# R Workshop

*Vanessa*

*2019-01-21*

## Creating Objects in R

Super basic, using R as a fancy calculator:

```r
3+5
```

```
## [1] 8
```

```r
12/7
```

```
## [1] 1.714286
```

```r
5*5
```

```
## [1] 25
```

```r
# assign value to object/variable
weight_kg <- 55

2.2*weight_kg
```

```
## [1] 121
```

```r
weight_lb <- 2.2*weight_kg

sqrt(weight_kg)
```

```
## [1] 7.416198
```

```r
round(pi)
```

```
## [1] 3
```

```r
round(3.14159)
```

```
## [1] 3
```

```r
round(3.14159, digits=2)
```

```
## [1] 3.14
```

```r
round(pi, digits = 6)
```

```
## [1] 3.141593
```

```r
round(pi, 10)
```

```
## [1] 3.141593
```

### Vectors and Data Types

This section will describe some basic data types in R:

```r
weight_g <- c(50, 60, 65, 82)

animals <- c("mouse", "rat", "dog")
```

Vector types in R:

- numeric
- character
- logical (TRUE or FALSE)
- factors (categorical data i.e. species)
- Dates

A vector is a data structure in R.

Other data structures:

- lists
- data frames
- matrices
- of course vectors

Often you want to convert lists and matrices to data frames or vectors.

## Data Frames

Next we're going to look at the structure of Data Frames.

```r
library(tidyverse)

download.file(url="https://ndownloader.figshare.com/files/2292169", destfile = "read_data/portal_data_jo

library(here)
#this package makes working directories and file paths easy

surveys <- read_csv(here("read_data", "portal_data_joined.csv"))
# THIS PART IS NEW AND RELEVANT TO ME

str(surveys)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    34786 obs. of  13 variables:
##  $ record_id      : int  1 72 224 266 349 363 435 506 588 661 ...
##  $ month          : int  7 8 9 10 11 11 12 1 2 3 ...
##  $ day            : int  16 19 13 16 12 12 10 8 18 11 ...
##  $ year           : int  1977 1977 1977 1977 1977 1977 1977 1978 1978 1978 ...
##  $ plot_id        : int  2 2 2 2 2 2 2 2 2 2 ...
##  $ species_id     : chr  "NL" "NL" "NL" "NL" ...
##  $ sex            : chr  "M" "M" NA NA ...
##  $ hindfoot_length: int  32 31 NA NA NA NA NA NA NA NA ...
##  $ weight         : int  NA NA NA NA NA NA NA NA 218 NA ...
##  $ genus          : chr  "Neotoma" "Neotoma" "Neotoma" "Neotoma" ...
##  $ species        : chr  "albigula" "albigula" "albigula" "albigula" ...
##  $ taxa           : chr  "Rodent" "Rodent" "Rodent" "Rodent" ...
##  $ plot_type      : chr  "Control" "Control" "Control" "Control" ...
##  - attr(*, "spec")=List of 2
##   ..$ cols   :List of 13
##   .. ..$ record_id      : list()
##   .. .. ..- attr(*, "class")= chr  "collector_integer" "collector"
##   .. ..$ month          : list()
##   .. .. ..- attr(*, "class")= chr  "collector_integer" "collector"
##   .. ..$ day            : list()
##   .. .. ..- attr(*, "class")= chr  "collector_integer" "collector"
##   .. ..$ year           : list()
##   .. .. ..- attr(*, "class")= chr  "collector_integer" "collector"
##   .. ..$ plot_id        : list()
##   .. .. ..- attr(*, "class")= chr  "collector_integer" "collector"
##   .. ..$ species_id     : list()
##   .. .. ..- attr(*, "class")= chr  "collector_character" "collector"
##   .. ..$ sex            : list()
##   .. .. ..- attr(*, "class")= chr  "collector_character" "collector"
##   .. ..$ hindfoot_length: list()
##   .. .. ..- attr(*, "class")= chr  "collector_integer" "collector"
##   .. ..$ weight         : list()
##   .. .. ..- attr(*, "class")= chr  "collector_integer" "collector"
##   .. ..$ genus          : list()
##   .. .. ..- attr(*, "class")= chr  "collector_character" "collector"
##   .. ..$ species        : list()
##   .. .. ..- attr(*, "class")= chr  "collector_character" "collector"
##   .. ..$ taxa           : list()
##   .. .. ..- attr(*, "class")= chr  "collector_character" "collector"
##   .. ..$ plot_type      : list()
##   .. .. ..- attr(*, "class")= chr  "collector_character" "collector"
##   ..$ default: list()
##   .. ..- attr(*, "class")= chr  "collector_guess" "collector"
##   ..- attr(*, "class")= chr "col_spec"
```

```
dim(surveys)
```

```
## [1] 34786    13
```

```
nrow(surveys)
```

```
## [1] 34786
```

```
ncol(surveys)
```

```
## [1] 13
```

```
summary(surveys)
```

```
##    record_id         month              day             year     
##  Min.   :    1   Min.   : 1.000   Min.   : 1.0   Min.   :1977  
##  1st Qu.: 8964   1st Qu.: 4.000   1st Qu.: 9.0   1st Qu.:1984  
##  Median :17762   Median : 6.000   Median :16.0   Median :1990  
##  Mean   :17804   Mean   : 6.474   Mean   :16.1   Mean   :1990  
##  3rd Qu.:26655   3rd Qu.:10.000   3rd Qu.:23.0   3rd Qu.:1997  
##  Max.   :35548   Max.   :12.000   Max.   :31.0   Max.   :2002  
##                                                                
##     plot_id       species_id            sex            hindfoot_length
##  Min.   : 1.00   Length:34786       Length:34786       Min.   : 2.00  
##  1st Qu.: 5.00   Class :character   Class :character   1st Qu.:21.00  
##  Median :11.00   Mode  :character   Mode  :character   Median :32.00  
##  Mean   :11.34                                         Mean   :29.29  
##  3rd Qu.:17.00                                         3rd Qu.:36.00  
##  Max.   :24.00                                       Max.   :70.00
##                                                      NA's   :3348
##      weight          genus             species               taxa
##  Min.   :  4.00   Length:34786       Length:34786       Length:34786
##  1st Qu.: 20.00   Class :character   Class :character   Class :character
##  Median : 37.00   Mode  :character   Mode  :character   Mode  :character
##  Mean   : 42.67
##  3rd Qu.: 48.00
##  Max.   :280.00
##  NA's   :2503
##   plot_type
##  Length:34786
##  Class :character
##  Mode  :character
##
##
##
##
```

