

















**SciPy 2024**

July 8 - July 14, 2024

Proceedings of the 23<sup>rd</sup>  
Python in Science Conference  
ISSN: 2575-9752

# Echostack: A flexible and scalable open-source software suite for echosounder data processing

Wu-Jung Lee<sup>1</sup>  , Valentina Staneva<sup>2</sup>  , Landung “Don” Setiawan<sup>2</sup>  ,  
Emilio Mayorga<sup>1</sup>  , Caesar Tuguinay<sup>1</sup>  , Soham Butala<sup>2</sup>  , Brandyn  
Lucca<sup>1</sup>  , and Dingrui Lei<sup>2</sup>  

<sup>1</sup>Applied Physics Laboratory, University of Washington, <sup>2</sup>eScience Institute, University of Washington

## Abstract

Water column sonar data collected by echosounders are essential for fisheries and marine ecosystem research, enabling the detection, classification, and quantification of fish and zooplankton from many different ocean observing platforms. However, the broad usage of these data has been hindered by the lack of modular software tools that allow flexible composition of data processing workflows that incorporate powerful analytical tools in the scientific Python ecosystem. We address this gap by developing Echostack, a suite of open-source Python software packages that leverage existing distributed computing and cloud-interfacing libraries to support intuitive and scalable data access, processing, and interpretation. These tools can be used individually or orchestrated together, which we demonstrate in example use cases for a fisheries acoustic-trawl survey.

**Keywords** ocean sonar, echosounder, distributed computing, cloud computing, workflow orchestration, data standardization

## 1. INTRODUCTION

Echosounders are high-frequency sonar systems optimized for sensing fish and zooplankton in the ocean. By transmitting sound and analyzing the returning echoes, fisheries and ocean scientists use echosounders to “image” the distribution and infer the abundance of these animals in the water column [1], [2], [3] [Figure 1](#). As a remote sensing tool, echosounders are uniquely suitable for efficient, continuous biological monitoring across time and space, especially when compared to net trawls that are labor-intensive and discrete in nature, or optical imaging techniques that are limited in range due to the strong absorption of light in water.

In recent years, echosounders have been installed widely on many ocean observing platforms [Figure 1](#), resulting in a deluge of data accumulating at an unprecedented speed from all corners of the ocean. These extensive datasets contain crucial information that can help scientists better understand the marine ecosystems and their response to the changing climate. However, the volume of the data (100s of GBs to TBs [4]) and the complexity of the problem (e.g., how large-scale ocean processes drive changes in acoustically observed marine biota [5]) naturally call for a paradigm shift in the data analysis workflow.

It is crucial to have software tools that are developed and shared openly, scalable in response to data size and computing platforms, easily interoperable with diverse analysis tools and different types of oceanographic data, and straightforward to reproduce to facilitate iterative modeling, parameterization, and mining of the data. These requirements

**Published** Jul 10, 2024

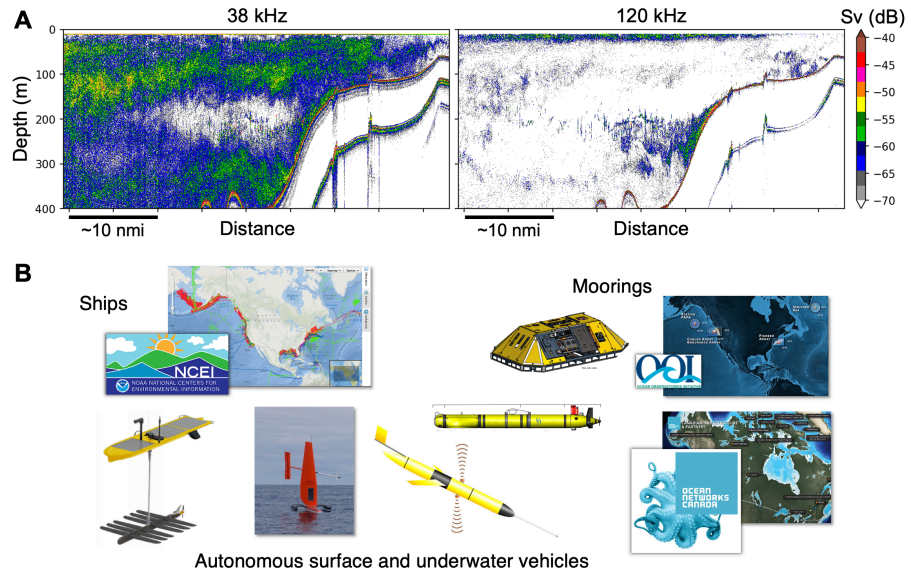
### Correspondence to

Wu-Jung Lee  
[leewj@uw.edu](mailto:leewj@uw.edu)

### Open Access



Copyright © 2024 Lee *et al.*. This is an open-access article distributed under the terms of the [Creative Commons Attribution 4.0 International](#) license, which enables reusers to distribute, remix, adapt, and build upon the material in any medium or format, so long as attribution is given to the creator.



