Environmental Computing

Better figures with ggplot2

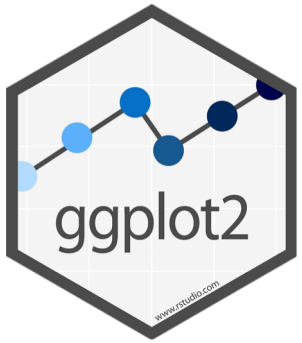
Fonti Kar

@fonti_kar

INTER-DISCIPLINARY ECOLOGY AND EVOLUTION LAB

R Ladies

VS.



https://flowingdata.com/2016/03/22/comparing-ggplot2-and-r-base-graphics/

VS.

- plot() and par()
- Error bars are a nightmare
- Long chunky scripts
- Very specific functions/packages for different plots e.g. lattice::

- **Tidy, intuitive coding**
- **Many handy built-in functions**
- **Less time spent on grumpy coding**
- **The ‘profesh’ look**
- Complex customising (if you need it)

https://flowingdata.com/2016/03/22/comparing-ggplot2-and-r-base-graphics/

The recipe

```
ggplot(data, aes(x_var, y_var))
    + geom_plot()
    + theme_ ()
```

'aesthetics' – what you want to be depict in your plot

what type of plot do you want?

the 'look' of the plot and other nitty gritty things (e.g. axes, font size, legend position)

An example

‘aesthetics’ – what you want to be depict in your plot

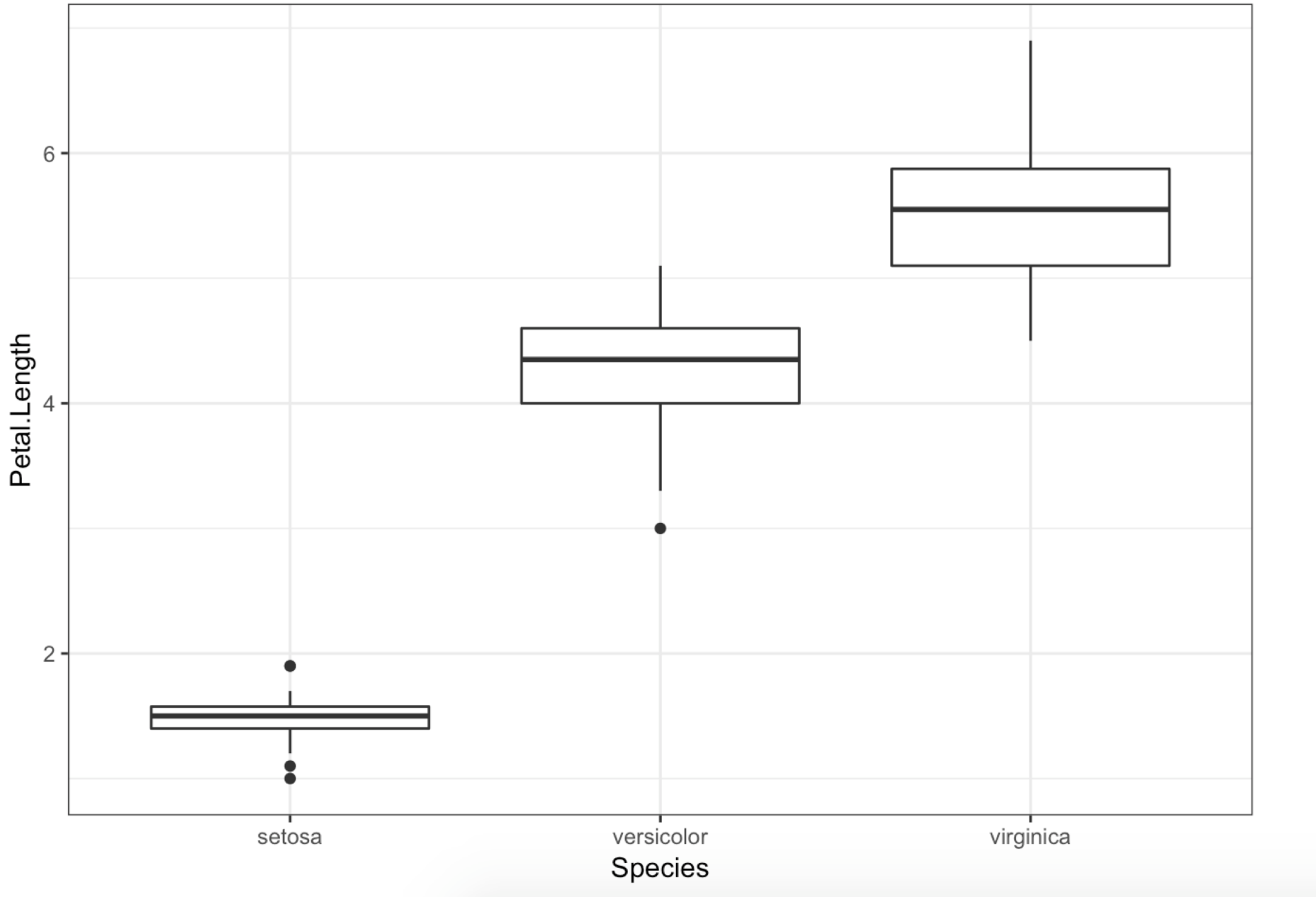
```
ggplot(iris, aes(x = Species,y = Petal.Length))
        + geom_boxplot()
        + theme_bw()
```

what type of plot do you want?

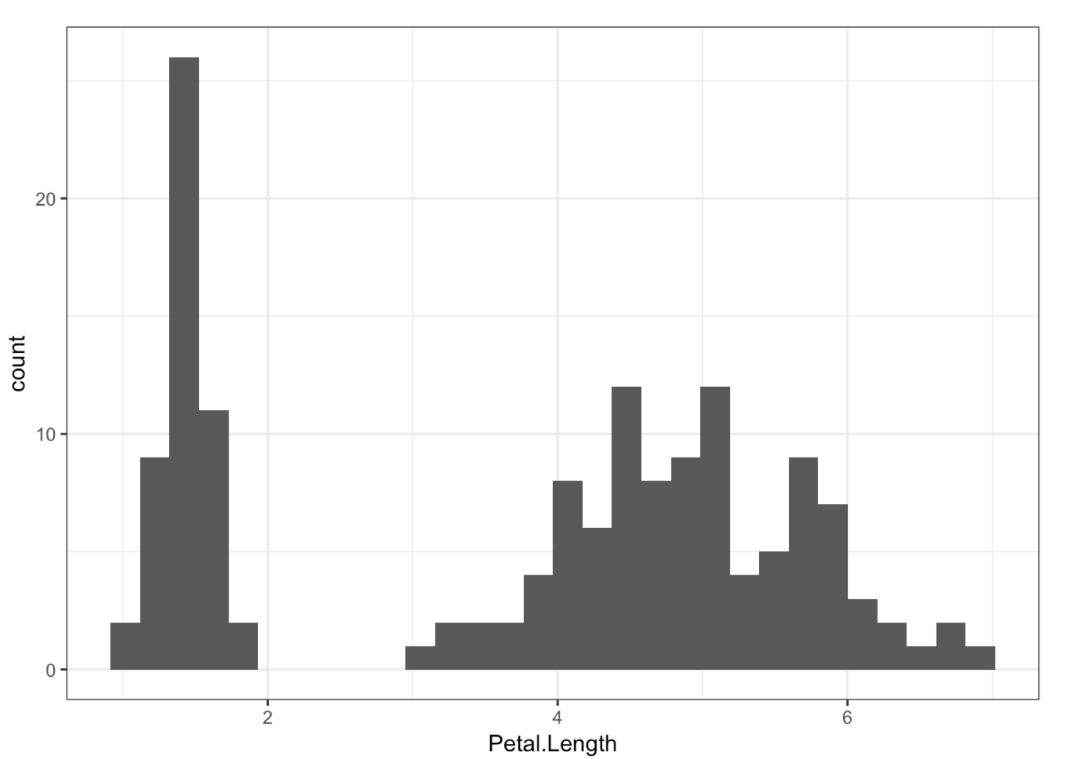
the ‘look’ of the plot and other nitty gritty things (e.g. axes, font size, legend position)

Iris Versicolor | Iris Setosa | Iris Virginica

```
ggplot(iris, aes(x = Species, y = Petal.Length)) + geom_boxplot() + theme_bw()
```

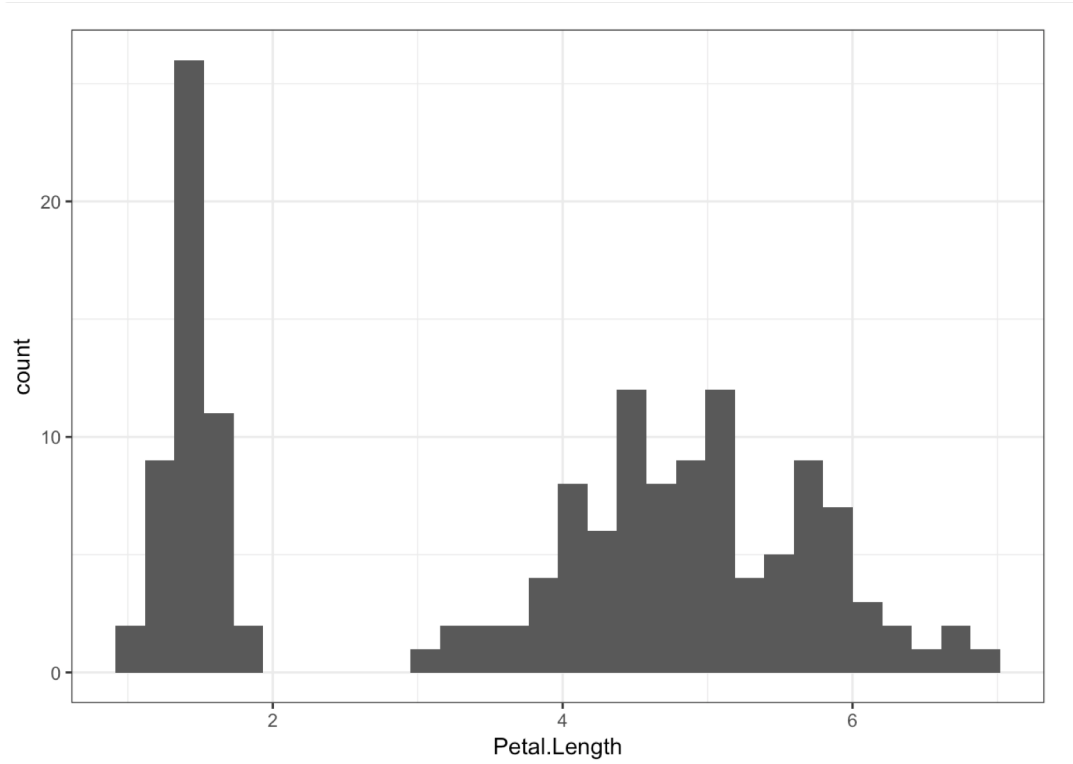


Some other key plots

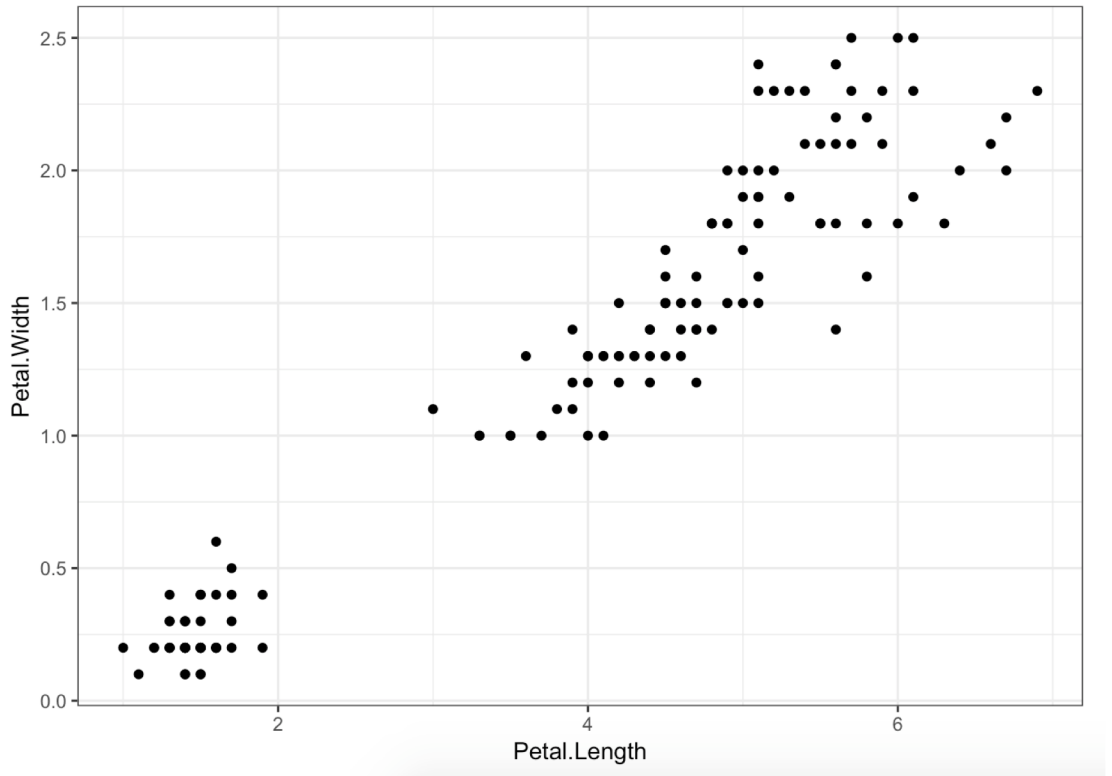



```
ggplot(iris, aes(x = Petal.Length))
+ geom_histogram()
+ theme_bw()
```

