Universidade Federal do Rio Grande do Norte
Instituto Metrópole Digital
**IMD0601 - Bioestatística**

# Visualização dos dados em R

Prof. Dr. Tetsu Sakamoto
Instituto Metrópole Digital - UFRN
Sala A224, ramal 182
Email: tetsu@imd.ufrn.br

# Baixe a aula (e os arquivos)

- Para aqueles que não clonaram o repositório:

```
> git clone https://github.com/tetsufmbio/IMD0601.git
```

- Para aqueles que já tem o repositório local:

```
> cd /path/to/IMD0601
```

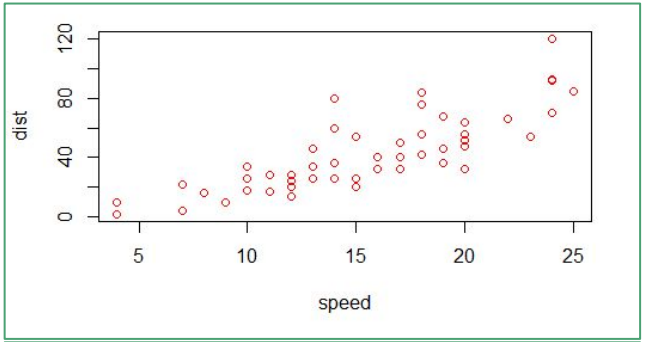
```
> git pull
```

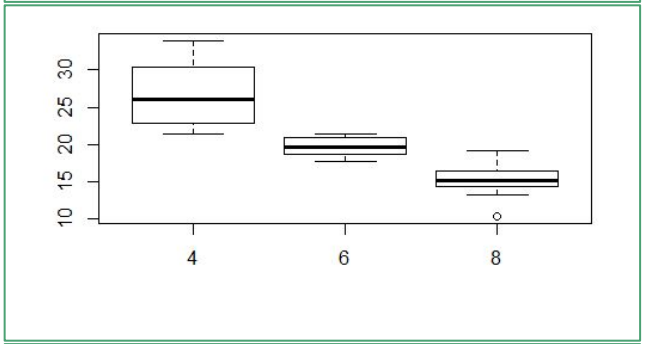
# R Base Graphics

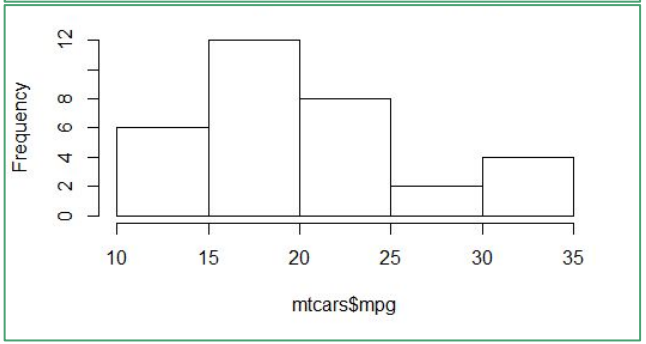
```
plot(x = cars$speed, y = cars$dist,
xlab = "Speed", ylab = "Stopping
Distance", col = 2)
```

```
boxplot(formula = mpg~cyl, data =
mtcars)
```

```
hist(mtcars$mpg)
```



# ggplot2

Hadley Wickham

“The Grammar of Graphics”

Adiciona camadas nos gráficos para melhor visualização dos dados;

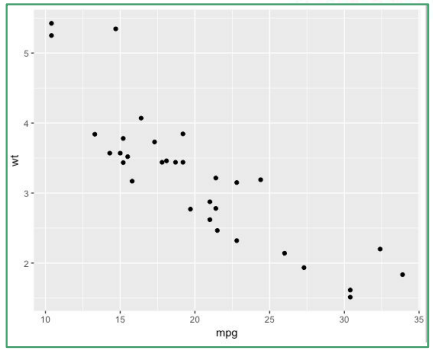
```
library(ggplot2)
```

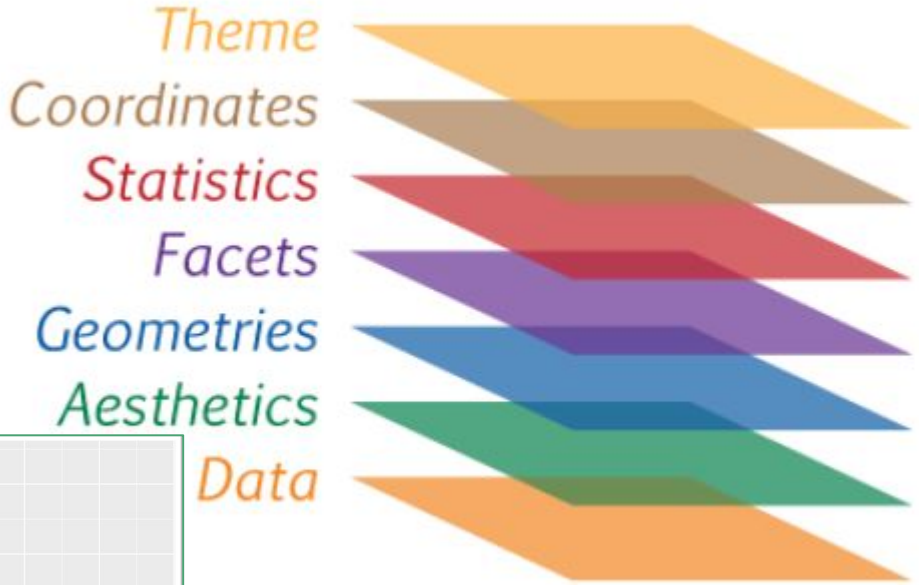
# ggplot2

**Geometries**

Camada que indica a forma como os dados devem ser apresentados no gráfico.

```
ggplot(mtcars, aes(x=mpg, y=wt)) + geom_point()
```



# Iris dataset

```
data(iris)

str(iris)

'data.frame':  150 obs. of  5 variables:
 $ Sepal.Length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
 $ Sepal.Width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
 $ Petal.Length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
 $ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
 $ Species     : Factor w/ 3 levels "setosa","versicolor",..: 1 1
1 1 1 1 1 1 1 1 ...
```

# ggplot2

**Exercício**

Adicione uma coluna na tabela **iris** que corresponda a um identificador único de cada observação.

```
> iris$Flower <- 1:nrow(iris)
```

Crie uma tabela onde as variáveis Length e Width estejam cada uma em uma coluna, como abaixo.

```
  Species Flower  part Length Width
1  setosa      1 Petal    1.4   0.2
2  setosa      1 Sepal    5.1   3.5
...
```

# ggplot2

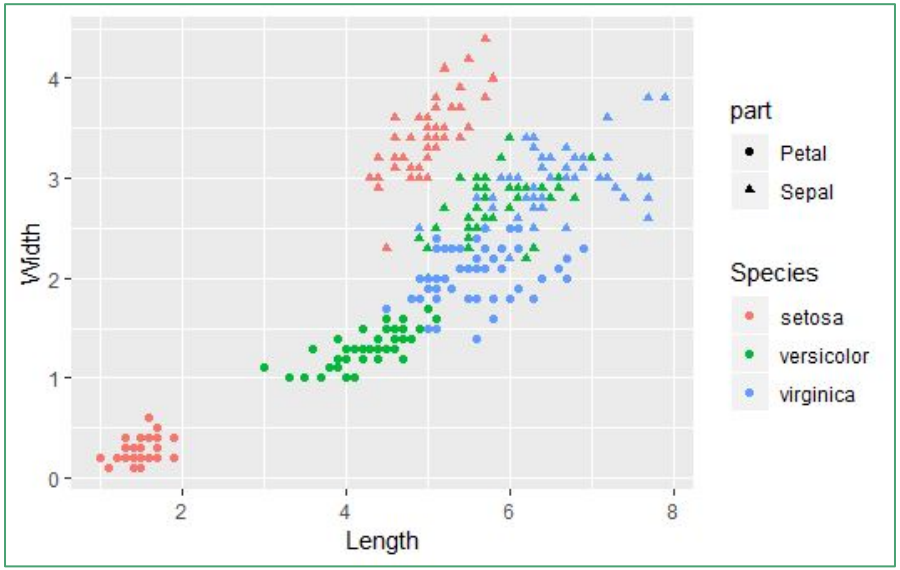
**Exercício**

```
> library(tidyr)

> iris$Flower <- 1:nrow(iris)

> iris.wide <- gather(iris, part_measure, val, -Species, -Flower )

> iris.wide <- separate(iris.wide, part_measure,
c("part","measure"))

> iris.wide <- spread(iris.wide, measure, val)
```

