Example sheet 4

Markov chains

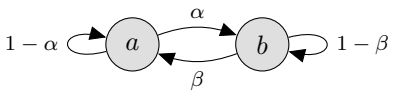
Foundations of Data Science—DJW—2019/2020

Question 1. For the Cambridge weather simulator, page 103 of lecture notes, show that

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

Explain your reasoning carefully.

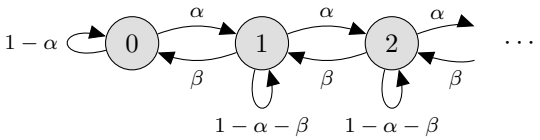
Question 2. Here is the state space diagram for a Markov chain. Find the stationary distribution.



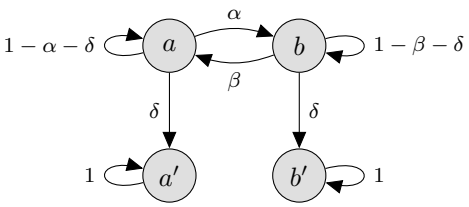
Question 3. For this Markov chain, draw the state space diagram, and give pseudocode to compute the stationary distribution.

```
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 4. Here is the state space diagram for a Markov chain, with state space $\{0, 1, 2, \dots\}$, 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 5. Here is the state space diagram for a Markov chain. It has two so-called *absorbing states*, a' and b': when it hits them it stays there for ever. The parameters are α, β, and δ, with $0 < \alpha < 1$, $0 < \beta < 1$, and $0 < \delta < \min(1 - \alpha, 1 - \beta)$.



Let ρ_x be the probability of hitting a' when starting from x,

$$\rho_x = \mathbb{P}\big(\text{eventually hit } a' \mid X_0 = x\big), \qquad \rho_{a'} = 1, \qquad \rho_{b'} = 0.$$

(a) Explaining your reasoning carefully, show that

$$\rho_a = (1 - \alpha - \delta)\,\mathbb{P}\big(\text{eventually hit } a' \mid X_1 = a\big) + \alpha\,\mathbb{P}\big(\text{eventually hit } a' \mid X_1 = b\big) + \delta$$
$$\rho_b = (1 - \beta - \delta)\,\mathbb{P}\big(\text{eventually hit } a' \mid X_1 = b\big) + \beta\,\mathbb{P}\big(\text{eventually hit } a' \mid X_1 = a\big)$$

(b) Explain briefly why $\mathbb{P}(\text{eventually hit } a' \mid X_1 = x)$ is equal to ρ_x.

(c) Calculate ρ_a.

Hints and comments

Question 1. In lecture 16 we calculated $\mathbb{P}(X_r = r \,|\, X_0 = g)$, using two tools: the law of total probability, and the memoryless nature of the causal diagram. Use the same method here. Start by using the law of total probability, conditioning on X_1. Then condition on X_2. (Or you can do X_2 first, or you can do both together. See example 9.6 in lecture notes.)

Question 2. The equations to solve are

$$\pi_x = \sum_y \pi_y P_{yx} \text{ for all } x \qquad \text{and} \qquad \sum_x \pi_x = 1.$$

In lecture 16 we derived these, but for answering questions you should just remember them and apply them. Also, in your aswer, you should mention *irreducibility* (lecture 16 slide 11, lecture notes page 108). You should end up with the answer

$$\pi_a = \frac{\beta}{\alpha+\beta} \qquad \pi_b = \frac{\alpha}{\alpha+\beta}.$$

Question 3. 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 typical states, and also edge cases, as in lecture 16 slide 7.

Your code should first of all create the transition matrix. Start with `numpy.zeros((10,10))`, and then fill in the values according to your state space diagram. Then give the code from lecture 16 slide 9, exercise 9.8 from lecture notes. For the exam you don't need to remember the syntax, but you do need to be able to describe it in enough detail for a first-year undergraduate to implement.

Question 4. You need to show that π satisfies the two equations from question 2. To show that $\sum_{n=0}^{\infty} \pi_n = 1$, either remember the formula for a geometric series ($\sum_{n=0}^{\infty} ar^n = 1/(1-r)$ for $|r| < 1$), or spot that π_n is the p.m.f. for a Geometric distribution and must necessarily sum to 1. (There are two flavours of the Geometric distribution. Wikipedia lists both.)

Question 5. Part (a) is like question 1: it uses the law of total probability, and the memoryless nature of the causal diagram. In this case, condition on X_1.

For part (b), it says "explain briefly", so give a one-sentence intuitive explanation rather than reaching for maths. There is in fact a general rule behind this — the rule is that, because a Markov chain uses the same procedure every timestep for generating the next value, it doesn't matter what time index it starts at. Example 9.7 from lecture notes gives another example.

Practical 4

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

Practical 4 can be found on Azure Notebooks, `prac4.ipynb`. In it, you will implement a particle filter for estimating the location of a moving object, given noisy readings. The following questions illustrate how the computation works, but in a simpler setting where it's possible to write out exact formulae. I recommend you answer these questions first, before embarking on the practical.

Consider the following code. It computes a sequence of x values, exactly the same as in question 3, and this sequence is a Markov chain. But we don't observe x directly, we only see noisy observations $x + e$. This is called a *hidden Markov model*.

