Data Visualisation with R

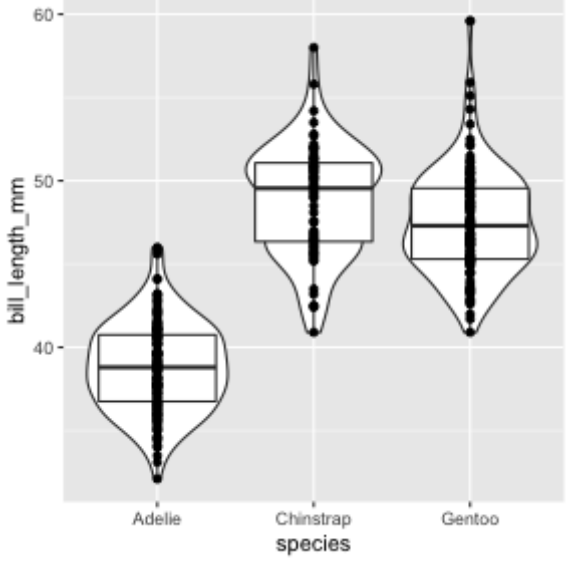
Workshop Part 1

Add multiple layers

```
ggplot(<data>, aes(<mappings>)) +
  <layer>() +
  <layer>() +
  <layer>()
```

Each layer inherits mapping and data from `ggplot` by default.

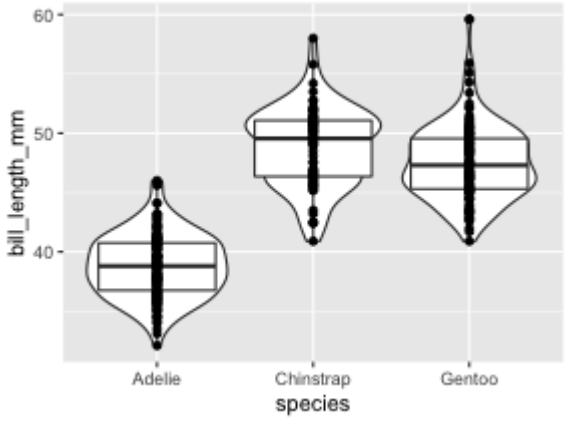
```
ggplot(penguins, aes(x = species, y = bill_length_mm)) +
  geom_violin() +
  geom_boxplot() +
  geom_point()
```



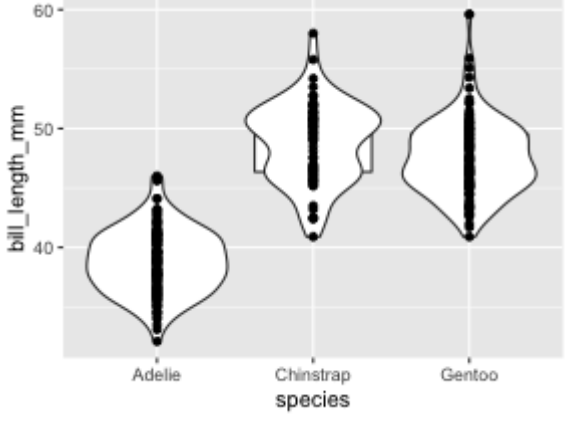
Order of the layers matters!

Boxplot and violin plot order are switched around.

```
ggplot(penguins, aes(species, bill_length_mm))  +
  geom_violin() +
  geom_boxplot() +
  geom_point()
```




```
ggplot(penguins, aes(species, bill_length_mm)) +
  geom_boxplot() +
  geom_violin() +
  geom_point()
```

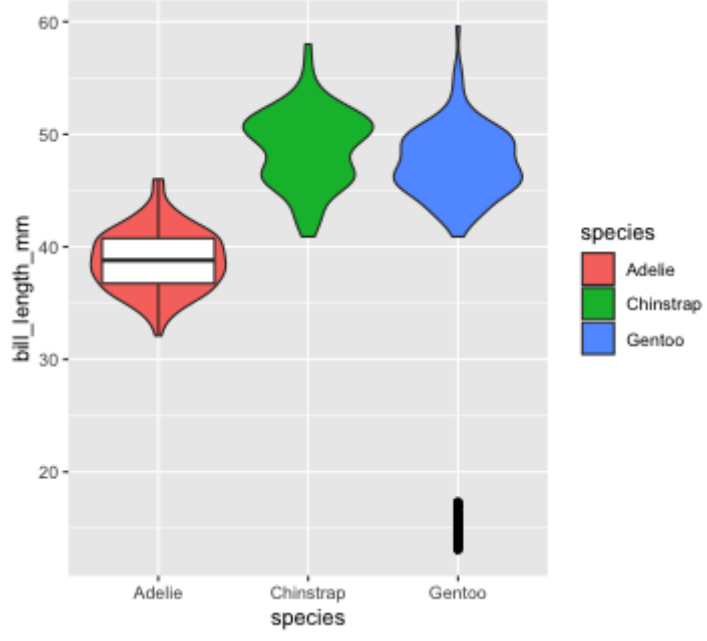


Layer-specific data and aesthetic mappings

```
ggplot(<data>, aes(<mappings>)) +
  <layer>(aes(<mappings>)) +
  <layer>(data = <data>) +
  <layer>(aes(<mappings>), <data>)
```