**Figure 1.** (A) Echograms (sonar images formed by aligning echoes of consecutive transmissions) at two different frequencies. Echo strength variation across frequency is useful for inferring scatterer identity.  $S_v$  denotes volume backscattering strength (units:  $\text{dB re } 1 \text{ m}^{-1}$ ). (B) The variety of ocean observing platforms with echosounders installed.

are challenging to meet by conventional echosounder data analysis workflows, which rely heavily on manual analysis on software packages designed to be used with Graphic User Interface (GUI) on a single computer [e.g., R. Korneliussen *et al.* [6], Y. Perrot *et al.* [7], Y. Ladroit, P. C. Escobar-Flores, A. C. G. Schimel, and R. L. O’Driscoll [8] Echoview (<https://echoview.com/>)]. Similarly, instead of storing data in manufacturer-specific binary formats, making echosounder data widely available in a standardized, machine-readable format will broaden their use beyond the original applications in fisheries surveys and specific research cruises.

In this paper, we introduce Echostack, an open-source Python software suite aimed at addressing these needs by providing the fisheries acoustics and ocean sciences communities with tools for flexible and scalable access, organization, processing, and visualization of echosounder data. Echostack is a domain-specific adoption of the core libraries in the Pandata stack, such as Zarr, Xarray, and Dask [9]. It streamlines the composition and execution of common echosounder data workflows, thereby allowing researchers to focus on the key interpretive stage of scientific data analysis. As echosounder data processing frequently requires combining heterogeneous data sources and models in addition to regular array operations, we designed the Echostack packages with modularity in mind, to 1) enable and facilitate broader code reuse, especially for routine processing steps that are often shared across echosounder data workflows [10], and 2) promote flexible integration of domain-specific operations with more general analytical tools, such as machine learning (ML) libraries. The Echostack also provides a friendly on-ramp for researchers who are not already familiar with the scientific Python software ecosystem but possess domain expertise and can quickly benefit from “out-of-the-box” capabilities such as native cloud access and distributed computing support.

Below, we will discuss what the Echostack tools aim to achieve in Section 2, outline the functionalities of individual libraries in Section 3, demonstrate how Echostack tools can be leveraged in Section 4, and conclude with a forward-looking discussion in the Section 5 section.

## 2. DESIGN APPROACH AND CONSIDERATIONS

### 2.1. Goals and motivations

The development of the Echostack was motivated by our core goals to create software tools that can be:

1. Easily leveraged to construct workflows tailored to different application scenarios,
2. Easily integrated with new data analysis methodologies, and
3. Easily scalable according to the available computing resources and platforms.

Traditionally, the bulk of echosounder data analysis takes place on shore in a post-processing scenario, based on the acoustic data and related biological information (e.g., animal community composition and sizes derived from net catches) collected at sea and stored on hard drives. More recently, thanks to increased access to high-bandwidth communication and powerful embedded processing, scenarios requiring real-time capabilities have emerged, including continuous streaming of data from long-term moorings [11] and close-loop, adaptive sampling on autonomous vehicles [12]. Similarly on the rise are the incorporation of new inference and ML methods into echosounder data analysis workflows [13], [14], [15], [16], as well as a need for scalable and platform-agnostic computations that work out-of-the-box on systems ranging from personal computers to the cloud. The latter is particularly important for data workflows associated with ocean instruments such as echosounders, as both small and big data scenarios are common and equally important: On a sea-going research vessel, newly collected data are typically processed in small batches on a few local machines; in oceanographic data centers, however, large volumes of data need to be promptly ingested and made available for open use.

These requirements are challenging to fulfill by existing echosounder data processing software packages, but are in fact key advantages of the Pandata open-source Python stack. While some existing software packages do support real-time applications (e.g., Echoview live viewing, MOVIES3D real-time display) and allow the incorporation of new analysis methods [17], the exact code implementation may be limited by the data structures and access allowed in a given package, as opposed to the flexibility afforded by Pythonic programmatic composition. In terms of scalability, even though some existing packages can achieve out-of-core computation via memory mapping [6], [8], the natural and optional scalability across computing resources and platforms offered by Pandata libraries [9] significantly reduces the code implementation overhead.

### 2.2. Other guiding principles

In addition to the above goals, our development was further guided by the following principles, with the long-term goal of catalyzing community collaboration and workflow transformation.

#### 2.2.1. Accessible and interoperable data

Echosounder data have traditionally been viewed as a highly specialized, niche data type that are hard to use and interpret despite the wealth of biological information embedded in the datasets. This is in part due to the highly heterogeneous, instrument-specific echosounder data format, as well as the inherent complexity in interpreting acoustic backscatter data [2]. We aim to provide convenient, open software tools that facilitate the creation of standardized data in a common, machine-readable format that is both easy to understand and conducive to integrative analysis with other oceanographic datasets.

### 2.2.2. Reproducible and shareable workflow

Just like in many scientific domains, echosounder data analysis workflows are typically built by researchers working closely within a single group, although knowledge sharing and exchanges do occur in community conferences or workshops. As data volumes rapidly increase and the urgency to comprehensively understand marine ecosystem functions escalates with global warming, efficient and effective collaboration across groups has become crucial. Ensuring that analysis workflows can be easily reproduced and shared alongside open, standardized data is key to scaling up our collective capability to extract information from these data.

