INTRODUCTION TO GGPLOT2

Code Club - 5th May 2022

Olivia Johnson

OUTLINE FOR TODAY

- Quick recap of last week
- Basic structure of ggplot
- Scatter plot
- Distribution plots
- Practise plotting!

QUICK RECAP

- Summary statistics calculated across groupings
  - `df %>% group_by(species) %>% summarise(mean=mean())`
- Create new columns and overwrite existing ones
  - `df %>% mutate(new_column = what_you_want)`
- Replace character strings (equivalent of find and replace)
  - `str_replace_all(string, pattern, replacement)`
- Apply a function across specific columns (only in dplyr functions)
  - `across(.cols=X, .fns = ~function)`

```
> df
# A tibble: 150 x 7
   sample_number sepal_length sepal_width petal_length_percent_number petal_width date_collected       species
           <dbl>        <dbl>       <dbl>                       <dbl>       <dbl> <dttm>              <chr>
 1             1          5.1         3.5                         1.4         0.2 2022-04-06 00:00:00 setosa
 2             2          4.9         3                           1.4         0.2 2022-04-07 00:00:00 setosa
 3             3          4.7         3.2                         1.3         0.2 2022-04-09 00:00:00 setosa
 4             4          4.6         3.1                         1.5         0.2 2022-04-10 00:00:00 setosa
 5             5          5           3.6                         1.4         0.2 2022-04-11 00:00:00 setosa
 6             6          5.4         3.9                         1.7         0.4 2022-04-12 00:00:00 setosa
 7             7          4.6         3.4                         1.4         0.3 2022-04-13 00:00:00 setosa
 8             8          5           3.4                         1.5         0.2 2022-04-14 00:00:00 setosa
 9             9          4.4         2.9                        NA           0.2 2022-04-15 00:00:00 setosa
10            10          4.9        NA                           1.5        NA   2022-04-16 00:00:00 setosa
# … with 140 more rows
```

OUR DATASET

We have a nice clean dataset, now we want to visualise it!

BASIC STRUCTURE


```
ggplot(data = <DATA>, mapping = aes(<MAPPINGS>)) +
<GEOM_FUNCTION>()
```

Mappings – x, y, col, size etc

GEOM_FUNCTION – specify what type of plot

```
> df
# A tibble: 150 x 7
   sample_number sepal_length sepal_width petal_length_percent_number petal_width date_collected      species
           <dbl>        <dbl>       <dbl>                       <dbl>       <dbl> <dttm>              <chr>
 1             1          5.1         3.5                         1.4         0.2 2022-04-06 00:00:00 setosa
 2             2          4.9         3                           1.4         0.2 2022-04-07 00:00:00 setosa
 3             3          4.7         3.2                         1.3         0.2 2022-04-09 00:00:00 setosa
 4             4          4.6         3.1                         1.5         0.2 2022-04-10 00:00:00 setosa
 5             5          5           3.6                         1.4         0.2 2022-04-11 00:00:00 setosa
 6             6          5.4         3.9                         1.7         0.4 2022-04-12 00:00:00 setosa
 7             7          4.6         3.4                         1.4         0.3 2022-04-13 00:00:00 setosa
 8             8          5           3.4                         1.5         0.2 2022-04-14 00:00:00 setosa
 9             9          4.4         2.9                        NA           0.2 2022-04-15 00:00:00 setosa
10            10          4.9        NA                           1.5        NA   2022-04-16 00:00:00 setosa
# … with 140 more rows
```

WHAT DO WE WANT TO PLOT

```
> df
# A tibble: 150 x 7
   sample_number sepal_length sepal_width petal_length_percent_number petal_width date_collected       species
           <dbl>        <dbl>       <dbl>                       <dbl>       <dbl> <dttm>              <chr>
 1             1          5.1         3.5                         1.4         0.2 2022-04-06 00:00:00 setosa
 2             2          4.9         3                           1.4         0.2 2022-04-07 00:00:00 setosa
 3             3          4.7         3.2                         1.3         0.2 2022-04-09 00:00:00 setosa
 4             4          4.6         3.1                         1.5         0.2 2022-04-10 00:00:00 setosa
 5             5          5           3.6                         1.4         0.2 2022-04-11 00:00:00 setosa
 6             6          5.4         3.9                         1.7         0.4 2022-04-12 00:00:00 setosa
 7             7          4.6         3.4                         1.4         0.3 2022-04-13 00:00:00 setosa
 8             8          5           3.4                         1.5         0.2 2022-04-14 00:00:00 setosa
 9             9          4.4         2.9                        NA           0.2 2022-04-15 00:00:00 setosa
10            10          4.9        NA                           1.5        NA   2022-04-16 00:00:00 setosa
# … with 140 more rows
```

WHAT DO WE WANT TO PLOT

```
> df
# A tibble: 150 x 7
   sample_number sepal_length sepal_width petal_length_percent_number petal_width date_collected       species
           <dbl>        <dbl>       <dbl>                       <dbl>       <dbl> <dttm>              <chr>
 1             1          5.1         3.5                         1.4         0.2 2022-04-06 00:00:00 setosa
 2             2          4.9         3                           1.4         0.2 2022-04-07 00:00:00 setosa
 3             3          4.7         3.2                         1.3         0.2 2022-04-09 00:00:00 setosa
 4             4          4.6         3.1                         1.5         0.2 2022-04-10 00:00:00 setosa
 5             5          5           3.6                         1.4         0.2 2022-04-11 00:00:00 setosa
 6             6          5.4         3.9                         1.7         0.4 2022-04-12 00:00:00 setosa
 7             7          4.6         3.4                         1.4         0.3 2022-04-13 00:00:00 setosa
 8             8          5           3.4                         1.5         0.2 2022-04-14 00:00:00 setosa
 9             9          4.4         2.9                        NA           0.2 2022-04-15 00:00:00 setosa
10            10          4.9        NA                           1.5        NA   2022-04-16 00:00:00 setosa
# … with 140 more rows
```

