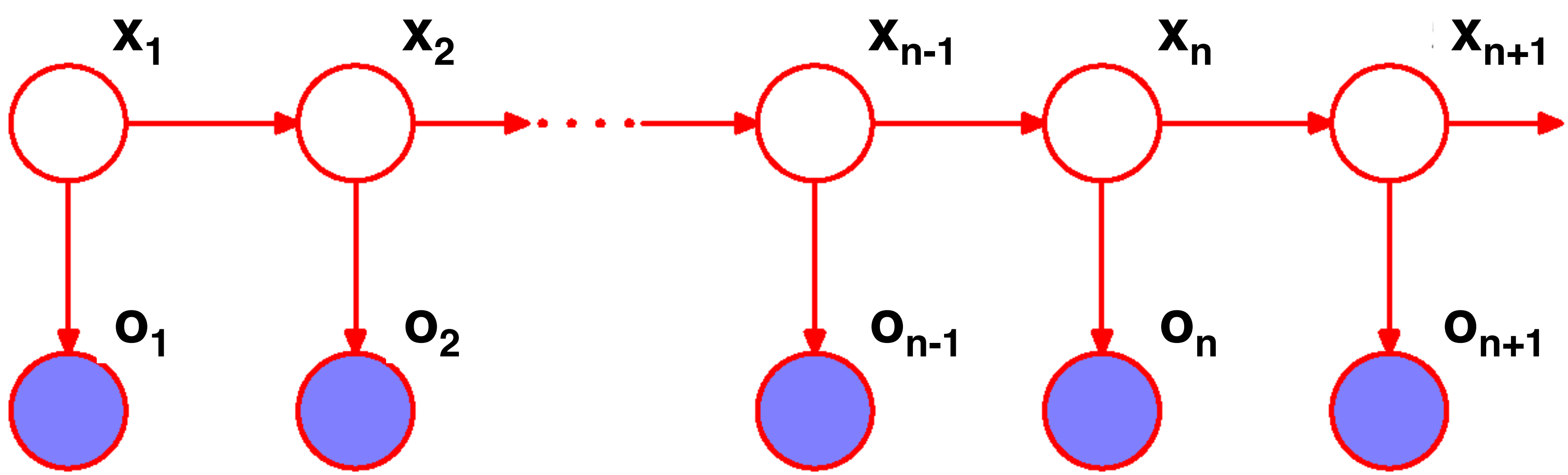
# CSE 586, Spring 2015
# Computer Vision II

## Hidden Markov Model and Kalman Filter

# Recall: Modeling Time Series

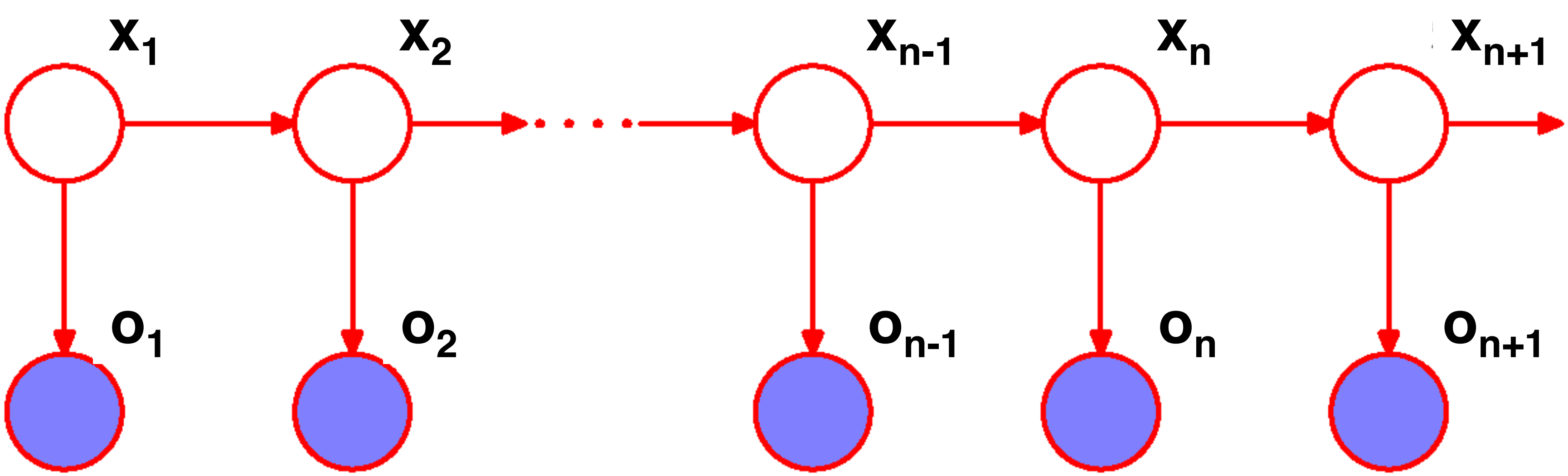
State-Space Model:

- You have a Markov chain of latent (unobserved) states
- Each state generates an observation



# Modeling Time Series

$$P(x1,x2,x3,x4,...,o1,o2,o3,o4,...) =$$
$$P(x1)P(o1|x1)P(x2|x1)P(o2|x2)P(x3|x2)P(o3|x3)P(x4|x3)P(o4|x4)......$$

