STA219 Assignment 6

12312706 Zhou Liangyu

1. Standard normal distribution is symmetric, i.e. the probability only depends on the distance from the origin. Therefore, we can consider using the polar coordinate system to describe it.

   Let $R = \sqrt{-2\ln U_1}$, $\theta = 2\pi U_2$, we have $\theta \sim \mathrm{U}(0, 2\pi)$, $f_\theta(\omega) = \dfrac{1}{2\pi}$, and $Z_1 = R\cos\theta$, $Z_2 = R\sin\theta$.

   $\because P(R^2 \le t) = P(-2\ln U_1 \le t) = P(U_1 \ge e^{-\frac{t}{2}}) = 1 - e^{-\frac{t}{2}}$

   $\therefore R^2 \sim \mathrm{Exp}(\dfrac{1}{2})$.

   $\therefore F_R(r) = F_{R^2}(r^2) = 1 - e^{-\frac{r^2}{2}}$, $f_R(r) = \dfrac{dF_R(r)}{dr} = re^{-\frac{r^2}{2}}$.

   $\therefore R$ and θ are independent, $f_{R,\theta}(r, \omega) = f_R(r)f_\theta(\omega) = \dfrac{re^{-\frac{r^2}{2}}}{2\pi}$.

   $\because Z_1 = R\cos\theta$, $Z_2 = R\sin\theta$

   $\therefore R = Z_1^2 + Z_2^2$, $f_{Z_1,Z_2}(z_1, z_2) = \dfrac{1}{r}f_{R,\theta}(r, \omega) = \dfrac{e^{-\frac{z_1^2+z_2^2}{2}}}{2\pi} = \dfrac{1}{\sqrt{2\pi}}e^{-\frac{z_1^2}{2}} \cdot \dfrac{1}{\sqrt{2\pi}}e^{-\frac{z_2^2}{2}}$.

   $\therefore Z_1$ and Z_2 are independent, and $f_{Z_1}(z_1) = \dfrac{1}{\sqrt{2\pi}}e^{-\frac{z_1^2}{2}}$, $f_{Z_2}(z_2) = \dfrac{1}{\sqrt{2\pi}}e^{-\frac{z_2^2}{2}}$.

   $\therefore P(Z_1 \le a, Z_2 \le b) = \displaystyle\int_\infty^b \int_\infty^a f_{Z_1,Z_2}(z_1, z_2)\, dz_1 dz_2 = \int_\infty^a f_{Z_1}(z_1)\, dz_1 \int_\infty^b f_{Z_2}(z_2)\, dz_2 = \Phi(a)\Phi(b)$.

   $\therefore Z_1$ and Z_2 are a pair of independent standard normal random variables.

2. (1) Sample mean: $\overline{X} = \dfrac{1}{n}\displaystyle\sum_{i=1}^n X_i = \dfrac{1}{10}(17.2 + 22.1 + 18.5 + 17.2 + 18.6 + 14.8 + 21.7 + 15.8 + 16.3 + 22.8) = 18.5$.

   Sample variance: $S^2 = \dfrac{1}{n-1}\displaystyle\sum_{i=1}^n (X_i - \overline{X})^2 = \dfrac{1}{9}[(17.2 - 18.5)^2 + (22.1 - 18.5)^2 + (18.5 - 18.5)^2 + (17.2 - 18.5)^2 +$

   $(18.6 - 18.5)^2 + (14.8 - 18.5)^2 + (21.7 - 18.5)^2 + (15.8 - 18.5)^2 + (16.3 - 18.5)^2 + (22.8 - 18.5)^2]$

   ≈ 7.88.

   Sample standard deviation: $S = \sqrt{S^2} \approx 2.81$.

   (2) Sample lower quartile: $Q_{0.25} = X_{(\lfloor 10\times 0.25+1\rfloor)} = X_{(3)} = 16.3$.

   Sample upper quartile: $Q_{0.75} = X_{(\lfloor 10\times 0.75+1\rfloor)} = X_{(8)} = 21.7$.

   Sample interquartile range: $Q_{0.75} - Q_{0.25} = 21.7 - 16.3 = 5.4$.

3. (1) $\because X \sim \mathrm{U}(0, \theta)$

   $\therefore \mathrm{E}(X) = \dfrac{\theta}{2}$, $\mathrm{Var}(X) = \dfrac{\theta^2}{12}$, $f_X(x) = \dfrac{1}{\theta}$, $F_X(x) = \dfrac{x}{\theta}$.

