EE596A – Dynamic Graphical Models
Winter Quarter 2013

Prof. Jeff Bilmes

University of Washington, Seattle
Department of Electrical Engineering
Winter Quarter, 2013
http://j.ee.washington.edu/~bilmes/classes/ee596a_winter_2013/

Lecture 3 - Jan 14th, 2013
Reminder: class web links and infrastructure

- Check in with our web page (http://j.ee.washington.edu/~bilmes/classes/ee596a_winter_2013/) for up to date announcements, homeworks, etc.
Reminder: class web links and infrastructure

- Check in with our web page (http://j.ee.washington.edu/~bilmes/classes/ee596a_winter_2013/) for up to date announcements, homeworks, etc.

- All homeworks will be due via our dropbox (https://catalyst.uw.edu/collectit/dropbox/bilmes/25379)
Reminder: class web links and infrastructure

- Check in with our web page (http://j.ee.washington.edu/~bilmes/classes/ee596a_winter_2013/) for up to date announcements, homeworks, etc.
- All homeworks will be due via our dropbox (https://catalyst.uw.edu/collectit/dropbox/bilmes/25379)
- All questions should be posted to our discussion board (https://catalyst.uw.edu/gopost/board/bilmes/31332)
Reminder: class web links and infrastructure

- Check in with our web page (http://j.ee.washington.edu/~bilmes/classes/ee596a_winter_2013/) for up to date announcements, homeworks, etc.
- All homeworks will be due via our dropbox (https://catalyst.uw.edu/collectit/dropbox/bilmes/25379)
- All questions should be posted to our discussion board (https://catalyst.uw.edu/gopost/board/bilmes/31332)
- You can contact me anonymously if you wish via anonymous email (https://catalyst.uw.edu/umail/form/bilmes/4144)
Announcements, Assignments, and Reminders

- No class this Wednesday, 1/16 (will either have makeup lecture, or video lecture on the web page). Stay tuned to the web page.
- No class on Monday, 1/21 official holiday.
Readings are in a sub-directory “reading_drafts” directly below our web page (http://j.ee.washington.edu/~bilmes/classes/ee596a_winter_2013/).
Readings are in a sub-directory “reading_drafts” directly below our web page (http://j.ee.washington.edu/~bilmes/classes/ee596a_winter_2013/).

uid is this class name (lower case) and pwd are the quarter/year of the class.
Readings are in a sub-directory “reading_drafts” directly below our web page (http://j.ee.washington.edu/~bilmes/classes/ee596a_winter_2013/).

uid is this class name (lower case) and pwd are the quarter/year of the class.

Note, the PDF file is password protected. Send me email if you have trouble (adobe reader should have no problems reading it).
Cumulative Outstanding Reading

- Read 8.1 - 8.3 in “doc.pdf”
- Start reading HMM section in readings.
Review section

- All slides in this section are either from previous lectures, or are a summary of material from previous lectures.
Review section

- All slides in this section are either from previous lectures, or are a summary of material from previous lectures.
- For more details, see the previous lectures.
Homogeneous Markov chains vs. Stationary Markov chains.
Homogeneous Markov chains vs. Stationary Markov chains.

State types: transient or positive-recurrent (or null-recurrent if infinite states).
Homogeneous Markov chains vs. Stationary Markov chains.

State types: transient or positive-recurrent (or null-recurrent if infinite states).

stationary & 1st-order $\Rightarrow$ homogeneous
Homogeneous Markov chains vs. Stationary Markov chains.

State types: transient or positive-recurrent (or null-recurrent if infinite states).

stationary & 1st-order $\Rightarrow$ homogeneous

probability flow
We can generalize this immediately to any $k > 1$ steps to give:

$$a_{ij}^k = \sum_{\ell} a_{i\ell}^m a_{\ell j}^n$$  \hspace{1cm} (3.20)

where $m, n \geq 0$ with $k = m + n$.

Can be further generalized to $n^{th}$-order and/or time-inhomogeneous chains. Example, we can define:

$$a_{ij}(t, t + \tau) = p(Q_{t+\tau} = i | Q_t = i)$$  \hspace{1cm} (3.21)

for $\tau > 0$. 

Definition 3.2.2 (period)

The period \( d(i) \) of a state is defined as 
\[
d(i) = \gcd \{ n : a_{ii}^n > 0 \},
\]
the greatest common divisor of the time intervals where return is possible.

Definition 3.2.3 (periodic/aperiodic)

A state \( i \) is said to be **periodic** if \( d(i) > 1 \) and **aperiodic** if \( d(i) = 1 \).

**Ex 1** If all states have self-loops, period is 1.

**Ex 2** concentric loops, in one case loops of length 3,5, period of state \( i \) is 1.

**Ex 3** another case of length 3,6, period of state \( i \) is 3.
Period of a state

- Period of a state is a probabilistic concept
- if period is $> 1$, then there will be certain times where state is impossible (e.g., 3,6 case above).
- If period $= 1$ (i.e., return times are co-prime), eventually, all states will have positive probability at successive time intervals. Exercise: When is the earliest $t$ when this happens in general?
- i.e., let $\mathbb{N} = \{1, 2, \ldots\}$ be the natural numbers, $i\mathbb{N} = \{i, 2i, 3i, \ldots\}$ for $i \in \mathbb{N}$, and $A + B = \{c : \exists a \in A, b \in B \text{ w. } c = a + b\}$.
- Then, in the above, we have that $3\mathbb{N} + 5\mathbb{N} = \mathbb{N} \cap \{i : i \geq 3 + 5\}$. 

Prof. Jeff Bilmes
EE596A/Winter 2013/DGMs – Lecture 3 - Jan 14th, 2013
State “communication”

Some definitions:

- State $i$ communicates with state $j$ ($i \to j$) if $a_{ij}^k > 0$ for some $k \geq 0$. States intercommunicate if $i \to j$ and $j \to i$, written $i \leftrightarrow j$.
- Set of states $C$ is closed if $a_{ij} = 0$ for $i \in C$ and $j \notin C$.
- Set of states $C$ is irreducible if $i \leftrightarrow j$ for all $i, j \in C$.

**Theorem 3.2.2**

If $i \leftrightarrow j$ then

(a) $i$ and $j$ have same period

(b) $i$ is transient iff $j$ is transient

(c) $i$ is null-recurrent iff $j$ is null-recurrent
State partitioning

Theorem 3.2.2

Set of states can be partitioned uniquely as

\[ S = T \cup C_1 \cup C_2 \cup \ldots \]  \hspace{1cm} (3.20)

where \( T \) is the set of transient states, and \( C_i \) are sets of irreducible closed sets of recurrent states.

- Intuition: When we run a Markov chain (by choosing the next state according to \( a_{ij} \)), states are either such that they eventually are impossible to reach, or we get into a “rut” where only a subset of the states are reachable, and we can never break out of that rut.

- It might be that no states are transient and there is only one \( C_i \) (e.g., a matrix with all strictly positive entries). The entire chain is then irreducible.
Perron-Frobenius

Note: For every component $C_i$ in the above decomposition, there is a unique stationary distribution. If the entire chain is irreducible, there is a single unique stationary distribution. In fact, we have

**Theorem 3.2.4 (Perron-Frobenius)**

If $A$ is the $N \times N$ transition matrix of a finite irreducible chain with period $d$, then

(a) $\lambda_1 = 1$ is an eigenvalue of $A$

(b) the $d$ complex roots of unity, $\lambda_k = W_d^k$, $k \in \{0, \ldots, d - 1\}$ with $W_d = \exp(2\pi \sqrt{-1}/d)$ are eigenvalues of $A$

(c) the remaining eigenvalues $\lambda_{d+1}, \ldots, \lambda_N$ satisfy $|\lambda_j| < 1$. 
Long-term evolution

We also have the following theorem

**Theorem 3.2.4**

*If $A$ is the transition matrix of a finite irreducible chain with period $d$, then*

$$A^k \rightarrow 1\xi^\top$$  \hspace{1cm} (3.24)

*where $\xi$ is the unique stationary distribution associated with the Markov chain, and the rate of convergence, moreover, is geometric.*

Hence, regardless of the initial distribution where we start, we will approach a (unique if irreducible) stationary distribution of the Markov chain, and we’ll do so quite fast. That is, we will forget from where we started from.
An $k$th-order Markov chain is a $k$-tree.
An $k^{\text{th}}$-order Markov chain is a $k$-tree.

Hence, exact inference costs (with the goal a distribution over a clique or subclique), would be $O(r^{k+1})$. 
An $k^{th}$-order Markov chain is a $k$-tree.

Hence, exact inference costs (with the goal a distribution over a clique or subclique), would be $O(r^{k+1})$.

The structure (and “meaning”) of the Markov chain is entirely encoded in the transition matrix $[a_{ij}]_{i,j}$. 
Markov Chains, some applications

- ∃ many applications of Markov chains
Markov Chains, some applications

- ∃ many applications of Markov chains
  - Random walk on a integer point line, with $a_{ij} = 0$ unless $j = i \pm 1$. 


Markov Chains, some applications
∃ many applications of Markov chains

- Random walk on a integer point line, with $a_{ij} = 0$ unless $j = i \pm 1$.
- Markov Chain Monte Carlo - goal is to generate samples of some difficult distribution $p(x)$. $p(x)$ corresponds to a stationary distribution of the Markov chain.
∃ many applications of Markov chains

- Random walk on a integer point line, with $a_{ij} = 0$ unless $j = i \pm 1$.
- Markov Chain Monte Carlo - goal is to generate samples of some difficult distribution $p(x)$. $p(x)$ corresponds to a stationary distribution of the Markov chain.
- Stochastic Finite State Automata - any discrete time series could be modeled by this. E.g., Language ($n$-gram modeling in speech/natural language processing).
Markov Chains, some applications

∃ many applications of Markov chains

- Random walk on a integer point line, with $a_{ij} = 0$ unless $j = i \pm 1$.
- Markov Chain Monte Carlo - goal is to generate samples of some difficult distribution $p(x)$. $p(x)$ corresponds to a stationary distribution of the Markov chain.
- Stochastic Finite State Automata - any discrete time series could be modeled by this. E.g., Language ($n$-gram modeling in speech/natural language processing).
- Various Genomic & Proteomic Sequencing
∃ many applications of Markov chains

- Random walk on a integer point line, with \( a_{ij} = 0 \) unless \( j = i \pm 1 \).
- Markov Chain Monte Carlo - goal is to generate samples of some difficult distribution \( p(x) \). \( p(x) \) corresponds to a stationary distribution of the Markov chain.
- Stochastic Finite State Automata - any discrete time series could be modeled by this. E.g., Language (\( n \)-gram modeling in speech/natural language processing).
- Various Genomic & Proteomic Sequencing
- And so on.
Markov Chains, some applications

- ∃ many applications of Markov chains
  - Random walk on a integer point line, with $a_{ij} = 0$ unless $j = i \pm 1$.
  - Markov Chain Monte Carlo - goal is to generate samples of some difficult distribution $p(x)$. $p(x)$ corresponds to a stationary distribution of the Markov chain.
  - Stochastic Finite State Automata - any discrete time series could be modeled by this. E.g., Language ($n$-gram modeling in speech/natural language processing).
  - Various Genomic & Proteomic Sequencing
  - And so on.
  - Many applications listed in Grimmett & Stirzaker text.
Markov Chains, some applications

- ∃ many applications of Markov chains

  - Random walk on a integer point line, with $a_{ij} = 0$ unless $j = i \pm 1$.
  - Markov Chain Monte Carlo - goal is to generate samples of some difficult distribution $p(x)$. $p(x)$ corresponds to a stationary distribution of the Markov chain.
  - Stochastic Finite State Automata - any discrete time series could be modeled by this. E.g., Language ($n$-gram modeling in speech/natural language processing).
  - Various Genomic & Proteomic Sequencing
  - And so on.
  - Many applications listed in Grimmett & Stirzaker text.
  - We will revisit Markov chains repeatedly in this course, since many of the properties of DGMs have properties generalized from a Markov chain.
Hidden Markov Models (HMMs)

Hidden Markov Models (HMMs) are a ubiquitously used model in many fields, including speech recognition, natural language processing, bioinformatics, financial markets, and many time-series problems.
Hidden Markov Models (HMMs)

- Hidden Markov Models (HMMs) are a ubiquitously used model in many fields, including speech recognition, natural language processing, bioinformatics, financial markets, and many time-series problems.

