Received: 14 January 2022 | Revised: 25 February 2022 | Accepted: 2 March 2022

DOI: 10.1002/jcc.26847

SOFTWARE NOTE

Journal of COMPUTATIONAL CHEMISTRY WILEY

prepareforleap: An automated tool for fast PDB-to-parameter generation

Daniel R. Roe[1] | Christina Bergonzo[2]

[1]Laboratory of Computational Biology, National Heart Lung and Blood Institute, National Institutes of Health, Bethesda, Maryland, USA

[2]Institute for Bioscience and Biotechnology Research, National Institute of Standards and Technology and University of Maryland, Rockville, Maryland, USA

Correspondence
Daniel R. Roe, Laboratory of Computational Biology, National Heart Lung and Blood Institute, National Institutes of Health, Bethesda, MD 20892, USA.
Email: daniel.roe@nih.gov

Abstract

Setting up molecular dynamics simulations from experimentally determined structures is often complicated by a variety of factors, particularly the inclusion of carbohydrates, since these have several anomer types which can be linked in a variety of ways. Here we present a stand-alone tool implemented in the widely-used software CPPTRAJ that can be used to automate building structures and generating a "ready to run" parameter and coordinate file pair. This tool automatically identifies carbohydrate anomer type, configuration, linkage, and functional groups, and performs topology modifications (e.g., renaming residue/atom names) required to build the final system using state of the art GLYCAM force field parameters. It will also generate the necessary commands for bonding carbohydrates and creating any disulfide bonds.

KEYWORDS

Amber, carbohydrates, CPPTRAJ, molecular dynamics, system preparation

1 | INTRODUCTION

As the field of structural biology expands to encompass more complex structures determined by new techniques and with higher precision, the world of all-atom molecular simulations must follow in suit. Building models, or, interpreting deposited models or 3D coordinate sets from the Protein Data Bank (PDB, www.rcsb.org) can quickly become complicated when they include multiple protein chains with attached glycans. A recent example of this complexity can be seen in the SARS-CoV-2 spike protein models released by E. Fadda and the Universirty of Georgia Complex Carbohydrate Research Center, among others.[1]

Generating topology and parameter files, which include necessary force field information to run a molecular dynamics (MD) simulation, can become difficult when many types of biological molecules are combined.

Setting up a MD simulation from an experimentally determined structure can sometimes be a difficult process. In particular, incorporating glycans into parameter files can be quite challenging due to the large variety of carbohydrate anomer types, configurations, and linkages. One very popular set of parameters for carbohydrates is the GLYCAM[2] force field. While a host of tools exist to interpret glycan PDB notation and translate into GLYCAM force field notation, none exists as part of a single-step solution for complex carbohydrates attached to a protein.

One common problem encountered when preparing structures from the PDB is that the residue and atom names in the PDB do not always correspond to residue and atom names used by the MD force field. For example, in Amber[3] force fields, regular cysteine is CYS just like in the PDB, but cysteines involved in disulfide bonds are CYX. The residue naming problem can be particularly difficult when carbohydrates are involved, since the unique combinations of anomer type, configuration (D or L), and linkage necessitate unique residue names (as is the case for the GLYCAM[2] force field). Typically, it is up to the user to make such changes manually, leaving room for error.

Several stand-alone utilities exist to model carbohydrates such as POLYS,[4] CarbBuilder,[5] DoGlycans,[6] and RosettaCarbohydrate.[7] Several web-based utilities for modeling carbohydrates exist as well, including GlyProt,[8] GLYCAM-Web,[9] and Charmm-GUI.[10,11] Out of all of these programs, only the web-based Charmm-GUI has the ability to go from a

PDB file containing carbohydrates to simulation-ready files, although this typically still requires some user intervention (at the very least users must click the web GUI manually to advance from step to step). Other programs need much more user input and either require the user to manually select or add carbohydrate anomer types, and/or do not change to residue names appropriate for the GLYCAM force field.

