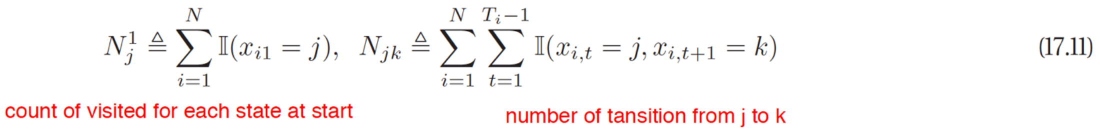
17.5 Learning for HMMs

- 모델 파라미터를 추정하는 것
  - inital state probabilites
  - state transition matrix, A = p(z_t|z_t-1)
  - observation probability, B = p(x_t | z_t)
- 두 가지 case
  - 상태열, z_1:T 를 안다고 가정할 때 <- 극단적인 쉬운 상황
  - 상태열을 모르는 상황 <- 일반적

17.5.1 Training with fully observed data

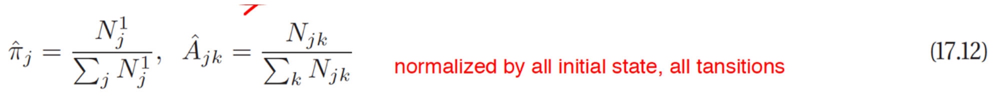
- 상태열을 안다고 가정할 때
- 초기 상태 확률과 천이 확률은 markov chain MLE와 동일하게 구할 수 있음 (17.2.2.1)



$$N_j^1 \triangleq \sum_{i=1}^{N} \mathbb{I}(x_{i1} = j), \quad N_{jk} \triangleq \sum_{i=1}^{N} \sum_{t=1}^{T_i - 1} \mathbb{I}(x_{i,t} = j, x_{i,t+1} = k) \tag{17.11}$$

count of visited for each state at start

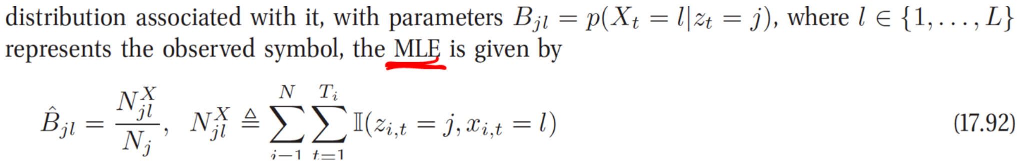
number of tansition from j to k



$$\hat{\pi}_j = \frac{N_j^1}{\sum_j N_j^1}, \quad \hat{A}_{jk} = \frac{N_{jk}}{\sum_k N_{jk}} \tag{17.12}$$

normalized by all initial state, all tansitions

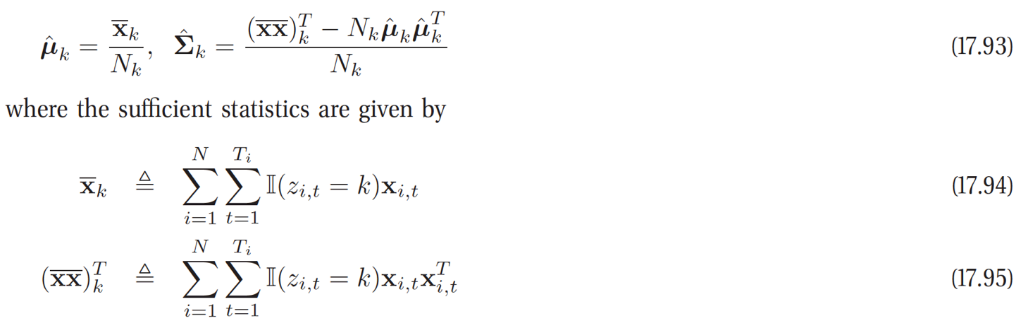
- 관측 확률은 어떤 관측 모델(ex. mulltinulli, gaussian)이냐에 따라 달라진다.
- 관측 모델이 multinoulli일 경우는 아래와 같다.



