution with probabilities $\{\overline{w_i}\}_{i=1}^{N} \Rightarrow$ Low weight samples are unlikely to be sampled.
  - Let N_i be the samples corresponding to $\underline{x}_i$
  - Approximate the target distribution via a weighted mixture of delta functions

$$\hat{p}(\underline{x}) = \sum_{i=1}^{N} \frac{N_i}{N} \delta(\underline{x} - \underline{x}_i)$$

- **Systematic resampling**
  - Draw a sample u_1 from a uniform distribution [0 , 1/N]
  - Form $u_{j+1} = u_j + \frac{1}{N}; j = 1, 2, .., N\text{-}1$
  - Compute N_i

$$N_i = cardinality\{all\ j : \sum_{k=1}^{i-1} \overline{w}_k \le u_j < \sum_{k=1}^{i} \overline{w}_k\}$$

  - Approximate the target distribution

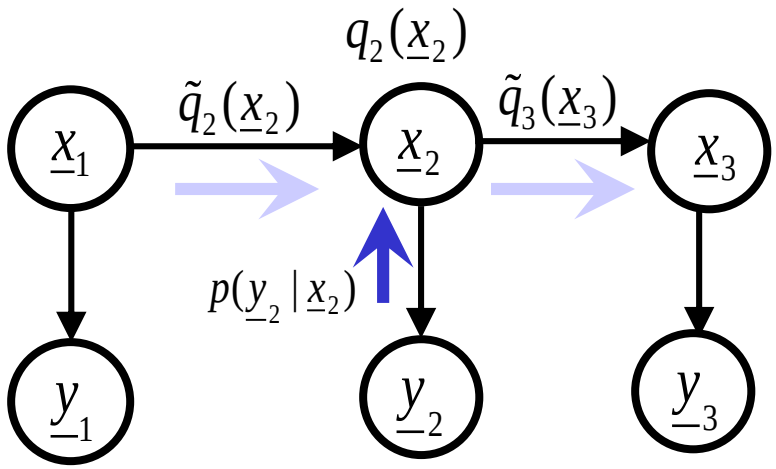
$$\hat{p}(\underline{x}) = \sum_{i=1}^{N} \frac{N_i}{N} \delta(\underline{x} - \underline{x}_i)$$

Resample if

$$N_{eff} = \frac{1}{\sum_{i=1}^{N} \overline{w}_i^{-2}} < \frac{N}{2}$$

Schematic of Particle Filter operation

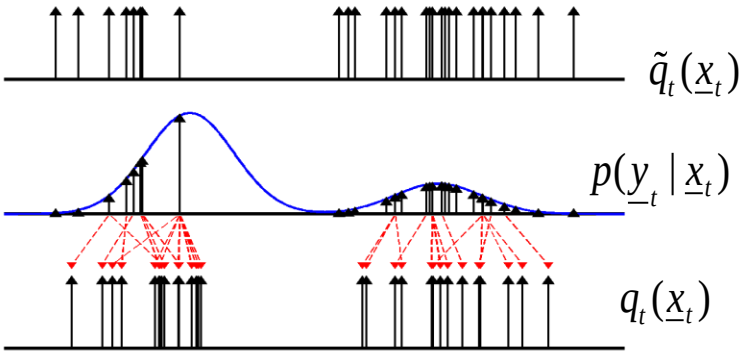
- Represent state estimates using a set of samples
- Dynamics provide proposal distribution for likelihood



Sample-based density estimate

Weight by observation likelihood

Resample & propagate by dynamics



A Generic Particle Filter

- Select the number of particles, N
- Obtain N samples $\{\underline{x}_{0,i}\}_{i=1}^{N}$ from the initial state distribution, and let $w_{0,i} = \frac{1}{N}$

For $t = 1, 2, 3, \ldots$

For $i = 1 : N$

Draw sample $\underline{x}_{t,i}$ from $p(\underline{x}_t \mid \underline{x}_{t-1,i})$

$w_{t,i} = w_{t-1,i}\, p(y_t \mid x_{t,i})$

End

- Normalize the weights. Compute $N_{eff} = \frac{1}{\sum_{i=1}^{N} w_{t,i}^2}$ using normalized weights.

$$\hat{p}(\underline{x}_t \mid \underline{y}_1, \underline{y}_2, .., \underline{y}_t) = \sum_{i=1}^{N} \overline{w}_i(x_{0,i}^t)\delta(\underline{x} - \underline{x}_{i,t})$$

You can compute any metric from this.

- If $N_{eff} \le \frac{N}{2}$

Resample using normalized weights and set $w_{t,i} = \frac{1}{N}$

End

End

After resampling, the weights are set to 1/*N* because by drawing according to the importance weight, all resampled particles are "equally" likely

Lecture Outline

- **Introduction to HMMs**
  - Markov Models
  - Hidden Markov Models
  - Applications of HMMs
- **Inference in HMMs**
  - Filtering, Smoothing, Viterbi, Classification
- **Learning HMM Parameters**
  - EM algorithm (Baum-Welch algorithm)
- **Generalizations of HMMs**