JCTC Journal of Chemical Theory and Computation

Article

pubs.acs.org/JCTC

Alchembed: A Computational Method for Incorporating Multiple Proteins into Complex Lipid Geometries

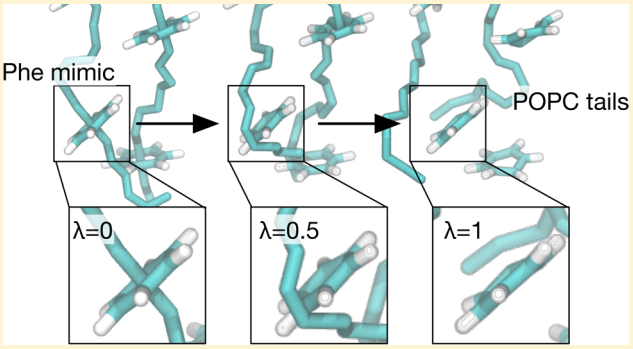
Elizabeth Jefferys,[†] Zara A. Sands,[‡] Jiye Shi,[‡] Mark S. P. Sansom,[†] and Philip W. Fowler*[,†]

[†]Department of Biochemistry, University of Oxford, Oxford OX1 3QU, United Kingdom

[‡]UCB NewMedicines, Chemin du Foriest, 1420 Braine-l'Alleud, Belgium

Supporting Information

ABSTRACT: A necessary step prior to starting any membrane protein computer simulation is the creation of a well-packed configuration of protein(s) and lipids. Here, we demonstrate a method, *alchembed*, that can simultaneously and rapidly embed multiple proteins into arrangements of lipids described using either atomistic or coarse-grained force fields. During a short simulation, the interactions between the protein(s) and lipids are gradually switched on using a soft-core van der Waals potential. We validate the method on a range of membrane proteins and determine the optimal soft-core parameters required to insert membrane proteins. Since all of the major biomolecular codes include soft-core van der Waals potentials, no additional code is required to apply this method. A tutorial is included in the Supporting Information.



INTRODUCTION

Membrane proteins are crucial players in a wide range of important cellular processes, from signaling to controlling the movement of ions and molecules into and out of cells; consequently, over 60% of drug targets are membrane proteins.[1] They are categorized depending on the strength and nature of their interaction with the cell membrane: peripherial membrane proteins (PMPs) bind only transiently to cell membranes, whereas integral membrane proteins (IMPs) are firmly inserted into the cell membrane. The latter may be further divided into those that associate only with one leaflet of the lipid bilayer (monotopic IMPs) and those that span the whole width of the lipid bilayer (polytopic IMPs).

Computer simulation of membrane proteins has proven to be a useful counterpart to experiment.[2−4] The first membrane protein molecular dynamics (MD) simulation was run 30 years ago[5] and comprised a single gramicidin channel surrounded by 16 DMPC lipids. The dynamics of all 4390 atoms were simulated for 0.6 ns. Since then, there have been huge improvements in the speed of central processing units (CPUs), the ability of simulation codes to use multiple CPUs to run a single simulation, and the accuracy of the atomistic (AT) force fields, including the introduction of additional lipids. The recent introduction of coarse-grained (CG) force fields, such as MARTINI,[6] has allowed longer and larger simulations to be run. In particular, CG simulations have enabled the simulation of mixtures of lipids, often leading to more complex geometries, such as undulating membrane patches, whole vesicles, or virus capsids.[7−14]

When setting up a simulation of a membrane protein(s), one is always faced with a problem: how does one embed the protein(s) in the lipids? This is really three distinct problems. First, one must decide how to orient and position the membrane protein relative to the lipids (the orientation problem); then, one must delete an appropriate number of lipids, if necessary, (the deletion problem) before finally inserting the protein(s) between the remaining lipids (the insertion problem). We shall focus on the insertion problem in this article. The orientation problem is difficult, and there is often little or no experimental data to guide how the protein should be placed relative to the bilayer; this is especially true for monotonic IMPs. Two common approaches are either to use the fact that the propensities of different amino acids vary with the distance from the center of the bilayer[15] or to run a coarse-grained simulation to self-assemble lipids around a single copy of the membrane protein.[16] In some cases, brute-force coarse-grained simulations can correctly predict the orientation of peripheral membrane proteins.[17]

A successful method must allow a user to easily and reproducibly insert any type of membrane protein into any arrangement of lipids, ideally in a high-throughput manner. It must be able to cope with the different varieties of membrane proteins and be able to simultaneously insert multiple membrane proteins. Bearing in mind the recent improvements in simulation methodology mentioned above, it also needs to be able to handle complex lipid geometries such as vesicles,

Received: December 9, 2014
Published: May 14, 2015

DOI: 10.1021/ct501111d
J. Chem. Theory Comput. 2015, 11, 2743−2754

viral envelopes, or large undulating patches of bilayer. The method must also be able to handle both atomistic and coarse-grained levels of representation. Finally, it should not affect either the structure of the protein or the overall arrangement and composition of the lipids.

A range of approaches has been proposed (reviewed elsewhere[18−22]). These broadly divide into two categories. The first assembles lipids one-by-one around the membrane protein, usually forming a small patch of bilayer, as was done in the first simulation of a membrane protein.[5,23] This method has been recently reinvigorated by the introduction of the CHARMM-GUI[24] and the use of CG simulations to self-assemble a lipid bilayer around a membrane protein[16] and then converting the result to atomistic coordinates.[25,26] The main disadvantages of this category of methods are that the final arrangement of lipids may need extensive simulation to become well-equilibrated and it is limited to simple, small patches of lipid bilayer, probably formed of only a single lipid species. A recent program, LipidWrapper, can build more complex lipid geometries, but it does not yet help the user insert proteins into the lipids.[27]

The second category takes the opposite approach and, starting from a preformed ensemble of lipids (such as a bilayer), deletes the overlapping lipids so the membrane protein fits.[28] Due to the tendency of lipid tails to become entangled, this method tends to result in too many lipids being removed, leading to a gap between the protein and the lipids and hence poor packing. More sophisticated methods first delete only the lipids with the highest degree of overlap before applying forces to the remaining overlapping lipids to gently move them away from the protein. The first implementation of this approach simply applied a weak cylindrical force centered on the protein,[29] which works well on cylindrical proteins, like pores and channels. Subsequent methods better matched the forces to the shape of the protein.[30,31] A distantly related method applies forces to the lipids indirectly by instructing the barostat to apply a high pressure, which forces the protein into the bilayer.[32] More subtly, one can expand the spacing between the lipids, insert the protein into a gap, and then gradually reduce the spacing between the molecules, allowing time for rearrangement.[18,33] Alternatively, one can shrink the protein, insert it among the lipids, and then gradually grow it back, allowing time for the lipids to "feel" the presence of the protein and move out of the way.[34] This last approach, g_membed, is included with the GROMACS molecular dynamics package[35] and is widely used. To avoid perturbing the lipids, only the coordinates of the protein in the plane of the bilayer are shrunk and therefore, as with many of these methods, the z axis is identified as the membrane normal.

There is a key limitation that is common to all of the methods mentioned above: they cannot be used to simultaneously insert multiple proteins into complex arrangements of lipids where the membrane normal varies. The canonical example of this is inserting multiple membrane proteins into a lipid vesicle. In addition, these methods are often tested only on integral polytopic membrane proteins, so it is also unclear how well they perform on integral monotonic and peripheral membrane proteins.

In this article, we shall demonstrate how soft-core van der Waals potentials, first introduced in free energy calculations,[36] may be used to simultaneously and rapidly embed multiple membrane proteins of a variety of types in lipid bilayers and a lipid vesicle. Since all of the major biomolecular simulation packages now include soft-core van der Waals potentials, no additional code is required: one simply activates the relevant options already present in the code. We call this procedure *alchembed*.

RESULTS

Underlying Theory. Consider inserting a membrane protein into a computational model of a lipid system by linearly scaling the nonbonded interactions between the protein and the rest of the system. This is difficult for two reasons, the first of which is that the Lennard-Jones 6−12 potential, $U_B(r)$, that models the van der Waals interactions and gives all of the particles spatial extent is hard due to the r^{12} term.

$$U_B(r) = 4\lambda\varepsilon\left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right] \quad (1)$$

where r is the distance between two atoms (or beads), λ is the linear scaling term, σ is the value of r when the potential is zero, and ε is the depth of the potential well. Linearly scaling this function does not much reduce the width of the singularity (Figure 1A). Second, biomolecular systems are very densely packed. If you randomly place an atom (or bead) into a simulation box containing a lipid bilayer, then the probability of there being another atom (or bead) within σ is very nearly unity (Figure S1). This ensures that as soon as λ is nonzero there will more than likely be a steric clash between the inserted molecule and the remainder of the system, leading to very high forces and the probable failure of the simulation.