distribution associated with it, with parameters $B_{jl} = p(X_t = l | z_t = j)$, where $l \in \{1, \ldots, L\}$ represents the observed symbol, the MLE is given by

$$\hat{B}_{jl} = \frac{N_{jl}^X}{N_j}, \quad N_{jl}^X \triangleq \sum_{i=1}^{N} \sum_{t=1}^{T_i} \mathbb{I}(z_{i,t} = j, x_{i,t} = l) \tag{17.92}$$

- 관측 모델이 gaussian 일 경우는 아래와 같다.



$$\hat{\boldsymbol{\mu}}_k = \frac{\overline{\mathbf{x}}_k}{N_k}, \quad \hat{\boldsymbol{\Sigma}}_k = \frac{(\overline{\mathbf{x}\mathbf{x}})_k^T - N_k \hat{\boldsymbol{\mu}}_k \hat{\boldsymbol{\mu}}_k^T}{N_k} \tag{17.93}$$

where the sufficient statistics are given by

$$\overline{\mathbf{x}}_k \triangleq \sum_{i=1}^{N} \sum_{t=1}^{T_i} \mathbb{I}(z_{i,t} = k) \mathbf{x}_{i,t} \tag{17.94}$$

$$(\overline{\mathbf{x}\mathbf{x}})_k^T \triangleq \sum_{i=1}^{N} \sum_{t=1}^{T_i} \mathbb{I}(z_{i,t} = k) \mathbf{x}_{i,t} \mathbf{x}_{i,t}^T \tag{17.95}$$

Expectation-Maximization 리뷰 (11.4)

언제 사용?

- full data가 주어지면 일반적으로 ML/MAP 구하는 건 누워서 떡먹기
- missing data나 latent variable가 있을 때는 매우 어려워짐
- 이럴 때 사용하는 게 EM
  - iterative, close-form update (hopefully)

기본 아이디어

- 아래처럼 일반적으로 data log likelihood를 최대화하는 파라미터를 구하고 싶다.
- 하지만 latent variable이 있고, 이것이 log sum 형태여서는 optimize하기 쉬운 형태가 아님

$$\ell(\boldsymbol{\theta}) = \sum_{i=1}^{N} \log p(\mathbf{x}_i|\boldsymbol{\theta}) = \sum_{i=1}^{N} \log \left[\sum_{\mathbf{z}_i} p(\mathbf{x}_i, \mathbf{z}_i|\boldsymbol{\theta}) \right] \tag{11.17}$$

- 대신 아래처럼 complete data log likelihood를 정의하고 이를 최대화하자
- 그럼에도 latent variable을 관측할 수 없고, 확률 분포가 multimodal일 수도 있어서 analytically하게 최적화 못함

$$\ell_c(\boldsymbol{\theta}) \triangleq \sum_{i=1}^{N} \log p(\mathbf{x}_i, \mathbf{z}_i|\boldsymbol{\theta}) \tag{11.18}$$

- 최초 guess -> 기대값 -> 기대값 최대화하게 guess 갱신 -> 수렴할 때까지 반복
- data log likelihood는 매 iteration마다 단조 증가한다.

