HW2_Brynestad_Cal

January 7, 2022

1 Homework 2: Discrete Probability

1.0.1 Name: Cal Brynestad

This assignment is due on Canvas by **6:00PM on Friday September 24th**. Your solutions to theoretical questions should be done in Markdown directly below the associated question. Your solutions to computational questions should include any specified Python code and results as well as written commentary on your conclusions. Remember that you are encouraged to discuss the problems with your classmates, but **you must write all code and solutions on your own**.

NOTES:

- If you're not familiar with typesetting math directly into Markdown then by all means, do your work on paper first and then typeset it later. Remember that there is a reference guide linked on Canvas on writing math in Markdown. **All** of your written commentary, justifications and mathematical work should be in Markdown.
- Because you can technically evaluate notebook cells is a non-linear order, it's a good idea to do Kernel → Restart & Run All as a check before submitting your solutions. That way if we need to run your code you will know that it will work as expected.
- It is **bad form** to make your reader interpret numerical output from your code. If a question asks you to compute some value from the data you should show your code output **AND** write a summary of the results in Markdown directly below your code.
- This probably goes without saying, but… For any question that asks you to calculate something, you **must show all work and justify your answers to receive credit**. Sparse or nonexistent work will receive sparse or nonexistent credit.

---

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pylab as plt
%matplotlib inline
```

---

2 [25 points] Problem 1:

As part of their holiiday celebrations, Luna and her cousin are having a cinnamon stick duel. The rules of the cinnamon stick duel are as follows: * The duel is composed of a number of rounds. * Each round consists of Luna trying to land a cinnamon stick on its end (aka "swirly edge"), then Luna's cousin trying to land a cinnamon stick it's "swirly edge". * If Luna manages to land the cinnamon stick on its end, then Luna is awarded a point, and similarly if Luna's cousin manages too land the cinnamon stick on its end, then Luna's cousin is awarded a point. * If a round ends in a tie, then Luna and cousin proceed to another round. If a round ends with one person scoring a point but the other one does not, then the person who scored a point wins the duel.

Let L be the event that "Luna wins the duel", E be the event that "the duel ends after the first round of cinnamon stick flips", and C be the event that "the duel continues after the first round of cinnamon stick flips". Suppose that the result of one cinnamon stick flip is independent of the other cinnamon stick flip, Luna successfully lands their flips with probability p, and Luna's cousin lands their flips with probability q.

Part A (1 Point): The duel will continue until Luna or cousin wins. What is the sample space Ω for this "experiment"? Represent the outcomes of individual cinnamon stick flips by S for a flip that lands on the end ("swirly" part) and B for a miss (the stick lands on its long bark side), and assume that Luna has the first flip in each round. At the very least, your answer should include all of the events in Ω that include one or two rounds.

Part A Answer here:

Solution: $\Omega = \{[S, B], [B, S], [(S, S), (S, B)], [(S, S), (B, S)], [(B, B), (S, B)], [(B, B), (B, S)], \ldots\}$ Ω is an infinite sample space.

Part B (5 Points): Determine $P(L \mid E)$ in terms of p and q.

Part B Answer here:

Solution: $P(L \mid E) = \frac{P(L \cap E)}{P(E)} = \frac{P(\{S,B\})}{P(\{S,B\} \cup \{B,S\})} = \frac{p(1-q)}{p(1-q)+q(1-p)} = \frac{p(1-q)}{p-pq+q-pq} = \frac{p(1-q)}{p+q-2pq}$

Part C (4 Points): Explain why events C and L are independent.

Part C Answer here:

Solution: We know that the two events are independent if we can show that $P(C \mid L) = P(C)$, $P(L \mid C) = P(L)$, or $P(C \cap L) = P(C) \cdot P(L)$. Take for example, $P(L \mid C)$, or the probability that Luna wins the duel, given that the duel continues after the first round of cinnamon stick flips. Let's say that the probability of $P(L)$ is p_1, and the conditional probability of $P(L \mid C)$ is p_2. If p_2 is equal to p_1 we know that the two events are independent of each other, as the extra information provided by C in $P(L \mid C)$ would have to have had no effect on the original probability of $P(L)$. So, in our example, the probability that Luna wins the duel, p_1, is the same as the probability that Luna wins the duel, given that the duel continues after the first round of cinnamon stick flips, or p_2, because for each given round that Luna may play, that round number has no effect on, (is independent of), her odds of winning that round and thereby winning the duel. Therefore, as p_2 = p_1, the events C and L are independent.

