Hidden Markov Models and Graphical Models

CS294: Practical Machine Learning

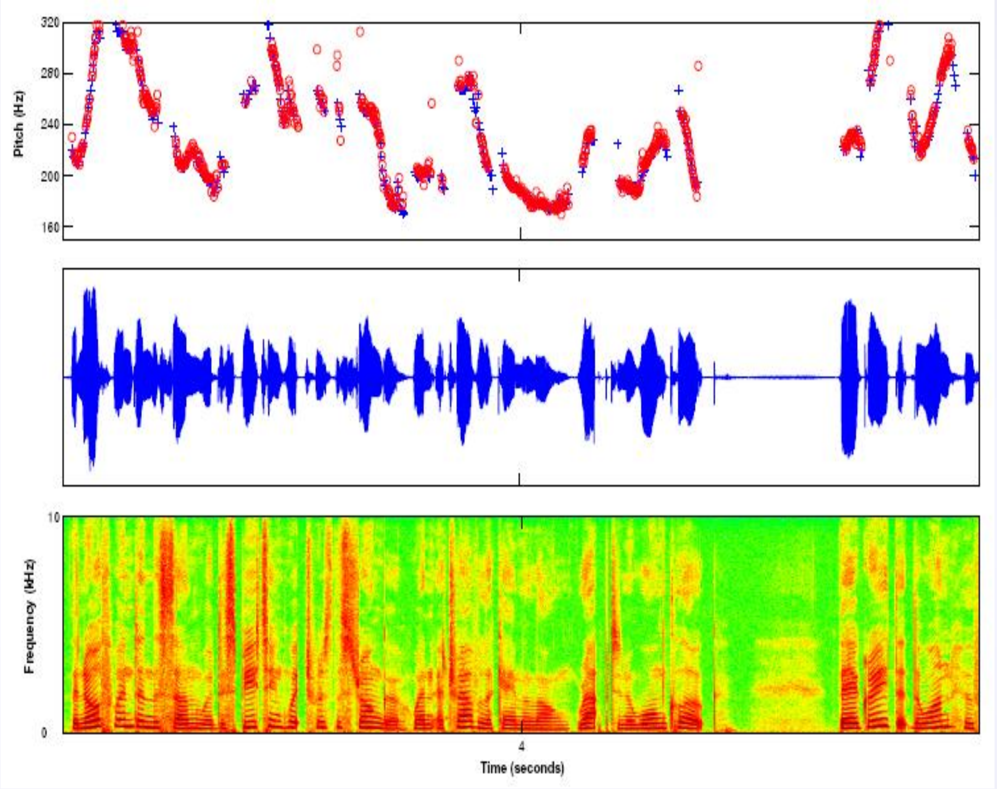
Oct. 8, 2009

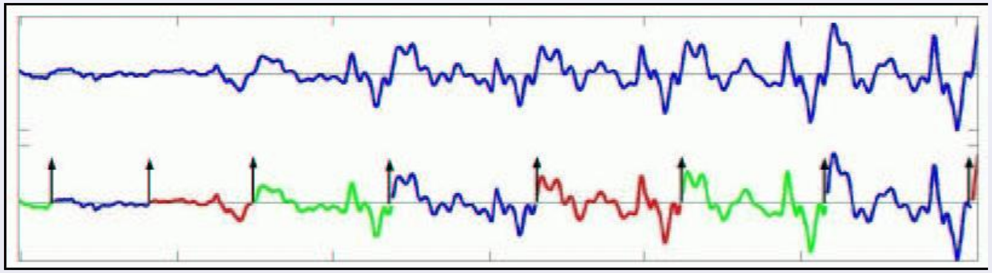
Alex Simma (asimma@eecs)

Based on slides by Erik Sudderth

Speech Recognition

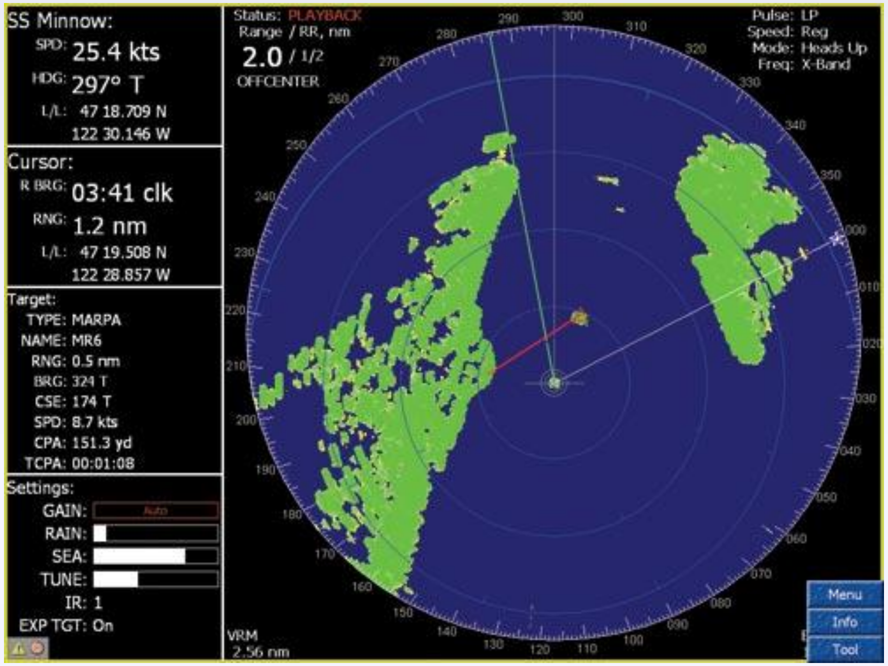
- Given an audio waveform, would like to robustly extract & recognize any spoken words
- Statistical models can be used to
  - ➢ Provide greater robustness to noise
  - ➢ Adapt to accent of different speakers
  - ➢ Learn from training



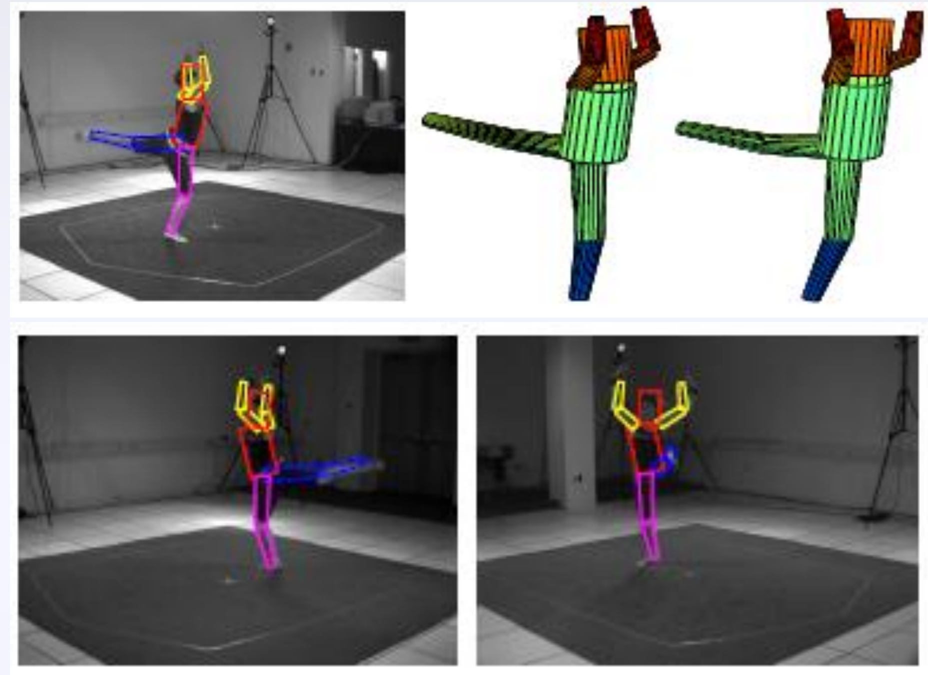


S. Roweis, 2004

Target Tracking



Radar-based tracking of multiple targets



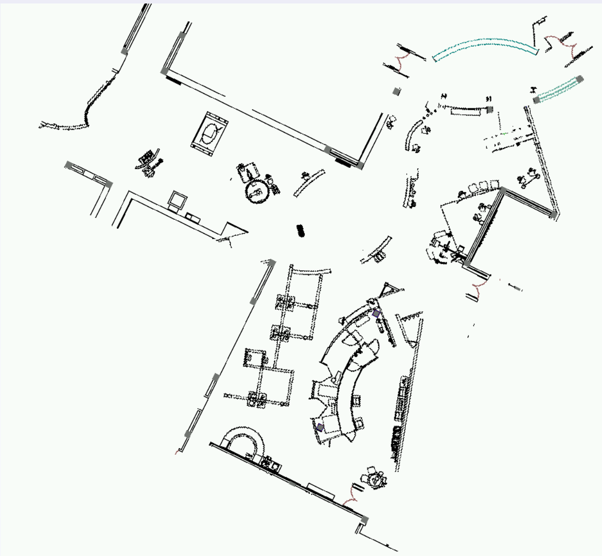
Visual tracking of articulated objects

(L. Sigal et. al., 2006)

- Estimate motion of targets in 3D world from indirect, potentially noisy measurements

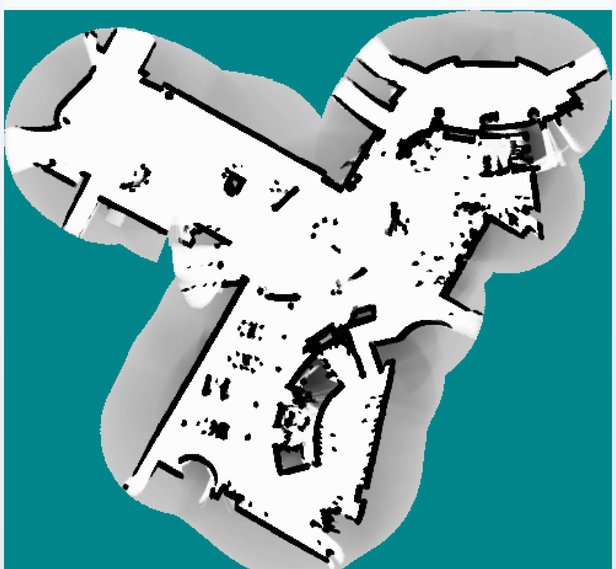
Robot Navigation: *SLAM*

Simultaneous Localization and Mapping



CAD Map

(S. Thrun, San Jose Tech Museum)

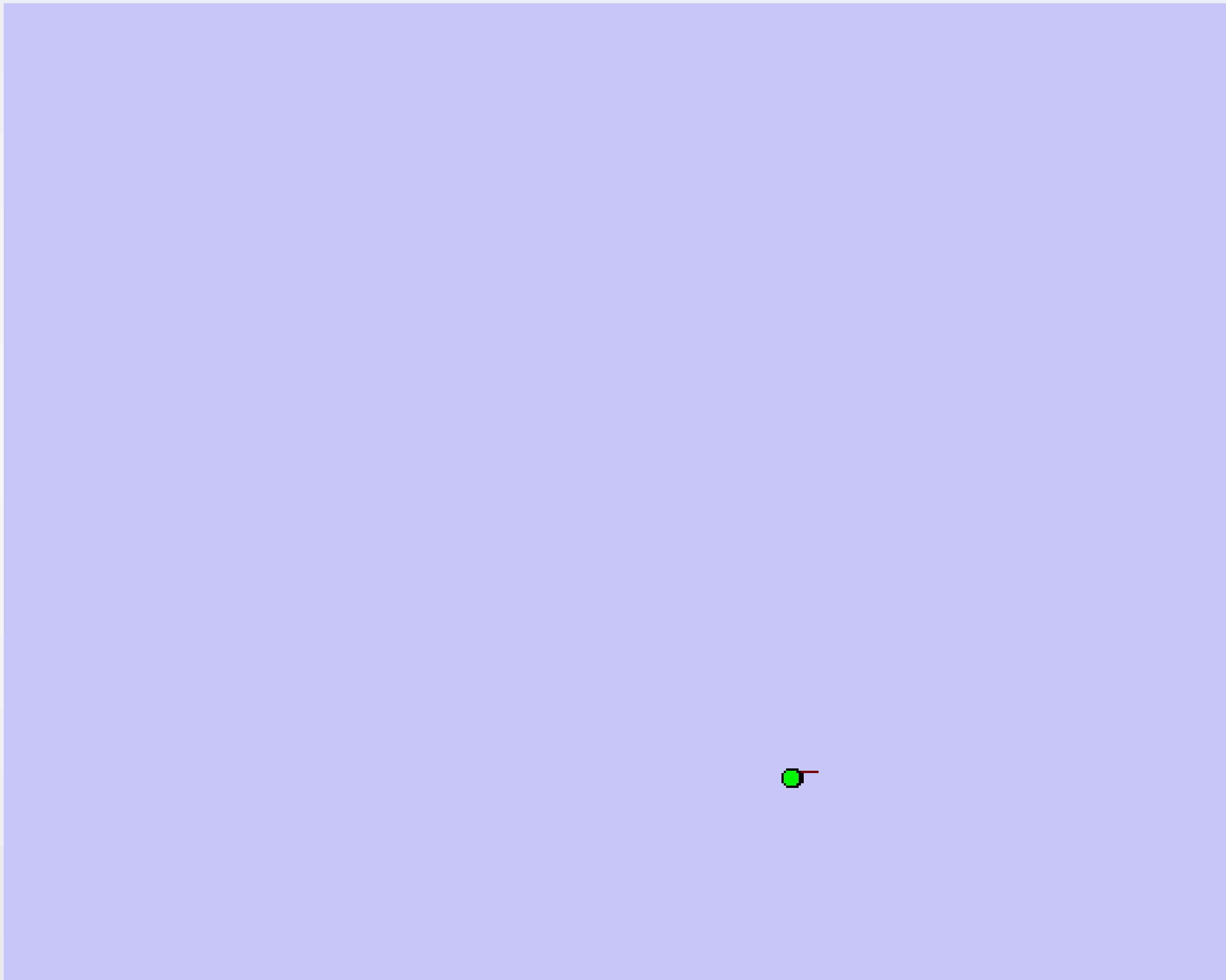


Estimated Map

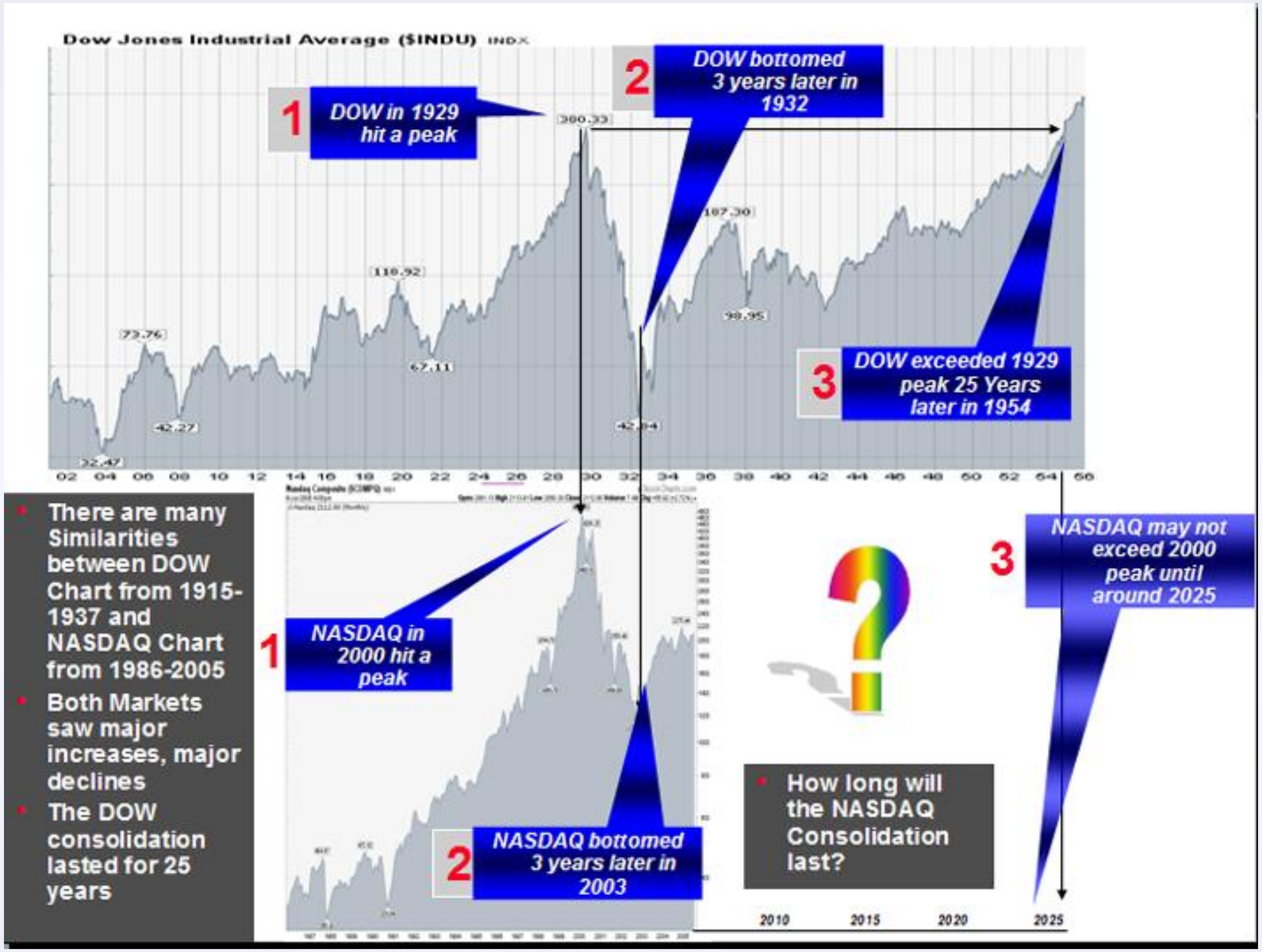
Landmark SLAM

(E. Nebot, Victoria Park)

