Commun. Comput. Phys.
doi: 10.4208/cicp.290711.181011s

Vol. **13**, No. 1, pp. 150-173
January 2013

An Adaptive, Finite Difference Solver for the Nonlinear Poisson-Boltzmann Equation with Applications to Biomolecular Computations

Mohammad Mirzadeh[1,*], Maxime Theillard[1], Asdís Helgadóttir[1], David Boy[1] and Frédéric Gibou[1,2]

[1] *Department of Mechanical Engineering, University of California, Santa Barbara, CA 93106, USA.*
[2] *Department of Computer Science, University of California, Santa Barbara, CA 93106, USA.*

Received 29 July 2011; Accepted (in revised version) 18 October 2011

Available online 12 June 2012

Abstract. We present a solver for the Poisson-Boltzmann equation and demonstrate its applicability for biomolecular electrostatics computation. The solver uses a level set framework to represent sharp, complex interfaces in a simple and robust manner. It also uses non-graded, adaptive octree grids which, in comparison to uniform grids, drastically decrease memory usage and runtime without sacrificing accuracy. The basic solver was introduced in earlier works [16, 27], and here is extended to address biomolecular systems. First, a novel approach of calculating the solvent excluded and the solvent accessible surfaces is explained; this allows to accurately represent the location of the molecule's surface. Next, a hybrid finite difference/finite volume approach is presented for discretizing the nonlinear Poisson-Boltzmann equation and enforcing the jump boundary conditions at the interface. Since the interface is implicitly represented by a level set function, imposing the jump boundary conditions is straightforward and efficient.

AMS subject classifications: 92-08, 65N06, 65N08, 65N50, 92C05, 35F21

Key words: Poisson-Boltzmann, non-graded adaptive grid, octree data structure, level set, irregular domain, hybrid finite volume/finite difference.

1 Introduction

The Poisson-Boltzmann equation is useful for calculating important biomolecular quantities, such as pKa values and energies of binding [12]. However, solving this equation

*Corresponding author. *Email addresses:* m.mirzadeh@engineering.ucsb.edu (M. Mirzadeh), maxime@engineering.ucsb.edu (M. Theillard), asdis@engineering.ucsb.edu (A. Helgadóttir), davidboy@engineering.ucsb.edu (D. Boy), fgibou@engineering.ucsb.edu (F. Gibou)

numerically has many challenges, the most significant of which are a) charge singularities, b) representing molecular surfaces, c) addressing exponential nonlinearities in the solution, and d) imposing the correct jump boundary condition. In this work, we present a solver that addresses some of the computational challenges in novel ways. First, we describe a simple and robust technique for implicitly representing biomolecular surfaces. Next, we demonstrate a novel discretization method for imposing the correct jump boundary conditions on the surface. Finally, we validate the solver and show its usefulness by calculating solvation free energies.

Since the pioneering work of Warwicker and Watson in the early 1980s [39], many different techniques for solving the Poisson-Boltzmann equation have been developed, most of which are based on finite difference, finite element, or boundary element methods. In this work, we do not intend to describe or compare them and refer the interested reader to [2,3,7,18,21] and the references therein for recent reviews.

An important characteristic of modern Poisson-Boltzmann solvers is the solver ability to use variable resolution. This allows to have coarse resolution where the solution is smooth and fine resolution where the solution varies rapidly. Indeed, one of the advantages of finite element methods over finite difference methods has been the robust adaptivity. Finite element methods are able to locally refine the computational mesh based on an error indicator, increasing resolution as needed, which enabled them to more efficiently address the exponential nonlinearity in the Poisson-Boltzmann equation [3,21]. Finite difference solvers can achieve similar results through the practice of focusing, in which the equation is solved on a coarse mesh, and the solution is used as a boundary condition for a finer mesh over an interesting subdomain [14].

Recent works have introduced adaptive finite difference methods that discretize the Poisson-Boltzmann equation on non-uniform grids. In [5], Boschitsch and Fenley introduced a first-order method that solves the nonlinear Poisson-Boltzmann equation on a graded octree mesh. In [27] and [16], Gibou and coworkers presented a second-order method for solving the Poisson-Boltzmann equation on a non-graded octree mesh. Neither use error estimates to refine the mesh, however. Instead, they refine the mesh based on distance from the molecular surface. The rational is based on the elliptic nature of the equation, which ensures that the solutions are smooth away from the interface.

Another difference between finite difference and finite element methods is that finite element methods ensure that cell edges align with interfaces. This is an appealing feature, as interfaces cutting through cells — as happens in finite difference schemes — complicate the discretization of boundary conditions. This property, however, comes at a price; creating a finite element mesh for a geometrically complicated domain, such as the surface of a protein, can be very expensive [8].

Finite difference methods, on the other hand, do not require the grid to conform to the boundary. As a result the grid generation, for uniform meshes, is trivial. However, since the grid does not conform to the boundary, special care must be taken to discretize the boundary conditions. This is specially important for biomolecular computations since one has to impose jump boundary conditions on complicated and, potentially, singular

geometries. Still finite difference solvers are quite popular and important improvements have been done over the years. Notably, Wei and coworkers have developed a method, termed "Matched Interface and Boundary" (MIB), that is able to produce second-order accurate results through accurately imposing the jump boundary conditions. For more information on this method, one may refer to [13, 40].

The work presented here builds directly on that of [27] and [16]. Those papers addressed colloidal systems and electrochemistry, and the solver did not address charge singularities. Therefore, it was not directly applicable to biomolecule studies. In Section 4, we show how to extend the method using the regularization scheme presented by Chern *et al.* [9], making it suitable for biomolecular computations.

2 Domain description using level set functions

There are multiple ways of defining a molecular surface. The simplest approach is to use the *van der Waals Surface* (vdWS), which represents a molecule with a set of intersecting spheres of radii r_i, where r_i is the van der Waals radius of the i-th atom in the molecule. This surface is not completely accessible to solvent molecules, though, and therefore not appropriate for implicit solvent models. To address this, one can use the *Solvent Accessible Surface* (SAS), which is the set of spheres with radii r_i+r_s, where r_s is the solvent radius [19]. The SAS is commonly used to represent the hydration effects. Unfortunately, both vdWS and SAS result in geometrical singularities due to self intersection between spheres. To remedy this problem, it is possible to use the *Solvent Excluded Surface* (SES), which are the boundary points that are in contact with a solvent molecule as the solvent molecule "rolls" over the vdWS [15, 33]. These three surfaces are schematically depicted in Fig. 1.

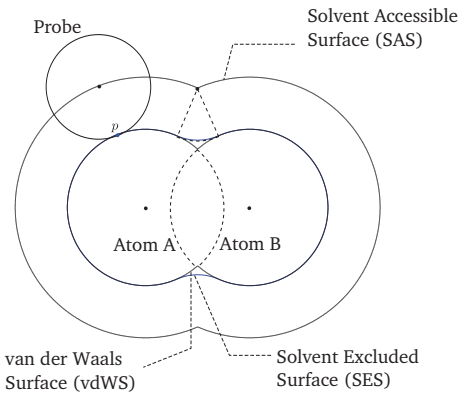


Figure 1: Common surfaces used in biomolecular computations.

Different methods have been proposed over the years to compute these surfaces. Connolly proposed an analytical algorithm for computing both the SAS and the SES [10, 11], and Sanner and Oslon described an algorithm for analytically computing the SES and

providing a triangulated representation of the surface [36]. Analytic representation of the SES can lead to very accurate computation of molecular surface and volume. However, this technique requires geometric singularities to be dealt with explicitly. Alternatively, by using an implicit representation of the molecular surface, one can potentially avoid to handle any, or most of, singularities explicitly.

