# INFO 523 Exercise

### Week 3: Getting to know your Data with R

## Table of contents

## Goal

Practice basic R commands/methods for descriptive data analysis. If you are already familiar with some of the commands/methods, practice the ones new to you.

**Note**: copying and pasting early in learning will not produce the results you are looking for, and will catch up to you eventually.

### Submission

Please submit `.r`, `.rmd`, or `.qmd` files ONLY.

### Additional resources

I wrote a Quarto book on Exploratory Data Analysis in R using the `dlookr` package. I won't be showing this package here, but you can use the functions within my book below as well.

## Installing required packages

```r
# First run this
install.packages("pacman")
```

```r
library(pacman)

p_load(dlookr,
       DMwR2, # Data Mining with R functions
       GGally, # Pair-wise plots using ggplot2
       Hmisc, # Data analysis
       palmerpenguins, # Alternative to the Iris dataset
       tidyverse) # Data wrangling, manipulation, visualization
```

## Loading data

```r
data(algae, package = "DMwR2")

algae |> glimpse()
```

```
Rows: 200
Columns: 18
$ season <fct> winter, spring, autumn, spring, autumn, winter, summer, autumn,~
$ size   <fct> small, small, small, small, small, small, small, small, small, ~
$ speed  <fct> medium, medium, medium, medium, medium, high, high, high, mediu~
$ mxPH   <dbl> 8.00, 8.35, 8.10, 8.07, 8.06, 8.25, 8.15, 8.05, 8.70, 7.93, 7.7~
$ mnO2   <dbl> 9.8, 8.0, 11.4, 4.8, 9.0, 13.1, 10.3, 10.6, 3.4, 9.9, 10.2, 11.~
```

```
$ Cl     <dbl> 60.800, 57.750, 40.020, 77.364, 55.350, 65.750, 73.250, 59.067,~
$ NO3    <dbl> 6.238, 1.288, 5.330, 2.302, 10.416, 9.248, 1.535, 4.990, 0.886,~
$ NH4    <dbl> 578.000, 370.000, 346.667, 98.182, 233.700, 430.000, 110.000, 2~
$ oPO4   <dbl> 105.000, 428.750, 125.667, 61.182, 58.222, 18.250, 61.250, 44.6~
$ PO4    <dbl> 170.000, 558.750, 187.057, 138.700, 97.580, 56.667, 111.750, 77~
$ Chla   <dbl> 50.000, 1.300, 15.600, 1.400, 10.500, 28.400, 3.200, 6.900, 5.5~
$ a1     <dbl> 0.0, 1.4, 3.3, 3.1, 9.2, 15.1, 2.4, 18.2, 25.4, 17.0, 16.6, 32.~
$ a2     <dbl> 0.0, 7.6, 53.6, 41.0, 2.9, 14.6, 1.2, 1.6, 5.4, 0.0, 0.0, 0.0, ~
$ a3     <dbl> 0.0, 4.8, 1.9, 18.9, 7.5, 1.4, 3.2, 0.0, 2.5, 0.0, 0.0, 0.0, 2.~
$ a4     <dbl> 0.0, 1.9, 0.0, 0.0, 0.0, 0.0, 3.9, 0.0, 0.0, 2.9, 0.0, 0.0, 0.0~
$ a5     <dbl> 34.2, 6.7, 0.0, 1.4, 7.5, 22.5, 5.8, 5.5, 0.0, 0.0, 1.2, 0.0, 1~
$ a6     <dbl> 8.3, 0.0, 0.0, 0.0, 4.1, 12.6, 6.8, 8.7, 0.0, 0.0, 0.0, 0.0, 0.~
$ a7     <dbl> 0.0, 2.1, 9.7, 1.4, 1.0, 2.9, 0.0, 0.0, 0.0, 1.7, 6.0, 1.5, 2.1~
```

The `|>` is the Base R pipe as opposed to the `magrittr` pipe `%>%`. The `|>` pipe can be utilized for most functions in R, while the `%>%` pipe is more restricted towards the `tidyverse`.

## Central tendency: mean, median, mode

### Mean

```
algae$a1 |>
  mean()
```

```
[1] 16.9235
```

### Median

```
algae$a1 |>
  median()
```

```
[1] 6.95
```

### Mode

Base R doesn't have a function for mode, create a simple one to illustrate how to create a function.
(this method works only for unimodal data.)

```
Mode <- function(x, na.rm=FALSE){
if(na.rm) x<-x[!is.na(x)]
ux <- unique (x)
return (ux[which.max(tabulate(match(x, ux)))])
}

algae$a2 |> Mode()
```

```
[1] 0
```

**DMwR `centralValue()` function:**

returns the median for numerical variable, or the mode for nominal variables.

```
# Numerical variable
algae$a1 |> centralValue()
```

```
[1] 6.95
```

```
# Nominal variable
algae$speed |> centralValue()
```

```
[1] "high"
```

## Statistics of spread (variation)

### Variance

```
algae$a1 |> var()
```

```
[1] 455.7532
```

### Standard deviation

```
algae$a1 |> sd()
```

```
[1] 21.34838
```

### Range

Note that this gives you both maximum and minimum values.

```
algae$a1 |> range()
```

```
[1]  0.0 89.8
```

### Maximum value

```
algae$a1 |> max()
```

```
[1] 89.8
```

### Minimum value

```
algae$a1 |> min()
```

```
[1] 0
```

### Interquartile range

3rd quartile (75%) - 1st quartile (25%)

```
algae$a1 |> IQR()
```

```
[1] 23.3
```

### Quantiles

```
algae$a1 |> quantile()
```

```
   0%   25%   50%   75%  100%
 0.00  1.50  6.95 24.80 89.80
```

Specifying specific quantiles:

```
algae$a1 |> quantile(probs = c(0.2, 0.8))
```

```
  20%   80%
 1.20 32.18
```

## Missing values

```
library(purrr)
# Compute the total number of NA values in the dataset
nas <- algae %>%
  purrr::map_dbl(~sum(is.na(.))) %>%
  sum()

cat("The dataset contains ", nas, "NA values. \n")
```

```
The dataset contains  33 NA values.
```

```
# Compute the number of incomplete rows in the dataset
incomplete_rows <- algae %>%
  summarise_all(~!complete.cases(.)) %>%
  nrow()
```

```
Warning: Returning more (or less) than 1 row per `summarise()` group was deprecated in
dplyr 1.1.0.
i Please use `reframe()` instead.
i When switching from `summarise()` to `reframe()`, remember that `reframe()`
  always returns an ungrouped data frame and adjust accordingly.
i The deprecated feature was likely used in the dplyr package.
  Please report the issue at <https://github.com/tidyverse/dplyr/issues>.
```

```
cat("The dataset contains ", incomplete_rows, "(out of ", nrow(algae),") incomplete rows.
```

```
The dataset contains  200 (out of  200 ) incomplete rows.
```

## Summaries of a dataset

### Base R's `summary()`

```
algae |> summary()
```

```
   season       size       speed         mxPH            mnO2       
 autumn:40   large :45   high  :84   Min.   :5.600   Min.   : 1.500  
 spring:53   medium:84   low   :33   1st Qu.:7.700   1st Qu.: 7.725  
 summer:45   small :71   medium:83   Median :8.060   Median : 9.800  
 winter:62                           Mean   :8.012   Mean   : 9.118  
                                     3rd Qu.:8.400   3rd Qu.:10.800  
                                     Max.   :9.700   Max.   :13.400  
                                     NA's   :1       NA's   :2       
       Cl                NO3                NH4                oPO4        
 Min.   :  0.222   Min.   : 0.050   Min.   :    5.00   Min.   :  1.00  
 1st Qu.: 10.981   1st Qu.: 1.296   1st Qu.:   38.33   1st Qu.: 15.70  
 Median : 32.730   Median : 2.675   Median :  103.17   Median : 40.15  
 Mean   : 43.636   Mean   : 3.282   Mean   :  501.30   Mean   : 73.59  
 3rd Qu.: 57.824   3rd Qu.: 4.446   3rd Qu.:  226.95   3rd Qu.: 99.33  
 Max.   :391.500   Max.   :45.650   Max.   :24064.00   Max.   :564.60  
 NA's   :10        NA's   :2        NA's   :2          NA's   :2       
      PO4               Chla               a1               a2        
 Min.   :  1.00   Min.   :  0.200   Min.   : 0.00   Min.   : 0.000  
 1st Qu.: 41.38   1st Qu.:  2.000   1st Qu.: 1.50   1st Qu.: 0.000  
 Median :103.29   Median :  5.475   Median : 6.95   Median : 3.000  
 Mean   :137.88   Mean   : 13.971   Mean   :16.92   Mean   : 7.458  
 3rd Qu.:213.75   3rd Qu.: 18.308   3rd Qu.:24.80   3rd Qu.:11.375  
 Max.   :771.60   Max.   :110.456   Max.   :89.80   Max.   :72.600  
 NA's   :2        NA's   :12                                        
       a3                a4                a5                a6        
 Min.   : 0.000   Min.   : 0.000   Min.   : 0.000   Min.   : 0.000  
 1st Qu.: 0.000   1st Qu.: 0.000   1st Qu.: 0.000   1st Qu.: 0.000  
 Median : 1.550   Median : 0.000   Median : 1.900   Median : 0.000  
 Mean   : 4.309   Mean   : 1.992   Mean   : 5.064   Mean   : 5.964  
```

```
 3rd Qu.: 4.925   3rd Qu.: 2.400   3rd Qu.: 7.500   3rd Qu.: 6.925
 Max.   :42.800   Max.   :44.600   Max.   :44.400   Max.   :77.600

       a7
 Min.   : 0.000
 1st Qu.: 0.000
 Median : 1.000
 Mean   : 2.495
 3rd Qu.: 2.400
 Max.   :31.600
```

