(Brief) Intro to (Scientific) Python!

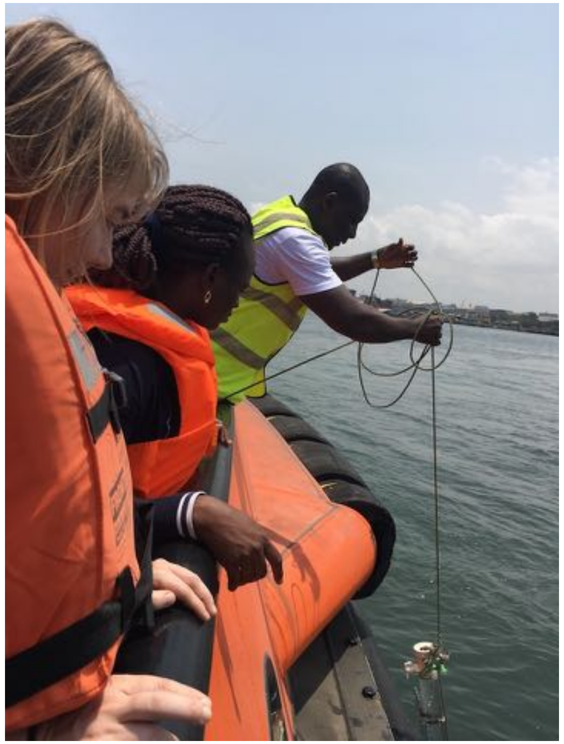
Dr. Paige Martin

COESSING 2019

What is oceanographic data?

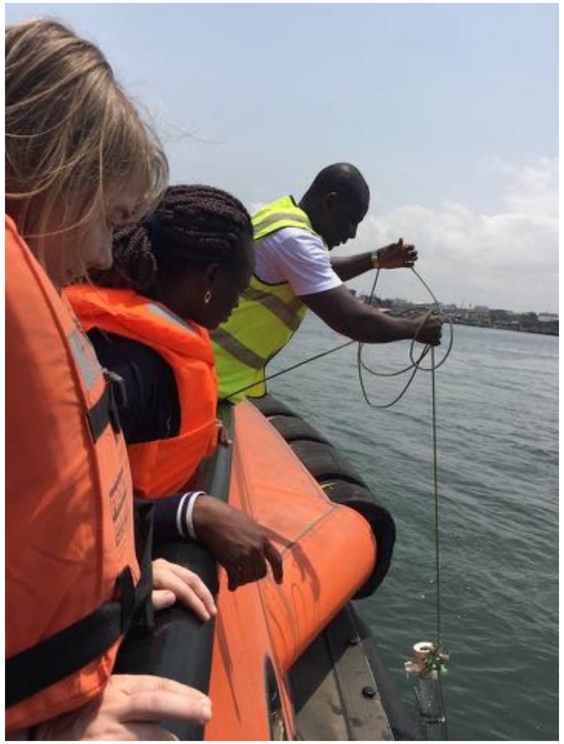
What is oceanographic data?

In-Situ Observations

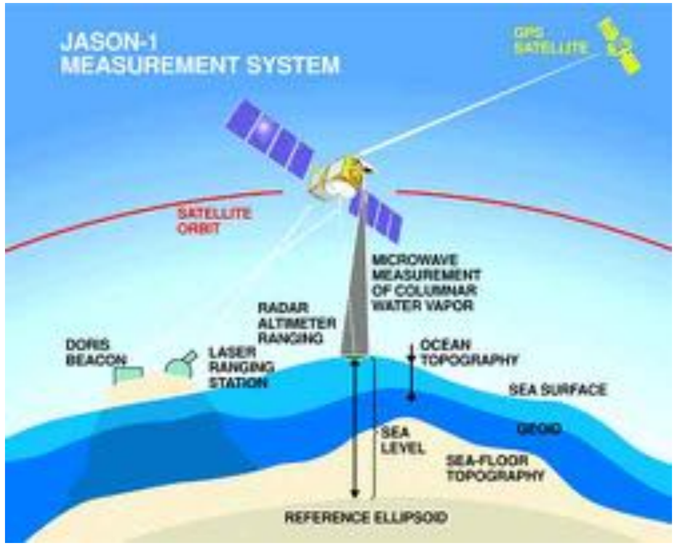


What is oceanographic data?

In-Situ Observations

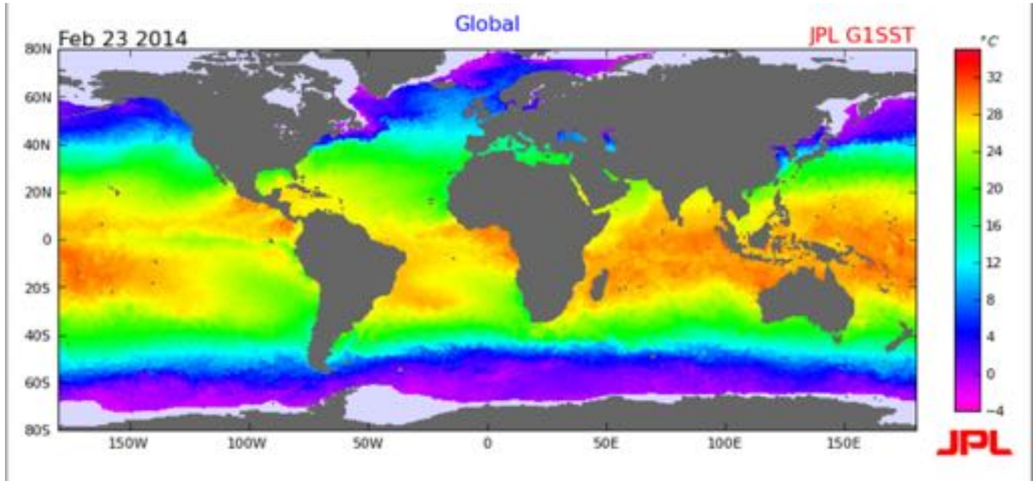


Satellite Data

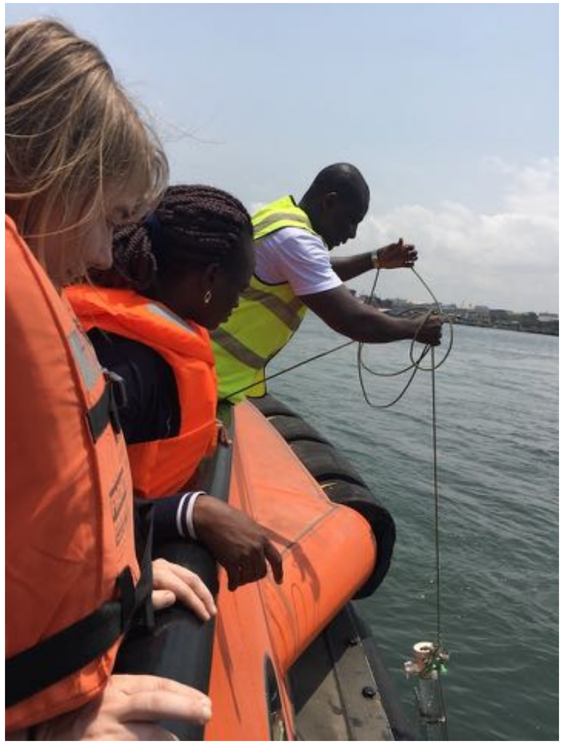


What is oceanographic data?

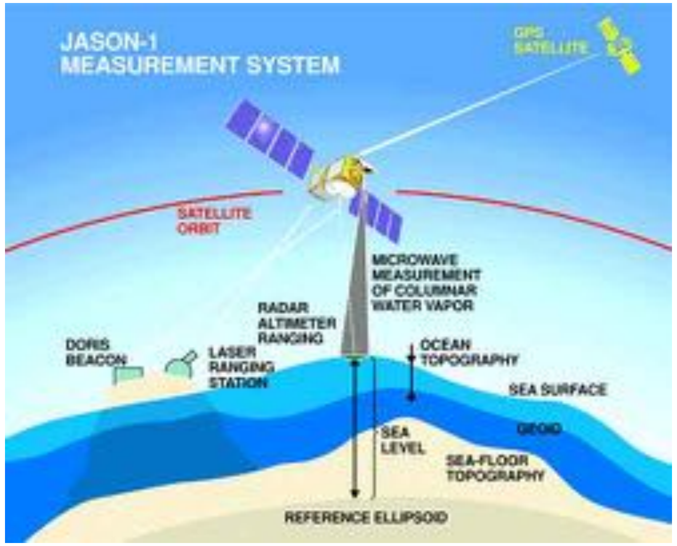
Model output



In-Situ Observations



Satellite Data



Why do we need data analysis?

Why do we need data analysis?

- Data analysis is how we translate raw data into interesting scientific results!

Why do we need data analysis?

- Data analysis is how we translate raw data into interesting scientific results!
- Data analysis:

Why do we need data analysis?

- Data analysis is how we translate raw data into interesting scientific results!

- Data analysis:
  - Manipulate data

Why do we need data analysis?

- Data analysis is how we translate raw data into interesting scientific results!

- Data analysis:
  - Manipulate data

Velocity v → Kinetic energy = $\frac{1}{2} m v^2$

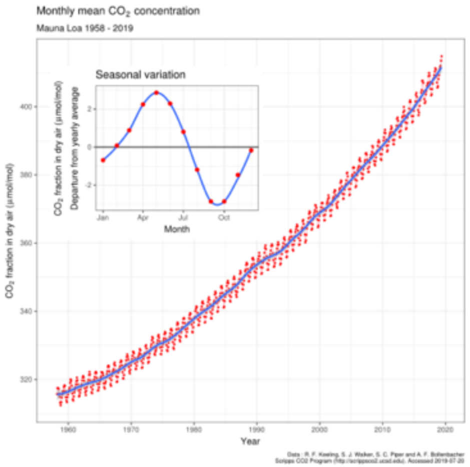
Why do we need data analysis?

- Data analysis is how we translate raw data into interesting scientific results!

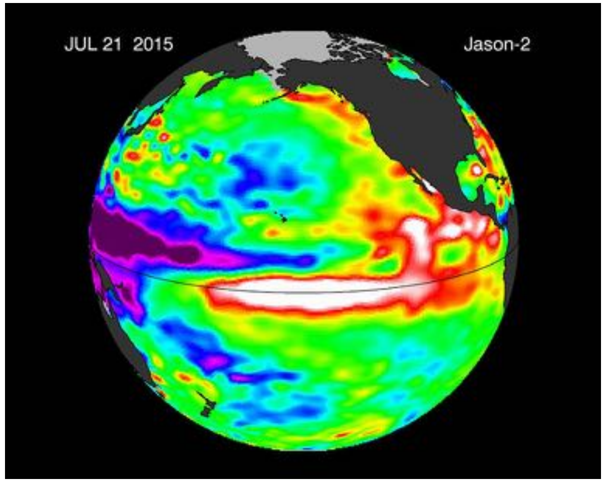
- Data analysis:
  - Manipulate data

Velocity v → Kinetic energy = ½ $m\, v^2$

  - Visualize data



Data : R. F. Keeling, S. J. Walker, S. C. Piper and A. F. Bollenbacher
Scripps CO2 Program (http://scrippsco2.ucsd.edu). Accessed 2019-07-20



What is Python?

What is Python?

- It’s a computer language – it allows us as humans to communicate with the computer and make it do what we want!

What is Python?

- It’s a computer language – it allows us as humans to communicate with the computer and make it do what we want!

What is Python?

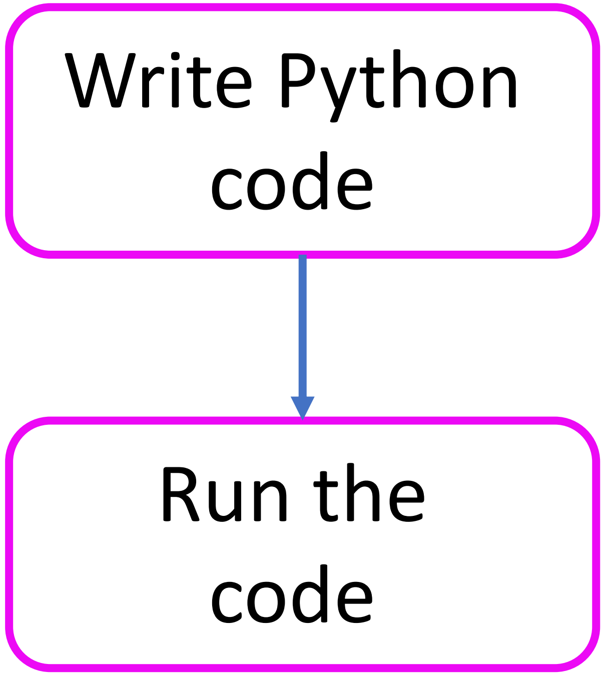
- It’s a computer language – it allows us as humans to communicate with the computer and make it do what we want!

How does Python work?

How does Python work?

Write Python code

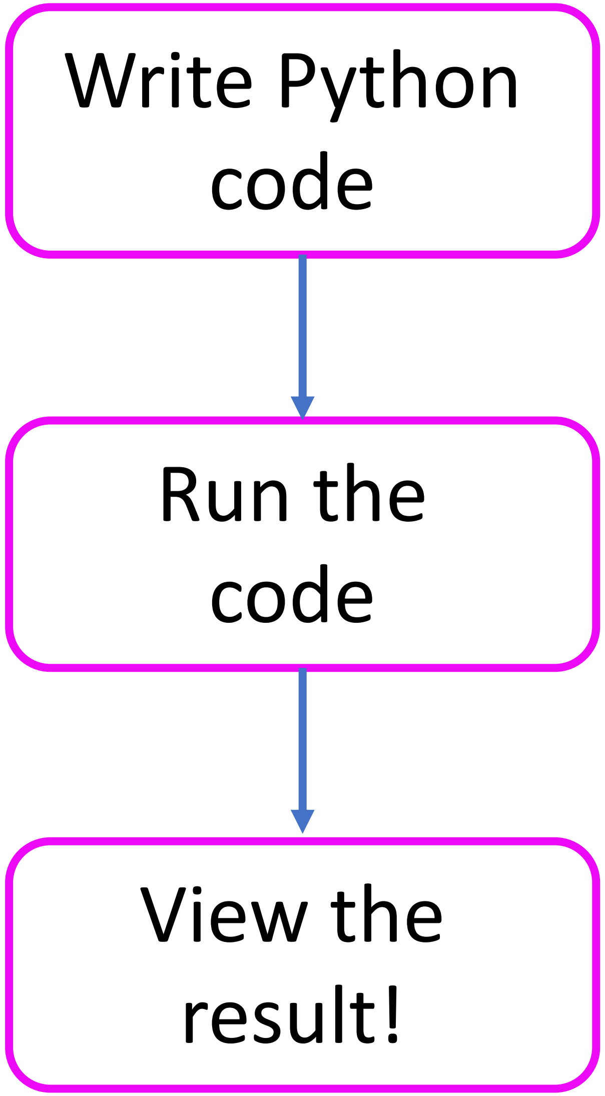
How does Python work?



Write Python code

Run the code

How does Python work?

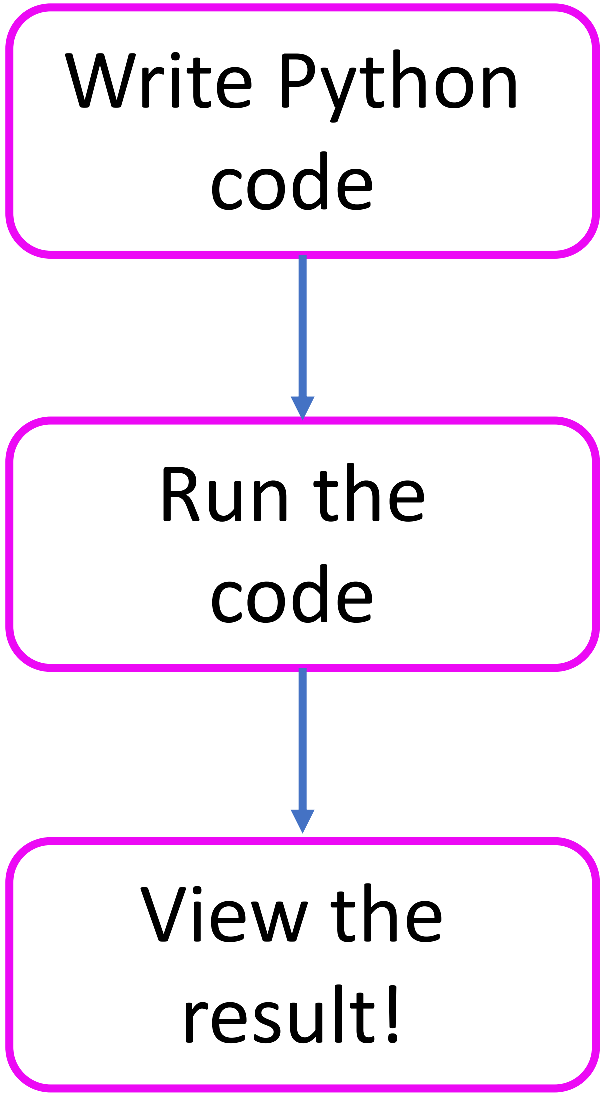


Write Python code

Run the code

View the result!

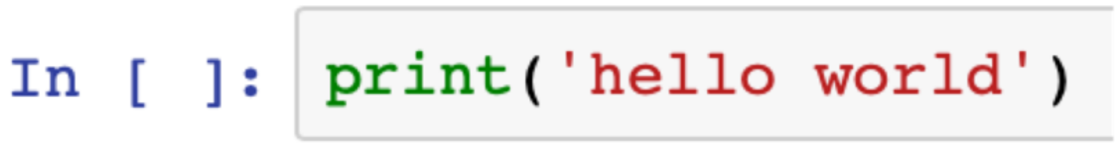
How does Python work?