Level set methods, originally proposed by Osher and Sethian in [30], are a general, robust, and flexible framework for implicitly representing and tracking interfaces that undergo complex topological changes. Applying this idea to biomolecules, a molecular surface — vdWS, SAS or SES — can be represented as the zero level set of a three dimensional function. More precisely, the level set function, $\phi(x,y,z)$, divides the domain, Ω, into two domains, Ω^+ and Ω^-, and the interface Γ,

$$\begin{cases} \Omega^+ \equiv \{\mathbf{x}\in \mathrm{R}^3 \,|\, \phi(x,y,z)>0\}, \\ \Gamma \equiv \{\mathbf{x}\in \mathrm{R}^3 \,|\, \phi(x,y,z)=0\}, \\ \Omega^- \equiv \{\mathbf{x}\in \mathrm{R}^3 \,|\, \phi(x,y,z)<0\}. \end{cases}$$

A large body of work has focused on computing level set functions for various geometries. Here we describe the essentials, and we refer the reader to [37] and [29] for a more thorough survey of the methods. To generate a level set function, one normally starts with an initial function that correctly predicts the location of boundary. In the case of the vdWS surface, for example, this initial function may simply be chosen as

$$\phi_0(x,y,z) = \max_i \left\{ r_i - \sqrt{(x-x_i)^2+(y-y_i)^2+(z-z_i)^2} \right\}.$$

However, it is not always possible to obtain a good initial function; for example, no such simple expression exists for the SES. Furthermore, to obtain a good adaptive grid and maintain robustness, it is required that the level set function be a distance function, i.e. $|\nabla\phi|=1$. To achieve this property, and once an initial level set function, ϕ_0, is chosen, the *reinitialization equation*,

$$\frac{\partial\phi}{\partial\tau} + S(\phi_0)(|\nabla\phi|-1) = 0, \tag{2.1}$$

must be solved where τ is a fictitious time and S is the numerical sign function, usually taken as

$$S(\phi_0) = \frac{\phi_0}{\sqrt{\phi_0^2+\Delta x^2}}.$$

Compared to traditional methods, using a level set function to represent the molecular surface is a relatively new technique that has been explored by many different authors. In [6] the authors described a level set method for capturing the SES by starting from the vdWS and moving the interface in the normal direction twice to obtain the SAS and SES. At the end of second pass, however, the SES may contain inaccessible cavities. These are removed by a third pass in which a sphere is constructed around the outer SES and

then "shrink-wrapped" to the molecule. Another approach obtains the SES by moving the boundary only once, but it must impose the correct curvature at toric segments [32]. Although defining the correct curvature seems reasonable, it leads to solving a nonlinear advection-diffusion equation describing the motion of level set, which is computationally expensive. Moreover, defining the correct curvature requires the definition of a reduced surface, leading to extra complication [36].

Alternatively, within the level set framework, it is relatively easy to "redefine" the molecular surface such that it minimizes the solvation free energy. Different approaches exists in the literature. For example, in [20] the authors obtain the molecular surface by evolving the level set function such that it minimizes a certain solvation free energy functional. In another article by Bates *e*t al. [31], the authors use similar ideas to obtain the *Minimal Molecular Surface* (MMS) obtained via mean curvature minimizations.

The method presented here to generate the level set function is similar to [6] in that we do not explicitly enforce the curvature and aim for the classical definition of SES. However, instead of moving the level set in the normal direction twice, which would be expensive, we only reinitialize the level set and note the following:

1. The reinitialization equation, (2.1), is closely related to moving the interface in the normal direction,
2. After reinitialization, the level set $\phi=\pm d$ is at the distance $\pm d$ away from the interface along the normal direction, and
3. Toric segments are automatically generated by the rarefaction waves when unit normal vectors diverge (see Fig. 2).

Within this new framework, generating the SES is quite simple and efficient. Our algorithm is the following:

1. Start with an approximation for the SAS by looping over all atoms in the molecule:

$$\phi_0(x,y,z)=\max_i\left\{r_i+r_s-\sqrt{(x-x_i)^2+(y-y_i)^2+(z-z_i)^2}\right\}. \tag{2.2}$$

2. Reinitialize the level set function ϕ' using Eq. (2.1) with ϕ_0 defined in Eq. (2.2) as the initial approximation.
3. Obtain the SES by taking the zero level set of $\phi=\phi'-r_s$.

Fig. 2 schematically illustrates the application of this algorithm. In the rest of this paper, we adopt the convention that Ω^+ refers to the inside of the molecule, Ω^- refers to the outside, and Γ represents the SES.

Finally we note that at the end of this algorithm, the SES may contain inner "cavities". To identify and remove these cavities, we incorporate a simple fix. The basic idea is based on the observation that inner cavities are not, topologically, "connected" to the

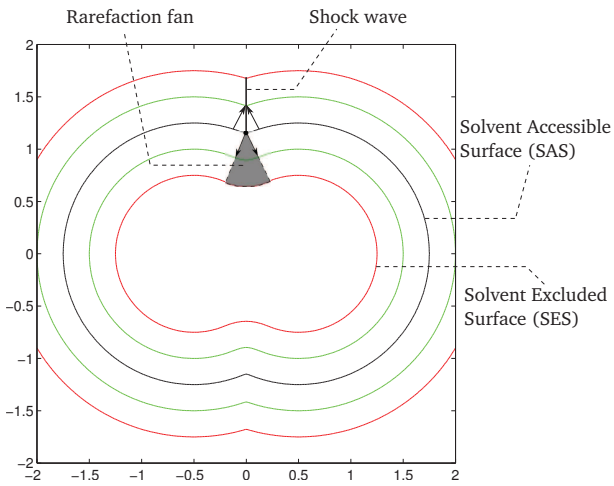
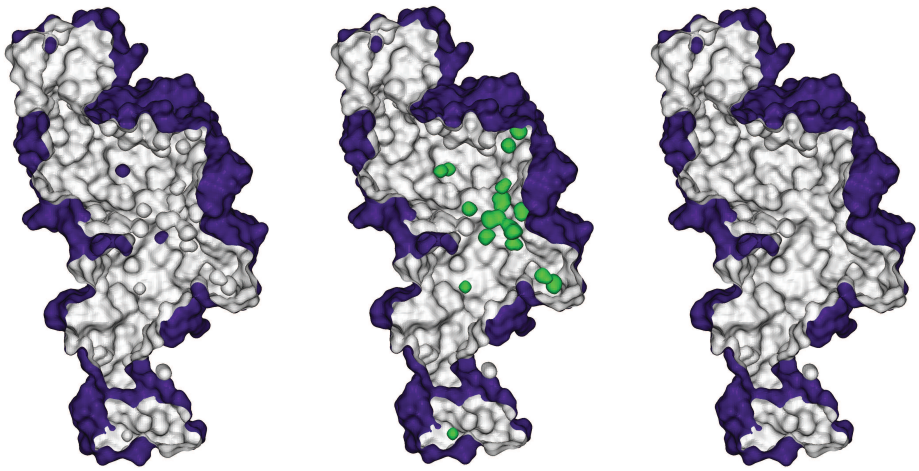


Figure 2: Schematic illustration of the SES generation algorithm. Once the SAS is reinitialized, rarefaction fan and shock waves propagate into the domain, depending on whether the normal vectors diverge or converge, respectively. This automatically ensures the formation of toric segments for all the level set contours in Ω^+, including the SES.

boundaries of the computational domain. As a result, any algorithm that can benefit from this observation, can detect the cavities and remove them. One such simple algorithm is to solve an auxiliary diffusion equation in the Ω^- domain subjected to a zero boundary condition on the SES surface and a nonzero boundary condition on the computational



(a) (b) (c)

Figure 3: Cavity removal process. (a) A cross-section of the SES generated for the 2C00 molecule with outside colored in purple and inside in grey. (b) Cavities, colored in green, are detected inside the molecule by checking if $\phi<0$ and $\mathcal{C}=0$ at each grid point. (c) Cavities are removed by simply changing the sign of the level set function at the corresponding grid points.

domain boundaries. Specifically, we solve:

$$\begin{aligned} \nabla^2 \mathcal{C} &= 0, \qquad \mathbf{x} \in \Omega^-, \\ \mathcal{C}(\mathbf{x}) &= 0, \qquad \mathbf{x} \in \Gamma, \\ \mathcal{C}(\mathbf{x}) &= b, \qquad \mathbf{x} \in \partial\Omega, \end{aligned}$$

where $\partial\Omega$ is the boundary of the computational domain, and b is a constant coefficient different from zero. It is clear that the solution of this problem, defined only in Ω^-, is equal to zero in the cavities, and is different from zero elsewhere. Once the solution is found, cavities are simply marked wherever $\phi < 0$ and $\mathcal{C} = 0$. The removal process simply consists of changing the sign of the level set function in the cavities. Note that a few iterations of the reinitialization equation may be required to avoid any discontinuity in the level set function due to changing the sign of the level set function inside the cavities. Fig. 3 illustrates the application of this algorithm in removing the cavities inside the 2C00 molecule.

3 Octree grid generation

Refining the mesh locally is often preferred to keeping a uniform grid. The solution to the Poisson-Boltzmann equation is smooth away from the interface, but due to boundary conditions and formation of the electric double layer, could have very large gradients near the interface. One way to address this problem is to introduce more grid cells near the interface. For three spatial dimensions, octree grids have been shown to be an optimal choice for local grid refinement [1].

When the refinement is performed near the interface, the number of grid points is proportional to the surface of the molecule rather than the volume of the computational domain. Moreover, for elliptic problems, the main factor determining the execution time and memory consumption is the size of the resulting linear system. As such, an octree-based method can be many times faster and memory-efficient than a uniform method. The computational advantage of adaptive grids is particularly great when there are jumps in the solution at the interface, since errors in the jump or its location will propagate into the entire domain. Adaptive grids are also excellent at resolving rapidly changing solutions, such as the electrostatic potential inside the electric double layer. Adaptive grids are therefore advantageous for solving the Poisson-Boltzmann equation with singular terms.

