ProtoCaller: Robust Automation of Binding Free Energy Calculations

Miroslav Suruzhon[†], *Tharindu Senapathi*[§], *Michael S. Bodnarchuk*[ϟ], *Russell Viner*[¶], *Ian D. Wall*[‡], *Christopher B. Barnett*[§], *Kevin J. Naidoo*[§,‖], *Jonathan W. Essex*[†*]

[†]School of Chemistry, University of Southampton, Highfield, Southampton SO17 1BJ, United Kingdom

[§]Scientific Computing Research Unit and Department of Chemistry, University of Cape Town, Rondebosch 7701, South Africa

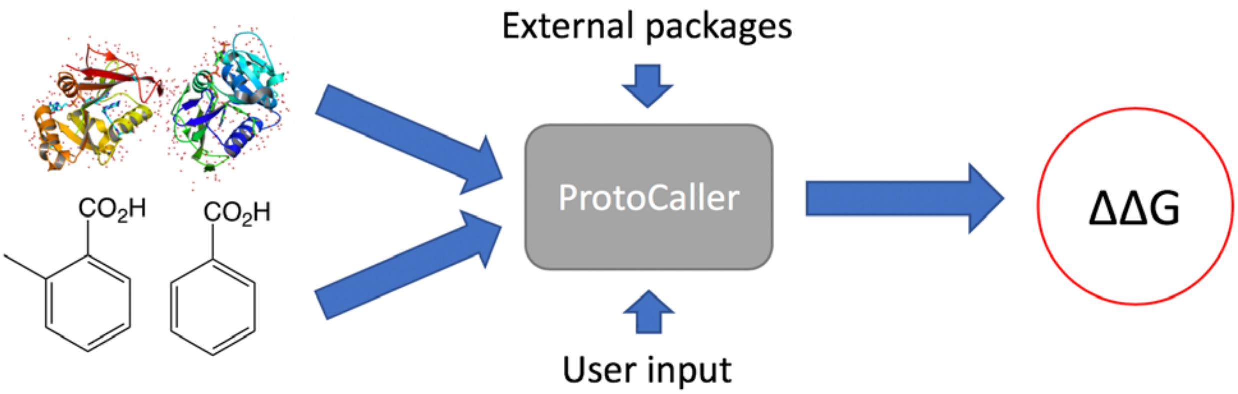
[ϟ]Computational Chemistry, R&D Oncology, AstraZeneca, Cambridge CB4 0WG, United Kingdom

[¶]Syngenta, Jealott's Hill International Research Centre, Bracknell RG42 6EY, United Kingdom

[‡]GSK Medicines Research Centre, Gunnels Wood Road, Stevenage SG1 2NY, United Kingdom

[‖]Institute for Infectious Disease and Molecular Medicine, Faculty of Health Science, University of Cape Town, Rondebosch 7701, South Africa

ABSTRACT



ProtoCaller is a Python library distributed through Anaconda which automates relative protein-ligand binding free energies in GROMACS. It links a number of popular specialized tools used to perform protein setup and parametrization, such as PDB2PQR, Modeller and AmberTools. ProtoCaller supports commonly used AMBER force fields with additional cofactor parameters, and AM1-BCC is used to derive ligand charges. ProtoCaller also comes with an extensive PDB parser, an enhanced maximum common substructure algorithm providing improved ligand-ligand mapping and a light GROMACS wrapper for running multiple molecular dynamics simulations. ProtoCaller is highly relevant to most researchers in the field of biomolecular simulation, allowing a customizable balance between automation and user intervention.

INTRODUCTION

Protein-ligand binding free energy simulations are an increasingly promising approach for facilitating drug discovery[1]. Current computational power makes high-throughput free energy calculations increasingly more practical, thereby enabling the streamlined use of atomistic theory to complex biochemical problems. Automating this process is therefore key for improving the general applicability of free energy methods.