Some other key plots




```
ggplot(iris, aes(x = Petal.Length))
+ geom_histogram()
+ theme_bw()
```




```
ggplot(iris, aes(x = Petal.Length,
                 y = Petal.Width))
+ geom_point()
+ theme_bw()
```

Geom bonanza!

Get the cheat sheet!

Geoms - Use a geom function to represent data points, use the geom's aesthetic properties to represent variables. Each function returns a layer.

Graphical Primitives

a <- ggplot(economics, aes(date, unemploy))
b <- ggplot(seals, aes(x = long, y = lat))

- a + **geom_blank()** (Useful for expanding limits)
- b + **geom_curve**(aes(yend = lat + 1, xend=long+1,curvature=z)) - x, xend, y, yend, alpha, angle, color, curvature, linetype, size
- a + **geom_path**(lineend="butt", linejoin="round', linemitre=1) x, y, alpha, color, group, linetype, size
- a + **geom_polygon**(aes(group = group)) x, y, alpha, color, fill, group, linetype, size
- b + **geom_rect**(aes(xmin = long, ymin=lat, xmax= long + 1, ymax = lat + 1)) - xmax, xmin, ymax, ymin, alpha, color, fill, linetype, size
- a + **geom_ribbon**(aes(ymin=unemploy - 900, ymax=unemploy + 900)) - x, ymax, ymin alpha, color, fill, group, linetype, size

Line Segments

common aesthetics: x, y, alpha, color, linetype, size

- b + **geom_abline**(aes(intercept=0, slope=1))
- b + **geom_hline**(aes(yintercept = lat))
- b + **geom_vline**(aes(xintercept = long))
- b + **geom_segment**(aes(yend=lat+1, xend=long+1))
- b + **geom_spoke**(aes(angle = 1:1155, radius = 1))

One Variable

Continuous

c <- ggplot(mpg, aes(hwy)); c2 <- ggplot(mpg)

- c + **geom_area**(stat = "bin") x, y, alpha, color, fill, linetype, size
- c + **geom_density**(kernel = "gaussian") x, y, alpha, color, fill, group, linetype, size, weight
- c + **geom_dotplot**() x, y, alpha, color, fill
- c + **geom_freqpoly**() x, y, alpha, color, group, linetype, size
- c + **geom_histogram**(binwidth = 5) x, y, alpha, color, fill, linetype, size, weight
- c2 + **geom_qq**(aes(sample = hwy)) x, y, alpha, color, fill, linetype, size, weight

Discrete

d <- ggplot(mpg, aes(fl))

- d + **geom_bar**() x, alpha, color, fill, linetype, size, weight

Two Variables

Continuous X, Continuous Y

e <- ggplot(mpg, aes(cty, hwy))

- e + **geom_label**(aes(label = cty), nudge_x = 1, nudge_y = 1, check_overlap = TRUE) x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust
- e + **geom_jitter**(height = 2, width = 2) x, y, alpha, color, fill, shape, size
- e + **geom_point**() x, y, alpha, color, fill, shape, size, stroke
- e + **geom_quantile**() x, y, alpha, color, group, linetype, size, weight
- e + **geom_rug**(sides = "bl") x, y, alpha, color, linetype, size
- e + **geom_smooth**(method = lm) x, y, alpha, color, fill, group, linetype, size, weight
- e + **geom_text**(aes(label = cty), nudge_x = 1, nudge_y = 1, check_overlap = TRUE) x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust

Continuous Bivariate Distribution

h <- ggplot(diamonds, aes(carat, price))

- h + **geom_bin2d**(binwidth = c(0.25, 500)) x, y, alpha, color, fill, linetype, size, weight
- h + **geom_density2d**() x, y, alpha, colour, group, linetype, size
- h + **geom_hex**() x, y, alpha, colour, fill, size

Continuous Function

i <- ggplot(economics, aes(date, unemploy))

- i + **geom_area**() x, y, alpha, color, fill, linetype, size
- i + **geom_line**() x, y, alpha, color, group, linetype, size
- i + **geom_step**(direction = "hv") x, y, alpha, color, group, linetype, size

Visualizing error

df <- data.frame(grp = c("A", "B"), fit = 4:5, se = 1:2)
j <- ggplot(df, aes(grp, fit, ymin = fit-se, ymax = fit+se))

- j + **geom_crossbar**(fatten = 2) x, y, ymax, ymin, alpha, color, fill, group, linetype, size
- j + **geom_errorbar**() x, ymax, ymin, alpha, color, group, linetype, size, width (also **geom_errorbarh**())
- j + **geom_linerange**() x, ymin, ymax, alpha, color, group, linetype, size
- j + **geom_pointrange**() x, y, ymin, ymax, alpha, color, fill, group, linetype, shape, size

Discrete X, Continuous Y

f <- ggplot(mpg, aes(class, hwy))

- f + **geom_col**() x, y, alpha, color, fill, group, linetype, size
- f + **geom_boxplot**() x, y, lower, middle, upper, ymax, ymin, alpha, color, fill, group, linetype, shape, size, weight
- f + **geom_dotplot**(binaxis = "y", stackdir = "center") x, y, alpha, color, fill, group
- f + **geom_violin**(scale = "area") x, y, alpha, color, fill, group, linetype, size, weight

Discrete X, Discrete Y

g <- ggplot(diamonds, aes(cut, color))

- g + **geom_count**() x, y, alpha, color, fill, shape, size, stroke

Maps

data <- data.frame(murder = USArrests$Murder, state = tolower(rownames(USArrests)))
map <- map_data("state")
k <- ggplot(data, aes(fill = murder))

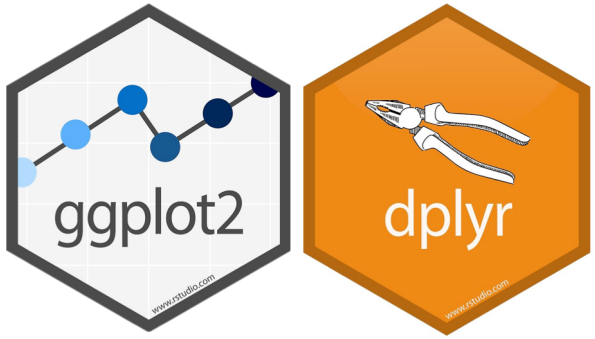
- k + **geom_map**(aes(map_id = state), map = map) + **expand_limits**(x = map$long, y = map$lat) map_id, alpha, color, fill, linetype, size

Three Variables

seals$z <- with(seals, sqrt(delta_long^2 + delta_lat^2))
l <- ggplot(seals, aes(long, lat))

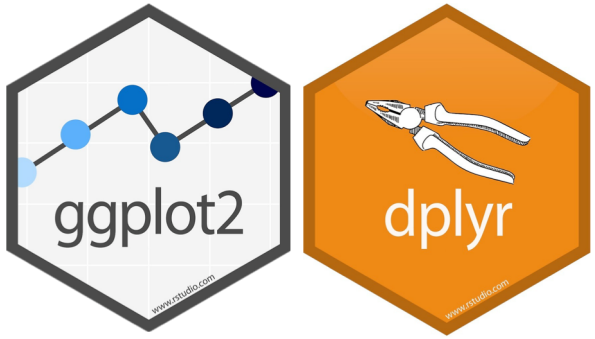
- l + **geom_contour**(aes(z = z)) x, y, z, alpha, colour, group, linetype, size, weight
- l + **geom_raster**(aes(fill = z), hjust=0.5, vjust=0.5, interpolate=FALSE) x, y, alpha, fill
- l + **geom_tile**(aes(fill = z)) x, y, alpha, color, fill, linetype, size, width

https://www.rstudio.com/wp-content/uploads/2015/03/ggplot2-cheatsheet.pdf



Wrangle data and plot it!

I want plot of **mean** Petal.Length and **standard error bars** for each species

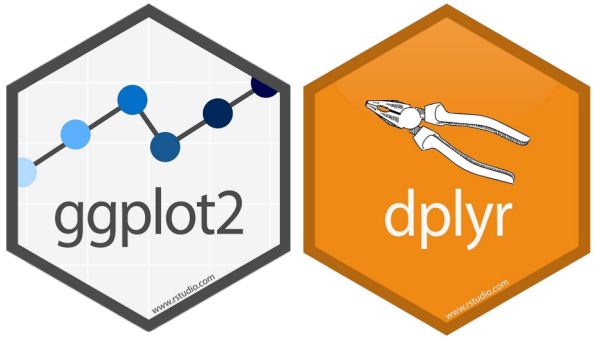


Wrangle data and plot it!

I want plot of **mean** Petal.Length and **standard error bars** for each species

1) Compute what you want

```
iris.barplot <- iris %>% group_by(Species) %>%
summarise(meanPetal.Length = mean(Petal.Length),
          stderPetal.Length = sd(Petal.Length)/sqrt(length(Petal.Length)),
          lowerErr = (meanPetal.Length - stderPetal.Length),
          upperErr = (meanPetal.Length + stderPetal.Length))
```

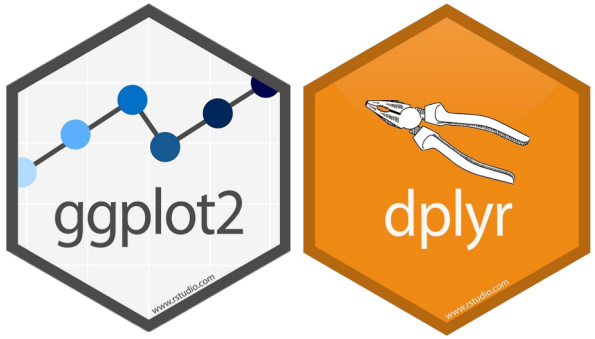


Wrangle data and plot it!

