```r
# Downloading the gene_expression.tsv file
download.file("https://raw.githubusercontent.com/ghazkha/Assessment4/refs/heads/main/gene_expression.ts
              "gene_expression.tsv")

# Install and load required packages
if (!require("seqinr")) install.packages("seqinr")
```

```
## Loading required package: seqinr
```

```r
# Load seqinr package
library(seqinr)

# Read downloaded file
gene_expression <- read.table("gene_expression.tsv", header = TRUE, sep = "\t", row.names = 1)

# Display the first few rows of the gene expression data
head(gene_expression)
```

```
##                              GTEX.1117F.0226.SM.5GZZ7 GTEX.1117F.0426.SM.5EGHI
## ENSG00000223972.5_DDX11L1                           0                        0
## ENSG00000227232.5_WASH7P                          187                      109
## ENSG00000278267.1_MIR6859-1                         0                        0
## ENSG00000243485.5_MIR1302-2HG                       1                        0
## ENSG00000237613.2_FAM138A                           0                        0
## ENSG00000268020.3_OR4G4P                            0                        1
##                              GTEX.1117F.0526.SM.5EGHJ
## ENSG00000223972.5_DDX11L1                           0
## ENSG00000227232.5_WASH7P                          143
## ENSG00000278267.1_MIR6859-1                         1
## ENSG00000243485.5_MIR1302-2HG                       0
## ENSG00000237613.2_FAM138A                           0
## ENSG00000268020.3_OR4G4P                            0
```

```r
# Create a new column with the mean of the other columns
gene_expression$Mean <- rowMeans(gene_expression)

# Display the first few rows with the new Mean column
head(gene_expression)
```

```
