A little bit of everything!

QnA T2 Week 9

```
library(tidyverse)
library(gt)
library(palmerpenguins)
library(corrplot)
library(ggeasy)
library(ggpubr)
library(janitor)
library(psych)
```

Exploratory analyses tips

- for each research question, create a mini dataset so it's easy to work with especially if you're in a group that has like 100 variables
- tips from Jenny:
  - not necessary to have literature to justify every question, just make sure it's a genuine/authentic question
  - questions do not have to produce entirely different plots, just have to show off a variety of skills

How to create groups?

- example: if one of your variables of interest is age and you want to create different age groups, what should you do?
- keep in mind, there is not one right answer, there are pros and cons to each method, but this is a decision that is up to you. Some options are:
  - create your own groups - mutate(age_group = case_when())
  - for an example of using case_when() see notes from Week 5 here (under descriptive stats)
  - cut_number() - creates groups that are relatively equal, but age ranges are not necessarily equal
  - cut_interval() - equal age ranges, but not necessarily equal number in each group
- Remember, there's not 1 "right" way to answer a research question
  - for example, age can be treated as a continuous variable OR you can create a new variable where you group age into categories… this is up to you as the researcher!

knit to pdf and word

- Step 1: run this line of code in your console: `tinytex::install_tinytex()` (this may take several minutes to install)
- Step 2: Close and re-open RStudio to restart your session

- Step 3: Check the knit drop down, what options do you see?
  - Try knitting your file to pdf and word if you see those options (this will automatically change the code at the very top of your .Rmd file and that's okay)
  - You can also change the output manually to one of the following and then just click the knit button
  - `output: html_document`
  - `output: pdf_document`
  - `output: word_document`
- Step 4: If this work, double check your output file and make sure everything looks good (double check your gt table is there if you use the gt function, if it's not see the notes below)
- Step 5: If you still can't successfully knit your file to word or pdf, check out this guide I made last term for possible solutions: https://rpubs.com/jsloane/knit__pdf__word
- Step 6: If none of the above work, message me on slack
- **make sure to carefully review your knitting document**
  - make sure the code isn't cut off (I think word is better at not cutting off code compared to pdf. You can also try writing your code on multiple lines rather than 1 long line)
  - make sure all of your figures and tables are there

What to do with gt() tables when knitting to word/pdf?

- first, test it out to see if it works in either pdf or word (again make sure to actually look at the output file)
- if it doesn't, rather than spending too much time trying to figure out why, I suggest including a screenshot of your gt table like this `![Caption for the picture.](/path/to/image.png)`
- In the end, you want to make sure to have your code and table visible in your final report
- If you try to knit your file and you get an error in the chunk with your gt code, set the chunk option to `eval = FALSE` (just make sure your gt code is in its own chunk). This way, you'll be able to show off your code and also include your final table
- Examples below (both work when knitting to html):
  - 1 simple table that seems to knit to pdf, but not word
  - 1 more complex table that doesn't knit to either word or pdf

```
my_table <- exibble %>%
  mutate(perc = c(10, 20, 30, 40, 50, 60, 70, 80),
         sd = c(.1, .1, .25, .2, .1, .1, .3, .25))

# this works if I try knitting to pdf
my_table %>%
  gt()

# this only seems to work for html
my_table %>%
  gt(
    rowname_col = "row",
    groupname_col = "group"
  ) %>%
  cols_hide( # hide columns
    columns = vars(date, time, fctr, char, datetime) # make sure to use vars()
  ) %>%
  fmt_number( # format numbers to 2 decimal places
```

```
    columns = vars(num),
    decimals = 2
  ) %>%
  fmt_number( ### get parentheses!
    columns = vars(sd),
    pattern = "({x})"
  )  %>%
  fmt_percent( # format percent
    columns = vars(perc),
    scale_values = FALSE, # by default this is set to try which multiplies the variable by 100
    decimals = 0
  ) %>%
  cols_merge( # merge columns into 1
     columns = vars(perc, sd) #,
    # pattern = "{1} ({2})" # or this also works to add parentheses in columns you are merging
  ) %>%
  cols_label( # added this
    perc = "% (sd)",
  ) %>%
  fmt_currency( # add currency
    columns = vars(currency),
    currency = "AUD"
  ) %>%
  tab_header( # title and subtitle
    title = md("Example Table from **exibble**"), # md is to allow markdown formatting
    subtitle = md("`exibble` is an gt dataset")
  ) %>%
  tab_footnote( # you can add 1 or more footnotes
    footnote = "These are lower prices",
    locations = cells_body(
      columns = vars(currency),
      rows = currency < 15
    )
  ) %>%
  tab_footnote(
    footnote = "These big numbers",
    locations = cells_body(
      columns = vars(num),
      rows = num > 500
    )
  )
```

More stats. . . .

