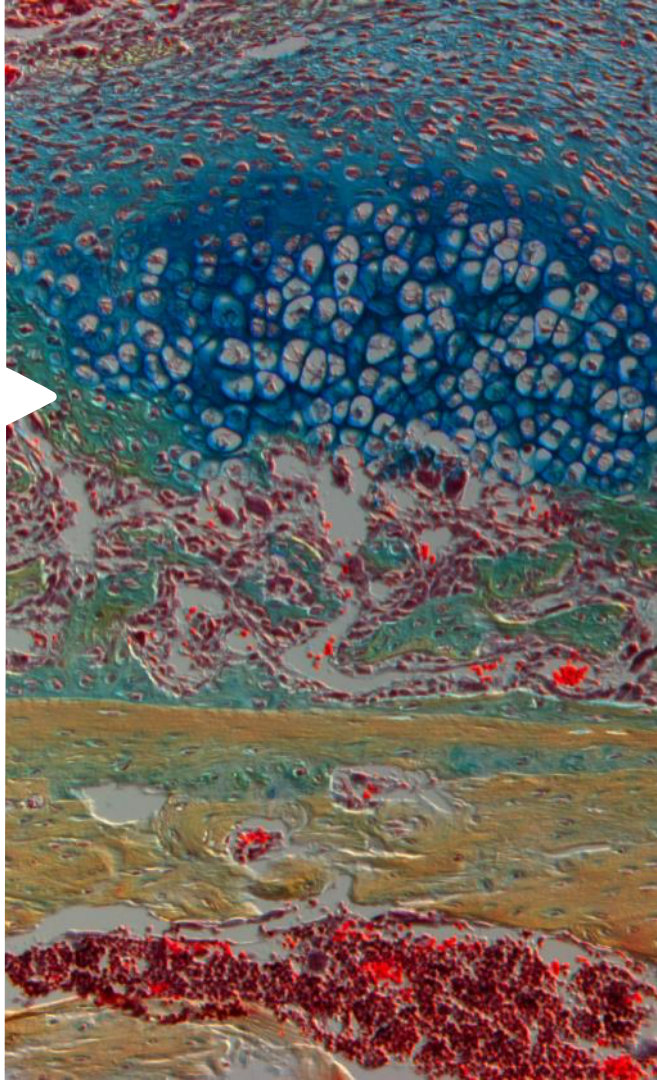
NYU Langone Health

Bioinformatics

Getting Started – 09/08/2020

Mark Grivainis



Todays Lecture

- Course Logistics
- History of Bioinformatics
- Breakout Session
- 10 Minute Break / Discussion
- Project Management

Lecture Format

- Lectures will start with a presentation covering a topic
- After the presentation there will be a worked example
  - This will mostly be in Jupyter / Google Colab
  - I will write some code and then give you time to reproduce the code
  - This format can adapt throughout the course
- At the end of each lecture a survey link will be posted in the chat
- A link to the recording of the lecture will be posted to Brightspace after the lecture

Course Modules

Two-week sections for each module

Each module will consist of four classes

- The first three will be lectures
- The fourth will be a presentation of your progress towards that module's assignment

Some sections have lectures with open topics that you can decide on during the course

The first module will not have an assignment

NYU Langone Health

Modules

1. Introduction

- Best Practices
- Data Manipulation
- Python

2. Biomarker Discovery

- Microarray Data
- A hint of Machine Learning
  - Scikit-Learn
- Python

3. NGS Alignment & Viz.

- Working with Sequencing Data
  - Acquisition (SRA Toolkit)
  - Alignment (BWA/Bowtie2)
  - Visualization (R/Python)
- Bash, HPC (Big Purple)

4. Gene Expression Analysis

- Running DSeq2 using data generated in Assignment 2
- Bash, HPC, R, Kallisto

5. Computational Proteomics

- Working with CPTAC Data
- Python

6. Chromosome Conformation Capture

- Working with Hi-C Data
- Bash, HPC , HiCExplorer

NYU Langone Health

Modules

7. Simulation

- Grid based simulation
- Code Optimization
- Python

Assignments

- Each module after the first will have an assignment
- Assignments will be completed in randomly assigned groups of four
- On the last Thursday of each module we will discuss your progress during the lecture
- On the Sunday before the next module starts you will need to submit a report
- Each assignment is worth fifteen percent
  - 10% for the report
  - 5% from self assessment within the group

