



MAX PLANCK INSTITUTE FOR **BIOLOGY OF AGEING**



# **Reproducible multilang workflows with Jupyter on Docker**

# Computers are not all the same



Created by Mike O'Brien  
from Noun Project



Created by Mister Pixel  
from Noun Project



Created by factor[e] design initiative  
from Noun Project

# Computers are not all the same



Created by Mike O'Brien  
from Noun Project

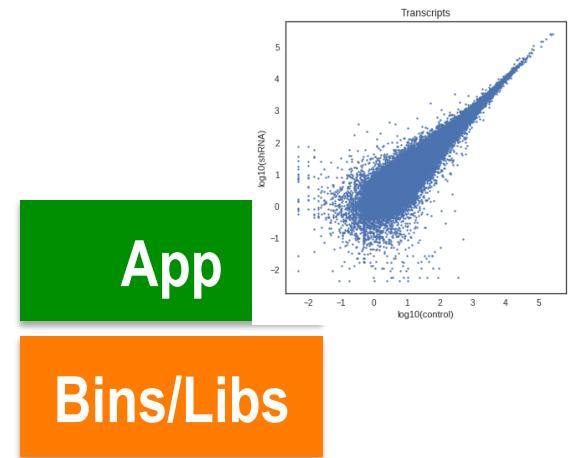
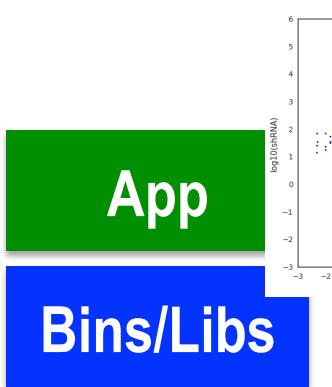


Created by Mister Pixel  
from Noun Project



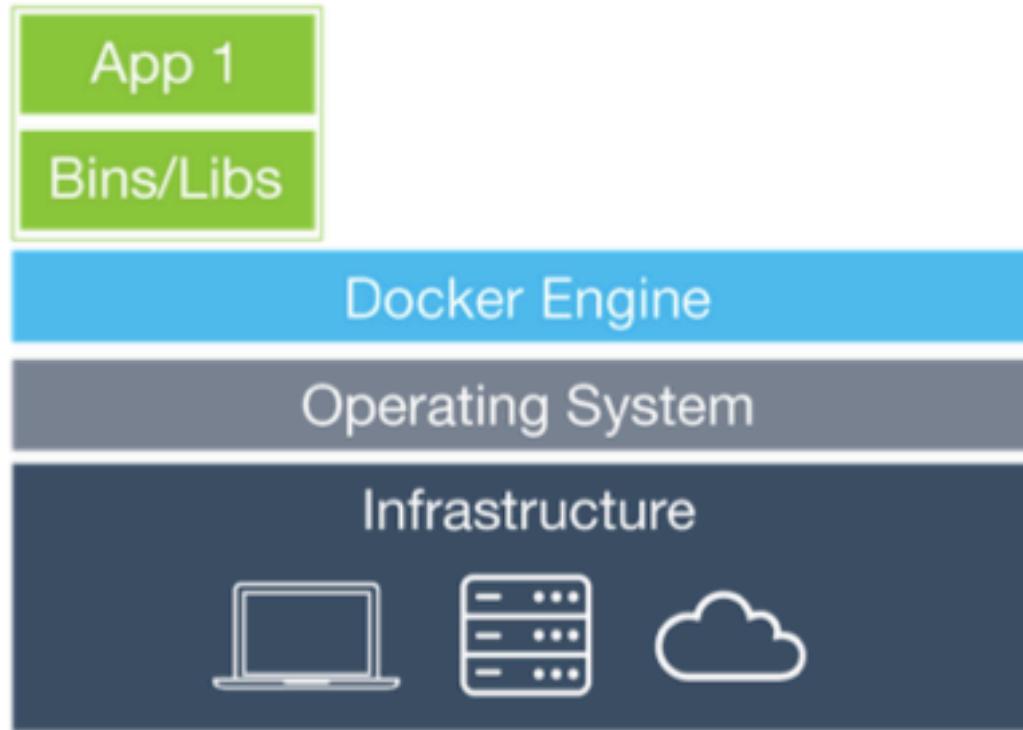
Created by factor[e] design initiative  
from Noun Project

# The same apparent code can give different results



Different version of dependency in bin/  
== different input in one step  
== code breaks

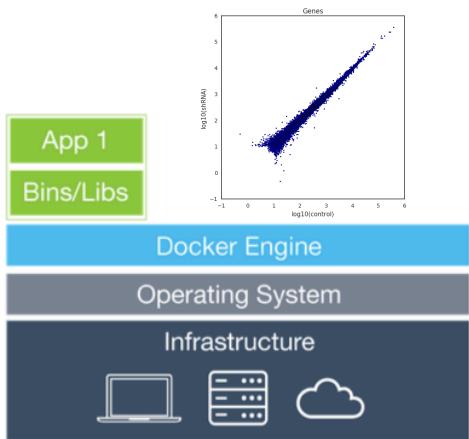
# Containers



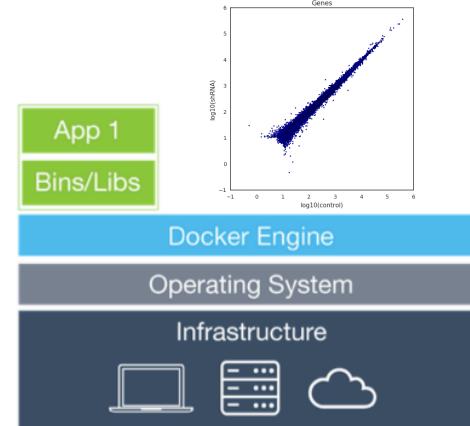
## Containers

<https://blog.logentries.com/2015/07/an-all-inclusive-log-monitoring-container-for-docker/>

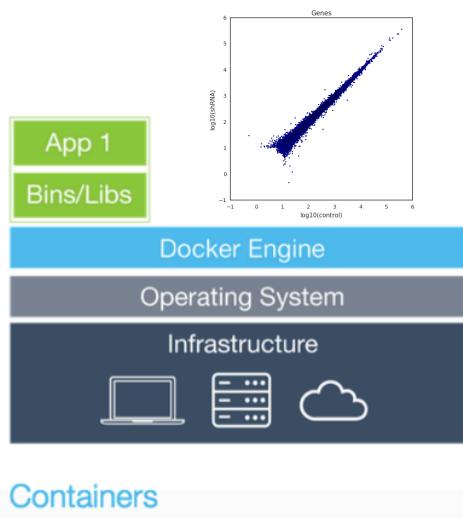
# Containers are independent of their host



