Example sheet 4

Random processes

Data Science—DJW—2022/2023

Question 1. Draw the state space diagram for this Markov chain.

```
def rw():
    MAX_STATE = 9
    x = 0
    while True:
        yield x
        d = numpy.random.choice([-1,0,1], p=[1/4,1/2,1/4])
        x = min(MAX_STATE, max(0, x + d))
```

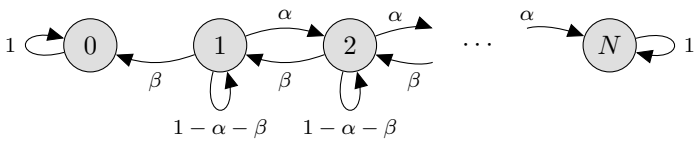
Question 2. For the Cambridge weather simulator, example 10.2.1 in lecture notes, show that

$$\mathbb{P}(X_3 = r \mid X_0 = g) = \sum_{x_1,x_2} P_{gx_1} P_{x_1x_2} P_{x_2r}.$$

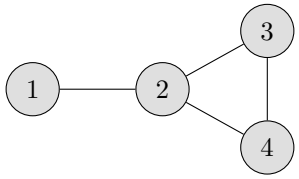
Question 3. Here is the state space diagram for a Markov chain with state space $\{0, 1, ..., N\}$. States 0 and N are absorbing states. At any other state x we jump to $x + 1$ with probability α, and to $x - 1$ with probability β. Assume $\alpha > 0$ and $\beta > 0$ and $\alpha + \beta \leq 1$. Let $h_x = \mathbb{P}(\text{hit } 0 \mid \text{start at } x)$.

- For $N = 3$, calculate h_x for all $x \in \{0, 1, 2, 3\}$.
- For arbitrary N, give code to compute the vector $[h_0, h_1, \ldots, h_N]$.

[Note. In a prior version, the text didn't match the state space diagram. Fixed 2022-12-01.]



Question 4. Consider a random walk on the vertices of this undirected graph, as follows: each timestep we take one of the edges chosen at random, each edge from our current vertex equally likely. Find the stationary distribution.



Question 5. Let X_n be a mean-reverting random walk,

$$X_{n+1} = \mu + \lambda(X_n - \mu) + N(0, \sigma^2) \quad \text{where} \quad -1 < \lambda < 1.$$

The stationary distribution for this process is a Normal distribution. Find its parameters.

Question 6. The Internet uses an algorithm called TCP to manage congestion. For every data flow, the sender maintains a congestion window $W_n \in \mathbb{R}$. It keeps roughly W_n packets in flight, thus the transmission rate is $W_n/\texttt{RTT}$ packets per second where `RTT` is the round trip time ('ping latency') between sender and receiver. The sender updates W_n every time it receives an acknowledgement of a packet, by

$$W_{n+1} = \begin{cases} W_n + 1/W_n & \text{if the packet didn't experience congestion} \\ W_n/2 & \text{if the packet did experience congestion.} \end{cases}$$

Suppose that packets experience congestion with probability p, independently. Write down a drift model for W_n and find the fixed point.

Question 7. We're given a sequence $(x_0, x_1, \ldots, x_n)$ starting at $x_0 = 0$, and we decide to model it as a mean-reverting random walk as in question 5. Explain how to estimate λ, μ, and σ.

Question 8. Consider a moving object with noisy location readings. Let X_n be the location at timestep $n \geq 0$, and Y_n the reading. Here's the simulator.

```
def hmm():
    MAX_STATE = 9
    x = numpy.random.randint(low=0, high=MAX_STATE+1)  # initial location X_0
    while True:
        e = numpy.random.choice([-1,0,1])
        y = min(MAX_STATE, max(0, x + e))  # noisy reading of location
        yield y
        d = numpy.random.choice([-1,0,1], p=[1/4,1/2,1/4])
        x = min(MAX_STATE, max(0, x + d))  # new location at next timestep
```

We'd like to infer the location X_n, given readings $y_0, \ldots, y_n$.

