Chapter 18

Large-Scale Molecular Dynamics Simulations of Cellular Compartments

Eric Wilson, John Vant, Jacob Layton, Ryan Boyd, Hyungro Lee, Matteo Turilli, Benjamín Hernández, Sean Wilkinson, Shantenu Jha, Chitrak Gupta, Daipayan Sarkar, and Abhishek Singharoy

Abstract

Molecular dynamics or MD simulation is gradually maturing into a tool for constructing in vivo models of living cells in atomistic details. The feasibility of such models is bolstered by integrating the simulations with data from microscopic, tomographic and spectroscopic experiments on exascale supercomputers, facilitated by the use of deep learning technologies. Over time, MD simulation has evolved from tens of thousands of atoms to over 100 million atoms comprising an entire cell organelle, a photosynthetic chromatophore vesicle from a purple bacterium. In this chapter, we present a step-by-step outline for preparing, executing and analyzing such large-scale MD simulations of biological systems that are essential to life processes. All scripts are provided via GitHub.

Key words Multiscale simulation, Molecular dynamics, Photosynthetic chromatophore, NAMD, VMD, Ensemble toolkit, High-performance computing

1 Introduction

Living cells are composed of hundreds of macromolecular complexes (proteins and nucleic acids) carrying out their assigned biological functions. These complexes are housed in key cellular compartments, such as the nucleus, mitochondrion, endoplasmic reticulum, and Golgi apparatus. These cell organelles form a detailed network to perform physicochemical reactions, which give rise to remarkable biological phenotypes such as growth, adaptation to environmental changes, and coaccommodation of competing functions [1, 2].

Eric Wilson and John Vant contributed equally to this work.

Ingeborg Schmidt-Krey and James C. Gumbart (eds.), *Structure and Function of Membrane Proteins*, Methods in Molecular Biology, vol. 2302, https://doi.org/10.1007/978-1-0716-1394-8_18,

Molecular simulations at atomistic resolution offer unprecedented detail of protein structure and functions [3]. Fulfilling this promise, the last decade has seen significant advancements in the development of multiphysics algorithms [4, 5] and their Graphical Processor Unit (GPU)–accelerated computational implementations [6]. Augmented by the deployment of peta- to exascale supercomputers application [7], these technologies are harnessed to estimate free-energy changes of dynamic biomolecules and ligands using "computational calorimetry" [8] in protein and ribonucleic acid (RNA) sequence analysis beginning the so-called big-data revolution in Biology and Chemistry [9], and to perform molecular simulations of living meso- to macroscopic systems (popularly called large systems) in chemical detail [10]. Here, we focus on how to perform molecular dynamics (MD) simulations of large biological systems to formulate a hands-on guide for modeling cell-scale objects (Fig. 1).

Notable large-scale all-atom simulations in recent years include the work on protein–protein interactions inside the bacterial cytoplasm characterized by a dense packing of many different macromolecules. This resulted in the generation of a cytoplasmic model comprising of 103-M atoms in a cubic box of 100 nm [11] and two different subsections of *Mycoplasma genitalium* (MG), MG_{m1} and MG_{m2}, with 12-M atoms were simulated [12]. Similar efforts have been taken to study viruses at unprecedented detail by Perilla and Schulten, simulating the Human immunodeficiency virus type

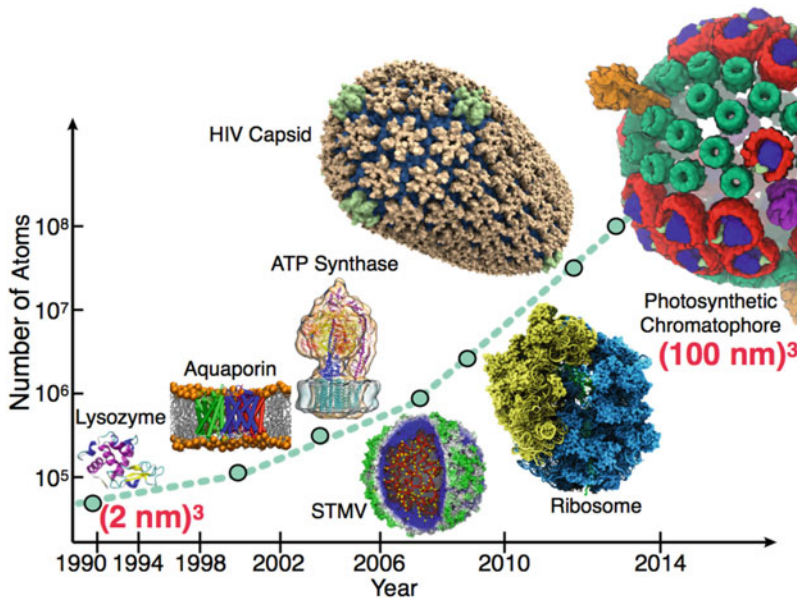

Fig. 1 Historical timeline of molecular dynamics simulations of biological systems. The system size increases from 1000 atoms in a lysozyme molecule (≈2 nm^3 system volume) to over 100 M atoms in the photosynthetic chromatophore (≈100 nm^3 system volume)

1 (HIV-1) capsid. The capsid is a large container, made of ≈1300 proteins with altogether four million atoms. Simulations of over 64-M atoms for over 1 μs allowed for a detailed study of the chemical–physical properties of an empty HIV-1 capsid, including its electrostatics, vibrational and acoustic properties, and the effects of solvent (ions and water) on the capsid. The simulations reveal critical details about the capsid with implications to biological function [13]. Other exemplary targets of large-system simulation include the ribosomal translocation machinery [14, 15], the protein degradation pathway in proteasomes [16], enveloped viruses [17] and the periodic table of nonenveloped viruses [18], membrane-bending proteins [19, 20], and folding patterns of deoxyribonucleic acid (DNA) [21]. Most of these simulations are driven by ground-breaking advances in microscopic and imaging experiments [22].

Recently, we have reported a half microsecond simulation of a 136-M atom-scale model of an entire photosynthetic organelle, a chromatophore membrane vesicle from a purple bacterium, which revealed the rate-determining steps of membrane-mediated energy conversion [2]. Bioenergetic membranes are the key cellular structures responsible for coupled energy-conversion processes, which supply adenosine triphosphate (ATP) and important metabolites to the cell. A physical model of the emergence of phenotypic properties from detailed atomistic interactions is expected to offer direct insights on the rules of life [23]. Initial efforts to provide the structural and functional model of the bacterial photosynthetic membrane vesicle were undertaken by Sener et al. [24] and by Goh et al. [3]. Later, Singharoy et al. performed atomic-level investigations of cellular processes that have thus far been impeded by the sheer complexity of the network of interactions, the timescales of a cell cycle, and the lack of essential experiment-inferred information. Molecular dynamics simulations of this bioenergetic organelle elucidate how the network of bioenergetic proteins influences membrane curvature and demonstrates the impact of thermal disorder on photosynthetic excitation transfer.

In subsequent sections, we provide a guide to large scale all-atom simulations, software required and description of input parameters. We then present a case study discussing how chromatophore converts sunlight into chemical energy, a process that is essential to life [25]. The necessary scripts and files to build the system and perform simulations are freely available to users on GitHub [26]. Note, while preparing this chapter it is assumed that the user feels comfortable using the UNIX command line and has basic experience with molecular dynamics using NAMD and scripting in VMD.

2 Materials

This section describes the input files and the software required to perform MD simulations on large systems (>100-M atoms). All simulations are performed using the popular MD tool called Nanoscale Molecular Dynamics or NAMD [27].

2.1 Input Files

Large-scale MD simulations with NAMD require the following four types of input files.

2.1.1 PDB

The starting structural coordinates of the given protein are provided by the Protein Data Bank (PDB) file. PDB files for the protein of interest can be accessed and downloaded from the PDB database https://www.rcsb.org.

