Plotting in **R**

Liam D. Bailey

Leibniz Institute of Zoo and Wildlife Research

June 2018

Why plot in **R**?

- Powerful (Large range of plot types)
- Customizable (Make your own style)
- Practical (Integrate your plots and your code together)

Plotting in **R**

1. Two ways to plot in **R**
2. Plotting with baseplot
3. Plotting with ggplot
4. Which one to use?

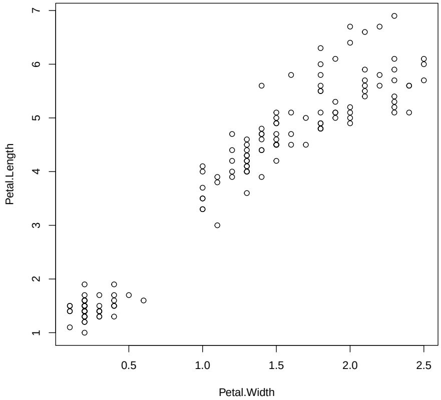
Two ways to plot in **R**

In **R** there are two (main) ways to make plots:

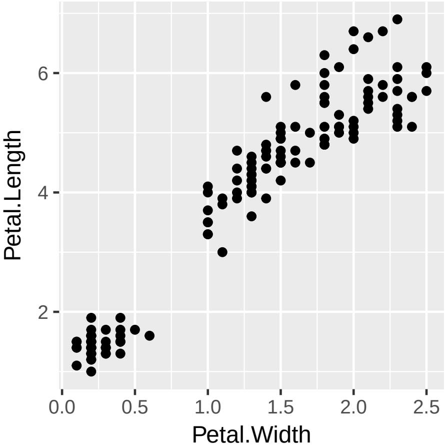
- **base plotting**: The standard way of plotting.
- **tidyverse plotting**: The most common alternative, using package *ggplot2*.

An example

```
plot(Petal.Length ~ Petal.Width, data = iris)
```



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



Plotting in R

1. Two ways to plot in R
2. Plotting with baseplot
3. Plotting with ggplot
4. Which one to use?

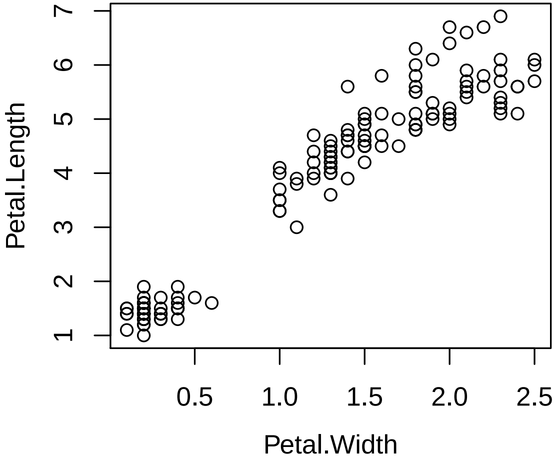
Plotting in R

1. Two ways to plot in R
2. Plotting with baseplot
   - Scatter plot
   - Boxplot
   - Histograms
   - Bargraphs
   - Aesthetics
   - Saving your plot
3. Plotting with ggplot
   - Scatter plot
   - Boxplot
   - Histograms
   - Bargraphs
   - Aesthetics
   - Saving your plot
4. Which one to use?

Baseplot: Scatter plots

In baseplot, you use slightly different functions for different plots.

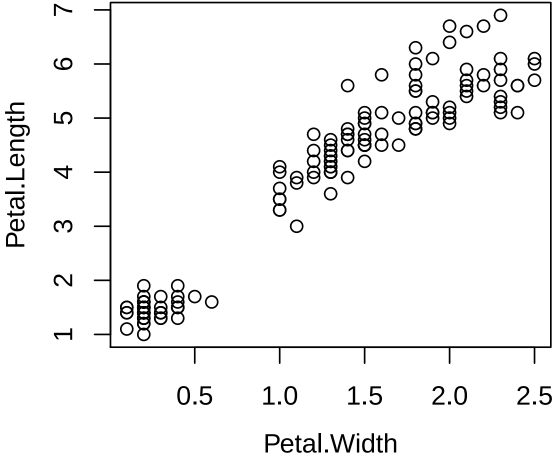
```
plot(Petal.Length ~ Petal.Width, data = iris)
```



Baseplot: Scatter plots

You can choose what type of scatter plot to display with argument type.

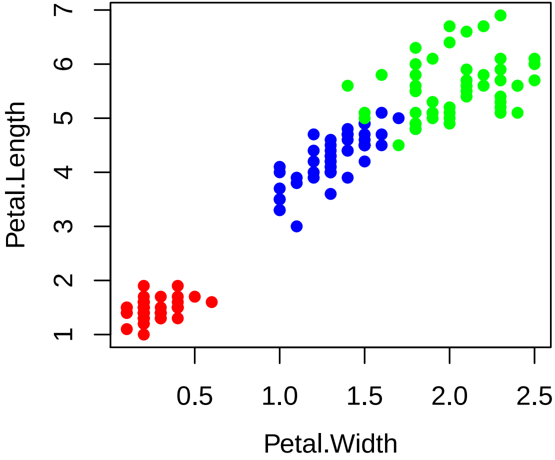
```
plot(Petal.Length ~ Petal.Width, data = iris)
```



Baseplot: Scatter plots

You can change point shapes (pch) and colour (col).

```
palette(c("red", "blue", "green"))
plot(Petal.Length ~ Petal.Width, data = iris,
     pch = 16, col = iris$Species)
```

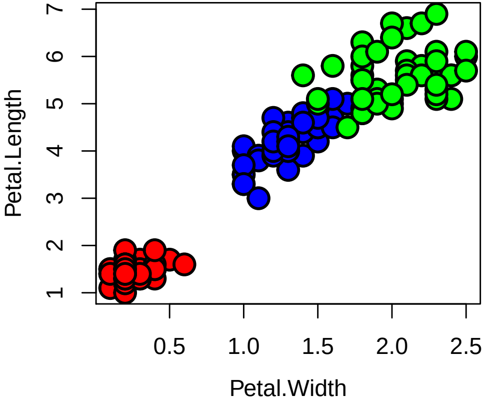


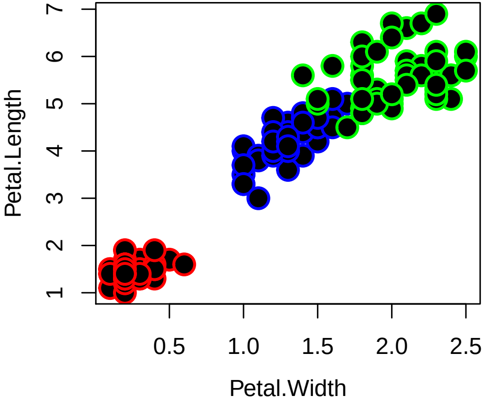
Beware

Be aware that for many elements they have both an outline colour (col) and background colour (bg).

```r
palette(c("red", "blue", "green"))
plot(Petal.Length ~ Petal.Width, data = iris, cex = 2,
     pch = 21, bg = iris$Species, col = "black", lwd = 2)
```

```r
palette(c("red", "blue", "green"))
plot(Petal.Length ~ Petal.Width, data = iris, cex = 2,
     pch = 21, col = iris$Species, bg = "black", lwd = 2)
```

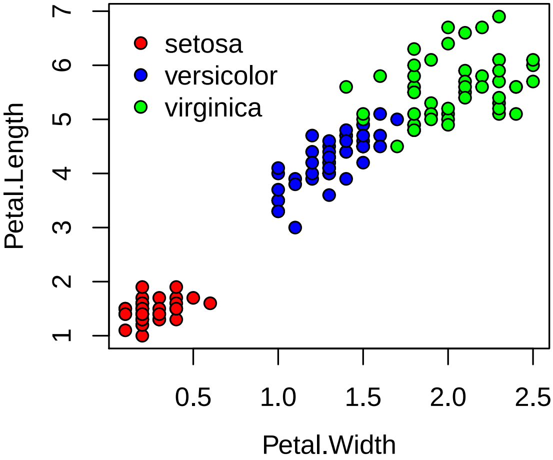




Baseplot: Scatter plots

Add a legend to make colours understandable.

