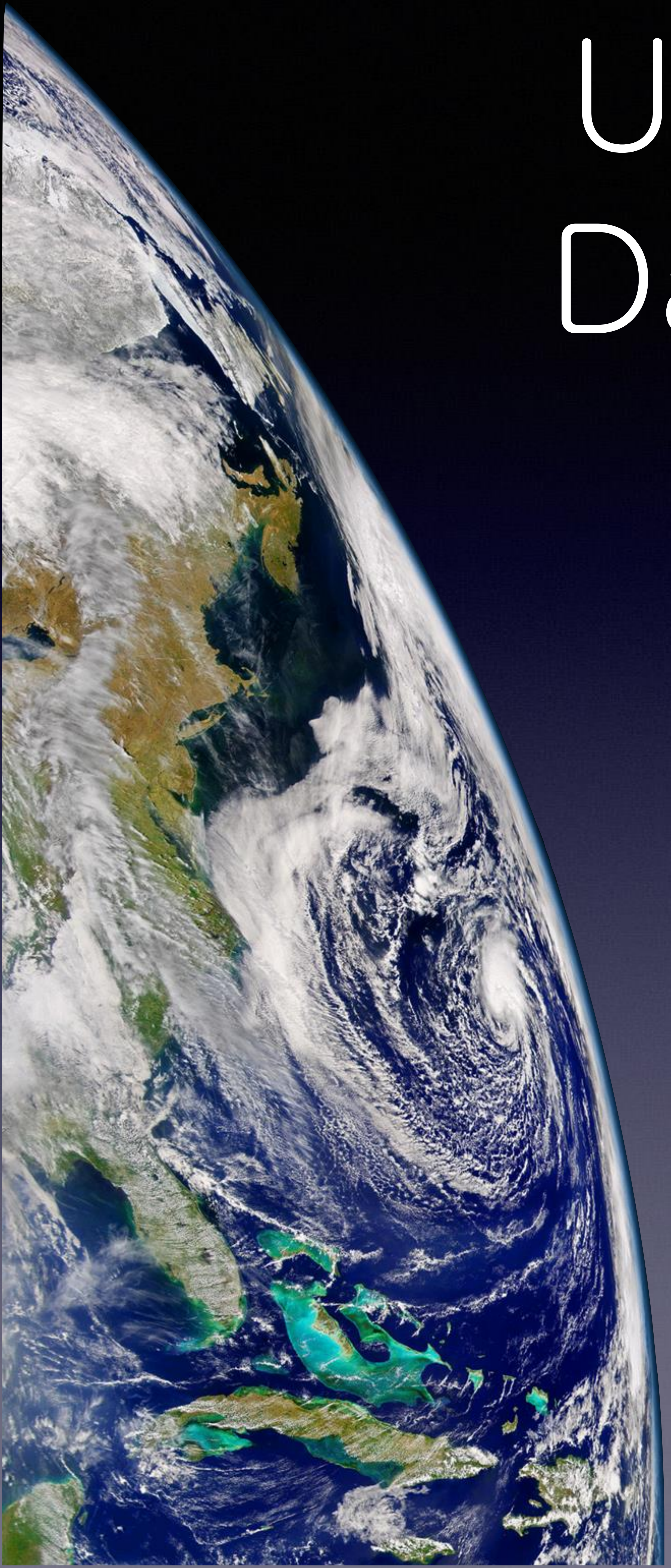
