

### Accessing OOI Data via THREDDS and ERDDAP

June 7, 2022 – Northeast Pacific Workshop

Christopher Wingard



INSF



DCEANOBSERVATORIES.ORG

## Housekeeping

- This is intended to be an interactive session. Questions? Please, do not hesitate to ask!
  - Raise your hand (real or virtual)
  - Post to the chat
  - Use the Breakout Session Worksheet
  - Visit the "Help Desk" during the Poster Sessions
- Introduction
  - OOI Endurance Array Data Lead (formerly Mooring Instrument Lead)
  - Been a member of OOI since Summer 2010
  - Spent some time as part of the UCSD CI Data Team
  - Helped develop, alongside Stuart Pearce, Russell Desiderio and Craig Risien, the python code used to calculate OOI data products
  - Actively work with OOI data (sensor deep-dives, HITL annotations, QARTOD test limits, etc.)







 $\triangle$ 

# Terminology (Handout: Decoder)

### Array

• One of 5 major research regions that make up OOI (e.g., Global Station Papa (**GP**) or Coastal Endurance (**CE**))

### Site

• A specific geographic location within an array (e.g., Coastal Endurance Oregon Inshore = **CE01**)

### Platform

 An assembly/infrastructure at a site that hosts a complement of integrated scientific instruments. May be fixed (mooring) or mobile (profilers or gliders) (e.g., Coastal Endurance Oregon Inshore Surface Mooring = CE01ISSM)

### Node

 A section of a platform with one or more data loggers and power controllers. Instruments on platforms are plugged into nodes. (e.g., the data logger on the Near Surface Instrument Frame (NSIF, 7 m), part of the mooring riser, of the Coastal Endurance Oregon Inshore Surface Mooring = CE01ISSM-RID16)

### Instrument/Sensor

• Terms often used interchangeably. An instrument is a piece of equipment used to collect data. A sensor is a part of an instrument which measures a specific quantity. Each instrument has a Unique ID (e.g., the dissolved oxygen sensor on the NSIF of the Coastal Endurance Oregon Inshore Surface Mooring = 03-DOSTAD000

### Reference Designator

- Identifies a particular instrument on a particular node/platform at a particular site
- Example: CE01ISSM-RID16-03-DOSTAD000 = Coastal Endurance Oregon Inshore Surface Mooring - Mooring Riser - Dissolved Oxygen Sensor

### Data Delivery Method

- **telemetered**: Data returned wirelessly. May be truncated or decimated due to size.
- **recovered\_host/recovered\_inst**: Data downloaded directly from either the platform computer and/or from the instrument after the system is recovered
- **streamed**: Data accessible in real-time, streamed over the fiber optic network from Cabled Array and select Endurance platforms

### Data Stream Name

• Data feed from a sensor that has been read, parsed, and separated based on content (e.g., "engineering", "science", "metadata", etc.) into a named dataset, or "stream". Stream names are often method specific

### Parameters

- Also sometimes called data variables
- A particular value returned from a sensor (e.g., practical salinity from a CTD). There are multiple parameters in a data stream, some of which may be identified as an OOI Data Product





## Thematic Real-Time Environmental Distributed Data Services (THREDDS)

- M2M Requests (async)
  - User and request specific
  - Temporal organization
  - Limited retention time (currently 6 months, under review)
- Gold Copy THREDDS

DCEAN

NITIATIVE

OBSERVATORIES

- <u>https://thredds.dataexplorer.oceanobservatories.org/thr</u> edds/catalog/ooigoldcopy/public/catalog.html
- Created by internal OOI processes, updated daily
- Organized by reference designator, delivery method and stream
- Covers most data sets provided by OOI through the M2M system (exceptions are "engineering" or "metadata" streams)
- Eliminates the need for most M2M requests
- Basis for the data served via Data Explorer



# Gold Copy THREDDS: Finding Data

### • Browser

- Scroll/CTRL-F to find the telemetered data from the Coastal Endurance Oregon Inshore Surface Mooring DO sensor on the midwater platform (NSIF, 7m)
  - <u>CE01ISSM-RID16-03-DOSTAD000-telemetered-</u> dosta\_abcdjm\_ctdbp\_dcl\_instrument

### Code-based

- <u>Siphon</u>, python software developed by UCAR/Unidata to access remote data servers (initially THREDDS, but expanding)
- Web scraping (<u>Beautiful Soup</u> module for python, webread function in <u>Matlab</u>, or <u>readLines</u> in <u>R</u>)
- URL for an instrument specific dataset can be constructed from the reference designator, data delivery method and stream name

