Tsdat: An Open-Source Data Standardization Framework for Marine Energy and Beyond

Preprint

Carina Lansing,[1] Maxwell Levin,[1] Chitra Sivaraman,[1] Rebecca Fao,[2] and Frederick Driscoll[2]

1 Pacific Northwest National Laboratory
2 National Renewable Energy Laboratory

Presented at Oceans Conference
San Diego, California
September 20–23, 2021

NREL is a national laboratory of the U.S. Department of Energy
Office of Energy Efficiency & Renewable Energy
Operated by the Alliance for Sustainable Energy, LLC

This report is available at no cost from the National Renewable Energy Laboratory (NREL) at www.nrel.gov/publications.

Contract No. DE-AC36-08GO28308

Conference Paper
NREL/CP-5700-80561
January 20222

Tsdat: An Open-Source Data Standardization Framework for Marine Energy and Beyond

Preprint

Carina Lansing,[1] Maxwell Levin,[1] Chitra Sivaraman,[1] Rebecca Fao,[2] and Frederick Driscoll[2]

1 Pacific Northwest National Laboratory
2 National Renewable Energy Laboratory

Suggested Citation
Lansing, Carina, Maxwell Levin, Chitra Sivaraman, Rebecca Fao, and Frederick Driscoll. 2022. Tsdat: *An Open-Source Data Standardization Framework for Marine Energy and Beyond: Preprint*. Golden, CO: National Renewable Energy Laboratory. NREL/CP-5700-80561. https://www.nrel.gov/docs/fy22osti/80561.pdf.

NREL is a national laboratory of the U.S. Department of Energy
Office of Energy Efficiency & Renewable Energy
Operated by the Alliance for Sustainable Energy, LLC

This report is available at no cost from the National Renewable Energy Laboratory (NREL) at www.nrel.gov/publications.

Contract No. DE-AC36-08GO28308

Conference Paper
NREL/CP-5700-80561
January 2022

National Renewable Energy Laboratory
15013 Denver West Parkway
Golden, CO 80401
303-275-3000 • www.nrel.gov

NOTICE

This work was authored in part by the National Renewable Energy Laboratory, operated by Alliance for Sustainable Energy, LLC, for the U.S. Department of Energy (DOE) under Contract No. DE-AC36-08GO28308. Funding provided by the U.S. Department of Energy Office of Energy Efficiency and Renewable Energy Water Power Technologies Office. The views expressed herein do not necessarily represent the views of the DOE or the U.S. Government. The U.S. Government retains and the publisher, by accepting the article for publication, acknowledges that the U.S. Government retains a nonexclusive, paid-up, irrevocable, worldwide license to publish or reproduce the published form of this work, or allow others to do so, for U.S. Government purposes.

Cover Photos by Dennis Schroeder: (clockwise, left to right) NREL 51934, NREL 45897, NREL 42160, NREL 45891, NREL 48097, NREL 46526.

Tsdat: An Open-Source Data Standardization Framework for Marine Energy and Beyond

Carina Lansing
Advanced Computing, Mathematics, and Data Division
Pacific Northwest National Laboratory
Richland, U.S.A.
https://orcid.org/0000-0002-2812-246X

Maxwell Levin
Advanced Computing, Mathematics, and Data Division
Pacific Northwest National Laboratory
Richland, U.S.A.
https://orcid.org/0000-0002-2536-8093

Chitra Sivaraman
Advanced Computing, Mathematics, and Data Division
Pacific Northwest National Laboratory
Richland, U.S.A.
https://orcid.org/0000-0001-9013-427X

Rebecca Fao
Water Power, Energy Conversion & Storage Systems
National Renewable Energy Laboratory
Golden, U.S.A.
https://orcid.org/0000-0002-1779-2302

Frederick Driscoll
Water Power, Energy Conversion & Storage Systems
National Renewable Energy Laboratory
Golden, U.S.A.
https://orcid.org/0000-0002-3490-656X

