STA510 Assignment 01 Solution

B M Nafis Fuad

2024-09-29

PROBLEM 01:

a)

Total possible outcome of three dices $= 6^3 = 216$

Combinations for the sum of three dices is 09
$= (6,1,2), (6,2,1), (5,1,3), (5,2,2), (5,3,1), (4,1,4), (4,2,3), (4,3,2),$
$(4,4,1), (3,1,5), (3,2,4), (3,3,3), (3,4,2), (3,5,1), (2,1,6), (2,2,5), (2,3,4),$
$(2,4,3), (2,5,2), (2,6,1), (1,2,6), (1,3,5), (1,4,4), (1,5,3), (1,6,2)$

Total 25 combination. So, $P(\text{Sum}=9) = 25/216$

Combinations for the sum of three dices is 10
$= (6,1,3), (6,2,2), (6,3,1), (5,1,4), (5,2,3), (5,3,2), (5,4,1), (4,1,5), (4,2,4),$
$(4,3,3), (4,4,2), (4,5,1), (3,1,6), (3,2,5), (3,3,4), (3,4,3), (3,5,2), (3,6,1),$
$(2,2,6), (2,3,5), (2,4,4), (2,5,3), (2,6,2), (1,3,6), (1,4,5), (1,5,4), (1,6,3)$

Total 27 combinations. So, $P(\text{Sum}=10) = 27/216$

b)

Probability of at least one six out of 4 dice throws ——

Probability of not getting a six in a single dice throw $= 5/6$
Probability of not getting a six on any four dice throws $= (5/6)^4$

So, $P(\text{at least one six}) = 1 - (5/6)^4 = 0.5177$

Probability of at least one time two sixes when throwing two dices 24 times —

Probability of getting one six in a single dice = 1/6

Probability of getting two six on a roll of two dice $= \frac{1}{6} \times \frac{1}{6} = 1/36$

We can use binomial distribution. The random variable X represents the number of sixes in 24 dice rolls. We have to find $P(X \geq 2)$, which is the probability of getting at least two sixes

Here, Total number of trials, $n = 24$

Probability of getting two six on a roll of two dice, $p = 1/36$

$X \sim \text{Binomial}(24, 1/36)$

$$f(x) = \binom{n}{x} p^x (1-p)^{n-x}, \quad x = 0, 1, \cdots n, \quad \left\| \binom{n}{x} = \frac{n!}{x!\,(n-x)!} \right.$$

Now, $P(X \geq 2) = 1 - \left(P(X=0) + P(X=1)\right)$

Probability of getting zero six in two dice in 24 rolls —

$$P(X=0) = \binom{24}{0} \left(\frac{1}{36}\right)^0 \left(\frac{35}{36}\right)^{24} = 1 \times 1 \times \left(\frac{35}{36}\right)^{24} = 0.5085$$

Probability of getting one six in two dice in 24 rolls —

$$P(X=1) = \binom{24}{1} \left(\frac{1}{36}\right)^1 \left(\frac{35}{36}\right)^{23} = \frac{24!}{1! \cdot 23!} \times \frac{1}{36} \times \left(\frac{35}{36}\right)^{23}$$

$$= \frac{24}{36} \times \left(\frac{35}{36}\right)^{23} = 0.3488$$

So, $P(X \geq 2) = 1 - \left(P(X=0) + P(X=1)\right) = 1 - 0.5085 - 0.3488 = 0.1427$

Thus, it is more likely to get at least one six out of 4 dice throws

c

At least one 6 in 6 dice throws, $X \sim \text{Binomial}(6, 1/6)$

$$P(X \geq 1) = 1 - P(X = 0)$$

$$P(X=0) = \binom{6}{0}\left(\frac{1}{6}\right)^0\left(\frac{5}{6}\right)^6 = \left(\frac{5}{6}\right)^6$$

So, $P(X \geq 1) = 1 - \left(\frac{5}{6}\right)^6 = 0.6651$

At least two 6 in 12 dice throws, $X \sim \text{Binomial}(12, 1/6)$

$$P(X \geq 2) = 1 - (P(X=0) + P(X=1))$$

$$P(X=0) = \binom{12}{0}\cdot\left(\frac{1}{6}\right)^0\cdot\left(\frac{5}{6}\right)^{12} = \left(\frac{5}{6}\right)^{12}$$

$$P(X=1) = \binom{12}{1}\cdot\left(\frac{1}{6}\right)^1\cdot\left(\frac{5}{6}\right)^{11} = 2\times\left(\frac{5}{6}\right)^{11}$$

So, $P(X \geq 2) = 1 - \left(\frac{5}{6}\right)^{12} - \left(2\times\left(\frac{5}{6}\right)^{11}\right) = 0.6187$

At least three 6 in 18 dice throws, $X \sim \text{Binomial}(18, 1/6)$

$$P(X \geq 3) = 1 - (P(X=0) + P(X=1) + P(X=2))$$

$$P(X=0) = \binom{18}{0}\left(\frac{1}{6}\right)^0\left(\frac{5}{6}\right)^{18} = \left(\frac{5}{6}\right)^{18}$$

$$P(X=1) = \binom{18}{1}\left(\frac{1}{6}\right)^1\left(\frac{5}{6}\right)^{17} = 3\cdot\left(\frac{5}{6}\right)^{17}$$

$$P(X=2) = \binom{18}{2}\left(\frac{1}{6}\right)^2\left(\frac{5}{6}\right)^{16} = \frac{18\times17}{2\times36}\times\left(\frac{5}{6}\right)^{16}$$

So, $P(X \geq 3) = 1 - \left(\frac{5}{6}\right)^{18} - \left(3\times\left(\frac{5}{6}\right)^{17}\right) - \left(17/4\times\left(\frac{5}{6}\right)^{16}\right)$

$= 0.5973$

d. Here, we use negative binomial distribution.

$$f(x) = \binom{x-1}{k-1} p^k (1-p)^{x-k}, \quad x = k, k+1, \cdots$$

Number of trials, $x = 8$

Number of success (three sixes) $= 3$

Probability of success (rolling a six) $= \frac{1}{6}$

So, $P(X=8) = \binom{7}{2} \left(\frac{1}{6}\right)^3 \left(\frac{5}{6}\right)^5 = \frac{7!}{2!\,5!} \cdot \left(\frac{1}{6}\right)^3 \cdot \left(\frac{5}{6}\right)^5 = 0.039$

Problem 01 (a) || R

```
n_sim <- 10000 # 10000 Simulations
set.seed(123)  # For reproducibility

# Simulate 10000 trials of rolling 3 dice and summing their results
three_dice_rolls <-
  replicate(n_sim, sum(sample(1:6, 3, replace = TRUE)))

# Calculate the probabilities
prob_sum_9 <- mean(three_dice_rolls == 9)
prob_sum_10 <- mean(three_dice_rolls == 10)

# Output
cat("Probability of Sum being 9: ", prob_sum_9,
    "& Sum being 10: ", prob_sum_10, "\n")
```

```
## Probability of Sum being 9:  0.1161 & Sum being 10:  0.1226
```

Problem 01 (b) || R

```
# At least one six in 4 rolls
one_six_in_4_rolls <-
  replicate(n_sim, any(sample(1:6, 4, replace = TRUE) == 6))
prob_one_six_in_4_rolls <- mean(one_six_in_4_rolls)

# At least two sixes in 24 trials of 2 dice
two_sixes_in_24_trials <-
  replicate(n_sim, sum(replicate(24, sum(sample(1:6, 2, replace = TRUE)) == 12)) >= 2)
prob_two_sixes_in_24_trials <- mean(two_sixes_in_24_trials)

# Output
cat("Probability of at least one six in 4 dice rolls: ",
    prob_one_six_in_4_rolls, "\n")
```

```
## Probability of at least one six in 4 dice rolls:  0.5136
```

```
cat("Probability of at least two sixes in 24 dice rolls: ",
    prob_two_sixes_in_24_trials, "\n")
```

```
## Probability of at least two sixes in 24 dice rolls:  0.1466
```

