Advanced graphics: practical 3 solutions

Dr Colin Gillespie

This practical aims to guide you through some of the key ideas in data manipulation. I've tried to construct this practical in such a way that you get to experiment with the various tools. Feel free to experiment!

1 Factors

When using ggplot2, the easiest way of rearranging the graph or to alter labels is to manipulate the data set. Consider the `mpg` data set:

```
library("ggplot2")
data(mpg)
```

Suppose we generate a scatter plot of engine displacement against highway mpg.

```
g = ggplot(data=mpg, aes(x=displ, y=hwy)) +
  geom_point()
```

Next, we add a loess line, conditional on the drive type:

```
g + stat_smooth(aes(colour=drv))
```

While this graph is suitable for exploring the data; for publication, we would like to rename the axis and legend labels. To change the axis labels, we can rename the data frame columns or use `xlab` and `ylab`. To change the order of the legend, we need to manipulate the data. If we try:

```
mpg[mpg$drv == "4",]$drv = "4wd"
mpg[mpg$drv == "f",]$drv = "Front"
mpg[mpg$drv == "r",]$drv = "Rear"
```

we get an error message. This is because `mpg$drv` is a *factor*.[1] Instead, we use the `levels` function to change the levels:

[1] See my introduction to R notes.

```
##Reload the data just to make sure
data(mpg)
levels(mpg$drv) = c("4wd", "Front", "Rear")
```

To change the order of the, we need to use the `factor` function:

```
mpg$drv = factor(mpg$drv,
                 levels = c("Front", "Rear", "4wd"))
```

The legend now displays the labels in the order: `Front`, `Rear` and `4wd`.

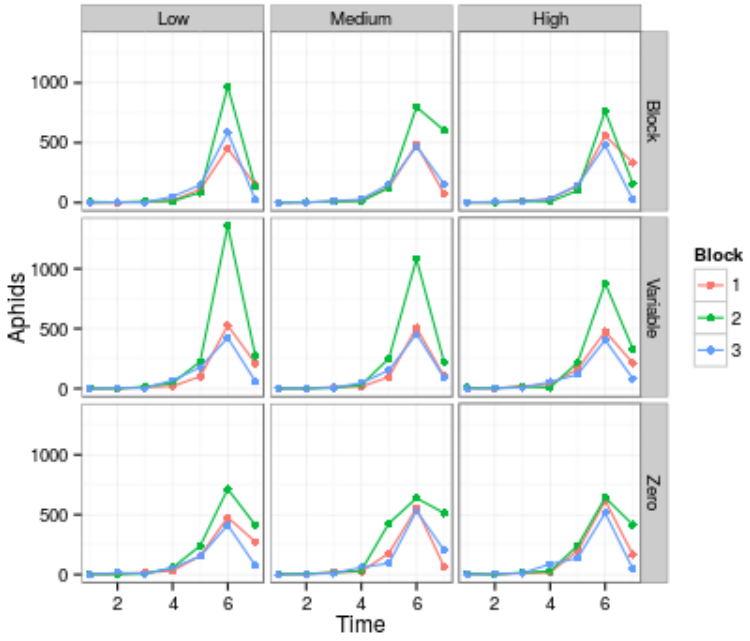

Figure 1: Final figure from section 2.

2 *Aphids*

This data set consists of five observations on cotton aphid counts on twenty randomly chosen leaves in each plot, for twenty-seven treatment-block combinations. The data were recorded in July 2004 in Lamesa, Texas. The treatments consisted of three nitrogen levels (blanket, variable and none), three irrigation levels (low, medium and high) and three blocks, each being a distinct area. Irrigation treatments were randomly assigned within each block as whole plots. Nitrogen treatments were randomly assigned within each whole block as split plots.

```
library(nclRggplot2)
data(aphids)
```

Note that the sampling times are $t = 0$, 1.14, 2.29, 3.57 and 4.57 weeks (i.e. every 7 to 8 days). Reproduce figure 1.