WHAT DO WE WANT TO PLOT

SCATTER PLOT

- Start by looking at sepal length x sepal width

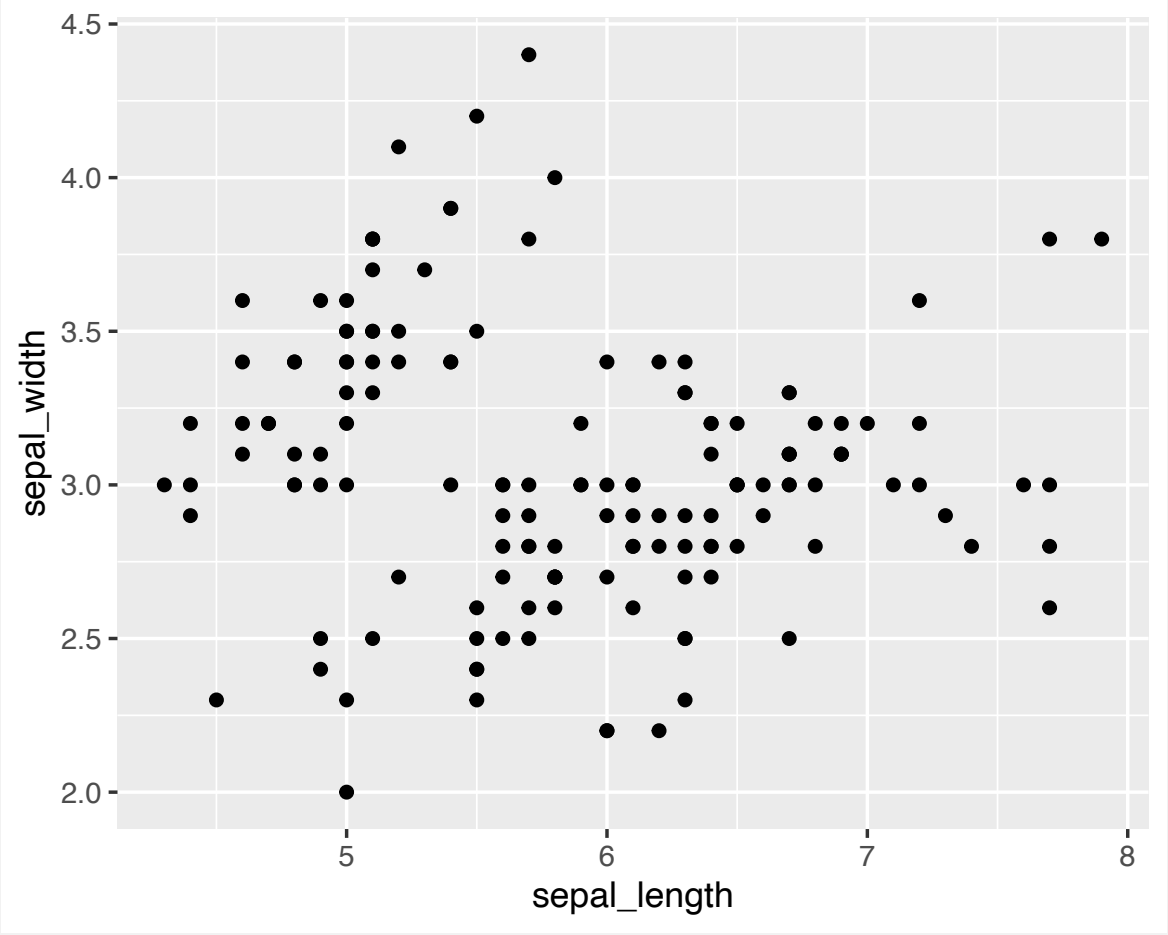
```
ggplot(data = df, aes(x=sepal_length, y=sepal_width)) +
geom_point()
```

SCATTER PLOT

- Start by looking at sepal length x sepal width

```
ggplot(data = df, aes(x=sepal_length,
y=sepal_width)) +
geom_point()
```

- No clear trend, but we could see how species affects the data



SCATTER PLOT

- Start by looking at sepal length x sepal width

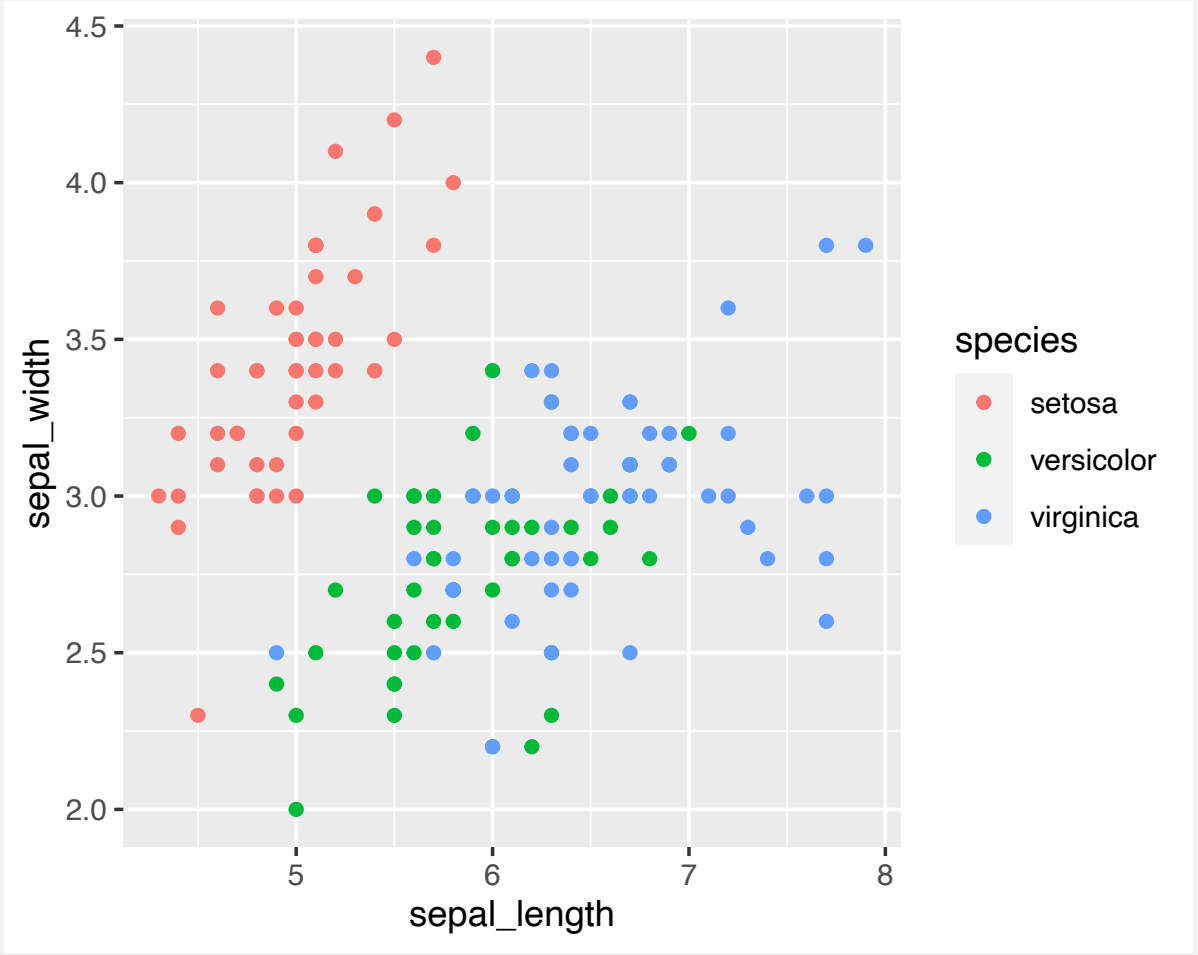
```
ggplot(data = df, aes(x=sepal_length, y=sepal_width, col = species)) +
geom_point()
```

