The Journal of Chemical Physics

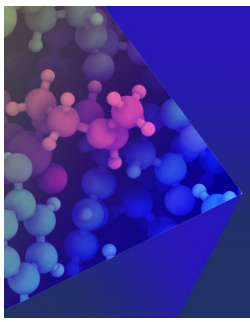
RESEARCH ARTICLE | JULY 03 2024

Force Field X: A computational microscope to study genetic variation and organic crystals using theory and experiment

Special Collection: Modular and Interoperable Software for Chemical Physics

Rose A. Gogal ; Aaron J. Nessler ; Andrew C. Thiel ; Hernan V. Bernabe; Rae A. Corrigan Grove ; Leah M. Cousineau; Jacob M. Litman ; Jacob M. Miller ; Guowei Qi ; Matthew J. Speranza ; Mallory R. Tollefson ; Timothy D. Fenn ; Jacob J. Michaelson ; Okimasa Okada ; Jean-Philip Piquemal ; Jay W. Ponder ; Jana Shen ; Richard J. H. Smith ; Wei Yang ; Pengyu Ren ; Michael J. Schnieders

J. Chem. Phys. 161, 012501 (2024)
https://doi.org/10.1063/5.0214652



The Journal of Chemical Physics SOFTWARE pubs.aip.org/aip/jcp

Force Field X: A computational microscope to study genetic variation and organic crystals using theory and experiment

Cite as: J. Chem. Phys. 161, 012501 (2024); doi: 10.1063/5.0214652
Submitted: 18 April 2024 • Accepted: 17 June 2024 •
Published Online: 3 July 2024

Rose A. Gogal,[1] Aaron J. Nessler,[1] Andrew C. Thiel,[1] Hernan V. Bernabe,[1] Rae A. Corrigan Grove,[2] Leah M. Cousineau,[3] Jacob M. Litman,[3] Jacob M. Miller,[1] Guowei Qi,[3] Matthew J. Speranza,[1] Mallory R. Tollefson,[1] Timothy D. Fenn,[4] Jacob J. Michaelson,[5] Okimasa Okada,[6] Jean-Philip Piquemal,[7] Jay W. Ponder,[8] Jana Shen,[9] Richard J. H. Smith,[10] Wei Yang,[11,12] Pengyu Ren,[13] and Michael J. Schnieders[1,3,a)]

AFFILIATIONS

[1]Roy J. Carver Department of Biomedical Engineering, University of Iowa, Iowa City, Iowa 52242, USA
[2]Theoretical Division, Los Alamos National Laboratory, Los Alamos, New Mexico 87545, USA
[3]Department of Biochemistry and Molecular Biology, University of Iowa, Iowa City, Iowa 52242, USA
[4]Analytical Development, LEXEO Therapeutics, New York, New York 10010, USA
[5]Department of Psychiatry, University of Iowa Hospitals and Clinics, Iowa City, Iowa 52242, USA
[6]Sohyaku Innovative Research Division, Mitsubishi Tanabe Pharma Corporation, 1000 Kamoshida-cho, Aoba-ku, Yokohama, Kanagawa 227-0033, Japan
[7]Department of Chemistry, Sorbonne Université, F-75005 Paris, France
[8]Department of Chemistry, Washington University in St. Louis, St. Louis, Missouri 63130, USA
[9]Department of Pharmaceutical Sciences, University of Maryland School of Pharmacy, Baltimore, Maryland 21201, USA
[10]Molecular Otolaryngology and Renal Research Laboratories, Department of Otolaryngology, University of Iowa Hospitals and Clinics, Iowa City, Iowa 52242, USA
[11]Department of Chemistry and Biochemistry, Florida State University, Tallahassee, Florida 32309, USA
[12]Institute of Molecular Biophysics, Florida State University, Tallahassee, Florida 32309, USA
[13]Department of Biomedical Engineering, University of Texas, Austin, Texas 78712, USA

Note: This paper is part of the JCP Special Topic on Modular and Interoperable Software for Chemical Physics.
[a)]**Author to whom correspondence should be addressed:** michael-schnieders@uiowa.edu

ABSTRACT

Force Field X (FFX) is an open-source software package for atomic resolution modeling of genetic variants and organic crystals that leverages advanced potential energy functions and experimental data. FFX currently consists of nine modular packages with novel algorithms that include global optimization via a many-body expansion, acid–base chemistry using polarizable constant-pH molecular dynamics, estimation of free energy differences, generalized Kirkwood implicit solvent models, and many more. Applications of FFX focus on the use and development of a crystal structure prediction pipeline, biomolecular structure refinement against experimental datasets, and estimation of the thermodynamic effects of genetic variants on both proteins and nucleic acids. The use of Parallel Java and OpenMM combines to offer shared memory, message passing, and graphics processing unit parallelization for high performance simulations. Overall, the FFX platform serves as a computational microscope to study systems ranging from organic crystals to solvated biomolecular systems.

INTRODUCTION

Force Field X (FFX) is an open-source software platform for atomic resolution modeling of organic materials and biomolecules that leverages molecular mechanics (MM) force fields and a family of novel electrostatics,[1–3] optimization,[4,5] thermodynamics,[6–8] and experimental refinement algorithms.[9–11] This article introduces the unique computational models and algorithms available in FFX, with emphasis placed on applications such as crystal structure prediction, understanding the mechanism of disease-causing protein missense variants, and the refinement of biomolecules against experimental datasets. FFX was envisioned as a platform to develop experimental refinement algorithms using advanced potential energy functions based on permanent atomic multipoles and induced dipoles, including the Atomic Multipole Optimized Energetics for Biomolecular Applications (AMOEBA) force field.[12–16] FFX's design has been influenced by lessons learned from packages such as Tinker,[17–19] OpenMM,[20] and Crystallography and NMR System (CNS),[21,22] while prioritizing the use of virtual machine (VM) technology and polyglot programming. FFX is distributed under the GPL v.3 license with the classpath exception, which is the same license used by OpenJDK (although v.2 in this case). The long-term goal for the FFX platform is to support the study of all major facets of organic materials, biological molecules, and their assembly into large complexes using the principles of chemical physics.

Polyglot architecture

FFX achieves portability and support for polyglot programming (i.e., the use of several programming languages) through VM technologies, including the Java VM (JVM),[23–25] GraalVM,[26,27] and TornadoVM[28,29] as shown in Fig. 1. The JVM, as part of the Java Development Kit (JDK), is one of the oldest VM platforms. Over the last decade, the GraalVM project has augmented the JVM with an alternative just-in-time compiler (Graal), the ability to create ahead-of-time native images, and support for polyglot programming to allow languages such as Python[30,31] to execute on the JVM and interoperate with canonical JVM languages (e.g., Java, Kotlin, Scala, and Groovy).

FIG. 1. An overview of Force Field X. The second row shows languages that are compatible with FFX. The third row represents the platforms that execute FFX on CPU cores on the left and coprocessors on the right. The final row indicates the CPU and GPU hardware that FFX can perform calculations on.

Novel algorithms

The platform includes many novel algorithms that will be discussed in more detail throughout this article. For example, FFX debuted the first implementation of x-ray crystallography refinement using the polarizable atomic multipole AMOEBA force field, the first constant-pH molecular dynamics (MD) support for a polarizable force field,[32] and the generalized Kirkwood (GK) implicit solvent models.[3,33–35] FFX also implements efficient inclusion of space group symmetry into long-range electrostatics via particle-mesh Ewald (PME) summation[1,35–37] and orthogonal space tempering (OST) for the calculation of free energy differences.[6,8,38,39] It can also perform global optimization using dead-end[40] (and Goldstein[41]) elimination for target functions that include higher-order many-body terms[11,43] and dual topology methods to compute free energy differences between potential energy models.[7,42]

Commands and parallelization

FFX offers more than 50 commands that implement a variety of methods to investigate organic and biomolecular systems, which are organized into nine Java packages. These packages will be discussed in more detail later in this article. FFX leverages GraalVM technologies for cross-platform polyglot programming in Java, Groovy, Python, and Kotlin. FFX utilizes the Parallel Java (PJ) package to facilitate parallelization across the central processing unit (CPU) cores/threads of a single process [i.e., shared memory (SM)] and among multiple processes using its message passing interface (MPI). Graphics processing unit (GPU) acceleration is currently achieved using OpenMM based on a Java class hierarchy that mirrors the OpenMM C++ application programming interface (API). The source code is freely available for download from the FFX website (http://ffx.biochem.uiowa.edu) or from GitHub (https://github.com/SchniedersLab/forcefieldx) and currently depends on JDK version 21 or greater. The software is compiled within a terminal window using the Apache Maven tool and is executed in either "headless" command line mode using the "ffxc" command or with a simple interactive graphical user interface using the "ffx" command.

Applications

Throughout this article, we will discuss applications of the novel algorithms available in FFX. One application is the refinement of atomic resolution models against x-ray and/or neutron diffraction data using advanced force fields.[10,11] FFX is also able to investigate the impact of missense variants on protein structure, dynamics, and thermodynamics.[43–53] In addition, FFX premiers a novel pipeline for crystal structure prediction.[54,55] The wide-ranging capabilities of FFX make it particularly useful for the computational study of biochemical mechanisms.

FEATURES AND ORGANIZATION

File types, coordinate representations, and conversions

FFX utilizes a variety of file types to describe molecular systems. Files typically consist of a base name that is shared among all files for a system and a unique suffix to denote the information contained therein (e.g., molecule.xyz, molecule.mtz, and molecule.properties). FFX uses a suffix versioning system where subsequent files of the same type are saved with an appended integer (e.g., generated coordinates from molecule.xyz would produce a file named molecule.xyz_2). Commonly used file types are found with a brief description in Table I.

FFX implements a variety of commands to facilitate alterations and conversions between the file types listed in Table I. General file commands are listed in Table II along with a brief description. The *Cart2Frac* and *Frac2Cart* commands convert atomic coordinates between Cartesian and fractional coordinate systems, where the latter defines each atomic position as a fractional (unitless) distance along each axis of a unit cell. *SaveAsP1* expands a periodic system by applying space group symmetry operators to the asymmetric unit to generate a P1 unit cell (or a larger replicated unit cell). *SaveAsXYZ* converts a system into an XYZ coordinate file. *SaveAsQE* generates a default script that can be used with Quantum ESPRESSO to perform a plain-wave self-consistent field (PWscf) calculation. *ImportCIF* converts an organic crystal from the Crystallographic information file (CIF) format (e.g., from the Cambridge Structural Database[56]) into XYZ format based on the constituent molecule(s) having already been parameterized. While reading systems, *ImportCIF* actively enforces space group restrictions with regard to lattice parameters and updates the space group in some cases (e.g., a rhombohedral space group with hexagonal lattice parameters is converted to the appropriate hexagonal space group). *ImportCIF* adds hydrogen atoms to a system if none are present in the original CIF file. It can also convert an XYZ system to a basic CIF format containing the coordinate and crystal information. *MoveIntoUnitCell* moves the molecules of a system to ensure each center of mass is located within the unit cell. Finally, the command *xray.MTZInfo* displays human readable information for a binary MTZ file, while *xray.CIFtoMTZ* generates an MTZ file for a corresponding CIF file.

Software organization

FFX is composed of nine Java packages organized by functional themes and capabilities. The nine packages are Parallel Java (PJ), Utilities, Numerics, Crystal, OpenMM, Potential, Algorithms, Refinement, and User Interfaces (UI). The organization of packages and their dependencies within FFX are shown in Fig. 2.

Parallel Java[57,58] provides APIs for both shared memory (SM) parallelization using threads and a message-passing interface (MPI)

TABLE I. File types read and written by FFX, including the origin of each format.

Atomic coordinates and variables		
ARC	Structural archive	Tinker
CIF	Crystallographic information file	IUCr
DYN	Molecular dynamics restart information	Tinker
ESV	Extended system variables	FFX
INT	Internal coordinates	Tinker
PDB	Protein databank structure	PDB
XPH	XYZ file with pH information	FFX
XYZ	Coordinates, atom types, and connectivity	Tinker
Control properties and force field parameters		
DST	Distance matrix from superposing structures	FFX
Properties/key	Control file with Java properties/tinker keywords	FFX/Tinker
PRM	Force field parameter file	Tinker
Structure factors and real space maps		
CCP4/MAP	Real space density map	CCP4
CIF	Structure factors	IUCr
CNS/HKL	Structure factors	CNS/Xplor
MTZ	Binary format for structure factors	CCP4
XPLOR	Real space density map	CNS/Xplor
Thermodynamics		
BAR	Window energy values for FEP/BAR	Tinker
HIS	Orthogonal space histogram	FFX
LAM	Lambda restart information	FFX
MBAR	Window energy values for MBAR	FFX

TABLE II. Structure manipulation commands available in FFX.

Command	Description
Cart2Frac	Convert from Cartesian to fractional coordinates
Frac2Cart	Convert from fractional to Cartesian coordinates
SaveAsP1	Expands a crystal to P1 or a replicated unit cell
SaveAsXYZ	Save the system as an XYZ file
SaveAsQE	Create a quantum ESPRESSO input script
ImportCIF	Import a system from a CIF file
MoveIntoUnitCell	Move all molecules into the unit cell
xray.MTZInfo	Log information for an MTZ file
xray.CIFtoMTZ	Convert diffraction data from CIF format to MTZ

for parallelization across JVM processes that are executing on the same node and/or between nodes. The former SM parallelization leverages concepts analogous to those defined by OpenMP, including support for parallelization of "for" loops, atomic operations, and scheduling of tasks. The PJ MPI approach defines its own scheduling and communication (e.g., "mpirun" is unnecessary) without any dependencies beyond the Java Runtime Environment (JRE). The Parallel Java package is used to accelerate operations in most other packages [e.g., 3D fast Fourier transforms (FFTs), force field energy evaluations, and replica-based sampling strategies]. Therefore, all packages have PJ as a dependency. Our fork of the original PJ code by Alan Kaminsky contains several changes geared toward freeing SM hardware threads that are no longer in use and enhanced logging of MPI communications.[57,58]

The Utilities package provides simple functionality for file handling (e.g., file copying and file naming conventions) and the implementation of system properties. Properties in FFX are built on the JVM standards for system properties. These properties (key-value pairs) are used to specify calculation details that vary between simulation goals. All packages except for PJ and OpenMM depend on the Utilities package.