Write Python code

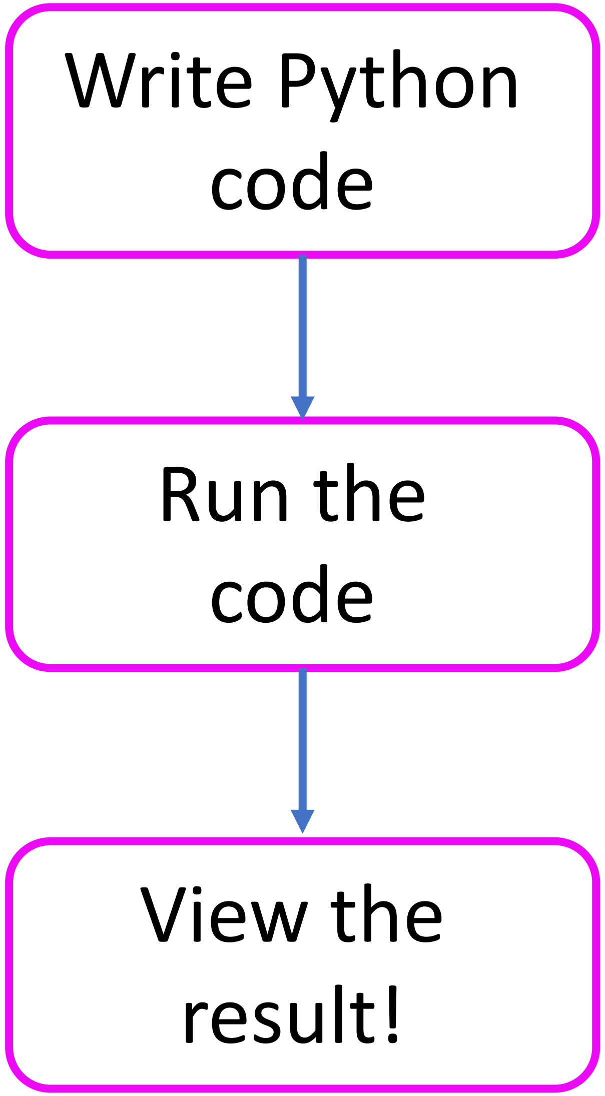
Run the code

View the result!



```
In [ ]: print('hello world')
```

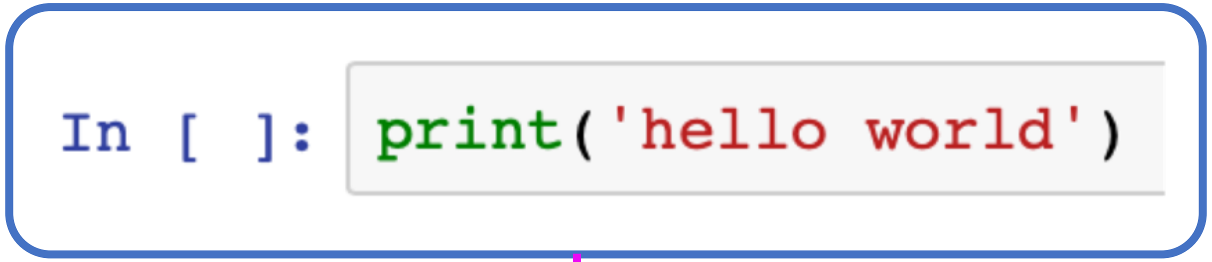
How does Python work?



Write Python code

Run the code

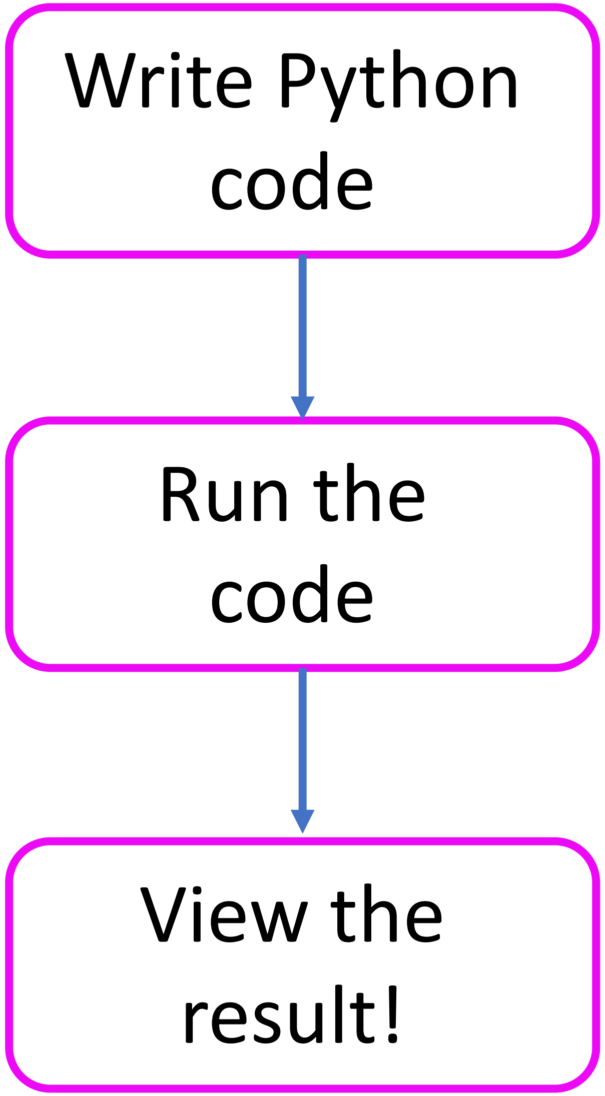
View the result!



```
In [ ]: print('hello world')
```

Run the code

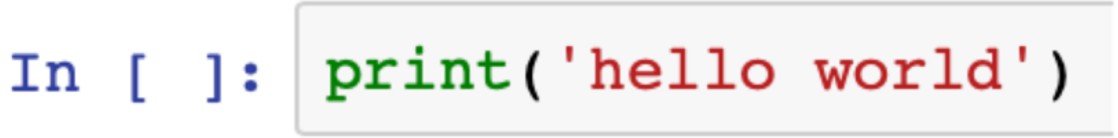
How does Python work?



Write Python code

Run the code

View the result!

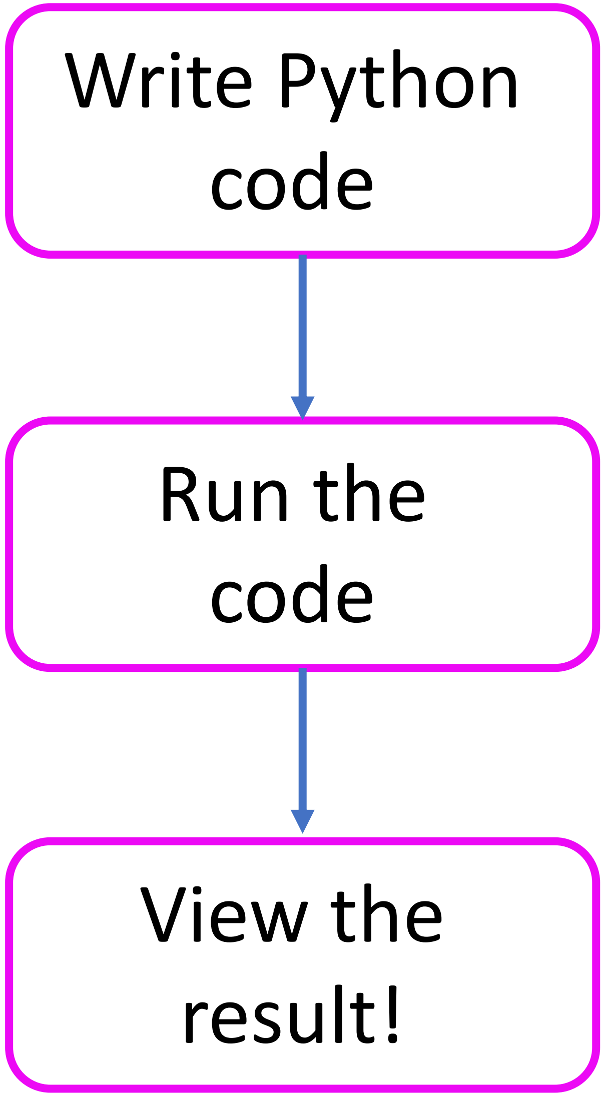


```
In [ ]: print('hello world')
```

Run the code

```
hello world
```

How does Python work?

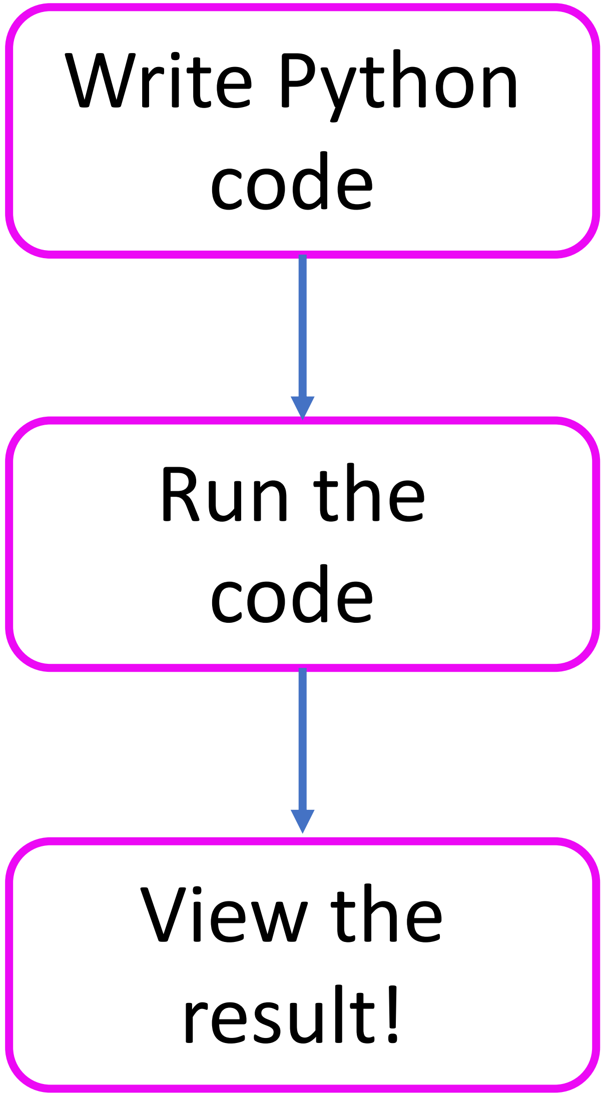


Write Python code

Run the code

View the result!

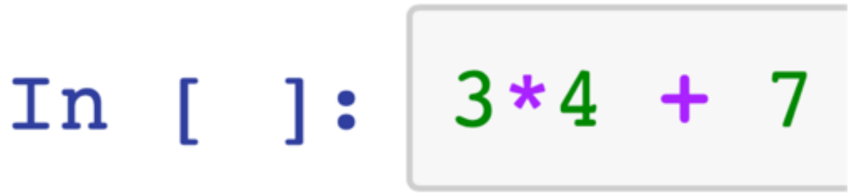
How does Python work?



Write Python code

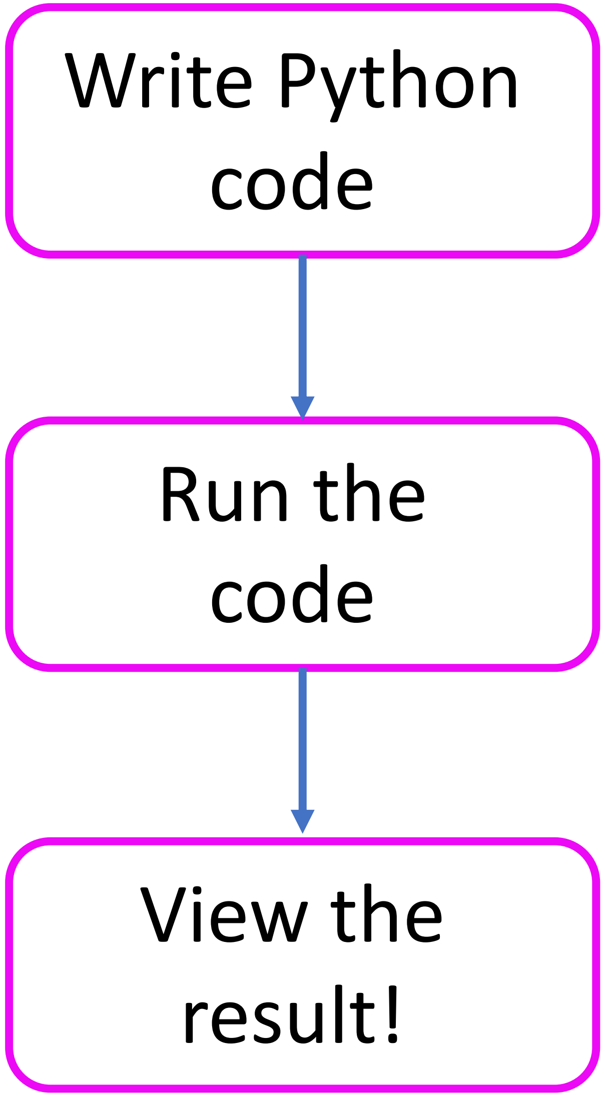
Run the code

View the result!



```
In [ ]: 3*4 + 7
```

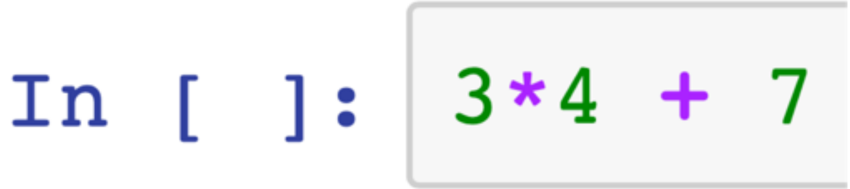
How does Python work?



Write Python code

Run the code

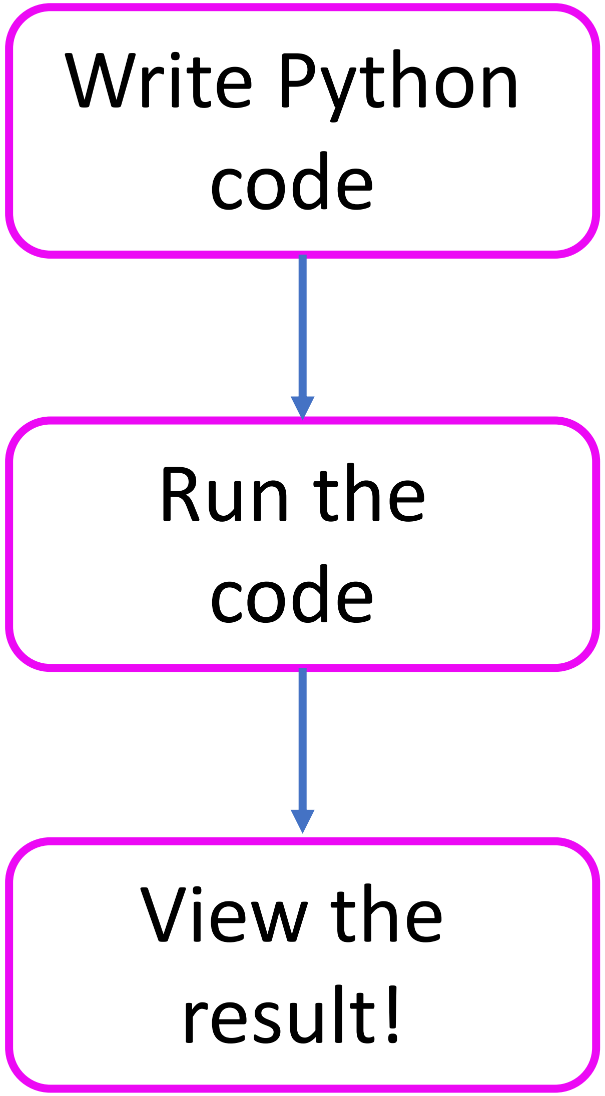
View the result!



```
In [ ]: 3*4 + 7
```

Run the code

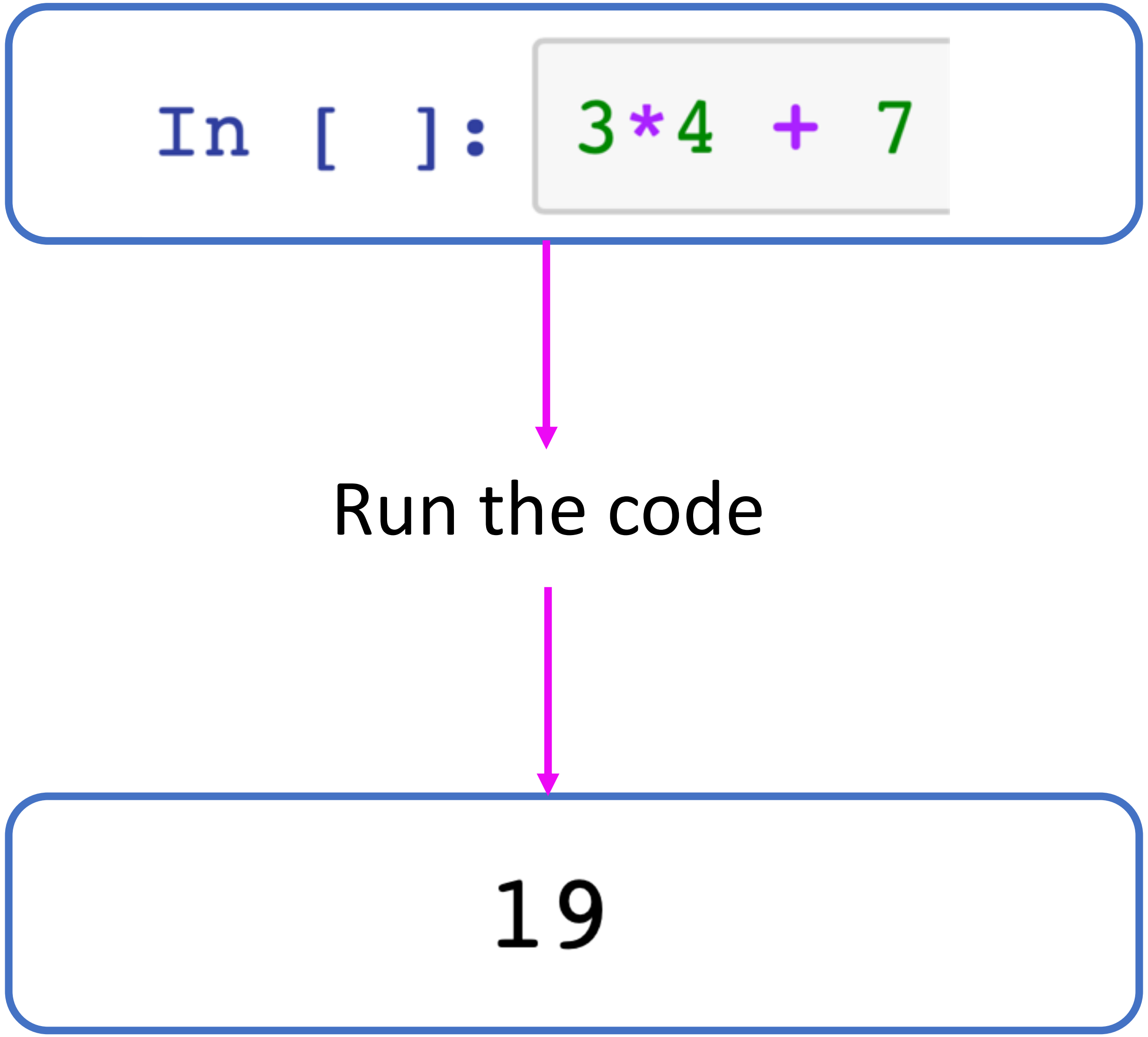
How does Python work?



