JCTC Journal of Chemical Theory and Computation

Article

pubs.acs.org/JCTC

# High-Throughput Automated Preparation and Simulation of Membrane Proteins with HTMD

Stefan Doerr,[†,∥] Toni Giorgino,[‡,∥] Gerard Martínez-Rosell,[†] João M. Damas,[¶] and Gianni De Fabritiis*[,†,§]

[†]Computational Biophysics Laboratory (GRIB-IMIM), Universitat Pompeu Fabra, Barcelona Biomedical Research Park (PRBB), C/Doctor Aiguader 88, 08003 Barcelona, Spain

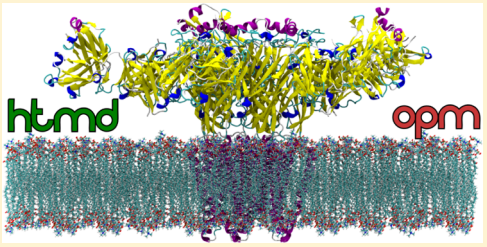
[‡]Institute of Neurosciences, National Research Council of Italy (IN-CNR), 35127 Padua, Italy

[¶]Acellera, Barcelona Biomedical Research Park (PRBB), C/Doctor Aiguader 88, 08003 Barcelona, Spain

[§]Institució Catalana de Recerca i Estudis Avançats (ICREA), Passeig Lluis Companys 23, Barcelona 08010, Spain

Supporting Information

**ABSTRACT:** HTMD is a programmable scientific platform intended to facilitate simulation-based research in molecular systems. This paper presents the functionalities of HTMD for the preparation of a molecular dynamics simulation starting from PDB structures, building the system using well-known force fields, and applying standardized protocols for running the simulations. We demonstrate the framework's flexibility for high-throughput molecular simulations by applying a preparation, building, and simulation protocol with multiple force-fields on all of the seven hundred eukaryotic membrane proteins resolved to-date from the orientation of proteins in membranes (OPM) database. All of the systems are available on www.playmolecule.org.




## 1. INTRODUCTION

Classical molecular-dynamics (MD) is a computationally-intensive technique which simulates the dynamic behavior of atomistic systems by modeling them as many-body systems moving under the Newtonian forces determined by a classical potential. The quality of the potential energy functions has undergone extensive refinements over the last decades and is vastly responsible for the predictive ability of MD which has been demonstrated by its application over a spectrum of biomolecular systems encompassing globular, transmembrane and unstructured proteins, drug−ligand interactions, and several others.[1−6] MD based discovery however still faces a variety of problems which hinder its wider application.

Historically, molecular simulation software introduced force-fields, file formats, and tools for preparing molecular systems and applying the force field to the corresponding models. More recently MD engines introduced support for multiple force fields (e.g., ACEMD,[7] Gromacs,[8] NAMD,[9] OpenMM,[10] Desmond[11]) either using force-field-dependent tools (e.g., `tleap`, `psfgen`) or by importing the force field file format into their own tools. Most of these tools only provide a simple translation of the model into a specific force field without helping the user to prepare the model correctly, for instance in terms of protonation states. Additionally, complex systems such as protein−membrane systems require specialized tools to correctly place the protein into the membrane and equilibrate it. For validation it is also often important to be able to test the same system setup using different force fields and simulation engines for comparison.

All these different steps hinder the simulation of more than a few protein systems at a time, even for an expert in molecular simulation. A few studies with multiple systems have been performed in the past,[12−14] but they most often require large amount of human intervention or the tools are not made publicly available. There is therefore a need for a protein-oriented framework to enable the building of systems in a fast, reproducible, and hands-off manner, making it possible to process structures with minimal intervention and then simulate them, possibly in a way independent of (a) the chosen force field and (b) the MD engine. Scriptable, end-to-end molecular environments of this type would pave the way toward more widespread adoption of MD in molecular discovery efforts.[15−17] HTMD aims to deliver such a framework as one of its goals.

In this paper we will focus on the first steps of an MD-based discovery workflow by describing the system preparation, building, and simulation capabilities of the open-source HTMD framework meant to satisfy the previously stated requirements. The capabilities of HTMD for large-scale and Markov-model-based postsimulation analysis have been described in a previous publication.[18]

Over the last years, various software packages have been developed to aid molecular dynamics system preparation, building, and simulation. These can be largely grouped into three categories: graphical user interface-based packages, web-based services, and programming frameworks. Commercial

**Received:** May 9, 2017
**Published:** July 19, 2017





DOI: 10.1021/acs.jctc.7b00480
J. Chem. Theory Comput. 2017, 13, 4003−4011

programs such as Maestro,[19] YASARA,[20] and MOE[21] already provide powerful graphical user interfaces for *interactive* molecule inspection, manipulation, and system preparation. This visual aspect makes them very accessible and allows users to easily and quickly verify the results of their actions as well as modify and undo changes. However, their commercial pricing models can restrict their availability to researchers and the visual character does not lend itself naturally to their integration in high-throughput automated procedures.

Web-based interfaces such as CHARMM-GUI[22] and MDWeb[23] provide web-browser-based services for preparing a system for simulation, with no software being required and accessibility from anywhere through an Internet connection. CHARMM-GUI,[22] for example, supports system building for multiple force fields and simulation software such as NAMD, GROMACS, AMBER, and OpenMM. The actions that can be performed are however restricted to the set of operations exposed through the web service and user modifications of the underlying protocols are relatively limited. Additionally, it is not practical to integrate these web-based interfaces into a unified molecular discovery workflow and it can be difficult to reproduce the whole procedure, which is documented in log files, and would often need to be repeated through manual interaction with the web interface. Furthermore, the use of a web interface limits its use to nonconfidential data sets. In summary, web servers do not provide a scriptable and reproducible framework for building and simulating MD systems.

Important efforts have already been made toward programmable frameworks for system building and simulation, including X-PLOR,[24] CNS,[25] VMD,[9] YASARA,[20] ParmEd,[26] OpenMM,[10] Ensembler,[27] and more. X-PLOR and CNS provide integrated environments for structure determination, manipulation, and simulation of experimental structures; VMD allows the building of CHARMM systems through the TCL scripting language; ParmEd allows topology and parameter modification and conversion between various force field formats; OpenMM allows building systems for AMBER, AMOEBA,[28] and CHARMM force fields; and Ensembler can build and equilibrate whole protein superfamilies through the use of template modeling.

