Laboratório Interinstitucional
de e-Astronomia

Tutorial Jupyter Notebook - RNP

Julia Gschwend

24/06/2020

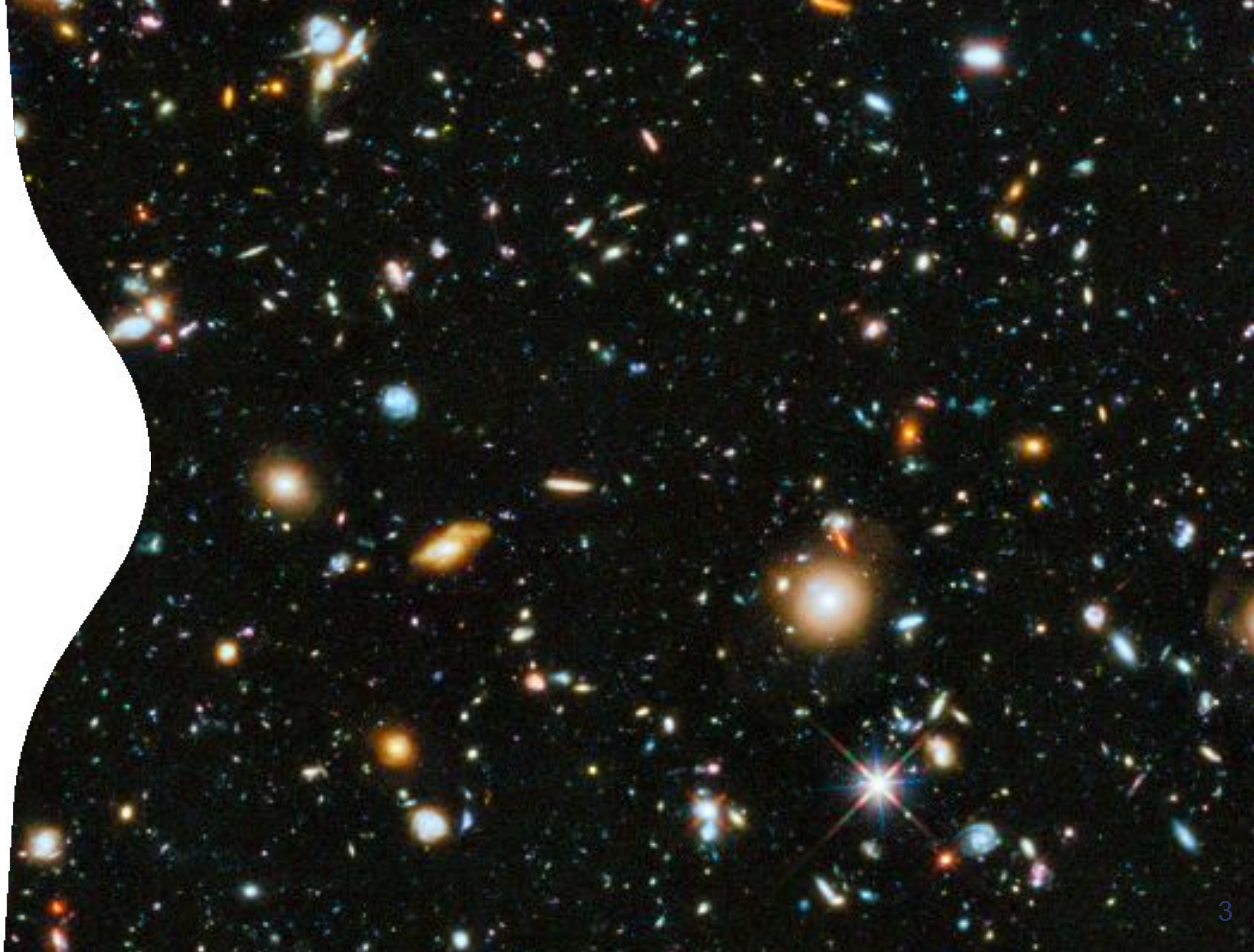
Agenda

- Parte I - Introdução (15 min)
- Parte II - Live demo: primeiros passos (30 min)
- Parte III - Live demo: Aplicações (30 min)

https://indico.rnp.br/event/21/

Parte 1

- ➢ Introdução
- ➢ Jupyter Hub @LIneA

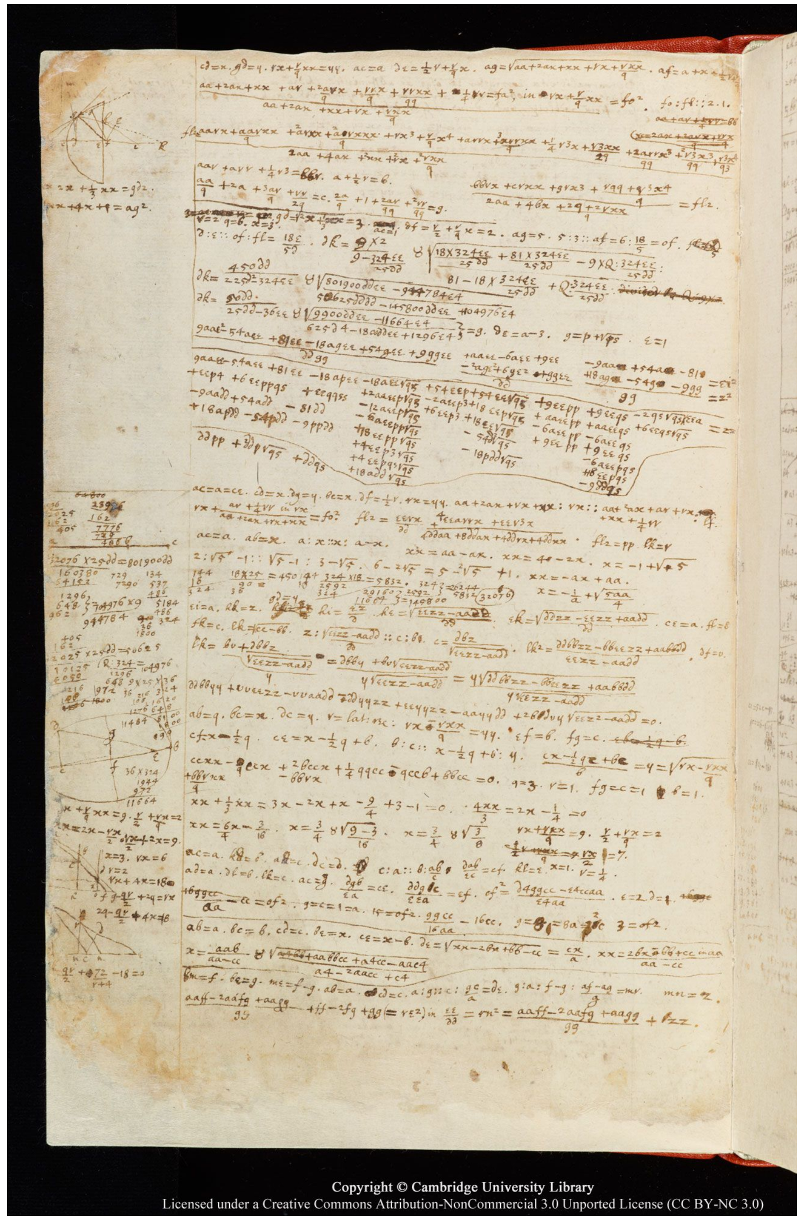


Introdução - conceitos básicos

Caderno de laboratório

- Documentação da atividade científica
    - Hipóteses
    - Testes, experimentos
    - Análise e interpretação dos resultados
- Fonte para as publicações
- Proteção da propriedade intelectual
- Regras definidas pela instituição
  (ex.: caderno brochura, numeração de páginas, *peer review*, etc)

Caderno de observações de Galileu Galilei de 1610 (fonte: Museo Galileo)



Caderno de Sir Isaac Newton de 1644 (fonte: Cambridge University Library)

Introdução - conceitos básicos

Caderno de laboratório, eletrônico

"Uma aplicação web que permite executar códigos, apresentar resultados com texto formatado, equações matemáticas, imagens, gráficos e ferramentas interativas de forma integrada."

