Today

• Simulation
  – More Gibbs
  – AA Algorithm
  – Backward Sampling (not)
• Structural Approximation
  – Boyen and Koller
  – Talk courtesy of Xavier Boyen,
    xb@cs.stanford.edu

Gibbs Sampling

• For each observed variable \( E_i \), set \( E_i = e_i \)
• Use any sampling technique (usually forward-sampling or importance-sampling) to set \( X_i \) to some random value.
• Repeat
  – Pick some unobserved variable \( X_i \)
  – Sample \( X_i[^j] \sim P(X_i|X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_n) \)
Transition Probability

- Transition Probability
  \[ P(X_t | X_{t-1}, \cdots, X_1) = P(X_t | MB(X_t)) \]
  \[ P(X_t | MB(X_t)) = \alpha P(X_t | \text{pa}(X_t)) \prod_{j \in \text{ch}(X_t)} P(X_j | \text{pa}(X_j), X_t) \]

Markov Chain

- Markov Chain Monte Carlo (MCMC)
- Transition Matrix
  \[ P(X_{t+1} | MB(X_t), t) \]
- Stationary distribution
  - Irreducible, Aperiodic
  - (Infinite state space) Positive recurrent.
- Convergence thought to be governed by rate of mixing:
  \[ \sup_{A,B} |P(\{x_{t+1} \in A, x_t \in B\}) - \pi(x_t \in A)\pi(x_{t+1} \in B)| \]
Procedures

- Parallel:
  - "Burn-in" by Gibbs for some period of time. Sample.
  - Sample is in P(X|E)
- Serial:
  - If x[i] is from P(X|E), then, so is x[i+1].
  - Say that x_1 is flipped:

\[
P(x[i + 1]) = \sum_{x_i} P(x'_i | x_2, \ldots, x_n) P(x_1, x_2, \ldots, x_n)
\]

\[
= \sum_{x_i} P(x'_i | x_2, \ldots, x_n) P(x_i | x_2, \ldots, x_n) P(x_1, \ldots, x_n)
\]

\[
= P(x'_1 | x_2, \ldots, x_n) P(x_1, \ldots, x_n) \sum_{x_i} P(x_i | x_2, \ldots, x_n)
\]

\[
= P(x'_1, x_2, \ldots, x_n)
\]

Gibbs Sampling Convergence

- Estimate average altitude of surfaces in room
- Statistics on samples suggest convergence, but really suggest only convergence on a given mode.
Simple Example

- \( P(X) = \[0.5, 0.5\] \)
- \( P(Y|X=T) = [1-a, a] \)
- \( P(Y|X=F) = [a, 1-a] \)

Transitions:

\[
\begin{align*}
P(X'|= T | Y = T) &= \frac{0.5(1-a)}{0.5a + 0.5(1-a)} = 1-a \\
P(Y'|= T | X = T) &= 1-a
\end{align*}
\]

Simple Gibbs Example

- Suppose \((X,Y) = (T, T)\)
- Sample \(X Y X Y X Y\)
- How long until we sample something other than \((T, T)\)?
  - \(1/a\)
Bad Convergence

MCMC

Change to another mode

Likelihood Weighting

Found 1 sample with high likelihood

Bounded Variance Algorithm

\[
upper(X_i = x_i) = \max_{\text{paths}(X_i)} P\{X_i = x_i \mid x_{pa}\}
\]

\[
BV[k, \delta, X_i, X, E] = S^* = 4(e - 2)\ln\left(\frac{2}{\delta}\right) + \frac{6}{e^2} + \frac{\delta}{e}
\]

\[
K_E = \prod_{i \in E} \text{upper}(X_i)
\]

\[
\mu_E = \text{LW_cumu} \{X, E, S^*, K_E\}
\]

\[
K_i = \text{upper}(X_i = x_i)K_E
\]

\[
\mu_i = \text{LW_cumu} \{X \setminus \{x_i\}, E \cup \{x_i\}, S^*, K_i\}
\]

return \((\mu_i, \mu_E)\)

\[
\text{LW_cumu}\{Z, E, S^*, K\}
\]

\[
S = 0
\]

\[
N = 0
\]

repeat while \(S < S^*\)

\[
Z[i] = \text{Forward_Sample}(Z, E, 1)
\]

\[
\phi[i] = \text{path likelihood}(Z[i])
\]

\[
S \leftarrow S + \frac{\phi[i]}{K}
\]

\[
N \leftarrow N + 1
\]

return \(\frac{KS}{N}\)
AA Algorithm

Estimate means
\[
(\mu_i, \mu_E) = B\{e^\frac{1}{\delta}, \delta = \frac{\delta}{3}, X_i, X, E\}
\]