Reporting Statistics in APA Style link here

When to use summary data?

- bar/column plot and/or a plot with just means and error bars
- for other plots (boxplot, violin plot, scatter plot) you want to show all of the data so you **don't** want to use summarized data here

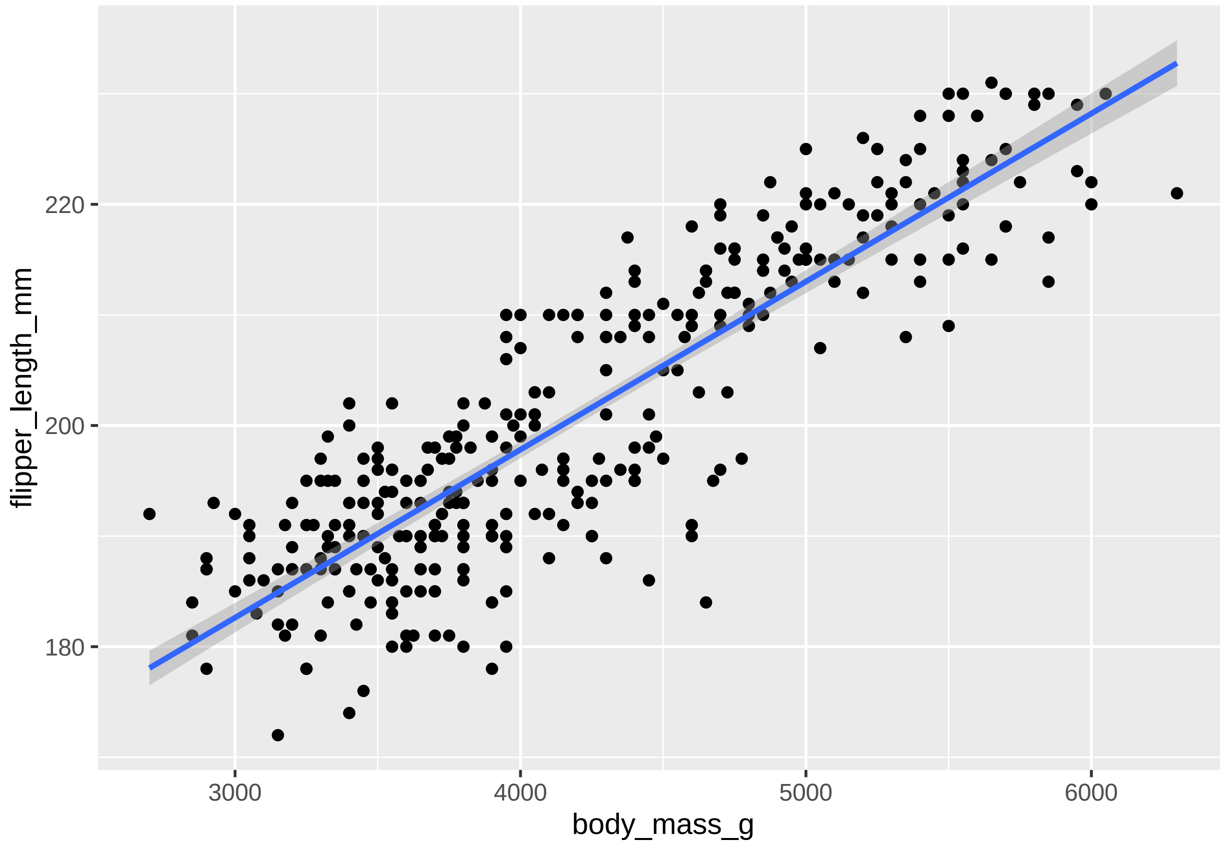
- for statistical analyses you probably also want to use the raw data (I typically think about it as wanting to have 1 data point/observation for participant or country or whatever you're interested in plotting)

```
# example of "summary data"
penguins <- penguins %>%
  na.omit()

body_mass_summary <- penguins %>%
  group_by(sex) %>%
  summarise(mean = mean(body_mass_g),
            sd = sd(body_mass_g),
            n = n(),
            se = sd/sqrt(n))
```