For each layer, aesthetic and/or data can be overwritten.

```
ggplot(penguins, aes(species, bill_length_mm)) +
  geom_violin(aes(fill = species)) +
  geom_boxplot(data = filter(penguins, species=="Adelie")) +
  geom_point(data = filter(penguins, species=="Gentoo"),
             aes(y = bill_depth_mm))
```



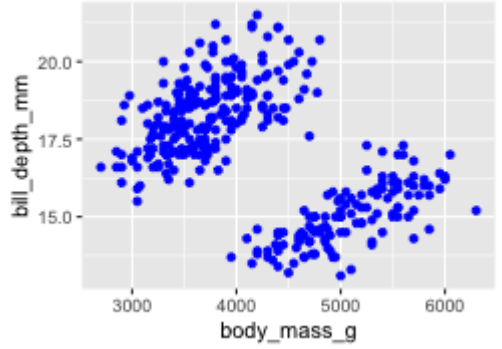
Aesthetic or Attribute?

Not what you want

```
ggplot(penguins) +
  geom_point(aes(body_mass_g,
                 bill_depth_mm,
                 color = "blue"))
```

What you want

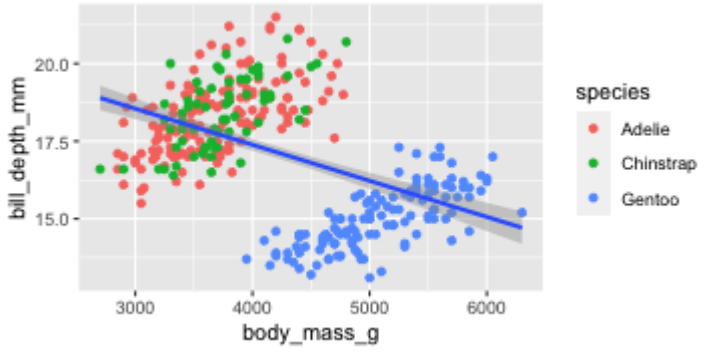
```
ggplot(penguins) +
  geom_point(aes(body_mass_g,
                 bill_depth_mm),
             color = "blue")
```




```
ggplot(penguins) +
  geom_point(aes(body_mass_g,
                 bill_depth_mm,
                 color = I("blue")))
```

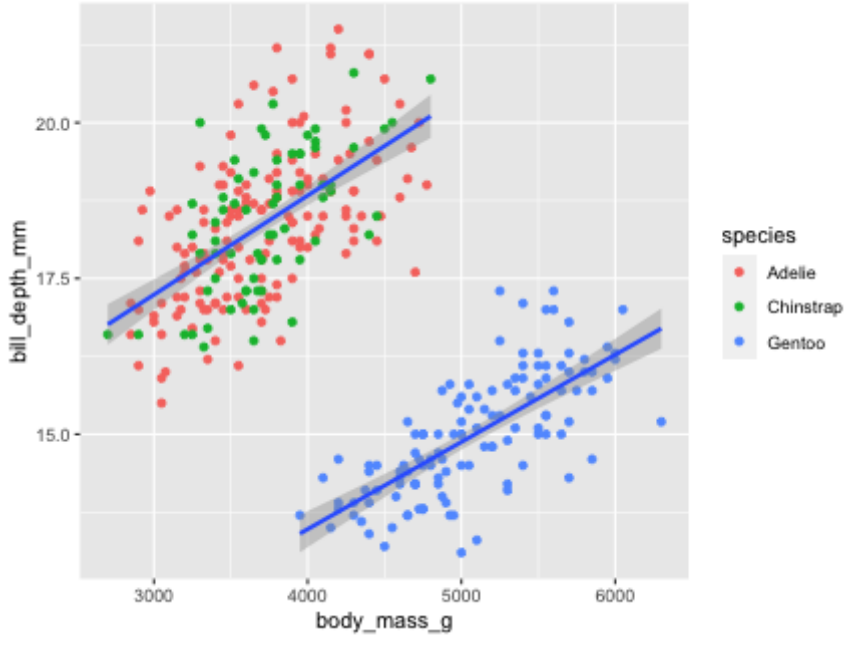
group in ggplot

```
ggplot(penguins,
       aes(body_mass_g,
           bill_depth_mm)) +
  geom_point(aes(color = species)) +
  geom_smooth(method = "lm")
```