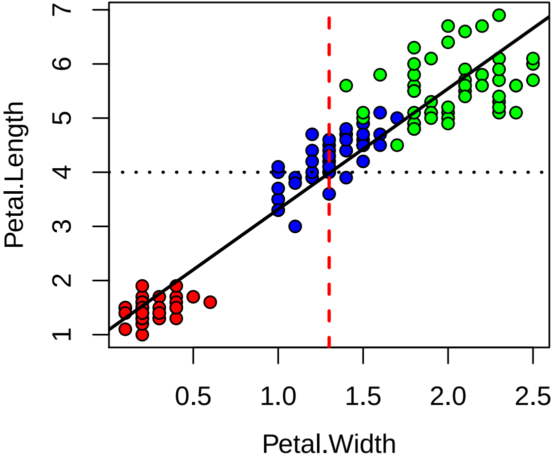
```
palette(c("red", "blue", "green"))
plot(Petal.Length ~ Petal.Width, data = iris, pch = 21, bg = iris$Species)
legend(x = 0.05, y = 7, c("setosa", "versicolor", "virginica"), pch = 21, pt.bg = c("red", "blue", "green"), bty = "n")
```



Baseplot: Scatter plots

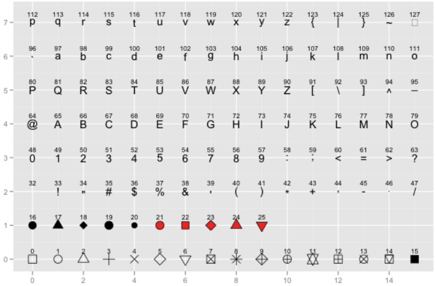
You can add lines to the plot.

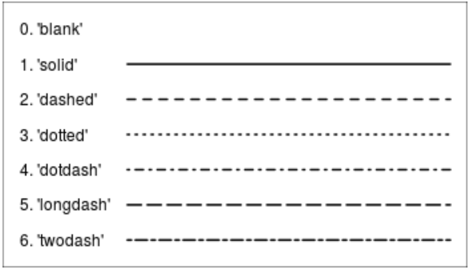
```
plot(Petal.Length ~ Petal.Width, data = iris, pch = 21, bg = iris$Species)
abline(v = 1.3, lty = 2, col = "red", lwd = 2)
abline(h = 4, lty = 3, col = "black", lwd = 2)
abline(a = 1.084, lty = 1, b = 2.23, lwd = 2)
```



A side note on points and lines

There are standard values for each point and line type. These are **the same** for ggplot and baseplot.

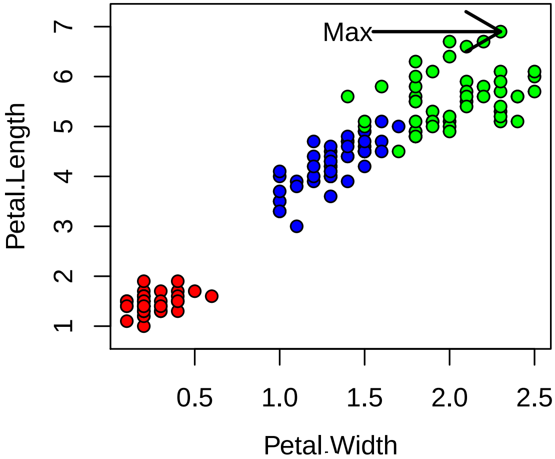




Baseplot: Scatter plots

You can add text and arrows.

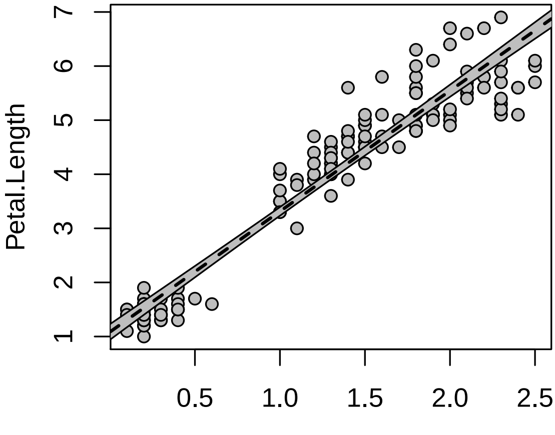
```
plot(Petal.Length ~ Petal.Width, data = iris, pch = 21, bg = iris$Species)
max.value <- iris[iris$Petal.Length == max(iris$Petal.Length), ]
arrows(x0 = max.value$Petal.Width - 0.75, y0 = max.value$Petal.Length,
       x1 = max.value$Petal.Width, y1 = max.value$Petal.Length, lwd = 2)
text(x = max.value$Petal.Width - 0.9, y = max.value$Petal.Length, labels = "Max")
```



Baseplot: Scatter plots

Including error around lines requires you to build a polygon

```
test_mod <- lm(Petal.Length ˜ Petal.Width, data = iris)
newdat   <- data.frame(Petal.Width = seq(0, 3, length.out = 100))
pred     <- predict(test_mod, newdata = newdat, interval = "confidence")
plot(Petal.Length ˜ Petal.Width, data = iris, pch = 21, bg = "grey")
polygon(x = c(rev(newdat$Petal.Width), newdat$Petal.Width), y = c(rev(pred[, 3]), pred[, 2]), col = "grey")
abline(a = coef(test_mod)[1], lty = 2, b = coef(test_mod)[2], lwd = 2)
```



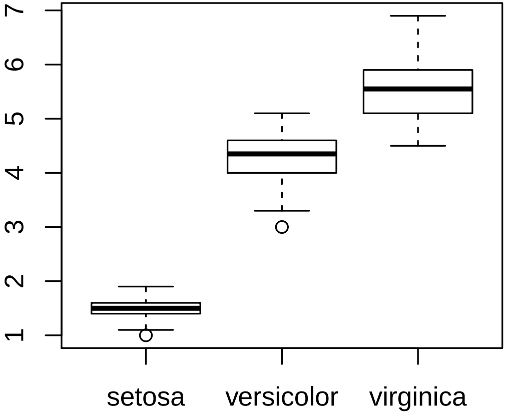
Plotting in R

1. Two ways to plot in **R**
2. Plotting with baseplot
   - Scatter plot
   - Boxplot
   - Histograms
   - Bargraphs
   - Aesthetics
   - Saving your plot
3. Plotting with ggplot
   - Scatter plot
   - Boxplot
   - Histograms
   - Bargraphs
   - Aesthetics
   - Saving your plot
4. Which one to use?

Baseplot: Boxplot

Other plot types use different functions.

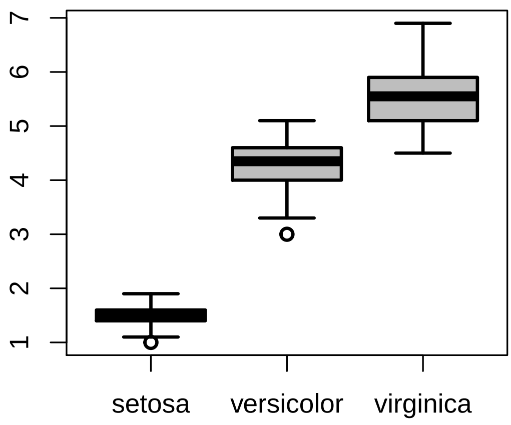
```
boxplot(Petal.Length ~ Species, data = iris)
```



Baseplot: Boxplot

Many of the same changes made to scatterplots can be made here.

