NCL, Python, and Related Software User Survey

Analysis and Summary of Results

Mary Haley
November 2018

National Center for Atmospheric Research
Visualization and Analysis Systems Technologies

Table of Contents

Executive Summary

This report is a detailed analysis of the "NCL, Python, and Related Software User Survey" (herein referred to as the "NCL survey") that was conducted in the late spring of 2018. It serves as a supporting document for the "NCL and the Pivot to Python: Discussion and Roadmap" (January 2019).

The NCL survey was conducted in order to better understand and mitigate the effect that a potential "pivot to Python" would have on NCL users, and to get feedback on how users feel about the software being moved to an open development model.

The survey was tailored to answer the following core questions:

- How many NCL users are actively using Python or considering it
- What NCL functionality is critical to its users
- What kind of Python support and training is needed
- How do users feel about an open development model for software

A high-level summary of the survey results was presented to an NCL Advisory Panel (NCL AP) meeting in August 2018, as part of a larger agenda where the pivot to Python was discussed and a roadmap was presented on how NCAR would help NCL users make the transition. Based on the meeting and the survey results, the NCL AP wrote a letter with their findings and recommendations, which is also provided as supporting documet for the "NCL and the Pivot to Python" report.

Introduction

The NCL survey was conducted from May 2 – June 5, 2018, resulting in 699 responses. It was anonymous; no names or email addresses were collected.

While the purpose of the survey was mainly to gather data on the "pivot to Python" decision, there were a high number of questions tailored around desired functionality in NCL. The reason for this was because the survey was being targeted at NCL users, and no assumptions could be made about their knowledge of Python functionality. The NCL functionality requests were evaluated in the context of their availability in Python and its wide collection of open source scientific modules (herein referred to as just "Python").

The survey was originally intended to be advertised only to NCL users, but then Python users in related fields became interested and were invited to participate as well. Their input to the survey was especially valuable in the sections asking about Python workflows.

A high percentage (84.5%) of the survey respondents indicated they were NCL users, while the remaining 15.5% were either inactive NCL users, Python users, or curious bystanders.

Over 81% of the respondents were associated with a college, university, or a research institution and were mainly scientists, researchers, postdocs, and students working in the climate and weather sciences.

Due to the anonymity of the survey, the raw survey data was only made available to the NCL team, NCAR upper management, and members of the NCL Advisory Panel. For details on how the survey was conducted and a full list of questions, see Appendices A and B.

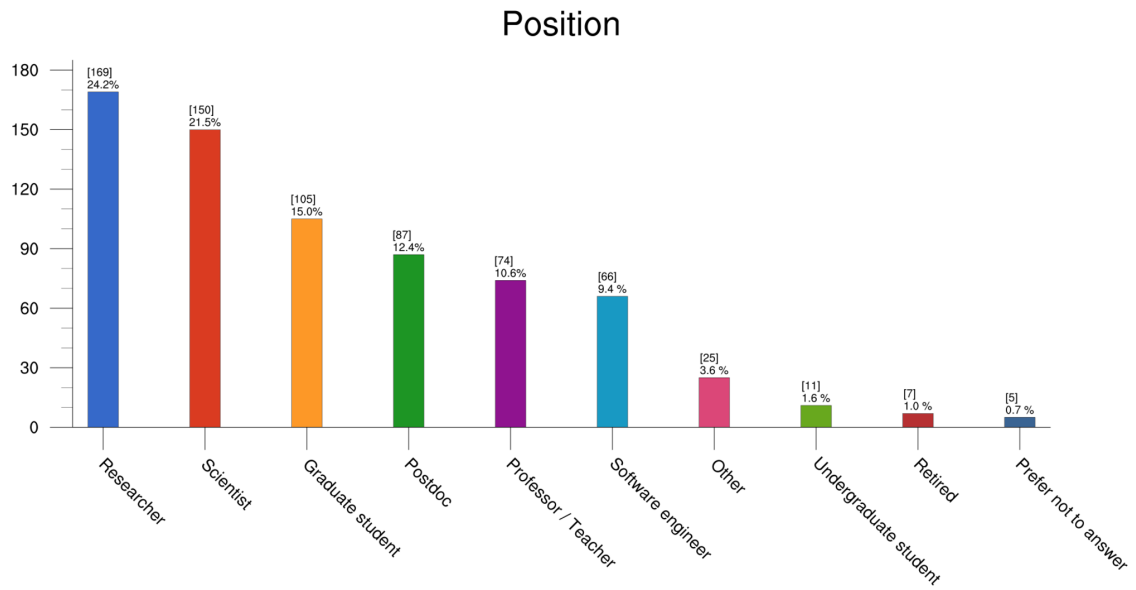
The high number of text-based comments made parts of the survey a challenge to summarize. Many of these comments ended up being quite informative, however, and provided valuable material for the NCL Advisory Panel meeting and subsequent "NCL and the Pivot to Python: Discussion and Roadmap" document.

Survey Section 1: General Demographics

This section of the survey consisted of general demographic questions, intended at getting a better picture of the respondents' research positions and backgrounds. The country of residence question was asked to determine whether the survey was reaching a global audience.

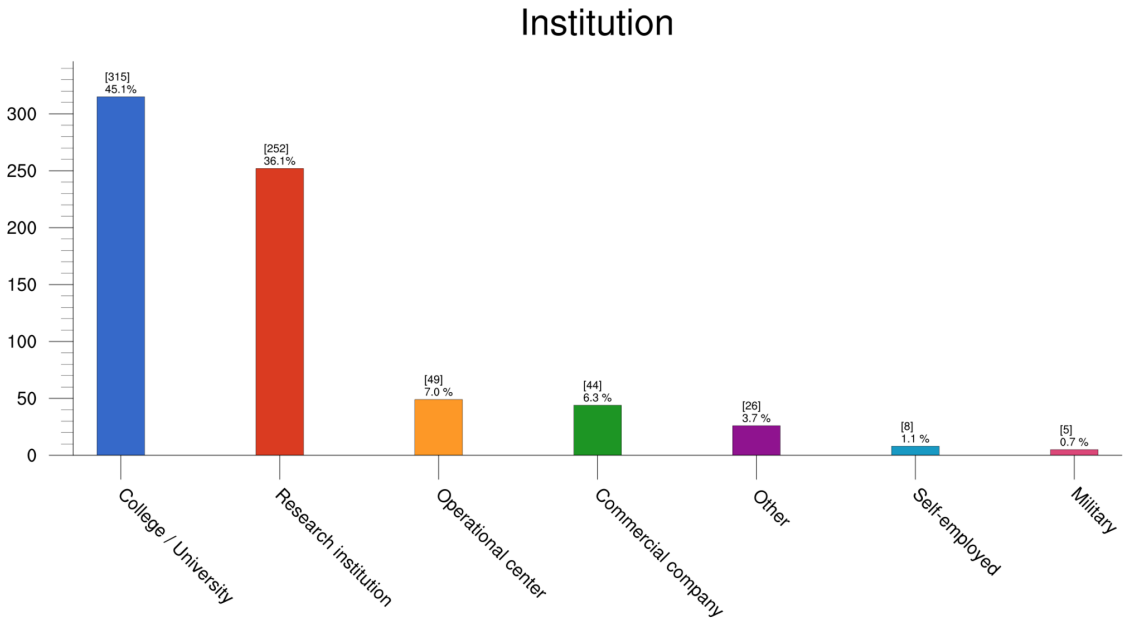
Question #1

The first question asked which title best described the respondent's position. The highest three groups were scientists and researchers (45.7%), graduate or undergraduate students (16.6%) and postdocs (12.4%). The "other" field included entries like "consultant", "programmer", "systems administrator", "operational forecaster", "HPC manager", "hydro-meteorological technician", and "public servant".



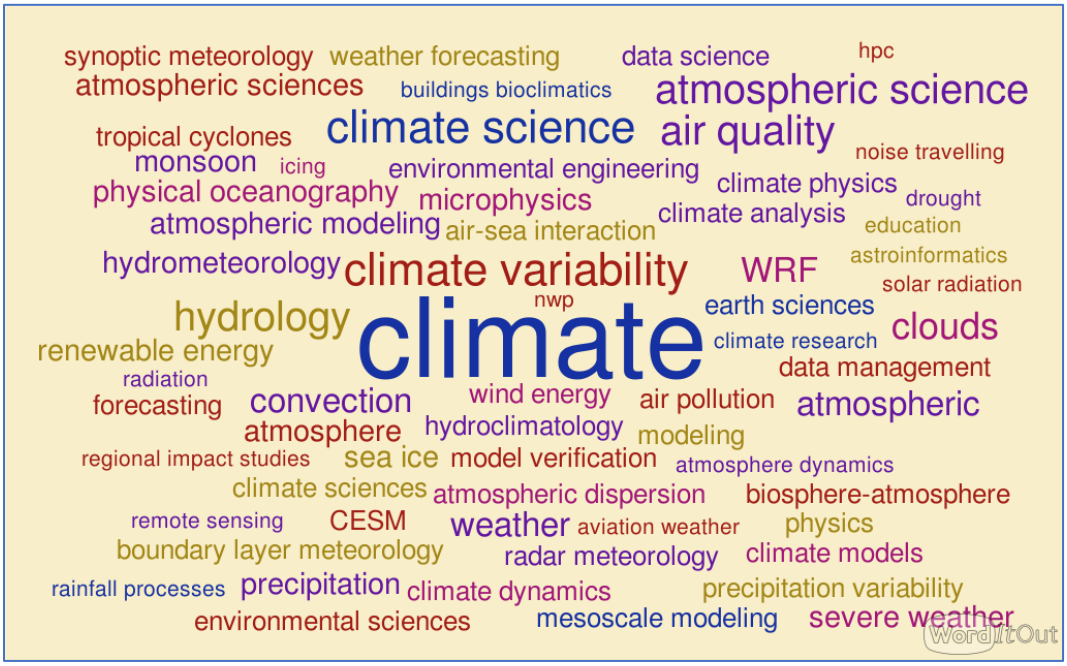
Question #2

This question asked what type of institution respondents were mainly associated with. The highest two groups, which significantly exceeded the other categories, were from a university (45.1%) and a research institution (36.1%). The 32 "other" responses included 15 entries with the word "government". Other responses included "National Weather Service", "private consulting company", and "administration".



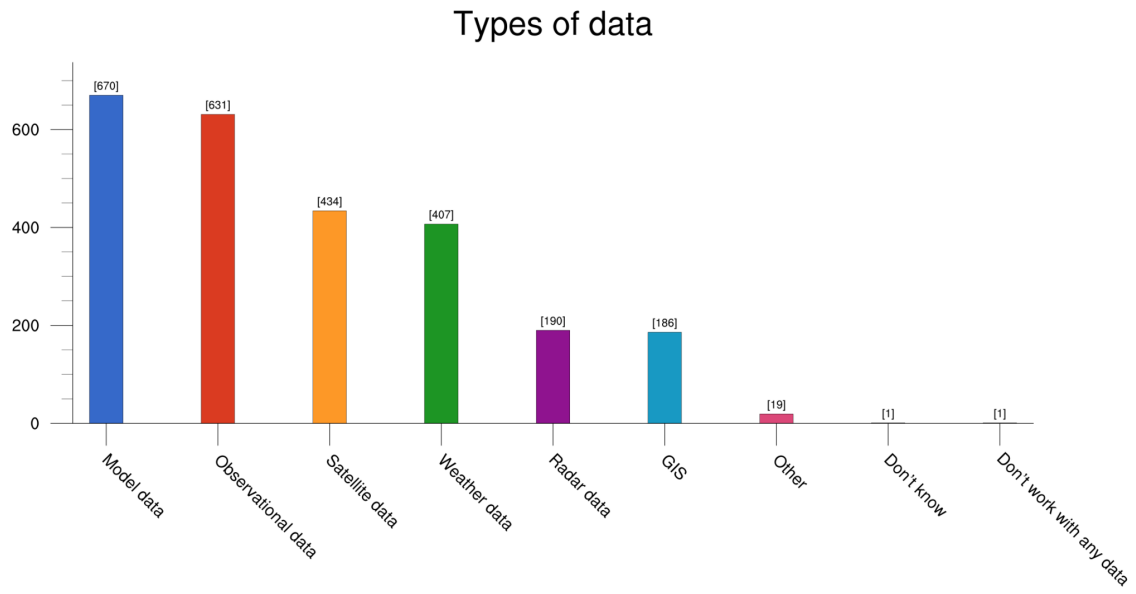
Question #3

This was an optional open text box that asked respondents about their main area of study or research. There were 680 responses to this question, of which 396 were unique. Given the wide variety of the way people answered this question (e.g. using multiple forms of a word like "atmosphere" and "atmospheric"), various free word cloud tools were employed to see what topics stood out. The graphic below is from WordItOut, showing the top 63 answers. There were no real surprises here. Words and phrases like climate, climate science, and atmospheric science were the most frequent entries. Other less common entries included synoptic meteorology, icing, wind energy, tropical convection, ocean wave dynamics, and marine ecosystems.



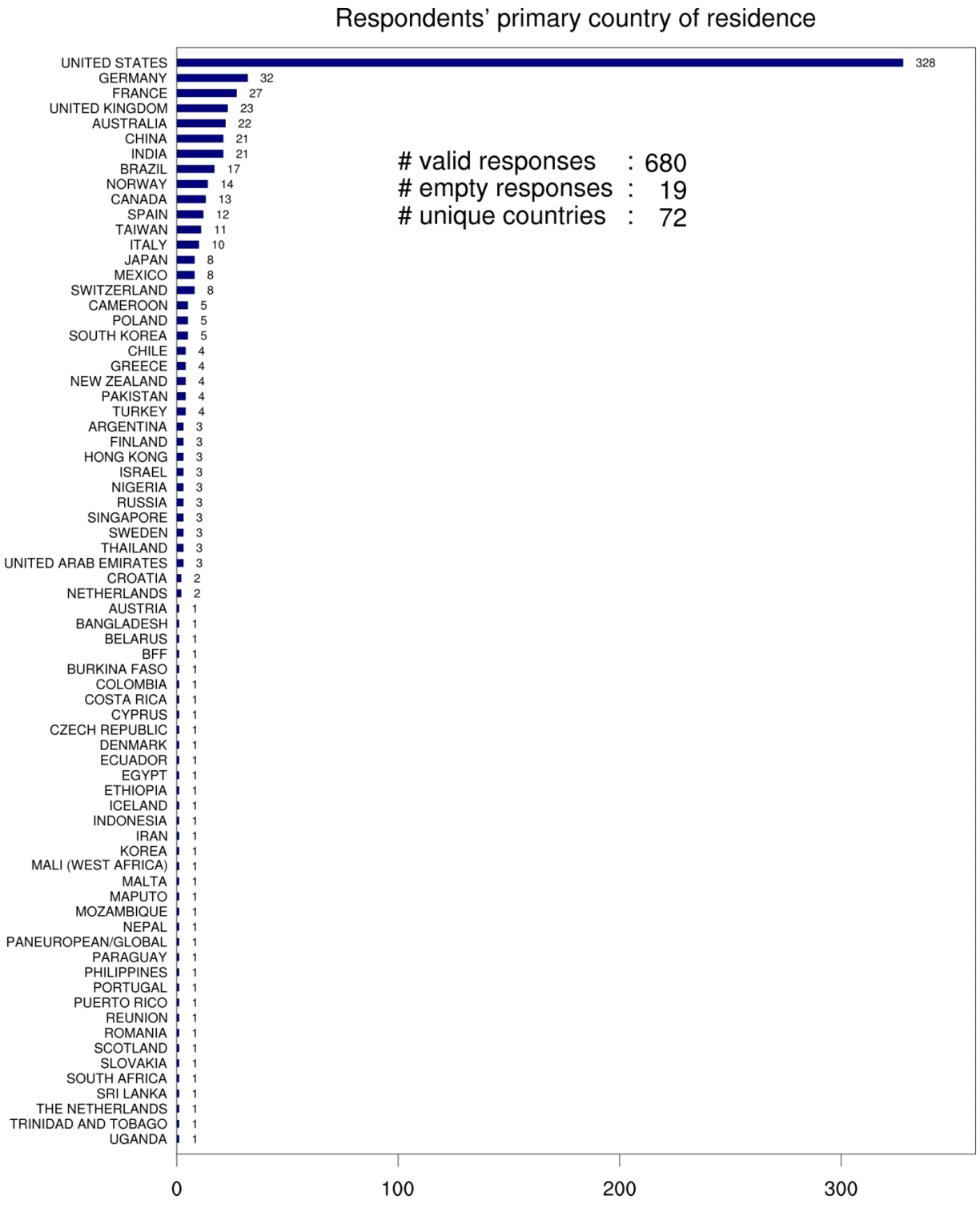
Question #4

This question asked about the types of data the respondents worked with; multiple selections were allowed. There were 2,539 total selections, of which the highest numbers were (unsurprisingly) for model data and observational data. The "other" field, which had 19 entries, included reanalysis data, aircraft data, hydrographic data, and power data (electricity).