Containers

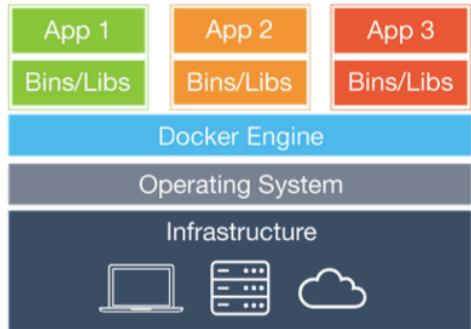


Containers

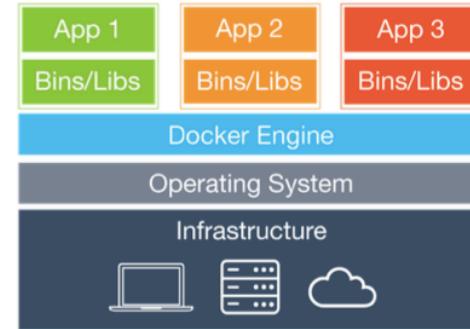


Containers

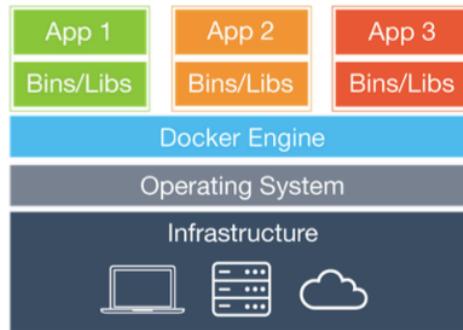
# Multiple environments per machine



Containers



Containers



Containers

<https://store.docker.com>

The screenshot shows the Docker Store interface. At the top, there's a navigation bar with the Docker logo, a search bar containing 'debian', and links for 'Explore', 'Publish', and 'Feedback?'. A user profile for 'jorgeboucas' is also visible. The main title 'The Docker Store' is centered above the subtitle 'Find Trusted and Enterprise Ready Containers'. On the left, there are filters for 'Type' (Store selected, Community available) and 'Categories' (Analytics, Application Frameworks, Application Infrastructure, Application Services, Base Images, Databases, DevOps Tools, Featured Images, Messaging Services, Monitoring, Operating Systems, Programming Languages, Storage). The search results for 'debian' show three items: 'debian' (Docker Official Image), 'ubuntu' (Docker Official Image), and 'neurodebian'. Each item has a thumbnail, the name, 'Docker Official Image | 10M+ Pulls', a brief description, and links to 'Base Images' and 'Operating Systems'. The results are sorted by 'Most Popular'.

Type

Store

Community

Categories

Analytics

Application Frameworks

Application Infrastructure

Application Services

Base Images

Databases

DevOps Tools

Featured Images

Messaging Services

Monitoring

Operating Systems

Programming Languages

Storage

1 - 3 of 3 results for **debian**. [Clear search](#)

**Most Popular**

**debian**  
Docker Official Image | 10M+ Pulls

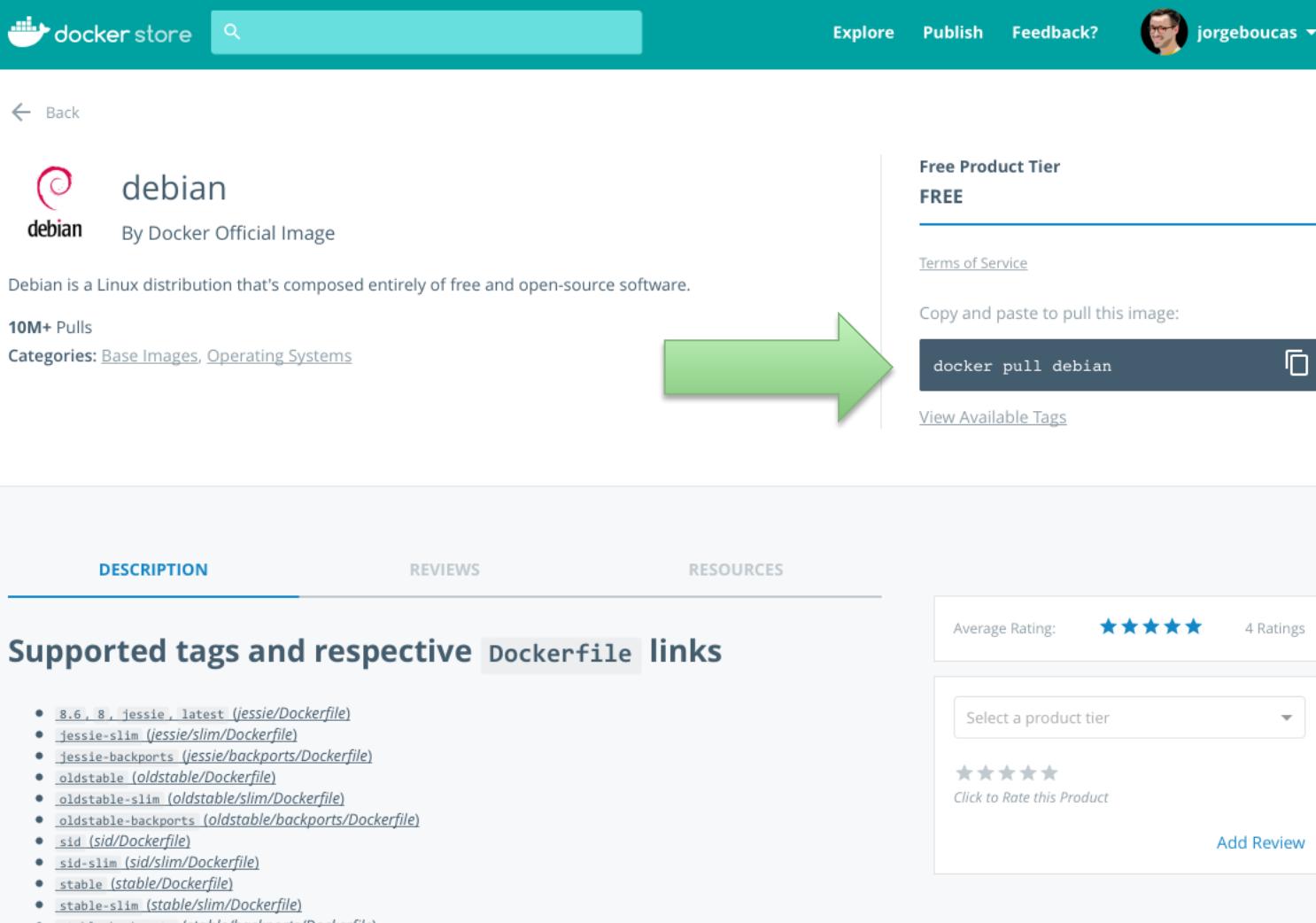
Debian is a Linux distribution that's composed entirely of free and open-source software.  
[Base Images](#), [Operating Systems](#)

**ubuntu**  
Docker Official Image | 10M+ Pulls

Ubuntu is a Debian-based Linux operating system based on free software.  
[Base Images](#), [Operating Systems](#)

