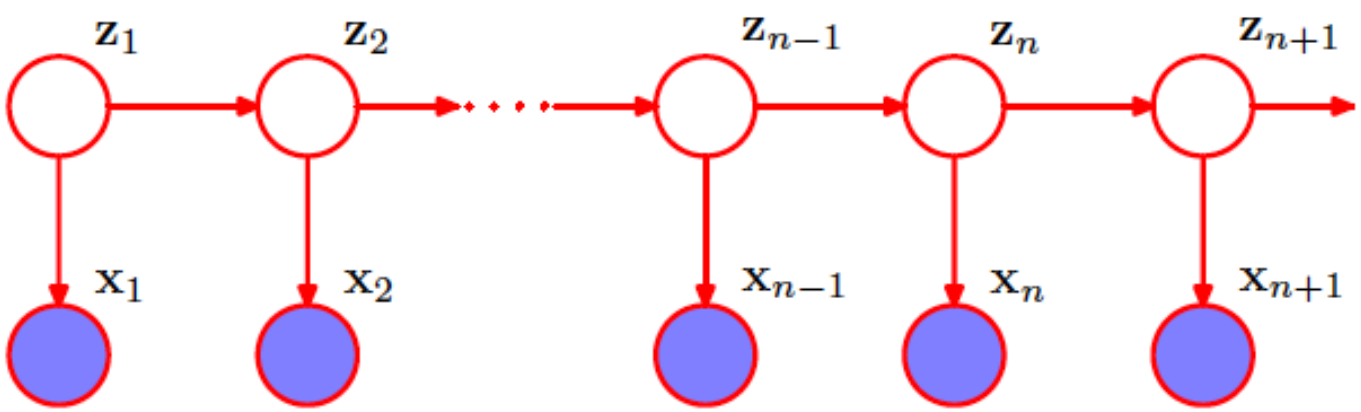
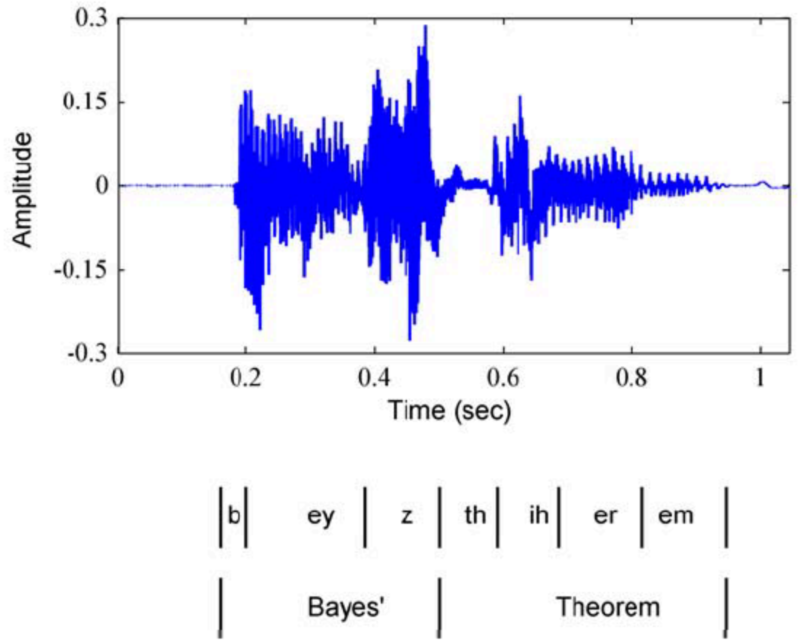
Hidden Markov Model (HMM)



Some of figures in the slides are from Bishop (2016)

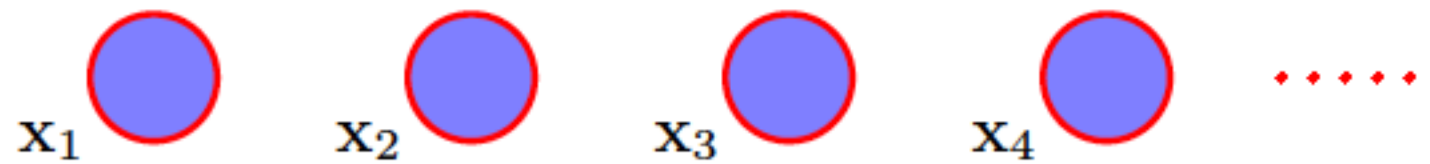
Hidden Markov Model

- A directed graphical model
- Widely used for modeling **sequential data** (beyond i.i.d assumption), e.g. speech recognition, natural language processing, etc.
- State space models: with latent variable for indicating the state of the observed data
- The latent variable of HMM is discrete
- **Markov assumption**



Markov Model

- The general: $$p(\mathbf{x}_1, \ldots, \mathbf{x}_N) = \prod_{n=1}^{N} p(\mathbf{x}_n | \mathbf{x}_1, \ldots, \mathbf{x}_{n-1})$$
- I.I.D. data

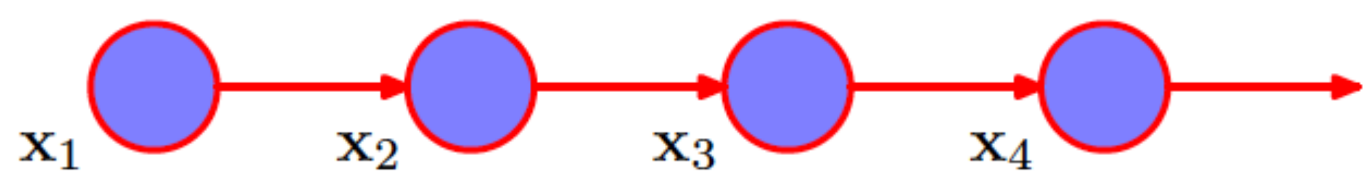


- First-order Markov chain

K(K-1) parameters

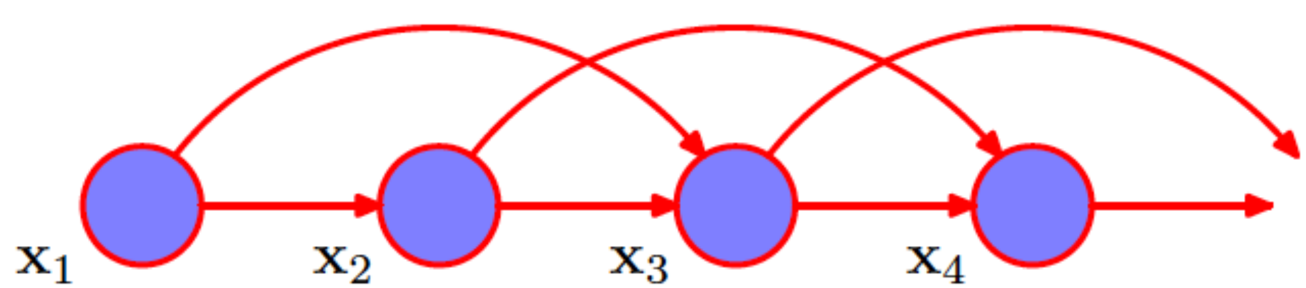
$$p(\mathbf{x}_1, \ldots, \mathbf{x}_N) = p(\mathbf{x}_1) \prod_{n=2}^{N} p(\mathbf{x}_n | \mathbf{x}_{n-1})$$

$$p(\mathbf{x}_n | \mathbf{x}_1, \ldots, \mathbf{x}_{n-1}) = p(\mathbf{x}_n | \mathbf{x}_{n-1})$$



- Second-order Markov chain

M-order: K^M(K-1) parameters



Hidden Markov Model

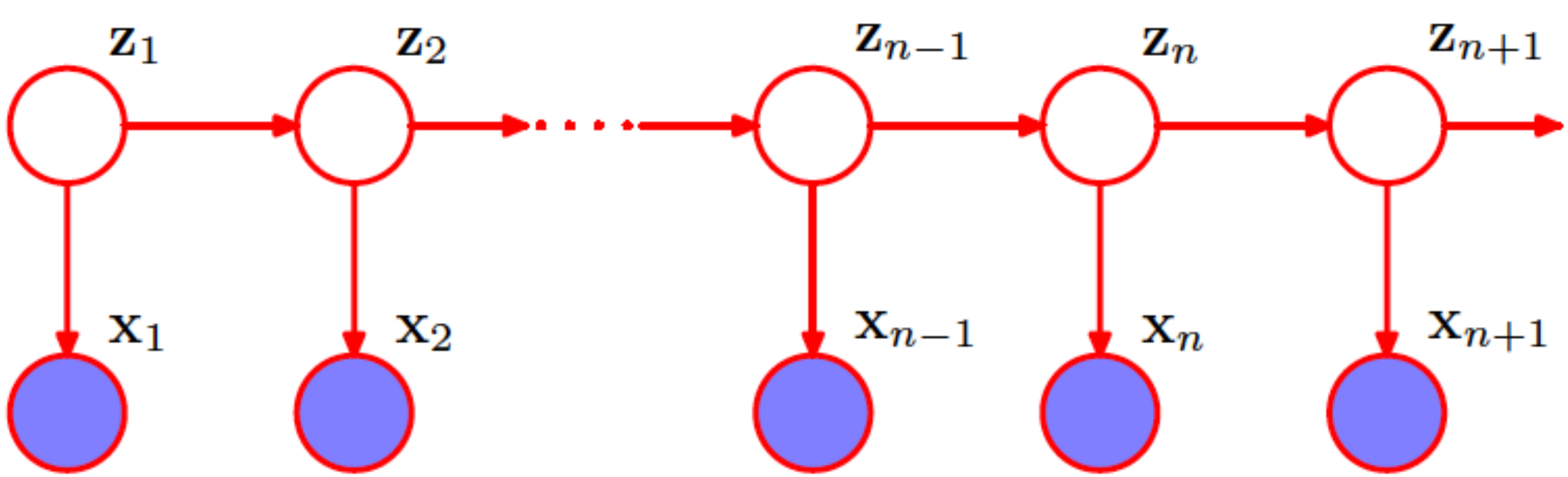
- State space model
  - If **z is discrete, HMM**, otherwise, linear dynamical system

$$\mathbf{z}_{n+1} \perp\!\!\!\perp \mathbf{z}_{n-1} \mid \mathbf{z}_n$$

$$p(\mathbf{x}_1, \ldots, \mathbf{x}_N, \mathbf{z}_1, \ldots, \mathbf{z}_N) = p(\mathbf{z}_1) \left[\prod_{n=2}^{N} p(\mathbf{z}_n | \mathbf{z}_{n-1}) \right] \prod_{n=1}^{N} p(\mathbf{x}_n | \mathbf{z}_n)$$