Write Python code

Run the code

View the result!



```
In [ ]: 3*4 + 7
```

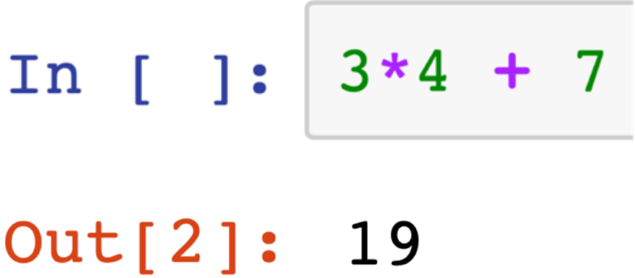
Run the code

```
19
```

How do I write Python code?

How do I write Python code?

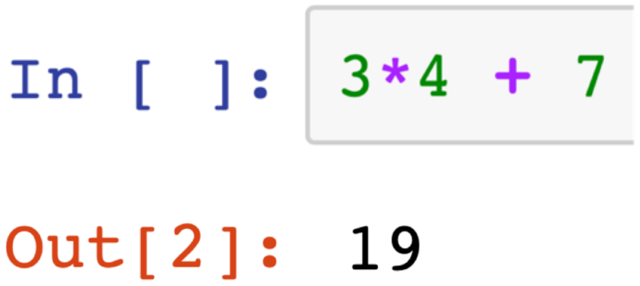
- Line-by-line:



```
In [ ]: 3*4 + 7

Out[2]: 19
```

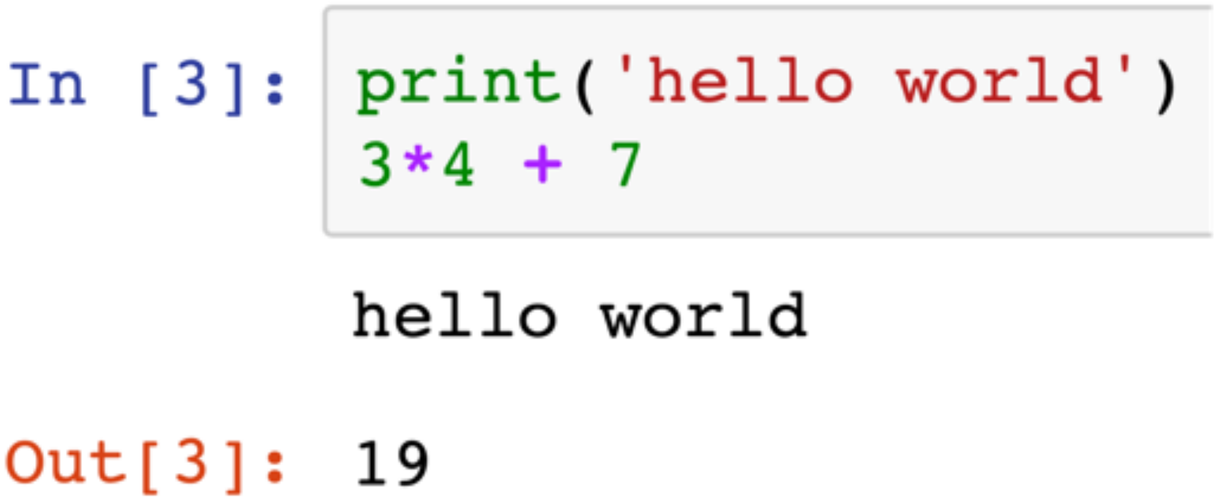
How do I write Python code?

- Line-by-line:



```
In [ ]: 3*4 + 7

Out[2]: 19
```

- Usually, we write scripts –
  - Many lines of code
  - Run the entire script all together



```
In [3]: print('hello world')
        3*4 + 7

        hello world

Out[3]: 19
```

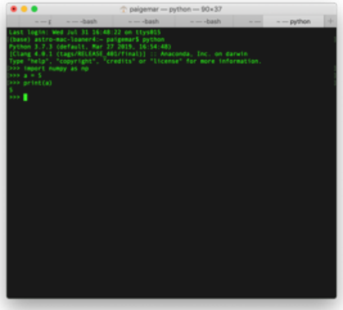
What editor should I use?

What editor should I use?

- There are many to choose from!

What editor should I use?

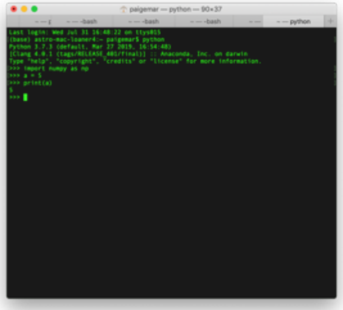
- There are many to choose from!
    - Terminal (command line)



```
Last login: Wed Jul 31 16:48:22 on ttys015
(base) astro-mac-loaner4:~ paigemar$ python
Python 3.7.3 (default, Mar 27 2019, 16:54:48)
[Clang 4.0.1 (tags/RELEASE_401/final)] :: Anaconda, Inc. on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> import numpy as np
>>> a = 5
>>> print(a)
5
>>> 
```

What editor should I use?

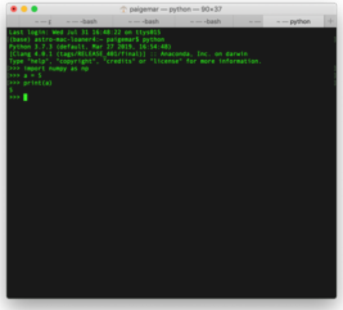
- There are many to choose from!
    - Terminal (command line)
    - Jupyter Notebook



```
Last login: Wed Jul 31 16:48:22 on ttys015
(base) astro-mac-loaner4:~ paigemar$ python
Python 3.7.3 (default, Mar 27 2019, 16:54:48)
[Clang 4.0.1 (tags/RELEASE_401/final)] :: Anaconda, Inc. on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> import numpy as np
>>> a = 5
>>> print(a)
5
>>> 
```

What editor should I use?

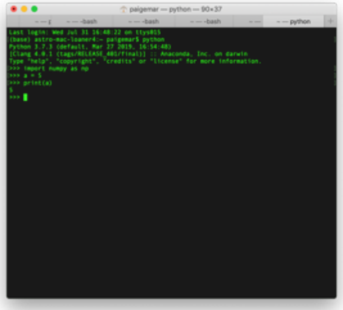
- There are many to choose from!
  - Terminal (command line)
  - Jupyter Notebook
  - Spyder, PyCharm, Atom, …



```
Last login: Wed Jul 31 16:48:22 on ttys015
(base) astro-mac-loaner4:~ paigemar$ python
Python 3.7.3 (default, Mar 27 2019, 16:54:48)
[Clang 4.0.1 (tags/RELEASE_401/final)] :: Anaconda, Inc. on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> import numpy as np
>>> a = 5
>>> print(a)
5
>>> 
```

What editor should I use?

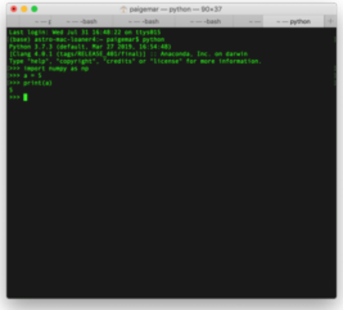
- There are many to choose from!
  - Terminal (command line)
  - Jupyter Notebook
  - Spyder, PyCharm, Atom, …
  - I will focus on Jupyter Notebook, because it’s what I use. It is a way of writing interactive Python code.



```
Last login: Wed Jul 31 16:48:22 on ttys015
(base) astro-mac-loaner4:~ paigemar$ python
Python 3.7.3 (default, Mar 27 2019, 16:54:48)
[Clang 4.0.1 (tags/RELEASE_401/final)] :: Anaconda, Inc. on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> import numpy as np
>>> a = 5
>>> print(a)
5
>>>
```

What editor should I use?

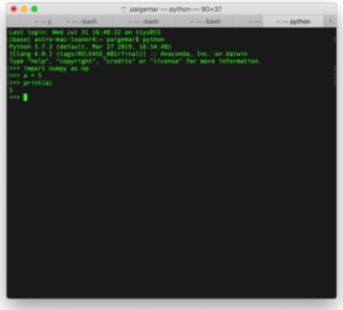
- There are many to choose from!
  - Terminal (command line)
  - Jupyter Notebook
  - Spyder, PyCharm, Atom, …
  - I will focus on Jupyter Notebook, because it’s what I use. It is a way of writing interactive Python code.
- I highly suggest downloading Python via Anaconda



```
Last login: Wed Jul 31 16:48:22 on ttys015
(base) astro-mac-loaner4:~ paigemar$ python
Python 3.7.3 (default, Mar 27 2019, 16:54:48)
[Clang 4.0.1 (tags/RELEASE_401/final)] :: Anaconda, Inc. on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> import numpy as np
>>> a = 5
>>> print(a)
5
>>> 
```

What editor should I use?

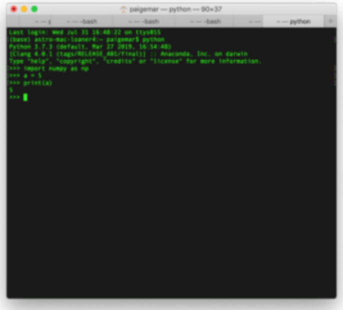
- There are many to choose from!
  - Terminal (command line)
  - Jupyter Notebook
  - Spyder, PyCharm, Atom, …
  - I will focus on Jupyter Notebook, because it’s what I use. It is a way of writing interactive Python code.
- I highly suggest downloading Python via Anaconda
  - Anaconda packages the Python language together with many useful libraries and script editors



```
Last login: Wed Jul 31 16:48:22 on ttys015
(base) astro-mac-loaner4:~ paigemar$ python
Python 3.7.3 (default, Mar 27 2019, 16:54:48)
[Clang 4.0.1 (tags/RELEASE_401/final)] :: Anaconda, Inc. on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> import numpy as np
>>> a = 5
>>> print(a)
5
>>> 
```

What editor should I use?

- There are many to choose from!
  - Terminal (command line)
  - Jupyter Notebook
  - Spyder, PyCharm, Atom, …
  - I will focus on Jupyter Notebook, because it's what I use. It is a way of writing interactive Python code.



```
Last login: Wed Jul 31 16:48:22 on ttys015
(base) astro-mac-loaner4:~ paigemar$ python
Python 3.7.3 (default, Mar 27 2019, 16:54:48)
[Clang 4.0.1 (tags/RELEASE_401/final)] :: Anaconda, Inc. on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> import numpy as np
>>> a = 5
>>> print(a)
5
>>> 
```

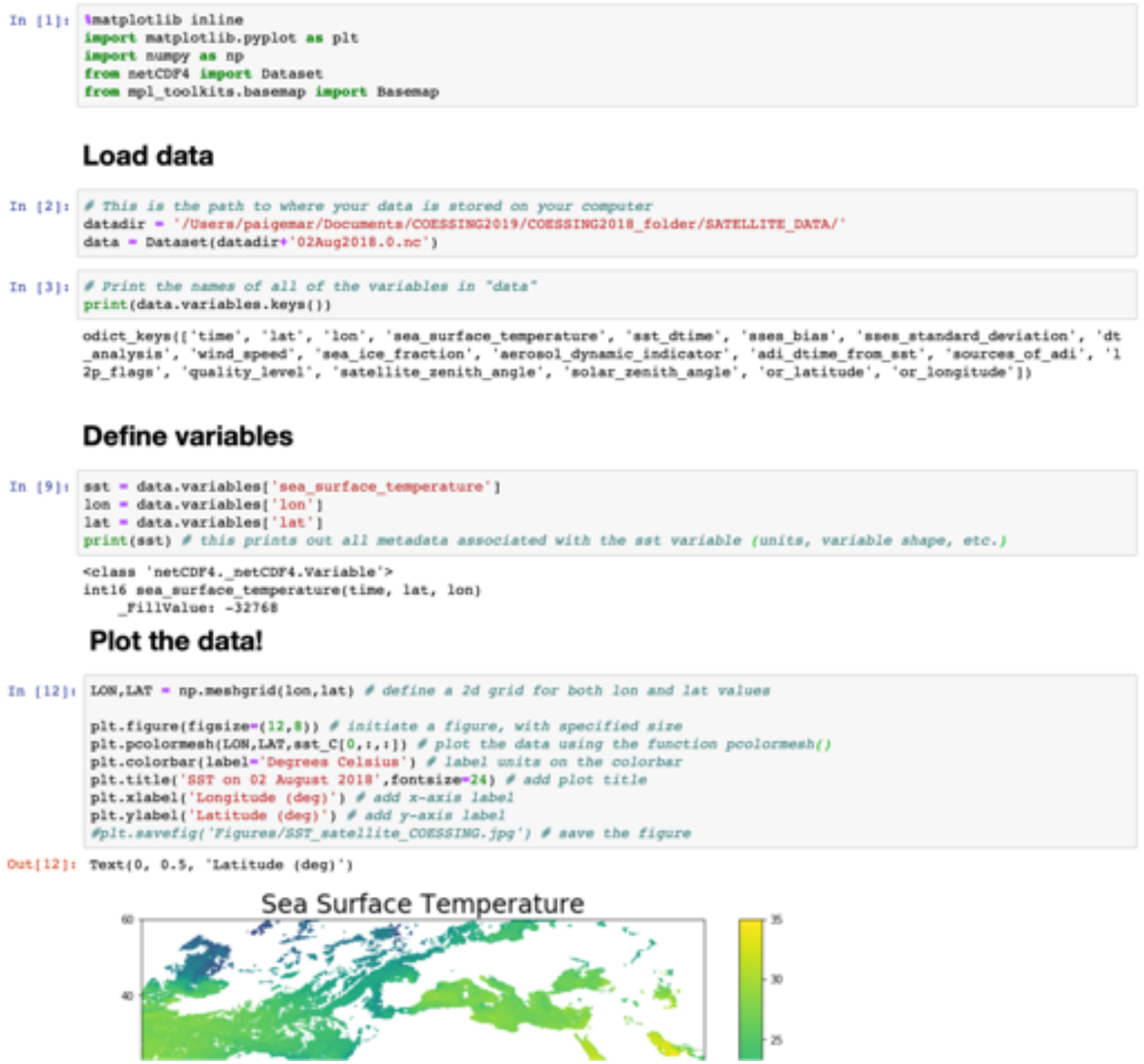
- I highly suggest downloading Python via Anaconda
  - Anaconda packages the Python language together with many useful libraries and script editors
  - Anaconda includes Jupyter notebook – an interactive Python editor

How do I write a Python script in Jupyter notebook?

How do I write a Python script in Jupyter notebook?

Let’s look at a sample script!

How do I write a Python script in Jupyter notebook?



```
In [1]: %matplotlib inline
        import matplotlib.pyplot as plt
        import numpy as np
        from netCDF4 import Dataset
        from mpl_toolkits.basemap import Basemap
```

Load data

```
In [2]: # This is the path to where your data is stored on your computer
        datadir = '/Users/paigemar/Documents/COESSING2019/COESSING2018_folder/SATELLITE_DATA/'
        data = Dataset(datadir+'02Aug2018.0.nc')
```

```
In [3]: # Print the names of all of the variables in "data"
        print(data.variables.keys())

        odict_keys(['time', 'lat', 'lon', 'sea_surface_temperature', 'sst_dtime', 'sses_bias', 'sses_standard_deviation', 'dt
        _analysis', 'wind_speed', 'sea_ice_fraction', 'aerosol_dynamic_indicator', 'adi_dtime_from_sst', 'sources_of_adi', 'l
        2p_flags', 'quality_level', 'satellite_zenith_angle', 'solar_zenith_angle', 'or_latitude', 'or_longitude'])
```

Define variables

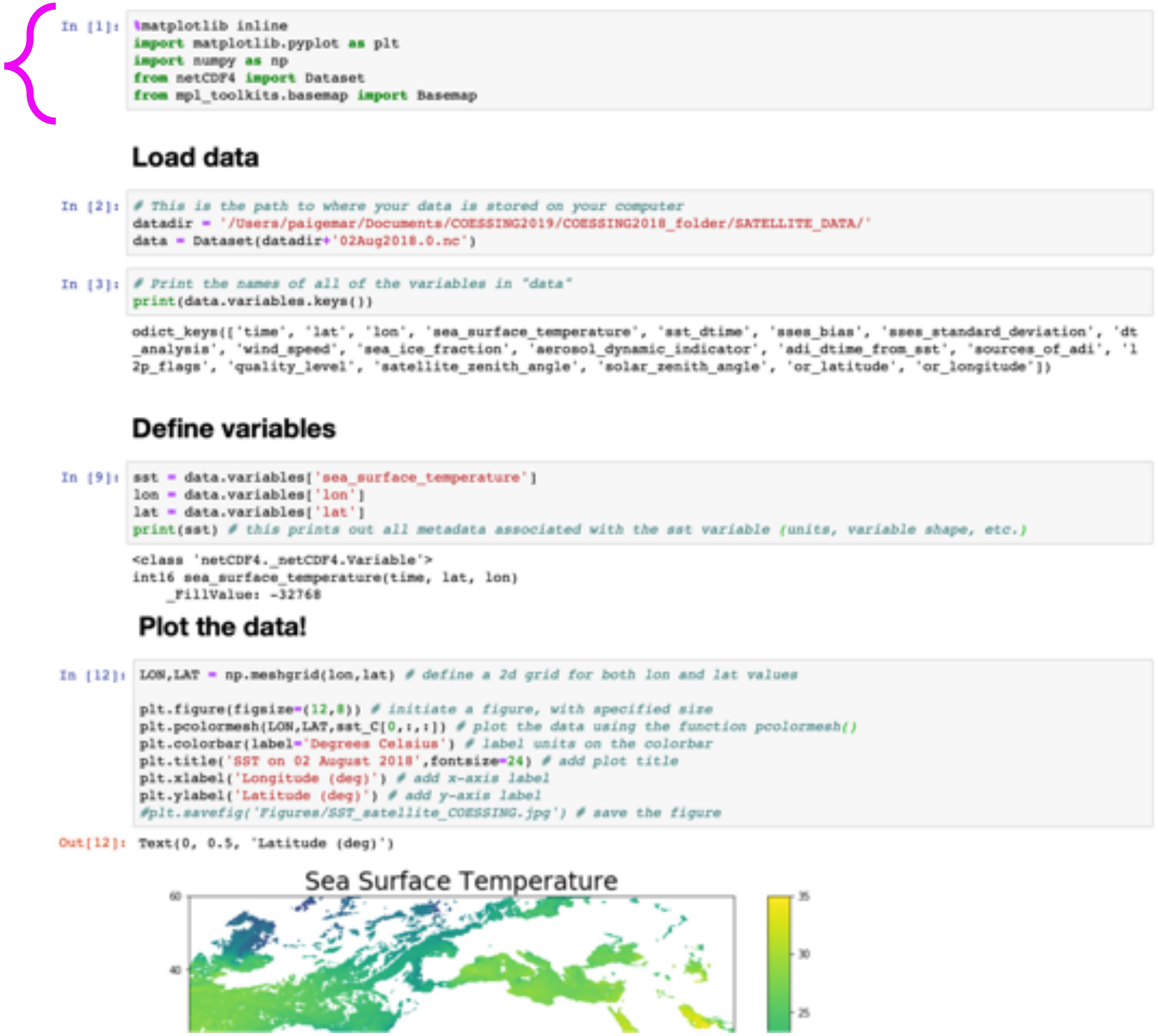
```
In [9]: sst = data.variables['sea_surface_temperature']
        lon = data.variables['lon']
        lat = data.variables['lat']
        print(sst) # this prints out all metadata associated with the sst variable (units, variable shape, etc.)

        <class 'netCDF4._netCDF4.Variable'>
        int16 sea_surface_temperature(time, lat, lon)
            _FillValue: -32768
```

Plot the data!

```
In [12]: LON,LAT = np.meshgrid(lon,lat) # define a 2d grid for both lon and lat values

         plt.figure(figsize=(12,8)) # initiate a figure, with specified size
         plt.pcolormesh(LON,LAT,sst_C[0,:,:]) # plot the data using the function pcolormesh()
         plt.colorbar(label='Degrees Celsius') # label units on the colorbar
         plt.title('SST on 02 August 2018',fontsize=24) # add plot title
         plt.xlabel('Longitude (deg)') # add x-axis label
         plt.ylabel('Latitude (deg)') # add y-axis label
         #plt.savefig('Figures/SST_satellite_COESSING.jpg') # save the figure

Out[12]: Text(0, 0.5, 'Latitude (deg)')
```

How do I write a Python script in Jupyter notebook?