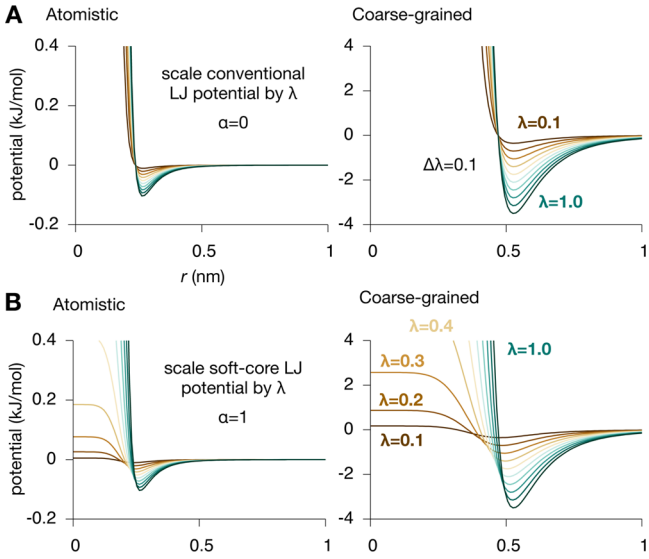


Figure 1. Linearly scaling a Lennard-Jones 6−12 potential is abrupt, and the singularity remains except when $\lambda = 0$. A soft-core van der Waals potential is much more smooth. (A) How a simple Lennard-Jones 6−12 potential (eq 1) between a leucine side chain and a saturated lipid tail varies with λ for both the fully atomistic CHARMM force field (σ = 0.24 nm and ε = 0.10 kJ mol^{-1}) and the coarse-grained MARTINI force field (σ = 0.47 nm and ε = 3.5 kJ mol^{-1}). Ten values of λ are plotted, λ = 0.1, 0.2, 0.3, ..., 1.0. (B) The same analysis is repeated for a soft-core van der Waals potential (eq 3) with $a = b = 1$, $c = 6$, and $\alpha = 1$.

One route to solving this problem is to use a so-called soft-core van der Waals potential.[36−38] Pham and Shirts[39] introduced a general functional form

DOI: 10.1021/ct501111d
J. Chem. Theory Comput. 2015, 11, 2743−2754

$$U_B(r,\lambda) = 4\epsilon\lambda^a\left[\left(\frac{1}{\alpha(1-\lambda)^b + (r/\sigma)^c}\right)^{12/c} - \left(\frac{1}{\alpha(1-\lambda)^b + (r/\sigma)^c}\right)^{6/c}\right] \quad (2)$$

where α is a positive constant that controls the softness of the potential and a, b, and c are positive exponents that control how quickly the softness is removed. All existing implementations of soft-core potentials can be described by combinations of these parameters. For example, if $a = 1$, $b = 2$, and $c = 6$, then we recover the form proposed by Beutler et al.;[38] setting $b = 1$ instead gives the form implemented in AMBER[40] and GROMACS[35] (although b may also be set to 2 in the latter code)

$$U_B(r,\lambda) = 4\epsilon\lambda\left[\frac{1}{(\alpha(1-\lambda) + (r/\sigma)^6)^2} - \frac{1}{\alpha(1-\lambda) + (r/\sigma)^6}\right] \quad (3)$$

If $a = b = 1$ and $c = 2$, then we recover the form proposed by Zacharias et al.,[37] which is implemented in both NAMD[41] and CHARMM.[42] In all four of these MD codes, α is a free parameter, so it can be varied.

Further improvements continue to be suggested,[39,43] but we shall constrain ourselves to these forms so that our approach may be applied using all of the major codes. In this article, we shall limit ourselves to using GROMACS since it offers the greatest flexibility in varying the form of the soft-core potential and can be easily used to run simulations using atomistic and coarse-grained force fields.

The effect of varying λ on the potential given by eq 3 is shown in Figure 1B. We have chosen $\alpha = 1$ for these examples since this value clearly illustrates the softness of the potential. In both cases, when $\lambda = 0$, the van der Waals potential is zero everywhere and hence there could be significant overlap between protein and lipid, resulting in multiple steric clashes, but the simulation would not crash since neither species would feel any force due to the other. As we gradually increase λ from 0 to 1, thereby switching on the van der Waals forces between the two species, then we can see how, unlike with a regular hard-core potential, the soft-core potential smoothly increases with no discontinuities. The lipids would, in principle, experience only a moderate force as λ is increased, resulting in them smoothly and gradually moving aside, making room for the protein and leading to good protein−lipid packing. Increasing α slows the rate at which the potential grows (Figure S2): too small a value of α and the transition occurs very rapidly when λ is small, too large, and the potential changes hugely in the last few increments of λ. Since the form of the soft-core potential implemented in NAMD and CHARMM is slightly different to the one in GROMACS and AMBER, the speed at which the van der Waals potential is switched on is also different (Figure S3), although the same broad behaviors are observed.

A Challenging Test: Disentangling Phenylalanine Side Chains from Lipid Tails. Let us first consider a challenging test case to demonstrate the validity of our approach. When one inserts a membrane protein into a preformed bilayer, one of the lipid tails may, on occasion, pass through the center of a protein aromatic group, for example, a phenylalanine side chain. This is clearly unphysical, but, due to the length of the lipid tails, it is difficult for the lipid to disentangle itself from the ring. A pathological example of this is shown in Figure 2A, where

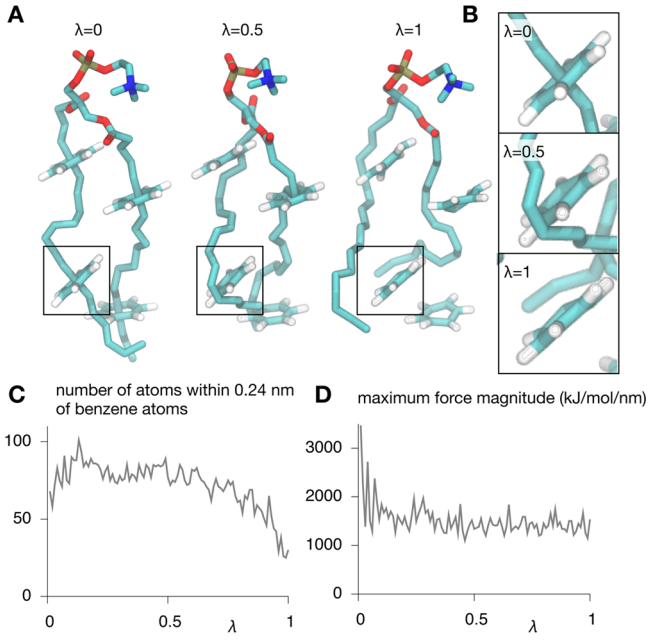


Figure 2. Gradually increasing the van der Waals interactions (described using a soft-core potential) between the four benzene molecules (modeling the side chain of phenylalanine) and the rest of the system allows the benzenes to become disentangled from the tails of the lipid. (A) Snapshots of the entangled lipid and the four benzene molecules are shown at $\lambda = 0$, 0.5, and 1. For clarity, neither the hydrogens on the lipid nor the remainder of the system are drawn. (B) The disentanglement of one of the benzene molecules is shown at a higher magnification. (C) The number of lipid and water atoms within 0.24 nm of any benzene atom. (D) How the magnitudes of the maximum force experienced by the lipids and waters change as the benzenes are embedded into the system. The soft-core parameters used throughout were $\alpha = 1$, $a = b = 1$, and $c = 6$. The simulation was 1000 steps long and took 190 s on a single core of an Intel Xeon E5 processor.

four benzene rings, representing four phenylalanine side chains, have been positioned around the tails of a single lipid in the middle of an equilibrated bilayer of 128 POPC lipids. The degree of steric clash is such that any molecular dynamics simulation would fail due to the resulting extremely high forces. Although unusual, such situations can be encountered when inserting membrane proteins into lipid bilayers, so they provide a good test of any method. We first minimized the energy of the system before introducing a soft-core van der Waals potential between the benzene molecules and the rest of the system. A harmonic restraint is also applied to the benzene atoms to ensure that they remain near their starting positions. A single molecular dynamics simulation of 1000 steps was then run during which λ was linearly increased from 0.0 to 1.0 in steps of 0.001 with $\alpha = 1$; therefore, the strength of the van der Waals interactions between the benzene molecules and the lipids and waters increases as shown in Figure 1B. During the course of the simulation, the lipid tails passed through the benzene molecules (Figure 2B).

At first, there is an increase in the number of lipid and waters atoms within σ (0.24 nm) of any benzene atoms, but as λ, and

DOI: 10.1021/ct501111d
J. Chem. Theory Comput. 2015, 11, 2743−2754

hence the strength of the interaction, increases, the number of lipid and waters within 0.24 nm starts to fall as they begin to feel the van der Waals potential and move out of the way of the benzene molecules (Figure 2C). During a molecular dynamics simulation, the maximum force that an atom (or bead) can experience without the simulation failing is effectively constrained by the size of the integration time step (here 0.5 fs). Following the maximum force as a function of λ is hence a useful way of checking that our insertion protocol is behaving correctly (Figure 2D). We observe that the magnitude of the largest force generated is only 4× the average; hence, the soft-core van der Waals potential ensures that the simulation does not fail.

Determining an Optimal Set of Parameters. Next, we systematically tested the *alchembed* method on five different membrane proteins (Table 1 and Figure 3), namely, phospholipase A2, N-BAR, cyclooxygenase 1, KcsA, and OmpF. These include examples of peripheral and (both mono- and polytopic) integral membrane proteins. We investigated the effect of modifying α and N since these can be varied in all of the major molecular dynamics codes; a and c cannot, so they are assumed to be 1 and 6 throughout. We only briefly investigated the effect of altering b since, although this can be changed in GROMACS, it cannot be altered in the other codes. The value of α was varied by 5 orders of magnitude (0.0001 to 10) in multiples of $10^{1/2}$, whereas N was taken to be either 1000 or 10 000 (which implies $\delta\lambda$ = 0.001 or 0.0001, respectively) and b was set to either 1 or 2. Lastly, each system was described using both fully atomistic (CHARMM) and coarse-grained (MARTINI) force fields. Repeat simulations were run as described.

