The Journal of Open Source Software

ROMS-Tools: Reproducible Preprocessing and Analysis for Regional Ocean Modeling with ROMS

Nora Loose[1], **Tom Nicholas**[2], **Scott Eilerman**[1], **Christopher McBride**[1], **Sam Maticka**[1], **Dafydd Stephenson**[1], **Scott Bachman**[1], **Pierre Damien**[3], **Ulla Heede**[1], **Alicia Karspeck**[1], **Matthew C. Long**[1], **M. Jeroen Molemaker**[3], and **Abigale Wyatt**[1]

1 [C]Worthy LLC, Boulder, CO, United States **2** Earthmover PBC **3** University of California, Los Angeles, CA, United States

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Open Journals

Reviewers:

- @openjournals

Submitted: 01 January 1970
Published: unpublished

Summary

The ocean regulates Earth's climate and sustains marine ecosystems by circulating and storing heat, carbon, oxygen, and nutrients, while exchanging heat and gases with the atmosphere. Scientists study these processes using ocean models, which simulate the ocean on a grid. **Regional ocean models** focus computational resources on a limited geographical area with fine grid spacing, and can resolve fine-scale phenomena such as mesoscale and submesoscale features, tidal dynamics, coastal currents, upwelling, and detailed biogeochemical (BGC) processes. A widely used regional ocean model is the **Regional Ocean Modeling System (ROMS)** (Shchepetkin & McWilliams, 2005). ROMS has been coupled to the Marine Biogeochemistry Library (MARBL) (Long et al., 2021; Molemaker & contributors, 2025a) to link physical and BGC processes. ROMS-MARBL supports research on environmental management, fisheries, regional climate impacts, and ocean-based carbon dioxide removal (CDR) strategies.

`ROMS-Tools` is a Python package that streamlines the **preparation and analysis of ROMS-MARBL simulations** by enabling users to generate regional grids, prepare model inputs efficiently, and analyze model outputs. A detailed overview of the package's functionality is available in the `ROMS-Tools` documentation. By providing a modern, user-friendly interface, `ROMS-Tools` lowers technical barriers, improves reproducibility, and allows scientists to focus on research rather than data preparation. The package is installable via Conda or PyPI and can be run interactively in Jupyter notebooks.

Statement of Need

Regional ocean models are essential tools for research in marine ecosystems, climate dynamics, and ocean-based CDR. However, configuring a regional ocean model like ROMS-MARBL is technically demanding. Model setup requires initialization and time-dependent forcing from oceanic and atmospheric datasets, drawn from multiple external sources in diverse formats. These global source datasets can span petabytes and must be subsetted, processed, and mapped onto the target model grid, producing 10–100 terabytes of input data for large regional domains. Generating these input files is time-consuming, error-prone, and difficult to reproduce. These challenges create a bottleneck for both new and experienced users, slow down science, and limit collaboration across groups.

Existing tools within the ocean modeling ecosystem do not fully address these challenges for ROMS-MARBL or ROMS users. While legacy MATLAB-based scripts developed at UCLA (Molemaker, 2024) and Python packages such as `pyroms` (Hedstrom & contributors, 2023) provide critical functionality, both rely on low-level, manually coordinated steps that limit reproducibility, maintainability, and accessibility. Moreover, frameworks developed for other ocean models cannot be directly applied to ROMS due to fundamental differences in grid geometry, vertical coordinates, and model input requirements. As a result, users lack a modern, integrated framework for reproducible model setup and analysis that is specifically designed for ROMS and ROMS-MARBL.

`ROMS-Tools` was developed to fill this gap. It is an open-source Python framework designed for researchers and practitioners who run ROMS or ROMS-MARBL regional ocean simulations, including users in physical oceanography, marine biogeochemistry, and ocean-based CDR applications. Current capabilities are fully compatible with UCLA-ROMS (Molemaker & contributors, 2025a, 2025b), with potential support for other ROMS implementations, such as Rutgers ROMS (Arango & contributors, 2024), in the future. The package handles large input and output datasets via parallel computation with `dask` (Dask Development Team, 2016), making workflows scalable from laptops to high-performance computing clusters. Built-in visualization tools enable quick inspection of regional grids as well as model input and output fields. For example, Figure 1 shows surface initial conditions for a California Current System simulation at 5 km horizontal resolution, generated and visualized directly using `ROMS-Tools`. By lowering technical barriers and improving transparency and reproducibility, `ROMS-Tools` enables more efficient model development, facilitates scientific collaboration, and supports applications such as verification of marine carbon removal strategies.