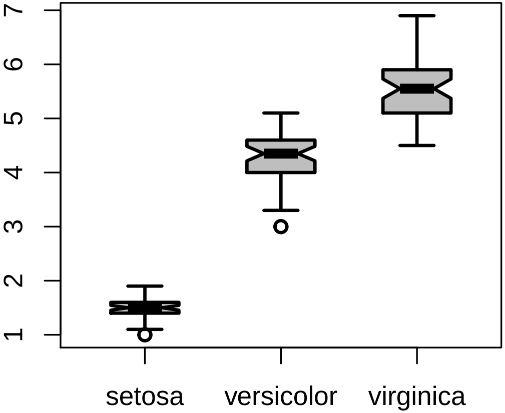
```
boxplot(Petal.Length ~ Species, data = iris, col = "grey", lwd = 2, lty = 1)
```



Baseplot: Boxplot

There are also some boxplot specific arguments.

```
boxplot(Petal.Length ~ Species, data = iris, col = "grey", lwd = 2, lty = 1,
        width = c(1, 2, 1), notch = TRUE)
```

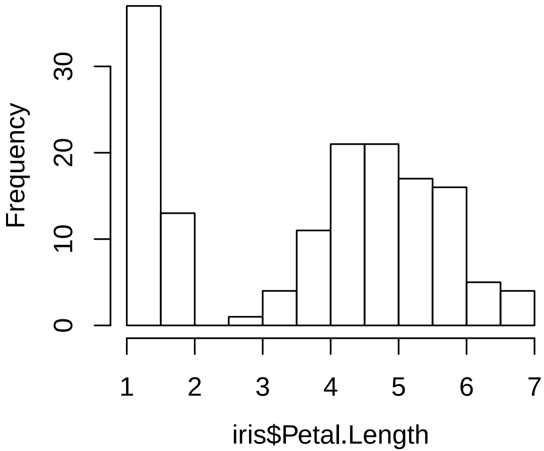


Plotting in R

1. Two ways to plot in R
2. Plotting with baseplot
   - Scatter plot
   - Boxplot
   - Histograms
   - Bargraphs
   - Aesthetics
   - Saving your plot
3. Plotting with ggplot
   - Scatter plot
   - Boxplot
   - Histograms
   - Bargraphs
   - Aesthetics
   - Saving your plot
4. Which one to use?

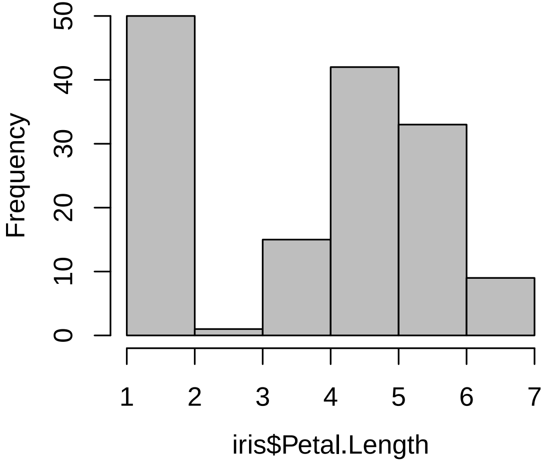
Baseplot: Histogram

```r
hist(iris$Petal.Length, main = "")
```



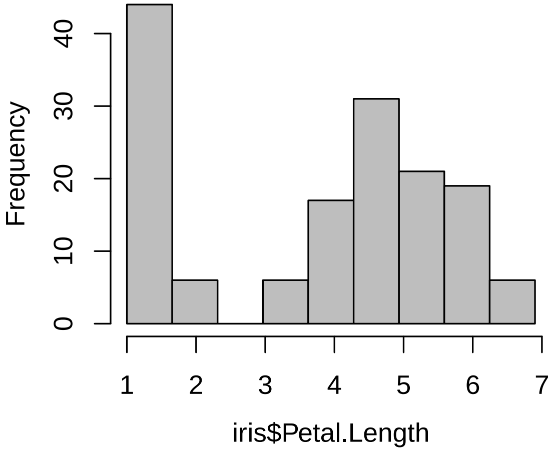
Baseplot: Histogram

```
hist(iris$Petal.Length, main = "", breaks = 5, col = "grey")
```



Baseplot: Histogram

```r
hist_breaks <- seq(min(iris$Petal.Length), max(iris$Petal.Length), length.out = 10)
hist(iris$Petal.Length, main = "", breaks = hist_breaks, col = "grey")
```

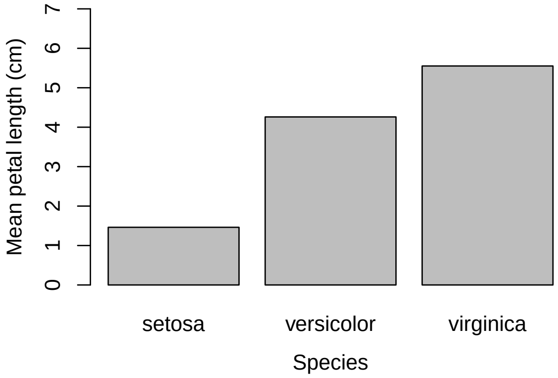


Plotting in R

1. Two ways to plot in R
2. Plotting with baseplot
   - Scatter plot
   - Boxplot
   - Histograms
   - Bargraphs
   - Aesthetics
   - Saving your plot
3. Plotting with ggplot
   - Scatter plot
   - Boxplot
   - Histograms
   - Bargraphs
   - Aesthetics
   - Saving your plot
4. Which one to use?

Baseplot: Bargraph

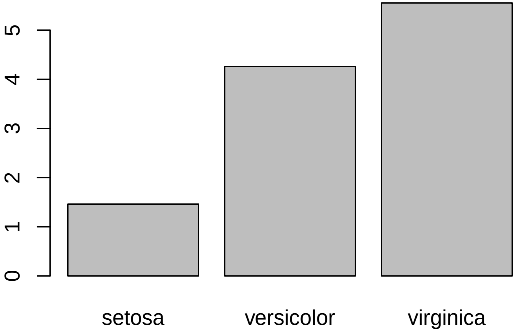
```
spp_means
barplot(height = spp_means$mean, names.arg = spp_means$Species, ylim = c(0, 7),
        xlab = "Species", ylab = "Mean petal length (cm)")
```



Baseplot: Bargraph

Adding errorbars can be done with the arrows function.

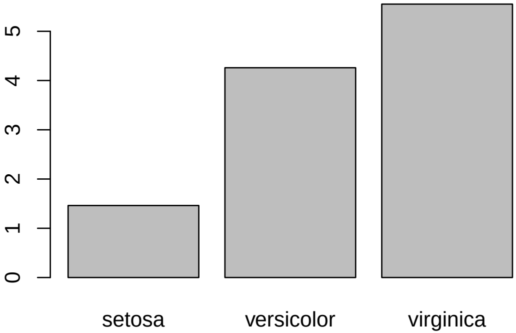
```
spp_means
bar_locations <- barplot(height = spp_means$mean, names.arg = spp_means$Species)
barplot(height = spp_means$mean, names.arg = spp_means$Species, ylim = c(0, 7),
        xlab = "Species", ylab = "Mean petal length (cm)")
arrows(x0 = bar_locations[, 1], x1 = bar_locations[, 1],
       y0 = spp_means$mean - spp_means$SE, y1 = spp_means$mean + spp_means$SE,
       angle = 90, lwd = 2, code = 3)
```



Baseplot: Bargraph

As before, there are similar arguments available.

```
spp_means
bar_locations <- barplot(height = spp_means$mean, names.arg = spp_means$Species)
barplot(height = spp_means$mean, names.arg = spp_means$Species, ylim = c(0, 7),
        xlab = "Species", ylab = "Mean petal length (cm)",
        col = "white", cex.axis = 0.75)
```



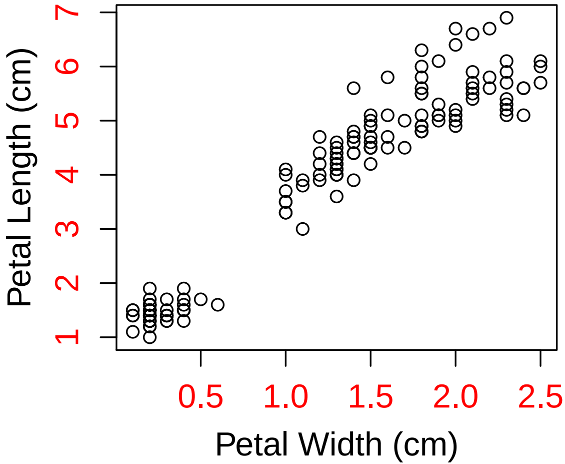
Plotting in R

1. Two ways to plot in R
2. Plotting with baseplot
   - Scatter plot
   - Boxplot
   - Histograms
   - Bargraphs
   - **Aesthetics**
   - Saving your plot
3. Plotting with ggplot
   - Scatter plot
   - Boxplot
   - Histograms
   - Bargraphs
   - Aesthetics
   - Saving your plot
4. Which one to use?