### 2.2.3. Nimble code adjustments

As a domain-specific adaptation of the Pandata stack, we aim to develop and maintain a “shallow” and nimble stack: we employ only moderate layers of abstraction and encapsulation of existing functions and prioritize existing data structures unless modification is absolutely necessary. This allows new features, enhancements, or bug fixes in the underlying dependency libraries to be either naturally reflected (in cases when there are no breaking changes) or quickly incorporated (in cases when code changes are required) into the Echostack tools. This approach also keeps the code easily readable for potential contributors, even though the exact operations may be highly domain-specific.

## 3. THE ECHOSTACK PACKAGES

### 3.1. Functional grouping for package organization

In light of our goals and design principles, we aim for the Echostack libraries to be modular software “building blocks” that can be mixed and matched, not only with other elements within Echostack but also with tools from the wider scientific Python software ecosystem. Rather than encapsulating all functionalities into a single, monolithic package like many existing echosounder data analysis software [e.g., R. Korneliussen *et al.* [6] Echoview], the Echostack packages are organized based on functional groupings that emerge in typical echosounder data analysis workflows to enable flexible and selective installation and usage. Similar to the design concept of the Pandata tools, such grouping also allows each package in the Echostack to be independently developed and maintained according to natural variations of project focus and needs.

Typical echosounder data analysis workflows consist of the following common steps [Figure 2](#):

1. Read, align, and aggregate the acoustic data and associated biological data (the “ground truth” from *in situ* samples) based on temporal and/or spatial relationships;
2. Infer the contribution of animals in different taxonomic groups to the total returned echo energy; and
3. Derive distributions of population estimates (e.g., abundance or biomass) and behavior (e.g., migration) over space and time for the observed animals.

The resulting biological estimates are then used in downstream fisheries and marine ecological research and for informing resource management decisions. Also embedded in these steps are the needs for:

- Interactive visualization of echo data and/or derived estimates, at different stages of the workflow, which is crucial for understanding data and motivating downstream analyses; and
- Robust provenance tracking of data origin and processing steps, which enhances the reproducibility and shareability of the workflows.

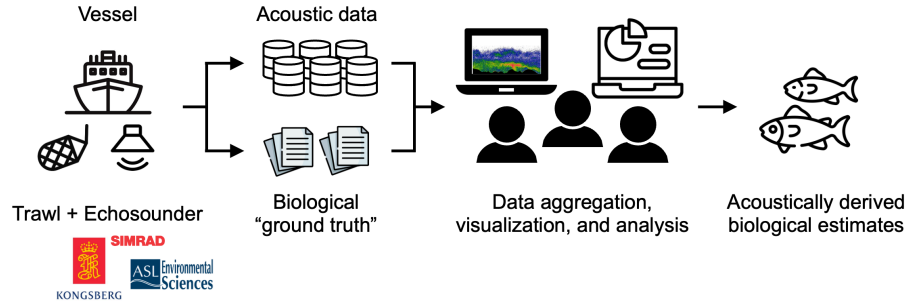


Figure 2. Typical echosounder data analysis workflows.

In a fisheries acoustic-trawl survey – a typical use case – the overall goal is to estimate the abundance and biomass of important fish and zooplankton species using a combination of observational data and analytical models. The observational data include acoustic data collected by the echosounders and biological data (animal species and sizes, etc) collected by trawls. These data are linked in the acoustic inference step, which partitions the acoustic observation into contributions from different animal groups in the water column based on empirical models (which rely heavily on morphological features of animal aggregations visualized on the echogram), physics-based models (which rely heavily on spectral or statistical features of the echo returns), ML models (which can combine various data features depending on the model setup), or a mixture of the above.

Based on the above, we identified natural punctuations in the typical workflows, and grouped computational operations that serve coherent high-level functionalities into separate packages Figure 3 that are described in the next section. The grouping decisions and the current implementations were driven by our own project goals to accelerate information extraction for a fisheries acoustic-trawl survey for Pacific hake (see Section 4 for detail) through ML and cloud technology. However, as the examples will show, these packages can be easily leveraged in different application scenarios.

