

ECCO Python Tutorials in the Cloud

MORE ADVANCED CALCULATIONS

Compute meridional heat transport

Compute MOC along the approximate OSNAP array from ECCO

ECCOv4 Global Volume Budget Closure

Global Heat Budget Closure

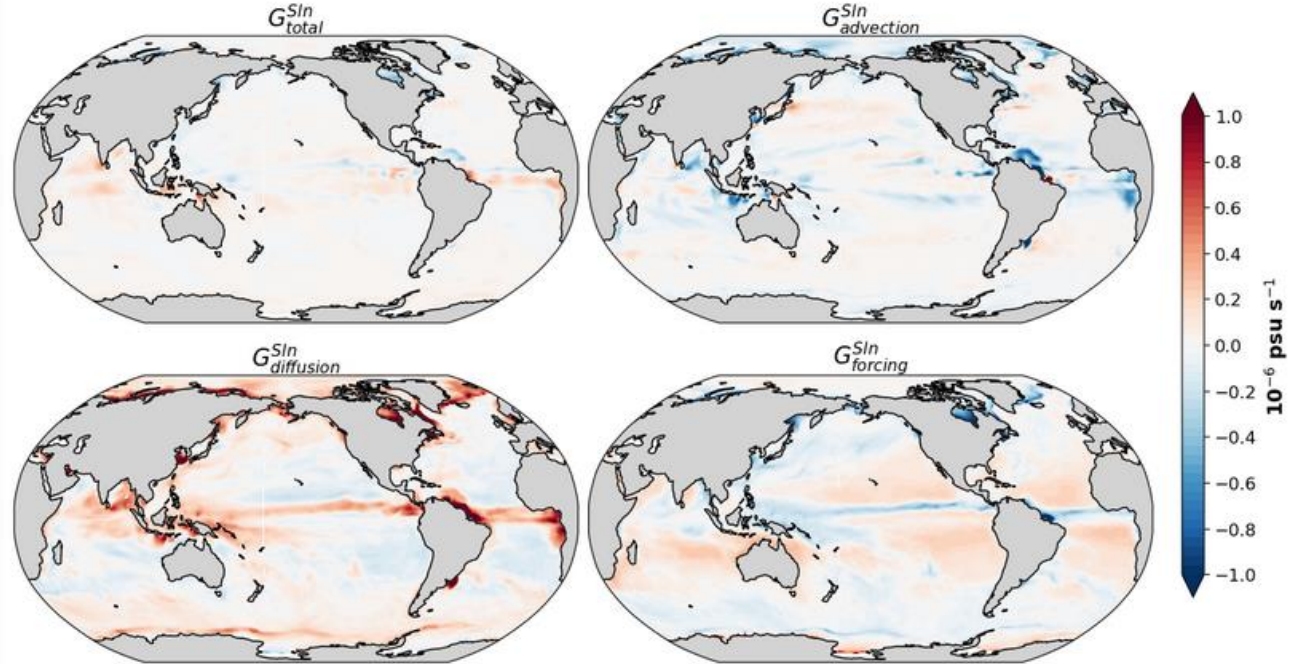
Salt, Salinity and Freshwater Budgets

Calculate ocean thermal forcing from ECCOv4r4 data, direct from PO.DAAC S3 storage

SUPPORT

Getting Help

Spatial distribution at $z = 5$ m of salinity budget components in May 2001



Andrew Delman¹, Ian Fenty²

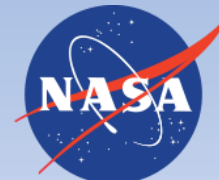
¹Joint Institute For Regional Earth System Science & Engineering (JIFRESSE), University of California Los Angeles, Los Angeles, CA, USA

²Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA, USA

Primary contact: andrewdelman@g.ucla.edu



ECCO Meeting
March 2024



Jet Propulsion Laboratory
California Institute of Technology

ECCO Python Tutorials

- Tutorials website to provide oceanographers with demos and code for ECCO analysis
 - Started by Ian Fenty a few years ago, with contributions from many others
- Comprised mostly of Jupyter notebooks that can be downloaded or git cloned to user's machine
- <https://ecco-v4-python-tutorial.readthedocs.io>
- The tutorials are also linked from the <https://www.ecco-group.org> website under Products → Analysis Tools

The screenshot shows the home page of the ECCO Version 4 Python Tutorial website. The header includes the site name and a search bar. The main content area features a 'Welcome to the ECCO Version 4 Tutorial' message, a brief description of the site's purpose, and a section for 'Additional Resources' with links to related topics. A sidebar on the left contains navigation links for 'GETTING STARTED', 'ECCO DATA STRUCTURES', and 'INPUT/OUTPUT, DATA STRUCTURE MANIPULATION'.

