

HPRC Data Workshop

Texas A&M Conference on Energy

Zhenhua He

12/03/2021



Data Labs

Lab I. JupyterLab

We will load a module with required libraries and run JupyterLab on the HPRC Terra Portal.

Lab II. Data Exploration

We will go through some examples with two popular Python libraries: Pandas and Matplotlib for data exploration.

Lab I. JupyterLab



Screenshot of JupyterLab interface showing a notebook titled "Lorenz.ipynb".

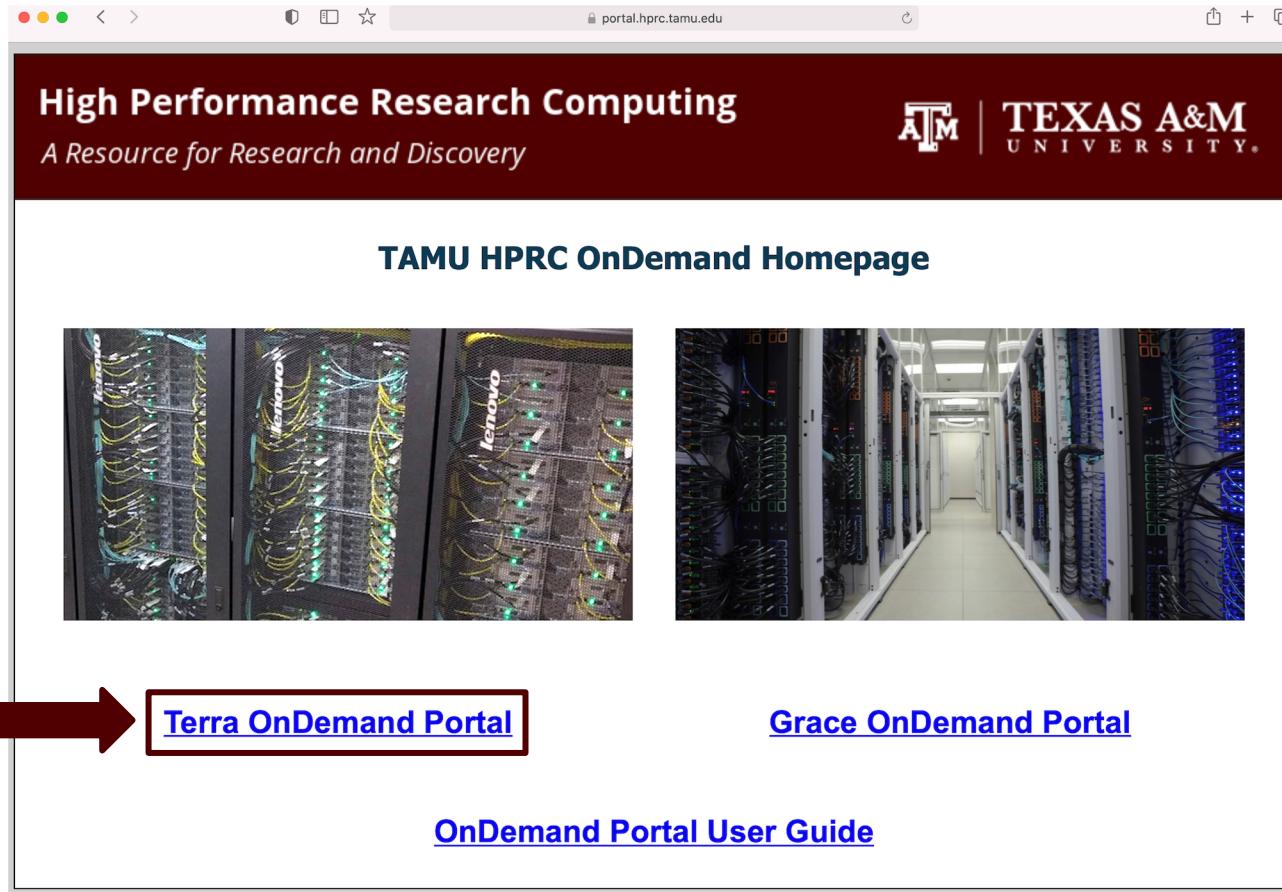
The interface includes:

- File Bar:** File, Edit, View, Run, Kernel, Tabs, Settings, Help.
- File Explorer:** Shows files like "notebooks", "Data.ipynb", "Fasta.ipynb", "Julia.ipynb", "Lorenz.ipynb", "R.ipynb", "iris.csv", "lightning.json", and "lorenz.py".
- Code Editor:** Displays the content of "Lorenz.ipynb". It shows the Lorenz system of differential equations:
$$\begin{aligned}\dot{x} &= \sigma(y - x) \\ \dot{y} &= \rho x - y - xz \\ \dot{z} &= -\beta z + xy\end{aligned}$$
A note says: "Let's call the function once to view the solutions. For this set of parameters, we see the trajectories swirling around two points, called attractors."
- Cell Tools:** Includes a code cell with the command `from lorenz import solve_lorenz` and a parameter cell with sliders for `sigma`, `beta`, and `rho`.
- Output View:** Shows a 3D plot of the Lorenz attractor.
- Code Editor:** Shows the "lorenz.py" file content, which contains the implementation of the Lorenz system and its derivative.

