Homework 10

1. In this question, we will see how to use Matlab to try to see whether a stopping time is a strong stationary time.

   Recall that if we look at our walk at a strong stationary time, conditioned on that time being any particular value of m, then the walk should be distributed precisely as the stationary distribution. Therefore, we will simulate running our walk until the stopping time, and see how the result is distributed if the stopping time is precisely m.

   We will be doing the example of the stopping times for the random transposition walk from the last homework: in particular, times τ_1 and τ_3.

   You will need to use if statements and while loops, and break statements. You should look these up in the official Matlab documentation as well as Googling them to find helpful examples.

   (a) [2 pts] Write a program that generates two random permutations, then checks whether they are the same. If they are, output 1; if they are not, output 0.

   Note To check whether the two vectors v and w are equal, you can use isequal(v, w).

   (b) [2 pts] Write code that uses a for loop, an if statement and a break to find the index of the label 4 in a permutation. Test it using a random permutation.

   (c) [5 pts] Let per be the permutation [1 2 3 4] and define the vector indicesUsed to be the zero vector of length 4. Write code which swaps a random pair of elements in per, and sets indicesUsed at i and j to 1, where i and j are the labels of the elements swapped.

   (d) [5 pts] Now, run the code in part (c) until every single label has been used in a transposition at least once – that is, until the time τ_1. You will want to use the vector indicesUsed as defined above. You can do this by either

   - Using a for loop and a break at the stopping time.
   - Using a while loop and a counter t which you increase by 1 at each iteration of the loop, to keep track of the time you're at

   Output the permutation you stop at, as well as the time you stopped.

   Note: The negation of any statement can be accomplished by putting either a $\sim$ or not in front of it.

   (e) [1 pt] Initialize the matrix persAtStop to be the empty 0×4 matrix by letting persAtStop = zeros(0, 4). (We will be testing for uniformity conditioning on τ_1 being equal to stopT. This matrix will eventually contain a list of the permutations we stop at if $\tau_1 = \text{stopT}$).

Note: To test whether your code is working, you will have to pick a value for stopT, run it, and see if your answers make sense.

(f) [5 pts] Now, run the code in (d) until the minimum of stopT + 1 and τ_1. Either do this with making your for loop run until stopT + 1, inserting a break into your while loop depending on the counter t, or inserting a condition into your while loop dependent on t.

(g) [5 pts] Let k be a variable. Run the code in (f) k times; at the end of each iteration, check whether τ_1 was equal to stopT. If it was, and we stopped at the permutation per, adjoin per as the last row to the matrix persAtStop by using the command persAtStop = [persAtStop; per].

(h) [5 pts] The rows of matrix persAtStop now are now the permutations corresponding to τ_1 = stopT. Define the matrix L to be all the permutations of [1 2 3 4] using the perms command. Create a vector freq, such that freq(i) is the number of times the permutation L(i) appears in persAtStop.

Hint: You can do this by cycling through the matrix persAtStop, and using the ismember function at each step, but this is very slow. Instead, use the sortrows function on persAtStop first – this will make sure that identical permutations appear next to each other. Figure out a way to only look up each permutation once!

(i) [5 pts] The percentage of the time each permutation appeared at τ_1 when τ_1 = stopT will be represented by freq/sum(freq). Run your code for a number of values of stopT and k. Are the distributions you get consistent with τ_1 being a strong stationary time?

(Note that with simulation we're not getting perfect samples of the distribution, so it won't be exactly uniform even if τ_1 is a strong stationary time. However, the more simulations you do, the closer it should get.)

(j) [10 pts] Change your code slightly so that it runs until time τ_3 (it's only a couple of lines in part (c)). Run the experiments in part (i) again. Do your answers change?

2. In this question, we will use the path method to find a bound on the mixing time of the random transposition walk. Let α and β be permutations in S_n, and let us define a path $\Gamma_{\alpha\beta}$ for the pair. This path goes 'left-to-right' – at each step, swap in the leftmost entry that doesn't agree into the correct place. For example, to get from 34512 to 12435 we do the following

$$34512 \to 14532 \to 12534 \to 12435$$

We will start by doing an example, then proceed to the bounds for the walk.

(a) Consider $n = 7$, let e be the edge $(1243657, 1245637)$, and let α and β be two permutations such that e is on the path $\Gamma_{\alpha\beta}$. For simplicity of notation we will write

$$\alpha = \alpha_1\alpha_2 \dots \alpha_7$$
$$\beta = \beta_1\beta_2 \dots \beta_7$$

i. [2 pts] Show that $\beta_1 = 1, \beta_2 = 2, \beta_3 = 4$ and $\beta_4 = 5$. Conclude that there are 3! possible choices for β.

ii. [3 pts] If α satisfies $\alpha_2 = 1, \alpha_3 = 4$, and $\alpha_6 = 2$, solve for α. **Hint:** What are the steps of the path $\Gamma_{\alpha\beta}$ leading up to e?

iii. [3 pts] As in part ii., argue that to solve for α it suffices to know the positions of the cards labelled $1, 2$ and 4. Conclude that there are $7 \cdot 6 \cdot 5$ choices for α.

iv. [2 pts] Finally, show that there are at most 7! pairs (α, β) such that $\Gamma_{\alpha\beta}$ goes through the edge e.

(b) [5 pts] Moving to the general case, let $e = (\omega_1, \omega_2)$ be an edge in the random transposition walk on n cards. Show that there are at most $n!$ pairs of permutations (α, β) such that $\Gamma_{\alpha\beta}$ goes through the edge e.

(c) [2 pts] Show that for any pair of permutations α, β, $|\Gamma_{\alpha\beta}| \leq n$. (As in class, we use the absolute value bars to indicate the length of the path.)

(d) [5 pts] Use parts (b) and (c) to find a bound on the spectral gap of the random transposition walk.

(e) [5 pts] For this walk, the absolute spectral gap is equal to the spectral gap (you're just going to have to believe me on this.) Use this fact, as well as part (d), to find a bound on the mixing time of the random transposition walk.