Octree data structures are described in detail in [34,35], and we present only the basics here. To construct an octree grid, we initialize a single cell covering the entire domain, and the grid is subsequently refined. A cell is first split into eight equally sized cubes. The larger cell is called a parent and the smaller cells are called the children. The difference in level between a parent and its children is one. Each child can then be split recursively as often as needed. For graded trees the difference of level between adjacent cells is at most one. Standard discretization schemes are easily ported to graded trees, but at

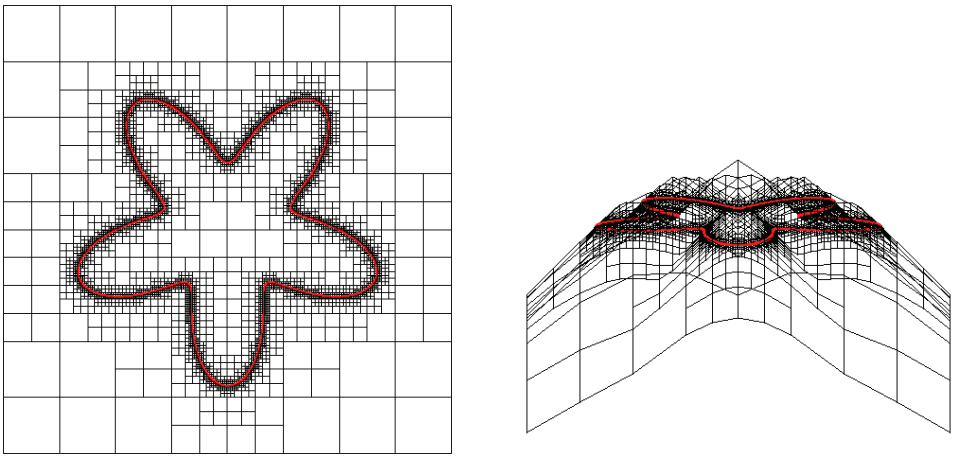


Figure 4: Left: An irregular domain with the adaptive mesh generated with (3.1). Right: A side view of the same domain representing the implicit function ϕ. The set of point $(x,y):\phi(x,y)>0$ defines the interior of the irregular domain.

the expense of efficiency and ease of grid generation. We use non-graded grids, which are more complicated to construct schemes for, but lead to significant reduction in the number of grid nodes. The splitting criteria for each grid point is given in [23, 25]. In summary, a cell, C, is split if

$$\min_{v\in \text{node}(C)} |\phi(v)| \leq \text{Lip}(\phi)\cdot \text{diagsize}(C), \tag{3.1}$$

where $\text{Lip}(\phi)$ is the Lipschitz constant for ϕ, $\text{node}(C)$ refers to the set of nodes of the cell C and $\text{diagsize}(C)$ is the length of the diagonal of C. The finest cells will be located on the interface where most accuracy is needed; the grid is locally uniform in a small band around the interface. We emphasize that the interface is captured and that the cells need not conform to the domain's boundary. Furthermore, the grid generation is straightforward, since the level set function is known. An example of the adaptive grid and associated level set function for two spatial dimensions is given in Fig. 4.

4 Governing equations

The electrostatic potential, Ψ, around a biomolecule in a symmetric, binary z:z electrolyte solution can be described by the nonlinear Poisson-Boltzmann (PB) equation,

$$-\nabla\cdot(\epsilon\epsilon_0\nabla\Psi)+2c^b(\mathbf{x})ez\sinh(\Psi)=\sum_{i=1}^{N_m} q_i\delta(\mathbf{x}-\mathbf{x}_i),$$

where ϵ is the relative permittivity of the electrolyte, ϵ_0 is the permittivity of a vacuum, e is the charge of a proton, z is the valence of the background electrolyte, c^b is the bulk salt

concentration, k_B is the Boltzmann coefficient, T is absolute temperature, q_i is the atomic partial charge, $\mathbf{x}_i$ is the location of each individual atom and N_m is the number of atoms in the molecule. Note that the explicit dependence of bulk salt concentration on position is only to indicate that mobile ions only exist in the solution; that is, $c^b(\mathbf{x})=0$ inside the molecule. In non-dimensional form, the Poisson-Boltzmann may be written as

$$-\nabla\cdot(\epsilon\nabla\psi)+\kappa^2(\mathbf{x})\sinh(\psi)=\sum_{i=1}^{N_m} z_i\delta(\mathbf{x}-\mathbf{x}_i), \tag{4.1}$$

where the potential has been scaled to the thermal voltage, z_i is the non-dimensional partial charge on the atoms, and κ is the non-dimensional inverse of Debye length. Inside the molecule, $\kappa=0$. Eq. (4.1) is accompanied with jump conditions at the molecular surface; that is, we require that

$$[\psi]_\Gamma=0, \qquad [\epsilon\nabla\psi\cdot\mathbf{n}]_\Gamma=0. \tag{4.2}$$

4.1 Technique for representing singular charges

Finite difference methods typically use a Dirac delta function to map discrete charges onto the grid. This method can obtain second-order accuracy, but comes with challenges. First, if the mesh is too coarse, then charges near the molecular interface may actually be smeared out so much as to extend outside the molecular interface. Geng *et al.* found this effect to significantly reduce the accuracy of their second-order MIB solver [13]. Our adaptive meshing technique could address this by introducing additional grid points, but that would increase the computational cost.

Instead, as in Geng *et al.* [13], we choose to use a formulation that allows us to represent the singular charges through their effect on the molecular interface. This technique, introduced by Chern *et al.* [9], is to separate the singular part of the solution, associated with the discrete charges in the molecule, from the regular part. This introduces a modified jump condition at the interface, which differs from Eq. (4.2) and is discussed in details in the appendix.

4.2 Newton's method to address nonlinearity

Another challenge that must be dealt with, is the existence of the nonlinear term. We have previously addressed this issue in [27] using a Newton's iteration method and here the same approach is repeated. Considering a series of solutions ψ^ν, and starting with an initial guess ψ^0, a Taylor series expansion for the nonlinear term is utilized to locally linearize the Poisson-Boltzmann equation as:

$$\sinh\left(\hat{\psi}^{\nu+1}\right)\approx\sinh\left(\hat{\psi}^{\nu}\right)+\left(\hat{\psi}^{\nu+1}-\hat{\psi}^{\nu}\right)\cosh\left(\hat{\psi}^{\nu}\right).$$

Using this linearization, the Poisson-Boltzmann equation becomes,

$$\kappa^2(\mathbf{x})\cosh\left(\hat{\psi}^{\nu}\right)\hat{\psi}^{\nu+1}-\nabla\cdot\left(\epsilon\nabla\hat{\psi}^{\nu+1}\right)=-\kappa^2(\mathbf{x})\sinh\left(\hat{\psi}^{\nu}\right)+\kappa^2(\mathbf{x})\hat{\psi}^{\nu}\cosh\left(\hat{\psi}^{\nu}\right)+f,$$

where the source term, f, represents the singular terms and is not affected by the iteration. Moreover, since the boundary conditions are linear, they are simply applied at each iteration step $\nu+1$. The iteration scheme is carried on until the difference in the solution reaches a given tolerance, i.e. we require,

$$\left\|\psi^{\nu+1}-\psi^{\nu}\right\|_{L_\infty}<10^{-6}.$$

5 Spatial discretization

The main difficulty in deriving numerical methods for adaptive Cartesian meshes is addressing T-junctions accurately (see Fig. 5). In [26], the authors showed that second-order accurate discretizations can be obtained on highly non-graded Cartesian meshes by compensating numerical error in one or two spatial directions with the derivative in the transverse direction. Furthermore, they showed that such a discretization is always possible in the case where the solution is sampled at the nodes of each cell. We do not go into the details of discretizing the Poisson-Boltzmann equation on octree grids and refer the interested reader to [27] and [16], where the authors detail these ideas and demonstrate second-order accuracy.

Away from the interface, this finite difference method is utilized to discretize Eq. (A.1). Close to interfaces we use a finite volume approach to handle the jump conditions in Eq. (A.2). This approach is presented next. For clarity, we present the jump for the Poisson-Boltzmann equation in two spatial dimensions, but extending this technique to three spatial dimensions is straightforward.

5.1 Discretization near the interface

The Poisson-Boltzmann equation with variable coefficient, ϵ, and jump conditions can be written as follows:

$$\begin{aligned}
&-\nabla\cdot\left(\epsilon\nabla\hat{\psi}\right)+\kappa^2(\mathbf{x})\sinh\left(\hat{\psi}\right)=f(\mathbf{x}), && \mathbf{x}\in\Omega,\\
&\left[\hat{\psi}\right]=a(\mathbf{x}), && \mathbf{x}\in\Gamma,\\
&\left[\epsilon\nabla\hat{\psi}\cdot\mathbf{n}\right]=b(\mathbf{x}), && \mathbf{x}\in\Gamma,
\end{aligned}$$

where $\hat{\psi}$ is a scalar to be solved for, $f(\mathbf{x})$, $a(\mathbf{x})$ and $b(\mathbf{x})$ are known scalar functions, and $\left[\hat{\psi}\right]$ and $\left[\epsilon\nabla\hat{\psi}\cdot\mathbf{n}\right]$ are the jump in the electrostatic potential, $\hat{\psi}$, and the normal component of the electric displacement field, $D=-\epsilon\nabla\hat{\psi}$, respectively. As previously described, Ω is the whole domain that is split into two subdomains Ω^- and Ω^+ by the interface Γ.

