Prob Set 8: due 11 May (end of classes)
Consider the long term behavior of a Markov chain: is there some set of probabilities $v_i$ for being in state $i$ after some large number of steps, independent of the starting state?

The answer is yes for an ergodic Markov chain, and we call $v_i$ the stationary distribution of the chain.

The $v_i$ can also be thought of as the long term frequency of being in state $i$.

If the $v_i$ are really stationary, in the sense that further steps leave them unchanged, then they satisfy the key relation

$$\sum_j v_j T_{ji} = v_i .$$
Consider for example the chain below, with two states:

The stationary condition above becomes the two equations 
\[ \frac{5}{2}v_1 + \frac{8}{2}v_2 = v_2, \quad \frac{2}{2}v_2 + \frac{5}{2}v_1 = v_1, \]
with solution \( 2v_2 = 5v_1 \), and hence normalized as probabilities we have \( v_1 = \frac{2}{7}, \ v_2 = \frac{5}{7}. \)
More generally, for transition probabilities $p_1$ and $p_2$:

The two equations $p_1 v_1 + (1 - p_2) v_2 = v_2$, $p_2 v_2 + (1 - p_1) v_1 = v_1$ have solution $p_1 v_1 = p_2 v_2$, and hence normalized as probabilities we have $v_1 = p_2 / (p_1 + p_2)$ and $v_2 = p_1 / (p_1 + p_2)$. 
The two equations \( p_1 v_1 + (1 - p_2) v_2 = v_2 \), \( p_2 v_2 + (1 - p_1) v_1 = v_1 \) have solution \( p_1 v_1 = p_2 v_2 \), and hence normalized as probabilities we have \( v_1 = p_2/(p_1 + p_2) \) and \( v_2 = p_1/(p_1 + p_2) \).

Intuitively it makes sense that in the steady state one spends more time at \( v_2 \) if the transition probability from \( v_1 \) to \( v_2 \) is greater than vice versa.

The steady state probabilities depend only on the ratio \( p_1/p_2 \), but recall that the waiting times depend on their values: the expected number of steps to leave \( v_i \) is \( 1/p_i \), so that when the \( p_i \) are small it takes longer to equilibrate to the steady state distribution.
HMM = “Hidden Markov Model”

Suppose we can no longer observe the state directly: it is “hidden”.

We can only see “observables” emitted by the state, in turn emitted according to some probability distribution.

(c.f. Kleinberg burst detection)
Hidden Markov Models and the Viterbi algorithm

Imagine a set of urns labeled $1, \ldots, N$ behind a curtain.

Each urn has balls of various colors (a total of $M$ possible colors).

A “genie”:

1) chooses an urn according to some initial probability distribution

2) chooses a ball from that urn, calls out its color, and replaces it in the urn

3) picks a new urn according to some probability that depends only on the current urn, and continues steps 2,3, calling out a series of colors.

Given a sequence of observations $O = O_1 O_2 \cdots$, and an HMM $H = (T_{ij}, e_i(a), w_i)$, we wish to find the maximum probability state path $Q = q_1 q_2 \cdots q_T$. This can be done recursively using the Viterbi algorithm.

\[
Q = 1 2 3 3 6 6 5 \ldots
\]

\[
O = RRGBMRB \ldots
\]
“Slightly Dishonest Casino”

Fair and Loaded dice as before, but now a hidden transition:

\[ T_{ij} = \begin{pmatrix} .95 & .05 \\ .1 & .9 \end{pmatrix} \]

Will generate a series of 6235116166522345…

Can we tell just from the sequence of observations during which periods it was likely loaded?
```python
def roll(die='F'):
    if die=='F': return randint(1,7)
    elif die== 'L': return random.choice([1,2,3,4,5,6,6,6,6,6])
    else: return None

def trial(state):
    r=rand()
    if state=='F' and rand() < .05: state = 'L'
    elif state=='L' and rand() < .1: state = 'F'
    return state,str(roll(state))
```

http://nbviewer.jupyter.org/url/courses.cit.cornell.edu/info2950_2017sp/resources/viterbi.ipynb
CpG Islands

“vulnerability of methylcytosines to spontaneously deaminate to thymine”

