Don’t start from scratch: Interacting with your graphics

Ethan Brown
Sunday, February 22, 2015
Why interactive graphics?

- Want to tweak just a small range of parameters
- Repetitive to run the same code over and over again
- Sometimes the best way to see or explore your data
- Now really easy in RStudio!
A quick example of an interactive graph

Don’t start from scratch: Interacting with your graphics
Two easiest interactive graphics packages

- manipulate: quick sliders & buttons right in RStudio
- shiny: sliders, buttons & more for documents, presentations, & web apps
Setting up for `manipulate`

It comes with RStudio! So all we need to do is:

```r
library(manipulate)
```

Also, to view these examples:

```r
library(ggplot2)

## Filter diamonds to reasonable subset

diamonds2 = subset(diamonds, x > 0)
```
A static graph

alpha parameter controls transparency

ggplot(diamonds2, aes(x = x, y = carat)) +
  geom_point(alpha = 0.5)
Interactive version

Click on the "gear" icon in RStudio.

Note the braces around the original command.

```r
manipulate({
  ggplot(diamonds2, aes(x = x, y = carat)) +
  geom_point(alpha = myalpha)
},
  myalpha = slider(min = 0, max = 0.5, initial = 0.25)
)
```
Widget types

- slider(min, max, initial): continuous change
- picker(..., initial): chose from several options,
- button: one-time change (like regenerating a simulation); a little tricky to set up
Choosing a subset with the picker

```r
manipulate({
  subdiamonds = subset(diamonds2, cut == mycut)

  ggplot(subdiamonds, aes(x = x, y = carat)) +
  geom_point(alpha = myalpha)
},
  myalpha = slider(min = 0, max = 0.5, initial = 0.25),
  mycut = picker("Fair", "Good", "Very Good", "Premium", "Ideal")
)
## Zoom & tweak text size

```r
## Zoom in on a particular region of plot
manipulate(
  ggplot(diamonds2, aes(x = x, y = carat)) +
    geom_point(alpha = myalpha) +
    coord_cartesian(xlim = c(xmin, xmax)) +
    theme_bw(base_size = textsize)
},
  myalpha = slider(0, 1, initial = 0.5),
  xmin = slider(0, 11, initial = 0),
  xmax = slider(0, 11, initial = 11),
  textsize = slider(10, 100, initial = 20)
)
```
Fitting a polynomial

```r
manipulate( {
    ggplot(diamonds2, aes(x = x, y = carat)) +
    geom_point() +
    geom_smooth(method = "lm", formula = y ~ poly(x, deg)) +
    ggtitle(paste("Fit with polynomial of degree", deg))
},
    deg = slider(1, 10))
```
Don’t start from scratch: Interacting with your graphics

**Histogram width**

```r
manipulate({
  ggplot(data = diamonds2, aes(x = price)) +
    geom_histogram(binwidth = mybin)
},
  mybin = slider(100, 1000, initial = 500))
```
shiny

- manipulate is easy, but what if you want to have several plots?
- Or, embed in presentation like this one?
- shiny is way more powerful (and a bit more complex)
- Easiest to use with R Markdown
Set up shiny

In RStudio, open File > New > R Markdown and choose Shiny (either an HTML document or presentation).

RStudio will prompt you to install or update the shiny package if you need to.
Setting alpha with shiny

renderPlot({
    ggplot(diamonds2, aes(x = x, y = carat)) +
    geom_point(alpha = input$myalpha)
})

inputPanel(
    sliderInput("myalpha", label = "Alpha value: ",
                min = 0, max = 0.5, value = 0.25)
)
Don’t start from scratch: Interacting with your graphics

Alpha result

![Graph showing a scatter plot with X and carat axes. The alpha value slider is set to 0.25.]
Two functions:

- **renderPlot**: all your plot-generating code. For a variable you want to control, refer to that variable as `input$myvariable`.
- **inputPanel**: Specifying the variables you're going to change using sliders or menus.

**NOTE**: You cannot repeat variable names in the same document.
**inputPanel**

Includes whatever input devices you want.

`sliderInput(inputID, label, min, max, value)`: Again, this is a slider.

- `inputID` is the variable name you refer to in `renderPlot`
- `label`: What the user sees
- `min, max` are the minimum and maximum of the slider
- `value` is the initial value of the slider
**selectInput**

- *inputID* is the variable name you refer to in `renderPlot`
- *label*: What the user sees
- *choices*: a vector of choices
- *selected*: the default choice
Choosing a subset with the selectInput (code)

