Group12

Tharushi Morais

2024-10-11

#TIP: To download a file with R, click on "view raw" and then you can copy the URL from the address bar and then use the download.file command in R.

```
url <- "https://raw.githubusercontent.com/ghazkha/Assessment4/main/gene_expression.tsv"
destfile <- "gene_expression.tsv"
download.file(url, destfile)
```

```
url <- "https://raw.githubusercontent.com/ghazkha/Assessment4/main/growth_data.csv"
destfile <- "growth_data.csv"
download.file(url, destfile)
```

#1.Read in the file, making the gene identifiers the row names. Show a table of values for the first six genes#

```
# Read the gene_expression.tsv file using read.table
gene_expression <- read.table("gene_expression.tsv", header = TRUE, sep = "\t", row.names = 1)
# Show a table of values for the first six genes
head(gene_expression, 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
```

#2.Make a new column which is the mean of the other columns. Show a table of values for the first six genes#

```
# Assuming gene_expression is already loaded
# Calculate the mean of the gene expression values (excluding row names)
gene_expression$mean_expression <- rowMeans(gene_expression)
# Show a table of values for the first six genes, including the new mean column
head(gene_expression, 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_expression
## 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
```

#3.List the 10 genes with the highest mean expression#

```r
# Assuming the mean_expression column has already been added
# Order the data frame by the mean expression in descending order
top_genes <- gene_expression[order(-gene_expression$mean_expression), ]

# Select the top 10 genes
top_10_genes <- head(top_genes, 10)

# Display the top 10 genes with their mean expression values
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_expression
## 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
```

#4.Determine the number of genes with a mean <10#

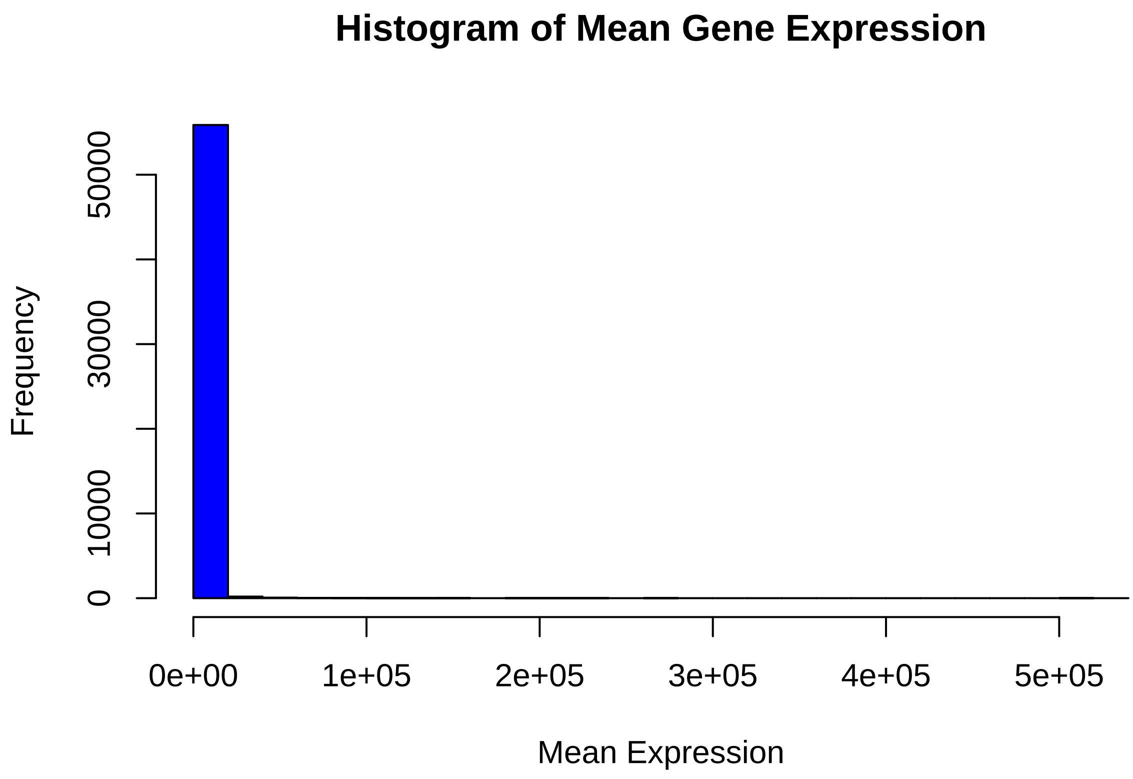
```r
# Assuming the mean_expression column has already been added
# Count the number of genes with mean expression less than 10
num_genes_below_10 <- sum(gene_expression$mean_expression < 10)

# Display the result
num_genes_below_10
```