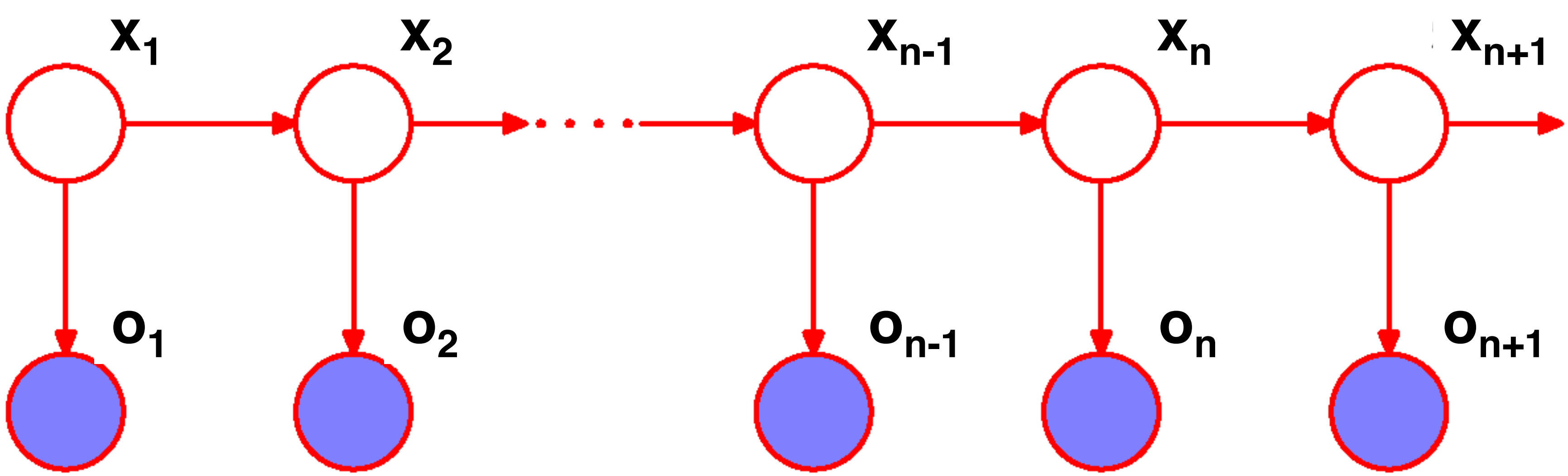


Robert Collins
CSE586

# Modeling Time Series

Examples of State Space models

- •Hidden Markov model
- •Kalman filter

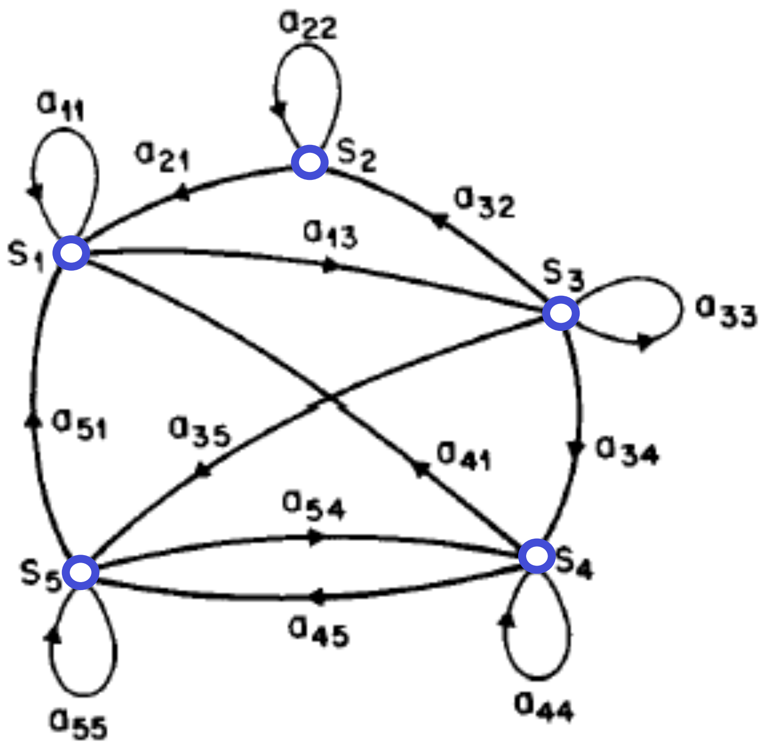


# Hidden Markov Models

Note: a good background reference is LR Rabiner, “A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition,” *Proc. of the IEEE*, Vol.77, No.2, pp.257-286, 1989.

# Markov Chain

Note: this picture is a state transition diagram, not a graphical model!




Set of states, e.g. {S1,S2,S3,S4,S5}

Table of transition probabilities ($a_{ij}$ = Prob of going from state $S_i$ to $S_j$)

$$\begin{matrix} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} \\ a_{31} & a_{32} & a_{33} & a_{34} & a_{35} \\ a_{41} & a_{42} & a_{43} & a_{44} & a_{45} \\ a_{51} & a_{52} & a_{53} & a_{54} & a_{55} \end{matrix} \qquad P(S_j \mid S_i)$$

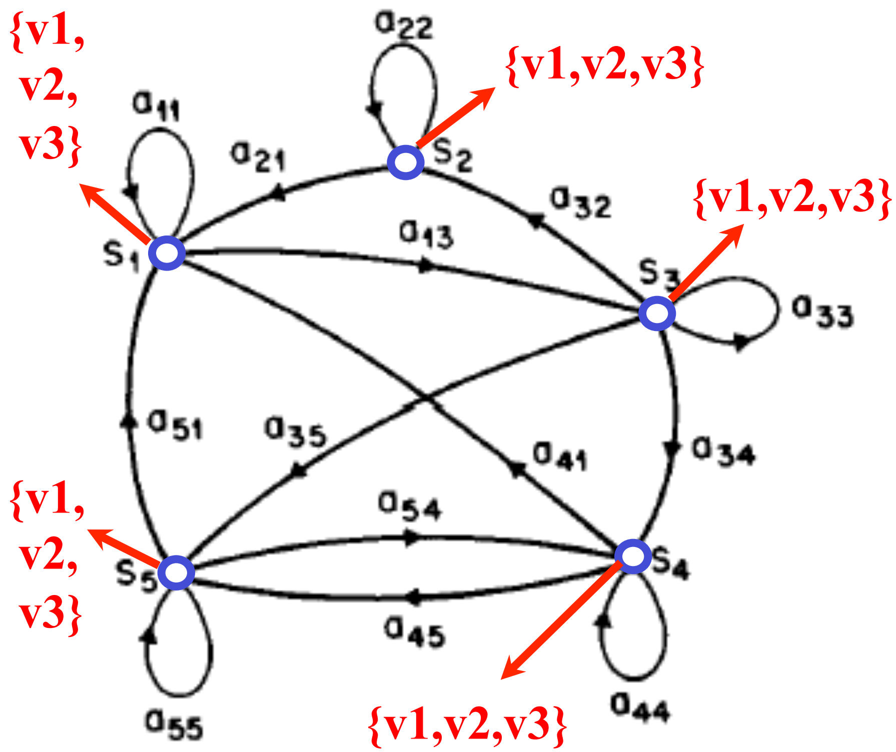
Prob of starting in each state e.g. $\pi_1, \pi_2, \pi_3, \pi_4, \pi_5$ $\qquad P(S_i)$

Computation: prob of an ordered sequence $S_2\ S_1\ S_5\ S_4$

$$P(S_2)\, P(S_1 \mid S_2)\, P(S_5 \mid S_1)\, P(S_4 \mid S_5) = \pi_2\, a_{21}\, a_{15}\, a_{54}$$

Robert Collins
CSE586

# Hidden Markov Model




Set of states, e.g. {S1,S2,S3,S4,S5}

Table of transition probabilities ($a_{ij}$ = Prob of going from state $S_i$ to $S_j$)

$$\begin{matrix} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} \\ a_{31} & a_{32} & a_{33} & a_{34} & a_{35} \\ a_{41} & a_{42} & a_{43} & a_{44} & a_{45} \\ a_{51} & a_{52} & a_{53} & a_{54} & a_{55} \end{matrix} \qquad P(S_j \mid S_i)$$

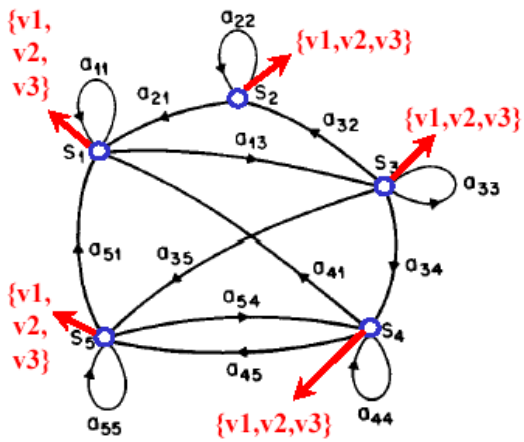
Prob of starting in each state e.g. $\pi_1, \pi_2, \pi_3, \pi_4, \pi_5$ $\qquad P(S_i)$

Set of observable symbols, e.g. {v1,v2,v3}

Prob of seeing each symbol in a given state ($b_{ik}$ = Prob seeing $v_k$ in state $S_i$)

$$\begin{matrix} b_{11} & b_{21} & b_{31} & b_{41} & b_{51} \\ b_{12} & b_{22} & b_{32} & b_{42} & b_{52} \\ b_{13} & b_{23} & b_{33} & b_{43} & b_{53} \end{matrix} \qquad P(v_k \mid S_i)$$

# Hidden Markov Model




Set of states, e.g. {S1,S2,S3,S4,S5}

Table of transition probabilities ($a_{ij}$ = Prob of going from state $S_i$ to $S_j$)

$$\begin{matrix} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} \\ a_{31} & a_{32} & a_{33} & a_{34} & a_{35} \\ a_{41} & a_{42} & a_{43} & a_{44} & a_{45} \\ a_{51} & a_{52} & a_{53} & a_{54} & a_{55} \end{matrix} \quad P(S_j \mid S_i)$$

Prob of starting in each state e.g. $\pi_1, \pi_2, \pi_3, \pi_4, \pi_5$ $\quad P(S_i)$