##                              GTEX.1117F.0226.SM.5GZZ7 GTEX.1117F.0426.SM.5EGHI
## ENSG00000223972.5_DDX11L1                           0                        0
## ENSG00000227232.5_WASH7P                          187                      109
## ENSG00000278267.1_MIR6859-1                         0                        0
## ENSG00000243485.5_MIR1302-2HG                       1                        0
## ENSG00000237613.2_FAM138A                           0                        0
## ENSG00000268020.3_OR4G4P                            0                        1
##                              GTEX.1117F.0526.SM.5EGHJ        Mean
## ENSG00000223972.5_DDX11L1                           0   0.0000000
## ENSG00000227232.5_WASH7P                          143 146.3333333
## ENSG00000278267.1_MIR6859-1                         1   0.3333333
## ENSG00000243485.5_MIR1302-2HG                       0   0.3333333
## ENSG00000237613.2_FAM138A                           0   0.0000000
## ENSG00000268020.3_OR4G4P                            0   0.3333333
```

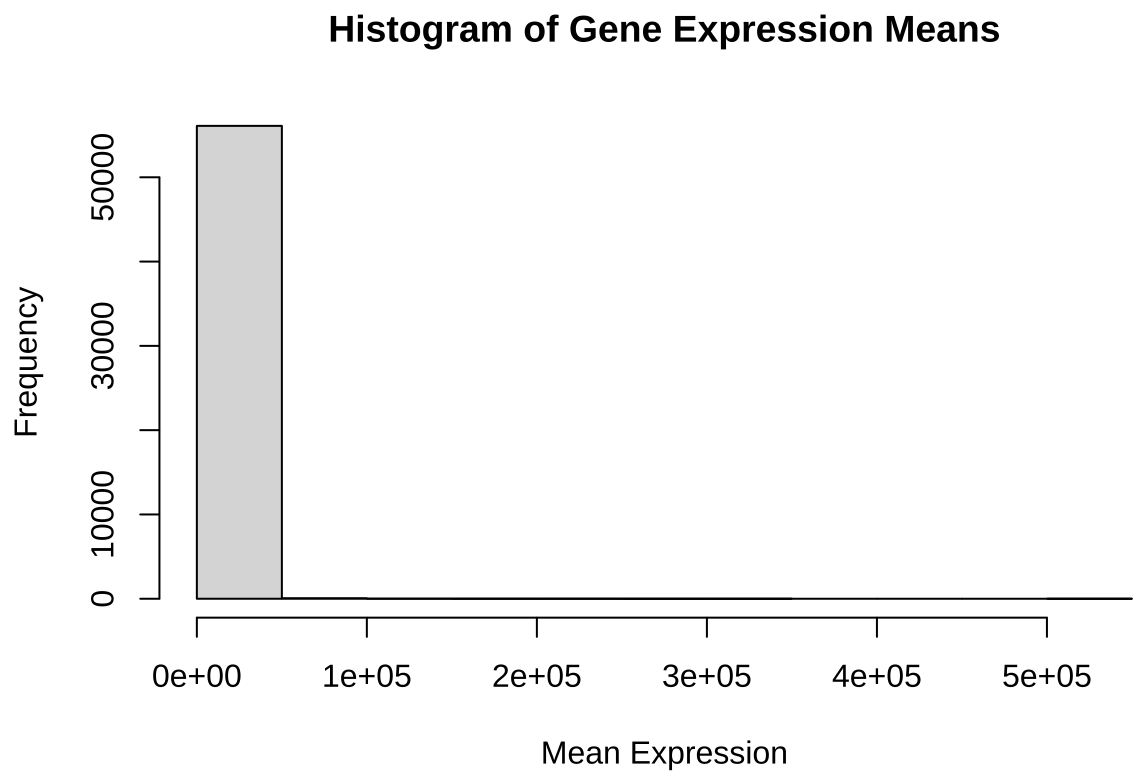
```r
# List the 10 genes with the highest mean expression
top_10_genes <- head(gene_expression[order(-gene_expression$Mean), ], 10)
```

```r
# Display the top 10 genes
top_10_genes
```

```
##                            GTEX.1117F.0226.SM.5GZZ7 GTEX.1117F.0426.SM.5EGHI
## ENSG00000198804.2_MT-CO1                     267250                  1101779
## ENSG00000198886.2_MT-ND4                     273188                   991891
## ENSG00000198938.2_MT-CO3                     250277                  1041376
## ENSG00000198888.2_MT-ND1                     243853                   772966
## ENSG00000198899.2_MT-ATP6                    141374                   696715
## ENSG00000198727.2_MT-CYB                     127194                   638209
## ENSG00000198763.3_MT-ND2                     159303                   543786