Here we introduce a new stand-alone and open-source software tool for curating a PDB file so that it can be subsequently read by the Amber system preparation tool LEaP (Link, Edit, and Parm) in order to generate the files necessary to carry out MD simulations. The main strengths of this tool are that it will take care of atom and residue renaming automatically for disulfides, carbohydrates, and (if hydrogen atoms are present in the PDB) histidine residues, and it will also generate the bond commands necessary for LEaP to properly handle branched systems (which it typically cannot natively do without user intervention). This tool has been implemented as the command "prepareforleap" in the CPPTRAJ[12] analysis program (version 6.4.0) in order to take advantage of CPPTRAJ's existing topology/coordinate file parsing and geometric calculations (for e.g., determining carbohydrate anomer types, disulfide bonds, etc.). Its stand-alone nature (execution and output is local) facilitates incorporation into automated and/or high-throughput workflows. Although the tool is currently designed for use with Amber/GLYCAM[13] force fields, it allows for customization via plain text files with the aim of enabling interoperability with other force fields.

2 | METHODS

First, the structure from a PDB file (or any other topology/coordinate file that can be read by CPPTRAJ) is loaded into CPPTRAJ. If the input structure is from a PDB that contains a MISSING RESIDUES section, CPPTRAJ will notify the user via a warning message. If the input coordinate file contains multiple frames (e.g., different MODELs in PDB files) a specific one can be chosen by the user; otherwise the first frame is chosen. Similarly, if the structure contains alternate atom locations a specific one can be chosen by the user, otherwise the first location (typically labeled "A") is chosen.

The structure can be further modified by removing solvent molecules, hydrogen atoms, and/or user-specified atoms via a given atom mask expression that follows Amber/CPPTRAJ syntax. Solvent atoms can be identified by a user-specified name (default is the PDB V3 water residue name "HOH"). Hydrogens are identified by name, or by mass if mass information is present in the topology file.

2.1 | Histidine detection

If hydrogen atoms are present in the structure, the command will attempt a simple and straightforward determination of the protonation state of any histidine residues (defined by a user-specifiable residue name, default "HIS") based on where hydrogens are bonded, and assign the appropriate residue name (also user-specifiable) for epsilon-protonated, delta-protonated, or double-protonated forms (respectively "HIE," "HID," and "HIP" by default).

2.2 | Disulfide bond determination

The command will identify any existing disulfide bonds as well as potential disulfide bonds and generate the corresponding LEaP "bond" commands which can be applied after the structure is loaded in LEaP. Potential disulfide bonding atoms can be identified via a user-specifiable mask expression (default is atoms named "SG" in residues named "CYS"). First, any existing disulfide bonds are identified, the bonds removed (since these bonds cannot currently be understood when LEaP loads a PDB), and the corresponding LEaP bond commands are written to a file. Next, the distances between all remaining potential disulfide atoms not yet bonded are calculated and sorted, and those less than a user-specifiable cutoff (default 2.5 Å) are identified as disulfide bonds. If any atom has the potential to make more than one disulfide bond, the pair with the shortest distance gets priority.

2.3 | Sugar preparation

The command will identify sugars, identify and prepare any functional groups that should be separate residues (e.g., sulfate, methyl, terminal hydroxyl, etc.), assign them the correct residue and atom names for the GLYCAM force field, and generate the necessary LEaP bond commands to properly connect the sugars after being loaded into LEaP. Coordinates for the sugars must be present in the input structure; the command does not yet support user-specified glycosylation.

Sugars are selected via a user-specifiable atom mask expression. If not specified, a default mask containing all PDB carbohydrate residue names corresponding to existing GLYCAM sugars is used (see SI for the full mask expression). Users can also specify if sugars already have GLYCAM naming, in which case the sugar names are checked for consistency but not renamed.

The command will first attempt to correct any topology issues such as functional groups that need to be separate residues and/or missing glycosidic linkages.