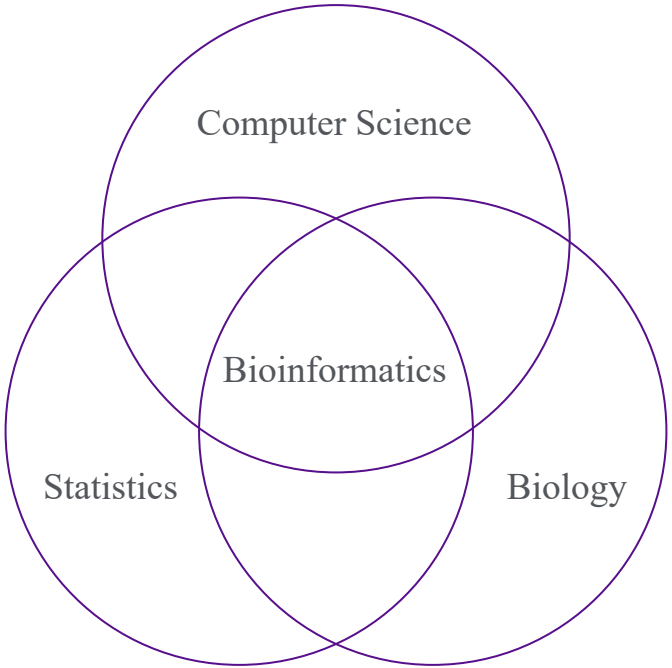
Assignment Progress Presentation

- Presentations
  - There will be a single PowerPoint presentation for all the groups
  - Each group can add up to two figures to the presentation
    - The figures must be different from the existing figures that have already been added
    - It is fine if your group does not add a figure due to your analysis matching the other groups
  - We will then discuss each figure in the presentation during the lecture
  - The purpose of this presentation is to ensure that each group is on the right track with their analysis

Assignment Report

- Each group will submit a single report
- The submission will be on Brightspace
- This should include:
  - The tools you used and steps you followed to complete your analysis
  - Your results and findings
  - Any changes you made after the progress presentation should be mentioned
  - A breakdown of how each group member contributed to the project and a score out of 5 per group member
  - Submit a zipped folder containing your code and a README.md file with instructions to run the code

What is Bioinformatics?



NYU Langone Health

A Brief History of Bioinformatics

NYU Langone Health

1950–1970: The origins

1950's | It starts with protein sequencing

1950 | Edman sequencing

1958 – 1962 | Margaret Dayhoff and Robert S. Ledley develop COMPROTEIN

- This system used three letter abbreviations for amino acids, Dayhoff later developed the single letters still used today

1963 | Paleogenetics (Zuckerkandl and Pauling)

1965 | Dayhoff and Eck's 1965 Atlas of Protein Sequence and Structure

- The first ever biological sequence database

1970 | Needleman and Wunsch, dynamic programming for pairwise protein comparisons

1970–1980: Paradigm shift from protein to DNA analysis

1972 | DNA Amplification using E. coli (Jackson, Symons and Berg)

1976 | Maxam–Gilbert sequencing method

1977 | Sanger DNA sequencing

1977 | First ready to use microcomputers (Commodore PET, Apple II and Tandy TRS-80)

- All three included the BASIC programming language

1979 | The Staden Package

- Roger Staden
- (i) search for overlaps between Sanger gel readings; ii) verify, edit and join sequence reads into contigs; and (iii) annotate and manipulate sequence files.
- Last release was in 2016

1980–1990: Parallel advances in biology and computer science

1983 | Polymerase Chain Reaction (PCR) (Kary Mullis)

1984 | GCG package

- 33 command-line tools to manipulate DNA, RNA or protein sequences

1984 | DNASTAR is formed

1985 | First bioinformatics journal - Computer Applications in the Biosciences (CABIOS)

1986 - 1987 | EMBL, GenBank and DDBJ

- in order, among other things, to standardize data formatting, to define minimal information for reporting nucleotide sequences and to facilitate data sharing between those databases
- Today this union still exists and is now represented by the International Nucleotide Sequence Database Collaboration (http://www.insdc.org/)

1987 | PERL (Practical Extraction and Reporting Language) Larry Wall

- Primary bioinformatic language until the late 2000s

1990–2000: Genomics, structural bioinformatics and the information superhighway

1991 | The Human Genome Project is initiated (National Institutes of Health)

1992 | NCBI takes over responsibility for GENBANK

1993 | the EMBL Nucleotide Sequence Data Library (that included several other databases such as SWISS-PROT and REBASE), was made available on the Web

1995 | First complete genome sequencing of a free-living organism (Haemophilus influenzae)\