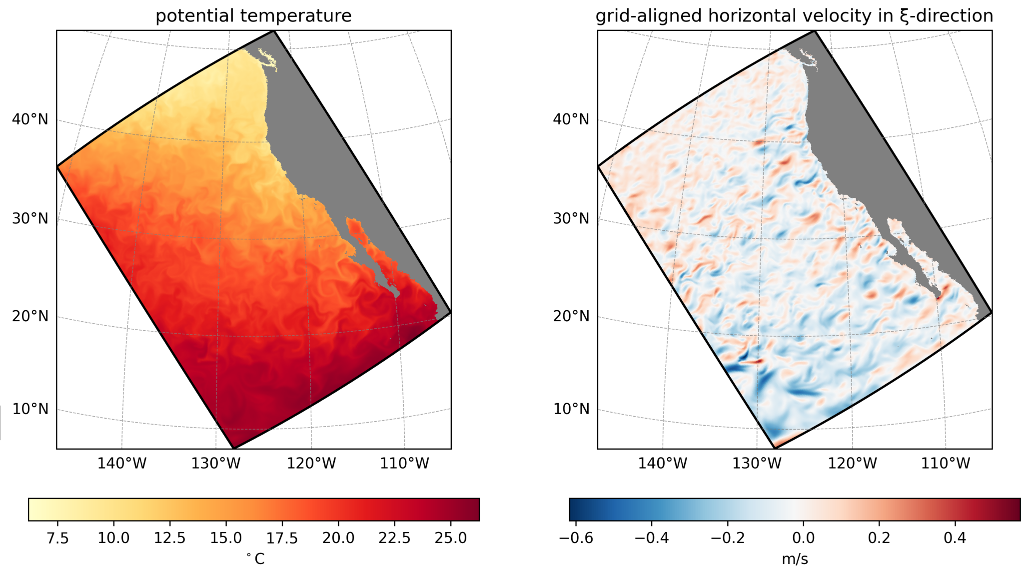

Figure 1: Surface initial conditions for the California Current System created and visualized with `ROMS-Tools`. Left: potential temperature. Right: grid-aligned horizontal velocity in ξ-direction. Shown for January 1, 2000.

State of the Field

Historically, setting up a regional ocean model required a patchwork of custom scripts and lab-specific workflows, resulting in error-prone and difficult-to-reproduce processes. Within the ROMS community, tools like `pyroms` (Hedstrom & contributors, 2023) addressed some of these issues by providing low-level Python utilities for preprocessing ROMS model inputs. However, `pyroms` has several limitations: installation is cumbersome due to Python/Fortran dependencies, the API is inconsistent, and documentation and tests are missing. The package does not support modern tools such as `xarray` (Hoyer & Hamman, 2017), nor reproducible workflows. Active development has ceased, and maintenance (including compatibility with newer Python versions) is no longer provided. Together, these limitations make it very difficult to add new features, such as support for BGC and CDR applications, and improvements to user-friendliness.

Tools from other modeling communities cannot be directly applied to ROMS because each model has distinct structural requirements and input conventions. For example, the `regional-mom6` package (Barnes et al., 2024), developed for regional configurations of the Modular Ocean Model v6 (MOM6) (Adcroft et al., 2019), cannot generate ROMS inputs. ROMS uses a terrain-following vertical coordinate system that requires specialized vertical regridding, whereas MOM6 accepts inputs on arbitrary depth levels and does not require vertical regridding at all. While ROMS and MOM6 differ in fundamental ways, `regional-mom6` represents the closest comparable tool to `ROMS-Tools` in the wider modeling ecosystem. Notably, the main development cycles of `regional-mom6` and `ROMS-Tools` overlapped (`regional-mom6`: 2023–2024; `ROMS-Tools`: 2024–2025, based on public GitHub commits). Had the developers been aware of each other, a shared framework could potentially have been created, with model-specific adaptations layered on top. Adapting one framework to the other now would require extensive architectural changes.