Problem 01 (c) || R

```
# At least one six in 6 rolls
at_least_one_six_in_6_rolls <-
  replicate(n_sim, sum(sample(1:6, 6, replace = TRUE) == 6) >= 1)
prob_at_least_one_six <- mean(at_least_one_six_in_6_rolls)

# At least two sixes in 12 rolls
at_least_two_sixes_in_12_rolls <-
  replicate(n_sim, sum(sample(1:6, 12, replace = TRUE) == 6) >= 2)
prob_at_least_two_sixes <- mean(at_least_two_sixes_in_12_rolls)

# At least three sixes in 18 rolls
```

```r
at_least_three_sixes_in_18_rolls <-
  replicate(n_sim, sum(sample(1:6, 18, replace = TRUE) == 6) >= 3)
prob_at_least_three_sixes <- mean(at_least_three_sixes_in_18_rolls)

# Output
cat("Probability of at least one six in 6 rolls: ",
    prob_at_least_one_six, "\n")
```

```
## Probability of at least one six in 6 rolls:  0.6656
```

```r
cat("Probability of at least two sixes in 12 rolls: ",
    prob_at_least_two_sixes, "\n")
```

```
## Probability of at least two sixes in 12 rolls:  0.6209
```

```r
cat("Probability of at least three sixes in 18 rolls: ",
    prob_at_least_three_sixes, "\n")
```

```
## Probability of at least three sixes in 18 rolls:  0.59
```

Problem 01 (d) || R

```r
# Simulate exactly 3 sixes in 8 rolls
three_sixes_in_8_rolls <- replicate(n_sim, {
  rolls <- sample(1:6, 8, replace = TRUE)
  sum(rolls == 6) == 3
})

prob_three_sixes_in_8_rolls <- mean(three_sixes_in_8_rolls)

# Output
cat("Probability of exactly 3 sixes in 8 rolls: ",
    prob_three_sixes_in_8_rolls, "\n")
```

```
## Probability of exactly 3 sixes in 8 rolls:  0.1069
```

Problem 02 || Theory

PROBLEM 02:

Given CDF, $f_a(t) = e^{-e^{-(t-a)}}$, $t \in \mathbb{R}$

a)

To verify that a given function is a proper CDF, it must satisfy following conditions —

01. As $t \to -\infty$, $f_a(t) \to 0$ & as $t \to +\infty$, $f_a(t) \to 1$
02. A CDF must be a non-decreasing function, meaning it never decreases as t increases
03. The CDF must satisfy $0 \leq f_a(t) \leq 1$ for all t.

Verify Condition 01 —

$$f_a(t) = e^{-e^{-(t-a)}} = e^{-e^{a-t}}$$

As $t \to -\infty$, $e^{a-t} \to +\infty$ [As e^{a-t} becomes $e^{+\infty}$]
So, $e^{-e^{a-t}} \to e^{-\infty}$ which is 0.
Thus, as $t \to -\infty$, $f_a(t) \to 0$

Again, As $t \to +\infty$, $e^{a-t} \to 0$ [As e^{a-t} becomes $e^{-\infty}$]
So, $e^{-e^{a-t}} \to e^{0}$ which is 1.
Thus, as $t \to +\infty$, $f_a(t) \to 1$

Verify Condition 02 —

To check if the function is non-decreasing, we check if the derivative of $f_a(t)$ is non-negative.

$$\frac{d}{dt} f_a(t) = \frac{d}{dt} e^{-e^{-(t-a)}}$$

$$= e^{-e^{-(t-a)}} \cdot \frac{d}{dt}\left(-e^{-(t-a)}\right)$$

$$= e^{-e^{-(t-a)}} \cdot \left(-e^{-(t-a)}\right) \cdot \frac{d}{dt}\left(-(t-a)\right)$$

$$= e^{-e^{-(t-a)}} \cdot \left(-e^{-(t-a)}\right) \cdot (-1)$$

$$= e^{-e^{-(t-a)}} \cdot e^{-(t-a)}$$

Since, both $e^{-e^{-(t-a)}}$ and $e^{-(t-a)}$ are positive for all t, the derivative is positive.

Thus, $f_a(t)$ is a non decreasing function.

Verify Condition 03 —

Since, $f_a(t)$ is non-decreasing and approaches 0 and 1 at the extremes, it must lie between 0 and 1 for all t.

As all the condition verifies, $f_a(t)$ is a PROPER CDF.

b/

To calculate the mean $E(T)$, we need to use the probability density function (PDF) $f_a(t)$, which is the derivative of the CDF $f_a(t)$ with respect to t.

So, $f_a(t) = \frac{d}{dt} f_a(t) = e^{-e^{-(t-a)}} \cdot e^{-(t-a)}$

Now, the mean $E(T)$ of a continuous random variable is given by

$$E(T) = \int_{-\infty}^{\infty} t \cdot f_a(t) \cdot dt$$

$$= \int_{-\infty}^{\infty} t \cdot e^{-(t-a)} \cdot e^{-e^{-(t-a)}} \cdot dt$$

Let, $s = e^{-(t-a)}$

$ds = -e^{-(t-a)}\,dt$

$dt = \frac{-ds}{s}$

Now, As $t \to -\infty, s \to \infty$

As $t \to \infty, s \to 0$

Again, $s = e^{-(t-a)}$

$\log s = \log(e^{-(t-a)})$

$\Rightarrow \log s = -(t-a)$

$\Rightarrow t = a - \log s$

So, $E(T) = \int_{\infty}^{0} (a - \log s) \cdot s \cdot e^{-s} \cdot \frac{-ds}{s}$

$= \int_{\infty}^{0} -\left((a - \log s) \cdot e^{-s} \cdot ds\right)$

$= \int_{0}^{\infty} (a - \log s) e^{-s}\, ds = \int_{0}^{\infty} (ae^{-s} - \log s \cdot e^{-s})\, ds$

Breaking the integral into two parts

$E(T) = a\int_{0}^{\infty} e^{-s} ds - \int_{0}^{\infty} \log s \cdot e^{-s}\, ds$

$= a[e^{-s}]_{0}^{\infty} + \gamma \quad [\text{As } \gamma = -\int_{0}^{\infty} \log(x) e^{-x} dx]$

$= a(-e^{-\infty} + e^{-0}) + \gamma$

$= a + \gamma$

c

$f_a(t) = e^{-e^{-(t-a)}}$, $t \in \mathbb{R}$

Given, $a = 5$

$P(T \le 4) = f_a(4) = e^{-e^{-(4-5)}} = e^{-e} = 0.0659$

$P(T > 9) = 1 - P(T \le 9) = 1 - f_a(9) = 1 - e^{-e^{-(9-5)}} = 1 - e^{-e^{-4}} = 1 - 0.9818 = 0.0182$

$P(5 < T \le 6) = P(T \le 6) - P(T \le 5) = f_a(6) - f_a(5) = e^{-e^{-(6-5)}} - e^{-e^{-(5-5)}} = e^{-e^{-1}} - e^{-e^{0}} = 0.3243$

d

let, $f_a(t) = U$, where $U \in (0,1)$ is the uniform random variable.