Table 1. Five Test Membrane Proteins[a]

type of membrane protein	name	PDB code	number of atoms (AT)	number of beads (CG)
peripherial membrane proteins	Phospholipase A2	1P2P[44]	154 897	13 882
	N-BAR	2RND[45]	126 854	13 797
monotopic integral membrane proteins	Cyclooxygenase 1	1PRH[46]	194 871	17 632
polytopic integral membrane proteins	KcsA	1K4C[47]	125 081	11 376
	OmpF	4GCS[48]	121 237	11 495

[a]All were embedded in a simple lipid bilayer of 512 POPC molecules and were tested in both atomistic and coarse-grained representations. The number of atoms (or beads) is for the full system, including waters. Images of the five proteins are in Figure 3.

As mentioned in the Introduction, our method does not help with the initial positioning of the membrane protein relative to the lipids (the orientation problem). The five test membrane proteins were therefore embedded in a simple patch of 512 POPC lipids using prior knowledge or published data, as described in the Methods. One must also decide whether to delete any lipids; for the small peripheral membrane proteins, PLA2 and N-BAR, and the monotonic protein, COX1, we decided not to since none of the proteins displaced a significant volume from the bilayer. Both of the polytonic integral membrane proteins, KcsA and OmpF, unavoidably take up a large volume of the bilayer; therefore, we deleted a number of lipids approximately equal to the surface area of the protein (measured in the plane of the bilayer). In this case, we simply deleted an equal number of lipids from both leaflets; however, it is likely that for asymmetric polytopic proteins, such as KcsA, and monotopic proteins with a larger interaction surface, one should delete different numbers of lipids from each leaflet. Failing to do so would likely lead to significant membrane curvature, unless it could be relieved by lipid flip-flop.

The atomistic *alchembed* simulations took 10−48 min (N = 1000) or 3.4−8.0 h (N = 10 000), depending on system size, on a single core of an Intel Xeon E5 processor. The *alchembed*

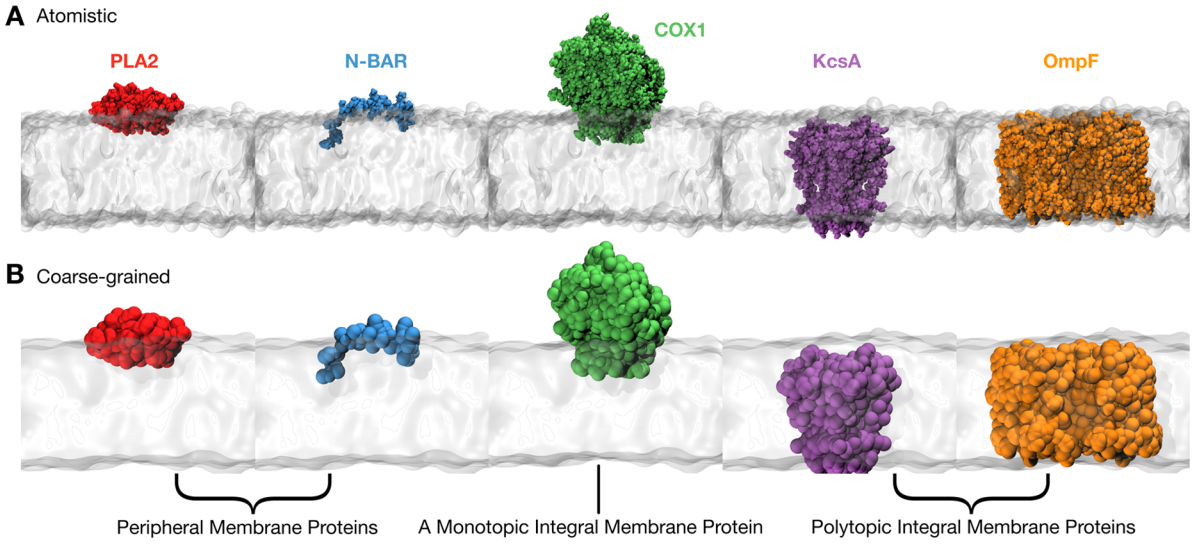


Figure 3. Structures and alignments of the five test membrane proteins shown in both (A) atomistic and (B) coarse-grained representations. The different modes of interaction with the lipid bilayer can be clearly seen. The bilayer in each case is represented by a transparent surface created by rolling a probe of radius 1.4 or 2.8 Å over the atomistic or coarse-grained lipid bilayers, respectively. Details of which experiment structures were used are given in Table 1. The colors in this figure will be used throughout the article to distinguish between the different proteins.

DOI: 10.1021/ct501111d
J. Chem. Theory Comput. 2015, 11, 2743−2754

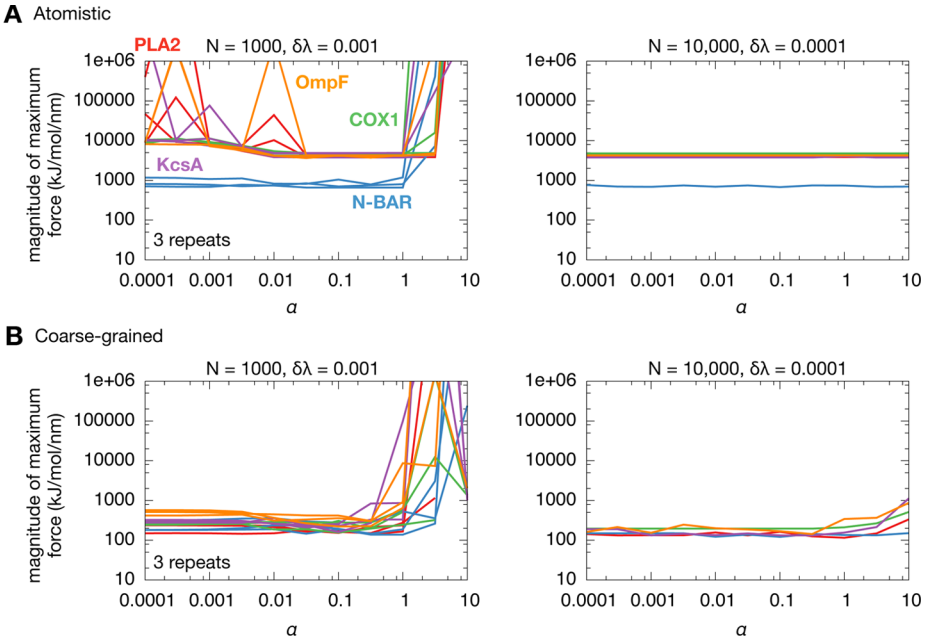


Figure 4. Maximum force experienced by a lipid or water atom (or bead) varies with the protein, α, and N as well as whether an (A) atomistic or (B) coarse-grained representation was used. Some of the simulations with very large forces failed and therefore did not complete; if this was, then the case with the largest force up to that point is recorded in the graph. There are three repeats of each of the $N = 1000$ simulations. Here, $b = 1$; a similar analysis for $b = 2$ can be found in Figure S4. Note that both axes have a logarithmic scale. The same color scheme has been used as that in Figure 3.

coarse-grained simulations were correspondingly faster, taking only 10−55 s ($N = 1000$) or 1.5−7.8 min ($N = 10\,000$).

As in the benzene test case, we calculated the maximum force experienced by any of the lipids and waters during the insertion process and also examined how the number of atoms (or beads) within a set distance of the inserted species varied. Examining these two metrics will be our basis for proposing a set of *alchembed* parameters that will allow any membrane protein to be inserted into an arrangement of lipids, using either an atomistic or a coarse-grained force field.

The maximum force generated during the insertion procedure is affected by three factors: the precise initial position of the protein relative to the lipids and waters, how quickly the soft-core potential is switched on, and how quickly λ increases, which is inversely proportional to N. For the shorter atomistic insertion simulations ($N = 1000$), very large magnitudes of the maximum force are seen for small (<0.0316) and large (>1) values of α (Figure 4). We infer that these large forces are due to the soft-core potential being introduced either too early or too late at small or large values of λ, respectively. Increasing N by an order of magnitude leads to the maximum force not changing with α, presumably because λ is increasing much more slowly between successive timesteps. A slightly different picture emerges from the coarse-grained simulations: again, very large forces are recorded for large values of α (>0.3) during the shorter insertion simulations, but there is no increase in the magnitude of the force recorded as $\alpha \to 0.0001$. The maximum force is approximately independent of α when N is increased to 10 000 time steps, with some small increases at very large values of α (>3). Overall, the shorter insertion simulations ($N = 1000$) appear to be well-behaved in the range $0.0316 \leq \alpha \leq 0.3162$, whereas the longer simulations ($N = 10\,000$) are less sensitive to the value of α.

This behavior is consistent with our earlier observation that soft-core potentials with large values of α change very slowly at first (Figure S2) but then very rapidly increase as $\lambda \to 1$, whereas soft-core potential with small values of α increase very rapidly in the first few steps of the simulation (Figure S2). The results above had b set to unity since this what most of the biomolecular simulation codes assume. Setting $b = 2$ significantly reduces the effects observed (Figure S4).

