CFRM 505 Homework 4

Eunki Chung
eunkich@uw.edu

February 24, 2023

Problem 1

Consider the integral

$$\theta = \int_0^1 \cos(x)\,\mathrm{d}x$$

Part 1a

Estimate this integral using direct Monte Carlo simulation. First use a pilot study of size $N_0 \geq 100$ to estimate how many samples you would need to ensure that $P(|\hat{\theta}_{2N} - \theta| \leq 0.01) \geq 0.95$, then use a full-sized simulation with the appropriate number of samples. Report the required number of samples, your estimate of the integral, a confidence interval for your estimate and the exact error.

$$\begin{aligned}\theta &= \int_0^1 \cos(x)\,\mathrm{d}x \\ &= E[cos(U)] \quad \text{where } U \sim Unif(0,1) \\ &= E[X] \quad \text{where } X = cos(U)\end{aligned}$$

Define a $2N$-sample mean estimator $\hat{\theta}_{2N}$ as follows,

$$\hat{\theta}_{2N} = \frac{1}{2N}\sum_{i=1}^{2N} X_i$$

where X_i's is an iid sample with a finite mean θ and variance σ^2.

Notice that $\hat{\theta}_{2N}$ can be written as follows,

$$\begin{aligned}\hat{\theta}_{2N} &= \frac{1}{2N}\sum_{i=1}^{2N} X_i \\ &= \frac{1}{2N}\left(X_1 + X_2 + X_3 + X_4 + \cdots X_{2N-1} + X_{2N}\right) \\ &= \frac{1}{N}\left(\frac{X_1 + X_2}{2} + \frac{X_3 + X_4}{2} + \cdots \frac{X_{2N-1} + X_{2N}}{2}\right)\end{aligned}$$

Now let $W_i = \frac{X_{2i-1}+X_{2i}}{2}$ for $i = 1, 2, \ldots N$. Notice that the pairs $(X_1, X_2), (X_3, X_4), \ldots, (X_{2N-1}, X_{2N})$ are iid, and hence W_i's are iid. Now we have,

$$\begin{aligned}&= \frac{1}{N}\left(W_1 + W_2 + \cdots W_N\right) \\ &= \frac{1}{N}\sum_{i=1}^{N} W_i \\ &= \overline{W}\end{aligned}$$

The population mean and variance of W_i is given by,

$$
\begin{aligned}
E[W] &= E[W_i] \\
&= E\left[\frac{X_{2i-1} + X_{2i}}{2}\right] \\
&= \frac{1}{2} E\left[X_{2i-1} + X_{2i}\right] \\
&= \frac{1}{2} 2E\left[X\right] \\
&= \theta
\end{aligned}
$$

$$
\begin{aligned}
Var[W] &= Var\left[W_i\right] \\
&= Var\left[\frac{X_{2i-1} + X_{2i}}{2}\right] \\
&= \frac{1}{4} Var\left[X_{2i-1} + X_{2i}\right] \\
&= \frac{1}{4} 2Var[X] \\
&= \frac{\sigma^2}{2}
\end{aligned}
$$

W_i's is an iid sample with a finite population mean θ and variance $\sigma_W^2 = \frac{\sigma^2}{2}$, by CLT, we have

$$\frac{\overline{W} - \theta}{\sigma_W/\sqrt{N}} \xrightarrow{D} N(0,1)$$

Since $\overline{W} = \hat{\theta}_{2N}$ we have the following asymptotic distribution as $N \to \infty$.

$$\frac{\hat{\theta}_{2N} - \theta}{\sigma/\sqrt{2N}} \xrightarrow{D} N(0,1)$$

Now we want to bound the absolute error of the estimate to be less or equal to ϵ with a probability $1-\delta$ as follows.

$$P(|\hat{\theta}_{2N} - \theta| \leq \epsilon) \geq 1 - \delta$$

By multiplying a constant $\frac{\sqrt{2N}}{\sigma} > 0$,

$$
\begin{aligned}
P(|\hat{\theta}_{2N} - \theta| \leq \epsilon) &\geq 1 - \delta \\
P\left(\frac{\sqrt{2N}}{\sigma}|\hat{\theta}_{2N} - \theta| \leq \frac{\sqrt{2N}}{\sigma}\epsilon\right) &\geq 1 - \delta \\
P\left(\left|\frac{\sqrt{2N}}{\sigma}\left(\hat{\theta}_{2N} - \theta\right)\right| \leq \frac{\sqrt{2N}}{\sigma}\epsilon\right) &\geq 1 - \delta
\end{aligned}
$$

Since $\frac{\sqrt{2N}}{\sigma}\left(\hat{\theta}_{2N} - \theta\right) \xrightarrow{D} Z \sim N(0,1)$, for sufficiently large N,

$$
\begin{aligned}
P\left(|Z| \leq \frac{\sqrt{2N}}{\sigma}\epsilon\right) &\geq 1 - \delta \\
P\left(-\frac{\sqrt{2N}}{\sigma} \leq Z \leq \frac{\sqrt{2N}}{\sigma}\epsilon\right) &\geq 1 - \delta
\end{aligned}
$$

