Technical Report: SPIDAL Summer REU 2015: Biomolecular benchmark systems

Ian Kenney* Oliver Beckstein†

September 14, 2015

Analysis of molecular dynamics (MD) trajectories is becoming more and more challenging with simulation times routinely exceeding microseconds (with millions of frames) and increasing in size (with millions of particles). The MDAnalysis library (mdanalysis.org) is a versatile open source package that provides a foundation for analysing trajectories, with a particular emphasis on biomolecular simulations. In order to improve MDAnalysis for large simulations, we developed a set of non-trivial benchmarking systems of variable sizes. We generated a modular library of lipid vesicles with the coarse-grained Dry Martini force field, ranging in size from 10 nm to 30 nm radius. These vesicles can be combined in larger multi-vesicle systems in order to produce a range of benchmark systems with variable particle numbers. The systems will also allow us to address the question under which conditions vesicles fuse. We generated simulation systems with up to six vesicles and 3.5 million particles and performed initial simulations on local computing resources. We benchmarked the performance of MDAnalysis to load and process large systems. Our trajectory data set and benchmarking results will be useful to guide future development of MDAnalysis and its integration with the SPIDAL library.

1 Introduction

Molecular dynamics (MD) simulations have become an important computational tool to study biomolecular systems[1–3], in particular membrane proteins and membrane system[4–7]. Analysis of molecular dynamics (MD) trajectories is becoming more and more challenging with simulation times routinely exceeding microseconds (with millions of frames) and increasing in size (with millions of particles)[8]. The increase in data volume is driven by (1) improvements in hardware (such as GPUs) and algorithms[8–11] (2) use of multi-copy enhanced sampling methods[10,12–15], and (3) new efficient representations of the physical interactions such as coarse-grained models, which allows simulations of larger systems and at longer time steps[16,17].

We have been leading the development of the open source MDAnalysis library (mdanalysis.org)[18]. It provides a Python-based tool kit to access trajectories generated by all major MD packages such as NAMD[19], Gromacs[20], CHARMM[21], Amber[22], LAMMPS[23], DL_Poly[24], Desmond[25], and

*Department of Physics, Arizona State University, Tempe AZ
†Department of Physics and Center for Biological Physics, Arizona State University, Tempe AZ. E-mail: oliver.beckstein@asu.edu

Table 1: Library of coarse-grained vesicles with dipalmitoylphosphatidylcholine (DPPC) lipids and two example benchmark systems with 3 and 6 large 30-nm vesicles.

label	vesicles	radius (nm)	atoms	lipids
R10.0	1	10	84192	7016
R12.5	1	12.5	122208	10184
R15.0	1	15	167352	13946
R17.5	1	17.5	219000	18250
R20.0	1	20	277728	23144
R22.5	1	22.5	343500	28625
R25.0	1	25	416208	34684
R27.5	1	27.5	496044	41337
R30.0	1	30	582984	48582
3R30.0	3	30	1748952	145746
6R30.0	6	30	3497904	291492

HOOMD[26]; in addition, common formats such as the protein databank (PDB) format can be read and written. Although many time-critical parts of MDAnalysis are written in C or Cython, not all parts of the library are yet optimized to handle very large systems. Here we report on the initial development of a set of non-trivial benchmarking systems of variable sizes that are to be used to determine performance bottlenecks in MDAnalysis and pin-point areas on which future improvement efforts should be focused. Instead of generating pure solvent systems that can be easily scaled to any desired particle number we instead looked for less homogeneous and scientifically more interesting systems. We focused on the question how biomolecular membranes (lipid bilayers) fuse and form larger aggregates. In particular we are interested in the process by which multiple vesicles—spheroidal bilayer-enclosed structures (Fig. 1)—aggregate. Understanding the underlying physics is important for biological transport processes in the synapses[27] and the Golgi apparatus[28,29] but might also be of interest for the development of drug delivery vehicles[30].

In typical simulations, a large fraction ($> 50\%$) of the simulated particles and interactions consists only of the solvent. Here we utilize the recently published implicit solvent coarse-grained *Dry Martini* force field[31] to avoid simulation of solvent and focus on the lipids alone. This approach enables us to simulate vesicles of realistic sizes and to include a large number of lipids so that we can test specific lipid analysis algorithms such as *LeafletFinder*[18] on large systems. Furthermore, by combining multiple vesicles of varying sizes we can generate inhomogeneous benchmark systems at arbitrary sizes.

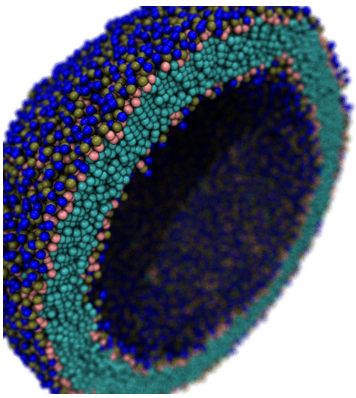


Figure 1: Vesicle with 35 nm diameter (17.5 nm radius) and represented with a coarse grained representation. One half is omitted to show the interior, which is filled with solvent (not shown).

2 Methods

2.1 Vesicle library and benchmark systems

We generated a library of lipid vesicles with the coarse-grained Dry Martini force field and the `martini_vesicle_builder.py` script[31], ranging in size from 10 nm to 30 nm radius (1). Each

vesicle was energy minimized and simulated in the NVT ensemble for 100 ns in order to relax it after model building (see Section 2 for details on the MD simulations).

These vesicles can be combined in larger multi-vesicle systems in order to produce a range of benchmark systems with variable particle numbers and to investigate the question how vesicles fuse. As example systems, we generated simulation systems with 3 and 6 vesicles (each with radius 30 nm), for a total of 1.7 million and almost 3.5 million particles (1).

2.2 MD simulations

MD simulations were performed with Gromacs 5.0.5[20]. Simulation settings were chosen as recommended for the Dry Martini force field[31]. In particular, dynamics were simulated with Langevin dynamics with a time step of 20 fs and a friction coefficient of 4 ps^{-1}. For runs that utilize GPUs in addition to CPUs, parameters were adjusted to be compatible with the specific requirements of Gromacs (namely, the Verlet neighbor search scheme). Simulations were run on local workstations equipped with 16 cores (2 Intel Xeon E5-2670 2.60 GHz and either a single NVIDIA GTX 980 or a dual GTX 690 GPU).

2.3 Benchmarking and Analysis

Analysis of simulations was carried out with Python scripts and Jupyter notebooks[32] based on MDAnalysis 0.8.1[18] together with the *NumPy* and *SciPy*[33] libraries for numerics and *matplotlib*[34] for plotting. Benchmarking was performed with Python scripts on a local workstation (see above). Simulation performance was measured in "ns/d", i.e., simulated time per wall-clock time, where higher is better. For I/O and frame-based analysis operations we measured the wall-clock time required to perform the same operation in many (typically 40) independent repeats and report mean and standard deviation.

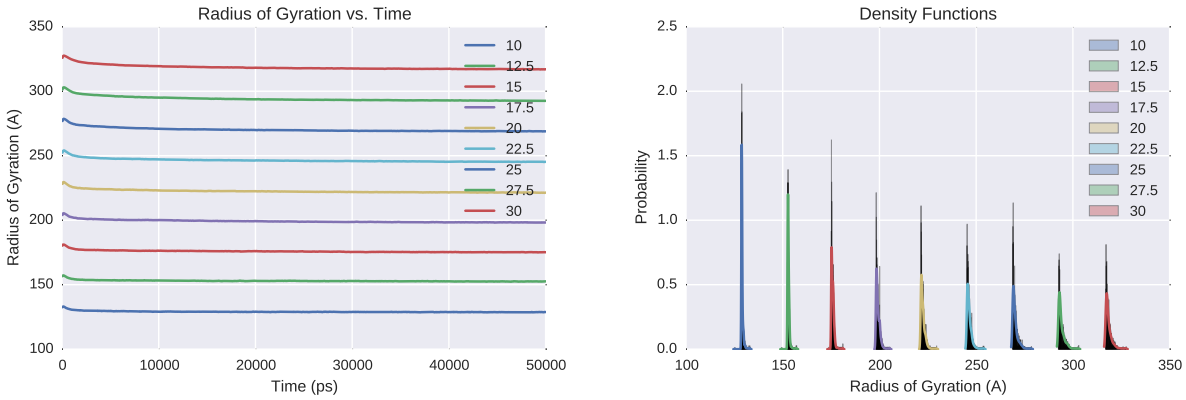
3 Results and Discussion

Our goals were to generate inhomogeneous benchmarking simulations in a modular fashion that will allow us to look at systems of increasing size and to identify bottlenecks in the MDAnalysis library that hinder the processing of large systems. We solved the first problem by developing a *vesicle library* whose members can be combined into larger multi-vesicle systems. We then used the vesicle library to benchmark MDAnalysis trajectory loading and processing operations.

3.1 Vesicle library

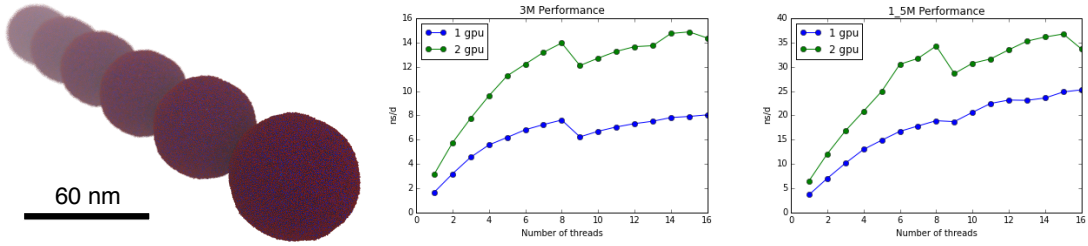
Each single-vesicle system in Table 1 was simulated for 100 ns. Although this is much shorter than the recommended simulation time of 1–10 μs[31], analysis of the radius of gyration of each vesicle indicates that these systems are already rather stable (Fig. 2). These preliminary results indicate that the library of vesicle structures is sufficiently equilibrated to be usable as building blocks for larger simulation systems.

For the construction of two initial multi-vesicle test systems, the largest (radius 30 nm) vesicle was chosen and replicated either three or six times (Fig. 3a) to arrive at systems with either 1.7 million or almost 3.5 million particles (Table 1). The systems were energy minimized and benchmarked on local workstation resources to run at $\sim$35 ns/d or $\sim$15 ns/d when using two GPUs and 16 cores together with 16 additional hyperthreaded cores (Fig. 3). For such large systems with so many lipids (150,000 and 300,000), the performance is rather high if one compares it to an all-atom bilayer system with explicit water and 256 lipids that runs on the same hardware at $<$ 15 ns/d (data not shown).



(a) time series $R_g(t)$ **(b)** probability distribution $P(R_g)$

Figure 2: Radius of gyration R_g of individual vesicles, collected from the first 50 ns of 100 ns MD simulations.



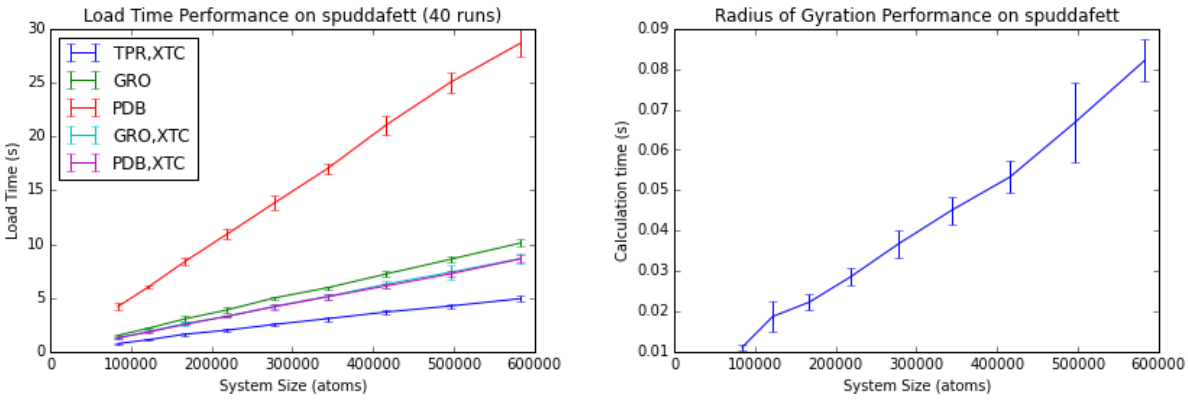
(a) *6R30.0* system **(b)** Performance of *6R30.0* **(c)** Performance of *3R30.0*

Figure 3: Multi-vesicle systems as benchmark simulations. The *6R30.0* system contains six vesicles with radius 30 nm, for a total of about 3.5 million particles. The *3R30.0* system contains three vesicles and about 1.7 million particles. Benchmarks in **b** and **c** were performed on a local workstation with 16 Intel Xeon E5-2670 2.60 GHz cores (which provide an additional 16 hyperthreaded cores) and one NVIDIA GTX 690 GPU (providing two GK104 GPU units). Performance (ns/d) is shown against number of OpenMP threads per GPU, for one (blue) and two (green) GPU units.

Benchmarks on larger machines with higher core numbers and more GPU accelerators will need to be performed in the future to reach a performance of about 250 ns/d so that microsecond-length simulations become feasible, which will be required to address scientific questions beyond benchmarking. Nevertheless, these two simulations demonstrate the feasibility of using the vesicle library to construct larger simulation systems and further work will explore other packing arrangements of vesicles of different sizes. Such longer simulations will lead to vesicle fusion[35] that can then be quantitatively studied as a function of thermodynamic parameters, lipid composition, and vesicle sizes.

3.2 MDAnalysis trajectory processing performance

We benchmarked the performance of MDAnalysis to load and process large systems by loading the simulations in the vesicle library. The first step in any MDAnalysis script is to load a topology file (which contains a list of particles and possibly additional static data such as bonds or partial charges) and a trajectory file. The trajectory contains the coordinates, which change for each time



(a) Initial loading time

(b) R_g analysis (average time per frame)

Figure 4: Performance of the MDAnalysis library for **a** initial loading of a trajectory (Gromacs XTC file) together with different options for a topology file and **b** average time to calculate the radius of gyration for all atoms for one frame in the trajectory (averaged over all frames). All benchmarks were performed in serial on a single Intel Xeon E5-2670 2.60 GHz core.

step. In MDAnalysis, the `Universe` object ties the topology and the trajectory together and part of the process of instantiating `Universe(`*`topology, trajectory`*`)` is to parse these files. MDAnalysis contains custom parsers for a wide range of topology and trajectory files. The benchmark of loading a `Universe` with different topology files (and typically a Gromacs XTC trajectory) showed that the load time increases linearly with the input size (Fig. 4a). Loading from a Gromacs binary run input file (TPR) together with the XTC appears to be fastest ($<$1 s up to 5 s) for all problem sizes up to the investigated maximum of about 600,000 particles. Although not immediately relevant for trajectory processing, loading only from a PDB file (which contains topology information and coordinates) was much worse (5 s up to 30 s) than any other method. This benchmark indicates the PDB reader as a prime candidate for further performance improvements and provides a baseline for increasing the reading speed of the TPR reader further to facilitate rapid reading of larger systems.

As a first simple case for benchmarking analysis we measured the time to calculate the radius of gyration

$$R_g(t) = \sqrt{\frac{1}{M}\sum_{i=1}^{N} m_i(\mathbf{r}_i(t) - \mathbf{R}(t))}, \tag{1}$$

where M is the total mass, $\mathbf{R}(t)$ the center of mass, and the sum runs over N particles with mass m_i and position $\mathbf{r}_i(t)$ each at time frame t. The calculation of R_g included all particles in the simulation. The average wall-clock time to analyze a single trajectory frame increased linearly with the number of particles, starting with about 10 ms for $\sim$100,000 particles and increasing to $\sim$93 ms for 600,000 particles. For long trajectories (order of 1 million frames), analysis times of 100 ms per frame are not acceptable because serial analysis would take about 28 h. Thus we need to improve the per-frame performance. We need to better understand if the performance is limited by loading the trajectory data from disk or by the speed of calculating Eq. 1. We will also need to investigate approaches to parallelize the workload when possible, e.g., by a simple map-reduce algorithm over blocks of the trajectory.

4 Conclusions

Our trajectory data set and benchmarking results will be useful to guide future development of MDAnalysis and its integration with the SPIDAL library.

Acknowledgements Funding was provided by the National Science Foundation for a REU supplement to award ACI-1443054.

References

1. Dror R O, Dirks R M, Grossman J P, Xu H and Shaw D E, 2012 Biomolecular simulation: a computational microscope for molecular biology. *Annu Rev Biophys* **41** 429–52.
2. Orozco M, 2014 A theoretical view of protein dynamics. *Chem. Soc. Rev.* **43** 5051–5066.
3. Perilla J R, Goh B C, Cassidy C K, Liu B, Bernardi R C, Rudack T, Yu H, Wu Z and Schulten K, 2015 Molecular dynamics simulations of large macromolecular complexes. *Current Opinion in Structural Biology* **31** 64 – 74.
4. Marrink S J, de Vries A H and Tieleman D P, 2009 Lipids on the move: Computer simulations of bilayer deformations. *Biochimica et Biophysica Acta (BBA) - Biomembranes* **1788** 149–168.
5. Stansfeld P J and Sansom M S P, 2011 Molecular simulation approaches to membrane proteins. *Structure* **19** 1562–72.
6. Koehler Leman J, Ulmschneider M B and Gray J J, 2014 Computational modeling of membrane proteins. *Proteins* **83** 1–24.
7. Li J, Wen P C, Moradi M and Tajkhorshid E, 2015 Computational characterization of structural dynamics underlying function in active membrane transporters. *Current Opinion in Structural Biology* **31** 96 – 105.
8. Cheatham T and Roe D, 2015 The impact of heterogeneous computing on workflows for biomolecular simulation and analysis. *Computing in Science Engineering* **17** 30–39.
9. Shaw D E, Dror R O, Salmon J K, Grossman J P, Mackenzie K M, Bank J A, Young C, Deneroff M M, Batson B, Bowers K J, Chow E, Eastwood M P, Ierardi D J, Klepeis J L, Kuskin J S, Larson R H, Lindorff-Larsen K, Maragakis P, Moraes M A, Piana S, Shan Y and Towles B, 2009 Millisecond-scale molecular dynamics simulations on anton, in *SC '09: Proceedings of the Conference on High Performance Computing Networking, Storage and Analysis*, 1–11 (ACM, New York, NY, USA).
10. Pierce L C, Salomon-Ferrer R, Augusto F. de Oliveira C, McCammon J A and Walker R C, 2012 Routine access to millisecond time scale events with accelerated molecular dynamics. *Journal of Chemical Theory and Computation* **8** 2997–3002.
11. Salomon-Ferrer R, Götz A W, Poole D, Le Grand S and Walker R C, 2013 Routine microsecond molecular dynamics simulations with amber on gpus. 2. explicit solvent particle mesh ewald. *Journal of Chemical Theory and Computation* **9** 3878–3888.
12. Zwier M C and Chong L T, 2010 Reaching biological timescales with all-atom molecular dynamics simulations. *Curr Opin Pharmacol* **10** 745–52.
13. Mitsutake A, Mori Y and Okamoto Y, 2013 Enhanced sampling algorithms. *Methods Mol Biol* **924** 153–95.
14. Seyler S L and Beckstein O, 2014 Sampling of large conformational transitions: Adenylate kinase as a testing ground. *Molec. Simul.* **40** 855–877.
15. Bernardi R C, Melo M C R and Schulten K, 2015 Enhanced sampling techniques in molecular dynamics simulations of biological systems. *Biochim Biophys Acta* **1850** 872–877.
16. Marrink S J and Tieleman D P, 2013 Perspective on the martini model. *Chem Soc Rev* **42** 6801–22.
17. Noid W G, 2013 Perspective: Coarse-grained models for biomolecular systems. *J Chem Phys* **139** 090901.
18. Michaud-Agrawal N, Denning E J, Woolf T B and Beckstein O, 2011 MDAnalysis: A toolkit for the analysis of molecular dynamics simulations. *J Comp Chem* **32** 2319–2327.
19. Nelson M, Humphrey W, Gursoy A, Dalke A, Kale L, Skeel R and Schulten K, 1996 Namd: A parallel, object oriented molecular dynamics program. *International Journal of Supercomputer Applications and High Performance Computing* **10** 251–268.
20. Pronk S, Páll S, Schulz R, Larsson P, Bjelkmar P, Apostolov R, Shirts M R, Smith J C, Kasson P M, van der Spoel D, Hess B and Lindahl E, 2013 GROMACS 4.5: a high-throughput and highly parallel open source molecular simulation toolkit. *Bioinformatics* **29** 845–54.
21. Brooks B R, Brooks III. C L, Mackerell A D J, Nilsson L, Petrella R J, Roux B, Won Y, Archontis G, Bartels C, Boresch S, Caflisch A, Caves L, Cui Q, Dinner A R, Feig M, Fischer S, Gao J, Hodoscek M, Im W, Kuczera K, Lazaridis T, Ma J, Ovchinnikov V, Paci E, Pastor R W, Post C B, Pu J Z, Schaefer M, Tidor B, Venable R M, Woodcock H L, Wu X, Yang W, York D M and Karplus M, 2009 CHARMM: the biomolecular simulation program. *J. Comp. Chem.* **30** 1545–1614.
22. Case D A, Cheatham 3rd T E, Darden T, Gohlke H, Luo R, Merz Jr K M, Onufriev A, Simmerling C, Wang B and Woods R J, 2005 The amber biomolec-

