SM-2302 Software for Mathematicians

R4: Plotting with `ggplot2` *[handout version]*

Dr. Haziq Jamil
Mathematical Sciences, Faculty of Science, UBD
`https://haziqj.ml`

Semester I 2022/23

Overview

The Grammar of Graphics

Breaking down the ggplot call

Aesthetics

Faceting

A brief plot Tour of ggplot2 plots

Themes

Other useful things

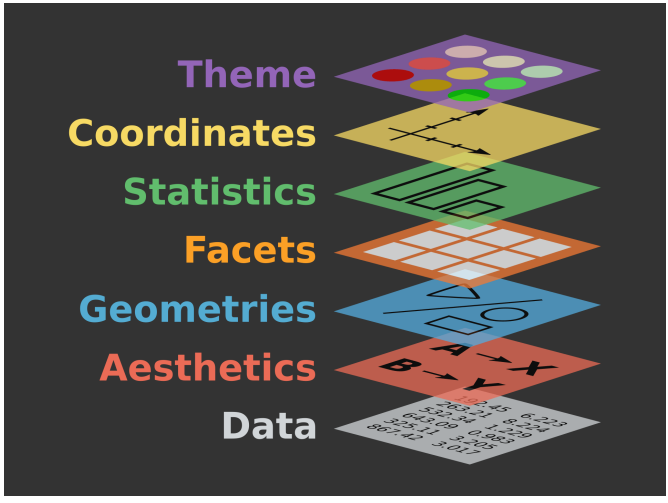
The Grammar of Graphics

- Visualization concept created by Leland Wilkinson (The Grammar of Graphics, 1999)–an attempt to taxonimize the basic elements of statistical graphics
- Adapted for R by Hadley Wickham (2009)
  - consistent and compact syntax to describe statistical graphics
  - highly modular as it breaks up graphs into semantic components
- ggplot2 is not meant as a guide to which graph to use and how to best convey your data (more on that later), but it does have some strong opinions.

Terminology

A statistical graphic is a...

- mapping of **data**
- which may be **statistically transformed** (summarized, log-transformed, etc.)
- to **aesthetic attributes** (color, size, xy-position, etc.)
- using **geometric objects** (points, lines, bars, etc.)
- and mapped onto a specific **facet** and **coordinate system**



http://r.qcbs.ca/workshop03/book-en/grammar-of-graphics-gg-basics.html

Anatomy of a ggplot call

```
ggplot(
     data = [dataframe],
  mapping = aes(x = [var x], y = [var y], color = [var color],
                shape = [var shape], ...)
) +
  geom_[some geom](
    mapping = aes(fill = [var geom color], ...),
    # other geometry options
  ) +
  ... # other geometries
  scale_[some axis]_[some scale]() +
  facet_[some facet]([formula]) +
  ... # other plot options
      # usually labels, titles & themes
```

Palmer Penguins

Measurements for penguin species, island in Palmer Archipelago, size (flipper length, body mass, bill dimensions), and sex.

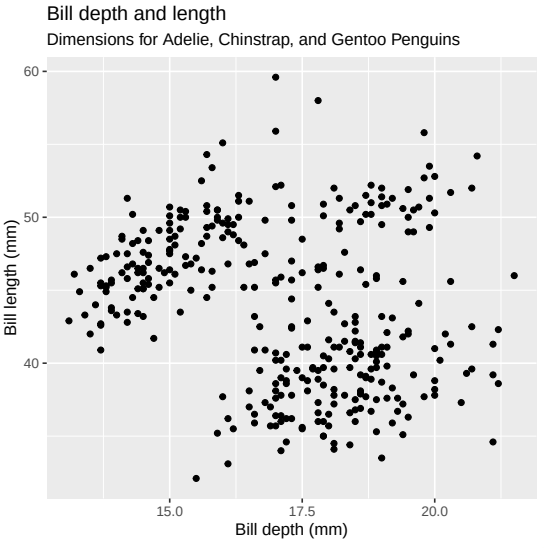

```
library(palmerpenguins)
penguins %>% print(n = 5)
```

```
## # A tibble: 344 x 8
##   species island  bill_~1 bill_~2 flipp~3 body_~4 sex
##   <fct>   <fct>     <dbl>   <dbl>   <int>   <int> <fct>
## 1 Adelie  Torger~    39.1    18.7     181    3750 male
## 2 Adelie  Torger~    39.5    17.4     186    3800 fema~
## 3 Adelie  Torger~    40.3    18       195    3250 fema~
## 4 Adelie  Torger~    NA      NA        NA      NA <NA>
## 5 Adelie  Torger~    36.7    19.3     193    3450 fema~
## # ... with 339 more rows, 1 more variable: year <int>,
## #   and abbreviated variable names 1: bill_length_mm,
## #   2: bill_depth_mm, 3: flipper_length_mm,
## #   4: body_mass_g
```

A basic ggplot

```
ggplot(data    = penguins,
       mapping = aes(x = bill_depth_mm,
                     y = bill_length_mm)) +
  geom_point() +
  labs(
    title = "Bill depth and length",
    subtitle = paste(
      "Dimensions for Adelie, Chinstrap,",
      "and Gentoo Penguins"
    ),
    x = "Bill depth (mm)",
    y = "Bill length (mm)"
  )
```

```
## Warning: Removed 2 rows containing missing va
## (geom_point).
```

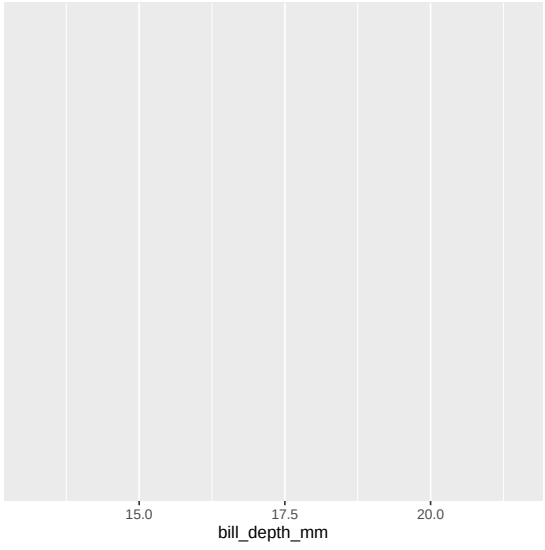


Start with the `penguins` data frame

```
ggplot(data = penguins)
```

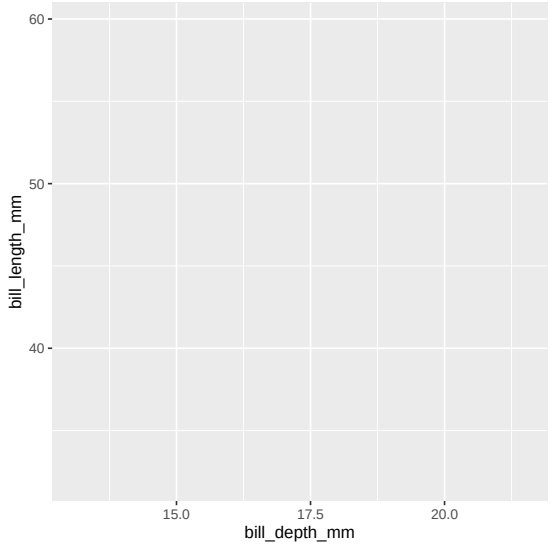
Start with the penguins data frame, **map bill depth to the x-axis**

```
ggplot(
  data = penguins,
  mapping = aes(x = bill_depth_mm)
)
```



