Connecting Users and Applications with PO.DAAC hosted GHRSST data

podaac
Physical Oceanography Distributed Active Archive Center

Edward M. Armstrong[1], Wen-Hao Li[1], Yibo Jiang[1], Chelle Gentemann[2]
[1]Jet Propulsion Laboratory, California Institute of Technology
[2] Farallon Institute

21st GHRSST Science Team Meeting
2020 June 1-4

Session 4: Services and Products

These activities were carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration. Dedicated funding for PO.DAAC activities is through a grant from NASA's ESDIS Project.

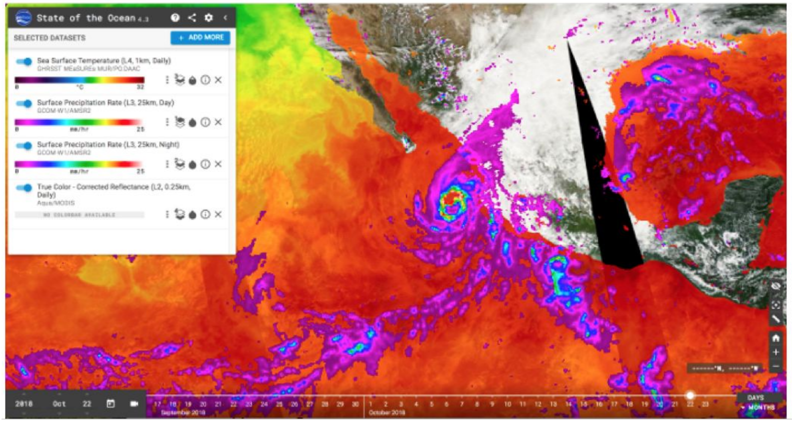
Outline

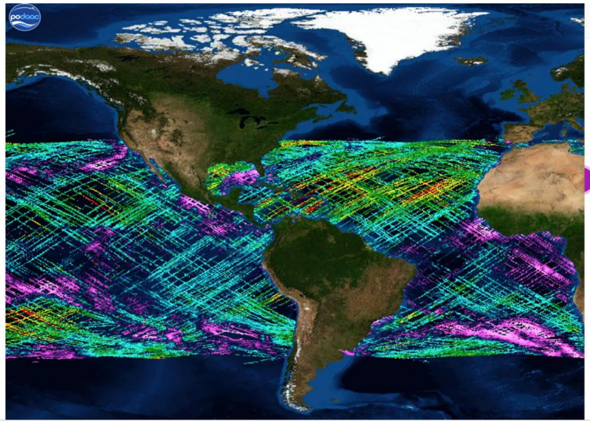
- Current PO.DAAC tools and services
- Data Recipes
  - Forum based python/NCO SST phenology and time series
- Jupyter recipes
  - podaacpy
- PO.DAAC GitHub
- Cloud based capabilities
- Future PO.DAAC capabilities

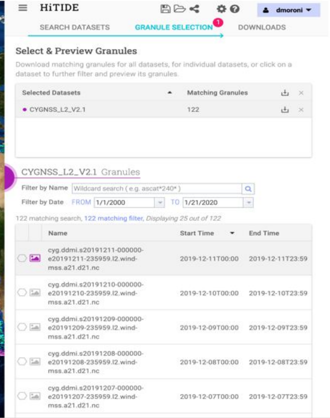
Existing Tools and Services

- Tools/Services (https://podaac.jpl.nasa.gov/dataaccess)

Service Name	Access	Visualization	Analysis	Subset	X-Format	FAIR
Drive	Direct, WebDav API	NO	NO	NO	NO	A
Web Services	Direct, API Query	YES	via API	YES	via API	FAIR
OPeNDAP	Direct, API Query	NO	via API	via API	YES	AIR
THREDDS	Direct, API Query	YES	via API	via API	via API	AIR
Live Access Server	Direct, GUI Query	YES + Animation	YES	YES	YES	AIR
SOTO	API/GUI Query	YES + Animation	TBD	NO	NO	AR
HiTIDE / L2SS	Direct, API/GUI Query	YES	via API	YES	YES	AR
JSON-LD Tagging	3rd Party Search	NO	NO	NO	NO	FA

