Introduction

Fundamentals of Computing and Data Display

Christoph Kern

c.kern@uni-mannheim.de

09/09/2019

Outline

1. Introduction
   - Course outline
   - Canvas
   - Credit points
2. Git and GitHub
   - Git
   - GitHub
   - Practical session
   - Resources

Introduction

Course Description

- Intro to computing tools for gathering, handling and exploring diverse (web) data
  - "Exploratory Data Science" with R
  - General web data tools, SQL, Markdown, git, ...
- Course structure follows data science pipeline

Course Objectives

- Gain computational skills to gather and process data from the web
- Learn to extract and display information from these data
- Organize reproducible coding projects

Introduction

Motivating Example: Prevalence, perception, and socio-geographic structure of criminal incidents in Chicago, IL

1. Gather data from (various) web sources
2. Set up database
3. Combine and re-structure data sets
4. Data wrangling, transforming variables
5. Data exploration, data display (“data products”)
6. Communicate, document results

Introduction

Motivating Example: Prevalence, perception, and socio-geographic structure of criminal incidents in Chicago, IL

1. Gather data from (various) web sources
   - → Web scraping, APIs
2. Set up database
   - → SQL
3. Combine and re-structure data sets
   - → `tidyr`
4. Data wrangling, transforming variables
   - → `plyr`, `dplyr`
5. Data exploration, data display ("data products")
   - → Clustering, PCA, `ggplot2`, `shiny`
6. Communicate, document results
   - → `rmarkdown`

Introduction

Motivating Example: Prevalence, perception, and socio-geographic structure of criminal incidents in Chicago, IL

1. Gather data from (various) web sources
   - → Web scraping, APIs
2. Set up database
   - → SQL
3. Combine and re-structure data sets
   - → `tidyr`
4. Data wrangling, transforming variables
   - → `plyr`, `dplyr`
5. Data exploration, data display ("data products")
   - → Clustering, PCA, `ggplot2`, `shiny`
6. Communicate, document results
   - → `rmarkdown`

git, GitHub

Course outline

Table: Course outline

Date	Content
09/09/2019	Introduction to git and GitHub
09/16/2019	Web scraping, html, xml, json, APIs, regular expressions
09/23/2019?	R style guide, data structures in R, functional programming
09/30/2019	Data Wrangling: split-apply-combine, dplyr, tidyr
10/07/2019	Collecting Twitter data
10/14/2019	Databases, SQL, bigrquery
10/21/2019	Big data processing, data.table, doParallel
10/28/2019	Mid-term presentations
11/04/2019	Data exploration, Clustering, PCA
11/11/2019	Data display with ggplot2
11/18/2019	Interactive graphs, shiny, plotly, ggvis
11/25/2019	Communicate with RMarkdown
12/02/2019	Analysis and programming tools, purrr, broom, simulations
12/09/2019	Final presentations

Course outline

Recommended Textbooks:

Baumer, B. S., Kaplan, D. T., and Horton, N. J. (2017). *Modern Data Science with R*. Boca Raton, FL: Chapman & Hall/CRC Press.

Foster, I., Ghani, R., Jarmin, R. S., Kreuter, F., and Lane, J. (Eds.). (2017). *Big Data and Social Science: A Practical Guide to Methods and Tools*. Boca Raton, FL: CRC Press Taylor & Francis Group.

Wickham, H. and Grolemund, G. (2017). *R for Data Science*. O'Reilly.

Wickham, H. (2015). *Advanced R*. Boca Raton, FL: CRC Press Taylor & Francis Group.

Course outline

Salganik, M. J. (2017). Bit by Bit: Social Research in the Digital Age. Princeton, NJ: Princeton University Press. https://www.bitbybitbook.com/

Canvas

U-M: `SURVMETH 727`

Course ID: 323061

Credit points

Coursework: 6 assignments

- Completed by each student
- Short coding exercises to recap topics studied in class

Data project: Using web data to tackle a social science research problem

- Teams of two (recommended)
- Mid-term presentation: Outline research question, data sources, data gathering
- Final presentation: Present preliminary results
- Term paper: Write-up of project
    - Includes short motivation, data (sources, gathering steps), results (e.g. graphs, exploratory analysis)
    - Extended **Rmarkdown document** with code of project (.rmd > .pdf, 5-10 pages)
    - Includes link to **GitHub repository**
    - Due 12/13/2019

Credit points

Grade Distribution

- Each assignment (per student), presentation and paper (per team) will be given a grade between 0 and 100
- A missing submission will be scored as zero
- A submitted term paper is a precondition for passing the course
- Final grade: 35% assignments (averaged, without lowest score), 10% mid-term presentation, 15% final presentation, 40% term paper

Credit points

Project examples from previous courses

- *'Nowcasting Gentrification' with Publicly Accessible Data – Toward Simpler Predictions of Neighborhood Change Using Yelp*
- *Estimating Crop Yields Using Alternative Data Sources – Twitter Approach*
- *Who Tweets about China's Politics? Political Discussion and Online Bots during China's 19th National Congress*
- *Are Millenials Killing Causal Dining? Investigating the Relationship Between Demographics and Business Locations*
- *Understanding the Opioid Crisis in Midwest, United States – Focusing on Naloxone*

Git and GitHub

1. Introduction
   - Course outline
   - Canvas
   - Credit points
2. Git and GitHub
   - Git
   - GitHub
   - Practical session
   - Resources

Git and GitHub

Organizing coding projects

- Data/ coding projects often involve many iterations
- Organizing versions particularly difficult in collaborative projects
- Manual versioning (final_paper_version_x.doc) can become cumbersome

→ Version control management systems!

