Project 3: Some Interesting DRV's

EE 511 – Section: Tuesday 5 pm

Name: Darshan Patil

Student ID: 9575227834

1.

i. Problem Statement

a. Sum of Uniform RV's

Define: N=$Min\{n: \sum_{i=1}^{n} U_{i>1}\}$ where $\{U_i\}$ are iid Uniform (0,1) RV's.

Find (by simulation): $\widehat{m} = E[N]$ an estimator for the mean.

Can you guess (or derive) the true value for E[N]?

ii. Theoretical Exploration

Let $S_n = \sum_{i=1}^{n} U_i$ and $f_n(s)$ is the probability density function for S_n.

For $0 < x < 1 = f_1(x) = 1$ and $f_{n+1}(x) = \int_0^x f_n(s).ds$

Therefore, $f_n(x) = \frac{x^{n-1}}{(n-1)!}$ $\quad 0 < x < 1.$

$$\Rightarrow P[S_n < 1] = \int_0^1 f_n(s)ds = 1/n!, \ \left\{ since, \int_0^x f_n(s)ds = f_n + 1\,(x) = \frac{x^n}{n!} \right\}$$

Now,

$$P[N = n] = P[S_{n-1} < 1 \le S_n] = P[S_{n-1} < 1] - P[S_n < 1]$$

$$= \left[\frac{1}{(n-1)!}\right] - \left[\frac{1}{n!}\right]$$

$$= \left[\frac{n-1}{n!}\right]$$

For Expected Value of N, note that,

$$P[N \ge 2] = 1 - P[N < 2] = 1$$

$$P[N \ge 3] = 1 - P[N < 3\} = P[U_1 + U_2 \le 1] = \frac{1}{2}$$

$$P[N \ge 4] = 1 - P[N < 4] = P[U_1 + U_2 + U_3 \le 1] = \frac{1}{3!}$$

Now, $P[N \geq j] = \frac{1}{(j-1)!}$ can be written as,

$$P[N \geq 1] = P[N = 1] + P[N = 2] + P[N = 3] + \ldots$$

$$P[N \geq 2] = P[N = 2] + P[N = 3] + P[N = 4] + \ldots$$

.
.
.

$$Sum = P[N = 1] + P[N = 2] + P[N = 3] + \cdots$$

$$E[N] = \sum_{j=1}^{\infty} jP[N = j] = \sum_{j=1}^{\infty} P[N \geq j] = 1 + 1 + \left(\frac{1}{2}\right) + \cdots$$

$$\Longrightarrow E[N] = e$$

Thus the true of E[N] must be equal to e.

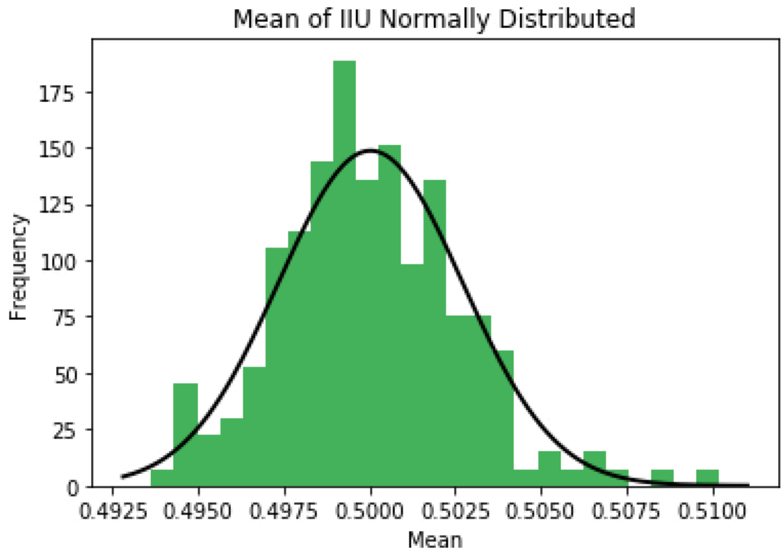
iii. Simulation Methodology

Simulation steps:

