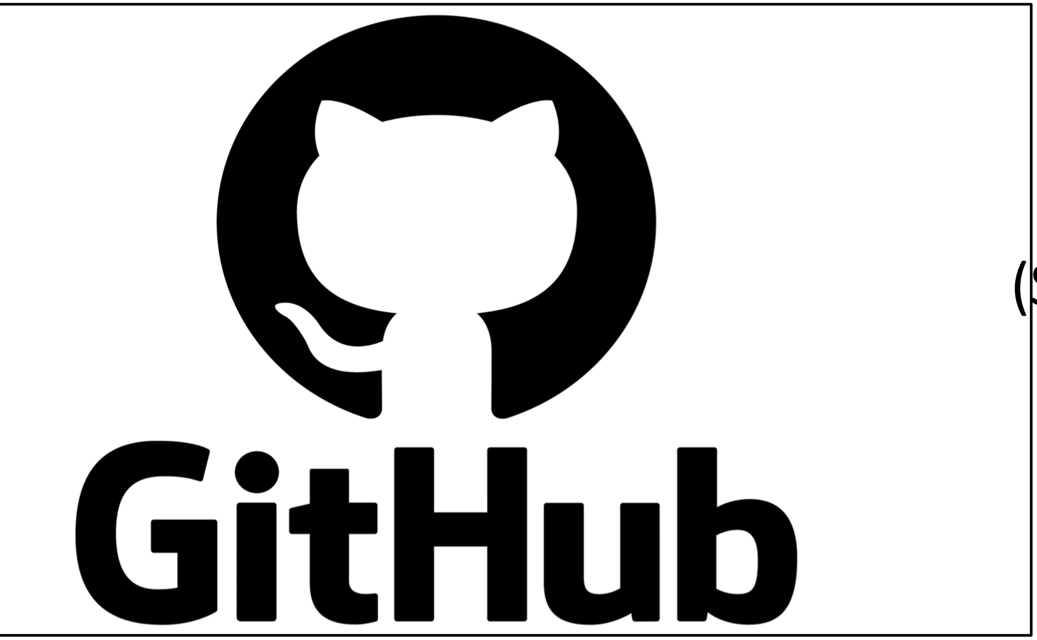
Setting Bioinformatics Pipelines

Reasoning with data

GitHub



January 2024

(Slides: Alberto Maillo)

جامعة الملك عبدالله
للعلوم والتقنية
King Abdullah University of
Science and Technology

1- Concepts and guideline

2-Presentation for publication

1- Concepts and guideline

GitHub

- Used for software development and version control
- It is a service hosted on Web
  - Repository: folder where your project is kept
    - Public or private
    - Files no to be tracked
    - License
- It provides graphical interfaces

Unit_Github

GitHub

- Used for software development and version control
- It is a service hosted on Web
  - Repository: folder where your project is kept
    - Public or private
    - Files no to be tracked
    - License
- It provides graphical interfaces

Unit_Github

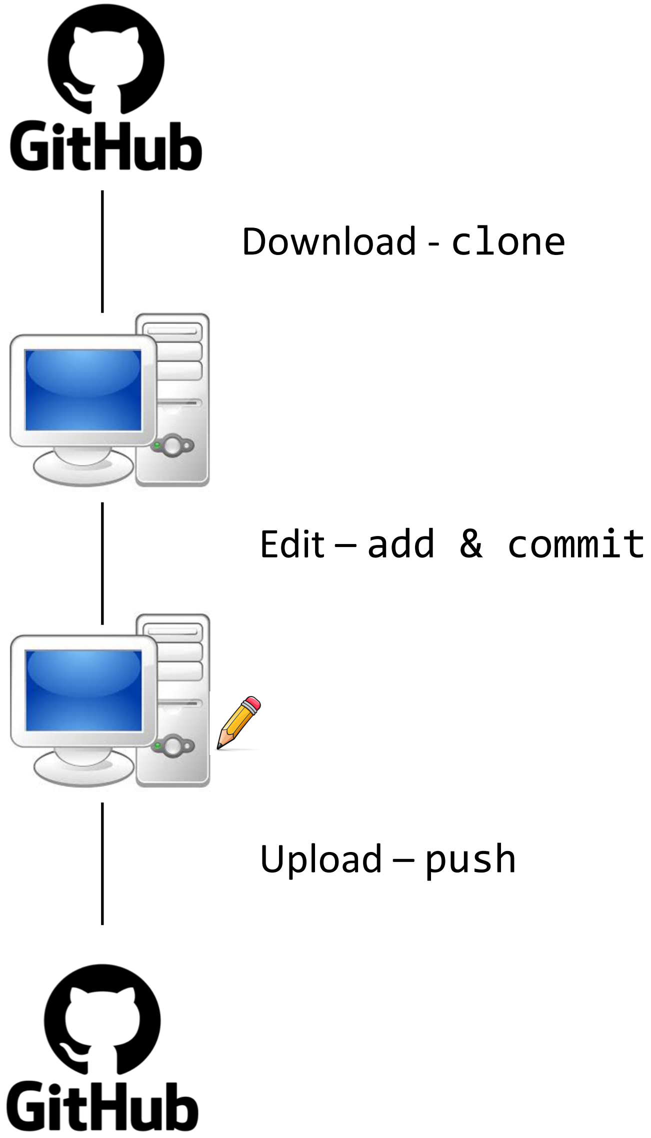
git

- Used to manage different versions edits
- It is a command line
- It is a free software installed locally
  - Mac and Linux installed by default

