

# RESEARCH WORKFLOW

ZACHARY LABE (@ZLABE)



# IS THERE A REPRODUCIBILITY CRISIS?



“ “ Today’s weather or climate scientist is far more likely to be debugging code written in Python... than to be poring over satellite images or releasing radiosondes. ” ”

# A MINIMUM STANDARD FOR PUBLISHING COMPUTATIONAL RESULTS IN THE WEATHER AND CLIMATE SCIENCES

BY DAMIEN IRVING

A procedure for publishing reproducible computational research is described that could be adopted as a minimum standard by journals in the weather and climate sciences.

The rise of computational science has led to unprecedented opportunities in the weather and climate sciences. Ever more powerful computers enable experiments that would have been considered impossible only a decade ago, while new hardware technologies allow data collection in even the most inaccessible places. To analyze the vast quantities of data now available to them, modern practitioners—most of whom are not computational experts—use an increasingly diverse set of software tools and packages. Today's weather or climate scientist is far more likely to be found debugging code written in Python,

MATLAB, Interactive Data Language (IDL), NCAR Command Language (NCL), or R than to be poring over satellite images or releasing radiosondes.

This computational revolution is not unique to the weather and climate sciences and has led to something of a reproducibility crisis in published research (e.g., Peng 2011). Most papers do not make the data and code underpinning key findings available, nor do they adequately specify the software packages and libraries used to execute that code. This means it is impossible to replicate and verify most of the computational results presented in journal articles today.



# WCRP

World Climate Research Programme

# GROWING DATA



NCAR  
UCAR

CISL

Computational & Information Systems Lab



**WCRP**

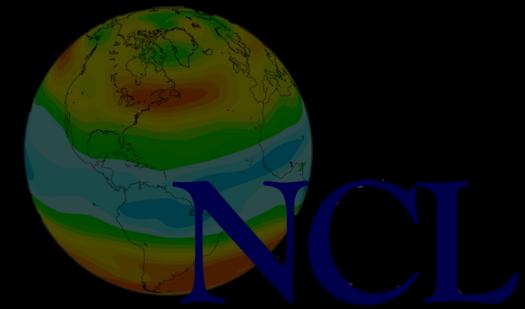
World Climate Research Programme

NCAR  
UCAR

**CISL**

Computational & Information Systems Lab

**ESGF**  
*Earth System Grid Federation*



# GROWING TOOLS



figshare

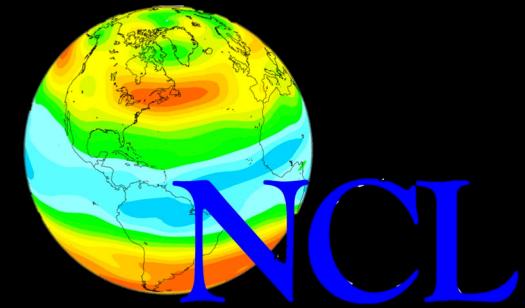


python™

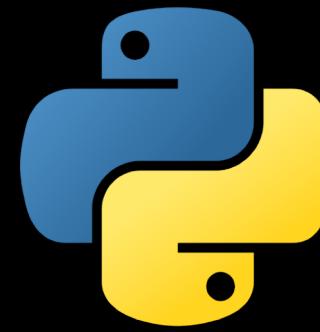
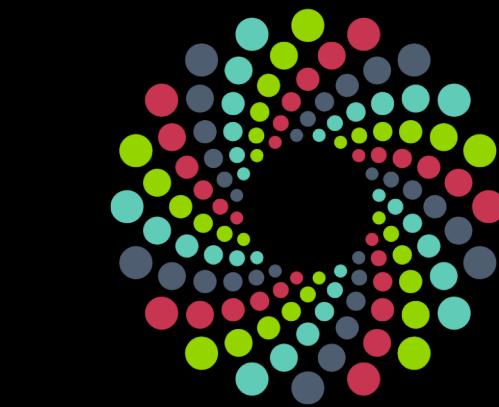
arXiv.org  
open access  
to the world's  
science



