Employing Artificial Neural Networks for Optimal Storage and Facile Sharing of Molecular Dynamics Simulation Trajectories

Abdul Wasim,[1,*] Lars V. Schäfer,[2,†] and Jagannath Mondal[1,‡]

[1] *Tata Institute of Fundamental Research Hyderabad, Telangana 500046, India*

[2] *Center for Theoretical Chemistry, Faculty of Chemistry and Biochemistry, Ruhr-Universität Bochum, 44801 Bochum, Germany*

Abstract

With the remarkable stride in computing power and advances in Molecular Dynamics simulation programs, a crucial challenge of storing and sharing large biomolecular simulation datasets has emerged. By leveraging AutoEncoders, a type of artificial neural network, we developed a method to compress MD trajectories into significantly smaller latent spaces. Our method can save upto 98% in disk space compared to XTC, a highly compressed trajectory format from the widely used MD program package GROMACS, thus facilitating easier storage and sharing of simulation trajectories. Atomic coordinates are very accurately reconstructed from compressed data. The method was tested across a variety of biomolecular systems, including folded proteins, intrinsically disordered proteins (IDPs), and protein-ligand complexes, showing consistent accuracy in reconstruction. Notably, the compression efficiency was particularly beneficial for larger systems. This approach enables the scientific community to more efficiently store and share large-scale biomolecular simulation data, potentially enhancing collaborative research efforts. The workflow, termed "compressTraj", is implemented in PyTorch and is publicly available at `https://github.com/SerpentByte/compressTraj` for use, offering a practical solution for managing the growing volumes of data generated in computational biomolecular studies.

* wabdul@tifrh.res.in

† lars.schaefer@ruhr-uni-bochum.de

‡ jmondal@tifrh.res.in

I. INTRODUCTION

Molecular Dynamics (MD) simulations have been extensively used to explore and understand biomolecular interactions and systems[1–7]. All-atom MD simulation trajectories, particularly for biomolecular simulations, can generate enormous amounts of data, ranging from hundreds of gigabytes to tens of terabytes. The storage and sharing of such data are understandably challenging. Since each researcher involved in MD simulations has different goals, perspectives, and understandings of the systems they study, these trajectories are often underutilized regarding the information extracted from their analysis. Effective distribution of these data sets would enable multifaceted exploration, increasing the efficiency of knowledge mining. The biomolecular simulation community has greatly benefited from established databases like PDB and SwissProt. However, a significant gap remains: the lack of a unified database for storing and sharing simulation trajectories. Efforts such as MDDB[8–10], mdCATH[11] and NMRLipids[12] have attempted to fill this void, but the vast size of the data presents a substantial challenge. While platforms like Zenodo, OSF, and Figshare host hundreds to a thousand terabytes of data[13], managing and accessing it is complex and resource-intensive. More importantly due to challenges faced in uploading such huge amounts of data, only a fraction of the global simulation data is actually uploaded. Hence a unified database could revolutionize data accessibility. In the light of these challenges, we here present a simple, quick and almost loss-less protocol to compress simulation trajectories into "latent spaces" via the usage of AutoEncoders, reaching compression levels of up to 98%.

AutoEncoders (AE) are self-supervised neural networks designed to "encode" input data into a "latent" space of reduced dimensionality and then reconstruct the original data from this latent space. Ideally, a perfectly trained AE would produce outputs identical to its inputs. However, in practice, no artificial neural network achieves perfect training, so the objective is to minimize the difference between the inputs and outputs of an AE. Because the outputs are intended to replicate the inputs, the network is considered self-supervised or auto-regressive. This means that the network independently maps the input data to the latent space and then maps the latent space back to the original input data.

AE have been utilized in biomolecular simulations primarily to project high-dimensional data, such as distance matrices, onto lower-dimensional latent spaces[14–16]. However, us-

ing coordinates as inputs to AutoEncoders is less common[17] due to their lack of Special Euclidean Group in 3 dimensions (SE(3)) invariance; a rotation or translation of the entire system results in a different set of coordinates, even though the system's fundamental features remain unchanged. In this study, we directly use coordinates to train AutoEncoders, employing the latent space as the "compressed" trajectory. We demonstrate that the latent spaces represent highly compressed MD trajectories, with decompression to coordinates being almost lossless. Consequently, our protocol enables the creation of highly compressed MD trajectories that can be easily shared and stored and quickly decompressed when needed.

II. METHODS

A. Details on the systems used

We have utilized the following biomolecular systems for this study: chignolin, trpcage, Hen Egg White Lysozyme (HEWL) as representatives of folded proteins of different sizes; Amyloid β40 (Aβ40), and ACtivator for Thyroid hormone and Retinoid receptors (ACTR) as representatives of Intrinscially disordered proteins (IDPs); the L99A mutant of T4 Lysozyme with benzene as the ligand (T4L L99A + benzene) and Cytochrome P450 with camphor (Cyt-P450 + camphor) as representatives of protein-ligand systems.

As shown in Figure 1, these systems cover a diverse spectrum of folded miniproteins/globular proteins (chignolin, trpcage, HEWL), intrinsically disordered proteins (IDP) (Aβ40, ACTR) and protein interacting with ligand (T4L L99A + benzene, Cyt-P450 + camphor). The availability of long time-scale simulation trajectories of these systems (either kindly shared by D. E. Shaw research or generated in-house) prompted their choice in the present investigation. Figure 1 summarizes the details of each system and their respective trajectory details are provided in Table I.