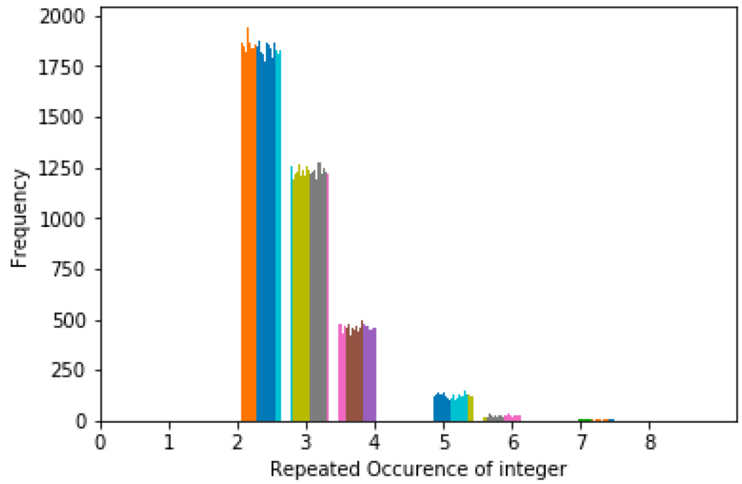
- Generate N random variable for each Trial.
- Calculated N=$Min\{n: \sum_{i=1}^{n} U_{i>1}\}$ using the function sumurv() which return array of n values for each Trial, store them in nArray[].
- Calculate mean of the nArray[] for each trial, and also Total mean.
- Calculate the standard deviation.
- Plot the Array on the histogram.

iv. Results

IID Uniform RV's are uniformly distributed. Hence, Mean of IID Uniform RV's are normally distributed.



For N=10000,



Mean and Standard Deviation for different values of N

N	Mean	Standard Deviation
10000	2.717992420137646	0.013330724647887255
12000	2.7197512193863997	0.01301335454119137
15000	2.719181849520082	0.012617835022617834
20000	2.718702735614303	0.009589071940246696

Discussion of results:
We can observe that the value of Mean i.e. $E[N] \cong e$ for N=$Min\{n: \sum_{i=1}^{n} U_{i>1}\}$.

v. References

- Simulation, 5th edition, by Sheldon Ross, Academia Press.
- Introduction to probability models, 10th edition, by Sheldon Ross.
- https://pypi.org/project/numpy/

vi. Source Code

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Wed Oct 10 00:38:02 2018
@author: darshan
"""

import random as  rnd       #import random library to generate random variable
import numpy as np          #import numpy for fundamental computation
from matplotlib import pyplot as plt   #import to plot histogram
```

```
from scipy.stats import norm             #import to show normal distribution

def rand(Trial,N):                                  #function to generate random variable
   randList=np.zeros((Trial,N))                    #initial matrix Trial,N with zero
   marray=[]
   for i in range(Trial):
      for j in range(N):
         randList[i][j]= rnd.random()              #generate N random variable
      marray.append(np.mean(randList[i]))
   return randList,marray


def sumurv(rList,N):                                #Calculate the condition Min{n: sigma U_i>1}
   s = 0
   n = 0
   nA=[]
   for i in range(N):                               #Loop in each Trial
      n = n + 1
      s = s + rList[i]
      if s > 1:
         nA.append(n)                               # Store the min values in nA
         n=0
         s=0
         continue                                   # append the min{n} and continue till end of Trial
   return nA



def normf(mr, Trial):                               # plot normal distribution graph
   mu, std = norm.fit(mr)                           # Plot the histogram.
   plt.hist(mr, bins=25, density=True, alpha=0.6, color='g')
   # Plot the PDF.
   xmin, xmax = plt.xlim()
   x = np.linspace(xmin, xmax, Trial)
   p = norm.pdf(x, mu, std)                         #plot normal distribution of mean
   plt.plot(x, p, 'k', linewidth=2)
   plt.xlabel('Mean')
   plt.ylabel('Frequency')
   #title = "Fit results: mu = %.2f,  std = %.2f" % (mu, std)
   plt.title("Mean of IIU's Normally Distributed")
   plt.show()
```