Installation_Git

Workflow

From GitHub to Local Machine



Download - `clone`

Edit – `add & commit`

Upload – `push`

git

- Proof GitHub your identity

1- Create ssh key

```
ssh-keygen -t rsa -b 4096 -C email_github_account
```

2- Two keys are generated

keyname -> private key. Keep secure in the local machine

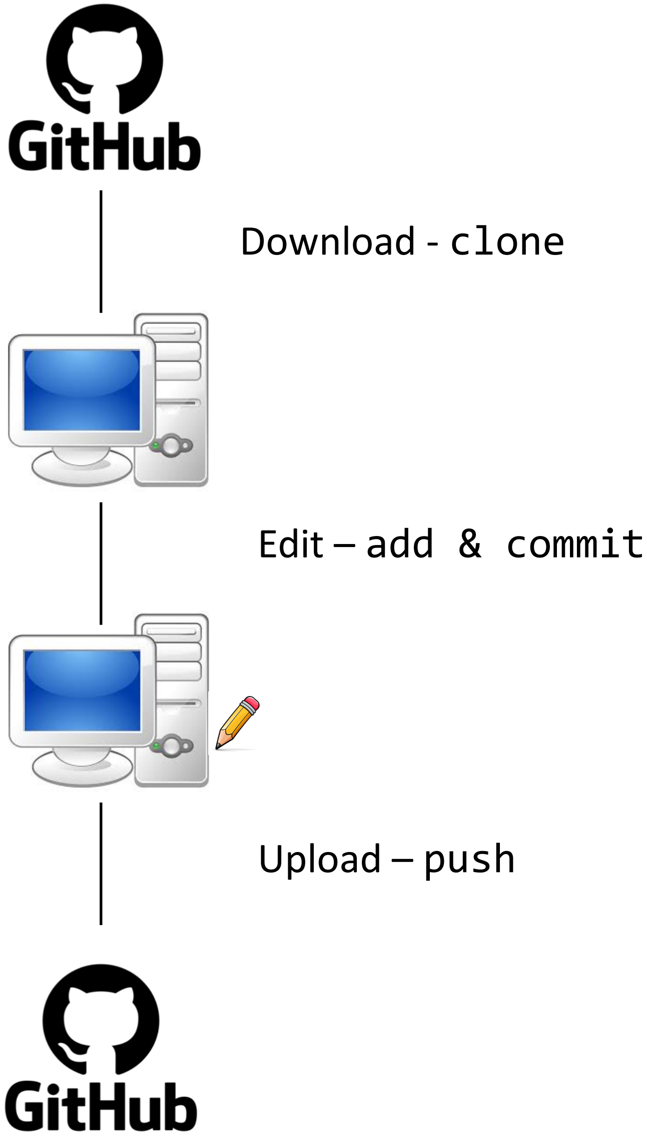
keyname.pub -> public key. Upload to GitHub

3- Local git know the key

```
eval "$(ssh-agent -s)"
ssh-add -K ~/.ssh/private_key
```