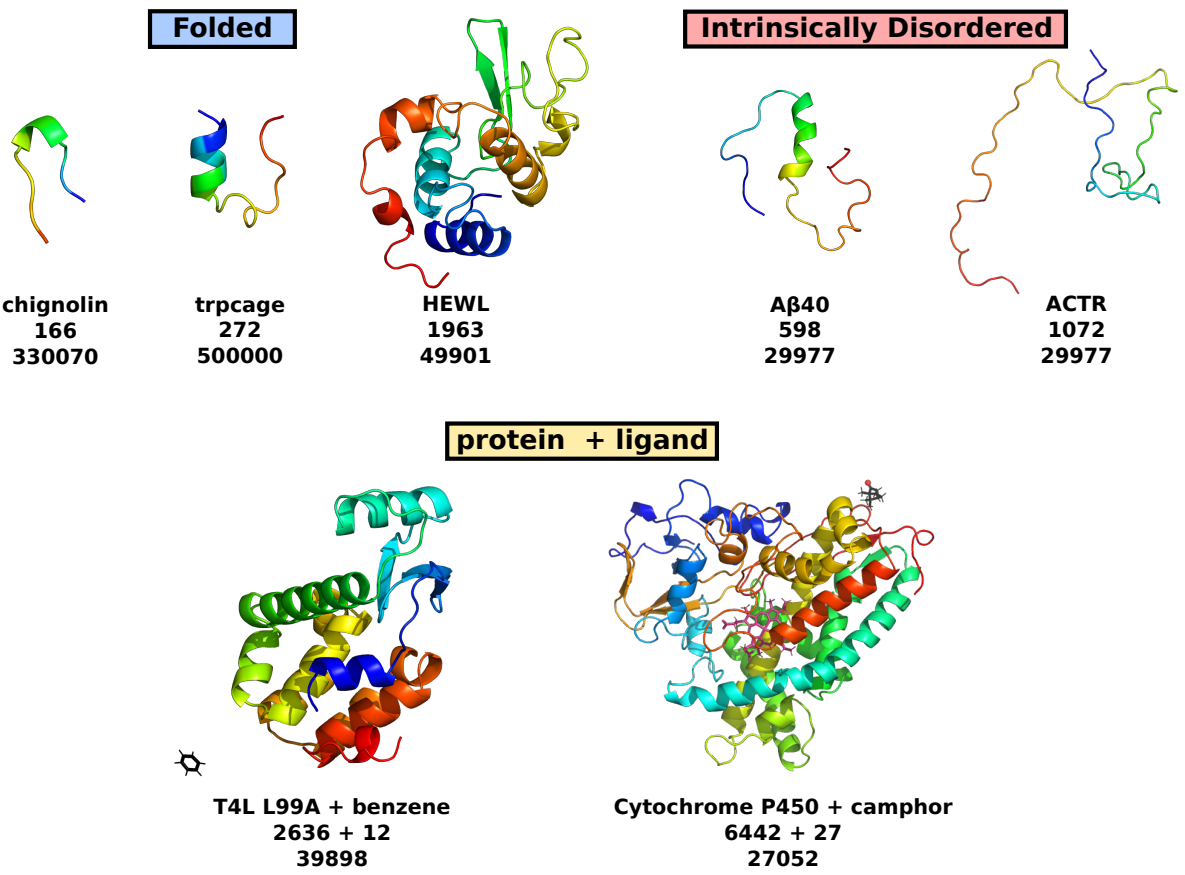

FIG. 1. Representative conformations of the systems used for this study along with the number of atoms present (middle) and respective number of frames present in each trajectory (bottom). The data for folded and intrinsically disordered proteins have been obtain from D.E Shaw Research[18]. The data for T4L L99A + benzene was obtained from [19]. Benzene is shown in black. The data for Cytochrome P450 + camphor was obtained from [7]. Camphor is shown in black with its oxygen being represented as a red sphere.

B. Processing of trajectories and the AutoEncoder architecture

The first step of the protocol is to choose a reference conformation. The particular choice of frame is arbitrary and it doesn't affect the algorithm as discussed below. The trajectory is first aligned with the reference conformation, and for each frame of the aligned trajectory, the center of mass of the system is set to the origin. These transformations help the resulting coordinates to become pseudo SE(3) invariant, meaning that as long as these steps are followed, coordinates can be used as input to an AE.

TABLE I. Details of different trajectories used for this study

system	**number of particles**	**number of frames**	**XTC file size (MB)**	**reference**
chignolin	166	330070	226	[18]
trpcage	272	50000	55	[18]
HEWL	1963	49901	369	[18]
Aβ40	598	29977	70	[18]
ACTR	1072	29977	123	[18]
T4L L99A + benzene	2636 + 12	39898	385	[2]
Cyt-P450 + camphor	6442 + 27	27052	636	[7]

For a system with N particles, the resulting trajectory has $3N$ Cartesian coordinates. For training the AE, we "flatten" the $(N, 3)$ dimensional coordinate space into a vector with $3N$ dimensions. All coordinates are then pooled. Thus, for f frames, the resultant dataset would be $(f, 3N)$. This dataset is then scaled using a Min-Max scaler, with the minimum value set to 0 and the maximum value set to 1. The scaled trajectory is then used as input to the AutoEncoder.

C. The architecture of the AutoEncoder

The architecture of the AutoEncoder used in this work is consistent across all systems studied (Figure 2). Here, we utilize a "dense" AutoEncoder, meaning only fully-connected feed-forward layers are employed. There are two dense layers in the encoder and two layers in the decoder as well. The number of neurons in each layer and the sequence of occurrences of layers can be summarized as: (input, 4096, 1024, L, 1024, 4096, output), where L is the number of latent dimensions, the inputs are original, scaled coordinates and the outputs are reconstructed, scaled coordinates.

