R – Trend Analysis and Plotting

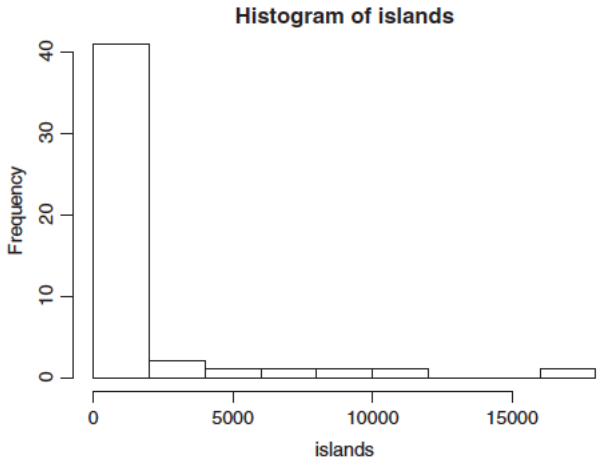
Lecture 5

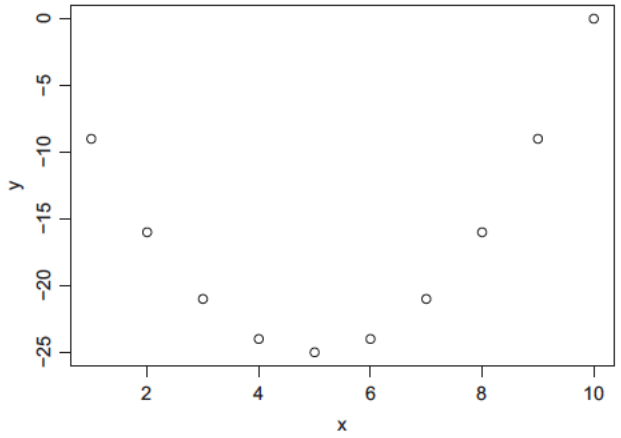
- Customize Plots
- Graphics functions.

Plotting

Two basic plots are the histogram and the scatterplot. The codes below were used to produce the graphs that appear in Figures 2.1 and 2.2:

```
hist(islands)
x <- seq(1, 10)
y <- x^2 - 10 * x
plot(x, y)
```



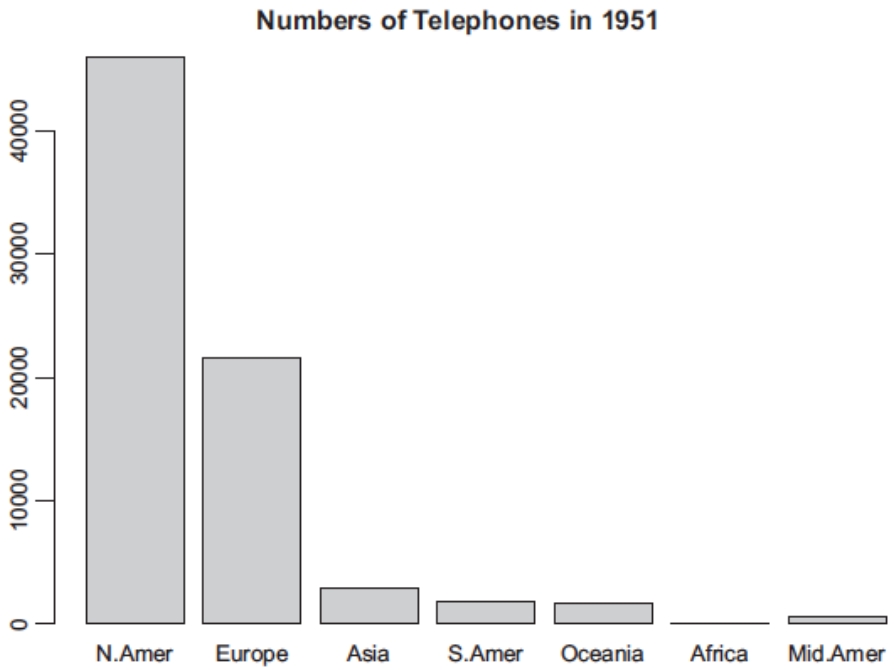


Source: A first course in statistical programming with R - W. John Braun and Duncan J. Murdoch, Cambridge, 2016.

Plotting

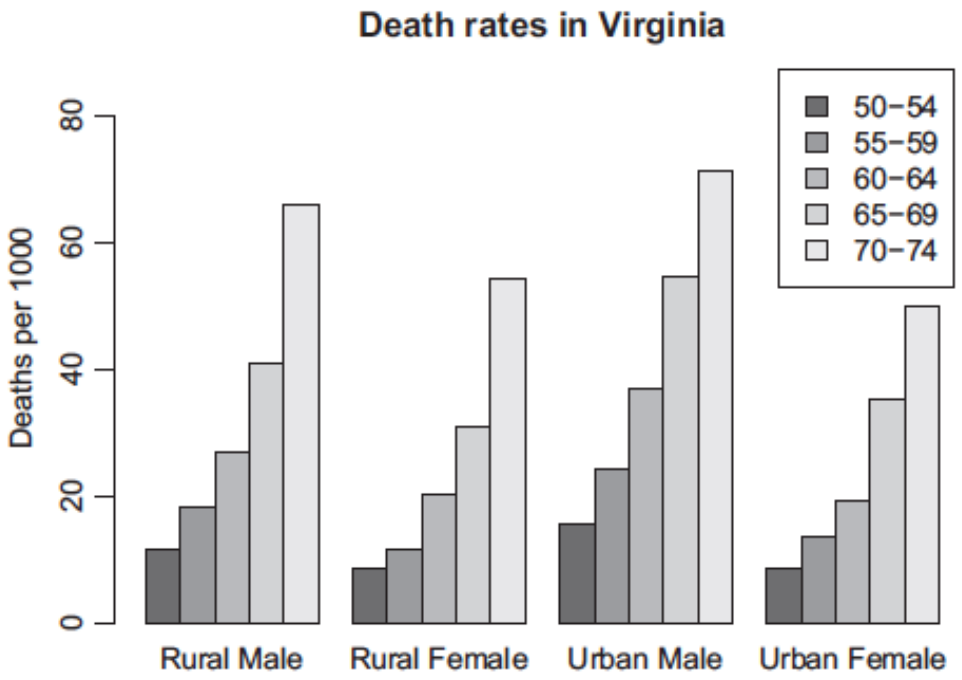
We could plot the bar chart using the `barplot()` function as

```
barplot(WorldPhones51)
```



Source: A first course in statistical programming with R - W. John Braun and Duncan J. Murdoch, Cambridge, 2016.