We shall now examine the number of lipid and water atoms (or beads) within σ of the inserting proteins. Our expectation is that this number should reduce during the *alchembed* simulation as the atoms (or beads) that are overlapping with the protein are gently moved out of the way by the soft-core van der Waals potential. Let us first consider two examples. In the first, we examine two atomistic simulations of COX1 with different values of α (Figure 5A). For a small value of α (e.g., $10^{1/2} \times 10^{-2} \sim 0.0316$), the number of atoms within σ (0.24 nm) increases slightly with λ but then falls over the remainder of the simulation. The behavior for larger values of α (e.g., $10^{1/2} \sim 3.16$) is markedly different: the number of atoms within 0.24 nm increases dramatically (by over 16-fold), reaching a peak at $\lambda \sim 0.8$ before falling rapidly and reaching a final value that is five times larger than the initial value. What has happened is that, because we solvated the system after the protein had been embedded, no waters were placed inside the perimeter of the protein (this also helps to avoid waters becoming trapped in cavities). At the start of the simulation when λ is small, the waters are therefore free to diffuse into the space occupied by the enzymatic domain of COX1. There are two important time scales to consider: how quickly the waters (and lipids) are diffusing and how quickly the van der Waals potential of the inserting protein is being switched on. If the former is quicker than the latter, then there is an ingress of waters and other molecules into the space occupied by the protein (Figure 5). These can get stuck in small internal cavities and are not all ejected as $\lambda \to 1$; this is particularly a problem when embedding large transmembrane membrane proteins, such as

DOI: 10.1021/ct501111d
J. Chem. Theory Comput. 2015, 11, 2743−2754

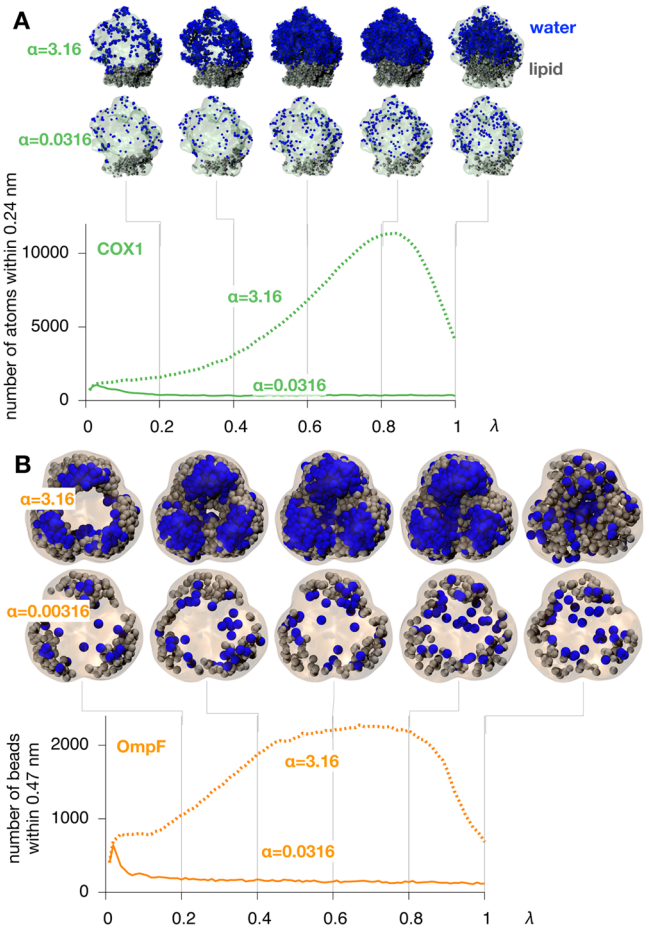


Figure 5. If the soft-core potential is switched on too slowly, then water and other molecules can enter the space occupied by the protein and become trapped. The two examples shown here illustrate this. *Alchembed* simulations of (A) atomistic COX1 and (B) coarse-grained OmpF. In both cases, the results of two simulations with $\alpha = 0.0316$ or 3.16 ($10^{1/2} \times 10^{-2}$ or 10^{0}) are plotted. $N = 10\,000$ time steps.

channels and porins. A similar effect can be seen in coarse-grained *alchembed* simulations of OmpF (Figure 5B). This time, the cavity into which the molecules are diffusing was created by the deletion of a number of lipids that were overlapping with the protein. As discussed in the Methods, the number of lipids removed corresponded to the surface area of the protein and was done to minimize the perturbation caused by the insertion of the integral membrane protein. Again, we see a large increase in the number of molecules within σ (0.47 nm for the coarse-grained force field) of the protein when $\alpha = 3.16$, reaching a maximum at $\lambda \sim 0.7$ before falling as $\lambda \to 1$.

Bearing both of these effects in mind, let us consider all five test membrane proteins (Figure 6). In all cases, the number of atoms (or beads) within σ of the inserting species rises initially but then starts decreasing. The maximum is smaller and occurs at lower values of λ when α is smaller. We infer that this is because the van der Waals potential of the protein is switched on faster at small values of α and this prevents the diffusion of waters and lipids into any cavities. At higher values of α, especially for proteins with vacant spaces introduced either during solvation (COX1 and PLA2) or by the deletion of lipids (KcsA and OmpF), there is a pronounced ingress of molecules, mainly water, as $\alpha \to 10$. Although many of these molecules are ejected as the protein appears, in all cases a larger number of molecules remain within σ of the protein at higher values of α. Since the integration time step is not changed, decreasing N from 10 000 to 1000 reduces this effect since there is less time for the waters to diffuse into any (temporarily) empty volumes. Setting $b = 2$ reduces, but does not remove, this effect (Figure S5).

A universal parameter set has to avoid both very large forces and the ingress of other molecules into any vacant regions. The latter requires α to be as small as possible, whereas the former suggests $0.0316 \leq \alpha \leq 0.3162$. Our results suggest that setting $\alpha = 0.1$ with $a = b = 1$, $c = 6$, and $N = 1\,000$ time steps will embed the majority of membrane proteins, when modeled using either an atomistic or coarse-grained force field. This parameter set should work in both GROMACS and AMBER since they both implement a soft-core potential in the form of eq 3. If possible, like in GROMACS, b should be set to 2. If the simulation crashes, then we recommend increasing N from 1000 to 10 000 timesteps.

Comparing *Alchembed* to an Established Method. To demonstrate that *alchembed* is an improvement over existing methods, we embedded all five test membrane proteins using g_membed[34] starting from exactly the same initial configurations. Since g_membed uses GROMACS, any differences in timing should be solely due to differences in the algorithms. We found that, unlike *alchembed*, g_membed could embed only seven of the 10 cases, but it was faster than *alchembed* by a factor of 1.7−2.6 for the PLA2, N-BAR, and COX1 proteins, rising to 3.8−12 faster for KcsA and OmpF, and in one case, it was 34× faster (Figure S6 and Table S1). *Alchembed*, however, resulted in the protein experiencing smaller forces during the embedding simulation. This was especially pronounced for the coarse-grained simulations; the maximum force recorded during the *alchembed* simulations was at least 2 orders of magnitude lower than that seen using g_membed (Figure S6B). Although the number of water and lipid atoms (or beads) close to the protein initially rises for the reasons discussed earlier during the *alchembed* simulations (Figure S7), by the end of the simulation there were fewer water or lipid atoms (or beads) proximal to the protein compared to that with g_membed. Taken together, this all suggests that *alchembed* is a more resilient method and results in a better embedded configuration than that with g_membed.

The Method Correctly Inserts the Test Proteins into the Lipid Bilayer. To demonstrate that the above parameter set has stably inserted the proteins into the lipid bilayers, we simulated each of the final atomistic configurations for 25 ns (or 250 ns for the coarse-grained configurations) using molecular dynamics. The area per lipid reached a plateau of ∼55 Å^2 in the control simulation of 512 atomistic POPC lipids. This is lower than the experimental value[49] and is a known problem with the CHARMM27 force field that has been recently corrected in CHARMM36.[50] Crucially, though, the area per lipid did not significantly change following insertion of the test proteins (Figure 7A). Although one would expect the N-BAR protein to perturb the bilayer, this was not observed in any of these simulations, we assume, due to the small size of the bilayers studied. Some initial transients are observed for the polytopic integral membrane proteins: these are due to the system adjusting to the deletion of some lipids. The deuterium lipid order parameter (S_z) measures the degree of order along the carbons in either of the lipid tails. The value of S_z for the eighth carbon in the sn-1 tail of POPC shows how the lipid tails initially became more ordered in the control simulations before

