

# Jupyter Notebooks for Performing & Sharing Bioinformatics Analyses

## Windows users: From the Anaconda R:

```
> install.packages(  
  c('rzmq', 'repr', 'IRkernel', 'IRdisplay'),  
  repos = c('http://irkernel.github.io/',  
           'http://cran.utstat.utoronto.ca/'))
```

## Everyone: Download and unpack

```
> install.packages(  
  c('rzmq', 'repr', 'IRkernel', 'IRdisplay'),  
  repos = c('http://irkernel.github.io/',  
           'http://cran.utstat.utoronto.ca/'))
```

# Jupyter Notebooks for Performing & Sharing Bioinformatics Analyses

Jonathan Dursi, OICR

[jonathan@dursi.ca](mailto:jonathan@dursi.ca)

<https://github.com/ljdursi/glbio-jupyter-workshop>



# The plan for this morning

- Run Jupyter notebooks on your laptop, either in R and Python
- How to perform interactive analyses in a web browser using Jupiter
- How does it all work?
- Using markdown and latex to format notebooks nicely
- “Port” an R bioinformatics workflow from scripts into a Jupyter notebook
- Sharing a Jupyter notebook online:
  - SageMathCloud
  - GitHub and git; nbviewer
  - [mybinder.org](https://mybinder.org)

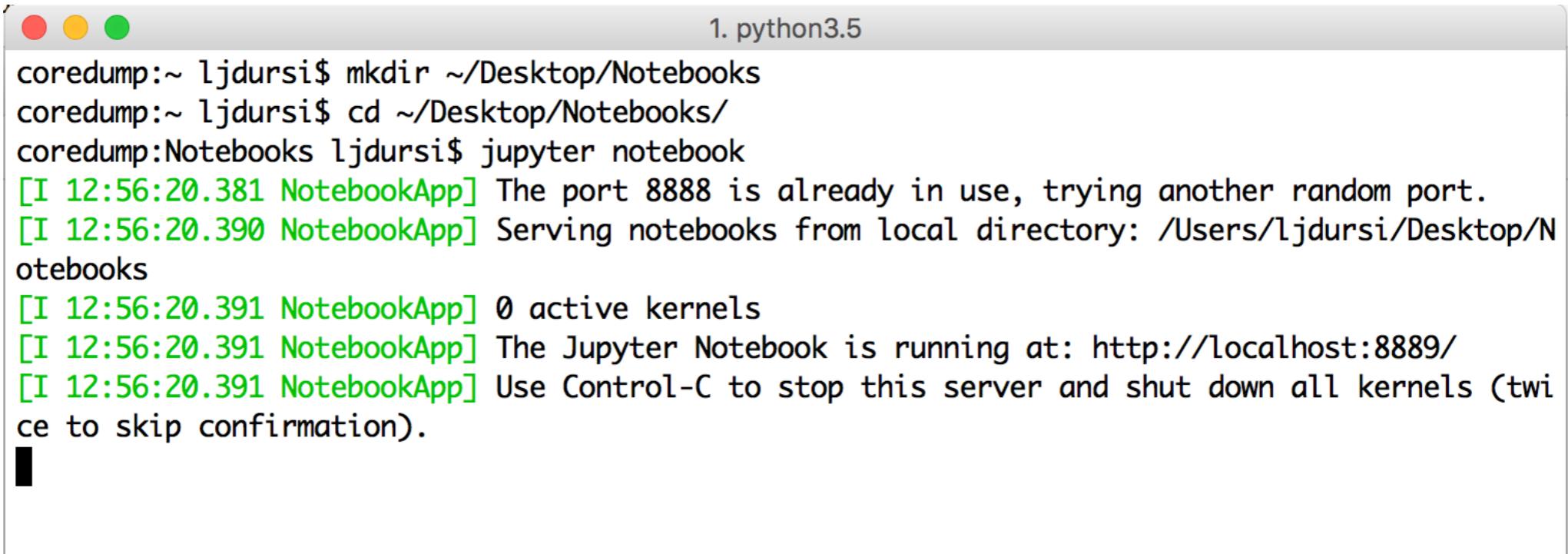
# The plan for this morning

Perfectly reasonable and interesting things to discuss that we **won't** be talking about today:

- Combining R and Python: cool and possible but requires a little setup
- IPython magics (probably)
- The zillion R bioinformatics packages that are available
- How to use R, Python

# Let's start up a notebook

- Make a Notebooks directory and start from there
- Command Line: Linux, Mac

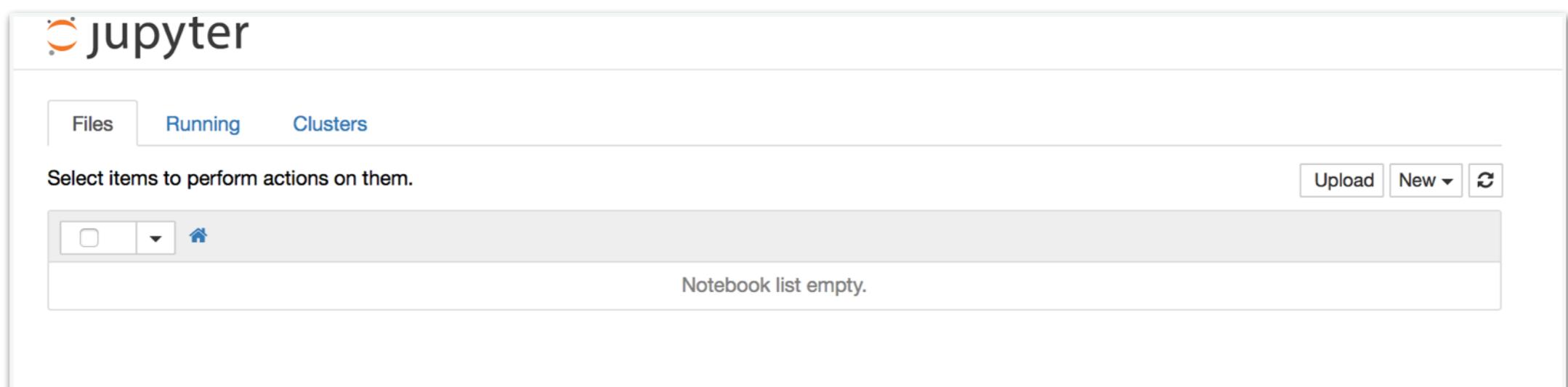


```
coredump:~ ljdursi$ mkdir ~/Desktop/Notebooks
coredump:~ ljdursi$ cd ~/Desktop/Notebooks/
coredump:Notebooks ljdursi$ jupyter notebook
[I 12:56:20.381 NotebookApp] The port 8888 is already in use, trying another random port.
[I 12:56:20.390 NotebookApp] Serving notebooks from local directory: /Users/ljdursi/Desktop/Notebooks
[I 12:56:20.391 NotebookApp] 0 active kernels
[I 12:56:20.391 NotebookApp] The Jupyter Notebook is running at: http://localhost:8889/
[I 12:56:20.391 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
|
```



# Let's start up a notebook

- Make a Notebooks directory and start from there
- Command Line: Linux, Mac
- Browser should open (or go to localhost:8888)



# Let's start up a notebook

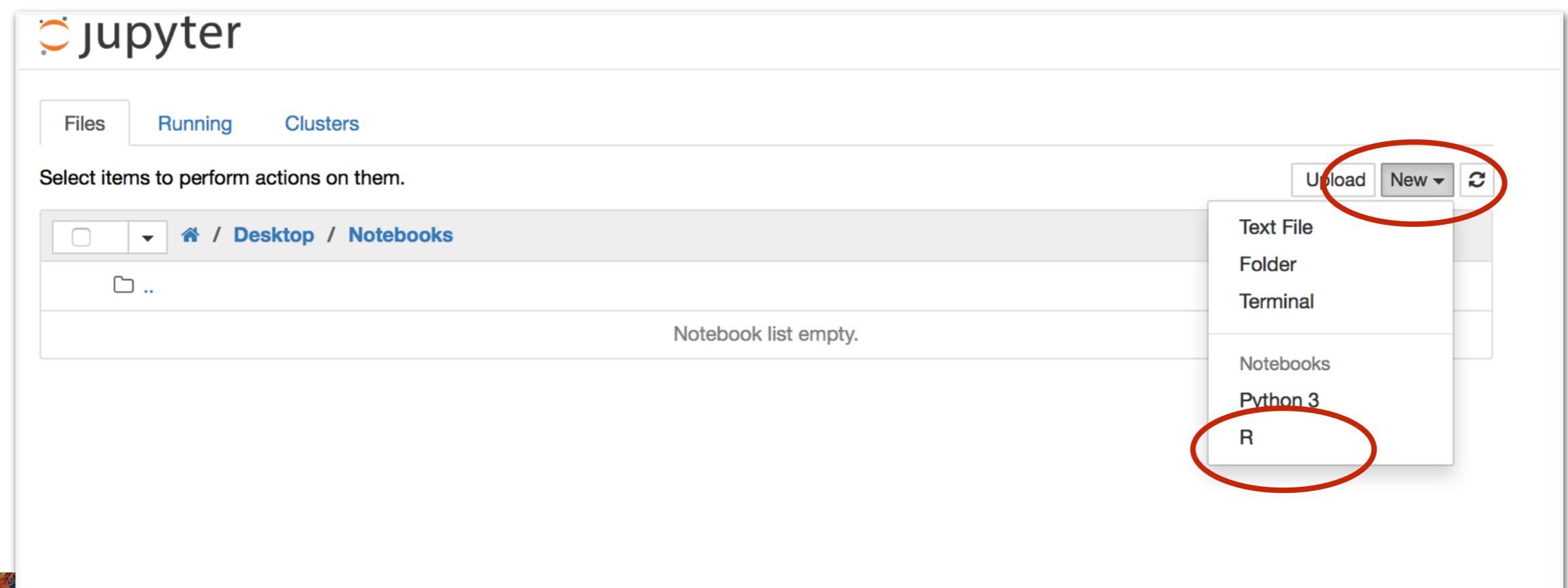
- Or you can start from home (eg, launch from start menu: Win) and navigate folders, and create new folders:

