Assessment 4 : SLE777

Nishita Sood

2025-09-24

Assignment 4: R Project

PART-1: Gene Expression & Tree Growth Analysis

Importing & downloading gene expression tsv file

We begin by downloading the RNA-seq count data file (`gene_expression.tsv`) from the provided GitHub repository. This dataset contains gene expression counts for multiple samples, and we will use it to calculate summary statistics and visualise expression distributions.

```
# Loading the required library
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 v1.27.1 (2025-05-02 21:00:05 UTC) successfully loaded. See ?R.oo for help.

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

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

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

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

## R.utils v2.13.0 (2025-02-24 21:20:02 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
```

```r
# Downloading the file directly from GitHub repo
URL= "https://raw.githubusercontent.com/ghazkha/Assessment4/refs/heads/main/gene_expression.tsv"
download.file(URL,destfile="gene_expression.tsv")
list.files()
```

```
##  [1] "A4.html"                        "Assessment-4---R-project.Rproj"
##  [3] "Assignment4-Part1.html"         "Assignment4-Part1.pdf"
##  [5] "Assignment4-Part1.Rmd"          "Assignment4-Part2.Rmd"
##  [7] "ecoli_cds.fa"                   "gene_expression.tsv"
##  [9] "growth_data.csv"                "LICENSE"
## [11] "README.md"                      "salmonella_cds.fa"
```

STEP 1: Generating a table of values for first 6 genes

After downloading the data, the first step is to read it into R and ensure that the first column (gene identifiers) is used as the row names for easier referencing and display the first six rows.

```r
# Reading & making gene identifiers as row name
gene_data <- read.table("gene_expression.tsv",
                        stringsAsFactors = FALSE,  # keeping the text as plain text
                        header = TRUE,             # first row has column names
                        sep = "\t",                # it's a TSV file (tab-separated)
                        row.names = 1)             # making gene IDs the row names
# Showing first six rows of gene data
head(gene_data, 6)
```

```
##                               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
```

The table shows the first six genes and their expression counts across the three samples. This confirms the dataset was imported correctly and that each gene's expression is measured numerically.

STEP 2: Generating a table of values for first 6 genes with a new "mean column"

We calculate the mean across all samples for each gene. This summary column allows us to quickly compare gene expression levels across the dataset.

```r
# Calculating row-wise mean across all samples
gene_data$mean_value <- rowMeans(gene_data)
# Displaying the first 6 rows including the new mean column
head(gene_data, 6)
```

```
##                               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_value
## 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
```

The new "mean_value" column represents the average expression level across all samples. This provides a single summary metric for comparing gene activity.

STEP 3: The top 10 genes with the highest mean value

To find the most highly expressed genes, we sort by mean values in descending order.The code displays the top 10 genes with the highest mean value.

```
# Sorting data frame in descending order of mean value
top10_genes <- gene_data[order(-gene_data$mean_value), ]
# Showing the top 10 genes
head(top10_genes, 10)
```

```
##                           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_value
## 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
```

The top 10 genes have the highest mean expression, indicating they are the most actively transcribed genes in this dataset.

STEP 4: Number of genes with a mean <10

At this step, we count how many genes have a mean expression below 10.This helps to identify genes with low read counts, which may be filtered in downstream analysis.

```
# Counting how many genes have mean value < 10
low_meanexpression_genes <- sum(gene_data$mean_value < 10)
# Results
low_meanexpression_genes
```

