R for life and Behavioural Sciences assignment 1: Git

M. Kampert and V. Buurman

18-09-2016

Introduction

This assignment is about using Git. The assignment is a scenario. In the scenario a researcher, or research group, have been performing a theoretical experiment in `R`. Their attempt was to:

- sample some random data
- use an `R` procedure to fit a linear model to this random data
- use R plotting procedures to make some graphical representation of the fit of the fitted linear model, compared to the fit of an ideal model.

The research group has been using Git to track their work.

However, disaster has struck! The researchers know a few things about Git, and have for example been able to start a branch. In this branch they experimented with some more 'advanced' `R` plotting options. Unfortunately, none of them know how Git really works and once everything got messed! They quit working on the project because all their work seemed lost…

Task

Your main assignment is to fix the project. We know from the researchers that:

- at some point the main research line (the master branch) was still OK and stable
- at some point there was a stable version of the experimental plot options development line

Merge two stable versions of the two development lines. Use Git commands, such as `git log`, `git status`, `git checkout`, etc. to find out what has been going on and what has been changed. Try to use the `R` scripts if possible to figure out what the researchers were trying to do, to make some sensible decisions about any merge conflicts that might crop up.

Assignment requirements

Of course we want to know how you fixed the project. That is why we want you to keep track of the git commands you used. Make a report using `R` Markdown. Call it `<surname>_<student_number>_rlb_git_assignment.Rmd`, e.g. `buurman_0730106_rlb_git_assignment.Rmd`. Use it to create a `pdf` document. In this report, write down the Git commands you used to fix the repository. Elaborate on these commands: why did you choose them, what did they do? Make sure you use `R` Markdown to nicely format the lines of code you display in your report. For example: `{bash, eval=FALSE, echo=TRUE}` will allow you to start a chunk of code with Git commands that are nicely highlighted:

```bash
git status # I asked to see the status of the current HEAD,
#what I saw was a bunch of commits.

git log # TO DO FOR SELF:
#change this to a nicer version of log that is more insightful
```

(Our advice: `Rmd` together with git commands in `bash` is different from what we have seen up till now. With the git command we make changes in the file structure of the working directory, whereas this was not the case

with R code chunks that we have seen so far. If you don't feel comfortable enough yet with `Rmd` and `bash` code chunks, then make sure that every `bash` code chunk the argument `eval` is set to `FALSE`)

Of course we don't want to see TO DO's TO SELF, but finalized versions. We DO want comments. In the report **also** right down the most important/interesting findings about the repository: did you notice any strange commit messages? What were the merge conflicts you encountered? etc. Put the finalized results of the `R` code that you managed to save in the report as well. For example: it seems the researchers were busy trying to make a plot, include the plot in your report!

Other recommendations for your report:

- Make sure you discuss to some extent what was going on the merge: e.g. Were there any conflicts? What changed?
- Comment on the changes you make, "maybe" (definately) the researchers made errors in the R files: when you fix these, make a comment!
- Finally we want you to tag your final commit to the project using as tag reference your studentnumber and as tag comment your full name.

Deadline: December 1st, 9:00.

Important

- Upload your report to blackboard. We will grade your assignment based on this file.
- See the list on blackboard for the folder number you were assigned

Hints

- Start out with reading the log of the git repository.
- Set the working directory of your R session to the directory of your git repository (not the .git folder), this is what the researchers did (and will thus make sure that at least the references in the R scripts are correct).
- If you've tried a lot of git commands and accidentally messed up the repository beyond repair, no worries! Remember: you can always 'reset' your entire Git project by removing the entire directory and replace it with the one you originally used to start the assignment.
- The first stable versions are exactly that: they're stable. So try those when you want to figure out what is going on in the R-scripts.
- `git log -p` is your friend
- The repository is 'fixed' when the `R` code is running according to the plans of the researchers
- The Git commands discussed in the slides and exercises are enough to succesfully complete the assignment
- We just want you to show us that you know how to work with `R`, Git and Markdown. Getting to an answer, deliberately, with careful consideration (as written down in your report), is for this assignment more important than getting the answer correct.