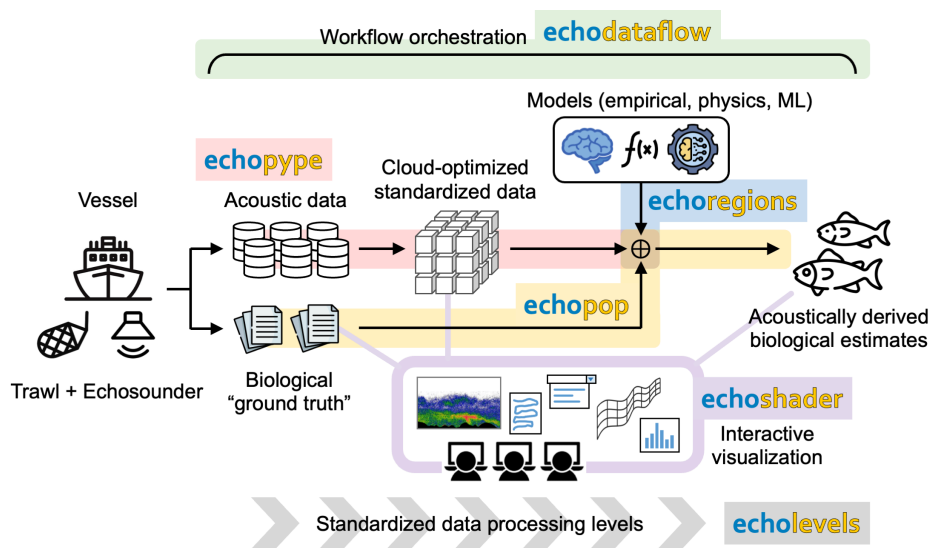


Figure 3. The Echostack packages are modularized based on functional grouping of computational operations in echosounder data analysis workflows.

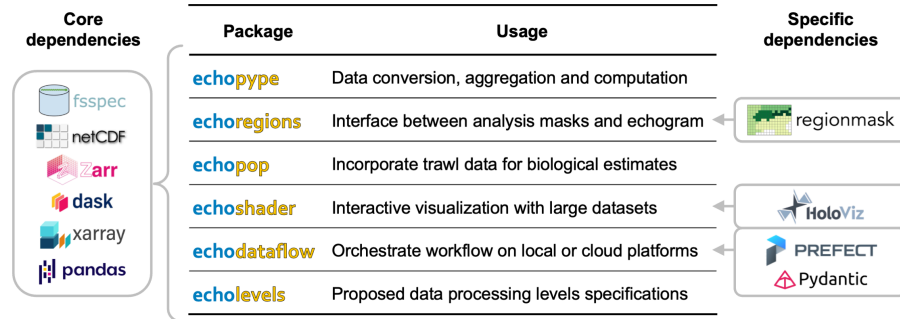


Figure 4. Summary of the Echostack packages and key dependencies.

### 3.2. Package functionalities

The Echostack includes five software packages (Echopype, Echoregions, Echopop, Echoshader, and Echodataflow) that each handles a specific set of functionalities in general echosounder data processing workflows Figure 3. In parallel, Echolevels provides specifications for a cascade of data processing levels from raw to highly processed data, to enhance data sharing and usage. The Echostack packages jointly relies on a set of core dependencies for interfacing with file systems (fsspec), data storage (netCDF and Zarr), and data organization and computation (Xarray, Pandas, and Dask). Additionally, some libraries have specific dependencies to support functionalities like interactive plotting (HoloViz), masking grids based on regions (Regionmask), and workflow configuration (Pydantic) and orchestration (Prefect), as shown in Figure 4.

The sections below provide brief descriptions of the functionalities of the Echostack packages, their current status, and future development plans.

#### 3.2.1. Echopype

Echopype (<https://github.com/OSOceanAcoustics/echopype>) is the workhorse that performs the heavy lifting of data standardization and routine computation to generate data products that are suitable for downstream analysis. Acoustic (backscatter) data collected by echosounders are typically stored in instrument-specific binary files along with metadata, such as deployment configurations. The raw data are often recorded in sensor units (e.g., counts) that require calibration to be mapped to physically meaningful quantities (e.g., sound pressure). Analyses are typically performed on aggregated data sections based on spatial and/or temporal logical groupings. For example, ship echosounder data are often grouped based on survey transects, and mooring echosounder data are often grouped into weekly or monthly sections. In addition, many analysis methods require data to be uniformly gridded across all major dimensions, such as time, geographic locations (latitude/longitude), and depth. Echopype provides functionalities for converting raw, binary data files into a standardized representation following the netCDF data model, which can be serialized into netCDF or Zarr files. It also provides functionalities for calibration, quality control, aggregating data across files, and data regridding. It forms the data churning backbone in Echostack-supported workflows, and encodes provenance and processing level information as specified in Echolevels (see below). Echopype was the first package developed in the Echostack and is described in detail in W.-J. Lee, E. Mayorga, L. Setiawan, I. Majeed, K. Nguyen, and V. Staneva [18].

#### 3.2.2. Echoregions

Echoregions (<https://github.com/OSOceanAcoustics/echoregions>) is a lightweight package that interfaces data products generated by Echopype, such as echograms, with annotations

from humans or data interpretation algorithms, such as ML algorithms. These annotations typically consist of “regions” that indicate the presence of specific animal species, or “lines” that delineate ocean boundaries, such as seafloor or sea surface. As a lightweight interfacing package, Echoregions makes it possible to construct independent workflow branches that handle data interpretation annotations in concert with the data processing backbone and the analysis algorithms. Currently, due to our project priorities, Echoregions primarily supports the organization and generation of echogram analysis “masks” in ML problems [19] with bi-directional interfacing functionalities: to parse and organize annotations to prepare training and test datasets for ML developments (annotation to ML), and to generate annotations from ML predictions to be used for further downstream processing (ML to annotation). The current supported annotation formats are those generated by Echoview (.EVR and .EVL files), which is widely used in the fisheries acoustics community to manually analyze echograms. We plan to expand support for additional annotation formats and update both the representation of annotations and the generated masks to align with community conventions that may emerge in the near future.

