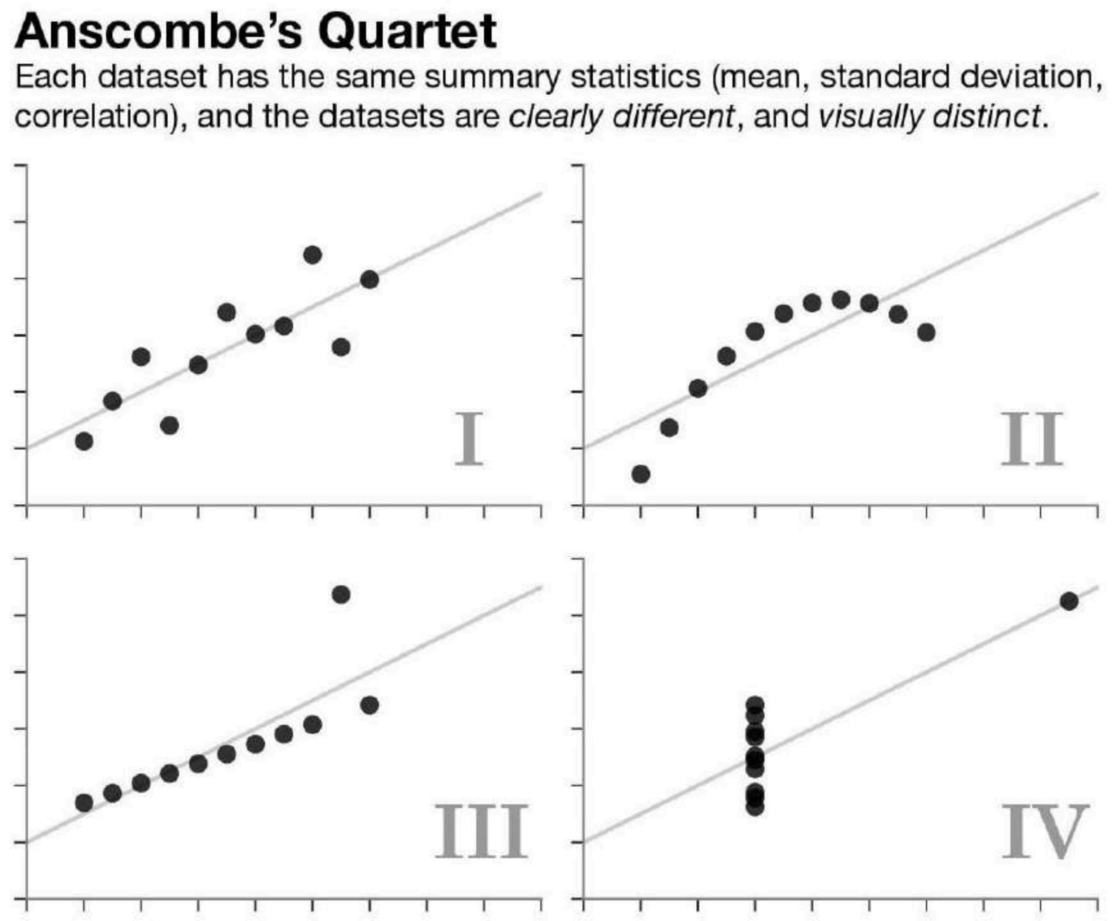
Learning `ggplot2` in `R`

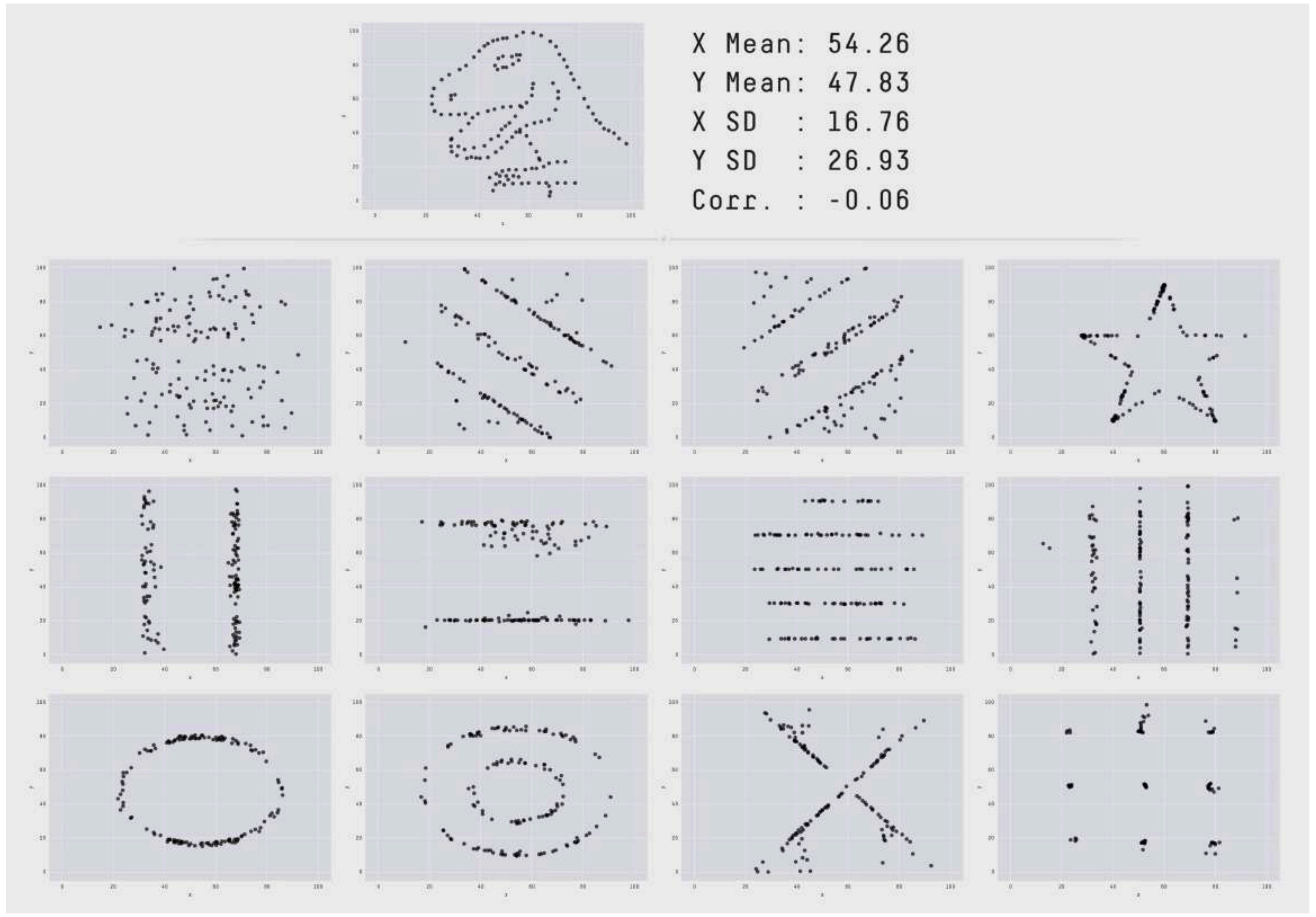
Abu Bakar Siddique, SLUBI

2025-04-07

Graphs

- Essential part of data analyses
- Data with same summary statistics can look very different when plotted out.

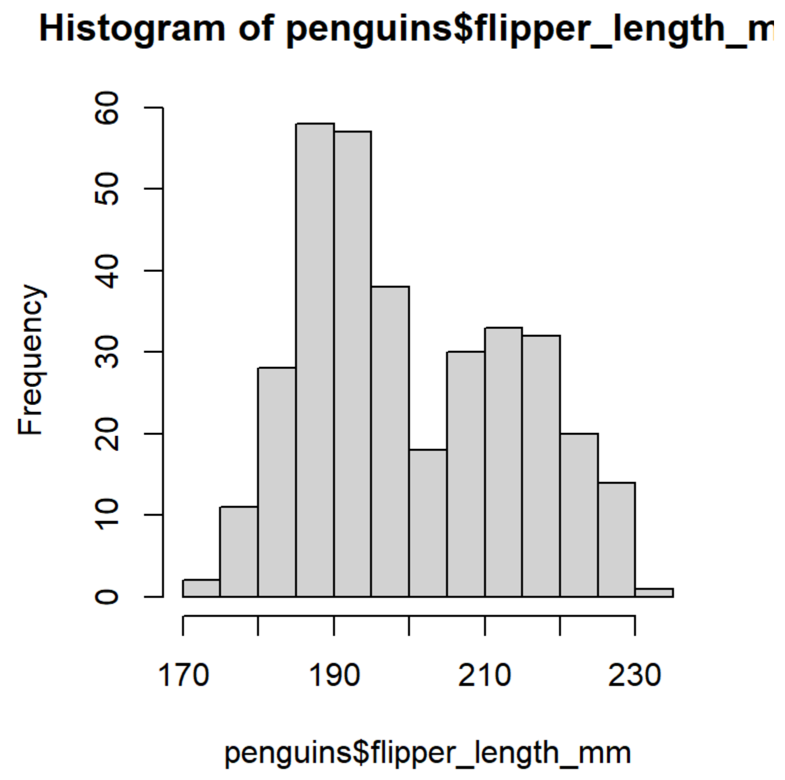




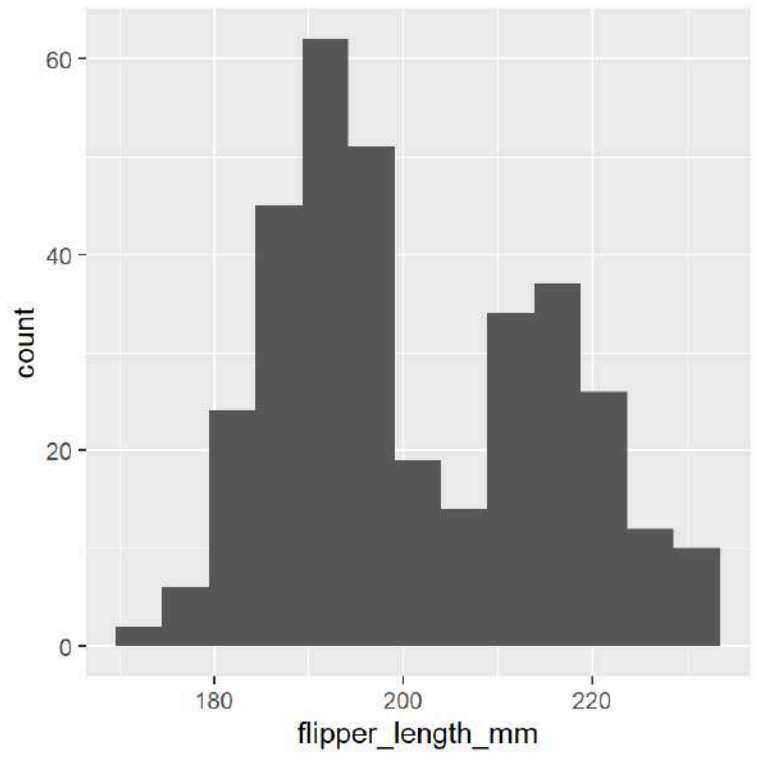
Anscombe's quartet, Datasaurus

Base Graphics vs ggplot2

```r
hist(penguins$flipper_length_mm)
```



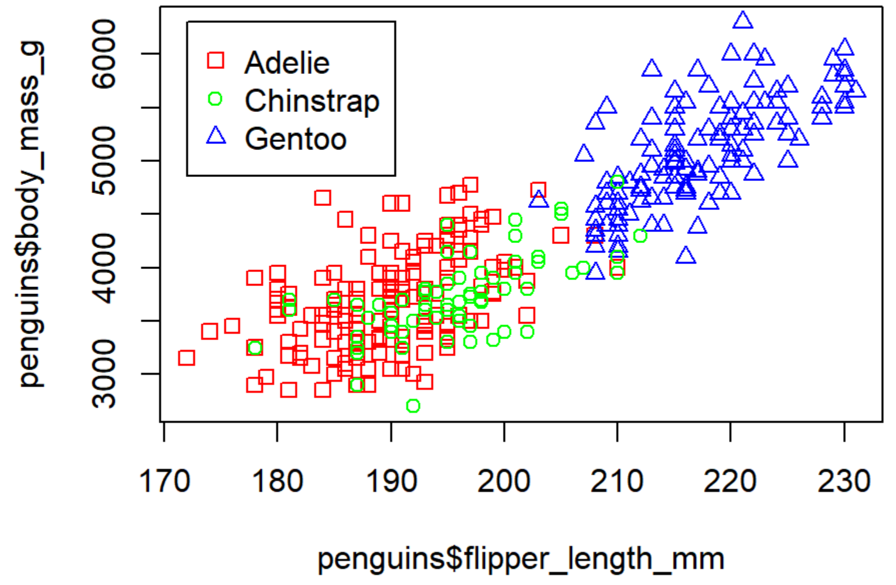
```r
ggplot(penguins,aes(flipper_length_mm))+
  geom_histogram(bins = 13)
```



Base Graphics vs ggplot2

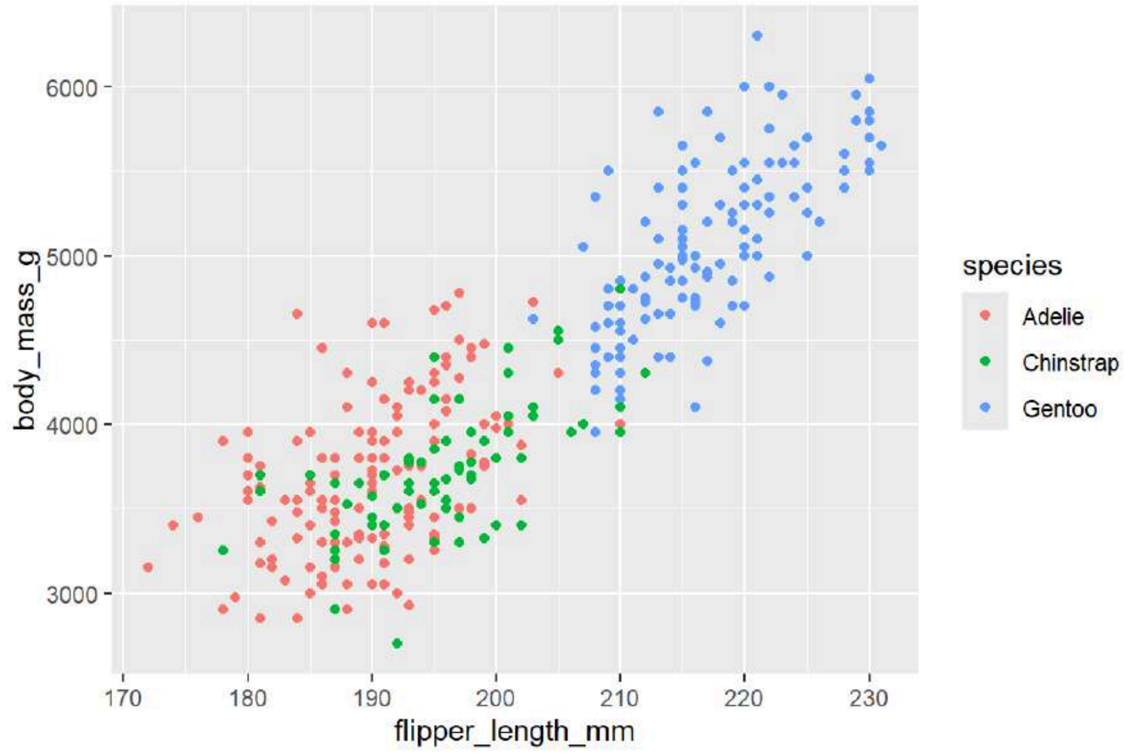
basic r plot

```r
plot(penguins$flipper_length_mm,penguins$body_mass_g,
      col=c("red","green","blue")[penguins$species],
      pch=c(0,1,2)[penguins$species])
legend(x=172,y=6300,
        legend=c("Adelie","Chinstrap","Gentoo"),
        pch=c(0,1,2),col=c("red","green","blue"))
```



ggplot

```r
ggplot(penguins, aes(flipper_length_mm,body_mass_g,
  color=species)) +
  geom_point()
```

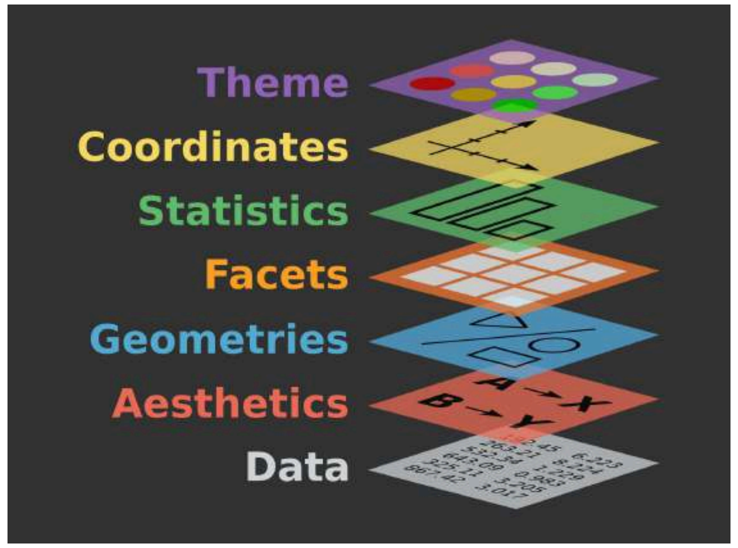


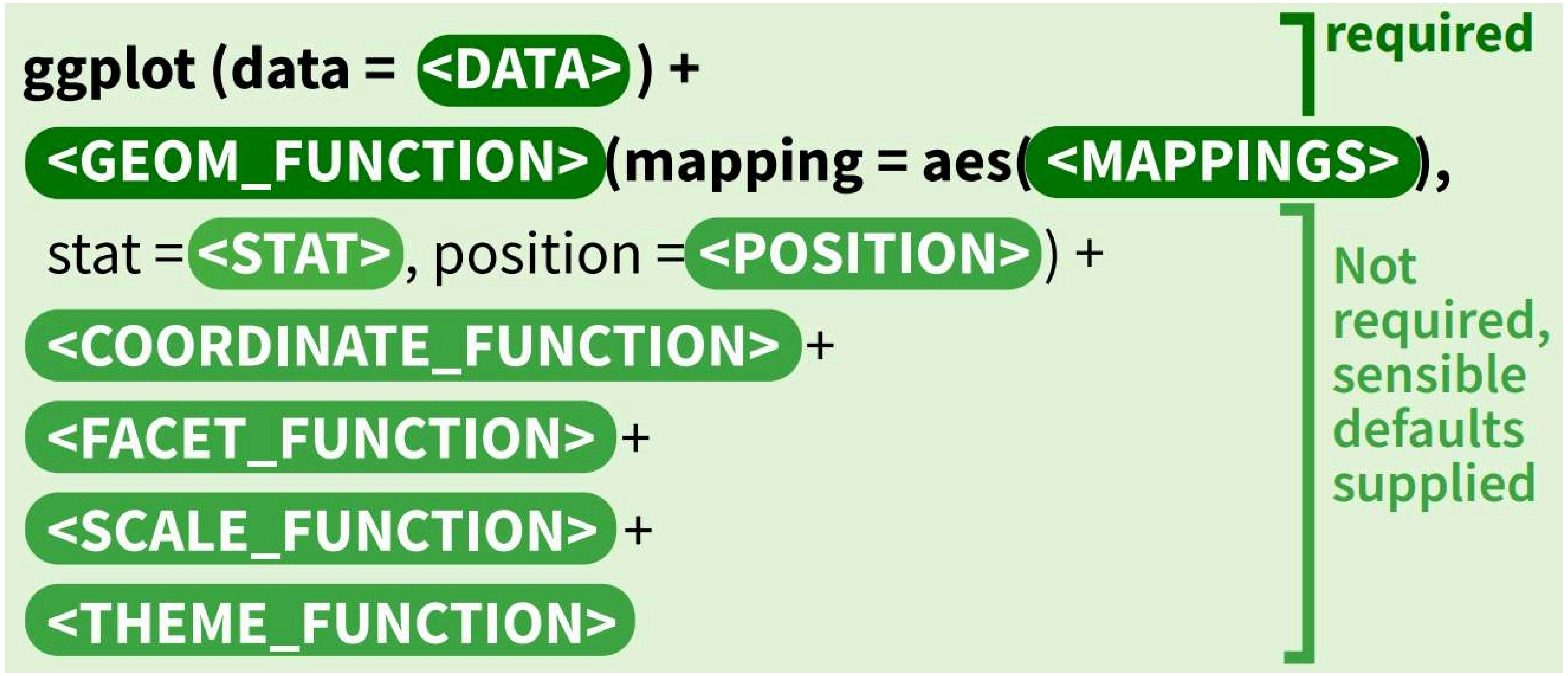
Why `ggplot2`?

- Consistent code
- Flexible (Add/remove components)
- Automatic legends, colors etc
- Save plot objects
- Themes for reusing styles
- Numerous add-ons/extensions
- Nearly complete structured graphing solution
- Adapted to other programming languages
  - Gadfly in Julia, gramm in MatLab, GGPlot in Perl, Vega in Javascript, PlotNine, ggpy, lets-plot in Python.