Unfortunately, it is not always feasible to completely automate the setup of every protein-ligand system in an unbiased way. Each system setup requires multiple steps with varying degrees of user intervention. This means that system preparation is arguably more time-consuming than data generation and has been suggested to be a crucial step in determining the resulting free energy[2]. Therefore, one needs a “grey-box” approach, where it is possible to achieve full automation of the setup process, while controlling an arbitrary number of intermediate steps and parameters depending on the user’s needs.

Another issue is software interoperability. Linking together different specialized pieces of software is an undesirable but necessary task which is usually solved by using in-house scripts or commercial software. This is also prone to human errors and can quickly become unmanageable when one starts interfacing with different pieces of software.

Several tools exist which tackle these issues. Notably, YANK[3] provides a fully-automated workflow from system setup to computation of absolute free energies. Protein Preparation Wizard[4] and HTMD[5] also handle system preparation for relative free energies in a robust way, providing seamless links to commercial molecular dynamics (MD) engines. Finally, alchemical setup[6], FESetup[7] and pmx[8] are open-source tools which automate system setup for relative free energy calculations on multiple MD engines.

In this paper we describe ProtoCaller, an open-source conda-installable Python library which attempts to solve the above challenges by providing a customizable unified interface to all of the steps of the free energy workflow. It utilizes freely available specialized libraries to set up and perform relative free energy calculations in an open-source MD engine, GROMACS[9]. Moreover, its modular nature means that the user could feasibly tailor it to their needs, even if it is not currently directly supported by the software. For example, ProtoCaller provides sufficient flexibility for performing calculations outside its originally intended scope, such as absolute and relative solvation free energies and simulations in MD engines supported by BioSimSpace[10], such as Sire[11] (addressed later).

The next section will describe the workflow in more detail, whilst giving an overview of the main algorithms and procedures used in the library.

PROTOCALLER

Protein Preparation

The first step of the workflow (Fig. 1) is choosing the protein crystal structure from the Protein Data Bank[12]. Since there are often a large number of relevant crystal structures, the typical guideline is to choose the protein crystal structure which has a bound ligand with a structure closest to the ligand of interest. This part of the workflow determines the initial ligand binding orientation making it the most crucial step. In ProtoCaller, one can either use a plain PDB code or provide a user-specified PDB file.

Experimental crystal structures are obtained using sophisticated models bridging theory and experiment. Because of this, it is very likely that some parts of the structure will be less reliable than others. For example, it is common that there will be missing atoms and residues which require modelling. In ProtoCaller, several tools achieving part or all of this task have been linked: Modeller[13], pdbfixer[14], CHARMM-GUI[15] and PDB2PQR[16], providing the user with the ability to choose the most appropriate tool for their system. The latter also removes steric clashes in the crystal structure and optimizes hydrogen bonding by rotating His, Asn and Gln residues.