Change text

You can size and colour of axis text easily.

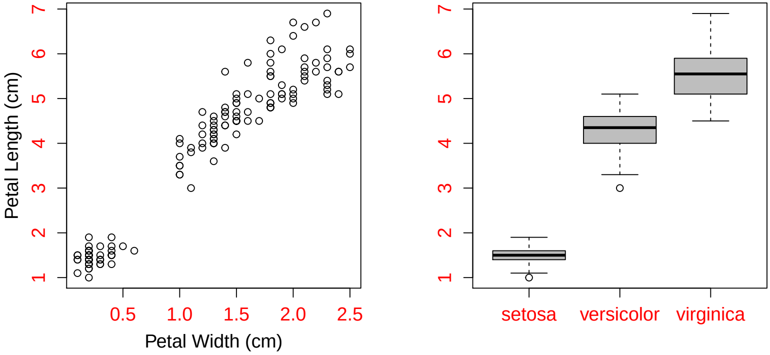
```
plot(Petal.Length ~ Petal.Width, data = iris,
     xlab = "Petal Width (cm)", ylab = "Petal Length (cm)",
     cex.lab = 1.5, cex.axis = 1.5, col.axis = "red")
```



Combining plots

You can easily combine multiple baseplots together by changing the global parameters.

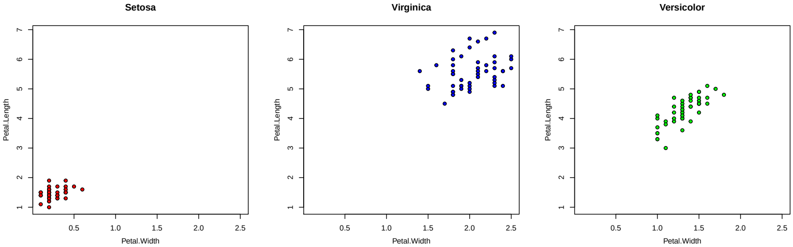
```
par(mfrow = c(1, 2))
plot(Petal.Length ~ Petal.Width, data = iris,
     xlab = "Petal Width (cm)", ylab = "Petal Length (cm)",
     cex.lab = 1.5, cex.axis = 1.5, col.axis = "red")
boxplot(Petal.Length ~ Species, data = iris, col = "grey")
par(mfrow = c(1, 1))
```



Facetting

This can be used to created 'facet' plots.

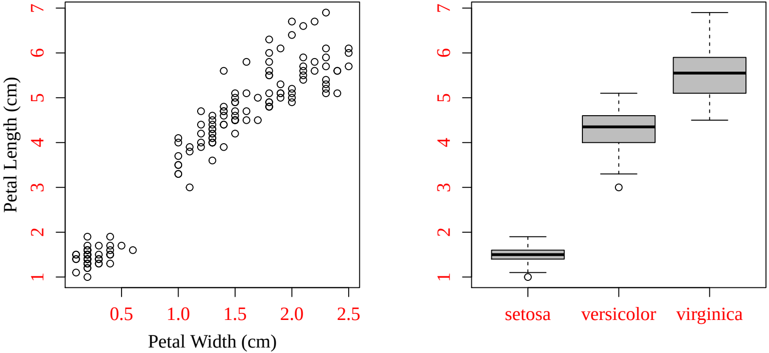
```
par(mgp = c(2.5, 1, 0))
par(mfrow = c(1, 3))
split_data <- split(iris, iris$Species)
plot(Petal.Length ~ Petal.Width, data = split_data$setosa, main = "Setosa", pch = 21, bg = "red", col = "black",
     xlim = range(iris$Petal.Width), ylim = range(iris$Petal.Length))
plot(Petal.Length ~ Petal.Width, data = split_data$virginica, main = "Virginica", pch = 21, bg = "blue", col = "black",
     xlim = range(iris$Petal.Width), ylim = range(iris$Petal.Length))
plot(Petal.Length ~ Petal.Width, data = split_data$versicolor, main = "Versicolor", pch = 21, bg = "green", col = "black",
     xlim = range(iris$Petal.Width), ylim = range(iris$Petal.Length))
```



Text family

Global parameters can also be used to change the font family.

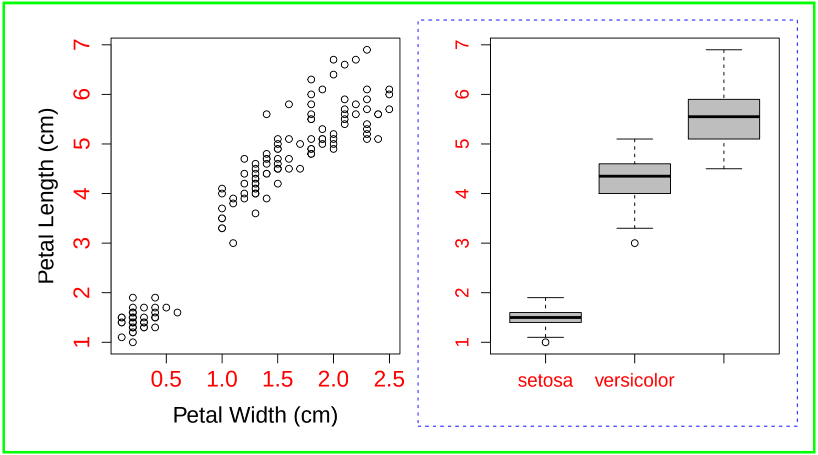
```
par(mfrow = c(1, 2), family = "serif")

plot(Petal.Length ~ Petal.Width, data = iris,
     xlab = "Petal Width (cm)", ylab = "Petal Length (cm)",
     cex.lab = 1.5, cex.axis = 1.5, col.axis = "red")
boxplot(Petal.Length ~ Species, data = iris, col = "grey")

par(mfrow = c(1, 1), family = "sans")
```



Plot margins

You can change individual plot margins (blue) and outer margins (green).

```
par(mfrow = c(1, 2), mar = c(4, 4, 1, 1),
    oma = c(1.5, 2, 1, 1))
plot(Petal.Length ~ Petal.Width, data = iris, xlab = "Petal Width (cm)", ylab = "Petal Length (cm)",
     cex.lab = 1.5, cex.axis = 1.5, col.axis = "red")
boxplot(Petal.Length ~ Species, data = iris, col = "grey", cex.lab = 0.75, cex.axis = 1.5, col.axis = "red")
```



Plotting in R

1. Two ways to plot in **R**
2. Plotting with baseplot
   - Scatter plot
   - Boxplot
   - Histograms
   - Bargraphs
   - Aesthetics
   - Saving your plot
3. Plotting with ggplot
   - Scatter plot
   - Boxplot
   - Histograms
   - Bargraphs
   - Aesthetics
   - Saving your plot
4. Which one to use?

