Graphics II

Example

This example show how to customise the limits on the *y* axis and the labels on the axes.

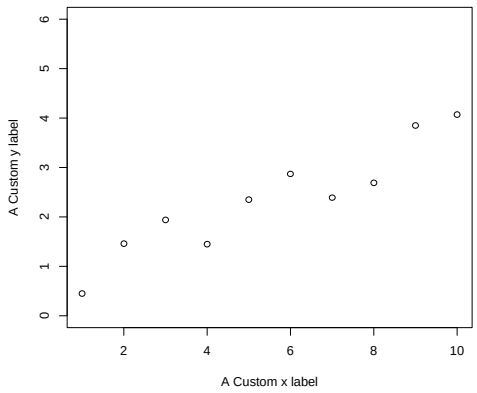
```
> x <- 1:10
> y <- c(0.45, 1.46, 1.94, 1.45, 2.35,
+       2.87, 2.39, 2.69, 3.85, 4.07)

> plot(x, y, ylim = c(0, 6),
+      xlab = "A Custom x label",
+      ylab = "A Custom y label")
```

The plot Function

The core S graphics command is **`plot`**. This is an all-in-one function which carries out a number of actions.

- It causes the graphics system to advance to a new "frame."
- It sets up a new plotting area using the current settings for margin size, plot type etc.
- It sets up the transformations which determine how data units are mapped to the plotting area.
- It plots the content of the graph (points, lines etc.).
- It plots *x* and *y* axes and box around the plot and produces the axis labels and title.



Controlling plot

There are a number of optional arguments to plot which provide control over its actions.

Separating Functionality

The **`plot`** function carries out many actions. It can be useful to separate out the actions which **`plot`** carries out. In particular, it is useful to have a function which prepares the way for plotting to take place, but does no actual plotting.

This can be done by simply cutting down what the plot function does. Here is is now this can be done.

```
> newplot <-
+   function(xlim, ylim, log="")
+   plot(NA, NA, xlim = xlim, ylim = ylim,
+        axes = FALSE, xlab = "", ylab = "")
```

`xlim=`, **`ylim=`** These arguments can be used to specify plot limits different from those computable from the data.

`xlab=`, **`ylab=`** These argument can be used to provide alternative labels for the *x* and *y* axes.

`log=` This argument can take the values **`""`**, **`"x"`**, **`"y"`** or **`"xy"`** and indicate which of the axes should have a logarithmic scale.

`axes=` This argument takes the values **`TRUE`** or **`FALSE`** and indicates whether or not the *x* and *y* axes should be plotted.

Assembling Plots From Components

Here is how the plot produced by the previous example can be produced using **`newplot`**.

```
> newplot(xlim = range(x), ylim = c(0, 6))
> points(x, y)
> axis(1)
> axis(2)
> box()
> title(xlab = "A Custom x label")
> title(ylab = "A Custom y label")
```

Understanding how to produce plots in this step-by-step way makes it very easy to produce custom displays.

Example

Consider how you would produce a plot of the function

$$f(x) = x(2x-3)(2x+3)$$

as you might find it in a a calculus book.

The requirements for such are graph are:

(i) Axes which cross at $(0,0)$ rather than lying at the sides of the plotting region.

(ii) Tickmarks chosen to avoid the path of the graph.

(iii) Tick labels which are aligned horizontally on each axis.

Circles

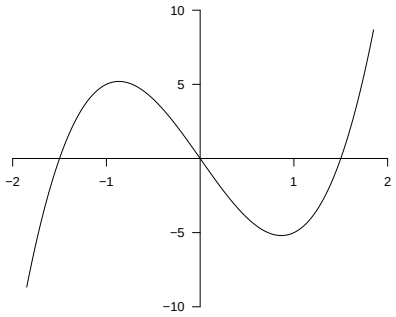
The equation of a circle with radius R centered at $(0,0)$ is

$$(R\cos\theta, R\sin\theta), \quad 0 \le \theta \le 2\pi.$$

This circle can be approximated by a polygon which has vertexes equally spaced around the circle.

By using enough vertexes it is possible to produce a polygon which is visually indistinguishable from a circle.

Any change of direction which is less that 5° is seen as a smooth change by the eye, so 72 equally-spaced vertexes will suffice.



A Cubic Polynomial

Drawing Circles with S

Here is some simple code to draw a coloured circle.

```
> par(pty="s")
> theta <- seq(0, 2*pi, length=73)[1:72]
> x <- cos(theta)
> y <- sin(theta)
> newplot(xlim=c(-1,1), ylim=c(-1,1))
> polygon(x, y, col="darkseagreen")
```

The statement

```
> par(pty="s")
```

is important. It ensures that the plot on the page is square and that it is a circle which is drawn rather than an ellipse.

S Code To Produce The Graph

We start by producing the the data for the curve. (1001 points may be overkill here.)