- HMMs are powerful and diverse enough to be interesting, but still simple enough so that they are often computationally feasible (it is also a good example to get a strong feel for exact probabilistic inference).
Hidden Markov Models (HMMs)

• Hidden Markov Models (HMMs) are a ubiquitously used model in many fields, including speech recognition, natural language processing, bioinformatics, financial markets, and many time-series problems.

• HMMs are powerful and diverse enough to be interesting, but still simple enough so that they are often computationally feasible (it is also a good example to get a strong feel for exact probabilistic inference).

• HMMs can be described in many ways, including using one of a variety of different graphical models (e.g., Bayesian networks or Markov random fields, which means that they are decomposable).
Hidden Markov Models (HMMs)

Hidden Markov Models (HMMs) are a ubiquitously used model in many fields, including speech recognition, natural language processing, bioinformatics, financial markets, and many time-series problems.

HMMs are powerful and diverse enough to be interesting, but still simple enough so that they are often computationally feasible (it is also a good example to get a strong feel for exact probabilistic inference).

HMMs can be described in many ways, including using one of a variety of different graphical models (e.g., Bayesian networks or Markov random fields, which means that they are decomposable).

As we will see, HMMs are already triangulated (after moralization if necessary), no triangulation step for forming a junction tree.
Hidden Markov Models (HMMs)

- Hidden Markov Models (HMMs) are a ubiquitously used model in many fields, including speech recognition, natural language processing, bioinformatics, financial markets, and many time-series problems.
- HMMs are powerful and diverse enough to be interesting, but still simple enough so that they are often computationally feasible (it is also a good example to get a strong feel for exact probabilistic inference).
- HMMs can be described in many ways, including using one of a variety of different graphical models (e.g., Bayesian networks or Markov random fields, which means that they are decomposable).
- As we will see, HMMs are already triangulated (after moralization if necessary), no triangulation step for forming a junction tree.
- As we will see, since the clique sizes are size 2, HMMs are easy to deal with (\(O(TN^2)\) complexity), at least in theory.
A note on HMM notation

- It is common to use either $Q$ and $Q_{1:T}$ or $Y$ and $Y_{1:T}$ as the underlying Markov chain in an HMM.
A note on HMM notation

- It is common to use either $Q$ and $Q_{1:T}$ or $Y$ and $Y_{1:T}$ as the underlying Markov chain in an HMM.
- Speech recognition traditionally has used $Q$ and $Q_{1:T}$. 
A note on HMM notation

- It is common to use either $Q$ and $Q_{1:T}$ or $Y$ and $Y_{1:T}$ as the underlying Markov chain in an HMM.
- Speech recognition traditionally has used $Q$ and $Q_{1:T}$.
- Machine learning has typically used $Y$ and $Y_{1:T}$.
A note on HMM notation

- It is common to use either $Q$ and $Q_{1:T}$ or $Y$ and $Y_{1:T}$ as the underlying Markov chain in an HMM.
- Speech recognition traditionally has used $Q$ and $Q_{1:T}$.
- Machine learning has typically used $Y$ and $Y_{1:T}$.
- We will use both, but there should be no confusion about this (if there is, ask).
What if we don’t observe the Markov chain but it is hidden. Let $y = y_1:T$ be a length $T$ 1st-order time-homogeneous Markov chain (for the time being, all Markov chains will be such unless otherwise indicated)
What if we don’t observe the Markov chain but it is hidden. Let $y = y_{1:T}$ be a length $T$ 1st-order time-homogeneous Markov chain (for the time being, all Markov chains will be such unless otherwise indicated)

If a sequence $x = x_{1:T}$ is related by $x = f(y)$, with $y = y_{1:T}$ then we have that $x$ is a function of a Markov chain.
What if we don’t observe the Markov chain but it is hidden. Let $y = y_{1:T}$ be a length $T$ 1st-order time-homogeneous Markov chain (for the time being, all Markov chains will be such unless otherwise indicated).

If a sequence $x = x_{1:T}$ is related by $x = f(y)$, with $y = y_{1:T}$ then we have that $x$ is a function of a Markov chain.

If we have a set functions $\{f_1, f_2, \ldots, f_s\}$, and we have a random variable $S$ with $p(S = s)$, then if we define $X = f_S(Y)$, then $X$ is a probabilistic function of a Markov chain.
What if we don’t observe the Markov chain but it is hidden. Let $y = y_{1:T}$ be a length $T$ 1st-order time-homogeneous Markov chain (for the time being, all Markov chains will be such unless otherwise indicated).

If a sequence $x = x_{1:T}$ is related by $x = f(y)$, with $y = y_{1:T}$ then we have that $x$ is a function of a Markov chain.

If we have a set functions $\{f_1, f_2, \ldots, f_s\}$, and we have a random variable $S$ with $p(S = s)$, then if we define $X = f_S(Y)$, then $X$ is a probabilistic function of a Markov chain.

We observe $\bar{x}$ which presumably has been generated by choosing one $f_i$ randomly based on some random assignment to $Y$. $Y$ is therefore hidden.
What if we don’t observe the Markov chain but it is hidden. Let $y = y_{1:T}$ be a length $T$ 1st-order time-homogeneous Markov chain (for the time being, all Markov chains will be such unless otherwise indicated)

If a sequence $x = x_{1:T}$ is related by $x = f(y)$, with $y = y_{1:T}$ then we have that $x$ is a function of a Markov chain.

if we have a set functions $\{f_1, f_2, \ldots, f_s\}$, and we have a random variable $S$ with $p(S = s)$, then if we define $X = f_S(Y)$, then $X$ is a probabilistic function of a Markov chain.

We observe $\bar{x}$ which presumably has been generated by choosing one $f_i$ randomly based on some random assignment to $Y$. $Y$ is therefore hidden.

Hence, the name “Hidden” Markov model.
HMMs - random functions of a Markov chain

- HMMs are less general this this, as we don’t allow every $x_t$ to be a function of all of $y$. 

Suppose we have a set of $|D| \ast T$ functions $f_{i,t}$, and $T$ independent random variables $S_t$ then given Markov chain $Y$, we can define $X_t = f_{S_t}(Y_t)$. This at last corresponds our picture.
HMMs - random functions of a Markov chain

- HMMs are less general this this, as we don’t allow every $x_t$ to be a function of all of $y$.
- Suppose we have a set of $|D_S| \times T$ functions $f_{i,t}$, and $T$ independent random variables $S_t$ then given Markov chain $Y$, we can define $X_t = f_{S_t}(Y_t)$. This at last corresponds our picture.
HMMs - random functions of a Markov chain

- HMMs are less general than this, as we don’t allow every $x_t$ to be a function of all of $y$.
- Suppose we have a set of $|D_s| \cdot T$ functions $f_{i,t}$, and $T$ independent random variables $S_t$ then given Markov chain $Y$, we can define $X_t = f_{S_t}(Y_t)$. This at last corresponds our picture.

- Alternatively, can draw with explicit rep. of the variables $\{S_t : t\}$

Can think of $x_t$ as a deterministic function of both $y_t$ and $s_t$, and all of the randomness relating $x$ and $y$ is captured by the variables $s$. 
HMMs as Mixture Distributions

- We can also view HMMs as a sequence of mixture distributions.

A mixture distribution is simply:

\[ p(x) = \sum c \cdot p(x,c) = \sum c \cdot p(x|c) \cdot p(c) \]

Suppose we have a set of random variables \( X_1, \ldots, X_T \) each of which is an independent mixture distribution, with mixture variable \( Y_t \). We get:

\[
\begin{array}{c c c c c}
\text{x1} & \text{x2} & \text{x3} & \text{x4} & \text{x5} \\
\text{y1} & \text{y2} & \text{y3} & \text{y4} & \text{y5} \\
\text{xt} & \text{yt} & \text{T} \\
\end{array}
\]

Left: “plate” notation, where the set of random variables is repeated \( T \) times. Right: \( T = 5 \).

So in the collection of mixtures, we have that \( Y_i \perp \perp Y_j \) for all \( i \neq j \). The mixtures are exchangeable.
HMMs as Mixture Distributions

- We can also view HMMs as a sequence of mixture distributions.
- A mixture distribution is simply \( p(x) = \sum_c p(x, c) = \sum_c p(x|c)p(c) \)

\[
\begin{align*}
X &\rightarrow C
\end{align*}
\]
We can also view HMMs as a sequence of mixture distributions. A mixture distribution is simply

\[ p(x) = \sum_c p(x, c) = \sum_c p(x|c)p(c) \]

Suppose we have a set of random variables \( X_1, \ldots, X_T \) each of which is an individual independent mixture distribution, with mixture variable \( Y_t \). We get:
HMMs as Mixture Distributions

- We can also view HMMs as a sequence of mixture distributions.
- A mixture distribution is simply
  \[ p(x) = \sum_c p(x, c) = \sum_c p(x|c)p(c) \]

Suppose we have a set of random variables \( X_1, \ldots, X_T \) each of which is an individual independent mixture distribution, with mixture variable \( Y_t \). We get:

- Left: “plate” notation, where the set of random variables is repeated \( T \) times. Right: \( T = 5 \).
HMMs as Mixture Distributions

- We can also view HMMs as a sequence of mixture distributions.
- A mixture distribution is simply \( p(x) = \sum_c p(x, c) = \sum_c p(x|c)p(c) \)

\[
\begin{array}{c}
X \\
\end{array}
\longleftrightarrow
\begin{array}{c}
C \\
\end{array}
\]

- Suppose we have a set of random variables \( X_1, \ldots, X_T \) each of which is an individual independent mixture distribution, with mixture variable \( Y_t \). We get:

\[
\begin{array}{c}
y_t \\
\end{array}
\begin{array}{c}
x_t \\
\end{array}
\]

\[
\begin{array}{c}
y_1 \\
x_1 \\
\end{array}
\begin{array}{c}
y_2 \\
x_2 \\
\end{array}
\begin{array}{c}
y_3 \\
x_3 \\
\end{array}
\begin{array}{c}
y_4 \\
x_4 \\
\end{array}
\begin{array}{c}
y_5 \\
x_5 \\
\end{array}
\]

- Left: “plate” notation, where the set of random variables is repeated \( T \) times. Right: \( T = 5 \).
- So in the collection of mixtures, we have that \( Y_i \perp \perp Y_j \) for all \( i \neq j \). The mixtures are exchangeable.
HMMs as Mixture Distributions

- We might want to relax the assumption that the \( Y_i \)'s are independent.
We might want to relax the assumption that the $Y_i$'s are independent.