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>C</th>
<th>G</th>
<th>T</th>
</tr>
</thead>
<tbody>
<tr>
<td>-</td>
<td>0.300</td>
<td>0.205</td>
<td>0.285</td>
<td>0.210</td>
</tr>
<tr>
<td>A</td>
<td>0.322</td>
<td>0.298</td>
<td>0.078</td>
<td>0.302</td>
</tr>
<tr>
<td>C</td>
<td>0.248</td>
<td>0.246</td>
<td>0.298</td>
<td>0.208</td>
</tr>
<tr>
<td>G</td>
<td>0.177</td>
<td>0.239</td>
<td>0.292</td>
<td>0.292</td>
</tr>
<tr>
<td>T</td>
<td>0.179</td>
<td>0.355</td>
<td>0.384</td>
<td>0.182</td>
</tr>
</tbody>
</table>

+- is now the hidden state for $O = \text{CAGCTTCGCGCGATC…}$
We want to calculate the probability that a sequence is generated by some transition matrix $T_{ij}$.

Consider the sequence $s=ATGGC$, then the probability is the product $p(A \rightarrow T) \ p(T \rightarrow G) \ p(G \rightarrow G) \ p(G \rightarrow C)$.

Calculated for the transition matrix $T^+$ that product will give one result, say $p(+|s)$, and calculated for $T^-$, another result, say $p(-|s)$.

One way to determine which is the larger probability is to divide $p(+|s) / p(-|s)$ to see if greater than one or less than one.

That would be sufficient …
Or, equivalently, take the log of the quotient $\log(p(+) / p(-))$ and see if greater or less than zero. But notice that taking the log of the quotient of the two products of transition probabilities is equivalent to summing the logs of the quotients of the individual transition probabilities:

$$\log(p(+) / p(-)) = \log(p_+(A \rightarrow T) / p_-(A \rightarrow T)) + \ldots + \log(p_+(G \rightarrow C) / p_-(G \rightarrow C))$$

where $p_+(\ldots)$ and $p_-(\ldots)$ are transition probabilities corresponding to $T^+$ and $T^-$.

But then observe that the individual entries for $T^+$ and $T^-$ are never needed: only the logs of their ratios, entry by entry, are needed and these can be calculated in advance once and for all: that's the quantity $r_{ab} = \log \left( \frac{T^+_{ab}}{T^-_{ab}} \right)$.

Sum those weights according to the transitions in a given sequence, and whether that sum is greater or less than zero is equivalent to whether the ratio of the original sequence transition probabilities $p(+) / p(-)$ is greater or less than 1.
(Same thing more formally:)

Suppose we’re given some snippets of genomic data in the form of strings of nucleobases \( s = s_0 s_1 \ldots s_{n-1} \) (each \( s_i \) is equal to one of A,C,G,T), and we want to determine whether they come from + or – regions characterized by the transition matrices \( T^{(-)} \) and \( T^{(+)} \).

The probability that string \( s \) is generated by either of the transition matrices is given by

\[
p(s|+) = \prod_{i=0}^{n-2} T^{(+)}_{s_i, s_{i+1}} \quad \quad p(s|-) = \prod_{i=0}^{n-2} T^{(-)}_{s_i, s_{i+1}}
\]

To use these for discrimination, we define the “score” as the log-odds ratio:

\[
\text{Score}(s) = \log_2 \left( \frac{p(s|+)}{p(s|-)} \right)
\]

A positive or negative score means that the string \( s \) is more likely to have been generated by \( T^{(+)} \) or \( T^{(-)} \), respectively.
To use these for discrimination, we define the “score” as the log-odds ratio:

\[
\text{Score}(s) = \log_2 \left( \frac{p(s|+)}{p(s|-)} \right)
\]

A positive or negative score means that the string \( s \) is more likely to have been generated by \( T^{(+)} \) or \( T^{(-)} \), respectively.

In terms of the log likelihood ratios \( r_{ab} \equiv \log_2 \left( \frac{T_{ab}^{(+)}}{T_{ab}^{(-)}} \right) \), the score takes the simple form

\[
\text{Score}(s) = \log_2 \left( \prod_{i=0}^{n-2} \frac{T^{(+)}_{s_{i}, s_{i+1}}}{T^{(-)}_{s_{i}, s_{i+1}}} \right) = \sum_{i=0}^{n-2} \log_2 \left( \frac{T^{(+)}_{s_{i}, s_{i+1}}}{T^{(-)}_{s_{i}, s_{i+1}}} \right) = \sum_{i=0}^{n-2} r_{s_{i}, s_{i+1}}.
\]

where \( r_{ab} \) can be considered a weight for each transition.

