# Plots

Pedro J. Aphalo

31 March 2015

# Slide with R Code and Output

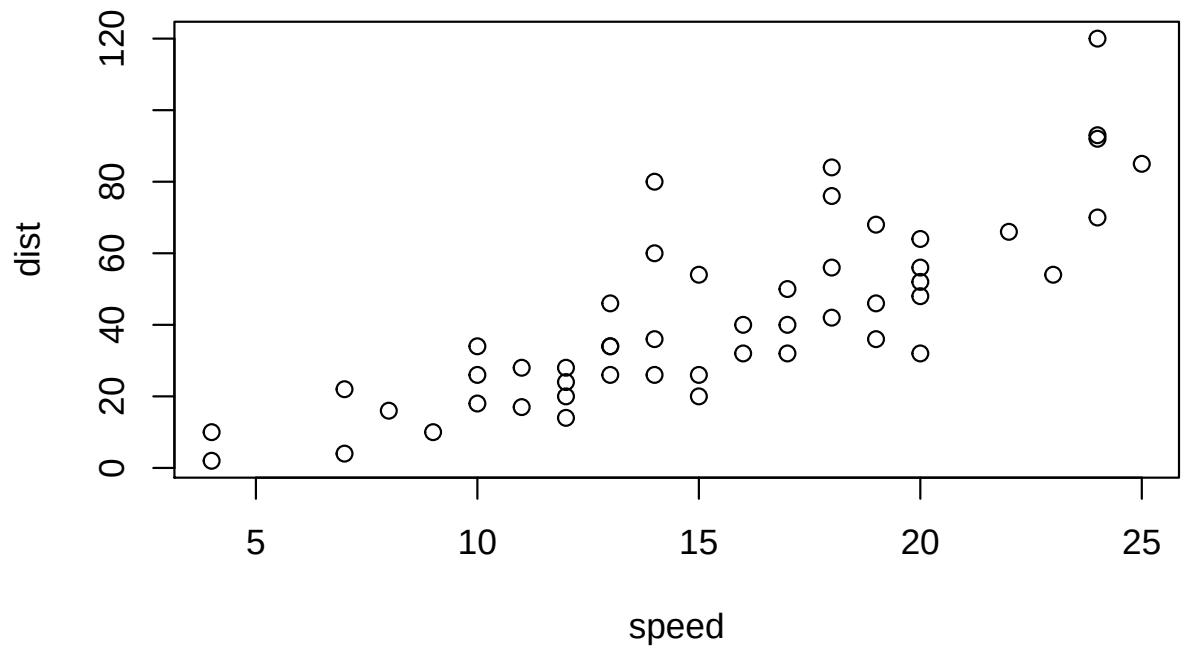
```
summary(cars)
```

```
##      speed           dist       
##  Min.   : 4.0   Min.   :  2.00  
##  1st Qu.:12.0   1st Qu.: 26.00  
##  Median :15.0   Median : 36.00  
##  Mean   :15.4   Mean   : 42.98  
##  3rd Qu.:19.0   3rd Qu.: 56.00  
##  Max.   :25.0   Max.   :120.00
```

# Slide with Plot

A very simple plot using example data included in R and base R plot functions

```
plot(cars)
```



# ggplot2

Nowadays the package most frequently used for *normal* plotting is `ggplot2`

- it is based on a grammar of graphics
- in plain words, we assemble plots by adding different *pieces*
- these pieces are quite modular
- this allows making many different types of plots from not so many *pieces*
- the basic pieces are *layers*
- the name reflects the fact that order in which we add them is significant
- later additions are plotted on top of the first ones

# what is a ggplot object?

- a `ggplot` object is independent of the output format
- it is not purely visual, it is to a significant extent semantic
- the size and the `theme` can be decided at the time of generating output
- a `ggplot` can be modified by adding and sometimes replacing elements
- a `ggplot` object can be large (in memory) because it contains a copy of the data plotted
- this means that the `ggplot` object is self-sufficient (you can copy it to a new 'empty' R session and still generate the output)

# What are the words of the grammar

- data
- aesthetics
- statistics
- scales
- geometries

# data

- `data` tells R where to look for the variables used in the plot, data is supplied as a `data.frame` object
- *aesthetics* (`aes`) tells how to map variables to features in the plot
- stablishes a connection between a variable an aesthetic *dimension*
- for example: `x` coordinate, `colour`, etc.
- *statistics* or some operation on the data before plotting _ *scales* the mapping between data values and aesthetic values
- e.g. is value 1 by plotted as red and value 2 with say blue, or using some other colours

# geometries

- they are easier to describe with examples
- `geom_line`, `geom_point`, `geom_text`
- each geom you add to a ggplot adds a new layer to the plot (on top of the layers added earlier)