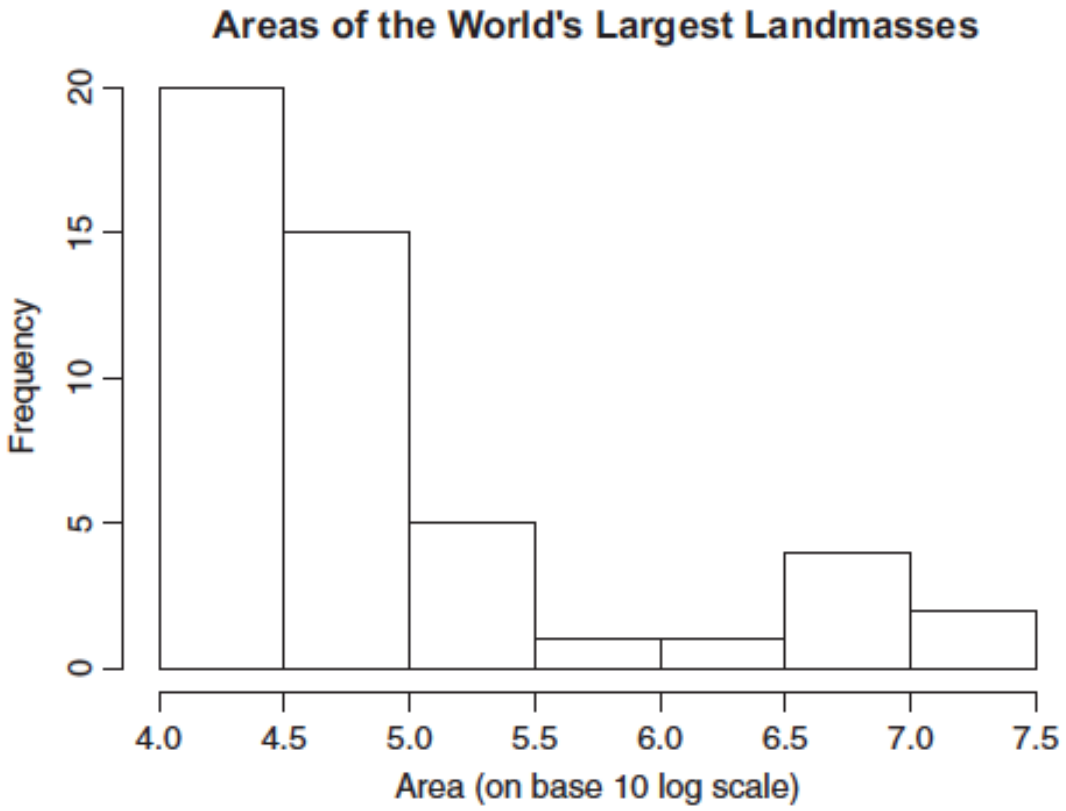
Plotting




```
barplot(VADeaths, beside = TRUE, legend = TRUE, ylim = c(0, 90),
        ylab = "Deaths per 1000",
        main = "Death rates in Virginia")
```


Source: A first course in statistical programming with R - W. John Braun and Duncan J. Murdoch, Cambridge, 2016.

Histrograms




```
hist(log(1000*islands, 10),  xlab = "Area (on base 10 log scale)",
    main = "Areas of the World's Largest Landmasses")
```


Source: A first course in statistical programming with R - W. John Braun and Duncan J. Murdoch, Cambridge, 2016.

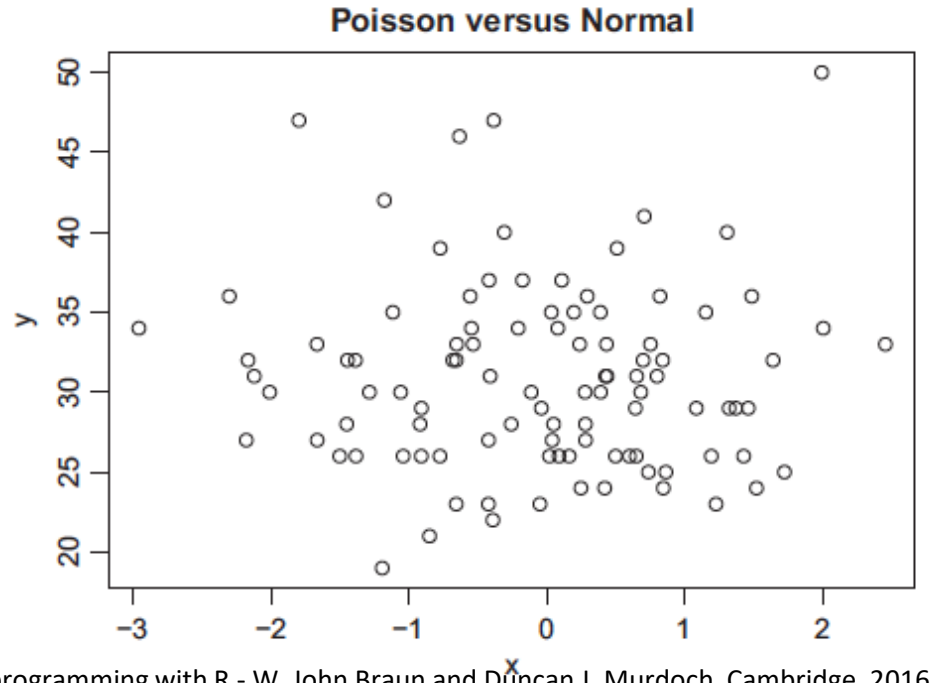
Scatter Plot

```
x <- rnorm(100)        # assigns 100 random normal observations to x
y <- rpois(100, 30)    # assigns 100 random Poisson observations
                       # to y; mean value is 30
mean(y)                # the resulting value should be near 30

## [1] 30.91
```

The `main` argument sets the main title for the plot. Figure 3.10 shows the result of

```
plot(x, y, main = "Poisson versus Normal")
```



Source: A first course in statistical programming with R - W. John Braun and Duncan J. Murdoch, Cambridge, 2016.