zenodo



arXiv.org  
open access  
to the world's  
science



ANACONDA

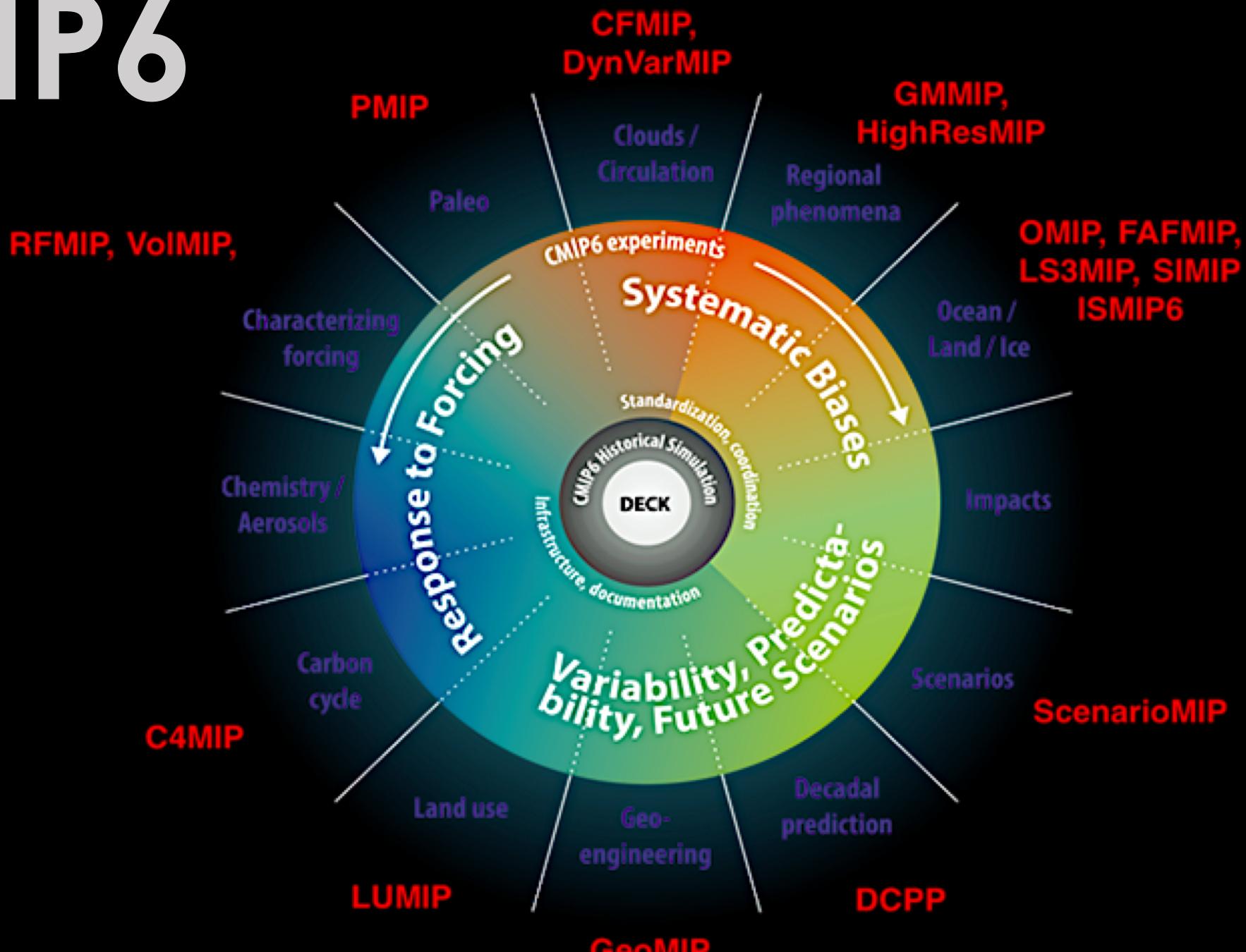


figshare

python™

“ “ It helps to reduce the shame many scientists have about the quality of their code (i.e., they see that their peers are no “better” at coding than they are)... , ,

# CMIP6



# WE HAVE ALL DONE IT...

Data is available from  
the corresponding author  
upon request

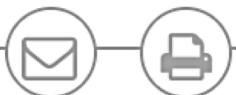
# WE HAVE ALL DONE IT...

