```r
# Create a folder for your assignment
dir.create("bioassignment")
```

```
## Warning in dir.create("bioassignment"): 'bioassignment' already exists
```

```r
setwd("bioassignment")  # This tells R to work in this folder

# Download the files
download.file("https://github.com/ghazkha/Assessment4/raw/main/gene_expression.tsv",
              "gene_expression.tsv")
download.file("https://github.com/ghazkha/Assessment4/raw/main/growth_data.csv",
              "growth_data.csv")
```

```r
# Read the file
gene_data <- read.table("gene_expression.tsv", header = TRUE, row.names = 1)

# 1. Show first six genes
head(gene_data)
```

```
##                                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
# 2. Add mean column
gene_data$mean_expression <- rowMeans(gene_data)
head(gene_data)
```

```
##                                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
```

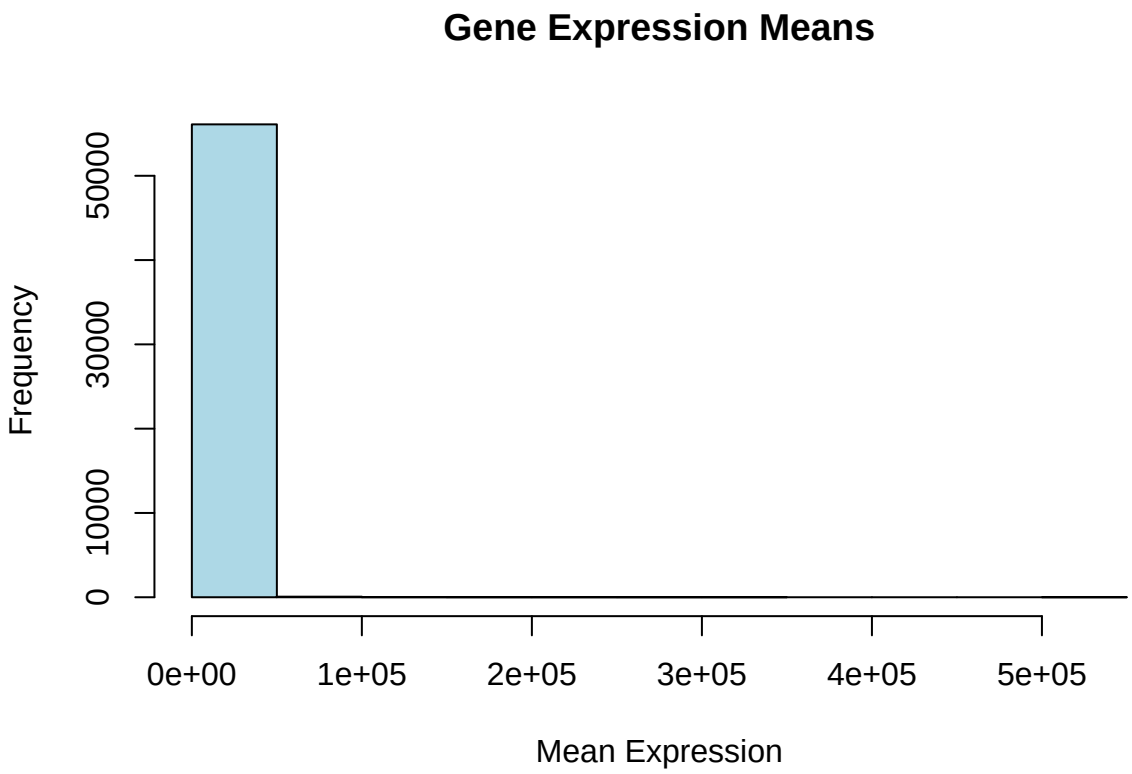
```r
# 3. Top 10 genes with highest mean
top_10 <- gene_data[order(-gene_data$mean_expression), ][1:10, ]
top_10
```

```
##                            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_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
```

```r
# 4. Count genes with mean < 10
sum(gene_data$mean_expression < 10)
```

