Logistics

Review

Cumulative Outstanding Reading

- read the section on dynamic Bayesian networks in 'doc.pdf'
- read the section on inference in 'doc.pdf'
Final Project - pending deadlines

Every Monday from now up until March 18th (our final presentations day). Specific deadlines are as follows:

- March 18th, **10:00am**: 4 page project writeup (conference style, with abstract).
- March 18th, **12:30pm**: presentation slides (pdf, pptx, or key)
- Final Presentations, Monday, March 18, 2013, 230-420 pm, PCAR 492

Note, March 18th deadlines are at 10:00am and 12:30pm, while the others are at 11:45pm at night.

Final Project

- Final presentations, 15 minutes
- Final reports, no more than 4 pages and no less than 11 point font, and no more than 1 inch margins on all sides!
Separator to Separator Messages

- Separator-to-Separator idea can be generalized to an arbitrary junction tree clique node. Cliques are never stored, only separators.

\[
\phi(x_{S_0}) = \sum_{x_C \setminus S_0} \psi(x_C) \phi(x_{S_1}) \phi(x_{S_2}) \phi(x_{S_3}) \phi(x_{S_4}) \tag{17.5}
\]

Sparsity and Sep-to-Sep Messages

- For this message, we call the separators inside the sum, the “incoming separators”, and the outgoing one \(S_o\) the outgoing separator.
- To exploit sparsity, before message propagation takes place, all incoming separators to a clique are ordered and reorganized into two parts:
  1. accumulated intersection between separator variables and previous separator variables
  2. a residual set of variables
- \(M\) incoming separators \(S_1, \ldots, S_M\), then potential functions organized as follows:

\[
\phi(S_1), \ \phi(S_2 \cap S_1; S_2 \setminus S_1), \ \ldots, \ \phi(S_m \cap S\{m\}; S_m \setminus S\{m\}), \ldots
\]

where \(S\{m\} \triangleq \bigcup_{i=1}^{m-1} S_i\).
Separators are iterated and scores multiplied together before entering into next separator.

Each separator checked if compatible with accumulated intersection, only surviving partial hypotheses continue onto next separator and ultimately clique.

Separators also used to provide sparsity information even for cliques that they are not incoming to. I.e., zero-one versions of the separators (or projections thereof) are used for other cliques (see figure).

Consider this dynamic junction tree segment between frames $t - 1$ and $t$.

![Separation Driven Example: Sparsity and Messages]
The goal is to expand factor $f(a, b, c, d, e) = \phi_1(a, b, c)\phi_2(b, c, d)\phi_3(d, e)\phi_4(a, b, f)\psi(a, b, c, d, e, f)$ which depends on factor $g(a, b, f, g, h) = \phi_5(a, b)\phi_6(g, h)\psi(a, b, f, g, h)$

- any of the factors might be sparse
- factor $\phi_1(a, b, c)$ corresponds to the clique in the previous chunk separated by the discovered chunk separator
- $\phi_1(a, b, c)$ might possess important sparsity to preserve from earlier portions of the DGM.

Separators incoming to clique $(a, b, c, d, e, f)$ organized so that once they are instantiated, only the non-zero entries are iterated

- non-zero entries of one separator are used to index directly into the entries of the neighboring separator that are guaranteed to have non-zero entries.
- Ex: once $(a, b, c)$ has value, the values $(b, c)$ are used to lookup the values of $d$ (if any) that can co-exist with the values $(b, c)$.
- Also, sparsity in intra-chunk separator $(a, b, c)$ is used to provide real scores to the clique $(a, b, c, d, e, f)$ but only $0/1$ sparsity information to other cliques in the chunk (namely $(\ldots, b, c, d)$ and $(a, b, i, j)$).
Sparsity and Messages

- Now, we partition the variables in these separators in to two (nec. disjoint) subsets.
- That is, for each $m \in 1 \ldots M$, partition $S_m$ into $S_m \cap S_m^\{m\}$ and $S_m \setminus S_m^\{m\}$, where $S_m^\{m\} \triangleq \cup_{i=1}^{m-1} S_i$ is the accumulation of all separators up to $m - 1$
- $S_m \setminus S_m^\{m\}$ is the “accumulated intersection”
- $S_m \setminus S_m^\{m\}$ is the novelty or innovation of separator $m$.
- Thus the separator potential functions are organized as follows:

$$
\phi(x_{S_1}), \phi(x_{S_2} \cap S_1; x_{S_2} \setminus S_1), \ldots, \phi(x_{S_m} \cap S_m^\{m\}; x_{S_m} \setminus S_m^\{m\}), \ldots
$$

(17.1)

- Now, what does $\phi(x_{S_m} \cap S_m^\{m\}; x_{S_m} \setminus S_m^\{m\})$ buy us?
- Given a set of values of the variables $x_{S_m^\{m\}}$ then $\phi(x_{S_m} \cap S_m^\{m\}; x_{S_m} \setminus S_m^\{m\})$ is so organized that we can quickly look up (and iterate over) only those values of $x_{S_m \setminus S_m^\{m\}}$ that, for a given value of $x_{S_m^\{m\}}$ are compatible (i.e., make $\phi(x_{S_m} \cap S_m^\{m\}; x_{S_m} \setminus S_m^\{m\}) > 0$).
- Since $x_{S_m^\{m\}}$ is already set, any values of $x_{S_m \setminus S_m^\{m\}}$ when combined with $x_{S_m^\{m\}}$ that lead to a zero should never be considered.
- But if $x_{S_m^\{m\}}$ is set, it can only effect separator $m$ via $x_{S_m} \cap S_m^\{m\}$
- We think of $\phi(x_{S_m} \cap S_m^\{m\}; x_{S_m} \setminus S_m^\{m\})$ as a sparse tensor, where we can quickly look up one dimension in the matrix via $x_{S_m} \cap S_m^\{m\}$ and iterate only over those entries in the matrix $x_{S_m} \setminus S_m^\{m\}$ that lead to $\phi(x_{S_m} \cap S_m^\{m\}; x_{S_m} \setminus S_m^\{m\}) = \phi(x_{S_m}) > 0$.
- Any other values need not be considered.
Separator driven