### Hmisc's describe()

```
data("penguins")
penguins |> Hmisc::describe()
```

```
penguins

 8  Variables      344  Observations
--------------------------------------------------------------------------------
species
       n  missing distinct
     344        0        3

Value         Adelie Chinstrap    Gentoo
Frequency        152        68       124
Proportion     0.442     0.198     0.360
--------------------------------------------------------------------------------
island
       n  missing distinct
     344        0        3

Value         Biscoe     Dream Torgersen
Frequency        168       124        52
Proportion     0.488     0.360     0.151
--------------------------------------------------------------------------------
bill_length_mm
       n  missing distinct     Info     Mean      Gmd      .05      .10
     342        2      164        1    43.92    6.274    35.70    36.60
     .25      .50      .75      .90      .95
```

```
   39.23    44.45    48.50    50.80    51.99

lowest : 32.1 33.1 33.5 34   34.1, highest: 55.1 55.8 55.9 58   59.6
---------------------------------------------------------------------------
bill_depth_mm
       n  missing distinct     Info     Mean      Gmd      .05      .10
     342        2       80        1    17.15    2.267     13.9     14.3
     .25      .50      .75      .90      .95
    15.6     17.3     18.7     19.5     20.0

lowest : 13.1 13.2 13.3 13.4 13.5, highest: 20.7 20.8 21.1 21.2 21.5
---------------------------------------------------------------------------
flipper_length_mm
       n  missing distinct     Info     Mean      Gmd      .05      .10
     342        2       55    0.999    200.9    16.03    181.0    185.0
     .25      .50      .75      .90      .95
   190.0    197.0    213.0    220.9    225.0

lowest : 172 174 176 178 179, highest: 226 228 229 230 231
---------------------------------------------------------------------------
body_mass_g
       n  missing distinct     Info     Mean      Gmd      .05      .10
     342        2       94        1     4202    911.8     3150     3300
     .25      .50      .75      .90      .95
    3550     4050     4750     5400     5650

lowest : 2700 2850 2900 2925 2975, highest: 5850 5950 6000 6050 6300
---------------------------------------------------------------------------
sex
       n  missing distinct
     333       11        2

Value      female   male
Frequency     165    168
Proportion  0.495  0.505
---------------------------------------------------------------------------
year
       n  missing distinct     Info     Mean      Gmd
     344        0        3    0.888     2008   0.8919

Value       2007  2008  2009
Frequency    110   114   120
Proportion 0.320 0.331 0.349
```

```
For the frequency table, variable is rounded to the nearest 0.02
--------------------------------------------------------------------------------
```

GMD is the mean absolute difference between any pairs of observations. A robust dispersion measure, especially for non-normally distributed data.

### dlookr's describe()

```
penguins |> dlookr::describe()
```

```
# A tibble: 5 x 26
  described_variables     n    na   mean      sd se_mean     IQR skewness
  <chr>               <int> <int>  <dbl>   <dbl>   <dbl>   <dbl>    <dbl>
1 bill_length_mm        342     2   43.9   5.46   0.295     9.27   0.0531
2 bill_depth_mm         342     2   17.2   1.97   0.107     3.1   -0.143
3 flipper_length_mm     342     2  201.   14.1    0.760    23      0.346
4 body_mass_g           342     2 4202.  802.    43.4    1200      0.470
5 year                  344     0 2008.    0.818  0.0441    2      -0.0537
# i 18 more variables: kurtosis <dbl>, p00 <dbl>, p01 <dbl>, p05 <dbl>,
#   p10 <dbl>, p20 <dbl>, p25 <dbl>, p30 <dbl>, p40 <dbl>, p50 <dbl>,
#   p60 <dbl>, p70 <dbl>, p75 <dbl>, p80 <dbl>, p90 <dbl>, p95 <dbl>,
#   p99 <dbl>, p100 <dbl>
```

## Summaries on a subset of data

dplyr's summarise() and summarise_all(), or use them with select() and group_by() to create summaries on
subset of data. Note: summarise() = summarize()

```
algae |>
  summarise(avgNO3 = mean(NO3, na.rm=TRUE),
            medA1 = median(a1))
```

```
# A tibble: 1 x 2
  avgNO3 medA1
   <dbl> <dbl>
1   3.28  6.95
```

`summarise_all()` can be used to apply any function that produces a scalar value to any column of a data
frame table.

```
algae |>
  select(mxPH:Cl) |>
  summarise_all(list(mean, median), na.rm = TRUE)
```

```
# A tibble: 1 x 6
  mxPH_fn1 mnO2_fn1 Cl_fn1 mxPH_fn2 mnO2_fn2 Cl_fn2
     <dbl>    <dbl>  <dbl>    <dbl>    <dbl>  <dbl>
1     8.01     9.12   43.6     8.06      9.8   32.7
```

```
algae |>
  select(a1:a7) |>
  summarise_all(funs(var))
```

```
Warning: `funs()` was deprecated in dplyr 0.8.0.
i Please use a list of either functions or lambdas:

# Simple named list: list(mean = mean, median = median)

# Auto named with `tibble::lst()`: tibble::lst(mean, median)

# Using lambdas list(~ mean(., trim = .2), ~ median(., na.rm = TRUE))

# A tibble: 1 x 7
     a1    a2    a3    a4    a5    a6    a7
  <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
1  456.  122.  48.3  19.5  56.1  136.  26.6
```

```
algae |>
  select(a1:a7) |>
  summarise_all(c("min", "max"))
```

```
# A tibble: 1 x 14
  a1_min a2_min a3_min a4_min a5_min a6_min a7_min a1_max a2_max a3_max a4_max
   <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>
1      0      0      0      0      0      0      0   89.8   72.6   42.8   44.6
# i 3 more variables: a5_max <dbl>, a6_max <dbl>, a7_max <dbl>
```

### Use `summarise()` with `group_by()`

```
algae |>
  group_by(season, size) |>
  summarise(nObs = n(), mA7 = median(a7))
```

```
`summarise()` has grouped output by 'season'. You can override using the
`.groups` argument.
```

```
# A tibble: 12 x 4
# Groups:   season [4]
   season size    nObs   mA7
   <fct>  <fct>  <int> <dbl>
 1 autumn large     11  0
 2 autumn medium    16  1.05
 3 autumn small     13  0
 4 spring large     12  1.95
 5 spring medium    21  1
 6 spring small     20  0
 7 summer large     10  0
 8 summer medium    21  1
 9 summer small     14  1.45
10 winter large     12  0
11 winter medium    26  1.4
12 winter small     24  0
```

```
penguins |>
  group_by(species) |>
  summarise(var = var(bill_length_mm, na.rm = TRUE))
```

```
# A tibble: 3 x 2
  species     var
  <fct>     <dbl>
1 Adelie     7.09
2 Chinstrap 11.2
3 Gentoo     9.50
```

### Aggregating data

Can be helpful for summary functions that don't return a scalar

```r
penguins |>
  group_by(species) |>
  reframe(var = quantile(bill_length_mm, na.rm = TRUE))
```

```
# A tibble: 15 x 2
   species     var
   <fct>     <dbl>
 1 Adelie     32.1
 2 Adelie     36.8
 3 Adelie     38.8
 4 Adelie     40.8
 5 Adelie     46
 6 Chinstrap  40.9
 7 Chinstrap  46.3
 8 Chinstrap  49.6
 9 Chinstrap  51.1
10 Chinstrap  58
11 Gentoo     40.9
12 Gentoo     45.3
13 Gentoo     47.3
14 Gentoo     49.6
15 Gentoo     59.6
```

`reframe()` expects a scalar result returned by the function, but quantile returns a vector.

**Note**: Aggregating data with summarize was deprecated in `dplyr 1.1.0`, `reframe()` should be used instead.

I recommend just using `dlookr`…

```r
penguins |>
  group_by(species) |>
  dlookr::describe(bill_length_mm)
```

```
# A tibble: 3 x 27
  described_variables species       n    na  mean    sd se_mean   IQR skewness
  <chr>               <fct>     <int> <int> <dbl> <dbl>   <dbl> <dbl>    <dbl>
1 bill_length_mm      Adelie      151     1  38.8  2.66   0.217  4      0.162
2 bill_length_mm      Chinstrap    68     0  48.8  3.34   0.405  4.73  -0.0906
3 bill_length_mm      Gentoo      123     1  47.5  3.08   0.278  4.25   0.651
# i 18 more variables: kurtosis <dbl>, p00 <dbl>, p01 <dbl>, p05 <dbl>,
```

```
#   p10 <dbl>, p20 <dbl>, p25 <dbl>, p30 <dbl>, p40 <dbl>, p50 <dbl>,
#   p60 <dbl>, p70 <dbl>, p75 <dbl>, p80 <dbl>, p90 <dbl>, p95 <dbl>,
#   p99 <dbl>, p100 <dbl>
```