- As robot moves, estimate its pose & world geometry



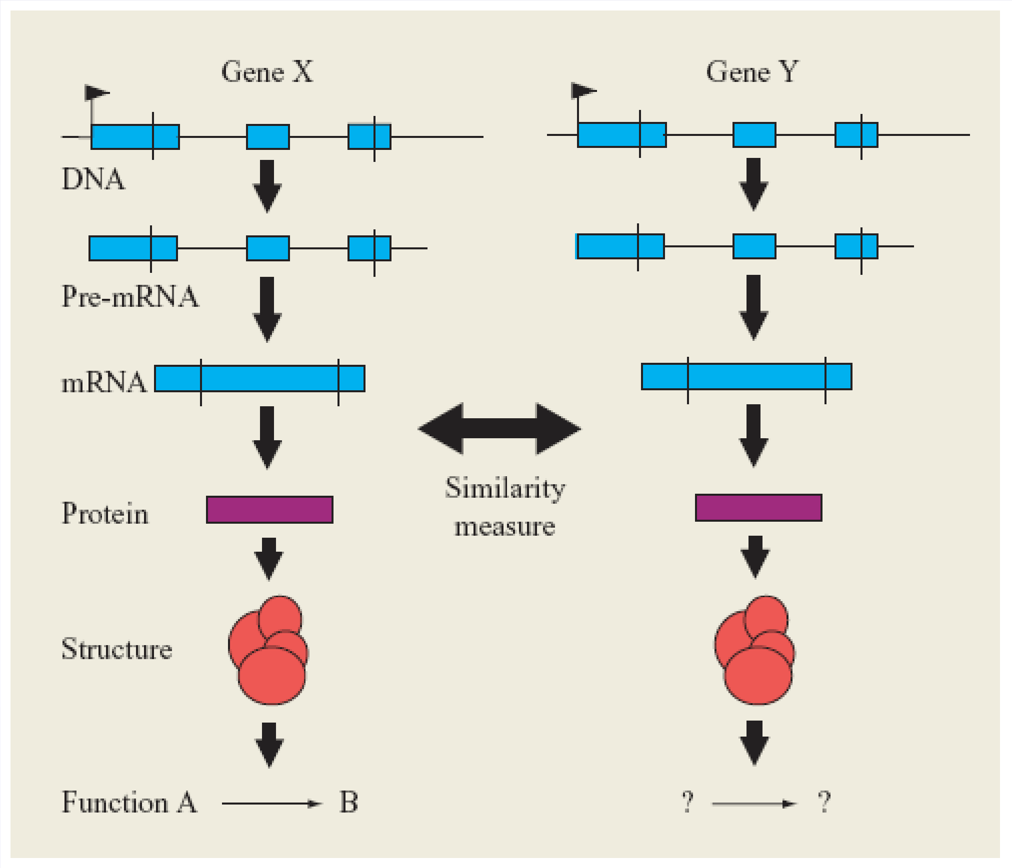
Financial Forecasting



http://www.steadfastinvestor.com/

- Predict future market behavior from historical data, news reports, expert opinions, …

Biological Sequence Analysis



(E. Birney, 2001)

- Temporal models can be adapted to exploit more general forms of *sequential* structure, like those arising in DNA sequences

Analysis of Sequential Data

- Sequential structure arises in a huge 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

Outline

Introduction to Sequential Processes

- Markov chains
- Hidden Markov models

Discrete-State HMMs

- Inference: Filtering, smoothing, Viterbi, classification
- Learning: EM algorithm

Continuous-State HMMs

- Linear state space models: Kalman filters
- Nonlinear dynamical systems: Particle filters

More on Graphical Models

Sequential Processes

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

  $x_t \in \{1, 2, \ldots, N\} \longrightarrow$ 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:

$$p(x_0, x_1, \ldots, x_T) = p(x_0) \prod_{t=1}^{T} p(x_t \mid x_0, \ldots, x_{t-1})$$

Markov Processes

- For a *Markov* process, the next state depends only on the current state:

$$p(x_{t+1} \mid x_0, \ldots, x_t) = p(x_{t+1} \mid x_t)$$

- This property in turn implies that

$$p(x_0, \ldots, x_{t-1}, x_{t+1}, \ldots, x_T \mid x_t)$$
$$= p(x_0, \ldots, x_{t-1} \mid x_t) p(x_{t+1}, \ldots, x_T \mid x_t)$$

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

State Transition Matrices

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

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

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

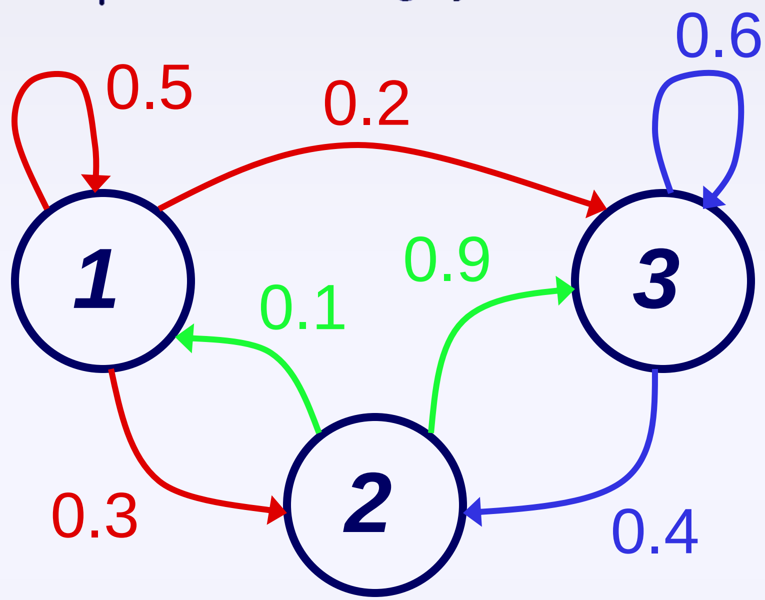
- Constraints on valid transition matrices:

$$q_{ij} \geq 0 \qquad \sum_{i=1}^{N} q_{ij} = 1 \quad \text{for all } j$$

State Transition Diagrams

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

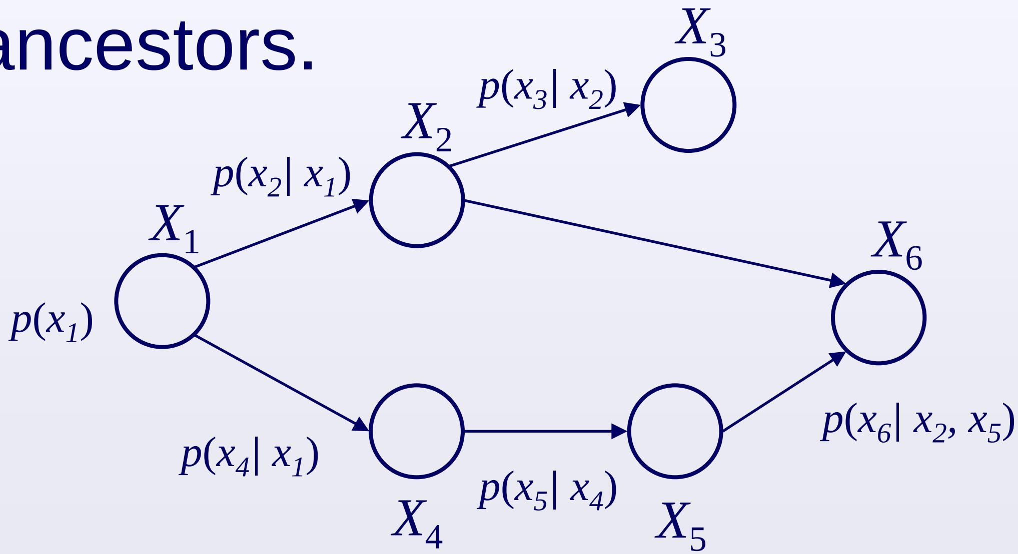
$$Q = \begin{bmatrix} 0.5 & 0.1 & 0.0 \\ 0.3 & 0.0 & 0.4 \\ 0.2 & 0.9 & 0.6 \end{bmatrix}$$



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

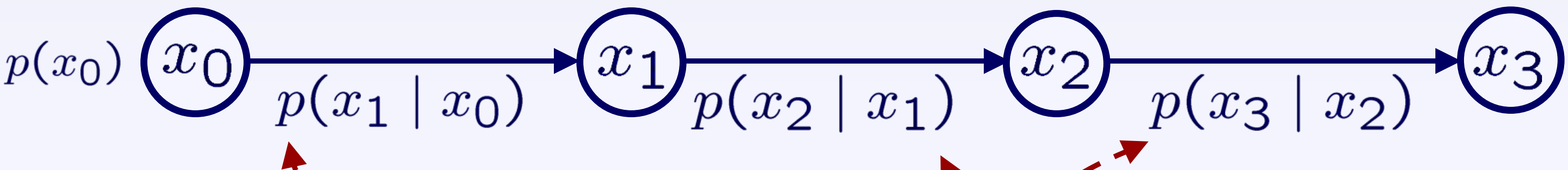
Graphical Models – A Quick Intro

- A way of specifying conditional independences.
- Directed Graphical Modes: a DAG
- Nodes are random variables.
- A node's distribution depends on its parents.
- Joint distribution: $p(x) = \Pi_i p(x_i | \text{Parents}_i)$
- A node's value conditional on its parents is independent of other ancestors.



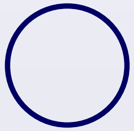
Markov Chains: Graphical Models

$$p(x_0, x_1, \ldots, x_T) = p(x_0) \prod_{t=1}^{T} p(x_t \mid x_{t-1})$$



$$Q = \begin{bmatrix} 0.5 & 0.1 & 0.0 \\ 0.3 & 0.0 & 0.4 \\ 0.2 & 0.9 & 0.6 \end{bmatrix}$$

- Graph interpretation differs from state transition diagrams:

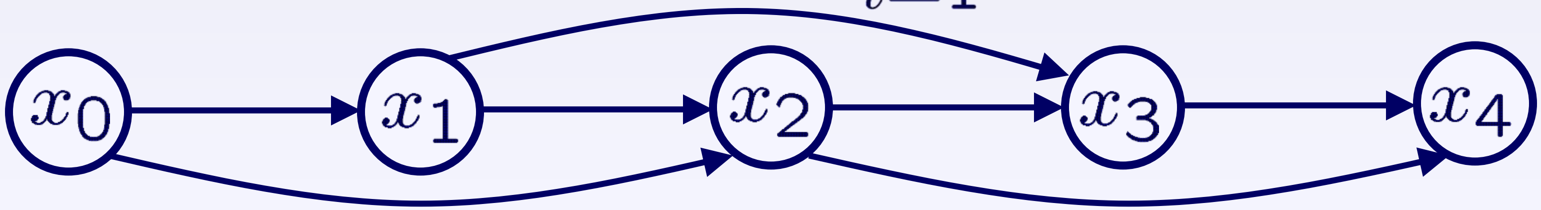


nodes $\Longleftrightarrow$ state values at particular times

edges $\Longleftrightarrow$ Markov properties

Embedding Higher-Order Chains

$$p(x_0, x_1, \ldots, x_T) = p(x_0) \prod_{t=1}^{T} p(x_t \mid x_{t-1}, x_{t-2})$$

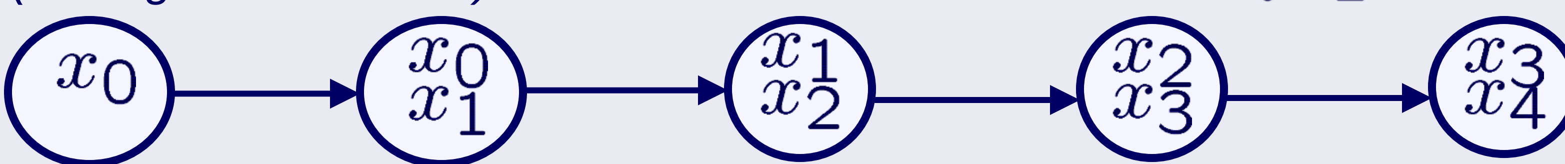


- Each new state depends on fixed-length *window* of preceding state values
- We can represent this as a first-order model via *state augmentation*:

$$\bar{x}_t \triangleq \{x_t, x_{t-1}\}$$

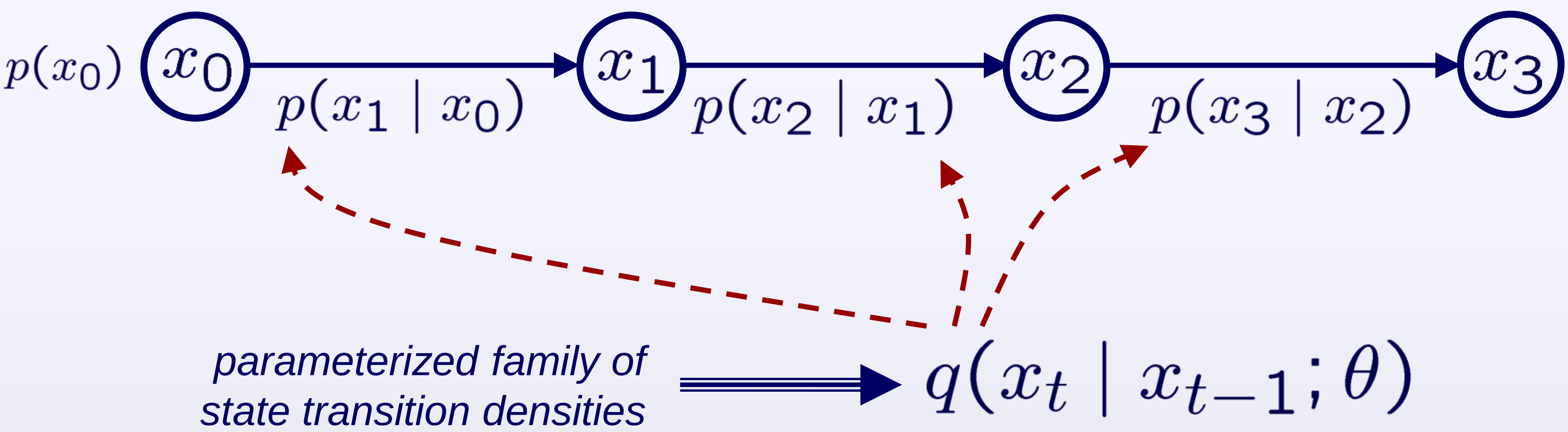
(N^2 augmented states)

$$p(\bar{x}) = p(\bar{x}_1) \prod_{t=2}^{T} p(\bar{x}_t \mid \bar{x}_{t-1})$$



Continuous State Processes

- In many applications, it is more natural to define states in some *continuous*, Euclidean space: $x_t \in \mathbb{R}^d$



