Lab1D: Grouping Variables

2024-09-16

Setup

Load libraries and data

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error
```

```
data <- read.csv("seachange2024_DARTs_data_240916.csv", header = TRUE)
str(data)
```

```
## 'data.frame':    1258 obs. of  16 variables:
##  $ year        : int  2024 2024 2024 2024 2024 2024 2024 2024 2024 2024 ...
##  $ month       : int  9 9 9 9 9 9 9 9 9 9 ...
##  $ day         : int  5 5 5 5 5 5 5 5 5 5 ...
##  $ project     : chr  "SeaChange_2024" "SeaChange_2024" "SeaChange_2024" "SeaChange_2024" ...
##  $ cruise_num  : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ stn         : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ depth       : num  0.78 0.787 0.789 0.79 0.792 0.791 0.79 0.791 0.796 0.801 ...
##  $ temp        : num  16.8 16.8 16.8 16.8 16.8 ...
##  $ salinity    : num  31.3 31.3 31.3 31.3 31.3 ...
##  $ density     : num  22.7 22.7 22.7 22.7 22.7 ...
##  $ par         : num  1400 1700 1670 1790 1640 1660 1490 1560 1430 1550 ...
##  $ fluor       : num  0.978 0.984 1.007 1.011 1.052 ...
##  $ turbidity   : num  1.16 1.15 1.15 1.15 1.15 ...
##  $ o2_umolPerKg: num  200 201 201 201 202 ...
##  $ o2_pctSat   : num  81.7 81.8 81.8 81.8 82.3 ...
##  $ ph          : num  8.02 8.03 8.03 8.03 8.03 ...
```

Calculating basic information and statistics

It's relatively straight forward to calculate basic statistics from the data. For example to get the mean and standard deviation from all salinity measurements:

```r
mean(data$salinity)
```

```
## [1] 31.88248
```

```r
sd(data$salinity)
```

```
## [1] 0.4583607
```

Other functions you could play with include: min() max() median()

Subsetting your data

You might want to group the data before calculating these statistics. For this, you can use the groupby function. For example, say we want to group the data by station and then calculation mean salinity. This is how we do this:

```r
sal_summary <- data |> group_by(stn) |> summarize(mean_salinity = mean(salinity), sd_sal = sd(salinity))
sal_summary
```

```
## # A tibble: 4 x 3
##     stn mean_salinity sd_sal
##   <int>         <dbl>  <dbl>
## 1     1          31.4 0.0463
## 2     2          31.6 0.0928
## 3     3          31.7 0.312
## 4     4          32.2 0.370
```

You can also filter a table before running these summary calculations if you only want to target specific measurements. For example, if you want to examine the salinity for everything above 10m, this is how you would do it:

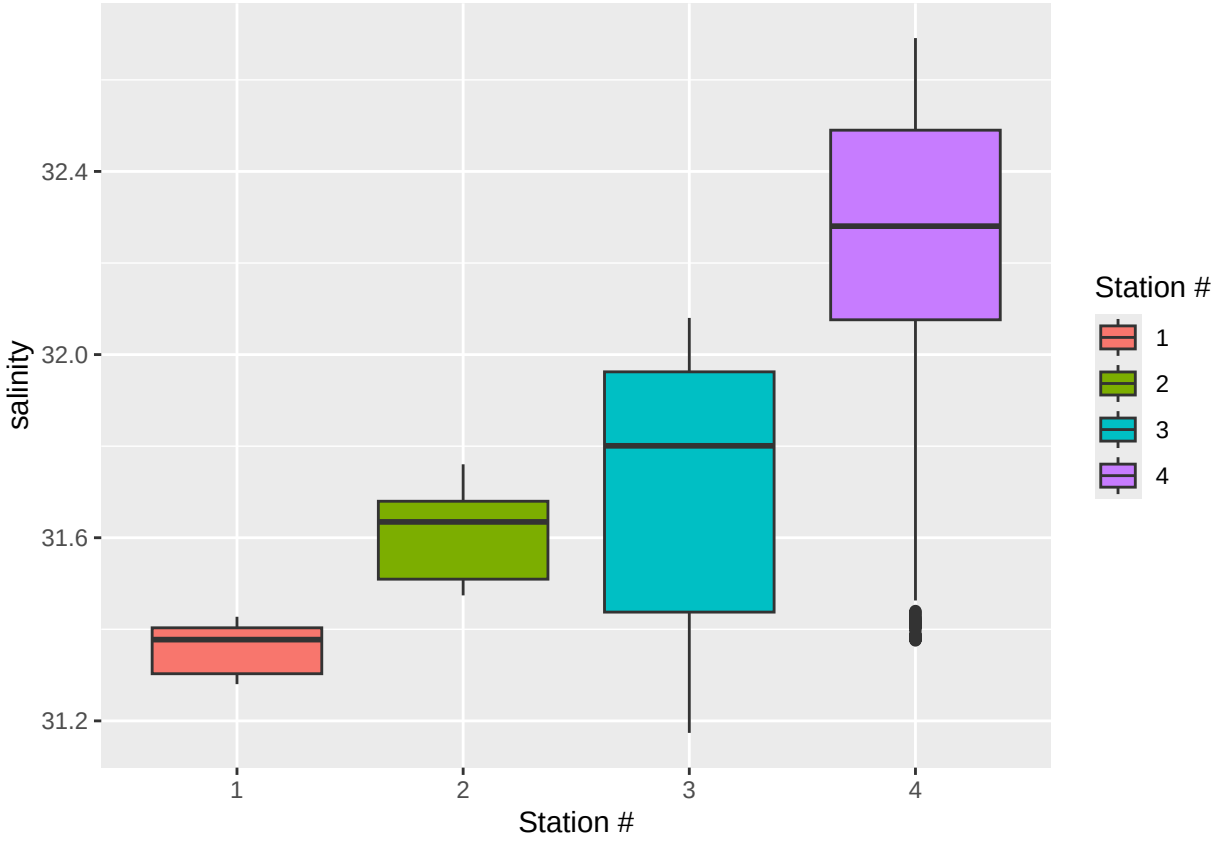
```r
sal_above_10 <- data |> filter(depth <= 10) |> group_by(stn) |> summarize(mean_salinity = mean(salinity))
sal_above_10
```

```
## # A tibble: 4 x 3
##     stn mean_salinity sd_salinity
##   <int>         <dbl>       <dbl>
## 1     1          31.3      0.0214
## 2     2          31.5      0.0523
## 3     3          31.3      0.161
## 4     4          31.5      0.128
```

Visualizing grouped calculations

There are some graphing functions that can help you visualize group statistics. One super helpful one is making boxplots. To do this with ggplot:

```r
ggplot(data, aes(x = as.factor(stn), y = salinity, fill = as.factor(stn))) +
  geom_boxplot() +
  xlab('Station #') +
  labs(fill = 'Station #')
```



David's Assignment Part 2:

2) Calculate the average Temperature (+/- STD) in the upper water column between 1-3m for all 4 stations. Plot that data with station on the X-axis.

Substitute geom_boxplot with other ggplot functions such as: geom_violin geom_jitter

Be sure to label the Axes and put in the correct units.