2.1.2 JS Files

.js files are a binary file format which is more efficient than typical protein structure files (PSF) for large systems. Files of other formats can be converted to the .js format using the script found here: https://www.ks.uiuc.edu/Research/vmd/minitutorials/largesystems/

2.1.3 Force Field Parameters

Force field files contain numerical parameters (masses, charges and spring constants) needed to set up a potential energy function for evaluating how the atoms from different chemical constituents move due to bonded, angular, dihedral and nonbonded (electrostatic or van der Waals) interactions. These files typically consist of a topology file and a parameter file which has been generated either by spectroscopic data, quantum mechanical calculations, or through comparison of atoms with novel connectivity to known parameters. Topology files contain information about atom names, bond angles, and charges and end in the .inp file suffix for the CHARMM series of force fields. Parameter files contain constants required for an energy function, namely bond angle force constants and equilibrium angles bond and angle constants, and typically ends in the .par file suffix for the CHARMM simulations. More information on these files can be found at https://www.ks.uiuc.edu/Training/Tutorials/science/forcefieldtutorial/forcefield-html/node6.html or https://www.ks.uiuc.edu/Training/Tutorials/ namd/namd-tutorial-unix-html/node25.html. Generation of a parameter file for a novel chemical constituent can be done with the CGenFF server (https://cgenff.umaryland.edu).

2.1.4 Gridforce Files

Since the construction of large molecular models involve iterative building and reequilibration of the simulation system, knowledge-based grid potentials will be employed as a constraint to maintain the stability of the intermediate models, while concomitantly maintaining the constant volume (NVT) or constant pressure (NPT)

conditions. The Gridforce simulations require a configuration file with the parameters of the potential represented as 3-dimensional volumetric data in .dx format, and also point to a PDB file which specifies which atoms will have forces applied to them. These grid potential or .dx files are also employed as input to Brownian Dynamics simulations of protein diffusion in coarse-grained computational tools such as ARBD [28]. A more in depth review of this method can be found at https://www.ks.uiuc.edu/Training/Tutorials/science/forces/forces-tutorial-html/node4.html/.

2.2 Software

The user should have access to the following software tools to run a large system simulation.

2.2.1 VMD

Visual Molecular Dynamics (VMD) is a computer program, designed for molecular modelling and visualization of biological systems [29]. It can be downloaded from http://www.ks.uiuc.edu/Research/vmd/.

2.2.2 NAMD

NAMD can be used to simulate large systems consisting of millions of atoms and is noted for its parallel efficiency, scaling to thousands of nodes (CPUs or GPUs). The software is widely used with CHARMM force fields but is also compatible with other popular force fields like AMBER and OPLS. It can be downloaded from http://www.ks.uiuc.edu/Research/namd/.

2.2.3 Charm++

Charm++ is a C++ based system for efficient parallelization of computational tasks. It was designed with efficient portability and latency tolerance in mind, being widely used by scientific and engineering applications for efficient computation across the entire range of hardware, from local computer clusters to large supercomputers. It can be downloaded as a binary and installed from http://charm.cs.illinois.edu/software.

2.2.4 APBS

Adaptive Poisson–Boltzmann Solver (APBS) numerically solves the Poisson–Boltzmann equation, which is a continuum model to describes the electrostatic interactions of molecular solutes in ionized solutions. APBS allows for the efficient evaluation of electrostatic properties of a wide range of length scales, including systems with millions of atoms. It can be downloaded from https://sourceforge.net/projects/apbs/. When working with large systems it can be useful to install an additional component called PDB2PQR. This software converts PDB files to PQR files which are read by the electrostatics software. PDB2PQR can be downloaded as a binary or compiled from source at https://apbs-pdb2pqr.readthedocs.io/en/latest/pdb2pqr/index.html.

2.2.5 ARBD

MD can be impractically expensive for monitoring the diffusive processes. ARBD or Atomic Resolution Brownian Dynamics simulations overcome this limitation of MD by approximating the proteins as rigid entities and capturing their diffusive dynamics within an implicit solvent environment assuming the so-called friction-dominated regime [30]. The potential of mean force required to drive Brownian dynamics is often derived from the average electrostatic potentials computed by APBS. ARBD supports point-like and grid-specified particles for representing the proteins. The performance is further enhanced by compatibility with GPUs.

All software tools used for performing large system simulations are free for academic use and binary executable are available for common operating systems and machine types.

2.3 Hardware

Many considerations need to be made when optimizing a simulation setup for a specific supercomputing architecture. Despite recent advancements in simulation software such as in NAMD 2.13, which redistributes computational workloads and reduces communication overhead, many bottlenecks remain a limiting factor. In today's biomolecular MD simulation literature, one typically reports production level runs of timescales on the order of microseconds. In order to achieve these timescales in a reasonable amount of time the software/hardware should produce 10–100 ns per day. The number of compute nodes needed to achieve the aforementioned performance varies with the computing architecture. Therefore, we provide a short description of the hardware with benchmarking from some example systems and our case study. It is important to note that before beginning production level runs for large system simulations, one should always thoroughly benchmark their system against the available computing architecture to ensure the most efficient use of resources.

2.3.1 Example of Applicable Hardware

Using the DOE's Summit supercomputer operated by the Oak Ridge Leadership Computing Facility (OLCF), a 21 M-atom system running on 64 compute nodes achieved a performance of 22 ns/day [31]. Each compute node of the IBM AC922 Summit system contains two 22 core-POWER9 CPUs and 6 NVIDIA Volta V100 GPUs. This benchmarking represents the state of the art for supercomputers.

2.3.2 Hardware Used for Case Study

The chromatophore model is comprised of 136 M-atoms. The MD software NAMD 2.12 was used for all equilibration and production runs. The production level runs of this model were conducted on the DOE's Titan supercomputer. Each compute node of the now decommissioned Cray XK7 Titan system had a 16 core AMD Opteron processor and an NVIDIA Kepler K20 GPU. Using a 1 femtosecond timestep and 4096 nodes, Titan was able to produce 16 ns of MD trajectory per day. Simulation parameters can be found at [26].

3 Methods

This section provides a general workflow for performing large-scale MD simulations of protein–membrane systems. A number of major simulation checkpoints will be covered, which includes model building, equilibration, analysis, and automation. Tcl scripts pertaining to the execution of these checkpoints in NAMD and VMD are provided.

3.1 Model Building

The first step for performing an MD simulation is to compose the system by assembling the models of individual components derived from X-ray crystallography, cryo-EM, homology and ab initio computations. In what follows, we outline how to construct a large system using VMD and Tcl scripts provided at [26]. First, the proteins are arranged based on prior information of their quaternary interactions. Second, these proteins are embedded within a membrane bilayer of biologically relevant shape and composition.

Third, the protein–membrane system is solvated and ionized. For the large systems, the solvation box is often truncated to noncubic symmetries. This setup reduces the number of atoms without compromising the protein–protein separation need to impose periodic boundary conditions [32]. Finally, the solvated system is equilibrated in a series of iterative steps.

3.1.1 Arrange Proteins

Relative position and orientation of the individual protein models within larger assemblies are determined from low-resolution imaging or microscopic experiments. For example, inter-protein distances are determined from atomic force microscopy (AFM) data using the so-called inverse-Mollweide transformation [33]. Similarly, the overall shape of the assembly is enforced on the model by employing restraints from electron density maps via the molecular dynamics flexible fitting scheme in NAMD [3]. The relative stoichiometry and copy number of the proteins, as well as the protein–lipid ratio can be determined from mass spectrometry [34]. These constraints originating from the protein–lipid ratio are nonunique [10]. The underlying large scale structural data involves quantifiable uncertainties in terms of distance (interprotein separation in AFM is prone to error of 10–15 Å [35, 36]), geometry (cryo-EM and tomography data ranges in resolution between 3 and 20 Å [22]) or mass measures (The error bounds on quantitative mass spectrometry is 0.1–0.7 mole, implying that the mass of the proteins and lipids were determined to within a 10–20% uncertainty [34]). Consequently, an ensemble of models should be constructed to fit to the same data set. Although every problem requires customization, tools such as IMP [37], multiple instancing and X-MAS builder modules on VMD [29], and symmetry builder in ViperDB [38] are broadly applicable for assembling the proteins.

