Exact Inference for Hidden Markov Models

Chris Williams
(based on slides by Michael U. Gutmann)

Probabilistic Modelling and Reasoning (INFR11134)
School of Informatics, The University of Edinburgh

Spring Semester 2024

Recap

- Assuming a factorisation / set of statistical independencies allowed us to efficiently represent the pdf or pmf of random variables
- Factorisation can be exploited for inference
  - by using the distributive law
  - by re-using already computed quantities
- Inference for general factor graphs (variable elimination)
- Inference for factor trees
- Sum-product and max-product/max-sum message passing

Program

1. Markov models

2. Inference by message passing

Program

1. Markov models
   - Markov chains
   - Transition distribution
   - Hidden Markov models
   - Emission distribution
   - Mixture of Gaussians as special case
   - Linear Dynamical System (LDS)

2. Inference by message passing

Applications of (hidden) Markov models

Markov and hidden Markov models have many applications, e.g.

- speech modelling (speech recognition)
- text modelling (natural language processing)
- gene sequence modelling (bioinformatics)
- spike train modelling (neuroscience)
- object tracking (robotics)

Markov chains

- Chain rule with ordering $x_1, \ldots, x_d$

$$p(x_1, \ldots, x_d) = \prod_{i=1}^{d} p(x_i | x_1, \ldots, x_{i-1})$$

- If p satisfies ordered Markov property, the number of variables in the conditioning set can be reduced to a subset $\pi_i \subseteq \{x_1, \ldots, x_{i-1}\}$
- Not all predecessors but only subset π_i is "relevant" for x_i.
- L-th order Markov chain: $\pi_i = \{x_{i-L}, \ldots, x_{i-1}\}$

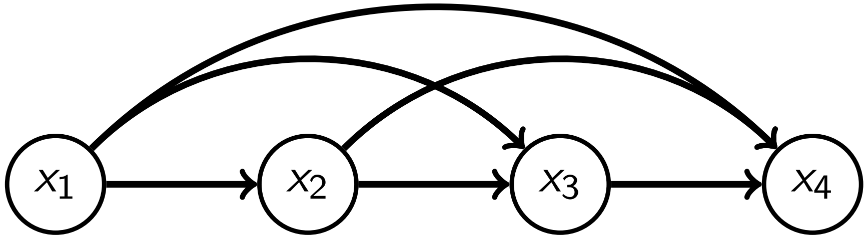
$$p(x_1, \ldots, x_d) = \prod_{i=1}^{d} p(x_i | x_{i-L}, \ldots, x_{i-1})$$

- 1st order Markov chain: $\pi_i = \{x_{i-1}\}$

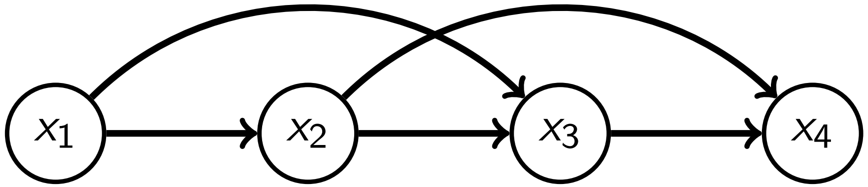
$$p(x_1, \ldots, x_d) = \prod_{i=1}^{d} p(x_i | x_{i-1})$$

Markov chain — DAGs

Chain rule



Second-order Markov chain

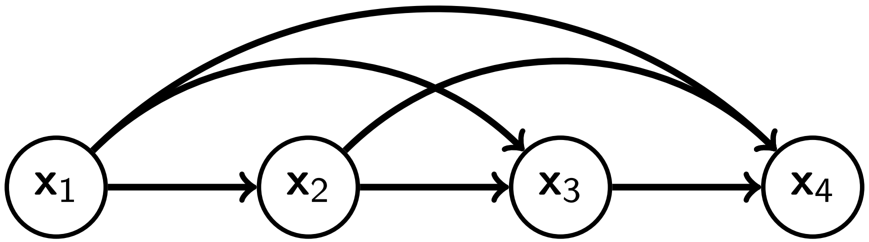


First-order Markov chain

Vector-valued Markov chains

- While not explicitly discussed, the graphical models extend to vector-valued variables.
- Chain rule with ordering $\mathbf{x}_1, \ldots, \mathbf{x}_d$

$$p(\mathbf{x}_1, \ldots, \mathbf{x}_d) = \prod_{i=1}^{d} p(\mathbf{x}_i | \mathbf{x}_1, \ldots, \mathbf{x}_{i-1})$$



- 1st order Markov chain:

$$p(\mathbf{x}_1, \ldots, \mathbf{x}_d) = \prod_{i=1}^{d} p(\mathbf{x}_i | \mathbf{x}_{i-1})$$

Modelling time series

- Index i may refer to time t
- For example, 1st order Markov chain of length T:

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

- Only the last time point x_{t-1} is relevant for x_t.

Transition distribution

(Consider 1st order Markov chain.)

- $p(x_i|x_{i-1})$ is called the transition distribution
- For discrete random variables, $p(x_i|x_{i-1})$ is defined by a transition matrix $\mathbf{A}^{(i)}$

$$p(x_i = k|x_{i-1} = k') = A^{(i)}_{k,k'} \qquad (A^{(i)}_{k',k} \text{ convention is also used})$$

- For continuous random variables, $p(x_i|x_{i-1})$ is a conditional pdf, e.g.

$$p(x_i|x_{i-1}) = \frac{1}{\sqrt{2\pi\sigma_i^2}} \exp\left(-\frac{(x_i - f_i(x_{i-1}))^2}{2\sigma_i^2}\right)$$

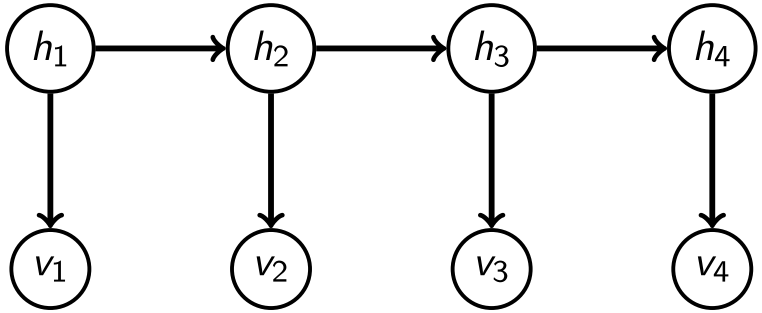
  for some function f_i
- Homogeneous Markov chain: $p(x_i|x_{i-1})$ does not depend on i, e.g.