### 3.2.3. Echopop

Echopop (<https://github.com/OSOceanAcoustics/echopop>) functions as the “glue” that combines echo measurements with biological data from trawls and acoustic backscattering models to estimate spatial distributions of animal abundance, biomass, and other biologically meaningful metrics. Here, “pop” stands for animal “population” information that can be obtained by analyzing echo data. Biological trawl samples typically yield species compositions and associated biometrics, such as sex, length, age, and weight. Many of these parameters are crucial for accurately configuring the acoustic backscattering models, which strongly affect the estimates of biological quantities from echo data [2], which are important for informing fishery management strategies and for characterizing regional and global energy budgets. At present, the biological data agglomeration and echo data analysis steps implemented in Echopop are configured to work specifically with a fisheries acoustic-trawl survey targeting Pacific hake that is the focus of our project. However, the majority of the functions are generally applicable to other fish and zooplankton species. We therefore plan to expand the package to incorporate more general analysis functions in the near future, including support for biological data obtained by other *in-situ* ground-truthing methods, such as optical cameras [20].

### 3.2.4. Echoshader

Echoshader (<https://github.com/OSOceanAcoustics/echoshader/>) enables interactive visualization of echosounder data to accelerate the data exploration and discovery process. Here, “shader” was inspired by Datashader, as echo data are often plotted as false color panels encoded with magnitude information (e.g., Figure 1). Building on the HoloViz suite of tools, Echoshader extends Xarray’s plotting functionality using accessors (<https://docs.xarray.dev/en/latest/internals/extending-xarray.html>) to include variations that are widely used and insightful for exploring and understanding echosounder data. For example, the absolute and relative echo magnitudes measured at different frequencies can be used to classify scatterers, and are typically visualized through joint displays of echograms at multiple frequencies. Echograms are water column “profiles” that are often more meaningful when inspected together with the ship track on a bathymetry map. Echoshader makes it possible to create these plots directly from echosounder datasets generated by Echopype (which are native Xarray datasets or data arrays), complete with control widgets such as frequency selectors and color range sliders. In the next stage of development, we plan to integrate external echogram region annotations (e.g., from Echoregions) into the functionality, optimize performance for visualizing large datasets to enable efficient screening of historical data, and compile a gallery of demo use cases as part of the documentation.

### 3.2.5. Echodataflow

Echodataflow (<https://github.com/OSOceanAcoustics/echodataflow>) is a package that provides users with an efficient way to define, configure, and execute complex echosounder data processing workflows on either local or cloud platforms. It combines the power of Prefect (<https://www.prefect.io/>), an open-source workflow orchestration tool, and the flexibility of YAML configurations to leverage Echostack data processing functionalities in a modular and user-friendly way. Instead of writing code, users prepare “recipes” that are text-based configurations specifying data processing steps and associated parameters, data sources and output storage locations, logging options, and computing resources. Given a recipe, Echodataflow automatically translates these specifications into actions and augments the processes with the robust error tracking and retry mechanisms from Prefect. This makes it possible to systematically prototype and experiment with workflow variations and efficiently transition a prototype to a stable data pipeline, improving the collaboration between researchers, data engineers, and data managers. Currently, due to our project priorities, Echodataflow supports Echopype and Echoregions functionalities as well as custom functions that perform ML predictions and cloud data upload. We plan to create functional templates for users to easily “wrap” their own workflow elements to use with Echodataflow in the near future.

### 3.2.6. Echolevels

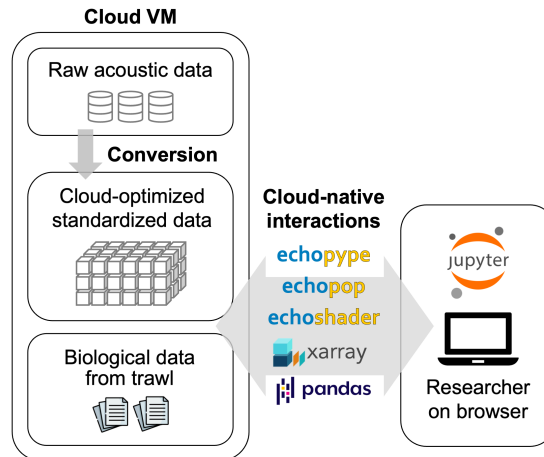
Echolevels (<https://github.com/OSOceanAcoustics/echolevels>) is a set of proposed data processing level definitions for echosounder data, rather than a software package. A concept originated and championed by the satellite remote sensing community [21], “data processing levels” are categorizations of the continuum of data at different processing stages, from raw instrument-generated files to derived, often gridded and georeferenced data products that are ready for analysis or interpretation. Similar to data conventions, clear processing level definitions enhance data understanding and provenance tracking, which contribute directly to broader data use. We have added the proposed processing levels as dataset attributes in the output data of different Echopype functions to test these definitions in practice, and are collaborating with the echosounder user community to revise and improve them.