$$U = e^{-e^{-(t-a)}}$$

$$\Rightarrow \log U = \log\left(e^{-e^{-(t-a)}}\right)$$

$$\Rightarrow \log U = -e^{-(t-a)}$$

$$\Rightarrow -\log U = e^{-(t-a)}$$

$$\Rightarrow \log(-\log U) = \log e^{-(t-a)}$$

$$\Rightarrow \log(-\log U) = -t + a$$

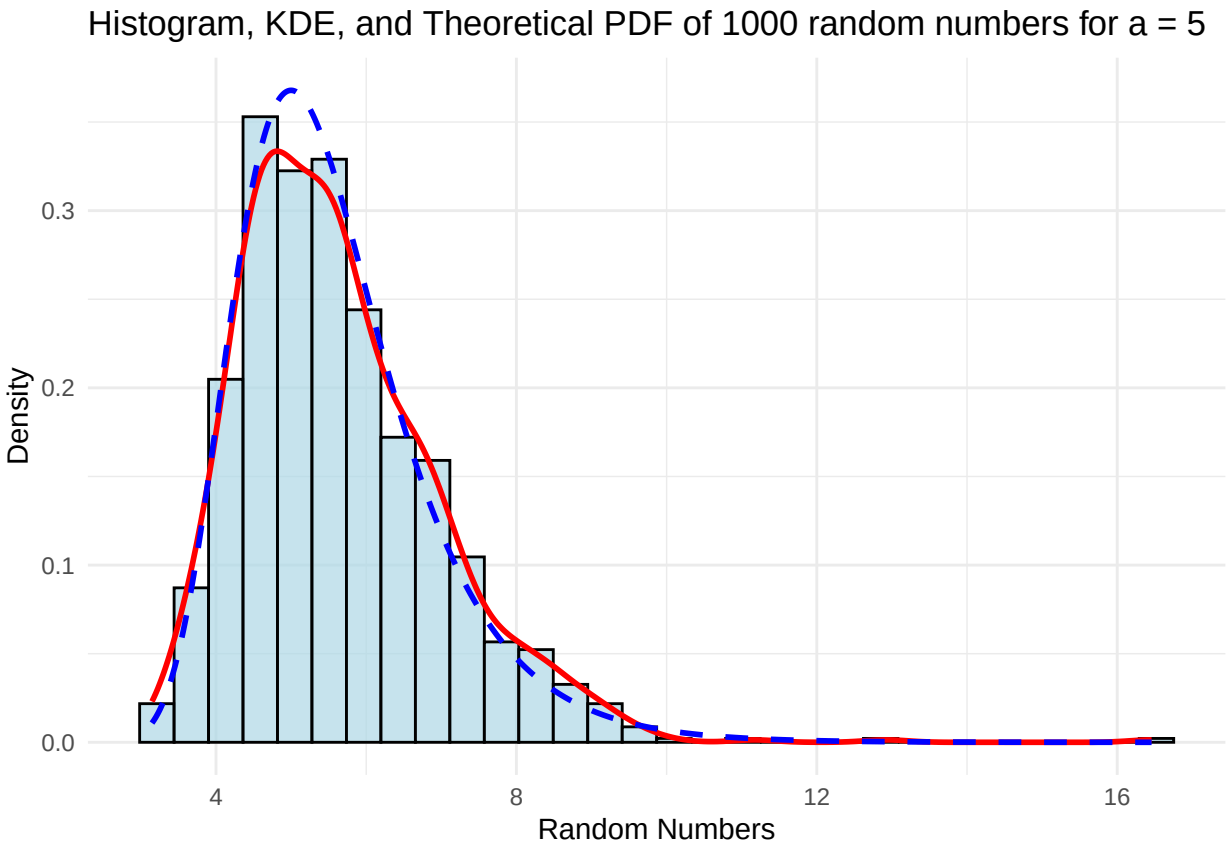
$$\Rightarrow t = a - \log(-\log U)$$

$$\Rightarrow t = a - \log(\log U^{-1}) \quad [\text{As } \log U^{-1} = -\log U]$$

Thus, the inverse CDF is $f_a^{-1}(U) = a - \log(\log U^{-1})$

Problem 02 (e) || R

```r
# Load necessary libraries
library(ggplot2)

# Define the inverse CDF function (using inverse transform sampling)
inverse_cdf <- function(n, a) {
  # Generate n uniform random numbers between 0 and 1
  U <- runif(n)
  # Use the inverse of the CDF to generate random numbers from the distribution
  T <- a - log(-log(U))
  return(T)
}

# Define the theoretical PDF function
theoretical_pdf <- function(t, a) {
  return(exp(-(t - a)) * exp(-exp(-(t - a))))
}

# Set parameters
n <- 1000
a <- 5

# Generate random numbers using the inverse CDF method
random_numbers <- inverse_cdf(n, a)

# Create a data frame for plotting
data <- data.frame(x = random_numbers)

# Generate theoretical PDF values for comparison
t_values <- seq(min(random_numbers), max(random_numbers), length.out = 1000)
pdf_values <- theoretical_pdf(t_values, a)

# Plot histogram, KDE, and theoretical PDF
ggplot(data, aes(x)) +
  # Histogram with density scaling using after_stat(density)
  geom_histogram(aes(y = after_stat(density)),
                 bins = 30, fill = "lightblue", color = "black", alpha = 0.7) +
  # KDE line using linewidth instead of size
  geom_density(color = "red", linewidth = 1) +
  # Theoretical PDF line using linewidth instead of size
  geom_line(aes(x = t_values, y = pdf_values),
            color = "blue", linetype = "dashed", linewidth = 1) +
  ggtitle(paste("Histogram, KDE, and Theoretical PDF of",
                n, "random numbers for a =", a)) +
  xlab("Random Numbers") +
  ylab("Density") +
  theme_minimal()
```



Problem 02 (f) || R

```r
# Set parameters
gamma <- 0.5772  # Euler's constant

# Calculate the sample mean from the generated random numbers
sample_mean <- mean(random_numbers)

# Calculate the theoretical mean E(T) = a + gamma
theoretical_mean <- a + gamma

# Output
cat("Sample Mean: ", sample_mean, "& Theoretical Mean: ", theoretical_mean, "\n")
```

```
## Sample Mean:  5.633345 & Theoretical Mean:  5.5772
```

```r
# Calculate and display the difference between the sample mean and the theoretical mean
difference <- abs(sample_mean - theoretical_mean)
cat("Difference between Sample Mean and Theoretical Mean: ", difference, "\n")
```

```
## Difference between Sample Mean and Theoretical Mean:  0.05614543
```

Problem 02 (g) || R

```r
# Calculate the sample variance
sample_variance <- var(random_numbers)
```

```r
sample_sd <- sqrt(sample_variance)

# Define the desired precision
desired_precision <- 0.01

# Calculate the number of samples needed for the desired precision
required_n <- (sample_sd / desired_precision)^2

# Output
cat("Sample Variance: ", sample_variance,
    "& Sample Standard Deviation: ", sample_sd, "\n")
```

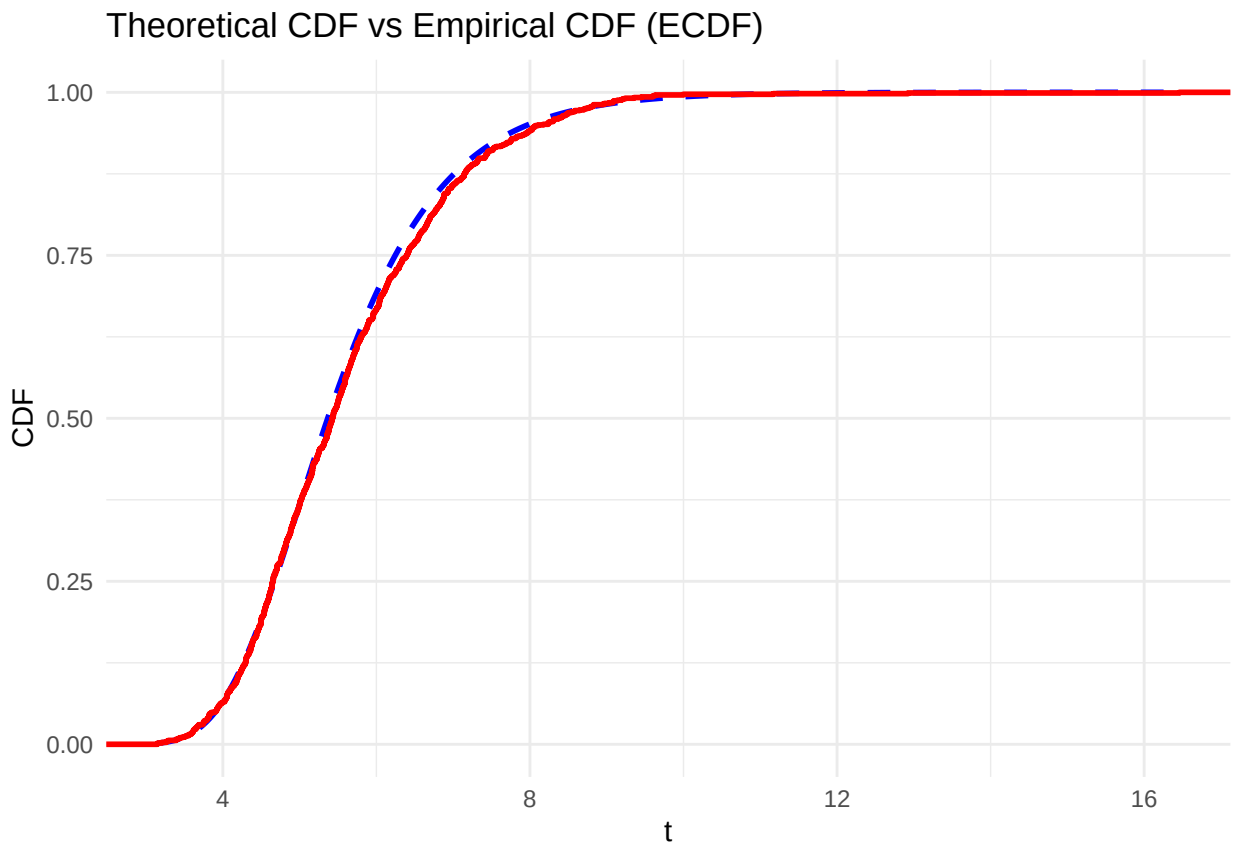
```
## Sample Variance:  1.798769 & Sample Standard Deviation:  1.341182
```

```r
cat("Required Sample Size to Achieve Precision of 0.01: ",
    ceiling(required_n), "\n")
```

```
## Required Sample Size to Achieve Precision of 0.01:  17988
```