HTMD is, however, in our opinion, the first software specifically focused toward providing an *integrated* high-level discovery environment for system preparation, building, simulation, analysis and visualization. This allows the user to perform molecular discovery, going from raw PDB files to the estimation of thermodynamics and kinetics of binding and folding, conveniently expressed and documented through the Python language. Its standalone nature combined with the open-source code allows its use on confidential data sets, makes the calculations transparent and traceable, and provides the user with the ability to modify any of its underlying algorithms. The use of a mature and rich programming platform such as Python allows the users to import and seamlessly integrate domain-specific third-party libraries, such as RDKit[29] for cheminformatics, Scikit-learn[30] for machine learning, Pandas[31] for data analysis, and innumerable others.

## 2. METHODS

**2.1. Protein Preparation.** The first step required for all-atom computational experiments is the preprocessing of the three-dimensional structures provided as PDB files like the ones found in the RCSB database. This step is required because the location of hydrogen atoms is normally not resolved in X-ray crystallography experiments, and must be determined by indirect methods.

One of the approximations done in most current MD simulation methods is the assumption of constant protonation states of chemical groups. Until constant-pH simulation methods become commonplace,[32] it is important to setup the simulated system so that residues are in the protonation states most likely for their chemical environment at a given pH. A related issue arising when preparing a system is whether neutral histidine (HIS) residues should be protonated at the $\delta$ or $\epsilon$ nitrogen. Finally, the carboxamide groups of asparagine (ASN) and glutamine (GLN) side chains have distinctly asymmetric hydrogen bonding donor−acceptor arrangements. Although they are usually free to flip during the dynamics, a proper collective arrangement of their orientation minimizes the electrostatic energy and optimizes the hydrogen-bonding network, thus allowing simulations to start from a more stable configuration.

HTMD allows the estimation of protonation states and optimization of the hydrogen-bonding network through the `proteinPrepare()` function[33] (Listing 1). The preparation procedure relies in the first place on the PROPKA3.1 software,[34,35] which provides the protonation state of each residue through an estimation of their p$K_a$ values based on residue's model compounds, desolvation effects, hydrogen bonding, and other electrostatic interactions. After the computation of per-residue protonation states, a modified version of the PDB2PQR software[36] is used to perform the combinatorial optimization of the hydrogen bonding network. PDB2PQR uses a Monte Carlo algorithm to sample $\chi$ dihedral angles of HIS, ASN, and GLN residues to optimize their hydrogen-bonding network. Furthermore, the $\delta$ or $\epsilon$ nitrogen protonation of histidines is also decided at this stage according to the most stable configuration, together with the resolution of major steric conflicts through a "debumping" algorithm.[37] Of note, PROPKA3.1's p$K_a$ determination takes into account small molecules; crystallographic water molecules, if present, are also included in the optimization step. Both PROPKA3.1 and PDB2PQR are automatically installed together with HTMD and their use does not require any further steps.

The system preparation function returns a protein structure protonated according to the optimum H bonding criterion and most likely protonation states at the specified pH. The structure can be used at later stages of the building process. In particular, the protonation states of residues are marked with names that are independent of the force field, which allows the later system building functions to apply the modifications in the way appropriate for the specific back-end (e.g., patches, in CHARMM's terminology).

The preparation function also returns a data structure with the decisions taken at the various stages of the preparation procedure, such as p$K_a$ values, protonation states, $\chi$-dihedral flipping, etc., which is suitable for both interactive and scripted inspections. In the case of transmembrane proteins, the data returned includes a warning flag for titratable residues exposed to the hydrophobic environment, whose p$K_a$ predictions may be inaccurate.[38]

**2.2. System Building.** HTMD provides a `Molecule` class with a suite of methods designed to facilitate system manipulation, preparation and visualization.[18] In particular, methods are available to convert a molecule taken from the PDB into a fully solvated and protonated system that can be


DOI: 10.1021/acs.jctc.7b00480
J. Chem. Theory Comput. 2017, 13, 4003−4011

Listing 1: Protein preparation

```
# The opiod mu receptor dimer (pre-oriented)
m = Molecule("4DKL.pdb")

# Prepare and optimize at pH 7
mopt, pd = proteinPrepare(m, pH=7.0,
    hydrophobicThickness=32.0,
    returnDetails=True)

# Save a report with the modification details
    (protonation, pKa, flipped for H-bonding,
    solvent exposure, etc.)
pd.data.to_excel("mor_report.xlsx")

# Verify histidines' protonation state
his = (pd.data.resname == "HIS")
pd.data[his][["resname", "resid",
    "protonation"]]

# Save a report of the membrane-exposed
    residues
memb_exp = pd.data.membraneExposed
pd.data[memb_exp] \
    .to_excel("mor_exposed_residues.xlsx")
```

used as a starting configuration for a simulation. It can be solvated in a water box, various ions can be added, and caps and other force field patches can be applied to the molecules at the system-building stage. Currently, HTMD has back-ends for building systems in both CHARMM (PSF) and AMBER (PRMTOP) formats. Importantly, to maximize compatibility with evolving features of said formats, the actual writing of the topologies is relayed to the well-tested software distributed with the corresponding force fields (namely `tleap`[26] and `psfgen`[9]).

Listing 2: System building

```
# Solvate a Molecule object
mol = solvate(mopt)

# Build using default arguments
bmol = charmm.build(mol, outdir='./build')

# Override various options
topos  = charmm.defaultTopos() +
    ['./benzamidine.rtf']
params = charmm.defaultParams() +
    ['./benzamidine.prm']
caps = { 'A': ['first ACE', 'last CT3'],
         'B': ['none', 'none'] }
disu = [DisulfideBridge('A',321,'A',18),
        DisulfideBridge('B',  2,'B',26)]
cmol = charmm.build(mol, topo=topos,
    param=params, caps=caps, disulfide=disu,
    saltconc=0.15, saltcation='POT')

# Build for AMBER
topos = ['./benzamidine.prepi']
params = ['./benzamidine.frcmod']
amol = amber.build(mol, topo=topos,
    param=params, disulfide=disu,
    saltconc=0.15, saltcation='K+')
```

The default behavior of HTMD building functions, `charmm.build()` and `amber.build()`, is to (1) neutralize the system by automatically adding ions; (2) add neutral caps to protein terminals; and (3) automatically detect cysteine−cysteine (CYS−CYS) disulfide bonds and define them. Any of these automatic choices can be overridden by the user, who can also change force field versions, provide his own topology and parameter files, disable ionization or capping of proteins, provide a desired salt concentration, select ion types, manually specify disulfide bonds, and specify any modifications or patches supported by the underlying force field (see Listing 2). This flexibility is especially important for the correct modeling of post-translational modifications and nonstandard residues.