Question #5

This question asked respondents about their primary country of residence. There were 680 non-empty responses with 72 unique entries. A significantly large percentage (47%) of respondents were from the United States. The next three largest percentages were from Germany (4.6%), France (3.9%) and United Kingdom (3.3%). Just under 3% of the respondents chose not to answer this question.


Respondents' primary country of residence
UNITED STATES 328
GERMANY 32
FRANCE 27
UNITED KINGDOM 23
AUSTRALIA 22
CHINA 21
INDIA 21
BRAZIL 17
NORWAY 14
CANADA 13
SPAIN 12
TAIWAN 11
ITALY 10
JAPAN 8
MEXICO 8
SWITZERLAND 8
CAMEROON 5
POLAND 5
SOUTH KOREA 5
CHILE 4
GREECE 4
NEW ZEALAND 4
PAKISTAN 4
TURKEY 4
ARGENTINA 3
FINLAND 3
HONG KONG 3
ISRAEL 3
NIGERIA 3
RUSSIA 3
SINGAPORE 3
SWEDEN 3
THAILAND 3
UNITED ARAB EMIRATES 3
CROATIA 2
NETHERLANDS 2
AUSTRIA 1
BANGLADESH 1
BELARUS 1
BFF 1
BURKINA FASO 1
COLOMBIA 1
COSTA RICA 1
CYPRUS 1
CZECH REPUBLIC 1
DENMARK 1
ECUADOR 1
EGYPT 1
ETHIOPIA 1
ICELAND 1
INDONESIA 1
IRAN 1
KOREA 1
MALI (WEST AFRICA) 1
MALTA 1
MAPUTO 1
MOZAMBIQUE 1
NEPAL 1
PANEUROPEAN/GLOBAL 1
PARAGUAY 1
PHILIPPINES 1
PORTUGAL 1
PUERTO RICO 1
REUNION 1
ROMANIA 1
SCOTLAND 1
SLOVAKIA 1
SOUTH AFRICA 1
SRI LANKA 1
THE NETHERLANDS 1
TRINIDAD AND TOBAGO 1
UGANDA 1
valid responses : 680
empty responses : 19
unique countries : 72
0
100
200
300

Section 1: Summary

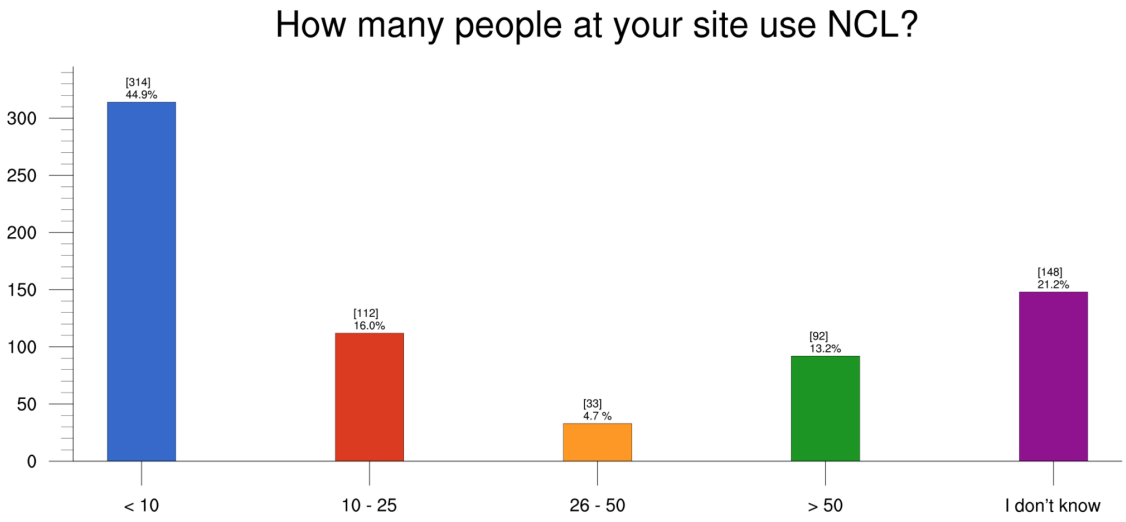
The demographics of the survey respondents align pretty well with what was expected, given who was invited to take survey. They were mainly scientists, researchers, postdocs, and students from research institutions and universities worldwide who study climate and weather.

Survey Section 2: NCL Demographics

This section of the survey asked two questions about the respondent's usage of NCL, to determine which following sections of the survey the respondents should be directed to.

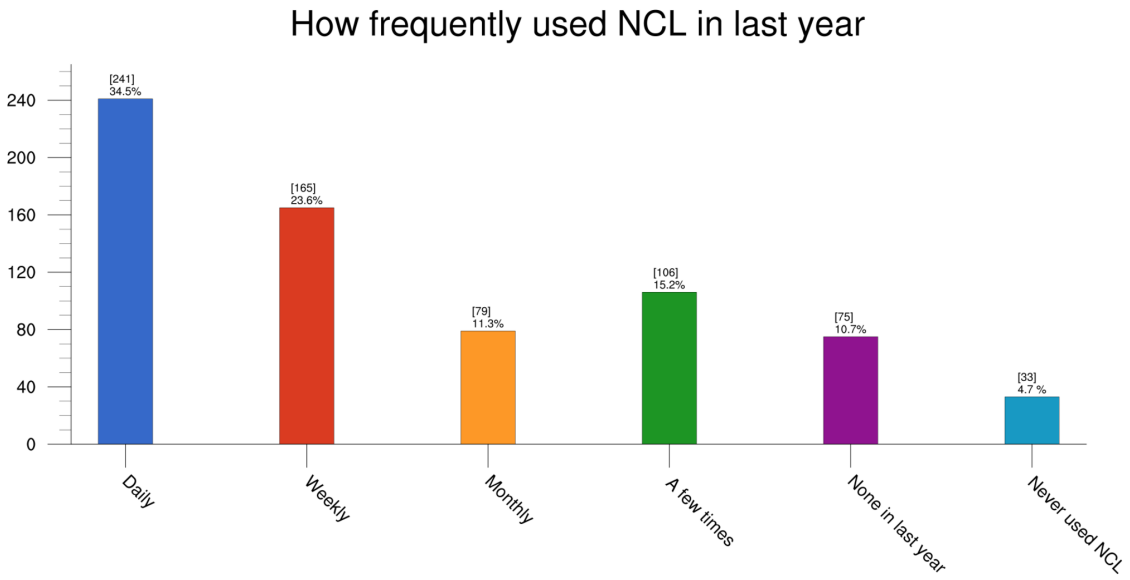
Question #6

This question asked how many people at their site use NCL, mainly for internal metric purposes.



Question #7

This question asked how frequently a respondent used NCL. The 591 respondents (84.5%) who answered with at least "a few times a year" or more were sent to the "NCL Usage and Development" section of the survey, while the remaining 108 (15.5%) who answered "None in the last year" or "I don't use NCL" were skipped over this section and sent directly to the "Python Demographics" section of the survey.



Section 2: Summary

This survey was advertised and geared towards NCL users, so the high number (84.5%) that indicated they had used NCL a few times or more in the last year was not a surprise. Of the 108 respondents that indicated they hadn't used NCL in the last year or at all, 86.1% said they use Python at least weekly and another 11.1% said they use Python monthly or yearly (see question #30).

There were just a handful of people that indicated they hadn't used NCL or Python in over a year or at all. One of these respondents seemed to be taking the survey out of curiosity and another stated they were planning to learn Python soon.

Survey Section 3: NCL Usage and Development - Conditional

This section had nine questions geared towards gathering more demographic information about respondents' use of NCL, and more importantly, what were the top reasons they use NCL.

This part of the survey was conditional; some respondents would not have been directed to this part of the survey based on how they answered question #7 in the previous section. Of the 699 respondents, 591 were sent to this section.

The first four questions in this section included in-depth NCL demographic questions, mainly to determine the range of beginner to seasoned users of NCL.

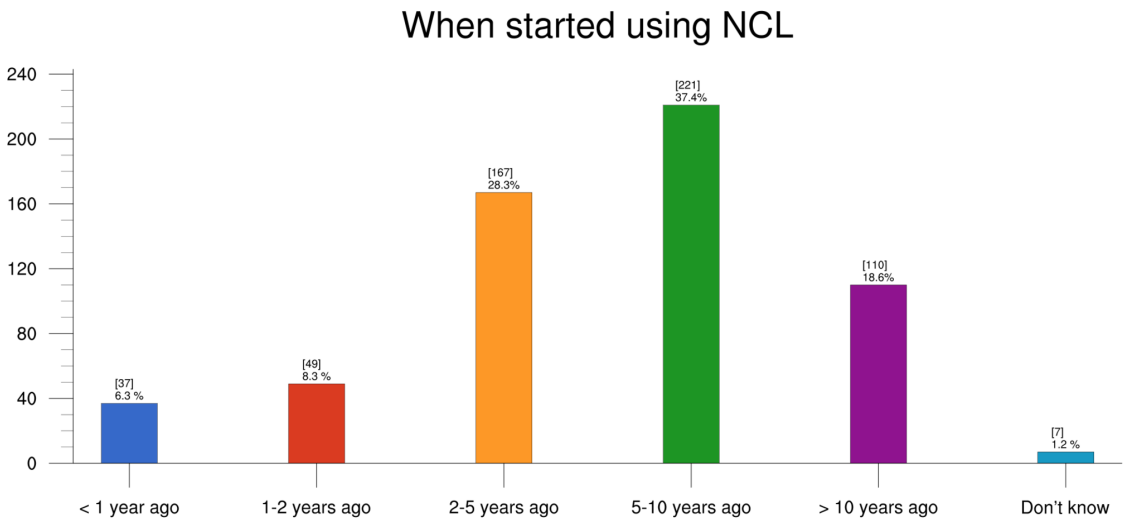
Questions #8 and #9

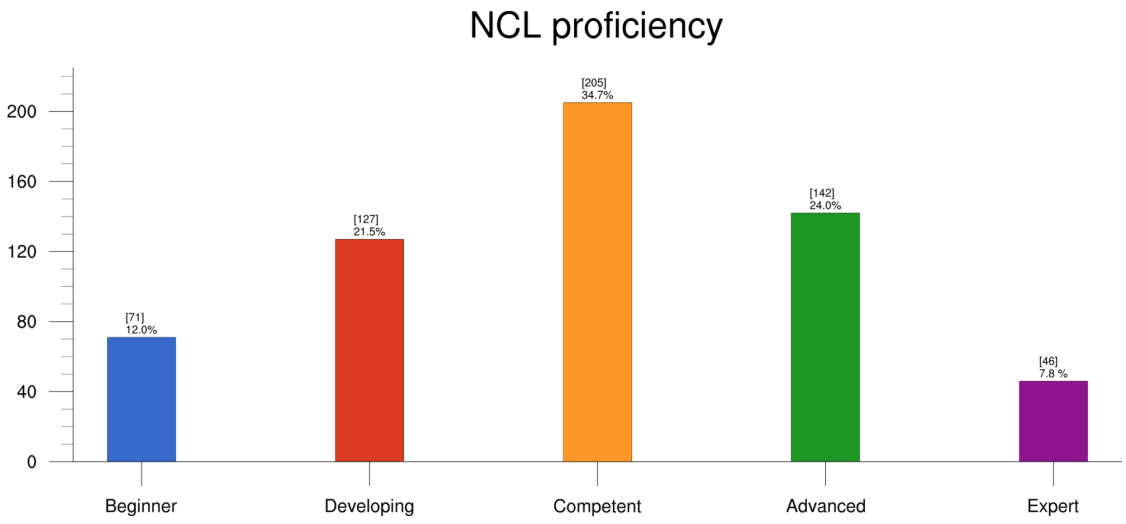
These questions asked when a respondent started using NCL and what they felt their proficiency level was.

A high percentage of the respondents (56%) said they have been using NCL for more than 5 years, while 14.6% have been using it for 2 years or fewer, and 28.3% have been using it in the 2- to 5-year range.

With the proficiency question, 33.5% said they were “developing” or “beginner” NCL users, 34.7% said they were “competent”, and 31.8% said they were “advanced” or “expert”.

While the majority of the respondents to this section have been using NCL for over 5 years, the second question indicated a pretty even spread of their experience levels.



