Accelerating the 3D-RISM theory of molecular solvation with treecode summation and cut-offs

Leighton Wilson*, Robert Krasny†, Tyler Luchko‡
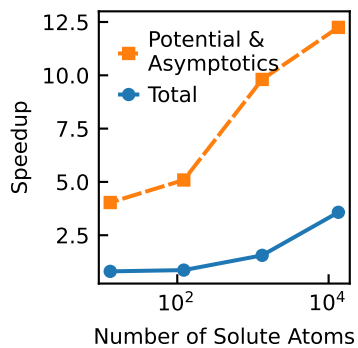
April 27, 2022

■

Abstract

The 3D reference interaction site model (3D-RISM) of molecular solvation is a powerful tool for computing the equilibrium thermodynamics and density distributions of solvents, such as water and co-ions, around solute molecules. However, 3D-RISM solutions can be expensive to calculate, especially for proteins and other large molecules where calculating the potential energy between solute and solvent requires more than half the computation time. To address this problem, we have developed and implemented treecode summation for long-range interactions and analytically corrected cut-offs for short-range interactions to accelerate the potential energy and long-range asymptotics calculations in non-periodic 3D-RISM in the AmberTools molecular modeling suite. For the largest single protein considered in this work, tubulin, the total computation time was reduced by a factor of 4. In addition, parallel calculations with these new methods scale almost linearly and the iterative solver remains the largest impediment to parallel scaling. To demonstrate the utility of our approach for large systems, we used 3D-RISM to calculate the solvation thermodynamics and density distribution of 7-ring microtubule, consisting of 910 tubulin dimers, over 1.2 million atoms.

---

*Department of Mathematics, University of Michigan, Ann Arbor, MI 48109
†Department of Mathematics, University of Michigan, Ann Arbor, MI 48109
‡Department of Physics and Astronomy, California State University, Northridge, Los Angeles, CA 91330

The 3D reference interaction site model (3D-RISM) of molecular solvation computes equilibrium thermodynamics and density distributions of solvents around solute molecules. To accelerate 3D-RISM for large biomolecules, we have developed treecode summation and analytically corrected cut-offs for long- and short-range interactions. This reduces the over all computation time by a factor of 4 for the protein tubulin and allowed the calculation of solvation thermodynamics for a 1.2 million atom microtubule.
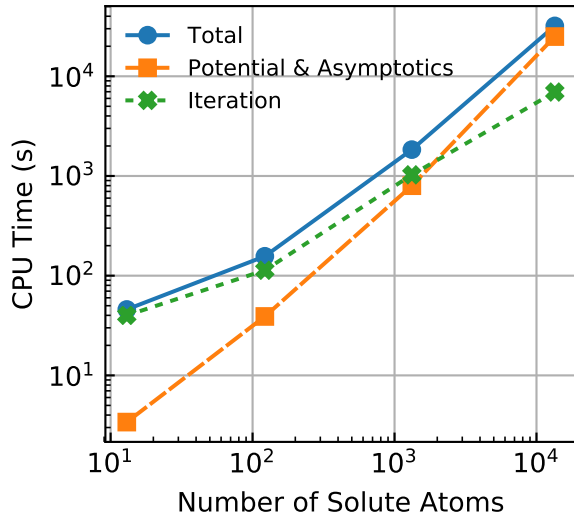
Figure 1: Time required for a single 3D-RISM calculation converged to a tolerance of 10^{-6} employing direct summation. At approximately 2000 solute atoms, the time to initialize the calculation (potential and asymptotics) becomes larger than the time required to iterate to a solution.

1   INTRODUCTION

Solvation thermodynamics and the structure of the surrounding liquid play an important role in determining the properties and interactions of molecular systems in solution. While explicit solvent approaches are commonly used, they can be computationally expensive and require elaborate protocols to calculate different physical properties where solvation is involved, such as solvation free energies[1], preferential interaction parameters[2], and binding free energy[3]. Various implicit solvent methods have been developed to simplify and accelerate the treatment of solvent. Among the most promising are integral equation theories, based on the Ornstein-Zernike equation[4], and closely related classical density functional theories[5–7] as they are complete theories, calculating approximate equilibrium distributions of explicit models, from which all solvation thermodynamics can be computed. The 3D-reference interaction site model of molecular solvation (3D-RISM)[8,9] is one such integral equation, which has been coupled with classical and quantum mechanics modeling software[10–14] and shown to provide solvation thermodynamics in good agreement with experiment and explicit solvent

calculations[15–18].

However, 3D-RISM can be computationally expensive, especially for large molecules (see Figure 1). 3D-RISM calculations consist of three sequential steps: initialization (calculating potential energy and long-range electrostatic interactions on a 3D grid), iteration to convergence, and integration of the solvent distribution to calculate thermodynamics. For small molecules, iteration time dominates the calculation, which scales with the number of grid points, N_{grid}, as $O\left(N_{\mathrm{grid}} \log N_{\mathrm{grid}}\right)$. Initialization time dominates for typical proteins, scaling with both the number of solute atoms, N_{atom}, and grid points as $O\left(N_{\mathrm{atom}} N_{\mathrm{grid}}\right)$. Integrating solvent thermodynamics is typically 1% or less of the total computation time. Depending on the precision of the calculation, initialization becomes the most expensive part of the calculation for solutes of 1000 atoms or more and is a major barrier to the practical application of 3D-RISM to large molecules.

Limited work has been done to address the computational cost of initialization. For the case of periodic boundary conditions, Heil and Kast[19] developed the "no real-space supercell" (NRS) method. The NRS method computes all long-range Coulomb interactions in reciprocal space, where they could be efficiently handled, scaling as $O\left(N_{\mathrm{grid}} \log N_{\mathrm{grid}}\right)$ and requiring only a small fraction of the total calculation time. This cannot be used for open boundary conditions, where the "supercell" method is used[9,20,21]. Because there is no periodic structure, the entire potential energy is calculated for a real-space grid. In addition, to capture contributions beyond the size of the solvent box, analytic long-range asymptotic (LRA) expressions of the solvent correlation functions must also be computed in real- and reciprocal-space. So far, little has been done to address the cost of computing these expressions.

In this paper, we will focus on the open boundary case and address the cost of computing potential energy and LRA functions. The potential energy is composed of Coulomb and Lennard-Jones contributions, while the LRA is a modified Coulomb potential. The long-range and short-range components of these calculations require different approaches. For short range calculations, cut-offs are often appropriate. Lennard-Jones calculations are one such case and analytic corrections are well known[22]. Long-range calculations require a different approach.

A promising approach for dealing with long-range Coulomb-like potentials is treecode summation. Originally developed for use in gravitational N-body simulations[23], in which the force of N point masses on each other must be computed at each time step of a dynamics simulation, they can be applied to a wide variety of problems involving the interaction between N source particles on M target sites which may or may not be coincident with the sources. For the case of coincident sources and targets, direct summation is $O(N^2)$, while treecode methods are $O(N \log N)$. The essential idea behind these methods is the replacement of particle-particle interactions with suitably chosen particle-cluster interactions, which can be computed approximately when the cluster and particle are well-separated. This requires the construction of a hierarchical tree of particle clusters and a criterion for determining when a particle and a cluster are well-separated.

The simplest interaction between a particle and cluster is a monopole approximation, in which, for a well-separated interaction, the cluster is replaced with a single particle at the center whose charge (or mass, or other property) is the sum of all particles contained within the cluster. This is the strategy of the original gravitational Barnes–Hut treecode[23]. The fast multipole method improved the accuracy by using higher order multipole expansions in terms of spherical harmonics[24]. Alternatively, the interaction between a particle and a cluster can be approximated with Cartesian Taylor expansions. However, since hard coded Taylor expansions to high order can be costly, an alternative approach using recurrence relations has been developed[25–27].

In the evaluation of LRA functions in 3D-RISM, the solute is represented by N source particles and the solvent grid by M target sites. Traditional particle-cluster treecodes build a tree on the source particles, with a computational cost that scales as $O(M \log N)$. For $M \gg N$, as is typically the case for the 3D-RISM solvent grid, these methods scale poorly. In this case, it is advantageous to consider an alternative cluster-particle treecode in which the tree is built on the targets, with a computational cost that scales as $O(N \log M)$[28]. The present work demonstrates that cluster-particle treecode summation with Taylor series recurrence relations provides an effective approach for evaluating the Coulomb potential and the Coulomb-like LRA functions within 3D-RISM.

In this paper, we detail the theoretical background of cluster-particle treecode summa-

tion and our approach to cut-offs and test our implementation in the *AmberTools* molecular modeling suite[29]. Section 2.1 and 2.2 provide a brief overview of 3D-RISM theory and potential energy and LRA functions. In Section 2.3 we detail the cluster-particle treecode method and its application to Coulomb and screened Coulomb electrostatic potentials. In Section 2.4 we extend the application of cluster-particle treecode methods to the LRA potentials. Our error-tolerance approach to cutoffs is presented in Section 2.5 and 2.6. We provide details of our benchmarking protocol in Section 3. Finally, in Section 4 we present the results for serial benchmarking, along with practical guidelines for selecting parameters and results for scaling with system size and over parallel processes and application to computing the solvation properties of microtubules.

2 THEORY

2.1 3D-RISM

As detailed descriptions of 3D-RISM can be found elsewhere[30–32], we will briefly review the theory, highlighting the parts where Coulomb interactions are involved. The 3D-RISM equation is given by

$$h_\gamma \left(\mathbf{r} \right) = \sum_\alpha \int c_\alpha \left(\mathbf{r} - \mathbf{r}^{'} \right) \chi_{\alpha\gamma} \left(r^{'} \right) d\mathbf{r}^{'}, \tag{1}$$

where α and γ indicate solvent sites (e.g., oxygen or hydrogen in the case of water), and $\mathbf{r}$ is a grid point location. Both $h_\gamma \left(\mathbf{r} \right)$, the total correlation function (TCF), and $c_\gamma \left(\mathbf{r} \right)$, the direct correlation function (DCF), are unknown quantities to be solved for and are represented on 3D grids. $\chi_{\alpha\gamma} \left(r \right)$ is the solvent site-site susceptibility, which is computed in advance, typically using extended RISM (XRISM)[33] or dielectrically consistent RISM (DRISM)[34]. To compute the convolution integral in equation (1), it is useful to express the 3D-RISM equation in reciprocal space,

$$\hat{h}_\gamma \left(\mathbf{k} \right) = \sum_\alpha \hat{c}_\gamma \left(\mathbf{k} \right) \hat{\chi}_{\alpha\gamma} \left(k \right), \tag{2}$$

where $\mathbf{k}$ is the wave vector in reciprocal space and $\hat{x} \left(\mathbf{k} \right)$ represents the Fourier transform of $x \left(\mathbf{r} \right)$.

As both the TCF and DCF are unknown, we require a closure relation to solve for the TCF and DCF in equation (1). In this work we use the Kovalenko-Hirata closure (KH)[20],

$$h_\gamma (\mathbf{r}) = \begin{cases} \exp\left(d_\gamma (\mathbf{r})\right) + 1 & d_\gamma (\mathbf{r}) \leq 0 \\ d_\gamma (\mathbf{r}) + 2 & d_\gamma (\mathbf{r}) > 0, \end{cases}$$

$$d_\gamma (\mathbf{r}) = -\beta u_\gamma (\mathbf{r}) + h_\gamma (\mathbf{r}) - c_\gamma (\mathbf{r}), \tag{3}$$

where $u_\gamma (\mathbf{r})$ is the potential energy between solvent site γ and the entire solute and $\beta = 1/k_B T$ where k_B is the Boltzmann constant and T is the temperature. Though other closures can provide better agreement with explicit solvent simulations, we use the KH closure in this work because it provides reliable numerical convergence of the 3D-RISM equations, allowing us to obtain solutions for all the treecode summation parameters explored. As the potential energy, which includes Coulomb and Lennard-Jones interactions, and long-range asymptotics (see Section 2.2) are computed in advance on 3D grids with the same dimension as that of the TCF and DCF, the choice of closure has no bearing on the direct or treecode summation results.

Once the TCF and DCF have been solved for, it is possible to compute thermodynamic properties of the solvent. The most commonly used is the excess chemical potential of the solute

$$\mu_{\text{ex}}^{\text{KH}} = k_B T \sum_\gamma \rho_\gamma \times \int d\mathbf{r} \left[\frac{1}{2}(h_\gamma(\mathbf{r}))^2 \Theta\left(-h_\gamma (\mathbf{r})\right) - \frac{1}{2}h_\gamma(\mathbf{r})c_\gamma(\mathbf{r}) - c_\gamma(\mathbf{r}) \right] \tag{4}$$

where Θ is the Heaviside function and ρ_γ is the bulk density of site γ. As 3D-RISM is treating a single solute at infinite dilution, the excess chemical potential is also the solvation free energy (SFE). Because the SFE is of general interest, we will use it to quantify the accuracy of our numerical methods and parameter choices.

2.2 Long-range asymptotics

The long-range asymptotic behavior of both the TCF and DCF are affected by Coulomb interactions introduced in the potential energy. In particular, the long-range asymptotics must be explicitly handled during the forward and backward Fourier transforms used to

7

compute the convolution integral in equation (2). Several approaches have been developed to handle this[9,19,20,32,35,36]. Since we employ open boundary conditions, we use the approach originated by Ng[37] and Springer[38] and extended for ionic solutions in 3D-RISM by Kovalenko and co-workers[9,21,36]. For any system where both solute and solvent have partial charges on atomic sites, the long-range asymptotics of the DCF are given by

$$c_\gamma^{(\text{lr})}(\mathbf{r}) = -\beta \sum_a \frac{Q_a^{\text{U}} q_\gamma}{|\mathbf{r} - \mathbf{R}_a|} \operatorname{erf}\left(\frac{|\mathbf{r} - \mathbf{R}_a|}{\eta}\right) \tag{5}$$

and

$$c_\gamma^{(\text{lr})}(\mathbf{k}) = -\frac{q_\gamma 4\pi\beta}{k^2} \exp\left(-\frac{k^2\eta^2}{4}\right) \times \sum_a Q_a^{\text{U}} \exp\left(i\mathbf{k} \cdot \mathbf{R}_a\right) \tag{6}$$

where a is the solute site with position $\mathbf{R}_a$ and partial charge Q_a^U, $\beta = \frac{1}{k_b T}$, k_b is the Boltzmann constant, T is temperature, ϵ is the dielectric constant, η is a charge smearing parameter, and q_γ is the partial charge on solvent site γ. Both equations are computed on a 3D grid, as with the potential energy. Prior to transforming the DCF into reciprocal space equation (5) is subtracted off. After the forward Fourier transform has been performed, equation (6) is added back to the DCF in reciprocal space and $\hat{h}_\gamma(\mathbf{k})$ is computed from equation (2).