The screenshot displays a tutorial page titled 'Compute meridional heat transport'. It includes a list of model variables to load, instructions for selecting a latitude band, and a description of the meridional heat transport (MHT) as a function of depth. Two heat transport plots are shown: 'Global time mean meridional heat transport' and 'Atlantic time mean meridional heat transport'. Both plots show depth (0 to 4000 m) on the y-axis and latitude (-60 to 40) on the x-axis. The global plot has a color scale from -0.3 to 0.3 PW, while the Atlantic plot has a scale from -0.08 to 0.08 PW. A sidebar on the left contains a table of contents for the tutorial.

MHT tutorial credit: Timothy Smith

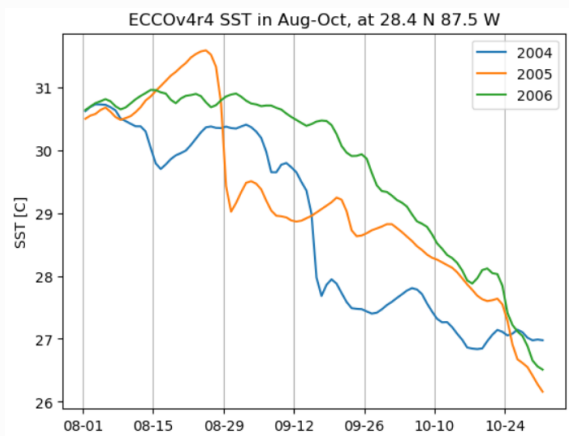
ECCO Python Tutorials: updates as of March 2024

- A release last year (May 2023) updated all of the tutorials for ECCOv4 release 4, accessed through PO.DAAC
 - All tutorials have the NASA Earthdata ShortNames of the datasets needed to run them listed at the beginning, with *ecco_download.py* module assisting with downloads to user's local machine
- **Intro to PO Tutorials** illustrate fundamental PO concepts with ECCO; 3 on the website with more planned
 - Have you done something with ECCO that is fun, interesting, and/or useful to students/educators? We'll help you adapt it into a tutorial!
- Recently added Python tutorials on:
 - Downloading spatial/temporal/variable subsets of ECCO datasets using OPeNDAP
 - Computing gradients (e.g., relative vorticity, wind stress curl) on the ECCO native grid

```
[13]: ecco_podaac_download_subset(ShortName='ECCO_L4_TEMP_SALINITY_LLC0090GRID_DAILY_V4R4',\
vars_to_include=['THETA'],\
times_to_include=['2004-08','2004-09','2004-10',\
'2005-08','2005-09','2005-10',\
'2006-08','2006-09','2006-10'],\
k_isel=[0,1,1],\
tile_isel=[10,11,1],\
j_isel=[28,48,1],\
i_isel=[66,82,1],\
subset_file_id='SST_GoM')
```

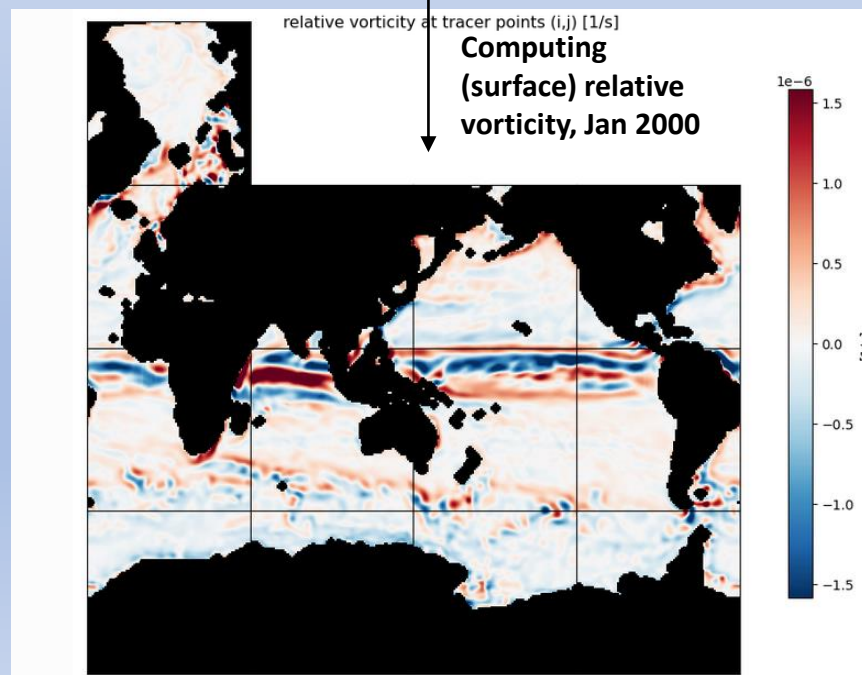
Download to directory C:\Users\adelman\Downloads\ECCO_V4r4_PO.DAAC\ECCO_L4_TEMP_SALINITY_LLC0090GRID_DA
Please wait while program searches for the granules ...