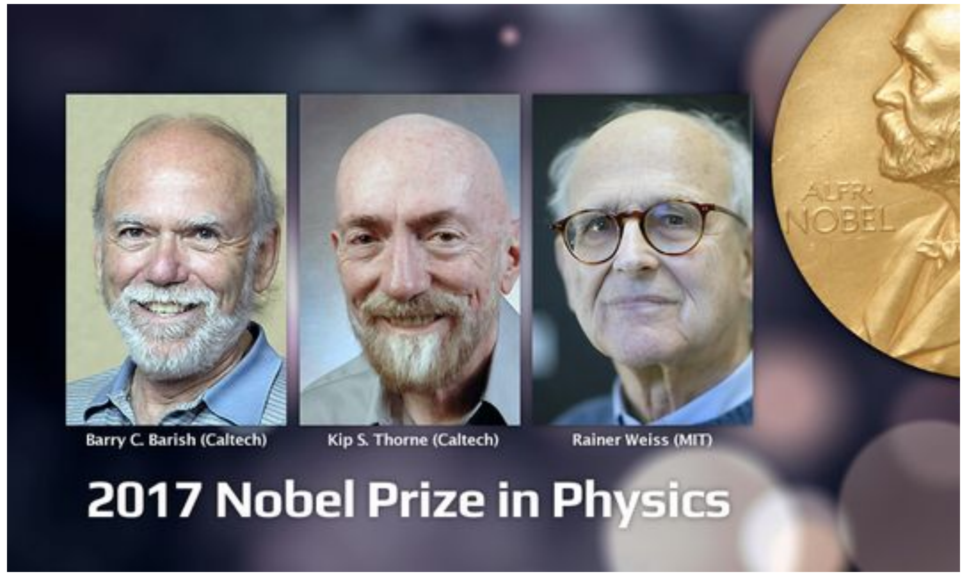
- Fornecedores
  - Jupyter Notebook
  - Colab Notebook (Google)
  - outros?
- Funcionalidades extras
  - Processamento dos dados
  - Visualização gráfica
  - Publicação de resultados
- Vantagens óbvias
  - Padronização
  - Backup, segurança
  - Colaborações simultâneas
- "*Open Notebook Science*"
  - *open source*
  - *open data*
  - *citizen science*

Introdução - conceitos básicos

Exemplo:

LIGO - GW detection

LIGO Hanford Observatory (image credit to LIGO Lab/Caltech/MIT).



[Last updated July 18, 2017]

This tutorial is no longer maintained. For an updated version, including additional binary black hole events, see the Binary Black Hole Events tutorial at https://losc.ligo.org/tutorials/.

SIGNAL PROCESSING WITH GW150914 OPEN DATA

Welcome! This ipython notebook (or associated python script GW150914_tutorial.py) will go through some typical signal processing tasks on strain time-series data associated with the LIGO GW150914 data release from the LIGO Open Science Center (LOSC):

- https://losc.ligo.org/events/GW150914/
- View the tutorial as a web page - https://losc.ligo.org/s/events/GW150914/GW150914_tutorial.html/
- Download the tutorial as a python script - https://losc.ligo.org/s/events/GW150914/GW150914_tutorial.py/
- Download the tutorial as iPython Notebook - https://losc.ligo.org/s/events/GW150914/GW150914_tutorial.ipynb/

To begin, download the ipython notebook, readligo.py, and the data files listed below, into a directory / folder, then run it. Or you can run the python script GW150914_tutorial.py. You will need the python packages: numpy, scipy, matplotlib, h5py.

On Windows, or if you prefer, you can use a python development environment such as Anaconda (https://www.continuum.io/why-anaconda) or Enthought Canopy (https://www.enthought.com/products/canopy/).

Questions, comments, suggestions, corrections, etc: email losc@ligo.caltech.edu

v20160208b

Intro to signal processing

This tutorial assumes that you know python well enough.

If you know how to use "ipython notebook", use the GW150914_tutorial.ipynb file. Else, you can use the GW150914_tutorial.py script.

This tutorial assumes that you know a bit about signal processing of digital time series data (or want to learn!). This includes power spectral densities, spectrograms, digital filtering, whitening, audio manipulation. This is a vast and complex set of topics, but we will cover many of the basics in this tutorial.

If you are a beginner, here are some resources from the web:

- http://101science.com/dsp.htm
- https://georgemdallas.wordpress.com/2014/05/14/wavelets-4-dummies-signal-processing-fourier-transforms-and-heisenberg/
- https://en.wikipedia.org/wiki/Signal_processing
- https://en.wikipedia.org/wiki/Spectral_density
- https://en.wikipedia.org/wiki/Spectrogram
- http://greenteapress.com/thinkdsp/
- https://en.wikipedia.org/wiki/Digital_filter

And, well, lots more - google it!

Download the data

- Download the data files from LOSC:
- We will use the hdf5 files, both H1 and L1, with durations of 32 and 4096 seconds around GW150914, sampled at 16384 and 4096 Hz :
  - https://losc.ligo.org/s/events/GW150914/H-H1_LOSC_4_V1-1126259446-32.hdf5
  - https://losc.ligo.org/s/events/GW150914/L-L1_LOSC_4_V1-1126259446-32.hdf5
  - https://losc.ligo.org/s/events/GW150914/H-H1_LOSC_16_V1-1126259446-32.hdf5
  - https://losc.ligo.org/s/events/GW150914/L-L1_LOSC_16_V1-1126259446-32.hdf5
  - https://losc.ligo.org/s/events/GW150914/GW150914_4_NR_waveform.txt
- Download the python functions to read the data: https://losc.ligo.org/s/sample_code/readligo.py
- From a unix/mac-osx command line, you can use wget; for example,
  - wget https://losc.ligo.org/s/events/GW150914/H-H1_LOSC_4_V1-1126257414-4096.hdf5
- Put these files in your current directory / folder. Don't mix any other LOSC data files in this directory, or readligo.py may get confused.

Here,

- "H-H1" means that the data come from the LIGO Hanford Observatory site and the LIGO "H1" datector;
- the "4" means the strain time-series data are (down-)sampled from 16384 Hz to 4096 Hz;
- the "V1" means version 1 of this data release;
- "1126257414-4096" means the data starts at GPS time 1126257414 (Mon Sep 14 09:16:37 GMT 2015), duration 4096 seconds;
  - NOTE: GPS time is number of seconds since Jan 6, 1980 GMT. See http://www.oc.nps.edu/oc2902w/gps/timsys.html or https://losc.ligo.org/gps/
- the filetype "hdf5" means the data are in hdf5 format: https://www.hdfgroup.org/HDF5/

Note that the the 4096 second long files at 16384 Hz sampling rate are fairly big files (125 MB). You won't need them for this tutorial:

- https://losc.ligo.org/s/events/GW150914/H-H1_LOSC_4_V1-1126257414-4096.hdf5
- https://losc.ligo.org/s/events/GW150914/L-L1_LOSC_4_V1-1126257414-4096.hdf5
- https://losc.ligo.org/s/events/GW150914/H-H1_LOSC_16_V1-1126257414-4096.hdf5
- https://losc.ligo.org/s/events/GW150914/L-L1_LOSC_16_V1-1126257414-4096.hdf5

In [2]:
```
# Standard python numerical analysis imports:
import numpy as np
from scipy import signal
from scipy.interpolate import interp1d
from scipy.signal import butter, filtfilt, iirdesign, zpk2tf, freqz

# the ipython magic below must be commented out in the .py file, since it doesn't work.
%matplotlib inline
%config InlineBackend.figure_format = 'retina'
import matplotlib.pyplot as plt
import matplotlib.mlab as mlab
import h5py

# LIGO-specific readligo.py
import readligo as rl
```

NOTE that in general, LIGO strain time series data has gaps (filled with NaNs) when the detectors are not taking valid ("science quality") data. Analyzing these data requires the user to loop over "segments" of valid data stretches. In https://losc.ligo.org/segments/ we provide example code to do this.

However, the 4096 seconds of released data around GW150914 is one unbroken segment, with no gaps. So for now, we will read it all in and treat it as one valid data segment, ignoring the extra complexity mentioned above.

This won't work for other LOSC data releases! See https://losc.ligo.org/segments/ for a more general way to find valid data segments in LOSC data.