### Indexing and Subsetting Data Frames

First let's use square bracket subsetting.

Square brackets are great for defining coordinates to extract data from. But what happens when the structure of the data frame changes.

```
#first define the row coordinate and then the column
#also write row and then column
surveys[1, 1]
```

```
## # A tibble: 1 x 1
##   record_id
##       <int>
## 1         1
```

```
surveys[1, 6]
```

```
## # A tibble: 1 x 1
##   species_id
##   <chr>
## 1 NL
```

```
#defining only which element we want will return a data frame
surveys[1]
```

```
## # A tibble: 34,786 x 1
##    record_id
##        <int>
##  1         1
##  2        72
##  3       224
##  4       266
##  5       349
##  6       363
##  7       435
##  8       506
## 9        588
## 10       661
## # ... with 34,776 more rows
```

```
surveys[1:3, 7]
```

```
## # A tibble: 3 x 1
##   sex
##   <chr>
## 1 M
## 2 M
## 3 <NA>
```

```
#give us all the rows and columns except column 7
surveys[, -7]
```

```
## # A tibble: 34,786 x 12
##    record_id month   day  year plot_id species_id hindfoot_length weight
##        <int> <int> <int> <int>   <int> <chr>                <int>  <int>
##  1         1     7    16  1977       2 NL                      32     NA
##  2        72     8    19  1977       2 NL                      31     NA
##  3       224     9    13  1977       2 NL                      NA     NA
##  4       266    10    16  1977       2 NL                      NA     NA
##  5       349    11    12  1977       2 NL                      NA     NA
##  6       363    11    12  1977       2 NL                      NA     NA
##  7       435    12    10  1977       2 NL                      NA     NA
##  8       506     1     8  1978       2 NL                      NA     NA
##  9       588     2    18  1978       2 NL                      NA    218
## 10       661     3    11  1978       2 NL                      NA     NA
## # ... with 34,776 more rows, and 4 more variables: genus <chr>,
## #   species <chr>, taxa <chr>, plot_type <chr>
```

```
surveys[, -c(1:3)]
```

```
## # A tibble: 34,786 x 10
##     year plot_id species_id sex   hindfoot_length weight genus   species
##    <int>   <int> <chr>      <chr>           <int>  <int> <chr>   <chr>
##  1  1977       2 NL         M                  32     NA Neotoma albigula
##  2  1977       2 NL         M                  31     NA Neotoma albigula
##  3  1977       2 NL         <NA>               NA     NA Neotoma albigula
##  4  1977       2 NL         <NA>               NA     NA Neotoma albigula
##  5  1977       2 NL         <NA>               NA     NA Neotoma albigula
##  6  1977       2 NL         <NA>               NA     NA Neotoma albigula
##  7  1977       2 NL         <NA>               NA     NA Neotoma albigula
##  8  1978       2 NL         <NA>               NA     NA Neotoma albigula
##  9  1978       2 NL         M                  NA    218 Neotoma albigula
## 10  1978       2 NL         <NA>               NA     NA Neotoma albigula
## # ... with 34,776 more rows, and 2 more variables: taxa <chr>,
## #   plot_type <chr>
```

## Data Manipulation

Key functions for data manipulation:

- `select()`: subsetting columns
- `filter()`: subsets of rows based on conditions

- `mutate()`: create new columns, based on information from other columns
- `group_by()`: creates groups based on categorical data in a column
- `summarize()`: creates summary stats on grouped data
- `arrange()`: sort results
- `count()`: gives a count of discrete values

```
select(surveys, plot_id, species_id, weight)
```

```
## # A tibble: 34,786 x 3
##    plot_id species_id weight
##      <int> <chr>       <int>
##  1       2 NL             NA
##  2       2 NL             NA
##  3       2 NL             NA
##  4       2 NL             NA
##  5       2 NL             NA
##  6       2 NL             NA
##  7       2 NL             NA
##  8       2 NL             NA
##  9       2 NL            218
## 10       2 NL             NA
## # ... with 34,776 more rows
```

```
#negative subsetting
select(surveys, -record_id)
```

```
## # A tibble: 34,786 x 12
##    month   day  year plot_id species_id sex   hindfoot_length weight genus
##    <int> <int> <int>   <int> <chr>      <chr>           <int>  <int> <chr>
##  1     7    16  1977       2 NL         M                  32     NA Neot~
##  2     8    19  1977       2 NL         M                  31     NA Neot~
##  3     9    13  1977       2 NL         <NA>               NA     NA Neot~
##  4    10    16  1977       2 NL         <NA>               NA     NA Neot~
##  5    11    12  1977       2 NL         <NA>               NA     NA Neot~
##  6    11    12  1977       2 NL         <NA>               NA     NA Neot~
##  7    12    10  1977       2 NL         <NA>               NA     NA Neot~
##  8     1     8  1978       2 NL         <NA>               NA     NA Neot~
##  9     2    18  1978       2 NL         M                  NA    218 Neot~
## 10     3    11  1978       2 NL         <NA>               NA     NA Neot~
## # ... with 34,776 more rows, and 3 more variables: species <chr>,
## #   taxa <chr>, plot_type <chr>
```

```
filter(surveys, year==1995,
       species_id=="NL")
```

```
## # A tibble: 8 x 13
##   record_id month   day  year plot_id species_id sex   hindfoot_length
##       <int> <int> <int> <int>   <int> <chr>      <chr>           <int>
## 1     22314     6     7  1995       2 NL         M                  34
## 2     22728     9    23  1995       2 NL         F                  32
## 3     22899    10    28  1995       2 NL         F                  32
## 4     23032    12     2  1995       2 NL         F                  33
## 5     22847    10    28  1995      12 NL         M                  34
## 6     22998    12     2  1995      12 NL         M                  33
## 7     23124    12    21  1995      12 NL         F                  32
## 8     22476     7    20  1995      24 NL         F                  31
## # ... with 5 more variables: weight <int>, genus <chr>, species <chr>,
## #   taxa <chr>, plot_type <chr>
```