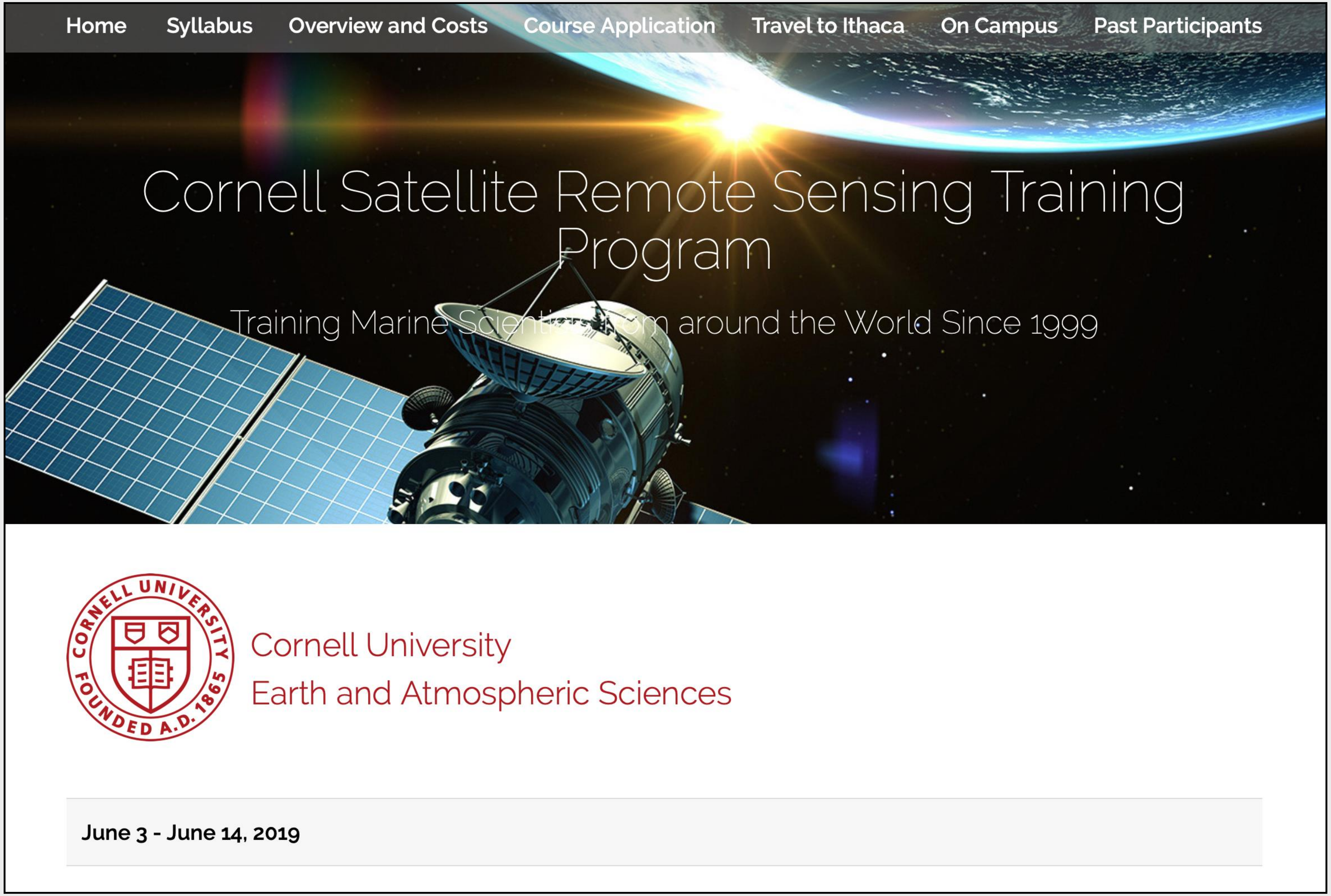