I want plot of **mean** Petal.Length and **standard error bars** for each species

2) Check your data

```
> iris.barplot
# A tibble: 3 x 5
  Species    meanPetal.Length stderPetal.Length lowerErr upperErr
  <fct>                 <dbl>             <dbl>    <dbl>    <dbl>
1 setosa                 1.46            0.0246     1.44     1.49
2 versicolor             4.26            0.0665     4.19     4.33
3 virginica              5.55            0.0780     5.47     5.63
```

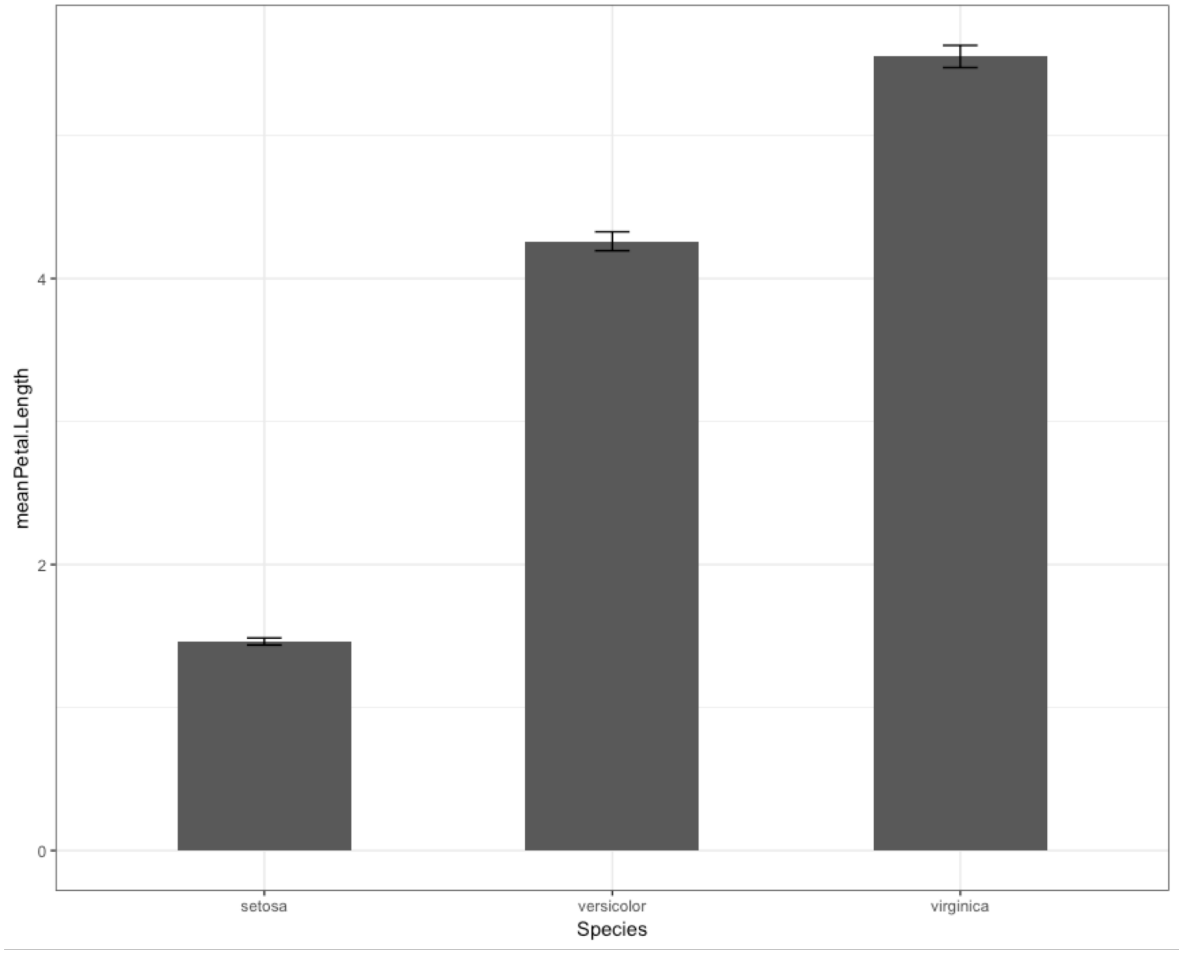


Wrangle data and plot it!

I want plot of **mean** Petal.Length and **standard error bars** for each species

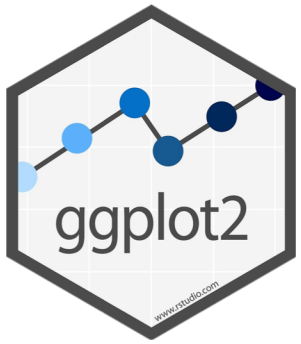
3) Plot your data

```
ggplot(iris.barplot,
aes(x = Species,
    y = meanPetal.Length))
+ geom_col()
+ geom_errorbar(
aes(ymin = lowerErr,
    ymax = upperErr),
    width = 0.2)
+ theme_bw()
```



A quick comparison between

vs.




```
meanPetal.length <- tapply(iris$Petal.Length, iris$Species,
mean)
nPetal.length <- tapply(iris$Petal.Length, iris$Species,
length)
sePetal.length <- tapply(iris$Petal.Length,
iris$Species,sd)/sqrt(nPetal.length)
```

WARNING: GRUMPY CODING AHEAD

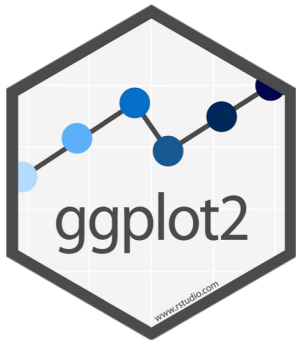
```
xx <- barplot(meanPetal.length,
ylim=c(0,max(meanPetal.length+sePetal.length)*1.1))

arrows(xx,meanPetal.length-sePetal.length,
xx, meanPetal.length+sePetal.length,
angle=90,code=3,length  = 0)

box()
```

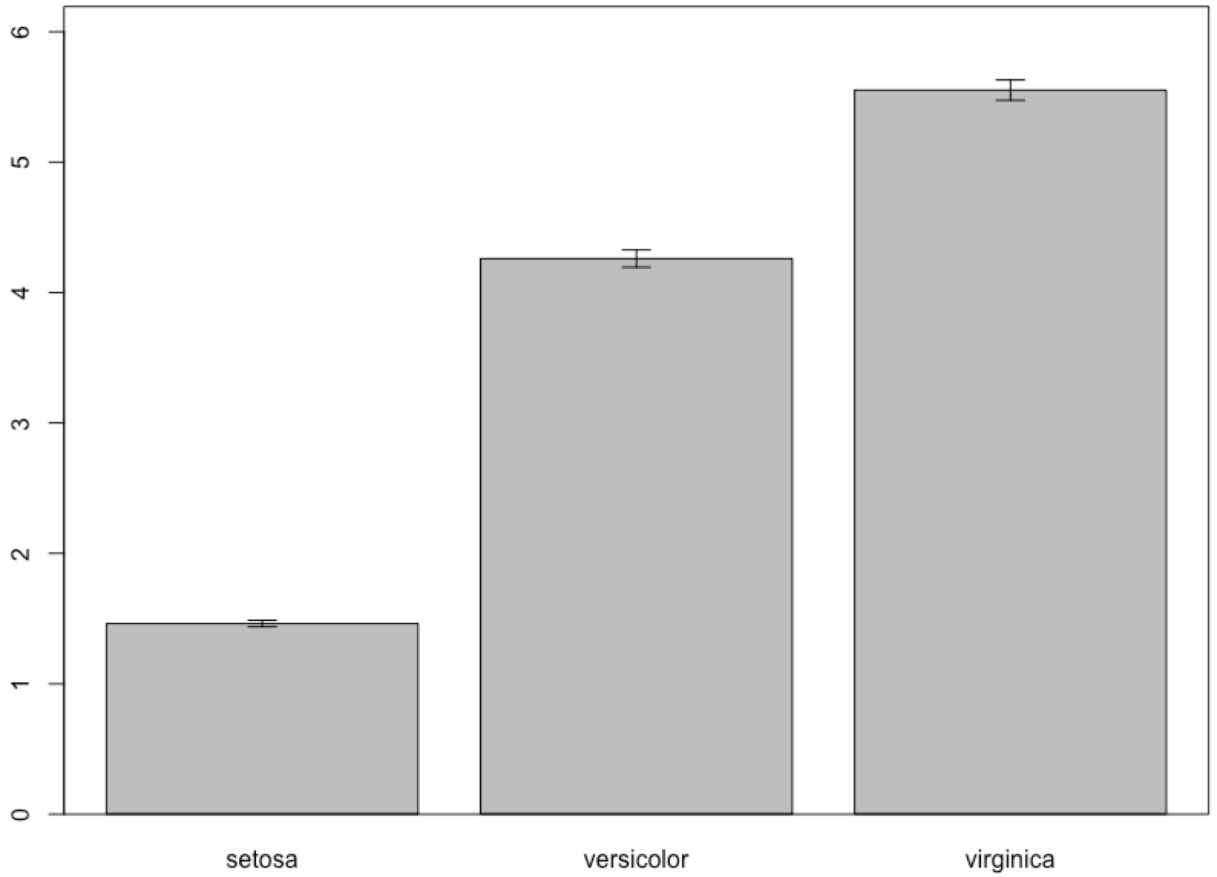
```
iris.barplot <- iris %>% group_by(Species) %>%
summarise(meanPetal.Length = mean(Petal.Length),
          stderPetal.Length =
sd(Petal.Length)/sqrt(length(Petal.Length)),
          lowerErr = (meanPetal.Length - stderPetal.Length),
          upperErr = (meanPetal.Length + stderPetal.Length))
```

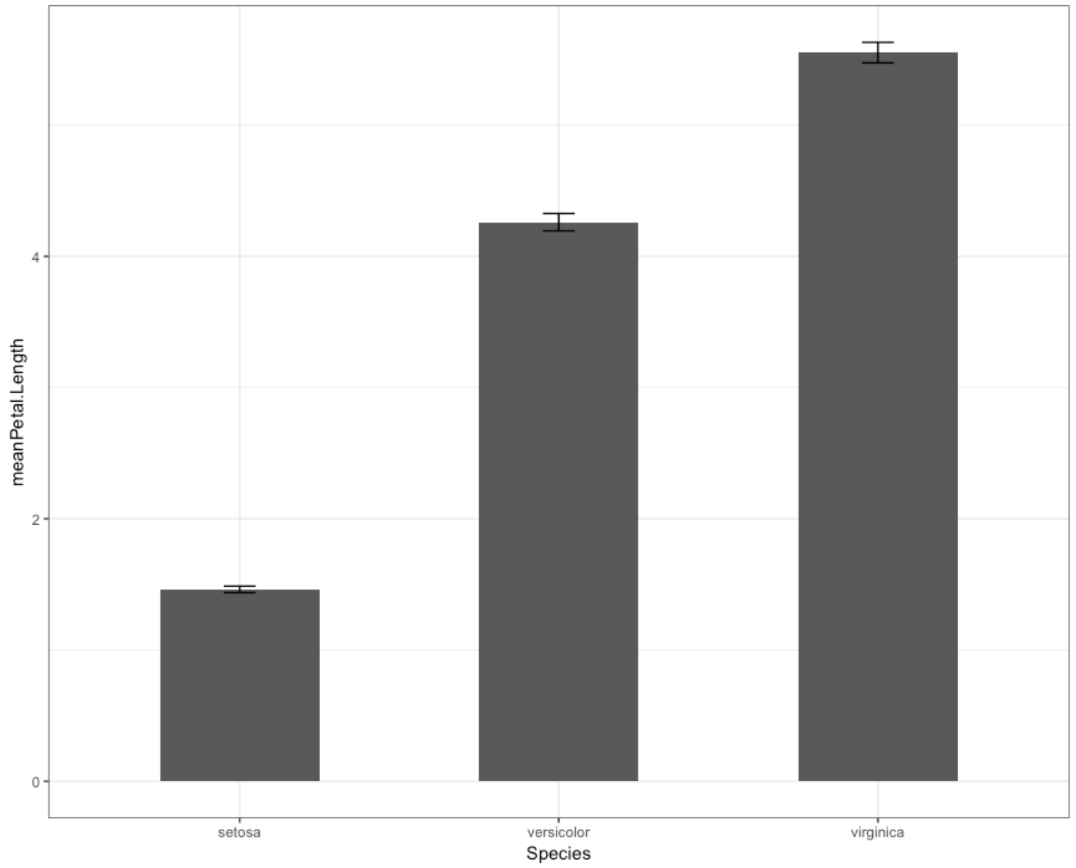
```
ggplot(iris.barplot,
aes(x = Species,
    y = meanPetal.Length)) + geom_col()
+ geom_errorbar(
aes(ymin = lowerErr,
    ymax = upperErr),
    width = 0)
+ theme_bw()
```

A quick comparison between

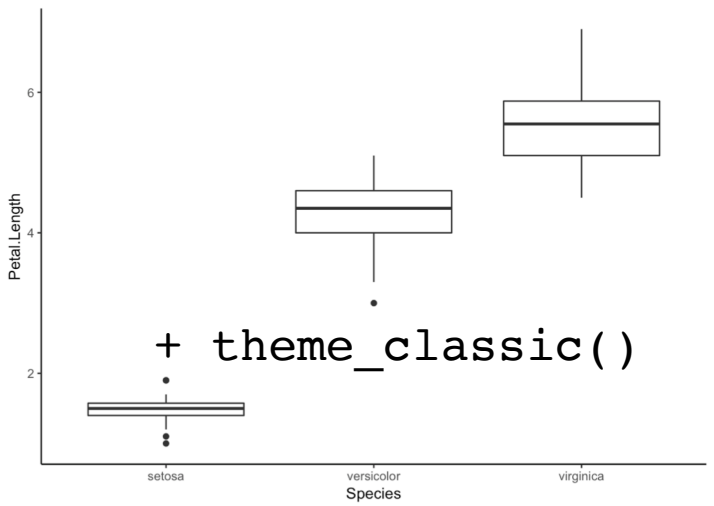
vs.