Part D (8 Points): Use the Law of Total Probability to show that $P(L) = p(1-q) + (2pq - p - q + 1)P(L \mid C)$.

Part D Answer here:

Solution: $P(L) = P(L \cap E) + P(L \cap C) = P(L \cap E) + P(L \mid C) \cdot P(C) = p(1-q) + P(L \mid C) \cdot P(C)$ (from Part B we know $P(L \cap E)$) We know that $P(E)$ and $P(C)$ are disjoint events whose probabilities sum to 1. Therefore: $P(C) = 1 - P(E) = 1 - (p + q - 2pq) = 2pq - p - q + 1$ (from Part B we know $P(E)$) So we now have: $P(L) = p(1-q) + P(L \mid C) \cdot (2pq - p - q + 1)$ or $P(L) = p(1-q) + (2pq - p - q + 1) \cdot P(L \mid C)$

Part E (5 Points): Use the fact from **Part C** that $P(L \mid C) = P(L)$ to determine $P(L)$ in terms of p and q.

Part E Answer here:

Solution: $P(L) = p(1-q) + (2pq - p - q + 1) \cdot P(L \mid C)$ $P(L) = p(1-q) + (2pq - p - q + 1) \cdot P(L)$ $1 = \frac{p(1-q)}{P(L)} + (2pq - p - q + 1)$ $1 - 2pq + p + q - 1 = \frac{p(1-q)}{P(L)}$ $-2pq + p + q = \frac{p(1-q)}{P(L)}$ $P(L) = \frac{p(1-q)}{p+q-2pq}$

Part F (2 Points): Explain why the answers to **Part B** and **Part E** are the same.

Part F Answer here:

Solution: In Part C we showed that events C and L are independent because $P(L \mid C) = P(L)$. For the same reasons as in Part C, events E and L are independent as $P(L \mid E) = P(L)$. No matter how many rounds Luna and her cousin have to play to determine the winner of the duel, Luna's prrobability of winning the duel will stay the same as this probability is independent of the round they are playing. So we have that $P(L \mid C) = P(L \mid E) = P(L)$, which explains why the answers to Part B and Part E are the same.

---

3 [25 points] Problem 2:

You are at a family gathering and have been sent to watch your baby nephew. You enter the room to see him opening a book to a random page, giggling a bit, then picking another book to do the same. You leave him alone to enjoy the simple fun of childhood, but you can't help but ask yourself "what are the chances he'd just so happen to open these specific pages in these books?" So, you do the math.

He has a total of 50 books, 28 with six pages, 18 with ten pages, and 4 with twenty pages. We will use S for the event of him picking a book with Six pages, T for the event of him picking a book with Ten pages, and W for the event of him picking a book with tWenty pages. There is also no logic behind his choices (being a baby and all), so assume that all of these colorful books are equally likely to be chosen.

Please be sure to show all of your work by hand unless otherwise stated, and round your answer to a max of 4 decimal places.

Part A (3 Points): What is the probability of your nephew opening a book to page 8, or P_8?

Part A Answer here:

Solution: $P(P_8) = P(P_8 \cap S) + P(P_8 \cap T) + P(P_8 \cap W) = P(P_8 \mid S) \cdot P(S) + P(P_8 \mid T) \cdot P(T) + P(P_8 \mid W) \cdot P(W) = 0 + \frac{1}{10} \cdot \frac{18}{50} + \frac{1}{20} \cdot \frac{4}{50} = \frac{18}{500} + \frac{4}{1000} = \frac{18}{500} + \frac{2}{500} = \frac{20}{500} = \frac{2}{50} = \frac{1}{25} = 4\%$

Part B (3 Points): He just turned a book to page 8 (P_8). What was the probability that he chose a book with 10 pages (T)?

Part B Answer here:

Solution: $P(T \mid P_8) = \frac{P(T \cap P_8)}{P(P_8)} = \frac{\frac{18}{50} \cdot \frac{1}{10}}{\frac{1}{25}} = \frac{\frac{18}{500}}{\frac{1}{25}} = \frac{18}{500} \cdot \frac{25}{1} = \frac{450}{500} = \frac{9}{10}$

Part C (3 Points): Are P_8 and T independent from one another? Use math to test your hypothesis.

Part C Answer here:

Solution: P_8 and T are independent if we can show that $P(P_8 \mid T) = P(P_8)$, $P(T \mid P_8) = P(T)$ or $P(P_8 \cap T) = P(P_8) \cdot P(T)$. $P(P_8 \mid T) = \frac{P(P_8 \cap T)}{P(T)} = \frac{\frac{1}{10} \cdot \frac{18}{50}}{\frac{18}{50}} = \frac{\frac{18}{500}}{\frac{18}{50}} = \frac{18}{500} \cdot \frac{50}{18} = \frac{50}{500} = \frac{1}{10}$ $P(P_8) = \frac{1}{25}$ so $P(P_8 \mid T)$ does not equal $P(P_8)$. $P(T \mid P_8) = \frac{P(T \cap P_8)}{P(P_8)} = \frac{\frac{18}{50} \cdot \frac{1}{10}}{\frac{1}{25}} = \frac{\frac{18}{500}}{\frac{1}{25}} = \frac{18}{500} \cdot \frac{25}{1} = \frac{450}{500} = \frac{9}{10}$ $P(T) = \frac{18}{50}$ so $P(T \mid P_8)$ does not equal $P(T)$. $P(P_8 \cap T) = P(P_8 \mid T) \cdot P(T) = \frac{1}{10} \cdot \frac{18}{50} = \frac{18}{500} = \frac{9}{250}$ $P(P_8) \cdot P(T) = \frac{1}{25} \cdot \frac{18}{50} = \frac{18}{1250}$ so $P(P_8 \cap T)$ does not equal $P(P_8) \cdot P(T)$. Given that none of what we wanted to prove true, is in fact true, we know that the two events P_8 and T are not independent from one another.