The Numerics package implements common mathematical methods and numerical recipes required for calculations available within FFX. The Numerics package computes real and complex 1D[59,60] and 3D fast Fourier transforms (FFTs) and offers both float (single precision) and double (double precision) vector math libraries and a range of special functions, such as Erf/Erfc

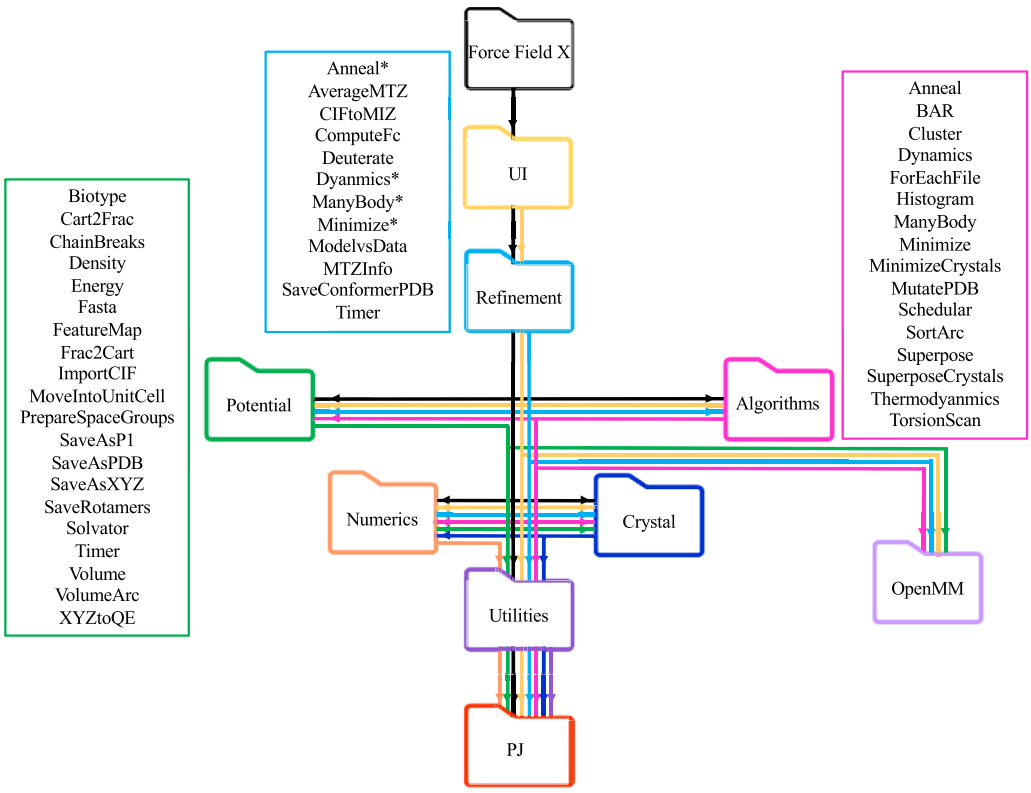


FIG. 2. A flowchart representation of the package dependencies in FFX. Each package is indicated by a separate color, and the lines between packages indicate a dependency relationship. For example, most colored lines connect to the PJ package as all packages depend on PJ except for OpenMM. Commands available in the Potential, Algorithms, and Refinement packages are displayed in the corresponding color-coded box. For Refinement commands, those with an asterisk indicate that both x-ray and Real Space versions of the command are available, while those without an asterisk currently have only an x-ray version.

and modified Bessel functions. Numerics offers a novel family of multipole tensor recursion[61] algorithms for Coulomb, Ewald, Thole, and GK interactions using both Cartesian and quasi-internal[62] coordinate frames. Finally, the Numerics package implements atomic operations on arrays, a limited-memory Broyden–Fletcher–Goldfarb–Shanno (BFGS)[63–66] optimizer, and support for uniform b-splines.[67] The Crystal, Potential, Algorithms, Refinement, and UI packages each depends on the Numerics package.

The Crystal package gives FFX the ability to perform operations on all 230 crystallographic space groups during the evaluation of a force field potential energy and during crystallographic refinement against x-ray and/or neutron diffraction data. It provides methods for application of the minimum image convention, symmetry operators, conversion between Cartesian and fractional coordinates, and the storage of reflection lists from diffraction experiments.[68] The Potential, Algorithms, Experiment, and UI packages each depends on the Crystal package.

The Potential package provides support for evaluating the potential energy of atomic resolution molecular systems using fixed partial atomic charge force fields, polarizable atomic multipole force fields, and preliminary support for using neural network potentials that are based on PyTorch.[69] When a molecular system is loaded from a file, an instance of the MolecularAssembly class is instantiated together with creation of an associated ForceFieldEnergy. The latter provides methods to compute the potential energy of the system and optionally its gradient for use with simulation methods defined in the Algorithms package described below. The Potential package supports calculation of long-range electrostatics using particle-mesh Ewald[35–37] (PME) summation with novel support for space group symmetry[1] and offers continuum treatment of solvent via our GK model.[3,33–35] The FFX commands defined within the Potential package are listed in Fig. 1. Some notable commands include *Energy* to calculate the potential energy of a system, *Solvator* to solvate a system into a water box with or without free salt, and the *SaveAs* family of commands to convert systems between file types. The Algorithms, Refinement, and UI packages depend on the Potential package.

The Algorithms package contains optimization and sampling methods that operate on the potential energy functions defined in the Potential package. The Algorithms package commands are listed in Fig. 2. A family of local optimization commands (*Minimize*, *CrystalMinimize*, and *PhMinimize*) leverage the L-BFGS method defined in the Numerics package. The Algorithms package also offers global optimization methods based on simulated annealing (*Anneal*) and via our many-body versions[4,11] of dead-end elimination[40] and Goldstein elimination[73] (*ManyBody*). The *Dynamics* command executes molecular dynamics via integrators that include velocity Verlet, Beeman,[70] stochastic dynamics,[71,72] and the reversible reference system propagation algorithm (r-RESPA).[73,74] In the absence of stochastic dynamics, temperature control is available via thermostats by Berendsen *et al.*[75] and Bussi *et al.*[76] Constant pressure is achieved using a novel Monte Carlo barostat[77] that respects space group constraints. The *Thermodynamics* command computes free energy differences via an alchemical path defined by a state variable (λ) either by sampling an array of windows at fixed λ values [followed by application of the Bennett Acceptance Ratio (BAR) method[78]] or using a unique implementation of the orthogonal space tempering[38,79] method that supports both polarizable force fields and space group symmetry.[6–8,80] The Refinement and User Interface packages depend on the Algorithms package.

The Refinement package implements target functions that combine a potential energy function defined in the Potential package (e.g., a fixed charge or polarizable force field) with a function that compares a molecular model (e.g., its atomic coordinates, b-factors, and/or occupancies) to either diffraction data in reciprocal space or a scalar field (e.g., electron density) in real space.[10] This includes a novel bulk scattering model that is differentiable with respect to atomic coordinates[9] and the unique ability to evaluate long-range electrostatics using the rigorous PME method.[1] Once the overall target function is defined, then most of the optimization and sampling methods of the Algorithms package can be employed for model refinement (e.g., local minimization, molecular dynamics, many-body side-chain optimization, and simulated annealing). Refinement against x-ray diffraction data, neutron diffraction data, or joint x-ray/neutron diffraction datasets is supported. Global optimization of side-chain conformations against either a reciprocal space or real space target function is supported.[11] The available commands in the Refinement package are listed in Fig. 2. Those with an asterisk (*) have both reciprocal and real space versions. The User Interface package depends on the Refinement package.

The User Interface package implements both command line and graphical user interfaces. This package depends on all other packages. We will elaborate on FFX functionalities in more explicit detail later in this article.

EXTERNAL LIBRARIES

FFX leverages a variety of external libraries to perform functions such as CIF file parsing, parallelization, and bioinformatics methods. These packages include CIF tools,[81] BioJava,[82] the Chemistry Development Kit (CDK),[83] TornadoVM,[84] Parallel Java,[57] the picocli (a command line interface package),[85] the Groovy language, and the GraalVM Python implementation. CIF tools give FFX the ability to read and write CIF files from the Cambridge Structural Database.[56] The BioJava library supports local installations of the Protein Data Bank (PDB), can load protein and nucleic acid sequences stored in FASTA format, and offers algorithms for structural alignment. The Chemistry Development Kit (CDK) is a collection of modular libraries for processing chemical information, including support for parsing files in the SMILES, SDF, and Mol2 format. CDK also allows substructure and SMARTS pattern matching and fingerprint methods for similarity searching. TornadoVM extends the JRE by offering programming constructs to facilitate translating a subset of Java code into PTX (CUDA), OpenCL, or SPIRV (Intel Level Zero) backends. As mentioned previously, the Parallel Java library implements both "OpenMP Style" shared memory coding constructs and a platform independent implementation of the canonical message passing interface operations for parallelization across processes. Picocli is an annotation-driven library for creating command line applications along with documentation in HyperText Markup Language, portable document format (PDF), or Unix man page formats. The Groovy scripting language is used to quickly prototype new ideas and create FFX

commands. Finally, with the emergence of a Python 3.10 implementation that runs on the JVM from the GraalVM team, FFX now also fully supports execution of Python scripts in a cross-platform manner.

POTENTIAL ENERGY FUNCTIONS

FFX supports a variety of potential energy functions that include both fixed partial charge and polarizable atomic multipole force fields. There is support for implicit solvents consisting of cavitation, dispersion, and generalized Kirkwood contributions, energy terms for refinement against x-ray and/or neutron diffraction data, and energy terms for refinement against real space maps (e.g., from either CryoEM or diffraction experiments). FFX also offers unique support for a family of dual-topology potential energy functions that facilitate computing free energy differences due to chemical modifications in the AMOEBA force field (e.g., in the context of relative binding affinity or relative hydration free energy), between two crystal polymorphs, or between alternative force field models. Force field parameter files adopt Tinker conventions when possible, however, supports for experimental refinement and dual-topology algorithms are unique to FFX.

Force field models

The overall force field functional form combines bonded and non-bonded interactions,

$$U_{\text{ForceField}} = U_{\text{Bonded}} + U_{\text{Non-Bonded}}. \tag{1}$$

The bonded energy may include bond stretching, Urey–Bradley stretching, angle bending, bond-angle cross-term, out-of-plane bending, improper torsion, torsional angle, stretch–torsion cross-term, angle–torsion cross-term, and/or a torsion–torsion cross-term,

$$\begin{aligned} U_{\text{Bonded}} = {} & U_{\text{Bonds}} + U_{\text{Urey-Bradley}} + U_{\text{Angles}} + U_{\text{Bond-Angles}} \\ & + U_{\text{Out-of-PlaneBends}} + U_{\text{Torsions}} + U_{\text{Improper-Torsions}} \\ & + U_{\text{Stretch-Torsions}} + U_{\text{Angle-Torsions}} + U_{\text{Torsion-Torsions}}. \end{aligned} \tag{2}$$

However, no currently supported force field includes all ten bonded terms. For example, the Optimized Potentials for Liquid Simulations - All Atom (OPLS-AA)[86] and OPLS-AA/L[87] force fields include the following four terms: bond stretching, angle bending, improper torsion, and torsional angle. On the other hand, a simulation that utilizes the AMOEBA nucleic acid force field[16] in explicit water[12] includes all terms except improper torsions. The non-bonded terms may include van der Waals interactions (using either 6-12 Lennard-Jones or Buffered-14-7 functional forms), permanent electrostatics based on fixed atomic charges, or multipoles (truncated at quadrupole order) and polarization energy via induced dipoles,

$$U_{\text{Non-Bonded}} = U_{\text{vdW}} + U_{\text{Elec}}^{\text{Permanent}} + U_{\text{Elec}}^{\text{Induced}}. \tag{3}$$

Currently supported force fields, in addition to those mentioned above, include Amber94,[88] Amber96,[89] Amber99,[90] Amber99sb,[91] Charmm22,[92,93] Charmm22 with CMAP correction,[94] and the AMOEBA model for small organic molecules[14] and proteins.[15]

IMPLICIT SOLVENT

FFX implements implicit solvents for applications where the use of explicit water molecules is cumbersome, including the repacking of protein side chains and the refinement of biomolecular models against experimental data. The implicit solvent is generally formulated as a sum of three free energy differences defined by a thermodynamic cycle: cavitation, dispersion, and electrostatics contributions. The cycle describes the transfer of the biomolecular system between vacuum into solvent phases. The combination of cavitation and dispersion free energy differences is usually referred to as the non-polar contribution,[56–59] while the latter electrostatic contribution is based on an analytic approximation to the Poisson equation as described below.

The Generalized Born (GB) model approximates the polar, electrostatic term for fixed partial charge force fields.[95,96] GB is formulated as a sum over pairwise and self-interactions to approximate the electrostatic solvation free energy difference (ΔG_{GB}),

$$\Delta G_{GB} = \frac{1}{2}\left(\frac{1}{\varepsilon_s} - \frac{1}{\varepsilon_h}\right)\sum_{i,j}\frac{q_i q_j}{f_{ij}}, \tag{4}$$

where ε_s is the permittivity of the solvent, ε_h is the permittivity of the homogenous reference state, and q_i and q_j are partial charges. The original form of the generalizing function f_{ij} is defined in the following form:

$$f_{ij} = \sqrt{r_{ij}^2 + a_i a_j \exp\left(-r_{ij}^2/c a_i a_j\right)}, \tag{5}$$

where r_{ij} is the separation distance between atoms, a_i and a_j are effective Born radii,[97] and c is a constant to control the transition from the Born regime to Coulomb's law regime. To support the polarizable atomic multipole AMOEBA force field, GB concepts were extended to define the GK model that handles multipole moments of arbitrary degree.[33] The GK monopole term ($\Delta G_{GK}^{(q,q)}$) is equivalent to GB [Eq. (4)]. To calculate the interactions between all permanent dipole moments, the GK dipole term is defined as

$$\Delta G_{GK}^{(\mu,\mu)} = \frac{1}{2}\left[\frac{1}{\varepsilon_h}\frac{2(\varepsilon_h - \varepsilon_s)}{2\varepsilon_s + \varepsilon_h}\right]\sum_{i,j}\mu_{i,\alpha}\mu_{j,\beta}\left[\frac{3r_\alpha r_\beta g_{ij}}{f_{ij}^5} + \frac{\delta_{\alpha\beta}}{f_{ij}^3}\right], \tag{6}$$

where $\boldsymbol{\mu_i}$ and $\boldsymbol{\mu_j}$ are permanent atomic dipole moments, the subscripts α and β denote the use of the Einstein summation convention, $\delta_{\alpha\beta}$ is the Kronecker delta, the separation along the α dimension is given by $r_\alpha = r_{j,\alpha} - r_{i,\alpha}$, and the chain rule term (g_{ij}) is given by

$$g_{ij} = \frac{\exp\left(-r_{ij}^2/c a_i a_j\right)}{c} - 1. \tag{7}$$

Both the GB and GK equations use effective radii, rather than the intrinsic radius of each atom, which represent the degree of burial within the solute. An atom that is deeply buried within the center of a protein has a larger effective radius than a surface exposed atom of the same type. An isolated atom's (i.e., an ion) effective radius will approach its intrinsic atomic radius.