Download SST (top layer T) in Gulf of Mexico, Aug-Oct, 2004-2006



```
[57]: omega = dv_phi_dlambda - du_lambda_dphi

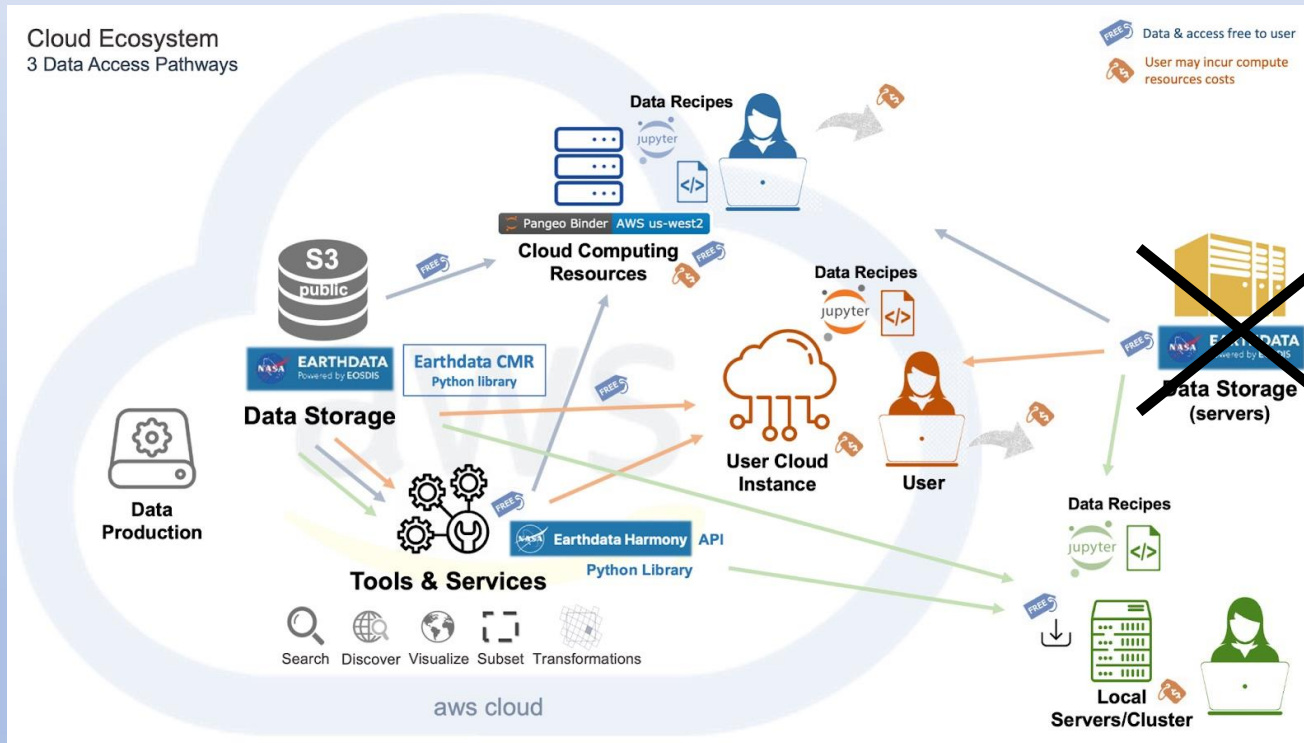
[58]: m=-5.8;
title='relative vorticity at tracer points (i,j) [1/s]'
ecco.plot_tiles(omega, cmin=-10*m, cmax=10*m,fig_size=10,
show_tile_labels=False,
rotate_to_latlon=True, layout='latlon', cmap=colMap,
show_colorbar=True, cbar_label='[1/s]', show_cbar_label=True);
plt.suptitle(title)
plt.show()
```



Tutorials on the Cloud

- You may know that PO.DAAC datasets are now archived on the NASA Earthdata Cloud, hosted by Amazon Web Services (AWS)
- But perhaps the idea of working in the Cloud seems as nebulous as...a cloud.