Workflow

From GitHub to Local Machine

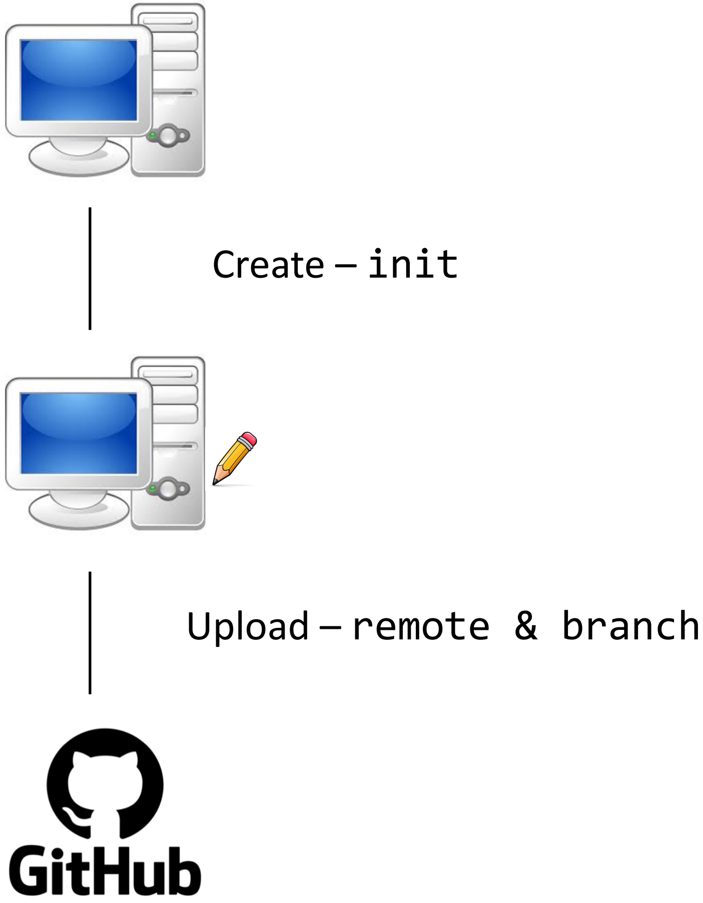


Download - clone

Edit – add & commit

Upload – push

From Local Machine to GitHub

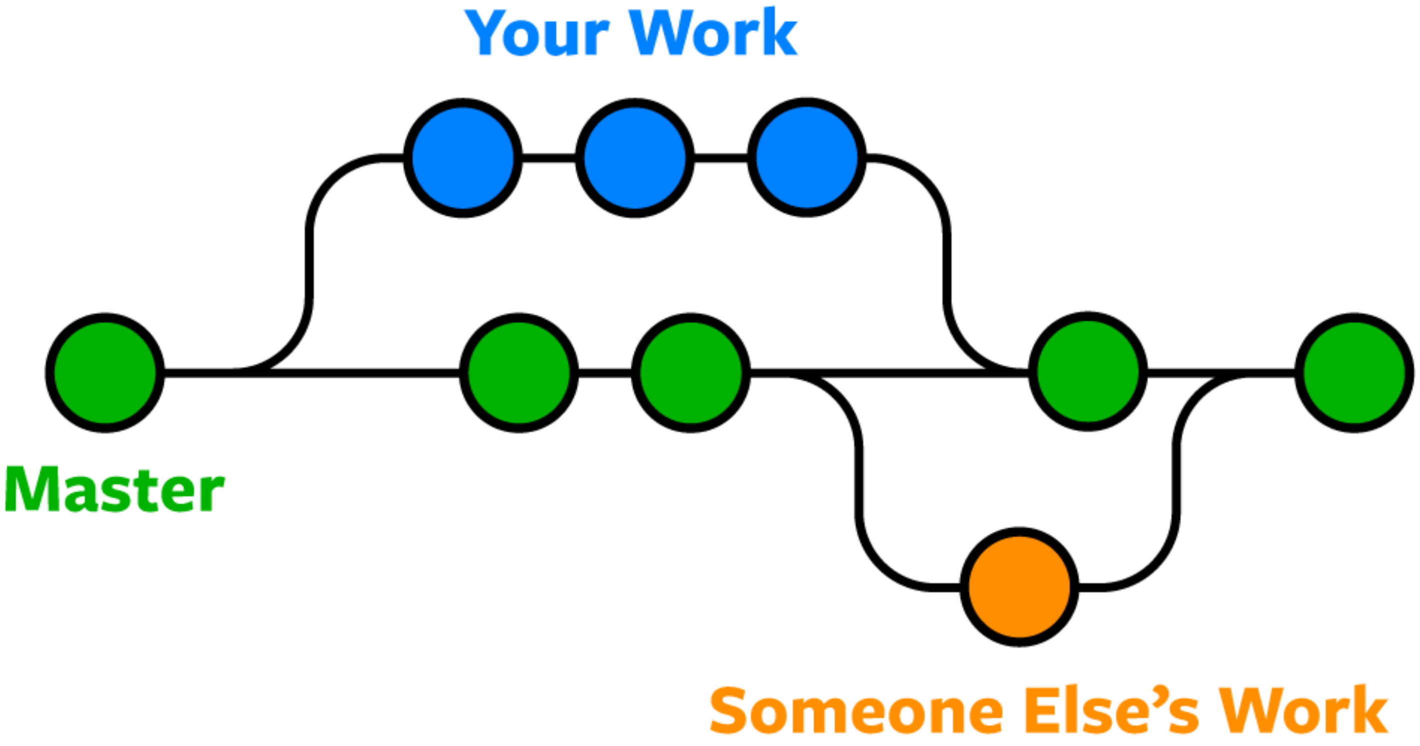


Create – init

Upload – remote & branch

git

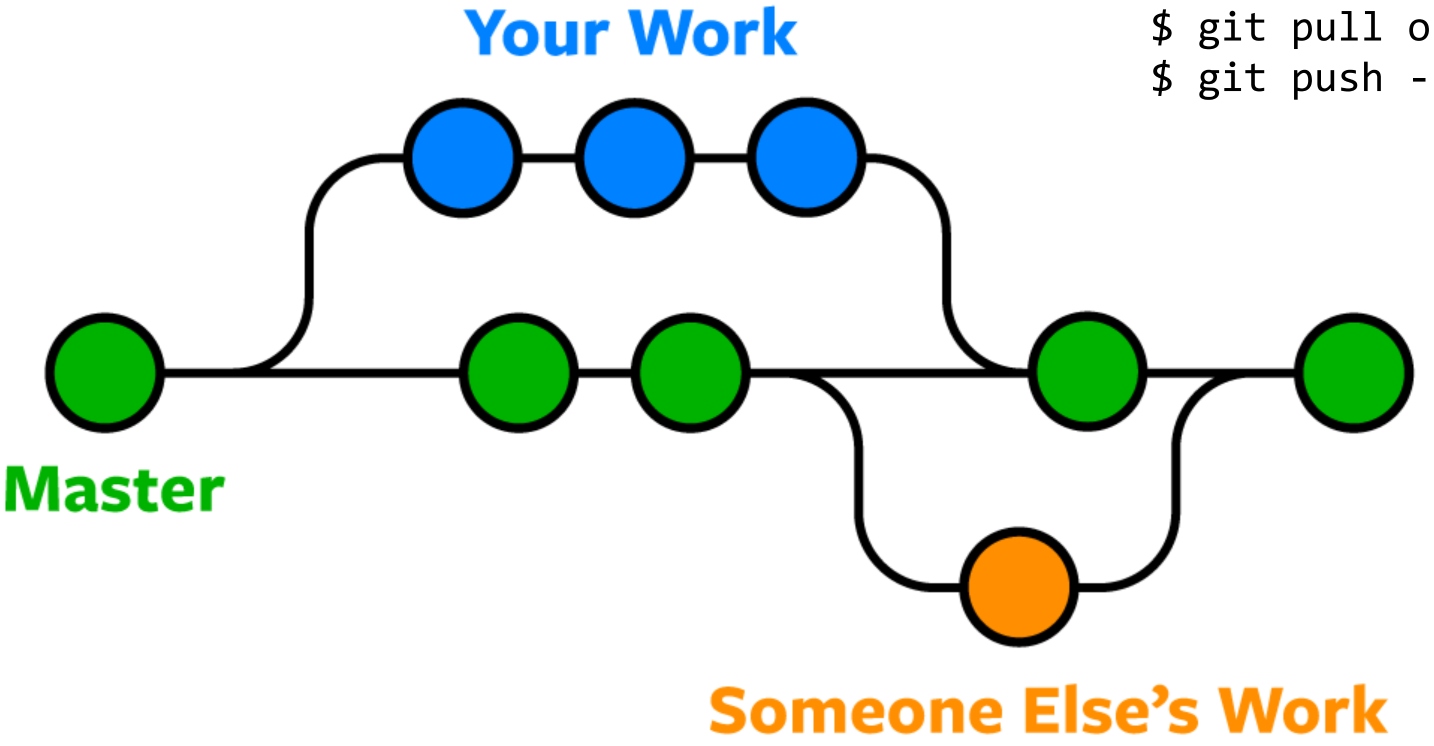
BRANCHING



BRANCHING

Command:

```
$ git branch  -> show branch
$ git checkout -b branch_name  -> create branch
$ git checkout -d branch_name  -> delete branch
$ git pull origin branch_name  -> from origin to branch
$ git push -u origin branch_name -> from branch to origin
```