- The Institute for Genomic Research (TIGR) led by geneticist J. Craig Venter

1996 | BioPerl

1998 | Celera Genomics (a biotechnology firm run by Venter) rival effort to HGP

2000–2010: High-throughput bioinformatics

Late 2000s | Python becomes a major bioinformatics language

2003 | Human Genome Project is completed

2005 | Genomic Standards Consortium

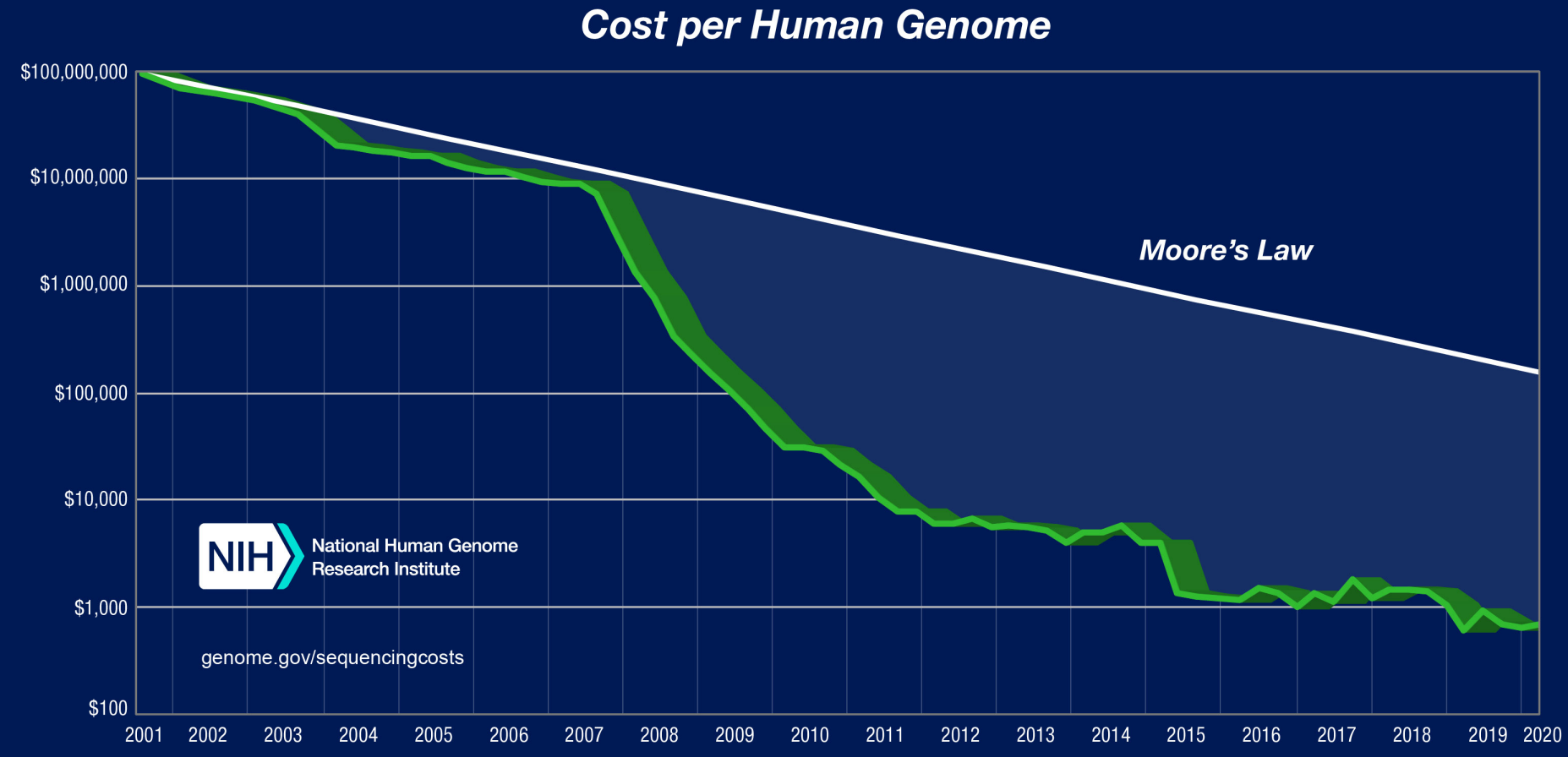
- Their goal is to make genomic data standardized and discoverable

2005 | Galaxy is released

2005 | The Cancer Genome Atlas (TCGA) pilot is launched

2008 | Moore's Law stops being an accurate predictor of DNA sequencing costs

2008 | The “1000 Genomes Project” launches


Cost per Human Genome
$100,000,000
$10,000,000
$1,000,000
$100,000
$10,000
$1,000
$100
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
Moore's Law
NIH
National Human Genome Research Institute
genome.gov/sequencingcosts

2010–Today: Present and future perspectives

2012 | TCGA launches the Cancer Genomics Hub (CGHub) as the new TCGA data repository

2015 | Identity Crisis, who can be a bioinformatician

2016 | Cloud Based bioinformatics (Seven Bridges)

Breakout Room

- I will randomly assign you to rooms
- Take this time to introduce yourselves, you will all be working together at some point during the course
- In future we will use breakout rooms for discussions
  - I will post the questions the day before each lecture at the latest
  - I will provide a link to a google doc so that you can write down your answers
- I will close the rooms in 5 minutes

