Plotting in R

Susan Holmes

```r
setwd("~/RWork")
library("dplyr")
```

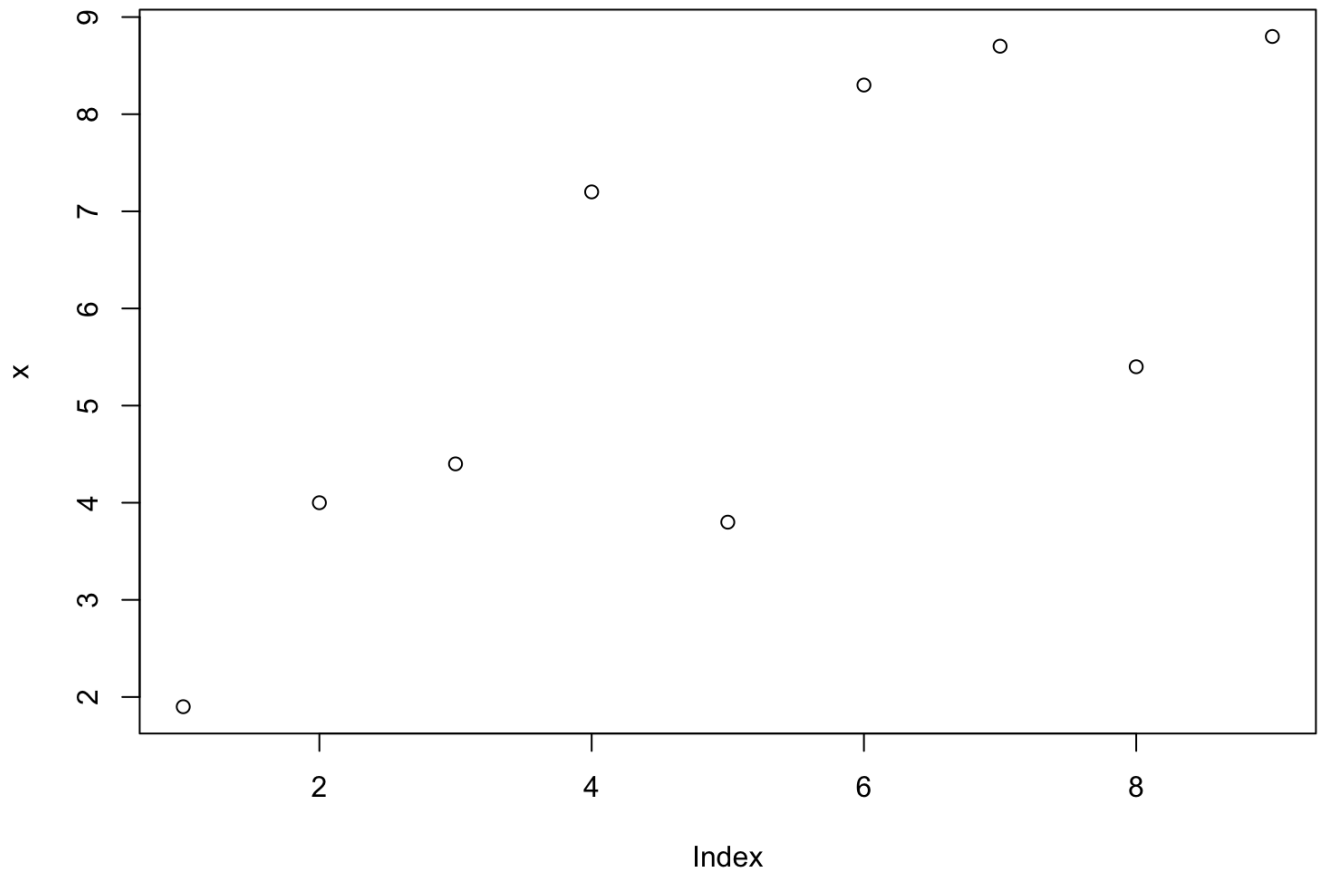
```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

Plotting in base R

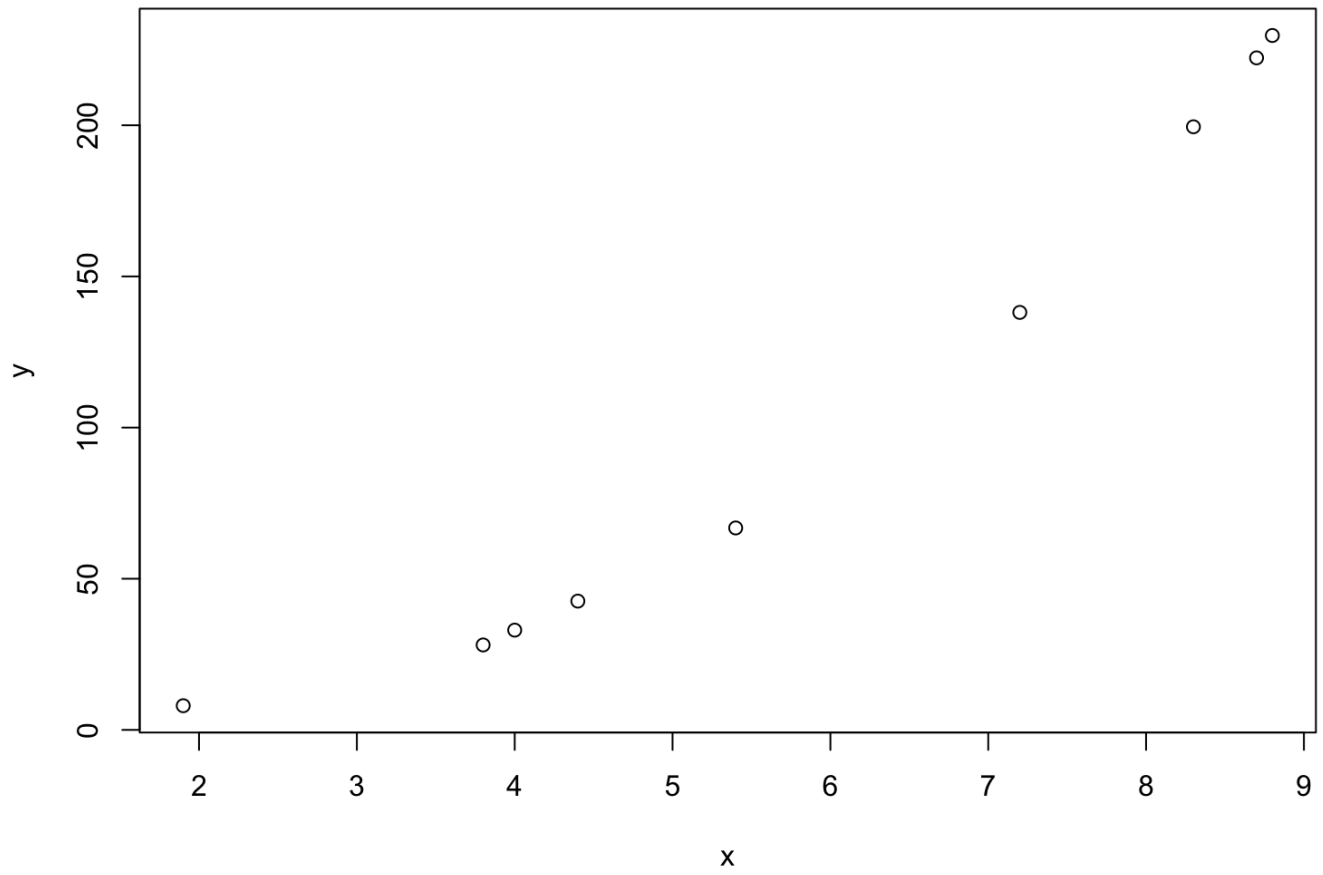
```r
x <- c(1.9, 4.0, 4.4, 7.2, 3.8, 8.3, 8.7, 5.4, 8.8)
plot(x)
```



What does this plot?

The index as the horizontal axis and the values on the vertical.

