Best Practices for Reproducible Research

Author: Oeslle Lucena

Reproducible research is a hot topic in the scientific community. Working towards reproducibility have gained more attention since some good reputational venues are requiring codes and data to publish works. This best practices tutorial intends to give you few tips on how to do a reproducible research based on the ideas of literate computing, documentation, workflow, version control, and infrastructure.

Here, it is described some do's and don't's for the following reproducible research tools: Jupyter Notebooks, Draw.io, Bitbucket and Github, Docker, and Amazon Web Services (AWS). The hints are based on my personal experience, therefore, there are many topics that will not be covered.

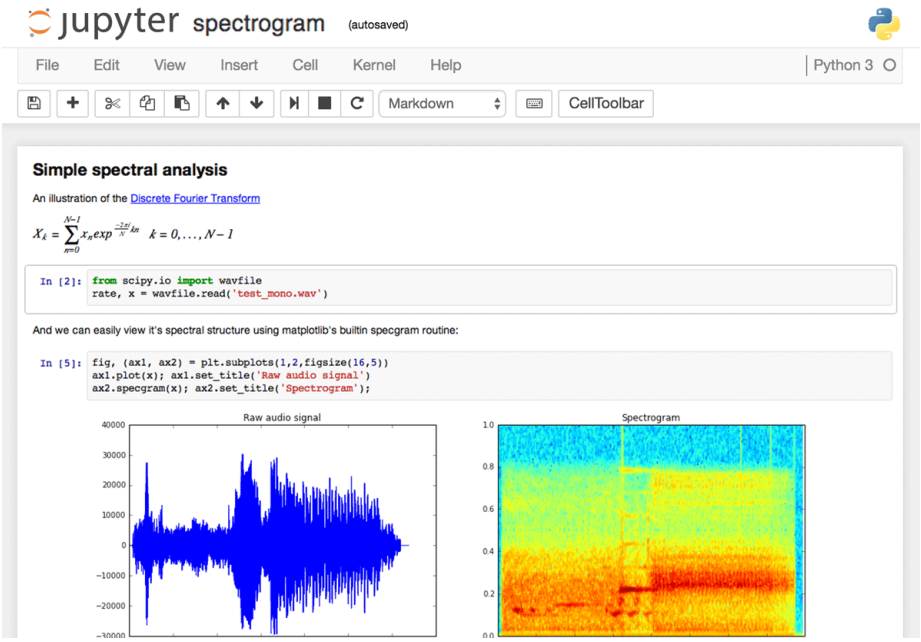
1 Literating computing and documentation

1.1 Jupyter Notebook

Jupyter Notebook is a very powerful web interface application that allows code, text, figures, etc, in a "notebook" format link. In this tool, it is possible to write documentation in markdown or latex format and run some code in the same notebook. The hints here described are for jupyter notebooks using python as a code language. Below it is shown a figure of how jupyter notebook looks like.

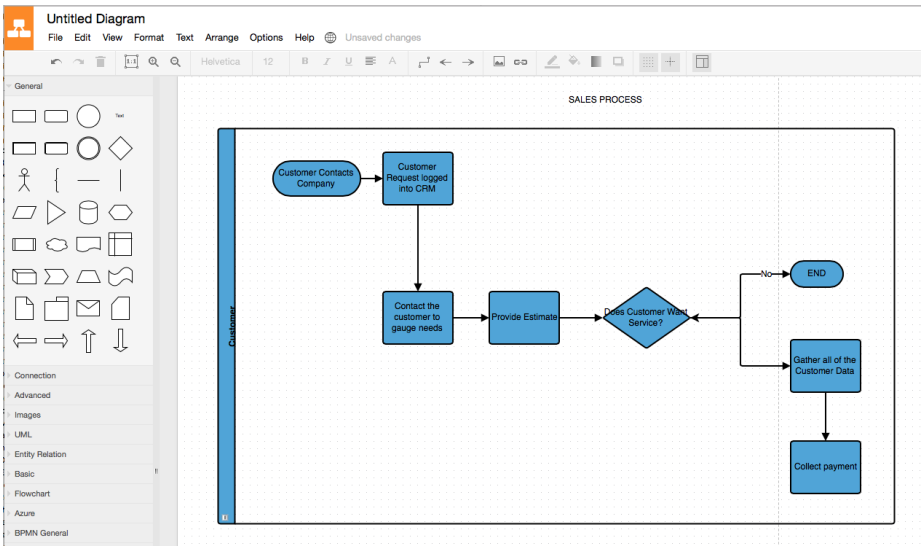
1.1.1 Do's and Don't's

- You can install the anaconda link, which comes with jupyter already set and other libraries. Alternatively, you can install the jupyter notebook following the official documentation link
- When you start a new project, at first, I strongly recommend you to create few directories that will help you to organize your ideas and make it easier



