Version control

How GIT can make your life better

Simon R. White

MRC Biostatistics Unit, University of Cambridge

2018/Apr/11

Outline and aims

Outline

- Version control
- GIT fundamentals (the why)
- GIT basics (the how)
- Advanced GIT

Aims

Open to discussion and happy to change focus

- Understand concepts of version control
- Highlight reasons to spend the time/effort to embrace version control

Backing up and version control

Version control and backing up your work are different:

- A backup is needed if, say, you lose your laptop
  within the BSU our home directories are backup up
- Version control is about managing and recording changes to a specific set of files/documents

Backing up and version control

Version control and backing up your work are different:

- A backup is needed if, say, you lose your laptop
  within the BSU our home directories are backup up
- Version control is about managing and recording changes to a specific set of files/documents

What you might be doing at the moment:

- Dropbox:
  a backup - does include some version control (ish)
- Saving copies:
  '`Paper Apr10.tex`', '`Paper Apr8 + SWcomment.tex`', etc.

Backing up and version control

Version control and backing up your work are different:

- A backup is needed if, say, you lose your laptop
  within the BSU our home directories are backup up
- Version control is about managing and recording changes to a specific set of files/documents

What you might be doing at the moment:

- Dropbox:
  a backup - does include some version control (ish)
- Saving copies:
  'Paper Apr10.tex', 'Paper Apr8 + SWcomment.tex', etc.

Question: what if your laptop died right now?

When to use version control?

- Most version control systems, including GIT, excel with text-based files
- Microsoft Word (and Excel) are not great formats for this
  - MS Word has its own track-changes features

A proper version control system

Benefits

- Robust record of work/code
- Collaborative work can be a lot easier

Issues

- Need to learn to use new software
- Can complicate things as you shift the way you work

Options

- subversion
- Mercurial
- GIT
- Many others

Features to consider:

- Server-Client or Distributed
- Direction of the wind today

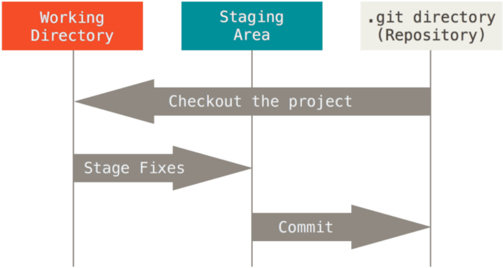
Ways to use GIT

- command line
- graphical user interface (GUI)
  `https://git-scm.com/download/gui/windows`

Today we'll do a little of both.

GIT fundamentals

- Every repository contains the complete history (this means they can get big)
- There are three parts to work with



- A repository on its own is not safe

Online resources

- https://git-scm.com/book/en/v2/Getting-Started-About-Version-Control
- https://git-scm.com/book/en/v2/Git-Basics-Getting-a-Git-Repository

GIT basic commands

- make a new repository

  ```
  git init
  ```

- staging files

  ```
  git add
  git add -u ./
  git add -A ./
  ```

- commit change-set to repo

  ```
  git commit -m"testing"
  ```

GIT overviews

- Details of commits

```
git log
git log -n2
git log --name-status -n2
```

- Details of current working directory

```
git status
```

- History

```
git hist
```

```
## Config file
[alias]
    hist   = log --all --pretty=format:\"%C(auto)%h%Creset %C(magenta)%ad%Creset
                                         |%C(auto)%d%Creset %C(auto)%s%Creset
                                         %C(blue)[%an]%Creset %C(green)<%cr>%Creset\"
                 --graph --date=short
```

GIT basic commands

- make a new repository

  ```
  git init
  ```

- staging files

  ```
  git add
  git add -u ./
  git add -A ./
  ```

- commit change-set to repo

  ```
  git commit -m"testing"
  ```

Reasons to use GIT

Easily integrate with latexdiff

Config file:

```
[alias]
    wdiff = diff --color-words --ignore-all-space
    ldiff = difftool -y -t latex
[difftool.latex]
    cmd = latexdiff "$LOCAL" "$REMOTE"
```

```
git diff v1.0..v1.1 Outline.tex
git ldiff v1.0..v1.1 Outline.tex > LDIFF.tex
```

GNU Meld for viewing differences

Config file:

```
[alias]
    mdiff = difftool -t meld

git mdiff v1.0..v1.1 Outline.tex
```

Ignoring files (for cleaner repositories)

See GIT-talk repo

The status command can get quite busy (especially with unnecessary LaTeXfiles).

```
## create a file in repository called .gitignore
## list patterns to ignore, such as

*.out
*.aux


## run the status command to see difference
git status
```

Working with multiple clones

See GIT-talk repo (and clones)

Using this example repository, make a clone

```
git clone /path/to/repo/on/your/computer NEW-NAME

## within clone

git remote -v ## shows remotes this repo is aware of

git fetch origin ## fetch commits from remote repo

## within first repo

git remote add DAVE /path/to/NEW-NAME

git fetch DAVE
```

Merges with multiple committers

See NSPN-live repo

Using this example repository

```
## bring branches together
git checkout master

git diff git-magic^..git-magic
## or
git diff 8c0c706..04aebee

git branch TEST
git checkout TEST

git merge git-magic
```

Cloned repos for code

See FM-p repos

Create local clones of repository for running code, while continuing to edit the original

```
git clone /path/to/repo Cluster-cpy-A
```

Dropbox trick (advanced paths)

See NSPN dropbox+repos

By default the `.git` *folder sits inside the working directory, but it can be elsewhere. For Dropbox, we can host a repo inside a dropbox folder without the associated* `.git`

```
git --git-dir=/path/to/.git
    --work-tree=/path/to/working-directory
```