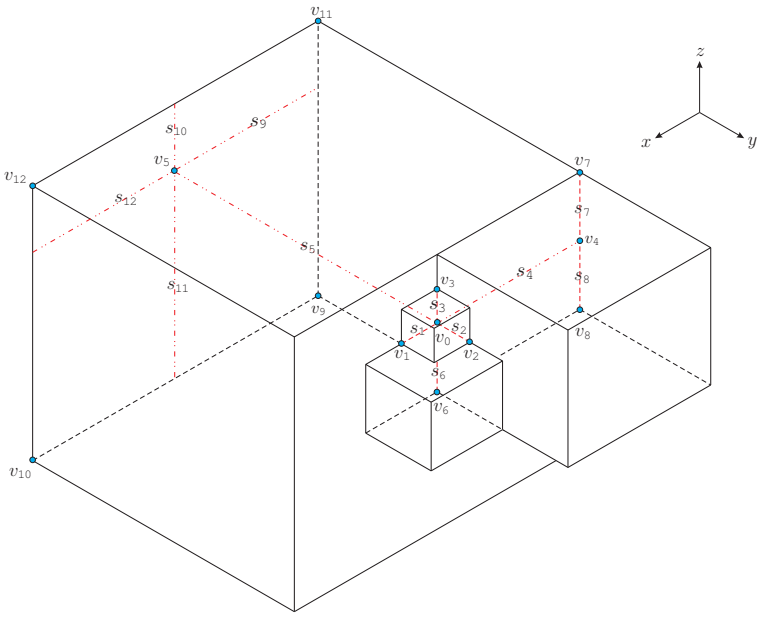


Figure 5: The general configuration for non-graded octree cells. A node in the grid is said to be uniform if it is directly connected to other nodes in each of six directions. Alternatively a node is said to have a *T-junction* if it does not have a direct neighbor in at least one of the six possible directions. A node can have at most one three dimensional and one two dimensional T-junctions.

Variables are continuous inside each domain, but ϵ, $\hat{\psi}$ and f can be discontinuous across the interface. The grid configuration near the interface is depicted in Fig. 6. Integrating inside both Ω^- and Ω^+ over the cell surrounding node (i,j), denoted by $C_{i,j}$, leads to the

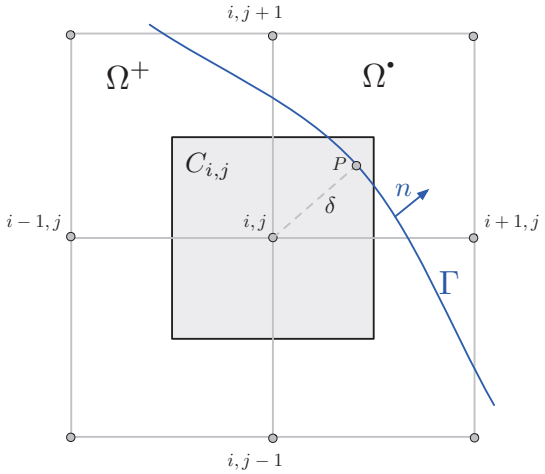


Figure 6: Cell $C_{i,j}$ surrounding the node located at i,j. The point P is the point on the interface Γ that is closest to the node.

following equations:

$$-\int_{C_{i,j}\cap\Omega^-}\nabla\cdot\left(\epsilon^-\nabla\hat{\psi}^-\right)d\mathcal{A}+\int_{C_{i,j}\cap\Omega^-}\kappa^2(\mathbf{x})\sinh\left(\hat{\psi}^-\right)d\mathcal{A}=\int_{C_{i,j}\cap\Omega^-}f^-d\mathcal{A},$$
$$-\int_{C_{i,j}\cap\Omega^+}\nabla\cdot\left(\epsilon^+\nabla\hat{\psi}^+\right)d\mathcal{A}+\int_{C_{i,j}\cap\Omega^+}\kappa^2(\mathbf{x})\sinh\left(\hat{\psi}^+\right)d\mathcal{A}=\int_{C_{i,j}\cap\Omega^+}f^+d\mathcal{A},$$

where superscript $^+$ represents quantities in Ω^+ and superscript $^-$ represents quantities in Ω^-. Since both equations can be treated in the same way, we only describe the equation in Ω^+ in more detail. By using the divergence theorem on the left-hand side we get the following:

$$-\int_{\partial\left(C_{i,j}\cap\Omega^+\right)}\epsilon^+\nabla\hat{\psi}^+\cdot\mathbf{n}d\ell+\int_{C_{i,j}\cap\Omega^+}\kappa^2(\mathbf{x})\sinh\left(\hat{\psi}^+\right)d\mathcal{A}=\int_{C_{i,j}\cap\Omega^+}f^+d\mathcal{A}.$$

The first term in the previous equation can be split into two integrals: the integral over the cell faces inside Ω^+ and the integral over the interface Γ inside the cell. The integration over the cell faces inside Ω^+ is easily approximated numerically using length fractions as demonstrated in [28]. For instance, the integration over the right face is discretized as

$$-\int_{L_R}\epsilon^+\nabla\hat{\psi}^+\cdot\mathbf{n}d\ell=L_{i+1/2,j}\cdot\epsilon^+\cdot\frac{\hat{\psi}^+_{i+1,j}-\hat{\psi}^+_{i,j}}{\Delta x}+\mathcal{O}\left(\Delta x^3\right).$$

Accurately integrating over the interface inside the cell is more complicated. This is because, even though the normal to the interface is easily found from the level set function, we do not know the normal derivative of the solution to the interface. To overcome this challenge, one can take the following steps. As shown in Fig. 6, we call point P to be the projection of node (i,j) on the interface Γ, and δ its distance to the interface. Using a first-order Taylor expansion, the jump in the normal derivative, at point P, may be written as:

$$\begin{aligned}\frac{\partial\hat{\psi}^+}{\partial n}-\frac{\partial\hat{\psi}^-}{\partial n}&=\frac{\hat{\psi}^+_P-\hat{\psi}^+_{i,j}}{\delta}-\frac{\hat{\psi}^-_P-\hat{\psi}^-_{i,j}}{\delta}+\mathcal{O}(\Delta x)\\&=\frac{a_P}{\delta}-\frac{\hat{\psi}^+_{i,j}-\hat{\psi}^-_{i,j}}{\delta}+\mathcal{O}(\Delta x),\end{aligned}\tag{5.1}$$

where a_p denotes the value of the a at node P. Also, the jump in the normal component of electric displacement field, $[\epsilon\nabla\hat{\psi}\cdot n]$, at node P can be expressed as:

$$\epsilon^+\frac{\partial\hat{\psi}^+}{\partial n}-\epsilon^-\frac{\partial\hat{\psi}^-}{\partial n}=b_P.\tag{5.2}$$

Eqs. (5.1) and (5.2), are two equations for the two unknowns, $\partial\hat{\psi}^+/\partial n$ and $\partial\hat{\psi}^-/\partial n$, and yield

$$\frac{\partial\hat{\psi}^+}{\partial n}=\frac{1}{\delta(\epsilon^--\epsilon^+)}\left(-\delta b_P+\epsilon^-\left(a_P-\left(\hat{\psi}^+_{i,j}-\hat{\psi}^-_{i,j}\right)\right)\right)+\mathcal{O}(\Delta x),\tag{5.3}$$

$$\frac{\partial\hat{\psi}^-}{\partial n}=\frac{1}{\delta(\epsilon^--\epsilon^+)}\left(-\delta b_P+\epsilon^+\left(a_P-\left(\hat{\psi}^+_{i,j}-\hat{\psi}^-_{i,j}\right)\right)\right)+\mathcal{O}(\Delta x).\tag{5.4}$$

Once the normal derivatives are found, integration over the interface is approximated as

$$-\int_{\Gamma\cap C_{i,j}} \epsilon^+ \frac{\partial\hat{\psi}^+}{\partial n} = |\Gamma| \frac{\epsilon^+}{\delta(\epsilon^- - \epsilon^+)} \left(\delta b_P + \epsilon^- \left(\hat{\psi}^+_{i,j} - \hat{\psi}^-_{i,j} - a_P\right)\right) + \mathcal{O}(\Delta x^2), \tag{5.5}$$

$$-\int_{\Gamma\cap C_{i,j}} \epsilon^- \frac{\partial\hat{\psi}^-}{\partial n} = |\Gamma| \frac{\epsilon^-}{\delta(\epsilon^- - \epsilon^+)} \left(-\delta b_P - \epsilon^+ \left(\hat{\psi}^-_{i,j} - \hat{\psi}^+_{i,j} - a_P\right)\right) + \mathcal{O}(\Delta x^2). \tag{5.6}$$

This sharp, first-order discretization leads to a system that has twice as many equations as computational nodes near the interface to account for the jump in the solution and its gradient. It is, however, symmetric positive definite and only involves first-degree neighbors. A major advantage of the scheme is that large differences in ϵ^+ and ϵ^- do not adversely affect the accuracy.