```
def main():
    Trial=200                                       #Number of Trial
    N=10000                                         #Number of IIU in each Trial
    marray=[]
    rList, marray=rand(Trial,N)                     #generate N random number for Trials
    #print(rList)
    nArray=[[] for Null in range(Trial)]            #initialize
    meanArray=[]
    for j in range(Trial):
        nArray[j]=sumurv(rList[j],N)                #call sumurv to calculate condition
        meanArray.append(np.mean(nArray[j]))    #calculate mean of nArray
    #print("nArray",nArray,"\nMeanArray:",meanArray)
    TMean = np.mean(meanArray)                      #Calculate total mean of nArray
    TStdDiv = np.std(meanArray)
    print("Mean:",TMean,"\nStandard Deviation:", TStdDiv)
    plt.hist(nArray)                                 #Plot histogram of nArray
    plt.xticks(range(0,9))
    plt.xlabel('Repeated Occurence of integer')
    plt.ylabel('Frequency')
    plt.show()                                      #show histogram
    normf(marray,Trial)                             #show that mean of IIU is normally distributed
main()
```

2.

i. Problem Statement

Minima of Uniform RV's

Define: $N = Min\{n: U_1 \leq U_2 \leq \cdots \leq U_{n-1} > U_n\}$

i.e, the n^{th} term is the first that is less than its predecessor, where $\{U_i\}$ are independent identically distributed (iid) Uniform (0,1) RV's.

Find (by simulation): $\widehat{m} = E[N]$ an estimator for the mean.

Can you guess (or derive) the true value for E[N]?

ii. Theoretical Exploration

$$E[X2] = \sum_{k=1}^{\infty} kP(X2 = k)$$

$$E[X2] = \int_0^1 \sum_{k=1}^{\infty} kP(X2 = k \mid U1 = u) f_u(u) du$$

$$E[X2] = \sum_{k=1}^{\infty} \int_0^1 [P(U < u)]^{k-1} P(U > u) du$$

$$E[X2] = \sum_{k=1}^{\infty} k \int_0^1 (u)^{k-1} (1-u) du$$

$$E[X2] = \sum_{k=1}^{\infty} k \left[\frac{u^k}{k} - \frac{u^{k+1}}{k+1} \right]_1^0$$

$$E[X2] = \sum_{k=1}^{\infty} \frac{1}{k+1}$$

iii. Simulation Methodology

Simulation steps:

- The function minima () generates the IIU and compare the previous RV with the next RV.
- Find the RV which is greater than all the previous RV's.
- Repeat the process for N RV's.
- These values are used to calculate mean and standard deviation.

iv. Results

N	Mean	Standard Deviation
10000	2.72	0.8789766777338293
12000	2.7143	0.8747730499328891
15000	2.71926	0.8801830087481177
20000	2.70935	0.8613202525774023

Discussion of Results

We can observe that the value of mean is approximately equal to e.

v. References

- Simulation, 5th edition, by Sheldon Ross, Academia Press.
- Introduction to probability models, 10th edition, by Sheldon Ross.
- https://pypi.org/project/numpy/

vi. Source Code

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Tue Oct  9 23:33:57 2018
@author: darshan
"""

import random
import numpy as np

#Calculate Minima in N RV's
def minima():
    n = 0
    t = 0
    while(True):
     t1 = 0
     t1 = random.random()      #Generate random number
     if t < t1:
        n = n + 1
        t = t1
     else:
        return n + 1

def main():
    aList = []
    for i in range(20000):
       aList.append(minima())
    mean = np.mean(aList)                  #Calculate Mean
    stdDiv = np.std(aList)                 #Calculate Standard Deviation
    print("Mean: ",mean, "\nSD: ",stdDiv);          #Display mean and SD
main()
```

3.

i. Problem Statement

Maxima of Uniform RV's

Consider the sequence of iid Uniform RV's$\{U_i\}$. If $U_j > \max_{i=1:j-1}\{U_i\}$ we say U_j is a record.

Example: the records are underlined.

$\{U_i\} = \{$<u>0.2314</u>, <u>0.4719</u>, 0.1133, <u>0.5676</u>, 0.4388, <u>0.9453</u>…}

(note that the U_i are on the real line and we are just showing 4 digits of precision).

Let X_i be an RV for the distance from the i-1st record to the i^{th} record. Clearly $X_1 = 1$ always. In this example, $X_2 = 1, X_3 = 2, X_4 = 2$.

Distribution of Records: Using simulation, obtain (and graph) a probability histogram for X_2 and X_3 and compute the sample means.

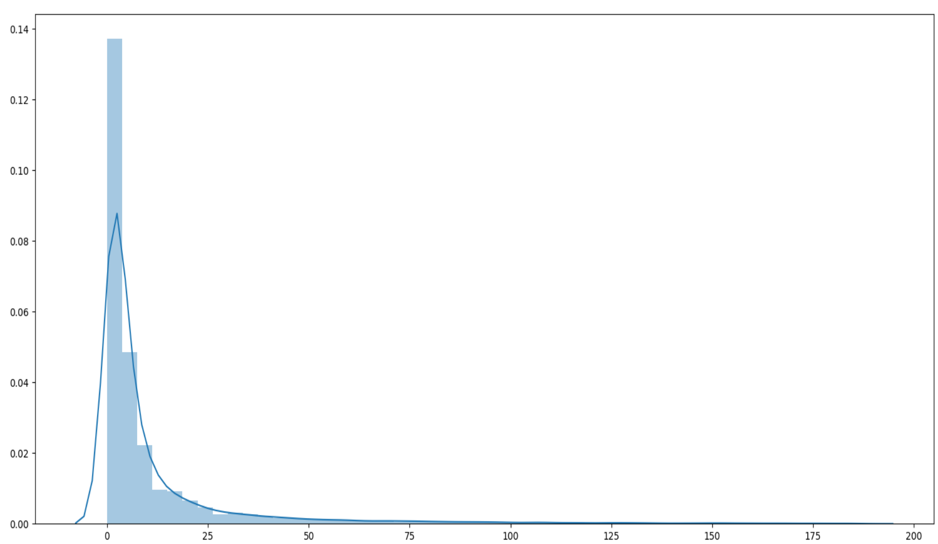
Can you find an analytical expression for $P(X_2 = k)$? (Hint: conditioned on U_1 and then unconditioned) What does this say about $E[X_2]$?