Import libraries



```
In [1]: %matplotlib inline
        import matplotlib.pyplot as plt
        import numpy as np
        from netCDF4 import Dataset
        from mpl_toolkits.basemap import Basemap
```

Load data

```
In [2]: # This is the path to where your data is stored on your computer
        datadir = '/Users/paigemar/Documents/COESSING2019/COESSING2018_folder/SATELLITE_DATA/'
        data = Dataset(datadir+'02Aug2018.0.nc')
```

```
In [3]: # Print the names of all of the variables in "data"
        print(data.variables.keys())

        odict_keys(['time', 'lat', 'lon', 'sea_surface_temperature', 'sst_dtime', 'sses_bias', 'sses_standard_deviation', 'dt
        _analysis', 'wind_speed', 'sea_ice_fraction', 'aerosol_dynamic_indicator', 'adi_dtime_from_sst', 'sources_of_adi', 'l
        2p_flags', 'quality_level', 'satellite_zenith_angle', 'solar_zenith_angle', 'or_latitude', 'or_longitude'])
```

Define variables

```
In [9]: sst = data.variables['sea_surface_temperature']
        lon = data.variables['lon']
        lat = data.variables['lat']
        print(sst) # this prints out all metadata associated with the sst variable (units, variable shape, etc.)

        <class 'netCDF4._netCDF4.Variable'>
        int16 sea_surface_temperature(time, lat, lon)
            _FillValue: -32768
```

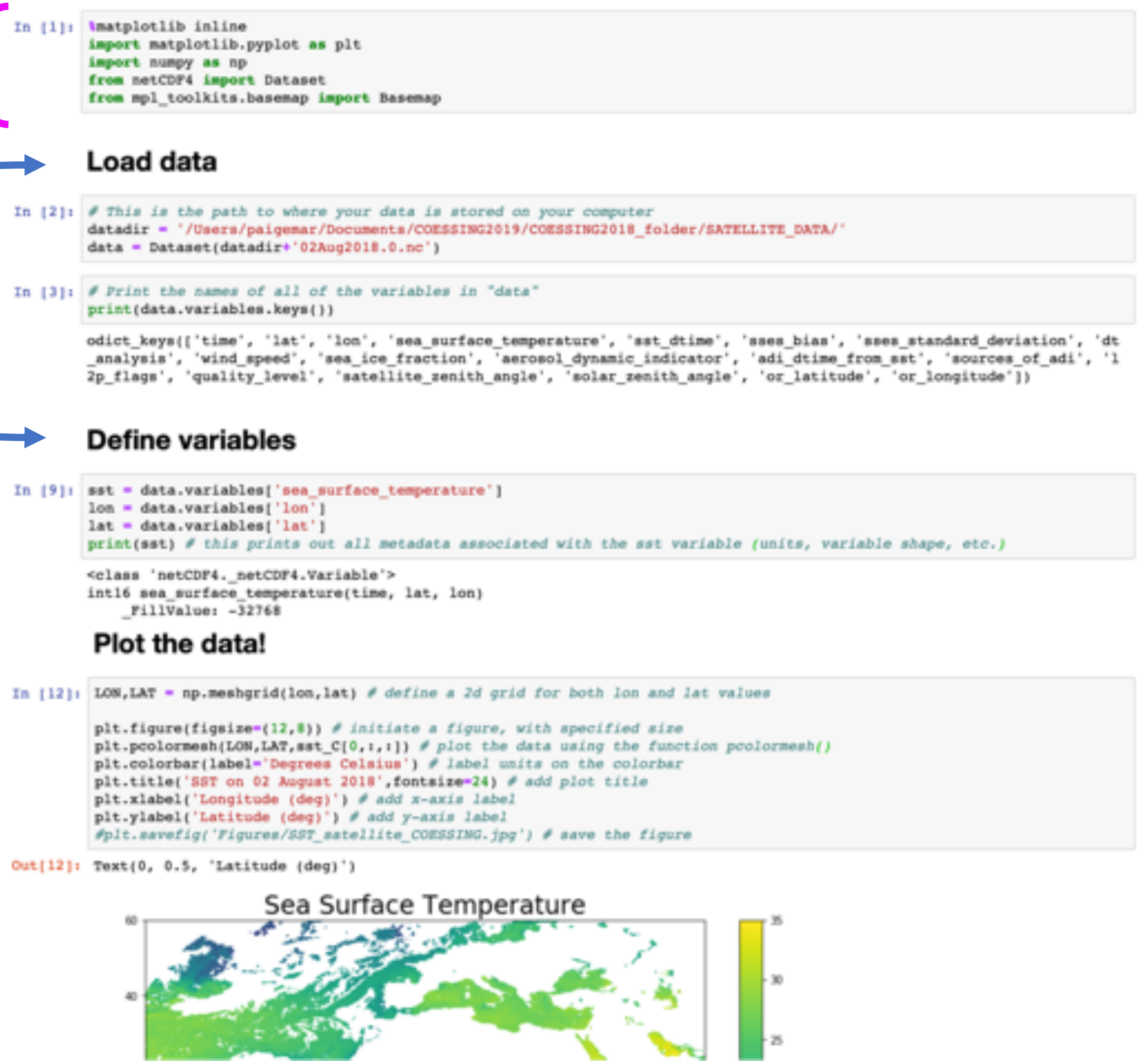
Plot the data!

```
In [12]: LON,LAT = np.meshgrid(lon,lat) # define a 2d grid for both lon and lat values

         plt.figure(figsize=(12,8)) # initiate a figure, with specified size
         plt.pcolormesh(LON,LAT,sst_C[0,:,:]) # plot the data using the function pcolormesh()
         plt.colorbar(label='Degrees Celsius') # label units on the colorbar
         plt.title('SST on 02 August 2018',fontsize=24) # add plot title
         plt.xlabel('Longitude (deg)') # add x-axis label
         plt.ylabel('Latitude (deg)') # add y-axis label
         #plt.savefig('Figures/SST_satellite_COESSING.jpg') # save the figure

Out[12]: Text(0, 0.5, 'Latitude (deg)')
```

How do I write a Python script in Jupyter notebook?

Import libraries

Comments



```
In [1]: %matplotlib inline
        import matplotlib.pyplot as plt
        import numpy as np
        from netCDF4 import Dataset
        from mpl_toolkits.basemap import Basemap
```

Load data

```
In [2]: # This is the path to where your data is stored on your computer
        datadir = '/Users/paigemar/Documents/COESSING2019/COESSING2018_folder/SATELLITE_DATA/'
        data = Dataset(datadir+'02Aug2018.0.nc')
```

```
In [3]: # Print the names of all of the variables in "data"
        print(data.variables.keys())
```

```
odict_keys(['time', 'lat', 'lon', 'sea_surface_temperature', 'sst_dtime', 'sses_bias', 'sses_standard_deviation', 'dt
_analysis', 'wind_speed', 'sea_ice_fraction', 'aerosol_dynamic_indicator', 'adi_dtime_from_sst', 'sources_of_adi', 'l
2p_flags', 'quality_level', 'satellite_zenith_angle', 'solar_zenith_angle', 'or_latitude', 'or_longitude'])
```

Define variables

```
In [9]: sst = data.variables['sea_surface_temperature']
        lon = data.variables['lon']
        lat = data.variables['lat']
        print(sst) # this prints out all metadata associated with the sst variable (units, variable shape, etc.)
```

```
<class 'netCDF4._netCDF4.Variable'>
int16 sea_surface_temperature(time, lat, lon)
    _FillValue: -32768
```

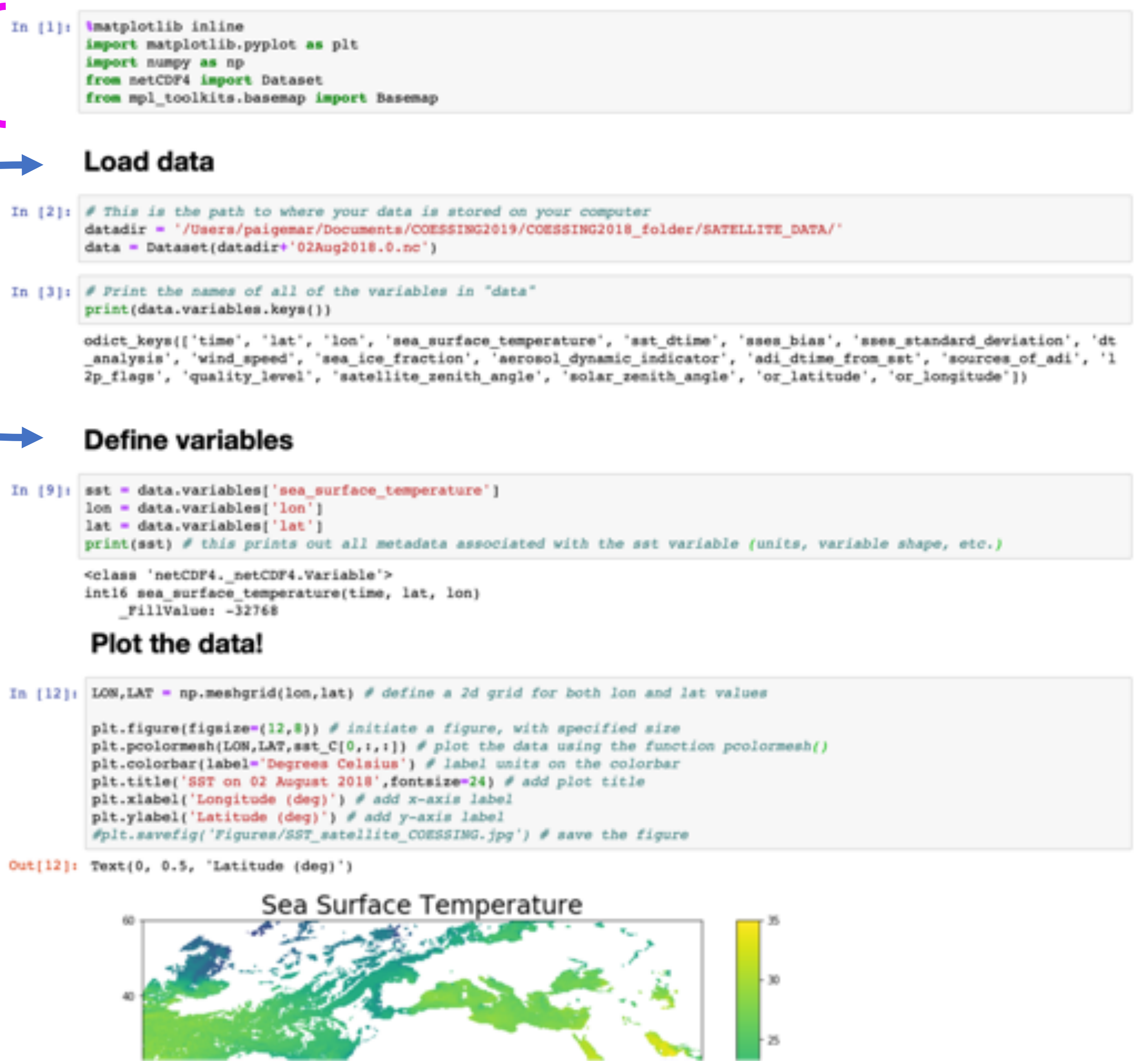
Plot the data!

```
In [12]: LON,LAT = np.meshgrid(lon,lat) # define a 2d grid for both lon and lat values

         plt.figure(figsize=(12,8)) # initiate a figure, with specified size
         plt.pcolormesh(LON,LAT,sst_C[0,:,:]) # plot the data using the function pcolormesh()
         plt.colorbar(label='Degrees Celsius') # label units on the colorbar
         plt.title('SST on 02 August 2018',fontsize=24) # add plot title
         plt.xlabel('Longitude (deg)') # add x-axis label
         plt.ylabel('Latitude (deg)') # add y-axis label
         #plt.savefig('Figures/SST_satellite_COESSING.jpg') # save the figure

Out[12]: Text(0, 0.5, 'Latitude (deg)')
```

How do I write a Python script in Jupyter notebook?

Import libraries

Comments



```
In [1]: %matplotlib inline
        import matplotlib.pyplot as plt
        import numpy as np
        from netCDF4 import Dataset
        from mpl_toolkits.basemap import Basemap
```

Load data

```
In [2]: # This is the path to where your data is stored on your computer
        datadir = '/Users/paigemar/Documents/COESSING2019/COESSING2018_folder/SATELLITE_DATA/'
        data = Dataset(datadir+'02Aug2018.0.nc')
```

```
In [3]: # Print the names of all of the variables in "data"
        print(data.variables.keys())

        odict_keys(['time', 'lat', 'lon', 'sea_surface_temperature', 'sst_dtime', 'sses_bias', 'sses_standard_deviation', 'dt
        _analysis', 'wind_speed', 'sea_ice_fraction', 'aerosol_dynamic_indicator', 'adi_dtime_from_sst', 'sources_of_adi', 'l
        2p_flags', 'quality_level', 'satellite_zenith_angle', 'solar_zenith_angle', 'or_latitude', 'or_longitude'])
```

Define variables

```
In [9]: sst = data.variables['sea_surface_temperature']
        lon = data.variables['lon']
        lat = data.variables['lat']
        print(sst) # this prints out all metadata associated with the sst variable (units, variable shape, etc.)

        <class 'netCDF4._netCDF4.Variable'>
        int16 sea_surface_temperature(time, lat, lon)
            _FillValue: -32768
```

Plot the data!

```
In [12]: LON,LAT = np.meshgrid(lon,lat) # define a 2d grid for both lon and lat values

         plt.figure(figsize=(12,8)) # initiate a figure, with specified size
         plt.pcolormesh(LON,LAT,sst_C[0,:,:]) # plot the data using the function pcolormesh()
         plt.colorbar(label='Degrees Celsius') # label units on the colorbar
         plt.title('SST on 02 August 2018',fontsize=24) # add plot title
         plt.xlabel('Longitude (deg)') # add x-axis label
         plt.ylabel('Latitude (deg)') # add y-axis label
         #plt.savefig('Figures/SST_satellite_COESSING.jpg') # save the figure

Out[12]: Text(0, 0.5, 'Latitude (deg)')
```

Main code

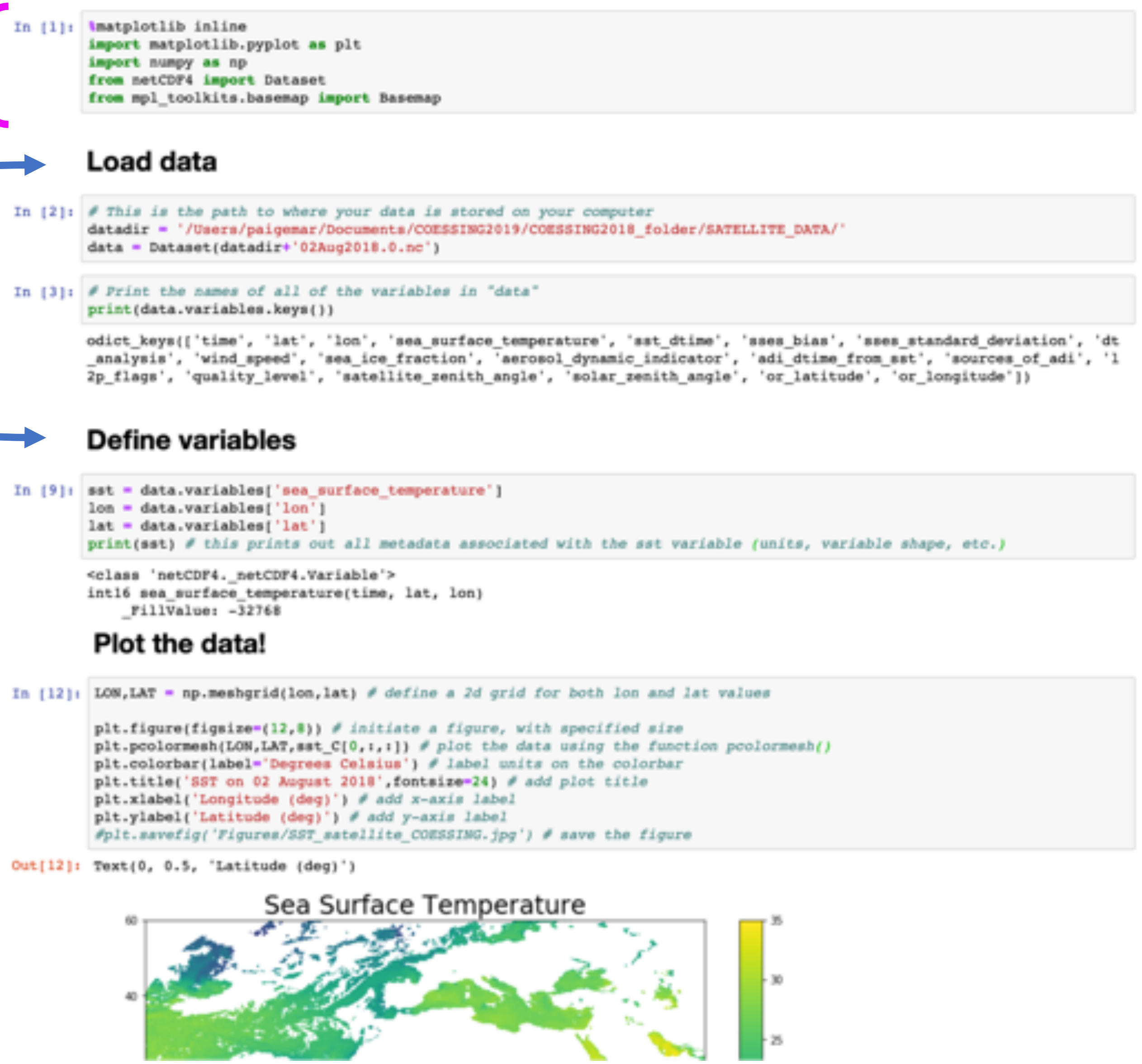
How do I write a Python script in Jupyter notebook?