```r
y<- c(8, 33, 42.6, 138.1, 28.1, 199.5, 222.3, 66.8, 229.7)
plot(x,y)
```



Plot y (y-axis) versus x (x-axis) in a new window

We can make a line instead of points:

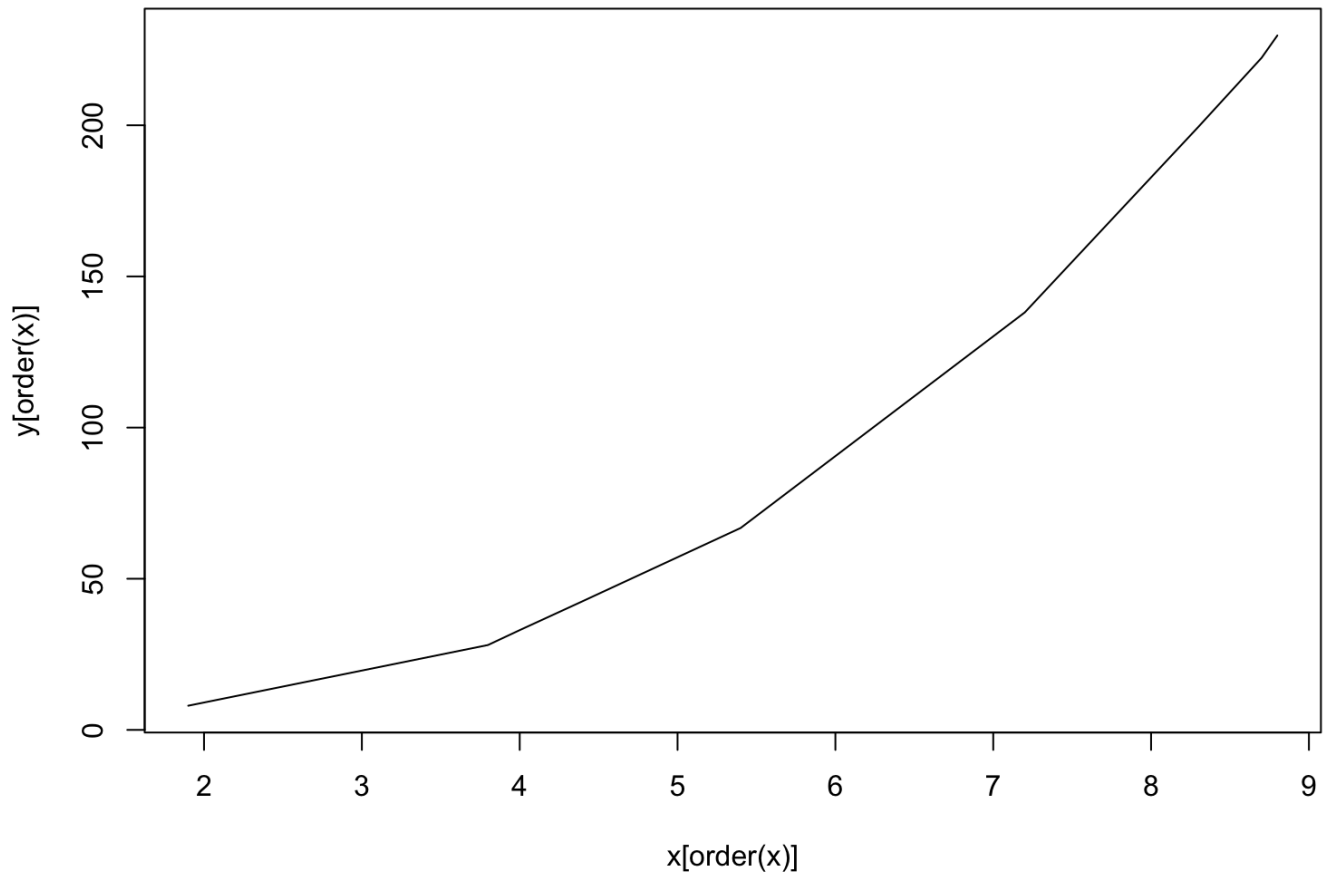
```
x
```

```
## [1] 1.9 4.0 4.4 7.2 3.8 8.3 8.7 5.4 8.8
```

```
order(x)
```

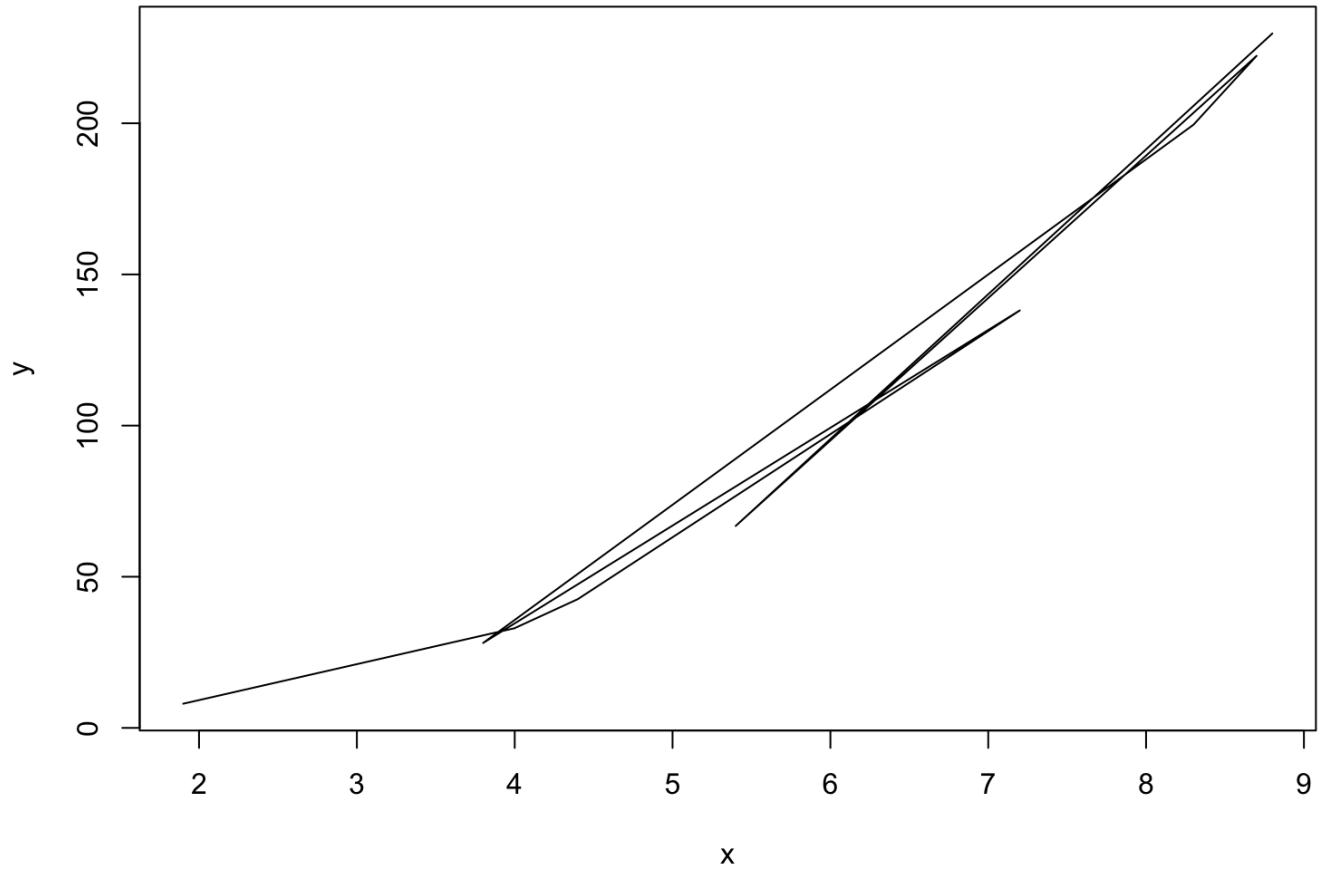
```
## [1] 1 5 2 3 8 4 6 7 9
```