# annotations

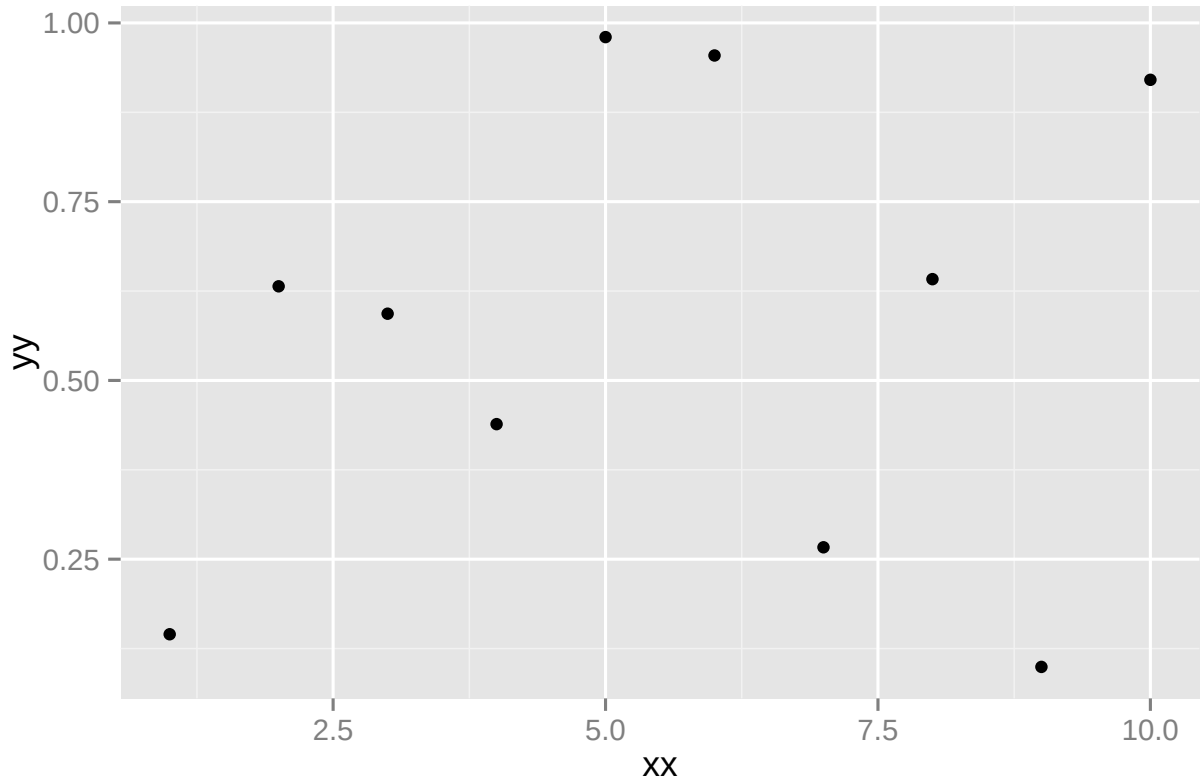
- they add layers that are not directly 'connected' to the data
- e.g. a label not directly related to a single observation
- annotations can also use different geoms and aesthetics, buy they are not inherited, they are selected locally to each annotation

# A very simple plot

```
library(ggplot2)
# runif generates random numbers
# we create a data frame to play with
my.df <- data.frame(xx = 1:10, yy = runif(10))
# we create a plot and save it as 'my.plot'
my.plot <- ggplot(data = my.df, aes(x = xx, y = yy)) +
  geom_point()
```

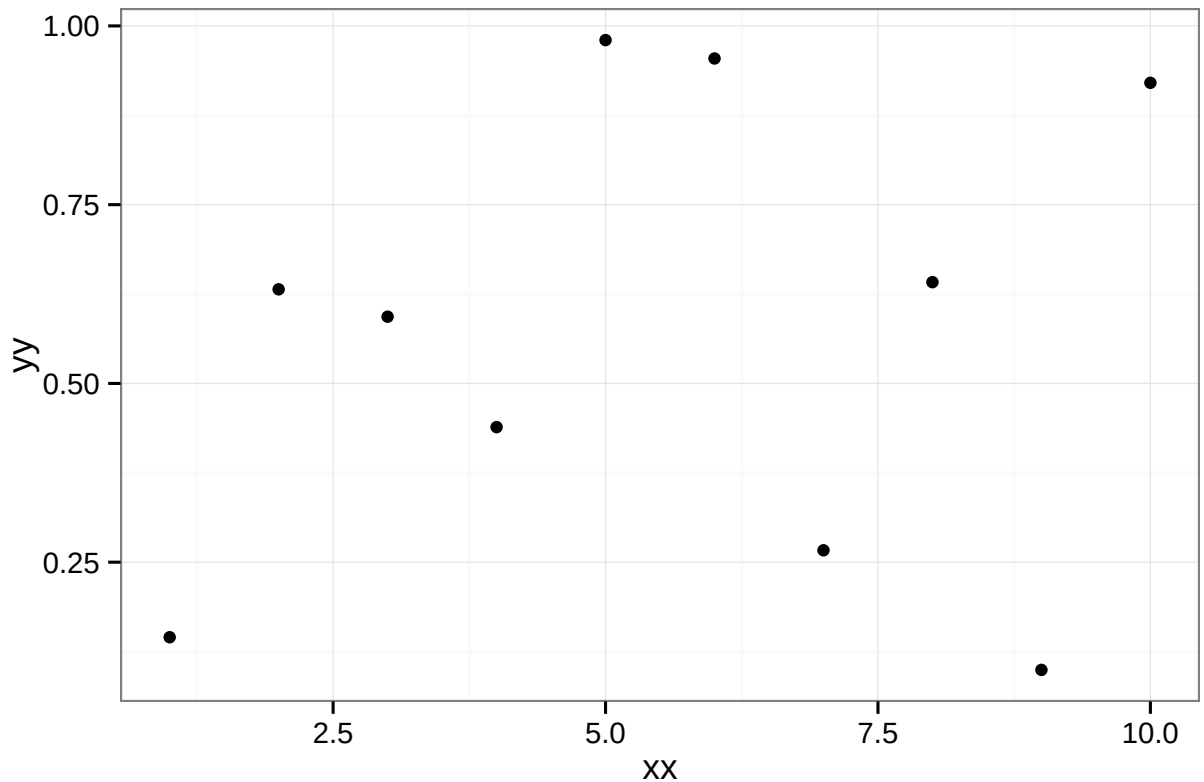
# *Printing* the plot

```
print(my.plot)
```