Set of observable symbols, e.g. {v1,v2,v3}

Prob of seeing each symbol in a given state ($b_{ik}$ = Prob seeing $v_k$ in state $S_i$)

$$\begin{matrix} b_{11} & b_{21} & b_{31} & b_{41} & b_{51} \\ b_{12} & b_{22} & b_{32} & b_{42} & b_{52} \\ b_{13} & b_{23} & b_{33} & b_{43} & b_{53} \end{matrix} \quad P(v_k \mid S_i)$$

Computation: Prob of ordered sequence $(S_2,v_3)\ (S_1,v_1)\ (S_5,v_2)\ (S_4,v_1)$

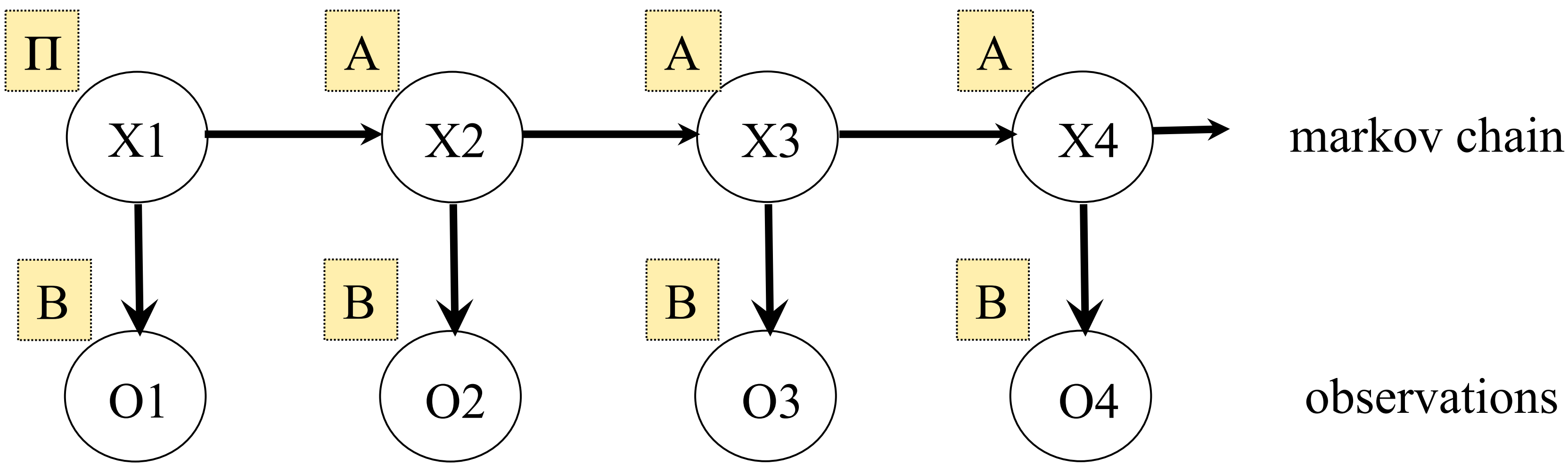
$$\begin{aligned} & P(S2,S1,S5,S4,v3,v1,v2,v1) \\ &= P(S_2)\, P(v_3|S_2)\, P(S_1|S_2)\, P(v_1|S_1)\, P(S_5|S_1)\, P(v_2|S_5)\, P(S_4|S_5)\, P(v_1|S_4) \\ &= \pi_2\, b_{23}\, a_{21}\, b_{11}\, a_{15}\, b_{52}\, a_{54}\, b_{41} \\ &= (\pi_2\, a_{21}\, a_{15}\, a_{54})\,(b_{23}\, b_{11}\, b_{52} b_{41}) \\ &= P(S1,S2,S3,S4)\, P(v1,v2,v3,v4|S1,S2,S3,S4) \end{aligned}$$

# HMM as Graphical Model

Given HMM with states S1,S2,...,Sn, symbols v1,v2,...,vk, transition probs $A=\{a_{ij}\}$, initial probs $\Pi=\{\pi_i\}$, and observation probs $B=\{b_{ik}\}$

Let state random variables be X1,X2,X3,X4,... with Xt being state at time t. Allowable values of Xt (a discrete random variable) are {S1,S2,...,Sn}

Let observed random variables be O1,O2,O3... with Ot observed at time t. Allowable values of Ot (a discrete random variable) are {v1,v2,...,vk}

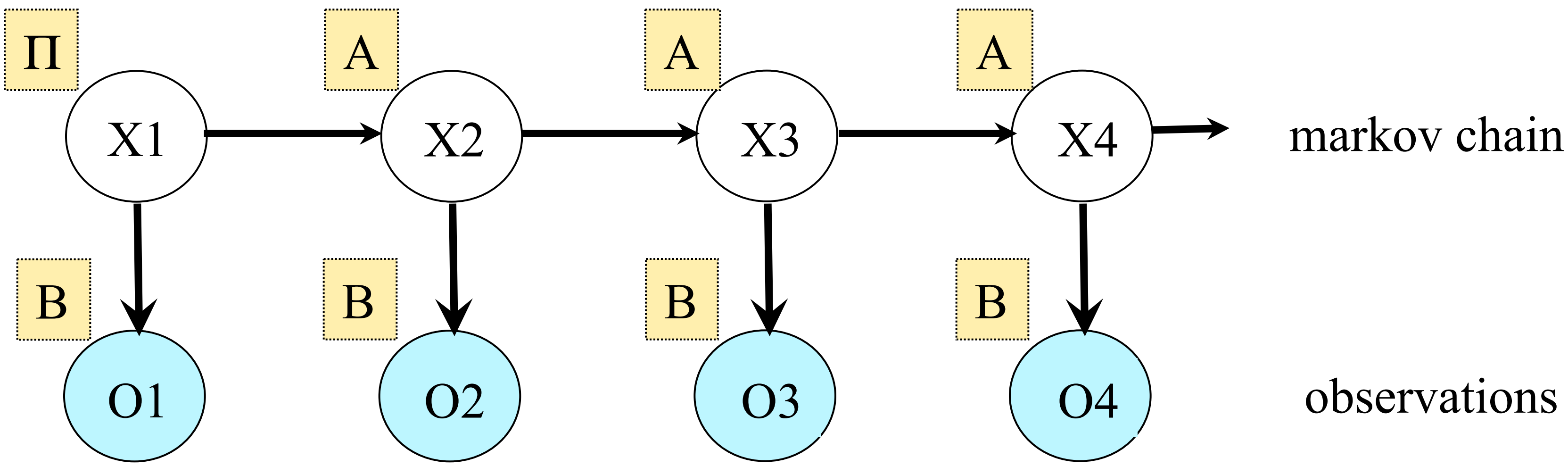


# HMM as Graphical Model

Given HMM with states S1,S2,...,Sn, symbols v1,v2,...,vk, transition probs $A=\{a_{ij}\}$, initial probs $\Pi=\{\pi_i\}$, and observation probs $B=\{b_{ik}\}$

Let state random variables be X1,X2,X3,X4,... with Xt being state at time t.
Allowable values of Xt (a discrete random variable) are {S1,S2,...,Sn}