***Abstract*— Many organizations are tasked with the collection and processing of large quantities of data from various measurement devices. Data reported from these sources are often not interoperable with datasets and software used by analysts and other organizations in the same domain, introducing barriers for collaboration on large-scale projects. This poses a particular problem for cross-device comparisons and machine learning applications, which rely on large quantities of data from multiple sources. To address these challenges, the open-source Time-Series Data Pipelines (Tsdat) Python framework was developed by Pacific Northwest National Laboratory, with strategic guidance and direction provided by the National Renewable Energy Laboratory and Sandia National Laboratories to facilitate collaboration and accelerate advancements in the marine energy domain through the development of an open-source ecosystem of tools. This paper will describe the Tsdat framework and the data standards within which it operates. A beta version of Tsdat has been released and is being used by several projects in marine energy, wind energy, and building energy systems.**

Keywords*— *Python, open-source, data standards, data lake, big data, interoperability

I. Introduction

Instrument manufacturers often use unique or proprietary methods of collecting and recording data from their devices, creating operational challenges for organizations and analysts aiming to derive insights from one or more instruments. A variety of file and data formats may be used to store the data after collection; variables measuring the same physical property may not be named consistently across instruments; metadata describing how data are reported may be present only in separate documents that are not readily available for data users; and data gaps, spikes, or other quality issues indicating instrument failure or improper calibration may go unnoticed for long periods of time. When working with unstructured or unstandardized datasets, analysts and data users must navigate all these factors and invest significant time and energy wrangling data into a format suitable for use in their analysis or application. Inconsistencies in the input data can easily lead to many one-off cases in cleaning code, making the codebase difficult to maintain or reuse for other applications or teams in the same organization and causing significant overhead and duplication of effort. This is inefficient and costly.

Many commercial solutions exist for developing data pipelines to read-in raw data and convert it to a standard format usable by scientists, analysts, and other data users. Talend, Domo, and Tableau, for example, are commercial data integration solution providers that offer end-to-end data ingestion pipelines to accelerate data analytics and generate business insights from big data [1,2,3]. These companies clearly provide great value to their corporate customers, but their pricing models are prohibitively expensive for use in research projects or by smaller start-ups. Additionally, their closed-source nature makes it difficult for developers and analysts to share pipeline components across projects and can also present challenges if customizations to the base functionality of the platform are needed.

In addition to commercial data processing and analytics frameworks, the ARM Data Integrator (ADI) data standardization platform has been developed for the Atmospheric Radiation Measurement (ARM) program [4,5]. This platform has been used for nearly 20 years to standardize, curate, and annotate hundreds of atmospheric datastreams continuously collected around the globe. While the ADI framework has flushed out many of the complex issues associated with collecting and standardizing time-series instrument data in a production environment, the framework

itself is complex to set up and customize, and is tied specifically to ARM conventions. A more lightweight, flexible, domain-agnostic framework is needed to support smaller programs and communities such as marine energy.

The Time Series Data Pipelines (Tsdat) Python framework was developed to accelerate advancements in the marine energy domain as part of an open-source ecosystem of tools [6]. Specifically, it was developed to provide an easy-to-use framework for running data ingestion pipelines either locally or in a large-scale production environment such as on the Amazon cloud. More specifically, it was intended to support data submitted to the Marine Hydrokinetic Data Repository (MHKDR) to enable large-scale analysis via a structured data lake as shown in Fig. 1 [7,8].

The rest of this paper provides detailed information about the design of Tsdat and the data standards within which the framework operates. In addition, it demonstrates Tsdat's application to an example problem related to oceanographic buoy data and provides a roadmap for future development.

II. Tsdat Framework

The goal of Tsdat is to provide an easy-to-use framework for creating data processing pipelines in either small-scale local or large-scale production environments. The intent was to develop a framework flexible enough to support a wide variety of data formats and use cases, simple enough that most users could stand up a basic data ingestion pipeline with little to no effort, and intuitive enough that semi-experienced Python developers could quickly find the information they need to create more advanced pipelines.

As illustrated in Fig. 2, Tsdat works by taking input data that can be of any format, applying a data processing pipeline that is declaratively specified using a set of configuration files and/or custom code modules, and then storing the output in a standardized format as specified by another configuration file.

