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 → X_N
```

```
E_1 → E_2 → E_3 → E_4 → E_N
```
Example

- An HMM is defined by:
  - Initial distribution: \( P(X_1) \)
  - Transitions: \( P(X_t | X_{t-1}) \)
  - Emissions: \( P(E | X) \)
Hidden Markov Models

- Defines a joint probability distribution:

\[ P(X_1, \ldots, X_n, E_1, \ldots, E_n) = \]

\[ P(X_{1:n}, E_{1:n}) = \]

\[ P(X_1)P(E_1|X_1) \prod_{t=2}^{N} P(X_t|X_{t-1})P(E_t|X_t) \]
Ghostbusters HMM

- $P(X_1) = \text{uniform}$
- $P(X' \mid X) = \text{ghosts usually move clockwise, but sometimes move in a random direction or stay put}$
- $P(E \mid X) = \text{same sensor model as before: red means close, green means far away.}$

<table>
<thead>
<tr>
<th>$P(X_1)$</th>
<th>$1/9$</th>
<th>$1/9$</th>
<th>$1/9$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P(X_1)$</td>
<td>$1/9$</td>
<td>$1/9$</td>
<td>$1/9$</td>
</tr>
<tr>
<td>$P(X_1)$</td>
<td>$1/9$</td>
<td>$1/9$</td>
<td>$1/9$</td>
</tr>
</tbody>
</table>

$P(X' \mid X = \langle 1,2 \rangle)$

<table>
<thead>
<tr>
<th>$P(X' \mid X = \langle 1,2 \rangle)$</th>
<th>$1/6$</th>
<th>$1/6$</th>
<th>$1/2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P(X' \mid X = \langle 1,2 \rangle)$</td>
<td>$0$</td>
<td>$1/6$</td>
<td>$0$</td>
</tr>
<tr>
<td>$P(X' \mid X = \langle 1,2 \rangle)$</td>
<td>$0$</td>
<td>$0$</td>
<td>$0$</td>
</tr>
</tbody>
</table>

$P(E \mid X)$

<table>
<thead>
<tr>
<th>$P(E \mid X)$</th>
<th>$P(\text{red} \mid 3)$</th>
<th>$P(\text{orange} \mid 3)$</th>
<th>$P(\text{yellow} \mid 3)$</th>
<th>$P(\text{green} \mid 3)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P(E \mid X)$</td>
<td>$0.05$</td>
<td>$0.15$</td>
<td>$0.5$</td>
<td>$0.3$</td>
</tr>
</tbody>
</table>

Etc… (must specify for other distances)
HMM Computations

- Given
  - parameters
  - evidence $E_{1:n} = e_{1:n}$

- Inference problems include:
  - **Filtering**, find $P(X_t|e_{1:t})$ for all $t$
  - **Smoothing**, find $P(X_t|e_{1:n})$ for all $t$
  - **Most probable explanation**, find
    $$x_{1:n}^* = \arg\max_{x_{1:n}} P(x_{1:n}|e_{1:n})$$
Real HMM Examples

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

- **Machine translation HMMs:**
  - Observations are words (tens of thousands)
  - States are translation options
Real HMM Examples

- Robot tracking:
  - Observations are range readings (continuous)
  - States are positions on a map (continuous)
HMMs have two important independence properties:
- Markov hidden process, future depends on past via the present
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
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]
Filtering / Monitoring

- Filtering, or monitoring, is the task of tracking the distribution $B(X)$ (the belief state) over time.
- We start with $B(X)$ in an initial setting, usually uniform.
- As time passes, or we get observations, we update $B(X)$.
- The Kalman filter (one method – Real valued values)
  - invented in the 60’s as a method of trajectory estimation for the Apollo program.
Example: Robot Localization

Sensor model: can read in which directions there is a wall, never more than 1 mistake
Motion model: may not execute action with small prob.