2.4 | Sugar atom identification

For each selected sugar, an attempt is made to identify sugar ring atoms. The sugar oxygen is identified as an oxygen atom bonded to two other carbon atoms in the same residue, referred to here as C0 and C1. The command checks if the three atoms are part of a ring by recursively searching from C0 to C1 (not going through the oxygen). If the oxygen, C0 and C1 are confirmed as ring atoms, C0 or C1 is chosen as the anomeric carbon by being the one with fewer bonds to other carbons, with the tie going to the atom with the lowest index, then the other atom is designated the ring end atom. The entire carbon chain is then identified by searching before the anomeric carbon and after the ring end atom. The ring is then designated pyranose or furanose depending on the number of ring atoms.

Potential stereocenters are then identified. A carbon in the chain is designated as a potential stereocenter if it has at least three bonds, all to unique entities (identified using the "atommap" functionality of CPPTRAJ[12]). The highest stereocenter in the chain is designated as the configurational carbon, and the highest stereocenter in the ring is designated as the anomeric reference atom.

2.5 | Missing sugar linkage identification

The command then checks for missing glycosidic linkages to the anomeric carbon. It checks the distances from the anomeric carbon to the first atom of every other non-solvent residue; if closer than 8.0 Å, the distances from the anomeric carbon to every atom of that residue are checked. If a distance is less than an internally determined cutoff based on the elements of the atoms involved, a bond between those atoms is added.

2.6 | Functional group identification and preparation

In the GLYCAM force field, certain functional groups are designated as separate residues. The command will check for terminal groups, that is, functional groups that are bonded to the anomeric carbon (e.g., a terminal hydroxyl or methyl), as well as functional groups attached to sugar oxygen atoms (e.g., sulfate, methyl, or acetyl). Any group of atoms identified as a functional group is split off as a new residue, with residue and atom names that correspond to the GLYCAM force field.

For example, if a hydroxyl group is bound to the anomeric atom of a sugar, the oxygen atom (and hydrogen if present) is split off into a separate "ROH" residue.

2.7 | Sugar type, anomer type, and configuration determination

First, the base sugar type (and corresponding GLYCAM residue code) is determined from the PDB residue name. If sugar atoms could not be identified in the previous step (because e.g., some are missing), the anomer type (alpha/beta) and configuration (D/L) are also determined from the PDB residue name. Users can indicate that the sugars already have GLYCAM residue names; in that case the command still follows the following procedures to check that the residue name is consistent with the atoms present. This provides a key validation step.

If sugar atoms were previously identified, anomer type and configuration are determined using standard conventions for carbohydrates[14] as follows. For furanoses, configuration is determined from the chirality around the configurational carbon (R = D, S = L). Anomer type is determined by comparing the chirality around the anomeric carbon to the chirality around the configurational carbon; if it matches the type is beta, otherwise the type is alpha. For pyranoses, orientation at the anomeric carbon, anomeric reference carbon, and configurational carbon (if different from the anomeric reference) is determined via improper torsions. For the anomeric carbon it is RO-AC-A0-AX, where RO is the ring oxygen atom, AC is the anomeric carbon, A0 is the non-oxygen ring atom bonded to the anomeric carbon, and AX is the anomeric carbon non-ring substituent. For the anomeric reference carbon it is A1-AR-A2-AY where A1 is the ring atom preceding the anomeric reference, AR is the anomeric reference, A2 is the ring atom after the anomeric reference, and AY is the anomeric reference non-ring substituent. For the configurational carbon it is C0-C-Z-C1 where C0 is the carbon preceding C in the chain, C1 is the carbon after C in the chain, and Z is the non-hydrogen substituent of C with highest priority (determined by atomic number); this is done to be consistent with the atom ordering around the anomeric reference carbon. Based on the torsion being positive or negative, the orientation around these atoms is considered "up" or "down," respectively. If the orientations around the anomeric and anomeric reference carbons match, the type is beta, otherwise the type is alpha. If the orientation around the configurational carbon is down, the configuration is D, otherwise it is L.