From PO.DAAC website

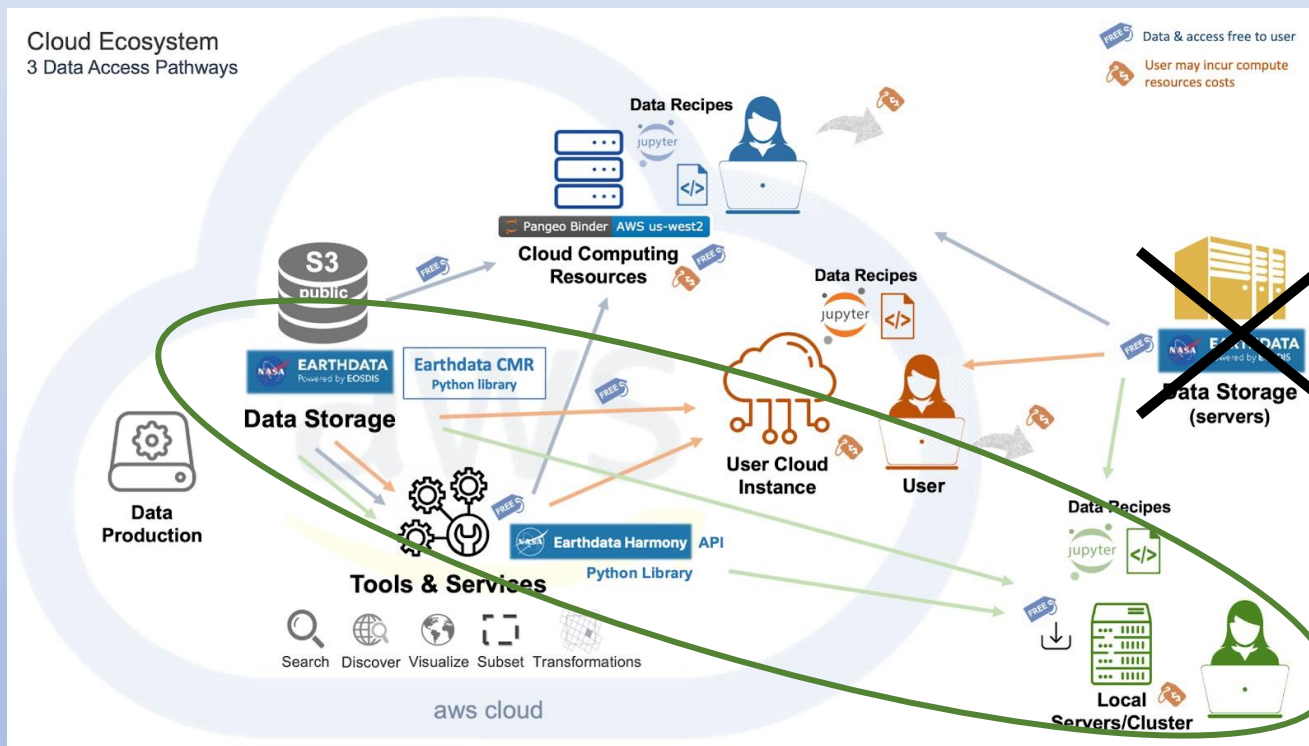


No more PO.DAAC data storage at JPL, only on AWS Cloud

Tutorials on the Cloud

- You may know that PO.DAAC datasets are now archived on the NASA Earthdata Cloud, hosted by Amazon Web Services (AWS)
- But perhaps the idea of working in the Cloud seems as nebulous as...a cloud.

From PO.DAAC website



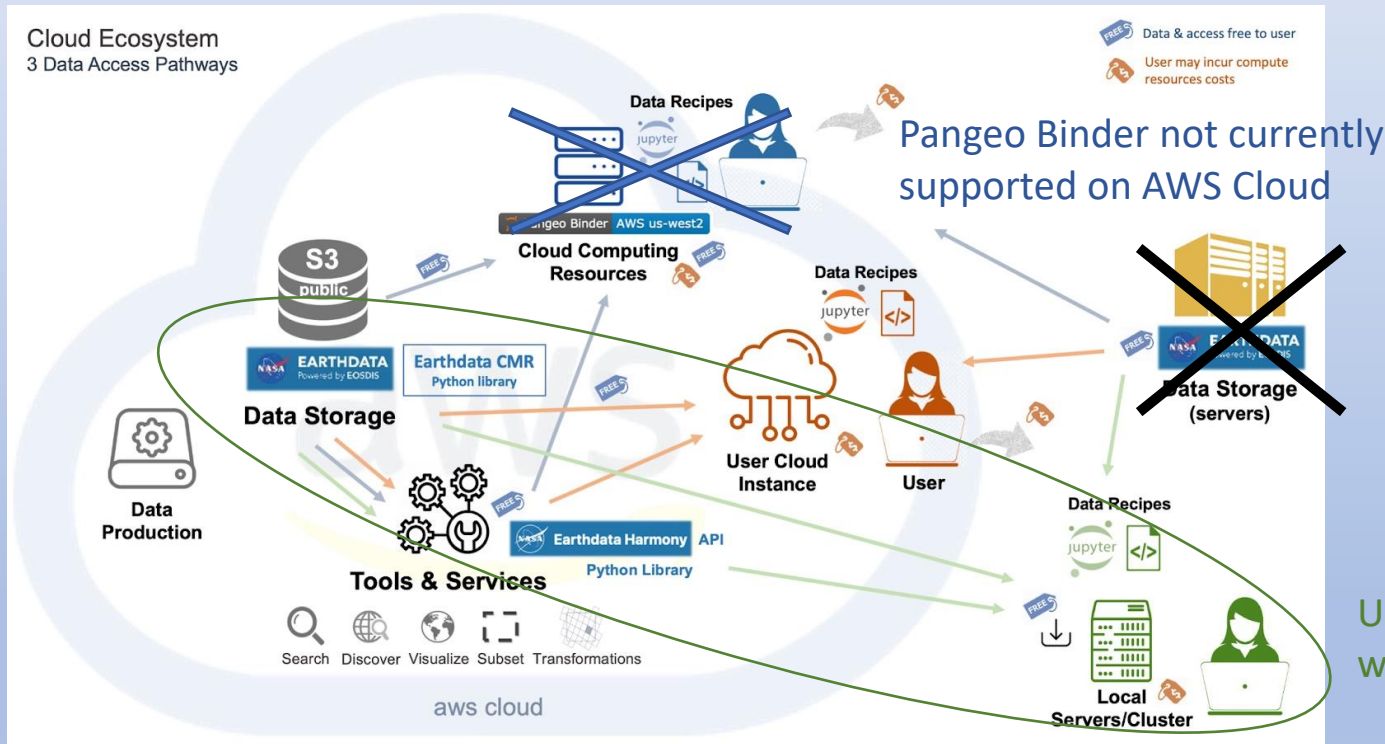
No more PO.DAAC data storage at JPL, only on AWS Cloud