jupyter



# Let's start up a notebook

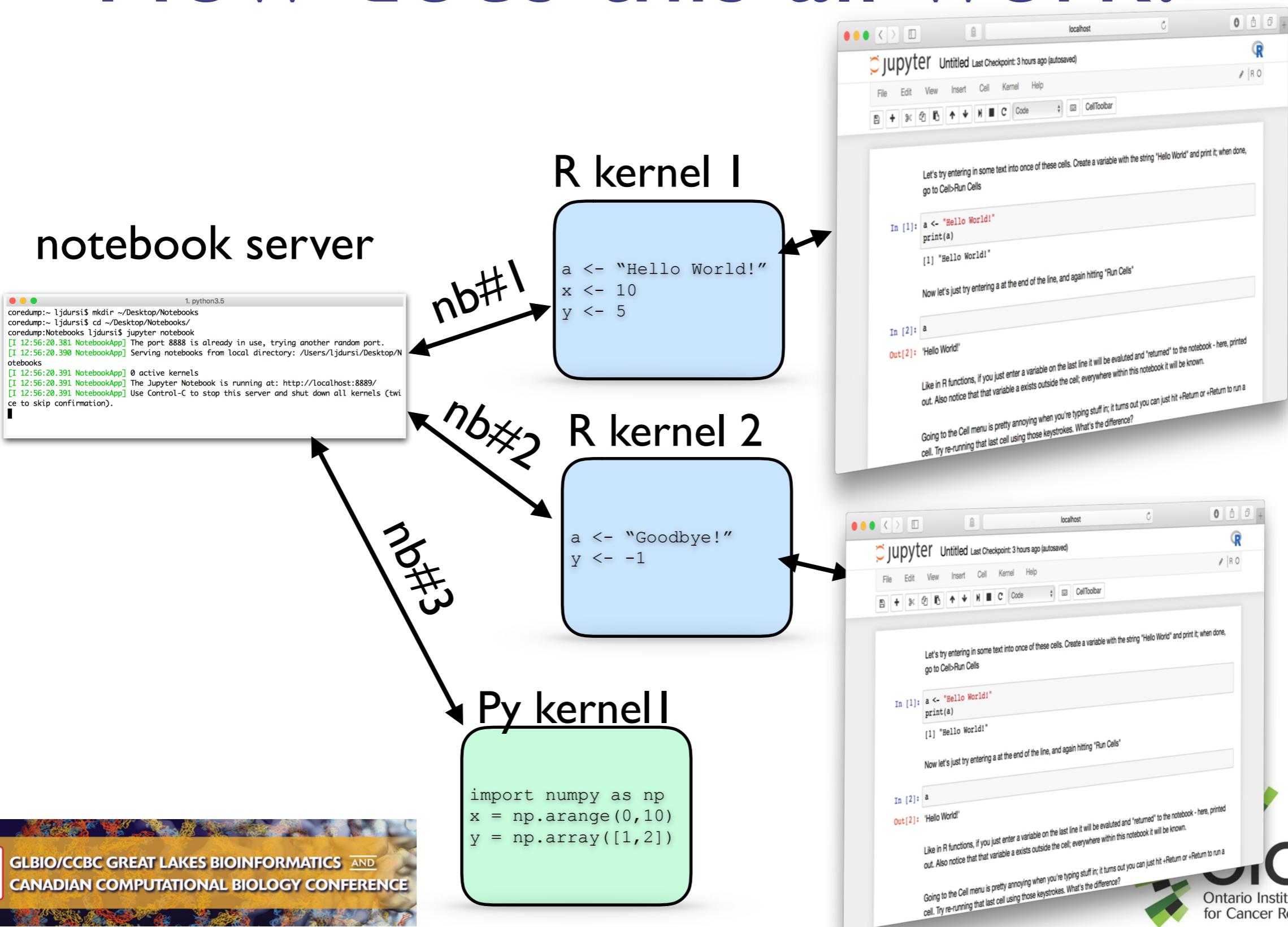
- Let's start up an R notebook



# Demo - starting notebook

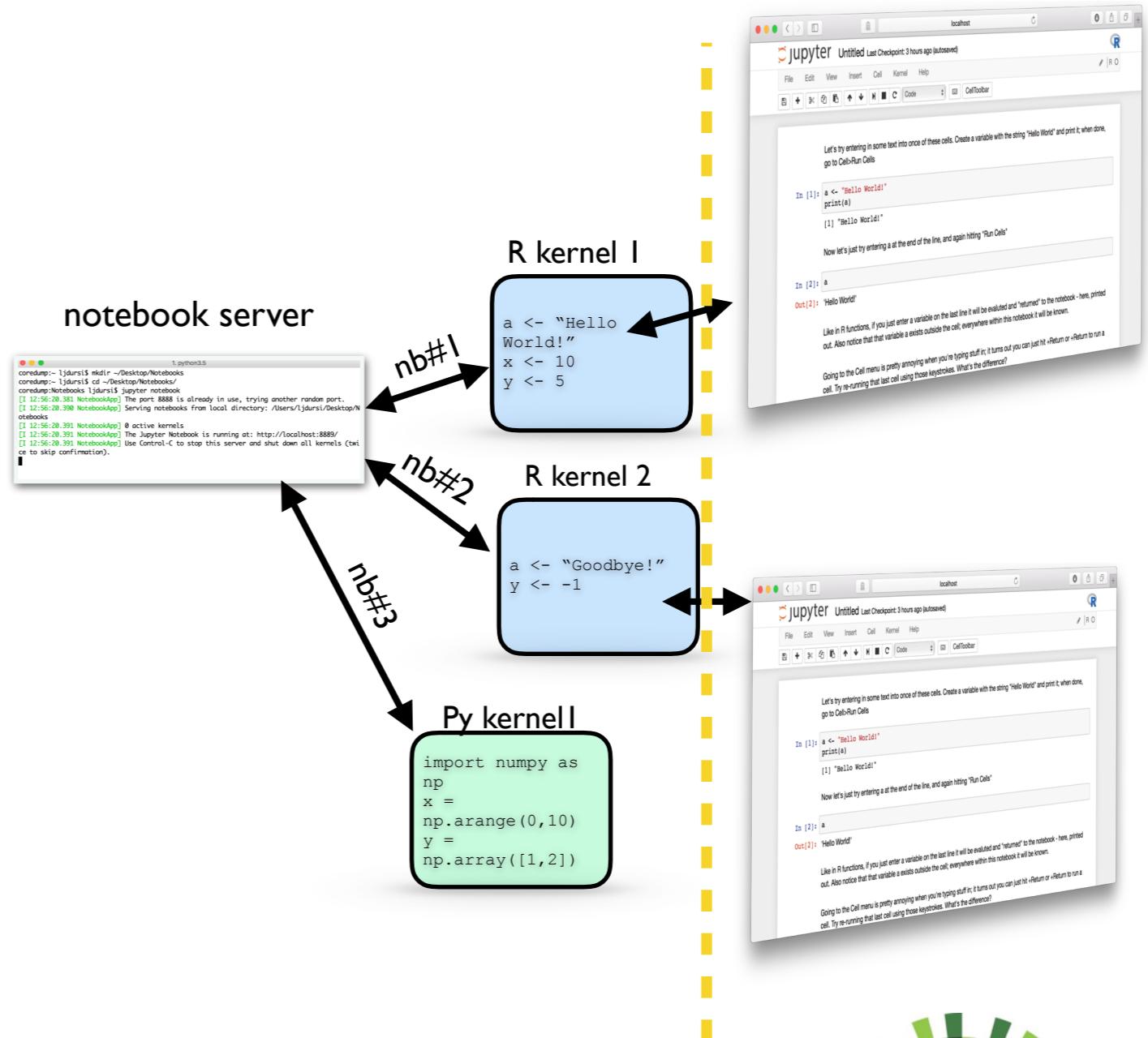
- I-Introduction

# How does this all work?



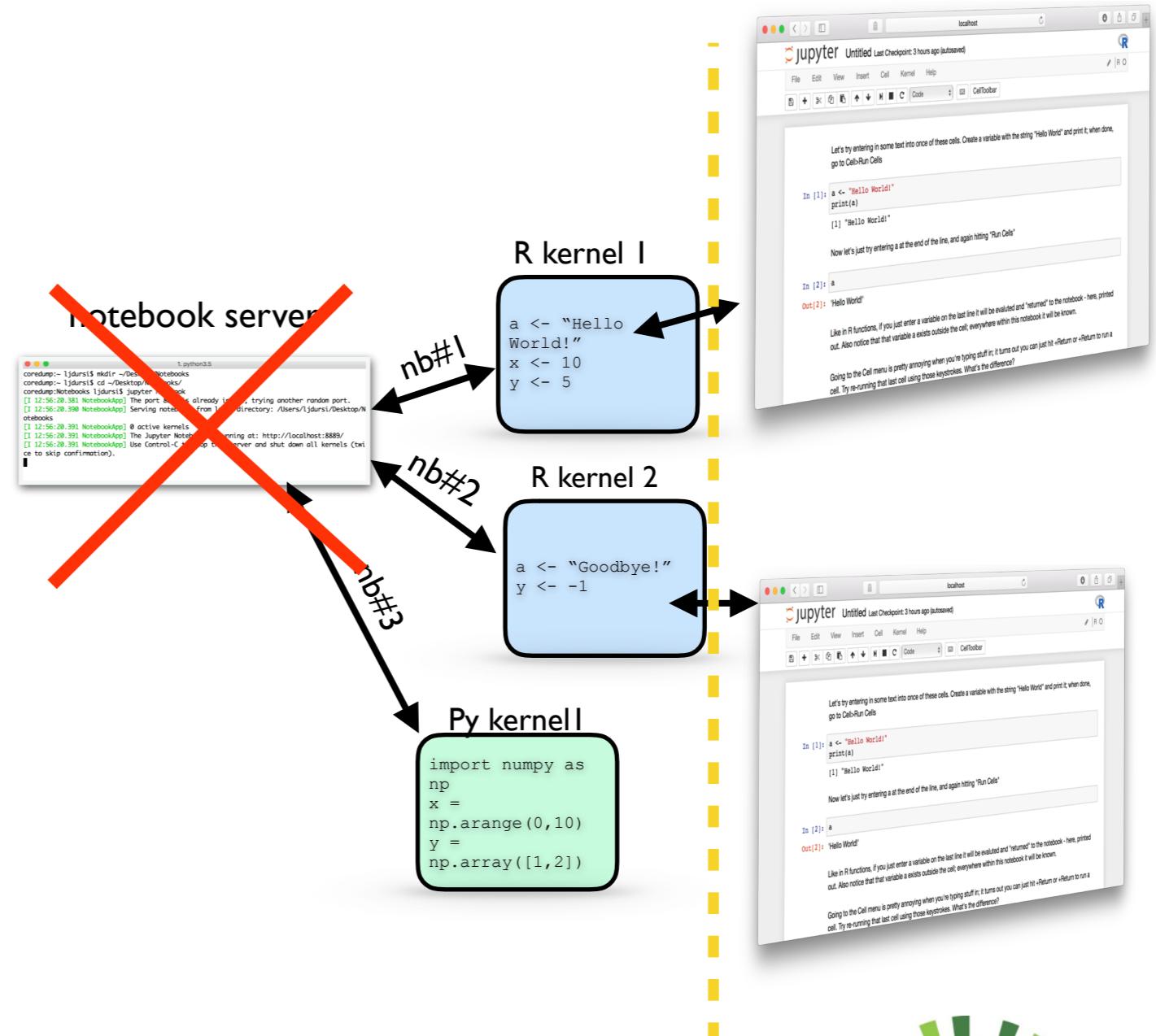
# How does this all work?

