**What’s in a netCDF file:**

**To take a quick look at an included netCDF with Matlab:**

*ncn = 'FRF\_wave\_metadata\_CS01-SBE26.nc';* % file name, add full path if not in current directory

*ncdisp(ncn);* % this is a way to take a quick look at what’s in a file, will spit out all attributes, dimensions, and variables (but not the data) to the command line

**Global attributes**: These are meant to apply to all data contained in the file and often include things like where the data was collected (e.g. lat, lon, depth), what instrument (e.g. make, model, serial number), and really anything else you as a data provider would want a user to know (e.g. summary, “This data was collected as part of whatever project, in whatever facility, …). These are specified in a global attributes .yml file. A list of example global yml files is below.

**Dimensions**: These specify the size of your data variables. If you just have time series data, you only need 1 dimension, time. If you have time series profile data (e.g. ADCP) you might have 2 dimensions, time and depth. Not every data variable has to have every dimension. For example, with ADCP data, the pressure data may just have dimension time while current profiles have dimensions time and depth (or distance from transducer or whatever is appropriate). Our convention (and a very common one) is that time is the only unlimited dimension. Unlimited dimensions can change size (for example if you want to append data). The size of all other dimensions is fixed at what you originally specify for a given netCDF. This is all done behind the scenes in the conversion code but is good to be aware of. Note: our files have dimension station\_name\_length which is weird and is something that is for some reason required for our data portal.

**Variables**: The data. Variables also have metadata including size, dimension, datatype, and attributes. The variables to be output as well as the variable attributes are specified in the instrument .yml file. A list of included example instrument yml files is below.

**Note on time:** We have been converting time from Matlab datenum (the time in your data struct should be Matlab datenum) to epoch time (seconds since 1970-01-01 00:00:00). This code automatically does that conversion when generating a netCDF file. I have included epoch2Matlab.m which will convert it back for you.

**yml files**

Basically a text file with standard formatting. I’ve found that most text editors handle them OK but not perfect. Mixed up formatting can cause the netCDF conversion to crash.

**Global yml files:**

Specifies the global attributes. This can include whatever you want it to. One of the requirements for our data to get onto our data portal is that we have *title* as a global attribute. We can remove this from the code but I have not done so. This also makes a dimension called *station\_name\_length* which is a little weird. If this gets on your nerves we can hack it and get rid of this.

A lot of people have spent a lot of time thinking about this. If you want to see some guidance:

<https://www.unidata.ucar.edu/software/thredds/current/netcdf-java/metadata/DataDiscoveryAttConvention.html>

Example files for reference:

*FRF\_wave\_metadata\_CS01-SBE26.yml*

*FRF\_waveMetadata\_xp125m.yml*

*FRF\_current\_metadata\_CS01-ADOP.yml* – note, I don’t think this one will run if you try. The data file I included needs to be run through 1 other program before its ready to covert.

*FRF\_waterlevel\_metadata\_CS01-SBE26.yml*

*FRF\_waterquality\_metadata\_CS03-Microcat.yml*

**Instrument yml files:**

This provides guidance to *matlab2netCDF.m* informing what variables to include from the data struct, any additional attributes found in the data struct that should be added to the global attributes, what dimensions are included, what to call the data struct variables in the netCDF file (you can use this to convert names), and variable attributes.

**Parts of Instrument yml files:**

*\_variables: ['time', 'lat', 'lon', 'station\_name', 'hs', 'fp', 'specS', 'freq', 'depth', 'depthP', 'qcFlagE']*

This specifies which fields to pull from the data struct and make into variables. You don’t have to include all fields in the data struct in the netCDF.

*\_attributes: ['Station', 'manufacturer', 'model', 'serialNumber']*

This specifies which fields to pull from the data struct and add to the global attributes. The name of the field will also be the name of the global attribute.

*\_dimensions: ['time', 'freq','station\_name\_length']*

Specifies dimensions

**Specifying variable names and attributes**

*time:*

*name: 'time'*

*units: 'seconds since 1970-01-01 00:00:00'*

*standard\_name: 'time'*

*long\_name: 'UTC Sample Time'*

*data\_type: 'f8'*

*dim: ['time']*

*calendar: 'gregorian'*

*fill\_value: '-999'*

This provides information about the variables.

*time:*

Specifies the name of the data field in the data struct

*name: 'time'*

Specifies what to call it in the netCDF file. In this case it is called *time* in the data struct and will be called *time* in the output netCDF file.

Another example:

*hs:*

*name: 'waveHs'*

*units: 'm'*

*standard\_name: 'sea\_surface\_wave\_significant\_height'*

*long\_name: 'Significant Wave Height'*

*data\_type: 'f4'*

*dim: ['time']*

*fill\_value: '-999'*

*short\_name: 'Wave Height'*

*coordinates: 'lat lon'*

*hs:*

Specifies the name of the data field in the data struct

*name: 'waveHs'*

Specifies what to call it in the netCDF file. In this case it is called *hs* in the data struct and will be called *waveHs* in the output netCDF file.

Example files for reference:

*currents.yml*

*waterquality\_Microcat.yml*

*waves1DCS.yml*

**mfiles**

*do\_matlab2netCDF.m*

Script I threw together to run a couple examples. Pretty sure I included all the necessary files. You should be able to change “fld” to specify wherever you put these files and it will run.

*matlab2netCDF.m*

Function that does the conversion from Matlab struct to netCDF.

*ymlGetMeta.m*

Something I wrote that you may find helpful. I use this when I want to put something in a yml file where it gets preserved in my netCDF but also want to use it in my initial processing. For example, an offset, instrument elevation, etc. I use it like this:

*metget = {‘offset’, ’elevation’};* % this specifies what I want to pull from the yml file. In the yml file would look like *offset: value* and *elevation: value*.

*metgot = ymlGetMeta(ymlFile,metget);*

I can put it into my data struct like this:

*for i = 1:length(metget)*

*data.(metget{i}) = metgot{i};*

*end*

*epoch2Matlab.m*

Convert epoch time to Matlab datenum.

The remaining .m files are used by *matlab2netCDF.m.*

**Accessing data from netCDF**

Matlab has built in functions and the help is pretty good. Here are a few examples to get you started:

*ncn = 'FRF\_wave\_metadata\_CS01-SBE26.nc';*  % file name, add full path if not in current directory

*ncdisp(ncn);* % this is a way to take a quick look at what’s in a file, will spit out all attributes, dimensions, and variables (but not the data) to the command line

*t = ncread(ncn, 'time');*% get the time vector

*dn = epoch2Matlab(t);* % convert to Matlab datenum

hs = ncread(ncn,'waveHs'); % read some data, in this case variable waveHs