User downloads data and works on local machine

Tutorials on the Cloud

- You may know that PO.DAAC datasets are now archived on the NASA Earthdata Cloud, hosted by Amazon Web Services (AWS)
- But perhaps the idea of working in the Cloud seems as nebulous as...a cloud.

From PO.DAAC website

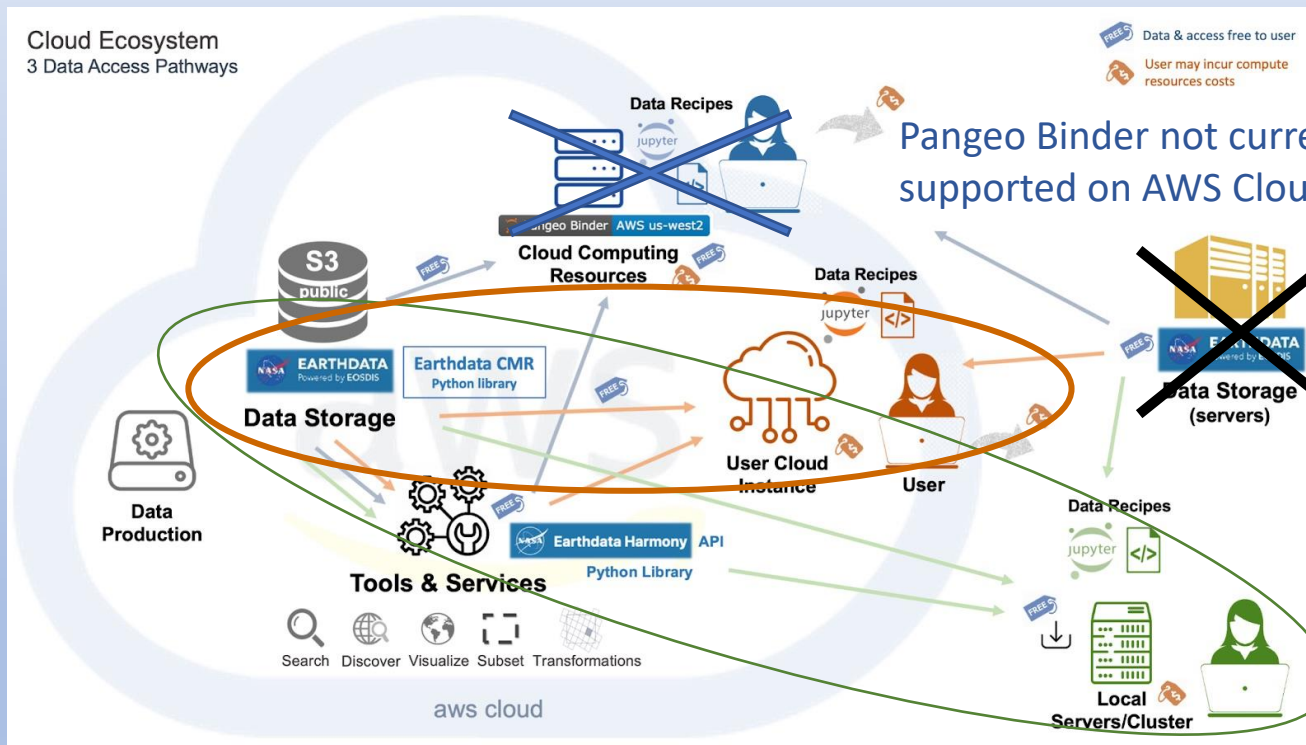


User downloads data and works on local machine

Tutorials on the Cloud

- You may know that PO.DAAC datasets are now archived on the NASA Earthdata Cloud, hosted by Amazon Web Services (AWS)
- But perhaps the idea of working in the Cloud seems as nebulous as...a cloud.

