COSC 348:
Computing for Bioinformatics

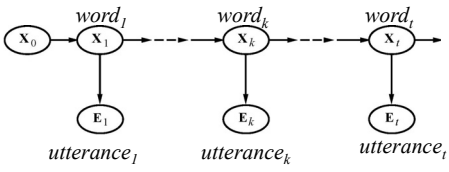
Lecture 9:
Hidden Markov models: theory

Lubica Benuskova

http://www.cs.otago.ac.nz/cosc348/

Hidden Markov Model (HMM)

- In a *Hidden Markov Model*, the state is not directly visible. Each state has a probability distribution over the possible output tokens (i.e. observations associated with that state).
- Hence, the sequence of tokens/observations generated by an HMM gives some information about the sequence of hidden states.
- E.g. speech – words are states and utterances are the observed tokens:

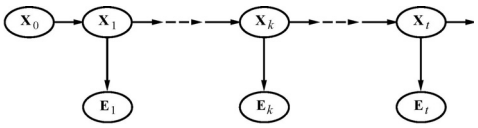


States and observations

- Let $\mathbf{X}_t$ denotes the set of state variables at time step t, and $\mathbf{E}_t$ denotes the set of (observations) evidence variables at time step t.
  - The concrete output/evidence at time t is $\mathbf{E}_t = \mathbf{e}_t$.
  - The concrete state at time t is $\mathbf{X}_t = \mathbf{x}_t$.
- Assumption: the same set of variables are state variables $\mathbf{X}_t$ and evidence variables $\mathbf{E}_t$, respectively, at each step t.
- We will assume that the state sequence starts at $t = 0$.
- We will assume that the evidence sequence starts at $t = 1$.

Transmission & observation (emission) model

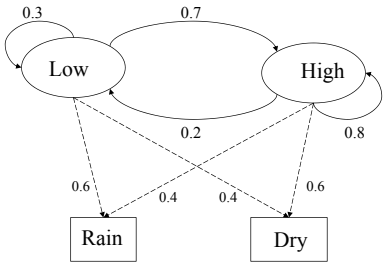
- We assume each state depends **only** on a previous state (1st order Markov process), hence the transition probability $\mathbf{P(X_t \mid X_{t-1})}$ for $\forall$t.
- We assume the evidence variables at time t depend **only** on the current state, hence the emission probability $\mathbf{P(E_t \mid X_t)}$ for $\forall$t.



- *Stationary process*: causal laws that govern the process of change of the world do not change over time.

Example of HMM for weather: Bayesian net

- We have one Boolean state variable that can have two values: *Atmospheric_Pressure* = {*Low, High*}.
- We cannot observe it directly, but we can observe whether it's raining or not – so evidence is also a Boolean variable: *Weather* = *{Rain, Dry}*

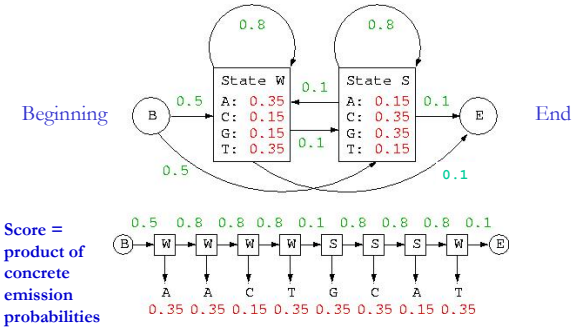


Example of HMM for weather: probabilities

- Two hidden states : 'Low' and 'High' atmospheric pressure.
- Two observations : 'Rain' and 'Dry' (which is negation of Rain).
- Transition probabilities: P('Low'|'Low') = 0.3; P('High'|'Low') = 0.7; P('Low'|'High') = 0.2 and P('High'|'High') = 0.8
- Observation probabilities : P('Rain'|'Low') = 0.6 , P('Dry'|'Low') = 0.4 , P('Rain'|'High') = 0.4 , P('Dry'|'High') = 0.3 .
- In order to perform any kind of probabilistic inference, we need to specify initial (i.e. prior) probabilities of states at time $t = 0$:
  - E.g.: P('Low') = 0.4 , P('High') = 0.6.