Code is available from  
the corresponding author  
upon request



# Increasing “Do You Expect Me to Just Give Away My Data?” **number of code and data policies**

The Editor-in-Chief of *JGR: Oceans* explains why the new AGU data policy is important for the rigor and long-term security of scientific research.



# “Do You Expect Me to Just Give Away My Data?”

The Editor-in-Chief of *JGR: Oceans* explains why the new AGU data policy is important for the rigor and long-term security of scientific research.



# “Do You Expect Me to Just Give Away My Data?” **This is a good thing**

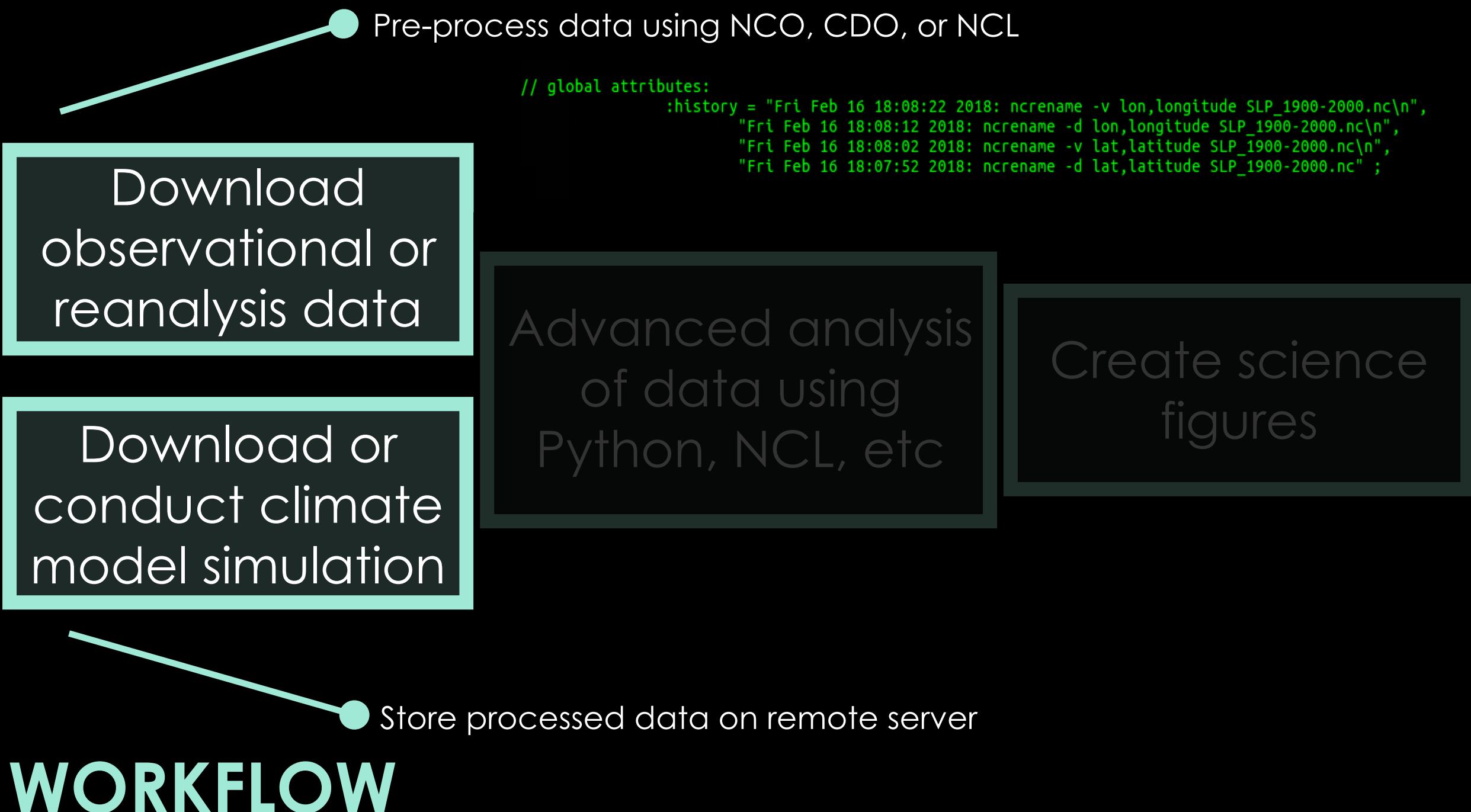
The Editor-in-Chief of *JGR: Oceans* explains why the new AGU data policy is important for the rigor and long-term security of scientific research.