If the detected anomer type/configuration does not match what is indicated by the name, a warning is printed. This is sometimes indicated in the PDB itself as a CAVEAT warning that certain atoms have the wrong chirality (e.g., 1glm, 1agm, 1dva, and 1g1s).[15] If desired, users can specify that the anomer type/configuration of sugars should be chosen based on the residue name instead of geometry. During relaxation of the system, the sugar should relax to the specified anomer type. An example of this using the PDB 1g1s is provided in the Supporting Information. It is important to note however that the user should verify that the final confirmation of the sugar matches the desired anomer type in such cases.

An important note for pyranoses is that the correspondence of "up" and "down" descriptors used in 3D to "left" and "right" descriptors in Fischer projection notation changes based on the relative positions of the carbons in question. For example, if the anomeric reference is an even number of atoms away from the anomeric carbon (e.g., in beta-D-xylopyranose), "up" and "down" are reversed for the purposes of determining anomer type.

2.8 | Sugar linkage determination

The GLYCAM linkage code is determined by determining which atoms in a sugar are bonded to other residues, and the position of these bonds with respect to the overall carbon chain. A "linkage string" is created with a bond to the anomeric carbon denoted as "T," and bonds to sugar oxygens denoted as "OX" where "X" denotes the position of the carbon the oxygen is bonded to. So for example, a sugar bonded at its anomeric carbon and C4 oxygen would have the link string "TO4." This string is then translated into the corresponding GLYCAM linkage code (see Supporting Information). If the linkage cannot be determined, a warning is printed. If a sugar is bonded to a non-sugar that is a recognized covalent "link" residue (e.g., a sugar residue bonded to a serine amino acid residue), the link residue name is changed to a GLYCAM recognized one (e.g., "OLS" for serine "SER"; see Supporting Information).

Once all linkages have determined all bonds from sugars to other residues are removed (since LEaP cannot handle branched structures) and the appropriate LEaP bond commands to recreate the bonds inside of LEaP are generated and written to a file.

2.9 | Assigning GLYCAM residue name

Once the sugar type, anomer type, configuration, and linkages have been determined, the GLYCAM residue name is constructed from the GLYCAM linkage code, the GLYCAM residue code, and the GLYCAM anomer type code. The linkage code was described in the previous section. The GLYCAM residue code is determined from a data file that contains mapping of PDB residue names to GLYCAM residue codes (see Supporting Information), upper case for D configuration and lower case for L configuration. The GLYCAM anomer type code is "A" (pyranose) or "D" (furanose) for alpha, and "B" (pyranose) or "U" (furanose) for beta. Any PDB atom names that need to be changed for GLYCAM are also done via a data file that contains name mapping (see Supporting Information).

2.10 | Final steps

The command will attempt to validate the structure and print helpful warnings to bring issues that may prevent the system from being built to the user's attention. These include residue names that may be unrecognized (and therefore may not have parameters), mismatches between detected sugar anomer type/configuration and the anomer type/configuration based on the sugar residue name, unrecognized glycosidic or sugar to non-sugar covalent linkages, and so on. Warnings will also be printed to indicate if any charges will need to be modified for functional groups after LEaP is run in order to retain the proper charge on the sugar. The command can be told to call LEaP directly, after which the command can automatically take care of any charge adjustments required by attached functional groups. For example, sulfate groups need to have the charge on the bonded oxygen adjusted by $+0.031$ e$-$, methyl groups need the charge on the carbon bonded to the bonded oxygen adjusted by -0.039 e$-$, and acetyl groups need the charge on the carbon bonded to the bonded oxygen adjusted by $+0.008$ e$-$.

3 | TEST SYSTEMS

The command was tested on 196 structures from the PDB containing both N- and O-linked carbohydrates (the PDB IDs used can be found in Supporting Information). Of these, 141 contained non-standard non-polymer residues (i.e., residues that are not one of the standard amino/nucleic acid or sugar residues found in Amber force fields), eight contained unrecognized modified sugars (e.g., deoxy sugars or sugars containing modifications not yet supported by GLYCAM like 4-thio-beta-D-glucopyranose), and eight contained both non-polymer and unrecognized sugar residues. A table of the names of these residues broken down by type can be found in the Supporting Information. All solvent (residues named HOH) and hydrogen atoms were removed by the command.

