02\_Class\_Activity

Bill Perry

# In class activity 2:



# What did we do last time?

* How to start RStudio and set up a project
* How to create a quarto markdown file
* How to add code at the start to make it work better
* How to load libraries library(tidyverse)
* What the <- is and what it does
* How to read in a file - csv and excel - name\_df <- read\_csv("data/file.csv")
* How to graph data - can you read this out loud in English words
* ggplot(name\_df, aes(x\_variable, y\_variable, color = categorical\_variable)) +  
  # dataframe, aesthetics(x and y variables, mapping of color or fill or shape) +   
   geom\_point() +  
  # this it the geometry you want and can add more layers like  
   geom\_line()
* What questions do you have and what is unclear - what did not work so far when you started the homework?

# Objectives and goals for today

* Data wrangling and setting up a project
  + variable names - directory names - object names
  + can be anything you want - choose wisely young padawan
* How can we save the data we modify and where to save it?
* Why is it important to use the same names for variables?
  + can reuse code
  + clearer names and use
* What is the %>% or the |> - the pipe will make you happy
* Summarize data to a new dataframe
* Improve graphs
  + adding labels
  + doing summary plots
  + getting fancy
* combining or splitting dataframes - the bind\_rows(1,2)
* Do we go wide to long and back again? We will see



# Before we start - Planning the workflow

1. What data do we have
   1. what is the controlled vocabulary?
   2. are there units?
2. What is the directory structure?
3. Do we have a metadata file?
4. Is the data entered in a tidy format?
5. What are we missing?

# Now lets create a new quarto file in the system I set up

* note I usually use this sort of system in an r\_projects directory
* I have redone it for the class to organize all of the terms data
* you should try making some of your own projects



## In RStudio:

1. click file - open project and select the 2025\_UMD\_BioStats\_Student\_Code.Rproj file or double click on it in the finder or data explorer.
2. your screen will now change as RStudio knows where home is



1. Note that in the upper right you will see 2025\_UMD\_BioStats\_Student\_Code so you know you are in the right spot
2. Now click File - New File - Quarto File



1. Create a file that starts with 02\_ and then something that will help you know what is going on like 02\_class\_activity\_in\_class.qmd
2. Now this file thinks this is home.
3. So I usually copy stuff for the header from another file as its just too hard to remember all this…

---  
title: "Title of your file" # Title of the file  
author: "Your Name" # who you are  
format: # this is the formats that it will render to  
 html:  
 toc: false # not table of contents  
 default: true  
 embed-resources: true # makes everything go into the html file  
 self-contained: true # also makes self contained  
editor: visual # type of editing  
project:  
 execute-dir: project # where it will look for files  
execute:  
 keep-md: true # retains the images when you start again  
 cache: true # also heps reatain images and code  
---

# Now to load the libraries

# install packages -----  
# install.packages("readxl")  
# install.packages("tidyverse")  
  
# # we will install a few new libraries  
# install.packages("skimr")

Each script you run from then on you will load the libraries from within the package.

# Load the libraries ----  
library(readxl) # allows to read in excel files  
library(tidyverse) # provides utilities seen in console  
library(skimr) # provide summary stats  
library(janitor) # it cleans ; )

Attaching package: 'janitor'

The following objects are masked from 'package:stats':  
  
 chisq.test, fisher.test

# Loading files

Now like we did before with x and y we will do this with a spreadsheet from a CSV file or excel file

We are going to work with the same data we did in the last class.

p\_df <- read\_csv("data/pine\_needles.csv")

Can you do this in excel?