3.1.2 Build Membrane and Generate Connectivity File

The scripts in the GitHub repository [26] provide tools for building four different types of membrane systems. The script GeneralTools.tcl defines four commands for producing a bilayer, vesicle, micelle or a curved membrane. These commands take the desired composition of the membrane as input and require a PDB file for each lipid type in your membrane. An example where the generate membrane commands gets called is in the wrapper 01-build-30 nm-vesicle.tcl. Once the membrane is built, the structural and connectivity information needs to be stored in a memory optimized file format (i.e., the .js file format). This is accomplished by sourcing the following script 02-write-js-file.tcl. The output of this step is a .js file which contains all the structural and connectivity information of the lipids in the membrane.

3.1.3 Create and Crop a Solvation Box

Once the membrane is built, it needs to be explicitly solvated. First, a large water box is created by patching together smaller equilibrated water boxes using the script 03-make-giant-waterbox.tcl. It is generally necessary to crop large water boxes in order to minimize the number of water molecules needed to simulate the system. In the case of a vesicle, the large water box can be cropped into a hexagonal water box with the script 04-crop-waterbox-for-30 nm-vesicle.tcl. The resulting water box needs to be renamed so that each water molecule has a unique identifier. The script 05-renumber-hexwaterbox.tcl renames the hexagonal water box made in the previous step. In order to efficiently solvate the membrane, the water box needs to be rotated to maximize the distance between the membrane and the edge of the water box. This can be accomplished with the script 06-rotate-waterbox.tcl. The resulting water box and lipids are then combined using 07-combine-waterbox-with-lipids.tcl. Finally, waters which clash with lipids are removed with 08-remove-clashing-waters.tcl, and the system is ionized with 09-ionize-example-part1.tcl.

3.1.4 Merge Membrane Water Box Assembly with Proteins

Once the membrane system is ionized and solvated, proteins are embedded into the membrane. First, a fixed atom file is generated for subsequent MD simulations by sourcing the script 10-makeFixedAtomsFile.tcl. The biomolecular scaffold .js file generated in the assemble proteins step is then merged with the solvated and ionized lipids .js file with the script 20-merge-with-TEMP-proteins.tcl. Next, the lipids, water, and ions that clash with the inserted proteins are removed with 30-combine-lipidsphereand-proteins-pack5.tcl and 32-carvelipids-SASA-pack5.tcl. A solvent accessible surface area calculation can be run as a quality check with the script 33-get-sasa-uncondensed.tcl. Finally, 40a-place-some-quinones-outerleaflet.tcl and 40a-place-some-quinones-innerleaflet.tcl can be used to place quinones in the membrane. Once the final model is assembled, 99-load-everything.tcl and check-vesicle.tcl can be used for quality assurance.

3.2 Equilibration

The model produced in the previous section is likely to be unstable due to the inaccurate number of lipids in the inner and outer leaflet and disproportionate number water molecules placed on either side of the membrane. Therefore, before moving onto production level runs, the model requires minimization and equilibration. This step is achieved through a combination of short MD simulations and Tcl scripts.

3.2.1 Water

First, an NPT MD simulation is run with lipid headgroups and proteins fixed so that the lipid tails are allowed to relax (0.5 ns). After the tails have been allowed to relax, another NPT MD simulation with no constraints is ran (10 ns). This will cause the membrane to explode and form holes. The holes are then used to allow the number of water molecules on each side of the membrane to converge. Once water has equilibrated, we use a combination of LipidWrapper [39] and grid forces capabilities of NAMD [40] to patch the holes in the membrane and equilibrate the rest of the model.

3.2.2 Protein & Lipids

First, we use the scripts in modeling/hole-fixing/Lipid-wrapper-scripts/ as described by [26].

Bad contacts are then removed by shrinking the relevant lipids with the scripts in modeling/hole-fixing/lipid-shrinking-scripts/. More detailed instructions are made available with the scripts. After patching the holes, a gridforce map is created by simulating the density of water in the system which effectively constrains the equilibrated water and prevents water from solvating the holes created in subsequent simulations. Lastly, another short NPT MD simulation is running, this time with gridforces on, to equilibrate the proteins and lipids as well as create more holes if the number of lipids is inadequate. This process is repeated iteratively until the membrane stabilizes and holes cease to form. A typical NAMD configuration script for equilibration chromtest.namd is provided. Assuming the successful completion of the previous steps, production level runs are now ready to be simulated.

3.3 Analysis

APBS and ARBD were both used to reveal the rate-limiting step of membrane-mediated energy conversion in the purple bacterium chromatophore. The following section will present a general outline for the implementation of each technique to large systems simulations. Links to sample inputs fills are provided.

3.3.1 APBS

Electrostatic properties of the system can be derived by solving the nonlinear Poisson–Boltzmann equation within each frame of the MD simulation trajectory and averaging over all frames. These computations can be repeated over a range of pH and salinity conditions to determine how robust the physical properties of the biological constructs are to environmental changes.

To begin, the .js file of the simulation system is converted to a .pdb file. Then, this .pdb for the system may be converted to a .pqr file which will be read by APBS. Notably a .pqr file resembles the PDB format with the exception of the occupancy and the beta columns being replaced by the charge and mass of every atom. Three additional commands can be utilized to allow APBS to read additional parameters, all of which have the format .dx, which is a flexible scalar data format known as OpenDX. First, the command diel tells APBS to read the dielectric function, and paths to the x-, y-, and z-shifted dielectric map files must be specified in the input file. Second, if the system has a nonzero ionic strength value, an additional command, kappa, must be implemented, allowing APBS to read the ion-accessibility function. Lastly, charge allows APBS to read the fixed molecular charge density function.

Five different types of electrostatics calculations can be performed on the system, however particular focus will be given to mg-para, the calculation used for the chromatophore. Automatic parallel focusing multigrid calculations (mg-para) perform single-point calculations on systems and evaluate the electrostatic potential on a large scale. All relevant keywords implemented in our input will be briefly covered below.

The amount of overlap between individual processors meshes is specified with ofrac, which is a value between 0 and 1. Generally, a value of 0.1 is sufficient to generate stable energies. npbe specifies that the nonlinear Poisson–Boltzmann equation will be solved. bcfl flag defines the type of boundary conditions, where flag can be zero, sdh, mdh, or focus. zero and focus are generally not used with mg-para calculations, while sdh and mdh refer to "Single-Debye-Huckel" and "MultipleDebye-Huckel" models, respectively. The former is used for larger systems, as it describes a model of a single sphere with a point charge. Pdievalue and sdievalue specify the dielectric constant of the molecule and solvent, respectively. These values range from 2 to 20 for molecules depending on the extent of polarization to be considered, while a value in the range of 78 - 80 is generally used for biological conditions. Srfm flag specifies the model used to construct the dielectric ion-accessibility coefficients, where flag is defined as mol, smol, or spl2. Here, we used spl2 to define the coefficients by a cubic-spline surface that is very stable relative to grid parameters. chgm flag, the method by which point charges are mapped onto the grid accordingly used spl2 as its discretization method. srad value specifies the radius of solvent molecules, which is usually set to 1.4 for water. Lastly, swin value defines the size of the support for spline-based definitions, with a usual value of 0.3. Examples of APBS input files can be found at [26].

3.3.2 ARBD

Brownian Dynamics (BD) simulations are a coarse grain simulation method for simulating the dynamics of molecules in solution at time scales which are not achievable with typical MD simulations. Coarse graining is necessary to model the relatively slow microsecond diffusion timescales of molecules in solution. In this case, two separate BD simulations were used to simulate the dynamics of both quinone and cytochrome c_2. These simulations make use of the electrostatic potential information created in the previous APBS calculations and diffusion coefficients created by the Hydropro software (http://leonardo.inf.um.es/macromol/programs/hydropro/hydropro.htm). The resulting diffusion coefficients and .dx files contain electrostatics information that is used by ARBD to calculate the forces on the various molecules in the simulation using a symplectic integrator [28].

