Demography 213
Graphics

Carl Mason
carlm@demog.berkeley.edu

October 17, 2016

Contents

Abstract

This week we will learn about R's graphics capabilities. Graphics are among the best features of R. The ability to add elements to graphs one at a time and to control just about every aspect of printing can lead to some pretty cool graphs. If you can make this week's assignment build on last week's IPUMS project–that's great, if not that's OK. Having some kind of vaguely demographic theme in your work is vaguely a goal of this project however you manage that is fine. Ultimately this week is your opportunity to ***create an object of transcendent beauty for display outside of the second floor bathroom.***

A second goal ***is to be able to create a stand alone R file that someone else can run*** to re-create your masterpiece. It should not be necessary to understand your code in order to run such a file, BUT since future generations of demographers will read your code, it would be an act of mercy (and embarrassment management) if your code contains lots of clear comments and very few unnecessary lines of code. And of course, the code should work.

1 Introduction

One of the best features of R is the way it produces graphs. To be more precise, I should say the "ways" it produces graphs as there are currently three different graphing system in R. All three systems are excellent at what they do and all have their zealous disciples. We will focus today on the original old fashioned R graphics because they are the most accessible and quickest to at least get started with, and because you have already seen lots of examples sprinkled throughout previous assignments.

The other two graphic system are ***Lattice*** (aka *Trellis*) (aka the graphic system that we do not use any more) and ***ggpot2***–which you saw last week. While both of these newer system share some features of the original graphics system and with each other, their differences are more prominent than their similarities. Further, they are entirely distinct from each other in that you cannot use the tools from any system to add features or modify graphs that were started with any of the other systems. All graphs must be all one thing or all the other.

2 Read Section 12 of *Introduction to R*

Read Chapter 12 (on Graphics) and Section 8.1, *R as a set of statistical tables* (page 33), in *Introduction to R.* The *Graphical Procedures* section contains a great deal of information, not all of which is perfectly intuitive. While you read it, be aware that you can use it as a reference as long as you understand what's in it. Don't try to memorize every graphic function and parameter, but understand the basic idea of how you create a plot and add lines, points and text to it.

A few things that are covered in *Introduction to R* require clarification:

2.1 R studio

R studio has some features that are not mentioned in *Introduction* because R studio itself is not mentioned in *Introduction.* As you know, graphs simply appear in the lower right panel whenever you drop a graph producing command into the console. This is a nice thing, but the graphics viewer should be viewed as only a graphics viewer. That is, do not count on simply saving a graph to a file by clicking on the menu item. While this *is* possible, it is not advisable. **Graphs that are saved from the R studio graphics viewer are generally not very good.**

2.2 Multiple plots on one page

There are two ways to put multiple figures on a single graphics device (or "page"). The one described in the book is the older simpler technique. It involves using `par(mfrow=c(n,m))` to divide the plot area into $n * m$ equal sized boxes. Most of the time this is good enough. Further, if you are planning to use your graph in a paper, then it is nearly always best to save each graph in a separate file as that will give you more freedom in formatting your paper. Similarly with titles, if you put the title of the graph on the graph in R, then you are stuck with it.

If you don't put a title in the graph, you can specify the title in your paper – which is generally a better plan.

Although I never use it myself for science, I reluctantly include a description the newer way to subdivide the printing area below – in the hopes that it has some use in art. The advantages of the newer way are

- It allows you to draw figures of different sizes on the same sheet. (which I never do)
- It allows you to redraw one of the figures on the page without redrawing the others (saves you a second or two now and then – probably less time than it took you to read this).
- It can allow you to make your graphs overlap each other in "artistic" ways.

The disadvantage is that they are more confusing, and completely incompatible with the old way of doing things. So note that this technique exists, just in case you want to use it someday – but most folks around here still use the older but still generally adequate method described in Section 12.5.4 of *An Introduction to R.*

Here is the newer way of drawing multiple graphs on single page/screen.