In [3]:
```
#----------------------------------------------------------------
# Load LIGO data from a single file
#----------------------------------------------------------------
# First from H1
fn_H1 = 'H-H1_LOSC_4_V1-1126259446-32.hdf5'
strain_H1, time_H1, chan_dict_H1 = rl.loaddata(fn_H1, 'H1')
# and then from L1
fn_L1 = 'L-L1_LOSC_4_V1-1126259446-32.hdf5'
strain_L1, time_L1, chan_dict_L1 = rl.loaddata(fn_L1, 'L1')

# sampling rate:
fs = 4096
# both H1 and L1 will have the same time vector, so:
time = time_H1
# the time sample interval (uniformly sampled!)
dt = time[1] - time[0]
```

Adding a numerical relativity template

Now let's also read in a theoretical (numerical relativity) template, generated with parameters favored by the output from the GW150914 parameter estimation (see the GW150914 detection paper, https://dcc.ligo.org/P150914/public).

This NR template corresponds to the signal expected from a pair of black holes with masses of around 36 and 29 solar masses, merging into a single black hole of 62 solar masses, at a distance of around 410 Mpc.

You can fetch the template time series from the following URL, and put it in your working directory / folder:

- https://losc.ligo.org/s/events/GW150914/GW150914_4_NR_waveform.txt

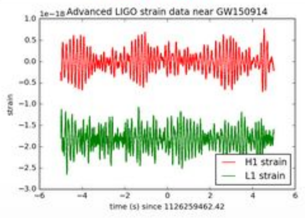
In [4]:
```
# read in the NR template
NRtime, NR_H1 = np.genfromtxt('GW150914_4_NR_waveform.txt').transpose()
```

First look at the data from H1 and L1

In [5]:
```
# First, let's look at the data and print out some stuff:

# this doesn't seem to work for scientific notation:
# np.set_printoptions(precision=4)

print '  time_H1: len, min, mean, max = ', \
   len(time_H1), time_H1.min(), time_H1.mean(), time_H1.max()
print 'strain_H1: len, min, mean, max = ', \
   len(strain_H1), strain_H1.min(),strain_H1.mean(),strain_H1.max()
print 'strain_L1: len, min, mean, max = ', \
   len(strain_L1), strain_L1.min(),strain_L1.mean(),strain_L1.max()

#What's in chan_dict? See https://losc.ligo.org/archive/dataset/GW150914/
bits = chan_dict_H1['DATA']
print 'H1     DATA: len, min, mean, max = ', len(bits), bits.min(),bits.mean(),bits.max()
bits = chan_dict_H1['CBC_CAT1']
print 'H1 CBC_CAT1: len, min, mean, max = ', len(bits), bits.min(),bits.mean(),bits.max()
bits = chan_dict_H1['CBC_CAT2']
print 'H1 CBC_CAT2: len, min, mean, max = ', len(bits), bits.min(),bits.mean(),bits.max()
bits = chan_dict_L1['DATA']
print 'L1     DATA: len, min, mean, max = ', len(bits), bits.min(),bits.mean(),bits.max()
bits = chan_dict_L1['CBC_CAT1']
print 'L1 CBC_CAT1: len, min, mean, max = ', len(bits), bits.min(),bits.mean(),bits.max()
bits = chan_dict_L1['CBC_CAT2']
print 'L1 CBC_CAT2: len, min, mean, max = ', len(bits), bits.min(),bits.mean(),bits.max()
print 'In both H1 and L1, all 32 seconds of data are present (DATA=1), '
print "and all pass data quality (CBC_CAT1=1 and CBC_CAT2=1)."
```

```
  time_H1: len, min, mean, max =  131072 1126259446.0 1126259462.0 1126259478.0
strain_H1: len, min, mean, max =  131072 -7.11996338709e-19 8.73279794057e-23 7.71483633765e-19
strain_L1: len, min, mean, max =  131072 -2.6788089173e-18 -1.82870749189e-18 -7.69266177024e-19
H1     DATA: len, min, mean, max =  32 1 1.0 1
H1 CBC_CAT1: len, min, mean, max =  32 1 1.0 1
H1 CBC_CAT2: len, min, mean, max =  32 1 1.0 1
L1     DATA: len, min, mean, max =  32 1 1.0 1
L1 CBC_CAT1: len, min, mean, max =  32 1 1.0 1
L1 CBC_CAT2: len, min, mean, max =  32 1 1.0 1
In both H1 and L1, all 32 seconds of data are present (DATA=1),
and all pass data quality (CBC_CAT1=1 and CBC_CAT2=1).
```

In [6]:
```
# plot +- 5 seconds around the event:
tevent = 1126259462.422          # Mon Sep 14 09:50:45 GMT 2015
deltat = 5.                      # seconds around the event
# index into the strain time series for this time interval:
indxt = np.where((time_H1 >= tevent-deltat) & (time_H1 < tevent+deltat))

plt.figure()
plt.plot(time_H1[indxt]-tevent,strain_H1[indxt],'r',label='H1 strain')
plt.plot(time_L1[indxt]-tevent,strain_L1[indxt],'g',label='L1 strain')
plt.xlabel('time (s) since '+str(tevent))
plt.ylabel('strain')
plt.legend(loc='lower right')
plt.title('Advanced LIGO strain data near GW150914')
plt.savefig('GW150914_strain.png')
```

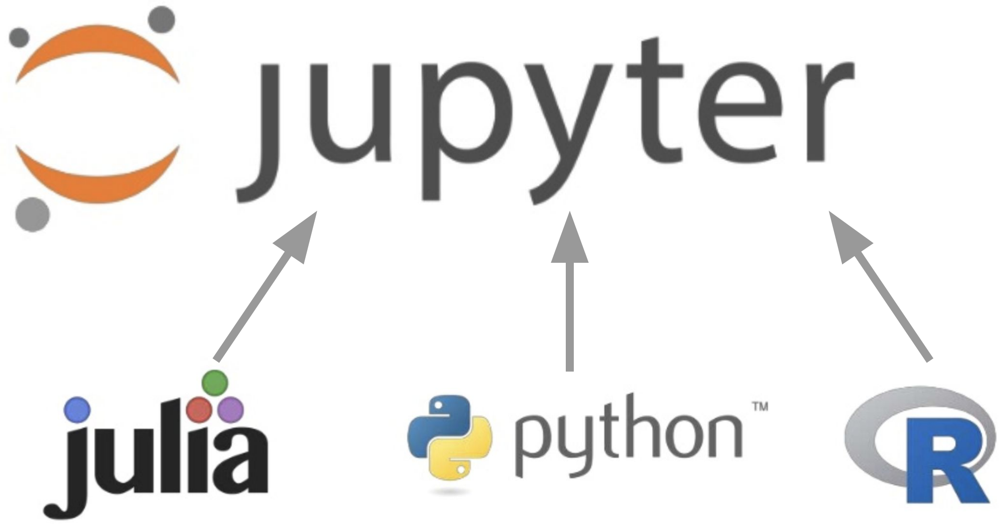


The data are dominated by **low frequency noise**; there is no way to see a signal here, without some signal processing.

There are very low frequency oscillations that are putting the mean of the L1 strain at -2.0e-18 at the time around this event, so it appears offset from the H1 strain. These low frequency oscillations are essentially ignored in LIGO data analysis (see bandpassing, below)