# ggplot2

## Exercício

Plote um gráfico abaixo usando o ggplot2:

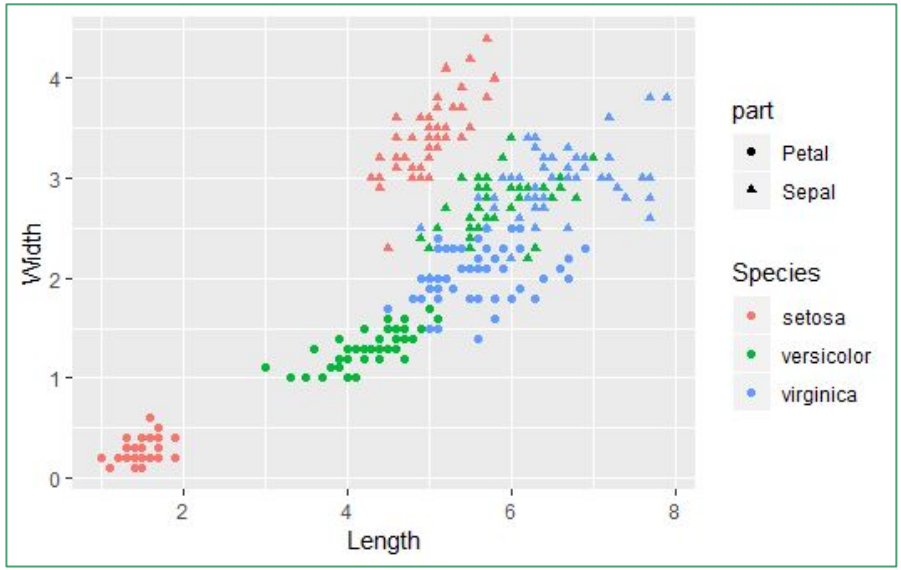


# ggplot2

**Exercício**

Plote um gráfico abaixo usando o ggplot2:

```
> ggplot(iris.wide, aes(Length, Width, col = Species, shape = part)) + geom_point()
```



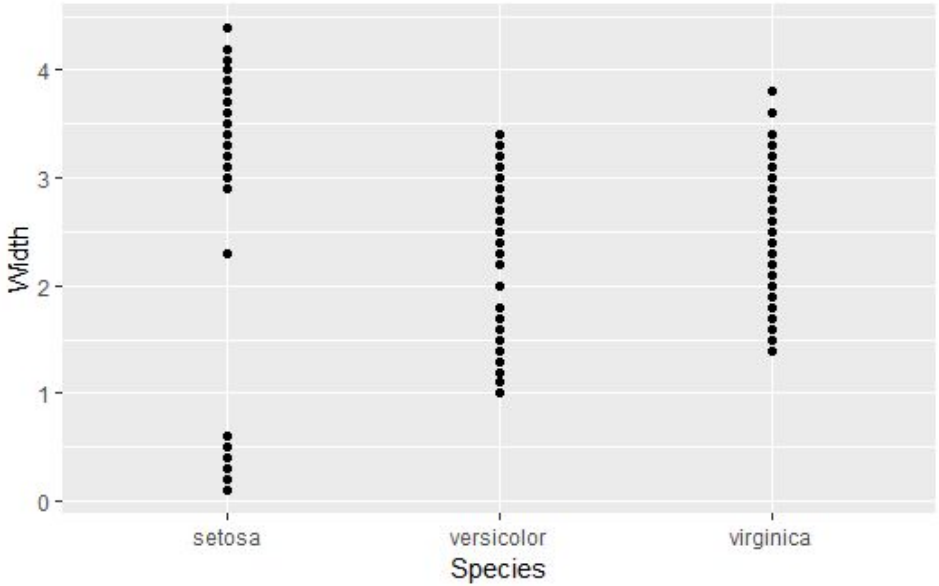
# Parâmetros típicos da estética

- **x** ➜ posição no eixo x;
- **y** ➜ posição no eixo y;
- **col** ➜ cor dos pontos, ou de outras formas;
- **fill** ➜ cor a ser preenchido;
- **size** ➜ diâmetro do ponto, largura da linha;
- **alpha** ➜ transparência;
- **linetype** ➜ padrão de tracejamento da linha;
- **labels** ➜ texto no gráfico;
- **shape** ➜ formas;

# ggplot2

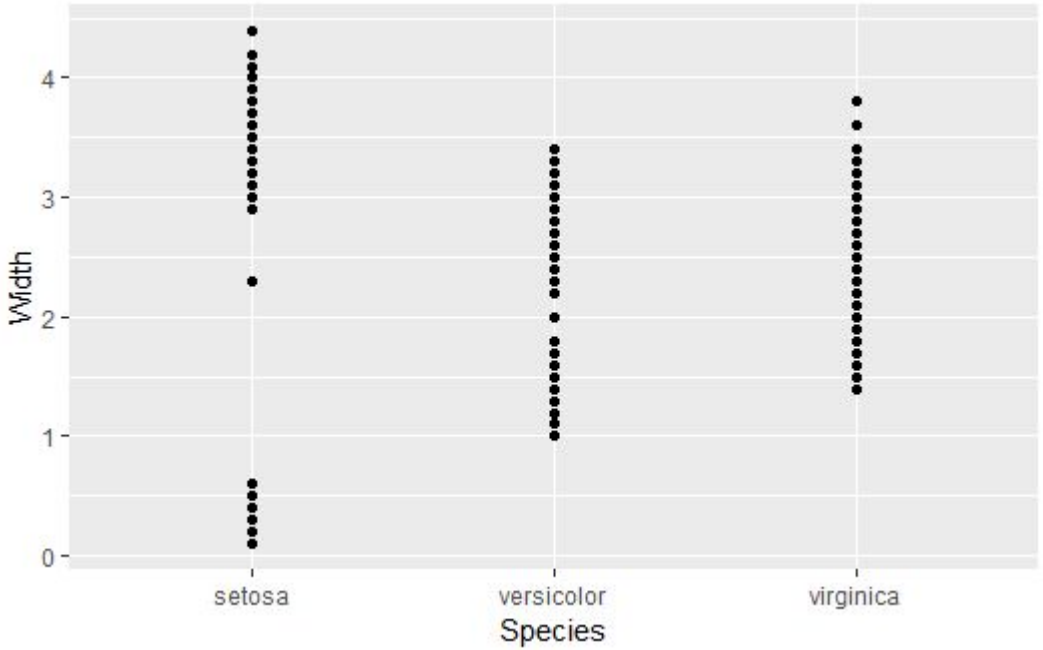
## Exercício

Plote um gráfico abaixo usando o ggplot2:



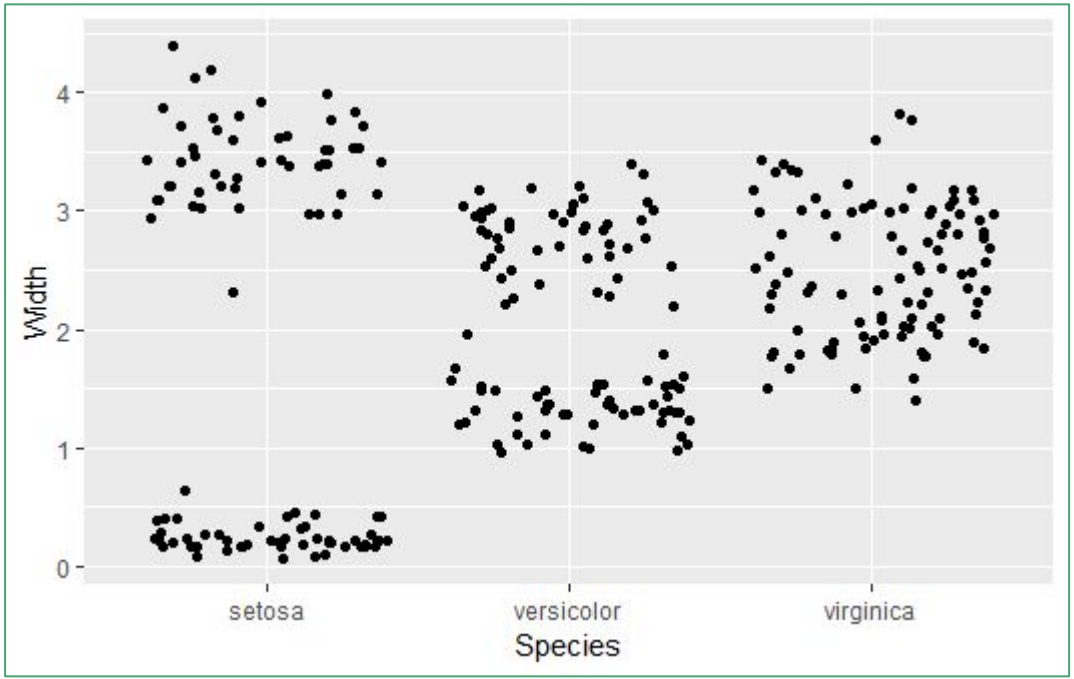
# ggplot

```
> ggplot(iris.wide, aes(Species, Width)) + geom_point()
```



# ggplot

```
ggplot(iris.wide, aes(Species, Width)) + geom_point(position =
"jitter")
```

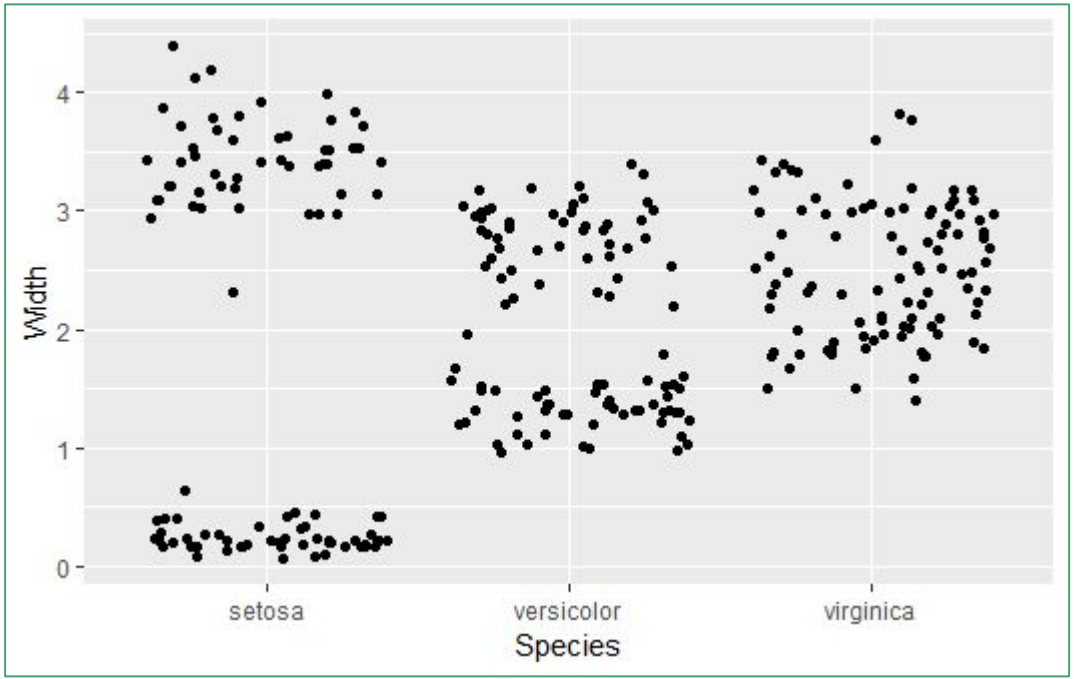



position:

- identity
- jitter
- dodge
- stack
- fill

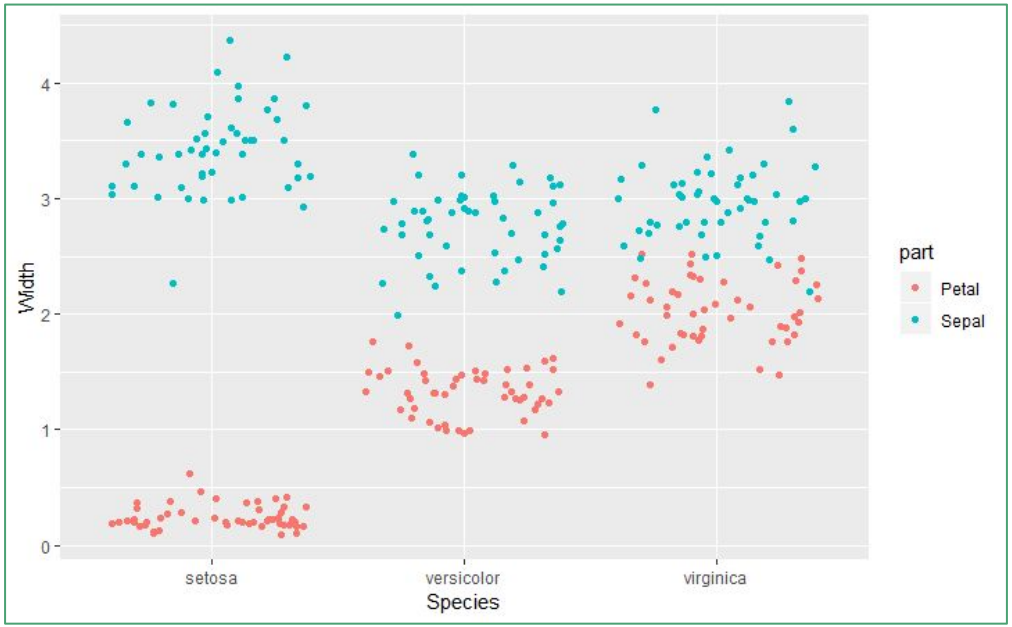
# ggplot

```
> ggplot(iris.wide, aes(Species, Width)) + geom_jitter()
```