$$\mathbf{A}^{(i)} = \mathbf{A} \quad \text{or} \qquad \sigma_i = \sigma, \quad f_i = f$$

- Inhomogeneous Markov chain: $p(x_i|x_{i-1})$ does depend on i

Hidden Markov model

DAG:



- 1st order Markov chain on hidden (latent) variables h_i.
- Each visible (observed) variable v_i only depends on the corresponding hidden variable h_i
- Factorisation

$$p(h_{1:d}, v_{1:d}) = p(v_1|h_1)p(h_1) \prod_{i=2}^{d} p(v_i|h_i)p(h_i|h_{i-1})$$

- The visibles are d-connected if hiddens are not observed
- Visibles are d-separated (independent) given the hiddens
- The h_is model/explain all dependencies between the v_is

Emission distribution

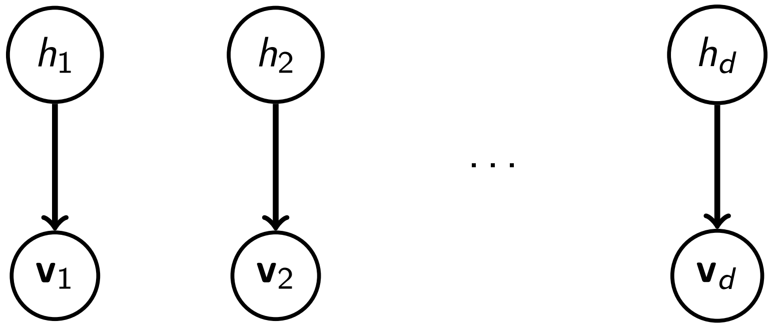
- $p(v_i|h_i)$ is called the emission distribution
- Discrete-valued v_i and h_i:
  $p(v_i|h_i)$ can be represented as a matrix
- Discrete-valued v_i and continuous-valued h_i:
  $p(v_i|h_i)$ is a conditional pmf.
- Continuous-valued v_i: $p(v_i|h_i)$ is a density
- As for the transition distribution, the emission distribution $p(v_i|h_i)$ may depend on i or not.
- If neither the transition nor the emission distribution depend on i, we have a stationary (or homogeneous) hidden Markov model (HMM).

Gaussian emission model with discrete-valued latents

- Special case: $h_i \perp\!\!\!\perp h_{i-1}$, and $\mathbf{v}_i \in \mathbb{R}^m, h_i \in \{1, \ldots, K\}$

  $$p(h = k) = p_k$$
  $$p(\mathbf{v}|h = k) = \frac{1}{|\det 2\pi\boldsymbol{\Sigma}_k|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{v} - \boldsymbol{\mu}_k)^\top \boldsymbol{\Sigma}_k^{-1}(\mathbf{v} - \boldsymbol{\mu}_k)\right)$$

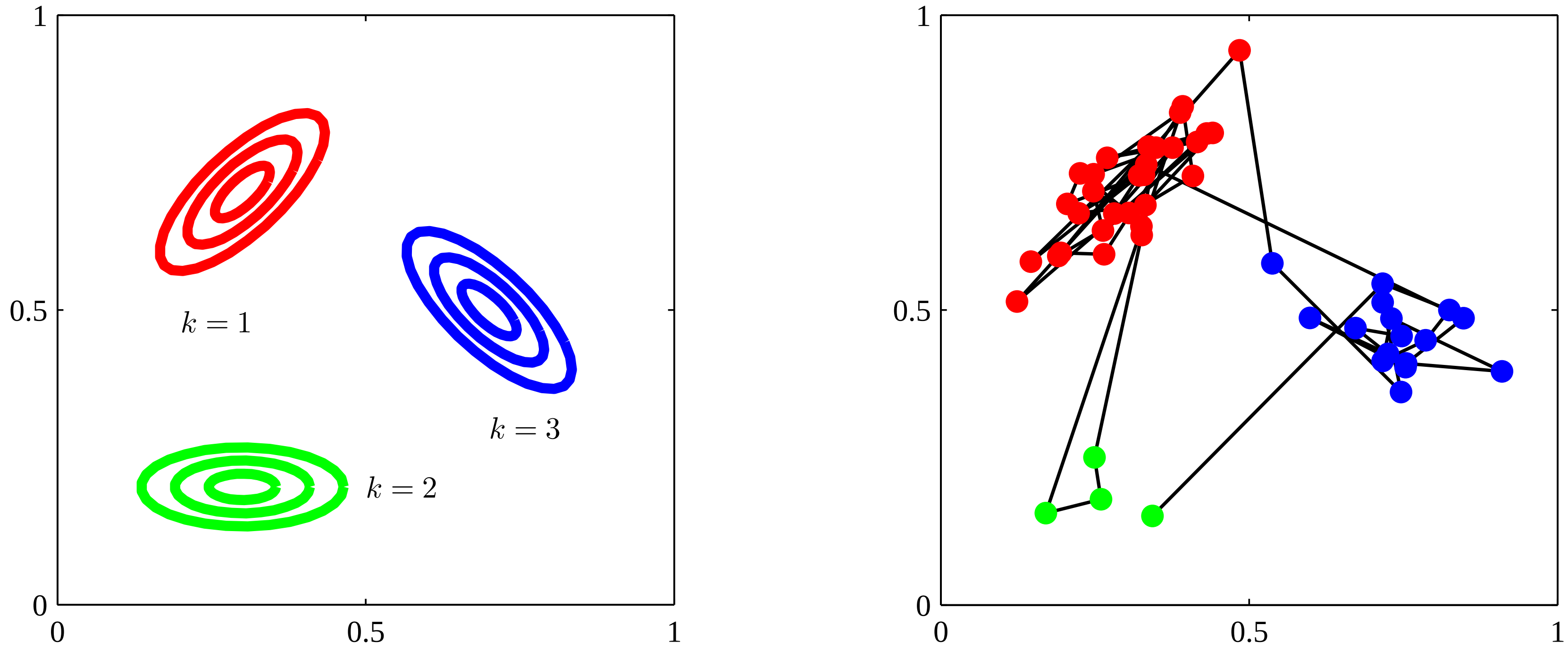
  for all h_i and $\mathbf{v}_i$.
- DAG



- Corresponds to d iid draws from a Gaussian mixture model with K mixture components
  - Mean $\mathbb{E}[\mathbf{v}|h = k] = \boldsymbol{\mu}_k$
  - Covariance matrix $\mathbb{V}[\mathbf{v}|h = k] = \boldsymbol{\Sigma}_k$

Gaussian emission model with discrete-valued latents

The HMM is a generalisation of the Gaussian mixture model where cluster membership at "time" i (the value of h_i) generally depends on cluster membership at "time" $i-1$ (the value of h_{i-1}).