Import libraries

Comments

Load your data

Main code



```
In [1]: %matplotlib inline
        import matplotlib.pyplot as plt
        import numpy as np
        from netCDF4 import Dataset
        from mpl_toolkits.basemap import Basemap
```

Load data

```
In [2]: # This is the path to where your data is stored on your computer
        datadir = '/Users/paigemar/Documents/COESSING2019/COESSING2018_folder/SATELLITE_DATA/'
        data = Dataset(datadir+'02Aug2018.0.nc')

In [3]: # Print the names of all of the variables in "data"
        print(data.variables.keys())

        odict_keys(['time', 'lat', 'lon', 'sea_surface_temperature', 'sst_dtime', 'sses_bias', 'sses_standard_deviation', 'dt
        _analysis', 'wind_speed', 'sea_ice_fraction', 'aerosol_dynamic_indicator', 'adi_dtime_from_sst', 'sources_of_adi', 'l
        2p_flags', 'quality_level', 'satellite_zenith_angle', 'solar_zenith_angle', 'or_latitude', 'or_longitude'])
```

Define variables

```
In [9]: sst = data.variables['sea_surface_temperature']
        lon = data.variables['lon']
        lat = data.variables['lat']
        print(sst) # this prints out all metadata associated with the sst variable (units, variable shape, etc.)

        <class 'netCDF4._netCDF4.Variable'>
        int16 sea_surface_temperature(time, lat, lon)
            _FillValue: -32768
```

Plot the data!

```
In [12]: LON,LAT = np.meshgrid(lon,lat) # define a 2d grid for both lon and lat values

         plt.figure(figsize=(12,8)) # initiate a figure, with specified size
         plt.pcolormesh(LON,LAT,sst_C[0,:,:]) # plot the data using the function pcolormesh()
         plt.colorbar(label='Degrees Celsius') # label units on the colorbar
         plt.title('SST on 02 August 2018',fontsize=24) # add plot title
         plt.xlabel('Longitude (deg)') # add x-axis label
         plt.ylabel('Latitude (deg)') # add y-axis label
         #plt.savefig('Figures/SST_satellite_COESSING.jpg') # save the figure

Out[12]: Text(0, 0.5, 'Latitude (deg)')
```

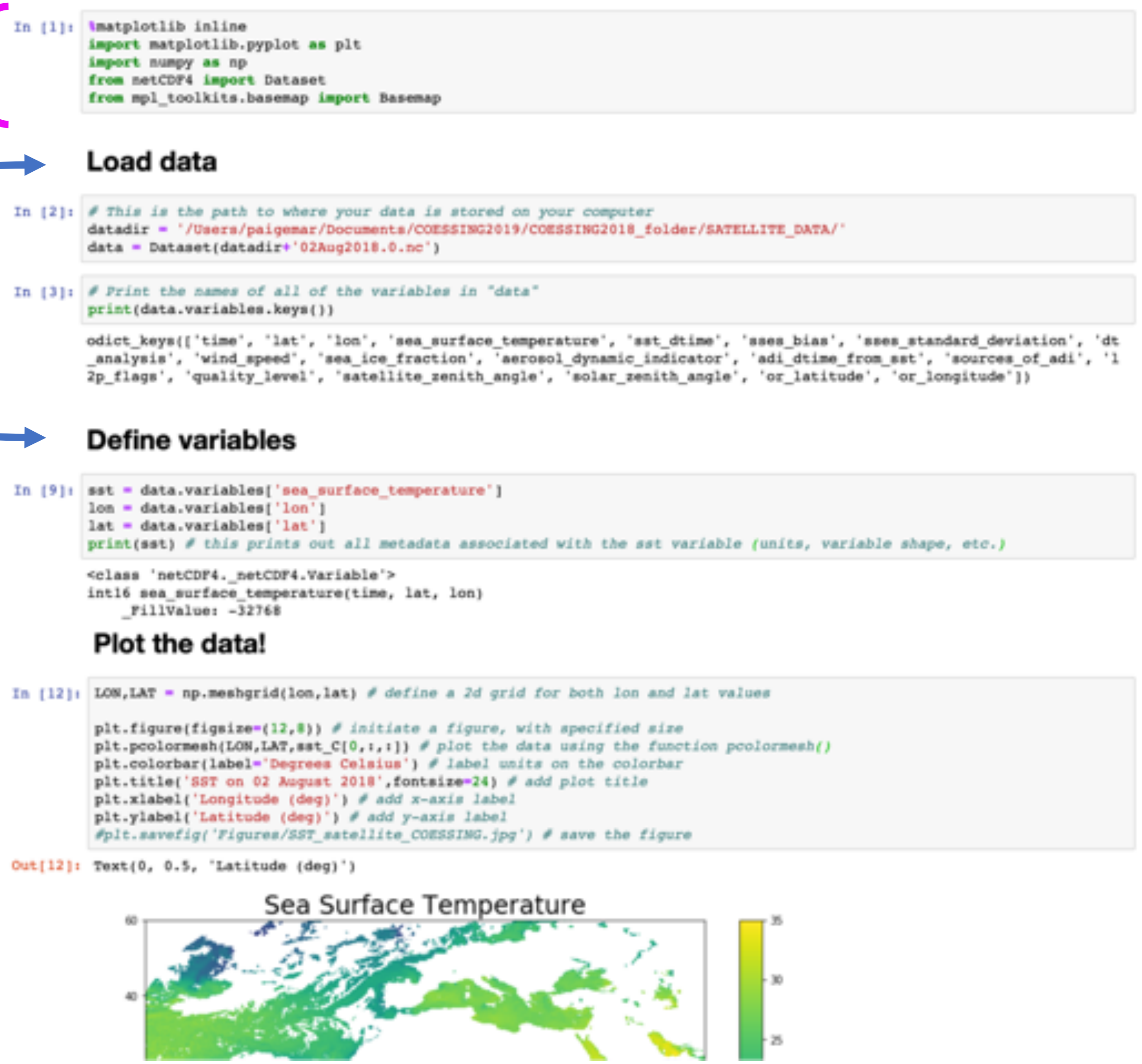
How do I write a Python script in Jupyter notebook?

Import libraries

Comments

Load your data

Data analysis



```
In [1]: %matplotlib inline
        import matplotlib.pyplot as plt
        import numpy as np
        from netCDF4 import Dataset
        from mpl_toolkits.basemap import Basemap
```

Load data

```
In [2]: # This is the path to where your data is stored on your computer
        datadir = '/Users/paigemar/Documents/COESSING2019/COESSING2018_folder/SATELLITE_DATA/'
        data = Dataset(datadir+'02Aug2018.0.nc')
```

```
In [3]: # Print the names of all of the variables in "data"
        print(data.variables.keys())

        odict_keys(['time', 'lat', 'lon', 'sea_surface_temperature', 'sst_dtime', 'sses_bias', 'sses_standard_deviation', 'dt
        _analysis', 'wind_speed', 'sea_ice_fraction', 'aerosol_dynamic_indicator', 'adi_dtime_from_sst', 'sources_of_adi', 'l
        2p_flags', 'quality_level', 'satellite_zenith_angle', 'solar_zenith_angle', 'or_latitude', 'or_longitude'])
```

Define variables

```
In [9]: sst = data.variables['sea_surface_temperature']
        lon = data.variables['lon']
        lat = data.variables['lat']
        print(sst) # this prints out all metadata associated with the sst variable (units, variable shape, etc.)

        <class 'netCDF4._netCDF4.Variable'>
        int16 sea_surface_temperature(time, lat, lon)
            _FillValue: -32768
```

Plot the data!

```
In [12]: LON,LAT = np.meshgrid(lon,lat) # define a 2d grid for both lon and lat values

         plt.figure(figsize=(12,8)) # initiate a figure, with specified size
         plt.pcolormesh(LON,LAT,sst_C[0,:,:]) # plot the data using the function pcolormesh()
         plt.colorbar(label='Degrees Celsius') # label units on the colorbar
         plt.title('SST on 02 August 2018',fontsize=24) # add plot title
         plt.xlabel('Longitude (deg)') # add x-axis label
         plt.ylabel('Latitude (deg)') # add y-axis label
         #plt.savefig('Figures/SST_satellite_COESSING.jpg') # save the figure

Out[12]: Text(0, 0.5, 'Latitude (deg)')
```

Main code

How do I write a Python script in Jupyter notebook?

Import libraries

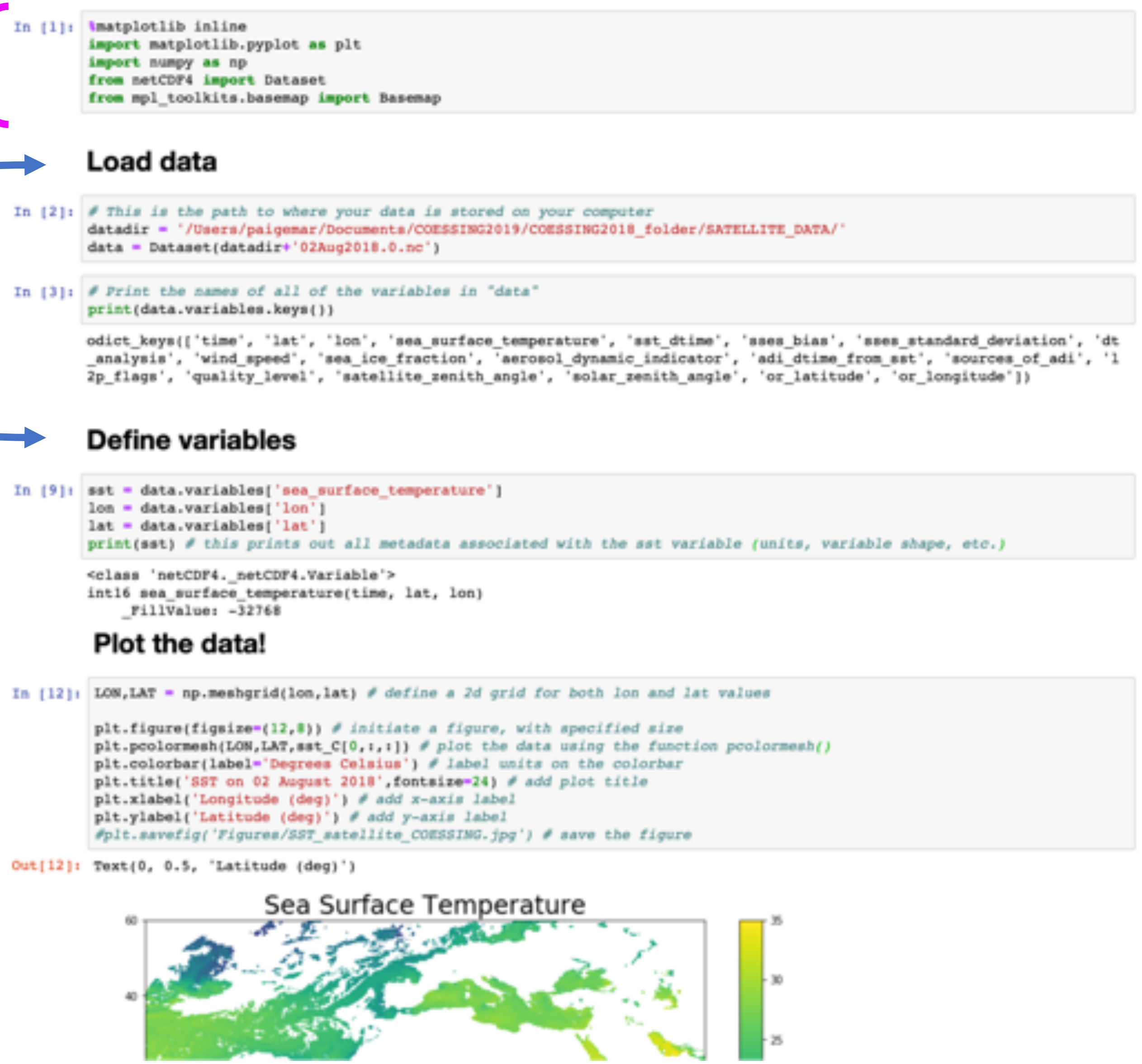
Comments

Load your data

Data analysis

Visualize your data

Main code



```
In [1]: %matplotlib inline
        import matplotlib.pyplot as plt
        import numpy as np
        from netCDF4 import Dataset
        from mpl_toolkits.basemap import Basemap
```

Load data

```
In [2]: # This is the path to where your data is stored on your computer
        datadir = '/Users/paigemar/Documents/COESSING2019/COESSING2018_folder/SATELLITE_DATA/'
        data = Dataset(datadir+'02Aug2018.0.nc')

In [3]: # Print the names of all of the variables in "data"
        print(data.variables.keys())

        odict_keys(['time', 'lat', 'lon', 'sea_surface_temperature', 'sst_dtime', 'sses_bias', 'sses_standard_deviation', 'dt
        _analysis', 'wind_speed', 'sea_ice_fraction', 'aerosol_dynamic_indicator', 'adi_dtime_from_sst', 'sources_of_adi', 'l
        2p_flags', 'quality_level', 'satellite_zenith_angle', 'solar_zenith_angle', 'or_latitude', 'or_longitude'])
```

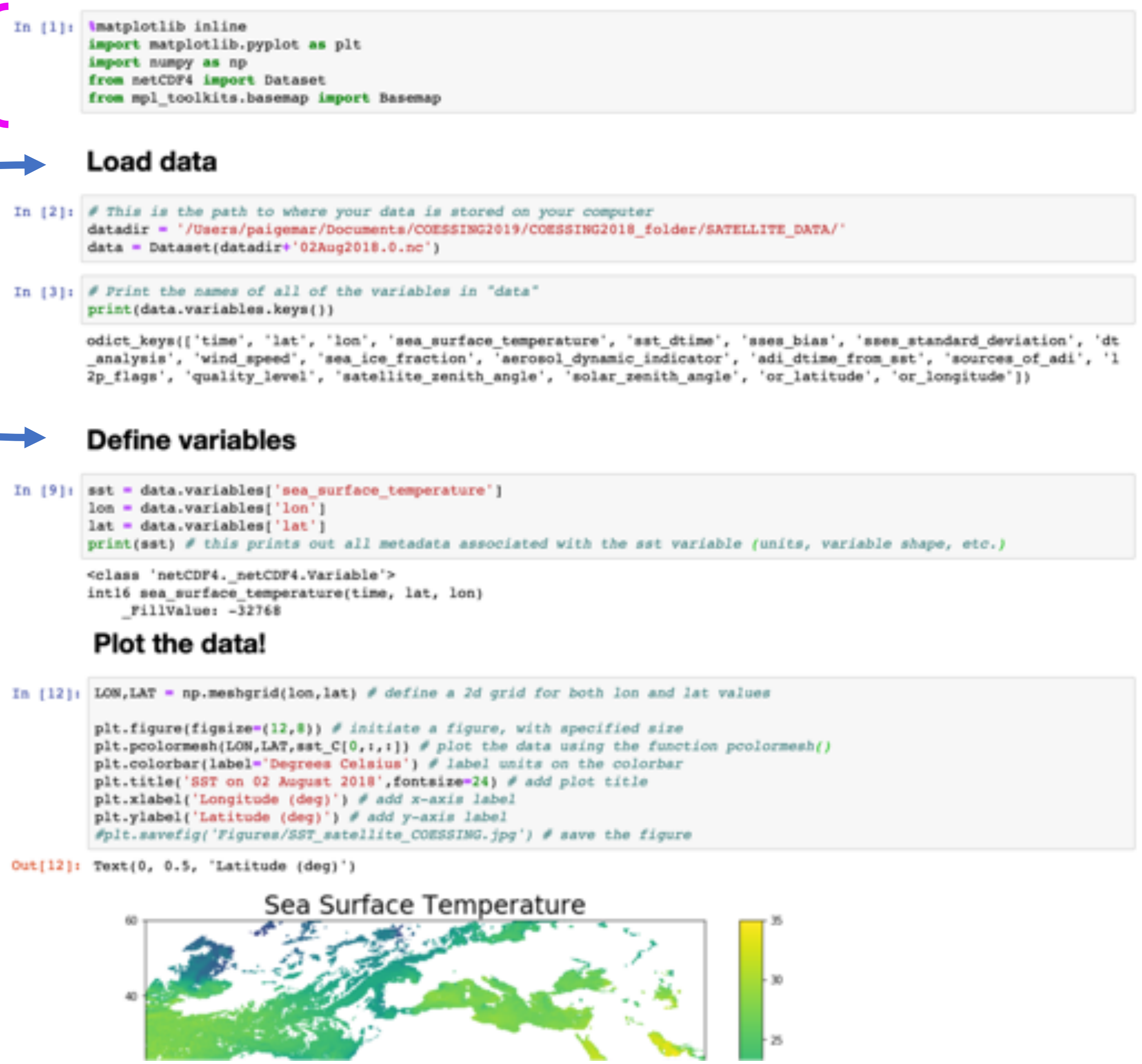
Define variables

```
In [9]: sst = data.variables['sea_surface_temperature']
        lon = data.variables['lon']
        lat = data.variables['lat']
        print(sst) # this prints out all metadata associated with the sst variable (units, variable shape, etc.)

        <class 'netCDF4._netCDF4.Variable'>
        int16 sea_surface_temperature(time, lat, lon)
            _FillValue: -32768
```

Plot the data!

```
In [12]: LON,LAT = np.meshgrid(lon,lat) # define a 2d grid for both lon and lat values

         plt.figure(figsize=(12,8)) # initiate a figure, with specified size
         plt.pcolormesh(LON,LAT,sst_C[0,:,:]) # plot the data using the function pcolormesh()
         plt.colorbar(label='Degrees Celsius') # label units on the colorbar
         plt.title('SST on 02 August 2018',fontsize=24) # add plot title
         plt.xlabel('Longitude (deg)') # add x-axis label
         plt.ylabel('Latitude (deg)') # add y-axis label
         #plt.savefig('Figures/SST_satellite_COESSING.jpg') # save the figure

Out[12]: Text(0, 0.5, 'Latitude (deg)')
```

How do I write a Python script in Jupyter notebook?



```
In [1]: %matplotlib inline
        import matplotlib.pyplot as plt
        import numpy as np
        from netCDF4 import Dataset
        from mpl_toolkits.basemap import Basemap
```

Load data

```
In [2]: # This is the path to where your data is stored on your computer
        datadir = '/Users/paigemar/Documents/COESSING2019/COESSING2018_folder/SATELLITE_DATA/'
        data = Dataset(datadir+'02Aug2018.0.nc')

In [3]: # Print the names of all of the variables in "data"
        print(data.variables.keys())

        odict_keys(['time', 'lat', 'lon', 'sea_surface_temperature', 'sst_dtime', 'sses_bias', 'sses_standard_deviation', 'dt
        _analysis', 'wind_speed', 'sea_ice_fraction', 'aerosol_dynamic_indicator', 'adi_dtime_from_sst', 'sources_of_adi', 'l
        2p_flags', 'quality_level', 'satellite_zenith_angle', 'solar_zenith_angle', 'or_latitude', 'or_longitude'])
```

Define variables

```
In [9]: sst = data.variables['sea_surface_temperature']
        lon = data.variables['lon']
        lat = data.variables['lat']
        print(sst) # this prints out all metadata associated with the sst variable (units, variable shape, etc.)

        <class 'netCDF4._netCDF4.Variable'>
        int16 sea_surface_temperature(time, lat, lon)
            _FillValue: -32768
```

Plot the data!

```
In [12]: LON,LAT = np.meshgrid(lon,lat) # define a 2d grid for both lon and lat values

         plt.figure(figsize=(12,8)) # initiate a figure, with specified size
         plt.pcolormesh(LON,LAT,sst_C[0,:,:]) # plot the data using the function pcolormesh()
         plt.colorbar(label='Degrees Celsius') # label units on the colorbar
         plt.title('SST on 02 August 2018',fontsize=24) # add plot title
         plt.xlabel('Longitude (deg)') # add x-axis label
         plt.ylabel('Latitude (deg)') # add y-axis label
         #plt.savefig('Figures/SST_satellite_COESSING.jpg') # save the figure

Out[12]: Text(0, 0.5, 'Latitude (deg)')
```

Comments

Import libraries

Load your data

Data analysis

Visualize your data

Main code

How do I write a Python script in Jupyter notebook?