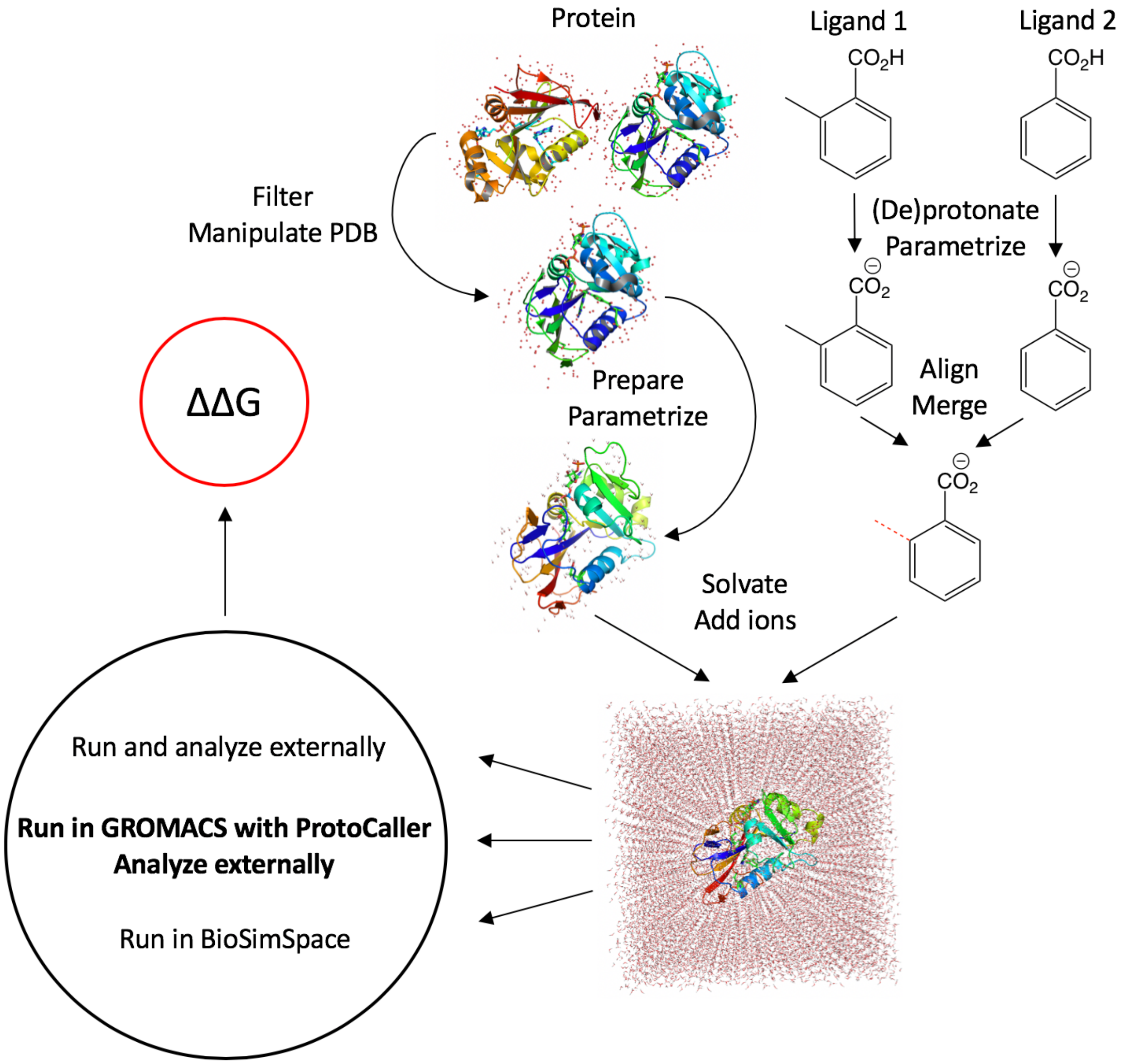


Figure 1. The full workflow for obtaining a relative binding free energy. Each of these steps utilizes one or more specialized tools. In this scheme ”prepare” stands for the addition of missing residues and atoms, protonation and removing steric clashes.

To build an interface to all of the above pieces of software, an extensive PDB parser was developed, which allows the user to manipulate PDB files in an object-oriented fashion inside Python. This parser allows seamless communication between different modules inside the software without any loss of relevant header data. This can also be extremely useful when one deals with incomplete PDB files or simply wants to introduce changes in the file in a way which is consistent with ProtoCaller.

The next step is the protonation of the protein. This is particularly difficult, since X-ray crystallography does not usually locate hydrogen nuclei and they must be modelled. Moreover, while in most cases the amino acid protonation state is straightforward to determine, the acidic and basic amino acids can exist in several protonation states. This is especially problematic for histidine, which has three different protonation states, all of which may be relevant at equilibrium. Finally, a good algorithm should be able to distinguish scenarios where there are exceptions - e.g. protonated aspartic acid in an Asp-Asp dyad[17]. There are several approaches which attempt to deal with protein protonation. The program of choice in ProtoCaller is PROPKA3.1[18], as utilized by PDB2PQR because of its extensive functionality in protein preparation and ease of incorporation into the workflow.