## ENSG00000211445.11_GPX3                      464959                    39396
## ENSG00000198712.1_MT-CO2                     128858                   545360
## ENSG00000156508.17_EEF1A1                    317642                    39573
##                            GTEX.1117F.0526.SM.5EGHJ     Mean
## ENSG00000198804.2_MT-CO1                     218923 529317.3
## ENSG00000198886.2_MT-ND4                     277628 514235.7
## ENSG00000198938.2_MT-CO3                     223178 504943.7
## ENSG00000198888.2_MT-ND1                     194032 403617.0
## ENSG00000198899.2_MT-ATP6                    151166 329751.7
## ENSG00000198727.2_MT-CYB                     141359 302254.0
## ENSG00000198763.3_MT-ND2                     149564 284217.7
## ENSG00000211445.11_GPX3                      306070 270141.7
## ENSG00000198712.1_MT-CO2                     122816 265678.0
## ENSG00000156508.17_EEF1A1                    339347 232187.3
```

```r
# Count the number of genes with a mean expression less than 10
low_mean_genes <- sum(gene_expression$Mean < 10)

# Display the number of genes with mean expression < 10
low_mean_genes
```

```
## [1] 35988
```



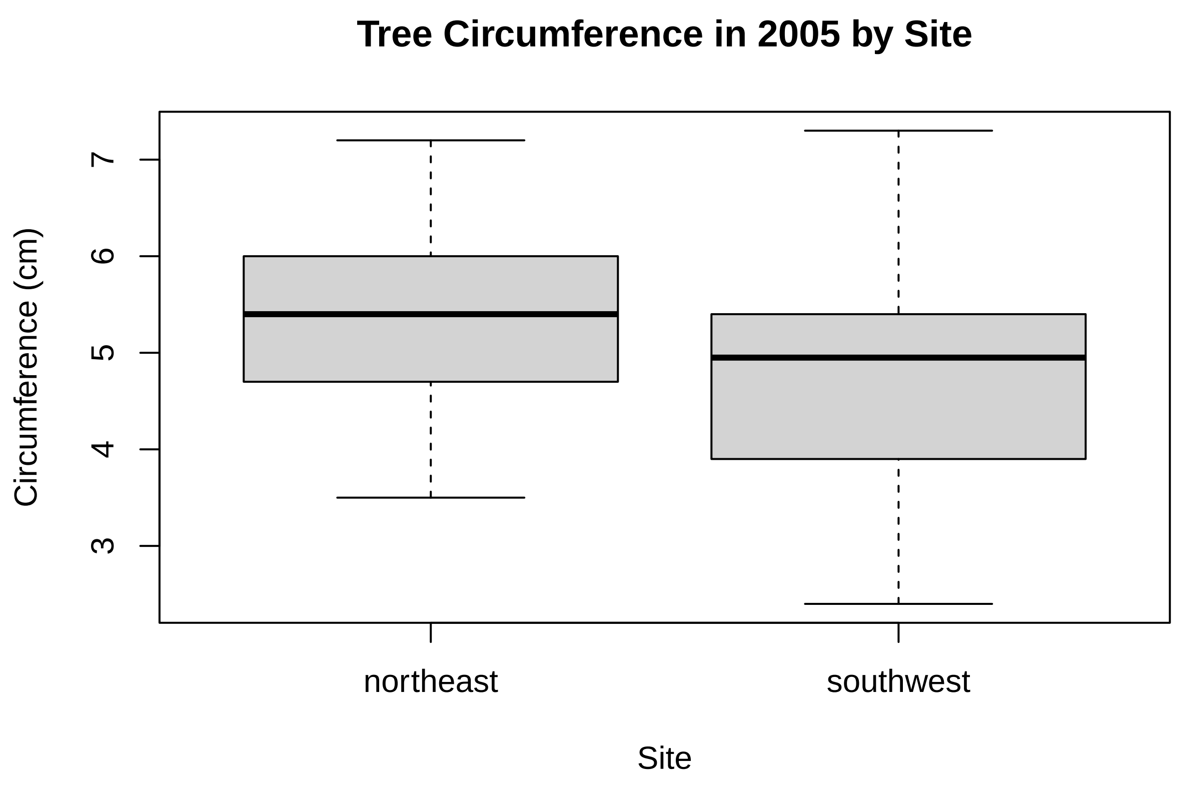
```r
# Downloading the growth_data.csv file
download.file("https://raw.githubusercontent.com/ghazkha/Assessment4/refs/heads/main/growth_data.csv",
              "growth_data.csv")

# Read and inspect the "growth_data.csv" file
growth_data <- read.csv("growth_data.csv")

# Display column names
colnames(growth_data)
```

```
## [1] "Site"            "TreeID"          "Circumf_2005_cm" "Circumf_2010_cm"
## [5] "Circumf_2015_cm" "Circumf_2020_cm"
```

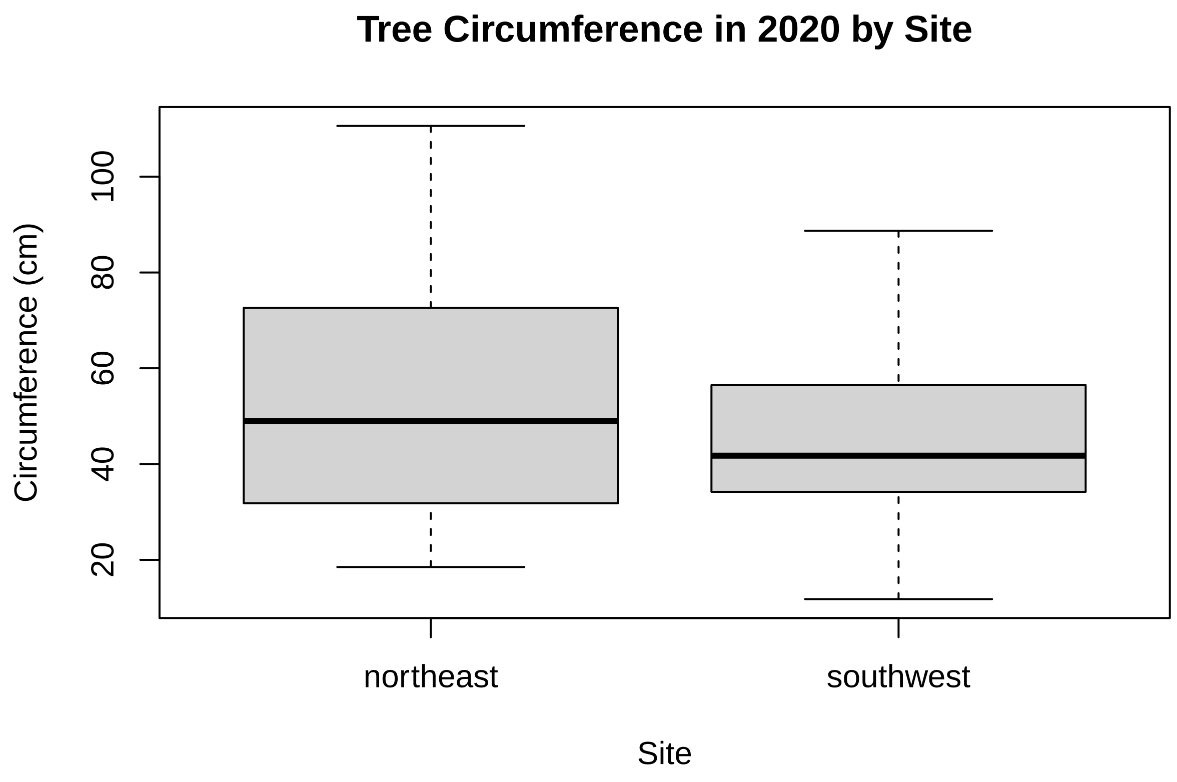
```r
# Calculate the mean and standard deviation of tree circumference at the start (2005) and end (2020) of
circumf_stats <- aggregate(cbind(Circumf_2005_cm, Circumf_2020_cm) ~ Site,