Import libraries

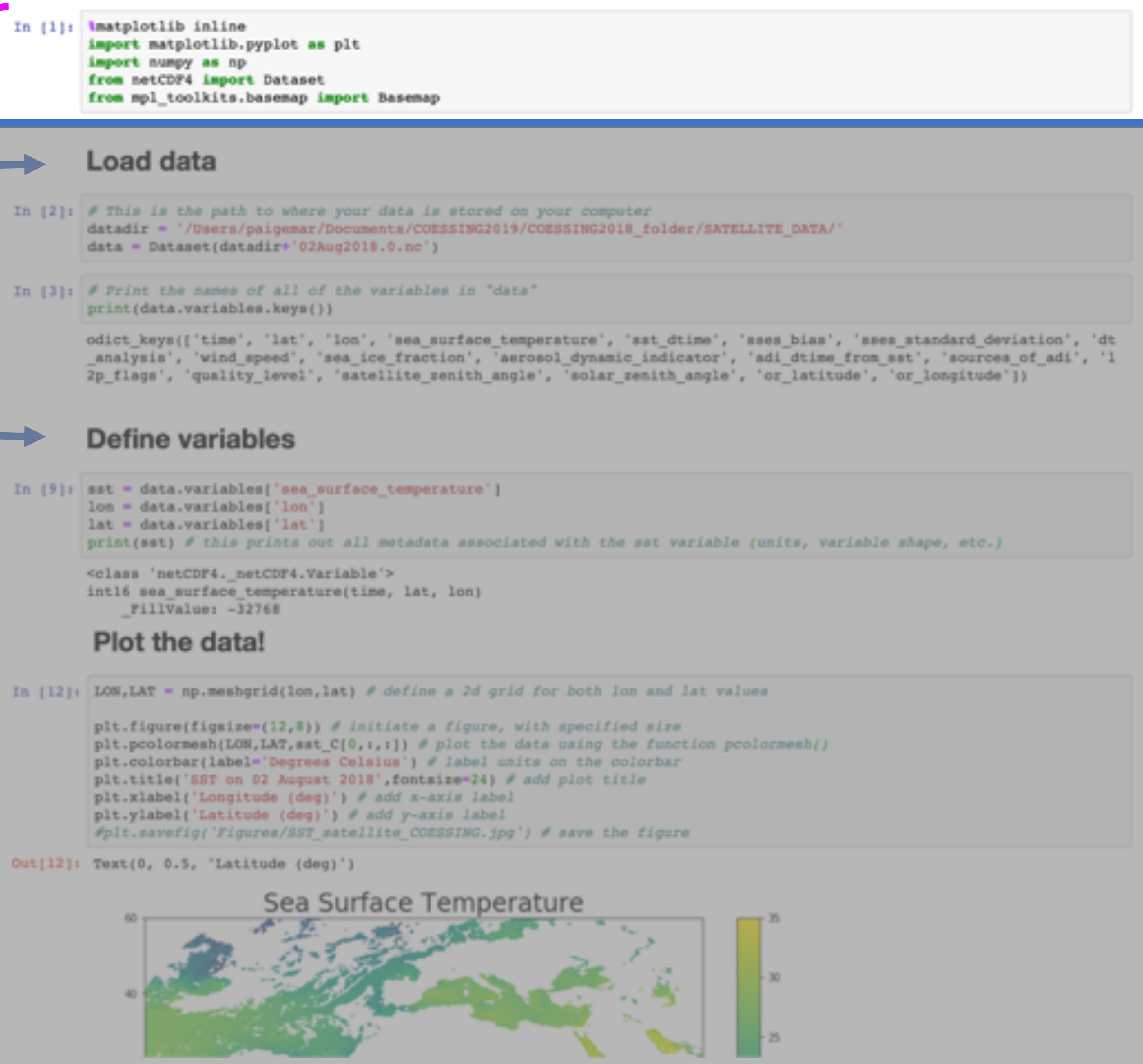
Comments

Load your data

Data analysis

Visualize your data

Main code



```
In [1]:  %matplotlib inline
         import matplotlib.pyplot as plt
         import numpy as np
         from netCDF4 import Dataset
         from mpl_toolkits.basemap import Basemap
```

Load data

```
In [2]:  # This is the path to where your data is stored on your computer
         datadir = '/Users/paigemar/Documents/COESSING2019/COESSING2018_folder/SATELLITE_DATA/'
         data = Dataset(datadir+'02Aug2018.0.nc')

In [3]:  # Print the names of all of the variables in "data"
         print(data.variables.keys())

         odict_keys(['time', 'lat', 'lon', 'sea_surface_temperature', 'sst_dtime', 'sses_bias', 'sses_standard_deviation', 'dt
         _analysis', 'wind_speed', 'sea_ice_fraction', 'aerosol_dynamic_indicator', 'adi_dtime_from_sst', 'sources_of_adi', 'l
         2p_flags', 'quality_level', 'satellite_zenith_angle', 'solar_zenith_angle', 'or_latitude', 'or_longitude'])
```

Define variables

```
In [9]:  sst = data.variables['sea_surface_temperature']
         lon = data.variables['lon']
         lat = data.variables['lat']
         print(sst) # this prints out all metadata associated with the sst variable (units, variable shape, etc.)

         <class 'netCDF4._netCDF4.Variable'>
         int16 sea_surface_temperature(time, lat, lon)
             _FillValue: -32768
```

Plot the data!

```
In [12]:  LON,LAT = np.meshgrid(lon,lat) # define a 2d grid for both lon and lat values

          plt.figure(figsize=(12,8)) # initiate a figure, with specified size
          plt.pcolormesh(LON,LAT,sst_C[0,:,:]) # plot the data using the function pcolormesh()
          plt.colorbar(label='Degrees Celsius') # label units on the colorbar
          plt.title('SST on 02 August 2018',fontsize=24) # add plot title
          plt.xlabel('Longitude (deg)') # add x-axis label
          plt.ylabel('Latitude (deg)') # add y-axis label
          #plt.savefig('Figures/SST_satellite_COESSING.jpg') # save the figure

Out[12]:  Text(0, 0.5, 'Latitude (deg)')
```

Common Python libraries for science

Common Python libraries for science

- **NumPy** – numeric library with lots of mathematical functions (e.g. average, trig functions, etc.), and also has array formatting that is very convenient

Common Python libraries for science

- **NumPy** – numeric library with lots of mathematical functions (e.g. average, trig functions, etc.), and also has array formatting that is very convenient
- **Matplotlib** – plotting library for Python

Common Python libraries for science

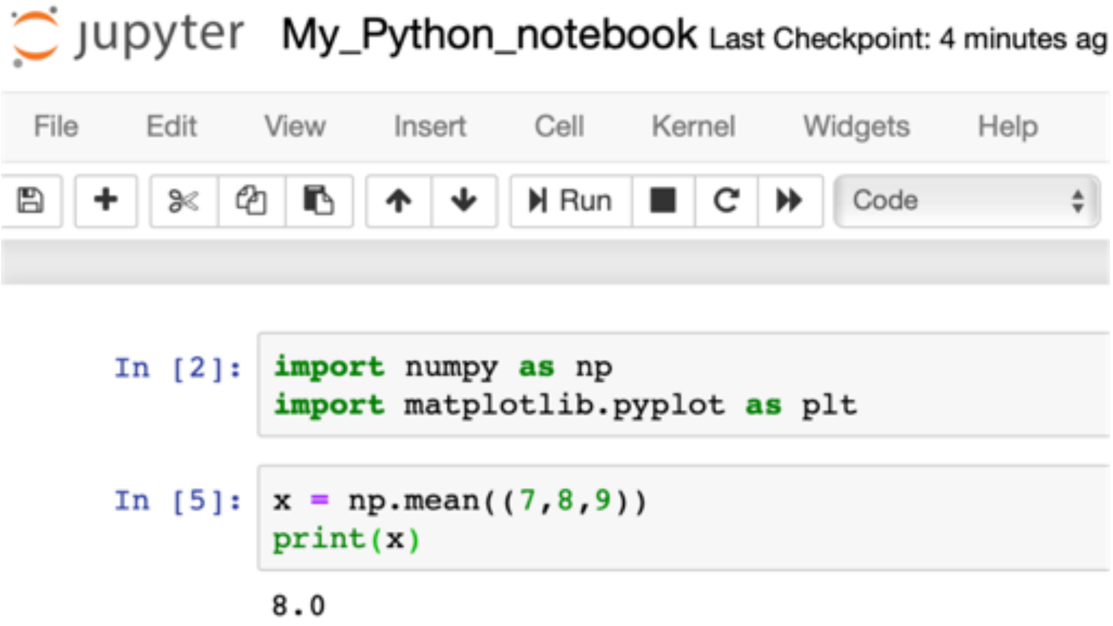
- **NumPy** – numeric library with lots of mathematical functions (e.g. average, trig functions, etc.), and also has array formatting that is very convenient
- **Matplotlib** – plotting library for Python
- **Pandas** – efficient and easy-to-use data structures and other data analysis tools

Common Python libraries for science

- **NumPy** – numeric library with lots of mathematical functions (e.g. average, trig functions, etc.), and also has array formatting that is very convenient
- **Matplotlib** – plotting library for Python
- **Pandas** – efficient and easy-to-use data structures and other data analysis tools
- We need to import libraries each time we start a new script

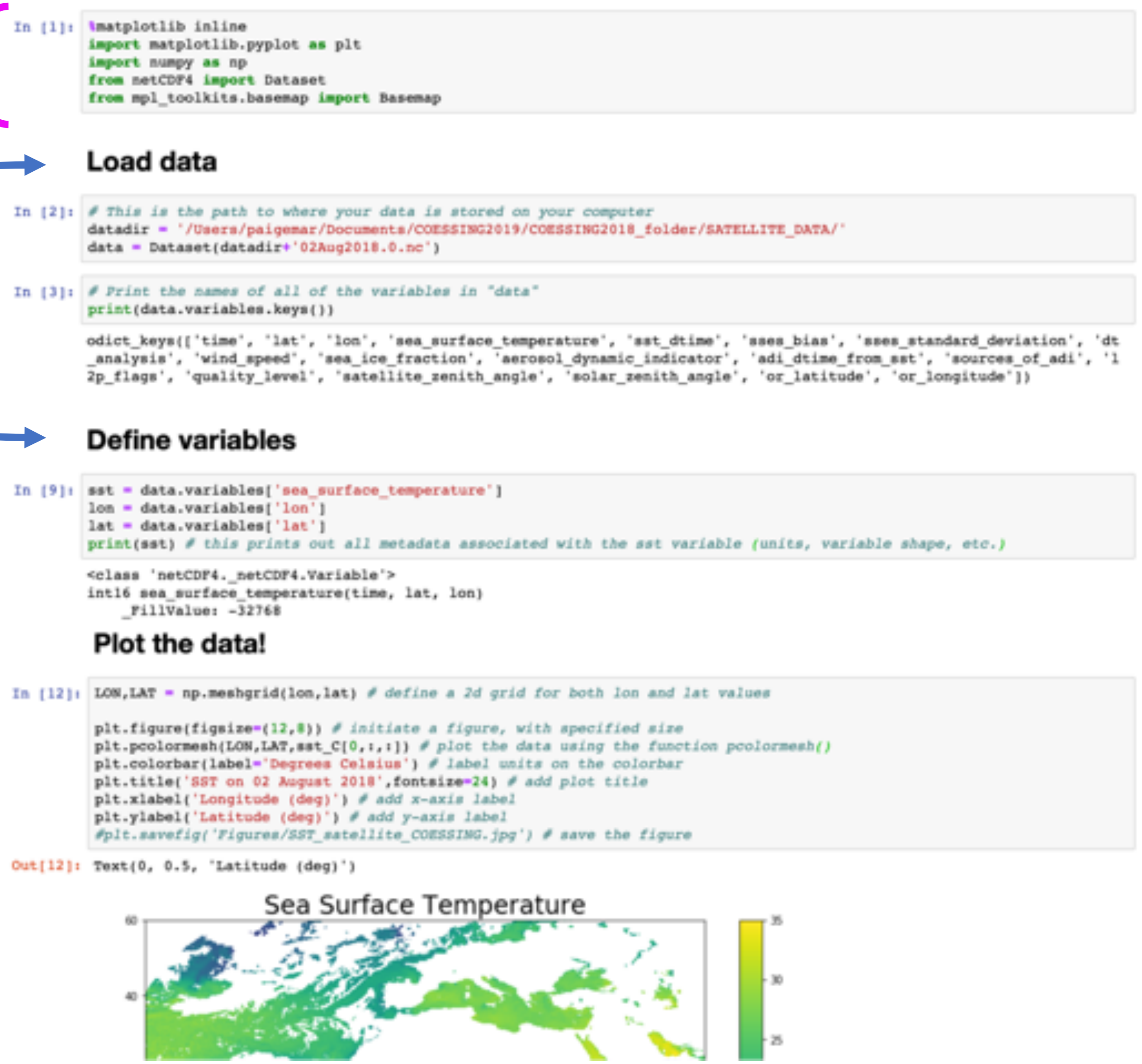
Common Python libraries for science

- **NumPy** – numeric library with lots of mathematical functions (e.g. average, trig functions, etc.), and also has array formatting that is very convenient
- **Matplotlib** – plotting library for Python
- **Pandas** – efficient and easy-to-use data structures and other data analysis tools
- We need to import libraries each time we start a new script
- We often import libraries as a shorter name to keep typing to minimum, but this helps organize so that we know which functions are from which libraries



```
In [2]: import numpy as np
        import matplotlib.pyplot as plt

In [5]: x = np.mean((7,8,9))
        print(x)

        8.0
```

How do I write a Python script in Jupyter notebook?