Since Z is symmetric,

$$
\begin{aligned}
2\left(P\left(Z \leq \frac{\sqrt{2N}}{\sigma}\epsilon\right) - \frac{1}{2}\right) &\geq 1 - \delta \\
2\left(\Phi\left(\frac{\sqrt{2N}}{\sigma}\epsilon\right) - \frac{1}{2}\right) &\geq 1 - \delta
\end{aligned}
$$

where Φ is cdf of standard normal Z

$$
\begin{aligned}
2\Phi\left(\frac{\sqrt{2N}}{\sigma}\epsilon\right) - 1 &\geq 1-\delta \\
2\Phi\left(\frac{\sqrt{2N}}{\sigma}\epsilon\right) &\geq 2-\delta \\
\Phi\left(\frac{\sqrt{2N}}{\sigma}\epsilon\right) &\geq 1-\frac{\delta}{2} \\
\frac{\sqrt{2N}}{\sigma}\epsilon &\geq \Phi^{-1}\left(1-\frac{\delta}{2}\right) \\
\sqrt{2N} &\geq \frac{\Phi^{-1}\left(1-\frac{\delta}{2}\right)\sigma}{\epsilon} \\
2N &\geq \left(\frac{\Phi^{-1}\left(1-\frac{\delta}{2}\right)\sigma}{\epsilon}\right)^2 \\
2N &\geq \left(\frac{z_\delta \sigma}{\epsilon}\right)^2
\end{aligned}
$$

Here, $\epsilon = 0.01, \delta = 0.05$ $z_\delta = \Phi^{-1}(1-\frac{0.05}{2}) = \Phi^{-1}(0.975) \approx 1.96$

$$
\begin{aligned}
2N &\geq \left(\frac{1.96\sigma}{\epsilon}\right)^2 \\
N &\geq \left(\frac{1.96\sigma}{\epsilon}\right)^2/2
\end{aligned}
$$

The confidence interval is given by,

$$
\hat{\theta}_{2N} \pm z_\delta \frac{\sigma}{\sqrt{2N}}
$$

```
import scipy.stats as stats
stats.norm.ppf(0.975)
```

```
1.959963984540054
```

Note that the true solution of the integral is given by,

$$
\begin{aligned}
\theta &= \int_0^1 \cos(x)\,\mathrm{d}x \\
&= \sin(x)\Big|_0^1 \\
&= \sin(1)
\end{aligned}
$$

```
import numpy as np
np.random.seed(12)

n0 = 100
zdelta = 1.96
epsilon = 0.01
true_solution = np.sin(1)

def f(x):
    return np.cos(x)
```

```python
# Direct method
# Pilot study
u = np.random.uniform(0, 1, size=n0)
integrand = f(u)
var_pilot = np.var(integrand)
N = int(np.ceil((zdelta ** 2 * var_pilot / epsilon ** 2) / 2))

# Full study
u = np.random.uniform(0, 1, size=2 * N)
X = f(u)
theta = np.mean(X)
var = np.var(X)
error = zdelta * np.sqrt(var / (2 * N))
lb = theta - error
ub = theta + error

print("Direct method ")
print("-"*50)
print("Required number of samples = {}".format(2 * N))
print("Estimated theta = {:.5f}".format(theta))
print("Confidence interval: [{:.5f}, {:.5f}]".format(lb, ub))
print(f"Half-length of CI: {error:.5f}")
print("Error = {:.5f}".format(theta - true_solution))
print("True theta = {:.5f}".format(true_solution))
```

```
Direct method
--------------------------------------------------
Required number of samples = 796
Estimated theta = 0.83551
Confidence interval: [0.82574, 0.84527]
Half-length of CI: 0.00977
Error = -0.00596
True theta = 0.84147
```

Part 1b

Repeat part (a), but use antithetic variates for both the pilot study and the full simulation.

Antithetic estimator $\hat{\theta}_{N,N}$ is defined as

$$\hat{\theta}_{N,N} = \frac{1}{2N}\sum_{i=1}^{N} Y_i + Z_i$$

where two random variables Y and Z that have the same mean and variance as X.

$\hat{\theta}_{N,N}$ can be written as follows,

$$\hat{\theta}_{N,N} = \frac{1}{N}\sum_{i=1}^{N} \frac{Y_i + Z_i}{2}$$

Similar to (a), notice that the pairs $(Y_1, Z_1), (Y_2, Z_2), \ldots, (Y_N, Z_N)$ are iid, hence $\left(\frac{Y_i+Z_i}{2}\right)$'s are iid.