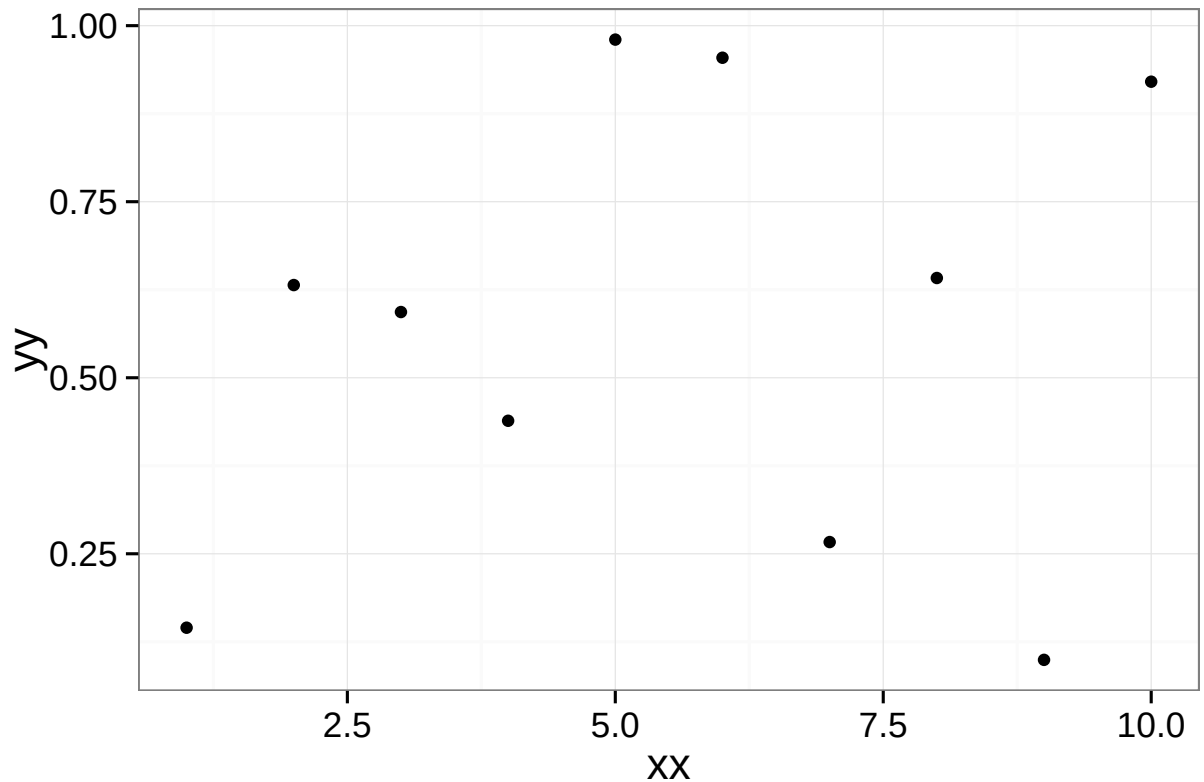
# *Printing* the plot using a different `theme`

```
print(my.plot + theme_bw())
```



# *Printing* the plot using a different base font size

```
print(my.plot + theme_bw(15))
```



# What is in my.plot

```
str(my.plot)
```

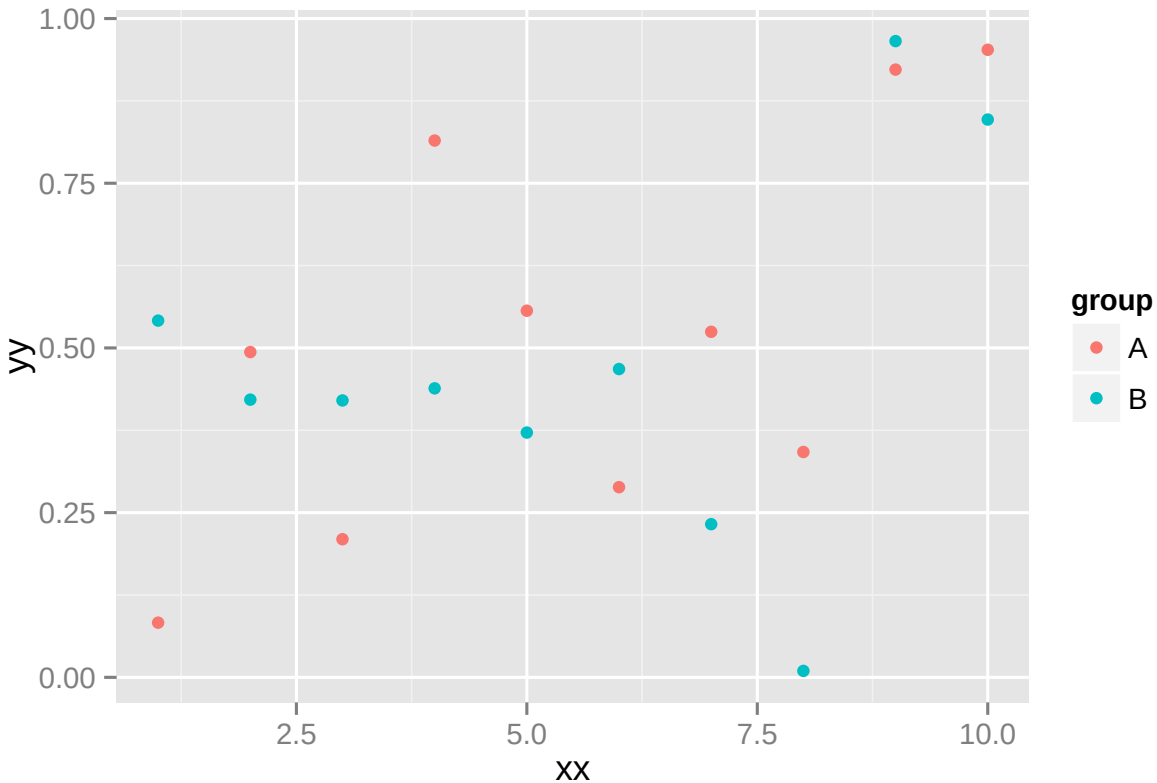
```
## List of 9
##  $ data       :'data.frame': 10 obs. of  2 variables:
##   ..$ xx: int [1:10] 1 2 3 4 5 6 7 8 9 10
##   ..$ yy: num [1:10] 0.145 0.632 0.593 0.439 0.98 ...
##  $ layers     :List of 1
##   ..$ :Classes 'proto', 'environment' <environment: 0x0(
##  $ scales     :Reference class 'Scales' [package "ggplot
##   ..$ scales: list()
##   ..and 21 methods, of which 9 are possibly relevant:
##   ..  add, clone, find, get_scales, has_scale, initializ
##   ..  non_position_scales
##  $ mapping    :List of 2
##   ..$ x: symbol xx
##   ..$ y: symbol yy
##  $ theme      : list()
##  $ coordinates:List of 1
```