## Pipes

Pipes allow you to chain together dplyr functions.

Pipe: %>% or cmd-shift-m

```
#write multiple arguments in a sentence using pipes
surveys %>%
  filter(weight<5) %>%
  select(species_id, sex, weight)
```

```
## # A tibble: 17 x 3
##    species_id sex   weight
##    <chr>      <chr>  <int>
##  1 PF         F          4
##  2 PF         F          4
##  3 PF         M          4
##  4 RM         F          4
##  5 RM         M          4
##  6 PF         <NA>       4
##  7 PP         M          4
##  8 RM         M          4
##  9 RM         M          4
## 10 RM         M          4
## 11 PF         M          4
## 12 PF         F          4
## 13 RM         M          4
## 14 RM         M          4
## 15 RM         F          4
## 16 RM         M          4
## 17 RM         M          4
```

```
surveys_sml <- surveys %>%
  filter(weight<5) %>%
  select(species_id, sex, weight)
```

Challenge #1

Using pipe, subset the surveys dataframe to include animals collected 1995 and retain only the columns year, sex and weight.

```
surveys %>%
  filter(year==1995) %>%
  select(year, sex, weight)
```

```
## # A tibble: 1,180 x 3
##     year sex   weight
##    <int> <chr>  <int>
##  1  1995 M         NA
##  2  1995 F        165
##  3  1995 F        171
##  4  1995 F         NA
##  5  1995 M         41
##  6  1995 F         45
##  7  1995 M         46
##  8  1995 F         49
##  9  1995 M         46
## 10  1995 M         48
## # ... with 1,170 more rows
```

```
surveys %>%
  mutate(weight_kg=weight/1000,
         weight_kg2=weight_kg*2)
```

```
## # A tibble: 34,786 x 15
##    record_id month   day  year plot_id species_id sex   hindfoot_length
##        <int> <int> <int> <int>   <int> <chr>      <chr>           <int>
##  1         1     7    16  1977       2 NL         M                  32
##  2        72     8    19  1977       2 NL         M                  31
##  3       224     9    13  1977       2 NL         <NA>               NA
##  4       266    10    16  1977       2 NL         <NA>               NA
##  5       349    11    12  1977       2 NL         <NA>               NA
##  6       363    11    12  1977       2 NL         <NA>               NA
##  7       435    12    10  1977       2 NL         <NA>               NA
##  8       506     1     8  1978       2 NL         <NA>               NA
##  9       588     2    18  1978       2 NL         M                  NA
## 10       661     3    11  1978       2 NL         <NA>               NA
## # ... with 34,776 more rows, and 7 more variables: weight <int>,
## #   genus <chr>, species <chr>, taxa <chr>, plot_type <chr>,
## #   weight_kg <dbl>, weight_kg2 <dbl>
```

```
surveys <- surveys %>%
  drop_na(weight) %>%
  mutate(mean_weight=mean(weight))
```

Challenge #2

Using the surveys data from create a new data frame that contains only the species_id column, has a new column called hindfoot_half: contains values that are half the hindfoot_length values. Also, in the new hindfoot_half column there are no NAs and values are all less than 30.

```
surveys_hindfoot_half <- surveys %>%
  drop_na(hindfoot_length) %>%
  mutate(hindfoot_half=hindfoot_length/2) %>%
  filter(hindfoot_half<30) %>%
  select(species_id, hindfoot_half, hindfoot_length)
```

```
surveys %>%
  group_by(sex) %>%
  summarize(mean_weight=mean(weight, na.rm = TRUE))
```

```
## # A tibble: 3 x 2
##   sex   mean_weight
##   <chr>       <dbl>
## 1 F            42.2
## 2 M            43.0
## 3 <NA>         64.7
```

```
surveys %>%
  group_by(sex, species_id) %>%
```

```r
  summarize(mean_weight=mean(weight, na.rm=TRUE),
            min_weight=min(weight, na.rm=TRUE)) %>%
  arrange(desc(min_weight))
```

```
## # A tibble: 64 x 4
## # Groups:   sex [3]
##    sex   species_id mean_weight min_weight
##    <chr> <chr>            <dbl>      <dbl>
##  1 M     SS               130          130
##  2 <NA>  SH               130          130
##  3 <NA>  NL               168.          83
##  4 <NA>  DS               120           78
##  5 F     SS                57           57
##  6 F     SF                69           46
##  7 F     DS               118.          45
##  8 <NA>  DO                50.7         44
##  9 <NA>  SF                40.5         36
## 10 M     SO                55.7         35
## # ... with 54 more rows
```

```r
surveys %>%
  count(sex, sort = TRUE)
```

```
## # A tibble: 3 x 2
##   sex       n
##   <chr> <int>
## 1 M     16879
## 2 F     15303
## 3 <NA>    101
```

```r
#the above code is synonomous with
surveys %>%
  group_by(sex) %>%
  summarise(count=n())
```

```
## # A tibble: 3 x 2
##   sex   count
##   <chr> <int>
## 1 F     15303
## 2 M     16879
## 3 <NA>    101
```