```
plot(x[order(x)],y[order(x)],type='l')
```



What happens if we do not reorder the points?

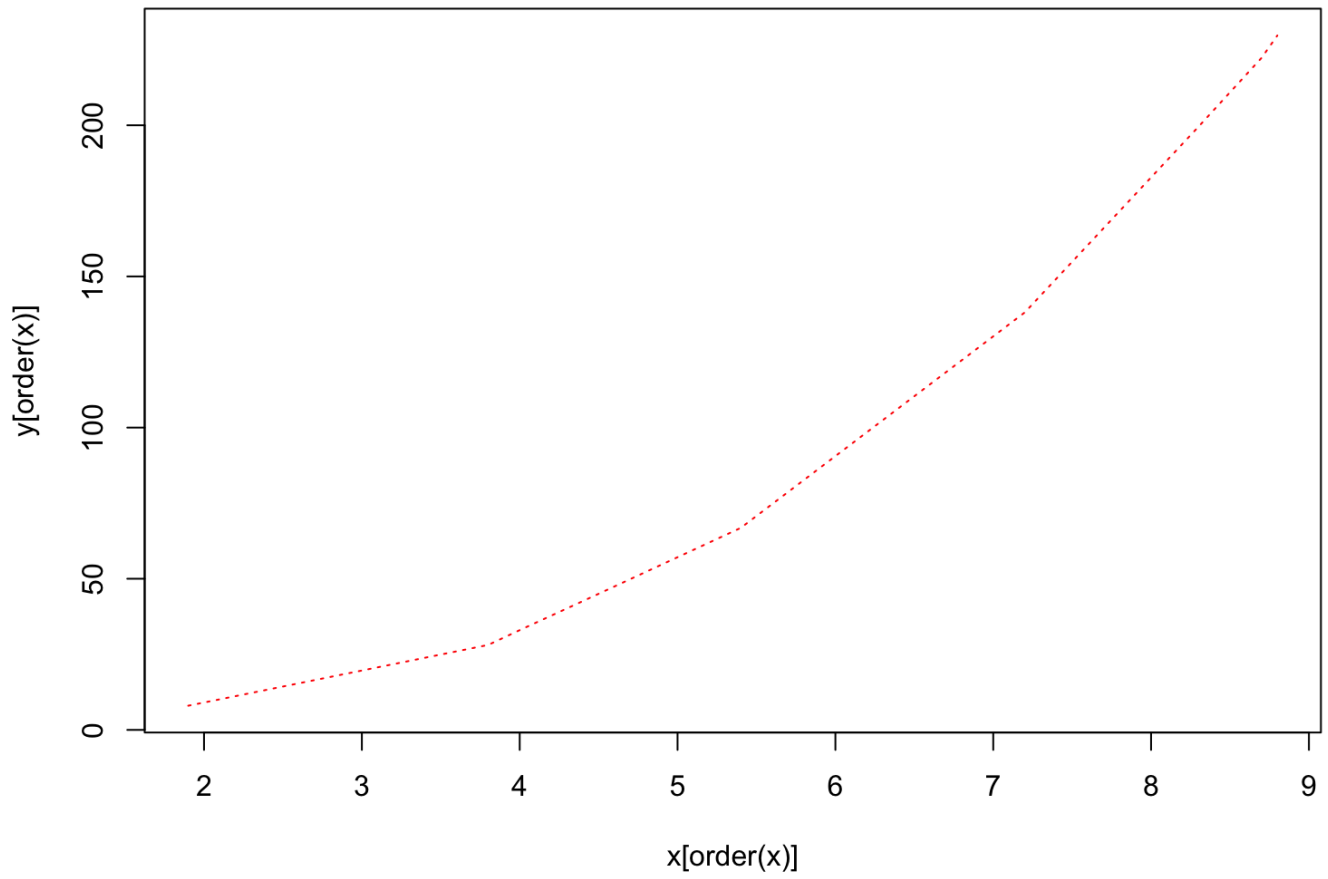
```
plot(x,y,type='l')
```


y
0
50
100
150
200
x
2
3
4
5
6
7
8
9

Extra arguments

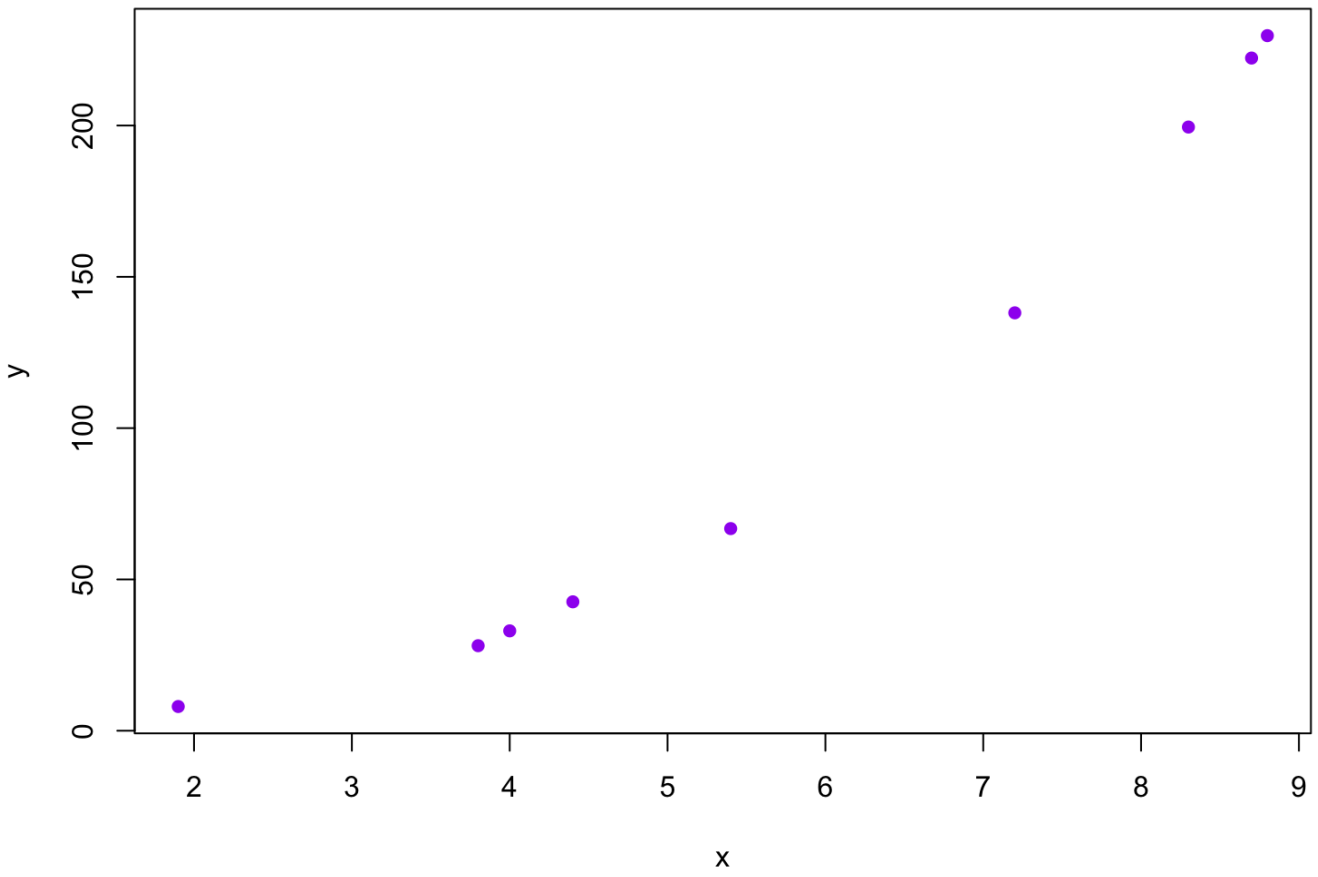
We have extra arguments we can add to plots using:

```
plot(x[order(x)],y[order(x)],type='l',col="red",lty=3)
```

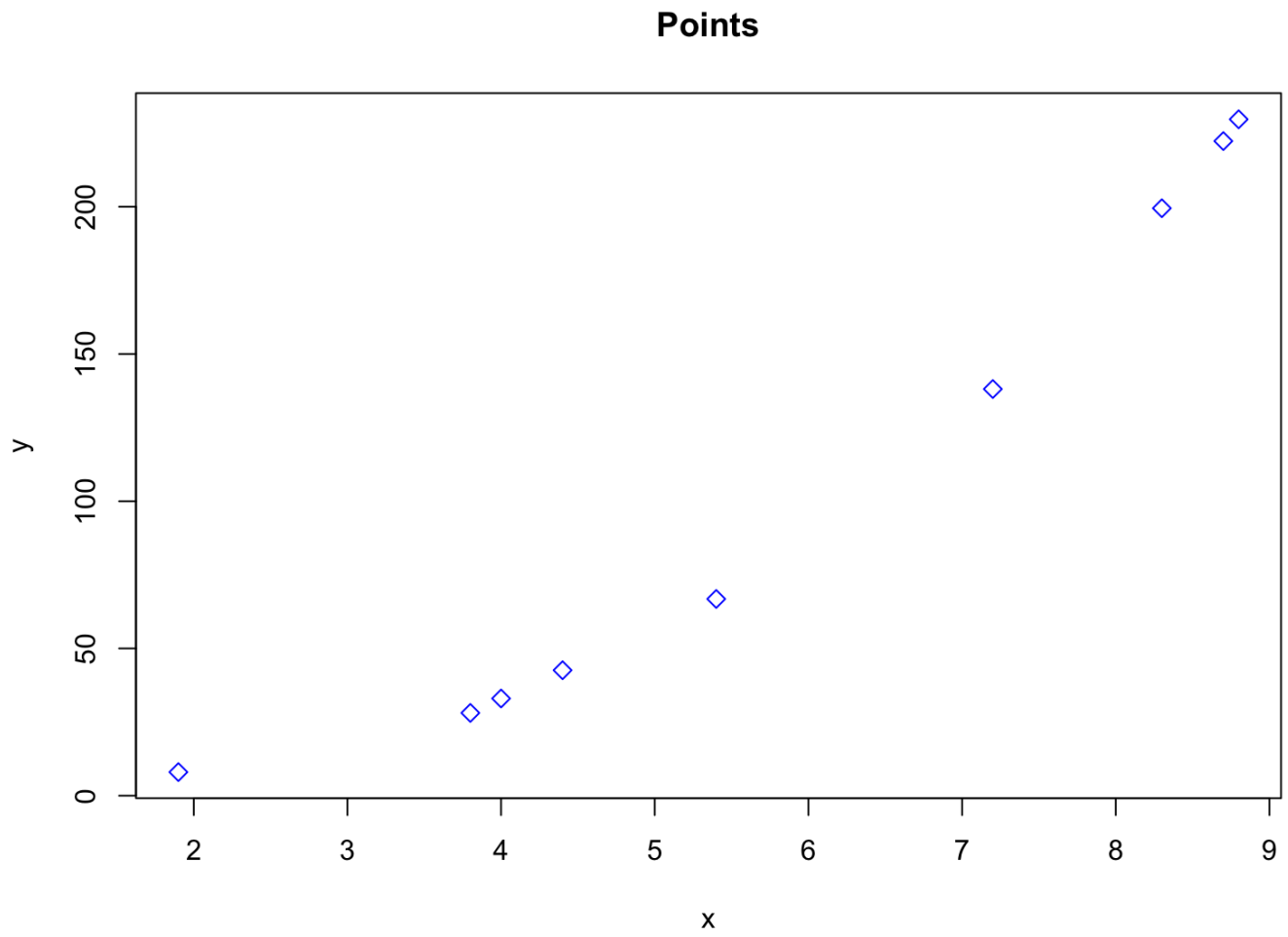


- type: “l”=lines, “p”=points, “n” empty, “b”, etc.
- col: color – “blue”, “red”, etc
- lty: line type – 1=solid, 2=dashed, etc.

```r
plot(x,y,type='p',col="purple",pch=16)
```



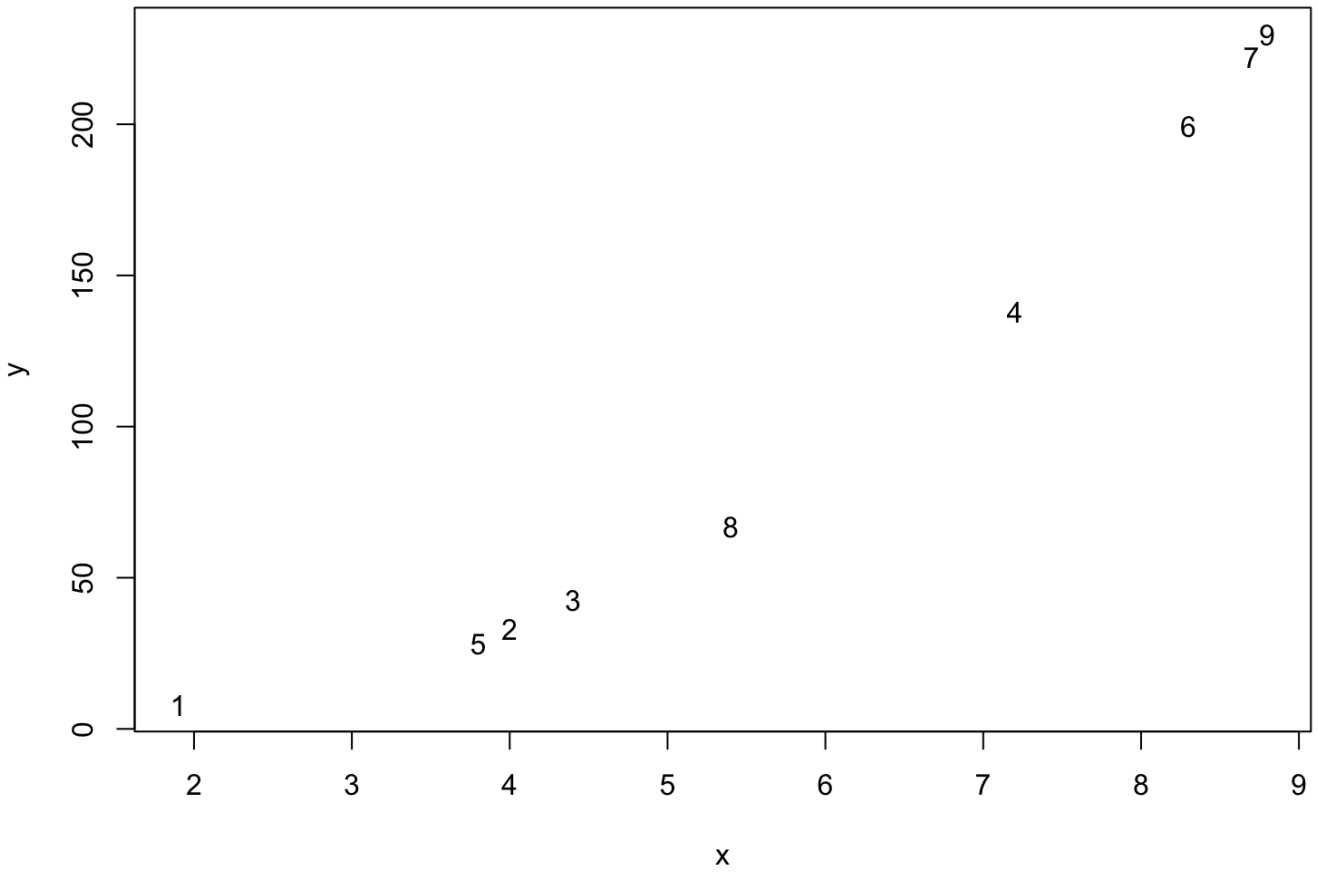