                           data = growth_data,
                           FUN = function(x) c(mean = mean(x), sd = sd(x)))
circumf_stats
```

```
##        Site Circumf_2005_cm.mean Circumf_2005_cm.sd Circumf_2020_cm.mean
## 1 northeast            5.2920000          0.9140267             54.22800
## 2 southwest            4.8620000          1.1474710             45.59600
##   Circumf_2020_cm.sd
## 1           25.22795
## 2           17.87345
```

```r
# Boxplot of tree circumference in 2005 by site
boxplot(Circumf_2005_cm ~ Site, data = growth_data,
        main = "Tree Circumference in 2005 by Site",
        xlab = "Site",
        ylab = "Circumference (cm)")
```



```
# Boxplot of tree circumference in 2020 by site
boxplot(Circumf_2020_cm ~ Site, data = growth_data,
        main = "Tree Circumference in 2020 by Site",
        xlab = "Site",
        ylab = "Circumference (cm)")
```



```
# Calculate mean growth over the last 10 years (2010-2020) at each site
growth_data$Growth_10_years <- growth_data$Circumf_2020_cm - growth_data$Circumf_2010_cm
mean_growth <- aggregate(Growth_10_years ~ Site, data = growth_data, mean)
mean_growth
```

```
##        Site Growth_10_years
## 1 northeast           42.94
## 2 southwest           35.49
```

```
# Perform a t-test to estimate if the 10-year growth differs between the two sites
t_test_growth <- t.test(Growth_10_years ~ Site, data = growth_data)
t_test_growth$p.value
```

```
## [1] 0.06229256
```

Part 2

```
# Part 2

# Install required packages
install.packages("R.utils")
```

```
## Installing package into '/home/s224409221/R/x86_64-pc-linux-gnu-library/4.1'
## (as 'lib' is unspecified)
```

```
# Load the necessary libraries
library(R.utils)
```

```
## Loading required package: R.oo

## Loading required package: R.methodsS3