One of the simplest ways to add interaction between the mixture variables is to say that the $Y_{1:T}$ variables form a tree, and a simple tree is a chain, leading to an HMM.
We might want to relax the assumption that the $Y_i$’s are independent.

One of the simplest ways to add interaction between the mixture variables is to say that the $Y_{1:T}$ variables form a tree, and a simple tree is a chain, leading to an HMM.

However, in such case we still have a mixture distribution in that

$$p(x_{1:T}) = \sum_{y_{1:T}} p(x_{1:T}, y_{1:T}) \quad (3.1)$$

where there is naively an exponential number of mixture components.
HMM: As generative process

- Classic urn/ball example. We have $N$ urns, each urn has a set of different colored balls in them.
HMM: As generative process

- Classic urn/ball example. We have $N$ urns, each urn has a set of different colored balls in them.

- We choose urns according to a 1st-order Markov chain. Each time we choose an urn, we sample (with replacement) a ball from that urn.
HMM: As generative process

- Classic urn/ball example. We have $N$ urns, each urn has a set of different colored balls in them.
- We choose urns according to a 1st-order Markov chain. Each time we choose an urn, we sample (with replacement) a ball from that urn.
- The output is the sequence of ball colors, but not the sequence of urn choices.
HMM: As generative process

- Classic urn/ball example. We have $N$ urns, each urn has a set of different colored balls in them.

- We choose urns according to a 1st-order Markov chain. Each time we choose an urn, we sample (with replacement) a ball from that urn.

- The output is the sequence of ball colors, but not the sequence of urn choices.

- A goal might be: based on the sequence of ball colors, and knowing the corresponding distributions, deduce the most probable sequence of urns that could possibly have generated that ball sequence. This is the recognition problem.
Figure: A 4-state HMM viewed as balls in urns. Each HMM state corresponds to an urn and the directed edges between urns give the state transition matrix, where missing edges correspond to zeros in the matrix. In each state, there is a distribution on colored balls. For example, in state one, we have $\Pr(\text{color} = \text{yellow}|\text{state} = 1) = 4/7$ while $\Pr(\text{color} = \text{red}|\text{state} = 1) = 2/7$, $\Pr(\text{color} = \text{green}|\text{state} = 1) = 1/7$, and $\Pr(\text{color} = \text{blue}|\text{state} = 1) = 0$. That is, in each state (urn), the probability of drawing a color is equal to the number of balls of that color in that urn divided by the total number of balls.
**HMM: As a set of independence properties over $2T$ r.v.s**

**Definition 3.4.1**

**Hidden Markov Model** A hidden Markov model (HMM) is a collection of random variables consisting of a set of $T$ discrete scalar variables $Q_{1:T}$ and a set of $T$ other variables $X_{1:T}$ which may be either discrete or continuous (and either scalar- or vector-valued). These variables, collectively, possess the following conditional independence properties:

$$\{Q_{t:T}, X_{t:T}\} \perp \perp \{Q_{1:t-2}, X_{1:t-1}\}|Q_{t-1} \quad (3.2)$$

and

$$X_t \perp \perp \{Q_{-t}, X_{-t}\}|Q_t \quad (3.3)$$

for each $t \in 1 : T$. No other conditional independence properties are true in general, unless they follow from Equations 3.2 and 3.3. The length $T$ of these sequences is itself an integer-valued random variable having a complex distribution.
HMM: As a set of independence properties

- $Q_t$ takes values from finite set, so $Q_t \in D_Q$ where $D_Q$ is called the state space, cardinality $|D_Q|$.

- Equation 3.2 states that the future is conditionally independent of the past given the present. Therefore, $Q_t \perp \perp Q_{1:t-2}|Q_{t-1}$ which means the variables $Q_{1:T}$ form a discrete-time, discrete-valued, first-order Markov chain.

- Also, $Q_t \perp \{Q_{1:t-2}, X_{1:t-1}\}|Q_{t-1}$ so that $X_\tau$ is unable, given $Q_{t-1}$, to affect $Q_t$ for $\tau < t$.

- Does **not** imply, given $Q_{t-1}$, that $Q_t$ is unaffected by future variables.
HMM: As a set of independence properties

What the definition of this class of models doesn’t specify:

- does not limit the number of states $|D_Q|$ in the Markov chain, only that it is finite.
HMM: As a set of independence properties

What the definition of this class of models doesn’t specify:

- does not limit the number of states \( |D_Q| \) in the Markov chain, only that it is finite.
- **Does not require the observations** \( X_{1:T} \) **to be either discrete, continuous, scalar-, or vector-valued**, does not designate the implementation of the dependencies (e.g., general regression, probability table, neural network, etc.),
HMM: As a set of independence properties

What the definition of this class of models doesn’t specify:

- does not limit the number of states $|D_Q|$ in the Markov chain, only that it is finite.
- Does not require the observations $X_{1:T}$ to be either discrete, continuous, scalar-, or vector-valued, does not designate the implementation of the dependencies (e.g., general regression, probability table, neural network, etc.),
- does not determine the model families for each of the variables (e.g., Gaussian, Laplace, etc.).
HMM: As a set of independence properties

What the definition of this class of models doesn’t specify:

- Does not limit the number of states $|D_Q|$ in the Markov chain, only that it is finite.
- Does not require the observations $X_{1:T}$ to be either discrete, continuous, scalar-, or vector-valued, does not designate the implementation of the dependencies (e.g., general regression, probability table, neural network, etc.),
- Does not determine the model families for each of the variables (e.g., Gaussian, Laplace, etc.),
- Does not force the underlying Markov chain to be time-homogeneous,
HMM: As a set of independence properties

What the definition of this class of models doesn’t specify:

- does not limit the number of states $|D_Q|$ in the Markov chain, only that it is finite.
- Does not require the observations $X_{1:T}$ to be either discrete, continuous, scalar-, or vector- valued, does not designate the implementation of the dependencies (e.g., general regression, probability table, neural network, etc.),
- does not determine the model families for each of the variables (e.g., Gaussian, Laplace, etc.),
- does not force the underlying Markov chain to be time-homogeneous,
- does not fix the parameters or any tying mechanism
HMM: As a set of independence properties

What the definition of this class of models doesn’t specify:

- does not limit the number of states $|D_Q|$ in the Markov chain, only that it is finite.
- Does not require the observations $X_{1:T}$ to be either discrete, continuous, scalar-, or vector-valued, does not designate the implementation of the dependencies (e.g., general regression, probability table, neural network, etc.),
- does not determine the model families for each of the variables (e.g., Gaussian, Laplace, etc.),
- does not force the underlying Markov chain to be time-homogeneous,
- does not fix the parameters or any tying mechanism
- HMM makes *no* marginal independence assumptions — nothing in an HMM is independent of anything else (no statements of form $A \perp \!\!\!\perp B$). only conditional independence statements exist.
Using the conditional independence statements mentioned above, we can derive the following factorization:

\[
p(x_{1:T}, q_{1:T}) = p(x_T, q_T | x_{1:T-1}, q_{1:T-1})p(x_{1:T-1}, q_{1:T-1})
\]
\[
= p(x_T | q_T, x_{1:T-1}, q_{1:T-1})p(q_T | x_{1:T-1}, q_{1:T-1})
\]
\[
p(x_{1:T-1}, q_{1:T-1})
\]
\[
= p(x_T | q_T)p(q_T | q_{T-1})p(x_{1:T-1}, q_{1:T-1})
\]
\[
= \ldots
\]
\[
= p(q_1) \prod_{t=2}^{T} p(q_t | q_{t-1}) \prod_{t=1}^{T} p(x_t | q_t)
\]

This last equation is the classical factorization expression for an HMM joint distribution over \( x_{1:T}, q_{1:T} \).
HMMs are a sequential model simultaneously over both sequences $X_{1:T}$ and $Q_{1:T}$, i.e., $p(x_{1:T}, q_{1:T})$.

HMMs represent a joint distribution, i.e., $p(x_{1:T}, q_{1:T})$, not a conditional distribution.

Just like any graphical model, that represents a joint distribution over a set of random variables.
HMM parameters

- Parameters of HMM, depend on nature of underlying Markov chain.
- If time-homogeneous, we have an initial state distribution (typically \( \pi \)) with \( p(Q_1 = i) = \pi_i \), and a state transition matrix \( A \).
- We also have the set of observation distributions \( b_j(x) = p(X_t = x | Q_t = j) \) in the time-homogeneous case. In time inhomogeneous case, we might have \( b_{t,j}(x) \).
- Sampling from an HMM means: 1) first randomly choose an assignment to \( Q_{1:T} \) and then 2) randomly choose an assignment to \( X_{1:T} \).
- Each new \( X \) sample requires a new \( Q \) sample.
An HMM is a tree (in the GM sense) and so we know that computing inference (over cliques) has cost $O(r^2)$. Since there are $T$ such cliques, overall cost should be $O(Tr^2)$.

Also, we view $X_{1:T}$ as the stochastic process with an underlying generative hidden chain $Q_{1:T}$. Thus, we might want to compute

$$p(\bar{x}_{1:T}) = \sum_{q_{1:T}} p(\bar{x}_{1:T}, q_{1:T})$$  \hspace{1cm} (3.10)$$

again, naively, an exponential computation.