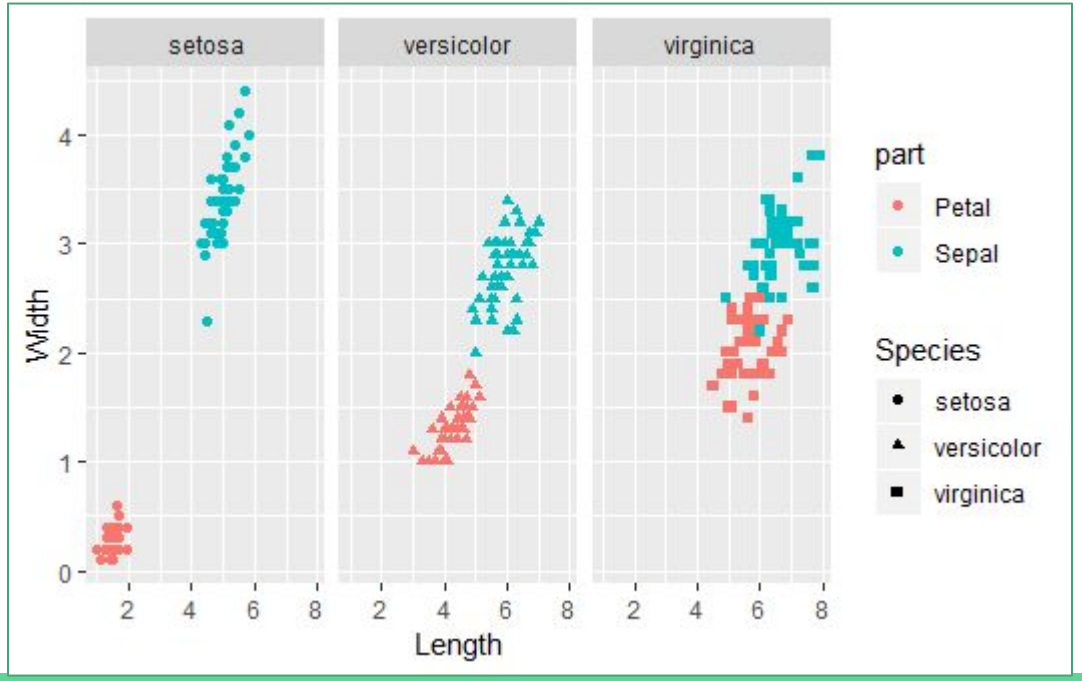
# ggplot

```
> ggplot(iris.wide, aes(Species, Width)) +
geom_jitter(aes(col=part))
```



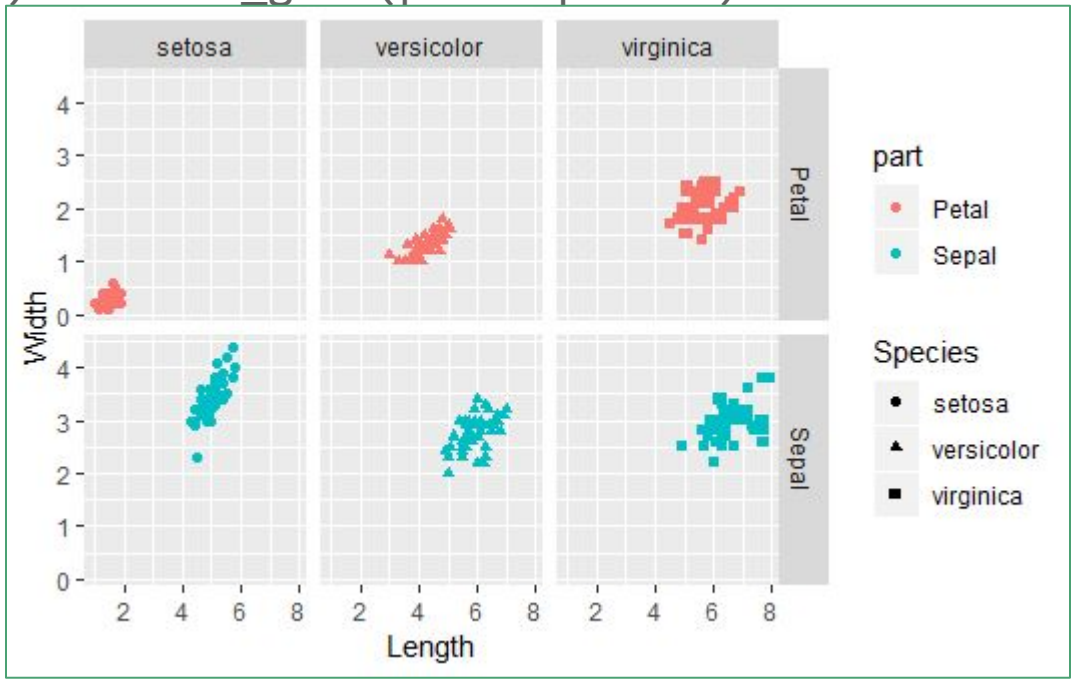
# ggplot2 - facet

```
ggplot(iris.wide, aes(Length, Width, col = part, shape = Species))
+ geom_point() + facet_grid(~Species)
```



# ggplot2

```
ggplot(iris.wide, aes(Length, Width, col = part, shape = Species))
+ geom_point() + facet_grid(part~Species)
```



# ggplot2 - adicionando camadas

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

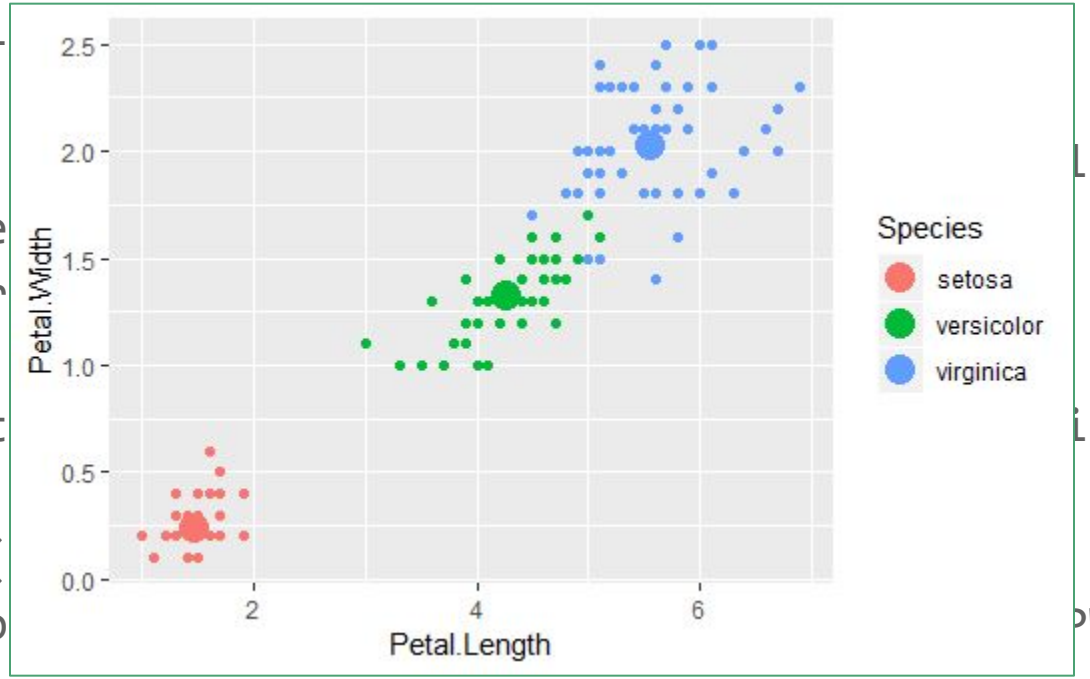

Como adicionar a média do comprimento e da largura de cada espécie?