SCATTER PLOT

- Start by looking at sepal length x sepal width

```
ggplot(data = df, aes(x=sepal_length,
y=sepal_width, col = species)) +
geom_point()
```

- Looks like increase in sepal length with sepal width, when look on species level.



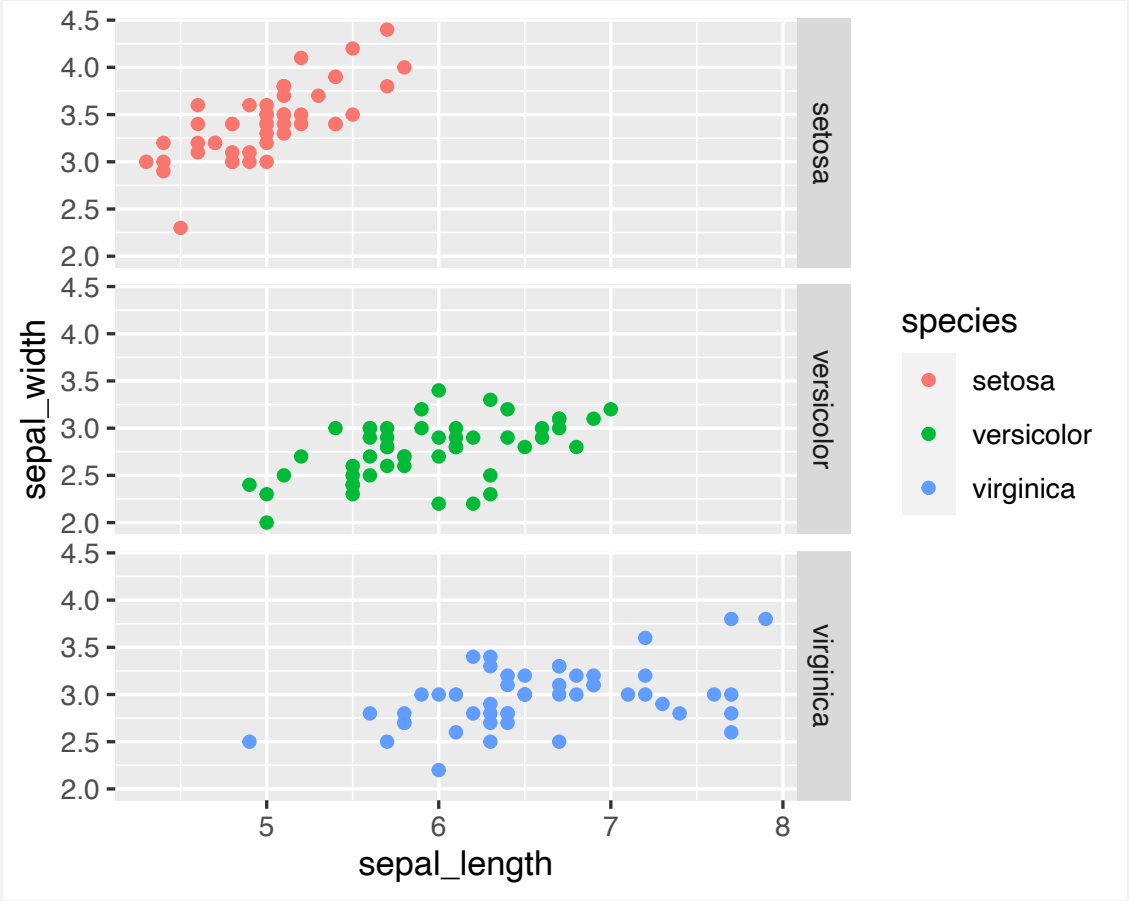
FACETS

- Can use facets to separate out data by a variable.
- Can use facet_wrap or facet grid

```
plot = ggplot(data = df, aes(x=sepal_length, y=sepal_width, col =
species)) +

geom_point() +

facet_grid(“species”)
```

