1. title

- 5 minutes not enough time to learn git and github
- only about 5% of git commands
- is enough time to have you answer:
- 1.what is git?
- 2.how does github relate?
- 3.can i use it for my EPA work?
- Plus cats…

2. Cats?

- Cats becuase its my ignite talk and I can do what I want
- Upon being reminded of this talk, i said all I was going to have time for was 5 minutes of cat slids
- so, cats!

3. Git?

- distributed version control system (DVCS)
- for text (code, manuscript, webiste, grocery list…)
- track all changes by multiple contributors and allows you to retrieve the text at any prior version
- think of it as Super Undo!!! ctrl-z that works for each and every time you save a document
- Based around concept of repositories (essentially a folder with every file init possible tracked)

4. Who created it and why?

- Created by Linus Torvald as a homegrown version control system for linux
- believe it or not, prior to developing git, linus version control done with a commercial solution (BitKeeper). in 2005 that stopped and Git was born.

5. Commands?

- git, like the linux community that developed it is a command line interface (CLI)
- submitted via the shell
- start with git, then command, then options, then arguments
- many GUIs exist (Rstudio, Github for windows,Git GUI)
- But CLI is quicker and easier to implement in most cases.

6. git add -A

- After some edits have been made, git will know that a file has changed since the last commit (more later)
- These are now "untracked files"
- use git add -A (or filename) to add all untracked files (or file(s)) to the staging area.
- These files are now being tracked by git and ready to commit

7. git commit -m

- git commit takes all the files in the staging area and submits them to the project history
- somewhat analagous to save (although different as you are saving a version not a file that needs to be done first)
- This version that you just created will now exist for ever to return to if you need it.

- This all happens in the same file, no need for v1 v2 v3_jwh v4 v4_realone…..
- All commits also require a message (-m) these are important as you used them to understand what changes were made at each commit

8. git clone URL

- First two commands are the basic workflow, next set are for managing your repositories
- clone, does just that, makes an exact copy of another repository
- Useful if you want to work on someone elses repo (code, manuscript, etc)
- can be local or via remote URL (think Github)

9. git remote -v

- remote URL can be URLs, other repos on networked filesytem tec.
- -v lists those that have been set
- this is the location from which you clone or where you submit and or get recent changes.
- the remote is usually the master repository (ie where everyone submits there changes)

10. git pull origin master

- So you have a cloned repo that others are also working on (or you are working on a separate machine)
- it is possible that the copy you are working on is not the most up-to-date
- pull gets those from your remote.
- origin is a short name for the URL of the remote
- master is the local branch into which your pull everything (bit more on branches later)
- simple case you can always use master and never really worry about branches

11. git push origin gh-pages

- So, you've made changes and committed to your local repository
- push is how you get those sent to the master remote repository
- again, origin is simply a shorthand for your remote
- gh-pages (could have been master, just messing with things) is the local branch that has committed changes to push

12. git checkout master

- if you have multiple branches (e.g master AND gh-pages)
- checkout lets you switch between them
- git branch lets you list them or create them
- So I have spent just over a minute and half on git commands.
- You are no presumably experts
- Next, I am going to talk about github

13. GitHub?

- so what is github?
- it is a company, founded in 2008 (with some recent bad press about bad behavior)
- it is a website, purpose is to provide easy way to share code
- it is a social network where you can follow, star, and discuss others work
- based on git, but . . .

14. Github != Git

- they are not the same.
- github is based on git, you need git to use the version control parts of github
- somethings on github can be used (ie. it is a website)
- but it is complimentary, but separate
- no some of the basic concepts of github

15. Fork it

- Forking
- Essentially this is cloning a github repo from one user to another user
- Both the source and the forked repo stay up on github

16. Pull Requests

- once you've forked a repo, you can work on it.
- if you make changes that you think would be good in the original repo you need how do you do that
- A pull request.

- You do that by pushing to your own version then requesting that owner of the original repo pull in your changes.

- getting the pull v push stuff straight can be a bit of a challenge
- good way to work with multiple authors on a paper. Same general concept of track changes and then one person "pulls" those in.
- only difference is the history of all the various versions are saved!

17. I've got issues

- term comes from software dev, but is essentially a way to discuss.
- discuss bugs
- discuss enhancements
- discuss new directs
- each are tracked, with numbers, can be assigned to people and so on.
- again think how this could be used to deal manuscript review!

18. Open Collaboration

- As github is a social network and repo's are, for the free versions, public (read, not write) open collaboration is at the heart of github
- can have organizations, team members, and collaborators

19. EPA?

- this all great, but can we use this at EPA?
- yes, sort of…
- US EPA organizational account
- Repos are private
- take about a week to get setup (many emails, approvals, listing in the RCS)
- Can be made public after review
- currently only have a limited OS license available to sue
- currently outisde collabs CANNOT be added, even with read only
- meeting about both issues today and early next week

20. References

- lastly some great stuff here.
- one that is missing is the Pro Git book available from the git-scm site.