Baseplot: Exporting

```
?pdf ?jpeg ?tiff ?bmp ?postscript
```

```
pdf("base_plot.pdf", width = 15, height = 5)
plot(Petal.Length ~ Petal.Width, data = iris)
dev.off()
```

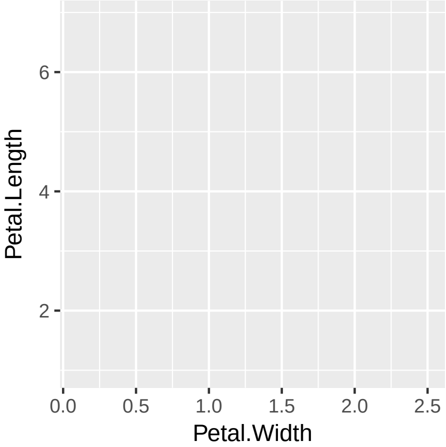
Plotting in R

1. Two ways to plot in R
2. Plotting with baseplot
3. Plotting with ggplot
4. Which one to use?

ggplot: An introduction

Unlike baseplot, ggplot works around a single function. We use different functions to add layers onto a ggplot.

```
ggplot(data = iris, aes(x = Petal.Width, y = Petal.Length))
```



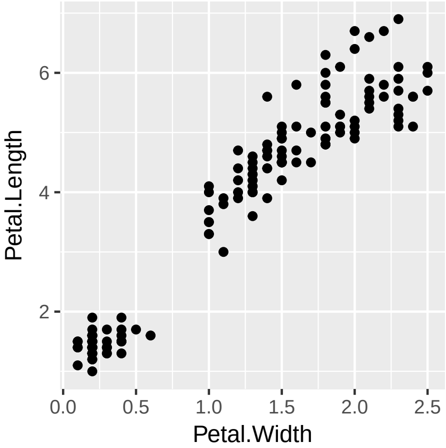
Plotting in R

1. Two ways to plot in **R**
2. Plotting with baseplot
   - Scatter plot
   - Boxplot
   - Histograms
   - Bargraphs
   - Aesthetics
   - Saving your plot
3. Plotting with ggplot
   - Scatter plot
   - Boxplot
   - Histograms
   - Bargraphs
   - Aesthetics
   - Saving your plot
4. Which one to use?

ggplot: Scatter plots

We build onto our initial ggplot argument.

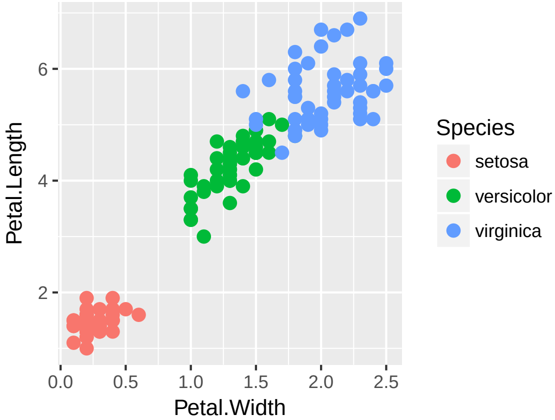
```
ggplot(data = iris, aes(x = Petal.Width, y = Petal.Length)) +
  geom_point()
```



ggplot: Scatter plots

You can change point shapes (shape) and colour (colour). The numbers used here are **the same** as the ones we used above.

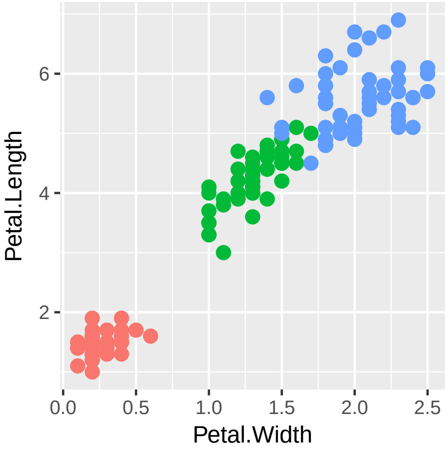
```
ggplot(data = iris, aes(x = Petal.Width, y = Petal.Length)) +
  geom_point(aes(colour = Species), shape = 16, size = 3)
```



ggplot: Scatter plots

In ggplot the legend is included by default. You will need to manually remove it.

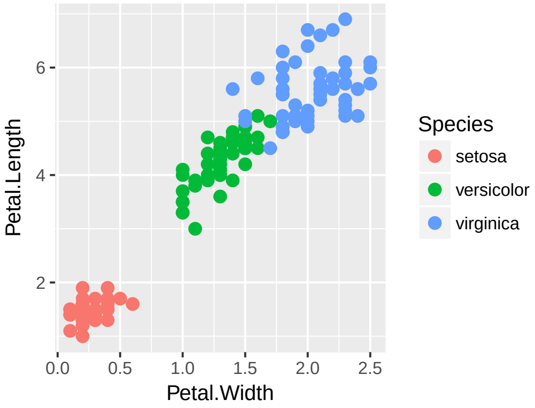
```
ggplot(data = iris, aes(x = Petal.Width, y = Petal.Length)) +
  geom_point(aes(colour = Species), shape = 16, size = 3)+
  theme(legend.position = "none")
```



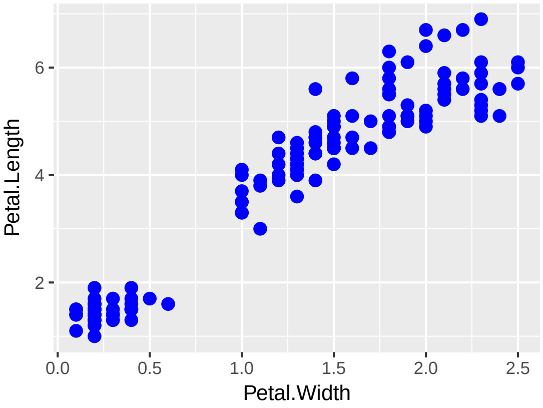
The aesthetic argument

The aesthetic argument in ggplot (aes) is a powerful tool for changing plot aesthetics. If you specify an aesthetic argument inside aes() it will give each data point a different aesthetic based on its value. If you specify the same aesthetic argument outside aes() it will give all data points the same aesthetic.

```r
ggplot(data = iris, aes(x = Petal.Width, y = Petal.Length)) +
  geom_point(aes(colour = Species), shape = 16, size = 3)
```



```r
ggplot(data = iris, aes(x = Petal.Width, y = Petal.Length)) +
  geom_point(colour = "blue", shape = 16, size = 3)
```

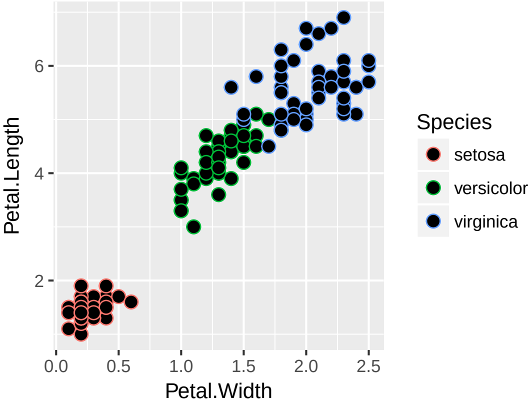


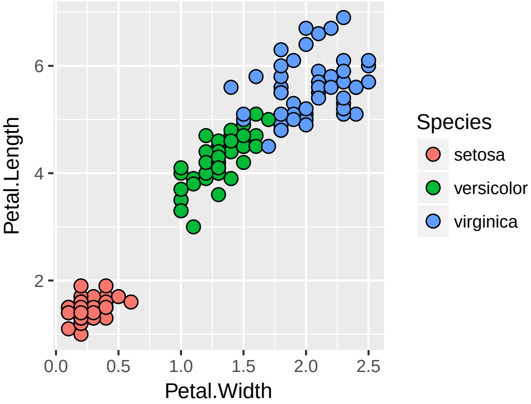
Beware

Again, there is a difference between outline colour (colour) and backgroun colour (fill).

```
ggplot(data = iris, aes(x = Petal.Width, y = Petal.Length)) +
  geom_point(aes(colour = Species), fill = "black",
             shape = 21, size = 3)
```

```
ggplot(data = iris, aes(x = Petal.Width, y = Petal.Length)) +
  geom_point(aes(fill = Species), colour = "black",
             shape = 21, size = 3)
```

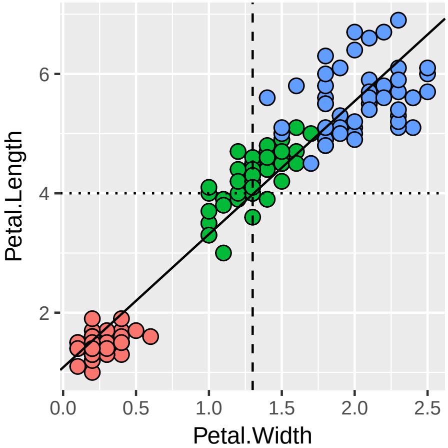




ggplot: Scatter plots

You can add lines to the plot.