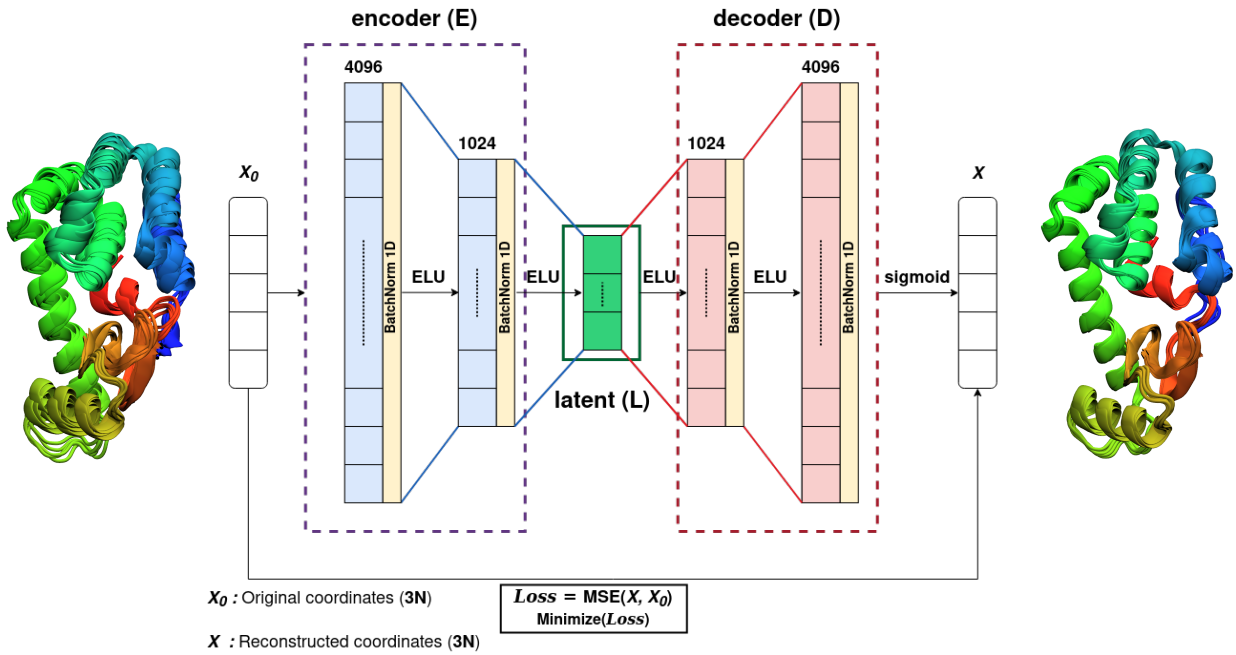

FIG. 2. The architecture of the AutoEncoder used in this study. x_0 and x are the input and the reconstructed input respectively. L is the latent space dimension. BatchNorm 1D is one dimensional batch normalization.

The models were built using PyTorch (see Figure 2 and *Supplementary Methods* for details on the architecture) and trained using PyTorch lightning. We use the exponential linear unit (ELU) [20] activation function for all layers, except for the output layer, where a sigmoid activation function is applied. The models were trained using a mean squared error (MSE) loss function and the AdamW[21] optimizer with the default hyper-parameters (βs=(0.9, 0.999), ϵ=1e-08, λ=0.01). We use a batch size of 256 and each model is trained for 200 epochs. We fix all seeds for each training. We have used a seed value of 42 for training all models.

D. Implementation of the algorithm and usage

To facilitate the preprocessing of trajectories and the training of AutoEncoders, we have developed a Python library named "compressTraj", accessible via GitHub (https://github.com/SerpentByte/compressTraj). The repository includes ready-to-use Python scripts: two for generic applications and two tailored for protein-ligand systems. Each pair

comprises a script to compress the trajectory and another script to decompress the latent space back into a trajectory. These tools are designed for out-of-the-box functionality upon library installation, streamlining the compression and decompression processes for simulation data.

1. Compression

For the script(s) to compress a trajectory, the center of mass of aligned system coordinates are first set to zero, followed by coordinate scaling between (0, 1). For all proteins, including both protein-only and protein-ligand systems, the scaled coordinates are used as inputs to the AutoEncoders, as described earlier in the *Methods* section.

For protein-ligand systems, once the system has been aligned with the reference conformation, the ligand centers of mass are recorded for each frame. The protein and ligand trajectories are then decoupled, resulting in two independent trajectories. The protein part is treated as previously described. For the ligand, the centers of mass are set to zero. No further alignment is performed to preserve ligand orientation. The coordinates are scaled between 0 and 1, then flattened, and used to train another AutoEncoder, which is different from the one trained for the protein part. It should be noted that the default behavior of the code handling the ligand part of protein-ligand systems, is to set the latent space of the ligand's AutoEncoder to $\frac{3N_{lig}}{20}$ (or $c_{lig} = 20$), where N_{lig} represents the number of atoms in the ligand. If $\frac{3N_{lig}}{20}$ is less than 2, the code will automatically set the latent space dimension to 2, ensuring a minimum latent space size for the ligand AutoEncoder. This has been done to emphasize on the accuracy of ligand reconstruction while still compressing the ligand coordinates to a certain extent.

Once the respective AutoEncoder models have been trained, the coordinates are projected onto the latent space using the encoder part of the AutoEncoders. This results in the compressed trajectories.

2. Decompression

The latent space is first expanded to scaled coordinates using the decoder part of the AutoEncoders. The coordinates are then rescaled back to real values and written in standard

molecular dynamics trajectory files, which is the default compressed trajectory file format in GROMACS. By default the codes use XTC files. However it can be changed to any other format readable and writable by the python package MDAnalysis.

For protein-ligand systems, the above steps are first followed individually for the protein and the ligand using their respective AutoEncoders. However before writing the trajectory, the protein and ligand trajectories are recoupled. This is achieved by translating the reconstructed ligand coordinates by its center of mass, pre-recorded for each frame.

This protocol can be easily extended for multi-component systems where the system can be decomposed into components other than the protein and the ligand.

E. List of Software used

We have used only open-source software for this study. Snapshots were generated using PyMOL 2.5.4[22–24] and VMD[25]. Analyses were performed using Python[26] and MDAnalysis[27, 28]. Figures were prepared using Matplotlib[29], Jupyter[30] and Inkscape[31]. All deep neural network models were built using Scikit-learn[32, 33] and PyTorch[34].

III. RESULTS AND DISCUSSION

A. Compressing Hen Egg White Lysozyme trajectory

We started our investigation of implementation of our AutoEncoder-based work-flow on Hen Egg White Lysozyme (HEWL) (PDB ID: 1AKI), a folded protein with 129 residues. Since it is a moderately sized folded protein and has served as a reference system for MD tutorials[35] and benchmark investigations[18], we first apply our algorithm on HEWL to gain a realistic idea on the extent of reconstruction and compression.

