Online publishing

Thomas de Graaff & Daniel Arribas-Bel

September 6, 2014

Introduction

Recap

- Ideally everything is based on text
- Markdown is a very easy markup language
- Pandoc can combines languages and convert to `pdf` and `HTML`
- This means that your writings (articles & presentation) can be converted simultaneously to:
  - `pdf` for dead trees
  - `HTML` to be published on *your* website

Goal

- to make your research as reproducable as possible and **open**
  - so that others can actually reproduce it!
- This means *publishing* your paper together with the rest of your analysis, including
  - code
  - data (if infeasible—descriptives)
  - figures

Why again?

- Actually makes *you* more visible
- Easier to collaborate
- Enforces your to work 'tidy'
- For the 'greater good'
  - faster dissemination
  - ultimately reduces errors (e.g., Piketty, Reinhart and Rogoff)

Final part of this workshop

1. Combining code and text
2. Creating presentations
3. Online publishing
4. Bring it all together

Knit it all together

Combining code and text

- `RStudio` uses the package `knitr`
- Actually knitting in chunks of code within text.
- example

Discussion: code within text or as separate files?

Knitr options:

- Typically:

````
{```}{r, echo=FALSE, results='hide',
        warning=TRUE, warning=TRUE, message=TRUE}
rnorm(10)
{```}
````

- Or:

````
{```}{r, echo=FALSE}
source("./file.R")
{```}
````

Presentations

Slides

- **pdf** — good for printing (handouts)
- **HTML**
  - enables dynamic presentations
  - enables incorporating slides in websites/blogs, etc.

Markdown, pandoc and a bit of LaTeX

- In RStudio remarkably easy to incorporate
- If you have ever made beamer slides this is a huge **time saver**
- Makes you flexible.
- And incorporates code!
- So, with one make file presentations are updated as well!

Publishing online

Github's repositories

- `git` is versioning application, but:
  - research is not backed up
  - and not yet open
- That is why we use `Github` :
  - requires inlogname + psswrd
  - Creates your own repository space (just like LinkedIn or, worse, Facebook!)
    - For all materials (extends `.txt` files)
    - Allows for corporation (with known and unknowns)
    - And finally allows to create your own website

How does Github work?

- For complete packages (it is not a file server)
- It is **open**
  - Everyone can download your stuff/you can download everything (datestamps!)
  - Which also means that everyone can contribute to your code (actually push & pull request)
  - Collaboration is a breeze (using the automatic `diff` commands)
  - Very intuitive gui's

How does Github work? (cnt.)

- So you `push` to a repository
- And you `pull` from a repository
- If you like something you can `fork` a repository (on Github) > Forking a repository allows you to freely experiment with changes without affecting the original project.
- If you would just like a copy of a repository on a computer you should use `clone`

And finally, websites

- Using Github pages (`gh-pages`); you get something like: `darribas.github.io/WooWii/`
- Useful for blogs
- Showcase of projects `http://darribas.org/bits/`

Other publication channels

- for Git other open repositories such as Bitbucket
- for R:
  - RPubs
  - create a package from your paper (and publish on CRAN)
  - iPython notebook viewer

Assignment

Assignment 5

- go to `/WooWii/Paper/Assignment5/`
- Incorporate source files using `knitr`
- publish on your own github repository
- if time permits and with OsX, create a makefile structure to create with one command
  - `pdf` output
  - `doc` output
  - `html` output