Among other general parameters in the BD input file are the information collected from Hydropro, a defined dummy particle and rigid body, and various grid files. First, it is important to note that the timestep is described in nanoseconds. Next, a dummy atom is defined to act as a reference particle if a rigid body is to be added. Here, we defined cytochrome *c2* as our rigid body, and its mass, inertia, and damping coefficients were obtained from Hydropro. Lastly, two additional files must be generated: charge density and Van Der Waals force potential grid files. The former can be obtained by utilizing the volmap density command in VMD, then opt to weigh charge. The latter can also be generated in VMD, using the Implicit Ligand Sampling (ILS) plugin. Examples of ARBD implementation scripts can be found at [26].

3.4 Conventional Workflow

The recent hardware improvements and software efficiency provide continuing growth of the capacity for molecular biology, climate science, and chemistry, particularly for MD simulations and data analytics. A robust workflow management tool creates an automated environment to build, simulate and analyze large systems. The workflow management tool consists of RADICAL-Cybertools (RCT), including the following middleware building blocks: Ensemble-Toolkit (EnTK), RADICAL-Pilot (RP), and RADICAL SAGA (RS).

RADICAL Ensemble Toolkit (EnTK) [41] provides scalable workflow management capabilities on high performance computing (HPC). EnTK is a Python implementation of a workflow engine, specialized in supporting the programming and execution of applications with ensembles of tasks. EnTK executes tasks concurrently or sequentially, depending on their arbitrary priority relation. Tasks are scalar, MPI, OpenMP, multiprocess, and multithreaded programs that run as self-contained executables. Tasks are not functions, methods, threads, or subprocesses. EnTK uses the Pipeline, Stage, and Task model (PST) model to encode workflows. Tasks are grouped in stages, indicating that they have no

input–output relationships and can be executed concurrently, depending on resource availability. Stages are collected into pipelines, indicating that tasks of different stages have input/output dependencies and have to be executed sequentially.

RP is a portable, modular and extensible Pilot system [42], written in the Python programming language. The pilot abstraction enables the separation between resource acquisition and the scheduling of tasks on those resources. RP acquires resources by submitting a job to the HPC batch system. Once the job is scheduled, RP enables task scheduling, placement, and launching on the job's resources. Tasks can be scheduled concurrently and sequentially, depending on resource availability. In this way, RP enables scalable execution of multitask applications with high throughput on HPC resources. Note that RP is fully compliant with the policies of each HPC platform: resources are acquired via the platform's batch system and used until their walltime expires.

RS is a Python implementation of the Open Grid Forum SAGA standard GFD.90 [43], a high-level interface to distributed infrastructure components like job schedulers, file transfer, and resource provisioning services. RS enables interoperability across heterogeneous distributed infrastructures, improving on their usability and enhancing the sustainability of services and tools [44].

3.5 Iterative Workflow Using NAMD-EnTK

Large scale MD simulations presented in Subheading 3.1 require submitting numerous scripts for model building, simulation, and analysis. This makes managing the execution of these simulations difficult, especially when considering the need to scale these execution large HPC platforms in the world. In EnTK, the PST model allows to create and submit heterogeneous pipelines to support multiple iterations of model building, simulation, and analysis tasks. Data sharing among tasks, stages, and pipelines is simplified by enabling transferring intermediate files (i.e., simulated maps) while the workflow is being executed. EnTK interface lets users label each element of the workflow via unique identifiers to locate and relate data at each step of the workflow execution. Finally, users can code fine-grained specification of computational requirements, including a number of CPU/GPU, and number and type of process/thread for each task of the workflow.

Using EnTK we created a nine-stage pipeline on XSEDE Bridges supercomputer. The pipeline automates the procedure to iteratively assemble protein structures in membrane: the iterative model building, refinement and analysis step, refined model solvation and, finally, equilibration (*see* Subheadings 3.1 and 3.2). After the final system is prepared and equilibrated in an NPT ensemble, we setup another iterative pipeline between the production run in NVT ensemble and analysis (*see* Subheading 3.3). We used two nodes, utilizing 52 compute cores in total but EnTK is designed to seamlessly scale large scale iterative pipelines to up to 100,000 cores [41].

Multiple Gridforce based NAMD simulations, as implemented in molecular dynamics flexible fitting (MDFF) are encoded as tasks of the PST. The VMD preprocessing steps (i.e., generating gridforce file .dx, building NAMD configuration files .namd) with the initial structure are codified as the stages of pipelines. These EnTK pipelines are used to perform simulations and analysis described in Subheading 3.2. The current implementation of the workflow is available on a dedicated GitHub repository [45], offering a simple example of an iterative pipeline for the execution of an MD simulation and analysis workflow. The current workflow executes on OLCF Summit—which currently occupies the Number 1 position in the Top 500 Supercomputing list.

In the near future, we will extend existing capabilities to support the execution of up to 1024 concurrent pipeline. As illustrated in Fig. 2, the pipeline will support iterative execution of a cycle of simulations, analysis, and model refinement.

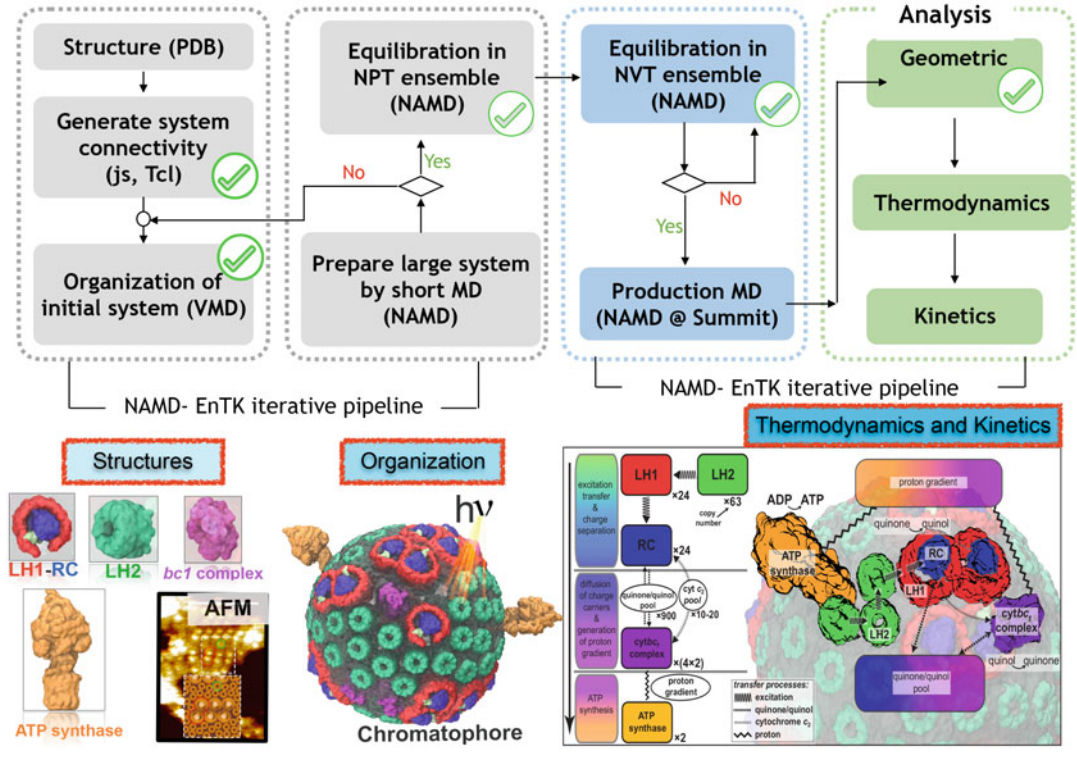

Fig. 2 NAMD-EnTK iterative pipeline on Summit. Flowchart illustrating the integration of molecular dynamics and visualization software (NAMD and VMD) in the ensemble toolkit (EnTK) iterative pipeline. Starting with initial structures of macromolecules, the structures are organized to assemble the photosynthetic chromatophore vesicle. Dynamics is performed at Summit, a GPU-accelerated petascale supercomputer. Checked boxes are steps of the algorithm that has been implemented using this pipeline [45]

4 Case Study

Here, we present an example of a large system simulation. This case study focuses on the simulation of a photosynthetic chromatophore [2], which absorbs solar energy to generate ATP, the so-called energy currency of life. The protein structure of the chromatophore vesicle was originally obtained by Cartron et al. [34].