In the case that the solvent also contains ionic species, such as Na$^+$ or Cl$^-$, it is also necessary to treat the long-range asymptotics of the TCF, given by

$$h_j^{(\text{lr})}(\mathbf{r}) = -\frac{\beta}{2\epsilon} \sum_a \frac{Q_a^{\text{U}} q_j}{|\mathbf{r} - \mathbf{R}_a|} \exp\left(\frac{\kappa_D^2 \eta^2}{4}\right) \left[e^{(-\kappa_D |\mathbf{r} - \mathbf{R}_a|)} \operatorname{erfc}\left(\frac{\kappa_D \eta}{2} - \frac{|\mathbf{r} - \mathbf{R}_a|}{\eta}\right) \right.$$
$$\left. - e^{(\kappa_D |\mathbf{r} - \mathbf{R}_a|)} \operatorname{erfc}\left(\frac{\kappa_D \eta}{2} + \frac{|\mathbf{r} - \mathbf{R}_a|}{\eta}\right) \right] \tag{7}$$

and

$$h_j^{(\text{lr})}(\mathbf{k}) = -\frac{q_j 4\pi\beta}{\epsilon\left(k^2 + \kappa_D^2\right)} \exp\left(-\frac{k^2\eta^2}{4}\right) \times \sum_a Q_a^{\text{U}} \exp\left(i\mathbf{k} \cdot \mathbf{R}_a\right) \tag{8}$$

where κ_D is the contribution to the inverse Debye length of ionic co-solvent species j with net charge q_j. After $\hat{h}_\gamma(\mathbf{k})$ has been computed with equation (2), equation (8) is subtracted off. Then equation (7) is added to the TCF after the backward Fourier transform has been applied. Equations (5) and (7) are also used when calculating thermodynamic observables to include the long-range contributions not captured in the finite 3D grids used to represent

the TCF and DCF. For example, equation (4) becomes

$$
\begin{aligned}
\mu_{\text{ex}}^{\text{KH}} = k_B T \sum_{\gamma} \rho_{\gamma} \Bigg\{ & \int_{\text{grid}} d\mathbf{r} \left[\frac{1}{2}(h_{\gamma}(\mathbf{r}))^2 \Theta\left(-h_{\gamma}(\mathbf{r})\right) - \frac{1}{2}(h_{\gamma}^{(\text{lr})}(\mathbf{r}))^2 \Theta\left(Q^{\text{U}} q_j\right) \right. \\
& \left. - \frac{1}{2} h_{\gamma}(\mathbf{r}) c_{\gamma}(\mathbf{r}) + \frac{1}{2} h_{\gamma}^{(\text{lr})}(\mathbf{r}) c_{\gamma}^{(\text{lr})}(\mathbf{r}) - c_{\gamma}(\mathbf{r}) \right] \\
& + \int d\mathbf{r} \left[\frac{1}{2}(h_{\gamma}^{(\text{lr})}(\mathbf{r}))^2 \Theta\left(Q^{\text{U}} q_j\right) - \frac{1}{2} h_{\gamma}^{(\text{lr})}(\mathbf{r}) c_{\gamma}^{(\text{lr})}(\mathbf{r}) \right] \Bigg\}. \quad (9)
\end{aligned}
$$

Note that the first two integrals are over the volume of the grid while the last integral is over all space and can be numerically computed via 1D integrals[21].

2.3 Cluster-Particle Treecode for Electrostatic Interactions

Consider a general potential function $\phi(\mathbf{x}, \mathbf{y})$, and a collection of M target sites $\mathbf{x}_i$ and N disjoint source particles $\mathbf{y}_j$ with charges q_j. Then the potential at a target site $\mathbf{x}_i$ is

$$
V(\mathbf{x}_i) = \sum_{j=1}^{N} q_j \phi(\mathbf{x}_i, \mathbf{y}_j), \quad (10)
$$

In 3D-RISM such expressions arise in computing the electrostatic potential and the LRA functions, where the target sites lie on a regular grid, and the source particles are the atomic solute sites, with $M \gg N$. The cost of evaluating these expressions by direct summation is $O(MN)$. In this section we describe the cluster-particle treecode we utilize to reduce the cost to $O((M+N)\log M)$[28].

2.3.1 Cluster-particle treecode algorithm

The treecode starts by building a hierarchical tree of clusters on the M target locations. The root cluster is the smallest rectangular box containing the targets. The root is divided along all Cartesian directions for which the side length of the root parallel to that direction is within $\sqrt{2}$ of the shortest side length; this yields 8, 4, or 2 child clusters. The child clusters are similarly divided until the cluster contains less than N_0 targets, a user-specified parameter. The tree has L levels, where level 1 is the root cluster and level L contains the leaves. Each target site $\mathbf{x}_i$ belongs to a nested set of clusters $\mathbf{x}_i \in C_L \subseteq ... \subseteq C_1$, where

9

cluster C_ℓ is at level ℓ. Let I_ℓ denote the set of source particles $\mathbf{y}_j$ with charge q_j that are well-separated from cluster C_ℓ but not from clusters $C_{\ell-1}, ..., C_1$, and let D denote the set of source particles $\mathbf{y}_j$ with charge q_j that are not well-separated from any cluster containing $\mathbf{x}_i$. Then equation (10) can be rewritten as

$$V(\mathbf{x}_i) = \sum_{\mathbf{y}_j \in D} q_j \phi(\mathbf{x}_i, \mathbf{y}_j) + \sum_{\ell=1}^{L} \sum_{\mathbf{y}_j \in I_\ell} q_j \phi(\mathbf{x}_i, \mathbf{y}_j), \tag{11}$$

where the first term on the right hand side accounts for sources that are close to $\mathbf{x}_i$ and the second term accounts for sources that are well-separated from $\mathbf{x}_i$. The first term is computed by direct summation, and the second term is computed by Taylor approximation. Expanding the second term $\phi(\mathbf{x}_i, \mathbf{y}_j)$ about $\mathbf{x}_c^\ell$, the center of cluster ℓ, gives

$$\sum_{\mathbf{y}_j \in I_\ell} q_j \phi(\mathbf{x}_i, \mathbf{y}_j) \approx \sum_{\mathbf{y}_j \in I_\ell} q_j \sum_{\|\mathbf{k}\|=0}^{p} \frac{1}{\mathbf{k}!} \partial_{\mathbf{x}}^{\mathbf{k}} \phi\left(\mathbf{x}_c^\ell, \mathbf{y}_j\right) \left(\mathbf{x}_i - \mathbf{x}_c^\ell\right)^{\mathbf{k}}$$
$$= \sum_{\|\mathbf{k}\|=0}^{p} m_{\mathbf{k}}\left(\mathbf{x}_c^\ell\right) \left(\mathbf{x}_i - \mathbf{x}_c^\ell\right)^{\mathbf{k}}, \tag{12}$$

where p is the order of the approximation, the coefficients $m_{\mathbf{k}}$ are

$$m_{\mathbf{k}}\left(\mathbf{x}_c^\ell\right) = \sum_{\mathbf{y}_j \in I_\ell} q_j (-1)^{\|\mathbf{k}\|} a_{\mathbf{k}}\left(\mathbf{x}_c^\ell, \mathbf{y}_j\right), \tag{13}$$

and the Taylor coefficients $a_{\mathbf{k}}$ are

$$a_{\mathbf{k}}\left(\mathbf{x}_c^\ell, \mathbf{y}_j\right) = \frac{1}{\mathbf{k}!} \partial_{\mathbf{y}}^{\mathbf{k}} \phi\left(\mathbf{x}_c^\ell, \mathbf{y}_j\right). \tag{14}$$

Note that equation (12) is a Taylor polynomial in three dimensions, where $\|\mathbf{k}\| = k_1 + k_2 + k_3$, $\mathbf{k}! = k_1! k_2! k_3!$, $\partial_{\mathbf{y}}^{\mathbf{k}} = \partial_{y_1}^{k_1} \partial_{y_2}^{k_2} \partial_{y_3}^{k_3}$, $(\mathbf{x}_i - \mathbf{x}_c)^{\mathbf{k}} = (x_{i1} - x_{c1})^{k_1} (x_{i2} - x_{c2})^{k_2} (x_{i3} - x_{c3})^{k_3}$, and $1, 2, 3$ denote the three respective Cartesian directions. Also note that the criterion for a target site and source cluster being well-separated is $r/R < \theta$, where r is the cluster radius, $R = |x - \mathbf{y}_j|$ is the distance between the target cluster and source cluster, and θ is the user-specified MAC parameter[23].

2.3.2 Recurrence relations for Taylor coefficients of Coulomb potential

We illustrate this approach using the example of the Coulomb potential,

$$\phi(\mathbf{x}_i, \mathbf{y}_j) = \frac{1}{|\mathbf{x}_i - \mathbf{y}_j|}. \tag{15}$$

In this case the Taylor coefficients in equation (14) can be calculated efficiently using the recurrence relation[25,27],

$$a_{\mathbf{k}}(\mathbf{x}, \mathbf{y}) = \frac{1}{|\mathbf{x} - \mathbf{y}|^2} \left[\left(2 - \frac{1}{\|\mathbf{k}\|} \right) \sum_{i=1}^{3} (x_i - y_i) a_{\mathbf{k} - \mathbf{e}_i} - \left(1 - \frac{1}{\|\mathbf{k}\|} \right) \sum_{i=1}^{3} a_{\mathbf{k} - 2\mathbf{e}_i} \right], \qquad (16)$$

where $\mathbf{e}_i$ is the ith Cartesian basis vector, and x_i represents the ith Cartesian component of $\mathbf{x}$. After explicitly computing the coefficients for $\|\mathbf{k}\| = 0, 1$, the rest may be computed using equation (16). Furthermore, if any index of $\mathbf{k}$ is negative, then $a_{\mathbf{k}} = 0$.

2.4 Cluster-Particle Treecode for Real-Space Long-Range Asymptotics

In addition to the Coulomb potential, the cluster-particle treecode is utilized in 3D-RISM to compute the LRA total and direct correlation functions, as described in this section.

2.4.1 Direct correlation function

Writing the asymptotic direct correlation function from equation (5) in the cluster-particle form shown in equation (11) yields,

$$c_{\gamma}^{(\mathrm{lr})}(\mathbf{r}_i) = \frac{-q_{\gamma}}{k_b T} \left[\sum_{\mathbf{R}_a \in D} Q_a^U \phi_c^{(\mathrm{lr})}(\mathbf{r}_i, \mathbf{R}_a) + \sum_{l=1}^{L} \sum_{\mathbf{R}_a \in I_l} Q_a^U \phi_c^{(\mathrm{lr})}(\mathbf{r}_i, \mathbf{R}_a) \right], \qquad (17)$$

where the DCF interaction potential is

$$\phi_c^{(\mathrm{lr})}(\mathbf{r}_i, \mathbf{R}_a) = \frac{1}{|\mathbf{r}_i - \mathbf{R}_a|} \operatorname{erf}\left(\frac{|\mathbf{r}_i - \mathbf{R}_a|}{\eta} \right). \qquad (18)$$

Following[39], the Taylor coefficients of the DCF potential function in equation (18) are computed by the recurrence,

$$a_{\mathbf{k}}(\mathbf{x}, \mathbf{y}) = \frac{1}{|\mathbf{x} - \mathbf{y}|^2} \left[\left(2 - \frac{1}{\|\mathbf{k}\|} \right) \sum_{i=1}^{3} (x_i - y_i) a_{\mathbf{k} - \mathbf{e}_i} - \left(1 - \frac{1}{\|\mathbf{k}\|} \right) \sum_{i=1}^{3} a_{\mathbf{k} - 2\mathbf{e}_i} + b_{\mathbf{k}} \right], \quad (19)$$

where the $b_{\mathbf{k}}(\mathbf{x}, \mathbf{y})$ are the Taylor coefficients of an auxiliary Gaussian function, $\exp\left(-|\mathbf{x} - \mathbf{y}|^2 / \eta^2 \right)$, whose recurrence is

$$b_{\mathbf{k}}(\mathbf{x}, \mathbf{y}) = \frac{2}{\eta^2 \|\mathbf{k}\|} \times \left(\sum_{i=1}^{3} (x_i - y_i) b_{\mathbf{k} - \mathbf{e}_i} - \sum_{i=1}^{3} b_{\mathbf{k} - 2\mathbf{e}_i} \right). \qquad (20)$$

11

2.4.2 Total correlation function

Similarly, writing the asymptotic total correlation function from equation (7) in the cluster-particle form shown in equation (11) yields,

$$h_\gamma^{(\mathrm{lr})}(\mathbf{r}_i) = \frac{-q_\gamma}{2\epsilon k_b T} \exp\left(\frac{\kappa_D^2 \eta^2}{4}\right) \left[\sum_{\mathbf{R}_a \in D} Q_a^U \phi_h^{(\mathrm{lr})}(\mathbf{r}_i, \mathbf{R}_a) + \sum_{l=1}^{L} \sum_{\mathbf{R}_a \in I_l} Q_a^U \phi_h^{(\mathrm{lr})}(\mathbf{r}_i, \mathbf{R}_a)\right], \quad (21)$$

where the TCF interaction potential is

$$\phi_h^{(\mathrm{lr})}(\mathbf{r}_i, \mathbf{R}_a) = \frac{1}{|\mathbf{r}_i - \mathbf{R}_a|} \left[e^{(-\kappa_D |\mathbf{r}_i - \mathbf{R}_a|)} \operatorname{erfc}\left(\frac{\kappa_D \eta}{2} - \frac{|\mathbf{r}_i - \mathbf{R}_a|}{\eta}\right) \right.$$

$$\left. - e^{(\kappa_D |\mathbf{r}_i - \mathbf{R}_a|)} \operatorname{erfc}\left(\frac{\kappa_D \eta}{2} + \frac{|\mathbf{r}_i - \mathbf{R}_a|}{\eta}\right)\right]. \quad (22)$$

The TCF potential function in equation (22) has a complicated form and computing its Taylor coefficients is a formidable task. Note however that the Taylor expansions are only used when a source particle and target cluster are well-separated, in other words when $|\mathbf{r}_i - \mathbf{R}_a|$ is large, and in that case we can take advantage of the asymptotic properties of the complementary error function. Thus for large values of $|\mathbf{r} - \mathbf{R}_a|$, we have

$$\operatorname{erfc}\left(\frac{\kappa_D \eta}{2} - \frac{|\mathbf{r} - \mathbf{R}_a|}{\eta}\right) \approx 2, \quad (23)$$

$$\operatorname{erfc}\left(\frac{\kappa_D \eta}{2} + \frac{|\mathbf{r} - \mathbf{R}_a|}{\eta}\right) \approx 0, \quad (24)$$

Using this observation, the TCF interaction potential in equation (22) is approximated by

$$\phi_h^{(\mathrm{lr})}(\mathbf{r}_i, \mathbf{R}_a) \approx \frac{2\exp\left(-\kappa_D |\mathbf{r}_i - \mathbf{R}_a|\right)}{|\mathbf{r}_i - \mathbf{R}_a|}. \quad (25)$$

Functionally, this is nothing more than a screened Coulomb interaction, so following [27], we may use the recurrence relation for its Taylor coefficients given in equation (26),

$$a_\mathbf{k}(\mathbf{x}, \mathbf{y}) = \frac{1}{|\mathbf{x} - \mathbf{y}|^2} \left[\left(2 - \frac{1}{\|\mathbf{k}\|}\right) \sum_{i=1}^{3} (x_i - y_i) a_{\mathbf{k} - \mathbf{e}_i} - \left(1 - \frac{1}{\|\mathbf{k}\|}\right) \sum_{i=1}^{3} a_{\mathbf{k} - 2\mathbf{e}_i} \right.$$

$$\left. + \kappa_D \left(\sum_{i=1}^{3} (x_i - y_i) b_{\mathbf{k} - \mathbf{e}_i} - \sum_{i=1}^{3} b_{\mathbf{k} - 2\mathbf{e}_i}\right)\right], \quad (26)$$

where the $b_\mathbf{k}(\mathbf{x}, \mathbf{y})$ are the Taylor coefficients of an auxiliary exponential function, $2\exp\left(-\kappa_D |\mathbf{x} - \mathbf{y}|\right)$, whose recurrence is

$$b_\mathbf{k}(\mathbf{x}, \mathbf{y}) = \frac{\kappa_D}{\|\mathbf{k}\|} \left(\sum_{i=1}^{3} (x_i - y_i) a_{\mathbf{k} - \mathbf{e}_i} - \sum_{i=1}^{3} a_{\mathbf{k} - 2\mathbf{e}_i}\right). \quad (27)$$

2.5 Frequency Cut-Off for Reciprocal-Space Long-Range Asymptotics

As with their real-space counterparts, direct calculation of the reciprocal-space long-range asymptotics, equations (6) and (8), requires $O(MN)$ operations. However, equations (6) and (8) decay rapidly with increasing k^2, allowing us to apply a cutoff in k^2 beyond which it is reasonable to approximate the asymptotics with 0. The upper bound of the error due to truncating the asympotics is then given by

$$
\begin{aligned}
\epsilon_{c,\text{tol}}^{(\text{lr})} &= \max \left| c_\gamma^{(\text{lr})} \left(\mathbf{k}_{\text{cut}} \right) \right| \\
&= \frac{q_\gamma 4\sqrt{2}\pi\beta}{k_{\text{cut}}^2} \exp\left(-\frac{k_{\text{cut}}^2 \eta^2}{4} \right) \sum_a Q_a^{\text{U}},
\end{aligned}
\tag{28}
$$

$$
\begin{aligned}
\epsilon_{h,\text{tol}}^{(\text{lr})} &= \max \left| h_j^{(\text{lr})} \left(\mathbf{k}_{\text{cut}} \right) \right| \\
&= \frac{q_j 4\sqrt{2}\pi\beta}{\epsilon \left(k_{\text{cut}}^2 + \kappa_D^2 \right)} \exp\left(-\frac{k_{\text{cut}}^2 \eta^2}{4} \right) \sum_a Q_a^{\text{U}},
\end{aligned}
$$

where $\mathbf{k}_{\text{cut}}$ is the highest frequency wave vector considered, and we have used the fact that $|\exp\left(i\mathbf{k} \cdot \mathbf{R}_a \right)| = |\cos\left(\mathbf{k} \cdot \mathbf{R}_a \right) + i\sin\left(\mathbf{k} \cdot \mathbf{R}_a \right)| \leq \sqrt{2}$. The user may then request arbitrary values for the error, or error tolerance, in the long-range asymptotics, and an appropriate cut off can be found using a numerical root-finder, such as Newton-Raphson. An appropriate value for the allowable error will depend on the residual tolerance that the 3D-RISM equations are solved to.