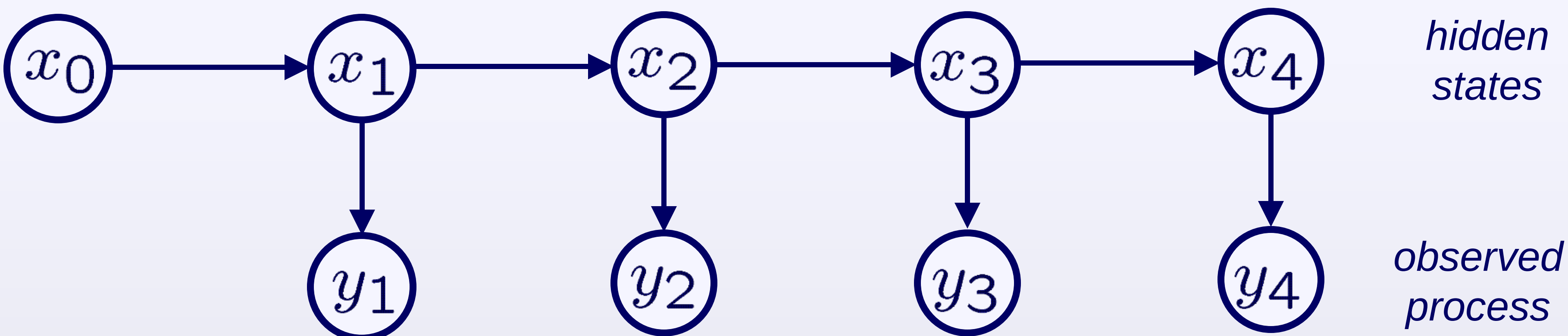
- *Examples:* stock price, aircraft position, …

Hidden Markov Models

- Few realistic time series directly satisfy the assumptions of Markov processes:

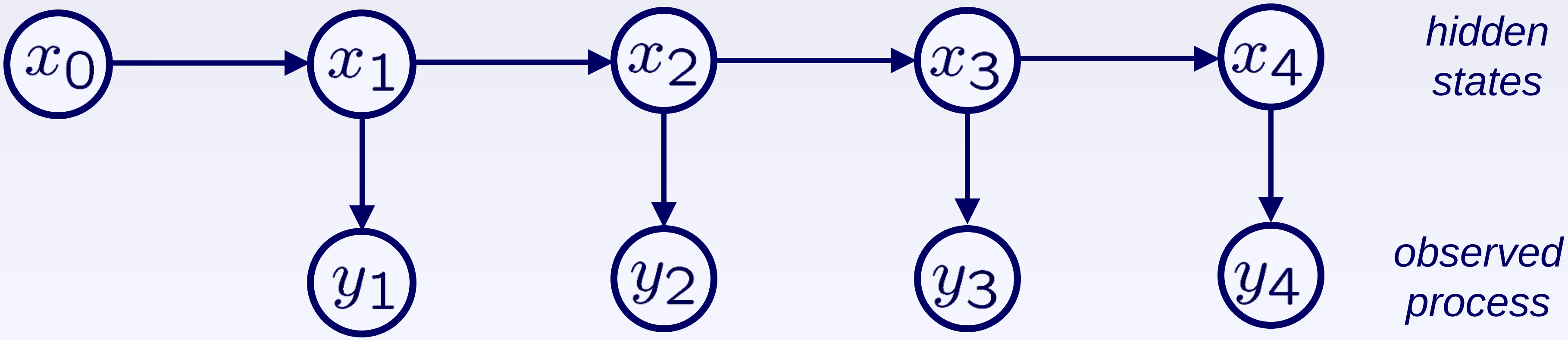
"Conditioned on the present,
the past & future are independent"

- Motivates *hidden Markov models (HMM)*:



$$p(x_0, x_1, \ldots, x_T) = p(x_0) \prod_{t=1}^{T} p(x_t \mid x_{t-1}) p(y_t \mid x_t)$$

Hidden states

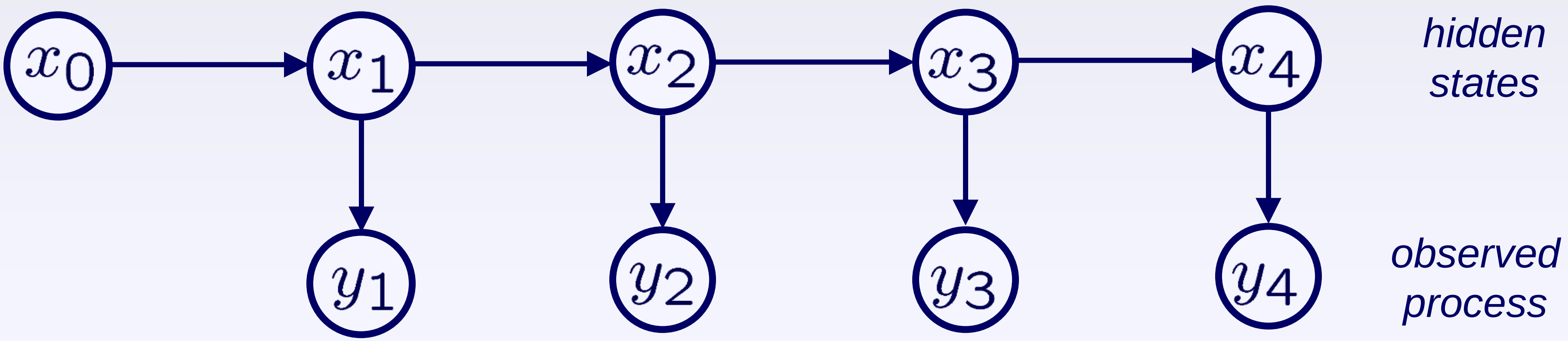


- Given x_t, earlier observations provide no *additional information* about the future:

$$p(y_t, y_{t+1}, \cdots \mid x_t, y_{t-1}, y_{t-2}, \ldots) = p(y_t, y_{t+1}, \cdots \mid x_t)$$

- Transformation of process under which dynamics take a *simple*, first-order form

Where do states come from?



- Analysis of a *physical phenomenon*:
  - Dynamical models of an aircraft or robot
  - Geophysical models of climate evolution
- Discovered from *training data*:
  - Recorded examples of spoken English
  - Historic behavior of stock prices

Outline

Introduction to Sequential Processes

- Markov chains
- Hidden Markov models

Discrete-State HMMs

- Inference: Filtering, smoothing, Viterbi, classification
- Learning: EM algorithm

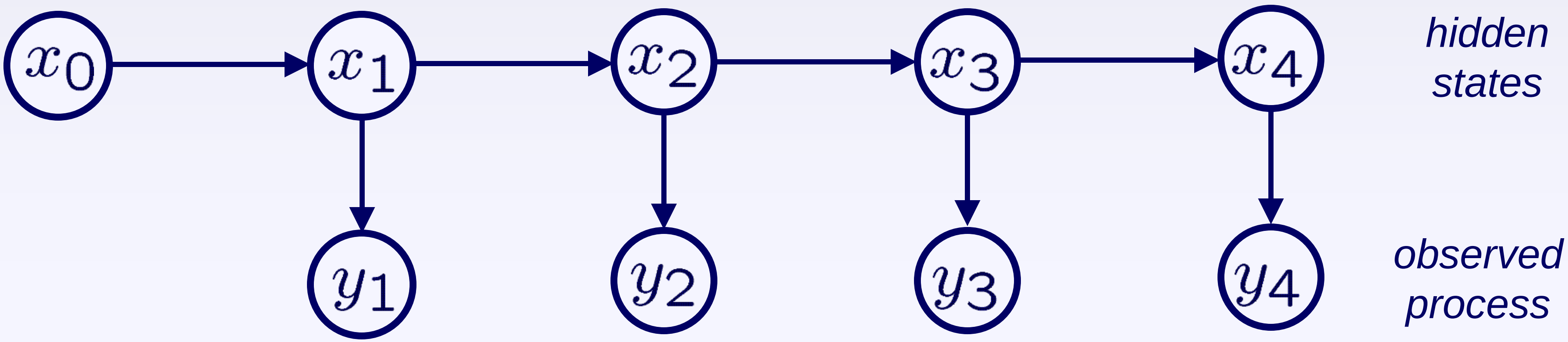
Continuous-State HMMs

- Linear state space models: Kalman filters
- Nonlinear dynamical systems: Particle filters

More on Graphical Models

Discrete State HMMs

$$x_t \in \{1, 2, \ldots, N\}$$



- Associate each of the *N* hidden states with a different observation distribution:

$$p(y_t \mid x_t = 1) \qquad p(y_t \mid x_t = 2) \quad \cdots$$

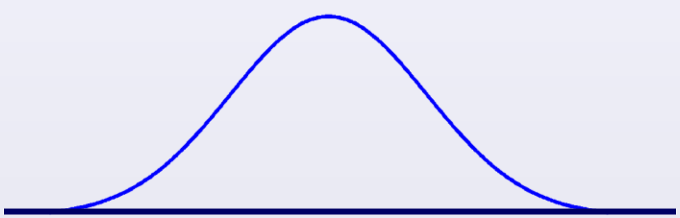
- Observation densities are typically chosen to encode domain knowledge

Discrete HMMs: Observations

Discrete Observations $y_t \in \{1, 2, \ldots, M\}$

$$p(y_t \mid x_t = 1) = \begin{bmatrix} 0.3 \\ 0.1 \\ 0.5 \\ 0.1 \end{bmatrix} \qquad p(y_t \mid x_t = 2) = \begin{bmatrix} 0.2 \\ 0.2 \\ 0.1 \\ 0.5 \end{bmatrix}$$

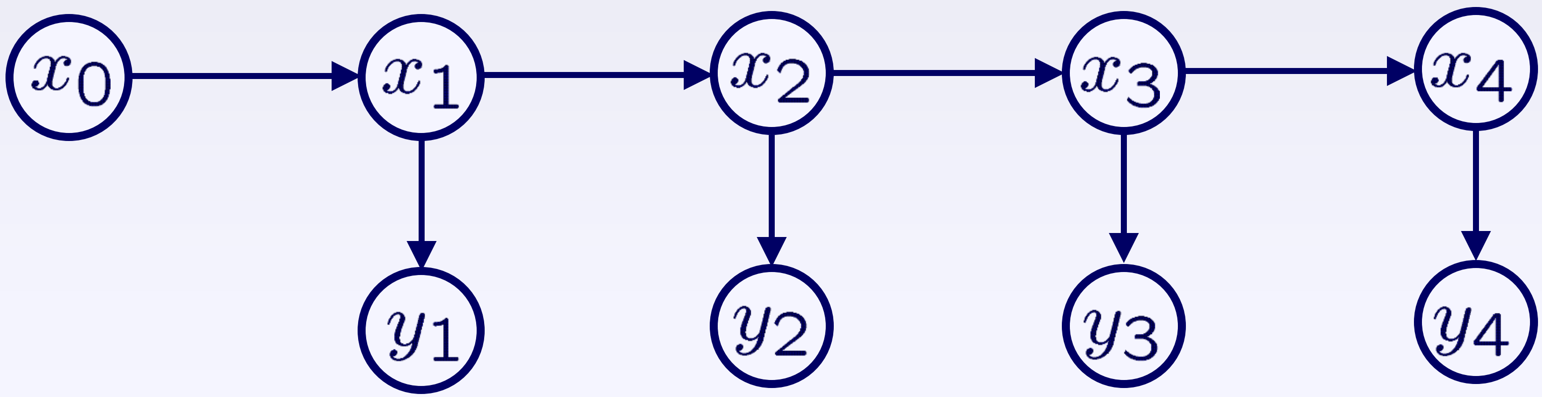
Continuous Observations $y_t \in \mathbb{R}^k$



$p(y_t \mid x_t = 1)$

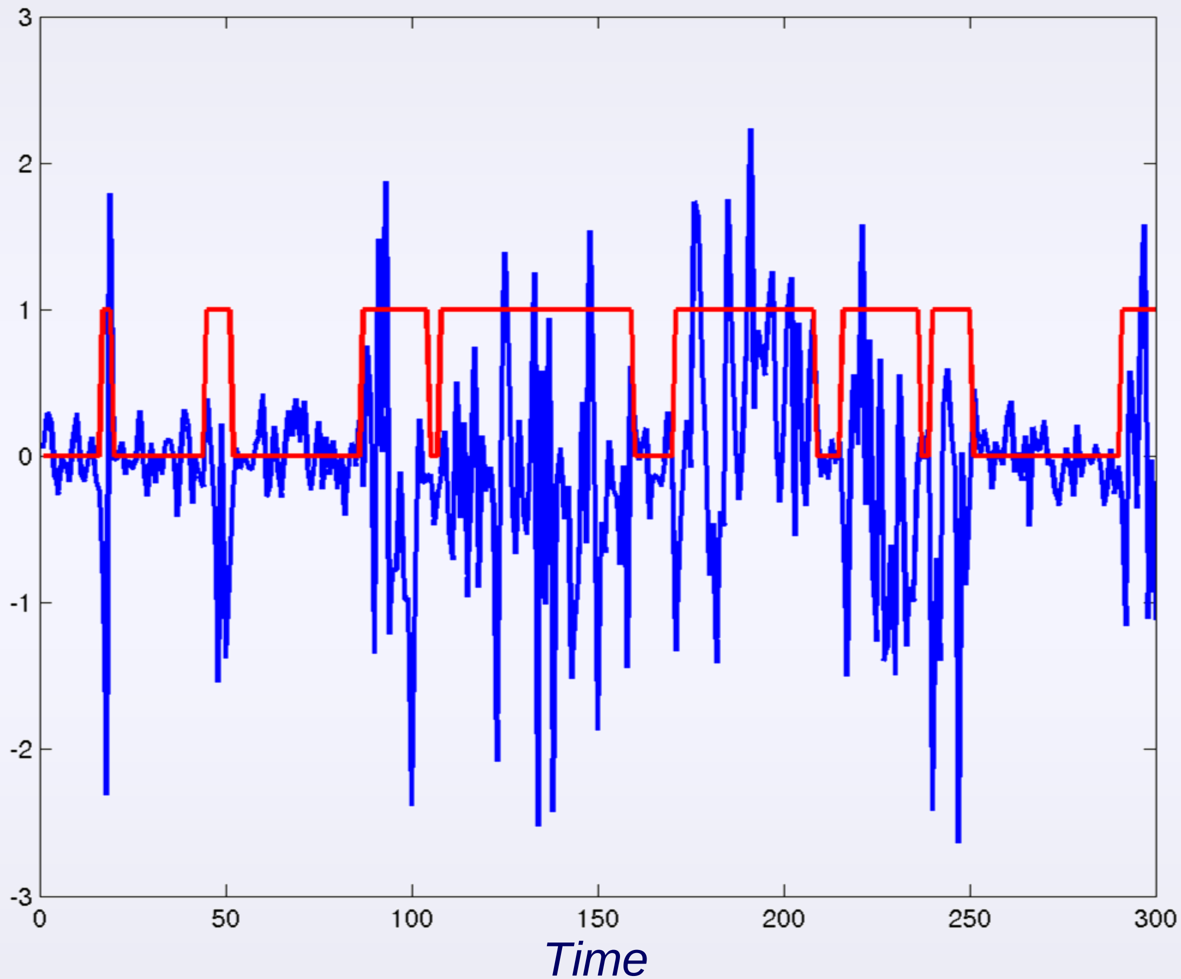
$p(y_t \mid x_t = 2)$

Specifying an HMM



- Observation model: $P(y_i|x_i)$
- Transition model: $P(x_i|x_{i-1})$
- Initial state distribution: $P(x_0)$

Gilbert-Elliott Channel Model



Hidden State:

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

$$Q = \begin{bmatrix} 1-\epsilon & \epsilon \\ \epsilon & 1-\epsilon \end{bmatrix}$$

Observations:

$$y_t \sim \mathcal{N}(0, \sigma_{x_t}^2)$$

$\sigma_0^2 =$ *small*

$\sigma_1^2 =$ *large*

Simple model for correlated, bursty noise

(Elliott, 1963)

Discrete HMMs: Inference

- In many applications, we would like to *infer* hidden states from observations
- Suppose that the *cost* incurred by an estimated state sequence *decomposes*:

$$C(x, \hat{x}) = \sum_{t=0}^{T} C_t(x_t, \hat{x}_t)$$

$x_t \longrightarrow$ *true state*

$\hat{x}_t \longrightarrow$ *estimated state*

- The *expected cost* then depends only on the *posterior marginal* distributions:

$$\mathbb{E}[C(x, \hat{x}) \mid y] = \sum_{t=0}^{T} \sum_{x_t} C_t(x_t, \hat{x}_t) p(x_t \mid y)$$

Filtering & Smoothing

- For online data analysis, we seek *filtered* state estimates given earlier observations:

$$p(x_t \mid y_1, y_2, \ldots, y_t) \qquad t = 1, 2, \ldots$$

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

$$p(x_t \mid y_1, y_2, \ldots, y_T) \qquad t = 1, 2, \ldots, T$$

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

$$p(x_t \mid y_1, \ldots, y_{t+\delta}) \qquad p(x_t \mid y_1, \ldots, y_{t-\delta})$$