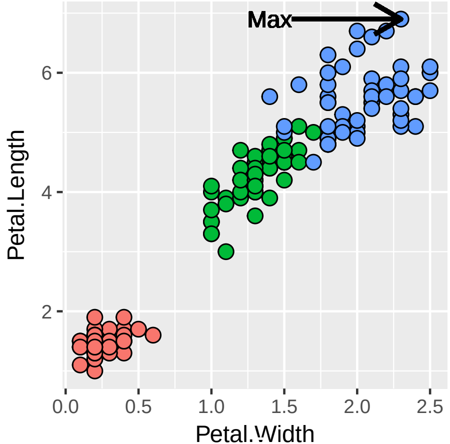
```
ggplot(data = iris, aes(x = Petal.Width, y = Petal.Length)) +
  geom_point(aes(fill = Species), shape = 21, size = 3)+
  theme(legend.position = "none")+
  geom_hline(yintercept = 4, lty = 3)+
  geom_vline(xintecept = 1.3, lty = 2)+
  geom_abline(intercept = 1.084, slope = 2.23, lty = 1)
```



Baseplot: Scatter plots

You can add text and arrows.

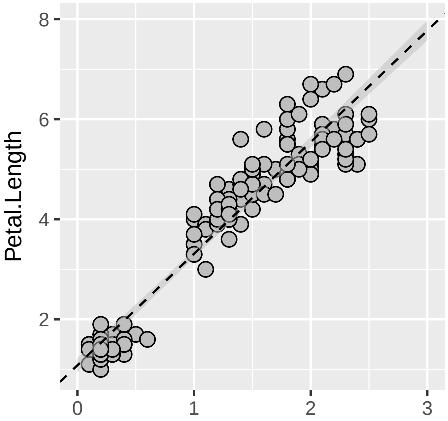
```
max.value <- iris[iris$Petal.Length == max(iris$Petal.Length), ]
ggplot(data = iris, aes(x = Petal.Width, y = Petal.Length)) +
  geom_point(aes(fill = Species), shape = 21, size = 3)+
  theme(legend.position = "none")+
  geom_segment(aes(x = max.value$Petal.Width - 0.75, xend = max.value$Petal.Width,
                   y = max.value$Petal.Length, yend = max.value$Petal.Length), size = 1, arrow = arrow(length = unit(0.5, "cm")))+
  geom_text(aes(x = max.value$Petal.Width - 0.9, y = max.value$Petal.Length, label = "Max"))
```



ggplot: Scatter plots

In ggplot, confidence intervals can be added with specialised function geom$_{r}ibbon$.

```
test_mod <- lm(Petal.Length ~ Petal.Width, data = iris)
newdat   <- data.frame(Petal.Width = seq(0, 3, length.out = 100))
pred     <- predict(test_mod, newdata = newdat, interval = "confidence")
ggplot() +
  geom_point(data = iris, aes(x = Petal.Width, y = Petal.Length), fill = "grey", shape = 21, size = 3) +
  geom_ribbon(data = NULL, aes(x = newdat$Petal.Width, ymin = pred[, 2], ymax = pred[, 3]), fill = "grey", alpha = 0.5) +
  geom_abline(intercept = coef(test_mod)[1], slope = coef(test_mod)[2], lty = 2) +
  theme(legend.position = "none")
```



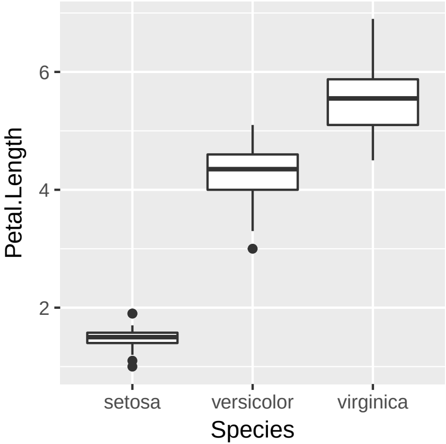
Plotting in R

1. Two ways to plot in R
2. Plotting with baseplot
   - Scatter plot
   - Boxplot
   - Histograms
   - Bargraphs
   - Aesthetics
   - Saving your plot
3. Plotting with ggplot
   - Scatter plot
   - Boxplot
   - Histograms
   - Bargraphs
   - Aesthetics
   - Saving your plot
4. Which one to use?

ggplot: Boxplot

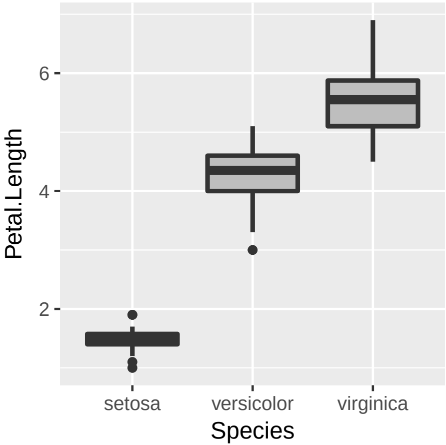
We build different plot types in a similar way.

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



ggplot: Boxplot

```
ggplot(iris, aes(x = Species, y = Petal.Length))+
  geom_boxplot(fill = "grey", size = 1, lty = 1)
```

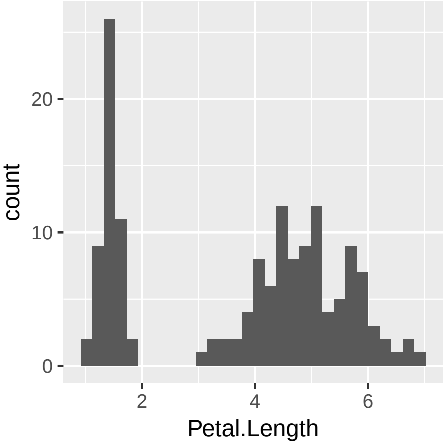


Plotting in R

1. Two ways to plot in R
2. Plotting with baseplot
   - Scatter plot
   - Boxplot
   - Histograms
   - Bargraphs
   - Aesthetics
   - Saving your plot
3. Plotting with ggplot
   - Scatter plot
   - Boxplot
   - Histograms
   - Bargraphs
   - Aesthetics
   - Saving your plot
4. Which one to use?

