assignment1

junichiro-ui

2025-03-18

Introduction

This document contains all the documentation for Assignment 1.

R Installation

Followed the instructions on CRAN and installed the latest R and RStudio.

Basic R Operations

Task1

1. Read the research article of the hands-on working group you are assigned to (see file "Student Groups.pdf" in shared folder General course material).

2. Answer the following questions

a. What is the medically relevant insight from the article? The study investigates selective insulin resistance in human white adipose tissue (WAT) in the obese state and how it is altered by weight loss following bariatric surgery. The key medical insight is that obesity leads to a selective insulin response in WAT, where some insulin-regulated genes remain active while others become unresponsive. Importantly, most insulin signaling and gene expression changes are restored after significant weight loss. This research enhances our understanding of adipose tissue dysfunction in obesity and type 2 diabetes, providing potential molecular targets for improving insulin action. ##### b. Which genomics technology/ technologies were used? The study utilized RNAseq, specifically 5' cap analysis of gene expression (CAGE), to analyze transcriptional responses to insulin in subcutaneous WAT. This method allowed the identification of transcription start sites and gene expression levels. The study also used bioinformatic analyses, including differential expression analysis and transcription factor binding site enrichment, to interpret the insulin response at the transcriptional level. #### 3. Further related research questions ##### a. List and explain at least three questions/ hypotheses you can think of that extend the analysis presented in the paper. **Does selective insulin resistance in WAT affect systemic metabolic homeostasis, and how does it compare to liver and muscle tissue?** Since insulin resistance in different tissues contributes to metabolic disorders, it would be valuable to compare the transcriptional response in WAT with that in liver and muscle to determine tissue-specific contributions to overall insulin resistance in obesity. **What are the epigenetic changes associated with insulin resistance in WAT, and are they reversible after weight loss?** The study identifies transcriptional changes after weight loss, but it does not explore whether these changes are associated with epigenetic modifications (e.g., DNA methylation, histone modifications). Investigating this could provide insights into the long-term effects of weight loss on insulin sensitivity. **Are there specific genetic**

or molecular markers that predict whether an individual will regain insulin sensitivity after weight loss? Some individuals may experience only partial recovery of insulin sensitivity after bariatric surgery. Identifying molecular predictors (e.g., gene expression profiles, transcript isoforms) could help personalize treatment approaches for obesity and diabetes. ##### b. [Optional] Devise a computational analysis strategy for (some of) the listed questions under 3a. **Epigenetic changes** Perform whole-genome bisulfite sequencing (WGBS) and ChIP-Seq for histone modifications in WAT **selective insulin resistance across tissues** Conduct multi-tissue RNAseq (WAT, liver, muscle, etc.)

Task4

```
sqrt(10)  # Square root of 10
```

```
## [1] 3.162278
```

```
log2(32)  # Log base 2 of 32
```

```
## [1] 5
```

```
sum(1:1000)  # Sum of numbers from 1 to 1000
```

```
## [1] 500500
```

```
sum(seq(2, 1000, by = 2))  # Sum of even numbers from 2 to 1000
```

```
## [1] 250500
```

```
choose(100, 2)  # Pairwise comparisons of 100 genes
```

```
## [1] 4950
```

```
choose(100, 3)  # Arranging 100 genes in triple
```

```
## [1] 161700
```

Task5

```
data("CO2")
?CO2 # description of CO2 dataset
mean(CO2$uptake[CO2$Type == "Quebec"])
```

```
## [1] 33.54286
```

```
median(CO2$uptake[CO2$Type == "Quebec"])
```

```
## [1] 37.15
```

```r
mean(CO2$uptake[CO2$Type == "Mississippi"])
```

```
## [1] 20.88333
```

```r
median(CO2$uptake[CO2$Type == "Mississippi"])
```

```
## [1] 19.3
```

Task6

```r
ratio_mean_median <- function(x) {
    mean(x) / median(x)
}
```

1. a function that calculates the ratio of the mean and the median of a given vector. This is a helpful measure to detect data with outlying values.

```r
trimmed_mean <- function(x) {
    mean(x[x != min(x) & x != max(x)])
}
```

2. a function that ignores the lowest and the highest value from a given vector and calculate the mean.

3. Read about piping from here:https://r4ds.had.co.nz/pipes.html#pipes (you don't have to learn everything, a basic understanding of the usage is enough). Write a short (max. 300 characters, no spaces) explanation of why, how, and when not to use pipes. Pipes (%>%) simplify sequential operations, improving readability. Use them for short, linear transformations. Avoid pipes for long sequences (10+ steps), multiple inputs/outputs, or complex dependencies. #### 4. Write a short explanation (max. 300 characters, no spaces) of why they could be useful in your work. The apply family streamlines data analysis by replacing explicit loops, making code more readable and efficient. In RNAseq and genomics, they help process large datasets, compute statistics across genes/samples, and apply transformations efficiently, crucial for differential expression and enrichment analyses.

Task7

```r
# read magic_guys.csv and check the dataflame structure
setwd("~/Google Drive/My Drive/0_Iden2/KI-UT_doctoral_course")
library(ggplot2)
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
```

```
df <- read.csv("magic_guys.csv")
str(df)
```

```
## 'data.frame':    100 obs. of  4 variables:
##  $ uniqId : chr  "p1" "p2" "p3" "p4" ...
##  $ species: chr  "jedi" "jedi" "jedi" "jedi" ...
##  $ length : num  175 252 230 176 213 ...
##  $ weight : num  71.3 70.8 70.7 80.4 82 64.2 71.8 75 74.3 72.8 ...
```

```
summary(df)
```

```
##     uniqId             species              length          weight
##  Length:100         Length:100         Min.   : 60.1   Min.   :35.20
##  Class :character   Class :character   1st Qu.:111.3   1st Qu.:48.33
##  Mode  :character   Mode  :character   Median :138.4   Median :58.65
##                                        Mean   :147.8   Mean   :59.94
##                                        3rd Qu.:178.7   3rd Qu.:72.03
##                                        Max.   :272.9   Max.   :82.80
```

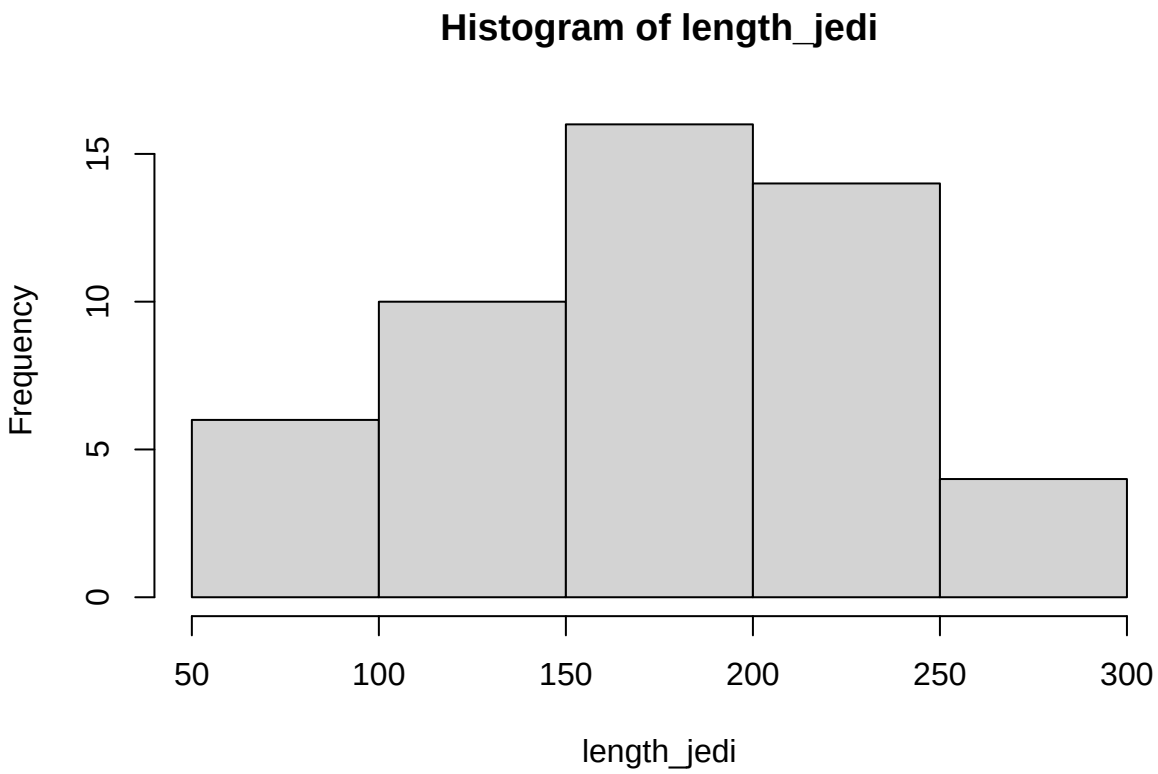
```
unique(df$species)
```

```
## [1] "jedi" "sith"
```

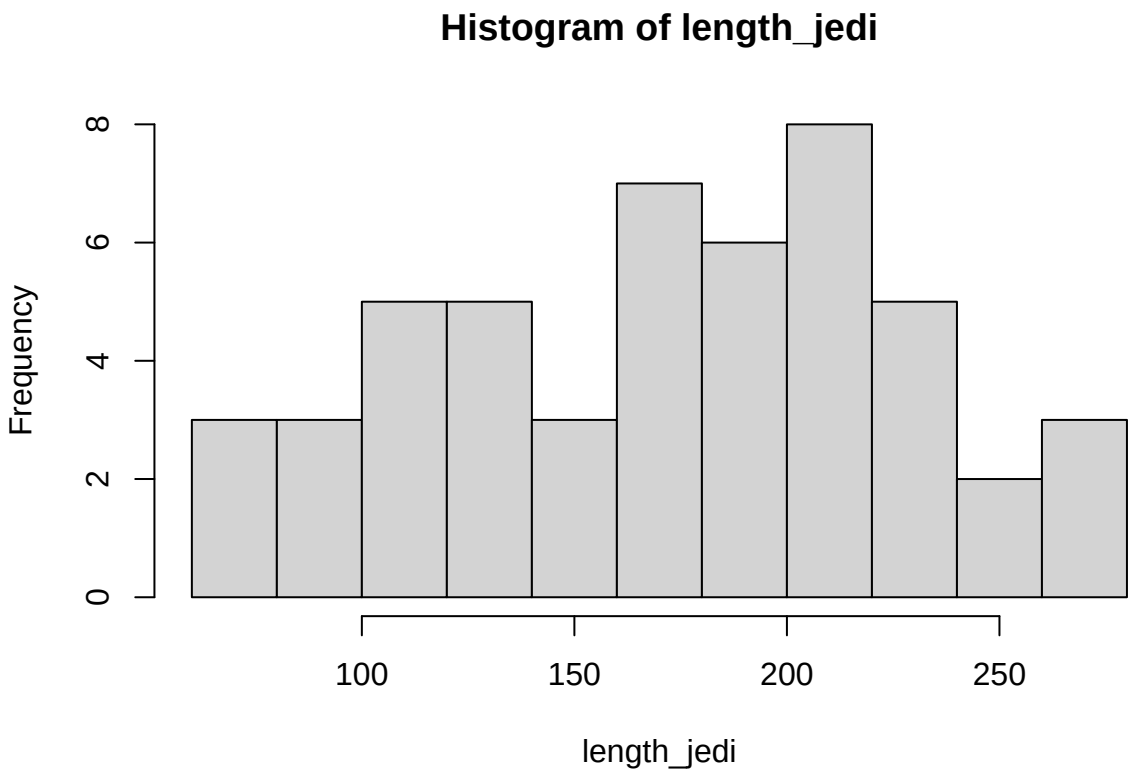
1. Compare the distributions of the body heights of the two species from the 'magic_guys.csv' dataset graphically

a. using the basic 'hist' function as well as 'ggplot' and 'geom_histogram' functions from the ggplot2 package. Optimize the plots for example by trying several different 'breaks'. Note that ggplot2-based functions give you many more options for changing the visualization parameters, try some of them. hist

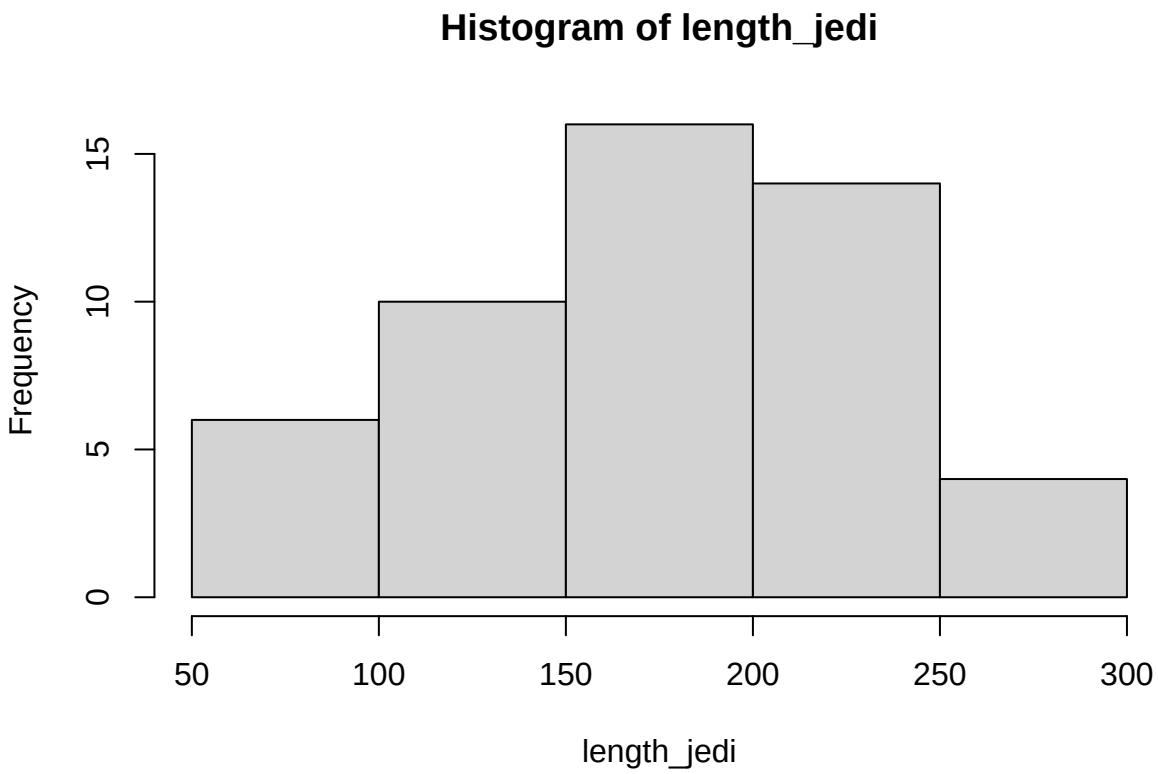
```
length_jedi <- df$length[df$species == "jedi"]
length_sith <- df$length[df$species == "sith"]
hist(length_jedi)
```