Advance Plotting

3.3 | Low level graphics functions

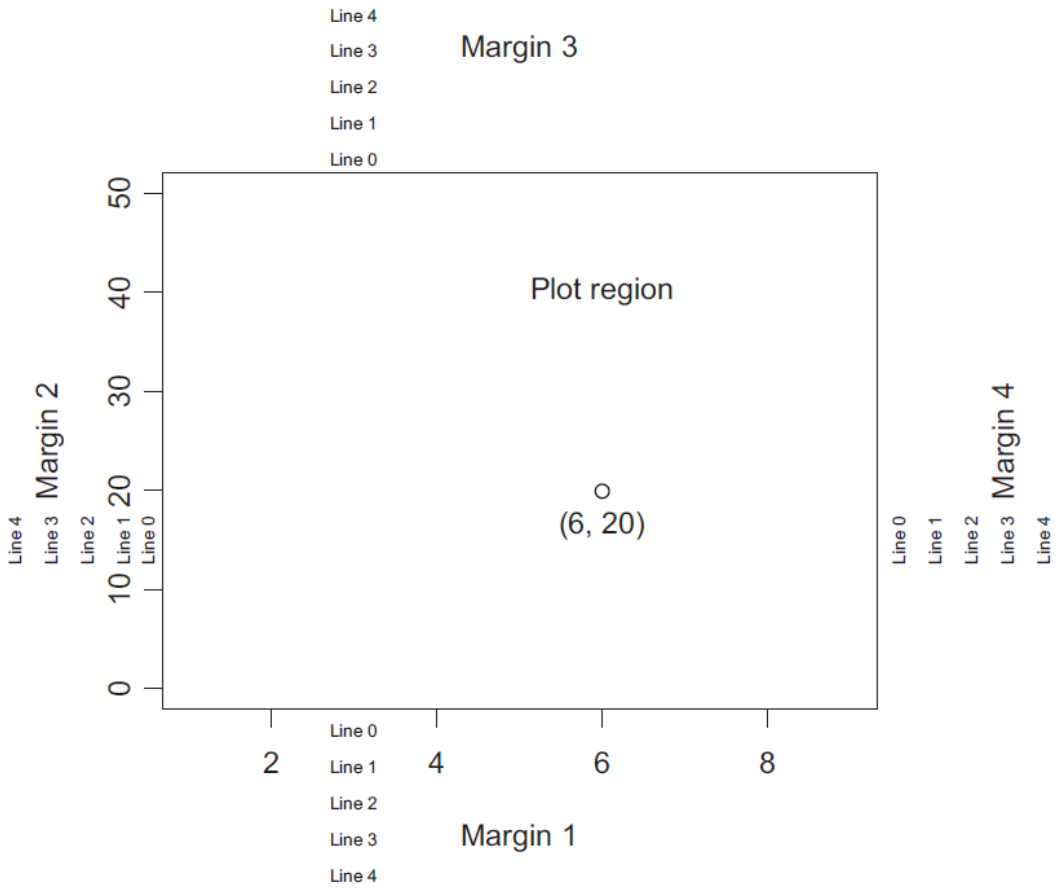
Functions like `barplot()`, `dotchart()`, and `plot()` do their work by using low level graphics functions to draw lines and points, to establish where they will be placed on a page, and so on.

In this section we will describe some of these low level functions, which are also available to users to customize their plots. We will start with a description of how R views the page it is drawing on, then show how to add points, lines, and text to existing plots, and finish by showing how some of the common graphics settings are changed.

Source: A first course in statistical programming with R - W. John Braun and Duncan J. Murdoch, Cambridge, 2016.

Margins and Plot Area

Base graphics in R divides up the display into several regions. The plot region is where data will be drawn. Within the plot region R maintains a coordinate system based on the data. The axes show this coordinate system. Outside the plot region are the margins, numbered clockwise from 1 to 4, starting at the bottom. Normally text and labels are plotted in the margins, and R positions objects based on a count of lines out from the plot region.



Source: A first course in statistical programming with R - W. John Braun and Duncan J. Murdoch, Cambridge, 2016.

Adding Details to Plot

3.3.2 Adding to plots

Several functions exist to add components to the plot region of existing graphs:

```
points(x, y, ...)                # adds points
lines(x, y, ...)                 # adds line segments
text(x, y, labels, ...)          # adds text into the graph
abline(a, b, ...)                # adds the line $y = a + bx$
abline(h = y, ...)               # adds a horizontal line
abline(v = x, ...)               # adds a vertical line
polygon(x, y, ...)               # adds a closed and possibly filled polygon
segments(x0, y0, x1, y1, ...)    # draws line segments
arrows(x0, y0, x1, y1, ...)      # draws arrows
symbols(x, y, ...)               # draws circles, squares, thermometers, etc.
legend(x, y, legend, ...)        # draws a legend
```


Source: A first course in statistical programming with R - W. John Braun and Duncan J. Murdoch, Cambridge, 2016.