(a) Give justifications for the following three equations, which give an inductive solution. First the base case,

$$\Pr(x_0 \mid y_0) = \text{const} \times \Pr(x_0) \Pr(y_0 \mid x_0),$$

and next two equations for the induction step,

$$\Pr(x_n \mid h) = \sum_{x_{n-1}} \Pr(x_{n-1} \mid h) \Pr(x_n \mid x_{n-1})$$
$$\Pr(x_n \mid h, y_n) = \text{const} \times \Pr(x_n \mid h) \Pr(y_n \mid x_n).$$

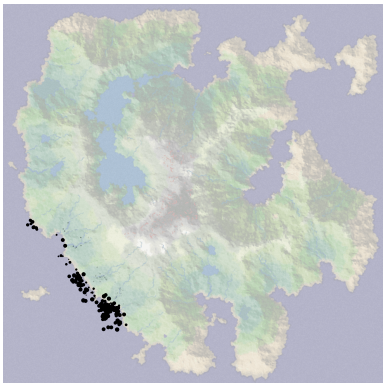
In these two equations, h stands for $(y_0, \ldots, y_{n-1})$, and we'll assume we've already found $\Pr(x_{n-1}|h)$.

(b) Give pseudocode for a function that takes as input a list of readings $(y_0, \ldots, y_n)$ and outputs the probability vector

$$\big[\pi_0, \ldots, \pi_{\texttt{MAX\_STATE}}\big], \qquad \pi_x = \mathbb{P}(X_n = x \mid y_0, \ldots, y_n).$$

(c) If your code is given the input $(3, 3, 4, 9)$, it should fail with a divide-by-zero error. Give an interpretation of this failure.

Question 9 (Stoat-finding challenge, not for supervision). Work through the notebook at `https://www.cl.cam.ac.uk/teaching/2223/DataSci/datasci/ex/ex4.html`, in which you will implement a 'particle filter' to solve question 8 on a larger map, using computational methods rather than exact calculations. Submit your answer on Moodle.



Hints and comments

Question 1. First identify the state space, i.e. the set of possible values for x. Looking at the code, we see that x can only ever be an integer in $\{0, 1, \ldots, 9\}$, so this is the state space. Next, draw arrows to indicate transitions between states. Make sure that at every node you draw, the probabilities on all outgoing edges sum up to one. You don't need to draw every state in your state space diagram: just show a typical state, and also the edge cases.

Question 2. There's a brute force solution, very similar to example 10.2.1 from lecture notes: first use the law of total probability to condition on X_1 AND X_2, giving us an expression that includes $\mathbb{P}(X_2 = x_2, X_1 = x_1 \mid X_0 = g)$, and then break this expression down further using the definition of conditional probability with baggage $\{X_0 = g\}$.

There's also a more elegant solution based on a more sophisticated use of memorylessness, which says in its most general form that "conditional on the present, the past and the future are independent". This includes the sort of equation stated at the top of section 10.2 of lecture notes,

$$\mathbb{P}(X_3 = x_3 \mid X_2 = x_2, X_1 = x_1, X_0 = x_0) = \mathbb{P}(X_3 = x_3 \mid X_2 = x_2),$$

but it also includes situations where there are gaps in the future e.g.

$$\mathbb{P}(X_3 = x_3 \mid X_1 = x_1, X_0 = x_0) = \mathbb{P}(X_3 = x_3 \mid X_1 = x_1) \quad \text{(present is } x_1\text{)}$$

and situations where there are gaps in the past, e.g.

$$\mathbb{P}(X_3 = x_3 \mid X_2 = x_2, X_0 = x_0) = \mathbb{P}(X_3 = x_3 \mid X_2 = x_2) \quad \text{(present is } x_2\text{)}.$$

Can you use the gaps-in-the-past version to answer this question? Can you prove the gaps-in-the-past and the gaps-in-the-future versions of memorylessness?

Question 3. This is like example 10.3.1 from notes. The edge cases are $h_0 = 1$ and $h_N = 0$. For the maths part: write down two equations, one for h_1 and one for h_2, and solve them simultaneously. (Or, if you're good at solving recurrence relations, write down the equations for arbitrary N and solve!) For the code part: write down the equations for arbitrary N, and convert to code as in example 10.3.1.

Supplementary question 12 looks at a Markov chain model for an epidemic, and asks you to find the probability of hitting state 0, i.e. the probability that the epidemic dies out.