Grammar Of Graphics

Building A Graph: · Syntax





```
ggplot (data = <DATA>) +
<GEOM_FUNCTION>(mapping = aes(<MAPPINGS>),
 stat = <STAT>, position = <POSITION>) +
<COORDINATE_FUNCTION> +
<FACET_FUNCTION> +
<SCALE_FUNCTION> +
<THEME_FUNCTION>
```

required

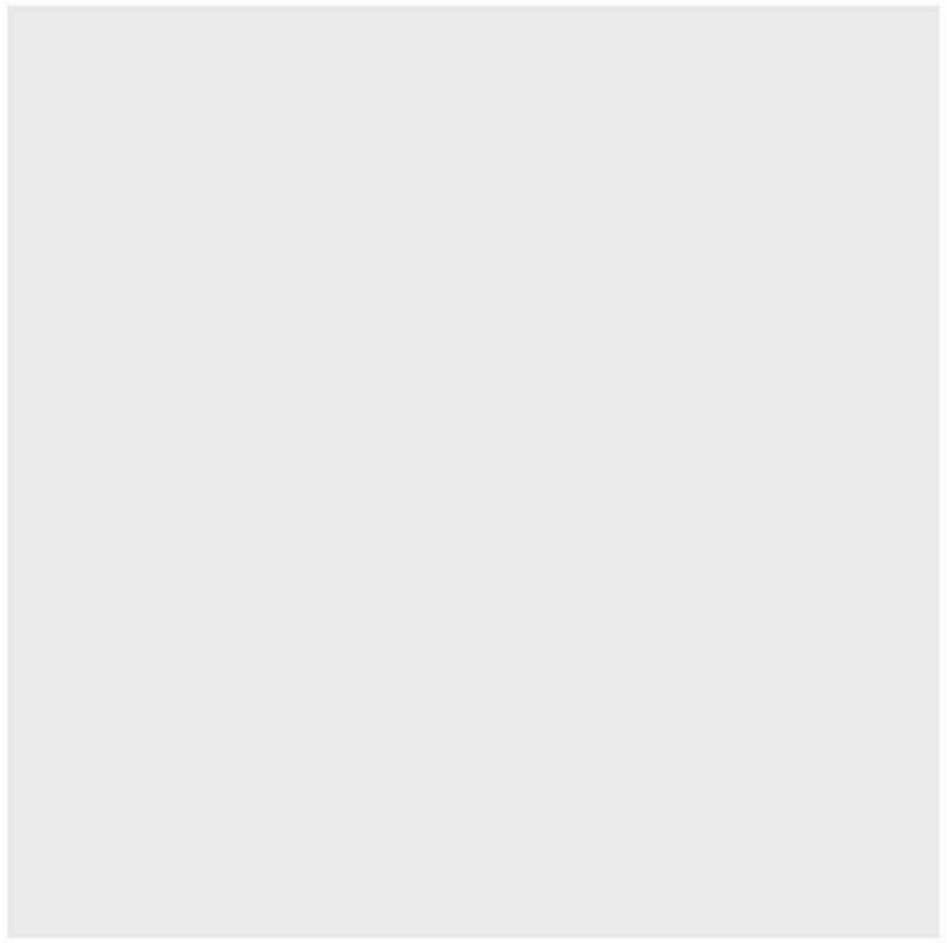
Not required, sensible defaults supplied

Building A Graph

```
require(ggplot2)         # load ggplot2
require(palmerpenguins) # load penguins data pack

data("penguins")         # load penguins data
```

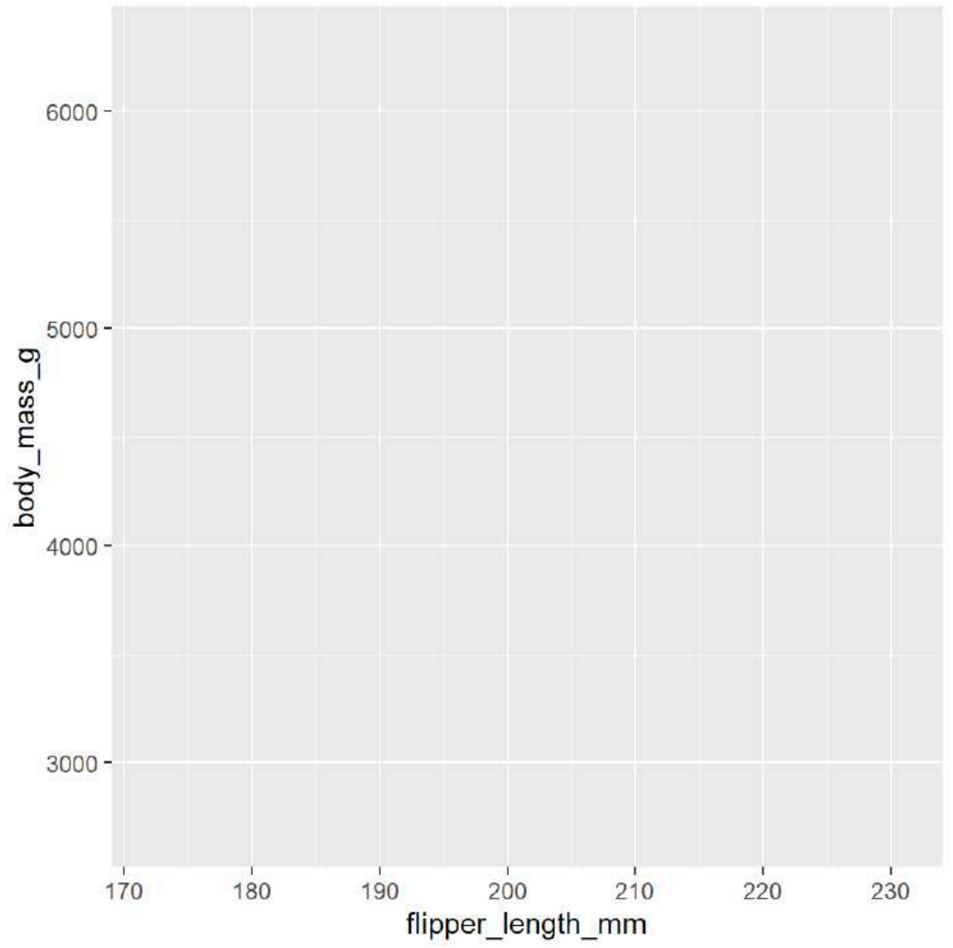
Building A Graph

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



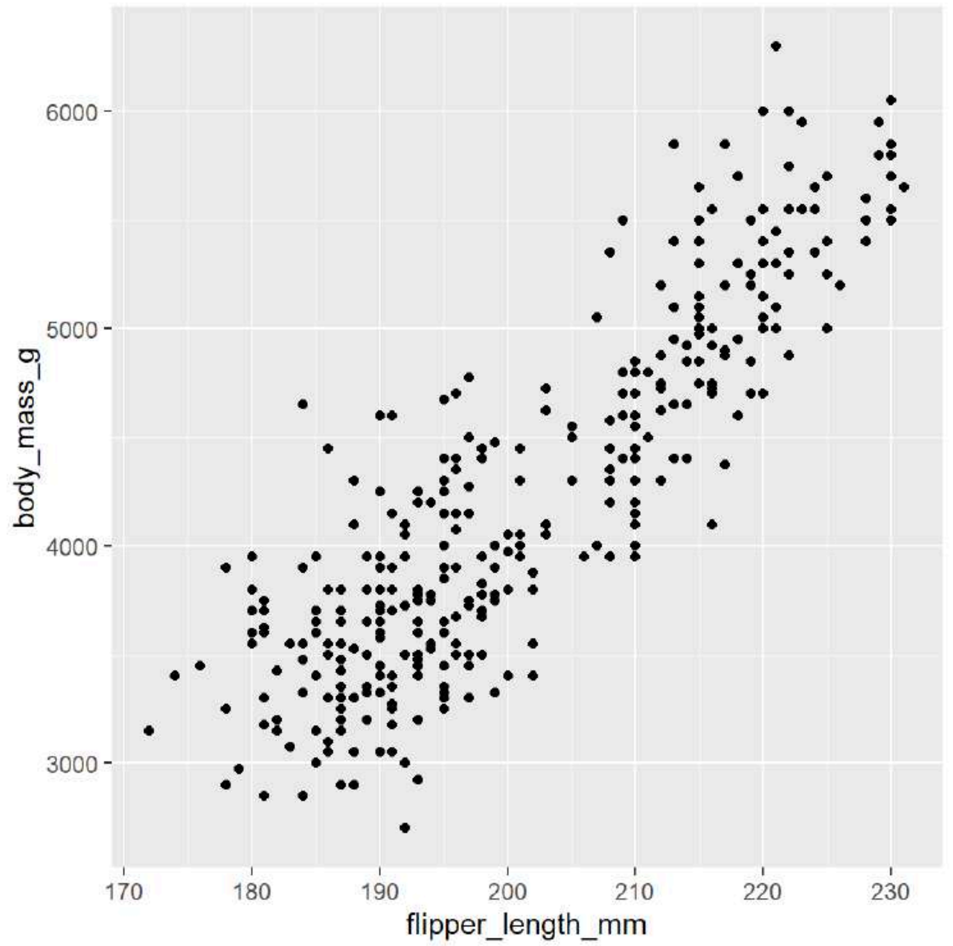
Building A Graph

```
ggplot(data = penguins,
mapping = aes(x = flipper_length_mm,
              y = body_mass_g))
```



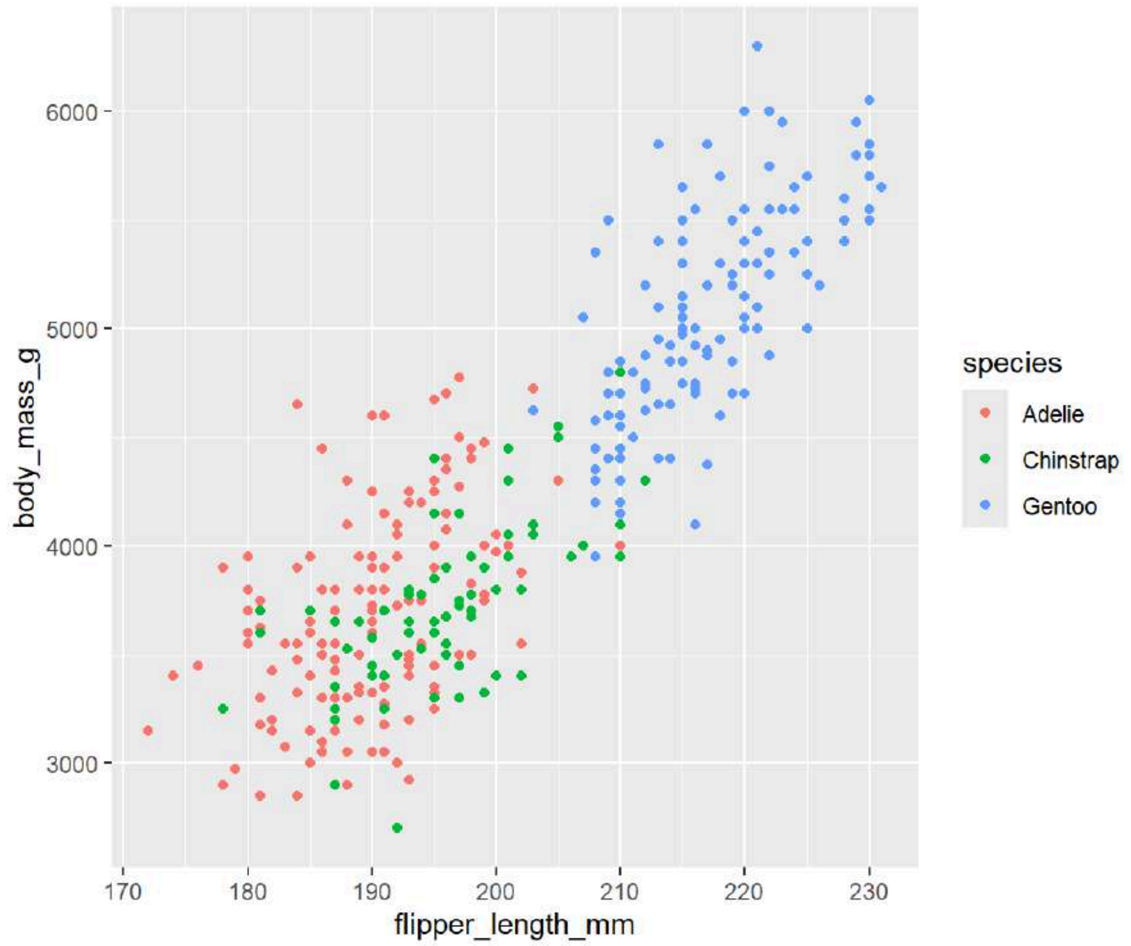
Building A Graph

```
ggplot(data = penguins,
mapping = aes(x = flipper_length_mm,
              y = body_mass_g)) +
geom_point()
```



Building A Graph

```r
ggplot(data = penguins,
mapping = aes(x = flipper_length_mm,
              y = body_mass_g,
              colour = species)) +
geom_point()
```

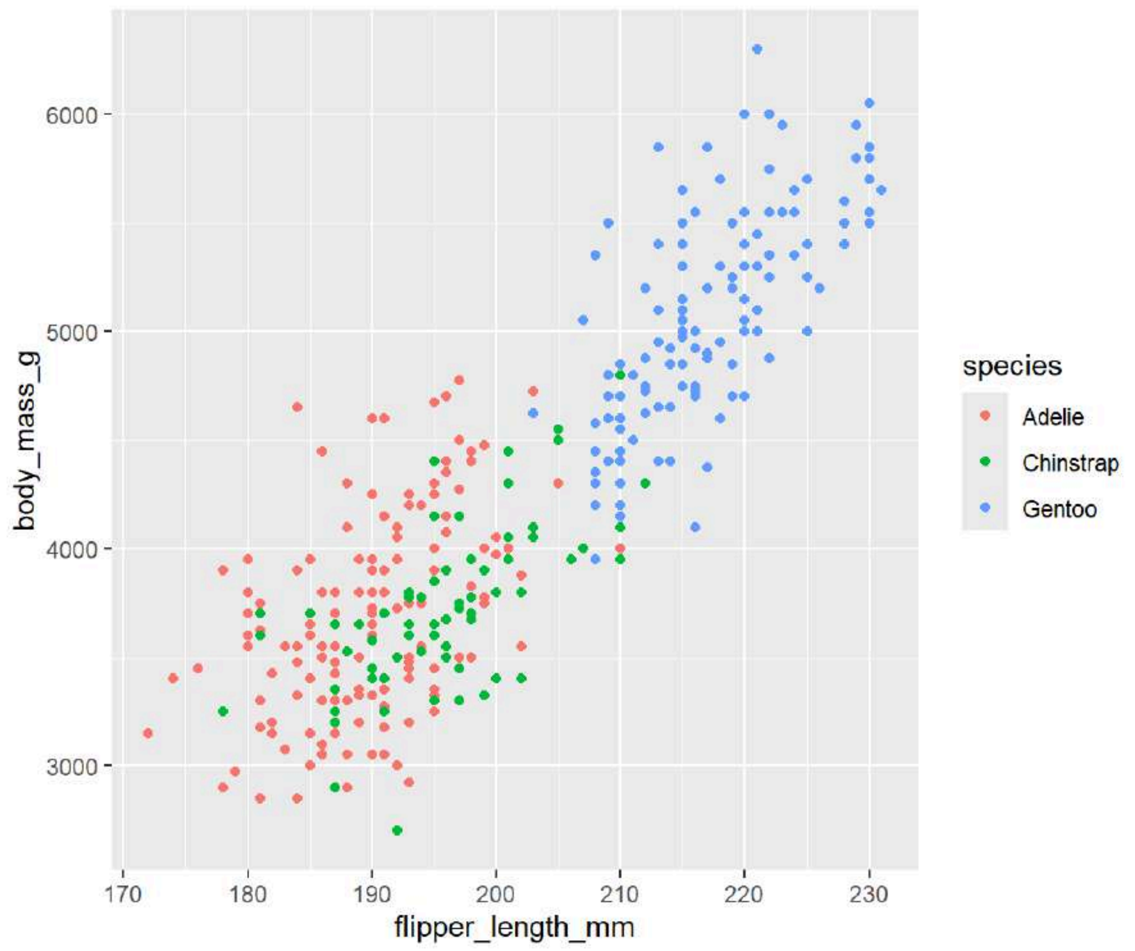


Building A Graph

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

Or

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

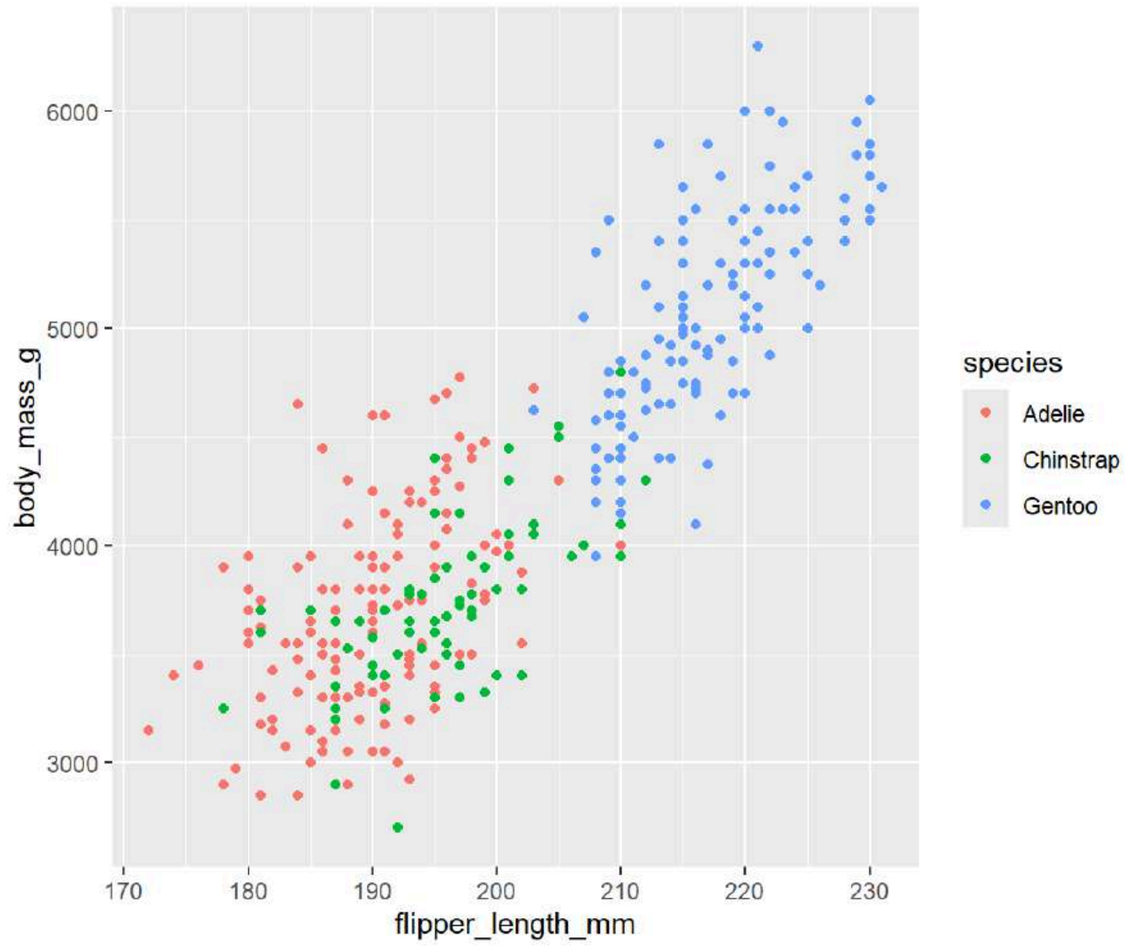


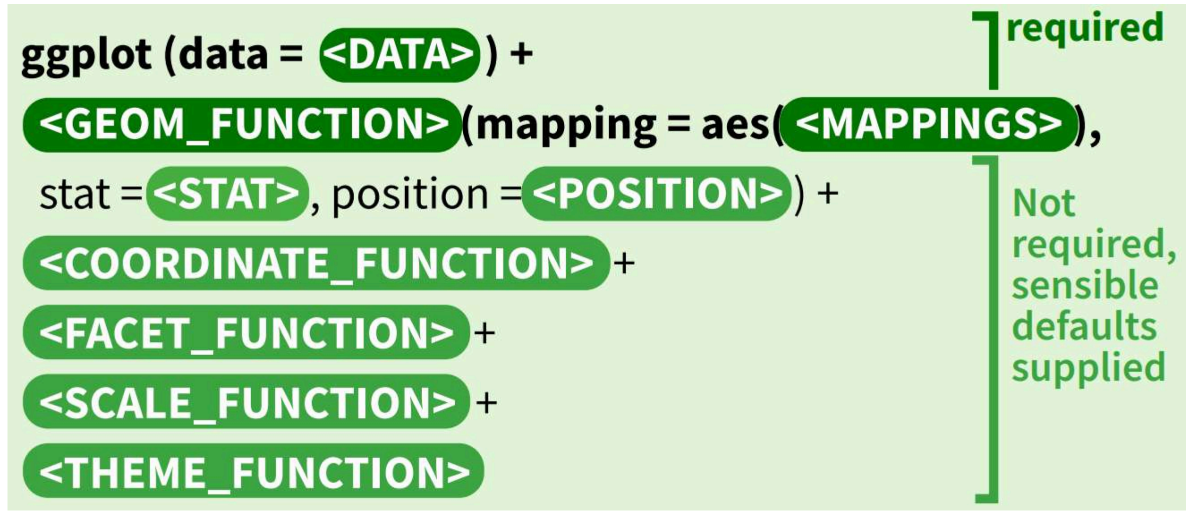
Building A Graph

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

Or

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





ggplot (data = <DATA>) +
<GEOM_FUNCTION>(mapping = aes(<MAPPINGS>),
stat = <STAT>, position = <POSITION>) +
<COORDINATE_FUNCTION> +
<FACET_FUNCTION> +
<SCALE_FUNCTION> +
<THEME_FUNCTION>

required

Not required, sensible defaults supplied

Data · `penguins`

- Input data is always an R `data.frame` object

species	island	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_
Adelie	Torgersen	39.1	18.7	181	3750
Adelie	Torgersen	39.5	17.4	186	3800
Adelie	Torgersen	40.3	18.0	195	3250

```r
str(penguins)
```

```
tibble [344 × 8] (S3: tbl_df/tbl/data.frame)
 $ species          : Factor w/ 3 levels "Adelie","Chinstrap",..: 1 1 1 1 1 1 1 1 1 1 ...
 $ island           : Factor w/ 3 levels "Biscoe","Dream",..: 3 3 3 3 3 3 3 3 3 3 ...
 $ bill_length_mm   : num [1:344] 39.1 39.5 40.3 NA 36.7 39.3 38.9 39.2 34.1 42 ...
 $ bill_depth_mm    : num [1:344] 18.7 17.4 18 NA 19.3 20.6 17.8 19.6 18.1 20.2 ...
 $ flipper_length_mm: int [1:344] 181 186 195 NA 193 190 181 195 193 190 ...
 $ body_mass_g      : int [1:344] 3750 3800 3250 NA 3450 3650 3625 4675 3475 4250 ...
 $ sex              : Factor w/ 2 levels "female","male": 2 1 1 NA 1 2 1 2 NA NA ...
 $ year             : int [1:344] 2007 2007 2007 2007 2007 2007 2007 2007 2007 2007 ...
```