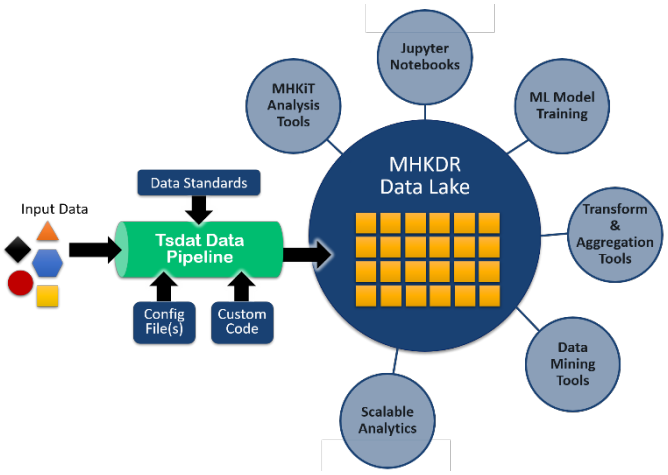

Fig. 1. Tsdat vision

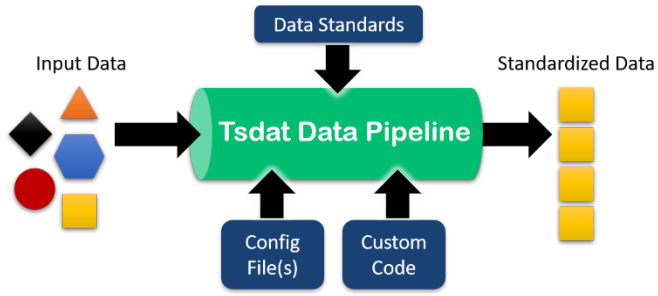

Fig. 2. Tsdat data pipeline overview

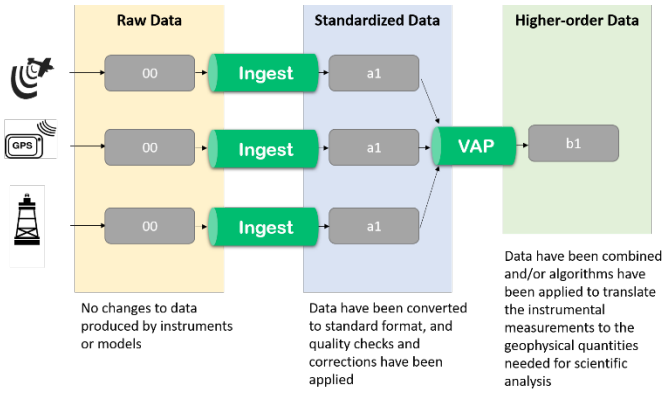

Fig. 3. Tsdat pipeline categories

3: ingestion pipelines and value-added product (VAP) pipelines. Ingestion pipelines use raw instrument data as input, apply quality checks and data cleaning algorithms, and convert the raw data into standard format. VAP pipelines combine multiple lower-level ingested datastreams and apply algorithms to translate the instrumental measurements into the geophysical quantities needed for scientific analysis. The initial version of Tsdat is focused on ingestion pipeline templates, but additional VAP pipeline templates will be provided within the next year.

This section explains the Tsdat architecture and usability features in more detail and demonstrates how Tsdat can be set up to feed data lakes residing in the cloud or on premises. More documentation, examples, and tutorials are available on our GitHub repository and our website [6,9].

A. Tsdat Architecture

Tsdat was developed following best practice data processing guidelines acquired by the climate community and more specifically the ARM program over the past 20 years. ARM data processing conventions were used as a baseline because they support time-series data, any dimensionality of data, produce high quality data which include detailed metadata and data quality information, and are general enough to apply to any domain. As Tsdat is intended for use by researchers as well as software developers, it was developed in Python and is based upon the popular Xarray library [10].