linear regression vs correlation

- correlation: are two variables related, as one increases does the other increase (or decrease?)
- linear regression: y = mx + b. this assumes your data is linear, but can be really useful if you're interested in prediction

```
penguins <- penguins %>%
  na.omit()

ggplot(penguins, aes(body_mass_g, flipper_length_mm)) +
  geom_point() +
  geom_smooth(method="lm")
```




```r
# m1
m1 <- lm(body_mass_g ~ flipper_length_mm, data = penguins)
summary(m1)
```

```
##
## Call:
## lm(formula = body_mass_g ~ flipper_length_mm, data = penguins)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1057.33  -259.79   -12.24   242.97  1293.89
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)        -5872.09     310.29  -18.93   <2e-16 ***
## flipper_length_mm     50.15       1.54   32.56   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 393.3 on 331 degrees of freedom
## Multiple R-squared:  0.7621, Adjusted R-squared:  0.7614
## F-statistic:  1060 on 1 and 331 DF,  p-value: < 2.2e-16
```

What to do with more than 1 predictor variable?

- in our m1, we were only looking to see if flipper length predicts body mass
- but what if we also want to include other variables like bill length and/or sex?
- if all the predictors are continuous, we will just expand our linear regression
- + to add in more predictors, will show main effects
- * to add in more predictors, while also looking for an interaction

Do flipper length and bill length predict body mass?

- this is actually not a great example because you don't want to include variables that are correlated with each other, but this is just an illustration

```
# 1 variable
m1 <- lm(body_mass_g ~ flipper_length_mm, data = penguins)

# 2 variables and just main effects
m2 <- lm(body_mass_g ~ flipper_length_mm + bill_length_mm, data = penguins)
summary(m2)
```

```
##
## Call:
## lm(formula = body_mass_g ~ flipper_length_mm + bill_length_mm,
##     data = penguins)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1083.08  -282.65   -17.18   242.95  1293.24
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)       -5836.299    312.604 -18.670   <2e-16 ***
## flipper_length_mm    48.890      2.034  24.034   <2e-16 ***
## bill_length_mm        4.959      5.214   0.951    0.342
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 393.4 on 330 degrees of freedom
## Multiple R-squared:  0.7627, Adjusted R-squared:  0.7613
## F-statistic: 530.4 on 2 and 330 DF,  p-value: < 2.2e-16
```

```
# 2 variables and main effects and interaction
m3 <- lm(body_mass_g ~ flipper_length_mm * bill_length_mm, data = penguins)
summary(m3)
```

```
##
## Call:
## lm(formula = body_mass_g ~ flipper_length_mm * bill_length_mm,
##     data = penguins)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
```

```
## -1036.28  -281.19   -22.75   232.21  1246.28
##
## Coefficients:
##                                Estimate Std. Error t value Pr(>|t|)
## (Intercept)                   4614.8799  2971.8474   1.553 0.121417
## flipper_length_mm               -4.6444    15.2726  -0.304 0.761244
## bill_length_mm                -221.4657    64.2446  -3.447 0.000640 ***
## flipper_length_mm:bill_length_mm    1.1549     0.3266   3.536 0.000465 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 386.7 on 329 degrees of freedom
## Multiple R-squared:  0.7714, Adjusted R-squared:  0.7693
## F-statistic: 370.1 on 3 and 329 DF,  p-value: < 2.2e-16
```

Do flipper length and sex predict body mass?

