How to Plot with qplot in the ggplot2 Package

There are several means of creating plots in R. The main ones are “base” graphics, lattice and ggplot2. Each has advantages and disadvantages which are better discussed elsewhere. For our purposes, the qplot function in the ggplot2 package is easy to use and can create a variety of well-designed plots (qplot stands for quick plot). ggplot2 is based upon a ”grammar of graphics” which we won’t discuss here, but it corresponds closely to how most people think about creating a plot, including how one might take an existing plot and add more data or decorations to it. ggplot2 also uses fairly plain and obvious language, which makes it easy for beginners to read the instructions for a complicated plot and understand what is intended. qplot is geared toward creating plots quickly, and so makes some decisions for you, and it doesn’t have the full power of other functions in ggplot2, but it is a good place to start and more than sufficient for many purposes.

Remember that you need to install and load ggplot2 before proceeding.

The general syntax of a simple call to qplot is as follows:

```r
qplot(x = X, y = X, data = X, color = X, shape = X, geom = X, main = "my plot title")
```

The arguments are:

- **x**: The x values to plot; they must be a variable in the data frame you specify and there are no quotes around the name. Note that if you give only x values (no y values), you are plotting univariate data and qplot figures this out. However, you have to give a geom that makes sense for univariate data.

- **y**: The y values to plot; they must be a variable in the data frame you specify, and again, no quotes around the name. If you are providing y values, you have to specify a geom that makes sense with bivariate data.

- **data**: The name of data frame which contains the x and y values.

- **color**: Perhaps surprisingly, not a set of colors to use, but rather a ”mapping” of the color scheme onto some variable in your data frame. You are basically telling qplot to use different colors for different values of the variable you specify; hence this variable should be a factor, not a number (your data frame may or may not have a factor entry available for this purpose). qplot decides which colors to use.

- **shape**: Exactly as for color, except different symbols will be used for each value of the variable you specify. Note that you can use either color = ??? or shape = ??? or both, depending upon how you want your plot to look. qplot decides which symbols to use.

- **geom**: A ”geom” specification, which is basically a list of keywords describing what to plot. Common examples are ”histogram”, ”density”, ”line”, ”point” which pretty much do what they say. The geom must make sense for the kind of data you are supplying.

- **main**: The title for the plot.

Below are some examples demonstrating some common plots. Try running these yourself to see the plot they produce. The examples use the data set UScereal which contains nutritional data of cereals for sale in the United States in 1993. To access the data, use ```library("MASS")``` then ```data(UScereal)``` Use ```?UScereal``` for more background, and ```str(UScereal)``` to get a sense of what's in there.

Examples of Univariate Plots

A simple histogram of the carbohydrate levels in cereal:

```r
qplot(x = carbo, data = UScereal, geom = "histogram")
```

You can change ”histogram” to ”density” if you want the density plot. This plot displays a histogram of all carbo values in the data set, independent of the manufacturer. Remember that the appearance (and therefore interpretation) of a histogram depends upon the bins chosen; this is also true for a density plot where the smoothing is affected. You can and should vary the number of bins in your histogram before deciding on a value that makes sense for your particular data. Here’s how to do it:

```r
qplot(x = carbo, data = UScereal, geom = "histogram", binwidth = 2)
```
For greater insight into the data you might want to break things out by manufacturer. This is done with facets:

```r
qplot(x = carbo, data = UScereal, geom = "histogram",
       facets = mfr ~ .)
```

The argument `facets = mfr ~ .` means create separate plots with each manufacturer in a separate row. Try using `facets = . ~ mfr` to see what it does. Note that the data is unchanged, only the presentation differs.

To create a boxplot, you have have to specify both x and y values. In this case, the variable `mfr` is a good choice for the x dimension because it is a factor. Even though you are specifying both a set of x values and a set of y values, you are creating a series of univariate plots: the y values are the actual univariate data, the x values are factors which serve to organize the series.

```r
qplot(x = mfr, y = carbo, data = UScereal, geom = "boxplot")
```

With the boxplot, you can also overlay all the data points in two ways: plot them in a single line, or spread them out if they overlap (jitter them). Try using `geom = c("boxplot", "point")` and compare it to `geom = c("boxplot", "jitter")`. Finally, if you want a dotplot or stripchart (two names for the same thing), use the command above, but set `geom = "point"` or `geom = "jitter"`. As before, these are really univariate data but two dimensions are specified to organize the data in a useful way.

Note that in all these plots, `qplot` has figures out some defaults for the title, and the x and y labels. You can override these by specifying the character strings you want. By the way, do you see any interesting trends or possible outliers in any of this data?

### Examples of Bivariate Plots

Let’s look at some typical bivariate plots. In this case the x values are both numeric. Let’s plot the `carbo` value against the `fibre` value to see if they are related:

```r
qplot(x = carbo, y = fibre, data = UScereal, geom = "point")
```

Remember this is the whole data set, all manufacturers combined. Try changing the `geom` to "line" to see what happens. To do a quick linear fit of the data, try this:

```r
qplot(x = carbo, y = fibre, data = UScereal, geom = c("point",
                                                    "smooth"),
       method = "lm")
```

You could facet this plot by manufacturer (try it), but another approach would be to either color it by manufacturer, or use different symbols for each manufacturer:

```r
qplot(x = carbo, y = fibre, data = UScereal, geom = c("point",
                                                    "point"),
       color = mfr)
```

Which plot is best? There is no general answer, it depends upon the data and what you want to convey.

UScereal contains a second factor, related to vitamin enrichment. This allows you to break the data down even further, by faceting in both dimensions:

```r
qplot(x = carbo, y = fibre, data = UScereal, geom = c("point"),
       facets = mfr ~ vitamins)
```

This plot tells you quite a bit about the data! Remember that you can subset the data too so that you can focus in on particular parts:

```r
enrich <- subset(UScereal, vitamins == "enriched")
qplot(x = carbo, y = fibre, data = enrich, geom = c("point"),
       facets = mfr ~ vitamins)
```

See the document about selecting data for more ways to do so.

There are many other things you can do with ggplot2 to customize your plots, but these tips should enable you to make some decent plots that allow you to explore a data set carefully. Enjoy!