Separator driven sparse joins in clique expansion

\begin{align*}
\text{for } & x_{S_1} \in \{x_{S_1} : \phi(x_{S_1}) > 0\} \text{ do} \\
& \quad \text{for } x_{S_2 \setminus S_1} \in \{x_{S_2 \setminus S_1} : \phi(x_{S_2 \cap S_1}; x_{S_2 \setminus S_1}) > 0\} \text{ do} \\
& \quad \quad \quad \ldots \\
& \quad \quad \quad \text{for } x_{S_m \setminus S_{\{m\}}} \in \{x_{S_m \setminus S_{\{m\}}} : \phi(x_{S_m \cap S_{\{m\}}}; x_{S_m \setminus S_{\{m\}}}) > 0\} \text{ do} \\
& \quad \quad \quad \quad \ldots \\
& \quad \quad \quad \quad \text{expand out clique using current value } x_{S_1:M}. \\
\end{align*}

Additional sparsity optimization in a JT

- Also, sparsity in intra-chunk separator \((a, b, c)\) is used to provide real scores to the clique \((a, b, c, d, e, f)\) but only 0/1 sparsity information to other cliques in the chunk (namely \((\ldots, b, c, d)\) and \((a, b, i, j))\).
For much of the next discussion, it is sufficient to consider a junction chain.

We use notation: let $C_t$ be the clique variables at time $t$, let $S_t = C_t \cap C_{t+1}$ be the separator between successive cliques. $X$ is our set of random variables.

Thus, $X_{C_t}$ and $X_{C_{t+1}}$ are the same random variables but shifted by one chunk.

$$
\phi(x_{S_{t+1}}) = \sum_{x_{C_{t+1}\setminus S_{t+1}}} \psi(x_{C_{t+1}}) \phi(x_{S_t})
$$

As before, we assume that $X_{S_t}$ is a clique in the chunk, and that the summation in this equation is free to use any method it wants (e.g., sparse junction tree message passing, etc.).

At some point, even this computation is too expensive. I.e., if $S$ contains many variables, even if $X_S$ has much local structure.

This is a consequence of Rose’s theorem: even if there is nice initial factorization over $S$ at the beginning of the model, these factorization properties quickly vanish as we eliminate in from the left to the right.

This is the reason for completing $X_S$ in the first place. While we used mincut to find the $S$ that is as small as possible, the best case might not be small enough.

What can we do?
DGM chains and pruning

\[ \phi(x_{S_{t+1}}) = \sum_{x_{C_{t+1}} \setminus S_{t+1}} \psi(x_{C_{t+1}}) \phi(x_{S_{t}}) \] (17.4)

- Since we now have sparsity preserving message passing routines, we might be able to “add sparsity” to the model, at the cost of some approximation error.
- \( \phi(x_{S_{t}}) \) might have some inherent sparsity in it already, but we can remove entries that fall below some threshold.

Beam score pruning

\[ \phi(x_{S_{t+1}}) = \sum_{x_{C_{t+1}} \setminus S_{t+1}} \psi(x_{C_{t+1}}) \phi(x_{S_{t}}) \] (17.5)

- Normal state space is \( \{x_{S} : \phi(x_{S}) > 0\} \)
- We can reduce it by defining minimum threshold \( \tau \) in
  \[ \{x_{S} : \phi(x_{S}) > \tau\} \] (17.6)
- Score based beam pruning
  \[ \tau = \left( \frac{\text{max}_{x_{S}} \phi(x_{S})}{b} \right) \] (17.7)

where \( b \) is a constant known as the “beam width”, set by the user.
Beam state pruning

- State-space based pruning: Keep the state space of the separator to be no larger than \( k \).
- Order values so that

\[
\phi(x_S^{(1)}) \geq \phi(x_S^{(2)}) \geq \cdots \geq \phi(x_S^{(N)}) \tag{17.8}
\]

- Then state space becomes

\[
\{ x_S : \phi(x_S) > \phi(x_S^{(k)}) \} \tag{17.9}
\]

- Note that we only keep the top \( k \), even if there ties at value \( \phi(x_S^{(k)}) \), this is to keep the number of entries bound by \( k \).
- This can useful as it bounds the computation based on \( k \), regardless of the cost of doing so.