This is the probability of evidence computation, and again due to the query and the fact that it is a tree, it is relatively easy, at least from the perspective of clique size (there are many difficulties when $r$ gets extremely large, as we’ll see).
To compute $p(x_{1:T})$, we start out with $p(x_{1:t})$

$$p(x_1:t, q_t, q_{t-1})$$

$$= p(x_1:t-1, q_{t-1}, x_t, q_t)$$

$$(A)$$

$$= p(x_t, q_t | x_1:t-1, q_{t-1})p(x_1:t-1, q_{t-1})$$

$$= p(x_t | q_t, x_1:t-1, q_{t-1})p(q_t | x_1:t-1, q_{t-1})p(x_1:t-1, q_{t-1})$$

$$(B)$$

$$= p(x_t | q_t)p(q_t | q_{t-1})p(x_1:t-1, q_{t-1})$$

where (A) follows from the chain rule of probability, and (B) follows since $X_t \perp \perp \{X_{1:t-1}, Q_{1:t-1}\} | Q_t$ and $Q_t \perp \perp \{X_{1:t-1}, Q_{1:t-2}\} | Q_{t-1}$
HMM Forward Recursion

This yields,

\[
p(x_{1:t}, q_t) = \sum_{q_{t-1}} p(x_{1:t}, q_t, q_{t-1})
\]

\[
= \sum_{q_{t-1}} p(x_t | q_t) p(q_t | q_{t-1}) p(x_{1:t-1}, q_{t-1})
\]

(3.11)

(3.12)
HMM Forward Recursion

- This yields,

\[ p(x_{1:t}, q_t) = \sum_{q_{t-1}} p(x_{1:t}, q_t, q_{t-1}) \]  
\[ = \sum_{q_{t-1}} p(x_t | q_t) p(q_t | q_{t-1}) p(x_{1:t-1}, q_{t-1}) \]  

(3.11) (3.12)

- If the following quantity is defined \( \alpha_q(t) \triangleq p(x_{1:t}, Q_t = q) \), then the preceding equations imply that

\[ \alpha_q(t) = p(x_t | Q_t = q) \sum_r p(Q_t = q | Q_{t-1} = r) \alpha_r(t-1) \]  

(3.13)
HMM Forward Recursion

- This yields,

\[ p(x_{1:t}, q_t) = \sum_{q_{t-1}} p(x_{1:t}, q_t, q_{t-1}) = \sum_{q_{t-1}} p(x_t | q_t) p(q_t | q_{t-1}) p(x_{1:t-1}, q_{t-1}) \]  

(3.11)

(3.12)

- If the following quantity is defined \( \alpha_q(t) \Delta p(x_{1:t}, Q_t = q) \), then the preceding equations imply that

\[ \alpha_q(t) = p(x_t | Q_t = q) \sum_r p(Q_t = q | Q_{t-1} = r) \alpha_r(t - 1) \]  

(3.13)

- This is called the \( \alpha \) (or forward) recursion for HMMs. It is also identical to message passing from left-to-right in the LBP algorithm.
Thus, \( p(x_{1:T}) = \sum_{q} \alpha_q(T) \), and the entire computation requires only \( O(|D_Q|^2 T) \) operations.
HMM Forward Recursion

- Thus, $p(x_{1:T}) = \sum_q \alpha_q(T)$, and the entire computation requires only $O(|D_Q|^2T)$ operations.

- To derive this recursion, it was necessary to use only the fact that $X_t$ was independent of its past given $Q_t$ (call this Assumption I) — in an HMM, $X_t$ is also independent of the future given $Q_t$, but this was not yet used (call this Assumption II).
Thus, \( p(x_{1:T}) = \sum_q \alpha_q(T) \), and the entire computation requires only \( O(|D_Q|^2T) \) operations.

To derive this recursion, it was necessary to use only the fact that \( X_t \) was independent of its past given \( Q_t \) (call this Assumption I) — in an HMM, \( X_t \) is also independent of the future given \( Q_t \), but this was not yet used (call this Assumption II).

Exercise: Are there models where \( X_t \) is independent of its past but not its future given \( Q_t \) where such a forward recursion can be defined?
This later assumption (Assumption II), however, is obligatory for the beta or backward recursion in HMMs as we will now see.

\[
p(x_{t+1:T}|q_t) = \sum_{q_{t+1}} p(q_{t+1}, x_{t+1}, x_{t+2:T}|q_t)
\]

\[
(A) \sum_{q_{t+1}} p(x_{t+2:T}|q_{t+1}, x_{t+1}, q_t)p(x_{t+1}|q_{t+1}, q_t)p(q_{t+1}|q_t)
\]

\[
(B) \sum_{q_{t+1}} p(x_{t+2:T}|q_{t+1}) p(x_{t+1}|q_{t+1}) p(q_{t+1}|q_t)
\]

where (A) follows from the chain rule probability, and (B) follows since \( X_{t+2:T} \perp \perp \{X_{t+1}, Q_t\}|Q_{t+1} \) and \( X_{t+1} \perp \perp Q_t|Q_{t+1} \).
HMM backwards recursion

- Using the definition $\beta_q(t) \triangleq p(x_{t+1:T} | Q_t = q)$, the above equations imply the beta-recursion

$$\beta_q(t) = \sum_r \beta_r(t+1) p(x_{t+1} | Q_{t+1} = r) p(Q_{t+1} = r | Q_t = q), \quad (3.14)$$

and another expression for the full probability

$$p(x_1:T) = \sum_q \beta_q(1) p(q) p(x_1 | q). \quad (3.15)$$
$p(x_{1:t})$ via $\alpha$ and $\beta$

- Recall, goal is to compute $p(x_{1:T})$, the probability of “evidence”, efficiently.
$p(x_{1:t})$ via $\alpha$ and $\beta$

- Recall, goal is to compute $p(x_{1:T})$, the probability of “evidence”, efficiently.
- This probability may be computed using a combination of the alpha and beta values at any $t$ since

$$p(x_{1:T}) = \sum_{q_t} p(q_t, x_{1:t}, x_{t+1:T}) = \sum_{q_t} p(x_{t+1:T} | q_t, x_{1:t}) p(q_t, x_{1:t})$$

$$(A) \quad \Rightarrow \quad \sum_{q_t} p(x_{t+1:T} | q_t) p(q_t, x_{1:t}) = \sum_{q_t} \beta_{q_t}(t) \alpha_{q_t}(t)$$

where (A) follows since $X_{t+1:T} \perp \! \! \! \! \perp X_{1:t} | Q_t$ in an HMM.
Recall, goal is to compute \( p(x_{1:T}) \), the probability of "evidence", efficiently.

This probability may be computed using a combination of the alpha and beta values at any \( t \) since

\[
p(x_{1:T}) = \sum_{q_t} p(q_t, x_{1:t}, x_{t+1:T}) = \sum_{q_t} p(x_{t+1:T} | q_t, x_{1:t}) p(q_t, x_{1:t})
\]

\[
\overset{(A)}{=} \sum_{q_t} p(x_{t+1:T} | q_t) p(q_t, x_{1:t}) = \sum_{q_t} \beta_{q_t}(t) \alpha_{q_t}(t)
\]

where (A) follows since \( X_{t+1:T} \perp X_{1:t} | Q_t \) in an HMM.

Note, above is true for any \( t \).
For any \( p \in \mathcal{F}(\text{HMM, } R) \), from the directed local Markov property, we can immediately write down the joint as

\[
p(x_{1:T}, q_{1:T}) = p(q_1)p(x_1|q_1) \prod_{t=2}^{T} p(x_t|q_t)p(q_t|q_{t-1})
\]

(3.16)

\[
= \prod_{t} p(x_t|q_t)p(q_t|q_{t-1})
\]

(3.17)

Hence, the factorization properties of an HMM are immediate consequences of the meaning of the HMM graph (perhaps just seeing the graph is less work).
The quantities we typically wish to compute for an HMM include:

- Compute $p(q_t|x_{1:t})$, or the filtering problem.
- Compute $p(q_t|x_{1:s})$, with $t > s$, or the prediction problem.
- Compute $p(q_t|x_{1:u})$, with $t < u$, or the smoothing problem.
- Above three named from linear systems literature in EE (e.g., Kalman filters).

Note: above includes $p(q_t|x_{1:T})$.

Also needed query is $p(q_t, q_{t+1}|x_r:s)$ (often $r = 1$ and $s = T$).

In all above cases, we need to sum out hidden variables from joint distributions. E.g., $p(q_t|x_{1:T}) = p(q_t, x_{1:T})/p(x_{1:T})$, so also need $p(x_{1:T})$. I.e., we compute both the numerator and denominator in each of the above queries.

Next few slides show how this relates to clique potentials in the HMM graph.
Recall parameter names, time-homogeneous case.

1. \( P(Q_t = i|Q_{t-1} = j) = a_{ij} \) or \([A]_{ij}\) is a first-order time-homogeneous transition matrix.

\( P(Q_1 = i) = \pi_i \) is the initial state distribution.

\( P(X_t = x|Q_t = i) = b_{ij}(x) \) is the observation distribution for the current state being in configuration \( i \).

Notice that there are a fixed number of parameters regardless of the length \( T \). In other words, parameters are shared across all time. This is a property of all dynamic graphical models.
HMM - parameter names, homogeneous case

Recall parameter names, time-homogeneous case.

1. \( P(Q_t = i | Q_{t-1} = j) = a_{ij} \) or \([A]_{ij}\) is a first-order time-homogeneous transition matrix.

2. \( P(Q_1 = i) = \pi_i \) is the initial state distribution.
Recall parameter names, time-homogeneous case.

1. $P(Q_t = i|Q_{t-1} = j) = a_{ij}$ or $[A]_{ij}$ is a first-order time-homogeneous transition matrix.

2. $P(Q_1 = i) = \pi_i$ is the initial state distribution.

3. $P(X_t = x|Q_t = i) = b_i(x)$ is the observation distribution for the current state being in configuration $i$. 

Notice that there are a fixed number of parameters regardless of the length $T$. In other words, parameters are shared across all time. This is a property of all dynamic graphical models.
Recall parameter names, time-homogeneous case.

1. \( P(Q_t = i|Q_{t-1} = j) = a_{ij} \) or \([A]_{ij}\) is a first-order time-homogeneous transition matrix.

2. \( P(Q_1 = i) = \pi_i \) is the initial state distribution.

3. \( P(X_t = x|Q_t = i) = b_i(x) \) is the observation distribution for the current state being in configuration \( i \).

Notice that there are a fixed number of parameters regardless of the length \( T \). In other words, parameters are shared across all time. This is a property of all dynamic graphical models.
Recall parameter names, time-homogeneous case.

1. \( P(Q_t = i | Q_{t-1} = j) = a_{ij} \) or \([A]_{ij}\) is a first-order time-homogeneous transition matrix.
2. \( P(Q_1 = i) = \pi_i \) is the initial state distribution.
3. \( P(X_t = x | Q_t = i) = b_i(x) \) is the observation distribution for the current state being in configuration \( i \).

Notice that there are a fixed number of parameters regardless of the length \( T \). In other words, parameters are \textbf{shared} across all time. This is a property of all dynamic graphical models.

What probabilistic queries would we need to learn these parameters?
HMM - learning with EM

- To decide which queries to compute, should know which ones we want. If learning HMM parameters with EM, what queries do we need?
To decide which queries to compute, should know which ones we want. If learning HMM parameters with EM, what queries do we need?

- $X_{1:T} = \bar{x}_{1:T}$ observed, $Q_{1:T}$ hidden variables.
HMM - learning with EM

To decide which queries to compute, should know which ones we want. If learning HMM parameters with EM, what queries do we need?

- $X_{1:T} = \bar{x}_{1:T}$ observed, $Q_{1:T}$ hidden variables.