Introdução - conceitos básicos

Jupyter Notebook

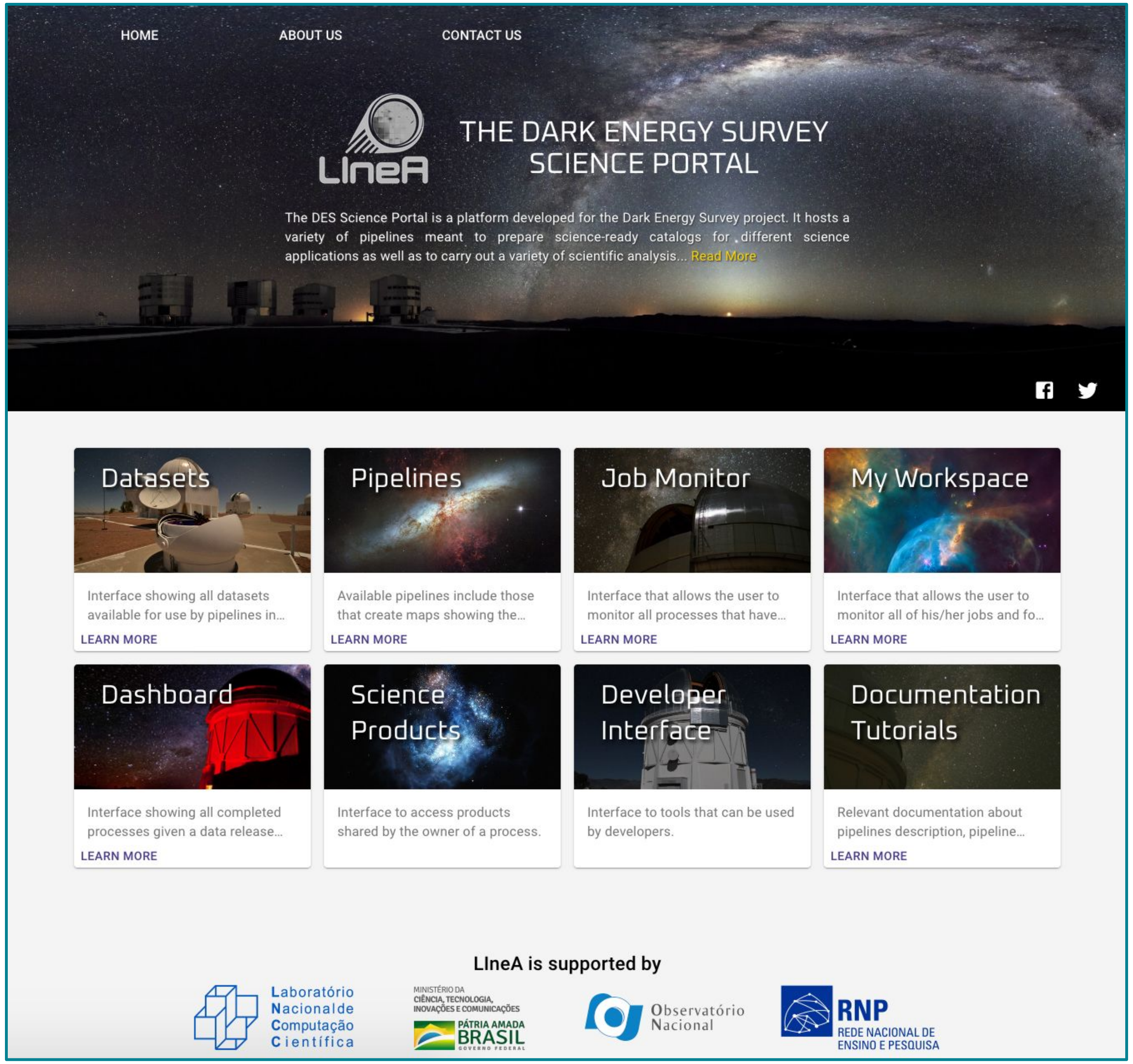


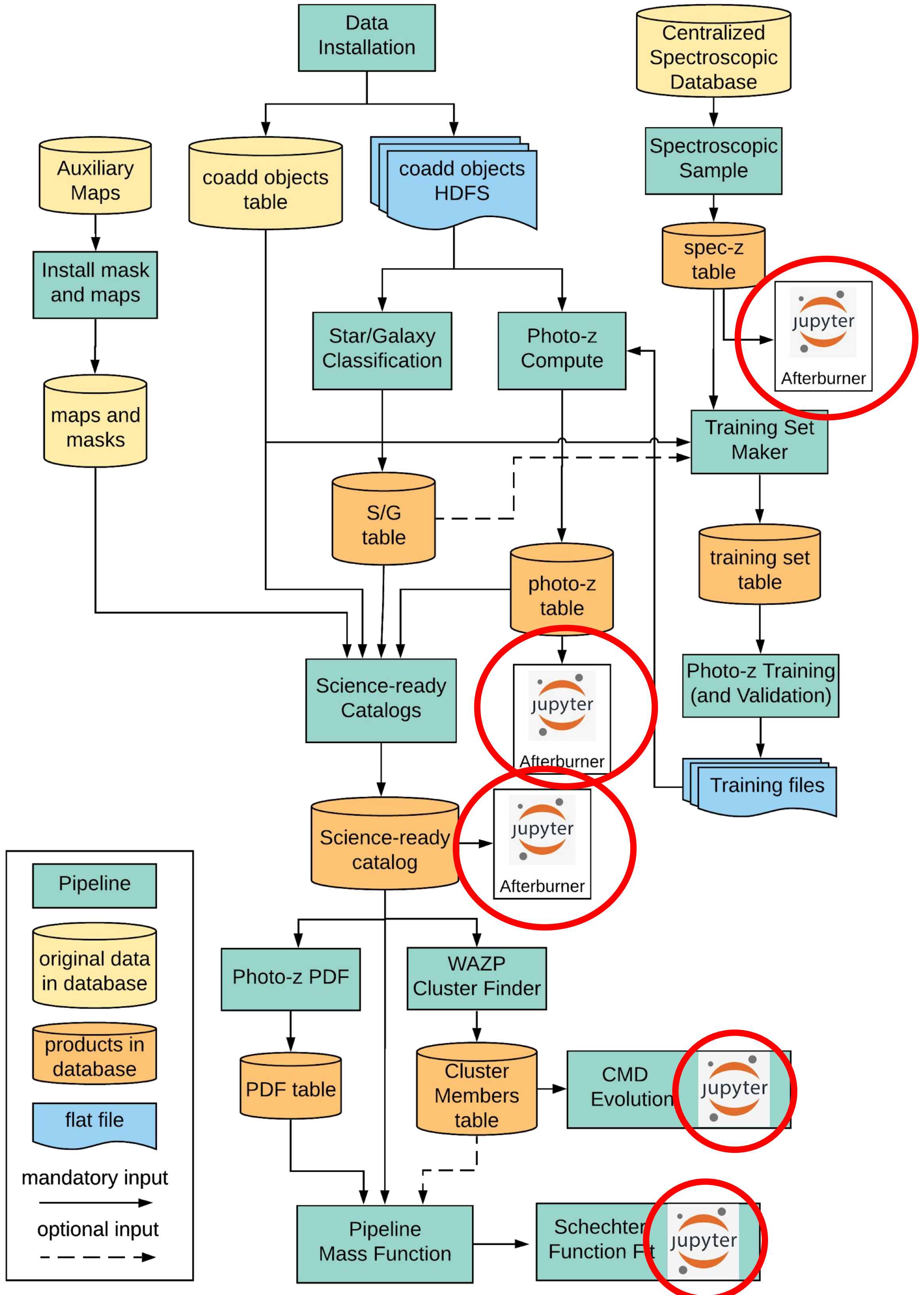
- Evolução histórica
  - iPython (Python interativo)
  - iPython Notebook
  - Jupyter Notebook (+100 linguagens)
  - Jupyter Lab (nova interface)
  - Jupyter Hub (plataforma multiusuário)

- Instalação
  - Instalação local padrão via Anaconda (apresentação Carlos)