# so what can we do?

```
zlabe@hrisey:~/green/simu/FPOL/monthly$ ls
ALB_1900-2000.nc          DEPF_1900-2000.nc        fourier_wavenumber_decomposition.proc   OMEGA_1900-2000.nc      SF850_1900-2000.nc      TP_1900-2000.nc      Z300_1900-2000.nc
calc_eady_midtrop.proc     divergent_wind.ncl       FPOL_1900-2000.nc        P_1900-2000.nc      SHFLX_1900-2000.nc      U10_1900-2000.nc     Z300xwave1_1900-2000.nc
calc_global_ave.proc       DU_1900-2000.nc         FSDS_1900-2000.nc       prog                  SIC_1900-2000.nc      U_1900-2000.nc      Z300xwave2_1900-2000.nc
calc_QBOindex.proc         DV_1900-2000.nc         FSNS_1900-2000.nc       PV200_1900-2000.nc    SLP_1900-2000.nc      U200_1900-2000.nc    Z30_1900-2000.nc
calc_QBOregion.proc        eady_growth_rate.ncl   FSNT_1900-2000.nc       PV850_1900-2000.nc    SST_1900-2000.nc      U300_1900-2000.nc    Z500_1900-2000.nc
calc_xwave_var.proc        EGR_1900-2000.nc        FSUS_1900-2000.nc       QBO_1900-2000.nc      stationnary_waves.ncl U30_1900-2000.nc      Z500xwave1_1900-2000.nc
change_coords_3D.ncl       EGR_500_850.nc         GEOP_1900-2000.nc       QBO_neg_FPOL.txt    streamfunction.ncl  U400_1900-2000.nc      Z500xwave2_1900-2000.nc
change_coords.ncl          EPY_1900-2000.nc        GEOPX_1900-2000.nc     QBO_non_FPOL.txt   SWE_1900-2000.nc      U500_1900-2000.nc      Z50_1900-2000.nc
CLDHGH_1900-2000.nc        EPZ_1900-2000.nc        GEOPxwave1_1900-2000.nc raw                  T2M_1900-2000.nc      V_1900-2000.nc
CLDLow_1900-2000.nc        extract.proc          GEOPxwave2_1900-2000.nc RNET_1900-2000.nc      T850_1900-2000.nc      V200_1900-2000.nc
composite_WACCM_QBO.proc  FLDS_1900-2000.nc       GEOPxwave_all_1900-2000.nc Rossby_wave_source.ncl T925_1900-2000.nc      VT_1900-2000.nc
compute_3d_waf.proc        FLNS_1900-2000.nc       LHFLX_1900-2000.nc     make_continuous_file.proc TEMP_1900-2000.nc      VT850_1900-2000.nc
compute_EP-flux.proc      FLNT_1900-2000.nc       make_continuous_file.proc RWS_1900-2000.nc      temp_potentielle.ncl WDF_1900-2000.nc
