Using Python in climate data analysis
(and plotting using NCL)

Baird Langenbrunner
AOS 218
Oct. 24th, 2013
Outline

1. Python – what is it, why use it?
   a) Intro to Python and the SciPy “ecosystem” (SciPy library, NumPy, Matplotlib, etc.)
   b) UVCDAT, pyngl, pyclim, geopy - extra stuff
   c) NCL for plotting NetCDF files

2. Model uncertainties in climate change projections
   a) Intermodel disagreement/uncertainty on future projections; Knutti and Sedláček, 2012
   b) US West Coast precipitation change as a hotbed of model disagreement; storm tracks as possible cause
Python

- Programming language that began in the 1980s
- Conceived by Guido van Rossum, first official released was in 1989
- Philosophy emphasizes code readability
- Free, open source, runs on Windows, Linux/Unix, Mac
- Comes preinstalled on Mac OS X
The “Zen of Python”

Beautiful is better than ugly.
Explicit is better than implicit.
Simple is better than complex.
Complex is better than complicated.
Flat is better than nested.
Sparse is better than dense.
Readability counts.
Special cases aren't special enough to break the rules.
Although practicality beats purity.
Errors should never pass silently.
Unless explicitly silenced.
In the face of ambiguity, refuse the temptation to guess.
There should be one-- and preferably only one -- obvious way to do it.
Although that way may not be obvious at first unless you're Dutch.
Now is better than never.
Although never is often better than *right* now.
If the implementation is hard to explain, it's a bad idea.
If the implementation is easy to explain, it may be a good idea.
Namespaces are one honking great idea -- let's do more of those!
SciPy and the “SciPy Stack”

- SciPy = “scientific python”
- The stack is a collection of open-source software and scientific computing tools for Python
SciPy and the “SciPy Stack”

• the SciPy Stack:
  – (Python)
  – SciPy library
  – NumPy
  – Matplotlib
  – IPython, pandas, SymPy, nose

• Python – the basic language on which the rest of the stack is built
• SciPy library – the package that provides high-end statistics and linear algebra functions; uses NumPy
• NumPy – “numerical python” – the package that gives Python the capability of handling large arrays and performing quick/meaty calculations
• Matplotlib – “mathematical plotting library” – what you use if you want to plot figures within Python
Python/SciPy packages for data analysis

**Data input/output**
- NumPy input/output module ("import numpy.io")
  - "npz", text, binary
- SciPy input/output module ("import scipy.io")
  - Read/write/save MATLAB, IDL, and NetCDF files

**Analyzing data**
- Statistics module ("import scipy.stats")
  - Mean, standard deviation, correlation and covariance, etc.
- Linear algebra module ("import scipy.linalg")
  - SVD, CCA, EOF analysis
  - Other matrix decompositions
- Other SciPy modules for:
  - Fourier transforms, interpolation, optimization, integration, signal processing, etc.
Python/SciPy packages specific to NetCDF files

**NetCDF input/output**

- "UV-CDAT" (separate download)
- "scipy.io.netcdf" (SciPy input/output module [as on previous slide])
- "netcdf4-python" (separate download)
- "PyNIO" (separate download, from NCAR)

**NetCDF plotting**

- "UV-CDAT" (separate download)
- "PyNGL" (Python interface to the NCL Graphics Library, from NCAR)
- Matplotlib "basemap" toolkit (requires netcdf4-python)
UV-CDAT
“Ultrascale Visualization – Climate Data Analysis Tools”

- Package for Python that is great for data processing
- Developed by PCMDI at LLNL
UV-CDAT

“Ultrascale Visualization – Climate Data Analysis Tools”

- **CDAT** is the most important part of UV-CDAT

- Has 4 pieces:
  - **cdms** – climate data management system (file i/o, variables, types, etc.)
  - **cdutil** – climate data specific utilities (spatial/temporal averages, custom seasons, climatologies)
  - **genutil** – general utilities (some rudimentary statistics)
  - **vcs** – visualization and control system (graphics/plotting) [deprecated in UV-CDAT...]
Example of a Python script

calculate DJF end-of-century precipitation change, and compute the statistical significance of this

• Plot the end-of-century precipitation change for a given model (CCSM4)

• Plot the rainfall change patterns with “stippling” where they are statistically significant at the 95% confidence level (based on a t-test)
Example of a Python script
calculate DJF end-of-century precipitation change, and compute the statistical significance of this
Example of a Python script to calculate DJF end-of-century precipitation change, and compute the statistical significance of this change.