4.1 Arrange Proteins

The first step is to arrange the proteins of the large complex in the shape of a sphere. Here, the vesicle patches imaged by AFM [34] are aligned relative to each other so that it is consistent with linear dichroism on intact membrane vesicles [2]. Overall, the model features 82 bioenergetic complexes (which includes 63 light-harvesting LH2 complexes, 11 dimeric and 2 monomeric RC-LH1 complexes, 4 bc_1 dimers, and 2 ATP synthases), together with 4011 light absorbing antenna molecules (comprised of 2469 bacteriochlorophyll and 1542 carotenoid). The spherical arrangement of these proteins are constructed by positioning the planar strips containing the RC–LH1 dimer arrays and their immediate LH2 neighbors along the north–south direction. The spaces between RC–LH1 strips are filled with LH2rich regions. An area-preserving map, the inverse-Mollweide transformation [24] is used to map planar regions from the two-dimensional AFM images onto three-dimensional spherical ones. Structural conflicts are subsequently removed and gaps which arise between the pigment–protein complexes as a result of this projection process are removed manually by shifting the center of the proteins on the chromatophore sphere ensuring a tight packing of the structure. The "southern polar" region is left empty as a potential contact zone with the rest of the membrane.

4.2 Initial Equilibration

The chromatophore consists of a vesicle from AFM [34], and MD-equilibrated structures of individual POPC-embedded LH1-RC, LH2 [46], bc_1 [47], and ATP synthase models. Each protein within the chromatophore scaffold is overlaid with their membrane-embedded counterpart. The original protein inside the chromatophore is replaced by equilibrating the model with one lipid ring surrounding it. A POPConly lipid vesicle of radius 30 nm is used to uniformly construct a lipid bilayer for the protein-excluded areas of the chromatophore surface. This lipid vesicle is then overlaid with the ring-encased proteins with a 2 Å exclusion radius, which is sufficient to avoid unfavorable steric interactions between the proteins and the lipids. The use of protein–lipid ring for membrane embedding instead of a protein-only embedding avoids the formation of ring-piercing artifacts. However, some unfavorable overlap remains due to the inter-twining of the lipid tails. Due to direct minimization being unable to remove

this artifact, the intercarbon distances in the lipid tail is shrunk (i.e., to a single carbon atom) until the intertwining was removed. Thereafter, brute-force energy minimization algorithm within NAMD resurrects the length of the lipid tails while avoiding the unphysical twining effect. After energy minimization, the vesicle is immersed in a water drop, followed by randomly mutating the lipid headgroups to obtain a lipid membrane composition containing 22% POPC, 22% POPG, and 56% POPE on the outer leaflet and 24% POPC, 10% POPG, and 66% POPE on the inner leaflet. The final step consisted of adding 900 quinone molecules resulting in the generation of a cubic simulation box of dimension 110 nm containing 136-M atoms.

4.3 Membrane Equilibration

A direct MD run starting with the 136-M atoms initial model leads to instabilities in the system within 1 ns of simulation (Fig. 3). These instabilities exhibit protein and/or lipid-excluded holes on the chromatophore surface. This is a result from a combination of (1) inaccurate number of lipid molecules and (2) water molecules on either side of the initial chromatophore membrane. The following strategy is employed to decouple the two issues and address them sequentially. First, a grid potential is defined at a resolution of 1 Å about the chromatophore surface, to softly repel the water away from the surface, while allowing the protein–lipid and lipid–lipid interactions to equilibrate during short 5 ns NVT simulations. Now, the present setup will only allow instabilities in the form of holes on the chromatophore surface due to insufficient number of lipid molecules. The imbalance in the number of water molecules on either side of the membrane will have minimal consequence on the chromatophore stability. The holes are allowed to form and equilibrate, but water passage is negated due to the presence of the soft grid-potential. The tool LipidWrapper [39] is employed to fix these holes through the insertion of filler lipids (Fig. 4). This script is available on GitHub [26]. Iteratively, after each round of hole formation, the holes are filled until LipidWrapper cannot identify any new holes. In the present case, a total of 4 iterations are performed for the number of lipids to converge.

4.4 Water Equilibration

In order to equilibrate the water on either side of the membrane, two LH1-monomers are removed from the chromatophore surface, creating few holes on the surface. Lipids within a 5 Å radius around these holes are constrained using harmonic restraints. Hole formation allows for passage of water molecules across the membrane, thereby letting the rest of the chromatophore relax without inducing any global instability. LH1 monomers were chosen for removal due to the observation that the area around these proteins were the most susceptible to instability. Thus, our choice of artificially creating holes in these areas and constraining the vicinity prevents any instability propagation through the membrane. It

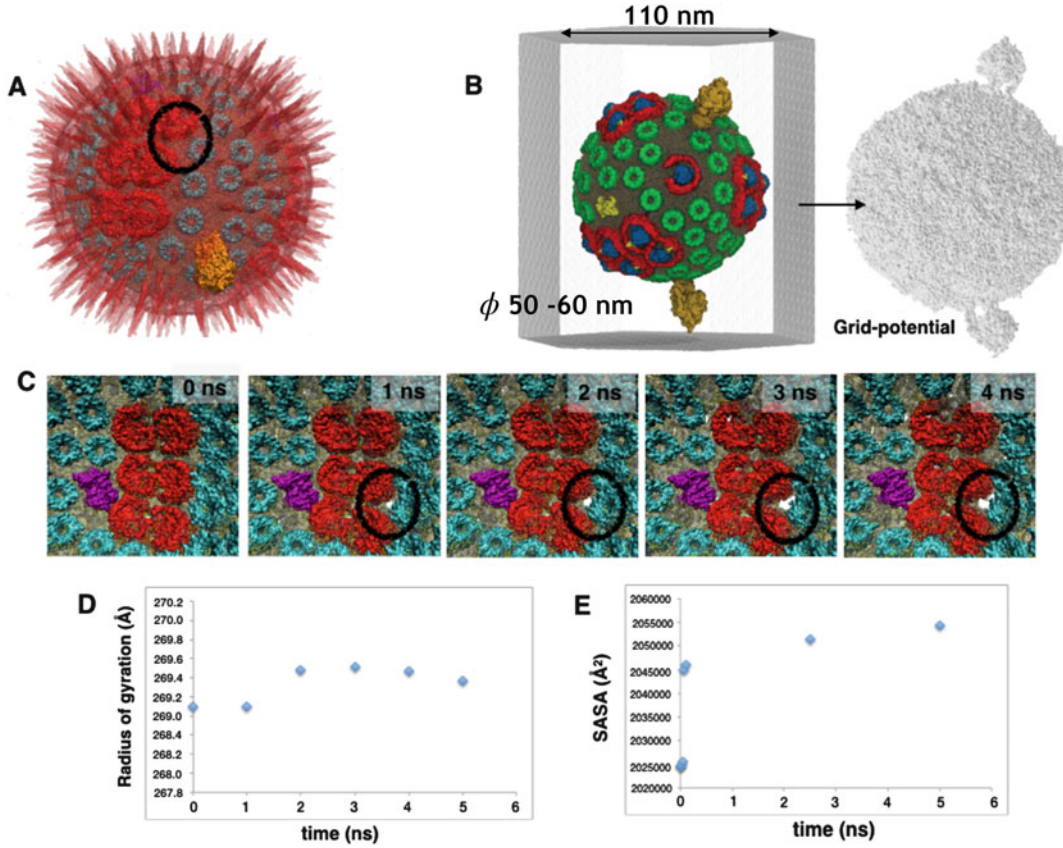

Fig. 3 Stepwise refinement of the initial chromatophore model. (**a**) Snapshot demonstrating instabilities in the 136 M-atom solvated and ionized chromatophore model after 1 ns of MD simulation. These instabilities, manifested as protein and/or lipid-excluded holes on the chromatophore surface, result from a combination of inaccurate number of lipid molecules and water molecules on either side of the initial chromatophore membrane. (**b**) A grid potential defined at the resolution of 1 Å about the chromatophore surface, which softly repels the water away from the surface, while allowing the allowing the protein–lipid and lipid–lipid interactions to equilibrate during short MD simulations for refining the initial model. (**c**) Snapshots illustrating equilibration of protein–lipid and lipid–lipid interactions during short 4 ns-MD simulations, which showcases hole formation (black ring) implying an inadequacy in the number of membrane lipids within the chromatophore model. The formation and propagation of holes due to inadequacy in the number of modeled lipids emerges as a surface artifact with minimal change in the overall radius of gyration (**d**), yet with significant alteration of the surface area (**e**)