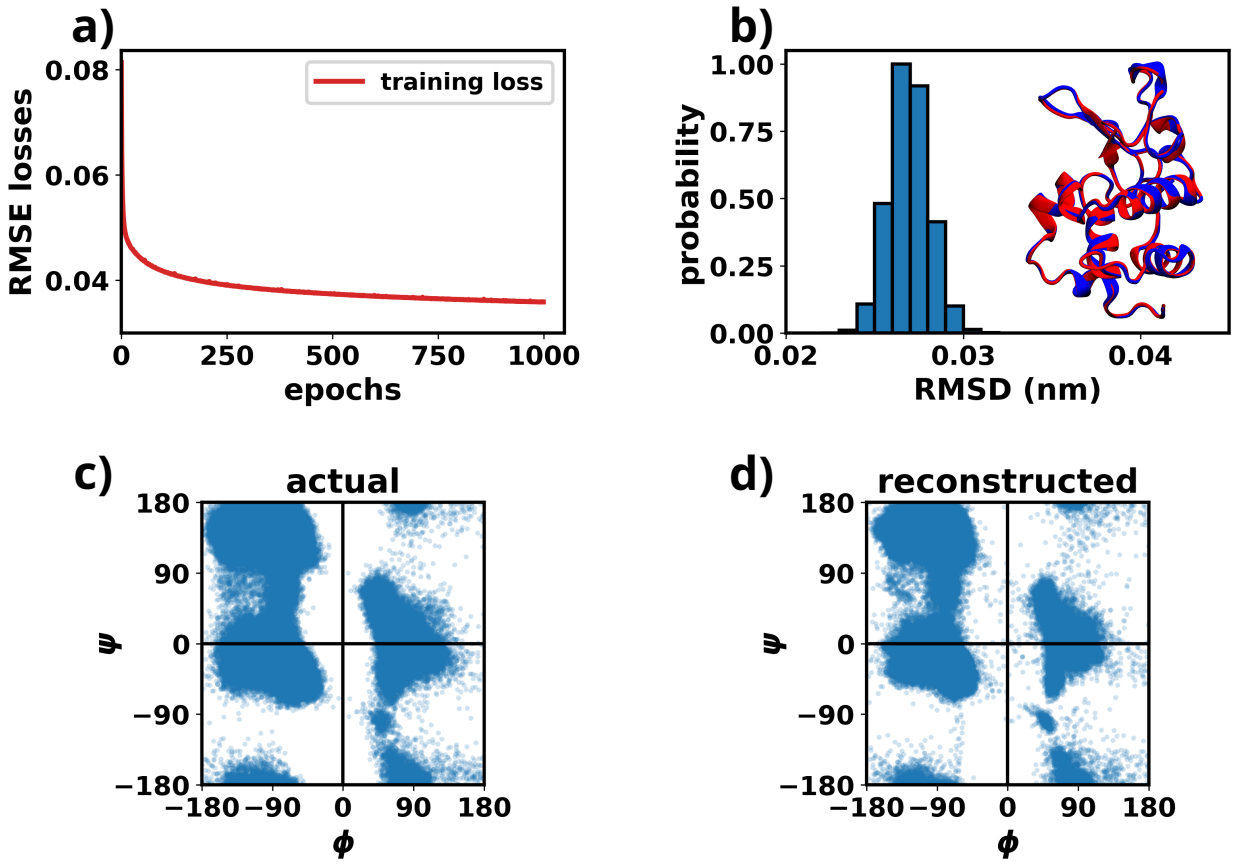

FIG. 3. HEWL compression and reconstruction for $c = 20$. **a)** Training loss for HEWL. **b)** Distribution of root mean squared deviation (RMSD) calculated between actual and reconstructed conformations. The inset shows two superimposed frames obtained from the original and the reconstucted trajectories. **c)** Ramachandran plot obtained from original HEWL trajectory. **d)** Ramachandran plot obtained from reconstructed HEWL trajectory.

We trained the AutoEncoder (AE) on a trajectory of 49901 frames of HEWL which contained the three-dimensional coordinates of all its atoms, including hydrogens. Using an AutoEncoder architecture detailed in the method section (see Figure 2) with coordinates of all the atoms as input features, we train the AE for 1000 epochs with a latent space dimension that is 20 fold smaller than the input dimension ($c = 20$), using root mean squared error (RMSE) between scaled input and output as the loss function. As seen from the training and validation losses in Figure 3a; the training loss drops quickly after which it changes very slowly. From Figure 3b it can be seen that the distribution of RMSDs between actual and reconstructed conformations is very narrow with a mean of 2.6 picometer (pm) and standard

deviation of 0.11 pm. This shows that the reconstruction of the atomic coordinates from the compressed trajectory is very accurate. Finally, to ascertain whether the reconstructed images retain the secondary structure sampled in the actual trajectory, we compare the Ramachandran plots obtained from the original (Figure 3c) and the reconstructed (Figure 3d) trajectories. These analyses show that the secondary structure is also preserved upon reconstruction of the trajectories from the compressed latent space. We also tested the compression and decompression of HEWL using $L = 128$, observing a higher compression albeit with a slight decrease in reconstruction quality (Figure S1a-b). However, the reconstruction remains sufficient to accurately reproduce the Ramachandran plot (Figure S1c-d).

As the most important result of the entire exercise, we obtain a compression of 85-95% in the latent space as compared to the corresponding trajectory in XTC format, which is the highly compressed default trajectory file format in GROMACS. The significantly reduced size of the latent space representation of the simulation trajectories, within the range of a few megabytes, makes it very convenient for facile local and cloud-storage along with electronically sharing with the community for future exploration of the data.

To verify how reconstruction quality varies with the size of the latent space, we trained AutoEncoders with latent spaces of dimensionality 2, 16, 64, 256, and 1024, while keeping the rest of the AutoEncoder architecture constant. A comparison of the RMSDs for AutoEncoders trained with different latent spaces is shown in Figure S2a. The results indicate that as the latent space increases, the reconstruction quality improves. However, for large latent spaces ($\geq$256), particularly for HEWL, the reconstruction quality reaches its upper limit. This trend is expected to be consistent for other systems, with the upper limit of the latent space dimensions, beyond which reconstruction saturates, varying proportionally with system size. Nonetheless the reconstruction across all latent space dimensions tested are accurate up to sub-angstrom level.

To demonstrate the robustness of the protocol concerning the selected reference conformation used to align the trajectories, we trained four models for HEWL, but using different, randomly chosen reference conformations, Figure S2b compares their reconstructions, showing that the training of models with the same architecture for the same system is independent of the choice of reference conformation.

B. The protocol trains well across diverse protein trajectories

To test whether the protocol can be generally applied and automated, we fixed the number of latent dimensions to $\frac{3N}{20}$ (abbreviated as $c = 20$) and standardized the neural network architecture. For each test case, the corresponding network was trained for 1000 epochs. Figures 4a and 4b display the training and validation losses for each test case. From these figures, it is evident that the networks train effectively using the aligned, centered, and scaled coordinates.