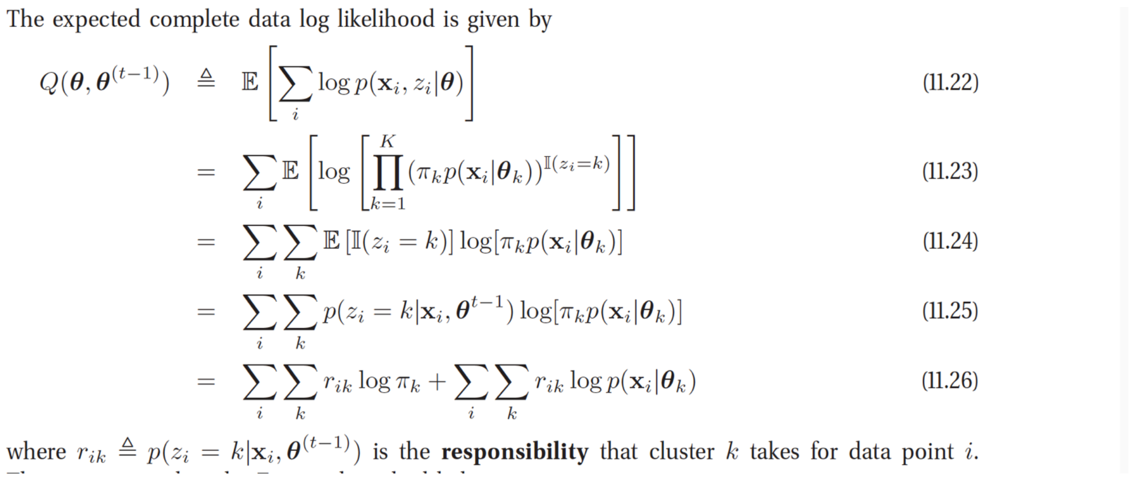
$$Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{t-1}) = \mathbb{E}\left[\ell_c(\boldsymbol{\theta}) \middle| \mathcal{D}, \boldsymbol{\theta}^{t-1}\right] \tag{11.19}$$

$$\boldsymbol{\theta}^t = \arg\max_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{t-1}) \tag{11.20}$$

주의할 점

- gaussian family와 같은 특수한 형태가 되야지 E-step, M-step이 analytically, close-form으로 풀린다.
- 구한 MLE는 local maxima일 수 있다.

예제 - EM for GMM



The expected complete data log likelihood is given by

$$Q(\boldsymbol{\theta},\boldsymbol{\theta}^{(t-1)}) \triangleq \mathbb{E}\left[\sum_i \log p(\mathbf{x}_i, z_i|\boldsymbol{\theta})\right] \quad (11.22)$$

$$= \sum_i \mathbb{E}\left[\log\left[\prod_{k=1}^{K}(\pi_k p(\mathbf{x}_i|\boldsymbol{\theta}_k))^{\mathbb{I}(z_i=k)}\right]\right] \quad (11.23)$$

$$= \sum_i\sum_k \mathbb{E}\left[\mathbb{I}(z_i=k)\right]\log[\pi_k p(\mathbf{x}_i|\boldsymbol{\theta}_k)] \quad (11.24)$$

$$= \sum_i\sum_k p(z_i=k|\mathbf{x}_i,\boldsymbol{\theta}^{t-1})\log[\pi_k p(\mathbf{x}_i|\boldsymbol{\theta}_k)] \quad (11.25)$$

$$= \sum_i\sum_k r_{ik}\log\pi_k + \sum_i\sum_k r_{ik}\log p(\mathbf{x}_i|\boldsymbol{\theta}_k) \quad (11.26)$$