takes approximately 25 ns for the solvent density to converge on either side of the membrane (Fig. 5). Subsequently, the missing LH1 monomers are reintroduced into the membrane by excluding some local water molecule.

The grid-constrained NVT simulations, for correcting the number of lipids on-the-fly, used a gscale of 0.3 [48]. After these short NVT simulations, the long NPT production runs are performed at 1 atm pressure using the Nose–Hoover Langevin piston with a period of 100 fs and damping timescale of 50 fs. Every model

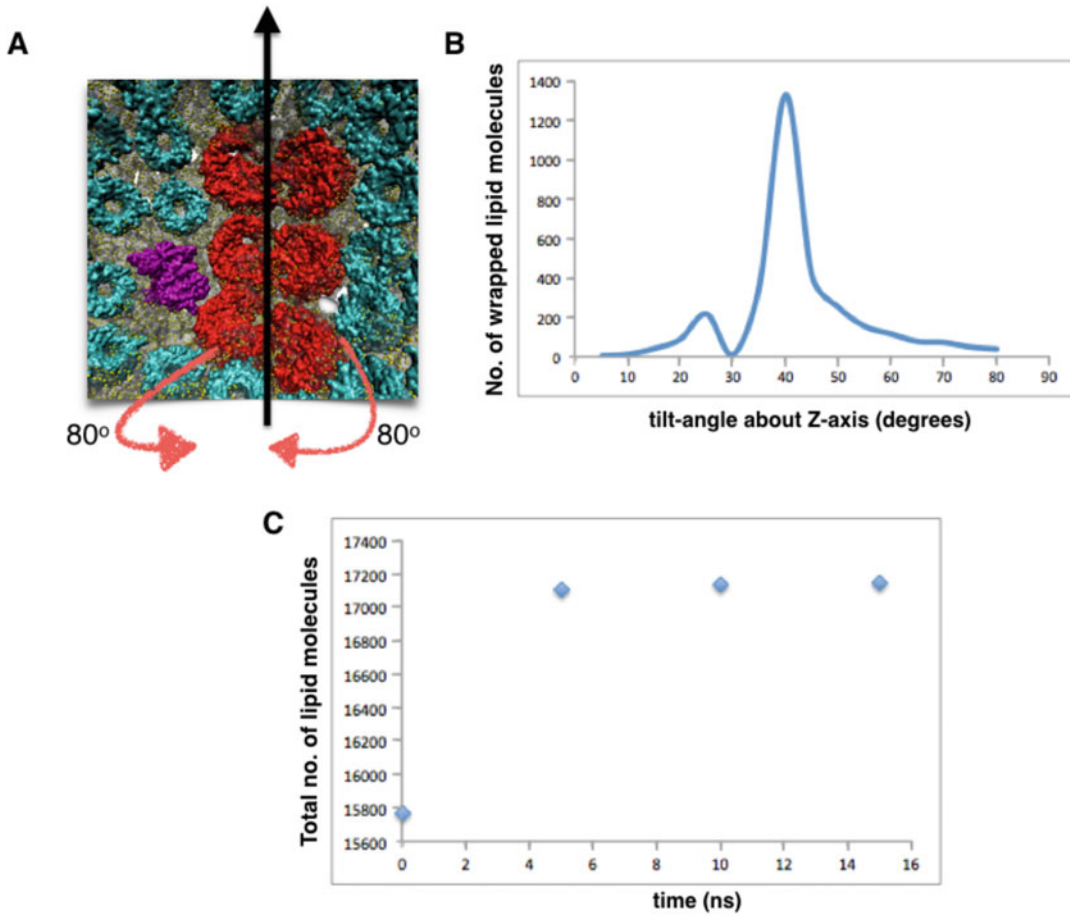

Fig. 4 Iterative rerefinement of the chromatophore membrane lipids with Lipid Wrapper. (**a**) schematic of lipid wrapper implementation [39], whereby a selected piece of membrane with holes is independently rotated about the *X*, *Y*, and *Z* axes to superimpose lipids from the rotated membrane with the holes from the original orientation. Lipid molecules are extracted from the rotated membrane and placed into holes of the original membrane to correct simultaneously the POPE, POPC, and POPE density of the chromatophore membrane. (**b**) Plot showing the number of extractable lipids at different rotation angles of a typical Lipid Wrapper scan. An angle of 40° is chosen about the *X*, *Y*, and *Z* axes for this particular instance as it provides the maximum number of extractable lipids for filling holes in the original membrane. (**c**) Four iterations of Lipid-wrapper-based lipid modeling each followed by a 4 ns of MD equilibration under grid forces were performed to converge to the final lipid count of 17,200

from an iteration of lipid updates (Fig. 5), is minimized for 10,000 steps prior to the adding more lipids in the next round; a total of $4 \times 10{,}000 = 40{,}000$ minimization steps are performed prior to solvent equilibration.

4.5 Production Run

Following the equilibration of water in the artificially water permeable chromatophore (created by removing two RC-LH1 monomers), these two proteins are reinserted into the equilibrated chromatophore.