Ligand Protonation

Similarly, the protonation state of the ligands in the system must be determined as well. Ideally, one should consider the protein-ligand complex as one entity while performing protonation. However, in practice, this can be sensitive to the ligand binding mode which is often unknown *a priori*. In ProtoCaller, ligand protonation is thus performed separately from the protein using Open Babel[19]. Alternatively, one can provide an already protonated ligand as an external file compatible with Open Babel or ParmEd[20]. If the ligands are sufficiently simple such that there are not multiple relevant conformations, one can also use SMILES[21] and InChI[22] strings to define ligands in ProtoCaller and generate starting conformations using Open Babel. These initial conformations are crucial during parametrization, since the charge derivation method is conformation-dependent.

Parametrization

The next step is force field parametrization. In ProtoCaller there is currently only support for the AMBER force field[23]. Since parametrization using AmberTools is straightforward to implement, protein, ligand, water and simple ion parametrizations are directly performed in a relevant wrapper using any supported force field (ff99SB[24], ff14SB[25], GAFF[26], GAFF2 and TIP3P[27] have currently been tested). Atomic charges are derived using the semi-empirical AM1-BCC[28,29] method where any subsequent floating point errors in the total charges are equally distributed between the atoms. In addition, some common cofactor parameters obtained from the AMBER parameter database[30] are also available in ProtoCaller. However, there is currently no support for modified/nonstandard amino acid residues or systems containing transition metals, since these require user intervention in any case and are unfortunately not currently automatable.

Mapping and Alignment

Determining the ligand binding pose is a nontrivial and critical task, even when there is a native ligand with a similar structure. This part of the workflow consists of two steps: determining the maximum common substructure (MCS) and physically aligning the ligand to the reference crystallographic binding pose.

The MCS algorithm, more commonly known as the maximum common subgraph algorithm, is an NP-complete[31] problem whose exact solution requires exponential time with respect to the number of nodes. Therefore, approximate solutions are needed. An open-source implementation is available in RDKit[32] and is the one used in ProtoCaller. However, some modifications in addition to this code were made in ProtoCaller to obtain a physically relevant MCS. First, ProtoCaller currently prohibits mapping between two rings of different sizes due to the nature of the subsequent alignment algorithm and the difficulties in opening a ring using a single-topology protocol[33]. Second, mapping of an acyclic chain to a reference ring is allowed, but the reverse is not, due to the hard position constraints imposed by the subsequent alignment algorithm. Finally, if there are chiral centers of different chiralities, only the longest MCS segment between two such atoms is considered a valid common substructure, which may or may not be the maximum one. The reason for this is that MCS is purely a graph theoretical algorithm with no regard to stereoisomers and the results from the algorithm have to be pruned to be physically relevant.

The alignment process is based on constraining the MCS atoms to the reference positions (e.g. crystal ligand binding orientation) and performing a force field minimization using MMFF in RDKit[34] on the rest of the atoms whilst preserving all bond angles outside of the MCS to their prior values. Afterwards, all of the external rotatable bonds of the target molecule are rotated until an optional target metric is minimized. The current heuristic metric used in ProtoCaller is a generalized squared error ε, computed in the following way:

$$\varepsilon = \sum_{i=1}^{N_{ref}} \sum_{j=1}^{N_{mol}} \left|\boldsymbol{X}_{ref,i} - \boldsymbol{X}_{mol,j}\right|^2 - \sum_{i \neq j}^{N_{mol}} \frac{1}{\left|\boldsymbol{X}_{mol,i} - \boldsymbol{X}_{mol,j}\right|^{12}} \quad (1)$$

Where N_{ref} and N_{mol} refer to the number of atoms in the reference molecule and the aligned molecule respectively, and $\mathbf{X}_{ref}$ and $\mathbf{X}_{mol}$ refer to the position vectors in both molecules. Here the first term ensures a good spatial match between the two molecules, while the second is a repulsion term which penalizes steric clashes arising from the clustering of target atoms. In

ProtoCaller, the second term is ignored above 1 Å, due to its negligible contributions to the sum and all MCS atoms are ignored in the first term due to the already imposed hard constraints. Finally, if there are several MCSs of the same length, those that maximize matching between the same atom types are prioritized over the others. Of these, the one with the lowest ε is then chosen as the optimal alignment. It has to be noted that this metric should be used with caution and is only relevant when there are small deviations from the reference crystallographic binding pose. Otherwise, a custom binding orientation set by the user is highly recommended.