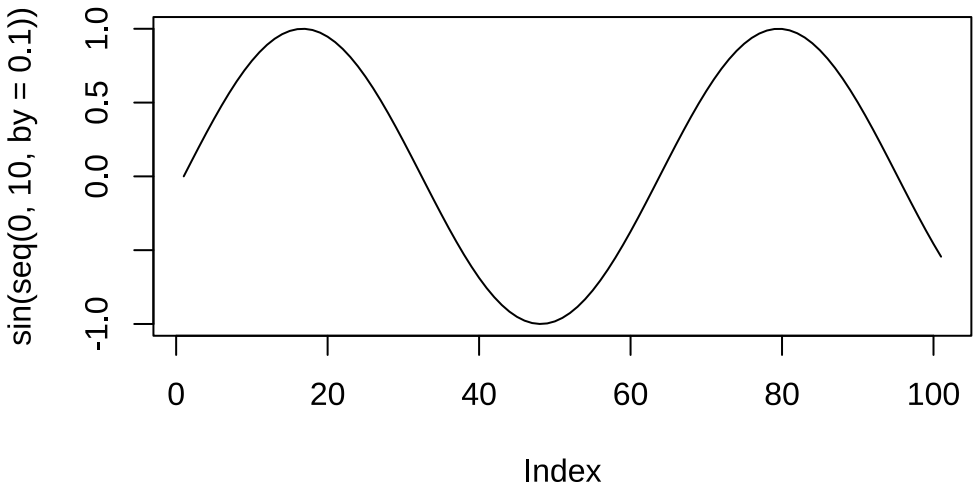
**Grouping functions**

## Data visualization: plotting in R

base R provides graphics and grid packages for statistical plotting powerful plotting packages, e.g., `ggplot2`, are build upon 'graphics' or 'grid' packages
10.

To be very frank, I am strongly biased towards `ggplot2` and the `tidyverse` in general. `ggplot2` has a much more extensive arsenal of extensions, plot types, and customization. In future exercises, I will be solely using `ggplot2`.

```
plot(sin(seq(0, 10, by=0.1)), type="l") # base R plot(), type "l" is a line drawing
```



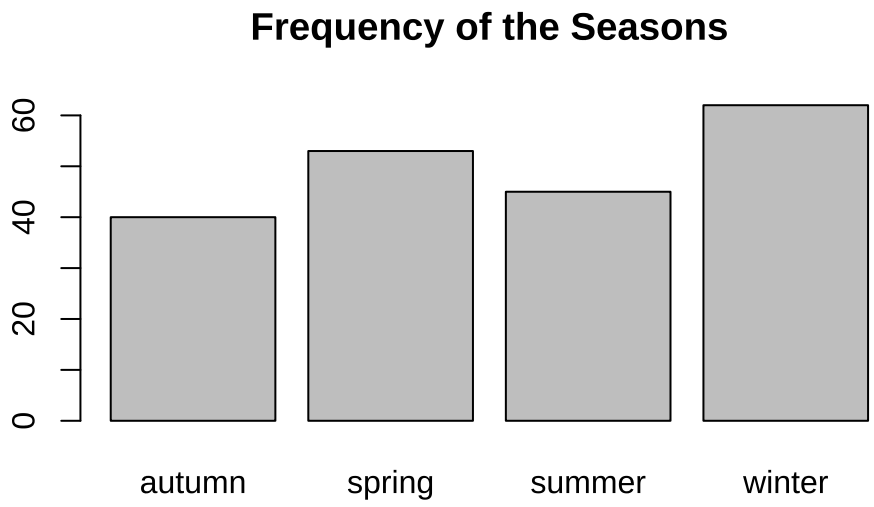

`ggplot2` allows the user to customize their plots in very flexible ways.

To ggplot2, a plot is a mapping from data properties into aesthetic attributes (color, shape, size, etc) of geometric objects (points, lines, bars, etc.)

### Use ggplot2 to visualize nominal variables

### Base R barplot

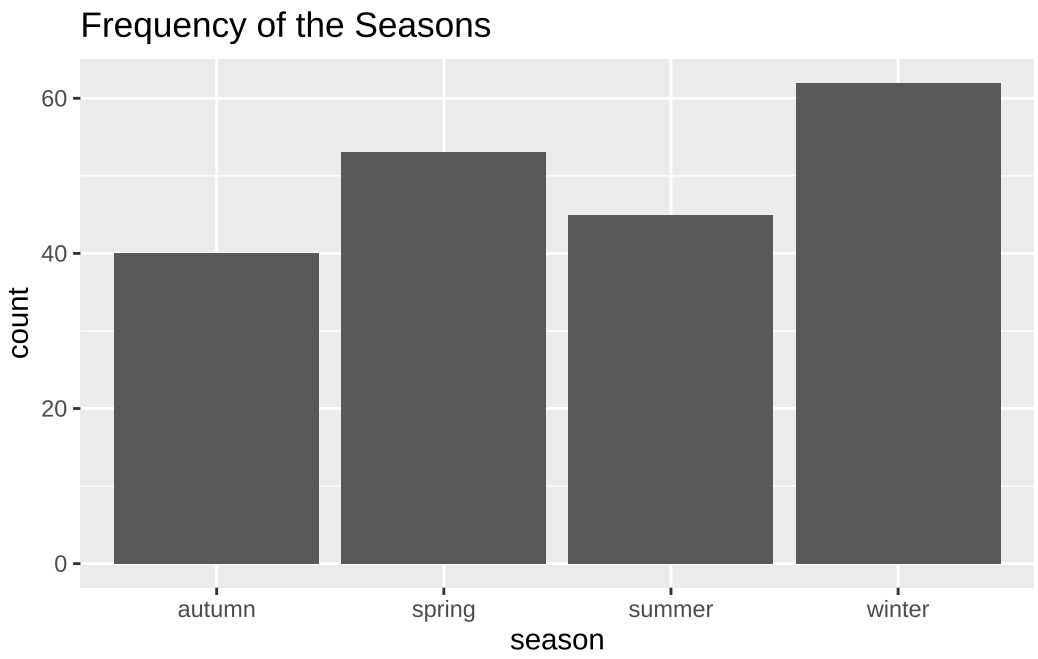
```
freqOcc <- table(algae$season)
barplot(freqOcc, main="Frequency of the Seasons")
```




### `ggplot2` barplot

**ggplot2**: notice how commands are layered using `+`

```
algae |>
  ggplot(aes(x = season)) +
  geom_bar() +
  ggtitle("Frequency of the Seasons")
```

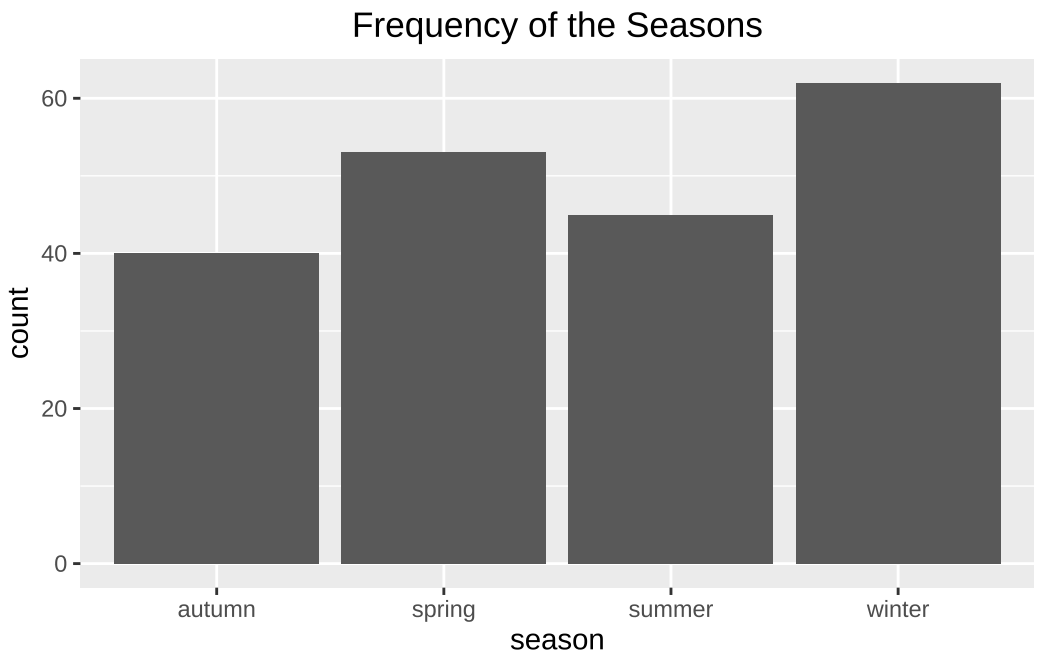



Note that I always pipe the dataset with `ggplot()` because you can add other `tidyverse` functions before the plot argument (e.g., `select()`, `filter()`).