2.6 Truncated Lennard-Jones Potential

It is also possible to truncate the Lennard-Jones (LJ) potential, which is given by

$$
u_{\gamma,a}^{\text{LJ}}(r) = \frac{A_{\gamma,a}}{r^{12}} - \frac{B_{\gamma,a}}{r^6},
$$

where $A_{\gamma,a}$ and $B_{\gamma,a}$ are parameters for the specific pair of interaction sites. Again, we can express the upper bound for the magnitude of the error due to truncation as

$$
\epsilon_{\text{tol}}^{\text{LJ}} = \left| u_{\gamma,a}^{\text{LJ}}(r_{\text{cut}}) \right| = \left| \frac{A_{\gamma,a}}{r_{\text{cut}}^{12}} - \frac{B_{\gamma,a}}{r_{\text{cut}}^6} \right|,
$$

where the cut-off distance, r_{cut}, can be determined for a specific error tolerance. An appropriate value of $\epsilon_{\text{tol}}^{\text{LJ}}$ will depend on the residual tolerance that the 3D-RISM equation is

13

converged to. The cut-off distance for a particular error tolerance depends on the LJ parameters of a and γ, so, in practice, we will determine separate cutoff distances for all pairs. The applied LJ potential is then

$$u_{\gamma,a}^{\text{LJ-trunc}}(r) = \begin{cases} u_{\gamma,a}^{\text{LJ}}(r) & r < r_{\text{cut},\gamma,a} \\ 0 & r > r_{\text{cut},\gamma,a} \end{cases} \tag{29}$$

and the omitted part is

$$u_{\gamma,a}^{\text{LJ-omitted}}(r) = \begin{cases} 0 & r < r_{\text{cut},\gamma,a} \\ u_{\gamma,a}^{\text{LJ}}(r) & r > r_{\text{cut},\gamma,a} \end{cases}. \tag{30}$$

However, even cut-offs with very small error tolerances can result in large systematic errors for various observables. To determine the impact on the excess chemical potential, we can take the functional derivative of equation (4) with respect to the potential energy,

$$\frac{\partial \mu_{\text{ex,KH}}}{\partial u(r)} = k_B T \sum_\gamma \rho_\gamma \int d\mathbf{r}$$

$$\left[\frac{h_\gamma(\mathbf{r})\partial h_\gamma(\mathbf{r})}{\partial u_\gamma(r)}\Theta\left(-h(r)\right) - \frac{\partial c_\gamma(\mathbf{r})}{\partial u_\gamma(r)} - \frac{1}{2}\left(\frac{\partial h_\gamma(\mathbf{r})}{\partial u_\gamma(r)}c_\gamma(\mathbf{r}) + \frac{\partial c_\gamma(\mathbf{r})}{\partial u_\gamma(r)}h_\gamma(\mathbf{r})\right) \right]. \tag{31}$$

For a discrete change, such as the truncation, we can rewrite this as

$$\Delta\mu_{\text{ex,KH}} = k_B T \sum_\gamma \rho_\gamma \int d\mathbf{r}$$

$$\left[h_\gamma(\mathbf{r})\Delta h_\gamma(\mathbf{r})\Theta\left(-h(r)\right) - \Delta c_\gamma(\mathbf{r}) - \frac{1}{2}\left(\Delta h_\gamma(\mathbf{r})c_\gamma(\mathbf{r}) + \Delta c_\gamma(\mathbf{r})h_\gamma(\mathbf{r})\right) \right].$$

Now, consider r_{cut} large enough that $u(r_{\text{cut}}) \ll 1$. Only the TCF and DCF near or beyond the cut-off will be affected, so we use

$$\Delta c_\gamma(\mathbf{r}) \approx -\beta \Delta u_\gamma(\mathbf{r}) \tag{32}$$

$$= -\beta\left(u_\gamma^{\text{LJ}} - u_\gamma^{\text{LJ-trunc}}\right) \tag{33}$$

$$= -\beta u_\gamma^{\text{LJ-tail}} \tag{34}$$

14

Solute	Number of Atoms	Net Charge
Phenol	13	$0e$
Cucurbit[7]uril	122	$0e$
Adhiron	1,324	$-1e$
Tubulin	13,456	$-36e$
Microtubule (1 ring)	174,681	$-455e$
Microtubule (7 ring)	1,222,767	$-3185e$

Table 1: Solutes used in this work.

and neglect terms with $h_\gamma(\mathbf{r})$ or $\Delta h_\gamma(\mathbf{r})$, as these decay much faster than the DCF (see Section 2.2). Equation (31) then simplifies to

$$
\begin{aligned}
\Delta\mu_{\text{ex}}^{\text{KH}} &= k_B T \sum_\gamma \rho_\gamma \int d\mathbf{r}\,[-\Delta c_\gamma(\mathbf{r})] \\
&= -4\pi \sum_a \int_{r_{\text{cut}}}^{\infty} dr\, u_{a,\gamma}^{\text{LJ}}(r)\, r^2 \\
&= -\frac{4\pi}{3} \sum_a \left(\frac{1}{3} \frac{A_{\gamma,a}}{r_{\text{cut},\gamma,a}^9} - \frac{B_{\gamma,a}}{r_{\text{cut},\gamma,a}^3} \right),
\end{aligned}
$$

which is the same correction that may be applied to explicit solvent calculations with truncated LJ interactions[22]. Corrections like this can similarly be determined for other thermodynamic observables.

3 METHODOLOGY

3.1 Benchmark System Preparation

Four solutes were selected for benchmarking and testing, giving a range in the number of atoms of over four orders of magnitude, from 13 to 13,456 atoms (see Table 1). For each solute, the `tleap` program in AmberTools 17[40] was used to assign the final parameters. OpenBabel[41,42] was used to create the 3D structure of phenol from the SMILES string "c1ccc(cc1)O". The general Amber force field parameters (GAFF)[43] and AM1-BCC (AM1

15

with bond charge corrections) charges[44] were assigned using `antechamber`. The 3D structure of cucurbit[7]uril (CB7), a neutral host molecule, was obtained from the Statistical Assessment of the Modeling of Proteins and Ligands 4 (SAMPL4) exercise data set[45]. GAFF parameters were used with charges derived using the pyR.E.D. server[46–50]. Adhiron (PDB ID: 4N6T) is an engineered scaffold protein[51] and was parameterized using Amber FF14SB[52]. A complete crystal structure of tubulin, the main constituent protein of microtubules, does not exist. We constructed a 3D model from PDB IDs 1TVK and 1SA0[53,54], using Modeller[55] to combine the structures and fill in residues missing from the H1-B2α-tubulin loop and the α- and β-tubulin N-termini. The C-terminal tails were not present in the crystal structures and replaced with N-methylamide (NME) caps. Amber FF14SB was used for the amino acids, the pyR.E.D. force field for GTP and GDP, and the MG^{2+} parameters for use with SPC/E water from Li et al.[56].

Solvent density distributions and thermodynamics were computed for microtubules consisting one to seven rings. A single equilibrated microtubule ring from Ref.[57] was used to construct microtubules of different lengths by replicating and translating the ring an integer multiple of 82.746 Å along the microtubule axis, which is the unit cell length from the original simulation. The resulting structure was parameterized using Amber ff14SB[52] for protein, R.E.DD.B.[58] for GTP and GDP and Li/Merz SPC/E 12-6 for Mg^{2+}[56].

3.2 3D-RISM Calculations

All RISM calculations were performed in AmberTools 19[29].

The solvent was prepared for 3D-RISM using the `rism1d` program and consisted of 55.2 M modified SPC/E water[10,59] with 0.1 M NaCl using the corresponding Joung-Cheatham parameters[60]. Dielectrically consistent RISM (DRISM) theory[34] was used with a dielectric constant of 78.44 and the Kovalenko-Hirata (KH) closure[9] at a temperature of 298.15 K. The solution was solved on 65536 grid points with 0.025 Å grid spacing using the default parameters for the modified direct inversion of the iterative subspace (MDIIS) solver[61].

The `rism3d.snglpnt` program was used for all 3D-RISM calculations. Default MDIIS settings, the KH closure, and a 0.5 Å grid spacing were used for all calculations. No cut-off was used for electrostatic interactions. The buffer distance between the solute and the edge

of the solvent box was either explicitly set or determined from the requested LJ tolerance. In all cases, `rism3d.snglpnt` automatically increased the buffer distance to ensure that all grid dimensions were divisible by factors of 2, 3, 5, and 7, and that the number of y- and z-grid points was divisible by the number of processes.

Performance and accuracy of the treecode summation was tested by performing calculations using direct summation for all calculations or using treecode for only one of DCF, TCF, or Coulomb calculations. The direct sum benchmark calculations use a buffer distance of 24 Å and were converged to a residual tolerance of 10^{-13}. All other 3D-RISM calculations detailed below were repeated five times to provide average timings. A buffer distance of 24 Å and grid spacing of 0.5 Å were selected as a compromise between precision and computational cost; obtaining a relative numerical error of 10^{-10} would require a solvent grid much too large to be considered. When using treecode summation, all combinations of the MAC parameter θ from 0.2 to 0.7 in steps of 0.1, the Taylor series order p from 2 to 20 in steps of 2, and maximum leaf size N_0 values of 60, 500, and 4000 were used. In all cases, the 3D-RISM equations were solved to a residual tolerance of 10^{-10}. Optimized serial and parallel jobs were run with the settings in Table 2. To test the parallel scaling of treecode summation, calculations were performed on 1, 2, 4, 8, 16, 24, 32, 48, 64, 72, and 96 processes for all solutes.

Reciprocal-space cut-offs for the long-range asymptotics were compared against the same benchmark calculations as were used for treecode summation. In this case, the residual tolerance was varied from 10^{-10} to 10^{-3}.

For the LJ cut-offs tests, we allowed the buffer distance to vary based on the LJ tolerance. This is required to ensure that the cut-offs fit within the solvent box so that the correction to excess chemical potential can be applied. Only the two neutral systems were considered as they allow us to isolate the LJ contribution from the electrostatics. At even the lowest tolerances, the grid sizes were still manageable. LJ and residual tolerances were independently varied from 10^{-10} to 10^{-3}. The benchmark calculation in this case used LJ and residual tolerances of 10^{-13}.

Serial and parallel calculations for phenol, CB7, 4N6T, and tubulin were run on our Linux cluster, Metropolis, which has seven nodes connected by QDR Infiniband interconnects, each

Solute	Tolerance	TCF		DCF		Coulomb		Reciprocal-
		MAC	Order	MAC	Order	MAC	Order	Space
Microtubules	10^{-6}	0.3	6	0.3	8	0.3	8	10^{-8}
Tubulin	10^{-6}	0.3	6	0.3	8	0.3	8	10^{-8}
Adhiron	10^{-6}	0.3	2	0.3	6	0.3	6	10^{-7}
CB7	10^{-6}	0.3	2	0.3	6	Direct		10^{-7}
Phenol	10^{-6}	0.3	2	Direct		Direct		10^{-7}

Table 2: Optimized 3D-RISM parameter settings. Treecode parameters MAC, order p. All LJ cutoffs were adjusted to fit inside the solvation box.

with 256 GB of memory and two 12 core Intel 2.4 GHz Xeon E5-2600 v2 ("Ivy Bridge-EP") CPUs. AmberTools was compiled with the Intel Fortran and C++ compilers 19.1.053[62] and the OpenMPI 3.1.3 MPI library[63]. Additional parallel benchmarking was performed on the Skylake nodes of Stampede2 at the Texas Advanced Computing Center through the Extreme Science and Engineering Discovery Environment (XSEDE)[64,65], which each have two 24 core Intel Xeon Platinum 8160 CPUs, 192 GB of memory and are connected by a 100 Gb/sec Intel Omni-Path network. In this case, the software was compiled with the Intel Fortran and C++ compilers 17.0.4[66] and MVAPICH2 2.3 MPI library[67]. Microtubule calculations were performed on Bridges at the Pittsburgh Supercomputing Center through XSEDE[64,65], using 16 to 24 cores per job.

4 RESULTS AND DISCUSSION

4.1 Numerical precision requirements

Computational efficiencies from treecode summation and cut-offs must not come at the cost of the numerical precision of computed thermodynamic observables. Generally, the SFE will be the most important value to be calculated with 3D-RISM. The numerical precision required depends on the application to be considered. For SFE calculations absolute errors up to $0.1\,\mathrm{kcal/mol}$ are generally acceptable. An absolute error $< 0.1\,\mathrm{kcal/mol}$ typically means
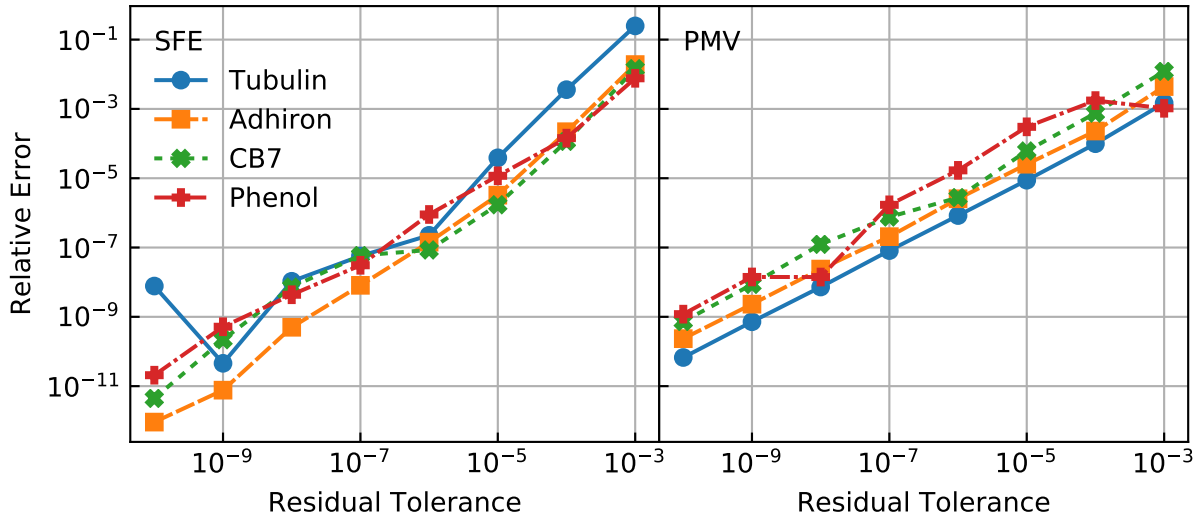
Figure 2: Dependence of the relative numerical error of the solvation free energy (SFE) and partial molar volume (PMV) on the 3D-RISM residual tolerance. Relative errors are calculated against a reference calculation converged to a residual tolerance of 10^{-13}.

relative errors as large as 10^{-3} for small molecules but may need to be less than 10^{-5} or even 10^{-6} for large proteins. To ensure stability, molecular dynamics simulations with 3D-RISM require relative errors less than 10^{-5} to ensure sufficient agreement between SFEs and their derivatives[10]. Energy minimization is even more demanding, requiring relative errors less than 10^{-10}.