	003/0010	
	â th	edds.dataexplorer.oceanobservatories.org/thredds/catalog/ooigoldcopy/public/catalog.html
		CE01ISSM-MFD37-03-CTDBPC000-telemetered-ctdbp_cdef_dcl_instrument/
		<pre>CE01ISSM-MFD37-03-DOSTAD000-Available-dosta_abcdjm_ctdbp_dcl_instrument/</pre>
		CE01ISSM-MFD37-03-DOSTAD000-recovered_host-dosta_abcdjm_ctdbp_dcl_instrument_recovered/
	_	CE01ISSM-MFD37-03-DOSTAD000-recovered_inst-dosta_abcdjm_ctdbp_instrument_recovered/
		CE01ISSM-MFD37-03-DOSTAD000-telemetered-dosta_abcdjm_ctdbp_dcl_instrument/
		CE01ISSM-MFD37-07-ZPLSCC000-recovered_host-zplsc_c_instrument/
		<pre>CE01ISSM-MFD37-07-ZPLSCC000-telemetered-zplsc_c_instrument/</pre>
		CE01ISSM-RID16-01-OPTAAD000-recovered_host-optaa_dj_dcl_instrument_recovered/
		<u>CE01ISSM-RID16-01-OPTAAD000-telemetered-optaa_dj_dcl_instrument/</u>
	_	CE01ISSM-RID16-02-FLORTD000-recovered_host-flort_sample/
		CE01ISSM-RID16-02-FLORTD000-telemetered-flort_sample/
		CE01ISSM-RID16-03-CTDBPC000-recovered_host-ctdbp_cdef_dcl_instrument_recovered/
		CE01ISSM-RID16-03-CTDBPC000-recovered_inst-ctdbp_cdef_instrument_recovered/
		CE01ISSM-RID16-03-CTDRPC000-telemetered etdbp_edef_del_inctnument/
<		CE011SSM-RID16-03-DOSTAD000-recovered_host-dosta_abcdjm_ctdbp_dcl_instrument_recovered/
	_	CE011SSM-RID10-03-00CTAD200-percovered_inst-dosta_abcdjm_ctdbp_instrument_Percovered/
		CE01ISSM-RID16-03-DOSTAD000-telemetered-dosta_abcdjm_ctdbp_dcl_instrument/
		CE01ISSM-RID16-04-VELPTA000-recovered_host-velpt_ab_dcl_instrument_recovered/
		CE01ISSM-RID16-04-VELPTA000-recovered_inst-velpt_ab_instrument_recovered/
	_	CE01ISSM-RID16-04-VELPTA000-telemetered-velpt_ab_dcl_instrument/
		<pre>CE01ISSM-RID16-05-PC02WB000-recovered_host-pco2w_abc_dcl_instrument_blank_recovered/</pre>
	_	<pre>CE01ISSM-RID16-05-PC02WB000-recovered_host-pco2w_abc_dcl_instrument_recovered/</pre>
		CE01ISSM-RID16-05-PCO2WB000-recovered_inst-pco2w_abc_instrument/
	_	CE01ISSM-RID16-05-PCO2WB000-recovered_inst-pco2w_abc_instrument_blank/
		<pre>CE01ISSM-RID16-05-PC02WB000-telemetered-pco2w_abc_dcl_instrument/</pre>
		<u>CE01ISSM-RID16-05-PC02WB000-telemetered-pco2w_abc_dcl_instrument_blank/</u>
		CE01ISSM-RID16-06-PHSEND000-recovered_host-phsen_abcdef_dcl_instrument_recovered/
		CE01ISSM-RID16-06-PHSEND000-recovered_inst-phsen_abcdef_instrument/
		<pre>CE01ISSM-RID16-06-PHSEND000-telemetered-phsen_abcdef_dcl_instrument/</pre>
		CE01ISSM-RID16-07-NUTNRB000-recovered_host-nutnr_b_dcl_conc_instrument_recovered/
		CE01ISSM-RID16-07-NUTNRB000-recovered_host-nutnr_b_dcl_dark_conc_instrument_recovered/
		CE01ISSM-RID16-07-NUTNRB000-recovered_host-suna_dcl_recovered/
		<pre>CE01ISSM-RID16-07-NUTNRB000-recovered_inst-nutnr_b_dark_instrument_recovered/</pre>
	_	CE01ISSM-RID16-07-NUTNRB000-recovered_inst-nutnr_b_instrument_recovered/
		CE01ISSM-RID16-07-NUTNRB000-recovered_inst-suna_instrument_recovered/
		CE01ISSM-RID16-07-NUTNRB000-telemetered-nutnr_b_dcl_full_instrument/
		CE01ISSM-RID16-07-NUTNRB000-telemetered-suna_dcl_recovered/
		CE01ISSM-RID16-08-SPKIRB000-recovered_host-spkir_abj_dcl_instrument_recovered/
		CE01ISSM-RID16-08-SPKIRB000-telemetered-spkir_abj_dcl_instrument/