Advance Plotting

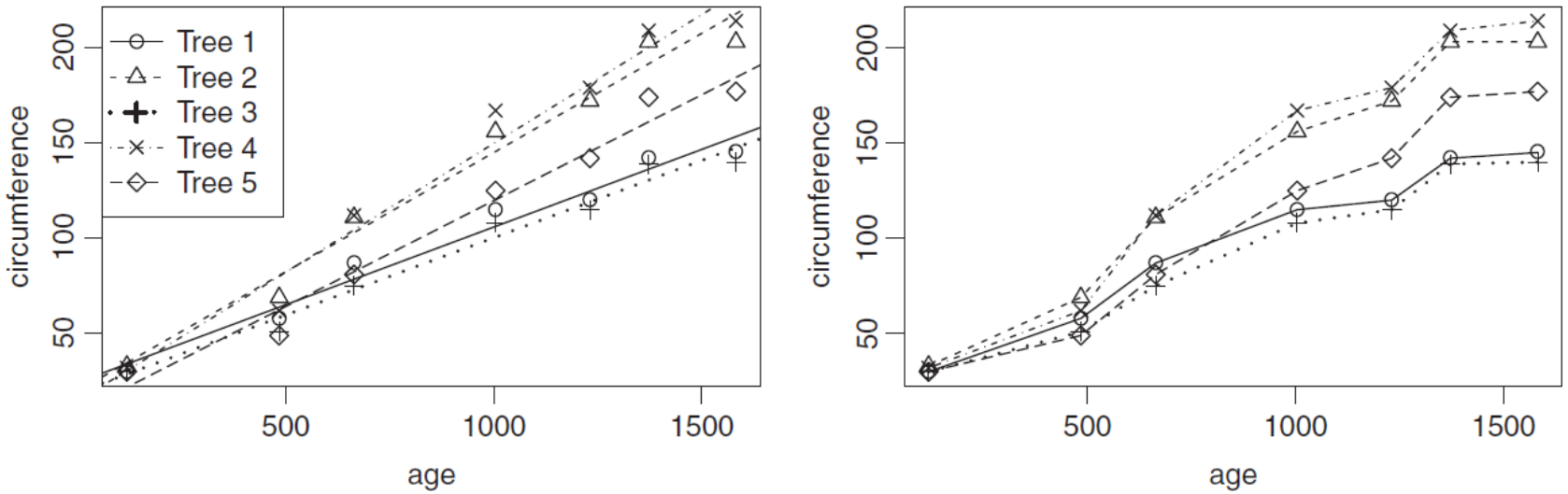
Example 3.7

Consider the `Orange` data frame again. In addition to using different plotting characters for the different trees, we will pass lines of best fit (i.e. least-squares regression lines) through the points corresponding to each tree.

The basic scatterplot is obtained from

```
plot(circumference ~ age, pch = as.numeric(as.character(Tree)),
     data = Orange)
```

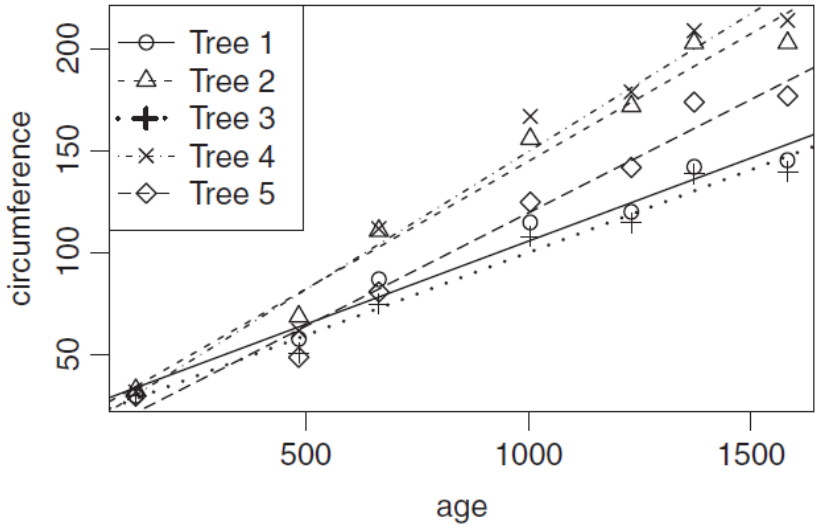
The best-fit lines for the five trees can be obtained using the `lm()` function which relates circumference to age for each tree. A legend has been added to identify which data points come from the different trees:

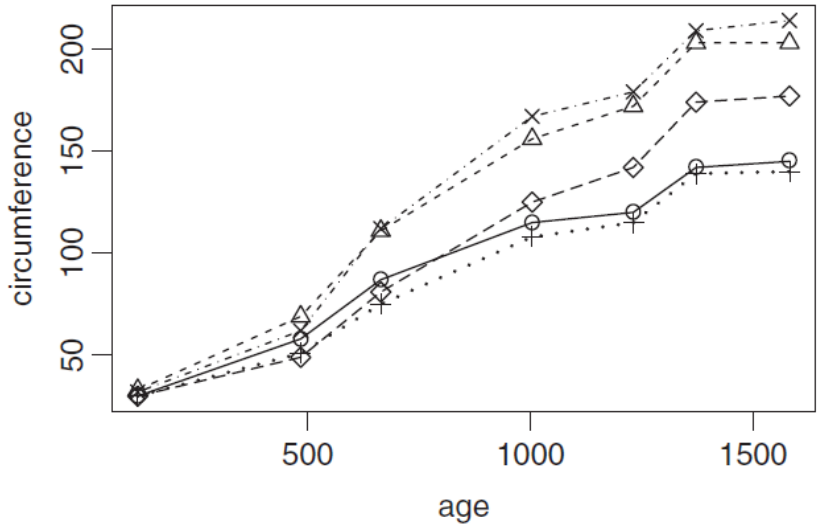


Source: A first course in statistical programming with R - W. John Braun and Duncan J. Murdoch, Cambridge, 2016.

Advance Plotting

```
abline(lm(circumference ~ age, data = Orange, subset = Tree == "1"),
       lty = 1)
abline(lm(circumference ~ age, data = Orange, subset = Tree == "2"),
       lty = 2)
abline(lm(circumference ~ age, data = Orange, subset = Tree == "3"),
       lty = 3, lwd = 2)
abline(lm(circumference ~ age, data = Orange, subset = Tree == "4"),
       lty = 4)
abline(lm(circumference ~ age, data = Orange, subset = Tree == "5"),
       lty = 5)
legend("topleft", legend = paste("Tree", 1:5), lty = 1:5, pch = 1:5,
       lwd = c(1, 1, 2, 1, 1))
```





Source: A first course in statistical programming with R - W. John Braun and Duncan J. Murdoch, Cambridge, 2016.