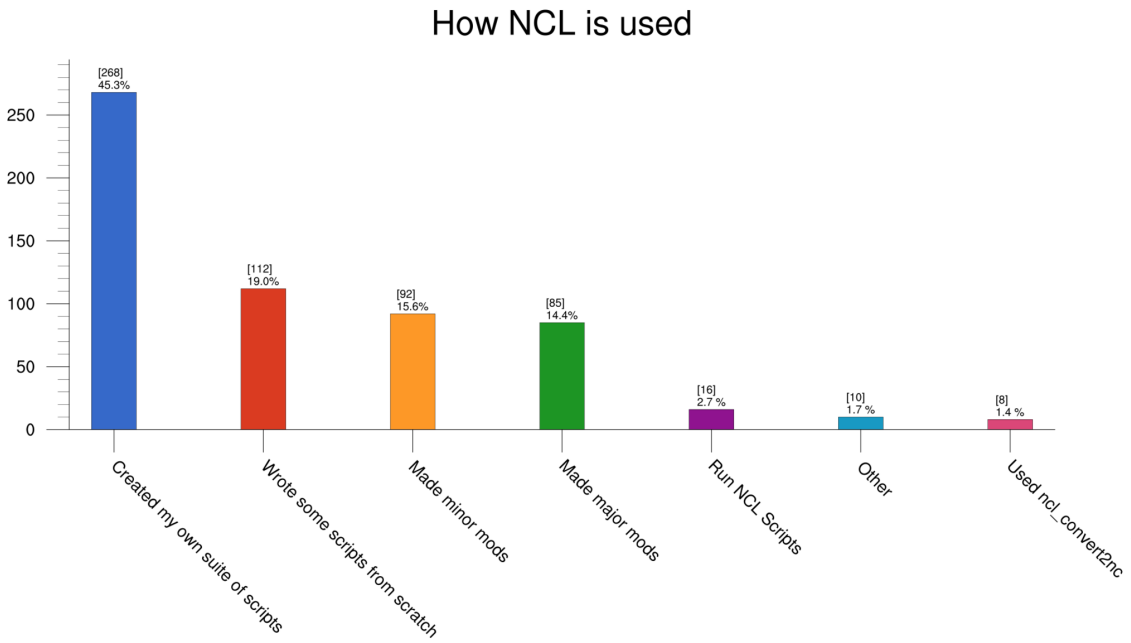


Question #10

This question asked what statement best described how respondents use NCL. Those that answered they were writing their own suite of scripts or writing them from scratch were

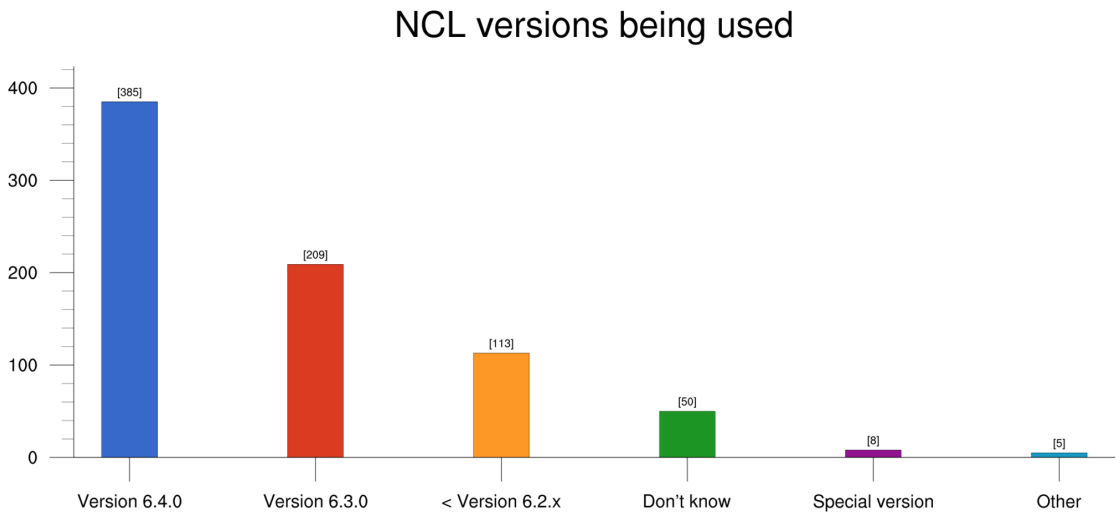
considered to be more advanced users than those making modifications to existing scripts or running scripts written by somebody else. A high percentage (45.3%) said they had developed their own suite of scripts while 33.4% either wrote some scripts from scratch or made major modifications. The rest (excluding the "other" field) made up 19.7% of the respondents.

The "other" field, which had 10 responses, included a couple of "all of the above" type of responses and two responses that said they were still using the old NCAR Graphics package.



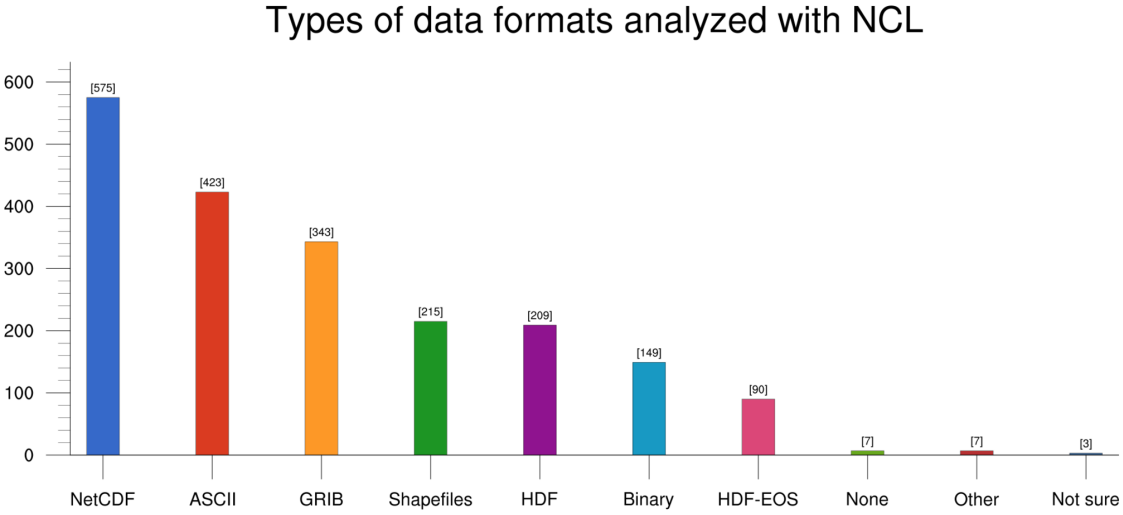
Question #11

This was a multiple-selection question that asked respondents to check all the versions of NCL they were using. This was mostly out of curiosity to see how many were using the most current version (which was 6.4.0 at the time of this the survey). The "other" field only had 5 respondents, with one respondent saying that different versions were used on different platforms.



Question #12

In addition to asking what type of data respondents worked with (question #4), this question asked what data formats they used. This was a multiple-selection question that resulted in 2021 selections. It was no surprise to see the most popular data format being NetCDF, with ASCII and GRIB as the second and third most popular formats. The "other" field, which had 7 entries, included "Bil", CMC's FST files, "CTL/GRD", "Grads", "IEG", "MATLAB old binary", and "matlab".



Questions #13-22

These questions made up a ten-category gridded multiple-choice question that asked about the importance of features in NCL, spanning file input/output, computational, and graphical features. This was a particularly important part of the survey, as it provided information on what features in NCL were important and would potentially need to be ported to Python.

The top three rated features based on the “Very important” responses were:

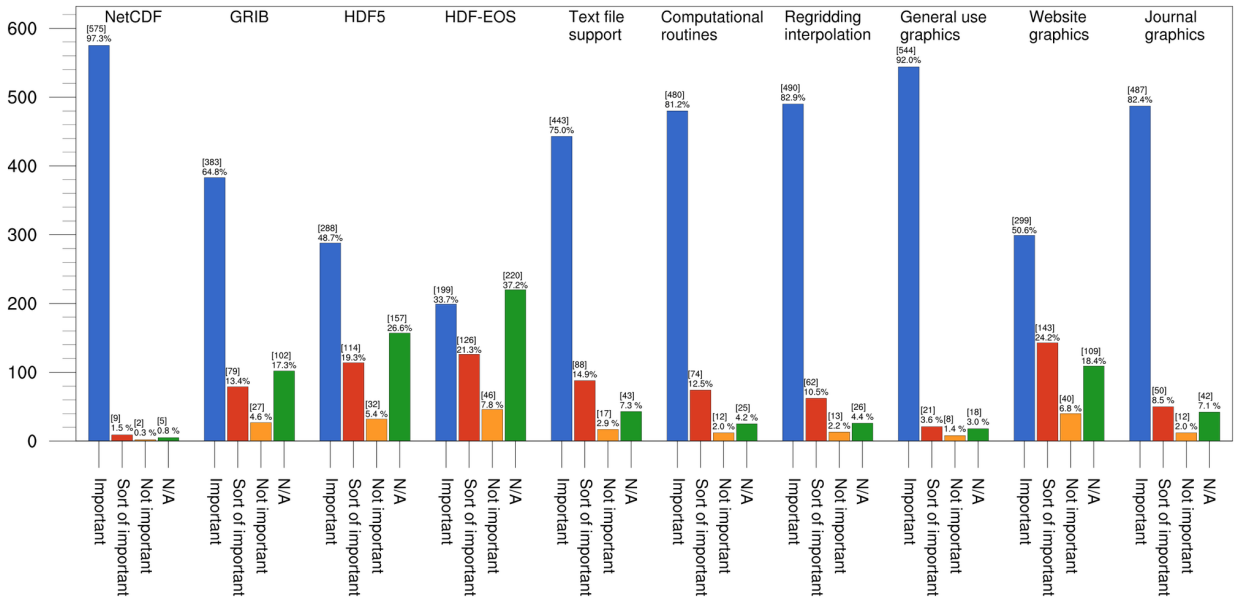
1. NetCDF
2. General use graphics
3. Journal graphics

The top three rated features based on combing the “Important” or “Very important” responses were:

1. NetCDF
2. General use graphics
3. Regridding / interpolation routines

For further grouping purposes, the “Moderately important” and “Somewhat important” responses were combined into one group. The table below displays the top results in descending order of rank. The bar plot displays the results graphically, with the categories in the same order as they appeared in the survey.

NCL Features	**Very Important / Important**	
	Count	**Percentage**
NetCDF	575	97.3
General use graphics	544	92.0
Regridding / interpolation	490	82.9
Journal graphics	487	82.4
Computational routines	480	81.2
Text file support	443	75.0
GRIB	383	64.8
Website graphics	299	50.6
HDF5	288	48.7
HDF-EOS	199	33.7



Question #23

This was an open text question that asked respondents to indicate "other features or aspects of NCL that are important or useful to you". There were 111 non-empty responses.

This question could have been worded better, because some respondents took this as an opportunity to request new features, while others stated what current features they considered the most important. In some cases, it was hard to tell if something was a feature they liked or a request for a new feature.

The top responses included computational and graphics functions tailored for WRF (Weather and Research Forecast Model), computational routines tailored for climate and weather (with a few mentions of statistical routines), the quality and customizability of the graphics, and the use of NCL's WRAPIT program for calling Fortran routines from NCL.

Top Categories of NCL Features	Count
WRF	11
Computational	11
General requests	10
Wrapping Fortran	10
Graphics	10
General File I/O	8
Parallelism / support for large data	6
Core language	6
Statistical	6
User support, training, and documentation	5
GRIB	3
Shapefiles	3

Metadata handling	3
IDE / Jupyter Notebook	2

Questions #24-28

These questions formed a gridded set of multiple-choice questions that asked about the usefulness of the NCL website, email lists, manuals, workshops, and webinars. The purpose was to gauge what kind of support is the most important to adopt for Python software that will be developed. In order to get a better idea of the top-rated categories, the "Very useful" and "Useful" responses were combined into one field and included in the table below.

Type of NCL Support	**Very Useful / Useful**	
	Count	**Percentage**
NCL website	548	92.7
Manuals	434	73.4
Email lists	380	64.3
NCL Workshops	188	31.8
Webinars	83	14.0

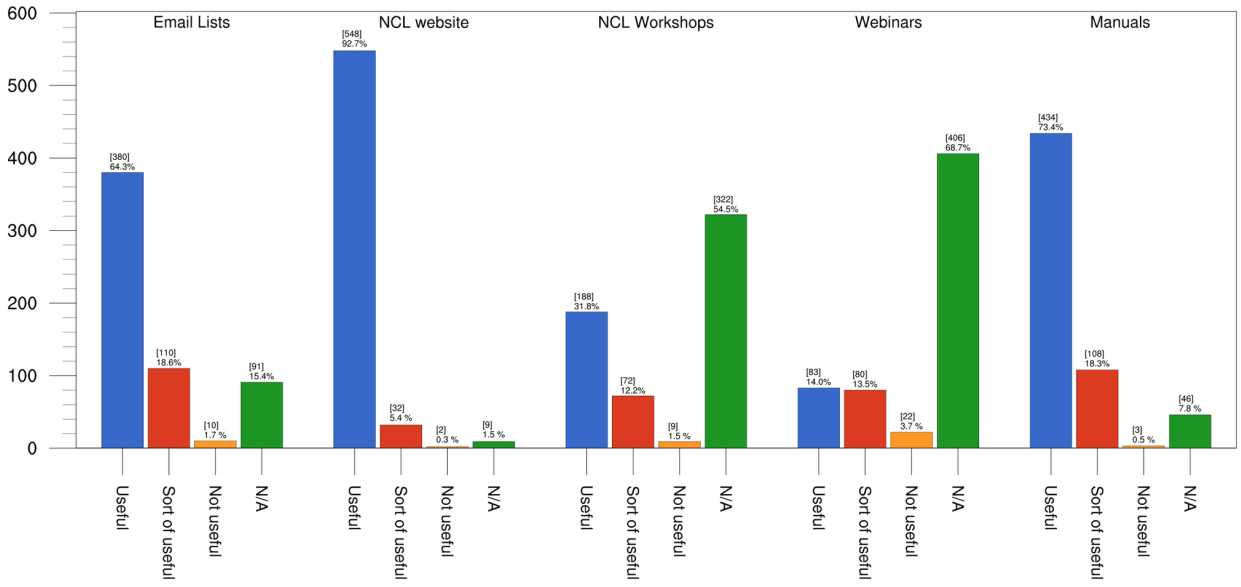
The high rating for the NCL website was somewhat expected, as UCAR's web statistics page (webstats.ucar.edu) indicates that the NCL pages are consistently in the top ten of all of NCAR's websites.

Not everybody has had the opportunity to take an NCL workshop or a webinar, so these answers were expectedly lower.

In retrospect, given the high rating for the manuals, it would have been of value to inquire which manuals were the most useful. The NCL User Guide—a 260+ page manual written by Karin Meier-Fleischer and Michael Böttinger of Deutches Klimarechenzentrum (DKRZ)—is the most likely source of this higher rating. This is an extensive user guide that includes a step-by-step section on creating graphics with NCL.

One difficulty with training is how to make it more accessible to people who can't afford to travel to Boulder for a local workshop. The NCL team did a series of webinars a few years ago, but of the 591 respondents, 68.7% said that the webinars were not applicable.

The graph below shows the full results of this question, with the "Very useful" and "Useful" fields combined into a "Useful" field and the "Moderately useful" and "Slightly useful" fields combined into one "Sort of useful" field.



Question #29

This was an open text question that allowed respondents to indicate other ways they get support for NCL or comment on support in general. There were 96 non-empty responses of which the top three methods were colleagues, Google, and stack overflow. A few people indicated they didn't know about the webinars, so this may be an area of interest if NCAR was able to offer more of them and advertise them better. There were some suggestions as well:

"would be good if you try to make NCL youtube channel."

"I wish there was an IRC room"

"More online courses"

"I like NCL, I think is a great tool for earth sciences, but I have not idea I can't migrate from Python, maybe for me the IDE is the key"

"Maybe the video for how to use the NCL is welcome too if possible."

"More tutorials and exercises to work with"

"More workshops"

"I haven't had much chance to work with NCL yet, but more information on WRF within NCL would be nice. A lot of it was pretty lacking in my experience."

"Didn't know there were webinars but I would find this very useful. Videos on different programming features and how to make plot would be extremely useful."

Section 3: Summary

The responses to questions #13-23 provided an excellent snapshot of the top reasons that people use NCL:

- Its close association with NetCDF and the ability to easily access and manipulate metadata
- The high number of specialized computational routines
- The variety of regridding and interpolation functions
- The suite of WRF-NCL computational routines
- Its support of the GRIB record format
- The ability to create high-quality graphics for journals and web
- The ability to wrap Fortran routines and call them from NCL

In thinking about a transition to Python, the responses to this section also highlighted the fact that existing Python packages will need to be reviewed by the NCL team not just for missing functionality, but for the robustness and usefulness of existing functionality like support for GRIB files and customizable and high-quality two-dimensional graphics.

Questions #24-29 were geared towards finding out which methods for training and support that NCL respondents preferred, and ultimately, what might be the best methods to emulate in answer to the core question "What kind of Python support and training would be useful?"