In practice, the convergence criterion for our iterative solver is to reach a given maximum allowable residual tolerance. Figure 2 shows the relative error of SFE and PMV thermodynamic quantities as the residual tolerance of the 3D-RISM calculation is adjusted. Overall, we find that that residual tolerance and relative error are directly proportional for observables we have considered. In general, we can say that

$$\epsilon_{\mathrm{SFE}} \gtrsim 10 \times \text{tolerance.} \tag{35}$$

For the SFE, there is no apparent dependence on the size of the solute, though tubulin has an anomalously large relative error for a residual tolerance of 10^{-10}. There does appear to be a dependence on the solute size for the PMV, with larger solutes achieving smaller relative errors for the same residual tolerance. The vast majority of 3D-RISM calculations should

19

use a residual tolerance of 10^{-5} or 10^{-6}.

4.2 Treecode summation

To determine the impact on speed and numerical precision of the treecode parameters MAC, p, and N_0 for TCF LRA, DCF LRA, or Coulomb potential energy, SFEs calculated from 3D-RISM for different size solutes with different treecode parameters were compared against direct sum calculations (Figures 3, 4 and 5). Each data point in the plots represents a different value of p for a given MAC, increasing from right to left. Only results for $N_0 = 500$ are shown, as we found that $N_0 = 60$ and $N_0 = 500$ performed almost identically, while $N_0 = 4000$ was generally slower for the same numerical precision. The cluster of data points in the lower left corner of each plot indicates that increasing p does not provide any additional precision. Though there is some noise in the timing, mostly due to interprocess interference, increasing p almost universally reduces the relative error, but also increases execution time. In all cases, a MAC ≤ 0.4 was sufficient to obtain solutions with the smallest possible error, provided that the number of Taylor series terms was large enough. Results for MAC = 0.7 were omitted, as the performance was consistently worse for all calculations. Otherwise, the best choice of parameters depended on the quantity being summed, TCF LRA, DCF LRA, or Coulomb potential energy, and the size of the solute.

Treecode summation shows the largest relative speedups for the TCF LRA. In fact, treecode is faster than direct summation for all solutes at all precisions and is nearly two orders of magnitude faster than direct summation for tubulin and adhiron for relative errors of 10^{-5}, which is sufficient for most calculations. However, the treecode parameters that give the best performance vary with the relative error and the solute. For tubulin, MAC = 0.4 and 0.5 have the best performance, while MAC = 0.3 is close. MAC = 0.3 provides the best performance for both adhiron and CB7, except for the largest relative errors, where MAC = 0.4 and even 0.5 are slightly faster. Even phenol shows speedups relative to direct summation for all MAC values with an appropriate p; however, the extreme values of MAC = 0.2 and 0.6 have the best performance.

The performance of treecode summation for the DCF LRA is still much better than direct summation for tubulin, adhiron, and CB7 but not for phenol. In contrast to TCF LRA, it
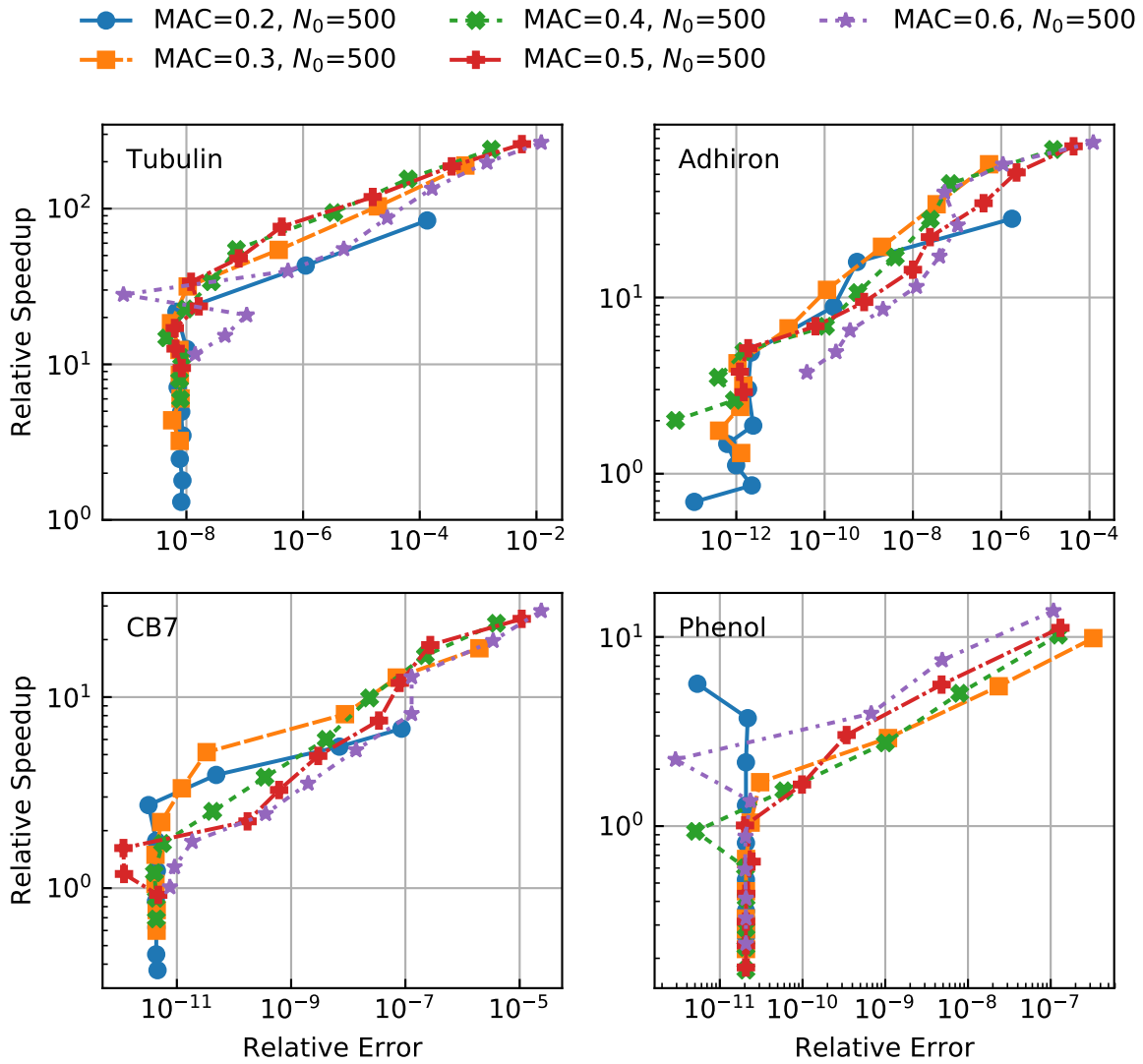
20

Figure 3: Relative speedup of treecode TCF LRA compared to direct summation vs. relative error in $\mu_{\mathrm{ex,kh}}$ for tubulin, adhiron, CB7, and phenol. Taylor series order $p = 2k$, $k = 1, \ldots, 10$, increasing from right to left for each line.
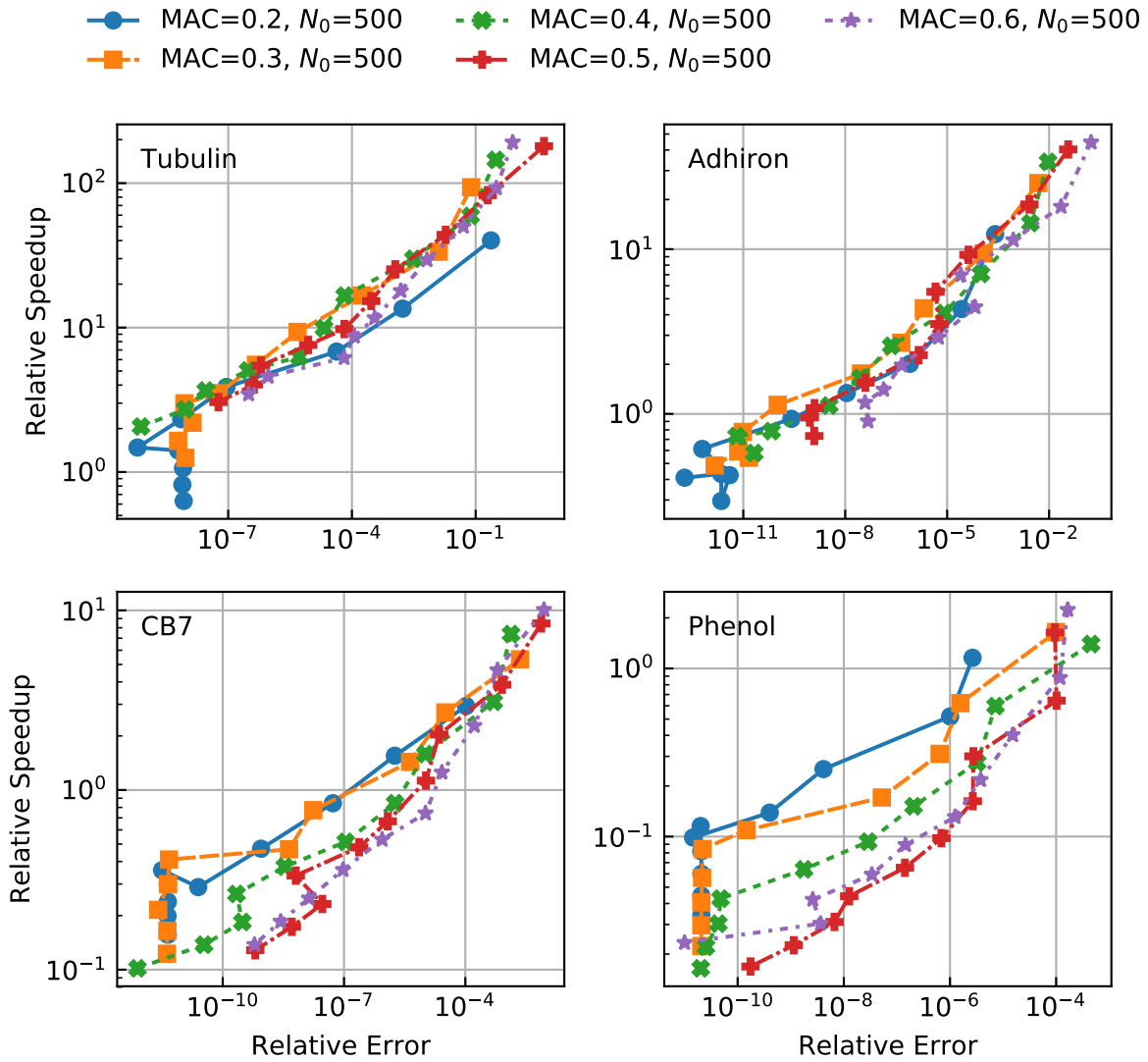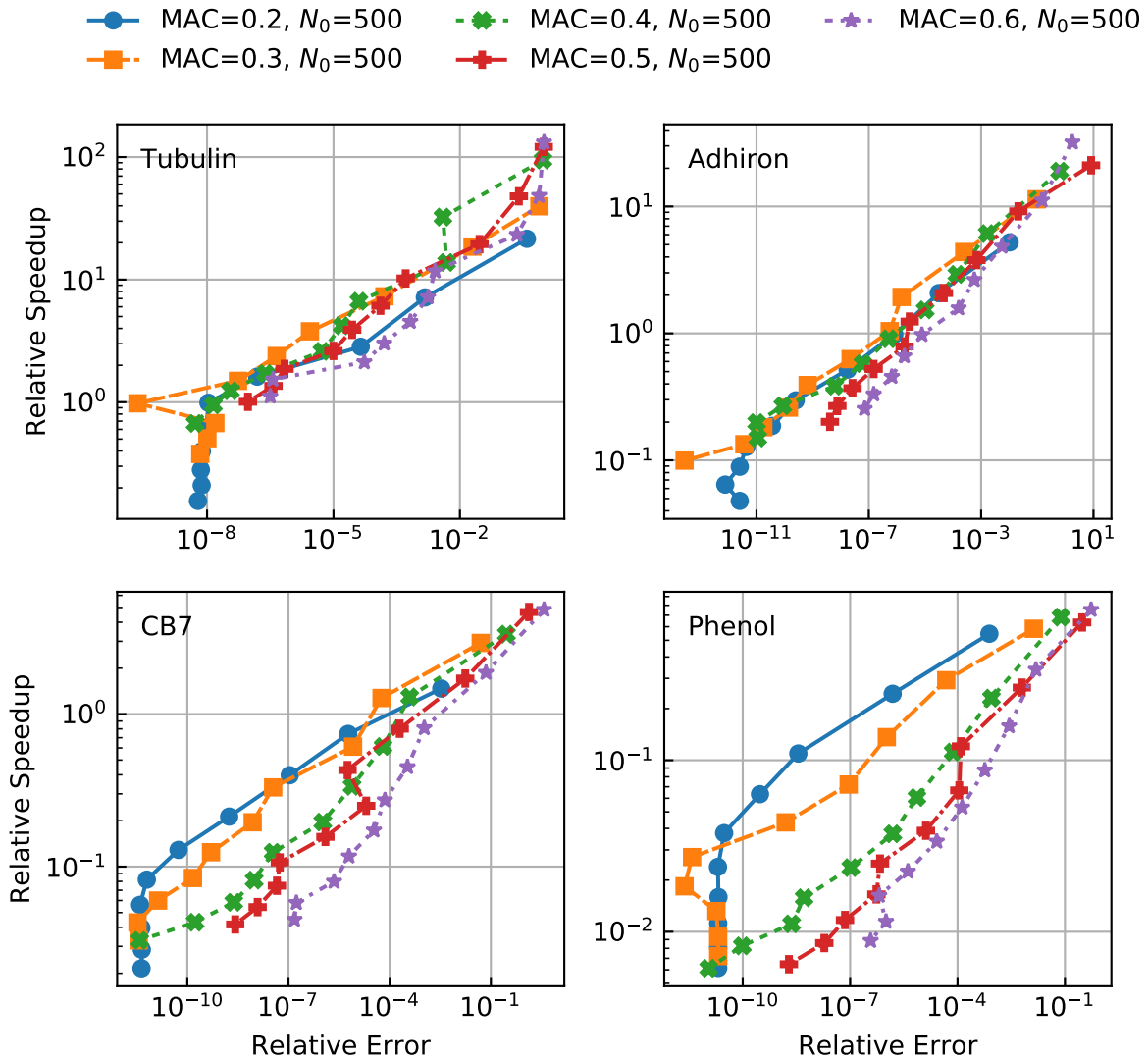
Figure 4: Relative speedup of treecode DCF LRA compared to direct summation vs. relative error in $\mu_{\mathrm{ex,kh}}$ for tubulin, adhiron, CB7, and phenol. Taylor series order $p = 2k$, $k = 1, \ldots, 10$, increasing from right to left for each line.

Figure 5: Relative speedup of treecode Coulomb potential energy compared to direct summation vs. relative error in $\mu_{\mathrm{ex,kh}}$ for tubulin, adhiron, CB7, and phenol. Taylor series order $p = 2k$, $k = 1, \ldots, 10$, increasing from right to left for each line.

	MAC	p	N_0
TCF	0.3	$\max\left(2, \frac{\log_{10}(\text{tolerance})+5.7}{-0.7}\right)$	500
DCF	0.3	$\max\left(2, \frac{\log_{10}(\text{tolerance})+1.9}{-0.8}\right)$	500
Coulomb	0.3	$\max\left(2, \frac{\log_{10}(\text{tolerance})+1.4}{-0.8}\right)$	500

Table 3: Guide to selecting treecode parameters for a given residual tolerance. Recommended parameters should be tested before production use.

is difficult to distinguish between the performance of different MAC values. MAC = 0.3, 0.4 and 0.5 have similar performance for tubulin and adhiron over almost the full range of relative errors. However, MAC = 0.5 is unable to achieve the lowest relative errors, even for $p = 20$, and is generally slower than MAC = 0.3 and 0.4 to achieve the same relative error. For CB7, MAC = 0.2 and 0.3 have nearly identical results, outperforming larger MAC values. The trend towards better performance from smaller MAC values continues for phenol, though the tree code is generally slower than direct summation for this small solute.