Markov Chain Statistics

$$
\begin{tikzcd}
p(x_0)\; x_0 \arrow[r, "p(x_1 \mid x_0)"'] & x_1 \arrow[r, "p(x_2 \mid x_1)"'] & x_2 \arrow[r, "p(x_3 \mid x_2)"'] & x_3
\end{tikzcd}
$$

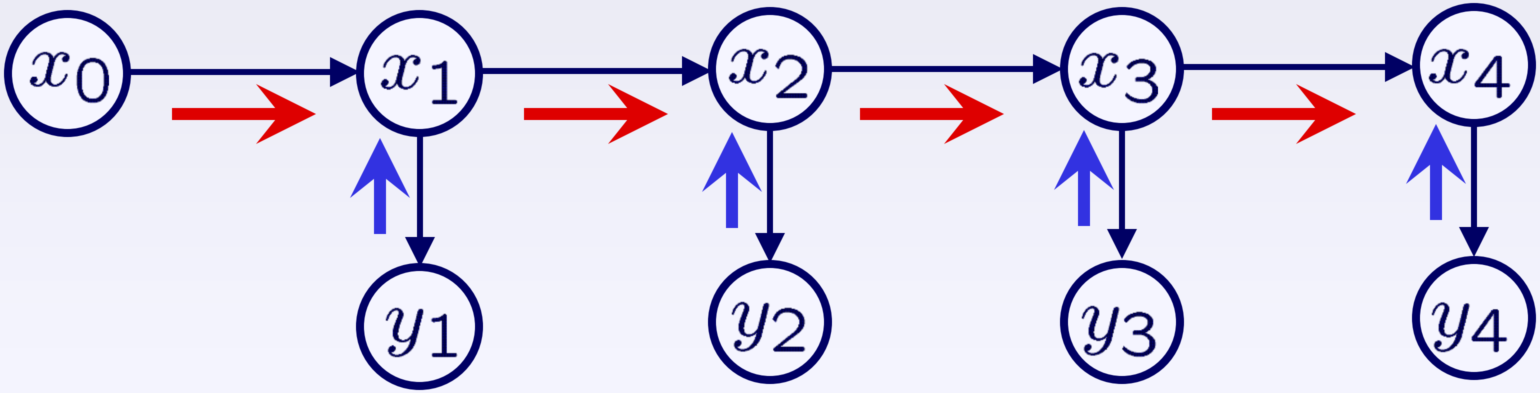
$$
\alpha_t \triangleq \begin{bmatrix} p(x_t = 1) \\ p(x_t = 2) \\ p(x_t = 3) \end{bmatrix}
\qquad
Q = \begin{bmatrix} q_{11} & q_{12} & q_{13} \\ q_{21} & q_{22} & q_{23} \\ q_{31} & q_{32} & q_{33} \end{bmatrix}
$$

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

- By definition of conditional probabilities,

$$
\alpha_1(i) = \sum_{j=1}^{N} q_{ij}\alpha_0(j)
\qquad
\left\{
\begin{aligned}
\alpha_1 &= Q\alpha_0 \\
\alpha_t &= Q^t\alpha_0 \\
\alpha_\infty &= ???
\end{aligned}
\right.
$$

Discrete HMMs: Filtering



$$
\begin{aligned}
\alpha_t(x_t) &\triangleq p(x_t \mid y_1, \ldots, y_t) \\
&= \frac{1}{Z_t} p(y_t \mid x_t) \sum_{x_{t-1}} p(x_t \mid x_{t-1}) \alpha_{t-1}(x_{t-1})
\end{aligned}
$$

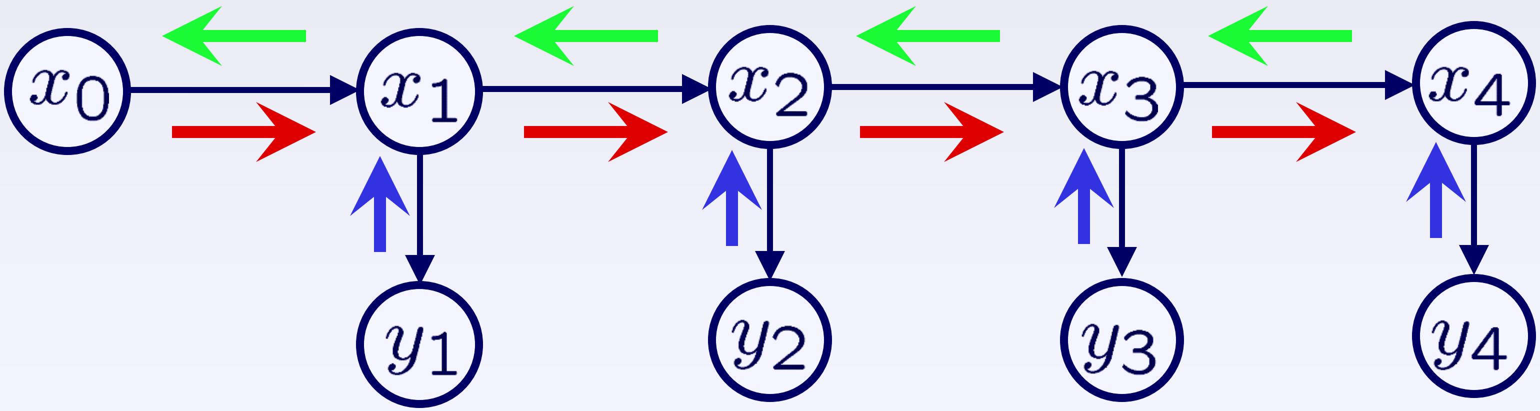
Normalization constant

Prediction: $p(x_t \mid y_1, \ldots, y_{t-1})$

Update: $p(x_t \mid y_1, \ldots, y_t)$

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

Discrete HMMs: Smoothing



$$p(x_t \mid y) \propto \underbrace{p(x_t \mid y_1, \ldots, y_t)}_{\alpha_t(x_t)} \underbrace{p(y_{t+1}, \ldots, y_T \mid x_t)}_{\beta_t(x_t)}$$

- The *forward-backward* algorithm updates filtering via a *reverse-time* recursion:

$$\beta_t(x_t) = \frac{1}{Z_t} \sum_{x_{t+1}} p(x_{t+1} \mid x_t) p(y_{t+1} \mid x_{t+1}) \beta_{t+1}(x_{t+1})$$

Optimal State Estimation

$$p(x_t \mid y) = \frac{1}{Z_t}\alpha_t(x_t)\beta_t(x_t)$$

- Probabilities measure the posterior *confidence* in the true hidden states
- The *posterior mode* minimizes the number of incorrectly assigned states:

$$C(x, \hat{x}) = T - \sum_{t=1}^{T} \delta(x_t, \hat{x}_t)$$

Bit or symbol error rate

- What about the state *sequence* with the highest *joint* probability?

Word or sequence error rate

Viterbi Algorithm

$$\hat{x} = \arg\max_{x} \; p(x_0, x_1, \ldots, x_T \mid y_1, \ldots, y_T)$$

- Use *dynamic programming* to recursively find the probability of the most likely state sequence ending with each $x_t \in \{1, \ldots, N\}$

$$\gamma_t(x_t) \triangleq \max_{x_1,\ldots,x_{t-1}} p(x_1, \ldots, x_{t-1}, x_t \mid y_1, \ldots, y_t)$$

$$\propto p(y_t \mid x_t) \cdot \left[\max_{x_{t-1}} \; p(x_t \mid x_{t-1}) \gamma_{t-1}(x_{t-1}) \right]$$

- A reverse-time, *backtracking* procedure then picks the maximizing state sequence

Time Series Classification

- Suppose I'd like to know which of 2 HMMs best explains an observed sequence
- This *classification* is optimally determined by the following log-likelihood ratio:

$$\log \frac{p(y_1, \ldots, y_T \mid \mathcal{M}_1)}{p(y_1, \ldots, y_T \mid \mathcal{M}_0)} = \log \frac{p(y \mid \mathcal{M}_1)}{p(y \mid \mathcal{M}_0)}$$

$$\log p(y \mid \mathcal{M}_i) = \log \sum_x p(y \mid x, \mathcal{M}_i) p(x \mid \mathcal{M}_i)$$

- These log-likelihoods can be computed from filtering *normalization constants*

Discrete HMMs: Learning I

- Suppose first that the latent state sequence is available during training
- The *maximum likelihood* estimate equals

$$(\widehat{Q}, \widehat{\theta}) = \arg\max_{Q,\theta} \; p(x \mid Q) p(y \mid x, \theta)$$

$$Q = \left[q_{ij} \right] = \left[p(x_{t+1} = i \mid x_t = j) \right]$$

$$\theta = \{\theta_i\}_{i=1}^{N}$$ *(observation distributions)*

- For simplicity, assume observations are *Gaussian* with known variance & mean θ_i

Discrete HMMs: Learning II

- The ML estimate of the transition matrix is determined by *normalized counts*:

$$n(i,j) \triangleq \begin{cases} \text{number of times} & x_t = j \\ \text{observed before} & x_{t+1} = i \end{cases}$$

$$\hat{q}_{ij} = \frac{n(i,j)}{\sum_k n(k,j)}$$

- Given x, *independently* estimate the output distribution for each state:

$$\hat{\theta}_i = \frac{1}{|\tau_i|} \sum_{t \in \tau_i} y_t \qquad \tau_i \triangleq \{t \mid x_t = i\}$$

Discrete HMMs: EM Algorithm

- In practice, we typically don't know the hidden states for our training sequences
- The *EM algorithm* iteratively maximizes a *lower bound* on the true data likelihood:

E-Step: Use current parameters to *estimate* state

$$\widehat{p}^{(s)}(x) = p(x \mid y, \widehat{Q}^{(s-1)}, \widehat{\theta}^{(s-1)})$$

M-Step: Use *soft* state estimates to update parameters

$$(\widehat{Q}^{(s)}, \widehat{\theta}^{(s)}) = \arg\max_{Q,\theta} \sum_{x} \widehat{p}^{(s)}(x) \log p(x \mid Q) p(y \mid x, \theta)$$

Applied to HMMs, equivalent to the Baum-Welch algorithm

Discrete HMMs: EM Algorithm

- Due to Markov structure, EM parameter updates use local statistics, computed by the *forward-backward* algorithm *(E-step)*
- The *M-step* then estimates observation distributions via a weighted average:

$$\hat{\theta}_i^{(s)} = \frac{\sum_t \hat{p}^{(s)}(x_t = i) y_t}{\sum_t \hat{p}^{(s)}(x_t = i)}$$

- Transition matrices estimated similarly…
- May encounter *local minima*; initialization important.

Outline

Introduction to Sequential Processes

- Markov chains
- Hidden Markov models

Discrete-State HMMs

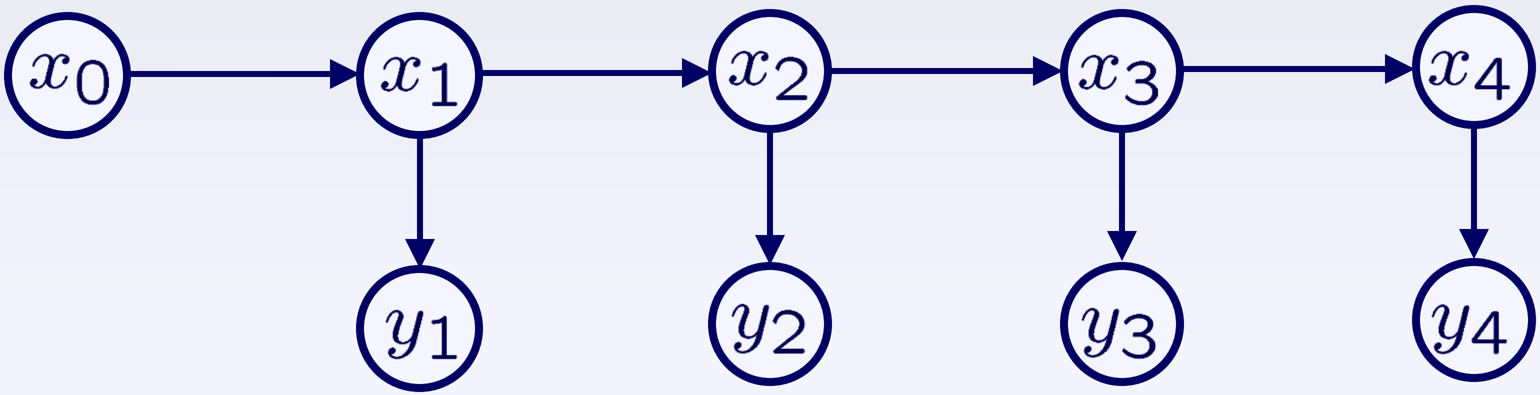
- Inference: Filtering, smoothing, Viterbi, classification
- Learning: EM algorithm

Continuous-State HMMs

- Linear state space models: Kalman filters
- Nonlinear dynamical systems: Particle filters

More on Graphical Models

Linear State Space Models



$x_t \in \mathbb{R}^d$

$y_t \in \mathbb{R}^k$

$$x_{t+1} = Ax_t + w_t \qquad w_t \sim \mathcal{N}(0, Q)$$

$$y_t = Cx_t + v_t \qquad v_t \sim \mathcal{N}(0, R)$$

- States & observations jointly Gaussian:
  - All marginals & conditionals Gaussian
  - Linear transformations remain Gaussian

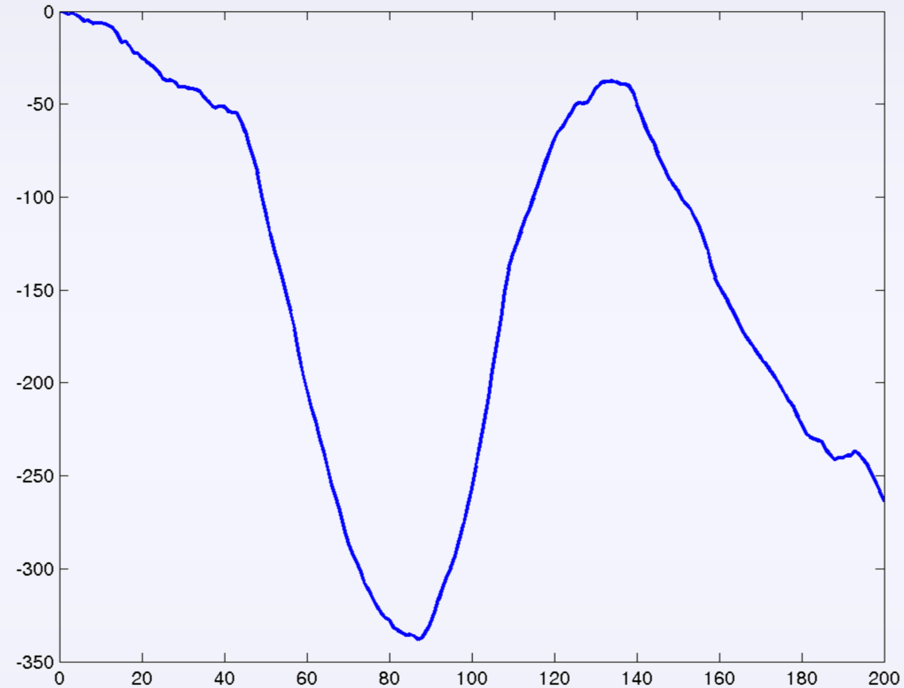
Simple Linear Dynamics

Brownian Motion

Time

$$x_{t+1} = x_t + w_t$$

Constant Velocity



Time

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

Kalman Filter

$$x_{t+1} = Ax_t + w_t \qquad w_t \sim \mathcal{N}(0, Q)$$
$$y_t = Cx_t + v_t \qquad v_t \sim \mathcal{N}(0, R)$$

- Represent Gaussians by *mean* & *covariance*:

$$p(x_t \mid y_1, \ldots, y_{t-1}) = \mathcal{N}(x; \tilde{\mu}_t, \tilde{\Lambda}_t)$$
$$p(x_t \mid y_1, \ldots, y_t) = \mathcal{N}(x; \mu_t, \Lambda_t)$$

Prediction:

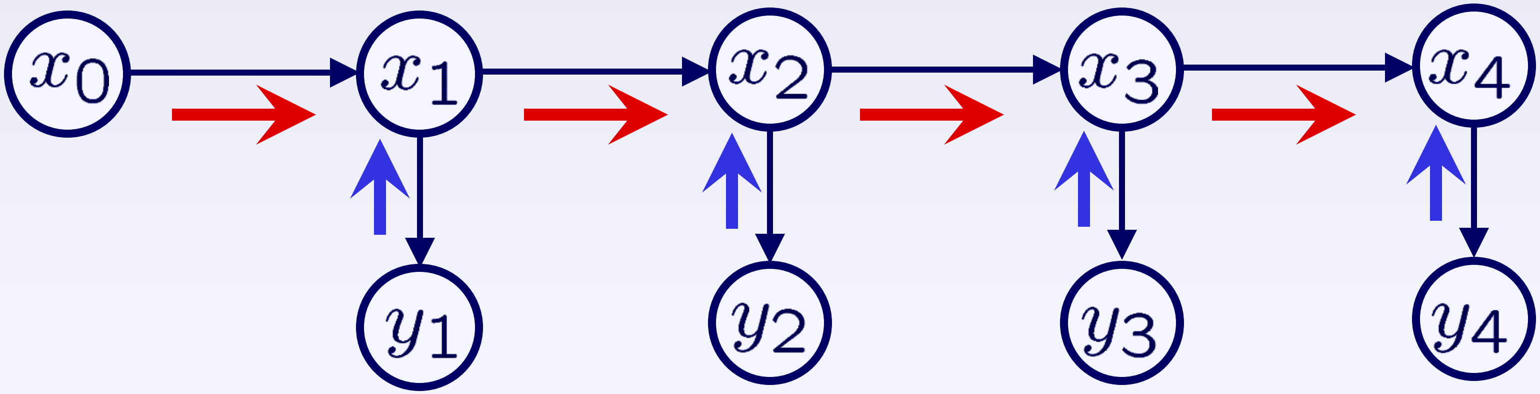
$$\boxed{\begin{aligned} \tilde{\mu}_t &= A\mu_{t-1} \\ \tilde{\Lambda}_t &= A\Lambda_{t-1}A^T + Q \end{aligned}}$$

Kalman Gain: $K_t = \tilde{\Lambda}_t C^T (C\tilde{\Lambda}_t C^T + R)^{-1}$

Update:

$$\boxed{\begin{aligned} \mu_t &= \tilde{\mu}_t + K_t(y_t - C\tilde{\mu}_t) \\ \Lambda_t &= \tilde{\Lambda}_t - K_t C\tilde{\Lambda}_t \end{aligned}}$$

Kalman Filtering as Regression



$$p(x_t \mid y_1, \ldots, y_t) = \mathcal{N}(x; \mu_t, \Lambda_t)$$

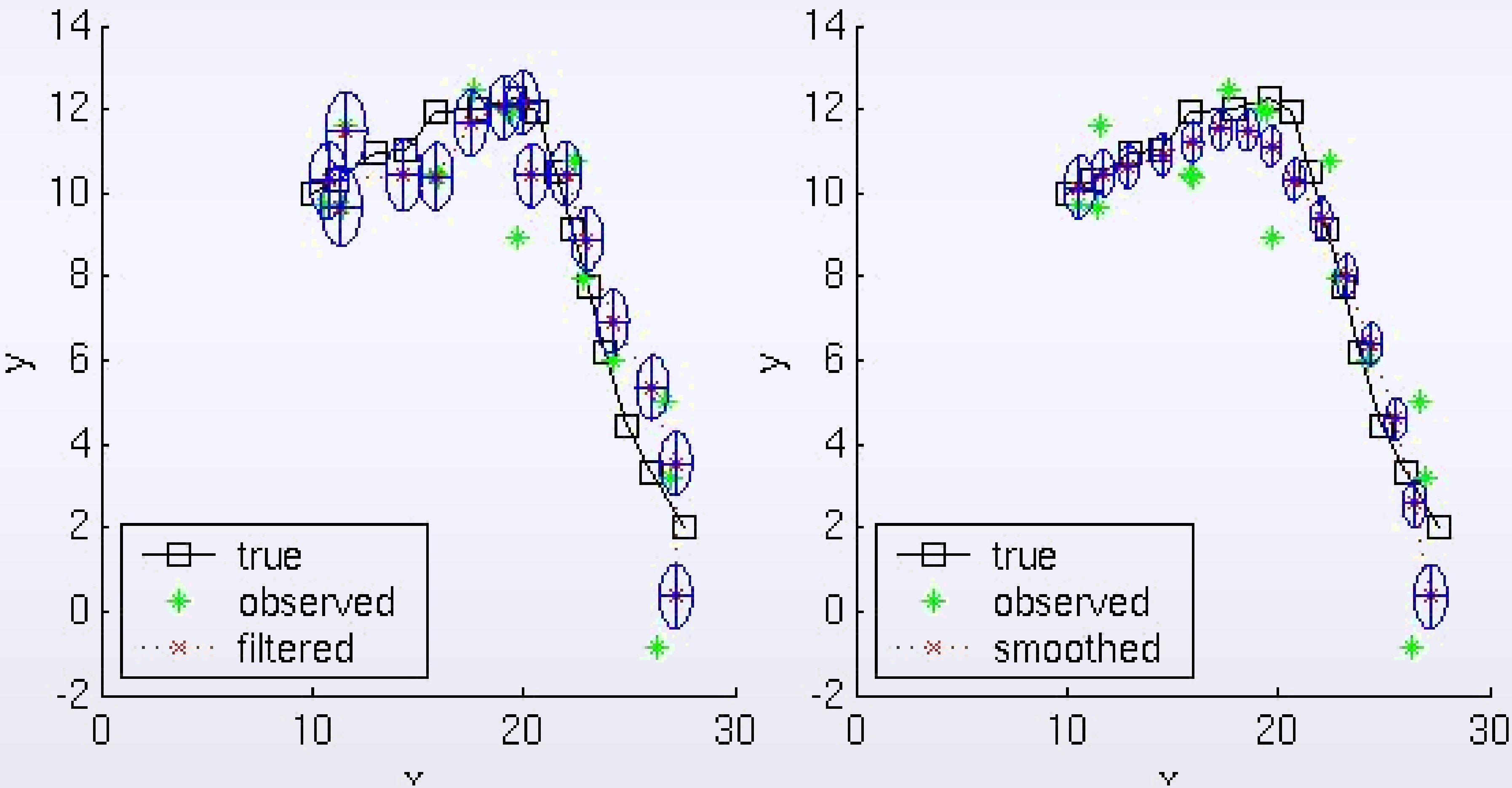
- The posterior mean minimizes the mean squared prediction error:

$$\mu_t = \arg\min_{\mu} \; \mathbb{E}\Big[\, ||x_t - \mu||^2 \mid y_1, \ldots, y_t \Big]$$

- The Kalman filter thus provides an optimal *online regression* algorithm

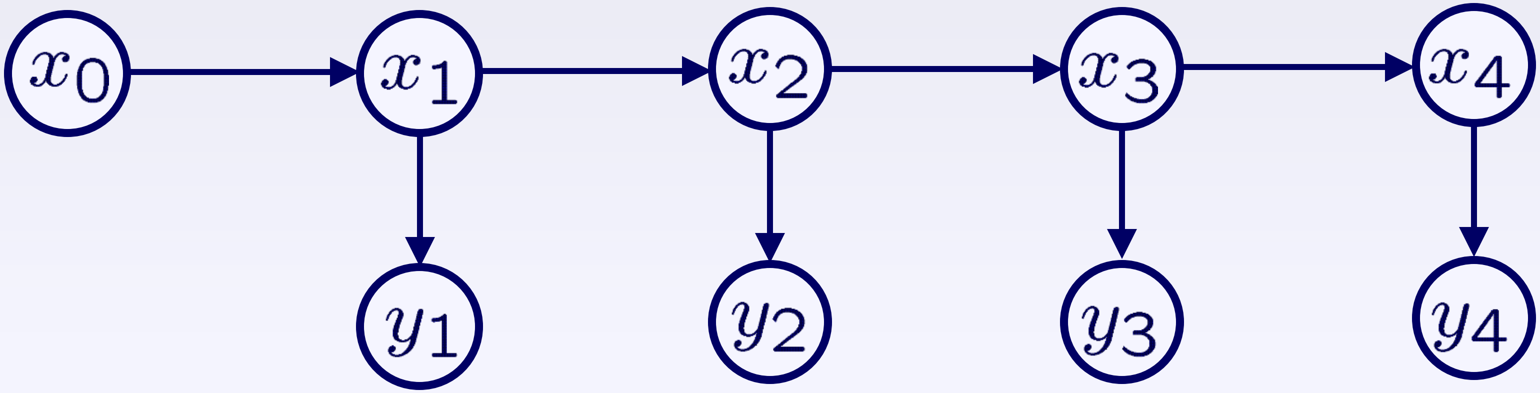
Constant Velocity Tracking

Kalman Filter *Kalman Smoother*



(K. Murphy, 1998)

Nonlinear State Space Models



$x_t \in \mathbb{R}^d$

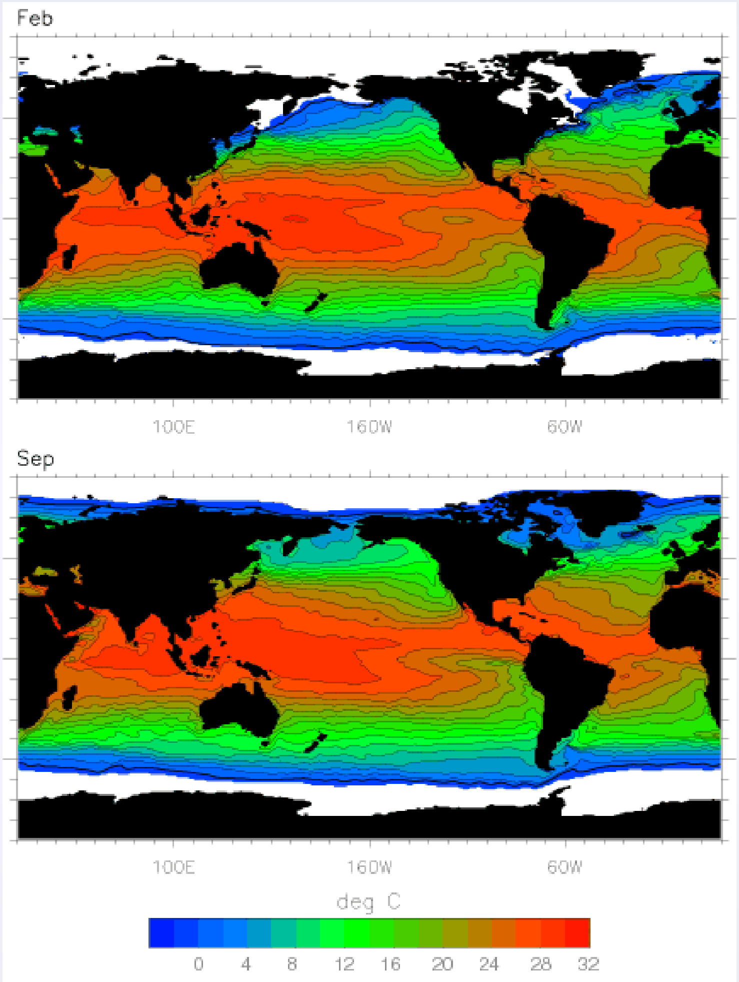
$y_t \in \mathbb{R}^k$

$$x_{t+1} = f(x_t, w_t) \qquad w_t \sim \mathcal{F}$$

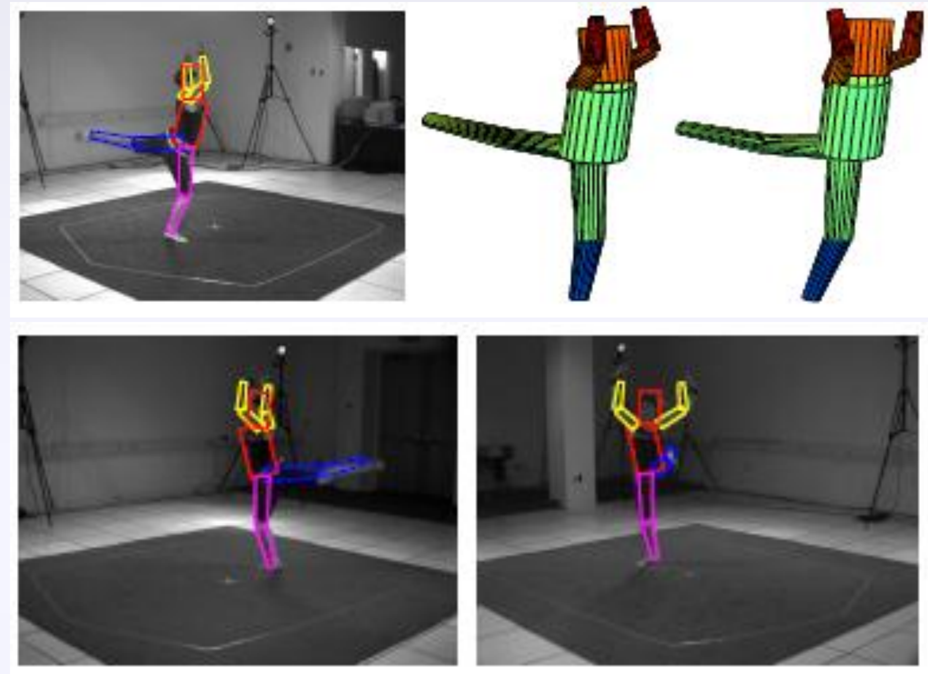
$$y_t = g(x_t, v_t) \qquad v_t \sim \mathcal{G}$$

- State dynamics and measurements given by potentially complex *nonlinear functions*
- Noise sampled from *non-Gaussian* distributions

Examples of Nonlinear Models

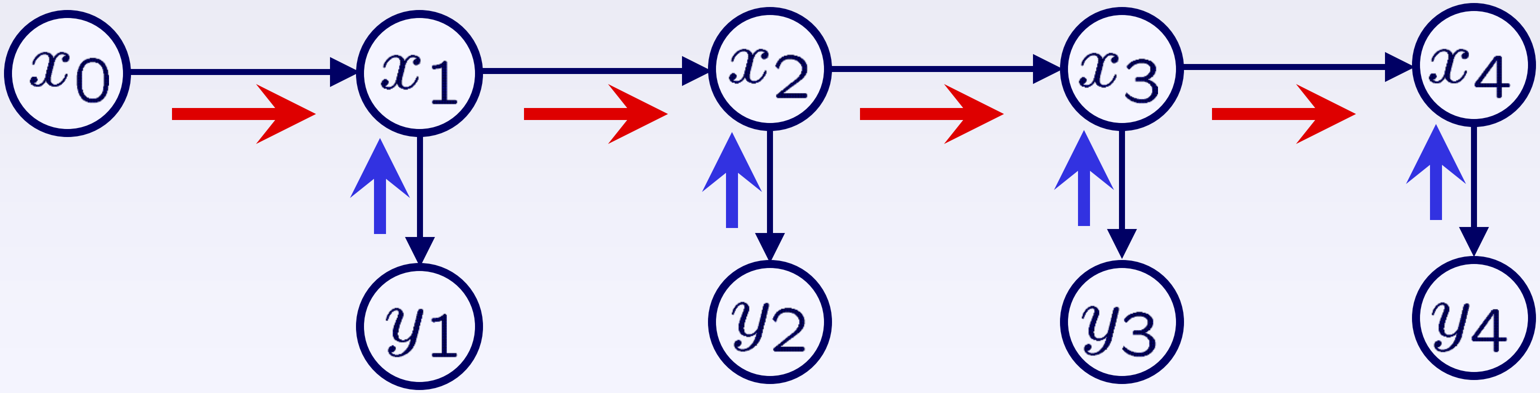


Dynamics implicitly determined by geophysical simulations



Observed image is a complex function of the 3D pose, other nearby objects & clutter, lighting conditions, camera calibration, etc.