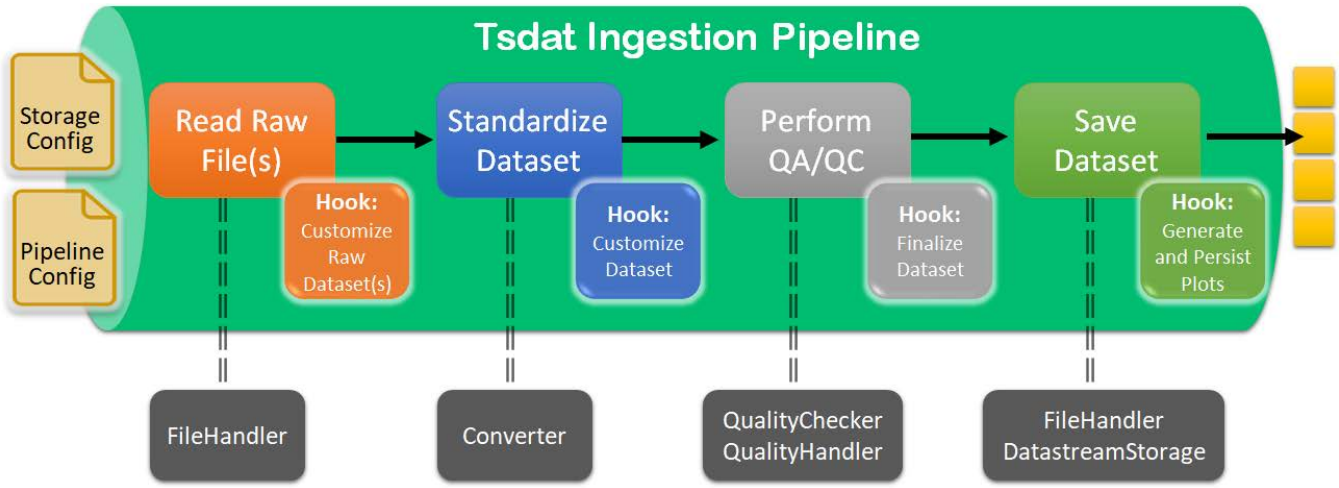

Fig. 4. Tsdat ingestion pipeline

. The pipeline's configuration files and code hooks are used to tailor the specific data and metadata that will be contained in the final, standardized dataset. Tsdat pipelines provide multiple layers of configuration to provide a low initial barrier of entry for basic ingests yet allow full customization of the pipeline for unique circumstances. Fig. 4 illustrates the different phases of the pipeline along with the multiple layers of configuration that Tsdat provides.

As shown in Fig. 4, the base Tsdat ingestion pipeline consists of an initial phase where raw files are read into an Xarray data structure. Then the Xarray variables and data are converted into standard names and units and global metadata are applied based upon the information specified in the pipeline config file. Next, the pipeline is run through a quality management step where any number of checks and cleaning algorithms are applied as specified by the pipeline configuration file. As part of the quality management phase, dataset provenance metadata including information about when the data were processed and the corrections applied to data variables, even at specific indices, are automatically recorded in the output dataset. Finally, the standardized, quality-controlled dataset is saved to the desired storage destination as also specified by the pipeline configuration. Fig. 4 also illustrates that code hooks are provided after each phase in the pipeline so that users can customize the exact data that are produced, such as deriving a new variable from other variables in the file. This baseline ingestion pipeline with code hooks should cover almost all processing scenarios, but users are also free to extend the base class to add additional phases if needed for unique circumstances.

Within the pipeline configuration files and code hooks, users can use pluggable helper classes (indicated by the dark gray boxes) to perform cross-pipeline functionality such as reading specific input file formats, performing quality assurance/quality control (QA/QC) checks, and storing output data. These helper classes can be contributed by the Tsdat library itself or by supplemental libraries that can be added for specific communities or use cases. This was done so that users can leverage Tsdat contributions across domains.

B. *Storage*

Tsdat is designed to be able to store processed data to any file format and location so that the pipelines can be tailored to work with any data repository and/or data lake infrastructure of choice. Tsdat currently supports netCDF and csv file formats, and support for the scalable data lake zarr and parquet file formats are being developed. Additionally, we plan to add support for databases to be used as input or output to Tsdat pipelines to accommodate a growing number of projects planning to use Tsdat as a streaming pipeline for standardizing record-based data. Users can easily extend Tsdat's default storage and I/O capabilities by defining their own Python classes. This plug-and-play modularity makes Tsdat data-format- and platform- agnostic.