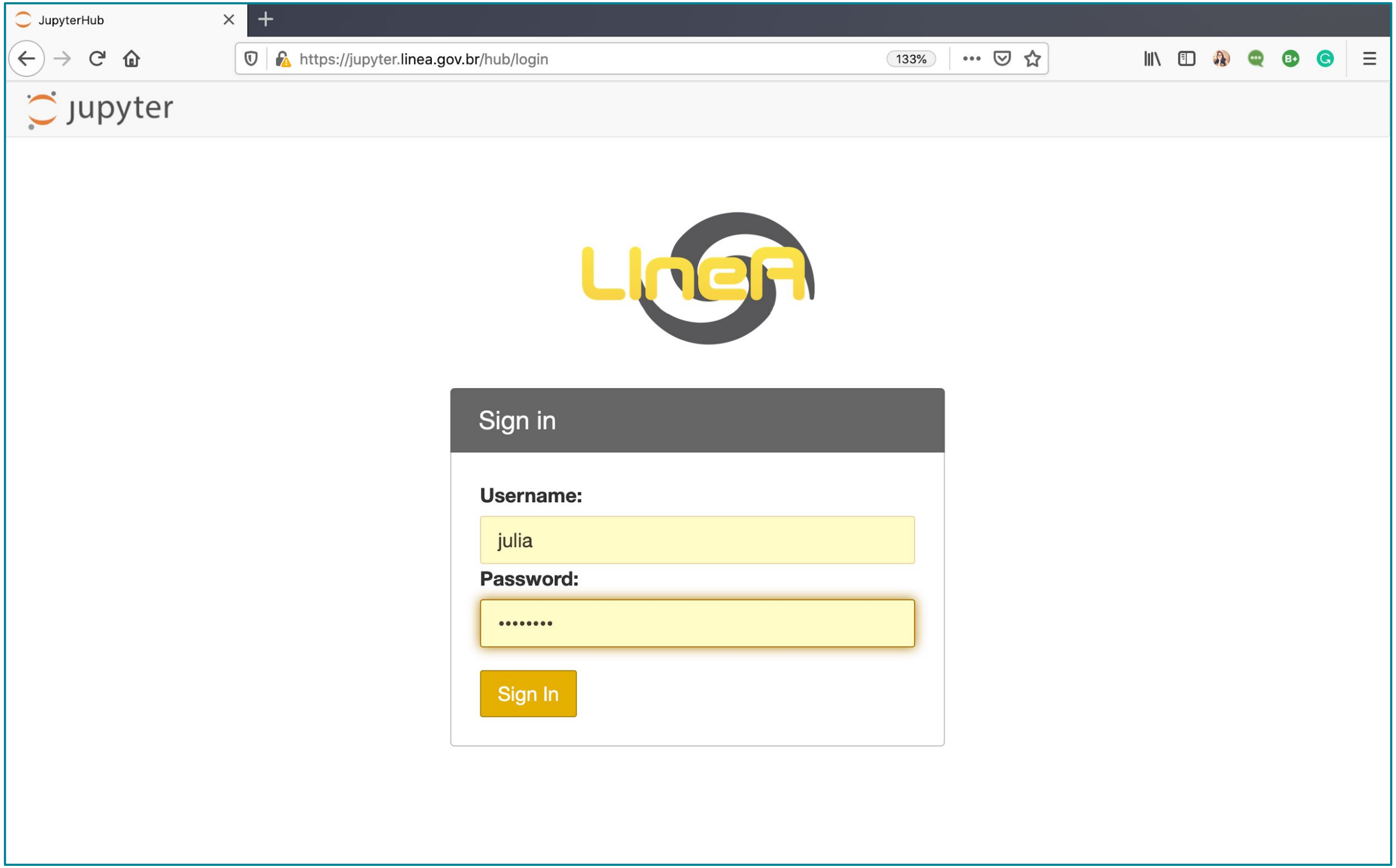
- Vantagens
  - Análise exploratória, caracterização dos dados
  - Desenvolvimento protótipos de aplicações
  - Visualização de resultados
  - Gráficos interativos
  - Integração com Github, Gitlab
  - Acesso remoto (interessante p/ EAD)
  - Processamento distribuído (servidor local ou nuvem)