- This is an obvious case of Simpson's paradox.
- What if we wanted to draw the fit of a simple linear model for each

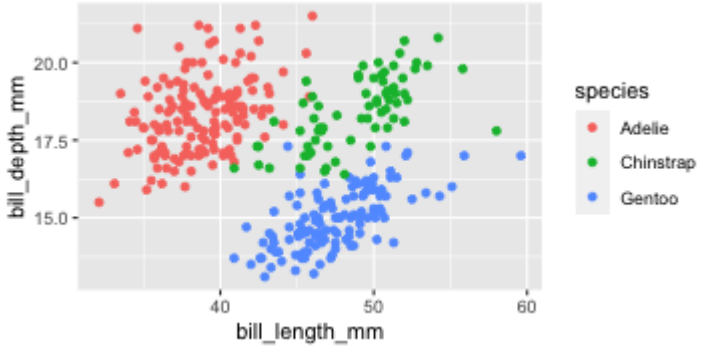
```
ggplot(penguins,
       aes(body_mass_g,
           bill_depth_mm)) +
  geom_point(aes(color = species)) +
  geom_smooth(method = "lm",
              aes(group = species=="Gentoo"))
```



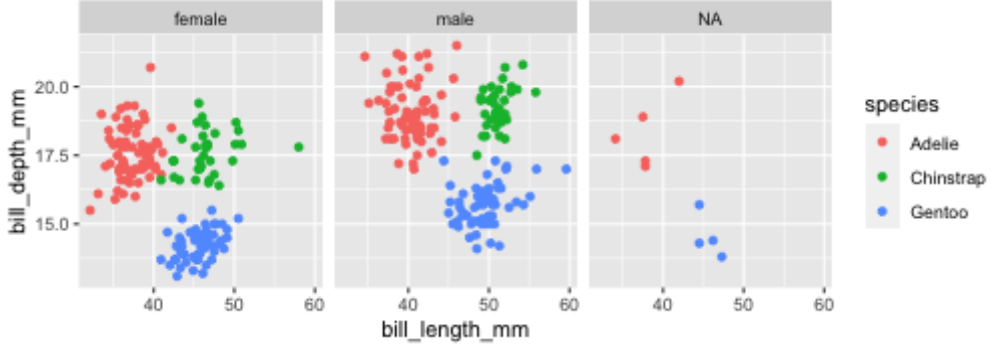
Facetting

```
g <- ggplot(penguins, aes(bill_length_mm, bill_depth_mm, color = species)) + geom_point()
```

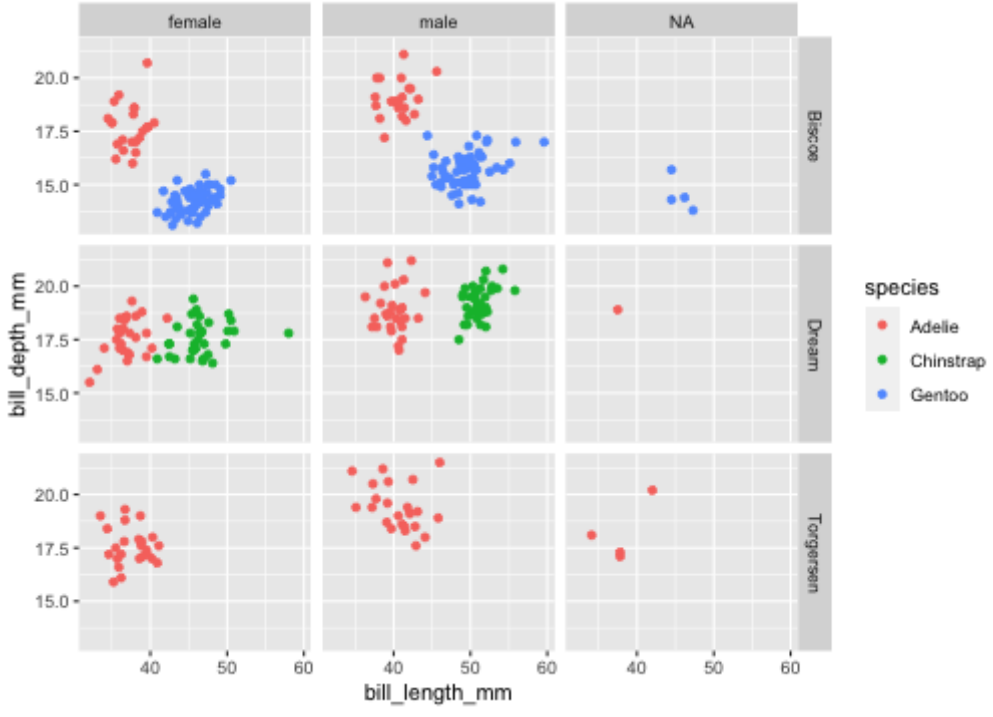
```
g
```




```
g + facet_wrap(~sex)
```