Question: do the observed variables satisfy Markov property?

$$p(\mathbf{x}_{n+1} | \mathbf{x}_1, \ldots, \mathbf{x}_n)$$



- Transition probabilities

$$A_{jk} \equiv p(z_{nk} = 1 | z_{n-1,j} = 1)$$

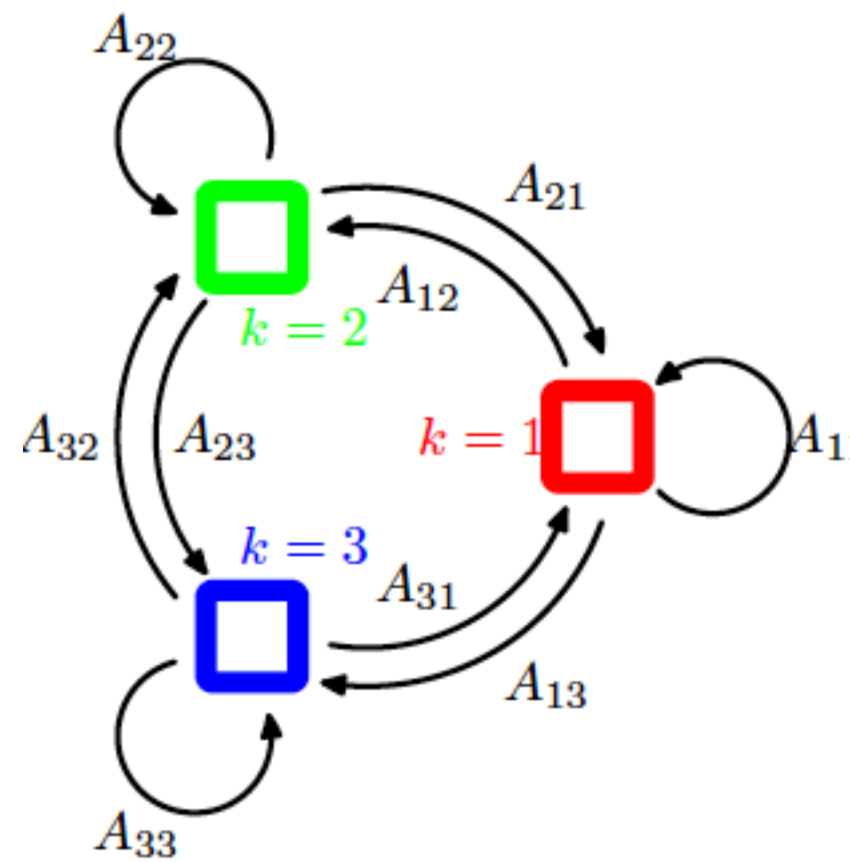
$$0 \leqslant A_{jk} \leqslant 1 \text{ with } \sum_k A_{jk} = 1$$

$$p(\mathbf{z}_n | \mathbf{z}_{n-1}, \mathbf{A}) = \prod_{k=1}^{K} \prod_{j=1}^{K} A_{jk}^{z_{n-1,j} z_{nk}}$$

$$p(\mathbf{z}_1 | \boldsymbol{\pi}) = \prod_{k=1}^{K} \pi_k^{z_{1k}}$$

- Emission probabilities

$$p(\mathbf{x}_n | \mathbf{z}_n, \phi) = \prod_{k=1}^{K} p(\mathbf{x}_n | \phi_k)^{z_{nk}}$$



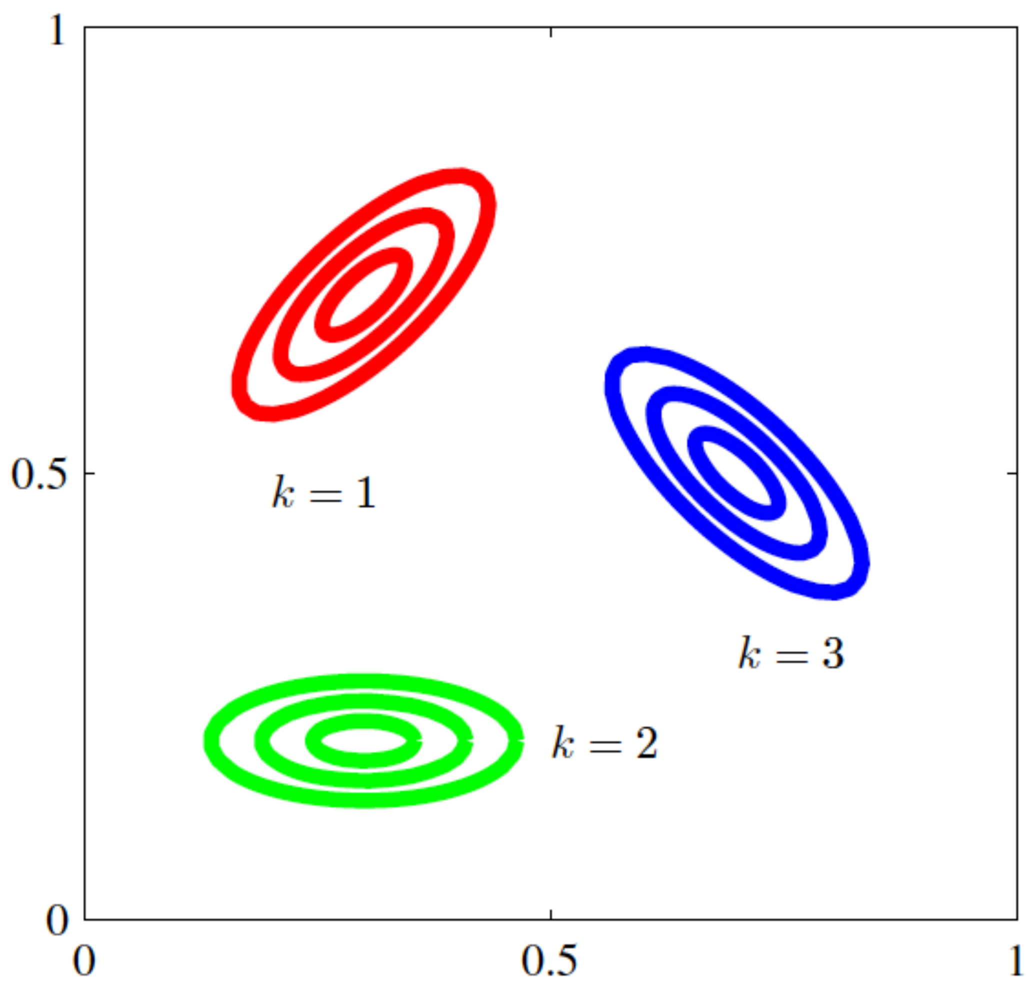
Homogeneous HMM:
transition and emission distributions
are the same for all times steps