# The Dataframe and variables

## The whole dataframe - wont show all

p\_df

# A tibble: 48 × 6  
 date group n\_s wind tree\_no len\_mm  
 <chr> <chr> <chr> <chr> <dbl> <dbl>  
 1 3/20/25 cephalopods n lee 1 20  
 2 3/20/25 cephalopods n lee 1 21  
 3 3/20/25 cephalopods n lee 1 23  
 4 3/20/25 cephalopods n lee 1 25  
 5 3/20/25 cephalopods n lee 1 21  
 6 3/20/25 cephalopods n lee 1 16  
 7 3/20/25 cephalopods s wind 1 15  
 8 3/20/25 cephalopods s wind 1 16  
 9 3/20/25 cephalopods s wind 1 14  
10 3/20/25 cephalopods s wind 1 17  
# ℹ 38 more rows

## The top of the datafame to see what it looks like

head(p\_df)

# A tibble: 6 × 6  
 date group n\_s wind tree\_no len\_mm  
 <chr> <chr> <chr> <chr> <dbl> <dbl>  
1 3/20/25 cephalopods n lee 1 20  
2 3/20/25 cephalopods n lee 1 21  
3 3/20/25 cephalopods n lee 1 23  
4 3/20/25 cephalopods n lee 1 25  
5 3/20/25 cephalopods n lee 1 21  
6 3/20/25 cephalopods n lee 1 16

# tail(p\_df) # the bottom

## How to look at one variable

p\_df$group # name of dataframe $ varaible

[1] "cephalopods" "cephalopods" "cephalopods" "cephalopods" "cephalopods"  
 [6] "cephalopods" "cephalopods" "cephalopods" "cephalopods" "cephalopods"  
[11] "cephalopods" "cephalopods" "salmon" "salmon" "salmon"   
[16] "salmon" "salmon" "salmon" "salmon" "salmon"   
[21] "salmon" "salmon" "salmon" "salmon" "crayfish"   
[26] "crayfish" "crayfish" "crayfish" "crayfish" "crayfish"   
[31] "crayfish" "crayfish" "crayfish" "crayfish" "crayfish"   
[36] "crayfish" "snail" "snail" "snail" "snail"   
[41] "snail" "snail" "snail" "snail" "snail"   
[46] "snail" "snail" "snail"

# Plotting

## XY Plot with dodged points

knitr::opts\_chunk$set(  
 comment = '', fig.width = 4, fig.height = 3)  
  
ggplot(data = p\_df, aes(x=wind, y=len\_mm)) +   
 geom\_point(position = position\_dodge2(width=0.2) )



# this dodges the points # position\_dodge2 or can use position\_dodge depending on grouping

## Histogram Color with Bins

knitr::opts\_chunk$set(  
 comment = '', fig.width = 4, fig.height = 3)  
  
ggplot(data = p\_df, aes(x=len\_mm, fill = wind)) +  
 geom\_histogram( binwidth = 2,   
# sets the width in units of the bins - try different nubmers  
 position = position\_dodge2(width = 0.5))



## Box and Whisker Plots

knitr::opts\_chunk$set(  
 comment = '', fig.width = 4, fig.height = 3)  
  
ggplot(data = p\_df, aes(x=wind, y=len\_mm, fill = wind)) + geom\_boxplot()



# How to save plots

xy\_plot <- ggplot(data = p\_df, aes(x=wind, y=len\_mm)) +   
 geom\_point(position = position\_dodge2(width=0.2) )  
xy\_plot



## xy\_plot is not in the environment

we can save this plot by typing

ggsave(xy\_plot, # name of the plot you called it  
 file = "figures/xy\_plot.pdf", # type of file you want - pdf is good  
 units = "in", # what are the units you want to measure in  
 width = 6, height = 6) # dimensions - not if large the fonts need to be adjusted

The file should be in the figures directory

# Now lets make the graph prettier

xy\_plot <- ggplot(data = p\_df, aes(x=wind, y=len\_mm, color = wind, shape = wind)) +   
 geom\_point(  
 size = 2,  
 position = position\_dodge2(width=0.2)) +  
 labs(x="Wind Side", y = "Length (mm)", color = "Wind Side", shape = "Wind Side")  
xy\_plot