Import libraries

```
In [1]: %matplotlib inline
        import matplotlib.pyplot as plt
        import numpy as np
        from netCDF4 import Dataset
        from mpl_toolkits.basemap import Basemap
```

Comments

Main code

Load data

Load your data

```
In [2]: # This is the path to where your data is stored on your computer
        datadir = '/Users/paigemar/Documents/COESSING2019/COESSING2018_folder/SATELLITE_DATA/'
        data = Dataset(datadir+'02Aug2018.0.nc')

In [3]: # Print the names of all of the variables in "data"
        print(data.variables.keys())

        odict_keys(['time', 'lat', 'lon', 'sea_surface_temperature', 'sst_dtime', 'sses_bias', 'sses_standard_deviation', 'dt
        _analysis', 'wind_speed', 'sea_ice_fraction', 'aerosol_dynamic_indicator', 'adi_dtime_from_sst', 'sources_of_adi', 'l
        2p_flags', 'quality_level', 'satellite_zenith_angle', 'solar_zenith_angle', 'or_latitude', 'or_longitude'])
```

Define variables

Data analysis

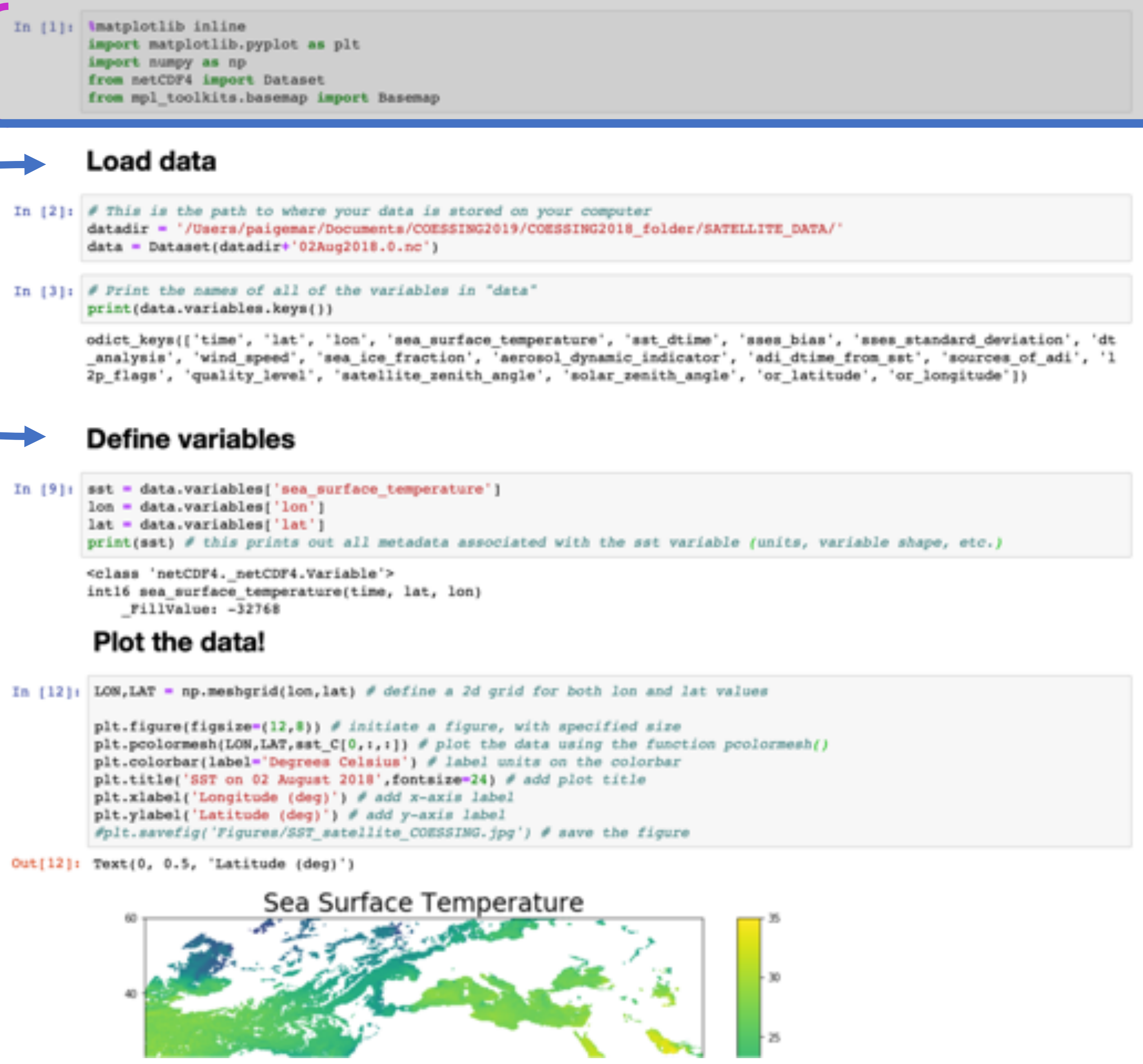
```
In [9]: sst = data.variables['sea_surface_temperature']
        lon = data.variables['lon']
        lat = data.variables['lat']
        print(sst) # this prints out all metadata associated with the sst variable (units, variable shape, etc.)

        <class 'netCDF4._netCDF4.Variable'>
        int16 sea_surface_temperature(time, lat, lon)
            _FillValue: -32768
```

Plot the data!

Visualize your data

```
In [12]: LON,LAT = np.meshgrid(lon,lat) # define a 2d grid for both lon and lat values

         plt.figure(figsize=(12,8)) # initiate a figure, with specified size
         plt.pcolormesh(LON,LAT,sst_C[0,:,:]) # plot the data using the function pcolormesh()
         plt.colorbar(label='Degrees Celsius') # label units on the colorbar
         plt.title('SST on 02 August 2018',fontsize=24) # add plot title
         plt.xlabel('Longitude (deg)') # add x-axis label
         plt.ylabel('Latitude (deg)') # add y-axis label
         #plt.savefig('Figures/SST_satellite_COESSING.jpg') # save the figure

Out[12]: Text(0, 0.5, 'Latitude (deg)')
```

How do I write a Python script in Jupyter notebook?



Import libraries

```
In [1]: %matplotlib inline
        import matplotlib.pyplot as plt
        import numpy as np
        from netCDF4 import Dataset
        from mpl_toolkits.basemap import Basemap
```

Comments

Load data

Load your data

```
In [2]: # This is the path to where your data is stored on your computer
        datadir = '/Users/paigemar/Documents/COESSING2019/COESSING2018_folder/SATELLITE_DATA/'
        data = Dataset(datadir+'02Aug2018.0.nc')

In [3]: # Print the names of all of the variables in "data"
        print(data.variables.keys())

        odict_keys(['time', 'lat', 'lon', 'sea_surface_temperature', 'sst_dtime', 'sses_bias', 'sses_standard_deviation', 'dt
        _analysis', 'wind_speed', 'sea_ice_fraction', 'aerosol_dynamic_indicator', 'adi_dtime_from_sst', 'sources_of_adi', 'l
        2p_flags', 'quality_level', 'satellite_zenith_angle', 'solar_zenith_angle', 'or_latitude', 'or_longitude'])
```

Define variables

Data analysis

```
In [9]: sst = data.variables['sea_surface_temperature']
        lon = data.variables['lon']
        lat = data.variables['lat']
        print(sst) # this prints out all metadata associated with the sst variable (units, variable shape, etc.)

        <class 'netCDF4._netCDF4.Variable'>
        int16 sea_surface_temperature(time, lat, lon)
            _FillValue: -32768
```

Plot the data!

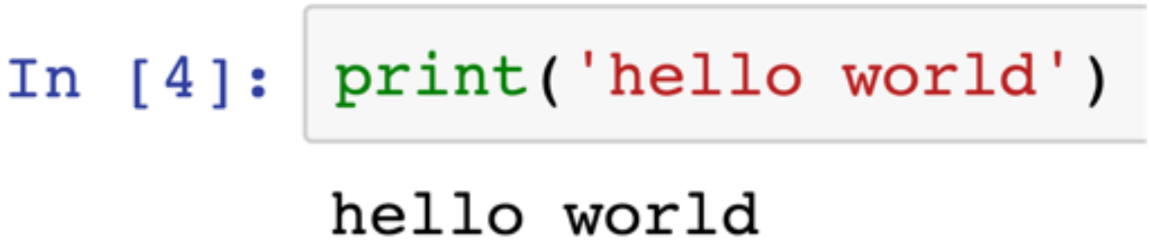
Visualize your data

```
In [12]: LON,LAT = np.meshgrid(lon,lat) # define a 2d grid for both lon and lat values

         plt.figure(figsize=(12,8)) # initiate a figure, with specified size
         plt.pcolormesh(LON,LAT,sst_C[0,:,:]) # plot the data using the function pcolormesh()
         plt.colorbar(label='Degrees Celsius') # label units on the colorbar
         plt.title('SST on 02 August 2018',fontsize=24) # add plot title
         plt.xlabel('Longitude (deg)') # add x-axis label
         plt.ylabel('Latitude (deg)') # add y-axis label
         #plt.savefig('Figures/SST_satellite_COESSING.jpg') # save the figure

Out[12]: Text(0, 0.5, 'Latitude (deg)')
```

Main code

Common actions in Python

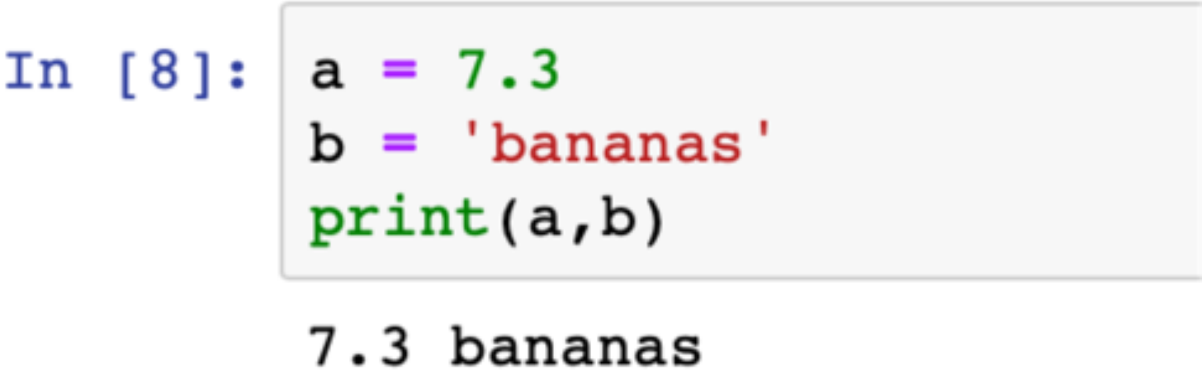
- **Print statements**



```
In [4]: print('hello world')

        hello world
```

Common actions in Python

- Print statements
- **Assign variables**

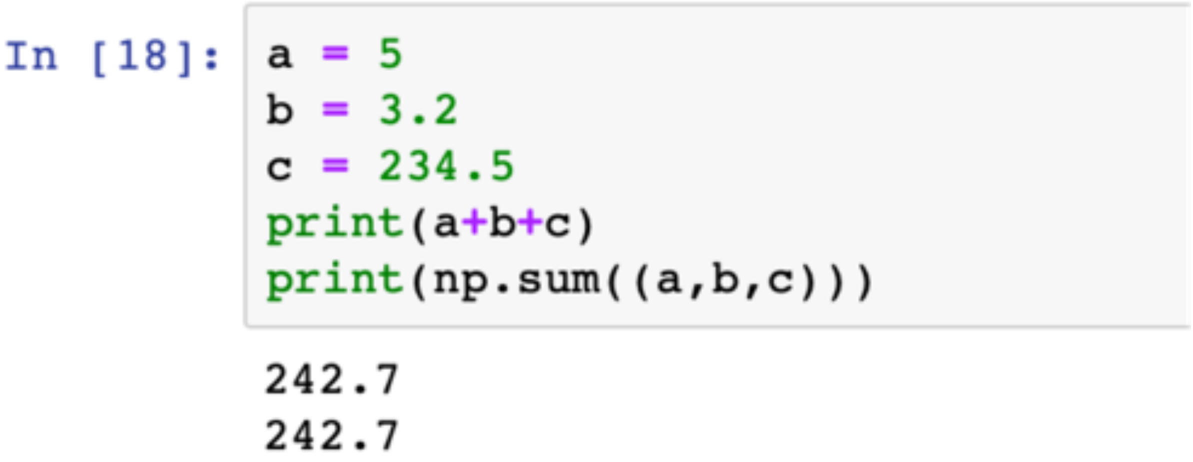


```
In [8]: a = 7.3
        b = 'bananas'
        print(a,b)
```

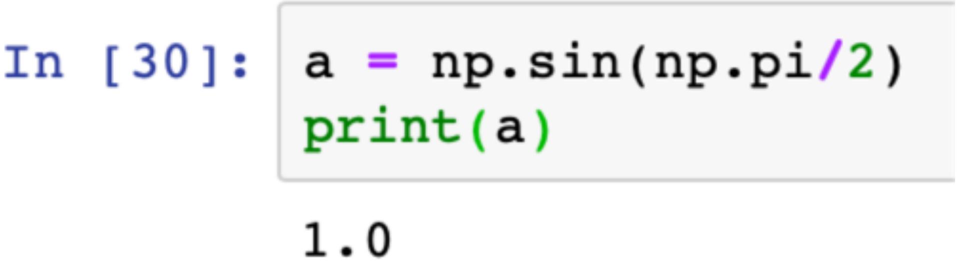
```
7.3 bananas
```

Common actions in Python

- Print statements
- Assign variables
- **Do math**



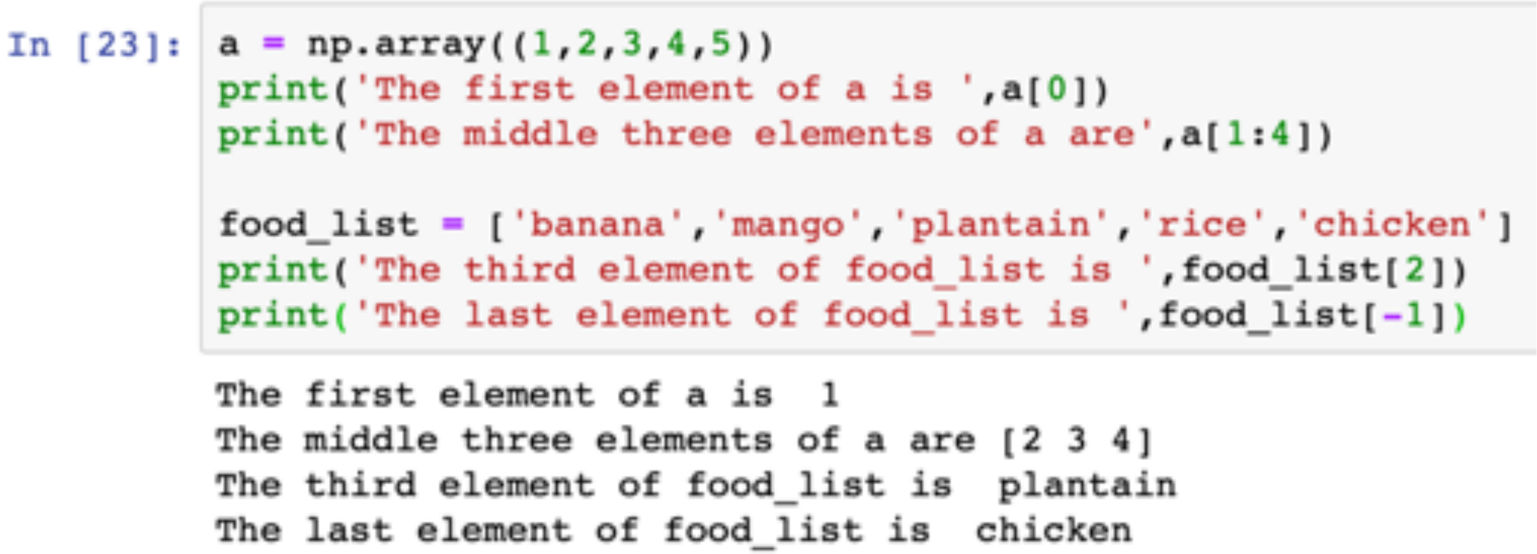
```
In [18]: a = 5
         b = 3.2
         c = 234.5
         print(a+b+c)
         print(np.sum((a,b,c)))

         242.7
         242.7
```



```
In [30]: a = np.sin(np.pi/2)
         print(a)

         1.0
```

Common actions in Python

- Print statements
- Assign variables
- Do math
- **Use arrays and lists**
    - **Accessing elements**

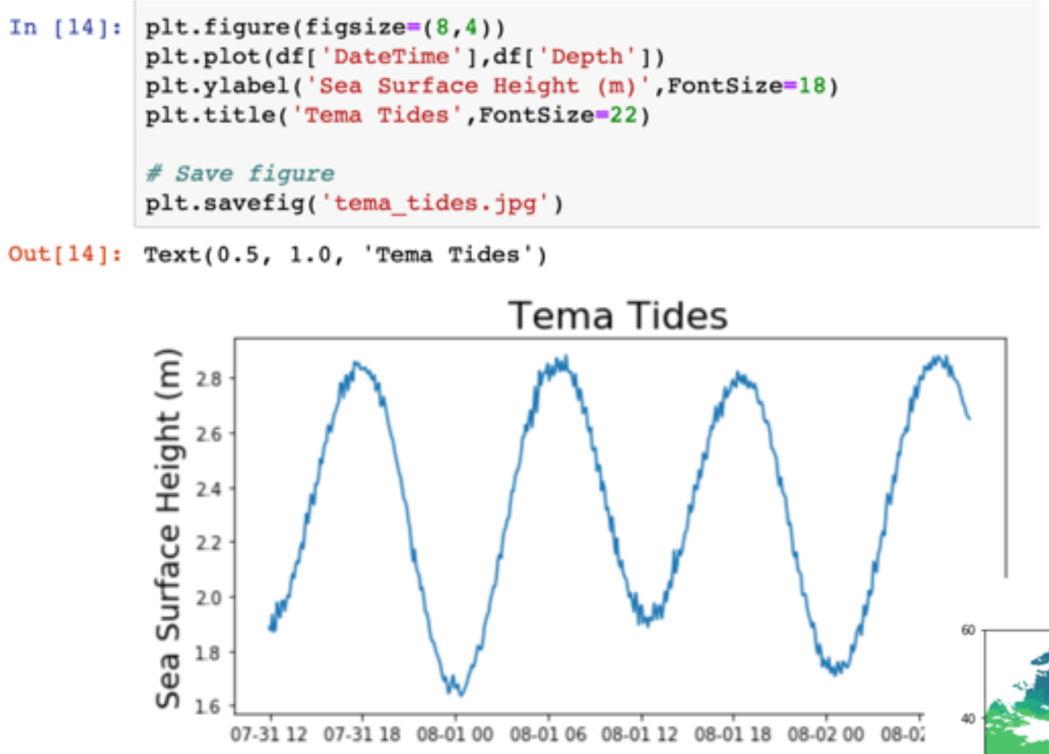


```
a = np.array((1,2,3,4,5))
print('The first element of a is ',a[0])
print('The middle three elements of a are',a[1:4])

food_list = ['banana','mango','plantain','rice','chicken']
print('The third element of food_list is ',food_list[2])
print('The last element of food_list is ',food_list[-1])
```

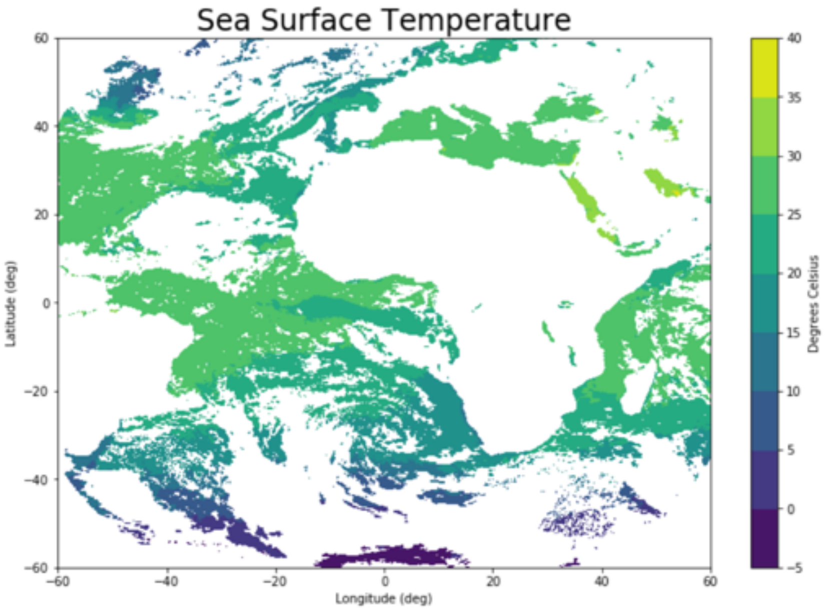
```
The first element of a is  1
The middle three elements of a are [2 3 4]
The third element of food_list is  plantain
The last element of food_list is  chicken
```

