Assignment

2025-10-06

Question 1: Read and inspect the gene_expression.tsv file

To solve this, I downloaded the `gene_expression.tsv` file from the provided GitHub link, read it into R using `read.table()`, and set the first column as row names so each row represents a gene. Then I displayed a table of values for the first six genes.

```r
# Download the file from GitHub
download.file("https://raw.githubusercontent.com/ghazkha/Assessment4/master/gene_expression.tsv",
              "gene_expression.tsv")

# Read the file and make gene IDs the row names
gene_expression <- read.table("gene_expression.tsv", header = TRUE, row.names = 1, sep = "\t")

# Show a table of values for the first six genes
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
```

The table above shows the first six genes from the gene_expression.tsv dataset. Each row represents a unique gene identifier, and each column shows the expression values of that gene across different samples. This step confirms that the file has been imported correctly and that the structure of the dataset is as expected.

Question 2: Calculate mean expression for each gene

To solve this, I created a new column called `mean_expression` by calculating the average expression across all samples for each gene using `rowMeans()`. Then I displayed a table of values for the first six genes, showing only the new column.

```r
# Question 2: Create a new column 'mean_expression' as the mean of all samples
gene_expression$mean_expression <- rowMeans(gene_expression)

# Show a table with gene names and their mean expression (first six genes)
head(gene_expression["mean_expression"], 6)
```

```
##                                mean_expression
## ENSG00000223972.5_DDX11L1          0.0000000
## ENSG00000227232.5_WASH7P         146.3333333
## ENSG00000278267.1_MIR6859-1        0.3333333
## ENSG00000243485.5_MIR1302-2HG      0.3333333
## ENSG00000237613.2_FAM138A          0.0000000
## ENSG00000268020.3_OR4G4P           0.3333333
```

The table above shows the first six genes along with their calculated mean expression values across all samples. The new column mean_expression represents the average expression for each gene, providing a single summary value that makes it easier to compare gene expression levels.

Question 3: List the 10 genes with the highest mean expression

To solve this, I sorted the dataset in descending order based on the `mean_expression` column created in Question 2. Then I displayed a table of the top 10 genes with the highest mean expression values.

```r
# Sort genes by mean expression in descending order
top10_genes <- gene_expression[order(-gene_expression$mean_expression), ]

# Show only the top 10 genes with their mean expression
head(top10_genes["mean_expression"], 10)
```

```
##                            mean_expression
## 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
```

The table above lists the 10 genes with the highest mean expression values. These genes show the strongest overall expression across all samples.

Question 4: Determine the number of genes with a mean < 10

To solve this, I used a logical condition to filter genes whose `mean_expression` value is less than 10, then used the `sum()` function to count how many such genes there are.

```r
# Count the number of genes with mean expression less than 10
num_genes_below_10 <- sum(gene_expression$mean_expression < 10)

# Display the result
num_genes_below_10
```

