NCL Pivot to Python

John Clyne

Computational and Information Systems Lab. (CISL), National Center for Atmospheric Research (NCAR)

Joint WRF/MPAS Users' Workshop
June 10, 2019

Geoscience Community Analysis Toolkit (GeoCAT)

John Clyne

Computational and Information Systems Lab. (CISL),
National Center for Atmospheric Research (NCAR)

Joint WRF/MPAS Users' Workshop
June 10, 2019

Outline

Address questions:

- What is the “Pivot to Python” (GeoCAT)?
- Why are we doing it?
- How will it impact current NCL users?
- What about existing *Pythonized* NCL capabilities?
- When will this all happen?

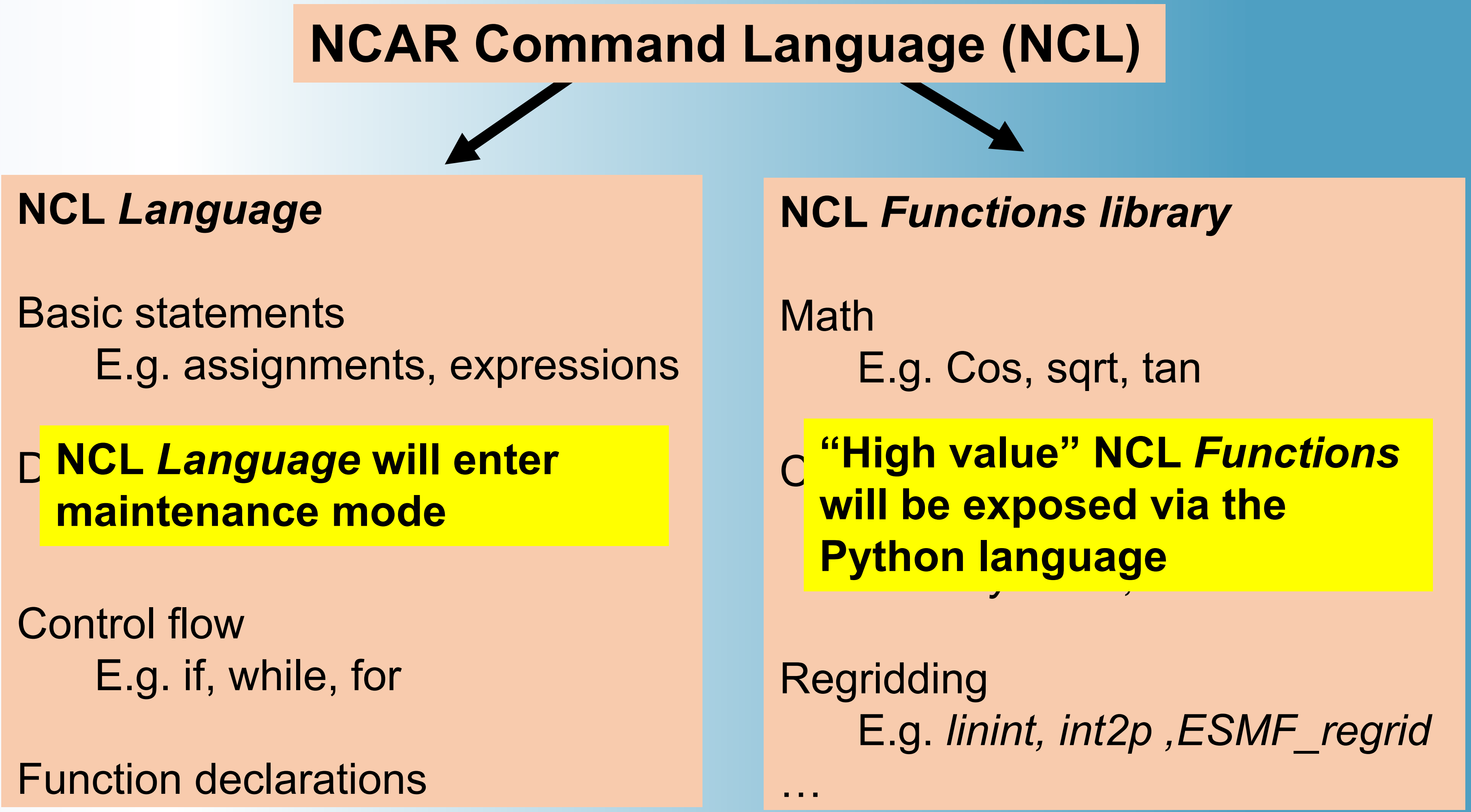
What is the “NCL Pivot to Python”?