C. *Pipeline Templates and Triggers*

To reduce the learning curve and accelerate pipeline setup, Tsdat provides repository templates that can be cloned by users to start with a working pipeline out of the box. Users can then customize the template as needed to suit their data processing requirements. We intend to provide multiple templates for the most common pipeline and storage configurations. Currently, we include two Ingestion Pipeline templates, one for running pipelines with a manual or cron-job based trigger on the local filesystem, and one for running pipelines with an S3 trigger on the Amazon Web Services (AWS). The AWS template includes a deployment script that sets up all cloud resources including the input and output S3 buckets and the serverless pipeline application. Longer term, we will work with the marine energy community to add the following usability enhancements: 1) a configurable command-line script that works with the templates to allow users to mix and match elements as needed, 2) templates for VAP pipelines, 3) more trigger options to support streaming data via message queues, and 4) templates for custom data models, as explained below in the Data Standards section.

III. Data Standards

Data standards ensure that all data produced by Tsdat pipelines are in a consistent format and can interoperate seamlessly. For Tsdat it was important to identify a core set of baseline standards that would work with any domain, but then allow each community to customize the core with specific data models applicable to their programs, data types, and research. This section describes the baseline data standards adopted by Tsdat, how they can be extended by each community via data models, and the strategy Tsdat has employed to engage the community in data standards development and curation.

A. Baseline Standards

We believe that Tsdat's baseline data standards should extend, not conflict, with well-established data standards and conventions for time-series data so it can leverage existing libraries and tools. With this focus on interoperability in mind, we chose to base our data standards on the climate and forecast (CF) conventions widely used by atmospheric science communities [12]. CF conventions are designed in conjunction with the netCDF data format widely used in atmospheric and meteorological science communities and there exists a large and thriving ecosystem of tools developed in compliance with these standards [13]. Additionally, the CF conventions have several guiding principles that we like and have adopted in our project: 1) data should be self-describing, 2) metadata should be user-friendly and machine-readable, and 3) data standards should be as simple as possible and not try to regulate non-existent use-cases [14]. In addition to CF conventions, we also opted to include portions of the data standards used by the Atmospheric Radiation Measurement (ARM) program relating to data quality checks and controls and file-based data organization [15]. By building on CF conventions and ARM data standards, we are inherently including support for an entire suite of open-source time-series data tools and software widely used by these communities, including Python libraries such as Xarray, MetPy, Cartopy, ACT-ATMOS, Py-ART, MHKiT-Python, and more [10,16,17,18,19,20].

More information on Tsdat's baseline data standards can be found at [21], which provides guidance on several topics regarding data provenance metadata, standard names for variables, organization and stewardship of file-based data, and variables and metadata regarding quality checks and controls.

B. Data Models

In addition to general baseline data standards that apply to any data, specific metadata, data quality, or analysis requirements may apply, depending upon the type of dataset being produced. These additional requirements are referred to as data models, because they are specific to a particular type of data (e.g., wave energy converter field tests). As shown in Fig. 5, data models extend the baseline data standards with additional metadata, variables, computations, and data quality checks. Each domain is responsible for defining any applicable data models. For marine energy, Tsdat will work with the MHKDR repository team and key project stakeholders to identify and develop a core set of data models and their corresponding pipeline templates. In developing the templates, we anticipate making heavy use of the I/O and data processing modules from the Marine Hydrokinetic Toolkit (MHKiT) [20].

C. Community Engagement

We believe that community engagement is a critical part of the development process for standards and tools that will be broadly used. By seeking community input we can expand our perspective and develop solutions that are more widely applicable and more likely to be adopted by the community. For this reason, we have uploaded our baseline data standards to a public GitHub repository where users can contribute to the standards document by raising concerns, requesting changes, or engaging in discussion with other community members [21]. Domain-specific data model specifications may also be contributed to this repository, provided as separate documents. We will work with the MHKDR team to identify and engage key project stakeholders who can help to review and curate marine energy data model submissions.

IV. Example Application Developed Using Tsdat

The Atmosphere to Electrons program is a multiyear research and development initiative funded by the U.S. Department of Energy tasked with improving wind energy performance and reducing the cost of wind energy production [22]. In 2020, two buoys hosting a suite of surface and ocean meteorology instruments were deployed off the coast of Morro Bay, California, and Humboldt, California, to evaluate the sites for compatibility with commercial offshore wind development [23].