If the non-standard and unsupported modified sugar residues are removed, the command successfully processes 159 of the 196 structures (81%). A success is defined as LEaP being able to successfully generate working topology and coordinate files solely from the PDB file along with the LEaP input written from the command using a standard Amber protein and carbohydrate force field (FF14SB[16] and GLYCAM06j-1,[13] respectively); the final topology must contain proper glycosidic and sugar to non-sugar covalent linkages, disulfide bonds, and so forth. Of the 37 that fail, 18 are due to sugars linked to amino acids that do not have covalent glycosylation (sugar-protein covalent linkage) forms in GLYCAM (TRP, TYR, and LYS), 11 fail because they contain sugars with glycosidic linkages that are currently unsupported by GLYCAM (for example, PDB 2YGQ has a beta-D-furanose YYJ_2_B linked at O1, O2, O3, O4, and O6, which according to GLYCAM nomenclature should have the residue name PCU, but no such residue exists in GLYCAM06j-1), three contain sugars which could not be completely identified because they are missing key atoms and/or linkages (e.g., nothing is bonded to the C1 atom), and the remaining four have some combination of the aforementioned problems. It should be noted that some of the unrecognized glycosidic linkage issues are related to the removal of unsupported sugars; for example, in PDB 1QJW a GLC residue is missing glycosidic linkage to the C1 atom due to a removed SGC (4-thio-beta-D-glucopyranose) residue. The one remaining failure is PDB 6RS6 which contains isolated glycine residues (residues 308 and 309), which are not natively supported by the Amber protein force field (due to OXT atom not being present in the GLY residue templates). If these isolated GLY residues are removed, the PDB can be processed successfully.

It is important to note that all PDBs that fail do so because they contain residues which are not present in the protein/nucleic/carbohydrate force fields natively included with Amber. Parameters for these residues would have to be carefully considered, and could ultimately be obtained via another method (e.g., by using the antechamber program with GAFF[17]).

3.1 | Applied test case: NISTmAb

The NISTmAb was used to benchmark the "prepareforleap" command in CPPTRAJ. The atomic model structure of the NISTmAb was based on human and murine IgG1κ monoclonal antibody structures, and accompanies the highly characterized NISTmAb reference material.[18] The structure of the mAb consists of four chains total, with two each heavy and light chain, linked through inter and intra-chain disulfide

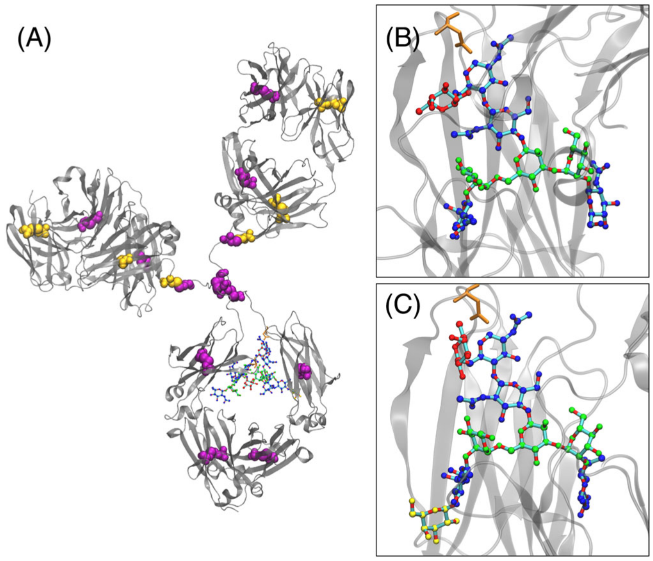

FIGURE 1 Automatic assignments from "prepareforleap" on a complex IgG1. (A) NISTmAb model structure with assigned disulfides in van der Waals sphere representation, where purple are heavy chain and gold are light chain cysteines. Close up of (B) G0F glycoform and (C) G1F glycoform, colored by glycan type (red fucose, green mannose, blue GlcNAc, and yellow galactose following standard symbol nomenclature for glycans). Asn300 attachment residues are represented in orange licorice

bonds. Additionally, the model contains glycans covalently linked at each heavy chain's Asn300 residue. Two different glycoforms are present: G0F and G1F.[19]