The Coulomb potential energy has the simplest functional form and also shows the least benefit from treecode summation. Only tubulin has speedups at all relative errors. However, treecode summation is faster than direct summation for adhiron for relative errors $> 10^{-7}$ and for CB7 for relative errors $> 10^{-4}$. Treecode is slower than direct summation for all phenol calculations. Otherwise, the performance with different MAC values is similar to that observed for DCF LRA. The best performance for tubulin and adhiron is achieved with MAC = 0.3 and 0.4, while MAC = 0.5 has similar performance for larger relative errors but does not reach the lowest relative errors, even for $p = 20$. MAC = 0.2 and 0.3 again show similar performance for CB7, though they are faster than direct summation only for $p < 4$ and $p < 6$, respectively.

4.2.1 Treecode summation parameter selection

Even when considering just biological molecules, there is a wide range of shapes, sizes and charges for both the solutes and solvents that may be studied with 3D-RISM. As a result, it is not possible to prescribe a uniform set of parameters for treecode summation and cut-
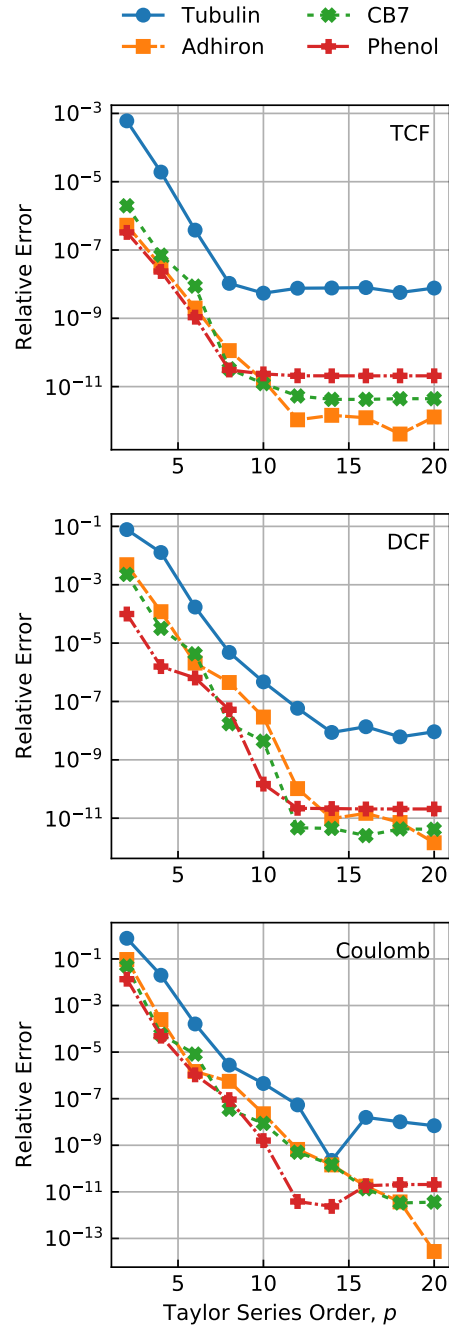
Figure 6: Relative error in $\mu_{\mathrm{ex,kh}}$ of 3D-RISM calculations with treecode parameters MAC = 0.3 and $N_0 = 500$ vs. Taylor series order, p, for tubulin, adhiron, CB7, and phenol.

off methods developed here; some testing will always need to be done before starting a large calculation. However, we can provide guidance to narrow the search for parameters that minimize computation time while preserving necessary numerical precision. Numerical precision is set by the user by specifying the residual tolerance at the beginning of the calculation. As shown in Figure 2, relative error has a linear relationship with the residual tolerance. Therefore, we specify our guidelines relative to the residual tolerance.

Treecode summation requires the user to specify maximum leaf size N_0, MAC parameter, and Taylor series order p. Of these, N_0 and MAC have clear best choices. $N_0 = 500$ is a safe and close to ideal choice for all calculations; $N_0 = 60$ provides almost identical performance while $N_0 = 4000$ gives slower performance in some cases. If a smaller grid spacing of $0.25\,\text{Å}$ is used, then a cluster of $N_0 = 500$ at this smaller grid spacing will occupy about the same volume as $N_0 = 60$ for a grid spacing of $0.5\,\text{Å}$ and we would not expect a significant change in performance. We also recommend MAC $= 0.3$ for all calculations. While other values can be considered for the TCF LRA calculation, MAC $= 0.3$ performs well for all calculations where treecode is faster than direct summation. As observed for TCF and DCF LRA and Coulomb calculations, larger MAC values perform better for larger solutes; MAC $= 0.4$ may be a better choice for solutes larger than those considered here.

The Taylor series order is the most difficult parameter to select as it depends on both the size of the solute, the type of calculation being approximated, and the desired numerical precision. Figure 6 shows the relationship between relative error and Taylor series order for MAC $= 0.3$ and $N_0 = 500$ from Figures 3 to 5 grouped by calculation type across solutes. For all solutes and all calculations, we observe a linear relationship between $\log_{10}(\text{error})$ and p until the error due to the treecode is smaller than the error due to reaching the convergence criterion of the iterative solver, which is a residual tolerance of 10^{-10} in this case. The slope in all cases appears similar, but there are different y-intercepts for the different solutes and calculation types. In addition, tubulin has systematically higher errors, likely due to the convergence anomaly shown in Figure 2. In Table 3, we provide expressions for p-values based on the input residual tolerance, where we have used equation (35) to relate expected error to the input tolerance. As the case of tubulin demonstrates, these expressions are not exact. Rather, we recommend checking the relative error for a given p by performing a test
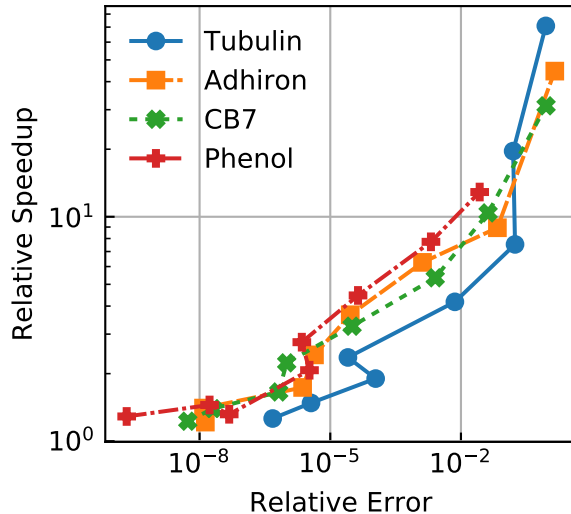
26

Figure 7: Relative speedup in $\mu_{\text{ex,kh}}$ vs. relative error of reciprocal space cut-offs for TCF and DCF LRA compared to the full direct summation. Error tolerance $\epsilon_{\text{cut}}^{(\text{lr})} = 10^{-10}, \ldots, 10^{-3}$, increasing from left to right for each line.

calculation with the prescribed p and another with $p + 2$. If the error is sufficiently small, then other calculations can be performed with different conformations.

It is also worth remembering that treecode summation is not always faster than direct summation. In particular, for small molecules, it may be better to use direct summation for the DCF LRA and Coulomb potential.

4.3 Reciprocal-space cut-offs for long-range asymptotics

Performance data for reciprocal-space LRA cut-offs is given in Figure 7. Here, the cut-off wave number was determined from equation (28) and applied to both the DCF and TCF LRA calculations. Tubulin, again, differs from the other solutes as it requires smaller tolerances for equation (28) to achieve the same relative error in the SFE. The cutoff is based on the magnitude of the wave vector $\mathbf{k}$, the largest value of which is determined by the grid spacing in real-space rather than the size of the grid. In this case, we used a grid spacing of $0.5\,\text{Å}$, which is coarse but still of practical use. Finer grid spacings of 0.25 or $0.3\,\text{Å}$ are typically used for SFE calculations and would produce larger speed-ups for the same cut-off. Regardless,
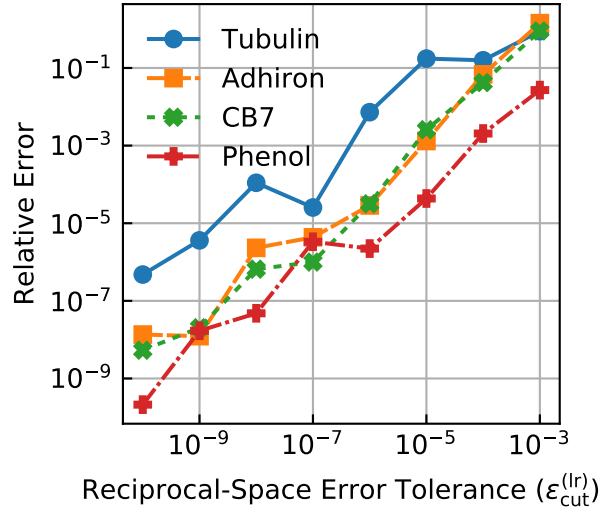
27

Figure 8: Relative error in $\mu_{\text{ex,kh}}$ due to applying reciprocal space cut-offs for TCF and DCF LRA compared to full summation.

cut-offs are always faster than the full direct summation, even if the speedups may be small in some cases.

4.3.1 Reciprocal-space cut-off parameter selection

To select the error tolerance for the reciprocal-space LRA cutoff, it is useful to compare the relative error of the calculation against the selected error tolerance, as in Figures 8. We observe, when plotted on a log-log scale, there is a nearly linear relationship between the relative error in the SFE and the cutoff error tolerance, suggesting that the cut-off tolerance can be selected using the relationship

$$\text{error} = A \cdot \epsilon_{\text{cut}}^{B} \tag{36}$$

where a and b fit to the data. Using the fact that the relative error in the SFE is typically 10 times larger than the specified residual tolerance, equation (35), we can rewrite this expression as

$$\epsilon_{\text{cut}} = a \cdot \text{tolerance}^{b}$$
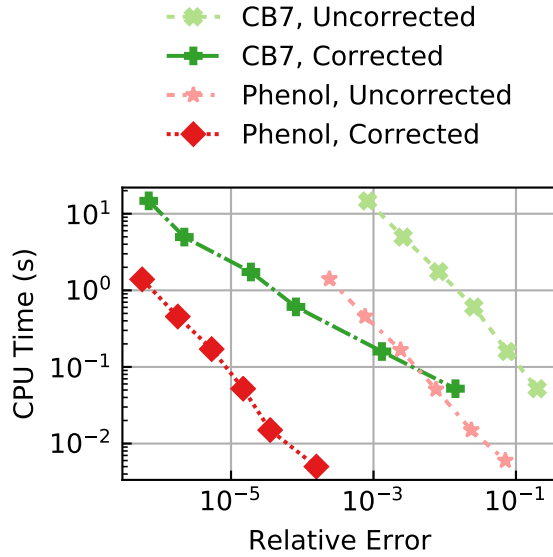
where $a = (10/A)^{1/B}$ and $b = 1/B$.

28

Figure 9: Calculation time for Lennard-Jones potential energy vs. SFE relative error due to truncation with and without error correction for the CB7 and phenol test cases. Error tolerance $\epsilon_{\mathrm{cut}}^{(\mathrm{LJ})} = 10^{-8}, \ldots, 10^{-3}$, increasing from left to right for each line.

From Figure 8, we observe that the reciprocal-space LRA for tubulin incurs a larger error for the same value of $\epsilon_{\mathrm{cut}}^{(\mathrm{lr})}$ compared to other solutes. Fitting to just the adhiron, CB7 and phenol data, we arrive at

$$\epsilon_{\mathrm{cut}}^{(\mathrm{lr})} \approx 0.04 \cdot \mathrm{tolerance}^{0.9}$$

while fitting tubulin gives

$$\epsilon_{\mathrm{cut}}^{(\mathrm{lr})} \approx 0.006 \cdot \mathrm{tolerance}^{1.1}.$$

This roughly equates to using a cut-off tolerance that is a factor of 10 smaller than the residual tolerance for most solutes, though some, like tubulin, may require the cutoff tolerance to be a factor of 100 smaller than the residual tolerance to avoid losing numerical precision.

4.4 Real-space cut-offs for the Lennard-Jones potential

LJ real-space cut-offs differ from the previous calculations as we have an analytic correction for the omitted part of the calculation. In cases where the solute is neutral or the solvent is non-ionic, long-range LJ iterations are the largest source of error. In these cases, the

solvation box can be safely trimmed to include the LJ cut-off distance and nothing more. If the solute is charged and ionic solvents or co-solvents are present, long-range electrostatics will dominate the size of the box and the LJ calculation need only be considered for a small part of it.

For testing purposes, we used only phenol and CB7, which are electrostatically neutral, to clearly see the effect of the LJ cutoff and to allow us to set the size of the solvent box to exactly accommodate the cut-off. Figure 9 compares the cut-off with and without the correction where each data point from right to left reduces the LJ error tolerance by a factor of 10 and increases the box size accordingly. As the box-size is determined by cut-off tolerance, only the corners are omitted from the LJ calculation; therefore, not applying cut-offs would provide only slight differences from the uncorrected cut-offs. While increasing the cut-off distance reduces the SFE relative error for both CB7 and phenol, the cut-off correction cancels a significant amount of error compared to the cut-off alone for very little computation cost. For phenol, as the cut-off error tolerance, $\epsilon_{\text{cut}}^{(\text{LJ})}$, is decreased, the relative error in the SFE decreases at the same rate for both the corrected and uncorrected data (Figures 9 and 10). However, the correction reduces the relative error by a factor of more than 10^{-2}. As a result, the same relative error is achieved from using the correction with $\epsilon_{\text{cut}}^{(\text{LJ})} = 10^{-3}$, a buffer of about $10\,\text{Å}$, as for the uncorrected calculation with $\epsilon_{\text{cut}}^{(\text{LJ})} = 10^{-8}$, a buffer of almost $70\,\text{Å}$. Furthermore, because the grid size is reduced, the corrected calculation is more than 100X faster than the uncorrected value for both the LJ part of the calculation and the total time. The relationship between the relative error and $\epsilon_{\text{cut}}^{(\text{LJ})}$ is slightly different for CB7 as the correction becomes more effective as $\epsilon_{\text{cut}}^{(\text{LJ})}$ is lowered. Still, to achieve the same relative error, the corrected calculation is at least 10X faster for the same precision and is typical more than 100X faster.

4.4.1 Lennard-Jones cut-off parameter selection

Only CB7 and phenol were examined for the Lennard-Jones cut-off error tolerance and these show significantly different responses at high values of $\epsilon_{\text{cut}}^{(\text{LJ})}$. As with the reciprocal-space LRA cut-off, the real-space Lennard-Jones cut-off can be selected by comparing the relative error of the calculation against the selected error tolerance (10) and fitting with equation (36).
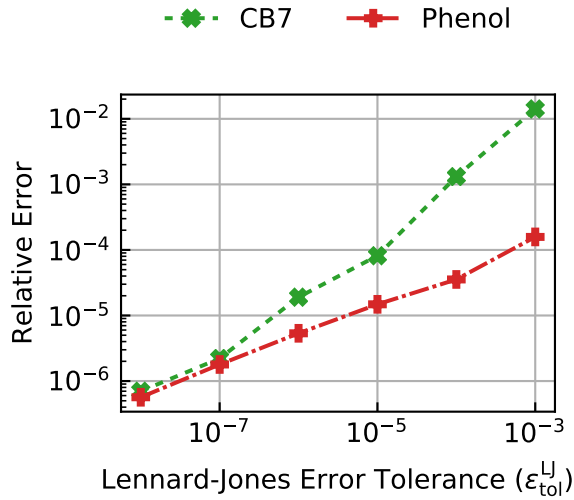
30

Figure 10: Relative error in $\mu_{\text{ex,kh}}$ due to applying real-space cutoffs for the Lennard-Jones potential with different tolerances.