**neurodebian**

# docker pull debian



The screenshot shows the Docker Store interface for the 'debian' image. At the top, there's a navigation bar with 'Explore', 'Publish', 'Feedback?', and a user profile for 'jorgeboucas'. Below the navigation is a search bar and a back button. The main content area features the 'debian' image card, which includes the Docker logo, the name 'debian', the tag 'By Docker Official Image', a brief description stating 'Debian is a Linux distribution that's composed entirely of free and open-source software.', and metrics like '10M+ Pulls'. It also lists categories: 'Base Images, Operating Systems'. To the right of the card, under the heading 'Free Product Tier FREE', are links for 'Terms of Service' and a 'Copy and paste to pull this image:' button containing the command 'docker pull debian'. A large green arrow points from the left towards this command. Below the card, there are tabs for 'DESCRIPTION', 'REVIEWS', and 'RESOURCES'. The 'DESCRIPTION' tab is active, showing a section titled 'Supported tags and respective Dockerfile links' with a list of tags: 8.6, 8, jessie, latest (jessie/Dockerfile), jessie-slim (jessie/slim/Dockerfile), jessie-backports (jessie/backports/Dockerfile), oldstable (oldstable/Dockerfile), oldstable-slim (oldstable/slim/Dockerfile), oldstable-backports (oldstable/backports/Dockerfile), sid (sid/Dockerfile), sid-slim (sid/slim/Dockerfile), stable (stable/Dockerfile), stable-slim (stable/slim/Dockerfile), and stable-backports (stable/backports/Dockerfile). The 'REVIEWS' tab shows an average rating of 5 stars from 4 ratings, with a 'Select a product tier' dropdown and a 'Click to Rate this Product' button. The 'RESOURCES' tab is currently empty.

# running a container

```
## pull an image from the docker store

jboucas-mbpr:~$ docker pull debian

Using default tag: latest
latest: Pulling from library/debian
5040bd298390: Pull complete
Digest: sha256:abbe80c8c87b7e1f652fe5e99ff1799cdf9e0878c7009035afe1bccac129cad8
Status: Downloaded newer image for debian:latest

## start a container from the downloaded image
## and initiate an interactice bash session

jboucas-mbpr:~$ docker run --name my_first_container -t -i debian /bin/bash

root@43530985a83d:/# pwd
/
root@43530985a83d:/# ls
bin  boot  dev  etc  home  lib  lib64  media  mnt  opt  proc  root  run  sbin
srv  sys  tmp  usr  var
```

# Working on a container

```
## change the content of the container

root@43530985a83d:/# touch jorge.txt
root@43530985a83d:/# ls
bin  boot  dev  etc  home  jorge.txt  lib  lib64  media  mnt  opt  proc  root
run  sbin  srv  sys  tmp  usr  var
root@43530985a83d:/# exit
exit

## list containers

jboucas-mbpr:~$ docker ps -a

CONTAINER ID        IMAGE       COMMAND                  CREATED             STATUS              NAMES
43530985a83d        debian      "/bin/bash"               58 minutes ago    Exited (0) 3 seconds ago   my_first_container
c3ad2c600b24        encode_rnaseq_eclip "tini -- start-not..."   10 days ago     Up 3 days          0.0.0.0:8888->8888/tcp  encode_rnaseq_eclip
```

# Restarting a container

```
## start a container

jboucas-mbpr:~$ docker start my_first_container
my_first_container

## start an interactive bash session on a running container

jboucas-mbpr:~$ docker exec -i -t my_first_container /bin/bash

root@43530985a83d:/# ls
bin  boot  dev  etc  home  jorge.txt  lib  lib64  media  mnt  opt  proc  root
run  sbin  srv  sys  tmp  usr  var
root@43530985a83d:/# exit
exit

## stop a container

jboucas-mbpr:~$ docker stop my_first_container
my_first_container
```

# Listing containers and images

```
## list containers

jboucas-mbpr:~$ docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED            NAMES
STATUS              PORTS              NAMES
43530985a83d        debian              "/bin/bash"         About an
hour ago           Exited (0) 36 seconds ago
my_first_container
c3ad2c600b24        encode_rnaseq_eclip   "tini -- start-not..."  10 days ago
Up 3 days          0.0.0.0:8888->8888/tcp    encode_rnaseq_eclip

## list images

jboucas-mbpr:~$ docker images
REPOSITORY          TAG                 IMAGE ID
CREATED             SIZE
debian              latest              e5599115b6a6
6 days ago          123 MB
encode_rnaseq_eclip latest              40aed2032ff1
10 days ago         4.02 GB
mpgabioinformatics/encode_rnaseq_eclip latest              8a22f9e82bf6
8 weeks ago         4.02 GB
```

# from container to image to cloud

```
## commit a containter into an image

jboucas-mbpr:~$ docker commit my_first_container my_first_image
sha256:686efbb118cdef3a5cda0413a684ab77c5d67645f37c85e7450212fc5d406575

## tag, push and pull your image

jboucas-mbpr:~$ docker tag 686efbb118cd jorgeboucas/meetup_example:v1
jboucas-mbpr:~$ docker login
Login with your Docker ID to push and pull images from Docker Hub. If you don't
have a Docker ID, head over to https://hub.docker.com to create one.
Username (jorgeboucas): jorgeboucas
Password:
Login Succeeded
```

# cloud.docker.com

The screenshot shows the Docker Cloud interface for creating a new repository. The top navigation bar includes the Docker Cloud logo, a '+' button, 'Get Help', and a user profile for 'jorgeboucas'. The left sidebar has sections for BUILD (Repositories), APPLICATIONS (Stacks, Services, Containers), INFRASTRUCTURE (Node Clusters, Nodes), and SETTINGS (Cloud Settings, Status). The main content area is titled 'Repositories / Create' and shows a 'Create Repository' form. The repository path is set to 'jorgeboucas/meetup\_example'. The name 'My meetup repo' is entered in the 'Repository Name' field. Under 'Visibility', the 'Private' option is selected. In the 'Build Settings (optional)' section, it says 'Autobuild triggers a new build with every git push to your source code repository' with a 'Learn more' link. At the bottom, there are two status indicators: 'Disconnected' next to a GitHub icon and 'Disconnected' next to a Jenkins icon. A red-bordered 'Cancel' button and a blue 'Create' button are at the bottom right.

BUILD

[Repositories](#)

APPLICATIONS

Stacks

Services

Containers

INFRASTRUCTURE

Node Clusters

Nodes

SETTINGS

Cloud Settings

Status

DOCKER CLOUD

+

Get Help

jorgeboucas

## Repositories / Create

### Create Repository

jorgeboucas / [meetup\\_example](#)

My meetup repo

Visibility

Using 1 of 1 private repositories. [Get more](#)

**Public** Public repositories appear in Docker Store search results

**Private** Only you can see private repositories

Build Settings (optional)

Autobuild triggers a new build with every **git push** to your source code repository [Learn more](#)

Disconnected

Disconnected

[Cancel](#) [Create](#)

# pushing an image

```
jboucas-mbpr:~$ docker push jorgeboucas/meetup_example:v1
The push refers to a repository [docker.io/jorgeboucas/meetup_example]
570a668954b5: Pushed
a2ae92ffcd29: Mounted from library/debian
v1: digest:
sha256:0d4c34cdd21c80c32791c09b5902dfd7c28815a2a990e2c75edc0fb66c3d57fb size:
736

jboucas-mbpr:~$ docker tag 686efbb118cd jorgeboucas/meetup_example:latest

jboucas-mbpr:~$ docker push jorgeboucas/meetup_example:latest
The push refers to a repository [docker.io/jorgeboucas/meetup_example]
570a668954b5: Layer already exists
a2ae92ffcd29: Layer already exists
latest: digest:
sha256:0d4c34cdd21c80c32791c09b5902dfd7c28815a2a990e2c75edc0fb66c3d57fb size:
736
```

The screenshot shows the Docker Cloud interface for a repository named `jorgeboucas/meetup_example`. The repository is described as "My meetup repo". It was last pushed "a few seconds ago". The "General" tab is selected, showing tags `latest` and `v1`, both pushed "a few seconds ago". The "ReadMe" section is empty. On the right, there's a "Docker commands" section with the command `$ docker push jorgeboucas/meetup_example:tagname` and a "Recent builds" section.