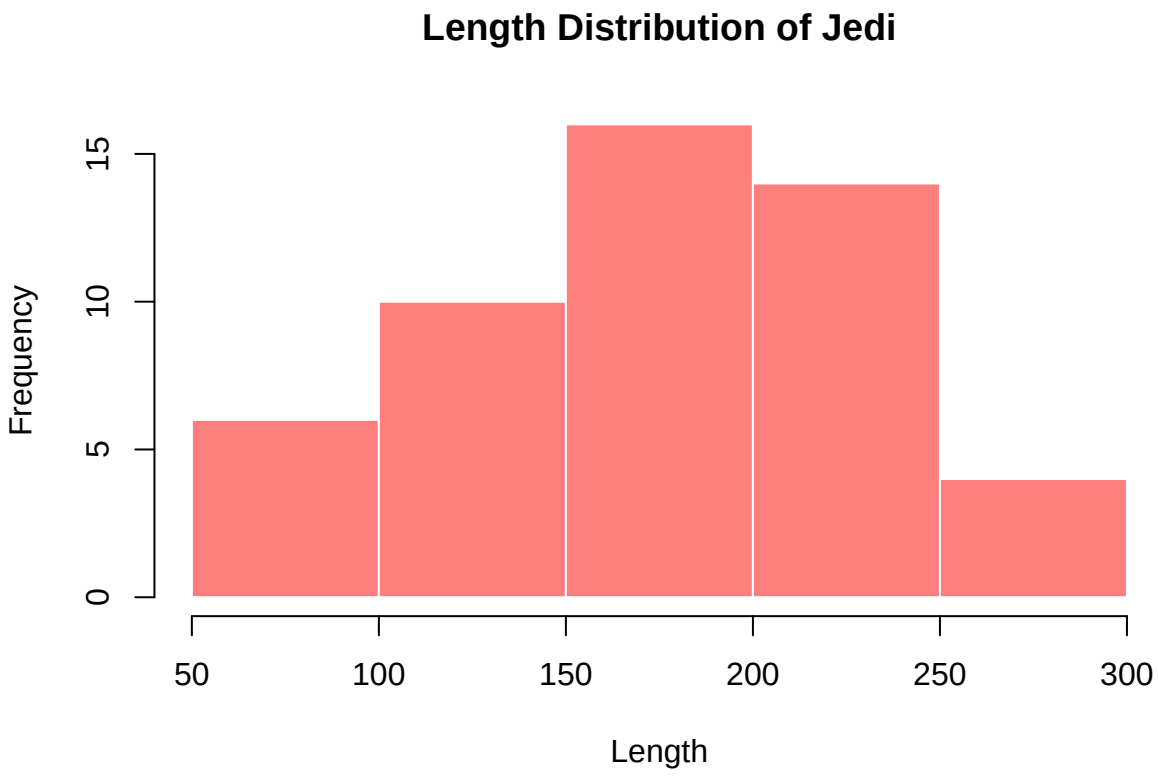

```
hist(length_jedi, breaks = 10) # Set bin numver to ten. Does not look good.
```



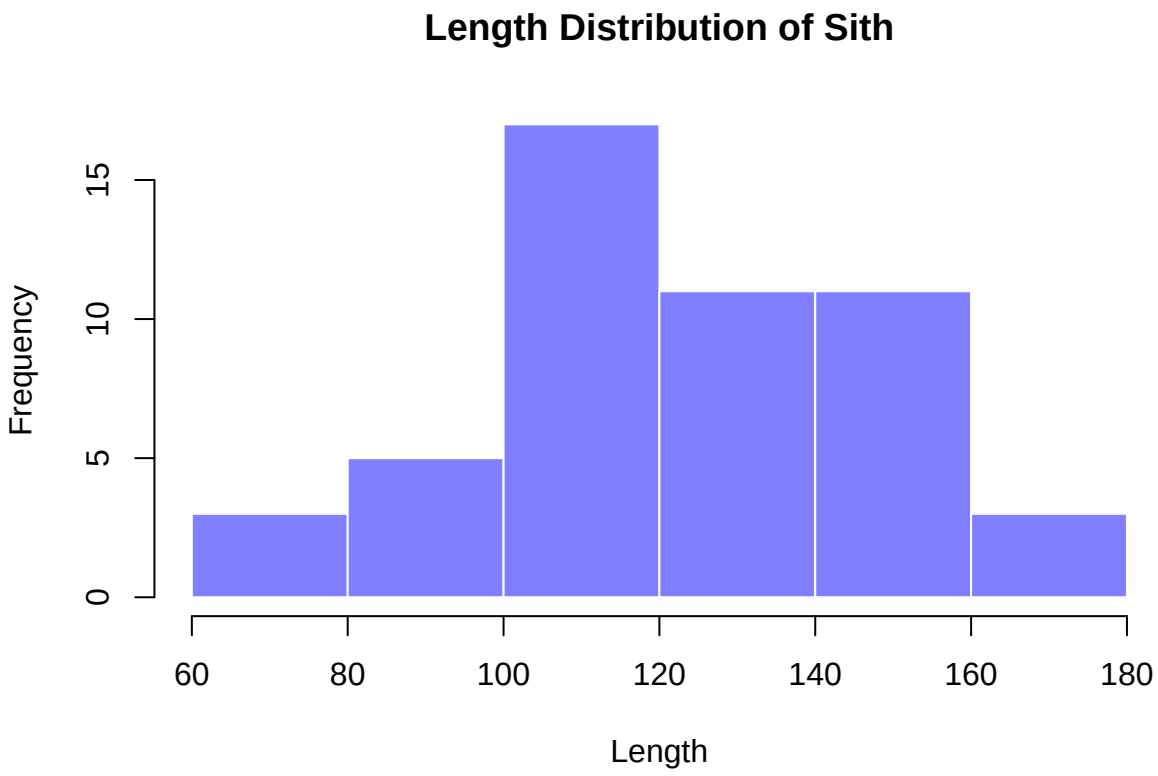

```
hist(length_jedi, breaks = "FD") # Freedman-Diaconis's rule. Consider data variation, so can choose bet
```



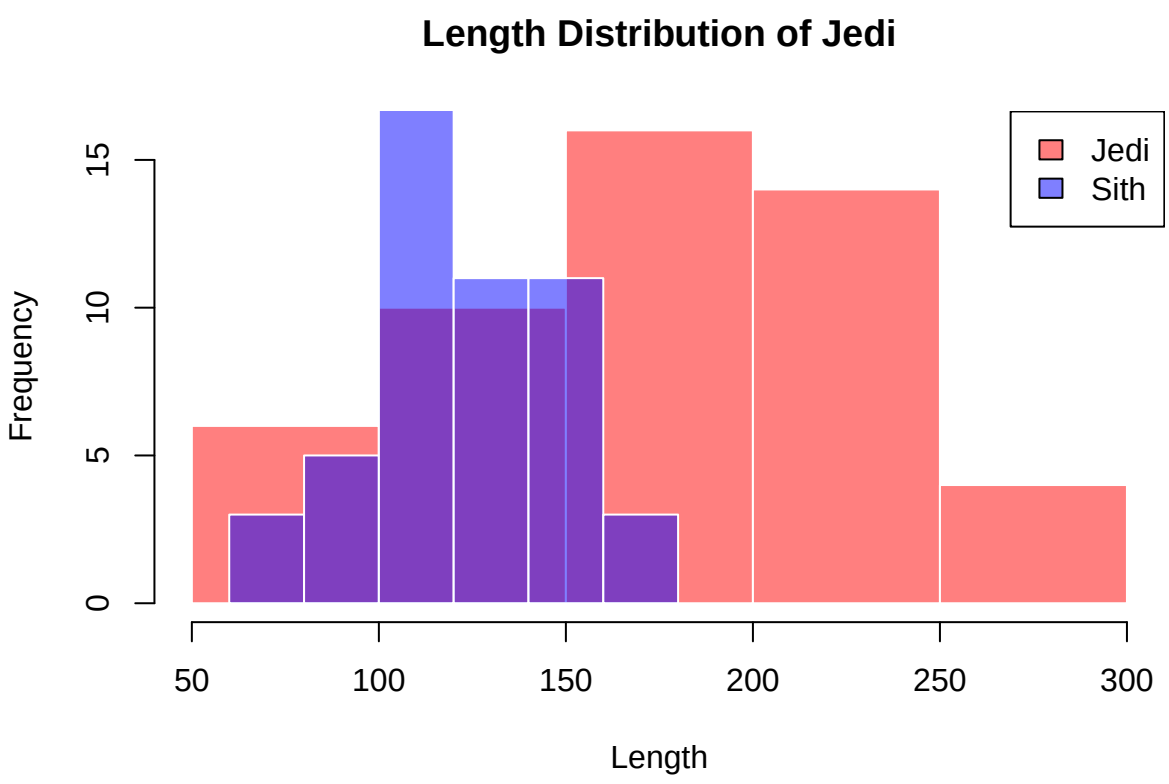

```
hist(length_jedi, col = rgb(1, 0, 0, 0.5), main = "Length Distribution of Jedi",
xlab = "Length", border = "white") # Changed color and added title.
```




```
hist(length_sith, col = rgb(0, 0, 1, 0.5), main = "Length Distribution of Sith", xlab = "Length", borde
```

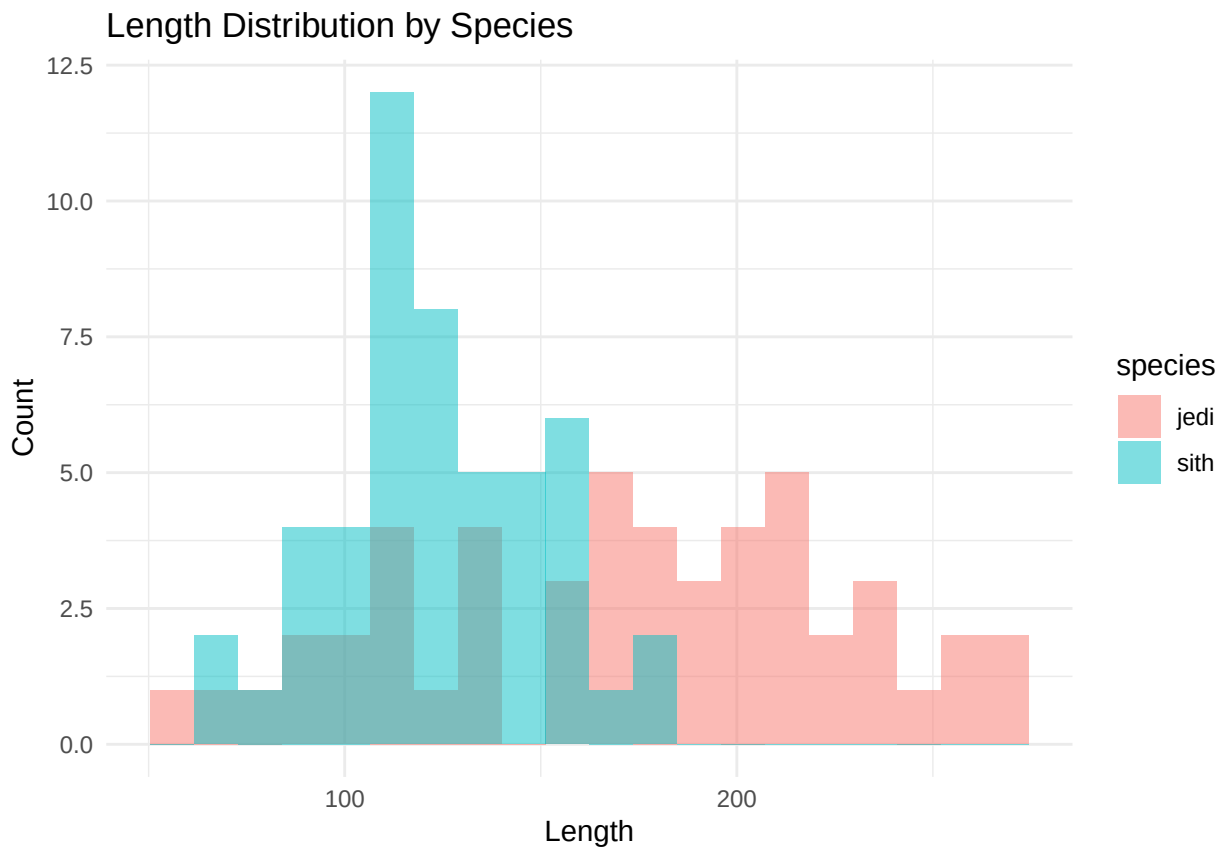



```r
# add two datasets by using same bin number (failed because bin is diffferent)
hist(length_jedi, breaks = 7, col = rgb(1, 0, 0, 0.5), main = "Length Distribution of Jedi",
xlab = "Length", border = "white")
hist(length_sith, breaks = 7, col = rgb(0, 0, 1, 0.5), main = "Length Distribution of Sith",
xlab = "Length", add = TRUE, border = "white") # add histogram to the histgram generated before.
legend("topright", legend = c("Jedi", "Sith"), fill = c(rgb(1, 0, 0, 0.5), rgb(0, 0, 1, 0.5))) # add ca
```

