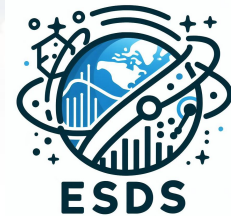


CESM Unified Postprocessing and Diagnostics (CUPiD)

Mike Levy and Teagan King
June 11th, 2024 - SEWG Meeting



CUPiD Collaborators

- **AMP:** *Dani Coleman, Cecile Hannay, Brian Medeiros, Christina McCluskey, Jesse Nusbaumer, Justin Richling*
- **CAS:** John Fasullo, Adam Phillips, Isla Simpson
- **CCR:** Gary Strand
- **CSEG:** Brian Dobbins
- **CESM:** Dave Lawrence
- **ESDS:** Katie Dagon, Teagan King, Mike Levy
- **ESMF:** Bill Sacks
- **GeoCAT (CISL):** Orhan Eroglu, Katelyn FitzGerald, Anissa Zacharias
- **OS:** Anna Deppenmeier, Gustavo Marques, *Lev Romashkov*
- **PPC:** Dave Bailey, Kate Thayer-Calder, Feng Zhu
- **TSS:** Sam Levis, Will Wieder, Naoki Mizukami
- **Students & Interns:** Shivani Kumar, Hilary Lam, Ingrid Carlson

CESM Diagnostics and ESDS



- CESM users were hitting the limits of NCL-based analysis
 - NCL is no longer being developed
 - Advantage to everyone replacing NCL with the same product
 - ESDS helped community find a solution
 - Pangeo stack: numpy, xarray, matplotlib, dask, etc

ADF



- Still had several independent efforts for analyzing CESM output
 - ADF, NBscuid, individuals writing one-off notebooks



- Next step: build a common framework
 - **CESM Unified Postprocessing and Diagnostics (CUPiD)**