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

#5.Make a histogram plot of the mean values and include it into your report#

```
# Create a histogram of the mean expression values using base R
hist(gene_expression$mean_expression,
     breaks = 20,              # Number of bins
     col = "blue",            # Fill color
     border = "black",        # Border color
     main = "Histogram of Mean Gene Expression", # Title
     xlab = "Mean Expression", # x-axis label
     ylab = "Frequency")       # y-axis label
```



#6.Import this csv file into an R object. What are the column names#

```
# Read the CSV file into an R object
growth_data <- read.csv("growth_data.csv")
# Display the column names of the data frame
column_names <- colnames(growth_data)
print(column_names)
```

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

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

```
# Read in the CSV file
growth_data <- read.csv("growth_data.csv")

# Display the first few rows of the data to understand its structure
```

```r
head(growth_data)
```

```
##        Site TreeID Circumf_2005_cm Circumf_2010_cm Circumf_2015_cm
## 1 northeast   A012             5.2            10.1            19.9
## 2 southwest   A039             4.9             9.6            18.9
## 3 southwest   A010             3.7             7.3            14.3
## 4 northeast   A087             3.8             6.5            10.9
## 5 southwest   A074             3.8             6.4            10.9
## 6 northeast   A008             5.9            10.0            16.8
##   Circumf_2020_cm
## 1            38.9
## 2            37.0
## 3            28.1
## 4            18.5
## 5            18.4
## 6            28.4
```

```r
# Assuming the first half of the data is for the Control site
# and the second half is for the Treatment site. Adjust as necessary.

# Split the data based on site
control_data <- growth_data[1:(nrow(growth_data) / 2), ]
treatment_data <- growth_data[((nrow(growth_data) / 2) + 1):nrow(growth_data), ]

# Calculate mean and standard deviation for the Control site
mean_start_control <- mean(control_data$Circumf_2005_cm, na.rm = TRUE)
sd_start_control <- sd(control_data$Circumf_2005_cm, na.rm = TRUE)

mean_end_control <- mean(control_data$Circumf_2020_cm, na.rm = TRUE)
sd_end_control <- sd(control_data$Circumf_2020_cm, na.rm = TRUE)

# Calculate mean and standard deviation for the Treatment site
mean_start_treatment <- mean(treatment_data$Circumf_2005_cm, na.rm = TRUE)
sd_start_treatment <- sd(treatment_data$Circumf_2005_cm, na.rm = TRUE)

mean_end_treatment <- mean(treatment_data$Circumf_2020_cm, na.rm = TRUE)
sd_end_treatment <- sd(treatment_data$Circumf_2020_cm, na.rm = TRUE)

# Display the results
results <- data.frame(
  Site = c("Control", "Control", "Treatment", "Treatment"),
  Measurement = c("Start (2005)", "End (2020)", "Start (2005)", "End (2020)"),
  Mean = c(mean_start_control, mean_end_control, mean_start_treatment, mean_end_treatment),
  SD = c(sd_start_control, sd_end_control, sd_start_treatment, sd_end_treatment)
)

print(results)
```

```
##        Site  Measurement   Mean        SD
## 1   Control Start (2005)  5.078  1.059127
## 2   Control   End (2020) 40.052 16.904428
## 3 Treatment Start (2005)  5.076  1.060527
## 4 Treatment   End (2020) 59.772 22.577839
```