10 Minute Break / Discussion

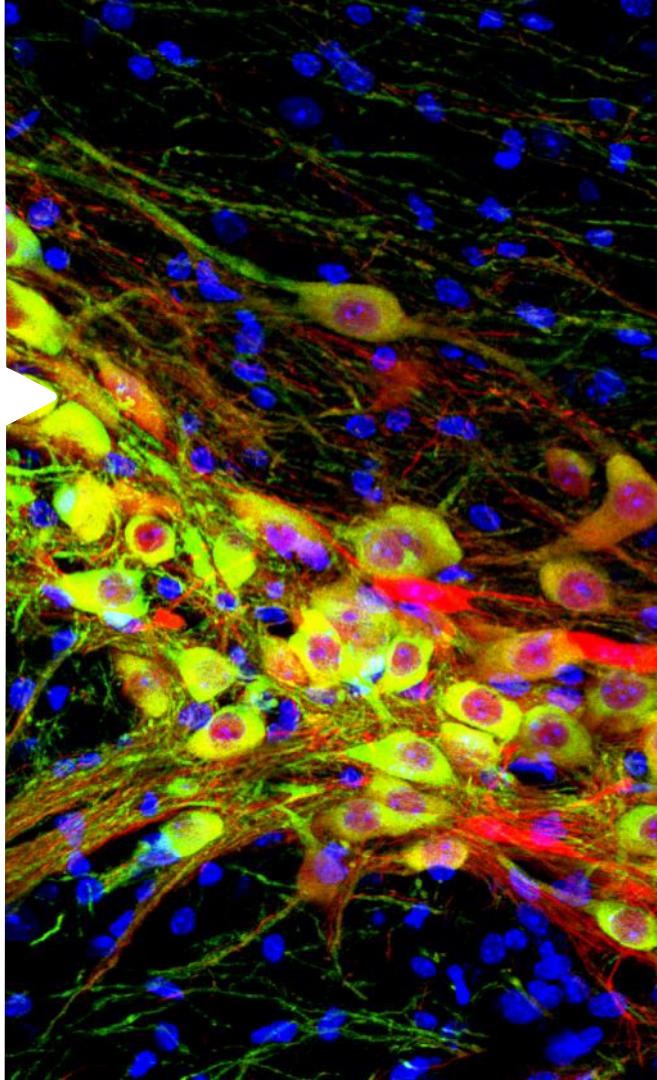
NYU Langone Health

NYU Langone Health

Project Management

Write Subtitle In This Area

Division Name



Project Directory

- Follow a Guideline
  - eg: Cookiecutter Data Science (link)
- Benefits
  - Shortens the time for newcomers to understand the project
  - If all your projects have the same structure it simplifies the process of switching between projects
  - Provides structure and workflow to your projects

Cookiecutter Data Science Template

```
|- data/
|- models/
|- notebooks/
|  |- exploratory/
|  |- reports/
|- main.py
|- README.md
|- requirements.txt (If you are using pip)
|- .gitignore
|- LICENCE
|- src/
|  |- __init__.py
|  |- data/
|  |- make_dataset.py
|  |- features
|  |- build_features.py
|- Makefile (Good practice, but optional) (Guide on makefiles http://zmjones.com/make/)
|- environment.yml (If you are using conda)
```

Version Control Using Git

NYU Langone Health

What is a Version Control System (VCS)

- Software for keeping track of changes within files
  - git, subversion, mercurial
- There are two main uses for version control:
  1. Keeping track of changes and being able to go back to earlier versions of files
  2. Allowing multiple users to work on the same project simultaneously

Most VCS systems allow for files to be backed up and stored in online repositories. GitHub is the online repository for the git VCS, though, there are others such as BitBucket and GitLab.

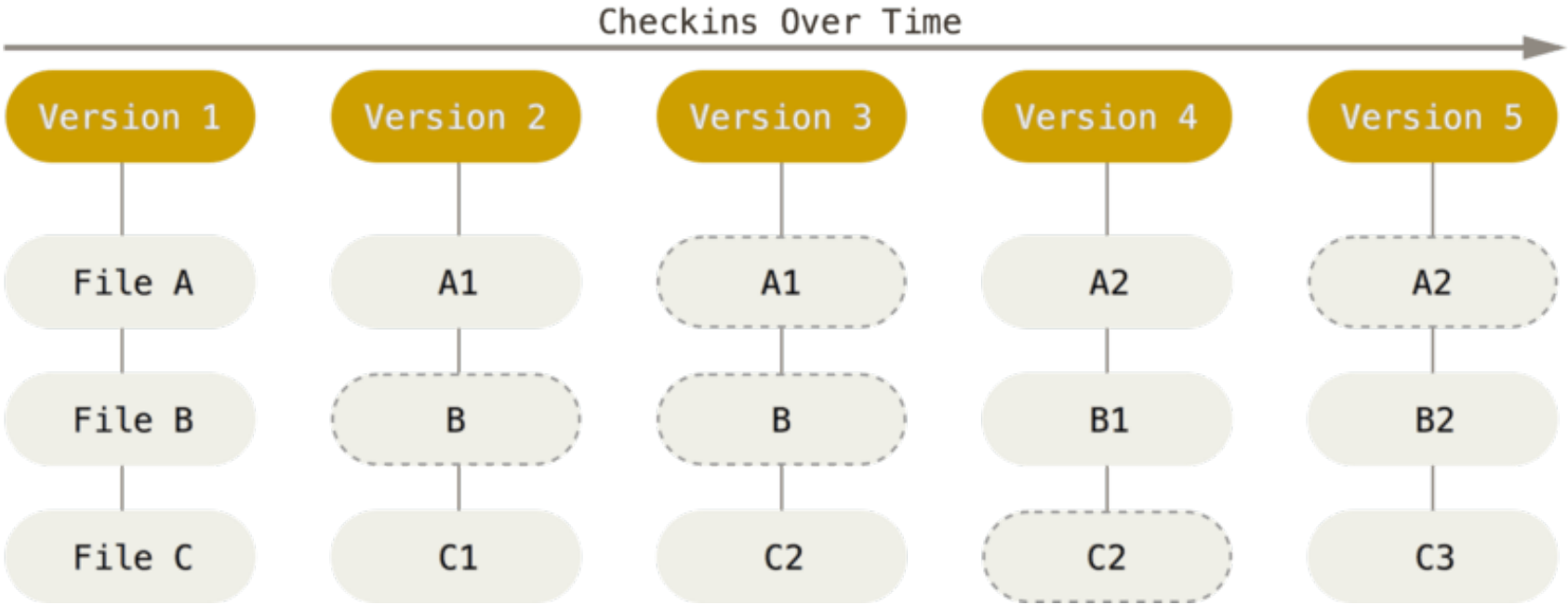
Git

- Birthed in 2005 due to a dispute between BitKeeper and the community who develop the Linux kernel
- This community then decided to take the lessons that they had learnt using BitKeeper and create a new tool with the following goals:
  - Speed
  - Simple Design
  - Strong support for non-linear development
  - Fully Distributed
  - Able to handle large projects
- This book covers git in detail: https://git-scm.com/book/en/v2