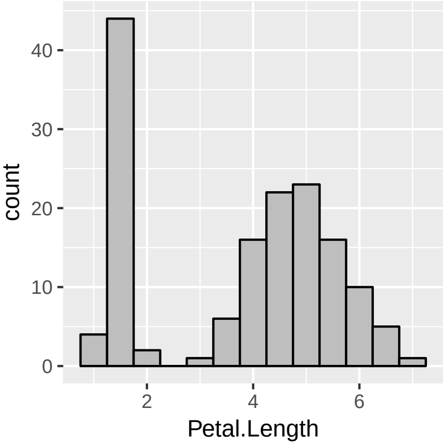
ggplot: Histogram

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



ggplot: Histogram

```
ggplot(iris, aes(x = Petal.Length))+
  geom_histogram(binwidth = 0.5, colour = "black", fill = "grey")
```

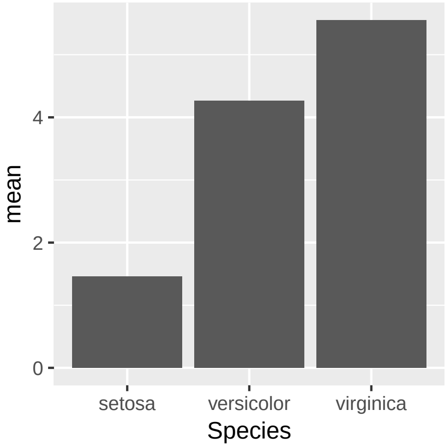


Plotting in **R**

1. Two ways to plot in **R**
2. Plotting with baseplot
   - Scatter plot
   - Boxplot
   - Histograms
   - Bargraphs
   - Aesthetics
   - Saving your plot
3. Plotting with ggplot
   - Scatter plot
   - Boxplot
   - Histograms
   - Bargraphs
   - Aesthetics
   - Saving your plot
4. Which one to use?

ggplot: Bargraph

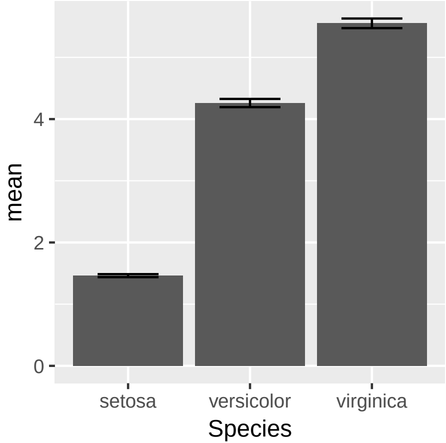
```
ggplot(spp_means, aes(x = Species, y = mean))+
  geom_col()
```



ggplot: Bargraph

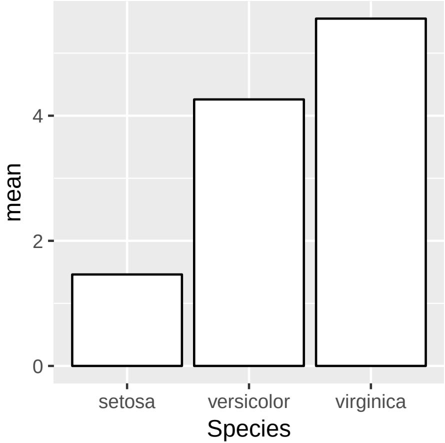
Adding errorbars in ggplot is much easier than baseplot.

```
ggplot(spp_means, aes(x = Species, y = mean))+
  geom_col()+
  geom_errorbar(aes(ymin = mean - SE, ymax = mean + SE), size = 1, width = 0.5)
```



ggplot: Bargraph

```
ggplot(spp_means, aes(x = Species, y = mean))+
  geom_col(fill = "white", colour = "black")
```



Plotting in R

1. Two ways to plot in R
2. Plotting with baseplot
   - Scatter plot
   - Boxplot
   - Histograms
   - Bargraphs
   - Aesthetics
   - Saving your plot
3. Plotting with ggplot
   - Scatter plot
   - Boxplot
   - Histograms
   - Bargraphs
   - Aesthetics
   - Saving your plot
4. Which one to use?

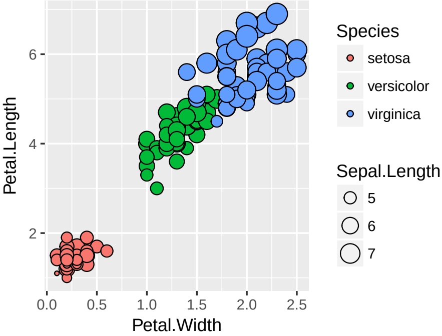
Introduction to themes

In ggplot, you can change aesthetics in individual segments of the code *or* you can change information for the whole plot using theme.

Using the aesthetic argument more

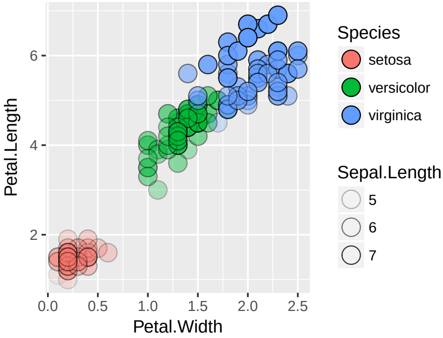
We'll start by looking back at the aesthetic (aes) argument. We can use it to change multiple different aesthetics of a plot.

```
ggplot(iris, aes(x = Petal.Width, y = Petal.Length))+
  geom_point(aes(fill = Species, size = Sepal.Length), shape = 21)
```



Using the aesthetic argument more

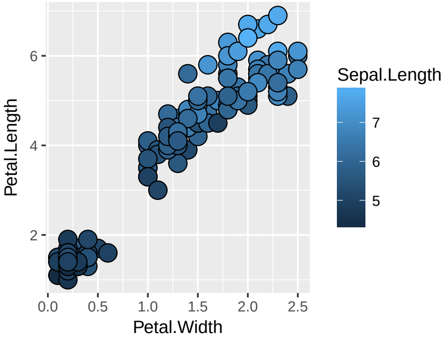
```
ggplot(iris, aes(x = Petal.Width, y = Petal.Length))+
  geom_point(aes(fill = Species, alpha = Sepal.Length), shape = 21, size = 5)
```



Using the aesthetic argument more

Applying aesthetics to continuous variables will be different to categorical variables.

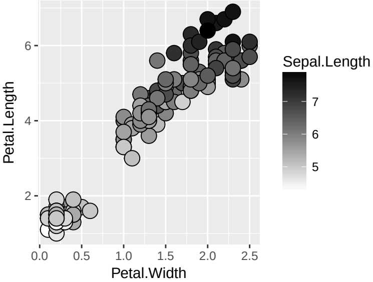
```
ggplot(iris, aes(x = Petal.Width, y = Petal.Length))+
  geom_point(aes(fill = Sepal.Length), shape = 21, size = 5)
```



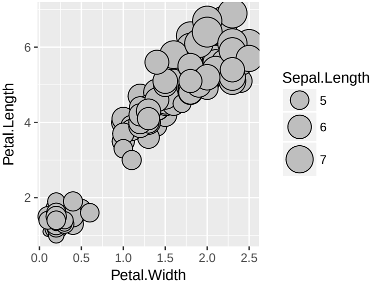
Using the aesthetic argument more

We can adjust the way the aesthetics are applied.

```
ggplot(iris, aes(x = Petal.Width, y = Petal.Length))+
  geom_point(aes(fill = Sepal.Length), shape = 21, size = 5)+
  scale_fill_continuous(low = "white", high = "black")
```



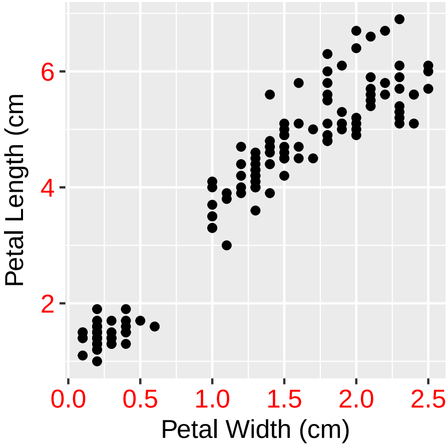
```
ggplot(iris, aes(x = Petal.Width, y = Petal.Length))+
  geom_point(aes(size = Sepal.Length), shape = 21, fill = "grey")+
  scale_size_continuous(range = c(3, 10))
```



Change text

You can change size and colour of axis text easily. Note that text size uses different measurement units.