- sex is categorical, so we'll now want to use anova
- you can see how similar anovas and linear models are!

```
# main effects
m4 <- aov(body_mass_g ~ flipper_length_mm + sex, data = penguins)
summary(m4)
```

```
##                    Df    Sum Sq   Mean Sq F value   Pr(>F)
## flipper_length_mm   1 164047703 164047703 1295.26  < 2e-16 ***
## sex                 1   9416589   9416589   74.35 2.78e-16 ***
## Residuals         330  41795374    126653
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

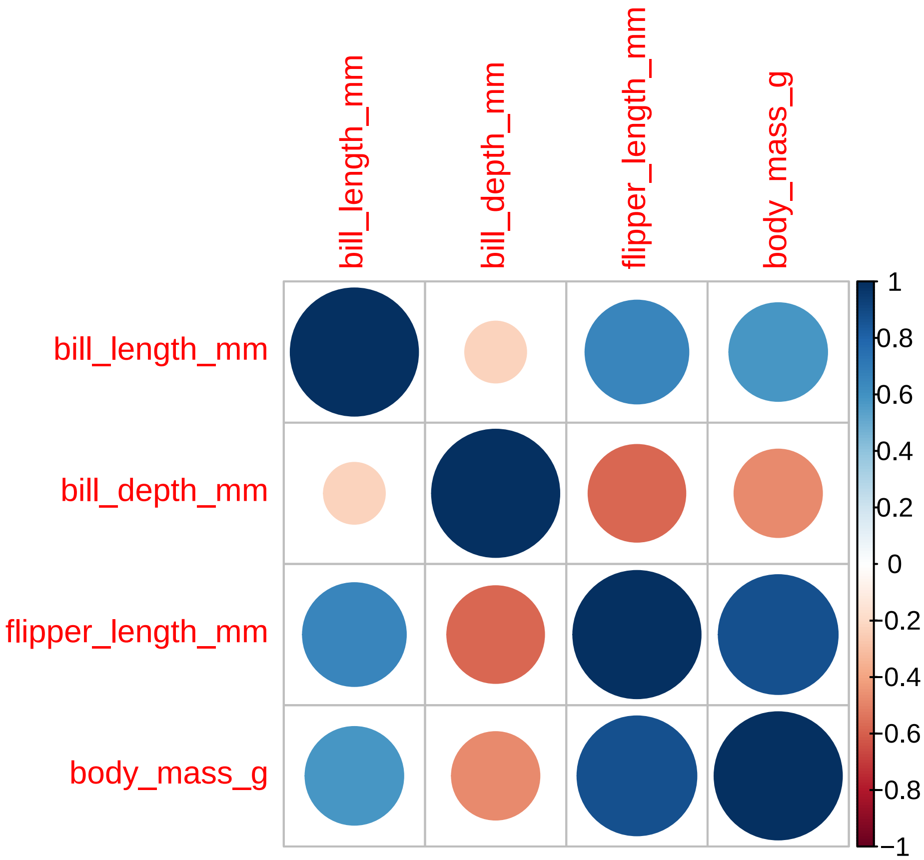
```
# interaction
m5 <- aov(body_mass_g ~ flipper_length_mm * sex, data = penguins)
summary(m5)
```

```
##                        Df    Sum Sq   Mean Sq F value   Pr(>F)
## flipper_length_mm       1 164047703 164047703 1291.37  < 2e-16 ***
## sex                     1   9416589   9416589   74.13 3.08e-16 ***
## flipper_length_mm:sex   1      1286      1286    0.01     0.92
## Residuals             329  41794088    127034
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

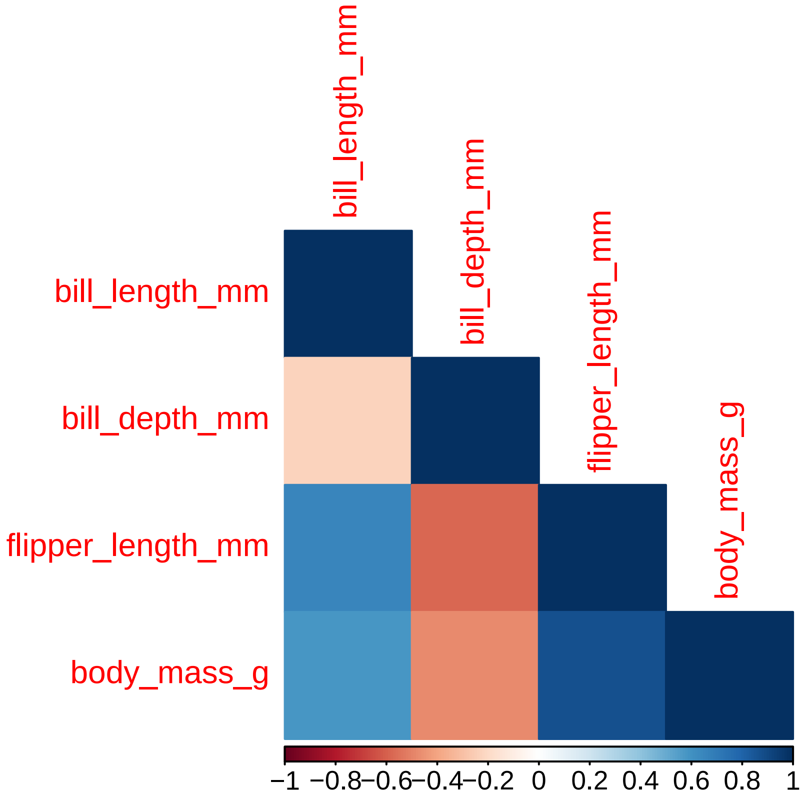
How to know if variables are correlated with each other?