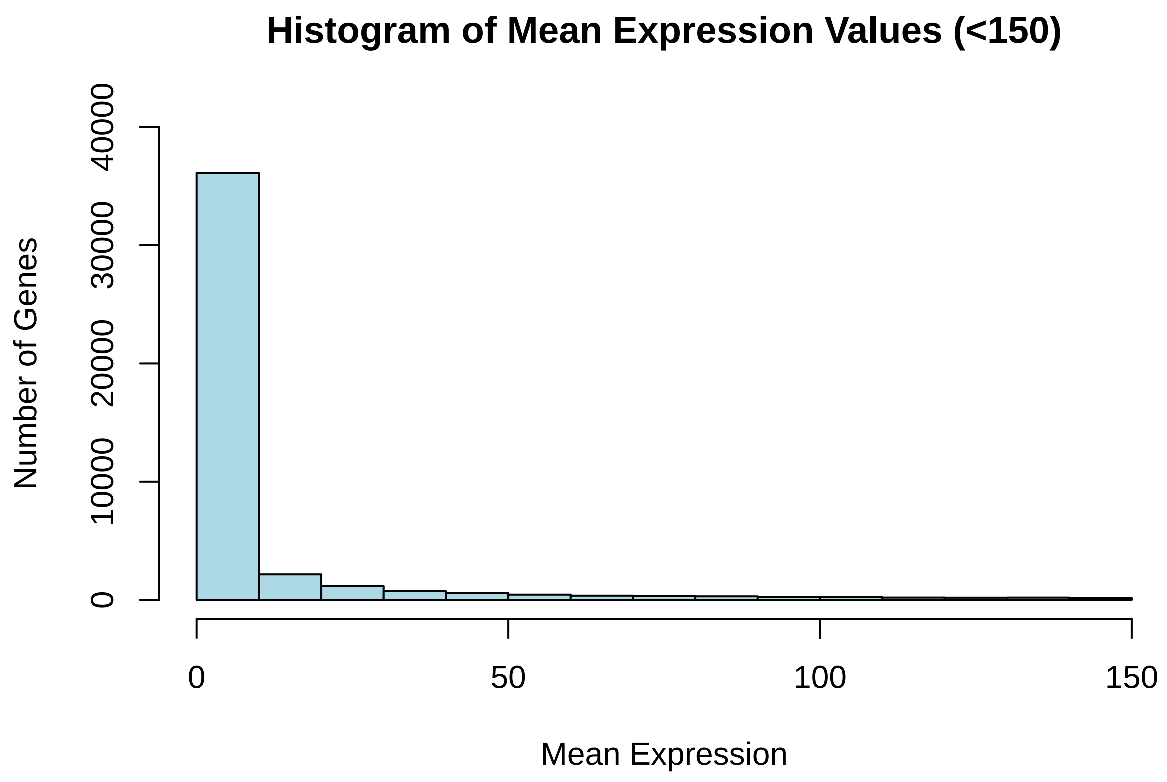
```
## [1] 35988
```

The result shows that 35,988 genes have a mean expression value below 10. This indicates that the majority of genes in the dataset are expressed at relatively low levels.

Question 5: Make a histogram plot of the mean values

To solve this, I used the hist() function to plot the distribution of mean_expression values across all genes. Because the dataset is highly skewed, with most genes showing very low expression and only a few showing extremely high values, the full histogram did not clearly show the distribution. To make the pattern easier to interpret, I focused the x-axis on genes with mean expression values below 150. This allowed a clearer view of the distribution of the majority of genes while still using the same data.

```r
# Plot histogram focused on genes with mean expression below 150
hist(gene_expression$mean_expression[gene_expression$mean_expression < 150],
     main = "Histogram of Mean Expression Values (<150)",
     xlab = "Mean Expression",
     ylab = "Number of Genes",
     col = "lightblue",
     ylim = c(0, 40000))
```



The histogram above shows the distribution of mean expression values for genes, focused on values below 150 to make the pattern clearer. Most genes have very low mean expression levels, while only a small number reach higher values. By restricting the x-axis, the distribution of the majority of genes becomes easier to visualise, highlighting the skewed nature of the dataset.

Question 6: Import the CSV file into an R object and show the column names

To solve this, I downloaded the `growth_data.csv` file from the provided GitHub link and imported it into R as a data frame using the `read.csv()` function. Then I used the `colnames()` function to display the names of all columns in the dataset.

```r
# Download the CSV file from GitHub
download.file("https://raw.githubusercontent.com/ghazkha/Assessment4/master/growth_data.csv",
              "growth_data.csv")

# Import the CSV file into R
```

```r
growth_data <- read.csv("growth_data.csv", header = TRUE)

# Display the column names of the dataset
colnames(growth_data)
```

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

The CSV file was successfully imported into R as a data frame named growth_data. The dataset contains six columns: Site, TreeID, Circumf_2005_cm, Circumf_2010_cm, Circumf_2015_cm, and Circumf_2020_cm.