## **Gold Copy THREDDS: Datasets**

- Files named by deployment number, RefDes, method, stream, file contents:
  - Annotations: Should include all annotations associated with the reference designator but does not. Users will need to use the M2M system to obtain the full set of annotations
  - NCML: Bug in stream engine used to generate these files, don't use them (other issues as well)
  - NC: netCDF files time stamped based on time coverage in the file. These contain the instrument data
  - Provenance: Firehose of metadata. Includes information about every deployment and every calibration coefficient for every instance of an instrument deployed at the site. In addition, contains information about the request, the parameters and files used in any processing as well as the functions called

### Co-located instrument(s) if required by processing

#### :p://thredds.dataexplo × + ☆ thredds.dataexplorer.oceanol OCEAN **OBSERVATORIES** INITIATIVE Dataset CE01ISSM-RID16-03-D0STAD000-telemetered-dosta\_abcdjm\_ctdbp\_dcl\_instrument CE01ISSM-RID16-03-DOSTAD000-telemetered-dosta abcdjm ctdbp dcl instrument annotations.jso deployment0002\_CE01ISSM-RID16-03-CTDBPC000-telemetered-ctdbp\_cdef\_dcl\_instrument.ncm deployment0002 CE01ISSM-RID16-03-CTDBPC000-telemetered-ctdbp cdef dcl instrument 20141010T183018.184000-20150411T233023.203000.r deployment0002 CE01ISSM-RID16-03-CTDBPC000-telemetered-ctdbp cdef dcl instrument aggregate provenance.jsor deployment0002\_CE01ISSM-RID16-03-DOSTAD000-telemetered-dosta\_abcdjm\_ctdbp\_dcl\_instrument.ncml deployment0002 CE01ISSM-RID16-03-DOSTAD000-telemetered-dosta abcdjm ctdbp dcl instrument 20141010T183018.184000-20150411T233023.203000. deployment0002 CE01ISSM-RID16-03-DOSTAD000-telemetered-dosta abcdjm ctdbp dcl instrument aggregate provenance.json deployment0003 CE01ISSM-RID16-03-CTDBPC000-telemetered-ctdbp cdef dcl instrument.ncm deployment0003 CE01ISSM-RID16-03-CTDBPC000-telemetered-ctdbp cdef dcl instrument 20150603T182918.940000-20150825T002916.970000.nv deployment0003\_CE01ISSM-RID16-03-CTDBPC000-telemetered-ctdbp\_cdef\_dcl\_instrument\_aggregate\_provenance.json deployment0003 CE01ISSM-RID16-03-DOSTAD000-telemetered-dosta abcdim ctdbp dcl instrument.ncml deployment0003 CE01ISSM-RID16-03-DOSTAD000-telemetered-dosta abcdim ctdbp dcl instrument 20150603T182918.940000-20150825T002916.970000.n deployment0003 CE01ISSM-RID16-03-DOSTAD000-telemetered-dosta abcdim ctdbp dcl instrument aggregate provenance.iso deployment0004 CE01ISSM-RID16-03-CTDBPC000-telemetered-ctdbp cdef dcl instrument.ncm deployment0004 CE01ISSM-RID16-03-CTDBPC000-telemetered-ctdbp cdef dcl instrument 20151008T192918.398000-20160131T212930.268000.nv deployment0004 CE01ISSM-RID16-03-CIDBPC000-telemetered-ctdbp\_cdef\_dcl\_instrument\_aggregate\_provenance.iso deployment0004 CE01ISSM-RID16-03-DOSTAD000-telemetered-dosta abcdim ctdbp dcl instrument.ncm deployment0004 00STAD000-telemetered-dosta abcdim ctdbp dcl instrument 20151008T192918.398000-20160131T212930.268000.n deployment0004 CE01ISSM-RID16-03-DOSTAD000-telemetered-dosta abcdjm ctdbp dcl instrument aggregate provenance.jso deployment0005 CE01ISSM-RID16-03-CTDBPC000-telemetered-ctdbp cdef dcl instrument.ncm deployment0005 CE01ISSM-RID16-03-CTDBPC000-telemetered-ctdbp cdef dcl instrument 20160518T162924.314000-20160809T002925.085000.n deployment0005 CE01ISSM-RID16-03-CTDBPC000-telemetered-ctdbp cdef dcl instrument aggregate provenance.isor deployment0005 CE01ISSM-RID16-03-DOSTAD000-telemetered-dosta abcdim ctdbp dcl instrument.ncm deployment0005 CE01ISSM-RID16-03-DOSTAD000-telemetered-dosta abcdjm ctdbp dcl instrument 20160518T162924.314000-20160809T002925.085000.r deployment0005 CE01ISSM-RID16-03-DOSTAD000-telemetered-dosta abcdim ctdbp dcl instrument aggregate provenance.jsor deployment0006 CE01ISSM-RID16-03-CTDBPC000-telemetered-ctdbp cdef dcl instrument.ncm deployment0006 CE01ISSM-RID16-03-CTDBPC000-telemetered-ctdbp cdef dcl instrument 20160930T173527.074000-20170201T113526.763000.nd deployment0006 CE01ISSM-RID16-03-CTDBPC000-telemetered-ctdbp cdef dcl instrument aggregate provenance.isor deployment0006 CE01ISSM-RID16-03-DOSTAD000-telemetered-dosta abcdim ctdbp dcl instrument.ncm deployment0006 CE01ISSM-RID16-03-DOSTAD000-telemetered-dosta abcdim ctdbp dcl instrument 20160930T173527.074000-20170201T113526.763000.pc deployment0006 CE01ISSM-RID16-03-DOSTAD000-telemetered-dosta abcdim ctdbp dcl instrument aggregate provenance.jsor deployment0007 CE01ISSM-RID16-03-CTDBPC000-telemetered-ctdbp\_cdef\_dcl\_instrument.ncml