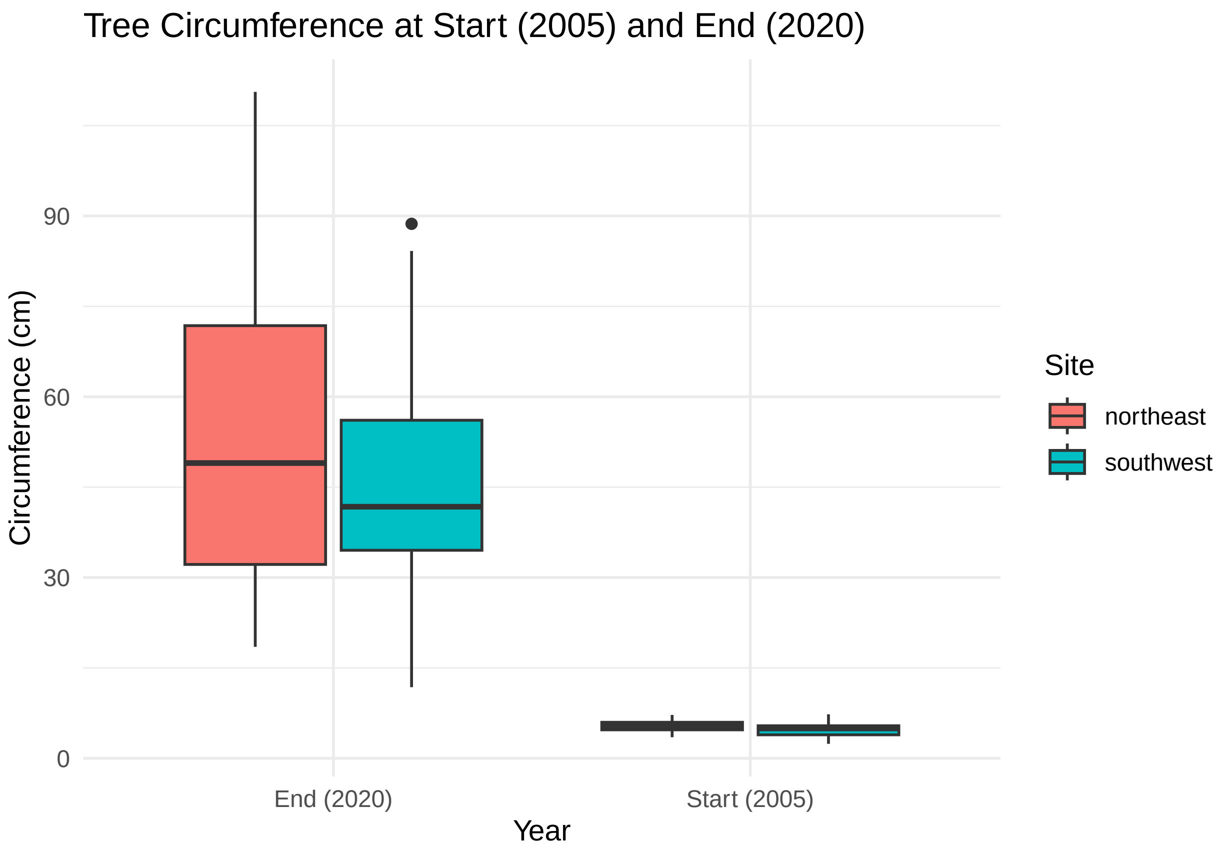
#8.Make a box plot of tree circumference at the start and end of the study at both sites

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

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

# Create a new data frame for plotting
plot_data <- data.frame(
  Circumference = c(growth_data$Circumf_2005_cm, growth_data$Circumf_2020_cm),
  Year = rep(c("Start (2005)", "End (2020)"), each = nrow(growth_data)),
  Site = rep(growth_data$Site, 2)  # Adjust according to your structure
)

# Create the box plot
ggplot(plot_data, aes(x = Year, y = Circumference, fill = Site)) +
  geom_boxplot() +
  labs(title = "Tree Circumference at Start (2005) and End (2020)",
       y = "Circumference (cm)",
       x = "Year") +
  theme_minimal()
```



#9.Calculate the mean growth over the last 10 years at each site#

```r
install.packages("dplyr")
```

```
## Installing package into '/home/s224650194/R/x86_64-pc-linux-gnu-library/4.1'
## (as 'lib' is unspecified)
```

```
# Load necessary library
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
# Read the CSV file
growth_data <- read.csv("growth_data.csv")

# Calculate growth and summarize mean growth for each site
growth_summary <- growth_data %>%
  mutate(Growth = Circumf_2020_cm - Circumf_2010_cm) %>%
  group_by(Site) %>%
  summarise(Mean_Growth = mean(Growth, na.rm = TRUE))

# Display the results
print(growth_summary)
```

```
## # A tibble: 2 x 2
##   Site      Mean_Growth
##   <chr>           <dbl>
## 1 northeast        42.9
## 2 southwest        35.5
```

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

```
# Load necessary library
library(dplyr)

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

# Calculate growth for each site
growth_data$Growth <- growth_data$Circumf_2020_cm - growth_data$Circumf_2010_cm

# Perform t-test comparing growth between Control and Treatment
t_test_result <- t.test(Growth ~ Site, data = growth_data)

# Display the results
print(t_test_result)
```

```
##
##  Welch Two Sample t-test
##
## data:  Growth 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
```