# ggplot2 - adicionando camadas

```
> library(dplyr)
> iris.group <- group_by(iris, Species)
> iris.group <- summarise(iris.group, PLM = mean(Petal.Length), PWM = mean(Petal.Width))
> str(iris.group)

> g <- ggplot(iris, aes(x = Petal.Length, y = Petal.Width, col = Species)) +
  geom_point()
> g + geom_point(data = iris.group, aes(x = PLM, y = PWM), size = 5)
```

# ggplot2 - adicionando camadas

```
> library(dpl
> iris.group
> iris.group                                                    l.Length),
PWM = mean(Pe
> str(iris.gr

> g <- ggplot                                                   idth, col =
Species)) +
  geom_point(
> g + geom_po                                                   PWM), size =
5)
```



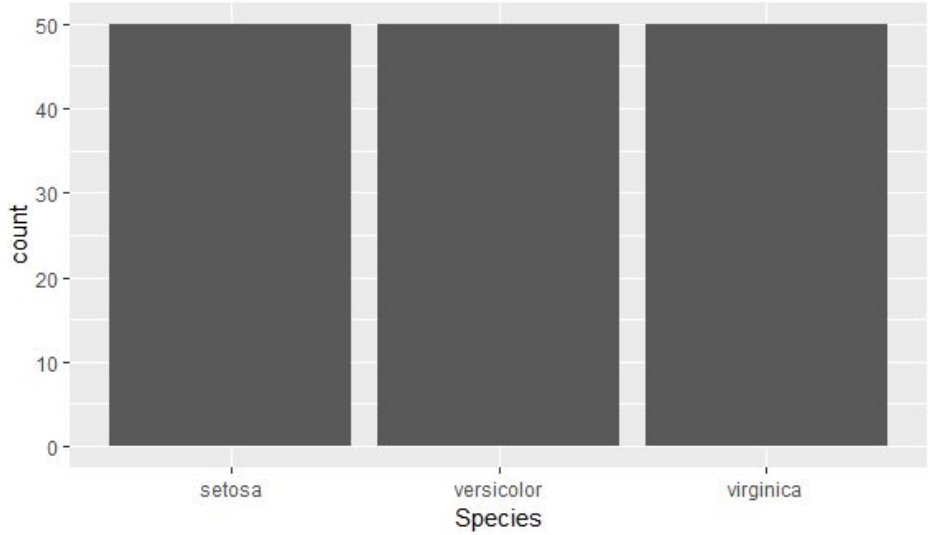
# ggplot2 - Camada Geométrica

37 geometrias

| | | | | | | |
|---|---|---|---|---|---|---|
| abline | contour | errorbarh | line | polygon | segment | vline |
| area | crossbar | freqpoly | linerange | quantile | smooth | |
| bar | density | hex | map | raster | step | |
| bin2d | density2d | histogram | path | rect | text | |
| blank | dotplot | hline | point | ribbon | tile | |
| boxplot | errorbar | jitter | pointrange | rug | violin | |

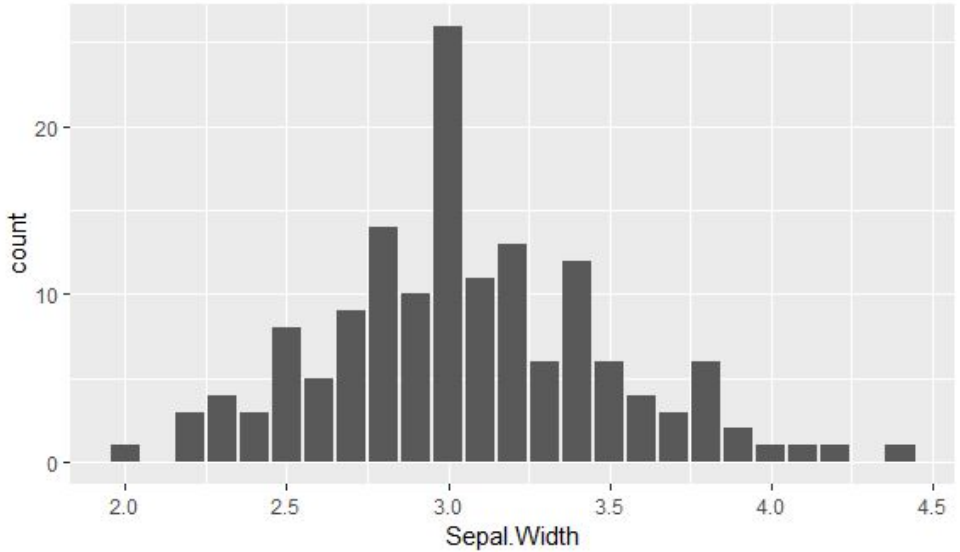
# ggplot2 - gráfico de barra

```
data(iris)
str(iris)
ggplot(iris, aes(Species)) + geom_bar()
```



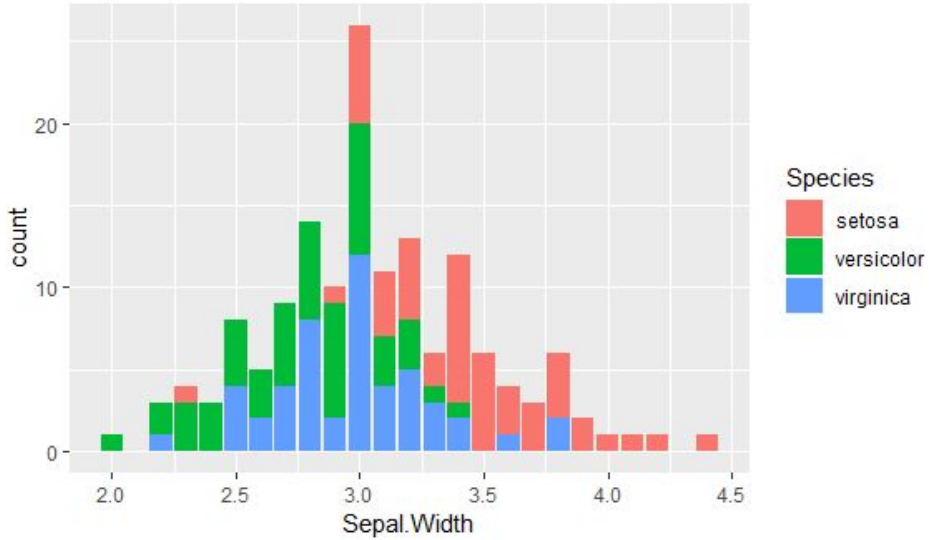
# ggplot2 - gráfico de barra

```
data(iris)
str(iris)
ggplot(iris, aes(Sepal.Width)) + geom_bar()
```



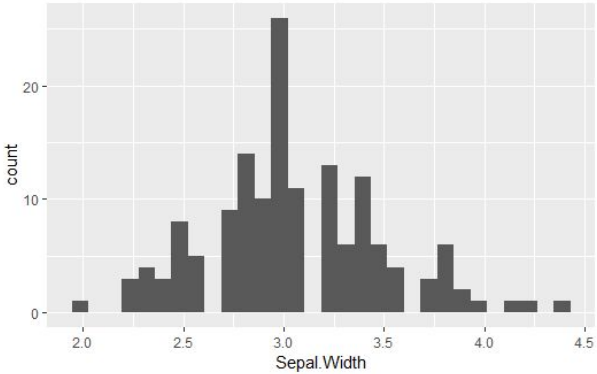
# ggplot2 - gráfico de barra

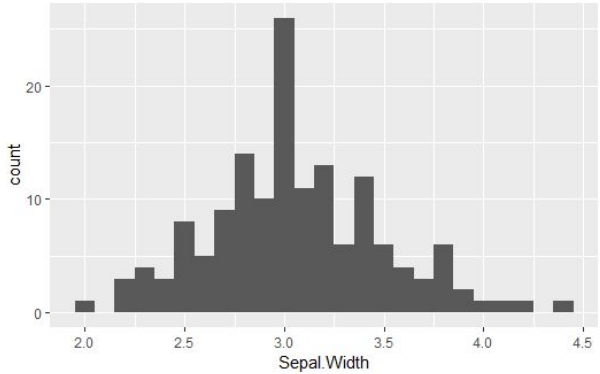
```
data(iris)
str(iris)
ggplot(iris, aes(Sepal.Width, fill = Species)) + geom_bar()
```