Example for $\mathbf{v}_i \in \mathbb{R}^2$, $h_i \in \{1, 2, 3\}$. Left: $p(\mathbf{v}|h = k)$. Right: samples

(Bishop, Figure 13.8)

Linear Dynamical System (LDS)

- Continuous-valued hidden and visible state
- Transition model is linear

$$\mathbf{h}_{t+1} = A\mathbf{h}_t + \mathbf{n}^h_{t+1}, \qquad \mathbf{n}^h_{t+1} \sim N(0, \Sigma^h)$$

- Stable dynamics if all eigenvalues of A have magnitude < 1
- Emission model is linear

$$\mathbf{v}_t = C\mathbf{h}_t + \mathbf{n}^v_t, \qquad \mathbf{n}^v_t \sim N(0, \Sigma^v)$$

- If $p(\mathbf{h}_1)$ is Gaussian, the whole model is jointly Gaussian
- Computation of $p(\mathbf{h}_t|\mathbf{v}_{1:t})$ is the filtering problem: for the LDS, this was solved by Kalman (1960), hence it is termed *Kalman filtering*
- Uses: navigational and guidance systems

Program

1. Markov models
   - Markov chains
   - Transition distribution
   - Hidden Markov models
   - Emission distribution
   - Mixture of Gaussians as special case
   - Linear Dynamical System (LDS)

2. Inference by message passing

Program

1. Markov models

2. Inference by message passing
   - Inference: filtering, prediction, smoothing, Viterbi
   - Filtering: Sum-product message passing yields the α-recursion
   - Smoothing: Sum-product message passing yields the α-β recursion

The classical inference problems

(Considering the index i to refer to time t)

Filtering	(Inferring the present)	$p(h_t \mid v_{1:t})$	
Smoothing	(Inferring the past)	$p(h_t \mid v_{1:u})$	$t < u$
Prediction	(Inferring the future)	$p(h_t \mid v_{1:u})$	$t > u$
		$p(v_t \mid v_{1:u})$	$t > u$
Most likely hidden path	(Viterbi algorithm)	$\mathrm{argmax}_{h_{1:t}}\, p(h_{1:t} \mid v_{1:t})$	
Posterior sampling	(Forward filtering backward sampling)	$h_{1:t} \sim p(h_{1:t} \mid v_{1:t})$	

For the HMM, all tasks can be solved via message passing (sum-product or max-sum/max-product algorithm).

The classical inference problems

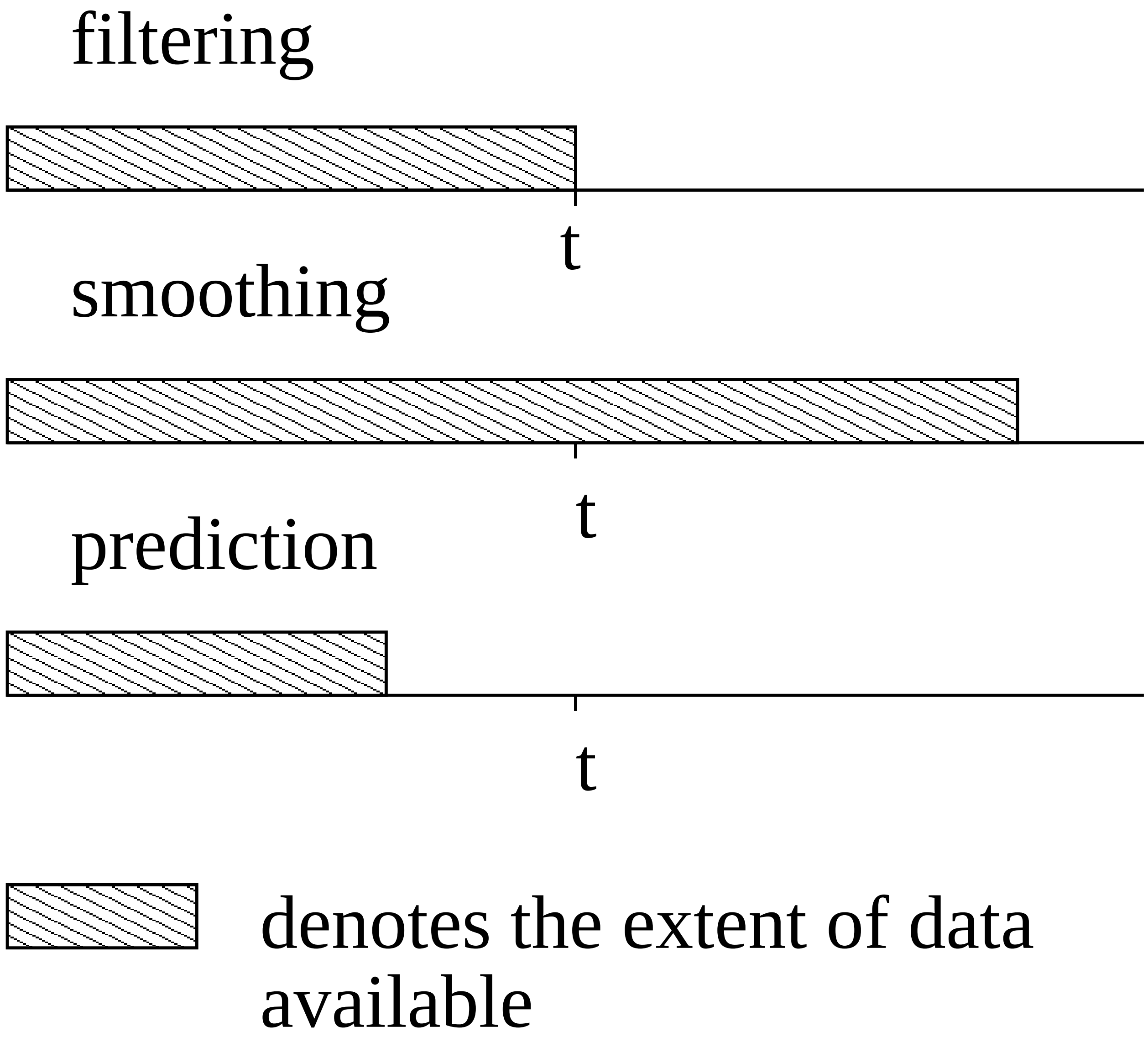
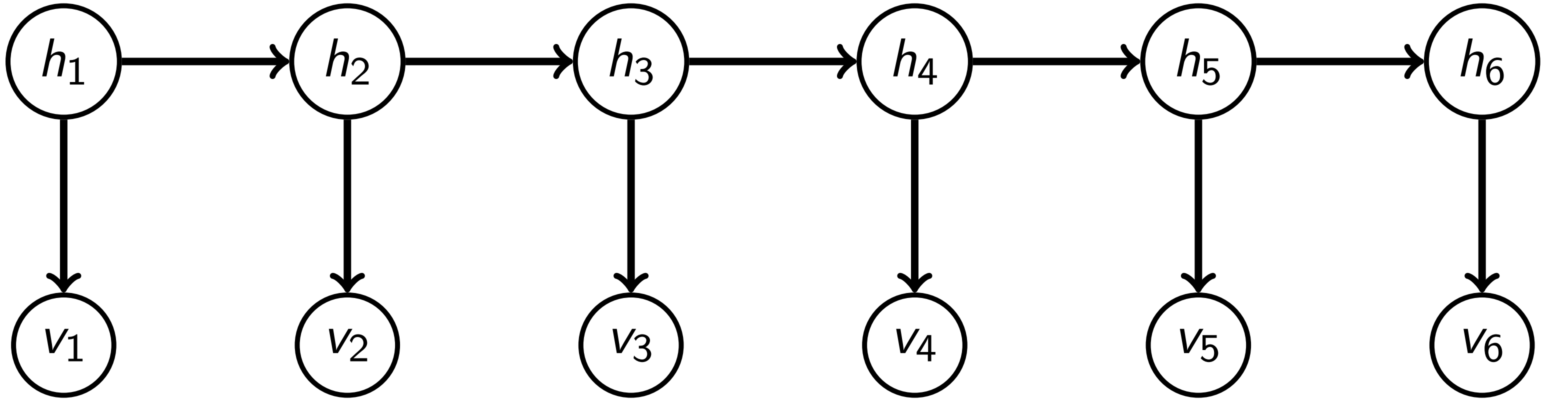