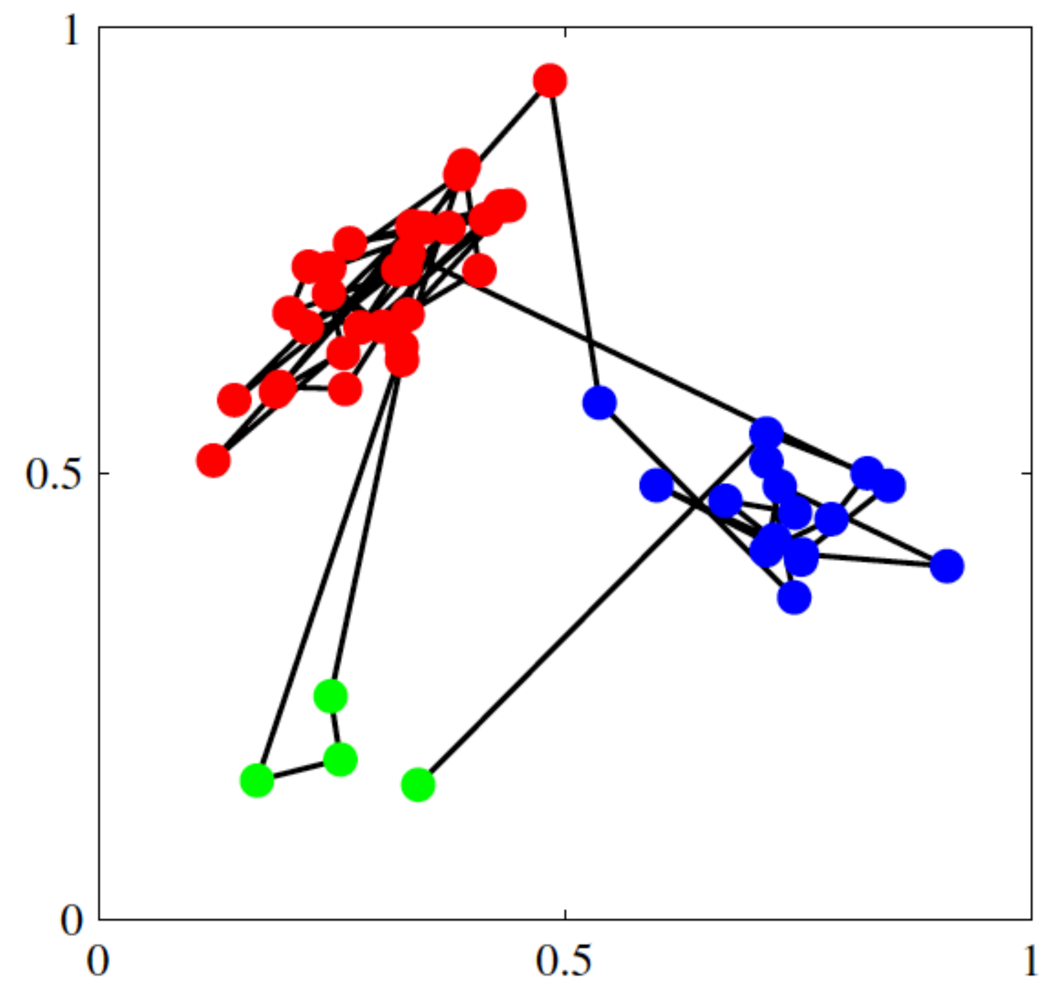
- Joint distribution

$$p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\theta}) = p(\mathbf{z}_1|\boldsymbol{\pi}) \left[\prod_{n=2}^{N} p(\mathbf{z}_n|\mathbf{z}_{n-1}, \mathbf{A})\right] \prod_{m=1}^{N} p(\mathbf{x}_m|\mathbf{z}_m, \phi)$$

- Generative process

$$\boldsymbol{\theta} = \{\boldsymbol{\pi}, \mathbf{A}, \phi\}$$





Learning HMM

- Maximum likelihood solution: naive summation induces exponential computation w.r.t. length of chain. INTRACTABLE!

$$p(\mathbf{X}|\boldsymbol{\theta}) = \sum_{\mathbf{Z}} p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\theta})$$

- EM for maximizing likelihood

$$Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}}) = \sum_{\mathbf{Z}} p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\theta}^{\text{old}}) \ln p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\theta})$$

E step

$$\gamma(\mathbf{z}_n) = p(\mathbf{z}_n|\mathbf{X}, \boldsymbol{\theta}^{\text{old}})$$

K-dimensional vector

$$\xi(\mathbf{z}_{n-1}, \mathbf{z}_n) = p(\mathbf{z}_{n-1}, \mathbf{z}_n|\mathbf{X}, \boldsymbol{\theta}^{\text{old}})$$

K*K matrix

$$\begin{aligned}\gamma(z_{nk}) &= \mathbb{E}[z_{nk}] = \sum_{\mathbf{z}} \gamma(\mathbf{z}) z_{nk} \\ \xi(z_{n-1,j}, z_{nk}) &= \mathbb{E}[z_{n-1,j} z_{nk}] = \sum_{\mathbf{z}} \gamma(\mathbf{z}) z_{n-1,j} z_{nk}\end{aligned}$$

$$Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}}) = \sum_{k=1}^{K} \gamma(z_{1k}) \ln \pi_k + \sum_{n=2}^{N} \sum_{j=1}^{K} \sum_{k=1}^{K} \xi(z_{n-1,j}, z_{nk}) \ln A_{jk} + \sum_{n=1}^{N} \sum_{k=1}^{K} \gamma(z_{nk}) \ln p(\mathbf{x}_n | \boldsymbol{\phi}_k).$$

if we can evaluate these terms efficiently

$$\pi_k = \frac{\gamma(z_{1k})}{\sum_{j=1}^{K} \gamma(z_{1j})}$$

M step

$$A_{jk} = \frac{\sum_{n=2}^{N} \xi(z_{n-1,j}, z_{nk})}{\sum_{l=1}^{K} \sum_{n=2}^{N} \xi(z_{n-1,j}, z_{nl})}$$

If Gaussian emission distribution $p(\mathbf{x}|\phi_k) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$

$$\boldsymbol{\mu}_k = \frac{\sum_{n=1}^{N} \gamma(z_{nk})\mathbf{x}_n}{\sum_{n=1}^{N} \gamma(z_{nk})}$$

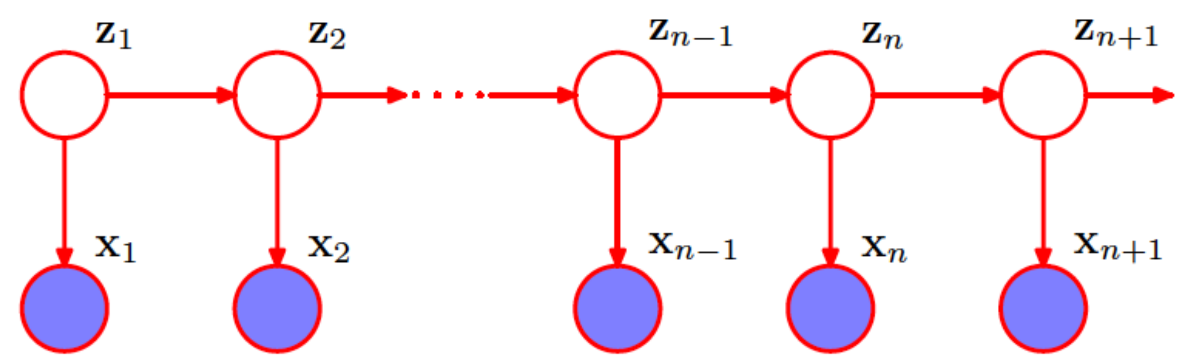
$$\boldsymbol{\Sigma}_k = \frac{\sum_{n=1}^{N} \gamma(z_{nk})(\mathbf{x}_n - \boldsymbol{\mu}_k)(\mathbf{x}_n - \boldsymbol{\mu}_k)^{\mathrm{T}}}{\sum_{n=1}^{N} \gamma(z_{nk})}$$

If multinomial distribution $p(\mathbf{x}|\mathbf{z}) = \prod_{i=1}^{D} \prod_{k=1}^{K} \mu_{ik}^{x_i z_k}$

$$u_{ik} = \frac{\sum_{n=1}^{N} \gamma(z_{nk}) x_{ni}}{\sum_{n=1}^{N} \gamma(z_{nk})}$$

The Forward-Backward Algorithm

- Also known as Baum-Welch algorithm, we focus on alpha-beta variant
- Evaluating $\gamma(z_{nk})$ and $\xi(z_{n-1,j}, z_{nk})$
- Let's derive **alpha-beta algorithm** step by step:



Conditional independence by D-separation

$$
\begin{aligned}
p(\mathbf{X}|\mathbf{z}_n) &= p(\mathbf{x}_1,\ldots,\mathbf{x}_n|\mathbf{z}_n) \\
&\quad\; p(\mathbf{x}_{n+1},\ldots,\mathbf{x}_N|\mathbf{z}_n) \\
p(\mathbf{x}_1,\ldots,\mathbf{x}_{n-1}|\mathbf{x}_n,\mathbf{z}_n) &= p(\mathbf{x}_1,\ldots,\mathbf{x}_{n-1}|\mathbf{z}_n) \\
p(\mathbf{x}_1,\ldots,\mathbf{x}_{n-1}|\mathbf{z}_{n-1},\mathbf{z}_n) &= p(\mathbf{x}_1,\ldots,\mathbf{x}_{n-1}|\mathbf{z}_{n-1}) \\
p(\mathbf{x}_{n+1},\ldots,\mathbf{x}_N|\mathbf{z}_n,\mathbf{z}_{n+1}) &= p(\mathbf{x}_{n+1},\ldots,\mathbf{x}_N|\mathbf{z}_{n+1}) \\
p(\mathbf{x}_{n+2},\ldots,\mathbf{x}_N|\mathbf{z}_{n+1},\mathbf{x}_{n+1}) &= p(\mathbf{x}_{n+2},\ldots,\mathbf{x}_N|\mathbf{z}_{n+1}) \\
p(\mathbf{X}|\mathbf{z}_{n-1},\mathbf{z}_n) &= p(\mathbf{x}_1,\ldots,\mathbf{x}_{n-1}|\mathbf{z}_{n-1}) \\
&\quad\; p(\mathbf{x}_n|\mathbf{z}_n)p(\mathbf{x}_{n+1},\ldots,\mathbf{x}_N|\mathbf{z}_n) \\
p(\mathbf{x}_{N+1}|\mathbf{X},\mathbf{z}_{N+1}) &= p(\mathbf{x}_{N+1}|\mathbf{z}_{N+1}) \\
p(\mathbf{z}_{N+1}|\mathbf{z}_N,\mathbf{X}) &= p(\mathbf{z}_{N+1}|\mathbf{z}_N)
\end{aligned}
$$

Evaluate $\gamma(z_{nk})$.

P(X) will be canceled in EM.