compute_meridional_heat...  FLUS_1900-2000.nc       MHF100_1900-2000.nc    SF200_1900-2000.nc    THICK_1900-2000.nc    Z200_1900-2000.nc
zlabe@hrisey:~/green/simu/FPOL/monthly$ cd ..
zlabe@hrisey:~/green/simu/FPOL$ ls
daily monthly
zlabe@hrisey:~/green/simu/FPOL$ cd daily
zlabe@hrisey:~/green/simu/FPOL/daily$ ls
calc_global_ave.proc      FPOL100  FPOL16  FPOL21  FPOL27  FPOL32  FPOL38  FPOL43  FPOL49  FPOL54  FPOL6  FPOL65  FPOL70  FPOL76  FPOL81  FPOL87  FPOL92  FPOL98
change_coords_3D.ncl       FPOL11  FPOL17  FPOL22  FPOL28  FPOL33  FPOL39  FPOL44  FPOL5  FPOL55  FPOL60  FPOL66  FPOL71  FPOL77  FPOL82  FPOL88  FPOL93  FPOL99
extract.proc                FPOL12  FPOL18  FPOL23  FPOL29  FPOL34  FPOL4  FPOL45  FPOL50  FPOL56  FPOL61  FPOL67  FPOL72  FPOL78  FPOL83  FPOL89  FPOL94  make_continuous_file.proc
fourier_wavenumber_decomposition.proc  FPOL13  FPOL19  FPOL24  FPOL3  FPOL35  FPOL40  FPOL46  FPOL51  FPOL57  FPOL62  FPOL68  FPOL73  FPOL79  FPOL84  FPOL9  FPOL95  prog
FPOL1                      FPOL14  FPOL2  FPOL25  FPOL30  FPOL36  FPOL41  FPOL47  FPOL52  FPOL58  FPOL63  FPOL69  FPOL74  FPOL8  FPOL85  FPOL90  FPOL96  raw
FPOL10                     FPOL15  FPOL20  FPOL26  FPOL31  FPOL37  FPOL42  FPOL48  FPOL53  FPOL59  FPOL64  FPOL7  FPOL80  FPOL86  FPOL91  FPOL97
zlabe@hrisey:~/green/simu/FPOL/daily$ cd FPOL100
zlabe@hrisey:~/green/simu/FPOL/daily/FPOL100$ ls
FPOL100_day.nc            GEOP_100.nc        GEOPxwave2_100...      TEM_100.nc      U_100.nc      V_100.nc
GEOP_100_mean.nc           GEOPxwave1_100.nc  GEOPxwave_all_100...  Umean.nc      Vmean.nc      Z500_100.nc
zlabe@hrisey:~/green/simu/FPOL/daily/FPOL100$ cd ..
zlabe@hrisey:~/green/simu/FPOL/daily$ more make_continuous_file.proc
#!/bin/bash
#
# Group daily outputs of a simulation together by year and
# fill the missing values for missing days

NENS=100
VAR="GEOP"

KEY=1
if [[ $KEY == "1" ]]; then
rm -f tmp*.nc part*.nc
for i in $(seq 1 $NENS); do
    if [[ $i == "1" ]]; then
cdo selmon,9,10,11,12 FITS{i}/{$VAR} ${i}.nc tmp.nc
```