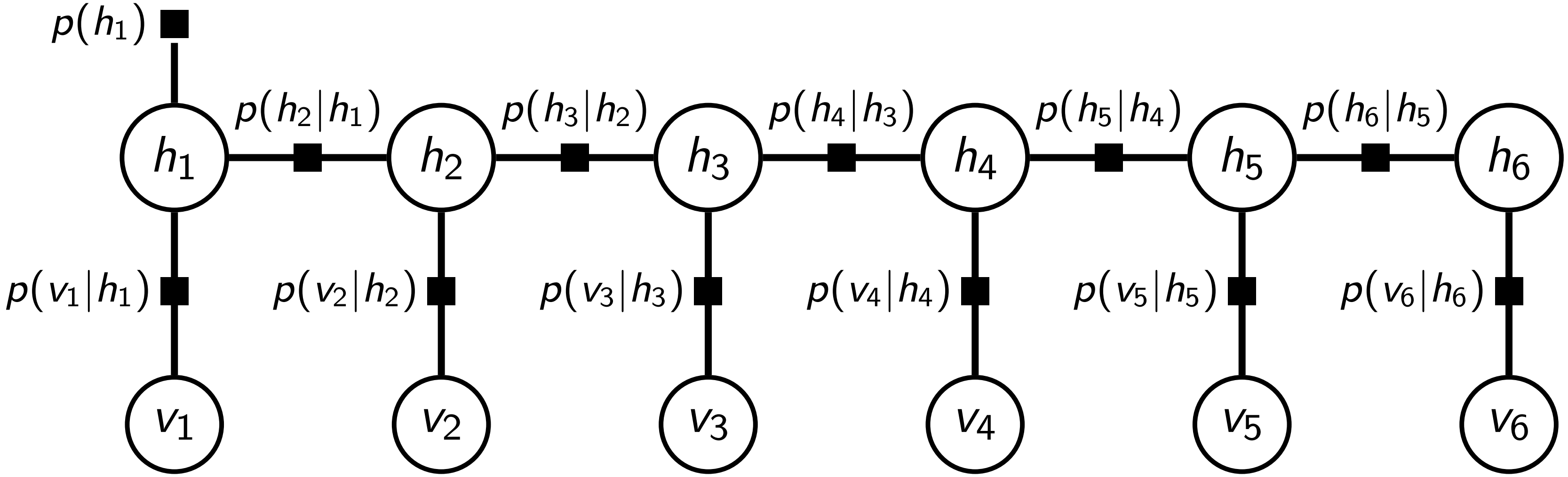
Figure based on Fig. 1.0-1 of Gelb et al (1974)

Factor graph for hidden Markov model

DAG:



Factor graph:

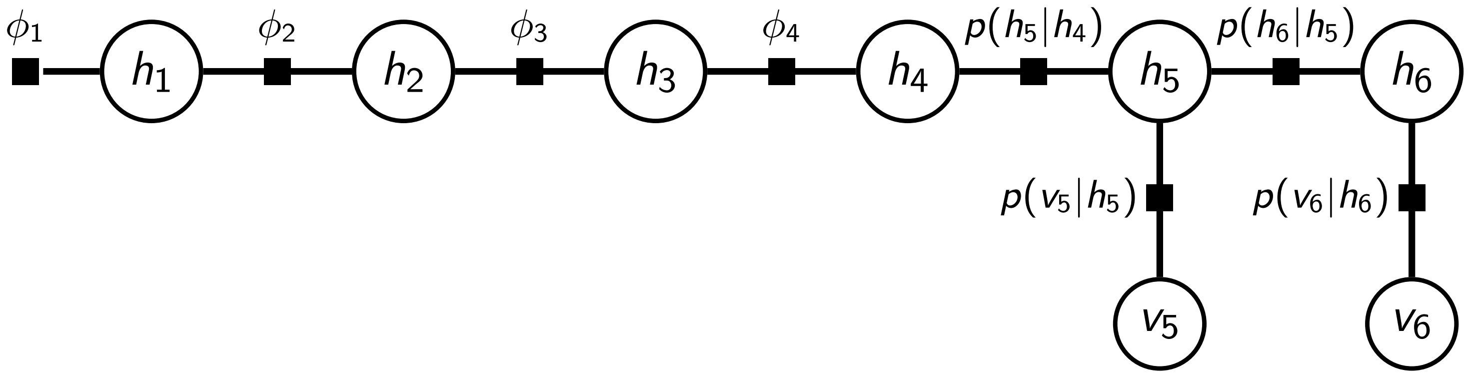


Filtering $p(h_t|v_{1:t})$: factor graph

- When computing $p(h_t|v_{1:t})$, the $v_{1:t} = (v_1, \ldots, v_t)$ are assumed known and are kept fixed (e.g. $t = 4$)
- For $s = 1, \ldots, t$, the factors $p(v_s|h_s)$ depend only on h_s. Combine them with $p(h_s|h_{s-1})$ and form new factors ϕ_s