Nonlinear Filtering



$$p(x_t \mid y_1, \ldots, y_{t-1}) = \tilde{q}_t(x_t)$$

$$p(x_t \mid y_1, \ldots, y_t) = q_t(x_t)$$

Prediction:

$$\tilde{q}_t(x_t) = \int p(x_t \mid x_{t-1}) q_{t-1}(x_{t-1})\, dx_{t-1}$$

Update:

$$q_t(x_t) = \frac{1}{Z_t} \tilde{q}_t(x_t) p(y_t \mid x_t)$$

Approximate Nonlinear Filters

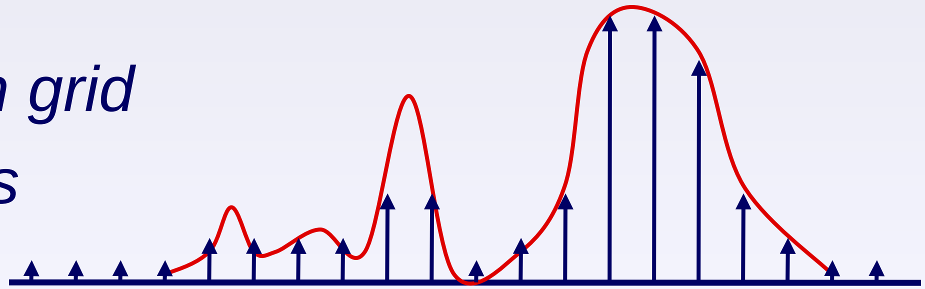
$$q_t(x_t) \propto p(y_t \mid x_t) \cdot \int p(x_t \mid x_{t-1}) q_{t-1}(x_{t-1}) \, dx_{t-1}$$

- Typically cannot directly *represent* these continuous functions, or determine a closed form for the prediction *integral*
- A wide range of approximate nonlinear filters have thus been proposed, including
  - *Histogram filters*
  - *Extended & unscented Kalman filters*
  - *Particle filters*
  - *…*

Nonlinear Filtering Taxonomy

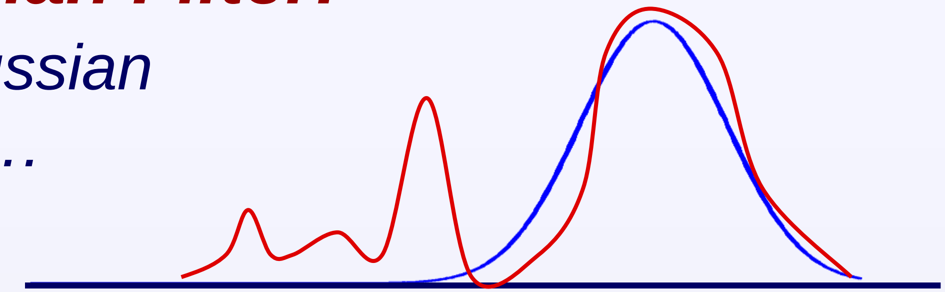
Histogram Filter:

- Evaluate on fixed discretization grid
- Only feasible in low dimensions
- Expensive or inaccurate



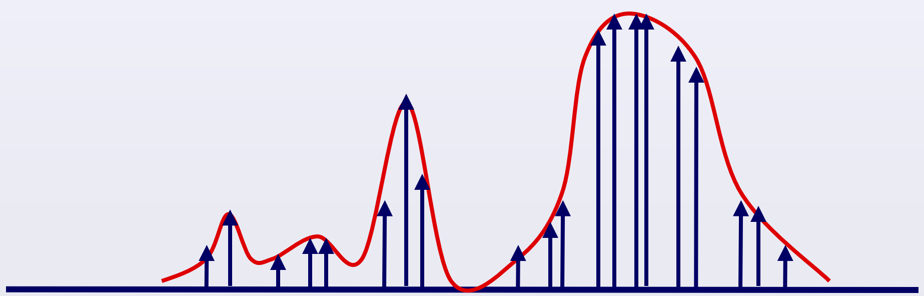
Extended/Unscented Kalman Filter:

- Approximate posterior as Gaussian via linearization, quadrature, …
- Inaccurate for multimodal posterior distributions



Particle Filter:

- Dynamically evaluate states with highest probability
- Monte Carlo approximation



Importance Sampling

$p(x) \longrightarrow$ true distribution (difficult to sample from)
assume may be evaluated *up to normalization Z*

$q(x) \longrightarrow$ proposal distribution (easy to sample from)

- Draw N *weighted* samples from proposal:

$$x_i \sim q(x) \qquad\qquad w_i = \frac{p(x_i)}{q(x_i)}$$

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

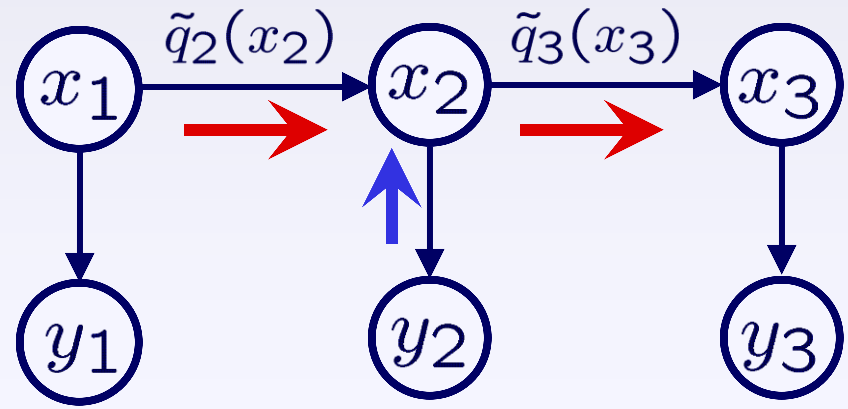
$$\hat{p}(x) = \sum_{i=1}^{N} \bar{w}_i \delta(x, x_i) \qquad\qquad \bar{w}_i = \frac{w_i}{\sum_j w_j}$$

- Nice asymptotic properties as $N \to \infty$

Particle Filters

Condensation, Sequential Monte Carlo, Survival of the Fittest,…

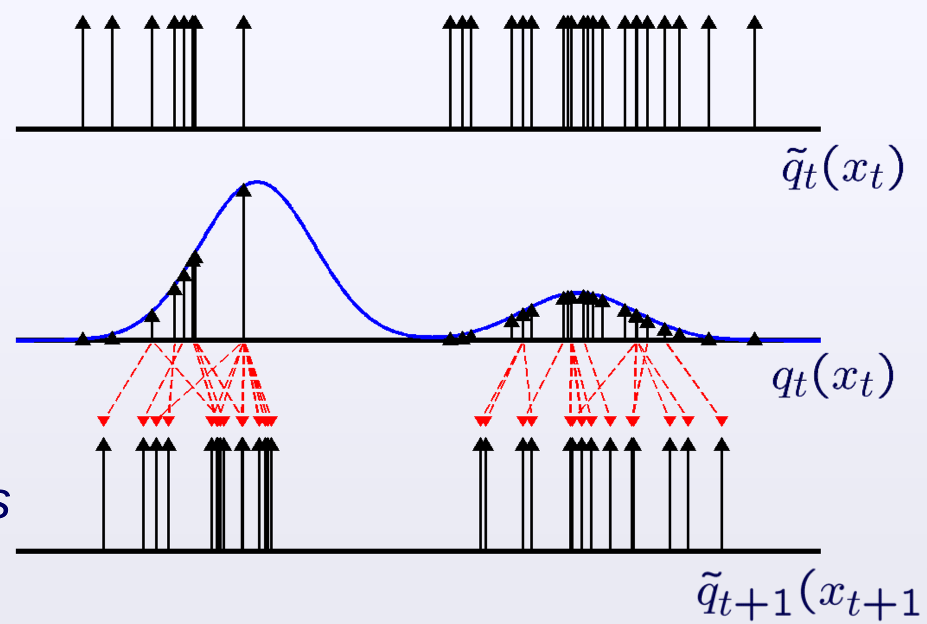
- Represent state estimates using a set of samples
- Dynamics provide proposal distribution for likelihood



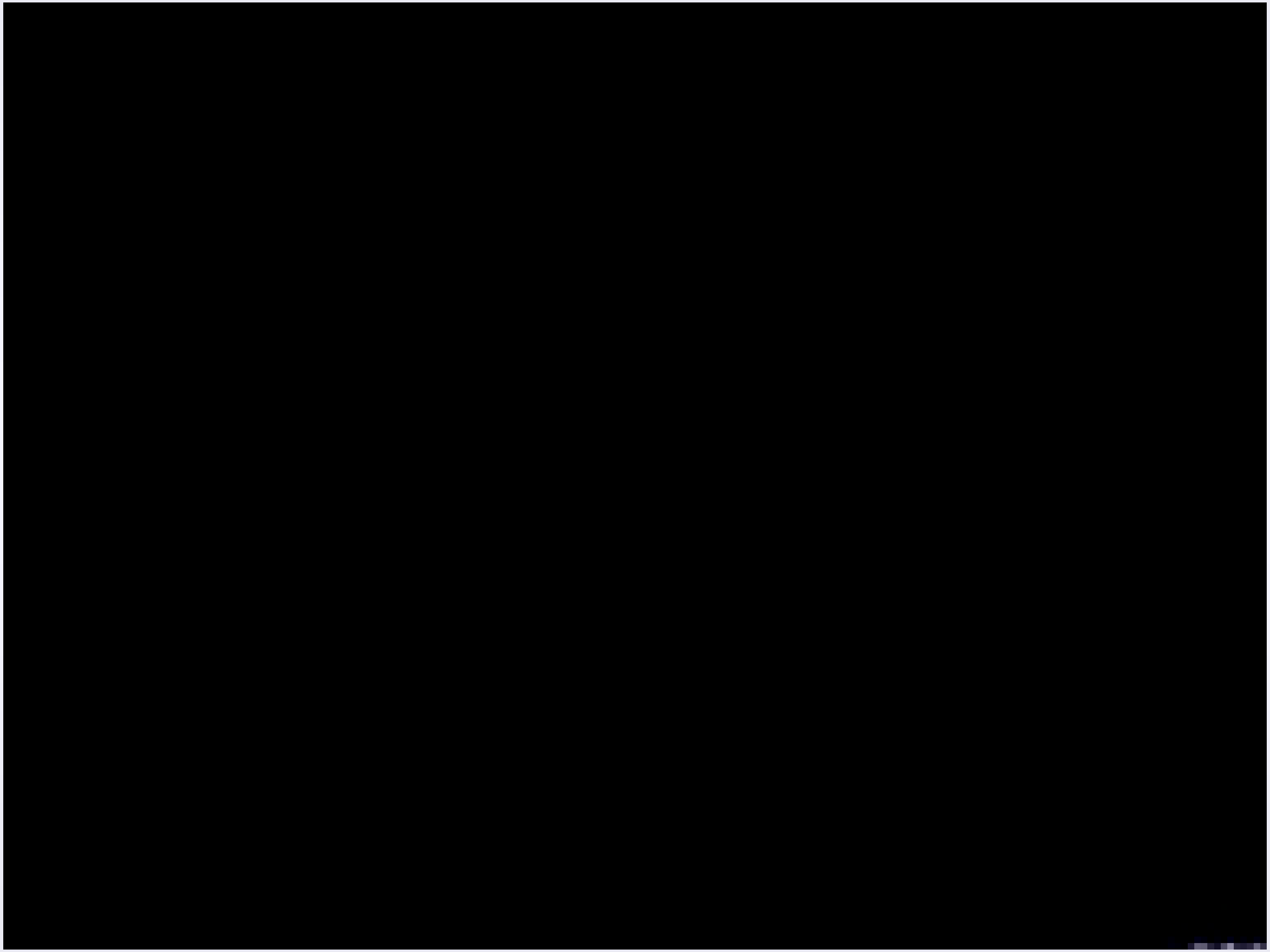
Sample-based density estimate

Weight by observation likelihood

Resample & propagate by dynamics



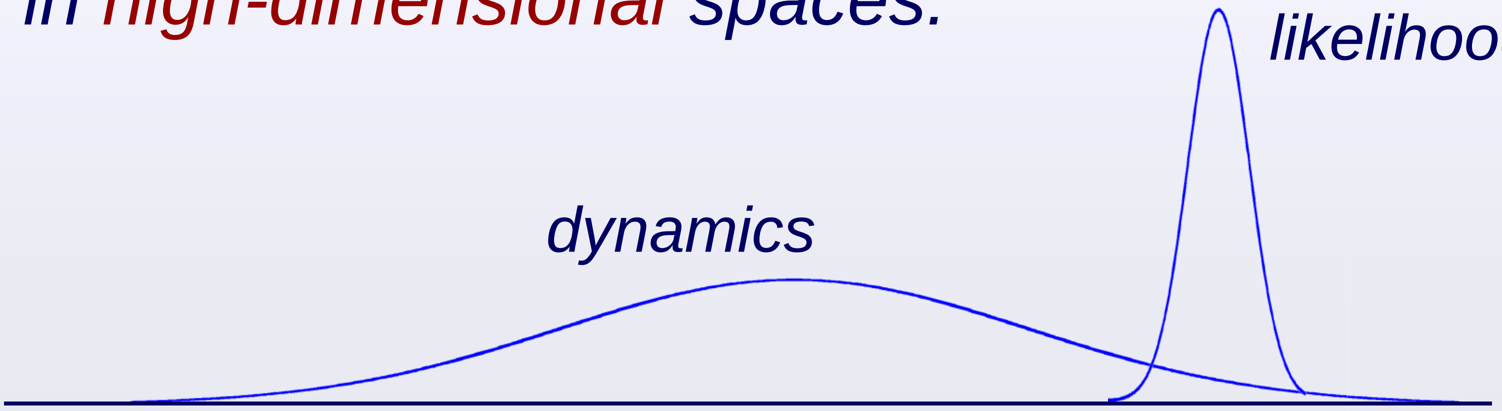
Particle Filtering Movie



(M. Isard, 1996)

Particle Filtering Caveats

- Easy to implement, effective in many applications, BUT
  - *It can be difficult to know how many samples to use, or to tell when the approximation is poor*
  - *Sometimes suffer catastrophic failures, where NO particles have significant posterior probability*
  - *This is particularly true with "peaky" observations in high-dimensional spaces:*



Continuous State HMMs

- There also exist algorithms for other learning & inference tasks in continuous-state HMMs:
    - *Smoothing*
    - *Likelihood calculation & classification*
    - *MAP state estimation*
    - *Learning via ML parameter estimation*
- For linear Gaussian state space models, these are easy generalizations of discrete HMM algorithms
- Nonlinear models can be more difficult…

Outline

Introduction to Sequential Processes

- Markov chains
- Hidden Markov models

Discrete-State HMMs

- Inference: Filtering, smoothing, Viterbi, classification
- Learning: EM algorithm

Continuous-State HMMs

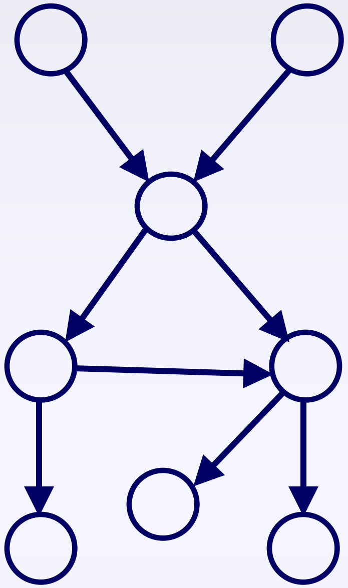
- Linear state space models: Kalman filters
- Nonlinear dynamical systems: Particle filters

More on Graphical Models

More on Graphical Models

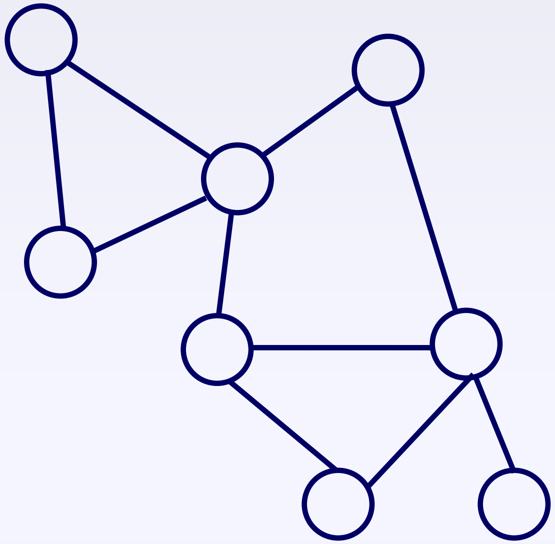
- Many applications have rich structure, but are not simple time series or sequences:
  - *Physics-based model of a complex system*
  - *Multi-user communication networks*
  - *Hierarchical taxonomy of documents/webpages*
  - *Spatial relationships among objects*
  - *Genetic regulatory networks*
  - *Your own research project?*
- Graphical models provide a framework for:
  - *Specifying statistical models for complex systems*
  - *Developing efficient learning algorithms*
  - *Representing and reasoning about complex joint distributions.*

Types of Graphical Models



Nodes ⟺ Random Variables

Edges ⟺ Probabilistic (Markov) Relationships



Directed Graphs

Specify a hierarchical, causal generative process (child nodes depend on parents)

$$p(x) = \Pi_i p(x_i | \mathrm{Parents}_i)$$

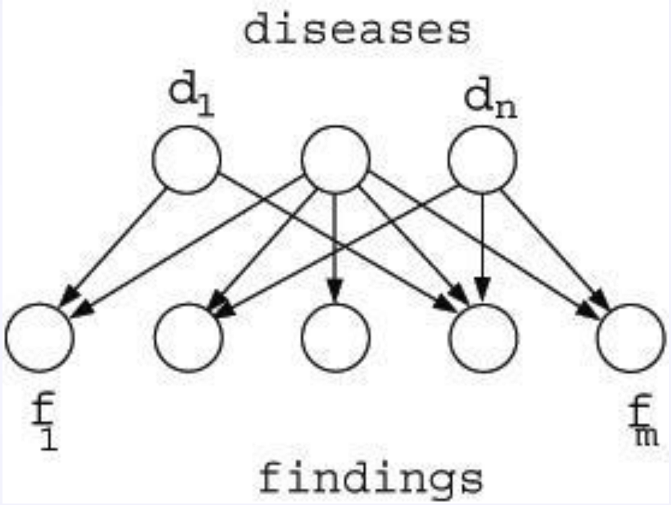
Undirected Graphs

Specific symmetric, non-causal dependencies (soft or probabilistic constraints)

$$p(x) = \prod_{\text{cliques}} \Psi(x_{\text{clique}})$$

Quick Medical Reference (QMR) model

- A probabilistic graphical model for diagnosis with 600 *disease* nodes, 4000 *finding* nodes



- Node probabilities $p(f_i|d)$ were assessed from an expert (Shwe et al., 1991)
- Want to compute posteriors: $p(d_i|f)$
- Is this tractable?