for version control. For instance, create folders like src, dev, deliver for libraries, development notebooks, and deliverable notebooks, respectively, likewise in this guide. Also, save each notebook in a standard format, followed by data, initials, and a short description, for example, 2017-05-05-OL-start.ipynb.

- Document the jupyter notebook with short markdown descriptions.
- Importing notebooks as modules are bit tricky. You can follow the official documentation and use some classes to do that for you link, which is very handy. However, I personally recommend you first to convert to .py, using the nbconvert (`jupyter nbconvert --to python xxxx.ipynb`), and then import the notebook as a python module. The advantage of the second method is the speed. Also, importing using the first way forces the notebook module to be in the same folder of the main notebook.
- Don't update a module notebook once it was already imported in another notebook. Unless you restart the main notebook, the modification will not be considered. As a safety rule once started with and version of the module, that version will be considered along the script with no changes.
- Be careful with data usage! Jupyter notebooks save every variable in the memory from each cell you run. Therefore, if you are dealing with big data, I strongly recommend you to define functions in the cells and call them instead of plain coding.



2 Workflow

Here I refer to workflow as a simple diagram that represents a code pipeline.

2.1 Draw.io

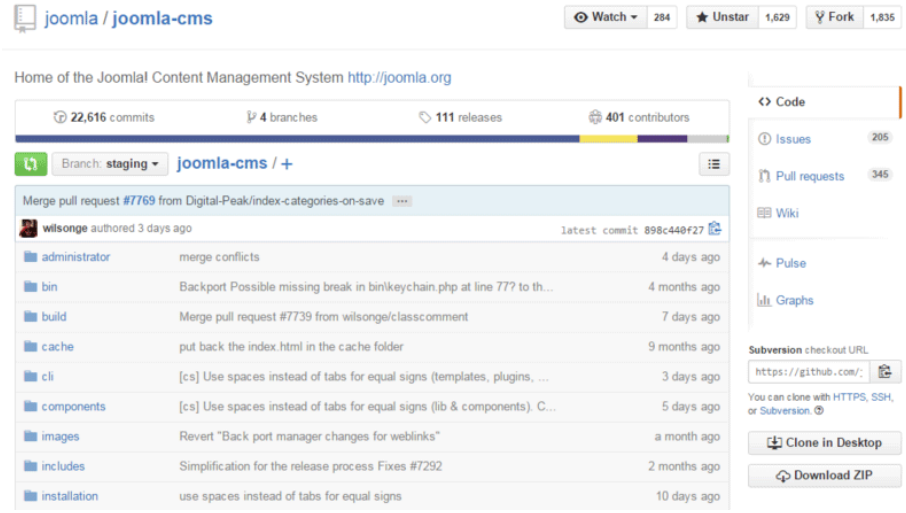
This tool is a completely free online diagram editor. It allows designing from simple block diagrams to complex circuits diagram. Moreover, the tool is linked to Google drive and Github link. Below it is shown a figure of how Draw.io looks like.