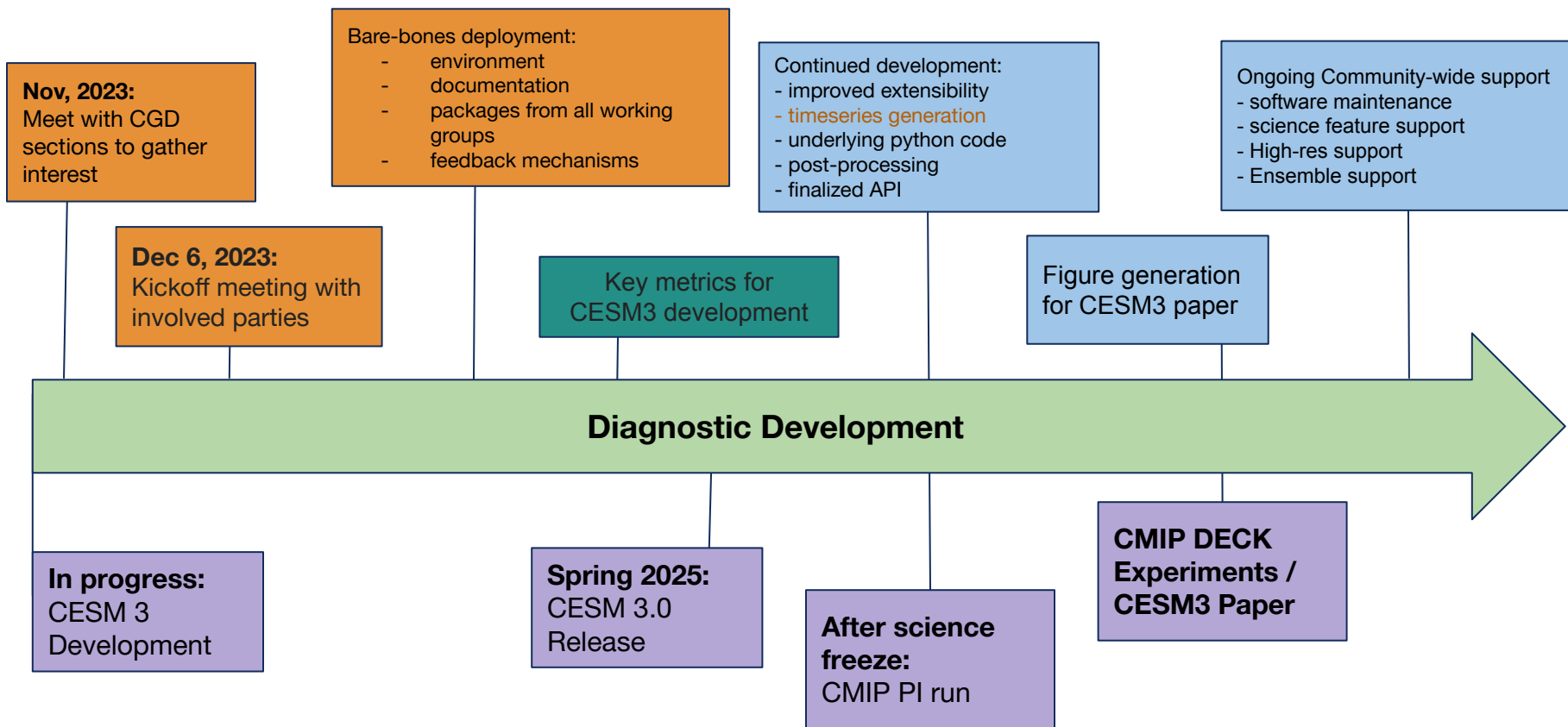
CUPiD Project Vision

CUPiD is a “one stop shop” that enables and integrates timeseries file generation, data standardization, diagnostics, and metrics from all CESM components.

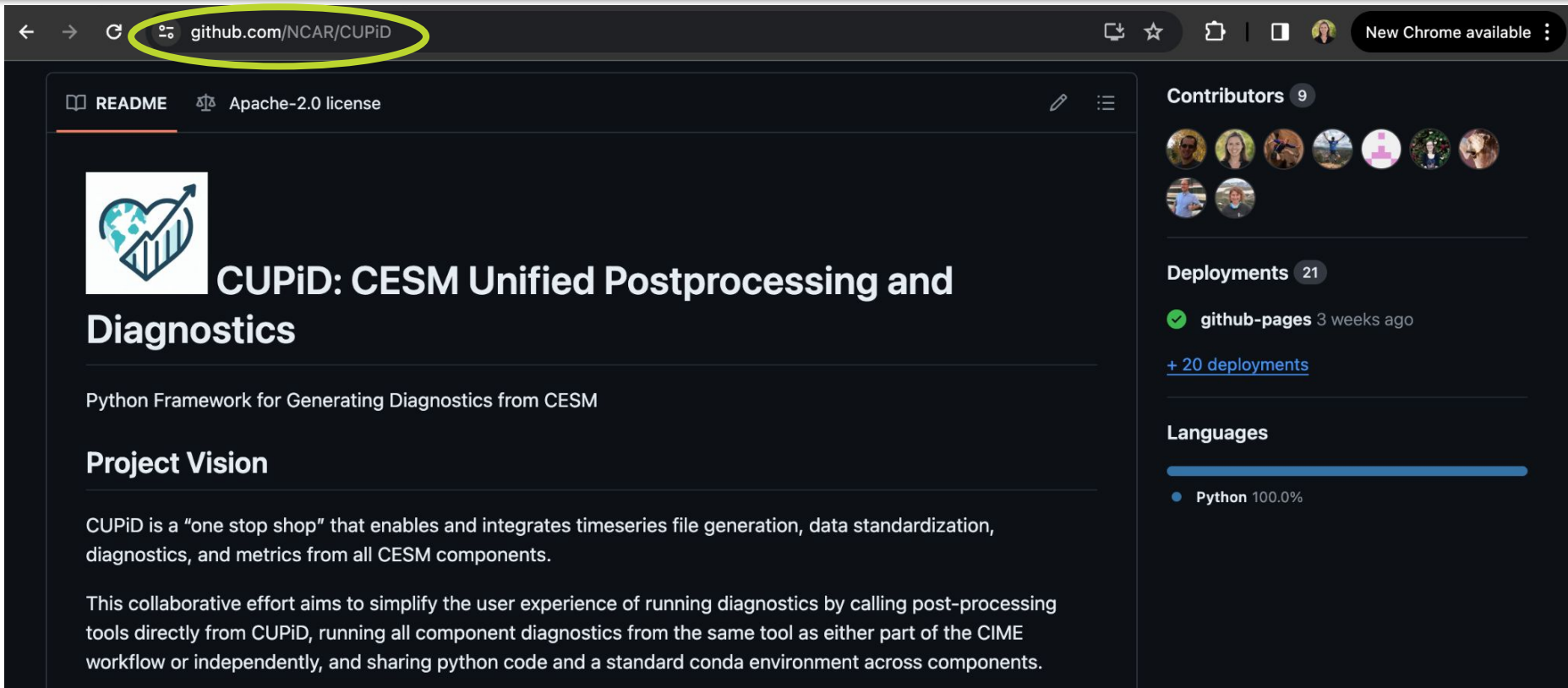
This collaborative effort aims to simplify the user experience of running diagnostics by calling post-processing tools directly from CUPiD, running all component diagnostics from the same tool as either part of the CIME workflow or independently, and sharing python code and a standard conda environment across components.