Beam mass percentage pruning

- Keep the states that ensure that a certain percentage of the mass is retained.
- Let \( \kappa \) be the fraction of mass to retain.
- Order values so that

\[
\phi(x_S^{(1)}) \geq \phi(x_S^{(2)}) \geq \cdots \geq \phi(x_S^{(N)}) \tag{17.10}
\]

- Find smallest \( k \) such that

\[
\frac{\sum_{i=1}^{k} \phi(x_S^{(i)})}{\sum_{i=1}^{N} \phi(x_S^{(i)})} \geq \kappa \tag{17.11}
\]

- Sometimes useful to put a lower bound \( k_{\text{min}} \) on the state space (i.e., if one state already is above threshold). \( k = \max(k, k_{\text{min}}) \).
- Then state space becomes

\[
\{ x_S : \phi(x_S) > \phi(x_S^{(k)}) \} \tag{17.12}
\]

- This can useful to ensure local cost of pruning is not too bad.
Pruning - the good

- In general, we are approximating the DGM by using a (sparser) potential function $\tilde{\phi}(x_S)$ in place of $\phi(x_S)$.
- With our sparse message passing scheme, this can significantly reduce the computation.
- Pruning can occur anywhere, including within the chunk to produce $\tilde{\psi}(x_{Ct+1})$ in place of $\psi(x_{Ct+1})$ — clique pruning can potentially be more nuanced while separator pruning is more of a bludgeon (both can work well though).
- Pruning is extremely effective in practice. Almost all speech recognition systems (with massive state spaces) use such pruning approaches.

Pruning - the (not so) good

- Pruning makes local decisions. A candidate state that is pruned off at time $t$ might end up being the only way to “explain” an observation that occurs later at time $t + \tau$.
- Could do generalized pruning – search with backtracking to backup. Asynchronous state expansion with backtracking. A* search, etc.
- Pruning does not enable factorization - all interactions are potentially retained. I.e., with pruning, there is no $S_1, S_2 \subset S$ such that $S_1 \cup S_2 = S$, $S_1 \cap S_2 = \emptyset$ where $\tilde{\phi}(x_S) = \tilde{\phi}(x_{S_1})\tilde{\phi}(x_{S_2})$
- Therefore, loss of structural properties of the factored state (Rose’s entanglement theorem) space is still present.
Persistent variables independent only at first frame!
If separator was structured

\[ \phi(x_{S_{t+1}}) = \sum_{x_{C_{t+1} \setminus S_{t+1}}} \psi(x_{C_{t+1}}) \phi(x_{S_t}) \]  

(17.13)

- In above equation, no way to uncouple variables within the interface separator (compulsory completion of interface separator).

Suppose instead \( G[S] \) was not a clique, so that \( \phi(x_{S_t}) \) itself factorized, say \( \phi(x_{S_t}) = \phi(x_{S_t}^{(1)}) \phi(x_{S_t}^{(2)}) \)

- On the left, interface separator is factored:

- With right factorization, could lead to significant computational savings.
If separator was structured

\[
\phi(x_{S_{t+1}}) = \sum_{x_{C_{t+1}\setminus S_{t+1}}} \psi(x_{C_{t+1}}) \phi(x_{S_t}) \quad (17.14)
\]

- On left, many more variables coupled than on right.

If separator was structured

\[
\phi(x_{S_{t+1}}) = \sum_{x_{C_{t+1}\setminus S_{t+1}}} \psi(x_{C_{t+1}}) \phi(x_{S_t}) \quad (17.15)
\]

- Note that the problem repeats itself: even if separator \( S_t \) was structured, after one expansion and summation, next separator \( S_{t+1} \) would loose structure again (by Rose’s thm).
- But we could recover structure by “projecting” down to structured distribution once again, and then repeat.
- Basic left-to-right process: use factored incoming separator

\begin{verbatim}
for \( t \in \{1 \ldots T\} \) do
    \( \phi(x_{S_{t+1}}) = \sum_{x_{C_{t+1}\setminus S_{t+1}}} \psi(x_{C_{t+1}}) \tilde{\phi}(x_{S_t}) \)
    Project \( \phi(x_{S_{t+1}}) \) down to more structured approximation \( \tilde{\phi}(x_{S_{t+1}}) \)
\end{verbatim}
Questions:

- Will this work?
- What is incurred error?
- Won’t the error accumulate over time?
- How to chose way to project down (i.e., \( \tilde{\phi} \))
- How best to do expansion given factored form \( \tilde{\phi} \)
- How to analyze
- How much savings can we expect?

We next try to address some of these questions.

Note: such a method, in the context of a traditional DBN, is called the Boyen and Koller algorithm (and is a special case of “assumed density filtering” which is also related to EP (expectation propagation)).

HMM Case

- We first discuss this in the context of HMMs.
- Key issue: the transition matrix \( A \) mixes things up and in fact reduces any distortion between previous approximate distributions.
- Consider step-by-step updates in HMM of the posterior over the HMM state.

\[
p(Q_t = q | x_1:t) \Rightarrow p(Q_{t+1} = q | x_1:t+1) \quad (17.16)
\]

- Notation for this:

\[
p_{1:t,t}(q) \overset{\Delta}{=} p(Q_t = q | x_1:t) \quad (17.17)
\]

- Note that this is related to our \( \alpha \) recursion \( \alpha_t(q_t) = p(x_1:t, q_t) \) Thus,

\[
p(Q_t = q | x_1:t) = \frac{p(x_1:t, q_t)}{\sum_{q_t'} p(x_1:t, q_t')} = \frac{\alpha_t(q_t)}{\sum_{q_t'} \alpha_{t'}(q_{t'})} \quad (17.18)
\]
HMM case

- We can update state posterior in two steps (like Kalman filtering):
  - **Prior state distribution**: prior to seeing the observation at time $t$, we can estimate a distribution over $Q_t$ as follows
    