```
def hmm():
    MAX_STATE = 9
    # Pick the initial state uniformly from
    x = numpy.random.randint(low=0, high=MAX_STATE+1)
    while True:
        e = numpy.random.choice([−1,0,1])
        yield min(MAX_STATE, max(0, x + e))
        d = numpy.random.choice([−1,0,1], p=[1/4,1/2,1/4])
        x = min(MAX_STATE, max(0, x + d))
```

Question 6. Let $X = (X_0, X_1, \dots)$ be the sequence of x values computed inside this code, and let $Y = (Y_0, Y_1, \dots)$ be the observations, where Y_n is X_n plus noise. Draw the state space diagram for X. Draw a causal diagram for $\{X_0, Y_0, X_1, Y_1, X_2, \dots\}$.

Question 7. Define $\pi_x^{(0)}$ and $\delta_x^{(0)}$ by

$$\delta_x^{(0)} = \mathbb{P}(X_0 = x) = 1/10 \quad \text{for } x \in \{0, \dots, 9\}$$
$$\pi_x^{(0)} = \mathbb{P}(X_0 = x \mid Y_0 = y).$$

(Since $\pi_x^{(0)}$ depends on y_0 we ought to write it as a function of x and y_0, but for the sake of conciseness we won't write out the y_0 dependency.) Use Bayes's rule to find a formula for $\pi_x^{(0)}$ in terms of $\delta^{(0)}$ and the matrix $Q_{xy} = \mathbb{P}(Y_n = y \mid X_n = x)$.

Question 8. Let $\delta_x^{(1)} = \mathbb{P}(X_1 = x \mid Y_0 = y_0)$. Use the law of total probability to find a formula for $\delta_x^{(1)}$ in terms of $\pi^{(0)}$ and the transition matrix $P_{xx'} = \mathbb{P}(X_{n+1} = x' \mid X_n = x)$.

Question 9. Let

$$\delta_x^{(n)} = \mathbb{P}(X_n = x \mid Y_0 = y_0, \dots, Y_{n-1} = y_{n-1})$$
$$\pi_x^{(n)} = \mathbb{P}(X_n = x \mid Y_0 = y_0, \dots, Y_{n-1} = y_{n-1}, Y_n = y_n)$$

(a) Show that

$$\pi_x^{(n)} = \frac{\delta_x^{(n)} Q_{xy}}{\sum_z \delta_z^{(n)} Q_{zy}} \qquad \text{and} \qquad \delta_x^{(n)} = \sum_z \pi_z^{(n-1)} P_{zx}.$$

(b) Write pseudocode for a function that takes as input a list of readings $y = [y_0, y_1, \dots, y_n]$ and outputs the vector $\pi^{(n)}$. Your pseudocode should include defining the P and Q matrices.

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

Supplementary questions

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

Question 10. The code from question 9 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 $\{0, 1, \ldots,$ `MAX_STATE`$\}$. Modify your code from question 9(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 11. The Markov model for motion that we are using 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.

(b) Interpret V_n as the velocity of our moving object at timestep n, and let $X_{n+1} = \max(0, \min(9, X_n + V_n))$. Update your code from question 9(b) to reflect this model.

Question 12 (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