- To split the screen into a 2X2 grid:

▪ > `split.screen(figs=c(2,2))`

- To split the lower left box horizontally into 2 equal boxes:

▪ > `split.screen(figs=c(2,1),screen=3)`

- To stick a boring histogram in the top half of the lower left quarter of the page:

▪ > `screen(5); hist(rnorm(100))`

- To undo all the screen splitting stuff and return to normal behavior:

▪ > `close.screen(all.screen=T)`

3 Postscript, pdf, SVG, or png files

There are a lot of graphics file formats in popular use today. The right one for your work, depends on whether you plan to use your graphic creation on a web page, or in a publication, or in an informal document or in some sort of presentation software, or some other use.

While it is possible to convert any format to any other graphic format, some are **much** better for this than others. Postscript, Pdf, and SVG are ***vector graphic*** format, which means that they consist essentially of code instructing the printer to draw a line here and put a dodecahedron there. The alternative is a ***raster*** format such as .gif, jpg, or png. Raster files simply say which color each pixel should be. The advantage of vector formats are that they are smaller, more editable, and easier to convert into raster formats.

For most of the reasons above, it is generally advantageous to write your graphs to vector format files and the most common and familiar of these is pdf– so let's do that. But if you are stubborn, R can produce graphs in just about any format.

To confuse the user, R has the concept of the ***graphic device*** and all plotting commands write their stuff to the currently open graphic device. Generally this is R studio's lower right hand panel. But for reasons noted above, when you are ready to produce your final version, you cannot rely on R studio's graphic viewer to produce camera ready 21st century high quality pdfs. To do this you must:

1. open a pdf-graphic-device, specifying the file name into which the "device" will write.
2. execute the graph producing commands
3. close the pdf-graphic-device thus completing the pdf file.

For example:

```
> pdf(file="junk.pdf")  # open the graphic device
> rnorm(1000)           # create the plot
> dev.off()             # close the current graphic device
```

After executing the dev.off() command, you should be able to display the junk.pdf file by clicking on it in R studio's file viewer, or from a terminal by typing

```
@:> okular junk.pdf
```

For information on converting postscript/pdf to other formats, see the man pages on:

- `convert` for converting postscript files to raster formats like .jpg, .gif, or .png.
- `pstoedit` for converting postscript files to other vector formats such as .fig (for editing in xfig) or .svg.
- `ps2pdf` for converting postscript files to .pdf

4 Assignment

As you have done many many times now... open Rstudio and create a new project in your already existing WeekN directory. If your instructor failed to tell you about using Rstudio in a web browser – make sure you ask about it.

For this week you will use the graphics capabilities of R to create a plot of transcendent beauty. Relevance to science is neither a necessary nor sufficient ingredient in the production of beauty, however, we all know that beauty without science is an soulless gesture void of humanity and worthy of at most an A-. So if you can include at least an echo of a shadow of your results from last week, you will be a better person for it.

The requirements are:

- That **all** the code for producing your graph **and nothing else** be in a single .r file. If that file references a data file then make sure that the full path to the data file is specified so that the file will be found regardless of the current working directory. This is a very good time to get your head clear about working directories.
- That your name or a name that you answer to is on the graph somewhere
- That work deal vaguely with a theme that is vaguely demographic in nature.
- That your work **not** deal with rushing water themes. Be considerate of you audience after all.
- That you send email to cmason@berkeley.edu, with either the one complete self contained file that produces your graph (the R code NOT the graphic file) or the full path to that file.

Some suggestions:

Visit the following websites AND OTHERS in search for inspiration and for a new and different way to illuminate the exciting contribution to human learning which you produced last week.

- `http://www.statmethods.net/graphs/boxplot.html`
- `http://archive.today/addictedtor.free.fr`
- Look at the Code from previous years works. You'll need the password. If your instructor fails to provide it, please ask in class.

Look to your predecessors The course website has lots of graphs done by previous cohorts. Check them out and make yours better.