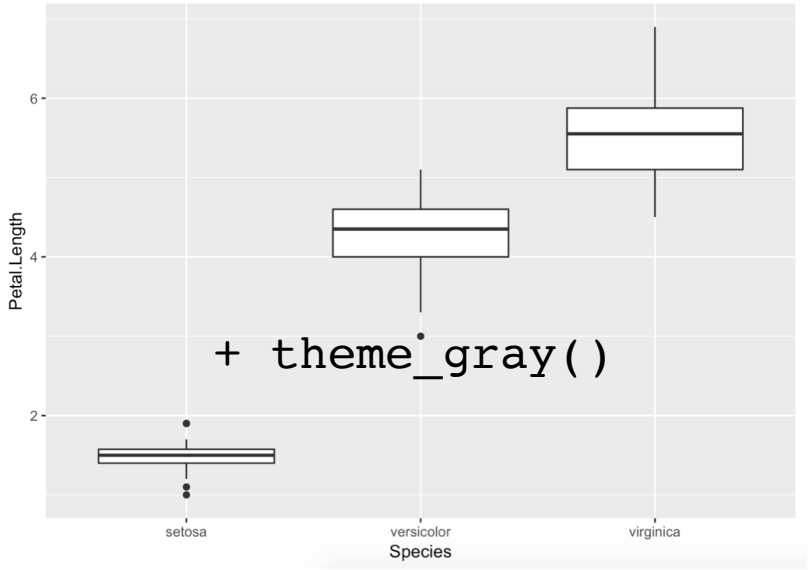


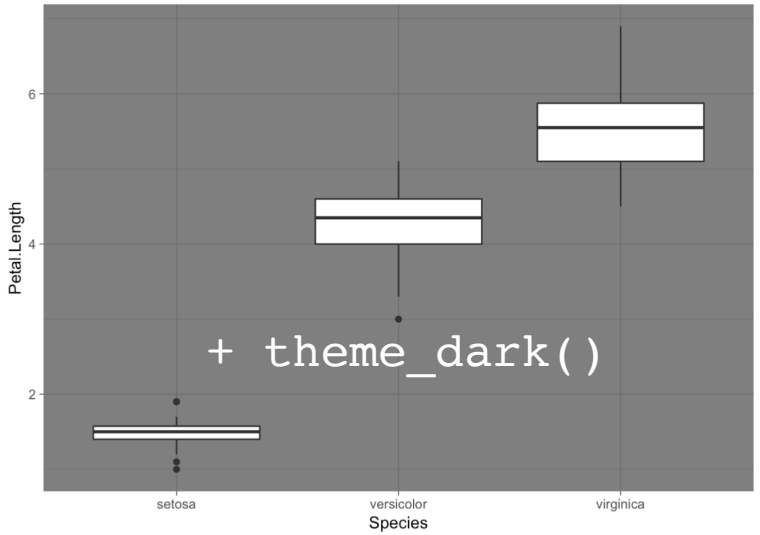


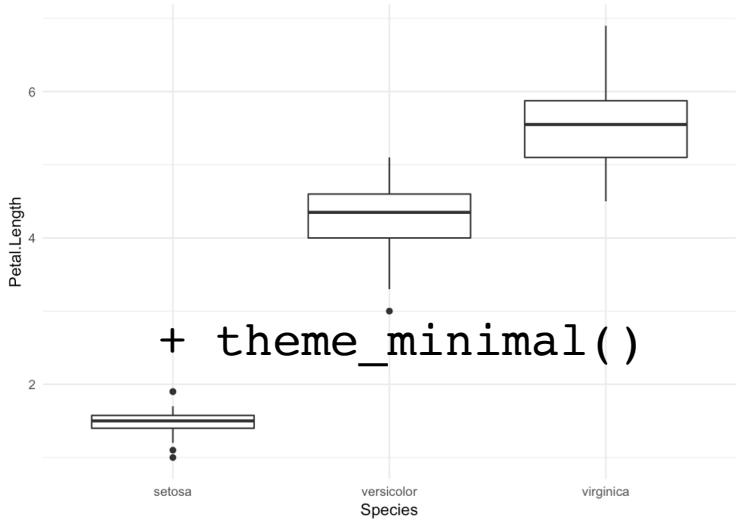


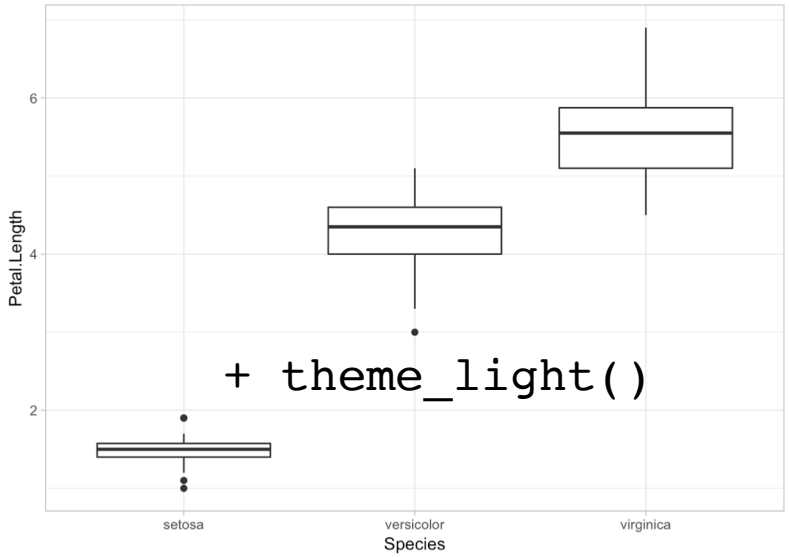
A quick look at themes

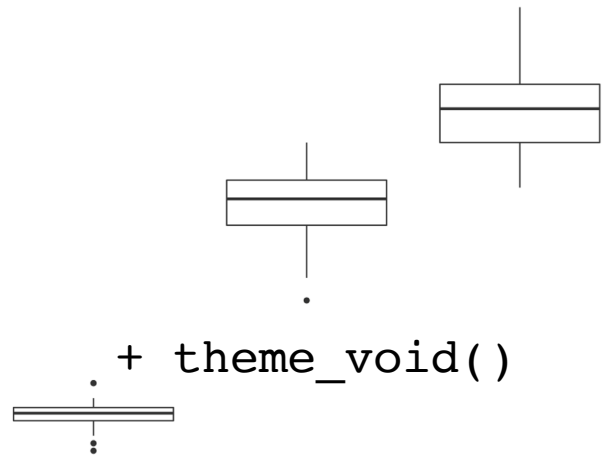












Customising your plot a bit more

Iris Versicolor **Iris Setosa** **Iris Virginica**

Colour/shapes?

```
ggplot(data = iris, aes(x = Petal.Width, y = Petal.Length, color = Species)) +
  geom_point(size = 2.5, alpha = 0.6) +
  theme_bw() +
  theme(legend.position = "bottom",
        legend.text = element_text(size = 12, family="Comic Sans MS"),
        legend.title = element_text(size = 12, face = "bold", family="Comic Sans MS"),
        panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        axis.text= element_text(size = 18),
        axis.title= element_text(size = 20,face = "bold", family="Comic Sans MS"))
```