Instruments on these buoys each produce a single daily CSV file, shown in Table 1, each of which consists of a series of timestamped measurements. Table 2 shows a sample of acoustic doppler current profiler (ADCP) data as recorded in the buoy.z06.00.20201201.000000.currents.csv file at Morro Bay. Note that many columns are needed to report current velocity measurements taken in each of the 50 bins.

The Atmosphere to Electrons program used Tsdat to develop data ingestion pipelines to standardize and consolidate these data into a format more suitable for higher-level analysis. Specifically, Tsdat ingestion pipelines were created to consolidate and standardize data for the raw instrument files shown in Table 1. Configuration files were developed in collaboration with an instrument specialist to embed human-readable metadata with the output dataset and to define data limits and quality checks to be performed by Tsdat as part of the data standardization processing. Aside from plotting code and 48 lines of Python written to consolidate the ADCP data into two two-dimensional variables, no additional user-written Python code was needed to read-in the data, standardize it, embed user-friendly metadata, provide quality check reports on the data, and store the data and plots in an AWS S3 bucket. Fig. 6 and Fig. 7 show a snippet of the data output by this pipeline and a sample plot of the ocean currents data respectively.

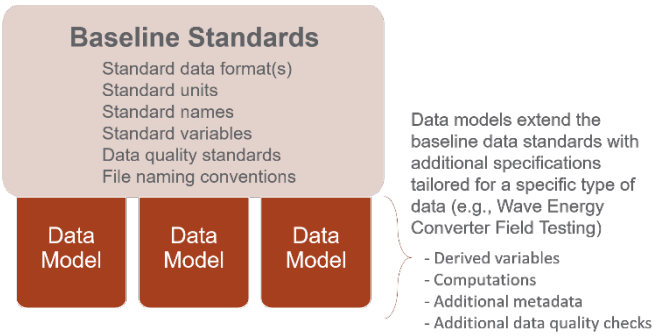

Fig. 5. Tsdat baseline standards and data models

TABLE I. FILES PRODUCED BY THE BUOY DEPLOYED AT MORRO BAY ON DECEMBER 1, 2020.

Filename
buoy.z06.00.20201201.000000.conductivity.csv
buoy.z06.00.20201201.000000.currents.csv
buoy.z06.00.20201201.000000.gill.csv
buoy.z06.00.20201201.000000.gps.csv
buoy.z06.00.20201201.000000.pressure.csv
buoy.z06.00.20201201.000000.pyranometer.csv
buoy.z06.00.20201201.000000.rh.csv
buoy.z06.00.20201201.000000.surfacetemp.csv
buoy.z06.00.20201201.000000.temperature.csv
buoy.z06.00.20201201.000000.wind.csv

TABLE II. SAMPLE OF RAW OCEAN CURRENTS DATA AT MORRO BAY ON DECEMBER 1, 2020.

DataTimeStamp	NumberOfBins	BinSpacing	Vel1 (mm/s)	Dir1 (deg)	Vel2 (mm/s)	Dir2 (deg)	...	Vel50 (mm/s)	Dir50 (deg)
2020-12-01 00:00:00	50	4	240	260.8	230	261.3	...	46340	225
2020-12-01 00:10:00	50	4	230	272	230	258.7	...	46340	225
2020-12-01 00:20:00	50	4	200	259.3	210	255.9	...	46340	225
...	...	...	...	...	...	...	...	...	...
2020-12-01 23:50:00	50	4	360	234.6	320	233.7	...	46340	225