   $\because f_{\min}(x) = nf(x)[1 - F(x)]^{n-1} = \dfrac{3}{\theta}(1 - \dfrac{x}{\theta})^2 = \dfrac{2(\theta - x)^2}{\theta^3}$, $f_{\max}(x) = nf(x)F(x)^2 = \dfrac{3}{\theta}(\dfrac{x}{\theta})^2 = \dfrac{3x^2}{\theta^3}$.

   $\therefore \mathrm{E}(X_{(1)}) = \displaystyle\int_0^\theta x f_{\min}(x)\, dx = \int_0^\theta \frac{3x(\theta - x)^2}{\theta^3}\, dx = -\frac{1}{\theta^3}\int_0^\theta x\, d(\theta - x)^3 = -\frac{1}{\theta^3}[0 - \int_0^\theta (\theta - x)^3\, dx] = \frac{\theta}{4}$,

   $\mathrm{E}(X_{(3)}) = \displaystyle\int_0^\theta x f_{\max}(x)\, dx = \int_0^\theta \frac{3x^3}{\theta^3}\, dx = \frac{3x^4}{4\theta^3}\bigg|_0^\theta = \frac{3\theta}{4}$.

   $\therefore \mathrm{E}(\hat{\theta}_1) = \dfrac{4}{3}\mathrm{E}(X_{(3)}) = \theta$, $\mathrm{E}(\hat{\theta}_2) = 4\mathrm{E}(X_{(1)}) = \theta$.

   $\therefore \hat{\theta}_1$ and $\hat{\theta}_2$ are both unbiased estimators of θ.

$$(2) \because \mathrm{E}(X_{(1)}^2) = \int_0^\theta x^2 f_{\min}(x)\,dx = \int_0^\theta \frac{3x^2(\theta-x)^2}{\theta^3}\,dx = \frac{1}{\theta^3}\int_0^\theta (\theta-x)^2\,dx^3 = \frac{1}{\theta^3}[0 - \int_0^\theta x^3\,d(\theta-x)^2] = \frac{\theta^2}{10},$$

$$\mathrm{E}(X_{(3)}^2) = \int_0^\theta x^2 f_{\max}(x)\,dx = \int_0^\theta \frac{3x^4}{\theta^3}\,dx = \frac{3x^5}{5\theta^3}\Big|_0^\theta = \frac{3\theta^2}{5}.$$

$$\therefore \mathrm{Var}(X_{(1)}) = \mathrm{E}(X_{(1)}^2) - \mathrm{E}(X_{(1)})^2 = \frac{\theta^2}{10} - \frac{\theta^2}{16} = \frac{3\theta^2}{80},\ \mathrm{Var}(X_{(3)}) = \mathrm{E}(X_{(3)}^2) - \mathrm{E}(X_{(3)})^2 = \frac{3\theta^2}{5} - \frac{9\theta^2}{16} = \frac{3\theta^2}{80}.$$

$$\therefore \mathrm{Var}(\hat\theta_1) = \frac{16}{9}\mathrm{Var}(X_{(3)}) = \frac{\theta^2}{15},\ \mathrm{Var}(\hat\theta_2) = 16\mathrm{Var}(X_{(1)}) = \frac{3\theta^2}{5}.$$

$\because \mathrm{Var}(\hat\theta_1) < \mathrm{Var}(\hat\theta_2)$

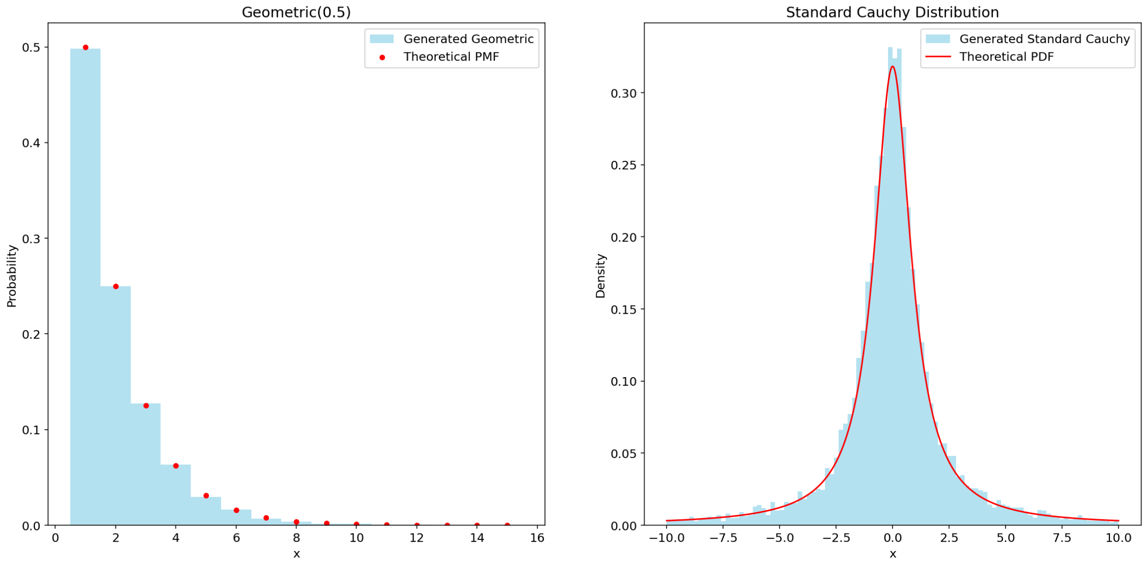
$\therefore \hat\theta_1$ is more efficient than $\hat\theta_2$.