Setting up a Git Repository

1. Always **`cd`** into the correct directory
   - A common error beginners make is to initialize their home directory as a git repo
     - This will cause errors when working in sub directories
2. Run **`git init`** to initialize an empty git repository
   - This will create a folder called '.git' in the current directory
   - The .git folder contains all the files git requires to keep track of your repository
3. Common Git commands
   - **`git add`** - Add any new changes to the staging area
   - **`git commit`** - Take all the staged changes and save a snapshot of them

Git File Tracking

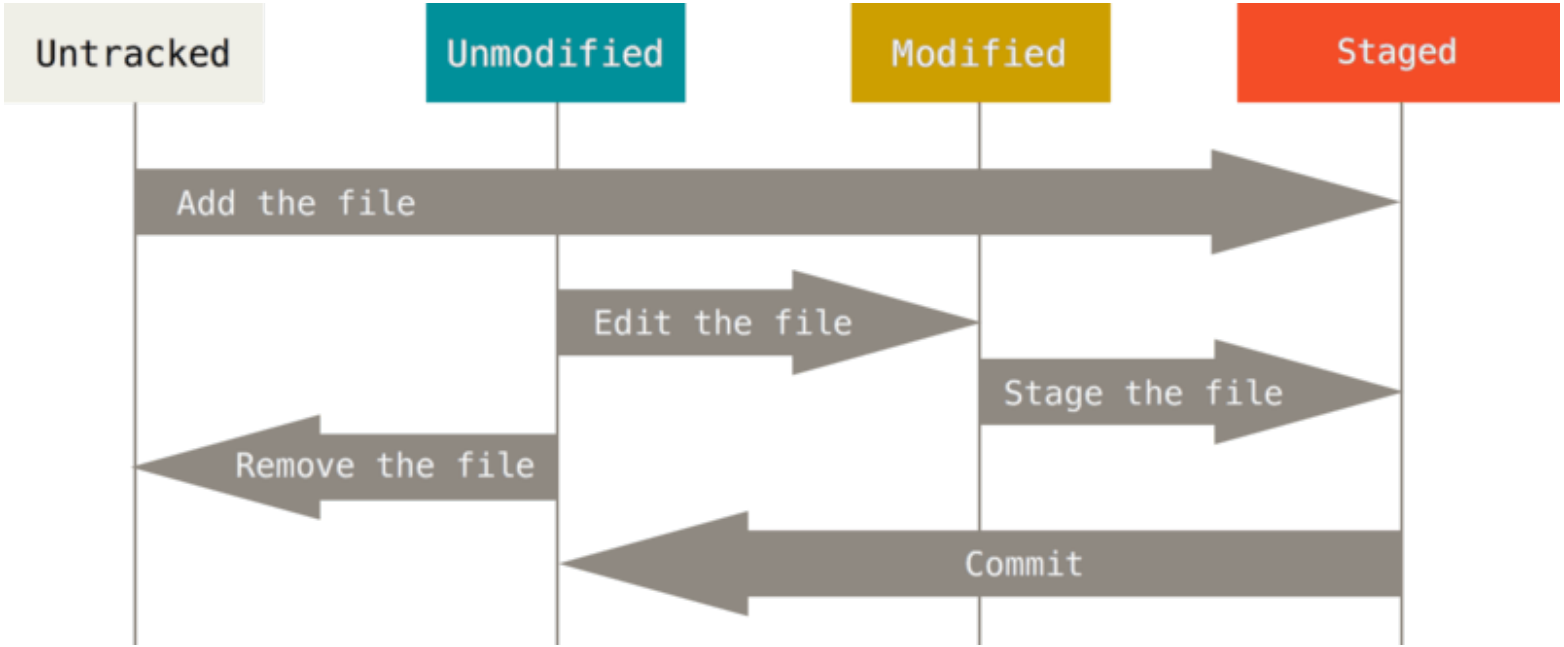


NYU Langone Health

Ignoring Files

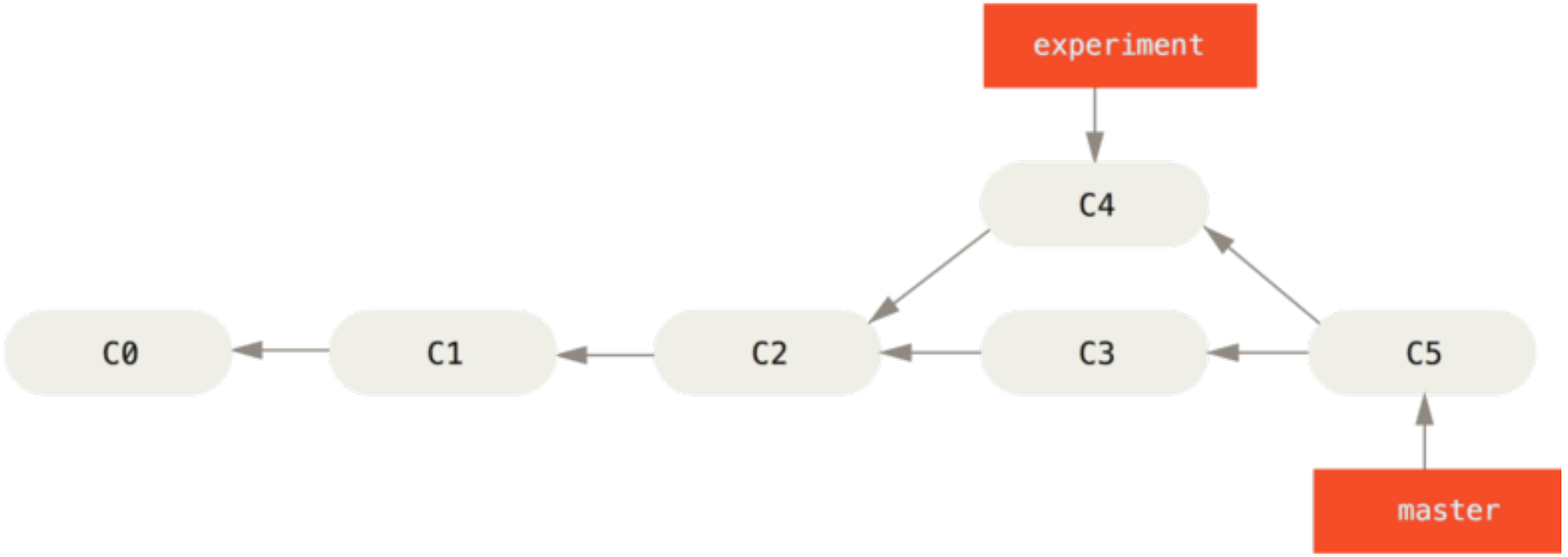
- Add a '**.gitignore**' file to the git repository
- List the files / file types / directories the .gitignore file
- You can copy pre-generated templates from https://www.gitignore.io/
- When you create a project on GitHub an option is provided to generate this file
- Example Patterns
  - `/foldername     # will ignore the folder and all of its contents`
  - `*.csv           # will ignore all csv files in the project directory`
  - `/output/*.png   # ignore all png images in an output directory`