Problem 02 (h) || R

```r
# Define the theoretical CDF function
theoretical_cdf <- function(t, a) {
  return(exp(-exp(-(t - a))))
}

# Calculate the ECDF using the random numbers
ecdf_func <- ecdf(random_numbers)

# Generate theoretical CDF values for comparison
t_values <- seq(min(random_numbers), max(random_numbers), length.out = 1000)
cdf_values <- theoretical_cdf(t_values, a)

# Plot the theoretical CDF and the ECDF
ggplot() +
  # Plot the theoretical CDF
  geom_line(aes(x = t_values, y = cdf_values),
            color = "blue", linewidth = 1, linetype = "dashed", show.legend = TRUE) +
  # Plot the ECDF
  stat_ecdf(aes(x = random_numbers),
            color = "red", linewidth = 1, show.legend = TRUE) +
  ggtitle("Theoretical CDF vs Empirical CDF (ECDF)") +
  xlab("t") +
  ylab("CDF") +
  theme_minimal() +
  labs(color = "") +
  scale_color_manual(values = c("Theoretical CDF" = "blue", "ECDF" = "red")) +
  theme(legend.position = "right")
```




```
# Comments on Finding
# The theoretical CDF is the exact function, based on the formula for Fa(t), plotted as
# a smooth curve. And The ECDF is based on the 1000 simulated observations and is
# represented as a step function.

# The empirical CDF closely follow the theoretical CDF. Any small deviations between the
# two curves are due to randomness in the sample. As the sample size increases, the
# empirical CDF will converge to the theoretical CDF.
```

Problem 03 || Theory

PROBLEM 03:

Given, $f_\lambda(x) = \frac{1}{2\lambda} \exp\left(-\frac{|x|}{\lambda}\right)$, $x \in \mathbb{R}, \lambda > 0$

where $|x|$ is the absolute value of x with $E(|x|) = \lambda$

a//

The likelihood function $L(\lambda)$ is the product of the individual densities for the observations ——

$$L(\lambda | x_1, x_2, \cdots x_n) = \prod_{i=1}^{n} f_\lambda(x_i)$$

$$= \prod_{i=1}^{n} \frac{1}{2\lambda} \exp\left(-\frac{|x_i|}{\lambda}\right)$$

$$= \left(\frac{1}{2\lambda}\right)^n \prod_{i=1}^{n} \exp\left(-\frac{|x_i|}{\lambda}\right)$$

$$= \left(\frac{1}{2\lambda}\right)^n \exp\left(\frac{-1}{\lambda} \sum_{i=1}^{n} |x_i|\right)$$

$$\log L(\lambda | x_1, x_2, \cdots x_n) = \log\left(\left(\frac{1}{2\lambda}\right)^n \exp\left(\frac{-1}{\lambda} \sum_{i=1}^{n} |x_i|\right)\right)$$

$$= \log\left(\frac{1}{2\lambda}\right)^n + \log\left(\exp\left(\frac{-1}{\lambda} \sum_{i=1}^{n} |x_i|\right)\right) \left[\log(ab) = \log a + \log b\right]$$

$$= n \log \frac{1}{2\lambda} + \left(\frac{-1}{\lambda} \sum_{i=1}^{n} |x_i|\right) \left[\log a^n = n \log a ; \log(\exp(a)) = a\right]$$

$$= -n \log 2\lambda - \frac{1}{\lambda} \sum_{i=1}^{n} |x_i| \left[\log \frac{a}{b} = -\log \frac{b}{a}\right]$$

To find the MLE of λ, we take the derivative of the log likelihood function with respect to λ and set it equal to zero.

$$\frac{d}{d\lambda} \log L(\lambda | x_1, x_2, \cdots, x_n) = \frac{d}{d\lambda} \left(-n \log 2\lambda - \frac{1}{\lambda} \sum_{i=1}^{n} |x_i|\right)$$

$$= \frac{d}{d\lambda}(-n\log 2\lambda) - \frac{d}{d\lambda}\left(\frac{1}{\lambda}\sum_{i=1}^{n}|x_i|\right)$$

$$= \frac{-n}{2\lambda}\frac{d}{d\lambda}(2\lambda) - \left(-\frac{1}{\lambda^2}\sum_{i=1}^{n}|x_i|\right) \quad \left[\frac{d}{dx}\log x = \frac{1}{x}\cdot\frac{d}{dx}x\right]$$

$$= \frac{-n}{\lambda} + \frac{1}{\lambda^2}\sum_{i=1}^{n}|x_i|$$

Set the derivative equal to zero,

$$\frac{-n}{\lambda} + \frac{1}{\lambda^2}\sum_{i=1}^{n}|x_i| = 0$$

$$\Rightarrow -n\lambda + \sum_{i=1}^{n}|x_i| = 0 \quad [\text{multiply by } \lambda^2 \text{ on both side}]$$

$$\Rightarrow \lambda = \frac{1}{n}\sum_{i=1}^{n}|x_i|$$

So, the maximum likelihood estimator (MLE) of λ is

$$\hat{\lambda} = \frac{1}{n}\sum_{i=1}^{n}|x_i|$$

Now, an estimator $\hat{\lambda}$ is unbiased if $E(\hat{\lambda}) = \lambda$

Given, $E(|x|) = \lambda$

So, $E(\hat{\lambda}) = E\left(\frac{1}{n}\sum_{i=1}^{n}|x_i|\right)$

$$= \frac{1}{n}\sum_{i=1}^{n}E(|x_i|)$$

$$= \frac{1}{n}\sum_{i=1}^{n}\lambda = \frac{1}{n}\cdot n\lambda = \lambda$$

Thus, the estimator $\hat{\lambda}$ is an unbiased estimator of λ

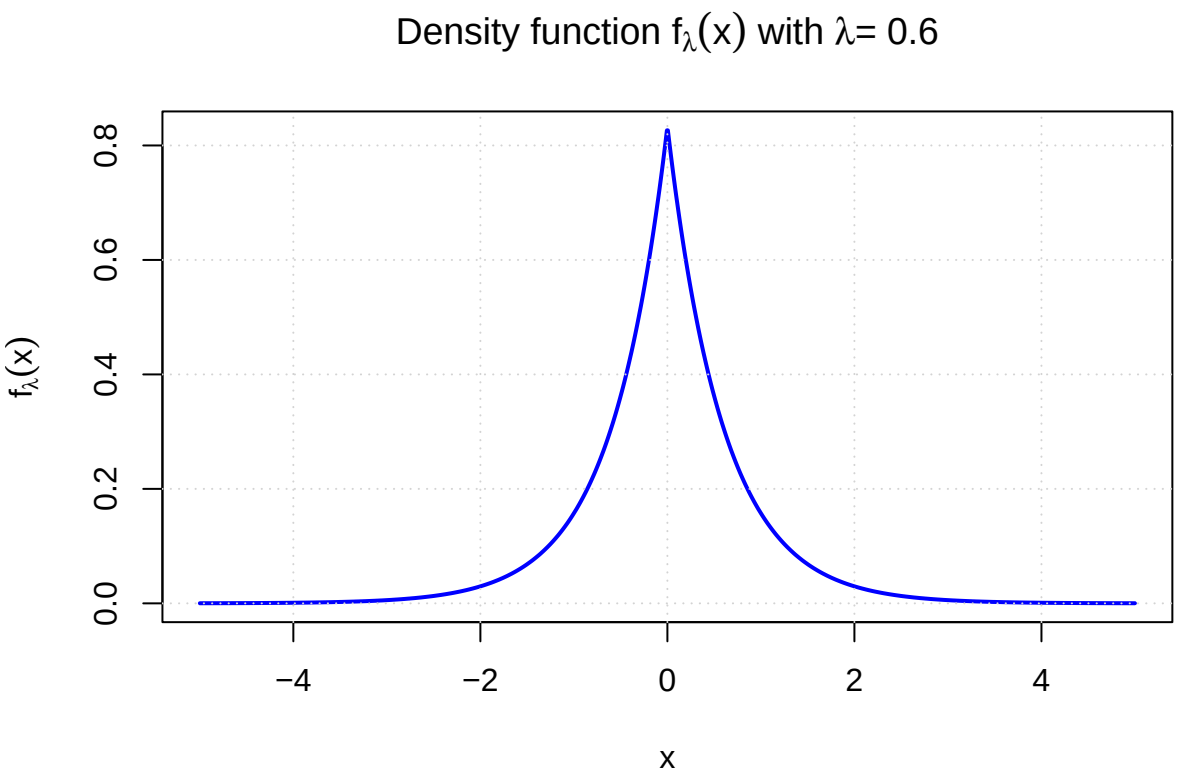
Problem 03 (b) || R

```r
# Sample data
x <- c(1.3, -0.6, 0.2, 0.4, -0.8)

# Calculate MLE for lambda
lambda_hat <- mean(abs(x))

# Output
cat("MLE: ", lambda_hat, "\n")
```

```
## MLE:  0.66
```