Introdução - Jupyter Hub @LIneA

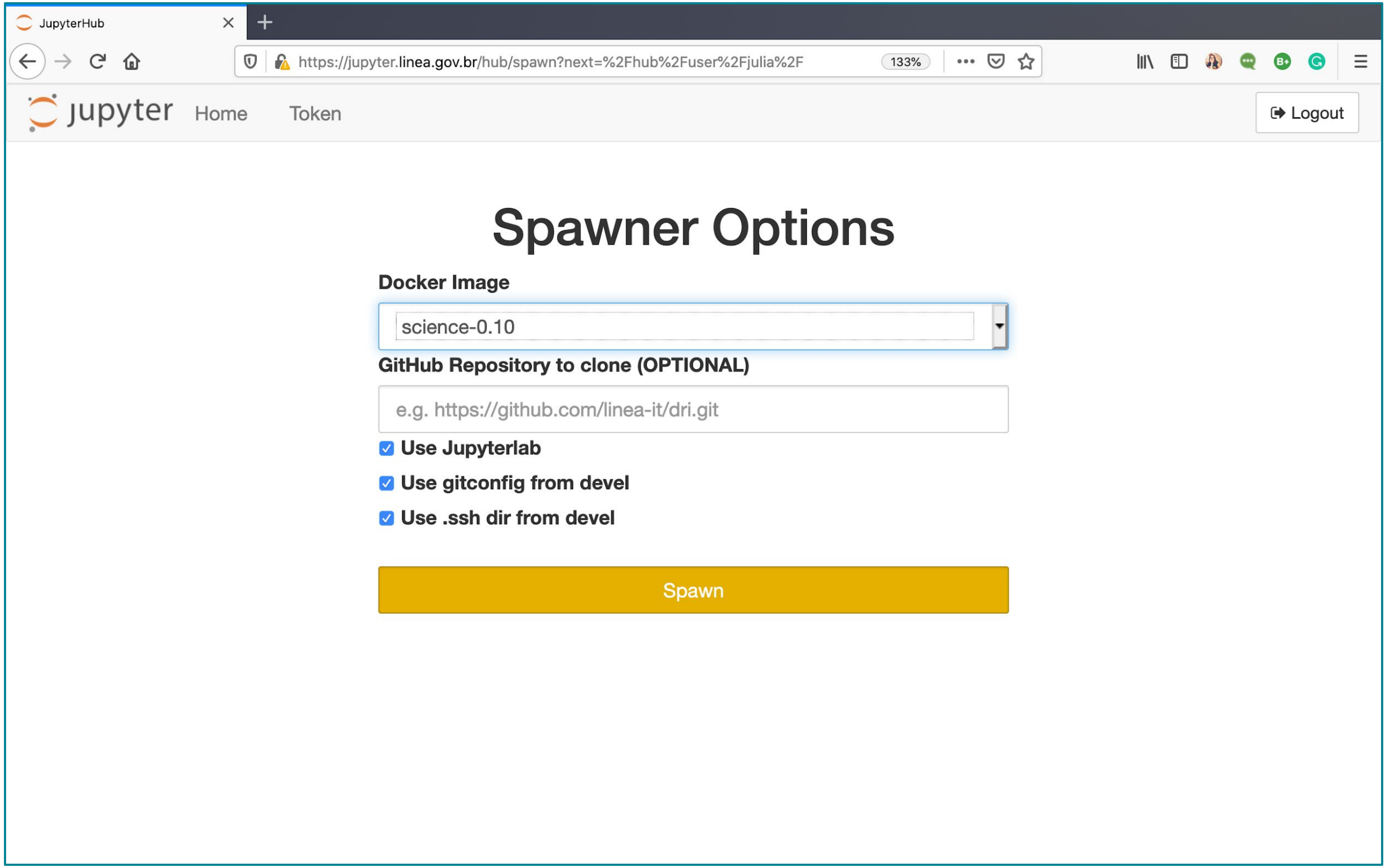




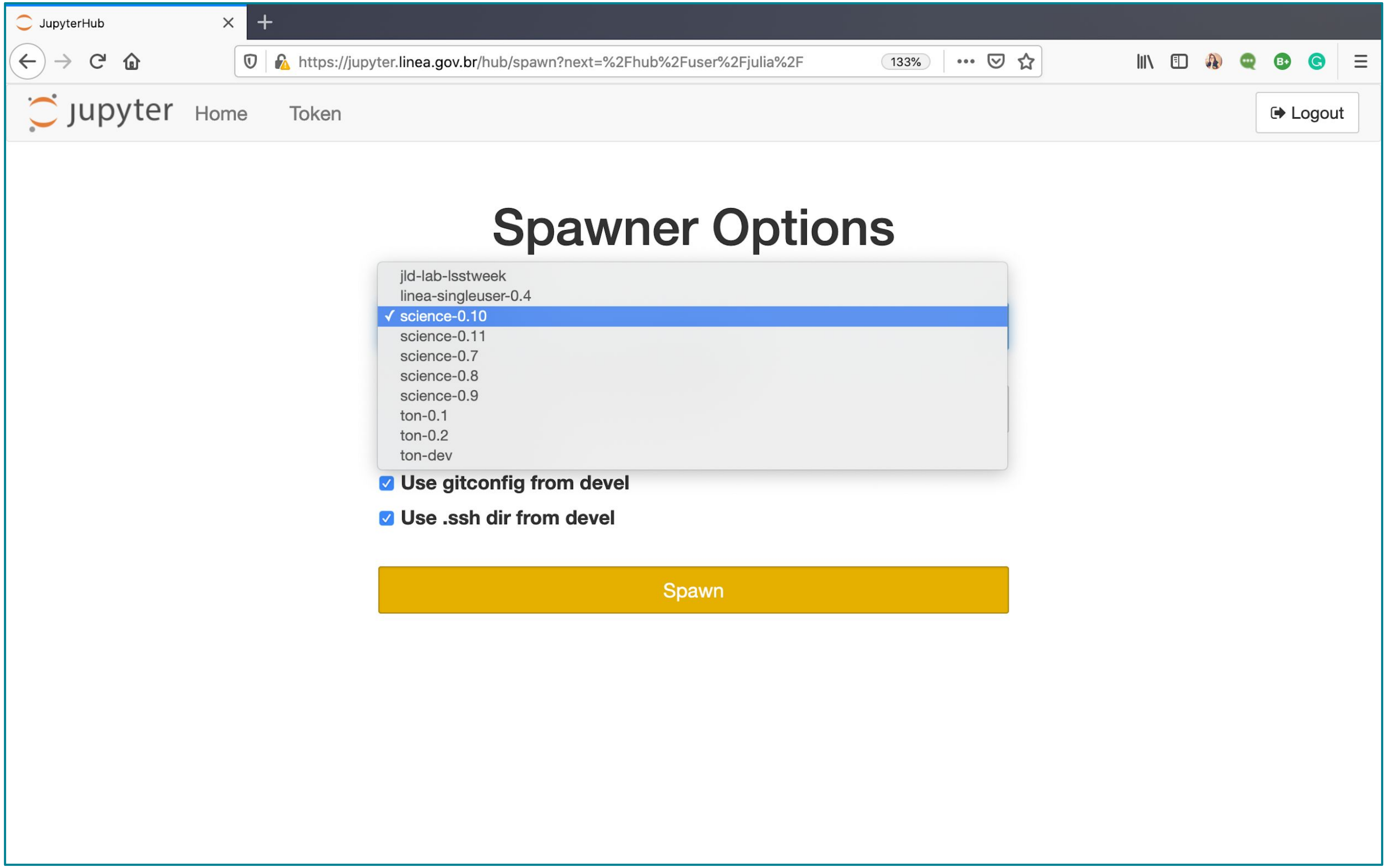
Introdução - Jupyter Hub @LIneA



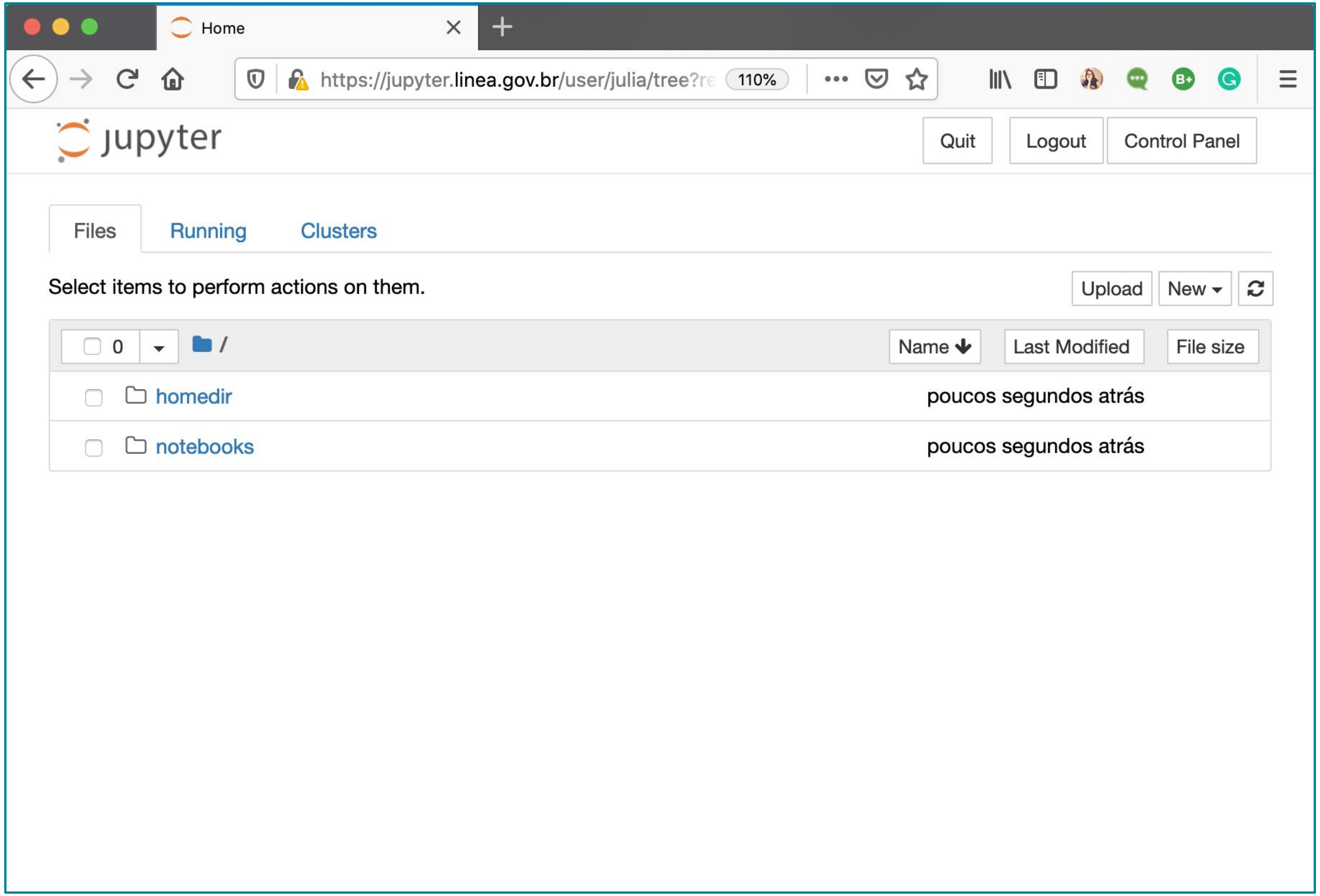
Introdução - Jupyter Hub @LIneA



Introdução - Jupyter Hub @LIneA



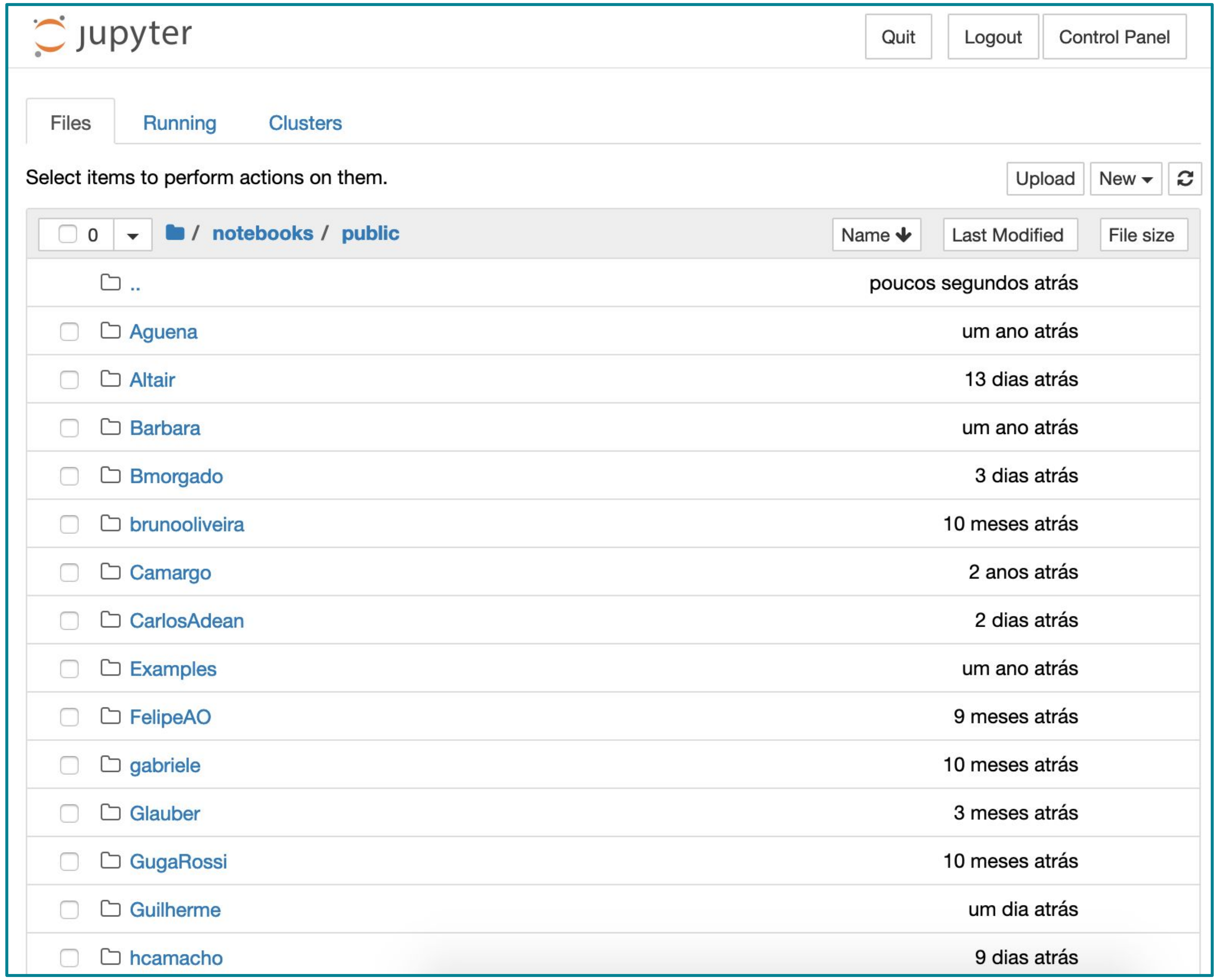
Introdução - Jupyter Hub @LIneA



Introdução - Jupyter Hub @LIneA

Serviços oferecidos

- Plataforma para desenvolvimento e compartilhamento de códigos
- Acesso a dados
