The Journal of Open Source Software

Zarrtraj: A Python package for streaming molecular dynamics trajectories from cloud services

Lawson Woods [1,2], **Hugo MacDermott-Opeskin** [3], **Edis Jakupovic** [4,5], **Yuxuan Zhuang** [6,7], **Richard J Gowers** [8], **and Oliver Beckstein** [4,5]

1 School of Computing and Augmented Intelligence, Arizona State University, Tempe, Arizona, United States of America **2** School of Molecular Sciences, Arizona State University, Tempe, Arizona, United States of America **3** Open Molecular Software Foundation, Davis, CA, United States of America **4** Center for Biological Physics, Arizona State University, Tempe, AZ, United States of America **5** Department of Physics, Arizona State University, Tempe, Arizona, United States of America **6** Department of Computer Science, Stanford University, Stanford, CA 94305, USA. **7** Departments of Molecular and Cellular Physiology and Structural Biology, Stanford University School of Medicine, Stanford, CA 94305, USA. **8** Charm Therapeutics, London, United Kingdom

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Open Journals

Reviewers:

- @openjournals

Submitted: 01 January 1970
Published: unpublished

Summary

Molecular dynamics (MD) simulations provide a microscope into the behavior of atomic-scale environments otherwise prohibitively difficult to observe. However, the resulting trajectory data are too often siloed in a single institutions' HPC environment, rendering it unusable by the broader scientific community. Additionally, it is increasingly common for trajectory data to be entirely stored in a cloud storage provider, rather than a traditional on-premise storage site. *Zarrtraj* enables these trajectories to be read directly from cloud storage providers like AWS, Google Cloud, and Microsoft Azure into MDAnalysis, a popular Python package for analyzing trajectory data, providing a method to open up access to trajectory data to anyone with an internet connection. Enabling cloud streaming for MD trajectories empowers easier replication of published analysis results, analyses of large, conglomerate datasets from different sources, and training machine learning models without downloading and storing trajectory data.

Statement of need

The computing power in HPC environments has increased to the point where running simulation algorithms is often no longer the constraint in obtaining scientific insights from molecular dynamics trajectory data. Instead, the ability to process, analyze and share large volumes of data provide new constraints on research in this field (Abraham et al., 2019).

Other groups in the field recognize this same need for adherence to FAIR principles (Stall et al., 2019) including MDsrv, a tool that can stream MD trajectories into a web browser for visual exploration (Kampfrath et al., 2022), GCPRmd, a web service that builds on MDsrv to provide a predefined set of analysis results and simple geometric features for G-protein-coupled receptors (Hildebrand et al., 2019) (Rodríguez-Espigares et al., 2020), MDDB (Molecular Dynamics Data Bank), an EU-scale repository for bio-simulation data (Amaro et al., 2024), and MDverse, a prototype search engine for publicly-available GROMACS simulation data (Tiemann et al., 2024).

While these efforts currently offer solutions for indexing, searching, and visualizing MD trajectory data, the problem of distributing trajectories in way that enables *NumPy*-like slicing and parallel reading for use in arbitrary analysis tasks remains.

Although exposing download links on the open internet offers a simple solution to this problem,

on-disk representations of molecular dynamics trajectories often range in size up to TBs in scale (Tu et al., 2010) (*Foldingathome COVID-19 Datasets*, n.d.), so a solution which could prevent this duplication of storage and unnecessary download step would provide greater utility for the computational molecular sciences ecosystem, especially if it provides access to slices or subsampled portions of these large files.