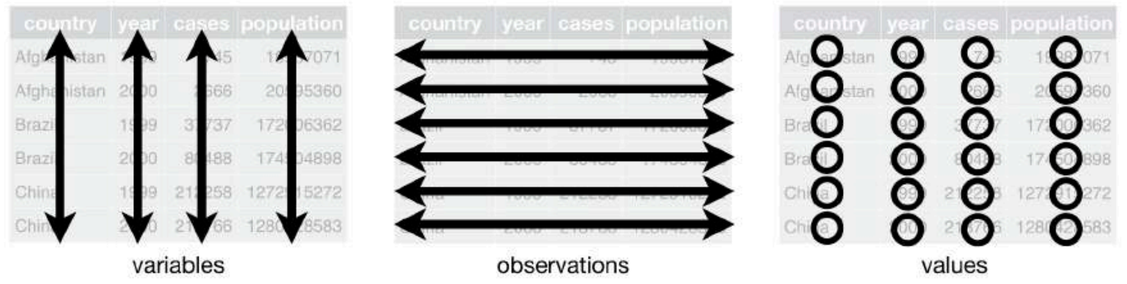
Data · `diamonds`

carat	cut	color	clarity	depth	table	price	x	y	z
0.23	Ideal	E	SI2	61.5	55	326	3.95	3.98	2.43
0.21	Premium	E	SI1	59.8	61	326	3.89	3.84	2.31
0.23	Good	E	VS1	56.9	65	327	4.05	4.07	2.31
0.29	Premium	I	VS2	62.4	58	334	4.20	4.23	2.63
0.31	Good	J	SI2	63.3	58	335	4.34	4.35	2.75
0.24	Very Good	J	VVS2	62.8	57	336	3.94	3.96	2.48

```
tibble [53,940 × 10] (S3: tbl_df/tbl/data.frame)
 $ carat  : num [1:53940] 0.23 0.21 0.23 0.29 0.31 0.24 0.24 0.26 0.22 0.23 ...
 $ cut    : Ord.factor w/ 5 levels "Fair"<"Good"<..: 5 4 2 4 2 3 3 3 1 3 ...
 $ color  : Ord.factor w/ 7 levels "D"<"E"<"F"<"G"<..: 2 2 2 6 7 7 6 5 2 5 ...
 $ clarity: Ord.factor w/ 8 levels "I1"<"SI2"<"SI1"<..: 2 3 5 4 2 6 7 3 4 5 ...
 $ depth  : num [1:53940] 61.5 59.8 56.9 62.4 63.3 62.8 62.3 61.9 65.1 59.4 ...
 $ table  : num [1:53940] 55 61 65 58 58 57 57 55 61 61 ...
 $ price  : int [1:53940] 326 326 327 334 335 336 336 337 337 338 ...
 $ x      : num [1:53940] 3.95 3.89 4.05 4.2 4.34 3.94 3.95 4.07 3.87 4 ...
 $ y      : num [1:53940] 3.98 3.84 4.07 4.23 4.35 3.96 3.98 4.11 3.78 4.05 ...
 $ z      : num [1:53940] 2.43 2.31 2.31 2.63 2.75 2.48 2.47 2.53 2.49 2.39 ...
```

Data · `format`

- Transforming data into 'long' or 'wide' formats



Wide

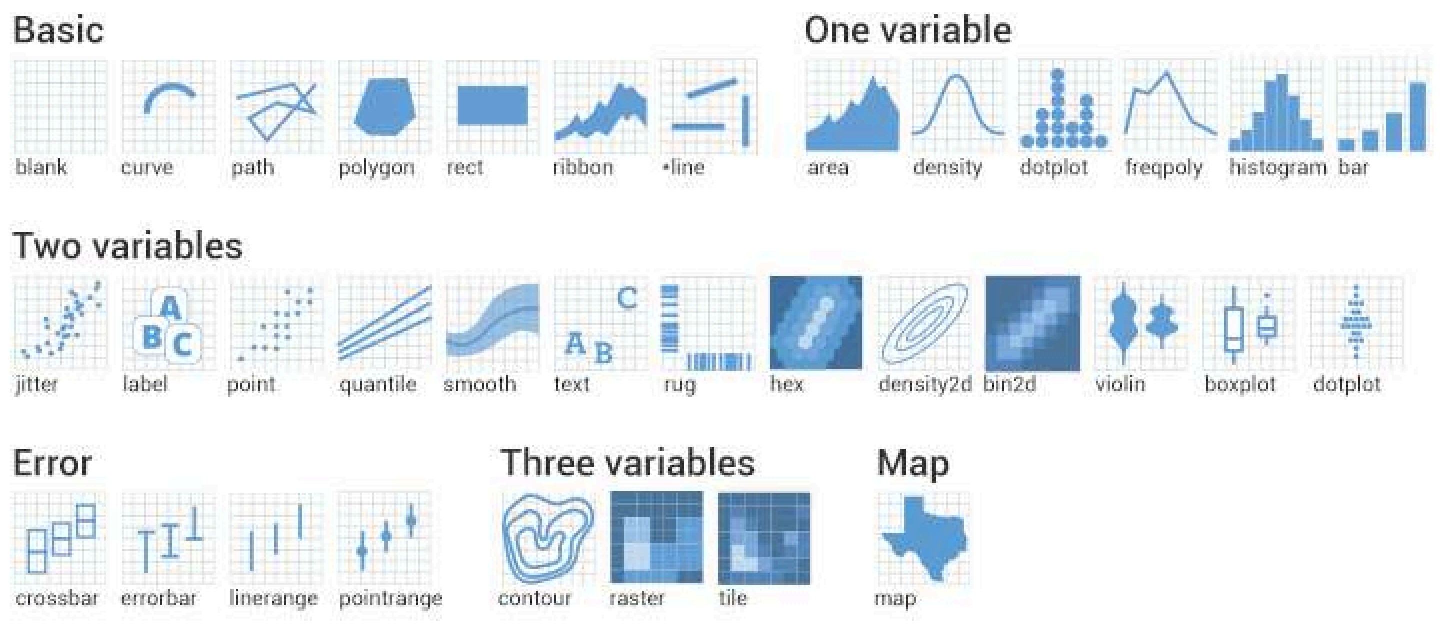
```r
head(penguins, n=4)
```

```
# A tibble: 4 × 8
  species island    bill_length_mm bill_depth_mm flipper_length_mm body_mass_g
  <fct>   <fct>              <dbl>         <dbl>             <int>       <int>
1 Adelie  Torgersen           39.1          18.7               181        3750
2 Adelie  Torgersen           39.5          17.4               186        3800
3 Adelie  Torgersen           40.3          18                 195        3250
4 Adelie  Torgersen           NA            NA                  NA          NA
# ℹ 2 more variables: sex <fct>, year <int>
```

Long

```
  species    island  sex year         variables  value
1  Adelie Torgersen male 2007    bill_length_mm   39.1
2  Adelie Torgersen male 2007     bill_depth_mm   18.7
3  Adelie Torgersen male 2007 flipper_length_mm  181.0
4  Adelie Torgersen male 2007       body_mass_g 3750.0
```

Geoms · types

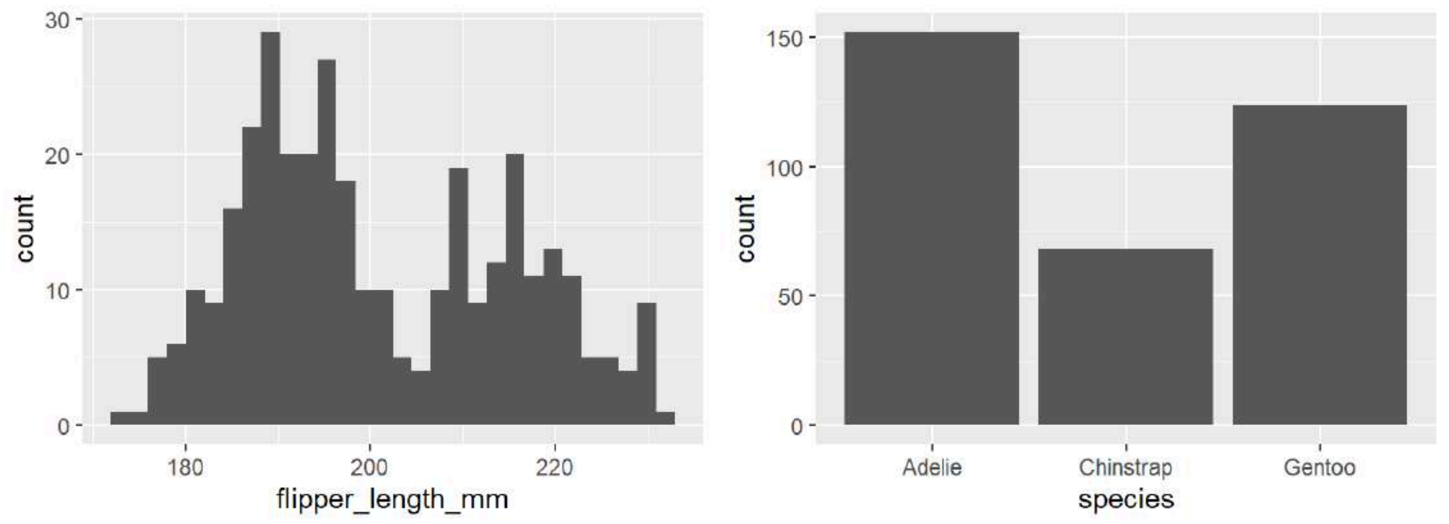

Basic
blank
curve
path
polygon
rect
ribbon
•line
One variable
area
density
dotplot
freqpoly
histogram
bar
Two variables
jitter
label
point
quantile
smooth
text
rug
hex
density2d
bin2d
violin
boxplot
dotplot
Error
crossbar
errorbar
linerange
pointrange
Three variables
contour
raster
tile
Map
map

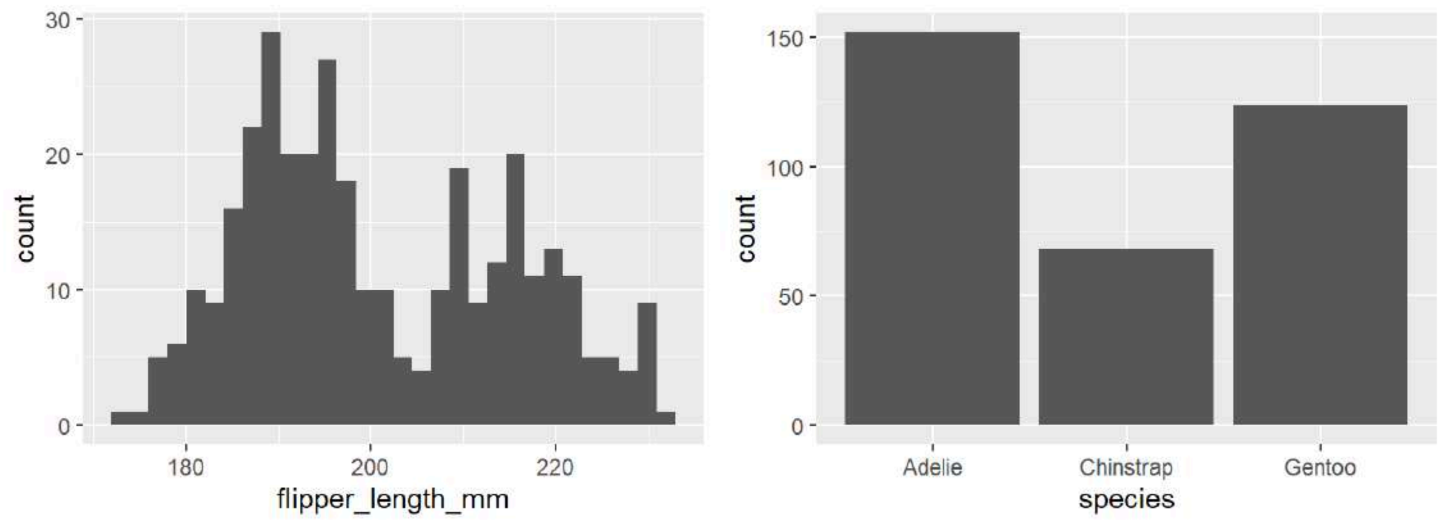
Stats

- `Stats` compute new variables from input data.
- `Geoms` have default stats.
- Plots can be built with stats.

```r
x <- ggplot(data = penguins) +
  geom_bar(aes(x=flipper_length_mm),stat="bin")

y <- ggplot(data = penguins) +
  geom_bar(aes(x=species),stat="count")

wrap_plots(x,y,nrow=1)
```

```r
x <- ggplot(data = penguins) +
  stat_bin(aes(x=flipper_length_mm),geom="bar")

y <- ggplot(data = penguins) +
  stat_count(aes(x=species),geom="bar")

wrap_plots(x,y,nrow=1)
```





Stats

- Stats have default geoms.

plot	stat	geom
histogram	bin	bar
smooth	smooth	line
boxplot	boxplot	boxplot
density	density	line
freqpoly	freqpoly	line

Use `args(geom_bar)` to check arguments.

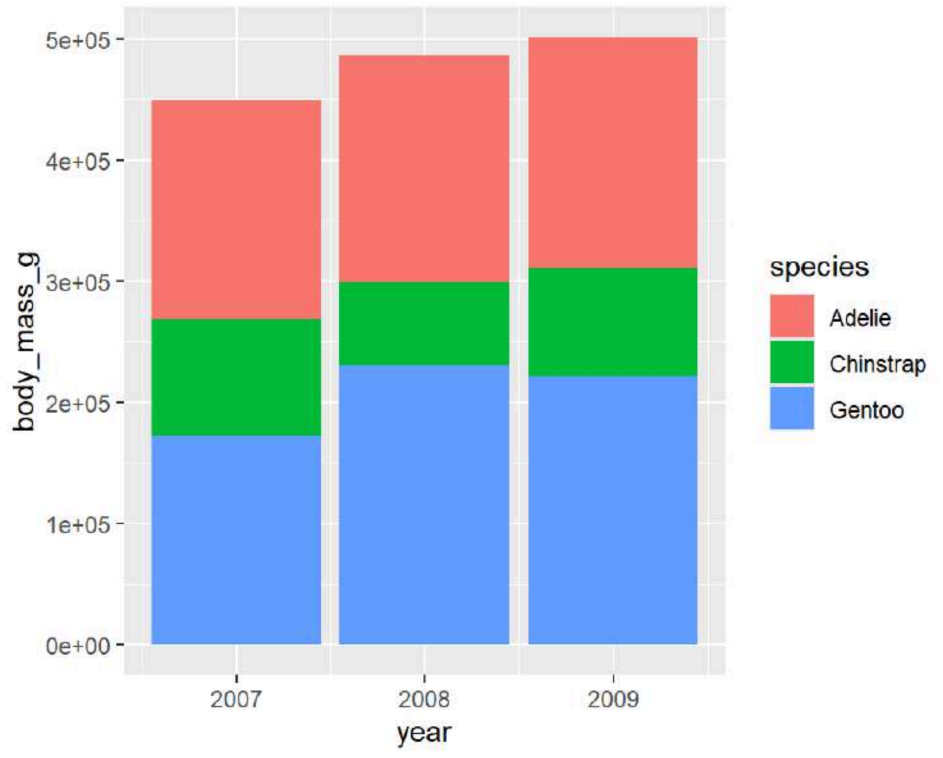
Position

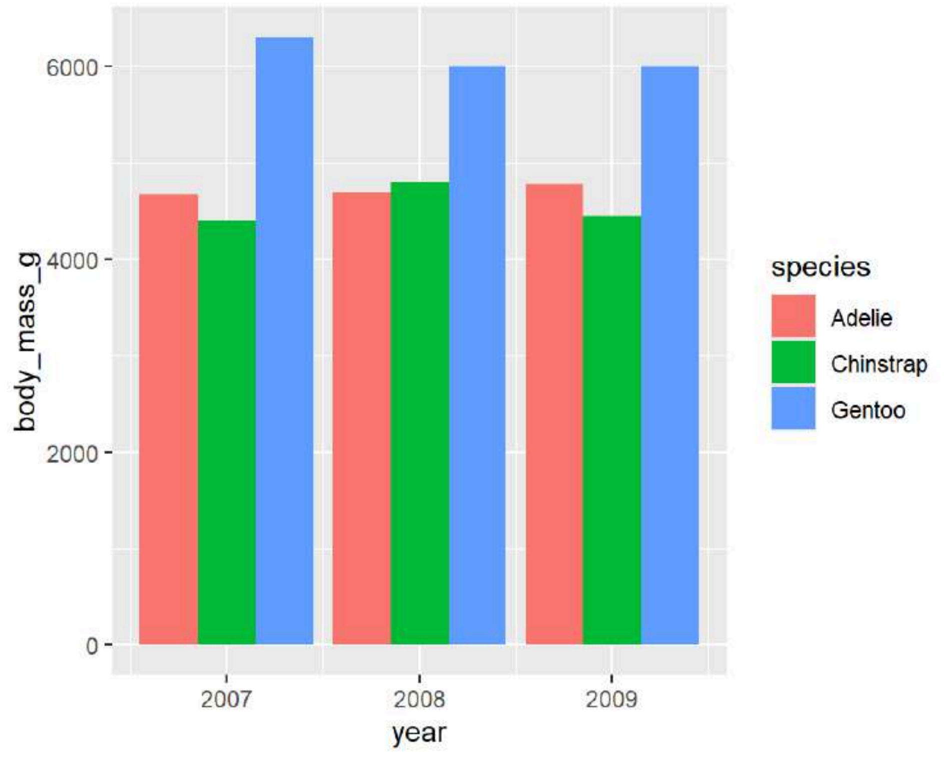
```r
p <- ggplot(penguins,aes(x=year,y=body_mass_g,fill=species))
```

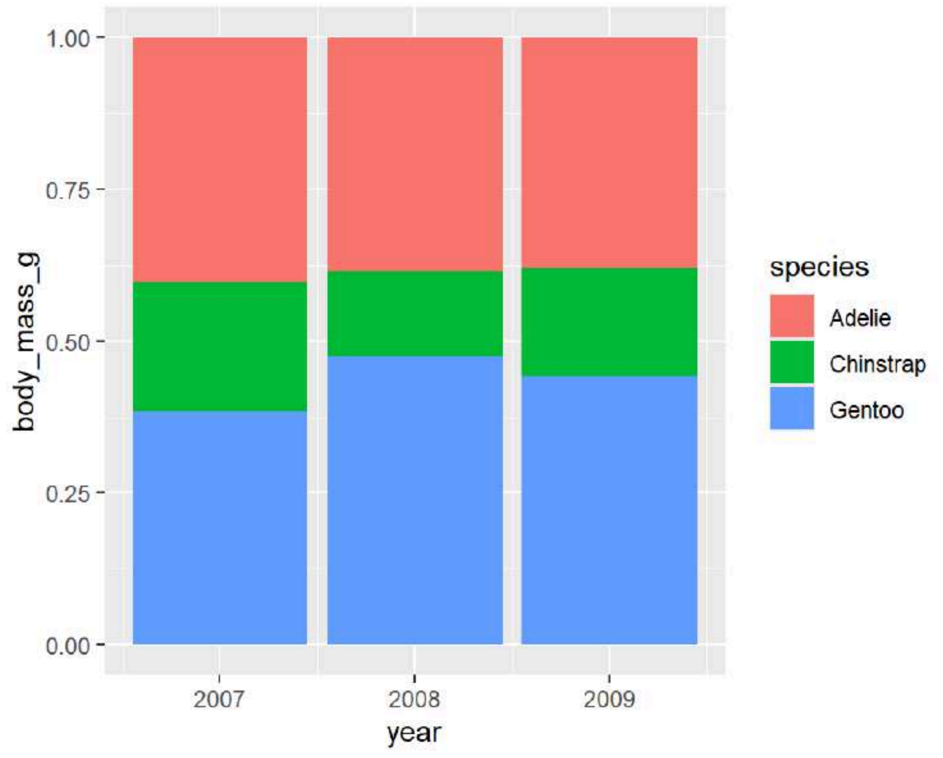
```r
p + geom_bar(stat="identity",
             position="stack")
```

```r
p + geom_bar(stat="identity",
             position="dodge")
```

```r
p + geom_bar(stat="identity",
             position="fill")
```



