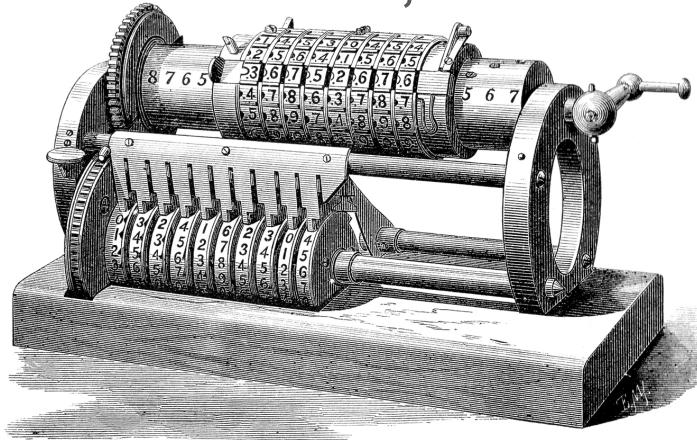


Programming in R

Bender
Niebisch, Burk



2022-04-28

This Session

1. Motivation
2. Orga
3. Lecture: Git
4. First homework

Andreas Bender

- 2nd year PostDoc "Statistical Learning and Data Science" (Bischl)
- Coordinator of the Machine Learning Consulting Unit @MCML
- 3 packages on CRAN,
 - pammtools
 - coalitions
 - mlr3proba
- Often on the lookout for HiWis
 - [adibender](#) on GitHub and Mattermost
 - andreas.bender@stat.uni-muenchen.de

Lukas Burk

- 2nd year PhD
- 50/50 at
 - “Statistical Learning and Data Science” (Bischl)
 - Leibniz Institute for Prevention Research and Epidemiology - BIPS GmbH
- >6 years of R stuff
 - Handful of packages
 - Lots of analyses
- [jemus42](#) on GitHub
- [@lukas.burk](#) on Mattermost
- Lukas.Burk@stat.uni-muenchen.de

Julia Niebisch

- 2nd year PhD
 - "Statistical Learning and Data Science"
(Bischl)
 - [3Julia3](#) auf GitHub
 - julia.niebisch@stat.uni-muenchen.de
 - @julia.niebisch on mattermost
-
- *Einführung in die statistische Software im SS20*
 - Hoping to teach you some R!

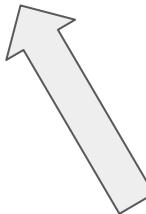
Sven Lorenz

- Currently doing my master thesis
- [svenlorenz](#) on GitHub and mattermost
- Here mostly to answer your questions
- Will have a weekly recurring Zoom Session to answer questions regarding the course, and **homework tasks**
- For technical Issues or more pressing matters, always feel free to get in touch via mattermost!

Programming in R

Why are we here?

Programming in R



Shouldn't need explanation

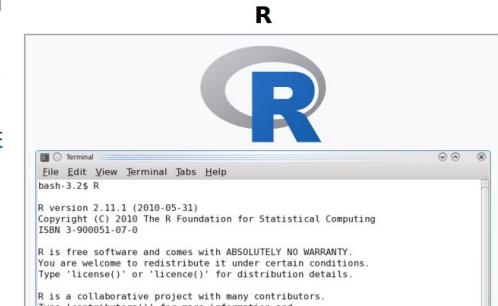
R (programming language)

From Wikipedia, the free encyclopedia

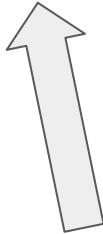
R is a programming language and free software environment for statistical computing and graphics supported by the R Foundation for Statistical Computing.^[6] The R language is widely used among statisticians and data miners for developing statistical software^[7] and data analysis.^[8] Polls, data mining surveys, and studies of scholarly literature databases show substantial increases in popularity;^[9] as of February 2020, R ranks 13th in the TIOBE index, a measure of popularity of programming languages.^[10]

A GNU package,^[11] the official R software environment is written primarily in C, Fortran, and R itself^[12] (thus, it is partially self-hosting) and is freely available under the GNU General Public License. Pre-compiled executables are provided for various operating

... etc.



Programming in R



We will spend some time on this!

Programming in R: Programming is Useful

- Programming is about letting something else do the work for you
- Most of (scientific / industry) work will have *some* degree of programming
- Your course work *will* get easier if you know programming

Programming in R: Programming is Valuable



Amazon Data Scientist Salary

The average Amazon Data Scientist earns \$164,114 annually, which includes a base salary of \$125,296 with a \$38,818 bonus. This total compensation is \$35,131 more than the US average for a Data Scientist. Data Scientist salaries at Amazon can range from \$110,000 - \$217,000 with equity ranging from 0-150K+.

The Engineering Department at Amazon earns \$14,890 more on average than the HR Department. Comparably data has a total of 21 salary records from Amazon Data Scientists.

Last updated 3 days ago.



(To be fair, it is harder to earn this kind of money in Germany)

Programming in R: Programming is Valuable

The image shows a screenshot of the Amazon Jobs website. At the top, there is a dark header bar with the "amazon jobs" logo on the left, a search bar in the center containing the placeholder "Search for jobs by title or keyword", and a "Location" input field on the right. Below the header, the main content area has a light gray background. On the left side of this area, the job title "Data Scientist" is displayed in large, bold, dark blue text. Underneath the title, the text "Job ID: 991758 | Amazon.com Services LLC" is shown in a smaller, gray font. To the right of the job title, there is a large, semi-transparent gray arrow pointing from the bottom-left towards the center. In the bottom-right corner of the main content area, there is a white rectangular callout box containing the "Key Responsibilities" section and a bulleted list of responsibilities.

Key Responsibilities:

- Implement statistical methods to solve specific business problems utilizing code (**Python, R, Scala, etc.**).
- Improve upon existing methodologies by developing new data sources, testing model enhancements, and fine-tuning model parameters.
- Directly contribute to the design and development of automated forecasting systems.
- Build customer-facing reporting tools to provide insights and metrics which track forecast performance and explain variance.
- Collaborate with researchers, software developers, and business leaders to define product requirements, provide analytical

DESCRIPTION
Where will A quintuple In product sele the Supply C

Programming in R: Programming is Valuable

 PwC
Data Scientist (w/m/d)

📍 Berlin, Düsseldorf, Frankfurt a. M., Hamburg, München, Stuttgart 🏢 Feste Anstellung ⏳ Vollzeit
🕒 Erschienen: vor 1 Woche

Je 

Dein Profil

Du hast erfolgreich dein Studium der Informatik / Mathematik / Physik oder ein vergleichbares naturwissenschaftliches Studium abgeschlossen und verfügst über Knowhow in der Datenanalyse sowie über grundfundierte Statistik- Mathematik Kenntnisse. Das Verstehen von betriebswirtschaftlichen Grundlagen unterstützt deine technischen Fertigkeiten.

Du verfügst über gute Kenntnisse in mindestens einer Data-Science-nahen Programmierungsumgebung wie R, Python, Matlab (oder ähnliche) sowie Erfahrung relationalen Datenbanken und NoSQL-Datenbanken. Die Möglichkeiten des Apache Hadoop Frameworks verstehst du und kannst sie aktiv einsetzen.

Programming in R: Programming Courses are Valuable

Programming in R: Programming Courses are Valuable

The screenshot shows the App Academy website's course selection page. At the top, there are navigation links for "On Campus", "Online", and "Enterprise". A banner at the top of the main content area states, "In response to COVID-19, all of our p". Below this, three course options are listed:

Free Plan	Mentorship Plan	Software Engineering Track: Online
Identical curriculum to the in-person full-time course	Free plan plus App Academy instructors on call every weekday	Full-time online (Mon-Fri), w/ live lecture, pair programming, staff help, career services and
\$0	\$29.99/month	\$0 upfront, 15% of salary for three years (\$31k max) OR other payment options available (see FAQ)

Programming in R: Programming Courses are Valuable

The screenshot shows the App Academy website. At the top, there's a navigation bar with the logo 'a/A App Academy' and dropdown menus for 'On Campus', 'Online', and 'Enterprise'. A black banner at the top of the main content area states 'In response to COVID-19, all of our p...'. Below this, three course options are displayed: 'Free Plan', 'Mentorship Plan', and 'Software Engineering'. The 'Free Plan' section features large text: 'Learn programming on your own and save up to USD 31'000!'. It shows a price of '\$0' and '\$29.99/month'. The 'Software Engineering' section has a price of '\$0 upfront, 15% of salary for three years (\$31k max) OR other payment options available (see FAQ)'.

In response to COVID-19, all of our p...

Free Plan Mentorship Plan Software Engineering

Learn programming on your own
and save up to USD 31'000!

Plan	Price
Free Plan	\$0
Mentorship Plan	\$29.99/month
Software Engineering	\$0 upfront, 15% of salary for three years (\$31k max) OR other payment options available (see FAQ)

Programming in R: Programming Courses are Valuable

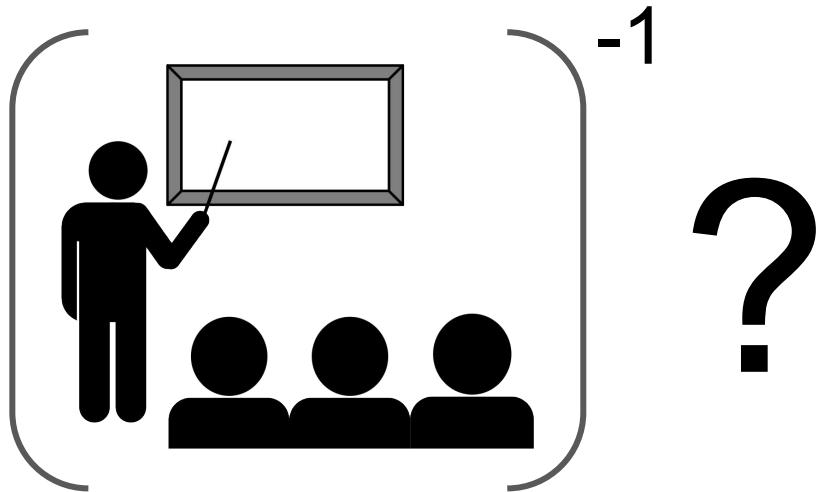
Of course that is not a fair comparison, but....

- There are lots of free courses online, we try to give you more than that
- Make use of this opportunity now or regret it later
- You should not be coming here for the credit, you are coming here to learn programming

Structure of this Course

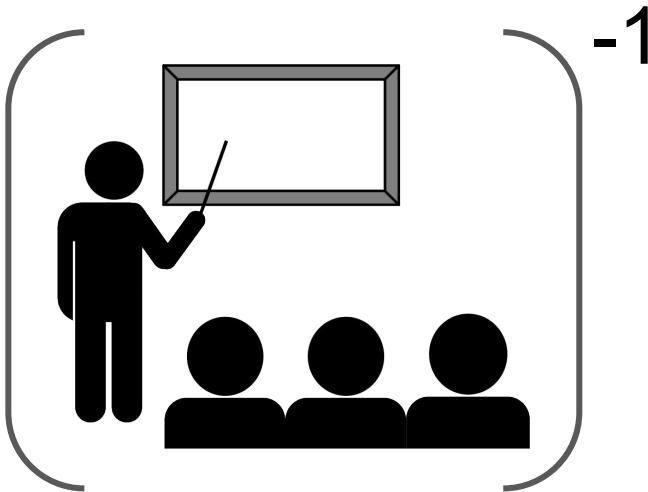
Structure of this Course

- This is an online course by ~~necessity~~ choice.
- "Inverted Classroom"



Structure of this Course

- This is an online course by choice.
- "Inverted Classroom"
- Lectures and Q&A Sessions every week
- We stay in contact with you!
 - Moodle Forum
 - Mattermost Chat
 - Noah's Office Hours



Structure of this Course

- Please give us feedback!
 - Anonymous
 - Helps us improve the course



Home > My courses > s20_progr

Ankündigungen

Zentrales Forum

Feedback Lecture 2020-04-21

Wir geben unser Bestes um euch R-Programmierung und Softwareentwicklung beizubringen. Bitte helfen Sie uns, den Kurs zu verbessern und sagen Sie uns was Ihnen gefallen oder nicht gefallen hat, und was Sie verbessern wuerden.

Diese Umfrage ist anonym, wir koennen nicht sehen wer welche Antworten gegeben hat. Wir sind auch fuer direktes Feedback ueber Mattermost oder Email offen.

L
A
2
M
C
C

Structure of this Course

- Please give us feedback!
 - Anonymous
 - Helps us improve the course
- Please talk to us!
 - Forum



Home > My courses > s20_progr

L
A
2
M
C
C

- Ankündigungen
- Zentrales Forum
- Feedback Lecture 2020-04-21

Wir geben unser Bestes um euch R-Programmierung und Softwareentwicklung beizubringen. Bitte helfen Sie uns, den Kurs zu verbessern und sagen Sie uns was Ihnen gefallen oder nicht gefallen hat, und was Sie verbessern wuerden.

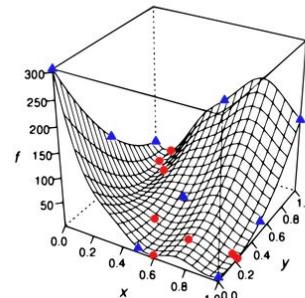
Diese Umfrage ist anonym, wir koennen nicht sehen wer welche Antworten gegeben hat. Wir sind auch fuer direktes Feedback ueber Mattermost oder Email offen.

Structure of this Course

- Please give us feedback!
 - Anonymous
 - Helps us improve the course
- Please talk to us!
 - Forum
 - Mattermost



- **Mattermost:** Wir haben einer Kommunikation der Teilnehmer untereinander ermöglicht. Bitte Sollten Sie noch keinen Zugriff auf dieses Invite-Link. Sie müssen



Compstat LMU Headquarters

Fitness-Maximizers, not Adaptation-Executers.

Sign in

[I forgot my password.](#)

Structure of this Course

- Please give us feedback!
 - Anonymous
 - Helps us improve the course
- Please talk to us!
 - Forum
 - Mattermost
 - Office Hours



Support English ▾

Click **Open link** on the dialog shown by your browser
If you don't see a dialog, click **Launch Meeting** below

Launch Meeting

Don't have Zoom Client installed? [Download Now](#)

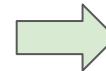
Having issues with Zoom Client? [Join from Your Browser](#)

Structure of this Course

- Experience shows that students
 - do not read book chapters and
 - do not watch lecture videos
- But apparently they read slides...
- And they do homework if we incentivise it

Structure of this Course

You are here



- We publish a **Homework Task** on **Mondays (except 1st homework)**
- You may try to solve the Task
- You meet us on **Thursday during the lecture timeslot**
 - Ask questions
 - Listen to us tell you some relevant things about programming
 - Come even if you don't understand anything, we will try to help
 - Come even if you understand everything ;-)
- Collect questions in our **Etherpad**
- Meet **Sven Friday during his Office Hours**
- Hand in your solution no later than **end of Sunday after the lecture timeslot**
- Automatic checks pass

Assignment I



Assignment II



Lecture	Date
1	2022-04-28
2	2022-05-05
3	2022-05-12
4	2022-05-19
5	2022-05-26
6	2022-06-02
7	2022-06-09
8	2022-06-16
9	2022-06-23
10	2022-06-30
11	2022-07-07
12	2022-07-14
13	2022-07-21
14	2022-07-28

Structure of this Course

Homework assignments with "Automatic checks pass"

- This is even easier than it sounds, because you get immediate feedback. You will always know if your solution is correct.
- Team work: Honour system
 - We *encourage* you to work together to make sure everyone understands what is going on.
 - We *discourage* you from blindly copying each other's work. You *will* fail the exam if you can not solve the homework tasks.