BUILD

**Repositories**

APPLICATIONS

- Stacks
- Services
- Containers

INFRASTRUCTURE

- Node Clusters
- Nodes

SETTINGS

- Cloud Settings

Status

Repositories / jorgeboucas / meetup\_example

**General** Tags Builds Timeline Settings

**jorgeboucas / meetup\_example**

My meetup repo

Last pushed: a few seconds ago

**Tags and Scans**

Add [Docker Security Scanning](#) to see vulnerabilities in your repos.

Tag	Pushed
latest	a few seconds ago
v1	a minute ago

[See all](#)

**ReadMe**

Repository description is empty. Click [here](#) to edit.

**Docker commands**

You can always push a new image to this repository.

```
$ docker push  
jorgeboucas/meetup_example:tagname
```

**Recent builds**

Link a source provider and run a build to see build results here.

# removing images

```
## remove images

jboucas-mbpr:~$ docker images
REPOSITORY          TAG      IMAGE ID
CREATED             SIZE
debian              latest   e5599115b6a6
6 days ago          123 MB
jorgeboucas/meetup_example
7 hours ago         latest   686efbb118cd
jorgeboucas/meetup_example
7 hours ago         v1      686efbb118cd
123 MB

jboucas-mbpr:~$ docker rmi jorgeboucas/meetup_example
Untagged: jorgeboucas/meetup_example:latest

jboucas-mbpr:~$ docker rmi jorgeboucas/meetup_example:v1
Untagged: jorgeboucas/meetup_example:v1
Untagged: jorgeboucas/
meetup_example@sha256:0d4c34cdd21c80c32791c09b5902dfd7c28815a2a990e2c75edc0fb66
c3d57fb
```

# pull your image back

```
jboucas-mbpr:~$ docker images
REPOSITORY          TAG      IMAGE ID
CREATED            SIZE
debian              latest   e5599115b6a6
6 days ago         123 MB

## pull your own image and check the content

jboucas-mbpr:~$ docker pull jorgeboucas/meetup_example
Using default tag: latest
latest: Pulling from jorgeboucas/meetup_example
Digest: sha256:0d4c34cdd21c80c32791c09b5902dfd7c28815a2a990e2c75edc0fb66c3d57fb
Status: Downloaded newer image for jorgeboucas/meetup_example:latest

jboucas-mbpr:~$ docker images
REPOSITORY          TAG      IMAGE ID
CREATED            SIZE
jorgeboucas/meetup_example    latest   686efbb118cd
7 hours ago        123 MB
debian              latest   e5599115b6a6
6 days ago         123 MB
```

# run your image again

```
jboucas-mbpr:~$ docker run --name my_own_second_container -t -i jorgeboucas/meetup_example /bin/bash

root@cdaa6dcb587b:/# ls
bin  boot  dev  etc  home  jorge.txt  lib  lib64  media  mnt  opt  proc  root
run  sbin  srv  sys  tmp  usr  var
root@cdaa6dcb587b:/# exit
exit
```

# Private Docker Registry

The screenshot shows the Docker Store interface. At the top, there's a search bar with 'docker registry' and navigation links for 'Explore', 'Publish', and 'Feedback?'. A user profile for 'jorgeboucas' is also visible.

**registry**  
By Docker Official Image

Containerized docker registry  
10M+ Pulls  
Categories: [Featured Images](#), [Storage](#), [Application Services](#)

**Free Product Tier**  
**FREE**

[Terms of Service](#)

Copy and paste to pull this image:  
`docker pull registry:2.5.0`

[View Available Tags](#)

**DESCRIPTION** **REVIEWS** **RESOURCES**

**Supported tags and respective Dockerfile links**

- [2](#), [2.5](#), [2.5.1](#), [latest \(Dockerfile\)](#)

For more information about this image and its history, please see [the relevant manifest file \(library/registry\)](#). This image is updated via pull requests to the [docker-library/official-images](#) GitHub repo.

For detailed information about the virtual/transfer sizes and individual layers of each of the above supported tags, please see the [repos/registry/tag-details.md](#) file in the [docker-library/repo-info](#) GitHub repo.