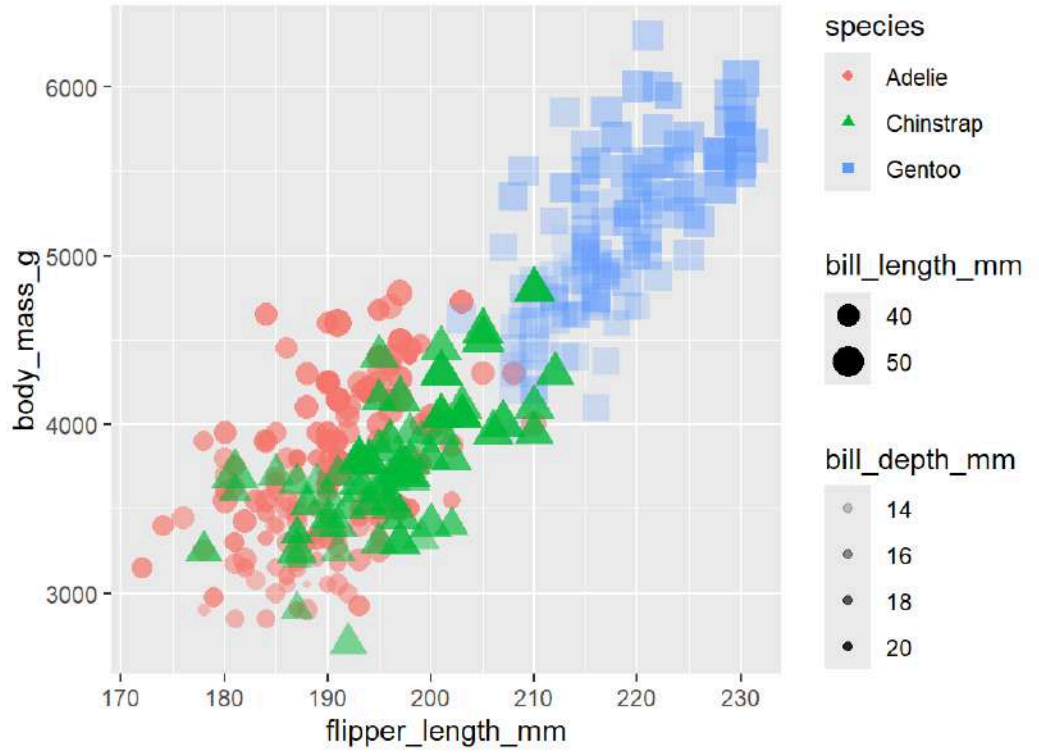




Aesthetics

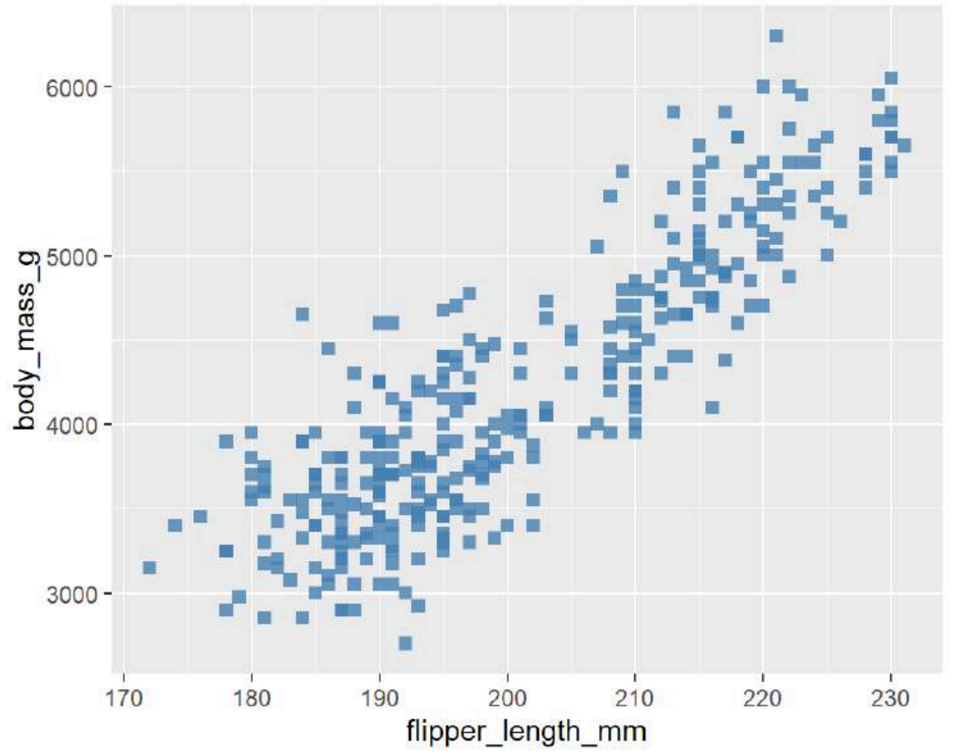
- Aesthetic `mapping`

```
ggplot(data = penguins)+
  geom_point(aes(x=flipper_length_mm,
                 y=body_mass_g,
                 size=bill_length_mm,
                 alpha=bill_depth_mm,
                 shape=species,
                 color=species))
```



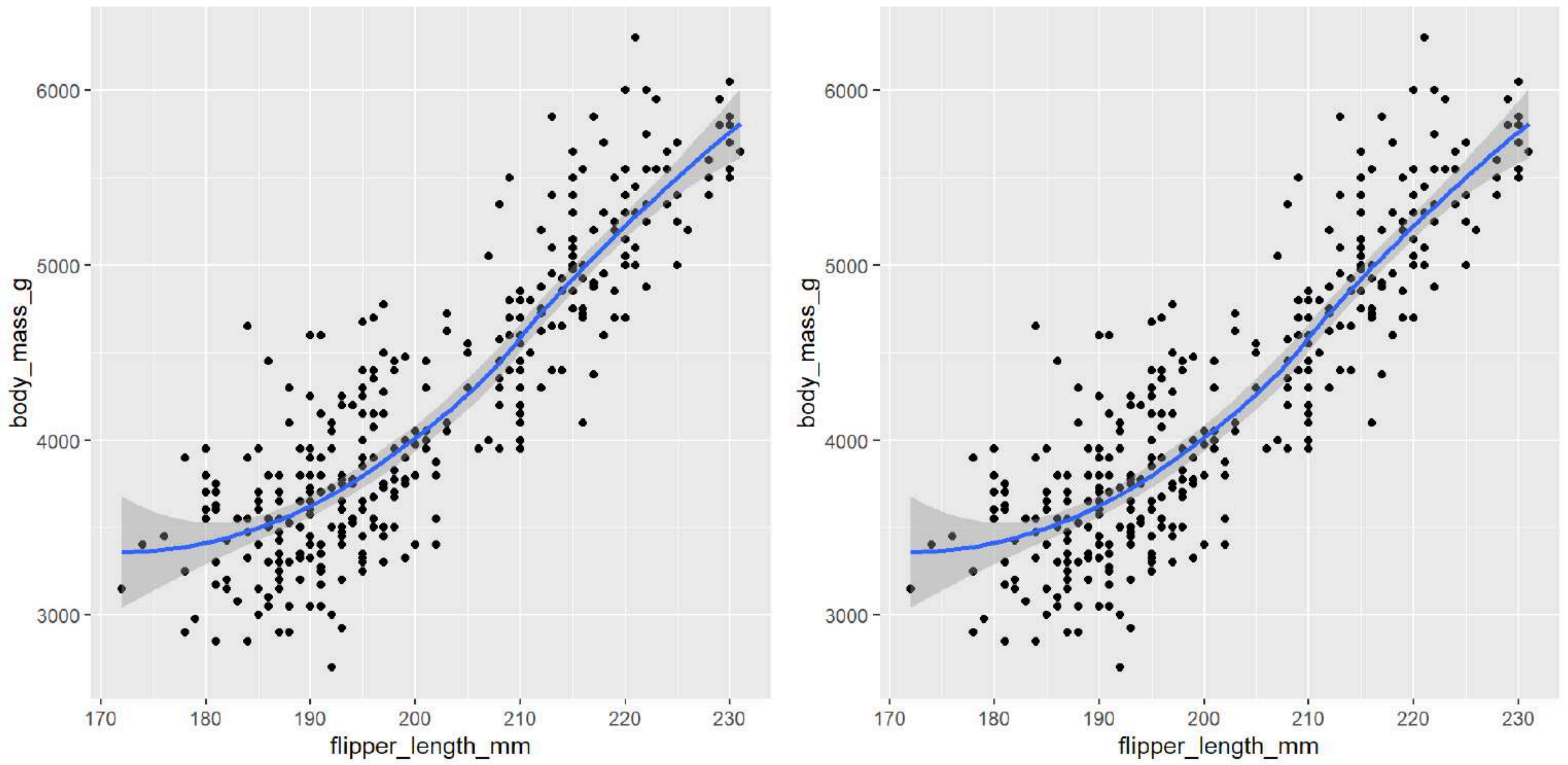
- Aesthetic `parameter`

```
ggplot(data = penguins)+
  geom_point(aes(x=flipper_length_mm,
                 y=body_mass_g),
                 size=2,
                 alpha=0.8,
                 shape=15,
                 color="steelblue")
```



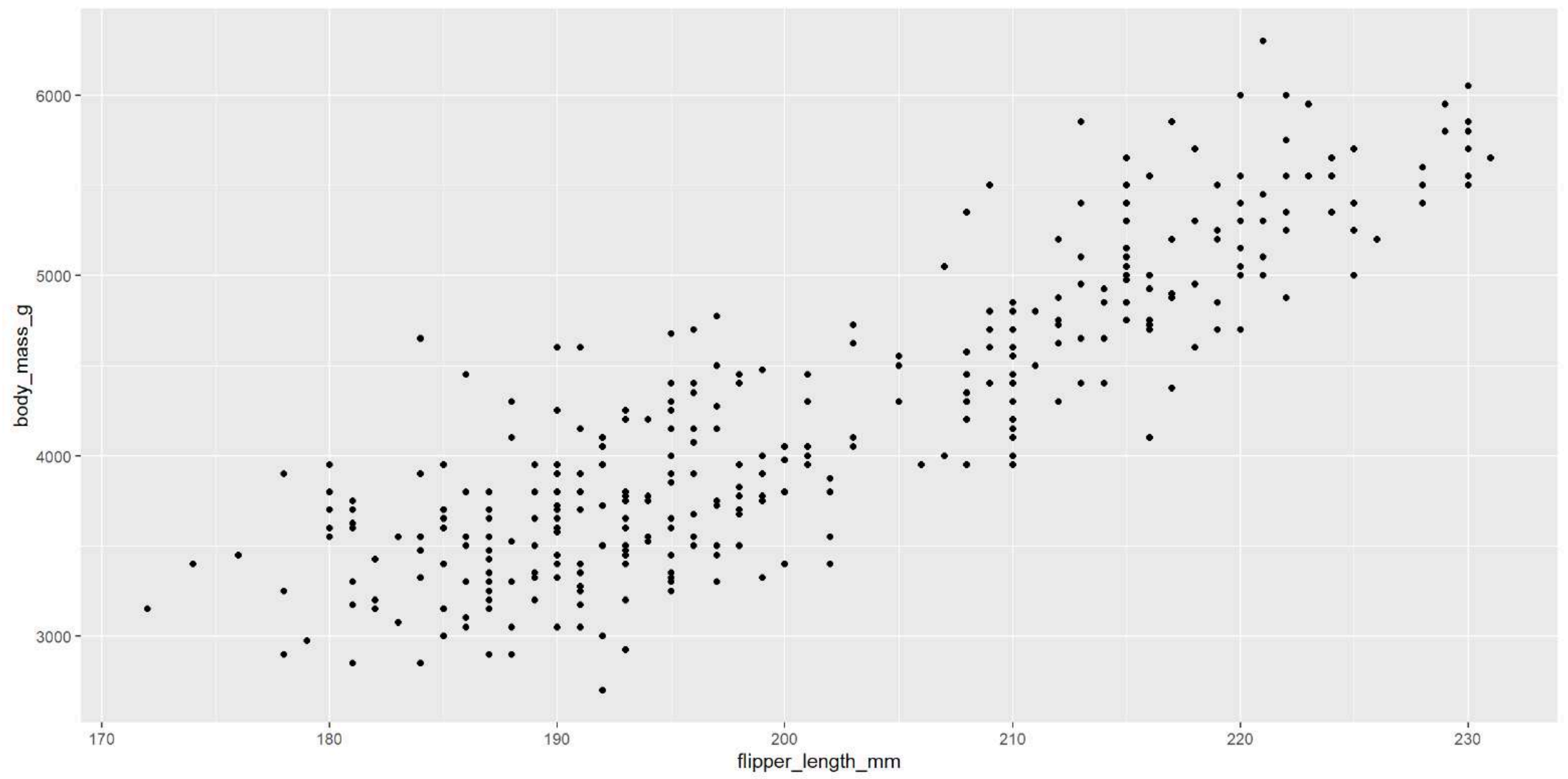
Aesthetics

```
x1 <- ggplot(penguins) +
  geom_point(aes(x=flipper_length_mm,  y=body_mass_g))+
  stat_smooth(aes(x=flipper_length_mm, y=body_mass_g))

x2 <- ggplot(penguins, aes(x=flipper_length_mm, y=body_mass_g))+
  geom_point() +
  geom_smooth()

wrap_plots(x1,x2,nrow=1,ncol=2)
```



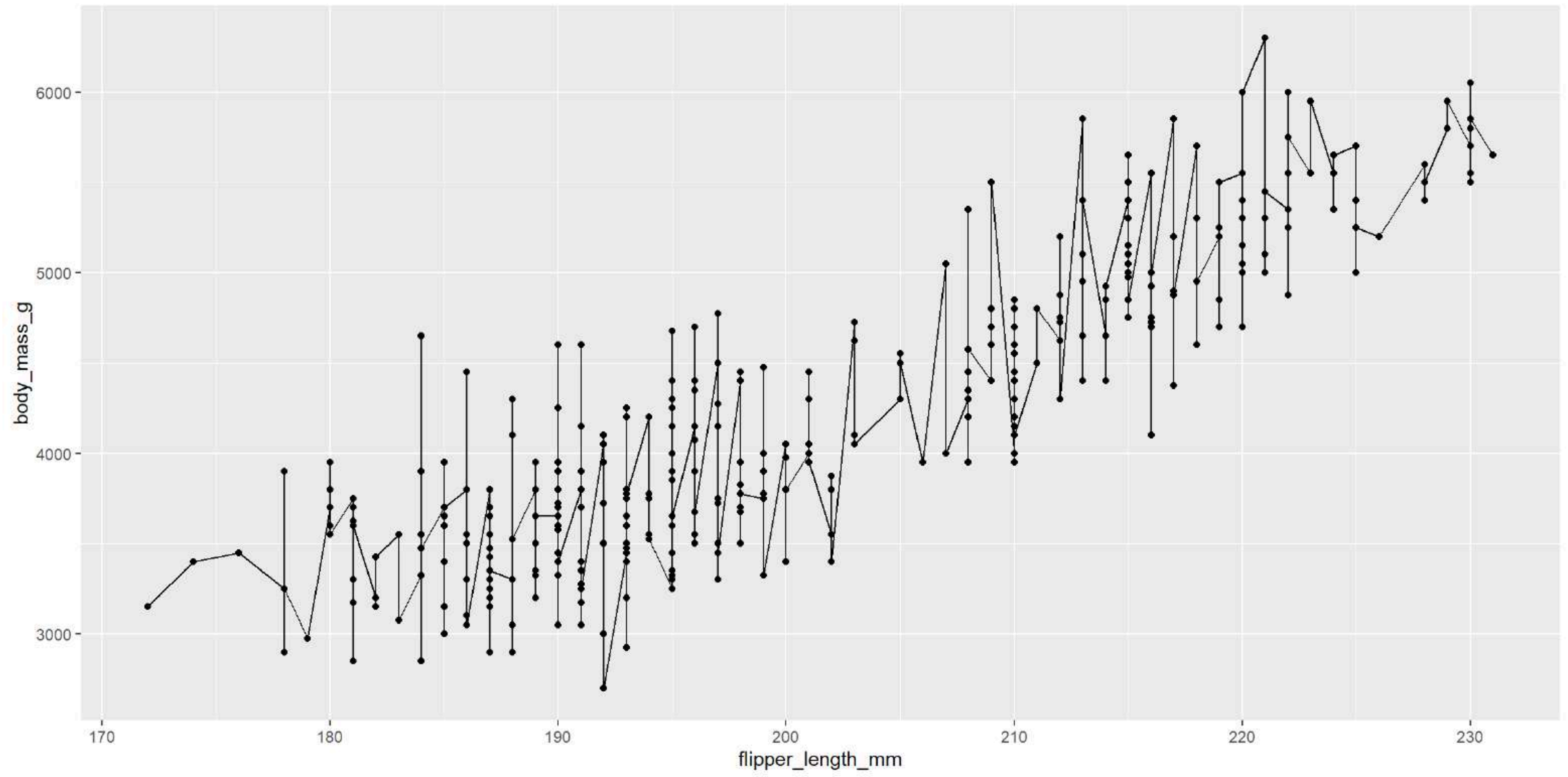
Multiple Geoms

```r
p <- ggplot(penguins,aes(x=flipper_length_mm,y=body_mass_g))+
      geom_point()






p
```



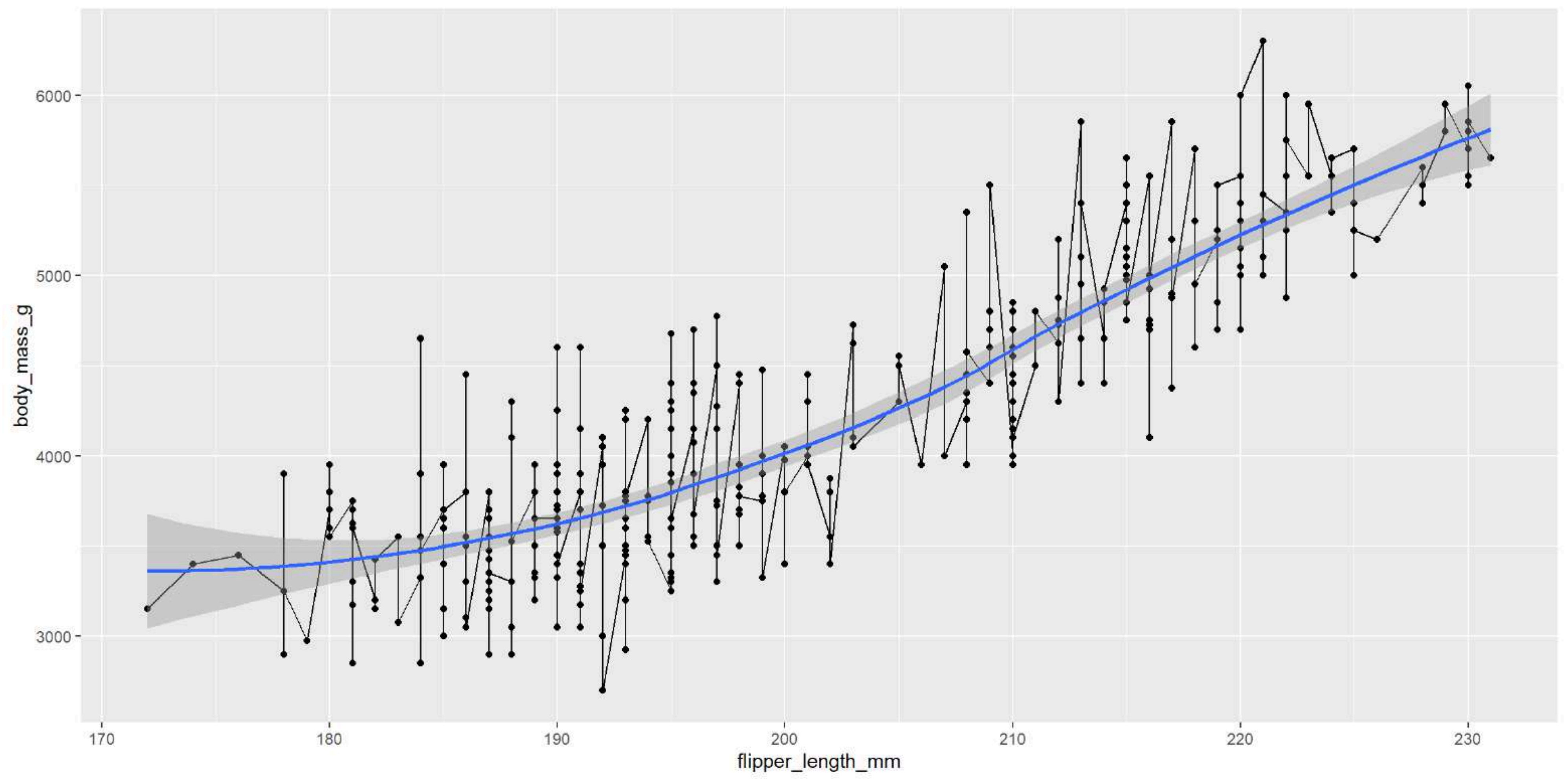
Multiple Geoms

```r
p <- ggplot(penguins,aes(x=flipper_length_mm,y=body_mass_g))+
      geom_point()+
      geom_line()





p
```



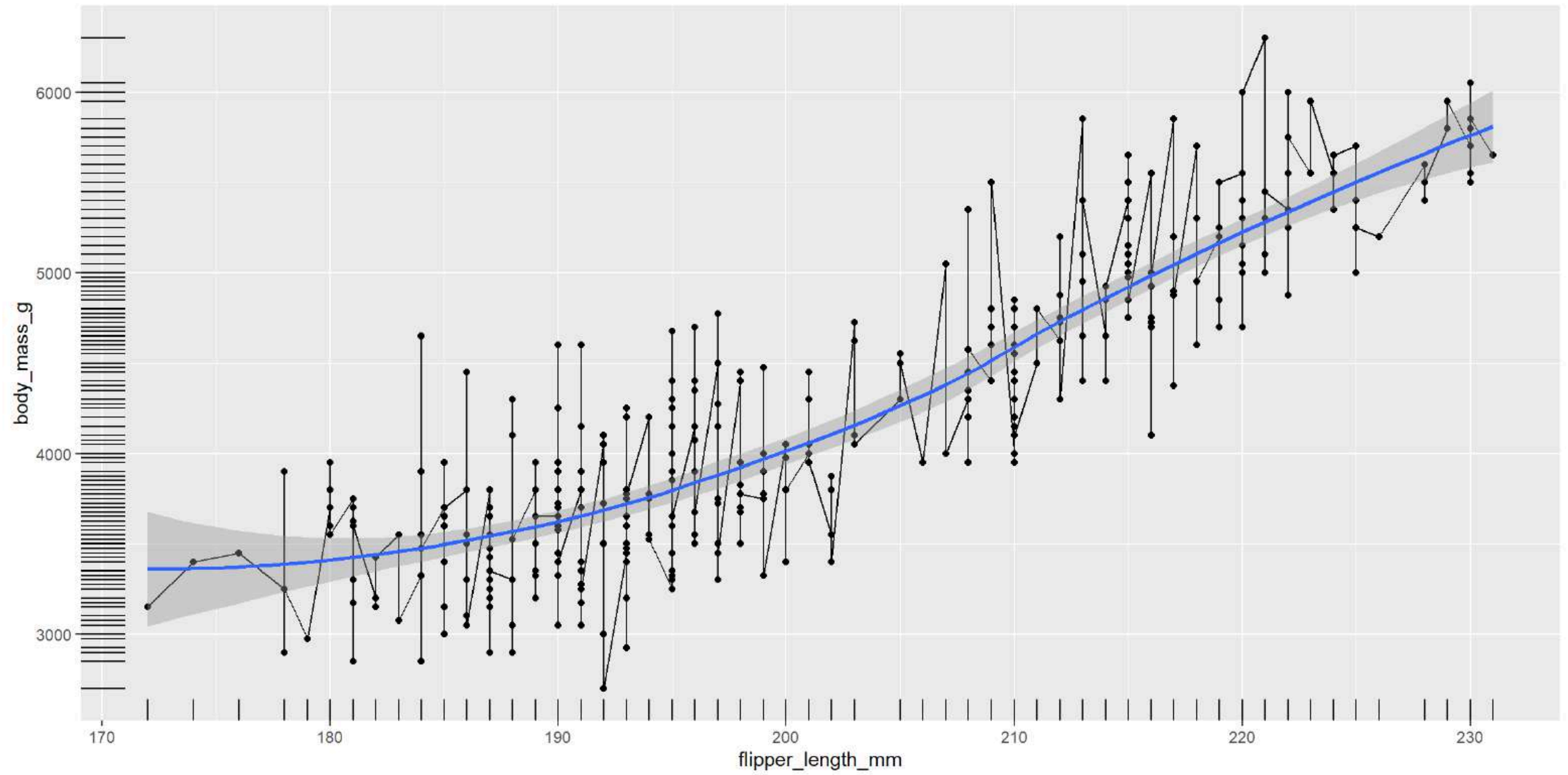
Multiple Geoms

```
p <- ggplot(penguins,aes(x=flipper_length_mm,y=body_mass_g))+
      geom_point()+
      geom_line()+
      geom_smooth()




p
```