# A not so simple plot

```
# runif generates random numbers
# we create a data frame to play with
my.2nd.df <-
  data.frame(xx = rep(1:10, 2),
             yy = runif(20),
             group = factor(rep(c("A", "B"), c(10, 10))) )
# we create a plot and save it as 'my.plot'
my.2nd.plot <- ggplot(data = my.2nd.df,
                      aes(x = xx, y = yy, colour = group))
  geom_point()
```

# *Printing* the plot

```
print(my.2nd.plot)
```

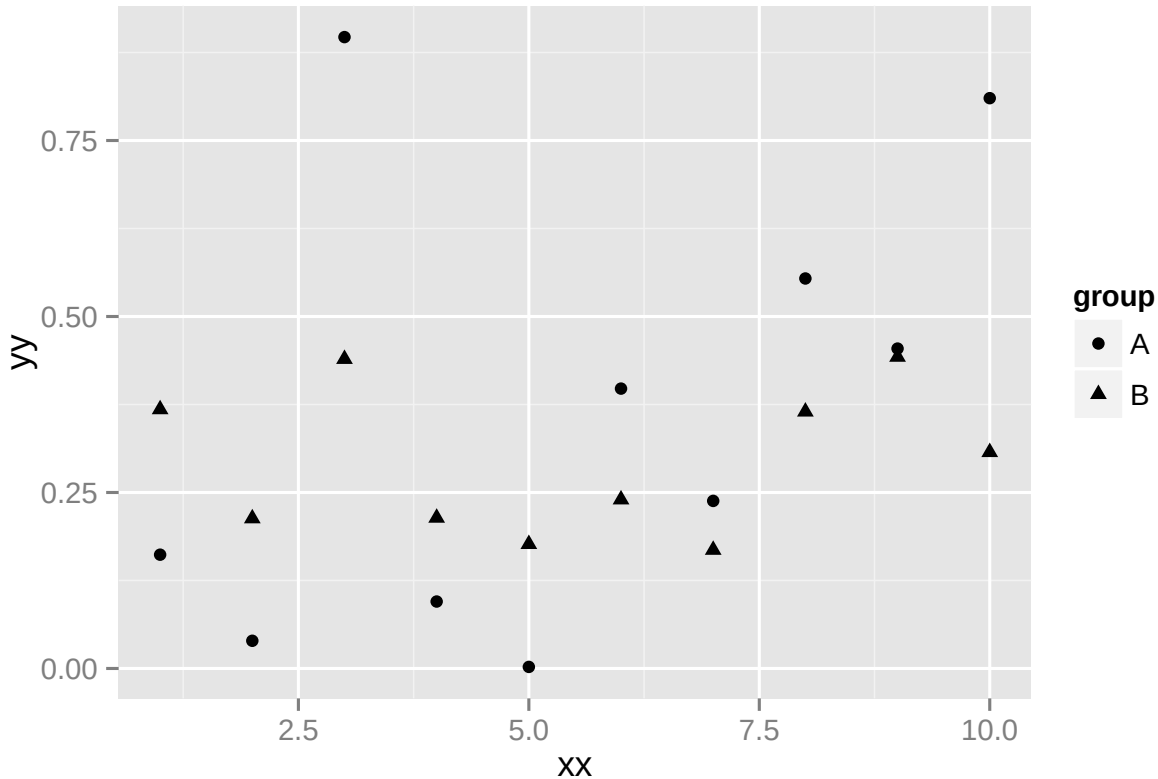


# A not so simple plot

```
# runif generates random numbers
# we create a data frame to play with
my.2nd.df <-
  data.frame(xx = rep(1:10, 2),
             yy = runif(20),
             group = factor(rep(c("A", "B"), c(10, 10))) )
# we create a plot and save it as 'my.plot'
my.3rd.plot <- ggplot(data = my.2nd.df,
                      aes(x = xx, y = yy, shape = group)) +
  geom_point()
```