**Docker Registry**

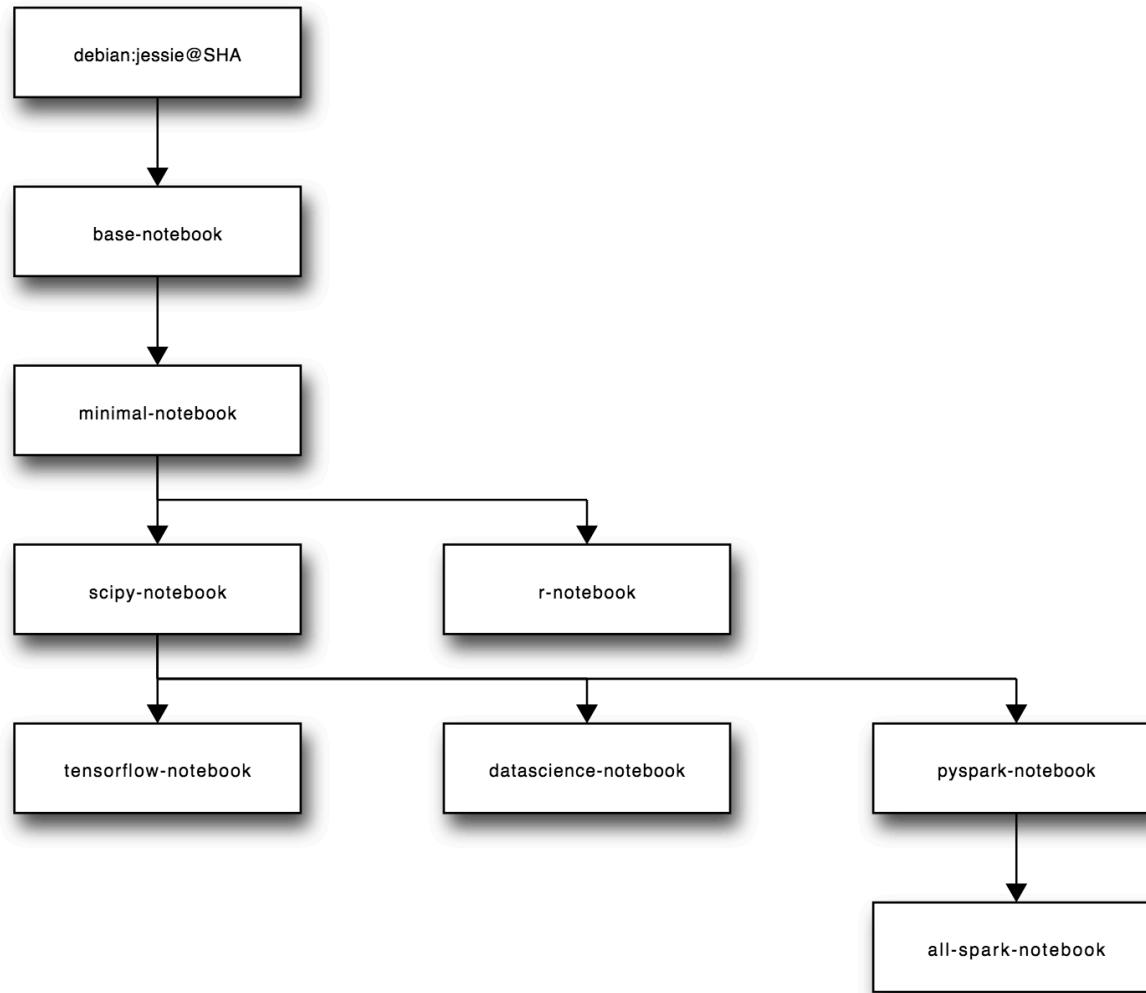
Average Rating: 5 Ratings

Select a product tier

Click to Rate this Product

Add Review

# [github.com/jupyter/docker-stacks](https://github.com/jupyter/docker-stacks)



# github.com/jupyter/docker-stacks/blob/master/base-notebook/Dockerfile

```
1 # Copyright (c) Jupyter Development Team.
2 # Distributed under the terms of the Modified BSD License.
3
4 # Debian Jessie image released 2016 May 03.
5 FROM debian@sha256:f7062cf040f67f0c26ff46b3b44fe036c29468a7e69d8170f37c57f2eec1261b
6
7 MAINTAINER Jupyter Project <jupyter@googlegroups.com>
8
9 USER root
10
11 # Install all OS dependencies for notebook server that starts but lacks all
12 # features (e.g., download as all possible file formats)
13 ENV DEBIAN_FRONTEND noninteractive
14 RUN REPO=http://cdn-fastly.deb.debian.org \
15     && echo "deb $REPO/debian jessie main\ndeb $REPO/debian-security jessie/updates main" > /etc/apt/sources.list \
16     && apt-get update && apt-get -yq dist-upgrade \
17     && apt-get install -yq --no-install-recommends \
18         wget \
19         bzip2 \
20         ca-certificates \
21         sudo \
22         locales \
23     && apt-get clean \
24     && rm -rf /var/lib/apt/lists/*
25
26 RUN echo "en_US.UTF-8 UTF-8" > /etc/locale.gen && \
27     locale-gen
28
29 # Install Tini
30 RUN wget --quiet https://github.com/krallin/tini/releases/download/v0.10.0/tini && \
31     echo "1361527f39190a7338a0b434bd8c88ff7233ce7b9a4876f3315c22fce7eca1b0 *tini" | sha256sum -c - && \
32     mv tini /usr/local/bin/tini && \
33     chmod +x /usr/local/bin/tini
--
```

# github.com/jupyter/docker-stacks/blob/master/base-notebook/Dockerfile

```
34
35 # Configure environment
36 ENV CONDA_DIR /opt/conda
37 ENV PATH $CONDA_DIR/bin:$PATH
38 ENV SHELL /bin/bash
39 ENV NB_USER jovyan
40 ENV NB_UID 1000
41 ENV HOME /home/$NB_USER
42 ENV LC_ALL en_US.UTF-8
43 ENV LANG en_US.UTF-8
44 ENV LANGUAGE en_US.UTF-8
45
46 # Create jovyan user with UID=1000 and in the 'users' group
47 RUN useradd -m -s /bin/bash -N -u $NB_UID $NB_USER && \
48     mkdir -p $CONDA_DIR && \
49     chown $NB_USER $CONDA_DIR
50
51 USER $NB_USER
52
53 # Setup jovyan home directory
54 RUN mkdir /home/$NB_USER/work && \
55     mkdir /home/$NB_USER/.jupyter && \
56     echo "cacert=/etc/ssl/certs/ca-certificates.crt" > /home/$NB_USER/.curlrc
57
58 # Install conda as jovyan
59 RUN cd /tmp && \
60     mkdir -p $CONDA_DIR && \
61     wget --quiet https://repo.continuum.io/miniconda/Miniconda3-4.2.12-Linux-x86_64.sh && \
62     echo "c59b3dd3cad550ac7596e0d599b91e75d88826db132e4146030ef471bb434e9a *Miniconda3-4.2.12-Linux-x86_64.sh" | sha256sum -c - && \
63     /bin/bash Miniconda3-4.2.12-Linux-x86_64.sh -f -b -p $CONDA_DIR && \
64     rm Miniconda3-4.2.12-Linux-x86_64.sh && \
65     $CONDA_DIR/bin/conda config --system --add channels conda-forge && \
66     $CONDA_DIR/bin/conda config --system --set auto_update_conda false && \
67     conda clean -tipsy
--
```

# github.com/jupyter/docker-stacks/blob/master/base-notebook/Dockerfile

```
--  
69  # Install Jupyter Notebook and Hub  
70  RUN conda install --quiet --yes \  
    'notebook=4.3*' \  
    jupyterhub=0.7 \  
    && conda clean -tipsy  
74  
75  USER root  
76  
77  EXPOSE 8888  
78  WORKDIR /home/$NB_USER/work  
79  
80  # Configure container startup  
81  ENTRYPOINT ["tini", "--"]  
82  CMD ["start-notebook.sh"]  
83  
84  # Add local files as late as possible to avoid cache busting  
85  COPY start.sh /usr/local/bin/  
86  COPY start-notebook.sh /usr/local/bin/  
87  COPY start-singleuser.sh /usr/local/bin/  
88  COPY jupyter_notebook_config.py /home/$NB_USER/.jupyter/  
89  RUN chown -R $NB_USER:users /home/$NB_USER/.jupyter  
90  
91  # Switch back to jovyan to avoid accidental container runs as root  
92  USER $NB_USER
```

<https://github.com/krallin/tini/issues/8>

[https://docs.docker.com/engine/userguide/eng-image/dockerfile\\_best-practices/](https://docs.docker.com/engine/userguide/eng-image/dockerfile_best-practices/)

<https://docs.docker.com/engine/reference/builder/>

# [github.com/mpg-age-bioinformatics/ ENCODE\\_RNAseq\\_eCLIP](https://github.com/mpg-age-bioinformatics/ENCODE_RNAseq_eCLIP)

## **ENCODE\_RNAseq\_eCLIP**

This is a *Jupyter* notebook using *Python*, *R* and *bash* to integrate RNAseq and eCLIP data from ENCODE. It uses an RNAseq data set of KHSRP-shRNA on K562 cells and respective shRNA-mock control. It uses as well an eCLIP data set on KHSRP.