From PO.DAAC website



- Goal here is to “de-mist-ify” the cloud a bit, and to show how we are setting up the ECCO Python tutorials to run in the cloud (these can also be used to inform your own codes/workflows)

Setting up an “instance” in the AWS Cloud

- An instance is a self-contained computing environment with its own OS, memory, and storage
- A “free-tier” instance is available, though if possible it is recommended you seek project or institutional support

The image shows a dual-screen setup. The left screen displays a web browser with the URL `https://ecco-v4-python-tutorial.readthedocs.io`. The page title is "ECCO Version 4 Python Tutorial" and the article is "AWS Cloud: getting started and retrieving ECCO datasets". The article includes an introduction and a section titled "Set up an AWS cloud instance". The right screen shows the AWS Management Console Home page for the "us-west-2" region. The "Recently visited" section lists EC2, IAM, Billing and Cost Management, and EC2 Image Builder. The "Applications (0)" section shows the current region as "us-west-2 (Current Region)".

Setting up an “instance” in the AWS Cloud

- An instance is a self-contained computing environment with its own OS, memory, and storage
- A “free-tier” instance is available, though if possible it is recommended you seek project or institutional support

3 minutes later...

Setting up an “instance” in the AWS Cloud

- An instance is a self-contained computing environment with its own OS, memory, and storage
- A “free-tier” instance is available, though if possible it is recommended you seek project or institutional support

The image shows a browser window on the left and a terminal window on the right. The browser window displays a tutorial page for setting up a Jupyter environment on an AWS instance. The terminal window shows the execution of a shell script that clones a GitHub repository and sets up the environment.

Browser Window:

- URL: `https://ecco-v4-python-tutorial.readthedocs`
- Page Title: `AWSCloud: getting started and`
- Page Content: A tutorial page for "ECCO-v4-Python-Tutorial.git". The page includes sections for "PLOTTING & INTERPOLATION", "SCALAR AND VECTOR CALCULATIONS", "INTRO TO PO TUTORIALS", "MORE ADVANCED CALCULATIONS", and "SUPPORT".

Terminal Window:

```
cygwin/Xc0.0
ec2-user@ip-172-31-26-168:~
Main Options VT Options VT Fonts
perl-Pod-Escapes-1:1.07-460.el9.noarch
perl-Pod-Perldoc-3.28.01-461.el9.noarch
perl-Pod-Simple-1:3.42-4.el9.noarch
perl-Pod-Usage-4:2.01-4.el9.noarch
perl-Scalar-List-Utils-4:1.56-461.el9.x86_64
perl-SelectSaver-1.02-480.el9.noarch
perl-Socket-4:2.031-4.el9.x86_64
perl-Storable-1:3.21-460.el9.x86_64
perl-Symbol-1.08-480.el9.noarch
perl-Term-ANSIColor-5.01-461.el9.noarch
perl-Term-Cap-1.17-460.el9.noarch
perl-TermReadKey-2.38-11.el9.x86_64
perl-Text-ParseWords-3.30-460.el9.noarch
perl-Text-Tabs+Wrap-2013.0523-460.el9.noarch
perl-Time-Local-2:1.300-7.el9.noarch
perl-URI-5.09-3.el9.noarch
perl-base-2.27-480.el9.noarch
perl-constant-1.33-461.el9.noarch
perl-if-0.60.800-480.el9.noarch
perl-integerpreter-4:5.32.1-480.el9.x86_64
perl-lib-0.65-480.el9.x86_64
perl-libnet-3.13-4.el9.noarch
perl-libs-4:5.32.1-480.el9.x86_64
perl-mro-1.23-480.el9.x86_64
perl-overload-1.31-480.el9.noarch
perl-overloading-0.02-480.el9.noarch
perl-parent-1:0.238-460.el9.noarch
perl-podlators-1:4.14-460.el9.noarch
perl-subs-1.03-480.el9.noarch
perl-vars-1.05-480.el9.noarch

Complete!
Cloning into 'ECCO-v4-Python-Tutorial'...
remote: Enumerating objects: 1679, done.
remote: Counting objects: 100% (609/609), done.
remote: Compressing objects: 100% (247/247), done.
remote: Total 1679 (delta 378), reused 485 (delta 355), pack-reused 1870
Receiving objects: 100% (1679/1679), 93.70 MiB | 23.84 MiB/s, done.
Resolving deltas: 97% (947/976)
```

Setting up an “instance” in the AWS Cloud

- An instance is a self-contained computing environment with its own OS, memory, and storage
- A “free-tier” instance is available, though if possible it is recommended you seek project or institutional support

6 minutes later...

