Pycurrents requires two files as an input: a raw data binary file, and an associated metadata .csv. I am going to create a new folder in the WSL2 file system for my project data metadata.

Here is the location:   
\\wsl.localhost\Ubuntu\home\ethan\adcp\pycurrents\_ADCP\_processing\project\_data

I am starting with the first Gordon’s Bay deployment, and the first thing that I need to do is get the metadata file sorted out. I think I might try and see if AI comes in handy here (I will obviously go through it myself to check).

So ChatGPT is relatively good at extracting and recognizing key variables, but with the free version it can’t do much. It is still useful as a sanity check. I am doing the same with Perplexity and Claude. Both are not as good as ChatGPT.

I am now going to use a combination of setup files and the recommended metadata values from ChatGPT.

| **Name (in template)** | **Value to Enter** |
| --- | --- |
| instrument\_type | Teledyne RDI Workhorse Sentinel 600 kHz |
| instrument\_frequency | 614400 |
| ensemble\_interval | 00:06:00 |
| ping\_interval | 7.5 |
| pings\_per\_ensemble | 48 |
| number\_of\_cells | 25 |
| cell\_size | 1.0 |
| transducer\_depth | 20 |
| deployment\_date | 2014-05-16 08:00:00 |
| salinity | 35 |
| temperature | 14 |
| magnetic\_variation | -23 |
| first\_cell\_range | 1.72 |
| last\_cell\_range | 25.72 |
| max\_range | 43.80 |
| standard\_deviation | 1.01 |
| burst\_duration | 15 |
| time\_between\_bursts | 60 |
| samples\_per\_wave\_burst | 1800 |
| mode | 1 |
| burst\_size | 140480 |
| ensemble\_size | 654 |
| storage\_required | 127.87 |
| power\_usage | 409.01 |

I am editing it as an excel worksheet so that I can have it sync to OneDrive. I will then save the final version as a .csv locally.

This is the website to get all the standard names and codes required for the metadata:

[Index of /data/oceans/nodc/codelists](https://www.ncei.noaa.gov/data/oceans/nodc/codelists/)

I have set the water\_depth and instrument\_depth to 20 m. This is what the setup file has as the transducer\_depth.

I saved a copy of the workbook as a .csv and copied it over to the WSL2 mentioned earlier. I now need to create a modified copy of the create\_nc.py to take my raw data and metadata file for processing.

**Usage:**

1. Launch an Ubuntu terminal and set up the working directory in Jupyter Lab

In Ubuntu:

cd ~/adcp

conda activate adcp37

jupyter lab --port 8888 --no-browser

In Jupyter Lab, launch a terminal and activate the venv.

conda activate adcp37

1. Create a working copy of create\_nc.py in ~/adcp/pycurrents\_ADCP\_processing/pycurrents\_ADCP\_processing

For GB1 I duplicated the original create\_nc.py file and named it create\_nc\_GB1.py

1. Open the file and define the raw ADCP file and associated metadata file

raw data: "\\wsl.localhost\Ubuntu\home\ethan\adcp\pycurrents\_ADCP\_processing\project\_data\CSIR\_RDI\_Sentinel\_FB\_20150520\_V3000.000"

metadata: "\\wsl.localhost\Ubuntu\home\ethan\adcp\pycurrents\_ADCP\_processing\project\_data\CSIR\_RDI\_Sentinel\_FB\_20150520\_V3000\_metadata.csv"

1. Change the destination directory

dest\_dir = ‘GB1\_dir’

1. Run the script

Navigate to ~/adcp/pycurrents\_ADCP\_processing

python pycurrents\_ADCP\_processing/create\_nc\_GB1.py

**Troubleshooting**

This is the first error I get:

Traceback (most recent call last):

File "pycurrents\_ADCP\_processing/create\_nc\_GB1.py", line 24, in <module>

ncnames\_L0 = ADCP\_processing\_L0\_L1.nc\_create\_L0\_L1(in\_file=f, file\_meta=meta, dest\_dir=dest\_dir, level=0)