L1 - Resources

- [Texas A&M High Performance Research Computing \(HPRC\)](#)
- [Terra Quick Start Guide](#)
- [HPRC Portal](#)
- [HPRC YouTube Channel](#)
- [Jupyter Project](#)
- help@hprc.tamu.edu

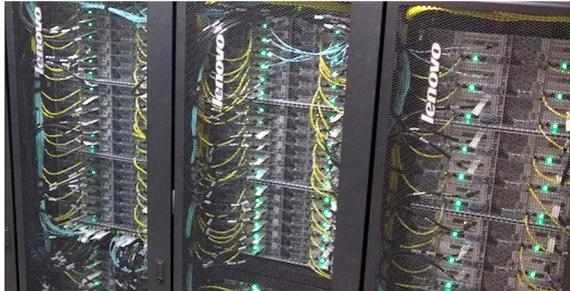
Login HPRC Portal



A screenshot of a web browser displaying the TAMU HPRC OnDemand Homepage. The page has a dark header with the text "High Performance Research Computing" and "A Resource for Research and Discovery". To the right of the header is the Texas A&M University logo. Below the header, the text "TAMU HPRC OnDemand Homepage" is centered. Two photographs of server racks are displayed: one showing three server units with yellow cables, and another showing a long corridor between server racks. At the bottom of the page, there are three blue links: "Terra OnDemand Portal" (with a red arrow pointing to it), "Grace OnDemand Portal", and "OnDemand Portal User Guide".

High Performance Research Computing
A Resource for Research and Discovery

TAMU HPRC OnDemand Homepage

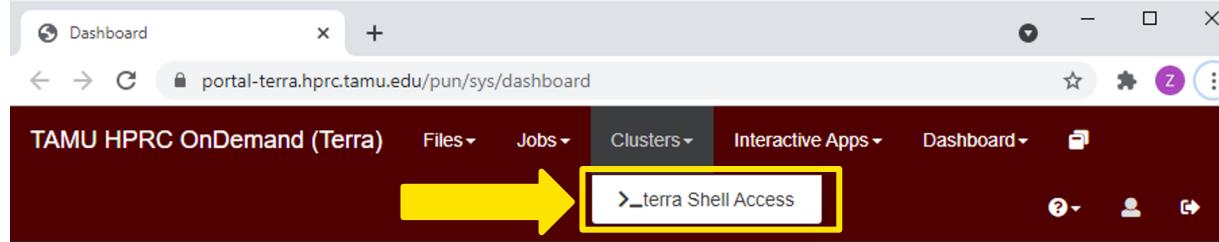


[Terra OnDemand Portal](#)

[Grace OnDemand Portal](#)

[OnDemand Portal User Guide](#)

Shell Access - I



A screenshot of a web browser window titled "Dashboard" with the URL "portal-terra.hprc.tamu.edu/pun/sys/dashboard". The browser has a standard top bar with back, forward, and search buttons. Below the bar is a dark red header bar containing the text "TAMU HPRC OnDemand (Terra)" and several dropdown menu items: "Files", "Jobs", "Clusters", "Interactive Apps", "Dashboard", and a folder icon. A yellow arrow points from the left towards the "Clusters" menu item. To the right of the arrow, a yellow box highlights the "Clusters" menu item, which is currently active. Below the header is a photograph of server racks in a data center, with the letters "D", "C", and "B" visible above them. At the bottom of the page, there is a section titled "Message of the Day" and another titled "IMPORTANT POLICY INFORMATION" containing a bulleted list of rules.

OnDemand provides an integrated, single access point for all of your HPC resources.

Message of the Day

IMPORTANT POLICY INFORMATION

- Unauthorized use of HPRC resources is prohibited and subject to criminal prosecution.
- Use of HPRC resources in violation of United States export control laws and regulations is prohibited. Current HPRC staff members are US citizens and legal residents.
- Sharing HPRC account and password information is in violation of State Law. Any shared accounts will be DISABLED.
- Authorized users must also adhere to ALL policies at: <https://hprc.tamu.edu/policies>

Shell Access - II

A screenshot of a terminal window titled "happidence1@login-0502:~". The window shows a terminal session with the following content:

```
*****  
This computer system and the data herein are available only for authorized  
purposes by authorized users: use for any other purpose is prohibited and may  
result in administrative/disciplinary actions or criminal prosecution against  
the user. Usage may be subject to security testing and monitoring to ensure  
compliance with the policies of Texas A&M University, Texas A&M University  
System, and the State of Texas. There is no expectation of privacy on this  
system except as otherwise provided by applicable privacy laws. Users should  
refer to Texas A&M University Standard Administrative Procedure 29.01.03.M0.02,  
Rules for Responsible Computing, for guidance on the appropriate use of Texas  
A&M University information resources.  
*****  
  
Password:  
Duo two-factor login for happidence1  
  
Enter a passcode or select one of the following options:  
  
1. Duo Push to XXX-XXX-9472  
2. Phone call to XXX-XXX-9472  
3. SMS passcodes to XXX-XXX-9472 (next code starts with: 1)  
  
Passcode or option (1-3): 1  
Success. Logging you in...  
Last login: Sun Jan 31 20:33:32 2021 from 165.91.16.42  
=====| Texas A&M University High Performance Research Computing |  
| Website: https://hprc.tamu.edu |  
| Consulting: help@hprc.tamu.edu (preferred) or (979) 845-0219 |  
| Ada Documentation: https://hprc.tamu.edu/wiki/Ada |  
| Terra Documentation: https://hprc.tamu.edu/wiki/Terra |  
| Grace Documentation: https://hprc.tamu.edu/wiki/Grace |
```

Check out Exercises

The screenshot shows a GitHub repository page for 'happidence1/DataLabs'. The repository is public and contains 1 branch and 0 tags. The most recent commit is 'happidence1 Update README.md' by f055687, made 29 minutes ago with 11 commits. The repository has 0 stars and 0 forks. The 'Code' tab is selected. On the right, there's an 'About' section for the 2021 Texas A&M Conference on Energy, a 'Readme' link, and a 'Releases' section indicating no releases have been published.