Here are some hints to get you started. The key idea is to think of the plot in terms of layers. So

- Leave the ordering of factors to the end
- The plot contains a combination of `geom_line` and `geom_point`.
- You can change the x-axis label using

```
+ xlab("Time")
```

- Change the theme using `theme_bw()`

```
##Code for figure 1
aphids$Block = factor(aphids$Block)
aphids$Water = factor(aphids$Water,
                      levels=c("Low", "Medium", "High"))
ga = ggplot(data=aphids) +
  geom_point(aes(Time, Aphids, colour=Block)) +
  facet_grid(Nitrogen ~ Water) +
  geom_line(aes(Time, Aphids, colour=Block)) +
  theme_bw()

print(ga)
```

3 *The Beauty data set*

First load the beauty data set

```
data(Beauty)
```

In practical 1, we split data up by both gender and age:

```
Beauty$dec = signif(Beauty$age, 1)
g = ggplot(data=Beauty)
g1 = g + geom_bar(aes(x=gender, fill=factor(dec)))
```

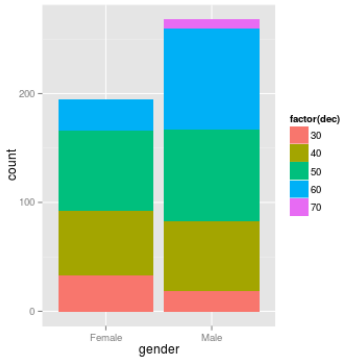

Figure 2: Stacked bar chart of the Beauty data set.

to get figure 2. Rather than using the `fill` aesthetic, redo the plot but use `facet_grid` and `facet_wrap`. For example,

```
g2 = g + geom_bar(aes(x=gender)) +
  facet_grid(. ~ dec)
```

Experiment with:

- the `margins` argument

```
g + geom_bar(aes(x=gender)) + facet_grid(. ~ dec, margins=TRUE)
```

- the `scales='free_y'` argument

```
## Notice that the females have disappeared from the "70" facet.
## Probably not what we wanted.
g + geom_bar(aes(x=gender)) + facet_grid(. ~ dec, scales="free_x")
```

- the layout, i.e. column or row.

```
g + geom_bar(aes(x=gender)) + facet_grid(dec ~.)
```

How would you change the panel labels?

```
##Relabel the factor
Beauty$dec = factor(Beauty$dec,
                    labels=c("Thirties", "Forties", "Fifties", "Sixties", "Seventies"))

## Plot as before
ggplot(data=Beauty) +
  geom_bar(aes(x=gender)) +
  facet_grid(dec ~.)
```

4 *The Google data set*

Google recently released a data set of the top 1000 websites. The data set contains the following categories: `Rank`, `Site`, `Category`, `Users`, `Views` and `Advertising`. First we load the data

```
data(google)
```

1. Create a scatter of plot of `Rank` and `Views`.
2. Using `scale_y_log10()` transform the `y` scale.
3. Uses facets to split the plot by its advertising status.
4. Use another `geom_point()` layer to highlight the *Social Networks* sites to figure 3.

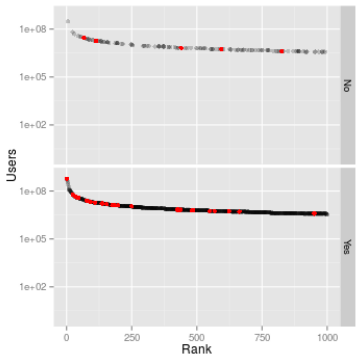

Figure 3: The google data set.

```
g = ggplot(google) +
  geom_point(aes(Rank, Users), alpha=0.2) +
  scale_y_log10(limit=c(1e0, 1e9))  +
  facet_grid(Advertising~.) +
  geom_point(data=subset(google, Category=="Social Networks"),
             aes(Rank, Users), colour="Red",  size=2)
```

5 *An example data set*

An experiment was conducted: two treatments, `A` and `B`, were tested. The data can be downloaded into R using the following commands:

```
data(cell_data)
```

Within each treatment group, there are two patient types: `Case` and `Control`. What plot do you think would be most suitable for this data?

First we'll create a base object

```
##This doesn't plot anything
g = ggplot(cell_data, aes(treatment, values)) +
  facet_grid(.~type)
```

Experiment plotting the data using boxplots, jittered points, histograms, errors, etc. Which methods is optimal (for this data set).

```
##Boxplot
g + geom_boxplot()

## Dotplot
g + geom_dotplot(binaxis="y", stackdir="center",
                 colour="blue", fill="blue")
```

```
## Plain (jittered) points
g + geom_jitter()

## Dotplots + error bars
g +  geom_dotplot(binaxis="y", stackdir="center",
                  colour="blue", fill="blue") +
  stat_summary(geom="errorbar",
               fun.ymin=min, fun.ymax=max, fun.y=mean,
               width=0.2)
```

Solutions

Solutions are contained within this package:

```
library(nclRggplot2)
vignette("solutions3", package="nclRggplot2")
```