Assessment 4_SLE777_R Project

Ameeta

2025-10-05

Gene expression Step 1: Download file for geneexpression.tsv

The link for raw file were copied from the github and then downloaded in R-markdown using te download.file function and then the file was downloaded locally with the name geneexpression.tsv using destfile argument. The file was sucessfully downloaded.

```
# download the gene expression file

download.file("https://raw.githubusercontent.com/ghazkha/Assessment4/refs/heads/main/gene_expression.ts
```

Step 1: Reading gene expression file with gene identifiers as the row names and dispaying first 6 genes

The downloaded file was read using the read.table function, where the header argument specifies that the first row contain the name of the columns, while sep= "^ indicates that the file is in TSV format. The row.names =1 was used to set the first column as the names of gene identifiers, stringAsFactors=FALSE ensures that the data are represented as plain texts rather than factors. The obtained dataset was saved in the object gene_data. Following that, to inspect the contents, the head function was used display the first 6 rows, showing 6 gene identifiers.The value 6 was used to ensure onky 6 rows are displayed in the table. The first six rows containig 6 gene identifiers were obtained with 3 columns.

```
# Read the downloaded file
gene_data <- read.table("geneexpression.tsv",     # path to the file
                        header=TRUE,       # indicates first row of the file contains column names
                        sep = "\t",        # \t is the standard for tsv files
                        row.names = 1,    #indicates that first colum contains row names
                        stringsAsFactors = FALSE )  # keep as plain character strings

# display the first 6 genes of the file
head(gene_data, 6) # 6 inidcates number of rows to be displayed
```

```
##                                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
```

Step 2: Generating a new column with mean value of other columns and displaying 6 genes

A new column called meanexpression, which contain the mean value of other columns was created in the table using rowMeans function which calculate the mean of other columns for a given row. The output was saved in meanexpression cloumn under gene_data object. To display the output, the rows 1:6 values were selected to display first six genes and column 1 and ncol(last column) was selected to display the first and the last column (meanexpression column). The column for mean value of other columns were successfully generated.

```
gene_data$meanexpression <- rowMeans(gene_data) # rowMeans calculate means across all columns for each
gene_data[1:6, c(1, ncol(gene_data))] # 1:6 selects forst 6 genes of the data
```

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

```
                                    # c(1, ncol(gene_data) selects the first and the last column
```

Step 3: Listing the 10 genes with the highest mean expression

Firstly, the meanexpression in gene_data were ordered in the descending order using order function followed by negative (-) symbol just before the gene_data$meanexpression and then it was saved in gene_data_sorted file. After that, first 10 rows containing 10 genes with highest meanexpression were displayed in gene_data_sorted using a drop argument to ensure the datas are obtained in table format and not in vector format. The 10 genes with the highest mean expression was generated. The gene with the highest mean expression value was ENSG00000198804.2_MT-CO1 with mean value of 529317.3.

```
# order the "meanexpression" of the gene-data in descending order and save in gene_data-sorted file
gene_data_sorted <- gene_data[order(-gene_data$meanexpression), ] # order(-gene_data$meanofcolumns) sor
# show 10 genes with the highest mea expression values
gene_data_sorted[1:10, "meanexpression", drop = FALSE] # drop =FALSE ensures datas are expressed in dat
```

```
##                           meanexpression
## ENSG00000198804.2_MT-CO1       529317.3
## ENSG00000198886.2_MT-ND4       514235.7
## ENSG00000198938.2_MT-CO3       504943.7
## ENSG00000198888.2_MT-ND1       403617.0
## ENSG00000198899.2_MT-ATP6      329751.7
## ENSG00000198727.2_MT-CYB       302254.0
## ENSG00000198763.3_MT-ND2       284217.7
## ENSG00000211445.11_GPX3        270141.7
## ENSG00000198712.1_MT-CO2       265678.0
## ENSG00000156508.17_EEF1A1      232187.3
```

Step 4. Determining the number of genes with a mean <10

to determine the genes with the meanexpression less than 10, firstly the logical vector was created that tests each gene's mean expression value where the line checks for every gene in the dataset whethere its meanexpression value is less than 10. So, if the value is less than 10, the result is true and if it is not, the result is FALSE. The output was then saved in gene_data_mean_10. Then to check how many genes meet the condition of meanexpression < 10, the sum() function was used. As the TRUE is treated as 1 and FALSE as 0 in R, summing the logical vector gives the total number of genes with meanexpression value less than 10. A total of 35,988 genes contain the mean expression value less then 10.

```
# create logical vectors for genes with meanexpression <10
gene_data_mean_10 <- gene_data_sorted$meanexpression <10
# count the total number of genes with mean <10
```

```r
sum(gene_data_mean_10) # sum was used for summing all the logocal vectors
```