    $p_{1:t-1,t}(q) \triangleq p(Q_t = q | \bar{x}_{1:t-1}) = \sum_r p(q|r)p(Q_{t-1} = r | \bar{x}_{1:t-1})$
    
    $= \sum_r p(q|r)p_{1:t-1,t-1}(r) \quad \text{(17.19)}$

- **Posterior state distribution**: after seeing the observation at time $t$, we can multiply in the observation distribution, renormalize, and we'll get the update:
    
    $p_{1:t,t}(q) = p(Q_t = q | \bar{x}_{1:t}) = \frac{p_{1:t-1,t}(q)p(\bar{x}_t|q)}{\sum_{q'} p_{1:t-1,t}(q')p(\bar{x}_t|q')} \quad \text{(17.21)}$

I.e., standard “filtering” problem, in Kalman filtering (also used in general for sensor monitoring in dynamical systems with hidden state). Essentially identical to $\alpha$ recursion.
HMM case: two step prior and posterior state distribution updates

- Now suppose we started with an approximation of the posterior belief state at time $t$, $\tilde{p}_{1:t,t}$ rather than $p_{1:t,t}$.
- If this approximation goes through the same two-step transformation as above, we would get an approximation $\hat{p}_{1:t+1,t+1} = \mathcal{O}[\mathcal{T}[\tilde{p}_{1:t,t}]]$ of the posterior at time $t+1$ rather than the exact $p_{1:t+1,t+1} = \mathcal{O}[\mathcal{T}[p_{1:t,t}]]$.
- Suppose we have some measure of how $\tilde{p}_{1:t,t}$ and $p_{1:t,t}$ differ.
- Does the difference between $\tilde{p}_{1:t,t}$ and $p_{1:t,t}$ increase or decrease after the transformation?
- That is, is difference $(\tilde{p}_{1:t,t}, p_{1:t,t})$ larger or smaller than difference $(\hat{p}_{1:t+1,t+1}, p_{1:t+1,t+1})$?
- Do the operations $\mathcal{O}[\mathcal{T}[\cdot]]$ tend to diminish any differences or do they tend to enhance them?

Measure of difference: Kullback-Leibler Divergence

- Given two distributions $p$ and $q$ we have
  \[
  D(p_1||p_2) = \sum_x p_1(x) \log \frac{p_1(x)}{p_2(x)} \tag{17.25}
  \]
- $\text{KL}$-divergence is convex in the pair, i.e., given $p_1, q_1$ and $p_2, q_2$ we have that
  \[
  \lambda_1 D(p_1||q_1) + \lambda_2 D(p_2||q_2) \geq D(\lambda_1 p_1 + \lambda_2 p_2 || \lambda_1 q_1 + \lambda_2 q_2) \tag{17.26}
  \]
  for convex mixture $(\lambda_1, \lambda_2)$.
- $\text{KL}$-divergence is never negative, and
  \[
  D(p||q) = 0 \iff p = q \tag{17.27}
  \]
- $\text{KL}$-divergence is not a metric, not symmetric, but measures bit-penalty in coding/compressing with $q$ when real distribution is $p$. 

KL-divergence and Markov chains

- What happens to distributions that might be different as they go through a time-homogeneous Markov chain.
- Let $p_1(q_t)$ and $p_2(q_t)$ be state distributions in a Markov chain at time $t$.
- Let $p(q|r)$ be a stochastic transition matrix.
- If both $p_1$ and $p_2$ undergo the same stochastic transition, do they get closer or farther apart?
- Can use the chain rule in two ways based on $p(a,b) = p(a)p(b|a) = p(b)p(a|b)$:

  \[
  D(p_1(q_t, q_{t+1}) || p_2(q_t, q_{t+1})) = D(p_1(q_t) || p_2(q_t)) + D(p_1(q_{t+1}|q_t) || p_2(q_{t+1}|q_t))
  \]
  \[
  = D(p_1(q_{t+1}) || p_2(q_{t+1})) + D(p_1(q_t|q_{t+1}) || p_2(q_t|q_{t+1}))
  \]
KL-divergence and Markov chains - transition

- But $D(p_1(q_{t+1}|q_t) || p_2(q_{t+1}|q_t)) = 0$ since this is the Markov chain transition matrix. Therefore,

$$D(p_1(q_t) || p_2(q_t)) = D(p_1(q_{t+1}) || p_2(q_{t+1})) + D(p_1(q_t|q_{t+1}) || p_2(q_t|q_{t+1}))$$

(17.33)

or

$$D(p_1(q_t) || p_2(q_t)) \geq D(p_1(q_{t+1}) || p_2(q_{t+1}))$$

(17.34)

- With $p_i(q_{t+1}) = \sum_{q_t} p_i(q_t)p(q_{t+1}|q_t)$
- So state distributions get closer in a Markov chain after undergoing stochastic transition matrix operation $\mathcal{T}$.

$$D(\mathcal{T}[p_1] || \mathcal{T}[p_2]) \leq D(p_1 || p_2)$$

(17.35)