Structure of this Course

Graded Assignments:

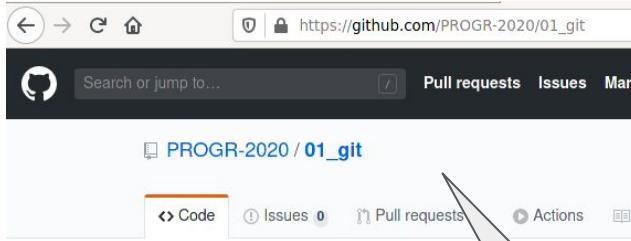
- 2 assignments during the semester (20 points each)
- 1 larger assignment at the end of the semster (80 points)
- Final grade will be calculated from total number of points over all assignments
- Homework assignments mimic the graded assignments
- A portion of the graded assignment will also have tasks with automated checks (but not all!!)

Structure of this Course

Please do the homework

- Please, please, please
- Some students failed before because they were not familiar with the homework grading system

Anatomy of a Homework Task



Homework Task I -- git and GitHub

Github

Link to the Slides

Further Material

Task Description

README.md	define the tasks	19 minutes ago
sier.png	define the tasks	19 minutes ago
sierpinski.Rmd	framework setup	3 hours ago

[README.md](#)

Programming in R

Homework Task I -- git and GitHub

This is your first homework task. Since it will not be graded there is no deadline. Nevertheless, we strongly encourage you to try out this task, both to get to know the homework task format and to get to know git. All the following tasks will assume that you know how to use `git` and `GitHub`.

Information Material

- [Slides](#)
- Recommended resources
 - [Software Carpentry Git Course](#): A very good and thorough first `git` course that teaches how to use `git` from the command line. It is strongly recommended that you know the command line `git`, since you will understand what is going on under the hood even when using other tools. Expected reading time: 3 hours.
 - [Happy Git with R](#): A course of `git`, `GitHub`, and `RStudio`. This will teach you how to use all your tools together efficiently.
- Additional resources
 - [Git Immersion](#): Similar to the "[Software Carpentry Git Course](#)" but with less explanation.
 - [Git and GitHub](#) by Hadley Wickham: A quick intro on how to use `git`, `GitHub`, and `RStudio` that is shorter than "[Happy Git with R](#)" above and may be for you if you are a quick learner.
 - [GitHub Introduction](#): Get to know the `GitHub` web interface
 - [Cheat Sheets](#) are an excellent way to keep a reference around and to look up commands you may have forgot. The big `git` companies all offer one (and `GitHub` isn't necessarily the best): [GitHub](#), [GitLab](#), [Atlassian](#). Look at all of them and see if you find them useful.

The Task

This is a project that was set up by different characters, who submitted their work to different branches. However, in the current state their individual contributions are in different branches and have not yet been brought together. Your task is to find out who

Anatomy of a Homework Task

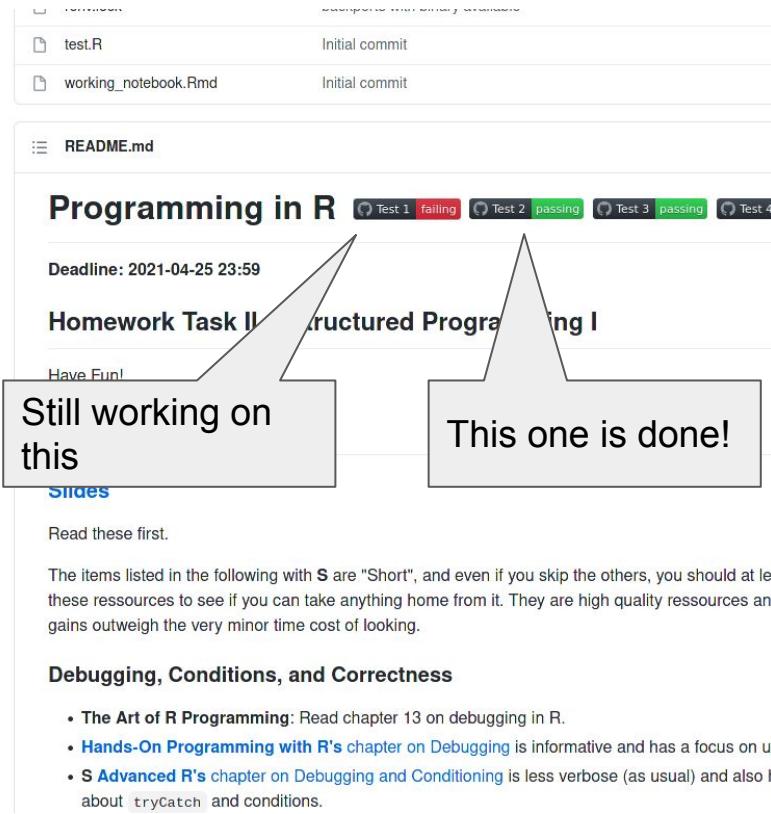
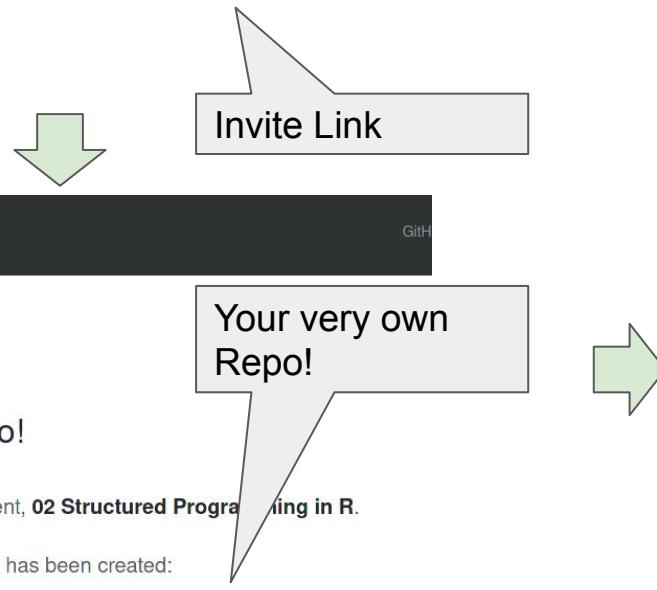


Erste Hausaufgabe

by Martin Binder - Monday, 27 April 2020, 10:49 AM

Liebe Teilnehmer,

Um die Aufgabe zu bearbeiten, geht auf [https://classroom.github.com/a/\[REDACTED\]](https://classroom.github.com/a/[REDACTED]) und klic



What We Expect You to Know

git commands to know:

- git init -- create a new (empty) repository
- git clone -- create a repository linked to another (online) repo
- git status -- see if files have changed
- git checkout -- check out files or branches
 - git checkout -b "new_branch_name" -- create new branch
- git branch -- find out on what branch you're on
- git merge -- take changes from another branch into the current one
 - know how to handle merge conflicts!
- git diff -- see what has changed (default: since last commit)
- git add -- add changes to the staging area to be committed
- git commit -- commit changes in the staging area
 - git commit -a -- commit all changes
- git fetch -- get changes from remote repo and DON'T merge
- git pull -- get changes from remote repo and DO merge
- git push -- push changes to remote repo
- The .gitignore file -- let git ignore certain files / file types

The "What We
Expect You to Know"
Slide

What We Expect You to Know

git commands to know:

- git init -- create a new (empty) repository
- git clone -- create a repository linked to another (online) repo
- git status -- see if files have changed
- git checkout -- check out files or branches
 - git checkout -b "new_branch_name" -- create new branch
- git branch -- find out on what branch you are
- git merge -- take changes from another branch into the current one
 - know how to handle merge conflicts!
- git diff -- see what has changed (default: since last commit)
- git add -- add changes to the staging area to be committed
- git commit -- commit changes in the staging area
 - git commit -a -- commit all changes
- git fetch -- get changes from remote repo and DON'T merge
- git pull -- get changes from remote repo and DO merge
- git push -- push changes to remote repo
- The .gitignore file -- let git ignore certain files / file types

Anatomy of a Homework Task

- Slides for you to read
- Lots of links to read further
- Task description of what you are supposed to do
- Code repository that you build up on
- Correctness is checked automatically (except for first homework)

The screenshot shows a GitHub repository interface. At the top, there's a file tree with 'test.R' and 'working_notebook.Rmd' both listed as 'Initial commit'. Below the file tree is the 'README.md' file, which contains the following content:

```
Programming in R
Test 1 failing | Test 2 passing | Test 3 passing | Test 4 passing

Deadline: 2021-04-25 23:59

Homework Task II -- Structured Programming I

Have Fun!

Information Material

Slides

Read these first.

The items listed in the following with S are "Short", and even if you skip the others, you should at least look at these resources to see if you can take anything home from it. They are high quality resources and the potential gains outweigh the very minor time cost of looking.

Debugging, Conditions, and Correctness



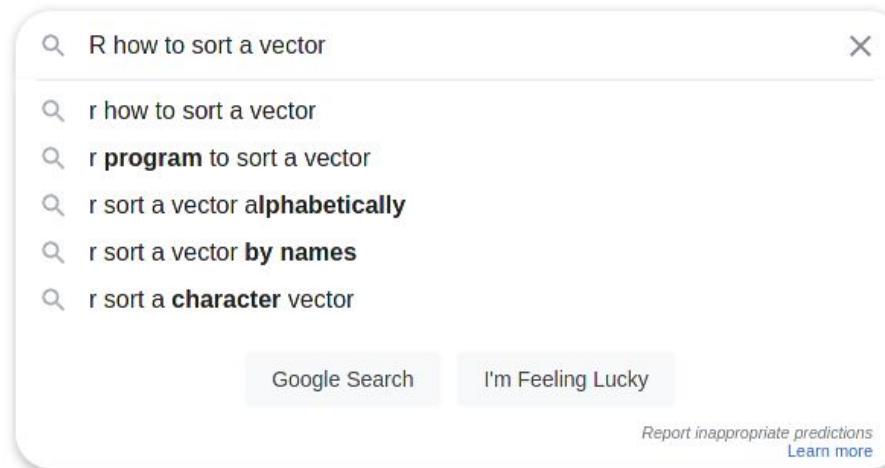
- The Art of R Programming: Read chapter 13 on debugging in R.
- Hands-On Programming with R's chapter on Debugging is informative and has a focus on using R's built-in tools.
- S Advanced R's chapter on Debugging and Conditioning is less verbose (as usual) and also has information about tryCatch and conditions.

```

How to Solve the Homework

- Read the slides and source material
- Pay attention to the questions
- Try things out and experiment
- google.com

How to Solve the Homework



How to Solve the Homework

[https://stackoverflow.com › questions › how-do-i-sort-a...](https://stackoverflow.com/questions/1549730/how-do-i-sort-a-vector-based-on-values-of-another) ▾

How do I sort one vector based on values of another - Stack ...

15 Oct 2009 — How do I sort one **vector** based on values of another · sorting r. I have a **vector** x, that I would like to sort based on the **order** of values in **vector** y ...

[7 answers](#) · 196 votes: what about this one x[order(match(x,y))]

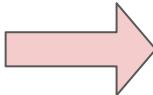
[How to sort a vector in R, keeping names and values ...](#) 1 answer 16 Oct 2015

[How to sort a character **vector** according to a specific ...](#) 2 answers 10 Jun 2013

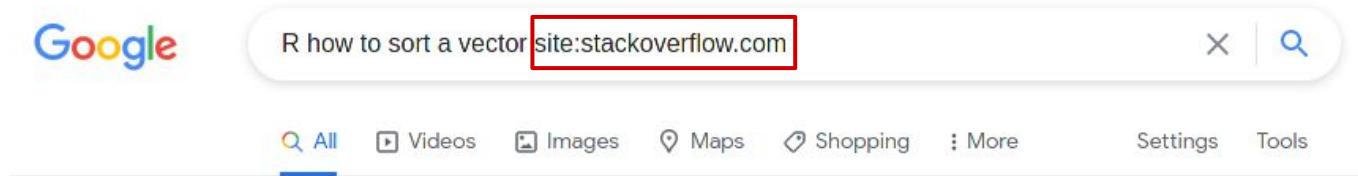
[R: Sort a data frame based on the **order** of a **vector** ...](#) 3 answers 12 Apr 2016

[How to make a date **vector** be ordered so it starts with ...](#) 1 answer 8 May 2016

[More results from stackoverflow.com](#)



How to Solve the Homework



A screenshot of a Google search results page. The search query is "R how to sort a vector site:stackoverflow.com". The "site:" operator is highlighted with a red box. The search results show two main links from Stack Overflow:

- How do I sort one vector based on values of another - Stack ...**
15 Oct 2009 — How do I sort one vector based on values of another · sorting r. I have a vector x, that I would like to sort based on the order of values in vector y ...
7 answers · 196 votes: what about this one x[order(match(x,y))]
Block stackoverflow.com
- Sort one vector based on other - Stack Overflow**
11 Apr 2017 — Sort one vector based on other · r sorting. I have 2 vectors, x <- c (1,4 ...
2 answers · Top answer: We can use that as index i1 <- sort(x, index.return=TRUE)\$ix Or with ...
Block stackoverflow.com

<https://stackoverflow.com/questions/sort-one-vector...> ▾

Sort one vector based on other - Stack Overflow

11 Apr 2017 — Sort one vector based on other · r sorting. I have 2 vectors, x <- c (1,4 ...

2 answers · Top answer: We can use that as index i1 <- sort(x, index.return=TRUE)\$ix Or with ...

Block stackoverflow.com

<https://stackoverflow.com/questions/how-to-sort-a-v...> ▾

How to Solve the Homework

Stack Overflow About Products For Teams Search...

Home
PUBLIC
Questions
Tags
Users
FIND A JOB
Jobs
Companies
TEAMS
Stack Overflow for Teams – Collaborate and share knowledge with a private group.
Create a free Team
What is Teams?

How do I sort one vector based on values of another

Asked 11 years, 6 months ago Active 1 year, 11 months ago Viewed 76k times

I have a vector x, that I would like to sort based on the order of values in vector y. The two vectors are not of the same length.

115

```
x <- c(2, 2, 3, 4, 1, 4, 4, 3, 3)
y <- c(4, 2, 1, 3)
```

40

The expected result would be:

```
[1] 4 4 4 2 2 1 3 3 3
```

sorting r

Share Improve this question Follow edited Oct 15 '09 at 18:40 asked Oct 14 '09 at 19:41
learner
5,629 ● 4 ● 25 ● 23

Add a comment

7 Answers

Active Oldest Votes

what about this one

196

```
x[order(match(x,y))]
```

Share Improve this answer Follow answered Jan 22 '10 at 12:13
George Donatas
27.4k ● 17 ● 102 ● 140

Etherpad

The screenshot shows a web-based Etherpad editor. At the top, there's a toolbar with icons for back, forward, refresh, and a search bar containing the URL <https://etherpad.wikimedia.org/p/ProgR2021>. Below the toolbar is a set of rich-text editing icons: bold (B), italic (I), underline (U), strikethrough (S), lists (three levels), code (C), and a link icon.

The main content area contains several text blocks, each preceded by a line number (1, 2, 3, 4, 5, 6, 7, 8, 9, 10) on the left:

- Line 1: **This is the LMU ProgR 2021 Etherpad for the week 2021-04-13 -- 2021-04-20**
- Line 2: Ask questions, write comments, etc.
- Line 3: (empty)
- Line 4: **Question of the Week:** What is your first pick Mario Kart character?
- Line 5: Martin: Yoshi
- Line 6: Noah: Dry Bones
- Line 7: (empty)
- Line 8: (empty)
- Line 9: **Questions, comments etc. about the course:**
- Line 10: (empty)

DataCamp

- We have a DataCamp Classroom over the time of this semester
 - you have full access to all DataCamp courses
 - Invite-Link on moodle
- You are not obliged to do anything on DataCamp to pass the course
 - it is voluntary
 - intended as an additional benefit of the course

We don't have unlimited accounts though, so please only subscribe if you really intend to use it - if you are not sure, use your free test account first!