Problem 03 (c) || R

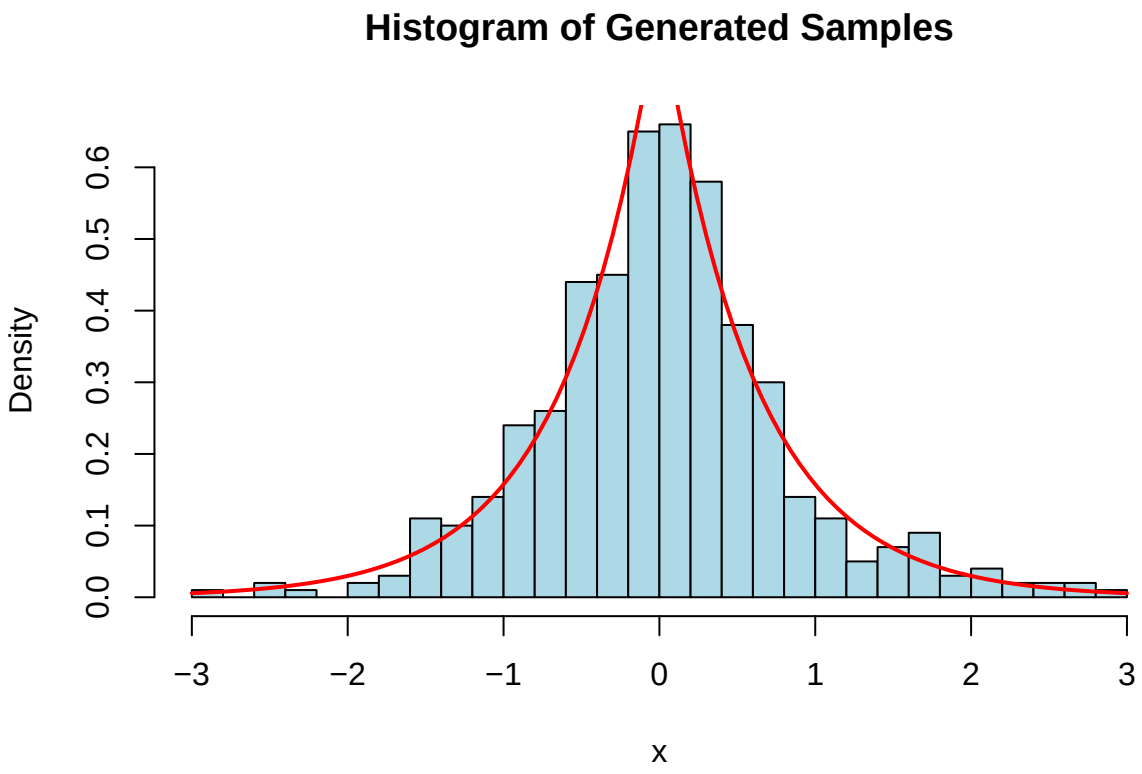
```r
# Define the density function for f_lambda(x)
f_lambda <- function(x, lambda) {
  (1 / (2 * lambda)) * exp(-abs(x) / lambda)
}

# Set lambda = 0.6
lambda <- 0.6

# Generate x values from -5 to 5
x_values <- seq(-5, 5, length.out = 1000)

# Compute f_lambda(x) for the x values
y_values <- f_lambda(x_values, lambda)

# Plot the density function
plot(x_values, y_values, type = "l", col = "blue", lwd = 2,
     xlab = "x", ylab = expression(f[lambda](x)),
     main = expression(paste("Density function ",
                             f[lambda](x), " with ", lambda, "= 0.6")))
grid()
```



Problem 03 (d) || R

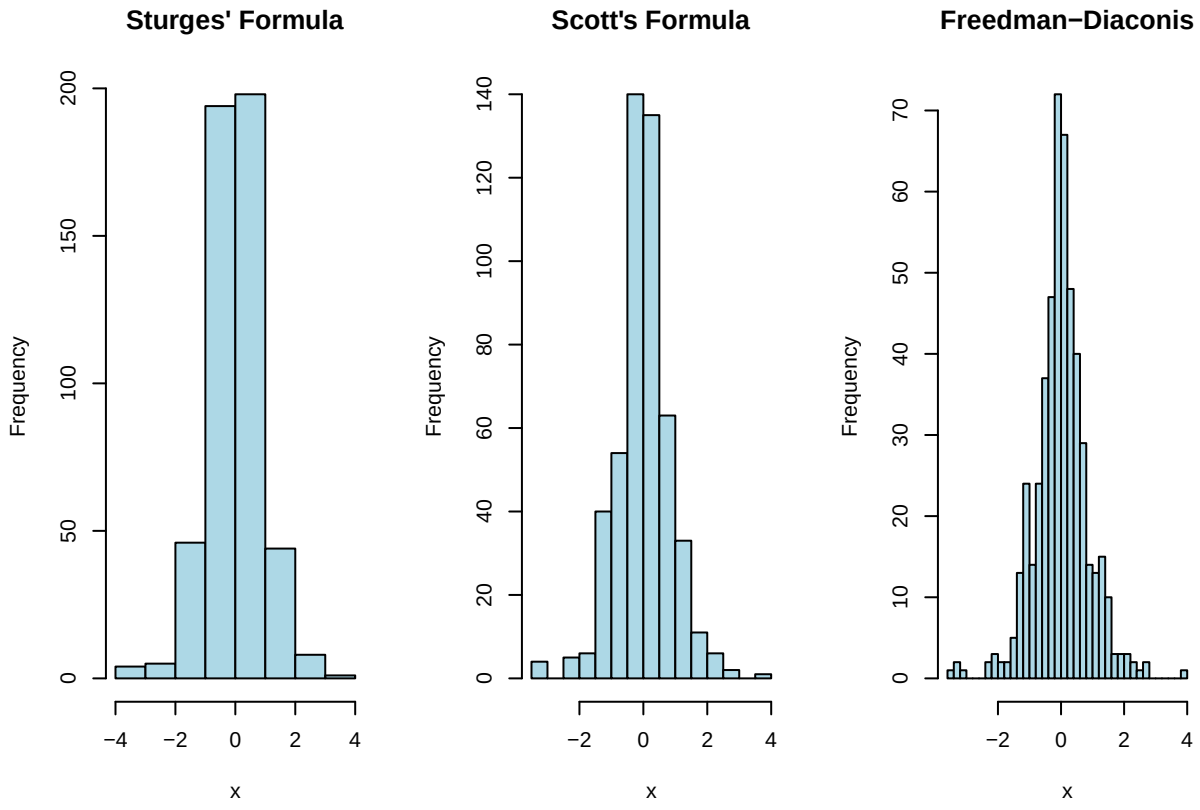
```r
# Define the target density function f_lambda(x)
f_lambda <- function(x, lambda) {
  (1 / (2 * lambda)) * exp(-abs(x) / lambda)
}

# Proposal distribution g(x) is uniform(-5, 5), with density 1/10
g <- function(x) {
  if (x >= -5 && x <= 5) {
    return(1 / 10)
  } else {
    return(0)
  }
}

# Accept-reject sampling function
accept_reject <- function(n, lambda) {
  samples <- numeric(n)  # Vector to store the samples
  count <- 0             # Counter for accepted samples
  M <- max(sapply
           (seq(-5, 5, length.out = 1000), function(x) f_lambda(x, lambda) / g(x)))

  while (count < n) {
    # Generate a candidate from the proposal distribution g(x)
```

```r
    x_candidate <- runif(1, -5, 5)

    # Generate a uniform random number for acceptance check
    u <- runif(1)

    # Check acceptance condition
    if (u < f_lambda(x_candidate, lambda) / (M * g(x_candidate))) {
      samples[count + 1] <- x_candidate
      count <- count + 1
    }
  }

  return(samples)
}

# Set lambda = 0.6 and generate n = 500 random numbers
lambda <- 0.6
n <- 500
random_samples <- accept_reject(n, lambda)

# Plot the histogram and KDE of the generated samples
hist(random_samples, prob = TRUE, breaks = 30, main = "Histogram of Generated Samples",
     xlab = "x", col = "lightblue", border = "black")

# Add the true density function to the plot
curve(f_lambda(x, lambda), add = TRUE, col = "red", lwd = 2)
```