Setting up an “instance” in the AWS Cloud

- An instance is a self-contained computing environment with its own OS, memory, and storage
- A “free-tier” instance is available, though if possible it is recommended you seek project or institutional support

The image shows a web browser window on the left and a terminal window on the right. The browser window displays a tutorial page for setting up a Jupyter lab session on an AWS instance. The terminal window shows the execution of a shell script that sets up the environment and starts a Jupyter lab session.

Browser Window:

- URL: <https://ecco-v4-python-tutorial.readthedocs.io>
- Page Title: **Setting up a Jupyter lab session**
- Text: "To run `jupyter_env_setup.sh`, copy, paste, and execute the following two commands on the instance:"
- Code Block:

```
sh && ~/ECCO-v4-Python-Tutorial/C/Cloud_Setup/jupyter_env_setup.sh
```
- Text: "The script takes a few minutes to run, but it should set up our environment with the packages we need, even within the memory constraints of a free-tier t2.micro instance."
- Section: **Step 4: Start a Jupyter lab session**
- Text: "With the `jupyter` environment set up and credentials archived, we can run one more shell script to start a Jupyter lab session on the instance. The session will be opened using a `tmux` window; `tmux` is a program (similar to `screen`) that allows a session to be detached from the user's window to run in the background, and persist on a remote machine even after the user disconnects. The previous script created a symbolic link in the home directory that allows us to launch a Jupyterlab session easily after log in. Execute the following in the instance:"
- Code Block:

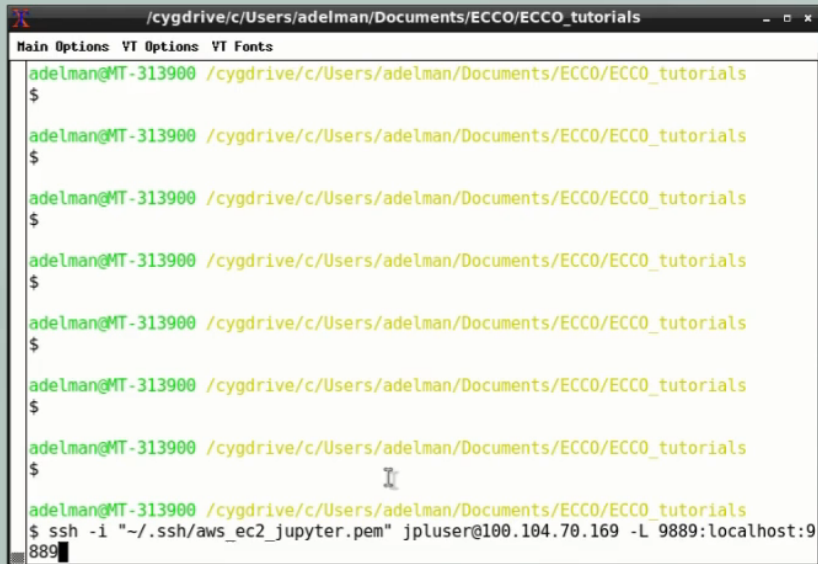
```
~/jupyter_lab_start.sh
```
- Text: "You will get a prompt for a password (optional), or you can leave it blank and press enter. After this is done (and while still connected to your instance through port 9889), open up a window in your local machine's web browser and put `http://127.0.0.1:9889/` or `http://localhost:9889/` in the URL field. If you set a password for your session, enter it when prompted. A Jupyter lab should open up in the ECCOv4 tutorial Github repository on your instance. Go to the Tutorials as Jupyter Notebooks directory, and you will see a