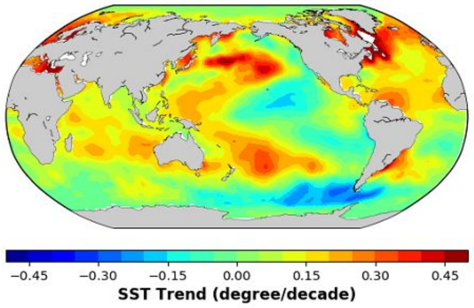


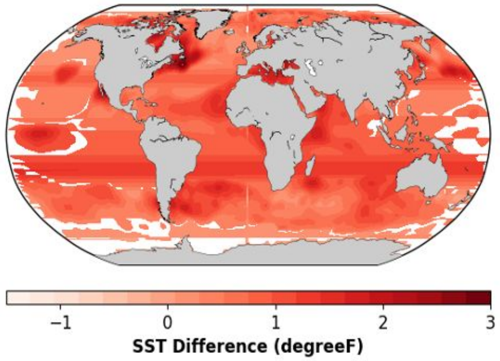


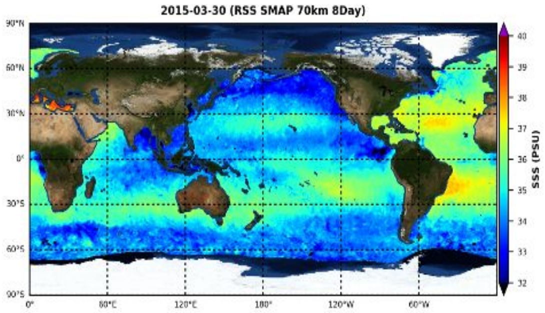


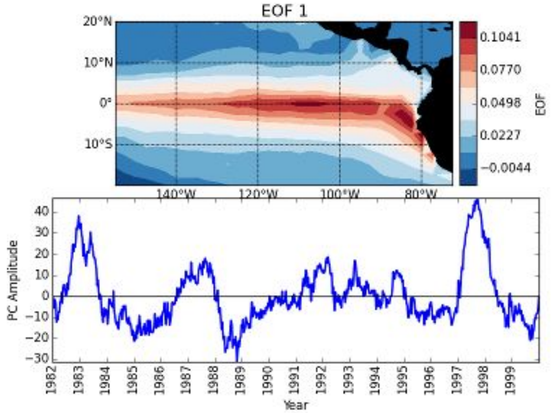
- In progress: Evaluating ERDDAP and improving SOTO v5 analytics

Data Recipes on PO.DAAC forum







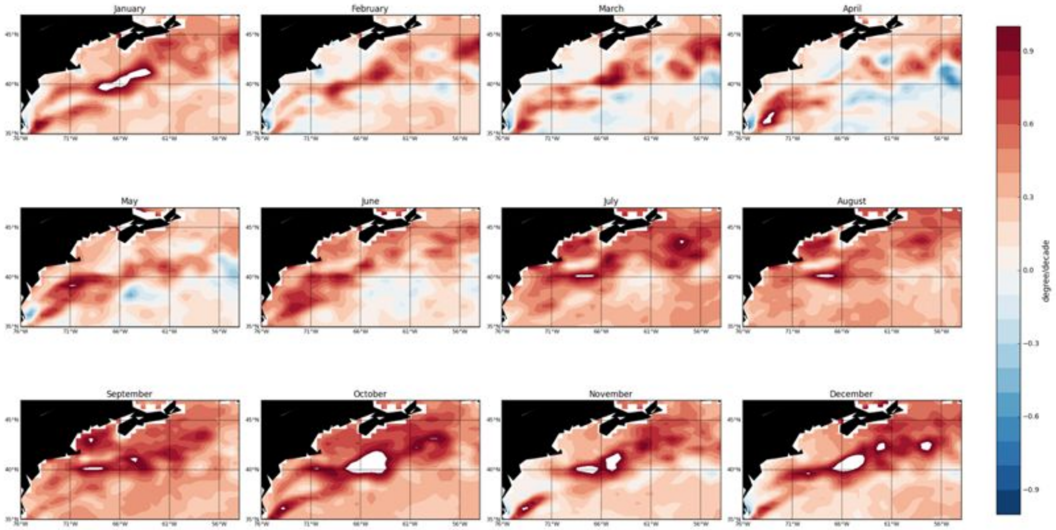


- Data Recipes - https://podaac.jpl.nasa.gov/recipes
    - Reading/Translation (5),
    - Access/Services (30)
    - Visualization (13)
    - Numerical Analysis (8)
    - Tutorials (3)