## R.methodsS3 v1.8.2 (2022-06-13 22:00:14 UTC) successfully loaded. See ?R.methodsS3 for help.

## R.oo v1.26.0 (2024-01-24 05:12:50 UTC) successfully loaded. See ?R.oo for help.

##
## Attaching package: 'R.oo'

## The following object is masked from 'package:R.methodsS3':
##
##     throw

## The following object is masked from 'package:seqinr':
##
##     getName

## The following objects are masked from 'package:methods':
##
##     getClasses, getMethods

## The following objects are masked from 'package:base':
##
##     attach, detach, load, save

## R.utils v2.12.3 (2023-11-18 01:00:02 UTC) successfully loaded. See ?R.utils for help.

##
## Attaching package: 'R.utils'

## The following object is masked from 'package:utils':
##
##     timestamp

## The following objects are masked from 'package:base':
##
##     cat, commandArgs, getOption, isOpen, nullfile, parse, warnings
```

```
# URL for Saprospirales bacterium CDS
URL_saprospirales <- "https://ftp.ensemblgenomes.ebi.ac.uk/pub/bacteria/release-59/fasta/bacteria_58_col

# Download the file
download.file(URL_saprospirales, destfile = "saprospirales_cds.fa.gz")

# Read the FASTA file using seqinr's read.fasta
saprospirales_cds <- read.fasta(file = "saprospirales_cds.fa.gz", seqtype = "DNA")
saprospirales_cds_count <- length(saprospirales_cds)



# URL for E. coli CDS
URL_ecoli <- "http://ftp.ensemblgenomes.org/pub/bacteria/release-53/fasta/bacteria_0_collection/escheri

# Download the file
download.file(URL_ecoli, destfile = "ecoli_cds.fa.gz")

# Read the FASTA file for E. coli using seqinr's read.fasta
ecoli_cds <- read.fasta(file = "ecoli_cds.fa.gz", seqtype = "DNA")
```

```r
ecoli_cds_count <- length(ecoli_cds)