```r
plot(x,y,type='p',col="blue",pch=5,main="Points")
```



- pch: point type – 1=circle, 2=triangle, etc.
- main: title - character string
- xlab and ylab: axis labels – character string

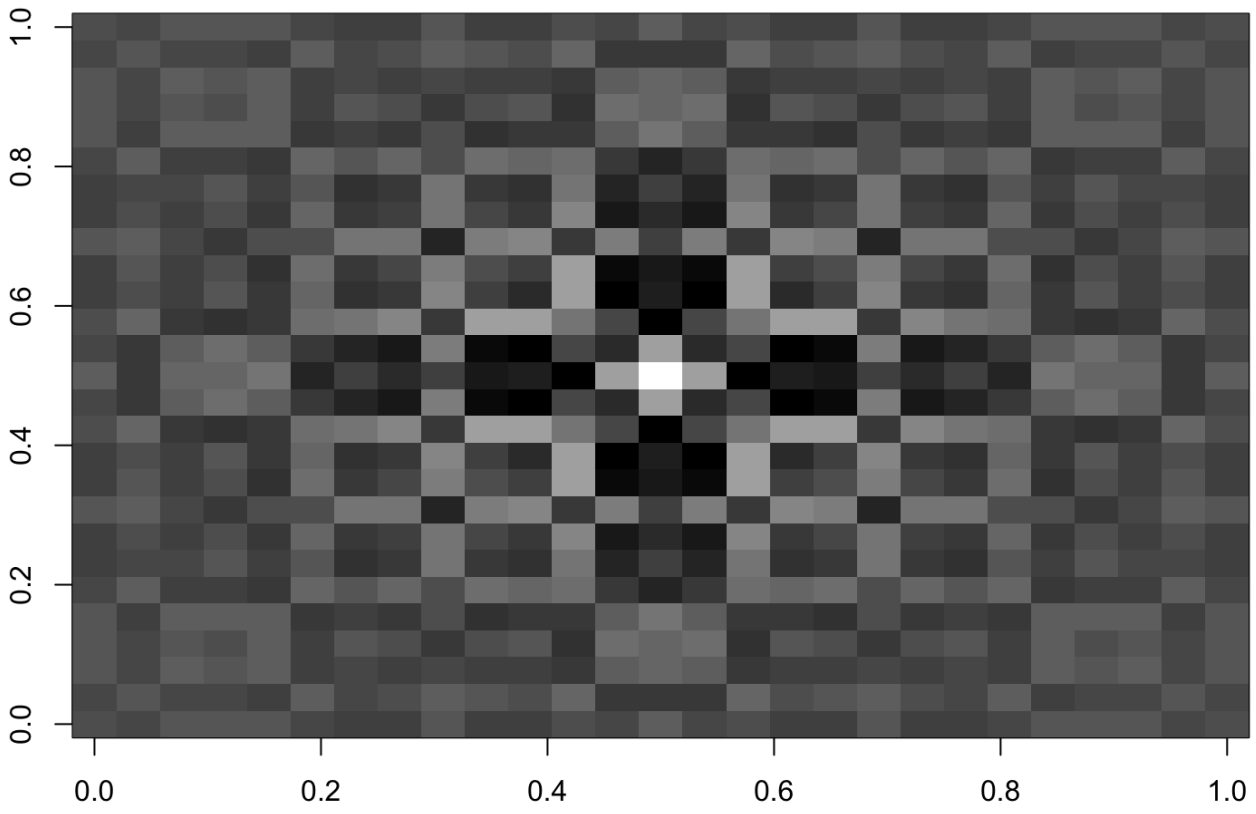
Adding text

```
plot(x,y,type='n')
text(x,y,1:9)
```

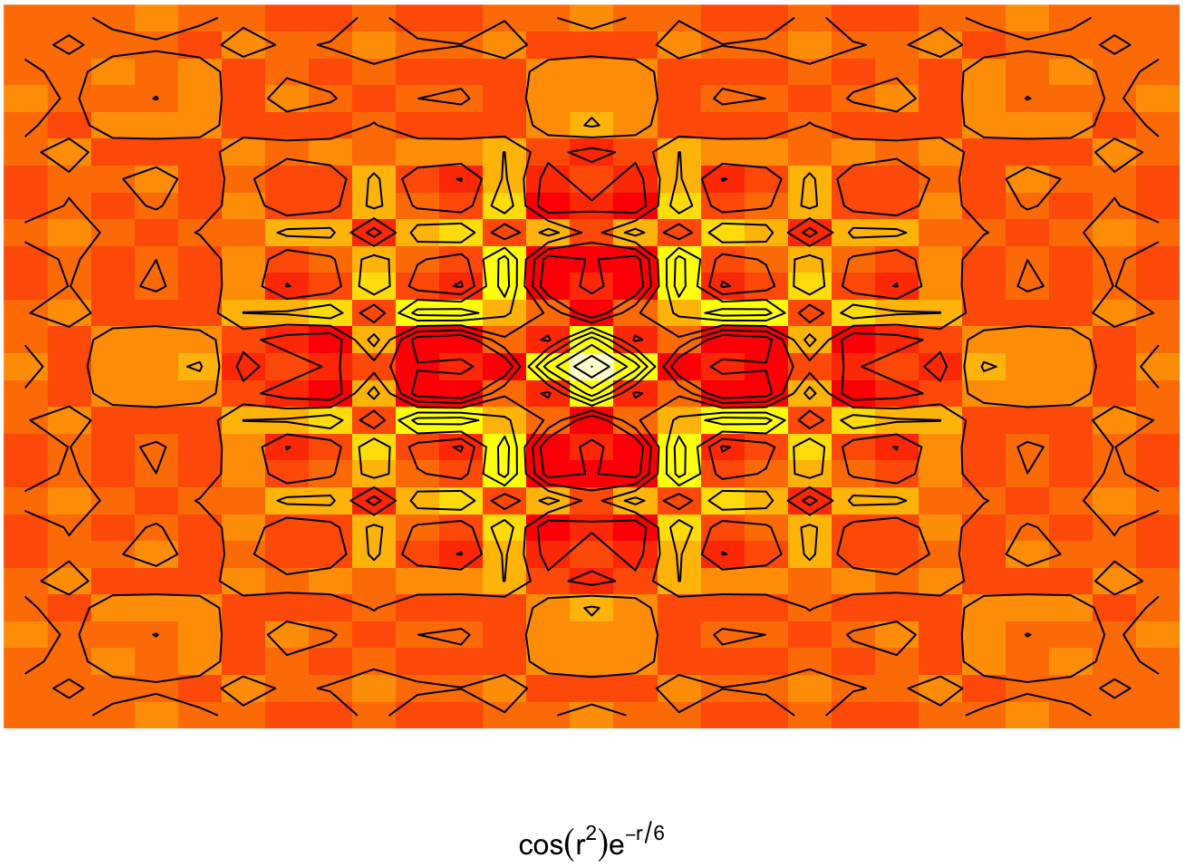


A complicated example for you to decode:

```
require(grDevices) # for colours
x <- y <- seq(-4*pi, 4*pi, len = 27)
r <- sqrt(outer(x^2, y^2, "+"))
image(z = z <- cos(r^2)*exp(-r/6), col  = gray((0:32)/32))
```




```
image(z, axes = FALSE, main = "Math quilt",
      xlab = expression(cos(r^2) * e^{-r/6}))
contour(z, add = TRUE, drawlabels = FALSE)
```



We can also add elements to the plot with the functions:
- legend: add legend with given symbols (lty or pch and col) and text (legend) at location (x="topright")
- axis: add axis (arguments: side – 1=bottom, 2=left, 3=top, 4=right)
- mtext: add text on axis (arguments: text (character string) and side)
- grid: add grid
- par: plotting parameters to be specified before the plots. Arguments: e.g. mfrow=c(1,3)): number of figures per page (1 row, 3 columns); new=TRUE: draw plot over previous plot.

Plot is object-oriented

Meaning it adapts to the objects in the argument (between brackets)