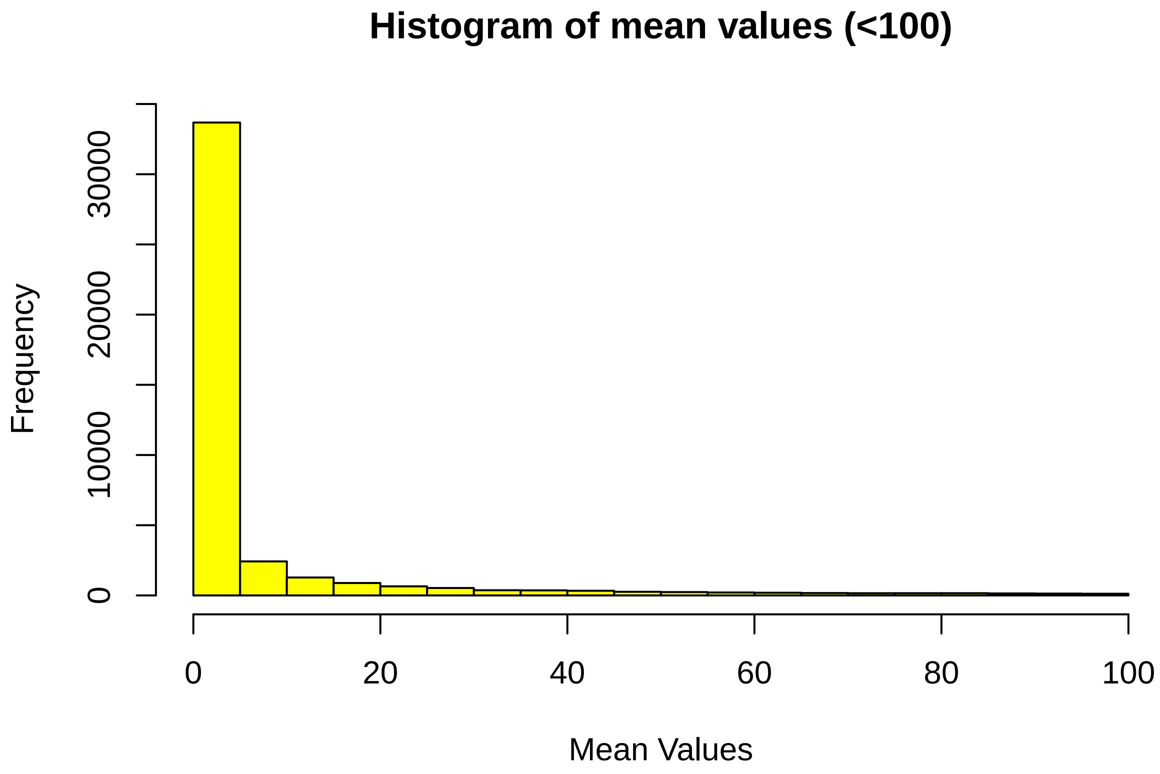
```
## [1] 35988
```

This value shows how many genes are weakly expressed. Low-expression genes may be filtered out before differential expression analysis to reduce noise.

STEP 5: Histogram plot of the mean values

A histogram is created to visualise the distribution of mean values. For clarity, we subset the data and only plot genes with a mean value below 100.

```
# Subsetting genes with mean < 100 for better visualisation
low_mean_genes <- subset(gene_data, mean_value < 100)
# Making a histogram from the subset
hist(low_mean_genes$mean_value,
     xlab = "Mean Values",
     ylab = "Frequency",
     main = "Histogram of mean values (<100)",
      col = "yellow",
     border = "black")
```



The histogram shows that most genes have relatively low mean expression values, while only a few are highly expressed — a typical pattern in RNA-seq data.

Importing & downloading tree growth csv file

Next, we download the tree growth dataset (growth_data.csv) from the GitHub repository.This dataset contains circumference measurements of trees in 2005 and 2020 at two sites: northeast and southwest. This dataset help assess tree growth trends over time.

```
# Downloading tree growth dataset
URL="https://raw.githubusercontent.com/ghazkha/Assessment4/refs/heads/main/growth_data.csv"
download.file(URL,destfile="growth_data.csv")
list.files()
```

```
##  [1] "A4.html"                         "Assessment-4---R-project.Rproj"
##  [3] "Assignment4-Part1_files"         "Assignment4-Part1.html"
##  [5] "Assignment4-Part1.pdf"           "Assignment4-Part1.Rmd"
##  [7] "Assignment4-Part2.Rmd"           "ecoli_cds.fa"
##  [9] "gene_expression.tsv"             "growth_data.csv"
## [11] "LICENSE"                         "README.md"
## [13] "salmonella_cds.fa"
```

STEP 6:The column names

After downloading the data, we read it into R and display the column names.

```
# Importing the CSV file
growth_data <- read.csv("growth_data.csv", header = TRUE)
# Results
colnames(growth_data)
```

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

The dataset includes tree circumference data from two sites (northeast and southwest) at two time points (2005 and 2020), which allows comparison of growth over time.

Step 7: Calculating mean & SD of circumference (2005 vs 2020 at both sites)

We calculate the mean and standard deviation of tree circumference at both sites in 2005 and 2020. This provides a summary of tree growth over the 15-year period.

```
# Sub-setting data for each site
siteNE <- subset(growth_data, Site == "northeast")
siteSW <- subset(growth_data, Site == "southwest")

# Creating a results table with mean & SD Values
results <- data.frame(
  Site = c("Northeast", "Northeast", "Southwest", "Southwest"),
  Year = c("2005", "2020", "2005", "2020"),
  Mean = c(mean(siteNE$Circumf_2005_cm),
           mean(siteNE$Circumf_2020_cm),
           mean(siteSW$Circumf_2005_cm),
           mean(siteSW$Circumf_2020_cm)),
  SD = c(sd(siteNE$Circumf_2005_cm),
         sd(siteNE$Circumf_2020_cm),
         sd(siteSW$Circumf_2005_cm),
         sd(siteSW$Circumf_2020_cm)),
  stringsAsFactors = FALSE
)
```

```
# Displaying the results table
results
```