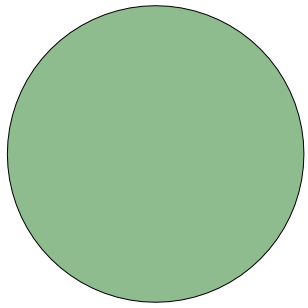
```
> x <- seq(-1.85, 1.85, length = 1001)
> y <- x * (2 * x - 3) * (2 *  x + 3)
```

Now we set up the plot and draw the actual curve.

```
> newplot(xlim=c(-2, 2), ylim = c(-10, 10))
> lines(x, y)
```

Finally, we draw the custom axes. The **pos** argument gives the position of the axis in data coordinates.

```
> axis(1, pos = 0, at = c(-2, -1, 1, 2))
> axis(2, pos = 0, at = c(-10, -5, 5, 10),
```



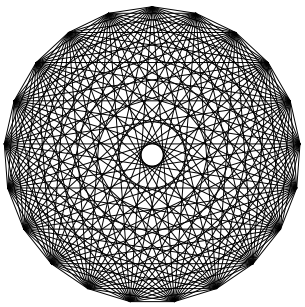
```
        las = 1)
> title(xlab = "A Cubic Polynomial")
```

Rosettes

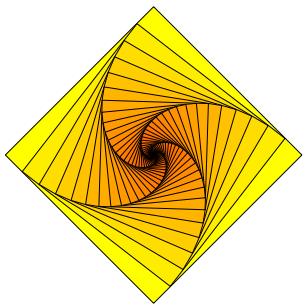
A rosette is a figure obtained by taking n points equally spaced around the circumference of circle, and joining every point to every other point.

```
> n <- 21
> theta <- seq(0, 2*pi, length=n+1)[1:n]
> x <- sin(theta)
> y <- cos(theta)
> par(pty="s")
> newplot(xlim = c(-1, 1), ylim = c(-1, 1))
> for(i in 2:n)
+   for(j in 1:(i-1))
+     segments(x[i], y[i], x[j], y[j])
> title(main="A 21 Rosette")
```

A 21 Rosette



A Spiral Square − Coloured



Spiral Squares

Here is how to create an interesting figure by rotating and shrinking a square multiple times.

```
> par(pty="s")
> newplot(xlim = c(-1, 1), ylim = c(-1, 1))
> square <- seq(0, 2*pi, length = 5)[1:4]
> n <- 51
> r <- 1.12 / cumprod(rep(1.12, n))
> theta <- seq(0, 2 * pi, length = n)
> for(i in 1:n) {
+   x <- r[i] * cos(theta[i] + square)
+   y <- r[i] * sin(theta[i] + square)
+   polygon(x, y)
+ }
```

Advanced Examples

If S graphics is regarded as providing low-level functionality rather than high-level packaged recipes, then there is almost no limit to what to can be done with it.

The following graphs show some of what I have produced in this way.

```
> title(main="A Spiral Square")
```

A “Fancy” Bar Chart

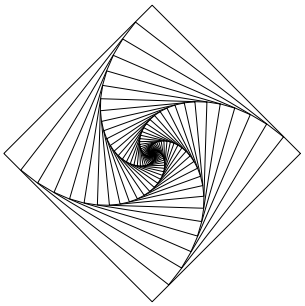
This graph is a reproduction from page 66 of Ed Tufte’s book *The Visual Display of Quantitative Information*.

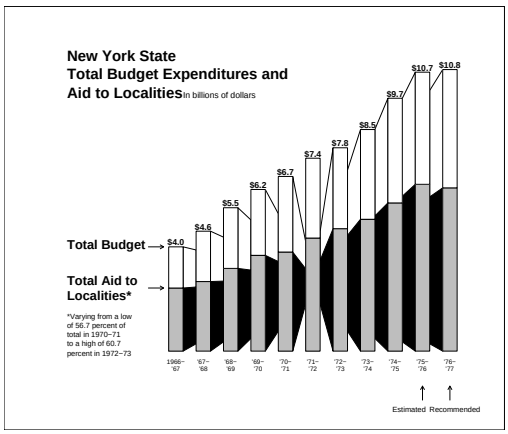
The graph shows a strong attempt to hide the essential message in the plot by introducing non-essential “chartjunk.”

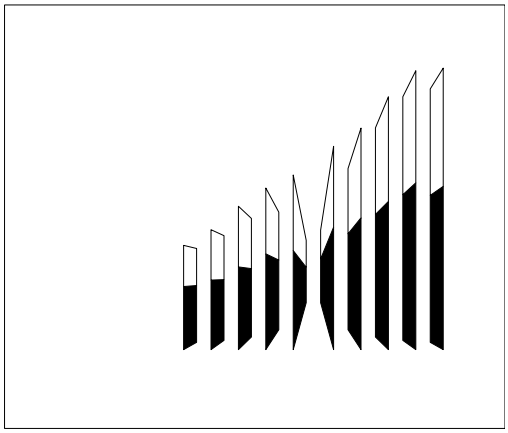
I reproduced this graph by measuring the lengths of the bars with a ruler and drew the bars on the screen to match my measurements.

I also tried to keep the fonts roughly comparable.