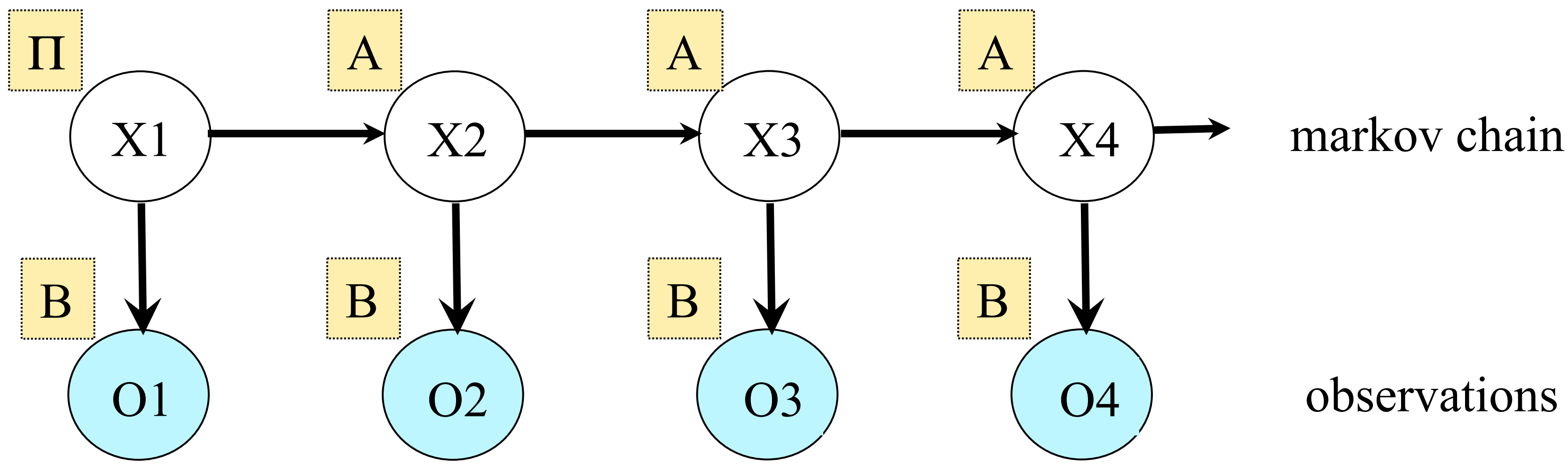
Let observed random variables be O1,O2,O3... with Ot observed at time t.
Allowable values of Ot (a discrete random variable) are {v1,v2,...,vk}




Oi are observed variables. Xj are hidden (latent) variables.

# HMM as Graphical Model

Verify: What is P(X1,X2,X3,X4,O1,O2,O3,O4)?




P(X1,X2,X3,X4,O1,O2,O3,O4) =
P(X1) P(O1|X1) P(X2|X1) P(O2|X2) P(X3|X2) P(O3|X3) P(X4|X3)P(O4|X4) • • •

# Three Computations for HMMs

(From Rabiner tutorial)

*Problem 1:* Given the observation sequence $O = O_1 O_2 \cdots O_T$, and a model $\lambda = (A, B, \pi)$, how do we efficiently compute $P(O|\lambda)$, the probability of the observation sequence, given the model?

*Problem 2:* Given the observation sequence $O = O_1 O_2 \cdots O_T$, and the model $\lambda$, how do we choose a corresponding state sequence $Q = q_1 q_2 \cdots q_T$ which is optimal in some meaningful sense (i.e., best "explains" the observations)?

*Problem 3:* How do we adjust the model parameters $\lambda = (A, B, \pi)$ to maximize $P(O|\lambda)$?

# HMM Problem 1

What is the likelihood of observing a sequence, e.g. O1,O2,O3 ?

Note: there are multiple ways that a given sequence could be emitted, involving different sequences of hidden states X1,X2,X3

One (inefficient) way to compute the answer: generate all sequences of three states Sx,Sy,Sz and compute P(Sx,Sy,Sz,O1,O2,O3) [which we know how to do]. Summing up over all sequences of three states gives us our answer.

Drawback, there are 3^N subsequences that can be formed from N states.

# HMM Problem 1

Better (efficient) solution is based on message passing / belief propagation.

marginal $$P(O_1,O_2,O_3)=\sum_{x_1}\sum_{x_2}\sum_{x_3}P(x_1,x_2,x_3,O_1,O_2,O_3)$$

$$=\sum_{x_1}\sum_{x_2}\sum_{x_3} P(x_1)P(O_1|x_1)P(x_2|x_1)P(O_2|x_2)p(x_3|x_2)P(O_3|x_3)$$

$$=\sum_{x_1}P(x_1)P(O_1|x_1)\left(\sum_{x_2}P(x_2|x_1)P(O_2|x_2)\left(\sum_{x_3}p(x_3|x_2)P(O_3|x_3)\right)\right)$$

**This is the sum-product procedure of message passing. It is called the *forward-backward* procedure in HMM terminology**

# HMM Problem 2

What is the most likely sequence of hidden states S1,S2,S3 given that we have observed O1,O2,O3 ?

$$X_{MAP} = \underset{X}{\operatorname{argmax}} \; P(x_1, x_2, x_3 | O_1, O_2, O_3)$$

$$= \underset{X}{\operatorname{argmax}} \; P(x_1, x_2, x_3, O_1, O_2, O_3)$$

Again, there is an inefficient way based on explicitly generating the exponential number of possible state sequences... AND, there is a more efficient way using message passing.

# HMM Problem 2

$$X_{MAP} = \underset{X}{\text{argmax}}\ P(\underbrace{x_1, x_2, x_3}_{X}, O_1, O_2, O_3)$$

$$= \underset{X}{\text{argmax}}\ P(x_1)P(O_1|x_1)P(x_2|x_1)P(O_2|x_2)p(x_3|x_2)P(O_3|x_3)$$

**Define:**

$$\Phi(X) = \max_{x_1} P(x_1)P(O_1|x_1)\left(\max_{x_2} P(x_2|x_1)P(O_2|x_2)\left(\max_{x_3} p(x_3|x_2)P(O_3|x_3)\right)\right)$$

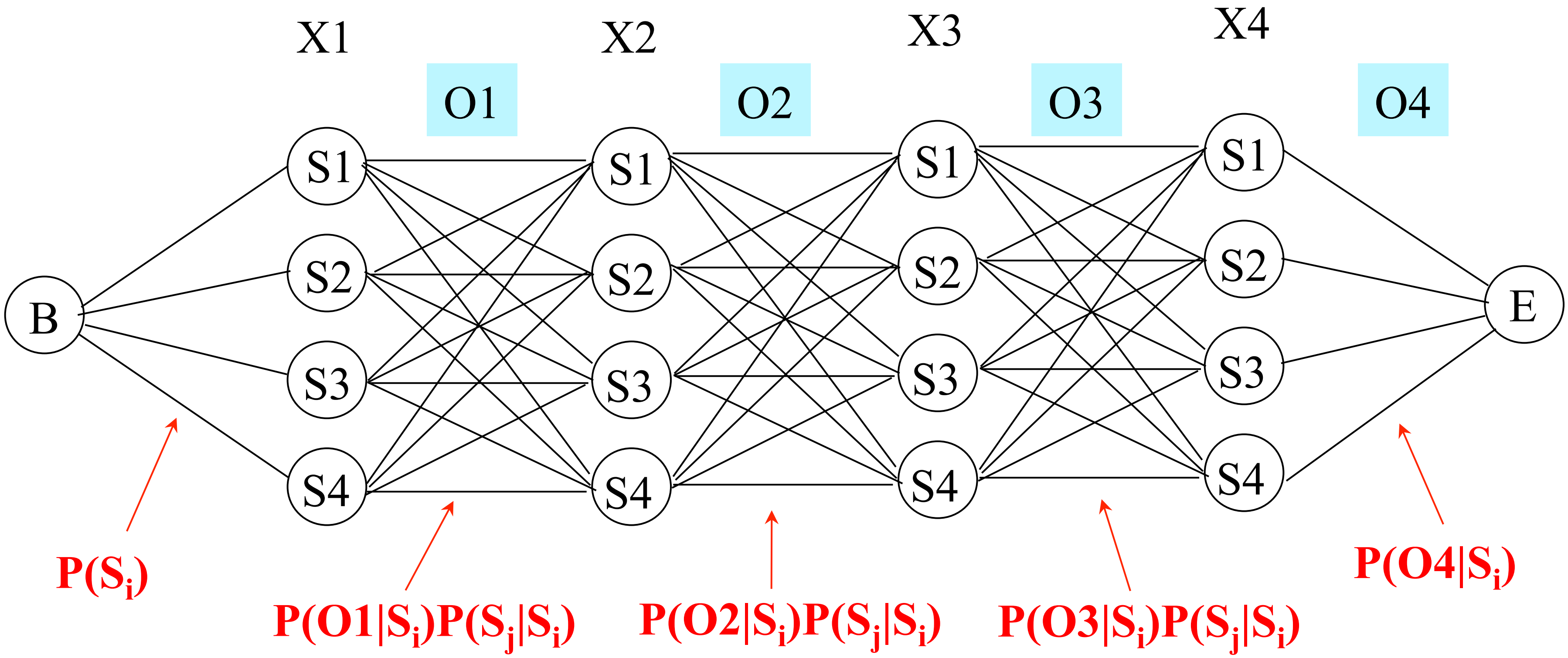
**Then** $X_{MAP} = \underset{X}{\text{argmax}}\ \Phi(X)$