# *Printing* the plot

```
print(my.3rd.plot)
```

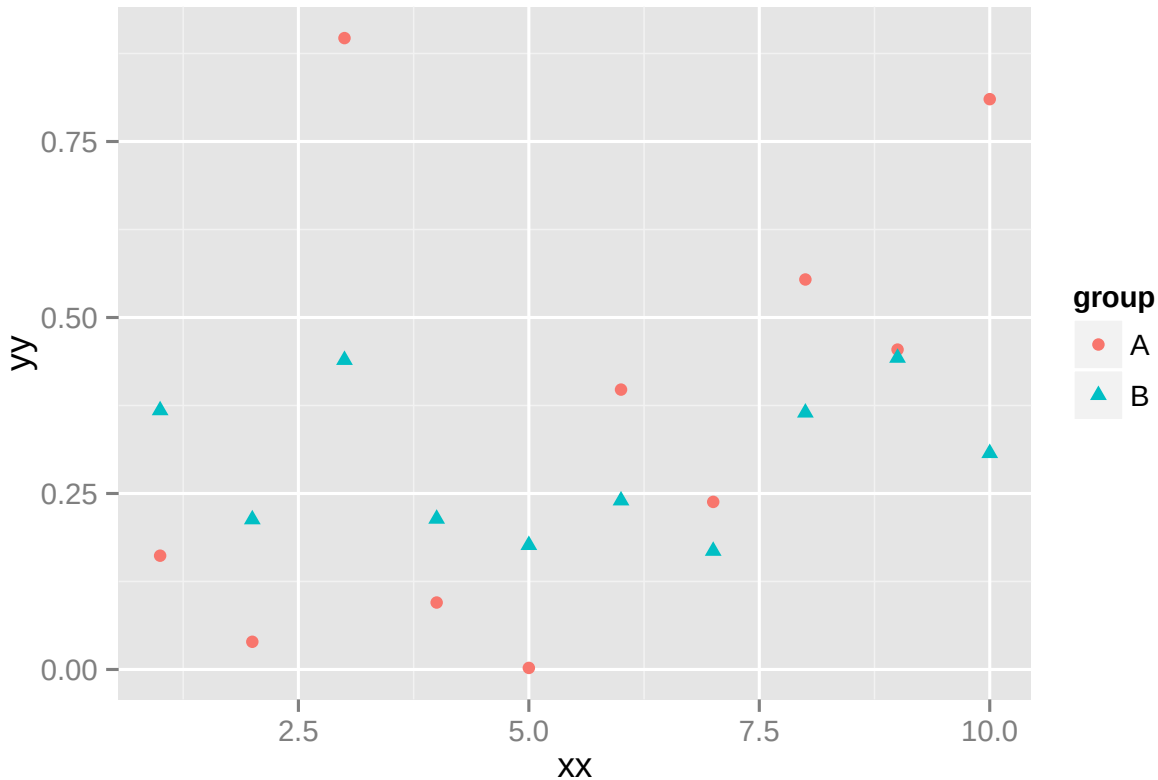


# A plot with two aesthetics for group

```
# we create a plot and save it as 'my.plot'
my.4th.plot <- ggplot(data = my.2nd.df,
                      aes(x = xx, y = yy,
                          shape = group,
                          colour = group)) +
  geom_point()
```

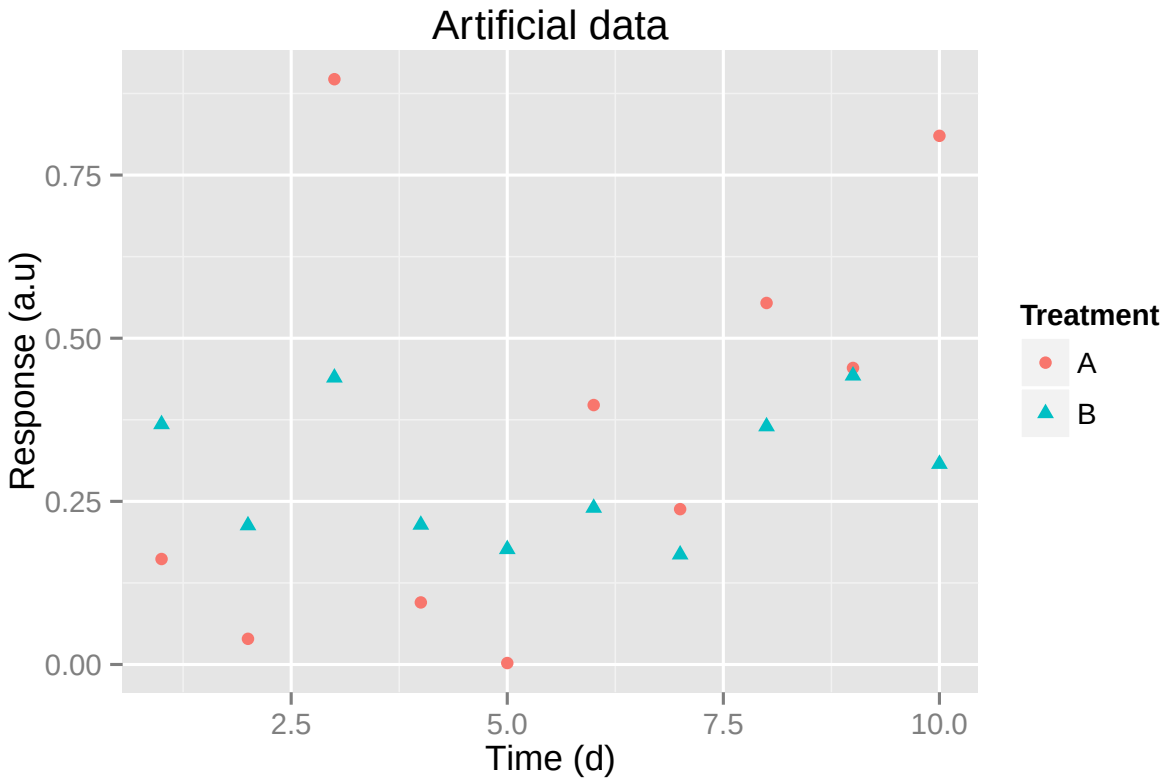
# *Printing* the plot

```
print(my.4th.plot)
```