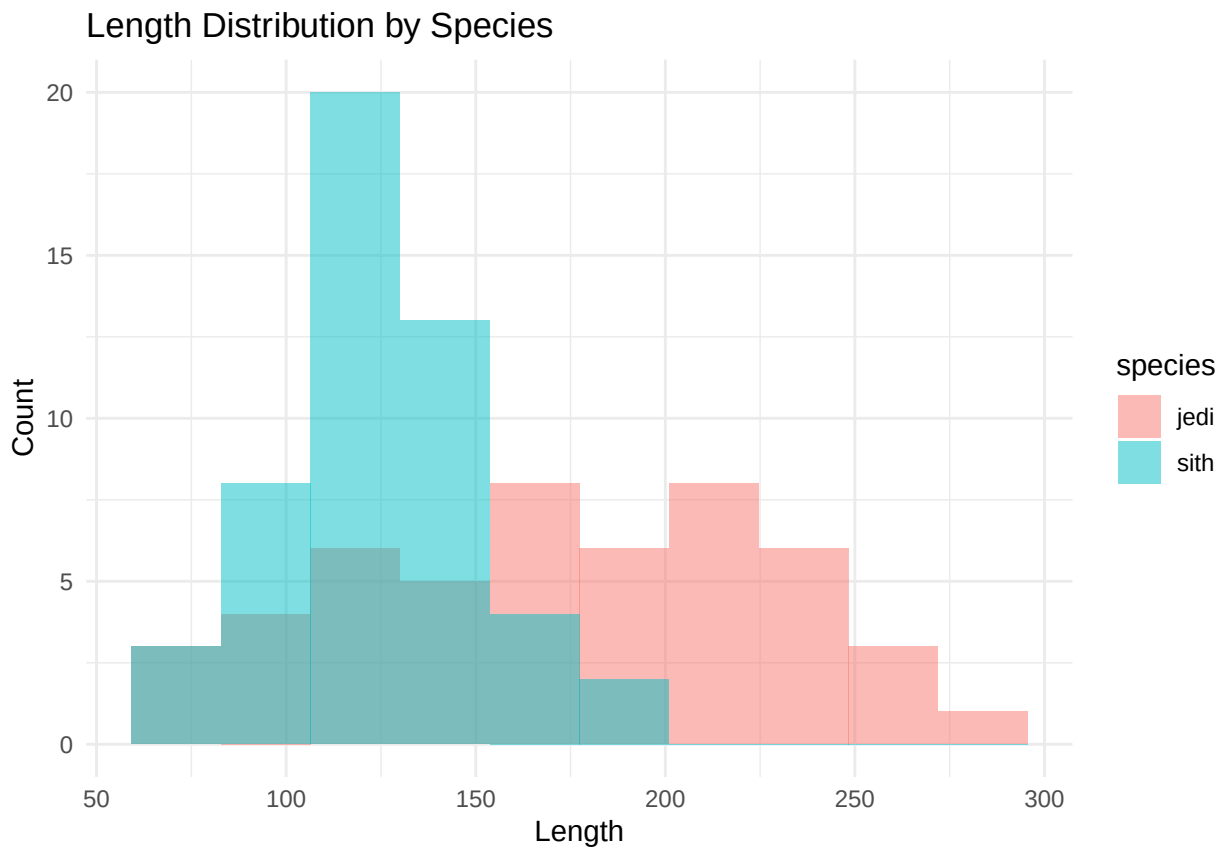


ggplot and geom_histogram (ggplot2)

```
ggplot(df, aes(x = length, fill = species)) +
geom_histogram(alpha = 0.5, position = "identity", bins = 20) +
labs(title = "Length Distribution by Species", x = "Length", y = "Count") +
theme_minimal() # bins=20
```

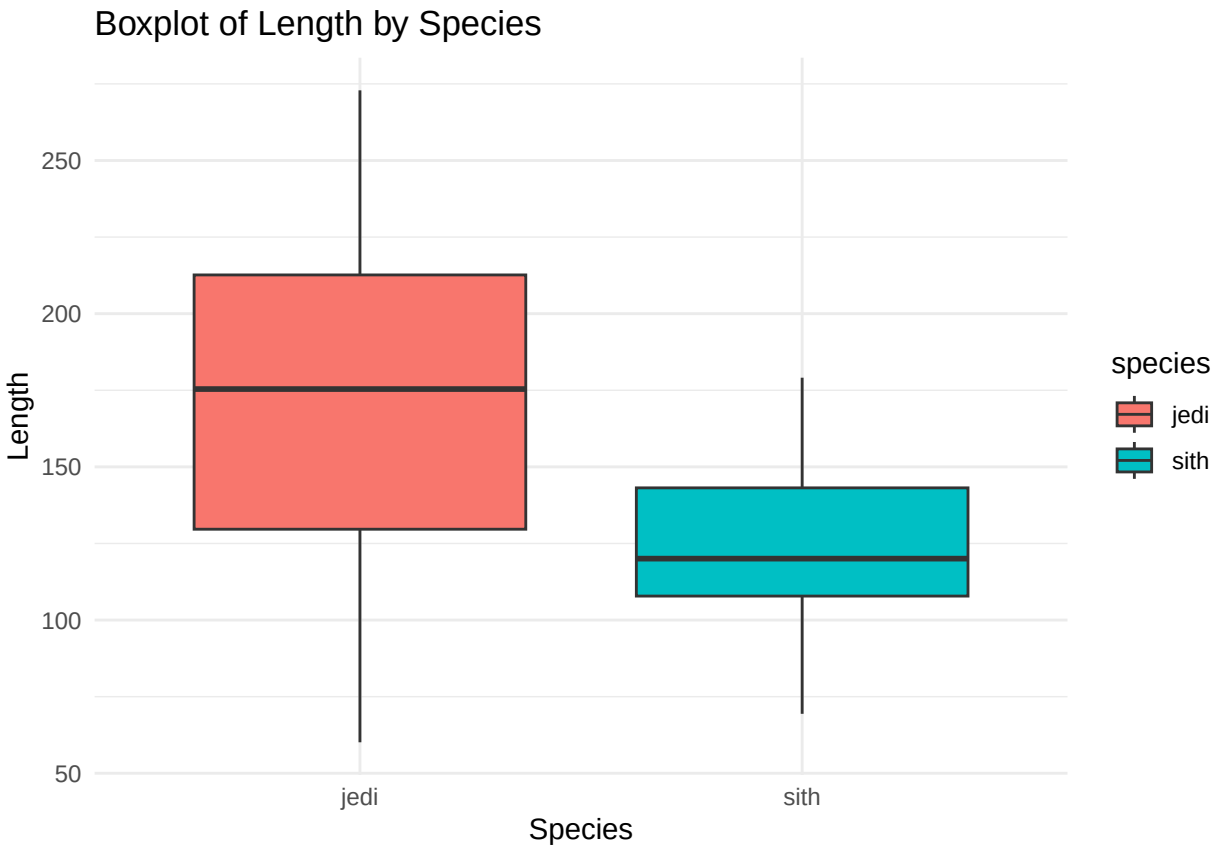



```
ggplot(df, aes(x = length, fill = species)) +
geom_histogram(alpha = 0.5, position = "identity", bins = 10) +
labs(title = "Length Distribution by Species", x = "Length", y = "Count") +
theme_minimal() # bins=10. Looks smoother than bins=20.
```



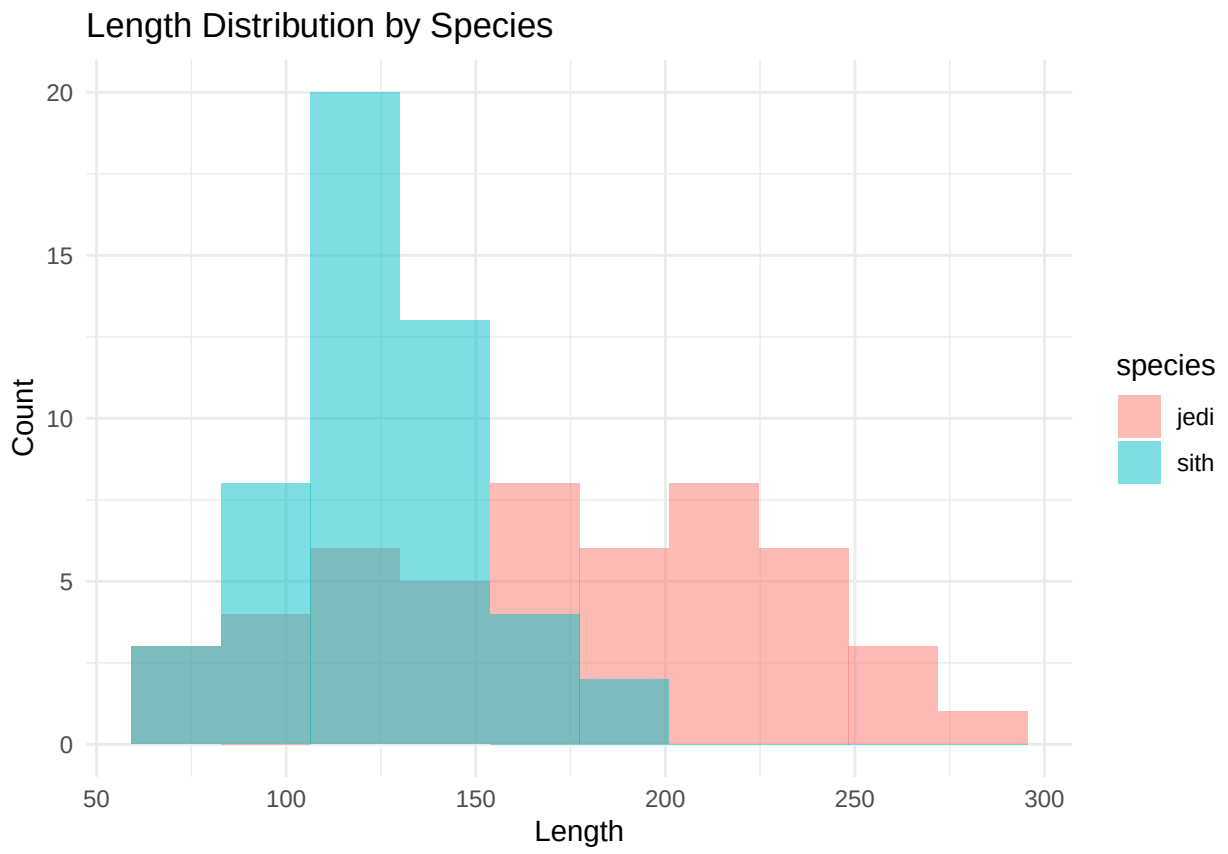
b. Do the same comparison as in a. but with boxplots. If you want to use the ggplot2-package, use the functions 'ggplot' and 'geom_boxplot'

```
ggplot(df, aes(x = species, y = length, fill = species)) +
geom_boxplot() +
labs(title = "Boxplot of Length by Species", x = "Species", y = "Length") +
theme_minimal()
```



c. Save the plots with the 'png', 'pdf', and 'svg' formats. In which situation would you use which file format?

```
# ggsave function saves the most recently created plot, so I first ran ggplot and then saved the plots.
ggplot(df, aes(x = length, fill = species)) +
geom_histogram(alpha = 0.5, position = "identity", bins = 10) +
labs(title = "Length Distribution by Species", x = "Length", y = "Count") +
theme_minimal() # bins=10. Looks smoother than bins=20.
```




```
ggsave("histogram.png")
```

```
## Saving 6.5 x 4.5 in image
```