---

**Theorem 17.7.1**

Let $p(q_t)$ and $\hat{p}(q_t)$ be distributions at time $t$, and when they undergo the same transition $p(q_{t+1}|q_t)$, giving $p(q_{t+1})$ and $\hat{p}(q_{t+1})$ respectively. Then

$$D(p(q_t) || \hat{p}(q_t)) \geq D(p(q_{t+1}) || \hat{p}(q_{t+1}))$$

(17.36)
KL-divergence and Markov chains - observation

- What if we multiply two state distributions by a vector of likelihoods. I.e., \( p(x_t | q_t) \geq 0 \). I.e., compare \( p(q) \) and \( p(q) \) with \( p'(q) = \frac{p(q)p(x|q)}{\sum_{q'} p(q')p(x|q')} \) and \( \tilde{p}'(q) = \frac{\tilde{p}(q)p(x|q)}{\sum_{q'} \tilde{p}(q')p(x|q')} \).
- This is not a Markov transition matrix operation, since the fundamental operation of \( O \) is a element wise vector multiply and then renormalize, which is different than the vector-matrix multiply operation of \( T \).
- For a fixed value \( \bar{x} \), this could increase the KL divergence.
- Example: \( p(q) = (1, 3, 1) / 5 \) and \( \tilde{p}(q) = (3, 1, 1) / 5 \), then \( D(p||\tilde{p}) = 0.63399 \). Multiply by vector \( (1, 1, 0.1) \) gives \( p'(q) = (1, 3, .1) / 4.1 \) and \( \tilde{p}'(q) = (3, 1, .1) / 4.1 \), with \( D(p'||\tilde{p}') = 0.77315 \).

- However, assume that we have a model-based prior distribution over the observations, i.e., there is a probability distribution \( p(x) = \sum_q p(x|q)p(q) \) over vectors \( x \) based on the model. Then we have

\[
\sum_x p(x) \left( D(p'(q)||\tilde{p}'(q)) \right) \leq D(p(q)||\tilde{p}(q)) \tag{17.37}
\]
More precisely, we have following theorem.

**Theorem 17.7.2**

Let $p_1(q)$ and $p_2(q)$ be two distributions over states with a well defined KL-divergence. Let $p(x|q)$ be any distribution. Define

$$p'_1(q) = \frac{p_1(q)p(x|q)}{\sum_{q'} p_1(q')p(x|q')} \quad \text{and} \quad p'_2(q) = \frac{p_2(q)p(x|q)}{\sum_{q'} p_2(q')p(x|q')}$$  \hspace{1cm} (17.38)

Also define $p_1(x) = \sum_q p(x|q)p_1(q)$. Then

$$\sum_x p_1(x)D(p'_1(q)||p'_2(q)) \leq D(p_1(q)||p_2(q))$$  \hspace{1cm} (17.39)

Note - subscripts are not time indices for the moment.

**Proof.**

$$\sum_x p_1(x) \sum_q \frac{p_1(q)p(x|q)}{\sum_{q'} p_1(q')p(x|q')} \log \left( \frac{p_1(q)p(x|q) \sum_{q'} p_2(q')p(x|q')}{p_2(q)p(x|q) \sum_{q'} p_1(q')p(x|q')} \right)$$  \hspace{1cm} (17.40)

$$= \sum_q p_1(q) \sum_x \frac{p_1(x)p(x|q)}{p_1(x)} \log \left( \frac{p_1(q)p_2(x)}{p_2(q)p_1(x)} \right)$$  \hspace{1cm} (17.41)

$$= \sum_{q,x} p_1(q)p(x|q) \left( \log \frac{p_1(q)}{p_2(q)} - \log \frac{p_1(x)}{p_2(x)} \right)$$  \hspace{1cm} (17.42)

$$= D(p_1(q)||p_2(q)) - D(p_1(x)||p_2(x)) \leq D(p_1(q)||p_2(q))$$  \hspace{1cm} (17.43)
In our case, this means that: Given
- prior state distributions \( p_{1:t,t+1}(q) \) and \( \hat{p}_{1:t,t+1}(q) \), which might be different
- and given transformation \( \mathcal{O}[] \),
- and given a model-based observation distribution

\[
p_{1:t,t+1}(x) \overset{\Delta}{=} \sum_{q} p_{1:t,t+1}(q)p(x|q) \tag{17.44}
\]

then we have

\[
\sum_{x} p_{1:t,t+1}(x) D(p_{1:t+1,t+1}(q)||\hat{p}_{1:t+1,t+1}(q)) \tag{17.45}
\]

\[
= \sum_{x} p_{1:t,t+1}(x) D(\mathcal{O}[p_{1:t,t+1}(q)]||\mathcal{O}[\hat{p}_{1:t,t+1}(q)]) \tag{17.46}
\]

\[
\leq D(p_{1:t,t+1}(q)||\hat{p}_{1:t,t+1}(q)) \tag{17.47}
\]

Therefore, what we have is that if \( \hat{p}_{1:t,t} \) and \( p_{1:t,t} \) differ in the KL-sense, then:

- after the transformation \( \mathcal{T} \), \( \hat{p}_{1:t,t+1} \) and \( p_{1:t,t+1} \) differ by no more than this amount and, hopefully, this difference decreases exponentially over time (see below), and
- after the transformation \( \mathcal{O} \), \( \hat{p}_{1:t+1,t+1} \) and \( p_{1:t+1,t+1} \) also differ by not as much in the expectation over observation \( x_t \), where the expectation is based on the prior prediction of \( x_t \) by the model just before \( x_t \) occurs.

2nd point is perhaps the weakest link in the theory, as model might be quite wrong about observations (it is only left-to-right).