## we can also add themes

xy\_plot <- ggplot(data = p\_df, aes(x=wind, y=len\_mm, color = wind, shape = wind)) +   
 geom\_point(  
 size = 2,  
 position = position\_dodge2(width=0.2)) +  
 labs(x="Wind Side", y = "Length (mm)", color = "Wind Side", shape = "Wind Side") +  
 theme\_classic()  
xy\_plot



## change the scale by zooming in

xy\_plot <- ggplot(data = p\_df, aes(x=wind, y=len\_mm, color = wind, shape = wind)) +   
 geom\_point(  
 size = 2,  
 position = position\_dodge2(width=0.2)) +  
 labs(x="Wind Side", y = "Length (mm)", color = "Wind Side", shape = "Wind Side") +  
 theme\_classic() +  
 coord\_cartesian(ylim = c(0,30))  
xy\_plot



# Summarizing data - two ways

lets say we want to summarize the data and need to get n, means, standard deviation, standard error

We could do the following - if we had missing cells the code below would give an error

mean(p\_df$len\_mm) # removes missing values

[1] 17.66667

mean(p\_df$len\_mm, na.rm = TRUE) # removes missing values

[1] 17.66667

length(p\_df$len\_mm)

[1] 48

* the length counts missing and non-missing data
* however this would get old if we had to do this for everything and then to do it for the different groupings - lee and windward…

## we need to learn to pipe things

* the dataframe –> pipe command that feed the dataframe into –> next command

p\_df %>% summarize(mean\_length = mean(len\_mm, na.rm = TRUE))

# A tibble: 1 × 1  
 mean\_length  
 <dbl>  
1 17.7

## What is cool is we can do a lot of different things now

p\_df %>%   
 summarize(  
 mean\_length = mean(len\_mm, na.rm = TRUE),  
 sd\_length = sd(len\_mm, na.rm = TRUE),  
 n\_length = n())

# A tibble: 1 × 3  
 mean\_length sd\_length n\_length  
 <dbl> <dbl> <int>  
1 17.7 3.53 48

## Super cool code in case there are missing values

p\_df %>%   
 summarize(  
 mean\_length = mean(len\_mm, na.rm = TRUE),  
 sd\_length = sd(len\_mm, na.rm = TRUE),  
 n\_length = sum(!is.na(len\_mm)))

# A tibble: 1 × 3  
 mean\_length sd\_length n\_length  
 <dbl> <dbl> <int>  
1 17.7 3.53 48

## what else do we want to know though

* we want to know the mean and such for each group of the treatments
* how could we do this?
  + we need to add a command - `group\_by(wind)
  + but how can we do this for this code?

p\_df %>%   
 summarize(  
 mean\_length = mean(len\_mm, na.rm = TRUE),  
 sd\_length = sd(len\_mm, na.rm = TRUE),  
 n\_length = sum(!is.na(len\_mm)))

# A tibble: 1 × 3  
 mean\_length sd\_length n\_length  
 <dbl> <dbl> <int>  
1 17.7 3.53 48

## what if we wanted to save this output

what else do we need to do to save the output

# The mean and standard error plot

One of the most common ways to present data is to show a mean and standard error plot with the means as a point and the error bars as standard error - we will talk about this next but you have all seen this…

ggplot(p\_df, aes(x = wind, y = len\_mm , color = wind)) +  
 stat\_summary(  
 fun = mean,   
 na.rm = TRUE,   
 geom = "point",   
 size = 3) +  
 stat\_summary(  
 fun.data = mean\_se,   
 na.rm = TRUE,   
 geom = "errorbar",   
 width = 0.2) +  
 # annotate("rect",   
 # xmin = -Inf, xmax = Inf, # Cover the entire x range  
 # ymin = -Inf, ymax = Inf, # Cover the entire y range  
 # fill = "white",   
 # alpha = 1) +  
 labs(  
 x = "Wind Side",  
 y = "Length (mm)",  
 color = "Windy Side"  
 ) +  
 theme\_classic()



