Planning and Optimal Control

2. Hidden Markov Models

Lars Schmidt-Thieme

Information Systems and Machine Learning Lab (ISMLL)
Institute for Computer Science
University of Hildesheim, Germany

Syllabus

Tue. 24.10.	(1)	1. Markov Models
Tue. 31.10.	—	— *Luther Day* —
Tue. 7.11	(2)	2. Hidden Markov Models
Tue. 14.11.	(3)	2b. (ctd.)
Tue. 21.11.	(4)	3. State Space Models
Tue. 28.11.	(5)	4. Markov Random Fields
Tue. 5.12.	(6)	5. Markov Decision Processes
Tue. 12.12.	(7)	6. Partially Observable Markov Decision Processes
Tue. 19.12.	(8)	
Tue. 26.12.	—	— *Christmas Break* —
Tue. 9.1.	(9)	7. Reinforcement Learning
Tue. 16.1.	(10)	
Tue. 23.1.	(11)	
Tue. 30.1.	(12)	
Tue. 6.2.	(13)	

Outline

1. Hidden Markov Models (HMMs)

2. Inference in HMMs

3. Inference in HMMs II: MAP

4. Learning HMMs

Outline

HMMs

Markov models cannot easily represent long-range dependencies:

- state of a single observation is not rich enough to represent full prior sequence
- state sequence of h last observations are rich enough (for h sufficiently large),
  but yield a huge state space (exponentially in h)

Idea:

- do not use observed states to represent the state of an instance, but introduce artificial latent states z
- latent states represents full state of an instance:
  - Markov model $p(z_{t+1} \mid z_t)$ of latent states
  - **observation model** $p(x_t \mid z_t)$
    - observed states depend on current latent state only:

HMMs

- observation model:
  - for discrete observations:

$$B := (p(x_t = i \mid z_t = h))_{h=1:H, i=1:I} \quad H \times I \text{ \textbf{observation matrix}}$$

  - for continuous observations: Gaussian observations model

$$p(x_t \mid z_t = h) = \mathcal{N}(x_t; \mu_h, \sigma_h^2)$$

- the number H of hidden states parametrizes model complexity
- joint distribution:

$$p(x, z) = p(z)p(x \mid z) = p(z_1) \prod_{t=2}^{T} z_t \prod_{t=1}^{T} p(x_t \mid z_t)$$

HMM Applications

- Automatic speech recognition
  - x_t: (features extracted from) speech signal
  - z_t: word/phoneme being spoken
  - observation model $p(x_t \mid z_t)$: acoustic model
  - transition model $p(z_{t+1} \mid z_t)$: language model
- Activity recognition
  - x_t: (features extracted from) video frame
  - z_t: activity person is involved in (running, walking etc.)
- Part of speech tagging:
  - x_t: word in a sentence
  - z_t: part-of-speech of the word (noun, verb, adjective, ...)
- Gene finding:
  - x_t: DNA nucleotide (A,C,G,T)

Outline

Types of Inferences for Temporal Models

- **Filtering**: $p(z_t \mid x_{1:t})$
  - can be done online
  - less noisy state estimation than $p(z_t \mid x_t)$
- **Smoothing**: $p(z_t \mid x_{1:T})$
  - offline, requires access to whole sequence
  - allows to explain sequence in hindsight
- **Fixed Lag Smoothing**: $p(z_{t-\ell} \mid x_{1:t})$, $\ell > 0$ **lag**
  - compromise between filtering ($\ell = 0$) and smoothing ($\ell = \infty$)
  - online with delay ℓ