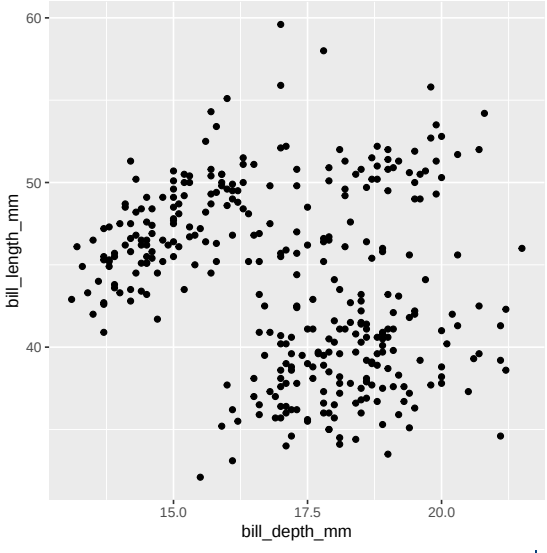
Start with the `penguins` data frame, map bill depth to the x-axis, **and map bill length to the y-axis.**

```
ggplot(
  data = penguins,
  mapping = aes(x = bill_depth_mm,
                y = bill_length_mm)
)
```



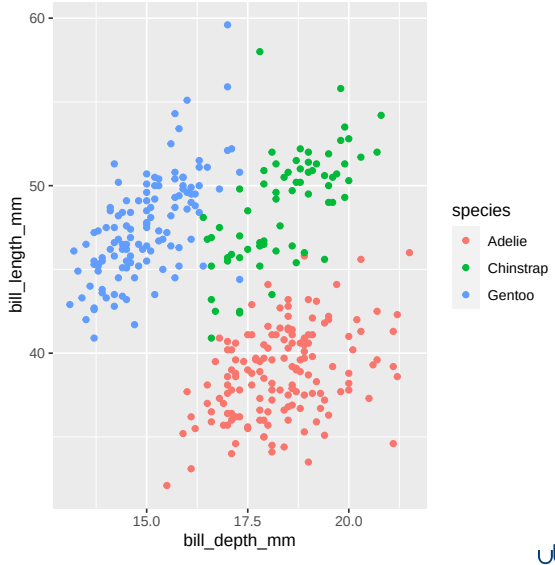
Start with the `penguins` data frame, map bill depth to the x-axis, and map bill length to the y-axis. **Represent each observation with a point**

```
ggplot(
  data = penguins,
  mapping = aes(x = bill_depth_mm,
                y = bill_length_mm)
) +
  geom_point()
```



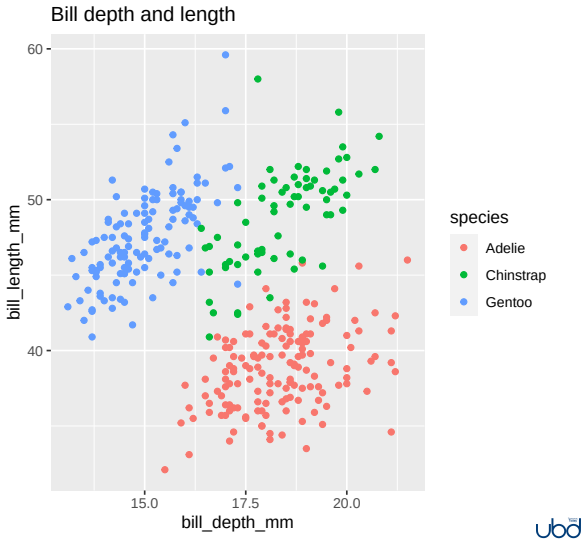
Start with the `penguins` data frame, map bill depth to the x-axis, and map bill length to the y-axis. Represent each observation with a point **and map species to the color of each point.**

```
ggplot(
  data = penguins,
  mapping = aes(x = bill_depth_mm,
                y = bill_length_mm)
) +
  geom_point(
    mapping = aes(colour = species)
  )
```



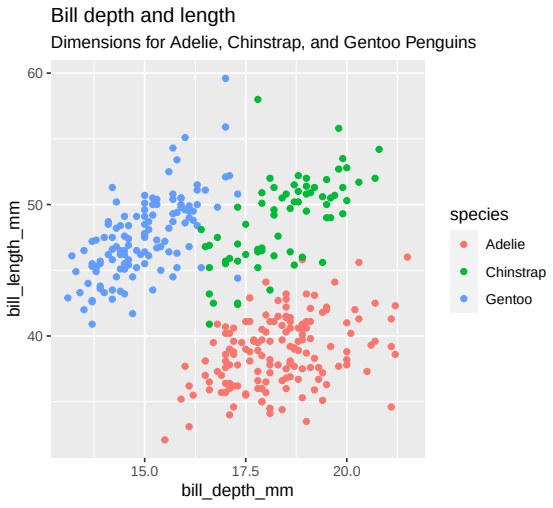
Start with the `penguins` data frame, map bill depth to the x-axis, and map bill length to the y-axis. Represent each observation with a point and map species to the color of each point. **Title the plot "Bill depth and length"**

```
ggplot(
  data = penguins,
  mapping = aes(x = bill_depth_mm,
                y = bill_length_mm)
) +
  geom_point(
    mapping = aes(colour = species)
  ) +
  labs(title = "Bill depth and length")
```



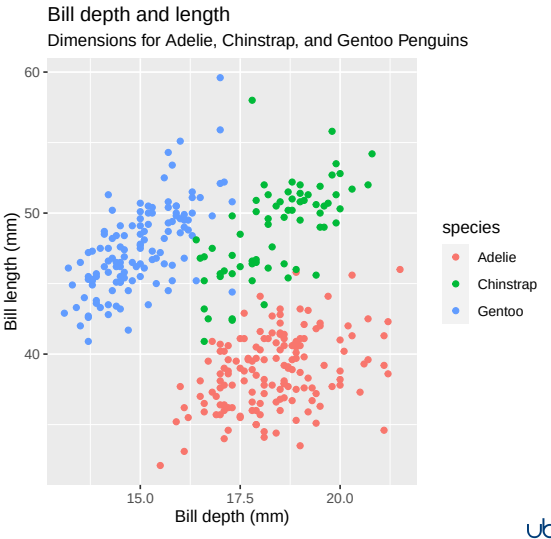
Start with the `penguins` data frame, map bill depth to the x-axis, and map bill length to the y-axis. Represent each observation with a point and map species to the color of each point. Title the plot "Bill depth and length" **and add the subtitle "Dimensions for Adelie, Chinstrap, and Gentoo Penguins"**

```
ggplot(
  data = penguins,
  mapping = aes(x = bill_depth_mm,
                y = bill_length_mm)
) +
  geom_point(
    mapping = aes(colour = species)
  ) +
  labs(
    title = "Bill depth and length",
    subtitle = paste("Dimensions for Adelie,",
                     "Chinstrap, and Gentoo",
                     "Penguins")
  )
```



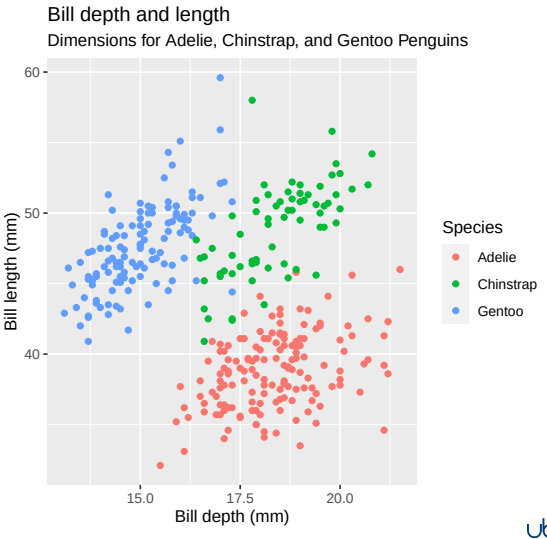
Start with the `penguins` data frame, map bill depth to the x-axis, and map bill length to the y-axis. Represent each observation with a point and map species to the color of each point. Title the plot "Bill depth and length" **label the x and y axes as "Bill depth (mm)" and "Bill length (mm)", respectively**

```
ggplot(
  data = penguins,
  mapping = aes(x = bill_depth_mm,
                y = bill_length_mm)
) +
  geom_point(
    mapping = aes(colour = species)
  ) +
  labs(
    title = "Bill depth and length",
    subtitle = paste("Dimensions for Adelie,",
                     "Chinstrap, and Gentoo",
                     "Penguins"),
    x = "Bill depth (mm)",
    y = "Bill length (mm)"
  )
```