- There's no rule that says the notebook server/kernels has to be on same machine as browser!
- Can be running on remote server (or cluster)
- AWS demo

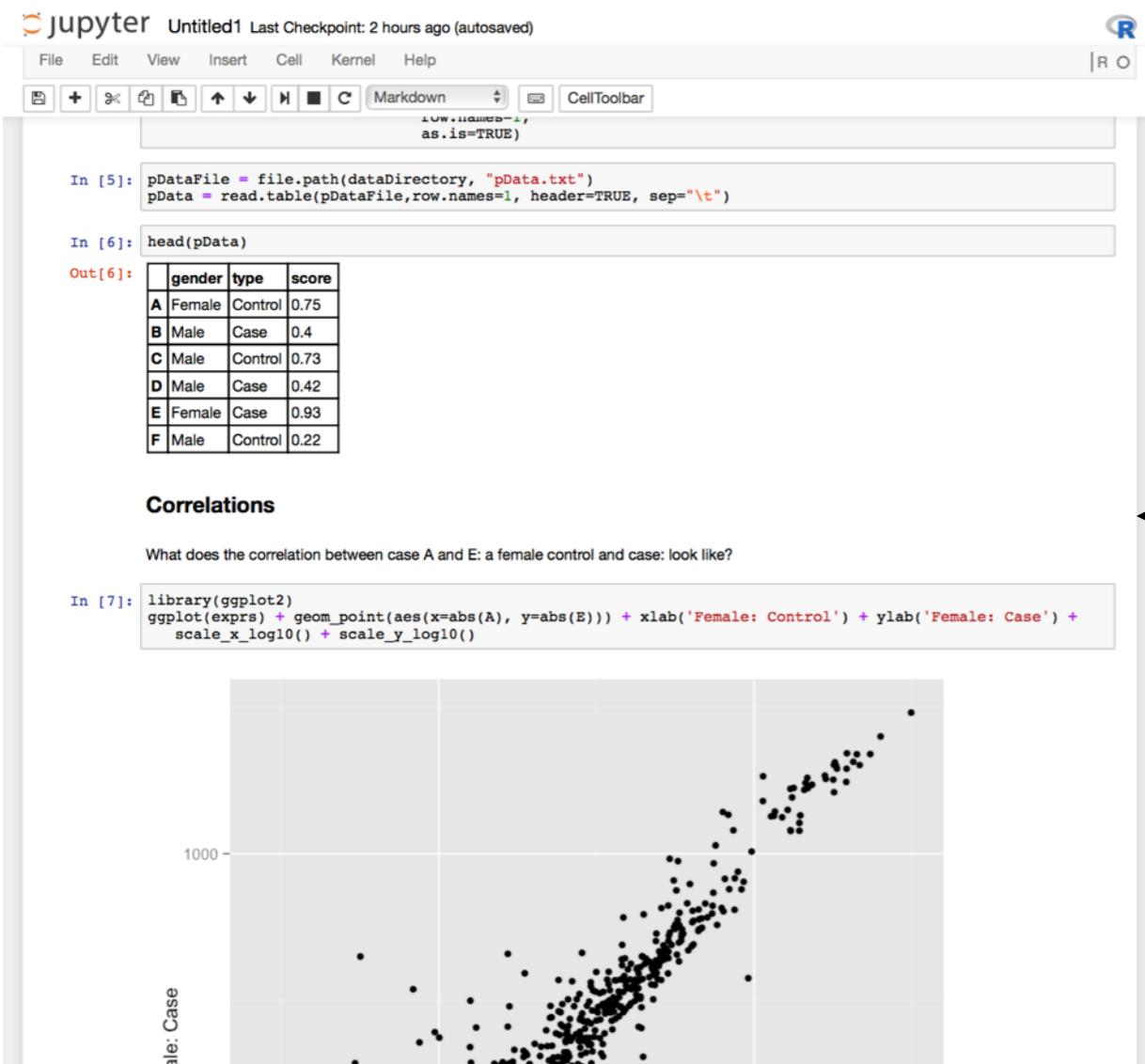


# How does this all work?

- AWS demo...
- What happens if the notebook server dies?
- Demo



# All input+output in ipynb file



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

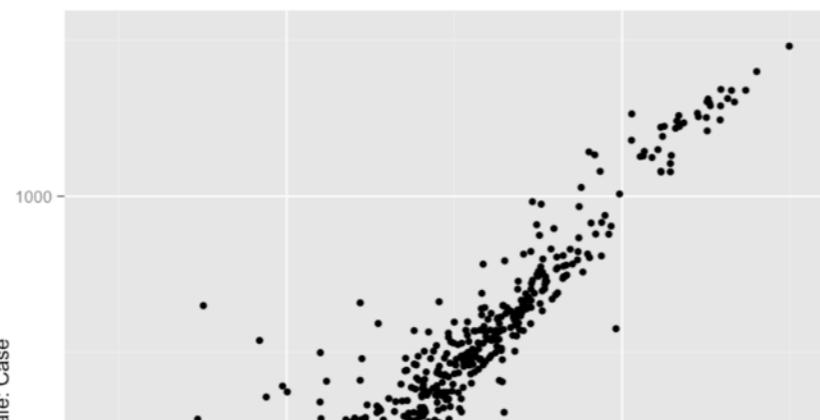
```
In [5]: pDataFile = file.path(dataDirectory, "pData.txt")
pData = read.table(pDataFile, row.names=1, header=TRUE, sep="\t")

In [6]: head(pData)
Out[6]:
   gender type score
A Female Control 0.75
B Male Case 0.4
C Male Control 0.73
D Male Case 0.42
E Female Case 0.93
F Male Control 0.22
```

**Correlations**

What does the correlation between case A and E: a female control and case: look like?

```
In [7]: library(ggplot2)
ggplot(exprs) + geom_point(aes(x=abs(A), y=abs(E))) + xlab('Female: Control') + ylab('Female: Case') +
  scale_x_log10() + scale_y_log10()
```



The right side of the image shows the JSON representation of the Jupyter Notebook content, with a double-headed arrow indicating the equivalence between the two representations.

```
{
  "cells": [
    ...
    "outputs": [],
    "source": [
      "data <- read.csv('http://dursi.ca/content/images/GSE37704_featurecounts.csv')\n",
      "metadata <- read.csv('http://dursi.ca/content/images/GSE37704_metadata.csv')"
    ],
    ...
  ],
  "outputs": [
    {
      "name": "stdout",
      "output_type": "stream",
      "text": [
        "'data.frame': t19808 obs. of 8 variables:\n",
        " $ ensgene : Factor w/ 19808 levels \"ENSG0000000003\",...: 15488 19676 19628 19554 19334 15792 16078 15
        ...
        " $ length : int 918 718 1982 939 939 3214 5539 3395 2833 3424 ...",
        " $ SRR493366: int 0 0 23 0 0 124 1637 120 24 4 ...",
        " $ SRR493367: int 0 0 28 0 0 123 1831 153 48 9 ...",
        " $ SRR493368: int 0 0 29 0 0 205 2383 180 65 16 ...",
        " $ SRR493369: int 0 0 29 0 0 207 1226 236 44 14 ...",
        " $ SRR493370: int 0 0 28 0 0 212 1326 255 48 16 ...",
        " $ SRR493371: int 0 0 46 0 0 258 1504 357 64 16 ..."
      ]
    }
  ],
  ...
}
```

- All the input+output is located in the .ipynb file

# Demo - nb formatting

- 2-Making Nice Notebooks

# “Hands on”: make a nice notebook

- 3 - Pretty Notebook

# nbconvert

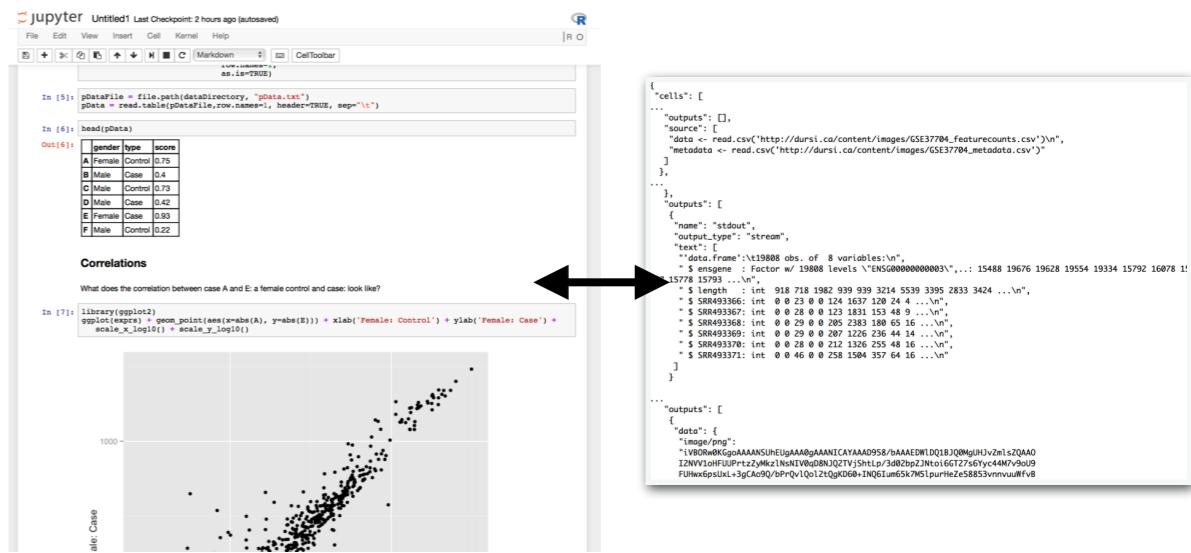
- Once you have a nice notebook, there's a few things you can immediately do with jupyter nbconvert:
  - --to html
  - --to slides
- demo