Scope of this Course

Scope of this Course

- This is a **programming** course
 - Focus on **creating programs**
 - Not on data science / statistics (you won't learn about all the `mean()`s and `rnorm()`s here)
 - Not a survey of packages for specific tasks (you already had `tidy` + `ggplot` in StatSoft)
- Three "tracks"
 - Learn things about the R language: "**R**"
 - Get to know nice tools to use: "**Tools**"
 - Learn things about software development in general: "**Dev**"

Schedule

Lecture	Topic
KW 17	Introduction
KW 18	
KW 19	Structured Programming in R
KW 20	Debugging
KW 21	1st Assignment
KW 22	
KW 23	Tabular Data
KW 24	Feiertag
KW 25	Parallelization & Reproducibility
KW 26	2nd Assignment
KW 27	OOP: S3
KW 28	OOP: R6
KW 29	
KW 30	Software Engineering

Exam: ~ August 15th - September 11th
(Please respond to our survey!)

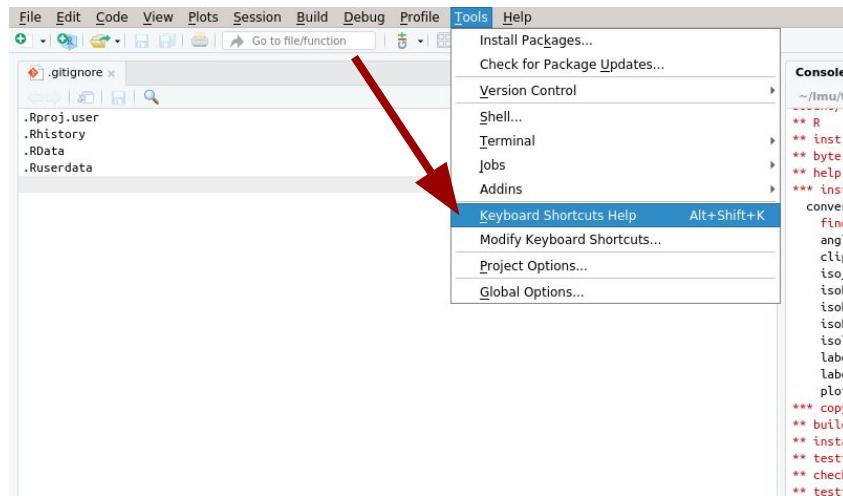
Your Setup

R / RStudio

- You should have R and RStudio installed
- If not already, you will necessarily get more familiar with RStudio as the course progresses
- A few things to keep in mind...

R / RStudio

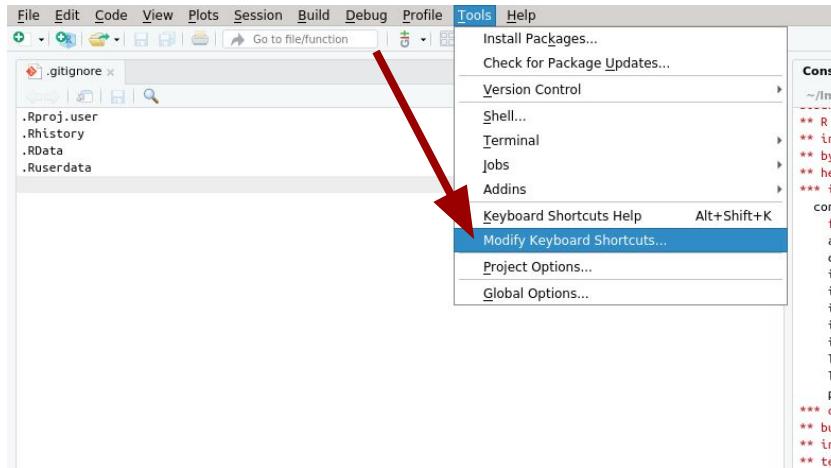
Learn the hotkeys!



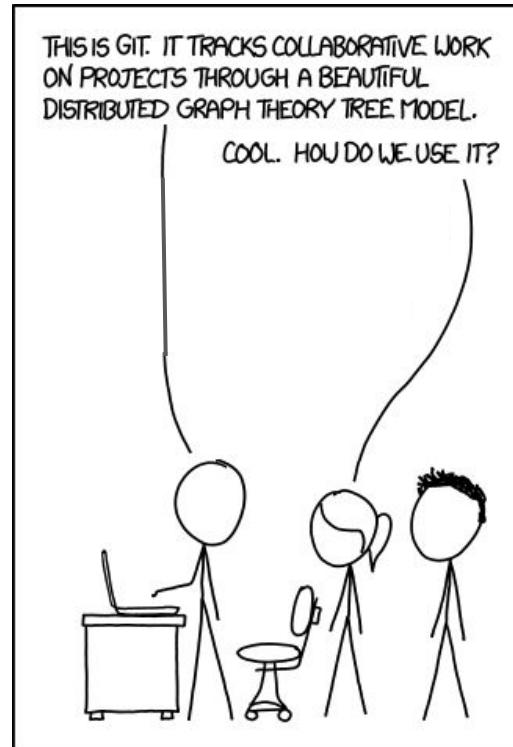
R / RStudio

You can adapt the hotkeys

(but this may make things hard when you switch work places)



Git

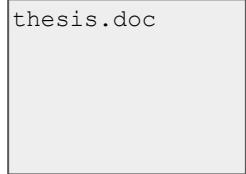


<https://xkcd.com/1597/>

Git -- "Version Control System" (VCS)

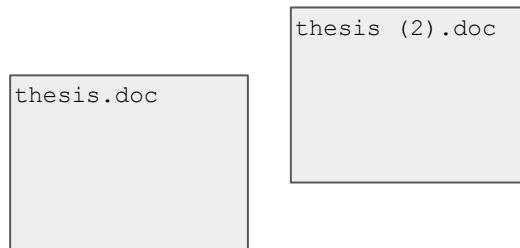
You probably know this:

thesis.doc



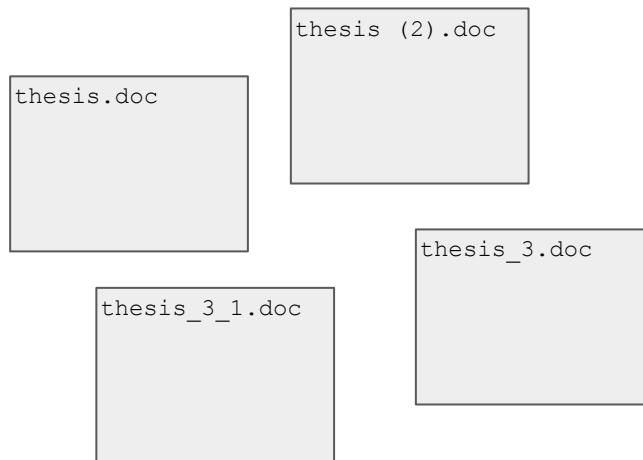
Git -- "Version Control System" (VCS)

You probably know this:



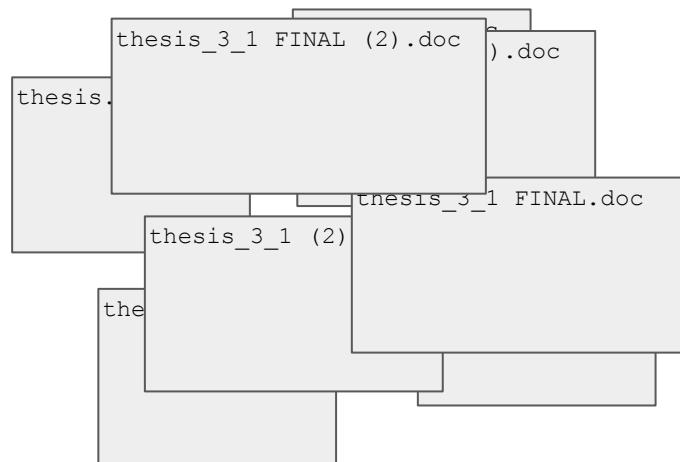
Git -- "Version Control System" (VCS)

You probably know this:



Git -- "Version Control System" (VCS)

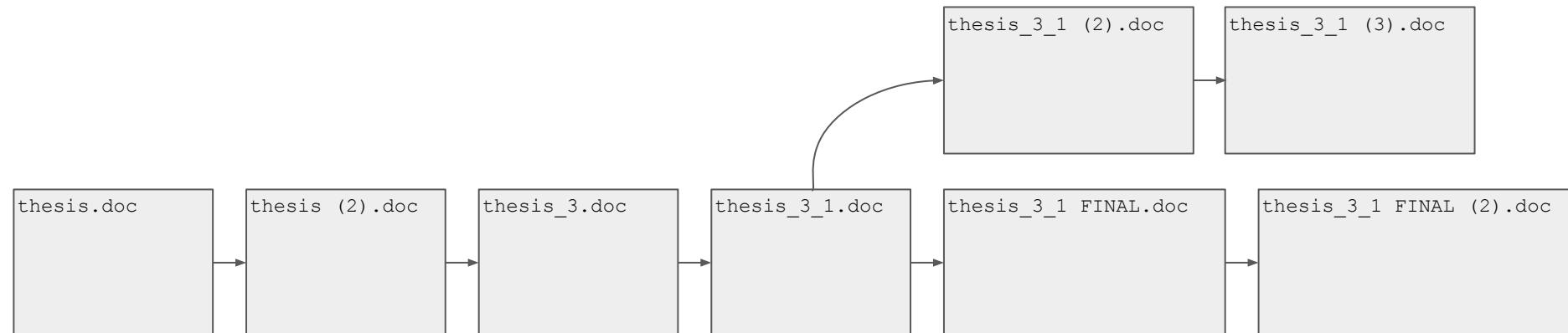
You probably know this:



Git -- "Version Control System" (VCS)

How about we organize things:

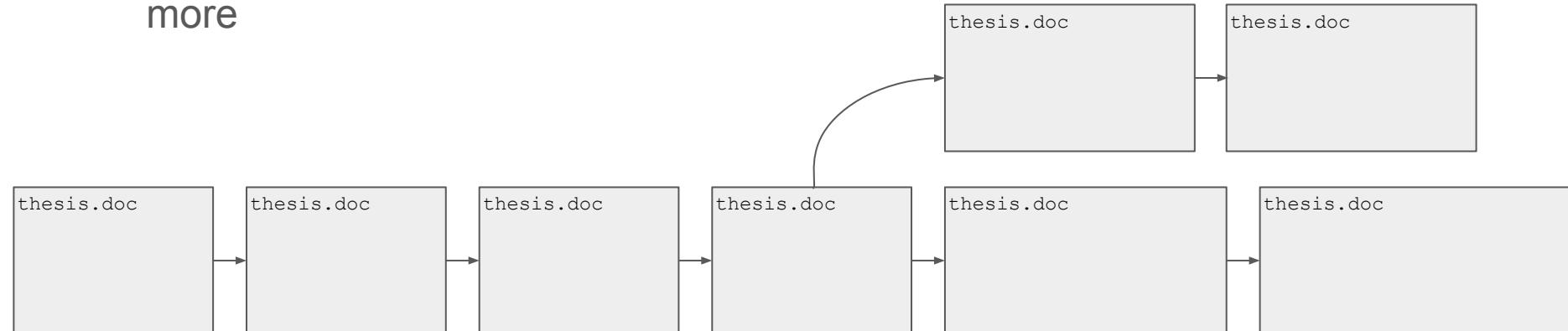
- Tree-like structure



Git -- "Version Control System" (VCS)

How about we organize things:

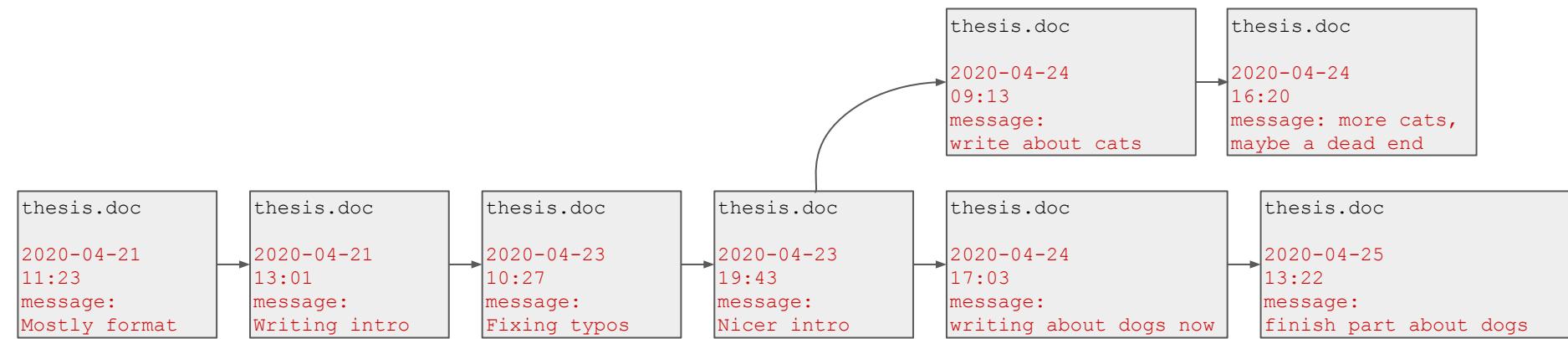
- We don't need different file names any more



Git -- "Version Control System" (VCS)

How about we organize things:

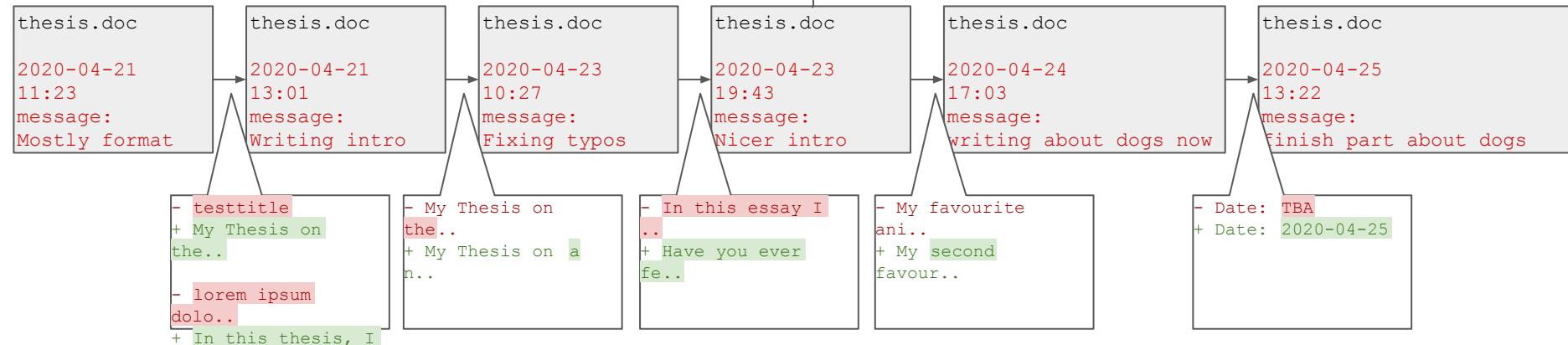
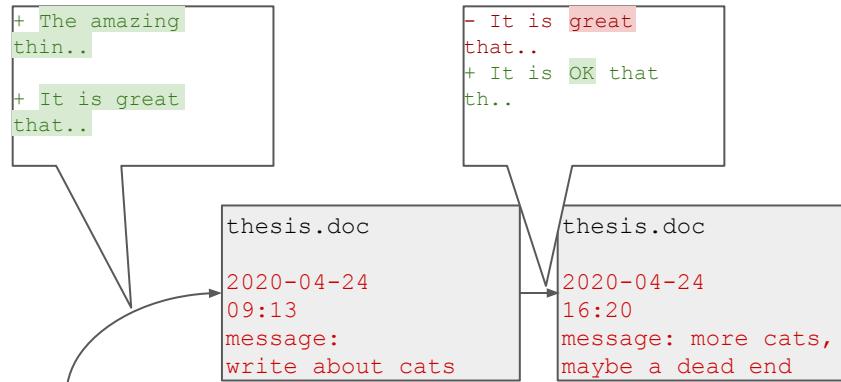
- Metadata