FACETS

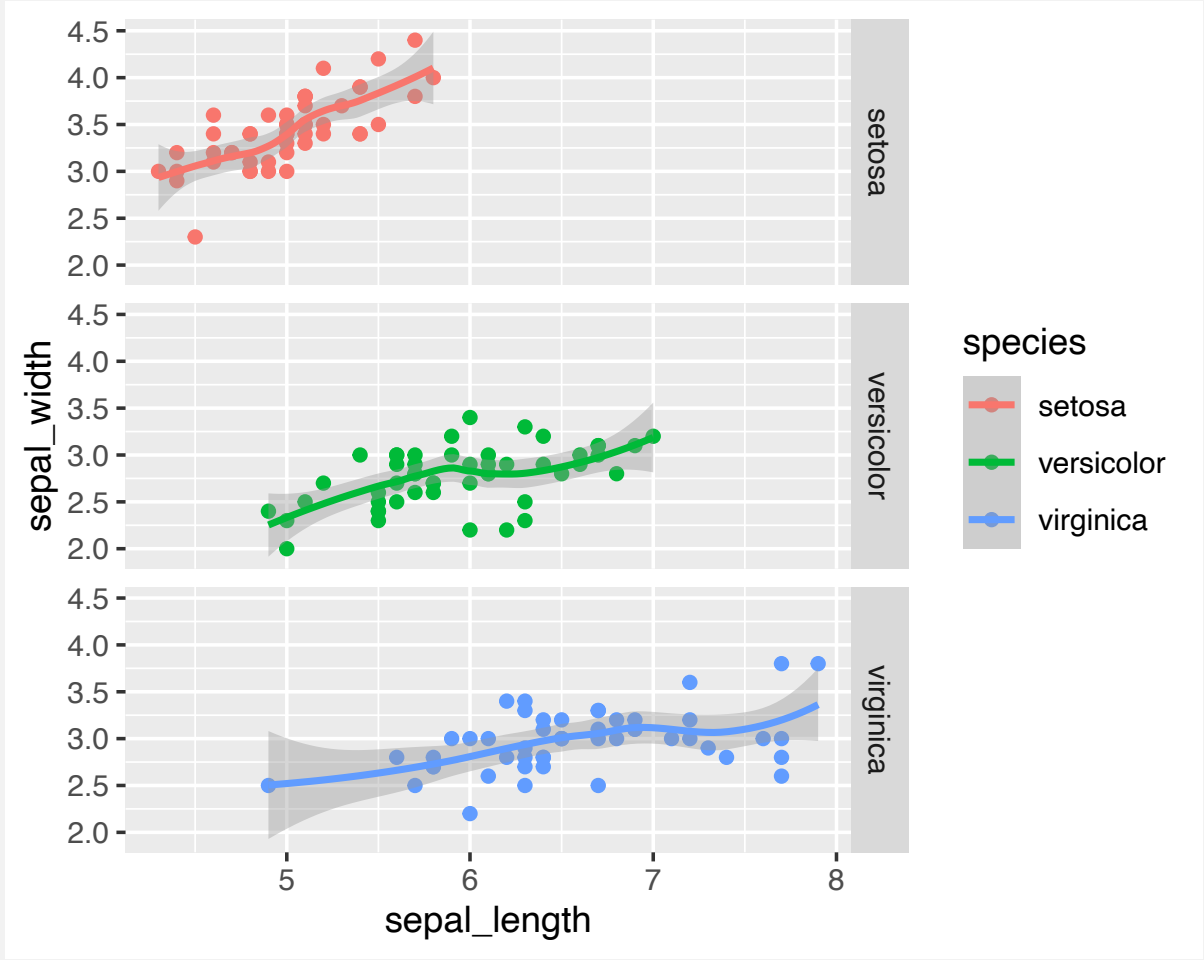
```
plot +

facet_grid("species")
```



TREND LINE

```
plot +
geom_smooth()+
facet_grid("species")
```

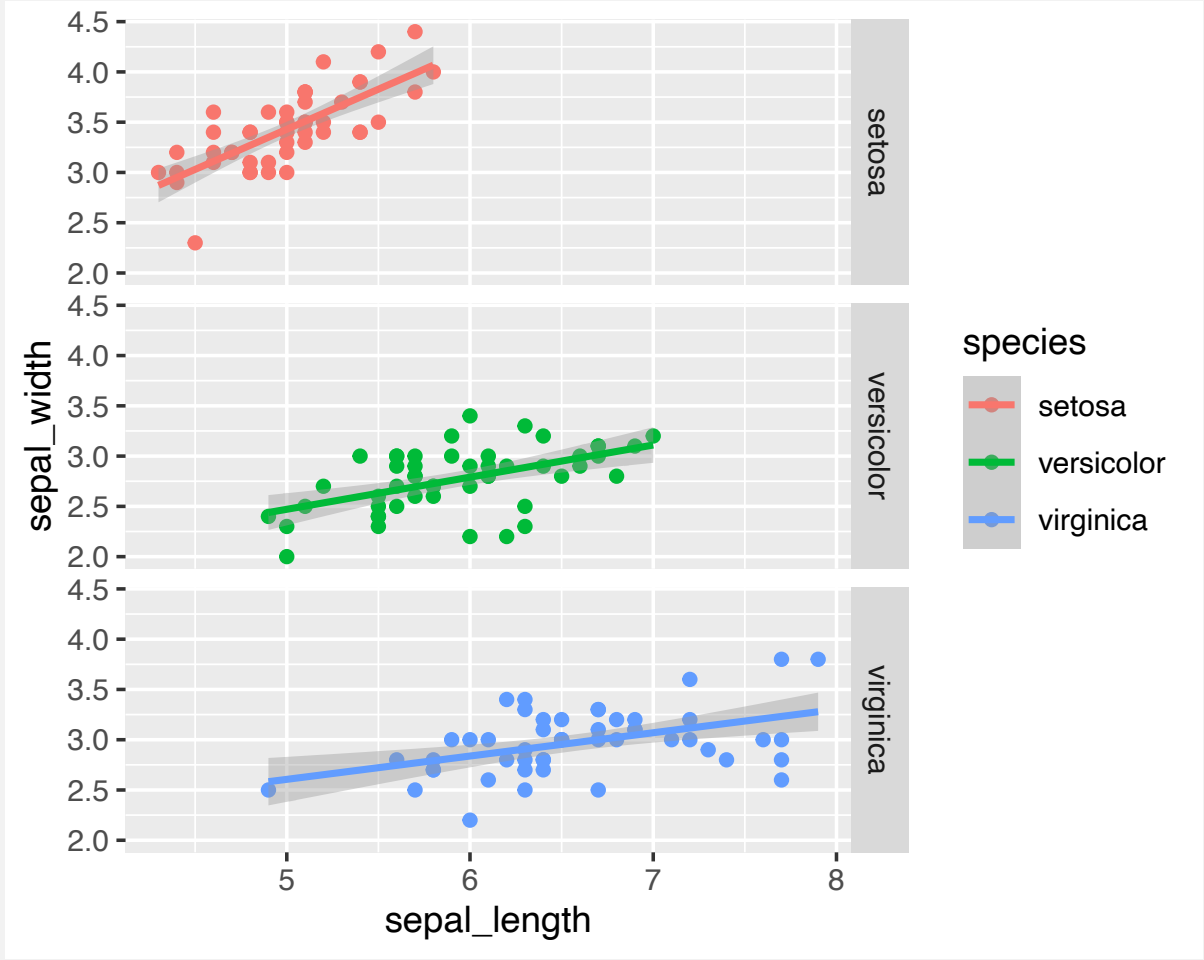
- Not a straight line, but can specify in the geom function.