1. **Providing Python language bindings to “high value” NCL functions**

2. **Improving scalability of those functions**

3. **Moving from Open Source to *Open Development***

NCAR Command Language (NCL)



NCL *Language*

Basic statements
    E.g. assignments, expressions

D

NCL *Language* will enter maintenance mode

Control flow
    E.g. if, while, for

Function declarations

NCL *Functions library*

Math
    E.g. Cos, sqrt, tan

C

“High value” NCL *Functions* will be exposed via the Python language

Regridding
    E.g. *linint, int2p ,ESMF_regrid*

…

“High Value” NCL Functions

CISL will port: domain specific functionality for which there is both a user demand **and** for which a suitable alternative does not already exist in the Python ecosystem

CISL will NOT port: functions for which a clear alternative already exists in Python, or is not widely used

Examples of types of functions that will be ported (tentatively)

- **Climatology**
  - E.g. *calcMonAnom, calcDayAnom, stdMon*
- **CESM**
  - E.g. *mjo*, vinth*, band_pass**
- **Interpolation**
  - E.g. *linint, int2p ,ESMF_regrid*
- **Empirical Orthogonal Functions**
  - E.g. *eofunc*, eof2data* eofcof**
- ***WRF functions***
  - *Note: already done (wrf-python 1.3.2 released in February)*

Examples of classes of functions that will NOT be ported (tentatively)

- **General math**
  - E.g. trig functions, simple statistics, *log, sqrt, pow*

- **Operating system functions**
  - E.g. *getenv, subprocess, system, file**

- **Date functions (still evaluating)**
  - E.g. *calendar*, cd*, ut*, time_**

Foundational technologies for future development

- **Xarray (xarray.pydata.org)**

- **Dask (xarray.pydata.org)**

Xarray

- **Extends NumPy N-dimensional arrays in two ways:**
    - **Data import/export**: Xarray object knows how to read and write NetCDF and GRIB files
    - **Metadata**: coordinate names, units, etc.

Xarray’s ability to read and write scientific file formats commonly used by ESS community gets NCAR developers out of the file format conversion business

- **Al**
    - `x.sel(time='2014-01-01') # selects data for particular day`

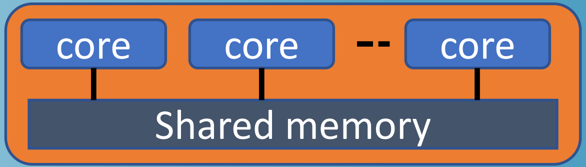
- **Supported by growing list of Open Source geoscience packages**

  => Facilitates compatibility between packages

Dask

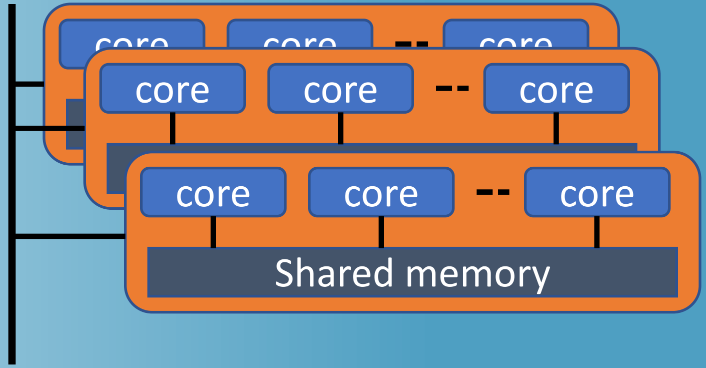
- **Provides parallelism to Xarray**
- **Write once, run everywhere**
  - Shared memory (single node)

    => Run in parallel on your laptop, workstation, etc.



  - Distributed memory (multi-node)

    => Run in parallel on your cluster



- **Scalable performance and memory**
  - Faster computation
  - Larger problem size

Example preliminary single node scaling results with Dask

Convective available potential energy (CAPE) computation

	Serial	Dask (8 threads)
Time (seconds)	380	84
Speedup	1	4.3

8 core Intel Xeon 3.6GHz
WRF 34x240x240 grid
107 time steps
13 GB data

Open Development

- **Open Source:** Software licensed in a way that permits modification and redistribution

- **Open Development:** Open Source software… ***plus* an environment that encourages and facilitates community involvement**
  - Fixing bugs
  - Answering questions
  - Adding new features
  - Porting to new platforms
  - Organizing workshops
  - … and so on

Open Development
Some essential ingredients

- **Contributors guide**
  - Unit test and documentation requirements
  - API documentation
  - Coding style (e.g. PEP8)
  - Submission process (e.g. Pull Request)
- **Open Development workflow platform (e.g. GitHub)**
  - Facilitates code review and source code revision control
  - Enforces maintainer and contributor roles
  - Provides communication channels
- **Continuous integration**
  - Builds code on all platforms, runs unit tests
  - Ensures quality
- **Transparency: developer discussions are public and recorded**
- **Frequent (continuous) releases**

Why Pivot to Python?