Part D (4 Points): Now consider the following scenario. We will use the event E_1 to represent the child picking any book and opening it to any page 1-6.

What is the probability that he picked up a book with six pages (S) given that he opens the book to one of the first 6 pages. In other words, find $P(S|E_1)$.

Part D Answer here:

Solution: $P(S|E_1) = \frac{P(S \cap E_1)}{P(E_1)}$ $P(E_1) = P(E_1 \cap S) + P(E_1 \cap T) + P(E_1 \cap W) = P(E_1|S) \cdot P(S) + P(E_1|T) \cdot P(T) + P(E_1|W) \cdot P(W) = 1 \cdot \frac{28}{50} + \frac{6}{10} \cdot \frac{18}{50} + \frac{6}{20} \cdot \frac{4}{50} = \frac{28}{50} + \frac{108}{500} + \frac{24}{1000} = \frac{280}{500} + \frac{108}{500} + \frac{12}{500} = \frac{400}{500} = \frac{4}{5}$ $P(S \cap E_1) = \frac{28}{50} \cdot 1 = \frac{28}{50}$ Given we have found $P(E_1)$ and $P(S \cap E_1)$, we know that: $P(S|E_1) = \frac{P(S \cap E_1)}{P(E_1)} = \frac{\frac{28}{50}}{\frac{4}{5}} = \frac{28}{50} \cdot \frac{5}{4} = \frac{140}{200} = \frac{7}{10}$

---

4 [25 points] Problem 3:

You and your friend have decided to explore a bit more of Boulder. You decide to stick to shopping areas around campus, just to keep yourselves from getting lost. This means you have access to the University of Colorado campuses, being U for the main campus and E for East Campus. On top of that, you can go to the Pearl Street Mall with its wide variety of specialty shops (P), The 29th Street Mall with its more practical stores (M), The Hill with its variety of unique restaurants (H), The more average variety of stores and restaurants around Baseline (B), and the more quaint shops around Table Mesa (T).

The two of you also decided on a couple of rules to get you out of your comfort zones:

- Your journey must include 3 locations. Begin at location A, proceed to location B, end at location C.
- The two of you will use a random number generator to choose which place you will go to next. This means an equal probability to go to each place. That way, you will not just go to your favorite coffee shop over and over again, even if that is what you want to happen.

- You and your friend will be going to these places **independently** of one another. That is to say, you will go off on your own to explore with your own random number generator. That way, you will have different adventures to tell each other over the phone later.
- You can only move to the next location based on connections on the following graph. That way, you cannot just skip a location.
- The bus system in Boulder is so good, you can assume similar commute times no matter if you are walking or using the bus.
- The random number generator is dumb, i.e. it doesn't think like a human, so it is just as possible to go back where you came. That means you can go, say, from the main campus to east campus and back to the main campus in this system ($U \to E \to U$). It is a small flaw, but you keep it in because it is funny to laugh at each other when it happens. BTW, It is not possible to "go to" your current location ($U \to U$ is not possible)

Part A (3 Points): You start at the East Campus after having a class there. You would like to get back to the main campus to get some burgers at the UMC, but you want to go somewhere else first. Your friend reminds you that you have to follow the rules, so you pull out your random number generator and get going.

Calculate the probability that you end up at the main campus (U) from the East Campus (E) after going to one other location first.

Note: do Parts A-C by hand (or at most a calculator, but plot the paths by hand), rounding to 4 decimal places. You will be confirming your answers by code in Part D.

Part A Answer here:

Solution: There are two ways to get go from (E) to another location first, and then to (U). These two ways are ($E \to M \to U$) and ($E \to B \to U$). The sum of the probabilities of these two things happening is the probability that you end up at the main campus (U) from the East Campus (E) after going to one other location first. $P(E \to M \to U) = \frac{1}{4} \cdot \frac{1}{3} = \frac{1}{12}$ $P(E \to B \to U) = \frac{1}{4} \cdot \frac{1}{4} = \frac{1}{16}$ $\frac{1}{12} + \frac{1}{16} = \frac{4}{48} + \frac{3}{48} = \frac{7}{48} =$ .145833 or 14.5833%