Question 7: Calculate the mean and standard deviation of tree circumference at the start and end of the study at both sites

To solve this, I used the `mean()` and `sd()` functions to calculate the mean and standard deviation of tree circumference at the start (2005) and end (2020) of the study for each site.

```r
# Mean and SD for northeast site at start (2005)
mean_ne_start <- mean(growth_data$Circumf_2005_cm[growth_data$Site == "northeast"], na.rm = TRUE)
sd_ne_start <- sd(growth_data$Circumf_2005_cm[growth_data$Site == "northeast"], na.rm = TRUE)

# Mean and SD for northeast site at end (2020)
mean_ne_end <- mean(growth_data$Circumf_2020_cm[growth_data$Site == "northeast"], na.rm = TRUE)
sd_ne_end <- sd(growth_data$Circumf_2020_cm[growth_data$Site == "northeast"], na.rm = TRUE)

# Mean and SD for southwest site at start (2005)
mean_sw_start <- mean(growth_data$Circumf_2005_cm[growth_data$Site == "southwest"], na.rm = TRUE)
sd_sw_start <- sd(growth_data$Circumf_2005_cm[growth_data$Site == "southwest"], na.rm = TRUE)

# Mean and SD for southwest site at end (2020)
mean_sw_end <- mean(growth_data$Circumf_2020_cm[growth_data$Site == "southwest"], na.rm = TRUE)
sd_sw_end <- sd(growth_data$Circumf_2020_cm[growth_data$Site == "southwest"], na.rm = TRUE)

# Display all results
mean_ne_start; sd_ne_start
```

```
## [1] 5.292

## [1] 0.9140267
```

```r
mean_ne_end; sd_ne_end
```

```
## [1] 54.228

## [1] 25.22795
```

```r
mean_sw_start; sd_sw_start
```

```
## [1] 4.862

## [1] 1.147471
```

```r
mean_sw_end; sd_sw_end
```