# Demo: Moving scripts to notebook

- 4 - Moving scripts to notebook.ipynb

# Sharing

- As we've seen, Jupiter notebooks can be pretty self-contained
- (May refer to local or remote data files, etc)
- For others to view or interact with, need to find a place willing to host +render the files (to view), or provide the compute to re-run the analyses (to reproduce)
- We'll look at:
  - SageMathCloud
  - GitHub, nbviewer
  - MyBinder



The screenshot shows a Jupyter Notebook interface with two code cells and their outputs. Cell 5 displays a table of gender, type, and score. Cell 6 shows a scatter plot of 'Case' vs 'Control' scores. A large black arrow points from the right side of the notebook interface towards a detailed JSON representation of the notebook's state, which includes the code cells, their outputs, and the resulting data frame.

```
Cell 5:
```

```
In [5]: pDatafile = file.path(dataDirectory, "Data.txt")
pData = read.table(pDatafile, row.names=1, header=TRUE, sep="\t")
```

```
Out[5]:
```

gender	type	score	
A	Female	Control	0.75
B	Male	Case	0.4
C	Male	Control	0.73
D	Male	Case	0.42
E	Female	Case	0.93
F	Male	Control	0.22

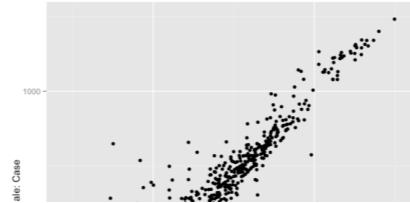
```
Cell 6:
```

```
In [6]: head(pData)
```

```
Correlations
```

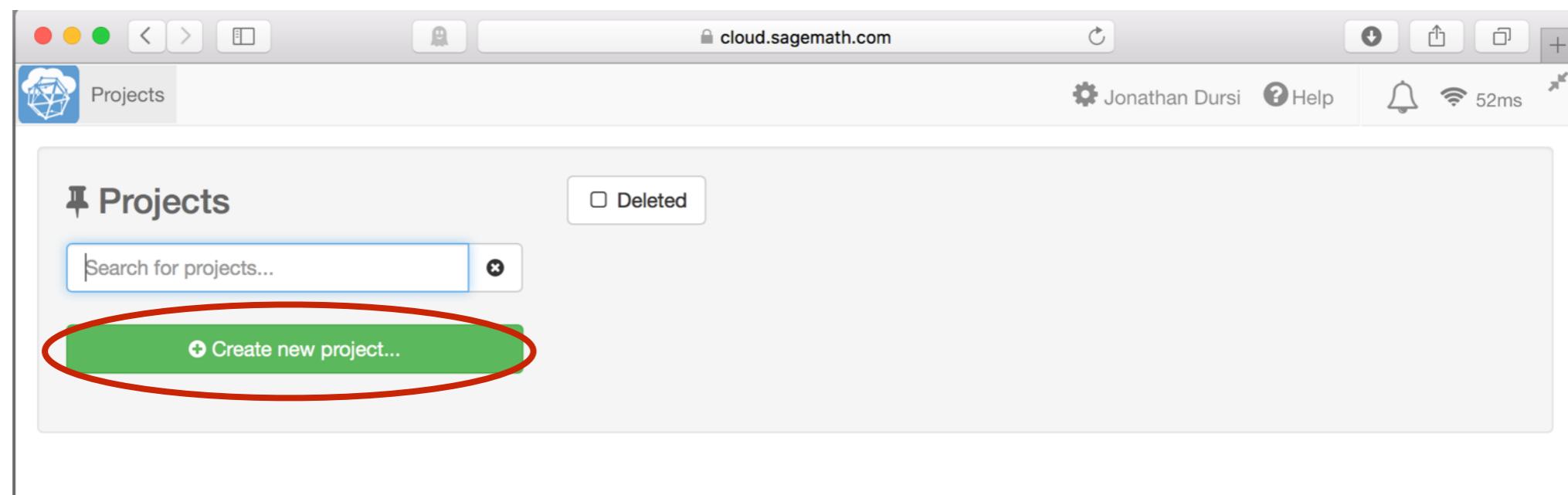
What does the correlation between case A and E: a female control and case: look like?

```
In [7]: library(ggplot2)
ggplot(exprs) + geom_point(aes(x=ab(A), y=ab(E))) + xlab('Female: Control') + ylab('Female: Case') +
  scale_x_log10() + scale_y_log10()
```



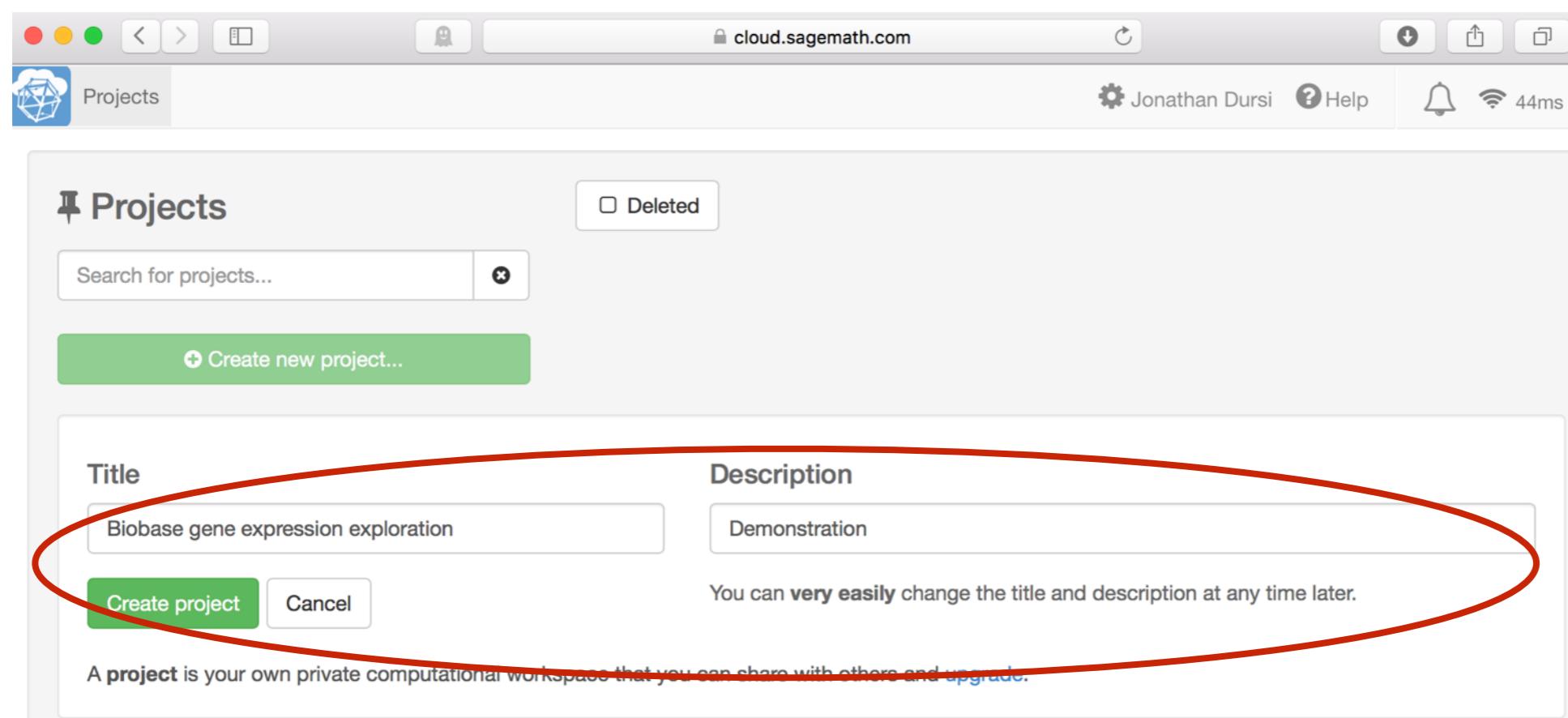
# Sharing: SageMathCloud

- Log into SageMathCloud ([cloud.sagemath.com](https://cloud.sagemath.com)); create new project



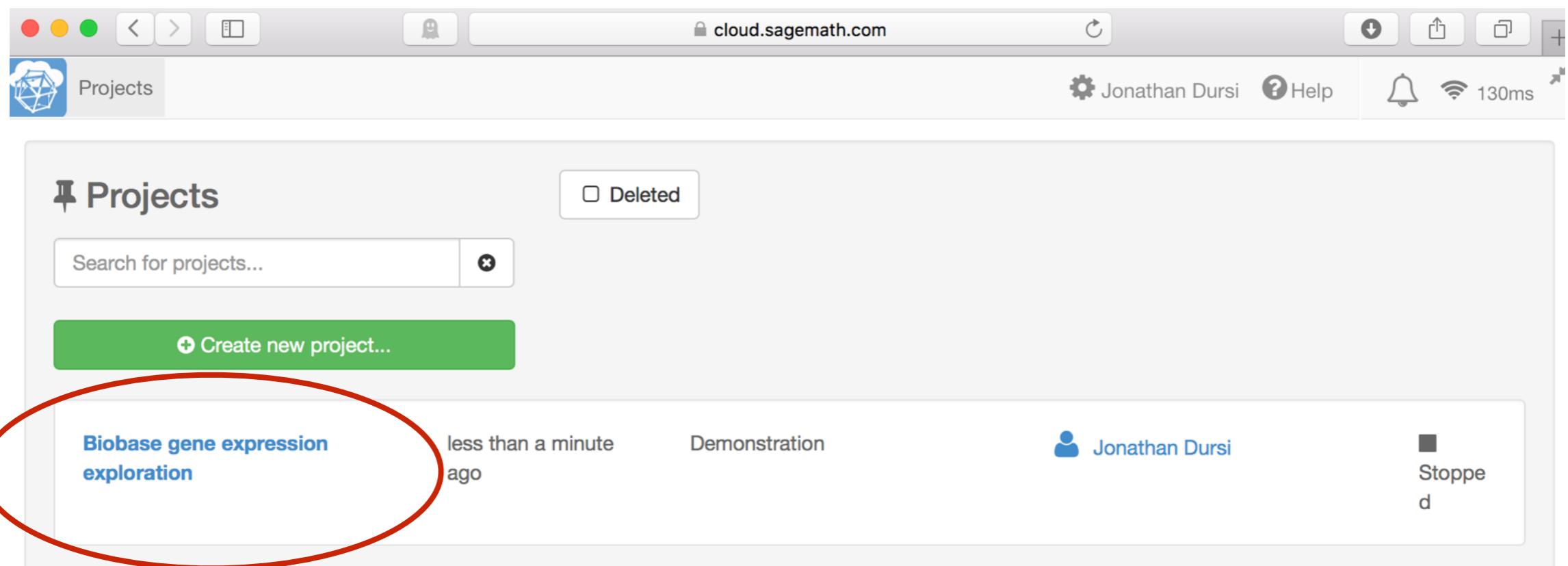
# Sharing: SageMathCloud

- Log into SageMathCloud ([cloud.sagemath.com](https://cloud.sagemath.com)); create new project