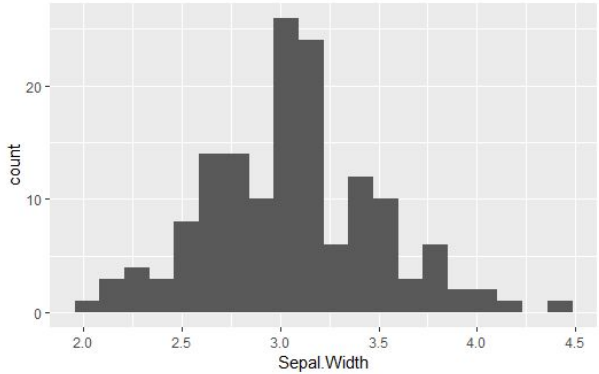


# ggplot2 - Histogramas

```
ggplot(iris, aes(x = Sepal.Width)) + geom_histogram()
# default: bins = 30, position = stacks;
ggplot(iris, aes(x = Sepal.Width)) + geom_histogram(binwidth = 0.1)
ggplot(iris, aes(x = Sepal.Width)) + geom_histogram(bins = 20)
```

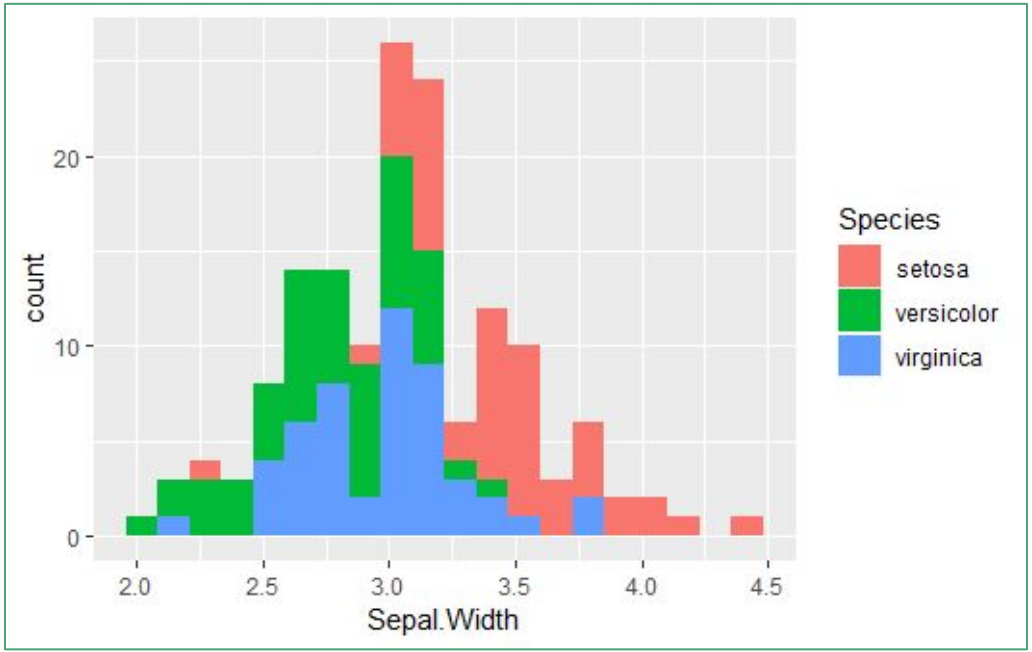


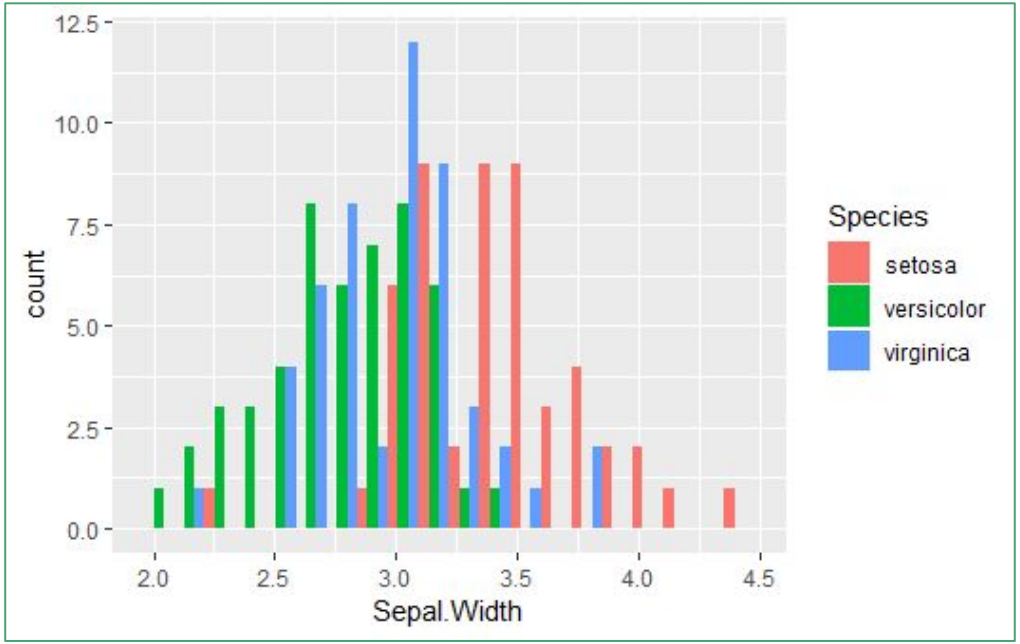
# ggplot2 - Histogramas

```
ggplot(iris, aes(x = Sepal.Width, fill = Species)) +
geom_histogram(bins = 20)
```



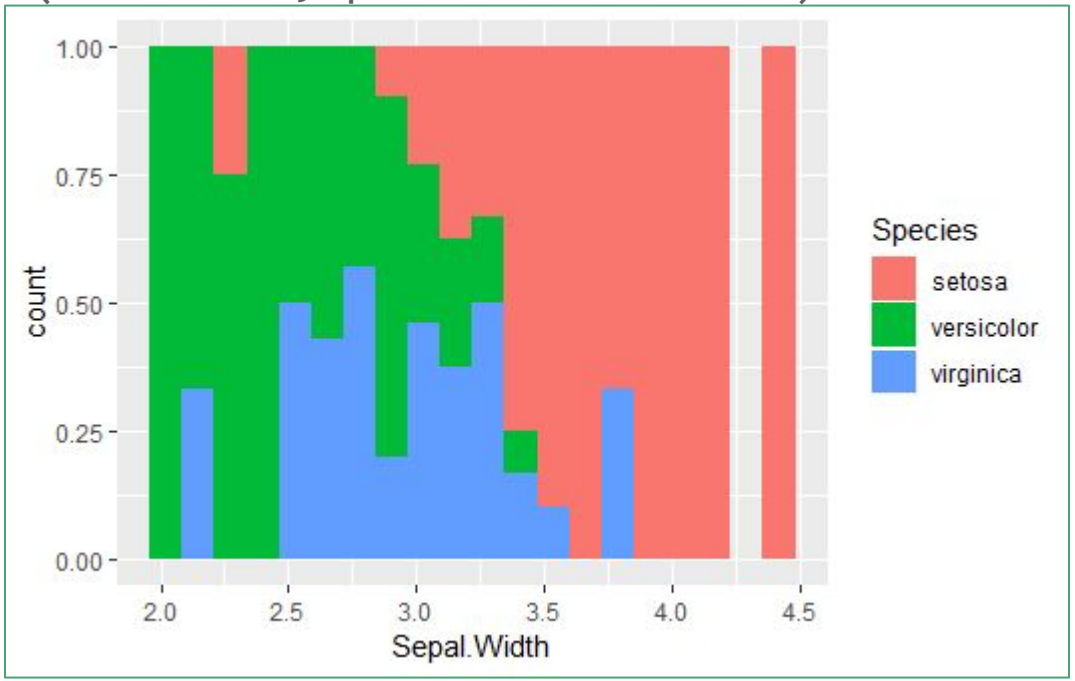
# ggplot2 - Histogramas

```
ggplot(iris, aes(x = Sepal.Width, fill = Species)) +
geom_histogram(bins = 20, position = "dodge")
```