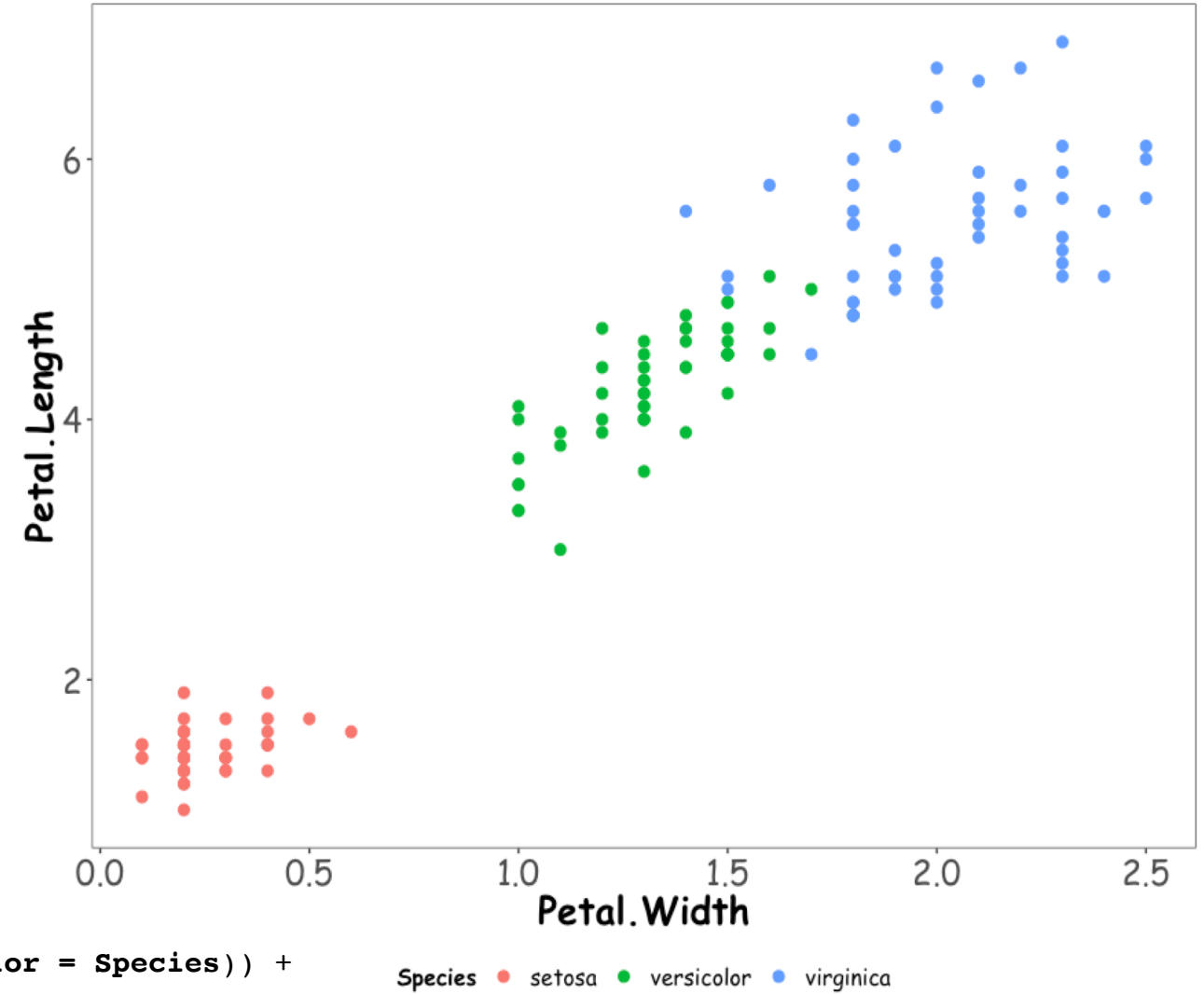
Fine-tuning your geom_

Fine-tuning your plot:
Legend?
Font type, size & style

Just because you can, doesn't mean you should…

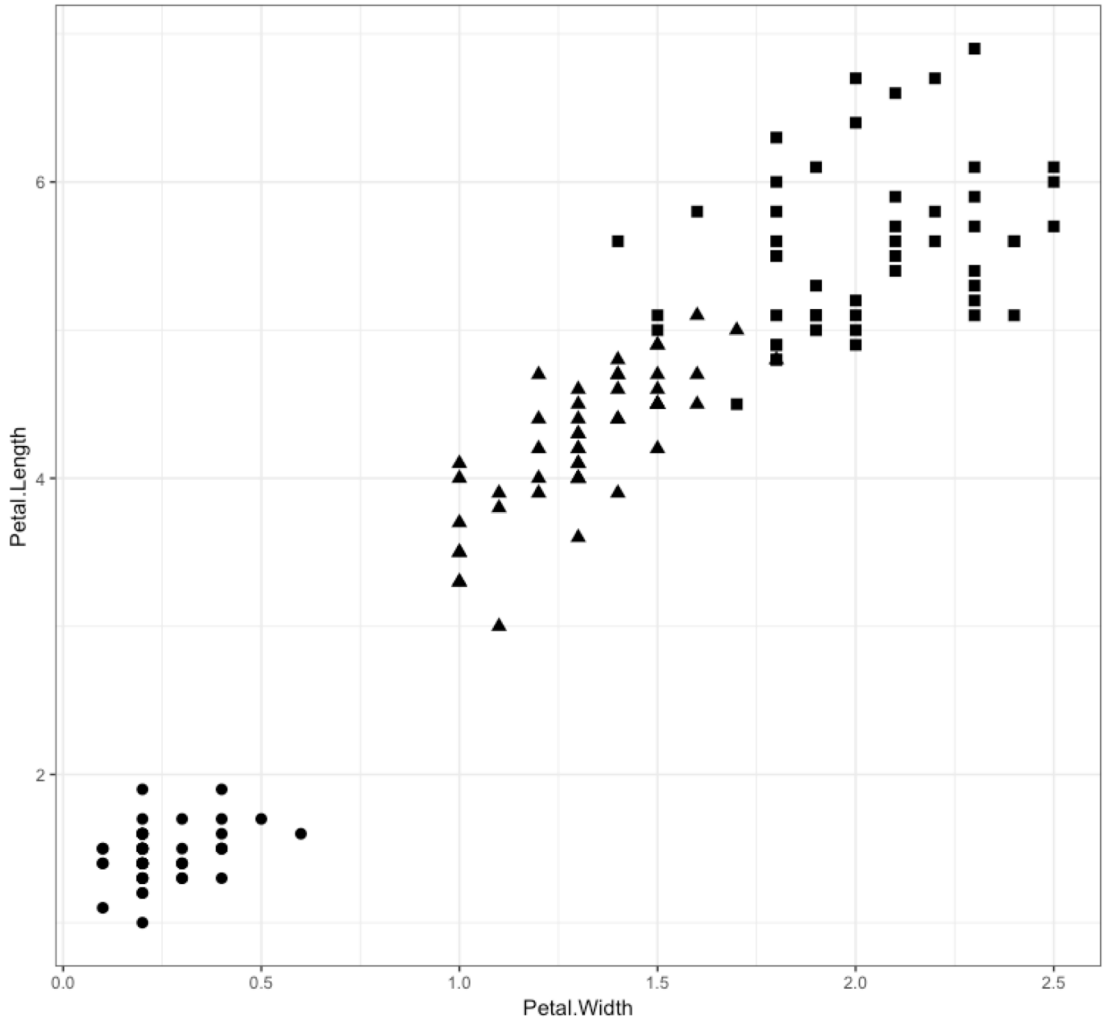
Consider:

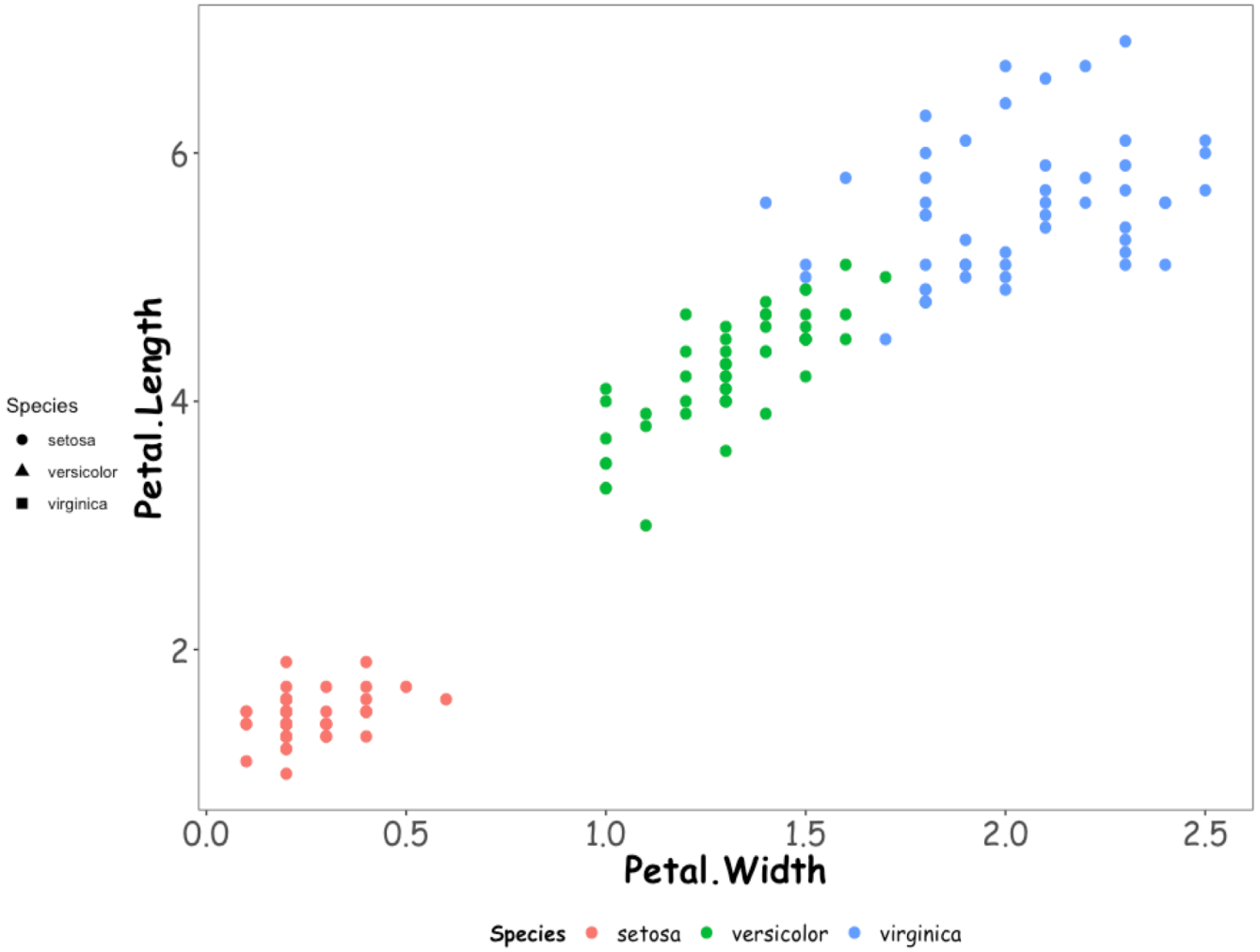
- Target audience
- What is the best way to depict what I want?
- Could I simplify it further?
- How would I describe this in the figure legend in my paper?




```
ggplot(data = iris, aes(x = Petal.Width, y = Petal.Length, color = Species)) +
  geom_point(size = 2.5, alpha = 0.6) +
  theme_bw() +
  theme(legend.position = "bottom",
        legend.text = element_text(size = 12, family="Comic Sans MS"),
        legend.title = element_text(size = 12, face = "bold", family="Comic Sans MS"),
        panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        axis.text= element_text(size = 18),
        axis.title= element_text(size = 20,face = "bold", family="Comic Sans MS"))
```

Less is *more*


Petal.Length
Petal.Width
0.0
0.5
1.0
1.5
2.0
2.5
2
4
6


Species
setosa
versicolor
virginica
Petal.Length
Petal.Width
0.0
0.5
1.0
1.5
2.0
2.5
2
4
6
Species setosa versicolor virginica

Creating your theme for consistency

This

```
my_thesis_theme <- theme(legend.position = "bottom",
                         legend.text = element_text(size = 12, family="Comic Sans MS"),
                         legend.title = element_text(size = 12, face = "bold", family="Comic Sans MS"),
                         panel.grid.major = element_blank(),
                         panel.grid.minor = element_blank(),
                         axis.text= element_text(size = 18),
                         axis.title= element_text(size = 20,face = "bold", family="Comic Sans MS"))

ggplot(data = iris, aes(x = Petal.Width, y = Petal.Length, color = Species)) +
  geom_point(size = 2.5, alpha = 0.6) +
  theme_bw() +
  my_thesis_theme
```

Add this for all your plots from now on!

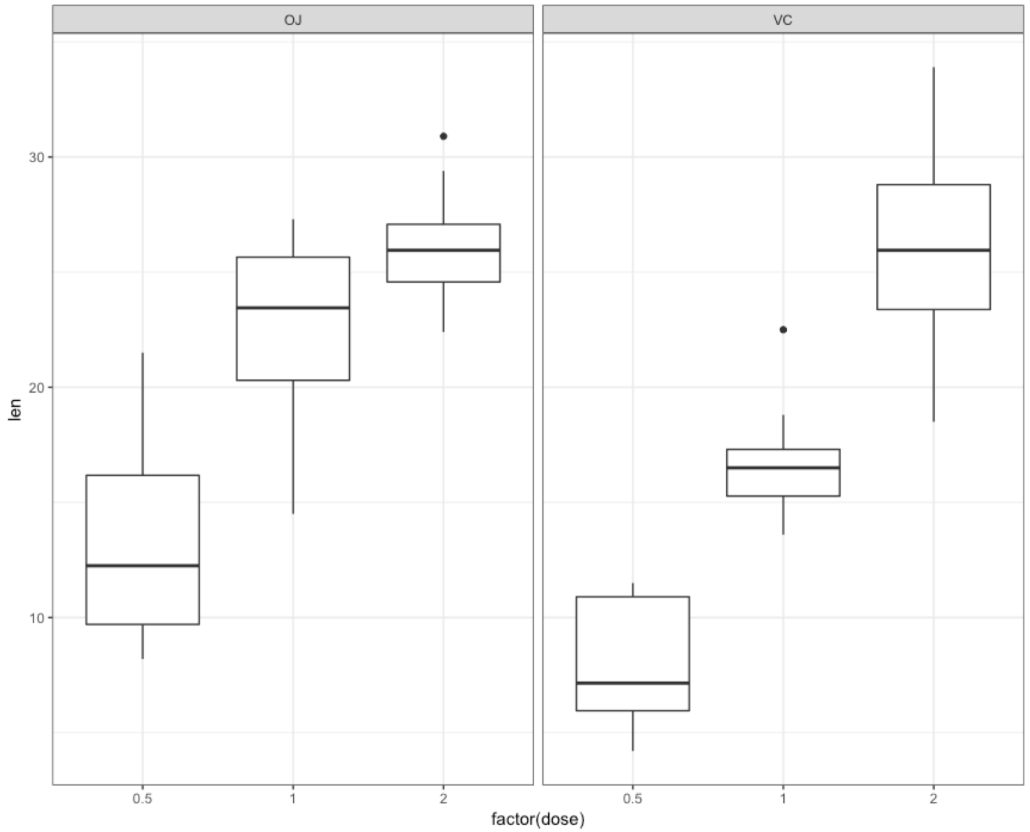
Instead of this

```
ggplot(data = iris, aes(x = Petal.Width, y = Petal.Length, color = Species)) +
  geom_point(size = 2.5, alpha = 0.6) +
  theme_bw() +
  theme(legend.position = "bottom",
        legend.text = element_text(size = 12, family="Comic Sans MS"),
        legend.title = element_text(size = 12, face = "bold", family="Comic Sans MS"),
        panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        axis.text= element_text(size = 18),
        axis.title= element_text(size = 20,face = "bold", family="Comic Sans MS"))
```

Facets! New favourite thing

Great for multivariate data (i.e. a factor and a continuous variable)

```
Toothplot <- ggplot(data = ToothGrowth, aes(x = factor(dose), y = len)) +
                    geom_boxplot() +
                    theme_bw()
```