Using Python and SeaDAS for Data Processing, Visualization and Analysis

Breakout Workshop on Open Source Computing Tools
IOCS Meeting, Busan South Korea, 9 April 2019

Bruce Monger, bcm3@cornell.edu
Department of Earth and Atmospheric Sciences
Cornell University, Ithaca NY 14853 USA



Cornell Satellite Remote Sensing Training Program

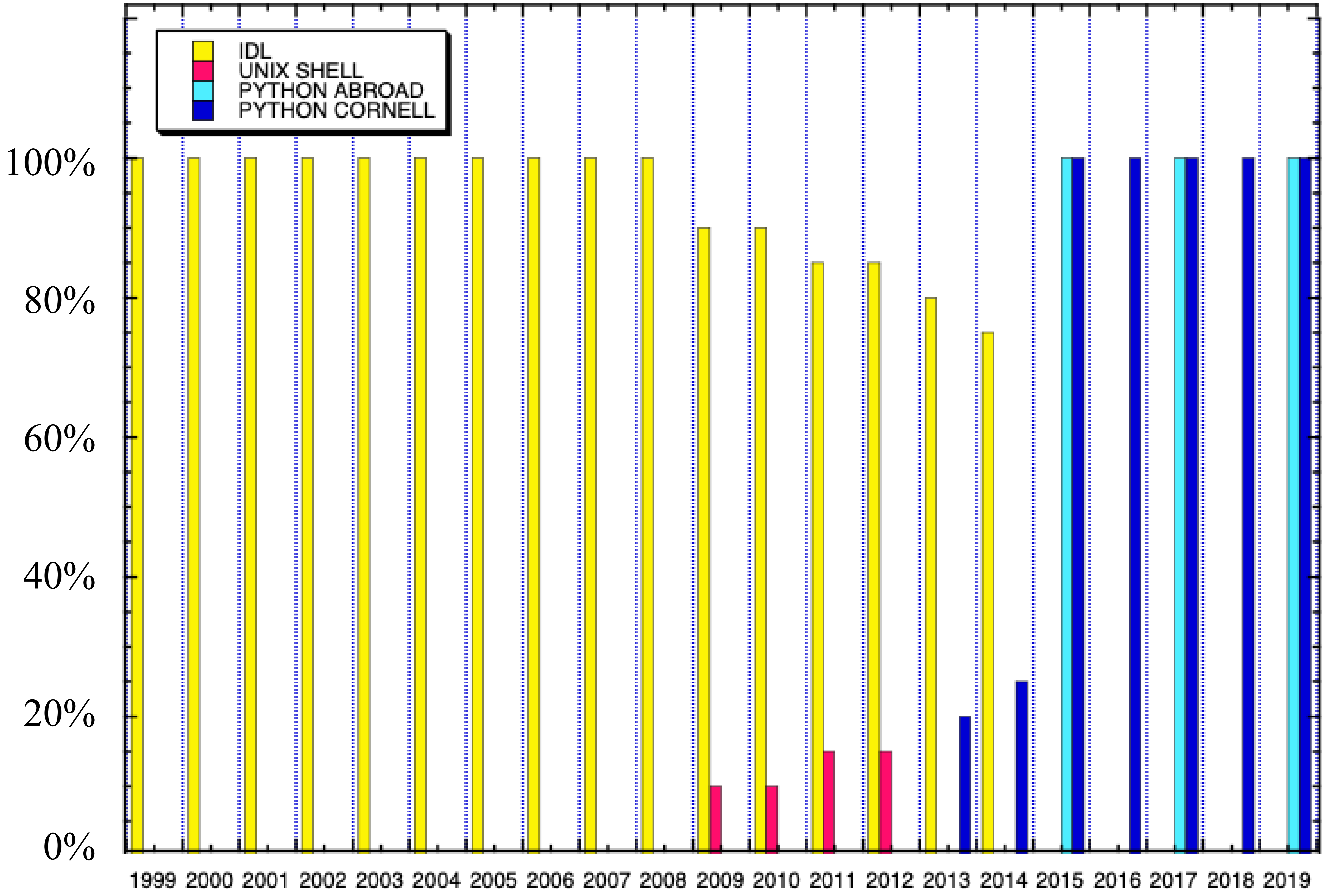
Training Marine Scientists from around the World Since 1999

Cornell University
Earth and Atmospheric Sciences

June 3 - June 14, 2019

http://oceanography.eas.cornell.edu/satellite/

Timeline of Transition From IDL to Python



1999–2008

- IDL Only

2009–2013

- Introduce some Unix shell script examples

2013

- Introduce some Python examples at Cornell

2015–

- 100% Python Only Abroad
- 100% Python Only Cornell

Why I Moved From IDL to Python…

1. Python is open source software
2. Installation of python packages became much easier in recent years
3. Teaching the Cornell Satellite Training Course
    - International participants without an IDL license back home
    - Participants from highly bureaucratic US government labs that would not authorize the purchase an IDL license
    - US graduate students coming from small labs with limited resources.
4. Teaching remote sensing abroad where the availability of

What is Python?