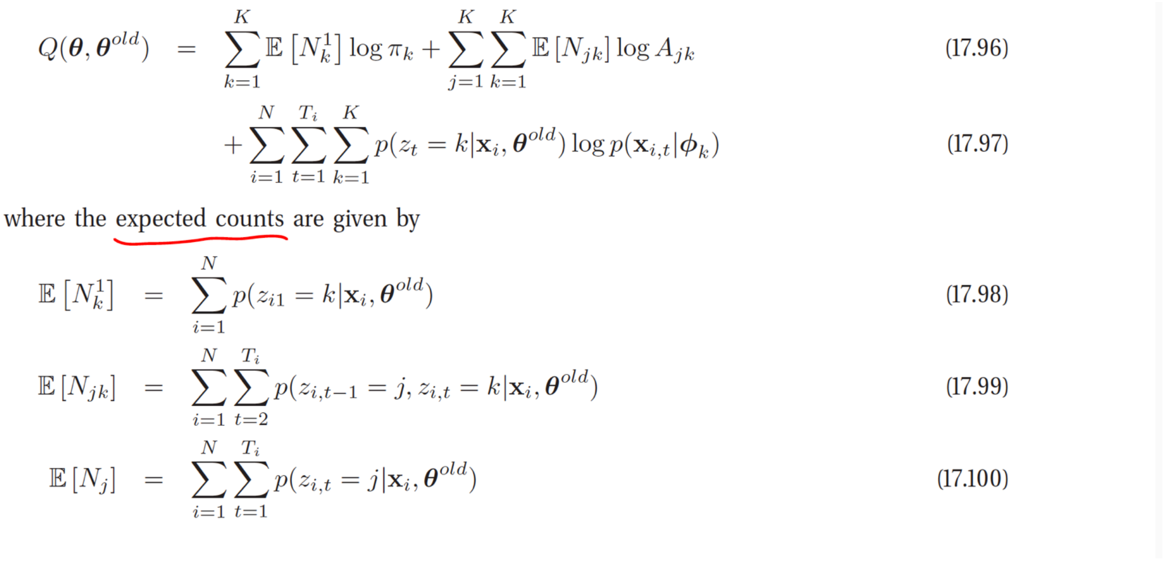
where $r_{ik} \triangleq p(z_i = k|\mathbf{x}_i, \boldsymbol{\theta}^{(t-1)})$ is the **responsibility** that cluster k takes for data point i.

17.5.2 EM for HMMs (Baum-Welch Algorithm)

- 상태열을 모를 때 (당연!)

E-step

- forward-backward algorithm 을 통해 얻을 수 있다.



$$Q(\boldsymbol{\theta},\boldsymbol{\theta}^{old}) = \sum_{k=1}^{K}\mathbb{E}\left[N_k^1\right]\log\pi_k + \sum_{j=1}^{K}\sum_{k=1}^{K}\mathbb{E}\left[N_{jk}\right]\log A_{jk} \quad (17.96)$$

$$+\sum_{i=1}^{N}\sum_{t=1}^{T_i}\sum_{k=1}^{K} p(z_t=k|\mathbf{x}_i,\boldsymbol{\theta}^{old})\log p(\mathbf{x}_{i,t}|\boldsymbol{\phi}_k) \quad (17.97)$$

where the expected counts are given by

$$\mathbb{E}\left[N_k^1\right] = \sum_{i=1}^{N} p(z_{i1}=k|\mathbf{x}_i,\boldsymbol{\theta}^{old}) \quad (17.98)$$

$$\mathbb{E}\left[N_{jk}\right] = \sum_{i=1}^{N}\sum_{t=2}^{T_i} p(z_{i,t-1}=j, z_{i,t}=k|\mathbf{x}_i,\boldsymbol{\theta}^{old}) \quad (17.99)$$

$$\mathbb{E}\left[N_j\right] = \sum_{i=1}^{N}\sum_{t=1}^{T_i} p(z_{i,t}=j|\mathbf{x}_i,\boldsymbol{\theta}^{old}) \quad (17.100)$$

$$\gamma_{i,t}(j) \triangleq p(z_t=j|\mathbf{x}_{i,1:T_i},\boldsymbol{\theta}) \quad (17.101)$$

$$\xi_{i,t}(j,k) \triangleq p(z_{t-1}=j, z_t=k|\mathbf{x}_{i,1:T_i},\boldsymbol{\theta}) \quad (17.102)$$

M-step

- 최초 상태 확률과 천이 확률은 아래처럼 단순히 normalization이다.
  - 아래 링크처럼 auxiliary function이 최적화 목적함수이고, 라그랑제 multiplier를 이용하면 유도된다.