Challenge # 3

How many animals were caught in each plot_type surveyed.

```r
surveys %>%
  count(plot_type)
```

```
## # A tibble: 5 x 2
##   plot_type                     n
##   <chr>                     <int>
## 1 Control                   14652
## 2 Long-term Krat Exclosure   4692
## 3 Rodent Exclosure           3818
## 4 Short-term Krat Exclosure  5407
## 5 Spectab exclosure          3714
```

Use group_by and summarize to find the mean, min and max of hindfoot length (using species_id) for each species. Also, add the number of observations (hint: see ?n)

```
surveys %>%
  group_by(species_id) %>%
  summarise(mean_length=mean(hindfoot_length, na.rm = TRUE),
            min_length=min(hindfoot_length, na.rm = TRUE),
            max_length=max(hindfoot_length, na.rm = TRUE), n=n())
```

```
## # A tibble: 25 x 5
##    species_id mean_length min_length max_length     n
##    <chr>            <dbl>      <dbl>      <dbl> <int>
##  1 BA                13            6         16    45
##  2 DM                36.0         16         50 10262
##  3 DO                35.6         26         64  2904
##  4 DS                50.0         39         58  2344
##  5 NL                32.3         21         42  1152
##  6 OL                20.5         12         39   970
##  7 OT                20.3         13         50  2160
##  8 OX                20.4         19         21     6
##  9 PB                26.1          2         47  2810
## 10 PE                20.2         11         30  1260
## # ... with 15 more rows
```

What was the heaviest animal measured in each year? Return the columns year, genus, species_id and weight.

```
surveys %>%
  group_by(year) %>%
  summarise(genus=first(genus),
            species_id=first(species_id),
    max_weight=max(weight, na.rm=TRUE)) %>%
  select(year, genus, species_id, max_weight)
```

```
## # A tibble: 26 x 4
##     year genus     species_id max_weight
##    <int> <chr>     <chr>           <dbl>
##  1  1977 Dipodomys DM                149
##  2  1978 Neotoma   NL                232
##  3  1979 Neotoma   NL                274
##  4  1980 Neotoma   NL                243
##  5  1981 Neotoma   NL                264
##  6  1982 Neotoma   NL                252
##  7  1983 Neotoma   NL                256
##  8  1984 Neotoma   NL                259
##  9  1985 Neotoma   NL                225
## 10  1986 Neotoma   NL                240
## # ... with 16 more rows
```

```
#my version
#incorrect! grabbed the first name and filled in the relevant value
#all neotoma except 1977 where it displayed dipodomys because no neot.

max_weights <- surveys %>%
  drop_na(weight) %>%
  group_by(year) %>%
```

```r
  filter(weight==max(weight)) %>%
  select(year, genus, species_id, weight) %>%
  arrange(year) %>%
  unique()
#brett's version - correct way to do it! wouldn't thought of filter
```

### Export Our Data

```r
write_csv(max_weights, here("write_data", "max_weights.csv"))
```

Git history is stored locally here. Hit diff/history for more details.

## Day 2

### Tidy Data in Spreadsheets

Today we are going to look at tidying data.

The functions we use for tidying data are:

- tidyr::spread()
- tidyr::gather()

Note: order of libraries being loaded matters for select, summarize, etc.

#### spread()

spread() takes three principle arguments:

1. the data
2. the *key* column variable will become the new column names
3. the value column variable which will fill the new column variables

We're going to use the surveys dataset

```r
library(tidyverse)
library(here)

surveys <- read_csv(here("read_data", "surveys.csv"))

#create a wide data format of surveys using spread

surveys_gw <- surveys %>%
  drop_na(weight) %>%
  group_by(species_id) %>%
  summarise(mean_weight=mean(weight))

str(surveys_gw)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    25 obs. of  2 variables:
##  $ species_id : chr  "BA" "DM" "DO" "DS" ...
##  $ mean_weight: num  8.6 43.2 48.9 120.1 159.2 ...
```

```r
#now we want to spread the dataset out into a wide format

wide_surveys_gw <- surveys_gw %>%
  spread(key=species_id, value=mean_weight)
#long format is usually better, group together species for analysis

#now we are going back to long from the wide dataset that we just created
```

#### gather()

gather() takes four arugments:

1. data
2. key
3. value
4. names of columns we use to fill the key variable (or drop)

```r
long_surveys <- wide_surveys_gw %>%
  gather(key=species_id, value=mean_weight)
```

## Sending Tidy Data

### Changelog

- Update your changelog with changes you make to your raw data or other significant changes or additions to your projects!

### Data Dictionary

- Create a data dictionary to define our variables

```r
tidy_gsi <- read_csv(here("write_data", "tidy_gsi.csv"))
```

```
## Parsed with column specification:
## cols(
##   hakai_id = col_character(),
##   stock_1 = col_character(),
##   region_1 = col_integer(),
##   prob_1 = col_double(),
##   stock_2 = col_character(),
##   region_2 = col_integer(),
##   prob_2 = col_double(),
##   stock_3 = col_character(),
##   region_3 = col_integer(),
##   prob_3 = col_double(),
##   stock_4 = col_character(),
##   region_4 = col_integer(),
##   prob_4 = col_double(),
##   stock_5 = col_character(),
##   region_5 = col_integer(),
##   prob_5 = col_double()
## )
```

```r
#noticed an error because comment in column wasn't deleted and created NA's
```

To sync with GitHub: commit then pull then push (not pull then commit then push?)

Having trouble connecting github and local computer when project created locally…

But I think it's fixed now. Create Github project then upload local files there.