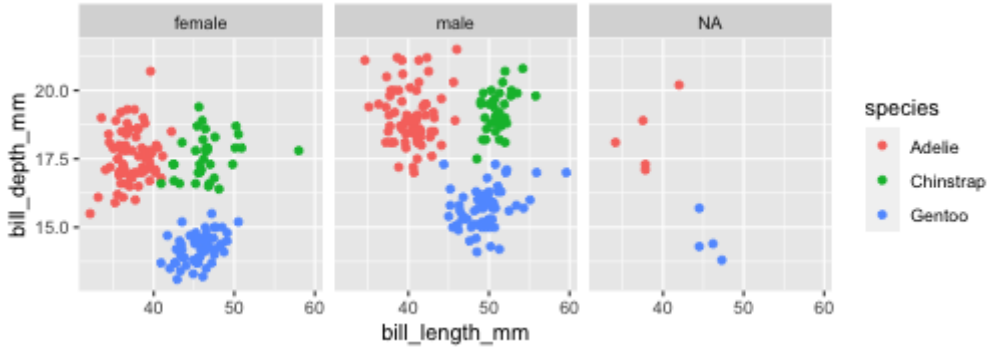



```
g + facet_grid(island ~ sex)
```

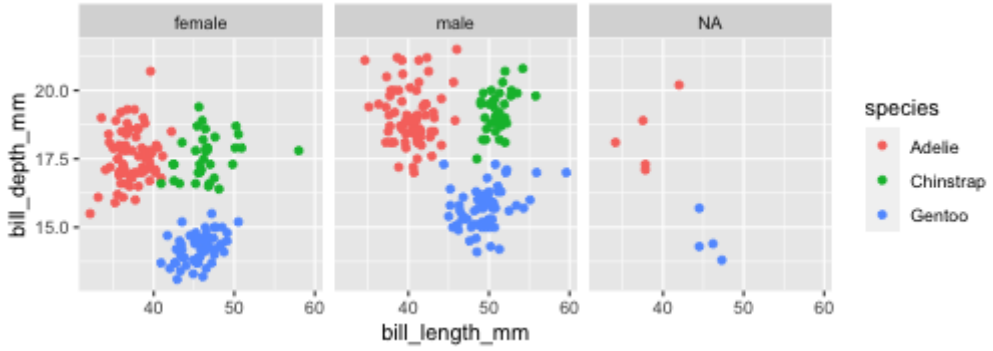


facet_wrap and facet_grid

```
g + facet_wrap( ~ sex)
```

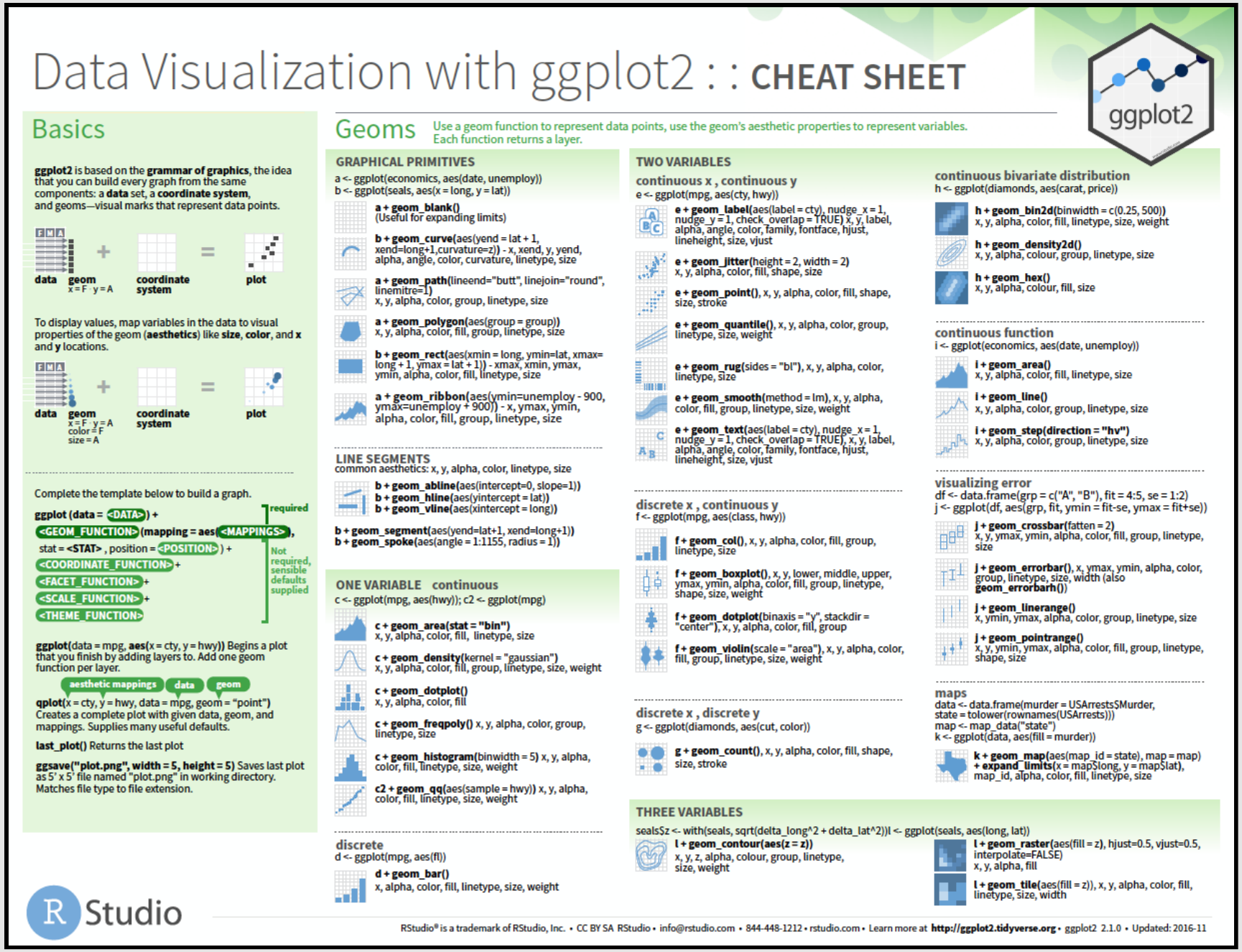



```
g + facet_grid(. ~ sex)
```




```
g + facet_wrap( ~ sex, ncol = 1)
```

```
g + facet_grid(sex ~ .)
```



HELP!

- RStudio > Help > Cheatsheets
- R4DS Community Slack
- Twitter with hastag #rstats
- RStudio Community
- Stackoverflow

2

Tidying your data

Wickham (2014) Tidy Data. *Journal of Statistical Software* 59 (10): 1–23.

Weight gain in pigs for different treatments