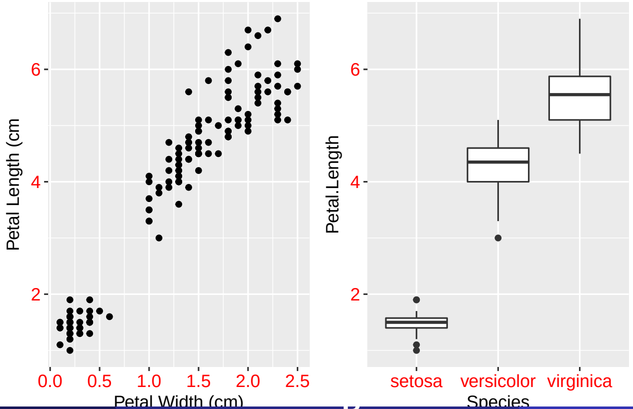
```
ggplot(iris, aes(x = Petal.Width, y = Petal.Length))+
  geom_point()+
  xlab("Petal Width (cm)")+
  ylab("Petal Length (cm")+
  theme(axis.text = element_text(size = 12, colour = "red"),
        axis.title = element_text(size = 12))
```



Combining plots

Combining plots it much less straightforward in ggplot. You need to use an additional package.

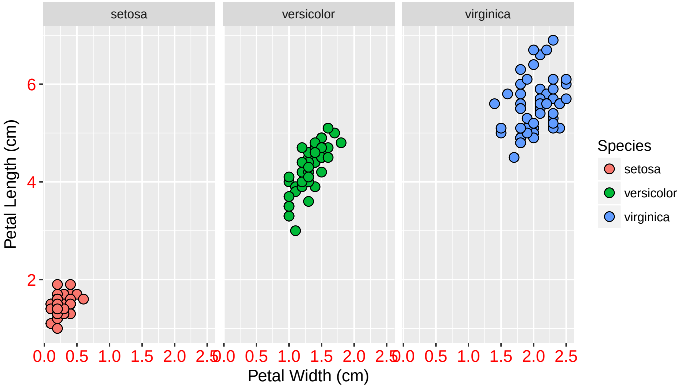
```
scatter <- ggplot(iris, aes(x = Petal.Width, y = Petal.Length))+
  geom_point()+
  xlab("Petal Width (cm)")+
  ylab("Petal Length (cm")+
  theme(axis.text = element_text(size = 12, colour = "red"), axis.title = element_text(size = 12))
box     <- ggplot(iris, aes(x = Species, y = Petal.Length))+
  geom_boxplot()+
  theme(axis.text = element_text(size = 12, colour = "red"), axis.title = element_text(size = 12))
gridExtra::grid.arrange(scatter, box, nrow = 1)
```



Faceting

Although combining multiple plots is cumbersome, there is an inbuilt option to create facets.

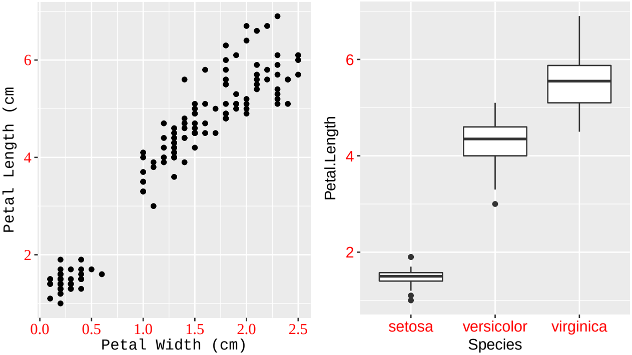
```
ggplot(iris, aes(x = Petal.Width, y = Petal.Length))+
  geom_point(aes(fill = Species), shape = 21, colour = "black", size = 3)+
  xlab("Petal Width (cm)")+
  ylab("Petal Length (cm)")+
  facet_wrap(˜Species) +
  theme(axis.text = element_text(size = 12, colour = "red"), axis.title = element_text(size = 12))
```



Text family

Unlike baseplot, in ggplot you can change font family of individual elements in theme.

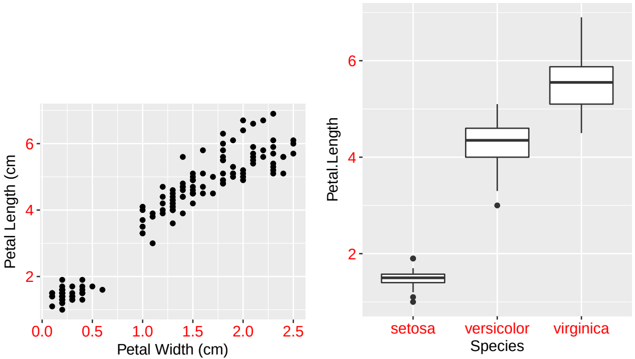
```
scatter <- ggplot(iris, aes(x = Petal.Width, y = Petal.Length))+
  geom_point()+
  xlab("Petal Width (cm)")+
  ylab("Petal Length (cm")+
  theme(axis.text = element_text(size = 12, colour = "red", family = "serif"), axis.title = element_text(size = 12, family = "mono"))
box     <- ggplot(iris, aes(x = Species, y = Petal.Length))+
  geom_boxplot()+
  theme(axis.text = element_text(size = 12, colour = "red", family = "sans"), axis.title = element_text(size = 12, family = "sans"))
gridExtra::grid.arrange(scatter, box, nrow = 1)
```



Plot margins

Plot margins are also controlled in theme of each plot individually.

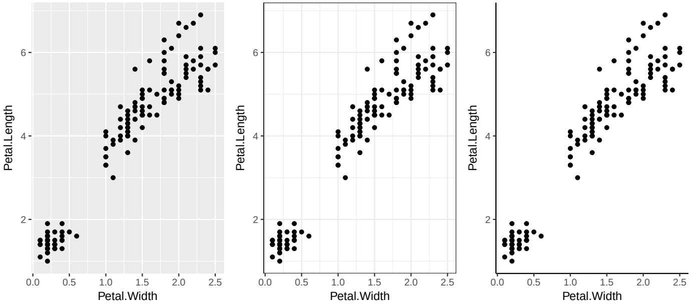
```r
scatter <- ggplot(iris, aes(x = Petal.Width, y = Petal.Length))+
  geom_point()+
  xlab("Petal Width (cm)")+
  ylab("Petal Length (cm")+
  theme(axis.text = element_text(size = 12, colour = "red"), axis.title = element_text(size = 12),
        plot.margin = unit(c(4, 4, 1, 1), "mm"))
box     <- ggplot(iris, aes(x = Species, y = Petal.Length))+
  geom_boxplot()+
  theme(axis.text = element_text(size = 12, colour = "red"), axis.title = element_text(size = 12))
gridExtra::grid.arrange(scatter, box, nrow = 1)
```



Preset themes

ggplot also has a number of preset themes that you can use.

```
grey <- ggplot(iris, aes(x = Petal.Width, y = Petal.Length))+
  geom_point()+
  theme_grey()
bw <- ggplot(iris, aes(x = Petal.Width, y = Petal.Length))+
  geom_point()+
  theme_bw()
classic <- ggplot(iris, aes(x = Petal.Width, y = Petal.Length))+
  geom_point()+
  theme_classic()
gridExtra::grid.arrange(grey, bw, classic, nrow = 1)
```



Plotting in **R**

1. Two ways to plot in **R**
2. Plotting with baseplot
   - Scatter plot
   - Boxplot
   - Histograms
   - Bargraphs
   - Aesthetics
   - Saving your plot
3. Plotting with ggplot
   - Scatter plot
   - Boxplot
   - Histograms
   - Bargraphs
   - Aesthetics
   - Saving your plot
4. Which one to use?

ggplot: Exporting

```
?ggsave
```

```
classic <- ggplot(iris, aes(x = Petal.Width, y = Petal.Length))+
  geom_point()+
  theme_classic()
ggsave("ggplot.pdf", plot = classic, width = 15, height = 5)
```

Plotting in R

1. Two ways to plot in R
2. Plotting with baseplot
3. Plotting with ggplot
4. Which one to use?

Which plotting tool should you use?

Baseplot:

- No new packages required
- Easy to combine many plots
- Looks good out of the box (but harder to customise)

Which plotting tool should you use?

Baseplot:

- No new packages required
- Easy to combine many plots
- Looks good out of the box (but harder to customise)

ggplot:

- Requires multiple packages for best results
- More difficult to combine many plots
- Easy to customise (but looks poor out of the box)
- Faster. Better for bigger datasets

Which plotting tool should you use?

It is useful to know both!!