Adding Details to Plot

One may also wish to annotate graphs outside the plot region. Several functions exist to do this:

```
title(main, sub, xlab, ylab, ...) # adds a main title, a subtitle,
                                  # an x-axis label and/or a y-axis label
mtext(text, side, line, ...)      # draws text in the margins
axis(side, at, labels, ...)       # adds an axis to the plot
box(...)                          # adds a box around the plot region
```

Example 3.8
Figure 3.14 was drawn using the following code:

```
par(mar = c(5, 5, 5, 5) + 0.1)
plot(c(1, 9), c(0, 50), type = 'n', xlab = "", ylab = "")
```


Source: A first course in statistical programming with R - W. John Braun and Duncan J. Murdoch, Cambridge, 2016.

Adding Details to Plot

```
text(6, 40, "Plot region")
points(6, 20)
text(6, 20, "(6, 20)", adj = c(0.5, 2))
mtext(paste("Margin", 1:4), side = 1:4, line = 3)
mtext(paste("Line", 0:4), side = 1, line = 0:4, at = 3,  cex = 0.6)
mtext(paste("Line", 0:4), side = 2, line = 0:4, at = 15, cex = 0.6)
mtext(paste("Line", 0:4), side = 3, line = 0:4, at = 3,  cex = 0.6)
mtext(paste("Line", 0:4), side = 4, line = 0:4, at = 15, cex = 0.6)
```

Understanding the code

The `mar` parameter sets the numbers of lines of margin on sides 2, 3, and 4, so that they are all the same width as the bottom margin (side 1). The `type = 'n'` plot is empty but specifies the scope of the plotting region, essentially from 1 through 9 on the horizontal axis and from 0 through 50 on the vertical axis. The `text()` and `mtext()` functions tell R where to place the given text, such as "Plot region," and so on. Using `paste("Line", 0:4)` avoids typing `c("Line 0", "Line 1", "Line 2", "Line 3", "Line 4")`.

Source: A first course in statistical programming with R - W. John Braun and Duncan J. Murdoch, Cambridge, 2016.

Adding Details to Plot

3.3.3 Adjusting axis tick labels

In Figure 3.6, a histogram of the `islands` data set was plotted on the base-10 log scale. Although the axis tick labels are displayed accurately, most individuals who look at such a histogram would need time to process this information in order to answer a simple question such as "How many of the landmasses have an area exceeding 100,000 square miles?"

In Figure 3.16, the axes have been removed initially, using the `axes = FALSE` argument. This removes the box as well, so it is replaced using the `box()` function. The horizontal axis is then re-drawn with labels at 4, 5, 6, and 7, but these correspond to 10^4, 10^5, 10^6, and 10^7 on the square mile scale. We have explicitly written out the character strings in the `labels` argument, because scientific notation would have been used with the quicker option `labels = 10^(4:7)`.[2]

[2] The `format()` function has options to do this automatically.

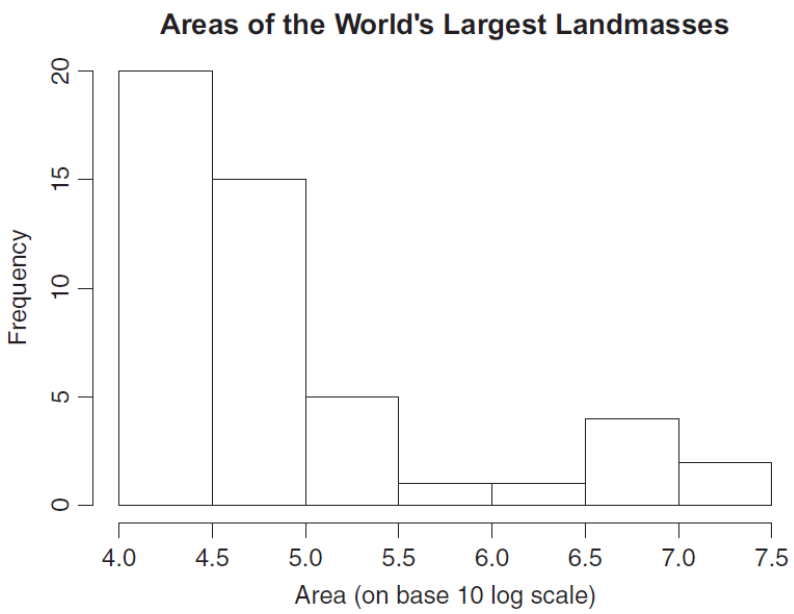
```
hist(log(1000*islands, 10),  axes = FALSE, xlab = "Area (in sq. miles)",
     main = "Areas of the World's Largest Islands")
box()
axis(side = 1, at = 4:7, labels = c("10,000", "100,000", "1,000,000",
     "10,000,000"))
axis(side = 2)
```

Incidentally, it is a quick exercise now to see that exactly 13 of the landmasses exceed 100,000 square miles in area.

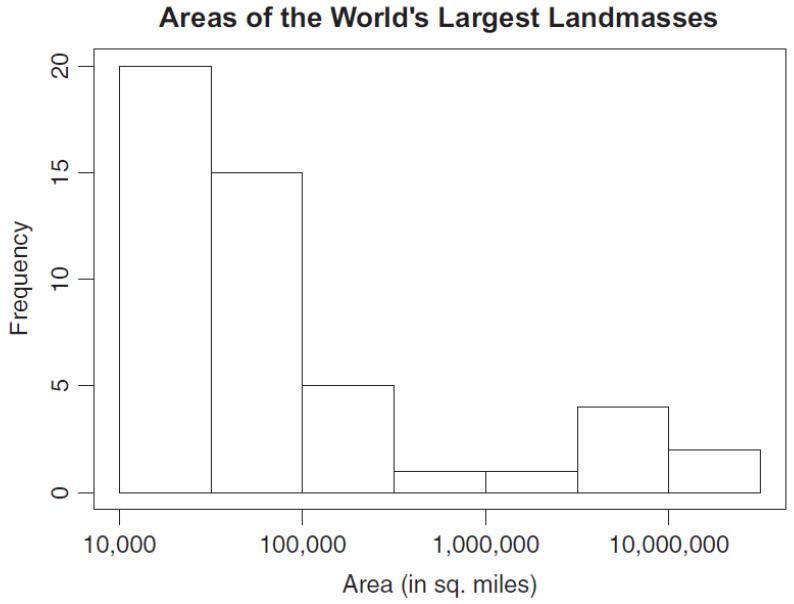
Source: A first course in statistical programming with R - W. John Braun and Duncan J. Murdoch, Cambridge, 2016.