FFX calculates effective radii by combining the analytic Hawkins, Cramer, and Truhlar (HCT) pairwise descreening approximation[98] with the solvent field approximation (SFA) proposed by Grycuk.[99] Contributions to effective radii due to interstitial spaces (spaces within a system where a water molecule cannot fit) are also included due to their importance when modeling proteins. The interstitial space corrections include a pairwise "neck" between nearby atoms[100] and a hyperbolic tangent (tanh) function[101] to help smoothly scale up the effective radius of an atom as it becomes more deeply buried. Currently, the FFX implementation of GK implicit solvent has been validated for protein simulations with work ongoing to support nucleic acids.[3]

DUAL-TOPOLOGY FRAMEWORK

Interpolation between force fields

FFX implements two indirect free energy (IFE) or bookending methods.[7,8] These methods seek to correct thermodynamic quantities obtained with a relatively low-resolution potential energy function (e.g., a molecular mechanics force field) to be consistent with a higher-resolution potential energy function (e.g., a polarizable molecular mechanics force field, neural network, or Quantum Mechanics/Molecular Mechanics potential).[102] The first approach is called the dual force field (DFF) method.[7] DFF creates an alchemical path between a relatively expensive potential at one end (i.e., the polarizable AMOEBA model U_{AMOEBA}) and a relatively inexpensive potential at the other end (i.e., a fixed partial charge force field U_{FC}). The potential energy along the path is then given by

$$U_{DFF} = \lambda * U_{\mathrm{AMOEBA}}(\mathbf{x}) - (1-\lambda)U_{\mathrm{FC}}(\mathbf{x}), \tag{8}$$

where λ is a state variable that ranges between 0 and 1 to parameterize the transition between force field resolutions and $\mathbf{x}$ are the atomic coordinates. This approach has been successfully used to compute the free energy difference for the sublimation of small organic molecules[7] and, more recently, in the context of computing relative anhydrous–hydrate stability.[42]

Despite these successes, a limitation of the DFF approach is that both end states must have the same degrees of freedom and constraints (e.g., bonds and angles must be either flexible or rigid under both potentials). DFF also has difficulty converging the free energy difference for large systems. This is due to the size extensive nature of the calculation in tandem with relatively large contributions from slight differences in the bonded terms' equilibrium values. The latter limitation is partially addressed by the second IFE method in FFX called Simultaneous Bookending (SB).[8] SB couples two DFF simulations running in opposite directions—the first coarsens the resolution of the system and the second refines the resolution back to the more accurate potential energy function—to compute the free energy correction for the entire transformation in one step rather than using two IFE simulations (i.e., one at each end of the thermodynamic path). SB does not entirely solve convergence issues, but it does allow for an approximation based on distance to the site of the original alchemical transformation. Atoms distal to the site of an alchemical transformation can be pinned or constrained to share identical coordinates. This constraint dramatically improves sample convergence because many energy terms do not contribute to the partial derivative of the total potential energy with respect to λ and thus do not contribute to the free energy difference. The SB potential energy along the alchemical path is given by a weighted sum of four force field energy contributions (or two DFF potentials),

$$\begin{aligned} U_{SB}(\mathbf{x}_C, \mathbf{x}_R, \mathbf{x}_S) = {} & \lambda * \left[U_{\mathrm{A,FC}}(\mathbf{x}_C, \mathbf{x}_S) + U_{\mathrm{B,AMOEBA}}(\mathbf{x}_R, \mathbf{x}_S)\right] \\ & + (1-\lambda) * \left[U_{\mathrm{B,FC}}(\mathbf{x}_R, \mathbf{x}_S) + U_{\mathrm{A,AMOEBA}}(\mathbf{x}_C, \mathbf{x}_S)\right], \end{aligned} \tag{9}$$

where $\mathbf{x}_S$ are the shared (or pinned) coordinates common to both sides of the SB simulation (e.g., atoms distal from the site of alchemical transformation), while $\mathbf{x}_C$ and $\mathbf{x}_R$ are independent degrees of freedom near the alchemical transformation for the "coarsen" and "refine" steps, respectively.

Overly aggressive coordinate constraints cause the calculated free energy correction to diverge from the actual free energy correction due to the approximation that both systems share an identical phase space. On the other hand, overly conservative constraints render the correction increasingly expensive to converge. The SB approach was used to correct the binding free energy differences for a set of divalent cation binding proteins to within statistical uncertainty of the true calculated AMOEBA values.[8] Compared to prior IFE methods, the SB approach allowed an order of magnitude more atoms to be converted between resolutions. Future IFE efforts would benefit from force field resolutions with identical bonded terms (e.g., between the fixed partial charge and polarizable atomic multipole resolutions) such that only non-bonded terms contribute to the corrections.

Interpolation between polymorphs

A recent addition to the dual topology framework, described here for the first time, employs a symmetry operator to interpolate between polymorphs and thereby estimate their free energy difference. The dual polymorph potential energy is defined as

$$U_{DP}(\mathbf{x}_A, \lambda) = (1-\lambda) * U_A(\mathbf{x}_A, \lambda) + \lambda * U_B(S_{A\to B}(\mathbf{x}_A), \lambda), \tag{10}$$

where the coordinates of the first polymorph ($\mathbf{x_A}$) are used to generate those of the second using a symmetry operator ($S_{A\to B}$) defined using the Progressive Alignment of Crystals (PAC) algorithm presented in the section titled Structure Manipulation. At $\lambda = 0$, the molecule(s) within the asymmetric unit experience the crystalline environment defined by the space group of polymorph A. At $\lambda = 1$, they experience the crystalline environment defined by the space group of polymorph B. At intermediate values of the state parameter (λ), the symmetry mate interactions are smoothly transformed as depicted in Fig. 3. At each molecular dynamics step, the inverted symmetry operator is applied to rotate forces produced by the second polymorph (B) back into the frame of the first polymorph (A). Overall, this relative free energy difference approach for polymorphs offers performance advantages relative to taking the difference between two absolute free energy differences (i.e., analogous to the advantages of relative binding free energy differences).

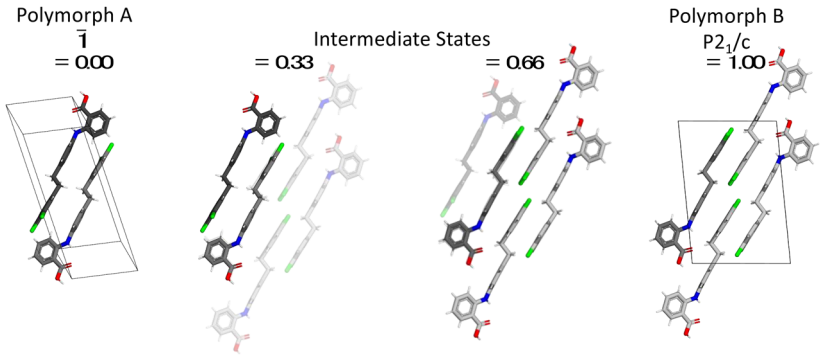


FIG. 3. Depiction of an alchemical path connecting two crystal polymorphs defined by different space groups via a dual topology framework [Eq. (10)]. Symmetry mates produced via the $P\bar{1}$ symmetry operators of polymorph A are shown with carbon atoms colored black, and those produced by the $P2_1/c$ symmetry operators of polymorph B are light gray. The atoms of the asymmetric unit of polymorph A are mapped via a custom symmetry operator into the asymmetric unit of polymorph B (asymmetric unit carbon atoms are colored gray for each state).

STRUCTURE MANIPULATION

FFX contains methods to alter systems as needed. *MutatePDB* changes the identity of a chosen amino acid to an alternative identity. *Solvator* creates a periodic box of water around the input system and optionally adds explicit counterions. The *Superpose* command calculates the coordinate root mean square deviation (RMSD) between two systems via quaternion superposition. A unique *SuperposeCrystals* command quantifies the packing similarity between two crystals. This is performed using the Progressive Alignment of Crystals (PAC) algorithm, which calculates an atomistic RMSD for the aligned molecular subclusters of each crystal.[54] The input options the user can select include the number of asymmetric units in each subcluster, the selection criteria for molecules in the subcluster, and atom(s) to be excluded from the comparison. *SuperposeCrystals* can save crystal systems that are within a user specified RMSD from a desired crystal or to write out a matrix of RMSD values for clustering and/or filtering out similar crystals within an ensemble. Furthermore, PAC can accumulate the rotations and translations into an overall symmetry operator to map moleculars between crystal polymorphs as discussed in the section titled Interpolation Between Polymorphs.

LOCAL OPTIMIZATION

FFX currently has two methods for local optimization: the steepest descent method and the Limited-memory Broyden–Fletcher–Goldfarb–Shanno (L-BFGS) method. The latter is a quasi-Newton approach that minimizes a nonlinear function by approximating the inverse Hessian matrix.[103,104] Both methods are available via the *Minimize* command to optimize atomic coordinates for a given potential energy function. The *MinimizeCrystals* command additionally supports optimization of lattice parameters, while the *MinimizePh* command supports local optimization of titration and tautomer states. Finally, the *xray.Minimize* and *realSpace.Minimize* commands facilitate optimization of coordinates, b-factors, and/or occupancies against experimental diffraction or real space data, respectively.

GLOBAL OPTIMIZATION

Simulated annealing

FFX implements simulated annealing functionality via the *Anneal* command.[50,105] The command allows for selection of the heating and cooling schedule, the molecular dynamics protocol and simulation length at each temperature, and selection of the atomic degrees of freedom that should remain fixed. Although the simulated annealing approach can locate the global minimum of a target function, its success depends on the simulations at each temperature being of sufficient length. Versions of the method are available for refinement against both experimental diffraction data (*xray.Anneal*) and real space maps (*realSpace.Anneal*).[106]

Methods based on a many-body expansion

Under a many-body potential, such as a polarizable force field,[15] implicit solvent,[3] and/or x-ray diffraction target,[11] the total energy of the system $E(\mathrm{r})$ can be defined to arbitrary precision using a many-body expansion,

$$E(\mathrm{r}) = E_{env} + \sum_i E_{self}(r_i) + \sum_i \sum_{j>i} E_2(r_i, r_j) + \sum_i \sum_{j>i} \sum_{k>j} E_3(r_i, r_j, r_k) + \cdots, \tag{11}$$

where E_{env} is the energy of the environment (e.g., a protein backbone and any residues that are not being optimized). $E_{self}(r_i)$ is the self-energy of residue i that includes its intra-molecular bonded energy terms and non-bonded interactions with the backbone. $E_2(r_i, r_j)$ is the two-body non-bonded interaction energy between residues i and j, and $E_3(r_i, r_j, r_k)$ is the three-body non-bonded interaction energy between residues i, j, and k. The self, two-body, and three-body energy terms from Eq. (11) are calculated as follows, where $\mathrm{E_{env/sc}}$ is the total energy of the environment and the side chain(s) of the selected residue(s):

$$E_{self} = E_{\mathrm{env/sc}}(r_i) - E_{env}, \tag{12}$$

$$E_2(r_i, r_j) = E_{\text{env/sc}}(r_i, r_j) - E_{self}(r_i) - E_{self}(r_j) - E_{env}, \quad (13)$$

$$E_3(r_i, r_j, r_k) = E_{\text{env/sc}}(r_i, r_j, r_k) - E_{self}(r_i) - E_{self}(r_j) - E_{self}(r_k) - E_2(r_i, r_j) - E_2(r_i, r_k) - E_2(r_j, r_k) - E_{env}. \quad (14)$$

In the past, global optimization algorithms over a discrete permutation space (i.e., side chain conformations and/or amino acid identity) using a many-body expansion were limited to truncation at pairwise interactions.[40,73] We demonstrated that the pairwise dead-end elimination and Goldstein elimination criteria can be modified to include three-body (or higher) energy terms.[11] However, computing the self, two-body, and three-body energy terms as a function of rotamer conformation is computationally expensive. To address this challenge, we demonstrated two complementary parallelization approaches in FFX, including (1) use of MPI parallelization to distribute terms among multiple processes and (2) use of the OpenMM API to perform energy evaluations on NVIDIA GPUs via CUDA kernels.[4]

In addition to extending the Goldstein elimination criteria and implementing parallelization strategies, four approximations were introduced to improve computational performance. In the context of rotamer optimization, the expansion can often be truncated at pairwise terms due to damping of three-body and higher-order terms by the generalized Kirkwood implicit solvent. However, previous work also demonstrated that inclusion of three-body terms is sometimes necessary (i.e., in the absence of implicit solvent or when using an x-ray diffraction target function).[11] The second approximation employs a distance cutoff to exclude interactions where the closest rotamers for a residue pair are more than 3 Å apart or for residue triples that are more than 2 Å apart. Pruning removed rotamers with self-energies 25 kcal/mol or more above the lowest self-energy of a residue, prior to calculation of two-body energies. The pruning criterion is based on the heuristic observation that rotamers with such an unfavorable self-energy (e.g., due to an atomic clash with backbone atoms) are not found in well-packed structures. The final approximation imposed a 3D grid over the protein, followed by optimization within each subdomain (cube) of the grid, rather than including all protein residues simultaneously. The repacking algorithm is a provable global optimizer within a single subdomain of the grid, but not for the whole protein because coordinated changes between subdomains are neglected.

PERIODIC SYSTEMS, SPACE GROUP SYMMETRY, AND NEIGHBOR LISTS

The conditional convergence of Coulomb's law under periodic boundary conditions can be treated by splitting the electrostatic potential into a strictly convergent short-ranged real space contribution and a smooth periodic contribution. The smooth periodic contribution is strictly convergent in Fourier space as described by Ewald.[107] For crystals of small organic molecules, both space group symmetry and handling of the minimum image convention require special consideration when building the neighbor list as shown in for Compound 23. In FFX, both issues are handled in a uniform fashion based on permuting space group symmetry operators with the translational operators needed to generate replicated copies of the unit cell (i.e., to generate an overall cell that is larger than twice the cutoff used during building of the neighbor list as shown in Fig. 4). In direct space, the non-bonded pairwise loops over neighbors (i.e., for van der Waals or electrostatics interactions) leverage an additional outer loop over the permuted symmetry operators. FFX then utilizes a customized version of the smooth particle mesh Ewald (PME) algorithm that handles both space group symmetry and small unit cells.[108] For NPT simulations, the number of replicates cells is adjusted dynamically to accommodate shrinking or growing unit cells.

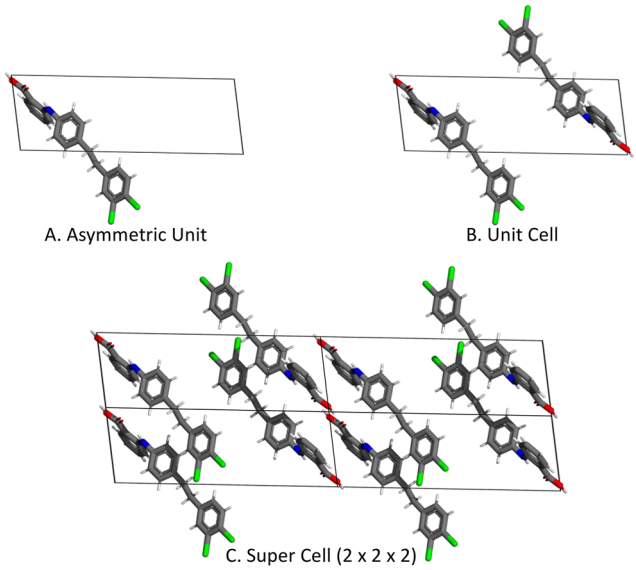


FIG. 4. FFX supports all 230 space groups, the use of unit cells whose interfacial radius is less than half of the nonbonded cutoff distance, and the combination of small unit cells with space group symmetry.