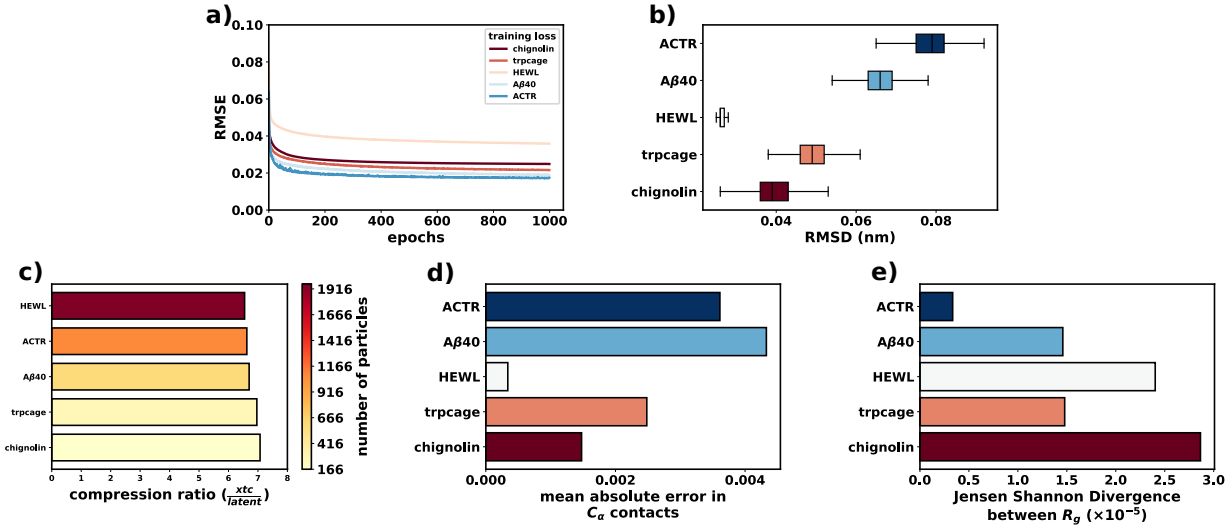

FIG. 4. **a)** Training losses. **b)** Boxplots depicting pair-wise root mean squared deviations (RMSDs). **c)** Extent of compression for $c = 20$. **d)** MAE of differences in residue-residue C_α distances between original and decompressed trajectories. **e)** Jensen Shannon Divergence (JSD) values between R_g distributions obtained from the original and the reconstructed trajectories.

We note that the reconstruction is more accurate for systems with folded proteins compared to intrinsically disordered proteins (IDPs). This difference likely arises because IDPs are complex systems with pronounced conformational fluctuations. This can result in poor aligment of the trajectory with the reference conformation, thereby affecting pseudo-SE(3) invariance. Nonetheless, we found that our protocol can still be effectively used for IDPs but with a slightly higher error in reconstruction. The mean RMSD between actual and reconstructed trajectory here also ranged between 0.01-0.08 nm.

We also calculated the frame-wise RMSD and plotted their distribution using boxplots for each test case in Figure 4b. The low RMSD values indicate that the reconstructed

coordinates closely match the original ones, suggesting near lossless reconstruction. These results demonstrate that the AutoEncoders achieve a high degree of accuracy, ensuring that the coordinates are accurately reconstructed upon decompression.

Figure 4c illustrates the extent of compression for different systems, with the color bar representing the number of particles (N) in each system. We achieve the same extent of compression (7 fold or 85%) across all systems because we use $c = 20$ here.

To measure the reconstruction of secondary structures, we used mean absolute errors (MAEs) between actual and reconstructed pairwise residue-residue C_α contact map (Figure 4d). The very low MAE values ($\sim 10^{-4}$) indicate that the backbone structure is accurately reconstructed during decompression. This is also supported by comparisons of Ramachandran plots between actual and reconstructed trajectories (Figure S3).

Furthermore, the Jensen-Shannon divergence (JSD) between radius of gyration (R_g) distributions of the actual and reconstructed trajectories for a protein was used as a measure of the quality of reconstruction of its spatial extension(s) (Figure 4e). A JSD value of 0 means that two distributions are identical while a value of 1 corresponds to two disjoint probability distributions. Figure 4e shows that most JSD values are in the order of 10^{-5}, demonstrating accurate reconstruction of the spatial extension(s) of the protein.

Finally, our protocol was tested to compress trajectories while maintaining a fixed number of latent dimensions ($L = 128$). We observed proper training of AutoEncoders across all investigated systems for $L = 128$ (Figure S4a). The reconstruction of atomic coordinates across all proteins is also very accurate (Figure S4b). Additionally, using $L = 128$ is very effective in terms of compression for larger proteins, with compression levels reaching upto 50 fold or 98% (Figure S4b). However $c = 20$ results in better reconstruction compared to fixing the latent space at $L = 128$. Consequently, the default value of C in our code is $c = 20$ (see *Methods*). Figures S4d-e and S5 show that the structures of the proteins are well preserved after reconstruction.

C. Extension of the algorithm to protein-ligand systems

In the previous section, we demonstrated how an AutoEncoder can efficiently compress MD trajectories of single proteins and achieve almost lossless reconstruction. However, one widely used application of molecular simulations is to explore biomolecular recognition

processes involving small molecules (ligands). Here, we show that a slightly nuanced protocol can achieve near lossless reconstruction of compressed MD trajectories also for protein-ligand systems. In particular, for protein-ligand systems, instead of training the coordinates of both protein and ligand in a single neural network, the previously described steps are followed individually for the protein and the ligand. Hence two AutoEncoders, one for the protein and one for the ligand, are trained and used to produce two compressed latent spaces. To reconstruct the trajectory, the trained AEs are utilized to decompress the respective protein and ligand trajectories. The decompressed trajectories are then recoupled and combined to generate the final reconstructed trajectory. This is achieved by translating the reconstructed ligand coordinates by its center of mass, pre-recorded for each frame (see *Methods* for details).