Estimate variance
\[
Y = 8(e - 2)\log\left(\frac{2}{\delta}\right)\frac{1}{\delta^2}
\]
\[
N_E = \frac{Y\epsilon}{\mu_E}, \quad N_j = \frac{Y\epsilon}{\mu_j}
\]
\[
Z_E[1...N] = F\_S(X, E, N_E)
\]
\[
\sigma_E^2 = \min\{\text{var}(\lambda(Z_E)), e\mu_E\}
\]
\[
Z_j[1...N] = F\_S(X \setminus X_j, E \cup X_j, N_E)
\]
\[
\sigma_j^2 = \min\{\text{var}(\lambda(Z_j)), e\mu_i\}
\]

Simulate Query
\[
N_{E2} = \frac{Y\sigma_E^2}{\mu_E^2}, \quad N_{j2} = \frac{Y\sigma_j^2}{\mu_j^2}
\]
\[
Z_{E2}[1...N] = F\_S(X, E, N_{E2})
\]
\[
Z_{j2}[1...N] = F\_S(X \setminus X_j, E \cup X_j, N_{E2})
\]
\[
\rho(X_i | E) = \frac{\sum_{k=1}^{N_{j2}} \lambda(Z_{j2}[k])}{\sum_{k=1}^{N_{E2}} \lambda(Z_{E2}[k])}
\]

Theorem: Within a small constant of the optimal number of samples!

Tractable Inference for Complex Stochastic Processes

Xavier Boyen
Daphne Koller
Stanford University
Stochastic dynamic system

- Process evolves over time
- Dynamics are noisy or unpredictable
- Process state only partially observable

Monitoring

- Task: Online monitoring of current process state, based on observations obtained until now.

- Exact state is unknown ⇒ maintain distribution $\phi(t)$ over possible states at time $t$
Monitoring in theory

Maintaining belief state is easy (in theory):

\[ \varphi^{(t+1)} \]

Propagate to next time slice, using process dynamics.

\[ \varphi^{(t)} \]

Update to account for new observation, using Bayesian conditioning.

Monitoring in practice

• Sometimes, belief state admits compact representation & manipulation
  – e.g., Kalman filters
  \Rightarrow supports effective monitoring algorithm.

• Is this the case for other, more expressive representations?
  – dynamic Bayesian networks (DBNs);
  – hybrid processes (discrete + continuous).
Dynamic Bayesian networks

- For DBNs, belief state = distribution over all possible assignments to state variables
  ⇒ Explicit belief state is exponential in #variables
- But, surely we can exploit the structure, “as usual”?
  Structure ⇒ fast inference ... (??)

No, it’s a myth!

- Problem: even in highly structured DBNs, belief state variables become fully correlated
  DBN structure ⊄ belief state structure

Not even conditionally independent (within this time slice)
Approximate monitoring

• Exact inference in DBNs is intractable !!
  ⇒ Major barrier to DBN inference in realistic networks.

• **Idea:** Maintain approximate belief state

  • Choose some predetermined subspace $C$ of compactly representable distributions, e.g.:
    – Gaussian mixtures with few components;
    – decomposable distributions.

**Approximate monitoring**

• **propagate** current belief $\rightarrow$ expected next belief
• **condition** on evidence $\rightarrow$ next belief
• **project** onto compact distribution in $C$.