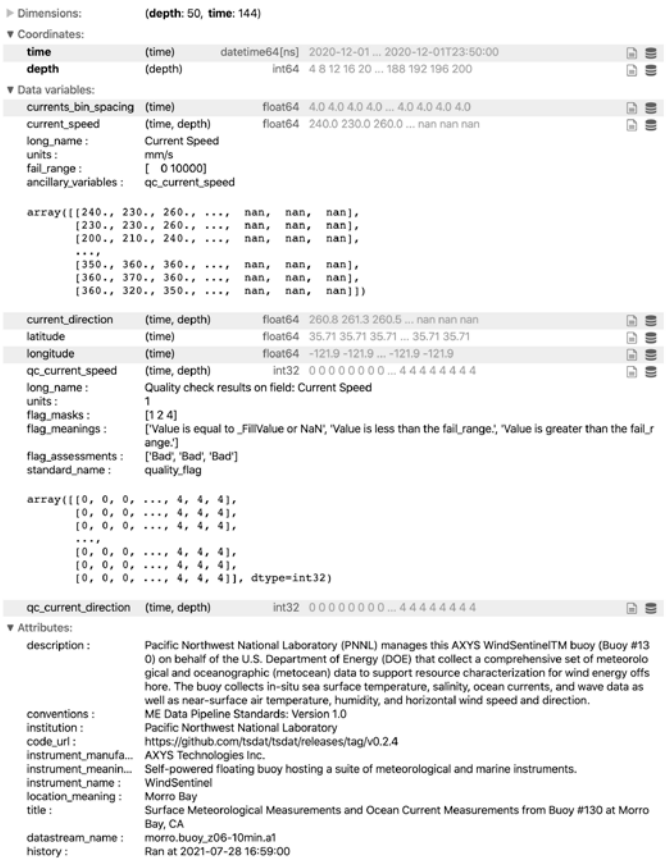

Fig. 6. Snippet of data output by a Tsdat pipeline

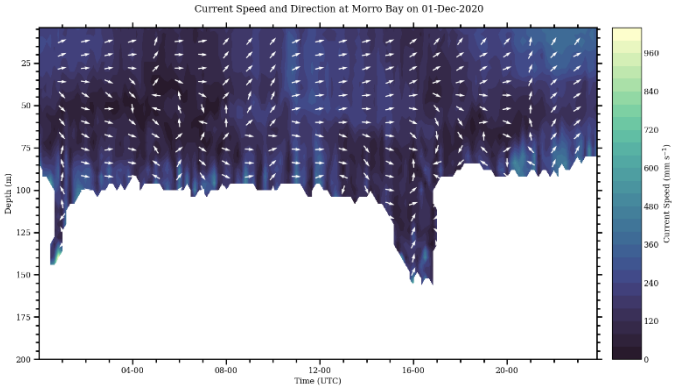

Fig. 7. Plot of ocean current velocities produced by a Tsdat pipeline

V. CONCLUSION

Open-source tools and data standards will serve the marine energy community by providing shared solutions to common data needs, greatly reducing duplication of effort across domains, and reducing the time to commercialization. We developed Tsdat to reduce the operational barriers preventing organizations from maintaining high-quality standardized datasets that can be easily used in downstream analyses and reports. Tsdat can be used to read and standardize data from data acquisition systems deployed in marine energy environments, instruments deployed in wind energy, and sensors from the buildings domain. By using Tsdat to read, curate and standardize data, the time to process and analyze data will be reduced dramatically, thereby increasing success rates for projects. By reusing code, higher-quality data products will be created during field testing, which will provide timely feedback to project members to improve collection or configuration of instruments and detect issues with instruments.

Our next step is to host the ingested standardized data in data lakes to allow organizations to collaborate, filter, and query the data easily; generate insights on historical or streaming data; and apply machine learning algorithms to predict quality to achieve the optimal result.

As Tsdat gains traction, we continue to seek opportunities for community engagement—whether that be by users asking questions and providing feedback, or, ideally, by users contributing new features to Tsdat so we can support a broader range of applications. We invite readers interested in using Tsdat for their own projects to visit our GitHub at https://github.com/tsdat.

ACKNOWLEDGMENT

This material is based upon work conducted in support of the Water Power Technologies Office within the U.S. Department of Energy's Office of Energy Efficiency and Renewable Energy. The use case to test the ingestion pipeline was conducted in support of the Wind Energy Technologies Office within the U.S. Department of Energy's Office of Energy Efficiency and Renewable Energy.

We would like to thank Raghavendra Krishnamurthy, Matthew MacDuff, and the rest of the Atmosphere to Electrons program sponsored by the U.S. Department of Energy for their adoption of Tsdat early in our development process and for their continued support and use of Tsdat. We also extend a special thank you to Calum Kenny, whose questions, feedback, and strategic vision have been highly beneficial to this project. Finally, we would like to thank Eddie Schuman, Yangchao (Nino) Lin, Jonathan Whiting, Emma Cotter, and Daniel Zalkind for providing feedback and interesting use cases for Tsdat. Their input has helped shape various features and processes within Tsdat.