Problem 03 (e) || R

```r
# Load necessary libraries
library(MASS)  # For bandwidth selection

# Generate 500 random samples using accept-reject method
lambda <- 0.6
n <- 500
random_samples <- accept_reject(n, lambda)

# Plot Histogram using different bin-width rules
hist_sturges <- hist(random_samples, breaks = "Sturges", plot = FALSE)
hist_scott <- hist(random_samples, breaks = "Scott", plot = FALSE)
hist_fd <- hist(random_samples, breaks = "FD", plot = FALSE)

# Plot histograms with different bin-width rules
par(mfrow = c(1, 3))  # Plot side by side
hist(random_samples, breaks = "Sturges",
     main = "Sturges' Formula", xlab = "x", col = "lightblue", border = "black")
hist(random_samples, breaks = "Scott",
     main = "Scott's Formula", xlab = "x", col = "lightblue", border = "black")
hist(random_samples, breaks = "FD",
     main = "Freedman-Diaconis", xlab = "x", col = "lightblue", border = "black")
```

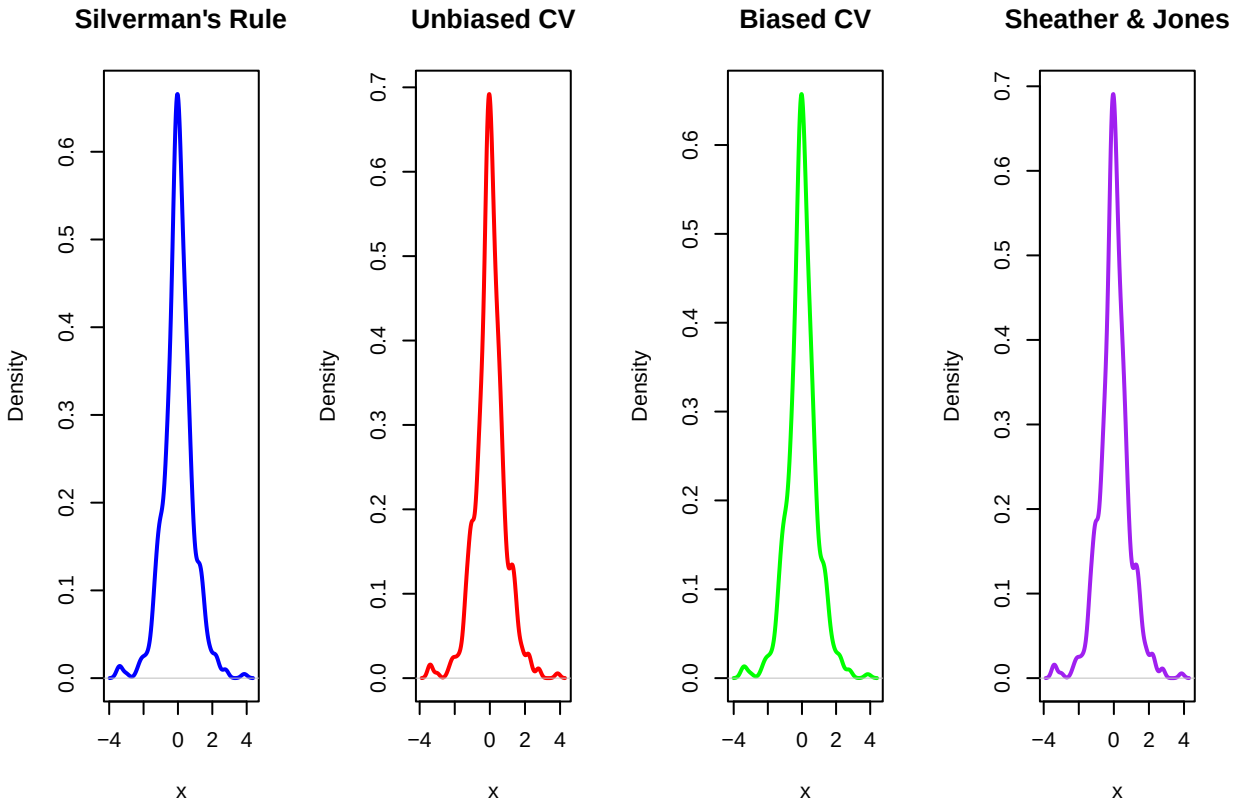



```r
par(mfrow = c(1, 1))  # Reset to single plot

# Calculate and print selected bin-width for each rule
cat("Sturges' bin-width:", diff(hist_sturges$breaks)[1],
    "|| Scott's bin-width:", diff(hist_scott$breaks)[1],
    "|| Freedman-Diaconis bin-width:", diff(hist_fd$breaks)[1], "\n")
```