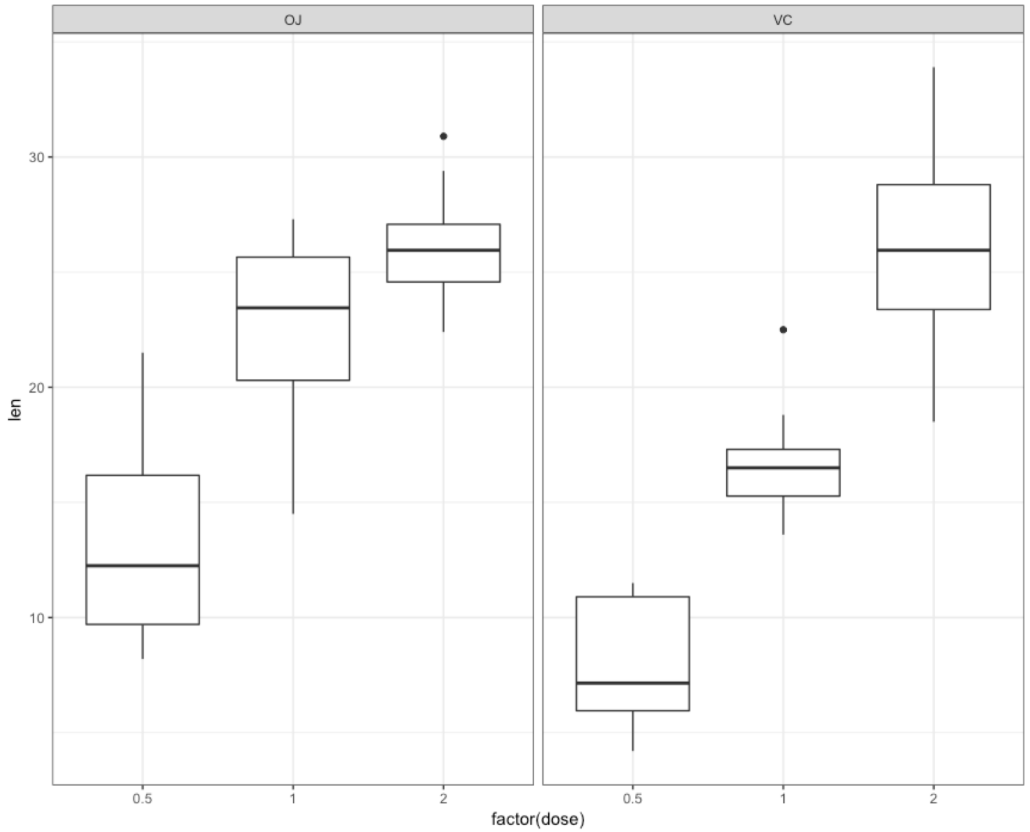



```
Toothplot + facet_wrap(~ supp)
```

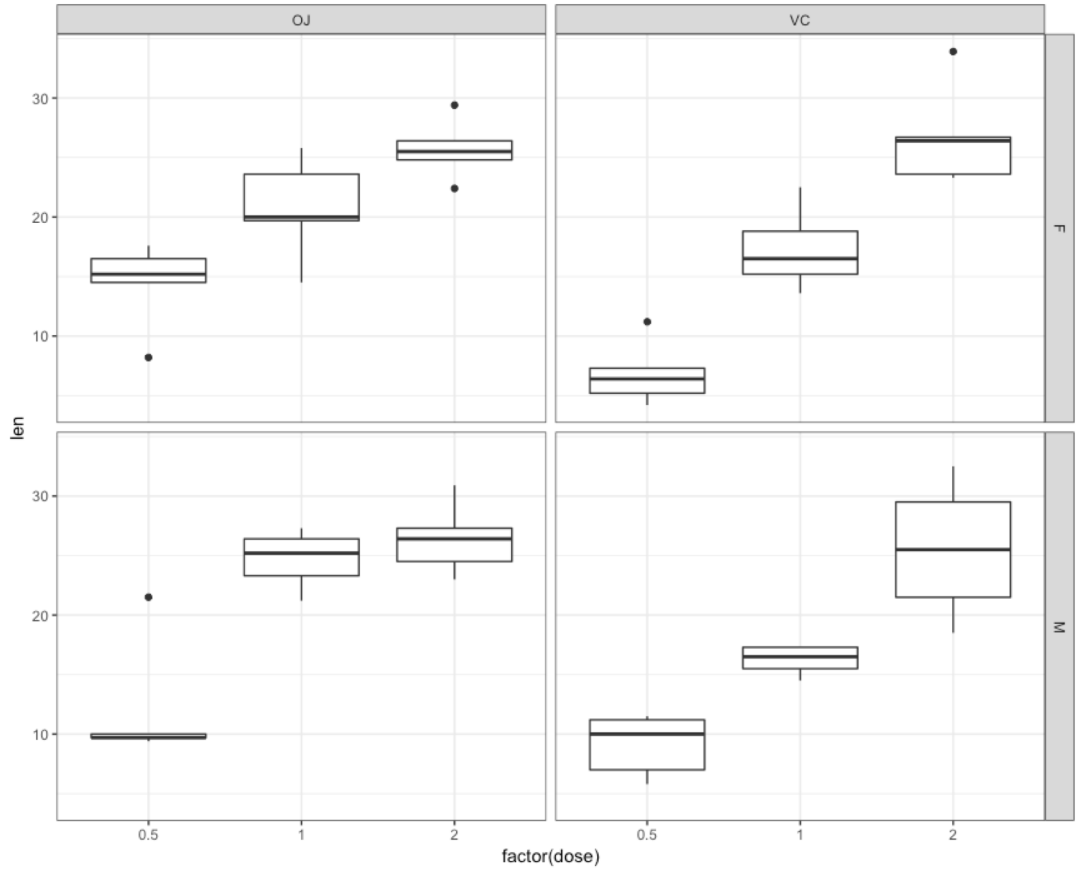
Facets! New favourite thing

Great for multivariate data (i.e. a factor and a continuous variable)

```
Toothplot <- ggplot(data = ToothGrowth, aes(x = factor(dose), y = len)) +
                    geom_boxplot() +
                    theme_bw()
```



Toothplot + **facet_wrap(~ supp)**



*Toothplot + **facet_grid(Sex ~ supp)**

* Run this first ToothGrowth$Sex <- rep(c("F", "M"), 2, 30)

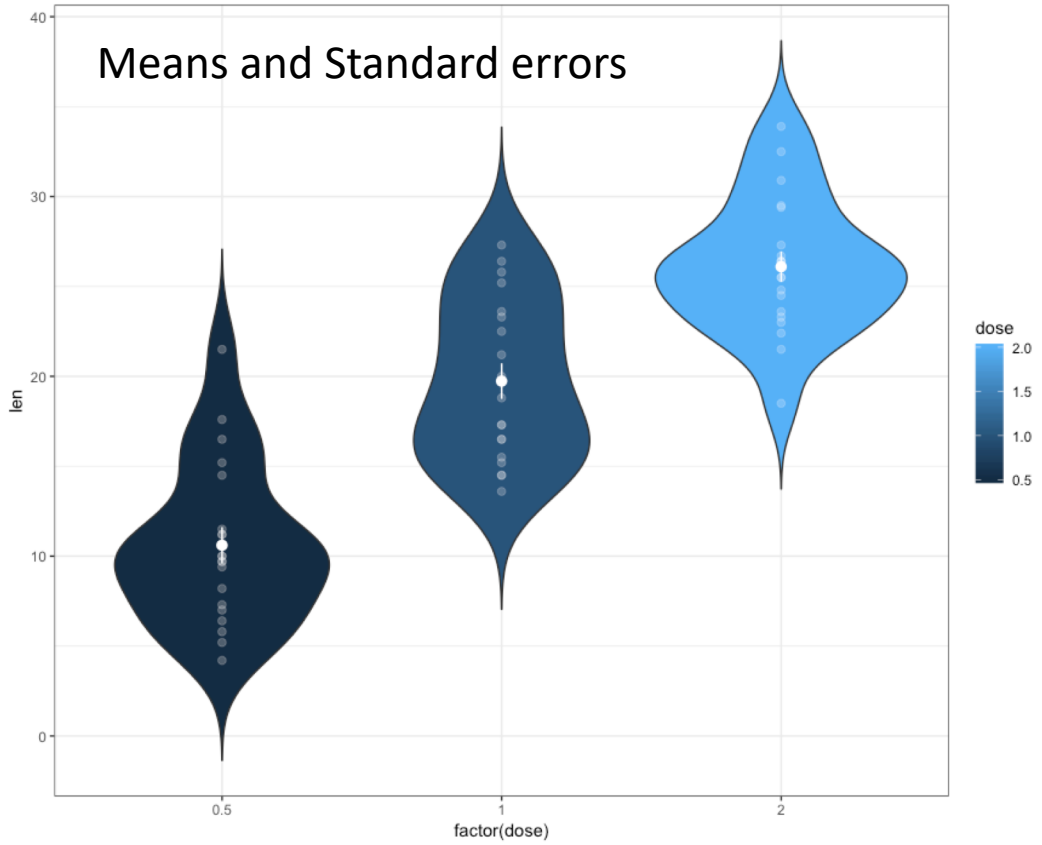
Built in functions

Some quick ways to explore data

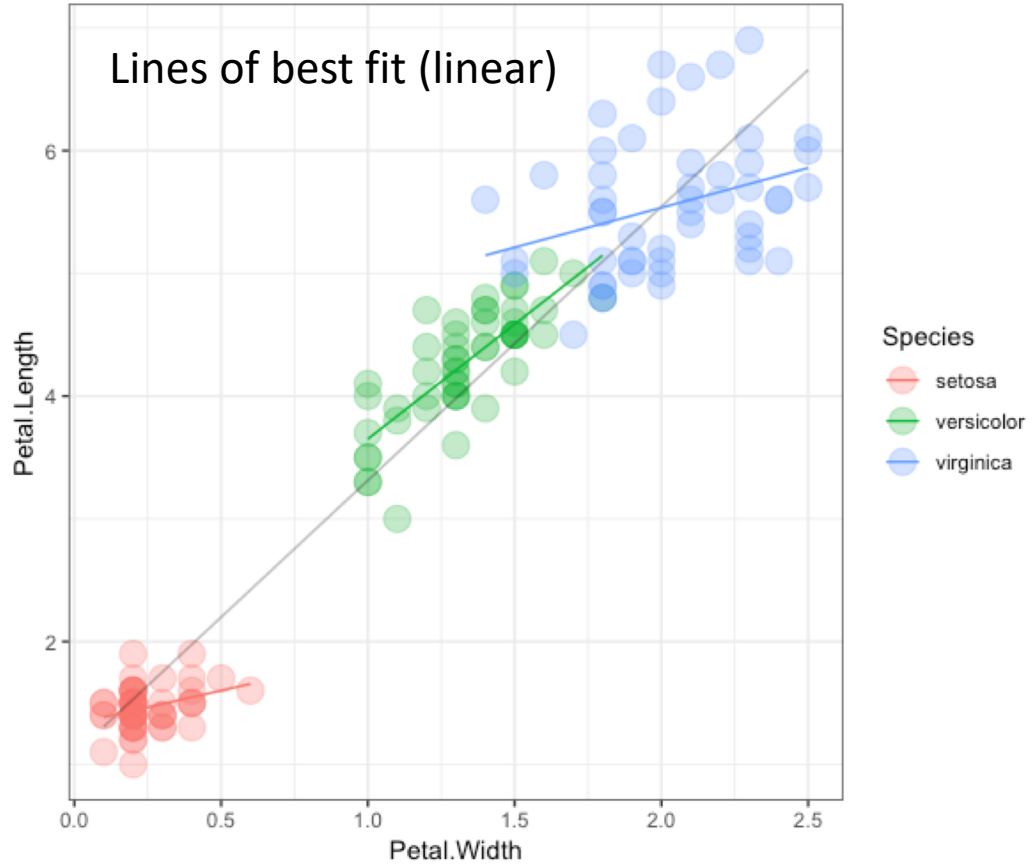

```
ggplot(data = ToothGrowth,
aes(x = factor(dose), y = len, fill = dose)) +
  geom_violin(trim = F) +
  geom_point(col = "white", size = 2, alpha = 0.2) +
  stat_summary(fun.data = "mean_se", col = "white") +
  theme_bw()
```