However, as model becomes more accurate (and thus costly), predictions should improve.

Second issue is counterbalanced by first if exponential decay, so a momentary increase difference due to observations could damp down quickly.
Markov chains, KL-divergence, and rate

- So far, we assume all state variables over $D_Q$. Will be useful to allow different domains, so we have two random variables $Q$ and $R$, with $Q \in D_Q$ and $R \in D_R$. Could have $|D_Q| \neq |D_R|$.
- “time-homogeneous” stochastic transition matrix $p(r|q)$.
- Two initial distributions over $D_Q$, which are $p(q)$ and $\tilde{p}(q)$.
- Two final distributions over $D_R$, which are $p(r)$ and $\tilde{p}(r)$, with $p(r) = \sum_q p(q)p(r|q)$ and $\tilde{p}(r) = \sum_q \tilde{p}(q)p(r|q)$
- Assume that $p(r|q)$ is strictly positive (for simplicity).
- Worst case: $p(q)$ and $\tilde{p}(q)$ agree on nothing. E.g., $p(q) = \delta(q = q_1)$ and $\tilde{p}(q) = \delta(q = q_2)$ with $q_1 \neq q_2$. In such case, after transformation, we have $p(r) = p(r|q_1)$ and $\tilde{p}(r) = p(r|q_2)$, the corresponding matrix rows.
- Thus, now $p(r)$ and $\tilde{p}(r)$ “agree”, at least, on something. I.e., for $r \in D_R$, they agree on $r$ with mass $\min(p(r|q_1), p(r|q_2))$.

Want to characterize how quickly two chains “mix” (become more similar to each other) when applying stochastic transition matrix.

**Definition 17.7.3**

For Markov chain with stochastic transition matrix $A = [p(r|q)]$, the minimal mixing rate of $p(r|q)$ is defined as:

$$\gamma_A \triangleq \min_{q, q'} \sum_r \min(p(r|q), p(r|q'))$$

(17.48)
Markov chains, KL-divergence, and rate

\[ \gamma_A \triangleq \min_{q,q'} \sum_r \min(p(r|q), p(r|q')) \] (17.49)

- \( \gamma_A \) is the rate at which the chains mix, in the sense that at every step the KL-divergence will reduce by a factor at most \( 1 - \gamma_A \), which can be bounded away from 1 if the transition matrices are positive.

- This, therefore, is geometric convergence!

- \( \gamma_A = 1 \) (fastest rate) means same final distributions regardless of initial distributions.

- \( \gamma_A = 0 \), there exists (at least one) pair of states such that distributions will not mix.

- Let \([A_{qr}]_{q,r}\) be the transition matrix. I.e., \( p(r|q) = A_{q,r} \).

Contraction decomposition

\( A \) can be decomposed into a contraction component \( A^c \) and a residual \( A^n \) where \( p(q) \) and \( \tilde{p}(q) \) are rendered indistinguishable via \( A^c \).

**Lemma 17.7.4**

For any \( \gamma \leq \gamma_A \) and any \( p(q), \tilde{p}(q) \), matrix \( A \) admits an additive decomposition

\[ A = A^c + A^n \] (17.50)

where 1) both \( A^c \) and \( A^n \) are non-negative; 2) all rows of \( A^c \) sum to \( \gamma \); and 3) where, for all \( r \), we have:

\[ \sum_q p(q) A^c_{q,r} = \sum_q \tilde{p}(q) A^c_{q,r} \] (17.51)
Markov chains, KL-divergence, and rate

- Since $\sum_q p(q) A_{q,r}^a = \sum_q \tilde{p}(q) A_{q,r}^a$, then after application of $A$, at least a certain portion of the mass of $p(r)$ and $\tilde{p}(r)$ will agree.
- That is, for any given $r$, we have:

  $$p(r) = \sum_q p(q) A_{q,r} = \sum_q p(q) (A_{q,r}^a + A_{q,r}^n) \tag{17.52}$$

  $$= \sum_q p(q) A_{q,r}^a + \sum_q p(q) A_{q,r}^n \tag{17.53}$$

  $$= \sum_q \tilde{p}(q) A_{q,r}^a + \sum_q \tilde{p}(q) A_{q,r}^n \tag{17.54}$$

  and

  $$\tilde{p}(r) = \sum_q \tilde{p}(q) A_{q,r} \tag{17.55}$$

  $$= \sum_q \tilde{p}(q) A_{q,r}^a + \sum_q \tilde{p}(q) A_{q,r}^n \tag{17.56}$$

- Also, this happens with probability at least $\gamma$.

Theorem 17.7.5

Given transition matrix $A$, and distributions $p(q)$, $\tilde{p}(q)$, $p(r)$, and $\tilde{p}(r)$, we have

$$D(p(r)||\tilde{p}(r)) \leq (1 - \gamma_A) D(p(q)||\tilde{p}(q)) \tag{17.57}$$

where

$$\gamma_A \triangleq \min_{q,q'} \sum_r \min(p(r|q), p(r|q')) \tag{17.58}$$

- This is geometric convergence in the KL divergence.
HMM approximations

- Results so far are for Markov chains and HMMs in particular.
- In an HMM, \( \tilde{p}(q) \) can be approximation of \( p(q) \).
- Each time we make an approximation from \( p(q_t) \) to \( \tilde{p}(q_t) \), an error term, say \( \epsilon \), is introduced. The error is reduced by factor \( 1 - \gamma_A \) when we go to time \( t + 1 \) with \( p(q_{t+1}) \) to \( \tilde{p}(q_{t+1}) \).
- For pruning distributions, above results don’t apply since KL-divergence not defined when \( \tilde{p}(q) = 0 \) for some \( q \).
- But what computational benefit would this be?
- For HMMs, the results could apply, if instead of \( p(q_t) \) we compute \( \tilde{p}(q_t) \) (e.g., say we don’t compute all entries and use instead a fixed small value for every low probability event).
- Due to minimal structure in HMM, not too many options.
- Recall: key thing is we want large mixing rates, since in such case, any error introduced in going from \( p \) to \( \tilde{p} \) will diminish over time, and want rate that differences diminish to be \( \geq \) rate of error increase.