Part B (3 Points): Your friend, on the other hand, wants to pass thru or end up at Baseline to chat with his Barista friend at the Baseline Starbucks. He knows the rules and must have a journey of 3 locations.

Calculate the probability that your friend is able to pass thru or end at Baseline (B) after leaving East Campus (E) in the first or second leg of his journey.

Part B Answer here:

Solution: There are six ways to pass thru or end up at Baseline to chat with his Barista friend at the Baseline Starbucks. These six ways are ($E \to T \to B$), ($E \to U \to B$), ($E \to B \to U$), ($E \to B \to H$), ($E \to B \to T$) and ($E \to B \to E$). The sum of the probabilities of these six things happening is the probability that our friend is able to pass thru or end at Baseline (B) after leaving East Campus (E) in the first or second leg of his journey. $P(E \to T \to B) = \frac{1}{4} \cdot \frac{1}{2} = \frac{1}{8}$ $P(E \to U \to B) = \frac{1}{4} \cdot \frac{1}{5} = \frac{1}{20}$ $P(E \to B \to U) = \frac{1}{4} \cdot \frac{1}{4} = \frac{1}{16}$ $P(E \to B \to H) = \frac{1}{4} \cdot \frac{1}{4} = \frac{1}{16}$ $P(E \to B \to T) = \frac{1}{4} \cdot \frac{1}{4} = \frac{1}{16}$ $P(E \to B \to E) = \frac{1}{4} \cdot \frac{1}{4} = \frac{1}{16}$ $\frac{1}{8} + \frac{1}{20} + \frac{1}{16} + \frac{1}{16} + \frac{1}{16} + \frac{1}{16} = \frac{10}{80} + \frac{4}{80} + \frac{20}{80} = \frac{34}{80} =$ .425 or 42.5%

Part C (6 Points): A couple days later, you decide to explore a little more. You just got some refreshing bubble tea from the Hill (H) and have some time to look around. At the same time, your friend decides to do the same, having just left the Whole Foods at Table Mesa (T) with a dragonfruit in hand.

- 1. What is the probability the two of you will meet at the first location you go to?
- 2. What is the probability that both of you end up at the 29th Street Mall (M) on your second move?

Part C Answer here:

Solution: 1) The only place that the two of us could meet at is (B). Let (B_1) be the probability that the first location I travel to from H is (B) and let (B_2) be the probability that the first location my friend travels to from (H) is (B). So, the probability the two of us will meet at the first location we go to is the $P(B_1 \cap B_2)$. $P(B_1 \cap B_2) = P(B_1) \cdot P(B_2)$ as these two events are independent. $P(B_1) \cdot P(B_2) = \frac{1}{3} \cdot \frac{1}{2} = \frac{1}{6} = .166666$ or 16.6666% 2) There are two ways that I could end up at the 29th Street Mall (M) on my second move, $(H \to P \to M)$ or $(H \to U \to M)$. There is one way that my friend could end up at the 29th Street Mall (M) on their second move, $(T \to E \to M)$. Lets calculate the probabilities that I end up at (M) and that my friend ends up at (M), both on our second moves. My probability: $P(H \to P \to M) = \frac{1}{3} \cdot \frac{1}{3} = \frac{1}{9}$ $P(H \to U \to M) = \frac{1}{3} \cdot \frac{1}{5} = \frac{1}{15}$ The sum of the probabilities of these two things happening is the probability that I end up at (M) on my second move: $\frac{1}{9} + \frac{1}{15} = \frac{5}{45} + \frac{3}{45} = \frac{8}{45}$ My friends probability: $P(T \to E \to M) = \frac{1}{2} \cdot \frac{1}{4} = \frac{1}{8}$ Now let (M_1) be the probability that I travel to (M) on my second move and let (M_2) be the probability that my friend travels to from (M) on his second move. The probability the two of us will meet at at (M) on both of our second moves is the $P(M_1 \cap M_2)$. $P(M_1 \cap M_2) = P(M_1) \cdot P(M_2)$ as these two events are independent. $P(M_1) \cdot P(M_2) = \frac{8}{45} \cdot \frac{1}{8} = \frac{8}{360} = \frac{1}{45} = .022222$ or 2.2222%

Part D (11 Points): Write some code to affirm your answers from Parts A, B, and C. Be sure to run each test enough times to make sure it comes close to your probability estimates (10000 should be fair, but you could always play with more or less to see how the accuracy changes).

[2]:
```
#Code answer here
U = {'nodes' : np.array(["B", "H", "P", "M", "E"]), 'probs' : np.array([1/5, 1/
↪5, 1/5, 1/5, 1/5])}
E = {'nodes' : np.array(["M", "U", "B", "T"]), 'probs' : np.array([1/4, 1/4, 1/
↪4, 1/4])}
B = {'nodes' : np.array(["H", "U", "E", "T"]), 'probs' : np.array([1/4, 1/4, 1/
↪4, 1/4])}
M = {'nodes' : np.array(["P", "U", "E"]), 'probs' : np.array([1/3, 1/3, 1/3])}
H = {'nodes' : np.array(["P", "U", "B"]), 'probs' : np.array([1/3, 1/3, 1/3])}
P = {'nodes' : np.array(["H", "U", "M"]), 'probs' : np.array([1/3, 1/3, 1/3])}
T = {'nodes' : np.array(["B", "E"]), 'probs' : np.array([1/2, 1/2])}