For phenol, we have an error estimate of

$$\epsilon_{\text{cut}}^{(\text{LJ})} \approx 10^7 \cdot \text{tolerance}^{2.1}$$

and, for CB7, we have

$$\epsilon_{\text{cut}}^{(\text{LJ})} \approx 3 \cdot \text{tolerance}^{1.1}.$$

The enormous difference in parameters for phenol and CB7 is due to difference in slopes in Figure 10. A safe default choice would be to set $\epsilon_{\text{cut}}^{(\text{LJ})}$ to be one tenth the residual tolerance. This should guarantee the desired precision over a wide range of residual tolerances and, for small solutes, will have little impact on the calculation time compared to smaller values of $\epsilon_{\text{cut}}^{(\text{LJ})}$. If a more aggressive optimization is desired for larger solutes, the phenol values can be used as a starting point and compared against calculations using larger cutoff tolerances.

4.5   Accuracy of density distributions

While the solvation free energy is the most commonly used output from 3D-RISM calculations, it is the result of integrating over the TCF and DCF, which are the solutions to the 3D-RISM equations. A more direct way to assess the accuracy of the solution is through the

Kirkwood-Buff integral for each solvent site[68,69],

$$I_\alpha = \int_{\text{all space}} h_\alpha \left(\mathbf{r} \right) \, d\mathbf{r},$$

which involves only the TCF and is used to calculate various properties, such as the excess number of atoms. In Table 4, we have employed direct summation with Lennard-Jones and long-range asymptotics cutoffs solved to a tolerance of 10^{-10} as the reference calculation to assess the impact of approximations of treecode summation and cutoff separately. When treecode summation solutions with the same cutoffs are compared against the reference, the difference in the Kirkwood-Buff integral ranges from 5.1×10^{-5} to 1.2×10^{-7}, which is within the expected error as the treecode calculations are solved to a tolerance of 10^{-6}. Conversely, the errors are larger when compared against direct summation with no cutoffs, but this is primarily due to the Lennard-Jones corrections for the Kirkwood-Buff integral, of the type described in Section 2.6. In this case, the result from the cutoff with correction is more accurate than truncating the LJ interaction at the edge of the solvation grid, though the difference is small.

Alternatively, the accuracy of the full solution can be assessed by calculating the root-mean-squared difference between the TCF grids. Table 4 shows that the RMSD for the treecode summation are of similar magnitude to the errors in the Kirkwood-Buff integrals. Furthermore, they are only slightly larger than the target residual for the solution, which is the RMSD between TCF from the 3D-RISM equation, equation (1), and the closure, equation (3). When we compare the direct summation with no cutoffs, we see that the errors are generally lower in magnitude but larger than the 10^{-10} residual tolerance of the solver. This is due to tolerance of the cutoffs (see Table 2) and shows that the cutoffs have little impact on the solution.

4.6 Scaling with solute size

Using parameters determined in Section 4.2.1, 4.3.1 and 4.4.1, we can compare the computing time required for total cost of the calculation with treecode summation and cut-offs with the performance of direct summation (Figure 11). For comparison to direct summation (Figure 1), we again use a residual tolerance of 10^{-6}, which is sufficient for most 3D-RISM

Solute	Method	Kirkwood-Buff				RMSD
		O	H	Na$^+$	Cl$^-$	$\sum_\gamma h_\gamma(\mathbf{r})$
Tubulin	Direct sum, no cutoffs	3.9×10^{-4}	3.9×10^{-4}	1.2×10^{-3}	7.6×10^{-4}	1.8×10^{-4}
	Treecode sum, cutoffs	5.7×10^{-7}	1.7×10^{-6}	1.9×10^{-6}	1.2×10^{-7}	3.8×10^{-5}
Adhiron	Direct sum, no cutoffs	2.3×10^{-4}	2.3×10^{-4}	2.9×10^{-3}	5.3×10^{-4}	9.8×10^{-5}
	Treecode sum, cutoffs	7.5×10^{-7}	4.6×10^{-6}	1.9×10^{-5}	1.1×10^{-6}	1.9×10^{-4}
CB7	Direct sum, no cutoffs	6.8×10^{-4}	6.8×10^{-4}	1.0×10^{-3}	1.0×10^{-3}	4.7×10^{-5}
	Treecode sum, cutoffs	1.1×10^{-5}	1.1×10^{-6}	1.5×10^{-5}	8.1×10^{-6}	7.3×10^{-4}
Phenol	Direct sum, no cutoffs	6.1×10^{-4}	6.1×10^{-4}	7.5×10^{-4}	7.1×10^{-4}	1.3×10^{-5}
	Treecode sum, cutoffs	3.7×10^{-5}	5.1×10^{-5}	2.4×10^{-6}	1.1×10^{-5}	2.5×10^{-4}

Table 4: Relative error compared to direct summation with Lennard-Jones and long-range asymptotic correction cut-offs for the Kirkwood-Buff integral of each solvent species and the root-mean-squared deviation of the total correlation function. Treecode summation and cutoff parameters are from Table 2.
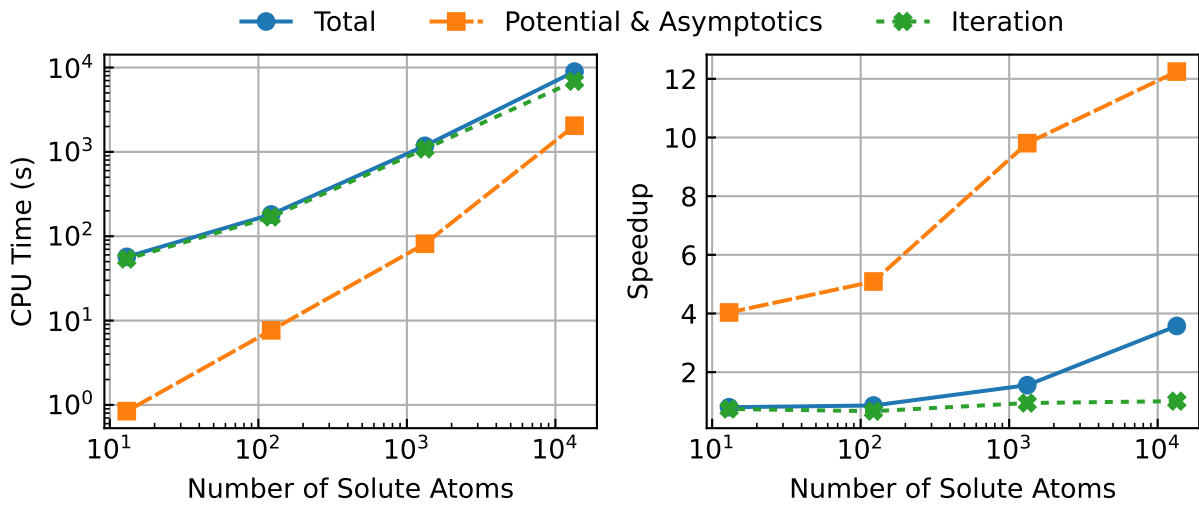
Figure 11: Total runtime of 3D-RISM converged to a tolerance of 10^{-6} with potential and asymptotics calculated using treecode summation and speedup relative to direct summation (Figure 1). Required runtime is shown for setting up the calculations (potential and asymptotics) and iterating to a converged solution. Treecode and cut-off parameters can be found in Table 2.
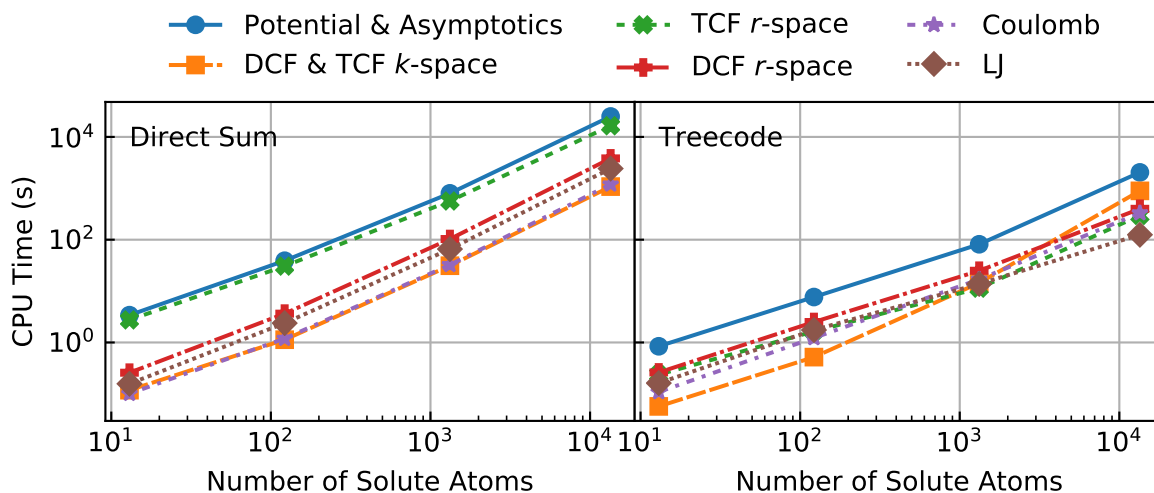
Figure 12: Runtime for different components of the potential and asymptotics calculation for Figure 11 using direct and treecode summation. Calculations were solved to a residual tolerance of 10^{-6}.

calculations. For larger tolerances, more aggressive parameters can be used, resulting in potentially larger speedups. Combined, treecode summation and cut-off methods can significantly reduce the total calculation time – nearly 4X faster in the case of tubulin and 1.6X for adhiron. In the case of tubulin, computing the potential and asymptotics accounts for about 20% of the total runtime when using treecode and cut-offs versus nearly 80% using direct summation. Smaller solutes obtain similar results; potential and asymptotics calculations are accelerated by a factor of 3X to 10X and, with the exception of tubulin, account for less than 10% of the total runtime when treecode summation is used. Overall, iteration time is now the dominant computational cost for all solute sizes.

To assess how the treecode summation and cut-off methods individually perform, we have broken down the potential and asymptotics into their various components (Figure 12). For the direct summation calculations, the real-space TCF LRA calculation dominates the runtime, followed by the real-space DCF LRA and the Coulomb potential energy calculations. After applying our treecode summation and cut-off methods, the real-space DCF LRA is the most expensive part of the calculation for all but tubulin while the real-space TCF LRA and Coulomb potential energy require about the same amount of time as the Lennard-Jones

potential energy. Tubulin is an exception, as the reciprocal-space DCF and TCF LRA require the largest fraction of time, about 25% of the total time for the potential and asymptotics.

Using a cutoff for the reciprocal-space DCF and TCF LRA is the only optimization that does not improve the scaling with system size. As it is a cut-off in reciprocal space, only large values of k are omitted, which are determined by the grid spacing used and have nothing to do with solute size. As a result, we observe a performance improvement of 2.5-3.5X for all solutes and anticipate even greater speedups for finer grid-spacings. In fact, there should be little or no additional computation time for calculating the reciprocal-space DCF and TCF LRA on finer grids. Despite the fact that the scaling remains $O\left(N_{\mathrm{atom}}N_{\mathrm{grid}}\right)$, the use of cut-offs means that this part of the calculation remains a small fraction of the total and may be further reduced by other means, such as lookup-tables.

4.7 Parallel scaling

3D-RISM in AmberTools is parallelized using the message passing interface (MPI) with a distributed memory model. This allows 3D-RISM to make use of the aggregate memory of multiple nodes for large systems but means that the code must follow the memory model of the underlying FFT library for all of the solvation grids. We use the Fastest Fourier Transform in the West (FFTW)[70] library, which decomposes the memory in real-space along the z-axis into slabs. Each process gets one slab of each grid, whether or not that grid is directly processed by FFTW, and includes potential energy and LRA grids. In order to ensure adequate load balancing, 3D-RISM uses equal sized slabs for all nodes and will automatically increase the total grid size to ensure this if necessary. At the same time, each process gets a full copy of the solute information. This accounts for much less memory than the grids and is only a small fraction of the the aggregate memory footprint, even for 96 processes.

Treecode summation and cut-off methods have a small effect on the overall parallel scaling of 3D-RISM (Figure 13). On the Metropolis cluster, with only 24 cores per node, calculations on all solutes scale well until 24 cores for both types of calculations. Adding resources beyond 24 cores causes the solution for phenol to slow down. CB7 is the next to saturate at about 72 cores for direct summation while adhiron and tubulin do not exhibit any slow down.
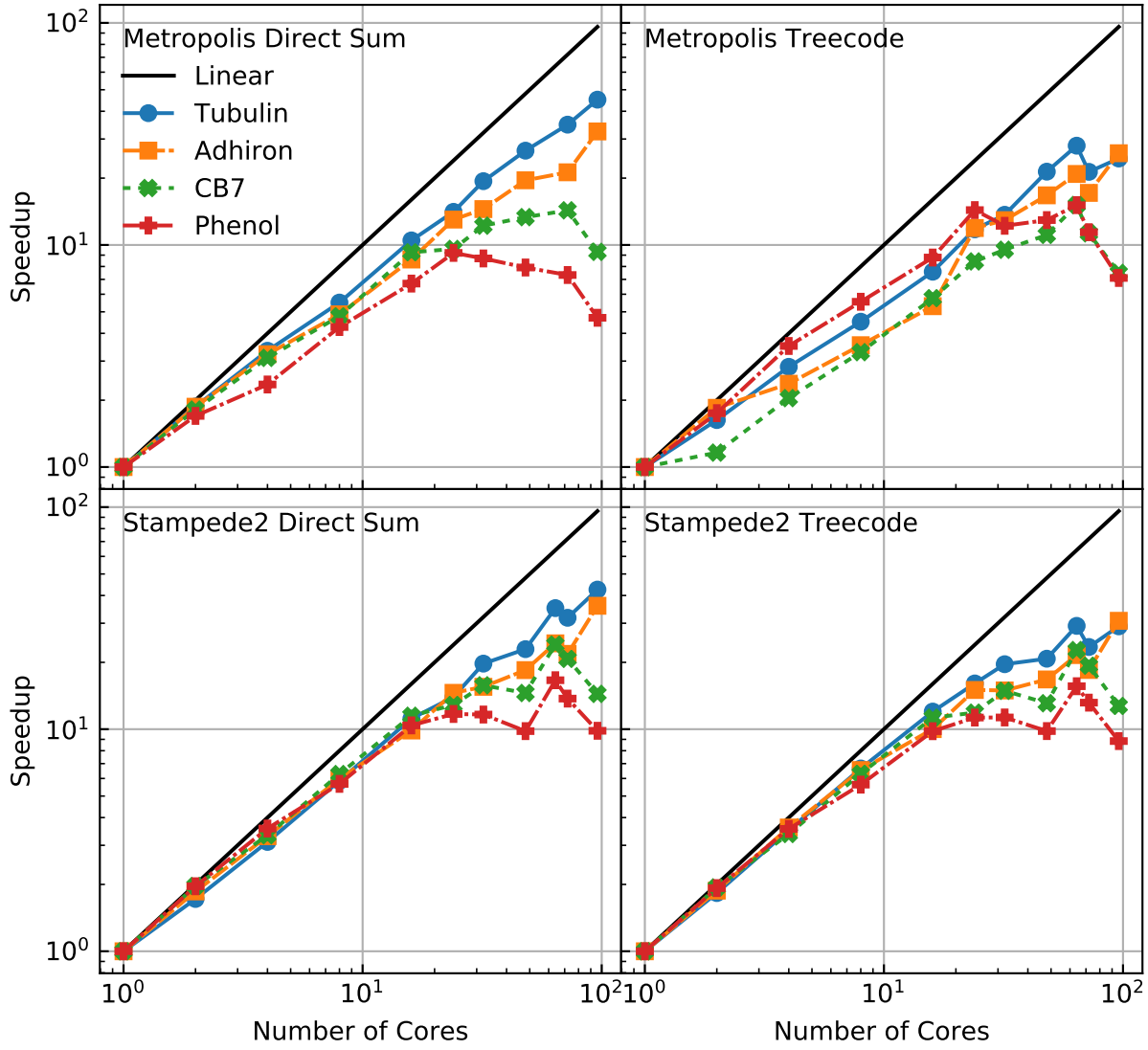
Figure 13: Speedup over multiple cores of the total calculation time for direct and treecode summation 3D-RISM calculations converged to a tolerance of 10^{-6} on Metropolis and Stampede2 clusters. Treecode and cut-off parameters can be found in Table 2.