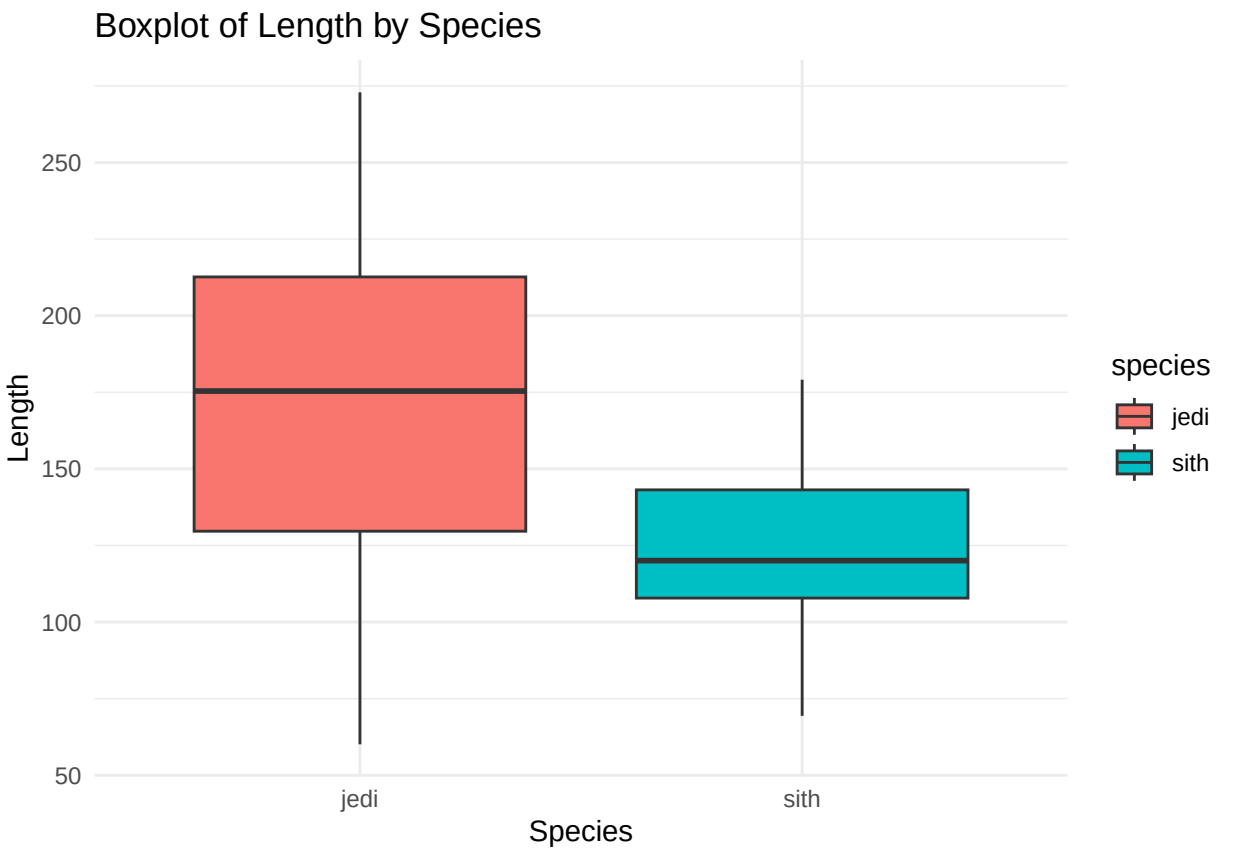
```
ggsave("histogram.pdf")
```

```
## Saving 6.5 x 4.5 in image
```

```
ggsave("histogram.svg")
```

```
## Saving 6.5 x 4.5 in image
```

```
ggplot(df, aes(x = species, y = length, fill = species)) +
geom_boxplot() +
labs(title = "Boxplot of Length by Species", x = "Species", y = "Length") +
theme_minimal()
```




```
ggsave("boxplot.png")
```

```
## Saving 6.5 x 4.5 in image
```

```
ggsave("boxplot.pdf")
```

```
## Saving 6.5 x 4.5 in image
```

```
ggsave("boxplot.svg")
```

```
## Saving 6.5 x 4.5 in image
```

Table 1: Comparison of File Formats for ggplot2

Format	Best.For	Pros	Cons
PNG	Web, presentations, quick sharing	Good balance of quality and file size, transparency support	Resolution-dependent, loses quality when resized
PDF	Printing, scientific papers, LaTeX	High-quality, vector format	Large file sizes, not ideal for web
SVG	Web graphics, scalable vector editing	Infinite scalability, editable	Large file size for complex plots, limited support in some platforms

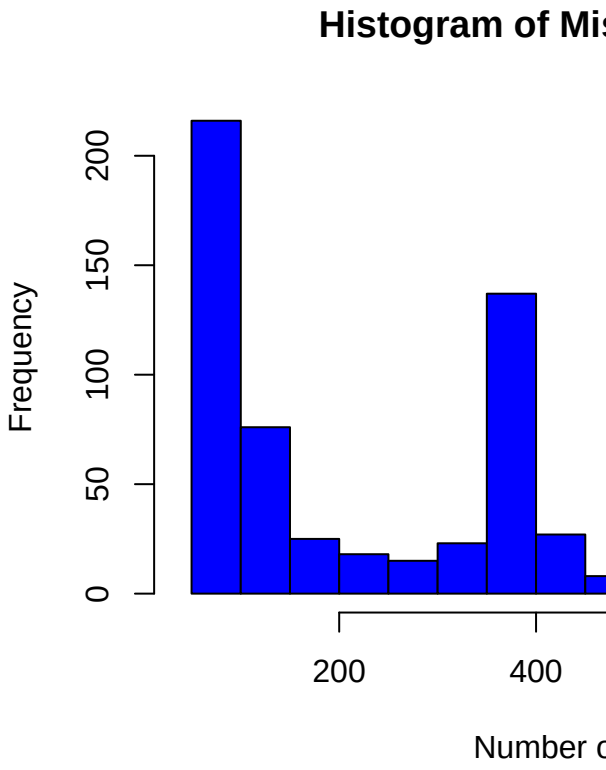
2. Load the gene expression data matrix from the 'microarray_data.tab' dataset provided in the shared folder, it is a big tabular separated matrix.

```
microarray <- read.table("microarray_data.tab", header = TRUE, sep = "\t") # There's no row name in col
dim(microarray) # (row number, column number)
```

a. How big is the matrix in terms of rows and columns?

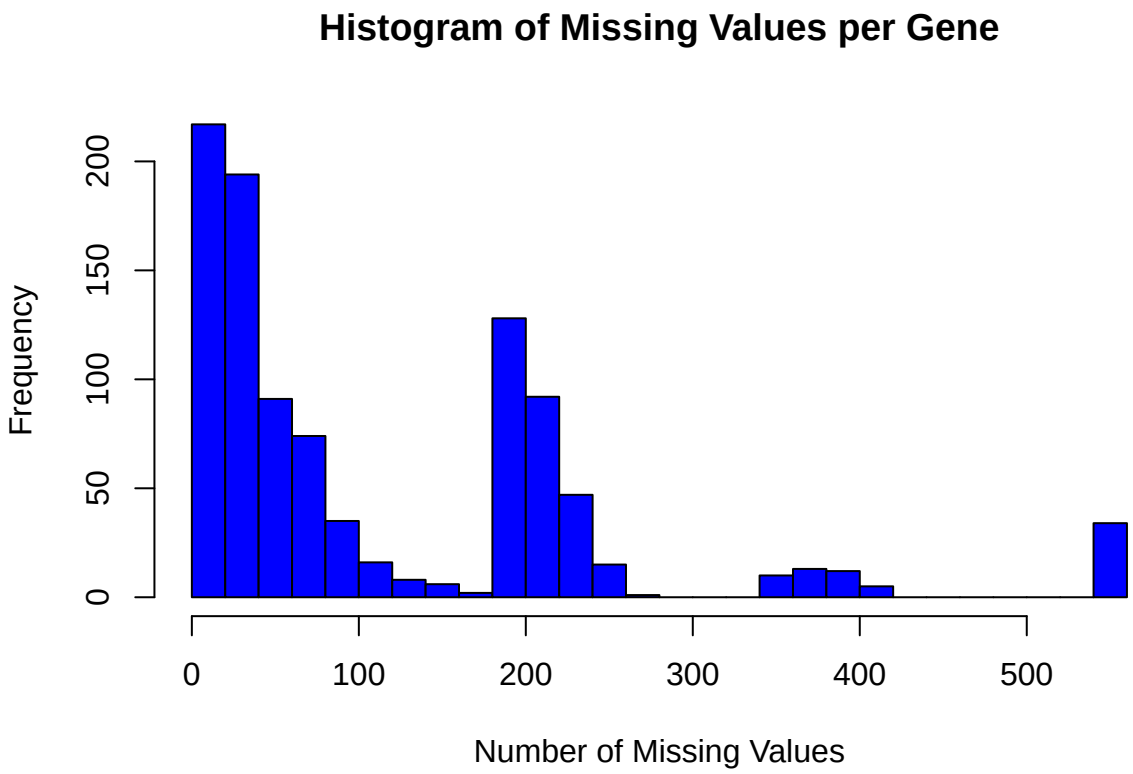
```
## [1]  553 1000
```

```
library(pheatmap) # Maybe don't need to load this package. is.na() is a standard function of R.
missing_values <- rowSums(is.na(microarray)) # Count missing values.
hist(missing_values, breaks = 20, main = "Histogram of Missing Values per Gene",
xlab = "Number of Missing Values", col = "blue") # Histogram of missing values. Generated using hist fu
```



b. Count the missing values per gene and visualize this result.

```
# I'm not sure, but the column name (g1, g2, ...) means gene name? If that's true, I must count missing
missing_values_col <- colSums(is.na(microarray))
hist(missing_values_col, breaks = 20, main = "Histogram of Missing Values per Gene",
xlab = "Number of Missing Values", col = "blue")
```