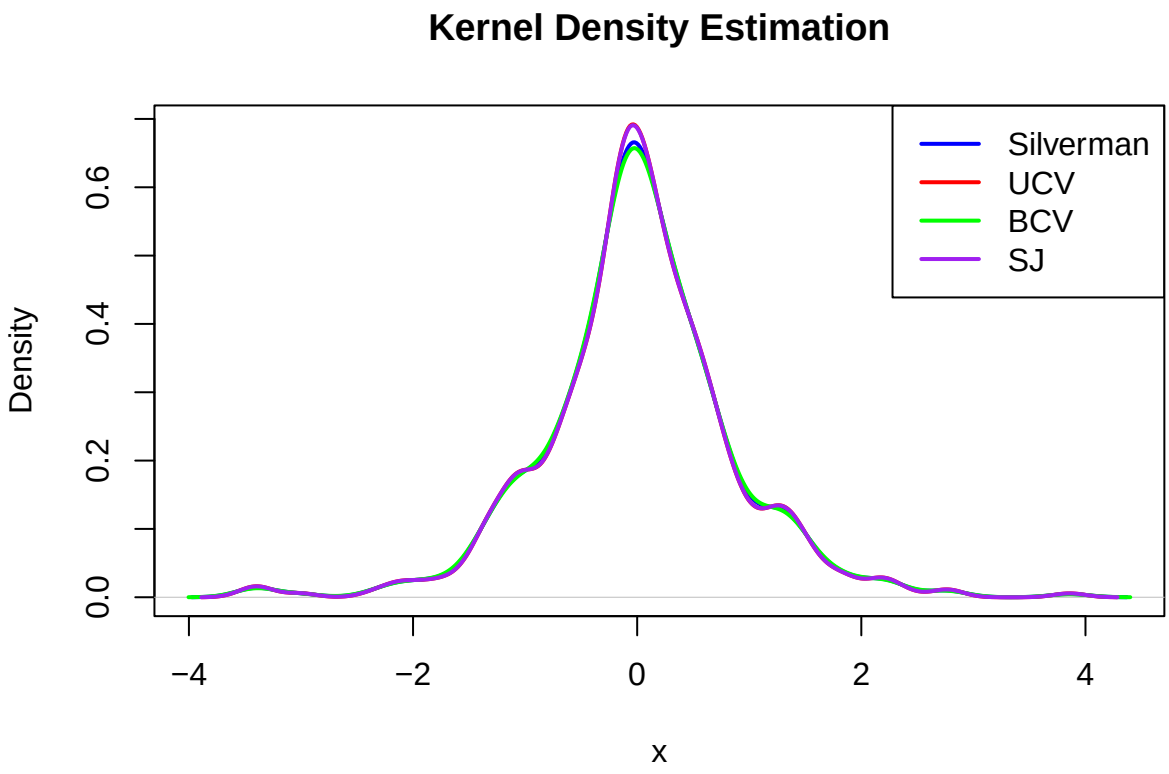
```
## Sturges' bin-width: 1 || Scott's bin-width: 0.5 || Freedman-Diaconis bin-width: 0.2
```

```r
# Comments on Findings
# Sturges' formula gives a broad, smooth representation but lacks detail.
# Scott's formula provides a good middle-ground between smoothness and detail.
# Freedman-Diaconis is the most sensitive and provides the most detailed view
# of the data, but it may overfit in some cases.

# KDE with different bandwidth selection methods
kde_silverman <- density(random_samples, bw = "nrd0")  # Silverman's Rule of Thumb
kde_ucv <- density(random_samples, bw = "ucv")  # Unbiased Cross Validation
kde_bcv <- density(random_samples, bw = "bcv")  # Biased Cross Validation
kde_sj <- density(random_samples, bw = "SJ")  # Sheather & Jones

# Plot each KDE separately in side-by-side plots
par(mfrow = c(1, 4))  # Set up the plotting area for 4 side-by-side plots

# Silverman's Rule of Thumb
plot(kde_silverman, main = "Silverman's Rule", col = "blue", lwd = 2,
     xlab = "x", ylab = "Density", ylim = c(0, max(kde_silverman$y)))
```

```r
# Unbiased Cross Validation (UCV)
plot(kde_ucv, main = "Unbiased CV", col = "red", lwd = 2,
     xlab = "x", ylab = "Density", ylim = c(0, max(kde_ucv$y)))

# Biased Cross Validation (BCV)
plot(kde_bcv, main = "Biased CV", col = "green", lwd = 2,
     xlab = "x", ylab = "Density", ylim = c(0, max(kde_bcv$y)))

# Sheather & Jones (SJ) Method
plot(kde_sj, main = "Sheather & Jones", col = "purple", lwd = 2,
     xlab = "x", ylab = "Density", ylim = c(0, max(kde_sj$y)))
```




```r
# Reset the plotting layout
par(mfrow = c(1, 1))

# Plot KDE with different bandwidth rules
plot(kde_silverman, main = "Kernel Density Estimation", col = "blue", lwd = 2,
     xlab = "x", ylab = "Density", ylim = c(0, max(kde_silverman$y, kde_ucv$y,
                                                   kde_bcv$y, kde_sj$y)))
lines(kde_ucv, col = "red", lwd = 2)
lines(kde_bcv, col = "green", lwd = 2)
lines(kde_sj, col = "purple", lwd = 2)
legend("topright", legend = c("Silverman", "UCV", "BCV", "SJ"),
       col = c("blue", "red", "green", "purple"), lwd = 2)
```




```
# Print bandwidths for different KDE methods
cat("Silverman's bandwidth:", kde_silverman$bw, "\n")
```

```
## Silverman's bandwidth: 0.1704621
```

```
cat("Unbiased Cross Validation bandwidth:", kde_ucv$bw, "\n")
```

```
## Unbiased Cross Validation bandwidth: 0.1411365
```

```
cat("Biased Cross Validation bandwidth:", kde_bcv$bw, "\n")
```

```
## Biased Cross Validation bandwidth: 0.1803577
```

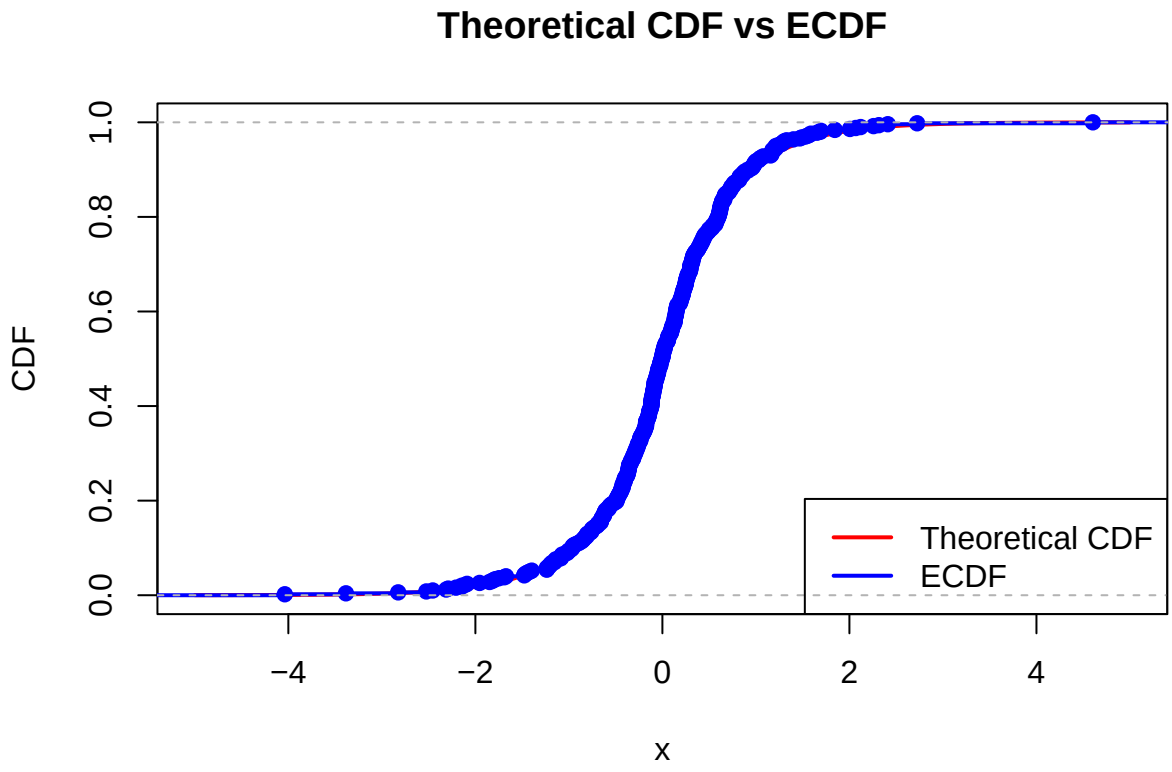
```
cat("Sheather & Jones bandwidth:", kde_sj$bw, "\n")
```

```
## Sheather & Jones bandwidth: 0.1425234
```

Problem 03 (f) || R

```
# Define the target CDF F_lambda(x)
F_lambda <- function(x, lambda) {
  0.5 + 0.5 * sign(x) * (1 - exp(-abs(x) / lambda))
}

# Generate random samples using accept-reject method
lambda <- 0.6
n <- 500
random_samples <- accept_reject(n, lambda)
```

```r
# Generate a sequence of x values to compute the theoretical CDF
x_values <- seq(-5, 5, length.out = 1000)
cdf_values <- F_lambda(x_values, lambda)

# Plot the theoretical CDF using ggplot2
plot(x_values, cdf_values, type = "l", col = "red", lwd = 2,
     main = "Theoretical CDF vs ECDF",
     xlab = "x", ylab = "CDF", ylim = c(0, 1))

# Add the ECDF based on simulated data
ecdf_plot <- ecdf(random_samples)
lines(ecdf_plot, col = "blue", lwd = 2)

# Add legend to differentiate the CDF and ECDF
legend("bottomright", legend = c("Theoretical CDF", "ECDF"),
       col = c("red", "blue"), lwd = 2)
```




```r
# Comments on Finding
# The close alignment between the theoretical CDF and the empirical ECDF suggests that
# the simulated data accurately follows the Laplace distribution with the given parameter
# lambda = 0.6. While small deviations can occur due to sampling variability, the overall
# match indicates that the sampling method is effective and the data represents the
# theoretical model well. Increasing sample size would further improve this alignment.
```

Problem 04 (a)

```
# Function to simulate the number of trials until a winner when players have k lives
simulate_rps_trials <- function(k) {
  player1_lives <- k
  player2_lives <- k
  trials <- 0

  while (player1_lives > 0 && player2_lives > 0) {
    trials <- trials + 1
    outcome <- sample(c("win", "tie"), 1, prob = c(2/3, 1/3))

    if (outcome == "win") {
      # Randomly decide which player wins
      winner <- sample(c("player1", "player2"), 1)
      if (winner == "player1") {
        player2_lives <- player2_lives - 1
      } else {
        player1_lives <- player1_lives - 1
      }
    }
  }

  return(trials)
}
```