deployment0007\_CE01ISSM-RID16-03-CTDBPC000-telemetered-ctdbp\_cdef\_dcl\_instrument\_20170419T053045.284000-20170621T233540.



# **Gold Copy THREDDS: Downloading**

- Direct downloads via browser
  - OPenDAP: An OPenDAP dataset access form with the OPenDAP link to the data as well as the metadata used to describe the data file
  - NCML: an NcML representation of the dataset
  - UDDC: an evaluation of how well the metadata contained in the dataset conforms to the <u>netCDF Attribute Convention for Data</u> <u>Discovery (NACDD)</u>
  - ISO: an ISO 19115 metadata representation of the dataset
  - HTTPServer: Direct download of the file to your computer
- Via Code







## **Gold Copy THREDDS: Code Examples**

#### Matlab:

https://matlab.mathworks.com/users/wingardc @onid.oregonstate.edu/Published/nep 2022/ matlab thredds erddap.html

#### Python:

https://nbviewer.ipython.org/gist/cwingard/08e d1a10414962d5d51c173b207074e3

 $\leftarrow$   $\rightarrow$  C 🏠 🔒 matlab.mathwerks.com/users/wingardc@onid.oregonstate.edu/Published/nep\_2022/matlab\_thredds\_erddap.html 😰 🏠 💽 🏶 🗖 🚭

#### Using Matlab to Access OOI Data from THREDDS and ERDDAP

The following code provides a basic set of tools a user could use to access data from the OOI THREDDS and ERDDAP servers using Matlab. This example is specifically focused on the OOI Gold Copy THREDDS catalog, the OOI Gold Copy and Data Explorer Merged ERDDAP servers, and the OOI Gilder data available from the GilderDAC

#### Finding the Data

OOI data is organized according to the Reference Designator, data delivery method and the data stream name. For most of this example, we will be working with data from the dissolved Contrains organized accounting to the reference cossignation, data determining internot and the data substrainter. For most oxygen sensor on the midwater platform (aka Near-Suirface instrument Frame (NSIF) at 7 m) of the Coastal Endurance i We'll use the telenctered data contained in the dosta\_abcdjm\_ctdpi\_ctdjm\_ctdpi\_ct. Oregon Inshore Surface Mooring (CE01ISSM-RID16-03-DOSTAD00

% setup defaults to use in subsequent data queries refdes = "CE01ISSM-RID16-03-DOSTAD000"; nethod = "telemetered"; stream = "dosta abcdim ctdbp dcl instrument";

% construct the OOI Gold Copy THREDDS catalog URL for this data set

base unl = uese\_uri = nttps://threads.dataexplorer.oteanobservatories.org/threads/catalog/o url = join([base\_url, join([refdes, method, stream], '-'), '/catalog.html'], ''); url

url = "https

#### Listing Data Files

We can use the webnead function to scrape the above catalog URL for the data files of interest. To do that, we need to create a simple function using the above URL as an input and a rege lag that can be used to specify the specific files we are after. Because this is Matlab, we need to put that fund scroll down to the section of local functions to see the contents of the list files function. Jupyter Notebook Viewer × +

Constructing the tag The tag can be used in many ways. A simple tag like '.\*\.nc\$' would list all of the netCDF files in the catalog.

salinity), there are CTDBP files present in the catalog as well. Since we don't want those, we can change our tag every DOSTA netCDF file in the catalog. If we were after just the files from Deployment 15, then we can tweat

tag = '.\*deployment0015.\*DOSTA.\*\
nc\_files = list\_files(url, tag);
nc\_files(3)

ans = "ooigoldcopy/public/CE01ISSM-RID16-03-DOSTAD000-telemetered-dosta\_abcdjm\_ctdb

#### Loading Data

We can use the Matlab websave function to download netCDF file(s) from the list, and then load the conten data into Matlab, this has proven buggy. Downloading the file and the loading the downloaded file seems to wor dods\_un1 = 'http

nc\_url = join([dods\_url, nc\_files(3)], ``) websave('data.nc', nc unl);