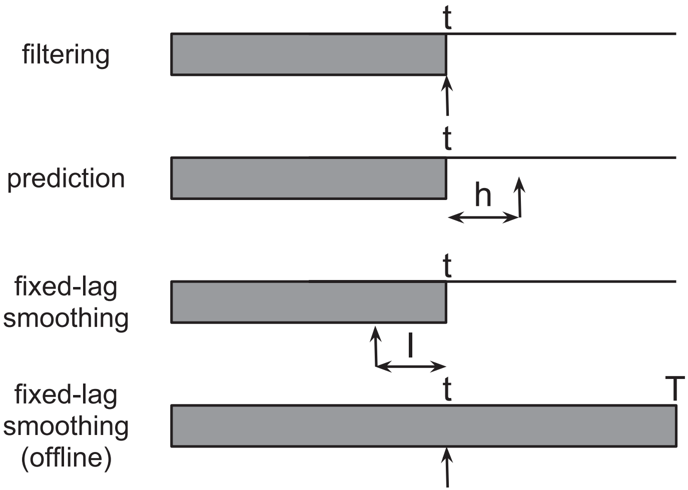
Types of Inferences for Temporal Models

- **Forecasting**: $p(x_{t+h} \mid x_{1:t})$, $h > 0$ **horizon**

$$\begin{aligned} p(x_{t+h} \mid x_{1:t}) &= \sum_{z_{t+h}} p(x_{t+h} \mid z_{t+h}) p(z_{t+h} \mid x_{1:t}) \\ &= \sum_{z_{t+h}} p(x_{t+h} \mid z_{t+h}) \sum_{z_{1:t+h-1}} \prod_{s=t}^{t+h-1} p(z_{s+1} \mid z_s) p(z_t \mid x_{1:t}) \\ &= B^T (A^T)^h p(z_t \mid x_{1:t}) \end{aligned}$$

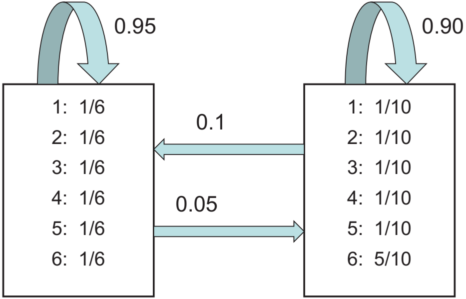
- **MAP estimation**: $\arg\max_{z_{1:t}} p(z_{1:T} \mid x_{1:T})$
  - most probable state sequence to generate observation sequence
  - **Viterbi decoding**
- **Posterior samples**: $z_{1:T} \sim p(z_{1:T} \mid x_{1:T})$
  - richer information than smoothing
- **Probability of the evidence**: $p(x_{1:T}) = \sum_{z_{1:T}} p(z_{1:T}, x_{1:T})$
  - useful as density estimator

Types of Inferences for Temporal Models



[source: Murphy 2012, p.608]

Example: Occasionally Dishonest Casino HMM



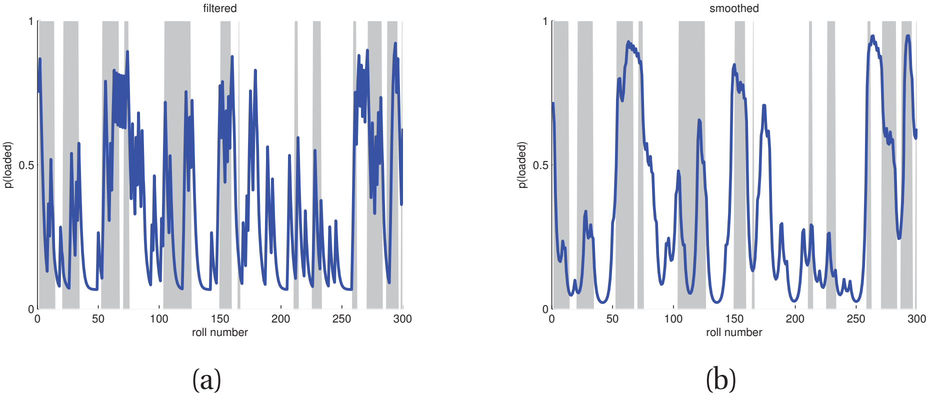
- occasionally dishonest casino: [source: Murphy 2012, p.607]
  - $x_t \in \{1, 2, 3, 4, 5, 6\}$ dice
  - $z_t \in \{1, 2\}$ dice being used
  - $p(x_t \mid z_t = 1) = (\frac{1}{6}, \frac{1}{6}, \frac{1}{6}, \frac{1}{6}, \frac{1}{6}, \frac{1}{6})$ fair dice,
    $p(x_t \mid z_t = 2) = (\frac{1}{10}, \frac{1}{10}, \frac{1}{10}, \frac{1}{10}, \frac{1}{10}, \frac{5}{10})$ loaded dice