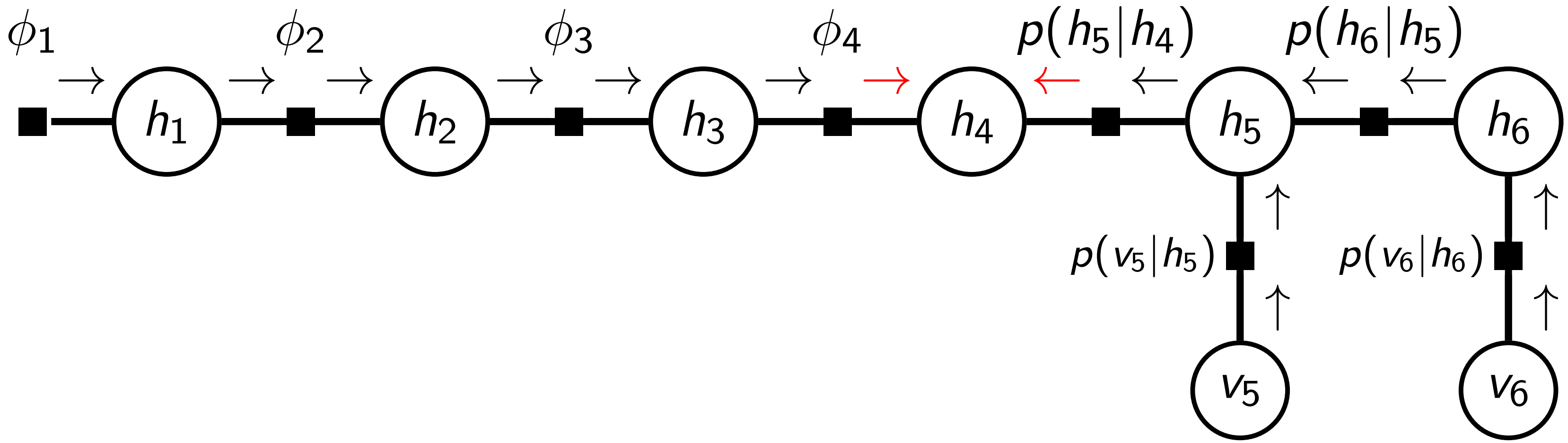
$$\phi_1(h_1) = p(v_1|h_1)p(h_1), \quad \phi_s(h_{s-1}, h_s) = p(v_s|h_s)p(h_s|h_{s-1})$$

- Factor graph



Filtering $p(h_t|v_{1:t})$: messages

Messages needed to compute $p(h_4|v_{1:4})$: $(t = 4)$



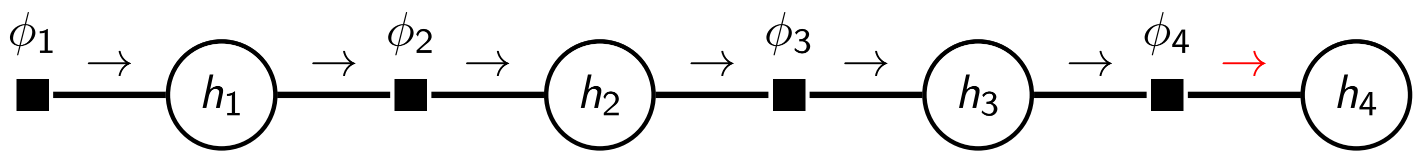
There is a simplification:

- The message from $p(h_5|h_4)$ to h_4 equals 1!
- Follows from message passing starting at leaves v_5 and v_6 since the factors $p(.|.)$ are conditionals and sum to one, e.g.

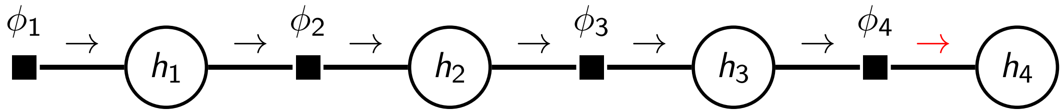
$$\sum_{v_6} p(v_6|h_6) = 1 \qquad \sum_{h_6} p(h_6|h_5) = 1$$

Filtering $p(h_t | v_{1:t})$: reduce to inference on chain

- A message is an effective factor obtained by summing out all variables downstream from where the message is coming from.
- This means that we can replace the factor sub-graph to the right of the last observed variable v_t and latent h_t (here v_4 and h_4) with the effective factor.
- Effective factor is 1, so that we can just remove the sub-graph.
- Also can be seen by "marginalising out" the unobserved future
- Reduces problem to message passing on a chain.



Filtering $p(h_t|v_{1:t})$: message passing on the chain



- Initialisation: $\mu_{\phi_1 \to h_1}(h_1) = \phi_1(h_1)$
- Variable node h_1 copies the message:

$$\mu_{h_1 \to \phi_2}(h_1) = \mu_{\phi_1 \to h_1}(h_1)$$

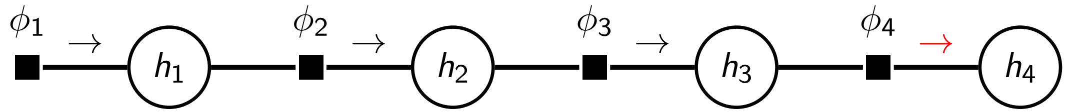
- Same for other variable nodes. Let us write the algorithm in terms of $\mu_{\phi_i \to h_i}(h_i)$ messages only.
- Message from ϕ_2 to h_2:

$$\mu_{\phi_2 \to h_2}(h_2) = \sum_{h_1} \phi_2(h_1, h_2)\mu_{\phi_1 \to h_1}(h_1)$$