% use custom developed no reader % netCDF file in as a timetable data = mc\_reaver( datains ;;)
data(1:10, {'dissolved\_oxygen', 'temp', 'practical\_salinity'})

	Time	dissolved_caygen	temp	practical_salinity
1	25-Sep-202	227.5812	10.4148	33.3995
2	25-Sep-202	228.4416	10.7999	33.4049
3	26-Sep-202	208.6362	10.5952	33.4210
4	26-Sep-202	195.0380	10.4172	33.4456
5	26-Sep-202	194.0393	10.4353	33.4182
6	26-Sep-202	253.0121	11.0372	33.3162
7	26-Sep-202	270.8489	11.2875	33.2557
8	26-Sep-202	260.0114	11.0878	33.3001
2	26-Sep-202	259.2609	11.1241	33.2580
10	26-Sep-202	271.3849	11.3019	33,2318

% use a for-loop to download all the data files and create a single timetable

#### ✓ - □ × 🗠 🖈 🕼 🖈 🗆 🎬

JUPYTER FAQ </> Q ର 📩

#### Using Python to Access OOI Data from THREDDS and ERDDAP

The following code provides a basic set of tools a user could use to access data from the OOI THREDDS and ERDDAP servers using python. This example is pecifically focused on the OOI Gold Copy THREDDS catalog, the OOI Gold Copy and Data Explorer Merged ERDDAP servers, and the OOI Glider data available

#### Finding the Data

QQI data is organized according to the Reference Designator, data delivery method and the data stream name. For most of this example, we will be working with Constants organized according to the reference designation, data delivery include and use data siteam initiate. In insist or the coaling, we will be writing and the dissolved oxygen sensor on the indivate platform (aka Near-Surface Instrument Frame (NSIF) at 7 m) of the Coastall Endurance Oregon Inshot Surface Monitoring (CE015SH-R1016-69-005TM0066). We'll use the tellmetered data contained in the dosta\_abcdgr\_ctdbg\_dcl\_instrument stream.

In [18]: # Import the tools we are going to need today import io import matplotlib.pyplot as plt # plotting library import marpy as np # numerical library import re import requests import xarray as xr # netCDF library

← → C ☆ a noviewer.ipython.org/gist/cwingard/08ed1a10414962d5d51c173b207074e3

from bs4 import BeautifulSoup

%matplotlib inline

In [1]: # setup defaults to use in subsequent data querie efdes = "CE01ISSM-RID16-03-DOSTAD000" method = "telemetered"; stream = "dosta\_abcdjm\_ctdbp\_dcl\_instrument";

ruct the OOI Gold Copy THREDDS catalog URL for this data set

base\_url = "https://thredds.dataexplorer.oceanobservatories.org/thredd url = base\_url + ('-').join([refdes, method, stream]) + '/catalog.html talog/ooigoldcopy/public/

Out[1]: 'https://thredds.dataexplorer.oceanobservatories.org/thredds/catalog/ooigoldcopy/public/CE01ISSM-RID16-03-DOSTAD000-telemeter

#### Listing Data Files

We can use the Beautiful Soup module to scrape the above catalog for the data files of interest. To do that, we need to create a simple function using the above of as an input and a regex tag that can be used to specify the specific files we are after

In [2]: def list files(unl, tag=n'.\*\.nc\$'):

Function to create a list of the netCDF data files in the THREDDS catalog created by a request to the M2M system.

:param unl: URL to a THREDDS catalog specific to a data request :param tog: regue pattern used to distinguish files of interest :return: list of files in the catalog with the URL path set relative to the catalog

with requests.session() as page = s.get(url).tex

soup = BeautifulSoup(page, 'html.parser') soup = beautrussouppers. pattern = re.comple(tag) nc\_files = [node.get('href') for node in soup.find\_all('a', text-pattern)] nc\_files = [node.get('href') for node in soup.find\_all('a', text-pattern)] nc\_files = [re.sub('catalog.html\?dataset=',
return nc files

#### Constructing the tac

The tag can be used in many ways. A simple tag like r'.\*\.nc\$' would list all of the netCDF files in the catalog. Because the DOSTA requires data from a co





from the GliderDAC

🗂 Jupyter

## **Environmental Research Division's Data Access Program (ERDDAP)**

### Gold Copy ERDDAP

- https://erddapgoldcopy.dataexplorer.oceanobservatories.org/erdd ap/index.html
- netCDF files from the Gold Copy THREDDS catalogs loaded into ERDDAP datasets
- Datasets are organized by reference designator, delivery method, stream and deployment number
- Based on the coded examples above, our dataset ID would be .... missing ....
  - Some 26+ instances of ERDDAP used to represent the GC FRDDAP
  - Memory and management issues
  - Rethinking it