Git -- "Version Control System" (VCS)

How about we organize things:

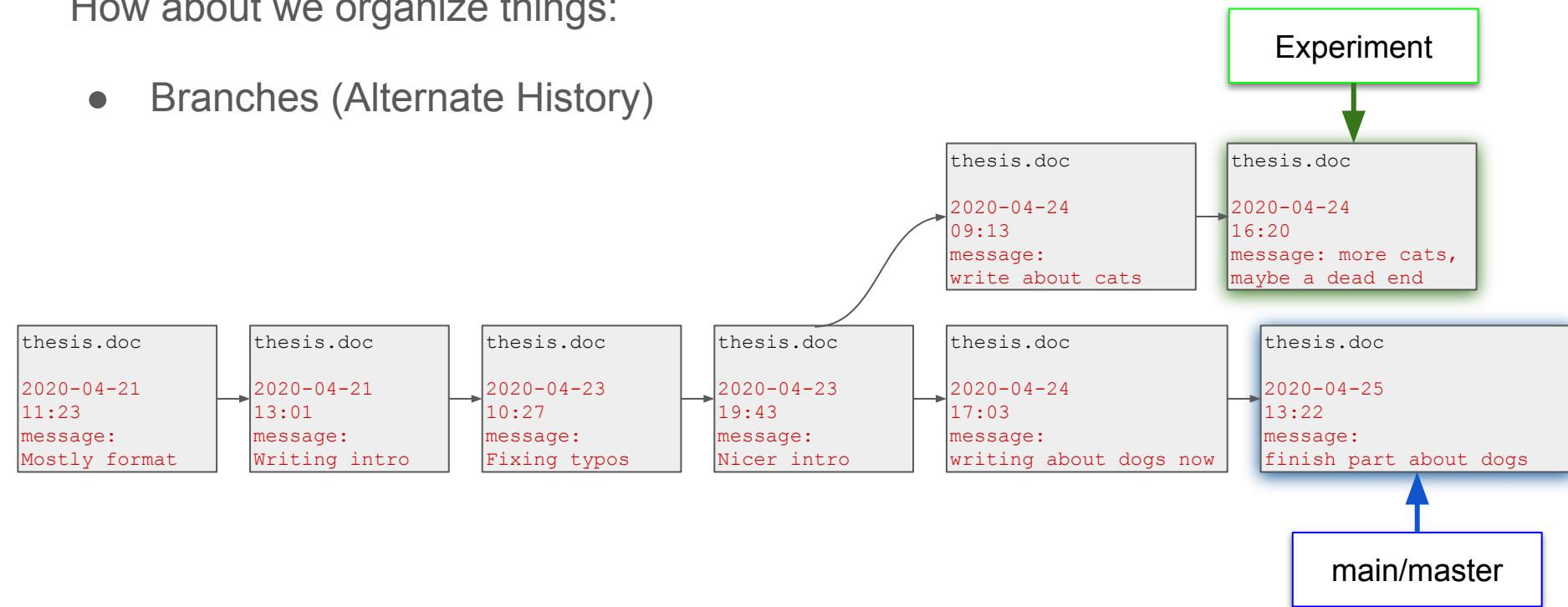
- Diffs



Git -- "Version Control System" (VCS)

How about we organize things:

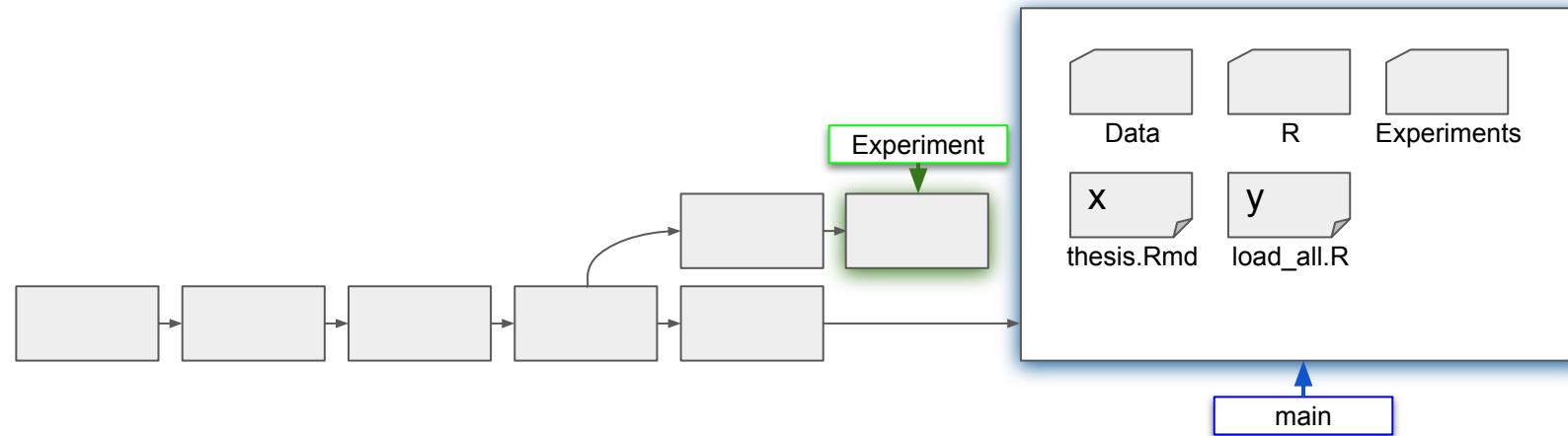
- Branches (Alternate History)



Git -- "Version Control System" (VCS)

How about we organize things:

- More than one File



Git -- "Version Control System" (VCS)

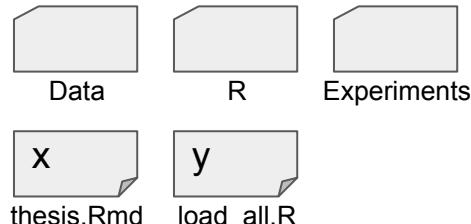
How about we organize things:

-

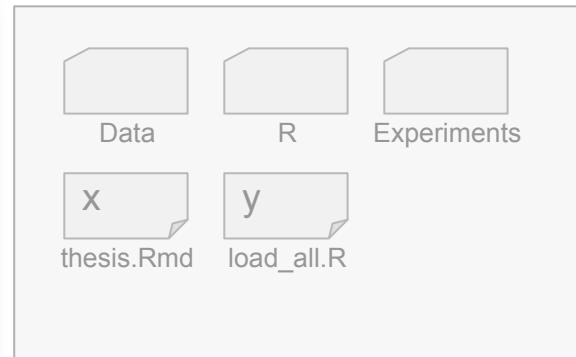
Abstractions

Git Repository

periment

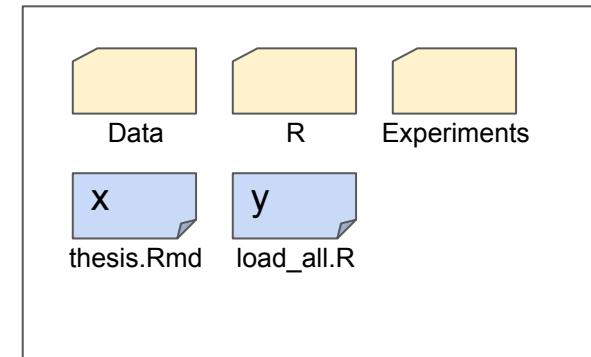


Staging Area



Your actual Folder

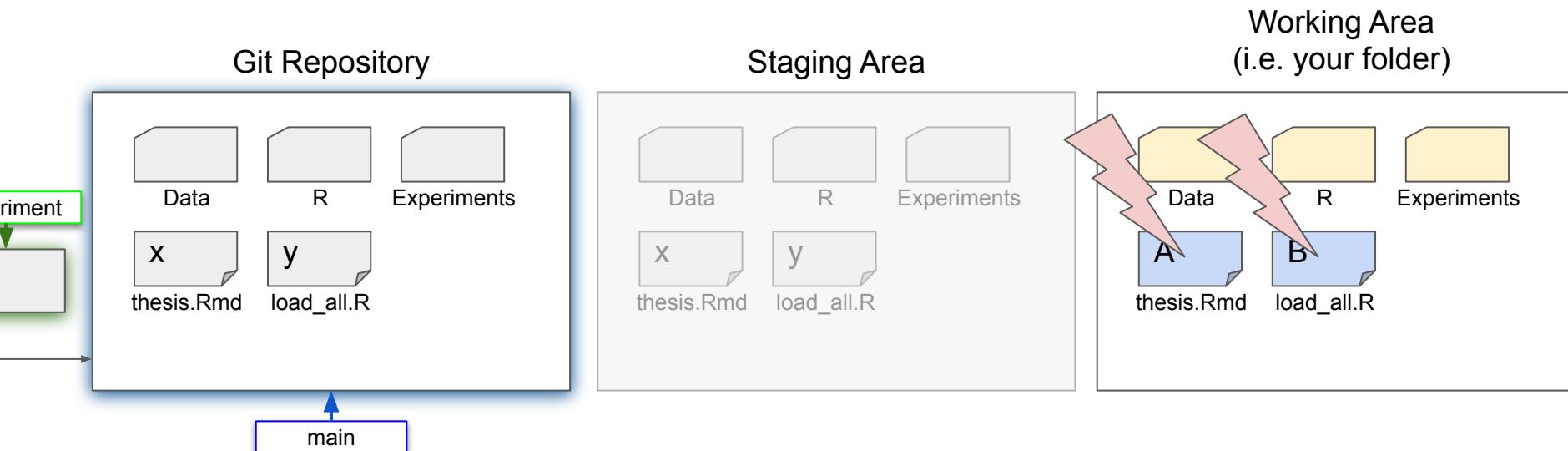
Working Area
(i.e. your folder)



Git -- "Version Control System" (VCS)

How about we organize things:

- Changing Files



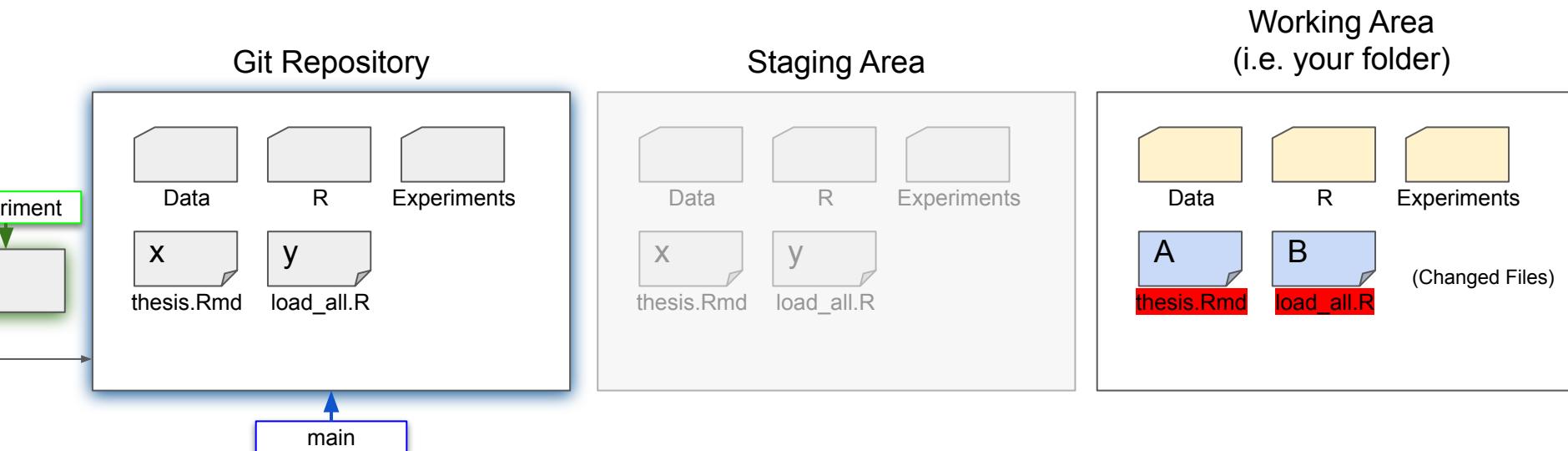
Git -- "Version Control System" (VCS)

How about we organize things:

- Changing Files: > git status

```
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will
  be committed)

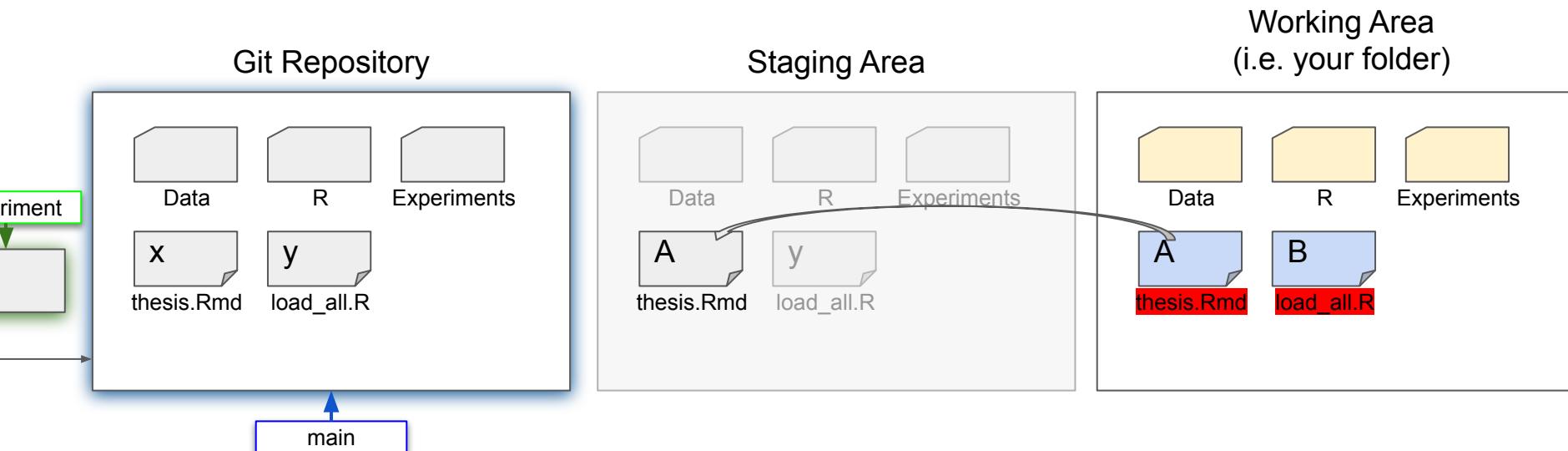
        modified:  load_all.R
        modified:  thesis.Rmd
```



Git -- "Version Control System" (VCS)

How about we organize things:

- "Staging" Files: > git add <filename>



Git -- "Version Control System"