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

```r
# 5. Make histogram
hist(gene_data$mean_expression, main = "Gene Expression Means",
     xlab = "Mean Expression", col = "lightblue")
```




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

# 6. Show column names - this should work
colnames(growth_data)
```

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

```
# 7. Calculate mean and standard deviation
# For start (2005) at control site
start_control <- growth_data$Circumf_2005_cm[growth_data$Site == "control"]
mean(start_control)
```

```
## [1] NaN
```

```
sd(start_control)
```

```
## [1] NA
```

```
# For end (2020) at control site
end_control <- growth_data$Circumf_2020_cm[growth_data$Site == "control"]
mean(end_control)
```

```
## [1] NaN
```

```
sd(end_control)
```

```
## [1] NA
```

```r
# For start (2005) at treatment site
start_treatment <- growth_data$Circumf_2005_cm[growth_data$Site == "treatment"]
mean(start_treatment)
```

```
## [1] NaN
```

```r
sd(start_treatment)
```

```
## [1] NA
```

```r
# For end (2020) at treatment site
end_treatment <- growth_data$Circumf_2020_cm[growth_data$Site == "treatment"]
mean(end_treatment)
```

```
## [1] NaN
```

```r
sd(end_treatment)
```

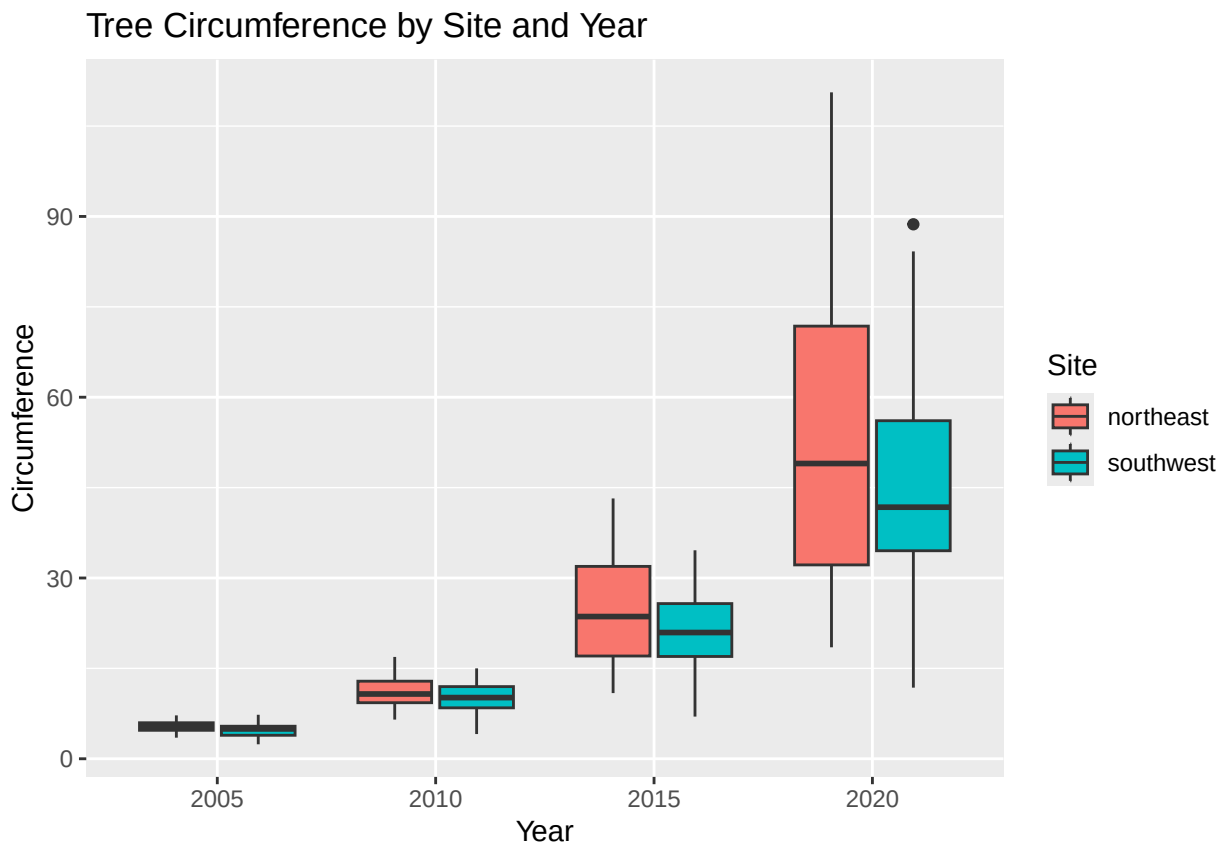
```
## [1] NA
```