OBSERVATORIES NITIATIVE

Gold Copy THREDDS == full resolution data



- ERDDAP acts as a middleman between you and various remote data servers. When you request data from ERDDAP, ERDDAP reformats the request into the format required by the remote server, sends the request to the remote server, gets the data, reformats the data into the format that you requested, and sends the data to you. You no longer have to go to different data servers to get data from different datasets
- ERDDAP offers an easy-to-use, consistent way to request data: via the OPeNDAP



Protocols are the standards which specify how to

applications.

request data. Different protocols are appropriate for different types of data and for different client

# Data Explorer (Merged) ERDDAP

- <u>https://erddap.dataexplorer.oceanobservatories.org/er</u> <u>ddap/index.html</u>
- Datasets are organized by the reference designator
- Data from different data delivery methods (and streams) is combined into a single, merged time-series
- Select parameters available:
  - <u>CF Standard Name</u> (e.g., moles\_of\_oxygen\_per\_unit\_mass\_in\_sea\_water)
  - OOI Data Product (e.g. DOCONCS)
  - Data Explorer generated QC flags based on stock QARTOD style tests
- High resolution data resampled to max 1 minute resolution
- Overlapping deployments are combined to create a single timeseries
- Not fully 1:1 with the Gold Copy THREDDS catalogs. Some of the more complex data sets are still being worked on







 $\triangle$ 

# **Data Explorer ERDDAP: Finding Data**

- Browser
  - Advanced Search:

https://erddap.dataexplorer.oceanobservatories.org/erd dap/search/advanced.html

• Datasets:

https://erddap.dataexplorer.oceanobservatories.org/erd dap/info/index.html

- Scroll/CTRL-F
- Code
  - <u>RESTful API with extensive documentation</u>
  - Matlab: User needs to construct requests (combination of RESTful API, browser queries or OOI Decoder)
  - Python: erddapy (https://github.com/ioos/erddapy)
  - R: rerrdap (https://github.com/ropensci/rerddap)







## **Data Explorer ERDDAP: Datasets**

- Datasets are currently all represented as tabular, time-series (TableDAP) data
- Organized by Dataset ID (based on Reference Designator)
- Dataset Title human readable version of the ID
- Data access either through the "data" or "graph" links
- Datasets are updated daily from the Gold Copy THREDDS catalog





## **Data Explorer ERDDAP: Downloading**

- Direct downloads via browser
  - Use the Data Access form to determine list of variables to request, set constraints (e.g., time range) and construct the URL for the file type to download.
  - ERDDAP supports 30+ file formats for downloading (in particular: csv, json, netCDF, mat)
  - Copy/paste the generated URL, or click on the Submit button (not shown, just below the File type)
  - Can also generate a URL for downloading from the Make a Graph form
- Via Code



#### ERDDAP > tabledap > Data Access Form @

 Coastal Endurance: Oregon Inshore Surface Mooring: Near Surface Instrument

 Dataset Titl:
 Frame: Dissolved Oxygen 20 3000

 Institution:
 Ocean Observatories Initiative (OOI) (Dataset ID: ool-ce01issm-rid16-03-dostad000)

 Information:
 Summary @ I License @ I FGOC [ISO 19115] Metadata [ Background & ] Files [ Make a graph

Variable @ Check All Uncheck All	Optional Constraint #1 @	Optional Constraint #2 @	Minimum @	Maximum 🛛
✓ time (UTC) Ø	>= ¥ 2022-05-30T20:30:00Z	<= ¥	2014-04-17T22:00:00Z	2022-06-02T20:30:00Z
□ latitude (degrees_north) Ø	>= 🗸	<= V	44.65708	44.65708
longitude (degrees_east) @	>= 🗸	<= •	-124.09447	-124.09447
z (Altitude, m) 🛛	>= 🗸	<= •	0.0	0.0
mole_concentration_of_dissolved_molecular_oxygen_in_sea_water (micromol.L-1) @	>= 🗸	<= ¥	-9.999	663.4801
mole_concentration_of_dissolved_molecular_oxygen_in_sea_water_qc_agg @	>= 🗸	<= ¥		
mole_concentration_of_dissolved_molecular_oxygen_in_sea_water_qc_tests @	>= 🗸	<= ¥		
moles_of_oxygen_per_unit_mass_in_sea_water (micromol.kg-1) @	>= 🗸	<= ¥	-8.4854518482	620.2331300322
moles_of_oxygen_per_unit_mass_in_sea_water_qc_agg @	>= 🗸	<= ¥		
moles_of_oxygen_per_unit_mass_in_sea_water_qc_tests @	>= 🗸	<= ¥		
sea_water_practical_salinity (1e-3) @	>= 🗸	<= ¥	23.0929863398	33.9586257223
sea_water_practical_salinity_qc_agg	>= •	<= V		
sea_water_practical_salinity_qc_tests @	>= 🗸	<= V	4 000000007	
sea_water_pressure (decidars) @	>= •	<= V	1.3899999857	9.2019996643
sea_water_pressure_qc_agg @	>= •	<= V		
sea_water_pressure_qc_tests	>= •	<= V	7.050000705	47 700 4050005
Sea_water_temperature (degree_Cersius) @	>= V	<= V	1.003899760	17.7984306200
sea_water_temperature_qc_agg	>= •	<= v		
sea_water_temperature_qc_tests	>- •			
	>= •		J	
Server-side Functions @				
distinct() Ø				
	")			
File type: (more info)				
nc - Download a flat table-like NatCDE-3 binary file with COARDS/CE/ACDD metadata				
Just generate the LIRI :				
(Documentation / Bypass this form @)				