  - [https://people.eecs.berkeley.edu/~stephentu/writeups/hmm-baum-welch-derivation.pdf (https://people.eecs.berkeley.edu/~stephentu/writeups/hmm-baum-welch-derivation.pdf)](https://people.eecs.berkeley.edu/~stephentu/writeups/hmm-baum-welch-derivation.pdf)

$$\underline{\hat{A}_{jk}} = \frac{\mathbb{E}[N_{jk}]}{\sum_{k'} \mathbb{E}[N_{jk'}]}, \quad \underline{\hat{\pi}_k} = \frac{\mathbb{E}[N_k^1]}{N} \tag{17.103}$$

- 관측 확률은 multinoulli model인 경우
  - 상태 j에 머물르면서 관측 l을 할 기대 count를 normaliza 한 것

$$\mathbb{E}[M_{jl}] = \sum_{i=1}^{N} \sum_{t=1}^{T_i} \gamma_{i,t}(j) \mathbb{I}(x_{i,t} = l) = \sum_{i=1}^{N} \sum_{t:x_{i,t}=l} \gamma_{i,t}(j) \tag{17.104}$$

$$\underline{\hat{B}_{jl}} = \frac{\mathbb{E}[M_{jl}]}{\mathbb{E}[N_j]} \tag{17.105}$$

17.5.2.3 Initailization of parameters

초기화가 어설프면 poor local minima에서 헤어나오지 못한다.

실용적인 방법들

- 조금이라도 fully labeled data가 있으면 이걸 가지고 naive하게나마 초기 파라미터 값을 정한다. (17.5.1)
- markov 속성을 무시하고, 그냥 k-means 와 같은 방법으로 추정
- 그냥 무식하게 random multiple restart한다.

deterministic annealing 으로 local minima 완화

- [http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.522.8071&rep=rep1&type=pdf (http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.522.8071&rep=rep1&type=pdf)](http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.522.8071&rep=rep1&type=pdf)

17.5.3 Baysian methods for fitting HMMs * - SKIP

17.5.4 Discriminative traning - SKIP

17.5.5 Model selection

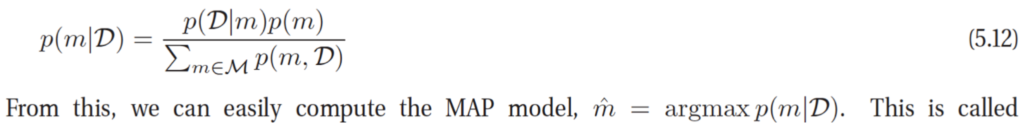
hidden state 몇개 쓸까? state transition topology를 어떻게 할까?

17.5.5.1 Choosing the number of hidden states

CV-based

- generalization error 를 지표로 비교
- cross-validated likelihood 을 지표로 비교
  - 일종의 test log likelihood
  - 각 모델(K=3,4,5..)별 training set으로 학습 후 test set에 대한 test data log likelihood, P(D|K)
  - data가 적을 때는 CV 형식으로 해서, partition 개수만큼 반복후 평균한 값
  - https://link.springer.com/content/pdf/10.1023/A:1008940618127.pdf (https://link.springer.com/content/pdf/10.1023/A:1008940618127.pdf)

Baysian model selection (5.3)

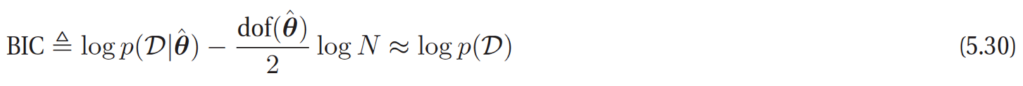


$$p(m|\mathcal{D}) = \frac{p(\mathcal{D}|m)p(m)}{\sum_{m \in \mathcal{M}} p(m, \mathcal{D})} \tag{5.12}$$

From this, we can easily compute the MAP model, $\hat{m} = \operatorname{argmax} p(m|\mathcal{D})$. This is called

BIC, AIC (5.3.2.4)