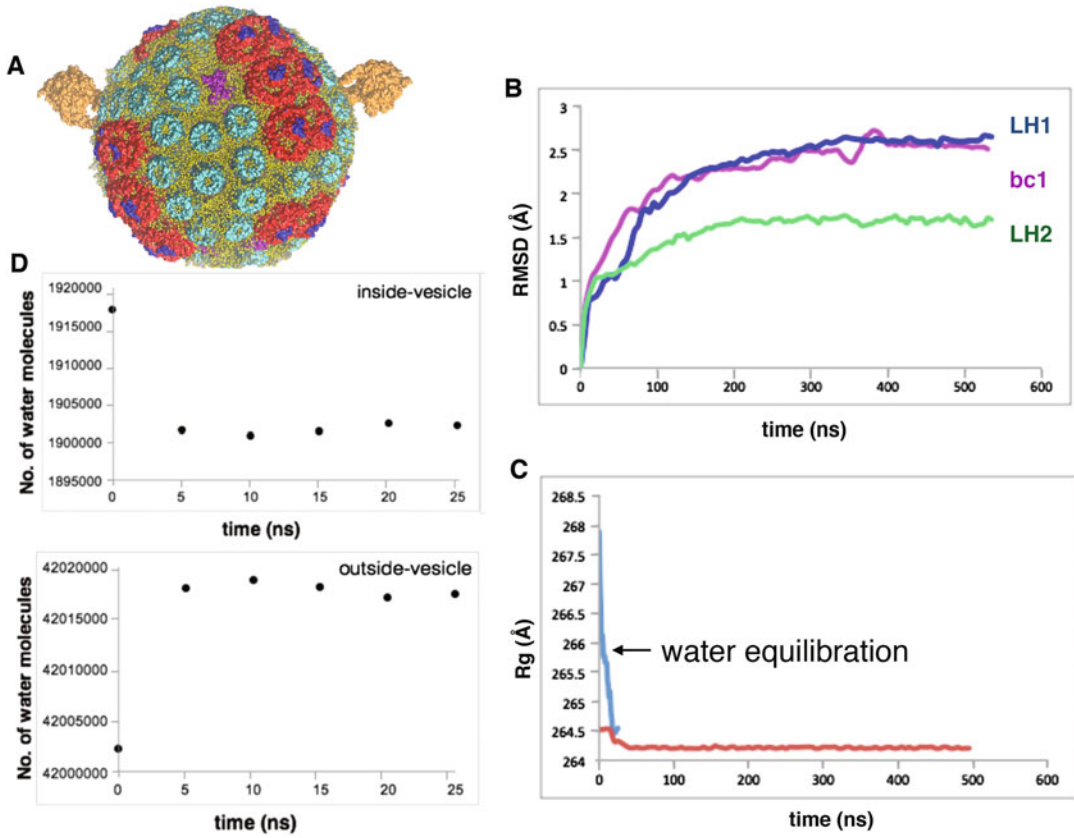

Fig. 5 Equilibration of a stable chromatophore model. (**a**) An all-atom model of the photosynthetic chromatophore vesicle in Rhodobacter sphaeroides derived from 50 ns of model-refinement and 0.5 ns of equilibrium MD simulations. The final simulation model features 82 bioenergetic complexes (67 Light-Harvesting LH2 complexes [green], 11 dimeric and 2 monomeric RC-LH1complexes [LH1:red; RC:blue], 4 cytochrome bc1 complex dimers [magenta], and 2 ATP synthases [orange]), together with 4011 light absorbing antenna molecules (2469 bacteriochlorophylls and 1542 carotenoid) embedded in a membrane of 17,200 lipid molecules (Lipid phosphate indicated in yellow). (**b**) Root mean square deviation (RMSD) plotted as a function of simulated time showing that bioenergetic proteins have been sufficiently equilibrated over 0.5 ns. Consistent with simulations of a flat chromatophore patch 32], LH1 dimer is found to be the most flexible and LH2 the least flexible even when the chromatophore assembles as a vesicle. (**c**) Change in the radius of gyration of the vesicle showing a drop of 4 Å during the final 25 ns of model refinement stage, wherein the protein–lipid packing tightens and lipid composition reorganizes in the vicinity of the proteins, following which the value plateaus reinstating the achievement of a stable chromatophore model. (**d**) Number of water molecules inside and outside the chromatophore vesicle (defined by a radius of 280 Å from the center) showing the excess water molecules from inside are released through the hole outside the vesicle over 5 ns, after which the water molecules equilibrated

After 5000 more minimization steps are performed in the vicinity of the LH2 complex, restraining the rest of the chromatophore. Thereafter the long production run is performed for 500 ns.

All simulations are performed with the MD software NAMD 2.12 using the CHARMM36 force field for proteins and lipids

[27]. The force field parameters for the photosynthetic cofactors such as chlorophyll and quinone molecules have been obtained from our past studies of the individual proteins [32, 46]). A sample NAMD configuration file is provided on GitHub [26], including all the necessary input arguments employed by the simulation. Some key arguments are mentioned and requires special attention. It is assumed that the reader is familiar with basic NAMD configuration script and parameters employed by the simulation.

Simulations are performed with an integration time step of 1 fs where bonded interactions are computed every time step, short-range nonbonded interactions every two timesteps, and long-range electrostatic interactions every four timesteps. A cutoff of 12 Å is used for short-range electrostatic and van der Waals interactions: a switching function is started at 10 Å for van der Waals interactions to ensure a smooth cutoff. Periodic boundary conditions are used, with full-system, long-range electrostatics calculated by using the PME method with a grid point density of 1/Å. To avoid short-range interactions between adjacent copies of the system, it is ensured that the unit cell is large enough. Since the chromatophore dimensions vary between 50 and 60 nm, a water box of 110 nm (*see* Fig. 3b) maintains a padding of 25 nm on all sides. The systems were kept at constant temperature using Langevin dynamics for all nonhydrogen atoms with a Langevin damping coefficient of 5 ps^{-1}.

4.6 APBS Analysis

Cytochrome c_2 is responsible for shuttling electrons from the bc_1 to RC-LH1. Binding dynamics of Cytochrome c_2 to the integral membrane protein complexes is influenced by the charge state of the inside of the chromatophore [48]. Brownian dynamics simulations of cytochrome required APBS calculations at six different conditions: pH 4 with 0.02 M NaCl, pH 7 with 0.15 M NaCl, and pH 7 with 0.25 M NaCl, for both the reduced and oxidized forms of the chromatophore to represent microenvironments that single electron transfer proteins might experience while shuttling charges across a membrane. In the reduced chromatophore, bc_1 complexes were modeled as reduced, cytochrome c_2 was oxidized, and the RC-LH1 were neutral. In the oxidized chromatophore, the bc_1 complexes were set to neutral, cytochrome c_2 was reduced, and the RC-LH1 were oxidized. APBS simulations revealed that between the pH values of 6–7 and salt concentrations of 0.15 M to 0.95 M, the effective charge of the surface of the chromatophore was relatively unchanged. However, as salt was reduced below 0.15 M, the chromatophore charge dropped precipitously due to an absence of counter ions. Additionally, pH below 4 resulted in protonation of acidic side chains, where at higher pH, deprotonation enhances surface charge and protein–protein interactions with cytochrome c_2.

4.7 ARBD Analysis

Each rigid body simulation began with a single cytochrome c_2 at the center of the chromatophore that was allowed to diffuse in solution and interact with the various surfaces on the inside of the chromatophore. For each condition, 500 independent 10 μs simulations each with 100-fs timesteps were performed and cyt. c_2 coordinates were recorded every nanosecond. Interactions were tracked for each of the conditions, with an interaction defined as cytochrome c_2 being within 1 nm of a nonsolvent component of the chromatophore. Simulations in the low-salt conditions displayed higher surface binding overall; however, the specificity of binding was greatly reduced compared to physiological salt conditions. In simulations where proteins were modeled with oxidized conditions, cytochrome c_2 favored interactions with RC-LH1 complex as opposed to bc_1. Conversely, in simulations under reducing conditions cytochrome c_2 was more likely to interact with bc_1 than RH. This follows the logic of the physiological role of cytochrome c_2 being to dock to the surface of a reduced bc_1, accept an electron and become reduced, then unbind and interact with oxidized RC-LH1, reducing it and starting the cycle over again. The association times of cytochrome c_2 between bc_1 and RH-LH1 were 13 ns and 70 ns respectively when BD simulations were performed at pH 7 with 0.15 M NaCl. Reversible binding times (combined association and dissociation times) were 1 ms and 0.2 ms respectively. This binding time was found to be slower than the time it took cytochrome c_2 to shuffle between proteins.

Executing concurrent pipelines using represents an effective way of processing a series of simulations, model refinement, and analysis at supercomputers. Middleware solutions such as those provided by EnTK present the opportunity to utilize high-performance computers and large-scale computing. They provide the building blocks for iterative pipeline based workflows. In the near future, we plan to extend capabilities to support multiple pipelines of chromatophore simulations and analysis.

Acknowledgments

The authors acknowledge start-up funds from the School of Molecular Sciences and Center for Applied Structure Discovery at Arizona State University, and the resources of the OLCF at the Oak Ridge National Laboratory, which is supported by the Office of Science at DOE under Contract No. DEAC05-00OR22725, made available via the INCITE program. We also acknowledge NAMD and VMD developments supported by NIH (P41GM104601) and R01GM098243-02 for supporting our study of membrane proteins.