File	Action	Time
happidence1 Update README.md	f055687 29 minutes ago	11 commits
data	Add files via upload	yesterday
images	Add files via upload	yesterday
01_jupyterlab.ipynb	Add files via upload	yesterday
02_data_exploration_pand...	Add files via upload	yesterday

```
# Change to /scratch/user/netid directory
cd $SCRATCH
# git clone (check out) the Jupyter notebooks for the labs
git clone https://github.com/happidence1/DataLabs.git
```

Go to JupyterLab Page

The screenshot shows the TAMU HPRC OnDemand (Terra) dashboard. At the top, there's a navigation bar with links for Dashboard, Apps, Grace, LintCode, Spring 2021 - Goog..., John Peas, Other bookmarks, and Reading list. Below the navigation bar, the main content area has a dark red header with "TAMU HPRC OnDemand (Terra)" and dropdown menus for Files, Jobs, Clusters, Interactive Apps, and Dashboard. A sidebar on the left displays a photograph of server racks with the caption: "OnDemand provides an integrated, single access to all resources." Below this, there's a "Message of the Day" section and an "IMPORTANT POLICY INFORMATION" section with a bulleted list of rules. The central part of the dashboard shows a list of interactive applications under the "Interactive Apps" menu, which is currently open. The "JupyterLab" option is highlighted with a large red arrow pointing to it. To the right of the JupyterLab entry, there's a "Servers" section with links for Jupyter Notebook, JupyterLab (which is also highlighted with a red box), RStudio, and Spark-Jupyter Notebook.

Dashboard

portal-terra.hprc.tamu.edu/pun/sys/dashboard

TAMU HPRC OnDemand (Terra)

Files ▾ Jobs ▾ Clusters ▾

Interactive Apps ▾

BIO

- Beauti
- DIYABC
- FigTree
- IGV
- JBrowse
- Krait
- Mauve
- Structure
- Tracer

GUI

- ANSYS Workbench
- Abaqus/CAE
- LS-PREPOST
- LS-PREPOST (workshop)
- MATLAB
- Paraview
- VNC

Imaging

- Chimera
- Coot

Message of the Day

IMPORTANT POLICY INFORMATION

- Unauthorized use of HPRC resources is prohibited and subject to disciplinary action.
- Use of HPRC resources in violation of United States export control laws is prohibited. Current HPRC staff members are US citizens and legal residents.
- Sharing HPRC account and password information is in violation of University policy and will be DISABLED.
- Authorized users must also adhere to ALL policies at: <https://hpc.tamu.edu/policy/>

Servers

Jupyter Notebook

JupyterLab

RStudio

Spark-Jupyter Notebook

https://portal-terra.hprc.tamu.edu/pun/sys/dashboard/batch_connect/sys/chimera/session_contexts/new

Load module

The screenshot shows a web browser window for JupyterLab on the TAMU HPRC OnDemand (Terra) platform. The URL is https://portal-terra.hprc.tamu.edu/pun/sys/dashboard/batch_connect/sys/jupyterlab/session_contexts/new. The page title is "JupyterLab - TAMU HPRC OnD". The top navigation bar includes links for Home, My Interactive Sessions, JupyterLab, Files, Jobs, Clusters, Interactive Apps, Dashboard, Help, User Profile, and Log Out.

The main content area is titled "Interactive Apps" and lists various modules:

- BIO
- Beauti
- DIYABC
- FigTree
- IGV
- JBrowse
- Krait
- Mauve
- Structure
- Tracer
- GUI
- ANSYS Workbench
- Abaqus/CAE
- LS-PREPOST

The "JupyterLab" section is highlighted. It contains the following information:

Module: Anaconda3/2020.07 (highlighted with a red box and a large black arrow pointing to it)

Anaconda/3-x.x.x.x and Anaconda3 use Python3

Optional Environment to be activated: (empty input field)

Enter the name of the environment to be activated. (Optional)

Leave blank to use the **default** environment for the selected Module.

Your optional conda environment must have been previously built with one of the Anaconda or Python modules listed in the Module option above. See [instructions](#).

Number of hours: 2 (highlighted with a red box and a large black arrow pointing to it)

Number of cores: (input field)

