Markov Models

- A Markov model is a chain-structured BN
  - Each node is identically distributed (stationarity)
  - Value of X at a given time is called the state
  - As a BN:

```
X_1 ----> X_2 ----> X_3 ----> X_4 ---->
```

\[ P(X_1) \quad P(X|X_{-1}) \]

- Parameters: called transition probabilities or dynamics, specify how the state evolves over time (also, initial probs)
Mini-Forward Algorithm

- Better way: cached incremental belief updates

\[ P(x_t) = \sum_{x_{t-1}} P(x_t|x_{t-1})P(x_{t-1}) \]

\[ P(x_1) = \text{known} \]

---

Example

- From initial observation of sun

\[
\begin{align*}
0.9 &\quad 0.82 &\quad 0.5 \\
0.1 &\quad 0.18 &\quad 0.5
\end{align*}
\]

\[
P(X_1) \quad P(X_2) \quad P(X_3) \quad P(X_\infty)
\]

- From initial observation of rain

\[
\begin{align*}
0.1 &\quad 0.18 &\quad 0.5 \\
1.0 &\quad 0.9 &\quad 0.82
\end{align*}
\]

\[
P(X_1) \quad P(X_2) \quad P(X_3) \quad P(X_\infty)
\]
Stationary Distributions

- If we simulate the chain long enough:
  - What happens?
  - Uncertainty accumulates
  - Eventually, we have no idea what the state is!

- Stationary distributions:
  - For most chains, the distribution we end up in is independent of the initial distribution
  - Called the stationary distribution of the chain
  - Usually, can only predict a short time out

Web Link Analysis

- PageRank over a web graph
  - Each web page is a state
  - Initial distribution: uniform over pages
  - Transitions:
    - With prob. c, uniform jump to a random page (dotted lines)
    - With prob. 1-c, follow a random outlink (solid lines)

- Stationary distribution
  - Will spend more time on highly reachable pages
  - E.g. many ways to get to the Acrobat Reader download page!
  - Somewhat robust to link spam
  - Google 1.0 returned the set of pages containing all your keywords in decreasing rank, now all search engines use link analysis along with many other factors
Hidden Markov Models

- Markov chains not so useful for most agents
  - Eventually you don’t know anything anymore
  - Need observations to update your beliefs

- Hidden Markov models (HMMs)
  - Underlying Markov chain over states S
  - You observe outputs (effects) at each time step
  - As a Bayes’ net:

```
X_1 -> X_2 -> X_3 -> X_4
  |                     
  |                      
  E_1  E_2  E_3  E_4
```

Example

- An HMM is
  - Initial distribution: \( P(X_1) \)
  - Transitions: \( P(X|X_{-1}) \)
  - Emissions: \( P(E|X) \)
Conditional Independence

- HMMs have two important independence properties:
  - Markov hidden process, future depends on past via the present
  - Current observation independent of all else given current state

- Quiz: does this mean that observations are independent given no evidence?
  - [No, correlated by the hidden state]

Real HMM Examples

- Speech recognition HMMs:
  - Observations are acoustic signals (continuous valued)
  - States are specific positions in specific words (so, tens of thousands)

- Machine translation HMMs:
  - Observations are words (tens of thousands)
  - States are translation positions (dozens)

- Robot tracking:
  - Observations are range readings (continuous)
  - States are positions on a map (continuous)
Filtering / Monitoring

- Filtering, or monitoring, is the task of tracking the distribution $B(X)$ (the belief state)
- We start with $B(X)$ in an initial setting, usually uniform
- As time passes, or we get observations, we update $B(X)$

Example: Robot Localization

Sensor model: never more than 1 mistake
Motion model: may not execute action with small prob.
Example: Robot Localization

t=1

Example: Robot Localization

t=2
Example: Robot Localization

Example: Robot Localization

\[ t=3 \]

\[ t=4 \]
Passage of Time

- Assume we have current belief \( P(X | \text{evidence to date}) \)
  
  \[ B(X) = P(X_t | e_{1:t}) \]

- Then, after one time step passes:
  
  \[ P(X_{t+1} | e_{1:t}) = \sum_{x_t} P(X_{t+1} | x_t) P(x_t | e_{1:t}) \]