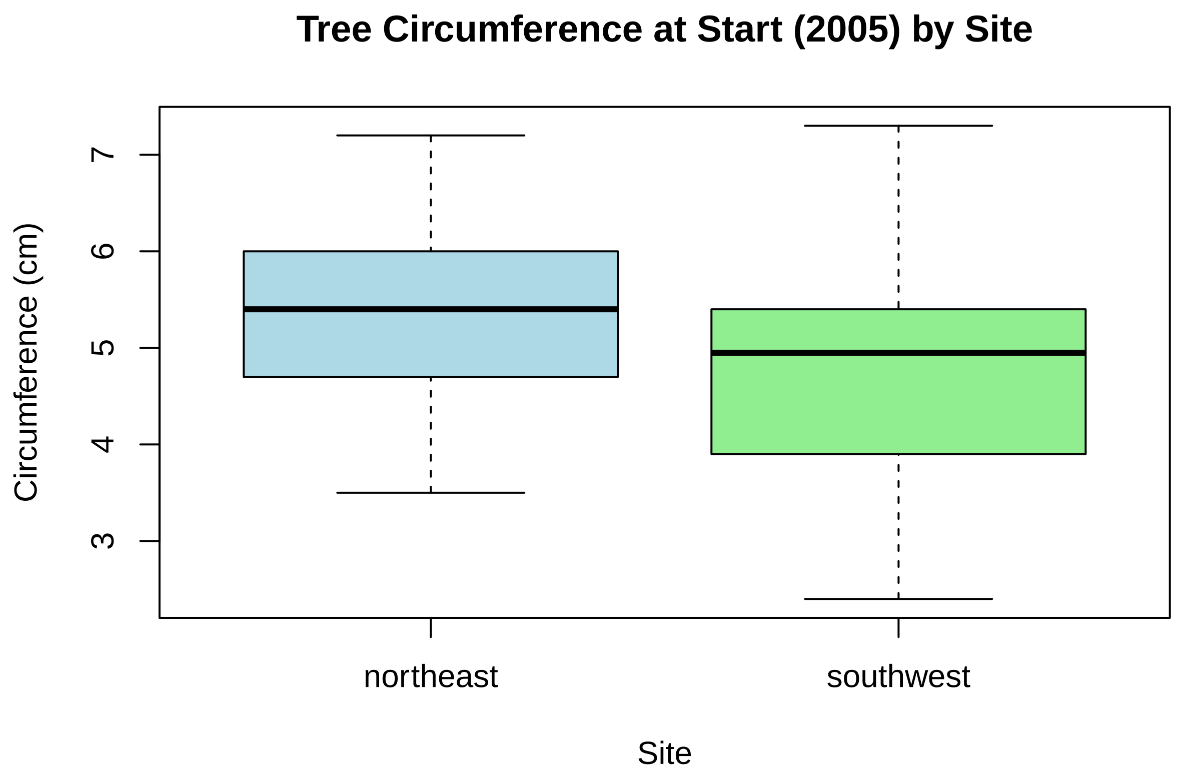
```
## [1] 45.596

## [1] 17.87345
```

The results show the mean and standard deviation of tree circumference at the start (2005) and end (2020) of the study for both sites: Northeast site: Mean circumference in 2005: 5.292 cm, SD: 0.914 cm Mean circumference in 2020: 54.228 cm, SD: 25.228 cm Southwest site: Mean circumference in 2005: 4.862 cm, SD: 1.147 cm Mean circumference in 2020: 45.596 cm, SD: 17.873 cm These results show that tree circumference increased substantially at both sites over the 15-year period, while the larger standard deviations in 2020 reflect increased variation in growth among trees by the end of the study.

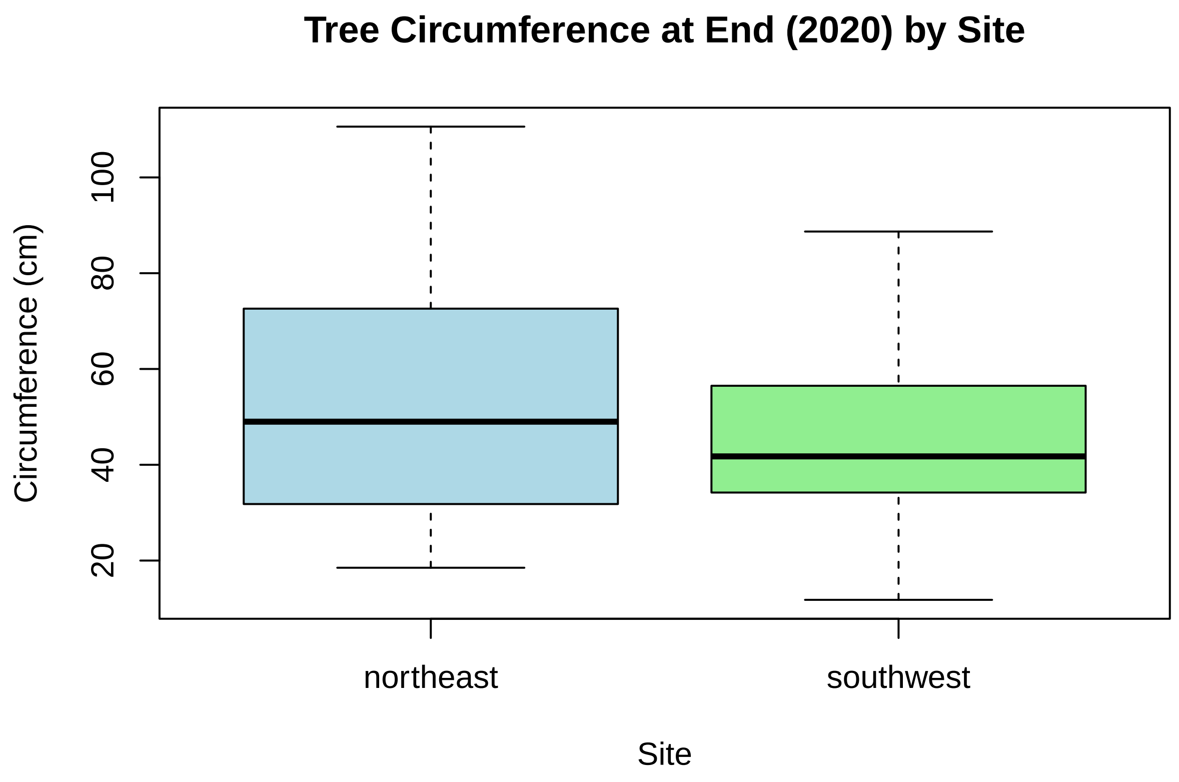
Question 8: Make a box plot of tree circumference at the start and end of the study at both sites

