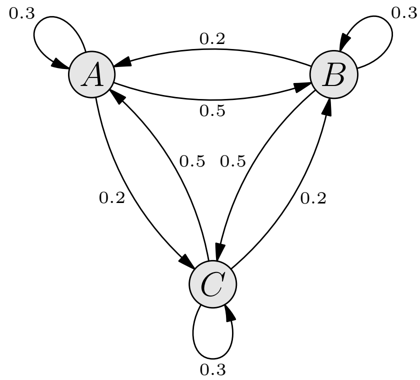
Inference with Markov models

Recall the state transition diagram and transition probability matrix of a Markov chain λ_1 which models the behavior of an NPC who is "patrolling the map"



$$\boldsymbol{P}_{\lambda_1} = \begin{bmatrix} 0.3 & 0.2 & 0.5 \\ 0.5 & 0.3 & 0.2 \\ 0.2 & 0.5 & 0.3 \end{bmatrix}$$

The probability of the event E_1 = "the NPC goes from room C to room B and then to room A" amounts to

$$p\big(E_1\big) = p\big(X_1 = B \mid X_0 = C\big) \cdot p\big(X_2 = A \mid X_1 = B\big)$$

and the corresponding log-likelihood amounts to

$$\mathcal{L}\big(E_1\big) = \ln p\big(X_1 = B \mid X_0 = C\big) + \ln p\big(X_2 = A \mid X_1 = B\big)$$

Compute both these values for the model λ_1. Round your results to *two* decimal places.

$$p\big(E_1\big) = 0.04$$

$$\mathcal{L}\big(E_1\big) = -3.22$$

Also for model λ_1, compute the probability and log-likelihood of the event E_2 = "the NPC goes from room C to room A and then to room B"

$$p\big(E_2\big) = 0.25$$

$$\mathcal{L}\big(E_2\big) = -1.39$$

Sampling a Markov model

Given a given Markov chain λ over some state space $\mathcal{S}$, the following procedure generates a sequence of states $\boldsymbol{s} = s[0]s[1]s[2]\cdots$

manually initialize $s[0] \in \mathcal{S}$

if $s[t-1] = s_i$, *draw* $s[t] = s_j$ according to $p\big(s_j \mid s_i\big)$, that is select

$$s_j \sim \big(\boldsymbol{P}_\lambda\big)_{ji}$$

where $\boldsymbol{P}_\lambda$ is the state transition probability matrix of the process λ.

Using matrix $\boldsymbol{P}_{\lambda_1}$, generate a sequence $\boldsymbol{s}$ of 200 states where $s[0] = A$. That is, replace the dots in the following expression by 199 symbols sampled according to the above algorithm.

$\boldsymbol{s}$ = A A B B A A B A B B C A A C A A A C C B B A C C A B C A B
A B A B C A B C A B A C A A A B C C B C A B C A A A A A B C C A B B B
C A B A B C C C C A C C A B C A B C A B B C C A A B B C C B C C A B
C B A B C A C C A B C C B C B B C A A B C B C B B A A B C A A B C A
C C A B B C C A B B C A B B A C C C B C A C A B C A B C A C B A B A
B B C B A B C C B C B C A A B B C A A A B C A B C C A C A A A B A B

Using matrix $\boldsymbol{P}_{\lambda_1}$, generate $n = 1000$ sequences $\boldsymbol{s}_l$ of length 200 where $s_l[0] = A$ for all l. Compute their average log-likelihood

$$\frac{1}{n}\sum_{l=1}^{n}\mathcal{L}\big(\boldsymbol{s}_l\big) = -205.06$$

Learning a Markov model

Given a state sequence $\boldsymbol{s}$ such as the one you just created, we may use maximum likelihood estimation techniques to estimate or “learn” a Markov model λ_{MLE} that is most likely to have produced $\boldsymbol{s}$.
Maximum likelihood estimates of the entries of the corresponding matrix $\boldsymbol{P}_{\lambda_{MLE}}$ are given by

$$\big(\boldsymbol{P}_{\lambda_{MLE}}\big)_{ji} = \frac{n_{ji}}{\sum_j n_{ji}}$$

where the quantity n_{ji} counts how often the transition $s_i \rightarrow s_j$ occurs in sequence $\boldsymbol{s}$.
Given the "training" sequence $\boldsymbol{s}$ you created above, compute the entries of $\boldsymbol{P}_{\lambda_{MLE}}$. Round your results to two decimals.

$$\boldsymbol{P}_{\lambda_{MLE}} = \begin{bmatrix} 0.3 & 0.2 & 0.5 \\ 0.5 & 0.3 & 0.2 \\ 0.2 & 0.5 & 0.3 \end{bmatrix}$$

Stationary distribution a Markov model

Compute the stationary distribution

$$\boldsymbol{\pi} = \boldsymbol{P}_{\lambda_1} \boldsymbol{\pi}$$

of the Markov chain λ_1. Proceed as follows: introduce

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{I} - \boldsymbol{P}_{\lambda_1} \\ \boldsymbol{1}^\intercal \end{bmatrix} \qquad \text{and} \qquad \boldsymbol{b} = \begin{bmatrix} \boldsymbol{0} \\ 1 \end{bmatrix}$$

where the matrix $\boldsymbol{I}$ and the vector $\boldsymbol{0}$ and $\boldsymbol{1}$ have to be of appropriate sizes. Then solve

$$\boldsymbol{A}\,\boldsymbol{\pi} = \boldsymbol{b}$$

for $\boldsymbol{\pi}$. Round your result to two decimals.

$$\boldsymbol{\pi} = \begin{bmatrix} 0.33333333 \\ 0.33333333 \\ 0.33333333 \end{bmatrix}$$

If you like, paste the code you used to practically compute your result after the following `import` statements

```
import numpy as np
import numpy.linalg as la
```