## 4. EXAMPLE USE CASES

In this section, we present three example use cases of the Echostack libraries under the application scenario of a fisheries acoustic-trawl survey. Here, we use the Joint U.S. and Canada Pacific Hake Integrated Ecosystem and Pacific Hake Acoustic Trawl Survey (aka the “Hake survey”) as an example, as we are familiar with its context and setup for our project to accelerate biological information extraction using ML and cloud technology. The Hake survey dates back to the late 1990s when echosounder technology was first widely adopted for fishery surveys. In recent years, this survey has occurred every odd-numbered year, covering the coastal waters, shelf break, and open ocean along the entire west coasts of the US and Canada [22].

The example use cases presented below demonstrate how Echostack tools are leveraged in echosounder data analysis and management. The examples utilize both raw data files archived in a data center and near real-time data continuously generated from a ship-mounted echosounder. We show how these tools can be used in a mix-and-match manner with each other and with other tools in the scientific Python software ecosystem to achieve user goals. Note that not all Echostack libraries are used in all use cases, which highlights the advantage of modular package design based on functional grouping.





**Figure 5.** Use case 1: Interactive data exploration and experimentation.

#### 4.1. Use case 1: Interactive data exploration and experimentation

This is a common use case in which researchers interact with echosounder data through the browser-based Jupyter computing interface on a cloud virtual machine [Figure 5](#). They first assemble a Zarr store consisting of calibrated echo data for a single survey transect by directly processing a collection of raw files from a data center server (Echopype). They then interactively visualize the data (Echoshader) and experiment with the influence of different data cleaning options (Echopype and custom functions) on the resulting acoustically inferred biomass of a fish species (Echopop). These experimentations help the research identify parameters for setting up batch processing across all survey transects.

#### 4.2. Use case 2: A ship-to-cloud ML pipeline

This is an experimental use case in which researchers construct a ship-to-cloud pipeline that uses an ML model to analyze ship echosounder data in near real-time and send the predictions and reduced data to the cloud for shore-side monitoring [Figure 6](#). The pipeline is orchestrated by Echodataflow, which manages data organization actions triggered by new files and analysis-oriented actions through a scheduler. A “watchdog” monitors the ship hard drive for newly generated raw echosounder data files, and continuously converts, calibrates, regrid, and appends new echo data into a local Zarr store dedicated to ML analysis (Echopype). At regular intervals, the last section in the Zarr store is sliced and fed to an ML model to detect the occurrence of a target fish species on the echogram (custom functions). The echo data are applied with the ML predictions (Echoregions) and used to compute the abundance and biomass estimates (Echopop). The echo data slices are also further downsampled (Echopype) and sent to a cloud Zarr store for interactive visual monitoring (Echoshader). This pipeline paves the way for later optimization for an edge implementation on autonomous platforms.

#### 4.3. Use case 3: Mass production of analysis-ready, cloud-optimized (ARCO) data products

This is a use case in which researchers or data managers create analysis-ready, cloud-optimized (ARCO) echosounder data stores that can be readily drawn for analysis and experimentation [Figure 7](#). The pipeline is orchestrated by Echodataflow, which handles the efficient distribution of bulk computing tasks, communicates with data sources and destinations, and automatically logs and retries any processing errors. In the pipeline, instrument-generated binary data files sourced from a data center progress through multiple

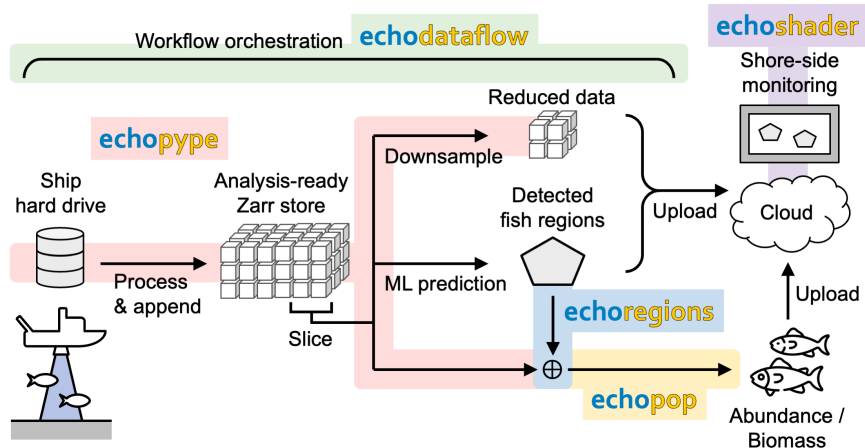


Figure 6. Use case 2: A ship-to-cloud ML pipeline.

processing steps and are enriched with auxiliary data and metadata that may not exist in the original binary form (Echopype). These steps span through multiple data processing levels (Echolevels), at which data products are stored with full provenance to maximize re-processing flexibility and usability. The data stores assembled under this use case can be easily leveraged by researchers in Use case 1 to jump start their experimentation of data cleaning parameters.