DGM case — from separator to separator

- Recall basic process:

\[
1 \text{ for } t \in \{1 \ldots T\} \text{ do} \\
2 \quad \phi(x_{S_{t+1}}) = \sum_{x_{C_{t+1}\setminus S_{t+1}}} \psi(x_{C_{t+1}}) \tilde{\phi}(x_{S_t}) \\
3 \quad \text{Project } \phi(x_{S_{t+1}}) \text{ down to more structured approximation } \tilde{\phi}(x_{S_{t+1}})
\]

- At each step we make approximations via projection.
- After going through the transformations from \( t \) to \( t + 1 \), the approximations decrease.
- Can the rate of error accumulation due to the approximations get damped out by the rate of difference decrease due to \( T \) and \( O \)?
What about DBNs?

- In the DBN case, $Q_t$ is a highly structured variable.
- For simplicity, let's assume that $Q_t = (R_t^{(1)}, R_t^{(2)}, \ldots, R_t^{(N)})$ and where there is graphical model structure over $R_t$.
- Example: $R_t$ might be the variables in the chunk, and $R_t$ and its left (or right) interface would be the modified chunk.
- We know that due to Rose’s theorem, all of $R_t$ are eventually (and quickly) connected, leading us back to $p(Q_t)$.
- Approximation $\tilde{p}(Q_t) = \tilde{p}(R_1, R_2, \ldots, R_N)$ would correspond to a factorization. This factorization has better memory properties and also results in fewer edges upon further elimination steps.
- Example: mean-field type factorization

$$\tilde{p}(Q_t) = \tilde{p}(R_1, R_2, \ldots, R_N) = \prod_{i=1}^{N} p(R_i)$$

Example: $N$ independent Markov chains, where each Markov chain is over binary states, with flip probability $\delta$

Each individual (local) Markov chain thus has local mixing rate $\gamma_A = 2 \min(\delta, (1 - \delta))$ which is fast for appropriate $\delta$.

Considered as a joint “compound” process, flatten everything down to a single Markov chain $Q_t = (R_t^{(1)}, R_t^{(2)}, \ldots, R_t^{(N)})$ with flattened transition matrix $p(q_t|q_{t-1})$.

Might the mixing rate of entire factorial chain then be quite fast as well?
What is mixing rate of this compound process?
Unfortunately, “compound” mixing rate of $R_t$, even though it comes from independent chains, is no more than $(4\delta)^{N/2}$.
This can be quite small (and thus have slow mixing).
Intuition:
- mixing coefficient $\gamma$ is worst case amount by which chains will “mix” together based on transition matrix.
- When there are $N$ independent variables, and we have two chains with opposite initial states, mixing rate is related to twice the probability that at least half of the variables flip, which is an exponentially small (in $N$) probability event.

Another example: assume $N$ chains and consider starting configuration $y_t = (0, 0, \ldots, 0)$ vs. $\hat{y}_t = (1, 1, \ldots, 1)$ (i.e., two distributions that put all mass on these two points).
Single step through flattened transition matrix diffuses mass around these starting points.
The ability for the diffusion to have significant overlap (or be identical) becomes in general exponentially difficult as $N$ grows in general.
DBN approximations - compound indep. chains

- More intuition for slow mixing.
- More intuition: Perhaps this is not too surprising - as we add states to a model, it has more memory, and therefore things may evolve more slowly (e.g., recall example ways of producing arbitrary length distributions in an HMM by adding states for either negative binomial or ladder/histogram distributions).
- Worst case is achieved when all the variables are coupled (i.e., it is one big Markov chain, e.g., a counter).

DBN approximations

- One thing we could do is assume that the true distribution decomposes well, leading to more independence, but this loss of structure: 1) assumes exactly what we don’t want to assume, i.e., we want the DGM to be highly structured and interacting, and 2) would not help, since the chains ultimately couple together in structure (due to Rose’s entanglement theorem).
- Hence, the overall mixing rate for a large stochastic process can be small, and this general idea might not work well.
- However, we also may make assumptions about the approximation \( \hat{p} \) (which is now a belief state over the structured state at time \( t \)).
- In particular, if the approximation we make \( \hat{p} \) matches the true process, then the mixing rate is not so bad after all.
- Note, here the mixing rate is not the mixing rate of the exact stochastic process, rather this is the rate at which the approximate and exact state approaches each other.
Theorem 17.7.6

Let \( R_1, R_2, \ldots, R_N \) be \( N \) independent Markov chains (subprocesses) and let \( \gamma_\ell \) be the mixing rate of \( R_\ell \) with \( \gamma = \min_\ell \gamma_\ell \). Let \( p(R_1, \ldots, R_N) \) be the true state distribution over the chains, and let \( \tilde{p}(R_1, \ldots, R_N) \) be an approximate one where the \( R_\ell \) processes are marginally independent. Let \( p' \) and \( \tilde{p}' \) correspond to the one-step evolution of the distributions (i.e., via \( T[O[\cdot]] \) as appropriate).