Number of cores: 2
Total memory (GB): 4

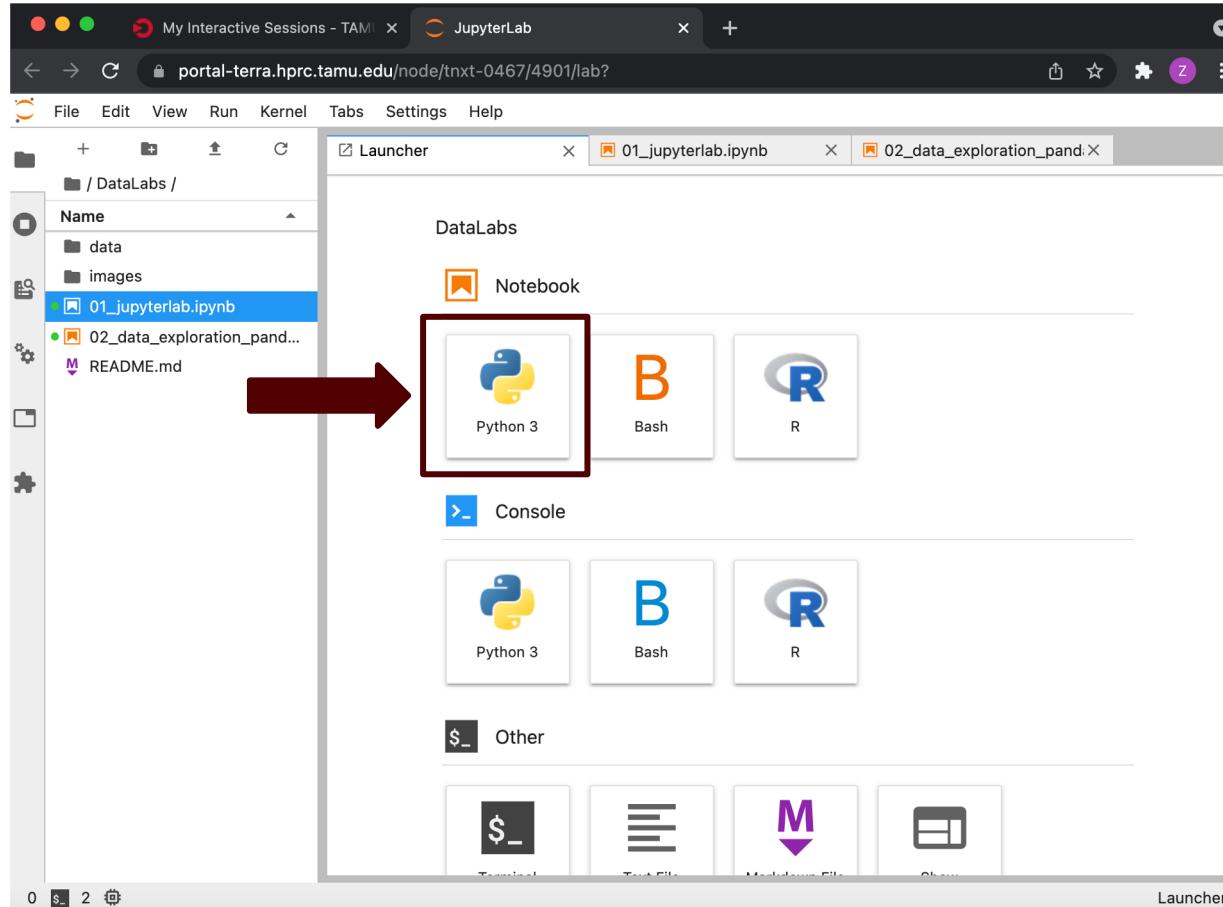
Connect to JupyterLab

The screenshot shows a web browser window for the TAMU HPRC OnDemand (Terra) system. The URL is `portal-terra.hprc.tamu.edu/pun/sys/dashboard/batch_connect/sessions`. The page displays a success message: "Session was successfully deleted." Below this, the "My Interactive Sessions" section is shown. A specific JupyterLab session is listed with the following details:

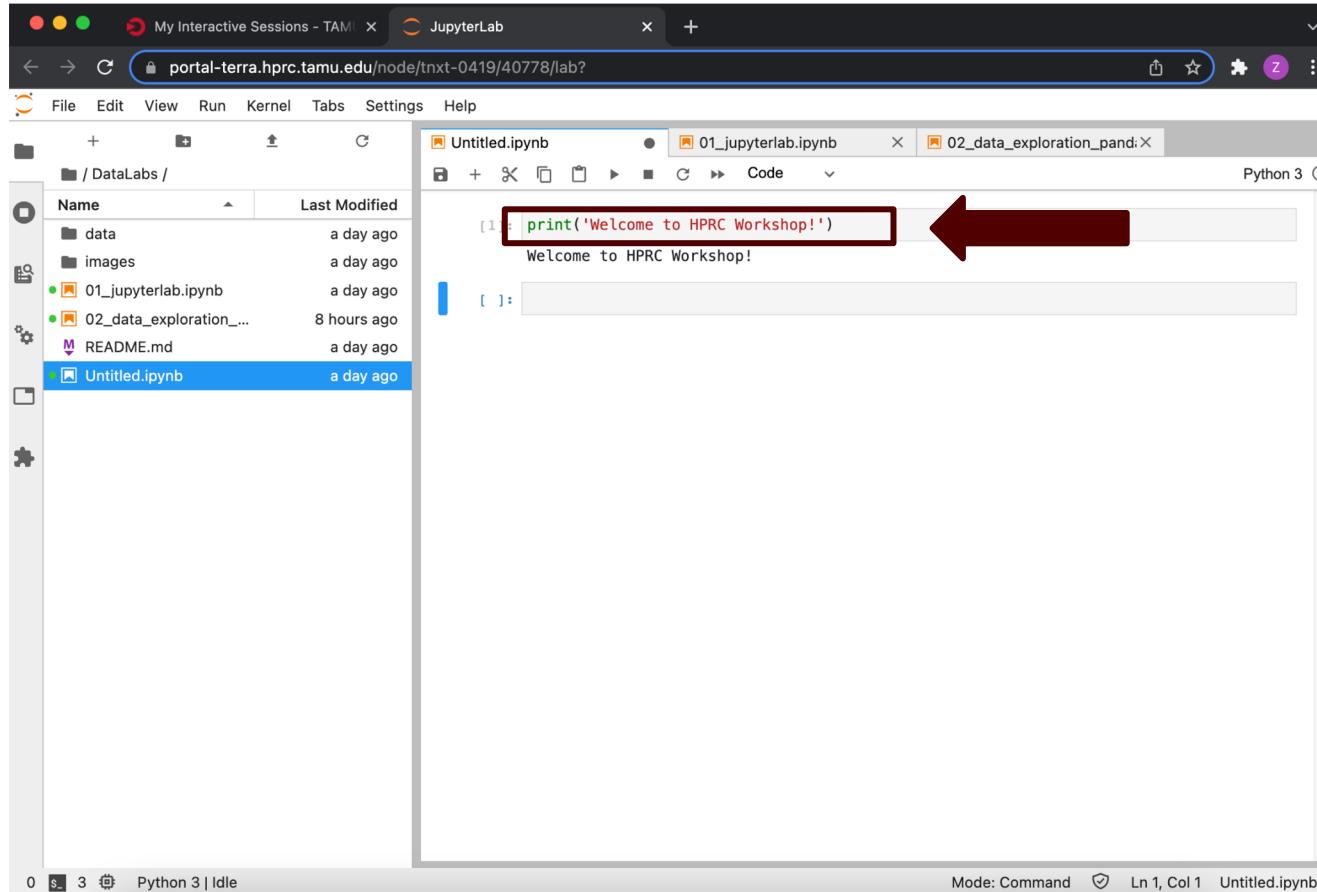
- JupyterLab (9638675)**
- Host:** >_tnxt-0467
- Created at:** 2021-12-01 14:46:42 CST
- Time Remaining:** 1 hour and 57 minutes
- Session ID:** 4150abe8-a7a8-46a1-b228-83f39a373c59

A large red arrow points to the blue "Connect to JupyterLab" button.

