An Example of Sharing Graphical Objects in Grid

August 28, 2001

⟨*⟩≡

Grid graphical objects are implemented as "external pointers". This is easiest to explain by example:

⟨*sharing*⟩≡
```
l <- grid.lines()
l2 <- l
grid.edit(l2, gp=gpar(col="green"))
lget(l)
```

The final command produces the following output, which shows that `l` is also now "green".

⟨*sharingOutput*⟩≡
```
$x
[1] 0npc 1npc

$y
[1] 0npc 1npc

$col
[1] "green"

$lty
[1] "solid"

$vp
NULL

attr(,"class")
[1] "lines" "glist"
```

In the above, we create a graphical object and assign it to `l`. We then assign `l` to `l2`. In most cases in R, this would copy the *value* of `l` to `l2`, but in this case it actually copies a *reference* to the value of `l` to `l2`. In other words, `l` and `l2` "point" to the same value. Thus when we modify `l2`, the change is reflected in `l`.

Shared Axes

Suppose that I am drawing two graphs on a page, which are the results from two subjects in an experiment. I want the graphs to have the same axes to aid in comparison of the subjects.

First of all, I will split the page up into two bits for the two graphs.

⟨*sharedAxes*⟩≡
```
push.viewport(viewport(layout=grid.layout(1, 2, respect=TRUE)))
```

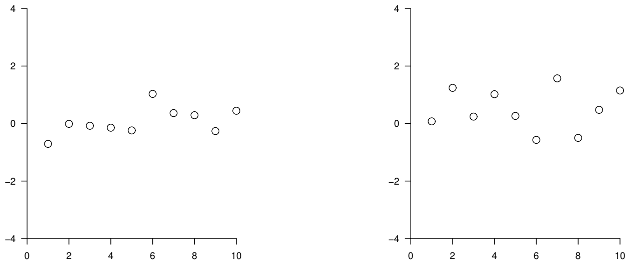
Now I generate some data and draw the first plot.

⟨*sharedAxes*⟩+≡
```
x <- 1:10
y1 <- rnorm(10)
vp1a <- viewport(layout.pos.col=1)
vp1b <- viewport(width=0.6, height=0.6,
                 xscale=c(0, 11), yscale=c(-4, 4))
push.viewport(vp1a, vp1b)
xa <- grid.xaxis()
ya <- grid.yaxis()
grid.points(x, y1)
pop.viewport(2)
```

Notice that I have saved the graphical objects for the axes. Now when I draw the second plot I will just reuse these graphical objects rather than creating new ones.

⟨*sharedAxes*⟩+≡
```
y2 <- rnorm(10)
vp2a <- viewport(layout.pos.col=2)
vp2b <- viewport(width=0.6, height=0.6,
                 xscale=c(0, 11), yscale=c(-4, 4))
push.viewport(vp2a, vp2b)
grid.draw(xa)
grid.draw(ya)
grid.points(x, y2)
pop.viewport(2)
```

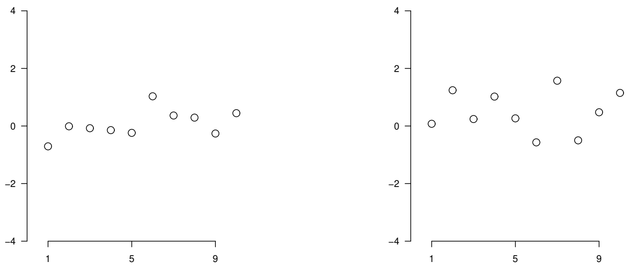
The output looks like the figure below.



Because I have used the same graphical object in both plots, I can edit the axes for both plots simultaneously rather than having to edit each one in turn. For example ...

⟨*sharedAxes*⟩+≡
```
grid.edit(xa, at=c(1, 5, 9))
```

The output now looks like the figure below.



This might seem a very small gain in this example, but it is potentially of great use in, for example, the implementation of a scatterplot matrix.