Solvation and Simulation

Afterwards, solvation, neutralization and NaCl addition to the system is performed using GROMACS. It has to be noted that in each case the resulting structures have to be minimized before any simulations, since there might be some structural distortions introduced by the addition of missing amino acids or the constrained ligand mapping. The resulting output provides files for the complete free energy cycle, which can be run externally using GROMACS. Alternatively, there are several routines in ProtoCaller which provide presets for some typical protocols for performing an MD simulation in GROMACS at some user-specified simulation parameter values. One can afterwards run external tools to analyze the resulting energy files to obtain a free energy, such as the native Bennett acceptance ratio[35] implementation in GROMACS (gmx bar) or the alchemical analysis script[36] available online[37]. Another possibility is obtaining the resulting system as a BioSimSpace object which can then be manipulated inside the library. At the time of writing, BioSimSpace supports free energy calculations in GROMACS and Sire.

CODE STRUCTURE AND USAGE

Central to ProtoCaller is the Ensemble class which contains a Protein and a list of Perturbations. Each Protein contains all information about a protein (including a PDB file object) and all of its associated cofactors, ions and ligands, and is responsible for calling all routines relating to protein preparation. Similarly, every ligand and cofactor is represented by the Ligand class, which completely handles ligand preparation. Both Protein and Ligand perform parametrization using the auxiliary Params class, which is a simple descriptor of the force fields used. The Perturbation class consists of two Ligands and performs all methods related to ligand mapping and alignment. Simulations are performed through the RunGMX class which is responsible for calling GROMACS using MD parameters from associated Protocol objects. A more detailed description of the code structure and associated examples can be found on https://protocaller.readthedocs.io.

Since ProtoCaller is a Python library, the user interacts with the software by writing a Python script which can then be run on any supported platform. ProtoCaller is thus readily usable on any server with a job submission system, as long as the user loads the appropriate conda environment before calling the Python interpreter.

TEST SYSTEMS

To demonstrate the capabilities of ProtoCaller, we chose two systems which can be considered to be moderately challenging to set up. The first test case is dihydrofolate reductase, a protein which contains a cofactor with known parameters. In this test case we consider two crystal structures (PDB codes 5HPB[38] and 6DAV[39]), where we demonstrate ProtoCaller's capabilities to select a single protein chain and even select atoms with several possible locations. Moreover, we show how one can use a user-specified ligand orientation in the binding pocket. The second test case is coagulation factor X (PDB code 2P3T[40]), where there are several missing residues which require modelling using Modeller, disulfide bonds, a calcium ion and a chloride ion which we exclude from the simulation with little effort. Both of these systems have been described in more detail in the Supporting Information. After preparing the initial structures, we also validated ProtoCaller for a variety of ligand perturbations by calculating cycle closure errors and comparing ΔΔG values to experimental data. In all cases the cycle closure errors were less than 1 kcal mol^{-1} and the mean absolute deviation to experiment was satisfactory for the force field, ranging between 0.47 and 1.33 kcal mol^{-1}.

FINAL NOTES AND CONCLUSIONS

In this work we have described ProtoCaller, a Python library which links existing free tools to aid high-throughput relative binding free energy calculations with an arbitrary degree of automation and user involvement. We hope that it will be useful to the free energy community by facilitating the tedious parts of the system setup and allowing researchers to focus on the science, rather than the technicalities. ProtoCaller is available to use in Linux and macOS. Even though ProtoCaller was thoroughly tested to the best of our resources, issues with the code are inevitable and users are encouraged to submit any notes or code contributions to the Github page https://github.com/protocaller/protocaller.