- penalty on model complexity



$$\text{BIC} \triangleq \log p(\mathcal{D}|\hat{\boldsymbol{\theta}}) - \frac{\text{dof}(\hat{\boldsymbol{\theta}})}{2} \log N \approx \log p(\mathcal{D}) \tag{5.30}$$

$$\text{AIC}(m, \mathcal{D}) \triangleq \log p(\mathcal{D}|\hat{\boldsymbol{\theta}}_{MLE}) - \text{dof}(m) \tag{5.35}$$

samplimg based

- reversible jump MCMC
  - https://www.seas.harvard.edu/courses/cs281/papers/hastie-green-2011.pdf (https://www.seas.harvard.edu/courses/cs281/papers/hastie-green-2011.pdf)
- variational bayes
  - https://www.cse.buffalo.edu/faculty/mbeal/papers/beal03.pdf (https://www.cse.buffalo.edu/faculty/mbeal/papers/beal03.pdf)
- Infinite HMM
  - http://mlg.eng.cam.ac.uk/zoubin/papers/ihmm.pdf (http://mlg.eng.cam.ac.uk/zoubin/papers/ihmm.pdf)

17.5.5.2 Structure learning - SKIP

17.6 Generalization of HMMs

Hidden Semi Markov Model (HSMM)

semi-markov property

- next state는 현재 state와 현재 state에 얼마나 머물렀는지에 따른다.
- P(z_t+1 | z_t, duration)

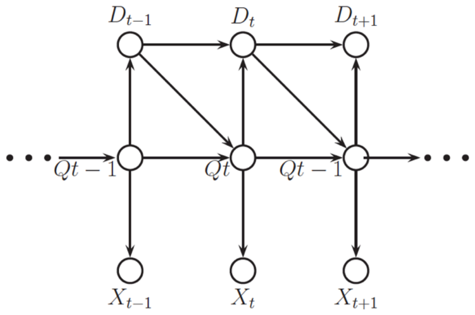
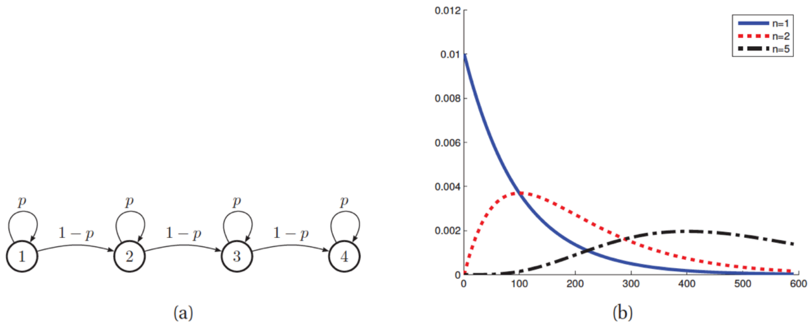


Figure 17.14 Encoding a hidden semi-Markov model as a DGM. D_t are deterministic duration counters.

as augmented HMM

- repeated states
- with state duration counter



(a)

(b)

Figure 17.15 (a) A Markov chain with $n = 4$ repeated states and self loops. (b) The resulting distribution over sequence lengths, for $p = 0.99$ and various n. Figure generated by `hmmSelfLoopDist`.

Hidden Semi Markov Model (HSMM)