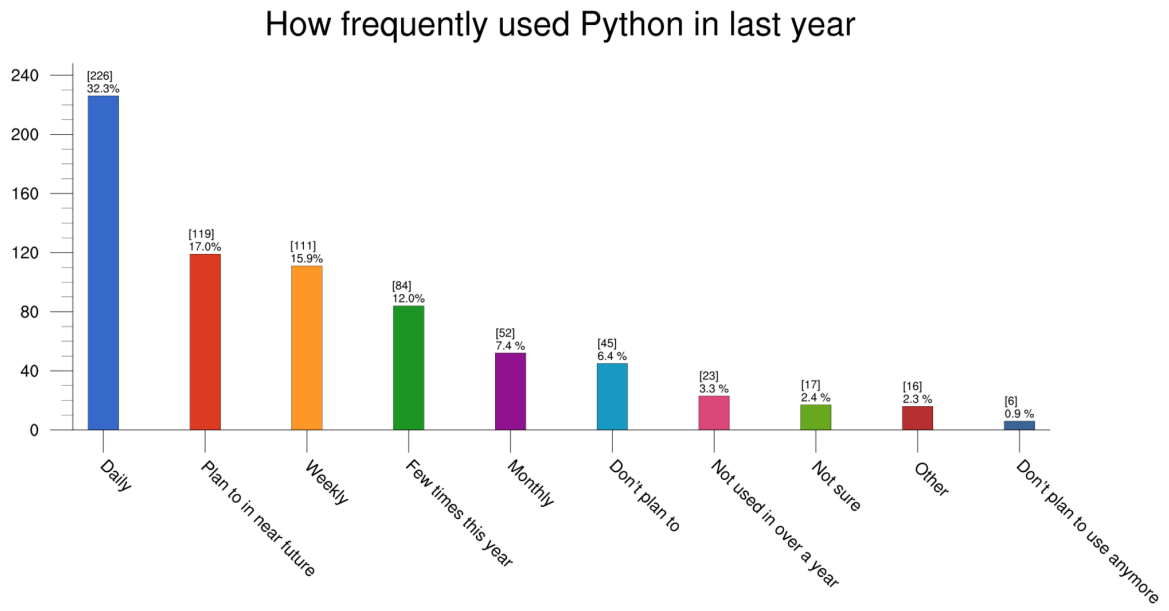
It is no surprise that the online documentation had the highest ratings, especially given the global usage of NCL. One challenge that was identified is how to better provide online training. One possible answer comes from a later response in the Python training question that mentioned a set of Data Carpentry lessons for using Python in the atmospheric and oceanic sciences (see section 6 on "Python Training").

Survey Section 4: Python Demographics

All respondents were directed to this section of the survey. It only had one question, which was included to gauge whether a respondent should be sent to the section on "Python Usage".

Question #30

The question was intended to determine the frequency of Python usage. Of the 699 respondents, 51 of them answered with either "I have used it in the past, but don't plan to anymore" or "I have never used it and don't plan to", so they skipped over the sections on "Python Usage" and "Python Training" and were sent directly to the "Software Development Priorities" section. Another 17 respondents answered "I'm not sure", so they skipped the "Python Usage" section and were sent to the "Python Training" section. Everybody else (631 respondents) was sent to the "Python Usage" section.



There was an "other" field allowing respondents to indicate other ways they use Python. Here's a sample of the responses:

> "I use python in ArcGIS but not with climate/weather data"
>
> "I use Python daily, but only for generating other scripts"
>
> "I've been slowly learning python for a number of years, though have never had a project to 'dive into it'. Due to wrf-python, I am planning to use it a lot in the future."
>
> "I use it randomly, mostly other people's scripts when needed"
>
> "Just started using Python, time to time I practice on how to use Python (with NCL)."
>
> "I have used other people's scripts occasionally, currently unsure if I need to learn it more seriously"
>
> "I have been wanting to learn Python and implement it in my classes but have just not had the time to concentrate on it."
>
> "I have never used it, but I can imagine that I might need to in the future."
>
> "I do not know Python yet, but I plan to learn it as soon as I have time. I would like to transition into using it as my sole language rather than pairing Matlab, NCL, and R."

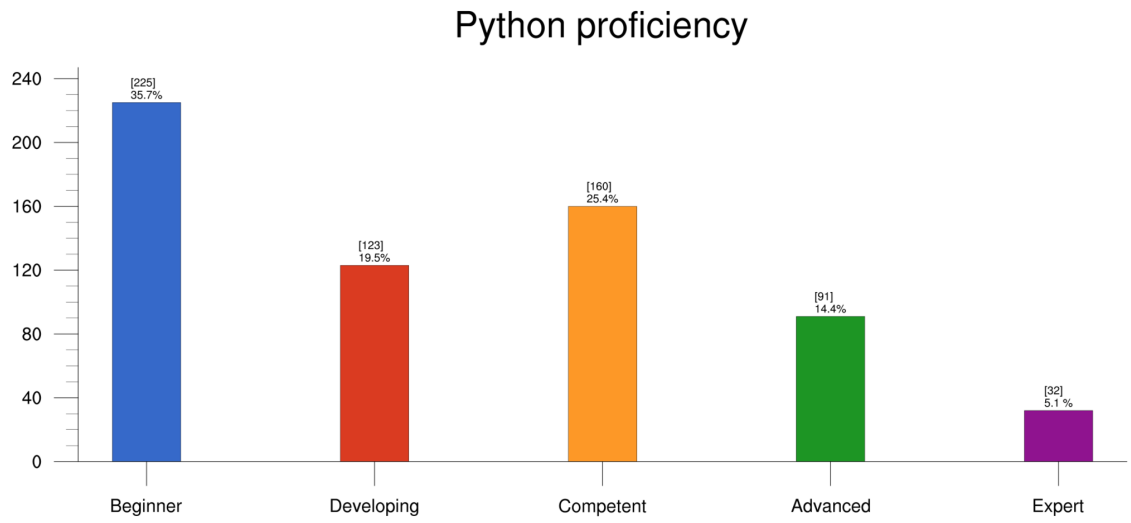
Survey Section 5: Python Usage – Conditional

This section contained nine questions geared towards respondents' level of proficiency with Python, how they use it, and what Python modules they use. There were also asked specific questions about usage of PyNIO, and PyNGL, and WRF-Python.

This part of the survey was conditional. Some respondents would not have been given the opportunity to answer these questions, based on how they answered question #30 in the previous section. Of the 699 respondents, 631 (just over 90%) were sent to this section of the survey.

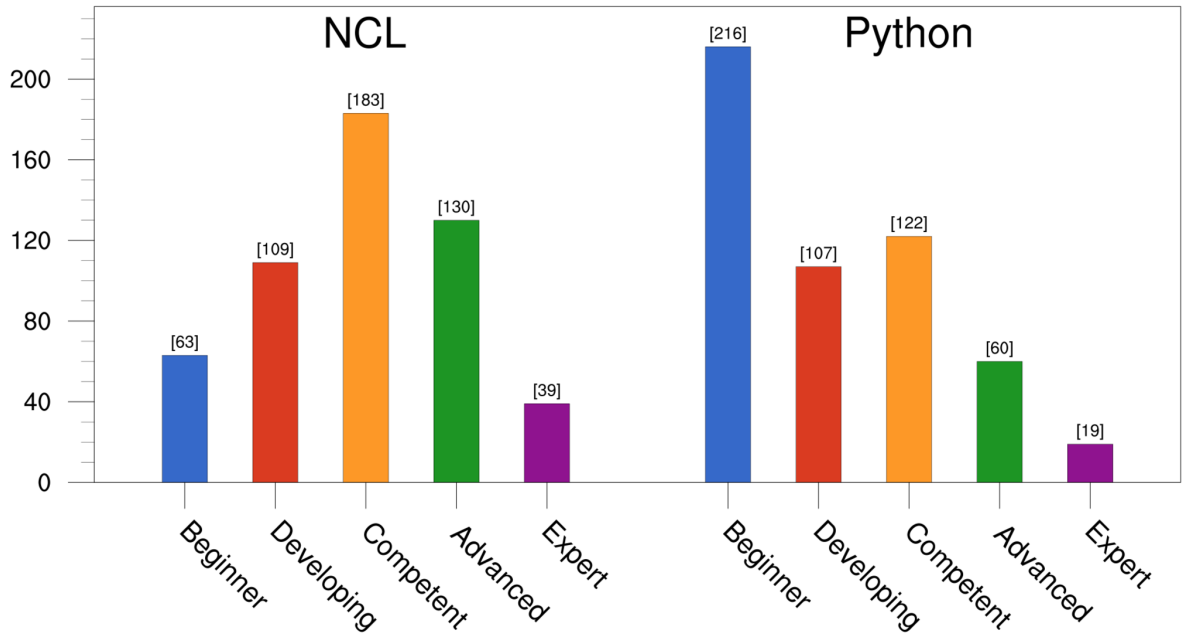
Question #31

Similar to question #9 that asked about the level of proficiency with NCL, this question asked the same of Python. Most respondents (55.2%) said they were "beginner" or "developing" Python users, 25.4% said they were "competent", and 19.6% said they were "advanced" or "expert".



The number of active NCL users (those that made it to the "NCL Usage" section of the survey) who also made it to the "Python Usage" part of the survey was 524 (88.7%). In order to better understand what the Python proficiency was of these 524 NCL users, a bar graph was created showing the two side-by-side.

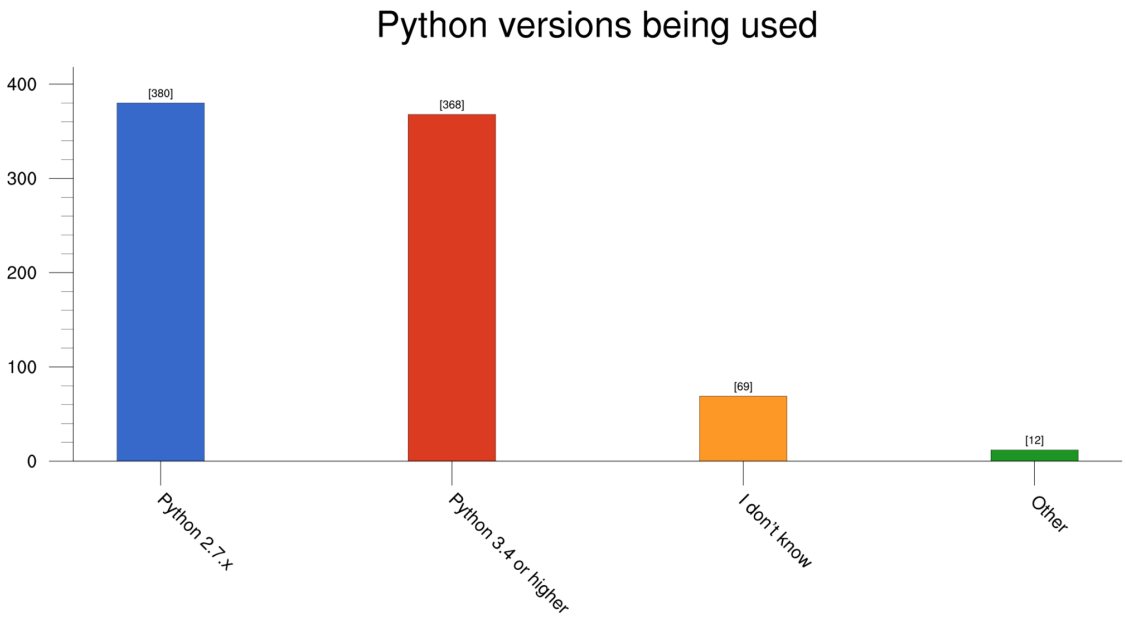
The graph shows that 61.6% percent of these NCL users are at the beginner or developing level of Python proficiency, 23.3% at the competent level, and the remaining 15.1% are at the advanced or expert level.



Question #32

This multi-selection question asked respondents to check all the versions of Python they were using. It was included to get an idea of whether people were using Python 2.*x* or 3.*x*. Python 2.*x* is being phased out 2020 (see pythonclock.org), and Python 3.*x* introduces a number of backwards-incompatible features that makes it difficult for some Python packages to be updated.

There was a total of 829 entries. More than half of the 631 respondents to this part of the survey indicated they were using Python 3.x and a slightly higher number said they were still using Python 2.*x*. There was an "other" option with 12 responses. Of these, three people indicated they were still on Python 2.6.6. Six people responded in a way that implied that this question wasn't applicable. The important takeaway from this question is the encouragingly high number of respondents that have adopted Python 3 and subsequently how urgent it is to support Python packages under the new version.

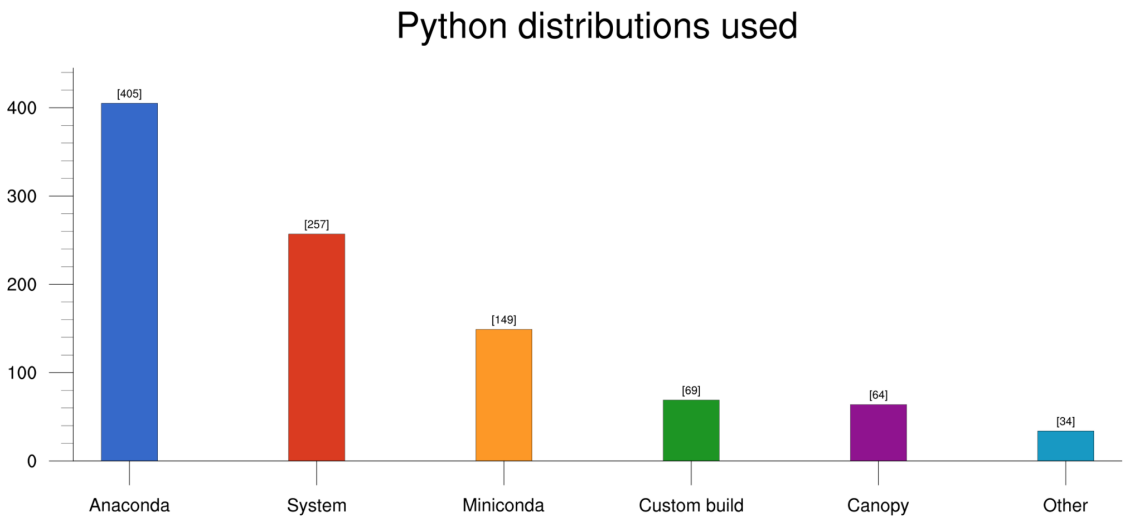


Question #33

This was a multi-selection question that asked respondents to select all distributions of Python they use. The NCL team uses conda (conda.io) for distributing its three Python packages, so this question was of particular interest to see if other distribution methods were popular. There were 978 selections made.

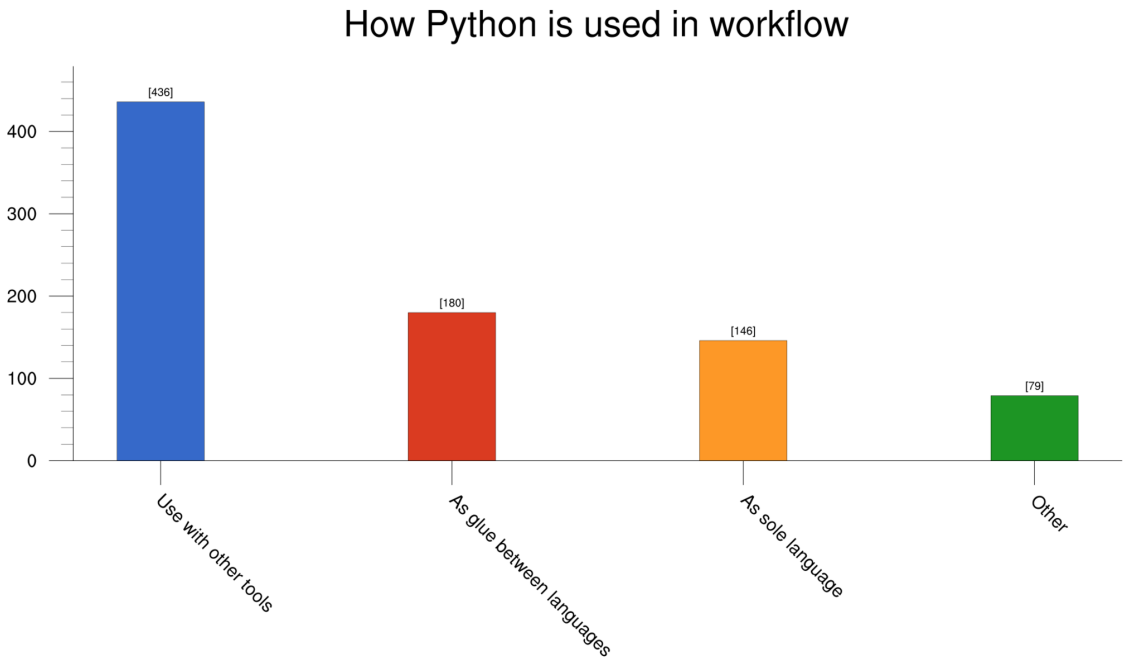
Of the 631 respondents, 64% indicated they were using Anaconda and 23.6% were using Miniconda, which are both part of the "conda" distribution. For the 41.7% that selected "system", this could be whatever came with a system, or it could be a version of Python installed using one of the other distribution methods. For the "other" field, there were 34 responses which included entries like "MacPorts", "Homebrew", "pypi", and "pip".