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

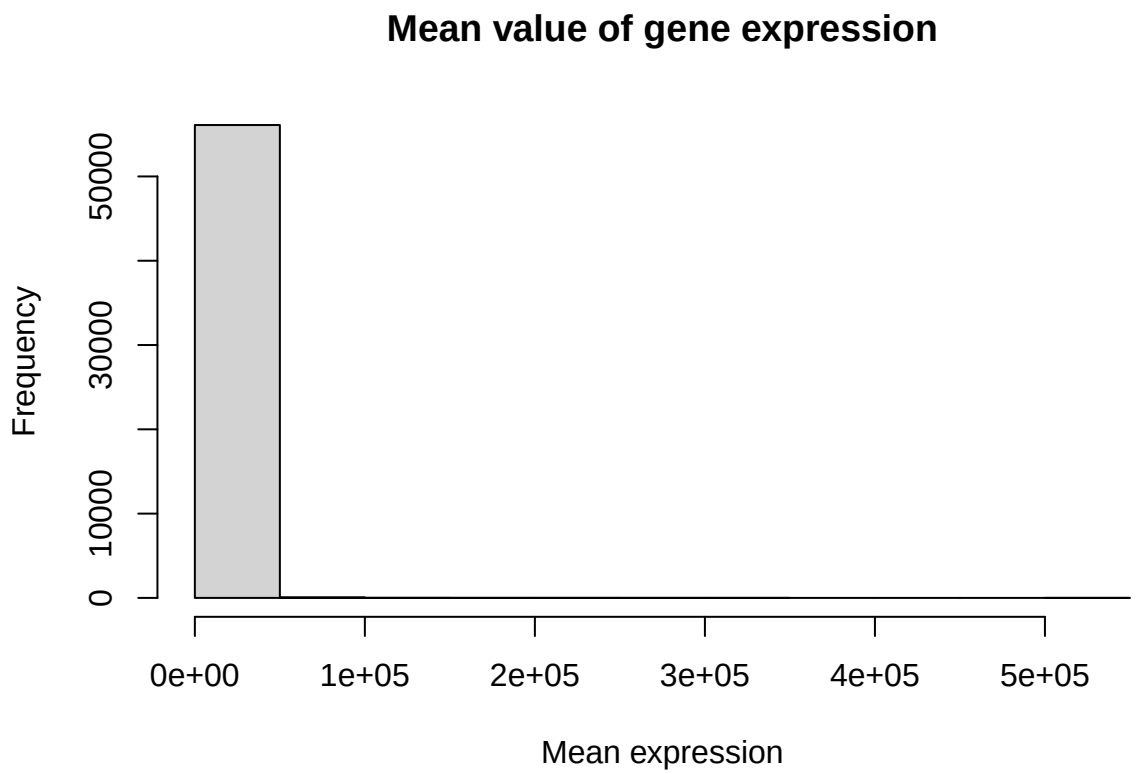
Step 5: Making a histogram plot of the mean values

A histogram was generated using hist() function to represent the distribution of the mean expression value of the genes. The hist() function takes the meanexpression column from the gene_data dataset and plot its frequency distribution, where the gene_data$expression specifies the data to plot, xlab="mean expression", add descriptive label to the x-axis and main="mean value of gene expression" add the title for the plot. As shown in the graph, the majority of the genes have a very low mean expression as indicated by the tallest bin near zero. However, there are a few genes which has extremely higher mean expression (5e+05) but appeared to be empty due to the dominance by the tall bar for low gene expression.

```r
# Step 6: Histogram of mean values of gene expression
data(gene_data)
```

```
## Warning in data(gene_data): data set 'gene_data' not found
```

```r
hist(gene_data$meanexpression, # data to be plotted
     xlab="Mean expression", # label for the x-axis
     main="Mean value of gene expression") # Main title of the histogram
```



GROWTH DATA INTERPRETATION

Step 6: Importing the csv file into an R object

The csv raw file for growth data was downloaded using download.file function and was saved locally as growth_data. then the file was imported into R as a data frame using the read.csv() function. This command reads the CSV file and then the file was stored in the growth_data object. Following that the colnames()

function was used to display all the column names of the data set. This allowed the identification of all the column headers present in the imported dataset. The dataset contain six columns namely site, TreeID, Circumf_2005_cm, Circumf_2015_cm, and Circumf_2020_cm.

```
# download the growth data file

download.file("https://raw.githubusercontent.com/ghazkha/Assessment4/refs/heads/main/growth_data.csv", d

# Read file
growth_data <- read.csv("growthdata.csv")
colnames(growth_data)
```

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

Step 7: Calculating the mean and standard deviation of tree circumference at the start and end of the study at both sites.