Create a Jupyter Notebook



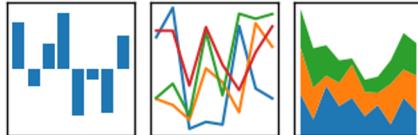
Test JupyterLab



Lab II. Data Exploration

pandas

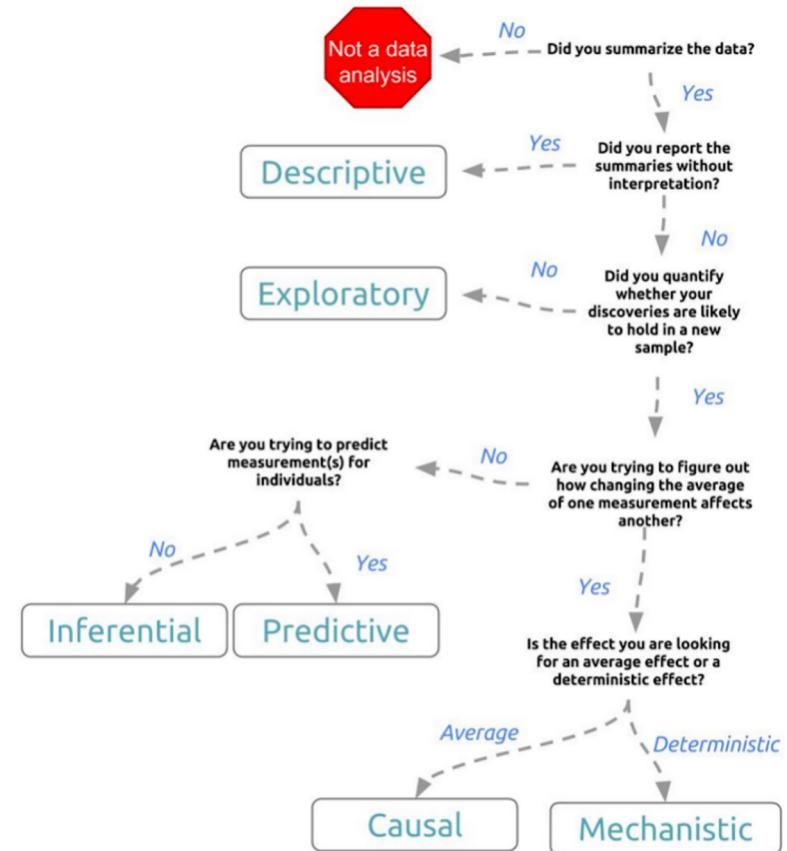
$$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$$



matplotlib

Types of Data Science Problems

- **Descriptive** (summaries, e.g., census)
- **Exploratory** (search for unknowns, e.g., four-planet solar system)
- **Inferential** (find correlations, e.g., many social studies)
- **Predictive** (make predictions, e.g., Face ID, Echo, Siri)
- **Causal** (explore causation, e.g., smoking versus lung cancer)
- **Mechanistic** (determine governing principles, e.g., experimental science)



Credit: Jeff Leek - The Elements of Data Analytic Style

Data Structures

Pandas has two data structures that are descriptive and optimized for data with different dimensions.

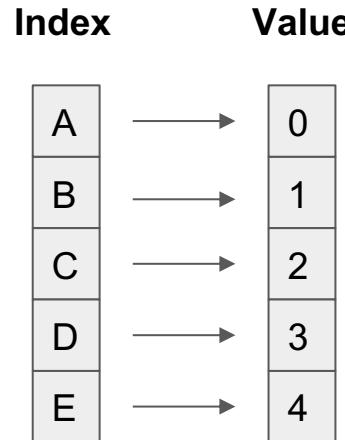
- **Series:** 1D labeled array
- **DataFrame:** General 2D labeled, size-mutable tabular structure with potentially heterogeneously-typed columns

Series in pandas

- One-dimensional labeled array
- Capable of holding any data type (integers, strings, floating point numbers, etc.)
- Example: time-series oil price data



(source: oilprice.com)

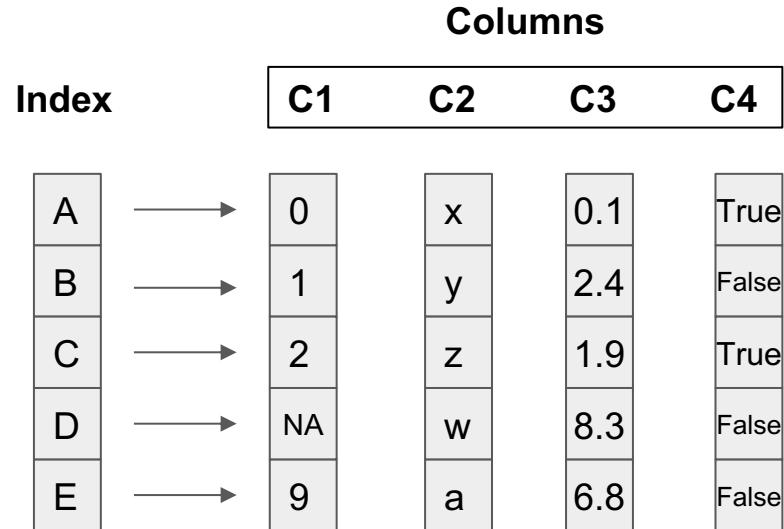


DataFrame in pandas

- Primary Pandas data structure
- A dict-like container for Series objects
- Two-dimensional size-mutable
- Heterogeneous tabular data structure