a) Filtering

$p(z_t \mid x_{1:t})$

b) Smoothing

$p(z_t \mid x_{1:T})$

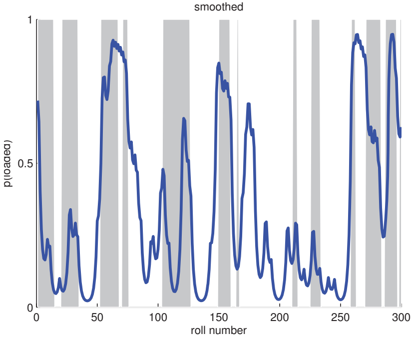


[source: Murphy 2012, p.607]

gray: ground truth $\mathbb{I}(z_t = 2)$, i.e., loaded

b) Smoothing

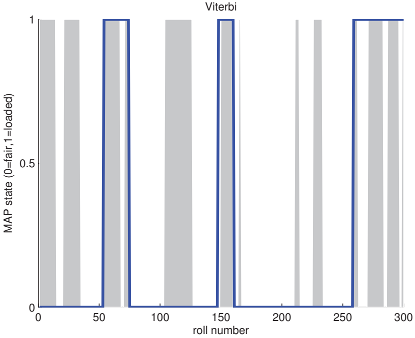
$$p(z_t \mid x_{1:T})$$



(b)

b) MAP

$$\arg\max_{z_{1:T}} p(z_{1:T} \mid x_{1:T})$$



(c)

[source: Murphy 2012, p.607]

Filtering

The filtered latent state

$$\alpha_t := p(z_t \mid x_{1:t})$$

can be computed recursively:

$$\alpha_1 = p(z_1 \mid x_1) = \text{normalize}(B_{.,x_1} \odot \pi)$$
$$\alpha_t = p(z_t \mid x_{1:t}) = \text{normalize}(B_{.,x_t} \odot A^T \alpha_{t-1})$$

proof:

$$p(z_1 \mid x_1) = \frac{p(z_1, x_1)}{\sum_{z_1'} p(z_1', x_1)} = \text{normalize}(p(z_1, x_1))$$
$$= \text{normalize}(p(x_1 \mid z_1) p(z_1)) = \text{normalize}(B_{.,x_1} \odot \pi)$$

Note: $x \odot y := (x_n y_n)_{n=1:N}$ elementwise product of $x, y \in \mathbb{R}^N$,
normalize$(x) = x / \sum_{n=1}^{N} x_n$ normalization to sum 1 of $x \in \mathbb{R}^N$.

Filtering

proof (ctd.):

$$
\begin{aligned}
p(z_t \mid x_{1:t}) &= \text{normalize}(p(z_t, x_{1:t})) \\
&= \text{normalize}(\sum_{z_{t-1}} p(x_t \mid z_t) p(z_t \mid z_{t-1}) p(z_{t-1} \mid x_{1:t-1})) \\
&= \text{normalize}(\sum_{z_{t-1}} p(x_t \mid z_t) A^T \alpha_{t-1}) \\
&= \text{normalize}(\sum_{z_{t-1}} B_{.,x_t} \odot A^T \alpha_{t-1})
\end{aligned}
$$