## **Data Explorer ERDDAP: Examples**

#### Matlab:

https://matlab.mathworks.com/users/wingardc @onid.oregonstate.edu/Published/nep 2022/ matlab thredds erddap.html

#### Python:

https://nbviewer.ipython.org/gist/cwingard/08e d1a10414962d5d51c173b207074e3

 $\leftarrow$   $\rightarrow$  C 🏠 🔒 matlab.mathwerks.com/users/wingardc@onid.oregonstate.edu/Published/nep\_2022/matlab\_thredds\_erddap.html 😰 🏠 💽 🏶 🗖 🚭

#### Using Matlab to Access OOI Data from THREDDS and ERDDAP

The following only provides a basic set of tools a user could use to access data from the COLTHEEDDS and EDDDAD servers using Matlah. This example is specifically focused on the COL Gold Copy THREDDS catalog, the OOI Gold Copy and Data Explorer Merged ERDDAP servers, and the OOI Gilder data available from the GilderDAC

#### Finding the Data

OOI data is organized according to the Reference Designator, data delivery method and the data stream name. For most of this example, we will be working with data from the dissolved Contrains organized accounting to the reference cossignation, data determining internot and the data substrainter. For most oxygen sensor on the midwater platform (aka Near-Suirface instrument Frame (NSIF) at 7 m) of the Coastal Endurance i We'll use the telenctered data contained in the dosta\_abcdjm\_ctdpi\_ctdjm\_ctdpi\_ct. Oregon Inshore Surface Mooring (CE01ISSM-RID16-03-DOSTAD00

% setup defaults to use in subsequent data queries refdes = "CE01ISSM-RID16-03-DOSTAD000"; nethod = "telemetered"; stream = "dosta abcdim ctdbp dcl instrument";

% construct the OOI Gold Copy THREDDS catalog URL for this data set

base unl = base\_url = `https://thredds.dataexplorer.oceanobservatories.org/thredds/catalog/o url = join([base\_url, join([refdes, method, stream], '-'), '/catalog.html'], ''); url

url = "https://th

#### Listing Data Files

We can use the webnead function to scrape the above catalog URL for the data files of interest. To do that, we need to create a simple function using the above URL as an input and a rege tag that can be used to specify the specific files we are after. Because this is Matlab, we need to put that funct scroll down to the section of local functions to see the contents of the list files function. Jupyter Notebook Viewer × +

Constructing the tag The tag can be used in many ways. A simple tag like '.\*\.nc\$' would list all of the netCDF files in the catalog.

salinity), there are CTDBP files present in the catalog as well. Since we don't want those, we can change our tag every DOSTA netCDF file in the catalog. If we were after just the files from Deployment 15, then we can tweat

tag = '.\*deployment0015.\*DOSTA.\*\
nc\_files = list\_files(url, tag);
nc\_files(3)

ans = "ooigoldcopy/public/CE01ISSM-RID16-03-DOSTAD000-telemetered-dosta abcdim ctdb

#### Loading Data

We can use the Matlab websave function to download netCDF file(s) from the list, and then load the conten data into Matlab, this has proven buggy. Downloading the file and the loading the downloaded file seems to wor dods\_un1 = 'http

nc\_url = join([dods\_url, nc\_files(3)], ''); websave('data.nc', nc\_url);

% use custom developed no reader % netCDF file in as a timetable

data = mc\_reaver( datains ;;)
data(1:10, {'dissolved\_oxygen', 'temp', 'practical\_salinity'})

	Time	dissolved_oxygen	temp	practical_salinity
1	25-Sep-202	227.5812	10.4148	33.3995
	25-Sep-202	228.4416	10.7999	33.4049
	26-Sep-202	208.6362	10.5952	33.4210
	26-Sep-202	195.0380	10.4172	33.4458
	26-Sep-202	194.0393	10.4353	33.4182
	26-Sep-202	253.0121	11.0372	33.3162
	26-Sep-202	270.8489	11.2875	33.2557
	26-Sep-202	260.0114	11.0878	33.3001
	26-Sep-202	259.2609	11.1241	33.2580
10	26-Sep-202	271.3849	11.3019	33,2318