**This is the max-product procedure of message passing. It is called the *Viterbi* algorithm in HMM terminology**

Robert Collins
CSE586

# Viterbi, under the hood

1) Build a state-space *trellis*. 2) Add a source and sink node.
3) Given observations. 4) Assign weights to edges based on observations.




**Do multistage dynamic programming to find the max product path.**

(note: in some implementations, each edge is weighted by $-\log(w_i)$ of the weights shown here, so that standard min length path DP can be performed.)

# HMM Problem 3

Given a *training set* of observation sequences

{{O1,O2,...}, {O1,O2,...}, {O1,O2,...}}

how do we determine the transition probs $a_{ij}$, initial state probs $p_i$, and observation probs $b_{ik}$?

This is a learning problem. It is the hardest problem of the three. We assume topology of the HMM is known (number and connectivity of the states) otherwise it is even harder.

**Two popular approaches:**

- **Segmental K-means algorithm**
- **Baum-Welch algorithm (EM-based)**

Recall clustering lectures earlier in the semester!

# Problem 3

- Given some training data, build a HMM
  - The training data is a set of observation sequences
  - We assume that these sequences are representative and independent
  - We want the HMM to be the one most likely to give rise to the training data
  - We'll look at a solution based on *k*-means, which uses the solution to Problem 2 to define the best HMM

# Assumptions

- We make a number of simplifying assumptions
  - Each observation symbol is a vector having some fixed length – this means all our observations have the same basic shape
  - Each observation is a series of *T* symbols – we don't need to consider varying lengths (one long sequence can be split up)
  - We know (or guess) the number of states, *n* (we can try different values if we like)

# Segmental K-Means

## Algorithm Overview

1. Initialise the HMM states and assign observation symbols to these states
2. Compute the initial state and transition functions given the current HMM
3. Compute some statistics about each state
4. Find the observation function for each state
5. Find the optimal path through the HMM for each observation sequence, and reassign its observation symbols as appropriate
6. If any changes have been made goto 2



Steve Mills, U.Nottingham

# Step 1

- We need to set up some initial states
  - We know there are *n* of them
  - Choose *n* (different) observation symbols (vectors) and assign these to the *n* states at random
  - Assign all the other observed vectors to the state which they are closest to (using Euclidean distance)



Note: this is essentially k-means clustering!

# Example

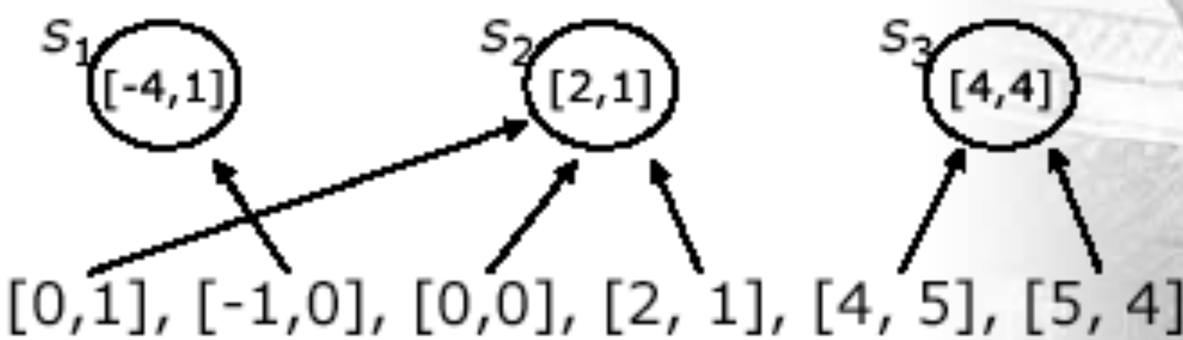
- Suppose our observation symbols are velocity measurements, [*u*, *v*]
  - We're given three observation sequences:
    - [0,1], [-1,0], [0,0], [2, 1], [4, 5], [5, 4]
    - [-3, 0], [-4, 1], [-3, -1], [-2, -1], [0, 0], [-1,1]
    - [-3,-1], [-1, 0], [0, 1], [3, 3], [4, 4], [4, 3]
  - We pick $k=3$ at random:
    - [0,1], [-1,0], [0,0], **[2, 1]**, [4, 5], [5, 4]
    - [-3, 0], **[-4, 1]**, [-3, -1], [-2, -1], [0, 0], [-1,1]
    - [-3,-1], [-1, 0], [0, 1], [3, 3], **[4, 4]**, [4, 3]



Steve Mills, U.Nottingham

# Example

- Each of these three vectors becomes a state, and we assign each symbol to the state nearest it







Steve Mills, U.Nottingham

# Example

- Doing this for all 3 sequences gives
  - In the state, $s_1$, centred around [-4,1]:
    - [-1,0], [-3, 0], [-4, 1], [-3, -1], [-2, -1], [-3,-1], [-1, 0]
  - In the state, $s_2$, centred around [2,1]:
    - [0,1], [0,0], [2, 1], [0, 0], [-1,1], [0, 1]
  - In the state, $s_3$, centred around [4,4]:
    - [4, 5], [5, 4] , [3, 3], [4, 4], [4, 3]

# Step 2

- We now compute the initial probabilities and transition functions
  - I($s$) is just the proportion of times a sequence starts in state $s$
  - T($s_i$,$s_j$) is the proportion of times a sequence takes us from state $s_i$ to state $s_j$

# Example

- Initial state function:
  - The first sequence starts with [0,1], which is in state $s_2$
  - The second sequence starts with [-3,0], which is in state $s_1$
  - The third sequence starts with [-3,-1], which is in state $s_1$
  - So $I(s_1)=2/3$, $I(s_2)=1/3$, $I(s_3)=0$



- We're given three observation sequences:
  - [0,1], [-1,0], [0,0], [2, 1], [4, 5], [5, 4]
  - [-3, 0], [-4, 1], [-3, -1], [-2, -1], [0, 0], [-1,1]
  - [-3,-1], [-1, 0], [0, 1], [3, 3], [4, 4], [4, 3]

# Example

- The transitions in the sequences are:

[0,1], [-1,0], [0,0], [2, 1], [4, 5], [5, 4]

$s_2 \rightarrow s_1 \rightarrow s_2 \rightarrow s_2 \rightarrow s_3 \rightarrow s_3$

[-3, 0], [-4, 1], [-3, -1], [-2, -1], [0, 0], [-1,1]

$s_1 \rightarrow s_1 \rightarrow s_1 \rightarrow s_2 \rightarrow s_2 \rightarrow s_2$