Anticipated Timeline




CUPiD Repository



The screenshot shows the GitHub repository page for CUPiD. The browser's address bar is highlighted in yellow, showing the URL `github.com/NCAR/CUPiD`. The repository page includes a README section with a logo of a heart containing a globe and an upward-pointing arrow. The repository is licensed under Apache-2.0. On the right side, there are sections for Contributors (9), Deployments (21), and Languages (Python 100.0%).

github.com/NCAR/CUPiD

README Apache-2.0 license



CUPiD: CESM Unified Postprocessing and Diagnostics

Python Framework for Generating Diagnostics from CESM

Project Vision

CUPiD is a "one stop shop" that enables and integrates timeseries file generation, data standardization, diagnostics, and metrics from all CESM components.

This collaborative effort aims to simplify the user experience of running diagnostics by calling post-processing tools directly from CUPiD, running all component diagnostics from the same tool as either part of the CIME workflow or independently, and sharing python code and a standard conda environment across components.

Contributors 9

Deployments 21

github-pages 3 weeks ago

[+ 20 deployments](#)

Languages

- Python 100.0%

How Do I Set Up CUPiD?

Install package:

```
$ git clone --recurse-submodules https://github.com/NCAR/CUPiD.git  
$ cd CUPiD  
$ ./manage_externals/checkout_externals
```

Build the CUPiD environments:

```
$ mamba env create -f environments/dev-environment.yml  
$ mamba env create -f environments/cupid-analysis.yml
```

How Can I Use CUPiD?

Provided example:

```
$ cd examples/coupled_model
$ cupid-run      # runs notebooks (from cupid-dev environment)
$ cupid-build   # builds website
```

Run on new cases:

modify [config.yml](#) file

```
$ cupid-run # from directory containing config.yml
$ cupid-build
```


Current Status Overview

- ✓ Mechanism for running notebooks in parallel
- ✓ Timeseries file generation
- ✓ Examples for most components
- ✓ Command line arguments
- ✓ Common environment
- ✓ [Documentation](#)



Coming Soon

- Provide quick metrics for CESM3 development runs
- Run python scripts in addition to notebooks
- Run on machines other than Derecho / Casper
- Run as part of CESM Workflow
- Run notebooks that import other diagnostic packages (ILAMB, etc)



Technical Details: Current & Proposed implementation

- Overview of configuration (YAML) file
- Mechanism for running notebooks in parallel
- Timeseries file generation



Details for config.yml

```
##### SETUP #####

#####
# Data Sources #
#####

data_sources:
  # sname is any string used as a nickname for this configuration. It will be
  ### used as the name of the folder your computed notebooks are put in
  sname: quick-run

  # run_dir is the path to the folder you want
  ### all the files associated with this configuration
  ### to be created in
  run_dir: .

  # nb_path_root is the path to the folder that cupid will
  ### look for your template notebooks in. It doesn't have to
  ### be inside run_dir, or be specific to this project, as
  ### long as the notebooks are there
  nb_path_root: ../nblibrary
```