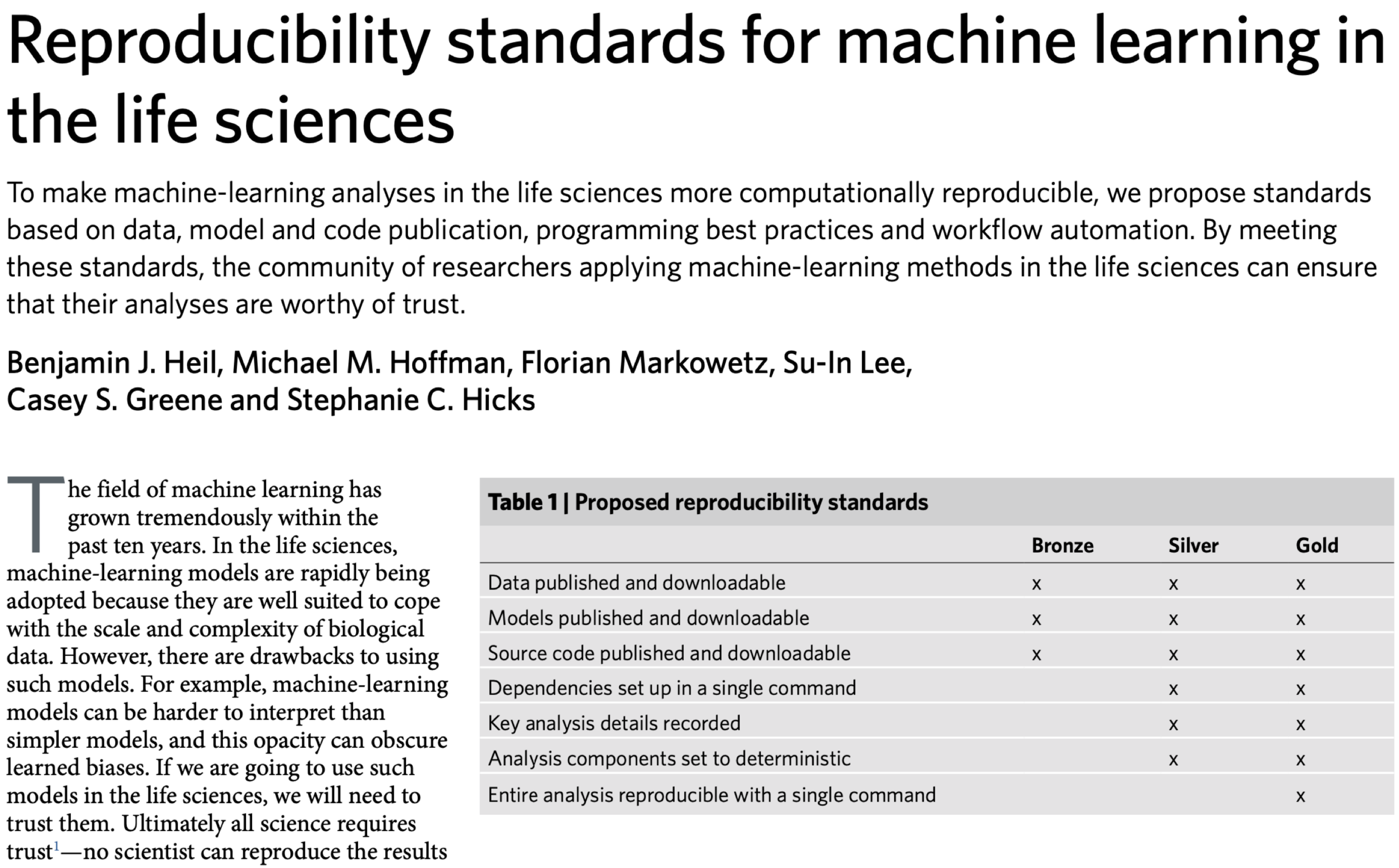


Text editor: Visual Studio Code
https://code.visualstudio.com/

- Command:

```
$ git clone -> bring a repository from GitHub to your local machine
$ git add -> track your files and changes in Git
$ git commit -> save your files
$ git status -> show all changes not saved yet
$ git push -> upload git commits to a GitHub
$ git init -> create a new git repository
$ git remote add -> assign/add remote repository
$ git log -> show all commits
$ git pull -> download changes from the remote repo to your local machine
$ git reset -> reset to a specific commit
```

2-Presentation for publication



Reproducibility standards for machine learning in the life sciences

To make machine-learning analyses in the life sciences more computationally reproducible, we propose standards based on data, model and code publication, programming best practices and workflow automation. By meeting these standards, the community of researchers applying machine-learning methods in the life sciences can ensure that their analyses are worthy of trust.

Benjamin J. Heil, Michael M. Hoffman, Florian Markowetz, Su-In Lee,
Casey S. Greene and Stephanie C. Hicks

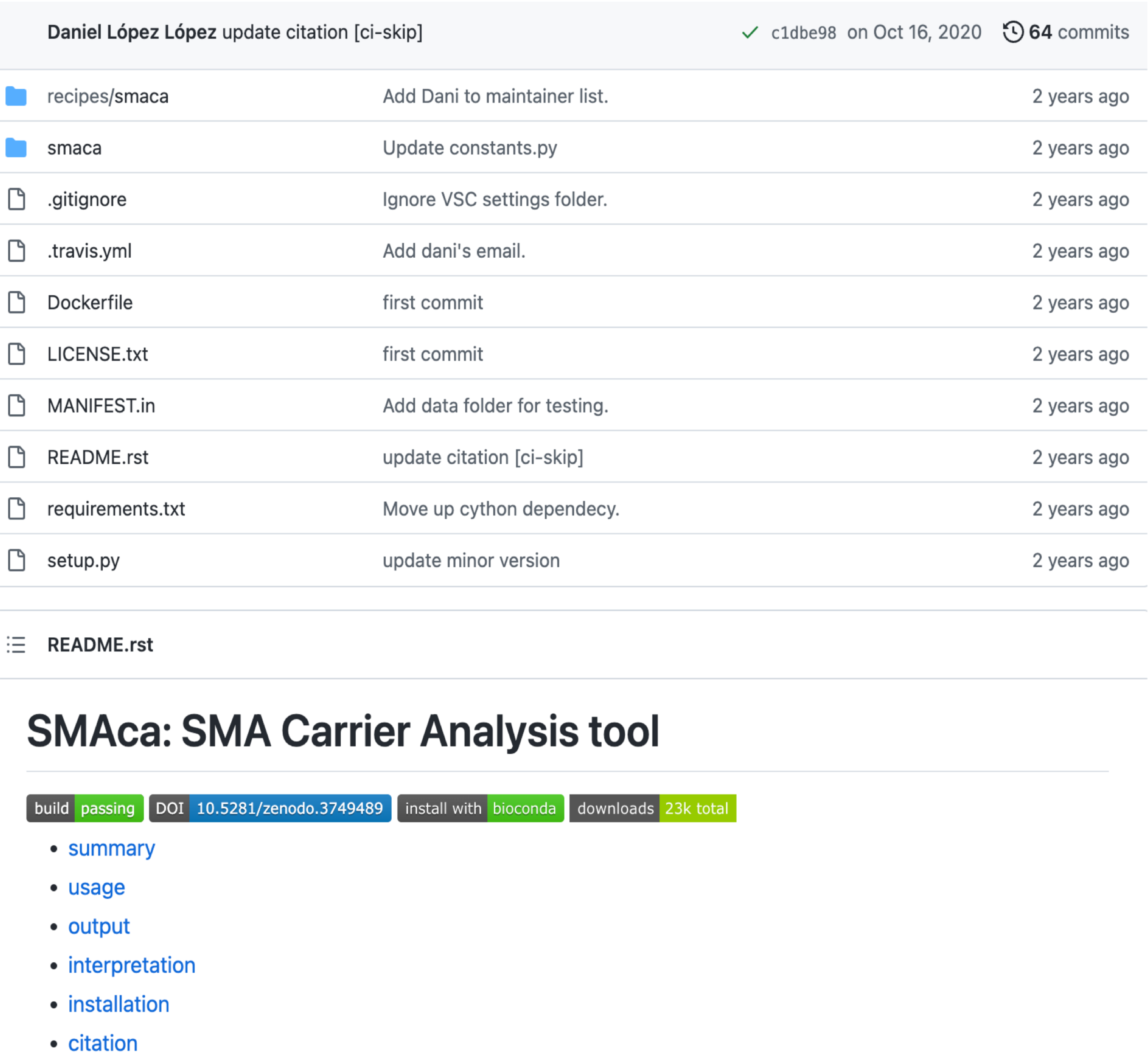
The field of machine learning has grown tremendously within the past ten years. In the life sciences, machine-learning models are rapidly being adopted because they are well suited to cope with the scale and complexity of biological data. However, there are drawbacks to using such models. For example, machine-learning models can be harder to interpret than simpler models, and this opacity can obscure learned biases. If we are going to use such models in the life sciences, we will need to trust them. Ultimately all science requires trust[1]—no scientist can reproduce the results