ASSOCIATED CONTENT

Methods and test cases (PDF)

All of the files, python scripts and a README file which describes how to regenerate all of the presented data (ZIP)

AUTHOR INFORMATION

Corresponding Author

*Jonathan W. Essex. E-mail: J.W.Essex@soton.ac.uk

ACKNOWLEDGMENT

All of the simulations in this study have been performed on the Iridis 5 computing cluster at the University of Southampton. We thank AstraZeneca, GSK and Syngenta for funding this study. We are also grateful for the support from the EPSRC Centre for Doctoral training, Theory and Modelling in Chemical Sciences, under grant EP/L015722/1. KJN is supported by the South African Research Chairs Initiative (SARChI) of the Department of Science and Technology (DST) and National Research Foundation (NRF) grant 48103. We finally thank Lester Hedges for general help with BioSimSpace.

ABBREVIATIONS

AM1, Austin model 1; BCC, bond charge correction; InChI, IUPAC international chemical identifier; MCS, maximum common substructure; MD, molecular dynamics; MMFF, Merck molecular force field; NP, nondeterministic polynomial time; PDB, protein data bank; SMILES, simplified molecular-input line-entry system.

REFERENCES

(1) Wang, L.; Chambers, J.; Abel, R. Protein-Ligand Binding Free Energy Calculations with FEP. *Methods Mol. Biol.* **2019**, *2022*, 201–232.

(2) Pérez-Benito, L.; Casajuana-Martin, N.; Jiménez-Rosés, M.; van Vlijmen, H.; Tresadern, G. Predicting Activity Cliffs with Free-Energy Perturbation. *J. Chem. Theory Comput.* **2019**, *15*, 1884–1895.

(3) Wang, K.; Chodera, J. D.; Yang, Y.; Shirts, M. R. Identifying Ligand Binding Sites and Poses Using GPU-Accelerated Hamiltonian Replica Exchange Molecular Dynamics. *J. Comput. Aided Mol. Des.* **2013**, *27*, 989–1007.

(4) Schrödinger Release 2019-3: Schrödinger Suite 2019-2 Protein Preparation Wizard; Epik, Schrödinger, LLC, New York, NY, 2019; Impact, Schrödinger, LLC, New York, NY, 2019; Prime, Schrödinger, LLC, New York, NY, 2019.

(5) Doerr, S.; Harvey, M. J.; Noé, F.; De Fabritiis, G. HTMD: High-Throughput Molecular Dynamics for Molecular Discovery. *J. Chem. Theory Comput.* **2016**, *12*, 1845–1852.

(6) Klimovich, P. V.; Mobley, D. L. A Python Tool to Set up Relative Free Energy Calculations in GROMACS. *J. Comput. Aided Mol. Des.* **2015**, *29*, 1007–1014.

(7) Loeffler, H. H.; Michel, J.; Woods, C. FESetup: Automating Setup for Alchemical Free Energy Simulations. *J. Chem. Inf. Model.* **2015**, *55*, 2485–2490.

(8) Gapsys, V.; Michielssens, S.; Seeliger, D.; de Groot, B. L. Pmx: Automated Protein Structure and Topology Generation for Alchemical Perturbations. *J. Comput. Chem.* **2015**, *36*, 348–354.

(9) Abraham, M. J.; Murtola, T.; Schulz, R.; Páll, S.; Smith, J. C.; Hess, B.; Lindahl, E. GROMACS: High Performance Molecular Simulations through Multi-Level Parallelism from Laptops to Supercomputers. *SoftwareX* **2015**, *1-2*, 19–25.

(10) Hedges, L. O.; Mey, A. S. J. S.; Laughton, C. A.; Gervasio, F. L.; Mulholland, A. J.; Woods, C. J.; Michel, J. BioSimSpace: An Interoperable Python Framework for

Biomolecular Simulation. *J. Open Source Softw.* **2019**, *4*, 1831–1841.

(11) Woods, C. J.; Michel, J. M. Sire: An advanced, multiscale, molecular simulation framework http://siremol.org/ (accessed Aug 15, 2019).