계층 구조 반영

- ex) word - phone - subphone

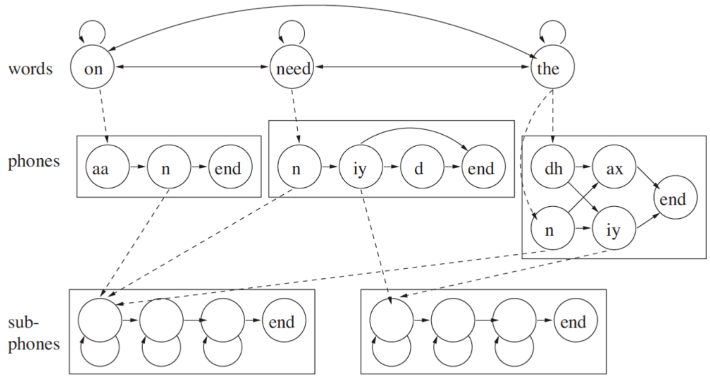


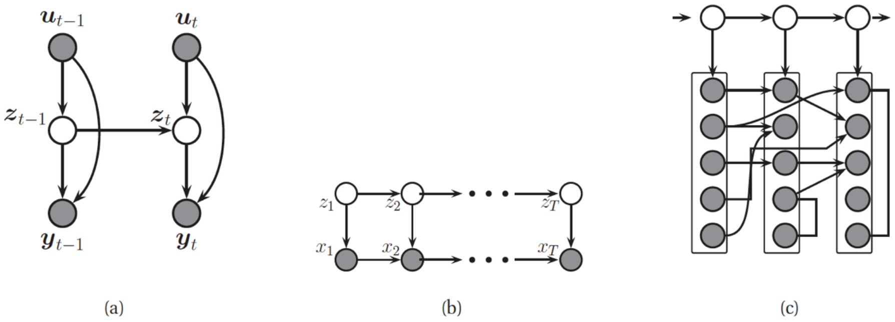
Figure 17.16 An example of an HHMM for an ASR system which can recognize 3 words. The top level represents bigram word probabilities. The middle level represents the phonetic spelling of each word. The bottom level represents the subphones of each phone. (It is traditional to represent a phone as a 3 state HMM, representing the beginning, middle and end.) Based on Figure 7.5 of (Jurafsky and Martin 2000).

Input-Output HMM (IOHMM)

- 다음처럼 input(control) signal이 확률 변수로 은닉 상태와 observation에 영향을 미침

$$p(\mathbf{y}_{1:T}, \mathbf{z}_{1:T}|\mathbf{u}_{1:T}, \boldsymbol{\theta}) \tag{17.115}$$

$$\begin{aligned} p(z_t|\mathbf{x}_t, z_{t-1} = i, \boldsymbol{\theta}) &= \mathrm{Cat}(z_t|\mathcal{S}(\mathbf{W}_i\mathbf{u}_t)) && (17.116) \\ p(\mathbf{y}_t|\mathbf{x}_t, z_t = j, \boldsymbol{\theta}) &= \mathcal{N}(\mathbf{y}_t|\mathbf{V}_j\mathbf{u}_t, \boldsymbol{\Sigma}_j) && (17.117) \end{aligned}$$



Auto-regressive HMM (AR-HMM)

- regular HMM 에서는 은닉상태을 알때는 observation끼리 조건부 독립
- 이러한 가정을 완화시켜서 observation끼리 영향을 미치게..
- 이를 1step or L-step linear regression 으로 모델링

$$p(\mathbf{x}_t|\mathbf{x}_{t-1}, z_t = j, \boldsymbol{\theta}) = \mathcal{N}(\mathbf{x}_t|\mathbf{W}_j\mathbf{x}_{t-1} + \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) \tag{17.118}$$

$$p(\mathbf{x}_t|\mathbf{x}_{t-L:t-1}, z_t = j, \boldsymbol{\theta}) = \mathcal{N}(\mathbf{x}_t|\sum_{\ell=1}^{L}\mathbf{W}_{j,\ell}\mathbf{x}_{t-\ell} + \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) \tag{17.119}$$

Buried HMM

- more complex dependencies between observation nodes
- dynamic basysian multi-net

Factorial HMM

- distributed representation of hidden state
- 여러 다른 context를 함께 표현
    - ex) speech words and speaking style