- correlation matrix!
- `corrplot` package
- make sure you're using continuous variables here

```
corr_data <- penguins %>%
  select(bill_length_mm, bill_depth_mm, flipper_length_mm, body_mass_g) %>%
  cor()

corrplot(corr_data)
```




```
corrplot(corr_data, method="color", type="lower")
```



For more complicated Anovas

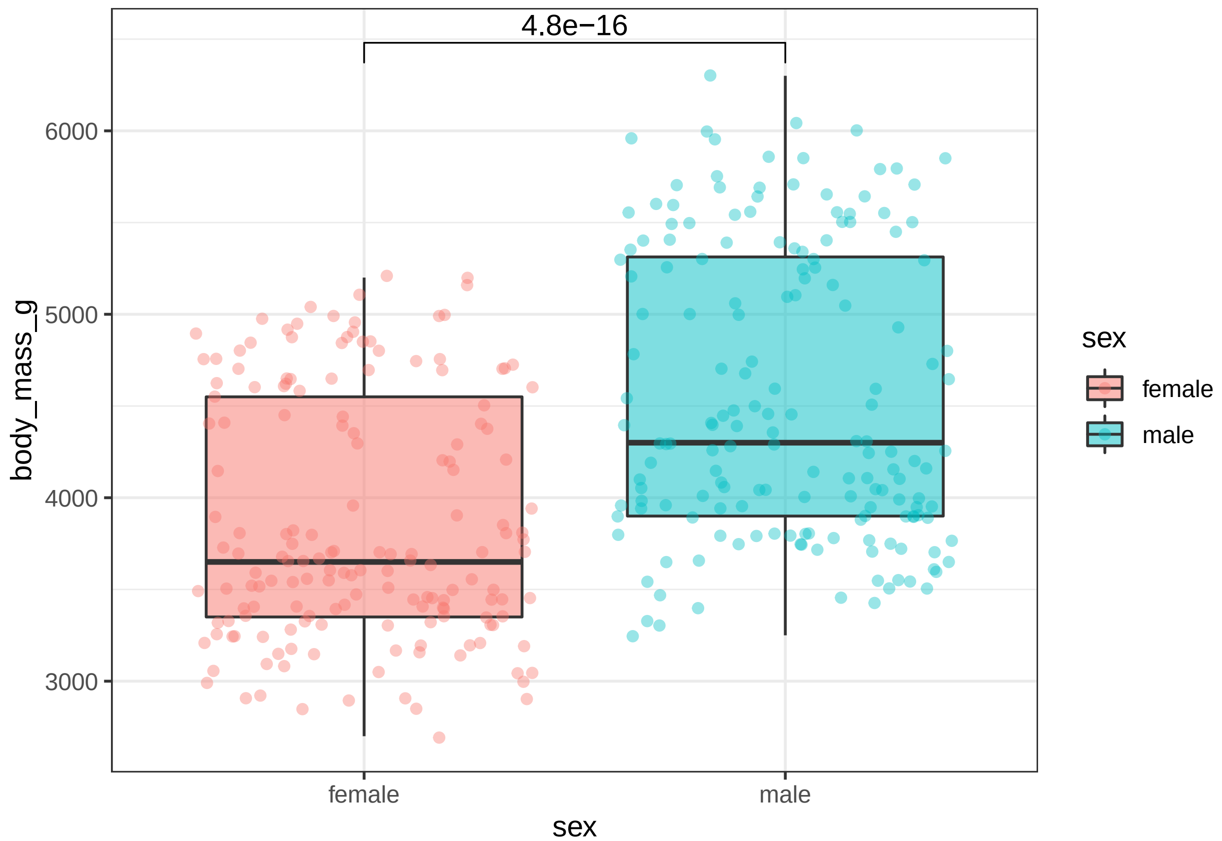
- whenever I need to use anovas, I always go to this webiste: http://www.cookbook-r.com/Statistical_analysis/ANOVA/
- gives examples for different types of anovas and how to properly write them out using aov()
- your model will be slightly different depending on whether you have within subject variables or between subject variables (or a mix of both)
  - one-way within anova
  - Mixed design anova
  - two within variables
  - one between variable and two within variables