Table 1 | Proposed reproducibility standards

	Bronze	Silver	Gold
Data published and downloadable	x	x	x
Models published and downloadable	x	x	x
Source code published and downloadable	x	x	x
Dependencies set up in a single command		x	x
Key analysis details recorded		x	x
Analysis components set to deterministic		x	x
Entire analysis reproducible with a single command			x

GitHub Presentation

- Organized folder
- Descriptive names
- Structure
- Archiving in Zenodo
  - Provide a DOI



Daniel López López update citation [ci-skip] ✓ c1dbe98 on Oct 16, 2020 **64** commits

recipes/smaca	Add Dani to maintainer list.	2 years ago
smaca	Update constants.py	2 years ago
.gitignore	Ignore VSC settings folder.	2 years ago
.travis.yml	Add dani's email.	2 years ago
Dockerfile	first commit	2 years ago
LICENSE.txt	first commit	2 years ago
MANIFEST.in	Add data folder for testing.	2 years ago
README.rst	update citation [ci-skip]	2 years ago
requirements.txt	Move up cython dependecy.	2 years ago
setup.py	update minor version	2 years ago

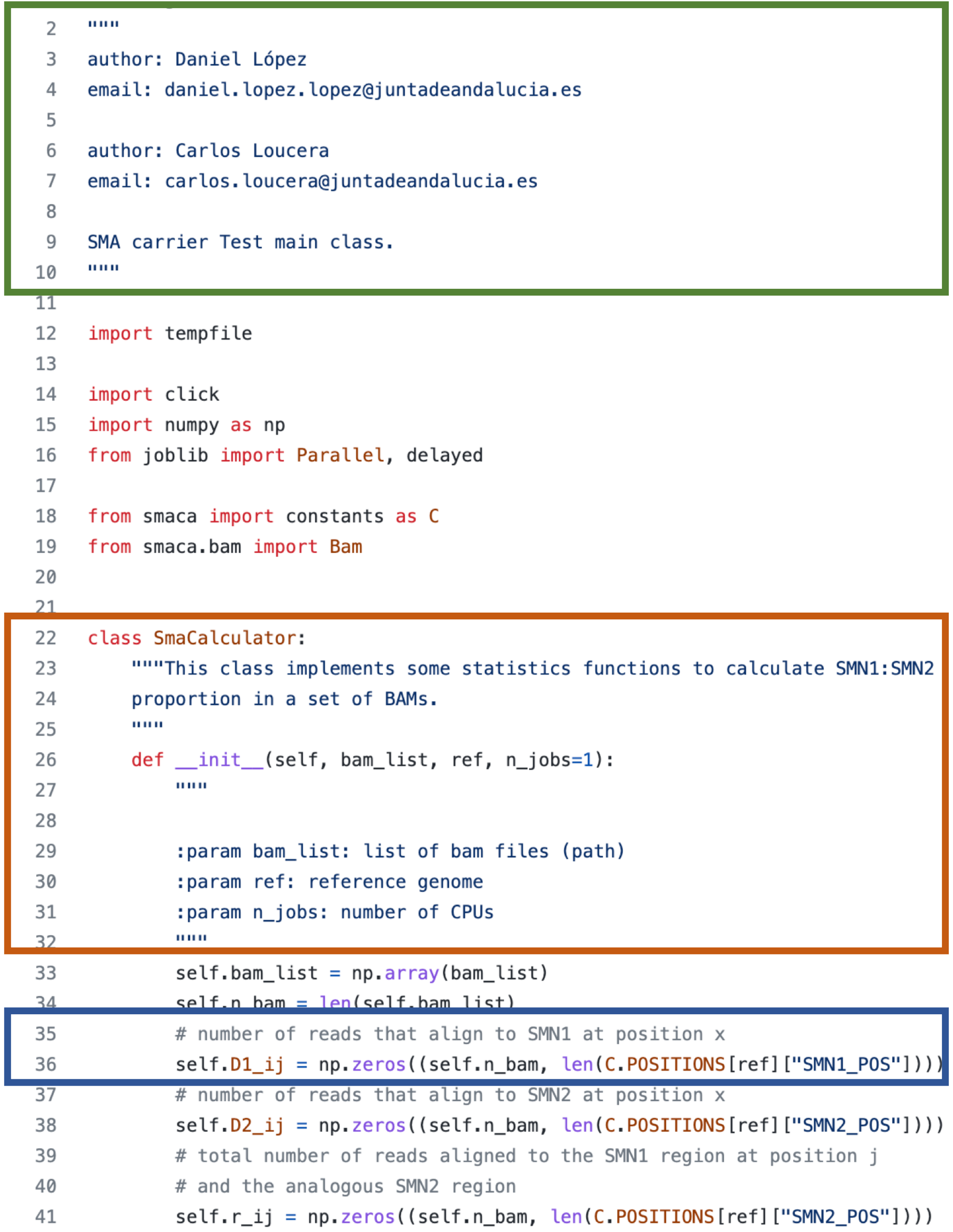
README.rst

SMAca: SMA Carrier Analysis tool

build passing | DOI 10.5281/zenodo.3749489 | install with bioconda | downloads 23k total