Firstly, the mean and standard deviation for each combination of site and year was calculated to summarize the tree circumference at the start (2005) and end (2020) of study for both sites. For this the data was subset by using logocal codition northeast and southwest site followed by application of mean() fuction to calculate the average circumference and the sd() function for calculating the mean deviation. Similarly, the steps were repeated for the northeast end (2020), southwest start(2005), southwest end (2020). After all the calculations were completed, they were combined into one clear summary atble using the data.frame() function and finally the table was displayed.This provided the summary of the average tree size and their variabilitiy at each site over the period of study. As shown in the table, the mean of the tree circumference increased at both the sites from 2005 to 2020. For the northeast site, the mean increased from 5.292 cm to 54.248 cm whereas for the southwest site, the mean increased from 4.862 to 45.596 cm. When compared at both sites, the northeast site recorded slightly higher mean circumfernece than the southwest at both start and end of study periods which could be attributed to slightly better growth conditions or initial size advantage. The standard variation increased dramatically from 2005 to 2020 from 0.91 to 25.22 for northeast site and from 1.14 to 17.87 for southwest site, indicating tree sizes become more variable at the end where some trees grew larger than others at both the sites.

```
# 1. Mean for Northeast site at the start (Circumf_2005_cm)

meannortheast_start<- mean(growth_data$Circumf_2005_cm[growth_data$Site== "northeast"]) # growth_data$S

# 2. standard deviation for the Northeast site at the start (Circumf_2005_cm)

sdnortheast_start <- sd(growth_data$Circumf_2005_cm[growth_data$Site== "northeast"])

# 3. Mean for Northeast site at the start (Circumf_2020_cm)

meannortheast_end <- mean(growth_data$Circumf_2020_cm[growth_data$Site== "northeast"])

# 4. standard deviation for the Northeast site at the start (Circumf_2005_cm)

sdnortheast_end <- sd(growth_data$Circumf_2020_cm[growth_data$Site== "northeast"])

# 5. Mean for Southwest site at the start (Circumf_2005_cm)

meansouthwest_start<- mean(growth_data$Circumf_2005_cm[growth_data$Site== "southwest"]) # growth_data$S

# 6. standard deviation for the Southwest site at the start (Circumf_2005_cm)
```

```r
sdsouthwest_start <- sd(growth_data$Circumf_2005_cm[growth_data$Site== "southwest"])


# 7. Mean for Southwest site at the end (Circumf_2020_cm)

meansouthwest_end<- mean(growth_data$Circumf_2020_cm[growth_data$Site== "southwest"])

# 8. standard deviation for the Southwest site at the end (Circumf_2020_cm)

sdsouthwest_end <- sd(growth_data$Circumf_2020_cm[growth_data$Site== "southwest"])

# creating summary table for mean and standard deviation of two sites at the start and end of study per

summary_table <- data.frame(
  Site = c("Northeast", "Northeast", "Southwest", "Southwest"),
  Year = c("2005 (Start)", "2020 (End)", "2005 (Start)", "2020 (End)"),
  Mean = c(meannortheast_start, meannortheast_end, meansouthwest_start, meansouthwest_end),
  SD = c(sdnortheast_start, sdnortheast_end, sdsouthwest_start, sdsouthwest_end)
)

# Displaying the table
summary_table
```

```
##        Site         Year   Mean         SD
## 1 Northeast 2005 (Start)  5.292  0.9140267
## 2 Northeast   2020 (End) 54.228 25.2279489
## 3 Southwest 2005 (Start)  4.862  1.1474710
## 4 Southwest   2020 (End) 45.596 17.8734549
```

step 8: Making a box plot for the circumference at the start and end of the study at both sites.