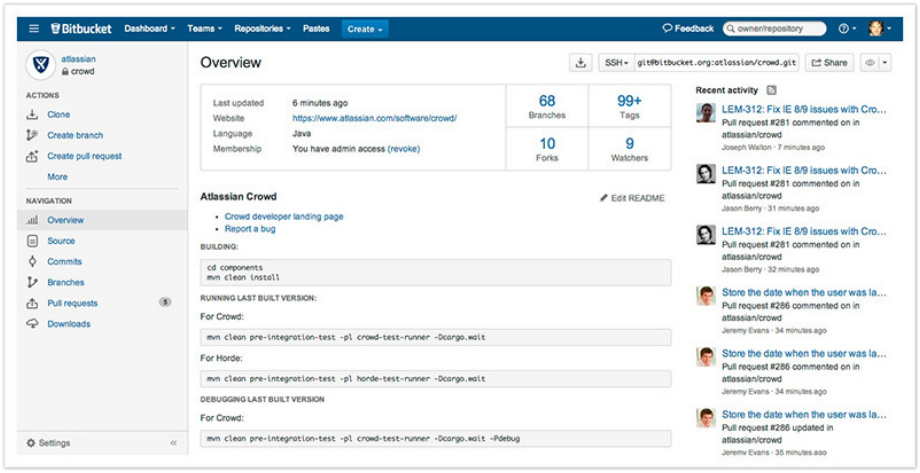
2.1.1 Do's and Don't's

- Unless you cannot do it, I always recommend you export your workflow as a vectorized image (SVG or PDF formats). These formats have better quality than usual png or jpeg files, and they will not get depreciated in case you have to reshape them.
- Always export with a transparent background. Doing that, you will have no white squares in coloured backgrounds.

3 Version control

There are a bunch of version control tools. However, I strongly recommend two: Bitbucket and Github. The both tools are well known in the coders community and are constantly getting updates. Also, both tools allows you to visualize jupyter notebooks, making it easier to get shareable jupyter projects. Below it is shown figures of how Github and Bitbucket look like.


joomla / joomla-cms
Watch 284
Unstar 1,629
Fork 1,835
Home of the Joomla! Content Management System http://joomla.org
22,616 commits
4 branches
111 releases
401 contributors
Branch: staging
joomla-cms / +
Merge pull request #7769 from Digital-Peak/index-categories-on-save
wilsonge authored 3 days ago
latest commit 898c440f27
administrator merge conflicts 4 days ago
bin Backport Possible missing break in bin\keychain.php at line 77? to th... 4 months ago
build Merge pull request #7739 from wilsonge/classcomment 7 days ago
cache put back the index.html in the cache folder 9 months ago
cli [cs] Use spaces instead of tabs for equal signs (templates, plugins, ... 3 days ago
components [cs] Use spaces instead of tabs for equal signs (lib & components). C... 5 days ago
images Revert "Back port manager changes for weblinks" a month ago
includes Simplification for the release process Fixes #7292 2 months ago
installation use spaces instead of tabs for equal signs 10 days ago
Code
Issues 205
Pull requests 345
Wiki
Pulse
Graphs
Subversion checkout URL
https://github.com/
You can clone with HTTPS, SSH, or Subversion.
Clone in Desktop
Download ZIP


Bitbucket
Dashboard
Teams
Repositories
Pastes
Create
Feedback
owner/repository
atlassian
crowd
ACTIONS
Clone
Create branch
Create pull request
More
NAVIGATION
Overview
Source
Commits
Branches
Pull requests
Downloads
Settings
Overview
SSH
git@bitbucket.org:atlassian/crowd.git
Share
Last updated 6 minutes ago
Website https://www.atlassian.com/software/crowd/
Language Java
Membership You have admin access (revoke)
68 Branches
99+ Tags
10 Forks
9 Watchers
Atlassian Crowd
Edit README
Crowd developer landing page
Report a bug
BUILDING:
cd components
mvn clean install
RUNNING LAST BUILT VERSION:
For Crowd:
mvn clean pre-integration-test -pl crowd-test-runner -Dcargo.wait
For Horde:
mvn clean pre-integration-test -pl horde-test-runner -Dcargo.wait
DEBUGGING LAST BUILT VERSION
For Crowd:
mvn clean pre-integration-test -pl crowd-test-runner -Dcargo.wait -Pdebug
Recent activity
LEM-312: Fix IE 8/9 issues with Cro...
Pull request #281 commented on in atlassian/crowd
Joseph Walton · 7 minutes ago
LEM-312: Fix IE 8/9 issues with Cro...
Pull request #281 commented on in atlassian/crowd
Jason Berry · 31 minutes ago
LEM-312: Fix IE 8/9 issues with Cro...
Pull request #281 commented on in atlassian/crowd
Jason Berry · 32 minutes ago
Store the date when the user was la...
Pull request #286 commented on in atlassian/crowd
Jeremy Evans · 34 minutes ago
Store the date when the user was la...
Pull request #286 commented on in atlassian/crowd
Jeremy Evans · 34 minutes ago
Store the date when the user was la...
Pull request #286 updated in atlassian/crowd
Jeremy Evans · 35 minutes ago