"prepareforleap" was initiated using the "noh" keyword to remove hydrogen atoms and specifying that LEaP and PDB formatted output files be written. The output LEaP instructions were appended to a file sourcing both the FF14SB and GLYCAM06j-1 force fields, and successfully used to build the parameter/topology and coordinate files. Figure 1 illustrates the modifications "prepareforleap" now automatically handles. Figure 1A shows the overall fold of the NISTmAb, where both inter and intra-strand disulfides are automatically assigned, since the native PDB is missing disulfide link records. The Asn300 residue attachment points for the N-linked glycans (shown in orange) have been renamed and the glycans explicitly bonded. Glycans, shown close-up in Figure 1B,C, have been translated from accepted PDB nomenclature to GLYCAM notation. Details are printed to the CPPTRAJ stdout, and include PDB notation, expanded notation, GLYCAM notation, stereochemistry (L/D), linkage (alpha/beta) and linking atoms. This is significant because the glycoforms differ for each N-linked attachment point, with Figure 1B showing built G0F glycoform and Figure 1C showing the G1F glycoform.

4 | CONCLUSIONS

The new "prepareforleap" command implemented in CPPTRAJ can greatly facilitate preparing systems from PDB files for successful processing by the Amber program LEaP in order to run simulations with the Amber MD software package. The command deals with common issues such as choosing alternate atom locations, identifying disulfide bonds, and removing solvent residues/hydrogen atoms, but arguably its greatest strength is in facilitating the automatic preparation of carbohydrates (through identifying anomer type, configuration (D or L), glycosidic linkages, protein/sugar linkages, and functional groups) for use with the GLYCAM force field. The command was able to successfully process the majority (>80%) of PDBs once residues for which no parameters exist in the force fields included with Amber are removed, and successfully processed all PDBs (100%) where glycosidic (sugar to sugar) and sugar to non-sugar covalent linkages were supported by the GLYCAM force field. Processing is also extremely fast; the average execution time was 0.23 s ± 0.54 s. We expect the implementation of "prepareforleap" to considerably decrease time to simulation burden for structural biologists, as well as decrease errors associated with manual conversion of glycan parameters.

ACKNOWLEDGMENTS

This work was supported in part by the intramural research program of the National Heart, Lung and Blood Institute (NHLBI) of the National Institutes of Health, NHLBI Z01 HL001051-23, and by the NIST Biomanufacturing Program. Computational resources provided by the LoBoS cluster (NIH/NHLBI).

DISCLAIMER

Certain commercial equipment, instruments, or materials are identified in this paper to foster understanding. Such identification does not imply recommendation or endorsement by the National Institute of Standards and Technology, nor does it imply that the materials or equipment identified are necessarily the best available for the purpose.

DATA AVAILABILITY STATEMENT

The source code used in this study is openly available in the CPPTRAJ GitHub repository at https://github.com/Amber-MD/cpptraj. The PDBs used in this study are openly available in the RSCB PDB at https://www.rcsb.org/. Other data that support the findings of this study are available from the corresponding author upon reasonable request.

ORCID

Daniel R. Roe https://orcid.org/0000-0002-5834-2447
Christina Bergonzo https://orcid.org/0000-0003-1990-2912

REFERENCES

[1] L. Casalino, Z. Gaieb, J. A. Goldsmith, C. K. Hjorth, A. C. Dommer, A. M. Harbison, C. A. Fogarty, E. P. Barros, B. C. Taylor, J. S. McLellan, E. Fadda, R. E. Amaro, *ACS Cent. Sci.* **2020**, *6*, 1722. https://doi.org/10.1021/acscentsci.0c01056
[2] R. J. Woods, R. A. Dwek, C. J. Edge, B. Fraser-Reid, *J. Phys. Chem.* **1995**, *99*, 3832. https://doi.org/10.1021/j100011a061
[3] D. A. Case, T. E. Cheatham, T. Darden, H. Gohlke, R. Luo, K. M. Merz, A. Onufriev, C. Simmerling, B. Wang, R. J. Woods, *J. Comput. Chem.* **2005**, *26*, 1668. https://doi.org/10.1002/jcc.20290