Ocean Phenology toolkit

- Python based recipe for SST/Ocean Color phenology (change) detection and characterization
  - https://github.jpl.nasa.gov/ybj/PhenologyPy/tree/master
  - In principle can be extended to any long time series gridded satellite measurement with proper configuration (e.g., ocean wind, precipitation, sea level etc.)
- Inputs:
  - SST/OC L4/L3 time series
  - Configuration parameters: time/space/climatology/phenology thresholds
- Output Phenology indices (for SST):
  - Summer/Winter Start/End Date and Duration
  - Spring Transition Start Date
  - Start/End Data trends
  - Seasonal/Annual trends
  - Warmest/Coolest day of year

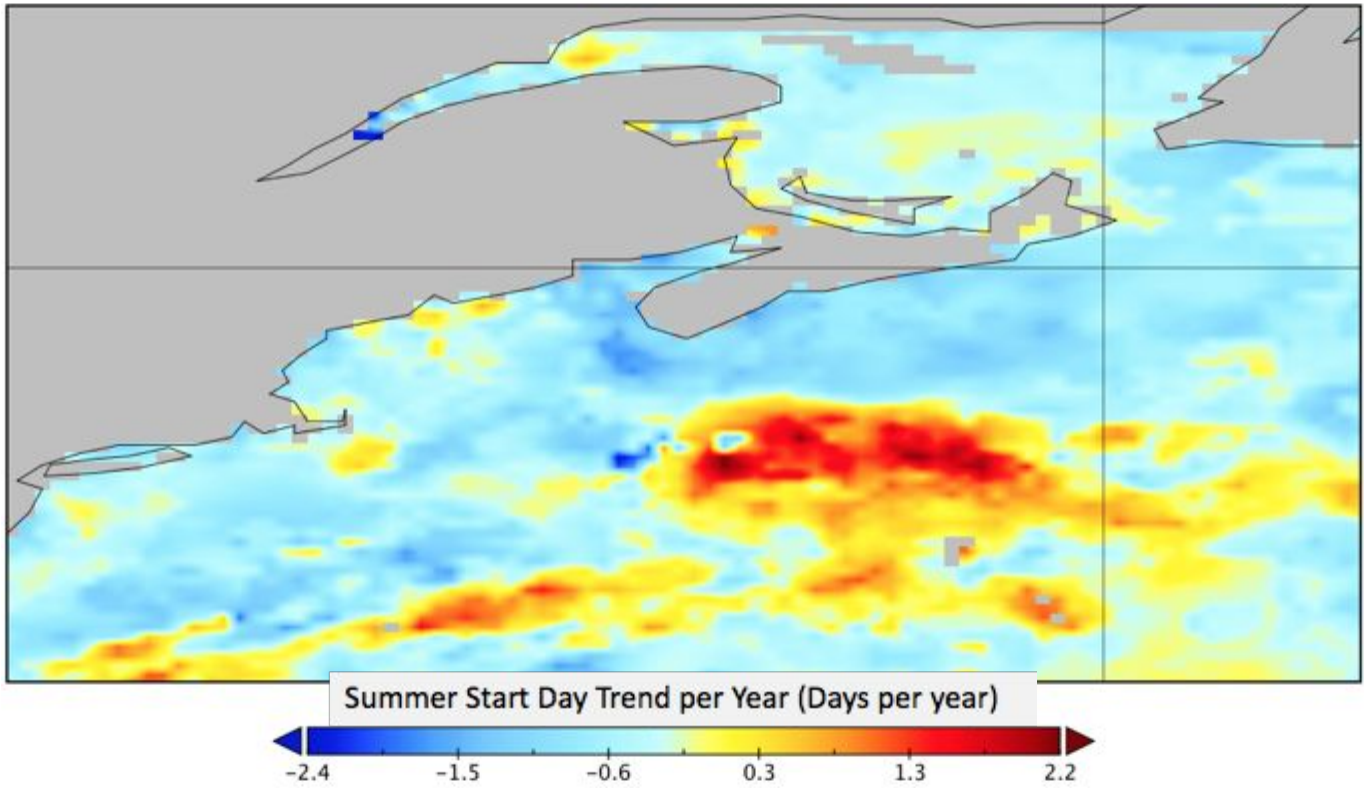
NetCDF container outputs



Monthly rates of SST trend in the NE Atlantic region using the entire NCEI dataset (30 year times series)

Phenology trends and longterm change

Trends in the **Start Day of the summer season** in the New England coastal region based on the CMC0.2deg dataset (from 1992-2016). In this coastal region there is an overall trend to the summer season starting **earlier and earlier**. This combined with summer duration output can be used to assess the impact of regional warming and change.