Boxplots was created to represent the distribution of the tree circumferences at the start and end of the study for both the sites. Firstly, the datas were splitted into Northeast and Southwest. Then, the boxplot() function was used to display the tree circumferences for both sites at both time points. To plot the box for both the sites, multiple datatsets were placed one after another inside the same boxplot() function, separated by commas. This automatically placed the boxplots side by side in the same figure for easy comparison.

For both northeast and southwest, the tree circumference increased substantially from 2005 to 2020, indicating significant tree growth over period of time. The northeast exhibited slightly higher median and tree circumference than southwest site at both the time points, indicating that trees in the northeast experience slightly greater growth. The data was spreaded largely in 2020 box as shown by interquartile range, reflecting greater variation in tree sizes after 15 years of growth. This indicated that some trees grew significantly larger than others which could be attributed to environmental factors. Overall, the box displayes increased tree size and variability over time at both study sites.

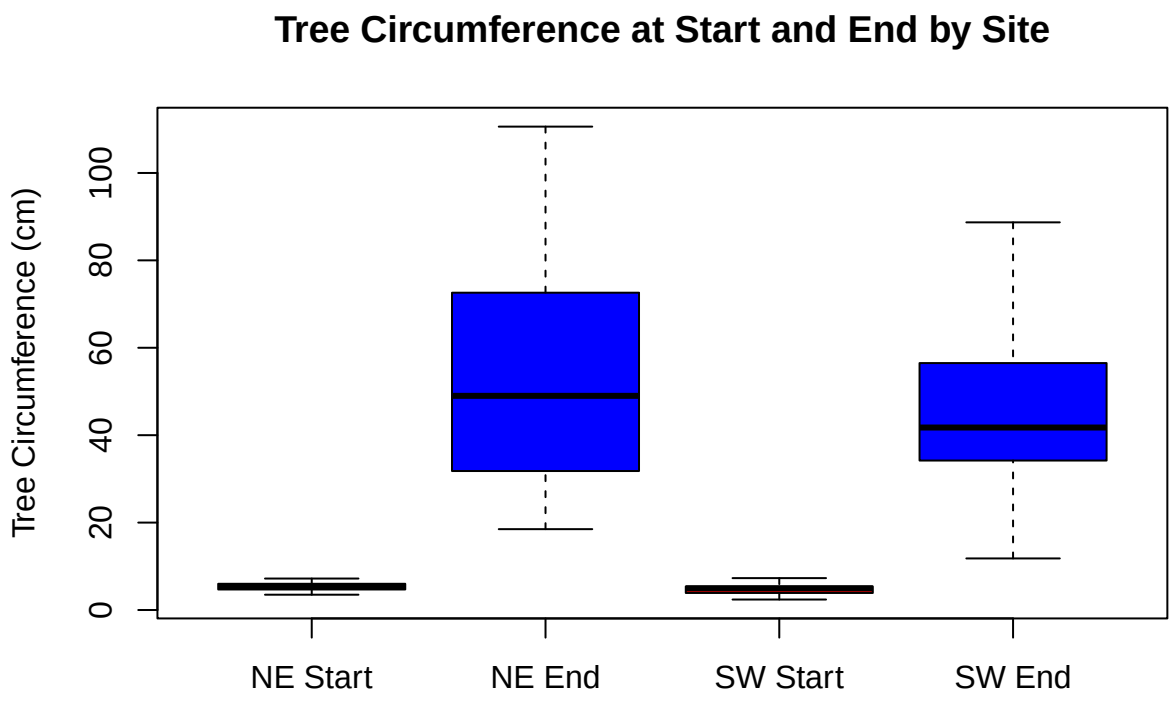
```r
# Splitting of data into Northeast and Southwest

northeast <- growth_data[growth_data$Site == "northeast", ]
southwest <- growth_data[growth_data$Site == "southwest", ]


# Creating boxplots for two different sites at the start and end of the study periods

boxplot(northeast$Circumf_2005_cm, northeast$Circumf_2020_cm,
        southwest$Circumf_2005_cm, southwest$Circumf_2020_cm,
        names = c("NE Start","NE End", "SW Start", "SW End"), # Provides the names to each boxplot
```

```
        col = c("red", "blue", "red", "blue"), # provides red color to the boxplot at the start and blu
        ylab = "Tree Circumference (cm)",
        main = "Tree Circumference at Start and End by Site")
```



Step 9: Calculating the mean growth over the last 10 years at each site.