c. Find the genes for which there are more than X% (X=10%, 20%, 50%) missing values.

```
total_samples <- ncol(microarray)
print(total_samples)
```

```
## [1] 1000
```

```
threshold_10 <- total_samples * 0.1
threshold_20 <- total_samples * 0.2
threshold_50 <- total_samples * 0.5
genes_10 <- rownames(microarray)[missing_values > threshold_10]
genes_20 <- rownames(microarray)[missing_values > threshold_20]
genes_50 <- rownames(microarray)[missing_values > threshold_50]
print(genes_10)
```

```
##   [1] "2"   "5"   "7"   "13"  "17"  "18"  "19"  "21"  "24"  "25"  "26"  "27"
##  [13] "28"  "29"  "30"  "31"  "32"  "33"  "34"  "35"  "36"  "37"  "38"  "39"
##  [25] "40"  "41"  "42"  "43"  "44"  "45"  "46"  "47"  "48"  "49"  "50"  "51"
##  [37] "52"  "53"  "54"  "55"  "56"  "57"  "58"  "59"  "60"  "61"  "62"  "63"
##  [49] "64"  "65"  "66"  "67"  "68"  "69"  "70"  "71"  "72"  "75"  "76"  "77"
##  [61] "78"  "79"  "80"  "81"  "82"  "83"  "84"  "85"  "86"  "87"  "88"  "89"
##  [73] "90"  "91"  "92"  "93"  "94"  "95"  "96"  "97"  "98"  "99"  "100" "101"
##  [85] "102" "103" "104" "105" "106" "107" "108" "109" "110" "111" "112" "113"
##  [97] "115" "116" "118" "119" "120" "121" "122" "123" "124" "126" "127" "128"
## [109] "129" "130" "131" "132" "133" "134" "135" "136" "137" "138" "139" "140"
## [121] "141" "142" "143" "144" "145" "146" "147" "148" "149" "150" "151" "152"
```

```
## [133] "153" "154" "155" "157" "158" "159" "160" "161" "162" "163" "164" "165"
## [145] "166" "168" "169" "170" "171" "172" "173" "174" "176" "177" "178" "179"
## [157] "182" "183" "187" "193" "195" "202" "203" "209" "214" "215" "216" "217"
## [169] "219" "226" "237" "238" "240" "241" "243" "249" "250" "251" "260" "262"
## [181] "263" "264" "270" "273" "276" "277" "280" "282" "285" "286" "288" "290"
## [193] "291" "294" "295" "303" "304" "308" "311" "312" "320" "322" "323" "327"
## [205] "328" "331" "332" "333" "335" "336" "337" "338" "339" "340" "341" "342"
## [217] "343" "344" "345" "346" "347" "348" "349" "350" "351" "352" "353" "354"
## [229] "355" "356" "357" "358" "359" "360" "361" "362" "363" "364" "365" "366"
## [241] "367" "368" "369" "371" "372" "373" "374" "375" "376" "377" "379" "382"
## [253] "383" "386" "387" "388" "389" "390" "391" "392" "393" "394" "395" "396"
## [265] "400" "401" "406" "417" "418" "419" "420" "424" "427" "428" "432" "435"
## [277] "436" "437" "439" "440" "441" "442" "443" "444" "445" "446" "447" "448"
## [289] "449" "450" "451" "452" "453" "455" "456" "457" "458" "461" "462" "465"
## [301] "469" "478" "479" "483" "493" "495" "498" "500" "502" "503" "504" "505"
## [313] "506" "509" "511" "513" "514" "515" "516" "517" "518" "519" "520" "521"
## [325] "522" "523" "527" "529" "534" "536" "537" "541" "546" "547" "548" "550"
## [337] "551"
```

```
print(genes_20)
```

```
##   [1] "2"   "7"   "17"  "18"  "19"  "21"  "24"  "25"  "26"  "27"  "28"  "29"
##  [13] "30"  "31"  "32"  "33"  "34"  "35"  "36"  "37"  "38"  "39"  "40"  "41"
##  [25] "42"  "43"  "44"  "45"  "46"  "47"  "48"  "49"  "50"  "51"  "52"  "53"
##  [37] "54"  "55"  "56"  "57"  "58"  "59"  "60"  "61"  "62"  "63"  "64"  "65"
##  [49] "66"  "67"  "68"  "69"  "70"  "71"  "72"  "75"  "76"  "77"  "78"  "79"
##  [61] "80"  "81"  "82"  "83"  "84"  "85"  "86"  "87"  "88"  "89"  "90"  "91"
##  [73] "92"  "93"  "94"  "95"  "96"  "97"  "98"  "99"  "100" "101" "102" "103"
##  [85] "104" "105" "106" "107" "109" "110" "112" "113" "115" "116" "118" "119"
##  [97] "120" "121" "122" "123" "124" "126" "127" "128" "129" "130" "132" "133"
## [109] "134" "135" "136" "137" "138" "139" "140" "141" "142" "143" "144" "145"
## [121] "146" "147" "148" "149" "150" "151" "152" "153" "154" "155" "157" "158"
## [133] "159" "160" "161" "162" "163" "164" "165" "166" "168" "169" "170" "171"
## [145] "172" "173" "174" "177" "182" "183" "193" "195" "209" "214" "215" "237"
## [157] "241" "250" "251" "282" "285" "308" "323" "328" "335" "336" "337" "338"
## [169] "339" "340" "341" "342" "343" "344" "345" "346" "347" "348" "349" "350"
## [181] "351" "352" "353" "354" "355" "356" "357" "358" "359" "360" "361" "362"
## [193] "363" "364" "365" "366" "367" "368" "369" "371" "373" "377" "382" "395"
## [205] "396" "401" "418" "424" "427" "428" "435" "436" "437" "441" "443" "444"
## [217] "445" "446" "447" "448" "450" "451" "455" "461" "498" "506" "514" "515"
## [229] "519" "520" "522" "534" "536" "541" "546" "548"
```

```
print(genes_50)
```

```
## [1] "17"  "57"  "135" "241" "250" "335" "340" "541"
```

```
# Again, I'm not sure, but the column name (g1, g2, ...) means gene name? If that's true, I must count
total_samples_row <- nrow(microarray)
print(total_samples_row)
```

```
## [1] 553
```

```r
threshold_10_row <- total_samples_row * 0.1
threshold_20_row <- total_samples_row * 0.2
threshold_50_row <- total_samples_row * 0.5
genes_10_row <- colnames(microarray)[missing_values_col > threshold_10_row]
genes_20_row <- colnames(microarray)[missing_values_col > threshold_20_row]
genes_50_row <- colnames(microarray)[missing_values_col > threshold_50_row]
print(genes_10_row)
```

```
##   [1] "g1"   "g2"   "g3"   "g4"   "g5"   "g10"  "g11"  "g12"  "g14"  "g15"
##  [11] "g16"  "g18"  "g21"  "g22"  "g23"  "g24"  "g25"  "g26"  "g28"  "g29"
##  [21] "g35"  "g36"  "g37"  "g38"  "g39"  "g40"  "g41"  "g42"  "g44"  "g45"
##  [31] "g46"  "g47"  "g48"  "g49"  "g50"  "g51"  "g52"  "g53"  "g54"  "g55"
##  [41] "g56"  "g57"  "g58"  "g59"  "g60"  "g61"  "g62"  "g63"  "g64"  "g65"
##  [51] "g66"  "g67"  "g68"  "g69"  "g70"  "g71"  "g72"  "g73"  "g74"  "g75"
##  [61] "g76"  "g77"  "g78"  "g79"  "g80"  "g81"  "g82"  "g83"  "g84"  "g85"
##  [71] "g86"  "g87"  "g88"  "g89"  "g90"  "g91"  "g92"  "g93"  "g94"  "g95"
##  [81] "g96"  "g97"  "g98"  "g99"  "g100" "g101" "g102" "g103" "g104" "g105"
##  [91] "g106" "g107" "g108" "g109" "g110" "g111" "g112" "g113" "g114" "g115"
## [101] "g116" "g117" "g118" "g119" "g120" "g130" "g131" "g132" "g133" "g134"
## [111] "g135" "g136" "g137" "g138" "g139" "g140" "g142" "g147" "g148" "g151"
## [121] "g152" "g153" "g154" "g155" "g156" "g157" "g158" "g159" "g160" "g165"
## [131] "g171" "g172" "g173" "g174" "g175" "g176" "g177" "g178" "g179" "g180"
## [141] "g194" "g196" "g200" "g204" "g210" "g221" "g223" "g233" "g239" "g241"
## [151] "g242" "g243" "g244" "g252" "g258" "g260" "g263" "g264" "g268" "g274"
## [161] "g280" "g281" "g284" "g285" "g286" "g287" "g288" "g290" "g292" "g294"
## [171] "g295" "g296" "g297" "g298" "g299" "g301" "g309" "g311" "g312" "g313"
## [181] "g314" "g315" "g316" "g320" "g321" "g322" "g323" "g324" "g327" "g329"
## [191] "g331" "g332" "g333" "g334" "g335" "g336" "g337" "g339" "g344" "g347"
## [201] "g348" "g351" "g352" "g353" "g354" "g355" "g356" "g357" "g358" "g359"
## [211] "g360" "g361" "g362" "g363" "g364" "g365" "g366" "g367" "g368" "g369"
## [221] "g370" "g372" "g374" "g377" "g378" "g379" "g380" "g381" "g382" "g383"
## [231] "g384" "g385" "g386" "g387" "g388" "g389" "g390" "g391" "g392" "g396"
## [241] "g400" "g401" "g402" "g403" "g404" "g405" "g406" "g407" "g408" "g409"
## [251] "g410" "g411" "g413" "g415" "g416" "g417" "g418" "g419" "g421" "g423"
## [261] "g425" "g429" "g430" "g431" "g432" "g433" "g434" "g435" "g436" "g437"
## [271] "g438" "g439" "g440" "g445" "g450" "g453" "g455" "g459" "g460" "g461"
## [281] "g462" "g463" "g464" "g465" "g466" "g467" "g468" "g469" "g470" "g476"
## [291] "g477" "g479" "g481" "g483" "g491" "g492" "g493" "g494" "g495" "g496"
## [301] "g497" "g498" "g499" "g500" "g502" "g507" "g510" "g513" "g514" "g515"
## [311] "g518" "g519" "g520" "g522" "g524" "g525" "g527" "g530" "g531" "g532"
## [321] "g533" "g534" "g535" "g536" "g537" "g538" "g539" "g540" "g541" "g544"
## [331] "g547" "g553" "g555" "g556" "g558" "g559" "g561" "g563" "g564" "g567"
## [341] "g569" "g570" "g571" "g572" "g573" "g574" "g575" "g576" "g577" "g578"
## [351] "g579" "g580" "g583" "g585" "g586" "g589" "g591" "g592" "g595" "g597"
## [361] "g599" "g607" "g610" "g611" "g612" "g613" "g614" "g615" "g616" "g617"
## [371] "g618" "g619" "g620" "g631" "g638" "g650" "g653" "g657" "g660" "g663"
## [381] "g666" "g669" "g672" "g681" "g689" "g691" "g694" "g696" "g700" "g707"
## [391] "g709" "g711" "g715" "g718" "g719" "g722" "g724" "g726" "g743" "g744"
## [401] "g747" "g748" "g749" "g751" "g752" "g753" "g754" "g755" "g756" "g757"
## [411] "g758" "g759" "g760" "g761" "g762" "g763" "g764" "g765" "g766" "g767"
## [421] "g768" "g769" "g770" "g776" "g781" "g782" "g783" "g784" "g785" "g786"
## [431] "g787" "g788" "g789" "g790" "g795" "g796" "g801" "g802" "g803" "g804"
## [441] "g805" "g806" "g807" "g808" "g809" "g810" "g812" "g814" "g818" "g820"
```

```
## [451] "g822" "g824" "g831" "g832" "g833" "g834" "g835" "g836" "g837" "g838"
## [461] "g839" "g840" "g843" "g849" "g850" "g851" "g854" "g861" "g862" "g863"
## [471] "g864" "g865" "g866" "g867" "g868" "g869" "g870" "g872" "g874" "g882"
## [481] "g884" "g891" "g892" "g893" "g894" "g895" "g896" "g897" "g898" "g899"
## [491] "g900" "g903" "g909" "g910" "g911" "g919" "g922" "g926" "g931" "g932"
## [501] "g933" "g934" "g935" "g936" "g937" "g938" "g939" "g940" "g945" "g947"
## [511] "g948" "g951" "g952" "g956" "g963" "g965" "g970" "g971" "g972" "g973"
## [521] "g974" "g975" "g976" "g977" "g978" "g979" "g980" "g985" "g989"
```

```
print(genes_20_row)
```

```
##   [1] "g1"   "g14"  "g18"  "g21"  "g22"  "g24"  "g25"  "g26"  "g29"  "g39"
##  [11] "g40"  "g41"  "g48"  "g51"  "g52"  "g53"  "g54"  "g55"  "g56"  "g57"
##  [21] "g58"  "g59"  "g60"  "g61"  "g62"  "g63"  "g64"  "g65"  "g66"  "g67"
##  [31] "g68"  "g69"  "g70"  "g71"  "g72"  "g73"  "g74"  "g75"  "g76"  "g77"
##  [41] "g78"  "g79"  "g80"  "g81"  "g82"  "g83"  "g84"  "g85"  "g86"  "g87"
##  [51] "g88"  "g89"  "g90"  "g91"  "g92"  "g93"  "g94"  "g95"  "g96"  "g97"
##  [61] "g98"  "g99"  "g100" "g101" "g102" "g103" "g104" "g105" "g106" "g107"
##  [71] "g108" "g109" "g110" "g111" "g112" "g113" "g114" "g115" "g116" "g117"
##  [81] "g118" "g119" "g120" "g130" "g131" "g132" "g133" "g134" "g135" "g136"
##  [91] "g137" "g138" "g139" "g140" "g142" "g147" "g148" "g151" "g152" "g153"
## [101] "g154" "g155" "g156" "g157" "g158" "g159" "g160" "g165" "g171" "g172"
## [111] "g173" "g174" "g175" "g176" "g177" "g178" "g179" "g180" "g196" "g200"
## [121] "g204" "g210" "g233" "g252" "g260" "g290" "g297" "g301" "g321" "g329"
## [131] "g332" "g333" "g334" "g335" "g344" "g351" "g352" "g353" "g354" "g355"
## [141] "g356" "g357" "g358" "g359" "g360" "g361" "g362" "g363" "g364" "g365"
## [151] "g366" "g367" "g368" "g369" "g370" "g379" "g381" "g382" "g383" "g384"
## [161] "g385" "g386" "g387" "g388" "g389" "g390" "g391" "g396" "g400" "g401"
## [171] "g402" "g403" "g404" "g405" "g406" "g407" "g408" "g409" "g410" "g415"
## [181] "g417" "g418" "g431" "g432" "g433" "g434" "g435" "g436" "g437" "g438"
## [191] "g439" "g440" "g445" "g450" "g455" "g461" "g462" "g463" "g464" "g465"
## [201] "g466" "g467" "g468" "g469" "g470" "g476" "g491" "g492" "g493" "g494"
## [211] "g495" "g496" "g497" "g498" "g499" "g500" "g510" "g513" "g515" "g518"
## [221] "g519" "g520" "g522" "g527" "g531" "g532" "g533" "g534" "g535" "g536"
## [231] "g537" "g538" "g539" "g540" "g544" "g547" "g558" "g561" "g569" "g570"
## [241] "g571" "g572" "g573" "g574" "g575" "g576" "g577" "g578" "g579" "g580"
## [251] "g583" "g585" "g586" "g591" "g599" "g611" "g612" "g613" "g614" "g615"
## [261] "g616" "g617" "g618" "g619" "g620" "g650" "g657" "g663" "g669" "g689"
## [271] "g691" "g694" "g744" "g751" "g752" "g753" "g754" "g755" "g756" "g757"
## [281] "g758" "g759" "g760" "g761" "g762" "g763" "g764" "g765" "g766" "g767"
## [291] "g768" "g769" "g770" "g781" "g782" "g783" "g784" "g785" "g786" "g787"
## [301] "g788" "g789" "g790" "g801" "g802" "g803" "g804" "g805" "g806" "g807"
## [311] "g808" "g809" "g810" "g814" "g831" "g832" "g833" "g834" "g835" "g836"
## [321] "g837" "g838" "g839" "g840" "g849" "g850" "g851" "g854" "g861" "g862"
## [331] "g863" "g864" "g865" "g866" "g867" "g868" "g869" "g870" "g872" "g891"
## [341] "g892" "g893" "g894" "g895" "g896" "g897" "g898" "g899" "g900" "g910"
## [351] "g919" "g926" "g931" "g932" "g933" "g934" "g935" "g936" "g937" "g938"
## [361] "g939" "g940" "g947" "g951" "g970" "g971" "g972" "g973" "g974" "g975"
## [371] "g976" "g977" "g978" "g979" "g980" "g985" "g989"
```

```
print(genes_50_row)
```

```
##  [1] "g18"  "g48"  "g55"  "g58"  "g60"  "g66"  "g73"  "g79"  "g83"  "g91"
```

```
## [11] "g93"  "g94"  "g99"  "g105" "g109" "g132" "g135" "g137" "g138" "g172"
## [21] "g260" "g290" "g301" "g329" "g352" "g355" "g362" "g363" "g368" "g383"
## [31] "g388" "g389" "g390" "g391" "g406" "g417" "g431" "g432" "g440" "g461"
## [41] "g462" "g498" "g519" "g527" "g531" "g532" "g538" "g572" "g575" "g576"
## [51] "g577" "g585" "g615" "g619" "g663" "g669" "g751" "g753" "g766" "g768"
## [61] "g788" "g802" "g804" "g838" "g851" "g854" "g864" "g892" "g893" "g898"
## [71] "g919" "g932" "g971" "g980"
```

```r
microarray_filled <- microarray
for (i in 1:nrow(microarray_filled)) {
microarray_filled[i, is.na(microarray_filled[i, ])] <- mean(microarray_filled[i, ], na.rm = TRUE)
} # Not good. Yieled more than 50 warnings.
```

d. Replace the missing values by the average expression value for the particular gene. (Note: Imputing data has to be used with caution!)

```
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
```

```
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
```

```
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
```

```
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
```

```
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
```

```
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
```

```
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
```

```
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
```

```
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
```

```
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
```

```
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
```

```
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
```

```
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
```

```
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
```

```
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
```

```
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
```

```
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
```

```
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
```

```
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
```

```
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
```