## 5. DISCUSSION

Echostack is an open-source software suite under active development. It will continue to evolve with the needs of the users and advancements in the scientific Python software ecosystem, in particular the Pandata and derived libraries in the wider geosciences domain. The Echostack tools were initially developed to address needs arising from our own research projects, but we quickly realized the potential value of these tools as open resources in the fisheries acoustic and ocean sciences communities. This paper serves as an interim report that describes our design considerations and discusses current capabilities and future plans together with three concrete use cases. Moving forward, we have two main goals: 1) enhancing Echostack features and performance, and 2) cultivating community user engagement. For the libraries, we aim to incorporate a wider set of echosounder data processing and visualization functions, ensure scalability with large datasets, improve documentation, and provide benchmark reports for library performance. For the community, we aim to broaden the discussion on development priorities and encourage code

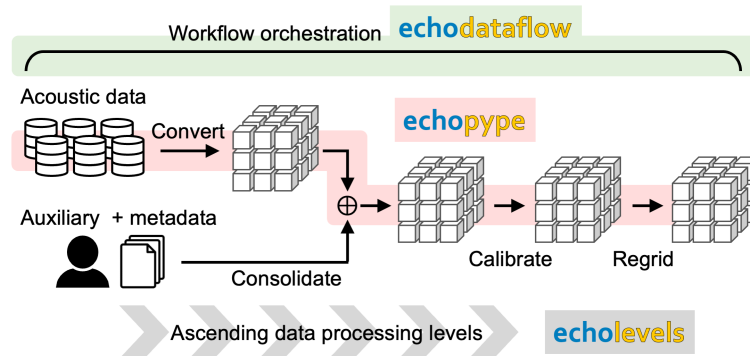


Figure 7. Use case 3: Mass production of cloud-optimized, analysis-ready data products

contribution from the user community, through intentional outreach efforts via both formal (conferences and workshops) and informal (interactions on code repositories and emails) channels. These goals are interdependent and form a positive feedback loop, as the more useful the software tools are, the more likely people will use it and contribute to its maintenance and development. The echosounder user community is diverse and includes members with a wide range of technical experiences and practical considerations, and would benefit from a suite of flexible tools that are both plug-and-play and customizable to meet specific requirements. It is our hope that the Echostack software packages can function as a catalyst for community-wide discussions and collaborations on advancing data processing workflow and information extraction in this domain.

#### ACKNOWLEDGEMENTS

We thank Alicia Billings, Dezhang Chu, Julia Clemons, Elizabeth Phillips, and Rebecca Thomas for discussions on code implementations and use cases, and Imran Majeed, Kavin Nguyen, Praneeth Ratna, and Brandyn Reyes for contributing to the code. The software development received funding support from NOAA Fisheries, NOAA Ocean Exploration, NSF, NASA, and the Gordon and Betty Moore and Alfred P. Sloan Foundations (MSDSE Environments), as well as engineering support from the University of Washington Scientific Software Engineering Center funded by Schmidt Futures as part of the Virtual Institute for Scientific Software.

#### REFERENCES

- [1] H. Medwin and C. S. Clay, *Fundamentals of acoustical oceanography*. Academic Press, 1998. doi: [10.1016/B978-0-12-487570-8.X5000-4](https://doi.org/10.1016/B978-0-12-487570-8.X5000-4).
- [2] T. K. Stanton, “30 years of advances in active bioacoustics: A personal perspective,” *Methods in Oceanography*, pp. 49–77, 2012, doi: [10.1016/j.mio.2012.07.002](https://doi.org/10.1016/j.mio.2012.07.002).
- [3] K. J. Benoit-Bird and G. L. Lawson, “Ecological insights from pelagic habitats acquired using active acoustic techniques,” *Annual Review of Marine Science*, vol. 8, no. 1, pp. 463–490, 2016, doi: [10.1146/annurev-marine-122414-034001](https://doi.org/10.1146/annurev-marine-122414-034001).
- [4] C. C. Wall, J. M. Jech, and S. J. McLean, “Increasing the accessibility of acoustic data through global access and imagery,” *ICES Journal of Marine Science: Journal du Conseil*, vol. 73, no. 8, pp. 2093–2103, 2016, doi: [10.1093/icesjms/fsw014](https://doi.org/10.1093/icesjms/fsw014).
- [5] T. A. Klevjer, X. Irigoien, A. Rstad, E. Fraile-Nuez, V. M. Benítez-Barrios, and S. Kaartvedt, “Large scale patterns in vertical distribution and behaviour of mesopelagic scattering layers,” *Scientific Reports*, vol. 6, no. 1, p. 19873, 2016, doi: [10.1038/srep19873](https://doi.org/10.1038/srep19873).
- [6] R. Korneliussen et al., “The Large Scale Survey System - LSSS,” in *Proceedings of the 29th Scandinavian Symposium on Physical Acoustics*, Ustaoset, Norway, 2006. [Online]. Available: <https://api.semanticscholar.org/CorpusID:204802910>
- [7] Y. Perrot et al., “Matecho: An Open-Source Tool for Processing Fisheries Acoustics Data,” *Acoustics Australia*, vol. 46, no. 2, pp. 241–248, 2018, doi: [10.1007/s40857-018-0135-x](https://doi.org/10.1007/s40857-018-0135-x).
- [8] Y. Ladrout, P. C. Escobar-Flores, A. C. G. Schimel, and R. L. O’Driscoll, “ESP3: An open-source software for the quantitative processing of hydro-acoustic data,” *SoftwareX*, vol. 12, p. 100581, 2020, doi: [10.1016/j.softx.2020.100581](https://doi.org/10.1016/j.softx.2020.100581).
- [9] J. A. Bednar and M. Durant, “The Pandata Scalable Open-Source Analysis Stack,” *Proceedings of the 22nd Python in Science Conference*, pp. 85–92, 2023, doi: [10.25080/gerudo-f2bc6f59-00b](https://doi.org/10.25080/gerudo-f2bc6f59-00b).
- [10] K. Haris, R. J. Kloser, T. E. Ryan, R. A. Downie, G. Keith, and A. W. Nau, “Sounding out life in the deep using acoustic data from ships of opportunity,” *Scientific Data*, vol. 8, no. 1, p. 23, 2021, doi: [10.1038/s41597-020-00785-8](https://doi.org/10.1038/s41597-020-00785-8).
- [11] J. Trowbridge et al., “The Ocean Observatories Initiative,” *Frontiers in Marine Science*, vol. 6, 2019, doi: [10.3389/fmars.2019.00074](https://doi.org/10.3389/fmars.2019.00074).
- [12] M. A. Moline, K. Benoit-Bird, D. O’Gorman, and I. C. Robbins, “Integration of scientific echo sounders with an adaptable autonomous vehicle to extend our understanding of animals from the surface to the bathypelagic,” *Journal of Atmospheric and Oceanic Technology*, vol. 32, no. 11, pp. 2173–2186, 2015, doi: [10.1175/JTECH-D-15-0035.1](https://doi.org/10.1175/JTECH-D-15-0035.1).
- [13] O. Brautaset et al., “Acoustic classification in multifrequency echosounder data using deep convolutional neural networks,” *ICES Journal of Marine Science*, vol. 77, no. 4, pp. 1391–1400, 2020, doi: [10.1093/ICESJMS/FSZ235](https://doi.org/10.1093/ICESJMS/FSZ235).