**Benefit:** belief state representation remains small
**Problem:** error might grow unboundedly
- previous errors accumulate
- each stage introduces new error

**Opposing forces**
- Conditioning on observation usually brings distributions closer
- Stochastic transition causes ‘smearing’, resulting in error reduction
- Approximation causes error to increase

True distribution 
Our approximation
**Contraction & approximation**

If:
- propagation through transition model decreases error by $\gamma$ (contraction property);
- observations decrease error on expectation;
- approximation error at each stage bounded by $\varepsilon$;

Then, total expected error is, at all times:

$$\leq \varepsilon + (1-\gamma)\varepsilon + (1-\gamma)^2\varepsilon + \ldots \leq \varepsilon / \gamma$$

**Contraction**

Transition Matrix $P_{X[t+1]|X[t]}$

Mixing $\gamma = \min_{s_{i(0)}=t} \left( \sum_{s_{0}, \Omega, \theta_{j(0)}} \min_{s_{0}} P_{s_{j(0)}|s_{i(0)}} P_{s_{i(0)}|s_{j(0)}} \right)$

$\Omega = X_1 \times \ldots \times X_N$

KL $D[\phi || \psi] = \sum_{x \in \Omega} \phi(x) \ln \frac{\phi(x)}{\psi(x)}$

Contraction $D[\phi || \psi'] \leq (1-\gamma) D[\phi || \psi']$
Factored processes

• Problem: Contraction rate $\gamma$ for large processes can be exponentially low. (even if structured !)

• Clever design of approximation scheme can exploit process structure, if:
  – process is composed of weakly interacting subprocesses
  – our approximation decomposes the belief state according to these subprocesses.

Theorem

If each subprocess
  • depends on at most $r$ others,
  • influences at most $q$ others,
  • contracts at rate at least $\gamma$,
Then,

$$\gamma_{Whole} \geq \left(\frac{\gamma}{p}\right)^q$$

Thus, interactions between processes are costly:
  – Incoming influences reduce contraction rate linearly
  – Outgoing influences reduce it exponentially.

E.g., for $A$,
  $r = 3$, $q = 2$. 

• So, the DBN structure can be exploited for *approximate* inference!

• We partition state variables into “subprocesses”
  – maintain approximate belief state as product of marginally independent “sub-beliefs”;
  – update belief and project back using junction trees.

1. Start from current approximate belief.
1. Start from current approximate belief.
2. Want to compute next belief.
3. Create + calibrate clique tree.
1. Start from current approximate belief.
2. Want to compute next belief.
3. Create + calibrate clique tree.
4. Extract components of next belief by marginalization.
5. Continue with next belief ...
DBN Algorithm: discussion

- Smaller partitions ⇒
  + faster inference
  + better contraction
  - worse approximation

- Better contraction & approximation if partition along weak/sparse interactions.

- Very simple procedure.

- First algorithm to provably exploit structure in general DBNs.

Experimental setup

- Used BAT network for freeway traffic monitoring [Forbes et al.]
- Maintained approximate belief states given evidence sequences generated from network.
Typical evolution of error

Comparing partitions
Experimental results

<table>
<thead>
<tr>
<th>Network</th>
<th>Max error</th>
<th>Avg error</th>
<th>Speedup</th>
</tr>
</thead>
<tbody>
<tr>
<td>BAT</td>
<td>0.065</td>
<td>0.0007</td>
<td>15</td>
</tr>
<tr>
<td></td>
<td>0.02</td>
<td>0.0013</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0.07</td>
<td>0.0019</td>
<td></td>
</tr>
<tr>
<td>WATER</td>
<td>0.14</td>
<td>0.06</td>
<td>31</td>
</tr>
<tr>
<td></td>
<td>0.018</td>
<td>0.0015</td>
<td>13</td>
</tr>
</tbody>
</table>

- Error remains bounded indefinitely
- Good partitions help, as predicted
- Conditionally independent belief states are even better.

Conclusions

- Simple fast approximate inference in dynamic systems
  - Guaranteed error bounds.
  - Uses a new contraction property for relative entropy.
- Achieves orders of magnitude speedup for DBNs at minor cost in accuracy.
  - Savings dramatically larger for more complex processes.
  - First algorithm to exploit structure of general DBNs.
  - Practical inference algorithm for large DBNs!
- Major step towards reasoning about complex, real-life dynamic systems.
More Conclusions

• Doesn’t matter what the approximation algorithm is…
  – If contraction holds under the algorithm, approximation has bounded error.