Advance Plotting

Original



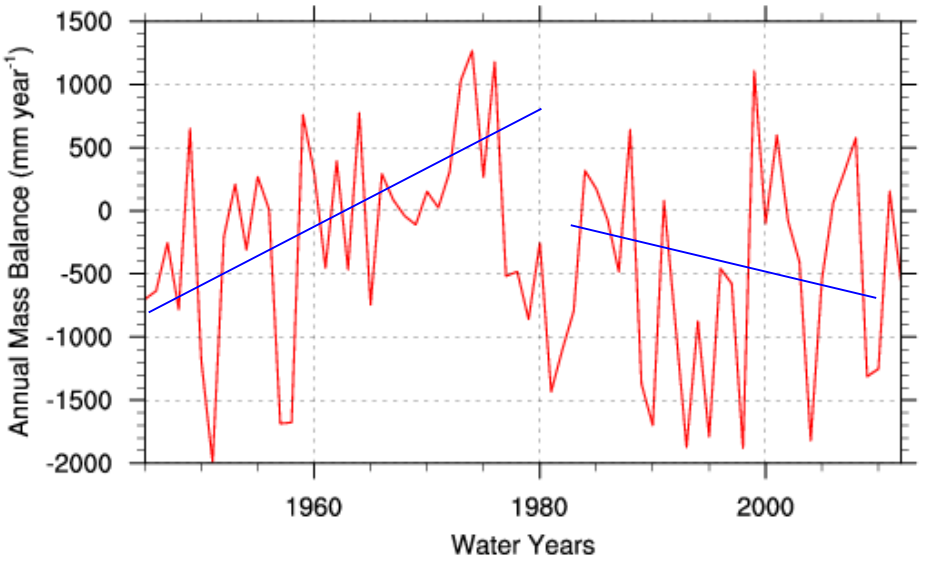
Modified



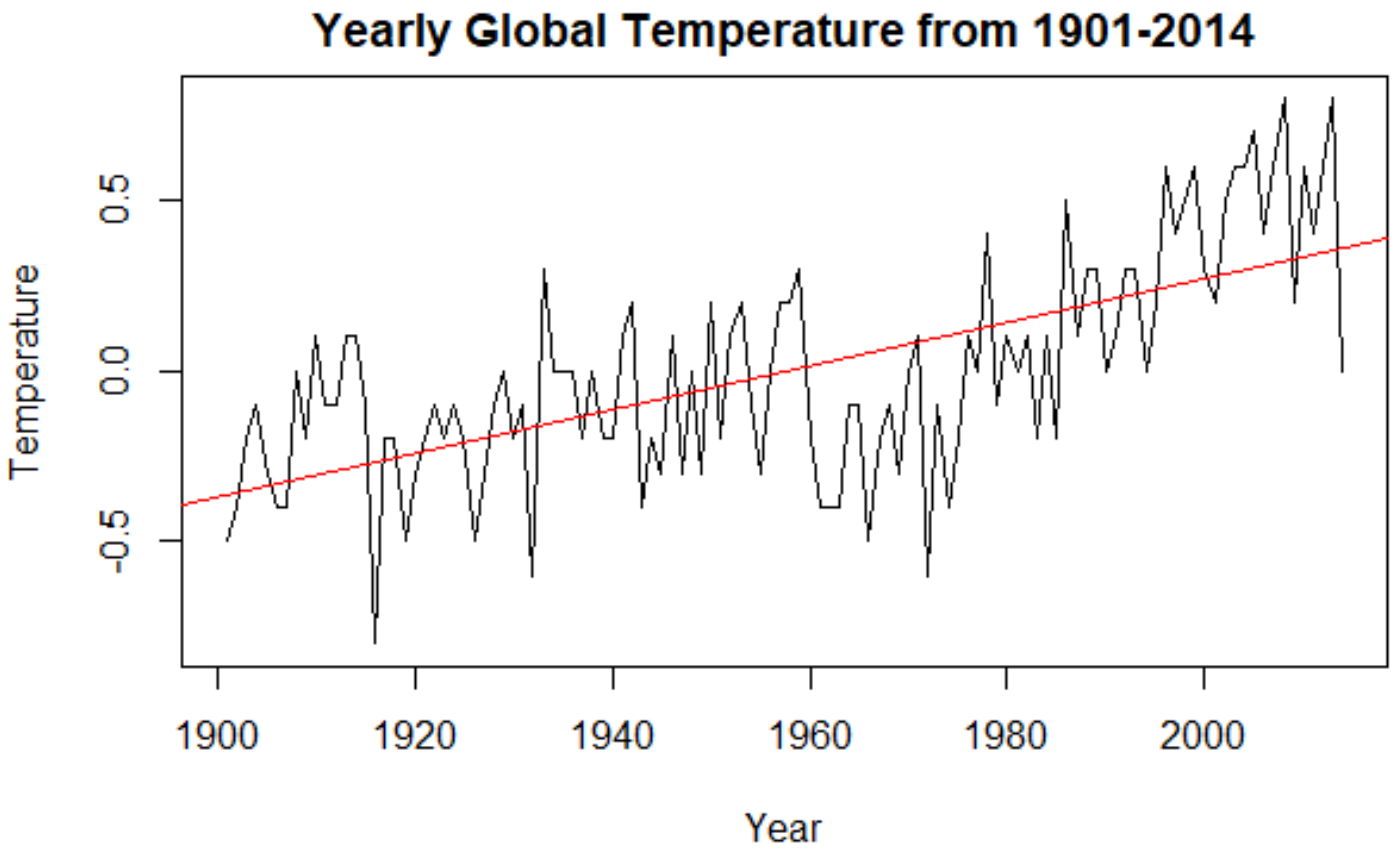
Source: A first course in statistical programming with R - W. John Braun and Duncan J. Murdoch, Cambridge, 2016.