To center the title, run this command first, then all the plots drawn after will have titles

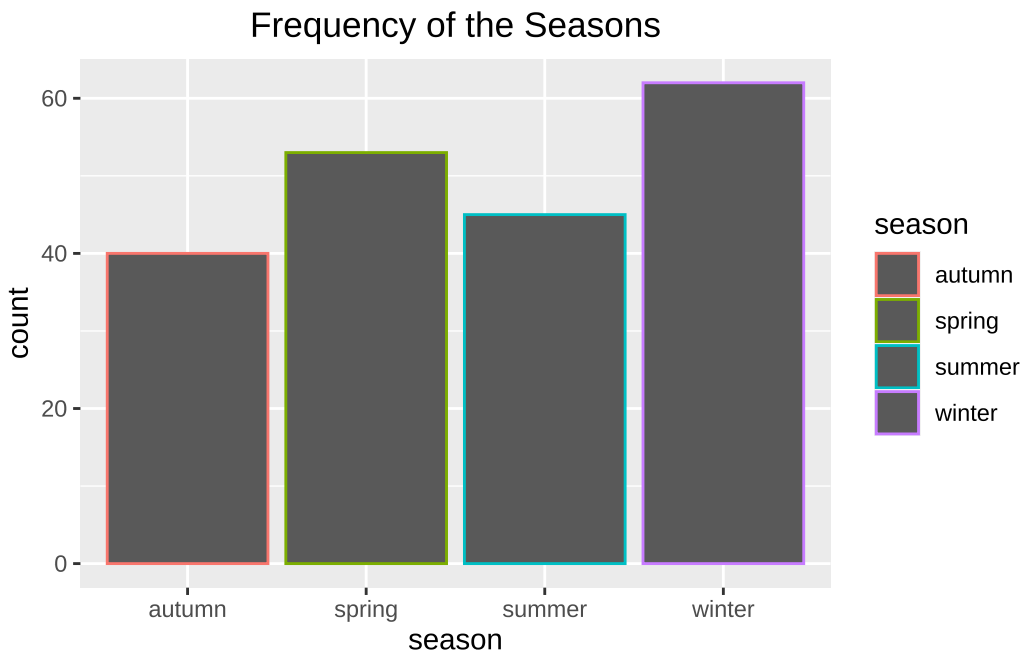
```
theme_update(plot.title = element_text(hjust = 0.5))
```

```
algae |>
  ggplot(aes(x = season)) +
  geom_bar() +
  ggtitle("Frequency of the Seasons")
```



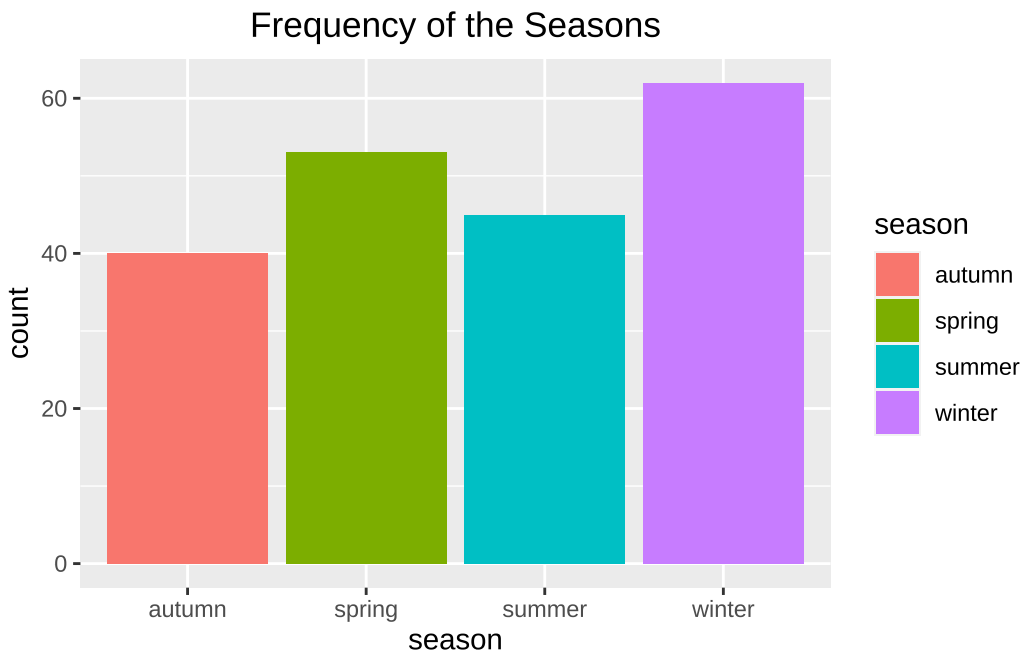

Color different bars: bars of different seasons get different color outline

```
algae |>
  ggplot(aes(x = season, color = season)) +
  geom_bar() +
  ggtitle("Frequency of the Seasons")
```




Use fill to make the entire bars colored

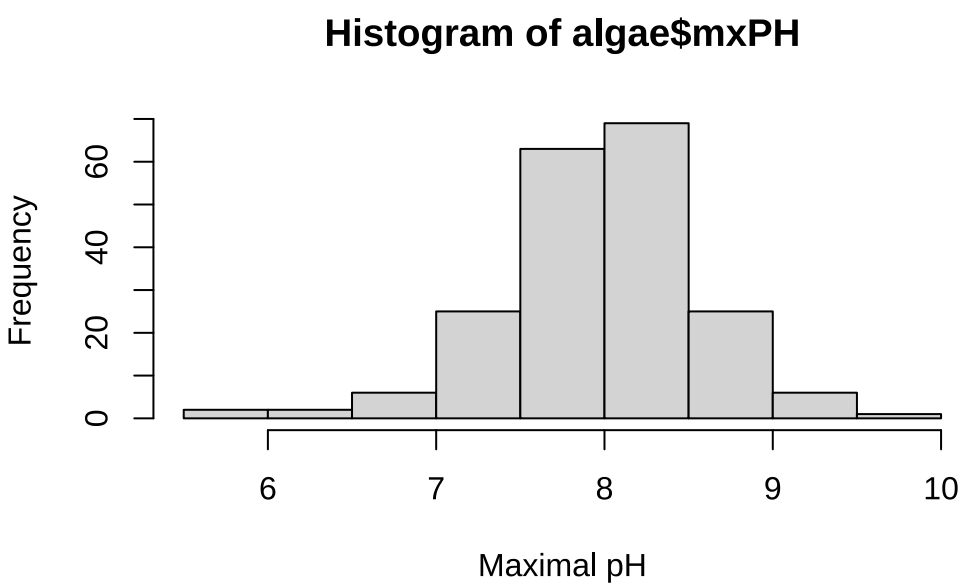
```
algae |>
  ggplot(aes(x = season, fill = season)) +
  geom_bar() +
  ggtitle("Frequency of the Seasons")
```