ggpubr

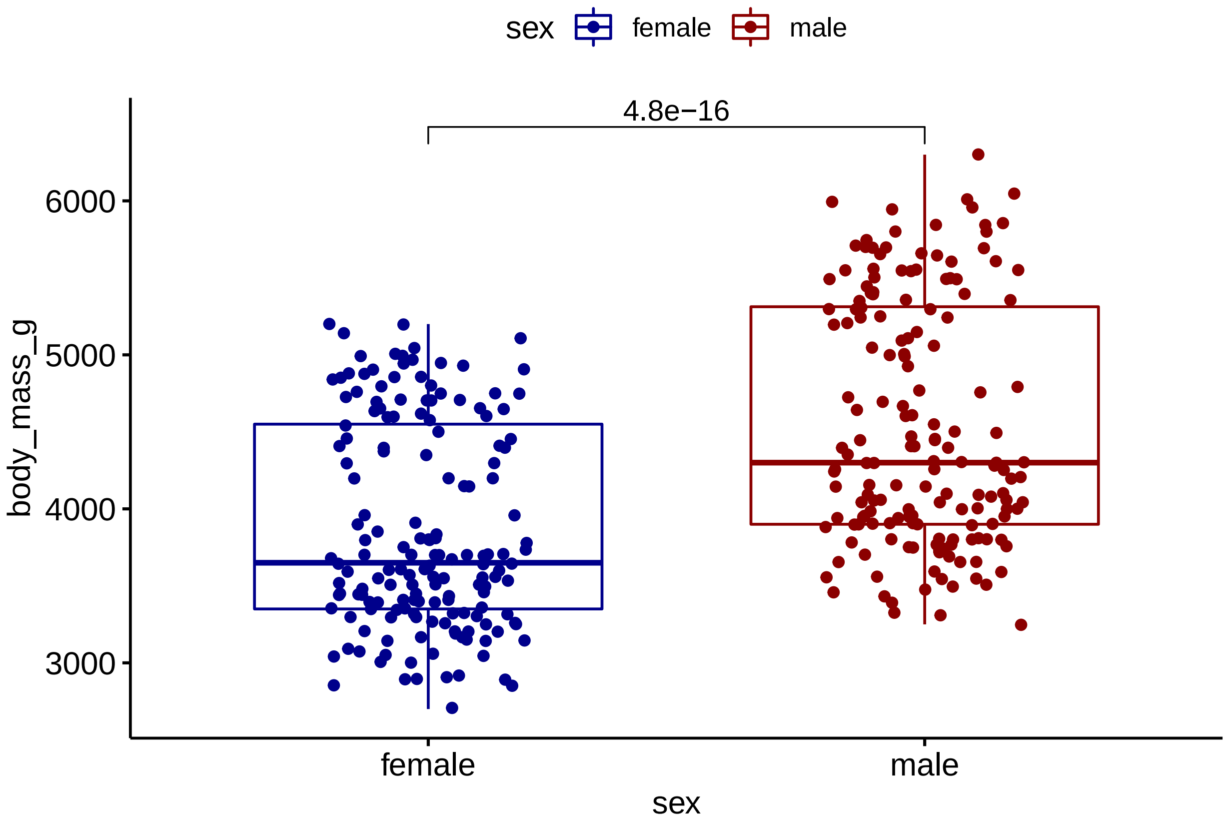
- "publication ready" ggplots!
- has default settings that will make your graphs look beautiful with minimal lines of code
- this is just something extra, if you are feeling ambitious and want to test out some new functions
- here are just a couple of examples, but there are many other ggpubr functions
- with ggpubr functions, note that you have to put variable names in quotes

```
my_comparisons <- list(c("female", "male"))

# regular ggplot
```