In such case (where the approximation matches the true factorization), we have that

\[
D(p' || \tilde{p}') \leq (1 - \gamma) D(p || \tilde{p})
\]

(17.59)

Now perhaps this is not too surprising, if the approximation matches the structure in the chains, we’ve recovered a good mixing rate (no slower than the worst of the individual chains).

But things are better than this, as if there is structure and interaction amongst chains, and the approximation matches the structure approximately, mixing rate isn’t horrible.
Theorem 17.7.7

Let $R_1, R_2, \ldots, R_N$ be $N$ independent Markov chains (subprocesses). Assume that each subprocess depends on at most $k_i$ others, and each influences at most $k_o$ others. Assume each has a mixing rate $\gamma_i \geq \gamma$. Let $p(R_1, \ldots, R_N)$ be the true state distribution over the chains, and let $\tilde{p}(R_1, \ldots, R_N)$ be an approximate one where the $R_\ell$ processes are marginally independent. Let $p'$ and $\tilde{p}'$ correspond to the one-step evolution of the distributions (i.e., via $T[O[]]$ as appropriate).

In such case we have that

$$D(p'||\tilde{p}') \leq (1 - \gamma^*)D(p||\tilde{p})$$

(17.60)

where $\gamma^* = (\gamma/k_i)^k_o$

So $\gamma^* = (\gamma/k_i)^k_o$.

If $k_i = k_o = 1$ then we’ve recovered the previous theorem.

As $k_i$ and $k_o$ get large, the all independence assumption becomes worse, and mixing rate decreases.

Hence, interaction and interconnectivity hurts the mixing rate.

Outgoing influence hurts mixing more since as degree of influence increases, less chance that that information will be “forgotten.”
DBN factor approximations

- So the process is similar to what we did in the HMM.
- We start with the “exact” distribution $p(Q_t)$ at time $t$.
- We project down to an approximate distribution $\tilde{p}(Q_t)$ where
  $\tilde{p}(Q_t) = \tilde{p}(R_1^t, R_2^t, \ldots, R_N^t) = \prod_{i=1}^{N} p(R_i)$, but other factorizations could be possible as well. We incur error $\epsilon$ from this projection.
- Update the distribution with transition structure and observations. The transition structure could be a highly structured network:

  ![Transition Structure Diagram]

  we get $\hat{p}(Q_{t+1}) = \mathcal{T}[\mathcal{O}[\tilde{p}(Q_t)]]$ with a mixing rate of $\gamma$.
- $\hat{p}(Q_{t+1})$ no longer has nice factorization structure, so we project it to $\hat{p}(Q_{t+1}) = \hat{p}(R_1^{t+1}, R_2^{t+1}, \ldots, R_N^{t+1}) = \prod_{i=1}^{N} \hat{p}(R_{i+1}^t)$ where $\hat{p}(R_{i+1}^t)$ is a marginal of $\hat{p}(Q_{t+1})$.
- Repeat this for increasing $t$:

Definition 17.7.8

The approximation $\tilde{p}$ of $\hat{p}$ incurs error $\epsilon$ relative to $p$ if

$$D(p||\tilde{p}) - D(p||\hat{p}) = \int p \log \frac{\hat{p}}{\tilde{p}} \leq \epsilon$$

(17.61)
At each step, have mixing rate $\gamma$ and error $\epsilon$.
Error is reduced to $(1 - \gamma)\epsilon$ due to mixing via transition structure, but we incur another error leading to $(1 - \gamma)\epsilon + \epsilon$
This continues, and the overall error of such a scheme becomes

$$\epsilon + (1 - \gamma)\epsilon + \cdots + (1 - \gamma)^{t-1}\epsilon < \epsilon \sum_{t} (1 - \gamma)^t = \epsilon / \gamma \tag{17.62}$$

So if $\gamma > \epsilon$, such a scheme can have quite a small overall error.
How do we quantify the error?

---

**Key theorem:**

**Theorem 17.7.9**

Given stochastic process $p$ with mixing rate $\gamma$, and we have approximation scheme that, at each point $t$, incurs error $\epsilon$ relative to $p$. Then for any $t$, we have total average error:

$$\sum_{x_{1:t}} p(x_{1:t}) \mathcal{D}(p_{1:t}(q) || \tilde{p}_{1:t}(q)) \leq \epsilon / \gamma \tag{17.63}$$

where the average is taken w.r.t. the process specific observation distribution $p(x_{1:t})$.

Recall: $p$ is the true state distribution, $\tilde{p}$ is our approximation (via factorization), and $\hat{p}$ is the approximation after going through transition structure.
DBN factor approximations

- Modifications: The factorization assumption on the separator can be in terms of a junction tree. i.e., build a junction tree on a triangulated version of the graph approximation of the separator.
- In such case, the “cliques of the separator” can be marginals, and we use the standard clique/separator form of factors of the junction tree nodes to produce sub-potentials for the separator clusters.
- This has the effect that the large junction tree in the clique is not coupled together nearly as much as if the separator had been completed in the first place.

Sources for Today’s Lecture

- “doc.pdf”