- Or, compactly:
  
  \[ B'(X) = \sum_{x_t} P(X' | x) B(x) \]

- Basic idea: beliefs get "pushed" through the transitions
  - With the “B” notation, we have to be careful about what time step the belief is about, and what evidence it includes
Example: Passage of Time

- As time passes, uncertainty “accumulates”

\[
B'(X) = \sum_x P(X'|x)B(x)
\]

Transition model: ships usually go clockwise

Observation

- Assume we have current belief \(P(X | \text{previous evidence})\):

\[
B'(X) = P(X_t|e_{1:t-1})
\]

- Then:

\[
P(X_t|e_{1:t}) \propto P(e_t|X_t)P(X_t|e_{1:t-1})
\]

- Or:

\[
B(X) \propto P(e|X)B'(X)
\]

- Basic idea: beliefs reweighted by likelihood of evidence

- Unlike passage of time, we have to renormalize
Example: Observation

- As we get observations, beliefs get reweighted, uncertainty “decreases”

\[
B(X) \propto P(e|X)B'(X)
\]

Example with HMM
**Example with HMM**

- Every time step, we start with current $P(X \mid \text{evidence})$
- We must update for time:

\[ P(X_t|e_{1:t-1}) \propto \sum_{x_{t-1}} P(X_t|x_{t-1})P(x_{t-1}|e_{1:t-1}) \]

- We must update for observation:

\[ P(X_t|e_{1:t}) \propto P(e_t|X_t)P(X_t|e_{1:t-1}) \]

- So, linear in time steps, quadratic in number of states $|X|$
- Of course, can do both at once, too
The Forward Algorithm

- Can do belief propagation exactly as in previous slides, renormalizing each time step.
- In the standard forward algorithm, we actually calculate $P(X,e)$, without normalizing.

\[
P(x_t|e_{1:t}) \propto P(x_t, e_{1:t})
\]
\[
= \sum_{x_{t-1}} P(x_{t-1}, x_t, e_{1:t})
\]
\[
= \sum_{x_{t-1}} P(x_{t-1}, e_{1:t-1}) P(x_t|x_{t-1}) P(e_t|x_t)
\]
\[
= P(e_t|x_t) \sum_{x_{t-1}} P(x_t|x_{t-1}) P(x_{t-1}, e_{1:t-1})
\]

Particle Filtering

- Sometimes $|X|$ is too big to use exact inference:
  - $|X|$ may be too big to even store $B(X)$
  - E.g. $X$ is continuous
  - $|X|^2$ may be too big to do updates

- Solution: approximate inference:
  - Track samples of $X$, not all values
  - Time per step is linear in the number of samples
  - But: number needed may be large

- This is how robot localization works in practice
Particle Filtering: Time

- Each particle is moved by sampling its next position from the transition model:
  \[ x' = \text{sample}(P(X'|x)) \]
  - This is like prior sampling – samples are their own weights
  - Here, most samples move clockwise, but some move in another direction or stay in place
  - This captures the passage of time
    - If we have enough samples, close to the exact values before and after (consistent)

Particle Filtering: Observation

- Slightly trickier:
  - We don’t sample the observation, we fix it
  - This is similar to likelihood weighting, so we downweight our samples based on the evidence:
    \[ w(x) = P(e|x) \]
    \[ B(X) \propto P(e|X)B'(X) \]
  - Note that, as before, the probabilities don’t sum to one, since most have been downweighted (they sum to an approximation of \( P(e) \))
Particle Filtering: Resampling

- Rather than tracking weighted samples, we resample
- $N$ times, we choose from our weighted sample distribution (i.e. draw with replacement)
- This is equivalent to renormalizing the distribution
- Now the update is complete for this time step, continue with the next one

Robot Localization

- In robot localization:
  - We know the map, but not the robot’s position
  - Observations may be vectors of range finder readings
  - State space and readings are typically continuous (works basically like a very fine grid) and so we cannot store $B(X)$
  - Particle filtering is a main technique

[Demos]
SLAM

- SLAM = Simultaneous Localization And Mapping
  - We do not know the map or our location
  - Our belief state is over maps and positions!
  - Main techniques: Kalman filtering (Gaussian HMMs) and particle methods

- [DEMOS]