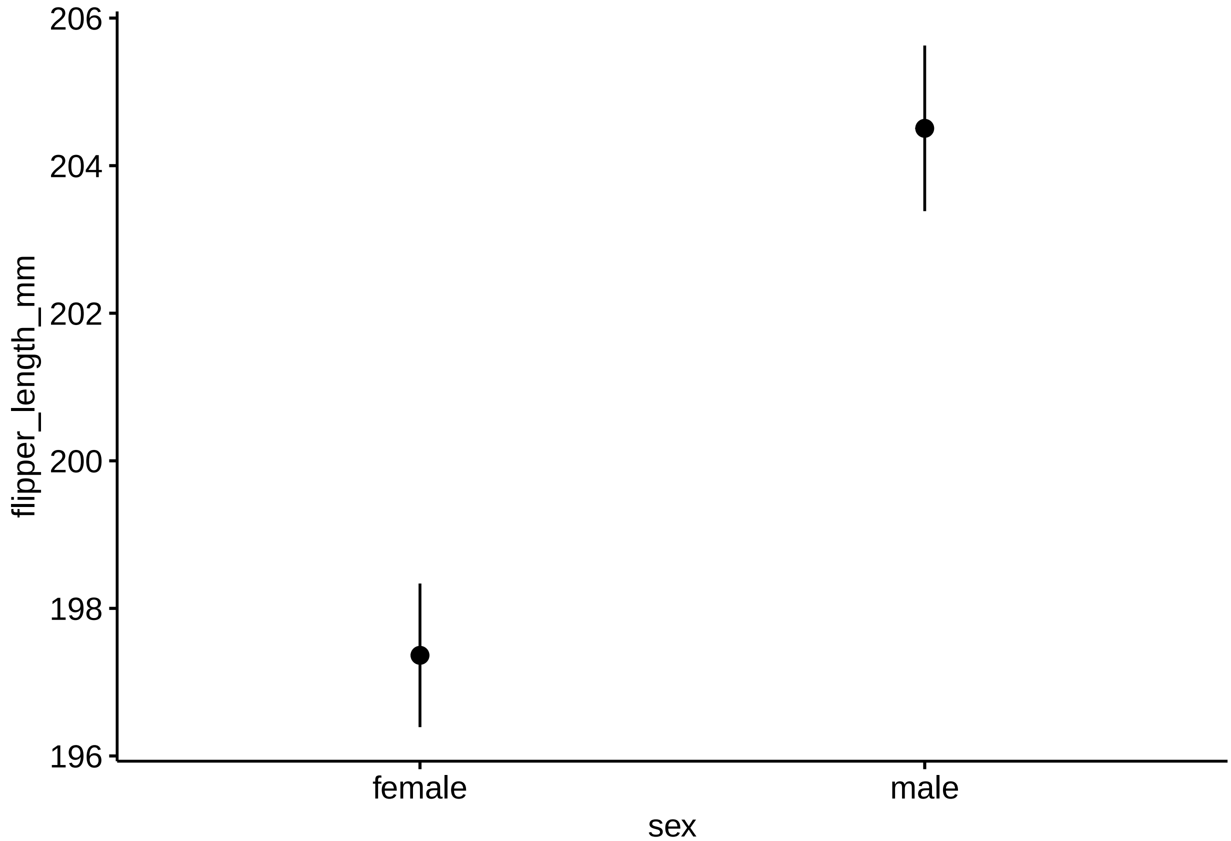
```
penguins %>%
  group_by(sex) %>%
  ggplot(aes(x = sex, y = body_mass_g, fill = sex)) +
  geom_boxplot(alpha = 0.5) +
  geom_jitter(alpha = 0.4, aes(color=sex)) +
  easy_text_size(15) +
  theme_bw() +
  stat_compare_means(comparisons = my_comparisons, method="t.test")
```




```
# ggpubr boxplot function
# of course this is the same data as above, just different aesthetics
ggboxplot(penguins, x = "sex", y = "body_mass_g",
                color = "sex", palette =c("darkblue", "darkred"),
                add = "jitter") +
  stat_compare_means(comparisons = my_comparisons, method="t.test")
```




```
# ggpubr error plot function
ggerrorplot(penguins, x="sex", y="flipper_length_mm", desc_stat="mean_se") # plots the error bars for y
```


206
204
202
200
198
196
flipper_length_mm
female
male
sex