# WORKFLOW

Download observational or reanalysis data

Download or conduct climate model simulation

Create a working directory and a backup directory for project

Advanced analysis of data using Python, NCL, etc

Create science figures

Use of GitHub for version control with 1 repository per project

# WORKFLOW

Download observational or reanalysis data

Download or conduct climate model simulation

Advanced analysis of data using Python, NCL, etc

The screenshot shows the Sphinx Python Documentation Generator website. At the top is the logo, which includes an Egyptian eye icon and the word "SPHINX" in large white letters, with "Python Documentation Generator" in smaller white text below it. Below the logo is a blue header bar with the word "Welcome" in white. The main content area has a white background with dark grey text. It starts with a brief introduction: "Sphinx is a tool that makes it easy to create intelligent and beautiful documentation, written by Georg Brandl and licensed under the BSD license." It then mentions its original creation for the Python documentation and its use for software projects in various languages. It highlights features like reStructuredText sources. At the bottom of this section is a horizontal navigation bar with links for "Documentation", "API", "Search", and "GitHub".

Assess code documentation



Create science figures

This screenshot shows a Zenodo project page. On the left, there's a sidebar with "Publication date: May 21, 2018", "DOI: 10.5281/zenodo.1250498", "Related identifiers: Supplement to https://github.com/zmlabe/ThicknessSensitivity/tree/v3.0.0", and "License (for files): Other (Open)". The main content area has a light blue header with "Versions". Below it is a table with three rows:

Version	Date
v3.0.0 10.5281/zenodo.1250498	May 21, 2018
v2.0.0 10.5281/zenodo.1210287	Mar 30, 2018

At the bottom right of the page, there's a note: "Cite all versions? You can cite all versions by using the DOI 10.5281/zenodo.1154985. This DOI represents all versions, and will always resolve to the latest one. [Read more](#).

Use Zenodo to create project DOI



## SOFTWARE METAPAPER

# eof: A Library for EOF Analysis of Meteorological, Oceanographic, and Climate Data

Andrew Dawson<sup>1</sup>

<sup>1</sup> Atmospheric, Oceanic & Planetary Physics, Department of Physics, University of Oxford, Oxford, UK  
[andrew.dawson@physics.ox.ac.uk](mailto:andrew.dawson@physics.ox.ac.uk)

# Behind each library

This article describes the `eof` Python library, which provides a high-level interface for performing empirical orthogonal function (EOFs) and related quantities, with a focus on correctness and ease of use. The library is implemented in a modular hierarchical fashion, allowing computations using plain arrays, or the inclusion of metadata. The software provides a convenient package for users wanting to perform EOF analysis in Python, and integrates with popular libraries from atmospheric and climate science. The software is available on Github.

**Keywords:** EOF analysis; Meteorology; Oceanography; Climate; Python

## (1) Overview

### Introduction

Data sets in meteorology, oceanography, and climate are typically very large, containing data covering large spatial areas, observed or modelled over long periods of time. Studying variability in these data sets can be chal-

a time-series of sea surface temperature on a latitude-longitude grid. First one must correctly weight the input data to account for spatial variability in the size of grid cells due to convergence of the meridians. The input data must then be reconfigured into a 2-dimensional form, and care taken to remove any missing values (e.g., values

## SOFTWARE METAPAPER

# **eof**s: A Library for EOF Analysis of Meteorological, Oceanographic, and Climate Data

Andrew Dawson<sup>1</sup>

<sup>1</sup> Atmospheric, Oceanic & Planetary Physics, Department of Physics, University of Oxford, Oxford, UK  
andrew.dawson@physics.ox.ac.uk

---

The *eof*s library provides a high-level Python interface for computing empirical orthogonal functions (EOFs) and related quantities, with a focus on correctness and ease of use. The library is implemented in a modular hierarchical fashion, allowing computations using plain arrays, or the inclusion of metadata. The software provides a convenient package for users wanting to perform EOF analysis in Python, and integrates with popular libraries from atmospheric and climate science. The software is available on Github.

---

**Keywords:** EOF analysis; Meteorology; Oceanography; Climate; Python

---

## (1) Overview

### Introduction

Data sets in meteorology, oceanography, and climate are typically very large, containing data covering large spatial areas, observed or modelled over long periods of time. Studying variability in these data sets can be chal-

a time-series of sea surface temperature on a latitude-longitude grid. First one must correctly weight the input data to account for spatial variability in the size of grid cells due to convergence of the meridians. The input data must then be reconfigured into a 2-dimensional form, and care taken to remove any missing values (e.g., values

# WHY

Is there a crisis in science?

Transparency

Climate science needs this!

Reproducibility

Big data is the future

Code-sharing

Python is a growing community

Innovation

Open-science

Science deserves to be shared