(12) Berman, H. M.; Battistuz, T.; Bhat, T. N.; Bluhm, W. F.; Bourne, P. E.; Burkhardt, K.; Feng, Z.; Gilliland, G. L.; Iype, L.; Jain, S.; Fagan P.; Marvin J.; Padilla D.; Ravichandran V.; Schneider B.; Thanki N.; Weissig H.; Westbrook J. D.; Zardecki C. The Protein Data Bank. *Acta Crystallogr., Sect. D: Biol. Crystallogr.* **2002**, *58*, 899–907.

(13) Webb, B.; Sali, A. Protein Structure Modeling with MODELLER. *Methods Mol. Biol.* **2017**, *1654*, 39–54.

(14) Eastman, P. Pdbfixer https://github.com/pandegroup/pdbfixer (accessed Aug 15, 2019).

(15) Jo, S.; Cheng, X.; Islam, S. M.; Huang, L.; Rui, H.; Zhu, A.; Lee, H. S.; Qi, Y.; Han, W.; Vanommeslaeghe, K.; MacKerell A. D.; Roux B.; Im W. CHARMM-GUI PDB Manipulator for Advanced Modeling and Simulations of Proteins Containing Nonstandard Residues. *Adv. Protein Chem. Struct. Biol.* **2014**, *96*, 235–265.

(16) Unni, S.; Huang, Y.; Hanson, R. M.; Tobias, M.; Krishnan, S.; Li, W. W.; Nielsen, J. E.; Baker, N. A. Web Servers and Services for Electrostatics Calculations with APBS and PDB2PQR. *J. Comput. Chem.* **2011**, *32*, 1488–1491.

(17) Bas, D. C.; Rogers, D. M.; Jensen, J. H. Very Fast Prediction and Rationalization of pKa Values for Protein-Ligand Complexes. *Proteins* **2008**, *73*, 765–783.

(18) Olsson, M. H. M.; Søndergaard, C. R.; Rostkowski, M.; Jensen, J. H. PROPKA3: Consistent Treatment of Internal and Surface Residues in Empirical pKa Predictions. *J. Chem. Theory Comput.* **2011**, *7*, 525–537.

(19) O'Boyle, N. M.; Banck, M.; James, C. A.; Morley, C.; Vandermeersch, T.; Hutchison, G. R.

Open Babel: An Open Chemical Toolbox. *J. Cheminform.* **2011**, *3*, 33.

(20) Swails, J.; Hernandez, C.; Mobley, D.; Nguyen, H.; Wang, L.; Janowski, P. ParmEd https://github.com/ParmEd/ParmEd (accessed Aug 15, 2019).

(21) Weininger, D. SMILES, a Chemical Language and Information System. 1. Introduction to Methodology and Encoding Rules. *J. Chem. Inf. Model.* **1988**, *28*, 31–36.

(22) Heller, S.; McNaught, A.; Stein, S.; Tchekhovskoi, D.; Pletnev, I. InChI - the Worldwide Chemical Structure Identifier Standard. *J. Cheminform.* **2013**, *5*, 7.

(23) Cornell, W. D.; Cieplak, P.; Bayly, C. I.; Gould, I. R.; Merz, K. M.; Ferguson, D. M.; Spellmeyer, D. C.; Fox, T.; Caldwell, J. W.; Kollman, P. A. A Second Generation Force Field for the Simulation of Proteins, Nucleic Acids, and Organic Molecules. *J. Am. Chem. Soc.* **1995**, *117*, 5179–5197.

(24) Lindorff-Larsen, K.; Piana, S.; Palmo, K.; Maragakis, P.; Klepeis, J. L.; Dror, R. O.; Shaw, D. E. Improved Side-Chain Torsion Potentials for the Amber ff99SB Protein Force Field. *Proteins* **2010**, *78*, 1950–1958.

(25) Maier, J. A.; Martinez, C.; Kasavajhala, K.; Wickstrom, L.; Hauser, K. E.; Simmerling, C. ff14SB: Improving the Accuracy of Protein Side Chain and Backbone Parameters from ff99SB. *J. Chem. Theory Comput.* **2015**, *11*, 3696–3713.