To address this need, we developed *Zarrtraj* as a prototype for streaming trajectories into analysis software using an established trajectory format. *Zarrtraj* extends MDAnalysis (Gowers et al., 2016), a popular Python-based library for the analysis of molecular simulation data in a wide range of formats, to also accept remote file locations for trajectories instead of local filenames. Instead of being integrated directly into MDAnalysis, *Zarrtraj* is built as an external MDAKit (Alibay et al., 2023) that automatically registers its capabilities with MDAnalysis on import and thus acts as a plugin. *Zarrtraj* enables streaming MD trajectories in the popular HDF5-based H5MD format (de Buyl et al., 2014) from AWS S3, Google Cloud Buckets, and Azure Blob Storage and Data Lakes without ever downloading them. *Zarrtraj* relies on the *Zarr* (Alistair Miles et al., 2024) package for streaming array-like data from a variety of storage mediums and on Kerchunk, which extends the capability of *Zarr* by allowing it to read HDF5 files. *Zarrtraj* leverages *Zarr*'s ability to read a slice of a file and to read a file in parallel and it implements the standard MDAnalysis trajectory reader API, which taken together make it compatible with analysis algorithms that use the "split-apply-combine" parallelization strategy (Wickham, 2011). In addition to the H5MD format, *Zarrtraj* can stream and write trajectories in the experimental ZarrMD format, which ports the H5MD layout to the *Zarr* file type.

This work builds on the existing MDAnalysis `H5MDReader` (Jakupovic & Beckstein, 2021), and uses *NumPy* (Harris et al., 2020) as a common interface in-between MDAnalysis and the file storage medium. *Zarrtraj* was inspired and made possible by similar efforts in the geosciences community to align data practices with FAIR principles (Stern et al., 2022).

With *Zarrtraj*, we envision research groups making their data publicly available via a cloud URL so that anyone can reuse their trajectories and reproduce their results. Large databases, like MDDB and MDverse, can expose a URL associated with each trajectory in their databases so that users can make a query and immediately use the resulting trajectories to run an analysis on the hits that match their search. Groups seeking to collect a large volume of trajectory data to train machine learning models (Jackson et al., 2023) can make use of our tool to efficiently and inexpensively obtain the data they need from these published URLs.

Features and Benchmarks

Once imported, *Zarrtraj* allows passing trajectory URLs just like ordinary files:

```
import zarrtraj
import MDAnalysis as mda

u = mda.Universe("topology.pdb", "s3://sample-bucket-name/trajectory.h5md")
```

Initial benchmarks show that *Zarrtraj* can iterate serially through an AWS S3 cloud trajectory (load into memory one frame at a time) at roughly 1/2 or 1/3 the speed it can iterate through the same trajectory from disk and roughly 1/5 to 1/10 the speed it can iterate through the same trajectory on disk in XTC format (Figure 1). However, it should be noted that this speed is influenced by network bandwidth and that writing parallelized algorithms can offset this loss of speed as in Figure 2.