### Continuous variables: histograms

#### Base R

```
hist(algae$mxPH, xlab = "Maximal pH")
```

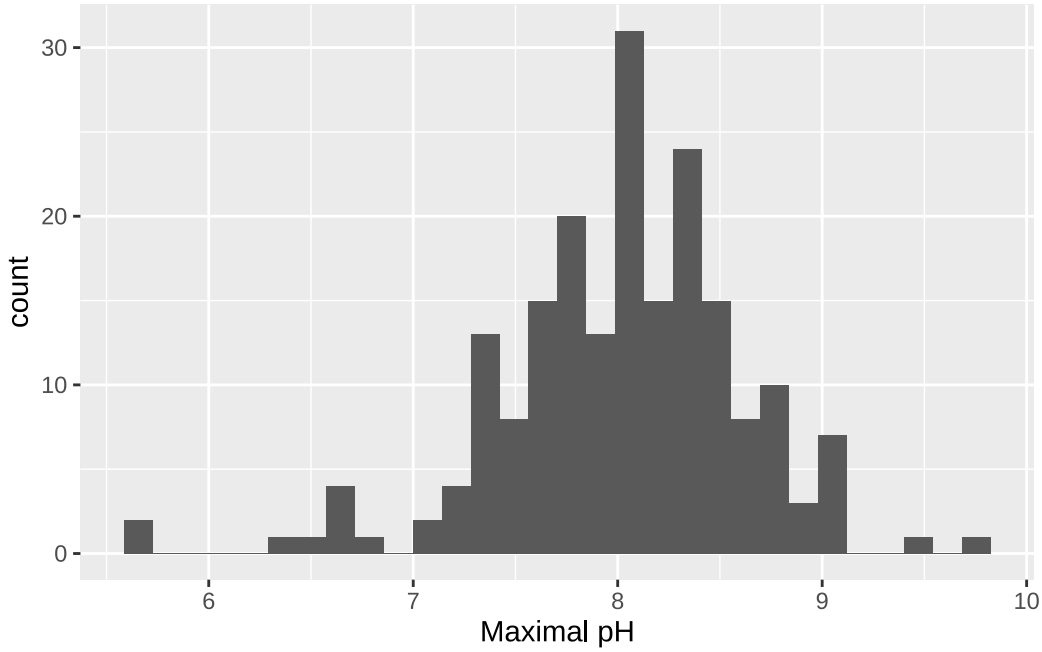



ggplot2

```
algae |>
ggplot(aes(x=mxPH)) +
  geom_histogram() + xlab("Maximal pH")
```

```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

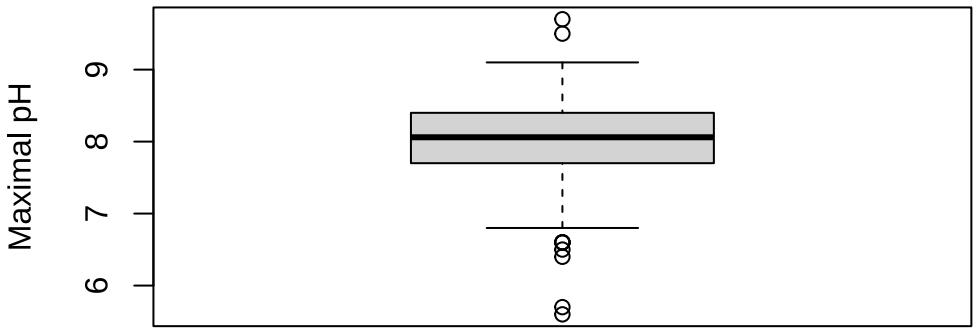
```
Warning: Removed 1 rows containing non-finite values (`stat_bin()`).
```




### Continuous variables: boxplots

#### Base R

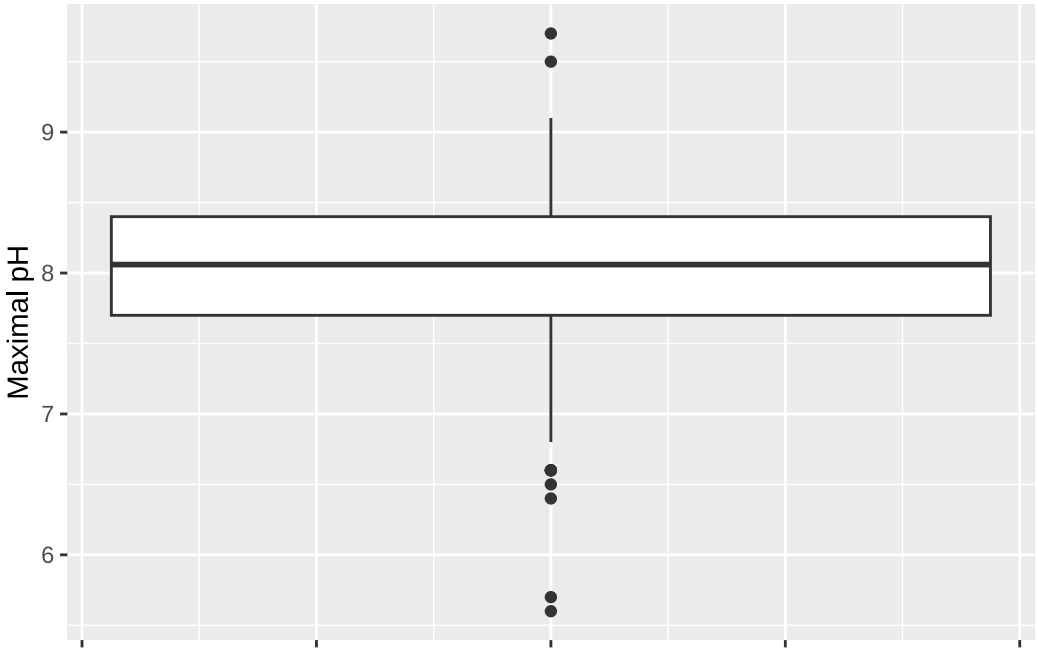
```
boxplot(algae$mxPH, ylab="Maximal pH")
```




ggplot2

```
algae |>
ggplot(aes(y=mxPH)) +
  geom_boxplot() +
  ylab("Maximal pH") +
  theme(axis.text.x = element_blank())
```

```
Warning: Removed 1 rows containing non-finite values (`stat_boxplot()`).
```



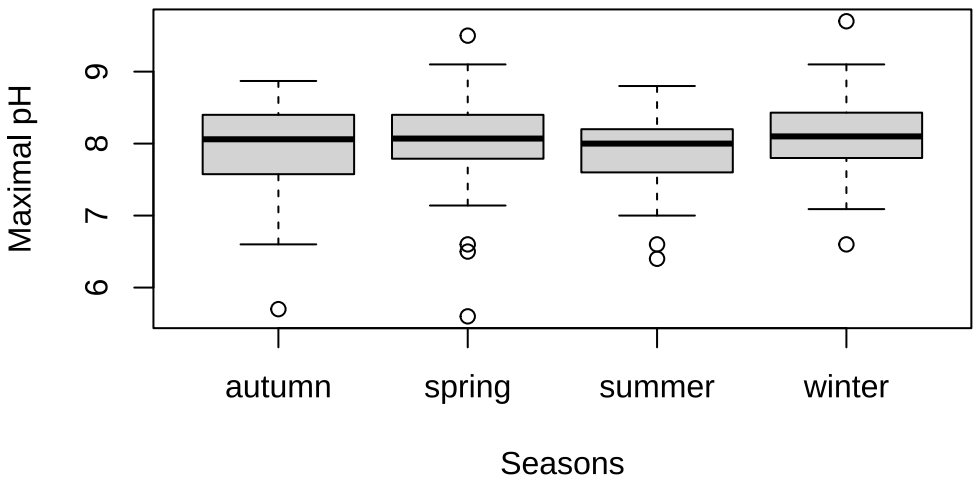

**A guide to customize tick marks and labels**

### Plotting sub-groups of a dataset

This is not easy with base R (except for boxplot), but facets in `ggplot2` make this more streamline.

#### Base R

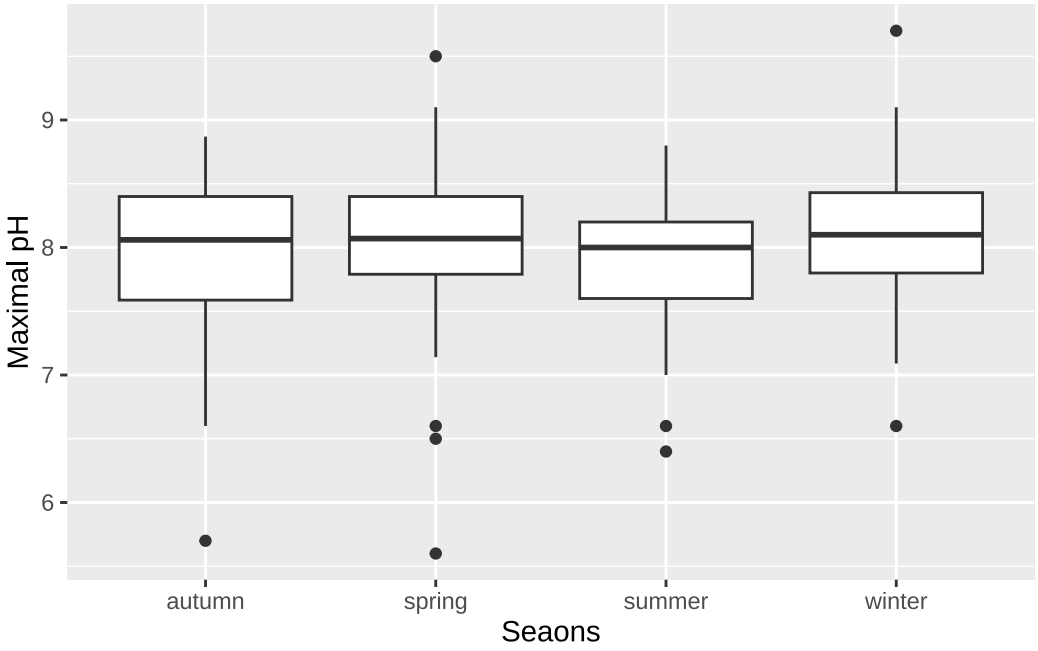
```
boxplot(mxPH ~ season, algae, ylab="Maximal pH", xlab="Seasons")
```




ggplot2

```
algae |>
  ggplot(aes(x = season, y = mxPH)) +
  geom_boxplot() +
  ylab("Maximal pH") +
  xlab("Seaons")
```