Both building functions accept an instance of the `Molecule` class as input and have the same interface. The user can therefore switch between force fields and simulation software-specific formats with minor modifications to the input parameters (see Listing 2).

**2.3. Protocols.** A successfully built system is ready for simulation. The typical simulation sequence, encompassing minimization, equilibration and production runs, is encoded in a set of protocols in HTMD. Protocols provide sensible default configurations as high-level procedures which can be used as-is, while still being flexible enough to be modified by advanced users for systems in which the default settings are not appropriate. An example usage of the protocols is given in Listing 3.

Listing 3: Equilibration and production runs using HTMD protocols.

```
# Set up an equilibration simulation for ACEMD
eq = Equilibration()
eq.runtime = 10
eq.timeunits = 'ns'
eq.temperature = 300
eq.write('./build', './equil')

# Run the equilibration on a local GPU
mdx = LocalGPUQueue()
mdx.submit('./equil')
mdx.wait()

# Set up a production simulation for ACEMD
md = Production()
md.runtime = 200
md.timeunits = 'ns'
md.temperature = 300
md.write('./equil', './prod')

# Run the production simulation on a local GPU
mdx.submit('./prod')
mdx.wait()
```

The equilibration protocol of HTMD consists of an initial energy minimization of 500 steps, followed by an equilibration run (Figure 1). During the first half of the equilibration, protein heavy atoms are constrained, to allow the rest of the system to equilibrate around the starting protein structure. Water molecules thus can reorganize and eventual protein side-chains clashes can be resolved. These constraints are gradually relaxed during the first half of the equilibration and the rest is run without constraints. Additionally, the first 500 steps of equilibration are run in the constant-volume (NVT) ensemble to account for changes in pressure due to the resolving of clashes. The rest of the equilibration is run in the constant-pressure (NPT) ensemble. The user is allowed to change the length of the equilibration runs, provide his own constraints


DOI: 10.1021/acs.jctc.7b00480
J. Chem. Theory Comput. 2017, 13, 4003−4011

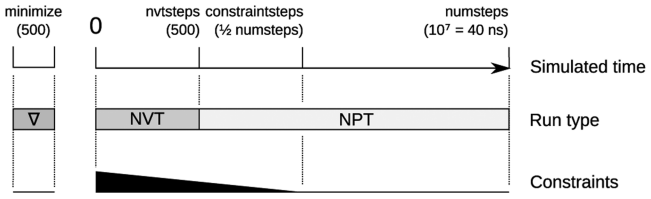



**Figure 1.** Time sequence scheme of the default equilibration protocol. A steepest descent minimization (∇) is followed by runs in the constant-volume (NVT) and constant-pressure (NPT) ensembles. Run times for the various steps are indicated by the corresponding keywords, with default lengths in parentheses (in simulation steps, usually 4 fs each).

and apply flat-bottom potentials to specific atoms to constrain their movement within a box. The production protocol provided by HTMD can be run either directly on a system provided by the user, or pick off where the equilibration protocol finished by using the produced output files. The production protocol then runs a standard NVT simulation, also allowing application of flat-bottom potentials.

**2.4. Simulation Queues: Abstracting Resources and Software from MD Runs.** Simulations can be run on a range of different infrastructures, from local computers, to remote clusters or even crowd-computing platforms. Additionally, MD simulations can be run using different MD engines and each of those engines can provide support for specific simulation devices such as CPUs and GPUs. Therefore, HTMD provides the `SimQueue` class which defines a unique interface for communicating with various computing resources. This allows users to change the computing resource they are using with minor modifications independent of the underlying MD software. More specifically, a `SimQueue` subclass handles all communication with a resource like a cluster or local queuing system. It exposes an interface consisting of three main methods, namely: `submit()` to handle the sending, queuing, and execution of simulations; `retrieve()`, which handles the retrieval of completed simulations from remote resources; and the `inprogress()` method which polls the queuing system for the number of currently running and queued simulations. Simulation queues can be run either asynchronously or synchronously using the `wait()` method which will block script execution until all queued simulations have completed. HTMD is packaged with `SimQueue` classes for running and queuing simulations, either locally (on single or multiple GPUs), on the Amazon EC2 cloud, on SLURM, or on LSF queuing systems. An example of the `LocalGPUQueue` class which implements a local queuing system for GPU simulations is shown in Listing 3. To keep the queuing systems independent from the MD software used for the simulation, they internally execute a `run.sh` script which calls the specific simulation software with any required options. This script is automatically created for ACEMD simulations through the protocols mentioned in section 2.3 but can also be provided by the user to run simulations using any desired MD software. In this way, extending support to other MD engines becomes as simple as writing a short script that executes the simulation stored in a given folder and then passing the folder to the `SimQueue.submit()` method.

## 3. RESULTS

To demonstrate the efficacy and power that a scriptable environment can provide for system building and simulation, we prepared and simulated most of the eukaryotic plasma membrane proteins available in the OPM database[39] in both CHARMM and AMBER force fields. The OPM database currently provides the predicted spatial arrangements of over 3000 membrane proteins, oriented with respect to the hydrophobic core of the bilayer. It provides PDB files, annotated with dummy atoms to indicate the position of the membrane. For this study we worked on a subset of 708 OPM proteins which are located in the eukaryotic plasma membrane, providing a comprehensive and realistic case for the design and test of an unsupervised automatic build-and-simulate protocol. This subset has grown since the start of the project, with more structures being added to the database ever since. We expect that the rest of the OPM can be built and simulated using the same protocol by substituting the membrane PDB file for the a membrane corresponding to the one the proteins are embedded in.