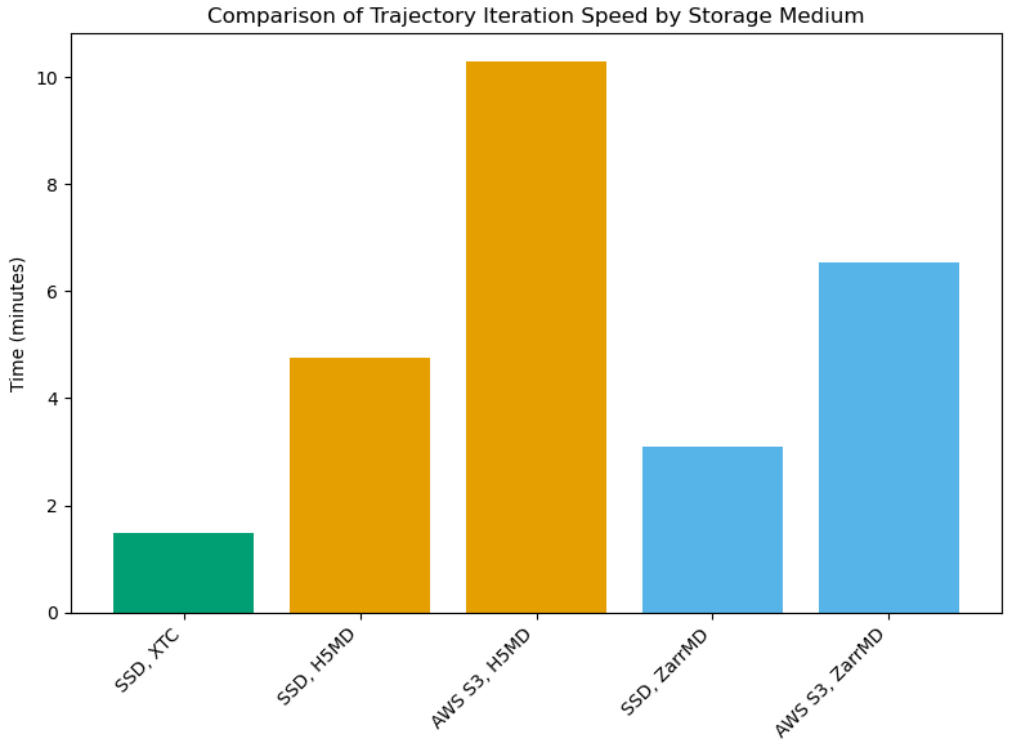

Figure 1: Benchmarks performed on a machine with 2 Intel Xeon 2.00GHz CPUs, 32GB of RAM, and an SSD configured with RAID 0. The trajectory used for benchmarking was the YiiP trajectory from MDAnalysisData (Fan & Beckstein, 2019), a 9000-frame (90ns), 111,815 particle simulation of a membrane-protein system. The original 3.47GB XTC trajectory was converted into an uncompressed 11.3GB H5MD trajectory and an uncompressed 11.3GB ZarrMD trajectory using the MDAnalysis `H5MDWriter` and *Zarrtraj* `ZarrMD` writers, respectively. XTC trajectory read using the MDAnalysis `XTCReader` for comparison.

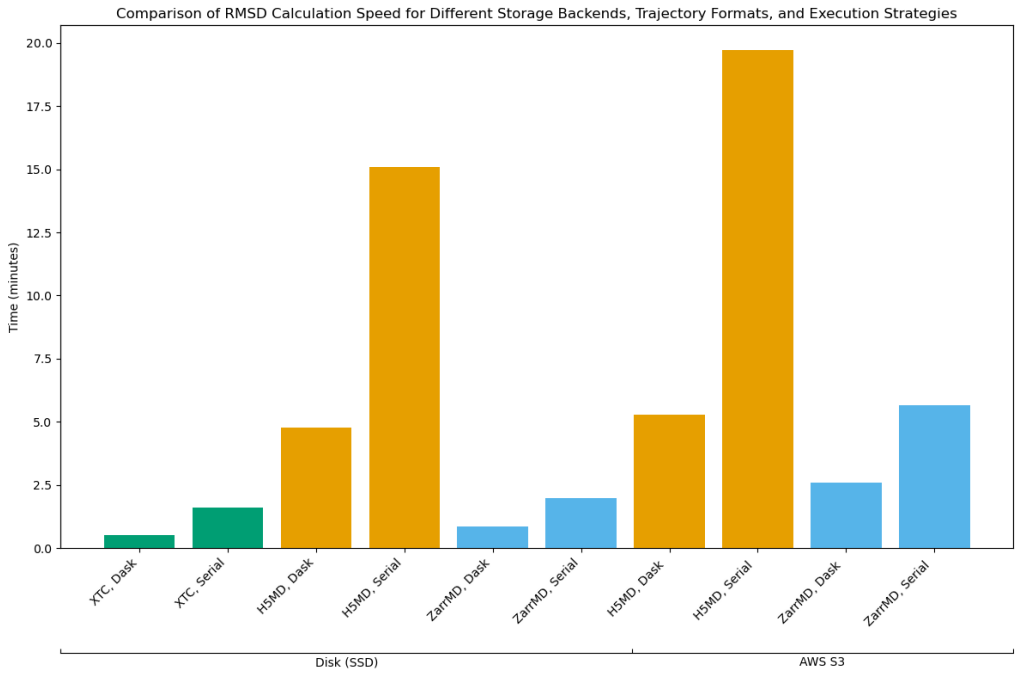

Figure 2: RMSD benchmarks performed on the same machine as Figure 1. YiiP trajectory aligned to first frame as reference using `MDAnalysis.analysis.align.AlignTraj` and converted to compressed, quantized H5MD (7.8GB) and ZarrMD (4.9GB) trajectories. RMSD performed using development branch of MDAnalysis (2.8.0dev) with "serial" and "dask" backends. See this notebook for full benchmark codes.