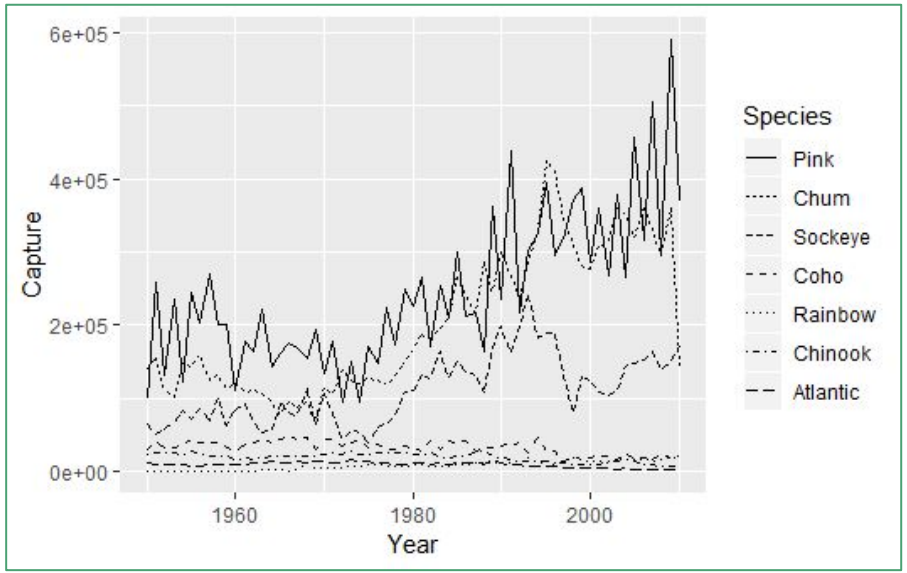
# ggplot2 - Histogramas

```
ggplot(iris, aes(x = Sepal.Width, fill = Species)) +
geom_histogram(bins = 20, position = "fill")
```



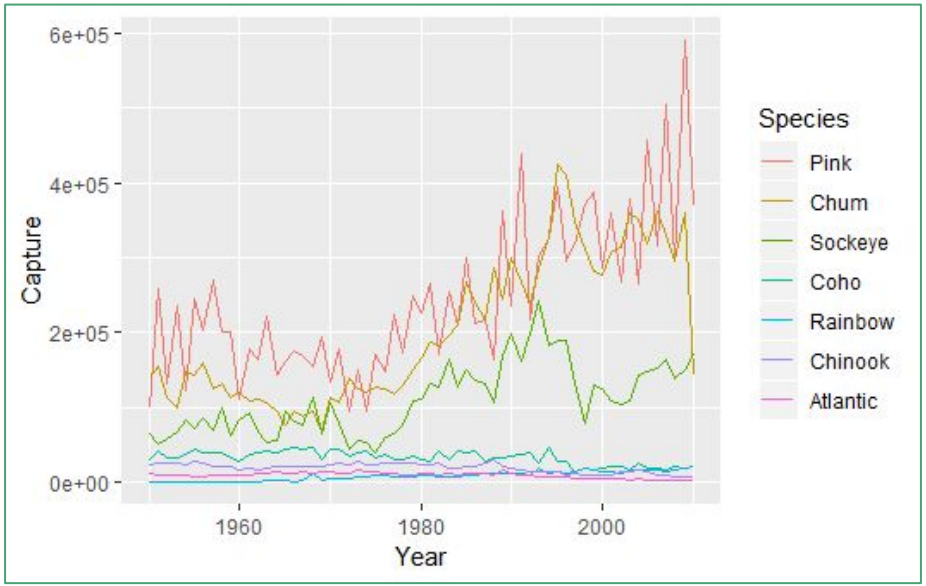
# ggplot2 - gráfico de linha

```
load("fish.RData")
str(fish.tidy)
ggplot(fish.tidy,aes(x=Year, y=Capture, linetype = Species)) +
geom_line()
```

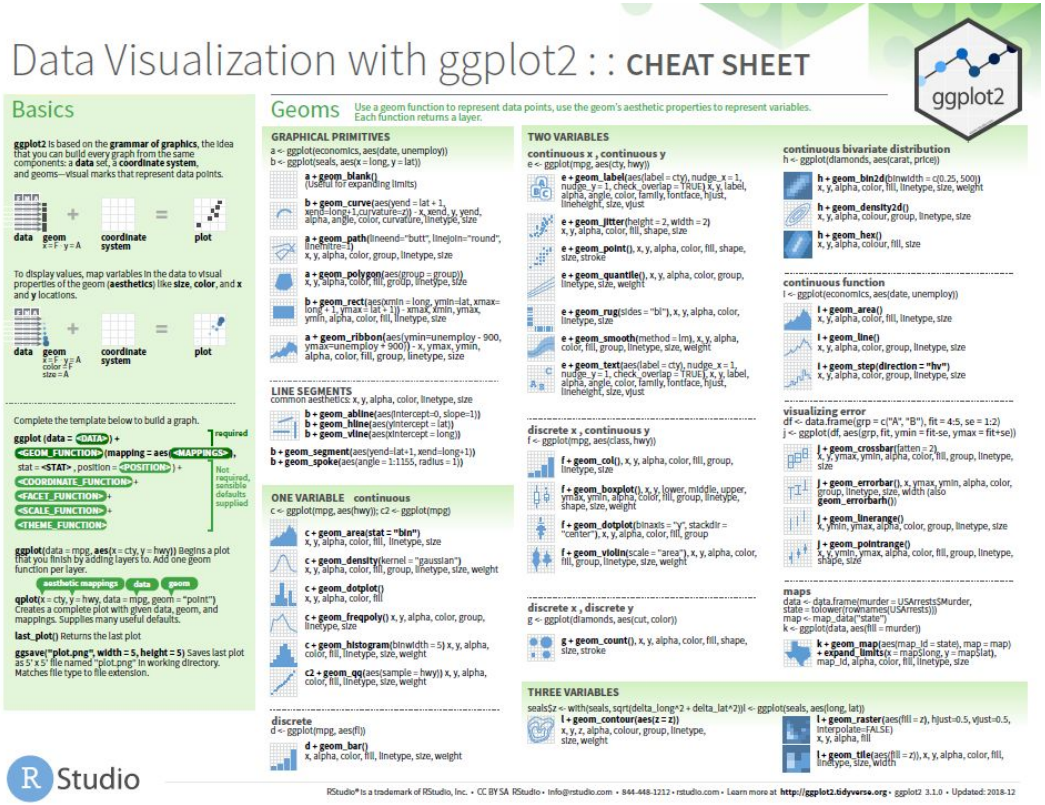


# ggplot2 - gráfico de linha

```
load("fish.RData")
str(fish.tidy)
ggplot(fish.tidy,aes(x=Year, y=Capture, col = Species)) +
geom_line()
```



# ggplot2 - cheat sheet



## 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.

data + geom (x = F, y = A) = coordinate system = plot

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

data + geom (x = F, y = A, color = F, size = A) = coordinate system = plot

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, xmax, 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