The `crampton.pig` is from the `agridat` 📦

```
library(agridat)
glimpse(crampton.pig)

## Rows: 50
## Columns: 5
## $ treatment <fct> T1, T1, T1, T1, T1, T1, T1, T1, T1, T1, T2, T2, T2, T2, T2,
## $ rep       <fct> R1, R2, R3, R4, R5, R6, R7, R8, R9, R10, R1, R2, R3, R4, R5
## $ weight1   <int> 30, 21, 21, 33, 27, 24, 20, 29, 28, 26, 26, 24, 20, 35, 25,
## $ feed      <int> 674, 628, 661, 694, 713, 585, 575, 638, 632, 637, 699, 626,
## $ weight2   <int> 195, 177, 180, 200, 197, 170, 150, 180, 192, 184, 194, 204,
```

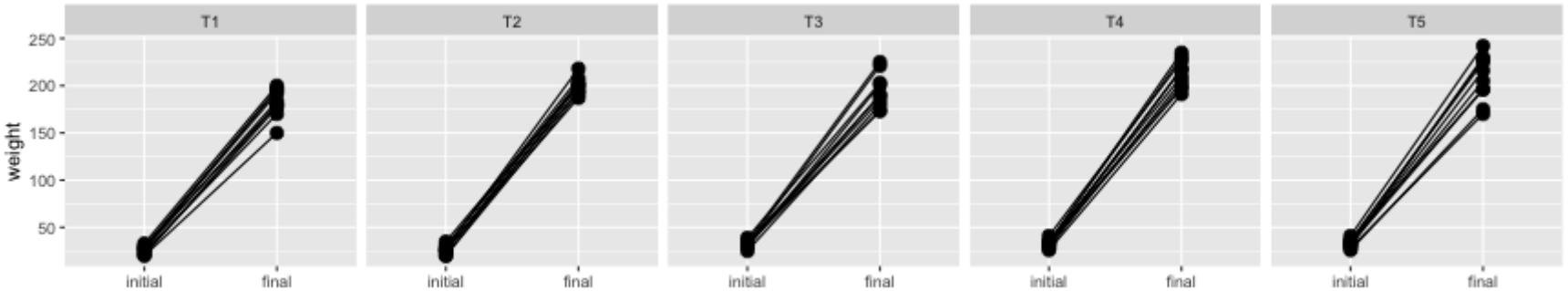
`weight1` is initial weight and `weight2` is final weight