## Analyzing Data

### Importing from the Hakai Data Portal

Switched to data_wrangling script to import data into our read_data file.

We have chla data, fish data, and sockeye stock id data. Let's start to analyze these datasets.

```r
fish <- read_csv(here("read_data", "fish.csv"))
```

```
## Parsed with column specification:
## cols(
##   .default = col_character(),
##   date = col_date(format = ""),
##   fork_length_field = col_integer(),
##   height_field = col_integer(),
##   date_processed = col_date(format = ""),
##   weight = col_double(),
##   standard_length = col_integer(),
##   fork_length = col_integer()
## )
```

```
## See spec(...) for full column specifications.
```

```r
chla <- read_csv(here("read_data", "chla.csv"))
```

```
## Parsed with column specification:
## cols(
##   .default = col_character(),
##   no = col_integer(),
##   event_pk = col_integer(),
##   rn = col_integer(),
##   date = col_date(format = ""),
##   sampling_bout = col_integer(),
##   lat = col_double(),
##   long = col_double(),
##   line_out_depth = col_integer(),
##   volume = col_integer(),
##   collected = col_datetime(format = ""),
##   preserved = col_datetime(format = ""),
##   analyzed = col_datetime(format = ""),
##   acetone_volume_ml = col_integer(),
##   flurometer_serial_no = col_integer(),
##   calibration = col_datetime(format = ""),
##   acid_ratio_correction_factor = col_double(),
##   acid_coefficient = col_double(),
##   calibration_slope = col_double(),
##   before_acid = col_double(),
##   after_acid = col_double()
##   # ... with 4 more columns
## )
## See spec(...) for full column specifications.
```

```
## Warning in rbind(names(probs), probs_f): number of columns of result is not
## a multiple of vector length (arg 1)
```

```
## Warning: 157 parsing failures.
## row # A tibble: 5 x 5 col     row col               expected               actual file
## ... ................. ... ..........................................................................
## See problems(...) for more details.
```

```
tidy_gsi <- read_csv(here("write_data", "tidy_gsi.csv"))
```

```
## Parsed with column specification:
## cols(
##   hakai_id = col_character(),
##   stock_1 = col_character(),
##   region_1 = col_integer(),
##   prob_1 = col_double(),
##   stock_2 = col_character(),
##   region_2 = col_integer(),
##   prob_2 = col_double(),
##   stock_3 = col_character(),
##   region_3 = col_integer(),
##   prob_3 = col_double(),
##   stock_4 = col_character(),
##   region_4 = col_integer(),
##   prob_4 = col_double(),
##   stock_5 = col_character(),
##   region_5 = col_integer(),
##   prob_5 = col_double()
## )
```

```
fish %>%
  count(species)
```

```
## # A tibble: 6 x 2
##   species     n
##   <chr>   <int>
## 1 CK         12
## 2 CO         98
## 3 CU       1689
## 4 HE        282
## 5 PI        860
## 6 SO       3497
```

```
fish %>%
  group_by(site_id) %>%
  count(species)
```

```
## # A tibble: 95 x 3
## # Groups:   site_id [20]
##    site_id species     n
##    <chr>   <chr>   <int>
##  1 D01     CO          1
##   2 D01     CU          10
##   3 D01     HE           2
##   4 D01     SO          30
##   5 D02     CU          30
##   6 D02     HE           1
##   7 D02     SO          68
##   8 D03     CO           1
##   9 D03     CU          21
##  10 D03     HE           4
## # ... with 85 more rows
```

```
fish_d09 <- fish %>%
  filter(site_id=="D09") %>%
  select(hakai_id, jsp_survey_id, seine_id, date,
         species, site_id, fork_length, weight) %>%
  mutate(k=(10^5*weight)/fork_length^3) %>%
  drop_na(k)
#fulton's condition factor calculation
```

## Factors

read_csv() reads words in as characters so you can determine what are factors read.csv() reads words in as factors as default, not ideal.

```
str(fish_d09)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    832 obs. of  9 variables:
##  $ hakai_id     : chr  "U4802" "U4776" "U4728" "U4801" ...
##  $ jsp_survey_id: chr  "DE112" "DE112" "DE112" "DE112" ...
##  $ seine_id     : chr  "DE112N1" "DE112N1" "DE112N1" "DE112N1" ...
##  $ date         : Date, format: "2015-05-20" "2015-05-20" ...
##  $ species      : chr  "SO" "SO" "SO" "SO" ...
##  $ site_id      : chr  "D09" "D09" "D09" "D09" ...
##  $ fork_length  : int  106 106 97 102 102 97 96 95 128 101 ...
##  $ weight       : num  10.1 11.3 8.8 9.9 8.7 8.4 7.9 8.1 19 9.8 ...
##  $ k            : num  0.848 0.949 0.964 0.933 0.82 ...
```

```
class(fish_d09$species)
```

```
## [1] "character"
```

```
#coerce a column to be a factor, do this:

fish_d09$species <- factor(fish_d09$species)

class(fish_d09$species)
```

```
## [1] "factor"
```

```
levels(fish_d09$species)
```

```
## [1] "CO" "CU" "HE" "PI" "SO"
```

If you have factors that are numbers, don't try to do math with those factors. Under the hood R will be treating your factor levels as a different number. Gets messy.

Ask a question about most effectively/quickly releveling the order of factors?

## Dates

read_csv() treats ISO date standards (yyyy-mm-dd) as a DATE object read.csv() treats them as characters: not ideal.

Lubridate is a package for dealing with dates.