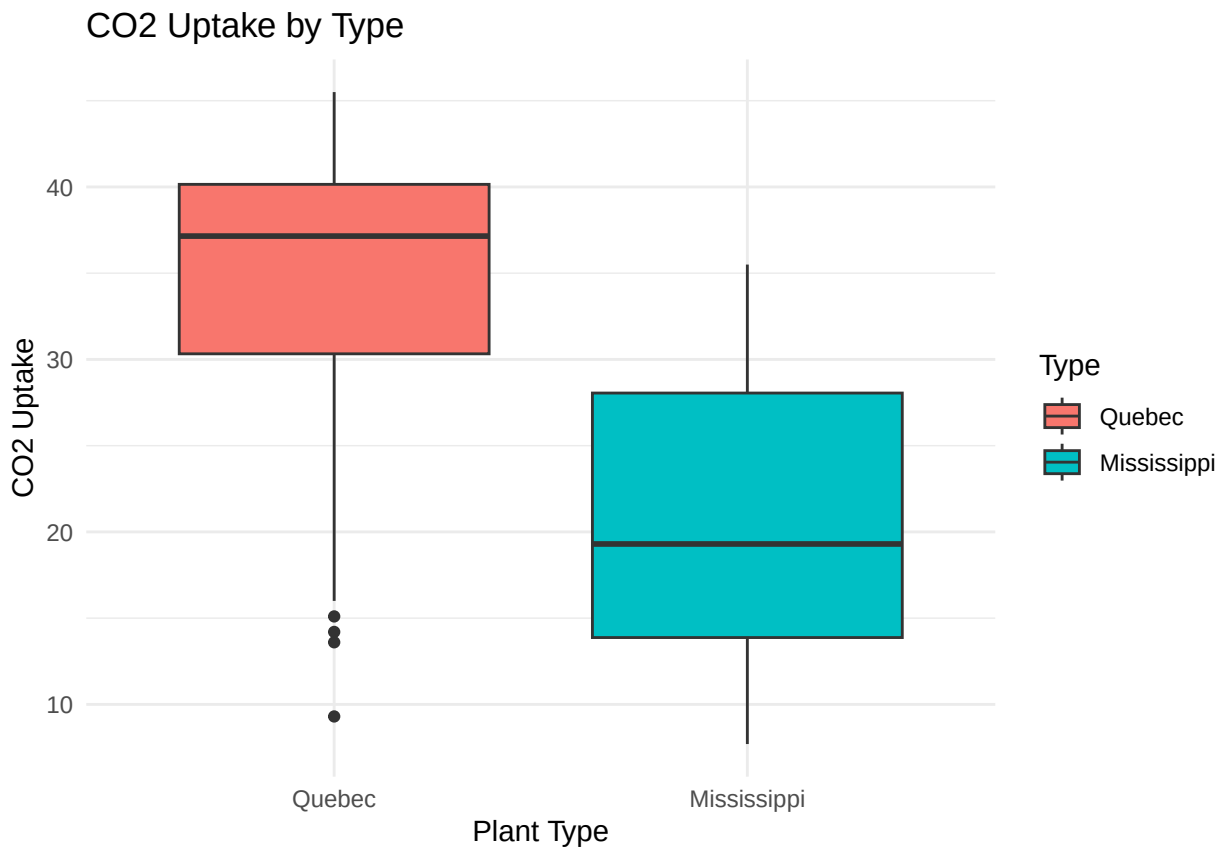
```
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
## Warning in mean.default(microarray_filled[i, ], na.rm = TRUE): argument is not
## numeric or logical: returning NA
```

```
# Fill missing values row by row using apply(), because the code above that uses for loop yielded more
microarray_filled <- microarray
microarray_filled <- t(apply(microarray_filled, 1, function(x) {
  x[is.na(x)] <- mean(x, na.rm = TRUE)
  return(x)
}))
```

I'm not sure if i succeeded in this. #### 3. Visualize the data in the CO2 dataset in a way that gives you

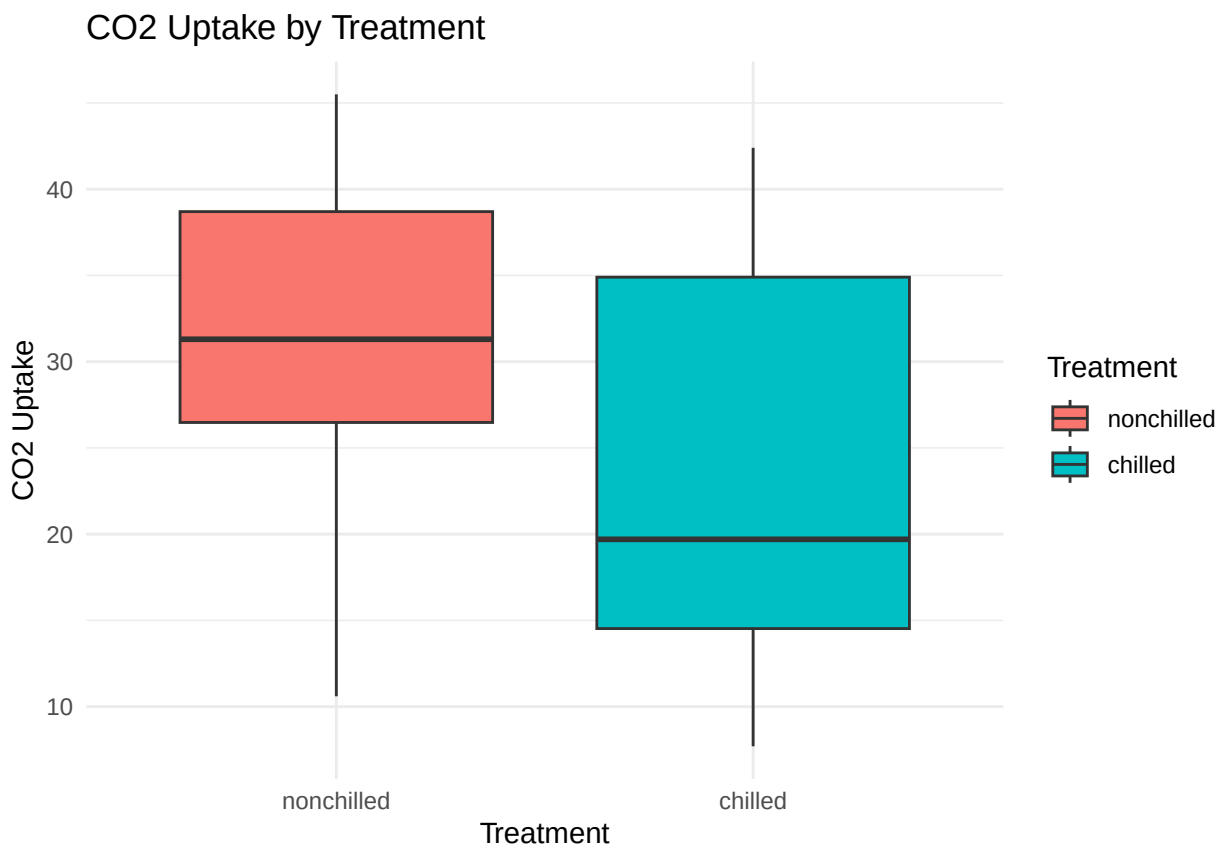
a deeper understanding of the data. What do you see?

```
# distribution of CO2 uptake by plant type (quebec or mississippi)
ggplot(CO2, aes(x = Type, y = uptake, fill = Type)) +
geom_boxplot() +
labs(title = "CO2 Uptake by Type", x = "Plant Type", y = "CO2 Uptake") +
theme_minimal()
```



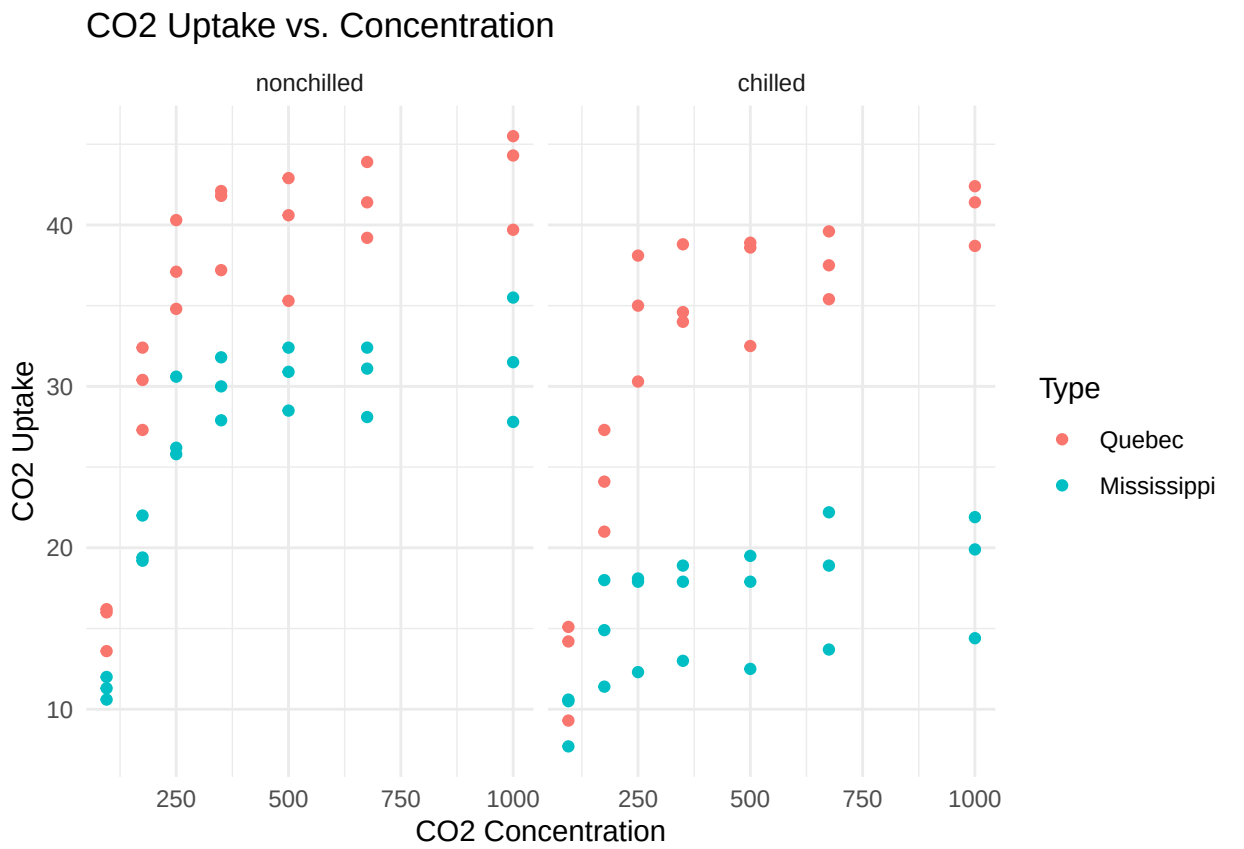
Quebec has higher CO2 uptake.

```
# distribution of CO2 uptake by treatment (chilled or nonchilled)
ggplot(CO2, aes(x = Treatment, y = uptake, fill = Treatment)) +
geom_boxplot() +
labs(title = "CO2 Uptake by Treatment", x = "Treatment", y = "CO2 Uptake") +
theme_minimal()
```



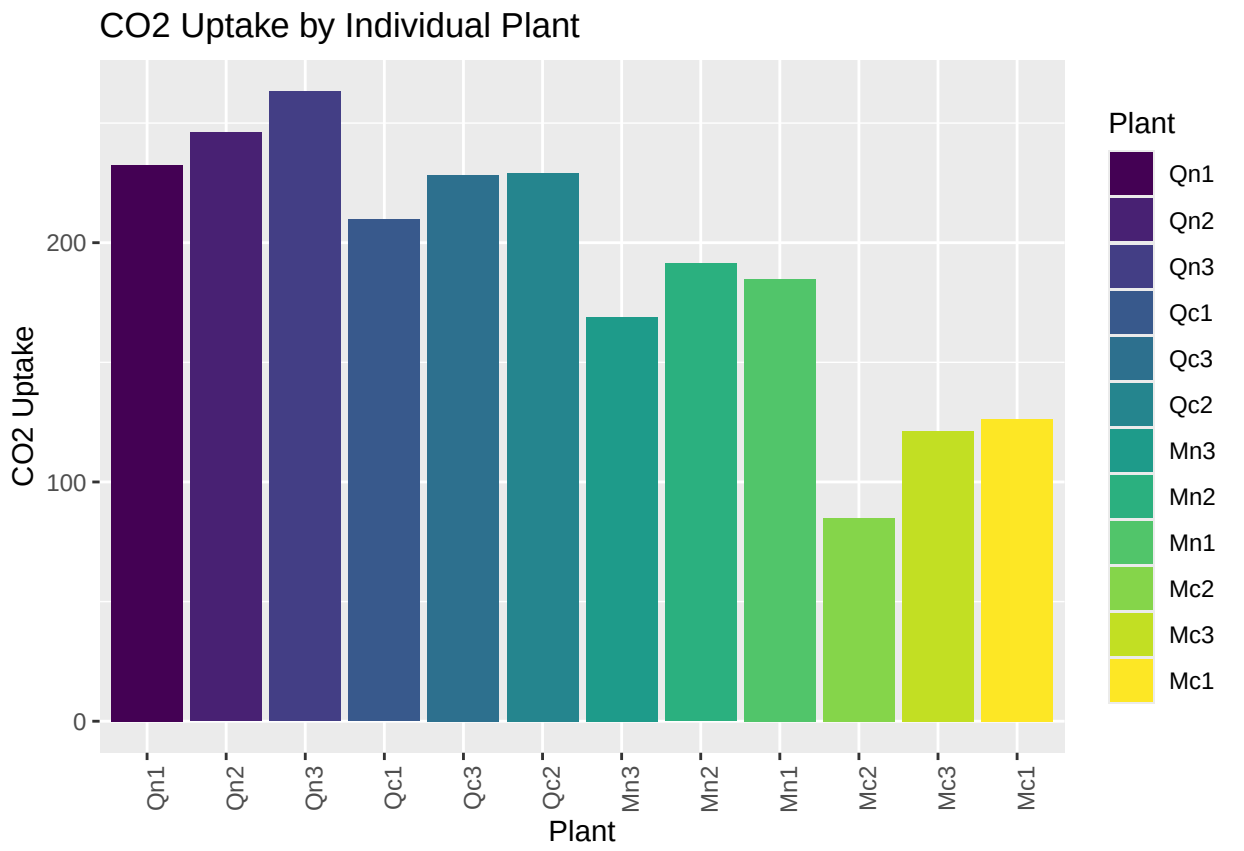
Nonchilled group has higher CO2 uptake.

```
# scatter plot of CO2 uptake and CO2 concentration. Chilled and nonchilled.
ggplot(CO2, aes(x = conc, y = uptake, color = Type)) +
geom_point() +
facet_wrap(~Treatment) +
labs(title = "CO2 Uptake vs. Concentration", x = "CO2 Concentration", y = "CO2 Uptake") +
theme_minimal()
```