**3.1. Preparation Protocol.** The protocol used for automatically building and simulating all eukaryotic membrane proteins (Figure 2) consists of the following steps. First, all

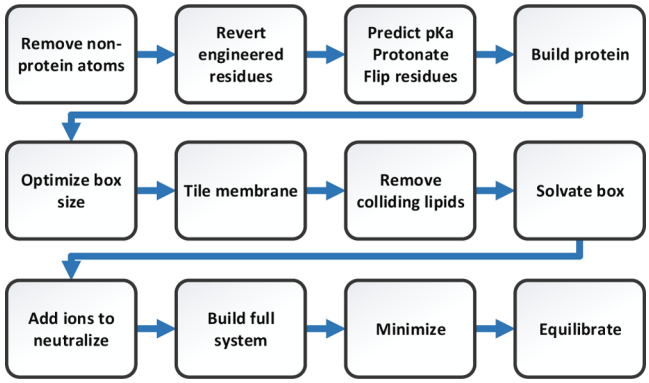



**Figure 2.** OPM building and simulation protocol workflow starting from a PDB file and ending with an equilibrated system.

nonprotein atoms are removed from the OPM PDB file and all engineered residues are mutated back to their parent residues according to the information automatically retrieved from the RCSB web site.[40] The reason for this is that the parameters of arbitrary ligands, cofactors, and engineered residues are not defined in the force fields or easily obtained. Protocols exist for the parametrization of ligands, but they are out of the scope of this paper; this step was omitted to decrease complexity and runtime. Next, the likely protonation states of the protein residues are calculated and assigned for pH 7, and the hydrogen bonding network is optimized as described in section 2.1. Disconnected protein segments are detected to allow terminal capping. Finally, the protein is built in the CHARMM and AMBER format to add the terminal caps, missing atoms, and residue side chains.

Missing protein loops are not modeled in this protocol. We recognize the importance of intra- and extra-cellular loops of many membrane proteins in both their stability and interactions with other molecules. Indeed, HTMD provides a easy-to-use wrapper for the Modeller[41] software allowing the users to model missing loops. However, we felt that the heterogeneity of structures and missing segments in the OPM did not allow yet for unsupervised reconstructions of missing regions.

To minimize the total system size, the protein is rotated around the *z*-axis to have its largest variation in the diagonal


DOI: 10.1021/acs.jctc.7b00480
J. Chem. Theory Comput. 2017, 13, 4003−4011

and thus best fit into the cubic simulation box. The simulation box is then created to the size of the protein plus 20 Å of additional space on the $x$ and $y$-axes and 5 Å in the $z$-axis to avoid interactions of the protein between periodic images. Of note, in most cases the user should use at least 10 Å of buffer (the default protocol value) along the $z$-axis to make sure that no self-interactions occur; in the example systems prepared in this study, a reduced 5 Å buffer was employed to keep the simulation time manageable.

Since we are working with a POPC pre-equilibrated membrane PDB files of fixed dimensions, the membrane is tiled along the $x$ and $y$ axes so as to cover the whole $x-y$ plane of the simulation box. Steric clashes between the membrane and the protein are resolved by removing lipid molecules having atoms within 1.3 Å of protein atoms.

Many proteins form pores inside the membrane, as in the case of ion channels, and the previous rule does not remove lipids inside those pores. Therefore, the protocol uses a convex hull method to detect lipids located inside the protein. It first calculates the convex hull of the protein atoms which are located inside the membrane and then sequentially tests each lipid if it is located inside this hull. Lipids located inside the hull are removed, thus clearing pores of lipids. Of note, more complex protocols exist for embedding a protein inside a membrane;[42,43] however, analysis of the built systems showed that even the simple embedding outlined above produces satisfactory results in terms of stability and area per lipid at the end of the equilibration (Figure S4).

The system is finally solvated with TIP3P water and neutralized with ions. The built system is minimized and equilibrated for 40 ns using ACEMD on the volunteer distributed computing resource GPUGRID,[44] following the protocol described in Figure 1. A Python script containing the building and simulation protocol can be found in the SI. For more information and statistics on the built systems, see section 1 of the SI.

**3.2. Building and Simulation Results.** As shown in Figure 3, out of the 708 OPM proteins, 699 were successfully built using the protocol for CHARMM and 691 for AMBER. Nine building errors for both force fields can be attributed to inconsistencies between nonstandard residue naming in the OPM database and the RCSB web site. The additional eight building errors in AMBER are mainly due the proteins containing deprotonated arginines, which are not supported in AMBER. The detailed list is provided in Table S1 (see the SI).

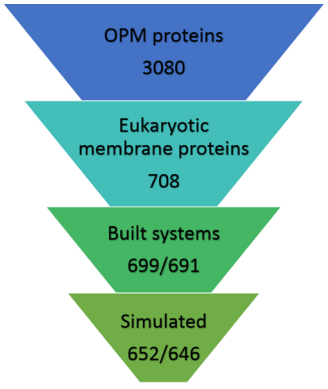



**Figure 3.** OPM currently consists of 3080 protein structures. From those we selected to build and simulate the 708 eukaryotic membrane proteins. Of those, 699 were successfully built for CHARMM, 691 for AMBER, and finally 652 were simulated in CHARMM and 646 in AMBER.

Out of the built systems, 652 and 646 were successfully simulated in CHARMM and AMBER, respectively. The two largest built and simulated systems can be seen in Figure 4. The

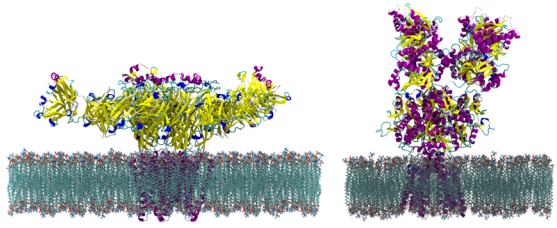


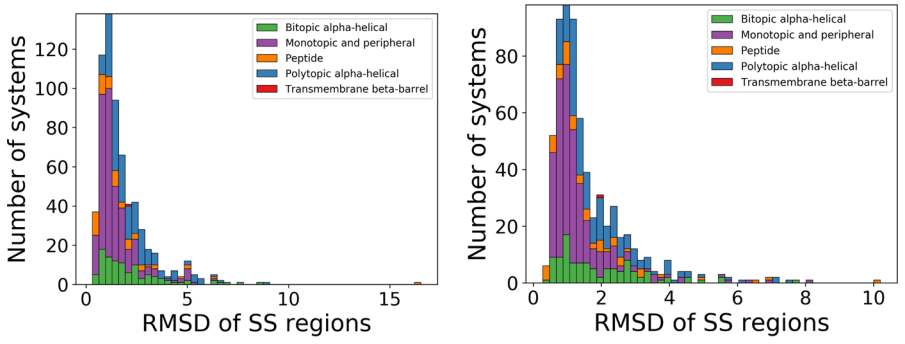

**Figure 4.** (a) Largest built system (3RHW) with 720 442 atoms. (b) Largest simulated system (4U1W) with 483 008 atoms.