Filtering / Forwards Algorithm

infer-filtering-forwards(x, A, B, π):
  $T := |x|$
  $\alpha_1 := \text{normalize}(B_{.,x_1} \odot \pi)$
  for $t = 2, \ldots, T$:
    $\alpha_t := \text{normalize}(B_{.,x_t} \odot A^T \alpha_{t-1})$
  return $\alpha_{1:T}$

where

- $x \in \{1, 2, \ldots, L\}^*$ observed sequence
- $A \in [0, 1]^{H \times H}$ latent state transition matrix
- $B \in [0, 1]^{H \times L}$ observation matrix
- $\pi \in [0, 1]^H$ latent state start vector

yields $\alpha_{1:T} = (p(z_t \mid x_{1:t}))_{t=1:T}$ filtered latent state

Smoothing

The smoothed latent state

$$\gamma_t := p(z_t \mid x_{1:T})$$

can be computed as

$$\gamma_t = \text{normalize}(\alpha_t \odot \beta_t)$$

from

$$\alpha_t := p(z_t \mid x_{1:t})$$
$$\beta_t := p(x_{t+1:T} \mid z_t)$$

proof:

$$p(z_t \mid x_{1:T}) \propto p(z_t, x_{t+1:T} \mid x_{1:t}) = p(z_t \mid x_{1:t})p(x_{t+1:T} \mid z_t, \cancel{x_{1:t}}) = \alpha_t \cdot \beta_t$$

Smoothing / Computing β

$\beta_{1:T} := p(x_{t+1:T} \mid z_t)$ can be computed recursively as

$$\beta_T = (1, 1, \ldots, 1)$$
$$\beta_t = A(B_{.,x_{t+1}} \odot \beta_{t+1})$$

proof:

$$\begin{aligned}
\beta_t &= p(x_{t+1:T} \mid z_t) \\
&= \sum_{z_{t+1}} p(x_{t+1:T} \mid z_{t+1}) p(z_{t+1} \mid z_t) \\
&= \sum_{z_{t+1}} p(x_{t+2:T} \mid z_{t+1}) p(x_{t+1} \mid z_{t+1}) p(z_{t+1} \mid z_t) \\
&= A(B_{.,x_{t+1}} \odot \beta_{t+1}) \\
\beta_{T-1} &= p(x_T \mid z_{T-1}) = \sum_{z_T} p(x_T \mid z_T) p(z_T \mid z_{T-1}) \\
&= AB_{.,x_T} = A(B_{.,x_T} \odot \beta_T) \quad \text{for } \beta_T := (1, 1, \ldots, 1)
\end{aligned}$$

Smoothing / Forwards-Backwards Algorithm

backwards(x, A, B):
  $T := |x|$
  $\beta_T := (1, 1, \ldots, 1)$
  for $t = T-1, \ldots, 1$ backwards:
    $\beta_t := A(B_{\cdot, x_{t+1}} \odot \beta_{t+1})$
  return $\beta_{1:T}$

infer-smoothing-forwards-backwards(x, A, B, π):
  $\alpha :=$ infer-filtering-forwards(x, A, B, π)
  $\beta :=$ backwards(x, A, B)
  $\gamma := \alpha \odot \beta$
  return γ

where

- x, A, B, π as for forwards algorithm

yields $\gamma_{1:T} = (p(z_t \mid x_{1:T}))_{t=1:T}$ smoothed latent state

Outline

1. Hidden Markov Models (HMMs)
2. Inference in HMMs
3. Inference in HMMs II: MAP
4. Learning HMMs

MAP vs MPM

- **Maximum Aposteriori estimation (MAP)**:

$$\arg\max_{z_{1:t}} p(z_{1:T} \mid x_{1:T})$$

  - (jointly) most probable state sequence to generate observation sequence

- **Maximum Posterior Marginals (MPM)**:

$$\arg\max_{z_{1:t}} \prod_{t=1}^{T} p(z_t \mid x_{1:T}) = (\arg\max_{z_t} p(z_t \mid x_{1:T}))_{t \in 1:T}$$

  - sequence of most probable states at each time

- Example:

	$X_1 = 0$	$X_1 = 1$	
$X_2 = 0$	0.04	0.3	0.34
$X_2 = 1$	0.36	0.3	0.66
	0.4	0.6	

MAP vs MPM

- **Maximum Aposteriori estimation (MAP)**:

$$\arg\max_{z_{1:t}} p(z_{1:T} \mid x_{1:T})$$

  - (jointly) most probable state sequence to generate observation sequence

- **Maximum Posterior Marginals (MPM)**:

$$\arg\max_{z_{1:t}} \prod_{t=1}^{T} p(z_t \mid x_{1:T}) = (\arg\max_{z_t} p(z_t \mid x_{1:T}))_{t \in 1:T}$$

  - sequence of most probable states at each time

- Example:

	$X_1 = 0$	$X_1 = 1$	
$X_2 = 0$	0.04	0.3	0.34
$X_2 = 1$	0.36	0.3	0.66
	0.4	0.6	

MAP $= (0, 1)$,
MPM $= (1, 1)$

MAP

The probabilities for aposteriori latent states

$$\delta_t(z_t) \propto \max_{z_{1:t-1}} p(z_{1:t} \mid x_{1:t})$$

can be computed recursively:

$$\delta_1 = p(z_1 \mid x_1) = B_{.,x_1} \odot \pi$$
$$\delta_t = \max_{z_{1:t-1}} p(z_{1:t} \mid x_{1:t}) = B_{.,x_t} \odot \text{rowmax}(A^T \text{diag}(\delta_{t-1}))$$

proof:

$$p(z_1 \mid x_1) \propto p(x_1 \mid z_1) p(z_1) = B_{.,x_1} \odot \pi$$

Note: $\text{rowmax}(A) := (\max_{m=1:M} A_{n,m})_{n=1:N}$ **rowwise maxima** of a matrix $A \in \mathbb{R}^{N\times M}$.

MAP

proof (ctd.):

$$
\begin{aligned}
&\max_{z_{1:t-1}} p(z_{1:t} \mid x_{1:t}) \\
&\propto \max_{z_{1:t-1}} p(x_t \mid z_t, \cancel{x_{1:t-1}}, \cancel{z_{1:t-1}})\, p(z_t \mid z_{t-1}, \cancel{x_{1:t-1}}, \cancel{z_{1:t-2}})\, p(z_{1:t-1} \mid x_{1:t-1}) \\
&= \max_{z_{t-1}} p(x_t \mid z_t) p(z_t \mid z_{t-1}) \max_{z_{1:t-2}} p(z_{1:t-1} \mid x_{1:t-1}) \\
&= B_{.,x_t} \odot (\max_{z_{t-1}} A_{z_{t-1},z_t} (\delta_{t-1})_{z_{t-1}})_{z_t} \\
&= B_{.,x_t} \odot \operatorname{rowmax}(A^T \operatorname{diag}(\delta_{t-1}))
\end{aligned}
$$

MAP / Traceback

The MAP latent states

$$z_{1:T} := \arg\max_{z_{1:T}} p(z_{1:T} \mid x_{1:T})$$

can be computed recursively:

$$z_T = \arg\max_{z_T} \, (\delta_T)_{z_T}$$
$$z_{t-1} = \arg\max_{z_{t-1}} \, (A^T_{.,z_t} \odot \delta_{t-1})_{z_{t-1}}$$

MAP / Viterbi Algorithm

```
infer-MAP-viterbi(x, A, B, π):
    T := |x|
    δ_1 := B_{.,x_1} ⊙ π
    for t = 2, ..., T:
        δ_t := B_{.,x_t} ⊙ rowmax(A^T diag(δ_{t-1}))

    z_T := arg max_{z_T} (δ_T)_{z_T}
    for t = T, ..., 2:
        z_{t-1} := arg max_{z_{t-1}} (A^T_{.,z_t} ⊙ δ_{t-1})_{z_{t-1}}
    return z_{1:T}
```