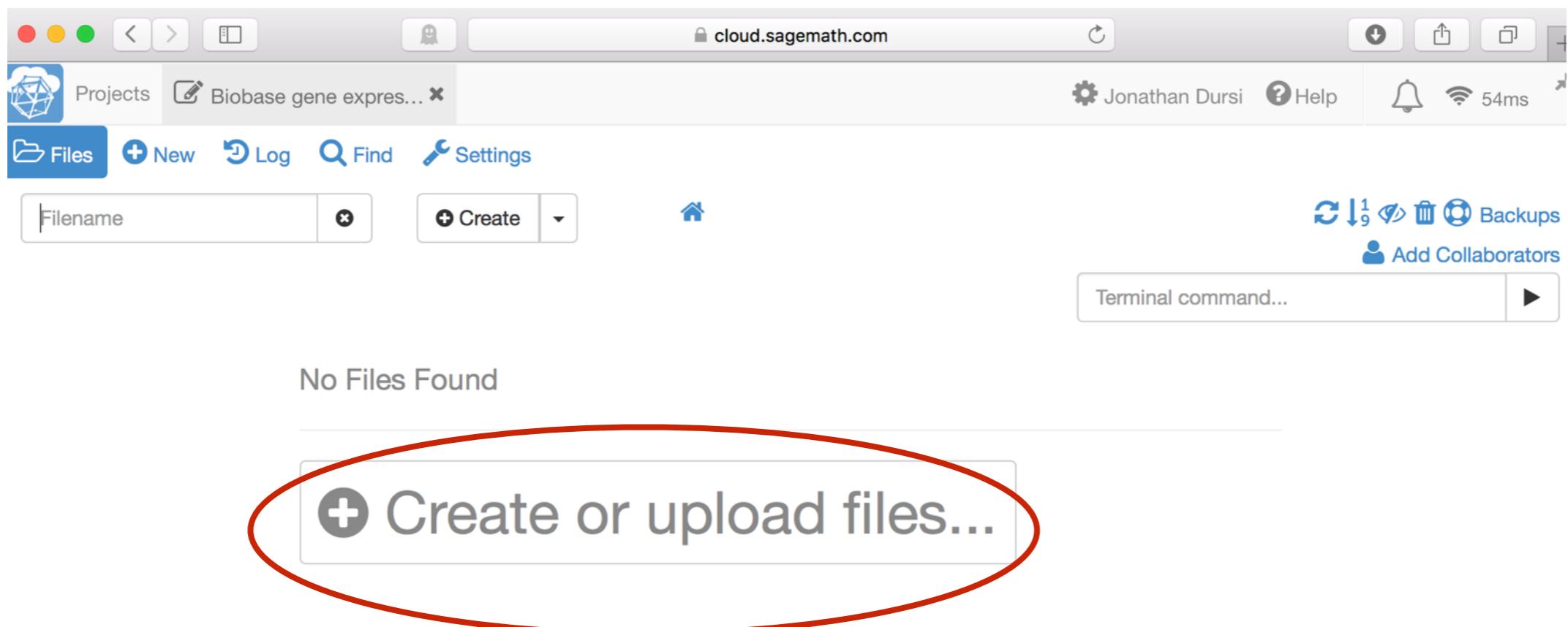
# Sharing: SageMathCloud

- Then Enter the project...



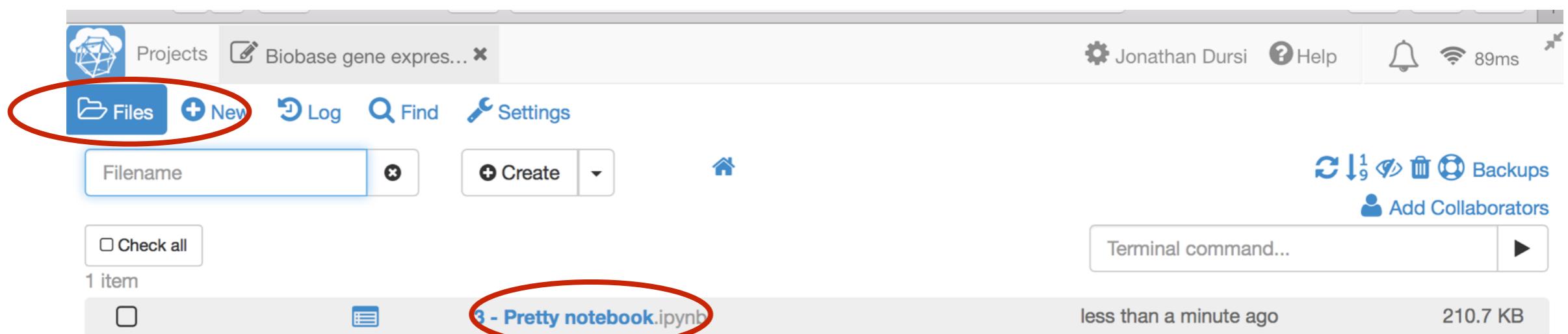
# Sharing: SageMathCloud

- And upload your notebook



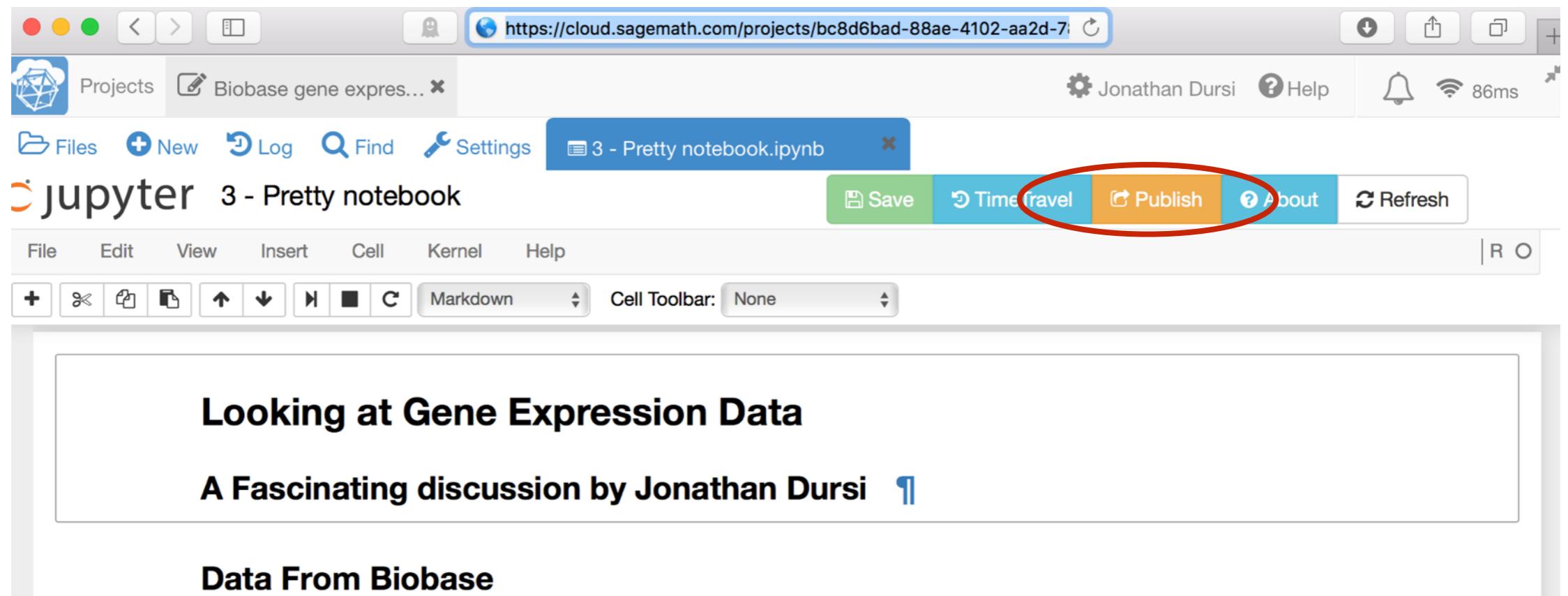
# Sharing: SageMathCloud

- Then go back to files and launch your notebook



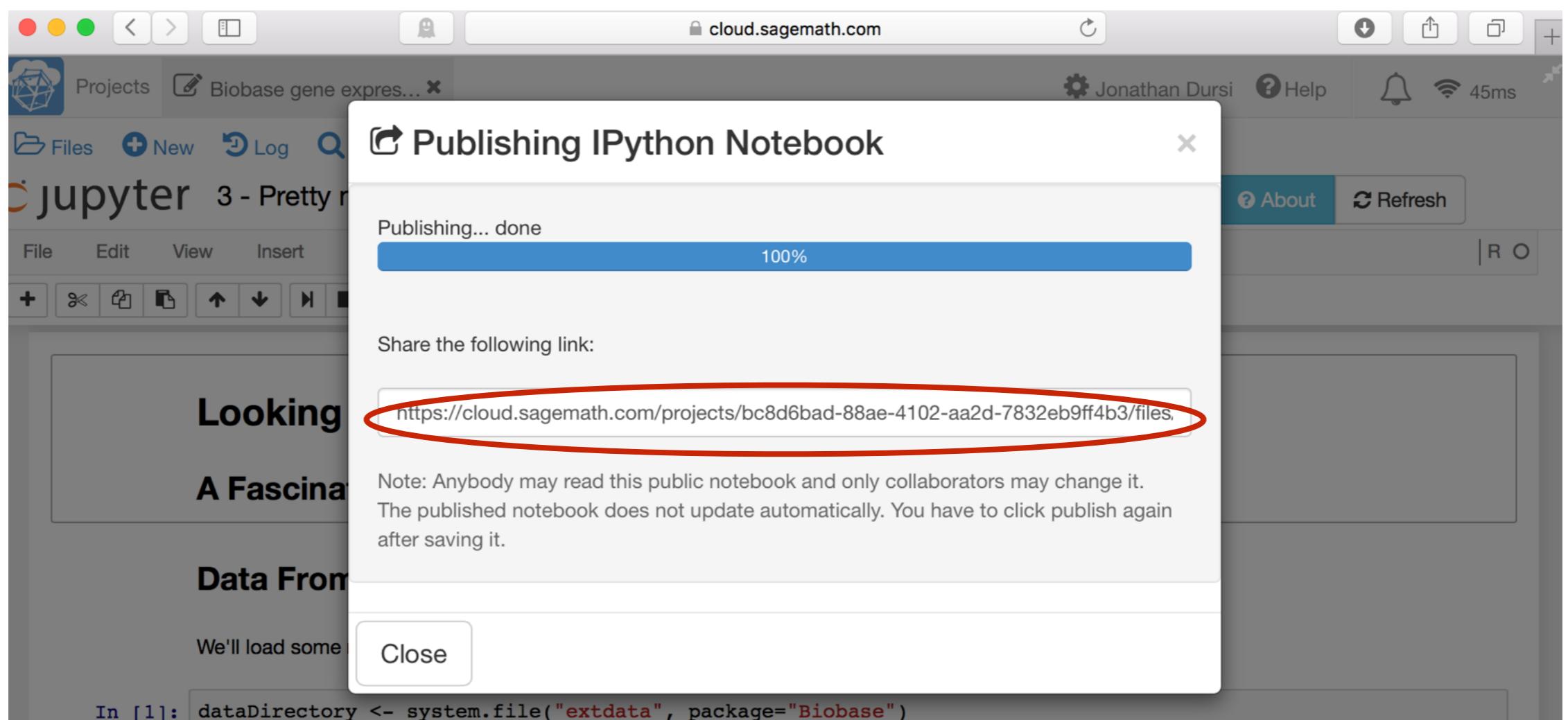
# Sharing: SageMathCloud

- ...and click publish



# Sharing: SageMathCloud

- You can now share it with anyone to view...