We use two systems as test cases: (i) T4L L99A and benzene[19], and (ii) Cytochrome P450 and camphor[7]. In the first case, the system features a buried hydrophobic pocket where the symmetric ligand, benzene, binds via a gate opening event, allowing the hydrophobic ligand to enter and bind to the pocket. In the second case, although cytochrome P450 also involves a buried pocket, there is no gate opening/closing motion. Instead, the ligand (camphor) is asymmetric, making it crucial to check whether the ligand orientations are properly reconstructed. We followed our described protocol to compress the two trajectories and subsequently reconstruct them. For the ligands, latent space dimensions were fixed to 4. For protein, we tested reconstruction accuracy for $c = 20$. The training losses (Figure 5a) and the system's (protein + ligand) pairwise RMSD values (Figure 5b) indicate successful compression and reconstruction of the trajectories.

For $c = 20$ for protein and $L = 4$ for the ligand, a 7 fold compression is obtained for both systems (Figure 5c). The MAE between residue-residue C_α contact probabilities (Figure 5d) for T4L L99A and Cytochrome P450 showcase almost exact reconstruction of the protein secondary structures for all frames.

1. T4L L99A and benzene

Next, we compared the protein dynamics and the ligand binding events between the actual and reconstructed trajectories (Figure 5e-f). The reconstructed trajectory accurately describes the actual ligand binding event demonstrating that the extended compression protocol is effective and accurate for protein-ligand systems.

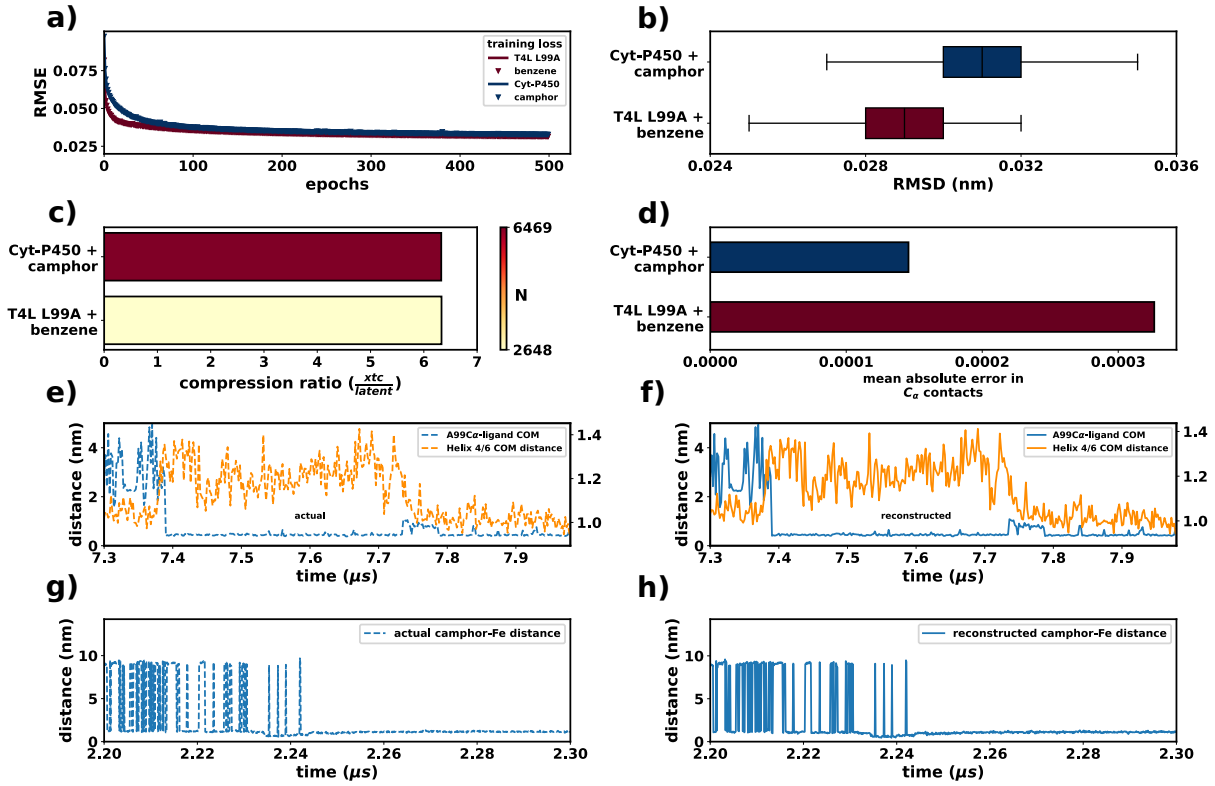

FIG. 5. **a)** Training losses for proteins (solid lines) and ligands (triangle markers). **b)** parwise RMSD distribution for full systems (protein + ligand) where the actual coordinates serve as the reference for the respective reconstructed coordinates. **c)** Compression ratio when $c = 20$. **d)** MAE of residue-residue C_α contact probabilities for the proteins. **e)** The real ligand binding profile for T4L L99A. **f)** The reconstructed ligand binding profile for T4L L99A. **g)** Real binding profile of camphor to Cytochrome P450. **h)** Reconstructed binding profile of camphor to Cytochrome P450.

2. *Cytochrome P450 and camphor*

We next tested accuracy of reconstruction from compressed trajectory for the binding of camphor to Cytochrome P450. Figures 5g-h show accurate reconstruction of the system dynamics. More importantly, the RMSD profile for camphor and the superimposed conformations (Figure S6) show that the coordinates are almost exactly reproduced, indicating that the orientation of the ligand is preserved during reconstruction.

For both the protein-ligand systems, tests were also performed using $L = 128$, where ligand latent space was again fixed at 4. We observed stable training of the models and high accuracy of reconstruction for both the systems as can be seen from Figure S7.

D. Reconstruction quality versus compression

We demonstrated above that the AE-based latent space compression method can compress simulation trajectories by several folds, with consecutive decompression yielding near-exact reconstructions of the original atomic coordinates. Although one can achieve compression levels of up to 85% with reconstruction RMSDs as low as a few picometers, users may still be concerned about potential stereochemistry issues that might not be adequately captured by the RMSD. More importantly, the protocol should be robust enough to allow quasi-exact reconstruction if needed, even if this requires sacrificing some degree of compression.