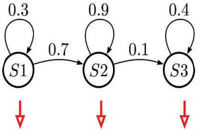
where

- $x \in \{1, 2, \ldots, L\}^*$ observed sequence
- $A \in [0,1]^{H \times H}$ latent state transition matrix
- $B \in [0,1]^{H \times L}$ observation matrix
- $\pi \in [0,1]^{H}$ latent state start vector

yields $z_{1:T} = \arg\max_{z_{1:T}} p(z_{1:T} \mid x_{1:T})$ MAP latent state

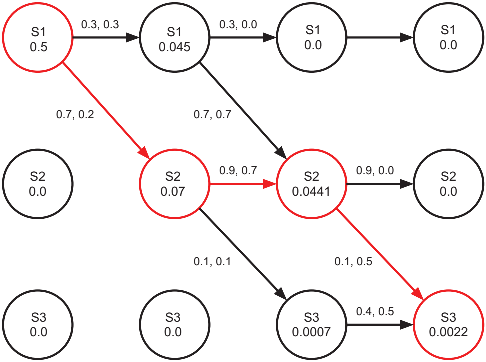
MAP / Example

$$x = (C1, C3, C4, C6)$$



C1	0.5	0	0
C2	0.3	0	0
C3	0.2	0.2	0
C4	0	0.7	0.1
C5	0	0.1	0
C6	0	0	0.5
C7	0	0	0.4

(a)



(b)

Note: Correct typo: $B_{S_1,C_2} = 0.2, B_{S_1,C_3} = 0.3$.

[source: Murphy 2012, p.615]

Posterior Samples

- MAP describes only the most likely posterior hidden state sequence.
- Often one is interested in more fine-grained information, also about other likely hidden state sequences.
- The Viterbi algorithm can be extended to deliver the top-K most likely hidden state sequences.
  - but they often turn out to be very similar to each other.
- better way: draw samples from the posterior:

$$z_{1:T} \sim \; p(z_{1:T} \mid x_{1:T})$$

Posterior Samples

$$z_{1:T} \sim p(z_{1:T} \mid x_{1:T})$$

- forwards inference – backwards sampling:

$$\begin{aligned}
z_T &\sim p(z_T \mid x_{1:T}) = \alpha_T \\
z_{t-1} \mid z_{t:T} &\sim p(z_{t-1} \mid z_{t:T}, x_{1:T}) \\
&\propto p(z_{t-1} \mid z_t, \cancel{z_{t+1:T}}, x_{1:t-1}, \cancel{x_{t:T}}) \\
&\propto p(z_t \mid z_{t-1}, \cancel{x_{1:t-1}})\, p(z_{t-1} \mid x_{1:t-1}) \\
&= A_{.,z_t} \odot \alpha_{t-1}
\end{aligned}$$

Posterior Samples / Forward-Inference—Backwards-Sample

sample-posterior(x, A, B, π, S):
  $T := |x|$
  $\alpha :=$ infer-filtering-forwards(x, A, B, π)
  $\mathcal{S} := \emptyset$
  for $s := 1 : S$:
    $z_T \sim \alpha_T$
    for $t := T : 2$:
      $z_{t-1} \sim \text{normalize}(A_{.,z_t} \odot \alpha_{t-1})$
    $\mathcal{S} := \mathcal{S} \cup \{z_{1:T}\}$
  return $\mathcal{S}$

where

- x, A, B, π as before,
- $S \in \mathbb{N}$ number of samples

yields $\mathcal{S} \subseteq \{1, \ldots, H\}^T$ set of posterior latent state samples

Outline

1. Hidden Markov Models (HMMs)

2. Inference in HMMs

3. Inference in HMMs II: MAP

4. Learning HMMs

Learning HMMs

Learning an HMM means to estimate its parameters $\Theta := (\pi, A, B)$ from observation data $\mathcal{D} \subset X^*$

$\pi := (p(z_1 = h))_{h=1:H}$ **hidden state start vector**

$A := (p(z_{t+1} = h \mid z_t = g))_{g=1:H,h=1:H}$ **hidden state transition matrix**