```
Warning: Removed 1 rows containing non-finite values (`stat_boxplot()`).
```

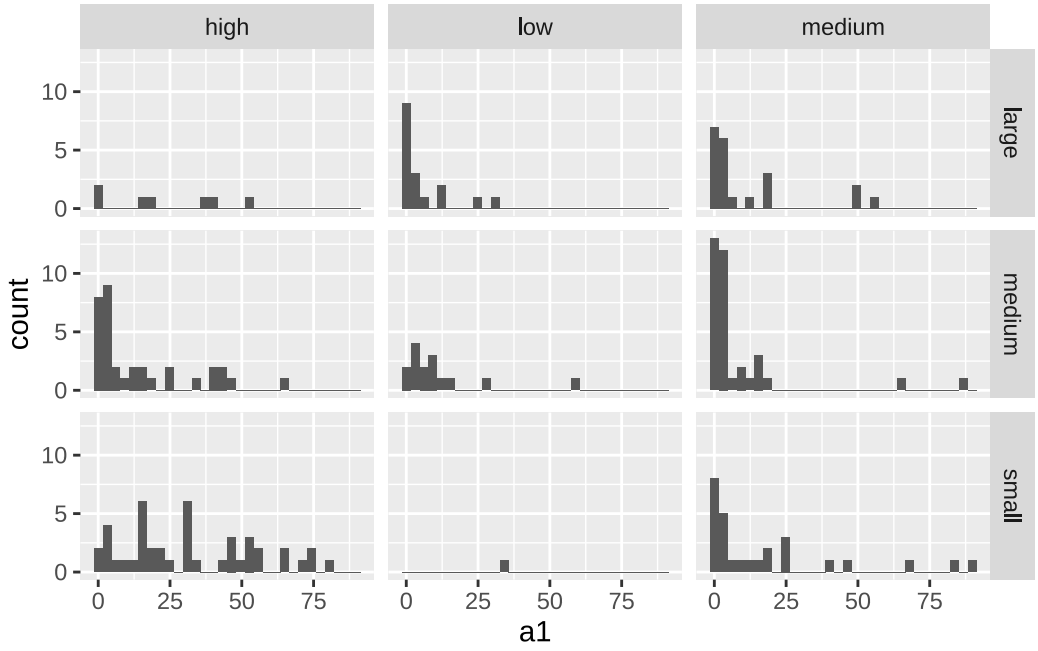



## Facets

Facets are variations of the same plot that are obtained with different subsets of a dataset. Subsetting are obtained often by using categorical variables.

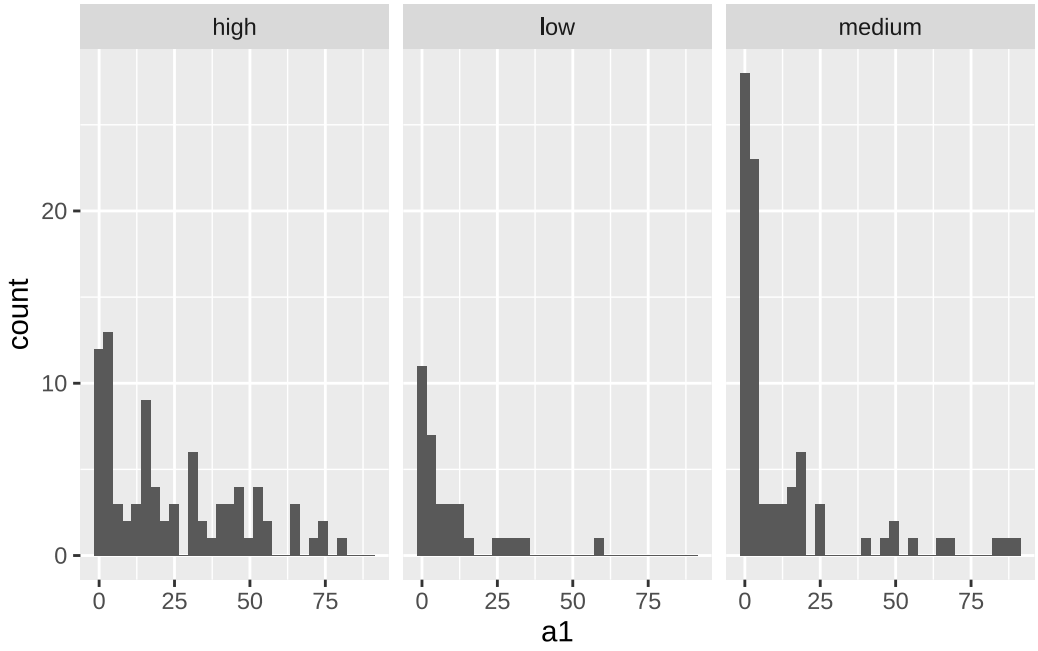
```
algae |>
ggplot(aes(x = a1)) +
  geom_histogram() +
  facet_grid(size ~ speed)
```

```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

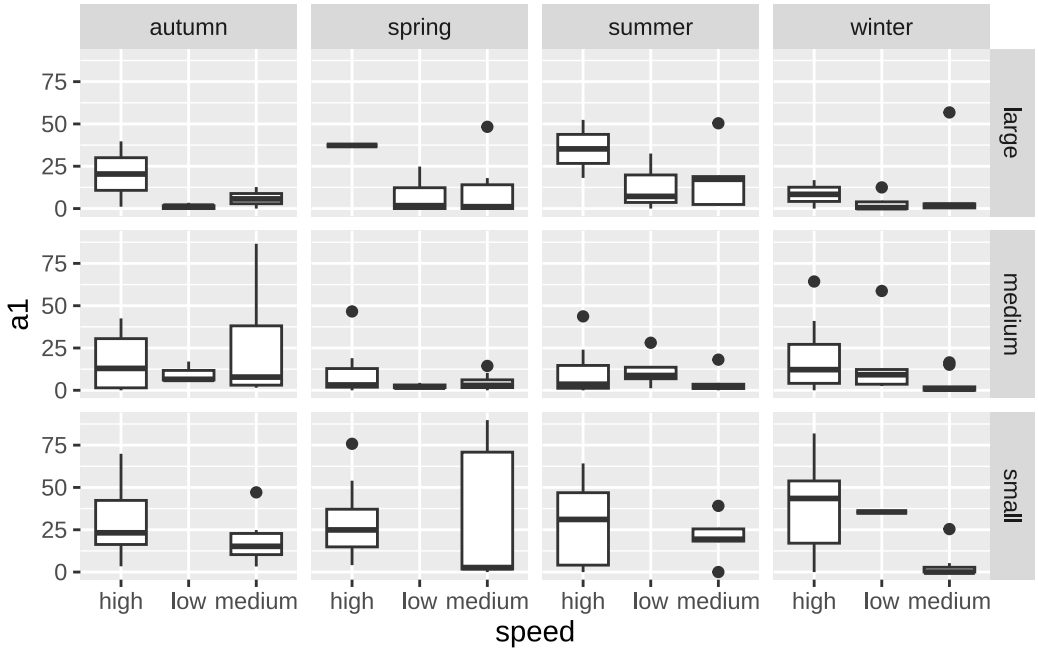



```
algae |>
ggplot(aes(x = a1)) +
  geom_histogram() +
  facet_grid(. ~ speed)
```

```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
algae |>
ggplot(aes(x = speed, y = a1)) +
  geom_boxplot() +
  facet_grid(size ~ season)
```

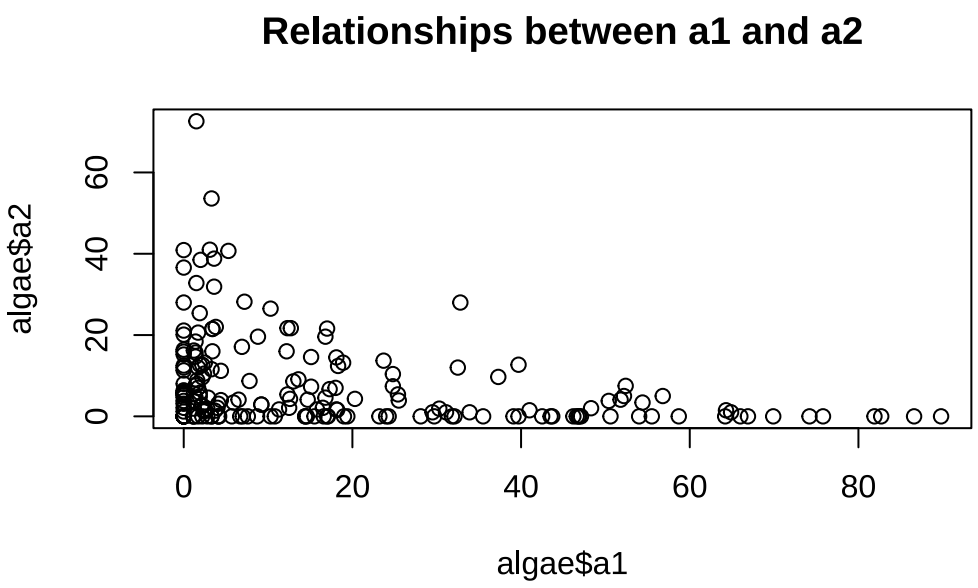



### Continuous variables: scatter plots

Useful to visualize the correlations between two numerical variables.

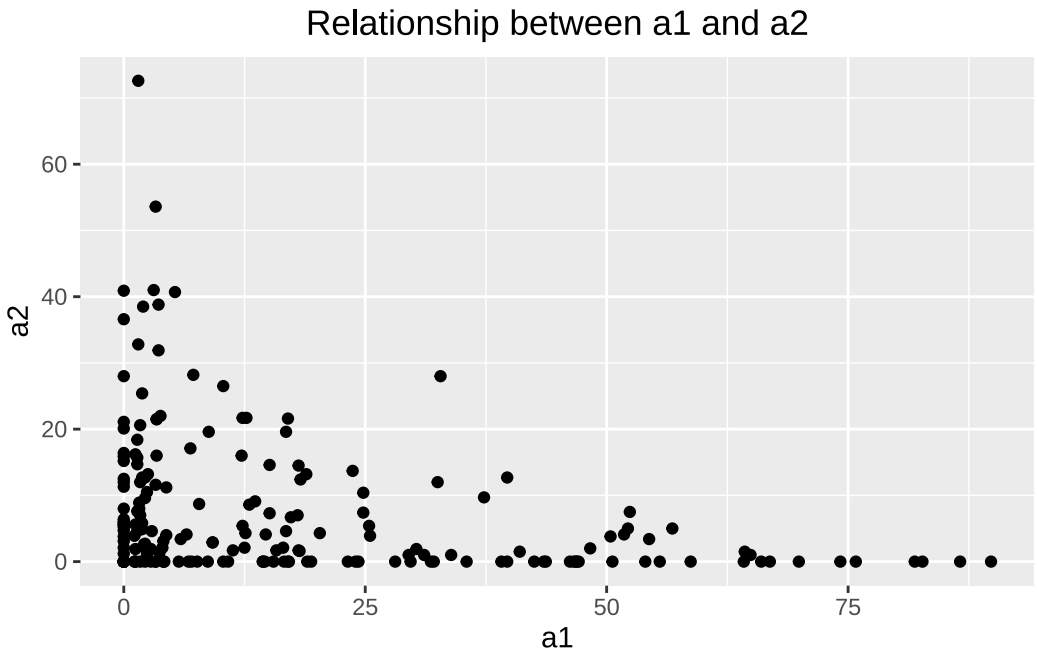
#### Base R

```
plot(algae$a1, algae$a2, main = "Relationships between a1 and a2")
```




ggplot2

```
algae |>
ggplot(aes(x = a1, y = a2)) +
  geom_point() +
  ggtitle("Relationship between a1 and a2")
```