Import the necessary packages into the Python script.
Example of a Python script to calculate DJF end-of-century precipitation change, and compute the statistical significance of this change.

Create a function that saves data in NetCDF format.
Example of a Python script
calculate DJF end-of-century precipitation change,
and compute the statistical significance of this

define seasons (regular seasons are already defined)
insert the file names that you want to open (here, CCSM4 historical and RCP8.5 monthly data, regridded)
define your base and end-of-century time periods
Example of a Python script

calculate DJF end-of-century precipitation change, and compute the statistical significance of this

Create lists to store climatologies, etc.

Open up data, extract the relevant time periods, calculate climatologies, anomalies, t-tests on these anomalies
Example of a Python script to calculate DJF end-of-century precipitation change, and compute the statistical significance of this change. Save climatologies, anomalies, and results of t-test as separate NetCDF files.
NCL
now plot your NetCDF file using NCL

- NCL – NCAR Command Language
- Really versatile language; easy/free to download and install, great support community
- Created for “scientific data analysis and visualization”
- Easily reads in NetCDF “.nc” files, handles attributes like latitude and longitude well
NCL

now plot your NetCDF file using NCL

Load NCL code necessary for plotting

Define the file names for what you will be plotting

Open the anomalies, the t-test results...
NCL
now plot your NetCDF file using NCL

... and the base/end-of-century climatologies

Define what you want to save the plot as

Open a work station

Declare the resources for each component of the plot individually (anomalies, the base map, the significance test results, and the end-of-century and base period climatologies)
NCL

now plot your NetCDF file using NCL

continued: Declare the resources for each component of the plot individually (anomalies, the base map, the significance test results, and the end-of-century and base period climatologies)
NCL

Now plot your NetCDF file using NCL

Now plot each component (the map, the anomalies, the p-values, the base and end-of-century 4mm/day contours)

Lastly, overlay them all on top of the map itself
Downloading and more info

<table>
<thead>
<tr>
<th>Language/package</th>
<th>Website</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(has links to NumPy, Matplotlib, etc.)</td>
</tr>
<tr>
<td>NCL</td>
<td><a href="https://www.earthsystemgrid.org/dataset/ncl.html">https://www.earthsystemgrid.org/dataset/ncl.html</a></td>
</tr>
</tbody>
</table>

- Some of the most useful tips for learning these languages and packages are contained within the “tutorials” on their separate websites. My own advice is always to start with a tutorial and build up from there.
# Other helpful links

<table>
<thead>
<tr>
<th>Python/SciPy/NCL/NCO tips and tricks</th>
</tr>
</thead>
<tbody>
<tr>
<td>- The PyAOS blog mentions that there are beginner, intermediate, and advanced courses on using Python in Climate and Meteorology at the 2014 AMS meeting in Atlanta</td>
</tr>
<tr>
<td>Think Python (learn the language)</td>
</tr>
<tr>
<td>Python in Hydrology (learn Python with examples in hydrology)</td>
</tr>
<tr>
<td>Great website with NCO tricks</td>
</tr>
<tr>
<td>Other links??</td>
</tr>
</tbody>
</table>
ERRRGGGGHHHHHHHHHHHHHHHH

troubleshooting

• Ask real people first!
  – Students in AOS use Python/SciPy, NCO, NCL, MATLAB, etc.
  – One of us has potentially already written a script to do what you need

• Or look within stackoverflow.com
  – “Question and answer site for professional and enthusiast programmers”
  – Big Python user base