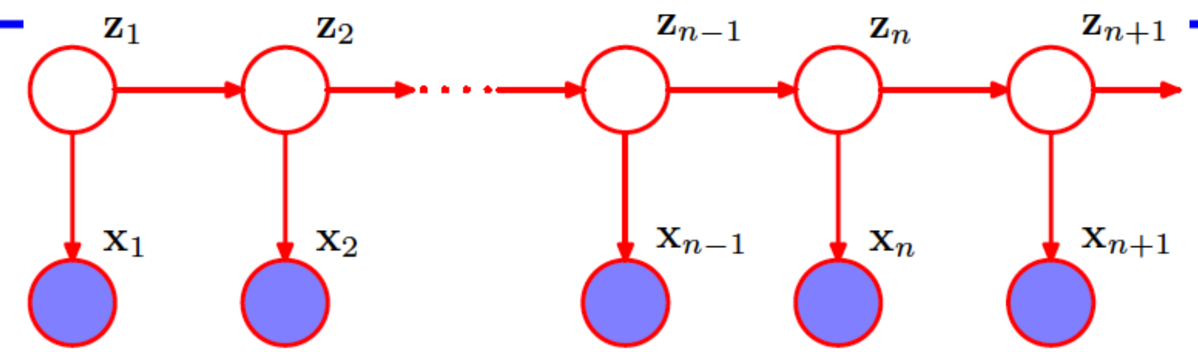
$$\gamma(\mathbf{z}_n) = p(\mathbf{z}_n|\mathbf{X}) = \frac{p(\mathbf{X}|\mathbf{z}_n)p(\mathbf{z}_n)}{p(\mathbf{X})}$$

$$\gamma(\mathbf{z}_n) = \frac{p(\mathbf{x}_1,\ldots,\mathbf{x}_n,\mathbf{z}_n)p(\mathbf{x}_{n+1},\ldots,\mathbf{x}_N|\mathbf{z}_n)}{p(\mathbf{X})} = \frac{\alpha(\mathbf{z}_n)\beta(\mathbf{z}_n)}{p(\mathbf{X})}$$

Represent set of K numbers

$$\begin{aligned}\alpha(\mathbf{z}_n) &\equiv p(\mathbf{x}_1,\ldots,\mathbf{x}_n,\mathbf{z}_n)\\ \beta(\mathbf{z}_n) &\equiv p(\mathbf{x}_{n+1},\ldots,\mathbf{x}_N|\mathbf{z}_n)\end{aligned}$$

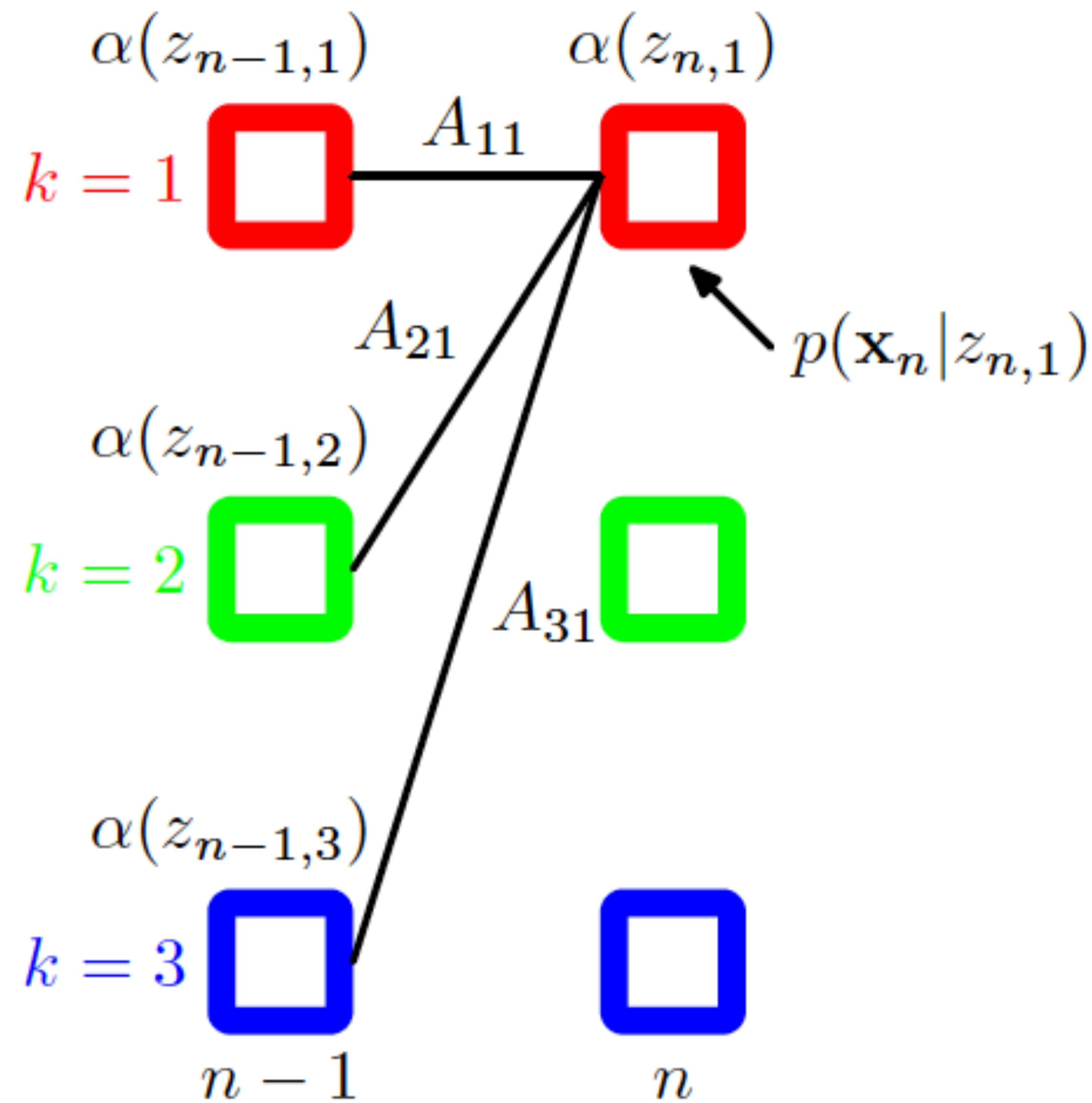
- Recursive formula for alpha



$$
\begin{aligned}
\alpha(\mathbf{z}_n) &= p(\mathbf{x}_1,\ldots,\mathbf{x}_n,\mathbf{z}_n) \\
&= p(\mathbf{x}_1,\ldots,\mathbf{x}_n|\mathbf{z}_n)p(\mathbf{z}_n) \\
&= p(\mathbf{x}_n|\mathbf{z}_n)p(\mathbf{x}_1,\ldots,\mathbf{x}_{n-1}|\mathbf{z}_n)p(\mathbf{z}_n) \\
&= p(\mathbf{x}_n|\mathbf{z}_n)p(\mathbf{x}_1,\ldots,\mathbf{x}_{n-1},\mathbf{z}_n) \\
&= p(\mathbf{x}_n|\mathbf{z}_n)\sum_{\mathbf{z}_{n-1}} p(\mathbf{x}_1,\ldots,\mathbf{x}_{n-1},\mathbf{z}_{n-1},\mathbf{z}_n) \\
&= p(\mathbf{x}_n|\mathbf{z}_n)\sum_{\mathbf{z}_{n-1}} p(\mathbf{x}_1,\ldots,\mathbf{x}_{n-1},\mathbf{z}_n|\mathbf{z}_{n-1})p(\mathbf{z}_{n-1}) \\
&= p(\mathbf{x}_n|\mathbf{z}_n)\sum_{\mathbf{z}_{n-1}} p(\mathbf{x}_1,\ldots,\mathbf{x}_{n-1}|\mathbf{z}_{n-1})p(\mathbf{z}_n|\mathbf{z}_{n-1})p(\mathbf{z}_{n-1}) \\
&= p(\mathbf{x}_n|\mathbf{z}_n)\sum_{\mathbf{z}_{n-1}} p(\mathbf{x}_1,\ldots,\mathbf{x}_{n-1},\mathbf{z}_{n-1})p(\mathbf{z}_n|\mathbf{z}_{n-1})
\end{aligned}
$$

$$
\alpha(\mathbf{z}_n) = p(\mathbf{x}_n|\mathbf{z}_n)\sum_{\mathbf{z}_{n-1}} \alpha(\mathbf{z}_{n-1})p(\mathbf{z}_n|\mathbf{z}_{n-1})
$$

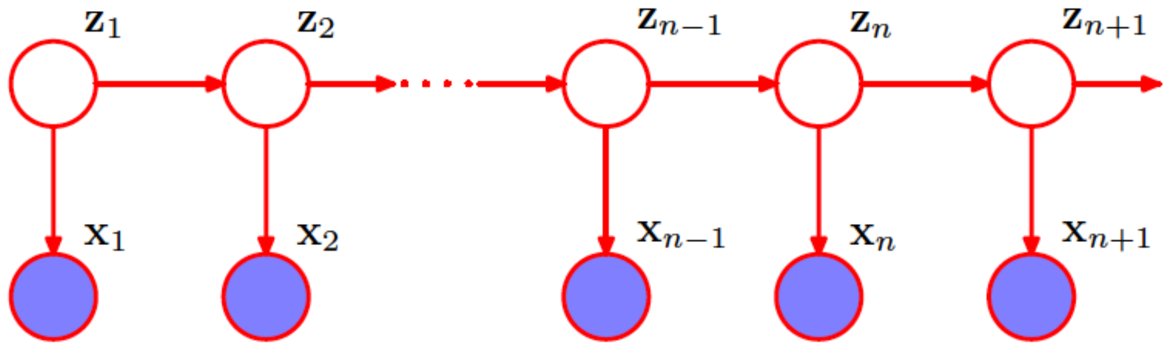
Illustration of the forward recursion evaluation of the α variables. In this fragment of the lattice, we see that the quantity $\alpha(z_{n1})$ is obtained by taking the elements $\alpha(z_{n-1,j})$ of $\alpha(\mathbf{z}_{n-1})$ at step $n-1$ and summing them up with weights given by A_{j1}, corresponding to the values of $p(\mathbf{z}_n|\mathbf{z}_{n-1})$, and then multiplying by the data contribution $p(\mathbf{x}_n|z_{n1})$.