As expected, large systems scale better than smaller systems. However, for the treecode summation and cut-off methods, 64 cores appears to be the limit for all solutes. In addition, phenol now exhibits the best scaling of all the systems until it saturates, while there is a notable decline in the scaling of CB7 and adhiron.

To investigate the role of hardware, we also ran calculations on Stampede2, which has double the cores and memory bandwidth of Metropolis (Figure 13). As with Metropolis, all solutes scale well up to 24 cores for both direct summation and treecode/cut-off methods. Unlike Metropolis, scaling is closer to linear and does not seem to be affected by solute size at these small core numbers. However, 24 cores remains the scaling limit for phenol, which indicates that this is a software limitation. After this point, larger solutes scale more efficiently and phenol and CB7 saturate at 24 and 64 cores respectively. Otherwise, treecode/cut-off calculations scale as well as direct summation calculations until the high core counts are reached.

As we did with single-core performance, to better understand the contributions of different parts of the calculation, we have decomposed the calculation into various components for the potential and asympytotics calculations and the iteration time, the latter of which we have not attempted to accelerate. We use tubulin for this discussion (Figure 14), though the same behavior is observed for the other molecules as well.

For direct-sum calculations, the largest bottleneck to scaling is the iterative stage of the calculation. The scaling of this part of the code is sub-linearly and becomes the most expensive part of the calculation when eight or more processes are used. In contrast, all other parts of the calculation scale almost linearly. As each MPI process has a full copy of the solute, the direct sum calculation is trivially parallel, with no communication between the processes, and should scale linearly as observed. The cause of the sub-linear scaling of the iterative calculation is beyond the scope of this paper, but is likely hardware dependent as the iterative calculation performs much better on Stampede2. Profiling data (not shown) indicates that the iterative calculation has much higher memory bandwidth requirements than the direct summation, and the higher memory bandwidth of Stampede2 could account for these differences.

Applying cut-offs to LJ and reciprocal-space calculations has little impact on scaling.
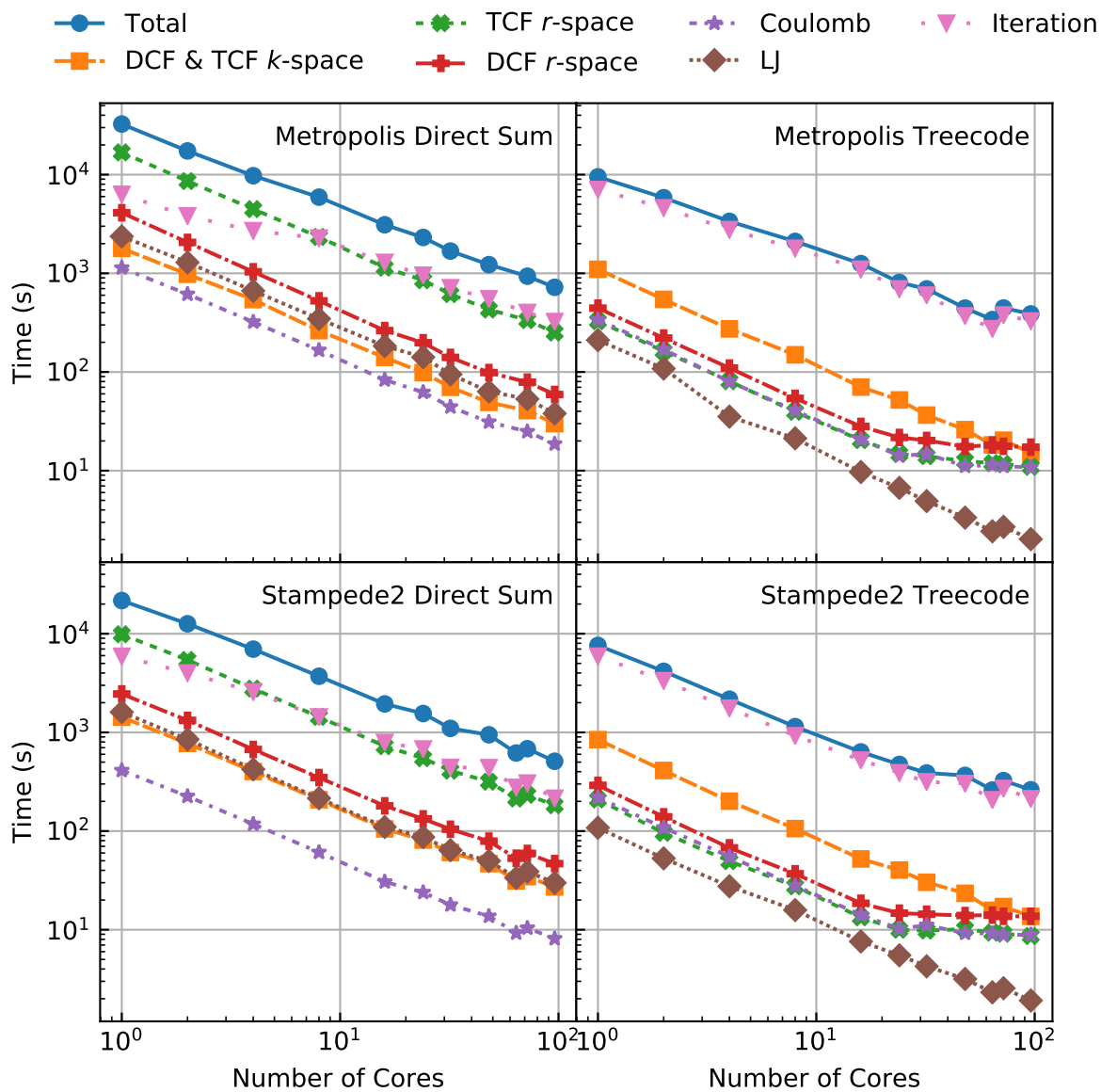
Figure 14: Computation time over multiple cores of the total calculation time and various components for direct and treecode summation 3D-RISM calculations on a tubulin dimer converged to a tolerance of 10^{-6} on Metropolis and Stampede2 clusters. Treecode and cut-off parameters can be found in Table 2.

For tubulin, the selected error tolerance for the reciprocal-space cut-offs excludes few grid points, so there is little difference from the no-cut-off calculation. LJ cut-off calculations do not require any communication but do limit the amount of work some processes are required to do. However, because the cut-offs are set to fit inside the solvation box and each solute atom has the cut-off applied independently, the work load remains relatively balanced. On Metropolis, the LJ cut-off calculation is slightly super-linear but this is likely due to some small variance in the single processor calculation. On Stampede2, scaling of LJ and reciprocal-space LRA with and without cut-offs is nearly identical.

TCF and DCF LRA and Coulomb potential energy with treecode summation all scale well until around 32 cores on both Metropolis and Stampede2. The most likely reason for the scaling to plateau is that each process performs its own treecode decomposition on its own piece of the grid. Because a slab-decomposition memory layout is required by the FFTW3 library we use for the iterative part of the code, the memory that each process receives becomes narrower as the process count increases. As tree nodes narrow, it is more difficult to satisfy the MAC and the Taylor expansion becomes less efficient. To partially alleviate this constraint, when the tree is built, nodes are only subdivided along a given Cartesian direction if the node box length parallel to that direction is within a factor of $\sqrt{2}$ of the shortest box length. However, this can result in only two or four children in a given tree level, and the top levels of the tree will still have node boxes with uneven aspect ratios, so narrow tree root nodes may still affect performance. Additionally, slabs near the middle of the grid where the solute is located may end up doing more local source particle-target particle direct sums, while slabs near the edges of the grid will be able to use the Taylor expansion much more often.

Overall, the performance of treecode summation for high process counts does not adversely affect the overall parallel scaling of the calculation as the total time and scaling is dominated by the iterative solver. This is because treecode summation is so much faster than direct summation, even at the highest node counts, that it is an almost negligible part of the calculation.
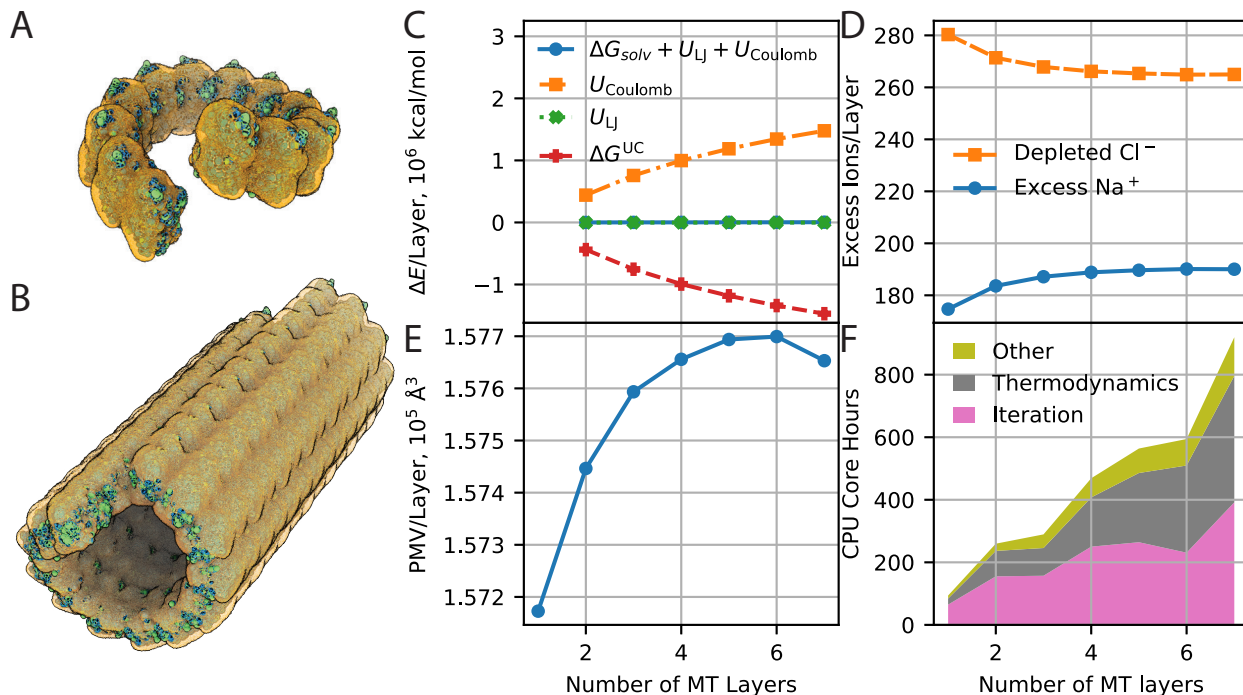
Figure 15: 3D-RISM calculations on microtubules of various lengths. (A) A single micro-tubule layer of 13 tubulin dimers and (B) a microtubule composed of seven layers. Solvent isosurfaces are two times the bulk density each species; transparent orange : Na^+, green : Cl^-, blue : water oxygen, white : water hydrogen. Thermodynamic properties and computation time as a function of microtubule layers: (C) Non-bonded potential energy and solvation free energy with the Universal Correction, (D) number of excess Na^+or depleted CL^- ions per layer, (E) partial molar volume per layer, and (F) CPU core hours for each calculation.

4.8   Application to microtubule stability and growth

Microtubules are components of the cytoskeleton found in all eukaryotic cells and self-assemble from tubulin dimers.[71] Stability is critical to the function of microtubules, par-ticularly during cell mitosis when they go through phases of linear growth and rapid col-lapse, known as "dynamic-instability". Despite decades of work, the physical mechanisms of microtubule stability and growth are still not well understood. Here, we demonstrate how 3D-RISM can quantify the role of ions in stabilizing microtubules. Figure 15 A and B shows the distribution of solvent around a single ring of 13 tubulin dimers and a microtubule com-posed of seven rings. It is clear that a large excess of sodium ions surrounds the microtubule

but most of the neutralization is due to a depletion of chloride ions, as can be seen from the preferential interaction parameter (PIP) in molar units[2,72] (Figure 15 D),

$$\Gamma_\alpha^{(M)} = \rho_\alpha \int_{\text{all space}} g_\alpha(\mathbf{r}) - 1 \, d\mathbf{r}.$$

This is due to both the displacement of chloride and sodium ions – approximately 95 ion pairs per layer – as well as the electrostatic repulsion and attraction. As the number of MT layers grows, the partial molar volume of the layers remains almost constant (Figure 15 E), so the change in the number of excluded ions per layer is not due to changes in the excluded volume but displacement of chloride ions at the ring interface.

To understand the role of solvation energetically, we use the molecular mechanics with 3D-RISM[73] approach without entropy to calculate the change in interaction energy and the solvation free energy (Figure 15 C) for adding a ring to a microtubule. Here, the change in the effective potential energy is

$$\Delta E = \Delta E_{\text{mm}} + \Delta \Delta G_{\text{solv}}$$
$$= \left(E_{\text{mm}}^{\text{MT}} - E_{\text{mm}}^{\text{MT-L}} - E_{\text{mm}}^{\text{L}} \right)$$
$$\left(\Delta G_{\text{solv}}^{\text{MT}} - \Delta G_{\text{solv}}^{\text{MT-L}} - \Delta G_{\text{solv}}^{\text{L}} \right),$$

where MT and L denote the final microtubule and the added layer, E_{mm} is the molecular mechanics energy, and ΔG_{solv} is the solvation free energy, calculated with the partial molar volume correction of Palmer et al.,[16]

$$\Delta G_{\text{solv}} = \Delta G_{\text{3D-RISM}} + av + b,$$

which compensates for the mechanical work due to the large pressure artifacts present in 3D-RISM. Here v is the partial molar volume, $a = -0.1499 \, \text{kcal/mol/Å}^3$ compensates for the effective pressure and $b = -0.1 \, \text{kcal/mol}$ corrects for artifacts from other sources, such as the grid spacing. The constants a and b are taken from previous work for the specific closure, grid spacing, and water model, temperature and density used here.[15] As the rings are identical and rigid, both the internal energy of the dimers and the entropy are omitted and only intermolecular contributions to the energy are included: Coulomb, Lennard-Jones and solvation solvation free energy. Overall, the ions in the solvent neutralize the overall

charge of the microtubule and combine with short-range inter-dimer salt bridges to almost stabilize the structure. While not visible in Figure 15 C due to the scale, the change in total energy of adding one ring slightly increases from 75 kcal/mol to 4063 kcal/mol as the microtubule grows, while the LJ contribution is consistent at -2333 kcal/mol per layer. The slightly positive binding energy is consistent with the fact that we used GDP-tubulin in our calculations, which is known to cause instability and collapse in microtubules. Since the ΔE is only about 1% of the magnitude of $\Delta\Delta G_{\text{solv}}$ and $\Delta U_{\text{Coulomb}}$, small changes in structure, such as an extended conformation of GTP-tubulin, could tip the balance to stability.

Our results show the role of the ionic environment in stabilizing microtubules but does not elucidate the mechanism that leads to instability when GTP is hydrolyzed to GDP. A leading hypothesis is that hydrolysis induces conformational changes in tubulin, causing strain in the lattice due to the kinked conformation of the GDP-tubulin or compaction of dimers within the lattice.[74] Numerous molecular dynamics simulations of free tubulin have failed to show a clear conformational change between GTP- and GDP-tubulin, with both adopting a kinked conformation, though there is evidence that GTP-tubulin is more flexible and less strained in the microtubule lattice.[75–81] Compaction of GDP-tubulin relative to GTP-tubulin in simulation of protofilaments has been recently reported[75] and the structure used in this calculation represents a compact lattice, as it is composed of GDP-tubulin with a dimer repeat length of 81.2 Å[57]. However, microtubule lattice compaction is not observed in all species[82], so it is unclear what role it plays. Our calculation could be extended to include different conforms and isotypes to capture the role of both inter- and intramolecular energy in microtubule stability.