[4] S. B. Engelsen, P. I. Hansen, S. Pérez, *Biopolymers* **2014**, *101*, 733. https://doi.org/10.1002/bip.22449

[5] M. M. Kuttel, J. Ståhle, G. Widmalm, *J. Comput. Chem.* **2016**, *37*, 2098. https://doi.org/10.1002/jcc.24428

[6] R. Danne, C. Poojari, H. Martinez-Seara, S. Rissanen, F. Lolicato, T. Róg, I. Vattulainen, *J. Chem. Inf. Model.* **2017**, *57*, 2401. https://doi.org/10.1021/acs.jcim.7b00237

[7] J. W. Labonte, J. Adolf-Bryfogle, W. R. Schief, J. J. Gray, *J. Comput. Chem.* **2017**, *38*, 276. https://doi.org/10.1002/jcc.24679

[8] A. Bohne-Lang, C. W. Von der Lieth, *Nucleic Acids Res.* **2005**, *33*, W214. https://doi.org/10.1093/nar/gki385

[9] GLYCAM Web. http://glycam.org

[10] S. J. Park, J. Lee, Y. Qi, N. R. Kern, H. S. Lee, S. Jo, I. Joung, K. Joo, J. Lee, W. Im, *Glycobiology* **2019**, *29*, 320. https://doi.org/10.1093/glycob/cwz003

[11] J. Lee, X. Cheng, J. M. Swails, M. S. Yeom, P. K. Eastman, J. A. Lemkul, S. Wei, J. Buckner, J. C. Jeong, Y. Qi, S. Jo, V. S. Pande, D. A. Case, C. L. Brooks, A. D. MacKerell, J. B. Klauda, W. Im, *J. Chem. Theory Comput.* **2016**, *12*, 405. https://doi.org/10.1021/acs.jctc.5b00935

[12] D. R. Roe, T. E. Cheatham, *J. Chem. Theory Comput.* **2013**, *9*, 3084. https://doi.org/10.1021/ct400341p

[13] K. N. Kirschner, A. B. Yongye, S. M. Tschampel, J. González-Outeiriño, C. R. Daniels, B. L. Foley, R. J. Woods, *J. Comput. Chem.* **2008**, *29*, 622. https://doi.org/10.1002/jcc.20820

[14] *J. Carbohydr. Chem.* **1997**, *16*, 1191. https://doi.org/10.1080/07328309708005748

[15] J. Agirre, G. Davies, K. Wilson, K. Cowtan, *Nat. Chem. Biol.* **2015**, *11*, 303. https://doi.org/10.1038/nchembio.1798

[16] J. A. Maier, C. Martinez, K. Kasavajhala, L. Wickstrom, K. E. Hauser, C. Simmerling, *J. Chem. Theory Comput.* **2015**, *11*, 3696. https://doi.org/10.1021/acs.jctc.5b00255

[17] J. Wang, R. M. Wolf, J. W. Caldwell, P. A. Kollman, D. A. Case, *J. Comput. Chem.* **2004**, *25*, 1157. https://doi.org/10.1002/jcc.20035

[18] C. Bergonzo, D. T. Gallagher, *J. Res. Natl. Inst. Stand. Technol.* **2021**, *126*, 126012. https://doi.org/10.6028/jres.126.012

[19] M. Kiyoshi, K. Tsumoto, A. Ishii-Watabe, J. M. M. Caaveiro, *Int. Immunol.* **2017**, *29*, 311. https://doi.org/10.1093/intimm/dxx038

SUPPORTING INFORMATION

Additional supporting information may be found in the online version of the article at the publisher's website.

How to cite this article: D. R. Roe, C. Bergonzo, *J. Comput. Chem.* **2022**, 1. https://doi.org/10.1002/jcc.26847