SST Analysis using NCO + python

- NCO toolkit: netCDF Command Operators
- PO.DAAC forum post: https://podaac.jpl.nasa.gov/forum/viewtopic.php?f=87&t=700
- Reproducing figures from 4th National Climate Assessment Report using the NOAA ERSST v5 dataset

Sea Surface Temperature Analysis Using NCO Utilities

Forum home ‹ PODAAC Forums ‹ Data Recipes ‹ Numerical Analysis

POSTREPLY | Search this topic... | Search | 2 posts • Page 1 of 1

Sea Surface Temperature Analysis Using NCO Utilities QUOTE

by **yiboj** » Wed Oct 25, 2017 10:11 am

NCO utilities are the versatile tools for simple and quick manipulation on netCDF data files including numerical analyses with mathematical and statistical algorithms of the GSL (GNU Scientific Library) package. Users can consult with the online NCO help manual (http://nco.sourceforge.net/) for details. Other NCO script tricks can be found in another post on this Forum "More Examples of Using the NCO Toolkit for Oceans Data".

Inspired by a recent climate change report, *Climate Science Special Report, Fourth National Climate Assessment*, we have developed a data recipe for the analysis of global Sea Surface Temperature (SST) differences and long term trends by using the NCO utilities. This recipe demonstrates the power and elegance of using this toolkit to process satellite data. NCO always produces netCDF output files which can be plotted using a popular program like Panoply.

yiboj
Posts: 109
Joined: Mon Mar 30, 2015 11:22 am

Step 0: PO.DAAC web services retrieve the ERSST filenames necessary for the time periods of interest
Step 1: NCO creates a monthly climatology
Step 2: NCO computes monthly averages in individual years of 1986 to 2015 and then calculates the monthly anomaly using the climatology from the step 1
Step 3: NCO performs the linear regression across anomaly time series on a pixel-by-pixel basis
Step 4: NCO finds (and appends) the regression end points in time to determine long term SST changes

The climate_linear_trend.csh script is shown below:

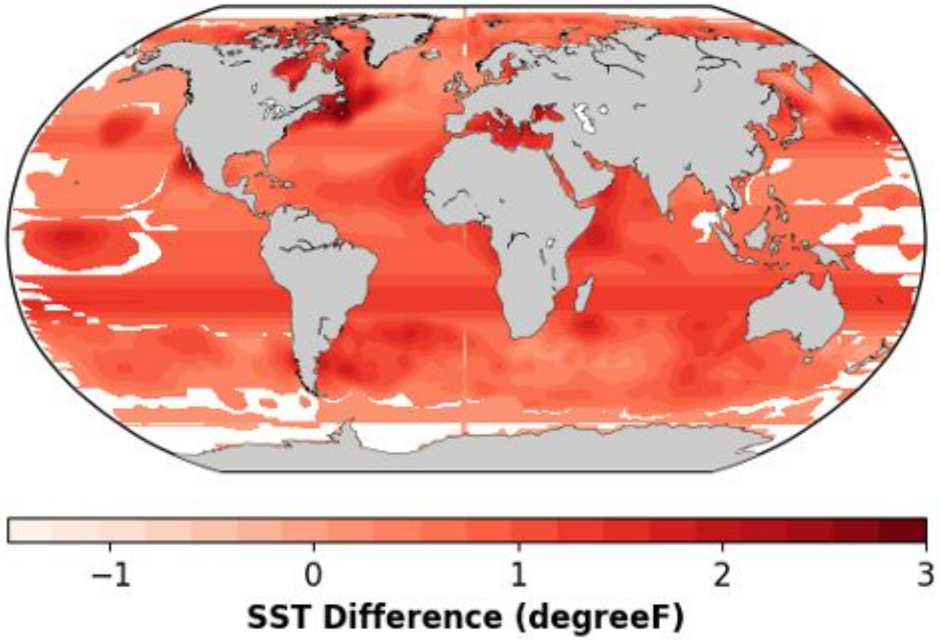
CODE: SELECT ALL

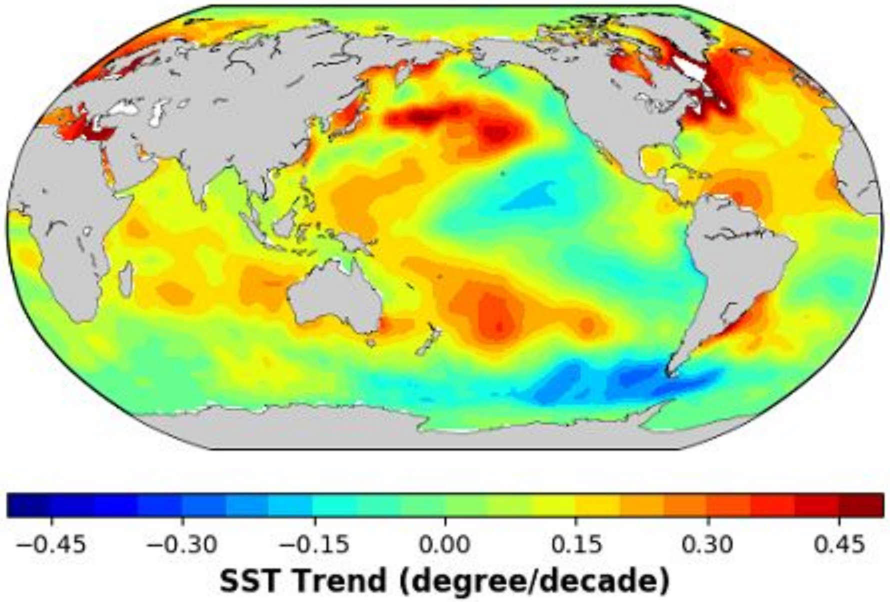
```
#! /bin/tcsh

# ersst_climate_linear_trend.csh
# 20 Sept 2017
# Ed Armstrong NASA/JPL/CalTech

# Calculate the difference in global anomaly temperature using NCO between
1986 and 2015 based on
# different input SST datasets.  A linear fit is made on a pixel-by-pixel
basis for
# the annual anomaly SST from 1986 to 2015. ERSST is used to calculate the
climatology
# between 1901 and 1960
```

Outputs: SST trends and differences





- Global difference in SST between the average of a recent 1986 to 2015 period vs. the average of an earlier 1901 to 1960 period. Reproduces ocean signal figure from Assessment Report
- Longterm SST trend 1901-2015

podaacpy - interacting with PO.DAAC services

- A python module for interacting with PO.DAAC web services for search and discovery, and extraction (and more)
- What are its capabilities ?
  - Dataset Metadata - retrieves the metadata of a dataset
  - Granule Metadata - retrieves the metadata of a granule
  - Search Dataset - searches PO.DAAC's dataset catalog, over Level 2, Level 3, and Level 4 datasets
  - Search Granule - does granule searching on PO.DAAC level 2 swath datasets (individual orbits of a satellite), and level 3 & 4 gridded datasets (time averaged to span the globe)
  - Image Granule - renders granules in the PO.DAAC's catalog to images such as jpeg and/or png
  - Extract Granule - subsets a granule in PO.DAAC catalog and produces either netcdf3 or hdf4 files
  - Metadata Compliance Checker: an online tool and web service designed to check and validate the contents of netCDF and HDF granules for the Climate and Forecast (CF) and Attribute Convention for Dataset Discovery (ACDD) metadata conventions.
  - Level 2 Subsetting: allows users to subset and download popular PO.DAAC level 2 (swath) datasets.

  - NASA OceanColor Web:
  - File Search - locate publically available files within the NASA Ocean Data Processing System (ODPS)
  - Bulk data downloads via HTTP - mimic FTP bulk data downloads using the HTTP-based data distribution server.

