Today.

Continuing Hidden Markov Models!
Hidden Markov Models
Video of Demo Pacman – Sonar (no beliefs)
Hidden Markov Models

Markov chains not so useful for most agents
- Need observations to update your beliefs

Hidden Markov models (HMMs)
- Underlying Markov chain over states $X$
- You observe outputs (effects) at each time step
Example: Weather HMM

An HMM is defined by:

- Initial distribution: $P(X_1)$
- Transitions: $P(X_t | X_{t-1})$
- Emissions: $P(E_t | X_t)$
Example: Ghostbusters HMM

\[ P(X_1) = \text{uniform} \]

\[ P(X|X') = \text{usually move clockwise, but sometimes move in random direction or stay in place} \]

\[ P(R_{ij}|X) = \text{same sensor model as before: red means close, green/yellow means farther away.} \]

[Demo: Ghostbusters – Circular Dynamics – HMM (L14D2)]
Video of Demo Ghostbusters – Circular Dynamics – HMM
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

Are $E_1$ and $E_3$ independent now? No!!
Need to condition on $X_1$, or $X_2$ or $X_3$. 
Real HMM Examples

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

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 options
Filtering, or monitoring, is the task of tracking the distribution $B_t(X) = P_t(X_t | e_1, \ldots, e_t)$ (the belief state) over time.

We start with $B_1(X)$ in an initial setting, usually uniform.

As time passes, or we get observations, we update $B(X)$.

The Kalman filter was invented in the 60’s and first implemented 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 from Michael Pfeiffer
Example: Robot Localization

Lighter grey: was possible to get the reading, but less likely b/c required 1 mistake
Inference: Find State Given Evidence

We are given evidence at each time and want to know

$$B_t(X) = P(X_t|e_{1:t})$$

Idea: start with $P(X_1)$ and derive $B_t$ in terms of $B_{t-1}$

- equivalently, derive $B_{t+1}$ in terms of $B_t$. 
Two Steps: Passage of Time + Observation

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

Diagram:

- \( X_1 \rightarrow X_2 \rightarrow X_3 \rightarrow X_4 \)
- \( E_1 \rightarrow E_2 \rightarrow E_3 \rightarrow E_4 \)
- \( B(X_{t+1}) \)
Inference: Base Cases

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

\[ P(X_2) = \sum_{X_1} P(X_1, X_2) = \sum_{X_1} P(X_2 | X_1)P(X_1) \]
Passage of Time

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

\[
P(X_1) \quad P(X_2|X_1)
\]

$B(X_t) = P(X_t|e_{1:t})$

Then, after one time step passes:

\[
P(X_{t+1}|e_{1:t}) = \sum_{x_t} P(X_{t+1}, x_t|e_{1:t})
\]
\[
= \sum_{x_t} P(X_{t+1}|e_{1:t}, x_t)P(x_t|e_{1:t})
\]
\[
= \sum_{x_t} P(X_{t+1}|x_t)P(x_t|e_{1:t})
\]

Or, compactly: $B'_{t+1}(X_{t+1}) = \sum_{x_t} P(X_{t+1}|x_t)B_t(x_t)$

Basic idea: beliefs get “pushed” through the transitions

- With the “B” notation: $B'_t(\cdot)$, without $e_t$, $B_t(\cdot)$, with $e_t$. 
Example: Passage of Time

As time passes, uncertainty “accumulates”. 

[Images of matrices and robots with question marks]
Assume we have current belief $P(X|\text{previous evidence})$:

$$B'(X_{t+1}) = P(X_{t+1}|e_{1:t})$$

Then, after evidence comes in:

$$P(X_{t+1}|e_{1:t+1}) = P(X_{t+1}, e_{t+1}|e_{1:t})/P(e_{t+1}|e_{1:t})$$

$$\propto P(e_{t+1}|e_{1:t}, X_{t+1})P(X_{t+1}|e_{1:t})$$

$$= P(e_{t+1}|X_{t+1})P(X_{t+1}|e_{1:t})$$

Or, compactly: $B(X_{t+1}) \propto P(e_{t+1}|X_{t+1})B(X_t)$

Basic idea: beliefs “reweighted” by likelihood of evidence

Unlike passage of time, we have to renormalize
As we get observations, beliefs get reweighted, uncertainty “decreases”

\[ B(X) \propto P(e|X)B'(X). \]
Example: Weather HMM:

- **Rain_0**
  - $B(r) = 0.5$
  - $B(-r) = 0.5$

- **Rain_1**
  - $B'(r) = 0.5$
  - $B'(-r) = 0.5$

- **Rain_2**
  - $B'(r) = 0.627$
  - $B'(-r) = 0.373$

- **Umbrella_1**
  - $B(r) = 0.818$
  - $B(-r) = 0.182$

- **Umbrella_2**
  - $B(r) = 0.883$
  - $B(-r) = 0.117$

| $R_t$ | $R_{t+1}$ | $P(R_{t+1}|R_t)$ |
|-------|-----------|------------------|
| +r    | +r        | 0.7              |
| +r    | -r        | 0.3              |
| -r    | +r        | 0.3              |
| -r    | -r        | 0.7              |

| $R_t$ | $U_t$ | $P(U_t|R_t)$ |
|-------|------|-------------|
| +r    | +u   | 0.9         |
| +r    | -u   | 0.1         |
| -r    | +u   | 0.2         |
| -r    | -u   | 0.8         |
Every time step, we start with current $P(X|\text{evidence})$

We update for time:

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

We update for evidence:

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

The forward algorithm does both at once (and doesn’t normalize)
The Forward Algorithm

We are given evidence at each time and want to know:

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

We can derive the following updates

\[
P(X | e_{1:t}) \propto P(X_t | e_{1:t}) \text{ Normalize anytime.} \\
= \sum_{x_{t-1}} P(x_{t-1}, x_t | e_{1:t}) \\
= \sum_{x_{t-1}} P(x_{t-1} | e_{1:t-1}) P(x_t | x_{t-1}) P(e_t | x_t) \\
= P(e_t | x_t) \sum_{x_{t-1}} P(x_t | x_{t-1}) P(x_{t-1} | e_{1:t-1})
\]
Pacman – Sonar (P4)

[Demo: Pacman – Sonar – No Beliefs(L14D1)]
Video of Demo Pacman – Sonar (with beliefs)
Particle Filtering and Applications of HMMs
Particle Filtering
Particle Filtering

Filtering: approximate solution

Sometimes $|X|$ is too big to use exact inference

- $|X|$ may be too big to even store $B(X)$
- E.g. $X$ is continuous

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

Particle is just new name for sample
Representation: Particles

Our representation of $P(X)$ is now a list of $N$ particles (samples)

- Generally, $N << |X|$
- Storing map from $X$ to counts would defeat the point

$P(x)$ approximated by number of particles with value $x$

- So, many $x$ may have $P(x) = 0!$
- More particles, more accuracy

For now, all particles have a weight of 1
Particle Filtering: Elapse Time
Each particle is moved by sampling its next position from the transition model:

- This is like prior sampling – samples’ frequencies reflect the transition probabilities.
- Here, most samples move clockwise, but some move in another direction or stay in place.

This captures the passage of time:

- If enough samples, close to exact values before and after (consistent).
Particle Filtering: Observe
Particle Filtering: Observe

Slightly trickier.

- Don’t sample observation, fix it
- Similar to likelihood weighting, downweight samples based on the evidence

\[ w(s) = P(e|x) \]

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

- As before, probabilities don’t sum to one, since downweighted
  
  (sum to \((N\text{ times})\) an approximation of \(P(e)\))
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
Recap: Particle Filtering

Particles: track samples of states rather than an explicit distribution

Elapse  Weight  Resample

[Demos: ghostbusters particle filtering (L15D3,4,5)]
Video of Demo – Moderate Number of Particles
Video of Demo – One Particle
Video of Demo – Huge Number of Particles
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
Particle Filter Localization (Sonar)

[Video: global-sonar-uw-annotated.avi]

[Dieter Fox, et al.]
SLAM: Simultaneous Localization And Mapping
- We do not know the map or our location
- State consists of position AND map!
- Main techniques: Kalman filtering (Gaussian HMMs) and particle methods

DP-SLAM, Ron Parr
[Demo: PARTICLES-SLAM-mapping1-new.avi]
Particle Filter SLAM – Video 1

[Demo: PARTICLES-SLAM-mapping1-new.avi]

[Sebastian Thrun, et al.]
[Demo: PARTICLES-SLAM-fastslam.avi]
[Dirk Haehnel, et al.]