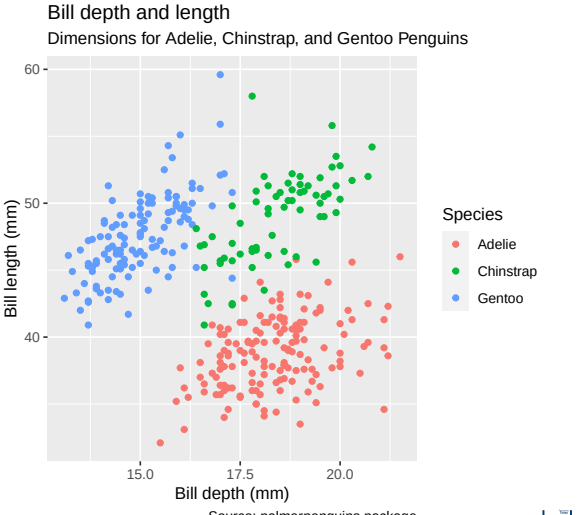
Start with the `penguins` data frame, map bill depth to the x-axis, and map bill length to the y-axis. Represent each observation with a point and map species to the color of each point. Title the plot "Bill depth and length" label the x and y axes as "Bill depth (mm)" and "Bill length (mm)", respectively. **Label the legend "Species"**

```
ggplot(
  data = penguins,
  mapping = aes(x = bill_depth_mm,
                y = bill_length_mm)
) +
  geom_point(
    mapping = aes(colour = species)
  ) +
  labs(
    title = "Bill depth and length",
    subtitle = paste("Dimensions for Adelie,",
                     "Chinstrap, and Gentoo",
                     "Penguins"),
    x = "Bill depth (mm)",
    y = "Bill length (mm)",
    colour = "Species"
  )
```



Start with the `penguins` data frame, map bill depth to the x-axis, and map bill length to the y-axis. Represent each observation with a point and map species to the color of each point. Title the plot "Bill depth and length" label the x and y axes as "Bill depth (mm)" and "Bill length (mm)", respectively. Label the legend "Species" **and add caption.**

```
ggplot(
  data = penguins,
  mapping = aes(x = bill_depth_mm,
                y = bill_length_mm)
) +
  geom_point(
    mapping = aes(colour = species)
  ) +
  labs(
    title = "Bill depth and length",
    subtitle = paste("Dimensions for Adelie,",
                     "Chinstrap, and Gentoo",
                     "Penguins"),
    x = "Bill depth (mm)", y = "Bill length (mm
    colour = "Species",
    caption = "Source: palmerpenguins package"
  )
```



Argument names

Often we omit the names of first two arguments when building plots with `ggplot()`.

```
ggplot(
  data = penguins,
  mapping = aes(x = bill_depth_mm,
                y = bill_length_mm)
) +
  geom_point(
    mapping = aes(color = species)
  )
```

```
ggplot(penguins, aes(x = bill_depth_mm,
                     y = bill_length_mm)) +
  geom_point(aes(col = species))
```

Note that `ggplot` and `geom_*` swap the order of the `data` and `mapping` arguments.

Aesthetics options

Commonly used characteristics of plotting geometries that can be **mapped to a specific variable** in the data, examples include:

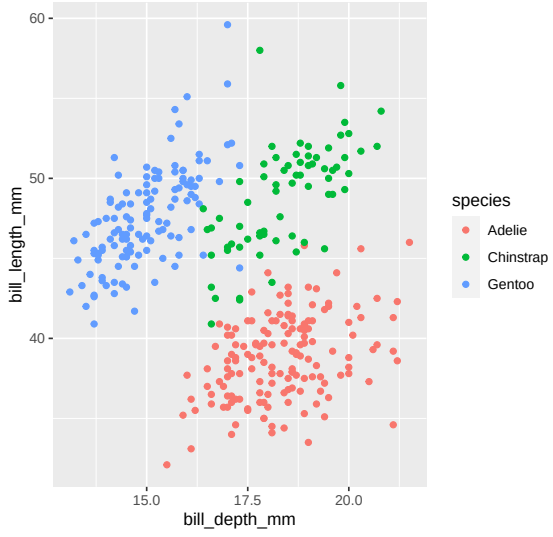
- position (`x`, `y`)
- `color`
- `shape`
- `size`
- `alpha` (transparency)

Different geometries have different aesthetics that can be used - see the ggplot2 geoms help files for listings.

- Aesthetics given in `ggplot()` apply to all `geoms`.
- Aesthetics for a specific `geom_*()` can be overridden via the `mapping` argument.

Colour

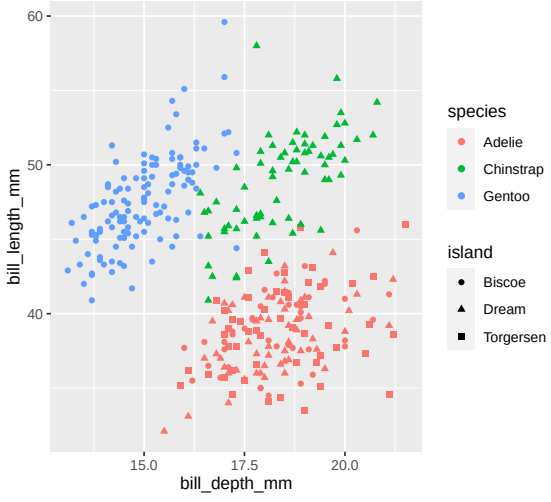
```
ggplot(penguins, aes(x = bill_depth_mm,
                     y = bill_length_mm)) +
  geom_point(aes(col = species))
```



Shape

Mapped to a different variable than `colour`

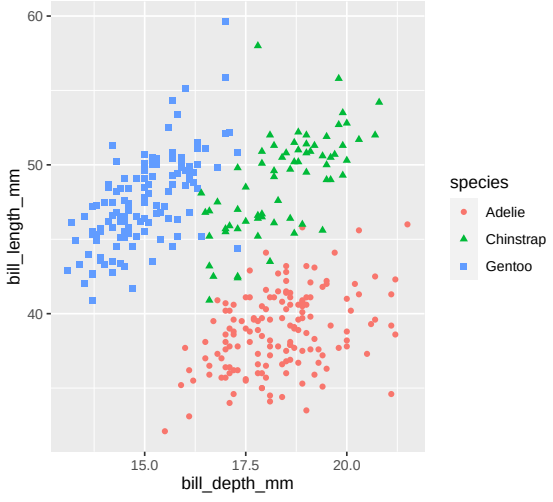
```
ggplot(penguins, aes(x = bill_depth_mm,
                     y = bill_length_mm)) +
  geom_point(aes(col = species,
                 shape = island))
```



Shape (cont.)

Mapped to same variable as `colour`

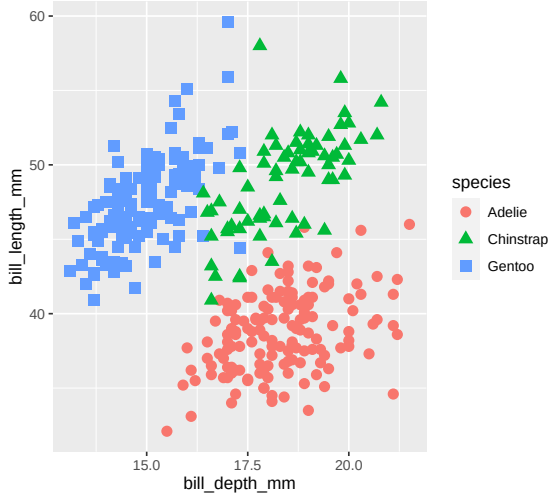
```
ggplot(penguins, aes(x = bill_depth_mm,
                     y = bill_length_mm)) +
  geom_point(aes(col = species,
                 shape = species))
```



Size

Control the size of the points. Note that this is a fixed value (outside of the aes call).