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'

## The following object is masked from 'package:here':
##
##     here

## The following object is masked from 'package:base':
##
##     date
```

```
#extract date components
fish_d09 <- fish_d09 %>%
  mutate(year=year(date),
         month=month(date),
         week=week(date),
         yday=yday(date))
#very helpful library! use yday when comparing dates between years
```

Lubridate: you can do math with dates

- periods
- intervals
- durations

# Joining Data

Data we have:

- chla
- tidy_gsi
- fish_d09

```
left_join(fish_d09, tidy_gsi, by="hakai_id")
```

```
## # A tibble: 832 x 28
##    hakai_id jsp_survey_id seine_id date       species site_id fork_length
##    <chr>    <chr>         <chr>    <date>     <fct>   <chr>         <int>
##  1 U4802    DE112         DE112N1  2015-05-20 SO      D09             106
##  2 U4776    DE112         DE112N1  2015-05-20 SO      D09             106
##  3 U4728    DE112         DE112N1  2015-05-20 SO      D09              97
##  4 U4801    DE112         DE112N1  2015-05-20 SO      D09             102
##  5 U4777    DE112         DE112N1  2015-05-20 SO      D09             102
##  6 U4779    DE112         DE112N1  2015-05-20 SO      D09              97
##  7 U4778    DE112         DE112N1  2015-05-20 SO      D09              96
##  8 U4800    DE112         DE112N1  2015-05-20 SO      D09              95
##  9 U4780    DE112         DE112N1  2015-05-20 SO      D09             128
## 10 U350     DE112         DE112N1  2015-05-20 SO      D09             101
## # ... with 822 more rows, and 21 more variables: weight <dbl>, k <dbl>,
## #   year <dbl>, month <dbl>, week <dbl>, yday <dbl>, stock_1 <chr>,
## #   region_1 <int>, prob_1 <dbl>, stock_2 <chr>, region_2 <int>,
## #   prob_2 <dbl>, stock_3 <chr>, region_3 <int>, prob_3 <dbl>,
## #   stock_4 <chr>, region_4 <int>, prob_4 <dbl>, stock_5 <chr>,
## #   region_5 <int>, prob_5 <dbl>
```

```
right_join(fish_d09, tidy_gsi, by="hakai_id")
```

```
## # A tibble: 1,187 x 28
##    hakai_id jsp_survey_id seine_id date       species site_id fork_length
##    <chr>    <chr>         <chr>    <date>     <fct>   <chr>         <int>
##  1 U10      <NA>          <NA>     NA         <NA>    <NA>             NA
##  2 U16      <NA>          <NA>     NA         <NA>    <NA>             NA
##  3 U17      <NA>          <NA>     NA         <NA>    <NA>             NA
##  4 U21      <NA>          <NA>     NA         <NA>    <NA>             NA
##  5 U25      <NA>          <NA>     NA         <NA>    <NA>             NA
##  6 U31      <NA>          <NA>     NA         <NA>    <NA>             NA
##  7 U35      <NA>          <NA>     NA         <NA>    <NA>             NA
##  8 U42      <NA>          <NA>     NA         <NA>    <NA>             NA
##  9 U43      <NA>          <NA>     NA         <NA>    <NA>             NA
## 10 U7       <NA>          <NA>     NA         <NA>    <NA>             NA
## # ... with 1,177 more rows, and 21 more variables: weight <dbl>, k <dbl>,
## #   year <dbl>, month <dbl>, week <dbl>, yday <dbl>, stock_1 <chr>,
## #   region_1 <int>, prob_1 <dbl>, stock_2 <chr>, region_2 <int>,
## #   prob_2 <dbl>, stock_3 <chr>, region_3 <int>, prob_3 <dbl>,
## #   stock_4 <chr>, region_4 <int>, prob_4 <dbl>, stock_5 <chr>,
## #   region_5 <int>, prob_5 <dbl>
```

```
anti_join(fish_d09, tidy_gsi, by="hakai_id")
```

```
## # A tibble: 685 x 13
##    hakai_id jsp_survey_id seine_id date       species site_id fork_length
##    <chr>    <chr>         <chr>    <date>     <fct>   <chr>         <int>
##  1 U4802    DE112         DE112N1  2015-05-20 SO      D09             106
##  2 U4776    DE112         DE112N1  2015-05-20 SO      D09             106
##  3 U4728    DE112         DE112N1  2015-05-20 SO      D09              97
##  4 U4801    DE112         DE112N1  2015-05-20 SO      D09             102
##  5 U4777    DE112         DE112N1  2015-05-20 SO      D09             102
##  6 U4779    DE112         DE112N1  2015-05-20 SO      D09              97
##  7 U4778    DE112         DE112N1  2015-05-20 SO      D09              96
##  8 U4800    DE112         DE112N1  2015-05-20 SO      D09              95
##  9 U4780    DE112         DE112N1  2015-05-20 SO      D09             128
## 10 U348     DE112         DE112N1  2015-05-20 SO      D09              94
## # ... with 675 more rows, and 6 more variables: weight <dbl>, k <dbl>,
## #   year <dbl>, month <dbl>, week <dbl>, yday <dbl>
```