Linear Trend Analysis

A linear trend analysis is often conducted to examine whether a particular data set (say, the global mean temperature) has increased or decreased over a period of time (i.e., the global warming). A trend line could simply be drawn by eye through a set of data points, but more properly their position and slope is calculated using statistical techniques like linear regression.



Trend Analysis

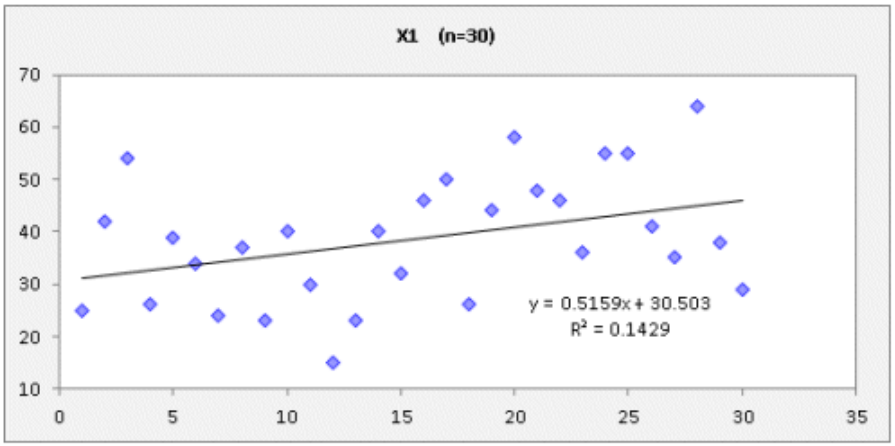


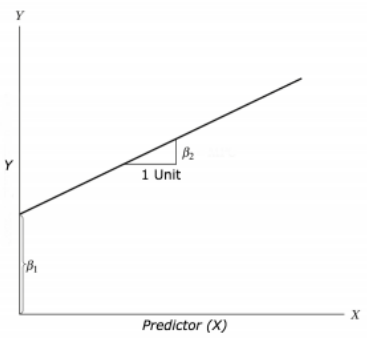
Linear Regression

Linear regression is one of the most commonly used predictive modelling techniques. The aim of linear regression is to find a mathematical equation for a continuous response variable Y as a function of one or more X variable(s). So that you can use this regression model to predict the Y when only the X is known.

$$y = mx + b + \epsilon$$

where, b is the intercept and m is the slope. Collectively, they are called regression coefficients and ϵ is the error term, the part of Y the regression model is unable to explain.





A valuable numerical measure of association between two variables is the correlation coefficient, which is a value between -1 and 1 indicating the strength of the association of the observed data for the two variables.

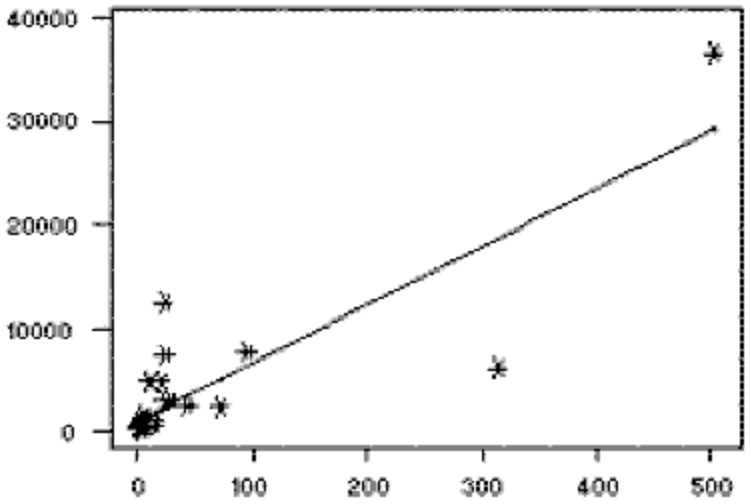
Source: https://www.machinelearningplus.com/machine-learning

Least-Squares Regression

- The most common method for fitting a regression line is the method of least-squares.
- This method calculates the best-fitting line for the observed data by minimizing the sum of the squares of the vertical deviations from each data point to the line (if a point lies on the fitted line exactly, then its vertical deviation is 0).
- Because the deviations are first squared, then summed, there are no cancellations between positive and negative values.

Least-Squares Regression Example

The correlation coefficient is 0.852 and the r^2 value is 0.726 (the square of the correlation coefficient), indicating that 72.6% of the variation in one variable may be explained by the other.