When combined with the finite differences approach described previously, the resulting linear system is symmetric for uniform grids and an invertible M-matrix for octree grids.

6 Numerical examples

In this part we provide numerical examples to support the accuracy and convergence of our method. The first example is simply intended to show that our level set method can accurately represent the complicated surface of complex proteins. The next two examples are pure mathematical examples built to demonstrate the accuracy of our method in imposing the jump conditions as described in Section 5. Next, we consider the Kirkwood's dielectric sphere problem [17], a physically meaningful problem that has an analytic solution. In Example 6.5, the solvation free energy is computed and the results are compared with the APBS software for a select number of proteins [4]. Finally, we conclude this section by solving the electrostatic potential on a DNA (1D65) molecule.

6.1 Surface accuracy

In this section, we will shortly comment on the accuracy of our method for generating the biomolecular surfaces. To do this, we consider two different studies. First, we consider a "simple" molecule made of three atoms of radii 2 Å, placed on the vortices of an equilateral triangle with the inter-atomic distance of 3 Å. The SES for this molecule is then generated using our new approach and compared with both analytical and triangulated representations obtained via the MSMS software [36] with the probe radius of 1.5 Å. The triangulated surface is generated with the vertex density of 100.

Table 1 illustrates the accuracy analysis of this test when the grid is refined. We consider two measures to evaluate the accuracy. The forth column depicts the maximum distance of the triangulated surface vortices to the zero level set, i.e our representation of the SES. This is a "local" accuracy measure. The second measure is the comparison between the surface area of the zero level set, denote as A_{LS}, and the exact analytical area

Table 1: Point-wise accuracy analysis of SES generation for a 3-atom molecule.

$(res_{\max}, res_{\min})$	# of Points	$\Delta x_{\min}$	$\lvert\phi_{TS}\rvert_{\max}$	Order	$\lvert A_{LS}-A_{Ex}\rvert / A_{Ex}$	Order
(32, 8)	6 393	0.47	1.12×10^{-1}	–	2.00×10^{-2}	–
(64, 16)	46 601	0.23	5.98×10^{-2}	0.91	6.06×10^{-3}	1.80
(128, 32)	353 025	0.12	3.15×10^{-2}	0.92	1.79×10^{-3}	1.76
(256, 64)	2 746 817	0.06	1.58×10^{-2}	1.00	6.24×10^{-4}	1.52
(512, 128)	21 668 481	0.03	7.92×10^{-3}	1.00	2.67×10^{-4}	1.22

Table 2: Comparison of total surface area between the present method and the MSMS software.

Protein ID	# of Atoms	Analytical	Triangulated	Present work	# of Points	$\Delta x_{\min}$
1AJJ	513	2 112.84	2 013.42	2 131.66	2 075 015	0.101
6RXN	667	2 341.41	2 226.35	2 350.71	2 488 633	0.097
2ERR	1 638	5 189.28	4 917.44	5 074.78	1 373 760	0.190
1AA2	1 755	4 891.97	4 749.54	4 826.41	2 091 216	0.151
2X6A	4 294	13 612.07	12 975.43	13 174.53	1 295 583	0.313
2TEC	4 936	10 568.21	10 072.36	10 187.27	2 253 537	0.210

computed via MSMS software, denoted as A_{Ex}. This is a more global measure. It is easy to see that the new approach can generate accurate representation of the SES when compared to the MSMS software and the error in the local measure decays linearly with grid spacing. We use the second-order accurate geometric approach of Min and Gibou [24] to compute the areas.

As the second test, we choose a set of proteins, compute the total surface area of the SES, and compare our results to those obtained through the MSMS software. For these tests, a probe radius of 1.5 Å has been chosen and all level set functions have been obtained on an adaptive grid with $(res_{\max}, res_{\min})=(512,32)$ while leaving parameters of the MSMS software to defaults. Table 2 illustrates the accuracy of our method by comparing surface area calculation results (in $Å^2$) with those of the MSMS software (triangulated) and analytical calculations (analytic). The results obtained here indicate that level set method can be used to easily generate accurate molecular surfaces without the need for explicit handling of geometric singularities.

6.2 Sphere example

In this example, we consider a spherical interface in three spatial dimensions. We take the exact solution to be $\psi^+(x,y,z)=x^3+y^3+z^3$ and $\psi^-(x,y,z)=-1-x^3-y^3-z^3$, where $\epsilon^+=2$ and $\epsilon^-=80$. The radius of the sphere is 1 and the domain is $[-2,2]^3$. Convergence results given in Table 3, indicate that our method is first-order accurate in L^1, L^2 and L^∞ norms. The resulting numerical error on the different grids used in this analysis is shown in Fig. 7.

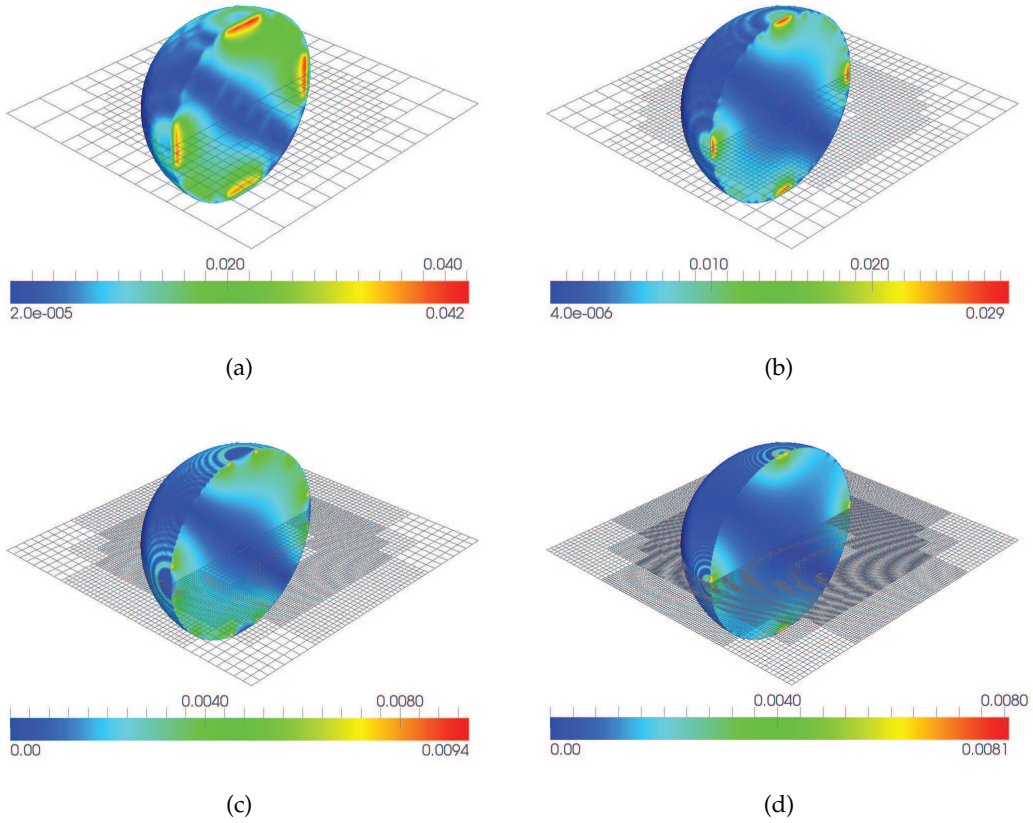


Figure 7: L_∞ error plotted for Example 6.2 on the different octree grids considered in the convergence analysis. Maximum grid resolution in each of figures are (a): $res_{\max}=64$, (b): $res_{\max}=128$, (c): $res_{\max}=256$, (d): $res_{\max}=512$.

Table 3: Convergence analysis for Example 6.2.

$res_{\max}$	Grid points	L^1 Error	Order	L^2 Error	Order	L^∞ Error	Order
32	12 739	5.98×10^{-3}	-	8.71×10^{-3}	-	4.22×10^{-2}	-
64	92 965	2.00×10^{-3}	1.58	3.20×10^{-3}	1.44	2.85×10^{-2}	0.57
128	708 745	7.33×10^{-4}	1.45	1.29×10^{-3}	1.31	9.42×10^{-3}	1.60
256	5 531 665	3.27×10^{-4}	1.16	6.64×10^{-4}	0.96	8.08×10^{-3}	0.22

6.3 Biomolecule

In this example, we consider the same parameters as in the previous example, except that now the interface is the 2ERR molecule. Convergence results are given in Table 4. Our approach still appears to be first-order in L^∞ norm, but second-order accurate in L^1 and L^2 norms. The numerical error is represented in Fig. 8.