- Instalação de bibliotecas sob demanda (imagens docker)
- Integração com Github e Gitlab

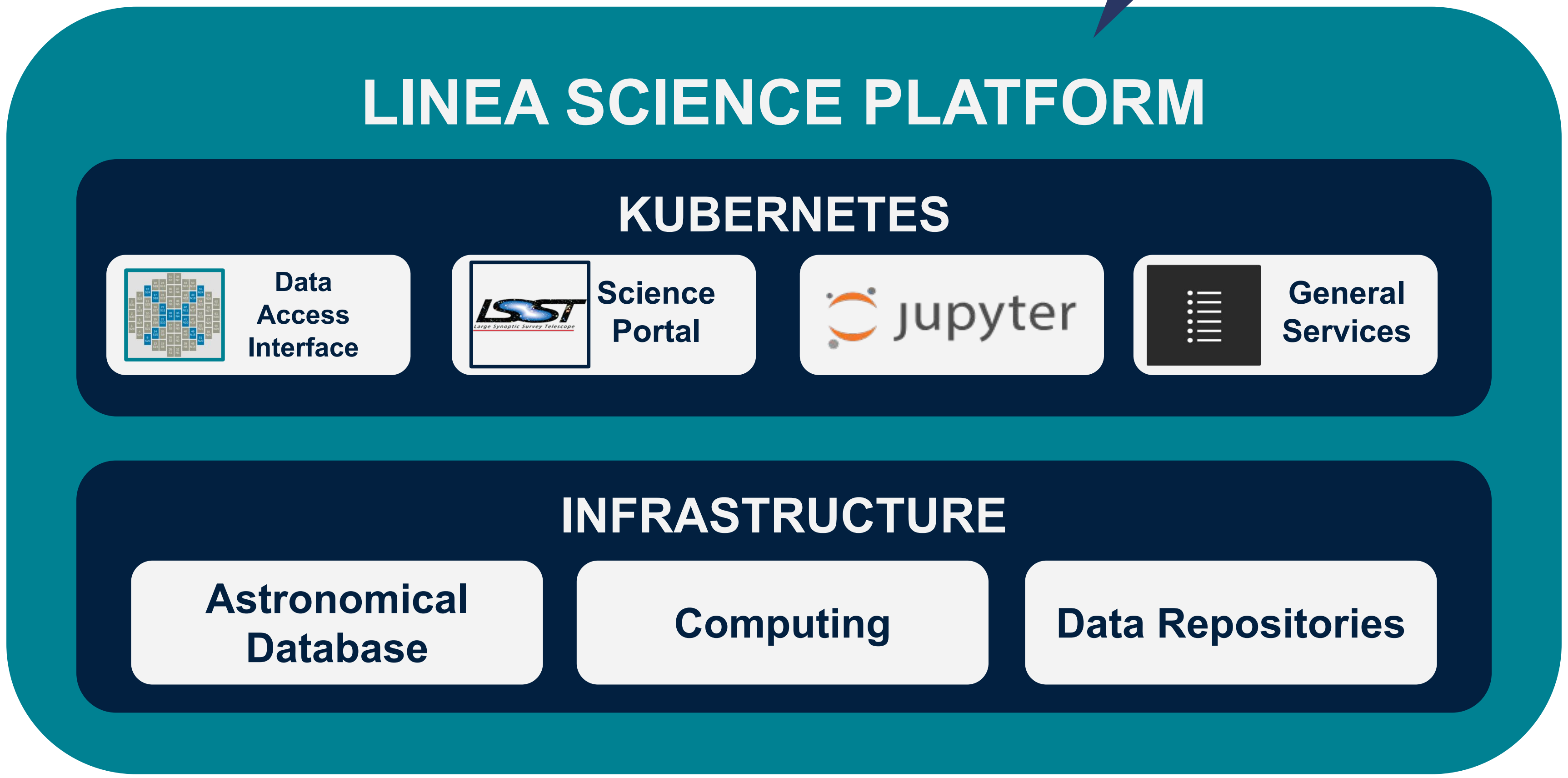


Planos para o futuro

- Melhorias
  - Aumento da capacidade computacional e processamento em paralelo com Kubernetes (colab. RNP)
  - Autenticação e autorização com base na CAFe (colab. Gidlab-RNP)

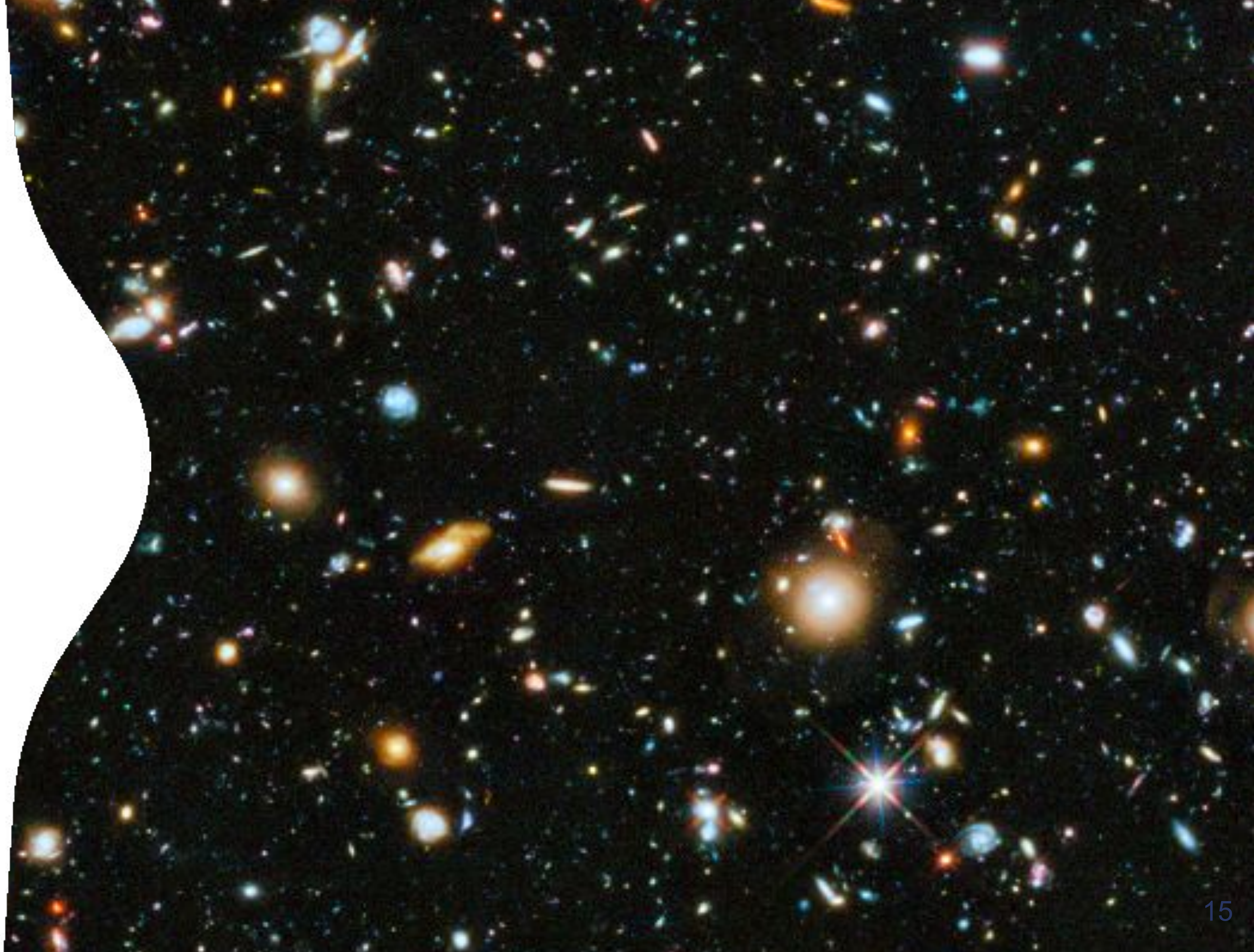
- Novos serviços
  - Abertura para usuários externos com acesso a dados públicos
  - Oferecimento de cursos extracurriculares para E.M., I.C. (colab. OBA)