1. Python is a high-level programming language.
   - Open-Source
   - Useful for rapid application development
   - Useful as a scripting language to connect existing components

2. Basic Python found on most computers has a limited set of features. However, individuals and organizations have created an extensive set of additional functional capabilities that can be installed and imported to create a powerful data analysis tool. (e.g., numpy, scipy, matplotlib and hdf4, netcdf libraries and utilities).

Two Approaches to Using Python

Interactive Mode and Text File Interpreter Mode

Interactive Mode

1. Open a Unix Terminal Window and then type: python
2. Start typing out python commands at the python prompt: >>>
3. This approach is really great for quickly checking on how a new python function works

Text File Interpreter Mode

4. Open a text file with a text editor
5. Write lines of python code into the open text window
6. Save the text file and run the python code contained in the text file typing the following in a Unix Terminal Window: python textfile.py

Some Simple Examples of Using Python with Satellite Data

Displaying Image Data

```
#!/usr/bin/env python
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.colors


# read in previously generated satellite data file
fname = '/rsclass/data/tutorial_data/S1998148172338_chlor_a.f999x999'
f = open(fname)
data = np.fromfile(f, dtype=np.float32)   # assumes data were previously written out to the
f.close()                                  # hard drive as 32bit floating point numbers

data = data.reshape([999, 999])            # you would have to know a priori that the data
                                           # had previously been written out as a 999x999 array

# define color scale…
mycmap = plt.get_cmap('spectral')          # load color rainbow palette
mycmap.set_bad('k')                        # set NaN values to display as black

# create and display the figure to your monitor with data in log scale…
plt.figure(1)
plt.imshow(data, cmap=mycmap, vmin= 0.01, vmax=20.0, norm=matplotlib.colors.LogNorm())
plt.colorbar()
plt.show()
```

Plotting Coastlines

```
#!/usr/bin/env python
import numpy as np;  import matplotlib.pyplot as plt;  import matplotlib.colors
from mpl_toolkits.basemap import Basemap

 # read in satellite file (with a known projection and lat/lon boundaries)
fname = '/rsclass/data/tutorial_data/S1998148172338_chlor_a.f999x999'
f = open(fname);  mapped_data = np.fromfile(f, dtype=np.float32); f.close(); mapped_data = mapped_data.reshape([999, 999])

# set the map projection space (must know this a priori)
north= 46.0;  west= -72.0; south= 37.0; east= -63.0
m = Basemap(projection='cyl', llcrnrlon= west, llcrnrlat= south,  urcrnrlon= east, urcrnrlat=north, resolution = 'h')

# set the color palette
mycmap= plt.get_cmap('spectral'); mycmap.set_bad('k')
# flip array upside down (for mapping only)...
mapped_data = np.flipud(mapped_data)

# display the satellite image in the map projection window
m.imshow(mapped_data,cmap=mycmap, vmin= 0.01, vmax=20.0, norm=matplotlib.colors.LogNorm())

# draw coastline, lat/lon grid and axis labels
m.drawcoastlines();   m.fillcontinents(color='grey', lake_color='white')
parallels = np.arange(south, north,2.);   m.drawparallels(parallels, labels=[True,False,False,False])      #Note: labels = [left,right,top,bottom]
meridians = np.arange(west, east,2.);    m.drawmeridians(meridians, labels=[False,False,True,False])
m.colorbar()

# save the mapped images as a png file
plt.savefig('/Users/bmonger/Desktop/test.png', bbox_inches='tight')

# show the plot to the monitor and then and then close (i.e., clear from memory) all plotting settings (useful for loops)
plt.show(); plt.close()
```