Given the high number of Anaconda and Miniconda responses compared to the other methods, "conda" seems to be a good choice for now. The NCL team is considering adding support for MacPorts, as they get a number of questions on this through the ncl-install email list.



Question #34

Respondents were asked to select all the ways Python is used in their scientific workflow. There were 841 selections including the "other" field. Just over 69% said they use it with other tools while 23.5% said it was their sole language.



The "other" field, had 79 responses. About 30 of these indicated they hadn't started using Python yet, while another 21 said they were planning to soon. The rest of the 28 responses were further details on how Python was being used in their environment and a few were rather vague.

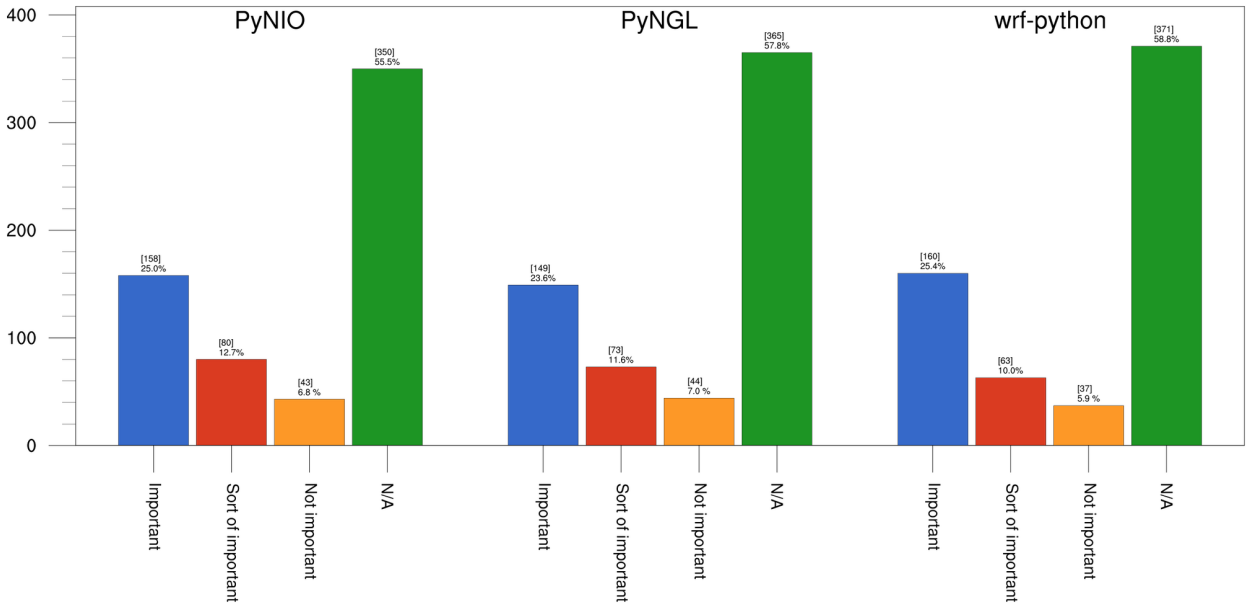
There were several comments that mentioned both NCL and Python:

> "I'm moving from NCL to Python. More flexibility, more packages and the opportunity to work with NCL-like packages later if required."
>
> "I use NCL for my research, Python for teaching to undergraduates and graduate students... I need to learn how to put them together better!"
>
> "I do not know Python yet, but I plan to learn it as soon as I have time. I would like to transition into using it as my sole language rather than pairing Matlab, NCL, and R."
>
> "For task parallelization (glue for parallelization of NCL code) and general scripting (because I prefer it over bash, etc.)"
>
> "I often use it for output from idealized models as I prefer to use NCL due its built-in functions for GCM output or observational data."
>
> "I extract data from netcdf files with NCL and then use in Python"
>
> "I used to use primarily NCL now it is a mixture of both languages. I find the graphics easier to create in python."
>
> "I use it to parse inputs on web forms to send to ncl to create custom images for web tools"
>
> "In order, I use Matlab, NCL, R, Fortran, Python but based on this question, I suppose I should check out Python"
>
> "I use NCL for all my plotting and analysis, but I use python mostly for moving files around and submitting model runs."

Questions #35-38

These questions were asked to gauge if and how the three Python packages developed by the NCL team—PyNGL, PyNIO, and WRF-Python—were being used. The first three questions were in a gridded table asking respondents how important these packages were in their research.

As with previous questions, the "Very important" and "Important" responses were combined into one "Important" field, and the "Moderately important" and "Slightly important" into one "Soft of important" field. The "N/A" responses were taken to mean respondents were not using these packages.



Just over 25% said PyNIO and WRF-Python were either "Important" or "Very important" in their research, while 23% said the same for PyNGL. There were quite a high number of people (55.5% or higher) that indicated they were not using these packages. Based on some of the negative comments encountered in later sections of the survey, many respondents said they felt that PyNGL was lacking in computational support and PyNIO didn't have full support for newer features in NetCDF4. WRF-Python is a much newer package than PyNIO or PyNGL, and it has shown a consistently higher number of conda downloads (averaging over 2,000 downloads a month on conda). It specifically serves the WRF user community and provides a unique functionality not available in other Python packages.

Question #38

This was an open text question that asked respondents how they used the three packages listed in the previous question. There were 152 responses, which mostly fell into the following categories:

- PyNIO
  - Reading GRIB data
  - Reading NetCDF
  - Reading HDF_EOS
  - Reading a mix of data formats
- PyNGL
  - Graphics for publications
  - Graphics for websites
  - Graphics for other purposes
- WRF-Python
  - Reading WRF output files
  - Calculating diagnostics

- Plotting WRF data
  [Note: WRF-Python itself doesn't actually do graphics, but the WRF-Python website has examples showing how to use other packages like matplotlib and cartopy, so this may be what respondents were referring to.]

About 18% of the 152 responses mentioned that they use a mix of two or more of the above packages, and 11% mentioned they were either unaware of this software or hadn't used them yet.

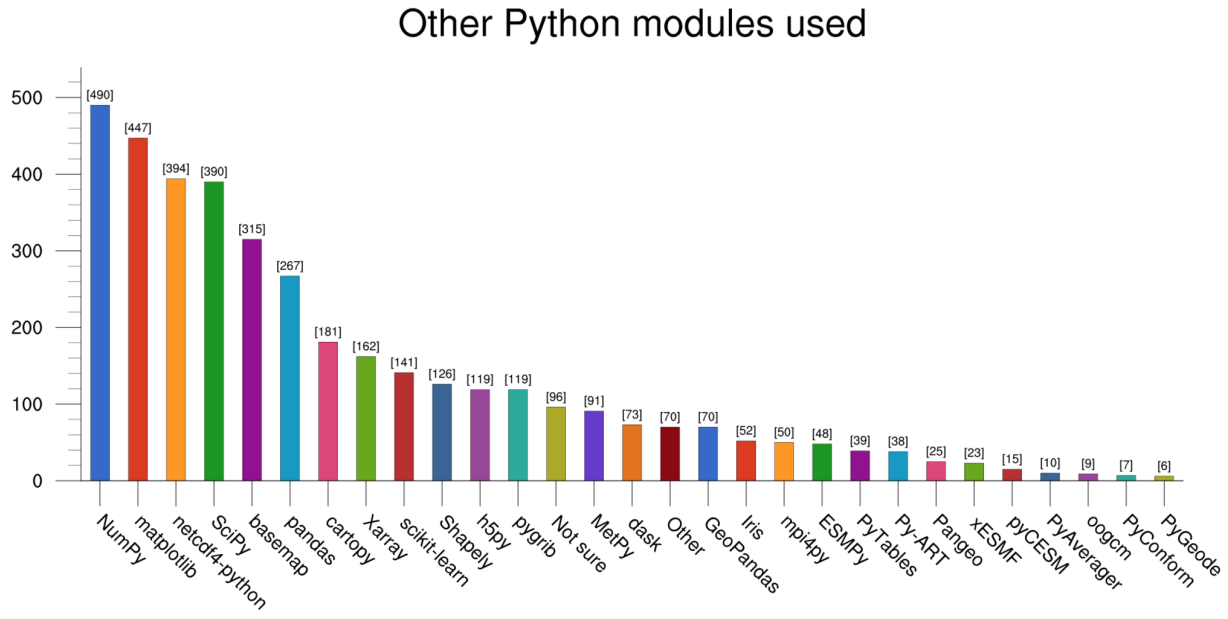
There were a few respondents that indicated that installation barriers kept them from being able to use the packages, for example, due to Python 3 not being supported or due to environment clashes with other Python packages.

WRF-Python has been available under Python 3 for a couple of years, while at the time this survey was conducted, the PyNIO Python 3 port was not widely advertised and PyNGL was still in beta testing.

It is not clear from the other installation comments what the issues might be, but one known issue is the difficulty of getting certain packages on conda-forge to play well together. This situation has gotten much better as conda-forge has stabilized.

Question #39

This was a multiple-selection question where respondents were asked to select all the Python modules they use. There were 3803 packages checked and an additional 70 responses in the "other" field.



The top three packages indicated in the bar graph (NumPy, matplotlib, and netcdf4-python) are to be expected. NumPy is necessary in order to do numerical calculations on multi-dimensional arrays, and is a dependent of many of the packages listed above. Matplotlib is the de facto two-

dimensional graphics plotting package in Python. Netcdf4-python—developed by the same team that develops NetCDF (Unidata)—is the most common package used for reading/writing NetCDF.

There were just over 70 unique packages entered ins the "other" field (some people listed multiple packages), which are included in the table below.

List of Python Packages Mentioned in the "Other" Field							
arcgis	bokeh	bottleneck	cdat	cdat8	cdms	cdms2	cmocean
cmor3	cv2	cython	dask	datetime	descartes	eccodes	ecmwf
eofs	fiona	gdal	geopandas	geoviews	glob	glob2	gobject
grib-api	gsw	holoviews	ipython	ipywidgets	json	jupyter notebook	mysql-python
nco	netcdf4	nose	numba	numexpr	ocgis	os	pika
pil	pillow	pyasaptools	pycdo	pycodestyle	pyfits	pygtk	pyhdf
pylint	pyparsing	pyproj	pyreshaper	pyreverse	pyspharm	pytest	pytest-regtest
pyx	requests	rpy2	scons	seaborn	sharppy	siphon	sympy
sys	urllib	urllib2	uvcdat	vcs	windspharm	wxpython	xmgrace

Section 5: Summary

Based on the responses to question #31, a high number of active NCL users (88.7%) have either adopted Python in their scientific workflows or are considering it in the future. Of these users, 61.6% consider themselves at the developing or beginning level of Python proficiency, while the remaining 38.4% consider themselves at the competent level or higher. This information is important in determining what level of support NCL users might need in a transition to Python, as there will likely need to be a beginner's tutorial on the scientific use of Python itself, as well as tutorials on converting NCL scripts to Python (see section 6).

The responses to questions #34, 38, and 39 were particularly useful in better understanding the workflows and software stacks used by the Python geoscientific community. The open text answers to the "what Python modules do you use" resulted in 70+ Python additional packages that might be worth a closer look when the NCL team researches the core question on what functionality is crucial or missing in NCL that needs to be ported to or implemented in Python.

Questions #35-37 were geared to determining the longevity and uniqueness of the three Python packages that the NCL team supports (PyNIO, PyNGL, and WRF-Python). PyNIO duplicates functionality found in other packages, but its robust support for GRIB needs to be carefully reviewed against other Python-based GRIB readers to make sure no functionality is lost if PyNIO development is ended. PyNGL is unique in its customizability of graphics and its direct support for complex unstructured meshes like MPAS. However, given matplotlib's ubiquity and huge development community, PyNGL also needs to be reviewed for its relevance and longevity factor. WRF-Python provides unique functionality that is not available in other Python packages.

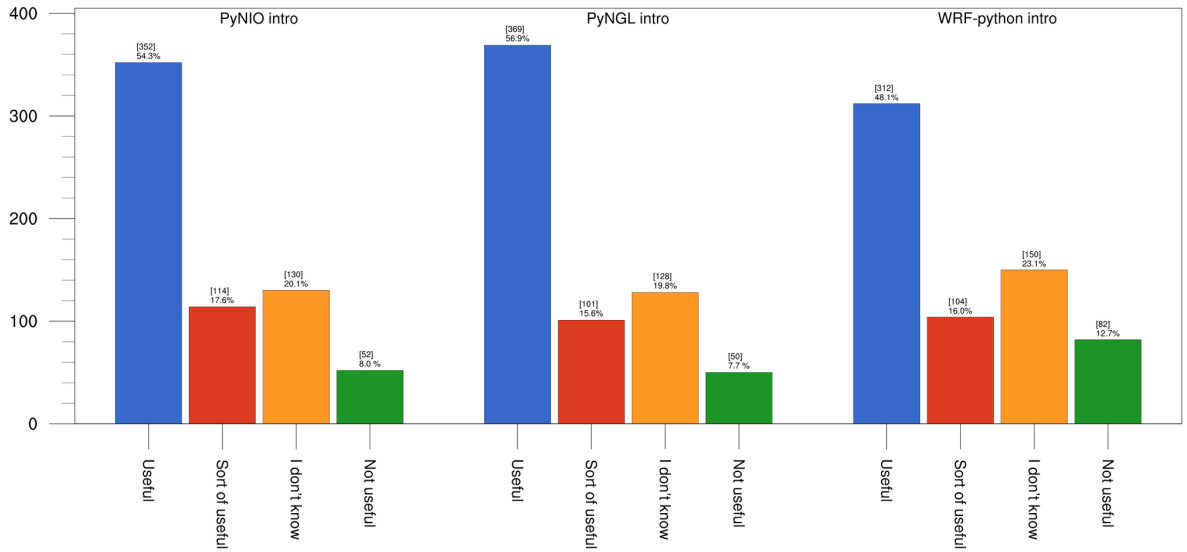
Responses to questions #38 and 39 resulted in over 100 unique Python packages being identified in respondents' post-processing Python workflows. The high number of packages is both a plus and minus for the scientific community. You have an abundance of functionality, but there's a potential lack cohesiveness between the packages (e.g. a common data model) and a lack of a common base of support. Both of these minuses are something that the Pangeo platform is attempting to address, through its endorsement and support of xarray and by providing a community that fosters collaboration around open source tools.