Detalhes na apresentação do Carlos



Parte 2
Live Demo

- ➢ Primeiros passos

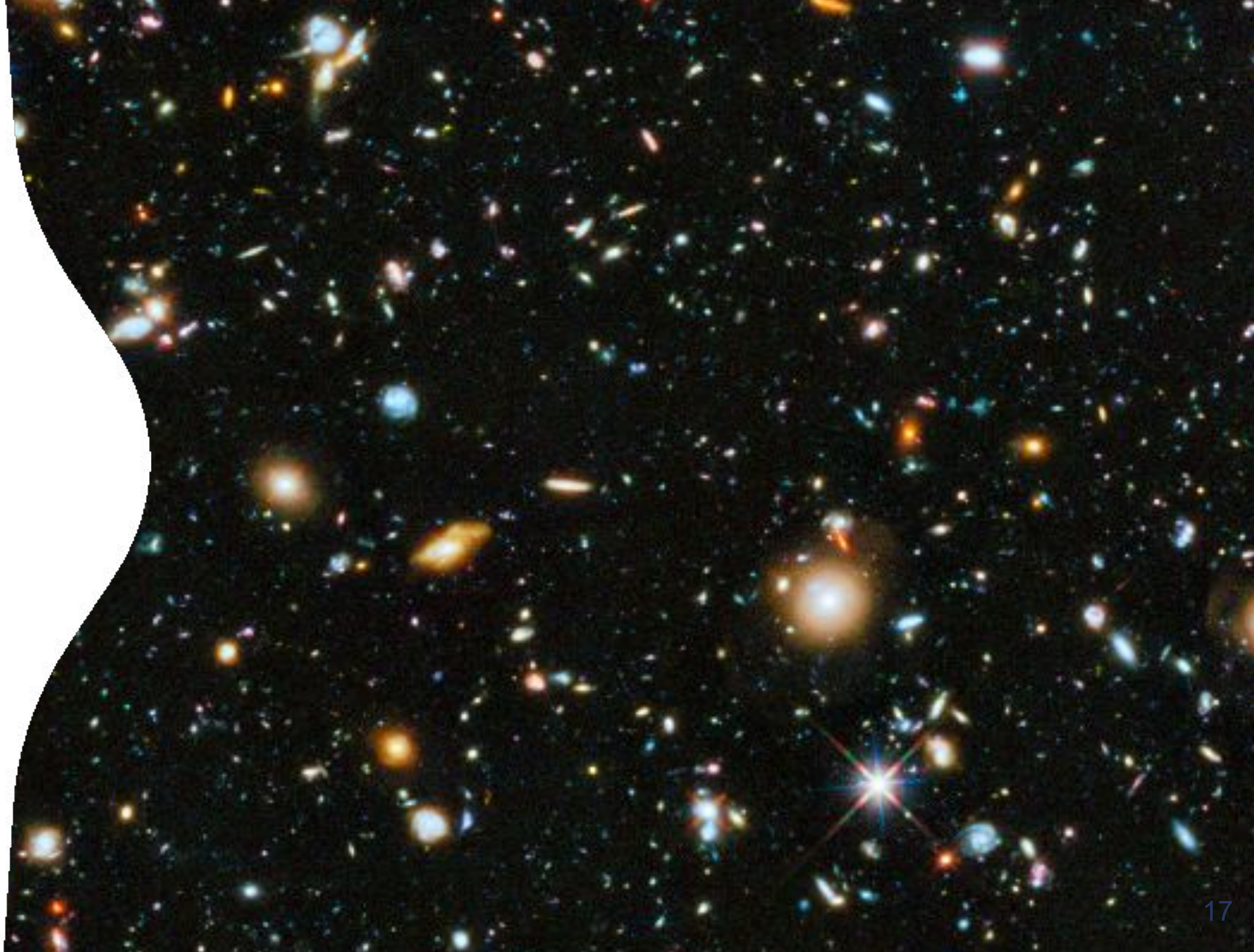


Parte 2 - Live demo: primeiros passos

- Inicializando o Jupyter Notebook (local, via terminal)
- Jupyter Notebook vs. Jupyter Lab
- Acessando o Jupyter Hub do LIneA
- Clonando um repositório Git: https://github.com/juliagschwend/jupyter-tutorial
- Navegando pela interface
- Notebook 1 - noções básicas

Parte 3
Live Demo

- Aplicações



Parte 3 - Live demo: exemplos de aplicação

- Notebook 2 - slideshow com a extensão RISE
    - Operações básicas com arrays
- Notebook 3 - análise exploratória
    - Importação de módulos e bibliotecas
    - Leitura e escrita de dados
    - Visualização de tabelas e estatísticas básicas
    - Exemplos de gráficos estáticos e interativos
    - Exportação de um notebook para outros formatos (HTML, PDF, etc)

Considerações finais

- Os notebooks se consolidaram como um novo jeito novo de trabalhar, tanto na academia quanto no mercado de tecnologia.

- O LIneA está comprometido com a difusão desse conhecimento através de tutoriais, cursos, etc e com o oferecimento desse serviço para a comunidade no novo site (2º semestre).

- Para os interessados em aprofundar os conceitos vistos aqui, ficaremos felizes em fazer um *follow-up* deste tutorial numa próxima oportunidade.

Links

- http://www.thehistoryblog.com
- https://en.wikipedia.org/wiki/Lab_notebook
- https://www.gw-openscience.org/start/
- https://blog.jupyter.org/congratulations-to-the-ligo-and-virgo-collaborations-from-project-jupyter-5923247be019
- https://ligo.org/news/index.php#nobel
- https://www.gw-openscience.org/GW150914data/GW150914_tutorial.html
- https://www.nature.com/articles/s41586-019-1129-z
- https://www.slideshare.net/eurosigdocacm/jupyter-127079129
- https://www.youtube.com/watch?v=zxkdO07L29Q
- https://www.youtube.com/watch?v=NSiPeoDpwuI
- https://jupyterlab.readthedocs.io/en/latest/getting_started/overview.html
- https://nbviewer.jupyter.org/
- https://jupyter-notebook.readthedocs.io/en/stable/index.html
- https://www.museogalileo.it/en/
- https://www.youtube.com/watch?v=zxkdO07L29Q
- https://www.youtube.com/watch?v=nLw1RNvfElg

Obrigada!

contato:
julia@linea.gov.br