File "/home/ethan/adcp/pycurrents\_ADCP\_processing/pycurrents\_ADCP\_processing/ADCP\_processing\_L0\_L1.py", line 1031, in nc\_create\_L0\_L1

data = rdiraw.rawfile(in\_file, meta\_dict['model'], trim=True)

File "/home/ethan/adcp/pycurrents/pycurrents/adcp/raw\_multi.py", line 47, in rawfile

return FileBBWHOS(fname, sonar, trim, yearbase)

File "/home/ethan/adcp/pycurrents/pycurrents/adcp/raw\_rdi.py", line 555, in \_\_init\_\_

trim=trim, yearbase=yearbase)

File "/home/ethan/adcp/pycurrents/pycurrents/adcp/raw\_base.py", line 154, in \_\_init\_\_

self.open()

File "/home/ethan/adcp/pycurrents/pycurrents/adcp/raw\_rdi.py", line 582, in open

FileBase.open(self)

File "/home/ethan/adcp/pycurrents/pycurrents/adcp/raw\_base.py", line 196, in open

self.header.read(self.fobj)

File "/home/ethan/adcp/pycurrents/pycurrents/adcp/raw\_rdi.py", line 514, in read

raise RuntimeError('first 6 bytes of header parse as %s'%str(tup))

RuntimeError: first 6 bytes of header parse as (127, 121, 86, 1)

I have a feeling that it might have something to do with the segment indices being incorrect, but who knows.

I think I am going to see if changing the model makes any difference, but I just don’t know what models the software can take.

1. Check the list of available models here: pycurrents/pycurrents/adcp/raw\_multi.py

The raw\_multi might not be useful because it is designed to read single-ping raw data from RDI ADCPs

'''

Functions and a class (Multiread) for reading one or more files of raw

(single-ping) data from RDI or Simrad ADCPs.

'''

1. Check the file format

(adcp37) ethan@bigtop:~/adcp/pycurrents\_ADCP\_processing$ hexdump -C project\_data/CSIR\_RDI\_Sentinel\_FB\_20150520\_V3000.000 | head

00000000 7f 79 56 00 00 01 08 00 03 01 10 1c cb 41 1e 60 |.yV..........A.`|

00000010 09 64 00 32 00 40 38 d2 00 05 c0 01 01 02 70 00 |.d.2.@8.......p.|

00000020 00 00 00 00 00 00 00 00 00 00 00 00 01 02 70 00 |..............p.|

00000030 00 00 00 00 00 00 00 00 00 00 00 00 14 0f 05 14 |................|

00000040 08 00 00 00 01 00 00 00 fb 00 00 02 fa d3 13 09 |................|

00000050 26 07 de 6e 02 c0 0c 0c 7f 79 4e 00 00 01 08 00 |&..n.....yN.....|

00000060 03 02 01 00 00 00 00 00 6c 00 00 00 5a 08 00 00 |........l...Z...|

00000070 51 08 00 00 74 08 00 00 50 08 00 00 00 80 00 80 |Q...t...P.......|

00000080 00 80 00 80 00 80 00 80 00 80 00 80 00 80 00 80 |................|

\*

I did some searching and found that the file signature does not match RDI PD0 format, but I already knew that. The signature suggests that I am dealing with Teledyne Sentinel V binary format instead. I am not sure if I trust this because the instrument is a Workhorse. If it turns out to be Sentinel V format it might cause problems because it is a proprietary format.

The problem is that pycurrents.adcp.raw\_rdi expects PD0 or single-ping RDI formats and cannot parse whatever I have in its current state. I might be able to solve this by changing the instrument type in the metadata to Sentinel V or Sentinel. I am going to try that first.

Options:

1. **Use Teledyne software to export**

If you have access to the raw .000 file and **Velocity**, you may:

* Export to **PD0** for pycurrents

I have requested an activation code for the velocity software, now we wait and see. I checked the processed data and there are PD0 files, which I think come from TM Velocity software. The only way might be to export the raw data from there to a PD0 and then import that into pycurrents for further processing.