Top producing oil fields in the United States, 2019* being updated				
Rank	Field	State	Discovery Year	Million Bbl/Day
1	Permian	Texas/New Mexico	1920	4.2
2	Eagle Ford Shale	Texas	2008	1.34
3	Bakken	North Dakota/Montana	1951	1.33
4	Prudhoe Bay Oil Field	Alaska	1967	.791
5	Wattenberg Gas Field	Colorado	1970	.473
6	Shenzi	Federal Gulf of Mexico	2002	.353
7	Kuparuk River oil field	Alaska	1969	.295
8	Midway-Sunset Oil Field	California	1901	.288
9	Atlantis Oil Field	Federal Gulf of Mexico	1998	.273
10	Sugarkane	Texas	2009	.258

(source: [wikipedia](#))



Pandas Learning Objectives

After this lesson, you will know how to:

- Create a DataFrame
- Drop Entries
- Index, Select, and Filter data
- Sort data
- Input and Output



JupyterLab Exercises

Pandas Cheat Sheet

Data Wrangling with pandas Cheat Sheet

<http://pandas.pydata.org>

Syntax – Creating DataFrames

```
df = pd.DataFrame({  
    "a": [4, 5, 6],  
    "b": [7, 8, 9],  
    "c": [10, 11, 12]],  
    index=[1, 2, 3])  
Specify values for each column.  
  
df = pd.DataFrame(  
    [[4, 7, 10],  
     [5, 8, 11],  
     [6, 9, 12]],  
    index=[1, 2, 3],  
    columns=['a', 'b', 'c'])  
Specify values for each row.
```

```
df = pd.DataFrame(  
    {"a": [4, 5, 6],  
     "b": [7, 8, 9],  
     "c": [10, 11, 12]},  
    index=pd.MultiIndex.from_tuples(  
        [('d',1),('d',2),('e',2)],  
        names=['n','v']))  
Create DataFrame with a MultiIndex.
```

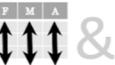
Method Chaining

Most pandas methods return a DataFrame so that another pandas method can be applied to the result. This improves readability of code.

```
df = (pd.melt(df)  
    .rename(columns={  
        'variable': 'var',  
        'value': 'val'})  
    .query('val > 200'))
```

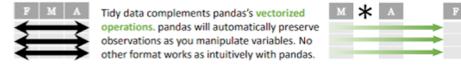
Tidy Data – A foundation for wrangling in pandas

In a tidy data set:



&

Tidy data complements pandas's **vectorized operations**. pandas will automatically preserve observations as you manipulate variables. No other format works as intuitively with pandas.



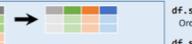
M * A

Reshaping Data – Change the layout of a data set

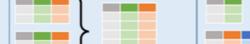
pd.melt(df) Gather columns into rows.



df.pivot(columns='var', values='val') Spread rows into columns.



pd.concat([df1, df2]) Append rows of DataFrames



pd.concat([df1, df2], axis=1) Append columns of DataFrames



Subset Observations (Rows)

df[df.Length > 7] Extract rows that meet logical criteria.

df.drop_duplicates() Remove duplicate rows (only considers columns).

df.head(n) Select first n rows.

df.tail(n) Select last n rows.

Subset Variables (Columns)

df[['width', 'length', 'species']] Select multiple columns with specific names.

df['width'] or df.width Select single column with specific name.

df.filter(regex='regex') Select columns whose name matches regular expression regex.

Logic in Python (and pandas)

Less than	!=	Not equal to
<	=	Not equal to
>	df.column.isin(values)	Group membership
==	pd.isnull(obj)	Is NaN
<=	pd.notnull(obj)	Is not NaN
>=	df.any(), df.all()	Logical and, or, not, any, all
Select all columns between x2 and x4 (inclusive).		
Select columns in positions 1, 2 and 5 (first column is 0).		
Select rows meeting logical condition, and only the specific columns.		

df.loc[:, 'x2': 'x4']
df.loc[:, [1, 2, 5]]
df.loc[df['a'] > 10, ['a', 'c']]

Summarize Data

```
df['w'].value_counts()  
Count number of rows with each unique value of variable  
len(df)
```

Handling Missing Data

```
df.dropna()  
Drop rows with any column having NA/null data.  
df.fillna(value)
```

Combine Data Sets



adf bdf cdf

x1 x2 x1 x3 A T

https://pandas.pydata.org/Pandas_Cheat_Sheet.pdf

Key Plotting Concepts in Matplotlib

• Matplotlib: Figure

Figure is the object that keeps the whole image output.