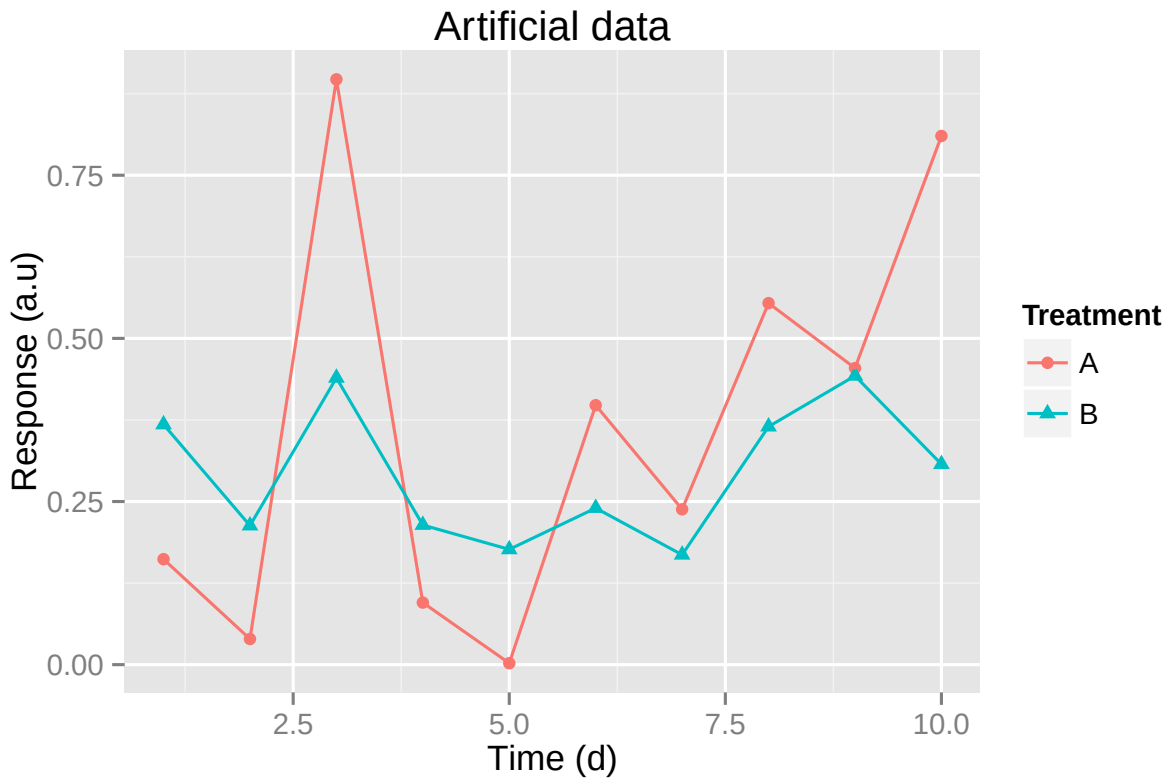
# Replacing the default labels

```
my.5th.plot <- my.4th.plot +
  labs(x="Time (d)", y = "Response (a.u)",
       colour = "Treatment", shape = "Treatment",
       title = "Artificial data")
print(my.5th.plot)
```



# Adding another layer on the fly

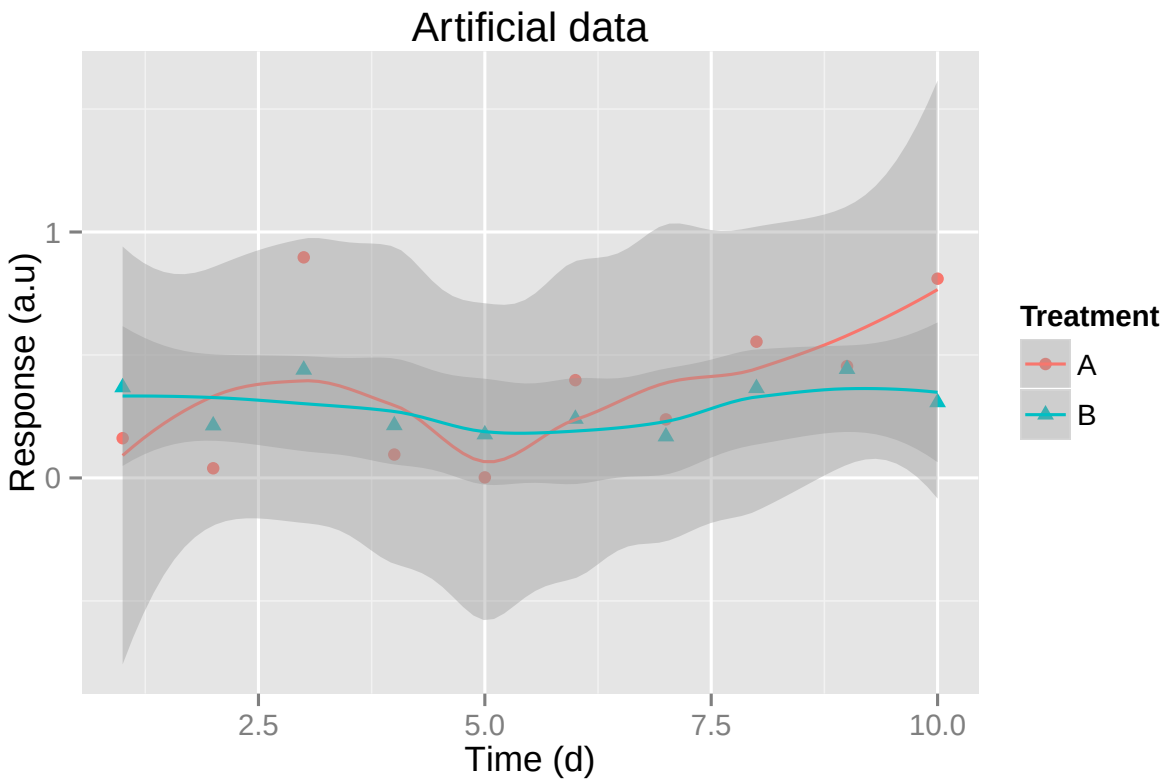
```
print(my.5th.plot + geom_line())
```