# Create a table comparing the number of coding sequences for both organisms
cds_table <- data.frame(
  Organism = c("E. coli", "Saprospirales"),
  Coding_Sequences = c(ecoli_cds_count, saprospirales_cds_count)
)

print(cds_table)
```

```
##        Organism Coding_Sequences
## 1       E. coli             4239
## 2 Saprospirales             4527
```

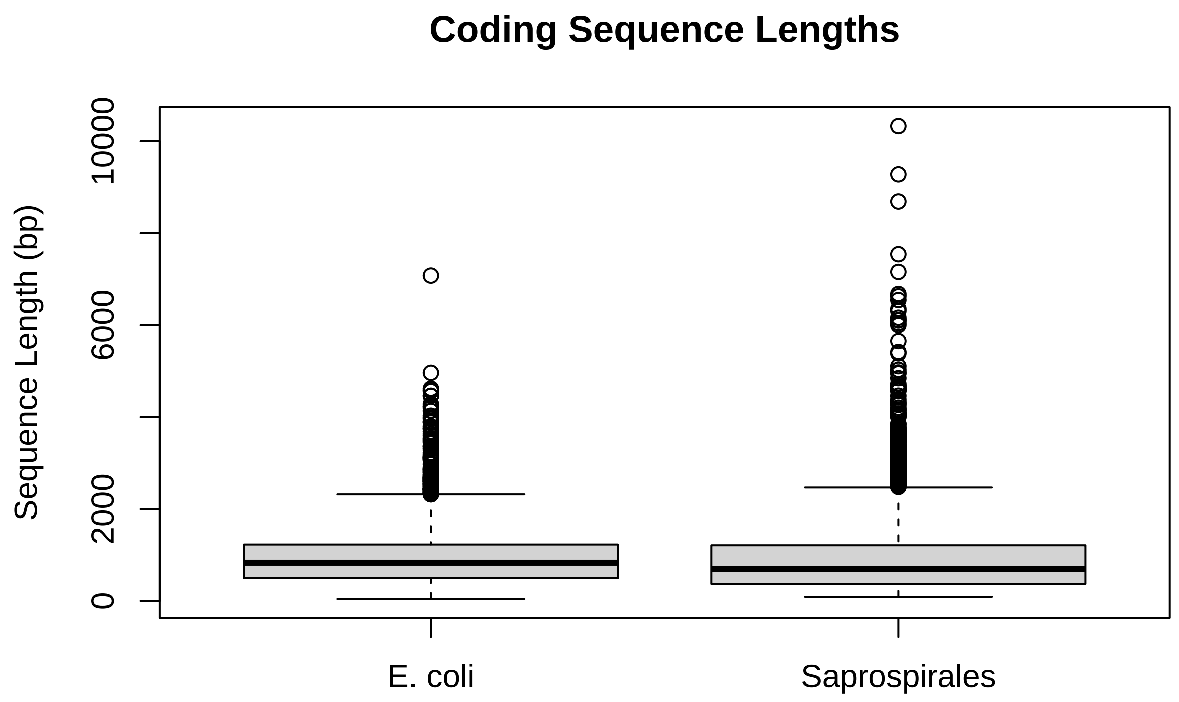
Difference between two organism:

Saprospirales bacterium* has a higher number of coding sequences (4,527) compared to *E. coli* (4,239), indicating a potentially more complex genome and greater functional diversity. This difference suggests that *Saprospirales bacterium* may possess specialized genes that enable it to adapt to specific environmental conditions or ecological niches, whereas *E. coli* has a more streamlined genome optimized for versatility in various laboratory and natural settings.

```r
# Calculate the total coding DNA length for both organisms
ecoli_total_coding_dna <- sum(sapply(ecoli_cds, length))
sapros_total_coding_dna <- sum(sapply(saprospirales_cds, length))

# Create a table comparing the total coding DNA lengths
total_coding_table <- data.frame(
  Organism = c("E. coli", "Saprospirales"),
  Total_Coding_DNA = c(ecoli_total_coding_dna, sapros_total_coding_dna)
)

print(total_coding_table)
```

```
##        Organism Total_Coding_DNA
## 1       E. coli          3978528
## 2 Saprospirales          4200321
```

```r
# Calculate sequence lengths for both organisms
ecoli_cds_lengths <- sapply(ecoli_cds, length)
sapros_cds_lengths <- sapply(saprospirales_cds, length)

# Boxplot of coding sequence lengths
boxplot(ecoli_cds_lengths, sapros_cds_lengths,