podaacpy in Jupyter

- Run podaacpy as a Jupyter notebook recipe
  - https://github.com/nasa/podaacpy/tree/master/examples
- ASCAT L2 example
- But easy to extend to GHRSST datasets and their variables
- Demonstrates the chaining capability.
  - Output of one service the input to another

```
In [ ]: from IPython.display import Image
        Image(filename='ASCAT_geometry.jpg')

In [ ]: #First lets import the libraries we require
        from pprint import pprint
        from podaac import podaac as podaac
        from podaac import podaac_utils as utils
        from podaac import drive as drive

In [ ]: #Then we can create instances of the classes we will use
        p = podaac.Podaac()
        u = utils.PodaacUtils()
        d = drive.Drive('podaac.ini', None, None)

In [ ]: # Let's discover PO.DAAC Wind data relating to Hurricane Florence, which
        #  was a very recent major hurricane to impact Southeastern US
        #    https://en.wikipedia.org/wiki/Hurricane_Florence
        # Using specific parameters to confine the discovery space, we opt for the full
        # metadata record in atom format
        ds_result = p.dataset_search(keyword='ASCAT',
                                     start_time='2018-09-12T00:00:01Z',
                                     end_time='2018-09-14T11:59:59Z',
                                     short_name='ASCATA-L2-Coastal',
                                     process_level='2',
                                     bbox='-81,28,-67,40',
                                     pretty='True',
                                     _format='atom',
                                     full='True')
        print(ds_result)

In [ ]: #Because we requested the Full response, we can actually extract the
        # PO.DAAC Drive URL for all granules contained within this dataset.
        search_str = 'https://podaac-tools.jpl.nasa.gov/drive/files/'
        drive_path = [ str(i) for i in ds_result.strip().split() if search_str in i ][0]
        print(drive_path[5:])

In [ ]: #Next, lets search for Granules of interest relating to the above discovery operation
        #Lets execute a search for specific granules from the following dataset
        # MetOp-A ASCAT Level 2 Ocean Surface Wind Vectors Optimized for Coastal Ocean
        # https://podaac.jpl.nasa.gov/dataset/ASCATA-L2-Coastal
        # ...based upon temporal (start and end) and spatial contraints.
        result = p.granule_search(dataset_id='PODAAC-ASOP2-12C01',
                                  start_time='2018-09-12T00:00:01Z',
                                  end_time='2018-09-14T11:59:59Z',
                                  bbox='-81,28,-67,40',
                                  sort_by='timeAsc',
                                  items_per_page='400',
                                  _format='atom')
        #print(result)
        searchStr = 'totalResults'
        numResultsStr = [ str(i) for i in result.strip().split() if searchStr in i ]
        print(numResultsStr)
```

PO.DAAC Recipes on GitHub

- https://github.com/nasa/podaac_tools_and_services
- Focuses on public-facing tools and services
- All software/code is licensed as open source under the Apache 2.0 license.
- Home for podaacpy but many other recipes need to be migrated

common-mapping-client @ 48575cb	Update submodules	3 months ago
data_animation	Update repository with all current PO.DAAC tooling	7 months ago
incubator-sdap-mudrod @ b3cd3ff	Update submodules	3 months ago
incubator-sdap-nexus @ 564bd02	Update submodules	3 months ago
modis_time	Update repository with all current PO.DAAC tooling	7 months ago
onearth @ 7ea3b6c	Update submodules	3 months ago
podaacpy @ 4fbebe6	Update submodules	3 months ago
read_ASCAT	Update repository with all current PO.DAAC tooling	7 months ago
read_geotiff	Update repository with all current PO.DAAC tooling	7 months ago
read_nc_py	Update repository with all current PO.DAAC tooling	7 months ago
retreive_CYGNSS	Update repository with all current PO.DAAC tooling	7 months ago
subset_GHRSST	Update repository with all current PO.DAAC tooling	7 months ago
subset_opendap	Update repository with all current PO.DAAC tooling	7 months ago
subset_opendap_matlab	Update repository with all current PO.DAAC tooling	7 months ago
subset_w10n	Update repository with all current PO.DAAC tooling	7 months ago
write_netcdf	Update repository with all current PO.DAAC tooling	7 months ago
.gitmodules	Push new entries to the registry	9 months ago
LICENSE	Initial commit	10 months ago
NOTICE	Update repository with all current PO.DAAC tooling	7 months ago
README.rst	Update submodules	3 months ago

What's in this repository?

This repository reflects an active catalog of all tools and services software pertaining to PO.DAAC data access. If you have a suggestion for a new tool or would like to update the content here, please open an issue or send a pull request.

Where do I find detailed information on tools and services included in this repository?