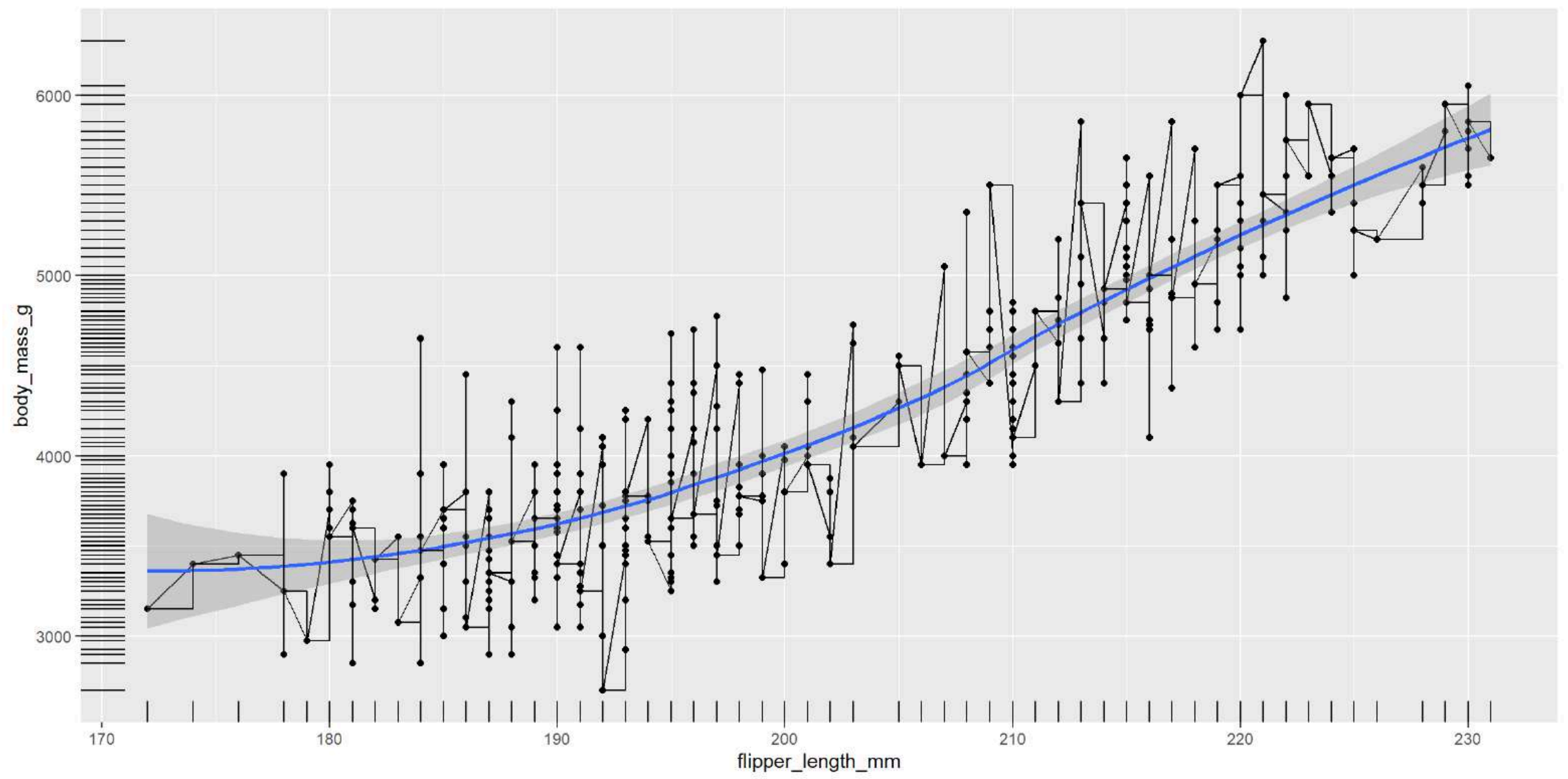
Multiple Geoms

```
p <- ggplot(penguins,aes(x=flipper_length_mm,y=body_mass_g))+
      geom_point()+
      geom_line()+
      geom_smooth()+
      geom_rug()



p
```



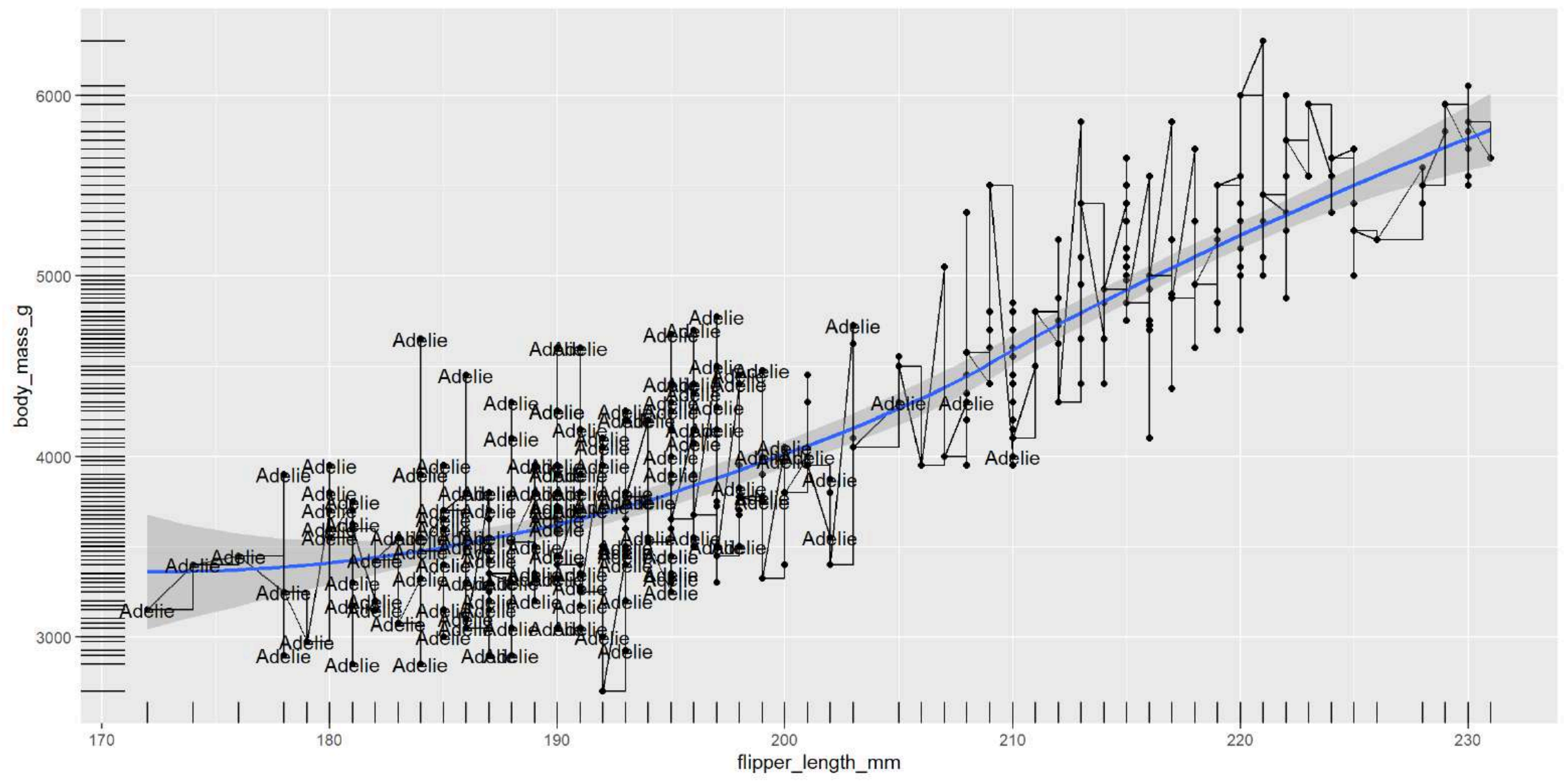
Multiple Geoms

```r
p <- ggplot(penguins,aes(x=flipper_length_mm,y=body_mass_g))+
      geom_point()+
      geom_line()+
      geom_smooth()+
      geom_rug()+
      geom_step()


p
```

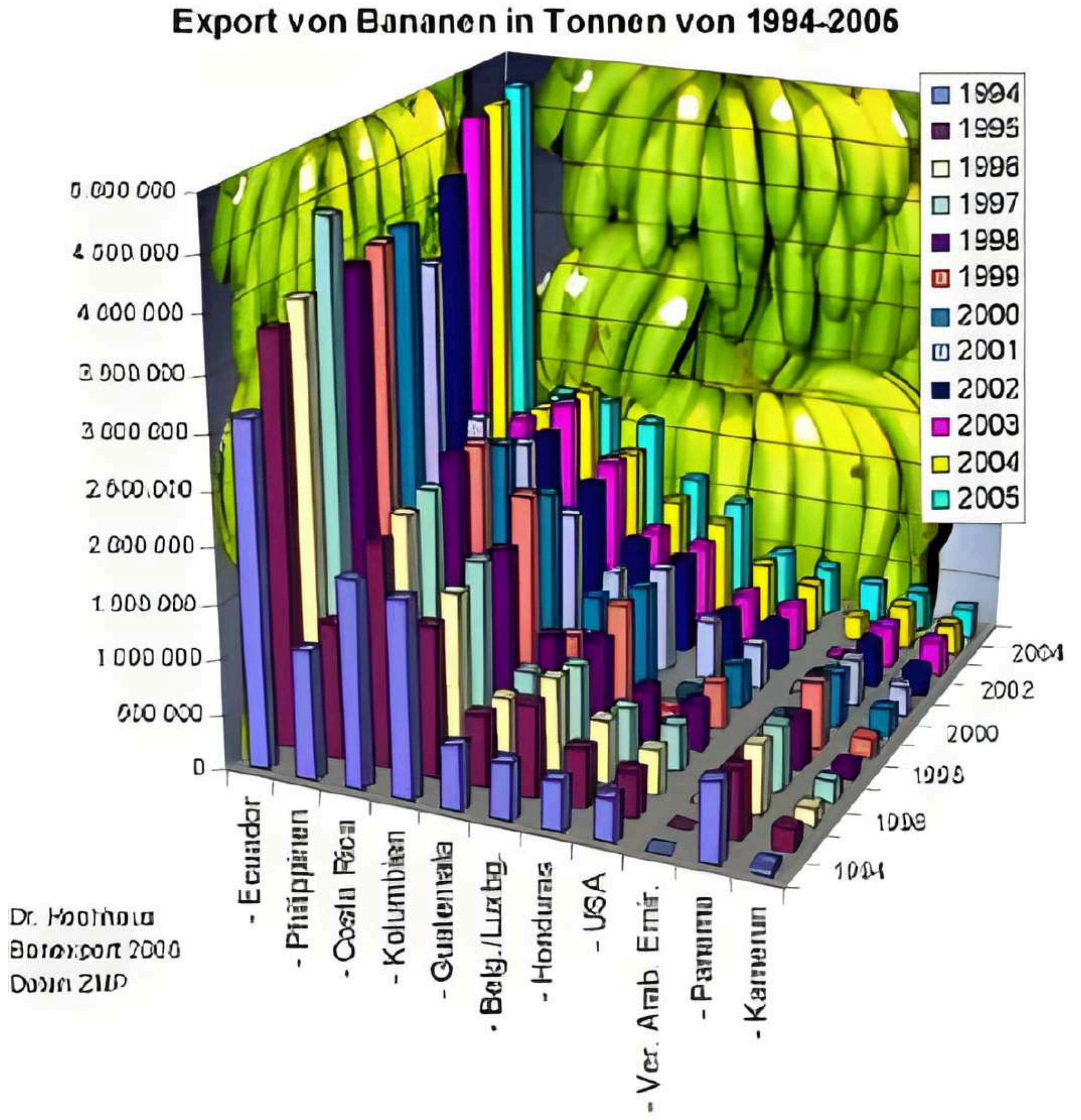


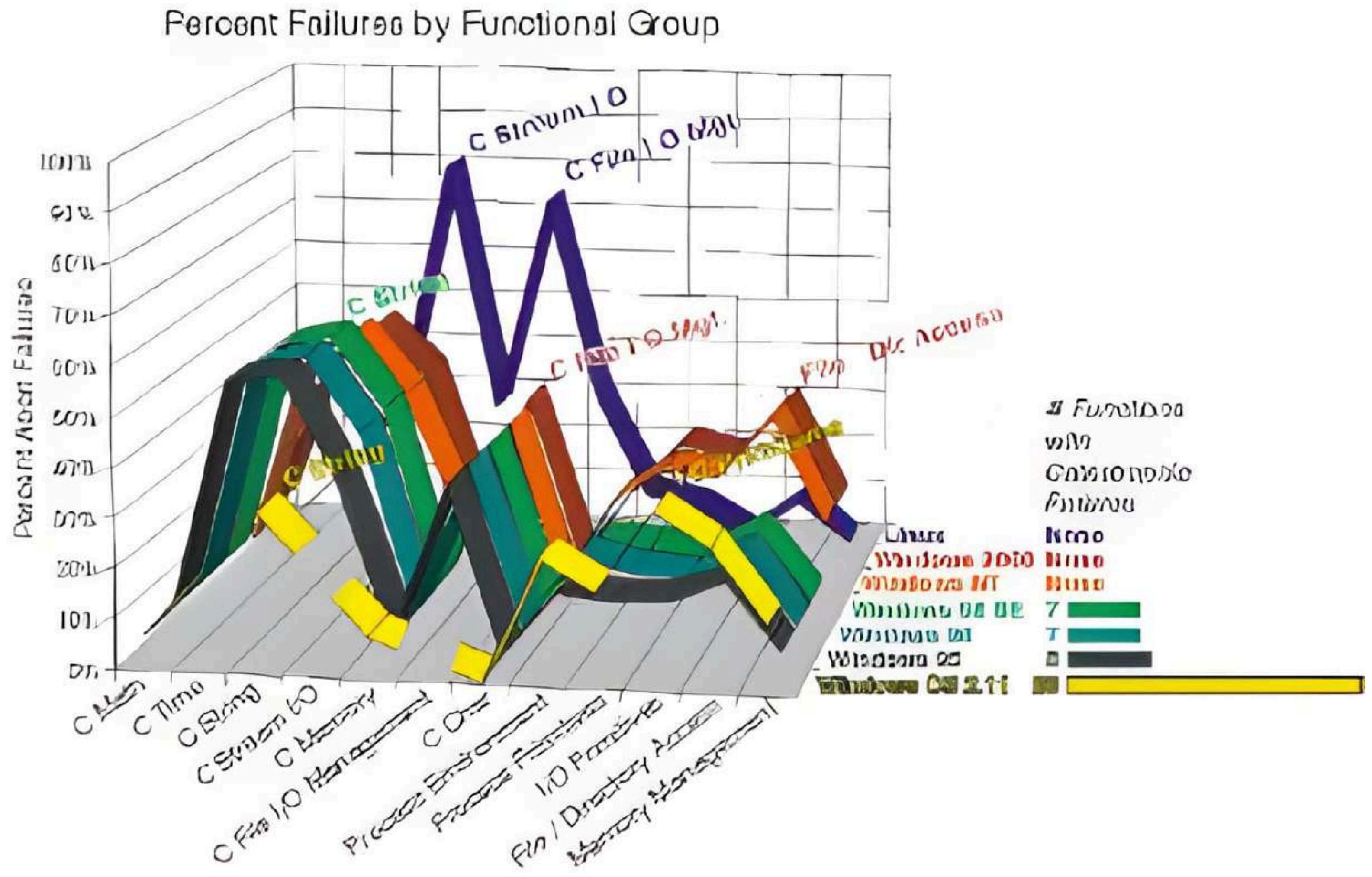
Multiple Geoms

```r
p <- ggplot(penguins,aes(x=flipper_length_mm,y=body_mass_g))+
      geom_point()+
      geom_line()+
      geom_smooth()+
      geom_rug()+
      geom_step()+
      geom_text(data=subset(penguins,penguins$species=="Adelie"),
                aes(label=species))
p
```



Just because you can doesn’t mean you should!

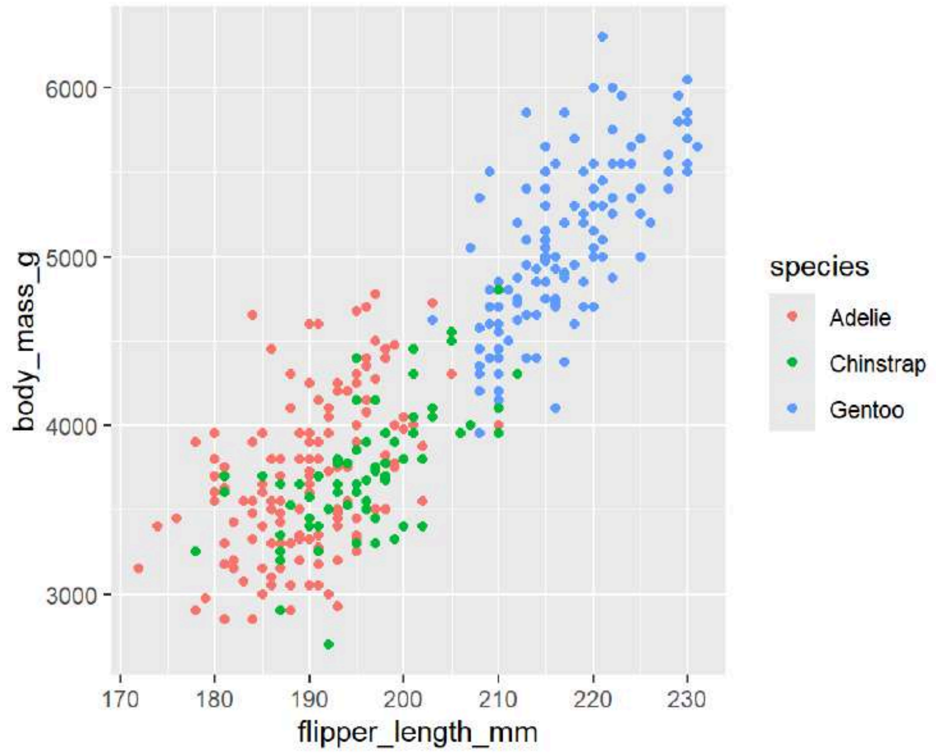




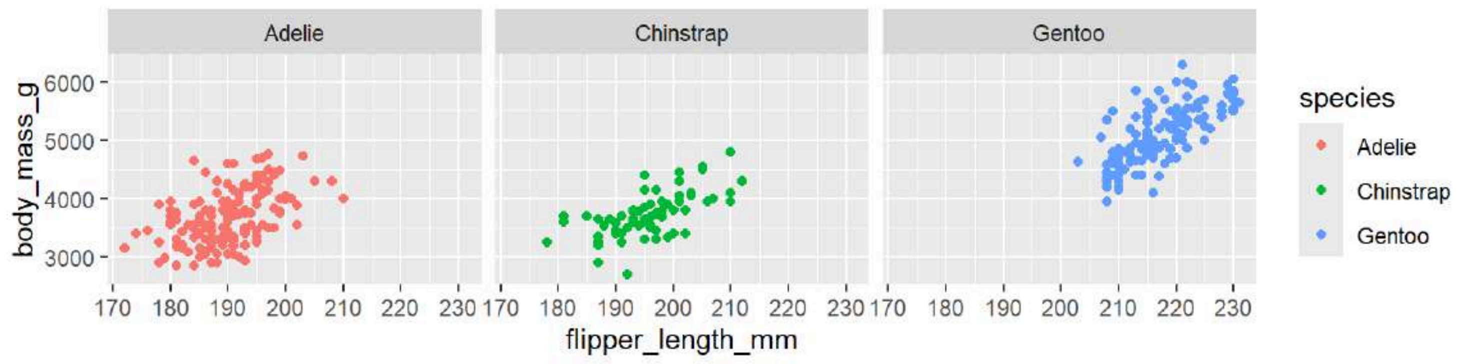
Facets · facet_wrap

- Split to subplots based on variable(s),
- Faceting in one dimension

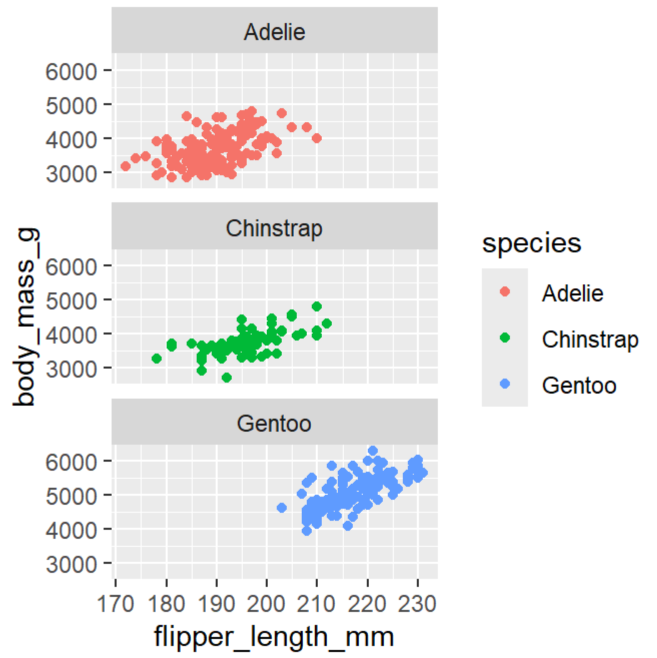
```r
p <- ggplot(penguins)+
      geom_point(aes(x=flipper_length_mm,
                     y=body_mass_g,
                     color=species))
p
```



```r
p + facet_wrap(~species)
```