        names = c("E. coli", "Saprospirales"),
        main = "Coding Sequence Lengths",
        ylab = "Sequence Length (bp)")
```



```r
# Calculate the mean and median of sequence lengths for both organisms
mean_ecoli <- mean(ecoli_cds_lengths)
median_ecoli <- median(ecoli_cds_lengths)
mean_sapros <- mean(sapros_cds_lengths)
median_sapros <- median(sapros_cds_lengths)

cat("E. coli: Mean =", mean_ecoli, ", Median =", median_ecoli, "\n")
```

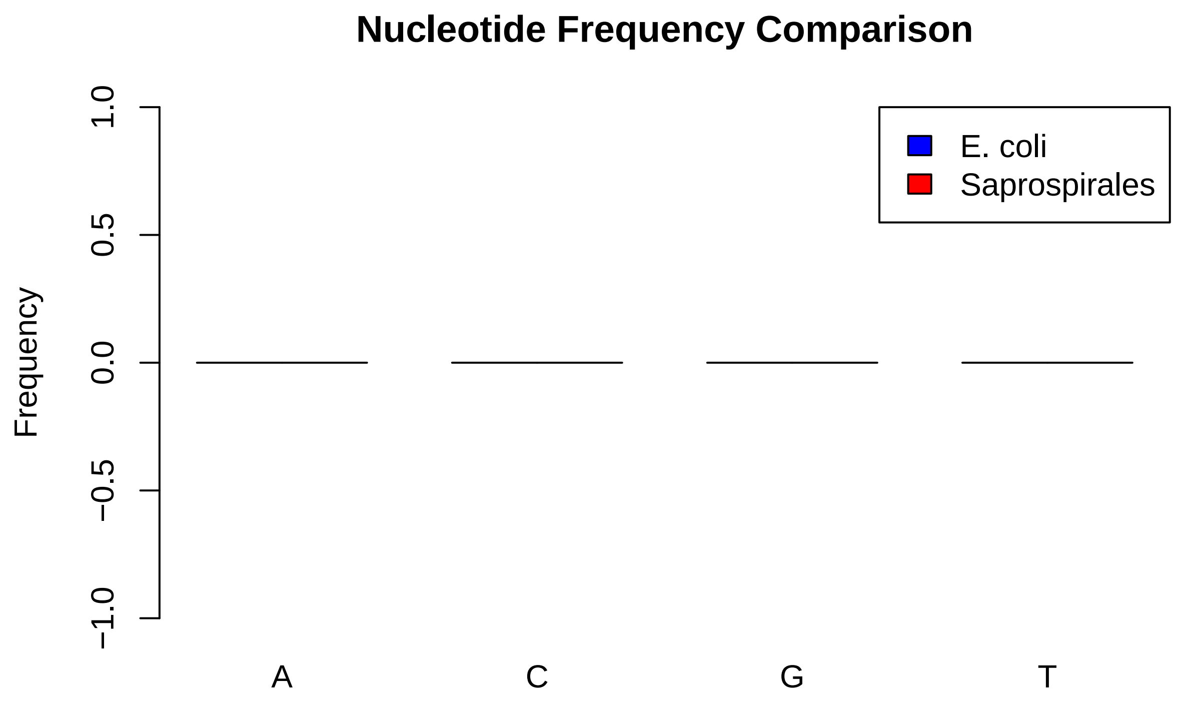
```
## E. coli: Mean = 938.5534 , Median = 831
```

```r
cat("Saprospirales: Mean =", mean_sapros, ", Median =", median_sapros, "\n")
```

```
## Saprospirales: Mean = 927.8376 , Median = 690
```

```r
# Define the standard nucleotides
nucleotides <- c("A", "C", "G", "T")

# Combine all E. coli sequences for nucleotide frequency calculation
ecoli_concat <- paste(unlist(ecoli_cds), collapse = "")  # Concatenate all sequences
ecoli_nuc_freq <- table(factor(strsplit(ecoli_concat, split = "")[[1]], levels = nucleotides))

# Combine all Saprospirales sequences for nucleotide frequency calculation
sapros_concat <- paste(unlist(saprospirales_cds), collapse = "")  # Concatenate all sequences
sapros_nuc_freq <- table(factor(strsplit(sapros_concat, split = "")[[1]], levels = nucleotides))

# Create a matrix for the bar plot
nuc_freq_matrix <- rbind(as.numeric(ecoli_nuc_freq), as.numeric(sapros_nuc_freq))
```

```r
# Create bar plots for nucleotide frequencies
barplot(nuc_freq_matrix, beside = TRUE,
        names.arg = nucleotides,
        legend.text = c("E. coli", "Saprospirales"),
        main = "Nucleotide Frequency Comparison",
        col = c("blue", "red"),
        ylab = "Frequency",