# Adding another layer on the fly

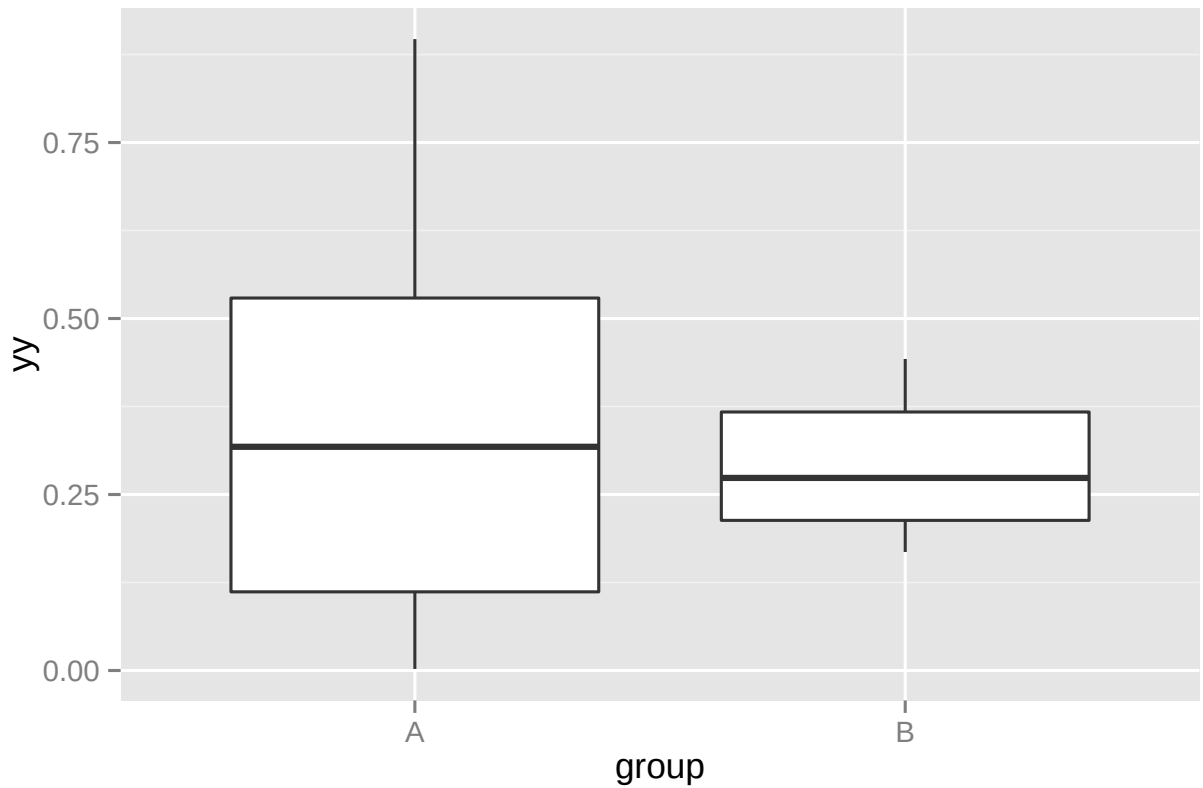
```
print(my.5th.plot + geom_smooth())
```

```
## geom_smooth: method="auto" and size of largest group is
```