Question 4. First write out the transition probability: $P_{xy} = 0$ if there's no $x \leftrightarrow y$ edge, and $P_{xy} = 1/n_x$ otherwise, where n_x is the number of edges incident at vertex x. In indicator notation, $P_{xy} = 1_{x \leftrightarrow y}/n_x$.

In general it's a good idea to try to solve the detailed balance equations first, and only if that fails is it worth trying to solve the full stationarity equations. In this case the detailed balance equations do indeed work.

Can you find the solution for a general connected undirected graph?

Question 5. The state space is $\mathbb{R}$ which is not countable, so all the sum-based equations from lectures don't work. We have to go right back to the definition of stationarity: *a distribution π is a stationary distribution if*

$$X_0 \sim \pi \quad \Longrightarrow \quad X_1 \sim \pi.$$

Review the calculations in section 10.4 where we derived $\pi = \pi P$ for discrete state-space Markov chains, and think: can I use similar reasoning to derive the parameters of the Normal distribution that the question tells us is stationary?

Question 6. See section 10.6. The calculation is very simple.

The average transmission rate is $\hat{w}/\mathsf{RTT}$ packets per second, where $\hat{w}$ is the fixed point you found. This should roughly agree with the famous TCP throughput equation

$$\text{throughput} = \frac{\text{packet size}\sqrt{3/2}}{\mathsf{RTT}\sqrt{p}} \text{ bytes/sec}$$

which is in widespread use in network engineering. This famous formula involves slightly different assumptions (it assumes periodic losses rather than random) so your answer won't be exactly the same.

Question 7. Follow the same pattern as example 11.1.1. You should conclude that you need to solve a least-squares problem for the model

$$x_i \approx \mu(1-\lambda) - \lambda x_{i-1}.$$

This isn't a proper linear model, because proper linear models have to be "sum of unknown coefficient times feature vector". Rewrite it with different parameters (as we did in section 2.2.4 for a different model), use `sklearn` to compute those parameters, and translate back to get the parameters we want.

Question 8. This is a hidden Markov model, as mentioned in section 11.3:

$$\begin{array}{ccccccc} X_0 & \longrightarrow & X_1 & \longrightarrow & X_2 & \longrightarrow & \cdots \\ \downarrow & & \downarrow & & \downarrow & & \\ Y_0 & & Y_1 & & Y_2 & & \end{array}$$

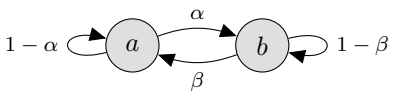
Part (a). The second equation requires the law of total probability (with baggage), and the third equation requires Bayes's rule (with baggage h). 'With baggage' is described in section 10.2. The idea of these manipulations is to put the probability expressions into a form where you can leverage memorylessness: "X_n is generated based *only* on X_{n-1}, and Y_n is generated based *only* on X_n".

Part (b). Let $\pi^{(n)}$ be the probability vector at timestep n. Compute $\pi^{(0)}$ from the first equation. Then, iteratively apply the next two equations, to compute $\pi^{(n)}$ from $\pi^{(n-1)}$. Your implementation should use two matrices, $P_{ij} = \mathbb{P}(X_n = j \mid X_{n-1} = i)$ and $Q_{xy} = \mathbb{P}(Y_n = y \mid X_n = x)$. The first is the transition matrix that we're used to from Markov Chains, and the second is called the emission matrix.

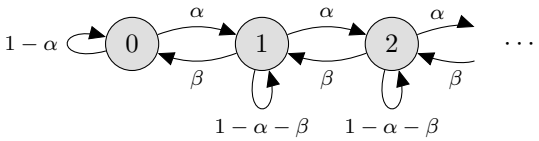
Supplementary questions

These questions are not intended for supervision (unless your supervisor directs you otherwise).

Question 10. Here is the state space diagram for a Markov chain. Find a stationary distribution. Is it unique?



Question 11. Here is the state space diagram for a Markov chain, with state space $\{0, 1, 2, \dots\}$. It is parameterized by α and β, with $0 < \alpha < \beta$ and $\alpha + \beta < 1$. Let $\pi_n = (1 - \alpha/\beta)(\alpha/\beta)^n$, $n \geq 0$. Show that π is a stationary distribution.