- Message from ϕ_s to h_s, for $s = 2, \ldots, t$:

$$\mu_{\phi_s \to h_s}(h_s) = \sum_{h_{s-1}} \phi_s(h_{s-1}, h_s)\mu_{\phi_{s-1} \to h_{s-1}}(h_{s-1})$$

Filtering $p(h_t|v_{1:t})$: message passing on the chain



- The messages $\mu_{\phi_s \to h_s}(h_s)$ are traditionally denoted by $\alpha(h_s)$.
- Message passing for filtering becomes:
  - Init: $\alpha(h_1) = \phi_1(h_1) = p(v_1|h_1)p(h_1)$
  - Update rule for $s = 2, \ldots t$:

$$\alpha(h_s) = \sum_{h_{s-1}} \phi_s(h_{s-1}, h_s)\alpha(h_{s-1})$$
$$= p(v_s|h_s) \sum_{h_{s-1}} p(h_s|h_{s-1})\alpha(h_{s-1})$$

- Algorithm known as "alpha-recursion".
- Desired probability:

$$p(h_t|v_{1:t}) = \frac{1}{Z_t}\alpha(h_t) \qquad\qquad Z_t = \sum_{h_t} \alpha(h_t)$$

Filtering $p(h_t|v_{1:t})$: likelihood

- Joint model for $h_{1:t}$ and $v_{1:t}$

$$p(h_{1:t}, v_{1:t}) = p(v_1|h_1)p(h_1)\prod_{i=2}^{t} p(v_i|h_i)p(h_i|h_{i-1})$$

- Conditional $p(h_{1:t}|v_{1:t})$ is *proportional* to the joint

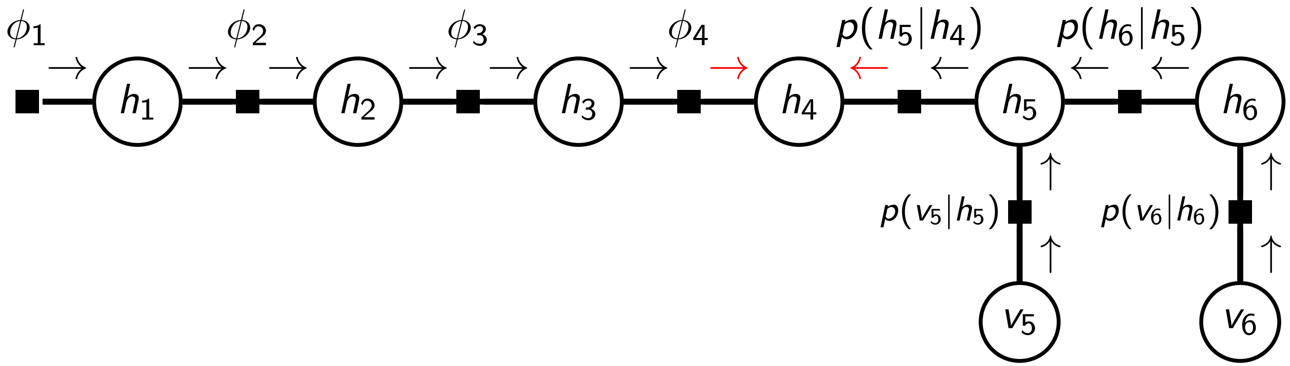
$$p(h_{1:t}|v_{1:t}) \propto p(v_1|h_1)p(h_1)\prod_{i=2}^{t} p(v_i|h_i)p(h_i|h_{i-1})$$

- Normalising constant Z is the likelihood/marginal $p(v_{1:t})$
- From results on message passing: Z_t that normalises the marginal is also the normaliser of $p(h_{1:t}|v_{1:t})$, i.e. $p(v_{1:t})$:

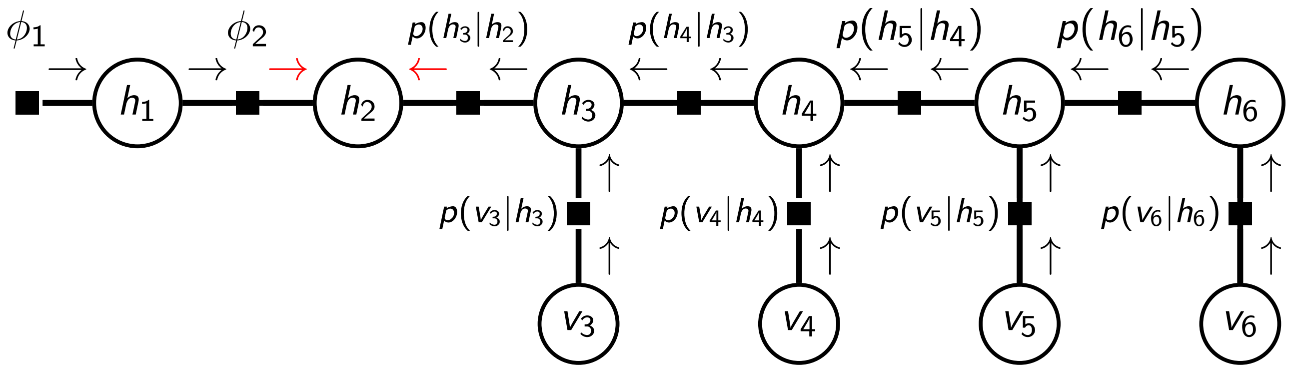
$$Z_t = \sum_{h_t} \alpha(h_t) = p(v_{1:t})$$

Filtering $p(h_t|v_{1:t})$: interpretation

- We have seen that $p(h_t|v_{1:t}) \propto \alpha(h_t)$.



- Consider $p(h_s|v_{1:s})$ with $s < t$ (e.g. $s = 2$ and $t = 4$)



- Messages to the left of h_s are the same as for $p(h_t|v_{1:t})$.
- Messages to the right of h_s are all equal to one.

Filtering $p(h_t|v_{1:t})$: interpretation

- This means that the intermediate $\alpha(h_s)$ that we compute when computing $p(h_t|v_{1:t})$ are unnormalised posteriors themselves:

  $$\alpha(h_s) \propto p(h_s|v_{1:s})$$

  Note that we condition on $v_{1:s}$ and not $v_{1:t}$.
- Moreover $p(v_{1:s}) = \sum_{h(s)} \alpha(h_s)$.
- Hence, the alpha-recursion gives us posteriors $p(h_s|v_{1:s})$ and likelihoods $p(v_{1:s})$ for $s = 1, \ldots, t$.