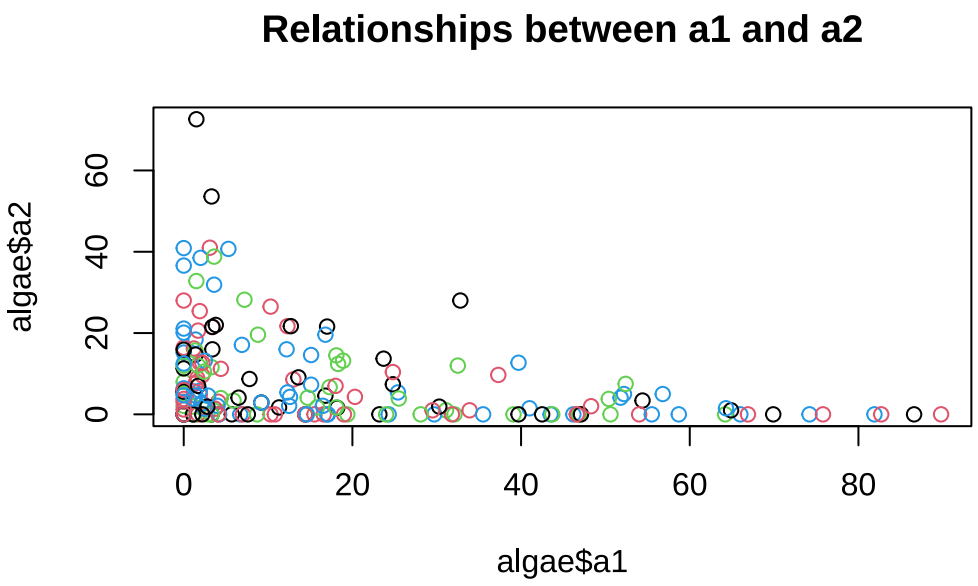



## Curve fitting

### Groups in a scatter plot

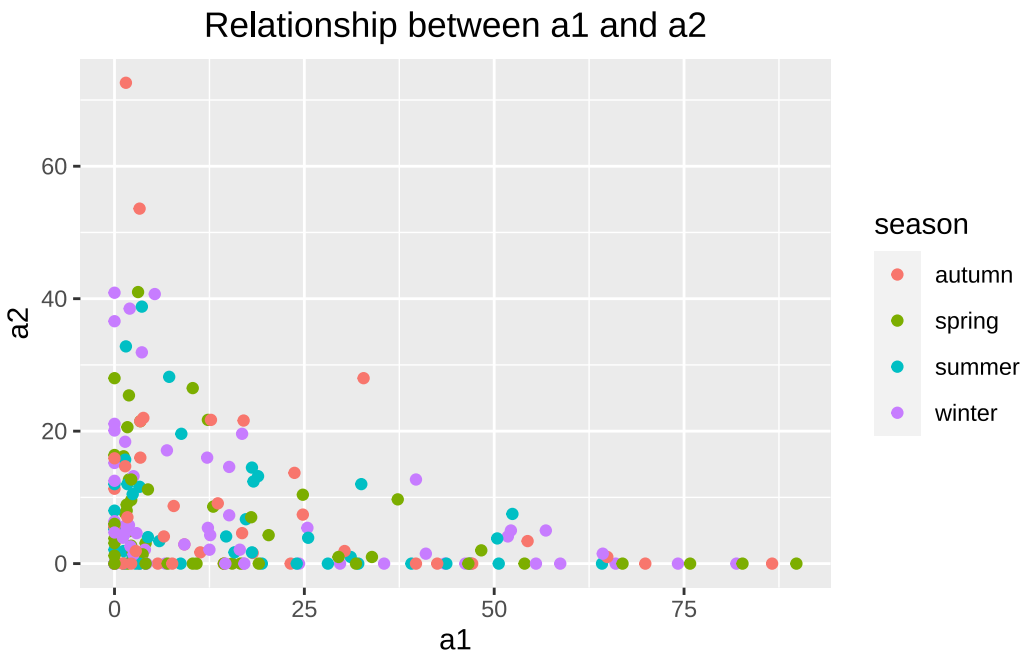
#### Base R

```
plot(algae$a1, algae$a2, col = algae$season, main = "Relationships between a1 and a2")
```



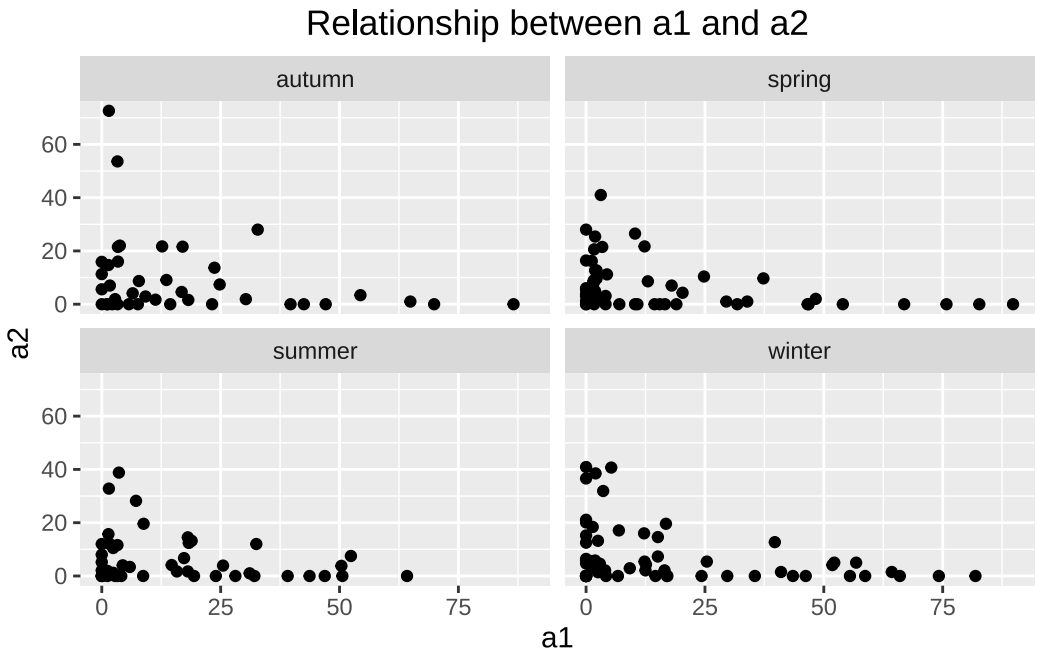

ggplot2

```
algae |>
ggplot(aes(x = a1, y = a2, color = season)) +
  geom_point() +
  ggtitle("Relationship between a1 and a2")
```




**Scatter plot facets**

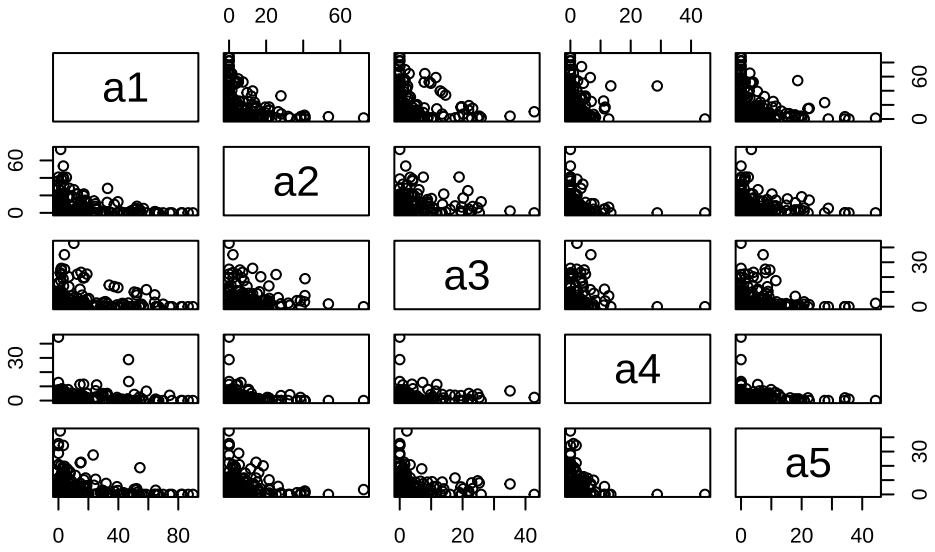
```
algae |>
ggplot(aes(x = a1, y = a2)) +
  geom_point() +
  ggtitle("Relationship between a1 and a2") +
  facet_wrap(~season)
```