PARTICLE MESH EWALD SUMMATION

As a part of FFX's potential energy capabilities, a general implementation of smooth particle-mesh Ewald summation (PME) is included to avoid using electrostatic cutoffs and boost performance.[35,36] The FFX implementation of PME for multipoles builds on the work of Sagui *et al.*[37] by adding unified support for symmetry operators for all 230 crystallographic space groups and replicates operators for small unit cells (i.e., interactions with neighboring unit cells are required to satisfy the real space cut-off) and their combination.[1] Here, we will emphasize notable changes in the algorithm compared to that available in other simulation packages due to the inclusion of symmetry operators for both the direct and reciprocal space terms.

Real space energy

The real space permanent atomic multipole interaction energy is tuned by the Ewald parameter (β),

$$U_{\text{real}} = \frac{1}{2}\sum_{s_j=1}^{n_r,*}\sum_{i=1}^{n_a}\sum_{j=1}^{n_a} L_i(\mathbf{I})L_j(\mathbf{R}_{s_j})\frac{\operatorname{erfc}\left(\beta\left|\mathbf{r}_i - \left(\mathbf{R}_{s_j}\mathbf{r}_j + \mathbf{t}_{s_j}\right)\right|\right)}{\left|\mathbf{r}_i - \left(\mathbf{R}_{s_j}\mathbf{r}_j + \mathbf{t}_{s_j}\right)\right|}, \quad (15)$$

where the outer summation is over n_r space group symmetry and/or replicates operators and the inner summations are over n_a atoms in the asymmetric unit. The distance from an atom in the asymmetric unit at $\mathbf{r}_i$ to another atom at $\mathbf{R}_{s_j}\mathbf{r}_j + \mathbf{t}_{s_j}$ is a function of operator s_j with Cartesian rotation matrix $\mathbf{R}_{s_j}$ and translation vector $\mathbf{t}_{s_j}$.The asterisk on the outer summation indicates that $i = j$ interactions are neglected and masked interactions are respected for the identity symmetry operator ($s_j = 1$ by convention). Finally, the operators L_i and L_j are

$$L_i(\mathbf{R}) = q_i + (\mathbf{R}\mathbf{d}_i)_\alpha \nabla_{i,\alpha} + \left(\mathbf{R}\Theta_i\mathbf{R}^{\mathrm{t}}\right)_{\alpha\beta}\nabla_{i,\alpha}\nabla_{i,\beta}\frac{1}{3} \tag{16}$$

and

$$L_j(\mathbf{R}) = q_j - (\mathbf{R}\mathbf{d}_j)_\alpha \nabla_{i,\alpha} + \left(\mathbf{R}\Theta_j\mathbf{R}^{\mathrm{t}}\right)_{\alpha\beta}\nabla_{i,\alpha}\nabla_{i,\beta}\frac{1}{3}, \tag{17}$$

where charge (q), dipole ($\mathbf{d}$), and traceless quadrupole ($\mathbf{\Theta}$) moments are operated on by a Cartesian rotation matrix. $\nabla_{i,\alpha}$ is one component of the del operator acting at $\mathbf{r}_i$, $\alpha \in \{x, y, z\}$, and $\{\alpha, \beta\}$ represent the use of the Einstein summation convention for summing over tensor elements. A replicate super-cell with $l \times m \times n$ copies of the unit cell and n_s space group symmetry operators will require $n_r = n_s \times l \times m \times n$ total symmetry operators. For large crystals, replicated copies of the unit cell are not required to satisfy the real space cutoff and $n_r = n_s$.

Reciprocal space energy

The PME reciprocal space multipolar electrostatic energy U_{rec} for a unit cell[37] is given by

$$U_{\mathrm{rec}}^{\mathrm{U\cdot C\cdot}} = \frac{1}{2}\sum_{m_1=0}^{K_1-1}\sum_{m_2=0}^{K_2-1}\sum_{m_3=0}^{K_3-1} Q^R(\mathbf{m})\cdot\left(G^R * Q^R\right)(\mathbf{m}), \tag{18}$$

where Q^R is the reciprocal lattice grid of dimension $\{K_1, K_2, K_3\}$ populated with multipoles using B-splines (θ_p),

$$Q^R(k_1,k_2,k_3) = \sum_{s_j=1}^{n_s}\sum_{i=1}^{n_a}\sum_{n_1,n_2,n_3} L_i\left(\mathbf{R}_{s_j}\right)[\theta_p(K_1u_{1i} - k_1 - n_1K_1) \times \theta_p(K_2u_{2i} - k_2 - n_2K_2)\theta_p(K_3u_{3i} - k_3 - n_3K_3)], \tag{19}$$

where L_i is given in Eq. (16), $\mathbf{u}_i$ are the fractional coordinates of atom i, the summation over all integers $\{n_1, n_2, n_3\}$ is finite due to the local support of B-splines, and G^R is the discrete Fourier transform of the coefficients arising from the structure factor. The reciprocal space electrostatic energy for the asymmetric unit is

$$U_{\mathrm{rec}}^{\mathrm{A\cdot U\cdot}} = \frac{1}{2}\sum_{m_1=0}^{K_1-1}\sum_{m_2=0}^{K_2-1}\sum_{m_3=0}^{K_3-1} Q^R_{\mathrm{A.U.}}(\mathbf{m})\cdot\left(G^R * Q^R\right)(\mathbf{m}), \tag{20}$$

where the multipoles experiencing the reciprocal space potential are limited to the asymmetric atoms,

$$Q^R_{\mathrm{A\cdot U\cdot}}(k_1,k_2,k_3) = \sum_{i=1}^{n_a}\sum_{n_1,n_2,n_3} L_i(\mathbf{I})[\theta_p(K_1u_{1i} - k_1 - n_1K_1) \times \theta_p(K_2u_{2i} - k_2 - n_2K_2)\theta_p(K_3u_{3i} - k_3 - n_3K_3)]. \tag{21}$$

Both the grid dimensions and spline order can be set by via FFX properties. Application of the multipoles onto the FFT grid is parallelized spatially using 3D domains,[1] over 2D slices of the grid, or over 1D rows. The convolution is then performed by a parallelized 3D FFT,[59,60] followed by a pointwise multiplication in reciprocal space, and finally the inverse 3D FFT.

POLARIZATION ALGORITHMS

Definition of the self-consistent field

To calculate the energy and gradient for a polarizable force field (e.g., AMOEBA), we must solve for the self-consistent field (SCF) that induces dipoles. FFX supports multiple iterative SCF solvers. The mutual (or self-consistent) induced dipoles ($\mathbf{u}$) are a function of the isotropic atomic polarizability of each atom ($\boldsymbol{\alpha}$) and the total electric field ($\mathbf{E}$) at each site,

$$\mathbf{u} = \boldsymbol{\alpha}\mathbf{E}, \tag{22}$$

where $\mathbf{u}$ and $\mathbf{E}$ are both vectors of dimension $3n$ and the polarizability $\boldsymbol{\alpha}$ is a diagonal $3n \times 3n$ matrix. The total electric field can be broken into a "direct" contribution from permanent multipoles ($\mathbf{E}_{\mathrm{direct}}$) and the contribution from induced dipoles,

$$\mathbf{u} = \boldsymbol{\alpha}(\mathbf{E}_{\mathrm{direct}} + \mathbf{T}\mathbf{u}), \tag{23}$$

where $\mathbf{T}$ is a $3n \times 3n$ matrix of interaction tensor elements that operate on the induced dipole vector to produce the induced field at atomic sites. The induced dipoles can be solved analytically,

$$\mathbf{u} = \mathbf{C}^{-1}\mathbf{E}_{\mathrm{direct}}, \tag{24}$$

where explicit inversion of the matrix $\mathbf{C} = \left(\boldsymbol{\alpha}^{-1} - \mathbf{T}\right)$ scales $O\left(n^3\right)$. Better scaling is achieved through iterative methods, including successive over-relaxation (SOR), a preconditioned conjugate gradient (PCG) solver, or an approach based on optimized perturbation theory (OPT). Each method will be briefly discussed here, while more details are available from the work of Lipparini *et al.*[109]

Successive over relaxation (SOR)

The SOR technique is a variant of the Gauss–Seidel method for solving a linear system of equations. It incorporates the previous solution of the system of linear equations with a variable called the relaxation factor (0.7 by default). Although the convergence rate can be improved slightly by tuning the relaxation factor to the system of interest, SOR requires significantly more iterations than the preconditioned conjugate gradient method described below to achieve the same convergence criteria. By default, the SCF convergence criteria are to reduce the change in induced dipole magnitude between iterations below a threshold of 1.0×10^{-6} rms D.

Preconditioned conjugate gradient (PCG)

The objective of the PCG technique is to minimize the residual found by moving all components of the linear system $\mathbf{Ax} = \mathbf{b}$ to the right-hand side (r.h.s.). The notation for AMOEBA is achieved by rearranging Eq. (23) to give the residual as $\mathbf{r} = \mathbf{E}_{\text{direct}} - \mathbf{Cu}$. A preconditioner is used to improve the condition number of the matrix $\mathbf{C}$. In FFX, the preconditioner is based on using a short real space cutoff with a default value of 4.5 Å (with no reciprocal space contribution to the field). The PCG method typically reduces the rms Debye change between iterations by an order of magnitude each cycle (i.e., just six or seven cycles are needed to reach the default convergence criteria).

FFX allows users to define the polarization model, where the default value of "mutual" (i.e., SOR or PCG is used to iteratively converge the SCF) can be changed to "direct" or "none." The "direct" option includes the field due to permanent multipoles, but not the field due to induced dipoles themselves (i.e., the second term on the r.h.s. of Eq. (23) is not included). Selecting "none" eliminates the polarization energy term from the potential energy and is useful for relaxing poor starting coordinates whose SCF is unstable.

DYNAMICS METHODS

Integrators and controls

Several integrators are implemented to propagate degrees of freedom based on Newton's equations of motion during molecular dynamics (MD) simulations. Velocity Verlet provides the simplest integration scheme that is numerically stable and time-reversible.[110,111] The closely related Beeman[70] algorithm for integration produces an identical trajectory to velocity Verlet with a modification that calculates velocities more accurately and better conserves energy. FFX implements a reversible-RESPA integrator[112] that offers multiple-time step simulations through the separation of long-range force calculations from the short-range. The short-range forces are calculated at each time step by a position Verlet algorithm. The position Verlet offers greater stability than velocity Verlet if large time steps are used. The long-range forces are calculated at n time steps, reducing the computational cost of integration. FFX offers two thermostats for use in conjunction with the integrators: the thermostat of Berendsen *et al.*[117] and Bussi–Donadio–Parrinello[113] thermostat. The Berendsen thermostat causes the system temperature to decay exponentially toward a target temperature, but it does not produce particle velocities that are consistent with sampling from the canonical ensemble. The Bussi thermostat can be considered a global version of the (local) Langevin thermostat described below, and it produces particle velocities consistent with rigorous sampling from the canonical ensemble.

The Langevin dynamics[119,120] integrator incorporates two forces: a viscous damping force proportional to particle velocity and a random force representing the effects of collisions with molecules in the environment. The random forces are pulled from a Gaussian distribution with a mean of zero. The magnitude of both the viscous damping and random forces are controlled by a collision frequency parameter (default in FFX is 91/ps to mimic water-like viscosity[114]). In this way, the Langevin integrator provides rigorous temperature control. Finally, pressure control is achieved through a Monte Carlo barostat that obeys lattice system constraints.[115]

Implementations

The default implementation for performing molecular dynamics leverages shared memory parallelism (SMP) constructs defined by the Parallel Java API. This code path is currently recommended for small systems (e.g., pharmaceutical crystals), for use with orthogonal space tempering, or for refinement against experimental data. For GPU acceleration of systems without space group symmetry, FFX offers an interface with OpenMM.[123] This latter code path is recommended for simulating large systems, such as proteins and/or DNA solvated in a periodic box of water.

Constant pH molecular dynamics

FFX comes packaged with the first continuous constant pH molecular dynamics (CpHMD) algorithm for a polarizable atomic multipole potential.[32] The dynamics of the constant pH method are governed by an extended Hamiltonian of the form[116,117]

$$\mathcal{H}(\mathbf{X},\boldsymbol{\theta}) = U_{bond}(\mathbf{X}) + U_{nbond}(\mathbf{X},\boldsymbol{\theta}) + U^{*}(\boldsymbol{\theta}) + \sum_{i}^{N} 0.5 m_i \dot{X}_i^{\,2} + \sum_{k}^{N_{titr}} 0.5 m_k \dot{\theta}_k^{\,2}, \tag{25}$$

where i is the index over atoms and k is the index over both titration and tautomer extended system variables. $\mathbf{X}$ is the Cartesian coordinate vector, and $\boldsymbol{\theta}$ is a vector of both titration and tautomer extended system particles. The titration (λ_k) and tautomer (ζ_k) states are bound between 0 and 1 through the relation $\lambda_k \text{ or } \zeta_k = sin^2(\theta_k)$. These particles are handled by a Langevin integrator with custom friction and particle mass parameters optimized for the algorithm. The θ particles propagate along with atomic particles, although calculation of the extended system forces is currently restricted to a CPU-only implementation. However, there is support for a hybrid CPU/GPU approach to accelerate the method. To achieve the acceleration, θ particles are "frozen," while the atomic coordinates are propagated on the GPU allowing coordinate/titration steps to be followed by a defined number of coordinate-only steps to achieve improved sampling.[32] Sampling is further enhanced through the use of pH replica exchange (RepEx).[118] The CpHMD RepEx protocol runs multiple simulations on a pH ladder simultaneously. Throughout the simulations, there are periodic attempts to exchange pH's between simulations according to the Metropolis criterion,

$$P = \begin{cases} 1 & \text{if } \Delta \le 0, \\ e^{-\beta \Delta E} & \text{otherwise}, \end{cases} \tag{26}$$

where β is given by $1/k_B T$ and ΔE is defined as

$$\Delta E = U_{pH}(\mathbf{X}_{A\ at\ B}, \boldsymbol{\theta}_{A\ at\ B}) + U_{pH}(\mathbf{X}_{B\ at\ A}, \boldsymbol{\theta}_{B\ at\ A}) - (U_{pH}(\mathbf{X}_A, \boldsymbol{\theta}_A) + U_{pH}(\mathbf{X}_B, \boldsymbol{\theta}_B)), \tag{27}$$

where A and B represent the two ensembles considered in the exchange. The pH exchange protocol also supports the CPU/GPU hybrid acceleration method described above. The combination of replica exchange and GPU acceleration results in the first tractable CpHMD simulations of proteins using the AMOEBA force field.