Built in functions

Some quick ways to explore data




```
ggplot(data = ToothGrowth,
aes(x = factor(dose), y = len, fill = dose)) +
  geom_violin(trim = F) +
  geom_point(col = "white", size = 2, alpha = 0.2) +
  stat_summary(fun.data = "mean_se", col = "white") +
  theme_bw()
```

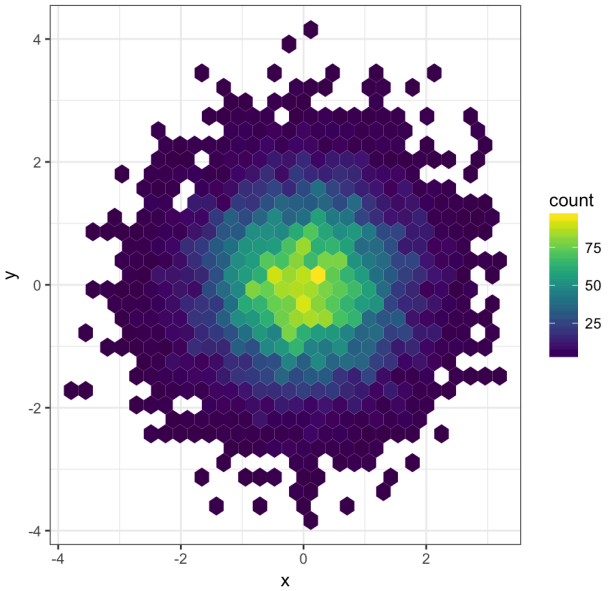


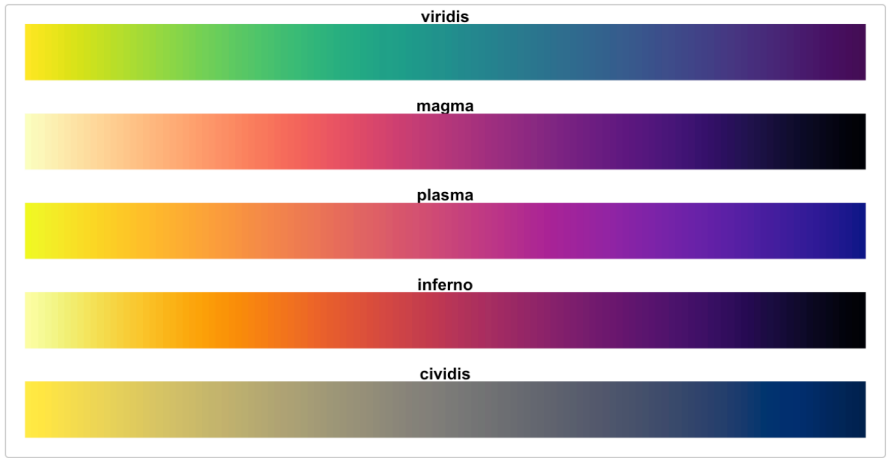

```
ggplot(data = iris, aes(x = Petal.Width, y = Petal.Length)) +
 geom_point(aes(color = Species),size = 5, alpha = 0.3) +
 geom_line(stat="smooth", method = "lm", alpha = 0.3) +
 geom_line(aes(group = Species, colour = Species), stat="smooth",
                method = "lm", lwd = 0.5) +
 theme_bw()
```

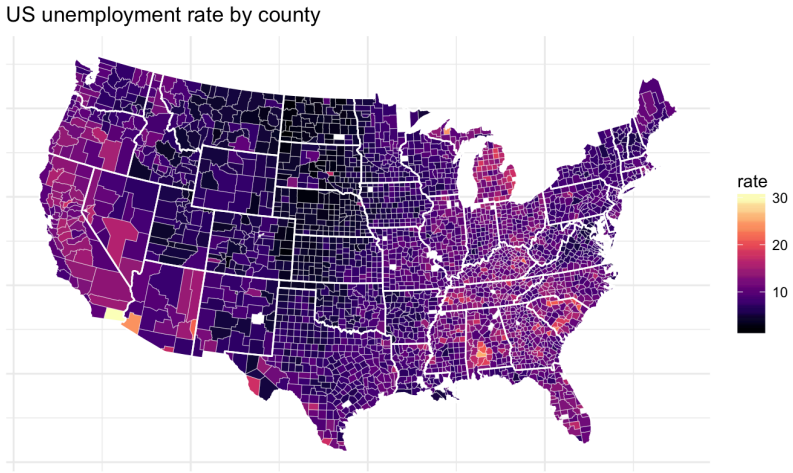
Working with palettes

Some quick ways to make your colours look cohesive

```
install.packages("viridis")

library(viridis)
```



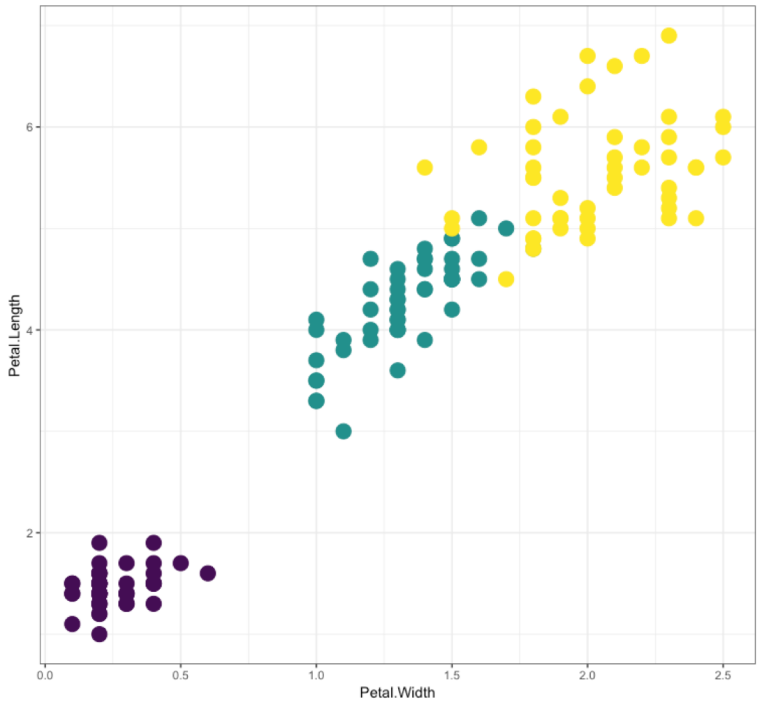


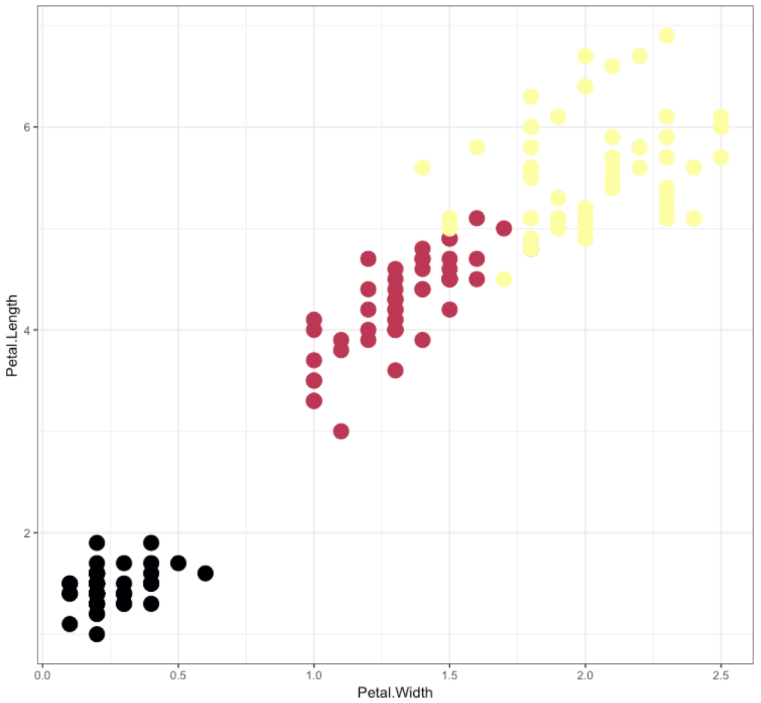


https://cran.r-project.org/web/packages/viridis/vignettes/intro-to-viridis.html

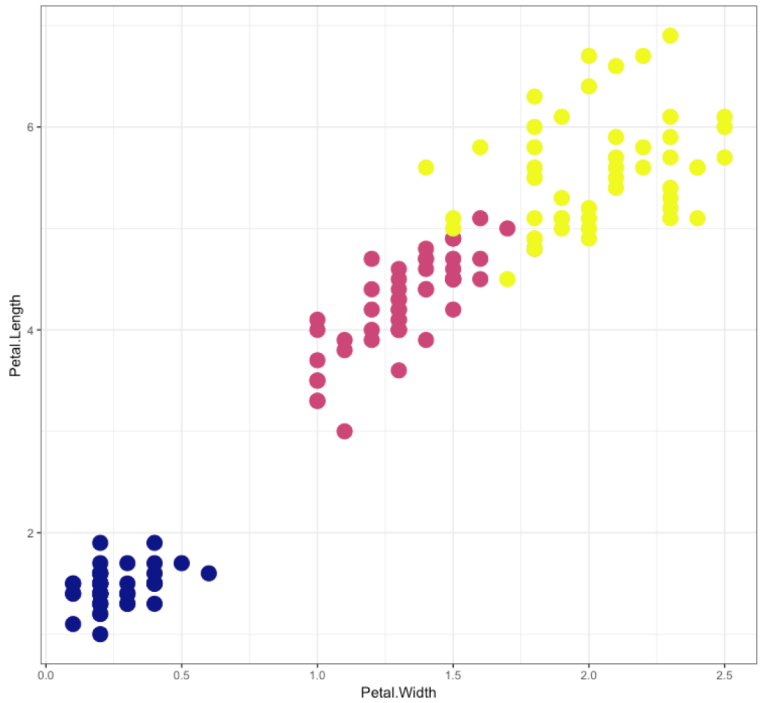
```
irisplot <- ggplot(data = iris,
                   aes(x = Petal.Width, y = Petal.Length, color = Species)) +
                       geom_point(size = 5) +
                       theme_bw()
```

```
irisplot  + scale_color_viridis(discrete = T)
```






```
scale_color_viridis(discrete = T,
                    option = "B")
```

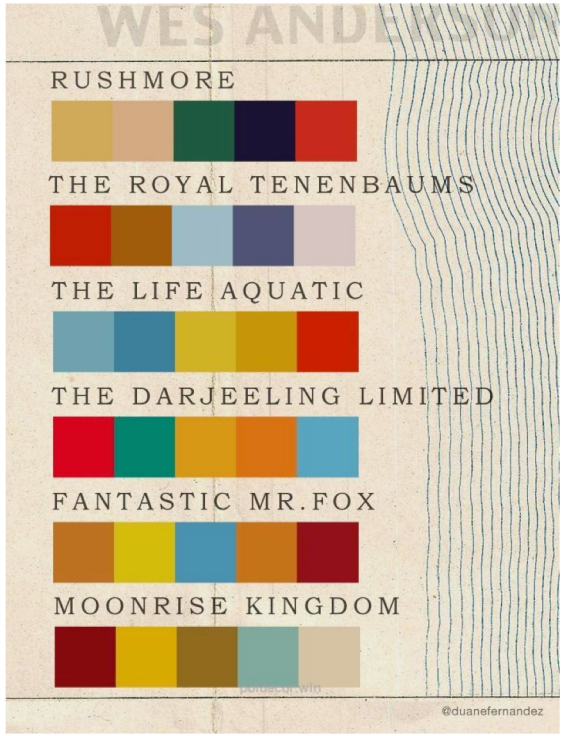


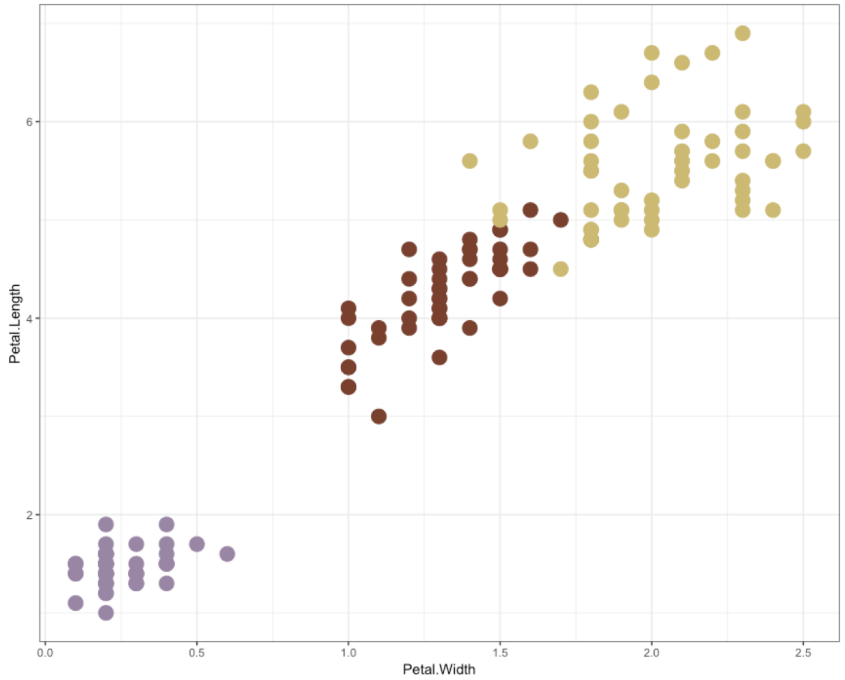

```
scale_color_viridis(discrete = T,
                    option = "C")
```