# Sharing: SageMathCloud

- ...Or add collaborators to edit.

The screenshot shows the SageMathCloud web interface. At the top, there's a navigation bar with icons for Projects, Biobase gene expres..., and a user profile for Jonathan Dursi. Below the bar, the title "Settings and configuration" is displayed next to a wrench icon. The main area is divided into several sections:

- Title and description:** Contains fields for "Title" (Biobase gene expression exploration) and "Description" (Demonstration).
- Project usage and quotas:** Includes a button to "Adjust your quotas...".
- Collaborators:** A section with a red oval around it. It contains a sub-section titled "Collaborators can modify anything in this project, except backups. They can add and remove other collaborators, but cannot remove owners." Below this is a "Add collaborators" field with a search bar ("Search by name or email address...") and a list of existing collaborators, starting with "Jonathan Dursi (6 minutes ago) (owner)".



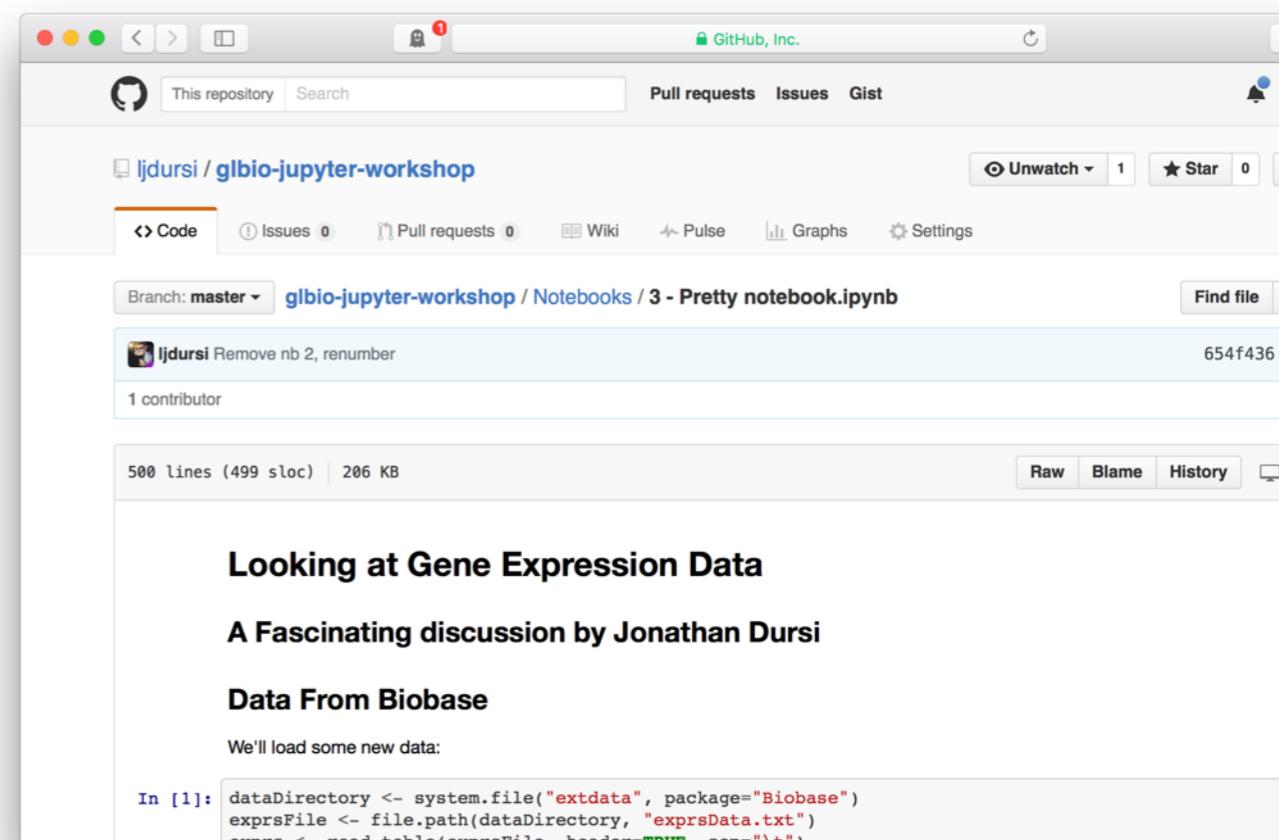
# Sharing: SageMathCloud

- Can share to-do lists, data files, writeups: but many features require \$7/mo subscription

The screenshot shows the SageMathCloud web interface. At the top, there's a navigation bar with 'Projects' (highlighted), a project name 'Biobase gene expres...', user info 'Jonathan Dursi', 'Help', a bell icon, and network status '78ms'. Below the bar, there are tabs for 'Files', 'New' (which is blue and highlighted), 'Log', 'Find', 'Settings', and a notebook titled '3 - Pretty notebook.ipynb'. A large blue button labeled '+ Create new files in home directory of project' is centered. To its right is a form for 'Name your file, folder or paste in a link' containing the text '2016-05-15-120329'. Below this is a section titled 'Select the type' with several options: 'SageMath Worksheet', 'Jupyter Notebook', 'File' (with a dropdown arrow), 'Folder', 'LaTeX Document' (circled in red), 'Terminal', 'Task List' (circled in red), 'Manage a Course', and 'Create a Chatroom'. At the bottom of this section is a note: 'Download from Internet (internet access blocked -- see project settings)'.

# GitHub

- Widely used for collaborating on scientific software
- Will render uploaded .ipynb files automatically
- Can get unlimited private repositories for researchers: <https://education.github.com>
- plus other stuff for students

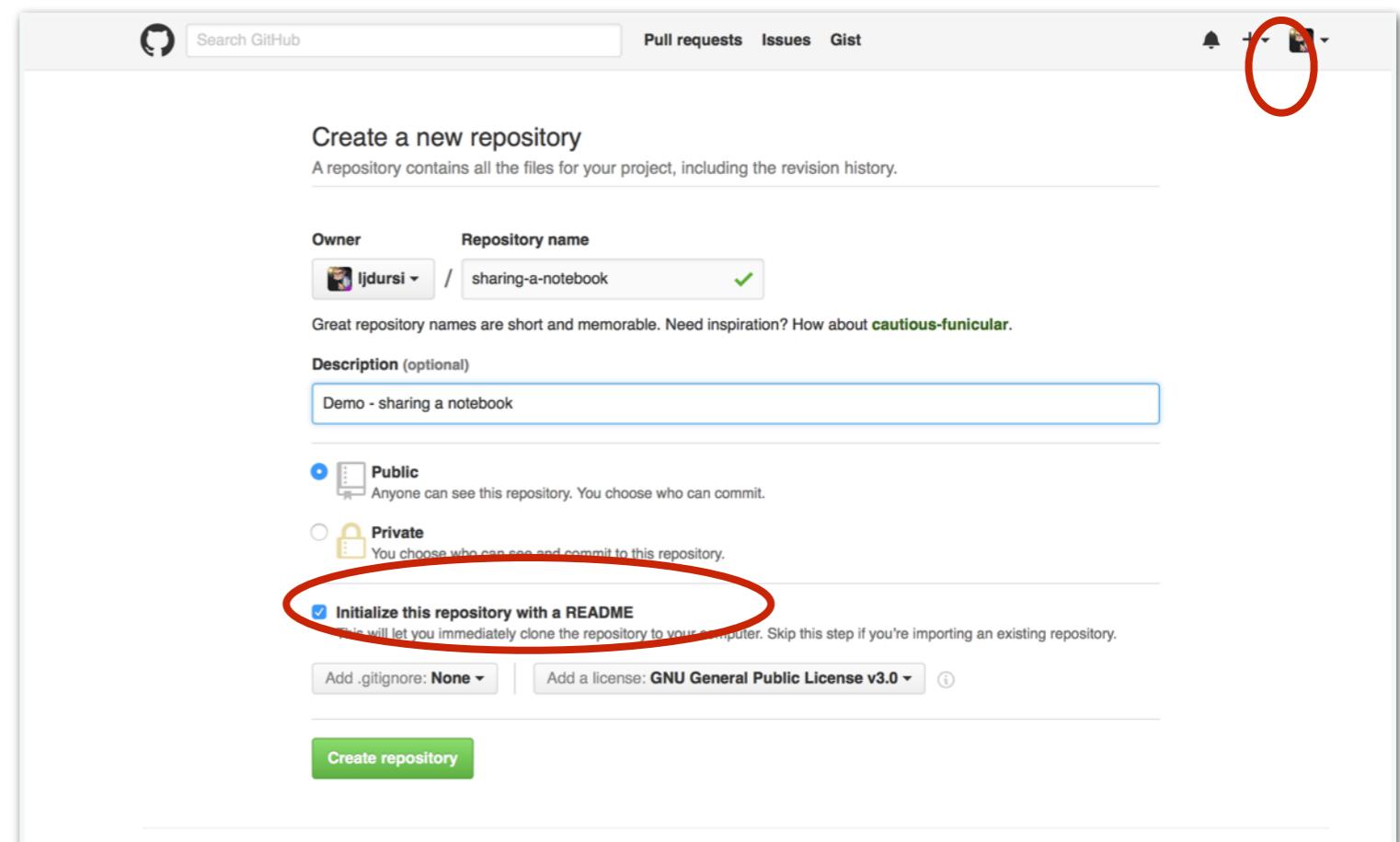


A screenshot of a GitHub repository page for 'ljdursi / glbio-jupyter-workshop'. The repository has 0 issues, 0 pull requests, 0 wiki pages, 0 pulses, 0 graphs, and 0 settings. The branch is master, and the file is 'glibio-jupyter-workshop / Notebooks / 3 - Pretty notebook.ipynb'. The commit hash is 654f436. The notebook content includes:

```
Looking at Gene Expression Data  
A Fascinating discussion by Jonathan Dursi  
Data From Biobase  
We'll load some new data:  
In [1]: dataDirectory <- system.file("extdata", package="Biobase")  
exprsFile <- file.path(dataDirectory, "exprsData.txt")  
exprs <- read.table(exprsFile, header=TRUE, sep="\t")
```