To determine the mean tree growth over the last 10 years(from 2010 to 2020) at each study site, the difference in circumference between 2020 and 2010 was calculated for each tree and the output was saved in the new column called growth_10_years under object growth_data. Then, data were sepa rated by site to calculate the mean growth for each location. Following that, the mean () function was used separately to the northeast and southwest subsets to obtain the average tree growth over the 10-year period for each site.

```
# calculate growth data from 2010 to 2020

growth_data$ growth_10_years <- (growth_data$Circumf_2020_cm - growth_data$Circumf_2010_cm)

# Extracting 10-year growth values for northsite
North_east_growth_data <- (growth_data$growth_10_years[growth_data$Site == "northeast"])

# calculating mean for 10-year growth values of northwest
North_east_mean_growth_data <- mean(North_east_growth_data)
North_east_mean_growth_data
```

```
## [1] 42.94
```

```
# Extracting 10-year growth values for southwest
South_west_growth_data <- (growth_data$growth_10_years[growth_data$Site == "southwest"])
```

```r
# calculating mean for 10-year growth values of southwest

Southwest_mean_growth_data <- mean (South_west_growth_data)
Southwest_mean_growth_data
```

```
## [1] 35.49
```

Step 10: Using the t.test to estimate the p-value that the 10 year growth is different at the two sites

The p-value for the two sites were determined using t-test function. The north_east_growth_data which contains the last 10 years growth data for northeast site and South_west_growth_data which contains last 10 years growt data for southwest site were used to compare the growth between the two sites.The study observed higher mean 10-year growth of 42.94 at the northeast site compared to soutwest site (35.49 cm) but the growth difference was not statistically significant at 5% significance level (t-value = 1.8882, df = 87.978, and p-value = 0.06229).

```r
t.test(North_east_growth_data,South_west_growth_data)
```

```
##
##  Welch Two Sample t-test
##
## data:  North_east_growth_data and South_west_growth_data
## t = 1.8882, df = 87.978, p-value = 0.06229
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.3909251 15.2909251
## sample estimates:
## mean of x mean of y
##     42.94     35.49
```

PART 2: Examining the biological sequence diversity

step 1: Download the whole set of coding DNA sequences for E. coli and your organism of interest. How many coding sequences are present in these organisms? Present this in the form of a table. Describe any differences between the two organisms.

```r
suppressPackageStartupMessages({
  library("seqinr") # is a package designed to process and analyse sequence data.
  library("R.utils") # general utilities like zip and unzip
})
# loading e.coli and Saprospirales data
library("R.utils")

URL="https://ftp.ensemblgenomes.ebi.ac.uk/pub/bacteria/release-62/fasta/bacteria_58_collection/saprospi
download.file(URL,destfile="saprospirales_cds.fa.gz")
URL="http://ftp.ensemblgenomes.org/pub/bacteria/release-53/fasta/bacteria_0_collection/escherichia_coli
download.file(URL,destfile="ecoli_cds.fa.gz")

gunzip("ecoli_cds.fa.gz", overwrite  = TRUE)
gunzip("saprospirales_cds.fa.gz", overwrite  = TRUE)

list.files()
```

```
##  [1] "Assessment 4_R Project_SLE777.Rmd"
##  [2] "Assessment-4_R-Project_SLE777_files"
##  [3] "Assessment-4_R-Project_SLE777.Rmd"
##  [4] "Assessment-4_SLE777_Finalised_Ameeta.Rproj"
```

```
##  [5] "ecoli_cds.fa"
##  [6] "geneexpression.tsv"
##  [7] "growthdata.csv"
##  [8] "LICENSE"
##  [9] "Part 1_Assessment 4.Rmd"
## [10] "Part-1_Assessment-4.pdf"
## [11] "README.md"
## [12] "saprospirales_cds.fa"
```

```r
# Reading the FASTA file
library("seqinr")
e.coli_cds <- seqinr::read.fasta("ecoli_cds.fa")
saprospirales_cds <- seqinr::read.fasta("saprospirales_cds.fa")
# count number of sequences
e.coli_number <- length(e.coli_cds)
saprospirales_num <- length(saprospirales_cds)
# create table
cds_table <- data.frame(
  Bacteria = c("E.coli", "Saprospirales"),
  CDS_count = c(e.coli_number, saprospirales_num)
)
cds_table
```

```
##        Bacteria CDS_count
## 1        E.coli      4239
## 2 Saprospirales      4527
```