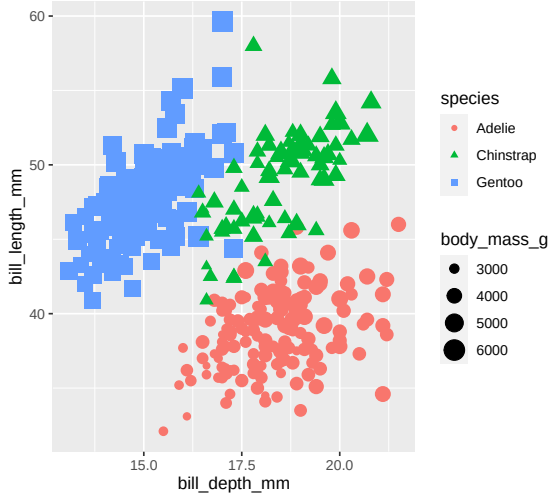
```
ggplot(penguins, aes(x = bill_depth_mm,
                     y = bill_length_mm)) +
  geom_point(aes(col = species,
                 shape = species),
             size = 3)
```



Size (cont.)

Mapping the size aesthetic to a variable.

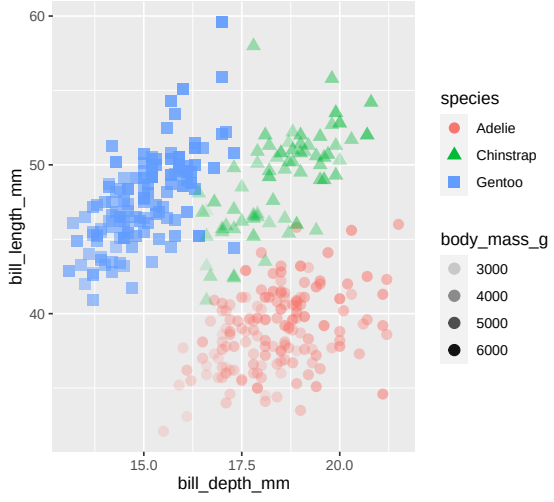
```
ggplot(penguins, aes(x = bill_depth_mm,
                     y = bill_length_mm)) +
  geom_point(aes(col = species,
                 shape = species,
                 size = body_mass_g))
```



Alpha

Mapping the transparency aesthetic to a variable.

```
ggplot(penguins, aes(x = bill_depth_mm,
                     y = bill_length_mm)) +
  geom_point(aes(col = species,
                 shape = species,
                 alpha = body_mass_g),
             size = 3)
```



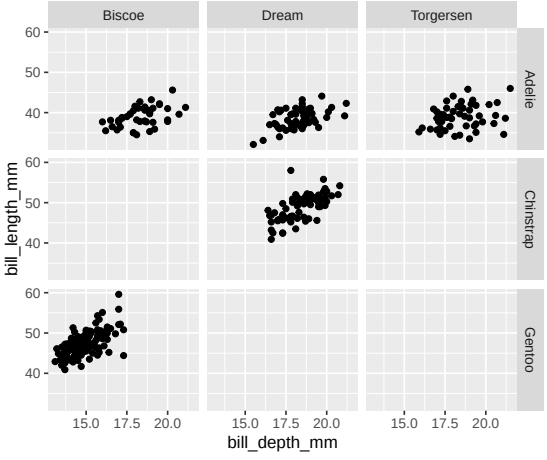
Mapping vs settings

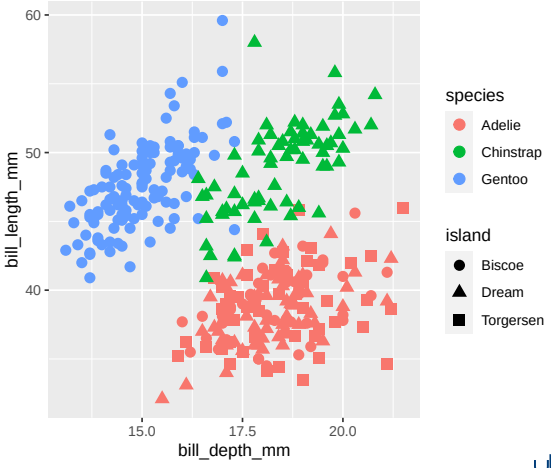
- **Mapping:** Determine an aesthetic (the size, alpha, etc.) of a geom based on the values of a variable in the data
  - wrapped by `aes()` and pass as `mapping` argument to `ggplot()` or `geom_*()`.
- **Setting:** Determine an aesthetic (the size, alpha, etc.) of a geom **not** based on the values of a variable in the data, usually a constant value.
  - passed directly into `geom_*()` as an argument.

From the previous slide `color`, `shape`, and `alpha` are all **aesthetics** while `size` is a **setting**.

Faceting

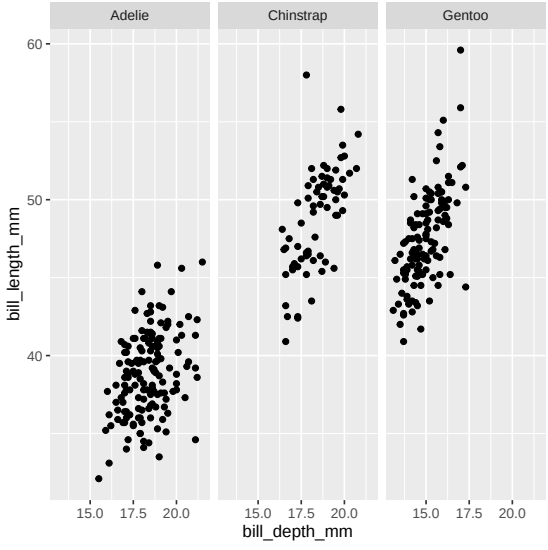
- Smaller plots that display different subsets of the data
- Useful for exploring conditional relationships and large data





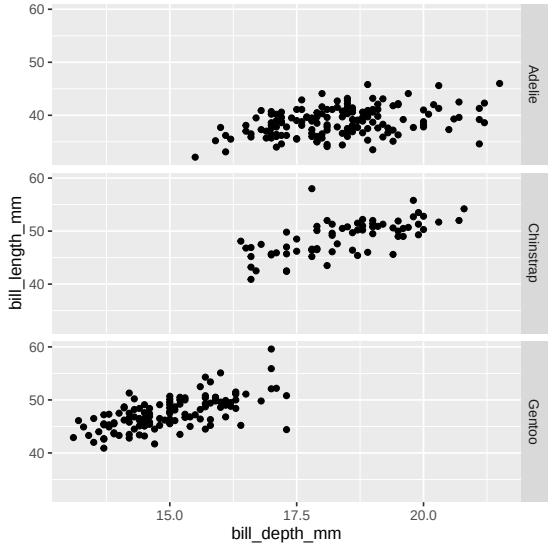
facet_grid() columns

```
ggplot(penguins, aes(x = bill_depth_mm,
                     y = bill_length_mm)
) +
  geom_point() +
  facet_grid(. ~ species)
```



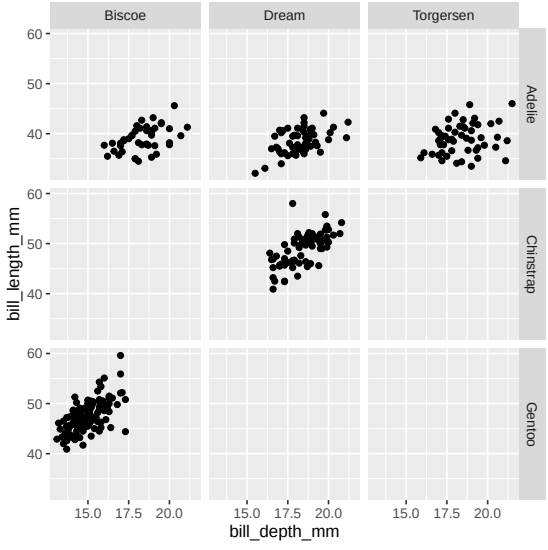
facet_grid() rows

```
ggplot(penguins, aes(x = bill_depth_mm,
                     y = bill_length_mm)
) +
  geom_point() +
  facet_grid(species ~ .)
```