References

1. Alberts B (2010) Cell biology: the endless frontier. Mol Biol Cell 21(22):3785
2. Singharoy A, Maffeo C, Delgado-Magnero K et al (2019) Atoms to phenotypes: molecular design principles of cellular energy metabolism. Cell 179(5):1098–1111.e23
3. Goh BC, Hadden JA, Bernardi RC et al (2016) Computational methodologies for real-space structural refinement of large macromolecular complexes. Annu Rev Biophys 45:253–278
4. Voth GA (2017) A multiscale description of biomolecular active matter: the chemistry underlying many life processes. Acc Chem Res 50(3):594–598
5. Davtyan A, Simunovic M, Voth GA (2016) Multiscale simulations of protein-facilitated membrane remodeling. J Struct Biol 196 (1):57–63
6. Van Meel JA, Arnold A, Frenkel D et al (2008) Harvesting graphics power for md simulations. Mol Simul 34(3):259–266
7. Ananthraj V, De K, Jha S et al (2018) Towards exascale computing for high energy physics: The atlas experience at ornl. In: 2018 IEEE 14th international conference on e-science (e-science), pp 341–342
8. Kilburg D, Gallicchio E (2016) Recent advances in computational models for the study of protein–peptide interactions. Adv Protein Chem Struct Biol 105:27–57
9. Ourmazd A (2019) Cryo-em, xfels and the structure conundrum in structural biology. Nat Methods 16(10):941–944
10. Marrink SJ, Corradi V, Souza PC et al (2019) Computational modeling of realistic cell membranes. Chem Rev 119(9):6184–6226
11. Feig M, Harada R, Mori T et al (2015) Complete atomistic model of a bacterial cytoplasm for integrating physics, biochemistry, and systems biology. J Mol Graph Model 58:1–9
12. Yu I, Mori T, Ando T et al (2016) Biomolecular interactions modulate macromolecular structure and dynamics in atomistic model of a bacterial cytoplasm. elife 5:e19274
13. Perilla JR, Schulten K (2017) Physical properties of the hiv-1 capsid from all-atom molecular dynamics simulations. Nat Commun 8:15959
14. Wickles S, Singharoy A, Andreani J et al (2014) A structural model of the active ribosome-bound membrane protein insertase yidc. elife 3:e03035
15. Trabuco LG, Villa E, Mitra K et al (2008) Flexible fitting of atomic structures into electron microscopy maps using molecular dynamics. Structure 16(5):673–683
16. Schweitzer A, Aufderheide A, al Rudack T (2016) Structure of the human 26s proteasome at a resolution of 3.9 Å. Proc Natl Acad Sci U S A 113(28):7816–7821
17. Durrant JD, Bush RM, Amaro RE (2016) Microsecond molecular dynamics simulations of influenza neuraminidase suggest a mechanism for the increased virulence of stalk-deletion mutants. J Phys Chem B 120 (33):8590–8599
18. Mannige RV, Brooks CL III (2010) Periodic table of virus capsids: implications for natural selection and design. PLoS One 5(3):e9423
19. Blood PD, Voth GA (2006) Direct observation of bin/amphiphysin/rvs (bar) domaininduced membrane curvature by means of molecular dynamics simulations. Proc Natl Acad Sci U S A 103(41):15068–15072
20. Arkhipov A, Yin Y, Schulten K (2008) Four-scale description of membrane sculpting by bar domains. Biophys J 95(6):2806–2821
21. Jung J, Nishima W, Daniels M et al (2019) Scaling molecular dynamics beyond 100,000 processor cores for large-scale biophysical simulations. J Comput Chem 40 (21):1919–1930
22. Renaud J-P, Chari A, Ciferri C et al (2018) Cryo-EM in drug discovery: achievements, limitations and prospects. Nat Rev Drug Discov 17(7):471–492
23. Camargo C (2018) Physics makes rules, evolution rolls the dice. Science 361 (6399):236–236
24. Şener MK, Olsen JD, Hunter CN et al (2007) Atomic-level structural and functional model of a bacterial photosynthetic membrane vesicle. Proc Natl Acad Sci 104(40):15723–15728
25. Blankenship RE (2014) Molecular mechanisms of photosynthesis. John Wiley & Sons, Hoboken, New Jersey
26. Vant, J. W. (2019). Chromatophore_large_system_simulation. https://github.com/jvant/Chromatophore_Large_System_Simulation GitHub
27. Phillips JC, Braun R, Wang W et al (2005) Scalable molecular dynamics with NAMD. J Comput Chem 26(16):1781–1802
28. Comer J, Aksimentiev A (2016) DNA sequence-dependent ionic currents in ultra-small solidstate nanopores. Nanoscale 8 (18):9600–9613

29. Humphrey W, Dalke A, Schulten K (1996) VMD: Visual molecular dynamics. J Mol Graph 14(1):33–38
30. Singharoy A, Cheluvaraja S, Ortoleva P (2011) Order parameters for macromolecules: application to multiscale simulation. J Chem Phys 134 (4):044104
31. Acun B, Hardy DJ, Kale LV et al (2018) Scalable molecular dynamics with NAMD on the summit system. IBM J Res Dev 62(6):1–9
32. Chandler DE, Strümpfer J, Sener M et al (2014) Light harvesting by lamellar chromatophores in rhodospirillum photometricum. Biophys J 106(11):2503–2510
33. Şener M, Strümpfer J, Timney JA et al (2010) Photosynthetic vesicle architecture and constraints on efficient energy and harvesting. Biophys J 99(1):67–75
34. Cartron ML, Olsen JD, Sener M et al (2014) Integration of energy and electron transfer processes in the photosynthetic membrane of rhodobacter sphaeroides. Biochim Biophys Acta 1837(10):1769–1780
35. Kumar S, Cartron ML, Mullin N et al (2016) Direct imaging of protein organization in an intact bacterial organelle using high-resolution atomic force microscopy. ACS Nano 11 (1):126–133
36. Scheuring S, Nevo R, Liu L-N et al (2014) The architecture of rhodobacter sphaeroides chromatophores. Biochim Biophys Acta 1837 (8):1263–1270
37. Russel D, Lasker K, Webb B et al (2012) Putting the pieces together: integrative modeling platform software for structure determination of macromolecular assemblies. PLoS Biol 10 (1):e1001244
38. Ho PT, Montiel-Garcia DJ, Wong JJ et al (2018) VIPERdb: a tool for virus research. Annu Rev Virol 5(1):477–488
39. Durrant JD, Amaro RE (2014) Lipidwrapper: an algorithm for generating large-scale membrane models of arbitrary geometry. PLoS Comput Biol 10(7):e1003720
40. Wells DB, Abramkina V, Aksimentiev A (2007) Exploring transmembrane transport through α-hemolysin with grid-steered molecular dynamics. J Chem Phys 127(12):09B619
41. Balasubramanian V, Turilli M, Hu W et al (2018) Harnessing the power of many: extensible toolkit for scalable ensemble applications. In: In 2018 IEEE international parallel and distributed processing symposium (ipdps). IEEE, New York, pp 536–545
42. Turilli M, Santcroos M, Jha S (2018) A comprehensive perspective on pilot-job systems. ACM Comput Surv 51(2):43:1–43:32
43. Goodale T, Jha S, Kaiser H et al (2006) SAGA: a simple API for grid applications, high-level application programming on the grid. Comput Methods Sci Technol 12(1):7–20
44. Merzky A, Weidner O, Jha S (2015) SAGA: a standardized access layer to heterogeneous distributed computing infrastructure. Software-X 1-2:3–8
45. MDFF Integration with EnTK on OLCF Summit. (2019). https://github.com/radical-collaboration/MDFF-Error.GitHub
46. Chandler DE, Hsin J, Harrison CB et al (2008) Intrinsic curvature properties of photosynthetic proteins in chromatophores. Biophys J 95(6):2822–2836
47. Singharoy A, Barragan AM, Thangapandian S et al (2016b) Binding site recognition and docking dynamics of a single electron transport protein: cytochrome c 2. J Am Chem Soc 138 (37):12077–12089
48. Singharoy A, Teo I, McGreevy R et al (2016a) Molecular dynamics-based model refinement and validation for sub-5 angstrom cryo-electron microscopy maps. eLife 5:e16105