```
apropos("plot")
```

```
##  [1] ".__C__recordedplot"  "assocplot"           "barplot"
##  [4] "barplot.default"     "biplot"              "boxplot"
##  [7] "boxplot.default"     "boxplot.matrix"      "boxplot.stats"
## [10] "cdplot"              "coplot"              "fourfoldplot"
## [13] "interaction.plot"    "lag.plot"            "matplot"
## [16] "monthplot"           "mosaicplot"          "plot"
## [19] "plot.default"        "plot.design"         "plot.ecdf"
## [22] "plot.function"       "plot.new"            "plot.spec.coherency"
## [25] "plot.spec.phase"     "plot.stepfun"        "plot.ts"
## [28] "plot.window"         "plot.xy"             "preplot"
## [31] "qqplot"              "recordPlot"          "replayPlot"
## [34] "savePlot"            "screeplot"           "spineplot"
## [37] "sunflowerplot"       "termplot"            "ts.plot"
```

Other visual summaries

- boxplot
- hist: plot histogram of the numbers in a vector
- barplot: bar plot of vector or data frame

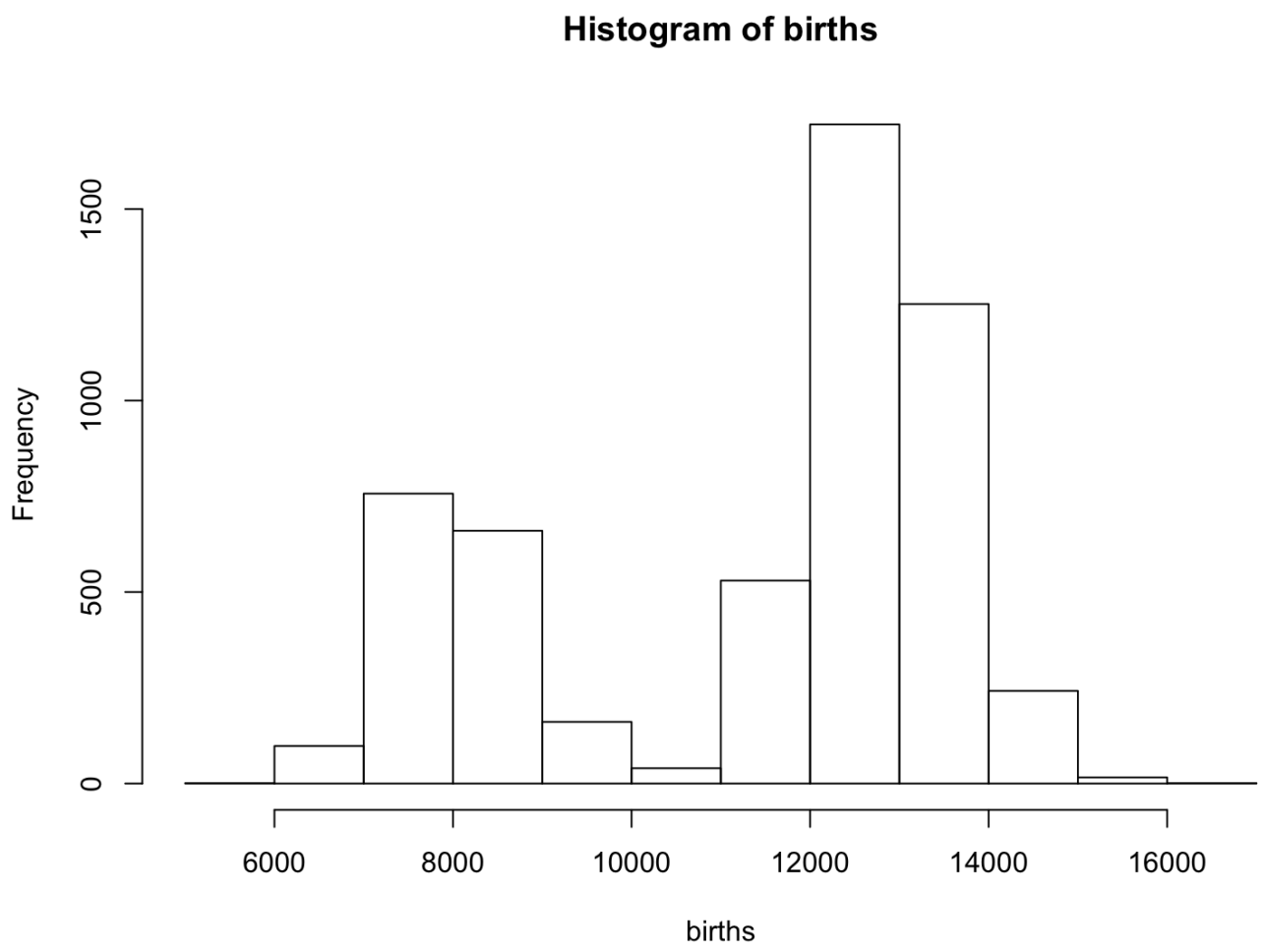
birthn data

```
load("birthn.RData")
attach(birthn)
summary(births)
```

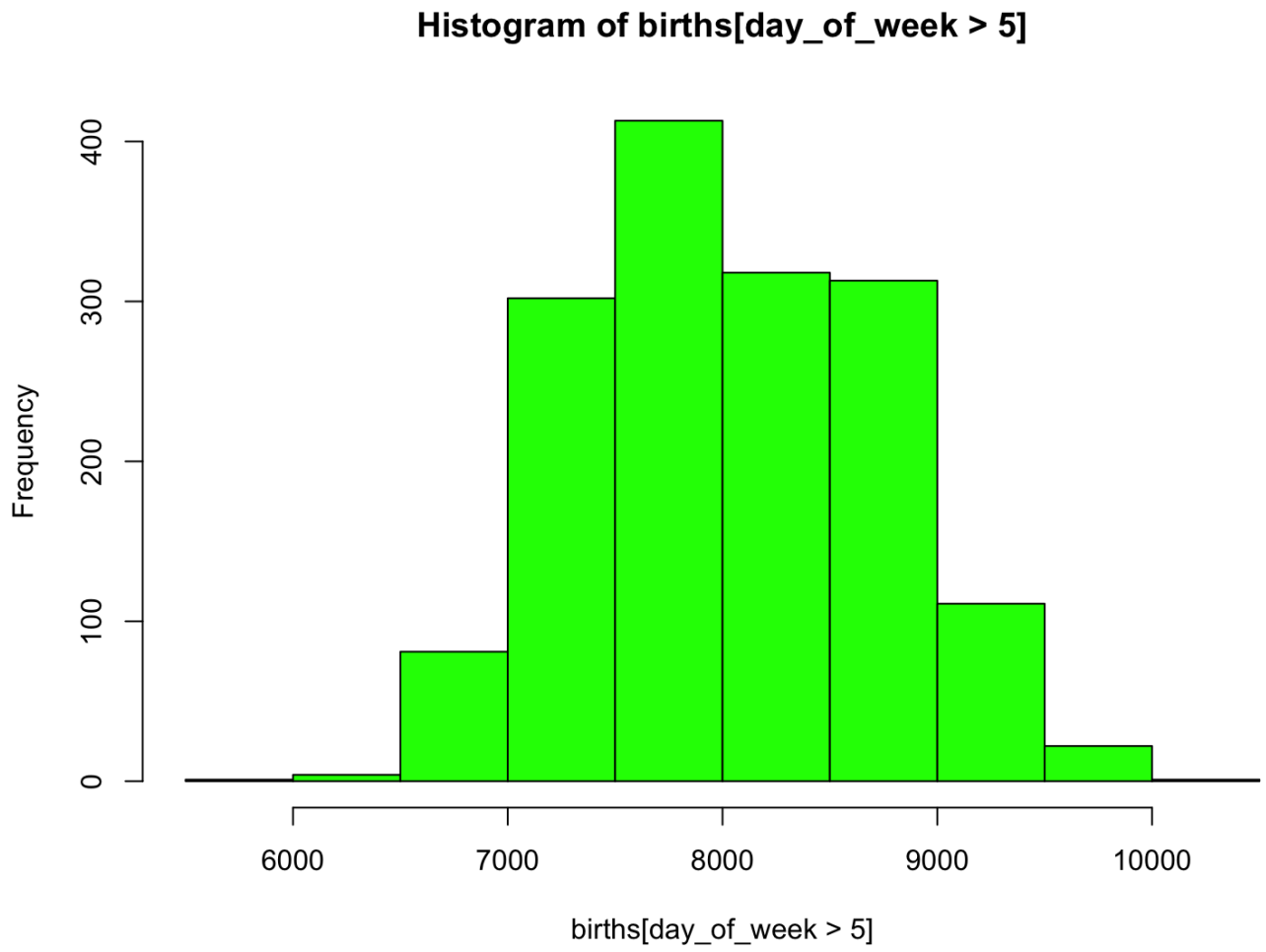
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    5728    8740   12340   11350   13080   16080
```