Details for config.yml

```
#####  
# Computation Config #  
#####
```

```
computation_config:
```

```
# default_kernel_name is the name of the environment that  
### the notebooks in this configuration will be run in by default.  
### It must already be installed on your machine. You can also  
### specify a different environment than the default for any  
### notebook in NOTEBOOK CONFIG
```

```
default_kernel_name: cupid-analysis
```



Details for config.yml

```
##### NOTEBOOK CONFIG #####  
  
#####  
# Notebooks and Parameters #  
#####  
  
# All parameters under global_params get passed to all the notebooks  
  
global_params:  
  CESM_output_dir: /glade/campaign/cesm/development/cross-wg/diagnostic_framework/CESM_output_for_testing  
  lc_kwargs:  
    threads_per_worker: 1  
  
timeseries:  
  num_procs: 8  
  ts_done: [False]  
  overwrite_ts: [False]  
  case_name: 'b.e23_alpha16b.BLT1850.ne30_t232.054'  
  
atm:  
  vars: ['ACTNI', 'ACTNL', 'ACTREI', 'ACTREL', 'AODDUST']  
  derive_vars: [] # {'PRECT':['PRECL', 'PRECC'], 'RESTOM':['FLNT', 'FSNT']}  
  hist_str: 'h0'  
  start_years: [2]  
  end_years: [102]  
  level: 'lev'
```



Details for config.yml

```
compute_notebooks:  
  
# This is where all the notebooks you want run and their  
# parameters are specified. Several examples of different  
# types of notebooks are provided.  
  
# The first key (here simple_no_params_nb) is the name of the  
# notebook from nb_path_root, minus the .ipynb  
  
  infrastructure:  
    index:  
      parameter_groups:  
        none: {}  
  
  atm:  
    adf_quick_run:  
      parameter_groups:  
        none:  
          adf_path: ../../../../externals/ADF  
          config_path: .  
          config_fil_str: "config_f.cam6_3_119.FLTHIST_ne30.r328_gamma0.33_soa.001.yaml"
```



Details for config.yml

```
#####  
# Jupyter Book Table of Contents #  
#####  
book_toc:  
  
# See https://jupyterbook.org/en/stable/structure/configure.html for  
# complete documentation of Jupyter book construction options  
  
format: jb-book  
  
# All filenames are notebook filename without the .ipynb, similar to above  
  
root: infrastructure/index # root is the notebook that will be the homepage for the book  
parts:  
  
# Parts group notebooks into different sections in the Jupyter book  
# table of contents, so you can organize different parts of your project.  
  
- caption: Atmosphere  
  
# Each chapter is the name of one of the notebooks that you executed  
# in compute_notebooks above, also without .ipynb  
chapters:  
  - file: atm/adf_quick_run
```



Requirement: Dask-based parallelization (Pangeo stack)

1. Current Implementation

- a. User requests resources upfront
- b. Parallelize with dask's LocalCluster

PRO:

- Portable across computers regardless of queue manager

CON:

- Requires running notebooks sequentially, and not all notebooks will use all resources

2. Desired Implementation

- Use CESM machine info to choose Cluster
 - May change CUPiD externals / interaction with CESM



Current Timeseries File Generation

[ADF / lib / adf_diag.py](#)

Code Blame 1160 lines (975 loc) · 44.2 KB

```
339
340     #####
341
342     def create_time_series(self, base
343         """
344         Generate time series version:
345         """
346
```

[CUPiD / cupid / timeseries.py](#) 