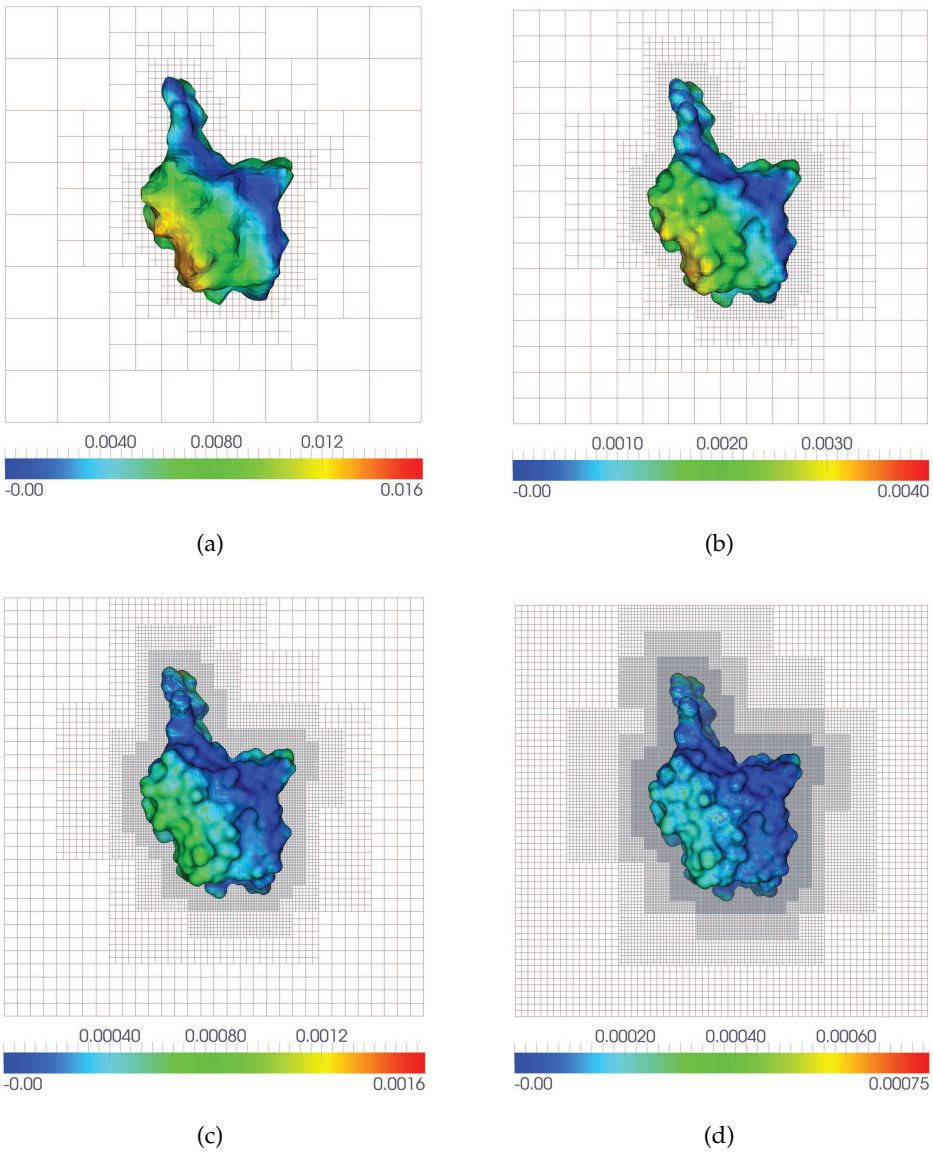


Figure 8: L_∞ error plotted for Example 6.3 on the different octree grids considered in the convergence analysis. Grid resolutions of each subfigure are (a): $res_{\max}=64$, (b): $res_{\max}=128$, (c): $res_{\max}=256$, (d): $res_{\max}=512$.

Table 4: Convergence analysis for Example 6.3.

$res_{\max}$	Grid points	L^1 Error	Order	L^2 Error	Order	L^∞ Error	Order
64	19 602	6.00×10^{-3}	-	7.43×10^{-3}	-	2.11×10^{-2}	-
128	146 087	1.41×10^{-3}	2.09	1.73×10^{-3}	2.10	7.2×10^{-3}	1.55
256	1 124 793	3.37×10^{-4}	2.06	4.15×10^{-4}	2.06	5.69×10^{-3}	0.34
512	8 871 297	8.76×10^{-5}	1.94	1.11×10^{-4}	1.90	3.13×10^{-3}	0.86

6.4 Kirkwood's solution

Here we consider a more physically relevant example, the Kirkwood's dielectric sphere. We refer the reader to the appendix for more details about this solution and its construction. The Kirkwood sphere chosen here has the radius of 30 Å, a shell thickness of 3 Å and one negative charge in the middle of the sphere. Moreover, a 1 mM electrolyte was chosen with $\epsilon^+ = 2$ and $\epsilon^- = 80$. The convergence results provided in Table 5, clearly demonstrate that our method is at least first-order. The same analysis is also performed close to the boundary. The results, presented in Table 6, suggest that indeed the maximum error appears near the boundary, where the jump conditions are imposed, and thus the idea of grid refinement near the boundary is justified.

Table 5: Convergence analysis for the Kirkwood's solution in Example 6.4 in the whole domain.

$res_{\max}$	Grid points	L^1 Error	Order	L^2 Error	Order	L^∞ Error	Order
64	18 747	5.61×10^{-3}	-	1.13×10^{-2}	-	5.02×10^{-2}	-
128	138 005	1.09×10^{-3}	2.36	2.72×10^{-3}	2.05	1.09×10^{-2}	2.20
256	1 057 257	3.11×10^{-4}	1.81	8.83×10^{-4}	1.62	3.47×10^{-3}	1.65
512	8 273 105	1.01×10^{-4}	1.62	3.32×10^{-4}	1.41	1.23×10^{-3}	1.49

Table 6: Convergence analysis for the Kirkwood's solution in Example 6.4 near the boundary.

$res_{\max}$	Grid points	L^1 Error	Order	L^2 Error	Order	L^∞ Error	Order
64	18 747	4.26×10^{-2}	-	4.32×10^{-2}	-	5.02×10^{-2}	-
128	138 005	8.31×10^{-3}	2.36	8.43×10^{-3}	2.34	1.09×10^{-2}	2.20
256	1 057 257	2.37×10^{-3}	1.81	2.44×10^{-3}	1.78	3.47×10^{-3}	1.65
512	8 273 105	7.68×10^{-4}	1.63	8.06×10^{-4}	1.60	1.23×10^{-3}	1.49

6.5 Solvation free energy

In this section, we provide numerical examples that illustrate the accuracy of our method in computing the electrostatic solvation free energy of certain proteins. One may refer

Table 7: Solvation free energy.

Protein ID	# of Atoms	ΔG^{sol}	$\Delta x_{\min}$	ΔG^{sol} (APBS)	Δx (APBS)	Rel. difference
1AJJ	513	-2.234×10^{3}	0.14	-2.228×10^{3}	0.52	1.23×10^{-1}
6RXN	667	-2.274×10^{3}	0.14	-2.313×10^{3}	0.51	1.71×10^{-2}
2ERR	1 638	-3.859×10^{3}	0.27	-4.016×10^{3}	0.56	4.00×10^{-2}
1AA2	1 755	-3.233×10^{3}	0.19	-3.327×10^{3}	0.52	2.87×10^{-2}
2X6A	4 294	-5.388×10^{3}	0.44	-5.699×10^{3}	0.71	5.61×10^{-2}
2TEC	4 936	-4.227×10^{3}	0.30	-4.486×10^{3}	0.71	5.96×10^{-2}

to the appendix for details on the method used in computing the free energy. To compare our results, we have used the APBS software to solve the same problem. In all of our calculations we have used an adaptive octree grid with $(res_{\max}, res_{\min}) = (512, 32)$. As for physical parameters, we set the bulk concentration to 10 mM and the dielectric coefficients to $\epsilon^+ = 2$ and $\epsilon^- = 78.3$ for the molecule and electrolyte, respectively. Table 7 illustrates the computed solvation free energy, using Eq. (C.1), in kJ/mol and compares the results with energies computed using APBS software. It is easily seen that the presented method is capable of producing accurate results even for complicated proteins.