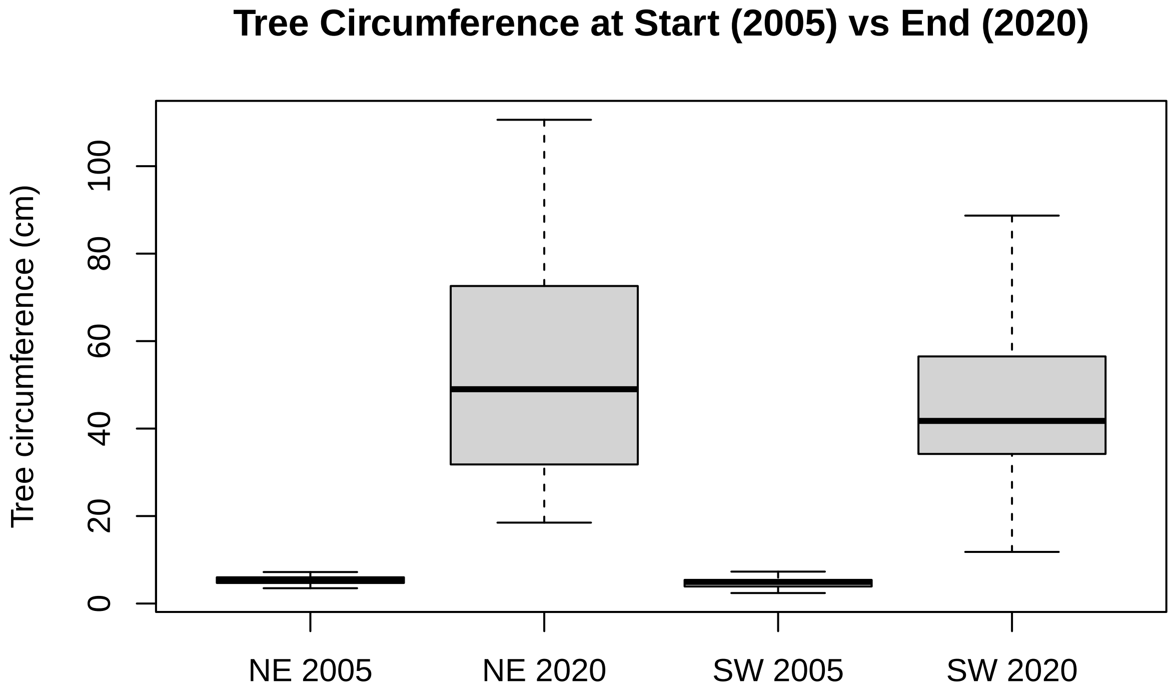
```
##        Site Year   Mean         SD
## 1 Northeast 2005  5.292  0.9140267
## 2 Northeast 2020 54.228 25.2279489
## 3 Southwest 2005  4.862  1.1474710
## 4 Southwest 2020 45.596 17.8734549
```

The table summarises average tree circumference and variability (SD) for each site and year. Trees in 2020 are generally larger, suggesting growth over the 15-year period.

Step 8: Box plot of tree circumference at the start and end of the study at both sites.

Here, We visualise the distributions of tree circumference at both sites in 2005 and 2020 using a boxplot.This provides a comparison of growth across time and location.

```
# Creating a boxplot for tree circumference in 2005 vs 2020
boxplot(siteNE$Circumf_2005_cm, siteNE$Circumf_2020_cm,
        siteSW$Circumf_2005_cm, siteSW$Circumf_2020_cm,
        ylab = "Tree circumference (cm)",
        main = "Tree Circumference at Start (2005) vs End (2020)",
        names = c("NE 2005", "NE 2020", "SW 2005", "SW 2020"))
```



The boxplot visually confirms increased tree circumference at both sites from 2005 to 2020, with northeast trees showing slightly greater median growth.

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

Then, we calculate the average growth in circumference between 2005 and 2020 for each site. This is done by subtracting 2005 values from 2020 values.

```
# Creating growth column = difference between 2020 and 2005
siteNE$Growth <- siteNE$Circumf_2020_cm - siteNE$Circumf_2005_cm
siteSW$Growth <- siteSW$Circumf_2020_cm - siteSW$Circumf_2005_cm

# Summarising mean growth for each site
growth_results <- data.frame(
  Site = c("Northeast", "Southwest"),
  Mean_Growth = c(mean(siteNE$Growth),
                  mean(siteSW$Growth)),
  stringsAsFactors = FALSE
)
# Displaying the results
growth_results
```

```
##        Site Mean_Growth
## 1 Northeast      48.936
## 2 Southwest      40.734
```

This table shows the average increase in circumference per tree for each site. The northeast site has higher mean growth, indicating better growth conditions or soil factors.

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

Finally, we use a t-test to compare the growth of trees between the northeast and southwest sites. The p-value tells us whether the difference in mean growth is statistically significant.

```
# Running t-test for difference in growth between sites
t.test(siteNE$Growth, siteSW$Growth)
```

```
##
##  Welch Two Sample t-test
##
## data:  siteNE$Growth and siteSW$Growth
## t = 1.9121, df = 88.083, p-value = 0.05912
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.3226343 16.7266343
## sample estimates:
## mean of x mean of y
##    48.936    40.734
```

The p-value (0.059) is slightly above 0.05, indicating that the growth difference between the northeast and southwest sites is not statistically significant.

Overall, the analysis of gene expression and tree growth data revealed clear trends — most genes showed low expression levels with a few highly expressed outliers, and trees at both sites exhibited increased circumference over time, though the difference in growth between sites was not statistically significant.