Directed Graphical Models

- AKA Bayes Net.
- Any distribution can be written as

$$p(x) = p(x_1)p(x_2|x_1)p(x_3|x_{1:2}), \ldots, p(x_n|x_{1:n-1})$$

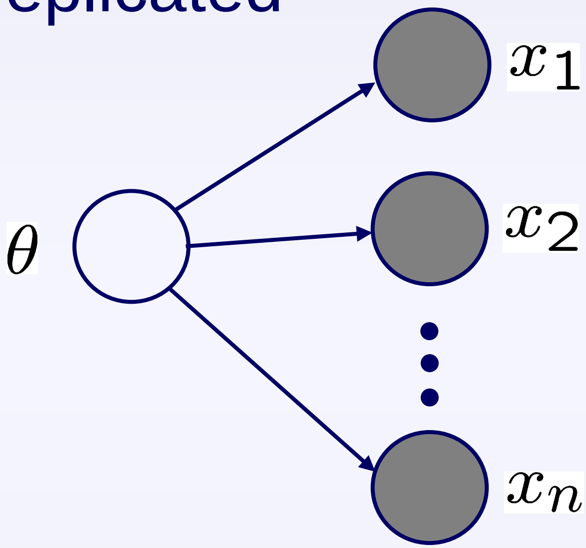
- Here, if the variables are topologically sorted (parents come before children)

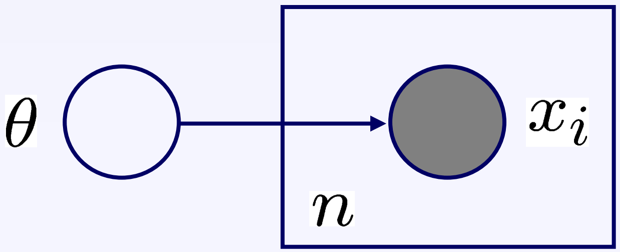
$$p(x_k|x_{1:k-i}) \triangleq p(x_k|x_{\text{parents}_k})$$

- Much simpler: an arbitrary $p(x_n|x_{1:n-1})$ is a huge (n-1) dimensional matrix.
- Inference: knowing the value of some of the nodes, infer the rest.
    - Marginals, MAP

Plates

- A plate is a "macro" that allows subgraphs to be replicated

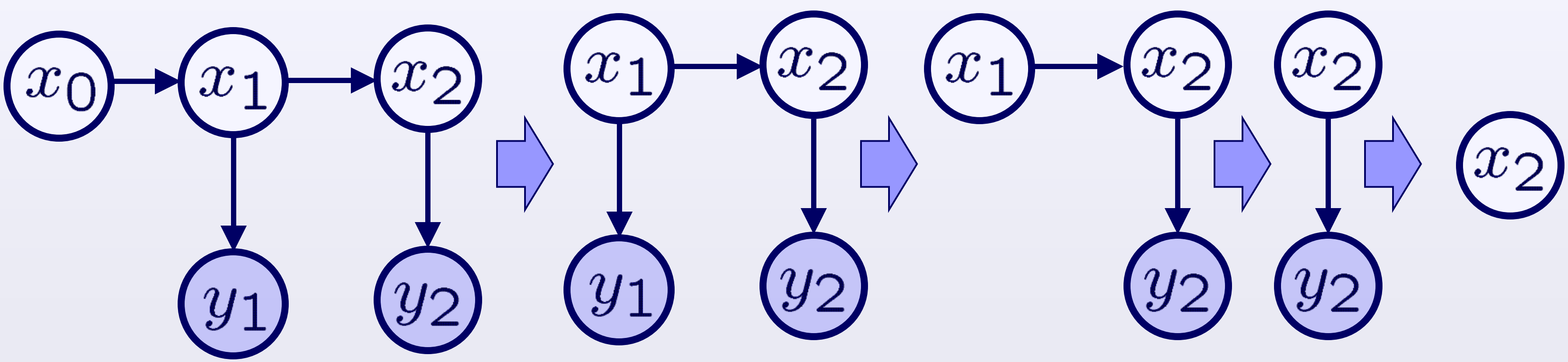




- Graphical representation of an exchangeability assumption for $(X_1, X_2, \ldots, X_n)$

Elimination Algorithm

- Takes a graphical model and produces one without a particular node puts the same probability distribution on the rest of the nodes.
- Very easy on trees, possible (though potentially computationally expensive) on general DAGs.
- If we eliminate all but one node, that tells us the distribution of that node.

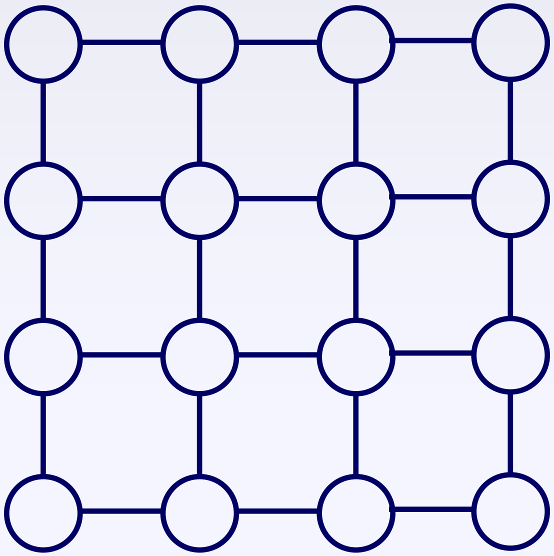


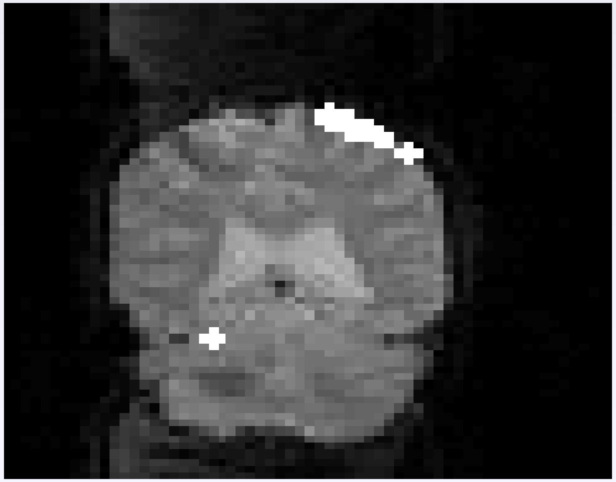
Elimination Algorithm (cont)

- The symbolic counterpart of elimination is equivalent to triangulation of the graph
- Triangulation: remove the nodes sequentially; when a node is removed, connect all of its remaining neighbors
- The computational complexity of elimination scales as exponential in the size of the largest clique in the triangulated graph

Markov Random Fields in Vision

Idea: Nearby pixels are similar.





fMRI Analysis *(Kim et. al. 2000)*

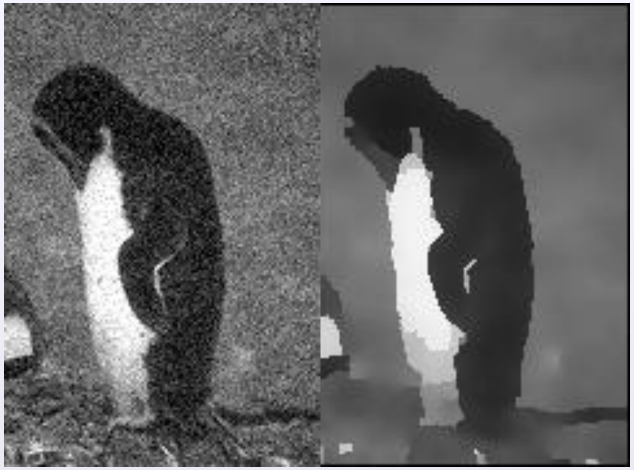
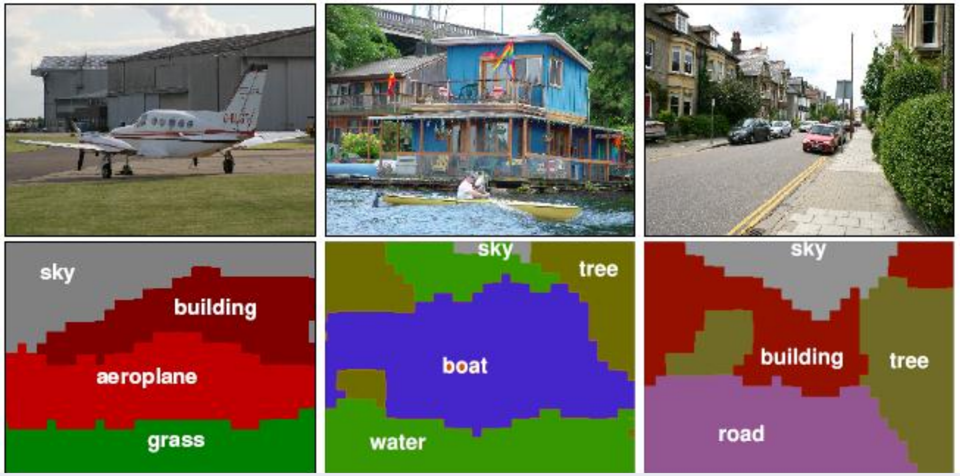


Image Denoising

(Felzenszwalb & Huttenlocher 2004)

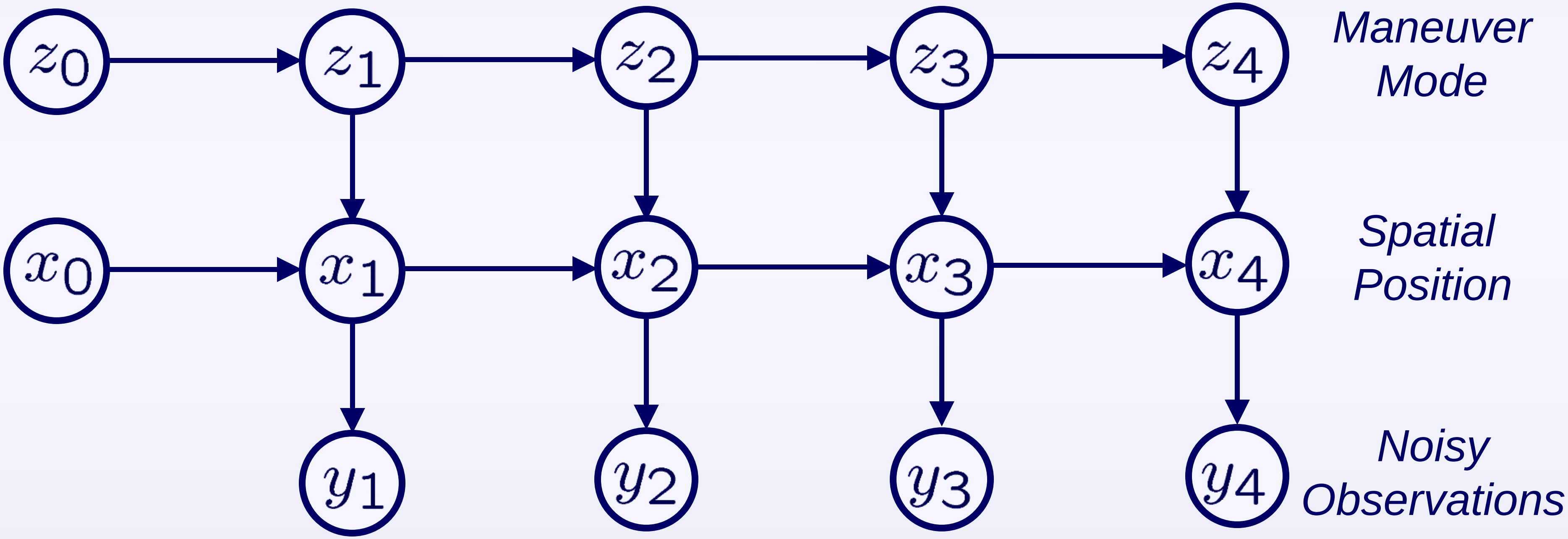


Segmentation & Object Recognition

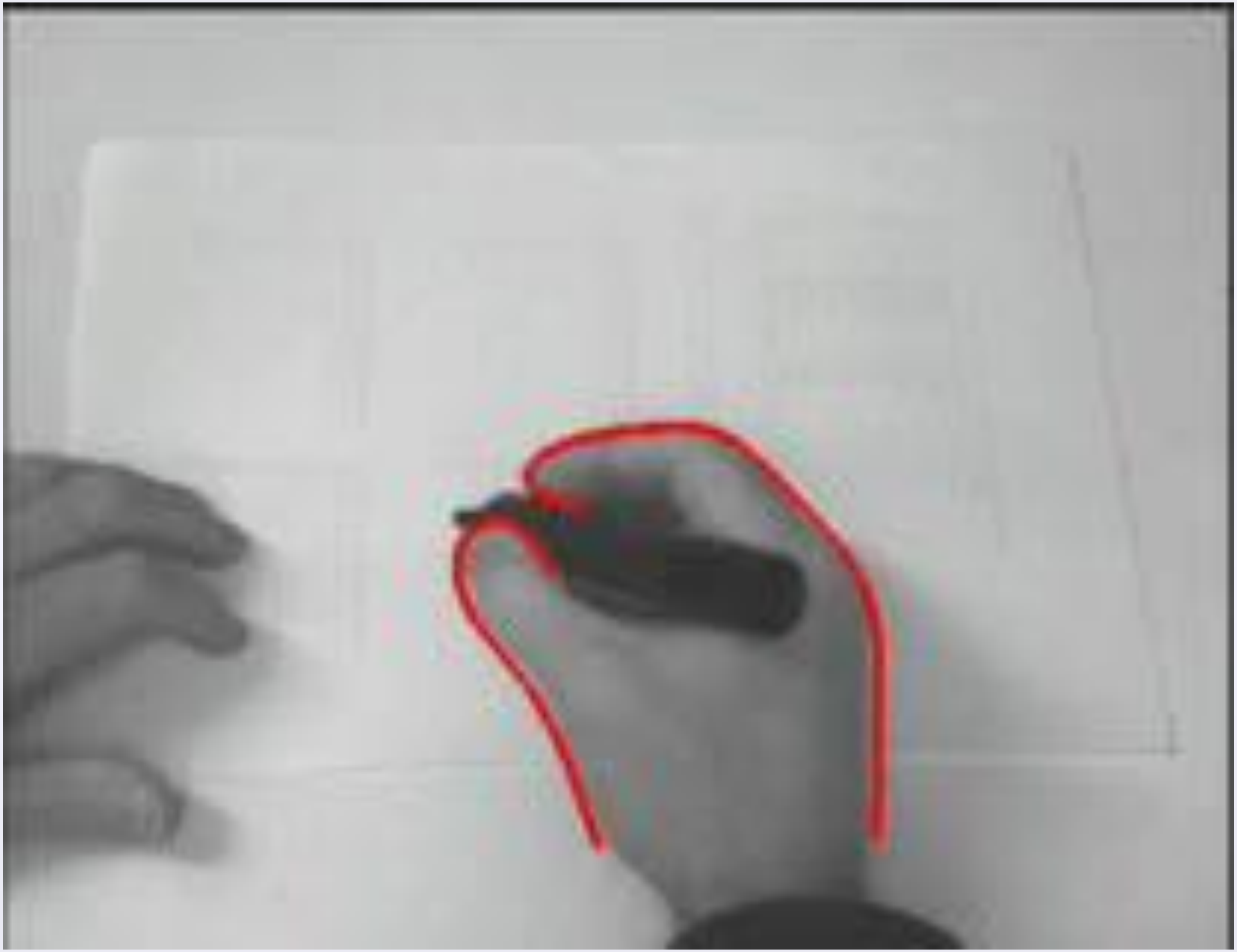
(Verbeek & Triggs 2007)