How about we organize things:

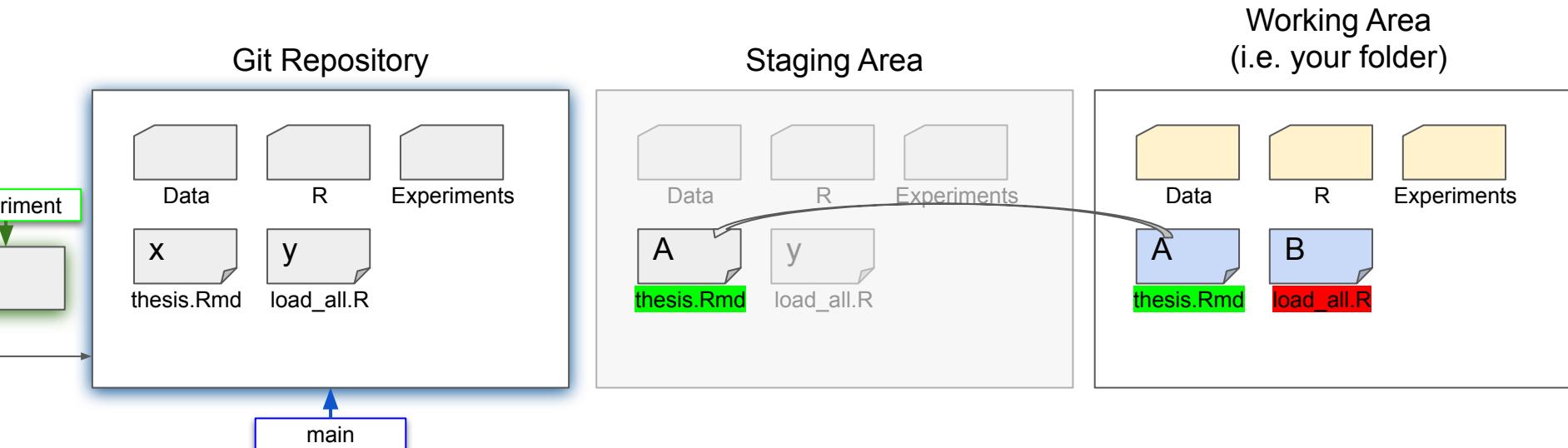
- "Staging" Files: > git status

```
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

    modified:  thesis.Rmd

Changes not staged for commit:
  (use "git add <file>..." to update what will
  be committed)
  (use "git checkout -- <file>..." to discard changes)

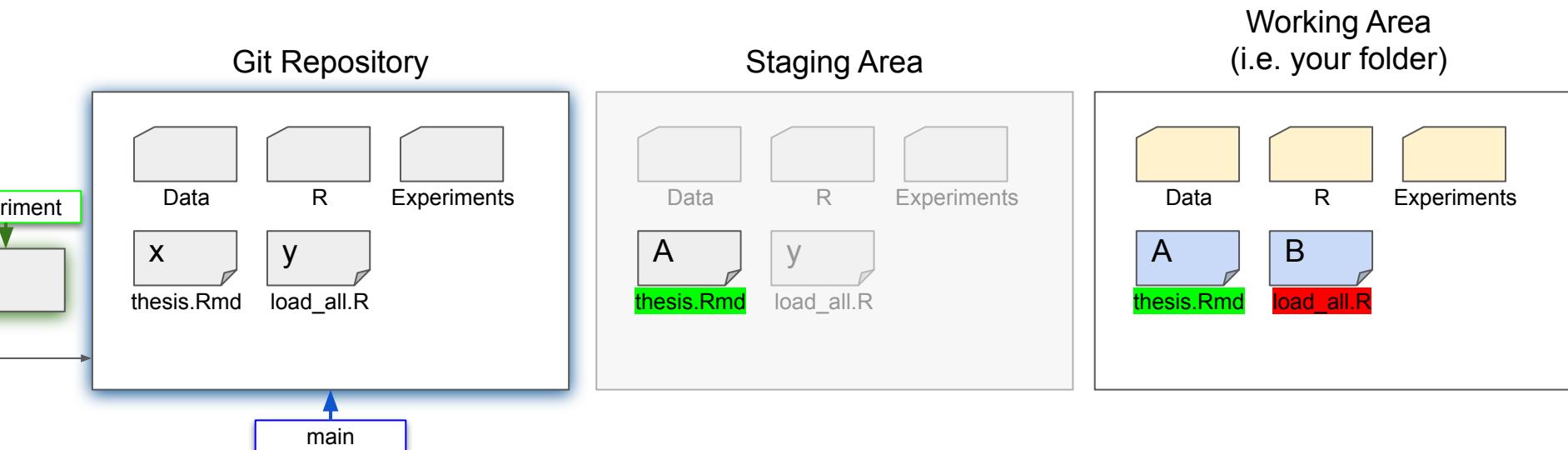
    modified:  load_all.R
```



Git -- "Version Control System" (VCS)

How about we organize things:

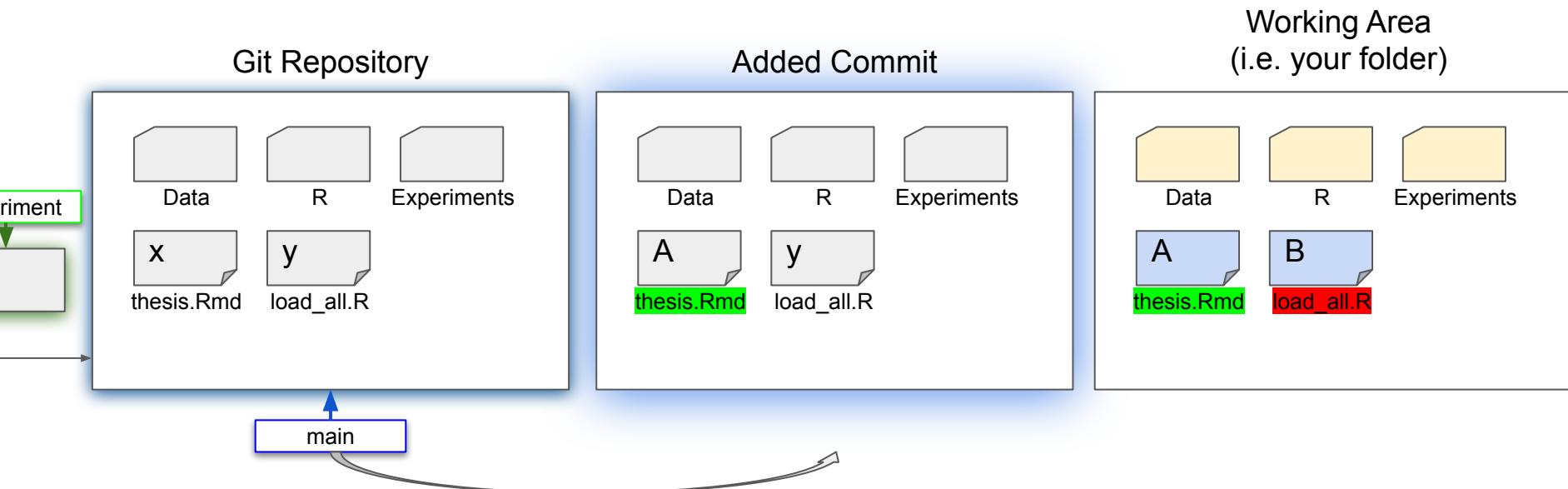
- "Commit" Files: > git commit



Git -- "Version Control System" (VCS)

How about we organize things:

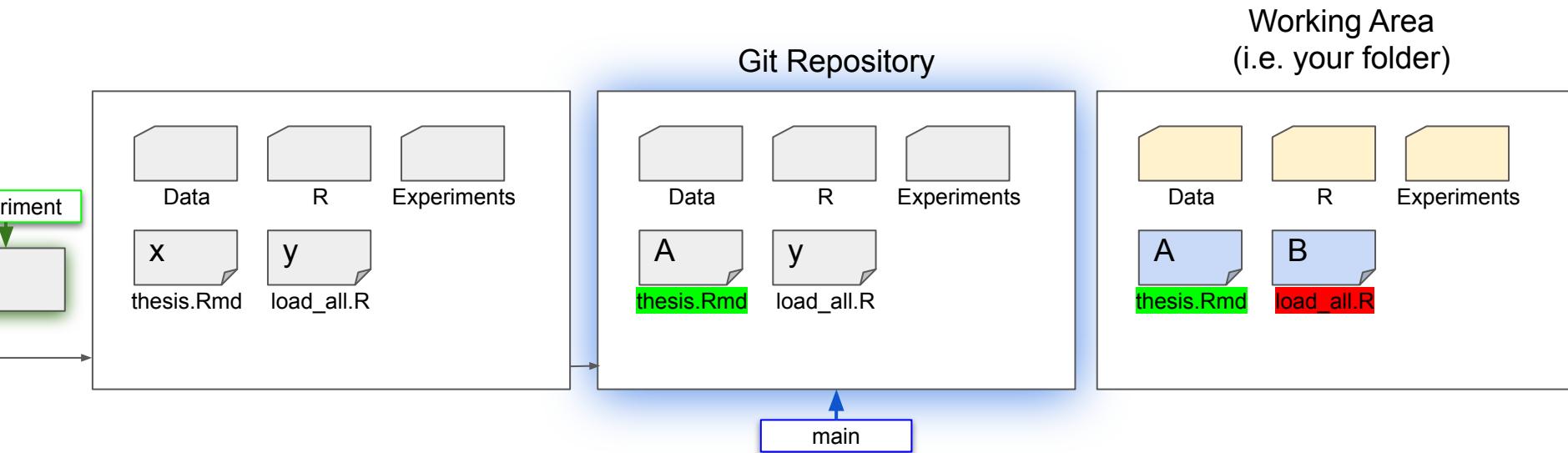
- "Commit" Files: > git commit



Git -- "Version Control System" (VCS)

How about we organize things:

- "Commit" Files: > git commit



Git -- "Version Control System" (VCS)

How about we organize things:

- Status now: > git status

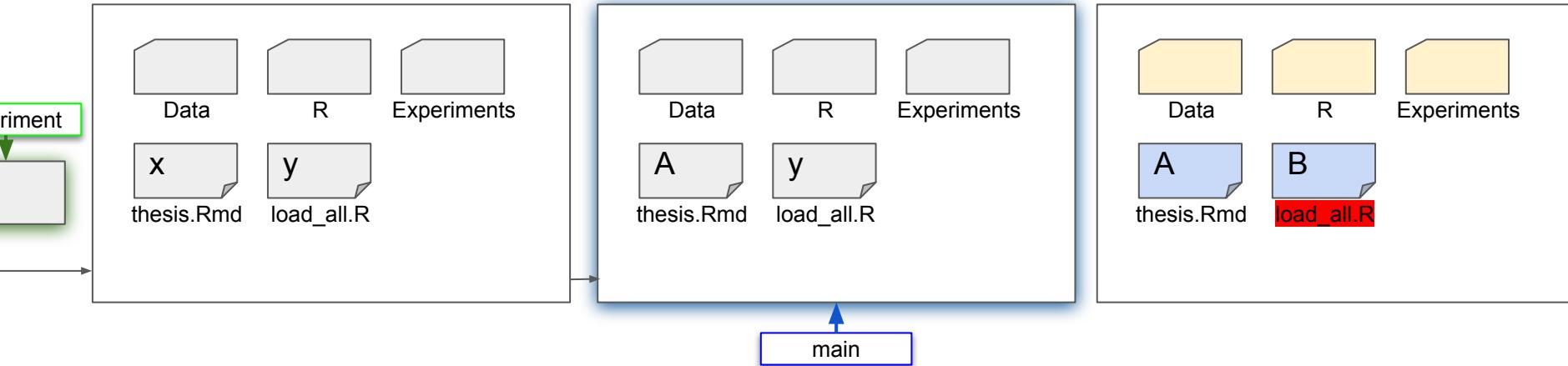


```
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will
  be committed)

        modified:   load_all.R
```

Working Area
(i.e. your folder)

Git Repository



Git -- "Version Control System" (VCS)

How about we organize things:

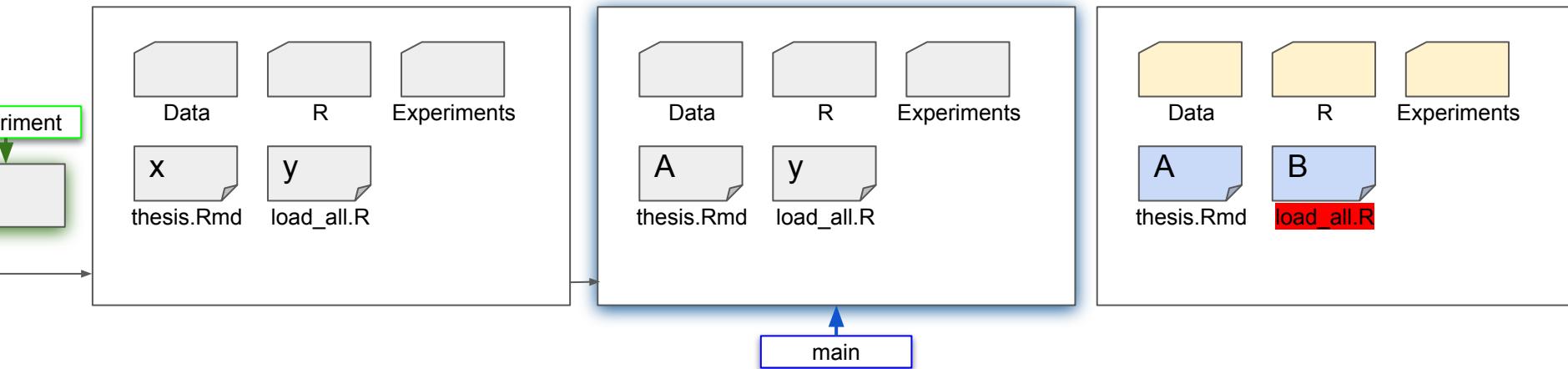
- Changed lines: > git diff



```
diff --git a/load_all.R b/load_all.R
index 975fbec..223b783 100644
--- a/load_all.R
+++ b/load_all.R
@@ -1 +1 @@
-y
+B
```

Working Area
(i.e. your folder)

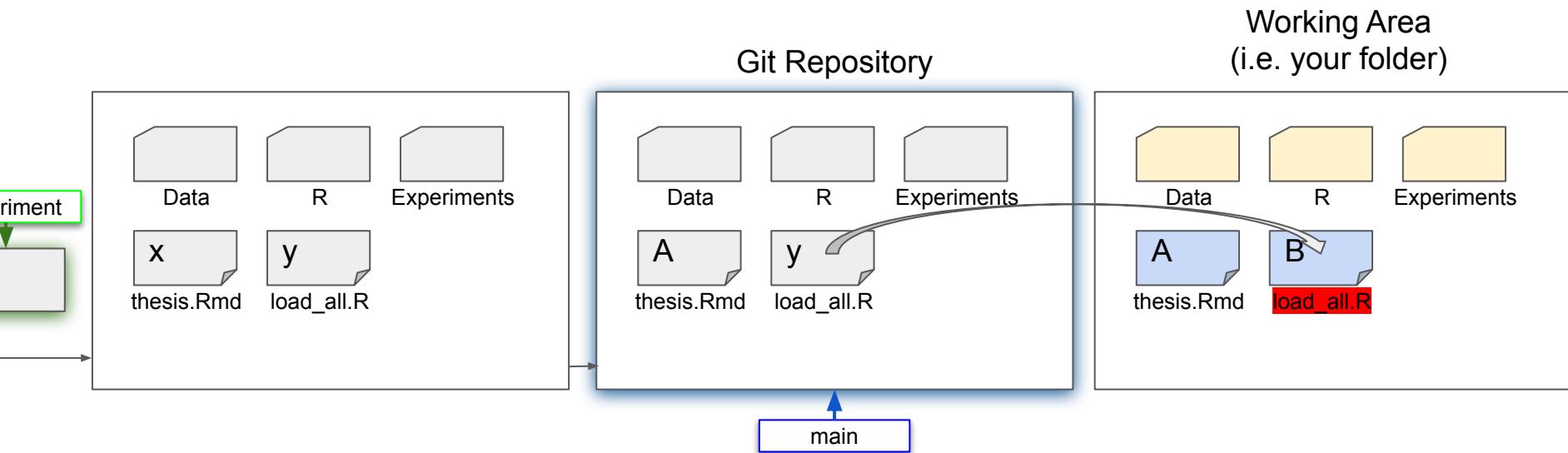
Git Repository



Git -- "Version Control System" (VCS)