- summary
- usage
- output
- interpretation
- installation
- citation

1-Source code



```
"""
author: Daniel López
email: daniel.lopez.lopez@juntadeandalucia.es

author: Carlos Loucera
email: carlos.loucera@juntadeandalucia.es

SMA carrier Test main class.
"""

import tempfile

import click
import numpy as np
from joblib import Parallel, delayed

from smaca import constants as C
from smaca.bam import Bam


class SmaCalculator:
    """This class implements some statistics functions to calculate SMN1:SMN2
    proportion in a set of BAMs.
    """
    def __init__(self, bam_list, ref, n_jobs=1):
        """


        :param bam_list: list of bam files (path)
        :param ref: reference genome
        :param n_jobs: number of CPUs
        """
        self.bam_list = np.array(bam_list)
        self.n_bam = len(self.bam_list)
        # number of reads that align to SMN1 at position x
        self.D1_ij = np.zeros((self.n_bam, len(C.POSITIONS[ref]["SMN1_POS"])))
        # number of reads that align to SMN2 at position x
        self.D2_ij = np.zeros((self.n_bam, len(C.POSITIONS[ref]["SMN2_POS"])))
        # total number of reads aligned to the SMN1 region at position j
        # and the analogous SMN2 region
        self.r_ij = np.zeros((self.n_bam, len(C.POSITIONS[ref]["SMN2_POS"])))
```

Header

- Author name
- Author contact
- Date
- Introduction sentence

Function brief comment

Comments

*Avoid:

Debuging comments

No commented code

2-Data

- Fundamental to be published
- Added in a specialist repository
  - Gene Expression Omnibus (GEO)
  - European Genome Archive (EGA)
  - European Nucleotide Archive (ENA)
- Added in no specialist repository
  - Zenodo: datasets < 50GB
  - Dryad: datasets > 50GB
  - Figshare
- Public data: links

3-Models

- Critical for reproducibility
- Deposit in Zenodo repository
- Show/ running example

4-Installation/Dependencies

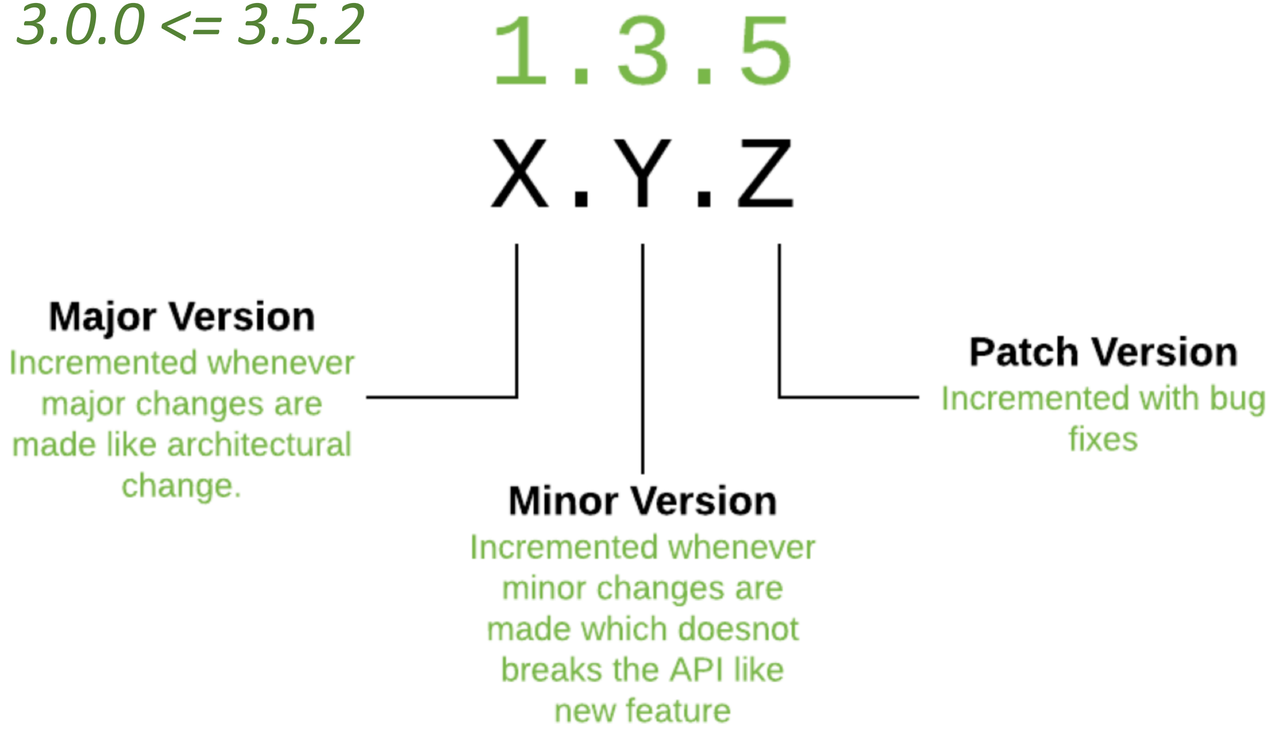
- In one single command -> eliminate all dependencies problem
- Dependency manage tools
  - *Conda* -> open environment system
  - *Packrat* -> dependency management system for R
  - Containerization
    - *Docker* -> integration of tools/packages in a container