Dynamic Bayesian Networks

Specify and exploit internal structure in the hidden states underlying a time series.
Generalizes HMMs

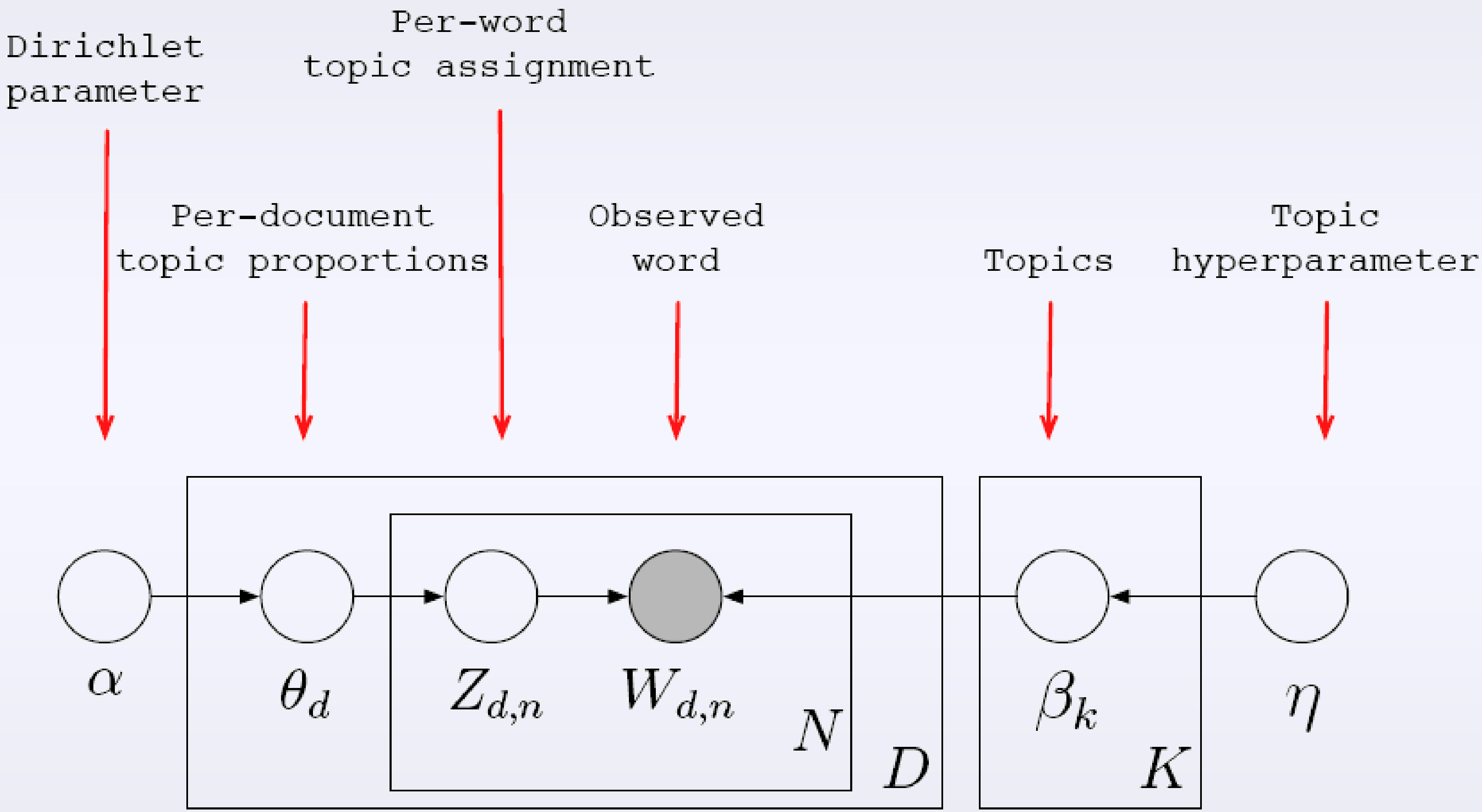


DBN Hand Tracking Video



Isard et. al., 1998

Topic Models for Documents



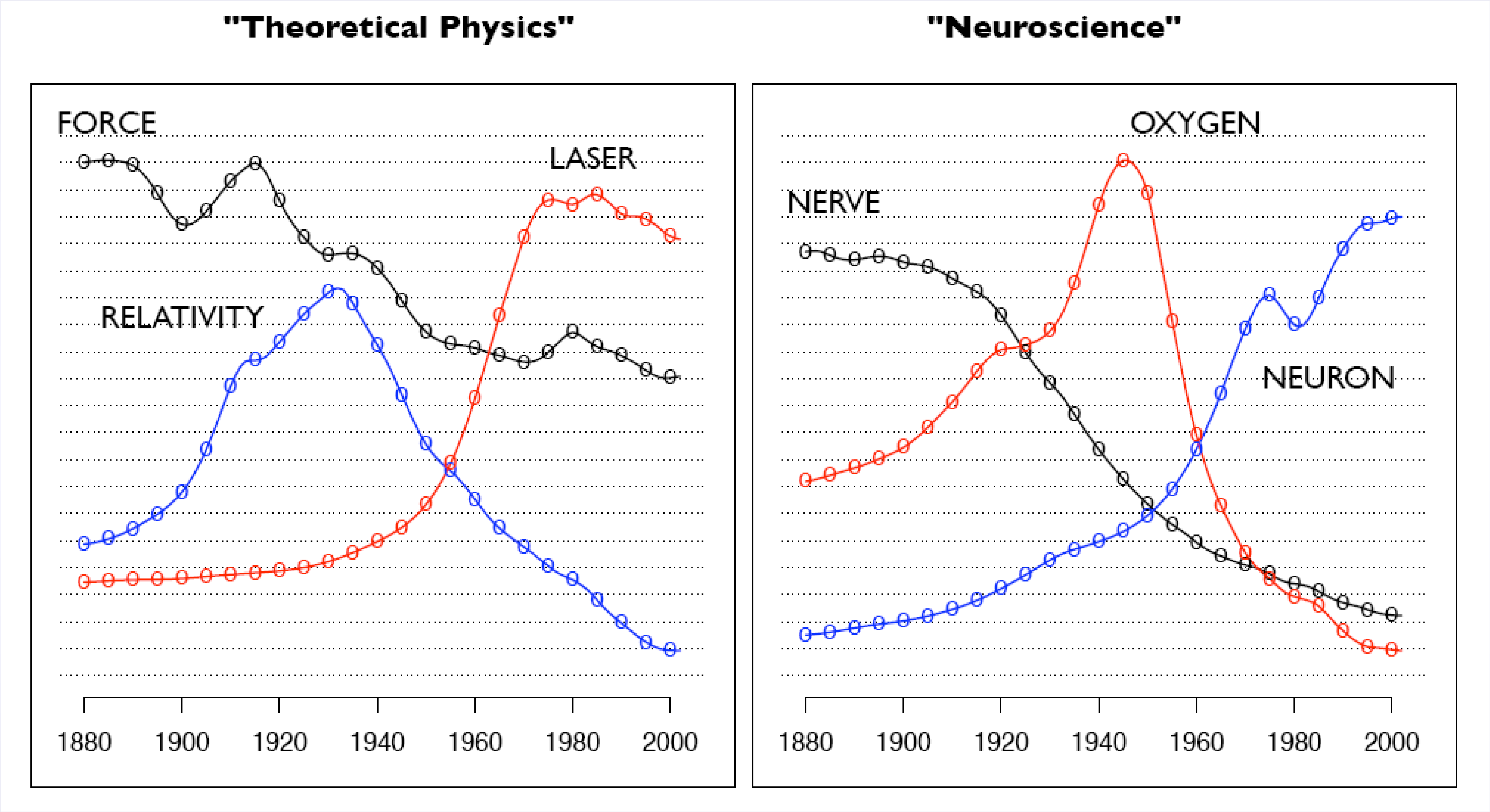
D. Blei, 2007

Topics Learned from *Science*

human	evolution	disease	computer
genome	evolutionary	host	models
dna	species	bacteria	information
genetic	organisms	diseases	data
genes	life	resistance	computers
sequence	origin	bacterial	system
gene	biology	new	network
molecular	groups	strains	systems
sequencing	phylogenetic	control	model
map	living	infectious	parallel
information	diversity	malaria	methods
genetics	group	parasite	networks
mapping	new	parasites	software
project	two	united	new
sequences	common	tuberculosis	simulations

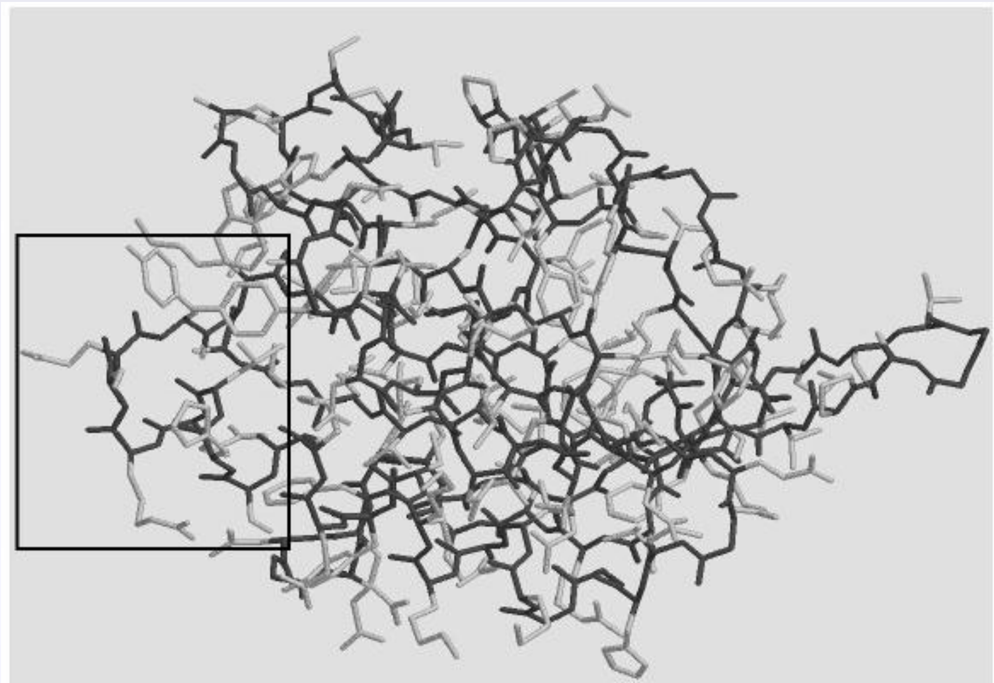
D. Blei, 2007

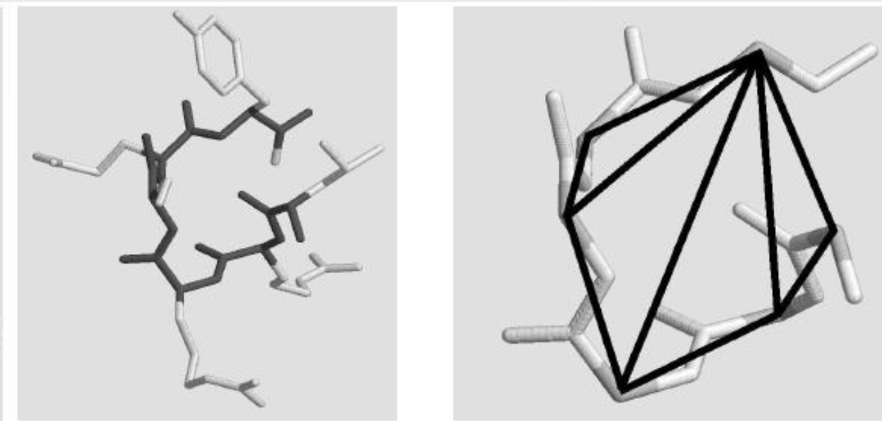
Temporal Topic Evolution



D. Blei, 2007

Bioinformatics



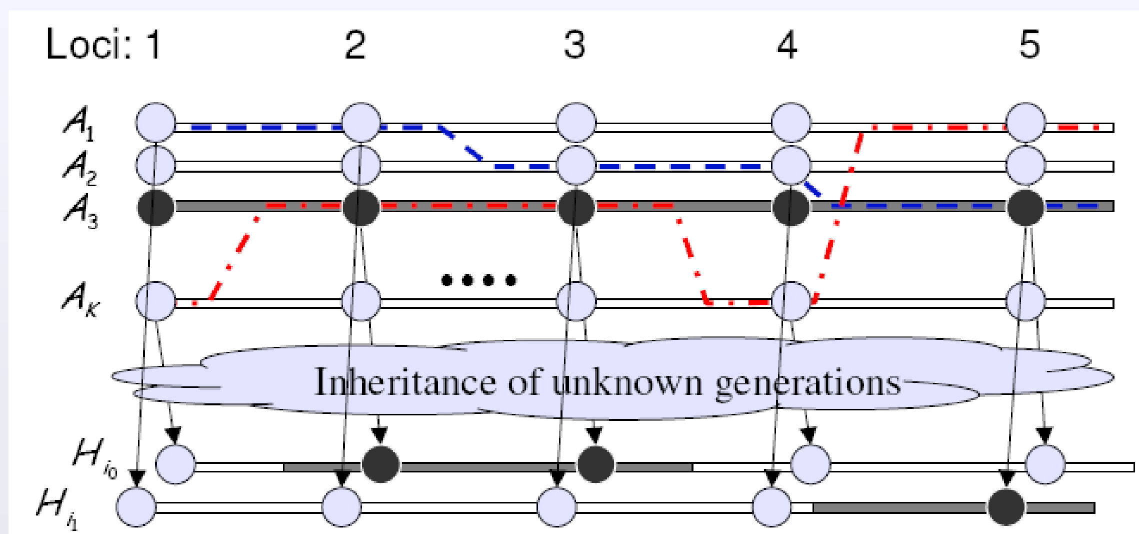


Protein Folding

(Yanover & Weiss 2003)

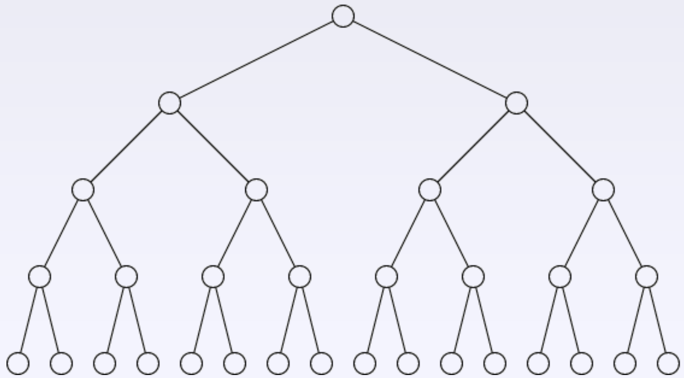
Computational Genomics

(Xing & Sohn 2007)



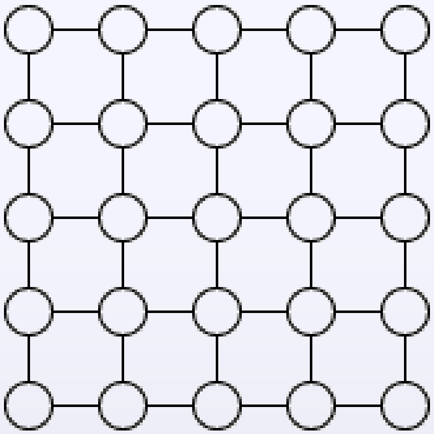
Learning in Graphical Models

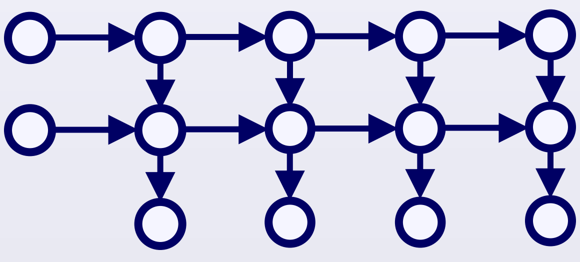
Tree-Structured Graphs



There are direct, efficient extensions of HMM learning and inference algorithms

Graphs with Cycles





- **Junction Tree:** *Cluster nodes to remove cycles (exact, but computation exponential in "distance" of graph from tree)*
- **Monte Carlo Methods:** *Approximate learning via simulation (Gibbs sampling, importance sampling, …)*
- **Variational Methods:** *Approximate learning via optimization (mean field, loopy belief propagation, …)*