- For convenience, define $\lambda$ as all parameters to be learnt, and $\lambda^p$ are the previous iteration’s parameters.
The EM algorithm then repeatedly optimizes the following objective:

\[ f(\lambda) \]
HMM - learning with EM

- The EM algorithm then repeatedly optimizes the following objective:

\[
f(\lambda) = Q(\lambda, \lambda^p)
\]
HMM - learning with EM

The EM algorithm then repeatedly optimizes the following objective:

\[ f(\lambda) = Q(\lambda, \lambda^p) = E_p(x_{1:T}, q_{1:T} | \lambda^p) \left[ \log p(x_{1:T}, q_{1:T} | \lambda) \right] \]  

(3.18)
The EM algorithm then repeatedly optimizes the following objective:

\[
f(\lambda) = Q(\lambda, \lambda^p) = E_p(x_{1:T}, q_{1:T}|\lambda^p)[\log p(x_{1:T}, q_{1:T}|\lambda)]
\]

\[
= E_p[\log \prod_t p(q_t|q_{t-1}, \lambda)p(x_t|q_t, \lambda)]
\]

So this means that for EM learning, we need for all \( t \), the queries

\[ p(Q_t = i|x_{1:T}) \] and \[ p(Q_t = j, Q_{t-1} = i|x_{1:T}) \]
in an HMM. Note again that these are clique posteriors.
HMM - learning with EM

The EM algorithm then repeatedly optimizes the following objective:

\[
f(\lambda) = Q(\lambda, \lambda^p) = E_p(x_{1:T}, q_{1:T}|\lambda^p)[\log p(x_{1:T}, q_{1:T}|\lambda)] = E_p[\log \prod_t p(q_t|q_{t-1}, \lambda)p(x_t|q_t, \lambda)]
\]  

\[
= E_p[\sum_t \log p(q_t|q_{t-1}, \lambda) + \sum_t \log p(x_t|q_t, \lambda)]
\]

(3.18)  

(3.19)  

(3.20)  

(3.21)
HMM - learning with EM

The EM algorithm then repeatedly optimizes the following objective:

\[ f(\lambda) = Q(\lambda, \lambda^p) = E_p(x_{1:T}, q_{1:T} | \lambda^p) \left[ \log p(x_{1:T}, q_{1:T} | \lambda) \right] \]  

\[ = E_p \left[ \log \prod_t p(q_t | q_{t-1}, \lambda) p(x_t | q_t, \lambda) \right] \]  

\[ = E_p \left[ \sum_t \log p(q_t | q_{t-1}, \lambda) + \sum_t \log p(x_t | q_t, \lambda) \right] \]  

\[ = \sum_t \sum_{ij} p(Q_t = j, Q_{t-1} = i | x_{1:T}, \lambda^p) \log p(Q_t = j | Q_{t-1} = i, \lambda) \]  

Equation (3.18)

Equation (3.19)

Equation (3.20)

Equation (3.21)
The EM algorithm then repeatedly optimizes the following objective:

\[ f(\lambda) = Q(\lambda, \lambda^p) = E_p(x_{1:T}, q_{1:T}|\lambda^p)[\log p(x_{1:T}, q_{1:T}|\lambda)] \tag{3.18} \]

\[ = E_p[\log \prod_t p(q_t|q_{t-1}, \lambda)p(x_t|q_t, \lambda)] \tag{3.19} \]

\[ = E_p[\sum_t \log p(q_t|q_{t-1}, \lambda) + \sum_t \log p(x_t|q_t, \lambda)] \tag{3.20} \]

\[ = \sum_t \sum_{ij} p(Q_t = j, Q_{t-1} = i|x_{1:T}, \lambda^p) \log p(Q_t = j|Q_{t-1} = i, \lambda) \]

\[ + \sum_t \sum_i p(Q_t = i|x_{1:T}, \lambda^p) \log p(x_t|Q_t = i, \lambda) \tag{3.21} \]
The EM algorithm then repeatedly optimizes the following objective:

\[ f(\lambda) = Q(\lambda, \lambda^p) = E_p(x_{1:T}, q_{1:T}|\lambda^p)[ \log p(x_{1:T}, q_{1:T}|\lambda) ] \]  \hspace{1cm} (3.18)

\[ = E_p[ \log \prod_t p(q_t|q_{t-1}, \lambda)p(x_t|q_t, \lambda) ] \]  \hspace{1cm} (3.19)

\[ = E_p[ \sum_t \log p(q_t|q_{t-1}, \lambda) + \sum_t \log p(x_t|q_t, \lambda) ] \]  \hspace{1cm} (3.20)

\[ = \sum_t \sum_{ij} p(Q_t = j, Q_{t-1} = i|x_{1:T}, \lambda^p) \log p(Q_t = j|Q_{t-1} = i, \lambda) \]

\[ + \sum_t \sum_i p(Q_t = i|x_{1:T}, \lambda^p) \log p(x_t|Q_t = i, \lambda) \]  \hspace{1cm} (3.21)

- So this means that for EM learning, we need **for all** \( t \), the queries \( p(Q_t = i|x_{1:T}) \) and \( p(Q_t = j, Q_{t-1} = i|x_{1:T}) \) in an HMM. Note again that these are clique posteriors.
EM isn’t the only way to learn parameters.

Suppose we wanted to use a gradient descent like algorithm on $f(\lambda) = \log p(x_{1:T}|\lambda)$, as in

$$
\frac{\partial}{\partial \lambda} f(\lambda) = \frac{\partial}{\partial \lambda} \log p(x_{1:T}|\lambda) = \frac{\partial}{\partial \lambda} \log \sum_{q_{1:T}} p(x_{1:T}, q_{1:T}|\lambda) \tag{3.22}
$$

$$
= \frac{\frac{\partial}{\partial \lambda} \sum_{q_{1:T}} p(x_{1:T}, q_{1:T}|\lambda)}{\sum_{q_{1:T}} p(x_{1:T}, q_{1:T}|\lambda)} = \frac{\frac{\partial}{\partial \lambda} \sum_{q_{1:T}} p(x_{1:T}, q_{1:T}|\lambda)}{p(x_{1:T}|\lambda)} \tag{3.23}
$$
HMM - learning with gradient descent (cont. II)

- Say we’re interested in $\partial / \partial a_{ij}$. Lets expand the numerator above:

\[
\text{numerator} = \frac{\partial}{\partial a_{ij}} \sum_{q_1:T} p(x_{1:T}, q_{1:T} | \lambda) = \frac{\partial}{\partial a_{ij}} \sum_{q_1:T} \prod_{t} p(x_t | q_t) p(q_t | q_{t-1})
\]  

(3.24)

- Define $T_{ij}(q_{1:T}) \triangleq \{ t : q_{t-1} = i, q_t = j \}$ in the following:

\[
\text{numerator} = \frac{\partial}{\partial a_{ij}} \sum_{q_1:T} \prod_{t \in T_{ij}(q_{1:T})} p(x_t | q_t) \prod_{t \in T_{ij}(q_{1:T})} a_{ij} \prod_{t \notin T_{ij}(q_{1:T})} p(q_t | q_{t-1})
\]  

(3.25)
HMM - learning with gradient descent

We get

\[
\text{num} = \sum_{q_1:T} \prod_{t} p(x_t|q_t) \frac{\partial}{\partial a_{ij}} |\mathcal{T}_{ij}(q_1:T)| \prod_{t \notin \mathcal{T}_{ij}(q_1:T)} p(q_t|q_{t-1})
\]

\[
= \sum_{q_1:T} \prod_{t} p(x_t|q_t) |\mathcal{T}_{ij}(q_1:T)| a_{ij}^{-1} \prod_{t \notin \mathcal{T}_{ij}(q_1:T)} p(q_t|q_{t-1})
\]

\[
= \sum_{q_1:T} \prod_{t} p(x_t|q_t)p(q_t|q_{t-1}) \frac{|\mathcal{T}_{ij}(q_1:T)|}{a_{ij}} = \sum_{q_1:T} p(x_1:T, q_1:T) \frac{|\mathcal{T}_{ij}(q_1:T)|}{a_{ij}}
\]

\[
= \frac{1}{a_{ij}} \sum_{q_1:T} p(x_1:T, q_1:T) \sum_{t} 1\{q_{t-1} = i, q_t = j\}
\]

\[
= \frac{1}{a_{ij}} \sum_{t} \sum_{q_1:T} p(x_1:T, q_1:T) 1\{q_{t-1} = i, q_t = j\}
\]

\[
= \frac{1}{a_{ij}} \sum_{t} p(x_1:T, q_{t-1} = i, q_t = j)
\]
HMM - learning with gradient descent

\[
\frac{\partial}{\partial \lambda} f(\lambda) = \frac{\partial}{\partial \lambda} \sum_{q_1:T} \frac{p(x_1:T, q_1:T | \lambda)}{p(x_1:T | \lambda)} = \frac{1}{a_{ij}} \sum_t p(x_1:T, q_{t-1} = i, q_t = j) \frac{p(x_1:T | \lambda)}{p(x_1:T | \lambda)}
\]

This means that, like in EM, for gradient descent learning, we also need for all \( t \) the queries \( p(Q_t = j, Q_{t-1} = i | x_1:T) \) from the HMM. A similar analysis shows that we also need \( \forall t \) \( p(Q_t = i | x_1:T) \). These are also needed when performing discriminative training. So clique posteriors are fundamental, we must have a procedure that produces them quickly.
HMMs and elimination - forward recursion

- HMM forward recursion is just the elimination algorithm on the graph.
- Choose elimination order:
  \[ X_1, X_2, Q_1, X_3, Q_2, X_4, Q_3, X_5, \ldots, X_T, Q_{T-1}, Q_T. \]
- Evidence from delta functions, i.e., \( x_1 = \bar{x}_1 \Rightarrow \delta(x_1, \bar{x}_1), \)
  \( x_2 = \bar{x}_2 \Rightarrow \delta(x_2, \bar{x}_2), \) and so on.
- We get:

\[
\begin{align*}
\sum_{x_4} \sum_{q_2} \sum_{x_3} \sum_{q_1} \sum_{x_2} \sum_{x_1} \prod_{t=1}^{T} p(x_t | q_t)p(q_t | q_{t-1}) \delta(x_t, \bar{x}_t) \\
= \sum_{x_3} \sum_{q_1} \sum_{x_2} \left( \prod_{t=2}^{T} p(x_t | q_t)p(q_t | q_{t-1}) \delta(x_t, \bar{x}_t) \right) \sum_{x_1} p(x_1 | q_1)\delta(x_1, \bar{x}_1)p(q_1) \\
\end{align*}
\]

\[ p(\bar{x}_1 | q_1)p(q_1) \triangleq \alpha_1(q_1) \]
HMMs and elimination - forward recursion

\[
\ldots \sum_{x_3} \sum_{q_1} \left( \prod_{t=3}^{T} p(x_t|q_t)p(q_t|q_{t-1}) \delta(x_t, \bar{x}_t) \right) \sum_{x_2} p(x_2|q_2)p(q_2|q_1) \delta(x_2, \bar{x}_2) \alpha(q_1)
\]

\[
= \ldots \sum_{x_4} \sum_{q_2} \sum_{x_3} \left( \prod_{t=3}^{T} p(x_t|q_t)p(q_t|q_{t-1}) \delta(x_t, \bar{x}_t) \right) \sum_{q_1} p(q_2|q_1)p(\bar{x}_2|q_2) \alpha(q_1)
\]

\[
= \ldots
\]

\[
= \ldots \sum_{x_{r+2}} \sum_{q_r} \sum_{x_{r+1}} \left( \prod_{t=r+1}^{T} p(x_t|q_t)p(q_t|q_{t-1}) \delta(x_t, \bar{x}_t) \right) \sum_{q_{r-1}} p(q_r|q_{r-1})p(\bar{x}_r|q_r) \alpha(q_{r-1})
\]

\[
\alpha_\text{-recursion becomes}
\]

\[
\alpha_{t+1}(j) = \sum_i \alpha_t(i)p(Q_{t+1} = j|Q_t = i)p(x_{t+1}|Q_{t+1} = j)
\]  \hspace{1cm} (3.28)

\[
\alpha_1(j) = p(Q_1 = j)p(\bar{x}_1|Q_1 = j)
\]  \hspace{1cm} (3.29)
HMMs and elimination - forward recursion