# Sharing in github

- Easy way: go to [github.com](https://github.com) and create a repo:
- Fill in the details and initialize a README



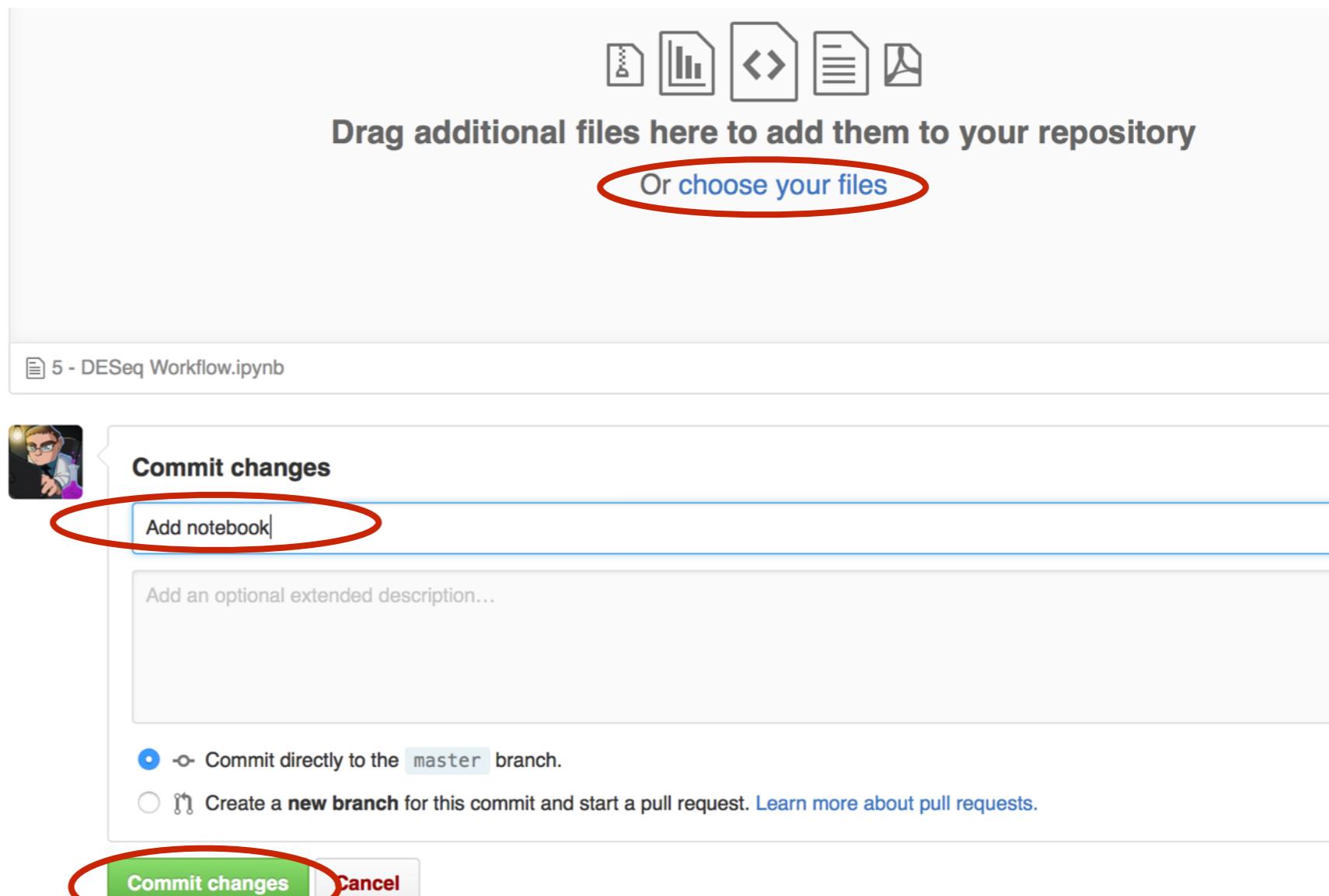
# Sharing in github

- Now upload your notebook:

The screenshot shows a GitHub repository page for 'ljdursi / sharing-a-notebook'. The repository has 1 commit, 1 branch, 0 releases, and 1 contributor. The 'Upload files' button is highlighted with a red circle. The repository contains files: LICENSE, README.md, and README.md (a preview). The README.md file content is displayed as 'sharing-a-notebook'.

# Sharing in github

- Now upload your notebook:



# Sharing in github

- And just view the file in the repo

Branch: master → [sharing-a-notebook / 5 - DESeq Workflow.ipynb](#) Find file Copy path  
7e2fb1f 20 seconds ago

ljdursi Add notebook 1 contributor

803 lines (802 sloc) | 733 KB Raw Blame History

## RNA-seq analysis with DESeq2

Largely from Stephen Turner, @genetics\_blog

Using data from GSE37704, with processed data available on Figshare DOI: 10.6084/m9.figshare.1601975. This dataset has six samples from GSE37704, where expression was quantified by either:

- Mapping to GRCh38 using STAR then counting reads mapped to genes with featureCounts under the union-intersection model, or
- Alignment-free quantification using Sailfish, summarized at the gene level using the GRCh38 GTFile. Both datasets are restricted to protein-coding genes only.

We're using the Sailfish data.

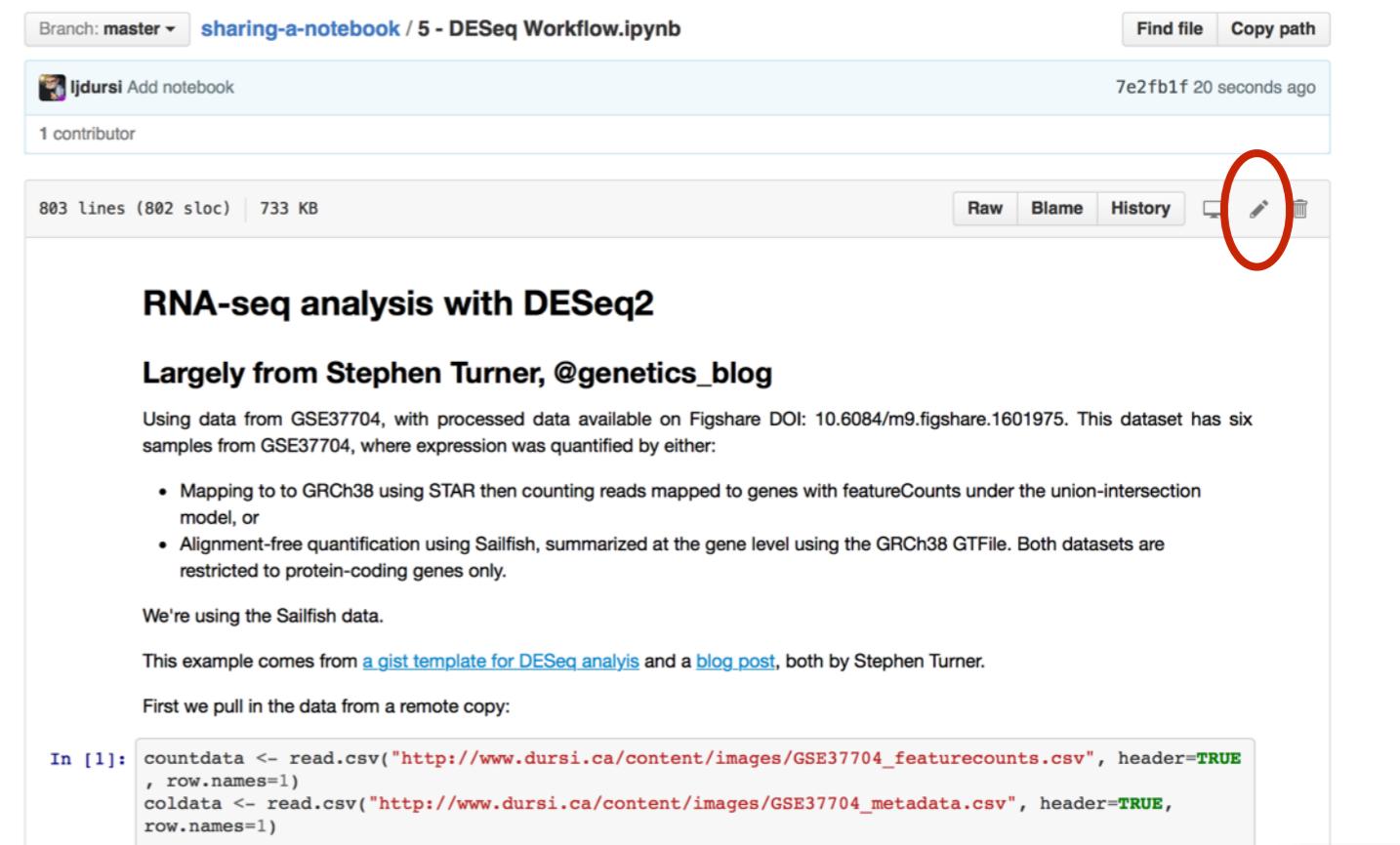
This example comes from [a gist template for DESeq analysis](#) and a [blog post](#), both by Stephen Turner.