```
1  d = 0.85
2  def next_page(v):
3      neighbours = list of pages w such that v → w
4      a = random.choice(['follow_link','teleport'], p=[d,1−d])
5      if a=='follow_link' and len(neighbours) > 0:
6          return random.choice(neighbours)
7      else:
8          V = list of all web pages
9          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 9.7. 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 13 (Sequential Bayes). I have a biased coin, with unknown probability of heads θ. I toss it n times, with outcomes $x_1, x_2, \ldots, x_n$ where $x_n = 1$ indicates heads and $x_n = 0$ indicates tails. My prior belief is $\Theta \sim \text{Uniform}[0, 1]$. Here are two approaches to applying Bayes's rule:

- *One-shot Bayes.* Use Bayes's rule to compute the posterior of Θ, given data $(x_1, \ldots, x_n)$, using prior $\Theta \sim \text{Uniform}[0, 1]$, and assumimg that coin tosses are independent.

- *Sequential Bayes.* Use Bayes's rule to compute the posterior of Θ given data x_1, using the uniform prior; let the posterior density be $p_1(\theta)$. Apply Bayes's rule again to compute the posterior of Θ given data x_2, but this time using $p_1(\theta)$ as the prior; let the posterior density be $p_2(\theta)$. Continue applying Bayes's rule in this way, until we have found $p_n(\theta)$.

State the posterior distribution found by one-shot Bayes. Prove by induction on n that sequential Bayes gives the same answer.

Sequential Bayes and one-shot Bayes give the same answer for any inference problem, not just this coin-tossing example. Can you prove the general case?

Question 14 (Thompson sampling). In reinforcement learning, an agent typically has a choice between exploring possible moves versus exploiting the best moves that have been learnt so far. Here is a concrete example:

A compulsive gambler has a choice of two machines to play. The first has probability θ_1 of paying out, the second has probability θ_2, and all payouts are £10. The gambler doesn't know the values of θ_1 and θ_2, so treats them as unknown parameters, both with uniform prior distributions. The gambler adopts the following strategy (known as *Thompson sampling*) for choosing which machine to play:

1. Find the posterior distribution of (Θ_1, Θ_2) given the number of wins and losses on each machine so far;
2. Generate a single sample (θ_1, θ_2) from this distribution;
3. Play machine 1 if $\theta_1 \geq \theta_2$, and play machine 2 otherwise;
4. Repeat.

After n plays, let $w_1^{(n)}$ be the number of wins on machine 1, $\ell_1^{(n)}$ the number of losses, and $w_2^{(n)}$ and $\ell_2^{(n)}$ likewise for machine 2.

(a) Find the posterior distribution of (Θ_1, Θ_2) given $(w_1^{(n)}, \ell_1^{(n)}, w_2^{(n)}, \ell_2^{(n)})$.

(b) Let $X_n = (w_1^{(n)}, \ell_1^{(n)}, w_2^{(n)}, \ell_2^{(n)})$. Explain why $(X_0, X_1, \dots)$ is a Markov chain, and sketch the state space diagram.

(c) Implement the Thompson sampling strategy, and simulate it for the case where the true values are $\theta_1 = 0.1$ and $\theta_2 = 0.05$. Plot the posterior histograms for Θ_1 and Θ_2 after 1, 50, 100, 500, 1000 plays.

(d) Describe your findings. How does Thompson sampling resolve the explore/exploit tradeoff?