47 and 45 respective simulation failures can be attributed to various causes. In some cases the systems were too large to simulate were thus discarded. In other cases, the failures were due to instabilities in the simulations such as interactions of the proteins with the membrane through periodic images in the $z$-axis or too large imbalances in lipid density in the two membrane layers. These are issues associated with the building protocol and the specific system and thus could be overcome with some minor modifications of the protocol, such as increasing the simulation box.

**3.3. Equilibration Analysis.** We calculated the root mean-squared distance (RMSD) from the initial configuration of all the successfully completed equilibration simulations to detect instabilities in the simulations. Figure 5 shows the distribution of RMSDs of the secondary structure regions of the proteins at the end of the equilibration simulations compared to the starting structure for the CHARMM and AMBER force fields. Loop regions were excluded from the calculation due to their high flexibility as well as due to NMR structures in the OPM containing extended loops. The total RMSD however does not change drastically if loops are included as can be seen in Figure S3 (see the SI). From these figures we can see that most proteins stayed quite stable during the equilibration, with small conformational changes mostly under 4 Å RMSD. Comparison between the two force fields shows very similar distributions with the exception of one outlier at 16 Å mentioned below.

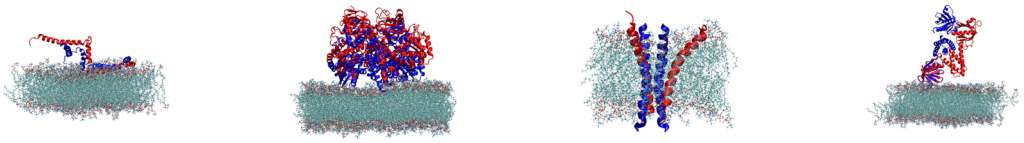
Inspection of the 26 CHARMM systems with RMSD over 5 Å reveals that most of them can be attributed to four main causes. First, loop regions in proteins function as hinges and allow a region of the protein which is not interacting strongly with the membrane to detach and flip to other conformations (e.g., Figure 6a). Second, some monotopic systems which are not embedded well in the membrane can detach totally from the membrane as can be seen in Figure 6b. Third, protein chains which are not bonded and are weakly interacting may separate from one another as seen in the example of two helices in the membrane in Figure 6c. Fourth, missing loops in PDB structures are not modeled by our protocol and thus the protein pieces before and after the missing loops are treated as separate chains which can detach from each other as in the case of Figure 6d which originally is a dimer but due to missing loops is modeled as four nonbonded chains. Membrane detachments as in Figure 6a can either be natural conforma-


DOI: 10.1021/acs.jctc.7b00480
J. Chem. Theory Comput. 2017, 13, 4003−4011




(a) CHARMM equilibration (b) AMBER equilibration

**Figure 5.** RMSD of the secondary structure regions of the last simulation frame after 40 ns of equilibration with respect to the starting configuration using the (a) CHARMM and (b) AMBER force fields.



(a) 4AXP 16.54 Å (b) 4BY0 6.22 Å (c) 2MEU 6.72 Å (d) 1U5E 6.40 Å

**Figure 6.** Four structures representing the most common causes of high RMSDs after equilibration. The blue conformation is the starting conformation, and red corresponds to the conformation after 40 ns of equilibration. (a) Hinged regions with little contact to the membrane detach from the membrane. (b) Weakly bound protein completely unbinds from the membrane. (c) Nonbonded helices tilt and separate in the membrane. (d) Missing loops cause separate chains to detach in bulk.