How about we organize things:

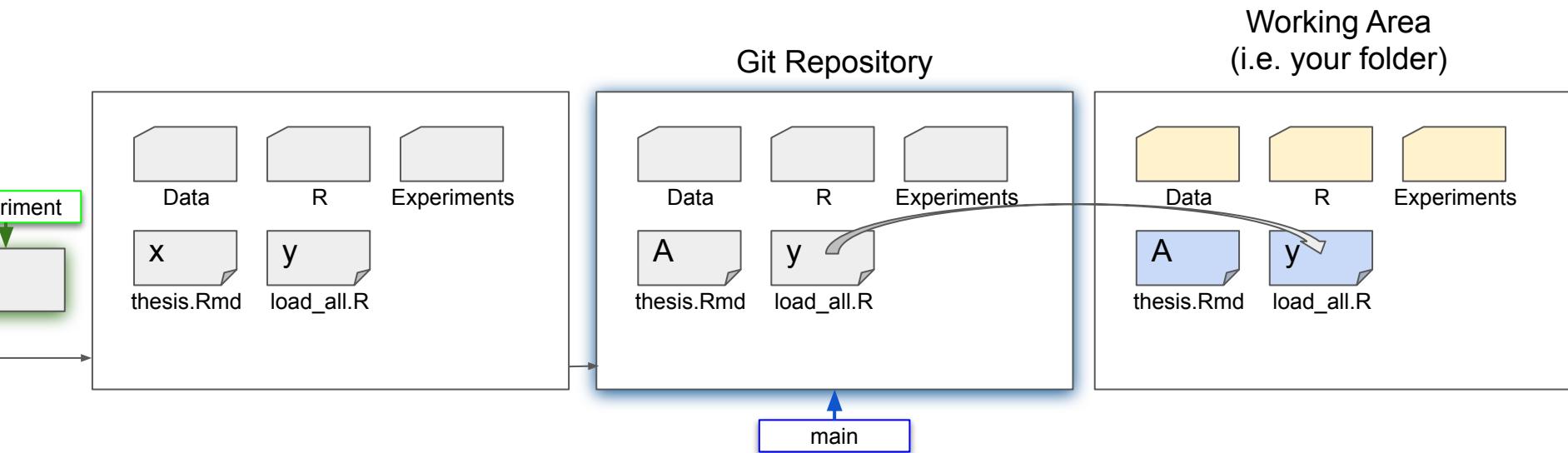
- "check out" Files: > git checkout load_all.R



Git -- "Version Control System" (VCS)

How about we organize things:

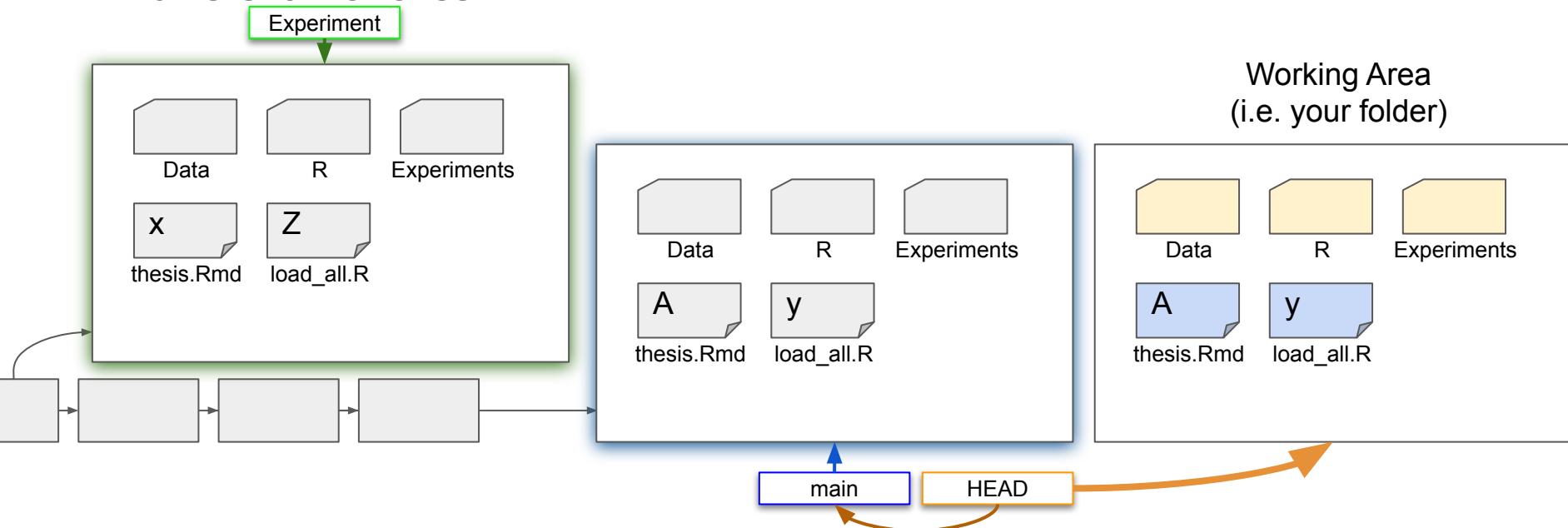
- "check out" Files: > git checkout load_all.R



Git -- "Version Control System" (VCS)

How about we organize things:

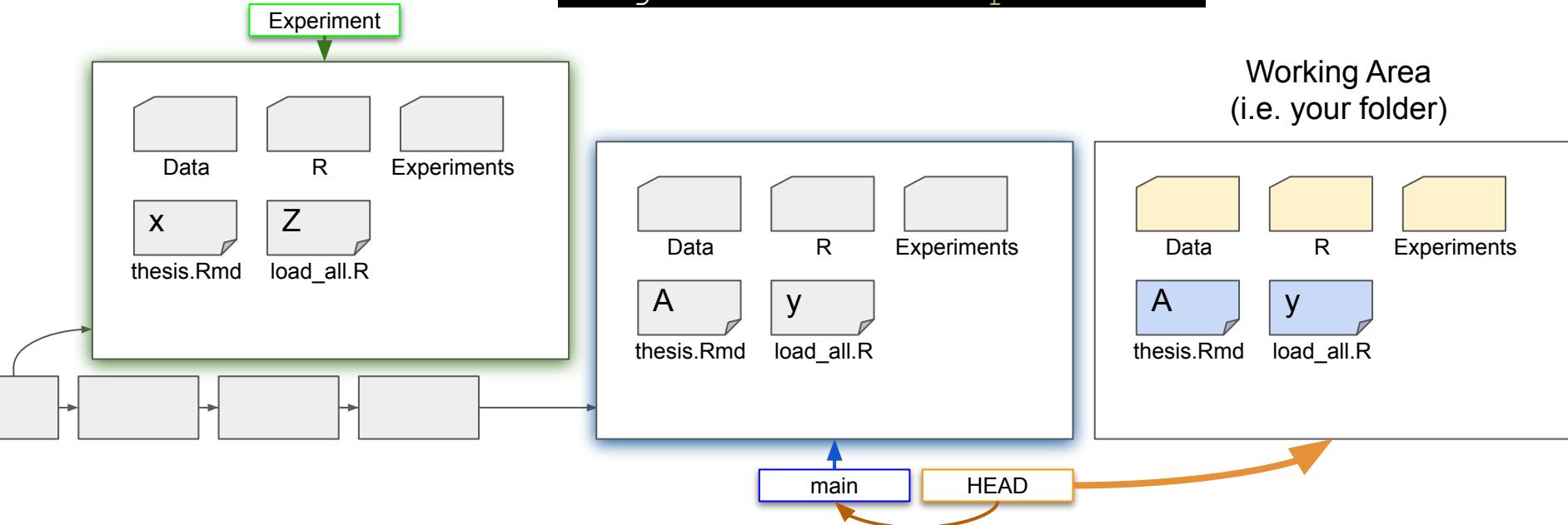
- different Branches



Git -- "Version Control System" (VCS)

How about we organize things:

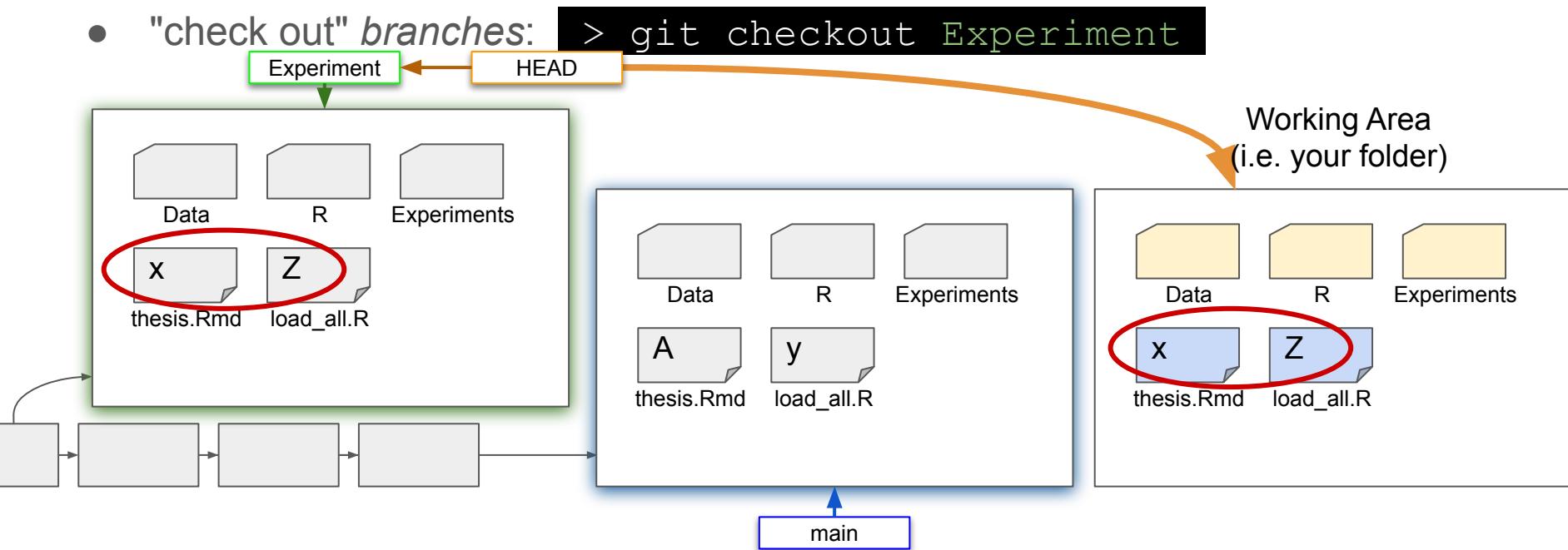
- "check out" *branches*: > git checkout Experiment



Git -- "Version Control System" (VCS)

How about we organize things:

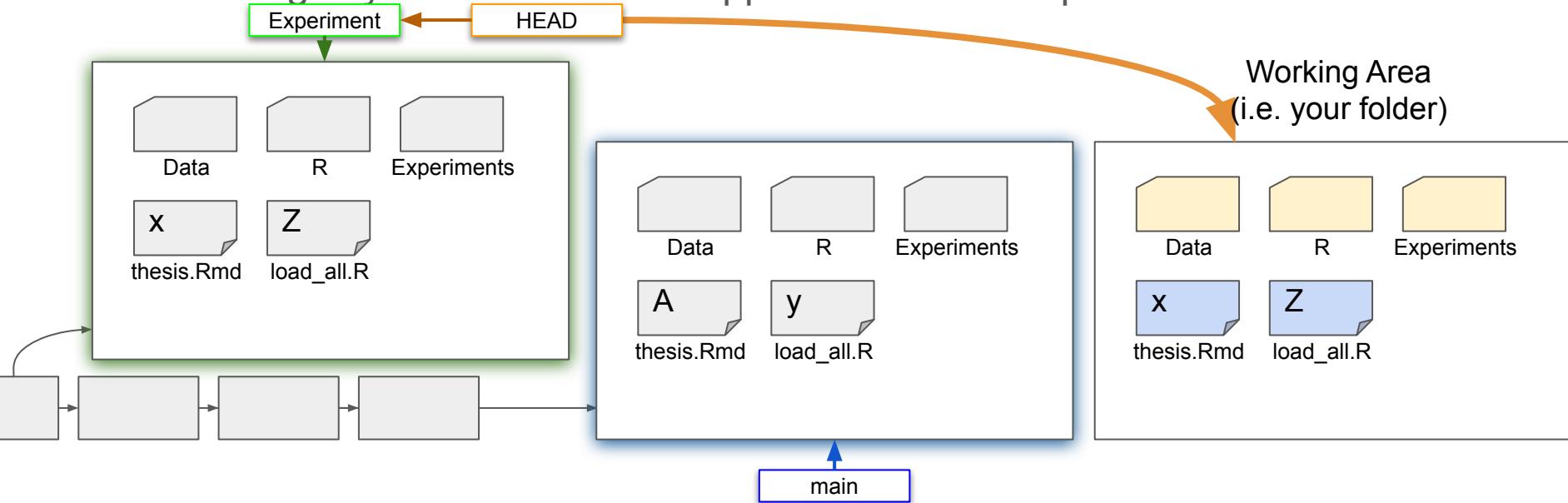
- "check out" *branches*: > git checkout Experiment



Git -- "Version Control System" (VCS)

How about we organize things:

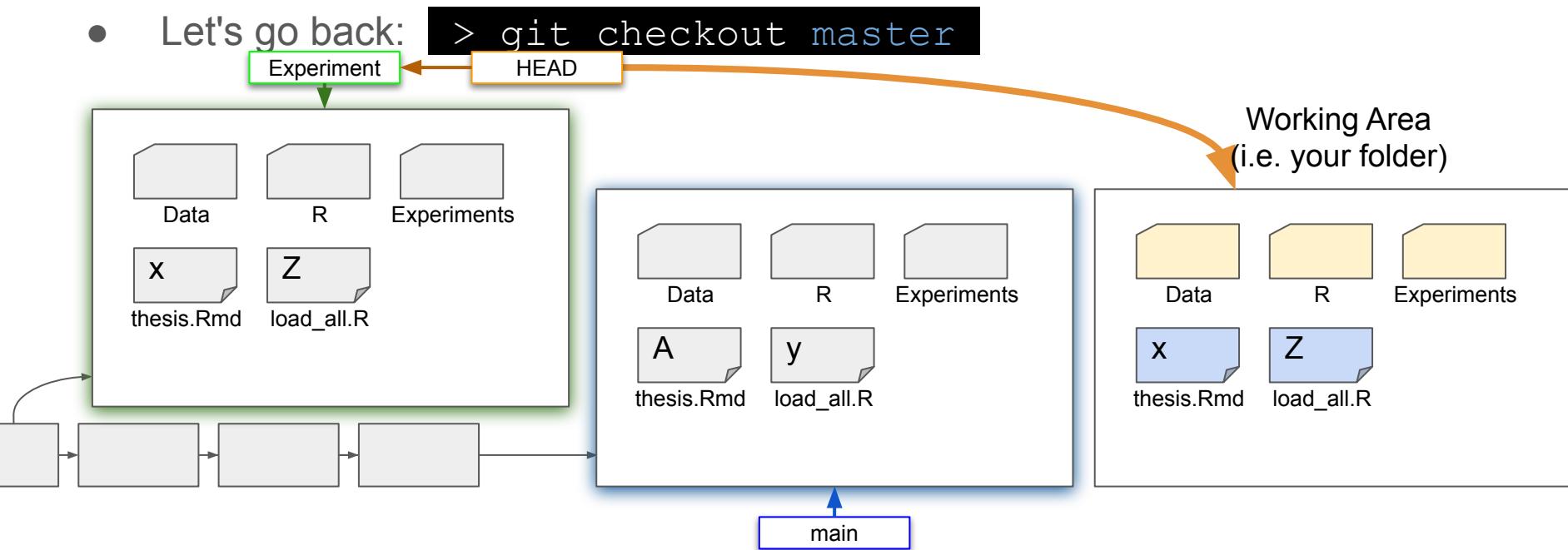
- all changes you make are now appended to the "Experiment" branch!