Histogram

```
hist(birthn$births)
```



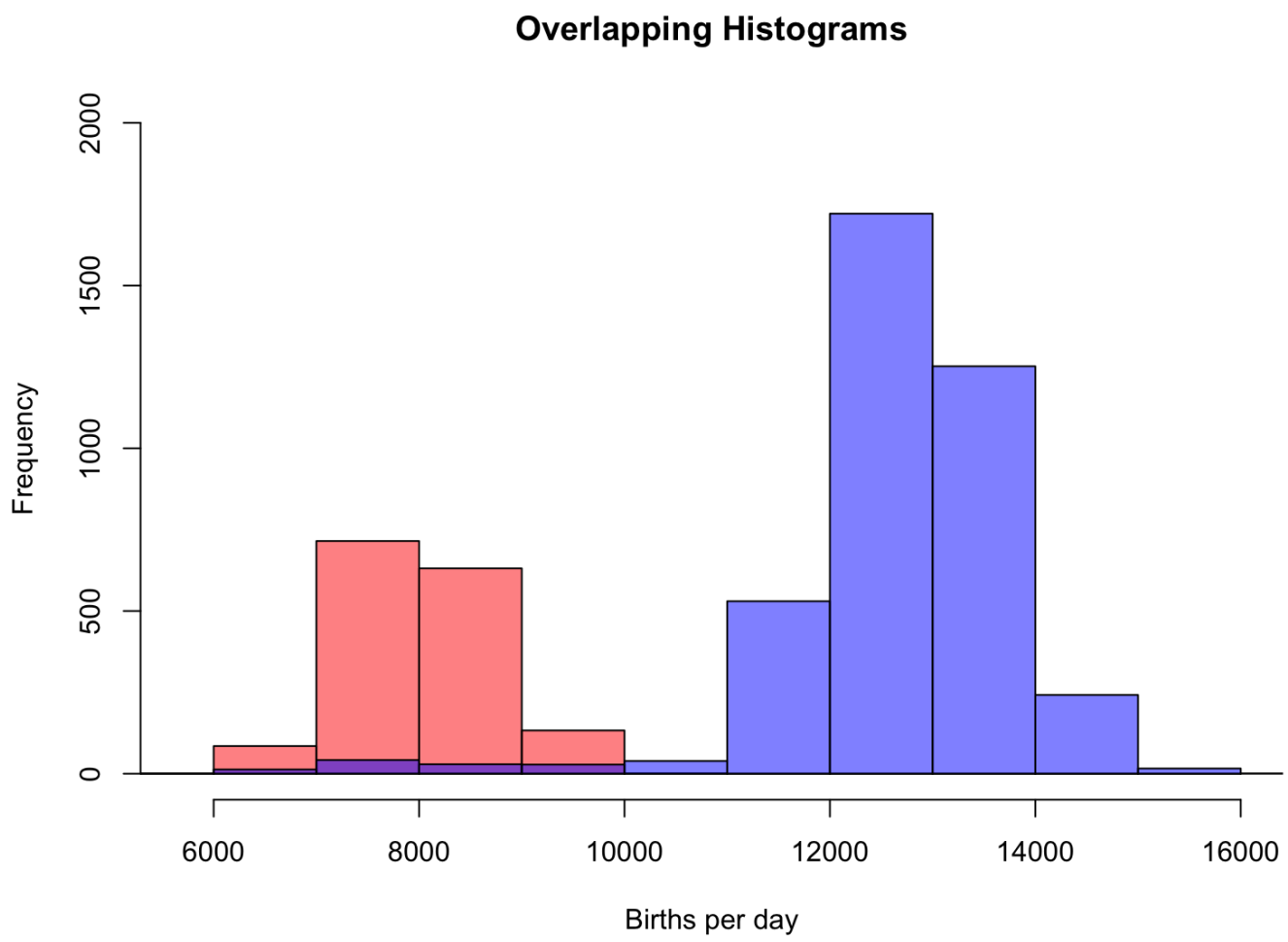

```
hist(birthn$births[day_of_week>5],col ="green")
```




```
summary(births)
```

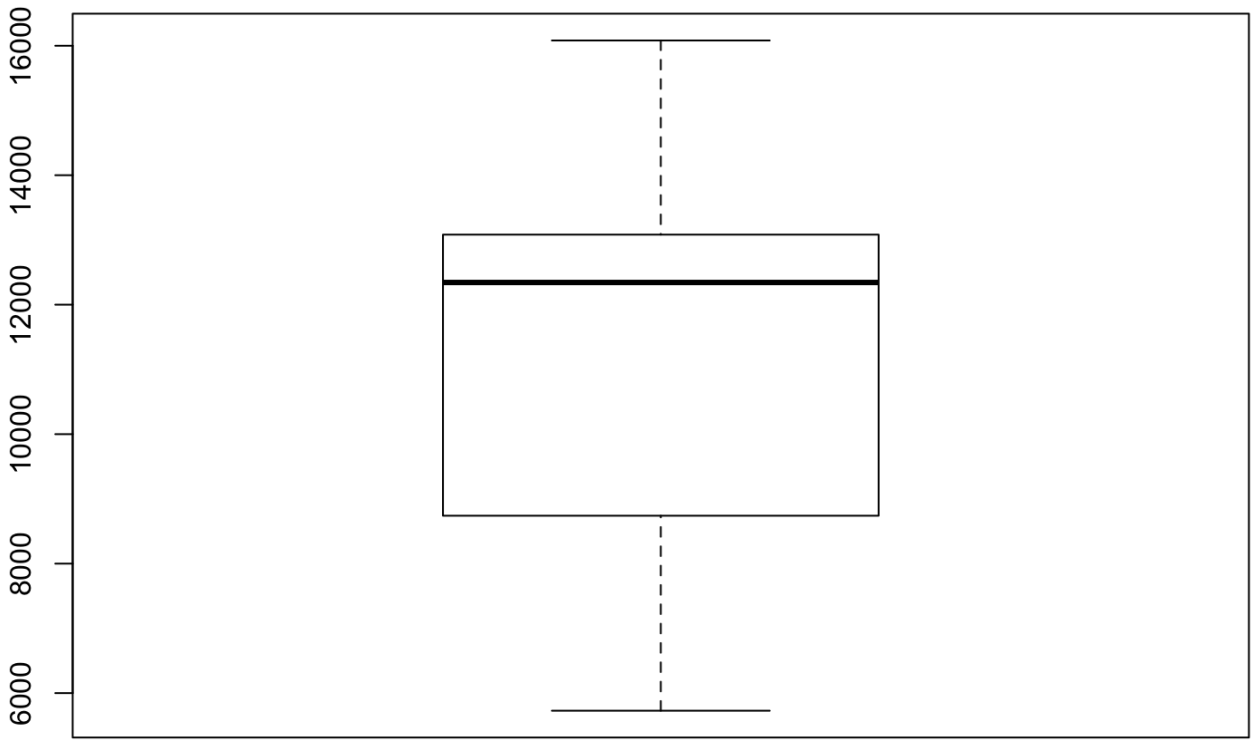
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    5728    8740   12340   11350   13080   16080
```