Survey Section 6: Python Training – Conditional

The NCL team considers training to be just as important as software development so the questions around Python training were put into their own section. There were 648 respondents to this section, due to 51 of the original 699 respondents indicating in question #30 that they weren't planning to use Python.

Questions #40-42

This gridded multiple question asked if there would be interest in PyNGL, PyNIO, or WRF-Python training. The NCL team has given 88 NCL extensive training workshops from the period February 2000 to July 2018, and the only training on the Python tools has been through short tutorials at WRF conferences, the NCAR SEA, the annual AMS meeting, and a combined WRF-Python and VAPOR Workshop class at Boise State University in late September 2018.



Comments in later sections of the survey indicated that just about any kind of basic and advanced Python training and across a variety of tools would be widely appreciated.

Question #43

This was an open text box asking respondents to list other types of Python training that would be useful. There were 102 responses, which were broken down into the following categories:

Useful Python Training	Count
Parallelization / big data / cloud	14
Visualization	9
Post-processing model or climate data	8
NCL to Python	6
Any kind of training	5
File I/O	5
Documentation, not training	4
Machine Learning	4
Installation	4
Advanced training	2
Jupyter Notebook	2
Regridding	2
None	12
General comments	16

The parallelization category included mention of dask, xarray, and Pangeo. The visualization category included mention of cartopy, which will eventually replace basemap to do plotting over geographic maps. In the general comments one respondent pointed out that Data Carpentry is developing a set of lessons for use in the atmosphere and ocean sciences (which were released September 2018 [https://carpentrieslab.github.io/python-aos-lesson/]). Below is a sample of the comments received:

> "Any generalized training on dealing with WRF and CESM output—specifically regridding and interpolating it—would be helpful. In particular, training that emphasizes wrf-python and any python-based CESM analysis modules would pique my interest."
>
> "Computational computing using distributed projects like Dask."
>
> "Advanced mapping with cartopy, efficient ways to read and plot large amounts of satellite data with python, proper array handling and manipulation in memory"
>
> "Weather/Climate model data analysis framework using python, in which python data processing procedures should be explicitly realized by python modules. In particular, how to utilize numpy to handle large weather/climate datasets. There are a few climate model data analysis tools such as RCMES, UV-CDAT, etc. However, these tools are a bit hard to follow unless one studies hard about their APIs spending a lot of time on top of the general knowledge of Python."
>
> "Xarray tutorials and other ways to optimize python operations for large data"

"Python training for Machine learning, big data analysis"

"Workflow using python and NCL"

"Tutorial for transitioning from NCL to python"

"I have found that I learn best through exploring materials and working through problems at my own pace, so simply having a greater abundance of materials like Unidata's python workshops' website is ideal"

Section 6: Summary

Based on responses to the first question in this section, there was a high level of interest (48% and higher) for training on PyNIO, PyNGL, and WRF-Python. The open text responses to the second question indicated a wide range of desired training for Python, from beginner's courses on Python, transitioning from NCL to Python, and advanced courses on climate analysis, big data, machine learning, and advanced graphics. The Data Carpentry lessons mentioned in the open text comments could provide an excellent first start in addressing this question.

> "Data Carpentry (http://www.datacarpentry.org/) is developing a set of lessons for using Python in the atmosphere and ocean sciences: https://data-lessons.github.io/python-aos-lesson/"

For transitioning NCL users to Python, the first step will be converting a subset of the hundreds of NCL example scripts to Python and providing those online for people who prefer to learn things on their own time. From this experience the NCL developers should have some excellent materials to put together for NCL-to-Python transition training at NCAR, and through webinars which could be recorded and provided online.

Karin Meier-Fleischer (DKRZ)—a member of the NCL Advisory Panel and the main author of the NCL User Guide—has written a first draft of a "NCL transition to Python" document, which contains includes dozens of NCL-to-Python example scripts.

Survey Section 7: NCL and Python Development Priorities

This section was geared towards both NCL and Python users and was tailored to answering the following questions:

- What users felt the top priorities should be for NCL and Python tools development
- What functionality is missing from NCL and its related Python tools
- What are the top NCL computational routines users want ported to Python
- What categories of NCL website examples should be ported to Python

As mentioned in the Introduction, the questions around NCL functionality were asked because the survey respondents were primarily NCL users who may not know Python well enough to state what functionality was desired or missing. The NCL-centric answers were mostly evaluated in the context of whether the functionality was already available in Python.

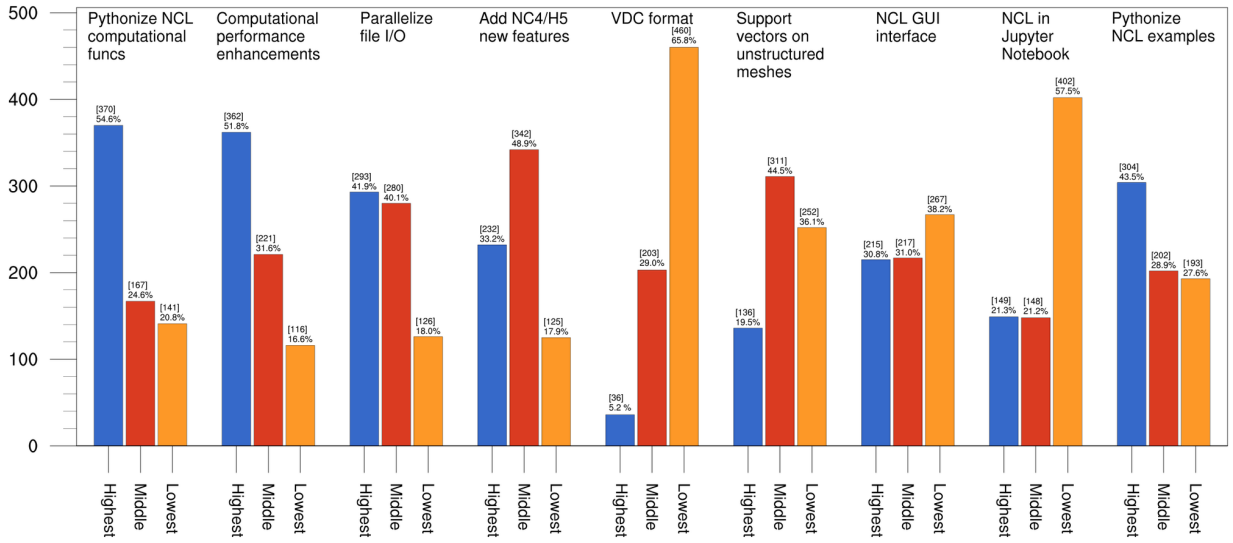
All survey respondents were sent to this part of the survey, which created some confusion for respondents who were mainly Python users that were taking this survey out of curiosity. Their text input was especially valuable to this section.

Questions #44-52

These questions formed a gridded ranking table, in which respondents were required to rank the importance of nine NCL and Python development priorities in the table from 1 to 9, with 1 being the most important and 9 being the least important.

The ranking table caused some frustration with respondents who either didn't realize they had to rank each item uniquely or they didn't care for this restriction. In an attempt to clarify this question, more explanatory text was added to the question after the survey went live. This ended up being a bad decision. Due to survey creator error, the first item in this table— "Pythonizing NCL's computational routines"—was accidentally removed from the table and not corrected until several hours later. This resulted in 21 fewer responses to this item compared to the other eight items in the table. Embarrassing lesson learned: do not edit surveys while they are still live!

In order to make the analysis of these results a little easier to view in a graph, ratings 1-3 were combined into a single "highest" category, ratings 4-6 into a "middle" category, and 7-9 into a "lowest" category.



The table below shows the categories listed in decreasing order of the "highest" combined ratings.

Category	# 1-3 rankings	Percentage
Pythonize NCL's computational routines	370	54.6
Apply performance enhancements to computational routines	362	51.8
Pythonize NCL's examples	304	43.5
Parallelize NetCDF and HDF in NCL / PyNIO	293	41.9
Support newer NetCDF4 / HDF5 features	232	33.2
Create a GUI interface for NCL	215	30.8
Integrate NCL into Jupyter Notebook	149	21.3
Add graphical support for drawing vectors on meshes	136	19.5
Add support for VAPOR's VDC format	36	5.2

Even with the 21 fewer people who were not offered "Pythonize NCL's computational routines" category, it still had the highest rating, followed closely by "Apply performance enhancements to existing routines" and "Pythonizing the NCL examples on the NCL website".

Questions #53-56

This set of four open text questions provided respondents with the opportunity to further comment on the following categories of features they felt should be added to NCL, PyNIO, PyNGL, and WRF-Python:

- Core language
- File I/O
- Computational
- Graphical

Question #53

This question asked about features respondents felt were missing from NCL's core language, which includes features like optional arguments, error handling, and interactivity. There were 223 responses, which was the largest number of responses received to an open text question in this particular section.

Many answers included requests for more than one feature. The top requests were for built-in debugging, better error/exception handling, an "elseif" statement, and optional arguments.

Core Language Features Desired in NCL	Count
built-in debugging	32
better error /exception handling	28
elseif statement *	14
optional arguments	10

better interactivity	8
some form of parallelization (task*, loops)	8
case/switch statement	8
more functions (statistical, regridding, date conversion)	8
better file I/O support	6
better performance / memory management	6
various graphics enhancements	6
complex number support	5
better support for ASCII / CSV files	5
a dictionary-like structure	3
support for UTF-8/special characters	3
support for three-dimensional graphics	3
better support for large data	2
dynamic arrays	2

* Added in NCL Version 6.5.0, released July 2018

There were 54 comments that didn't fall into the above categories and included "I don't know" answers, kudos for NCL, negative statements about NCL (too hard to use, old-fashioned), "I don't use NCL" answers, and "I don't like ranking table" answers. One person stated:

> "Neither NCL nor Python can reach the breadth or depth of data analytics like R, so I use neither for my essential workflow. I hope NCAR will support development of diagnostic tools with R."

This comment was repeated verbatim by the same respondent in seven more open text questions in the survey.

Question #54

This question asked what new file I/O features or data format support respondents felt should be added to NCL and/or PyNIO. There were 136 responses. The top three requests were for some form of parallelization and/or better performance for handling large files, support for GeoTIFF, and better support for advance features in NetCDF4 and HDF5 (e.g. compound types).

File I/O Features Desired in NCL / PyNIO	**Count**
Parallelization / better performance for large files	9
Support for GeoTIFF / TIFF	8
Support for advanced NetCDF/HDF features	7
Better support for ASCII	6
Better support for GRIB2	6
Python 3 support for PyNIO	5
GIS	4
Write GRIB	4
JSON	4
General support for NetCDF	3

VAPOR support	2
BUFR	2
KML	2
Shapefile	2

The remaining 24 responses included singular requests for other data formats support, general comments (checking for CF-compliance, metadata parsing in PyNIO), and "none".

Question #55

This question asked what capabilities or types of functions respondents would like to see added to NCL's computational routines. There were 139 responses. It was harder to group the responses in this question because of the rather wide variety of one-off requests. Here are some of the categories that had multiple responses:

Computational Features Desired in NCL	**Count**
Statistical routines	14
Parallelism / better performance	10
Interpolation / regridding	7
Machine learning / cluster	5
File I/O	5
Plotting	4
Climate metrics / indices	4
Date / time routines	3
Matrix operations	3
Meteorological routines	3
Oceanography	2
Functions for unstructured data	2

There were 31 one-off responses that didn't really fall in the above categories, and 42 "I don't know" or "none" type of responses.

Question #56

This question asked what new features or capabilities respondents would like to see added to NCL's and/or PyNGL's graphics. There were 135 responses.

Graphical Features Desired in NCL / PyNGL	**Count**
Three-dimensional capabilities	10
Simplification	9
More customization	5
Vector enhancements	5
Parallelism / better performance	4
Automated plots	4
PyNGL enhancements	4
Python 3 support	3

Animations	3
Font control	3
Interactivity	3
Color tables	3
Shapefiles	3
Skew-T	3
Cross-section plots	2
KML support	2
GIS support	2
PNG options	2
Scatter plots	2
Satellite plots	2

There were 19 responses with one-off answers, and 42 "I don't know" or "none" type of responses.

Question #57

This question asked what particular examples from the NCL Applications page (http://www.ncl.ucar.edu/Applications/) respondents would like to see Python equivalents of. There were 136 responses that fell into the following categories:

What NCL Examples Needed in Python	Count
All / most of them	20
Maps	10
Plotting WRF	9
Cross-section / vertical plots	8
Regridding / interpolation	6
Contours / isolines	4
Models	4
Three-dimensional graphics	3
Vectors / streamlines	3
Paneling	3
Meteograms	3
Histograms / bar plots	3
Wavelets	3
Regression / trend	3
XY	3
Basics	3
Trajectory	2
MJO	2
Box plots	2
Radar	2
Shapefiles	2
EOF	2

There were 14 responses that requested one-off features or contained general and vague comments. There were 42 responses of "None" or "Don't use" type of answers.

Question #58

This question was a general-purpose open text box giving respondents a final chance to enter features they would like to see in NCL, PyNIO, PyNGL, and/or WRF-Python. There were 111 responses that fell into a variety of categories. The table below is a sample of some of the requests:

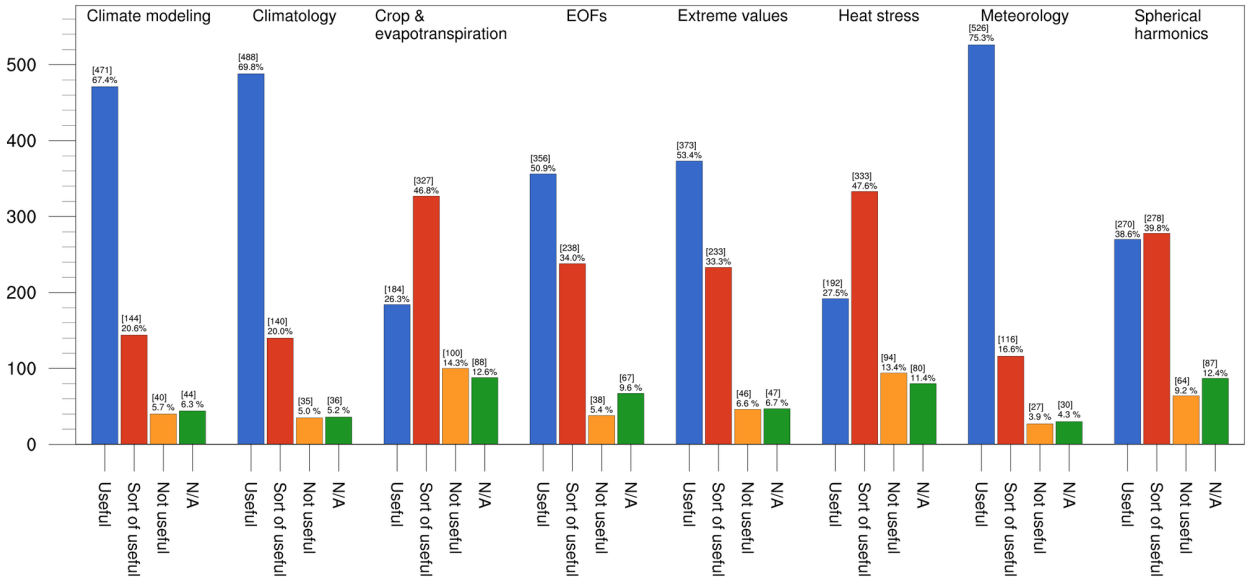
Other Features Desired in NCL and Python Tools	**Count**
Support for Python 3	12
WRF-Python • better xarray integration • better support for cross-sections • simplified plotting capability • more documentation / examples	11
Installation • issues with conda installation • want LINUX-based package manager	10
Graphics • improve speed of plotting • more dash patterns • more color tables and standardized color bars • Chinese character support • integration with Vapor • better inline support for animation	9
Computational routines • performance enhancements • more functions that work with regional data • convert NCL computational routines to Python • more statistical routines	7
PyNGL • same suite of graphical enhancements as NCL • more NCL-based regridding and computational routines • integration with xarray	5
File I/O • better handling of large data • support for geotiff	2

Questions #59-66

This was a gridded multiple-choice question asking respondents to indicate how useful it would be to them if particular categories of computational routines in NCL were ported to Python. As stated in the question itself, this task is one of the NCL team's highest priorities.