Filtering $p(h_t|v_{1:t})$: interpretation

- Proof by induction shows that $\alpha(h_s) = p(h_s, v_{1:s})$.
- Base case holds by definition: $\alpha(h_1) = p(h_1)p(v_1|h_1)$.
- Assume it holds for $\alpha(h_{s-1})$. Then:

$$
\begin{aligned}
\alpha(h_s) &= \sum_{h_{s-1}} p(v_s|h_s)p(h_s|h_{s-1})\alpha(h_{s-1}) \\
&\overset{\text{(induction hyp)}}{=} \sum_{h_{s-1}} p(v_s|h_s)p(h_s|h_{s-1})p(h_{s-1}, v_{1:s-1}) \\
&\overset{\text{(Markov prop)}}{=} \sum_{h_{s-1}} p(v_s|h_s, h_{s-1}, v_{1:s-1})p(h_s|h_{s-1}, v_{1:s-1})p(h_{s-1}, v_{1:s-1}) \\
&\overset{\text{(product rule)}}{=} \sum_{h_{s-1}} p(v_s|h_s, h_{s-1}, v_{1:s-1})p(h_s, h_{s-1}, v_{1:s-1}) \\
&\overset{\text{(product rule)}}{=} \sum_{h_{s-1}} p(v_s, h_s, h_{s-1}, v_{1:s-1}) \\
&\overset{\text{(marginalise)}}{=} p(v_s, h_s, v_{1:s-1}) \\
&= p(h_s, v_{1:s})
\end{aligned}
$$

Filtering $p(h_t|v_{1:t})$: interpretation

- Update rule as prediction-correction algorithm:

$$
\begin{aligned}
\alpha(h_s) &\overset{\text{(prev slide)}}{=} p(h_s, v_{1:s}) \\
&\overset{\text{(product rule)}}{=} p(v_s|h_s, v_{1:s-1})p(h_s, v_{1:s-1}) \\
&\overset{\text{(Markov prop)}}{=} p(v_s|h_s)p(h_s, v_{1:s-1}) \\
&\propto \quad \underbrace{p(v_s|h_s)}_{\text{correction}}\underbrace{p(h_s|v_{1:s-1})}_{\text{prediction}}
\end{aligned}
$$

- The correction term updates the predictive distribution $p(h_s|v_{1:s-1})$ to include the new data v_s.

Filtering $p(h_t|v_{1:t})$: summary

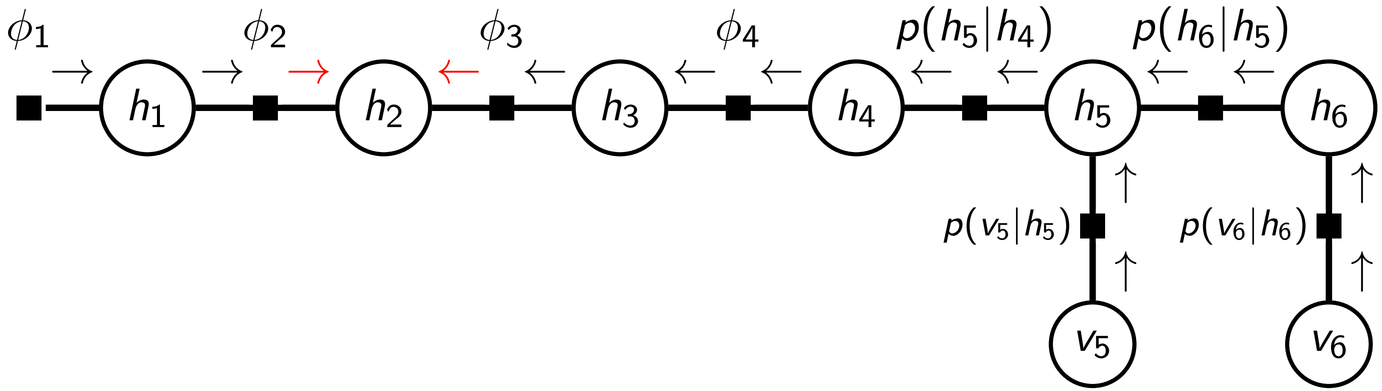
- Conditioning reduces the factor graph for the HMM to a chain.
- Message passing for filtering:
  - Init: $\alpha(h_1) = p(v_1|h_1)p(h_1)$
  - Update rule for $s = 2, \ldots t$:

    $$\alpha(h_s) = p(v_s|h_s) \sum_{h_{s-1}} p(h_s|h_{s-1})\alpha(h_{s-1})$$

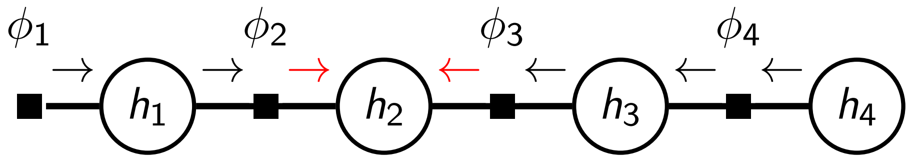
    which involves prediction of h_s given $v_{1:s-1}$ and correction using new datum v_s.
- $\alpha(h_s) = p(h_s, v_{1:s}) \propto p(h_s|v_{1:s})$ and $p(v_{1:s}) = \sum_{h_s} \alpha(h_s)$, for $s = 1, \ldots, t$

Smoothing $p(h_t|v_{1:u}), t < u$: reduce to inference on chain

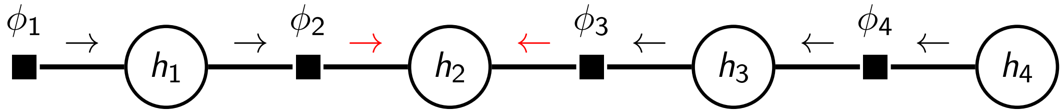
- Unlike in filtering where we predict h_t from data up to time t, in smoothing we have observations from later time points.
- Messages needed to compute $p(h_t|v_{1:u})$ (e.g. $t = 2$, $u = 4$)