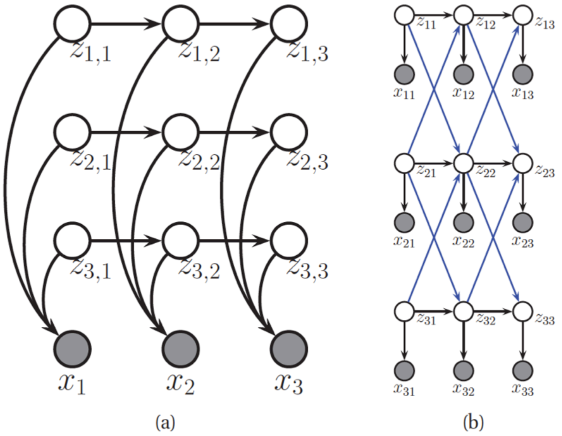
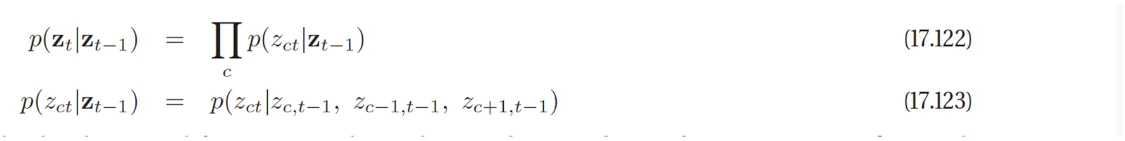


Figure 17.19 (a) A factorial HMM with 3 chains. (b) A coupled HMM with 3 chains.

Coupled HMM

- multiple related data streams
- state transition depends on neighboring chains
- ex) audio-visual speech recognition

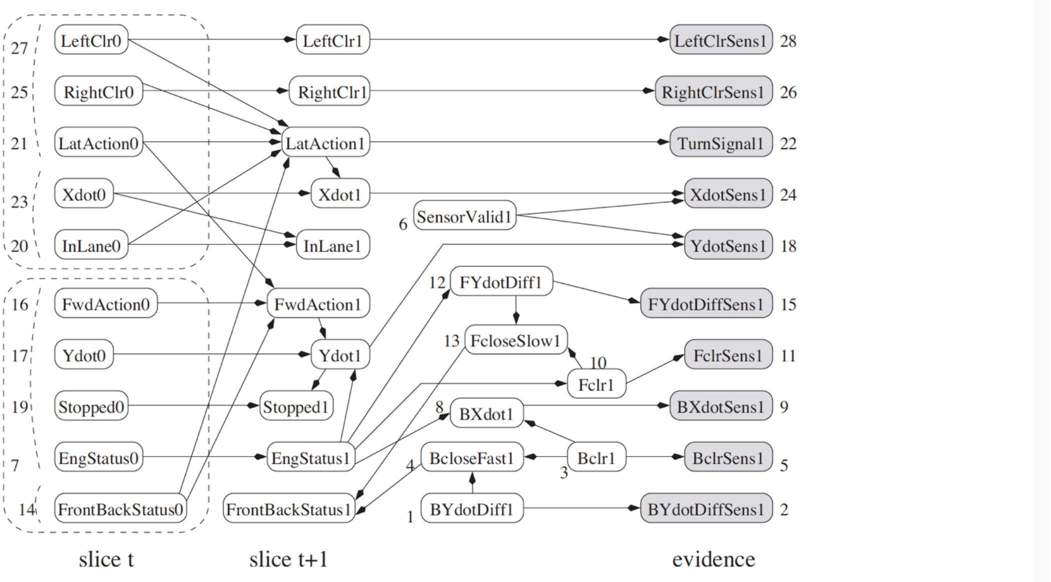


$$p(\mathbf{z}_t|\mathbf{z}_{t-1}) = \prod_c p(z_{ct}|\mathbf{z}_{t-1}) \quad (17.122)$$

$$p(z_{ct}|\mathbf{z}_{t-1}) = p(z_{ct}|z_{c,t-1}, z_{c-1,t-1}, z_{c+1,t-1}) \quad (17.123)$$

Dynamic Basian Network

- 모든 HMM variants들은 DBN의 일종이다.
- Domain 지식을 바탕으로 문제에 특화된 Graphical modeling을 하는 것
  - ex) 자율주행차 모델링



```
In [ ]:
```