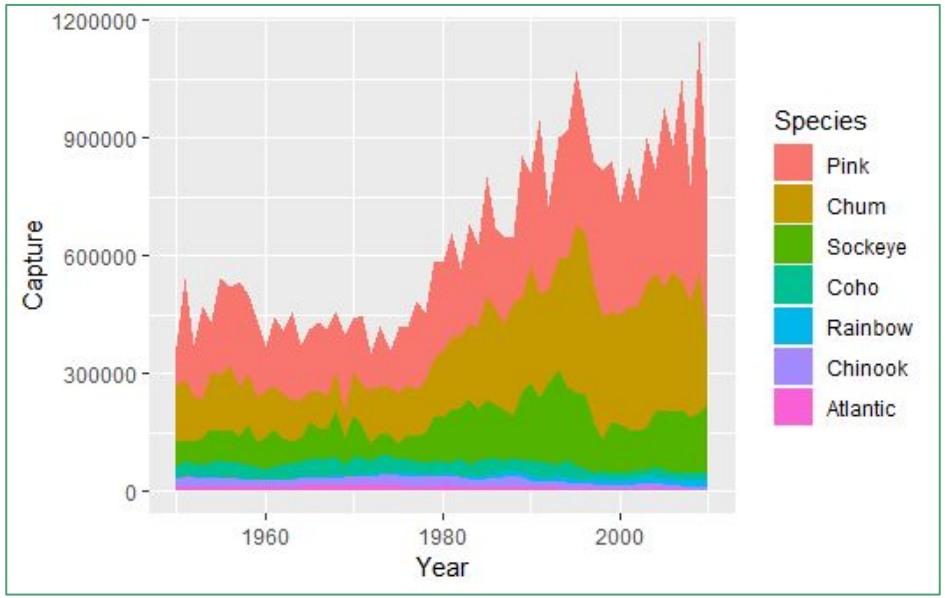
**l + geom_tile**(aes(fill = z)), x, y, alpha, color, fill, linetype, size, width

RStudio

# ggplot2 - gráfico de área

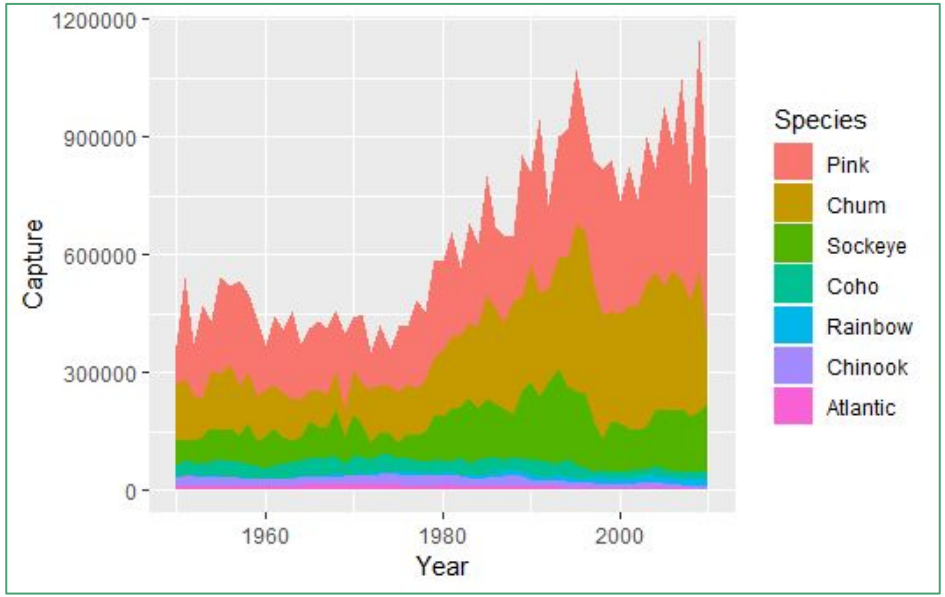
**Exercício**

Plote o gráfico abaixo usando o **ggplot2** e os dados **`fish.tidy`**:



# ggplot2 - gráfico de área

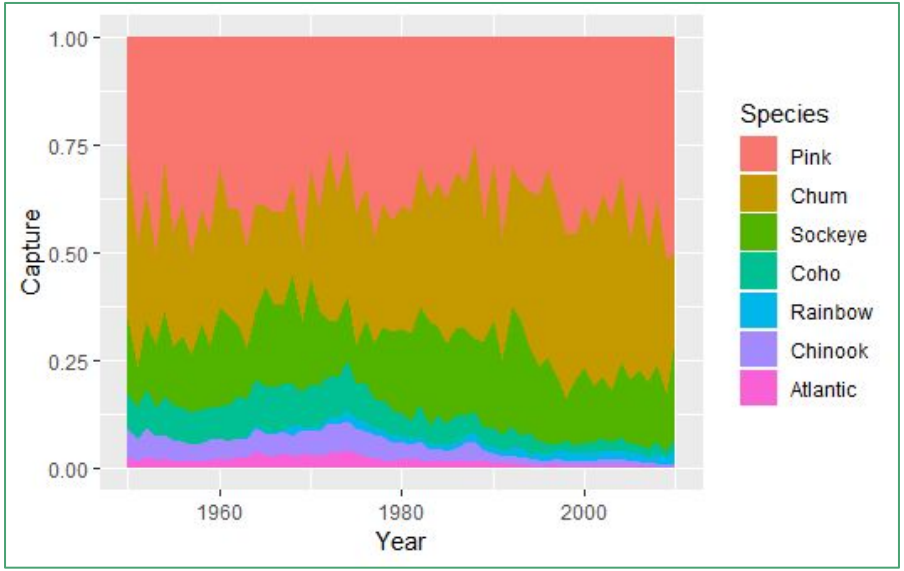
```
load(fish.RData)
str(fish.tidy)
ggplot(fish.tidy,aes(x=Year, y=Capture, fill = Species)) +
geom_area()
```



# ggplot2 - gráfico de área

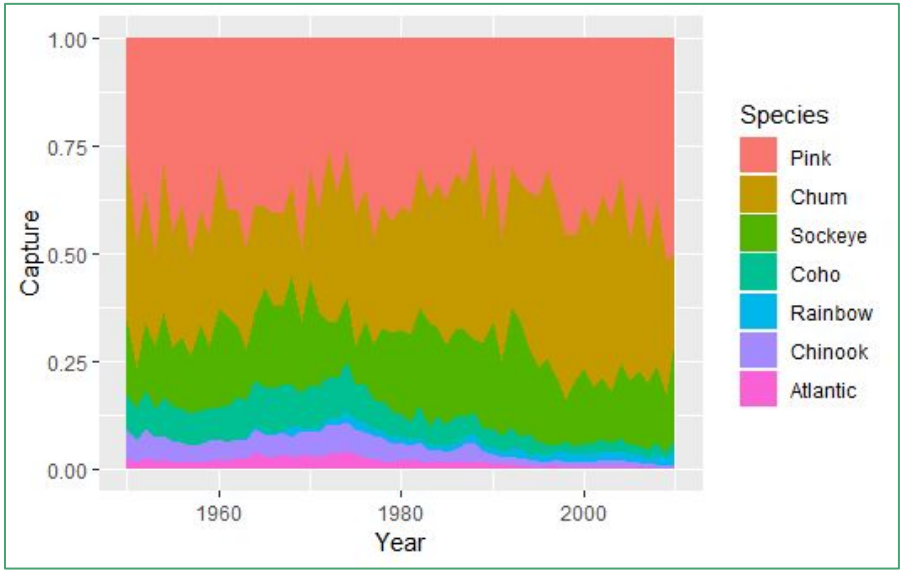
**Exercício**

Plote o gráfico abaixo usando o **ggplot2** e os dados **`fish.tidy`**:



# ggplot2 - gráfico de área

```
load(fish.RData)
str(fish.tidy)
ggplot(fish.tidy,aes(x=Year, y=Capture, fill = Species)) +
geom_area(position = "fill")
```



# Referência

https://skillgaze.com/2017/10/31/understanding-different-visualization-layers-of-ggplot/

Esta aula foi baseada no curso “**Data Visualization with ggplot2 (Part 1)**” de Rick Scavetta (https://www.datacamp.com/courses/data-visualization-with-ggplot2-1)