TREND LINE

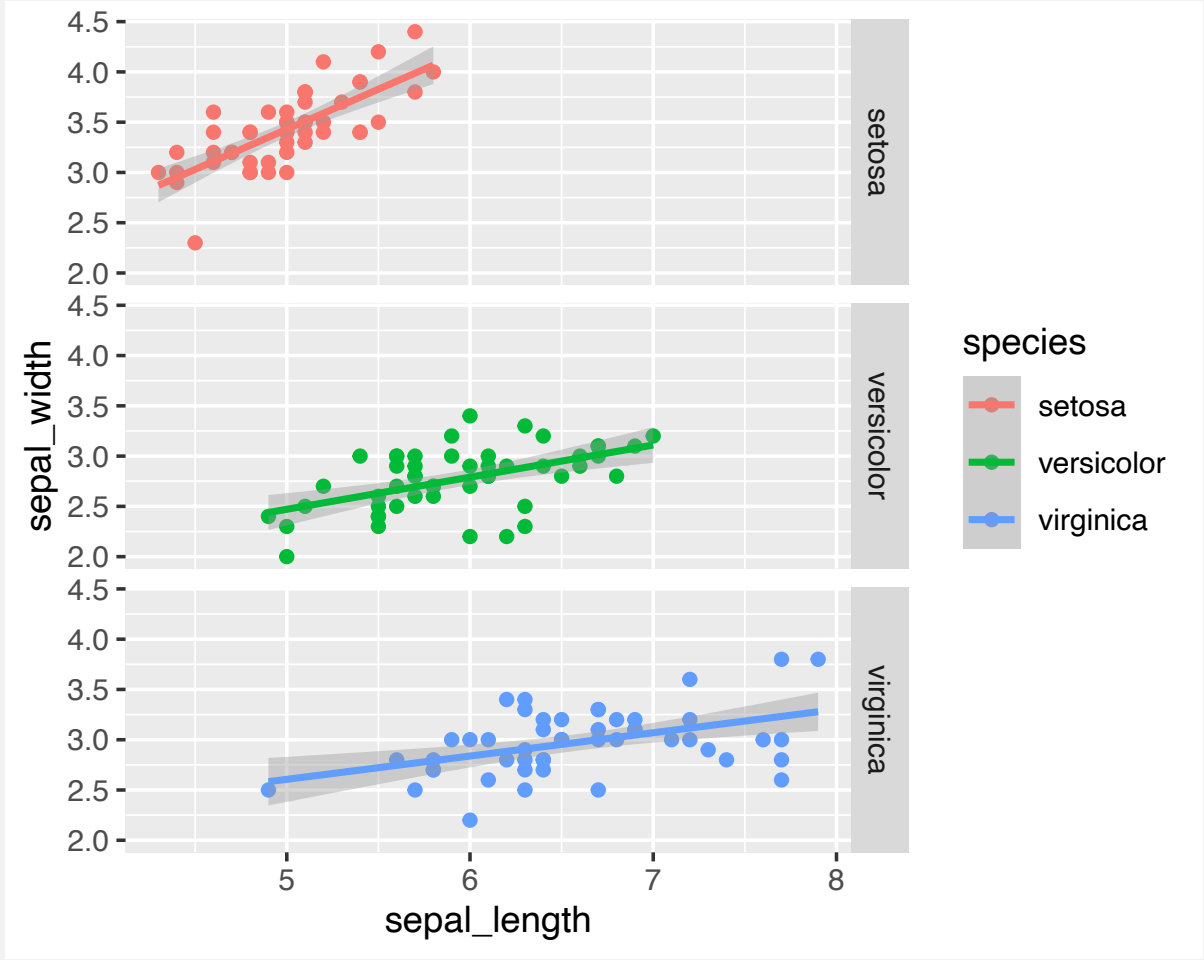
```
plot = plot +

geom_smooth(method="lm")+

facet_grid("species")
```

- Not a straight line, but can specify in the geom function.