4. Python Code:

```python
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import geom, cauchy

# Set seed
np.random.seed(20250510)

# Generate uniform samples
n_samples = 10000
unif_samples1 = np.random.uniform(0, 1, n_samples)
unif_samples2 = np.random.uniform(0, 1, n_samples)

# Transform
p_geom = 0.5
geo_samples = np.ceil(np.log(1 - unif_samples1) / np.log(1 - p_geom))
stdcauchy_samples = np.tan(np.pi * (unif_samples2 - 0.5))

# Theoretical PMF/PDF
geo_x = np.arange(min(geo_samples), max(geo_samples))
geo_pmf = geom.pmf(geo_x, p_geom)
stdcauchy_x = np.linspace(-10, 10, 1000)
stdcauchy_pdf = cauchy.pdf(stdcauchy_x)

# Adjust the plots
plt.figure(figsize=(19.5, 9), dpi=143.4)
plt.rcParams['font.size'] = 12
plt.tight_layout()

# Plot - Gerometric(0.5)
plt.subplot(1, 2, 1)
plt.hist(geo_samples, bins=np.arange(1, 17)-0.5, density=True, alpha=0.6, color='skyblue', label='Generated Geometric')
plt.scatter(geo_x, geo_pmf, color='red', s=20, marker='o', label='Theoretical PMF', zorder=2)
plt.xlabel('x')
plt.ylabel('Probability')
plt.title('Geometric(0.5)')
plt.legend()

# Plot - Standard Cauchy Distribution
plt.subplot(1, 2, 2)
plt.hist(stdcauchy_samples, bins=100, density=True, range=(-10, 10), alpha=0.6, color='skyblue', label='Generated Standard Cauchy')
plt.plot(stdcauchy_x, stdcauchy_pdf, 'r-', label='Theoretical PDF')
plt.xlabel('x')
```

```python
plt.ylabel('Density')
plt.title('Standard Cauchy Distribution')
plt.legend()

# Save the plot for assignment
plt.savefig('Geometric_StdCauchy_Generate.png')

# Show
plt.show()
```

Plots:

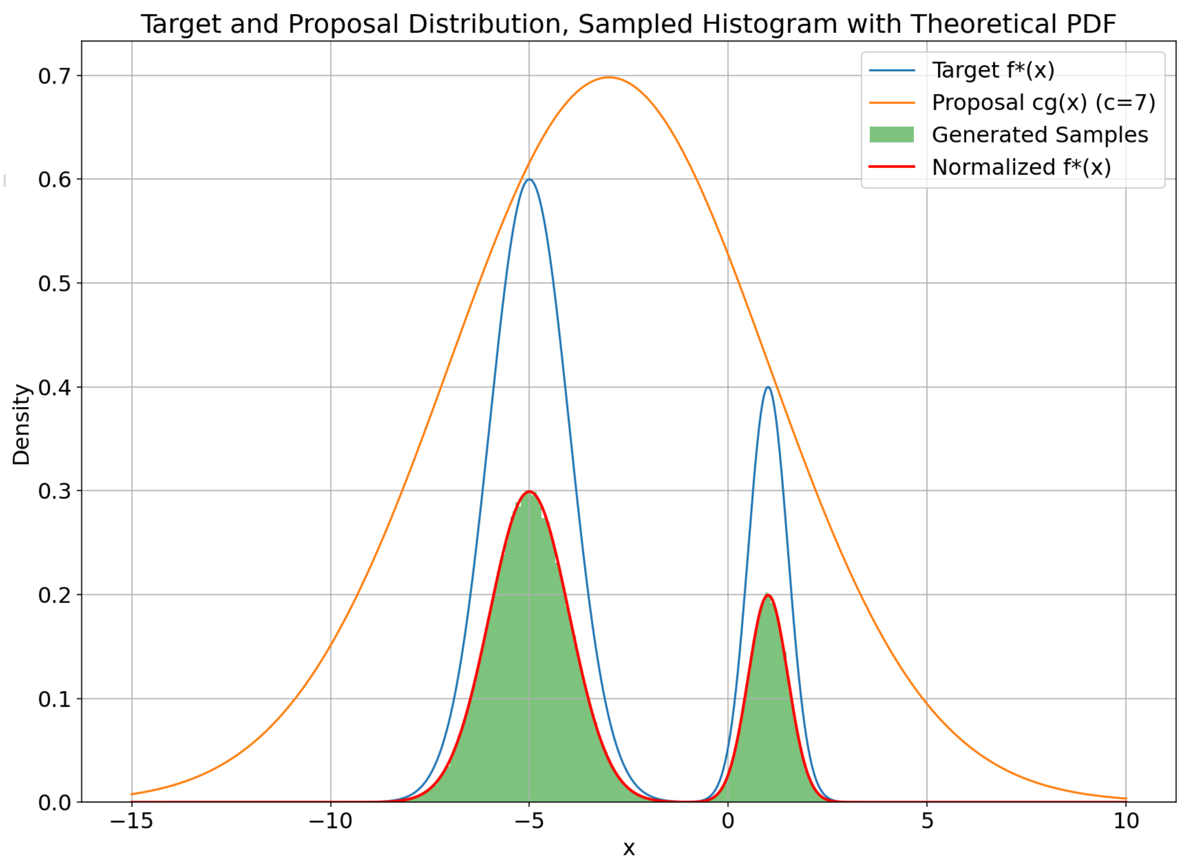


5. (1) (2) (3) Proposal distribution: $N(-3, 16)$. Acceptance proportion: 28.64%.

See the following Python code and results:

```python
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import norm
from scipy.integrate import quad
from scipy.optimize import minimize_scalar

# Reject sampling
def reject_sampling(n_samples, mu, sigma, c):
    samples = []
    accepted = 0
    total_trials = 0

    while accepted < n_samples:
        batch_size = 100000
        x_prop = np.random.normal(loc=mu, scale=sigma, size=batch_size)
        u = np.random.rand(batch_size)

        g_x = norm.pdf(x_prop, loc=mu, scale=sigma)
        accept_prob = fs(x_prop) / (c * g_x)

        accept_mask = u <= accept_prob
        accepted_samples = x_prop[accept_mask]

        samples.extend(accepted_samples)
```

```
        accepted += len(accepted_samples)
        total_trials += batch_size

    accept_rate = accepted / total_trials
    return np.array(samples[:n_samples]), accept_rate

fs = lambda x: 0.6 * np.exp(-((x + 5) ** 2) / 2) + 0.4 * np.exp(-((x - 1) ** 2) / 0.5)
c1, _ = quad(fs, -np.inf, np.inf)
mu = -3
sigma = 4
g = lambda x: norm.pdf(x, loc=mu, scale=sigma)
c = 7  # by trial and error

n_samples = 500000
samples, accept_rate = reject_sampling(n_samples, mu, sigma, c)
x = np.linspace(-15, 10, 1000)

# Set plot parameters
plt.figure(figsize=(14.3, 10), dpi=143.4)
plt.rcParams['font.size'] = 16
plt.tight_layout()

# Plot to show that it covers the target distribution, the histogram of the generated samples, and compare it with the theoretical PDF
plt.plot(x, fs(x), label='Target f*(x)')
plt.plot(x, c*g(x), label=f'Proposal cg(x) (c={c})')
plt.hist(samples, bins=200, density=True, alpha=0.6, label='Generated Samples')
plt.plot(x, fs(x)/c1, 'r-', label='Normalized f*(x)', linewidth=2)
plt.xlabel('x')
plt.ylabel('Density')
plt.legend()
plt.title("Target and Proposal Distribution, Sampled Histogram with Theoretical PDF")
plt.grid(True)
plt.savefig('RejectSampling.png')
plt.show()

print(f"Acceptance Rate = {accept_rate:.2%}")
```