Regardless of chilled/nonchilled and quebec/mississippi, Co2 uptake increases by ambient CO2 concentration. However, the difference of co2 uptake ability between quebec and mississippi broadens in chilled condition. Mississipi plants tend to be less tolerant to cold stimulus.

```
# CO2 uptake by each plant
ggplot(CO2, aes(x = Plant, y = uptake, fill = Plant)) +
  geom_bar(stat = "identity") +
  labs(title = "CO2 Uptake by Individual Plant", x = "Plant", y = "CO2 Uptake") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```



Task8

```
# I didn't have devtools, so I installed using install.packages("devtools")
library(devtools)
```

1. Install the Tidybiology package, which includes the data 'chromosome' and 'proteins'

```
## Loading required package: usethis
```

```
devtools::install_github("hirscheylab/tidybiology")
```

```
## Skipping install of 'tidybiology' from a github remote, the SHA1 (d03a810a) has not changed since las
##   Use 'force = TRUE' to force installation
```

```
library(tidybiology)

# read data
data(chromosome)
data(proteins)
```

```
head(chromosome)
```

a. Extract summary statistics (mean, median and maximum) for the following variables from the 'chromosome' data: variations, protein coding genes, and miRNAs. Utilize the tidyverse functions to make this as simply as possible.

```
## # A tibble: 6 x 14
##   id    length_mm basepairs variations protein_codinggenes pseudo_genes
##   <fct>     <dbl>     <dbl>      <dbl>               <int>        <int>
## 1 1            85 248956422   12151146                2058         1220
## 2 2            83 242193529   12945965                1309         1023
## 3 3            67 198295559   10638715                1078          763
## 4 4            65 190214555   10165685                 752          727
## 5 5            62 181538259    9519995                 876          721
## 6 6            58 170805979    9130476                1048          801
## # i 8 more variables: totallongnc_rna <int>, totalsmallnc_rna <int>,
## #   mi_rna <int>, r_rna <int>, sn_rna <int>, sno_rna <int>, miscnc_rna <int>,
## #   centromereposition_mbp <dbl>
```

```
str(chromosome)
```

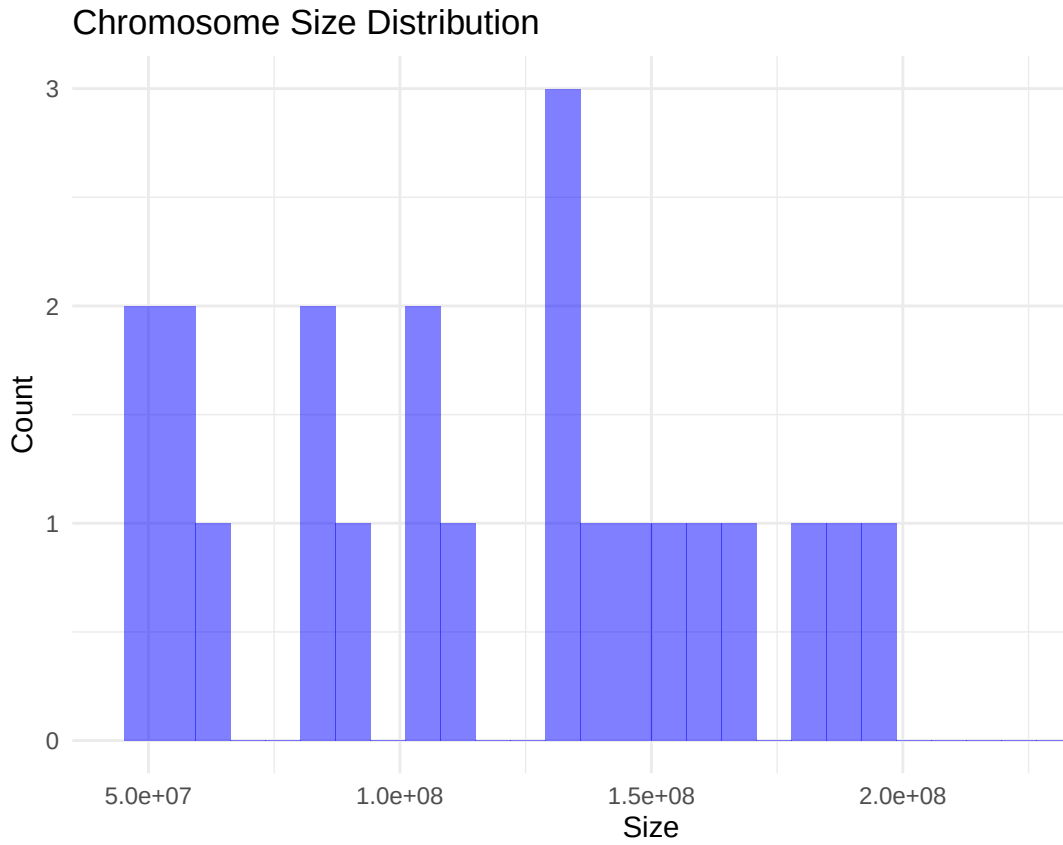
```
## tibble [24 x 14] (S3: tbl_df/tbl/data.frame)
##  $ id                    : Factor w/ 24 levels "1","2","3","4",..: 1 2 3 4 5 6 7 8 9 10 ...
##  $ length_mm             : num [1:24] 85 83 67 65 62 58 54 50 48 46 ...
##  $ basepairs             : num [1:24] 2.49e+08 2.42e+08 1.98e+08 1.90e+08 1.82e+08 ...
##  $ variations            : num [1:24] 12151146 12945965 10638715 10165685 9519995 ...
##  $ protein_codinggenes   : int [1:24] 2058 1309 1078 752 876 1048 989 677 786 733 ...
##  $ pseudo_genes          : int [1:24] 1220 1023 763 727 721 801 885 613 661 568 ...
##  $ totallongnc_rna       : int [1:24] 1200 1037 711 657 844 639 605 735 491 579 ...
##  $ totalsmallnc_rna      : int [1:24] 496 375 298 228 235 234 208 214 190 204 ...
##  $ mi_rna                : int [1:24] 134 115 99 92 83 81 90 80 69 64 ...
##  $ r_rna                 : int [1:24] 66 40 29 24 25 26 24 28 19 32 ...
##  $ sn_rna                : int [1:24] 221 161 138 120 106 111 90 86 66 87 ...
##  $ sno_rna               : int [1:24] 145 117 87 56 61 73 76 52 51 56 ...
##  $ miscnc_rna            : int [1:24] 192 176 134 104 119 105 143 82 96 89 ...
##  $ centromereposition_mbp: num [1:24] 125 93.3 91 50.4 48.4 61 59.9 45.6 49 40.2 ...
```

```
chromosome %>%
  summarise(
    mean_variations = mean(variations, na.rm = TRUE),
    median_variations = median(variations, na.rm = TRUE),
    max_variations = max(variations, na.rm = TRUE)
  )
```

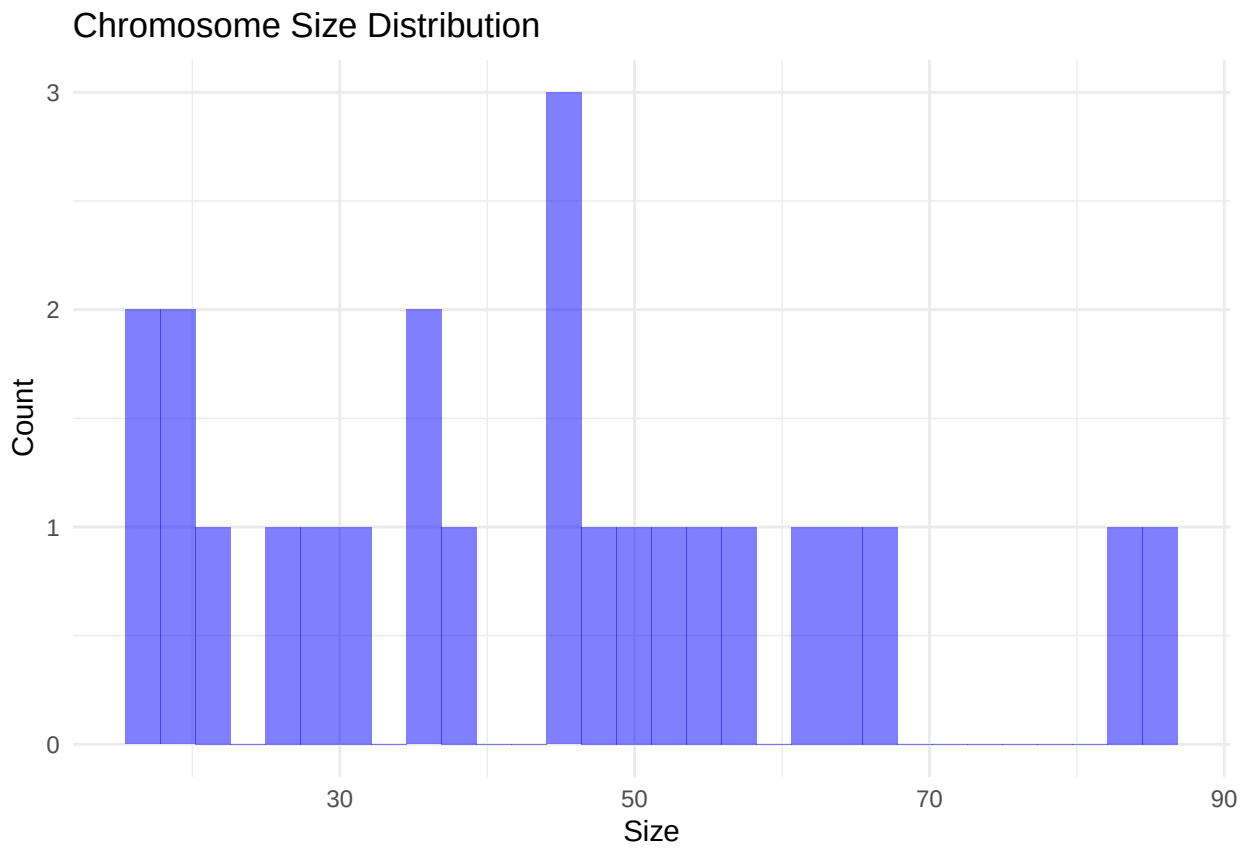
```
## # A tibble: 1 x 3
##   mean_variations median_variations max_variations
##             <dbl>             <dbl>          <dbl>
## 1        6484572.           6172346       12945965
```

```r
# show by chromosome DNA basepairs
ggplot(chromosome, aes(x = basepairs)) +
  geom_histogram(bins = 30, fill = "blue", alpha = 0.5) +
  labs(title = "Chromosome Size Distribution", x = "Size", y = "Count") +
  theme_minimal()
```

b. How does the chromosome size distribute? Plot a graph that helps to visualize this by using ggplot2 package functions.



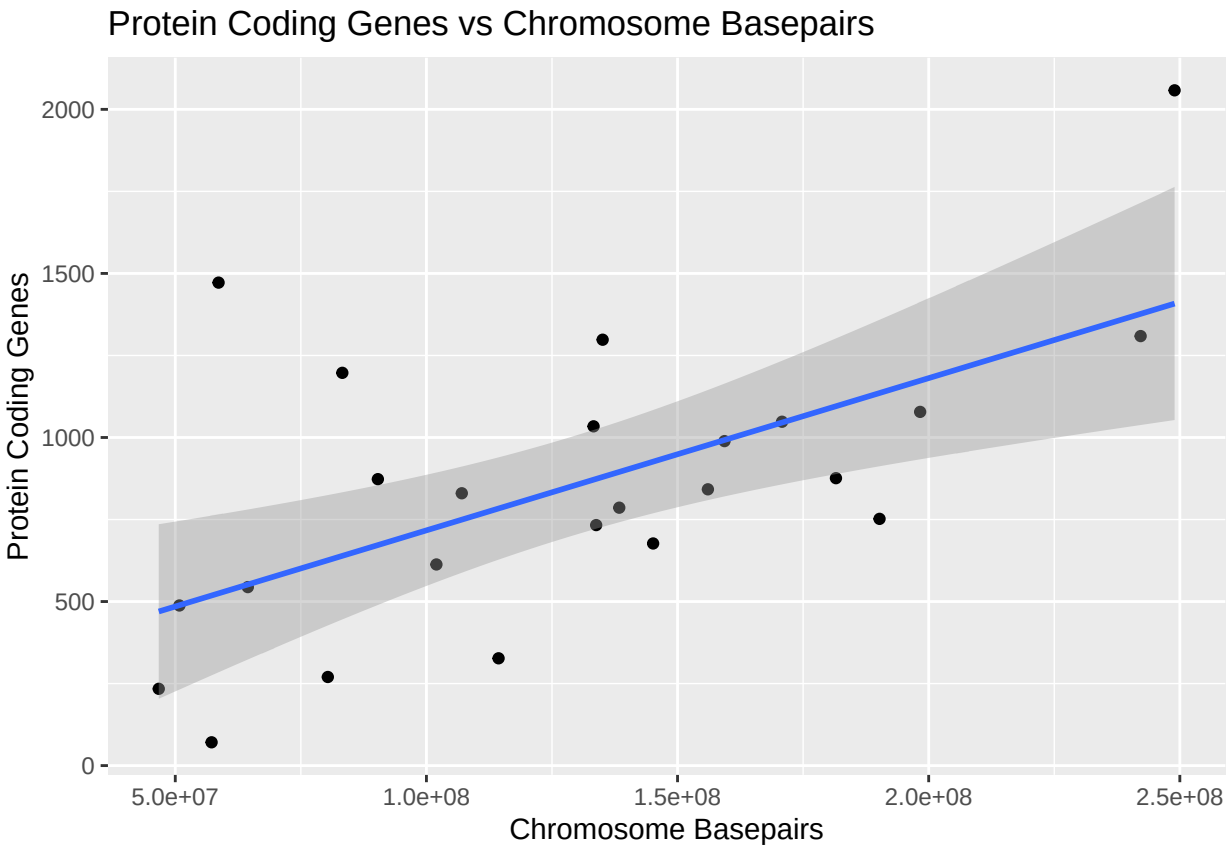