FREE ENERGY CALCULATIONS

The *Thermodynamics* command supports estimation of free energy differences using two complimentary approaches. The first approach applies free energy perturbation (FEP), the Bennett Acceptance Ratio (BAR) method, and/or Multi-State BAR (MBAR) to sample from ensembles defined by pairs of λ values. It then accumulates the overall free energy difference (and its statistical uncertainty) over a path defined by a series of λ windows.[127,119] The second approach uses the orthogonal space tempering (OST) biased sampling strategy to estimate the free energy difference while overcoming hidden barriers.[38,129] Future work will access opportunities to re-weight biased samples from OST and thereby permit evaluation with MBAR.

Free energy perturbation, BAR, and MBAR

For the BAR and MBAR methods, the λ values range from 0 to 1, where λ = 0 indicates the initial state and λ = 1 indicates the end state. Intermediate simulation windows sample from an ensemble that represents an unphysical alchemical state. The resulting samples are stored in an archive file (.arc) for each λ value, which can then read by the *BAR* or *MBAR* commands during the estimation of free energy differences.

For example, *Thermodynamics* can be used to estimate the free energy difference associated with amino acid substitutions within the protein structure. In this case, the thermodynamic path is broken into four sets of simulations to slowly remove the wildtype (WT) amino acid and slowly grow in the mutant (MT) amino acid. The first set turns off the electrostatics from the WT side chain, the second turns off van der Waals from the WT side chain, the third turns on the van der Waals for the MT side chain, and the final simulation turns on the electrostatics for the MT side chain.

To remove the electrostatics contribution of the WT side chain, FFX splits the simulation into a set number of alchemical intermediate simulations, typically referred to as windows. The user sets the alchemical atoms to be affected by the λ states (i.e., the side chain atoms). Each window will have a different λ value evenly distributed across *n* windows from zero to one. The λ value acts as a dial to tune the contribution from the electrostatics from on (λ = 1) to off (λ = 0). FFX initiates windowed alchemical simulations with *Thermodynamics* when *n* windows are set. Each simulation is run in parallel and has GPU acceleration. The result is *n* archive files with snapshots from thermodynamics simulations used for further analysis to estimate the overall free energy change. The *BAR* command estimates the free energy difference from the collected samples at each λ value.[127]

An extension of BAR is MBAR where equilibrium samples from multiple thermodynamic states can be utilized to construct a statistically optimal free energy estimator. MBAR reduces to BAR when used on two thermodynamic states.

Our recent expansion of the dual topology framework to directly estimate free energy differences between polymorphs is presented as an additional demonstration for both BAR and MBAR. A dual topology simulation via the *Thermodynamics* command was used to sample states between experimental crystals with lattice parameters and coordinates minimized to convergence criteria of 0.05 kcal/mol/Å according to parameters generated by PolType2[120]

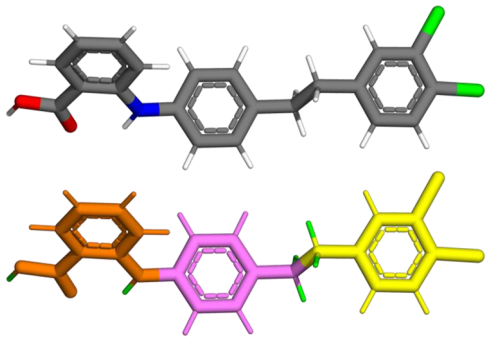


FIG. 5. The upper panel shows compound XXIII colored by an atomic number with chlorine green, oxygen red, nitrogen blue, carbon gray, and hydrogen white. The lower panel is colored by dual topology groups, with orange, violet, and yellow atoms each assigned a unique symmetry operator to map between polymorphs. The hydrogen atoms colored green in the lower panel were given independent degrees of freedom for the two topologies.

for 2-((4-(2-(3,4-dichlorophenyl)ethyl)phenyl)amino)benzoic acid (CSD ID: XAFPAY) also known as compound XXIII. Symmetry operators to map between the two polymorph end states were generated via the PAC algorithm. Individual symmetry operators were generated for each of the aromatic rings and their constituents as shown in Fig. 5. Several hydrogen atoms had inter-polymorph distances larger than 1.0 Å after the application of the symmetry operator that was applied to match their bonded heavy atom between polymorphs; therefore, those hydrogen atoms were treated as alchemical.

One CPU core was used to generate sampling for each of 21 lambda windows in parallel. Each window sampled for one nanosecond (~11.1 h on an Intel® Xeon® Gold 6330 CPU at 2.00 GHz) with coordinate snapshots saved each picosecond. The last 900 snapshots were evaluated MBAR to computed polymorph relative free energy differences, with results given in Tables III and IV. For comparison, relative lattice potential energy differences are provided for AMOEBA (which was used for the free energy difference calculations) along with several other approaches featured in the sixth blind test of organic crystal structure prediction organized by the Cambridge Crystallographic Data Centre.[121]

The estimated free energy differences from both BAR and MBAR follow the trends observed from the AMOEBA lattice potential energy differences. Individual free energy differences between each of the experimental polymorphs, their associated uncertainties, and cycle closure errors can be found in Table IV.

Orthogonal space tempering

First-order generalized ensemble (GE) methods (e.g., metadynamics[122]) can eliminate barriers along the chosen variable path (i.e., λ). However, free energy barriers perpendicular to the variable path can impede exhaustive sampling of the entire free energy surface (e.g., conformational barriers).[123] FFX implements the second-order GE orthogonal space tempering (OST) sampling[8] to cross free energy barriers by combining transition-tempered metadynamics[124] with the orthogonal space random walk method.[38]

TABLE III. Relative lattice energies for the experimental polymorphs of compound XXIII using a variety of models, compared to AMOEBA relative polymorph free energy differences (kcal/mol).

Model[a]	Method	Form A	B	C	D	E
Team 3: Day *et al.*	Multipoles and exp-6	0.3	1.3	0.0	0.6	0.1
Team 5: van Eijck	Charges and exp-6	1.0	0.0	1.3	1.3	1.1
Team 14: Neumann *et al.*	PBE + Neuman-Perrin	0.9	0.0	0.0	0.7	0.5
Team 18: Price *et al.*	Multipoles and exp-6	2.3	0.0	0.8	2.2	1.3
Potential energy	AMOEBA	1.30	0.00	1.80	1.34	1.78
MBAR relative ΔG	AMOEBA	0.39	0.00	0.87	0.66	0.84

[a]Model entries that start with "Team" were converted from Table S12 of the sixth blind test of organic crystal structure prediction.

The total potential energy of the OST ensemble, $U_{OST}(\lambda, \mathbf{x})$, is the summation of AMOEBA force field energy terms and the sum of a one-dimensional and two-dimensional time-dependent bias, $f_m(\lambda) + g_m(\lambda, \mathrm{F}_\lambda)$, as demonstrated in the following equation:

$$U_{OST}(\lambda, \mathbf{x}) = U_{\mathrm{AMOEBA}}(\lambda, \mathbf{x}) + f_m(\lambda) + g_m(\lambda, \mathrm{F}_\lambda). \tag{28}$$

The one-dimensional bias, $f_m(\lambda)$, is obtained through the thermodynamic integration as given by

$$f_m(\lambda) = -\int_0^\lambda \langle \partial U/\partial\lambda \rangle_{\lambda'} \mathrm{d}\lambda', \tag{29}$$

where the ensemble average is further clarified in Eq. (30) where $\beta = \frac{1}{k_B T}$,

$$\langle \partial U/\partial\lambda \rangle_{\lambda'} = \frac{\int_{\partial U/\partial\lambda} \partial U/\partial\lambda \cdot \exp\left[\beta g_m(\lambda, \mathrm{F}_\lambda)\right] \delta\left(\lambda - \lambda'\right)}{\int_{\partial U/\partial\lambda} \exp\left[\beta g_m(\lambda, \mathrm{F}_\lambda)\right] \delta\left(\lambda - \lambda'\right)}. \tag{30}$$

The time-dependent two-dimensional repulsive potentials, $g_m(\lambda, \mathrm{F}_\lambda)$, are defined in the following equation:

$$g_m(\lambda, F_\lambda) = \sum_{t_i} h(t_i) \cdot \exp\left[\frac{|\lambda - \lambda(t_i)|^2}{2{w_1}^2} \cdot \frac{|F_\lambda - F_\lambda(t_i)|^2}{2{w_2}^2}\right]. \tag{31}$$

The "tempering" in OST denotes a non-constant bias height, $h(t_i)$, that decreases as the simulation proceeds. The intention of the bias is to progressively flatten the path. Therefore, the bias height decreases asymptotically toward 0 based on the following expression:

$$h(t_i) = h(t_0) \cdot \exp\left[\frac{\min\left(0, V_{th} - V^*(t_i)\right)}{\Delta T}\right], \tag{32}$$

where the initial Gaussian bias height $h(t_0)$ is typically chosen as ~0.02 to 0.05 kcal/mol with the standard deviation equal to two bins in either dimension ($w_1 = 0.01$ and $w_2 = 4.0$ kcal/mol). The height is also truncated after five bins during the evaluation of the 2D bias. Tempering begins after a small amount of bias has been added along the entire path defined by the tempering threshold (V_{th}). The rate of exponential decay is determined by the ΔT parameter that has a default value of 2 $k_B T$. Finally, the amount of decay is tempered based on $V^*(t_i)$ defined in Eq. (33) where the max operation for each fixed λ is over the range of F_λ values (i.e., the 2D g_m histogram is reduced to a 1D function of λ), followed by the min operation over λ,

$$V^*(t_i) = \min_\lambda \left[\max_{F_\lambda} g_m(\lambda, F_\lambda)\right]. \tag{33}$$

TABLE IV. Free energy differences and uncertainties in kcal/mol per molecule between experimental polymorphs of compound XXIII computed using MBAR.

Transformation	ΔG	Uncertainty
B → D	0.656	0.067
D → E	0.183	0.097
E → C	0.027	0.092
C → A	−0.482	0.095
A → B	−0.386	0.065
Cycle closure	−0.001	

Figure 6 represents a visualization of the OST algorithm as applied to a simulation of carbamazepine as it transitions between vacuum to crystalline phases. In Fig. 6(a), the ensemble average partial derivative of the potential energy with respect to the path variable, λ, is plotted as a black solid line. The deposition free energy difference at each λ relative to the vacuum state (i.e., $\lambda = 0$) is represented as a blue dashed line. Figure 6(b) demonstrates a contour plot of the total OST bias (i.e., the summation of both 1D and 2D bias contributions). The addition of OST bias promotes sampling away from the free energy minimum, while in the crystalline phase (near or at $\lambda = 1$) encouraging the exploration of free energy barriers along $\partial U/\partial\lambda$ (orthogonal to the λ path). Figure 6(c) features only the two-dimensional bias contribution as a contour plot by removing the 1D bias, which is constant for each λ value. The partial derivatives needed for OST ($\partial U/\partial X$, $\partial U/\partial\lambda$, $\partial^2 U/\partial\lambda^2$, $\partial^2 U/\partial X\partial\lambda$) are supported for softcore van der Waals interactions, softcore charge (or multipolar) interactions, and polarization energy contributions within the CPU code path,[6] but not yet available within OpenMM.[129]

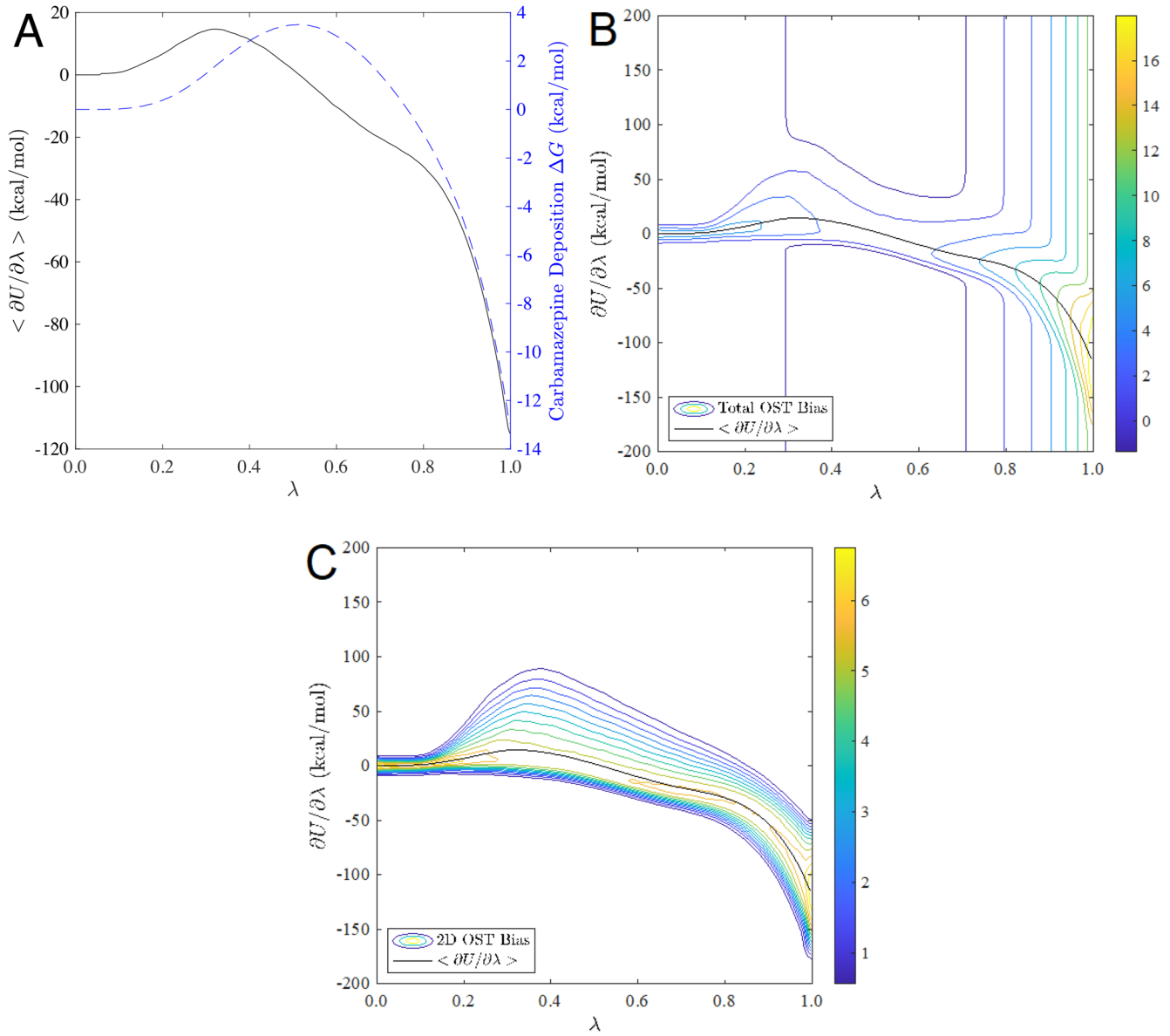


FIG. 6. Plots illustrating the use of the OST sampling approach on a carbamazepine simulation between vacuum ($\lambda = 0$) and crystalline ($\lambda = 1$) phases. (a) The ensemble average partial derivative of the potential energy with respect to the path variable λ, $\langle \partial U/\partial\lambda \rangle$, (black solid line) and the deposition free energy difference obtained as the integration over the phase transition path (blue dashed line). (b) A contour plot of the combined contributions from the 1D and 2D biases. (c) A contour plot of only the 2D bias contributions. Both (b) and (c) show $\langle \partial U/\partial\lambda \rangle$ as a function of λ.