Initial condition

$$\alpha(\mathbf{z}_1) = p(\mathbf{x}_1, \mathbf{z}_1) = p(\mathbf{z}_1)p(\mathbf{x}_1|\mathbf{z}_1) = \prod_{k=1}^{K} \{\pi_k p(\mathbf{x}_1|\phi_k)\}^{z_{1k}}$$

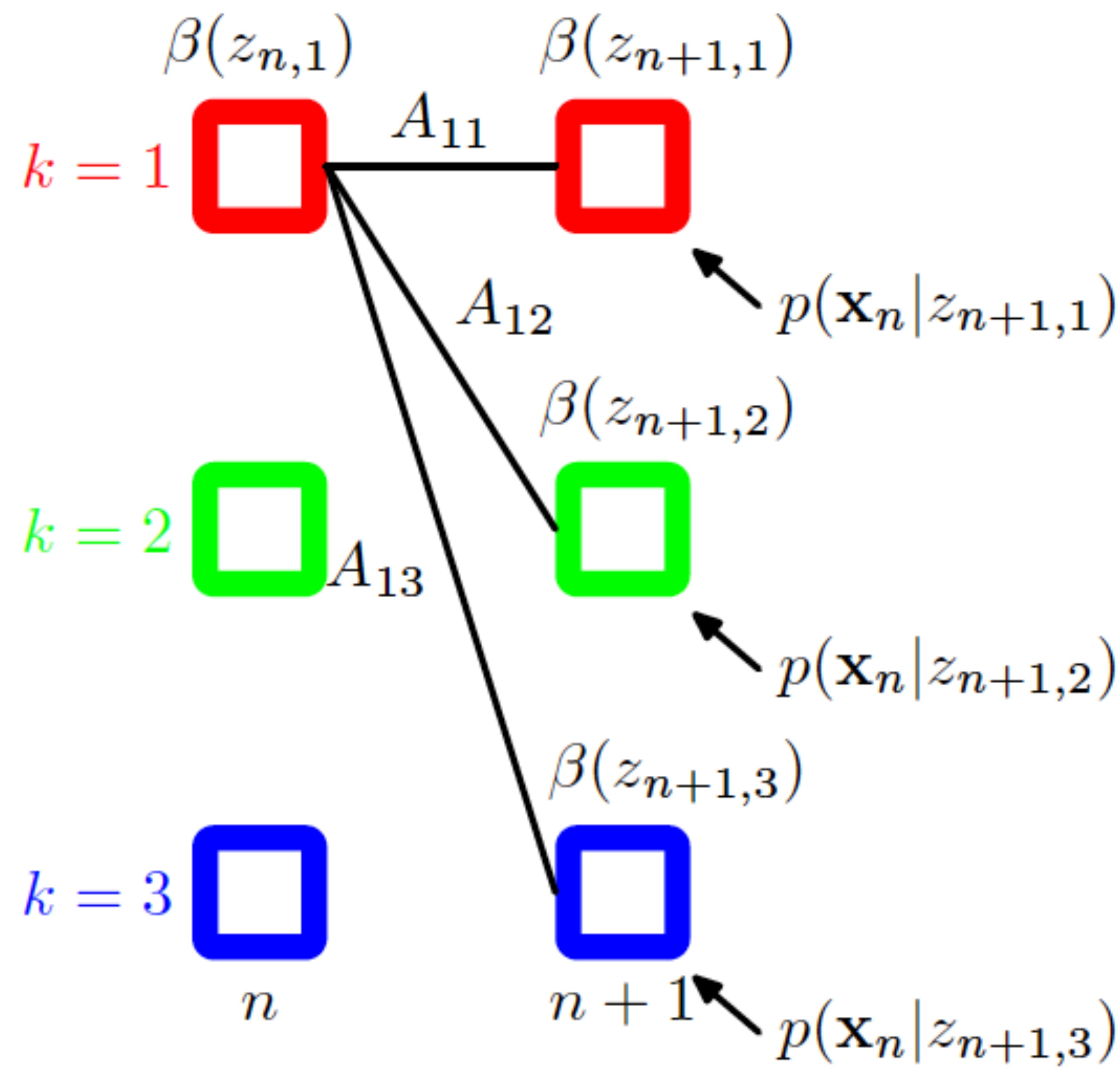
- Recursive formula for beta



$$
\begin{aligned}
\beta(\mathbf{z}_n) &= p(\mathbf{x}_{n+1}, \ldots, \mathbf{x}_N | \mathbf{z}_n) \\
&= \sum_{\mathbf{z}_{n+1}} p(\mathbf{x}_{n+1}, \ldots, \mathbf{x}_N, \mathbf{z}_{n+1} | \mathbf{z}_n) \\
&= \sum_{\mathbf{z}_{n+1}} p(\mathbf{x}_{n+1}, \ldots, \mathbf{x}_N | \mathbf{z}_n, \mathbf{z}_{n+1}) p(\mathbf{z}_{n+1} | \mathbf{z}_n) \\
&= \sum_{\mathbf{z}_{n+1}} p(\mathbf{x}_{n+1}, \ldots, \mathbf{x}_N | \mathbf{z}_{n+1}) p(\mathbf{z}_{n+1} | \mathbf{z}_n) \\
&= \sum_{\mathbf{z}_{n+1}} p(\mathbf{x}_{n+2}, \ldots, \mathbf{x}_N | \mathbf{z}_{n+1}) p(\mathbf{x}_{n+1} | \mathbf{z}_{n+1}) p(\mathbf{z}_{n+1} | \mathbf{z}_n)
\end{aligned}
$$

$$
\beta(\mathbf{z}_n) = \sum_{\mathbf{z}_{n+1}} \beta(\mathbf{z}_{n+1}) p(\mathbf{x}_{n+1} | \mathbf{z}_{n+1}) p(\mathbf{z}_{n+1} | \mathbf{z}_n)
$$

Illustration of the backward recursion for evaluation of the β variables. In this fragment of the lattice, we see that the quantity $\beta(z_{n1})$ is obtained by taking the components $\beta(z_{n+1,k})$ of $\beta(\mathbf{z}_{n+1})$ at step $n+1$ and summing them up with weights given by the products of A_{1k}, corresponding to the values of $p(\mathbf{z}_{n+1}|\mathbf{z}_n)$ and the corresponding values of the emission density $p(\mathbf{x}_n|z_{n+1,k})$.



Initial condition

$$p(\mathbf{z}_N|\mathbf{X}) = \frac{p(\mathbf{X}, \mathbf{z}_N)\overset{1}{\beta(\mathbf{z}_N)}}{p(\mathbf{X})}$$

$$Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}}) = \sum_{k=1}^{K} \gamma(z_{1k}) \ln \pi_k + \sum_{n=2}^{N} \sum_{j=1}^{K} \sum_{k=1}^{K} \xi(z_{n-1,j}, z_{nk}) \ln A_{jk} + \sum_{n=1}^{N} \sum_{k=1}^{K} \gamma(z_{nk}) \ln p(\mathbf{x}_n|\boldsymbol{\phi}_k).$$

M step

$$\boldsymbol{\mu}_k = \frac{\sum_{n=1}^{n} \gamma(z_{nk}) \mathbf{x}_n}{\sum_{n=1}^{n} \gamma(z_{nk})} = \frac{\sum_{n=1}^{n} \alpha(z_{nk}) \beta(z_{nk}) \mathbf{x}_n}{\sum_{n=1}^{n} \alpha(z_{nk}) \beta(z_{nk})}$$

$$p(\mathbf{X}) = \sum_{\mathbf{z}_n} \alpha(\mathbf{z}_n) \beta(\mathbf{z}_n)$$

$$p(\mathbf{X}) = \sum_{\mathbf{z}_N} \alpha(\mathbf{z}_N)$$

Alpha-beta recursion can still be used for evaluating:

$$\xi(\mathbf{z}_{n-1}, \mathbf{z}_n) = p(\mathbf{z}_{n-1}, \mathbf{z}_n | \mathbf{X})$$

$$= \frac{p(\mathbf{X}|\mathbf{z}_{n-1}, \mathbf{z}_n)p(\mathbf{z}_{n-1}, \mathbf{z}_n)}{p(\mathbf{X})}$$

$$= \frac{p(\mathbf{x}_1, \ldots, \mathbf{x}_{n-1}|\mathbf{z}_{n-1})p(\mathbf{x}_n|\mathbf{z}_n)p(\mathbf{x}_{n+1}, \ldots, \mathbf{x}_N|\mathbf{z}_n)p(\mathbf{z}_n|\mathbf{z}_{n-1})p(\mathbf{z}_{n-1})}{p(\mathbf{X})}$$

$$= \frac{\alpha(\mathbf{z}_{n-1})p(\mathbf{x}_n|\mathbf{z}_n)p(\mathbf{z}_n|\mathbf{z}_{n-1})\beta(\mathbf{z}_n)}{p(\mathbf{X})}$$

Summary for learning maximum likelihood solution for HMM

1. Initialize the parameters $\boldsymbol{\theta} = \{\boldsymbol{\pi}, \boldsymbol{A}, \boldsymbol{\phi}\}$;
2. Run α and β-recursion to evaluate $\gamma(\boldsymbol{z}_n)$ and $\xi(\boldsymbol{z}_{n-1}, \boldsymbol{z}_n)$, and obtain the Q-function;
3. Maximize the Q-function to update the parameters.