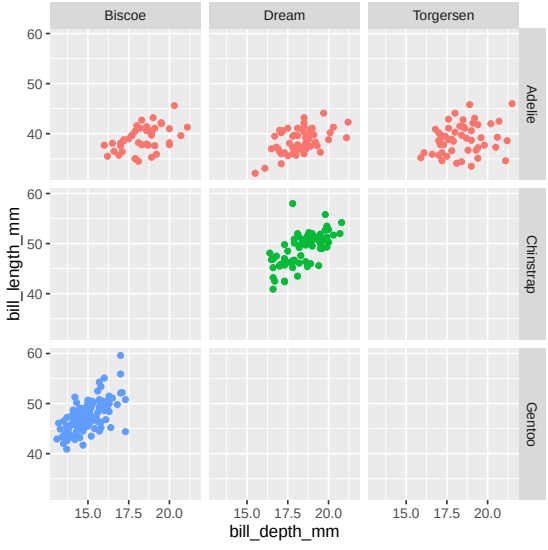
`facet_grid()` both rows and columns

```
ggplot(penguins, aes(x = bill_depth_mm,
                     y = bill_length_mm)
) +
  geom_point() +
  facet_grid(species ~ island)
```



Faceting and color

```
ggplot(penguins, aes(x = bill_depth_mm,
                     y = bill_length_mm)
) +
  geom_point(aes(col = species)) +
  facet_grid(species ~ island) +
  # this removes the legend
  guides(col = "none")
```

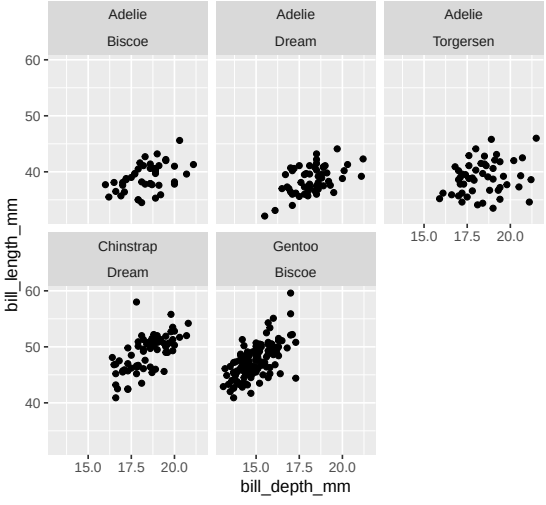


facet_wrap()

Instead of a matrix, `facet_wrap()` wraps a sequence of panels into 2 dimensions.

```
ggplot(penguins, aes(x = bill_depth_mm,
                     y = bill_length_mm)
) +
  geom_point() +
  facet_wrap(species ~ island)
```

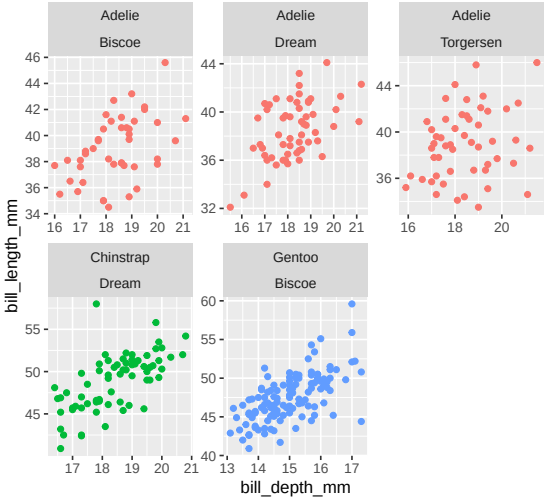
```
  # control number of rows and columns
  # by nrow = xxx and ncol = yyy
```



Free scales

It's not really recommended, but it is possible to free the scales of the x and y axis.

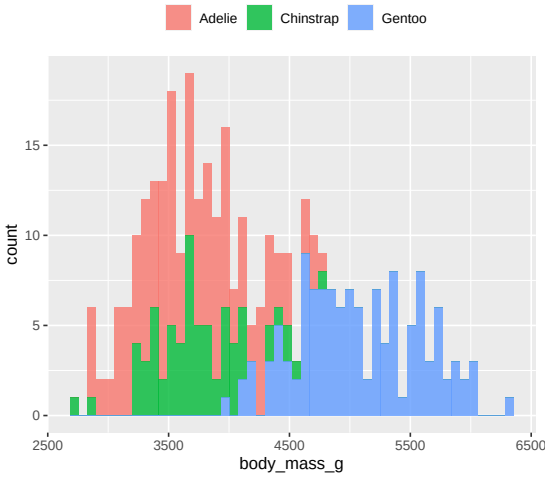
```
ggplot(penguins, aes(x = bill_depth_mm,
                     y = bill_length_mm)
) +
  geom_point(aes(col = species)) +
  facet_wrap(species ~ island, scales = "free")
  guides(col = "none")
```



Histograms

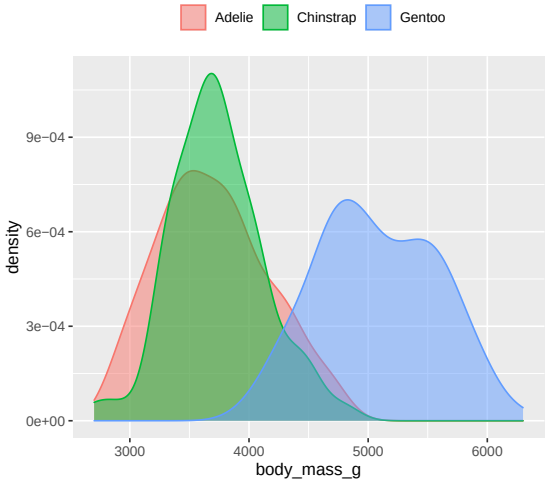
```
ggplot(penguins, aes(x = body_mass_g)) +
  geom_histogram(aes(fill = species),
                 bins = 50,
                 alpha = 0.8) +
  labs(fill = NULL) +
  theme(legend.position = "top")
```

```
  # more on themes later!
```



Density plots

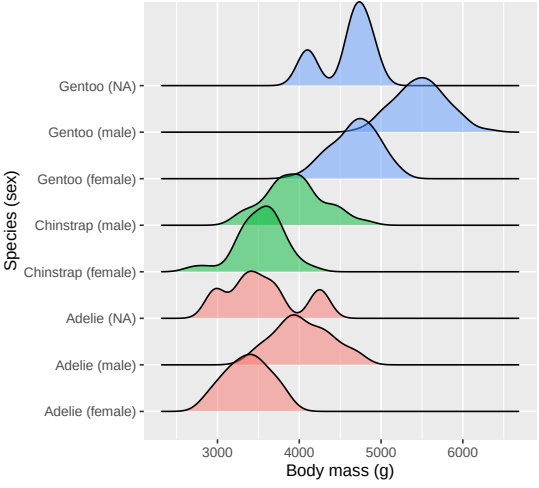
```
ggplot(penguins, aes(x = body_mass_g)) +
  geom_density(aes(fill = species,
                   col = species),
               alpha = 0.5) +
  labs(fill = NULL, col = NULL) +
  theme(legend.position = "top")
```



Ridge plots

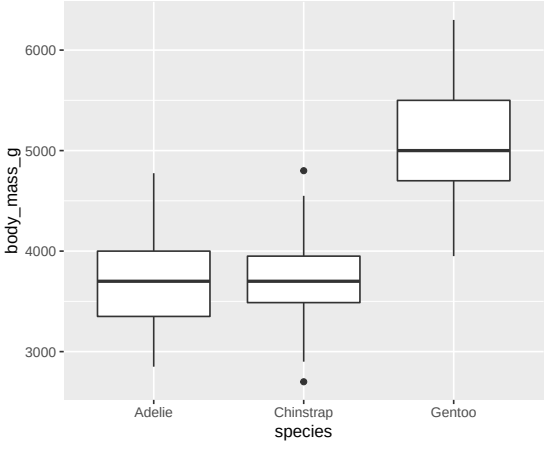
```
ggplot(
  penguins,
  aes(
    x = body_mass_g,
    y = paste0(species, " (", sex, ")"),
    fill = species
  )
) +
  ggridges::geom_density_ridges(alpha = 0.5) +
  labs(x = "Body mass (g)",
       y = "Species (sex)") +
  guides(fill = "none")
```

```
## Picking joint bandwidth of 127
```