3.0.1 Do's and Don't's

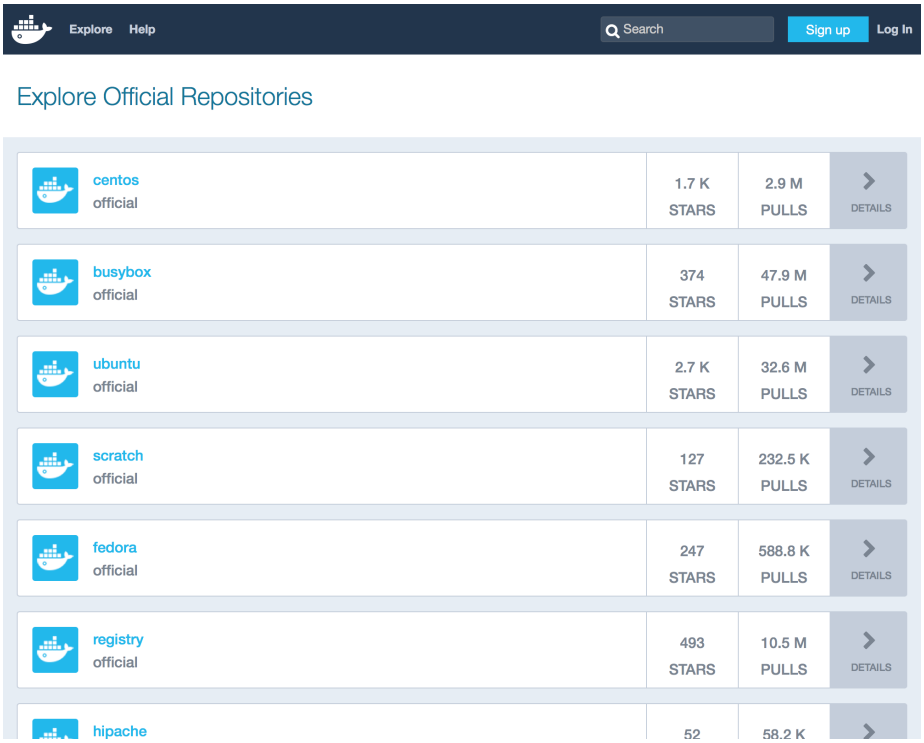
- If you do not have a problem in showing to the community each step of your project, I would recommend you to use Github. It is more user-friendly than Bitbucket, and it has more people working on the platform.
- If you want to have private projects, I would recommend you to use Bitbucket, because it allows you to create private projects for free. On Github, private projects are paid, so, unless you want to spend some money on that, stick with Bitbucket.
- Both platforms use the git commands, and they are sometimes very tricky. All the merging, branching, pushing, and pulling can be very hard to understand. These git tips are very useful; this website demystifies many things about git.
- Be careful! In both platforms, you are allowed to see your jupyter notebooks online. However, in private projects that is not allowed. Both Github and Bitbucket invocates nbviewer to show jupyter notebooks on the web browser and private projects blocks that operation.
- To see your jupyter notebooks on Bitbucket you to do some work, which is a weakness compared to Github. Originally, Bitbucket does not invocates nbviewer, but, fortunately to our coders colleagues, there are some add-ons for mozilla and google chrome.
- Mostly of the markdown commands are the same for Github and Bitbucket. However, adding figures and links are bit tricky on Github than in Bitbucket, a good list of those commands are found here.
- It always good to add the workflow on the README.md file. A good tip is to use Draw.io to create your workflow, tagging the notebook for each step of your project. An example of that is shown below.

4 Infrastructure

Here I refer to infrastructure as the hardware that you need to run your projects and the "wraps" for virtual machines that guarantee reproducibility in any workstation.

4.1 Docker

As the creators say "Docker is the world's leading software container platform. Developers use Docker to eliminate "works on my machine" problems when collaborating on code with co-workers." Docker is a platform that promotes light virtualization in the container format, working for any OS (see more details in the official website). Below it is shown a figure of how dokerhub (docker images repository) look like.