Iterate

Predictive Distribution

$$
\begin{aligned}
p(\mathbf{x}_{N+1}|\mathbf{X}) &= \sum_{\mathbf{z}_{N+1}} p(\mathbf{x}_{N+1}, \mathbf{z}_{N+1}|\mathbf{X}) \\
&= \sum_{\mathbf{z}_{N+1}} p(\mathbf{x}_{N+1}|\mathbf{z}_{N+1}) p(\mathbf{z}_{N+1}|\mathbf{X}) \\
&= \sum_{\mathbf{z}_{N+1}} p(\mathbf{x}_{N+1}|\mathbf{z}_{N+1}) \sum_{\mathbf{z}_N} p(\mathbf{z}_{N+1}, \mathbf{z}_N|\mathbf{X}) \\
&= \sum_{\mathbf{z}_{N+1}} p(\mathbf{x}_{N+1}|\mathbf{z}_{N+1}) \sum_{\mathbf{z}_N} p(\mathbf{z}_{N+1}|\mathbf{z}_N) p(\mathbf{z}_N|\mathbf{X}) \\
&= \sum_{\mathbf{z}_{N+1}} p(\mathbf{x}_{N+1}|\mathbf{z}_{N+1}) \sum_{\mathbf{z}_N} p(\mathbf{z}_{N+1}|\mathbf{z}_N) \frac{p(\mathbf{z}_N, \mathbf{X})}{p(\mathbf{X})} \\
&= \frac{1}{p(\mathbf{X})} \sum_{\mathbf{z}_{N+1}} p(\mathbf{x}_{N+1}|\mathbf{z}_{N+1}) \sum_{\mathbf{z}_N} p(\mathbf{z}_{N+1}|\mathbf{z}_N) \alpha(\mathbf{z}_N)
\end{aligned}
$$

Is there any practical issues regarding to alpha and beta recursion?

$$\alpha(\mathbf{z}_n) = p(\mathbf{x}_n|\mathbf{z}_n) \sum_{\mathbf{z}_{n-1}} \alpha(\mathbf{z}_{n-1}) p(\mathbf{z}_n|\mathbf{z}_{n-1})$$

$$\beta(\mathbf{z}_n) = \sum_{\mathbf{z}_{n+1}} \beta(\mathbf{z}_{n+1}) p(\mathbf{x}_{n+1}|\mathbf{z}_{n+1}) p(\mathbf{z}_{n+1}|\mathbf{z}_n)$$

Is there any practical issues regarding to alpha and beta recursion?

$$\alpha(\mathbf{z}_n) = p(\mathbf{x}_n|\mathbf{z}_n) \sum_{\mathbf{z}_{n-1}} \alpha(\mathbf{z}_{n-1}) p(\mathbf{z}_n|\mathbf{z}_{n-1})$$

$$\beta(\mathbf{z}_n) = \sum_{\mathbf{z}_{n+1}} \beta(\mathbf{z}_{n+1}) p(\mathbf{x}_{n+1}|\mathbf{z}_{n+1}) p(\mathbf{z}_{n+1}|\mathbf{z}_n)$$

When we have long length, the value of alpha and beta **will be extremely small**, even beyond the precision of computer.

$$\widehat{\alpha}(\mathbf{z}_n) = p(\mathbf{z}_n|\mathbf{x}_1, \ldots, \mathbf{x}_n) = \frac{\alpha(\mathbf{z}_n)}{p(\mathbf{x}_1, \ldots, \mathbf{x}_n)}$$

$$c_n = p(\mathbf{x}_n|\mathbf{x}_1, \ldots, \mathbf{x}_{n-1}) \qquad p(\mathbf{x}_1, \ldots, \mathbf{x}_n) = \prod_{m=1}^{n} c_m$$

$$\alpha(\mathbf{z}_n) = p(\mathbf{z}_n|\mathbf{x}_1, \ldots, \mathbf{x}_n) p(\mathbf{x}_1, \ldots, \mathbf{x}_n) = \left(\prod_{m=1}^{n} c_m \right) \widehat{\alpha}(\mathbf{z}_n)$$

$$c_n \widehat{\alpha}(\mathbf{z}_n) = p(\mathbf{x}_n|\mathbf{z}_n) \sum_{\mathbf{z}_{n-1}} \widehat{\alpha}(\mathbf{z}_{n-1}) p(\mathbf{z}_n|\mathbf{z}_{n-1})$$

$$\beta(\mathbf{z}_n) = \left(\prod_{m=n+1}^{N} c_m\right) \widehat{\beta}(\mathbf{z}_n)$$

$$\widehat{\beta}(\mathbf{z}_n) = \frac{p(\mathbf{x}_{n+1}, \ldots, \mathbf{x}_N | \mathbf{z}_n)}{p(\mathbf{x}_{n+1}, \ldots, \mathbf{x}_N | \mathbf{x}_1, \ldots, \mathbf{x}_n)}$$

$$c_{n+1}\widehat{\beta}(\mathbf{z}_n) = \sum_{\mathbf{z}_{n+1}} \widehat{\beta}(\mathbf{z}_{n+1}) p(\mathbf{x}_{n+1}|\mathbf{z}_{n+1}) p(\mathbf{z}_{n+1}|\mathbf{z}_n)$$

$$\begin{aligned}
\gamma(\mathbf{z}_n) &= \widehat{\alpha}(\mathbf{z}_n)\widehat{\beta}(\mathbf{z}_n) \\
\xi(\mathbf{z}_{n-1}, \mathbf{z}_n) &= c_n \widehat{\alpha}(\mathbf{z}_{n-1}) p(\mathbf{x}_n|\mathbf{z}_n) p(\mathbf{z}_n|\mathbf{z}_{-1}) \widehat{\beta}(\mathbf{z}_n)
\end{aligned}$$

The Viterbi Algorithm

- Goal: finding the most probable hidden states given an observed sequential data.
- Dynamic programming approach
- Application
    - Speech recognition: finding the most probable phoneme sequence given series of acoustic observations
    - Action recognition: finding the most probable action type given observed video frames
    - …

- The problem:

$$\mathrm{argmax}_{\boldsymbol{z}_{1:N}} p(\boldsymbol{z}_{1:N}|\boldsymbol{x}_{1:N})$$

$$\downarrow$$

$$\mathrm{argmax}_{\boldsymbol{z}_{1:N}} p(\boldsymbol{x}_{1:N}, \boldsymbol{z}_{1:N})$$

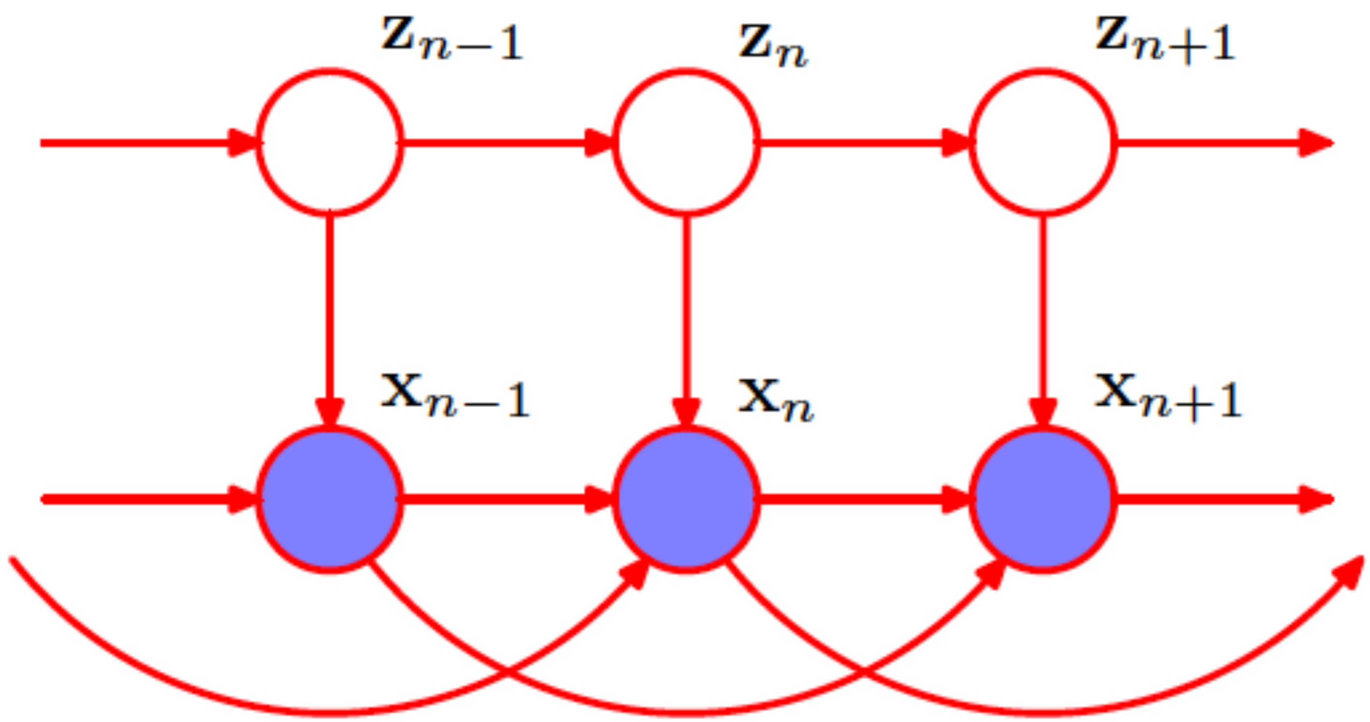
$$\omega(\boldsymbol{z}_n) = \max_{\boldsymbol{z}_{1:n-1}} \ln\, p(\boldsymbol{x}_{1:n}, \boldsymbol{z}_{1:n})$$

$$\omega(\mathbf{z}_1) = \ln p(\mathbf{z}_1) + \ln p(\mathbf{x}_1|\mathbf{z}_1)$$