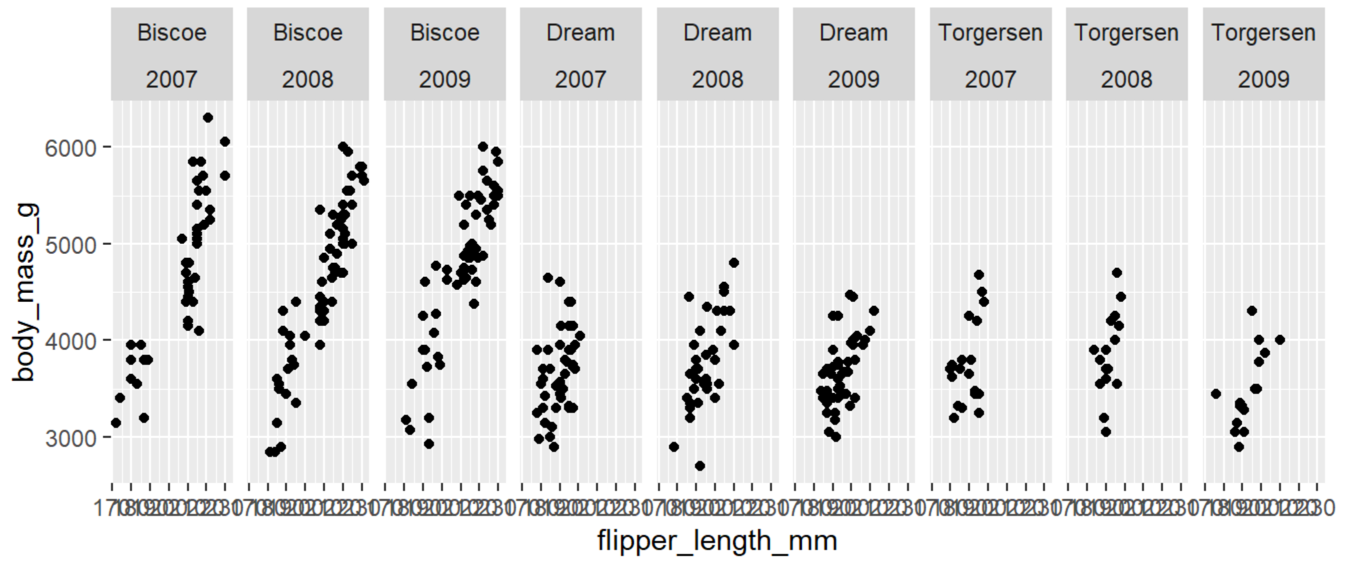
```r
p + facet_wrap(~species,nrow=3)
```



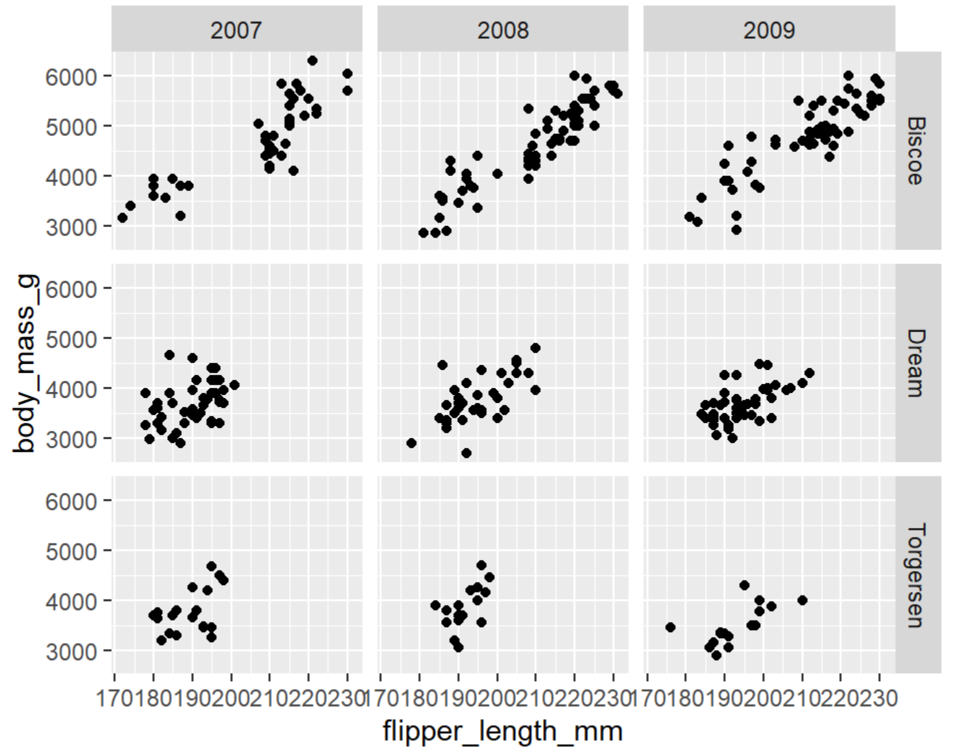
Facets · facet_grid

- Faceting in two dimensions

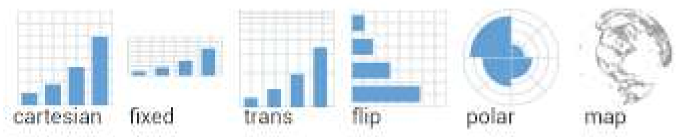
```r
p <- ggplot(data = penguins, aes(x=flipper_length_mm,
                                 y=body_mass_g))+
                                 geom_point()
p + facet_grid(~island+year)
```



```r
p + facet_grid(island~year)
```



Coordinate Systems



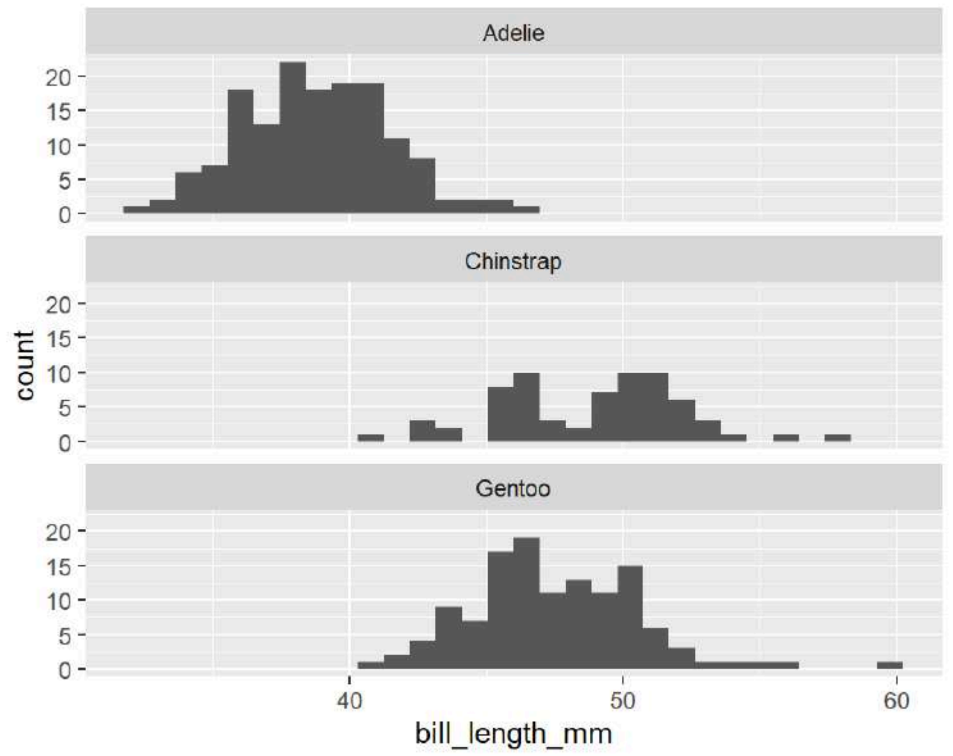
- `coord_cartesian(xlim=c(2,8))` for zooming in
- `coord_map` for controlling limits on maps
- `coord_polar` for polar cordinates

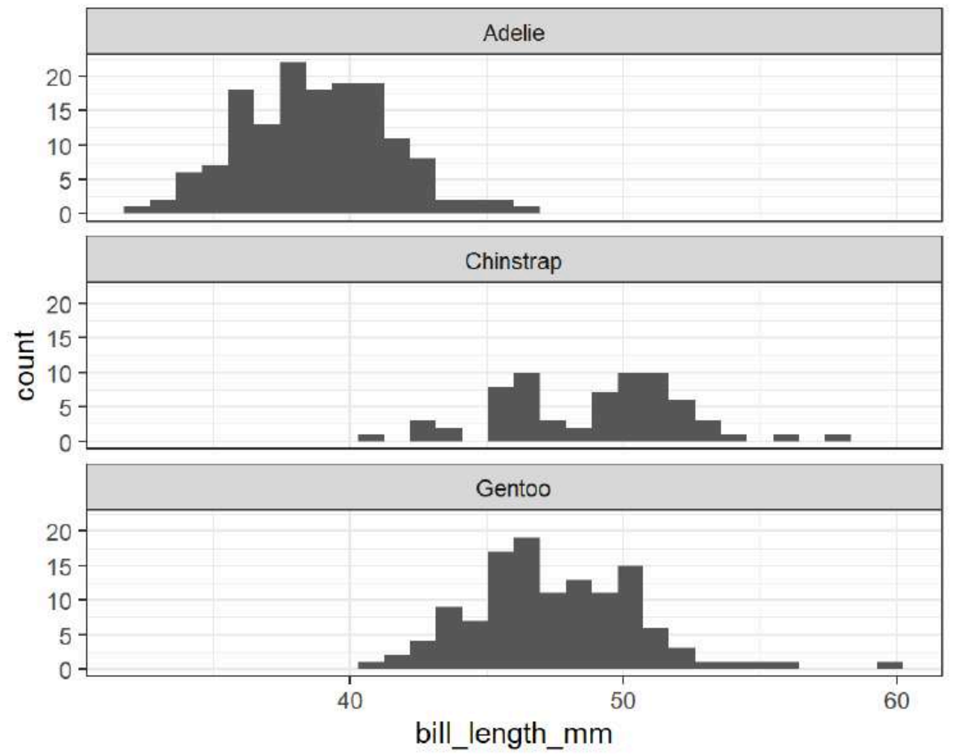
Theming

- Modify non-data plot elements/appearance
- Axis labels, panel colors, legend appearance etc
- Save a particular appearance for reuse
- `?theme`

```r
ggplot(penguins, aes(x=bill_length_mm)) +
    geom_histogram() +
    facet_wrap(~species, ncol = 1) +
    theme_grey()
```

```r
ggplot(penguins, aes(x=bill_length_mm)) +
    geom_histogram() +
    facet_wrap(~species, ncol = 1) +
    theme_bw()
```



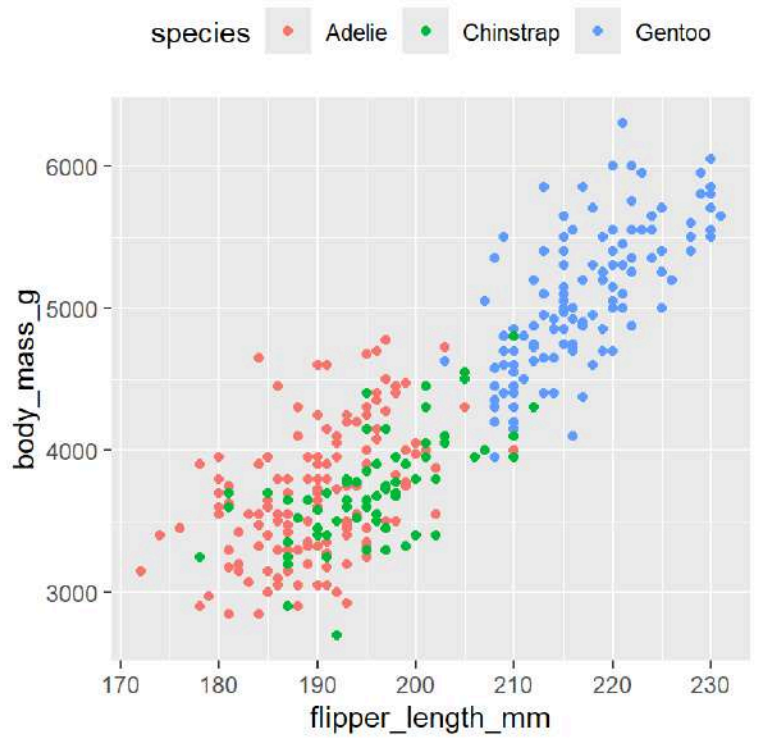


Theme · Legend

```r
p <- ggplot(penguins)+
      geom_point(aes(x=flipper_length_mm,
                     y=body_mass_g,
                     color=species))
```

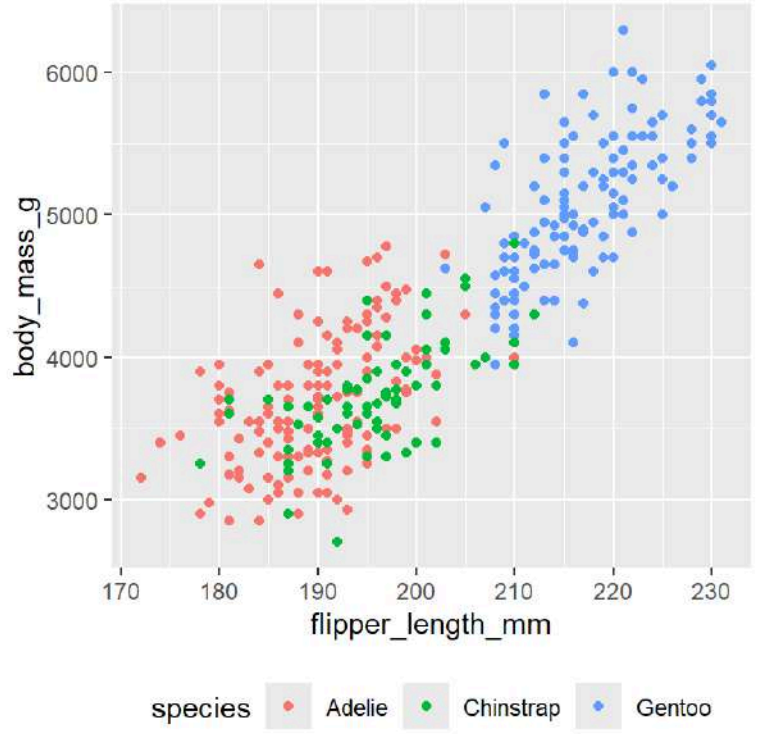
at top

```r
p + theme(legend.position="top")
```



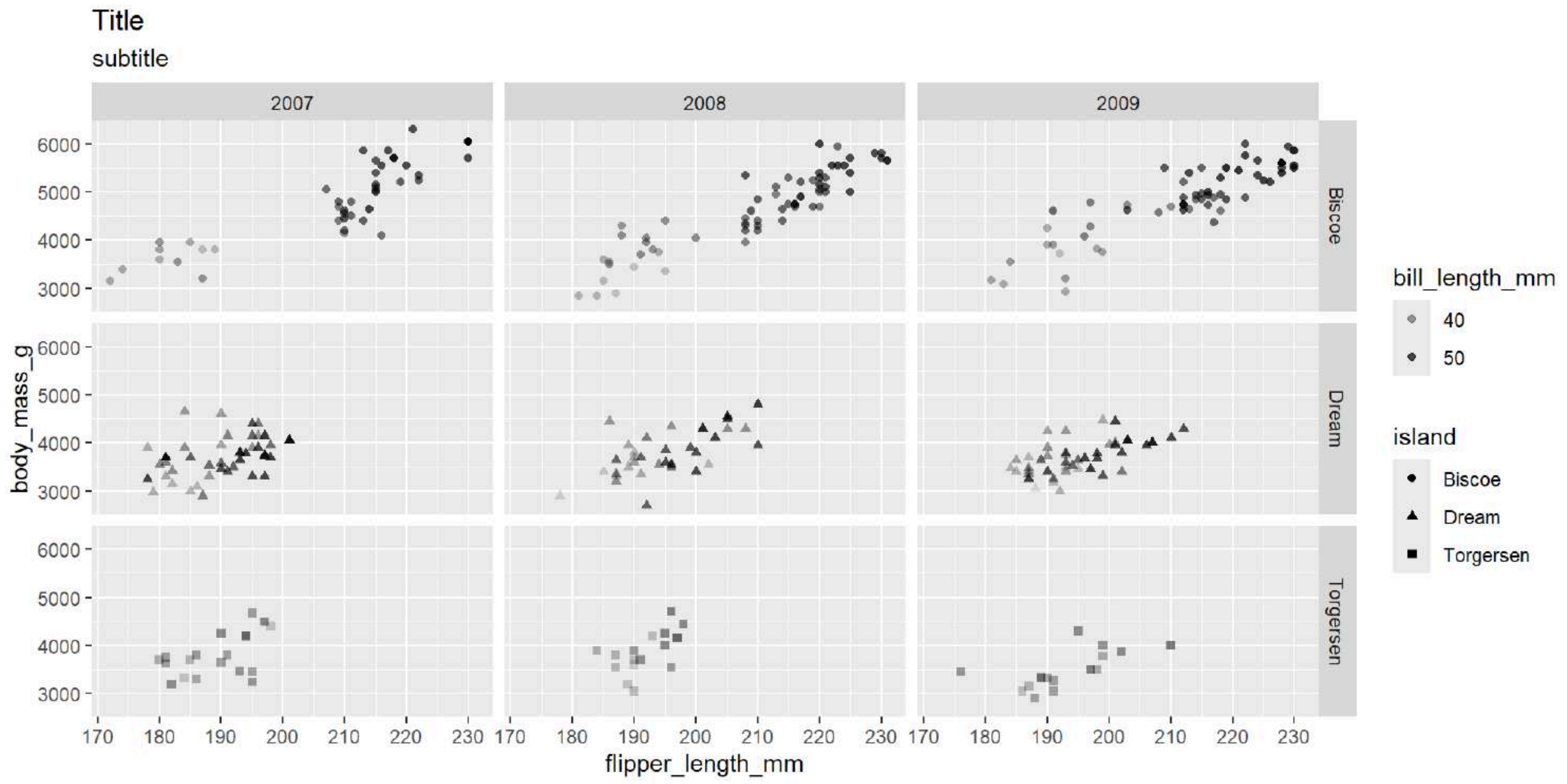
at bottom

```r
p + theme(legend.position="bottom")
```



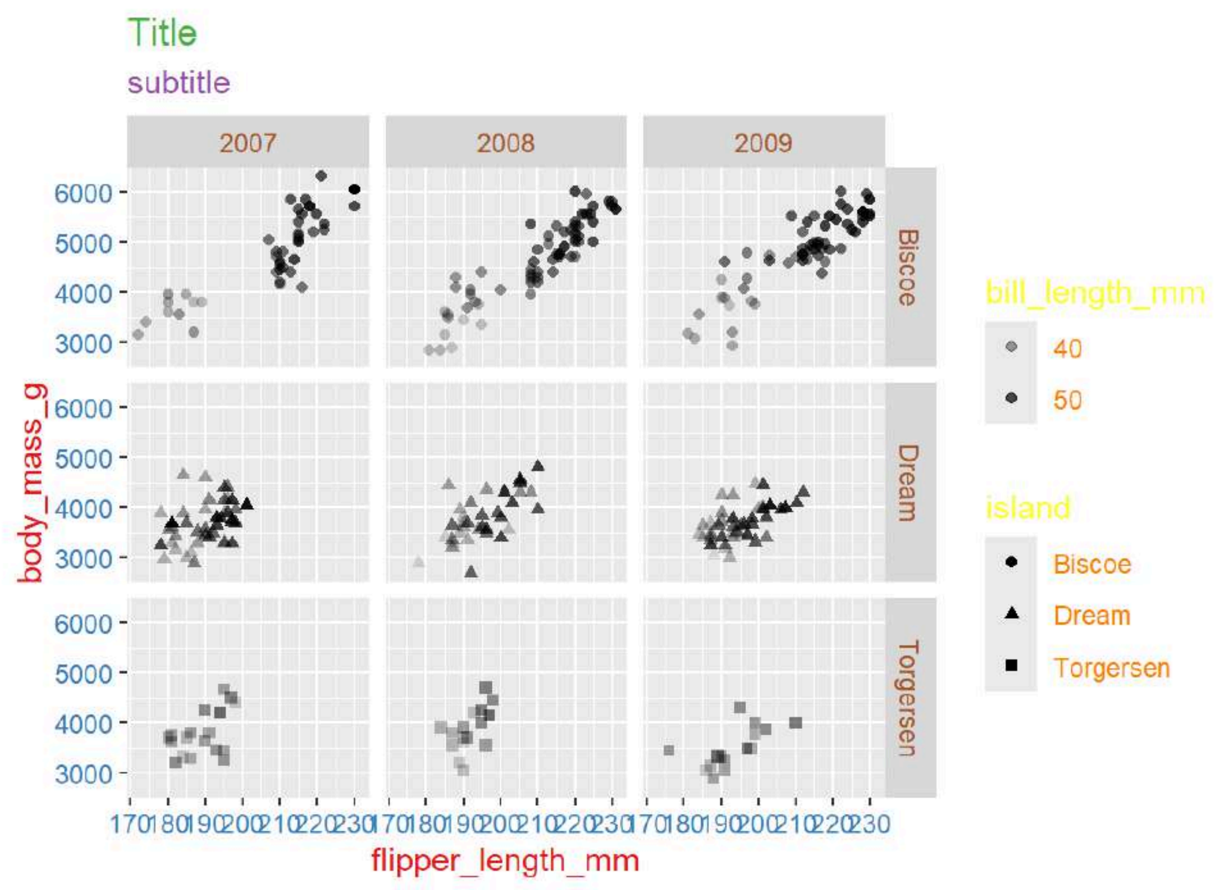
Theme · Text

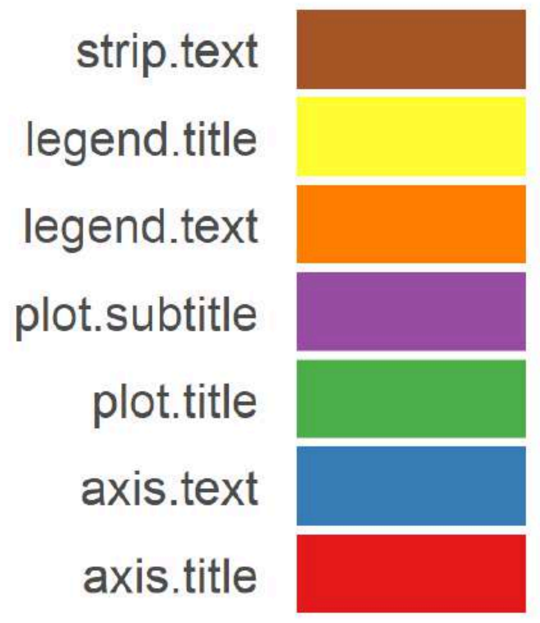
```r
p <- ggplot(penguins,
            aes(x = flipper_length_mm,
                y = body_mass_g,
                alpha = bill_length_mm,
                shape = island)) +
            geom_point() +
            facet_grid(island~year) +
            labs(title="Title",
            subtitle="subtitle")
p
```