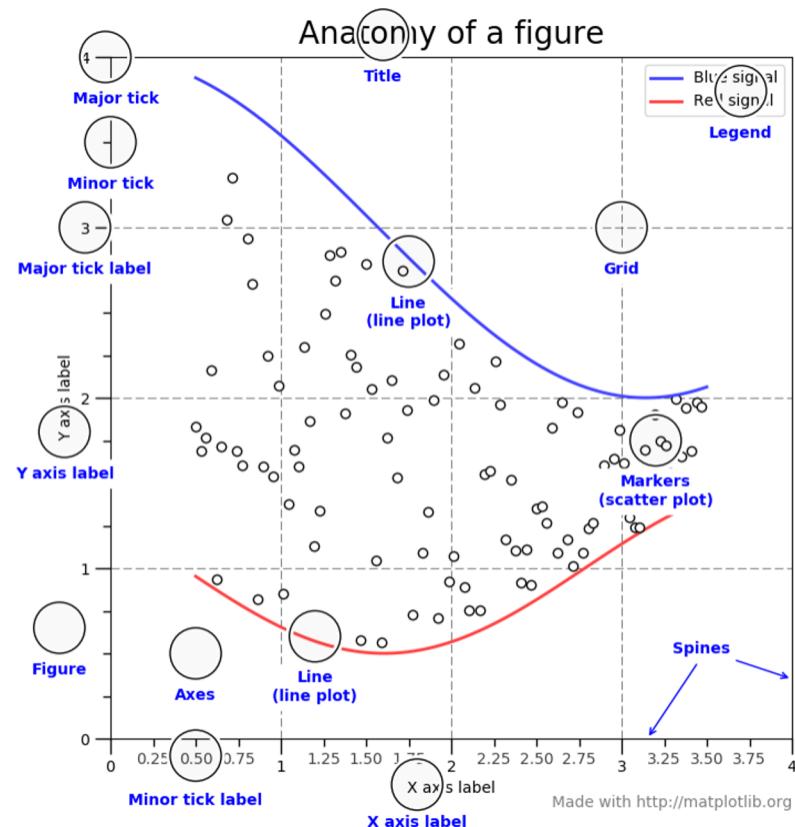
Adjustable parameters include:

1. Image size (`set_size_inches()`)
2. Whether to use `tight_layout` (`set_tight_layout()`)

• Matplotlib: Axes

Axes object represents the pair of axis that contain a single plot (x-axis and y-axis). The Axes object also has more adjustable parameters:

1. The plot frame (`set_frame_on()` or `set_frame_off()`)
2. X-axis and Y-axis limits (`set_xlim()` and `set_ylim()`)
3. X-axis and Y-axis Labels (`set_xlabel()` and `set_ylabel()`)
4. The plot title (`set_title()`)



(Credit: matplotlib.org)

Matplotlib Learning Objectives

After this lesson, you will know how to:

- Scatter plot and Line plot
- Subplots
- Color map
- Contour figures
- 3D figures
 - Surface plots
 - Wire-frame plot
 - Contour plots with projections



JupyterLab Exercises

Matplotlib Cheat Sheet

Python For Data Science Cheat Sheet

Matplotlib

Learn Python Interactively at www.DataCamp.com



Matplotlib
Matplotlib is a Python 2D plotting library which produces publication-quality figures in a variety of hardcopy formats and interactive environments across platforms.

1 Prepare The Data Also see Lists & NumPy

1D Data

```
>>> import numpy as np  
>>> x = np.linspace(0, 10)  
>>> y = np.cos(x)  
>>> z = np.sin(x)
```

2D Data or Images

```
>>> data = 2 * np.random.random((10, 10))  
>>> data[3] = 3 * np.random.random((10, 10))  
>>> data[3][3] = 1000  
>>> U = -1 + x**2 + y**2  
>>> V = 1 + x**2 - y**2  
>>> from matplotlib import pyplot as plt  
>>> img = np.loadtxt('sample_data/asm_grid/bivariate_normal.npy')
```

2 Create Plot

Figure

```
>>> fig = plt.figure()  
>>> fig2 = plt.figure(figsize=plt.rcParams['figure.figsize'])
```

Axes

All plotting is done with respect to an Axes. In most cases, a subplot will fit your needs. A subplot is an axes on a grid system.

```
>>> fig.add_axes()  
>>> ax1 = fig.add_subplot(221) # row-col-num  
>>> ax2 = fig.add_subplot(222)  
>>> fig3, axes = plt.subplots(nrows=2, ncols=2)  
>>> fig4, axes2 = plt.subplots(ncols=3)
```

3 Plotting Routines

1D Data

```
>>> lines = ax.plot(x,y)  
>>> ax.scatter(x,y)  
>>> ax.step([1,2,3],[3,4,5])  
>>> ax.pcolor([[0,1,2,3],[4,5,6,7]])  
>>> ax.pcolor([0,1,2,3],[4,5,6,7],cmap='hot')  
>>> ax.pcolor([0,1,2,3],[4,5,6,7],cmap='hot', interpolation='nearest')  
>>> ax.pcolor([0,1,2,3],[4,5,6,7],cmap='hot', interpolation='nearest', vmin=-2, vmax=21)  
>>> ax.fill_between(x,y,color='yellow')
```

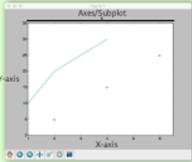
2D Data or Images

```
>>> fig, ax = plt.subplots()  
>>> im = ax.imshow(lsqg,  
>>> cmap='gist_earth',  
>>> interpolation='nearest',  
>>> vmin=-2,  
>>> vmax=21)
```

Colormapped or RGB arrays

Plot Anatomy & Workflow

Plot Anatomy



Figure

Workflow

The basic steps to creating plots with matplotlib are:

- 1 Prepare data
- 2 Create plot
- 3 Plot
- 4 Customize plot
- 5 Save plot
- 6 Show plot

```
>>> import matplotlib.pyplot as plt  
>>> x = [1,2,3,4]  
>>> y = [1,4,9,16]  
>>> fig = plt.figure() Step 1  
>>> ax = fig.add_subplot(111) Step 2  
>>> ax.plot(x, y, color='lightblue', linewidth=3) Step 3  
>>> ax.scatter([4,6,8],  
>>> [5,10,15],  
>>> color='darkgreen',  
>>> marker='*')  
>>> ax.set_xlim(1, 6.5)  
>>> plt.savefig('foo.png') Step 4  
>>> plt.show() Step 6
```