```
hist(birthn$births[day_of_week>5],breaks=seq(5000,16000,by=1000),
col=rgb(1,0,0,0.5),xlim=c(5700,16000))
```

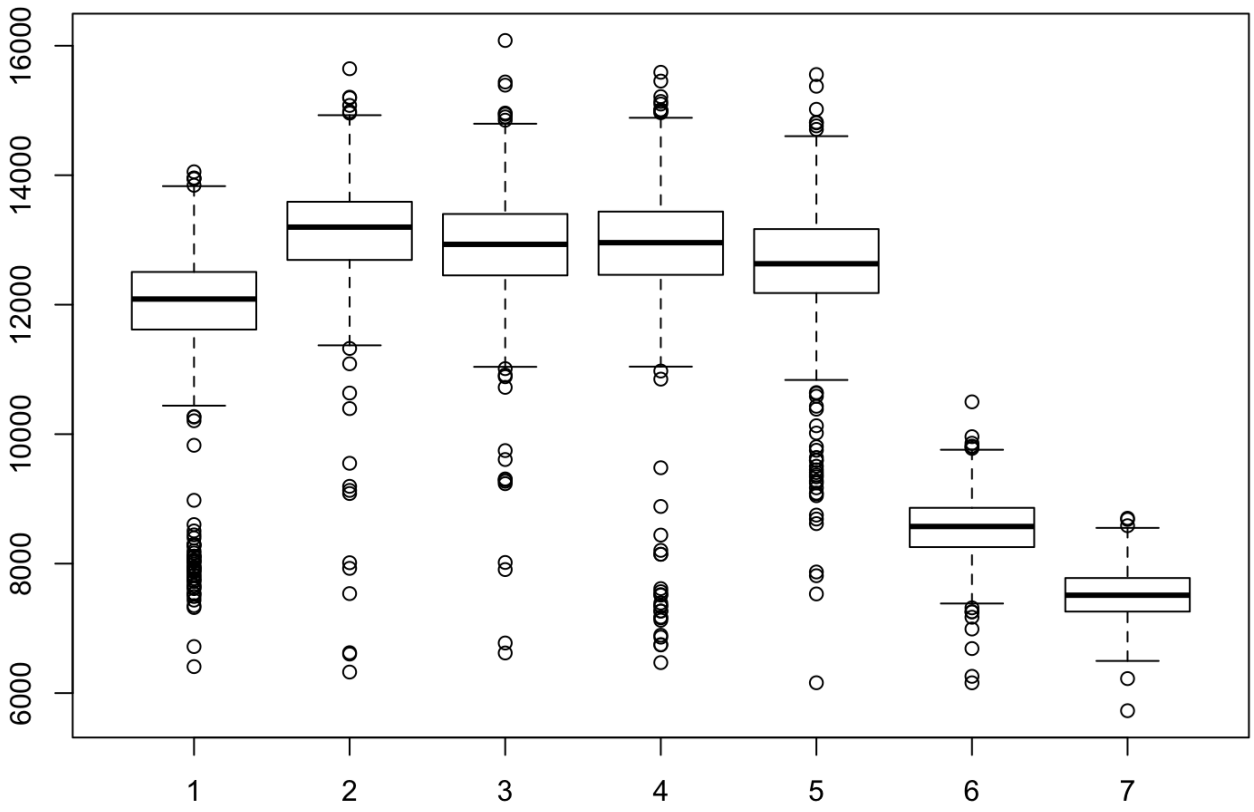


Boxplot

```
boxplot(birthn$births)
```




```
boxplot(birthn$births~day_of_week)
```



Notice the ~ sign this is useful for plotting with a formula.

Barplot

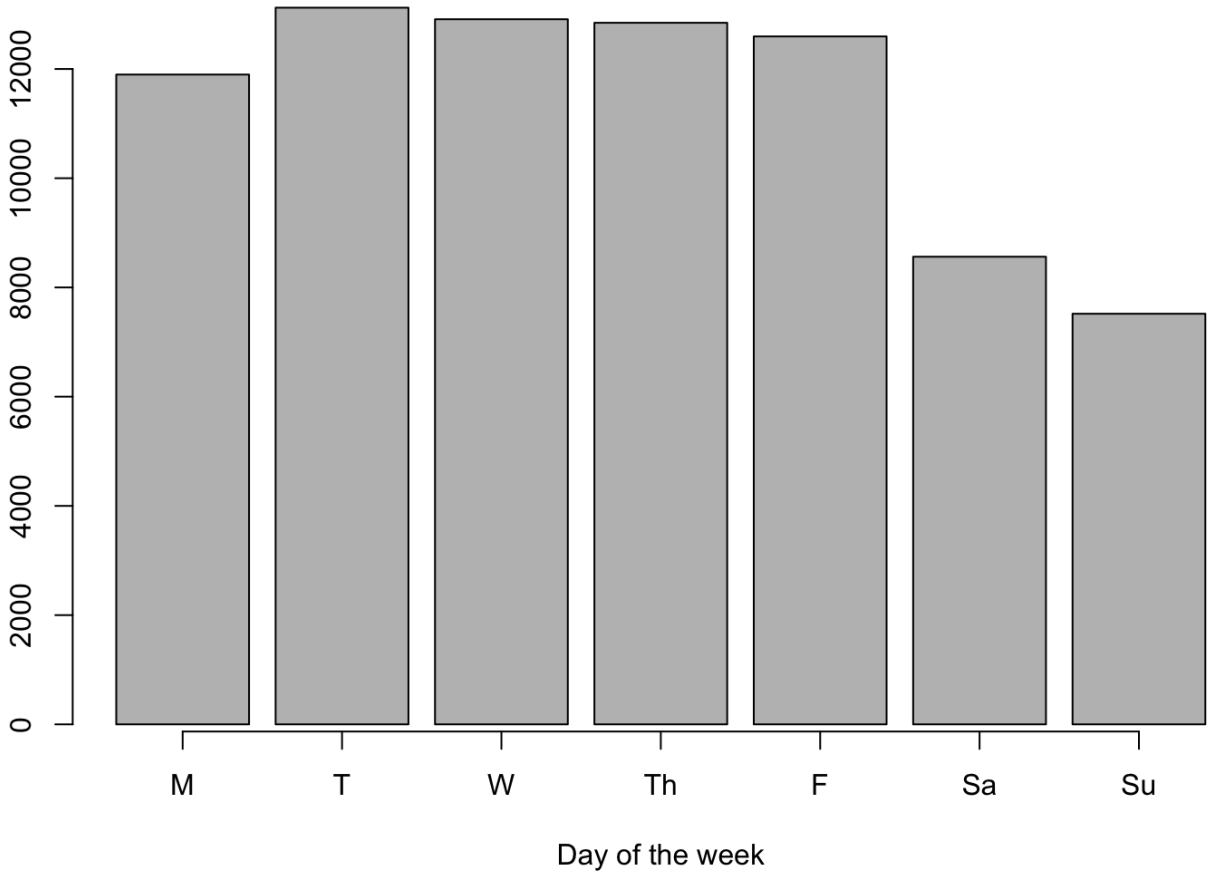
```
meansperday<- birthn %>%
 group_by(day_of_week) %>%
 summarise(ave=mean(births)) %>%
 arrange()
vectormeans <- meansperday%>%.$ave
```

```
barplot(vectormeans,xlab="Day of the week")
bp<- barplot(vectormeans,xlab="Day of the week")
bp
```

```
##      [,1]
## [1,]  0.7
```

```
## [2,]  1.9
## [3,]  3.1
## [4,]  4.3
## [5,]  5.5
## [6,]  6.7
## [7,]  7.9
```

```
axis(at=bp,labels=c("M","T","W","Th","F","Sa","Su"),side=1)
```



Summary of this Session:

- The `plot` function understands many different types of objects and plots accordingly.
- Simple scatterplots are achieved by having two vectors of the same length and eventually a plotting character `pch`.
- Other internal arguments such as color and line type can be added inside the function.
- Text can be added to the plot with `text()`.
- Many extra elements may be added through extra functions like `axis()`, `legend()`, `lines()`...
- Special plots for frequency or distribution visualization include histograms, boxplots and barplots.

Base R These plots are usually simple and not of publication quality. Better plots can be achieved using the `ggplot2` package which makes the plots by layers of graphical components.