### Scatter plot matrix to compare pairs of variables

#### Base R

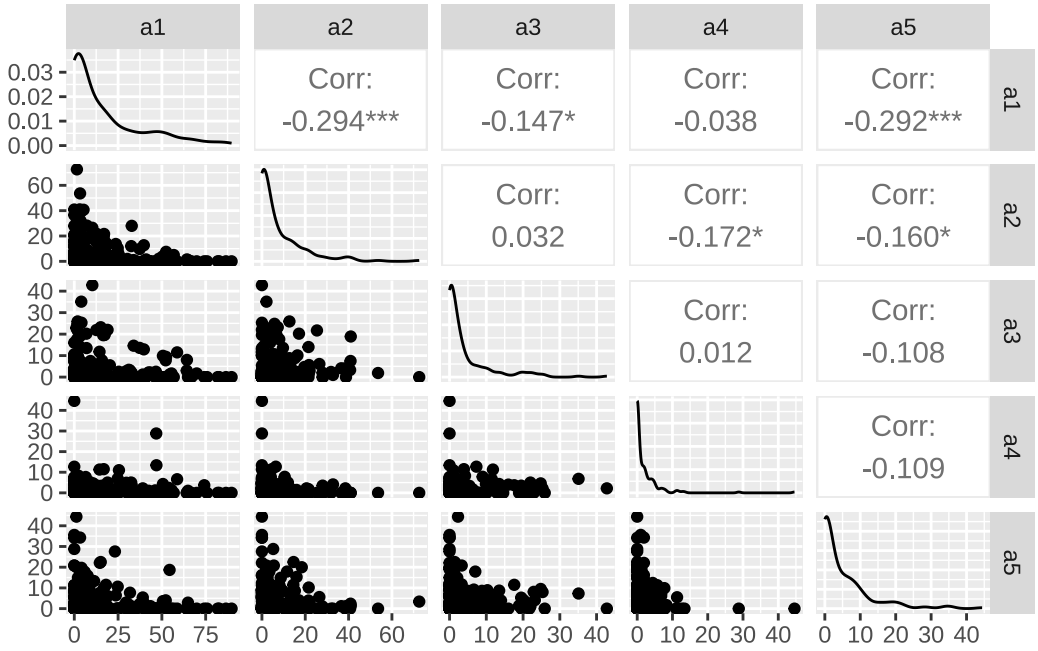
```
pairs(algae[, 12:16])
```




### Using the `ggplot2` extension `GGally`

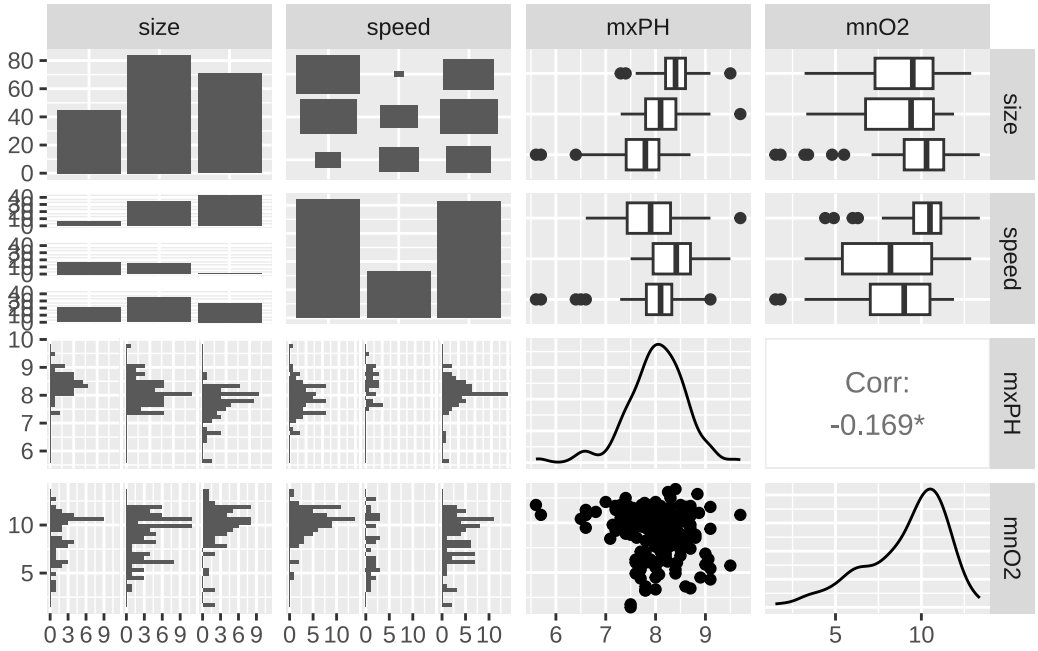
Pair-wise scatter plots in the lower triangle. On the diagonal, continuous approximation of the distribution of the respective variable is plotted. Upper triangle shows the correlation between two respective variables.

```
ggpairs(algae, columns = 12:16)
```




If variables paired up contains nominal and continuous variable, you will get different types of plots that make sense for respective variable types.

```
ggpairs(algae, columns = 2:5)
```




**`qqplot` compare two distributions: two vectors with values sorted from small to large**

Different from scatter plots – value pairs in scatter plots belong to one observation.

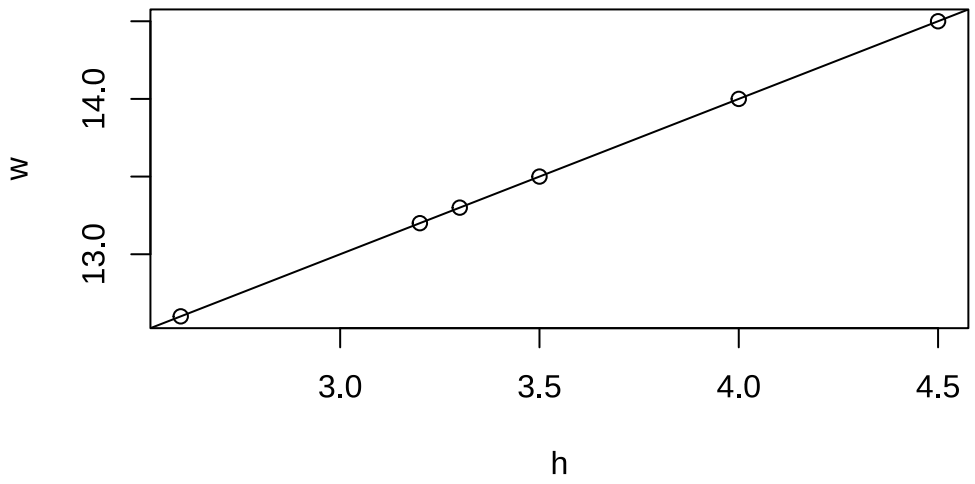
```
h <- c(3.5, 2.6, 4.0, 3.2, 4.5, 3.3 ) # height values

length(h)
```

```
[1] 6
```

```
w <- c(13.5, 12.6, 14.0, 13.2, 14.5, 13.3) # weight values

length(w)
```

```
[1] 6
```

```r
qqplot(h, w) # h and w values are sorted, paired, and then plotted

abline(lsfit(h, w)) # fit a line using the least square24
```

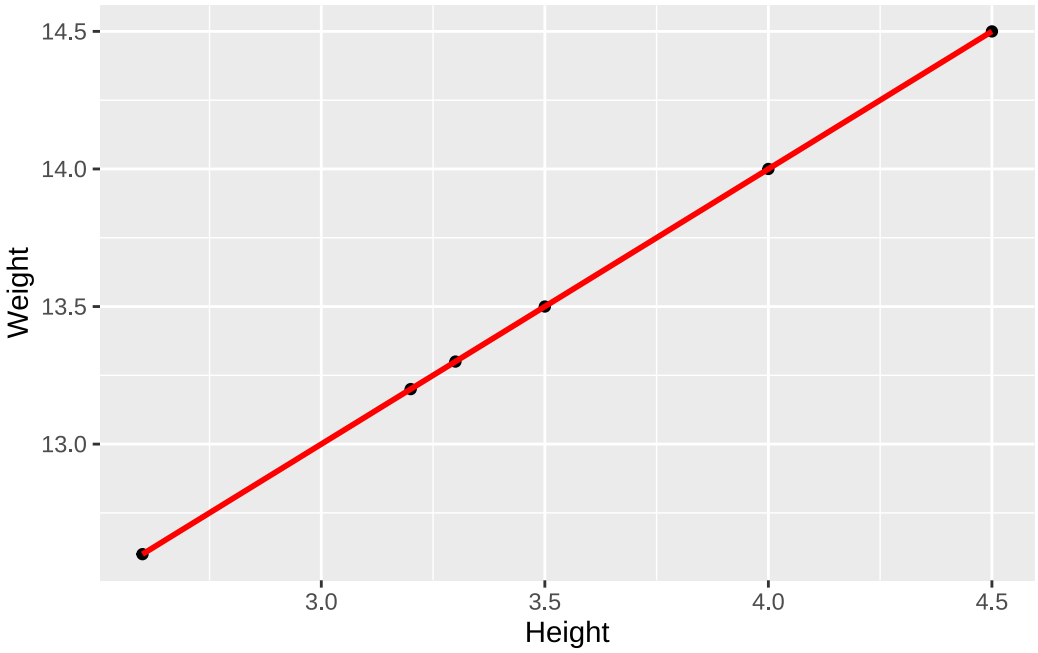



**Or... use ggplot2**

```r
library(ggplot2)

# create a data frame from the data
df <- data.frame(h = c(3.5, 2.6, 4.0, 3.2, 4.5, 3.3),
                 w = c(13.5, 12.6, 14.0, 13.2, 14.5, 13.3))

# sort and rank the data
df <- df %>% mutate(rank = rank(h),
                    h = sort(h),
                    w = sort(w))

# plot the data
ggplot(df, aes(x = h, y = w)) +
  geom_point() +
```

```
    geom_smooth(method = "lm", se = FALSE, color = "red") +
    labs(x = "Height", y = "Weight")
```

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




## [Advanced]

### Getting to know your dataset:

1. List data types of the attributes in your tidy dataset
2. Check for skewness in data distribution in the attributes
3. Check for correlations among attributes
4. Examine the extent of missing data. What would be the best way to deal with the missing data in this case?