PROPERTIES AND ANALYSIS

A family of commands leverage coordinate snapshots and trajectories to analyze the properties of organic materials and biomolecules. The *Energy* command provides a summary of the potential energy contributions for all terms in use (e.g., bonds, angles, torsion, van der Waals, permanent electrostatics, polarization, and various restraints). The *Volume* command supports calculation of molecular volume and surface area using either the GaussVol[125] or the Connolly algorithm.[126] The former is limited to calculation of van der Waals volume and surface area. The latter additionally supports calculation of both (1) molecular volume and surface area and (2) solvent excluded volume and solvent accessible surface area.

BAR and *MBAR* can be applied to single or dual topology systems after simulation with molecular dynamics or Metropolis

Monte Carlo techniques.[127,128] *BAR* can save Tinker.bar files that contain energy evaluations for snapshots in each λ value, which is particularly useful for re-evaluating free energy differences using subsets of the coordinate snapshots (e.g., to compare free energy difference estimates from the first and second half of trajectories). With an additional script *SortArc*, coordinate snapshots from thermodynamics simulations using the replica exchange algorithm can be reorganized and analyzed with BAR.[129] Specifically, *SortArc* sorts the multi-state snapshot files into archive files containing snapshots for only a single state (i.e., all snapshots at λ = 1 will be contained in a single .arc file).

Finally, the *Histogram* command is used to load a 2D count matrix from an OST simulation (stored in a histogram file *.his). *Histogram* then leverages the counts to first compute the ensemble average partial derivative of the potential energy with respect to the state variable $\langle \partial U/\partial \lambda \rangle$ followed by estimation of the free energy difference as a function λ using thermodynamic integration. The *Histogram* command can optionally save out text files that are input to a simple Matlab script that plots the 1D potential of mean force (PMF) and 2D OST bias to visualize the free energy surface as a function of both λ and $\langle \partial U/\partial \lambda \rangle$.

ROTAMER OPTIMIZATION OF MANY-BODY POTENTIAL

Rotamer optimization can be performed on a protein using the *ManyBody* command. The total energy of the protein is optimized with the many-body expansion through movement of defined side chain positions relative to the backbone (rotamers). The user can choose between two rotamer libraries with rotamers for every amino acid, including the protonated and deprotonated form of titratable residues.[130,131] *ManyBody* results in an optimized structure with side chains in the global minimum energy conformation (GMEC). The user can improve the rotamer optimization by removing default approximations within the many-body expansion. The distance cutoffs for both two- and three-body energy terms can be increased to capture interactions of more distal residues. Users can adjust energy cutoffs for clash pruning, add or remove the three-body energy term, or allow soft-coring of clashes. There are multiple residue selection algorithms that include providing start and end residues as well as a user-specified list of residues. Rotamer optimization with *ManyBody* improves the overall structural quality of proteins.

For example, rotamer optimization with local minimization was applied to AlphaFold2[132] deep learning algorithm predicted isoform-specific protein structures for 218 protein-coding genes found in the Deafness Variation Database (DVD).[133] The MolProbity algorithm evaluated structures before and after optimization to quantify the improvement in atomic clashes, backbone angles, and side chain conformations.[5] Structures from AlphaFold2 had an average clash score of 20.75 (i.e., number of unphysical overlaps per 1000 atoms), and the overall MolProbity score was 2.86 (i.e., the protein models were consistent with those from 2.86 Å resolution diffraction data). After optimization with FFX, the average clash score of the protein models dropped to only 0.11, and the protein structures had an average MolProbity score of 0.97, which is consistent with models from atomic resolution diffraction data.

REFINEMENT AGAINST X-RAY, NEUTRON, AND JOINT X-RAY/NEUTRON TARGETS

FFX serves as a platform to systematically explore the use of advanced potential energy functions (e.g., AMOEBA) coupled to rigorous long-range electrostatics (e.g., PME) in the context of both local and global optimization strategies during refinement of biomolecular models against experimental datasets. The target function weights the contributions of the data (e.g., x-ray diffraction) with prior chemical knowledge. This can be motivated within a Bayesian framework where the probability of the model parameters $\mathbf{X}$ (e.g., atomic coordinates, b-factors, and occupancies) given data $\mathbf{D}$ is given by $p(\mathbf{X}|\mathbf{D})$ and is proportional to

$$p(\mathbf{X}|\mathbf{D}) \propto p(\mathbf{D}|\mathbf{X})e^{(-U(\mathbf{X})/k_BT)}, \tag{34}$$

where $U(\mathbf{X})$ is a force field potential energy, k_B is Boltzmann's constant, T is the absolute temperature, and $p(\mathbf{D}|\mathbf{X})$ is the likelihood of the data, given the model. For convenience, the maximum of $p(\mathbf{X}|\mathbf{D})$ can be found by minimization of the negative logarithm of Eq. (34) as given by

$$E(\mathbf{X}) = -w_A \ln[p(\mathbf{D}|\mathbf{X})] + U(\mathbf{X}), \tag{35}$$

where the weight of the data $w_A = k_BT$ should be ~0.6 near 300 K.[10] FFX supports reciprocal space refinement against either x-ray or neutron diffraction data and joint x-ray plus neutron refinement using the target function,

$$E(\mathbf{X}) = -w_{\mathrm{A,Xray}} \ln[p(\mathbf{D}_{\mathrm{Xray}}|\mathbf{X})] - w_{\mathrm{A,Neutron}} \ln[p(\mathbf{D}_{\mathrm{Neutron}}|\mathbf{X})] + U(\mathbf{X}), \tag{36}$$

where the relative weight of each data source ($w_{\mathrm{A,Xray}}$, $w_{\mathrm{A,Neutron}}$) can be configured.

Real space refinement is also supported using CryoEM electron density maps or those from a Fourier synthesis, such as $(F_o - F_c)$ or $(2F_o - F_c)$, where F_o are observed structure factors and F_c are calculated structure factors. Prior work has shown that PME electrostatics, especially when combined with a polarizable multipole force field, improves structure quality.[1] Further model improvements are afforded by use of a differentiable bulk solvent model[9] and global optimization of sidechain rotamers using a many-body expansion of Eq. (35) coupled to Goldstein elimination criteria.[11] Work remains to implement a convergent and efficient implementation of generalized Kirkwood continuum electrostatics under periodic boundary conditions, which will open the door to estimation of free energy differences between model conformations and chemical compositions.

UNIT TESTING AND CONTINUOUS INTEGRATION

FFX currently includes more than 500 JUnit tests, with many building on the commands described previously (e.g., *Energy*, *Thermodynamics*, and *ManyBody*) to validate both core functionality and command line flags. The number of JUnit tests for each package and current percentage of code covered by each test is summarized in Table V. No JUnit tests specific to Parallel Java have been created due to the careful work by its original author.[57] We also

TABLE V. The number of unit tests and code coverage for most of the FFX packages.

Package	Number of tests	Code coverage (%)
Algorithms	100	32
Crystal	13	91
Numerics	175	82
OpenMM	···	58
Potential	201	60
Refinement	27	40
Utilities	2	14

lack JUnit tests for the graphical user interface. The OpenMM classes are covered by JUnit tests in the Potential and Algorithms packages. A Jenkins continuous integration server is used to automatically run all FFX unit tests and generate the Force Field X website based on polling the GitHub repository. The FFX website includes documentation for all commands and properties generated from annotations within Java code. Future work will focus on expanding test coverage for emerging methods within the Algorithms and Refinement packages.

SHARED MEMORY, MPI, AND GPU PARALLELIZATION

Shared memory parallelization

FFX leverages Parallel Java (PJ) for shared memory parallelism (SMP)[57] based on classes that offer functionality that is analogous to OpenMP style pragmas. Rather than annotating a loop with a pragma, PJ defines various "ForLoop" classes that are extended and then executed by a collection of threads called a "ParallelTeam."[58] By default, nonbonded forces are calculated using all available threads. For PME, the direct space and reciprocal terms can be calculated concurrently. The direct space contributions are organized using neighbor lists to distribute and balance the workload. Parallelization of the 3D convolution operation that is the basis of reciprocal space PME has been described previously.[1]

Message passing across nodes

FFX executes on a single process by default but supports the cooperation of multiple processes through the use of the PJ *Scheduler* and MPI implementation of PJ. The *Scheduler* organizes execution of a parallel job across a cluster of multiple processes with a user defined number of threads per process and memory per process. The *Scheduler* defaults to evenly splitting all available cores if this property remains unspecified by the user. The *Scheduler* and MPI constructs can be used in conjunction with both the SMP approach described above and the GPU support described below.

For example, OST free energy difference calculations can be accelerated through MPI parallelization over multiple walkers that each contributes counts to the same 2D histogram (Multiple Walker OST). The *Scheduler* launches individual trajectories that each starts from an identical value of the state variable λ or be distributed across the thermodynamic path (i.e., each walker has a unique starting value of λ). The samples from any given walker are communicated to all other walkers such that each process is applying the same OST bias and thereby providing the same estimated free energy difference. The addition of walkers enhances sampling and facilitates convergence. MPI approaches are also employed in the context of both many-body optimization and replica exchange constant pH MD.

Usage of GPUs via OpenMM

OpenMM binaries are bundled with FFX to allow for usage of GPUs.[20] Alternatively, a source code can be built and used via JNA or JExtract. FFX has Java implementations for the majority of C++ classes available in OpenMM (i.e., Context, Platform, and State). Many-body optimization employs OpenMM for force field energy calculations and in turn for self, pair, and triple energy calculations. The initialization of many-body occurs on the CPU, while the AMOEBA energy calculations are performed after creating an OpenMM context and moving to the GPU. The finalization of the global minimum energy conformation is passed back to the CPU.[4] GPU acceleration is also available for MD, AMOEBA/GK calculations, replica exchange constant pH MD (hybrid CPU–GPU

TABLE VI. Comparison of the asymmetric unit (ASU), unit cell (UC), and replicated unit cell that satisfies minimum image convention for several experimental polymorphs of carbamazepine.

CSD ID	Crystalline unit	Space group	Number of molecules	Degrees of freedom	Time for energy and force evaluations (s)
CBMZPN 02	ASU	$P2_1/n$	1	90	0.027
	UC	$P1$	4	360	0.029
	4 × 3 × 3 UC	$P1$	144	12 960	0.733
CBMZPN 12	ASU	$C2/c$	1	90	0.026
	UC	$P1$	8	720	0.055
	2 × 5 × 3 UC	$P1$	240	21 600	1.116
CBMZPN 16	ASU	$Pbca$	1	90	0.034
	UC	$P1$	8	720	0.046
	4 × 3 × 2 UC	$P1$	192	17 280	0.894

TABLE VII. AMOEBA/GK energy evaluations per second with different numbers of GPU's performing a many-body optimization on a 56-residue turkey-ovomucoid third domain protein.

Architecture	AMOEBA energy evaluations (s)
CPU	7.6
1 GPU	29.9
2 GPUs	48.8
4 GPUs	71.7

implementation), etc. SMP and MPI can be used in conjunction with OpenMM.

BENCHMARKS

Energy and force timings for simulating carbamazepine crystalline units are presented in Table VI for three polymorphs to show the efficiency gained by simulating asymmetric units relative to replicated unit cells. Furthermore, the carbamazepine polymorph deposition simulation that produced the plots in Fig. 6 was performed utilizing two threads of a recent Intel CPU (a Xeon Gold 6330 CPU at 2.00 GHz). Simulating for 3.6 h produced 1 ns of sampling using the AMOEBA force field or more than 350 ns/day using all 112 threads/56 cores of a dual CPU configuration.

Rotamer optimization was performed for a turkey-ovomucoid third domain, a 56 amino acid protein structure requiring ~112.5 thousand AMOEBA/GK energy calculations with zero, one, two, and four GPUs on Nvidia A10 s with 28 Intel CPU cores per GPU. The simulation experienced a 9.4× speed up from 7.6 AMOEBA energy calculations per second to 71.7 per second when increasing from no GPU to four GPUs (Table VII).

Finally, for molecular dynamics simulations that can be offloaded to OpenMM (e.g., unit cell lengths that are larger than twice the non-bonded cutoff and do not require symmetry operators), the benchmarks described by the OpenMM developers[20,134] are representative of the performance in FFX.

CONCLUSIONS

This article has demonstrated significant use cases and advancements available in FFX for atomic resolution modeling of organic materials and biomolecules. Specifically, we have highlighted FFX's handling of crystal structures and data, the generalized Kirkwood implicit solvent model, constant-pH MD for the polarizable AMOEBA force field, dual topology methods, and global side chain optimization. FFX development will continue with novel algorithms and advanced treatment of force fields, acid–base chemistry, and prediction of crystal properties. Some methods under active development include a statistical mechanics method for accelerated acid–base chemistry calculations, AMOEBA folding and binding free energy difference predictions for amino acid mutations, GPU acceleration of constant pH MD, and methods for relative polymorph free energy differences. It is our hope that the open-source and freely available FFX software can serve as a computational microscope to understand the biophysics of organic materials and biomolecules.

ACKNOWLEDGMENTS

A.C.T., R.A.C.G., and M.R.T. were supported by the NSF (National Science Foundation) Graduate Research Fellowship under Grant No. 000390183. R.A.C.G. and A.J.N. were partially supported by a Ballard and Seashore Dissertation Fellowship from the University of Iowa. R.A.C.G., G.Q., and L.M.C. were partially supported by fellowships from the University of Iowa Office of Undergraduate Research. J.W.P. and P.R. were supported by NIH under Grant Nos. R01GM114237 and R01GM106137. J.S. was supported by NIH under Grant No. R35GM148261. M.J.S. was supported by NSF under Grant No. CHE-1751688. M.J.S. and R.J.H.S. were supported by NIH under Grant No. R01DC012049. M.J.S. and J.J.M. were supported by the Simons Foundation SFARI under Award No. 1019623.