Common actions in Python

- Print statements
- Assign variables
- Do math
- Use arrays and lists
  - Accessing elements
- **Plot data**

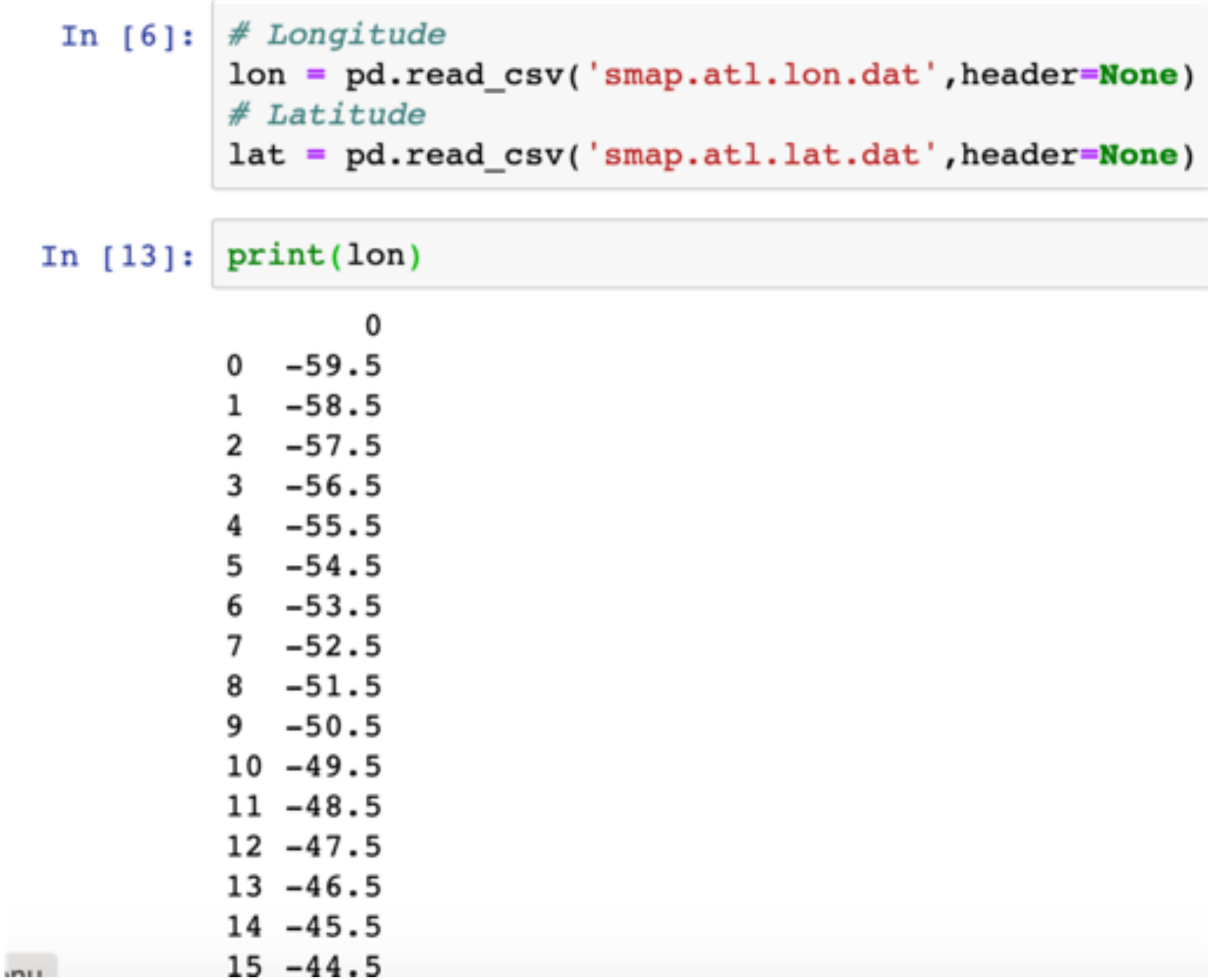


```
In [14]: plt.figure(figsize=(8,4))
         plt.plot(df['DateTime'],df['Depth'])
         plt.ylabel('Sea Surface Height (m)',FontSize=18)
         plt.title('Tema Tides',FontSize=22)

         # Save figure
         plt.savefig('tema_tides.jpg')

Out[14]: Text(0.5, 1.0, 'Tema Tides')
```



Common actions in Python

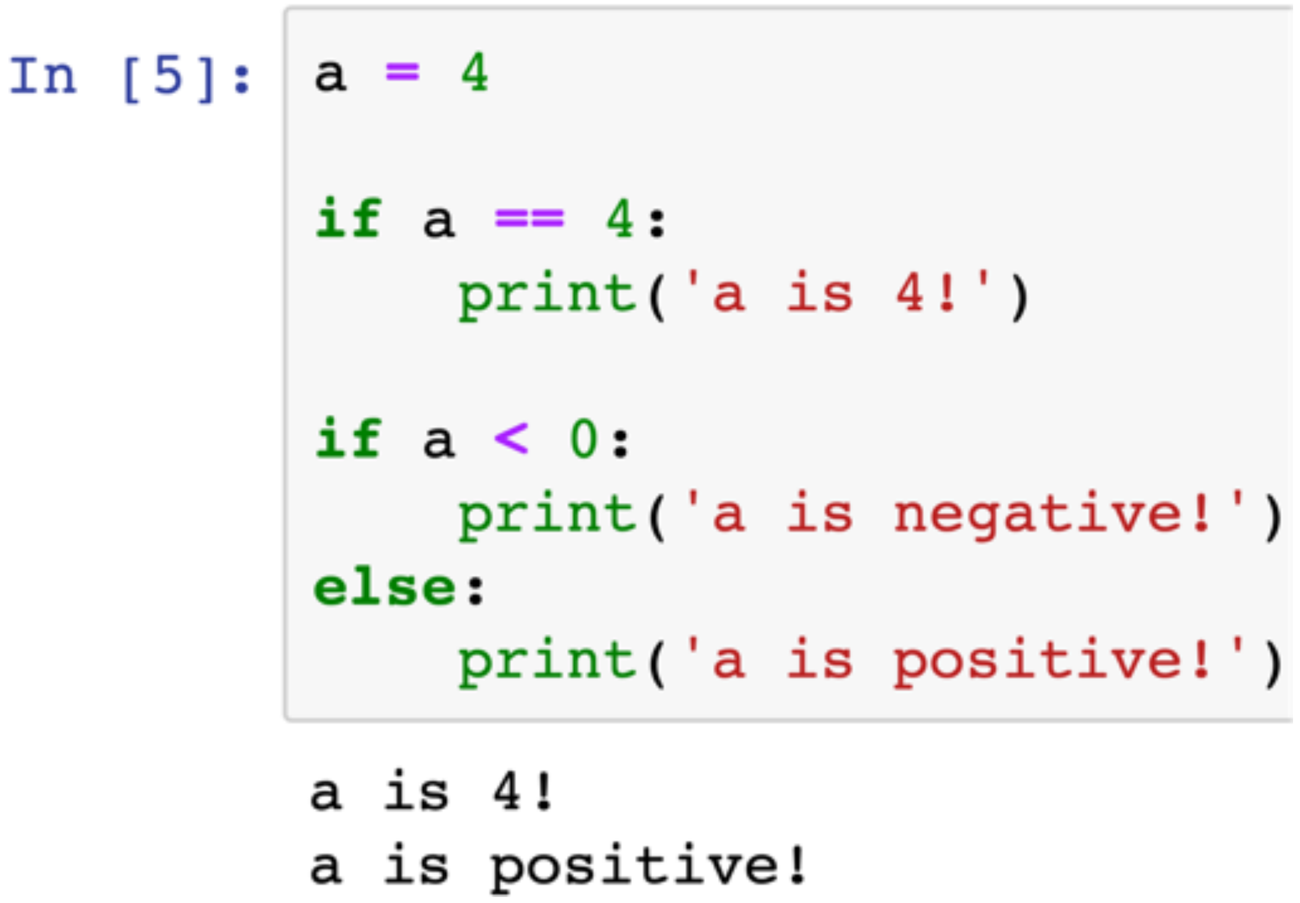
- Print statements
- Assign variables
- Do math
- Use arrays and lists
  - Accessing elements
- Plot data
- **Load data – we assign a name to the dataset (e.g. ‘lon’) - same as assigning a variable!**



```
In [6]:  # Longitude
         lon = pd.read_csv('smap.atl.lon.dat',header=None)
         # Latitude
         lat = pd.read_csv('smap.atl.lat.dat',header=None)

In [13]: print(lon)

                0
         0   -59.5
         1   -58.5
         2   -57.5
         3   -56.5
         4   -55.5
         5   -54.5
         6   -53.5
         7   -52.5
         8   -51.5
         9   -50.5
         10  -49.5
         11  -48.5
         12  -47.5
         13  -46.5
         14  -45.5
         15  -44.5
```

Common actions in Python

- Print statements
- Assign variables
- Do math
- Use arrays and lists
    - Accessing elements
- Plot data
- Load data – we assign a name to the dataset (e.g. lon) - same as assigning a variable!
- **if statements** and for loops



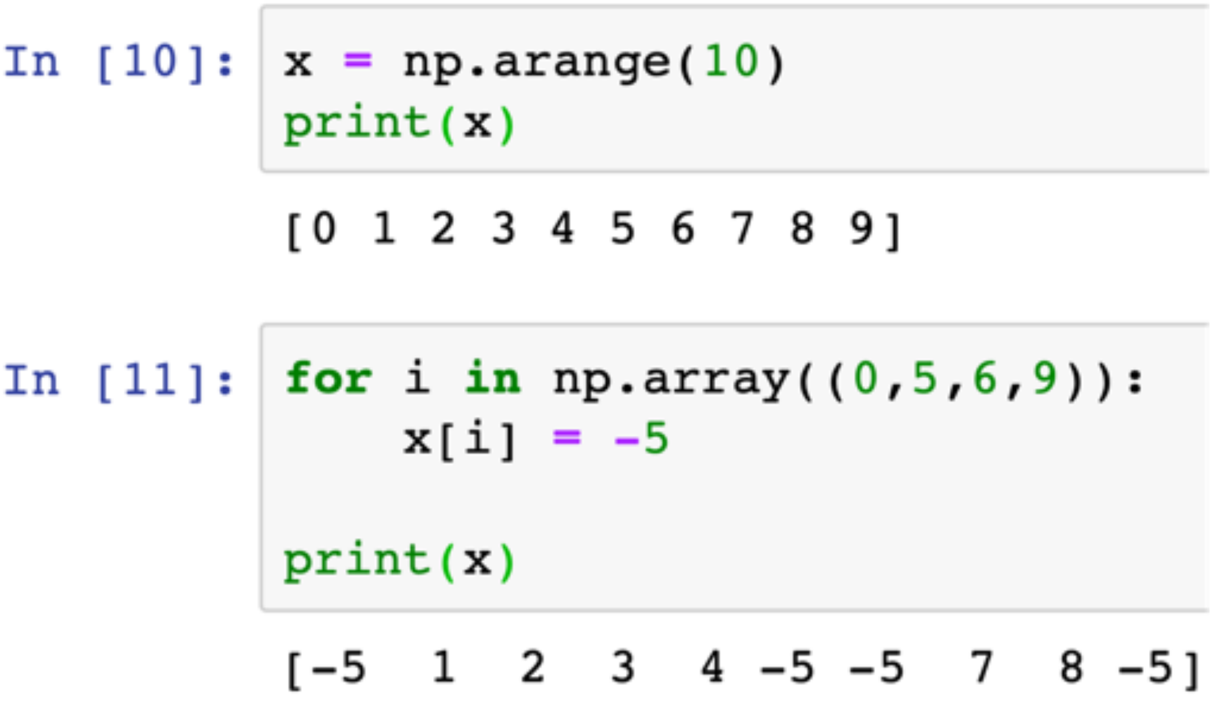
In [5]:

```
a = 4

if a == 4:
    print('a is 4!')

if a < 0:
    print('a is negative!')
else:
    print('a is positive!')
```

```
a is 4!
a is positive!
```

Common actions in Python

- Print statements
- Assign variables
- Do math
- Use arrays and lists
    - Accessing elements
- Plot data
- Load data – we assign a name to the dataset (e.g. lon) - same as assigning a variable!
- if statements and **for loops**

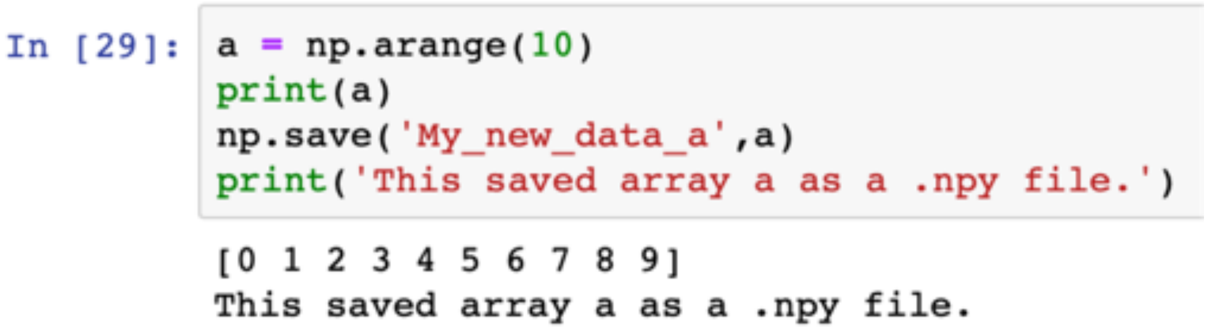


```
In [10]: x = np.arange(10)
         print(x)

[0 1 2 3 4 5 6 7 8 9]
```

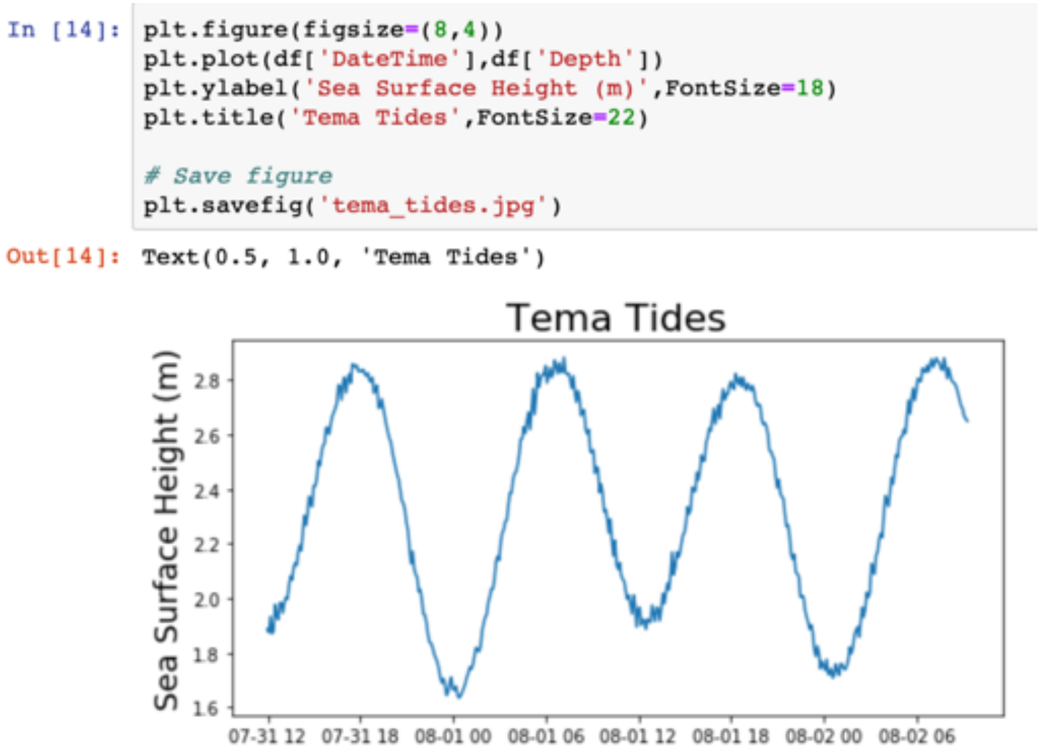
```
In [11]: for i in np.array((0,5,6,9)):
             x[i] = -5

         print(x)

[-5  1  2  3  4 -5 -5  7  8 -5]
```

Common actions in Python

- Print statements
- Assign variables
- Do math
- Use arrays and lists
  - Accessing elements
- Plot data
- Load data – we assign a name to the dataset (e.g. lon) - same as assigning a variable!
- if statements and for loops
- **Save new data or figure**



```
In [29]: a = np.arange(10)
         print(a)
         np.save('My_new_data_a',a)
         print('This saved array a as a .npy file.')

         [0 1 2 3 4 5 6 7 8 9]
         This saved array a as a .npy file.
```



```
In [14]: plt.figure(figsize=(8,4))
         plt.plot(df['DateTime'],df['Depth'])
         plt.ylabel('Sea Surface Height (m)',FontSize=18)
         plt.title('Tema Tides',FontSize=22)

         # Save figure
         plt.savefig('tema_tides.jpg')

Out[14]: Text(0.5, 1.0, 'Tema Tides')
```

There are other computer languages. Why Python?

There are other computer languages. Why Python?

- Relatively easy to learn

There are other computer languages. Why Python?

- Relatively easy to learn
- Free and open source

There are other computer languages. Why Python?

- Relatively easy to learn
- Free and open source
- Syntax is clean and easy to read

There are other computer languages. Why Python?

- Relatively easy to learn
- Free and open source
- Syntax is clean and easy to read
- Computationally efficient

There are other computer languages. Why Python?

- Relatively easy to learn
- Free and open source
- Syntax is clean and easy to read
- Computationally efficient
- Can be used for many different applications

There are other computer languages. Why Python?

- Relatively easy to learn
- Free and open source
- Syntax is clean and easy to read
- Computationally efficient
- Can be used for many different applications
- Other languages/software: Matlab, C, C++, Fortran, R, etc.

Download and install instructions

Download and install instructions

ANACONDA

- Download/install Python via Anaconda

Download and install instructions

- Download/install Python via Anaconda
  - Instructions for download are on COESSING website (”coessing.org”) under the “Resources” tab

Download and install instructions

- Download/install Python via Anaconda
  - Instructions for download are on COESSING website (”coessing.org”) under the “Resources” tab
- Introductory Python and Jupyter lessons also under the “Resources” tab

Download and install instructions

ANACONDA.

- Download/install Python via Anaconda
  - Instructions for download are on COESSING website (”coessing.org”) under the “Resources” tab
- Introductory Python and Jupyter lessons also under the “Resources” tab
- Tomorrow afternoon, Python lab in the RMU computer lab!

Let’s open Jupyter notebook!