Stochastic Simulation
Homework 3

Spring 2025

1. Input Analysis

Solution. The log likelihood function

$$l(X,\gamma,\beta) = \log \prod_{i=1}^{n} \frac{1}{\beta} e^{-(x_i-\gamma)/\beta} = -n\log(\beta) - \frac{\sum_{i=1}^{n}(x_i-\gamma)}{\beta} \tag{1}$$

Notice that $\gamma \le \min\{x_1,\dots,x_n\}$. To find the MLEs $\hat{\gamma}$ and $\hat{\beta}$ is equivalent to solve the following constrained optimization problem:

$$\max_{\gamma,\beta} l(X,\gamma,\beta)$$
$$\text{s.t. } \gamma \le \min\{x_1,\dots,x_n\}$$

Take derivatives with respect to γ and β respectively:

$$\frac{\partial l}{\partial \gamma} = \frac{n}{\beta} > 0, \quad \frac{\partial l}{\partial \beta} = -\frac{n}{\beta} + \frac{\sum_{i=1}^{n} x_i - n\gamma}{\beta^2}$$

Since $\frac{\partial l}{\partial \gamma} > 0$ implies l is monotonically increasing in γ, the maximum is achieved at $\hat{\gamma} = \min\{x_1,\dots,x_n\}$. Setting $\frac{\partial l}{\partial \beta} = 0$ yields:

$$\hat{\beta} = \frac{\sum_{i=1}^{n} x_i}{n} - \hat{\gamma} = \bar{x} - \min\{x_1,\dots,x_n\}$$

2. Output Analysis

Solution. The MLE for p is $\hat{p} = \frac{\sum_{i=1}^{n} x_i}{n} = \hat{x}_n$. The estimate for $1/p$ is $1/\hat{p} = 1/\hat{x}_n$.

To build the confidence interval of $1/\hat{p}$, we need to specify the variance of this estimator. Applying CLT:

$$\sqrt{n}(\hat{x}_n - \bar{x}) \xrightarrow{d} N\left(0,\sigma^2\right)$$

where $\sigma^2 = \mathrm{Var}(X_1) = p(1-p)$. Notice that the MLE for the covariance is $\hat{\sigma}^2 = \hat{x}_n(1-\hat{x}_n)$. Use Delta method with $h(p) = 1/p$ and $h'(p) = -1/p^2$:

$$\sqrt{n}\left(\frac{1}{\hat{p}} - \frac{1}{p}\right) \approx -\frac{1}{p^2}\sqrt{n}(\hat{p}-p) \xrightarrow{d} N\left(0, \frac{p(1-p)}{p^4}\right)$$

The 95% confidence interval for $1/p$ is:

$$\left[\frac{1}{\hat{p}} - \phi_{0.975}\frac{\sqrt{\hat{p}(1-\hat{p})}}{\hat{p}^2\sqrt{n}}, \frac{1}{\hat{p}} + \phi_{0.975}\frac{\sqrt{\hat{p}(1-\hat{p})}}{\hat{p}^2\sqrt{n}}\right]$$

or

$$\left[\frac{1}{\hat{x}_n} - \phi_{0.975}\frac{\sqrt{\hat{x}_n(1-\hat{x}_n)}}{\hat{x}_n^2\sqrt{n}}, \frac{1}{\hat{x}_n} + \phi_{0.975}\frac{\sqrt{\hat{x}_n(1-\hat{x}_n)}}{\hat{x}_n^2\sqrt{n}}\right].$$

#3 (Confidence Interval)

Recall that in the lecture we derived the variance of AR(1) time series:

$$Z_n = aZ_{n-1} + \varepsilon_n, \quad \varepsilon_n \overset{i.i.d.}{\sim} N(0,1) \text{ and } Z_1 \sim N\left(0, \frac{1}{1-a^2}\right).$$

We will use this model to test and compare the following different methods for constructing confidence intervals (CI) for $\bar{Z}_n$:

(1) **Construct CI using the true value of the limiting variance $\sigma^2 = \frac{1+a}{1-a} \cdot \frac{1}{1-a^2}$.**
(2) **Construct CI using sample variance**, which is incorrect as discussed in the lecture.
(3) **Construct CI using the sectioning method** without using information of the limiting variance.

Let $a = 0.5$ and repeat the following procedure for 1000 rounds:

1. Generate a sequence $Z_1, Z_2, \ldots, Z_n$ with $n = 500$.
2. Compute the 95% CIs for $\bar{Z}_n$ using the three approaches. *(For the sectioning method, choose the batch size via trial runs.)*
3. Count whether the CI covers the true value $z = 0$ for each approach.

Report:

- The proportion of times the true value is covered by the CI for each approach in 1000 rounds.
- Analysis and evaluation of the three approaches.

In [1]:
```python
import numpy as np
import scipy as sp
```

In [2]:
```python
## some basic functions and constants
def generate_z(n = 500, seed = 0):
    np.random.seed(seed)
    epsilon = np.random.normal(0, 1, n)
    z = np.zeros(n)
    # z[0] = sp.stats.norm.rvs(0, 1.0/0.75)
    z[0] = np.random.normal(0, 1.0/0.75)
    for i in range(1, n):
        z[i] = 0.5 * z[i-1] + epsilon[i]
    return z
sample_mean = lambda z : np.mean(z)
sample_var = lambda z : np.var(z, ddof=1) # ddof=1 for unbiased variance
phi = sp.stats.norm.ppf(0.975) # 1.96
n = 500
```

(1) Construct CI using the true limiting variance

When $a = 0.5$, we have the true limiting variance $\sigma^2 = 4$. Since we are using the correct variance, the frequency that $z = 0$ is covered in the CI should be close to 95%. The simulation result is as follows.

In [3]:
```
count_cover1 = 0
sigma_sq = 4
for iter in range(1000):
    z = generate_z(n, iter)
    ci_lb = sample_mean(z) - phi * np.sqrt(sigma_sq/n)
    ci_ub = sample_mean(z) + phi * np.sqrt(sigma_sq/n)
    if (ci_lb < 0) and (ci_ub > 0):
        count_cover1 += 1
print("When using the true limiting variance in the 1000 simulations,\n"
"the proportion that the true mean is covered by the confidence \n"
"interval is {:.2%}.".format(count_cover1/1000))
```

```
When using the true limiting variance in the 1000 simulations,
the proportion that the true mean is covered by the confidence
interval is 94.80%.
```

(2) Construct CI using the sample variance

Since we are using the wrong estimation of the variance, the probability that the true value is covered in the CI should be significantly lower than 95%. By numerical simulation we have a proportion of 76%.

In [4]:
```
count_cover2 = 0
for iter in range(1000):
    z = generate_z(n, iter)
    var = sample_var(z)
    ci_lb = sample_mean(z) - phi * np.sqrt(var/n)
    ci_ub = sample_mean(z) + phi * np.sqrt(var/n)
    if (ci_lb < 0) and (ci_ub > 0):
        count_cover2 += 1
print("When using the sample variance in the 1000 simulations,\n"
"the proportion that the true mean is covered by the confidence \n"
"interval is {:.2%}.".format(count_cover2/1000))
```

```
When using the sample variance in the 1000 simulations,
the proportion that the true mean is covered by the confidence
interval is 76.00%.
```

(3) Using the sectioning method

Since we are using sectioning method to reduce the dependence among the samples, we should have higher proportion than in (2). However, the dependence cannot be ignored, and thus the proportion should be slightly lower than 95%. Here we choose $m = 20$ and $k = 25$, and the numerical simulation result is as follows.

In [5]:
```
m=20; k=25
count_cover3 = 0
for iter in range(1000):
    z = generate_z(n, iter)
    z_hat = sample_mean(z)
    section = z.reshape(m, k) # reshape z to m x k matrix
    group_mean = np.mean(section, axis=1) # compute the variance of each row
    group_var = np.var(group_mean, ddof=1) # compute the variance of the group m
    var = group_var * k
    ci_lb = sample_mean(z) - phi * np.sqrt(var/n)
    ci_ub = sample_mean(z) + phi * np.sqrt(var/n)
    if (ci_lb < 0) and (ci_ub > 0):
        count_cover3 += 1
print("When using the sectioning method in the 1000 simulations,\n"
"the proportion that the true mean is covered by the confidence \n"
"interval is {:.2%}.".format(count_cover3/1000))
```

```
When using the sectioning method in the 1000 simulations,
the proportion that the true mean is covered by the confidence
interval is 92.90%.
```

```
In [ ]:
```