[-3,-1], [-1, 0], [0, 1], [3, 3], [4, 4], [4, 3]

$s_1 \rightarrow s_1 \rightarrow s_2 \rightarrow s_3 \rightarrow s_3 \rightarrow s_3$



- We're given three observation sequences:
  - [0,1], [-1,0], [0,0], [2, 1], [4, 5], [5, 4]
  - [-3, 0], [-4, 1], [-3, -1], [-2, -1], [0, 0], [-1,1]
  - [-3,-1], [-1, 0], [0, 1], [3, 3], [4, 4], [4, 3]

# Example

- 15 transitions:
  - 3 from $s_1$ to $s_1$
  - 3 from $s_1$ to $s_2$
  - 0 from $s_1$ to $s_3$
  - 1 from $s_2$ to $s_1$
  - 3 from $s_2$ to $s_2$
  - 2 from $s_2$ to $s_3$
  - 0 from $s_3$ to $s_1$
  - 0 from $s_3$ to $s_2$
  - 3 from $s_3$ to $s_3$
- Transition function
  - $T(s_1, s_1) = 3/15$
  - $T(s_1, s_2) = 3/15$
  - $T(s_1, s_3) = 0$
  - $T(s_2, s_1) = 1/15$
  - $T(s_2, s_2) = 3/15$
  - $T(s_2, s_3) = 2/15$
  - $T(s_3, s_1) = 0$
  - $T(s_3, s_2) = 0$
  - $T(s_3, s_3) = 3/15$

# Step 3

- Now we compute some statistics about the observations made in each state
  - Usually we compute the mean and variances or covariance
  - These are going to be used in Step 4, so if you change Step 4 then other statistics might be needed



**Note: This part assumes continuous-valued observations are output. For a discrete set of observation symbols, we could just do histograms here.**

# Example

- State $s_1$ has the vectors [-1,0], [-3, 0], [-4, 1], [-3, -1], [-2, -1], [-3,-1], and [-1, 0]
  - The mean of these vectors is
    $$\mu_1 = [-2.43,-0.29]$$
  - The variance of each component is
    $$\sigma_{u1}=1.1$$
    $$\sigma_{v1}=0.5$$
  - We compute this for each state

# Step 4

- The next step is to compute the observation probabilities
  - Since the observations are vectors, this is usually a probability function
  - Commonly a Gaussian model is used, but other models could be used if we want
  - Since the vectors are *d*-dimensional, this Gaussian is *d*-dimensional also



**Note: For a discrete set of observation symbols, we could use histograms normalized to sum to 1 as estimates of the probability mass function of outputting each symbol.**

Steve Mills, U.Nottingham

# Example

- Using the formula for a 2D Gaussian
  - We put in the mean and covariance computed in Step 3 for each state
  - This gives us a function of the observed symbol and state

$$O(a,s_t) = \frac{1}{2\pi\sigma_{ut}\sigma_{vt}} e^{-\left(\frac{(u_a-\mu_{ut})^2}{2\sigma_{ut}^2}+\frac{(v_a-\mu_{vt})^2}{2\sigma_{vt}^2}\right)}$$

# Step 5

- We now have estimates for the parameters of our HMM
    - As in normal *k*-means the next step is to reassign the observations to the states
    - We do this by running the Viterbi algorithm for each observation, giving an optimal assignment of observation symbols to states
    - If any observation has changed state then we go back to step 2 to revise our HMM

# Example

- Suppose the Viterbi algorithm gives the sequence $s_2 \rightarrow s_2 \rightarrow s_2 \rightarrow s_2 \rightarrow s_3 \rightarrow s_3$ as the best path for the first observation
  - The second symbol ([-1,0]) is now attributed to state $s_2$ rather than $s_1$
  - This affects the statistics related with these two states, and also the transition function
  - As a result, the paths explaining the observations may also change



- We're given three observation sequences:
  - [0,1], [-1,0], [0,0], [2, 1], [4, 5], [5, 4]
  - [-3, 0], [-4, 1], [-3, -1], [-2, -1], [0, 0], [-1,1]
  - [-3,-1], [-1, 0], [0, 1], [3, 3], [4, 4], [4, 3]

Steve Mills, U.Nottingham

# Training HMMs

- This algorithm, the *segmental k-means* algorithm, allows us to build a HMM from training data
  - It can be shown that this converges to an optimal result for a variety of observation functions (including Gaussian)
  - In these cases the starting point doesn't matter, although a poor choice might mean lots of iterations are needed

**Really?**



Steve Mills, U.Nottingham

# Baum-Welch Algorithm

Same thing, but with fractional assignments of observations to states.

Basic idea is to use EM instead of K-means as a way of assessing ownership weights before computing observation statistics at each state.

# Kalman Filter

Note: a good background reference is "An Introduction to the Kalman Filter", Greg Welch and Gary Bishop, University of North Carolina at Chapel Hill, Dept of Computer Science Tech Report 95-041.

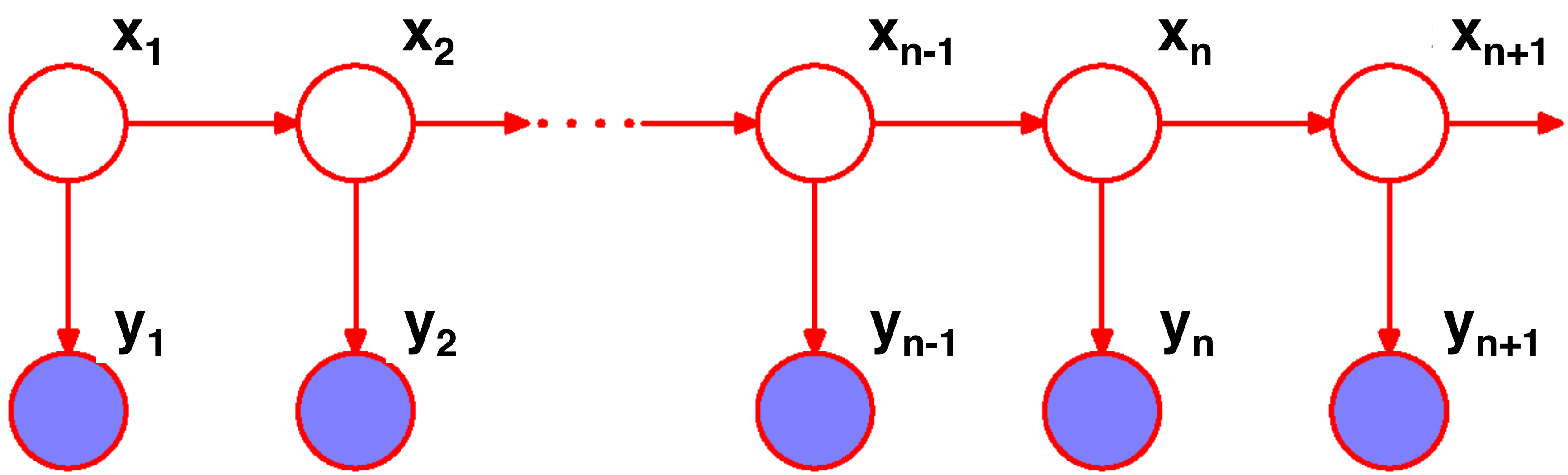
Welch also maintains an excellent website about all things Kalman:

http://www.cs.unc.edu/~welch/kalman/

# Kalman Filter for Tracking

P(x1,x2,x3,x4,...,y1,y2,y3,y4,...) =

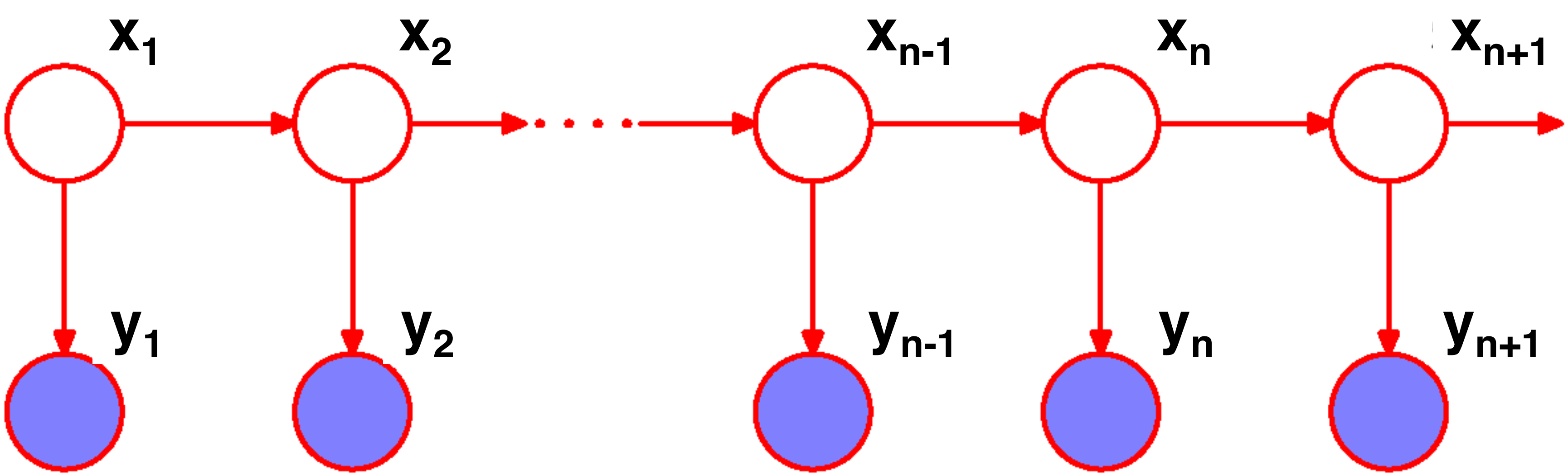
P(x1)P(y1|x1)P(x2|x1)P(y2|x2)P(x3|x2)P(y3|x3)P(x4|x3)P(y4|x4)......



# Kalman Filter for Tracking

P(x1,x2,x3,x4,...,y1,y2,y3,y4,...) =

P(x1)P(y1|x1)P(x2|x1)P(y2|x2)P(x3|x2)P(y3|x3)P(x4|x3)P(y4|x4)......




What we typically want to compute for tracking applications is P(xn | y1, y2, ..., yn)

# Linear Dynamical Systems

- Next state is a linear function of current state + zero-mean Gaussian noise
- Each observation is a linear function of current state + zero-mean Gaussian noise
- Initial prior distribution on first state is Gaussian

**=> All distributions remain Gaussian!**

**=> Means and covariances can be estimated over time by a Kalman filter**

# Kalman Filter Derivation (in 1D)

Assume:

$$\Rightarrow P(x_{k-1} \mid y_1 \cdots y_{k-1}) \sim N(\mu_{k-1}, \sigma^2_{k-1})$$

Linear motion and measurement models:

motion model

$$x_k = \underbrace{a\, x_{k-1} + b}_{\text{linear (affine actually) state transition}} + \underbrace{c}_{\text{zero-mean process noise}} \qquad c \sim N(0, r^2)$$

measurement model

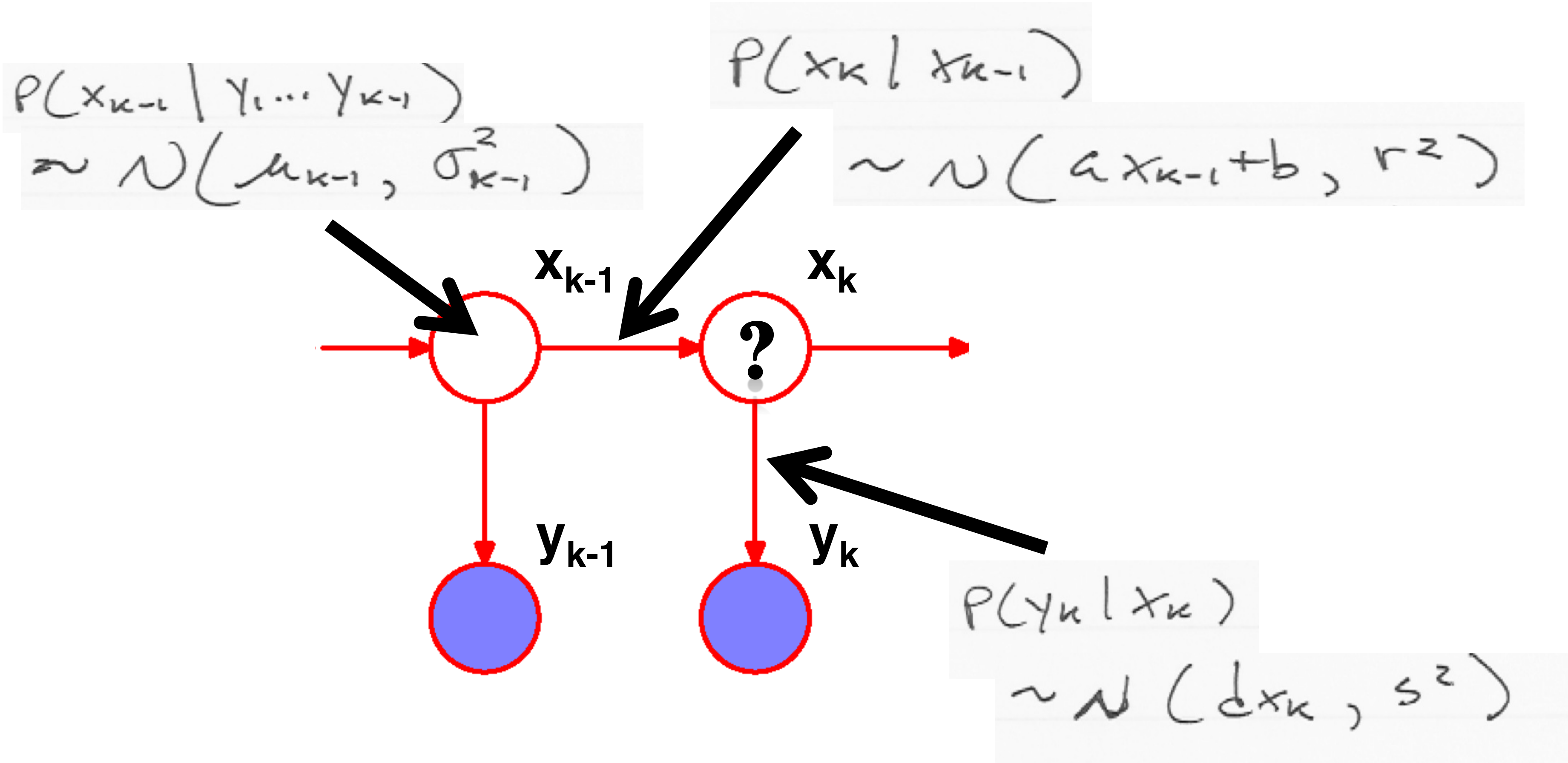
$$y_k = \underbrace{d\, x_k}_{\text{linear measurement}} + \underbrace{e}_{\text{zero-mean observation noise}} \qquad e \sim N(0, s^2)$$