The population mean and variance are given by,

$$
\begin{aligned}
E\left[\frac{Y_i+Z_i}{2}\right] &= \frac{1}{2}E\left[Y_i+Z_i\right] \\
&= \frac{1}{2}(E[Y_i]+E[Z_i]) \\
&= \frac{1}{2}2E[X] \\
&= \theta
\end{aligned}
$$

$$
\begin{aligned}
Var\left[\frac{Y_i+Z_i}{2}\right] &= \frac{1}{4}Var\left[Y_i+Z_i\right] \\
&= \frac{1}{4}\left(Var[Y_i]+Var[Z_i]+2Cov(Y_i,Z_i)\right) \\
&= \frac{1}{4}\left(2\sigma^2+2Cov(Y_i,Z_i)\right) \\
&= \frac{\sigma^2}{2}+\frac{Cov(Y_i,Z_i)}{2N} \\
&= \frac{\sigma^2}{2}\left(1+\frac{Cov(Y_i,Z_i)}{\sigma^2}\right) \\
&= \frac{\sigma^2}{2}\left(1+\rho\right)
\end{aligned}
$$

Since $\left(\frac{Y_i+Z_i}{2}\right)$'s are N number of iid samples with a finite population mean θ and variance $\frac{\sigma^2}{2}(1+\rho)$, by CLT, we have,

$$
\frac{\hat{\theta}_{N,N}-\theta}{\sigma(\sqrt{1+\rho})/\sqrt{2N}} \xrightarrow{D} N(0,1)
$$

Now we want to bound the absolute error of the estimate to be less than or equal to ϵ with a probability $1-\delta$ as follows.

$$
P(|\hat{\theta}_{N,N}-\theta| \le \epsilon) \ge 1-\delta
$$

By multiplying a constant $\frac{\sqrt{2N}}{\sigma\sqrt{1+\rho}} \ge 0$, $\because \rho \ge -1$

$$
\begin{aligned}
P(|\hat{\theta}_{N,N}-\theta| \le \epsilon) \ge 1-\delta \\
P\left(\frac{\sqrt{2N}}{\sigma\sqrt{1+\rho}}|\hat{\theta}_{N,N}-\theta| \le \frac{\sqrt{2N}}{\sigma\sqrt{1+\rho}}\epsilon\right) \ge 1-\delta
\end{aligned}
$$

Here we've only switched the σ of the inequality from (a) to $\sigma\sqrt{1+\rho}$ and since we have the asymptotic distribution of $\hat{\theta}_{N,N}-\theta$, the same derivation gives the following result,

$$
N \ge \left(\frac{1.96\sigma\sqrt{1+\rho}}{\epsilon}\right)^2/2 = \left(\frac{1.96\sigma_{N,N}}{\epsilon}\right)^2/2
$$

and the confidence interval

$$
\hat{\theta}_{N,N} \pm z_\delta\frac{\sigma\sqrt{1+\rho}}{\sqrt{2N}} = z_\delta\frac{\sigma_{N,N}}{\sqrt{2N}}
$$

```
np.random.seed(12)

n0 = 100
zdelta = 1.96
epsilon = 0.01
true_solution = np.sin(1)
```

```python
def f(x):
    return np.cos(x)

# Antithetic Variates
u = np.random.uniform(0, 1, size=n0)
integrand = (f(u) + f(1 - u)) / 2
var_av_pilot = np.var(integrand)
N_av = int(np.ceil((zdelta ** 2 * var_av_pilot / epsilon ** 2) / 2))

u = np.random.uniform(0, 1, size=N_av)
integrand_av = (f(u) + f(1 - u)) / 2
theta_av = np.mean(integrand_av)
var_av = np.var(integrand_av)
error_av = zdelta * np.sqrt((var_av) / (2 * N_av))
lb_av = theta_av - error_av
ub_ab = theta_av + error_av
rho_1 = np.corrcoef(f(u), f(1 - u))[0, 1]

print("Antithetic Variates method: ")
print("-"*50)
print("Required number of samples = {}".format(N_av))
print("Estimated theta = {:.5f}".format(theta_av))
print("Confidence interval: [{:.5f}, {:.5f}]".format(lb_av, ub_ab))
print(f"Half-length of CI: {error:.5f}")
print("Error = {:.5f}".format(theta_av - true_solution))
print("True theta = {:.5f}".format(true_solution))
```

```
Antithetic Variates method:
--------------------------------------------------
Required number of samples = 26
Estimated theta = 0.84726
Confidence interval: [0.84018, 0.85433]
Half-length of CI: 0.00977
Error = 0.00579
True theta = 0.84147
```

Part 1c

How much did the necessary sample size drop from part (a) to part (b)? (Be sure you're comparing equivalent values: For example, N vs N instead of $2N$ vs N.)

```python
print("Reduction in samples (number) = {:}".format(N-N_av))
print("Reduction in samples (percentage) = {:.5f}%".format(100 * (1 - N_av / N)))
```

```
Reduction in samples (number) = 372
Reduction in samples (percentage) = 93.46734%
```