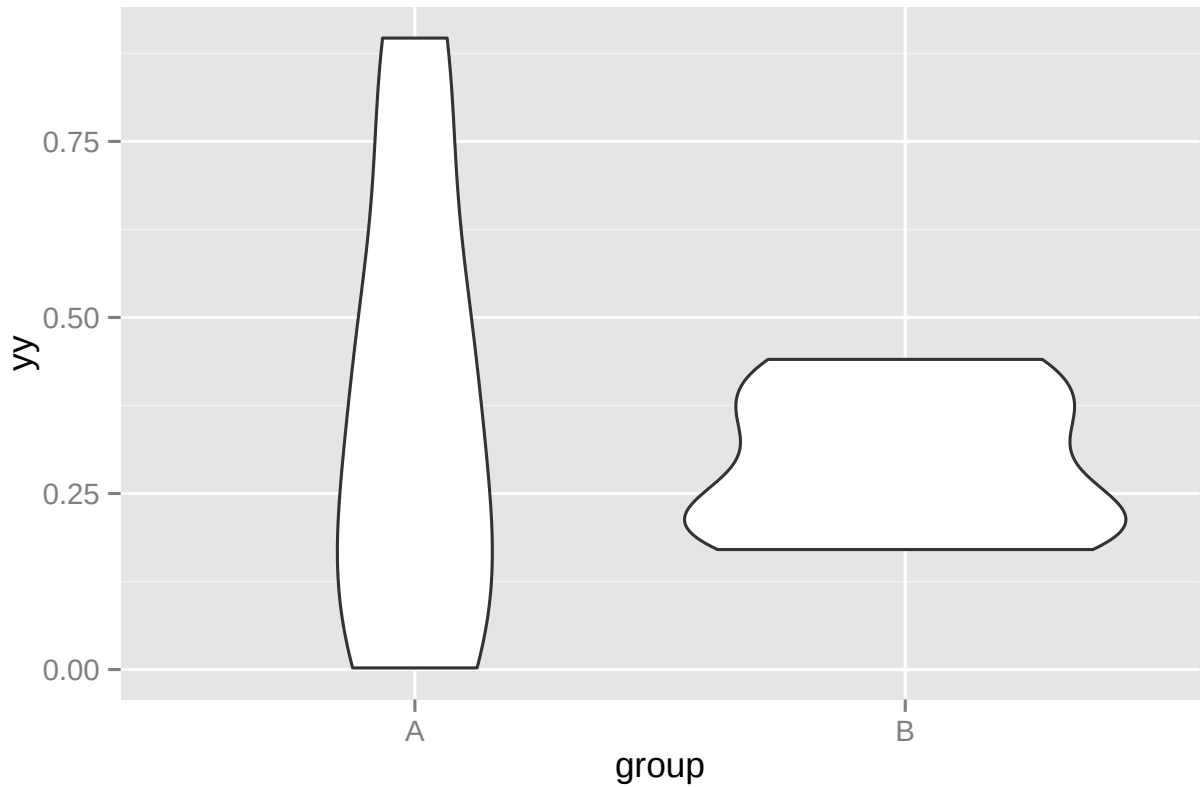
# Another way of plotting the same data

```
# we create a plot and save it as 'my.plot'
my.6th.plot <- ggplot(data = my.2nd.df,
                      aes(x = group, y = yy))
print(my.6th.plot + geom_boxplot())
```



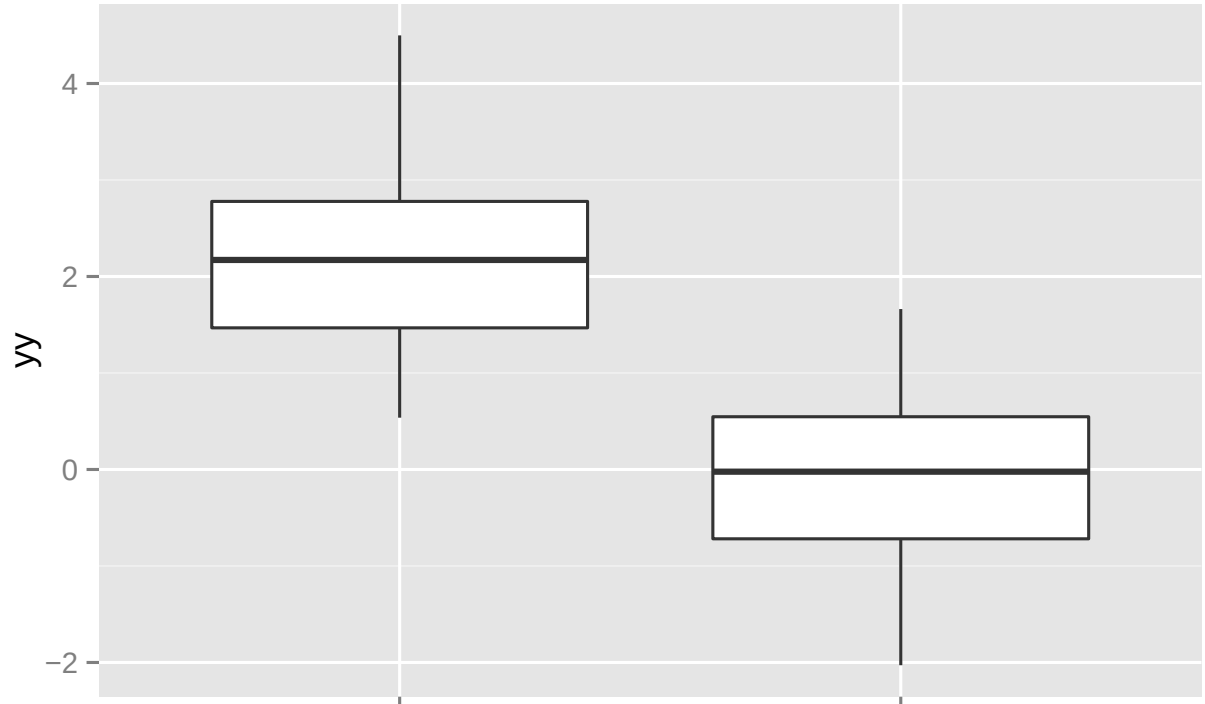
# Another way of plotting the same data

```
print(my.6th.plot + geom_violin())
```



# A larger set of normally distributed data

```
# rnorm generates random numbers
my.3rd.df <-
  data.frame(yy = c(rnorm(n = 50, mean = 2), rnorm(n = 50,
                group = factor(rep(c("A", "B"), c(50, 50))) )
my.7th.plot <- ggplot(data = my.3rd.df,
                      aes(x = group, y = yy))
print(my.7th.plot + geom_boxplot())
```



# The plot with a few additional layers

```
print(my.7th.plot +
        geom_violin() +
        geom_point(aes(colour = group)) +
        geom_rug(aes(colour = group),
                 side = "l", alpha = 0.5) )
```