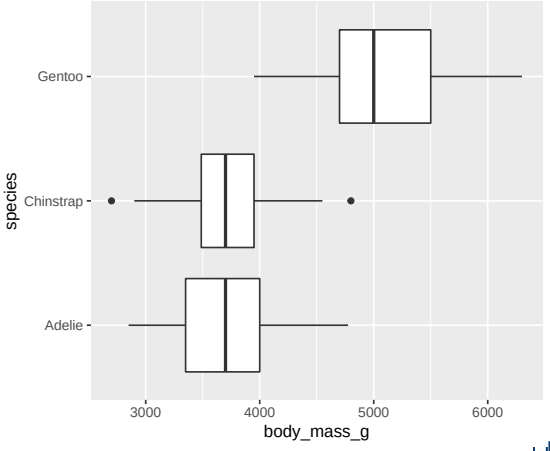


Box plots

```
ggplot(penguins, aes(x = species,
                     y = body_mass_g)) +
  geom_boxplot()
```

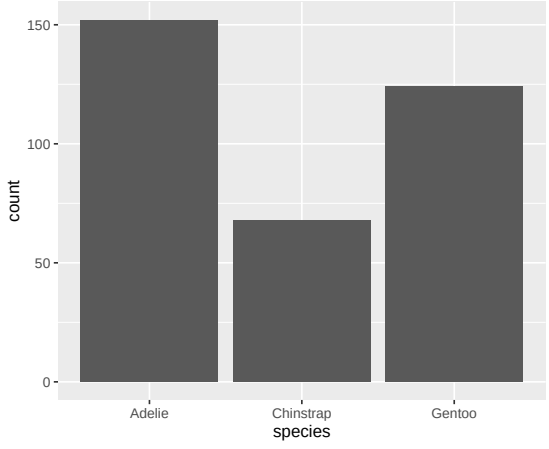



```
ggplot(penguins, aes(x = body_mass_g,
                     y = species)) +
  geom_boxplot()
```

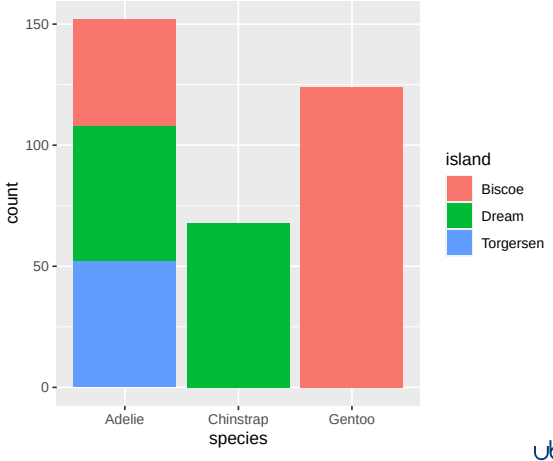


Bar plots

```
ggplot(penguins, aes(x = species)) +
  geom_bar()
```

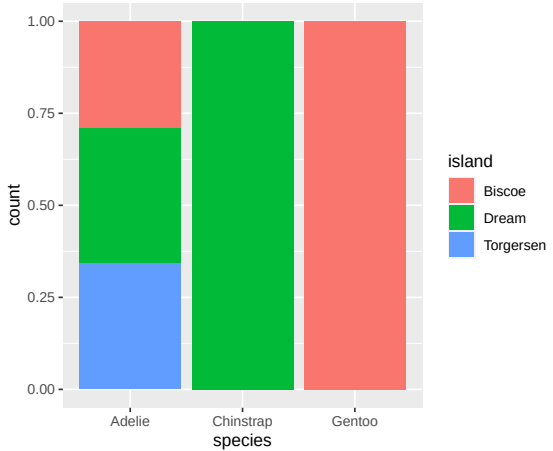



```
ggplot(penguins, aes(x = species,
                     fill = island)) +
  geom_bar()
```

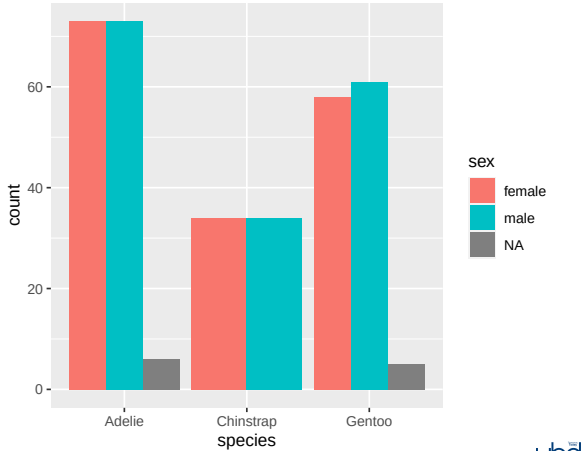


Bar plots (cont.)

```
ggplot(penguins, aes(x = species,
                     fill = island)) +
  geom_bar(position = "fill")
```

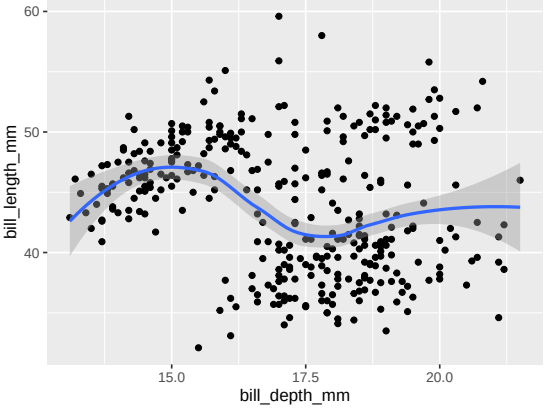



```
ggplot(penguins, aes(x = species,
                     fill = sex)) +
  geom_bar(position = "dodge")
```

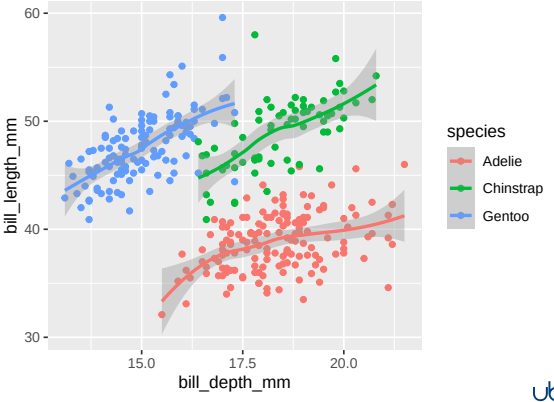


Scatter plot with geom_smooth()

```
ggplot(penguins, aes(x = bill_depth_mm,
                     y = bill_length_mm)) +
  geom_point() +
  geom_smooth()
```

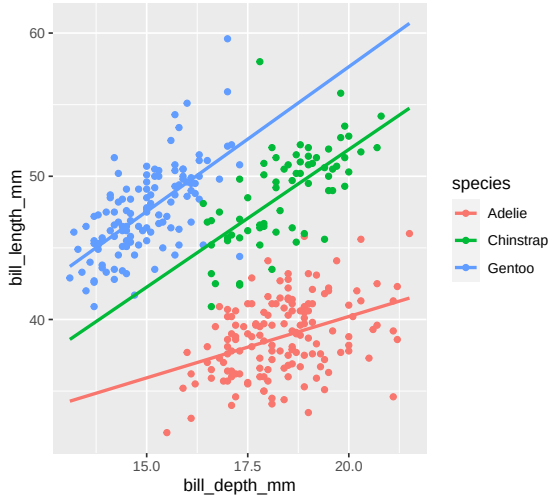



```
ggplot(penguins, aes(x = bill_depth_mm,
                     y = bill_length_mm,
                     col = species)) +
  geom_point() +
  geom_smooth()
```