$B := (p(x_t = i \mid z_t = h))_{h=1:H,i=1:I}$ **observation matrix** (discrete)

or

$B := (\mu_h, \Sigma_h^2)_{h=1:H}$ **observation means/var** (Gaussian

Learning HMMs from Complete Data

When data is completely observed, i.e., also "hidden" states are observed:

$$\mathcal{D} \subset (X \times \{1, 2, \ldots, H\})^*$$

- learning is straight-forward
- estimate π, A as for Markov models
- estimate B_h from the state-specific data subset

$$\mathcal{D}|_h := \{x \mid (x, h') \in \mathcal{D}, h' = h\}$$

  - e.g., for discrete observation models:

$$B_{h,i} := \frac{N_{h,i}}{N_h}$$

$$N_{h,i} := \sum_{n=1}^{N} \sum_{t=1}^{T_n} \mathbb{I}(h_{n,t} = h, x_{n,t} = i), \quad N_h := \sum_{n=1}^{N} \sum_{t=1}^{T_n} \mathbb{I}(h_{n,t} = h)$$

Learning HMMs from Complete Data

- estimate B_h from the state-specific subset $\mathcal{D}|_h$
  - e.g., for Gaussian observation models:

$$\mu_h := \overline{x}_h / N_h, \quad \Sigma_h^2 := (\overline{xx}_h - N_h \mu_h \mu_h^T)/N_h$$

$$\overline{x}_h := \sum_{n=1}^{N} \sum_{t=1}^{T_n} \mathbb{I}(h_{n,t} = h) x_{n,t}$$

$$\overline{xx}_h := \sum_{n=1}^{N} \sum_{t=1}^{T_n} \mathbb{I}(h_{n,t} = h) x_{n,t} x_{n,t}^T$$

Learning HMMs via EM / Naive

Complete loglikelihood:

$$\ell(\pi, A, B; z_{1:N}; x_{1:N}) = \sum_{n=1}^{N} \log \pi_{z_{n,1}} + \sum_{t=1}^{T_n - 1} \log A_{z_{n,t}, z_{n,t+1}} + \sum_{t=1}^{T_n} \log B_{z_{n,t}, x_{n,t}}$$

block coordinate descent / EM:

- maximize w.r.t. π, A, B (maximize, M-step):
  - as learning HMMs from complete data
- maximize w.r.t. z (estimate, E-step):

$$z_n := \arg\max_{z_{1:T}} p(z_{1:T} \mid x_{n,1:T_n})$$

  - MAP / Viterbi algorithm

Learning HMMs via EM (Baum-Welch)

- naive version is inefficient and brittle
  - as only a single completion $z_{1:T}$ per instance is used
- assume we would have access to the distribution $p(z_{1:T} \mid x_{1:T})$ of completions
  - we only would need
    - $p(z_1 \mid x_{1:T}) = \gamma_1$ to estimate π and
    - $p(z_t \mid x_{1:T}) = \gamma_t$ to estimate B and
    - $p(z_t, z_{t+1} \mid x_{1:T}) =: \xi_t$ to estimate A.

$$\xi_t := p(z_t, z_{t+1} \mid x_{1:T}) \quad \textbf{two-slice smoothed marginals}$$

$$\begin{aligned}
&\propto p(z_t \mid x_{1:t}) p(z_{t+1} \mid z_t, x_{t+1:T}) \\
&\propto p(z_t \mid x_{1:t}) p(x_{t+1:T} \mid z_t, z_{t+1}) p(z_{t+1} \mid z_t) \\
&\propto p(z_t \mid x_{1:t}) p(x_{t+1} \mid z_{t+1}) p(x_{t+2:T} \mid z_{t+1}) p(z_{t+1} \mid z_t) \\
&= \alpha_t (B_{.,x_{t+1}} \odot \beta_{t+1})^T \odot A
\end{aligned}$$