Legacy MATLAB preprocessing scripts developed at UCLA (Molemaker, 2024) encapsulate decades of expertise in configuring regional ocean models, but require users to edit source code directly, making workflows error-prone, difficult to reproduce, and challenging to extend to new datasets or applications. `ROMS-Tools` provides a modern, open-source Python implementation of these scripts, retaining core algorithms while offering high-level APIs, automated intermediate steps, and explicit workflow state management via YAML. This object-oriented design improves reproducibility, reduces user errors, and supports extensibility, while leveraging modern Python tools such as `xarray` and `dask`. In some cases, `ROMS-Tools` diverges from the original MATLAB implementation to incorporate improved methods or better integrate with the Python ecosystem.

Software Design

`ROMS-Tools` emphasizes ease of use, flexibility, reproducibility, and scalability through a modular architecture and high-level user interfaces.

Design Trade-Offs

A central design trade-off in `ROMS-Tools` is between **user control** and **automation**. Rather than enforcing a fixed workflow, the package exposes key choices such as physical options (e.g., corrections for radiation or wind), interpolation and fill methods, and computational backends. This approach contrasts with opinionated frameworks that fix defaults and directory structures to maximize automation. While users must make explicit decisions, some steps remain automated to prevent errors. For example, bathymetry smoothing is applied automatically using a fixed, non-tunable parameter, since insufficient or omitted smoothing can crash simulations due to pressure gradient errors. This design choice addresses issues encountered by new users of the original UCLA MATLAB scripts and balances flexibility with safety, enabling experimentation while avoiding common pitfalls.

Another key design consideration is balancing **modular, incremental workflow steps** with **reproducibility**. `ROMS-Tools` organizes tasks (such as creating `InitialConditions`, `BoundaryForcing`, and `SurfaceForcing`) into small, composable components that can be executed, saved, and revisited independently, rather than following a monolithic, fixed workflow. All components depend on the `Grid`, but once it is created, the remaining objects are independent. This modular approach avoids unnecessary recomputation when only some inputs change but requires careful tracking of workflow state. To ensure reproducibility, all configuration choices are stored in compact, text-based YAML files. These files are version-controllable, easy to share, and eliminate the need to transfer large model input NetCDF datasets.

Architecture

At the user-facing level, `ROMS-Tools` provides high-level objects such as `Grid`, `InitialConditions`, and `BoundaryForcing`. Each object exposes a consistent interface (`.ds`, `.plot()`, `.save()`, `.to_yaml()`), allowing users to apply the same methods across workflow steps and inspect standardized attributes that are always present. This consistency reduces cognitive overhead and makes workflows predictable.

Internally, `ROMS-Tools` follows a layered, modular architecture. Low-level dataset classes (`LatLonDataset`, `ROMSDataset`) handle data ingestion and preprocessing tasks such as subdomain selection and lateral land filling. Source-specific datasets (e.g., `ERA5Dataset`, `GLORYSDataset`, `SRTMDataset`) inherit from these base classes and encode dataset-specific conventions like variable names, coordinates, and masking. Supporting a new data source typically requires only a small subclass defining these mappings while reusing existing preprocessing logic, minimizing changes to the core code.

High-level classes (`Grid`, `InitialConditions`, `BoundaryForcing`) build on these low-level dataset abstractions to generate ready-to-use modeling inputs through operations such as regridding and final assembly. This layered design improves **extensibility and maintainability**.

Computational and Data Model Choices

`ROMS-Tools` is built on `xarray`, which provides a clear, consistent interface for exploring and inspecting labeled, multi-dimensional geophysical datasets. Users can take advantage of `xarray`'s intuitive indexing, plotting, and metadata handling. Optional `dask` enables parallel and out-of-core computation for very large input and output datasets.

Research Impact Statement

`ROMS-Tools` is used by two primary research communities. First, regional ocean modelers use it to generate input datasets for ROMS simulations; external users include researchers at PNNL, WHOI, Stanford University, and UCLA. Second, researchers in the ocean-based carbon dioxide removal (CDR) community use `ROMS-Tools` to configure reproducible ROMS–MARBL simulations of climate intervention scenarios, with adopters including [C]Worthy, Carbon to Sea, Ebb Carbon, and SCCWRP. All of these groups have contacted the developers directly or engaged with the project through GitHub or offline discussions. Several manuscripts from these communities are currently in preparation.