DOI: 10.1021/ct501111d
J. Chem. Theory Comput. 2015, 11, 2743−2754

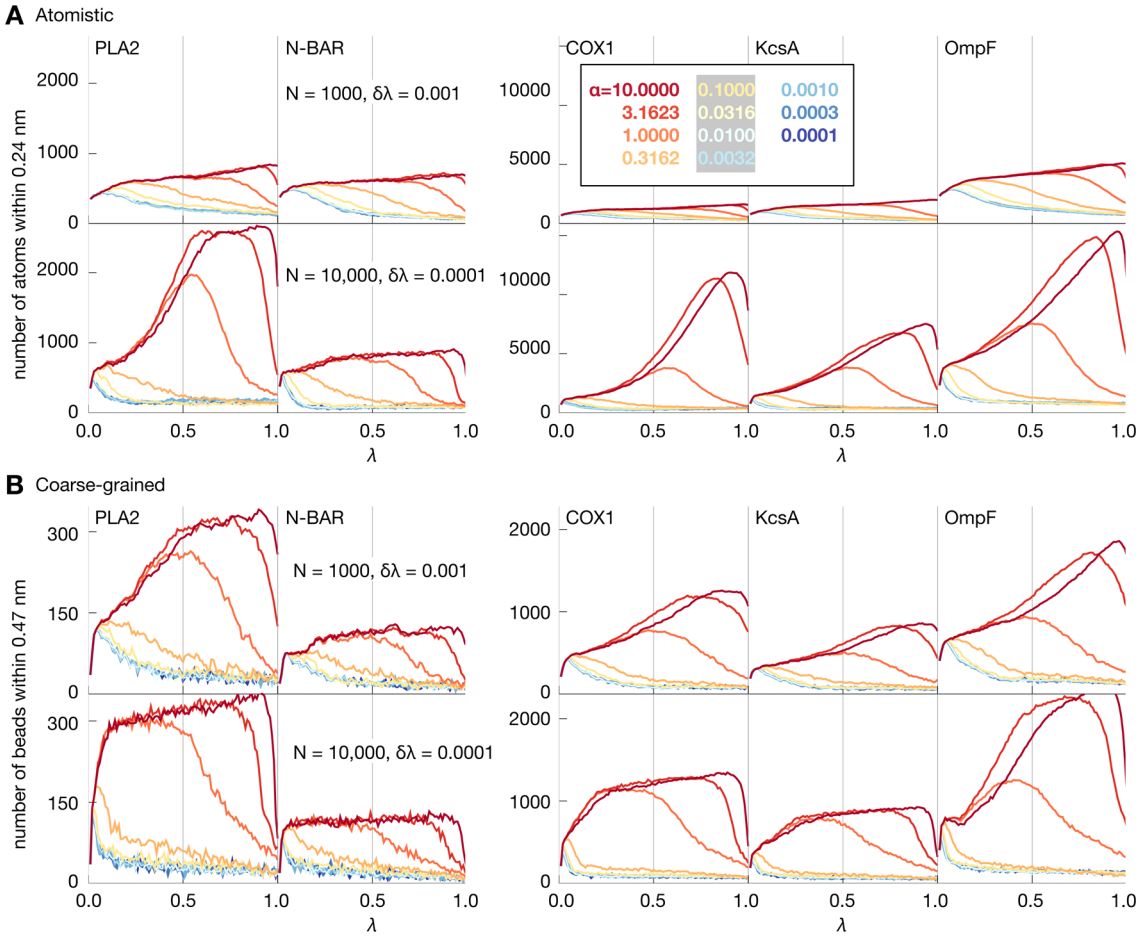


Figure 6. Number of atoms (or beads) within a set distance of the embedding protein varies as the van der Waals interactions between the protein and the remainder of the system are switched on (i.e., as $\lambda \rightarrow 1$) . The five test proteins are modeled using both (A) atomistic and (B) coarse-grained representations. In all cases, the results of 11 simulations spanning values of α from 0.0001 to 10 are plotted using a color scale that smoothly progresses from red to blue. Each set is also run for either $N = 1000$ or $N = 10\,000$ timesteps. For clarity, the results of only one of the three repeats for the shorter simulations are plotted. Here, $b = 1$; the same analysis for $b = 2$ can be found in Figure S5.

reaching a plateau of ∼0.3 (Figure 7B). There was no detectable change to S_z upon insertion of the peripheral or monotopic test proteins; however, there were transients for both the polytopic membrane proteins, especially OmpF, suggesting, again, that the bilayer is adjusting to the presence of the proteins.

As expected, the coarse-grained simulations equilibrate very rapidly, and the simulated area per lipid for POPC (∼62 Å^2) is closer to the experimental value (Figure 7C). Insertion of the test proteins leads to an increase in the area per lipid in all cases, possibly due to an underestimation of the protein area during the analysis. The lipid order parameter decreases following insertion of any of the test proteins, with the largest decrease observed for OmpF (Figure 7D), suggesting that it may be correlated with the size of the interface between protein and lipid.

Finally, we can examine how the structure of the protein changes compared to the original experimental structure. The C_α RMSD for the atomistic simulations (Figure S9) increases as expected and reaches a plateau after ∼10 ns. As is standard, the structures of the coarse-grained proteins have been maintained by an elastic network model. With the exception of N-BAR, which is very flexible using the default MARTINI parameters and therefore records high values of RMSD, the coarse-grained proteins also appear to be adequately stable during the 250 ns molecular dynamics simulations.

Embedding the Test Membrane Proteins into a Lipid Vesicle. Our development of the *alchembed* method in part was motivated by the inability of the current methods to embed multiple proteins simultaneously in complex lipid geometries. As a demonstration of the utility of the *alchembed* method, we simultaneously embedded all five test membrane proteins into a small, coarse-grained lipid vesicle. Although this system is nonphysiological, it is a challenging test, and, to our knowledge, no other existing method can simultaneously insert all five membrane proteins due to the different orientations in which they have been placed (Figure 8A). We carried out a series of *alchembed* simulations over a range of α values, as before, and similar trends in how the magnitude of the maximum force varies with α and how the number of beads within 0.47 nm of any of the proteins varies with both α and λ were observed. The final configuration from the $\alpha = 0.1$ simulation was used to seed a 250 ns molecular dynamics, confirming the proteins were stably inserted in the vesicle. As a final demonstration, we also embedded 108 copies of the truncated form of the cell signaling protein N-Ras[51] into a very large undulating coarse-grained ternary lipid bilayer (Figure S10 and S11). Although the proteins all have the same orientation, this test case is challenging due to its size and the number of proteins.

DOI: 10.1021/ct501111d
J. Chem. Theory Comput. 2015, 11, 2743−2754

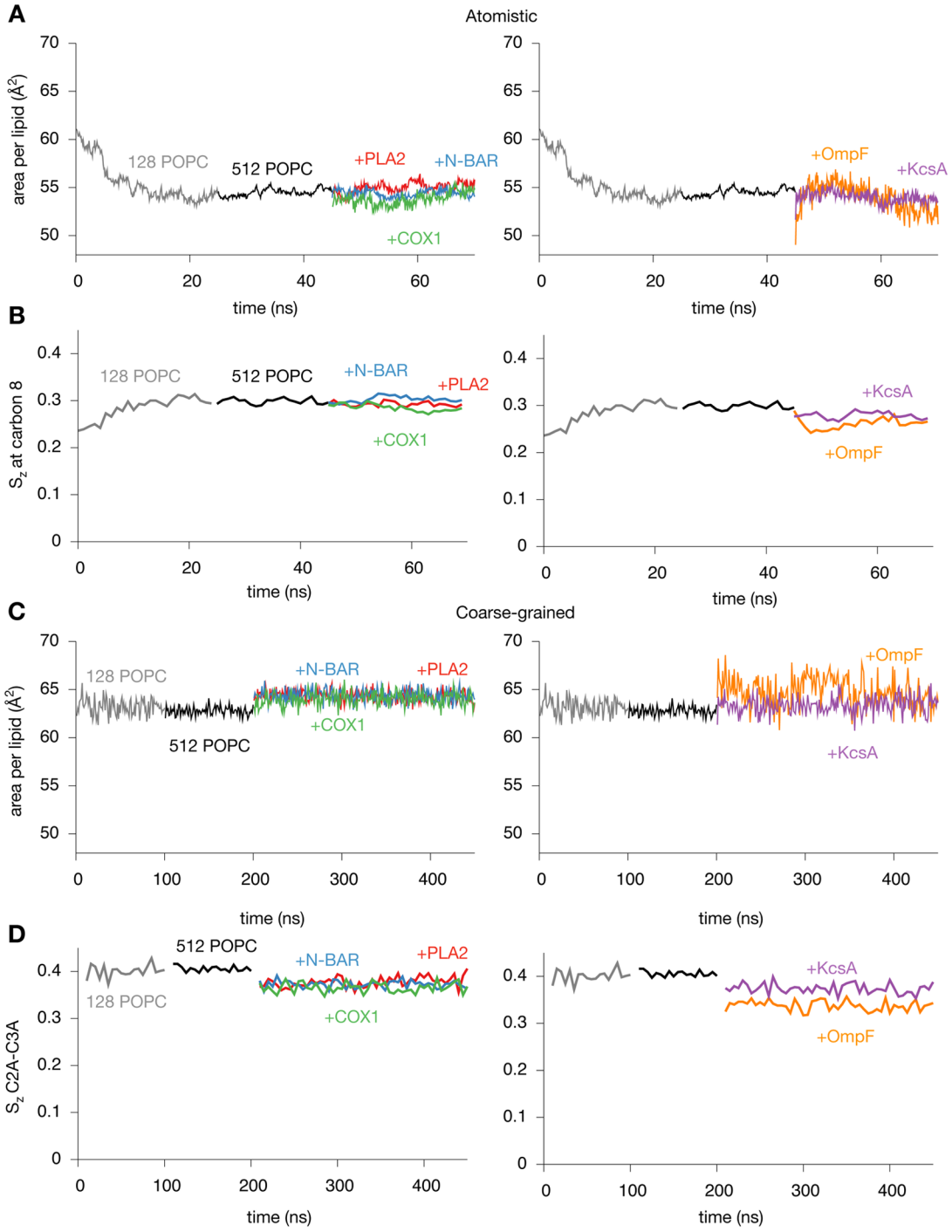


Figure 7. Embedding each of the five test cases does not significantly perturb the lipid bilayer. (A) The area per lipid decreases during the initial atomistic 128 POPC simulation. After 25 ns, this was tessellated to make a patch of 512 POPC lipids, which was simulated for a further 20 ns. Following embedding of the tests proteins in the bilayer using the standard parameter set, each test case was run for 25 ns. (B) The deuterium lipid order parameter, S_z, was calculated, and the value at the eighth carbon in the saturated tail was plotted against time to investigate changes in the ordering of the lipids. We used the eighth carbon after considering how S_z varied with time along the sn-1 chain of POPC (Figure S8) . (C) The area per lipid was analyzed for the coarse-grained simulations. (D) Since the saturated tail of POPC is modeled by only four coarse-grained beads, one cannot calculate the same lipid order parameter as in the atomistic simulation. Instead, a different lipid order parameter was followed. This was defined using the angle between the second and third beads and the membrane normal.