- As in filtering, we can simplify to a chain



Smoothing $p(h_t|v_{1:u})$, $t < u$: message passing on chain

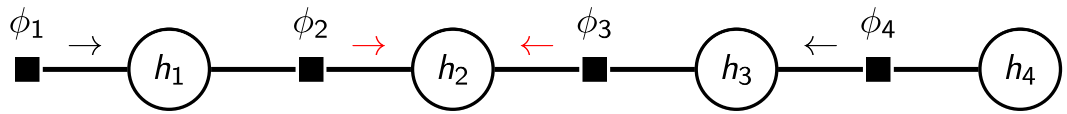


- Messages $\rightarrow$ from factor leaf ϕ_1 to h_t same as in filtering.
- Messages $\leftarrow$ from variable leaf h_u to h_t via message passing.
- Init: $\mu_{h_u \rightarrow \phi_u}(h_u) = 1$
- Next message $\mu_{\phi_u \rightarrow h_{u-1}}(h_{u-1}) = \sum_{h_u} \phi_u(h_{u-1}, h_u)$
- Variable nodes just copy the incoming message. Write the algorithm in terms of $\beta(h_s) = \mu_{\phi_{s+1} \rightarrow h_s}(h_s)$ only:

$$
\begin{aligned}
\beta(h_{s-1}) &= \sum_{h_s} \phi_s(h_{s-1}, h_s)\beta(h_s) \\
&= \sum_{h_s} p(v_s|h_s)p(h_s|h_{s-1})\beta(h_s)
\end{aligned}
$$

- Gives "alpha-beta recursion" for smoothing.

Smoothing $p(h_t|v_{1:u})$, $t < u$: message passing on chain



- $\rightarrow$ Forwards via alpha-recursion
  - Init: $\alpha(h_1) = p(v_1|h_1)p(h_1)$
  - Update rule for $s = 2, \ldots t$:

$$\alpha(h_s) = p(v_s|h_s) \sum_{h_{s-1}} p(h_s|h_{s-1})\alpha(h_{s-1})$$

- $\leftarrow$ Backwards via beta-recursion
  - Init: $\beta(h_u) = 1$
  - Update rule for $s = u, \ldots t+1$:

$$\beta(h_{s-1}) = \sum_{h_s} p(v_s|h_s)p(h_s|h_{s-1})\beta(h_s)$$

- Desired probability:

$$p(h_t|v_{1:u}) = \frac{1}{Z_t^u}\alpha(h_t)\beta(h_t) \qquad Z_t^u = \sum_{h_t} \alpha(h_t)\beta(h_t)$$

Smoothing $p(h_t|v_{1:u})$, $t < u$: interpretation

- We now show that $\beta(h_s)$ equals the probability of the upstream observations given h_s,

$$\beta(h_s) = p(v_{s+1:u}|h_s) \quad \text{for all } s < u$$

- First consider $\beta(h_{u-1})$:

$$\begin{aligned}
\beta(h_{u-1}) &= \sum_{h_u} p(v_u|h_u)p(h_u|h_{u-1})\underbrace{\beta(h_u)}_{1} \\
&\overset{\text{(Markov prop)}}{=} \sum_{h_u} p(v_u|h_u, h_{u-1})p(h_u|h_{u-1}) \\
&\overset{\text{(product rule)}}{=} \sum_{h_u} p(v_u, h_u|h_{u-1}) \\
&\overset{\text{(marginalise)}}{=} p(v_u|h_{u-1})
\end{aligned}$$

- Hence $\beta(h_s) = p(v_{s+1:u}|h_s)$ holds for $s = u - 1$. Provides the base case for a proof by induction.

Smoothing $p(h_t|v_{1:u})$, $t < u$: interpretation

Assume $\beta(h_s) = p(v_{s+1:u}|h_s)$ holds. Then:

$$
\begin{aligned}
\beta(h_{s-1}) &= \sum_{h_s} p(v_s|h_s)p(h_s|h_{s-1})\beta(h_s) \\
&\overset{\text{(induction hyp)}}{=} \sum_{h_s} p(v_s|h_s)p(h_s|h_{s-1})p(v_{s+1:u}|h_s) \\
&\overset{\text{(Markov prop)}}{=} \sum_{h_s} p(v_s|h_s)p(h_s|h_{s-1})p(v_{s+1:u}|h_s, v_s) \\
&\overset{\text{(product rule)}}{=} \sum_{h_s} p(v_{s:u}|h_s)p(h_s|h_{s-1}) \\
&\overset{\text{(Markov prop)}}{=} \sum_{h_s} p(v_{s:u}|h_s, h_{s-1})p(h_s|h_{s-1}) \\
&\overset{\text{(product rule)}}{=} \sum_{h_s} p(v_{s:u}, h_s|h_{s-1}) \\
&\overset{\text{(marginalise)}}{=} p(v_{s:u}|h_{s-1})
\end{aligned}
$$

By induction, $\beta(h_s) = p(v_{s+1:u}|h_s)$ for all $s < u$.

Doing more with the $\alpha(h_s), \beta(h_s)$

- Due to link to message passing: Knowing all $\alpha(h_s), \beta(h_s) \Longrightarrow$ knowing all marginals *and* all joints of neighbouring latents given the observed data, which will be needed when estimating the parameters of HMMs (see later).
- We can use the $\alpha(h_s)$ for predictions (see exercises).
- We can use the $\alpha(h_s)$ for sampling posterior trajectories, i.e. to sample from $p(h_1, \ldots h_t | v_1, \ldots, v_t)$ (see exercises).
- Algorithms extend to the case of continuous random variables: replace sums with integrals.

Example: Harmonizing Chorales in the Style of J S Bach

- Moray Allan and Chris Williams (NIPS 2004) "Harmonising Chorales by Probabilistic Inference"
- Visible states are the melody (quarter notes)
- Hidden states are the harmony (which chord)
- Trained using labelled melody/harmony data from Bach chorales
- Task: find Viterbi alignment for harmony given melody, or sample from $p(\text{harmony}|\text{melody}.)$
- Actually it is a bit more complicated. HMMs used for three subtasks: harmonic skeleton, chord skeleton, ornamentation

`https://homepages.inf.ed.ac.uk/ckiw/teach/pmr/hmmBach.html`

Further reading

Exact inference for Hidden Markov models is well-covered in the standard textbooks, e.g.

- Bishop (2006) secs. 13.2.2, 13.2.3, 13.2.5
- Barber sec. 23.2

Program recap

1. Markov models
   - Markov chains
   - Transition distribution
   - Hidden Markov models
   - Emission distribution
   - Mixture of Gaussians as special case
   - Linear Dynamical System (LDS)

2. Inference by message passing
   - Inference: filtering, prediction, smoothing, Viterbi
   - Filtering: Sum-product message passing yields the α-recursion
   - Smoothing: Sum-product message passing yields the α-β recursion

Credits

These slides are modified from ones produced by Michael Gutmann, made available under Creative Commons licence CC BY 4.0.