To solve this, I used the `boxplot()` function to compare tree circumference at the start (2005) and end (2020) of the study for both sites.

```
# Create side-by-side boxplots
boxplot(
  Circumf_2005_cm ~ Site,
  data = growth_data,
  main = "Tree Circumference at Start (2005) by Site",
  xlab = "Site",
  ylab = "Circumference (cm)",
  col = c("lightblue", "lightgreen")
)
```



```
boxplot(
  Circumf_2020_cm ~ Site,
  data = growth_data,
  main = "Tree Circumference at End (2020) by Site",
  xlab = "Site",
  ylab = "Circumference (cm)",
  col = c("lightblue", "lightgreen")
)
```



The boxplots above show the distribution of tree circumference at the start (2005) and end (2020) of the study for both sites. In 2005, trees at both sites had similar circumferences, with slightly higher median values at the northeast site. By 2020, the median and overall spread of tree circumferences increased substantially at both sites, with the northeast site showing slightly greater growth and variability.

Question 9: Calculate the mean growth over the last 10 years at each site

To solve this, I subtracted `Circumf_2010_cm` from `Circumf_2020_cm` to calculate growth over the last 10 years for each tree. Then I calculated the mean growth separately for each site using the `mean()` function.

```
# Calculate growth over the last 10 years for each tree
growth_data$Growth_last10yrs <- growth_data$Circumf_2020_cm - growth_data$Circumf_2010_cm

# Calculate mean growth for each site
mean_growth_northeast <- mean(growth_data$Growth_last10yrs[growth_data$Site == "northeast"])
mean_growth_southwest <- mean(growth_data$Growth_last10yrs[growth_data$Site == "southwest"])

# Display the results
mean_growth_northeast
```

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

```
mean_growth_southwest
```

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

The results show the mean tree growth over the last 10 years at each site: Northeast site: 42.94 cm Southwest site: 35.49 cm These values indicate that trees in the northeast site experienced greater growth on average compared to those in the southwest site over the same 10-year period.

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

To solve this, I used the `t.test()` function to compare the 10-year growth between the two sites. The null hypothesis is that there is no difference in mean growth, and the alternative hypothesis is that the growth differs between sites. The resulting p-value indicates whether the difference is statistically significant. If the p-value is less than 0.05, we reject the null hypothesis and conclude that growth differs significantly between the two sites.

```r
# Perform a t-test to compare growth between sites
t_test_result <- t.test(Growth_last10yrs ~ Site, data = growth_data)

# Display the t-test result
t_test_result
```

```
##
##  Welch Two Sample t-test
##
## data:  Growth_last10yrs by Site
## t = 1.8882, df = 87.978, p-value = 0.06229
## alternative hypothesis: true difference in means between group northeast and group southwest is not
## 95 percent confidence interval:
##  -0.3909251 15.2909251
## sample estimates:
## mean in group northeast mean in group southwest
##                   42.94                   35.49
```

The t-test produced a p-value of 0.06229, which is slightly above the standard significance level of 0.05. This means we fail to reject the null hypothesis, suggesting that the difference in 10-year growth between the two sites is not statistically significant.