Beyond standalone use, `ROMS-Tools` is integrated into broader scientific workflows, including C-Star (Stephenson & contributors, 2025), an open-source platform under development to provide scientifically credible monitoring, reporting, and verification (MRV) for the emerging marine carbon market.

Additional evidence of community uptake comes from public usage metrics. At the time of writing, the GitHub repository shows 119 unique cloners in the past 14 days, with stars from users at institutions including the University of Waikato, NCAR, University of Maryland, National Oceanography Centre, McGill University, UC Santa Cruz, and others. Distribution statistics indicate over 3,100 conda-forge downloads in the past six months, including 68 downloads of the most recent release (`v3.3.0`), and more than 48,000 total PyPI downloads. PyPI counts include automated continuous integration (CI) usage by `ROMS-Tools`, in addition to direct user installations. In contrast, conda-forge downloads of `v3.3.0` reflect exclusively human-initiated installs, as C-Star's CI workflows currently pin pre-`v3.3.0` releases of `ROMS-Tools`.

The Journal of Open Source Software

AI Usage Disclosure

Generative AI tools were used to help write docstrings, develop tests, and improve the clarity and readability of both the `ROMS-Tools` documentation and manuscript text. All AI-assisted content was reviewed and verified by the authors for technical accuracy and correctness.

Acknowledgements

Development of `ROMS-Tools` has been supported by ARPA-E (DE-AR0001838) and philanthropic donations to [C]Worthy from the Grantham Foundation for the Environment, the Chan Zuckerberg Initiative, Founders Pledge, and the Ocean Resilience Climate Alliance.

References

Adcroft, A., Anderson, W., Balaji, V., Blanton, C., Bushuk, M., Dufour, C. O., Dunne, J. P., Griffies, S. M., Hallberg, R., Harrison, M. J., Held, I. M., Jansen, M. F., John, J. G., Krasting, J. P., Langenhorst, A. R., Legg, S., Liang, Z., McHugh, C., Radhakrishnan, A., … Zhang, R. (2019). The GFDL Global Ocean and Sea Ice Model OM4.0: Model Description and Simulation Features. *Journal of Advances in Modeling Earth Systems*, *11*(10), 3167–3211. https://doi.org/10.1029/2019MS001726

Arango, H., & contributors. (2024). Rutgers ROMS. In *GitHub repository*. GitHub. https://github.com/myroms/roms

Barnes, A. J., Constantinou, N. C., Gibson, A. H., Kiss, A. E., Chapman, C., Reilly, J., Bhagtani, D., & Yang, L. (2024). Regional-mom6: A Python package for automatic generation of regional configurations for the Modular Ocean Model 6. *Journal of Open Source Software*, *9*(100), 6857. https://doi.org/10.21105/joss.06857

Dask Development Team. (2016). *Dask: Library for dynamic task scheduling*. http://dask.pydata.org

Hedstrom, K., & contributors. (2023). Pyroms. In *GitHub repository*. GitHub. https://github.com/ESMG/pyroms

Hoyer, S., & Hamman, J. (2017). Xarray: ND labeled arrays and datasets in python. *Journal of Open Research Software*, *5*(1). https://doi.org/10.5334/jors.148

Long, M. C., Moore, J. K., Lindsay, K., Levy, M., Doney, S. C., Luo, J. Y., Krumhardt, K. M., Letscher, R. T., Grover, M., & Sylvester, Z. T. (2021). *Simulations With the Marine Biogeochemistry Library (MARBL)*. https://doi.org/10.1029/2021MS002647

Molemaker, J. (2024). UCLA MATLAB tools. In *GitHub repository*. GitHub. https://github.com/nmolem/ucla-tools

Molemaker, J., & contributors. (2025a). UCLA-ROMS. In *GitHub repository*. GitHub. https://github.com/CESR-lab/ucla-roms

Molemaker, J., & contributors. (2025b). UCLA-ROMS. In *GitHub repository*. GitHub. https://github.com/CWorthy-ocean/ucla-roms

Shchepetkin, A. F., & McWilliams, J. C. (2005). The regional oceanic modeling system (ROMS): A split-explicit, free-surface, topography-following-coordinate oceanic model. *Ocean Modelling*, *9*(4), 347–404. https://doi.org/10.1016/j.ocemod.2004.08.002

Stephenson, D., & contributors. (2025). *C-star*. [C]Worthy. https://github.com/CWorthy-ocean/C-Star