```r
inputPanel(
  sliderInput("myalpha2", "Alpha:",
    min = 0, max = 0.5, value = 0.25),
  selectInput("mycut2", "Cut:",
    choices = c("Fair", "Good", "Very Good", "Premium"))
)

renderPlot({
  subdiamonds = subset(diamonds2, cut == input$micut2)

  ggplot(subdiamonds, aes(x = x, y = carat)) +
  geom_point(alpha = input$myalpha2)
})
```
Choosing a subset with the picker
(output)
## Zoom & tweak text size (code)

```r
inputPanel(
  sliderInput("myalpha3", "Alpha", 0, 1, value = 0.5),
  sliderInput("xmin3", "Minimum", 0, 11, value = 0),
  sliderInput("xmax3", "Max", 0, 11, value = 11),
  sliderInput("textsize3", "Text Size:", 10, 100, value = 20)
)

## Zoom in on a particular region of plot
renderPlot({
  ggplot(diamonds2, aes(x = x, y = carat)) +
  geom_point(alpha = input$myalpha3) +
  coord_cartesian(xlim = c(input$xmin3, input$xmax3)) +
  theme_bw(base_size = input$textsize3)
})
```
## Fitting a polynomial (code)

```r
inputPanel(
  sliderInput("deg", "Polynomial Degree", 1, 10, value = 1)
)

renderPlot( {
  ggplot(diamonds2, aes(x = x, y = carat)) +
  geom_point() +
  geom_smooth(method = "lm", formula = y ~ poly(x, input$deg),
  ggtitle(paste("Fit with polynomial of degree", input$deg)))
```
Fitting a polynomial (output)

Polynomial Degree

Fit with polynomial of degree 1

Don’t start from scratch: Interacting with your graphics
inputPanel(
    sliderInput("mybin", "Bin Width", 100, 1000, value = 500)
)

renderPlot({
    ggplot(data = diamonds2, aes(x = price)) +
    geom_histogram(binwidth = input$mybin)
})
Don’t start from scratch: Interacting with your graphics

Histogram width (output)

Bin Width

<table>
<thead>
<tr>
<th>100</th>
<th>500</th>
<th>1,000</th>
</tr>
</thead>
</table>

![Histogram Graph]

Count

Price
shiny does a lot more

- Widgets for inputting data
- Custom themes
- Pretty elaborate fancyness
# Table output

Columns in diamonds to show:
- carat
- x
- price

Show **10** entries

Search:

<table>
<thead>
<tr>
<th>carat</th>
<th>x</th>
<th>price</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.23</td>
<td>3.95</td>
<td>326</td>
</tr>
<tr>
<td>0.21</td>
<td>3.89</td>
<td>326</td>
</tr>
<tr>
<td>0.23</td>
<td>4.05</td>
<td>327</td>
</tr>
<tr>
<td>0.29</td>
<td>4.2</td>
<td>334</td>
</tr>
<tr>
<td>0.31</td>
<td>4.34</td>
<td>335</td>
</tr>
</tbody>
</table>

Showing 1 to 5 of 53,932 entries
Resources

- R Studio guide to Shiny and R Markdown (http://rmarkdown.rstudio.com/authoring_shiny.html)
- Official shiny documentation (http://shiny.rstudio.com/tutorial/)—advanced users only!