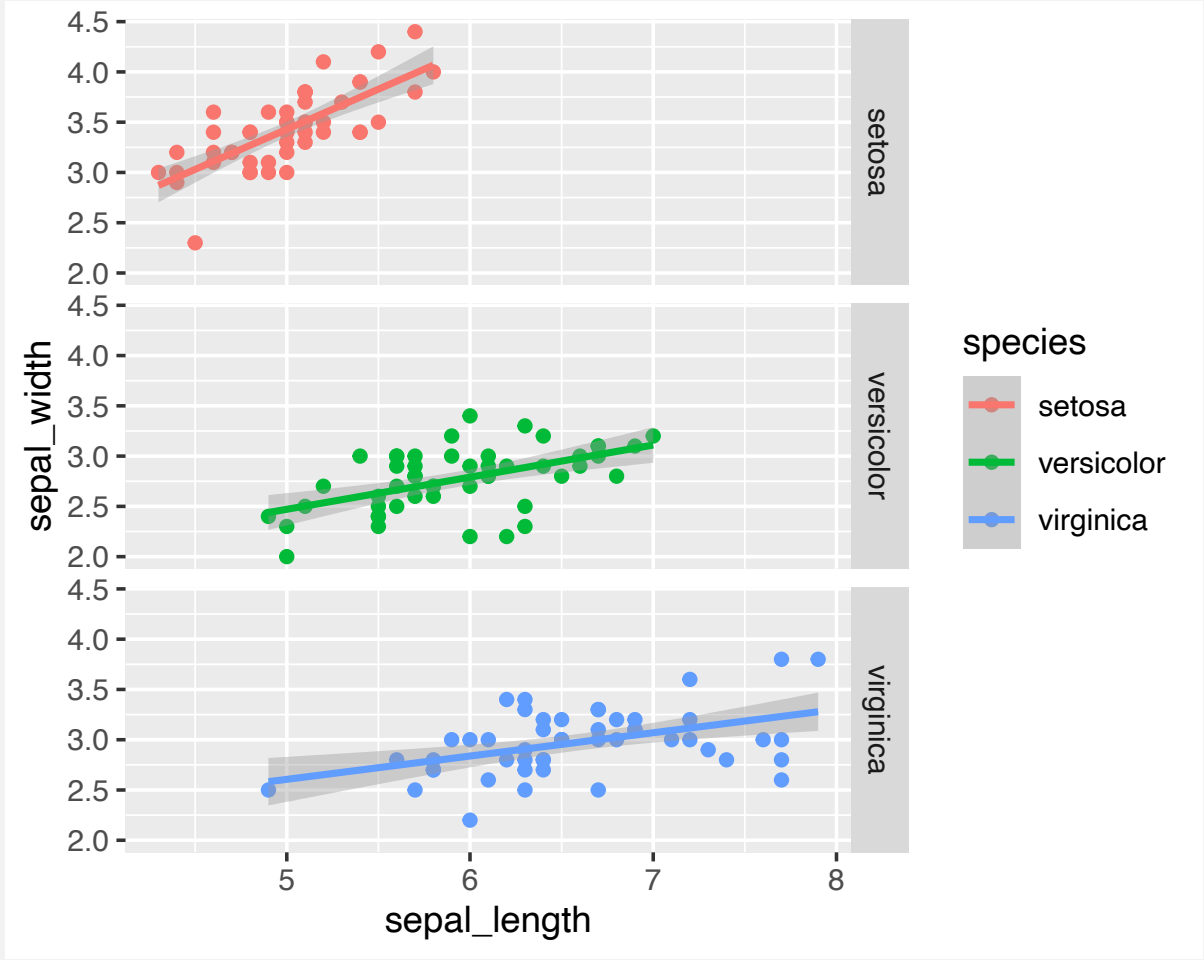
TIDY PLOT

- Relabel axes
- Remove redundant species label



TIDY PLOT

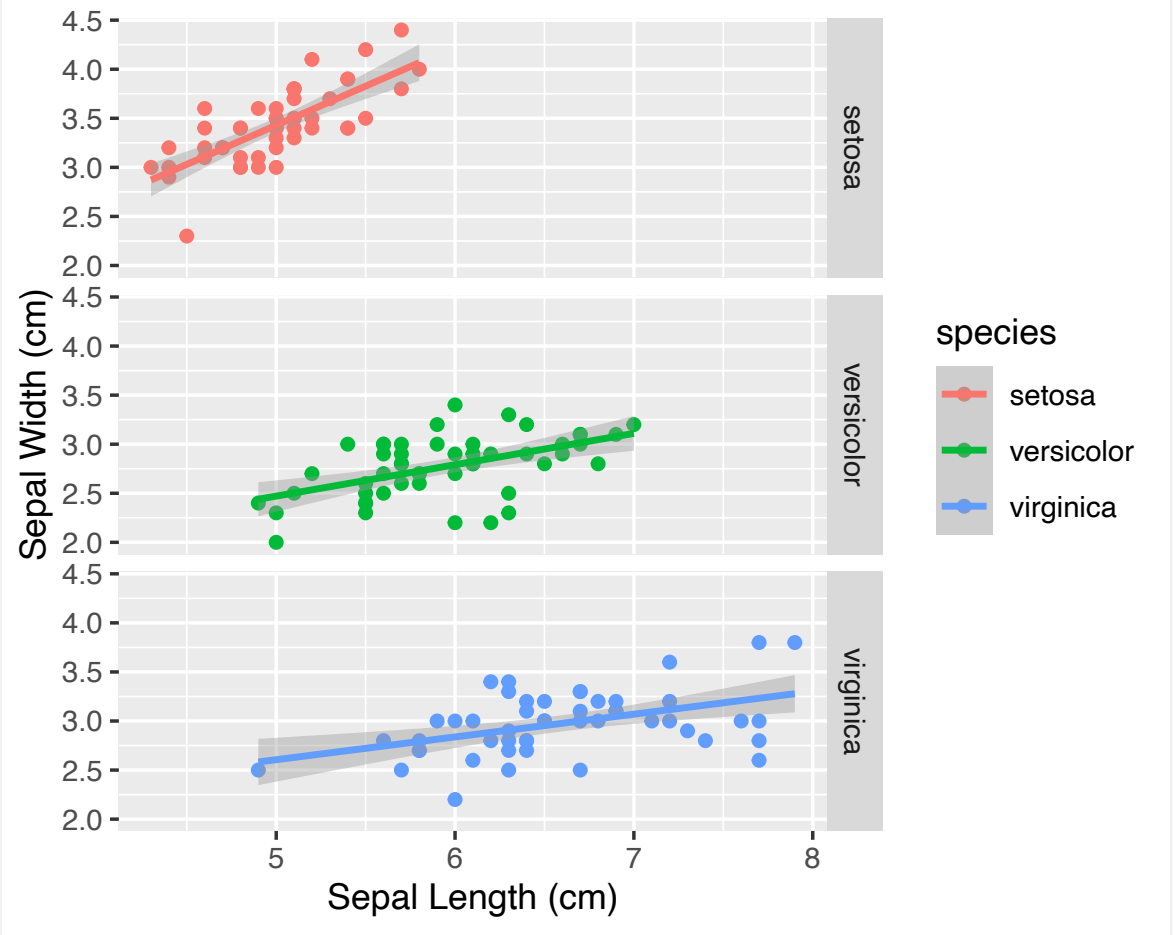
- Relabel axes
- Remove redundant species label



