More R

Hadley Wickham & Heike Hofmann

Outline

- Advanced graphics
- Creating new variables
- Functions and documentation
- Subsetting
- Loading data

Advanced graphics

- So far have only made graphics with a single layer, but it's often useful to have more
- `qplot(carat, price, data=diamonds) + stat_smooth(method=lm)`
- `qplot(cut, price, data=diamonds, geom="boxplot") + geom_jitter()`

Adjusting defaults

- Layers also allow us to adjust the default colours, sizes, shapes etc:
- `qplot(cut, price, data=diamonds, geom="boxplot") + geom_jitter(colour="red")`
- Sometimes we don't want anything on the first layer:
- `qplot(carat, price, data=diamonds, geom="blank")`

Transparent colours

- `qplot(carat, price, data=diamonds, geom="blank") + geom_point(colour=alpha("black", 0.5))`
- Useful when lots of data
- Run `colors()` to see all colour names. Alpha values should be between 0 (transparent) and 1 (opaque)

Your turn

- Explore the effect of varying transparency on a plot of price vs carat. What do you see when the points are very transparent? What about when they are very opaque?
- What does adding a smooth line tell you? Explore the examples on http://had.co.nz/ggplot2/stat_smooth.html and see if you can fit a better curve

Creating variables

- One at a time:
  - `diamonds$pricepc <- diamonds $price / diamonds$carat`
- Multiple
  - ```
    diamonds <- transform(diamonds,
      pricepc = price / carat,
      perimeter = x + y + z
    )
    ```
    ```

Notes

- As we starting doing more complex R, you will often need to type things into more than one line.
- This is a **pain** in R!
- So use a text editor (or word) and copy and paste between (remember Alt+Tab, Ctrl+C, Ctrl+V)
- Also useful for keeping track of what you did

Your turn

- Create new variables to estimate volume and density. Explore the new variables with graphics.
- Calculate depth for yourself (you'll need to look up the definition). How does it compare to the precalculated value?

Functions

- `abs, sign`
- `sqrt, exp, log`
- `floor, ceiling, trunc, round, signif, round_any`
- `cos, sin, tan, acos, asin, atan`
- `mean, median, sd, var, sum`

Getting help

- You can get help about any function by typing ? in front of it
- The documentation can be hard to read, but persevere and experiment! The examples are often helpful

Your turn

- Look up the help for the functions in the previous slide and confirm you know what they do
- What is different about the last row of functions?

Subsetting

- Like new variables, there are two ways. We'll just learn the easy one today
- ```
  dsmall <- subset(diamonds, carat < 1)
  lowqual <- subset(diamonds, clarity %in%
  c("I1", "SI2", "SI1"))
  ```
- You can use:
  - `< > <= >= == %in%`
  - `carat < 1 & price > 500`
  - `colour == "D" | cut == "Fair"`
  ```

Your turn

- Extract all diamonds that have a particularly low price per carat (use a histogram to figure out what low should be)
- Extract the diamonds with the best cut, colour and clarity
- Extract the diamonds with very high values of x, y, or z

Loading data

- We will use csv (comma separated files) because every program can both save and them
- `somedataset <- read.csv(file.choose())`
- Always check with `str()` that the file has loaded correctly

Your turn

- Open the Shangri La data in excel, save it as csv, and then load into R.
- Open the baseball data in excel, save it as csv, and then load into R.
- Check that they look ok using `str()`
- Open the csv in word. Try mucking around with to see if you can break the import

Feedback

- How have you liked learning R so far? Too fast? Too slow?