- From the last line of the elimination (when we sum out $q_{r-1}$, we see $\alpha$-recursion, which is:

$$\alpha_{t+1}(j) = \sum_{i} \alpha_{t}(i)p(Q_{t+1} = j|Q_{t} = i)p(x_{t+1}|Q_{t+1} = j)$$  \hspace{1cm} (3.30)$$

and

$$\alpha_{1}(j) = p(Q_{1} = j)p(\bar{x}_{1}|Q_{1} = j)$$  \hspace{1cm} (3.31)$$
HMMs and elimination - forward recursion

- From the last line of the elimination (when we sum out \( q_{r-1} \), we see \( \alpha \)-recursion, which is:

\[
\alpha_{t+1}(j) = \sum_i \alpha_t(i)p(Q_{t+1} = j|Q_t = i)p(x_{t+1}|Q_{t+1} = j)
\]  \( (3.30) \)

and

\[
\alpha_1(j) = p(Q_1 = j)p(\bar{x}_1|Q_1 = j)
\]  \( (3.31) \)

- We have that

\[
\alpha_1(Q_1 = j) = p(Q_1 = j)p(\bar{x}_1|Q_1 = j) = p(\bar{x}_1, Q_1 = j), \text{ and}
\]

\[
\alpha_1(q_1) = p(\bar{x}_1, q_1).
\]
HMMs and elimination - forward recursion

- From the last line of the elimination (when we sum out $q_{r-1}$, we see $\alpha$-recursion, which is:

$$\alpha_{t+1}(j) = \sum_i \alpha_t(i)p(Q_{t+1} = j|Q_t = i)p(x_{t+1}|Q_{t+1} = j)$$ (3.30)

and

$$\alpha_1(j) = p(Q_1 = j)p(\bar{x}_1|Q_1 = j)$$ (3.31)

- We have that

$$\alpha_1(Q_1 = j) = p(Q_1 = j)p(\bar{x}_1|Q_1 = j) = p(\bar{x}_1, Q_1 = j), \text{ and}$$

$$\alpha_1(q_1) = p(\bar{x}_1, q_1).$$

- Also, $\alpha_2(q_2) = \sum_{q_1} p(q_2|q_1)p(\bar{x}_2|q_2)\alpha(q_1) = \sum_{q_1} p(q_1)p(\bar{x}_1, q_1) = \sum_{q_1} p(q_1, q_2, \bar{x}_1, \bar{x}_2) \equiv p(\bar{x}_1, \bar{x}_2, q_2)$

In general, the forward ($\alpha$) recursion has meaning

$$\alpha_t(j) = p(x_{1:t}, Q_t = j)$$
HMMs and elimination - forward recursion

- From the last line of the elimination (when we sum out $q_{r-1}$, we see $\alpha$-recursion, which is:

$$
\alpha_{t+1}(j) = \sum_i \alpha_t(i)p(Q_{t+1} = j|Q_t = i)p(x_{t+1}|Q_{t+1} = j)
$$

and

$$
\alpha_1(j) = p(Q_1 = j)p(\bar{x}_1|Q_1 = j)
$$

- We have that

$$
\alpha_1(Q_1 = j) = p(Q_1 = j)p(\bar{x}_1|Q_1 = j) = p(\bar{x}_1, Q_1 = j), \text{ and}
$$

$$
\alpha_1(q_1) = p(\bar{x}_1, q_1).
$$

- Also, $\alpha_2(q_2) = \sum_{q_1} p(q_2|q_1)p(\bar{x}_2|q_2)\alpha(q_1) = \sum_{q_1} p(q_2|q_1)p(\bar{x}_2|q_2)p(\bar{x}_1, q_1) = \sum_{q_1} p(q_1, q_2, \bar{x}_1, \bar{x}_2) = p(\bar{x}_1, \bar{x}_2, q_2)$

- In general, the forward ($\alpha$) recursion has meaning

$$
\alpha_t(j) = p(x_{1:t}, Q_t = j)
$$

- So $\alpha$ (forward) recursion is just an instance of the elimination algorithm run on the GM for the HMM graph.
Next, consider elimination order $X_T, Q_T, X_{T-1}, Q_{T-1}, \ldots$

$$\ldots \sum_{x_{T-2}} \sum_{q_{T-1}} \sum_{x_{T-1}} \sum_{q_T} \sum_{x_T} \prod_{t=1}^{T} p(x_t|q_t)p(q_t|q_{t-1})\delta(x_t, \bar{x}_t)$$

$$= \ldots \sum_{x_{T-2}} \sum_{q_{T}} \left( \prod_{t=1}^{T-1} p(x_t|q_t)p(q_t|q_{t-1})\delta(x_t, \bar{x}_t) \right) \sum_{x_T} p(x_T|q_T)p(q_T|q_{T-1})\delta(x_T, \bar{x}_T)$$

$$= \ldots \sum_{x_{T-2}} \sum_{q_{T}} \left( \prod_{t=1}^{T-1} p(x_t|q_t)p(q_t|q_{t-1})\delta(x_t, \bar{x}_t) \right) p(\bar{x}_T|q_T)p(q_T|q_{T-1})$$
HMMs and elimination - backward recursion

\[ \sum_{x_{T-2}} \sum_{q_T} \left( \prod_{t=1}^{T-1} p(x_t|q_t) p(q_t|q_{t-1}) \delta(x_t, \bar{x}_t) \right) p(\bar{x}_T|q_T) p(q_T|q_{T-1}) \frac{1}{\beta_T(q_T)} \]

\[ \sum_{q_{T-1}} \sum_{x_{T-2}} \sum_{q_T} \left( \prod_{t=1}^{T-1} p(x_t|q_t) p(q_t|q_{t-1}) \delta(x_t, \bar{x}_t) \right) \sum_{q_T} p(\bar{x}_T|q_T) p(q_T|q_{T-1}) \beta_T(q_T) \]

This corresponds to the \( \beta \) recursion

\[ \beta_t(q_t) = \sum_{q_{t+1}} \beta_{t+1}(q_{t+1}) p(q_{t+1}|q_t) p(x_{t+1}|q_{t+1}) \quad (3.32) \]

\[ \beta_T(q_T) = 1 \quad (3.33) \]
HMMs and elimination - backward recursion, meaning

\[ \beta_T(j) = 1 \]  

(3.34)

\[ \beta_{T-1}(i) = \sum_j p(\bar{x}_T|Q_T = j)p(Q_T = j|Q_{T-1} = i)\beta_T(j) \]  

(3.35)

\[ \sum_j p(\bar{x}_T, Q_T = j|Q_{T-1} = i) = p(\bar{x}_T|Q_{T-1} = i) \]  

(3.36)

In general,

\[ \beta_t(i) = p(x_{t+1:T}|Q_t = i) \]  

(3.37)
HMMs and elimination - $p(x_{1:t})$

We can get $p(x_{1:t})$ in many different ways:

\[
p(x_{1:T}) = \sum_{q_t} p(x_{1:T}, q_t) \\
= \sum_{q_t} p(x_{1:t}, x_{t+1:T}, q_t) \\
= \sum_{q_t} p(x_{t+1:T} | q_t, x_{1:t}) p(q_t, x_{1:t}) \\
= \sum_{q_t} p(x_{t+1:T} | q_t) p(q_t, x_{1:t}) \\
= \sum_{q_t} \beta_t(q_t) \alpha_t(q_t)
\]

So this means that for any $t$, we can get $p(x_{1:T})$ by using the corresponding $\alpha_t(i)$ and $\beta_t(i)$ quantities, for any $t$. 

Prof. Jeff Bilmes
EE596A/Winter 2013/DGMs – Lecture 3 - Jan 14th, 2013
We can get $p(x_{1:t})$ in many different ways:

\[
p(x_{1:T}) = \sum_{q_t} p(x_{1:T}, q_t)
\]
\[
= \sum_{q_t} p(x_{1:t}, x_{t+1:T}, q_t)
\]
\[
= \sum_{q_t} p(x_{t+1:T}|q_t, x_{1:t}) p(q_t, x_{1:t})
\]
\[
= \sum_{q_t} p(x_{t+1:T}|q_t) p(q_t, x_{1:t})
\]
\[
= \sum_{q_t} \beta_t(q_t) \alpha_t(q_t)
\]

- So this means that for any $t$, we can get $p(x_{1:T})$ by using the corresponding $\alpha_t(i)$ and $\beta_t(i)$ quantities, for any $t$.
- Which one to choose?
HMMs and elimination - $p(x_{1:t})$

We can get $p(x_{1:t})$ in many different ways:

$$p(x_{1:T}) = \sum_{q_t} p(x_{1:T}, q_t)$$

$$= \sum_{q_t} p(x_{1:t}, x_{t+1:T}, q_t)$$

$$= \sum_{q_t} p(x_{t+1:T}|q_t, x_{1:t})p(q_t, x_{1:t})$$

$$= \sum_{q_t} p(x_{t+1:T}|q_t)p(q_t, x_{1:t})$$

$$= \sum_{q_t} \beta_t(q_t)\alpha_t(q_t)$$

- So this means that for any $t$, we can get $p(x_{1:T})$ by using the corresponding $\alpha_t(i)$ and $\beta_t(i)$ quantities, for any $t$.
- Which one to choose?
- primarily finite-precision arithmetic (i.e., numerical) reasons for choosing one or another.
So we have the forward recursion as an elimination order and the backwards recursion as an elimination order.

Green order is $\alpha$-recursion, and blue order is $\beta$-recursion.
So we have the forward recursion as an elimination order and the backwards recursion as an elimination order.

Green order is $\alpha$-recursion, and blue order is $\beta$-recursion.

Since HMM is a tree, there are no additional fill-in edges via the elimination orders we have chosen.
HMMs and elimination

- If we just eliminated the hidden variables, we’d get:
HMMs and elimination

- If we just eliminated the hidden variables, we’d get:

- This has the same complexity $O(TN^2)$ even though the cliques have gotten very big.
If we just eliminated the hidden variables, we’d get:

\[ q_1 \quad q_2 \quad q_3 \quad q_4 \quad q_5 \]
\[ x_1 \quad x_2 \quad x_3 \quad x_4 \quad x_5 \]

This has the same complexity \( O(TN^2) \) even though the cliques have gotten very big.

Why?
Another elimination order will produce a different recursion for HMMs. For example, an *inside-outside* algorithm of sorts can be defined (normally this is used for computing marginals for stochastic context-free grammars) via the following elimination order.

![Graph showing elimination order](image)

**Exercise:** Derive the equations and a recursion for this elimination order for general length $T$ sequence. How is it different than standard forward/backward? Are there any advantages to this order?
But as mentioned before, we want more than just $p(x_{1:T})$.