Wright (2018). agridat: Agricultural Datasets. R package version 1.16. https://CRAN.R-project.org/package=agridat

Crampton and Hopkins (1934). The Use of the Method of Partial Regression in the Analysis of Comparative Feeding Trial Data, Part II. *The Journal of Nutrition* 8 113-123.

```
names(crampton.pig)

## [1] "treatment" "rep"       "weight1"   "feed"      "weight2"
```

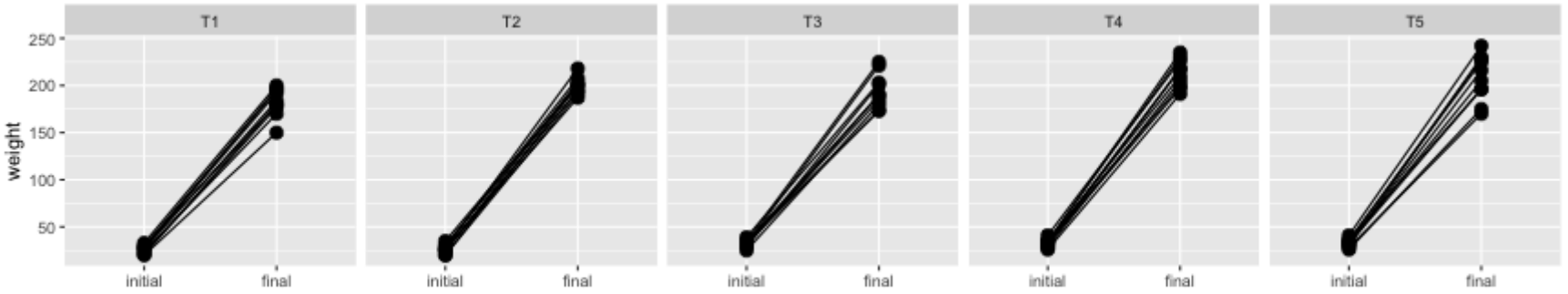


What are the mappings to get the above graph? 🤔

```
ggplot(crampton.pig, aes(x = ???, y = ???)) +
  geom_point() +
  geom_line() +
  facet_grid(. ~ treatment)
```

😐

Getting the data in the right form



- The x-axis is the time when pig was weighed
- The y-axis is the weight
- The facetting is by treatment

```
## # A tibble: 100 × 5
##    treatment  feed id    when    weight
##    <fct>     <int> <chr> <fct>    <int>
##  1 T1          674 pig1  initial     30
##  2 T1          674 pig1  final      195
##  3 T1          628 pig2  initial     21
```

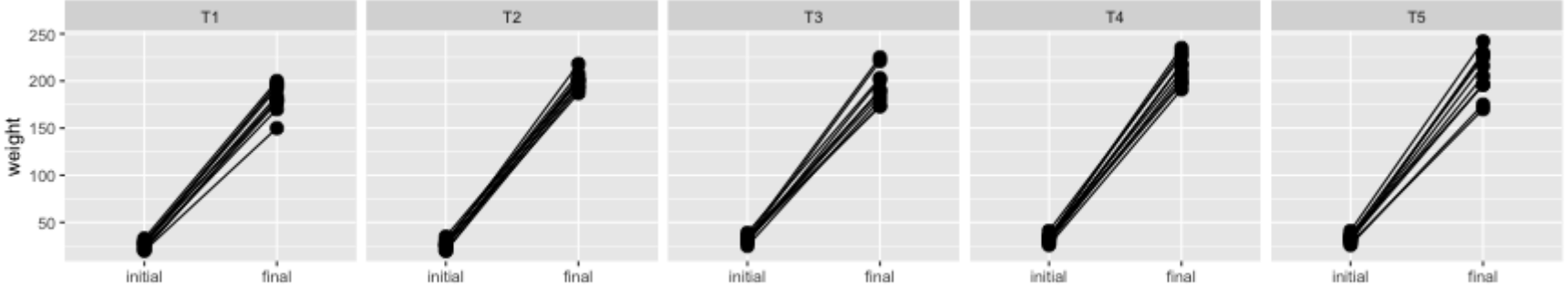
How I wrangled this data

```
pig_df <- crampton.pig %>%
 mutate(id = paste0("pig", 1:n())) %>%
 pivot_longer(c(weight1, weight2),
              names_to = "when",
              values_to = "weight") %>%
 mutate(when = factor(when,
         levels = c("weight1", "weight2"),
         labels = c("initial", "final")))
```

(note: teaching wrangling is not part of this workshop, please see here if you want to learn more)