ular simulation programs. *J Comput Chem* **26** 1668–1688.
23. Plimpton S, 1995 Fast parallel algorithms for short-range molecular dynamics. *J. Comp. Phys.* **117** 1–19.
24. Todorov I T, Smith W, Trachenko K and Dove M T, 2006 DL_POLY_3: new dimensions in molecular dynamics simulations via massive parallelism. *Journal of Materials Chemistry* **16** 1911–1918.
25. Bowers K, Chow E, Xu H, Dror R, Eastwood M, Gregersen B, Klepeis J, Kolossvary I, Moraes M, Sacerdoti F, Salmon J, Shan Y and Shaw D, 2006 Scalable algorithms for molecular dynamics simulations on commodity clusters, in *SC 2006 Conference, Proceedings of the ACM/IEEE*, 43–43.
26. Glaser J, Nguyen T D, Anderson J A, Lui P, Spiga F, Millan J A, Morse D C and Glotzer S C, 2015 Strong scaling of general-purpose molecular dynamics simulations on gpus. *Computer Physics Communications* **192** 97–107.
27. Sudhof T C, 2004 The synaptic vesicle cycle. *Annu Rev Neurosci* **27** 509–47.
28. Bonifacino J S and Glick B S, 2004 The mechanisms of vesicle budding and fusion. *Cell* **116** 153–66.
29. Raposo G and Stoorvogel W, 2013 Extracellular vesicles: exosomes, microvesicles, and friends. *J Cell Biol* **200** 373–83.
30. Bareford L M and Swaan P W, 2007 Endocytic mechanisms for targeted drug delivery. *Adv Drug Deliv Rev* **59** 748–58.
31. Arnarez C, Uusitalo J J, Masman M F, Ingólfsson H I, de Jong D H, Melo M N, Periole X, de Vries A H and Marrink S J, 2015 Dry Martini, a coarse-grained force field for lipid membrane simulations with implicit solvent. *Journal of Chemical Theory and Computation* **11** 260–275.
32. Pérez F and Granger B E, 2007 IPython: a system for interactive scientific computing. *Computing in Science and Engineering* **9** 21–29.
33. Jones E, Oliphant T, Peterson P *et al.*, 2001–, SciPy: Open source scientific tools for Python, [Online; accessed 2014-07-29].
34. Hunter J D, 2007 Matplotlib: A 2d graphics environment. *Computing In Science & Engineering* **9** 90–95.
35. Baoukina S and Tieleman D P, 2010 Direct simulation of protein-mediated vesicle fusion: lung surfactant protein b. *Biophys J* **99** 2134–42.