First we pull in the data from a remote copy:

```
In [1]: countdata <- read.csv("http://www.dursi.ca/content/images/GSE37704_featurecounts.csv", header=TRUE,  
, row.names=1)  
coldata <- read.csv("http://www.dursi.ca/content/images/GSE37704_metadata.csv", header=TRUE,  
row.names=1)
```

# Sharing in github

- Let's give it a better name; click on the pencil to edit the file:



Branch: master [sharing-a-notebook](#) / 5 - DESeq Workflow.ipynb [Find file](#) [Copy path](#)

ljdursi Add notebook 7e2fb1f 20 seconds ago

1 contributor

803 lines (802 sloc) | 733 KB [Raw](#) [Blame](#) [History](#) [Edit](#)

**RNA-seq analysis with DESeq2**

**Largely from Stephen Turner, @genetics\_blog**

Using data from GSE37704, with processed data available on Figshare DOI: 10.6084/m9.figshare.1601975. This dataset has six samples from GSE37704, where expression was quantified by either:

- Mapping to GRCh38 using STAR then counting reads mapped to genes with featureCounts under the union-intersection model, or
- Alignment-free quantification using Sailfish, summarized at the gene level using the GRCh38 GTFile. Both datasets are restricted to protein-coding genes only.

We're using the Sailfish data.

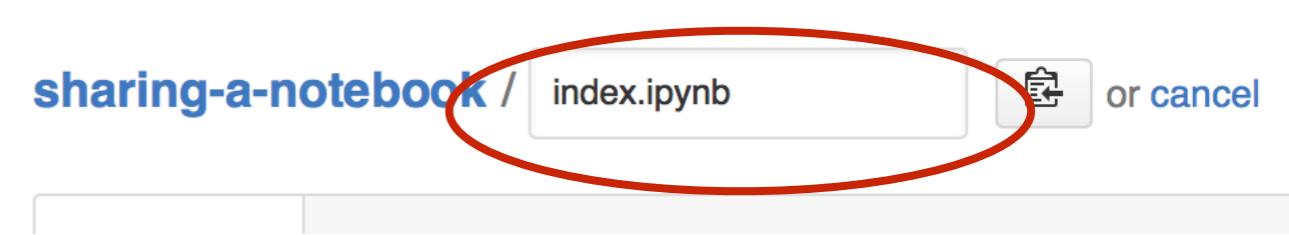
This example comes from [a gist template for DESeq analysis](#) and a [blog post](#), both by Stephen Turner.

First we pull in the data from a remote copy:

```
In [1]: countdata <- read.csv("http://www.dursi.ca/content/images/GSE37704_featurecounts.csv", header=TRUE, row.names=1)
coldata <- read.csv("http://www.dursi.ca/content/images/GSE37704_metadata.csv", header=TRUE, row.names=1)
```

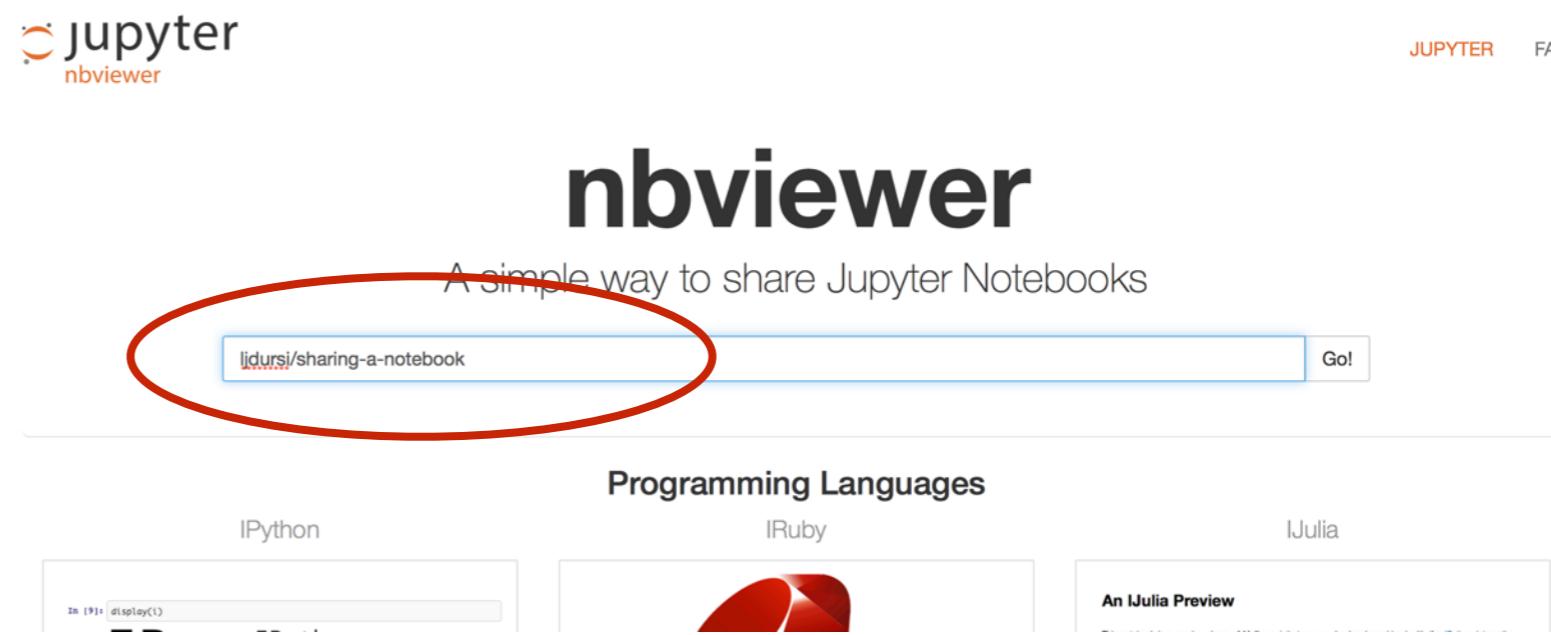
# Sharing in github

- and then  
rename it to  
**index.ipynb**



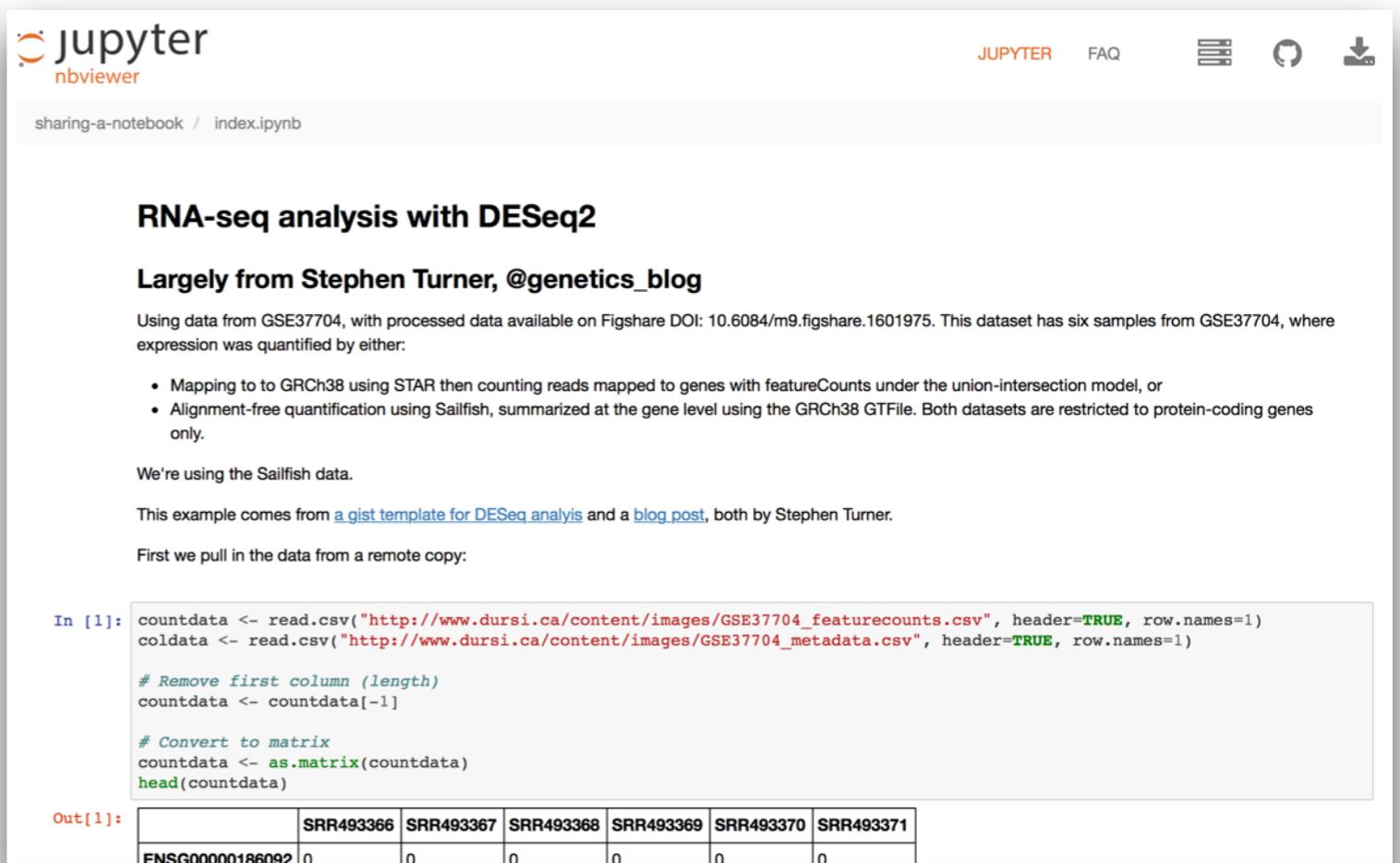
# nbviewer.jupyter.org

- Once the file is hosted somewhere (github, or anywhere) nbviewer can view it
- Enter name of repo:  
ljdursi/sharing-a-notebook



# nbviewer.jupyter.org

- Produces slightly nicer results
- You can share the link it goes to directly, or download the notebook



The screenshot shows a Jupyter notebook interface with the title "sharing-a-notebook / index.ipynb". The main content is a blog post about "RNA-seq analysis with DESeq2" by Stephen Turner (@genetics\_blog). It discusses using data from GSE37704 and provides code for reading CSV files and creating a matrix. The output cell shows a table with a single row of data.

```
In [1]: countdata <- read.csv("http://www.dursi.ca/content/images/GSE37704_featurecounts.csv", header=TRUE, row.names=1)
coldata <- read.csv("http://www.dursi.ca/content/images/GSE37704_metadata.csv", header=TRUE, row.names=1)

# Remove first column (length)
countdata <- countdata[-1]

# Convert to matrix
countdata <- as.matrix(countdata)
head(countdata)
```

	SRR493366	SRR493367	SRR493368	SRR493369	SRR493370	SRR493371
ENSG00000186092	0	0	0	0	0	0