A Spiral Square





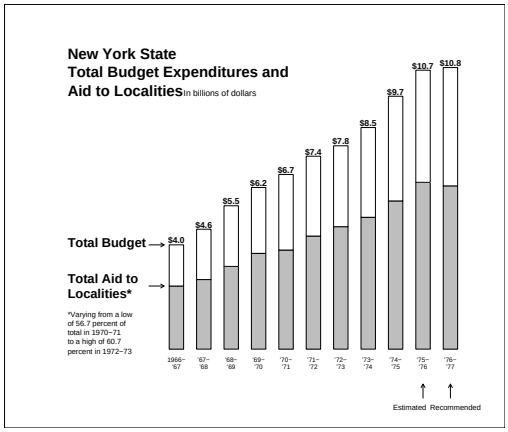


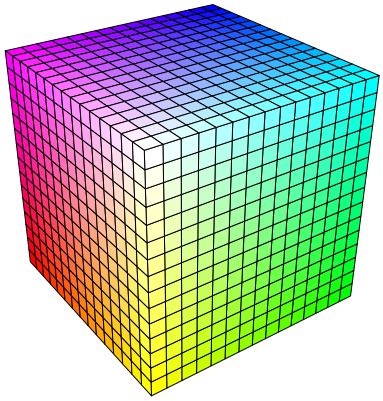
The RGB Colour Cube

Here is an image of the RGB colour cube which I use in teaching graphics.

The image was created with a small (< 100 lines of code) 3d graphics library I wrote for R.

The cube can be drawn from any angle, and the number of squares on the surface can be varied.





The Minard Map

This map shows the fate of Napoleon's army during his ill-fated invasion of Russia in 1812. This graph is presented on page 41 Tufte's book. It is often cited as one of the finest graphs every produced.

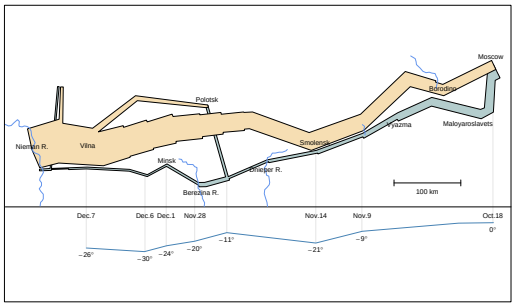
I produced this graph by scanning the original and then tracing its outline using xfig. I then extracted the coordinates from the xfig file and used them as data (in R) to reproduce the graph.

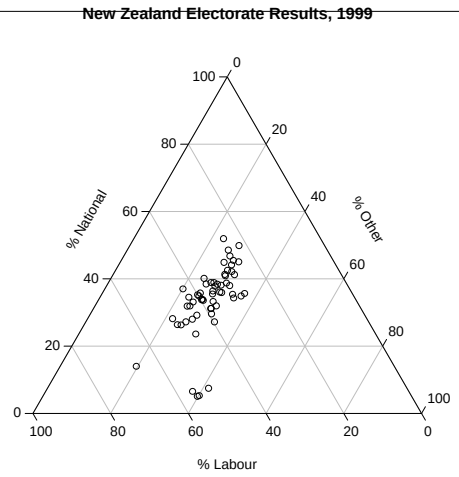
Having the graph as an R function means that I can experiment with features such as line types and colour.

A Mixture Plot

- The following plot shows a view of of the voting results in the 1999 New Zealand election.
- The plot shows how the party vote was distributed among the groups *Labour*, *National* and *Other*.
- This is a solution to an assignment in 782.

The Minard Map of Napoleon's 1812 Campaign in Russia

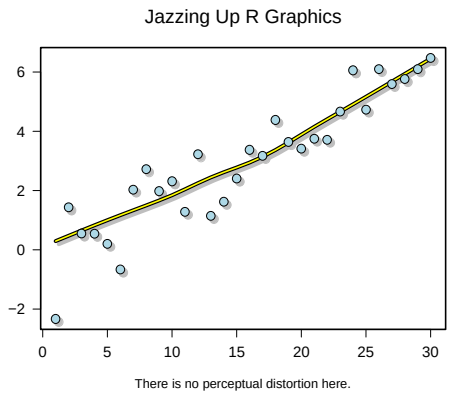
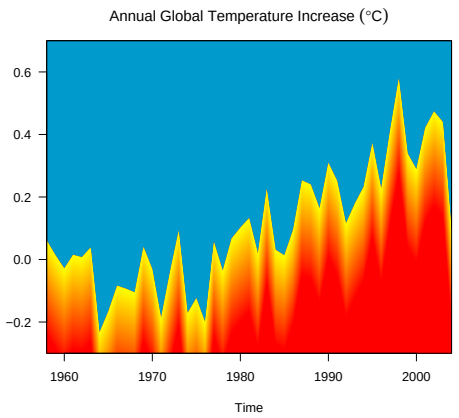




Eye-Catching Visual "Tricks"

In commercial presentation graphics it is common to add a variety of effects to make the the graphics more eye-catching.

One simple effect is the addition of *drop shadows* for objects which appear in the graph.

Shadows can be created by drawing objects twice. Once in gray and once in their real colour.

The use of colour gradients can also provide interesting "eye-candy" effects.