Recording Changes



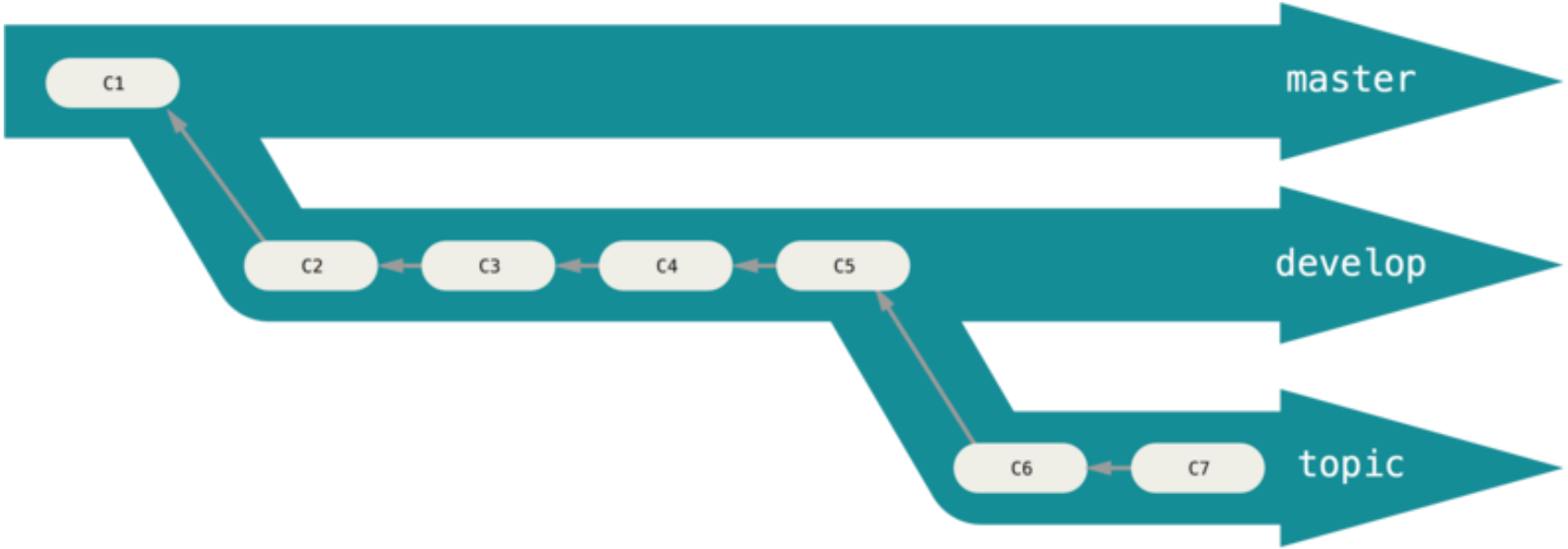
NYU Langone Health

Branches



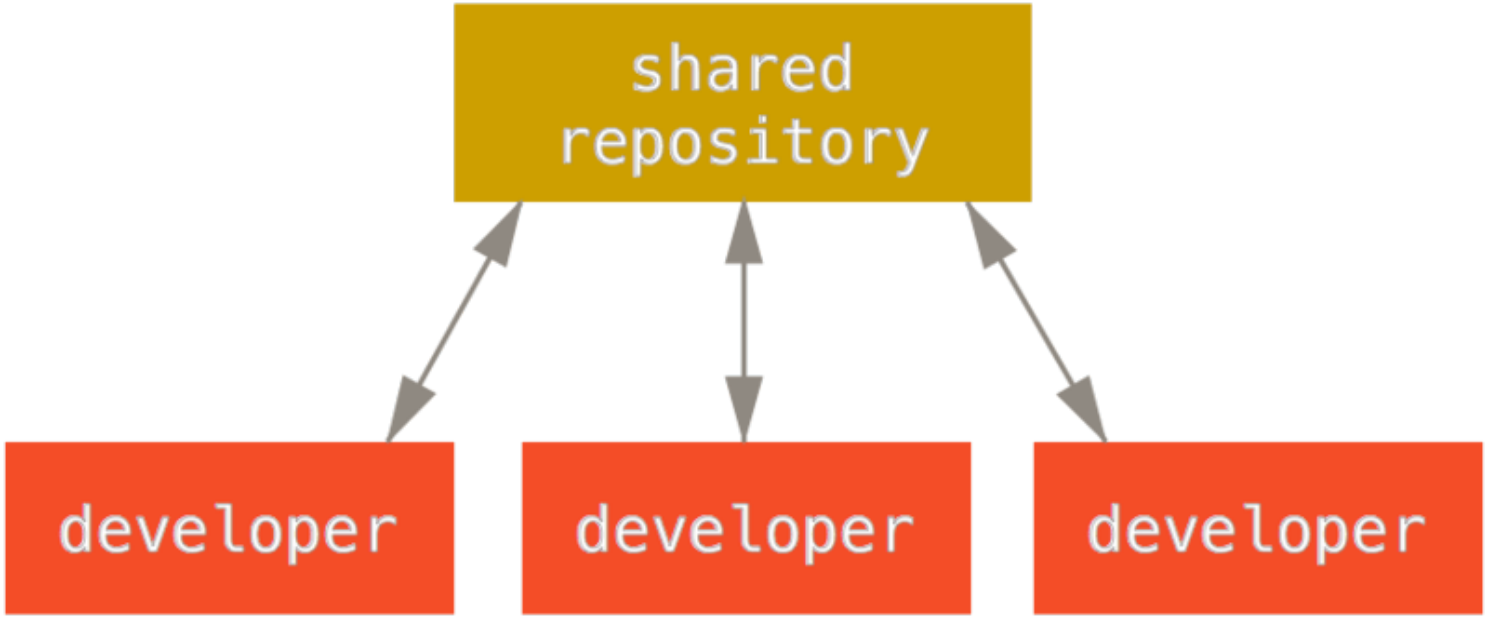