DISCUSSION

We have developed a method, *alchembed*, that can rapidly and reproducibly embed multiple proteins in a range of lipid geometries. Its effectiveness has been demonstrated on a series of challenging test cases, including a range of peripheral and integral membrane proteins. The heart of the method is the use of soft-core van der Waals potentials to gently grow in the protein(s). These potentials were originally developed to improve the convergence of alchemical free energy calculations[36] and are now implemented in all of the major biomolecular codes. Although our investigation used the GROMACS simulation package,[35] since AMBER[52] implements

DOI: 10.1021/ct501111d
J. Chem. Theory Comput. 2015, 11, 2743−2754

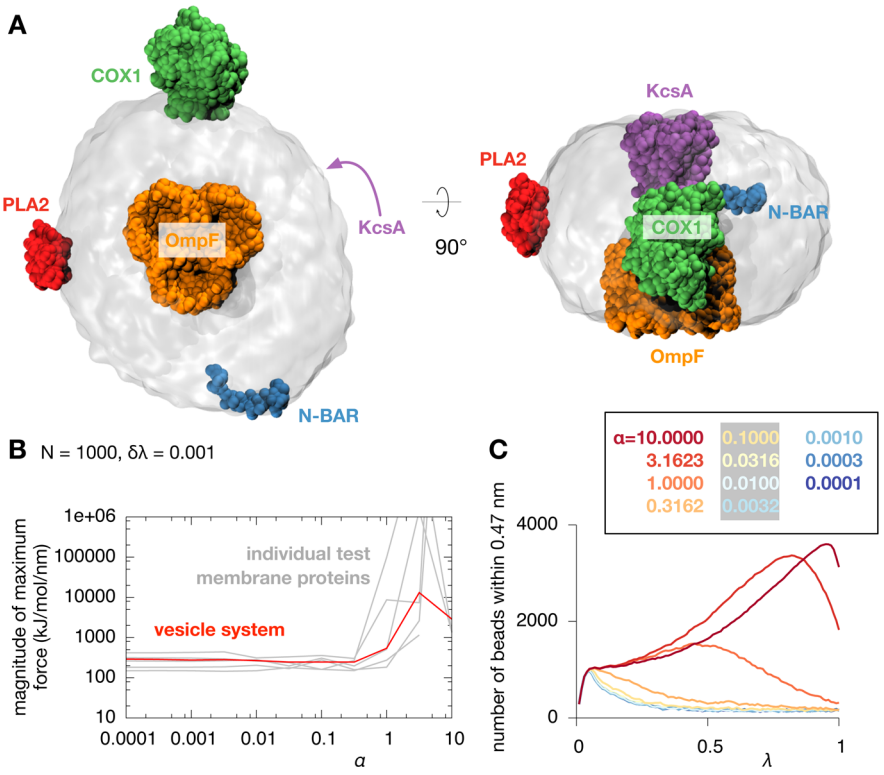


Figure 8. *Alchembed* easily and rapidly embeds the five test membrane proteins into a small coarse-grained lipid vesicle comprising 1286 POPC lipids. (A) Each protein was positioned manually using VMD, and, as before, the number of lipids corresponding to the embedded area of the integral membrane proteins, KcsA and OmpF, was removed, leaving 1097 POPC lipids. No lipids were removed to accommodate the other three membrane proteins. Each *alchembed* simulation took 7.2 min to run on a single core of an Intel Xeon E5 workstation. The total system size was 111 610 beads. (B) A series of short (N = 1000 steps) *alchembed* simulations was run, each with a different value of α. The maximum force experienced by a lipid or water bead (red line) was similar to that seen when embedding the proteins individually in planar POPC bilayers (gray lines, Figure 4). (C) The number of beads within 0.47 nm of any of the proteins displayed a similar trend as a function of λ compared to that seen for the proteins individually (Figure 6). To demonstrate that the system is stable, the final conformation was therefore taken from this *alchembed* simulation and 250 ns of molecular dynamics was run.

the same functional form for the soft-core van der Waals potential, our recommended parameter set (Table 2) should also apply for that simulation code. Because NAMD[41] and CHARMM[42] use a slightly different functional form for the soft-core potential, our results are not theoretically transferable. In practice, since the differences between the two functional forms at low values of α are small (Figures S2 and S3), we would expect our recommended set of parameters to work equally well in both of these molecular dynamics packages.

Table 2. Recommended Values of the Soft-Core Parameters That Our Test Simulations Suggest Will Embed a Membrane Protein in Either an Atomistic or Coarse-Grained Lipid Bilayer[a]

parameter	range tested	recommended value
a		1
b	1,2	1 or 2
c		6
α	0.0001−10	0.1
N	1000,10 000	1000

[a] b = 2 is preferred, but this is only currently supported by GROMACS. These values are subsequently referred to as the standard parameter set.

The main advantage of *alchembed* compared to other methods is that it can embed multiple proteins simultaneously into nonplanar lipid geometries. This makes it particularly well suited to setting up large complex coarse-grained simulations, for example, a virus capsid studded with integral membrane proteins.[14] Since the method uses functionality already present in the major simulation codes, it is simple for an experienced user to apply (a tutorial is included in the Supporting Information). Furthermore, the method naturally makes use of the extensive optimization present in all of the codes and is, therefore, also fast.

As discussed in the Introduction, the *alchembed* method addresses only the *insertion* problem; one must first correctly orient and position the membrane protein with respect to the lipids and then decide if it is necessary to delete some lipids. If one incorrectly positions the protein, for example, by placing a peripheral membrane protein too deep in a lipid bilayer, then following the *alchembed* process will insert the protein by moving the lipids out of the way, but ones does not know *a priori* how long it will take the protein to relax to its correct depth in the bilayer, and this may be longer than any subsequent molecular dynamics simulation. Likewise, the correct number of lipids have to be deleted to take account of the additional volume that the protein is introducing into the membrane. If not, then the lipids will be squeezed together as the protein is inserted, making it harder to equilibrate the resulting system. More subtly, if the protein is asymmetric, then not correctly deleting the right number from each leaflet will likely lead to buckling of the membrane. *Alchembed* does not

DOI: 10.1021/ct501111d
J. Chem. Theory Comput. 2015, 11, 2743−2754

address these problems; in common with other methods, the user must still solve the orientation and deletion problems.

Like the majority of the methods for embedding proteins into bilayers, one may have to delete some lipids to avoid too large a perturbation when the protein(s) appears. This can be avoided by using a larger number of lipids, thereby reducing the protein to lipid area, but that, of course, requires additional computational resource. Embedding proteins is not the intended use of the soft-core potential built into the biomolecular simulation packages and therefore it is also possible that idiosyncrasies in how the functionality is implemented in some of the codes may make it difficult to apply in practice. Finally, although it is trivial to check by visual inspection if a single protein has been inserted correctly into a lipid bilayer, it is less straightforward to visually check a large number of proteins. One must therefore be careful to check the final configuration when using *alchembed* to embed a large number of proteins into a complex lipid geometry.

We have deliberately limited our investigation to those parameters that can currently be varied (N and α) in the major biomolecular simulation packages. It is possible that altering other parameters (a, b, c) could improve the performance of the method. We have already seen how setting $b = 2$ is preferable to the default behavior ($b = 1$), but this is currently possible only in GROMACS. Likewise, it is probable that new, improved functional forms of the soft-core potential may also be better at embedding proteins into lipids. Our recommended parameter set is, therefore, the optimal set given the implementation(s) of soft-core potentials in the major codes at the time of writing. A tutorial is included in the Supporting Information.

Although we have studied embedding membrane proteins into lipid systems, this method may be generalized to inserting any molecular species into any soft-condensed matter system. For example, one can envisage using soft-core van der Waals potentials to mutate protein residues or lipid headgroups *in silico*. Alternatively, one could use an intermediate value of λ to soften the interactions between a ligand and a protein during a computational docking run to help the ligand squeeze into tight cavities. There have been some efforts to soften the van der Waals repulsive term in computational docking, but these have focused on reducing the power of the repulsive term from $1/r^{12}$ to $1/r^{9}$.[53] This will reduce the singularity at small values of r, but it will not abolish it (Figure 1A). Using soft-core van der Waals potentials, instead, could improve the ability of these codes to find poses that previously would have been rejected due to a slight steric clash.

METHODS

Equilibration of the Lipid Bilayers. The energy of a small patch of 128 POPC lipids, solvated by 5003 waters and 14 Na^+ and 14 Cl^+, was first minimized for 1200 steps using the steepest descent algorithm in GROMACS 5.0.[35] The temperature was then gradually increased from 100 to 310 K in increments of 20 K, with 40 ps of molecular dynamics run at each temperature using a Langevin thermostat with a damping coefficient of 2.0 ps^{-1}. The integration time step was 2 fs, and the lengths of all bonds involving a hydrogen were constrained using the LINCS algorithm.[54] The CHARMM27 force field,[55] as implemented in GROMACS,[56] was used in conjunction with the Verlet cutoff scheme. Electrostatic forces were calculating using PME with a real space cutoff of 1.2 nm; van der Waals forces were also cutoff at 1.2 nm. A Berendsen barostat was applied semi-isotropically with a target pressure of 1 bar, a compressibility of 4.46×10^{-5} bar^{-1}, and a relaxation time of 1 ps. The simulation unit cell was equilibrated for 25 ns with coordinates saved to disc every 10 ps. A larger patch of 512 POPC lipids was created by tessellating (2×2) the final configuration from the above simulation. This was then equilibrated for a further 20 ns. The five atomistic test membrane proteins (Figure 3) were embedded into the final configuration of this simulation.