We need clique posteriors $\gamma_t(i) = p(Q_t = i | x)$ and $\xi_t(i, j) = p(Q_{t-1} = i, Q_t = j | x)$. We can also get these from $\alpha$ and $\beta$.

\[
\gamma(q_t) = p(q_t | x_{1:T}) = p(x_{1:T} | q_t)p(q_t) / p(x_{1:T}) \\
= p(x_{1:t}, x_{t+1:T} | q_t)p(q_t) / p(x_{1:T}) \\
= p(x_{t+1:T} | q_t, x_{1:t})p(x_{1:t} | q_t)p(q_t) / p(x_{1:T}) \\
= p(x_{1:t}, q_t)p(x_{t+1:T} | q_t) / p(x) \\
= \alpha(q_t)\beta(q_t) / p(x) \\
= \alpha(q_t)\beta(q_t) / \sum_{q_t'} \alpha(q_t')\beta(q_t')
\]
HMM, posteriors

How best to compute $\gamma_t(i)$ for all $t$?

- **suboptimal way:**

```
1 for t = 1 \ldots T do
2 Compute $\alpha_t(j)$ starting at time 1 up to time $t$
3 Compute $\beta_t(j)$ starting at time $T$ down to time 1
4 Compute $\gamma_t(j)$
5 Update parameters based on current learning procedure.
```

- But this is extremely wasteful. Once we have computed $\alpha_t(j)$, for time $t$, should hold on to it for the next time step, $\alpha_{t+1}(j)$. Similarly, once we have $\beta_t(j)$ save it for previous time $\beta_{t-1}(j)$.

- Dynamic programming: optimal substructure & common subproblems of $\alpha_t(j)$, exactly the complete computation of $\alpha_{t-1}(j)$.

- This is clear when viewed as messages in the GM (messages don’t proceed until they have received appropriate incoming messages).
HMMs and message passing/LBP

Generic message definition for arbitrary $p \in \mathcal{F}(G, R)$

$$
\mu_{i \rightarrow j}(x_j) = \sum_{x_i} \psi_{i,j}(x_i, x_j) \prod_{k \in \delta(i) \backslash \{j\}} \mu_{k \rightarrow i}(x_i)
$$

(3.38)

- If graph is a tree, and if we obey message-passing protocol order, then we will reach a point where we’ve got marginals. I.e.,

$$
p(x_i) \propto \prod_{j \in \delta(i)} \mu_{j \rightarrow i}(x_i)
$$

(3.39)

and

$$
p(x_i, x_j) \propto \psi_{i,j}(x_i, x_j) \prod_{k \in \delta(i) \backslash \{j\}} \mu_{k \rightarrow i}(x_i) \prod_{\ell \in \delta(j) \backslash \{i\}} \mu_{\ell \rightarrow j}(x_j)
$$

(3.40)
LBP messages on an HMM are straightforward, and again since an HMM is a tree, we can use any message order that obeys the message-passing protocol.
HMM, and junction tree

- Can view this using junction tree algorithm with cliques and separators. One solution has each node a separator, & each edge a 2-clique, but there are others.
Can view this using junction tree algorithm with cliques and separators. One solution has each node a separator, & each edge a 2-clique, but there are others.
Using either of these junction trees, we can define a forward and/or backwards recursion, and we’ll again, in some cases, either recover exactly the $\alpha, \beta$ recursions, or computations that are quite similar.

For example, Hugin message passing computes the backward messages based on the forward (if we go right and then left), or computes the forward messages based on the backwards (if we go left and then right).
HMMs - summary so far

- Clique marginals are needed for standard learning procedures (EM and gradient calculation)
- Forward ($\alpha$) recursion is elimination from left-to-right (equivalently, just LBP message passing)
- Backward ($\beta$) recursion is elimination from right-to-left (again also LBP)
- “Inward” inference also possible via alternating elimination orders.
- HMM as a junction tree, various options for the cliques without increasing state space.
HMM, other recursions

It is possible to derive a temporal recursion for quantities other than $\alpha$ and $\beta$. E.g., here is a $\gamma_t(j) = p(Q_t = j|x_{1:T})$ backwards recursion.

$$\gamma(q_t) = \sum_{q_{t+1}} p(q_t, q_{t+1}|x_{1:T}) = \sum_{q_{t+1}} p(q_t|q_{t+1}, x_{1:T})p(q_{t+1}|x_{1:T})$$

$$= \sum_{q_{t+1}} p(q_t|q_{t+1}, x_{1:T})\gamma(q_{t+1}) = \sum_{q_{t+1}} p(q_t|q_{t+1}, x_{1:t})\gamma(q_{t+1})$$

$$= \sum_{q_{t+1}} \frac{p(q_t, q_{t+1}, x_{1:t})}{\sum_{q_t} p(q_t, q_{t+1}, x_{1:t})}\gamma(q_{t+1})$$

$$= \sum_{q_{t+1}} \frac{p(q_{t+1}|q_t)p(q_t, x_{1:t})}{\sum_{q_t} p(q_{t+1}|q_t)p(q_t, x_{1:t})}\gamma(q_{t+1})$$

$$= \sum_{q_{t+1}} \frac{p(q_{t+1}|q_t)\alpha_t(q_t)}{\sum_{q_t} p(q_{t+1}|q_t)\alpha_t(q_t)}\gamma(q_{t+1})$$

Therefore, there is a backwards pass recursion using just the $\alpha$’s without directly touching the observations again (better memory).
Filtering

- Filtering query is $p(q_t|x_{1:t})$
- If we need this for one particular $t$, this is identical to one of the posteriors we already have $p(q_T|x_{1:T})$
- More likely, we need recursion from $t$ to $t + 1$.
- This is obtained immediately from $\alpha$-recursion, since $\alpha_q(t) = p(x_{1:t}, Q_t = q)$, we have

$$p(q_t|x_{1:t}) = \frac{\alpha_{q_t}(t)}{\sum_{q'_t} \alpha_{q'_t}(t)} \quad (3.41)$$

- So, normalized $\alpha$s are just the filtering operation.
Prediction

- \( p(q_t|x_{1:s}), \) with \( t > s \).

This is also easily obtained from the \( \alpha \)s for we have

\[
p(q_t, x_{1:s}) = \sum_{q_s, q_{s+1}, \ldots, q_{t-1}} p(q_t, q_{t-1}, \ldots, q_{s+1}, q_s, x_{1:s}) \tag{3.42}
\]

\[
= \sum_{q_s, q_{s+1}, \ldots, q_{t-1}} p(q_t|q_{t-1}) p(q_{t-1}|q_{t-2}) \cdots p(q_{s+1}|q_s) \alpha_{q_s}(s) \tag{3.43}
\]

\[
= \sum_{q_t-1} p(q_t|q_{t-1}) \sum_{q_{t-2}} p(q_{t-1}|q_{t-2}) \cdots \sum_{q_{s}} p(q_{s+1}|q_s) \alpha_{q_s}(s) \tag{3.44}
\]
Thus, this is like running elimination (or message passing) on a graph where some of the observations are removed, as in:

\[ q_1 \quad q_2 \quad q_3 \quad q_4 \quad q_5 \]

\[ x_1 \quad x_2 \quad x_3 \]

for computing \( p(q_5|x_{1:3}) \).
Smoothing

- \( p(q_t|x_{1:u}) \), with \( t < u \).
- If we need this for one particular \( t \), this is identical to one of the posteriors we already have \( p(q_t|x_{1:T}) \).
- Otherwise, we might need to update \( p(q_t|x_{1:u}) \) for each \( t < u \) as \( u \) increases (observations come in).

- Involves re-sending messages (red). Each step requires \( O(T) \) additional messages, in total \( O(T^2) \) computation.
Edge Marginals

- Need $p(q_t, q_{t+1}|x_{1:T})$ for learning tasks
- Easy to obtain with both the $\alpha$ and $\beta$ quantities, since:

$$p(q_{t-1}, q_t|x_{1:T}) = \frac{p(q_{t-1}, q_t, x_{1:T})}{p(x_{1:T})} \quad (3.45)$$

and

$$p(q_{t-1}, q_t, x_{1:T}) = p(x_t|q_t)p(q_{t-1}, q_t, x_{1:t-1}, x_{t+1:T}) \quad (3.46)$$

$$= p(x_t|q_t)p(x_{t+1:T}|q_t)p(q_t, q_{t-1}, x_{1:t-1}) \quad (3.47)$$

$$= p(x_t|q_t)\beta_t(q_t)p(q_t|q_{t-1}, x_{1:t-1})p(q_{t-1}, x_{1:t-1}) \quad (3.48)$$

$$= p(x_t|q_t)\beta_t(q_t)p(q_t|q_{t-1})\alpha_{t-1}(q_{t-1}) \quad (3.49)$$
So all of the edge marginals can be computed using the standard recursions.

There are several aspects of HMMs we will discuss: 1) how flexible are HMMs, 2) real-world inference in HMMs (what to do when state space gets large), 3) time-space tradeoffs.
HMMs are more powerful than you might think. We’ll see that many DGMs can be represented by HMMs, so before we move on to DGMs, we should understand how flexible HMMs are (and then as we go through course, we’ll see what the penalties are for making such HMM representations).

We next visit a set of properties about HMMs that should be remembered.
Observations are not i.i.d.

- Joint probability under an HMM:
  \[
  p(X_{t:t+h} = x_{t:t+h}) = \sum_{q_{t:t+h}} \prod_{j=t}^{t+h} p(X_j = x_j | Q_j = q_j) a_{q_{j-1}q_j}.
  \]

- Unless only one state is possible, observations do not factorize.
- In an HMM, there are no statements of the form \( A \perp \perp B \).
- HMMs have conditional independence properties (like all DGMs have).

\[
p(X_{t:t+h} = x_{t:t+h} | Q_{t:t+h} = q_{t:t+h}) = \prod_{\tau=t}^{t+h} p(X_\tau = x_\tau | Q_\tau = q_\tau).
\]
Observations are not “Viterbi i.i.d.”

The Viterbi path (most-probable explanation) of an HMM is defined as follows:

\[ q_{1:T}^* = \arg \max_{q_{1:T}} p(X_{1:T} = x_{1:T}, q_{1:T}) \]

We may wish to max-marginalize the hidden states. Does that lead to i.i.d.?

The “Viterbi” distribution of the HMM is:

\[
\begin{align*}
    p_{\text{vit}}(X_{1:T} = x_{1:T}) &= c \, p(X_{1:T} = x_{1:T}, Q_{1:T} = q_{1:T}^*) \\
    &= c \max_{q_{1:T}} p(X_{1:T} = x_{1:T}, Q_{1:T} = q_{1:T}) \\
    &= c \max_{q_{1:T}} \prod_{t=1}^{T} p(X_t = x_t | Q_t = q_t) p(Q_t = q_t | Q_{t-1} = q_{t-1})
\end{align*}
\]

where \( c \) is a positive normalizing constant.
Observations are not “Viterbi i.i.d.”

- The Viterbi path (most-probable explanation) of an HMM is defined as follows:

\[ q_{1:T}^* = \text{argmax}_{q_{1:T}} p(X_{1:T} = x_{1:T}, q_{1:T}) \quad (3.51) \]

- This is a standard method for finding a mapping from observations \( \bar{x}_{1:T} \) to the candidate answers \( q_{1:T}^* \).