4-Installation/Dependencies

- Software semantic version
  - *R version3 or above*
  - *R v3.4.1*
  - *Under R 3.5.2 or higher R 3.X.X*

4-Installation/Software

- Software semantic version
  - *R version3.1 or above -> R > 3.1.0*
  - *R v3.4.1 -> in tested R 3.4.1*
  - *Under R 3.5.2 or higher R 3.X.X -> R 3.0.0 <= 3.5.2*



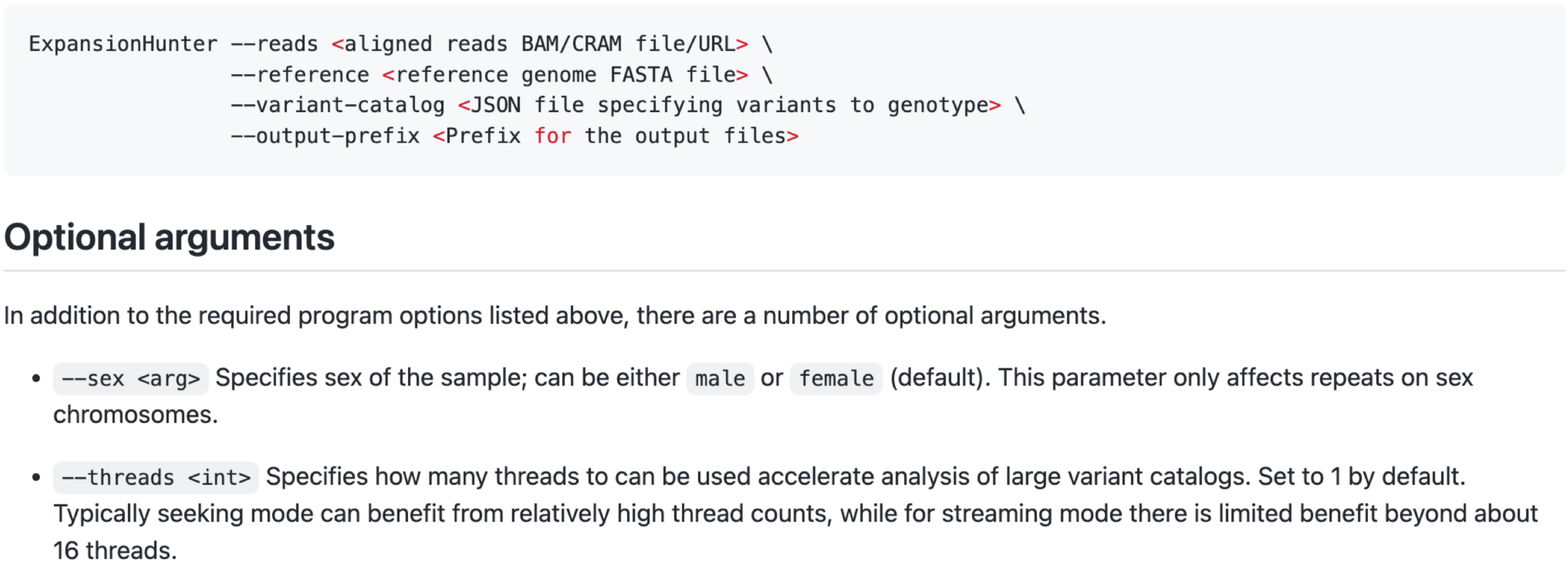
Major Version
Incremented whenever major changes are made like architectural change.

Minor Version
Incremented whenever minor changes are made which doesnot breaks the API like new feature

Patch Version
Incremented with bug fixes

5-Usage

- Guide
  - Commands/ functions
  - Arguments (specifying default values)
  - Scripts order
  - Python/R package



```
ExpansionHunter --reads <aligned reads BAM/CRAM file/URL> \
                --reference <reference genome FASTA file> \
                --variant-catalog <JSON file specifying variants to genotype> \
                --output-prefix <Prefix for the output files>
```

Optional arguments

In addition to the required program options listed above, there are a number of optional arguments.

- `--sex <arg>` Specifies sex of the sample; can be either `male` or `female` (default). This parameter only affects repeats on sex chromosomes.
- `--threads <int>` Specifies how many threads to can be used accelerate analysis of large variant catalogs. Set to 1 by default. Typically seeking mode can benefit from relatively high thread counts, while for streaming mode there is limited benefit beyond about 16 threads.

6-Other

- Computational requirements
  - Operating Systems compatibilities
  - RAM/CPU/GPU specification
- Coded checked using Travis CI
- Entire analysis in single command