Implied by motion and measurement models:

$$\Rightarrow P(x_k \mid x_{k-1}) \sim N(a x_{k-1} + b, r^2)$$

$$\Rightarrow P(y_k \mid x_k) \sim N(d x_k, s^2)$$

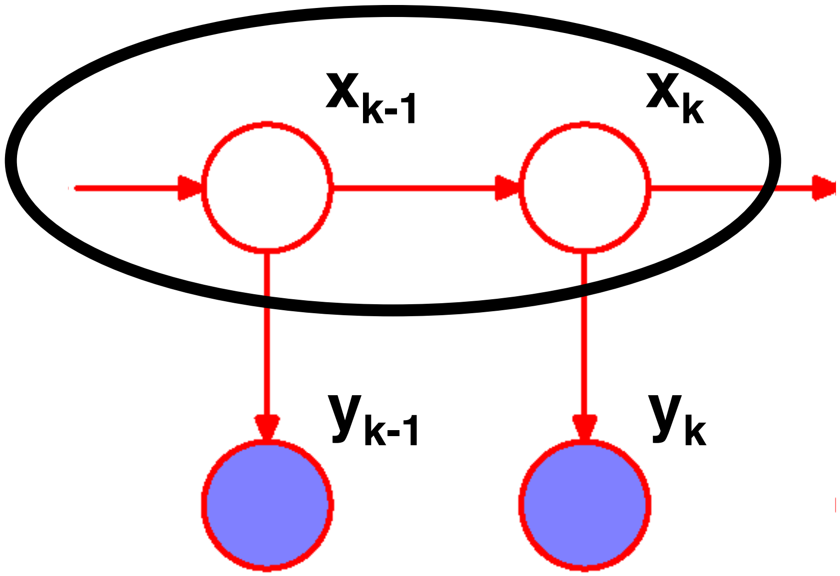
# Kalman Filter Derivation




**What is $P(x_k \mid y_1,\dots,y_k)$ ?**

# Derivation Strategy

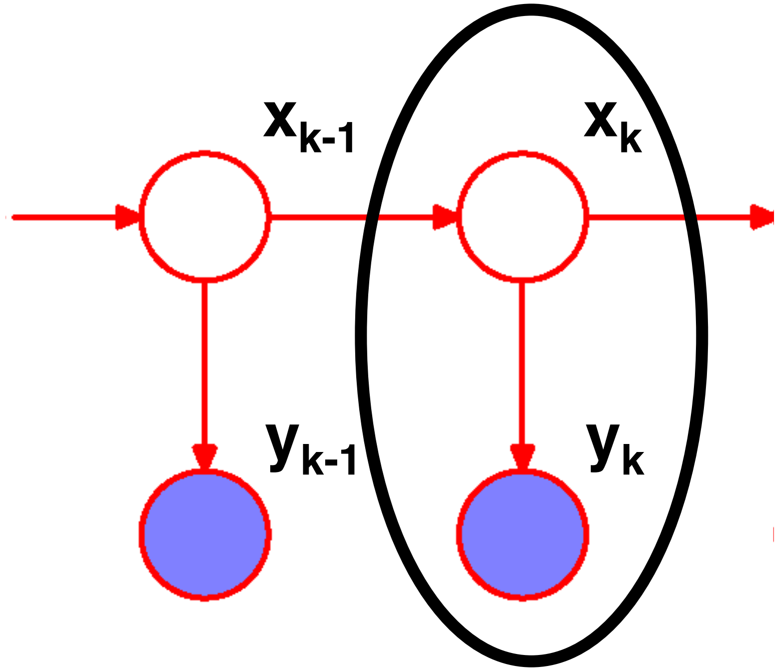
Combine $P(x_{k-1} \mid y_1,\ldots,y_{k-1})$ and $P(x_k|x_{k-1})$ to compute $P(x_k \mid y_1,\ldots,y_{k-1})$




Step 1: motion prediction

# Derivation Strategy

Combine $P(x_k | y_1,\ldots,y_{k-1})$ and $P(y_k|x_k)$ to compute $P(x_k | y_1,\ldots,y_k)$




Step 2: data correction

# Step 1: Motion Prediction

**Combine $P(x_{k-1} \mid y_1,\ldots,y_{k-1})$ and $P(x_k|x_{k-1})$ to compute $P(x_k \mid y_1,\ldots,y_{k-1})$**

$$P(x_k \mid y_1 \ldots y_{k-1}) = \int_{x_{k-1}} P(x_k \mid x_{k-1})\, P(x_{k-1} \mid y_1 \ldots y_{k-1})\, dx_{k-1}$$

$$= \int_{x_{k-1}} N(x_k \mid a x_{k-1} + b, r^2)\, N(x_{k-1} \mid \mu_{k-1}, \sigma_{k-1}^2)\, dx_{k-1}$$

$$= \text{constant} \int \exp\left\{ -\frac{1}{2} \left[ \frac{(a x_{k-1} + b - x_k)^2}{r^2} + \frac{(x_{k-1} - \mu_{k-1})^2}{\sigma_{k-1}^2} \right] \right\}$$

Quadratic form in $x_{k-1}$ and $x_k$. Therefore this is a joint Gaussian distribution over $x_{k-1}$ and $x_k$.

Integrated over $x_{k-1}$ yields marginal distribution $P(x_k|y_1,\ldots,y_{k-1})$

$$P(x_k \mid y_1 \ldots y_{k-1}) \sim N(a\mu_{k-1} + b,\ a^2\sigma_{k-1}^2 + r^2)$$

# Step 2: Data Correction

**Combine $P(x_k| {}_{y1,\ldots,yk-1})$ and $P(y_k|x_k)$ to compute $P(x_k \mid y_1,\ldots,y_k)$**

$$P(x_k \mid y_1 \ldots y_k) = \frac{P(y_k \mid x_k)\, P(x_k \mid y_1 \ldots y_{k-1})}{\int [\text{numerator}]}$$

This is just Bayes rule, applied to

$$P(x_k) \sim N(x_k \mid a\mu_{k-1} + b \,,\; a^2\sigma^2_{k-1} + r^2)$$

$$L(y_k \mid x_k) \sim N(y_k \mid d\, x_k \,,\; s^2)$$

**Through some algebra (completing the square), we find that:**

$$P(x_k \mid y_1 \ldots y_k) \sim N(\mu_k^+, \sigma_k^{2+})$$

where

$$\mu_k^+ = \frac{\sigma_k^{2-} \frac{y_k}{d} + \frac{s^2}{d^2}\mu_k^-}{\sigma_k^{2-} + \frac{s^2}{d^2}} = \frac{d\,\sigma_k^{2-} y_k + s^2 \mu_k^-}{d^2 \sigma_k^{2-} + s^2}$$

$$\sigma_k^{2+} = \frac{\sigma_k^{2-} \frac{s^2}{d^2}}{\sigma_k^{2-} + \frac{s^2}{d^2}} = \frac{\sigma_k^{2-} s^2}{d^2 \sigma_k^{2-} + s^2}$$

Robert Collins
CSE586

# Step 2: Data Correction

**Combine $P(x_k| y_1,\dots,y_{k-1})$ and $P(y_k|x_k)$ to compute $P(x_k \mid y_1,\dots,y_k)$**

$$P(x_k \mid y_1 \dots y_k) = \frac{P(y_k \mid x_k)\, P(x_k \mid y_1 \dots y_{k-1})}{\int [\text{numerator}]}$$

This is just Bayes rule, applied to

$$P(x_k) \sim N(x_k \mid a\mu_{k-1} + b,\; a^2\sigma_{k-1}^2 + r^2)$$

$$L(y_k \mid x_k) \sim N(y_k \mid d\, x_k,\; s^2)$$

**Through some algebra (completing the square), we find that:**

$$P(x_k \mid y_1 \dots y_k) \sim N(\mu_k^+, \sigma_k^{2+})$$

where

$\mu_k^+ =$

$\sigma_k^{2+} =$

This becomes the Gaussian prior for the next iteration of motion prediction and data correction. Thus we continue propagating forward into subsequent time steps.