- Other times, we might have \( k \) HMMs, corresponding to \( k \) classes

\[ \{ p_k(x_{1:T}, q_{1:T}) : k = 1 \ldots K \} \quad (3.52) \]

and we wish to perform classification amongst the \( K \) discrete set of classes as follows

\[ (k^*, q_{1:T}^*) = \text{argmax}_{q_{1:T}, k} p_k(x_{1:T}, q_{1:T}) \quad (3.53) \]

and \( k^* \) becomes the hypothesized answer.
Observations are not “Viterbi i.i.d.”

- We can view this as $K$ distributions over just the observations, i.e.,

$$p^k_{\text{vit}}(x_{1:T}) \propto p_k(x_{1:T}, q^*_1:T)$$

$$\propto \max_{q_1:T} p_k(x_{1:T}, q_1:T) \tag{3.55}$$

$$\propto \max_{q_1:T} \prod_{t=1}^T p_k(x_t|q_t)p_k(q_t|q_{t-1}) \tag{3.56}$$

- Note that this max-marginal can be completely different than the standard summation marginal

$$\sum_{q_1:T} \prod_{t=1}^T p_k(x_t|q_t)p_k(q_t|q_{t-1}) \tag{3.57}$$

- This is just a different semi-ring. The resulting distribution over observations does not in general factorize, so no i.i.d. here either. The distribution $p^k_{\text{vit}}(x_{1:T})$ also does not in general have any (marginal) independence assumptions.
HMMs and stationarity

- HMM is a stochastic process over $\{X_t\}_t$
- An HMM is stationary whenever
  \[ p(X_{t_1:h:n+h} = x_{1:n}) = p(X_{t_1:n} = x_{1:n}) \]  
  (3.58)

When might this be the case?
- We have
  \[
p(X_{t_1:n} + h = x_{1:n}) \\
  = \sum_{q_{1:n}} p(X_{t_1:n} + h = x_{1:n}, Q_{t_1:n} + h = q_{1:n}) \\
  = \sum_{q_{1:n}} p(Q_{t_1} + h = q_1)p(X_{t_1} + h = x_1|Q_{t_1} + h = q_1) \\
  \prod_{i=2}^{n} p(X_{t_i} + h = x_i|Q_{t_i} + h = q_i)p(Q_{t_i} + h = q_i|Q_{t_i-1} + h = q_{i-1})
\]
HMMs and stationarity (cont. II)

\[
\sum_{q_1} p(Q_{t_1+h} = q_1) p(X_{t_1+h} = x_1 | Q_{t_1+h} = q_1) \\
\sum_{q_1} \prod_{i=2}^{n} p(X_{t_i+h} = x_i | Q_{t_i+h} = q_i) p(Q_{t_i+h} = q_i | Q_{t_{i-1}+h} = q_{i-1}) \\
= \sum_{q_1} p(Q_{t_1+h} = q_1) p(X_{t_1+h} = x_1 | Q_{t_1+h} = q_1) \\
\sum_{q_1} \prod_{i=2}^{n} p(X_{t_i} = x_i | Q_{t_i} = q_i) p(Q_{t_i} = q_i | Q_{t_{i-1}} = q_{i-1}) \\
= \sum_{q_1} p(Q_{t_1+h} = q_1) p(X_{t_1} = x_1 | Q_{t_1} = q_1) f(x_{2:n}, q_1)
\]

where \( f(x_{2:n}, q_1) \) is a function that is independent of the variable \( h \).

For HMM stationarity to hold, it is required that

\( p(Q_{t_1+h} = q_1) = p(Q_{t_1} = q_1) \) for all \( h \).
Therefore, the HMM’s stationarity condition is entirely determined by if the underlying hidden Markov chain is stationary.

Consider the way in which HMMs are often used.

- Long chains
- Chains with cycle transition matrices
- Chains with upper-triangular matrices
- Chains with strictly left-to-right transitions. Ex: speech recognition

<table>
<thead>
<tr>
<th>Word</th>
<th>Pronunciation</th>
<th>Word</th>
<th>Pronunciation</th>
</tr>
</thead>
<tbody>
<tr>
<td>cat</td>
<td>[kæt]</td>
<td>goose</td>
<td>[gus]</td>
</tr>
<tr>
<td>cats</td>
<td>[kæts]</td>
<td>geese</td>
<td>[gis]</td>
</tr>
<tr>
<td>pig</td>
<td>[pɪɡ]</td>
<td>hedgehog</td>
<td>['hɛdʒ.hɔɡ]</td>
</tr>
<tr>
<td>pigs</td>
<td>[pɪɡz]</td>
<td>hedgehogs</td>
<td>['hɛdʒ.hɔɡz]</td>
</tr>
<tr>
<td>fox</td>
<td>[fɑks]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>foxes</td>
<td>['fɑks.riz']</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
One of the most widely used HMMs in practice is one where the observation distributions are Gaussian mixtures, where
\[
p(x|q) = \sum_c p(x|q, c)p(c|q)
\]
\[
= \sum_m \mathcal{N}(x|\mu_{qm}, \Sigma_{qm}) c_{mq}
\]
and where
\[
\mathcal{N}(x|\mu, \Sigma) = \frac{1}{|2\pi\Sigma|^{d/2}} \exp \left( -\frac{1}{2} (x - \mu)^T \Sigma^{-1} (x - \mu) \right)
\]

The HMM BN becomes
Correlated & Covariance

- Correlation between two real random vectors $X$ and $Y$
  \[
  \text{cor}(X, Y) = E[XY^\top]
  \]  
  (3.62)

- Covariance between two real random vectors $X$ and $Y$
  \[
  \text{cov}(X, Y) = E[(X - EX)(Y - EY)^\top] = E[XY^\top] - E[X]E[Y^\top]
  \]  
  (3.63)

- $X$ and $Y$ are **uncorrelated** if $\text{cov}(X, Y) = 0$.
- $\text{cov}(X, Y) = \text{cor}(X, Y)$ if either the means are zero.
- If $X \perp \perp Y$ then $\text{cov}(X, Y) = 0$ but vice verse only if $X, Y$ are jointly Gaussian.
- $\text{cov}(X, Y) = 0$ is indication of lack of linear dependence, and hint there might not be strong dependence.
Correlation over time of simple HMM

Consider single-component Gaussian HMM (i.e., for each state, the observation distribution is a single multivariate Gaussian).

Assume that the Markov chain is currently stationary, with stationary distribution $\pi$.

What about $\text{cov}(X_t, X_{t+h})$? Is it zero? How does it decay?

Computing $E[X_t]$

$$EX_t = \int xp(X_t = x)dx = \int x \sum_i p(X_t = x|Q_t = i)\pi_i dx \quad (3.64)$$

$$= \sum_i E[X_t|Q_t = i]\pi_i = \sum_i \mu_i \pi_i \quad (3.65)$$

so this is just the weighted sums of means, where the weights are the stationary distribution probabilities.
Correlation over time of simple HMM

Similarly,

\[ E[X_t X'_{t+h}] \]

\[ = \int xy' p(X_t = x, X_{t+h} = y) dx dy \]  \hspace{1cm} (3.66)

\[ = \int xy' \sum_{ij} p(X_t = x, X_{t+h} = y \mid Q_t = i, Q_{t+h} = j) \]  \hspace{1cm} (3.67)

\[ p(Q_{t+h} = j \mid Q_t = i) \pi_i dx dy \]  \hspace{1cm} (3.68)

\[ = \sum_{ij} E[X_t X'_{t+h} \mid Q_t = i, Q_{t+h} = j] (A^h)_{ij} \pi_i \]  \hspace{1cm} (3.69)

The above equations follow from \( p(Q_{t+h} = j \mid Q_t = i) = (A^h)_{ij} \) (i.e., the Chapman-Kolmogorov equations) where \((A^h)_{ij}\) is the \(i, j^{th}\) element of the matrix \(A\) raised to the \(h\) power.
Because of the HMM conditional independence properties, we have

\[
E[X_t X'_{t+h} | Q_t = i, Q_{t+h} = j] = E[X_t | Q_t = i] E[X'_{t+h} | Q_{t+h} = j] = \mu_i \mu'_j
\]

(3.72)

yielding

\[
E[X_t X'_{t+h}] = \sum_{ij} \mu_i \mu'_j (A^h)_{ij} \pi_i
\]

(3.74)

Covariance is therefore:

\[
\text{cov}(X_t, X_{t+h}) = \sum_{ij} \mu_i \mu'_j (A^h)_{ij} \pi_i - \left( \sum_i \mu_i \pi_i \right) \left( \sum_i \mu_i \pi_i \right)'
\]
Correlation over time of simple HMM

- Thus, \( \text{cov}(X_t, X_{t+h}) \) is in general not equal to zero.

- But recall, \( A^h \xrightarrow{h} 1\pi^\top \) from lecture 3, and this is a matrix with all rows equal to the stationary distribution. Thus,

\[
\text{cov}(X_t, X_{t+h}) = \sum_{ij} \mu_i \mu_j' (A^h)_{ij} \pi_i - \left( \sum_i \mu_i \pi_i \right) \left( \sum_i \mu_i \pi_i \right)' \]

\[
\xrightarrow{h} \sum_{ij} \mu_i \mu_j' \pi_j \pi_i - \left( \sum_i \mu_i \pi_i \right) \left( \sum_i \mu_i \pi_i \right)' = 0
\]

- Thus, while the covariance between to observations is not necessarily zero in an HMM, once we are at a (unique) stationary distribution, this covariance goes to zero exponentially fast.
Correlation over time of simple HMM

- Example of the decay in the mutual-information correlation from a real-world HMM. i.e., we see $f(\tau) = I(X_t; X_{t+\tau})$, where $I()$ is the mutual information function.
- This is compared against i.i.d. samples (high peak at $\tau = 0$ is expected).
Markov chain, state duration distribution is geometric

Let $D$ be such a random variable, then

$$P(D = d) = p^{d-1}(1 - p) \quad (3.75)$$

where $d \geq 1$ is an integer and $p = a_{ii}$, if $D$ is the duration random variable for state $i$ of the chain, giving:

Many sequential tasks have sub-segments that do not follow this distribution
State Duration Modeling

- Many “tricks” to using an HMM, can alleviate such problems.
- “state-tying”, where multiple states have the same observation distribution (parameters are shared).
- That is, state $q$ and $q'$ are tied if it is the case that

$$p(x|Q_t = q) = p(x|Q_t = q') \quad \forall x \in D_X$$

(3.76)

- If $n$ states in a series are strung together, in a series, all of which share the same observation distribution, that observation distribution will be active for as long as we are in that state. For example:
This corresponds to the sum of random variables. Let \( \{D_i\}_i \) be a collection of independent geometrically distributed random variables with parameter \( p \), and let \( W_r = \sum_{i=1}^r D_r \), then

\[
p(W_r = k) = \binom{k - 1}{r - 1} p^r (1 - p)^{k-r}, \quad k = r, r + 1, \ldots
\]  

(3.77)

This is a “negative binomial distribution”, and looks like the following:
State Duration Modeling

- If we have multiple parallel states in series, all of which share the same observation distribution, we can construct much more interesting (multimodal) distributions.
- For example, the following left chain will have distribution as shown on the right (a mixture of negative binomial distributions).

Other examples: very long chains, ladders, fixed-length distributions (histograms), and so on.

HMMs can have flexible distributions, cost of extra states.
Sources for Today’s Lecture

- “doc.pdf” sections 8.1 - 8.3