IEOR 4404 Simulation **Fall 2024**

Homework 5

Instructor: Henry Lam

Problem 1 For $Unif(0,1)$ random variables $U_1, U_2, \dots$ define

$$N_1 = \text{Minimum}\left\{n : \sum_{i=1}^{n} U_i > 1\right\}.$$

That is, N_1 is equal to the number of random numbers that must be summed to exceed 1. Write a pseudo-code for generating N_1. Give a point estimate and a 95% confidence interval for $\mathbb{E}[N_1]$, by generating 100, 1000 and 10,000 copies of N_1 respectively on a computer.

Do the same for N_2, the number of random numbers that must be summed to exceed 2. That is,

$$N_2 = \text{Minimum}\left\{n : \sum_{i=1}^{n} U_i > 2\right\},$$

From your results, do you think $E[N_2]$ increases from $E[N_1]$? Explain briefly.

Problem 2 Write a pseudo-code to generate a simulation run that is appropriate for numerically approximating the integral

$$\int_0^\infty \int_0^{x^2} e^{-(x+y)} \sin(xy)dydx.$$

Implement 1000 simulation runs on a computer. Give a point estimate and a 95% confidence interval for the value of this integral.

Problem 3 Suppose we estimate $\alpha = \mathbb{E}[X]$ via

$$\hat{\alpha}_n = \frac{1}{n}\sum_{j=1}^{n} X_j,$$

where the X_j's are i.i.d. simulation replications of X. Assume that $Var(X) = 2$. Find the smallest value n such that $\hat{\alpha}_n$ is within 0.05 units of α with 95% confidence. Briefly justify your answer.

Problem 4 To estimate θ, we generate 20 i.i.d. replications each having mean θ. Suppose the obtained values are 102, 112, 131, 107, 114, 95, 133, 145, 139, 117, 93, 111, 124, 122, 136, 141, 119, 122, 151, 143.

a) Find a 95% confidence interval for θ.

b) How many additional replications do you think we will have to generate if we want to be 99% certain that our final estimate of θ is correct to within ± 0.5?

Problem 5 Let $X_1, ..., X_{40}$ be i.i.d. copies of random variable X generated from a computer routine. Let

$$p = \mathbb{P}((X-1)^2 < 30)$$

Give a point estimate and a 95% confidence interval of p if, among all the values of the X_i, 2 appears 11 times, 4 appears 5 times, 5 appears 10 times, 6 appears 2 times, 8 appears 7 times, 9 appears 4 times and 24 appears 1 time.

Problem 6 Suppose that the following are i.i.d. copies of random variable X generated from a computer routine: 5, 4, 9, 6, 21, 12, 7, 14, 17, 11, 20, 7, 10, 21, 15, 26, 9, 13, 8, 6. We want to estimate $\mathbb{E}[X^2]$.

(a) How many additional replications will we need to generate if we want to be 99% certain that our estimate has a margin of error 0.1?

(b) What assumptions have you made in part (a)?

Alexandre
Duhamel
afd2153
MSOR

HW 5: Simulation

Problem 1:

to generate N_2

- Pseudo-code to generate N_1: 1. Initialize $\begin{cases} N_1 = 0 \quad (N_2 = 0) \\ sum = 0 \end{cases}$

  2. While sum <= 1 (<= 2): $\begin{cases} \text{Generate } U \sim Unif(0,1) \\ sum \mathrel{+}= U \\ N_1 \mathrel{+}= 1 \end{cases}$

  3. Return N_1

- Code for point estimate & confidence interval for $E[N_1]$ and $E[N_2]$:

```
def generate_N(threshold, num_samples):
    N_values = []

    for _ in range(num_samples):
        total_sum = 0
        count = 0

        while total_sum <= threshold:
            total_sum += np.random.uniform(0, 1)
            count += 1

        N_values.append(count)

    return np.array(N_values)

num_samples = [100, 1000, 10000]
N1_2 = [1, 2]

for threshold in N1_2:
    print(f"N°{threshold} : ")
    for samples in num_samples:
        N_values = generate_N(threshold, samples)
        mean_N = np.mean(N_values)
        std_N = np.std(N_values, ddof=1)
        conf_interval = 1.96 * std_N / np.sqrt(samples)
                        └ z for 5% CI
        print(f"{samples} samples:\n\tPoint estimate (me
    print()
```

```
N°1 :
100 samples:
        Point estimate (mean): 2.790
        95% Confidence Interval: [2.618, 2.962]

1000 samples:
        Point estimate (mean): 2.726
        95% Confidence Interval: [2.673, 2.779]

10000 samples:
        Point estimate (mean): 2.720
        95% Confidence Interval: [2.703, 2.738]

N°2 :
100 samples:
        Point estimate (mean): 4.530
        95% Confidence Interval: [4.310, 4.750]

1000 samples:
        Point estimate (mean): 4.659
        95% Confidence Interval: [4.583, 4.735]

10000 samples:
        Point estimate (mean): 4.667
        95% Confidence Interval: [4.642, 4.691]
```