We sum over the transitions to see whether the positive or negative dominate.
Matrix multiplication calculates overall multi-step probabilities:

Consider, e.g., the A,G element of $T^2$:

it’s the sum over $i$ of $T_{Ai} T_{iG}$, hence computes the one step probabilities.

Similarly, the A,G element of $T^3$ is the sum over $i$ and $j$ of $T_{Ai} T_{ij} T_{jG}$, hence computes the two step probabilities.

The use $\text{T.dot(T)}$ for the square of a matrix array. For higher powers, use, e.g., $\text{np.linalg.matrix_power(T, 10)}$ for $T^{10}$.

Note: this is not $\text{T**10}$, which is the tenth power of the individual entries, rather than the matrix multiplication ten times.
Cumulative sums of $r_{ab} = \log \left( \frac{T^+_{ab}}{T^-_{ab}} \right)$
Imagine a set of urns labeled 1,\ldots, N behind a curtain.

Each urn has balls of various colors (a total of \( M \) possible colors).

A “genie”:

1) chooses an urn according to some initial probability distribution

2) chooses a ball from that urn, calls out its color, and replaces it in the urn

3) picks a new urn according to some probability that depends only on the current urn, and continues steps 2,3, calling out a series of colors.

\[
O = \text{RRGBMRB} \ldots
\]
\[
Q = 1\, 2\, 3\, 3\, 6\, 6\, 5 \ldots
\]
The metaphor we used for a hidden Markov model was
Imagine a set of urns labeled 1,...,N behind a curtain.
Each urn has balls of various colors (a total of M possible colors).
A "genie":
1) chooses an urn according to some initial probability distribution
2) chooses a ball from that urn, calls out its color, and replaces it in the urn
3) picks a new urn according to some probability that depends only on the current urn, and continues steps 2,3, calling out a series of colors.
The colors are the “observations” O and the series of urns correspond to the “hidden states” q which we can’t access.

The initial probability weights in step 1 are given by \( w_i \), the “emission” probabilities in step 2 are given by \( e_i(a) = p(O = a | q = i) \), and the urn transitions in step 3 are given by an underlying set of Markov transitions characterized by a transition matrix \( T_{ij} \).

An HMM \( H = (T_{ij}, e_i(a), w_i) \) is understood to have N hidden Markov states labelled by \( i \) (1 \( \leq \) i \( \leq \) N), and M possible observables for each state, labelled by \( a \) (1 \( \leq \) a \( \leq \) M).
An HMM $H = (T_{ij}, e_i(a), w_i)$ is understood to have $N$ hidden Markov states labelled by $i$ ($1 \leq i \leq N$), and $M$ possible observables for each state, labelled by $a$ ($1 \leq a \leq M$).

The state transition probabilities are $T_{ij} = p(q_{t+1} = j \mid q_t = i)$, $1 \leq i, j \leq N$ (where $q_t$ is the hidden state at time $t$), the emission probability for the observable $a$ from state $i$ is $e_i(a) = p(O_t = a \mid q_t = i)$ (where $O_t$ is the observation at time $t$), and the initial state probabilities are $w_i = p(q_1 = i)$.

Given a sequence of observations $O = O_1 O_2 \cdots O_T$, and an HMM $H = (T_{ij}, e_i(a), w_i)$, we wish to find the maximum probability state path $Q = q_1 q_2 \cdots q_T$. This can be done recursively using the Viterbi algorithm.
Given a sequence of observations $O = O_1 O_2 \cdots O_T$, and an HMM $H = (T_{ij}, e_i(a), w_i)$, we wish to find the maximum probability state path $Q = q_1 q_2 \cdots q_T$. This can be done recursively using the Viterbi algorithm.