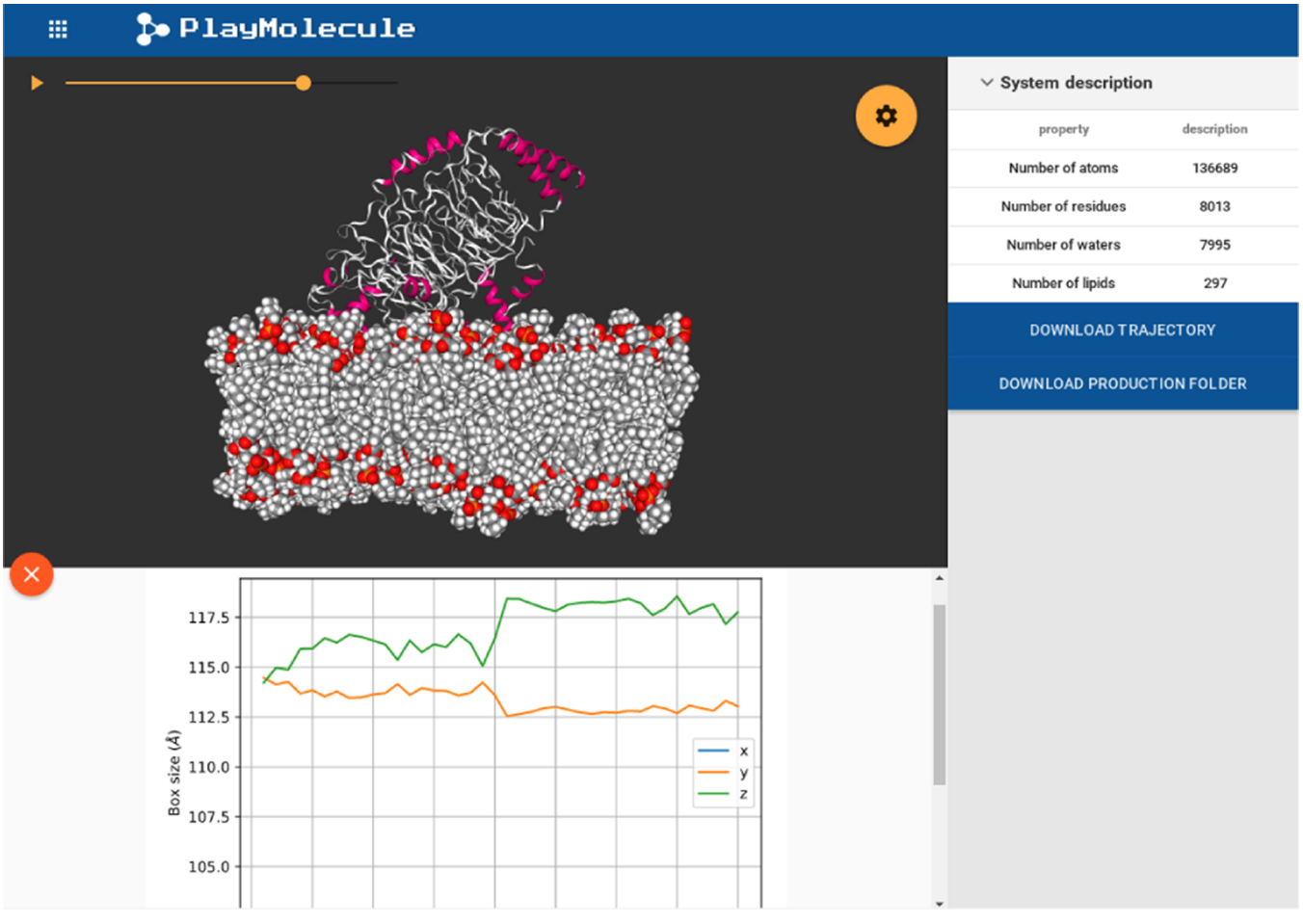



**Figure 7.** PlayMolecule service allows users to explore and download the equilibrated systems.



DOI: 10.1021/acs.jctc.7b00480
J. Chem. Theory Comput. 2017, 13, 4003−4011

tional changes of the proteins, issues in the OPM predictions, force field issues or problems in our building protocol as is probably the case with Figure 6b. However, in various outliers, the changes of RMSD seem like positive improvements to the original conformation where the protein moves to interact more stably with the membrane. In general, we note that not all of the structures deposited in the OPM may be suitable for membrane embedding in the first place, owing to a variety of reasons such as missing domains in the initial structure, limited lipophilic surface, low embedding, and so on; the automated identification of such edge cases is beyond the scope of this paper.

In conclusion, the successful automated simulation of more than 600 protein−membrane systems is reassuring with respect to the stability of systems produced by our building protocol. The complete data set presented in this paper is available on the www.playmolecule.org web service.[33] The web service utilizes the NGL viewer[45] for interactive trajectory visualization of the equilibration runs and shows RMSD plots for individual trajectories as seen in Figure 7. Additionally, it provides links for downloading the equilibration trajectory as well as a folder containing all files necessary to start production runs from the last frame of the equilibration using ACEMD. This way users can perform further analysis on individual systems or use them as starting points for their own simulations.

## 4. CONCLUSION

HTMD provides a unified framework for MD based discovery. It currently allows structural manipulation as well as all necessary functionality such as protonation, solvation, ionization, capping and more, for building systems in both CHARMM and AMBER formats. Additionally, through the use of predefined protocols and `SimQueues` it simplifies the simulation setup and execution on a variety of computational resources. Options are available to the user at every step to accommodate both starting and advanced users.

In this work, the flexibility of HTMD is demonstrated by building and simulating most eukaryotic membrane proteins of the OPM. Many improvements are still possible on the given protocol. Protein−ligand interactions can be important for the stability of the membrane proteins. Parametrization of ligands found bound to the protein in PDB files, or added in silico, would allow us to simulate them together with the rest of the system and monitor the dynamics between the proteins and the ligands; automated ligand parametrization is a planned extension to the HTMD and PlayMolecule suites. Unresolved loops of proteins can also be modeled automatically by integrating a loop-modeling software like Modeller[41] or Rosetta[46] into the pipeline, improving the protein models such as in the case of 1U5E. The following step would then be to build and simulate the remaining noneukaryotic OPM proteins using their corresponding membrane compositions.

The demonstrated flexibility, simplicity, and power of the software accommodates users with a wide range of expertises allowing them to apply MD to their problems with minimal coding. The integration of all necessary tools of the building and simulation pipeline into a single framework increases reproducibility and allows the easy automation of high-throughput simulations as in the example given of the OPM proteins where it is able to build hundreds of different proteins each with its own peculiarities in an automated and timely manner for two different force fields. We expect these developments to open the gates to high-throughput MD to more scientists, allow for larger scale MD-based discovery, and increase reproducibility, thus allowing for more rapid progress in this field.

## ■ ASSOCIATED CONTENT

### Supporting Information

The Supporting Information is available free of charge on the ACS Publications website at DOI: 10.1021/acs.jctc.7b00480.

Python script `opmbuild.py` together with a membrane PDB file `popc36_box.pdb` and a small test protein `2bau.pdb` for setting up and simulating an OPM system. Figures for statistics on the OPM data set used, the positioning of proteins in the membrane, explanations for all building errors, the total RMSD of the last simulation frame, and the area per lipid distribution (ZIP)

## ■ AUTHOR INFORMATION


### Corresponding Author

*E-mail: gianni.defabritiis@upf.edu.

### ORCID

Stefan Doerr: 0000-0002-8678-8657
Toni Giorgino: 0000-0001-6449-0596
Gerard Martínez-Rosell: 0000-0001-6277-6769
João M. Damas: 0000-0003-3454-2572
Gianni De Fabritiis: 0000-0003-3913-4877


### Author Contributions

‖S.D. and T.G. contributed equally to this work.

### Notes

The authors declare the following competing financial interest(s): S.D., G.D.F., and J.M.D. are funded by Acellera Ltd which commercializes the HTMD software for non-academic users.

## ■ ACKNOWLEDGMENTS

We thank Acellera Ltd for funding. G.D.F. acknowledges support from MINECO (BIO2014-53095-P) and FEDER. We thank the volunteers of GPUGRID for donating their computing time for the simulations. We would also like to thank Matt J. Harvey for helpful contributions to HTMD.

## ■ REFERENCES


(1) Lindorff-Larsen, K.; Piana, S.; Dror, R. O.; Shaw, D. E. How Fast-Folding Proteins Fold. *Science* **2011**, *334*, 517−520.

(2) Jensen, M. O.; Jogini, V.; Borhani, D. W.; Leffler, A. E.; Dror, R. O.; Shaw, D. E. Mechanism of voltage gating in potassium channels. *Science* **2012**, *336*, 229−233.

(3) Buch, I.; Giorgino, T.; De Fabritiis, G. Complete reconstruction of an enzyme-inhibitor binding process by molecular dynamics simulations. *Proc. Natl. Acad. Sci. U. S. A.* **2011**, *108*, 10184−10189.

(4) Kohlhoff, K. J.; Shukla, D.; Lawrenz, M.; Bowman, G. R.; Konerding, D. E.; Belov, D.; Altman, R. B.; Pande, V. S. Cloud-based simulations on Google Exacycle reveal ligand modulation of GPCR activation pathways. *Nat. Chem.* **2013**, *6*, 15−21.