- $E[N_2]$ increases from $E[N_1]$. This is due to the fact that N_2 requires the sum to exceed the threshold of 2 instead of 1, so it takes more loop and creates a higher value.

Problem 2:

- Pseudo-code using Monte-Carlo:

1. Initialize $\begin{cases} N = \text{samples wanted} \\ x_{max} = \text{upper value of } x \text{ for } +\infty \\ \text{sum} = 0 \end{cases}$

2. For $i = 1$ to N: Generate $X \sim \text{Uniform}(0, x_{max})$
   Generate $Y \sim \text{Uniform}(0, X^2)$
   Compute: $f(x,y) = e^{-x-y} \sin(xy)$
   sum += $f(x,y)$

3. Return $\dfrac{x_{max} \times \text{sum}}{N}$

- The idea is that $\displaystyle\int_0^{+\infty}\int_0^{x^2} e^{-(x+y)} \sin(xy)\,dy\,dx \approx \int_0^{x_{max}}\int_0^{x^2} e^{-(x+y)} \sin(xy)\,dy\,dx \approx \frac{x_{max}}{N} \sum_{i=1}^{N} e^{-(x_i+y_i)} \sin(x_i y_i)$

volume of integral since y depends of x_{max}

- Here is the code:

```
def monte_carlo_integral(num_samples, x_max):
    results = []

    for _ in range(num_samples):
        x = np.random.uniform(0, x_max)
        y = np.random.uniform(0, x**2)
        f_xy = np.exp(-(x + y)) * np.sin(x * y)
        results.append(f_xy)

    estimate = (x_max / num_samples) * np.sum(results)

    return estimate, np.array(results)

num_samples = 1000
x_max = 80

point_estimate, results = monte_carlo_integral(num_samples, x_max)

std_dev = np.std(results, ddof=1)

z_score = 1.96
margin_of_error = z_score * std_dev / np.sqrt(num_samples)
conf_interval = (point_estimate - margin_of_error, point_estimate + margin_of_error)

print(f"Point Estimate: {point_estimate:.3f}")
print(f"95% Confidence Interval: {conf_interval}")
```

I tried multiple values until I found one where the output stabilizes (to less than 1%)

```
Point Estimate: 0.096
95% Confidence Interval: (0.09537085834560255, 0.09666106448183479)
```

Problem 3:

We have $\hat{\alpha}_m = \frac{1}{m}\sum_{j=1}^{m} X_j$ where X_i are iid and $Var(X) = 2$

We're searching for m that offers a 95% and 0.05 precision of α ie:

$P\left(-z\frac{\sigma}{\sqrt{m}} + \alpha < \hat{\alpha}_m < z\frac{\sigma}{\sqrt{m}} + \alpha\right) = 1-\alpha$ where $\alpha = 5\%$ so $z = 1.96$ and $\sigma = \sqrt{\frac{1}{m-1}\sum(x_i - \hat{\mu})^2}$

We want a margin of error to be under 0.05 ie $\underbrace{z}_{1.96 \text{ gives } 95\%} \cdot \underbrace{\frac{\sigma}{\sqrt{m}}}_{CI} \leq \overbrace{0.05}^{\text{precision}} \Rightarrow m \geq \left(\frac{1.96 \cdot \sqrt{2}}{0.05}\right)^2 = 3073.28$

So $m = 3074$

Problem 4:

a) We can define the sample mean of θ (realisation of X) as $\hat{\theta} = \frac{1}{m}\sum_{i=1}^{m} X_i$

Since the sample size is relatively small and we don't know its stdev, we'll use the t-distribution:

$t_{\frac{5\%}{2}, 19} \approx 2.093$

we only of the sample stdev

t Table

							$t_{a)}$		$t_{b)}$		
cum. prob	$t_{.50}$	$t_{.75}$	$t_{.80}$	$t_{.85}$	$t_{.90}$	$t_{.95}$	$t_{.975}$	$t_{.99}$	$t_{.995}$	$t_{.999}$	$t_{.9995}$
one-tail	0.50	0.25	0.20	0.15	0.10	0.05	0.025	0.01	0.005	0.001	0.0005
two-tails	1.00	0.50	0.40	0.30	0.20	0.10	0.05	0.02	0.01	0.002	0.001
df											
1	0.000	1.000	1.376	1.963	3.078	6.314	12.71	31.82	63.66	318.31	636.62
2	0.000	0.816	1.061	1.386	1.886	2.920	4.303	6.965	9.925	22.327	31.599
3	0.000	0.765	0.978	1.250	1.638	2.353	3.182	4.541	5.841	10.215	12.924
4	0.000	0.741	0.941	1.190	1.533	2.132	2.776	3.747	4.604	7.173	8.610
5	0.000	0.727	0.920	1.156	1.476	2.015	2.571	3.365	4.032	5.893	6.869
6	0.000	0.718	0.906	1.134	1.440	1.943	2.447	3.143	3.707	5.208	5.959
7	0.000	0.711	0.896	1.119	1.415	1.895	2.365	2.998	3.499	4.785	5.408
8	0.000	0.706	0.889	1.108	1.397	1.860	2.306	2.896	3.355	4.501	5.041
9	0.000	0.703	0.883	1.100	1.383	1.833	2.262	2.821	3.250	4.297	4.781
10	0.000	0.700	0.879	1.093	1.372	1.812	2.228	2.764	3.169	4.144	4.587
11	0.000	0.697	0.876	1.088	1.363	1.796	2.201	2.718	3.106	4.025	4.437
12	0.000	0.695	0.873	1.083	1.356	1.782	2.179	2.681	3.055	3.930	4.318
13	0.000	0.694	0.870	1.079	1.350	1.771	2.160	2.650	3.012	3.852	4.221
14	0.000	0.692	0.868	1.076	1.345	1.761	2.145	2.624	2.977	3.787	4.140
15	0.000	0.691	0.866	1.074	1.341	1.753	2.131	2.602	2.947	3.733	4.073
16	0.000	0.690	0.865	1.071	1.337	1.746	2.120	2.583	2.921	3.686	4.015
17	0.000	0.689	0.863	1.069	1.333	1.740	2.110	2.567	2.898	3.646	3.965
18	0.000	0.688	0.862	1.067	1.330	1.734	2.101	2.552	2.878	3.610	3.922
19	0.000	0.688	0.861	1.066	1.328	1.729	2.093	2.539	2.861	3.579	3.883
20	0.000	0.687	0.860	1.064	1.325	1.725	2.086	2.528	2.845	3.552	3.850
21	0.000	0.686	0.859	1.063	1.323	1.721	2.080	2.518	2.831	3.527	3.819
22	0.000	0.686	0.858	1.061	1.321	1.717	2.074	2.508	2.819	3.505	3.792
23	0.000	0.685	0.858	1.060	1.319	1.714	2.069	2.500	2.807	3.485	3.768
24	0.000	0.685	0.857	1.059	1.318	1.711	2.064	2.492	2.797	3.467	3.745
25	0.000	0.684	0.856	1.058	1.316	1.708	2.060	2.485	2.787	3.450	3.725
26	0.000	0.684	0.856	1.058	1.315	1.706	2.056	2.479	2.779	3.435	3.707
27	0.000	0.684	0.855	1.057	1.314	1.703	2.052	2.473	2.771	3.421	3.690
28	0.000	0.683	0.855	1.056	1.313	1.701	2.048	2.467	2.763	3.408	3.674
29	0.000	0.683	0.854	1.055	1.311	1.699	2.045	2.462	2.756	3.396	3.659
30	0.000	0.683	0.854	1.055	1.310	1.697	2.042	2.457	2.750	3.385	3.646
40	0.000	0.681	0.851	1.050	1.303	1.684	2.021	2.423	2.704	3.307	3.551
60	0.000	0.679	0.848	1.045	1.296	1.671	2.000	2.390	2.660	3.232	3.460
80	0.000	0.678	0.846	1.043	1.292	1.664	1.990	2.374	2.639	3.195	3.416
100	0.000	0.677	0.845	1.042	1.290	1.660	1.984	2.364	2.626	3.174	3.390
1000	0.000	0.675	0.842	1.037	1.282	1.646	1.962	2.330	2.581	3.098	3.300
z	0.000	0.674	0.842	1.036	1.282	1.645	1.960	2.326	2.576	3.090	3.291
	0%	50%	60%	70%	80%	90%	95%	98%	99%	99.8%	99.9%
	Confidence Level										

as announcement said, we can say that the sample size is large enough and we don't know the real stdev. So we compute:

$S = \sqrt{\frac{1}{m-1}\sum_{i=1}^{m}(X_i - \bar{X}_i)}$

And use it as σ in the CI formula + z table instead of t

Here is the code to compute the value of the 95% CI:

```
data = [102, 112, 131, 107, 114, 95, 133, 145, 139, 117, 93, 111, 124, 122, 136, 141, 119, 122, 151, 143]
n = len(data)

sample_mean = np.mean(data)
sample_std = np.std(data, ddof=1)

confidence_level_95 = 0.95
alpha_95 = 1 - confidence_level_95
t_95 = 2.093

margin_of_error_95 = t_95 * sample_std / np.sqrt(n)
ci_95 = (sample_mean - margin_of_error_95, sample_mean + margin_of_error_95)

print(f"95% Confidence Interval: {ci_95}")
```

```
95% Confidence Interval: (114.98745566576802, 130.71254433423195)
```

b)

```
margin_of_error_99 = 0.5
confidence_level_99 = 0.99
alpha_99 = 1 - confidence_level_99
t_99 = 2.861

required_n = (t_99 * sample_std / margin_of_error_99) ** 2
required_n = np.ceil(required_n)

additional_replications = required_n - n

print(f"Additional replications needed: {additional_replications}")
```

like previous exercise

```
Additional replications needed: 9221.0
```

Problem 5:

$$(X-1)^2 < 30 \leftrightarrow \sqrt{(X-1)^2} < \sqrt{30}$$

$$\leftrightarrow |X-1| < \sqrt{30}$$

$$\leftrightarrow 1-\sqrt{30} < X < 1+\sqrt{30}$$

$$\leftrightarrow -4.477 < X < 6.477$$

So X must be in the interval: $]-4.477, 6.477[$

The given values that fit in this interval are: $\{2, 4, 5, 6\}$ with respective occurences: $\{11, 5, 10, 2\}$

For a total values that satisfy the interval: $11+5+10+2 = 28$

We can estimate p as $\hat{p} = \dfrac{\text{\# values that satisfy}}{\text{Total \# of observations}} = \dfrac{28}{40} = 0.7$

variance of Bernoulli trial either it's in or not

The 95% CI is : CI = $\hat{p} \pm \underbrace{z \cdot \sqrt{\frac{\hat{p}(1-\hat{p})}{n}}}_{= 0.1420} = 0.7 \pm 0.1420 = \boxed{[0.55798, 0.84201]}$

Problem 6:

a) $X^2 = \{X_i^2 \text{ for } X_i \text{ in } X\} = \{25, 16, 81, \dots\}$

So point of estimate of $E[X^2]$: $\hat{\mu}_2 = \frac{1}{n}\sum_{i=1}^{n} X_i^2$

ie the margin of error is $t \cdot \frac{S}{\sqrt{n}} = MOE = 0.1$

like the previous problems:

$$n = \left(\frac{t \cdot s}{MOE}\right)^2$$

I used the t table for the same reason as problem 4

But we can compute S and use the z table if we assume there are enough data

Code to compute it:

```
data = np.array([5, 4, 9, 6, 21, 12, 7, 14, 17, 11, 20, 7, 10, 21, 15, 26, 9, 13, 8, 6])
x_squared = data**2
n = len(data)

mean_x_squared = np.mean(x_squared)
std_x_squared = np.std(x_squared, ddof=1)

margin_of_error = 0.1
confidence_level = 0.99
alpha = 1 - confidence_level

t_critical = 2.8609

required_n = (t_critical * std_x_squared / margin_of_error) ** 2
required_n = np.ceil(required_n)

additional_replications = required_n - n

print(f"Point estimate of E[X^2]: {mean_x_squared:.3f}")
print(f"Sample standard deviation of X^2: {std_x_squared:.3f}")
print(f"Number of additional replications needed: {additional_replications}")
```

```
Point estimate of E[X^2]: 181.950
Sample standard deviation of X^2: 180.031
Number of additional replications needed: 26527826.0
```

b) Assumptions made : $\begin{cases} X_i \text{ are iid} \\ X^2 \text{ follow a normal distribution} \longrightarrow \text{use of central limit theorem for CI} \\ Var(X^2) < +\infty \\ \text{constant variance, so that as we increase the number of replications, the variability of the sample stays the same} \end{cases}$