In order to get a better idea of what features respondents considered "Useful" or "Very useful", these two ratings were grouped into one rating called "Useful". Likewise, the "Moderately useful" and "Slightly useful" ratings were combined into a "Sort of useful" field.

The top three categories were meteorology, climatology, and climate modeling functions.



Usefulness of Computational Routines Ported to Python		
Category	**Count**	**Percentage**
Meteorology	526	75.3
Climatology	488	69.8
Climate modeling	471	67.4
Extreme values	373	53.4
EOFs	356	50.9
Spherical harmonics	270	38.6
Heat stress	192	27.5
Crop & evapotranspiration	184	26.3

Question #67

This question was a general-purpose open text box giving respondents a chance to indicate other types of functions they would like to see ported to Python. There were 49 responses to this question, of which 20 responded with some form of "none" or "don't use Python". There were a few general comments about being careful not to duplicate Python efforts that were already available, like the "eofs" and scikit-learn packages for calculating EOFs. One respondent offered to help with this effort and included a link to a climate-indices package that might be worth checking out. Of the rest, the only category that stood out was regridding functions.

Below is a sample of some of the comments entered:

> "I find ESMF regridding in NCL very useful, missing that in pyngl"
>
> "As I said in other responses, regridding/interpolation options are a huge priority for me!"
>
> "The only two things I've had to use NCL for is to read GRIB files and pot_vort_isobaric. Those are my priorities."
>
> "Additional meteorology functions from GEMPAK (some are already available, and others have been converted recently) would allow this software to become standard educational materials in most junior/senior level synoptic-dynamic coursework and help educators provide students with a reason to learn a more broadly used language like Python as well."
>
> "I would very much like to see the ported functions / demos built around xarray and the Pangeo suite of tools wherever possible. This is clearly becoming the lingua franca of geo-scientific computing, and would be absolutely invaluable."
>
> "Do not waste time duplicating functions that already exist in multiple standard python libraries. For example, EOFs can already be calculated using functions in scikit-learn, numpy, and scipy.linalg, and probably others."

Section 7: Summary

This section of the survey was particularly difficult to summarize due to the wealth and wide variety of feature requests in the text comments (questions #53-57 resulted in 633 text responses). The text comments responses were grouped by categories and then counted to see which topics came up the most frequently. From this and the answers to the multiple-choice questions, the top three priorities with a Python focus include:

Pythonize NCL's computational routines and apply performance enhancements

NCL has hundreds of computational routines, many of which are unique for climate and weather and may not be available in Python. The results from questions #59-66 will be used to help determine which group of computational routines to focus on first. The NCL team will engage

the Pangeo community to further narrow down which functions are critical and which ones are already available in Python. Pangeo is an NSF EarthCube funded effort that provides an "open source scientific Python ecosystem for ocean / atmosphere / land / climate science" and is focused on providing tools and support for handling petabyte-scale datasets on HPC and cloud platforms.

Extend and improve WRF-Python

WRF-Python received a high number of specific mentions in comments throughout the NCL survey and clearly provides unique and crucial functionality in the Python community. Some of the enhancement requests for this package include better integration with xarray, more flexibility for calculating cross-sections, and a simplified plotting interface. Finally, there were some requests around documentation, including a need for more examples and a contributor's guide.

Pythonize NCL's examples

The examples on the NCL website provide over a thousand scripts that use NCL for a variety of file I/O, computational, and graphical tasks and has been touted frequently by users as a critical resource for their post-processing workflows. A suite of examples like this for Python would be a highly unique resource and would go a long way in helping NCL users transition to Python. It also provides an excellent opportunity for open development contributions. As mentioned in the "Python Training" summary, Karin Meier-Fleischer has already created a first draft of a "NCL transition to Python" document.

Update and evaluate PyNIO and PyNGL

Many respondents complained that these tools were not available under Python 3 and that they had serious installation issues. The NCL team has since ported these tools to Python 3 and released new versions on the conda-forge channel. The comments on installation issues were a bit unexpected, as the NCL team feels these issues have stabilized in the last year since putting the tools under conda-forge.

The evaluation part of this priority comes from the NCL team not wanting to duplicate efforts by developing Python packages with similar functionality as other Python packages.

As an example, the "xarray" package has been widely adopted by the Python climate and weather community for its powerful data model that provides easy access to metadata. This model and metadata access is similar to NCL's (and subsequently PyNIO's), which was touted several times in this survey as a crucial feature for NCL users. Also similar to NCL and PyNIO, xarray provides a file I/O interface to handling the NetCDF and GRIB file formats. The xarray GRIB support is currently handled via a backend to PyNIO, but ECMWF (European Centre for Medium-Range Weather Forecasts) is developing a Python GRIB reader called "cfgrib" that is being discussed as a likely replacement for PyNIO in xarray. The NCL team would be happy to relinquish GRIB support so the experts at ECMWF who have far more experience with this format.

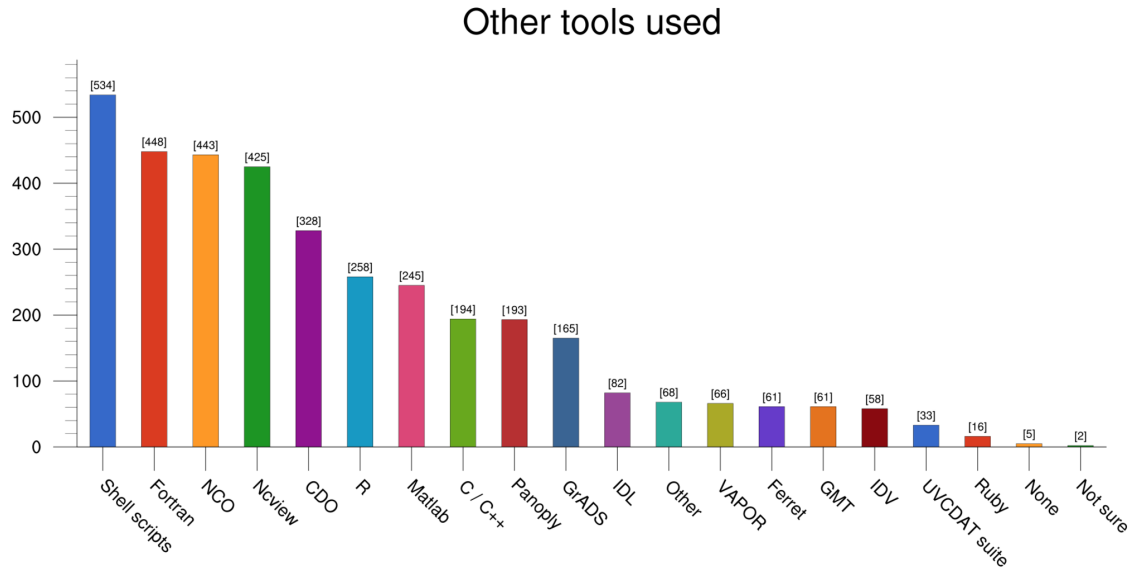
As a second example, matplotlib is a highly popular and widely used two-dimensional Python plotting package that has a long history and strong development support in the Python community. It provides a lot of the same graphical functionality as PyNGL, making it hard to justify continued support of PyNGL. The interfaces and capabilities of the two packages are quite different, however. NCL users would likely prefer to transition to PyNGL for graphics since it provides the same type of interface and high level of customizability as NCL graphics. The NCL team needs to evaluate whether PyNGL is still a viable package and if and how it can coexist with and complement matplotlib.

Survey Section 8: Other Tools

This section contained two questions tailored to finding out what software beyond NCL and Python respondents use in their scientific processing workflows and why they are important. These questions were asked out of curiosity, to identify other tools that may provide unique or crucial functionality not available in NCL or Python.

Question #68

This was a required multiple selection question asking respondents to check all the tools they use in their scientific workflow. There were 3,685 total boxes checked. The bar chart below ordered the tools in descending order of usage.



There were 68 responses in the "other" field which are tabulated below. Perl had the highest number of multiple mentions (15).

List of Tools Mentioned in "Other" Field						
arcgis	aspect	c#	cdo	climaf	cuda	delphi
eccodes	excel	gdal	gempak	globalmapper	gnuplot	googleearth
hdfview	houdini	igor	imagemagick	ive	java	javascript
jquery	julia	jupyter notebook	lazarus	lisp	maxima	mcidas
mcidas-v	mcidas-x	met	metpy	mysql	ncl	octave
pangeo	paraview	pascal	pdl	perl	perl netcdf	php
prolog	pyart	python	qgis	raob	scilab	soloii
sql	tecplot	visit	wgrib2			

Question #69

This was an open text box giving respondents a chance to espouse on why particular tools were important to their workflow. Here's sample of the 68 responses received:

> "NCO is important because of the ncregrid script."
>
> "Ferret is a very good 'quick and dirty' plotter but making sophisticated plots is a chore; I use Imagemagick to panel plots; it is often much more easier to create them separately and then paste them together."
>
> "Fortran custom code still vital, shell scripts as glue, R increasingly important"
>
> "Excel is simple to plot the regression line compared to NCL, and it is easy to get the outputs of some statistics (although NCL has improvements in the recent versions)."
>
> "Ncview is easy to have a quick look of some parameters. CDO and NCO are easier (much simple scripts) to handle netcdf. CDO is also more convenient for multi-tasking (say, interpolating multiple netcdf at the same time)."
>
> "I heavily use bash and ncl combinations in my workflow. I also rely on fortran wrappers for my more computational-heavy functions tied into NCL code. I would love to see the wrapper examples updated to a more recent fortran syntax vs the F77 style. Maybe that's not possible, but it seems doable?"
>
> "cdo -> fast processing large data files, shell scripts -> use ncl & cdo codes on supercomputers, etc"
>
> "I can't live without NCO. R I use but only if I don't have a choice. Everything I do is wrapped in shell scripts. Ncview for a quick view of results from the command line. Panoply for telling students/collaborators who aren't using unix how to view my data.

GMT has capabilities that ncl still doesn't have (easily going back and forth between raster and vector worlds)."

Section 8: Summary

There were over 80 tools mentioned when the explicit tools listed in question #68 were combined with the additional tools listed in the "Other" box.

Tools like ncview, panoply, GrADS, R, and Excel were mentioned explicitly for their ability to produce a quick view of a respondent's data or to generate simple plots that didn't require as much work as an NCL plot does.

Tools like NCO and CDO were mentioned for their ability to do quick operations across files from the UNIX command line, without having to write code or use a GUI interface.

A number of the tools listed were GUI-based, which are particularly useful for people who don't want to learn to program or need to have direct and immediate interactivity with their data.

The takeaway from this section is something that it can take a number of tools to accomplish a series of post-processing tasks. Many times, the tools used are simply the ones that users are most comfortable with.

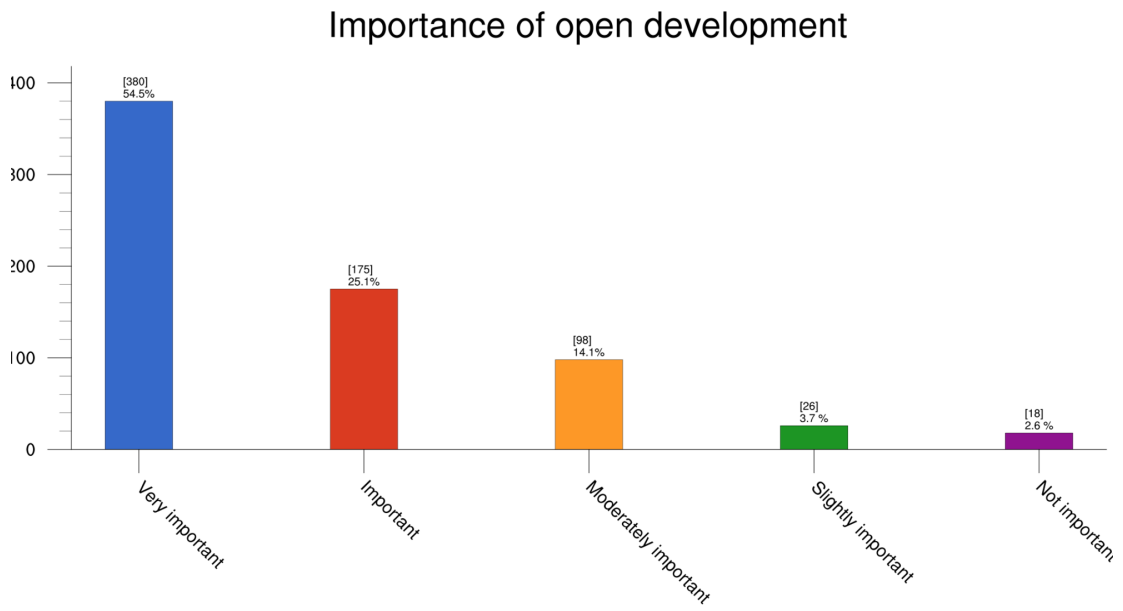
Survey Section 9: Open Development

Based on the strong recommendation of the CISL Advisory Panel, the NCL team is moving its software stack from an open source model to a more open development model. The purpose of this section of the survey was to introduce respondents to the idea of open development and get their thoughts on it.

Open development is a software development model in which external users of an open source software package are encouraged to help develop the software and define its development path. This is in contrast to a closed development model where an internal group of developers sets the priorities and develops the software.

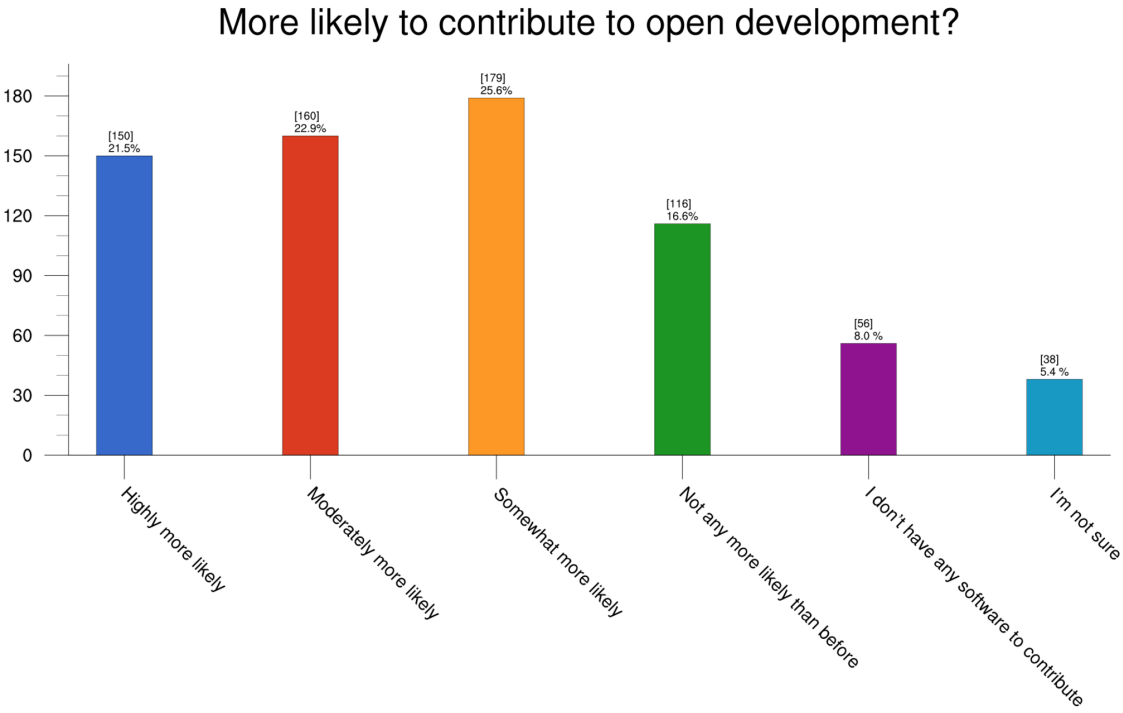
Question #70

This was a required multiple-choice question asking respondents how important they felt it was for NCL and its Python tools to be moved to a more open development model. From the bar chart below, 79.6% said it was either important or very important to consider, 17.8% said it was moderately or slightly important, and 2.6% didn't consider it important at all.