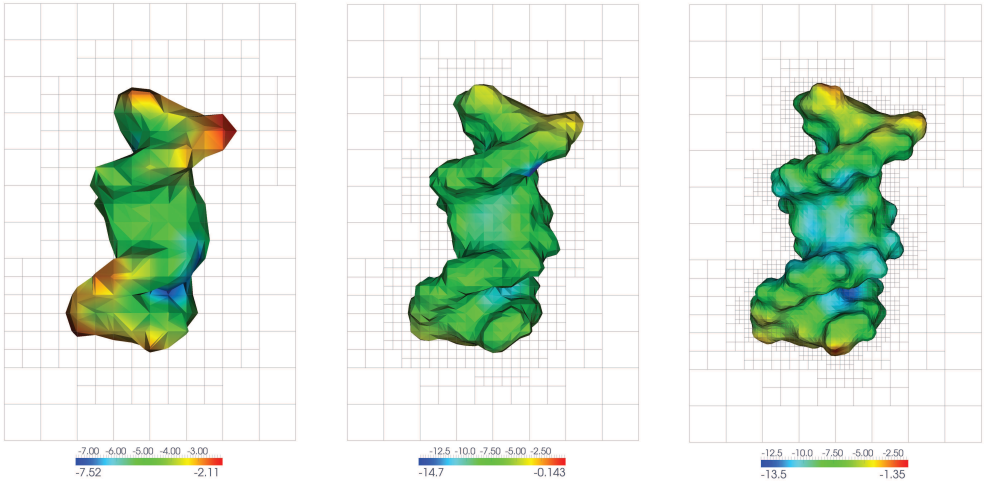
6.6 Application: Electrostatic potential on a DNA strand

In this last section we illustrate the results of a Poisson-Boltzmann computation for a DNA strand (1D65). Fig. 9 illustrates the electrostatic potential, in units of thermal voltage, on the surface of the molecule. It is interesting to note how the shape of the protein and the electrostatic potential are affected as the grid is refined. As shown in Fig. 9, coarser grids can only capture the overall shape of the protein whereas more details are only obtained on octree grids with higher resolutions. This, indeed, is a good example that illustrates certain levels of accuracy are only attainable with very high levels of refinement. One should note that this level of resolution ($res_{\max} = 1\,024$ for level 10) is only feasible on an adaptive grid. Where a uniform grid of the same maximum resolution would require about one billion grid points, this calculation was made on an adaptive octree grid with only about four million grid points ($\sim 0.4\%$).

7 Conclusion

In this paper, we have incorporated the idea of level set methods as a central framework for developing Poisson-Boltzmann solvers for biomolecular electrostatics computation on non-graded adaptive Cartesian grids. By using a Green's function formulation for singular charges and enforcing jump conditions at interfaces, the solver can efficiently handle singular charges. This is an extension of our earlier Poisson-Boltzmann solver, which was designed for simulating supercapacitors and colloids. Because the scheme can now address singular charges, it is suitable for simulating biomolecular electrostatics.

The solver is validated and its accuracy in computing the correct SES, electrostatic potential and solvation free energy is measured. Convergence tests suggest it is first-order in the L^∞ norm but close to second-order accurate in L^1 and L^2 norms. Due to the solver's adaptivity and rapid mesh-generation, it is suitable for simulating small biomolecules on desktop workstations. Additionally, the finite-difference discretization of the Poisson-Boltzmann equation is straightforward to implement and very robust. Lastly, we believe that other researchers in this field will find that implicit geometry representations and capturing methods provide a robust framework.



(a) Level 5, $res_{\max} = 32$ (b) Level 6, $res_{\max} = 64$ (c) Level 7, $res_{\max} = 128$

(d) Level 8, $res_{\max} = 256$ (e) Level 9, $res_{\max} = 512$ (f) Level 10, $res_{\max} = 1\,024$

Figure 9: The electrostatic potential on a DNA strand for different levels of refinement.

Acknowledgments

The research was supported in part by the W.M. Keck Foundation, by the Institute for Collaborative Biotechnologies through contract no. W911NF-09-D-0001 from the U.S. Army Research Office, by ONR under grant agreement N00014-11-1-0027, by the National Science Foundation under grant agreement CHE 1027817 and by the Department of Energy under grant agreement DE-FG02-08ER15991. The authors would also like to thank the constructive comments and suggestions from the reviewers.

Appendix

A Technique for handling singular charges

Here the technique introduced by Chern *et al.* [9], treating singularity as a jump condition, is described in detail. First, the solution to the electrostatic potential, ψ, is split into regular, $\hat{\psi}$, and singular,$\bar{\psi}$, parts:

$$\psi=\hat{\psi}+\bar{\psi}.$$

The singular part of the potential is defined such that

$$\bar{\psi}=\begin{cases}\psi^*+\psi^0, & \text{if } \mathbf{x}\in\Omega^+,\\ 0, & \text{if } \mathbf{x}\in\Omega^-,\end{cases}$$

where ψ^* is the Coulombic potential due to singular charges,

$$\psi^*=\sum_{i=1}^{N_m}\frac{z_i}{4\pi\epsilon^+}\frac{1}{|\mathbf{x}-\mathbf{x}_i|},$$

and ψ^0 fulfills:

$$\begin{aligned}&\nabla^2\psi^0=0, && \text{if } \mathbf{x}\in\Omega^+,\\ &\psi^0=-\psi^*, && \text{on } \Gamma.\end{aligned}$$

Using this decomposition, the regular part of the solution may be obtained by solving

$$-\nabla\cdot\left(\epsilon\nabla\hat{\psi}\right)+\kappa^2(\mathbf{x})\sinh\left(\hat{\psi}\right)=0, \tag{A.1}$$

subjected to the following modified jump conditions:

$$\left[\hat{\psi}\right]_\Gamma=0, \qquad \left[\epsilon\nabla\hat{\psi}\cdot\mathbf{n}\right]_\Gamma=-\epsilon^+\nabla\left(\psi^*+\psi^0\right)\cdot\mathbf{n}|_\Gamma. \tag{A.2}$$

B Kirkwood's dielectric sphere

To test the accuracy of our solutions, we use the Kirkwood dielectric sphere, as presented in [17]. Consider an ionic solution with dielectric constant ϵ^-. In the electrolyte, a sphere of radius b is placed with dielectric constant ϵ^+, which we take to be unity. Inside the sphere, we consider M discrete point charges, $q_1,\cdots,q_M$. Using a polar coordinate system, with the origin at the center of the sphere, the solution to the electrostatic potential, for $r<b$, is given by

$$\begin{aligned}V_1&=\sum_{k=1}^{M}\frac{q_k}{4\pi\epsilon^+|\mathbf{r}-\mathbf{r}_k|}+\psi,\\ \psi&=\sum_{n=0}^{\infty}\sum_{m=-n}^{n}B_{mn}r^nP_n^m(\cos\theta)e^{im\phi},\end{aligned}$$

where $|\mathbf{r}-\mathbf{r}_k|$ is the distance from the charge q_k and ψ is the contribution to the potential from the charge distribution in the surrounding electrolyte. The functions P_n^m are the associated Legendre functions. The potential in the shell, $b<r<a$, is given by

$$V_2=\sum_{n=0}^{\infty}\sum_{m=-n}^{n}\left(\frac{C_{mn}}{r^{n+1}}+G_{mn}r^n\right)P_n^m(\cos\theta)e^{im\phi},$$

and the potential in the electrolyte solution, $r>a$, is given by

$$V_3=\sum_{n=0}^{\infty}\sum_{m=-n}^{n}\left(\frac{A_{mn}}{r^{n+1}}\right)K_n(\kappa r)\exp(-\kappa r)P_n^m(\cos\theta)e^{im\phi},$$
$$K_n(x)=\sum_{s=0}^{n}\frac{2^s n!(2n-s)!}{s!(2n)!(n-s)!}x^s,$$

where κ is the usual Debye parameter. The coefficients A_{mn}, B_{mn}, C_{mn}, and G_{mn} are found, for each set of m and n, by imposing the interface jump conditions. The interested reader is referred to [17] for more details on the derivation of these equations.

C Calculating electrostatic energies

The total electrostatic free energy of a biomolecule is given by [38]:

$$\Delta G^{\mathrm{T}}=\int_{\Omega}\left(\rho^{\mathrm{f}}\psi-2k_BTc^{\mathrm{b}}\left(\cosh\left(\frac{\psi e}{k_BT}\right)-1\right)-\frac{\epsilon}{2}|\nabla\psi|^2\right)\,\mathrm{d}\mathcal{V},$$

where ρ^{f} is the charge density of the singular charges, c^{b} is the ion concentration of the bulk, k_B is the Boltzmann constant, T is the absolute temperature, e is the charge of a proton, ϵ is the dielectric constant of the electrolyte and ψ is the electrostatic potential field. In this equation, the first, second and last term correspond to the energy due to singular charge interactions, osmotic pressure and the energy stored in the electric field. Since the existence of a jump in the dielectric coefficient indicates a discontinuity in the electric field, it is desired to replace the last term by an equivalent term that only depends on the potential itself and not its gradient. This is easily done by incorporating the Poisson-Boltzmann equation [22] which results in the following equation:

$$\Delta G^{\mathrm{T}}=\int_{\Omega}\left(\frac{1}{2}\rho^{\mathrm{f}}\psi-2k_BTc^{\mathrm{b}}\left(\cosh\left(\frac{\psi e}{k_BT}\right)-1\right)+\psi ec^{\mathrm{b}}\sinh\left(\frac{\psi e}{k_BT}\right)\right)\,\mathrm{d}\mathcal{V}$$
$$-\frac{1}{2}\int_{\partial\Omega}\epsilon\frac{\partial\psi}{\partial n}\psi d\mathcal{A}.$$