AUTHOR DECLARATIONS

Conflict of Interest

J.-P. Piquemal, J. W. Ponder, and P. Ren are cofounders of Qubit Pharmaceuticals.

Author Contributions

R.A.G., A.J.N., and A.C.T. contributed equally to this work.

Rose A. Gogal: Conceptualization (equal); Data curation (equal); Formal analysis (equal); Investigation (equal); Methodology (equal); Software (equal); Validation (equal); Visualization (equal); Writing – original draft (equal); Writing – review & editing (equal). **Aaron J. Nessler**: Conceptualization (equal); Data curation (equal); Formal analysis (equal); Investigation (equal); Methodology (equal); Software (equal); Validation (equal); Visualization (equal); Writing – original draft (equal); Writing – review & editing (equal). **Andrew C. Thiel**: Conceptualization (equal); Data curation (equal); Formal analysis (equal); Methodology (equal); Software (equal); Validation (equal); Visualization (equal); Writing – original draft (equal); Writing – review & editing (equal). **Hernan V. Bernabe**: Data curation (equal); Methodology (equal); Software (equal); Writing – review & editing (equal). **Rae A. Corrigan Grove**: Data curation (equal); Formal analysis (equal); Investigation (equal); Methodology (equal); Software (equal); Validation (equal); Visualization (equal); Writing – original draft (supporting); Writing – review & editing (supporting). **Leah M. Cousineau**: Data curation (supporting); Investigation (supporting); Methodology (supporting); Software (supporting); Visualization (supporting); Writing – review & editing (supporting). **Jacob M. Litman**: Data curation (supporting); Formal analysis (supporting); Methodology (equal); Software (equal); Validation (equal); Writing – original draft (supporting); Writing – review & editing (equal). **Jacob M. Miller**: Data curation (supporting); Formal analysis (supporting); Investigation (supporting); Methodology (supporting); Software (supporting); Validation (equal); Writing – original draft (supporting); Writing – review & editing (equal). **Guowei**

Qi: Formal analysis (supporting); Methodology (supporting); Software (supporting); Validation (supporting); Writing – review & editing (supporting). **Matthew J. Speranza**: Data curation (equal); Formal analysis (equal); Investigation (equal); Methodology (equal); Software (equal); Validation (equal); Writing – original draft (equal); Writing – review & editing (equal). **Mallory R. Tollefson**: Data curation (supporting); Formal analysis (supporting); Investigation (supporting); Methodology (supporting); Software (supporting); Validation (supporting); Writing – original draft (supporting); Writing – review & editing (supporting). **Timothy D. Fenn**: Conceptualization (supporting); Data curation (supporting); Investigation (supporting); Methodology (supporting); Software (supporting); Validation (supporting); Visualization (supporting); Writing – review & editing (supporting). **Jacob J. Michaelson**: Funding acquisition (supporting); Resources (supporting); Supervision (supporting); Writing – review & editing (supporting). **Okimasa Okada**: Data curation (supporting); Formal analysis (supporting); Investigation (equal); Methodology (equal); Software (supporting); Validation (equal); Writing – review & editing (supporting). **Jean-Philip Piquemal**: Data curation (equal); Funding acquisition (equal); Methodology (equal); Resources (equal); Validation (equal); Writing – review & editing (equal). **Jay W. Ponder**: Funding acquisition (supporting); Methodology (supporting); Resources (supporting); Software (supporting); Validation (supporting); Writing – review & editing (supporting). **Jana Shen**: Formal analysis (supporting); Investigation (supporting); Methodology (equal); Software (supporting); Validation (supporting); Writing – review & editing (supporting). **Richard J. H. Smith**: Funding acquisition (supporting); Methodology (supporting); Resources (supporting); Writing – review & editing (supporting). **Wei Yang**: Investigation (supporting); Methodology (supporting); Resources (supporting); Software (supporting); Writing – review & editing (supporting). **Pengyu Ren**: Data curation (supporting); Funding acquisition (equal); Methodology (equal); Project administration (supporting); Resources (supporting); Software (supporting); Validation (supporting); Writing – review & editing (supporting). **Michael J. Schnieders**: Conceptualization (equal); Data curation (equal); Formal analysis (equal); Funding acquisition (equal); Investigation (equal); Methodology (equal); Project administration (equal); Resources (equal); Software (equal); Supervision (equal); Validation (equal); Visualization (equal); Writing – original draft (equal); Writing – review & editing (equal).

DATA AVAILABILITY

The Force Field X code is available on GitHub (https://github.com/SchniedersLab/forcefieldx) under the GNU General Public License, version 3, with the Classpath Exception. Force Field X documentation, including instructions to install precompiled versions or build the software from source code, is available from the University of Iowa (https://ffx.biochem.uiowa.edu).

REFERENCES

[1] M. J. Schnieders, T. D. Fenn, and V. S. Pande, "Polarizable atomic multipole X-ray refinement: Particle mesh Ewald electrostatics for macromolecular crystals," J. Chem. Theory Comput. **7**, 1141–1156 (2011).

[2] R. A. Corrigan *et al.*, "Implicit solvents for the polarizable atomic multipole AMOEBA force field," J. Chem. Theory Comput. **17**, 2323–2341 (2021).

[3] R. A. Corrigan *et al.*, "A generalized Kirkwood implicit solvent for the polarizable AMOEBA protein model," J. Chem. Phys. **159**, 054102 (2023).

[4] M. R. Tollefson *et al.*, "Structural insights into hearing loss genetics from polarizable protein repacking," Biophys. J. **117**, 602–612 (2019).

[5] M. R. Tollefson *et al.*, "Assessing variants of uncertain significance implicated in hearing loss using a comprehensive deafness proteome," Hum. Genet. **142**, 819–834 (2023).

[6] M. J. Schnieders *et al.*, "The structure, thermodynamics, and solubility of organic crystals from simulation with a polarizable force field," J. Chem. Theory Comput. **8**, 1721–1736 (2012).

[7] I. J. Nessler, J. M. Litman, and M. J. Schnieders, "Toward polarizable AMOEBA thermodynamics at fixed charge efficiency using a dual force field approach: Application to organic crystals," Phys. Chem. Chem. Phys. **18**, 30313–30322 (2016).

[8] J. Litman, A. C. Thiel, and M. J. Schnieders, "Scalable indirect free energy method applied to divalent cation-metalloprotein binding," J. Chem. Theory Comput. **15**, 4602–4614 (2019).

[9] T. D. Fenn, M. J. Schnieders, and A. T. Brunger, "A smooth and differentiable bulk-solvent model for macromolecular diffraction," Acta Crystallogr., Sect. D: Biol. Crystallogr. **66**, 1024–1031 (2010).

[10] T. D. Fenn and M. J. Schnieders, "Polarizable atomic multipole X-ray refinement: Weighting schemes for macromolecular diffraction," Acta Crystallogr., Sect. D: Biol. Crystallogr. **67**, 957–965 (2011).

[11] S. D. LuCore *et al.*, "Dead-end elimination with a polarizable force field repacks PCNA structures," Biophys. J. **109**, 816–826 (2015).

[12] P. Ren and J. W. Ponder, "Polarizable atomic multipole water model for molecular mechanics simulation," J. Phys. Chem. B **107**, 5933–5947 (2003).

[13] J. W. Ponder *et al.*, "Current status of the AMOEBA polarizable force field," J. Phys. Chem. B **114**, 2549–2564 (2010).

[14] P. Ren, C. Wu, and J. W. Ponder, "Polarizable atomic multipole-based molecular mechanics for organic molecules," J. Chem. Theory Comput. **7**, 3143–3161 (2011).

[15] Y. Shi *et al.*, "Polarizable atomic multipole-based AMOEBA force field for proteins," J. Chem. Theory Comput. **9**, 4046–4063 (2013).

[16] C. Zhang *et al.*, "AMOEBA polarizable atomic multipole force field for nucleic acids," J. Chem. Theory Comput. **14**, 2084–2108 (2018).

[17] M. Harger *et al.*, "Tinker-OpenMM: Absolute and relative alchemical free energies using AMOEBA on GPUs," J. Comput. Chem. **38**, 2047–2055 (2017).

[18] J. A. Rackers *et al.*, "Tinker 8: Software tools for molecular design," J. Chem. Theory Comput. **14**, 5273–5289 (2018).

[19] L. Lagardère *et al.*, "Tinker-HP: A massively parallel molecular dynamics package for multiscale simulations of large complex systems with advanced point dipole polarizable force fields," Chem. Sci. **9**, 956–972 (2018).

[20] P. Eastman *et al.*, "OpenMM 7: Rapid development of high performance algorithms for molecular dynamics," PLoS Comput. Biol. **13**, e1005659 (2017).

[21] A. T. Brünger *et al.*, "Crystallography & NMR system: A new software suite for macromolecular structure determination," Acta Crystallogr., Sect. D: Biol. Crystallogr. **54**, 905–921 (1998).

[22] A. T. Brunger, "Version 1.2 of the crystallography and NMR system," Nat. Protoc. **2**, 2728–2733 (2007).

[23] J. Gosling and H. McGilton, The Java language environment, Sun Microsystems Computer Company, 1995.

[24] J. Gosling, *The Java Language Specification* (Addison-Wesley Professional, 2000).

[25] K. Arnold, J. Gosling, and D. Holmes, *The Java Programming Language* (Addison Wesley Professional, 2005).

[26] T. Würthinger *et al.*, in *Proceedings of the 2013 ACM International Symposium on New Ideas, New Paradigms, and Reflections on Programming and Software* (Association for Computing Machinery, 2013), pp. 187–204.

[27] See https://www.graalvm.org for Oracle, Build faster, smaller, leaner applications, 2023.

[28] J. Clarkson *et al.* in *Proceedings of the 15th International Conference on Managed Languages and Runtimes* (Association for Computing Machinery, Linz, Austria, 2018), Article 4.

[29]J. Fumero *et al.*, in *Proceedings of the 15th ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments* (Association for Computing Machinery, Providence, RI, 2019), pp. 165–178.
[30]Q. Zhang, L. Xu, and B. Xu, "Python meets JIT compilers: A simple implementation and a comparative evaluation," Software: Pract. Exper. **54**, 225 (2024).
[31]See https://www.graalvm.org/python for Oracle, High-performance modern Python, 2023.
[32]A. C. Thiel *et al.*, "Constant-pH simulations with the polarizable atomic multipole AMOEBA force field," J. Chem. Theory Comput. **20**, 2921–2933 (2024).
[33]J. G. Kirkwood, "Theory of solutions of molecules containing widely separated charges with special application to zwitterions," J. Chem. Phys. **2**, 351–361 (1934).
[34]M. J. Schnieders and J. W. Ponder, "Polarizable atomic multipole solutes in a generalized Kirkwood continuum," J. Chem. Theory Comput. **3**, 2083–2097 (2007).
[35]T. Darden, D. York, and L. Pedersen, "Particle mesh Ewald: An $N\log(N)$ method for Ewald sums in large systems," J. Chem. Phys. **98**, 10089–10092 (1993).
[36]U. Essmann *et al.*, "A smooth particle mesh Ewald method," J. Chem. Phys. **103**, 8577–8593 (1995).
[37]C. Sagui, L. G. Pedersen, and T. A. Darden, "Towards an accurate representation of electrostatics in classical force fields: Efficient implementation of multipolar interactions in biomolecular simulations," J. Chem. Phys. **120**, 73–87 (2004).
[38]L. Zheng, M. Chen, and W. Yang, "Random walk in orthogonal space to achieve efficient free-energy simulation of complex systems," Proc. Natl. Acad. Sci. U. S. A. **105**, 20227–20232 (2008).
[39]D. Min *et al.*, "Practically efficient QM/MM alchemical free energy simulations: The orthogonal space random walk strategy," J. Chem. Theory Comput. **6**, 2253–2266 (2010).
[40]J. Desmet, M. D. Maeyer, B. Hazes, and I. Lasters, "The dead-end elimination theorem and its use in protein side-chain positioning," Nature **356**, 539–542 (1992).
[41]R. F. Goldstein, "Efficient rotamer elimination applied to protein side-chains and related spin glasses," Biophys. J. **66**, 1335–1340 (1994).
[42]E. C. Dybeck *et al.*, "A comparison of methods for computing relative anhydrous–hydrate stability with molecular simulation," Cryst. Growth Des. **23**, 142–167 (2023).
[43]F. Bu *et al.*, "High-throughput genetic testing for thrombotic microangiopathies and C3 glomerulopathies," J. Am. Soc. Nephrol. **27**, 1245–1253 (2016).
[44]A. P. DeLuca *et al.*, "Hypomorphic mutations in *TRNT1* cause retinitis pigmentosa with erythrocytic microcytosis," Hum. Mol. Genet. **25**, 44–56 (2016).
[45]A. Simpson *et al.*, "LADD syndrome with glaucoma is caused by a novel gene," Mol. Vis. **23**, 179–184 (2017) PMID: 28400699; PMCID: PMC5373035.
[46]L. A. Lansdon *et al.*, "The use of variant maps to explore domain-specific mutations of FGFR1," J. Dent. Res. **96**, 1339–1345 (2017).
[47]M. J. Schnieders *et al.*, "A novel mutation (LEU396ARG) in OPA1 is associated with a severe phenotype in a large dominant optic atrophy pedigree," Eye **32**, 843–845 (2018).
[48]E. A. Boese *et al.*, "Novel intragenic PAX6 deletion in a pedigree with aniridia, morbid obesity, and diabetes," Curr. Eye Res. **45**, 91–96 (2020).
[49]J. Hagedorn *et al.*, "Nanophthalmos patient with a THR518MET mutation in MYRF, a case report," BMC Ophthalmol. **20**, 388 (2020).
[50]J. Bi *et al.*, "Characterization of a TP53 somatic variant of unknown function from an ovarian cancer patient using organoid culture and computational modeling," Clin. Obstet. Gynecol. **63**, 109–119 (2020).
[51]B. E. Hinz *et al.*, "In silico and in vivo analysis of amino acid substitutions that cause laminopathies," Int. J. Mol. Sci. **22**, 11226 (2021).
[52]W. Awotoye *et al.*, "Whole-genome sequencing reveals de-novo mutations associated with nonsyndromic cleft lip/palate," Sci. Rep. **12**, 11743 (2022).
[53]E. A. Boese *et al.*, "GJA3 genetic variation and autosomal dominant congenital cataracts and glaucoma following cataract surgery," JAMA Ophthalmol. **141**, 872 (2023).
[54]A. J. Nessler, O. Okada, M. J. Hermon, H. Nagata, and M. J. Schnieders, "Progressive alignment of crystals: Reproducible and efficient assessment of crystal structure similarity," J. Appl. Crystallogr. **55**, 1528–1537 (2022).
[55]A. J. Nessler *et al.*, "Crystal polymorph search in the NPT ensemble via a deposition/sublimation alchemical path," Cryst. Growth Des. **24**, 3205–3217 (2024).
[56]C. R. Groom, I. J. Bruno, M. P. Lightfoot, and S. C. Ward, "The Cambridge structural Database," Acta Crystallogr., Sect. B: Struct. Sci., Cryst. Eng. Mater. **72**, 171–179 (2016).
[57]A. Kaminsky, in *IEEE International Parallel and Distributed Processing Symposium* (IEEE, 2007).
[58]A. Kaminsky, *Building Parallel Programs: SMPs, Clusters and Java* (Course Technology Press, 2009).
[59]J. W. Cooley and J. W. Tukey, "An algorithm for the machine calculation of complex Fourier series," Math. Comput. **19**, 297–301 (1965).
[60]C. Temperton, "Self-sorting mixed-radix fast Fourier transforms," J. Comput. Phys. **52**, 1–23 (1983).
[61]M. Challacombe, E. Schwegler, and J. Almlöf, *Computational Chemistry: Reviews of Current Trends*, *Computational Chemistry: Reviews of Current Trends Vol. 1* (World Scientific, 1996), pp. 53–107.
[62]A. C. Simmonett, F. C. Pickard, H. F. Schaefer, and B. R. Brooks, "An efficient algorithm for multipole energies and derivatives based on spherical harmonics and extensions to particle mesh Ewald," J. Chem. Phys. **140**, 184101 (2014).
[63]C. G. Broyden, "The convergence of a class of double-rank minimization algorithms 1. General considerations," IMA J. Appl. Math. **6**, 76–90 (1970).
[64]R. Fletcher, "A new approach to variable metric algorithms," Comput. J. **13**, 317–322 (1970).
[65]D. Goldfarb, "A family of variable-metric methods derived by variational means," Math. Comput. **24**, 23–26 (1970).
[66]D. F. Shanno, "Conditioning of quasi-Newton methods for function minimization," Math. Comput. **24**, 647–656 (1970).
[67]C. de Boor, *A Practical Guide to Splines* (Springer, 2001).
[68]K. Cowtan, "Generic representation and evaluation of properties as a function of position in reciprocal space," J. Appl. Crystallogr. **35**, 655–663 (2002).
[69]A. Paszke *et al.*, in *Proceedings of the 33rd International Conference on Neural Information Processing Systems* (Curran Associates, Inc., 2019), Article 721.
[70]D. Beeman, "Some multistep methods for use in molecular dynamics calculations," J. Comput. Phys. **20**, 130–139 (1976).
[71]M. P. Allen, "Brownian dynamics simulation of a chemical reaction in solution," Mol. Phys. **40**, 1073–1087 (1980).
[72]F. Guarnieri and W. C. Still, "A rapidly convergent simulation method: Mixed Monte Carlo/stochastic dynamics," J. Comput. Chem. **15**, 1302–1310 (1994).
[73]D. D. Humphreys, R. A. Friesner, and B. J. Berne, "A multiple-time-step molecular dynamics algorithm for macromolecules," J. Phys. Chem. **98**, 6885–6892 (1994).
[74]X. Qian and T. Schlick, "Efficient multiple-time-step integrators with distance-based force splitting for particle-mesh-Ewald molecular dynamics simulations," J. Chem. Phys. **116**, 5971–5983 (2002).
[75]H. J. C. Berendsen, J. P. M. Postma, W. F. van Gunsteren, A. DiNola, and J. R. Haak, "Molecular dynamics with coupling to an external bath," J. Chem. Phys. **81**, 3684–3690 (1984).
[76]G. Bussi, T. Zykova-Timan, and M. Parrinello, "Isothermal-isobaric molecular dynamics using stochastic velocity rescaling," J. Chem. Phys. **130**, 074101 (2009).
[77]D. Frenkel and B. Smit, *Understanding Molecular Simulation*, 2nd ed. (Academic Press, 2002), pp. 1–638.
[78]C. H. Bennett, "Efficient estimation of free energy differences from Monte Carlo data," J. Comput. Phys. **22**, 245–268 (1976).
[79]S. Lee, M. Chen, W. Yang, and N. G. J. Richards, "Sampling long time scale protein motions: OSRW simulation of active site loop conformational free energies in formyl-CoA:oxalate CoA transferase," J. Am. Chem. Soc. **132**, 7252–7253 (2010).
[80]J. Park *et al.*, "Absolute organic crystal thermodynamics: Growth of the asymmetric unit into a crystal via alchemy," J. Chem. Theory Comput. **10**, 2781–2791 (2014).
[81]D. Sehnal *et al.*, "BinaryCIF and CIFTools—Lightweight, efficient and extensible macromolecular data management," PLoS Comput. Biol. **16**, e1008247 (2020).