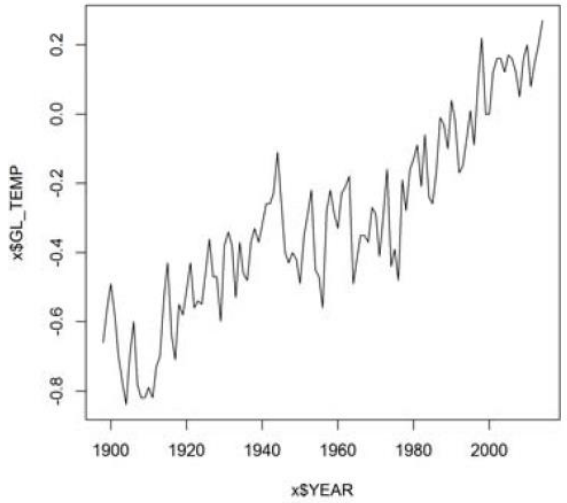
Source: http://www.stat.yale.edu

Trend Analysis

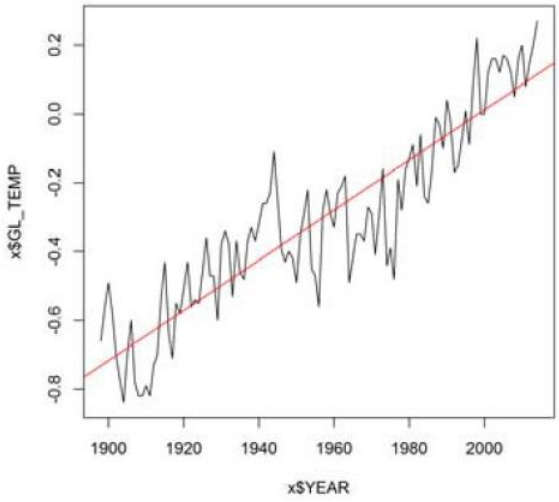
Lets estimate the trend used by the least square fit in R.

```
> plot(x$YEAR, x$GL_TEMP, type=”l”)

# Y = a0 + a1 * X
# x$YEAR as X. x$GL_TEMP as Y.
# lm is a function for least square fit
> res <- lm(x$GL_TEMP ~ x$YEAR)
# show summary of results.
> summary(res)
```




```
> slope <- res$coefficients[2]
> intercept <- res$coefficients[1]
> plot(x$YEAR, x$GL_TEMP, type="l")
> abline(res, col="red")
```



The Trend in the global surface temperature anomaly is ??°C/year i.e. (??°C/100year).

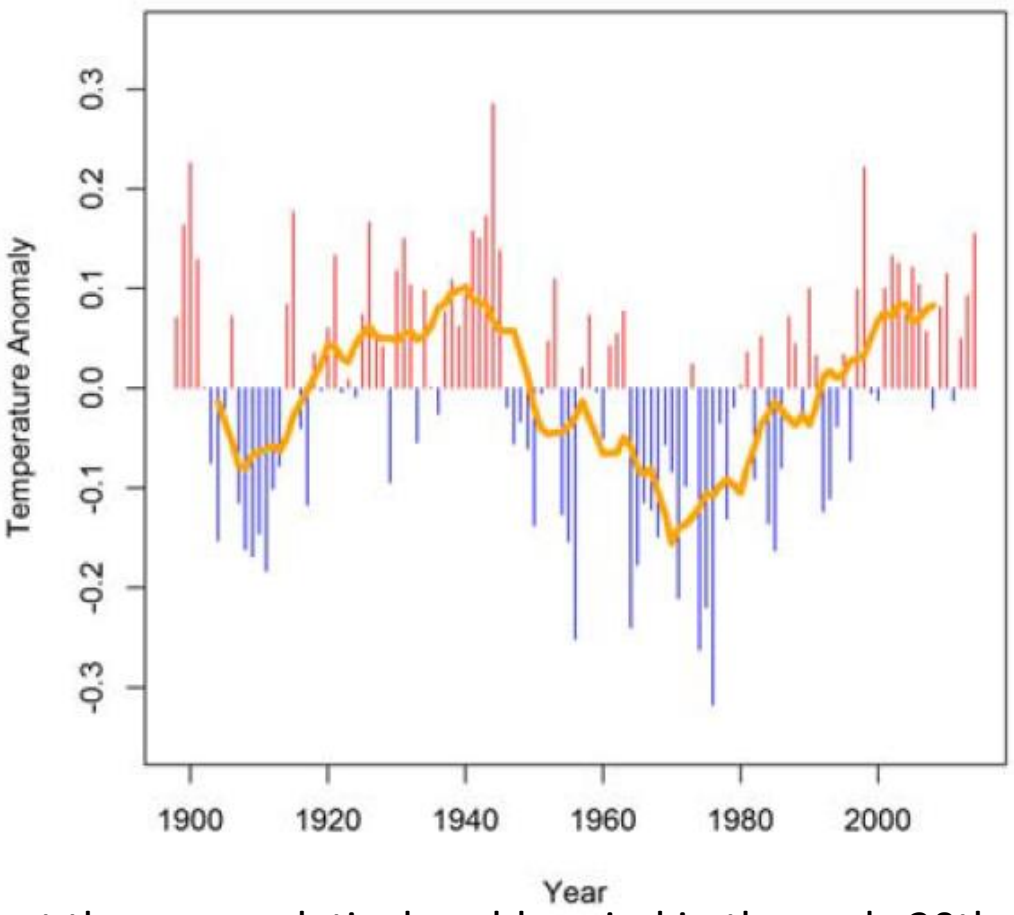
Source: Nobuhiko ENDO, Japan Agency for Marine-Earth Science and Technology

Quality Plotting in R

```
# plot the residual anomaly. Do not draw axes, labels.
# positive (negative) anomaly is drawn in red (blue)
# set limit range for y-axis from -0.35 to 0.35
# type = “h”means vertical bar plot
> plot( x$YEAR, res$residual, type = "h", ylim = c(-0.35,0.35), axes = FALSE, xlab = "",
ylab = "", col = ifelse(res$residual > 0.0, "red", "blue"))
# to overlay second plot, use “par()” command.
> par( new = TRUE )
# plot running average time series. “lwd”is thickness of line.
> plot( t.runave$year1, t.runave$t.runave, type = "l", ylim = c(-0.35,0.35), lwd = 4,
col = "orange", xlab = "Year", ylab = "Temperature Anomaly")
```


Source: Nobuhiko ENDO, Japan Agency for Marine-Earth Science and Technology

Averaged Temperature



It is obvious that there are relatively cold period in the early 20th century and from mid-1940's to mid-1990's and relatively warm period from 1920's to mid-1940's and recent decades

Source: Nobuhiko ENDO, Japan Agency for Marine-Earth Science and Technology

How to Save Plot and Data Frame

We can save a graph as “PNG”, and “PDF”.

```
# to save a graph to PNG file. First set resolution in ppi unit.
> ppi <- 300
# width and height are in inch.
> png("fig_1.png", width = 6 * ppi, height = 3 * ppi, res = ppi)
> plot(nvec,x,type="l")
> dev.off()
# to save a PDF file.
> pdf("fig_1.pdf")
> plot(nvec,x,type="l")
> dev.off()
```