 **TeaganKing** Include GitHub Actions (#98)  

Code Blame 423 lines (372 loc) · 16.8 KB

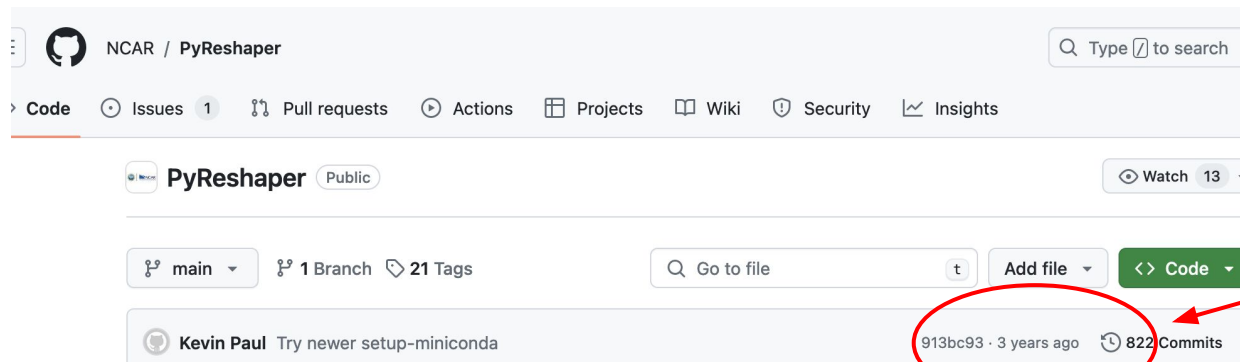
```
1     """
2     Timeseries generation tool adapted from ADF for general CUPiD use.
3     """
```

Cons: lightweight tool that doesn't handle all edge cases



Desired Timeseries File Generation

Call an external tool if time series generation is desired



NCAR / PyReshaper

Code Issues 1 Pull requests Actions Projects Wiki Security Insights

PyReshaper Public Watch 13

main 1 Branch 21 Tags Go to file Add file Code

Kevin Paul Try newer setup-miniconda 913bc93 · 3 years ago 822 Commits

Something like
PyReshaper, but still
maintained

Summary

- Work is on-going towards a portable and extensible CESM postprocessing / diagnostics package
- We welcome feedback, suggestions, testers & contributors
- Current development version is available!
- Bonus slides show screenshots of generated web page

Thank You!



Example project

Q Search

⌘ + K

Atmosphere

ADF Diagnostics In Jupyter

Ocean

Analysis of Surface Fields

Land

Simple example comparing land variables from two simulations

Sea Ice

Sea Ice Diagnostics for two CESM3 runs



Index homepage!

```
# Parameters
CESM_output_dir = "/glade/campaign/cesm/development/cross-wg/diagnostic_framework/CESM_outp
lc_kwargs = {"threads_per_worker": 1}
serial = False
subset_kwargs = {}
product = "/glade/work/mlevy/codes/CUPiD/examples/coupled_model/computed_notebooks/quick-run
```

[ADF Diagnostics In Jupyter](#) Next >

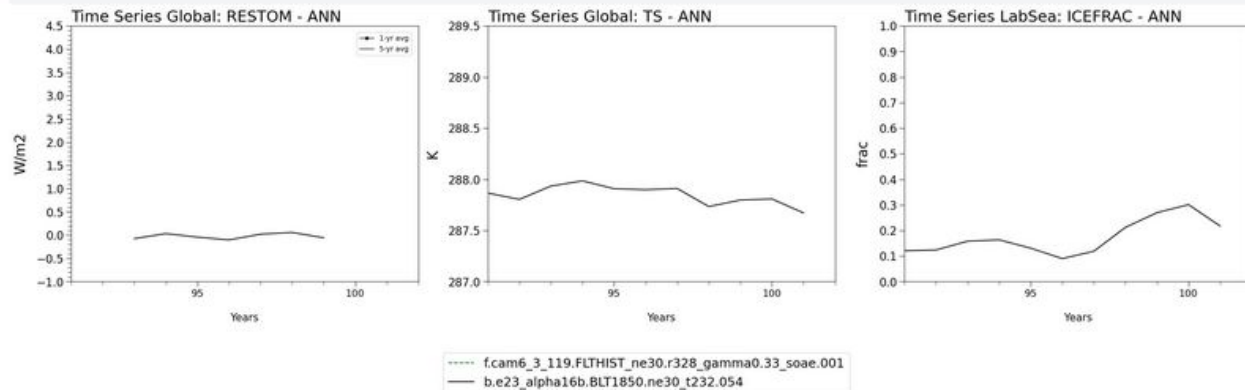
“Live” Demo Screenshots

Plotting variable: RESTOM

Plotting variable: TS

Plotting variable: ICEFRAC

FigureCanvasAgg is non-interactive, and thus cannot be shown



Exploration of the Output Data

Let's grab the case names, time series locations, variable defaults dictionary and climo years

Time Series Plotting Functions

Plot the time series!