Theme · Text

```
p <- p + theme(
    axis.title=element_text(color="#e41a1c"),
    axis.text=element_text(color="#377eb8"),
    plot.title=element_text(color="#4daf4a"),
    plot.subtitle=element_text(color="#984ea3"),
    legend.text=element_text(color="#ff7f00"),
    legend.title=element_text(color="#ffff33"),
    strip.text=element_text(color="#a65628")
)
```

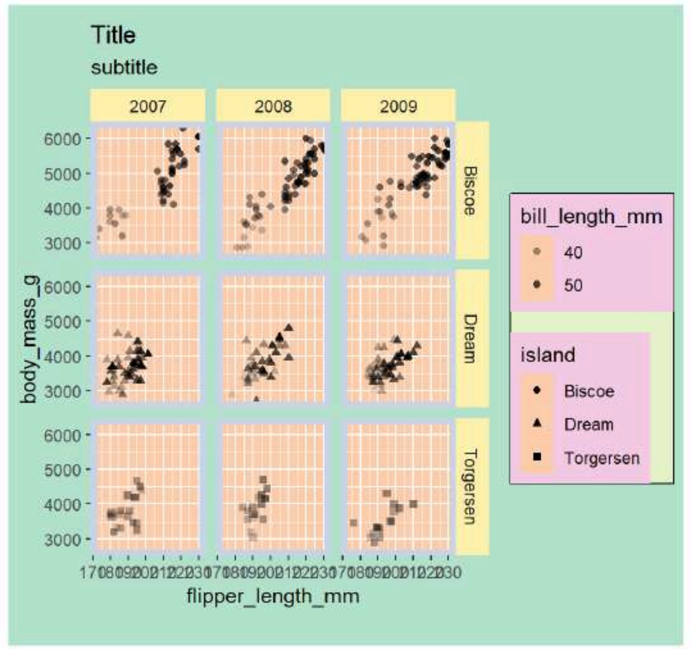




Theme · Rect

```
List of 5
 $ fill         : NULL
 $ colour       : NULL
 $ linewidth    : NULL
 $ linetype     : NULL
 $ inherit.blank: logi FALSE
 - attr(*, "class")= chr [1:2] "element_rect" "element"
```

```r
p <- p + theme(
    plot.background=element_rect(fill="#b3e2cd"),
    panel.background=element_rect(fill="#fdcdac"),
    panel.border=element_rect(fill=NA,color="#cbd5e8",size=3),
    legend.background=element_rect(fill="#f4cae4"),
    legend.box.background=element_rect(fill="#e6f5c9"),
    strip.background=element_rect(fill="#fff2ae")
)
```

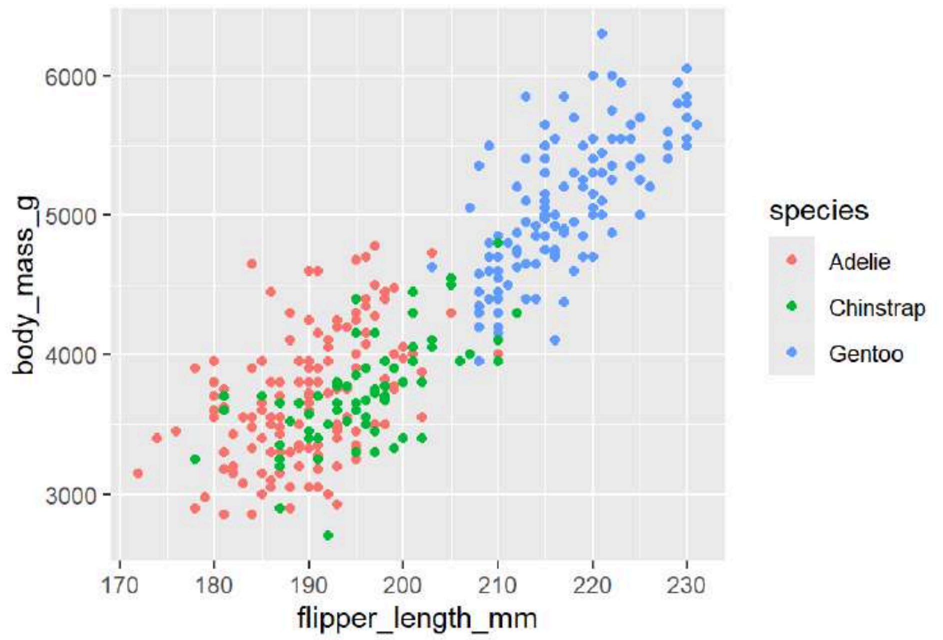


strip.background 🟨
legend.box.background 🟩
legend.background 🟪
panel.border 🟦
panel.background 🟧
plot.background 🟩

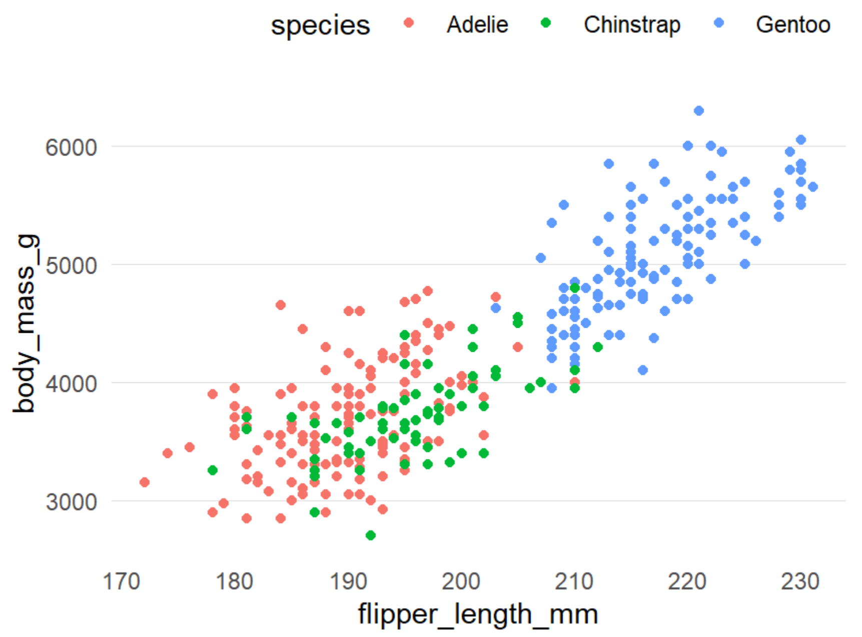
Theme · Reuse

```r
newtheme <- theme_bw() + theme(
  axis.ticks=element_blank(), panel.background=element_rect(fill="white"),
  panel.grid.minor=element_blank(), panel.grid.major.x=element_blank(),
  panel.grid.major.y=element_line(size=0.3,color="grey90"), panel.border=element_blank(),
  legend.position="top", legend.justification="right"
)
```

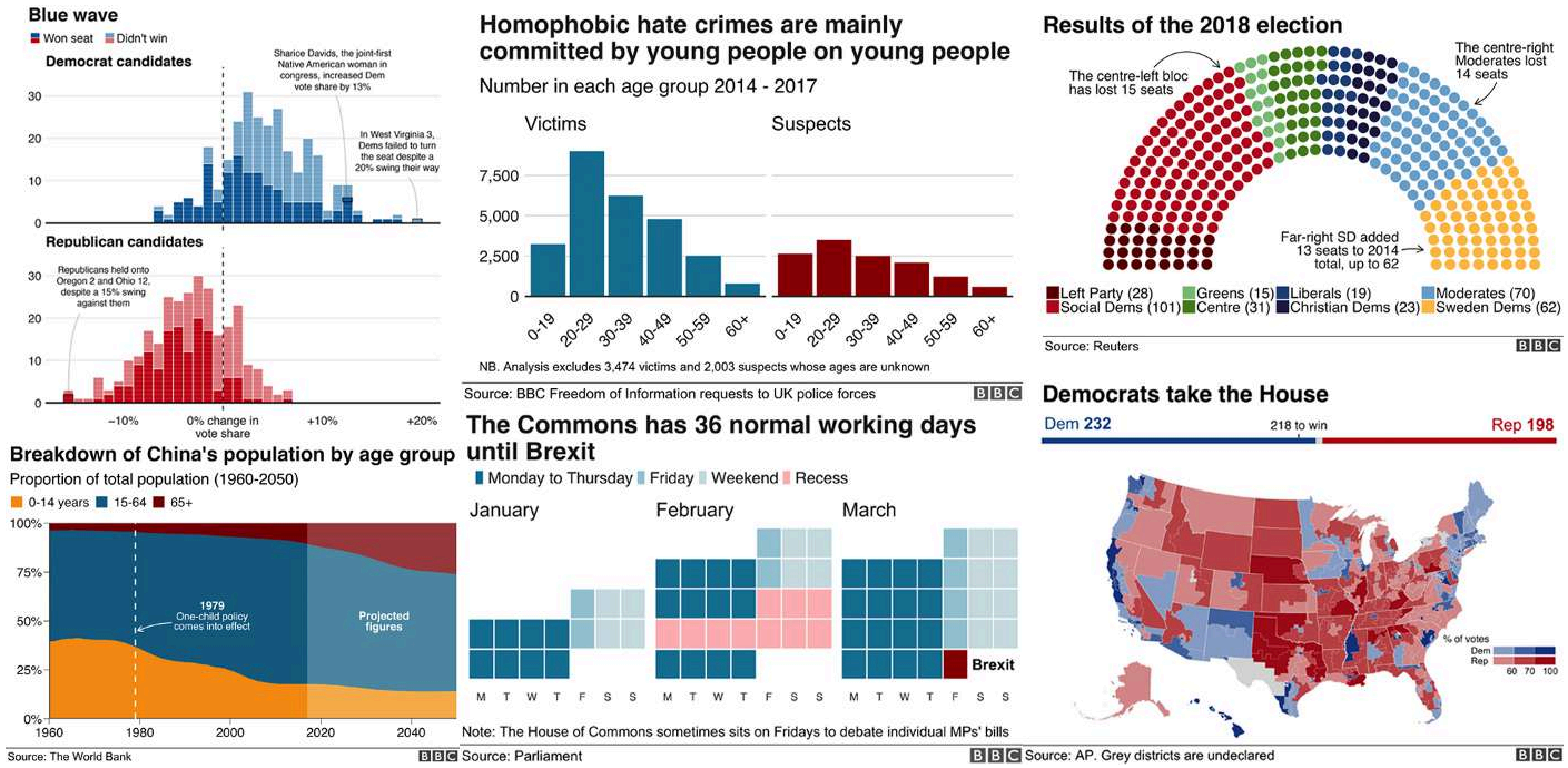
```r
p
```



```r
p + newtheme
```



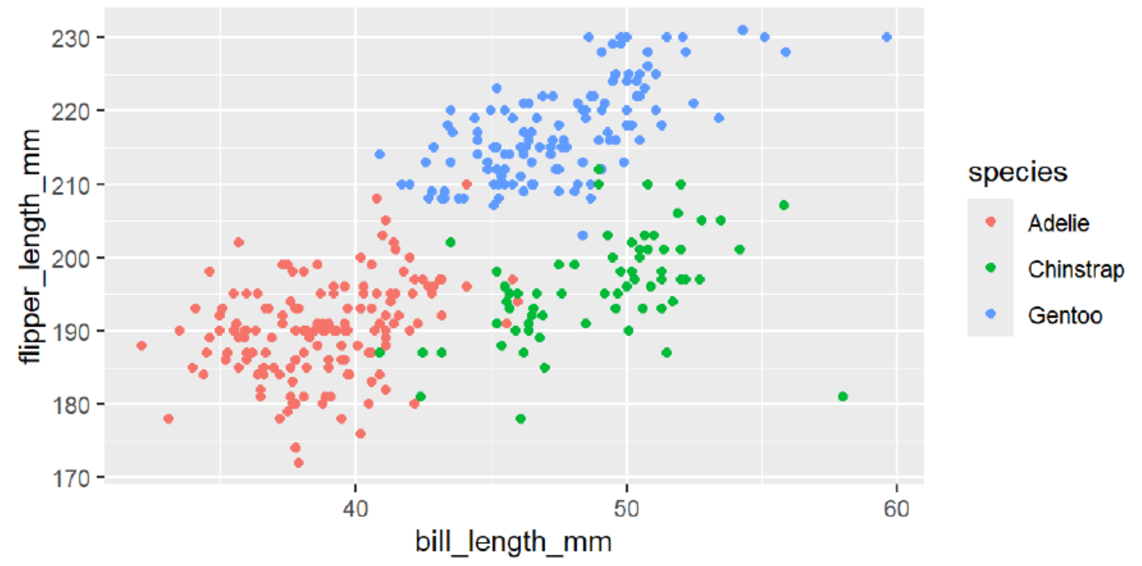
Professional themes



How BBC works with R graphics

Saving plots

```r
p <- ggplot(penguins,aes(x=bill_length_mm,y=flipper_length_mm,color=species)) +
  geom_point()
p
```



- ggplot2 package offers a convenient function

```r
ggsave("plot.png",p,height=5,width=7,units="cm",dpi=200)
# Note that default units in png is pixels while in ggsave it’s inches
```

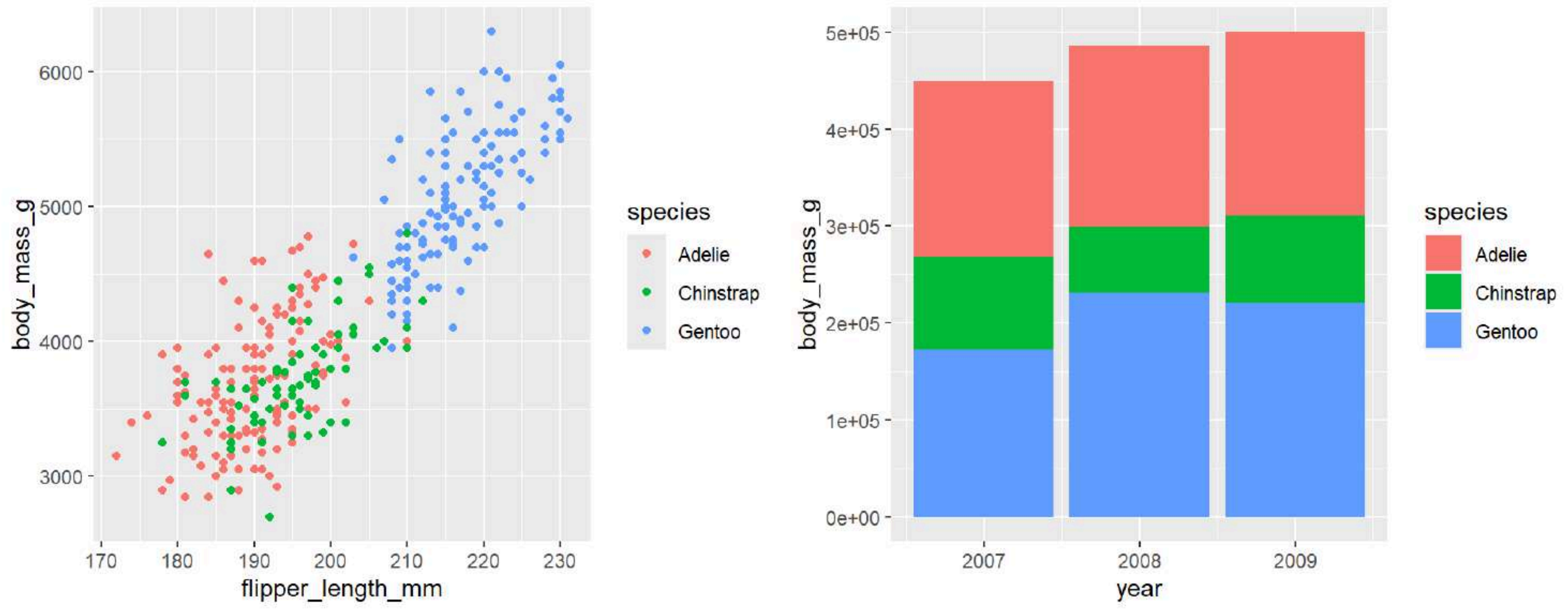
- ggplot2 plots can be saved just like base plots

```r
png("plot.png",height=5,width=7,units="cm",res=200)
print(p)
dev.off()
```

Combining Plots

```r
p <- ggplot(penguins, aes(x=flipper_length_mm, y=body_mass_g,color=species)) + geom_point()
q <- ggplot(penguins, aes(x=year, y=body_mass_g, fill=species)) + geom_bar(stat="identity")
```

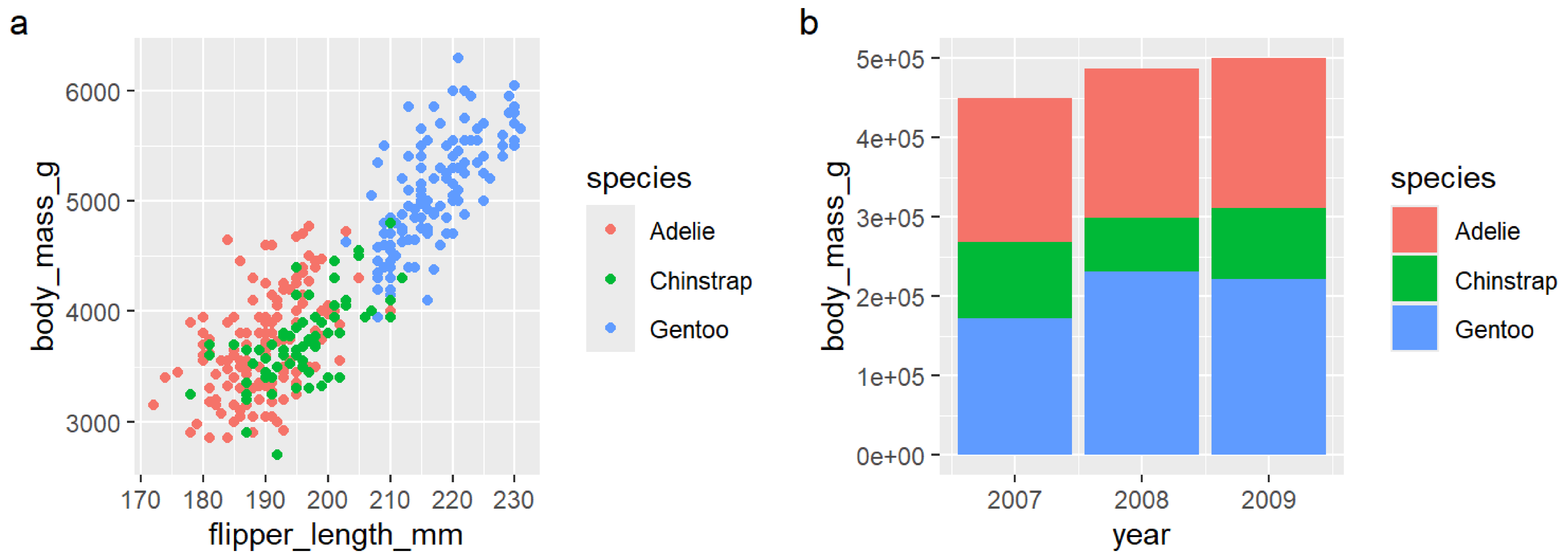
```r
patchwork::wrap_plots(p,q)
```



Combining Plots

```r
p <- ggplot(penguins, aes(x=flipper_length_mm, y=body_mass_g,color=species)) + geom_point()
q <- ggplot(penguins, aes(x=year, y=body_mass_g, fill=species)) + geom_bar(stat="identity")
```

```r
patchwork::wrap_plots(p,q) +
  plot_annotation(tag_levels = 'a')
```



patchwork documentation.