```r
# show by chromosome physical length
ggplot(chromosome, aes(x = length_mm)) +
geom_histogram(bins = 30, fill = "blue", alpha = 0.5) +
labs(title = "Chromosome Size Distribution", x = "Size", y = "Count") +
theme_minimal()
```



c. Does the number of protein coding genes or miRNAs correlate with the length of the chromosome? Make two separate plots to visualize these relationships.

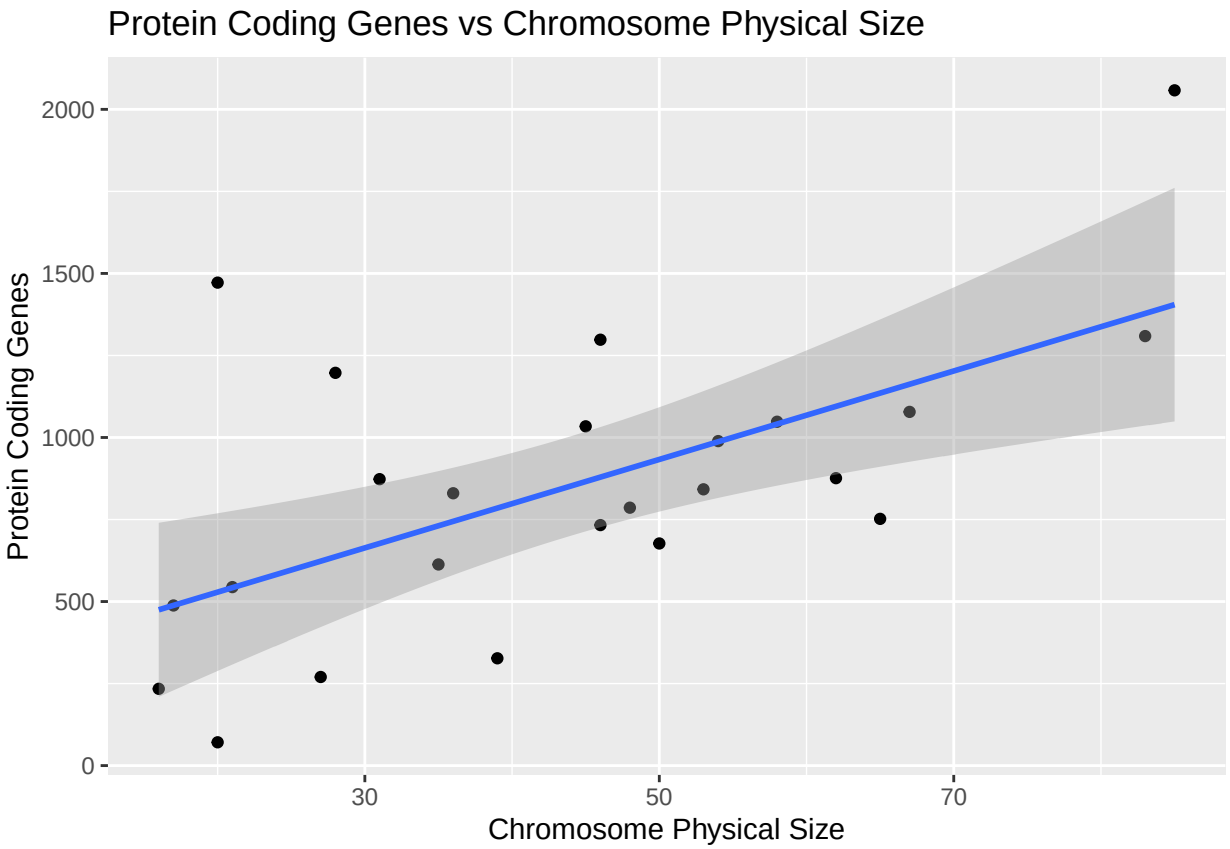
```
# by basepairs
ggplot(chromosome, aes(x = basepairs, y = protein_codinggenes)) +
geom_point() +
geom_smooth(method = "lm") +
labs(title = "Protein Coding Genes vs Chromosome Basepairs", x = "Chromosome Basepairs", y = "Protein C
```

```
## `geom_smooth()` using formula = 'y ~ x'
```




```
# by physical length
ggplot(chromosome, aes(x = length_mm, y = protein_codinggenes)) +
geom_point() +
geom_smooth(method = "lm") +
labs(title = "Protein Coding Genes vs Chromosome Physical Size", x = "Chromosome Physical Size", y = "Pr
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



d. Calculate the same summary statistics for the 'proteins' data variables length and mass. Create a meaningful visualization of the relationship between these two variables by utilizing the ggplot2 package functions. Play with the colors, theme- and other visualization parameters to create a plot that pleases you.

```
# check structure
head(proteins)
```

```
## # A tibble: 6 x 8
##   uniprot_id gene_name gene_name_alt   protein_name    protein_name_alt sequence
##   <chr>      <chr>     <chr>           <chr>           <chr>            <chr>
## 1 P04217     A1BG      <NA>            "Alpha-1B-glyc~ Alpha-1-B glyco~ MSMLVVF~
## 2 Q9NQ94     A1CF      ACF ASP         "APOBEC1 compl~ APOBEC1-stimula~ MESNHKS~
## 3 P01023     A2M       CPAMD5 FWP007   "Alpha-2-macro~ Alpha-2-M) (C3 ~ MGKNKLL~
## 4 A8K2U0     A2ML1     CPAMD9          "Alpha-2-macro~ C3 and PZP-like~ MWAQLLL~
## 5 U3KPV4     A3GALT2   A3GALT2P IGBS3S "Alpha-1,3-gal~ EC 2.4.1.87) (I~ MALKEGL~
## 6 Q9NPC4     A4GALT    A14GALT A4GALT1 "Lactosylceram~ EC 2.4.1.228) (~ MSKPPDL~
## # i 2 more variables: length <dbl>, mass <dbl>
```

```
str(proteins)
```

```
## tibble [20,430 x 8] (S3: tbl_df/tbl/data.frame)
##  $ uniprot_id      : chr [1:20430] "P04217" "Q9NQ94" "P01023" "A8K2U0" ...
##  $ gene_name       : chr [1:20430] "A1BG" "A1CF" "A2M" "A2ML1" ...
##  $ gene_name_alt   : chr [1:20430] NA "ACF ASP" "CPAMD5 FWP007" "CPAMD9" ...
##  $ protein_name    : chr [1:20430] "Alpha-1B-glycoprotein " "APOBEC1 complementation factor " "Alpha
##  $ protein_name_alt: chr [1:20430] "Alpha-1-B glycoprotein)" "APOBEC1-stimulating protein)" "Alpha-2
```

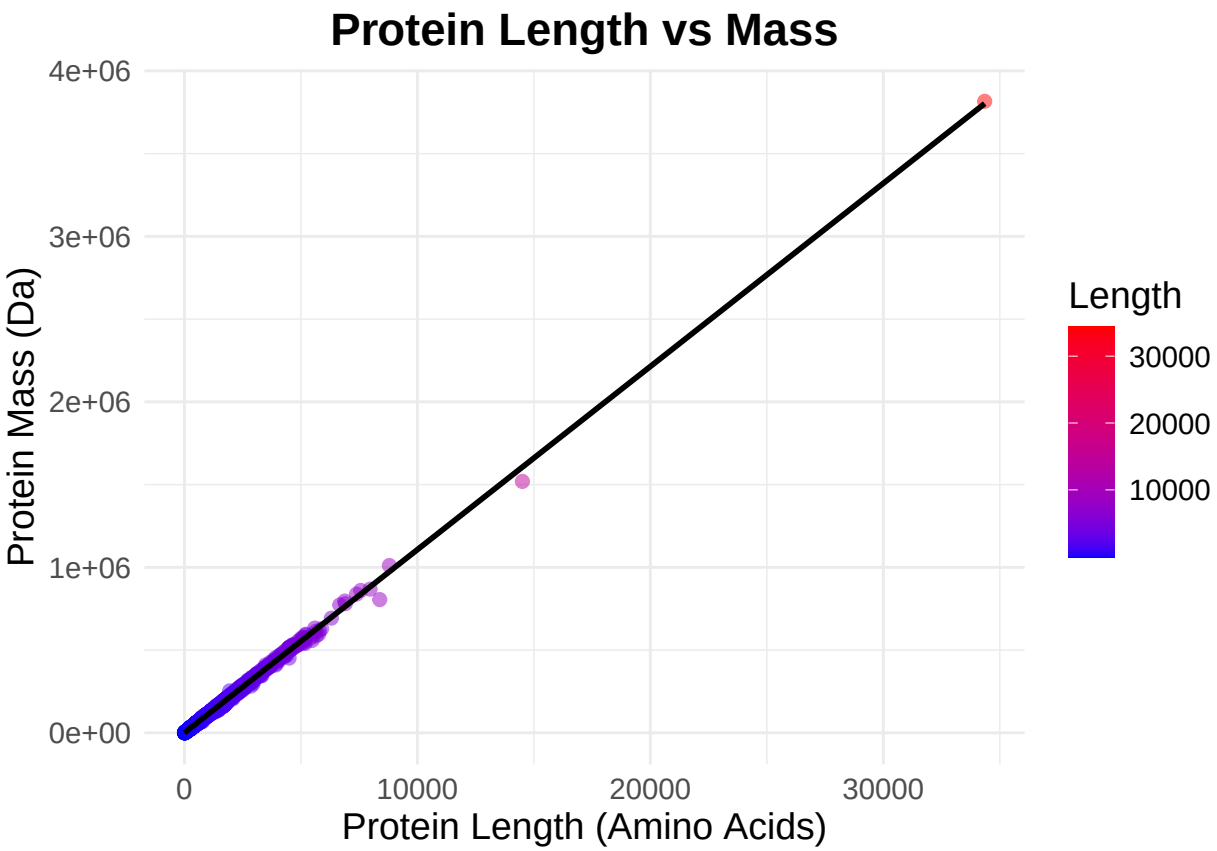
```
##  $ sequence        : chr [1:20430] "MSMLVVFLLLWGVTWGPVTEAAIFYETQPSLWAESESLLKPLANVTLTCQAHLETPDFQLFKNG
##  $ length          : num [1:20430] 495 594 1474 1454 340 ...
##  $ mass            : num [1:20430] 54254 65202 163291 161107 38754 ...
```

```
# summary statistics
proteins %>%
  summarise(
    mean_length = mean(length, na.rm = TRUE),
    median_length = median(length, na.rm = TRUE),
    max_length = max(length, na.rm = TRUE),
    mean_mass = mean(mass, na.rm = TRUE),
    median_mass = median(mass, na.rm = TRUE),
    max_mass = max(mass, na.rm = TRUE)
  )
```

```
## # A tibble: 1 x 6
##   mean_length median_length max_length mean_mass median_mass max_mass
##         <dbl>         <dbl>      <dbl>     <dbl>       <dbl>    <dbl>
## 1        557.           414      34350    62061.      46140.  3816030
```

```
# relationship between protein length and protein mass
ggplot(proteins, aes(x = length, y = mass, color = length)) +
  geom_point(alpha = 0.5, size = 2) +  # scatter plot
  geom_smooth(method = "lm", color = "black", se = TRUE) +  # regression line
  scale_color_gradient(low = "blue", high = "red") +  # color gradation
  labs(title = "Protein Length vs Mass",
       x = "Protein Length (Amino Acids)",
       y = "Protein Mass (Da)",
       color = "Length") +
  theme_minimal() +
  theme(
    text = element_text(size = 14),  # text size
    plot.title = element_text(hjust = 0.5, face = "bold")  # centered title
  )
```

```
## `geom_smooth()` using formula = 'y ~ x'
```


Protein Length vs Mass
4e+06
3e+06
2e+06
1e+06
0e+00
Protein Mass (Da)
0
10000
20000
30000
Protein Length (Amino Acids)
Length
30000
20000
10000