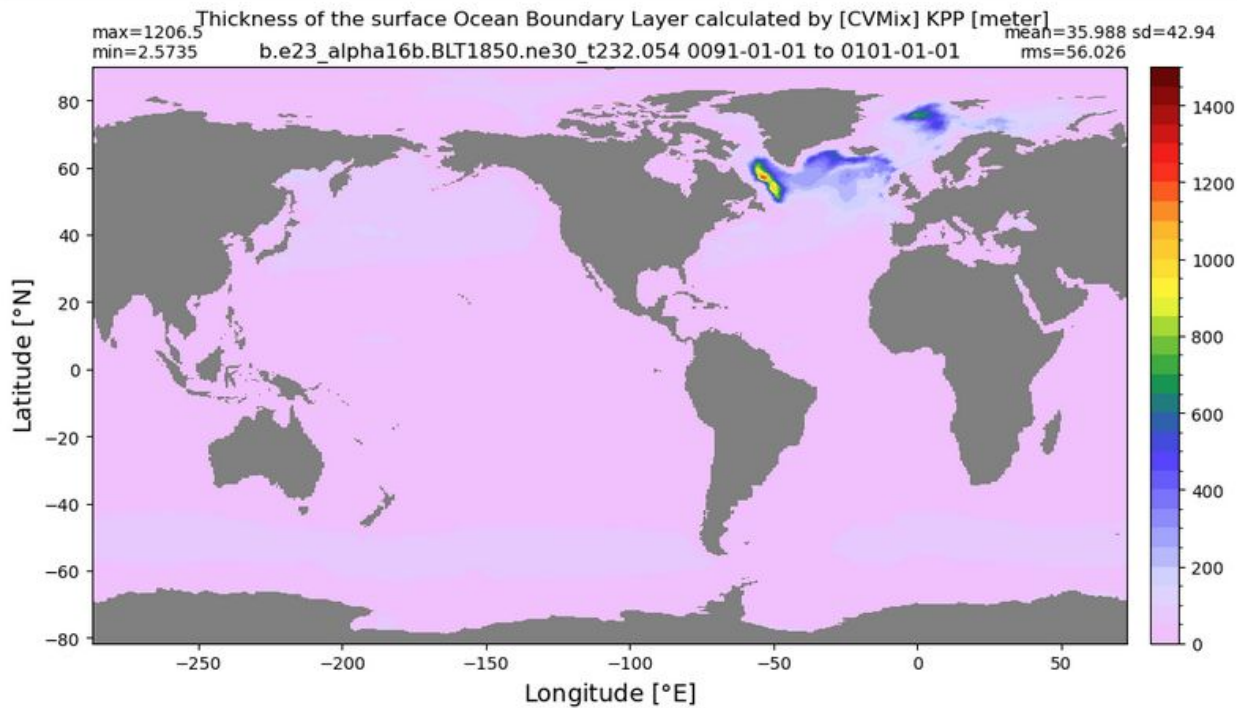
“Live” Demo Screenshots

Time elapsed: 0:00:13.230260

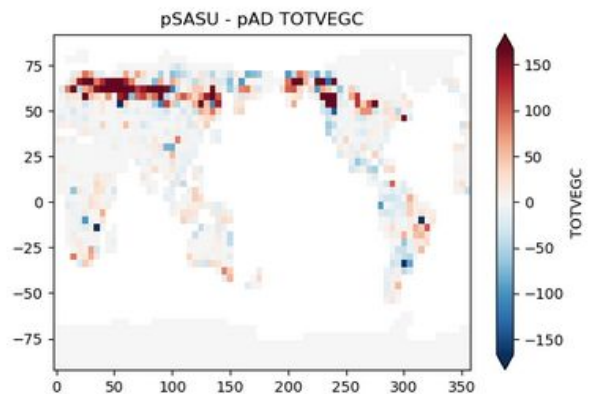
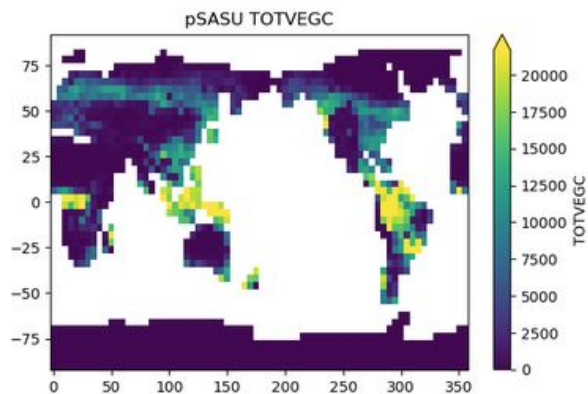
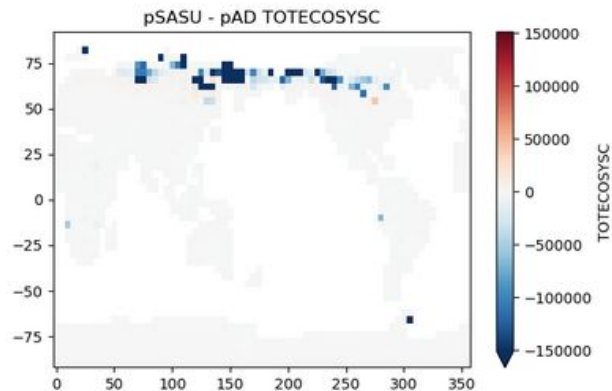
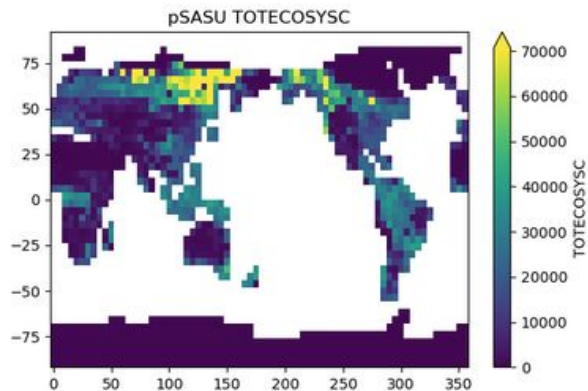
Plotting...

Mixed layer depth

Boundary layer depth



“Live” Demo Screenshots



Quickplots of pSASU results & difference from pAD

lower soil C stocks at high latitudes

“Live” Demo Screenshots

Example project

Q Search

⌘ + K

Atmosphere

ADF Diagnostics In Jupyter

Ocean

Analysis of Surface Fields