- [14] C. Choi, M. Kampffmeyer, N. O. Handegard, A.-B. Salberg, and R. Jenssen, "Deep semisupervised semantic segmentation in multifrequency echosounder data," *IEEE Journal of Oceanic Engineering*, pp. 1–17, 2023, doi: [10.1109/joe.2022.3226214](https://doi.org/10.1109/joe.2022.3226214).
- [15] S. S. Urmay, A. De Robertis, and C. Bassett, "A Bayesian inverse approach to identify and quantify organisms from fisheries acoustic data," *ICES Journal of Marine Science*, p. fsad102, 2023, doi: [10.1093/icesjms/fsad102](https://doi.org/10.1093/icesjms/fsad102).
- [16] Y. Zhang, C. C. Wall, J. M. Jech, and Q. Lv, "Developing a hybrid model with multiview learning for acoustic classification of Atlantic herring schools," *Limnology and Oceanography: Methods*, vol. 22, no. 5, pp. 351–368, 2024, doi: [10.1002/lom3.10611](https://doi.org/10.1002/lom3.10611).
- [17] L.-M. K. Harrison, M. J. Cox, G. Skaret, and R. Harcourt, "The R package EchoviewR for automated processing of active acoustic data using Echoview," *Frontiers in Marine Science*, vol. 2, 2015, doi: [10.3389/fmars.2015.00015](https://doi.org/10.3389/fmars.2015.00015).
- [18] W.-J. Lee, E. Mayorga, L. Setiawan, I. Majeed, K. Nguyen, and V. Staneva, "Echopype: A Python library for interoperable and scalable processing of water column sonar data for biological information," *arXiv:2111.00187 [eess]*, 2021, doi: [10.48550/arXiv.2111.00187](https://doi.org/10.48550/arXiv.2111.00187).
- [19] M. Hauser, A. Spring, J. Busecke, M. v. Driel, R. Lorenz, and readthedocs assistant, "regionmask/regionmask: version 0.12.1." [Online]. Available: <https://zenodo.org/records/10849860>
- [20] K. Williams, N. Lauffenburger, M.-C. Chuang, J.-N. Hwang, and R. Towler, "Automated measurements of fish within a trawl using stereo images from a Camera-Trawl device (CamTrawl)," *Methods in Oceanography*, vol. 17, pp. 138–152, 2016, doi: [10.1016/j.mio.2016.09.008](https://doi.org/10.1016/j.mio.2016.09.008).
- [21] R. Weaver, "Processing Levels," in *Encyclopedia of Remote Sensing*, E. G. Njoku, Ed., in Encyclopedia of Earth Sciences Series., New York, NY: Springer, 2014, pp. 517–520. doi: [10.1007/978-0-387-36699-9\\_36](https://doi.org/10.1007/978-0-387-36699-9_36).
- [22] Northwest Fisheries Science Center, Fishery Resource Analysis and Monitoring Division, "The 2021 Joint U.S.-Canada Integrated Ecosystem and Pacific Hake Acoustic Trawl Survey: Cruise Report SH-21-06," 2022, doi: [10.25923/0979-6D84](https://doi.org/10.25923/0979-6D84).