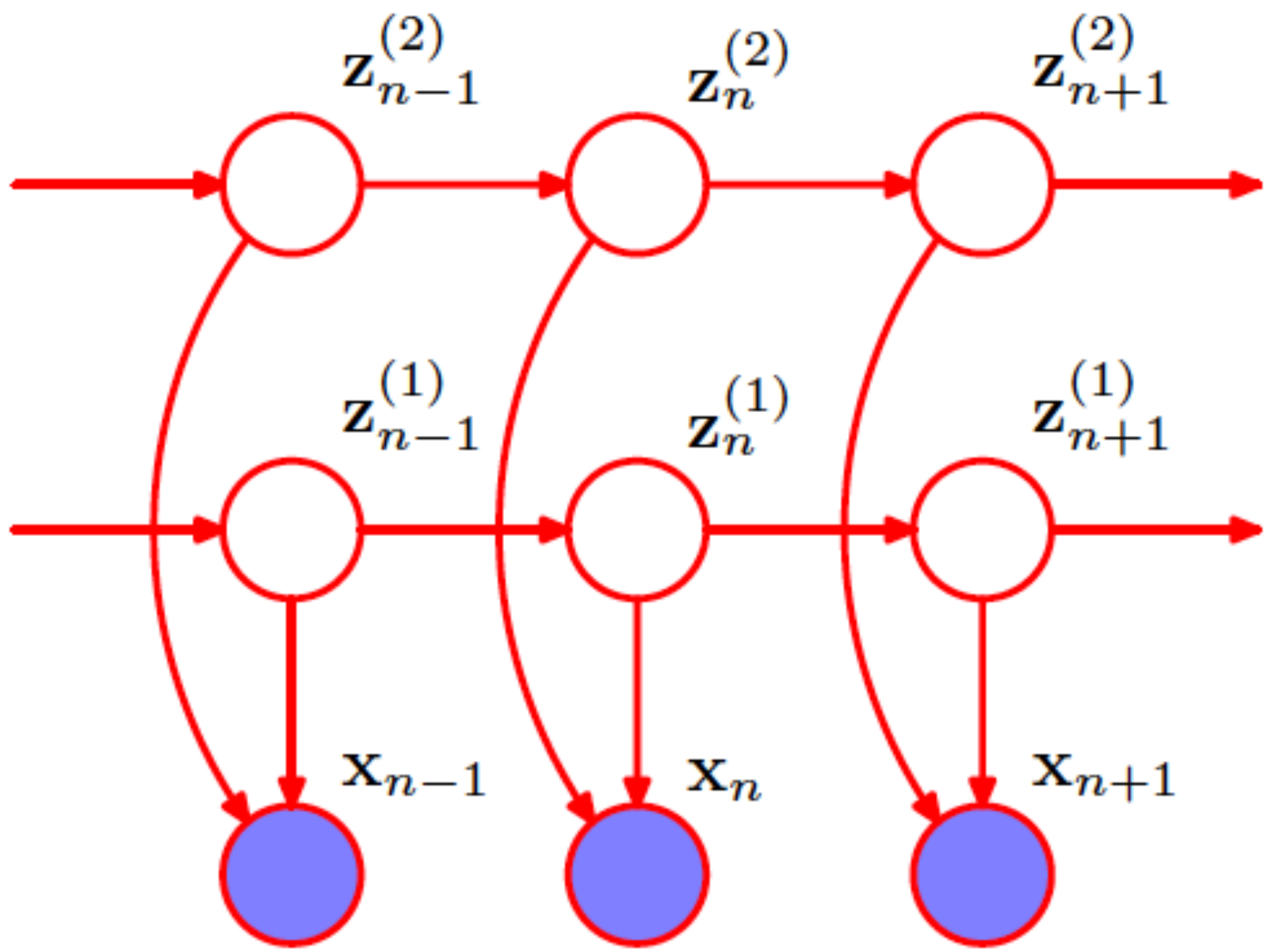
Dynamic programming

$$\omega(\mathbf{z}_{n+1}) = \ln p(\mathbf{x}_{n+1}|\mathbf{z}_{n+1}) + \max_{\mathbf{z}_n}\{\ln p(\mathbf{x}_{+1}|\mathbf{z}_n) + \omega(\mathbf{z}_n)\}$$

Variants of HMM



Autoregressive HMM
for capturing long-range dependency



Factorial HMM

Application**: energy disaggregation**

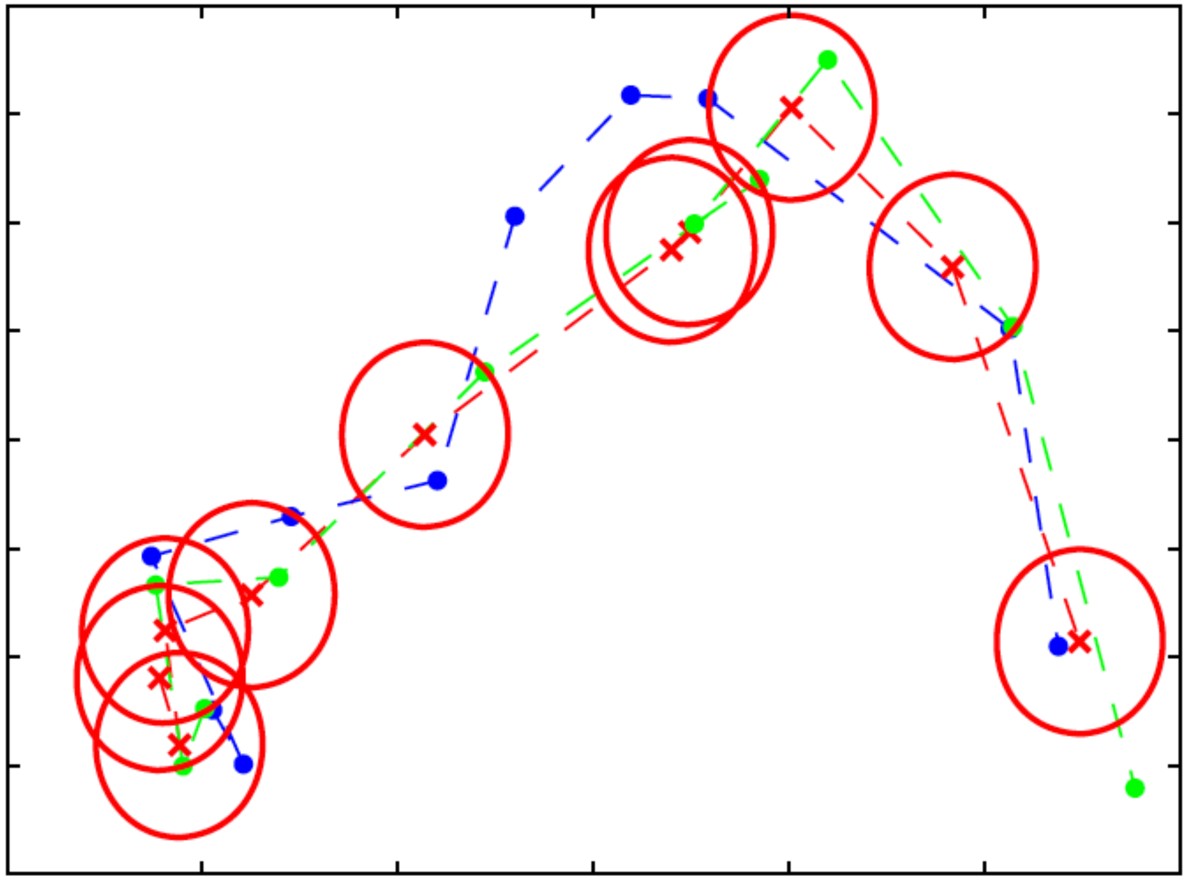
- Linear dynamical systems (LDS)

  - Continuous state variables

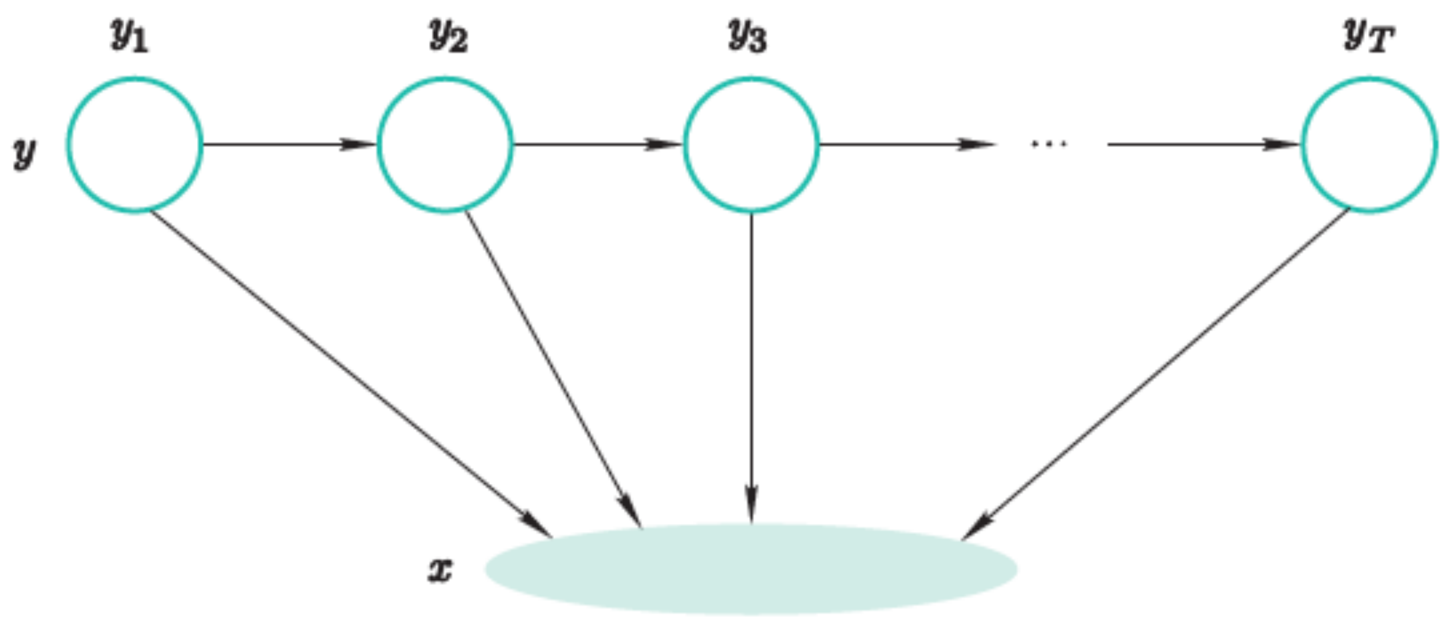
$$
\begin{aligned}
p(\mathbf{z}_n|\mathbf{z}_{n-1}) &= \mathcal{N}(\mathbf{z}_n|\mathbf{A}\mathbf{z}_{n-1}, \boldsymbol{\Gamma}) \\
p(\mathbf{x}_n|\mathbf{z}_n) &= \mathcal{N}(\mathbf{x}_n|\mathbf{C}\mathbf{z}_n, \boldsymbol{\Sigma}).
\end{aligned}
$$

$$
p(\mathbf{z}_1) = \mathcal{N}(\mathbf{z}_1|\boldsymbol{\mu}_0, \mathbf{V}_0)
$$