Land

Simple example comparing land variables from two simulations

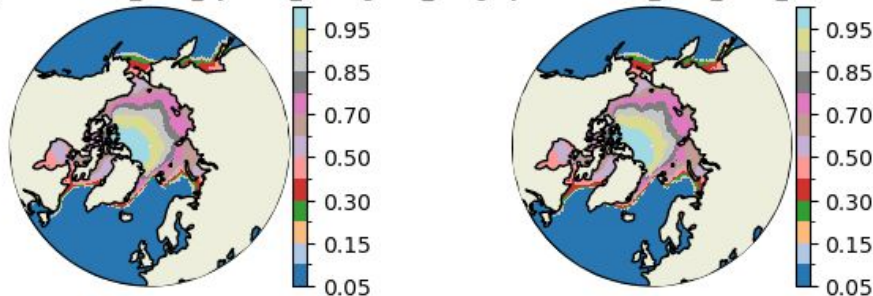
Sea Ice

[Sea Ice Diagnostics for two CESM3 runs](#)

```
field2 = us2_dmi[vars].isel(time=slice(-nyears,None)).mean('time').squeeze()  
plot_diff(field1, field2, levels, case1, case2, title, "N", TLAT, TLON)
```

Sea Ice Concentration

g.e23_a16g.GJRAv4.TL319_t232_hycom1_N75.2024.001-g.e23_a16g.GJRAv4.TL319_t232_zstar_N65.2024.004



g.e23_a16g.GJRAv4.TL319_t232_zstar_N65.2024.004-g.e23_a16g.GJRAv4.TL319_t232_hycom1_N75.2024.005