Problem 2

Consider the integral

$$\theta = \int_0^1 \cos(6x)\,\mathrm{d}x$$

Part 2a

Estimate this integral using direct Monte Carlo simulation. First use a pilot study of size $N_0 \geq 100$ to estimate how many samples you would need to ensure that $P(|\hat{\theta}_{2N} - \theta| \leq 0.01) \geq 0.95$, then use a full-sized simulation with the appropriate number of samples. Report the required number of samples, your estimate of the integral, a confidence interval for your estimate and the exact error.

```
np.random.seed(12)

n0 = 10000
zdelta = 1.96
epsilon = 0.01
true_solution = np.sin(6) / 6

def f(x):
    return np.cos(6*x)

# Direct method

# Direct method
# Pilot study
u = np.random.uniform(0, 1, size=n0)
integrand = f(u)
var_pilot = np.var(integrand)
N = int(np.ceil((zdelta ** 2 * var_pilot / epsilon ** 2) / 2))

# Full study
u = np.random.uniform(0, 1, size=2 * N)
X = f(u)
theta = np.mean(X)
var = np.var(X)
error = zdelta * np.sqrt(var / (2 * N))
lb = theta - error
ub = theta + error

print("Direct method ")
print("-"*50)
print("Required number of samples = {}".format(2 * N))
print("Estimated theta = {:.5f}".format(theta))
print("Confidence interval: [{:.5f}, {:.5f}]".format(lb, ub))
print(f"Half-length of CI: {error:.5f}")
print("Error = {:.5f}".format(theta - true_solution))
print("True theta = {:.5f}".format(true_solution))
```

```
Direct method
--------------------------------------------------
Required number of samples = 18062
Estimated theta = -0.04612
Confidence interval: [-0.05615, -0.03610]
Half-length of CI: 0.01003
Error = 0.00045
True theta = -0.04657
```

Part 2b

Repeat part (a), but use antithetic variates for both the pilot study and the full simulation.

```python
np.random.seed(12)

n0 = 100
zdelta = 1.96
epsilon = 0.01
true_solution = np.sin(6) / 6

def f(x):
    return np.cos(6*x)

# Antithetic Variates
u = np.random.uniform(0, 1, size=n0)
integrand = (f(u) + f(1 - u)) / 2
var_av_pilot = np.var(integrand)
N_av = int(np.ceil((zdelta ** 2 * var_av_pilot / epsilon ** 2) / 2))

u = np.random.uniform(0, 1, size=N_av)
integrand_av = (f(u) + f(1 - u)) / 2
theta_av = np.mean(integrand_av)
var_av = np.var(integrand_av)
error_av = zdelta * np.sqrt((var_av) / (2 * N_av))
lb_av = theta_av - error_av
ub_ab = theta_av + error_av
rho_2 = np.corrcoef(f(u), f(1 - u))[0, 1]

print("Antithetic Variates method: ")
print("-"*50)
print("Required number of samples = {}".format(N_av))
print("Estimated theta = {:.5f}".format(theta_av))
print("Confidence interval: [{:.5f}, {:.5f}]".format(lb_av, ub_ab))
print(f"Half-length of CI: {error:.5f}")
print("Error = {:.5f}".format(theta_av - true_solution))
print("True theta = {:.5f}".format(true_solution))
```

```
Antithetic Variates method:
--------------------------------------------------
Required number of samples = 10496
Estimated theta = -0.05137
Confidence interval: [-0.06054, -0.04219]
Half-length of CI: 0.01003
Error = -0.00480
True theta = -0.04657
```

Part 2c

How much did the necessary sample size drop from part (a) to part (b)? (Be sure you're comparing equivalent values: For example, N vs N instead of $2N$ vs N.)

Note that if your pilot study is too small, then the estimates for N will be very noisy. You might want to run this several times or experiment with different values of N_0 to get a sense for the typical behavior.

```python
print("Reduction in samples (number) = {:}".format(N-N_av))
print("Reduction in samples (percentage) = {:.5f}%".format(100 * (1 - N_av / N)))
```

```
Reduction in samples (number) = -1465
Reduction in samples (percentage) = -16.22190%
```

Part 2d

You should find that problems 1 and 2 have drastically different behavior. Explain why.

```
print("Correlation of antithetic variates in (a)= {:.5f}".format(rho_1))
print("Correlation of antithetic variates in (b) = {:.5f}".format(rho_2))
```

```
Correlation of antithetic variates in (a)= -0.91440
Correlation of antithetic variates in (b) = 0.95532
```

The variance of antithetic estimator $\hat{\theta}_{N,N}$ is proportional to $(1+\theta)$. Since the $\rho_b > 0$, the antithetic variates method does not reduce the variance, rather increase it. This is why the problem 1 and 2 show drastically different behavior.

Problem 3

In this problem, you will explore several methods to estimate

$$\theta = \mathbb{E}\left[X^2\right]$$

where $X \sim N(0,4)$ is a normal random variable with $\mu = 0$ and $\sigma^2 = 4$.