RELABEL AXES

- Can use `labs(x="label", y="label")`

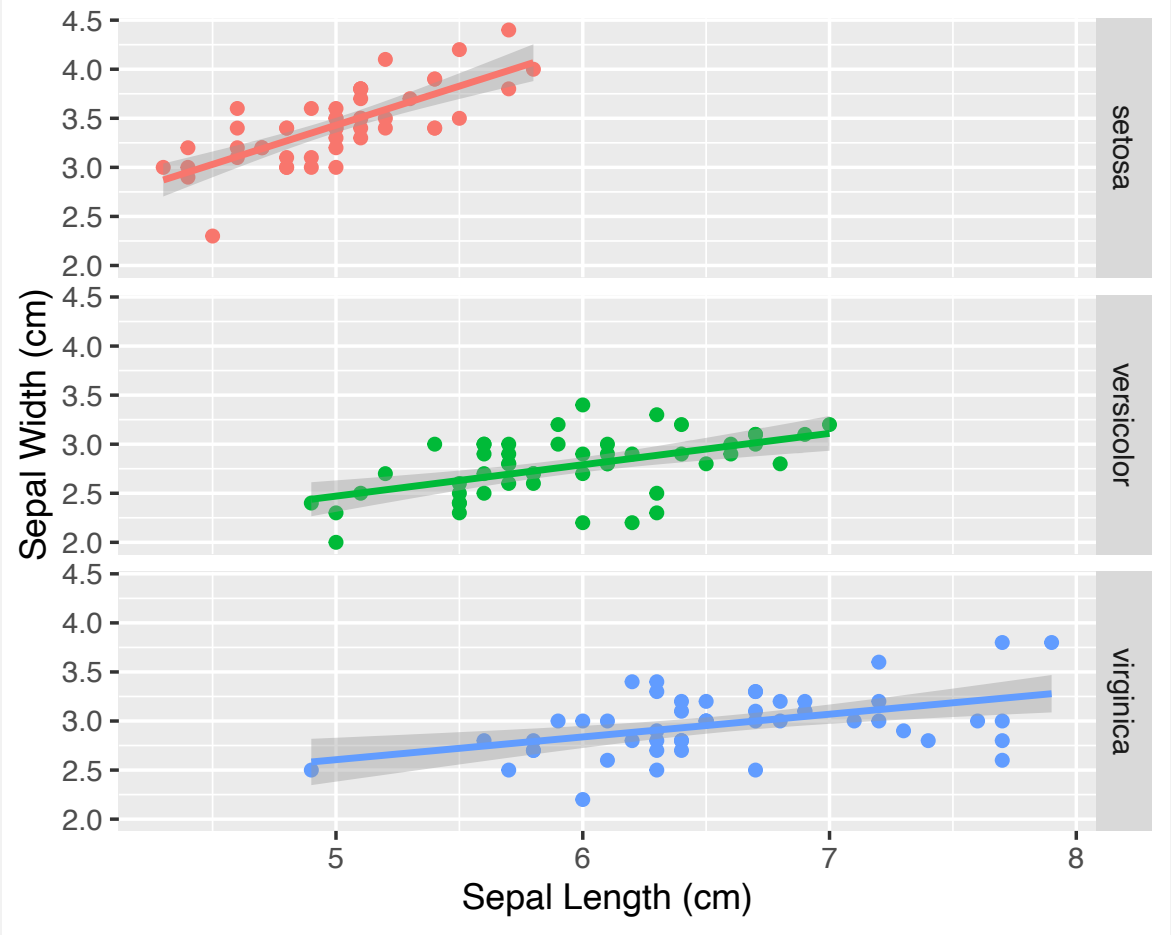
```
plot +
labs(x="Sepal Length (cm)", y="Sepal Width (cm)")
```



REMOVE REDUNDANT LABELS

- Don't need key on left edge

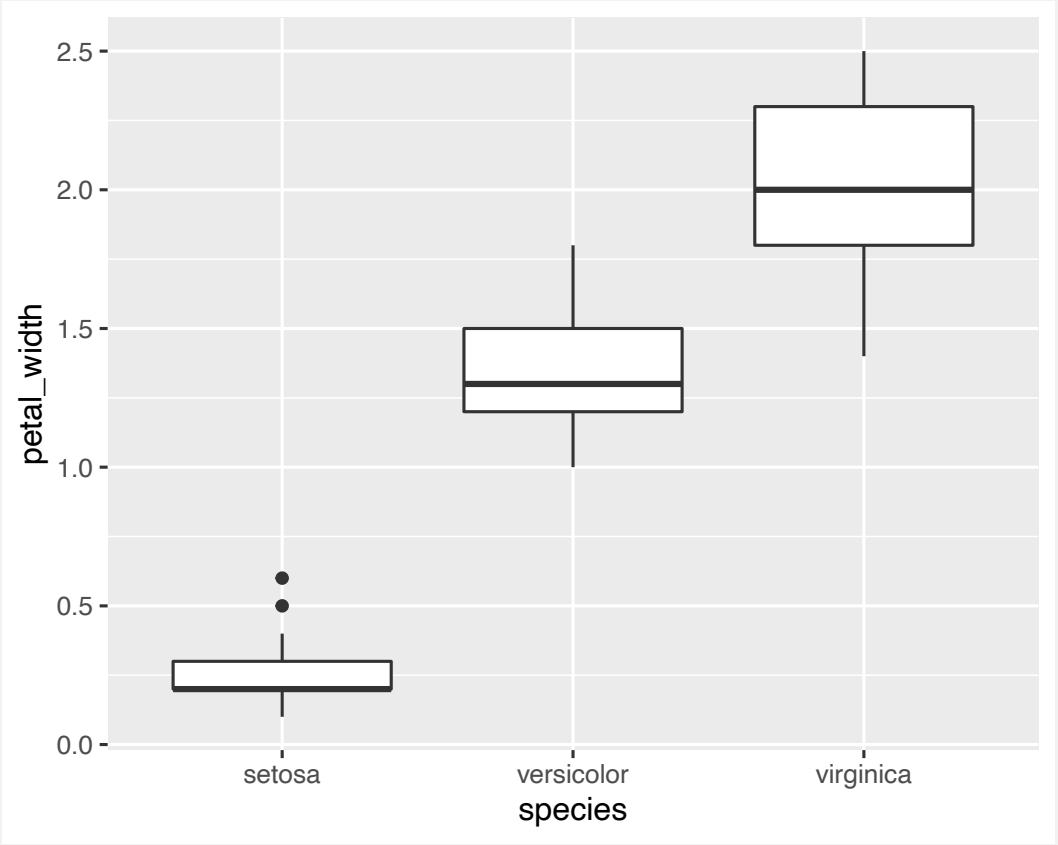
```
plot +
labs(x="Sepal Length (cm)", y="Sepal
Width (cm)")+
theme(legend.position="none")
```



DISTRIBUTIONS

- Good to look at distribution of a specific variable

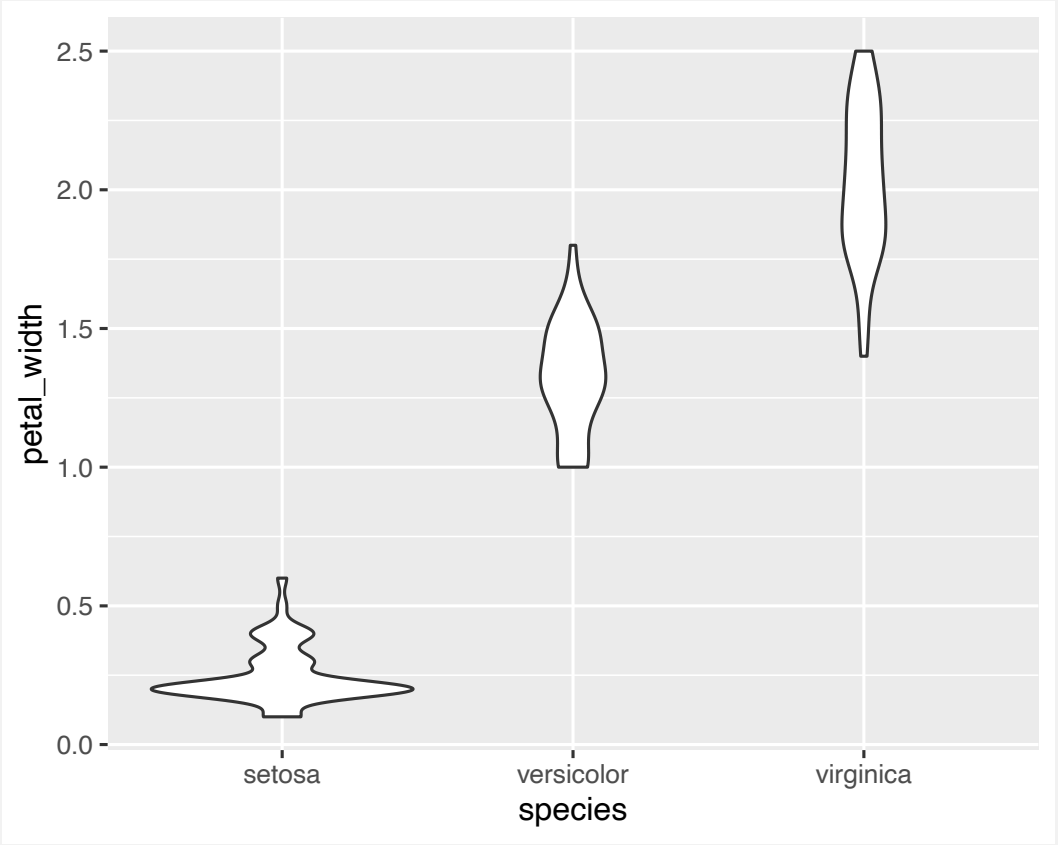
```
ggplot(df)+
   geom_boxplot(aes(x=species, y=petal_width))
```



DISTRIBUTIONS

- Good to look at distribution of a specific variable

```
ggplot(df)+
  geom_boxplot(aes(x=species, y=petal_width))


ggplot(df)+
  geom_violin(aes(x=species, y=petal_width))
```



SAVING PLOTS

```
ggsave("plot.jpg", plot = plot, width = 5, height = 5)
```

SUMMARY

- Create scatter plots
- Plot distributions
- Facet data
- Tidy plots

Data Visualization with ggplot2 : : CHEAT SHEET

Basics

ggplot2 is based on the **grammar of graphics**, the idea that you can build every graph from the same components: a **data** set, a **coordinate system**, and geoms—visual marks that represent data points.

To display values, map variables in the data to visual properties of the geom (**aesthetics**) like **size**, **color**, and **x** and **y** locations.

Complete the template below to build a graph.

required

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

Not required, sensible defaults supplied

ggplot(data = mpg, **aes**(x = cty, y = hwy)) Begins a plot that you finish by adding layers to. Add one geom function per layer.

aesthetic mappings · data · geom

qplot(x = cty, y = hwy, data = mpg, geom = "point") Creates a complete plot with given data, geom, and mappings. Supplies many useful defaults.

last_plot() Returns the last plot

ggsave("plot.png", width = 5, height = 5) Saves last plot as 5' x 5' file named "plot.png" in working directory. Matches file type to file extension.

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

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

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

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

Cheat sheets available online

https://www.rstudio.com/resources/cheatsheets/