This pipeline comes with a *Docker* image containing all required software.

### **Installation**

You will need to have [Docker](#) and [git](#) installed.

Start by cloning this repository into your home directory:

```
cd ~/  
git clone https://github.com/mpg-age-bioinformatics/ENCODE_RNAseq_eCLIP
```

Then build the Docker image:

```
cd ~/ENCODE_RNAseq_eCLIP  
sudo docker build -t encode_rnaseq_eclip .
```

# github.com/mpg-age-bioinformatics/ ENCODE\_RNAseq\_eCLIP

## Usage

The analysis pipeline was tested on a Docker image running with 8gb of memory and 4 cpus.

Start your container with:

```
sudo docker run -d -p 8888:8888 \
-e GRANT_SUDO=yes -e NB_UID=1000 --user root \
-e PASSWORD="YOURPASS" -e USE_HTTPS=yes \
-v ~/ENCODE_RNAseq_eCLIP/notebooks:/home/jovyan/work/notebooks \
-v ~/ENCODE_RNAseq_eCLIP/results:/home/jovyan/work/results \
--name encode_rnaseq_eclip \
-i -t encode_rnaseq_eclip
```

Instead of `"YOURPASS"` type in a password of your choice.

Then, on your web browser connect to <https://localhost:8888>.

You will be requested to accept the self-signed certificate and to give in the password you used in `"YOURPASS"`.

# Jupyter, <https://localhost:8888>



# Jupyter, <https://localhost:8888>

The screenshot shows a Jupyter Notebook interface with the following components:

- Header:** Jupyter Untitled Last Checkpoint: 18 minutes ago (unsaved changes) Python 2 Logout
- Toolbar:** File Edit View Insert Cell Kernel Help
- Cell Buttons:** New Cell, Insert Cell Below, Insert Cell Above, Run Cell, Stop Cell, Cell Toolbar
- Section Header:** Some cool markdown header
- Text:** with some descriptive text
- Code Cells:**
  - In [1]: `print "some python2"`  
some python2
  - In [2]: `%%bash  
printf "some bash"`  
some bash
  - In [3]: `%%load_ext rpy2.ipython`
  - In [5]: `%%R  
print("some R")`  
[1] "some R"
- Section Header:** Part 2
- Text:** Some more descriptive markdown text with
- Empty Cells:** In [ ]: (four empty code cells)

# Jupyter, <https://localhost:8888>

jupyter Untitled Last Checkpoint: 3 minutes ago (autosaved) Logout Python 2

In [21]:

```
import pandas as pd
# Make a pandas DataFrame
df = pd.DataFrame({'Alphabet': ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i'],
                   'A': [4, 3, 5, 2, 1, 7, 7, 5, 9],
                   'B': [0, 4, 3, 6, 7, 10, 11, 9, 13],
                   'C': [1, 2, 3, 1, 2, 3, 1, 2, 3]})
```

Out[21]:

	A	Alphabet	B	C
0	4	a	0	1
1	3	b	4	2
2	5	c	3	3
3	2	d	6	1
4	1	e	7	2

In [18]:

```
%matplotlib inline
print "Inline graphics"
plt.scatter(df["A"], df["B"], c=df["C"])
```

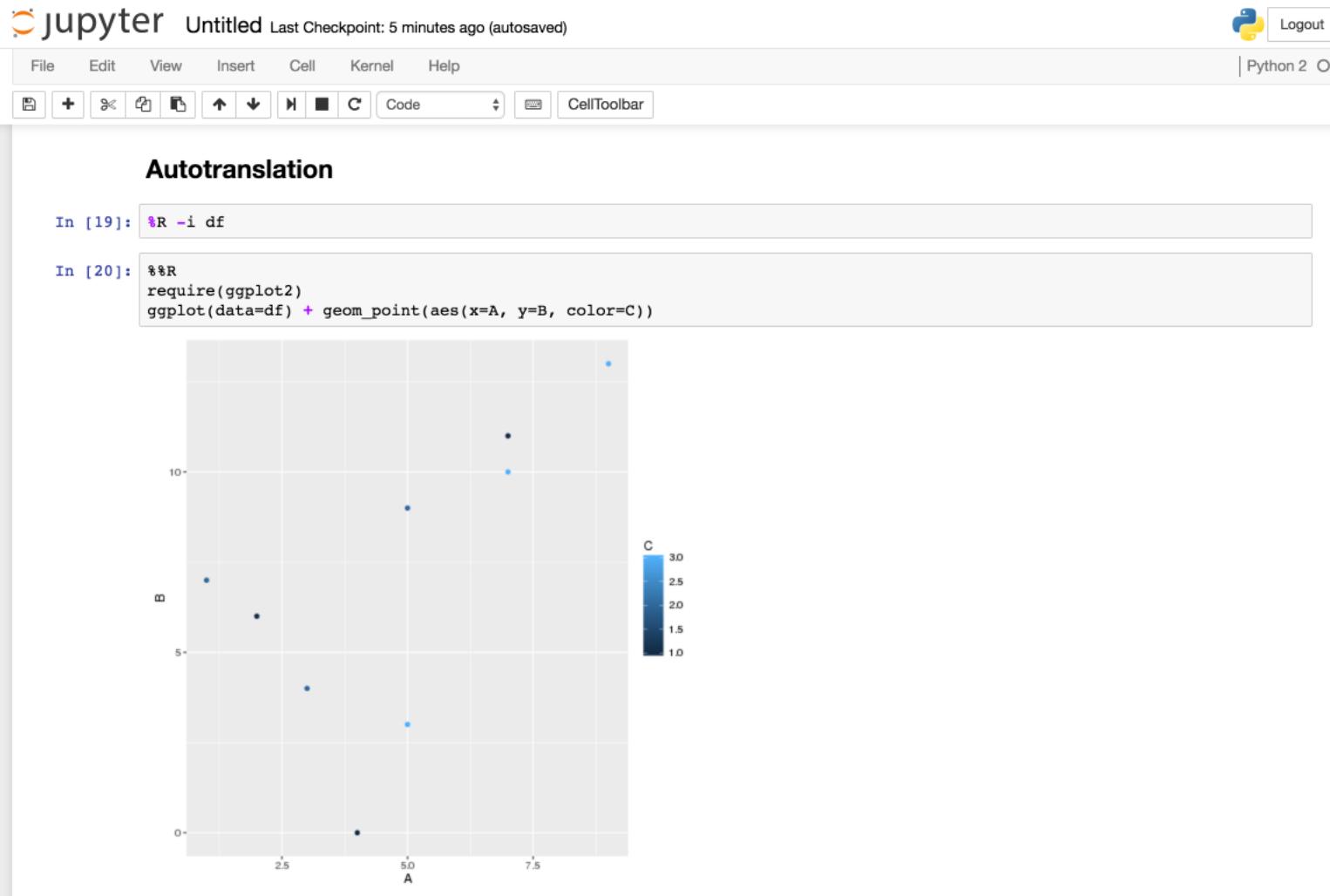
Inline graphics

Out[18]:

```
<matplotlib.collections.PathCollection at 0x7f3df7b4b8d0>
```

Detailed description: This scatter plot displays the data from the DataFrame. The x-axis is labeled 'A' and ranges from 0 to 10. The y-axis is labeled 'B' and ranges from -2 to 14. There are nine data points. The points are colored based on their value in column 'C'. The colors correspond to the values: 1 (green), 2 (black), 3 (blue), and 4 (red). The points are approximately at (1, 7), (2, 6), (3, 4), (4, 0), (5, 9), (7, 11), (7, 10), (9, 13), and (9, 12).