Zarrtraj is capable of making use of *Zarr*'s powerful compression and quantization when writing ZarrMD trajectories. The uncompressed MDAnalysisData YiiP trajectory in ZarrMD format is reduced from 11.3GB uncompressed to just 4.9GB after compression with the Zstandard algorithm (Collet & Kucherawy, 2021) and quantization to 3 digits of precision. See performance considerations for more.

Example

The YiiP membrane protein trajectory (Fan & Beckstein, 2019) used for benchmarking in this paper is publicly available for streaming from the Google Cloud Bucket *gcs://zarrtraj-test-data/yiip.zarrmd*. The topology file in PDB format, which contains information about the chemical composition of the system, can also be accessed remotely from the same bucket (*gcs://zarrtraj-test-data/YiiP_system.pdb*) using fsspec, although this is currently an experimental feature and details may change.

In the following example (see also the YiiP Example in the zarrtraj docs), we access the topology file and the trajectory from the *gcs://zarrtraj-test-data* cloud bucket. We initially create an `MDAnalysis.Universe`, the basic object in MDAnalysis that ties static topology data and dynamic trajectory data together and manages access to all data. We iterate through a slice of the trajectory, starting from frame index 100 and skipping forward in steps of 20 frames:

```
import zarrtraj
import MDAnalysis as mda
import fsspec

with fsspec.open("gcs://zarrtraj-test-data/YiiP_system.pdb", "r") as top:
```

```
u = mda.Universe(top, "gcs://zarrtraj-test-data/yiip.zarrmd",
                 topology_format="PDB")

for timestep in u.trajectory[100::20]:
    print(timestep)
```

Inside the loop over trajectory frames we print information for the current frame `timestep` although in principle, any kind of analysis code can run here and process the coordinates available in `u.atoms.positions`.

The `Universe` object can be used as if the underlying trajectory file were a local file. For example, we can use u from the preceeding example with one of the standard analysis tools in MDAnalysis, the calculation of the root mean square distance (RMSD) after optimal structural superposition (Liu et al., 2010) in the `MDAnalysis.analysis.rms.RMSD` class. In the example below we select only the C_α atoms of the protein with a MDAnalysis selection. We run the analysis with the `.run()` method while stepping through the trajectory at increments of 100 frames. We then print the first and last data point from the results array:

```
>>> import MDAnalysis.analysis.rms
>>> R = MDAnalysis.analysis.rms.RMSD(u, select="protein and name CA").run(
                                     step=100, verbose=True)
100%|████████████████████████████████████| 91/91 [00:28<00:00,  3.21it/s]
>>> print(f"Initial RMSD (frame={R.results.rmsd[0, 0]:g}): "
          f"{R.results.rmsd[0, 2]:.3f} Å")
Initial RMSD (frame=0) : 0.000 Å
>>> print(f"Final RMSD (frame={R.results.rmsd[-1, 0]:g}): "
          f"{R.results.rmsd[-1, 2]:.3f} Å")
Final RMSD (frame=9000) : 2.373 Å
```

This example demonstrates that the *Zarrtraj* interface enables seamless use of cloud-hosted trajectories with the standard tools that are either available with MDAnalysis itself, through MDAKits (Alibay et al., 2023) (see the MDAKit registry for available packages), or any script or package that uses MDAnalysis for file I/O.