Example: Robot Localization

Lighter grey: was possible to get the reading, but less likely b/c required 1 mistake
Example: Robot Localization

t=2
Example: Robot Localization

\[ t=3 \]
Example: Robot Localization

\[ t=4 \]
Example: Robot Localization

\[ t=5 \]
Inference Recap: Simple Cases

\[
P(X_1|e_1)
\]

\[
P(x_1|e_1) = \frac{P(x_1, e_1)}{P(e_1)}
\]
\[
\propto_{X_1} P(x_1, e_1)
\]
\[
= P(x_1)P(e_1|x_1)
\]

\[
P(x_2) = \sum_{x_1} P(x_1, x_2)
\]
\[
= \sum_{x_1} P(x_1)P(x_2|x_1)
\]
Online Belief Updates

- Every time step, we start with current $P(X \mid \text{evidence})$
- We update for time:
  \[ P(x_t|e_{1:t-1}) = \sum_{x_{t-1}} P(x_{t-1}|e_{1:t-1}) \cdot P(x_t|x_{t-1}) \]
- We update for evidence:
  \[ P(x_t|e_{1:t}) \propto_x P(x_{t}|e_{1:t-1}) \cdot P(e_t|x_t) \]
- The forward algorithm does both at once (and doesn’t normalize)
- Problem: space is $|X|$ and time is $|X|^2$ per time step
Passage of Time

- Assume we have current belief \( P(X \mid \text{evidence to date}) \)

\[
B(X_t) = P(X_t|e_{1:t})
\]

- 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:

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

- Basic idea: beliefs get “pushed” through the transitions
  - With the “B” notation, we have to be careful about what time step \( t \) 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: ghosts usually go clockwise
Observation

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

\[
B'(X_{t+1}) = P(X_{t+1} \mid e_{1:t})
\]

- Then:

\[
P(X_{t+1} \mid e_{1:t+1}) \propto P(e_{t+1} \mid X_{t+1}) P(X_{t+1} \mid e_{1:t})
\]

- Or:

\[
B(X_{t+1}) \propto P(e \mid X) B'(X_{t+1})
\]

- 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”

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

- We want to know: $B_t(X) = P(X_t|e_{1:t})$
- We can derive the following updates

$$P(x_t|e_{1:t}) \propto_X P(x_t, e_{1:t})$$

$$= \sum_{x_{t-1}} P(x_{t-1}, x_t, e_{1:t})$$

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

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

- To get $B_t(X)$ compute each entry and normalize
Example: Run the Filter

An HMM is defined by:

- Initial distribution:
  \[ P(X_1) \]
  \[ P(X_t | X_{t-1}) \]

- Transitions:

- Emissions:
  \[ P(E | X) \]
Example HMM
Example Pac-man
Filtering is the inference process of finding a distribution over $X_T$ given $e_1$ through $e_T$:

$$P(X_T | e_{1:t})$$

We first compute $P(X_1 | e_1)$:

For each $t$ from 2 to $T$, we have $P(X_{t-1} | e_{1:t-1})$

**Elapse time:** compute $P(X_t | e_{1:t-1})$

$$P(x_t | e_{1:t-1}) = \sum_{x_{t-1}} P(x_{t-1} | e_{1:t-1}) \cdot P(x_t | x_{t-1})$$

**Observe:** compute $P(x_t | e_{1:t-1}, e_t) = P(x_t | e_{1:t})$

$$P(x_t | e_{1:t}) \propto P(x_t | e_{1:t-1}) \cdot P(e_t | x_t)$$
Recap: Reasoning Over Time

- **Stationary Markov models**

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

- **Hidden Markov models**

  \[
  \begin{array}{c|c|c|c|c}
    \text{X} & \text{E} & \text{P} \\
    \hline
    \text{rain} & \text{umbrella} & 0.9 \\
    \text{rain} & \text{no umbrella} & 0.1 \\
    \text{sun} & \text{umbrella} & 0.2 \\
    \text{sun} & \text{no umbrella} & 0.8 \\
  \end{array}
  \]
Recap: Filtering

- **Elapse time**: compute $P( X_t | e_{1:t-1} )$
  
  $$P(x_t | e_{1:t-1}) = \sum_{x_{t-1}} P(x_{t-1} | e_{1:t-1}) \cdot P(x_t | x_{t-1})$$

- **Observe**: compute $P( X_t | e_{1:t} )$
  
  $$P(x_t | e_{1:t}) \propto P(x_t | e_{1:t-1}) \cdot P(e_t | x_t)$$

Belief: $<P(\text{rain}), P(\text{sun})>$

- $P(X_1) <0.5, 0.5>$ Prior on $X_1$
- $P(X_1 | E_1 = \text{umbrella}) <0.82, 0.18>$ Observe
- $P(X_2 | E_1 = \text{umbrella}) <0.63, 0.37>$ Elapse time
- $P(X_2 | E_1 = \text{umb}, E_2 = \text{umb}) <0.88, 0.12>$ Observe
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
  - Samples are called particles
  - Time per step is linear in the number of samples
  - But: number needed may be large
  - In memory: list of particles, not states

- This is how robot localization works in practice
Our representation of $P(X)$ is now a list of $N$ particles (samples)
- Generally, $N \ll |X|$
- Storing map from $X$ to counts would defeat the point

$P(x)$ approximated by number of particles with value $x$
- So, many $x$ will have $P(x) = 0!$
- More particles, more accuracy

For now, all particles have a weight of 1
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’ frequencies reflect the transition probs
- 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: Observe

- Slightly trickier:
  - Don’t do rejection sampling (why not?)
  - We don’t sample the observation, we fix it
  - This is similar to likelihood weighting, so we downweight our samples based on the evidence

\[
\begin{align*}
    w(x) &= P(e|x) \\
    B(X) &\propto P(e|X)B'(X)
\end{align*}
\]

- Note that, as before, the probabilities don’t sum to one, since most have been downweighted (in fact they sum to an approximation of P(e))
Particle Filtering: Resample

- 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.