Smoothing / Forwards-Backwards Algorithm

with two-sliced smoothed marginals

sinfer-moothing-forwards-backwards(x, A, B, π):
  $\alpha :=$ filtering-forwards(x, A, B, π)
  $\beta :=$ backwards(x, A, B)
  $\gamma := \alpha \odot \beta$
  for $t = 1 : T$:
    $\xi_t := \alpha_t (B_{.,x_{t+1}} \odot \beta_{t+1})^T \odot A$
  return $\gamma, \xi_{1:T}$

where

- x, A, B, π as for forwards algorithm

yields $\gamma_{1:T} = (p(z_t \mid x_{1:T}))_{t=1:T}$ smoothed latent state
and $\xi_{1:T} = (p(z_t, z_{t+1} \mid x_{1:T}))_{t=1:T}$ two-slice smoothed marginals

Learning HMMs via EM

block coordinate descent / EM:

- maximize w.r.t. π, A, B (maximize, M-step):

$$\pi := \text{normalize}(\sum_{n=1}^{N} \gamma_{n,1})$$

$$A := \text{normalize-rows}(\sum_{n=1}^{N} \sum_{t=1}^{T_n - 1} \xi_{n,t})$$

$$\tilde{B}_{.,i} := \sum_{n=1}^{N} \sum_{t=1}^{T} \gamma_{n,t} \mathbb{I}(x_{n,t} = i), \quad i = 1, \ldots, I$$

$$B := \text{normalize-rows}(\tilde{B})$$

- maximize w.r.t. γ, ξ (estimate, E-step):
  - estimate γ_n, ξ_n using forwards-backwards algorithm for x_n, $n = 1 : N$

Learning HMMs via EM

```
learn-HMM-EM(x_{1:N}):
    initialize  π, A, B
  do until convergence:
     for n = 1 : N:
       γ_n, ξ_n := smoothing-forwards-backwards(x_n, π, A, B)
     π := normalize(∑_{n=1}^{N} γ_{n,1})
     A := normalize-rows(∑_{n=1}^{N} ∑_{t=1}^{T_n−1} ξ_{n,t})
     for i = 1 : I:
       B̃_{.,i} := ∑_{n=1}^{N} ∑_{t=1}^{T} γ_{n,t} 𝕀(x_{n,t} = i)
     B := normalize-rows(B̃)
  return π, A, B
```

where

- $x_{1:N}$ with $x_n \in \{1, 2, \ldots, L\}^*$ observed sequences

yields π, A, B HMM parameters

Learning HMMs via EM

- $\gamma_{n,t,h}$ is the case weight for case $(h, x_{n,t})$ for the observation model

- $\xi_{n,t,g,h}$ is the case weight for case (g, h) (for instance n, at time t) for the transition model

- this way EM generalizes to any observation and transition model by just replacing the M-step

Summary

- **Hidden Markov Models** (HMMs) model **sequences** via
  - a Markov Model on hidden states: **transition model** $p(z_{t+1} \mid z_t)$ and
  - a model for observations per hidden state: **observation model** $p(x_t \mid z_t)$.
- The number of hidden states describes the **complexity** of a HMM.
- The probability $p(z_t \mid x_{1:t})$ of the current hidden state based on past observations can be inferred online (**filtering**; **forwards algorithm**).
- The probability $p(z_t \mid x_{1:T})$ of a hidden state based on past and future observations can be inferred by a two-pass algorithm (**smoothing**; **forwards-backwards algorithm**).
- The jointly most-probable hidden state sequence can be inferred using a two-pass algorithm (**MAP**; **Viterbi algorithm**).

Summary (2/2)

- If "hidden" states are observed, HMMs are just Markov models and parameters can be learnt from observations by counting.
- For truely hidden states, HMMs can be learnt by an **EM algorithm** (**Baum-Welche algorithm**)
  - forwards-backwards algorithm is used for the E-step.

Further Readings

- Hidden Markov Models:
  Murphy 2012, chapter 17.

References

Kevin P. Murphy. *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012.