        args.legend = list(x = "topright"))
```



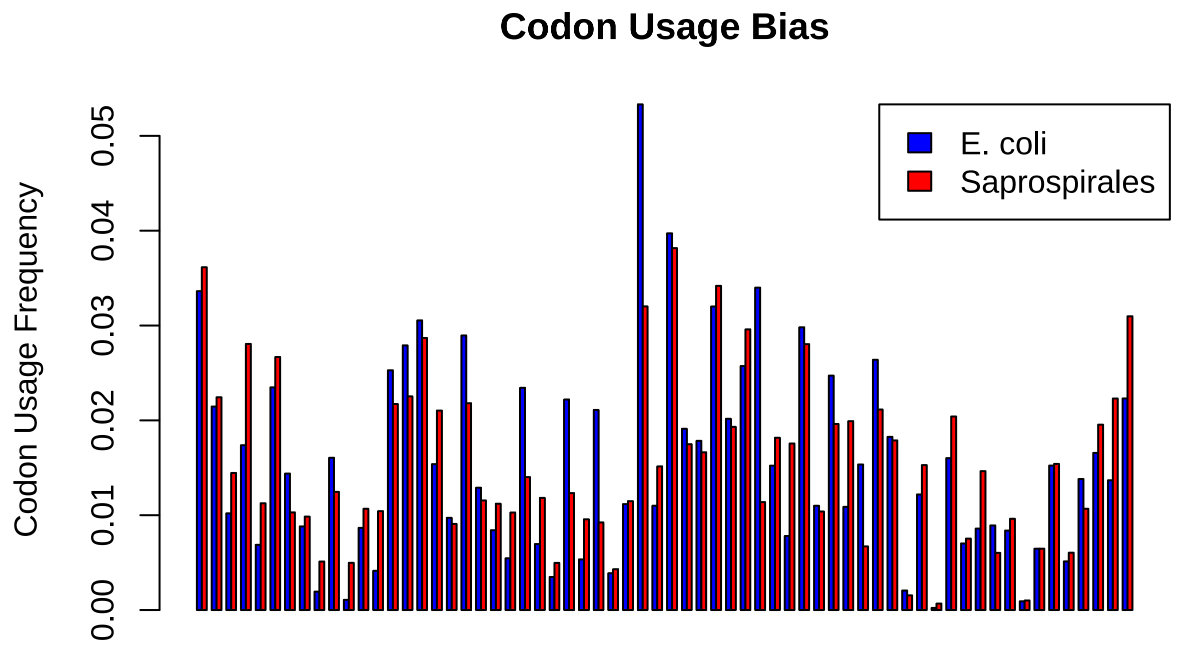
```r
# Calculate codon usage for E. coli
ecoli_codon_usage <- uco(unlist(ecoli_cds), frame = 0, index = "freq")

# Calculate codon usage for Saprospirales
sapros_codon_usage <- uco(unlist(saprospirales_cds), frame = 0, index = "freq")

# Check the structure of codon usage outputs
str(ecoli_codon_usage)
```

```
##  'table' num [1:64(1d)] 0.03362 0.02146 0.0102 0.01738 0.00687 ...
##  - attr(*, "dimnames")=List of 1
##   ..$ : chr [1:64] "aaa" "aac" "aag" "aat" ...
```

```r
str(sapros_codon_usage)
```

```
##  'table' num [1:64(1d)] 0.0361 0.0224 0.0145 0.0281 0.0113 ...
##  - attr(*, "dimnames")=List of 1
##   ..$ : chr [1:64] "aaa" "aac" "aag" "aat" ...
```

```r
# Extract the codon usage frequency values directly (uco might already be a vector)
ecoli_codon_usage_vector <- as.vector(ecoli_codon_usage)
sapros_codon_usage_vector <- as.vector(sapros_codon_usage)

# Ensure both vectors have the same length for comparison
if (length(ecoli_codon_usage_vector) == length(sapros_codon_usage_vector)) {

  # Create a matrix for codon usage comparison
  codon_usage_matrix <- rbind(ecoli_codon_usage_vector, sapros_codon_usage_vector)

  # Create a bar plot to compare codon usage bias
  barplot(codon_usage_matrix, beside = TRUE,
          main = "Codon Usage Bias",
          col = c("blue", "red"),
          ylab = "Codon Usage Frequency",
          legend.text = c("E. coli", "Saprospirales"),
          args.legend = list(x = "topright"))

} else {
  warning("Codon usage vectors for E. coli and Saprospirales have different lengths.")
}
```


Codon Usage Bias
Codon Usage Frequency
E. coli
Saprospirales