Old Particles:
- (3,3) w=0.1
- (2,1) w=0.9
- (2,1) w=0.9
- (3,1) w=0.4
- (3,2) w=0.3
- (2,2) w=0.4
- (1,1) w=0.4
- (3,1) w=0.4
- (2,1) w=0.9
- (3,2) w=0.3

New Particles:
- (2,1) w=1
- (2,1) w=1
- (2,1) w=1
- (3,2) w=1
- (2,2) w=1
- (2,1) w=1
- (1,1) w=1
- (1,1) w=1
- (3,1) w=1
- (2,1) w=1
- (1,1) w=1
Recap: Particle Filtering

At each time step $t$, we have a set of $N$ particles / samples

- **Initialization**: Sample from prior, reweight and resample
- **Three step procedure**, to move to time $t+1$:
  1. **Sample transitions**: for each particle $x$, sample next state

$$ x' \leftarrow \text{sample}(P(X'|x)) $$

  2. **Reweight**: for each particle, compute its weight given the actual observation $e$

$$ w(x) \leftarrow P(e|x) $$

  3. **Resample** and sample $N$ new particles from the resulting distribution over states
Particle Filtering Summary

- Represent current belief $P(X \mid \text{evidence to date})$ as set of $n$ samples (actual assignments $X=x$)
- For each new observation $e$:
  1. Sample transition, once for each current particle $x$
     \[ x' = \text{sample}(P(X' \mid x)) \]
  2. For each new sample $x'$, compute importance weights for the new evidence $e$:
     \[ w(x') = P(e \mid x') \]
  3. Finally, normalize the importance weights and resample $N$ new particles
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
Robot Localization

QuickTime™ and a GIF decompressor are needed to see this picture.
Which Algorithm?

Exact filter, uniform initial beliefs
Which Algorithm?

Particle filter, uniform initial beliefs, 300 particles
Which Algorithm?

Particle filter, uniform initial beliefs, 25 particles
P4: Ghostbusters

- **Plot:** Pacman's grandfather, Grandpac, learned to hunt ghosts for sport.

- He was blinded by his power, but could hear the ghosts’ banging and clanging.

- **Transition Model:** All ghosts move randomly, but are sometimes biased.

- **Emission Model:** Pacman knows a “noisy” distance to each ghost.

Noisy distance prob

<table>
<thead>
<tr>
<th>True distance = 8</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
</tr>
<tr>
<td>13</td>
</tr>
<tr>
<td>11</td>
</tr>
<tr>
<td>9</td>
</tr>
<tr>
<td>7</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>1</td>
</tr>
</tbody>
</table>
Dynamic Bayes Nets (DBNs)

- We want to track multiple variables over time, using multiple sources of evidence
- Idea: Repeat a fixed Bayes net structure at each time
- Variables from time $t$ can condition on those from $t-1$

- Discrete valued Dynamic Bayes Nets are also HMMs
Exact Inference in DBNs

- Variable elimination applies to dynamic Bayes nets
- Procedure: “unroll” the network for $T$ time steps, then eliminate variables until $P(X_T | e_{1:T})$ is computed

- Online belief updates: Eliminate all variables from the previous time step; store factors for current time only.
DBN Particle Filters

- A particle is a complete sample for a time step
- **Initialize**: Generate prior samples for the t=1 Bayes net
  - Example particle: \( G_1^a = (3,3) \ G_1^b = (5,3) \)
- **Elapse time**: Sample a successor for each particle
  - Example successor: \( G_2^a = (2,3) \ G_2^b = (6,3) \)
- **Observe**: Weight each entire sample by the likelihood of the evidence conditioned on the sample
  - Likelihood: \( P(E_1^a \mid G_1^a) \ast P(E_1^b \mid G_1^b) \)
- **Resample**: Select prior samples (tuples of values) in proportion to their likelihood
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]

DP-SLAM, Ron Parr
Best Explanation Queries

- Query: most likely seq:

\[
\arg\max_{x_{1:t}} P(x_{1:t}|e_{1:t})
\]
State Path Trellis

- State trellis: graph of states and transitions over time

### Diagram

- Each arc represents some transition
- Each arc has weight
- Each path is a sequence of states
- The product of weights on a path is the seq’s probability
- Can think of the Forward (and now Viterbi) algorithms as computing sums of all paths (best paths) in this graph

\[ x_{t-1} \rightarrow x_t \]
\[ P(x_t|x_{t-1})P(e_t|x_t) \]
Viterbi Algorithm

\[ x^*_{1:T} = \arg \max_{x_{1:T}} P(x_{1:T} | e_{1:T}) = \arg \max_{x_{1:T}} P(x_{1:T}, e_{1:T}) \]

\[ m_t[x_t] = \max_{x_{1:t-1}} P(x_{1:t-1}, x_t, e_{1:t}) \]

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

\[ = P(e_t | x_t) \max_{x_{t-1}} P(x_t | x_{t-1}) \max_{x_{1:t-2}} P(x_{1:t-1}, e_{1:t-1}) \]

\[ = P(e_t | x_t) \max_{x_{t-1}} P(x_t | x_{t-1})m_{t-1}[x_{t-1}] \]
Example

Rain_1
true
false

Rain_2
true
false

Rain_3
true
false

Rain_4
true
false

Rain_5
true
false

state space paths
true
false

umbrella
true
false

most likely paths
.8182
.5155
.0361
.0334
.0210

.1818
.0491
.1237
.0173
.0024

m_{1:1}
m_{1:2}
m_{1:3}
m_{1:4}
m_{1:5}