Example: a 2-state HMM for DNA

- Top: HMM architecture and parameters. (Note: States are abstract!)
  - green: state transition probabilities, red: emission probabilities.
- Bottom: sequence generation. Sequence score = $\Pi\, P(\, letter_t \mid state_t)$



Score = product of concrete emission probabilities

C-style pseudocode for generating output tokens

```
// Generate a random number between 0 and 1 according to
uniform distribution
double unifRand() {
                    return rand() / double(RAND_MAX);
}
int main(){
       seed();
       while (state[t] = W){
              rnd = unifRand();
              if (0.0 < rnd <= 0.35) letter[t] = A;
              if (0.35 < rnd <= 0.5) letter[t] = C;
              if (0.5 < rnd <= 0.65) letter[t] = G;
              if (0.65 < rnd <= 1.0) letter[t] = T;
       }
return(0);
}
```


C-style pseudocode for generating state transitions

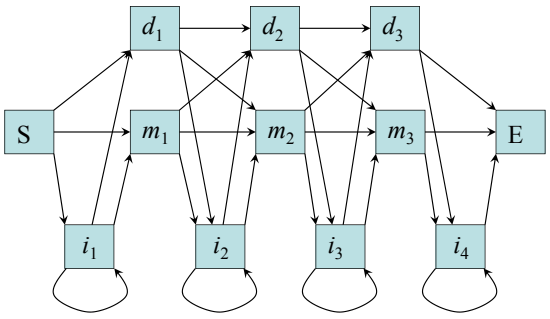
```
// Generate random number (0, 1] according to uniform distrib
double unifRand() {
                    return rand() / double(RAND_MAX);
}
int main(){
       seed();
       state[0] = B;
       rnd = unifRand();
       if (0.0 < rnd <= 0.5) state[1] = W;
                        else state[1] = S;
       until (state[t] = E){
              rnd = unifRand();
              if (0.0 < rnd <= 0.8) state[t+1] = state[t];
              if (0.8 < rnd <= 0.9) state[t+1] = the other state;
              if (0.9 < rnd <= 1.0) state[t+1] = E;
              t = t + 1;
       }
return(0);
}
```


Pseudocode for generating state transitions & observations

```
double unifRand() {
                    return rand() / double(RAND_MAX);
}
int main(){
       seed();  // new random seed
       state[0] = B;
       rnd = unifRand();
       if (0.0 < rnd <= 0.5) state[1] = W;
                        else state[1] = S;
       until (state[t] = E){
              rnd = unifRand();
              …  // transition to a new state
              if(state[t] = W){
                    rnd = unifRand(); … // new output token
                    }
               if(state[t] = S){
                    rnd = unifRand(); … // new output token
                    }
              t = t + 1;
       }
return(0);
}
```

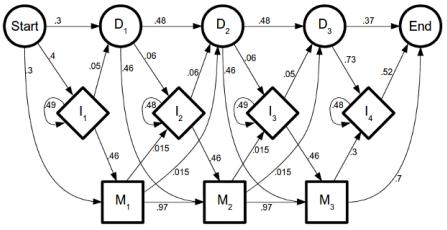

General architecture of HMM for biosequences

- S is the start state, E is the end state, d_i denotes deletion at position i, m_i denotes match sequence letter at position i; i_i denotes insertion of a letter at i. Each transition (arrow) is accompanied with a probability.



Example of trained HMM for DNA (lab 5)

- **Left:** state transition model; **Right:** emission model for M and I states.
- We have 4 insertion states, 3 match states and 3 deletion states (# of D and I states depends on # of M states).



	A	C	G	T
M1	0.10	0.80	0.04	0.06
M2	0.40	0.55	0.04	0.01
M3	0.73	0.18	0.02	0.07
I1	0.15	0.29	0.33	0.23
I2	0.44	0.51	0.02	0.03
I3	0.70	0.08	0.19	0.03
I4	0.01	0.01	0.08	0.90