Part 3a

Use the direct approach with $N = 10,000$ samples. Report your estimate of θ along with a confidence interval and the half-length of the confidence interval (i.e., the size of the error bars).

```
import numpy as np

n = 10000
zdelta = 1.96
true_solution = 4

def box_muller(n:int = 1000):
    out = np.empty(n)

    m = (n + 1) // 2
    size = (2, m)
    lamda = 1 / 2

    u1, u2 = np.random.random(size)

    v = -np.log(u1) / lamda
    w = u2 * 2 * np.pi

    x = np.sqrt(v) * np.cos(w)
    y = np.sqrt(v) * np.sin(w)

    out[:m] = x
    out[-m:] = y

    return out

# Direct method
# x = np.random.normal(0, 2, size=n)
z = box_muller(n)
x = 2 * z
```

```python
x = x ** 2
theta = np.mean(x)
var = np.var(x)
error = zdelta * np.sqrt(var / n)

print("Direct method: ")
print("Estimated theta = {:.5f}".format(theta))
print(f"Confidence interval = {np.round(theta + np.array([-1, 1]) * error, 5)}")
print(f"Half-length of CI: {error:.5f}")
print("True error = {:.5f}".format(theta - true_solution))
```

```
Direct method:
Estimated theta = 3.95153
Confidence interval = [3.84248 4.06059]
Half-length of CI: 0.10905
True error = -0.04847
```

Part 3b

Use antithetic variates with $N = 5,000$ samples. (Each sample should be the average of two antithetic variates - one generated through Z and the other through $-Z$. Report your estimate of θ along with a confidence interval and the half-length of the confidence interval (i.e., the size of the error bars). By what percentage did you reduce the CI length from part (a)? Explain why this happens.

```python
import numpy as np
np.random.seed(12)

n = 5000
zdelta = 1.96
true_solution = 4

# Antithetic variates
# x = np.random.normal(0, 2, size=n)
z = box_muller(n)
x = 2 * z
f = lambda x: x**2
integrand_av = (f(x) + f(-x)) / 2
theta_av = np.mean(integrand_av)
var_av = np.var(integrand_av)
error_av = zdelta * np.sqrt(var_av / n)

print("Antithetic Variates method: ")
print("Estimated theta = {:.5f}".format(theta_av))
print(f"Confidence interval = {np.round(theta_av + np.array([-1, 1]) * error_av, 5)}")
print(f"Half-length of CI = {error_av:5f}")
print("True error = {:.5f}".format(theta_av - true_solution))
print("\nError reduction = {:.5f}%".format(100 * (1 - error_av / error)))
# print("Correlation of antithetic variates = {:.5f}".format(np.corrcoef(f(u), f(1 - u))[0, 1]))
```

```
Antithetic Variates method:
Estimated theta = 4.01451
Confidence interval = [3.85987 4.16915]
Half-length of CI = 0.154641
True error = 0.01451
```

```
Error reduction = -41.80507%
```

The function $f(x) = x^2$ is not monotonic, rather f is an even function such that $f(x) = f(-x)$.

Part 3c

Use X as a control variate and use $N = 10,000$ samples. Report your estimate of θ along with a confidence interval and the half-length of the confidence interval (i.e., the size of the error bars). By what percentage did you reduce the CI length from part (a)? Explain why this happens.

$$
\begin{aligned}
Cov(X, X^2) &= E[(X - E[X])(X^2 - E[X^2])] \\
&= E[(X - 0)(X^2 - 4)] \\
&= E[X^3 - 4X] \\
&= E[X^3] - 4E[X] \\
&= 0
\end{aligned}
$$

$$
c^* = \frac{Cov(X, X^2)}{Var(X)} = 0
$$

```python
import numpy as np

c = 0
zdelta = 1.96
n = 10000

# Control variate
x_c = x**2 + c * (x - 0)
theta_c = np.mean(x_c)
variance_c = np.var(x_c)
error_c = zdelta * np.std(x_c) / np.sqrt(n)

print("Control variate: ")
print("Expected value = {:.5f}".format(theta_c))
print("Error = {:.5f}".format(theta_c - 4))
print("Confidence Interval: lb = {:.5f}, ub = {:.5f}".format(theta_c - error_c, theta_c + error_c))

print("Error reduction = {:.5f}%".format(100 * (1 - error_c / error)))
print("Sqrt(1 - rho^2) = {:.5f}".format(1 - np.sqrt(1 - np.corrcoef(X, X**2)[0,1])))
```

```
Control variate:
Expected value = 4.01451
Error = 0.01451
Confidence Interval: lb = 3.90516, ub = 4.12385
Error reduction = -21.35130%
Sqrt(1 - rho^2) = 0.69385
```

The size of our confidence interval decrease by a factor of $\sqrt{1 - \rho^2} \approx 0.7$. Thus, we get variance reduction.

Asian Options

An Asian option is an option on the time average of an underlying asset. An Asian call option has payoff $\max\left\{\overline{S} - K, 0\right\}$, where K is the strike price and $\overline{S}$ is the time-averaged value of the underlying asset. This means that the value of the option at time zero is given by $e^{-rT}\left\{\overline{S} - K, 0\right\}$, where T is the maturity. There are several different flavors of Asian options, depending on exactly how $\overline{S}$ is defined.

An Asian option is said to be *discretely monitored* if $\overline{S}$ depends on the value of S_t at a discrete set of times t_1, ..., t_n. (The alternative is a *continuously monitored* option, where $\overline{S}$ depends on the entire trajectory of S_t.)

For example, we might define

$$\overline{S} \equiv S_A = \frac{1}{n} \sum_{i=1}^{n} S_{t_i}$$

The quantity S_A is called an *arithmetic* average.

We could also define

$$\overline{S} \equiv S_G = \left(\prod_{i=1}^{n} S_{t_i} \right)^{1/n}$$

The quantity S_G is called a *geometric* average.