first_picks = {'firsts' : np.array([M, U, B, T]), 'probs' : np.array([1/4, 1/4,␣
↪1/4, 1/4])}

def pick_from_E(first_picks):
    # randomly choose a node adjacent to E
    node = np.random.choice(first_picks['firsts'], p = first_picks['probs'])
```

```
    # randomly choose a node adjacent to our first pick
    return np.random.choice(node['nodes'], p = node['probs'])
    #returns a node name from first nodes adjacency list

def prob_of_U(node_desired, first_picks, num_samples):
    #find a bunch of end points
    end_points = np.array([pick_from_E(first_picks) for ii in␣
 ↪range(num_samples)])
    #compute fraction of desired endpoints
    return np.sum(end_points == node_desired) / num_samples
# reurn the fraction of endpoints that were the desired endpoint
```

[3]:
```
print("A) The probability that you end up at the main campus (U) from the East␣
 ↪Campus (E) after going to one other location first is approximately {:.4f}".
 ↪format(prob_of_U("U", first_picks, num_samples=30000)))
```

```
A) The probability that you end up at the main campus (U) from the East Campus
(E) after going to one other location first is approximately 0.1442
```

[4]:
```
U = {'nodes' : np.array(["B", "H", "P", "M", "E"]), 'probs' : np.array([1/5, 1/
 ↪5, 1/5, 1/5, 1/5])}
E = {'nodes' : np.array(["M", "U", "B", "T"]), 'probs' : np.array([1/4, 1/4, 1/
 ↪4, 1/4])}
B = {'nodes' : np.array(["H", "U", "E", "T"]), 'probs' : np.array([1/4, 1/4, 1/
 ↪4, 1/4])}
M = {'nodes' : np.array(["P", "U", "E"]), 'probs' : np.array([1/3, 1/3, 1/3])}
H = {'nodes' : np.array(["P", "U", "B"]), 'probs' : np.array([1/3, 1/3, 1/3])}
P = {'nodes' : np.array(["H", "U", "M"]), 'probs' : np.array([1/3, 1/3, 1/3])}
T = {'nodes' : np.array(["B", "E"]), 'probs' : np.array([1/2, 1/2])}

first_picks = {'firsts' : np.array([M, U, B, T]), 'probs' : np.array([1/4, 1/4,␣
 ↪1/4, 1/4])}

def pick_from_E(first_picks):
    # randomly choose a node adjacent to E
    node = np.random.choice(first_picks['firsts'], p = first_picks['probs'])

    # randomly choose a node adjacent to our first pick
    return np.random.choice(node['nodes'], p = node['probs'])
    #returns a node name from first nodes adjacency list

def prob_of_B_on_second(node_desired, first_picks, num_samples):
    #find a bunch of end points
    end_points = np.array([pick_from_E(first_picks) for ii in␣
 ↪range(num_samples)])
    #compute fraction of desired endpoints
    return np.sum(end_points == node_desired) / num_samples
```

```
    # return the fraction of endpoints that were the desired endpoint

def prob_of_B_on_first(node_desired_array, first_picks, num_samples):
    #calculates probability B is the virst visit
    sum=0

    for ii in range(num_samples):
        node = np.random.choice(first_picks['firsts'], p = first_picks['probs'])
        #pick node from E's adjacent nodes

        if np.array_equal(node, node_desired_array):
            sum+=1
            #if it's B increment sum

    return sum / num_samples
    #return fraction of times we visit B from E

print("B) The probability that your friend is able to pass thru or end at␣
 ↪Baseline (B) after leaving East Campus (E) in the first or second leg of his␣
 ↪journey is approximately {:.4f}".format(prob_of_B_on_second("B", first_picks,␣
 ↪num_samples=30000)+ prob_of_B_on_first(B, first_picks, num_samples=30000)))
```

```
B) The probability that your friend is able to pass thru or end at Baseline (B)
after leaving East Campus (E) in the first or second leg of his journey is
approximately 0.4198
```

[5]:
```
H = {'nodes' : np.array(["P", "U", "B"]), 'probs' : np.array([1/3, 1/3, 1/3])}
T = {'nodes' : np.array(["B", "E"]), 'probs' : np.array([1/2, 1/2])}

def prob_of_meet(num_samples):
    sum=0
    for ii in range(num_samples):
        myFirstPick = np.random.choice(H['nodes'], p = H['probs'])
        friendsFirstPick = np.random.choice(T['nodes'], p = T['probs'])

        if myFirstPick == friendsFirstPick:
            sum+=1

    return sum / num_samples

print("C) The probability that me and my friend will meet at the first location␣
 ↪we go to is approximately {:.4f}".format(prob_of_meet(50000)))
```

```
C) The probability that me and my friend will meet at the first location we go
to is approximately 0.1649
```

[6]:
```
U = {'nodes' : np.array(["B", "H", "P", "M", "E"]), 'probs' : np.array([1/5, 1/
 ↪5, 1/5, 1/5, 1/5])}
```