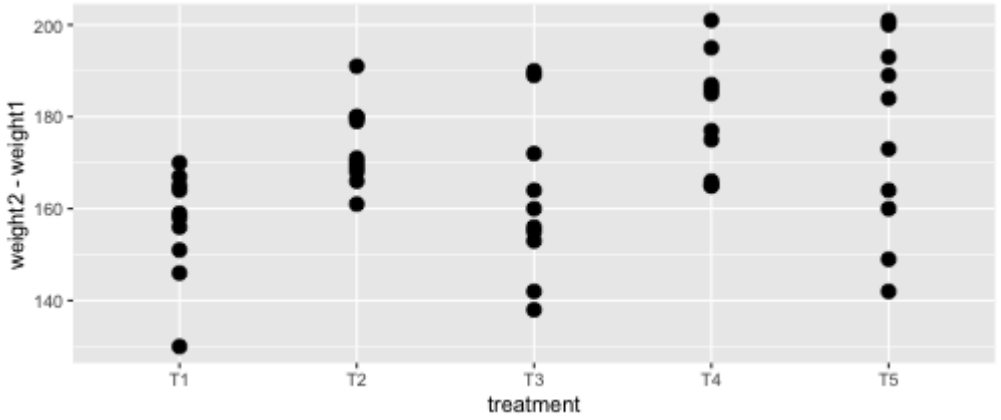
Putting it all together

```
ggplot(pig_df, aes(when, weight)) + # tidying your data for plotting
  geom_point(size = 3) + # attribute not aesthestic
  geom_line(aes(group = id)) + # grouping
  facet_grid(. ~ treatment) + # facetting
  labs(x = "") # we'll learn this in the last session
```



Meaningfully order categorical variables

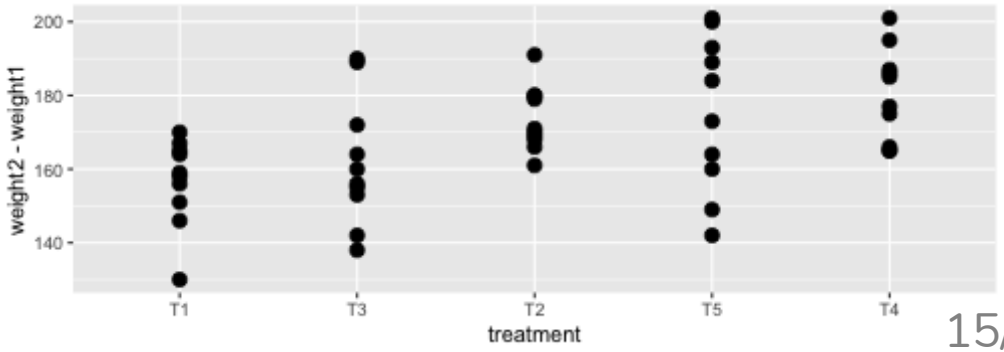
```
ggplot(crampton.pig,
  aes(treatment, weight2 - weight1)) +
  geom_point(size = 3)
```



- Treatments are ordered alphabetically by default
- It's better to order categorical variables meaningfully

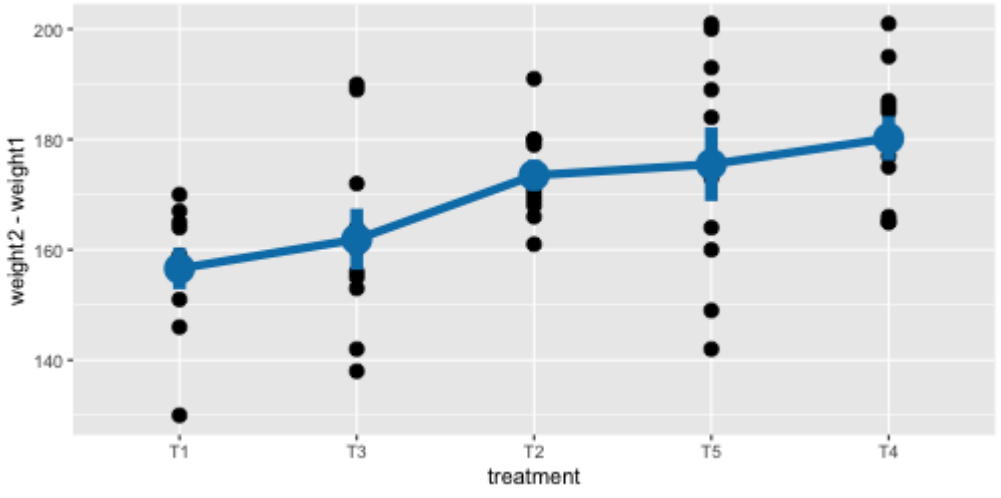
Order factor levels by the mean of the weight difference.

```
library(forcats) #  for easy factor manipulatio
crampton.pig2 <- crampton.pig %>%
 mutate(
  treatment = fct_reorder(treatment,
                          weight2 - weight1,
                          mean))
ggplot(crampton.pig2,
       aes(treatment, weight2 - weight1)) +
  geom_point(size = 3)
```