(5) Stanley, N.; Esteban-Martín, S.; de Fabritiis, G. Kinetic modulation of a disordered protein domain by phosphorylation. *Nat. Commun.* **2014**, *5*, 5272.

(6) Reubold, T. F.; Faelber, K.; Plattner, N.; Posor, Y.; Ketel, K.; Curth, U.; Schlegel, J.; Anand, R.; Manstein, D. J.; Noé, F.; Haucke, V.; Daumke, O.; Eschenburg, S. Crystal structure of the dynamin tetramer. *Nature* **2015**, *525*, 404−408.




DOI: 10.1021/acs.jctc.7b00480
J. Chem. Theory Comput. 2017, 13, 4003−4011

(7) Harvey, M. J.; Giupponi, G.; De Fabritiis, G. ACEMD: Accelerating Biomolecular Dynamics in the Microsecond Time Scale. *J. Chem. Theory Comput.* **2009**, *5*, 1632−1639.
(8) Abraham, M. J.; Murtola, T.; Schulz, R.; Páll, S.; Smith, J. C.; Hess, B.; Lindahl, E. GROMACS: High performance molecular simulations through multi-level parallelism from laptops to supercomputers. *SoftwareX* **2015**, *1*−*2*, 19−25.
(9) Phillips, J. C.; Braun, R.; Wang, W.; Gumbart, J.; Tajkhorshid, E.; Villa, E.; Chipot, C.; Skeel, R. D.; Kalé, L.; Schulten, K. Scalable molecular dynamics with NAMD. *J. Comput. Chem.* **2005**, *26*, 1781−1802.
(10) Eastman, P.; Friedrichs, M. S.; Chodera, J. D.; Radmer, R. J.; Bruns, C. M.; Ku, J. P.; Beauchamp, K. A.; Lane, T. J.; Wang, L.-P.; Shukla, D.; Tye, T.; Houston, M.; Stich, T.; Klein, C.; Shirts, M. R.; Pande, V. S. OpenMM 4: A Reusable, Extensible, Hardware Independent Library for High Performance Molecular Simulation. *J. Chem. Theory Comput.* **2013**, *9*, 461−469.
(11) Bowers, K. J.; Chow, E.; Xu, H.; Dror, R. O.; Eastwood, M. P.; Gregersen, B. A.; Klepeis, J. L.; Kolossvary, I.; Moraes, M. A.; Sacerdoti, F. D.; Salmon, J. K.; Shan, Y.; Shaw, D. E. Scalable algorithms for molecular dynamics simulations on commodity clusters. *Proceedings of the 2006 ACM/IEEE conference on Supercomputing*, Tampa, Florida, 2006; p 84.
(12) Zimmermann, M. T.; Urrutia, R.; Oliver, G. R.; Blackburn, P. R.; Cousin, M. A.; Bozeck, N. J.; Klee, E. W.; Caldwell, J.; Kollman, P. Molecular modeling and molecular dynamic simulation of the effects of variants in the TGFBR2 kinase domain as a paradigm for interpretation of variants obtained by next generation sequencing. *PLoS One* **2017**, *12*, e0170822.
(13) van der Kamp, M. W.; Schaeffer, R. D.; Jonsson, A. L.; Scouras, A. D.; Simms, A. M.; Toofanny, R. D.; Benson, N. C.; Anderson, P. C.; Merkley, E. D.; Rysavy, S.; Bromley, D.; Beck, D. A.; Daggett, V. Dynameomics: A Comprehensive Database of Protein Dynamics. *Structure* **2010**, *18*, 423−435.
(14) Meyer, T.; D'Abramo, M.; Hospital, A.; Rueda, M.; Ferrer-Costa, C.; Pérez, A.; Carrillo, O.; Camps, J.; Fenollosa, C.; Repchevsky, D.; Gelpí, J. L.; Orozco, M. MoDEL (Molecular Dynamics Extended Library): A Database of Atomistic Molecular Dynamics Trajectories. *Structure* **2010**, *18*, 1399−1409.
(15) De Vivo, M.; Masetti, M.; Bottegoni, G.; Cavalli, A. Role of Molecular Dynamics and Related Methods in Drug Discovery. *J. Med. Chem.* **2016**, *59*, 4035−4061.
(16) Borhani, D. W.; Shaw, D. E. The future of molecular dynamics simulations in drug discovery. *J. Comput.-Aided Mol. Des.* **2012**, *26*, 15−26.
(17) Martínez-Rosell, G.; Giorgino, T.; Harvey, M. J.; de Fabritiis, G. Drug Discovery and Molecular Dynamics: Methods, Applications and Perspective Beyond the Second Timescale. *Curr. Top. Med. Chem.* **2017**, *17*, 1−1.
(18) Doerr, S.; Harvey, M. J.; Noé, F.; De Fabritiis, G. HTMD: High-Throughput Molecular Dynamics for Molecular Discovery. *J. Chem. Theory Comput.* **2016**, *12*, 1845−1852.
(19) *Schrödinger Release 2016-1*; Maestro, 2016.
(20) Krieger, E.; Vriend, G. YASARA View - molecular graphics for all devices - from smartphones to workstations. *Bioinformatics* **2014**, *30*, 2981−2982.
(21) *Molecular Operating Environment (MOE)*; 2016.
(22) Lee, J.; Cheng, X.; Swails, J. M.; Yeom, M. S.; Eastman, P. K.; Lemkul, J. A.; Wei, S.; Buckner, J.; Jeong, J. C.; Qi, Y.; Jo, S.; Pande, V. S.; Case, D. A.; Brooks, C. L.; MacKerell, A. D.; Klauda, J. B.; Im, W. CHARMM-GUI Input Generator for NAMD, GROMACS, AMBER, OpenMM, and CHARMM/OpenMM Simulations Using the CHARMM36 Additive Force Field. *J. Chem. Theory Comput.* **2016**, *12*, 405−413.
(23) Hospital, A.; Andrio, P.; Fenollosa, C.; Cicin-Sain, D.; Orozco, M.; Gelpí, J. L. MDWeb and MDMoby: an integrated web-based platform for molecular dynamics simulations. *Bioinformatics* **2012**, *28*, 1278−1279.
(24) Schwieters, C. D.; Kuszewski, J. J.; Tjandra, N.; Clore, G. M. The Xplor-NIH NMR molecular structure determination package. *J. Magn. Reson.* **2003**, *160*, 65−73.