Acknowledgements

We thank Dr. Jenna Swarthout Goddard for supporting the GSoC program at MDAnalysis and Dr. Martin Durant, author of Kerchunk, for helping refine and merge features in his upstream code base necessary for this project. LW was a participant in the Google Summer of Code 2024 program. Some work on *Zarrtraj* was supported by the National Science Foundation under grant number 2311372.

References

Abraham, M., Apostolov, R., Barnoud, J., Bauer, P., Blau, C., Bonvin, A. M. J. J., Chavent, M., Chodera, J., Čondić-Jurkić, K., Delemotte, L., Grubmüller, H., Howard, R. J., Jordan, E. J., Lindahl, E., Ollila, O. H. S., Selent, J., Smith, D. G. A., Stansfeld, P. J., Tiemann, J. K. S., … Zhmurov, A. (2019). Sharing data from molecular simulations. *Journal of Chemical Information and Modeling*, *59*(10), 4093–4099. https://doi.org/10.1021/acs.jcim.9b00665

Alibay, I., Wang, L., Naughton, F., Kenney, I., Barnoud, J., Gowers, R., & Beckstein, O. (2023). MDAKits: A framework for FAIR-compliant molecular simulation analysis. *Proceedings of the Python in Science Conference*, 76–84. https://doi.org/10.25080/gerudo-f2bc6f59-00a

Alistair Miles, jakirkham, M Bussonnier, Josh Moore, Dimitri Papadopoulos Orfanos, Davis Bennett, David Stansby, Joe Hamman, James Bourbeau, Andrew Fulton,

Gregory Lee, Ryan Abernathey, Norman Rzepka, Zain Patel, Mads R. B. Kristensen, Sanket Verma, Saransh Chopra, Matthew Rocklin, AWA BRANDON AWA, … shikharsg. (2024). *Zarr-developers/zarr-python: v3.0.0-alpha*. Zenodo. https://doi.org/10.5281/ZENODO.3773449

Amaro, R., Åqvist, J., Bahar, I., Battistini, F., Bellaiche, A., Beltran, D., Biggin, P. C., Bonomi, M., Bowman, G. R., Bryce, R., Bussi, G., Carloni, P., Case, D., Cavalli, A., Chang, C.-E. A., au2, T. E. C. I., Cheung, M. S., Chipot, C., Chong, L. T., … Orozco, M. (2024). *The need to implement FAIR principles in biomolecular simulations*. https://doi.org/10.48550/arXiv.2407.16584

Collet, Y., & Kucherawy, M. (2021). *Zstandard Compression and the 'application/zstd' Media Type* (No. 8878). RFC 8878; RFC Editor. https://doi.org/10.17487/RFC8878

de Buyl, P., Colberg, P. H., & Höfling, F. (2014). H5MD: A structured, efficient, and portable file format for molecular data. *Computer Physics Communications*, *185*(6), 1546–1553. https://doi.org/10.1016/j.cpc.2014.01.018

Fan, S., & Beckstein, O. (2019). *Molecular Dynamics trajectories of membrane protein YiiP*. https://doi.org/10.6084/m9.figshare.8202149.v1

Foldingathome COVID-19 datasets. (n.d.). https://registry.opendata.aws/foldingathome-covid19

Gowers, Richard J., Linke, Max, Barnoud, Jonathan, Reddy, Tyler J. E., Melo, Manuel N., Seyler, Sean L., Domański, Jan, Dotson, David L., Buchoux, Sébastien, Kenney, Ian M., & Beckstein, Oliver. (2016). MDAnalysis: A Python Package for the Rapid Analysis of Molecular Dynamics Simulations. In Sebastian Benthall & Scott Rostrup (Eds.), *Proceedings of the 15th Python in Science Conference* (pp. 98–105). https://doi.org/10.25080/Majora-629e541a-00e

Harris, C. R., Millman, K. J., Walt, S. J. van der, Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., Kerkwijk, M. H. van, Brett, M., Haldane, A., Río, J. F. del, Wiebe, M., Peterson, P., … Oliphant, T. E. (2020). Array programming with NumPy. *Nature*, *585*(7825), 357–362. https://doi.org/10.1038/s41586-020-2649-2