```
E = {'nodes' : np.array(["M", "U", "B", "T"]), 'probs' : np.array([1/4, 1/4, 1/
↪4, 1/4])}
B = {'nodes' : np.array(["H", "U", "E", "T"]), 'probs' : np.array([1/4, 1/4, 1/
↪4, 1/4])}
M = {'nodes' : np.array(["P", "U", "E"]), 'probs' : np.array([1/3, 1/3, 1/3])}
H = {'nodes' : np.array(["P", "U", "B"]), 'probs' : np.array([1/3, 1/3, 1/3])}
P = {'nodes' : np.array(["H", "U", "M"]), 'probs' : np.array([1/3, 1/3, 1/3])}
T = {'nodes' : np.array(["B", "E"]), 'probs' : np.array([1/2, 1/2])}

my_first_picks = {'my_firsts' : np.array([P, U, B]), 'probs' : np.array([1/3, 1/
↪3, 1/3])}
friends_first_picks = {'friends_firsts' : np.array([B, E]), 'probs' : np.
↪array([1/2, 1/2])}

def prob_we_meet(num_samples, destination):
    sum=0
    for ii in range(num_samples):
        myFirstPick = np.random.choice(my_first_picks['my_firsts'], p =␣
↪my_first_picks['probs'])
        mySecondPick = np.random.choice(myFirstPick['nodes'], p =␣
↪myFirstPick['probs'])

        friendsFirstPick = np.random.
↪choice(friends_first_picks['friends_firsts'], p = friends_first_picks['probs'])
        friendsSecondPick = np.random.choice(friendsFirstPick['nodes'], p =␣
↪friendsFirstPick['probs'])

        if ((mySecondPick == friendsSecondPick) and (mySecondPick ==␣
↪destination)):
            sum+=1

    return sum / num_samples

print("C) The probability that both me and my friend end up at the 29th Street␣
↪Mall (M) on both our second moves is approximately {:.4f}".
↪format(prob_we_meet(30000, "M")))
```

```
C) The probability that both me and my friend end up at the 29th Street Mall (M)
on both our second moves is approximately 0.0219
```

Part E (2 Points): Describe what is happening in your code. Does it accurately show what you and your friend are doing? How so?

Part E Answer here:

Solution: My code does accurately show what me and my friend are doing. In my for loop, one experimental run, me and friend both first pick a location, or node, that is adjacent to our starting points, (*H*) and (*T*) respectively. These first picks are picked randomly just as me and my friend use the random number generator to see where we go. After the first picks are made, the code then randomly picks a location, or node, that is adjacent to our first picks, and again this second pick is random. The code then checks after both our second picks whether or not we both ended up at location (M). These experimental runs are done 30000 times in the code and then the amount of times me and my friend happened to meet at (M) after both our second picks is divided by the number of total experimental runs to determine the probaility that we meet at (M) after both our second picks.

---

5 [25 points] Problem 4:

Suppose you roll two fair six-sided dice. Let C be the event that the two rolls are *close* to one another in value, in the sense that they're either equal or differ by only 1.

Part A (3 Points): Compute $P(C)$ by hand.

Part A Answer here:

Solution: Let S be the probability that you roll the same value on each die and let D be the probability that the dice values only differ by one. $P(C) = P(S \cup D)$ Let number values $1, 2, 3, ...$ etc. represent the probability that that number is rolled on one die. $P(S) = P(1 \cap 1) + P(2 \cap 2) + P(3 \cap 3) + P(4 \cap 4) + P(5 \cap 5) + P(6 \cap 6)$ $P(D) = P(1 \cap 2) + P(2 \cap 3) + P(3 \cap 4) + P(4 \cap 5) + P(5 \cap 6)$ $P(S) = 6 \cdot \frac{1}{36}$ as there are six ways to roll the same number on both dice and the probaibility of doing this is $\frac{1}{36}$ so $P(S) = \frac{1}{6}$ $P(D) = 5 \cdot \frac{2}{36}$ as there are five scenarios where dice differ by a value of one and there is a $\frac{2}{36}$ probability of doing this as, for example, you could roll a 1 on the first die and a 2 on the second, or you could roll a 2 on the first die and a 1 on the second so $P(D) = \frac{10}{36}$. Since S and D are disjoint events $P(C) = P(S \cup D) = P(S) + P(D) = \frac{6}{36} + \frac{10}{36} = \frac{16}{36} = \frac{4}{9} = .444444$ or 44.444%.

Part B (10 Points): Write a simulation to run many trials (at least 10,000) of the pair of rolls and estimate the value of $P(C)$ you calculated in **Part A**. Does your estimate agree with the exact calculation you did in **Part A**? If not, try increasing the number of trials in your simulation.

[7]:
```
#Code answer to Part B here
die1 = {'values' : np.array([1, 2, 3, 4, 5, 6])}
die2 = {'values' : np.array([1, 2, 3, 4, 5, 6])}

def roll_simulation(num_samples):
    sum = 0
    for ii in range(num_samples):
        firstDie = np.random.choice(die1['values'])
        secondDie = np.random.choice(die2['values'])

        if ((firstDie - secondDie) >= -1) and ((firstDie - secondDie) <= 1):
            sum+=1
    return sum / num_samples