(26) Wang, J.; Wolf, R. M.; Caldwell, J. W.; Kollman, P. A.; Case, D. A. Development and Testing of a General Amber Force Field. *J. Comput. Chem.* **2004**, *25*, 1157–1174.

(27) Jorgensen, W. L.; Chandrasekhar, J.; Madura, J. D.; Impey, R. W.; Klein, M. L. Comparison of Simple Potential Functions for Simulating Liquid Water. *J. Chem. Phys.* **1983**, *79*, 926–935.

(28) Jakalian, A.; Bush, B. L.; Jack, D. B.; Bayly, C. I. Fast, Efficient Generation of High-

Quality Atomic Charges. AM1-BCC Model: I. Method. *J. Comput. Chem.* **2000**, *21*, 132–146.

(29) Jakalian, A.; Jack, D. B.; Bayly, C. I. Fast, Efficient Generation of High-Quality Atomic Charges. AM1-BCC Model: II. Parameterization and Validation. *J. Comput. Chem.* **2002**, *23*, 1623–1641.

(30) Bryce, R. AMBER parameter database http://research.bmh.manchester.ac.uk/bryce/amber/ (accessed Aug 15, 2019).

(31) Garey, M. R.; Johnson, D. S. *Computers and Intractability: A Guide to the Theory of NP-Completeness*; W.H. Freeman, 1979.

(32) Landrum, G. RDKit: Open-source cheminformatics http://www.rdkit.org (accessed Aug 15, 2019).

(33) Liu, S.; Wang, L.; Mobley, D. L. Is Ring Breaking Feasible in Relative Binding Free Energy Calculations? *J. Chem. Inf. Model.* **2015**, *55*, 727–735.

(34) Tosco, P.; Stiefl, N.; Landrum, G. Bringing the MMFF Force Field to the RDKit: Implementation and Validation. *J. Cheminform.* **2014**, *6*, 37.

(35) Bennett, C. H. Efficient Estimation of Free Energy Differences from Monte Carlo Data. *J. Comput. Phys.* **1976**, *22*, 245–268.

(36) Klimovich, P. V.; Shirts, M. R.; Mobley, D. L. Guidelines for the Analysis of Free Energy Calculations. *J. Comput. Aided Mol. Des.* **2015**, *29*, 397–411.

(37) Beauchamp, K. A.; Lim, N. M.; Mobley, D. L. Alchemical Analysis https://github.com/MobleyLab/alchemical-analysis (accessed Aug 15, 2019).

(38) Cody, V.; Ganjee, A. Human Dihydrofolate Reductase Complex with NADPH and 5-Methyl-6-(phenylthio-4’trifluoromethyl)thieno[2,3-D]pyrimidine-2,4-Diamine. 2017.

https://doi.org/10.2210/pdb5hpb/pdb.

(39) Mayclin, S. J.; Dranow, D. M.; Lorimer, D. D. Crystal Structure of Human DHFR Complexed with NADP and N10 Formyltetrahydrofolate. 2018. https://doi.org/10.2210/pdb6dav/pdb.

(40) Ye, B.; Arnaiz, D. O.; Chou, Y.-L.; Griedel, B. D.; Karanjawala, R.; Lee, W.; Morrissey, M. M.; Sacchi, K. L.; Sakata, S. T.; Shaw, K. J.; Wu S. C.; Zhao Z.; Adler M.; Cheeseman S.; Dole W. P.; Ewing J.; Fitch R.; Lentz D.; Liang A.; Light D.; Morser J.; Post J.; Rumennik G.; Subramanyam B.; Sullivan M. E.; Vergona R.; Walters J.; Wang Y.-X.; White K. A.; Whitlow M.; Kochanny M. J. Thiophene-Anthranilamides as Highly Potent and Orally Available Factor Xa Inhibitors. *J. Med. Chem.* **2007**, *50*, 2967–2980.