Code running result:

```
PS C:\Users\Chaos Zhou\PyCharmProjects\STA219> python -u 'c:\Users\Chaos Zhou\PyCharmProjects\STA219\Assignment6_P5.py'
Acceptance Rate = 28.65%
PS C:\Users\Chaos Zhou\PyCharmProjects\STA219> python -u 'c:\Users\Chaos Zhou\PyCharmProjects\STA219\Assignment6_P5.py'
Acceptance Rate = 28.62%
PS C:\Users\Chaos Zhou\PyCharmProjects\STA219> python -u 'c:\Users\Chaos Zhou\PyCharmProjects\STA219\Assignment6_P5.py'
Acceptance Rate = 28.64%
PS C:\Users\Chaos Zhou\PyCharmProjects\STA219> python -u 'c:\Users\Chaos Zhou\PyCharmProjects\STA219\Assignment6_P5.py'
Acceptance Rate = 28.64%
PS C:\Users\Chaos Zhou\PyCharmProjects\STA219> python -u 'c:\Users\Chaos Zhou\PyCharmProjects\STA219\Assignment6_P5.py'
Acceptance Rate = 28.64%
PS C:\Users\Chaos Zhou\PyCharmProjects\STA219> python -u 'c:\Users\Chaos Zhou\PyCharmProjects\STA219\Assignment6_P5.py'
Acceptance Rate = 28.60%
PS C:\Users\Chaos Zhou\PyCharmProjects\STA219>
行 16，列 39   空格: 4
```

6. (1) Let $\hat{p}_n$ be the frequency of that more than 30% of the forest will eventually be burning.

By the CLT, we have $\hat{p}_n \overset{\text{approx.}}{\sim} \mathrm{N}(p, \dfrac{p(1-p)}{n})$.

$\therefore P(|\hat{p}_n - p| \leq 0.005) \approx 2\Phi(\dfrac{0.005\sqrt{n}}{\sqrt{p(1-p)}}) - 1 \geq 0.95 \Rightarrow \dfrac{0.005\sqrt{n}}{\sqrt{p(1-p)}} \geq \Phi^{-1}(0.975) \approx 1.96.$

$\therefore n \geq \dfrac{1.96^2 p(1-p)}{0.005^2} \geq \dfrac{1.96^2 \cdot 0.25}{0.005^2} = 38416.$

Python code:

```python
import random as rd
from collections import deque
import numpy as np

def forest_mc():
    rows, cols = 20, 50
    forest = [[False] * cols for _ in range(rows)]
```

```python
    forest[0][0] = True
    q = deque([(0, 0)])
    burning_tree_count = 1
    while q:
        i, j = q.popleft()
        # right
        if j + 1 < cols and not forest[i][j + 1]:
            if rd.random() < 0.8:
                forest[i][j + 1] = True
                burning_tree_count += 1
                q.append((i, j + 1))
        # down
        if i + 1 < rows and not forest[i + 1][j]:
            if rd.random() < 0.3:
                forest[i + 1][j] = True
                burning_tree_count += 1
                q.append((i + 1, j))
    return burning_tree_count

n = 38416
samples = []
count = 0
for _ in range(n):
    x = forest_mc()
    samples.append(x)
    if x > 300:
        count += 1
prob = count / n
mean_x = np.mean(samples)
std_x = np.std(samples, ddof=1)

print(f"(1) The probability of X > 30%: {prob:.4f}")
print(f"(2) Mean of X: {mean_x:.2f}")
print(f"(3) Standard deviation of X: {std_x:.2f}")
```

The probability that more than 30% of the forest will eventually be burning: 0.0035.

(2) The total number of affected trees X: 43.

(3) The corresponding standard deviation of X: 62.99.

Code running result:

```
PS C:\Users\Chaos Zhou\PyCharmProjects\STA219> python -u "c:\Users\Chaos Zhou\PyCharmProjects\STA219\a.py"
(1) The probability of X > 30%: 0.0035
(2) Mean of X: 43
(3) Standard deviation of X: 62.89
PS C:\Users\Chaos Zhou\PyCharmProjects\STA219> python -u "c:\Users\Chaos Zhou\PyCharmProjects\STA219\a.py"
(1) The probability of X > 30%: 0.0036
(2) Mean of X: 44
(3) Standard deviation of X: 63.24
PS C:\Users\Chaos Zhou\PyCharmProjects\STA219> python -u "c:\Users\Chaos Zhou\PyCharmProjects\STA219\a.py"
(1) The probability of X > 30%: 0.0037
(2) Mean of X: 43
(3) Standard deviation of X: 63.11
PS C:\Users\Chaos Zhou\PyCharmProjects\STA219> python -u "c:\Users\Chaos Zhou\PyCharmProjects\STA219\a.py"
(1) The probability of X > 30%: 0.0034
(2) Mean of X: 43
(3) Standard deviation of X: 62.70
PS C:\Users\Chaos Zhou\PyCharmProjects\STA219>
行 46, 列 35   空格: 4   UTF-8   CRLF   {} Python   3.12.7
```