Arithmetic Asian options are more common, but the geometric version admits a closed form solution. In particular, if we choose $t_i = i\Delta t$ where $\Delta t = T/n$ and $i = 1, \dots, n$ for some positive integer n, then the value of the geometric call at time 0 is

$$C_G = S_0 \exp\left(-r \left(\frac{\Delta t(n-1)}{2} \right) - \frac{\sigma^2}{2} \left(\frac{\Delta t(n^2-1)}{6n} \right) \right) \Phi\left(d + \sigma \sqrt{(2n^2+3n+1)\Delta t/(6n)} \right) - K e^{-rT} \Phi\left(d\right)$$

where

$$d = \frac{\ln\left(S_0/K\right) + (r - \sigma^2/2) \cdot (\Delta t(n+1)/2)}{\sigma \sqrt{(2n^2+3n+1)\Delta t/(6n)}}$$

and Φ is the cdf of the standard normal distribution.

Problem 4

Consider a geometric Asian call option with the following parameters:

$$\begin{aligned} S_0 &= 50,\ K = 50, \\ T &= 1,\ n = 5 \\ r &= 0.05,\ \sigma = 0.2 \end{aligned}$$

and $t_i = i\Delta t$ where $\Delta t = T/n$ and $i = 1, \dots, n$.

Choose a value of N no less than $5,000$ to use for both of the following parts.

Assuming $S \sim \text{GBM}(r, \sigma)$,

$$S_t = S_0 \exp\left(\left(r - \frac{\sigma^2}{2} \right) t + \sigma W_t \right), \qquad \text{where } W_t \sim N(0, t)$$

Part 4a

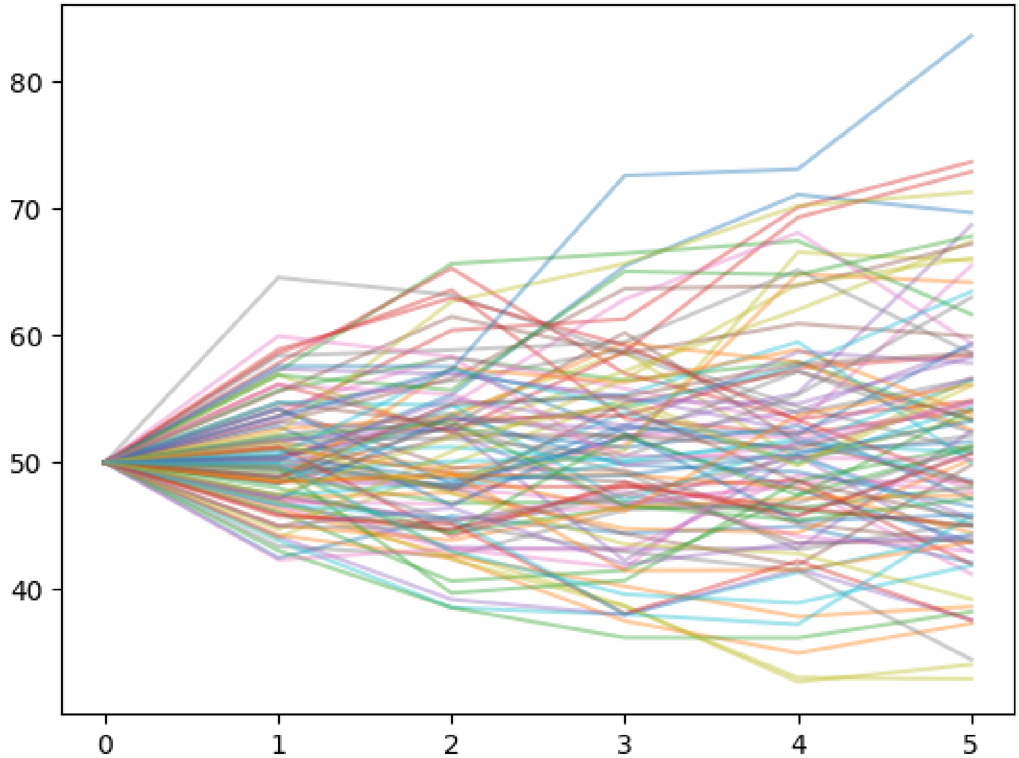
Simulate the underlying asset S_t, then use that simulation to calculate $X = e^{-rT} \max\{S_G - K,\, 0\}$. Average together $2N$ i.i.d. samples of X to estimate the value of the option. Report your estimate of θ along with a confidence interval and the half-length of the confidence interval (i.e., the size of the error bars) and the error between your estimate and the exact value.

```python
# Define parameters
S0 = 50
K = 50
T = 1
n = 5
r = 0.05
sig = 0.2
N = 5000
dt = T/n
time = np.arange(n+1) * dt


# Closed form solution
d = np.log(S0 / K) + (r - sig**2/2) * (dt * (n+1) / 2)
A = sig * np.sqrt((2*n**2 + 3 * n + 1) * dt / (6*n))
d /= A
phi = stats.norm.cdf
B = -r * (dt * (n-1) / 2) - (sig**2 / 2) * ((dt * (n**2 - 1)) / (6 * n))

true_solution = S0 * np.exp(B) * phi(d + A) - K * np.exp(-r * T) * phi(d)
print(f"true solution = {true_solution}")
```

```
true solution = 3.2472467790281065
```

```python
np.random.seed(12)

S = np.ones((2*N, len(time)))
S[:, 0] = S0

mean = (r - sig ** 2 / 2) * dt
sd = sig * np.sqrt(dt)

for k in range(2*N):
    z = box_muller(n)
    log_diff = mean + sd * z
    S[k, 1:] = S[k, 0] * np.exp(log_diff).cumprod()
```

```python
import matplotlib.pyplot as plt

for s in S[:100]:
    plt.plot(s, alpha=0.4)
```




```
SG = np.exp(np.log(S[:, 1:]).mean(axis=1))
X = np.exp(-r * T) * np.maximum(SG-K, 0)

zdelta = 1.96

# Direct method
theta = np.mean(X)
variance = np.var(X)
error = zdelta * np.std(X) / np.sqrt(2 * N)

print("Direct method: ")
print("Estimated theta = {:.5f}".format(theta))
print("Error = {:.5f}".format(theta - true_solution))
print("Confidence Interval: lb = {:.5f}, ub = {:.5f}".format(theta - error, theta + error))
print(f"Half-length of CI = {error:.5f}")
```

```
Direct method:
Estimated theta = 3.24016
Error = -0.00708
Confidence Interval: lb = 3.15280, ub = 3.32753
Half-length of CI = 0.08736
```