Extensions

- **patchwork**: Combining plots
- **ggrepel**: Text labels including overlap control
- **ggforce**: Circles, splines, hulls, voronoi etc
- **ggpmisc**: Miscellaneaous features
- **ggthemes**: Set of extra themes
- **ggthemr**: More themes
- **ggsci**: Color palettes for scales
- **ggmap**: Dedicated to mapping
- **ggraph**: Network graphs
- **ggiraph**: Converting ggplot2 to interactive graphics

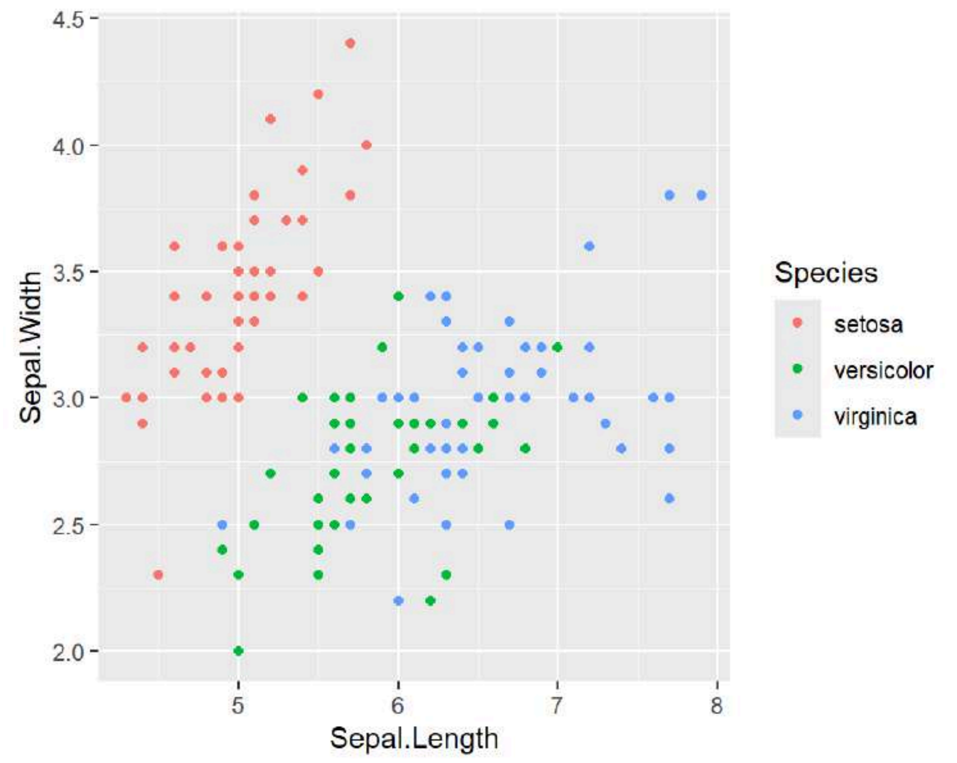
A collection of ggplot extension packages: https://exts.ggplot2.tidyverse.org/.
Curated list of ggplot2 links: https://github.com/erikgahner/awesome-ggplot2.

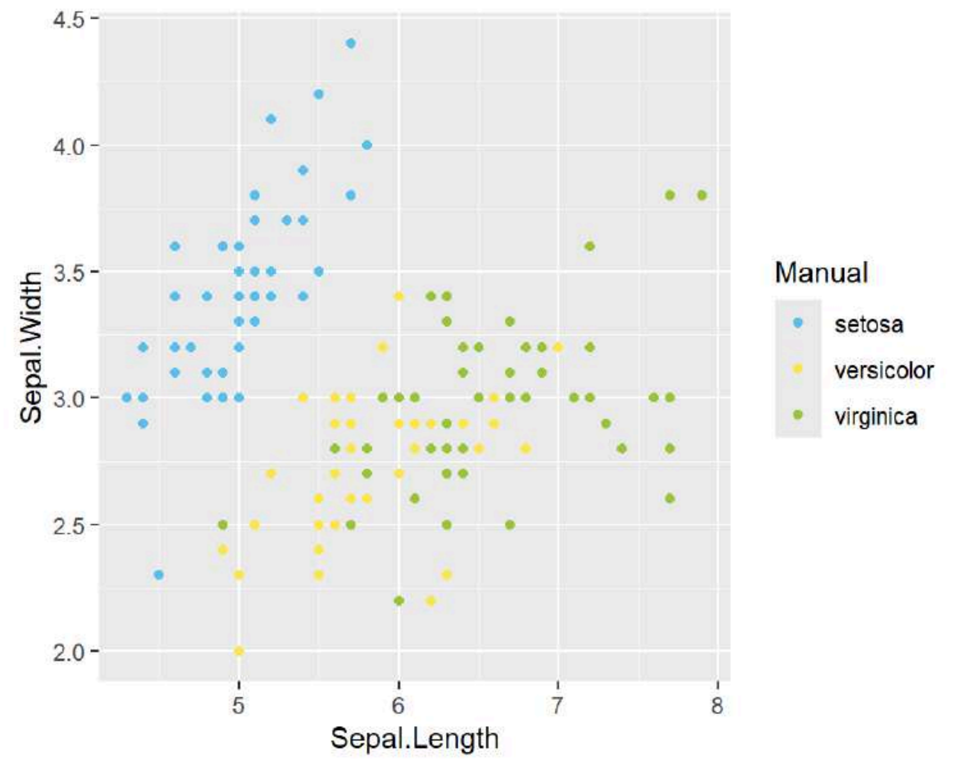
Scales · `Discrete Colors`

- scales: position, color, fill, size, shape, alpha, linetype
- syntax: `scale_<aesthetic>_<type>`

```r
p <- ggplot(iris)+geom_point(aes(x=Sepal.Length,
                                 y=Sepal.Width,color=Species))
p
```

```r
p + scale_color_manual(
      name="Manual",
      values=c("#5BC0EB","#FDE74C","#9BC53D"))
```



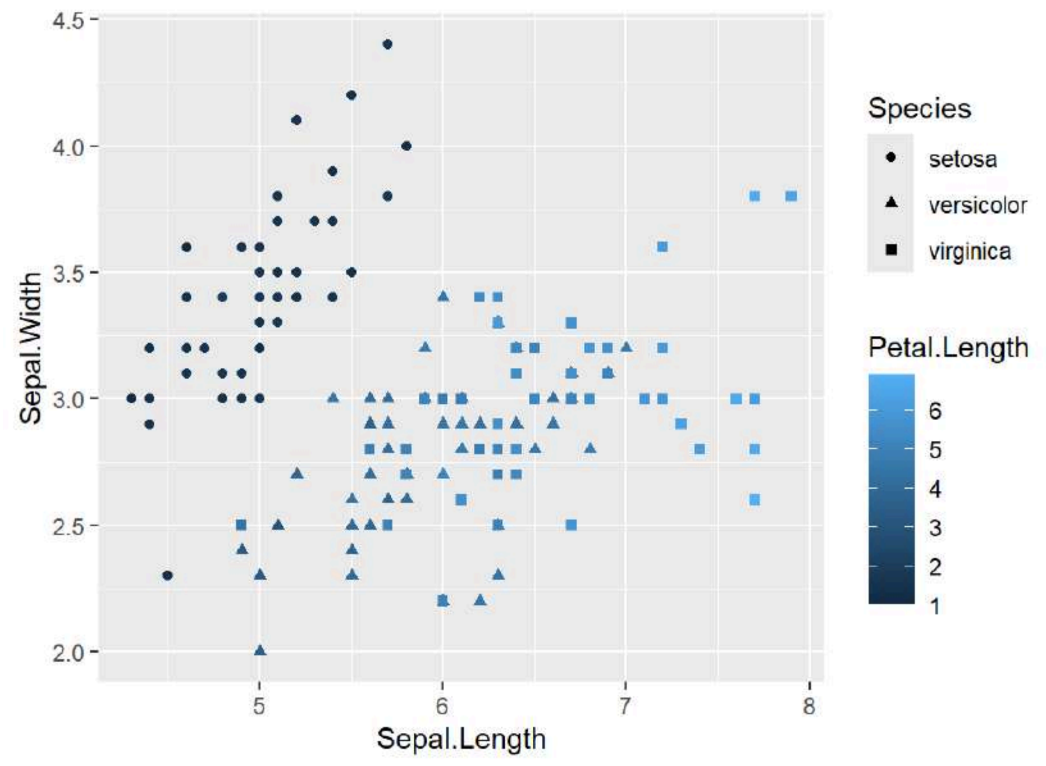


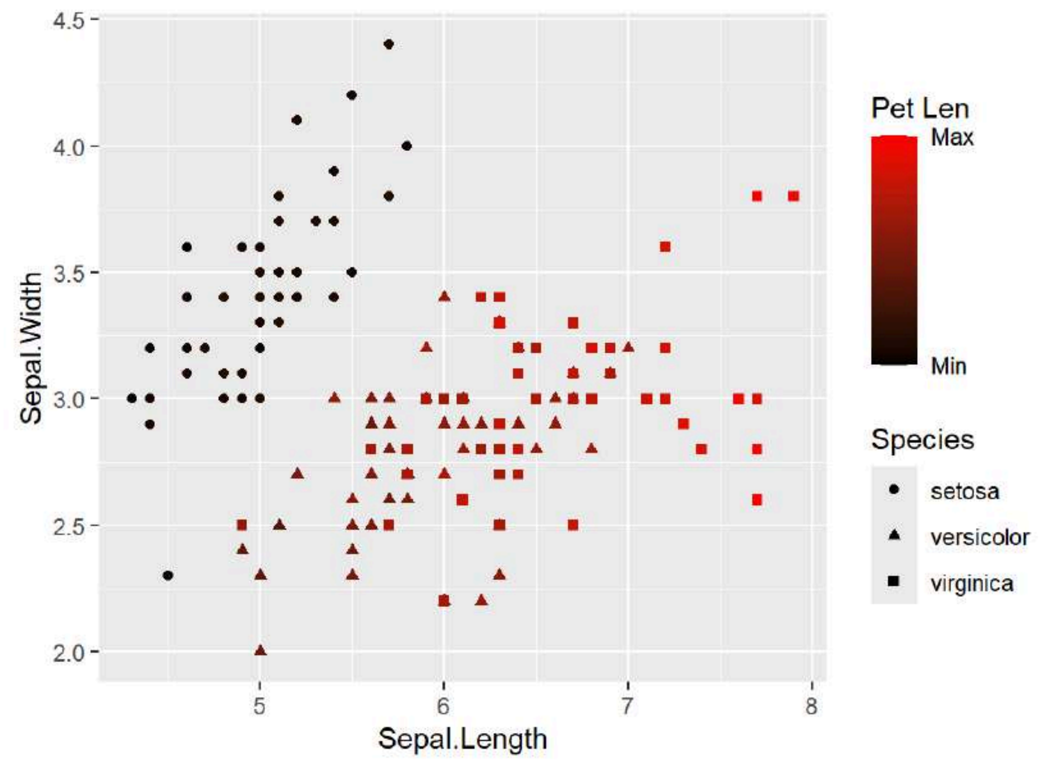
Scales · `Continuous Colors`

- In RStudio, type `scale_`, then press **TAB**

```
p <- ggplot(iris)+
      geom_point(aes(x=Sepal.Length,
                     y=Sepal.Width,
      shape=Species,color=Petal.Length))
p
```

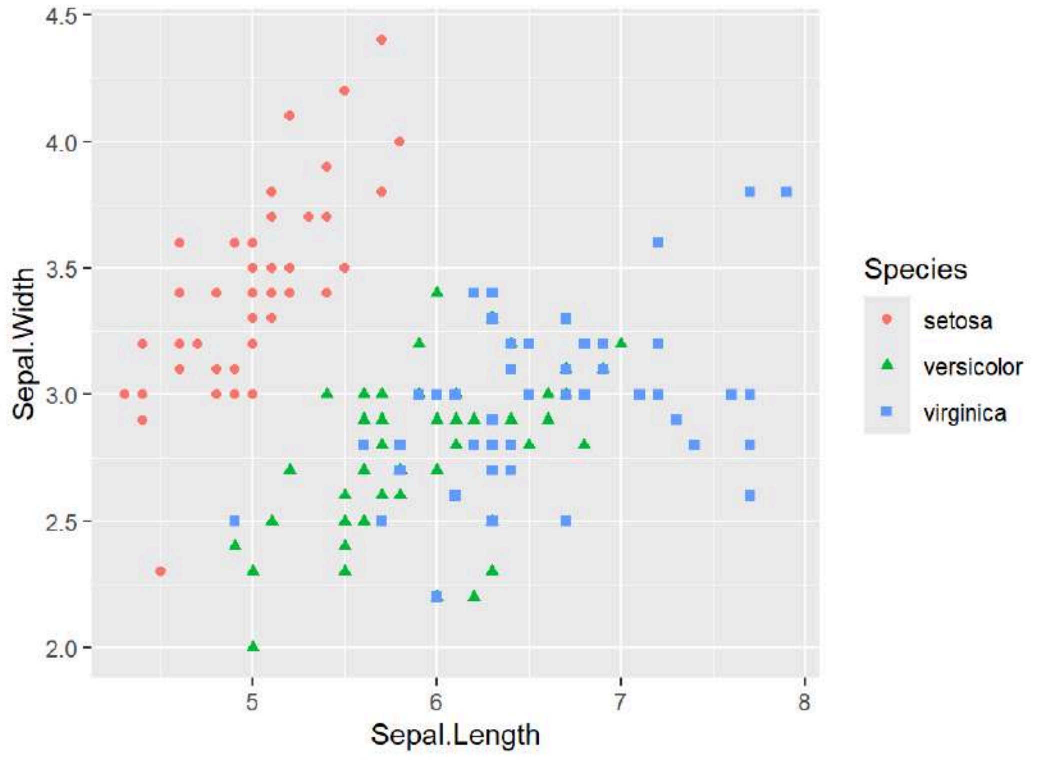
```
p +
scale_color_gradient(name="Pet Len",
  breaks=range(iris$Petal.Length),
  labels=c("Min","Max"),
  low="black",high="red")
```



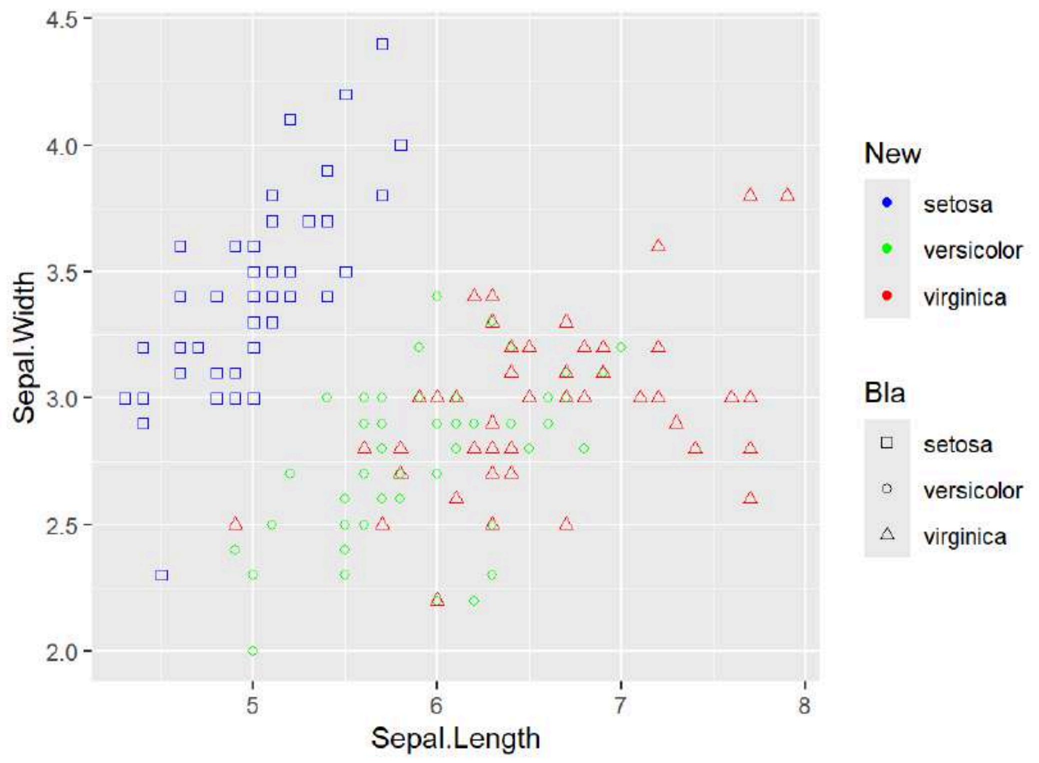


Scales · Shape

```r
p <- ggplot(iris)+
      geom_point(aes(x=Sepal.Length,
                     y=Sepal.Width,
      shape=Species,color=Species))
p
```



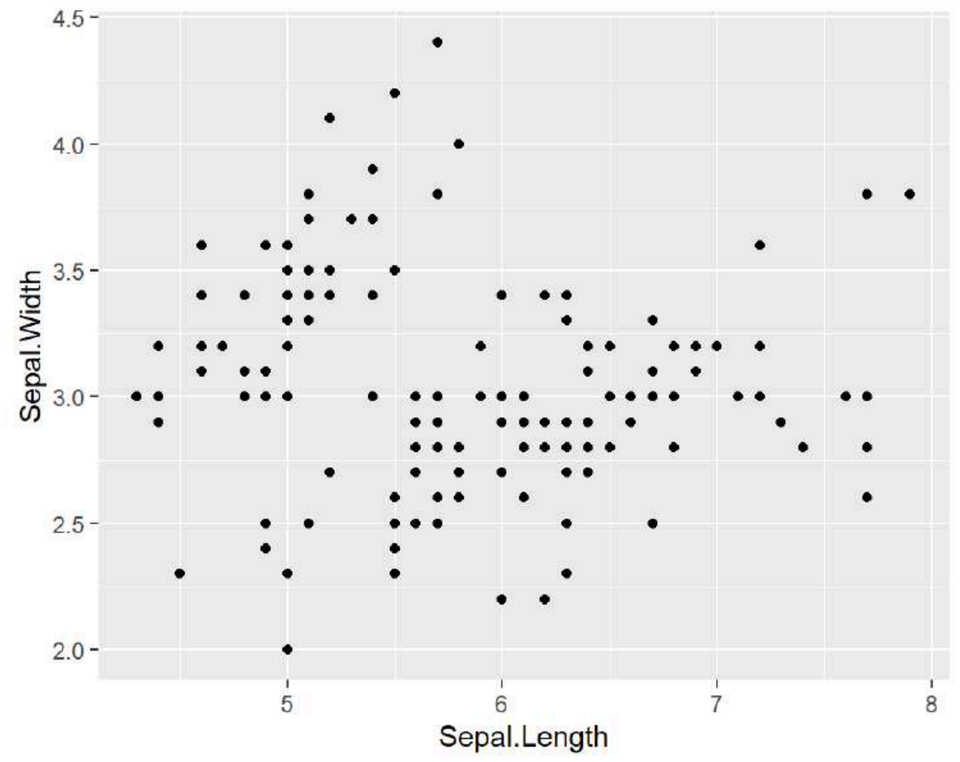
```r
p +
scale_color_manual(name="New",
   values=c("blue","green","red"))+
scale_shape_manual(name="Bla",values=c(0,1,2))
```



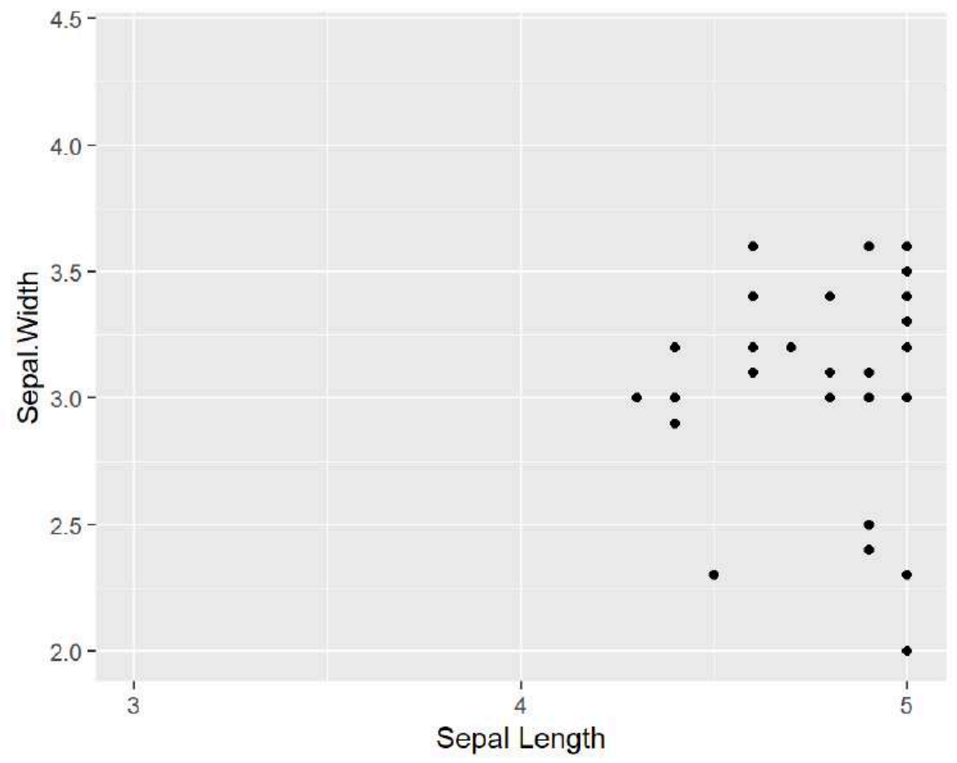
Scales · Axes

- scales: x, y
- syntax: `scale_<axis>_<type>`
- arguments: name, limits, breaks, labels

```r
p <- ggplot(iris)+geom_point(
  aes(x=Sepal.Length,y=Sepal.Width))
p
```

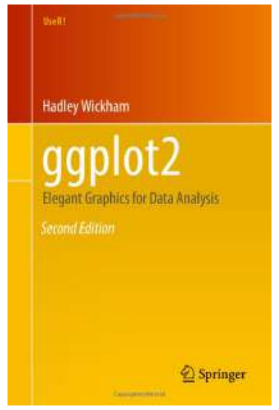


```r
p + scale_x_continuous(name="Sepal Length",
          breaks=seq(1,8),limits=c(3,5))
```

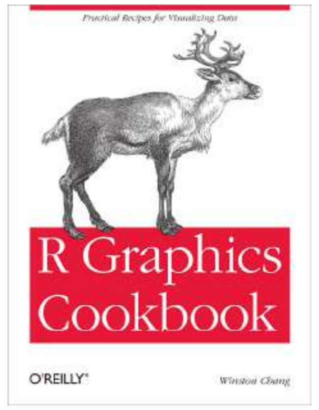


Help

- ggplot2 book



- The R cookbook



- ggplot2 official reference
- RStudio cheatsheet
- r-statistics ggplot2 cheatsheet
- StackOverflow
- Blogs, R-Bloggers, Cedric Scherer etc.

Thank you! Questions?

Acknowledgements:

• SLUBI • 3Bs • Slides adapted from RaukR • GPL-3 License