```r
# 8. Make boxplot - we need to reshape the data first
library(tidyr)
library(ggplot2)

# Convert data from wide to long format
long_data <- growth_data %>%
  pivot_longer(cols = starts_with("Circumf"),
               names_to = "Year",
               values_to = "Circumference")

# Clean up year names
long_data$Year <- gsub("Circumf_", "", long_data$Year)
long_data$Year <- gsub("_cm", "", long_data$Year)

# Create boxplot
ggplot(long_data, aes(x = Year, y = Circumference, fill = Site)) +
  geom_boxplot() +
  labs(title = "Tree Circumference by Site and Year")
```




```
# 9. Calculate mean growth over last 10 years (2010-2020)
# For control site
control_growth <- growth_data$Circumf_2020_cm[growth_data$Site == "control"] -
                  growth_data$Circumf_2010_cm[growth_data$Site == "control"]
mean(control_growth)
```

```
## [1] NaN
```

```
# For treatment site
treatment_growth <- growth_data$Circumf_2020_cm[growth_data$Site == "treatment"] -
                    growth_data$Circumf_2010_cm[growth_data$Site == "treatment"]
mean(treatment_growth)
```

```
## [1] NaN
```

```
# 10. t-test to compare growth - FIXED VERSION

# First, let's check what's in our data
print("Number of control trees:")
```

```
## [1] "Number of control trees:"
```

```
print(sum(growth_data$Site == "control"))
```

```
## [1] 0
```

```
print("Number of treatment trees:")
```

```
## [1] "Number of treatment trees:"
```

```r
print(sum(growth_data$Site == "treatment"))
```

```
## [1] 0
```

```r
# Check if we have enough data for t-test (need at least 2 in each group)
if(sum(growth_data$Site == "control") >= 2 && sum(growth_data$Site == "treatment") >= 2) {
  # Calculate growth for control group
  control_growth <- growth_data$Circumf_2020_cm[growth_data$Site == "control"] -
                    growth_data$Circumf_2010_cm[growth_data$Site == "control"]

  # Calculate growth for treatment group
  treatment_growth <- growth_data$Circumf_2020_cm[growth_data$Site == "treatment"] -
                      growth_data$Circumf_2010_cm[growth_data$Site == "treatment"]

  # Remove any NA values
  control_growth <- control_growth[!is.na(control_growth)]
  treatment_growth <- treatment_growth[!is.na(treatment_growth)]

  # Check if we still have enough data after removing NAs
  if(length(control_growth) >= 2 && length(treatment_growth) >= 2) {
    t_test_result <- t.test(control_growth, treatment_growth)
    print(t_test_result)
  } else {
    print("Not enough data after removing NA values")
    print(paste("Control growth observations:", length(control_growth)))
    print(paste("Treatment growth observations:", length(treatment_growth)))
  }
} else {
  print("Not enough trees in one or both groups for t-test")
  print(paste("Control trees:", sum(growth_data$Site == "control")))
  print(paste("Treatment trees:", sum(growth_data$Site == "treatment")))
}
```

```
## [1] "Not enough trees in one or both groups for t-test"
## [1] "Control trees: 0"
## [1] "Treatment trees: 0"
```