Terminal Window:

```
ec2-user@ip-172-31-26-168:~$ sh && ~/ECCO-v4-Python-Tutorial/C/Cloud_Setup/jupyter_env_setup.sh
Requirement already satisfied: tblib==1.6.0 in /tmp/conda/envs/jupyter/lib/python3.11/site-packages (from distributed==2024.3.1->dask[complete]->ecco_v4_py) (3.0.0)
Requirement already satisfied: tornado==6.0.4 in /tmp/conda/envs/jupyter/lib/python3.11/site-packages (from distributed==2024.3.1->dask[complete]->ecco_v4_py) (6.4)
Requirement already satisfied: urllib3==1.24.3 in /tmp/conda/envs/jupyter/lib/python3.11/site-packages (from distributed==2024.3.1->dask[complete]->ecco_v4_py) (2.0.7)
Requirement already satisfied: zict==3.0.0 in /tmp/conda/envs/jupyter/lib/python3.11/site-packages (from distributed==2024.3.1->dask[complete]->ecco_v4_py) (3.0.0)
Requirement already satisfied: xyzservices==2021.09.1 in /tmp/conda/envs/jupyter/lib/python3.11/site-packages (from bokeh==2.4.2->dask[complete]->ecco_v4_py) (2023.10.1)
Requirement already satisfied: MarkupSafe==2.0 in /tmp/conda/envs/jupyter/lib/python3.11/site-packages (from Jinja2==2.10.3->dask[complete]->ecco_v4_py) (2.1.5)
Downloading ecco_v4_py-1.5.5-py3-none-any.whl (90 kB)
 90.3/90.3 kB 4.9 MB/s eta 0:00:00
Downloading xgcm-0.8.1-py3-none-any.whl (95 kB)
 95.6/95.6 kB 9.3 MB/s eta 0:00:00
Downloading cmocean-3.1.3-py3-none-any.whl (222 kB)
 222.1/222.1 kB 20.1 MB/s eta 0:00:00
Downloading future-1.0.0-py3-none-any.whl (491 kB)
 491.3/491.3 kB 42.3 MB/s eta 0:00:00
Downloading pyresample-1.28.2-cp311-cp311-manylinux_2_17_x86_64_manylinux2014_x86_64.whl (4.5 MB)
 4.5/4.5 MB 18.5 MB/s eta 0:00:00
Downloading xmitgcm-0.5.2-py3-none-any.whl (100 kB)
 100.0/100.0 kB 15.3 MB/s eta 0:00:00
Downloading pykdtree-1.3.11-cp311-cp311-manylinux_2_17_x86_64_manylinux2014_x86_64.whl (370 kB)
 370.2/370.2 kB 37.2 MB/s eta 0:00:00
Downloading cachetools-5.3.3-py3-none-any.whl (9.3 kB)
Downloading configobj-5.0.8-py2.py3-none-any.whl (36 kB)
Downloading donfig-0.8.1.post0-py3-none-any.whl (24 kB)
Installing collected packages: pykdtree, future, donfig, configobj, cachetools, pyresample, cmocean, xmitgcm, xgcm, ecco_v4_py
```

Running a tutorial on an AWS Cloud instance

- Sample calculation: ECCOv4r4 global mean ocean temperature (full depth) for 12 months in 2010
- Downloading the files needed to my laptop for this calculation took 3 minutes (on hotel wi-fi), versus...



```

/cygdrive/c/Users/adelman/Documents/ECCO/ECCO_tutorials
Main Options VT Options VT Fonts
adelman@MT-313900 /cygdrive/c/Users/adelman/Documents/ECCO/ECCO_tutorials
$
adelman@MT-313900 /cygdrive/c/Users/adelman/Documents/ECCO/ECCO_tutorials
$
adelman@MT-313900 /cygdrive/c/Users/adelman/Documents/ECCO/ECCO_tutorials
$
adelman@MT-313900 /cygdrive/c/Users/adelman/Documents/ECCO/ECCO_tutorials
$
adelman@MT-313900 /cygdrive/c/Users/adelman/Documents/ECCO/ECCO_tutorials
$
adelman@MT-313900 /cygdrive/c/Users/adelman/Documents/ECCO/ECCO_tutorials
$
adelman@MT-313900 /cygdrive/c/Users/adelman/Documents/ECCO/ECCO_tutorials
$
adelman@MT-313900 /cygdrive/c/Users/adelman/Documents/ECCO/ECCO_tutorials
$ ssh -i ~/.ssh/aws_ec2_jupyter.pem" jpluser@100.104.70.169 -L 9889:localhost:9889

```

Final remarks/thoughts

- A short snippet of code is being added to each of the tutorials to enable users to access ECCO output and run tutorials on the AWS Cloud
 - Powered by the `ecco_s3_retrieve.py` module (analogous to `ecco_download.py`, but for in-cloud access)

```
ShortNames_list = ["ECCO_L4_TEMP_SALINITY_LLC0090GRID_MONTHLY_V4R4",\
                  "ECCO_L4_GEOMETRY_LLC0090GRID_V4R4"]

## download files to instance (only if download size is less than available storage threshold from max_avail_frac)
retrieved_files = ecco_podaac_s3_get_diskaware(ShortNames=ShortNames_list,\
                                              StartDate="2010-01", EndDate="2010-12",\
                                              max_avail_frac=0.5, n_workers=2)

## open files in workspace
ds = xr.open_mfdataset(retrieved_files[ShortNames_list[0]],\
                      data_vars='minimal', coords='minimal',\
                      compat='override',\
                      chunks={'time':1, 'k':10, 'tile':13, 'j':90, 'i':90})

ds_grid = xr.open_mfdataset(retrieved_files[ShortNames_list[1]], chunks={'k':10, 'tile':13, 'j':90, 'i':90})
```

- AWS Cloud instances can be run for free (for 12 months) but to run code successfully, it needs to be designed in a way to minimize memory footprint (e.g., using `dask` chunking, “lazy” computations)
- Here’s the URL for the tutorials again: <https://ecco-v4-python-tutorial.readthedocs.io>
 - Can also be accessed through the ECCO Group website under Products -> Analysis Tools
- Questions or ideas (including tutorial ideas)? Please contact me: andrewdelman@g.ucla.edu.