To obtain the solvation free energy, it is required that we subtract the energy associated with a reference state. This reference state is taken to be the energy stored in the

electric field that is only due to the singular charges in a medium with the same dielectric constant as that of the molecule. Using Green functions, this energy is given by

$$\Delta G^{\text{ref}} = \frac{1}{2}\sum_{i=1}^{N_m}\sum_{\substack{j=1\\ j\neq i}}^{N_m}\frac{q_i q_j}{4\pi\epsilon^+|\mathbf{r}_i-\mathbf{r}_j|},$$

where q_i is the charge of the i-th atom, ϵ^+ is the dielectric constant of the molecule, N_m is the total number of atoms in the molecule and $\mathbf{r}_i$ is the position vector of the center of the i-th atom. Using the decomposition for the potential, as described in Section A, and the fact that $\psi = 0$ on $\partial\Omega$, the final form of solvation free energy may be written as:

$$\begin{aligned}\Delta G^{\text{sol}} =& \Delta G^{\text{T}} - \Delta G^{\text{ref}}\\ =& \frac{1}{2}\sum_{i=1}^{N_m} q_i\left(\psi^0+\hat{\psi}\right) - 2k_{BT}c^{\text{b}}\int_{\Omega^-}\left(\cosh\left(\frac{\hat{\psi}e}{k_{BT}}\right)-1\right)\mathrm{d}\mathcal{V}\\ &+ ec^{\text{b}}\int_{\Omega^-}\hat{\psi}\sinh\left(\frac{\hat{\psi}e}{k_{BT}}\right)\mathrm{d}\mathcal{V}.\end{aligned} \tag{C.1}$$

References

[1] M. J. Aftosmis, M. J. Berger, and J. E. Melton. Adaptive Cartesian Mesh Generation. In CRC Handbook of Mesh Generation (Contributed Chapter), 1998.

[2] N. A. Baker, D. Bashford, and D. A. Case. Implicit solvent electrostatics in biomolecular simulation. New Algorithms for Macromolecular Simulation, 49(5):263–295, 2006.

[3] N. A. Baker. Improving implicit solvent simulations: A Poisson-centric view. Curr. Opin. Struc. Biol., 15(2):137–143, April 2005.

[4] N. A. Baker, D. Sept, S. Joseph, M. J. Holst, and J. A. McCammon. Electrostatics of nanosystems: Application to microtubules and the ribosome. Proc. Natl. Acad. Sci. USA, 98(18):10037–10041, August 2001.

[5] A. H. Boschitsch and M. O. Fenley. A fast and robust Poisson-Boltzmann solver based on adaptive cartesian grids. J. Chem. Theory Comput., 7(5):1524–1540, 2011.

[6] T. Can, C.-I. Chen, and Y.-F. Wang. Efficient molecular surface generation using level-set methods. J. Mol. Graphics Modell., 25(4):442–454, 2006.

[7] J. Chen, C. L. Brooks, and J. Khandogin. Recent advances in implicit solvent-based methods for biomolecular simulations. Curr. Opin. Struc. Biol., 18(2):140–148, April 2008.

[8] L. Chen, M. J. Holst, and J. Xu. The finite element approximation of the nonlinear Poisson-Boltzmann equation. SIAM J. Nume. Anal., 45(6):2298–2320, 2007.

[9] I.-L. Chern, J.-G. Liu, and W. C. Wang. Accurate evaluation of electrostatics for macromolecules in solution. Methods Appl. Anal., 10:309–328, 2003.

[10] M. L. Connolly. Analytical molecular surface calculation. J. Appl. Crystallogr., 16:548–558, 1983.

[11] M. L. Connolly. The molecular surface package. J. Mol. Graphics, 11(2):139–141, 1993.

[12] F. Fogolari, A. Brigo, and H. Molinari. The Poisson-Boltzmann equation for biomolecular electrostatics: A tool for structural biology. J. Mol. Recognit., 15 (6):377–392, 2002.

[13] W. Geng, S. Yu, and G. W. Wei. Treatment of charge singularities in implicit solvent models. J. Chem. Phys, 127(11):114106, 2007.
[14] M. K. Gilson, K. A. Sharp, and B. H. Honig. Calculating the electrostatic potential of molecules in solution: Method and error assessment. J. Comput. Chem., 9(4):327–335, June 1988.
[15] J. Greer and B. L. Bush. Macromolecular shape and surface maps by solvent exclusion. Proc. Natl. Acad. Sci. USA, 75(1):303, 1978.
[16] A. Helgadottir and F. Gibou. A Poisson-Boltzmann solver on irregular domains with Neumann or Robin boundary conditions on non-graded adaptive grid. J. Comput. Phys., 230:3830–3848, 2011.
[17] J.G. Kirkwood. Theory of solutions of molecules containing widely separated charges with special application to zwitterions. J. Chem. Phys., 2(7):351, 1934.
[18] P. Koehl. Electrostatics calculations: Latest methodological advances. Curr. Opin. Struc. Biol., 16(2):142–151, 2006.
[19] B. Lee and F. M. Richards. The interpretation of protein structures: Estimation of static accessibility. J. Mol. Biol., 55(3):379–400, 1971.
[20] J. Dzubiella J. A. McCammon L.-T. Cheng and Bo. Li. Application of the level-set method to the implicit solvation of nonpolar molecules. J. Chem. Phys., 127:084503, 2007.
[21] B. Z. Lu, Y. C. Zhou, M. J. Holst, and J. A. McCammon. Recent Progress in numerical methods for the Poisson Boltzmann equation in biophysical applications. Commun. Comput. Phys., 3(5):973–1009, 2008.
[22] A. M. Micu, B. Bagheri, A. V. Ilin, L. R. Scott, and B. M. Pettitt. Numerical Considerations in the Computation of the electrostatic free energy of interaction within the Poisson-Boltzmann theory. J. Comput. Phys., 136:263–271, 1997.
[23] C. Min. Local level set method in high dimension and codimension. J. Comput. Phys., 200:368–382, 2004.
[24] C. Min and F. Gibou. Geometric integration over irregular domains with application to level set methods. J. Comput. Phys., 226:1432–1443, 2007.
[25] C. Min and F. Gibou. A second order accurate level set method on non-graded adaptive Cartesian grids. J. Comput. Phys., 225:300–321, 2007.
[26] C. Min, F. Gibou, and H. Ceniceros. A supra-convergent finite difference scheme for the variable coefficient Poisson equation on non-graded grids. J. Comput. Phys., 218:123–140, 2006.
[27] M. Mirzadeh, M. Theillard, and F. Gibou. A second-order discretization of the nonlinear Poisson–Boltzmann equation over irregular geometries using non-graded adaptive Cartesian grids. J. Comput. Phys., 230:2125–2140, 2010.
[28] Y-T. Ng, C. Min, and F. Gibou. An efficient fluid–solid coupling algorithm for single–phase flows. J. Comput. Phys., 228:8807–8829, 2009.
[29] S. Osher and R. Fedkiw. Level Set Methods and Dynamic Implicit Surfaces. Springer-Verlag, 2002. New York, NY.
[30] S. Osher and J. Sethian. Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations. J. Comput. Phys., 79:12–49, 1988.
[31] G. W. Wei, P. Bates, and S. Zhao. Minimal molecular surfaces and their applications. J. Comput. Chem., 29:380–391, 2008.
[32] Q. Pan and X-C. Tai. Model the solvent-excluded surface of 3d protein molecular structures using geometric pde-based level-set method. Commun. Comput. Phys., 6:777–792, 2009.
[33] F. M. Richards. Areas, Volumes, Packing, and Protein Structure. Annu. Rev. Biophys. Bio.,

6(1):151–176, 1977.
[34] H. Samet. The Design and Analysis of Spatial Data Structures. Addison-Wesley, New York, 1989.
[35] H. Samet. Applications of Spatial Data Structures: Computer Graphics, Image Processing and GIS. Addison-Wesley, New York, 1990.
[36] M. F. Sanner and A. J. Olson. Reduced surface: An efficient way to compute molecular surfaces. Biopolymers, 38(3):305–320, 1996.
[37] J. A. Sethian. Level Set Methods and Fast Marching Methods. Cambridge University Press, 1999. Cambridge.
[38] K. A. Sharp and B. Honig. Calculating total electrostatic energies with the nonlinear Poisson-Boltzmann equation. J. Phys. Chem., 94(19):7684–7692, September 1990.
[39] J. Warwicker and H. C. Watson. Calculation of the electric potential in the active site cleft due to alpha-helix dipoles. J. Mol. Biol., 157(4):671–679, 1982.
[40] Y. C. Zhou, M. Feig, and G. W. Wei. Highly accurate biomolecular electrostatics in continuum dielectric environments. J. Comput. Chem., 29:87–97, 2007.