  - Could be used for object tracking by Kalman filtering



Conditional Random Field (CRF)



Conditional Random Field (Lafferty et.al 2001)

- A discriminative approach for prediction, i.e. modeling conditional distribution directly, not a generative model

tagging

$$p(z_{1:N}|x_{1:N})$$

observation

- More powerful modeling than HMMs on segmenting and labeling sequence data
  - Modeling overlapping and non-independent features, particularly in the task of tagging natural language processing
  - Special case: linear chain CRF = the undirected graphical version of HMM

- Linear chain CRF

$$p(z_{1:N}|x_{1:N}) = \frac{1}{Z}\exp\left(\sum_{n=1}^{N}\sum_{i=1}^{F}\lambda_i f_i(z_{n-1}, z_n, x_{1:N}, n)\right)$$

weight **feature function**

Partition function/
normalization constant:

$$Z = \sum_{z_{1:N}}\exp\left(\sum_{n=1}^{N}\sum_{i=1}^{F}\lambda_i f_i(z_{n-1}, z_n, x_{1:N}, n)\right)$$

- Weights are parameters to be learning from data

- Need to specify the feature functions

Feature Engineering

- Some simple example of feature functions

$$f_1(z_{n-1}, z_n, x_{1:N}, n) = \begin{cases} 1 & \text{if } z_n = \text{PERSON and } x_n = \text{John} \\ 0 & \text{otherwise} \end{cases}$$

lambda1, f1 together are equivalent to the logarithm of emission probability

$$p(x = \text{John} | z = \text{PERSON})$$

$$f_2(z_{n-1}, z_n, x_{1:N}, n) = \begin{cases} 1 & \text{if } z_n = \text{PERSON and } x_{n+1} = \text{said} \\ 0 & \text{otherwise} \end{cases}$$

note f_1 and f_2 can be both active for a sentence like "John said so." and $z_1 = \text{PERSON}$. This is an example of *overlapping features*. It boosts up the belief of $z_1 = \text{PERSON}$ to $\lambda_1 + \lambda_2$. This is something HMMs cannot do: HMMs cannot look at the next word, nor can they use overlapping features.

$$f_3(z_{n-1}, z_n, x_{1:N}, n) = \begin{cases} 1 & \text{if } z_{n-1} = \text{OTHER and } z_n = \text{PERSON} \\ 0 & \text{otherwise} \end{cases}$$

CRF Training

- Training data

$$\{(\mathbf{x}^{(1)}, \mathbf{z}^{(1)}), \ldots, (\mathbf{x}^{(m)}, \mathbf{z}^{(m)})\}, \text{ where } \mathbf{x}^{(1)} = x_{1:N_1}^{(1)}$$

- Maximization problem, or regularized version

$$\sum_{j=1}^{m} \log p(\mathbf{z}^{(j)} | \mathbf{x}^{(j)})$$

$$\sum_{j=1}^{m} \log p(\mathbf{z}^{(j)} | \mathbf{x}^{(j)}) - \sum_{i}^{F} \frac{\lambda_i^2}{2\sigma^2}$$

- Gradient-based learning (L-BFGS)

$$\frac{\partial}{\partial \lambda_k} \sum_{j=1}^{m} \log p(\mathbf{z}^{(j)} | \mathbf{x}^{(j)}) - \sum_{i}^{F} \frac{\lambda_i^2}{2\sigma^2}$$

$$= \frac{\partial}{\partial \lambda_k} \sum_{j=1}^{m} \left(\sum_{n} \sum_{i} \lambda_i f_i(z_{n-1}^{(j)}, z_n^{(j)}, \mathbf{x}^{(j)}, n) - \log Z^{(j)} \right) - \sum_{i}^{F} \frac{\lambda_i^2}{2\sigma^2}$$

$$= \underbrace{\sum_{j=1}^{m} \sum_{n} f_k(z_{n-1}^{(j)}, z_n^{(j)}, \mathbf{x}^{(j)}, n)}_{\text{data term}}$$

$$\underbrace{- \sum_{j=1}^{m} \sum_{n} E_{z'_{n-1}, z'_n} [f_k(z'_{n-1}, z'_n, \mathbf{x}^{(j)}, n)]}_{\text{model term}} - \frac{\lambda_k}{\sigma^2},$$

Matching the two terms if we ignore the regularization

$$
\begin{aligned}
\frac{\partial}{\partial \lambda_k} \log Z &= E_{\mathbf{z}'}[\sum_n f_k(z'_{n-1}, z'_n, \mathbf{x}, n)] \\
&= \sum_n E_{z'_{n-1}, z'_n}[f_k(z'_{n-1}, z'_n, \mathbf{x}, n)] \\
&= \sum_n \sum_{z'_{n-1}, z'_n} p(z'_{n-1}, z'_n | \mathbf{x}) f_k(z'_{n-1}, z'_n, \mathbf{x}, n)
\end{aligned}
$$

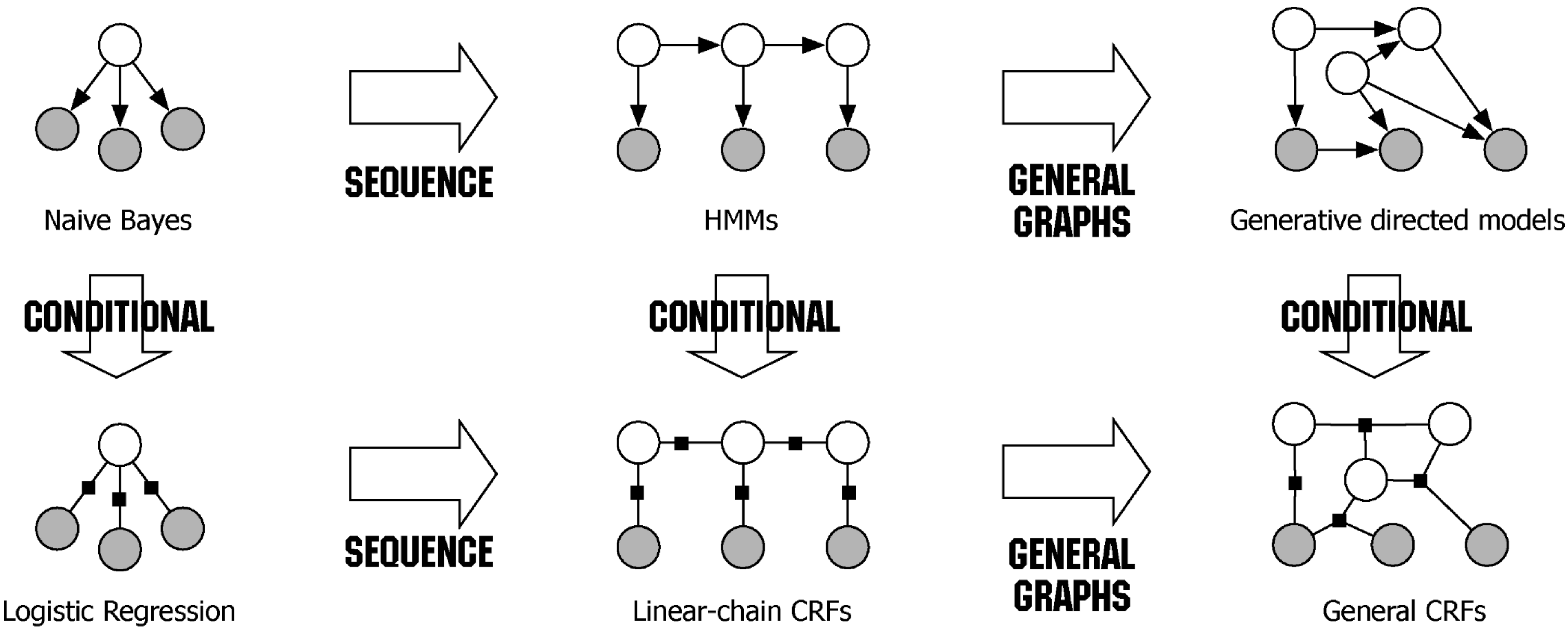
Feature Selection

- Common practice
  - Define a very large number of candidate features and let the data determine the optimal subset
- Two stages for building candidate features
  - Atomic candidate features
    - Simple test on a specific combination of words and tags.

$(x = \text{John},\ z = \text{PERSON})$ $(x = \text{John},\ z = \text{ORGANIZATION})$

  - "Grow" candidate features
    - Combine simple feature to form complex ones

CRF and Directed GM



Charles and McCallum (2012)

Exercise

- Implement the MLE estimation of HMM parameters, forward-backward alg. and Viterbi alg.
- Optional readings
    - https://homepages.inf.ed.ac.uk/csutton/publications/crftut-fnt.pdf
    - https://www.seas.upenn.edu/~strctlrn/bib/PDF/crf.pdf