REFERENCES

[1] "Clean, complete, uncompromised data for everyone." Talend. https://www.talend.com/ (retrieved July 26, 2021).

[2] "Modern BI for AllTM." Domo. https://www.domo.com/ (retrieved July 26, 2021).

[3] "Become a data-driven organization with Tableau." Tableau. https://www.tableau.com/ (retrieved July 26, 2021).

[4] K. Gaustad, T. Shippert, B. Ermold, S. Beus, J. Daily, A. Borsholm, K. Fox, "A scientific data processing framework for time series NetCDF data," *Environmental Modelling & Software*, 60, pp. 241-249, Oct. 2014, https://doi.org/10.1016/j.envsoft.2014.06.005

[5] "ARM." U.S. Department of Energy, Office of Science. https://www.arm.gov/ (retrieved July 26, 2021)

[6] "Tsdat Time Series Data Pipelines." GitHub. https://github.com/tsdat (retrieved July 26, 2021).

[7] "Marine and Hydrokinetic Data Repository, U.S. Department of Energy." Open EI, Marine and Hydrokinetic Data Repository. https://mhkdr.openei.org/ (retrieved July 26, 2021).

[8] H. Fang. (June 2015) "Managing data lakes in big data era: What's a data lake and why has it became popular in data management ecosystem." Presented at 5th Annual IEEE International Conference on Cyber Technology in Automation, Control and Intelligent Systems, Shenyang, China. [Online] 10.1109/CYBER.2015.7288049

[9] "Tsdat." Tsdat. https://tsdat.readthedocs.io/en/latest/ (retrieved July 26, 2021).

[10] "xarray: N-D labeled arrays and datasets in Python." xarray. http://xarray.pydata.org/en/stable/ (retrieved July 26, 2021).

[11] "YAML: YAML Ain't Markup Language." %YAML 1.2. https://yaml.org/ (retrieved July 26, 2021).

[12] "CF Conventions and Metadata." CF MetaData. https://cfconventions.org/ (retrieved July 26, 2021).

[13] "Network Common Data Form (NetCDF)." UCAR Community Programs, UNIDATA, Data Services and Tools for Geoscience. https://www.unidata.ucar.edu/software/netcdf/ (retrieved July 26, 2021).

[14] J. Gregory, "The CF metadata standard," University of Reading, UK and Met Office Hadley Centre, Exeter, UK, November 6, 2003. [Online]. Available:http://cfconventions.org/Data/cf-documents/overview/article.pdf

[15] ARM Standards Committee, "ARM Data File Standards Version: 1.3," U.S. Department of Energy, Office of Science, DOE/SC-ARM-15-004, September 2020. [Online]. Available: https://www.arm.gov/publications/programdocs/doe-sc-arm-15-004.pdf

[16] May, R. M., Arms, S. C., Marsh, P., Bruning, E., Leeman, J. R., Goebbert, K., Thielen, J. E., and Bruick, Z., 2021: MetPy: A Python Package for Meteorological Data. Unidata, https://github.com/Unidata/MetPy, doi:10.5065/D6WW7G29.

[17] Met Office, "Cartopy: a cartographic python library with a matplotlib interface,"

[18] "ARM-DOE/ACT." Version 1.0.4. GitHub. https://github.com/ARM-DOE/ACT (retrieved July 26, 2021).

[19] J. Helmus, S. Collis, "The Python ARM radar toolkit (Py-ART), a library for working with weather radar data in the Python programming language," *Journal of Open Research Software*, 4, 1, p.e25, http://doi.org/10.5334/jors.119

[20] K. Klise et. al., "MHKiT (marine hydrokinetic toolkit) – Python," *Computer Software*, January 2020, https://doi.org/10.5281/zenodo.3924683.

[21] "ME Data-Pipeline-Software/data_standards." GitHub. https://github.com/ME-Data-Pipeline-Software/data_standards (retrieved July 26, 2021).

[22] Atmosphere to Electrons, https://a2e.energy.gov/ (accessed July 26, 2021).

[23] "BUOY, Offshore Wind Energy - Buoy Lidar Project." Atmosphere to Electrons, U.S. Department of Energy. https://a2e.energy.gov/projects/buoy (retrieved July 26, 2021).