4 Customize Plot

Colors, Color Bars & Color Maps

```
>>> plt.plot(x, x, x**2, x**3)  
>>> ax.plot(x, y, alpha=0.4)  
>>> ax.set_color_cycle(['red','blue'])  
>>> fig.colorbar(im, orientation='horizontal')  
>>> im = ax.imshow(img,  
>>> cmap='seismic')
```

Markers

```
>>> fig, ax = plt.subplots()  
>>> ax.plot(x, y, marker='o')  
>>> ax.plot(x, y, marker='*')
```

Line Styles

```
>>> plt.plot(x,y,linewidth=4.0)  
>>> plt.plot(x,y,'solid')  
>>> plt.plot(x,y,'dashed')  
>>> plt.plot(x,y,'--',x**2,y**2,'-')  
>>> plt.setp(lines,color='r',linewidth=4.0)
```

Text & Annotations

```
>>> ax.text(2,1,  
>>> "Example Graph",  
>>> style='italic')  
>>> ax.annotate("Sine",  
>>> xy=(0, 1),  
>>> xycoords='data',  
>>> xytext=(10.5, 0),  
>>> textcoords='offset pixels',  
>>> arrowprops=dict(arrowstyle=">",  
>>> connectionstyle="arc3",
```

Mathtext

```
>>> plt.title(r"\Sigma_{i=1}^{10} i^2", fontsize=20)
```

Limits, Legend & Layouts

Limits & Autoscaling

```
>>> ax.set_xlim(0,0.1)  
>>> ax.axis('equal')  
>>> ax.set_xlim(0,10.5); ylim=[-1.5,1.5]  
>>> ax.set_xlim(0,10.5)
```

Legends

```
>>> ax.set_title('An Example Axes',  
>>> ylabel='Y-axis',  
>>> xlabel='X-axis')  
>>> ax.legend(loc='best')
```

Ticks

```
>>> ax.xaxis.set_ticks(range(1,5),  
>>> ticks=[3,10,-12,"foo"])  
>>> ax.tick_params(axis='y',  
>>> direction='inout',  
>>> length=10)
```

Subplot Spacing

```
>>> fig.subplots_adjust(wspace=0.5,  
>>> hspace=0.3,  
>>> left=0.125,  
>>> right=0.9,  
>>> top=0.9,  
>>> bottom=0.1)
```

Fit subplot(s) in to the figure area

```
>>> fig.tight_layout()
```

Axis Spines

```
>>> ax.spines["top"].set_visible(False)  
>>> ax.spines["bottom"].set_position(("outward",10))
```

Adjust the spacing between subplots

Add padding to a plot

Set the aspect ratio of the plot to 1

Set limits for x and y axes

Set title and x and y axis labels

No overlapping plot elements

Manually set ticks

Make y ticks longer and go in and out

Fit subplot(s) in to the figure area

Make the top axis line for a plot invisible

Move the bottom axis line outward

5 Save Plot

Save Figures

```
>>> plt.savefig('foo.png')  
>>> plt.savefig('foo.png', transparent=True)
```

6 Show Plot

```
>>> plt.show()
```

Close & Clear

Clear axis

```
>>> plt.cla()
```

Close the figure

```
>>> plt.close()
```

Close a window

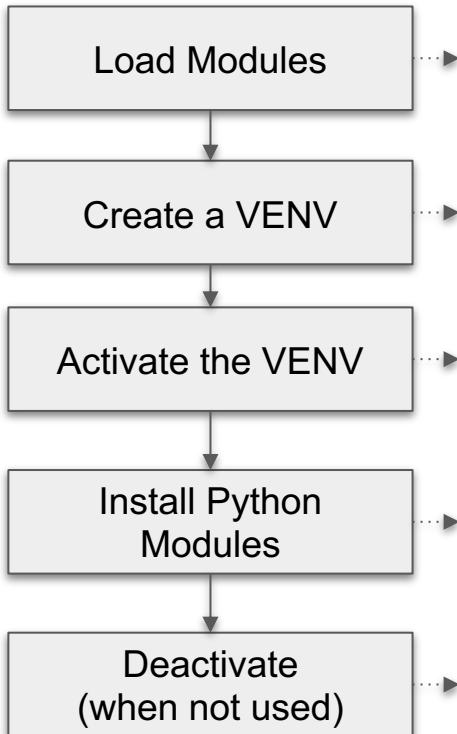
```
>>> plt.close()
```

DataCamp
Learn Python For Data Science Interactively 

https://s3.amazonaws.com/assets.datacamp.com/blog_assets/Python_Matplotlib_Cheat_Sheet.pdf

Additional Materials

Python Virtual Environment (VENV)



```
# Change to /scratch/user/netid directory
cd $SCRATCH

# clean up and load Anaconda
module purge
module load Anaconda/3-5.0.0.1

# create a Python virtual environment
conda create -n mylab

# activate the virtual environment
source activate mylab

# install required package to be used in the portal
conda install -c conda-forge jupyterlab=1.2.2
conda install pandas matplotlib
conda install scikit-learn
conda install tensorflow

# deactivate the virtual environment
# source deactivate
```

Common Anaconda Commands

```
# Conda virtual environment
conda info                                # show Conda installation
conda create -n VENV                      # create a virtual environment
conda create -n VENV python=3.4             # create a venv with a py version
conda env list                            # list installed venv

# Conda package management
conda list                                 # list all installed packages
conda search PACKAGENAME                  # search a Conda package
conda install PACKAGENAME                 # install a Conda package
conda update PACKAGENAME                  # update a Conda package
conda remove PACKAGENAME                  # remove a Conda package
```