(25) Brünger, A. T.; Adams, P. D.; Clore, G. M.; DeLano, W. L.; Gros, P.; Grosse-Kunstleve, R. W.; Jiang, J. S.; Kuszewski, J.; Nilges, M.; Pannu, N. S.; Read, R. J.; Rice, L. M.; Simonson, T.; Warren, G. L. Crystallography & NMR system: A new software suite for macromolecular structure determination. *Acta Crystallogr., Sect. D: Biol. Crystallogr.* **1998**, *54*, 905−21.
(26) Case, D.; Betz, R.; Botello-Smith, W.; Cerutti, D.; Cheatham, T.; Darden, T.; Duke, R.; Giese, T.; Gohlke, H.; Goetz, A.; Homeyer, N.; Izadi, S.; Janowski, P.; Kaus, J.; Kovalenko, A.; Lee, T.; LeGrand, S.; Lin, C.; Luchko, T.; Luo, R.; Madej, B.; Mermelstein, D.; Merz, K.; Monard, G.; Nguyen, H.; Nguyen, H.; Omelyan, I.; Onufriev, A.; Roe, D.; Roitberg, A.; Sagui, C.; Simmerling, C.; Swails, J.; Walker, R.; Wang, J.; Wolf, R.; Wu, X.; Xiao, L.; York, D.; Kollman, P. *AMBER 2016*; 2016.
(27) Parton, D. L.; Grinaway, P. B.; Hanson, S. M.; Beauchamp, K. A.; Chodera, J. D. Ensembler: Enabling High-Throughput Molecular Simulations at the Superfamily Scale. *PLoS Comput. Biol.* **2016**, *12*, e1004728.
(28) Shi, Y.; Xia, Z.; Zhang, J.; Best, R.; Wu, C.; Ponder, J. W.; Ren, P. Polarizable Atomic Multipole-Based AMOEBA Force Field for Proteins. *J. Chem. Theory Comput.* **2013**, *9*, 4046−4063.
(29) RDKit: Open-source cheminformatics. http://www.rdkit.org (accessed on July 30, 2017).
(30) Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; Vanderplas, J.; Passos, A.; Cournapeau, D.; Brucher, M.; Perrot, M.; Duchesnay, E. Scikit-learn: Machine Learning in Python. *J. Mach. Learn. Res.* **2011**, *12*, 2825−2830.
(31) McKinney, W. Data Structures for Statistical Computing in Python. *Proceedings of the 9th Python in Science Conference*, 2010; pp 51−56.
(32) Chen, W.; Morrow, B. H.; Shi, C.; Shen, J. K. Recent development and application of constant pH molecular dynamics. *Mol. Simul.* **2014**, *40*, 830−838.
(33) Martinez-Rosell, G.; Giorgino, T.; De Fabritiis, G. PlayMolecule ProteinPrepare: A Web Application for Protein Preparation for Molecular Dynamics Simulations. *J. Chem. Inf. Model.* **2017**, *57*, 1511.
(34) Olsson, M. H. M.; Søndergaard, C. R.; Rostkowski, M.; Jensen, J. H. PROPKA3: Consistent Treatment of Internal and Surface Residues in Empirical pKa Predictions. *J. Chem. Theory Comput.* **2011**, *7*, 525−537.
(35) Søndergaard, C. R.; Olsson, M. H. M.; Rostkowski, M.; Jensen, J. H. Improved Treatment of Ligands and Coupling Effects in Empirical Calculation and Rationalization of pKa Values. *J. Chem. Theory Comput.* **2011**, *7*, 2284−2295.
(36) Dolinsky, T. J.; Czodrowski, P.; Li, H.; Nielsen, J. E.; Jensen, J. H.; Klebe, G.; Baker, N. A. PDB2PQR: expanding and upgrading automated preparation of biomolecular structures for molecular simulations. *Nucleic Acids Res.* **2007**, *35*, W522−W525.
(37) Dolinsky, T. J.; Nielsen, J. E.; McCammon, J. A.; Baker, N. A. PDB2PQR: an automated pipeline for the setup of Poisson-Boltzmann electrostatics calculations. *Nucleic Acids Res.* **2004**, *32*, W665−W667.
(38) Teixeira, V. H.; Vila-Viçosa, D.; Reis, P. B. P. S.; Machuqueiro, M. pKa Values of Titrable Amino Acids at the Water/Membrane Interface. *J. Chem. Theory Comput.* **2016**, *12*, 930−934.
(39) Lomize, M. A.; Lomize, A. L.; Pogozheva, I. D.; Mosberg, H. I. OPM: Orientations of Proteins in Membranes database. *Bioinformatics* **2006**, *22*, 623−625.
(40) Berman, H. M.; Westbrook, J.; Feng, Z.; Gilliland, G.; Bhat, T. N.; Weissig, H.; Shindyalov, I. N.; Bourne, P. E. The Protein Data Bank. *Nucleic Acids Res.* **2000**, *28*, 235−242.
(41) Fiser, A.; Do, R. K.; Sali, A. Modeling of loops in protein structures. *Protein Sci.* **2000**, *9*, 1753−1773.
(42) Wolf, M. G.; Hoefling, M.; Aponte-Santamaría, C.; Grubmüller, H.; Groenhof, G. g_membed: Efficient insertion of a membrane


DOI: 10.1021/acs.jctc.7b00480
*J. Chem. Theory Comput.* 2017, 13, 4003−4011

protein into an equilibrated lipid bilayer with minimal perturbation. *J. Comput. Chem.* **2010**, *31*, 2169−2174.
(43) Kandt, C.; Ash, W. L.; Peter Tieleman, D. Setting up and running molecular dynamics simulations of membrane proteins. *Methods* **2007**, *41*, 475−488.
(44) Buch, I.; Harvey, M. J.; Giorgino, T.; Anderson, D. P.; De Fabritiis, G. High-throughput all-atom molecular dynamics simulations using distributed computing. *J. Chem. Inf. Model.* **2010**, *50*, 397−403.
(45) Rose, A. S.; Hildebrand, P. W. NGL Viewer: a web application for molecular visualization. *Nucleic Acids Res.* **2015**, *43*, W576−9.
(46) Rohl, C. A.; Strauss, C. E. M.; Misura, K. M. S.; Baker, D. Protein structure prediction using Rosetta. *Methods Enzymol.* **2004**, *383*, 66−93.



DOI: 10.1021/acs.jctc.7b00480
*J. Chem. Theory Comput.* 2017, 13, 4003−4011