[82]A. Lafita *et al.*, "BioJava 5: A community driven open-source bioinformatics library," PLoS Comput. Biol. **15**, e1006791 (2019).
[83]C. Steinbeck *et al.*, "Recent developments of the chemistry development kit (CDK)—An open-source java library for chemo- and bioinformatics," Curr. Pharm. Des. **12**, 2111–2120 (2006).
[84]J. Fumero *et al.*, in *Proceedings of the 15th ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments* (Association for Computing Machinery, 2019).
[85]See https://picocli.info for Picocli—A mighty tiny command line interface, 2024.
[86]W. L. Jorgensen, D. S. Maxwell, and J. Tirado-Rives, "Development and testing of the OPLS all-atom force field on conformational energetics and properties of organic liquids," J. Am. Chem. Soc. **118**, 11225–11236 (1996).
[87]G. A. Kaminski, R. A. Friesner, J. Tirado-Rives, and W. L. Jorgensen, "Evaluation and reparametrization of the OPLS-AA force field for proteins via comparison with accurate quantum chemical calculations on peptides," J. Phys. Chem. B **105**, 6474–6487 (2001).
[88]W. D. Cornell *et al.*, "A second generation force field for the simulation of proteins, nucleic acids, and organic molecules J. Am. Chem. Soc. 1995, 117, 5179–5197," J. Am. Chem. Soc. **118**, 2309 (1996).
[89]P. Kollman *et al.*, in *Computer Simulation of Biomolecular Systems: Theoretical and Experimental Applications*, edited by W. F. van Gunsteren, P. K. Weiner, and A. J. Wilkinson (Springer, Netherlands, 1997), pp. 83–96.
[90]J. Wang, P. Cieplak, and P. A. Kollman, "How well does a restrained electrostatic potential (RESP) model perform in calculating conformational energies of organic and biological molecules?," J. Comput. Chem. **21**, 1049–1074 (2000).
[91]V. Hornak *et al.*, "Comparison of multiple Amber force fields and development of improved protein backbone parameters," Proteins: Struct., Funct., Bioinf. **65**, 712–725 (2006).
[92]A. D. MacKerell *et al.*, "All-atom empirical potential for molecular modeling and dynamics studies of proteins," J. Phys. Chem. B **102**, 3586–3616 (1998).
[93]N. Foloppe and A. D. MacKerell, Jr., "All-atom empirical force field for nucleic acids: I. Parameter optimization based on small molecule and condensed phase macromolecular target data," J. Comput. Chem. **21**, 86–104 (2000).
[94]A. D. Mackerell, Jr., M. Feig, and C. L. Brooks III, "Extending the treatment of backbone energetics in protein force fields: Limitations of gas-phase quantum mechanics in reproducing protein conformational distributions in molecular dynamics simulations," J. Comput. Chem. **25**, 1400–1415 (2004).
[95]W. C. Still, A. Tempczyk, R. C. Hawley, and T. Hendrickson, "Semianalytical treatment of solvation for molecular mechanics and dynamics," J. Am. Chem. Soc. **112**, 6127–6129 (1990).
[96]D. Qiu, P. S. Shenkin, F. P. Hollinger, and W. C. Still, "The GB/SA continuum model for solvation. A fast analytical method for the calculation of approximate Born radii," J. Phys. Chem. A **101**, 3005–3014 (1997).
[97]A. Onufriev, D. A. Case, and D. Bashford, "Effective Born radii in the generalized Born approximation: The importance of being perfect," J. Comput. Chem. **23**, 1297–1304 (2002).
[98]G. D. Hawkins, C. J. Cramer, and D. G. Truhlar, "Pairwise solute descreening of solute charges from a dielectric medium," Chem. Phys. Lett. **246**, 122–129 (1995).
[99]T. Grycuk, "Deficiency of the Coulomb-field approximation in the generalized Born model: An improved formula for Born radii evaluation," J. Chem. Phys. **119**, 4817–4826 (2003).
[100]J. Mongan, C. Simmerling, J. A. McCammon, D. A. Case, and A. Onufriev, "Generalized Born model with a simple, robust molecular volume correction," J. Chem. Theory Comput. **3**, 156–169 (2007).
[101]A. Onufriev, D. Bashford, and D. A. Case, "Exploring protein native states and large-scale conformational changes with a modified generalized Born model," Proteins: Struct., Funct., Bioinf. **55**, 383–394 (2004).
[102]P. S. Hudson *et al.*, "Obtaining QM/MM binding free energies in the SAMPL8 drugs of abuse challenge: Indirect approaches," J. Comput. Aided Mol. Des. **36**, 263–277 (2022).
[103]J. Nocedal, "Updating quasi-Newton matrices with limited storage," Math. Comput. **35**, 773–782 (1980).
[104]D. C. Liu and J. Nocedal, "On the limited memory BFGS method for large scale optimization," Math. Program. **45**, 503–528 (1989).
[105]S. Kirkpatrick, C. D. Gelatt, Jr., and M. P. Vecchi, "Optimization by simulated annealing," Science **220**, 671–680 (1983).
[106]A. Brnger, "Simulated annealing in crystallography," Annu. Rev. Phys. Chem. **42**, 197–223 (1991).
[107]P. P. Ewald, "Die Berechnung optischer und elektrostatischer Gitterpotentiale," Ann. Phys. **369**, 253–287 (1921).
[108]M. J. Schnieders, T. D. Fenn, V. S. Pande, and A. T. Brunger, "Polarizable atomic multipole X-ray refinement: Application to peptide crystals," Acta Crystallogr., Sect. D: Biol. Crystallogr. **65**, 952–965 (2009).
[109]F. Lipparini *et al.*, "Scalable evaluation of polarization energy and associated forces in polarizable molecular dynamics: I. Toward massively parallel direct space computations," J. Chem. Theory Comput. **10**, 1638–1651 (2014).
[110]L. Verlet, "Computer 'experiments' on classical fluids. I. Thermodynamical properties of Lennard-Jones molecules," Phys. Rev. **159**, 98 (1967).
[111]W. C. Swope, H. C. Andersen, P. H. Berens, and K. R. Wilson, "A computer simulation method for the calculation of equilibrium constants for the formation of physical clusters of molecules: Application to small water clusters," J. Chem. Phys. **76**, 637–649 (1982).
[112]M. Tuckerman, B. J. Berne, and G. J. Martyna, "Reversible multiple time scale molecular dynamics," J. Chem. Phys. **97**, 1990–2001 (1992).
[113]G. Bussi and M. Parrinello, "Stochastic thermostats: Comparison of local and global schemes," Comput. Phys. Commun. **179**, 26–29 (2008).
[114]S. Yun-yu, W. Lu, and W. F. Van Gunsteren, "On the approximation of solvent effects on the conformation and dynamics of cyclosporin a by stochastic dynamics simulation techniques," Mol. Simul. **1**, 369–383 (1988).
[115]D. Frenkel and B. Smit, *Understanding Molecular Simulation: From Algorithms to Applications* (Academic Press, San Diego, 2002).
[116]M. S. Lee, F. R. Salsbury, Jr., and C. L. Brooks, "Constant-pH molecular dynamics using continuous titration coordinates," Proteins: Struct., Funct., Bioinf. **56**, 738–752 (2004).
[117]J. Khandogin and C. L. Brooks III, "Constant pH molecular dynamics with proton tautomerism," Biophys. J. **89**, 141–157 (2005).
[118]J. A. Wallace and J. K. Shen, "Continuous constant pH molecular dynamics in explicit solvent with pH-based replica exchange," J. Chem. Theory Comput. **7**, 2617–2629 (2011).
[119]M. R. Shirts and J. D. Chodera, "Statistically optimal analysis of samples from multiple equilibrium states," J. Chem. Phys. **129**, 124105 (2008).
[120]B. Walker, C. Liu, E. Wait, and P. Ren, "Automation of AMOEBA polarizable force field for small molecules: Poltype 2," J. Comput. Chem. **43**, 1530–1542 (2022).
[121]A. M. Reilly *et al.*, "Report on the sixth blind test of organic crystal structure prediction methods," Acta Crystallogr., Sect. B: Struct. Sci., Cryst. Eng. Mater. **72**, 439–459 (2016).
[122]A. Barducci, M. Bonomi, and M. Parrinello, "Metadynamics," Wiley Interdiscip. Rev.: Comput. Mol. Sci. **1**, 826–843 (2011).
[123]L. Zheng and W. Yang, "Practically efficient and robust free energy calculations: Double-integration orthogonal space tempering," J. Chem. Theory Comput. **8**, 810–823 (2012).
[124]J. F. Dama, G. Rotskoff, M. Parrinello, and G. A. Voth, "Transition-tempered metadynamics: Robust, convergent metadynamics via on-the-fly transition barrier estimation," J. Chem. Theory Comput. **10**, 3626–3633 (2014).
[125]B. Zhang, D. Kilburg, P. Eastman, V. S. Pande, and E. Gallicchio, "Efficient Gaussian density formulation of volume and surface areas of macromolecules on graphical processing units," J. Comput. Chem. **38**, 740–752 (2017).
[126]M. L. Connolly, "Solvent-accessible surfaces of proteins and nucleic acids," Science **221**, 709–713 (1983).
[127]N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, "Equation of state calculations by fast computing machines," J. Chem. Phys. **21**, 1087–1092 (1953).
[128]W. K. Hastings, "Monte Carlo sampling methods using Markov chains and their applications," Biometrika **57**, 97–109 (1970).
[129]R. Zhou, "Replica exchange molecular dynamics method for protein folding simulation," Methods Mol. Biol. **350**, 205–223 (2007).

[130]S. C. Lovell, J. M. Word, J. S. Richardson, and D. C. Richardson, "The penultimate rotamer library," Proteins: Struct., Funct., Bioinf. **40**, 389–408 (2000).
[131]J. W. Ponder and F. M. Richards, "Tertiary templates for proteins: Use of packing criteria in the enumeration of allowed sequences for different structural classes," J. Mol. Biol. **193**, 775–791 (1987).
[132]J. Jumper *et al.*, "Highly accurate protein structure prediction with AlphaFold," Nature **596**, 583–589 (2021).
[133]H. Azaiez *et al.*, "Genomic landscape and mutational signatures of deafness-associated genes," Am. J. Hum. Genet. **103**, 484–497 (2018).
[134]P. Eastman *et al.*, "OpenMM 8: Molecular dynamics simulation with machine learning potentials," J. Phys. Chem. B **128**, 109–116 (2024).