Markov Chains
Matrices to the rescue!

- Define a transition matrix $T$ as normal.
- Define a sequence of observation matrices $O_1$ through $O_t$.
- Each $O$ matrix is a diagonal matrix with the entries corresponding to that particular observation given each state.

$$f_{1:t+1} = \alpha f_{1:t} \cdot T \cdot O_{t+1}$$

where each $f$ is a row vector containing the probability distribution at state $t$. 
Forward algorithm

• Note that the forward algorithm only gives you the probability of $X_t$ taking into account evidence at times 1 through $t$.

• In other words, say you calculate $P(X_1 \mid e_1)$ using the forward algorithm, then you calculate $P(X_2 \mid e_1, e_2)$.
  – Knowing $e_2$ changes your calculation of $X_1$.
  – That is, $P(X_1 \mid e_1) \neq P(X_1 \mid e_1, e_2)$
Backward algorithm

• Updates previous probabilities to take into account new evidence.
• Calculates $P(X_k \mid e_{1:t})$ for $k < t$
  – aka smoothing.
Backward matrices

• Main equations:

\[ b_{k:t} = T \cdot O_k \cdot b_{k+1:t} \]

\[ b_{t+1:t} = [1; \cdots ; 1] \quad \text{(column vec of 1s)} \]

\[ P(X_k \mid e_{1:t}) = \alpha f_{1:k} \times b_{k+1:t} \]
Forward-backward algorithm

\[ f_{1:0} = P(X_0) \]

\[ f_{1:t+1} = \alpha f_{1:t} \cdot T \cdot O_{t+1} \]

Compute these forward from \( X_0 \) to wherever you want to stop.

\[ b_{t+1:t} = [1; \cdots ; 1] \]

\[ b_{k:t} = T \cdot O_k \cdot b_{k+1:t} \]

\[ P(X_k \mid e_{1:t}) = \alpha f_{1:k} \times b_{k+1:t} \]

Compute these backwards from \( X_t \) to \( X_0 \).
Viterbi algorithm

- Computes most likely sequence of states (not a single state).
- Just like forward algorithm, but compute max instead of sum in algorithm.