The same initial configuration of 128 POPC lipids was converted to a coarse-grained (CG) representation by placing CG beads at the positions of key atoms. The energy of the lipids was minimized for 500 steps using the steepest descent algorithm before the patch was solvated with 1028 water beads. The energy was minimized for a further 1000 steps before the system was equilibrated for 100 ns using GROMACS 5.0, an integration time step of 20 ps, and the Verlet cutoff scheme. Electrostatic forces were calculated using the reaction-field method with a real space cutoff at 1.2 nm and the dielectric constant set to 15. van der Waals forces were cutoff at 1.2 nm with a switching function applied from 0.9 nm. A velocity rescale thermostat with a relaxation time of 1.0 ps was applied to maintain the temperature at 310 K, and a Berendsen barostat was applied semi-isotropically with a target pressure of 1 bar, a compressibility of 3×10^{-4} bar^{-1}, and a relaxation time of 2 ps. Following a similar procedure as that for the atomistic simulation, a larger patch of 512 POPC lipids was created by tessellating the final configuration of the simulation; this was then equilibrated for a further 100 ns. The MARTINI v2.2 force field was used.[6]

Setup of the Test Cases. Four benzene molecules were added to the final configuration of the atomistic bilayer containing 128 POPC lipids. They were placed such that the tails of a single lipid exactly bisected the planes of the four molecules, as shown in Figure 2. The *alchembed* process was then applied, as described in the next section.

Each of the five test membrane proteins interacts with the lipid bilayer in different ways and therefore needs careful initial orientation. A previous simulation study of PLA2 identified Trp3, Leu19, and Met20 as being key hydrophobic anchoring residues.[57] The centers of mass of these residues from a crystal structure (PDB: 1P2P[44]) were then placed in the same plane as the phosphate atoms (or beads) as the upper leaflet of the bilayer containing 512 POPC lipids. In a similar fashion, a structure of N-BAR solved by NMR (PDB: 2RND[45]) was inserted into a copy of the same bilayer by moving the protein such that the center of mass of the helical portion (residues 11−33) coincided with the center of mass of the phosphates atoms or beads forming the upper leaflet. Helices A, B, and C of the membrane binding domain of COX1 (PDB: 1PRH[46]) were positioned in the same plane as the phosphates of the upper leaflet of the bilayer. This protocol has been previously shown to lead to the integration of this monotopic protein.[58,59] Since the volume taken up by all three of these proteins in the bilayer is small compared to the bilayer as a whole, it was assumed the lipids would be able to repack around the proteins, so no lipids were deleted.

It is a different picture for both of the transmembrane proteins (OmpF PDB: 4GCS[48] and KcsA PDB: 1K4C[47]). These were initially moved such that their centers of mass coincided with that of the bilayer and their pore (or channel) axes were parallel to the membrane normal. Small adjustments were made along the membrane normal to ensure that the

DOI: 10.1021/ct501111d
J. Chem. Theory Comput. 2015, 11, 2743−2754

loops were solvated and the bands of aromatic residues were approximately located around the level of the glycerols. Inserting both proteins created a significant excluded volume for lipids, compared to the volume of the bilayer as a whole, and therefore we deleted a number of lipids roughly equivalent to the surface area of each protein. For KcsA, 45 lipids were deleted. Considerable overlap between protein and lipids remained; in a previous study, nearly twice the number of lipids was deleted to ensure no steric hindrance between protein and lipid.[60] OmpF is considerably larger, and a total of 137 lipids were deleted. Again, considerable overlap remained, as demonstrated by an earlier study having to delete 194 POPE lipids to remove all steric hindrance.[28]

A single copy of each of the five test membrane proteins was then embedded in a pre-equilibrated coarse-grained vesicle. The proteins were positioned as before, and lipids were deleted from around the center of mass of KcsA and OmpF, resulting in a system of 111 610 beads. The coarse-grained model of N-Ras and the parametrization of the lipid anchors has been described previously.[61] The truncated form of N-Ras comprises only residues Gly175 to Cys181 with a farnesyl attached to the terminal cysteine and a palmitoyl to Cys-176. Each copy of the protein was lowered until the number of waters within 0.5 nm of either lipid anchor was less than three. This criterion was found to adequately embed the lipid anchors in the bilayer while keeping the peptide solvated.

***Alchembed* Process.** To embed these test cases using GROMACS 5.0, the free energy module must be activated and several additional lines must be added to the input file. These specify the name of the molecule being inserted, which interactions are present at the start and end of the simulation, the initial value of λ (zero), how much λ increases each time step ($\delta\lambda$), and values for α, b, and c. Since there is no error checking, it is important to make sure that $N = 1/\delta\lambda$. Finally, position restraints with a spring constant of 1000 kJ mol^{-1} nm^{-2} were applied to the heavy atoms of the protein. Integration time steps of 1 and 10 fs were used for the atomistic and coarse-grained simulations, respectively. The exception was the challenging benzene case, which required an integration time step of 0.5 fs. As shown in Figures 4 and 6, a range of parameters was tried: $N \in \{1000,10\,000\}$, $b \in \{1,2\}$, and $\alpha \in \{0.0001,0.0003,0.0010,0.0031,0.0100,0.0316,0.1000,0.3162,1.0000,3.1623,10.0000\}$. Each value of α is $10^{1/2}\times$ the previous one. All coordinates and forces were saved to disc every 10 frames. Only the van der Waals parameters of the protein were switched on during the *alchembed* simulations. Our recommended set of parameters can be found in Table 2.

Analysis. Both the number of lipids and waters within σ of the test protein and the maximum force recorded were analyzed using Python scripts that made use of MDAnalysis.[62] The area per lipid, taking into account the protein, was calculated using g_lomepro.[63] The lipid deuterium order parameters were calculated using VMD for the atomistic simulations[64] and using MDAnalysis for the coarse-grained simulations. All images were rendered using VMD, and all graphs were plotted using gnuplot.

Molecular Dynamics Simulations. The final configuration from the appropriate *alchembed* simulation ($N = 1000$, $\alpha = 0.1$, $b = 1$) was used to seed a molecular dynamics simulation of length 25 or 250 ns for each of the atomistic and coarse-grained test membrane proteins, respectively. Coordinates were saved to disc every 100 ps or 1 ns. The same parameters were used as those used to equilibrate the lipid bilayers with the exception that a Parinello–Rahman barostat was applied semi-isotropically with a relaxation time of 1.0 ps (atomistic) or 12 ps (coarse-grained).

A large coarse-grained lipid bilayer containing 43 200 DOPC lipids, 5400 sphingomyelin lipids, and 5364 cholesterols was constructed. It was solvated by 1 389 494 water beads, 2394 sodium beads, and 2502 chloride beads. Following equilibration, 108 copies of tN-Ras were embedded as described and then 100 ns of coarse-grained molecular dynamics was run using the same parameters as above.

■ ASSOCIATED CONTENT

Supporting Information

Alchembed tutorial. Figure S1: If you randomly place an atom (or coarse-grained bead) inside a typical simulation box, then there is a high probability it will be within σ of another atom (or bead). Figures S2 and S3: Increasing α slows the rate at which the van der Waals potential is introduced. Figure S4: The maximum force experienced by a lipid or water atom (or bead) varies the protein, α, N, and whether an atomistic or coarse-grained representation is used. Figure S5: The number of atoms (or beads) within a set distance of the embedding protein varies as the van der Waals interactions between the protein and the remainder of the system are switched on. Figure S6: The five test membrane proteins embedded into both the coarse-grained and atomistic simple POPC bilayer using g_membed. Figure S7: The number of atoms (or beads) within 0.24 nm (or 0.47 nm) of the protein is plotted as a function of λ for both *alchembed* and g_membed. Figure S8: The deuterium lipid order parameter for the sn-1 chain of POPC changes during the 25 ns simulation of the small patch of 128 POPC lipids. Figure S9: The test proteins are stable in molecular dynamics simulations following insertion by the *alchembed* method. Figure S10: *Alchembed* easily and rapidly embeds 108 copies of tN-Ras into a very large undulating coarse-grained ternary lipid bilayer. Figure S11: The number of atoms (or beads) within a set distance of the embedding protein varies as the van der Waals interactions between the protein and the remainder of the system are switched on. Table S1: The g_membed approach is faster than *alchembed*; however, only *alchembed* successfully embeds all of the test proteins into the simple POPC lipid bilayer. The Supporting Information is available free of charge on the ACS Publications website at DOI: 10.1021/ct501111d.

■ AUTHOR INFORMATION

Corresponding Author

*E-mail: philip.fowler@bioch.ox.ac.uk.

Funding

We would like to acknowledge the Wellcome Trust (092970/Z/10/Z) and UCB NewMedicines and BBSRC (BB/K011510/1) for funding and PRACE through project RA1923 for access to the CURIE Tier-0 supercomputer.

Notes

The authors declare no competing financial interest.

■ ACKNOWLEDGMENTS

We would also like to thank Phillip Stansfeld, Tyler Reddy, Heidi Koldsø, and Matthieu Chavent for helpful discussions.