NYU Langone Health

Branching Strategy



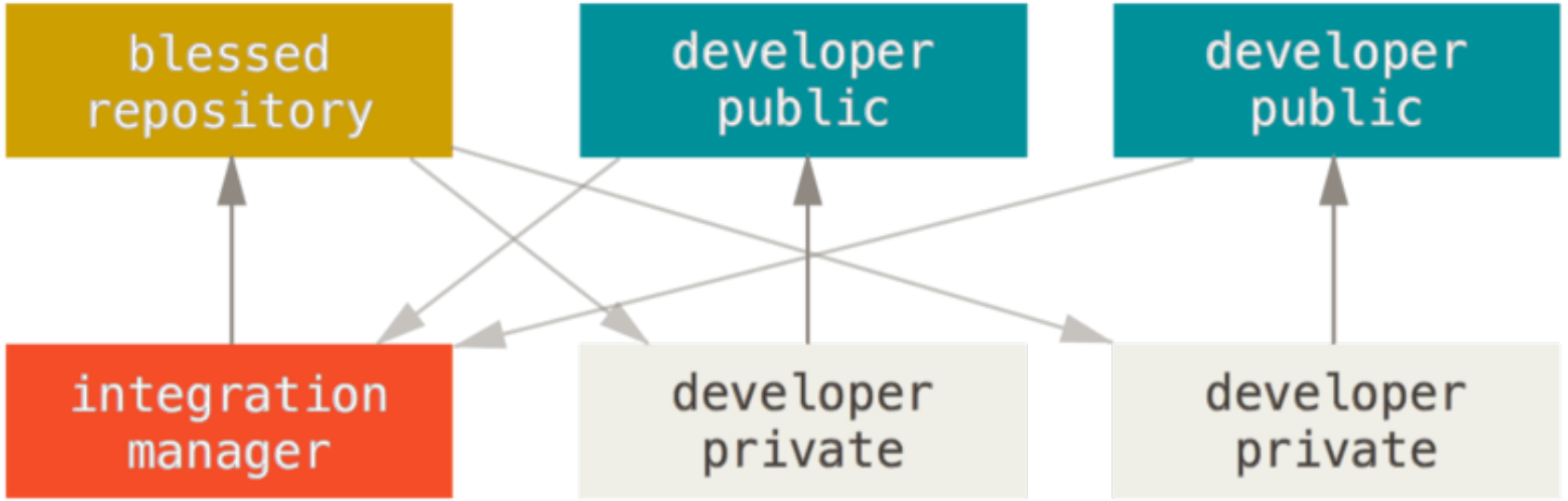
NYU Langone Health

Workflows: Centralized (Common)