## given enough time there are some other tweaks to make

in this code we can do some piping that will be very useful later on

mean\_se\_plot <- p\_df %>%   
 ggplot(aes(wind, len\_mm , color = wind)) +  
 stat\_summary(  
 fun = mean,   
 na.rm = TRUE,   
 geom = "point",   
 size = 3) +  
 stat\_summary(  
 fun.data = mean\_se,   
 na.rm = TRUE,   
 geom = "errorbar",   
 width = 0.2) +  
 # annotate("rect",   
 # xmin = -Inf, xmax = Inf, # Cover the entire x range  
 # ymin = -Inf, ymax = Inf, # Cover the entire y range  
 # fill = "white",   
 # alpha = 1) +  
 labs(  
 x = "Wind Side",  
 y = "Length (mm)",  
 color = "Windy Side"  
 ) +  
 theme\_classic()  
  
mean\_se\_plot



# Using skimr

p\_df %>% skim()

Data summary

|  |  |
| --- | --- |
| Name | Piped data |
| Number of rows | 48 |
| Number of columns | 6 |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  |
| Column type frequency: |  |
| character | 4 |
| numeric | 2 |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  |
| Group variables | None |

**Variable type: character**

| skim\_variable | n\_missing | complete\_rate | min | max | empty | n\_unique | whitespace |
| --- | --- | --- | --- | --- | --- | --- | --- |
| date | 0 | 1 | 7 | 7 | 0 | 1 | 0 |
| group | 0 | 1 | 5 | 11 | 0 | 4 | 0 |
| n\_s | 0 | 1 | 1 | 1 | 0 | 2 | 0 |
| wind | 0 | 1 | 3 | 4 | 0 | 2 | 0 |

**Variable type: numeric**

| skim\_variable | n\_missing | complete\_rate | mean | sd | p0 | p25 | p50 | p75 | p100 | hist |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| tree\_no | 0 | 1 | 2.50 | 1.13 | 1 | 1.75 | 2.5 | 3.25 | 4 | ▇▇▁▇▇ |
| len\_mm | 0 | 1 | 17.67 | 3.53 | 12 | 15.00 | 17.5 | 20.25 | 25 | ▆▇▅▆▃ |

p\_df %>%   
 group\_by(wind) %>%   
 skim()

Data summary

|  |  |
| --- | --- |
| Name | Piped data |
| Number of rows | 48 |
| Number of columns | 6 |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  |
| Column type frequency: |  |
| character | 3 |
| numeric | 2 |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  |
| Group variables | wind |

**Variable type: character**

| skim\_variable | wind | n\_missing | complete\_rate | min | max | empty | n\_unique | whitespace |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| date | lee | 0 | 1 | 7 | 7 | 0 | 1 | 0 |
| date | wind | 0 | 1 | 7 | 7 | 0 | 1 | 0 |
| group | lee | 0 | 1 | 5 | 11 | 0 | 4 | 0 |
| group | wind | 0 | 1 | 5 | 11 | 0 | 4 | 0 |
| n\_s | lee | 0 | 1 | 1 | 1 | 0 | 1 | 0 |
| n\_s | wind | 0 | 1 | 1 | 1 | 0 | 1 | 0 |

**Variable type: numeric**

| skim\_variable | wind | n\_missing | complete\_rate | mean | sd | p0 | p25 | p50 | p75 | p100 | hist |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| tree\_no | lee | 0 | 1 | 2.50 | 1.14 | 1 | 1.75 | 2.5 | 3.25 | 4 | ▇▇▁▇▇ |
| tree\_no | wind | 0 | 1 | 2.50 | 1.14 | 1 | 1.75 | 2.5 | 3.25 | 4 | ▇▇▁▇▇ |
| len\_mm | lee | 0 | 1 | 20.42 | 2.45 | 16 | 18.75 | 20.5 | 21.50 | 25 | ▂▅▇▃▂ |
| len\_mm | wind | 0 | 1 | 14.92 | 1.91 | 12 | 13.75 | 15.0 | 16.00 | 19 | ▅▃▇▂▂ |