Question 12. Let $X_n \in \mathbb{N}$ be the number of infectious people on day n of an epidemic, and consider the Markov chain model

$$X_{n+1} = X_n + \text{Poisson}(rX_n/d) - \text{Bin}(X_n, 1/d).$$

We would like to compute the probability that, starting from state $X_0 = x$, the epidemic dies out i.e. hits state 0. In order to solve this by computer, we'll cut the state space down to $\{0, \dots, N\}$, for some sufficiently large N, by amalgamating all the states with $\geq N$ infected and letting the transition from state N back to itself have probability 1.

(a) Give pseudocode to compute the transition matrix. You should give your answer in terms of `binom.pmf` and `poisson.pmf`, the likelihood functions for the two distributions in question.

(b) Give pseudocode to compute the probability that the epidemic dies out, starting from any initial state $x \in \{0, \dots, N\}$. (For $r = 1.1$ and $d = 14$, for $X_0 = 50$, the probability is 0.7%.)

Question 13 (Google PageRank). Consider a directed acyclic graph representing the web, with one vertex per webpage, and an edge $v \to w$ if page v links to page w. Consider a random web surfer who goes from page to page according to the algorithm

```
d = 0.85
def next_page(v):
    neighbours = list of pages w such that v → w
    a = random.choice(['follow_link','teleport'], p=[d,1-d])
    if a=='follow_link' and len(neighbours) > 0:
        return random.choice(neighbours)
    else:
        V = list of all web pages
        return random.choice(V)
```

This defines a Markov chain. Explain why the chain is irreducible. Show that the stationary distribution π solves

$$\pi_v = \frac{1-d}{|V|} + d \sum_{u:u\to v} \frac{\pi_u}{|\Gamma_u|}$$

where $|V|$ is the total number of web pages in the graph, and $|\Gamma_u|$ is the number of outgoing edges from u.

Compute the stationary distribution for this random web surfer model, for the graph in lecture notes example 10.3.1. Repeat with $d = 0.05$. What do you expect as $d \to 0$? What do you expect if $d = 1$?

The equation for π_v defines a scaled version of PageRank, Google's original method for ranking websites.

Question 14 (Hitting times). Consider the random web surfer model, for the graph in lecture notes example 10.3.1. Let t_x be the expected time, starting from vertex x, until the surfer reaches Twitter (vertex 5). Derive the equations

$$t_x = 1 + \sum_y P_{xy} t_y \text{ for all } x \neq 5, \quad t_5 = 0$$

and solve with numpy.

Expected answer: [6.67, 5.25, 6.17, 5.08, 6.08, 0].

Question 15. The code from question 8(c) can fail with a divide-by-zero error. This is undesirable in production code! One way to fix the problem is to modify the Markov model to include a 'random teleport'—to express the idea 'OK, our inference has gone wrong somewhere; let's allow our location estimate to reset itself'. We can achieve this mathematically with the following model: with probability $1-\varepsilon$ generate the next state as per line 9, otherwise pick the next state uniformly from $\{0, 1, \ldots, \texttt{MAX\_STATE}\}$. Modify your code from question (b) to reflect this new model, with $\varepsilon = 0.01$.

Alternatively, we could fix the problem by changing the model to express 'OK, this reading is glitchy; let's allow the code to discard an impossible reading'. How might you change the Markov model to achieve this?

Question 16. The Markov model for motion from question 1 is called a *simple random walk (with boundaries)*; it chooses a direction of travel independently at every timestep. This is not a good model for human movement, since people tend to head in the same direction for a while before changing direction.

(a) Let $V_n \in \{-1, 0, 1\}$ be a Markov chain: let $V_{n+1} = V_n$ with probability 0.9, and let V_{n+1} be chosen uniformly at random from $\{-1, 0, 1\}$ with probability 0.1. Draw a state space diagram for this Markov chain.

Interpret V_n as the velocity of our moving object at timestep n, and let $X_{n+1} = \max(0, \min(9, X_n + V_n))$.

(b) Draw the state space diagram for (X_n, V_n).

(c) Give pseudocode to compute the stationary distribution.