Explore
Help
Search
Sign up
Log In
Explore Official Repositories
centos
official
1.7 K
STARS
2.9 M
PULLS
DETAILS
busybox
official
374
STARS
47.9 M
PULLS
DETAILS
ubuntu
official
2.7 K
STARS
32.6 M
PULLS
DETAILS
scratch
official
127
STARS
232.5 K
PULLS
DETAILS
fedora
official
247
STARS
588.8 K
PULLS
DETAILS
registry
official
493
STARS
10.5 M
PULLS
DETAILS
hipache
52
58.2 K

4.1.1 Do's and Don't's

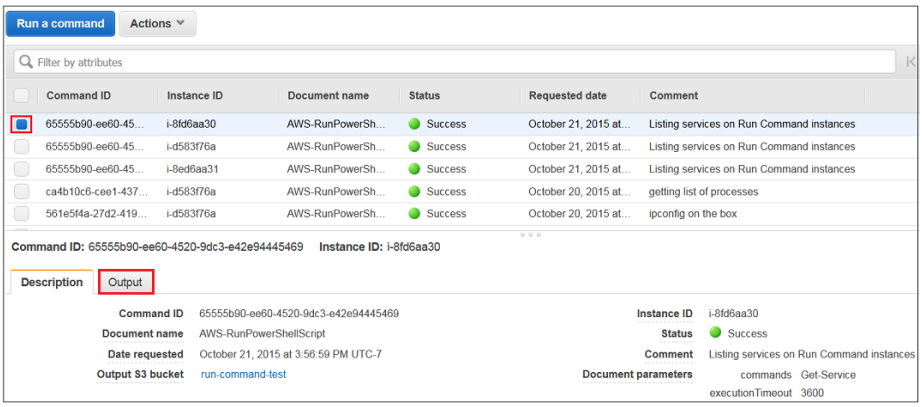
- Unless you really need to start from scratch, I strongly recommend you to pick a docker image on the docker hub and modify for your own purposes.
- Docker is a lightweight platform, which is its main advantages. Therefore, it is recommended to do not add data in docker images. If you do so, you are destroying the "lightweightness" from the docker image. Use any other tool to share your data.
- Docker is command line based, consequently not being user-friendly. Therefore, it is very important to use Docker options carefully. You may be running Docker as a background process not knowing. Hence, no copy and paste any command you see on the internet. Spend a bit of time trying to understand the Docker command line options. The official documentation and this tutorial website are good ways to start.
- Docker allows you sharing folders with the host OS, thus, you can put your data on your machine and share that path with the docker container. The docker option responsible for that is "-v".
- Any changes in the Docker container will be lost after its closing. That confirms the lightweightness of the container. So, if you really want to keep the changes, commit them before finish the process.
- Docker commits are always done in another terminal, keeping the running application open. Do no try to commit inside the application; it will not work.

4.2 Amazon Web Services (AWS)

As the creators say "Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides resizable computing capacity—literally, servers in Amazon's data centers—that you use to build and host your software systems". The AWS for cloud computing (EC2) is composed of paid servers that can be used by anyone. If you lack good hardware for your project or to reproduce one, the AWS machines are always a good start. They also have not paid machines too, but obviously with low capacity. Below it is shown a figure of how AWS EC2 instances interface look like.

4.2.1 Do's and Don't's

- AWS EC2 machines are very practical and easy to use. The worst part of the service is to create an account. Therefore, I strongly recommend you to follow all the steps in the getting start tutorial.
- Create an instance on AWS EC2 machines requires some steps. A good tutorial for that is this website. It has many figures explaining each step, just follow it.



- AWS EC2 allows to either use a Java interface or Linux command lines to access the machine. I personally used only the Linux command lines and had no problems with that. It is very straightforward. Once you launch a instance, click in connect and you have both command options to access your machine.
- AWS EC2 virtualizes hardware in its servers. You can find the best in the community to suit your personal interests. Also, you can create your virtualization using a docker upon the AWS machines virtualization, for instance. The official documentation has a good guide for that link.
- AWS EC2 charges the usage of its machines per hour, with different prices depending on the configuration. Therefore, if you are not using the machine, just stop it. The changes and saved data will not be lost. The only thing that changes is the IP address to access that machine.
- Be careful! Only terminate your instance when you will not using anymore. You are not stopping the machine; you are definitely ending your instance.
- Due to its elasticity, it is possible on AWS EC2 instances to change the configuration of your instance. Just need to stop the machine and then go to actions, instance settings, and click on change instance type.
- Data storage could not be changed after setting. Thus, before setting the default storage (80 Gb), spend some time thinking how much you will really need.