- Keep track of changes (**who** changed **what** code **when**?)
- Most recent version visible, previous versions can be restored
- Work best with text files (e.g. .txt, .R, .md, .tex)

Git and GitHub

Organizing coding projects

- Data/ coding projects often involve many iterations
- Organizing versions particularly difficult in collaborative projects
- Manual versioning (final_paper_version_x.doc) can become cumbersome

→ Version control management systems!

- Keep track of changes (**who** changed **what** code **when**?)
- Most recent version visible, previous versions can be restored
- Work best with text files (e.g. .txt, .R, .md, .tex)

Git and GitHub

(Why) Git and GitHub?

- Git is a version control management system
  - `https://git-scm.com/`
  - (Most) popular software for organizing coding projects
  - Other VCSs available, e.g. Subversion (SVN)
- GitHub is a **remote host** for **local** Git repositories
  - Widely used provider for sharing and storing Git projects online
  - Can also be used as a file hoster without Git
  - GitHub alternative; GitLab

Git

Working with Git

- Command line
- Graphical User Interface (GUI)
  - Git GUI
  - https://git-scm.com/downloads/guis
  - https://desktop.github.com/
  - RStudio

Git configuration

- First-Time Git Setup
  - `git config --global user.name "Your name"`
  - `git config --global user.email your@email.com`
  - `git config --global core.editor editor_name`

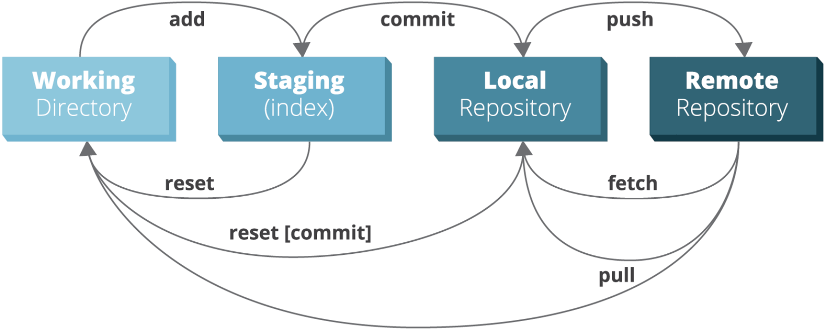
Git init

Initialize a Git repository

- Create a new (local) project
  - `git init`
- Create a new (local) R project
  - Initialize Git within an RStudio project
- Clone an existing remote project
  - `git clone`

Git workflow

Figure: Git workflow[1]



[1] https://zeroturnaround.com/rebellabs/git-commands-and-best-practices-cheat-sheet/

Example

```
> cd my_working_directory
> git init
# create some_file.R
# hack, do some work, hack
# hack
> git status # check current state of repository
> git add some_file.R
> git commit -m "Initial commit"
# hack
# more hacking
> git diff # show differences between file version
> git commit -am "Add important code"
> git log # list of previous commits
```

Git concepts

Branches

- A branch is a set of code changes that are kept separate
- Default branch created by `git init`: `master`
- Pointer to current branch: `HEAD`

`.gitignore`

- A file that tells Git which types of (e.g. temporary) files to ignore

Remotes

- A remote is an external repository to sync with
- Set up by `git clone` or `git remote add`
- Default remote name: `origin`

GitHub

Add a Git project to GitHub

1. Create a new clean repository via the GitHub web interface
2. Copy the resulting remote repository URL
   - `https://github.com/user_name/repo_name.git`
3. Add remote repo to Git project; `git remote add`
4. Interact with remote repo with `git push` and `git pull`

Notes!

- Public repositories are visible to everyone (open read access)
- Never push sensitive information (e.g. passwords) to a remote repository!

Example continued

```
> git remote add origin url
> git remote -v # list remote repositories
> git push origin master # push changes to GitHub
# sleep
> git pull origin master # update local repository
# hack, hack, hack, make some changes
> git commit -am "Some changes again"
> git push # push to GitHub
```

Branching

Git allows to play around with new/ experimental code via branches

1. Create a new branch "experimental" and switch to it: `git checkout -b experimental`
    1. `git branch experimental`
    2. `git checkout experimental`
2. Add and commit changes in new branch
    - ...until new branch is ready to be merged...
3. Switch back to master branch: `git checkout master`
4. Merge (no longer) experimental branch into master: `git merge experimental`
    - Resolve merge conflicts

GitHub flow

Workflow in a collaborative Git project

1. Clone or fork project and pull current version
2. Create a new branch and switch to it (`git checkout -b`)
3. Make changes, commit and push to remote branch
4. Create a pull request (via GitHub)
5. Proposed changes are reviewed and eventually merged into master branch (`git merge`)
6. Pull changes and tidy up branches (`git branch -d`)

Practical session

Setup: Install Git, get a GitHub account

1. Work with Git and GitHub via the command line
2. Collaborative coding with Git and GitHub

Resources

- Pro Git book
  - `https://git-scm.com/book/en/v2`
- Cheatsheet
  - `https://services.github.com/on-demand/downloads/github-git-cheat-sheet.pdf`
- Resource collection
  - `https://try.github.io/`
- Git and GitHub
  - `https://help.github.com/articles/git-and-github-learning-resources/`
- Git and R, RStudio
  - `http://happygitwithr.com`
  - Gandrud, C. (2015). Reproducible Research with R and R Studio. New York: Chapman and Hall/CRC.

Resources

`https://github.com/chkern/git-intro`