Python + SeaDAS

SeaDAS

SeaDAS is made up of executable binary functions/procedures (e.g., l2bin, l3bin, l2gen, l3mapgen), python scripts (e.g., modis_GEO.py) and data libraries (e.g., calibration tables)

Binaries and Libraries

mainly used for data processing (e.g. l2gen)
made from C and/or Fortran code
SeaDAS also includes HDF-NetCDF binaries and libraries
source code is available for all binaries (if you really want them)

Python Scripts

- mainly used as "wrappers" for calling other programs
- but there are also stand-alone utility scripts

Type the command in a terminal window to see the syntax for running the

A Simple Example of Batch Processing Ocean Color Data…

A Simple Example of a Batch Processing Aqua Level-1 to Level-3 Data

```
#! /usr/bin/env python
import matplotlib.pyplot as plt;  import matplotlib.colors;  from mpl_toolkits.basemap import
Basemap
import numpy as np;  import glob
from subprocess import call;  import sys, os
from korea_netcdf_utilities import *
L1a_list= glob.glob(‘/level-1a-data-directory/*L1A*.hdf')       # read in a list of Level-1A files

# loop through and sequentially process Level-1 to Level-3 and output a PNG image of the Level-3 data
for L1a_name in L1a_list:

    call('modis_GEO.py -v -o ' + fname_geo + ' ' + L1a_name, shell=True)

    call('modis_L1B.py -v -o ' + fname_l1b + ' ' + L1a_name + ' ' + fname_geo, shell=True)

    call(['l2gen',
          'ifile='          + fname_l1b,
          'ofile1='         + l2_fname,
          'l2prod1='        + prod_list,
          'geofile='        + fname_geo,
          'par='            + fname_ancil_list,
          'resolution='     + '1000'])
```

```
call(['l2bin',
        'infile='      + l2_fname,
        'ofile='       + l2b_fname,
        'l3bprod='  + product,
        'resolve='   + str(space_res).strip(),
        'flaguse='   + named_flags_2check])

call(['l3mapgen',
           'ifile=' + l2b_fname,
           'ofile=' + smi_fname,
           'prod=' + product,
           'deflate=' + '4',
           'scale_type=' + 'linear',
           'projection=' + 'platecarree',
           'resolution=' + space_res + 'km',
           'interp=' + 'nearest',
           'west='   + str(west).strip(),
           'east='    + str(east).strip(),
           'north='   + str(north).strip(),
           'south='   + str(south).strip()])

# read in netCDF mapped data (smi_fname above) and add coastline and lat/lon grid with matplotlib
# and save a png image of the data — as per the earlier example slide
…
```

Setting Up Python on Your Computer…

Setting Up Python on Your Computer…

1. Go to Anaconda Website and Download Python 2.7.
2. Open a Unix Terminal Window
3. Install and/or Update the following python packages by typing the following:
   - (a) conda update conda
   - (b) conda config --add channels conda-forge
   - (c) conda install netcdf4
   - (d) conda install -c cistools pyhdf
   - (e) conda install hdf5
   - (f) conda install basemap-data_hires
   - (g) conda install pyproj
   - (h) conda install pyresample
   - (i) conda update pip
   - (j) conda update --all

NOTE: You might need to downgrade numpy (using the command: install numpy=1.11.0), but only if when running the *pyresample* function and it calls numpy and causes a numpy error with *trying to index an array with a floating number.* The need for this will probably go away when pyresample is updated.

4. Modify the **pythonpath** environment variable in your ***.bashrc*** file to include the name of the directory (and any subdirectories) where you will save new python programs. For example…

```
PYHONPATH=$PYTHONPATH:~/python_programs:~/python_programs/utilities
```

Checking the Python + SeaDAS Output…

Going Forward…

Slides and the simple batch processing script and netCDF read function are posted online: http://www.geo.cornell.edu/iocs-meeting-2019

Note: You will also need to do the following…

1. Configure secure file transfer
2. Install SeaDAS *and* desired processing modules…
3. Register with EOSDIS: https://earthdata.nasa.gov