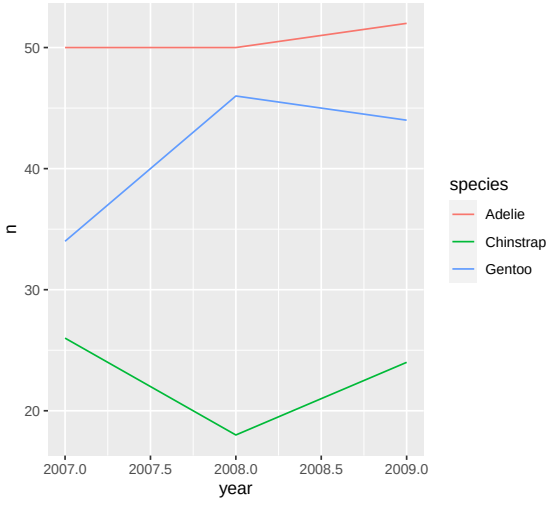
Scatter plot with `geom_smooth()` (cont.)

```
ggplot(penguins, aes(x = bill_depth_mm,
                     y = bill_length_mm,
                     col = species)) +
  geom_point() +
  geom_smooth(method = "lm",
              se = FALSE,
              fullrange = TRUE)
```



Line plots

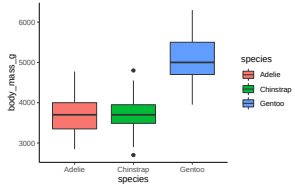
```
penguins %>%
  count(species, year) %>%
  ggplot(
    aes(
      x = year,
      y = n,
      color = species,
      group = species
    )
  ) +
  geom_line()
```



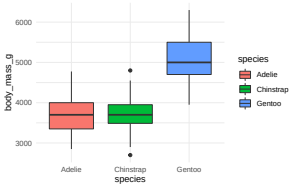
ggplot2 themes

```
g <- ggplot(penguins, aes(species, body_mass_g, fill = species)) + geom_boxplot()
```

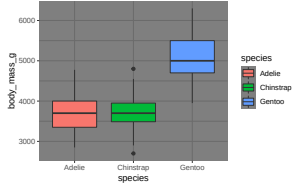
```
g + theme_classic()
```



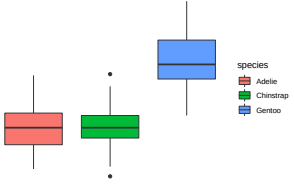

```
g + theme_minimal()
```




```
g + theme_dark()
```



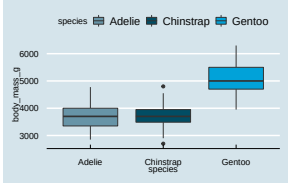

```
g + theme_void()
```



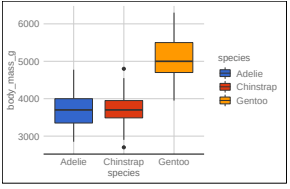
ggthemes

```
library(ggthemes)
```

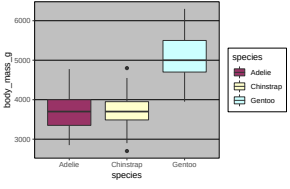
```
g + theme_economist() + scale_fill_economist()
```



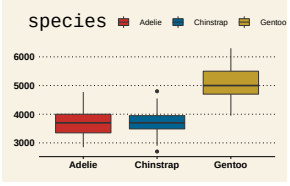

```
g + theme_gdocs() + scale_fill_gdocs()
```




```
g + theme_excel() + scale_fill_excel()
```



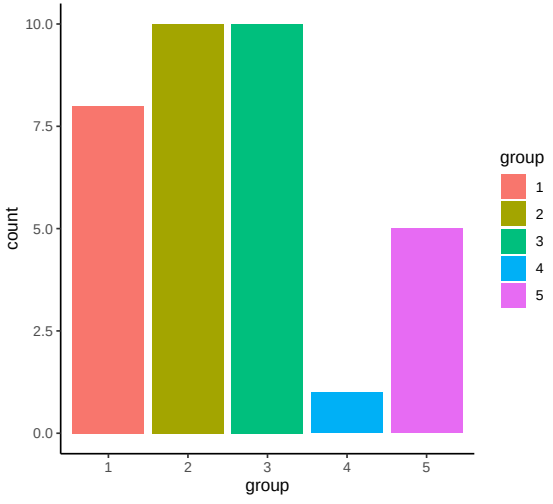

```
g + theme_wsj() + scale_fill_wsj()
```



Color scales

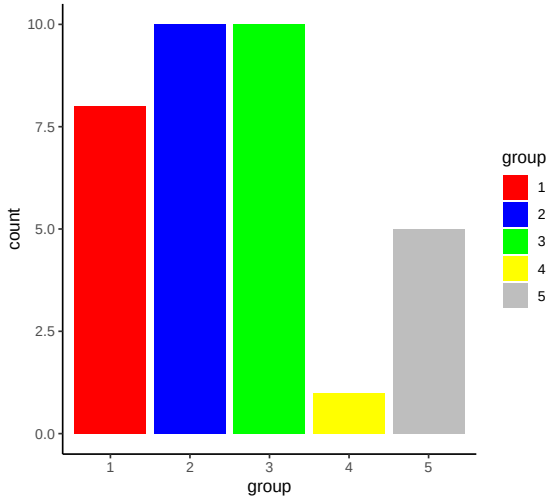
ggplot2's default colour scheme is simply an equally spaced hue around the colour wheel.

```
ngroup <- 5
tibble(
  group = factor(1:ngroup),
  count = sample(10, size = ngroup,
                 replace = TRUE)
) %>%
  ggplot(aes(group, count, fill = group)) +
  geom_bar(stat = "identity") +
  theme_classic() -> p
p
```



Manually changing colours

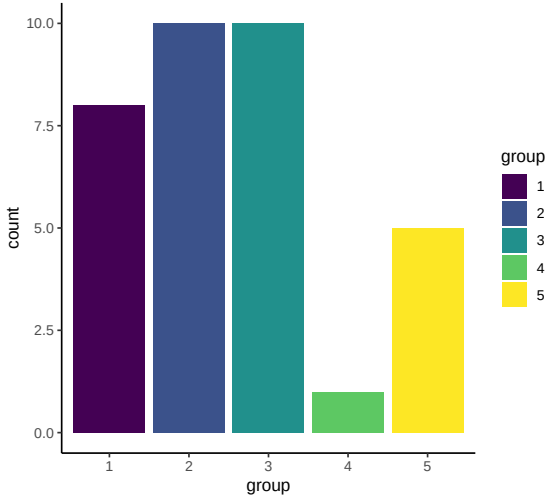
```
p +
  scale_fill_manual(
    values = c("red", "blue", "green",
               "yellow", "grey")
  )
```



Viridis colour scale

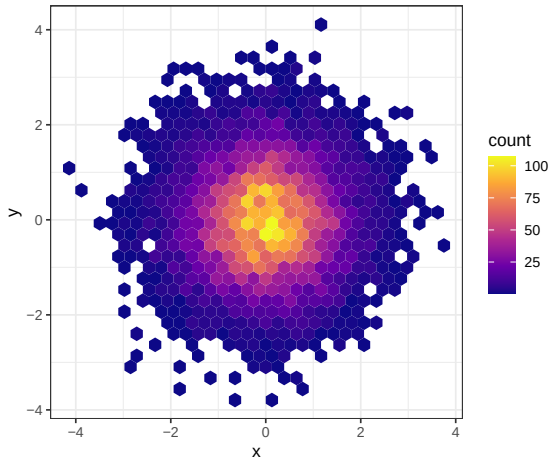
```
p +
  scale_fill_viridis_d()
```

```
  # Here the _d stands for discrete.
  # Other scales include
  # scale_fill_viridis_c() &
  # scale_fill_viridis_b()
```