print(roll_simulation(30000))
```

```
print("My estimate agrees with the exact calculation I did in Part A.")
```

```
0.4451333333333333
My estimate agrees with the exact calculation I did in Part A.
```

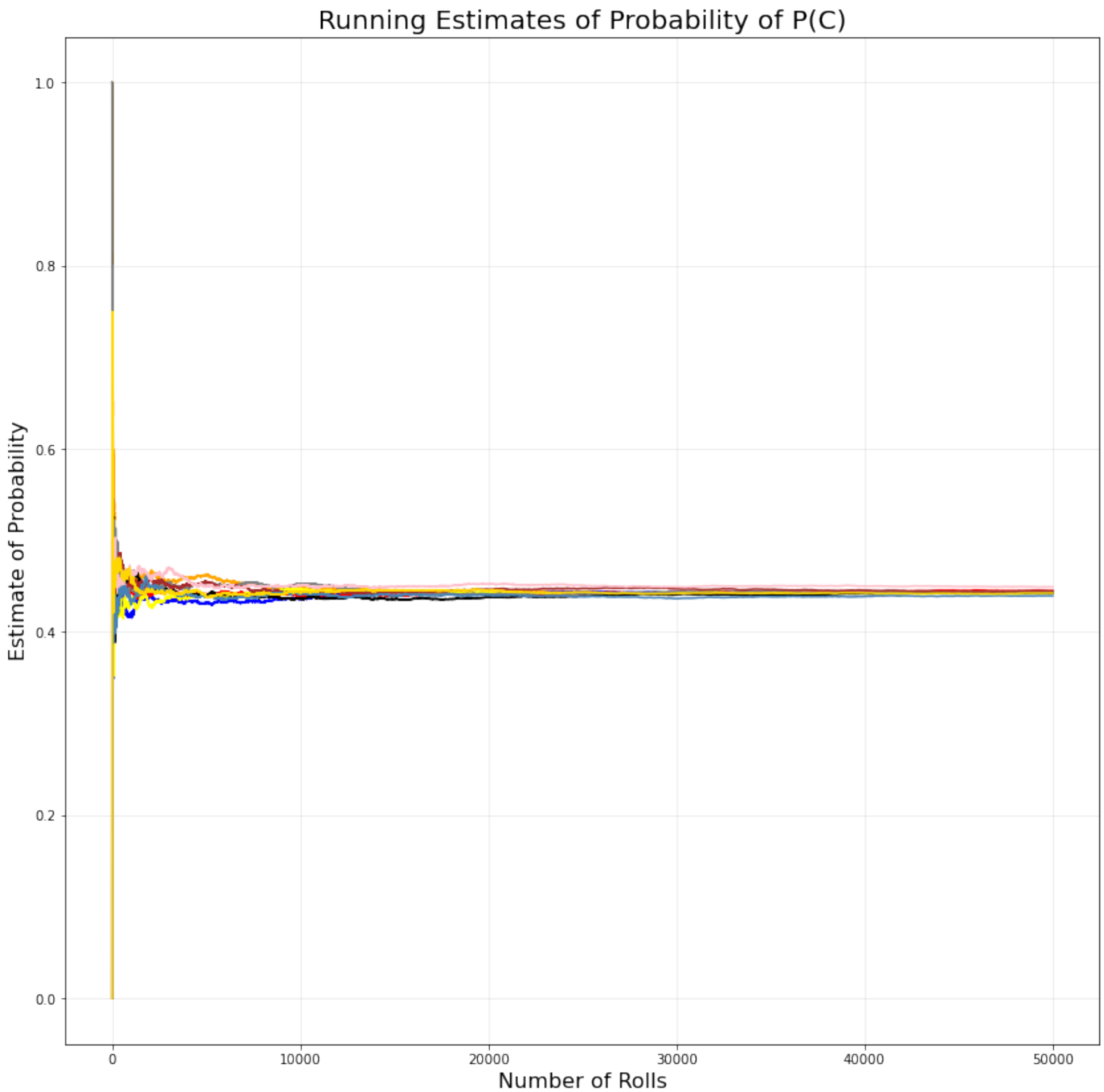
Part C (10 Points): Write code to run at least 10 independent simulations of at least 50,000 trials each to estimate $P(C)$ and plot their running estimate curves on the same set of axes. **Hint**: This is a lot of computation, so try to leverage Numpy as much as possible so that your code doesn't run forever.

[8]:
```
#Code answer to Part C here
die = np.array([1, 2, 3, 4, 5, 6])
num_rolls=50000

rolls1 = np.random.choice(die, size=num_rolls)
rolls2 = np.random.choice(die, size=num_rolls)

p = np.zeros(num_rolls)

for ii in range(num_rolls):
    p[ii] = np.sum(np.logical_and(((rolls1[:ii+1] - rolls2[:ii+1]) >= -1),␣
 ↪((rolls1[:ii+1] - rolls2[:ii+1]) <= 1))) / len(rolls1[:ii+1])