% use a for-loop to download all the data files and create a single timetable

JUPYTER FAQ </> - 🛽 🕹

#### Using Python to Access OOI Data from THREDDS and ERDDAP

The following code provides a basic set of tools a user could use to access data from the OOI THREDDS and ERDDAP servers using python. This example is pecifically focused on the OOI Gold Copy\_THREDDS catalog, the OOI Gold Copy and Data Explorer Merged ERDDAP servers, and the OOI Glider data available from the GliderDAC

#### Finding the Data

🗂 Jupyter

QQI data is organized according to the Reference Designator, data delivery method and the data stream name. For most of this example, we will be working with Constants organized according to the reference designation, data delivery include and use data siteam initiate. In insist or the coaling, we will be writing and the dissolved oxygen sensor on the indivate platform (aka Near-Surface Instrument Frame (NSIF) at 7 m) of the Coastall Endurance Oregon Inshot Surface Monitoring (CE015SH-R1016-69-005TM0066). We'll use the tellmetered data contained in the dosta\_abcdgr\_ctdbg\_dcl\_instrument stream.

In [18]: # Import the tools we are going to need today: import io import matplotlib.pyplot as plt # plotting library import marpy as np # numerical library import re import requests import xarray as xr # netCDF library

← → C ☆ a noviewer.ipython.org/gist/cwingard/08ed1a10414962d5d51c173b207074e3

from bs4 import BeautifulSoup

%matplotlib inline

In [1]: # setup defaults to use in subsequent data querie efdes = "CE01ISSM-RID16-03-DOSTAD000" method = "telemetered"; stream = "dosta\_abcdjm\_ctdbp\_dcl\_instrument";

> ruct the OOI Gold Copy THREDDS catalog URL for this data set base\_url = "https://thredds.dataexplorer.oceanobservatories.org/thredd url = base\_url + ('-').join([refdes, method, stream]) + '/catalog.html stalog/ooigoldcopy/public/

Out[1]: 'https://thredds.dataexplorer.oceanobservatories.org/thredds/catalog/ooigoldcopy/public/CE01ISSM-RID16-03-DOSTAD000-telemeter

#### Listing Data Files

We can use the Beautiful Soup module to scrape the above catalog for the data files of interest. To do that, we need to create a simple function using the above un as an input and a regex tag that can be used to specify the specific files we are after

In [2]: def list files(unl, tag=n'.\*\.nc\$'):

Function to create a list of the netCDF data files in the THREDDS catalog created by a request to the N2M system.

:param unl: URL to a THREDDS catalog specific to a data request :param tog: regue pattern used to distinguish files of interest :return: list of files in the catalog with the URL path set relative to the catalog

with requests.session() as a page = s.get(url).tex

soup = BeautifulSoup(page, 'html.parser') soup = beautrussouppers. pattern = re.comple(tag) nc\_files = [node.get('href') for node in soup.find\_all('a', text-pattern)] nc\_files = [node.get('href') for node in soup.find\_all('a', text-pattern)] nc\_files = [re.sub('catalog.html\?dataset=',
return nc files

#### Constructing the tac

The tag can be used in many ways. A simple tag like r'.\*\.nc\$' would list all of the netCDF files in the catalog. Because the DOSTA requires data from a co



## **GliderDAC ERDDAP**

- https://gliders.ioos.us/erddap/index.html
- Includes both the full-resolution, recovered data, as well as the decimated, telemetered data
- All OOI data is pushed to the GliderDAC on a regular schedule (actively deployed gliders are updated regularly throughout the day)
- Dataset IDs are organized by glider name and deployment date/time ('-delayed' appended to recovered datasets)
- Use 'OOI Coastal Endurance' or 'OOI Global Papa' as search terms
- For python users, see this <u>nice example on</u> <u>using erddapy</u> to access data from the GliderDAC

DCEAN

OBSERVATORIES





## **THREDDS versus ERDDAP?**

- Not really a true cross-comparison
- They both represent a way of accessing OOI data via a users preferred set of tools (GUI, python, Matlab, R ... mix and match, you choose)
- They both have well documented APIs and are commonly used in the bio-geosciences (resources available)
- They offer different variants of mostly the same data
  - THREDDS data is full resolution, covering all the different data delivery methods. User needs to stitch it all back together, but they get everything the system produces. Can be slower, more complex processing required
  - ERDDAP data has been reduced (number of variables, and temporally if data rates greater than 1 minute) but is in a more user-friendly format (already combined, standardized variable names). Data downloads are generally faster
  - Neither has the full set of OOI data, although that is the end goal









## **Questions?**

### **OOI Discourse**

https://discourse.oceanobservatories.org/

### **OOI HelpDesk**

helpdesk@oceanobservatories.org