Problem 04 (b)

```
# Compare simulation results with geometric distribution (k = 1)
simulate_geometric <- function(p, n) {
  rgeom(n, prob = p) + 1
}

k <- 1
n_sims <- 100000
sim_trials <- replicate(n_sims, simulate_rps_trials(k))

# Theoretical geometric distribution
p <- 2/3
geom_dist <- simulate_geometric(p, n_sims)

# Compare the two distributions
sim_mean <- mean(sim_trials)
geom_mean <- mean(geom_dist)

cat("Mean number of trials from simulation: ", sim_mean, "\n")
```

```
## Mean number of trials from simulation:  1.50008
```

```
cat("Mean number of trials from geometric distribution: ", geom_mean, "\n")
```

```
## Mean number of trials from geometric distribution:  1.49773
```

Problem 04 (c)

```r
# Estimate for k = 3 and k = 5
estimate_trials <- function(k, n_sims = 100000) {
  sim_trials <- replicate(n_sims, simulate_rps_trials(k))
  mean_trials <- mean(sim_trials)
  prob_8_or_more <- mean(sim_trials >= 8)

  return(list(mean_trials = mean_trials,
              prob_8_or_more = prob_8_or_more, trials = sim_trials))
}

# Simulate for k = 3 and k = 5
k3_results <- estimate_trials(3)
k5_results <- estimate_trials(5)

# Print results
cat("Expected number of trials for k = 3: ", k3_results$mean_trials, "\n")
```

```
## Expected number of trials for k = 3:  6.18698
```

```r
cat("Probability of 8 or more trials for k = 3: ", k3_results$prob_8_or_more, "\n\n")
```

```
## Probability of 8 or more trials for k = 3:  0.23707
```

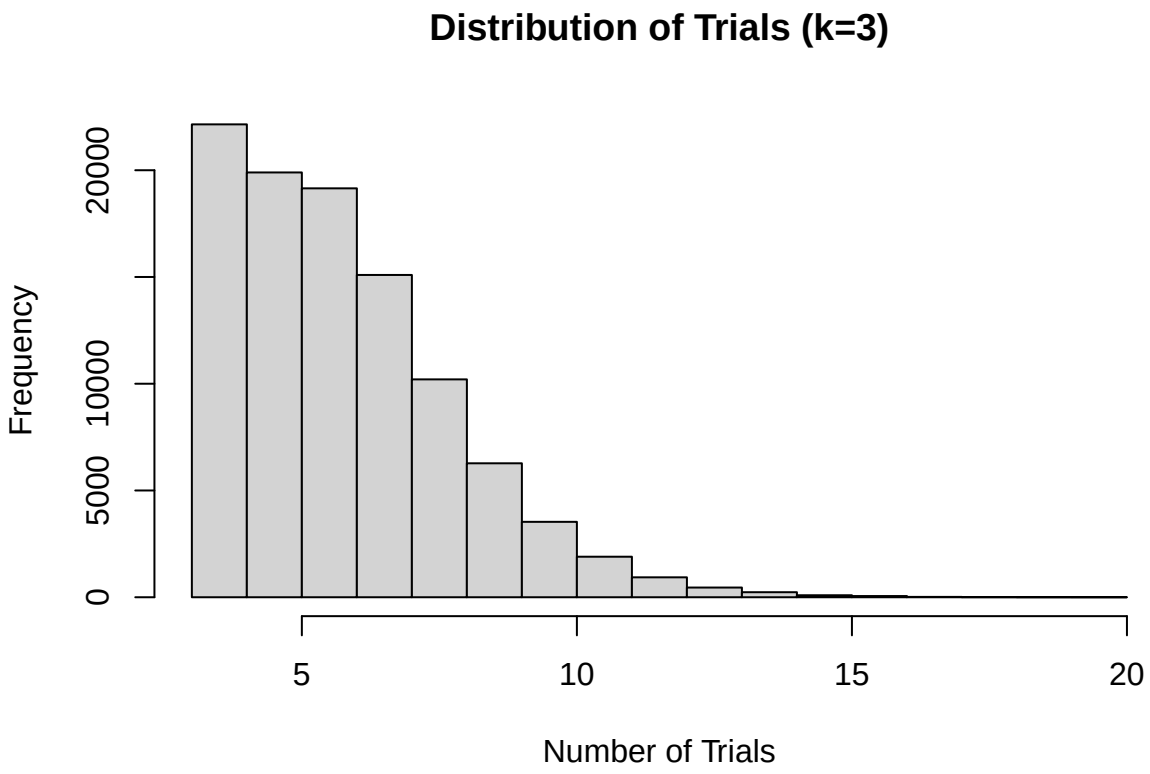
```r
cat("Expected number of trials for k = 5: ", k5_results$mean_trials, "\n")
```

```
## Expected number of trials for k = 5:  11.31925
```

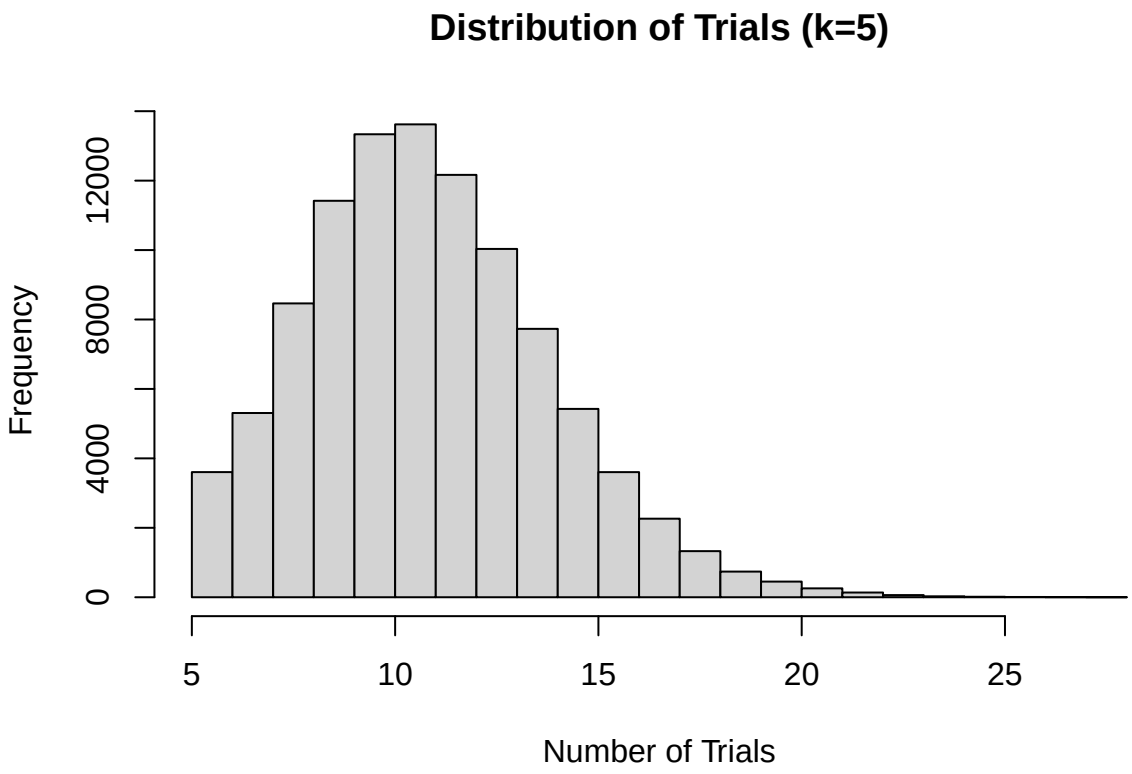
```r
cat("Probability of 8 or more trials for k = 5: ", k5_results$prob_8_or_more, "\n\n")
```

```
## Probability of 8 or more trials for k = 5:  0.91088
```

```r
# Plotting the distribution of Y3 and Y5
hist(k3_results$trials, breaks = 20,
     main = "Distribution of Trials (k=3)", xlab = "Number of Trials")
```




```r
hist(k5_results$trials, breaks = 20,
     main = "Distribution of Trials (k=5)", xlab = "Number of Trials")
```


Distribution of Trials (k=5)
Frequency
0
4000
8000
12000
5
10
15
20
25
Number of Trials