```
so_gsi <- inner_join(fish_d09, tidy_gsi, by="hakai_id")
```

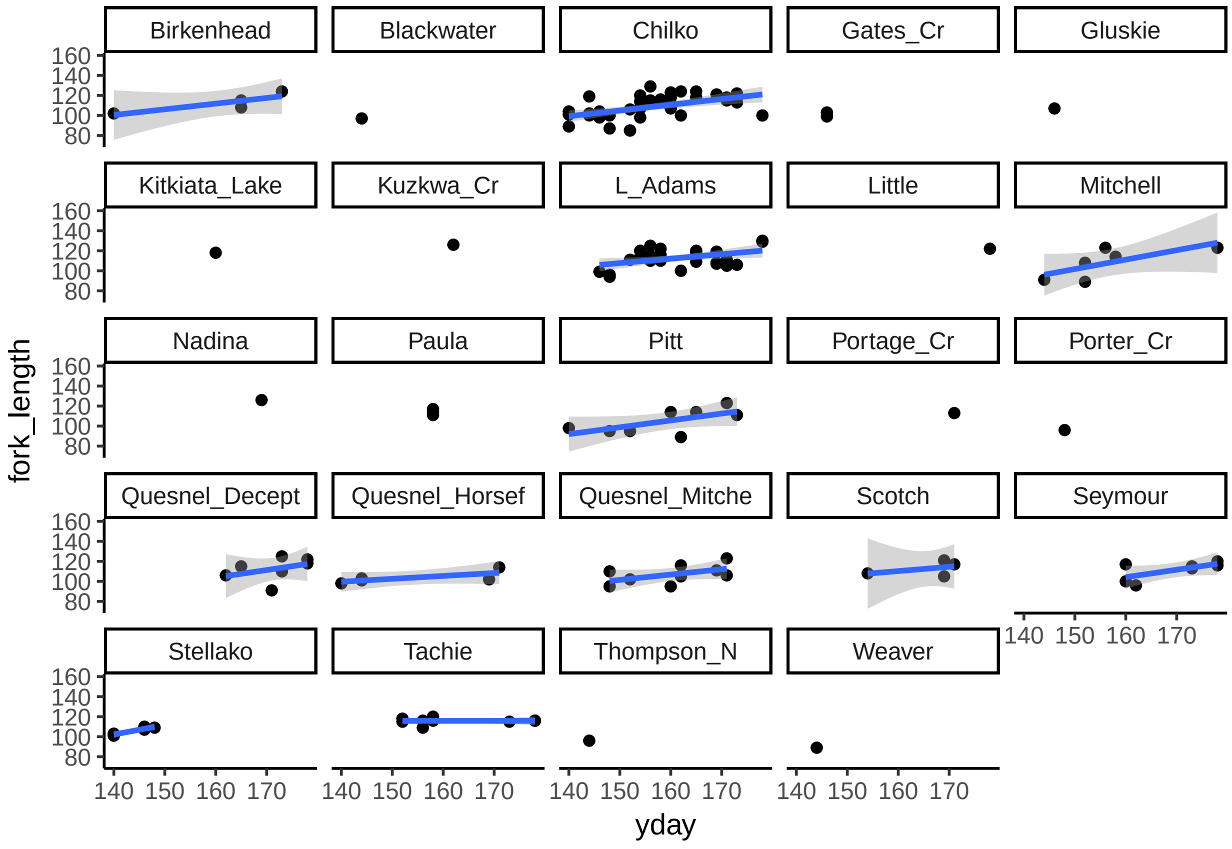
## ggplot2

To build a ggplot:

ggplot(data=DATA, mapping=aes(MAPPINGS)) + GEOM_FUNCTION()

```
#example
ggplot(data=surveys, mapping = aes(species_id, weight))+
  geom_point()
```

```
ggplot(so_gsi, aes())+
  geom_point(aes(x=yday, y=fork_length))+
  geom_smooth(aes(x=yday, y=fork_length), method = lm)+
  theme_classic()+
  facet_wrap(~stock_1)
```



Cookbook for R

How to adjust legends, make facets, adjust axes

http://cookbook-r.com/Graphs

Geom list:

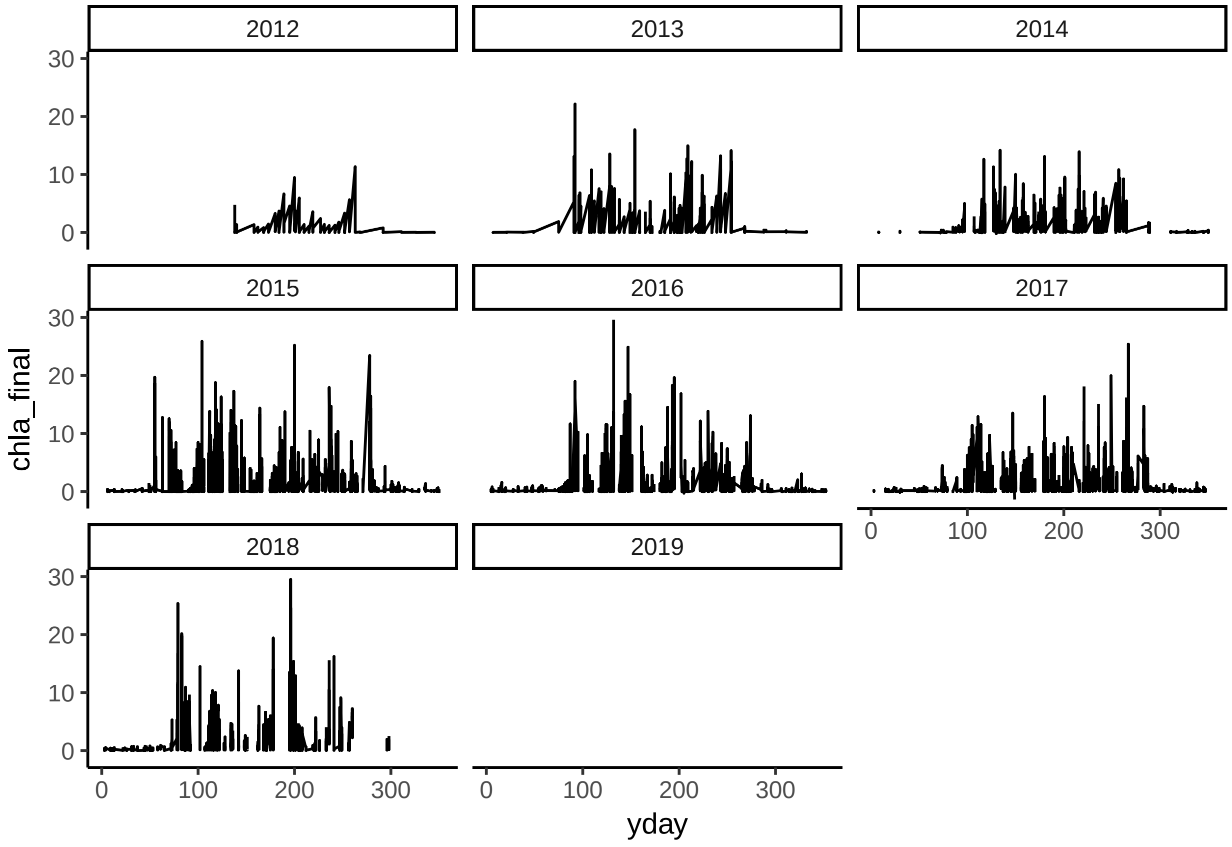
https://ggplot2.tidyverse.org/reference/