To demonstrate the robustness of the protocol, we trained 5 AE models on T4L L99A (protein only), each having a different latent space dimension. For each model, we compressed and subsequently decompressed the trajectory. From the resulting trajectories, we randomly selected 5 representative pairs of frames, where each pair consisted of a frame from the original trajectory and its corresponding decompressed frame. This resulted in 25 frame pairs, for which we calculated MolProbity scores [36]. Figure S8 shows the variation of MolProbity scores with latent space size for the reconstructed frames. For $L = 1024$ (60% compression), the score drops below 2.2, indicating that the reconstructed structures are of very high quality. We also observed that the chiral centers of the amino acids, along with other stereochemical features, are preserved. The number of Ramachandran favored and forbidden rotamers in the actual and reconstructed frames were also found to be very similar. A detailed comparison of multiple such metrics, including those discussed above, is summarized in *Supplementary File 1*, where a suffix of 0 indicates values obtained from the original frames.

Finally, going beyond atomic positions, we calculated the force field energies of the original and reconstructed configurations using CHARMM-GUI[37] (*Supplementary File 2*). We observed that, in a few cases, the total potential energy of the reconstructed configuration was high, primarily due to the Lennard-Jones contribution (atomic clashes). As a proof of concept, we selected the reconstructed frame with the highest energy ($L = 2048$, replica=2), solvated it in TIP3P water, and added ions for charge neutralization using CHARMM-GUI. Energy minimization of the system to a maximum force tolerance of 1000 kJ mol^{-1} nm^{-1} was achieved in 803 steepest descent steps. The superimposed conformations (Figure S9)

show that the energy-minimized conformation and the reconstructed conformation are nearly identical, with an RMSD of 0.3 Å. The high potential energy is mainly due to the steeply repulsive (r^{-12}) nature of the Lennard-Jones potential at short interatomic distances and also because of the very stiff bond and angle potentials, so that even very small positional deviations lead to very large increases in energy. Thus, energy minimization should be performed before using the reconstructed configurations as starting points for MD simulations.

IV. CONCLUSIONS

In this study, we demonstrated the effectiveness of AutoEncoders (AEs) in compressing Molecular Dynamics (MD) simulation trajectories. By employing a protocol that involves choosing a reference frame/structure, aligning the trajectory, and scaling the coordinates, we achieved near-lossless compression of atomic coordinates. Our approach used a consistent AutoEncoder architecture, trained for 200 epochs across various test cases, and yielded high accuracy in reconstructing the original data. The compression efficiency was particularly notable for larger systems, with data sizes reduced by orders of magnitude.

We validated our protocol by comparing distributions of radius of gyration (R_g) and observing that the reconstructed data closely matched the actual trajectories, as indicated by low Jensen-Shannon Divergence (JSD) values. Additionally, time profiles for protein-ligand systems showed that ligand binding is accurately retained in the compressed trajectories.

Our protocol enables efficient storage and sharing of MD simulation data, facilitating broader exploration and utilization of existing datasets. The robustness, flexibility and high accuracy of our method makes it a valuable tool for the scientific community, especially for storing and distributing large-scale biomolecular simulations data.

V. CODE AND DATA AVAILABILITY

The codes, along with examples, can be found at https://github.com/SerpentByte/compressTraj. Most of the MD data we have used as test systems have been obtained from D.E. Shaw Research with their permission[18]. The data pertaining to T4L L99A + benzene was obtained in-house [19] and has been shared in the compressed format as an example.

VI. ACKNOWLEDGEMENT

The study was performed while visiting Prof. Dr. Lars Schäfer at Ruhr-University Bochum. This work was supported by the Deutsche Forschungsgemeinschaft (DFG) under Germany's Excellence Strategy - EXC 2033 - 390677874 - RESOLV. We acknowledge the funding by the Alexander von Humboldt-Stiftung through an Experienced Researcher Fellowship that supported J.M.'s visit. We acknowledge support of the Department of Atomic Energy, Government of India, under Project Identification No. RTI 4007.

---

[1] A. D. Jr. MacKerell, D. Bashford, M. Bellott, R. L. Jr. Dunbrack, J. D. Evanseck, M. J. Field, S. Fischer, J. Gao, H. Guo, S. Ha, D. Joseph-McCarthy, L. Kuchnir, K. Kuczera, F. T. K. Lau, C. Mattos, S. Michnick, T. Ngo, D. T. Nguyen, B. Prodhom, W. E. Reiher, B. Roux, M. Schlenkrich, J. C. Smith, R. Stote, J. Straub, M. Watanabe, J. Wiórkiewicz-Kuczera, D. Yin, and M. Karplus. All-atom empirical potential for molecular modeling and dynamics studies of proteins. *The Journal of Physical Chemistry B*, 102(18):3586–3616, 1998. PMID: 24889800.

[2] Navjeet Ahalawat and Jagannath Mondal. Mapping the substrate recognition pathway in cytochrome p450. *Journal of the American Chemical Society*, 140(50):17743–17752, 2018.

[3] Lorenzo Casalino, Zied Gaieb, Jory A Goldsmith, Christy K Hjorth, Abigail C Dommer, Aoife M Harbison, Carl A Fogarty, Emilia P Barros, Bryn C Taylor, Jason S McLellan, et al. Beyond shielding: the roles of glycans in the sars-cov-2 spike protein. *ACS central science*, 6(10):1722–1734, 2020.

[4] Paulo CT Souza, Riccardo Alessandri, Jonathan Barnoud, Sebastian Thallmair, Ignacio Faustino, Fabian Grünewald, Ilias Patmanidis, Haleh Abdizadeh, Bart MH Bruininks, Tsjerk A Wassenaar, et al. Martini 3: a general purpose force field for coarse-grained molecular dynamics. *Nature methods*, 18(4):382–388, 2021.

[5] Jan A Stevens, Fabian Grünewald, PA Marco van Tilburg, Melanie König, Benjamin R Gilbert, Troy A Brier, Zane R Thornburg, Zaida Luthey-Schulten, and Siewert J Marrink. Molecular dynamics simulation of an entire cell. *Frontiers in Chemistry*, 11:1106495, 2023.

[6] Sneha Menon and Jagannath Mondal. Conformational plasticity in α-synuclein and how crowded environment modulates it. *The Journal of Physical Chemistry B*, 127(18):4032–4049, 2023.

[7] Mohammad Sahil, Tejender Singh, Soumya Ghosh, and Jagannath Mondal. 3site multisubstrate-bound state of cytochrome p450cam. *Journal of the American Chemical Society*, 145(43):23488–23502, 2023.

[8] Daniel Beltrán, Adam Hospital, Josep Lluís Gelpí, and Modesto Orozco. A new paradigm for molecular dynamics databases: the COVID-19 database, the legacy of a titanic community effort. *Nucleic Acids Research*, 52(D1):D393–D403, 11 2023.

[9] Rommie Amaro, Johan Åqvist, Ivet Bahar, Federica Battistini, Adam Bellaiche, Daniel Beltran, Philip C Biggin, Massimiliano Bonomi, Gregory R Bowman, Richard Bryce, et al. The need to implement fair principles in biomolecular simulations. *arXiv preprint arXiv:2407.16584*, 2024.

[10] Adam Hospital and Modesto Orozco. Md-data: the legacy of the abc consortium. *Biophysical Reviews*, pages 1–3, 2024.

[11] Antonio Mirarchi, Toni Giorgino, and Gianni De Fabritiis. mdcath: A large-scale md dataset for data-driven computational biophysics. *arXiv preprint arXiv:2407.14794*, 2024.

[12] Anne M Kiirikki, Hanne S Antila, Lara S Bort, Pavel Buslaev, Fernando Favela-Rosales, Tiago Mendes Ferreira, Patrick FJ Fuchs, Rebeca Garcia-Fandino, Ivan Gushchin, Batuhan Kav, et al. Overlay databank unlocks data-driven analyses of biomolecules for all. *Nature Communications*, 15(1):1136, 2024.

[13] Johanna KS Tiemann, Magdalena Szczuka, Lisa Bouarroudj, Mohamed Oussaren, Steven Garcia, Rebecca J Howard, Lucie Delemotte, Erik Lindahl, Marc Baaden, Kresten Lindorff-Larsen, et al. Mdverse: shedding light on the dark matter of molecular dynamics simulations. *bioRxiv*, pages 2023–05, 2023.

[14] Liu Jian-wei, Chi Guang-hui, Liu Ze-yu, Liu Yuan, Li Hai-en, and Luo Xiong-Lin. Predicting protein structural classes with autoencoder neural networks. In *2013 25th Chinese Control and Decision Conference (CCDC)*, pages 1894–1899. IEEE, 2013.

[15] Satyabrata Bandyopadhyay and Jagannath Mondal. A deep autoencoder framework for discovery of metastable ensembles in biomacromolecules. *The Journal of Chemical Physics*, 155(11), 2021.

[16] Sneha Menon, Subinoy Adhikari, and Jagannath Mondal. An integrated machine learning approach delineates entropy-mediated conformational modulation of α-synuclein by small molecule. *bioRxiv*, pages 2024–01, 2024.

[17] Aayush Gupta, Souvik Dey, Alan Hicks, and Huan-Xiang Zhou. Artificial intelligence guided conformational mining of intrinsically disordered proteins. *Communications biology*, 5(1):610, 2022.

[18] Paul Robustelli, Stefano Piana, and David E Shaw. Developing a molecular dynamics force field for both folded and disordered protein states. *Proceedings of the National Academy of Sciences*, 115(21):E4758–E4766, 2018.

[19] Jagannath Mondal, Navjeet Ahalawat, Subhendu Pandit, Lewis E Kay, and Pramodh Vallurupalli. Atomic resolution mechanism of ligand binding to a solvent inaccessible cavity in t4 lysozyme. *PLoS computational biology*, 14(5):e1006180, 2018.

[20] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). *arXiv preprint arXiv:1511.07289*, 2015.

[21] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[22] Schrödinger, LLC. The AxPyMOL molecular graphics plugin for Microsoft PowerPoint, version 1.8. November 2015.

[23] Schrödinger, LLC. The JyMOL molecular graphics development component, version 1.8. November 2015.

[24] Schrödinger, LLC. The PyMOL molecular graphics system, version 1.8. November 2015.

[25] William Humphrey, Andrew Dalke, and Klaus Schulten. VMD – Visual Molecular Dynamics. *Journal of Molecular Graphics*, 14:33–38, 1996.

[26] Guido Van Rossum and Fred L. Drake. *Python 3 Reference Manual*. CreateSpace, Scotts Valley, CA, 2009.

[27] Naveen Michaud-Agrawal, Elizabeth J Denning, Thomas B Woolf, and Oliver Beckstein. Mdanalysis: a toolkit for the analysis of molecular dynamics simulations. *Journal of computational chemistry*, 32(10):2319–2327, 2011.

[28] Richard J Gowers, Max Linke, Jonathan Barnoud, Tyler JE Reddy, Manuel N Melo, Sean L Seyler, Jan Domanski, David L Dotson, Sébastien Buchoux, Ian M Kenney, et al. Mdanalysis: a python package for the rapid analysis of molecular dynamics simulations. In *Proceedings of*

the 15th python in science conference, volume 98, page 105. SciPy Austin, TX, 2016.

[29] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95, 2007.

[30] Thomas Kluyver, Benjamin Ragan-Kelley, Fernando Pérez, Brian Granger, Matthias Bussonnier, Jonathan Frederic, Kyle Kelley, Jessica Hamrick, Jason Grout, Sylvain Corlay, Paul Ivanov, Damián Avila, Safia Abdalla, and Carol Willing. Jupyter notebooks – a publishing format for reproducible computational workflows. In F. Loizides and B. Schmidt, editors, *Positioning and Power in Academic Publishing: Players, Agents and Agendas*, pages 87 – 90. IOS Press, 2016.

[31] Inkscape Project. Inkscape.

[32] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011.

[33] Oliver Kramer and Oliver Kramer. Scikit-learn. *Machine learning for evolution strategies*, pages 45–53, 2016.

[34] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[35] Justin A. Lemkul. Gromacs tutorials, 2024. Accessed: 2024-08-12.

[36] Ian W Davis, Andrew Leaver-Fay, Vincent B Chen, Jeremy N Block, Gary J Kapral, Xueyi Wang, Laura W Murray, W Bryan Arendall III, Jack Snoeyink, Jane S Richardson, et al. Molprobity: all-atom contacts and structure validation for proteins and nucleic acids. *Nucleic acids research*, 35(suppl_2):W375–W383, 2007.

[37] Sunhwan Jo, Taehoon Kim, Vidyashankara G Iyer, and Wonpil Im. Charmm-gui: a web-based graphical user interface for charmm. *Journal of computational chemistry*, 29(11):1859–1865, 2008.