# Jupyter, <https://localhost:8888>



# Jupyter on a remote server

The sidebar on the left contains the following navigation:

- Jupyter Notebook** latest
- Search docs
- USER DOCUMENTATION**
  - The Jupyter Notebook
  - Installation
  - Running the Notebook
  - Migrating from IPython
  - UI Components
  - Comms
- CONFIGURATION**
  - Configuration Overview
  - Config file and command line options
- Running a notebook server**
  - Securing a notebook server
  - Running a public notebook server
    - Running the notebook with a customized URL prefix
    - Embedding the notebook in another website
  - Known issues
- Security in the Jupyter notebook server
- Security in notebook documents
- Configuring the notebook frontend

At the bottom, there are links for "Read the Docs" and a dropdown menu showing "v: latest".

Docs » Running a notebook server

Edit on GitHub

## Running a notebook server

The [Jupyter notebook](#) web application is based on a server-client structure. The notebook server uses a [two-process kernel architecture](#) based on [ZeroMQ](#), as well as [Tornado](#) for serving HTTP requests.

### Note

By default, a notebook server runs locally at 127.0.0.1:8888 and is accessible only from *localhost*. You may access the notebook server from the browser using <http://127.0.0.1:8888>.

This document describes how you can [secure a notebook server](#) and how to [run it on a public interface](#).

### Important

**This is not the multi-user server you are looking for.** This document describes how you can run a public server with a single user. This should only be done by someone who wants remote access to their personal machine. Even so, doing this requires a thorough understanding of the set-ups limitations and security implications. If you allow multiple users to access a notebook server as it is described in this document, their commands may collide, clobber and overwrite each other.

If you want a multi-user server, the official solution is [JupyterHub](#). To use JupyterHub, you need a Unix server (typically Linux) running somewhere that is accessible to your users on a network. This may run over the public internet, but doing so introduces additional [security concerns](#).

## Securing a notebook server

# multi-user Jupyter server

## JupyterHub

With JupyterHub you can create a **multi-user Hub** which spawns, manages, and proxies multiple instances of the single-user [Jupyter notebook](#) server. Due to its flexibility and customization options, JupyterHub can be used to serve notebooks to a class of students, a corporate data science group, or a scientific research group.

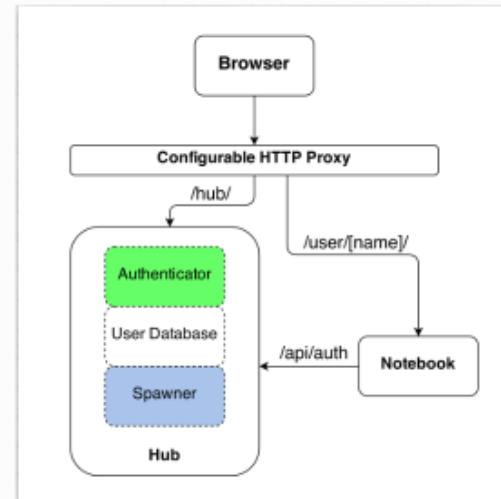
Three subsystems make up JupyterHub:

- a multi-user **Hub** (tornado process)
- a **configurable http proxy** (node-http-proxy)
- multiple **single-user Jupyter notebook servers** (Python/IPython/tornado)

JupyterHub's basic flow of operations includes:

- The Hub spawns a proxy
- The proxy forwards all requests to the Hub by default
- The Hub handles user login and spawns single-user servers on demand
- The Hub configures the proxy to forward URL prefixes to the single-user notebook servers

For convenient administration of the Hub, its users, and [Services](#) (added in version 7.0), JupyterHub also provides a [REST API](#).



# Notebooks on Github: the “actual scolarship”

The figure consists of two side-by-side screenshots of a Mac OS X desktop. On the left, a web browser window displays the Nature Genetics website. The main content is an article titled "Multi-tiered genomic analysis of head and neck cancer ties TP53 mutation to 3p loss". On the right, a GitHub interface shows a repository named "TCGA / Analysis\_Notebooks". A specific file, "TP53\_exploration.ipynb", is open in a code editor. The notebook contains several sections of Python code and associated plots. One plot, titled "HNSCC HPV- Cohort", shows survival probability over 5 years for different groups. Another section is titled "TP53 Mutation Clinical Coorelates". The GitHub interface also shows a list of commits and branches.

[https://figshare.com/articles/Project\\_Jupyter\\_interactive\\_computing\\_in\\_the\\_context\\_of\\_modern\\_science\\_gateways/4495976](https://figshare.com/articles/Project_Jupyter_interactive_computing_in_the_context_of_modern_science_gateways/4495976)

# Interactive notebooks: Sharing the code

<http://www.nature.com/news/interactive-notebooks-sharing-the-code-1.16261>

The screenshot shows a news article from the journal 'nature'. The title of the article is 'Interactive notebooks: Sharing the code'. Below the title, it says 'The free IPython notebook makes data analysis easier to record, understand and reproduce.' The author is Helen Shen, and the date is 05 November 2014. There are links for PDF and Rights & Permissions. The main image is a stylized illustration of a hand interacting with a spiral-bound notebook containing scientific data like graphs and charts. To the right of the main article, there is a sidebar with a Mars rover image and text about a \$2.4-billion plan to steal a rock from Mars. Below that is a social sharing section with 'Like' and 'Share' buttons. At the bottom, there is a news feed with two recent articles.

Search  Go ▶ Advanced search

Home | News & Comment | Research | Careers & Jobs | Current Issue | Archive | Audio & Video | For Authors

Archive > Volume 515 > Issue 7525 > Toolbox > Article

NATURE | TOOLBOX

E-alert RSS Facebook Twitter

## Interactive notebooks: Sharing the code

The free IPython notebook makes data analysis easier to record, understand and reproduce.

Helen Shen

05 November 2014

PDF Rights & Permissions

Next stop: Mars

The \$2.4-billion plan to steal a rock from Mars

NASA is now building the rover that it hopes will bring back signs of life on the red planet.

Like Share Silke Uhrig-Schmidt and 319k others like this.