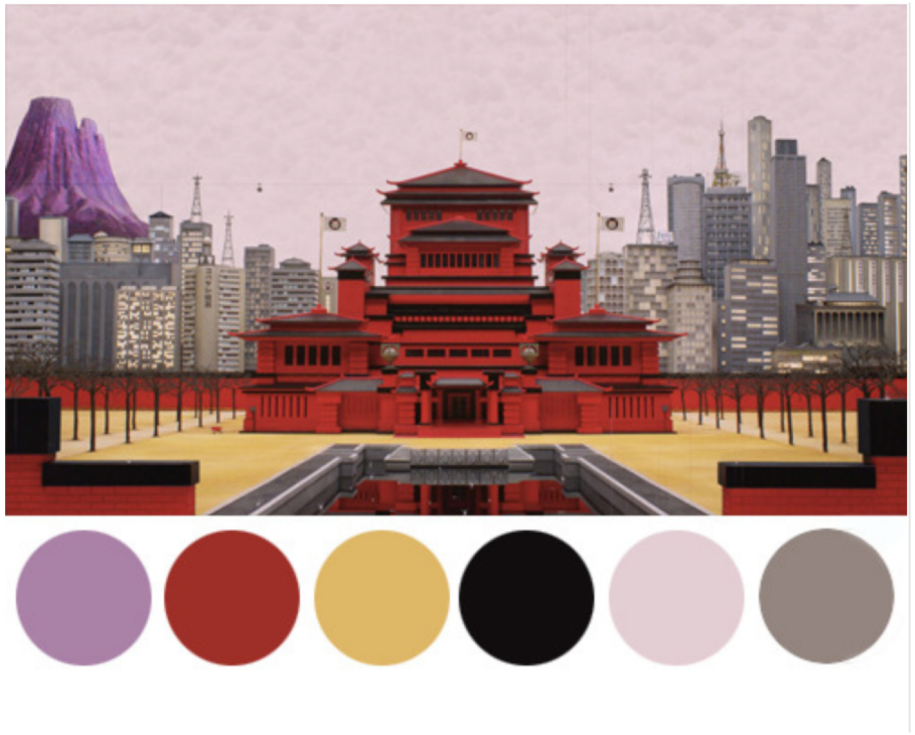
Working with palettes

Some quick ways to make your colours look cohesive

```
devtools::install_github("karthik/wesanderson")

library(wesanderson)
```



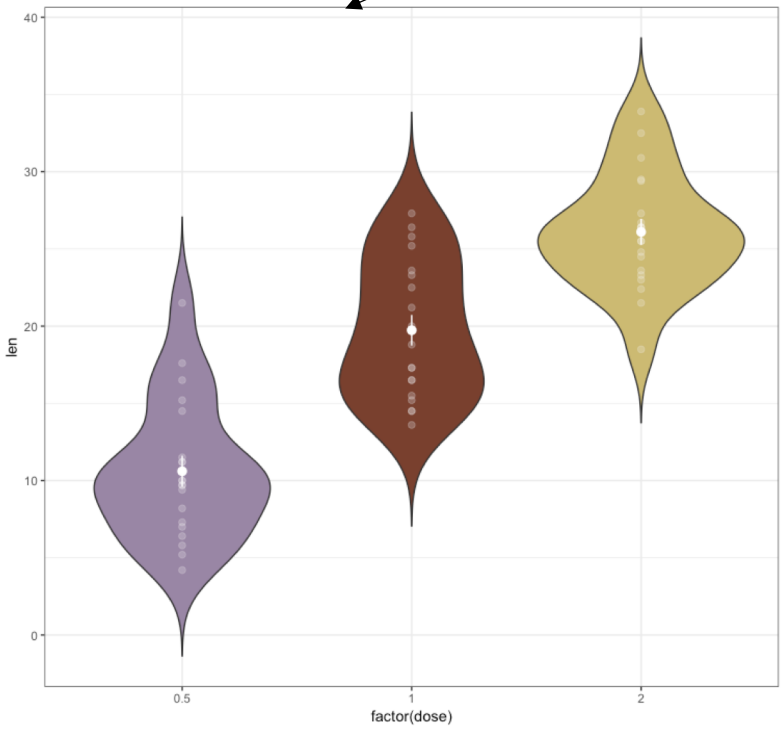


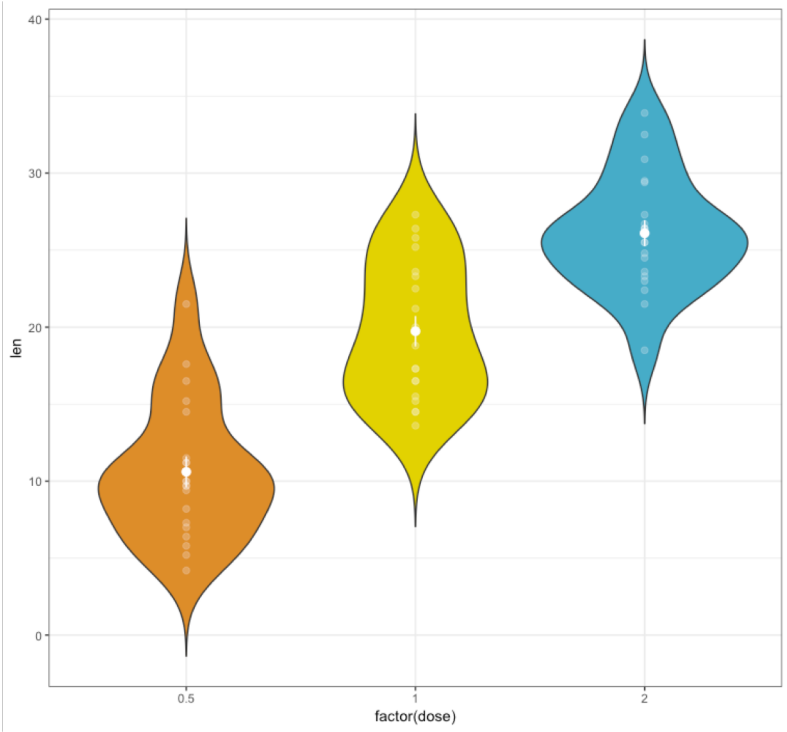


https://wesandersonpalettes.tumblr.com/ https://github.com/karthik/wesanderson

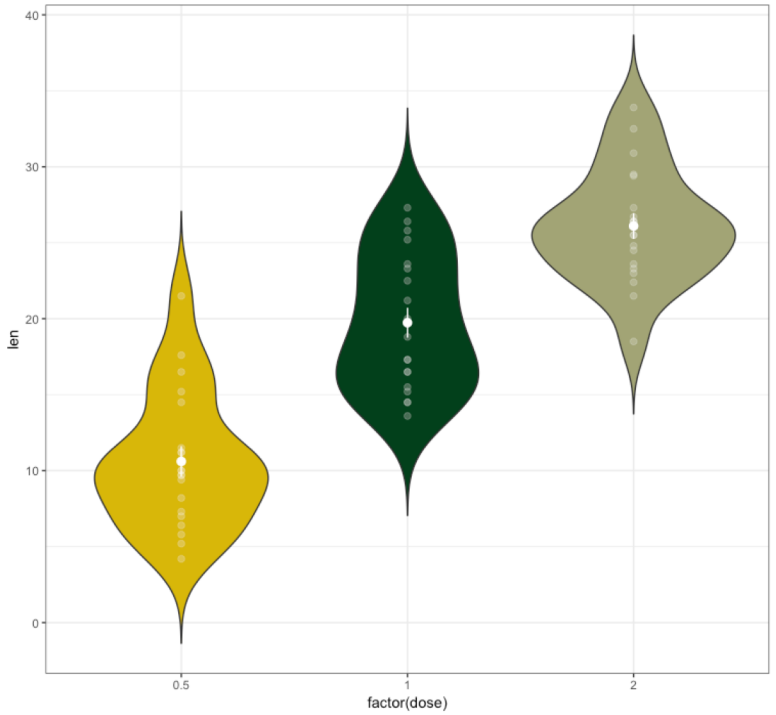
```
toothplot <- ggplot(data = ToothGrowth, aes(x = factor(dose), y = len, fill = factor(dose))) +
                    geom_violin(trim = F) +
                    geom_point(col = "white", size = 2, alpha = 0.2) +
                    stat_summary(fun.data = "mean_se", col = "white") +
                    theme_bw()
```

```
toothplot  +
scale_fill_manual(values = wes_palette("IsleofDogs1"))
```



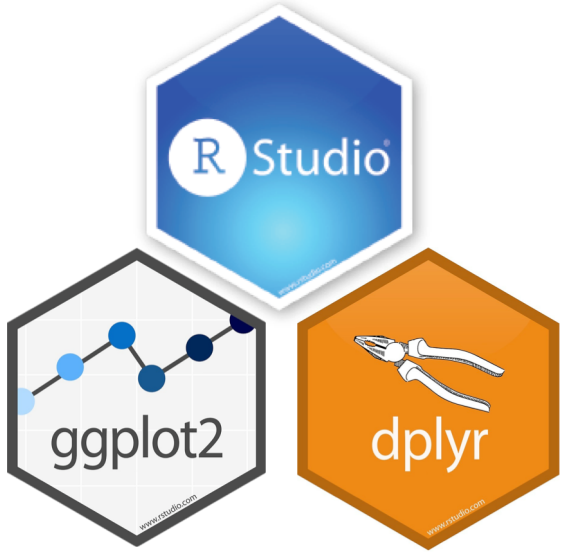


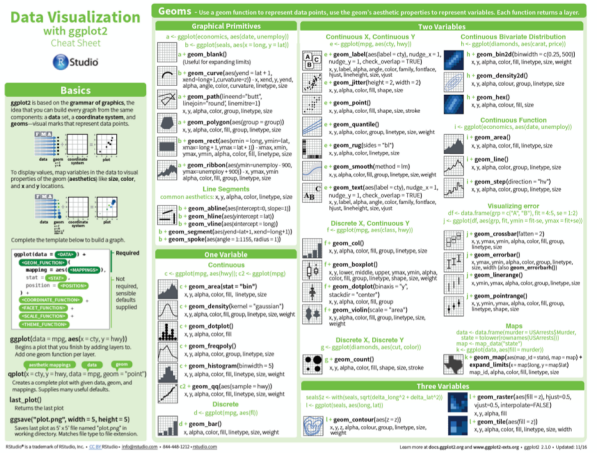
wes_palette("**FantasticFox1**")



wes_palette("**Cavalcanti1**"

Final words…





- This is only the beginning
- LOTS of google-able help!
- Always think about what the **key message** you are trying to graphically represent
- Less is ***more***

http://environmentalcomputing.net/plotting-with-ggplot/

Environmental Computing