Question #71

To gauge whether respondents actually had software to contribute, this question asked how much more likely they would be to contribute to the NCL software stack if it was moved to a more open development model. There were 21.5% responses in the "highly more likely" category, and 48.5% in the "moderately or somewhat more likely" category. On the other end of the spectrum, 16.6% were neutral, while 13.4% said they either didn't have software to contribute or they weren't sure.



Question #72

This was an open text box allowing respondents to comment on their thoughts about open development. There were 72 responses of which 53 fell into one of the following categories:

Categories of Open Development Comments	Count
Supportive of open development	17
Support open development but code needs quality assurance	12
I would like to contribute	8
Things are fine the way they are	6
I don't have anything to contribute	4
Have mixed feelings about open development	3
Prefer closed development	3
General comments on other things	19

Here's a sample of some of the responses:

> "I am okay with submitting functions and having the NCL dev team make tweaks to ensure its consistency with the rest of the NCL functions before releasing it to the public. But perhaps some functions could be pre-released to the public in the form of stand-alone files to download and load on their own, once they're deemed ready for inclusion by default in a future NCL version."
>
> "I've wanted to contribute to wrf-python in the past but couldn't find a contribution guide."
>
> "I've done some development for CVDP and actually quite like the curation that Adam Philips is doing"
>
> "Vital to have a vetted, stable version, but having a wider contrib and access to the stable code to modify would be great."
>
> "Some level of control is required in order to retain balance"
>
> "Pyngl is a slightly isolated package. Better integration with other tools would increase usage and lead to more involvement."
>
> "Open development is a great feature NCAR provides to develop the computational facilities for data analysis and visualization. I would like to thank NCAR for that because I have been participating in forums and have benefited a lot from the support teams and from the wider community."

"Open development can lead to less bugs and a much wider toolbox suite but naming conventions and calling conventions need to be self-consistent and enforced. Not a free-for-all as with R packages. NCL already had problems in the past in that too little care was taken on this, meaning that unlike IDL, FORTRAN or python, it was always impossible to simply code in NCL without having the webpage open to refer to, the function names and calling structures were inconsistent across modules. Thus if an open model is to be adopted a strict set of conventions need to be adopted for contributions that are policed carefully."

"NCL is a tool I use regularly for my research tasks and it is very important to me that the main developers keep control of what kind of tools are added."

Section 9: Summary

The NCL user community has contributed enormously to the software through code, documentation, examples, and consulting support. These contributions have usually been done through the "closed method" of back-and-forth emails and with the core NCL team doing the final legwork of incorporating the contributions either into a software release or the website. Moving to an open development model with issue tracking being be handled through a platform like GitHub, and with supporting contributors' guide documents will make the process far more transparent and provides immediate and proper credit to the contributors.

The responses to this section of the survey were in strong support of open development, but with some kind of quality assurance that the software is stable. There were also suggestions of developing a style guide for best practices and reminders that some respondents could benefit training on how to contribute. Many of these ideas were incorporated into the roadmap in the "NCL and the Pivot to Python" document.

Survey Section 10: Final comments

Question #73

The last question was an open text box giving people their final chance to enter whatever comments they wanted. There were 147 responses that fell into the following categories:

Type of Comment	Count
Kudos	104
Criticisms of NCL	9
General comments about Python	9
General comments about NCL	7
General comments	7
No final comments to add	6
Training and support	5

The criticisms of NCL included being hard to use, slow to run, inconsistency in function names, and/or buggy. The Python comments were mostly about graphics and needing more features or better support for existing modules (PyNGL, cartopy, and matplotlib). The NCL comments included some one-off requests for features and getting better performance (both in the build process and usage of the software). The training and support responses included people asking for specific workshops to be offered. Most of these comments were echoed in other parts of the NCL survey and there wasn't any new or unexpected information added.

Conclusion

The NCL survey was conducted mainly to gather data around the "pivot to Python" discussion and to collect opinions on moving the software to a more open development model.

The findings of this survey show that a high number of active NCL users (88.7%) have either already adopted Python in their scientific workflows or are considering it in the future. Many of the top features requested for NCL are implemented in Python and associated Python packages, making a strong case that a pivot to Python is a sound decision.

For functionality not available in Python, the survey results helped identify crucial categories of functions, like meteorology, specialized climate modeling, regridding, and interpolation, that are unique and need to be ported to Python.

It also highlighted the fact that in order for PyNIO and PyNGL to stay relevant, they need to be immediately updated to support Python 3 (at a minimum). Longer term, these packages must be evaluated for their viability in the Python community.

WRF-Python provides unique functionality in the Python community. Comments from the survey indicated that either more plotting examples or a simplified plotting interface similar to NCL's WRF-NCL plotting library would be useful.

The survey responses to the support and training related questions reinforced what the NCL team already knew: support, training, and documentation are critical for users, and you need a mix of media in order to best meet the needs of a global user base. For NCL users in particular, the pivot to Python is going require not only training in Python itself, but in how to transition from NCL to Python.

Finally, in the section on open development, there was a high level of support for going in this direction with caution around ensuring code stability, the need for contributors' documents, and using this as an opportunity to develop some coding style guides and best practices.

The results from this survey, combined with the NCL Advisory Panel's report, have been used to develop a two-year software roadmap around porting NCL functionality to Python, putting more focus on the existing NCL-based Python tools, helping NCL users with the transition to Python, and moving the software to an open development model. A roadmap and a further discussion on the decision to adopt Python can be found in the "NCL and the Pivot to Python: Discussion and Roadmap" document.

Appendix A: General Survey Information

Google Forms was used to create the survey, which consisted of 41 questions:

Type of question	# of questions
Multiple Choice	11
Multiple Selection	8
Gridded Multiple Choice	5
Gridded Ranking Table	1
Text Box	16
Total	**41**

The gridded questions and ranking table contained multiple questions in one, so altogether there were actually 73 questions. See Appendix B for the full list of questions.

A descriptive email and a link to the survey was emailed to addresses culled from the following groups of people:

1. Subscribers to ncl-talk, ncl-install, pyngl-talk, wrfpython-talk, pyaos email lists
2. Attendees of NCL workshops
3. People who had downloaded NCL and or one of its Python packages in the last two years.

Yet Another Mail Merge (YAMM) was used to manage the 7300+ emails that were sent. YAMM tracked which emails had bounced, were opened, were clicked, or were never opened. From this information, a second set of emails were culled from people who had not clicked the original email, resulting in a reminder message being sent on May 22 to 5800+ email addresses. In all, over 6500 emails were successfully sent.

A link to the survey was also included on the NCL website home page.

The survey contained a mix of NCL, Python, and other tools questions split over ten sections:

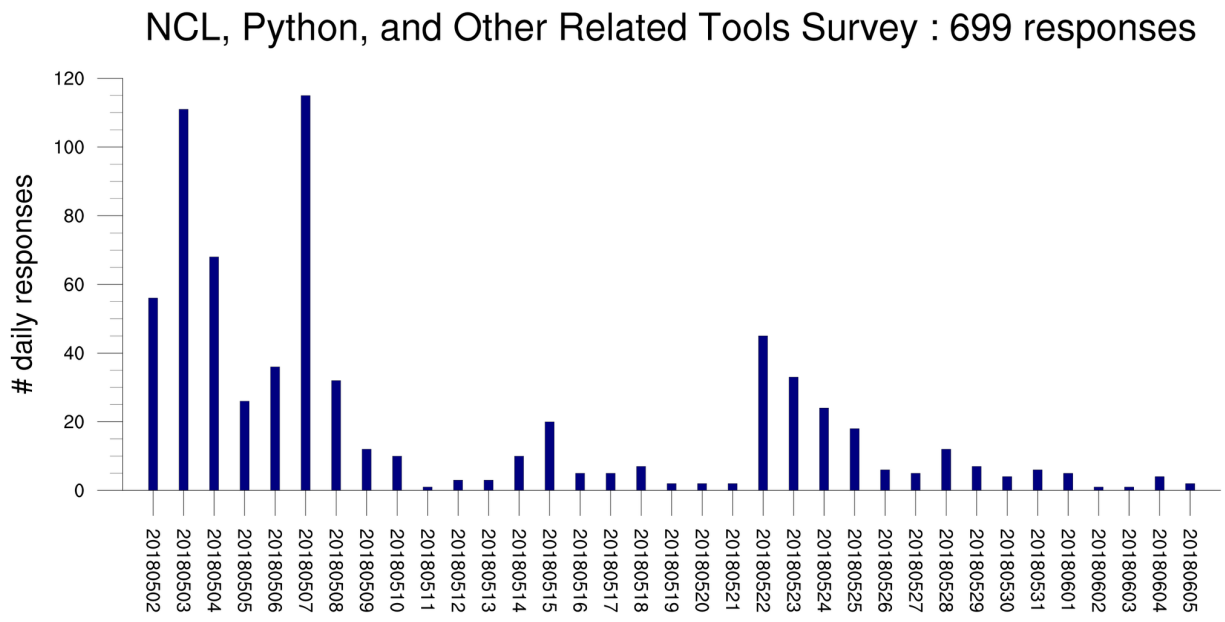
1. General Demographics
2. NCL Demographics
3. NCL Usage
4. Python Demographics
5. Python Usage
6. Python Training
7. NCL and Python Development Priorities
8. Other Tools Usage
9. Open Development
10. Final comments

The “NCL Usage”, “Python Usage”, and “Python Training” sections were conditional, meaning that these sections might have been skipped for some respondents depending on their answers to the “NCL Demographics” and “Python Demographics” sections.

All of the multiple-selection and multiple-choice questions required an answer, while all but one of the open text boxes were optional. Some of the multiple-choice or -selection questions included an “Other” field, which if checked required a typed answer. The one open text box that was not optional was the question about the primary country of residence; responders were told to enter “--” if they didn’t want to answer this question.

Most of the graphics included in this document were generated by NCL, using data from a CSV file exported by Google Forms. Google Forms did provide its own graphics, but they were hard to customize and, in some cases, you couldn’t view the full graph on the screen without scrolling. Using NCL allowed for more customization on how to analyze and view the data. The Google Forms color scheme was adopted for the NCL graphs.

The Google Forms CSV file included a time stamp field, which was used to generate the bar plot below that hows the number of daily responses received. Before a reminder email was sent on May 22, there were 526 responses. The small spike on May 15 may have been due to the survey being advertised to the “pyaos” list, which is an email list for Python users in the atmospheric and oceanic communities. The spike seen on May 22 was likely due to the reminder email that was sent that day.



Appendix B: Table of Survey Questions

This table contains all of the survey questions grouped by sections. The number of respondents to each section is included.

Legend	
MC	Multiple Choice
MS	Multiple Selection
GMC	Gridded Multiple Choice
GR	Gridded Ranking Table
TB	Text Box
A red asterisk (*) indicates a required question.	

General Demographics		**# Respondents: 699**
Q1*	What best describes your position?	MC
Q2*	What type of institution are you mainly associated with?	MC
Q3*	What is your main area of study or research?	TB
Q4*	Check all the types of data you work with.	MS
Q5*	What is your primary country of residence?	TB
NCL Demographics		**# Respondents: 699**
Q6*	How many people at your site use NCL?	MC
Q7*	What best describes how frequently you used NCL in the last year?	MC
NCL Usage and Development (conditional section)		**# Respondents: 591**
Q8*	When did you first start using NCL?	MC
Q9*	What is your level of proficiency with NCL?	MC
Q10*	Which statement best describes how you use NCL?	MC
Q11*	Select all the versions of NCL are you currently using.	MS
Q12*	Check all the types of data you read, analyze, and/or plot with NCL.	MS
Q13-22*	Indicate how important the following features in NCL are to you.	GMC
Q23	Use the text box below if you want to mention other features or aspects of NCL that are important or useful to you.	TB
Q24-28*	Indicate how useful the following have been for you in getting support for NCL	GMC
Q29	Use the text box if you want to indicate other ways you get support for NCL or comment on support in general.	TB
Python Demographics		**# Respondents: 699**
Q30*	What best describes your use of Python in the last year for scientific data analysis?	MC
Python Usage (conditional section)		**# Respondents: 631**
Q31*	What is your level of proficiency with Python?	MC
Q32*	Check which versions of Python you are currently using.	MS
Q33*	Select all distributions of Python you have used or are currently using.	MS
Q34*	Select all the ways Python is used in your scientific workflow.	MS
Q35-37*	PyNGL, PyNIO, and wrf-python are Python interfaces developed by the NCL team at NCAR that provide Python modules to NCL's file I/O, graphical, and WRF-NCL capabilities. How important are these packages in your research?	GMC
Q38	If you are using any of the above three packages, please indicate what it's used for.	TB
Q39*	What other Python modules do you use?	MS

Python Training (conditional section)		**# Respondents: 648**
Q40-42*	If NCAR were to offer Python training for tools developed by the NCL team, how useful would the following topics be for you?	GMC
Q43	Use the text box to indicate other Python training you would be interested in.	TB
NCL and Python Development Priorities		**# Respondents: 699**
Q44-52*	Rank these high-level development priorities from 1 to 9, with 1 being the most important and 9 being the least important.	GR
Q53	NCL core language: what are some important features you feel are missing?	TB
Q54	NCL and PyNIO file input / output: what new features and/or data formats would you like to see supported?	TB
Q55	NCL suite of computational routines: what new capabilities or types of functions would you like to see added?	TB
Q56	NCL / PyNGL graphics: what new features or capabilities would you like to see added?	TB
Q57	There are over 1,700 NCL example scripts on the NCL Application Examples web page. Are there any particular types of graphics you would like to see equivalent Python examples of?	TB
Q58	Feel free to comment on any other features or capabilities you would like to see implemented in NCL, PyNIO, PyNGL, and/or wrf-python.	TB
Q59-66*	One of the highest priorities is to port a subset of NCL's computational routines to Python. Please indicate how useful these groups of functions would be to you in Python.	GMC
Q67	Use this text box if you want to elaborate on other functions you want ported to Python.	TB
Other Tools		**# Respondents: 699**
Q68*	Indicate all the software tools or languages you have used with some regularity in the last year.	MS
Q69	Use this text box if you want to elaborate on any packages you selected and why they are important in your workflow.	TB
Open Development		**# Respondents: 699**
Q70*	How important is it to you that NCAR support open development of NCL and related Python tools to facilitate user contributions?	MC
Q71*	How much more likely would you be to contribute software, if NCAR moved to a more open development model compared to the current more closed development model?	MC
Q72	Use the text box if you have any comments you'd like to make about open development.	TB
Final Comments		**# Respondents: 699**
Q73	Final comments	TB