ii. Simulation Methodology

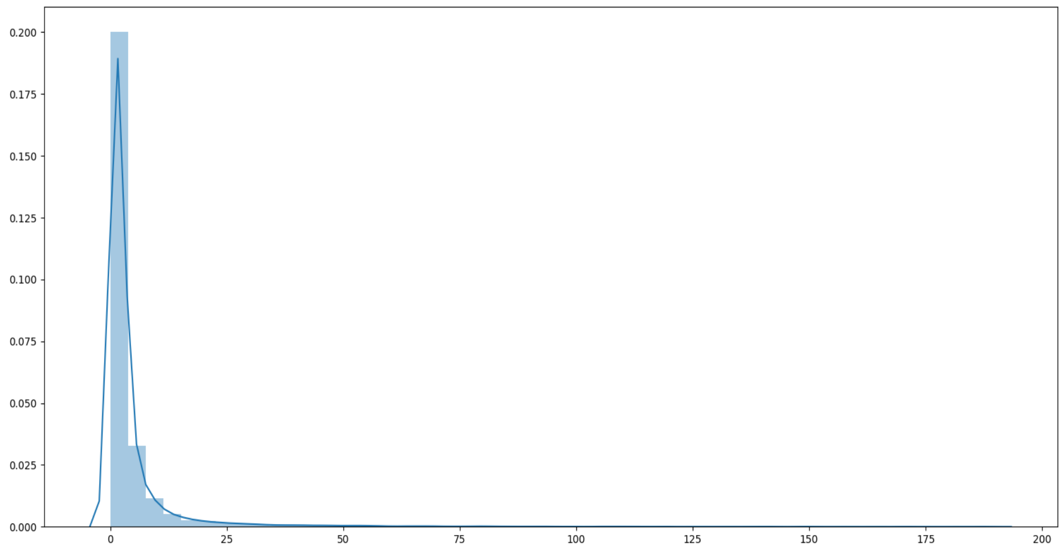
- Initialize 2 Lists to store the values of X_2 and X_3
- Generate n random numbers.
- In a loop, the 1^{st} RV in List is compared with the next RV and if its greater -- it goes to the next loop to compare with the other values.
- Multiple outputs are obtained using values from 100 to 400 with increment of 40 and its graph is plotted with and we will be able to identify that the values of $E[X_2]$ and $E[X_3]$ increases as N increases.

iii. Results

Probability histogram for X_2



Probability histogram for X_3



For N = 5000,

i. In graphs 1 and 2, as the number of counts increases the value of $E[X_2]$ and $E[X_3]$ increases. The values are un-bounded and keep on increasing with N.
ii. The histogram bars – list of X_2 values v/s bars of 25 values.
Sample Means –
X_2 - [4.1234, 4.437, 4.5214, 5.022, 5.0161, 5.1621, 5.336, 5.5108]
X_3 - [8.7196, 9.9101, 10.7591, 12.3241, 12.7261, 13.9736, 15.321, 15.251]

iv. References

- https://www.tutorialspoint.com/matlab/
- https://www.mathworks.com/help/matlab/ref/integral.html#btdd9x5
- Lecture 5 - Simple Monte-Carlo Methods (2up).pdf by professor Silvester

V. Source Code

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Sun Oct 14 17:57:22 2018
@author: darshan
"""

import numpy as np
import matplotlib.pyplot as plt
import seaborn as s
u=np.zeros(5000)                   # initialize 5000 sample trials for calculate X2
v=np.zeros(5000)                   # initialize for 5000 sample trials for calculate X3
for j in range(0,5000):            # trials 5000 times
  r=np.random.random_sample((50,1))  # generate 50 random variables
  for i in range(1,49):
    if(r[i]>r[0]):                 # compare if the ith value is greater than r[0]
      u[j]=i
      for k in range(2,49):        # repeat 49 times
        if(r[k]>r[i]):
           v[j]=k-i
           break
      break
xAxis = range(50)
x2=np.mean(u)                      # computing the value of x2
x3=np.mean(v)                      # computing the value of x3
print('The mean of X2: ',x2);
print('The mean of X3: ',x3);
s.distplot(u)                      # plot for X2
plt.show()
plt.figure()
s.distplot(v)                      # plot for X3
plt.show()
```