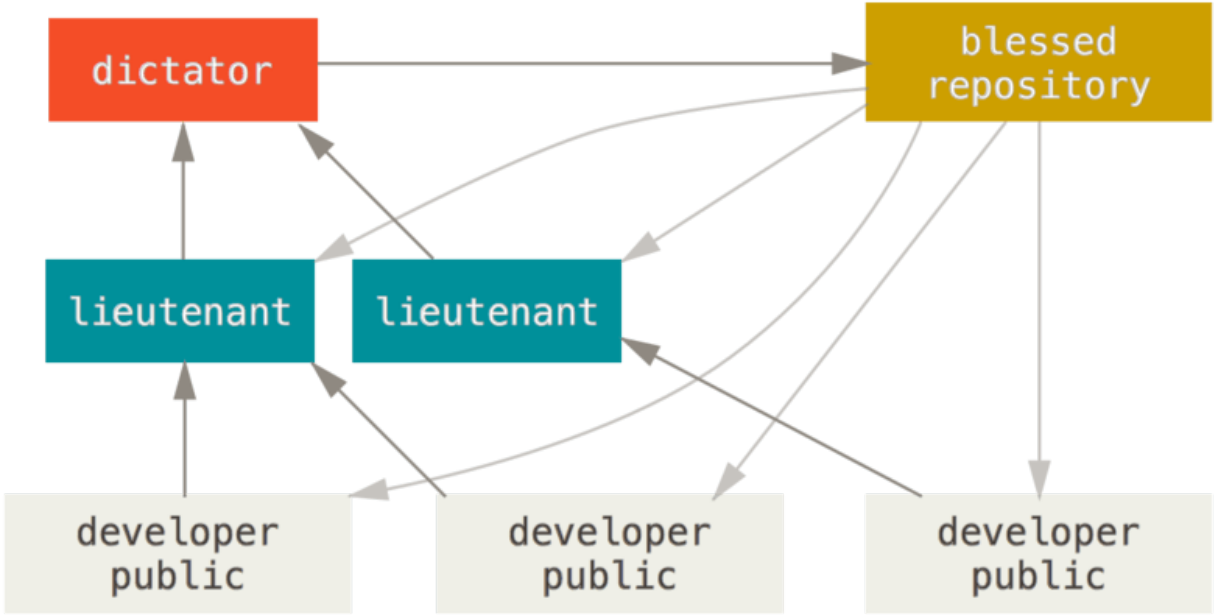
NYU Langone Health

Workflows: Integration-Manager (Fairly Common)



NYU Langone Health

Workflows: Dictator and Lieutenants (uncommon)



NYU Langone Health

Simple Git Workflow

- Steps

1. **git status** – Make sure your current area is clean
2. **git pull** – Get the latest version from the remote. This simplfies merging later
3. Edit your files and make your changes
    1. Remember to run your linter and do unit tests!
4. **git status** – Find all files that are changed. Make sure to watch untracked files too!
5. **git add [files]** – Add the changed files to the staging area.
6. **git commit -m "message"** – Make your new commit.
7. **git push origin [branch-name]** – Push your changes up to the remote.

GitHub Flow (Centralized Workflow)

So, what is GitHub Flow?

- Anything in the master branch is deployable
- To work on something new, create a descriptively named branch from **master**
  - ie: feature/generate_new_dataset
- Commit to that branch locally and regularly push your work to the same named branch on the server
- When you need feedback or help, or you think the branch is ready for merging, open a pull request
- After someone else has reviewed and signed off on the feature, you can merge it into master
- Once it is merged and pushed to ‘master’, you can and should deploy immediately
- Optional - Add a dev branch
  - a. Features are merged into the dev branch
  - b. The dev branch is merged into the master branch

Git – Tips and Tricks

- Moving files using regular copy/cut/mv and paste can break the history created by Git
  - If you need to rearrange your folder use the **`git mv`** command to move the files
- If you forget to add a file to a commit or want to make a small change use the --amend flag
  - **`git commit –amend`**
- Undo changes using the checkout command
  - **`git checkout <filename>`**
- Use **`git log`** to see the history of the repository

Course Survey

- https://nyumc.qualtrics.com/jfe/form/SV_6yztFdY5iQpnuBL

Citations

- Gauthier, Jeff, Antony T. Vincent, Steve J. Charette, and Nicolas Derome. 2019. “A Brief History of Bioinformatics.” *Briefings in Bioinformatics* 20 (6): 1981–96.
- https://git-scm.com/book/en/v2
- http://scottchacon.com/2011/08/31/github-flow.html

NYU Langone Health