```
chla <- chla %>%
  mutate(year=year(date),
         month=month(date),
         week=week(date),
         yday=yday(date))
#separate date variables from main date column, same as so_gsi
```

```
chla_so_gsi <- inner_join(so_gsi, chla, by=c("date", "week", "year",
                                             "month", "yday"))
#didn't work
```

```r
ggplot(chla, aes())+
  geom_line(aes(x=yday, y=chla_final))+
  theme_classic()+
  facet_wrap(~year)
```

```
## Warning: Removed 334 rows containing missing values (geom_path).
```



```r
library(cowplot)
```

```
##
## Attaching package: 'cowplot'

## The following object is masked from 'package:ggplot2':
##
##     ggsave
```

```r
min(so_gsi$yday)
```

```
## [1] 140
```

```r
max(so_gsi$yday)
```

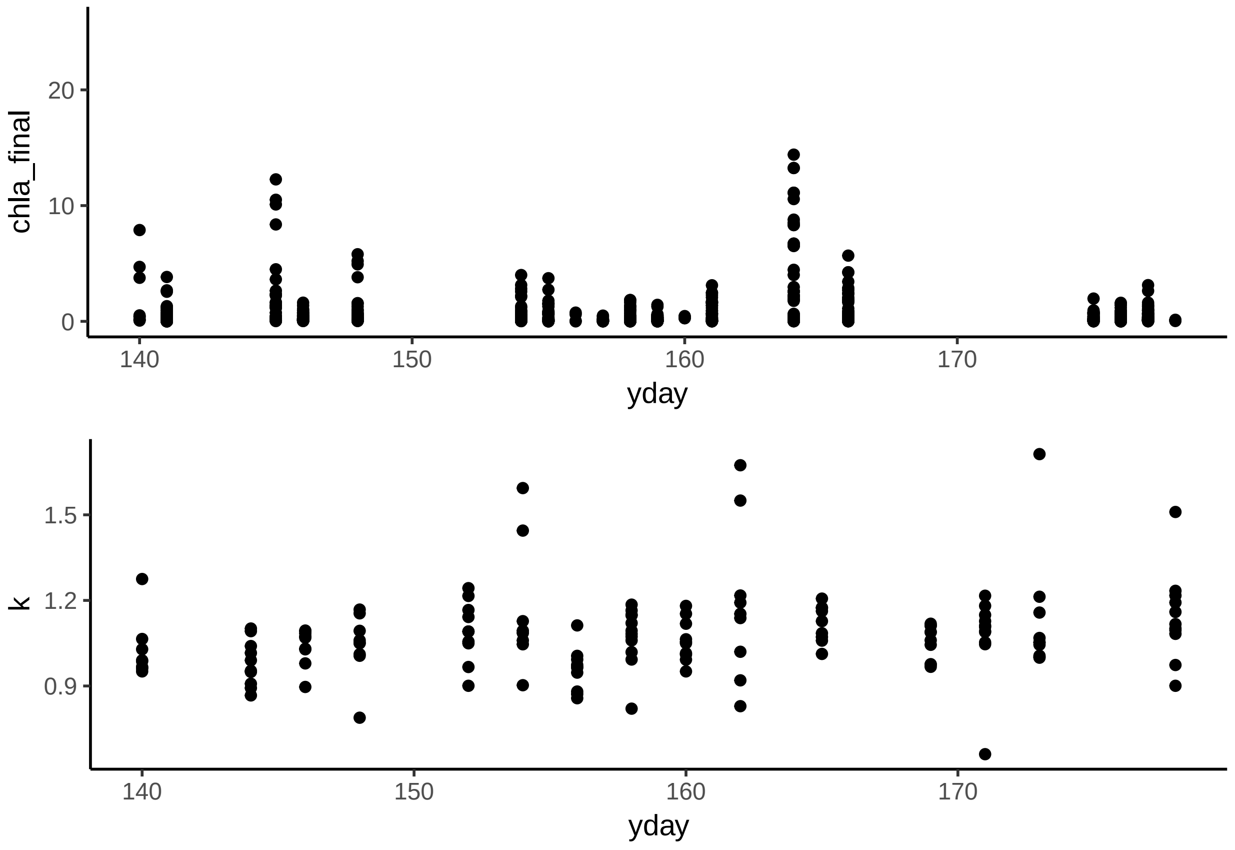
```
## [1] 178
```

```r
a <- chla %>%
  filter(year==2015) %>%
  ggplot()+
  geom_point(aes(x=yday, y=chla_final))+
  theme_classic()+
  scale_x_continuous(limits=c(140, 178))
```

```
b <- ggplot(so_gsi, aes())+
  geom_point(aes(x=yday, y=k))+
  theme_classic()

plot_grid(a, b, nrow=2)
```

```
## Warning: Removed 5748 rows containing missing values (geom_point).
```



```
#comparing chlorophyll in 2015 to fish condition (2015)
```