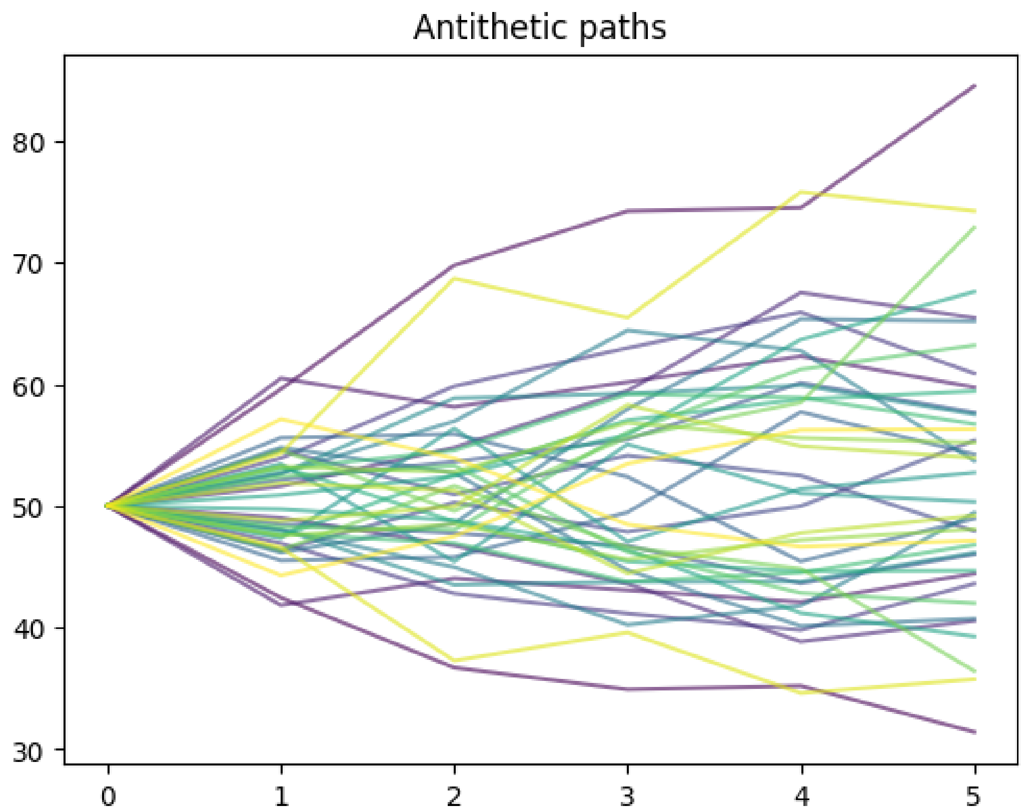
Part 4b

Repeat the previous part using antithetic variates. That is, use N samples of $(X+Y)/2$, where Y is identically distributed to X, but negatively correlated. Report your estimate of θ along with a confidence interval and the half-length of the confidence interval (i.e., the size of the error bars) and the error between your estimate and the exact value.

```python
# Antithetic variate

S = np.zeros((2*N, n+1))
S[:, 0] = S0

mean = (r - sig ** 2 / 2) * dt
sd = sig * np.sqrt(dt)

for k in range(N):
    z = box_muller(n)
    log_diff = mean + sd * z
    S[k, 1:] = S[k, 0] * np.exp(log_diff).cumprod()
    log_diff = mean + sd * -z
    S[N+k, 1:] = S[N+k, 0] * np.exp(log_diff).cumprod()

SG = np.exp(np.log(S[:, 1:]).mean(axis=1))
X = np.exp(-r * T) * np.maximum(SG-K, 0)

zdelta = 1.96

theta_av = np.mean(X)
variance_av = np.var(X)
error_av = zdelta * np.std(X) / np.sqrt(2 * N)
lb, ub = theta_av - error_av, theta_av + error_av

print("Antithetic variate: ")
print("Expected value = {:.5f}".format(theta_av))
print("Error = {:.5f}".format(theta_av - true_solution))
print("Confidence Interval: lb = {:.5f}, ub = {:.5f}".format(lb, ub))
print(f"Half-length of CI = {error_av:.5f}")
```

```
Antithetic variate:
Expected value = 3.25466
Error = 0.00741
Confidence Interval: lb = 3.16604, ub = 3.34327
Half-length of CI = 0.08862
```

```python
num_paths = 20
colors = np.linspace(0, 1, num_paths)
for k in range(num_paths):
    for s in (S[k], S[5000+k]):
        plt.plot(s, alpha=0.6, color=plt.cm.viridis(colors[k]))
plt.title("Antithetic paths")
plt.show()
```



Part 4c

How much did the method in part b reduce the size of your confidence interval? That is, what is

$$\left(1 - \frac{s_b}{s_a}\right) \times 100$$

where s_a and s_b are the standard deviations of your samples from parts (a) and (b), respectively.

```
sa = np.sqrt(variance)
sb = np.sqrt(variance_av)
print(f"Variance reduction {(1-sb/sa)*100:.5f}%")
```

```
Variance reduction -1.43405%
```

Problem 5

Consider an arithmetic Asian call option with the following parameters (the same as in the previous problem):

$$S_0 = 50,\ K = 50,$$
$$T = 1,\ n = 5$$
$$r = 0.05,\ \sigma = 0.2$$

and $t_i = i\Delta t$ where $\Delta t = T/n$ and $i = 1, \ldots, n$.

Choose a value of N no less than 10,000 to use for both of the following parts.

Part 5a