Treecode summation was essential to completing these calculations, as the total time was roughly linearly proportionate to the number of layers added (Figure 15 F). However, while the iterative solver remains the most costly part of the calculation, we see, unlike our smaller systems, that calculating the final thermodynamics now requires a significant fraction of the total time while the potential and asymptotics time is around 10% of the total calculation. The reason for the increased cost of the thermodynamics is the last integral in equation (9). Though negligible for smaller systems, it has a complexity of $O\left(N_{\text{atom}}^2\right)$[21] and becomes significant for systems of this size. This will need to be addressed for practical

extension to even larger systems.

5 CONCLUSIONS

In this work, we have developed and implemented treecode summation for long-range interactions and cut-offs for short-range interactions to accelerate the potential and long-range asymptotics calculations for non-periodic 3D-RISM calculations. The previous approach used direct sum calculations that scaled as $O\left(N_{\mathrm{grid}}N_{\mathrm{atom}}\right)$ and were a major impediment to studying large proteins and protein complexes. By implementing the numerical methods demonstrated here, we have reduced the computational complexity to at most $O((N_{\mathrm{grid}} + N_{\mathrm{atom}})\log N_{\mathrm{grid}})$ for almost all parts of the calculation. Furthermore, our analytic correction for the Lennard-Jones cut-off enables much smaller solvation boxes to be used for neutral solutes or non-ionic solvents, which can reduce the required computation time by a factor of 100 for many situations.

Though proteins of almost any size will benefit from using treecode summation and analytically corrected Lennard-Jones calculations, larger systems benefit more. For the largest protein in our benchmark calculations, tubulin, the total computation time was reduced by a factor of 4 and the potential and asymptotics now account for only 20% of the calculation time, compared to 80% when direct summation was used. These methods also enabled us to calculate the solvation thermodynamics of a microtubule composed of 910 tubulin dimers with 3D-RISM – a calculation impossible before now. Our results show the significant role that solvation plays in the balance between microtubule stability and instability.

Parallel calculations with these new methods scale almost linearly and the iterative solver remains the largest impediment to parallel scaling. Though calculating the long-range asymptotic correction to solvation free energy is a significant time cost for the largest microtubule system, the iterative solver is now the most expensive part of the calculation for almost all practical length scales and future work will focus on accelerating this.

6 ACKNOWLEDGMENTS

7 DATA AVAILABILITY STATEMENT

The code for this paper is freely available through AmberTools (`https://ambermd.org`). All additional data are available from the corresponding author upon reasonable request.

References

1. G. Duarte Ramos Matos, D. Y. Kyu, H. H. Loeffler, J. D. Chodera, M. R. Shirts, and D. L. Mobley, J. Chem. Eng. Data **62**, 1559 (2017).

2. G. M. Giambaşu, T. Luchko, D. Herschlag, D. M. York, and D. A. Case, Biophys. J. **106**, 883 (2014).

3. Z. Cournia, B. Allen, and W. Sherman, J. Chem. Inf. Model. **57**, 2911 (2017).

4. L. S. Ornstein and F. Zernike, Proc. Akad. Sci. (Amsterdam) **17**, 793 (1914).

5. R. Evans, Adv. Phys. **28**, 143 (1979).

6. Y. Liu, S. Zhao, and J. Wu, J. Chem. Theory Comput. **9**, 1896 (2013).

7. S. Zhao, R. Ramirez, R. Vuilleumier, and D. Borgis, J. Chem. Phys. **134**, 194102 (2011).

8. D. Beglov and B. Roux, J. Phys. Chem. B **101**, 7821 (1997).

9. A. Kovalenko and F. Hirata, J. Chem. Phys. **110**, 10095 (1999).

10. T. Luchko, S. Gusarov, D. R. Roe, C. Simmerling, D. A. Case, J. Tuszynski, and A. Kovalenko, J. Chem. Theory Comput. **6**, 607 (2010).

11. S. Gusarov, T. Ziegler, and A. Kovalenko, J. Phys. Chem. A **110**, 6083 (2006).

12. N. Yoshida and F. Hirata, J. Comput. Chem. **27**, 453 (2006).

13. T. Kloss, J. Heil, and S. M. Kast, J. Phys. Chem. B **112**, 4337 (2008).

14. T. Miyata and F. Hirata, J. Comput. Chem. **29**, 871 (2008).

15. J. Johnson, D. A. Case, T. Yamazaki, S. Gusarov, A. Kovalenko, and T. Luchko, J. Phys. Condens. Matter **28**, 344002 (2016).

16. D. S. Palmer, A. I. Frolov, E. L. Ratkova, and M. V. Fedorov, J. Phys. Condens. Matter **22**, 492101 (2010).

17. V. P. Sergiievskyi, G. Jeanmairet, M. Levesque, and D. Borgis, J. Phys. Chem. Lett. pp. 1935–1942 (2014).

18. J.-F. Truchon, B. M. Pettitt, and P. Labute, J. Chem. Theory Comput. **10**, 934 (2014).

19. J. Heil and S. M. Kast, J. Chem. Phys. **142**, 114107 (2015).

20. A. Kovalenko and F. Hirata, J. Chem. Phys. **112**, 10391 (2000).

21. J. W. Kaminski, S. Gusarov, T. A. Wesolowski, and A. Kovalenko, J. Phys. Chem. A **114**, 6082 (2010).

22. M. R. Shirts, D. L. Mobley, J. D. Chodera, and V. S. Pande, J. Phys. Chem. B **111**, 13052 (2007).

23. J. E. Barnes and P. Hut, Nature **324**, 446 (1986).

24. L. Greengard and V. Rokhlin, J. Comput. Phys. **73**, 325 (1987).

25. Z.-H. Duan and R. Krasny, J. Comput. Chem. **22**, 184 (2001).

26. Z.-H. Duan and R. Krasny, Proc. 2003 ACM Symp. Appl. Comput. pp. 172–177 (2003).

27. P. Li, H. Johnston, and R. Krasny, J. Comput. Phys. **228**, 3858 (2009).

28. H. A. Boateng and R. Krasny, J. Comput. Chem. **34**, 2159 (2013).

29. D. Case, I. Ben-Shalom, S. Brozell, D. Cerutti, T. C. III, V. Cruzeiro, T. Darden, R. Duke, D. Ghoreishi, G. Giambasu, et al., *AMBER 2019*, University of California, San Francisco (2019).

30. F. Hirata, in *Molecular Theory of Solvation*, edited by F. Hirata (Springer Netherlands, 2004), no. 24 in Understanding Chemical Reactivity, pp. 1–60.

31. A. Kovalenko, in *Molecular Theory of Solvation*, edited by F. Hirata (Springer Netherlands, 2004), no. 24 in Understanding Chemical Reactivity, pp. 169–275.

32. T. Luchko, I. S. Joung, and D. A. Case, in *Innovations in Biomolecular Modeling and Simulations: Volume 1*, edited by T. Schlick (Royal Society of Chemistry, 2012), vol. 23 of *RSC Biomolecular Sciences*, chap. 4, pp. 51–86.

33. F. Hirata and P. J. Rossky, Chem. Phys. Lett. **83**, 329 (1981).

34. J. Perkyns and B. M. Pettitt, J. Chem. Phys. **97**, 7656 (1992).

35. J. S. Perkyns, G. C. Lynch, J. J. Howard, and B. M. Pettitt, J. Chem. Phys. **132**, 064106 (2010).

36. A. Kovalenko and F. Hirata, J. Chem. Phys. **112**, 10403 (2000).

37. K.-C. Ng, J. Chem. Phys. **61**, 2680 (1974).

38. J. F. Springer, M. A. Pokrant, and F. A. S. Jr, J. Chem. Phys. **58**, 4863 (1973).

39. Z.-H. Duan and R. Krasny, J. Chem. Phys. **113**, 3492 (2000).

40. D. A. Case, D. S. Cerutti, T. E. Cheatham, III, T. A. Darden, R. E. Duke, T. J. Giese, H. Gohlke, A. W. Goetz, D. Greene, et al., *AMBER 2017*, University of California, San Francisco (2017).

41. N. M. O'Boyle, M. Banck, C. A. James, C. Morley, T. Vandermeersch, and G. R. Hutchison, J. Cheminformatics **3**, 33 (2011).

42. *The Open Babel Package, version 2.3.1*, URL http://openbabel.org.

43. J. Wang, R. M. Wolf, J. W. Caldwell, P. A. Kollman, and D. A. Case, J. Comput. Chem. **25**, 1157 (2004).

44. A. Jakalian, B. L. Bush, D. B. Jack, and C. I. Bayly, J. Comput. Chem. **21**, 132 (2000).

45. H. S. Muddana, A. T. Fenley, D. L. Mobley, and M. K. Gilson, J. Comput. Aided Mol. Des. **28**, 305 (2014).

46. C. I. Bayly, P. Cieplak, W. Cornell, and P. A. Kollman, J. Phys. Chem. **97**, 10269 (1993).

47. F.-Y. Dupradeau, A. Pigache, T. Zaffran, C. Savineau, R. Lelong, N. Grivel, D. Lelong, W. Rosanski, and P. Cieplak, Phys. Chem. Chem. Phys. **12**, 7821 (2010).

48. E. Vanquelef, S. Simon, G. Marquant, E. Garcia, G. Klimerak, J. C. Delepine, P. Cieplak, and F.-Y. Dupradeau, Nucleic Acids Res. **39**, W511 (2011).

49. F. Wang, J.-P. Becker, P. Cieplak, and F.-Y. Dupradeau, *R.E.D. Python: Object oriented programming for Amber force fields* (2013).

50. M. J. Frisch, G. W. Trucks, H. B. Schlegel, G. E. Scuseria, M. A. Robb, J. R. Cheeseman, G. Scalmani, V. Barone, G. A. Petersson, H. Nakatsuji, et al., *Gaussian 09*, Gaussian, Inc.

51. C. Tiede, A. A. S. Tang, S. E. Deacon, U. Mandal, J. E. Nettleship, R. L. Owen, S. E. George, D. J. Harrison, R. J. Owens, D. C. Tomlinson, et al., Protein Eng. Des. Sel. **27**, 145 (2014).

52. J. A. Maier, C. Martinez, K. Kasavajhala, L. Wickstrom, K. E. Hauser, and C. Simmerling, J. Chem. Theory Comput. **11**, 3696 (2015).

53. J. H. Nettles, H. Li, B. Cornett, J. M. Krahn, J. P. Snyder, and K. H. Downing, Science **305**, 866 (2004).

54. R. B. Ravelli, B. Gigant, P. A. Curmi, I. Jourdain, S. Lachkar, A. Sobel, and M. Knossow, Nature pp. 198–202 (2004).

55. A. Šali and T. L. Blundell, J. Mol. Biol. **234**, 779 (1993).

56. P. Li, B. P. Roberts, D. K. Chakravorty, and K. M. Merz, J. Chem. Theory Comput. **9**, 2733 (2013).

57. D. B. Wells and A. Aksimentiev, Biophysical Journal **99**, 629 (2010), ISSN 0006-3495, URL http://www.sciencedirect.com/science/article/pii/S0006349510005369.

58. F.-Y. Dupradeau, C. Cézard, R. Lelong, É. Stanislawiak, J. Pêcher, J. C. Delepine, and P. Cieplak, Nucleic Acids Research **36**, D360 (2008), ISSN 0305-1048, URL https://doi.org/10.1093/nar/gkm887.

59. H. J. C. Berendsen, J. R. Grigera, and T. P. Straatsma, J. Phys. Chem. **91**, 6269 (1987).

60. I. S. Joung and T. E. Cheatham, J. Phys. Chem. B **112**, 9020 (2008).

61. A. Kovalenko, S. Ten-no, and F. Hirata, J. Comput. Chem. **20**, 928 (1999).

62. *Intel Compilers. 19.1.0.53*, URL https://software.intel.com/en-us/compilers.

63. E. Gabriel, G. E. Fagg, G. Bosilca, T. Angskun, J. J. Dongarra, J. M. Squyres, V. Sahay, P. Kambadur, B. Barrett, A. Lumsdaine, et al., in *Recent Advances in Parallel Virtual Machine and Message Passing Interface*, edited by D. Kranzlmüller, P. Kacsuk, and J. Dongarra (Springer, Berlin, Heidelberg, 2004), Lecture Notes in Computer Science, pp. 97–104.

64. J. Towns, T. Cockerill, M. Dahan, I. Foster, K. Gaither, A. Grimshaw, V. Hazlewood, S. Lathrop, D. Lifka, G. D. Peterson, et al., Comput. Sci. Eng. **16**, 62 (2014).

65. N. Wilkins-Diehr, S. Sanielevici, J. Alameda, J. Cazes, L. Crosby, M. Pierce, and R. Roskies, in *High Performance Computer Applications: 6th International Conference, ISUM 2015, Mexico City, Mexico, March 9-13, 2015, Revised Selected Papers* (Springer, 2016), ISBN 978-3-319-32243-8, URL `http://link.springer.com/10.1007/978-3-319-32243-8`.

66. *Intel Compilers. 17.0.4*, URL `https://software.intel.com/en-us/compilers`.

67. *MVAPICH2, 2.3*, URL `https://mvapich.cse.ohio-state.edu/`.

68. J. G. Kirkwood and F. P. Buff, The Journal of Chemical Physics **19**, 774 (1951), ISSN 0021-9606, URL `https://aip.scitation.org/doi/abs/10.1063/1.1748352`.

69. J.-P. Hansen and I. R. McDonald, *Theory of Simple Liquids* (Elsevier, 2013), 4th ed.

70. M. Frigo and S. G. Johnson, Proc. IEEE **93**, 216 (2005).

71. B. Alberts, A. Johnson, J. Lewis, M. Raff, K. Roberts, and P. Walter, *Molecular biology of the cell* (Garland Science, 2002), 4th ed., ISBN 978-0-8153-3218-3 978-0-8153-4072-0.

72. P. E. Smith, The Journal of Physical Chemistry B **110**, 2862 (2006), ISSN 1520-6106, publisher: American Chemical Society, URL `https://doi.org/10.1021/jp056100e`.

73. S. Genheden, T. Luchko, S. Gusarov, A. Kovalenko, and U. Ryde, J. Phys. Chem. B **114**, 8505 (2010).

74. G. J. Brouhard and L. M. Rice, Nature reviews. Molecular cell biology **19**, 451 (2018), ISSN 1471-0072, URL `https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6019280/`.

75. M. Igaev and H. Grubmüller, PLOS Computational Biology **16**, e1008132 (2020), ISSN 1553-7358, publisher: Public Library of Science, URL `https://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1008132`.

76. M. Hemmat, B. T. Castle, J. N. Sachs, and D. J. Odde, Biophysical Journal **117**, 1234 (2019), ISSN 0006-3495, URL `https://www.sciencedirect.com/science/article/pii/S0006349519306976`.

77. M. Igaev and H. Grubmüller, eLife **7**, e34353 (2018), ISSN 2050-084X, URL `https://elifesciences.org/articles/34353`.

78. A. Grafmüller, E. G. Noya, and G. A. Voth, Journal of Molecular Biology **425**, 2232 (2013), ISSN 0022-2836, URL `https://www.sciencedirect.com/science/article/pii/S0022283613001927`.

79. J. R. André, M.-J. Clément, E. Adjadj, F. Toma, P. A. Curmi, and P. Manivet, Journal of Computer-Aided Molecular Design **26**, 397 (2012), ISSN 1573-4951, URL `https://doi.org/10.1007/s10822-012-9566-x`.

80. A. Grafmüller and G. A. Voth, Structure **19**, 409 (2011), ISSN 0969-2126, URL `https://www.sciencedirect.com/science/article/pii/S0969212611000311`.

81. M. J. Bennett, J. K. Chik, G. W. Slysz, T. Luchko, J. Tuszynski, D. L. Sackett, and D. C. Schriemer, Biochemistry **48**, 4858 (2009), ISSN 0006-2960, URL `http://dx.doi.org/10.1021/bi900200q`.

82. S. Chaaban, S. Jariwala, C.-T. Hsu, S. Redemann, J. M. Kollman, T. Müller-Reichert, D. Sept, K. H. Bui, and G. J. Brouhard, Developmental Cell **47**, 191 (2018), ISSN 1534-5807, URL `https://www.sciencedirect.com/science/article/pii/S1534580718306968`.