Git -- "Version Control System" (VCS)

How about we organize things:

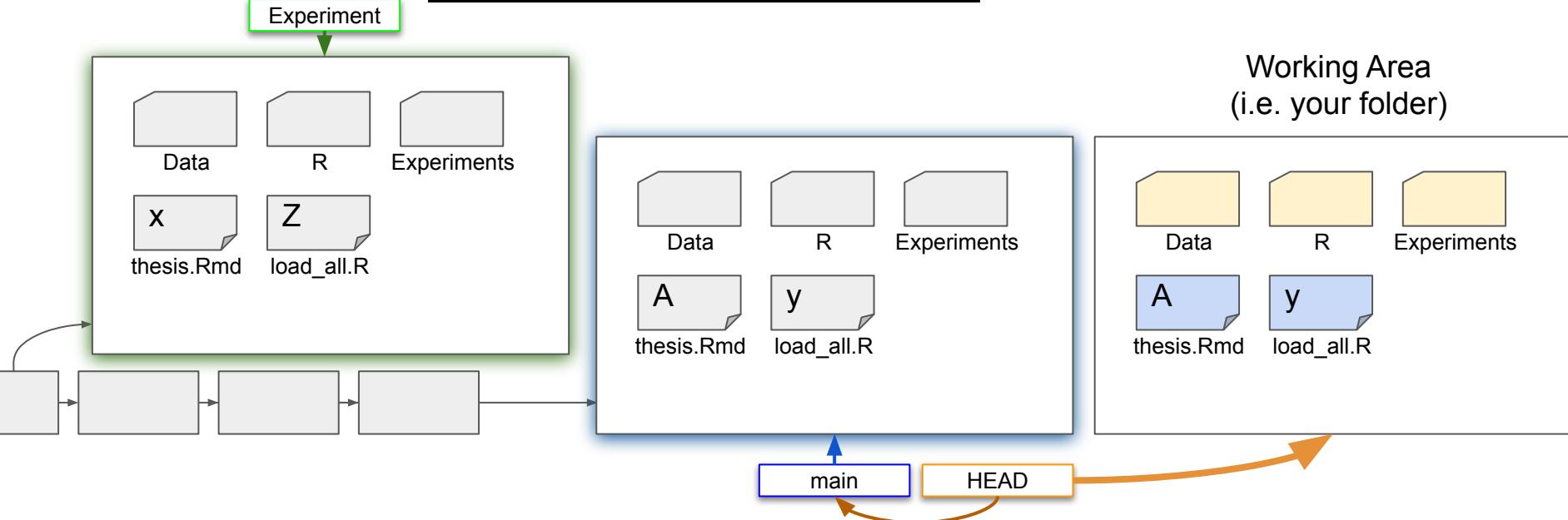
- Let's go back: `> git checkout master`



Git -- "Version Control System" (VCS)

How about we organize things:

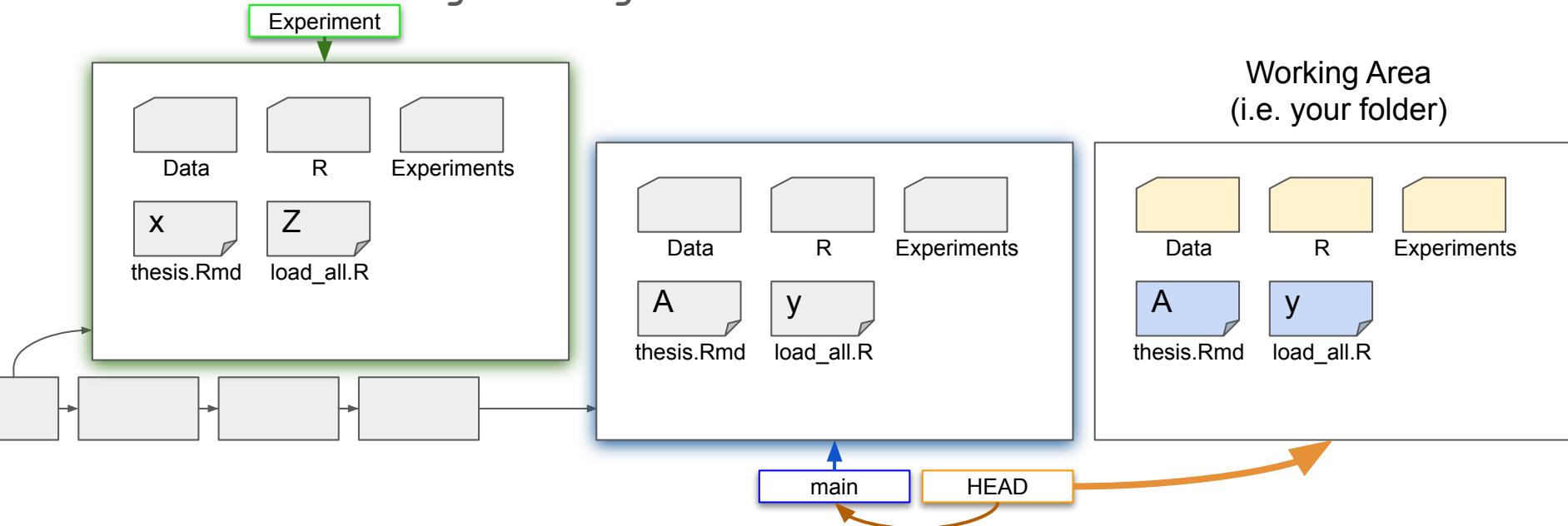
- Let's go back: > git checkout master



Git -- "Version Control System" (VCS)

How about we organize things:

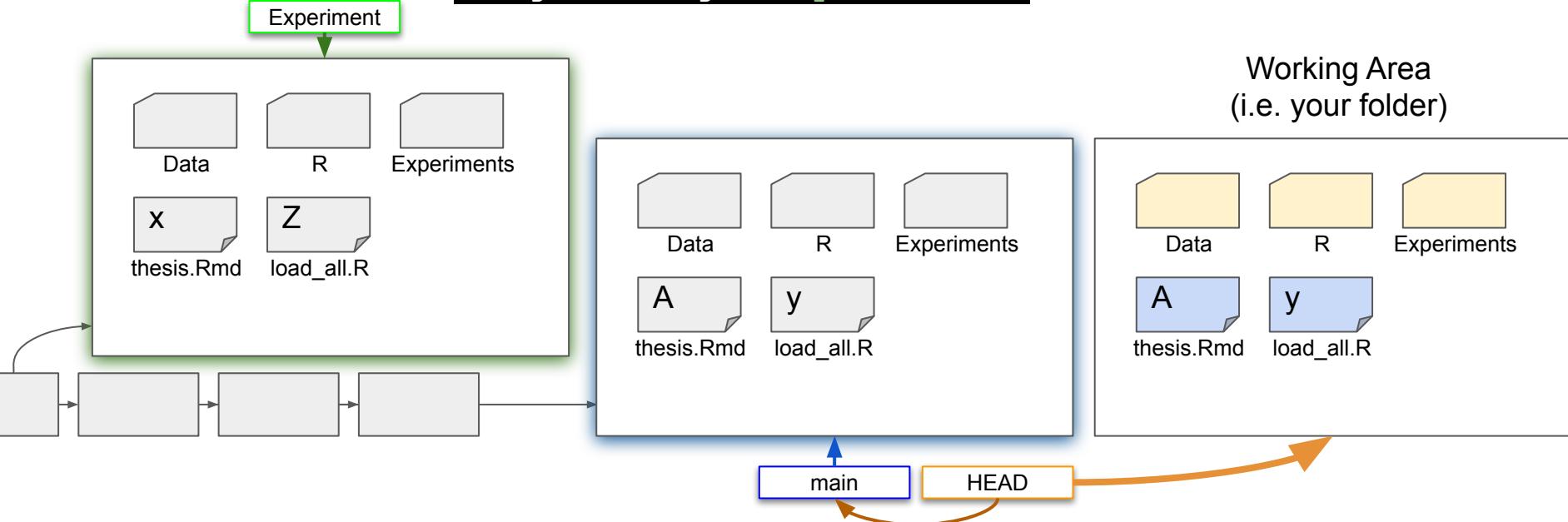
- Unite Histories: `git merge`



Git -- "Version Control System" (VCS)

How about we organize things:

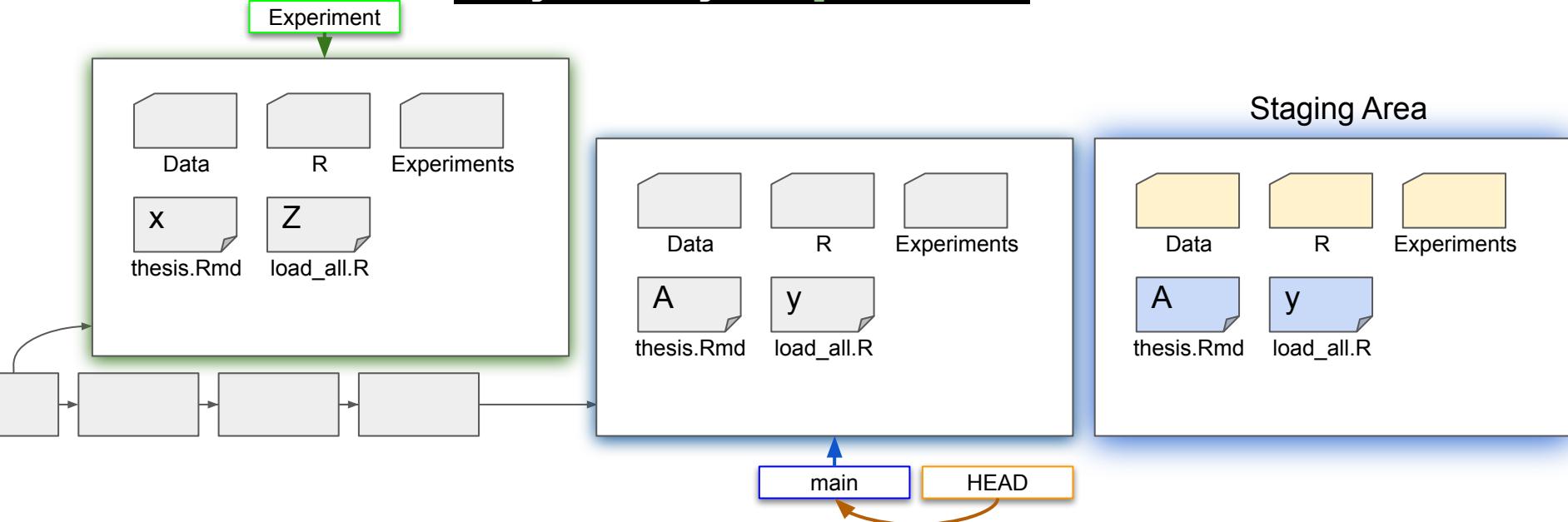
- Unite Histories: > git merge Experiment



Git -- "Version Control System" (VCS)

How about we organize things:

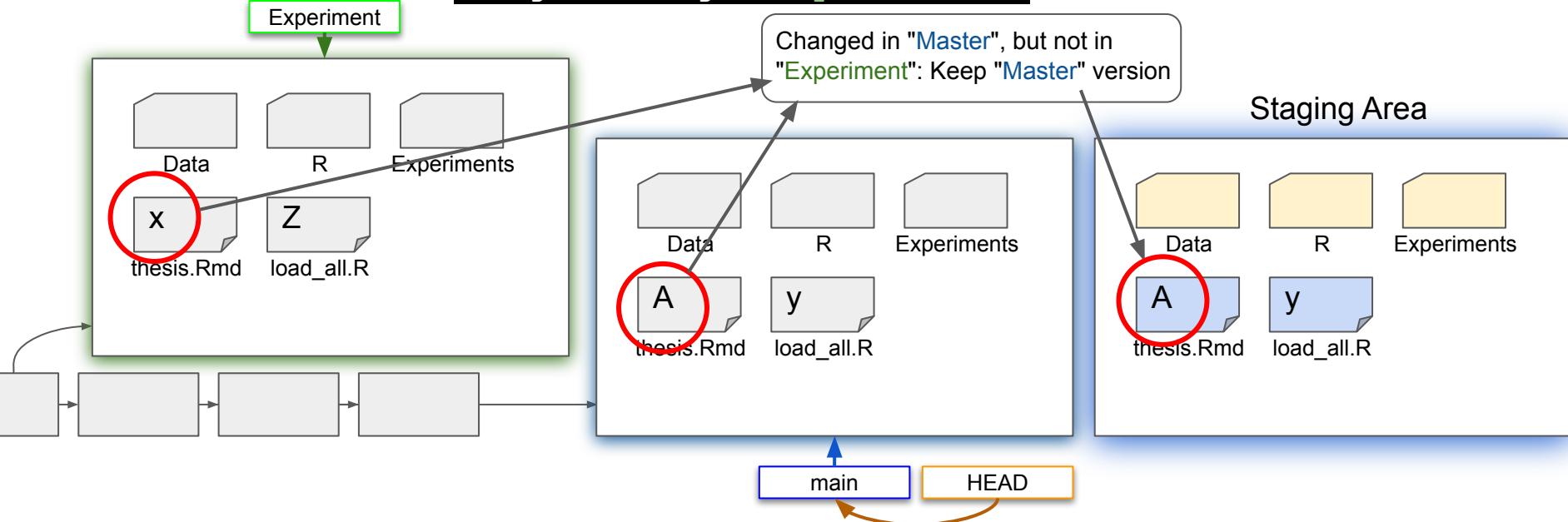
- Unite Histories: > git merge Experiment



Git -- "Version Control System" (VCS)

How about we organize things:

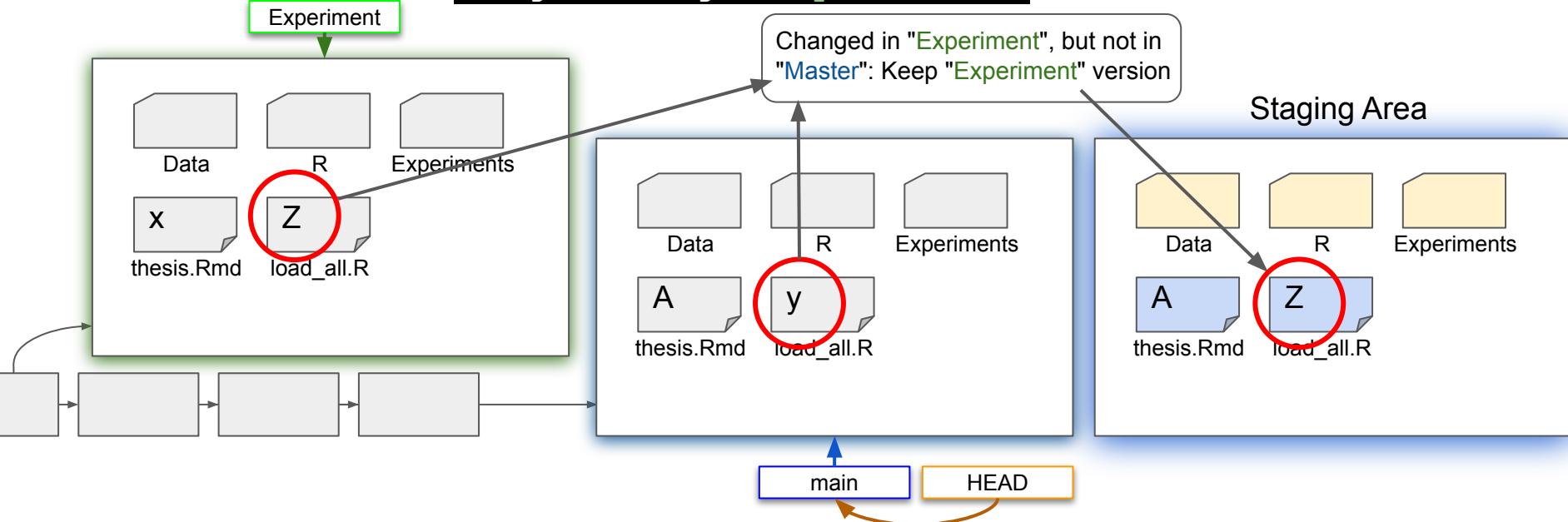
- Unite Histories: > git merge Experiment



Git -- "Version Control System" (VCS)

How about we organize things:

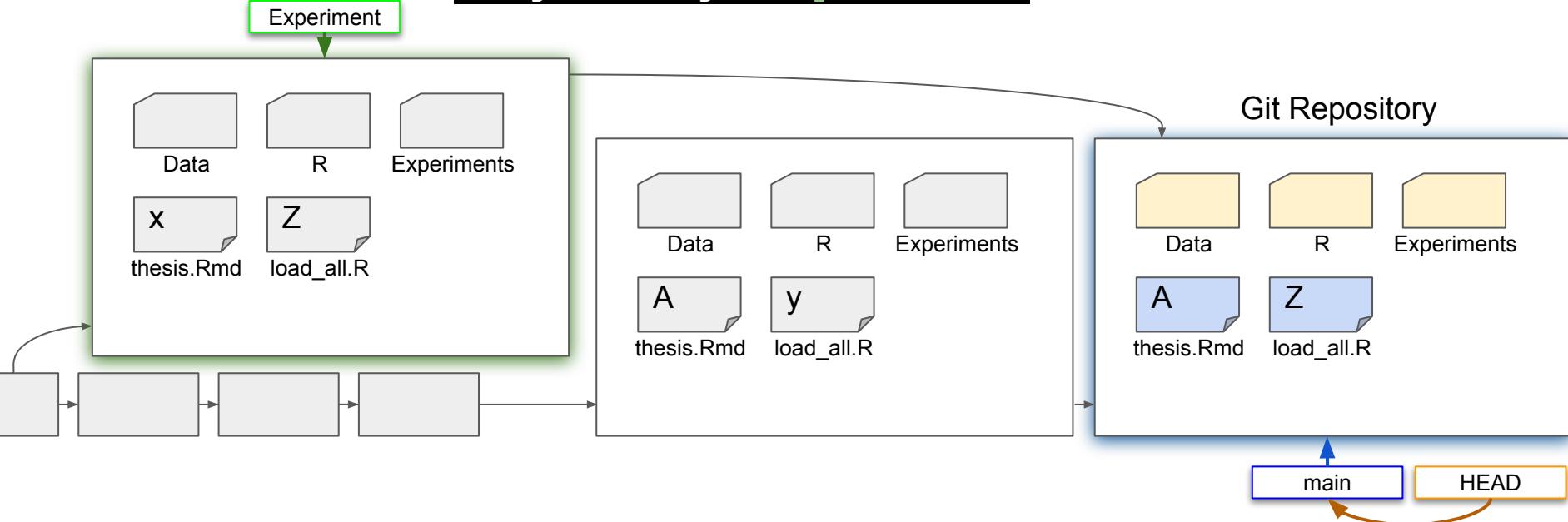
- Unite Histories: > git merge Experiment



Git -- "Version Control System" (VCS)

How about we organize things:

- Unite Histories: > git merge Experiment

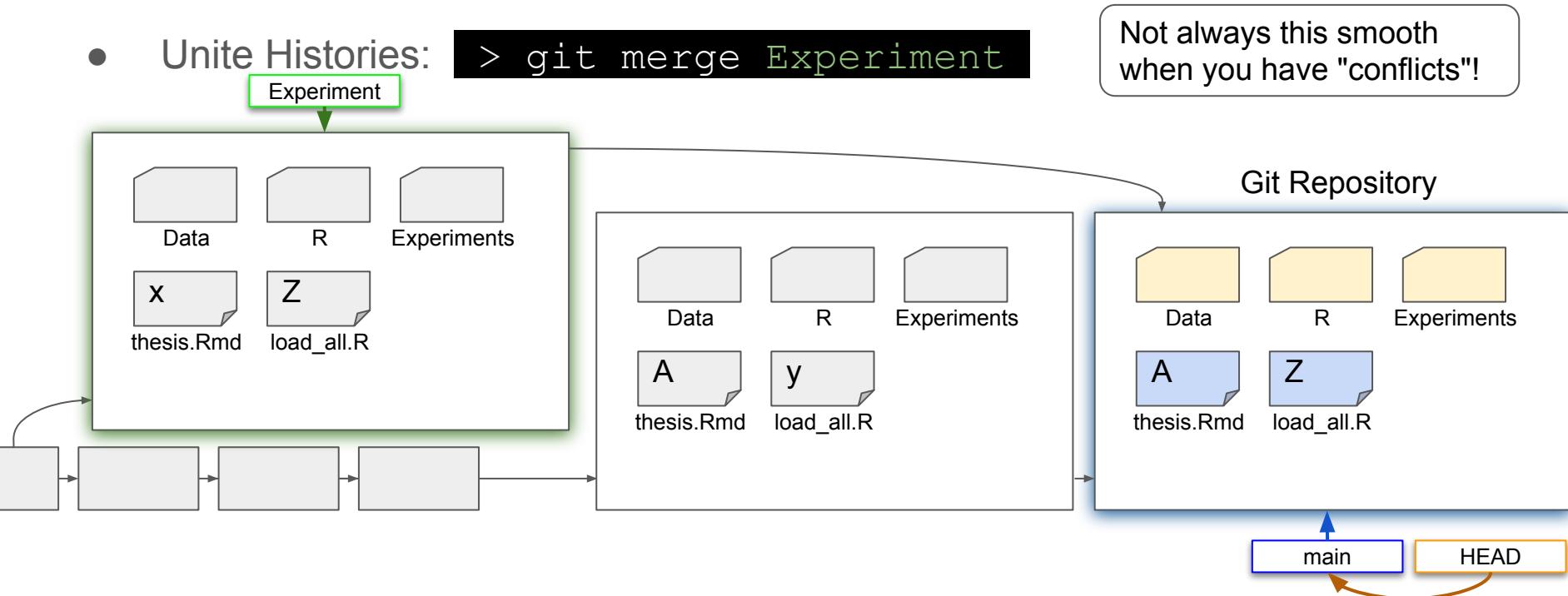


Git -- "Version Control System" (VCS)

How about we organize things:

- Unite Histories: > git merge Experiment

Not always this smooth
when you have "conflicts"!



Git -- "Version Control System" (VCS)

- You should now see that **git** is very useful
- Learning git is notoriously difficult, but you certainly get what you pay for
- Features of git we expect you to learn:
 - simple version control -- keep track of changed files
 - branching for alternate history of files
 - "merging" of branches to combine edits done in these branches
 - Not shown here: Using git for collaborative work
 - "fetch" / "pull" from *remote repositories*
 - "push" to remote repositories
 - You will learn this in your homework!

What We Expect You to Know

git commands to know:

- git init -- create a new (empty) repository
- git clone -- create a repository linked to another (online) repo
- git status -- see if files have changed
- git checkout -- check out files or branches
 - git checkout -b "new_branch_name" -- create new branch
- git branch -- find out on what branch you are
- git merge -- take changes from another branch into the current one
 - know how to handle merge conflicts!
- git diff -- see what has changed (default: since last commit)
- git add -- add changes to the staging area to be committed
- git commit -- commit changes in the staging area
 - git commit -a -- commit all changes
- git fetch -- get changes from remote repo and DON'T merge
- git pull -- get changes from remote repo and DO merge
- git push -- push changes to remote repo
- The .gitignore file -- let git ignore certain files / file types

Git



<https://xkcd.com/1597/>

Git



(Please be better than this guy)

Your first Homework Task

Your first homework Task

"Familiarize yourself with Git & GitHub"

(You will not get points for this task, but boy will you regret not doing this.)

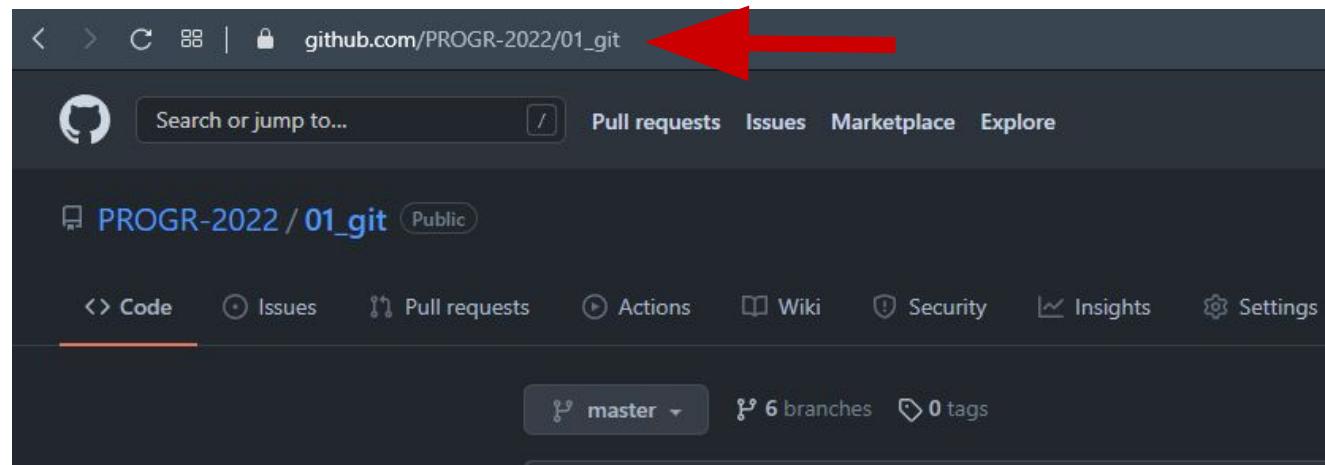
1. Log in to your GitHub account on <https://github.com>. (You already created an account, right?)

Your first homework Task

"Familiarize yourself with Git & GitHub"

(You will not get extra points for this task, but boy will you regret not doing this.)

1. Log in to your GitHub account on <https://github.com>. (You already created an account, right?)
2. Go to https://github.com/PROGR-2022/01_git

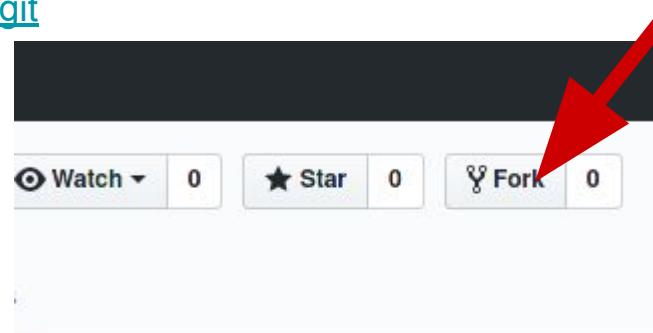


Your first homework Task

"Familiarize yourself with Git & GitHub"

(You will not get extra points for this task, but boy will you regret not doing this.)

1. Log in to your GitHub account on <https://github.com>. (You already created an account, right?)
2. Go to https://github.com/PROGR-2022/01_git
3. Click on "Fork" to create your own copy



Your first homework Task

"Familiarize yourself with Git & GitHub"

(You will not get extra points for this task, but boy will you regret not doing this.)

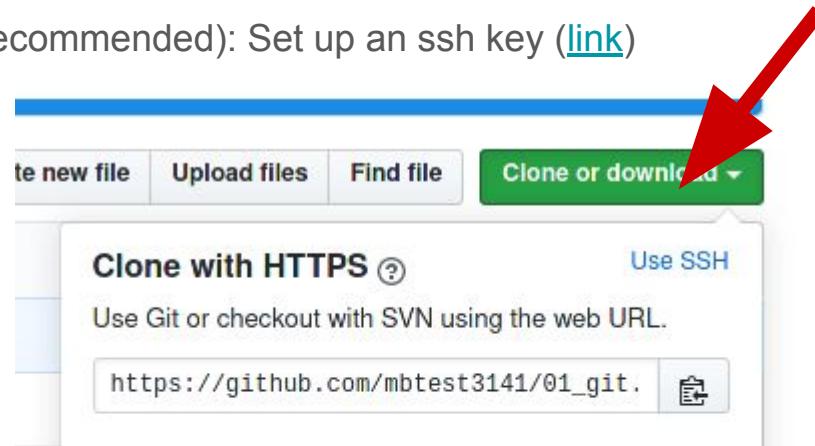
1. Log in to your GitHub account on <https://github.com>. (You already created an account, right?)
2. Go to https://github.com/PROGR-2022/01_git
3. Click on "Fork" to create your own copy
4. Learn what `git clone` does. Optional (but recommended): Set up an ssh key ([link](#))

Your first homework Task

"Familiarize yourself with Git & GitHub"

(You will not get extra points for this task, but boy will you regret not doing this.)

1. Log in to your GitHub account on <https://github.com>. (You already created an account, right?)
2. Go to https://github.com/PROGR-2022/01_git
3. Click on "Fork" to create your own copy
4. Learn what `git clone` does. Optional (but recommended): Set up an ssh key ([link](#))
5. Clone the repo to your own computer

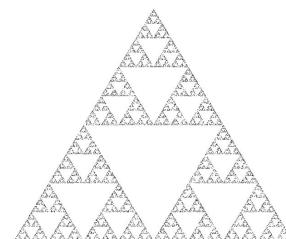


Your first homework Task

"Familiarize yourself with Git & GitHub"

(You will not get extra points for this task, but boy will you regret not doing this.)

1. Log in to your GitHub account on <https://github.com>. (You already created an account, right?)
2. Go to https://github.com/PROGR-2022/01_git
3. Click on "Fork" to create your own copy
4. Learn what `git clone` does. Optional (but recommended): Set up an ssh key ([link](#))
5. Clone the repo to your own computer
6. Do "The Task" as described in the repository!



Your second Homework Task

Your second homework Task

- ... will be published on Monday 2022-05-09
- ... and is due Sunday 2022-05-15
- ... and will assume that you know the basics of `git`