DOI: 10.1021/ct501111d
J. Chem. Theory Comput. 2015, 11, 2743–2754

REFERENCES

(1) Yildirim, M. A.; Goh, K.-I.; Cusick, M. E.; Barabási, A.-L.; Vidal, M. *Nat. Biotechnol.* **2007**, *25*, 1119−26.
(2) Stansfeld, P. J.; Sansom, M. S. *Structure* **2011**, *19*, 1562−72.
(3) Lindahl, E.; Sansom, M. S. P. *Curr. Opin. Struct. Biol.* **2008**, *18*, 425−31.
(4) Gumbart, J.; Wang, Y.; Aksimentiev, A.; Tajkhorshid, E.; Schulten, K. *Curr. Opin. Struct. Biol.* **2005**, *15*, 423−31.
(5) Woolf, T. B.; Roux, B. *Proc. Natl. Acad. Sci. U.S.A.* **1994**, *91*, 11631−5.
(6) Marrink, S. J.; Tieleman, D. P. *Chem. Soc. Rev.* **2013**, *42*, 6801−22.
(7) Risselada, H. J.; Marrink, S. J. *Phys. Chem. Chem. Phys.* **2009**, *11*, 2056−67.
(8) Parton, D. L.; Klingelhoefer, J. W.; Sansom, M. S. P. *Biophys. J.* **2011**, *101*, 691−9.
(9) Domanski, J.; Marrink, S. J.; Schåfer, L. V. *Biochem Biophys Acta* **2012**, *1818*, 984−94.
(10) Baoukina, S.; Marrink, S.; Tieleman, D. P. *Biophys. J.* **2012**, *102*, 1866−71.
(11) Goose, J. E.; Sansom, M. S. P. *PLoS Comput. Biol.* **2013**, *9*, e1003033.
(12) Chavent, M.; Reddy, T.; Goose, J.; Dahl, A. C. E.; Stone, J. E.; Jobard, B.; Sansom, M. S. P. *Faraday Discuss* **2014**, *169*, 455−75.
(13) Ingólfsson, H. I.; Melo, M. N.; van Eerden, F. J.; Arnarez, C.; López, C. A.; Wassenaar, T. A.; Periole, X.; De Vries, A. H.; Tieleman, D. P.; Marrink, S. J. *J. Am. Chem. Soc.* **2014**, *136*, 14554−9.
(14) Reddy, T.; Shorthouse, D.; Parton, D. L.; Jefferys, E.; Fowler, P. W.; Chavent, M.; Baaden, M.; Sansom, M. S. *Structure* **2015**, *23*, 584−97.
(15) Ulmschneider, M. B.; Sansom, M. S. P.; Di Nola, A. *Proteins* **2005**, *59*, 252−65.
(16) Scott, K. A.; Bond, P. J.; Ivetac, A.; Chetwynd, A. P.; Khalid, S.; Sansom, M. S. P. *Structure* **2008**, *16*, 621−30.
(17) Kalli, A. C.; Devaney, I.; Sansom, M. S. P. *Biochemistry* **2014**, *53*, 1724−32.
(18) Kandt, C.; Ash, W. L.; Tieleman, D. P. *Methods* **2007**, *41*, 475−88.
(19) Biggin, P. C.; Bond, P. J. *Methods Mol. Biol.* **2008**, *443*, 147−60.
(20) Tieleman, D. P. In *Molecular Simulations and Biomembranes: From Biophysics to Function*, 1st ed.; Sansom, M. S. P., Biggin, P. C., Eds.; Royal Society of Chemistry: Cambridge, 2010; pp 1−25.
(21) Sommer, B. *Comput. Struct. Biotechnol. J.* **2013**, *5*, e201302014.
(22) Tai, K.; Fowler, P. W.; Mokrab, Y.; Stansfeld, P.; Sansom, M. S. P. In *Methods in Nano Cell Biology*, 1st ed.; Jena, B. P., Ed.; Elsevier: Amsterdam, 2008; Vol. *90*, Chapter 12, pp 233−265.
(23) Woolf, T. B.; Roux, B. *Proteins* **1996**, *24*, 92−114.
(24) Jo, S.; Kim, T.; Im, W. *PLoS one* **2007**, *2*, e880.
(25) Stansfeld, P. J.; Sansom, M. S. P. *J. Chem. Theory Comput.* **2011**, *7*, 1157−66.
(26) Wassenaar, T. A.; Pluhackova, K.; Böckmann, R. A.; Marrink, S. J.; Tieleman, D. P. *J. Chem. Theory Comput.* **2014**, *10*, 676−90.
(27) Durrant, J. D.; Amaro, R. E. *PLoS Comput. Biol.* **2014**, *10*, e1003720.
(28) Tieleman, D. P.; Berendsen, H. J. *Biophys. J.* **1998**, *74*, 2786−801.
(29) Shen, L.; Bassolino, D.; Stouch, T. *Biophys. J.* **1997**, *73*, 3−20.
(30) Faraldo-Gómez, J. D.; Smith, G. R.; Sansom, M. S. P. *Eur. Biophys J.* **2002**, *31*, 217−27.
(31) Staritzbichler, R.; Anselmi, C.; Forrest, L. R.; Faraldo-Gómez, J. D. *J. Chem. Theory Comput.* **2011**, *7*, 1167−76.
(32) Javanainen, M. *J. Chem. Theory Comput.* **2014**, *10*, 2577−82.
(33) Schmidt, T. H.; Kandt, C. *J. Chem. Inf Model.* **2012**, *52*, 2657−69.
(34) Wolf, M. G.; Hoefling, M.; Aponte-Santamaría, C.; Grubmüller, H.; Groenhof, G. *J. Comput. Chem.* **2010**, *31*, 2169−74.
(35) Pronk, S.; Páll, S.; Schulz, R.; Larsson, P.; Bjelkmar, P.; Apostolov, R.; Shirts, M. R.; Smith, J. C.; Kasson, P. M.; van der Spoel, D.; Hess, B.; Lindahl, E. *Bioinformatics* **2013**, *29*, 845−54.
(36) Pitera, J.; van Gunsteren, W. *Mol. Simul.* **2002**, *28*, 45−65.
(37) Zacharias, M.; Straatsma, T. P.; McCammon, J. A. *J. Chem. Phys.* **1994**, *100*, 9025.
(38) Beutler, T. C.; Mark, A. E.; van Schaik, R. C.; Gerber, P. R.; van Gunsteren, W. F. *Chem. Phys. Lett.* **1994**, *222*, 529−39.
(39) Pham, T. T.; Shirts, M. R. *J. Chem. Phys.* **2011**, *135*, 034114.
(40) Steinbrecher, T.; Mobley, D. L.; Case, D. A. *J. Chem. Phys.* **2007**, *127*, 214108.
(41) Phillips, J. C.; Braun, R.; Wang, W.; Gumbart, J.; Tajkhorshid, E.; Villa, E.; Chipot, C.; Skeel, R. D.; Kalé, L.; Schulten, K. *J. Comput. Chem.* **2005**, *26*, 1781−802.
(42) Brooks, B. R.; et al. *J. Comput. Chem.* **2009**, *30*, 1545−614.
(43) Gapsys, V.; Seeliger, D.; de Groot, B. L. *J. Chem. Theory Comput.* **2012**, *8*, 2373−82.
(44) Edwards, S. H.; Thompson, D.; Baker, S. F.; Wood, S. P.; Wilton, D. C. *Biochemistry* **2002**, *41*, 15468−76.
(45) Löw, C.; Weininger, U.; Lee, H.; Schweimer, K.; Neundorf, I.; Beck-Sickinger, A. G.; Pastor, R. W.; Balbach, J. *Biophys. J.* **2008**, *95*, 4315−23.
(46) Picot, D.; Loll, P. J.; Garavito, R. M. *Nature* **1994**, *367*, 243−9.
(47) Zhou, Y.; Morais-Cabral, J. H.; Kaufman, A.; MacKinnon, R. *Nature* **2001**, *414*, 43−8.
(48) Ziervogel, B. K.; Roux, B. *Structure* **2013**, *21*, 76−87.
(49) Kucerka, N.; Tristram-Nagle, S.; Nagle, J. F. *J. Membr. Biol.* **2005**, *208*, 193−202.
(50) Klauda, J. B.; Venable, R. M.; Freites, J. A.; O'Connor, J. W.; Tobias, D. J.; Mondragon-Ramirez, C.; Vorobyov, I.; MacKerell, A. D.; Pastor, R. W. *J. Phys. Chem. B* **2010**, *114*, 7830−43.
(51) Castellano, E.; Santos, E. *Genes Cancer* **2011**, *2*, 216−31.
(52) Salomon-Ferrer, R.; Case, D. A.; Walker, R. C. *Wiley Interdiscip. Rev.: Comput. Mol. Sci.* **2013**, *3*, 198−210.
(53) Ferrari, A. M.; Wei, B. Q.; Costantino, L.; Shoichet, B. K. *J. Med. Chem.* **2004**, *47*, 5076−84.
(54) Hess, B.; Bekker, H.; Berendsen, H. J. C.; Fraaije, J. G. E. M. *J. Comput. Chem.* **1997**, *18*, 1463−72.
(55) Mackerell, A. D.; Feig, M.; Brooks, C. L. *J. Comput. Chem.* **2004**, *25*, 1400−15.
(56) Bjelkmar, P.; Larsson, P.; Cuendet, M. A.; Hess, B.; Lindahl, E. *J. Chem. Theory Comput* **2010**, *6*, 459−66.
(57) Wee, C. L.; Balali-Mood, K.; Gavaghan, D.; Sansom, M. S. P. *Biophys. J.* **2008**, *95*, 1649−57.
(58) Fowler, P. W.; Coveney, P. V. *Biophys. J.* **2006**, *91*, 401−10.
(59) Fowler, P. W.; Balali-Mood, K.; Deol, S.; Coveney, P. V.; Sansom, M. S. P. *Biochemistry* **2007**, *46*, 3108−15.
(60) Fowler, P. W.; Tai, K.; Sansom, M. S. P. *Biophys. J.* **2008**, *95*, 5062−72.
(61) Jefferys, E.; Sansom, M. S. P.; Fowler, P. W. *Faraday Discuss.* **2014**, *169*, 209−23.
(62) Michaud-Agrawal, N.; Denning, E. J.; Woolf, T. B.; Beckstein, O. *J. Comput. Chem.* **2011**, *32*, 2319−27.
(63) Gapsys, V.; de Groot, B. L.; Briones, R. *J. Comput.-Aided Mol. Des.* **2013**, *27*, 845−58.
(64) Humphrey, W.; Dalke, A.; Schulten, K. *J. Mol. Graphics* **1996**, *14*, 33−8.

DOI: 10.1021/ct501111d
J. Chem. Theory Comput. 2015, 11, 2743−2754