Hildebrand, P. W., Rose, A. S., & Tiemann, J. K. S. (2019). Bringing molecular dynamics simulation data into view. *Trends in Biochemical Sciences*, *44*(11), 902–913. https://doi.org/10.1016/j.tibs.2019.06.004

Jackson, N. E., Savoie, B. M., Statt, A., & Webb, M. A. (2023). Introduction to machine learning for molecular simulation. *Journal of Chemical Theory and Computation*, *19*(14), 4335–4337. https://doi.org/10.1021/acs.jctc.3c00735

Jakupovic, E., & Beckstein, O. (2021). MPI-parallel Molecular Dynamics Trajectory Analysis with the H5MD Format in the MDAnalysis Python Package. In M. Agarwal, C. Calloway, D. Niederhut, & D. Shupe (Eds.), *Proceedings of the 20th Python in Science Conference* (pp. 40–48). https://doi.org/10.25080/majora-1b6fd038-005

Kampfrath, M., Staritzbichler, R., Hernández, G. P., Rose, A. S., Tiemann, J. K. S., Scheuermann, G., Wiegreffe, D., & Hildebrand, P. W. (2022). MDsrv: visual sharing and analysis of molecular dynamics simulations. *Nucleic Acids Research*, *50*(W1), W483–W489. https://doi.org/10.1093/nar/gkac398

Liu, P., Agrafiotis, D. K., & Theobald, D. L. (2010). Fast determination of the optimal rotational matrix for macromolecular superpositions. *J Comput Chem*, *31*(7), 1561–1563. https://doi.org/10.1002/jcc.21439

Rodríguez-Espigares, I., Torrens-Fontanals, M., Tiemann, J. K. S., Aranda-García, D., Ramírez-Anguita, J. M., Stepniewski, T. M., Worp, N., Varela-Rial, A., Morales-Pastor, A., Medel-Lacruz, B., Pándy-Szekeres, G., Mayol, E., Giorgino, T., Carlsson, J., Deupi, X., Filipek,

S., Filizola, M., Gómez-Tamayo, J. C., Gonzalez, A., … Selent, J. (2020). GPCRmd uncovers the dynamics of the 3D-GPCRome. *Nature Methods*, *17*(8), 777–787. https://doi.org/10.1038/s41592-020-0884-y

Stall, S., Yarmey, L., Cutcher-Gershenfeld, J., Hanson, B., Lehnert, K., Nosek, B., Parsons, M., Robinson, E., & Wyborn, L. (2019). Make scientific data FAIR. *Nature*, *570*(7759), 27–29. https://doi.org/10.1038/d41586-019-01720-7

Stern, C., Abernathey, R., Hamman, J., Wegener, R., Lepore, C., Harkins, S., & Merose, A. (2022). Pangeo forge: Crowdsourcing analysis-ready, cloud optimized data production. *Frontiers in Climate*, *3*. https://doi.org/10.3389/fclim.2021.782909

Tiemann, J. K., Szczuka, M., Bouarroudj, L., Oussaren, M., Garcia, S., Howard, R. J., Delemotte, L., Lindahl, E., Baaden, M., Lindorff-Larsen, K., Chavent, M., & Poulain, P. (2024). MDverse, shedding light on the dark matter of molecular dynamics simulations. *eLife*, *12*, RP90061. https://doi.org/10.7554/eLife.90061

Tu, T., Rendleman, C. A., Miller, P. J., Sacerdoti, F., Dror, R. O., & Shaw, D. E. (2010). Accelerating parallel analysis of scientific simulation data via zazen. *Proceedings of the 8th USENIX Conference on File and Storage Technologies*, 10.

Wickham, H. (2011). The split-apply-combine strategy for data analysis. *Journal of Statistical Software*, *40*(1), 1–29. https://doi.org/10.18637/jss.v040.i01