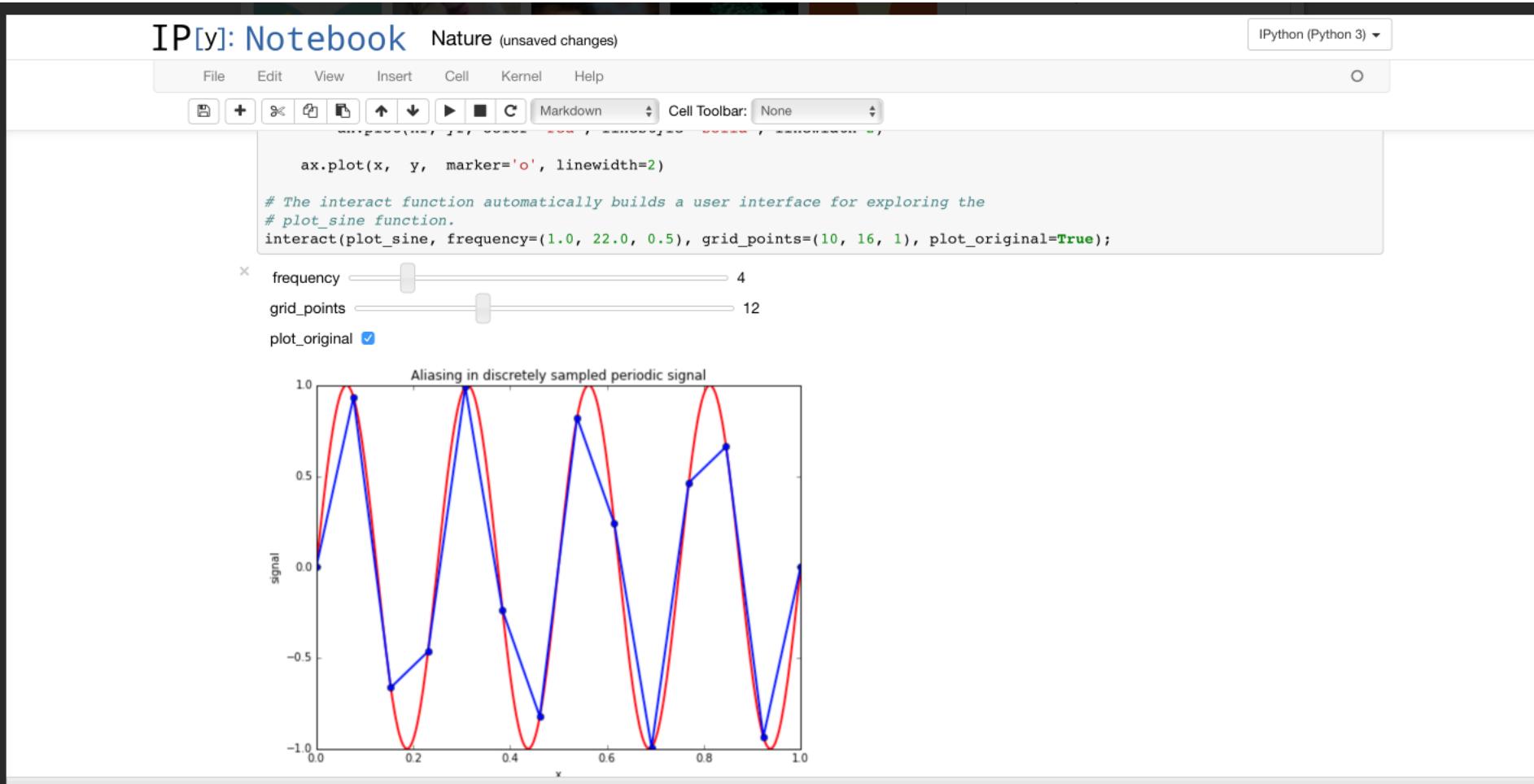
innovations.kaimrc.med.sa

Recent Read Commented

1. 'You never said my peer review was confidential' — scientist challenges publisher  
*Nature* | 23 January 2017
2. Scientists join massive protest against Trump

# Executable papers

[https://figshare.com/articles/Project\\_Jupyter\\_interactive\\_computing\\_in\\_the\\_context\\_of\\_modern\\_science\\_gateways/4495976](https://figshare.com/articles/Project_Jupyter_interactive_computing_in_the_context_of_modern_science_gateways/4495976)



# Alternatives to Jupyter

# The Perfect Tool for Iterative Experimentation

Beaker Lab is currently in Beta and so we are offering complete use of the system for free.

# Apache Zeppelin

A web-based notebook that enables interactive data analytics.  
You can make beautiful data-driven, interactive and collaborative documents with SQL, Scala and more.



MAX PLANCK INSTITUTE FOR BIOLOGY OF AGEING | BIOINFORMATICS

# Alternatives to Docker

- **VAGRANT** : <https://www.vagrantup.com>
- **Open Container Initiative (OCI)** : <https://www.opencontainers.org>
- **Kubernetes** : <https://kubernetes.io>
- **CoreOS and rkt** : <https://coreos.com/rkt/>
- **Apache Mesos and Mesosphere** : <http://mesos.apache.org>
- **Canonical and LXD** : <https://www.ubuntu.com/cloud/lxd>

<http://searchcloudapplications.techtarget.com/tip/Five-development-containers-to-consider-that-arent-Docker>

# Containers for HPC

- **Singularity:** <http://singularity.lbl.gov> (**sudo for Docker imports ?**)

Singularity natively supports InfiniBand, Lustre, and works seamlessly with all resource managers (e.g. SLURM, Torque, SGE, etc.) because it works like running any other command on the system.

```
$ singularity exec /tmp/Demo.img xterm  
$ singularity exec /tmp/Demo.img python script.py  
$ singularity exec /tmp/Demo.img python < /path/to/python/script.py  
$ cat /path/to/python/script.py | singularity exec /tmp/Demo.img python  
$ mpirun -np X singularity exec /path/to/container.img /usr/bin/  
mpi_program_inside_container (mpi program args)
```

# Containers for HPC

- Shifter: <https://github.com/NERSC/shifter>

With SLURM plugin, spank

<https://github.com/SchedMD/slurm/blob/master/slurm/spank.h>

```
```
#!/bin/bash
#SBATCH --image=dmjacobsen/mpitest
#SBATCH --volume=/scratch/dmj/i:/input
#SBATCH --volume=/scratch/dmj/o:/output
#SBATCH -N 250
```
srun shifter /app

shifterimg pull debian:latest # instead of docker pull
shifter --image=debian:latest bash -login # for interactive session
```

# Containers for HPC

- **Shifter:** <https://github.com/NERSC/shifter>

With SLURM plugin, spank

<https://github.com/SchedMD/slurm/blob/master/slurm/spank.h>

```
```
#!/bin/bash
#SBATCH --image=dmjacobsen/mpitest
#SBATCH --volume=/scratch/dmj/i:/input
#SBATCH --volume=/scratch/dmj/o:/output
#SBATCH -N 250
```
srun shifter /app

shifterimg pull debian:latest # instead of docker pull
shifter --image=debian:latest bash -login # for interactive session
```

**END**

[bioinformatics@age.mpg.de](mailto:bioinformatics@age.mpg.de)

<https://mpg-age-bioinformatics.github.io>