Each repository has it's own README file e.g. data_animation/README.rst

Keeping Git submodules up-to-date

In order to keep the submodules as defined in [.gitmodules](https://github.com/nasa/podaac_tools_and_services/blob/master/.gitmodules) up-to-date it is necessary to periodically push updates. You can safely execute this command to do so:

```
$ git submodule foreach git pull origin master
$ git status //you will then see the changes which have been mode
$ git add -A
$ git commit -m "Update submodules"
$ git push origin master
```

Cloud based AWS MUR Zarr

- Created by Chelle Gentemann et al.
- Part of the Amazon Web Service (AWS) Open Data Registry on their cloud platform
  - https://registry.opendata.aws/mur/
- GitHub: https://github.com/pangeo-gallery/osm2020tutorial
  - Provides a complete ecosystem based on Pangeo, Xarry, Dask, Jupyter and Binder to manipulate the MUR Zarr dataset on a few or >100 CPUs
- Compute credits provided by Amazon
- Supported by the NASA IMPACT program

AWS MUR Zarr analysis

- Existing Jupyter notebook
  - https://github.com/pangeo-gallery/osm2020tutorial
- Demonstrate SST time series, climatology and SST anomaly analyses and more on the complete MUR dataset in minutes vs hours

2. Explore the data

Let's explore the data

- look at all the SST data
- look at the SST data masked to only ocean and ice-free data
- With all data, it is important to explore it and understand what is contains before doing an analysis.
- The ice mask used by MUR SST is from NSIDC and is based on satellite passive microwave estimates of sea ice concentration
- The satellite data isn't available near land, so the is no estimate of sea ice concentration near land
- For this data, it means that there are some erroneous SSTs near land, that is likely ice and this is something to be aware of.

In [ ]:
```
sst = ds_sst['analysed_sst']

cond = (ds_sst.mask==1) & ((ds_sst.sea_ice_fraction<.15) | np.isnan(ds_sst.sea_ice_fraction))

sst_masked = ds_sst['analysed_sst'].where(cond)

sst_masked
```

Using `.groupby` and `.resample`

Xarray has a lot of nice build-in methods, such as .resampe which can upsample or downsample data and .mean. Here we use these to calculate a climatology and anomaly.

Create a daily SST anomaly dataset

- Calculate the daily climatology using `.groupby`
- Calculate the anomaly

Create a monthly SST anomaly dataset

- First create a monthly version of the dataset using `.resample`. Two nice arguments for `.resample`: `keep_addrs` which keeps the metadata and `skipna` which ensures that only data that is always present is included
- Calculate the monthly climatology using `.groupby`
- Calculate the anomaly

In [ ]:
```
%%time
#create a daily climatology and anomaly
climatology_mean = sst_masked.groupby('time.dayofyear').mean('time',keep_attrs=True,skipna=False)

sst_anomaly = sst_masked.groupby('time.dayofyear')-climatology_mean  #take out annual mean to remo
ve trends

#create a monthly dataset, climatology, and anomaly
sst_monthly = sst_masked.resample(time='1MS').mean('time',keep_attrs=True,skipna=False)

climatology_mean_monthly = sst_monthly.groupby('time.month').mean('time',keep_attrs=True,skipna=Fa
lse)
```

Emerging PO.DAAC capabilities

- PO.DAAC is creating a "test bed" repository for select oceanographic datasets in the AWS cloud (S3 buckets)
    - Datasets and parameters include:
        - GHRSST MODIS Terra L2P, ECCO model SST
        - Jason-1 and MEaSUREs SSH and SSHA
        - SMAP SSS
    - Developing Jupyter/Binder based tutorials to operate on these datasets
    - Availability Summer 2020
    - See online webinar https://earthdata.nasa.gov/learn/user-resources/webinars-and-tutorials/webinar-podaac-18-mar-2020
- Future PO.DAAC missions and datasets will be 100 % cloud-based
        - ECCO II (late 2020)
        - Sentinel-6 (early 2021)
        - SWOT mission (late 2021)
    - PO.DAAC has developed a complete end-to-end dataset search/discovery/transform/subset/access service capability for the SWOT mission that will be extended to other PO.DAAC datasets

Summary

- PO.DAAC has developed many data and services recipes as stand alone programs in familiar languages like python and NCO. Often GHRSST datasets are used in these recipes.
- Recipes are being migrated to the open source GitHub/Jupyter/Binder ecosystems including using cloud data storage
    - cloud optimization results in vastly improving the compute scale necessary for science and analysis
- PO.DAAC is evolving to a location agnostic data services model where consumers will interact/transform/extract data in cloud storage
    - Many tools and services have been developed or are in construction to do this
    - More datasets available in the cloud soon

Thanks !