Simulate the underlying asset S_t, then use that simulation to calculate $X = e^{-rT}\max\{S_A - K, 0\}$. Average together N i.i.d. samples of X to estimate the value of the option. Report your estimate of θ along with a confidence interval and the half-length of the confidence interval (i.e., the size of the error bars).

```
S0 = 50
K = 50
T = 1
n = 5
r = 0.05
sig = 0.2

N = 10000

dt = T/n
time = np.arange(n+1) * dt


S = np.ones((N, len(time)))
S[:, 0] = S0

mean = (r - sig ** 2 / 2) * dt
sd = sig * np.sqrt(dt)

for k in range(N):
    z = box_muller(n)
    log_diff = mean + sd * z
    S[k, 1:] = S[k, 0] * np.exp(log_diff).cumprod()

SA = S[:, 1:].mean(axis=1)
X = np.exp(-r * T) * np.maximum(SA-K, 0)

zdelta = 1.96

# Direct method
theta = np.mean(X)
variance = np.var(X)
error = zdelta * np.std(X) / np.sqrt(N)

print("Direct method: ")
print("Expected value = {:.5f}".format(theta))
print("Confidence Interval: lb = {:.5f}, ub = {:.5f}".format(theta - error, theta + error))
print(f"Half interval = {error:.5f}")
```

```
Direct method:
Expected value = 3.30879
Confidence Interval: lb = 3.21868, ub = 3.39890
Half interval = 0.09011
```

Part 5b

Use the value of the geometric call as a control variate. Report your estimate of θ along with a confidence interval and the half-length of the confidence interval (i.e., the size of the error bars).

```python
# Control variate

SG = np.exp(np.log(S[:, 1:]).mean(axis=1))
Z = np.exp(-r * T) * np.maximum(SG-K, 0)

SA = S[:, 1:].mean(axis=1)
X = np.exp(-r * T) * np.maximum(SA-K, 0)


zdelta = 1.96

c = -np.cov(X, Z)[0, 1] / np.var(Z)
x_c = X + c * (Z - np.mean(Z))
theta_c = np.mean(x_c)
variance_c = np.var(x_c)
error_c = zdelta * np.std(x_c) / np.sqrt(n)

print("Control variate: ")
print("Expected value = {:.5f}".format(theta_c))
print("Confidence Interval: lb = {:.5f}, ub = {:.5f}".format(theta_c - error_c, theta_c + error_c))
print(f"Half interval = {error_c:.5f}")
```

```
Control variate:
Expected value = 3.30879
Confidence Interval: lb = 3.19431, ub = 3.42327
Half interval = 0.11448
```

Part 5c

How much did the method in part b reduce the size of your confidence interval? That is, what is

$$\left(1 - \frac{s_b}{s_a}\right) \times 100$$

where s_a and s_b are the standard deviations of your samples from parts (a) and (b), respectively.

```python
sa = np.sqrt(variance)
sb = np.sqrt(variance_c)
print(f"Variance reduction {(1-sb/sa)*100:.5f}%")
```

```
Variance reduction 97.15915%
```