1. **Leverage the immense and ever-growing Python Ecosystem**

    - More functionality for users
    - Less work for developers

2. **Scalability afforded by Dask**

3. **Attract new users, particularly early-career**

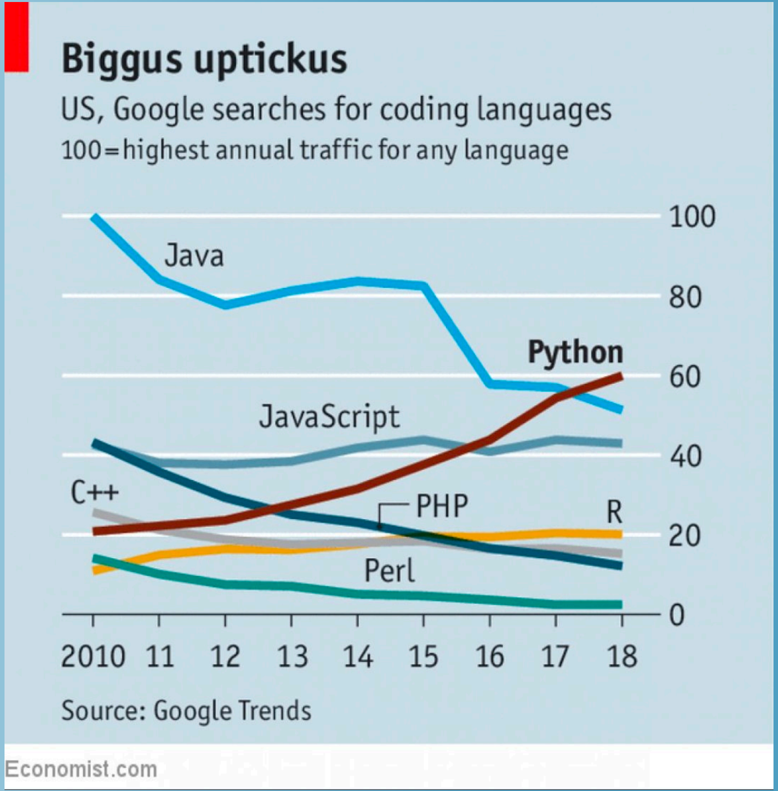
Python by the numbers

- **#3 most widely used language on GitHub**

- **#1 fastest growing language** **[source Google Trends]**

- **#1 for language data science** **[source opensource.com]**

- **30 years old**



Biggus uptickus

US, Google searches for coding languages

100=highest annual traffic for any language

Source: Google Trends

Economist.com

Python Ecosystem

SciPy

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

NumPy

matplotlib

xarray

jupyter

OPEN DATA CUBE

aospy

Iris

PANGEO

xESMF
Wrf-python
GeoViews
Basemap

METPY

What is the impact to current NCL users?

- **NCL will enter maintenance mode**
  - New features will no longer be added to NCL
  - Version 6.6.0 is the last feature release planned

- **CISL will continue to provide maintenance releases for foreseeable future**
  - Fix critical bugs
  - Ensure code builds on currently supported platforms
  - Build and distribute NCL binaries

- **Moving from *Open Source* to *Open Development***
  - Code on GitHub
  - Contributors guide in works
  - Accept contributions from community

  => **User community may continue to enhance, extend, port NCL**

What about existing Pythonized NCL capabilities (PyNIO, PyNGL, wrf-python)?

- **PyNIO**
  - Most likely will be deprecated, functionality replaced by Xarray

- **PyNGL**
  - Refactor for compatibility with Xarray/Dask

- **Wrf-python**
  - Refactor for compatibility with Xarray/Dask

2019 Workplan (tentative and evolving)

	Q1	Q2	Q3	Q4
Release NCL 6.6 and wrf-python 1.3	✓			
Open Development • Contributors guide • GitHub project page • **Publish function roadmap**		✓		
Release 1.0 (climatology focus)				
Release 1.1 (climatology focus)				✓

This is your opportunity to provide feedback!!!

Summary & key takeaways

- **NCL (the language) is in maintenance mode**
  - CISL will fix critical bugs, but not add features

- **Python bindings will be provided for key NCL functions**
  - Triaging key NCL functions is an on-going process that needs your input
  - Currently focusing on climate

- **Moving from Open Source to Open Development**
  - Better opportunities for community involvement and influence

- **Scalable performance is coming**
  - Single node
  - Multi-node

Questions and comments?

How you can get involved

- **Visit the web site often (coming soon, www.ncl.ucar.edu for now)**
- **Provide input**
  - Help triage NCL functions
  - Tell us what’s missing, or wrong
- **Help with user support**
  - Answer a support question
  - Write an example Python script
- **Contribute code**
  - Add a new feature
  - Fix a bug

More on Xarray

Contains N-dimensional NumPy arrays

⇒Everything you can do with a NumPy array can be done with an Xarray

- E.g array syntax:

```
C = A + B # adds all elements of A and B to C

m = C.mean() # computes average of all elements of C
```

Adds metadata (attributes)

- E.g. units, notes, history

Adds coordinate data (NumPy arrays containing coordinates)

- E.g. latitude, longitude, height