Plotting auxilliary data

Plot you may want:

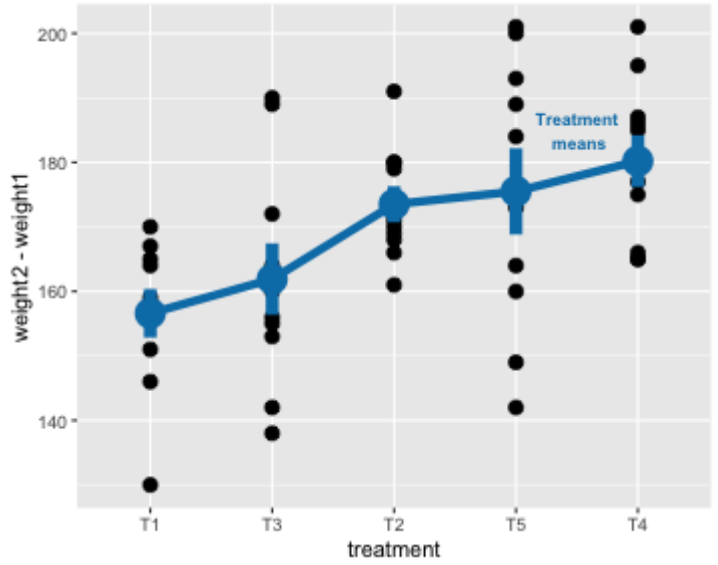


One way to do this:

```
fig <- ggplot(crampton.pig2,
              aes(treatment,
                  weight2 - weight1)) +
  geom_point(size = 3) +
  stat_summary(fun.data = mean_se,
               geom = "pointrange", fatten = 2,
               color = "#027EB6", size = 3) +
  stat_summary(fun = mean, geom = "line",
               group = 1, color = "#027EB6",
               size = 2)
```

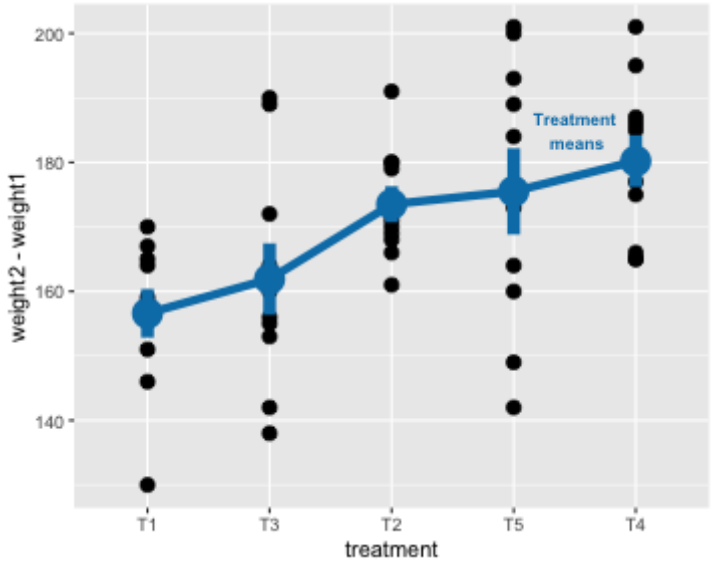
Plotting annotations

```
fig +
  geom_text(data = data.frame(treatment = 4.5,
                              weight2 = 185,
                              weight1 = 0),
            label = "Treatment\n means",
            size = 3,
            color = "#027EB6",
            fontface = "bold")
```



But it might be just easier to:

```
fig +
  annotate("text",
           x = 4.5, y = 185,
           label = "Treatment\n means",
           size = 3,
           color = "#027EB6",
           fontface = "bold")
```



</> Open part1-exercise-02.Rmd

15:00

Session Information

```
devtools::session_info()
```

```
## ─ Session info  🎅  👊  🎳  ──────────────────────────────────────────────
##  hash: Santa Claus: dark skin tone, oncoming fist: medium skin tone, bowling
##
##  setting  value
##  version  R version 4.1.2 (2021-11-01)
##  os       macOS Big Sur 10.16
##  system   x86_64, darwin17.0
##  ui       X11
##  language (EN)
##  collate  en_AU.UTF-8
##  ctype    en_AU.UTF-8
##  tz       Australia/Melbourne
##  date     2022-02-20
```