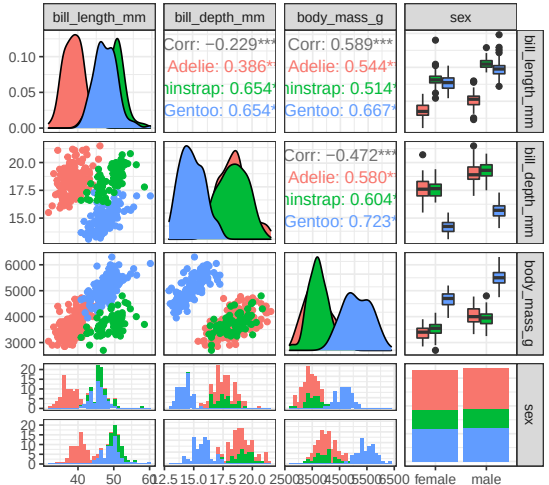
Viridis colour scale (cont.)

```
tibble(
  x = rnorm(10000),
  y = rnorm(10000)
) %>%
  ggplot(aes(x, y)) +
  geom_hex() +
  coord_fixed() +  # ensures fixed x/y scales
  scale_fill_viridis_c(option = "plasma") +
  theme_bw()
```



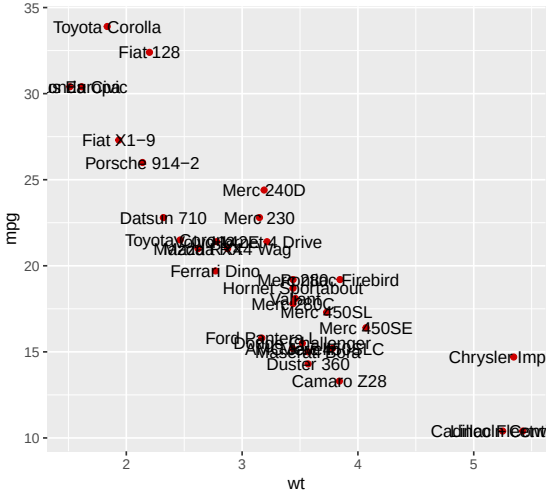
GGAlly's ggpairs()

```
GGally::ggpairs(
  data = drop_na(penguins),
  columns = c("bill_length_mm",
              "bill_depth_mm",
              "body_mass_g",
              "sex"),
  mapping = aes(col = species)
) +
  theme_bw()
```



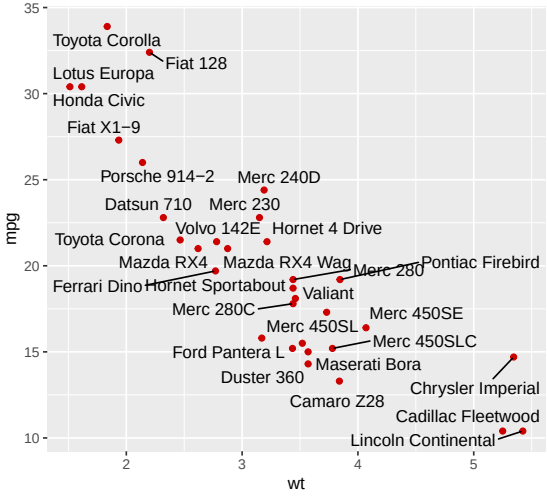
Label points

```
rownames_to_column(mtcars) %>%
  ggplot(aes(wt, mpg, label = rowname)) +
  geom_point(col = "red3") +
  geom_text()
```



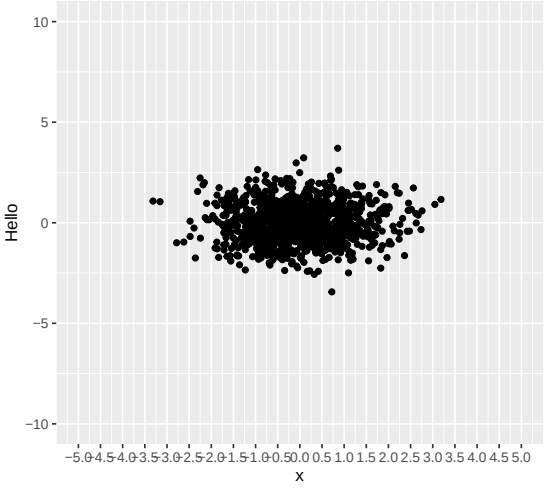
Label points with ggrepel

```
rownames_to_column(mtcars) %>%
  ggplot(aes(wt, mpg, label = rowname)) +
  geom_point(col = "red3") +
  ggrepel::geom_text_repel()
```



Adjusting axis scales

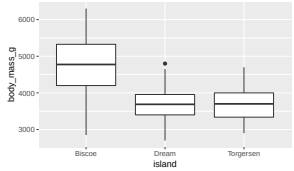
```
tibble(
  x = rnorm(1000),
  y = rnorm(1000)
) %>%
  ggplot(aes(x, y)) +
  geom_point() +
  scale_x_continuous(
    limits = c(-5, 5),
    breaks = seq(-5, 5, by = 0.5)
  ) +
  scale_y_continuous(
    limits = c(-10, 10),
    name = "Hello"
  )
```



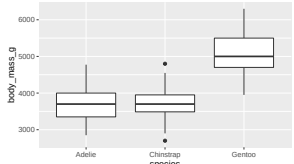
Plot composition

```
library(patchwork)
```

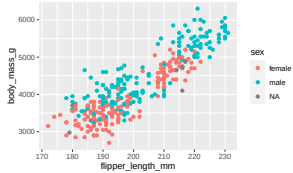
```
(p1 <- ggplot(penguins) +
  geom_boxplot(aes(island, body_mass_g)))
```




```
(p2 <- ggplot(penguins) +
  geom_boxplot(aes(species, body_mass_g)))
```



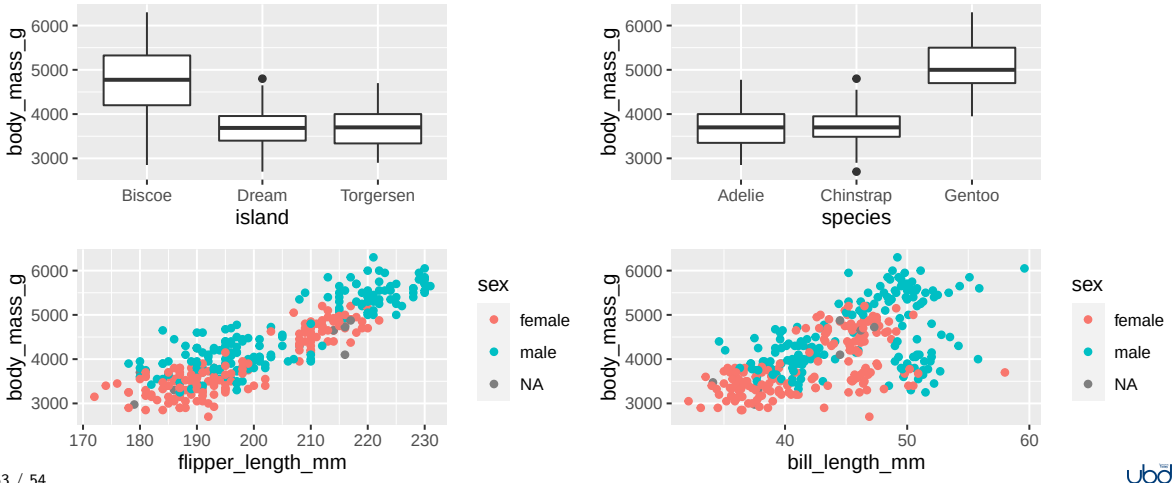

```
(p3 <- ggplot(penguins) +
  geom_point(aes(flipper_length_mm,
                 body_mass_g, color = sex)))
```




```
(p4 <- ggplot(penguins) +
  geom_point(aes(bill_length_mm,
                 body_mass_g, color = sex)))
```

Plot composition (cont.)

```
p1 + p2 + p3 + p4 + plot_layout(nrow = 2)
```



Plot composition (cont.)

```
p1 / (p2 + p3 + p4)
```