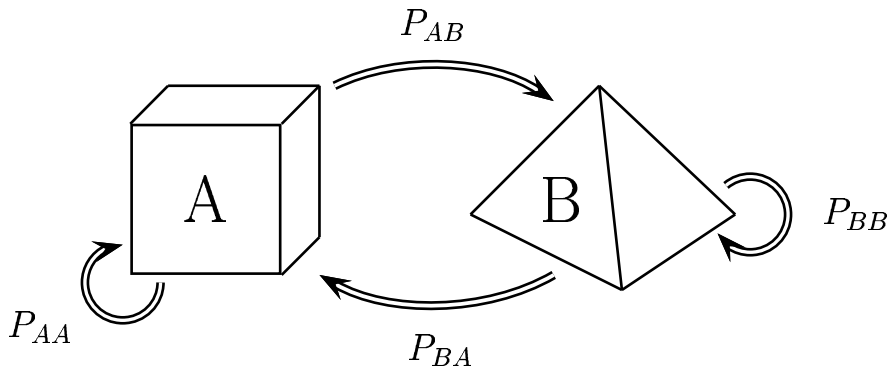
Simulation and Analysis of a Simple HMM

1. Simulate an HMM

First, build an HMM simulator that outputs a sequence of observations. We will use this in part 2, to predict optimal path through the statespace.

Assume we have two dice, A and B. Die A has six sides and generates numbers between 1 and 6, and die B has four sides and generate numbers between 1 and 4.



We start to role one of the dice according to the initial distribution

$$\pi = (\pi_A, \pi_B) = (0.7, 0.3)$$

and then we choose the subsequent die according to the transition probabilities

$$P = \begin{pmatrix} P_{AA} & P_{AB} \\ P_{BA} & P_{BB} \end{pmatrix} = \begin{pmatrix} 0.6 & 0.4 \\ 0.6 & 0.4 \end{pmatrix}$$

In each role the two dice output numbers according to the output distribution

$b_i(k)$	1	2	3	4	5	6
A	1/6	1/6	1/6	1/6	1/6	1/6
B	1/4	1/4	1/4	1/4	0	0

Simulate the HMM.

This involves creating a random generator and a function that chooses a random number from a given distribution.

In C you can use the function `random()`, where `random(100)` will produce a random number between 0 and 99. However, if you run the same program over and over, the same sequence of random numbers will be generated, since C always seeds the random number generator with the same starting number. To avoid this you can call `randomize()` first thing in your program (only needed once and not before every call to `random`), to create a seed based on the system clock (which always changes).

To choose a random number from a given distribution $\{q_1, q_2, \ldots, q_n\}$ say, where $q_1 + \ldots + q_n = 1$:

1. Generate a random number r between 0 and 1.
2. If $\sum_{i=1}^{k-1} q_i < r \leq \sum_{i=1}^{k} q_i$ then your choice is q_i.

Now create a program that produces an ouput sequence of length 10. Also, record the state space sequence as well, as we will use it in part 2. The pseudo-code for the program could look something like:

```
randomize(); /* to seed the random generator */
```
Select (and record) initial state i according to distribution π_i.
Select a die number k according to distribution $b_i(k)$.
```
for t=2 to 10
```
  Select (and record) next state j according to distribution p_{ij}.
  Select a die number m according to distribution $b_j(m)$.
```
end for
```
Print die sequence.
Print sequence of hidden states.

Report:

- The code for the simulation.
- The observed sequence of numbers from the dice.
- The hidden state sequence (of A's and B's) giving rise to those numbers.

2. Analyze the HMM

Now we assume we have observed the sequence of numbers you simulated in part 1, and we want to find the best underlying sequence of states that gave rise to this. It will be interesting to see if this differs from the sequence of A's and B's you already recorded.

You need to program the Viterbi algorithm. Let $Y_1, \ldots, Y_T$ be the sequence of observed numbers (all between 1 and 6), $X_1, \ldots, X_T$ the sequence of underlying states (A's and B's), P_{ij} the transition probability from state i to state j, and $b_i(k)$ the output probability of number k in state i. Then the Viterbi variables $\delta_i(t)$ are defined as the score (probability) of the best sequence of hidden states up to time t, giving rise to the observations $Y_1, \ldots, Y_t$ and ending in state i at t (*i.e.* $X_t = i$), or

$$\delta_i(t) \quad = \quad b_i(Y_t) \ \max_j \{\delta_j(t-1) P_{ji}\}$$

with $\delta_i(0) = \pi_i$.

Calculate $\delta_i(t)$ for both states $i \in \{A, B\}$ and all $t = 1, \ldots, 10$. In each step also record the best previous position, *i.e.* in position (i, t) if $\delta_A(t-1)P_{Ai} > \delta_B(t-1)P_{Bi}$ record $(A, t-1)$, otherwise record $(B, t-1)$.

To obtain the optimal path through the statespace (the path with the highest probability given your observations) backtrack through your recorded pointers starting

at $\max\{\delta_A(T), \delta_B(T)\}$.

It is also possible to calculate the joint probability of the observations and a given state sequence

$$\Pr(X_1, \ldots, X_T, Y_1, \ldots, Y_T) = \pi_{X_1} b_{X_1}(Y_1) \prod_{t=2}^{T} P_{X_{t-1}, X_t}\ b_{X_t}(Y_t).$$

Report:

- The code for the Viterbi algorithm.
- The optimal path through the state space obtained using the Viterbi.
- The probability of the observed sequence and the best underlying state sequence.
- The probability of the observed sequence and the state sequence recorded in part 1, if this is different from the best underlying state sequence.
- The probability of the observed sequence and a state sequence of all B's.