Let $v_i(t)$ be the probability of the most probable path ending in state $i$ at time $t$, i.e.,

$$v_i(t) = \max_{q_1, q_2, \cdots, q_{t-1}} P(q_1 q_2 \cdots q_{t-1}, q_t = i, O_1 O_2 \cdots O_t | H),$$

and let $w_i$ be the initial probabilities of the states $i$ at time $t = 1$. 
Viterbi Algorithm

Then \( v_j(t) \) can be calculated recursively using

\[
v_j(t) = \max_{1 \leq i \leq N} \left[ v_i(t - 1) T_{ij} \right] e_j(O_t)
\]

together with initialization

\[
v_i(1) = w_i e_i(O_1) \quad 1 \leq i \leq N
\]

and termination

\[
P^* = \max_{1 \leq i \leq N} [v_i(T)]
\]

(i.e., at the end we choose the highest probability endpoint, and then we backtrack from there to find the highest probability path).
This algorithm has properties similar to the Dijkstra algorithm.

There was only necessary to consider the shortest length path from the start point to any intermediate point, since any longer path to the intermediate point would necessarily result in a longer total path from start to endpoint.

Here only necessary to consider the maximum probability path to any intermediate observation, since a lower probability path would necessarily result in a lower total probability path in the state space from initial to final observation.

For a series of $T$ observations, the total number of possible paths for an $N$ state model is $N^T$ ($N$ possible states at each time), exponential in $T$, quickly grows prohibitively large.

($T = 100$ observations in an $N = 5$ model would have $5^{100} \approx 10^{70}$ possible paths.)

The Viterbi algorithm instead finds the most probable path in computational time linear in the number of observations $T$ (i.e., $O(TN^2)$)
Simple example: N=3 urns, M=2 observables, T=3 observations

Suppose we observe the sequence \( Q = \text{RBR} \)

To illustrate this, consider a three state HMM, with initial state probabilities

\[
T_{ij} = \begin{pmatrix}
0.3 & 0.6 & 0.1 \\
0.5 & 0.2 & 0.3 \\
0.4 & 0.1 & 0.5
\end{pmatrix}
\]

and given

find the most likely state sequence \( Q \)

\( e_1(R) = 1/2 \)
\( e_1(B) = 1/2 \)

\( j = 1 \)

\( e_2(R) = 1/3 \)
\( e_2(B) = 2/3 \)

\( j = 2 \)

\( e_3(R) = 3/4 \)
\( e_3(B) = 1/4 \)

\( j = 3 \)

\( O = \text{RBR} \)
To illustrate this, consider a three state HMM, with initial state probabilities $w_i = 1/3$ and transition matrix $T_{ij} = \begin{pmatrix} .3 & .6 & .1 \\
.5 & .2 & .3 \\
.4 & .1 & .5 \end{pmatrix}$.

The emitted symbols are:

- $e_1(R) = 1/2$
- $e_1(B) = 1/2$
- $e_2(R) = 1/3$
- $e_2(B) = 2/3$
- $e_3(R) = 3/4$
- $e_3(B) = 1/4$

The emitted symbol sequence is $O = RBR$ and the state sequence is $Q = 1\, 2\, 1$.
observed:
45562134556313624253541665626666616264166231514243642341662266566532312666662166554633665546622165
(actual) hidden:
FFFFFFFFFFFFFFFFFFFFFLLLLLL
inferred:
FFFFFFFFFFFFFFFFFFFFFLLLLLL

Gets the essential large-scale features of the state transitions.

The values of the transition matrix $T_{ij}$ used in the HMM can be inferred from the data: for example in a two hidden state model, starting from annotated data we can determine the average length of time in each state (corresponds to the wait time, or number of steps on average before transitioning to the other state — recall that average number of steps is equal to $1/p$, where $p$ is the transition probability out of the state).

These times then directly determine $T_{12}$ and $T_{21}$, and thereby $T_{11} = 1 - T_{12}$, $T_{22} = 1 - T_{21}$.
Note that the maximally likely path is not the only possible optimality criterion, for example choosing the most likely state at any given time provides a different algorithm and can give a slightly different result.

But the overall most likely path provided by the Viterbi algorithm provides an optimal state sequence for many purposes.

One metaphor for describing it is the “lazy instructor” who gives an exam but does not want to solve the problems him or herself.

After exams collected, the instructor has a choice of methodologies for solution sheet:

- use the consensus of all students on each problem to determine the correct solutions
- find a single student, known to be best in class, and use that student’s answers

The Viterbi algorithm corresponds to the latter methodology.