rolls1 = np.random.choice(die, size=num_rolls)
rolls2 = np.random.choice(die, size=num_rolls)

q = np.zeros(num_rolls)

for ii in range(num_rolls):
    q[ii] = np.sum(np.logical_and(((rolls1[:ii+1] - rolls2[:ii+1]) >= -1),␣
 ↪((rolls1[:ii+1] - rolls2[:ii+1]) <= 1))) / len(rolls1[:ii+1])

rolls1 = np.random.choice(die, size=num_rolls)
rolls2 = np.random.choice(die, size=num_rolls)

j = np.zeros(num_rolls)

for ii in range(num_rolls):
    j[ii] = np.sum(np.logical_and(((rolls1[:ii+1] - rolls2[:ii+1]) >= -1),␣
 ↪((rolls1[:ii+1] - rolls2[:ii+1]) <= 1))) / len(rolls1[:ii+1])

rolls1 = np.random.choice(die, size=num_rolls)
rolls2 = np.random.choice(die, size=num_rolls)

z = np.zeros(num_rolls)

for ii in range(num_rolls):
```

```
    z[ii] = np.sum(np.logical_and(((rolls1[:ii+1] - rolls2[:ii+1]) >= -1), ((rolls1[:ii+1] - rolls2[:ii+1]) <= 1))) / len(rolls1[:ii+1])

rolls1 = np.random.choice(die, size=num_rolls)
rolls2 = np.random.choice(die, size=num_rolls)

y = np.zeros(num_rolls)

for ii in range(num_rolls):
    y[ii] = np.sum(np.logical_and(((rolls1[:ii+1] - rolls2[:ii+1]) >= -1), ((rolls1[:ii+1] - rolls2[:ii+1]) <= 1))) / len(rolls1[:ii+1])

rolls1 = np.random.choice(die, size=num_rolls)
rolls2 = np.random.choice(die, size=num_rolls)

x = np.zeros(num_rolls)

for ii in range(num_rolls):
    x[ii] = np.sum(np.logical_and(((rolls1[:ii+1] - rolls2[:ii+1]) >= -1), ((rolls1[:ii+1] - rolls2[:ii+1]) <= 1))) / len(rolls1[:ii+1])

rolls1 = np.random.choice(die, size=num_rolls)
rolls2 = np.random.choice(die, size=num_rolls)

w = np.zeros(num_rolls)

for ii in range(num_rolls):
    w[ii] = np.sum(np.logical_and(((rolls1[:ii+1] - rolls2[:ii+1]) >= -1), ((rolls1[:ii+1] - rolls2[:ii+1]) <= 1))) / len(rolls1[:ii+1])

rolls1 = np.random.choice(die, size=num_rolls)
rolls2 = np.random.choice(die, size=num_rolls)

t = np.zeros(num_rolls)

for ii in range(num_rolls):
    t[ii] = np.sum(np.logical_and(((rolls1[:ii+1] - rolls2[:ii+1]) >= -1), ((rolls1[:ii+1] - rolls2[:ii+1]) <= 1))) / len(rolls1[:ii+1])

rolls1 = np.random.choice(die, size=num_rolls)
rolls2 = np.random.choice(die, size=num_rolls)

k = np.zeros(num_rolls)

for ii in range(num_rolls):
    k[ii] = np.sum(np.logical_and(((rolls1[:ii+1] - rolls2[:ii+1]) >= -1), ((rolls1[:ii+1] - rolls2[:ii+1]) <= 1))) / len(rolls1[:ii+1])
```

```
rolls1 = np.random.choice(die, size=num_rolls)
rolls2 = np.random.choice(die, size=num_rolls)

f = np.zeros(num_rolls)

for ii in range(num_rolls):
    f[ii] = np.sum(np.logical_and(((rolls1[:ii+1] - rolls2[:ii+1]) >= -1),␣
 ↪((rolls1[:ii+1] - rolls2[:ii+1]) <= 1))) / len(rolls1[:ii+1])



fig, ax = plt.subplots(figsize=(14,14))

ax.plot(p, color='blue')
ax.plot(q, color='orange')
ax.plot(j, color='red')
ax.plot(z, color='yellow')
ax.plot(y, color='black')
ax.plot(x, color='grey')
ax.plot(w, color='brown')
ax.plot(t, color='pink')
ax.plot(k, color='steelblue')
ax.plot(f, color='gold')

ax.set_title("Running Estimates of Probability of P(C)", fontsize=20)
ax.set_xlabel("Number of Rolls", fontsize=16)
ax.set_ylabel("Estimate of Probability", fontsize=16)

ax.grid(True, alpha=0.25)

print("Estimated probability after {} rolls is approximately {:.4f}".
 ↪format(num_rolls, p[-1]))
```

```
Estimated probability after 50000 rolls is approximately 0.4438
```



Part D (2 Points): Comment on the behavior of the running estimates as the number of trials increases.

Part D Answer here:

Solution: As the number of trials increases the running estimates converge to the value representing the actual probability of what we we are simulating, .4444 repeating.

[ ]: