EFFICIENT AND ACCURATE HIGHER-ORDER FAST MULTIPOLE BOUNDARY ELEMENT METHOD FOR POISSON BOLTZMANN ELECTROSTATICS

CHANDRAJIT BAJAJ
SHUN-CHUAN ALBERT CHEN

Abstract. The Poisson-Boltzmann equation is a partial differential equation that describes the electrostatic behavior of molecules in ionic solutions. Significant efforts have been devoted to accurate and efficient computation for solving this equation. In this paper, we developed a boundary element framework based on the linear time fast multipole method for solving the linearized Poisson-Boltzmann equation. A higher-order parametric formulation called algebraic spline model is used for accurately approximation of the unknown solution of the linearized Poisson-Boltzmann equation. The numerical test and experimental results show that these techniques offer an efficient and accurate solution for solving the electrostatic problem of molecules.

1. Introduction. Accurate and effective computational approaches for atomistic simulation of bio-molecules are significant topics of current computational biological research. Different biological activities such as drug design or molecular trajectory simulation can be performed based on numerical solutions of solvation energy [34].

The Molecular Energetics and Force Calculation Problem:

The Potential energy of a molecule in solution includes two different parts, bonded energy E_{bonded} and solvation energy G_{sol}. The total energy of the system is $U = E_{\text{bonded}} + G_{\text{sol}}$. The bonded energy E_{bonded} is measured by the following equation [29].

$$\begin{array}{ll} E_{\text{MM}} & = \sum_{\text{bonds}} k_b(r - r_{eq})^2 + \sum_{\text{bond angles}} k_\theta(\theta - \theta_{eq})^2 \\ & + \sum_{\text{torsion angles}} k_\phi(1 - \cos[n(\phi - \phi_{eq})]) \end{array} .$$

The first three sums represent bonded interaction: covalent bonds, valence bonds, and torsions around bounds [16, 17]. The effect of the solvation energy G_{sol} is used when a molecule in the solution. The solvation energy is the sum of the energy to form a cavity in the solvent G_{cav}, van der Waals interaction energy G_{vdw}, and the electrostatic potential energy change due to the solvation G_{pol}(also known as polarization energy) [21, 22, 23, 36, 39].

$$G_{\text{sol}} = G_{\text{cav}} + G_{\text{vdw}} + G_{\text{pol}}, \tag{1.1}$$

The electrostatic solvation energy is the change in the electrostatic energy due to the induced polarization in the solvent, and so the electrostatic component of the solvation energy can be written as

$$G_{\text{pol}} = \frac{1}{2} \int \phi_{\text{rxn}}(\vec{x}) \rho(\vec{x}) \, dV, \tag{1.2}$$

where $\rho(\vec{x})$ is the charge density at position $\vec{x}$ and the reaction electrostatic potential $\phi_{\text{rxn}}(\vec{x})$ at position $\vec{x}$ indicates the potential change from the air to the solvent, i.e., $\phi_{\text{rxn}} = \phi_{\text{sol}} - \phi_{\text{gas}}$.

A number of applications involve the computation of electrostatic solvation energy. For example, the binding effect of a drug (molecule 1) and its target (molecule 2) can be measured by the following binding energy relation.

$$\Delta G_{bind} = G_{complex} - (G_{molecule1} + G_{molecule2}),$$

which shows that the binding energy is the difference between the solvation energy of the complex of two molecules (e.g. target and drug) and the sum of the solvation energy of the two individual molecules.

On the other hand, the electrostatic forces of molecules can be computed to simulate their activity. The accuracy and computational cost of electrostatic force computation directly affect the simulation results. In order to compute the electrostatic force, the electric field of proteins themselves and the influence of their dielectric and ionic environment should be considered.

Background and Significance: Considerable research efforts have been devoted to calculating binding solvation energy and forces in the past two decades. Based on the solvent model used these different theoretical approaches can be divided into two broad categories: explicit and implicit.

Explicit solvent models adopt a microscopic treatment of both solvent and solute (molecule). Explicit approaches sample the solute-solvent space by molecular dynamics or Monte Carlo techniques which involve a large number of ions, water molecules and molecular atoms. This requires considerable computational effort for calculating the potential functions is needed and explicit solutions are often not practical especially for large domains. [42]

Implicit solvent models treat the solvent as a featureless dielectric material and adopt a semi-microscopic representation of the solute. The effects of the solvent are modeled in terms of dielectric and ionic physical properties. As as result, the computational cost is reduced in comparison with explicit solutions. Implicit continuum electrostatics approaches using the Poisson-Boltzmann (PB) equation are now widely used and have been successfully used to obtain good approximations.

Prior Work: Finite difference method (FDM), finite element method (FEM) and boundary element method (BEM) are three types of numerical solvers widely used to solve the PB equation.

FDM employs a box of three dimensional cubic grids where the molecule and surrounding solvent are contained. The electrostatic potentials are approximately solved on these grid points based on the PB equation. [3, 24, 35]

The idea of FEM is the approximation of partial differential equations in variational form over the space. FEM employs robust and various discretization of three dimensional space. The approximate solution of the PB equation is solved over these discrete elements while some iterative solution strategies, like inexact Newton methods and multilevel algorithms, are often applied for accurate and efficient numerical solution. [18, 19, 15, 9, 25].

Since R.J. Zauhar and R.S. Morgan introduced a BEM paper on continuum electrostatic of biological systems [44], in the past two decades, scientists have made contributions to improve and extend the BEM solution and performance. Some of these works focus on overcoming the difficulties of BEM which typically gives rise to fully populated matrices with numerous singular and hypersingular surface integrals. These works include the implementation of accelerating techniques for numerous singular and hypersingular surface integral operations [31, 11, 41, 14, 12, 27, 2] and, the analysis of and strategies for conditioning the linear system. [31, 14, 2, 30]

Other works make contributions to the methodological generalization including the solution from single molecule to multiple molecules [48, 32], the solution from the two-region case to the multiple-region case [2], and the extensive method for solving nonlinear PB equation. [40, 13]

Main Contributions:

The main contributions of this paper include: (a)A new method that produces a linear Poisson-Boltzmann system of reduced size by discretizing the energy functionals using algebraic spline boundary elements of a molecular surface. (b) The use of GMRES iterative method with KiFMM [43] for a very fast linear solver of the reduced linear system. The results show that our new approach is more accurate and efficient than other numerical solvers even with fewer boundary elements. (c) The consistent parametric formulation of the normal derivative of electrostatic potential is presented as a good approximation.

In the next section, a featureless continuum model widely used in solvated electrostatic simulation and Poisson-Boltzmann theory is introduced. Then we discuss the details of the main technique used to construct algebraic spline models and solve the electrostatic potential, and solvation energy via a kernel-independent, fast multipole method. In section 4, numerical implementation and the simulation results are presented in detail. Finally, we conclude the paper with our experimental results and analysis.

2. Implicit Continuum Model and the Poisson-Boltzmann Equation. A molecule is defined as a stable group of at least two atoms in a definite arrangement held together by very strong chemical covalent bonds. For a molecule embedded in an ionic solution, we separated the open domain ($\mathbb{R}^3$) into interior (Ω) and exterior regions ($\mathbb{R}^3 - \Omega$) by the molecular surface Γ [28].

The continuum model of a molecule in the solvent is then defined by these two regions and used for numerical computation of solvation electrostatic computation. Two important coefficients of a continuum model are dielectric and ionic strength. The dielectric coefficient $\epsilon(\vec{x})$ and ion strength $\mathbb{I}(\vec{x})$ at position $\vec{x}$ depends on which region $\vec{x}$ belongs to.

$$\epsilon(\vec{x}) = \begin{cases} \epsilon_I, & \vec{x} \in \Omega, \\ \epsilon_E, & \vec{x} \in \mathbb{R}^3 - \Omega. \end{cases}$$

$$\mathbb{I}(\vec{x}) = \begin{cases} 0, & \vec{x} \in \Omega, \\ \mathbb{I}, & \vec{x} \in \mathbb{R}^3 - \Omega. \end{cases}$$

where ϵ_I and ϵ_E are dielectric constants. $\mathbb{I}$ is the constant ionic strength of the solvent.

Based on the continuum model, the electrostatic potential in the interior and exterior of a molecule is governed by Poisson equation.

$$\nabla \cdot (\epsilon(\vec{x}) \nabla \phi(\vec{x})) = \rho_c(\vec{x}) + \rho_b(\vec{x})$$

$$\begin{aligned} \rho_c(\vec{x}) &= -4\pi \textstyle\sum_{k=1}^{n_c} \frac{q_k}{\epsilon_I} \delta(\vec{x} - \vec{\mathrm{x}}_k) \\ \rho_b(\vec{x}) &= \lambda(\vec{x}) \textstyle\sum_i e_c z_i c_i e^{-e_c z_i \phi(\vec{x})/k_B T} \\ &\quad \text{if the solvent only contains ions with 1 and } -1 \text{ charges,} \\ &= \lambda(\vec{x}) \bar{\kappa}^2(\vec{x}) \left(\frac{k_B T}{e_c}\right) \sinh\left(\frac{e_c \phi(\vec{x})}{k_B T}\right). \end{aligned}$$

where each notation in the equation is defined as follows.

$\epsilon(\vec{x})$	dielectric coefficient at x
q_k	charge of the atom k
$\vec{x}_k$	the position of charge point q_k (center of the atom k)
n_c	the number of point charges
$\lambda(\vec{x})$	1 outside molecule, 0 inside molecule
$\bar{\kappa}(\vec{x}) = \sqrt{\frac{8\pi e_c^2 \mathbb{I}(\vec{x})}{k_B T}}$	modified Debye-Huckel parameter
e_c	the charge of an electron
k_B	Boltzmann's constant
T	the absolute temperature
$\mathbb{I}(\vec{x}) = \frac{1}{2}\sum_i c_i z_i^2$	The ionic strengths at $\vec{x}$
c_i, z_i	concentration and charge of i^{th} ionic species

In the Poisson-Boltzmann equation, the charge density $\rho_c(\vec{x})$ is explicitly determined by atomic charges of a molecule and the charge density $\rho_b(\vec{x})$ is implicitly approximated by Boltzmann distribution of ionic charges.

The linearized PB equation approximated from linearizing the full PB equation is widely used and believed as an efficient approximation for the regular solvation electrostatic problem. [32][12][41]

$$\nabla \cdot (\epsilon(\vec{x})\nabla\phi(\vec{x})) = \rho_c(\vec{x}) + \rho_b^L(\vec{x})$$

where $\rho_b^L(\vec{x}) = \bar{\kappa}^2(\vec{x})\phi(\vec{x})$ is the first term of Taylor expansion of $\rho_b(\vec{x})$.

By solving this equation, we can obtain the electrostatic potential $\phi(\vec{x})$ over the entire region. Since it is often difficult to directly solve the Poisson-Boltzmann equation for this kind of complex molecule-solvent systems, several computer programs have been created to solve it numerically. We also developed a boundary element solver with fast multipole method to numerically solve the linearized Poisson-Boltzmann equation.

3. Boundary element solution of the Poisson-Boltzmann equation. In this paper, we solve the Poisson-Boltzmann molecular electrostatic problem for the real protein complex. The inputs of the solver are 3-D structures of bio-molecules obtained from the RCSB protein data bank (PDB) and the outputs are the PB numerical electrostatic results. The RCSB protein data bank (PDB) is a worldwide data repository for the distribution of 3-D structure data of large molecules of proteins. PDB is a file format which contains the exact locations of all atoms in a molecule and the list of amino acids making up a particular protein. The molecular model for continuum electrostatic calculations is obtained from a PDB file by assigning charge and radius parameters derived from a variety of force fields, e.g. AMBER, CHARMM, etc. For example, the adaptive Poisson-Boltzmann solver, APBS, applies the all-atom AMBER force field to prepare the setup of Poisson-Boltzmann electrostatic calculations. [20]

In the definition of the continuum model, the whole region is divided by its molecular surface Γ. In this paper, the molecular surface Γ is extracted using geometric flow evolution in a level set formulation [7, 8]. The basic idea of this technique is to drive an initial approximate surface obtained from the Gaussian density function to the final molecular surface in the level set formulation using geometric flow evolution. The surface is then discretized into triangular meshes which is extracted from the level set using dual contouring method. After the preparation of the continuum model,

our spline based boundary element method is applied to numerically solve the linear PB equation over the triangular mesh. The numerical treatment of the PB equation based on a C^1 algebraic spline model will be described and the consistency of the electrostatic potential is guaranteed by the parametrization of electrostatic potential using the algebraic spline model.

3.1. Boundary representations and boundary integral equations. The PB boundary formulation, also called PB boundary integral equations, is derived from the PB equation using the boundary conditions on the interface of the continuum model. The interface is now defined by the triangular mesh of the molecular surface Γ. Because of the zero ionic strength inside the molecule , the ionic charge density $\rho_b(\vec{x}) = 0$ in the molecular region. The molecular charge density $\rho_c(\vec{x}) = 0$ in the solvent region because the molecular charge density in the PB equation is defined by the delta functions of the atomic positions. Therefore, the linearized Poisson-Boltzmann equation can be separated into two formulations,

$$\begin{array}{ll} \nabla \cdot (\epsilon(\vec{x})\nabla\phi_i(\vec{x})) = -4\pi \sum_{k=1}^{n_c} q_k \delta(\vec{x} - \vec{\mathrm{x}}_k) & \vec{x} \in \Omega \\ \nabla \cdot (\epsilon(\vec{x})\nabla\phi_e(\vec{x})) = \kappa^2 \phi_e(\vec{x}) & \vec{x} \in \mathbb{R}^3 - \Omega \end{array} \tag{3.1}$$

where $\phi_i(\vec{x})$ and $\phi_e(\vec{x})$ are the electrosatic potential interior to and exterior to the point $\vec{x}$ on the surface Γ.

The boundary conditions on the surface Γ of Ω is then defined by the electrostiatic potential $\phi_i(\vec{x})$ and $\phi_e(\vec{x})$ and their normal derivatives.

$$\begin{array}{lll} \phi(\vec{x}) & = \phi_i(\vec{x}) & = \phi_e(\vec{x}) \\ \frac{\partial\phi}{\partial\vec{n}^{(x)}}(\vec{x}) & = \frac{\partial\phi_i}{\partial\vec{n}^{(x)}}(\vec{x}) & = \frac{\epsilon_E}{\epsilon_I}\frac{\partial\phi_e}{\partial\vec{n}^{(x)}}(\vec{x}) \end{array} \tag{3.2}$$

After applying Green's second identity to the interior Poisson equation and exterior PB equation (3.1) with the boundary conditions (3.2), the boundary integral equations of PB equation are derived as follows,

$$\frac{1}{2}\phi(\vec{x}) + \fint_\Gamma \left(\frac{\partial}{\partial\vec{n}^{(y)}} G_0(\vec{x},\vec{y})\phi(\vec{y}) - G_0(\vec{x},\vec{y})\frac{\partial\phi(\vec{y})}{\partial\vec{n}^{(y)}} \right) d\vec{y} = \sum_{k=1}^{n_c} \frac{q_k}{\epsilon_I} G_0(\vec{x},\vec{\mathrm{x}}_k) \tag{3.3}$$

$$\frac{1}{2}\phi(\vec{x}) + \fint_\Gamma \left(\frac{\epsilon_E}{\epsilon_I} G_\kappa(\vec{x},\vec{y})\frac{\partial\phi(\vec{y})}{\partial\vec{n}^{(y)}} - \frac{\partial}{\partial\vec{n}^{(y)}} G_\kappa(\vec{x},\vec{y})\phi(\vec{y}) \right) d\vec{y} = 0 \tag{3.4}$$

where $\fint$ is the principal value integral. The Green's functions, also called fundamental solutions, for the PB equation are

$$\begin{array}{llll} G_0(\vec{x},\vec{y}) = \frac{1}{4\pi\|\vec{x}-\vec{y}\|} & \frac{\partial G_0(\vec{x},\vec{y})}{\partial\vec{n}^{(y)}} = \frac{-\cos\theta_0}{4\pi\|\vec{x}-\vec{y}\|^2} \\ G_\kappa(\vec{x},\vec{y}) = \frac{e^{-\kappa\|\vec{x}-\vec{y}\|}}{4\pi\|\vec{x}-\vec{y}\|} & \frac{\partial G_\kappa(\vec{x},\vec{y})}{\partial\vec{n}^{(y)}} = \frac{-e^{-\kappa\|\vec{x}-\vec{y}\|}(1.0+\kappa\|\vec{x}-\vec{y}\|)\cos\theta_0}{4\pi\|\vec{x}-\vec{y}\|^2} \end{array}$$

where $\cos\theta_0 = \frac{(\vec{x}-\vec{y})\cdot\vec{n}^{(y)}}{\|\vec{x}-\vec{y}\|}$ and $\cos\theta = \frac{(\vec{x}-\vec{y})\cdot\vec{n}^{(x)}}{\|\vec{x}-\vec{y}\|}$ and $\vec{n}^{(y)}$ is the surface normal on the point $\vec{y}$ [44].

Figure 3.1 shows an example of the boundary element decomposition of a four-atom molecule. The molecular surface Γ has been discretized into elements Γ_i, $i = 1, ..., N$. $\vec{x}_i$ represents a point on an element Γ_i as $\vec{y}_j$ represents a point on Γ_j. Their normal vectors are written as $\vec{n}_i^{(x)}$ and $\vec{n}_j^{(y)}$. Boundary integral equations

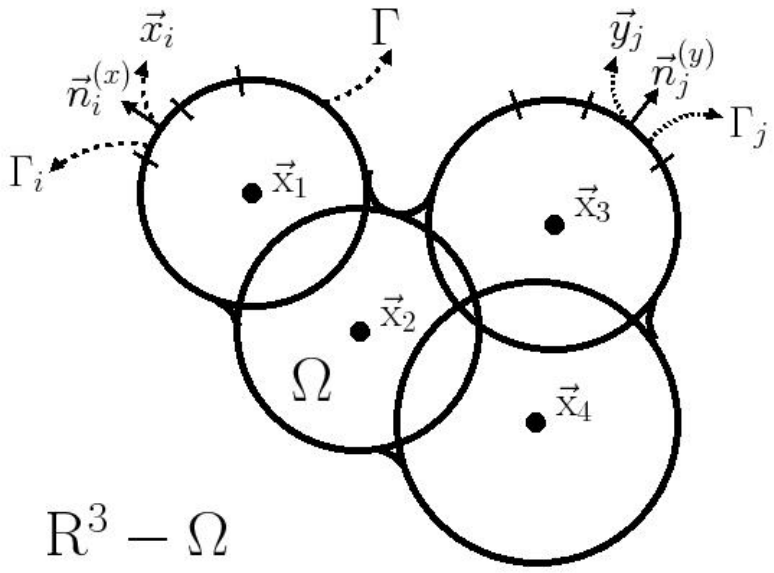


FIG. 3.1. *The example of boundary element decomposition of a four-atom molecule:* $\vec{x}_k$ *is the center of* k^{th} *atom,* $\vec{x}_i$ *and* $\vec{y}_j$ *are the points on the elements* Γ_i *and* Γ_j *of the surface* Γ *and* $\vec{n}_i^{(x)}$ *and* $\vec{n}_j^{(y)}$ *are their normal vectors.*

can be written as a linear system using this notation. An improved version of the boundary integral equations was proposed by A.H. Juffer and other researchers by linearly combining the derivative forms of these two boundary integral equations. The formulation is also called derivative boundary integral equations (dBIEs)[1].

$$\begin{array}{l} \frac{1}{2}(1+\frac{\epsilon_E}{\epsilon_I})\phi(\vec{x}) + \int\!\!\!\!\!\!-_\Gamma (\frac{\partial G_0(\vec{x},\vec{y})}{\partial \vec{n}^{(y)}} - \frac{\epsilon_E}{\epsilon_I}\frac{\partial G_\kappa(\vec{x},\vec{y})}{\partial \vec{n}^{(y)}})\phi(\vec{y})d\vec{y} \\ -\int\!\!\!\!\!\!-_\Gamma (G_0(\vec{x},\vec{y}) - G_\kappa(\vec{x},\vec{y}))\frac{\partial \phi(\vec{y})}{\partial \vec{n}^{(y)}}d\vec{y} = \sum_{k=1}^{n_c}\frac{q_k}{\epsilon_I}G_0(\vec{x},\vec{\mathrm{x}}_k) \end{array} \tag{3.5}$$

$$\begin{array}{l} \frac{1}{2}(1+\frac{\epsilon_I}{\epsilon_E})\frac{\partial \phi(\vec{x})}{\partial \vec{n}^{(x)}} + \int\!\!\!\!\!\!-_\Gamma (\frac{\partial^2 G_0(\vec{x},\vec{y})}{\partial \vec{n}^{(x)}\partial \vec{n}^{(y)}} - \frac{\partial^2 G_\kappa(\vec{x},\vec{y})}{\partial \vec{n}^{(x)}\partial \vec{n}^{(y)}})\phi(\vec{y})d\vec{y} \\ -\int\!\!\!\!\!\!-_\Gamma (\frac{\partial G_0(\vec{x},\vec{y})}{\partial \vec{n}^{(x)}} - \frac{\epsilon_I}{\epsilon_E}\frac{\partial G_\kappa(\vec{x},\vec{y})}{\partial \vec{n}^{(x)}})\frac{\partial \phi(\vec{y})}{\partial \vec{n}^{(y)}}d\vec{y} = \sum_{k=1}^{n_c}\frac{q_k}{\epsilon_I}\frac{\partial G_0(\vec{x},\vec{x}_k)}{\partial \vec{n}^{(x)}} \end{array} \tag{3.6}$$

where

$$\begin{array}{c} \frac{\partial G_0(\vec{x},\vec{y})}{\partial \vec{n}^{(x)}} = \frac{\cos\theta}{4\pi\|\vec{x}-\vec{y}\|^2} \\ \frac{\partial G_\kappa(\vec{x},\vec{y})}{\partial \vec{n}^{(x)}} = \frac{e^{-\kappa\|\vec{x}-\vec{y}\|}(1.0+\kappa\|\vec{x}-\vec{y}\|)\cos\theta}{4\pi\|\vec{x}-\vec{y}\|^2} \\ \frac{\partial^2 G_0(\vec{x},\vec{y})}{\partial \vec{n}^{(x)}\partial \vec{n}^{(y)}} = \frac{(\vec{n}^{(x)}\cdot\vec{n}^{(y)})-3\cos\theta_0\cos\theta}{4\pi\|\vec{x}-\vec{y}\|^3} \\ \frac{\partial^2 G_\kappa(\vec{x},\vec{y})}{\partial \vec{n}^{(x)}\partial \vec{n}^{(y)}} = e^{-\kappa\|\vec{x}-\vec{y}\|}(1.0+\kappa\|\vec{x}-\vec{y}\|)\frac{G_0(\vec{x},\vec{y})}{\partial \vec{n}^{(x)}\partial \vec{n}^{(y)}} - \frac{\kappa^2 e^{-\kappa\|\vec{x}-\vec{y}\|}}{4\pi\|\vec{x}-\vec{y}\|}\cos\theta_0\cos\theta \end{array}$$

The main advantage of this reformulation is that it leads to a well-conditioned system. The linear iterative solver converges much faster than the original boundary integral equations which are also called non-derivative boundary integral equations (nBIEs) [30].

4. Numerical Treatment of Boundary Integral Equations. The main difficulties to solving the PB boundary integral equations are the full populated matrix, the singular and hypersingular integral and numerous integral operations. The nonzero entries of the populated matrix is $O(N^2)$ where N is the number of unknowns. It is proportional to the number of discrete elements or vertices of the molecular surface. Because of the large number of elements necessary for discretizing the surface, we use a Krylov iterative linear solver instead of direct solver to solve the Poisson-Boltzmann boundary integral equations.

Meanwhile, we are unable to store and access the full matrix in the memory. However, for each iteration, the matrix-vector product should be evaluated. The cost of the straight forward evaluation is expensive, such that time complexity is $O(N^2)$ where N is the number of unknowns. Fast multipole method is a technique to improve the time complexity of matrix-vector product to linear time evaluation.

In addition, the numerical computation of the surface integral depends on the parametrization of triangular elements. In this paper, we compare the evaluation of the integrals of kernel functions between planar linear elements and higher-order algebraic elements. We do the parametrization on the triangulation of the molecular surface. The triangulation of the surface is composed of the vertices $\mathrm{V} = \{\vec{\mathrm{v}}_i\}_{i=1}^{P}$ with their unit normal vectors $\{\vec{\mathrm{n}}_i\}_{i=1}^{P}$ and the triangular elements $\Gamma = \{\Gamma_j | \Gamma_j = \vec{v}_{j1}\vec{v}_{j2}\vec{v}_{j3}$ where $j = 1, \cdots, L$ and $\vec{v}_{j1}, \vec{v}_{j2}, \vec{v}_{j3} \in \mathrm{V}\}$. Before we discuss the details of

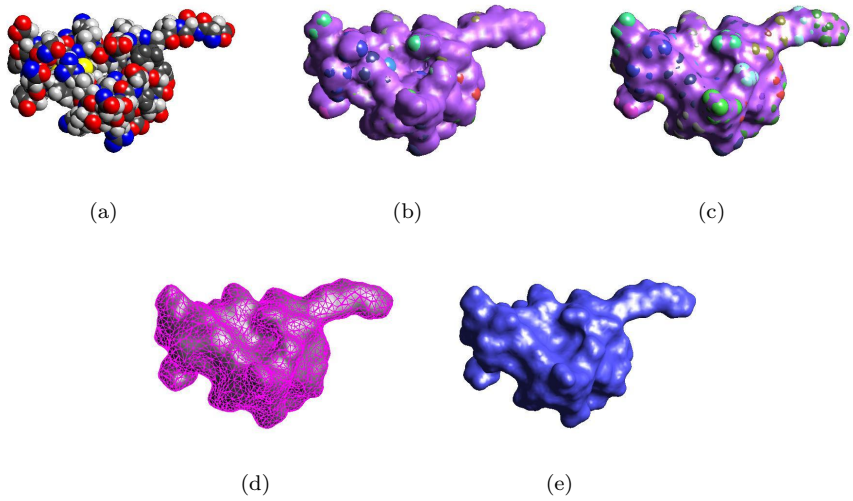


FIG. 4.1. *Molecular model of a protein (PDB id:1CGI); (a) The van der Waals surface of its 3D atomic model (852 atoms); (b) The surface generated using the Gauss density function from its 3D atomic model (c) Its solvent excluded surfaces (SES), also called molecular surface; (d) The decimated triangulation of SESs; (e) The piecewise algebraic surface patches generated from the decimated triangulation of SESs.*

our numerical treatment, we take a ligand protein (PDB id: 1CGI) as an example to show some important and useful types of molecular structures in Figure 4.1. Figure (a) is the van der Waals surface which represents the 3D atomic structure of the molecule, where the color of each sphere represents its residue type and the radius of each sphere represents the radius of the atom. Based on 3D atomic structure, the molecular surface is generated to apply the boundary condition (3.2) of our solution.

Figure (b) shows the approximate molecular surface extracted from the level set of Gaussian density function from the 3D atomic model. Figure (c) is the molecular surface of this protein generated using geometric flow evolution in level set formulation and this surface is a better approximation to the definition of molecular surface, also called solvent excluded surface [7, 8].

Based on the decimated triangulation of the molecular surface in Figure (d), the algebraic spline surface, composed by a higher order curve element, A-patches, is concluded, as is shown in the figure (e). This higher-order model is applied for improving the accuracy and efficiency of electrostatics computation. We will describe all the details in this section.

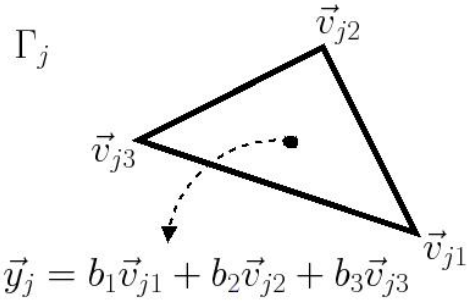


FIG. 4.2. *The parametric representation of a linear triangular element: v_{j1}, v_{j2}, v_{j3} are the vertices of the j^{th} triangular element Γ_j and y^j is an arbitrary point on the element.*

4.1. Parametric linear element. Figure 4.2 shows the point $\vec{y}_j$ on the element $\Gamma_j = \vec{v}_{j1}\vec{v}_{j2}\vec{v}_{j3}$. The parametric form of $\vec{y}_j$ is given by its barycentric coordinates $(b_1, b_2, b_3)^T$ as follows.

$$\begin{bmatrix} \vec{y}_j \\ 1 \end{bmatrix} = \begin{bmatrix} \vec{v}_{j1} & \vec{v}_{j2} & \vec{v}_{j3} \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

We then approximate those integrals with a kernel function $G(\vec{x}, \vec{y})$ using the Gaussian quadrature.

$$\oint_{\Gamma} G(\vec{x}, \vec{y}) f(\vec{y}) d\vec{y} = \sum_{j=1}^{L} \oint_{\Gamma_j} G(\vec{x}, \vec{y}_j) f(\vec{y}_j) d\vec{y}_j = \sum_{j=1}^{L} \sum_{m=1}^{M} W_m G(\vec{x}, \vec{y}_{jm}) f(\vec{y}_{jm}) J(\Gamma_j)$$

where $\vec{x}$ is the evaluation point; $\{W_m\}_{m=1}^{M}$ and $\{\vec{y}_{jm}\}_{m=1}^{M}$ are m^{th} weight and point of Gaussian quadrature, and $J(\Gamma_j)$ is the Jacobian (area) of the linear element Γ_j.

4.2. Algebraic-spline model and parametrization. Algebraic patches or A-patches are a kind of low degree algebraic surface finite elements with dual implicit and rational parametric representations [4]. The A-patch element is defined within a prism scaffold as shown in the figure 4.3. For some triangle element $\Gamma_j = \vec{v}_{j1}\vec{v}_{j2}\vec{v}_{j3}$ of a triangulation of the molecular surface, the A-patch $\{\bar{\Gamma}_j\}_{j=1}^{L}$ is defined on this prism.

$$\vec{v}_{jl}(\lambda) = \vec{v}_{jl} + \lambda\vec{n}_{jl}, \qquad l = 1, 2, 3$$

where the prism is defined by

$$D(\Gamma_j) := \{\vec{y} : \ \vec{y} = b_1 v_{j1}(\lambda) + b_2 v_{j2}(\lambda) + b_3 v_{j3}(\lambda), \ 0 \le \lambda \le 1\}$$

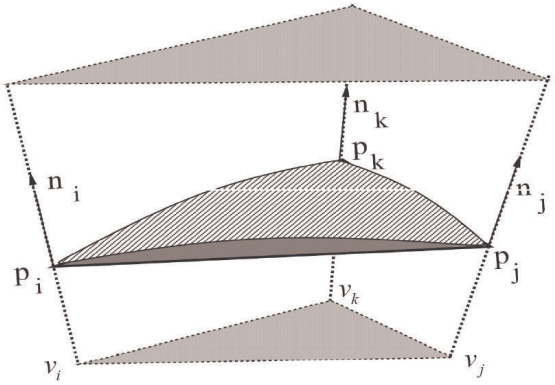


FIG. 4.3. *A prism scaffold of triangular element* $v_i v_j v_k$

where (b_1, b_2, b_3) are the barycentric coordinates of points in $\vec{v}_{j1}\vec{v}_{j2}\vec{v}_{j3}$.

According to the definition of algebraic patches, we define an implicit function over the prism $D(\Gamma_j)$ in Benstein-Bezier spline form.

$$F_d(b_1, b_2, b_3, \lambda) = \sum_{i+j+k=d} b_{ijk}(\lambda) B^d_{ijk}(b_1, b_2, b_3)$$

$$B^d_{ijk}(b_1, b_2, b_3) = \frac{d!}{i!j!k!} b_1^i b_2^j b_3^k$$

which is also called the algebraic spline model. The details of the parametrization of algebraic spline model are in [47]. The molecular surface Γ can be approximated by the zero contour of the implicit function F_d:

$$\{(b_1, b_2, b_3, \lambda) : F_d(b_1, b_2, b_3, \lambda) = 0\}$$

Now, given the barycentric coordinates $(b_1, b_2, b_3)^T$ on the triangle, the parametric form of the position $\vec{y}_j$ on the A-patch element $\bar{\Gamma}_j = \vec{v}_{j1}\vec{v}_{j2}\vec{v}_{j3}$ is

$$\begin{bmatrix} \vec{y}_j \\ 1 \end{bmatrix} = \begin{bmatrix} \vec{v}_{j1}(\lambda) & \vec{v}_{j2}(\lambda) & \vec{v}_{j3}(\lambda) \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

As what we did for linear element, we can also approximate those integrals using Gaussian quadrature on the A-patches [47],

$$\oint_{\bar{\Gamma}} G(\vec{x}, \vec{y}) f(\vec{y}) d\vec{y} = \sum_{j=1}^{L} \oint_{\bar{\Gamma}_j} G(\vec{x}, \vec{y}_j) f(\vec{y}_j) d\vec{y}_j = \sum_{j=1}^{L} \sum_{m=1}^{M} W_m G(\vec{x}, \vec{y}_{jm}) f(\vec{y}_{jm}) J(\bar{\Gamma}_j)$$

where $\vec{x}$ is the evaluation point. $\bar{\Gamma}_j$ is the zero contour of the cubic Bezier basis over j^{th} triangle where W_m and $\vec{y}_{jm} = b_{m1}\vec{v}_{j1}(\lambda_{jm}) + b_{m2}\vec{v}_{j2}(\lambda_{jm}) + b_{m3}\vec{v}_{j3}(\lambda_{jm})$ are the m^{th} weight and points of Gaussian quadrature on this patch $\bar{\Gamma}_j$. The Jacobian weight $J(\bar{\Gamma}_j)$ is described in the appendix as the area of the patch. Because $\vec{y}_{jm} =$

$\sum_{i=1}^{3} b_{mi}\vec{v}_{ji}(\lambda_{jm})$ is the m^{th} integration point on the element Γ_j and

$$\begin{aligned}
&G_0(\vec{x}_i,\vec{y}_{jm}) = \tfrac{1}{4\pi\|\vec{x}_i-\vec{y}_{jm}\|}\\
&G_\kappa(\vec{x}_i,\vec{y}_{jm}) = \tfrac{e^{-\kappa\|\vec{x}_i-\vec{y}_{jm}\|}}{4\pi\|\vec{x}_i-\vec{y}_{jm}\|}\\
&\tfrac{\partial G_0(\vec{x}_i,\vec{y}_{jm})}{\partial\vec{n}_{jm}^{(y)}} = \tfrac{-(\vec{x}_i-\vec{y}_{jm})\cdot\vec{n}_{jm}^{(y)}}{4\pi\|\vec{x}_i-\vec{y}_{jm}\|^3}\\
&\tfrac{\partial G_\kappa(\vec{x}_i,\vec{y}_j)}{\partial\vec{n}_{jm}^{(y)}} = \tfrac{-e^{-\kappa\|\vec{x}_i-\vec{y}_{jm}\|}(1+\kappa\|\vec{x}_i-\vec{y}_{jm}\|)(\vec{x}_i-\vec{y}_{jm})\cdot\vec{n}_{jm}^{(y)}}{4\pi\|\vec{x}_i-\vec{y}_{jm}\|^3},
\end{aligned}$$

the numerical treatment of nBIEs (3.3) and (3.4) becomes

$$\begin{aligned}
\tfrac{1}{2}\phi(\vec{x}_i) \quad &+\textstyle\sum_{j=1}^{L}\sum_{m=1}^{M}\phi(\vec{y}_{jm})W_m\frac{\partial G_0}{\partial\vec{n}_{jm}^{(y)}}(\vec{x}_i,\vec{y}_{jm})J(\vec{y}_{jm})\\
&-\textstyle\sum_{j=1}^{L}\sum_{m=1}^{M}\frac{\partial\phi}{\partial\vec{n}_{jm}^{(y)}}(\vec{y}_{jm})W_m G_0(\vec{x}_i,\vec{y}_{jm})J(\vec{y}_{jm}) = \sum_{k=1}^{n_c}\tfrac{q_k}{\epsilon_I}G_0(\vec{x}_i,\vec{\mathrm{x}}_k)\\
\tfrac{1}{2}\phi(\vec{x}_i) \quad &-\textstyle\sum_{j=1}^{L}\sum_{m=1}^{M}\phi(\vec{y}_{jm})W_m\frac{\partial G_\kappa}{\partial\vec{n}_{jm}^{(y)}}(\vec{x}_i,\vec{y}_{jm})J(\vec{y}_{jm})\\
&+\tfrac{\epsilon_I}{\epsilon_E}\textstyle\sum_{j=1}^{L}\sum_{m=1}^{M}\frac{\partial\phi}{\partial\vec{n}_{jm}^{(y)}}(\vec{y}_{jm})W_{mf}G_\kappa(\vec{x}_i,\vec{y}_{jm})J(\vec{y}_{jm}) = 0
\end{aligned}$$

where L is the number of patches, and the numerical treatment of dBIEs (3.5) and (3.6) is

$$\begin{aligned}
&\tfrac{1}{2}(1+\tfrac{\epsilon_E}{\epsilon_I})\phi(\vec{x}_i)\\
&+\textstyle\sum_{j=1}^{L}\sum_{m=1}^{M}\phi(\vec{y}_{jm})W_m\left(\frac{\partial G_0}{\partial\vec{n}_{jm}^{(y)}}(\vec{x}_i,\vec{y}_{jm})-\frac{\epsilon_E}{\epsilon_I}\frac{\partial G_\kappa}{\partial\vec{n}_{jm}^{(y)}}(\vec{x}_i,\vec{y}_{jm})\right)J(\vec{y}_{jm})\\
&-\textstyle\sum_{j=1}^{L}\sum_{m=1}^{M}\frac{\partial\phi}{\partial\vec{n}_{jm}^{(y)}}(\vec{y}_{jm})W_m\left(G_0(\vec{x}_i,\vec{y}_{jm})-G_\kappa(\vec{x}_i,\vec{y}_{jm})\right)J(\vec{y}_{jm})\\
&=\textstyle\sum_{k=1}^{n_c}\frac{q_k}{\epsilon_I}G_0(\vec{x}_i,\vec{\mathrm{x}}_k)\\
&\tfrac{1}{2}(1+\tfrac{\epsilon_I}{\epsilon_E})\tfrac{\partial\phi}{\partial\vec{n}_x^i}(\vec{x}_i)\\
&-\textstyle\sum_{j=1}^{L}\sum_{m=1}^{M}\phi(\vec{y}_{jm})W_m\left(\frac{\partial^2 G_0}{\partial\vec{n}_{jm}^{(y)}\partial\vec{n}_x^i}(\vec{x}_i,\vec{y}_{jm})-\frac{\partial^2 G_\kappa}{\partial\vec{n}_{jm}^{(y)}\partial\vec{n}_x^i}(\vec{x}_i,\vec{y}_{jm})\right)J(\vec{y}_{jm})\\
&+\textstyle\sum_{j=1}^{L}\sum_{m=1}^{M}\frac{\partial\phi}{\partial\vec{n}_{jm}^{(y)}}(\vec{y}_{jm})W_m\left(\frac{\partial G_0}{\partial\vec{n}_i^{(x)}}(\vec{x}_i,\vec{y}_{jm})-\frac{\epsilon_I}{\epsilon_E}\frac{\partial G_\kappa}{\partial\vec{n}_i^{(x)}}(\vec{x}_i,\vec{y}_{jm})\right)J(\vec{y}_{jm})\\
&=\textstyle\sum_{k=1}^{n_c}\frac{q_k}{\epsilon_I}\frac{G_0}{\vec{n}_i^{(x)}}(\vec{x}_i,\vec{\mathrm{x}}_k).
\end{aligned}$$

Now, the boundary integral equations are treated as a linear system For better elaboration, we can write the boundary integral equations in the following matrix form.

$$\begin{bmatrix}\frac{1}{2}I+\frac{\partial G_0}{\partial\vec{n}^{(y)}} & -G_0\\ \frac{1}{2}I-\frac{\partial G_\kappa}{\partial\vec{n}^{(y)}} & \frac{\epsilon_I}{\epsilon_E}G_\kappa\end{bmatrix}\begin{bmatrix}\phi\\ \frac{\partial\phi}{\partial\vec{n}}\end{bmatrix}=\begin{bmatrix}\sum_{k=1}^{n_c}\frac{q_k}{\epsilon_I}G_{0,k}\\ 0\end{bmatrix}$$

where

- ϕ_j and $\left(\frac{\partial\phi}{\partial\vec{n}}\right)_j$ are the j^{th} unknown electrostatic potential and its normal derivative at some point $\vec{y}_j$ on the patch $\bar{\Gamma}_j$
- I is the identity operator so that $I_{ij}\phi_j=\phi_j$

- The operators compute the potential at the point $\vec{x}_i$ due to the patch $\bar{\Gamma}_j$

$$\begin{aligned}
\left(\frac{\partial G_0}{\partial \vec{n}^{(y)}}\right)_{ij}\phi_j &= \oint_{\bar{\Gamma}_j}\frac{\partial G_0}{\partial \vec{n}_j^{(y)}}(\vec{x}_i,\vec{y}_j)\phi(\vec{y}_j)d\vec{y}_j \\
&= \sum_{m=1}^{M} W_m \frac{\partial G_0}{\partial \vec{n}_{jm}^{(y)}}(\vec{x}_i,\vec{y}_{jm})\phi(\vec{y}_{jm})J(\bar{\Gamma}_j) \\
(G_0)_{ij}\left(\frac{\partial \phi}{\partial \vec{n}}\right)_j &= \oint_{\bar{\Gamma}_j} G_0(\vec{x}_i,\vec{y}_j)\frac{\partial \phi(\vec{y}_j)}{\partial \vec{n}_j^{(y)}}d\vec{y}_j \\
&= \sum_{m=1}^{M} W_m G_0(\vec{x}_i,\vec{y}_{jm})\frac{\partial \phi(\vec{y}_{jm})}{\partial \vec{n}_{jm}^{(y)}}J(\bar{\Gamma}_j) \\
\left(\frac{\partial G_\kappa}{\partial \vec{n}^{(y)}}\right)_{ij}\phi_j &= \oint_{\bar{\Gamma}_j}\frac{\partial G_\kappa}{\partial \vec{n}_j^{(y)}}(\vec{x}_i,\vec{y}_j)\phi(\vec{y}_j)d\vec{y}_j \\
&= \sum_{m=1}^{M} W_m \frac{\partial G_\kappa}{\partial \vec{n}_{jm}^{(y)}}(\vec{x}_i,\vec{y}_{jm})\phi(\vec{y}_{jm})J(\bar{\Gamma}_j) \\
(G_\kappa)_{ij}\left(\frac{\partial \phi}{\partial \vec{n}}\right)_j &= \oint_{\bar{\Gamma}_j} G_\kappa(\vec{x}_i,\vec{y}_j)\frac{\partial \phi(\vec{y}_j)}{\partial \vec{n}_j^{(y)}}d\vec{y}_j \\
&= \sum_{m=1}^{M} W_m G_\kappa(\vec{x}_i,\vec{y}_{jm})\frac{\partial \phi(\vec{y}_{jm})}{\partial \vec{n}_{jm}^{(y)}}J(\bar{\Gamma}_j) \\
(G_{0,k})_i &= G_0(\vec{x}_i,\vec{\mathrm{x}}_k)
\end{aligned}$$

where $\vec{x}_i$ is a point on the patch $\bar{\Gamma}_i$ and $\vec{y}_{jm}$ is m^{th} Gaussian quadrature point on the patch $\bar{\Gamma}_j$.

dBIEs can also be written in a linear system as nBIEs.

$$\begin{bmatrix} \frac{1}{2}(1+\epsilon)I + \frac{\partial G_\kappa}{\partial \vec{n}^{(y)}} - \epsilon\frac{\partial G_0}{\partial \vec{n}^{(y)}} & G_0 - G_\kappa \\ \frac{\partial^2 G_0}{\partial \vec{n}^{(y)}\partial \vec{n}^{(x)}} - \frac{\partial^2 G_\kappa}{\partial \vec{n}^{(y)}\partial \vec{n}^{(x)}} & \frac{1}{2}(1+\frac{1}{\epsilon})I + \frac{\partial G_\kappa}{\partial \vec{n}^{(x)}} - \frac{\epsilon_I}{\epsilon_E}\frac{\partial G_\kappa}{\partial \vec{n}^{(x)}} \end{bmatrix}\begin{bmatrix} \phi \\ \frac{\partial \phi}{\partial \vec{n}} \end{bmatrix} = \begin{bmatrix} \sum_{k=1}^{n_c}\frac{q_k}{\epsilon_I}G_{0,k} \\ \sum_{k=1}^{n_c}\frac{q_k}{\epsilon_I}\frac{\partial G_{0,k}}{\partial \vec{n}^{(x)}} \end{bmatrix} \tag{4.1}$$

$$\begin{aligned}
\left(\frac{\partial G_0}{\partial \vec{n}^{(x)}}\right)_{ij}\left(\frac{\partial \phi}{\partial \vec{n}}\right)_j &= \oint_{\bar{\Gamma}_j}\frac{\partial G_0}{\partial \vec{n}_i^{(x)}}(\vec{x}_i,\vec{y}_j)\frac{\partial \phi(\vec{y}_j)}{\partial \vec{n}_j^{(y)}}d\vec{y}_j \\
&= \sum_{m=1}^{M} W_m \frac{\partial G_0}{\partial \vec{n}_i^{(x)}}(\vec{x}_i,\vec{y}_{jm})\frac{\partial \phi(\vec{y}_{jm})}{\partial \vec{n}_{jm}^{(y)}}J(\bar{\Gamma}_j) \\
\left(\frac{\partial^2 G_0}{\partial \vec{n}^{(x)}\partial \vec{n}^{(y)}}\right)_{ij}\phi_j &= \oint_{\bar{\Gamma}_j}\frac{\partial^2 G_0}{\partial \vec{n}_i^{(x)}\partial \vec{n}_j^{(y)}}(\vec{x}_i,\vec{y}_j)\phi(\vec{y}_j)d\vec{y}_j \\
&= \sum_{m=1}^{M} W_m \frac{\partial^2 G_0}{\partial \vec{n}_i^{(x)}\partial \vec{n}_{jm}^{(y)}}(\vec{x}_i,\vec{y}_{jm})\phi(\vec{y}_{jm})J(\bar{\Gamma}_j) \\
\left(\frac{\partial G_\kappa}{\partial \vec{n}^{(x)}}\right)_{ij}\left(\frac{\partial \phi}{\partial \vec{n}}\right)_j &= \oint_{\bar{\Gamma}_j}\frac{\partial G_\kappa}{\partial \vec{n}_i^{(x)}}(\vec{x}_i,\vec{y}_j)\frac{\partial \phi(\vec{y}_j)}{\partial \vec{n}_j^{(y)}}d\vec{y}_j \\
&= \sum_{m=1}^{M} W_m \frac{\partial G_\kappa}{\partial \vec{n}_i^{(x)}}(\vec{x}_i,\vec{y}_{jm})\frac{\partial \phi(\vec{y}_{jm})}{\partial \vec{n}_{jm}^{(y)}}J(\bar{\Gamma}_j) \\
\left(\frac{\partial^2 G_\kappa}{\partial \vec{n}^{(x)}\partial \vec{n}^{(y)}}\right)_{ij}\phi_j &= \oint_{\bar{\Gamma}_j}\frac{\partial^2 G_\kappa}{\partial \vec{n}_i^{(x)}\partial \vec{n}_j^{(y)}}(\vec{x}_i,\vec{y}_j)\phi(\vec{y}_j)d\vec{y}_j \\
&= \sum_{m=1}^{M} W_m \frac{\partial^2 G_\kappa}{\partial \vec{n}_i^{(x)}\partial \vec{n}_{jm}^{(y)}}(\vec{x}_i,\vec{y}_{jm})\phi(\vec{y}_{jm})J(\bar{\Gamma}_j) \\
\left(\frac{\partial G_{0,k}}{\partial \vec{n}^{(x)}}\right)_i &= \frac{\partial G_0}{\partial \vec{n}_i^{(x)}}(\vec{x}_i,\vec{\mathrm{x}}_k)
\end{aligned}$$

4.3. Normal derivative of electrostatic potential. In Figure 4.4, the parametric form of the position on a A-patch Γ of a triangulation $\vec{v}_1\vec{v}_2\vec{v}_3$ is shown as

$$\vec{x} = b_1\vec{v}_1(\lambda) + b_2\vec{v}_2(\lambda) + (1-b_1-b_2)\vec{v}_3(\lambda), \vec{x} \in \Gamma \tag{4.2}$$

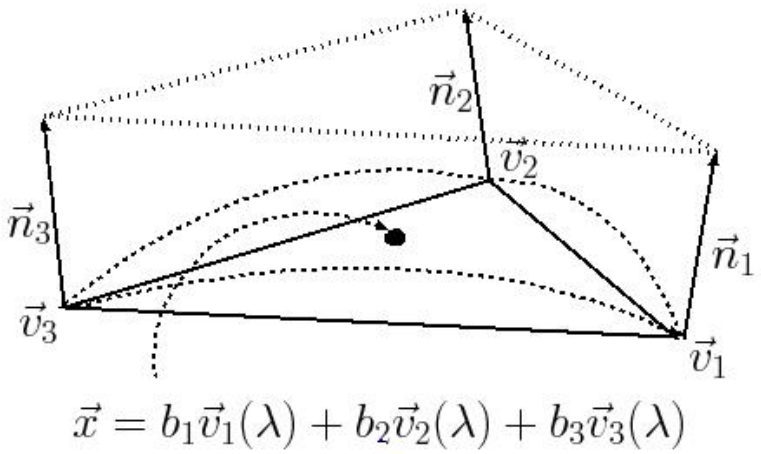


FIG. 4.4. *The representation of a point $\vec{x}$ on a algebraic patch $\vec{v}_1\vec{v}_2\vec{v}_3$.*

If we write the parameters as a vector $\vec{b} = (b_1, b_2, \lambda)$, the normal derivative of the electrostatic potential can be written in the following form using the chain rule

$$\begin{aligned} \tfrac{\partial\phi}{\partial\vec{n}}(\vec{x}) &= \vec{n}\cdot\nabla\phi(\vec{x}) = \vec{n}\cdot\begin{bmatrix}\frac{\partial\phi}{\partial x_1}(\vec{x})\\ \frac{\partial\phi}{\partial x_2}(\vec{x})\\ \frac{\partial\phi}{\partial x_3}(\vec{x})\end{bmatrix} \\ &= [n_1^{(x)}, n_2^{(x)}, n_3^{(x)}]\cdot\begin{bmatrix}\frac{\partial b_1}{\partial x_1} & \frac{\partial b_2}{\partial x_1} & \frac{\partial\lambda}{\partial x_1}\\ \frac{\partial b_1}{\partial x_2} & \frac{\partial b_2}{\partial x_2} & \frac{\partial\lambda}{\partial x_2}\\ \frac{\partial b_1}{\partial x_3} & \frac{\partial b_2}{\partial x_3} & \frac{\partial\lambda}{\partial x_3}\end{bmatrix}\begin{bmatrix}\frac{\partial\phi}{\partial b_1}(\vec{x})\\ \frac{\partial\phi}{\partial b_2}(\vec{x})\\ \frac{\partial\phi}{\partial\lambda}(\vec{x})\end{bmatrix} \end{aligned} \tag{4.3}$$

where $\vec{x} = (x_1, x_2, x_3)^T$ and $\vec{n}(\vec{x}) = (n_1^{(x)}, n_2^{(x)}, n_3^{(x)})^T$ is the unit vector of the normal at $\vec{x}$.

We can derive each term in the equation (4.3) in terms of the parametric parameters of the algebraic spline model. First,

$$\begin{aligned} \begin{bmatrix}\frac{\partial b_1}{\partial x_1}\\ \frac{\partial b_1}{\partial x_2}\\ \frac{\partial b_1}{\partial x_3}\end{bmatrix} &= (\vec{v}_1 - \vec{v}_3) + \lambda(\vec{n}_1 - \vec{n}_3) \\ \begin{bmatrix}\frac{\partial b_2}{\partial x_1}\\ \frac{\partial b_2}{\partial x_2}\\ \frac{\partial b_2}{\partial x_3}\end{bmatrix} &= (\vec{v}_2 - \vec{v}_3) + \lambda(\vec{n}_2 - \vec{n}_3) \\ \begin{bmatrix}\frac{\partial\lambda}{\partial x_1}\\ \frac{\partial\lambda}{\partial x_2}\\ \frac{\partial\lambda}{\partial x_3}\end{bmatrix} &= b_1\vec{n}_1 + b_2\vec{n}_2 + b_3\vec{n}_3 \end{aligned} \tag{4.4}$$

where $\vec{v}_i = (v_{i1}, v_{i2}, v_{i3})^T$ and its unit normal vector $\vec{n}_i = (n_{i1}, n_{i2}, n_{i3})^T$ for $i = 1, 2, 3$. Then, we approximate the electrostatic potential function in the region by

$$\begin{aligned} \phi(\vec{x}) &= b_1((1-\lambda)\phi(\vec{v}_1) + \lambda\phi(\vec{v}_1 + \vec{n}_1)) + b_2((1-\lambda)\phi(\vec{v}_2) + \lambda\phi(\vec{v}_2 + \vec{n}_2)) \\ &\quad +(1 - b_1 - b_2)((1-\lambda)\phi(\vec{v}_3) + \lambda\phi(\vec{v}_3 + \vec{n}_3)), \end{aligned} \tag{4.5}$$

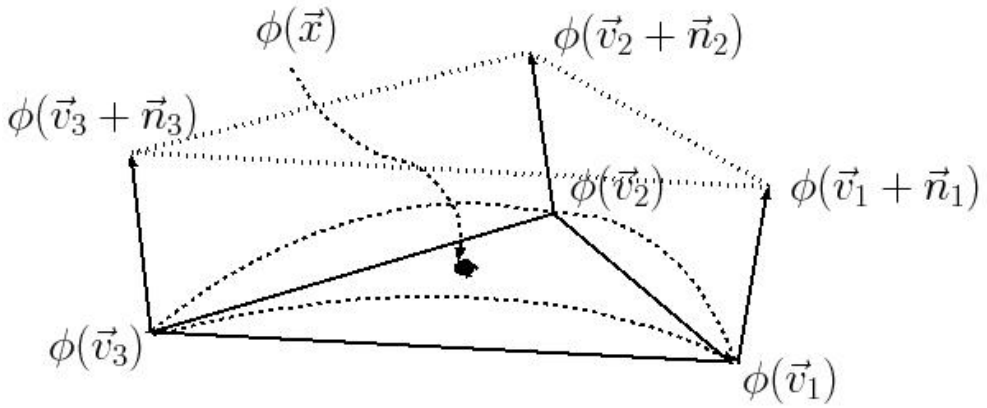


FIG. 4.5. *The representation of electrostatic potential $\phi(\vec{x})$ at a point $\vec{x}$ on a algebraic patch $\vec{v}_1\vec{v}_2\vec{v}_3$.*

and the derivative of the potential to the coordinate $\vec{b}$ becomes

$$\begin{aligned}
&\tfrac{\partial\phi}{\partial b_1}(\vec{x}) = ((1-\lambda)(\phi(\vec{v}_1)-\phi(\vec{v}_3))+\lambda(\phi(\vec{v}_1+\vec{n}_1)-\phi(\vec{v}_3+\vec{n}_3)))\\
&\tfrac{\partial\phi}{\partial b_2}(\vec{x}) = ((1-\lambda)(\phi(\vec{v}_2)-\phi(\vec{v}_3))+\lambda(\phi(\vec{v}_2+\vec{n}_2)-\phi(\vec{v}_3+\vec{n}_3)))\\
&\tfrac{\partial\phi}{\partial\lambda}(\vec{x}) = b_1(\phi(\vec{v}_1+\vec{n}_1)-\phi(\vec{v}_1))+b_2(\phi(\vec{v}_2+\vec{n}_2)-\phi(\vec{v}_2))+b_3(\phi(\vec{v}_3+\vec{n}_3)-\phi(\vec{v}_3))
\end{aligned} \tag{4.6}$$

Finally, we can get the normal derivative of electrostatic potential in the equation (4.3) in this A-patch by combining the above two equations (4.4) and (4.6).

The unknown electrostatic potential and its normal derivatives at the Gaussian quadrature points, in the matrix form of PB BIEs can be derived from the parametric representation of the electrostatic potential at the positions of the vertices and the vertices with a displacement of its unit normal. Here, the parametric form of a Gaussian point $\vec{y}_{jm}$ on the element $\bar{\Gamma}_j$ is

$$\vec{y}_{jm} = b_{m1}\vec{v}_{j1}(\lambda_{jm}) + b_{m2}\vec{v}_{j2}(\lambda_{jm}) + (1-b_{m1}-b_{m2})\vec{v}_{j3}(\lambda_{jm})$$

and the electrostatic potential $\phi(\vec{y}_{jm})$ and normal derivative of electrostatic potential $\frac{\partial\phi}{\partial\vec{n}_{jm}^{(y)}}(\vec{y}_{jm})$ at this point are computed using the equations (4.5) and (4.3).

4.4. Numerical linear system. The matrix to map the index of the vertices $\vec{v}_{j1}\vec{v}_{j2}\vec{v}_{j3}$ of some element Γ_j to the index of vertices of the surface is

$$E_{ij} = \begin{cases} 1 & \text{if } \vec{v}_{(i/3)(i \bmod 3)} = \vec{\mathrm{v}}_j \\ 1 & \text{if } \vec{v}_{(i/3-L)(i \bmod 3)} = \vec{\mathrm{v}}_{j-P} \\ 0 & \text{otherwise} \end{cases}$$

This matrix maps the index of electrostatic potential from the vertices of an element to the vertices of the triangulation.

$$\begin{aligned}
\phi(\vec{v}_{it}) &= \textstyle\sum_{j=1}^{P} E_{(3i+t)(j)}\phi(\vec{\mathrm{v}}_j), && i=1,\cdots,L \text{ and } t=1,2,3,\\
\phi(\vec{v}_{it}+\vec{n}_{it}) &= \textstyle\sum_{j=1}^{P} E_{(3(i+L)+t)(j+P)}\phi(\vec{\mathrm{v}}_j+\vec{\mathrm{n}}_j), && i=1,\cdots,L \text{ and } t=1,2,3
\end{aligned}$$

We define

- the coefficient matrix of the boundary integral equations in the equation (4.1) to be A
- the transform matrix from the unknowns in the equation (4.1) to the parametric representation of algebraic spline model to be B. The detailed derivation of the matrix form is in the appendix.
- the matrix mapping the vertex index of the patches to the vertex index of the surfaced to be E.

The final linear system will be

$$ABE\begin{bmatrix}\phi(\vec{\mathrm{v}}_1)\\ \phi(\vec{\mathrm{v}}_2)\\ \vdots\\ \phi(\vec{\mathrm{v}}_P)\\ \phi(\vec{\mathrm{v}}_1+\vec{\mathrm{n}}_1)\\ \phi(\vec{\mathrm{v}}_2+\vec{\mathrm{n}}_2)\\ \vdots\\ \phi(\vec{\mathrm{v}}_P+\vec{\mathrm{n}}_P)\end{bmatrix}=\begin{bmatrix}Q_{01}\\ Q_{02}\\ \vdots\\ Q_{0P}\\ Q'_{01}\\ Q'_{02}\\ \vdots\\ Q'_{0P}\end{bmatrix}$$

where $Q_{0i}=\sum_{k=1}^{n_c}\frac{q_k}{\epsilon_I}G_0(\vec{\mathrm{v}}_i,\vec{\mathrm{x}}_k), i=1,\cdots,P$ and $Q'_{0i}=\sum_{k=1}^{n_c}\frac{q_k}{\epsilon_I}\frac{\partial G_0}{\partial\vec{\mathrm{n}}_i}(\vec{\mathrm{v}}_i,\vec{\mathrm{x}}_k), i=1,\cdots,P$. The unknowns are now located on the vertices of triangulation $\{\vec{\mathrm{v}}_i\}_{i=1}^P$.

5. Postprocessing.

5.1. Interior and exterior electrostatic potential. If the PB electrostatic potential on the surface Γ is computed, we can obtain the electrostatic potential inside Γ (the interior region Ω) from the equation 5.1,

$$\begin{aligned}&\phi(\vec{x})=\oint_\Gamma(\tfrac{\epsilon_E}{\epsilon_I}\tfrac{\partial G_\kappa(\vec{x},\vec{y})}{\partial\vec{n}^{(y)}}-\tfrac{\partial G_0(\vec{x},\vec{y})}{\partial\vec{n}^{(y)}})\phi(\vec{y})d\vec{y}\\&+\oint_\Gamma(G_0(\vec{x},\vec{y})-G_\kappa(\vec{x},\vec{y}))\tfrac{\partial\phi(\vec{y})}{\partial\vec{n}^{(y)}}d\vec{y}+\sum_{k=1}^{n_c}\tfrac{q_k}{\epsilon_I}G_0(\vec{x},\vec{\mathrm{x}}_k)\end{aligned}\tag{5.1}$$

if $\vec{x}\in\Omega$ and the electrostatic potential outside Γ (the exterior region $\mathbb{R}^3-\Omega$) from the equation 5.2 [1].

$$\begin{aligned}&\tfrac{\epsilon_E}{\epsilon_I}\phi(\vec{x})=\oint_\Gamma(\tfrac{\epsilon_E}{\epsilon_I}\tfrac{\partial G_\kappa(\vec{x},\vec{y})}{\partial\vec{n}^{(y)}}-\tfrac{\partial G_0(\vec{x},\vec{y})}{\partial\vec{n}^{(y)}})\phi(\vec{y})d\vec{y}\\&+\oint_\Gamma(G_0(\vec{x},\vec{y})-G_\kappa(\vec{x},\vec{y}))\tfrac{\partial\phi(\vec{y})}{\partial\vec{n}^{(y)}}d\vec{y}+\sum_{k=1}^{n_c}\tfrac{q_k}{\epsilon_I}G_0(\vec{x},\vec{\mathrm{x}}_k)\end{aligned}\tag{5.2}$$

if $\vec{x}\in\mathbb{R}^3-\Omega$. The numerical solution of patch quadrature is applied to compute these surface integrals.

5.2. Numerical solution of electrostatic free energy and force. The electrostatic free energy is derived by Sharp using the variation principle from linearized PB equation[38]. The formulation of electrostatic free energy G_{pol} in the equation (1.2) is computed by the charge and the electrostatic potential at the atomic centers of a molecule . The electrostatic potential at the atomic centers can be computed through the interior electrostatic potential equation (5.1).

$$G_{pol}=\int_\Omega\phi_{\mathrm{rxn}}(\vec{x})\sum_{k=1}^{n_c}q_k\delta(\vec{x}-\vec{\mathrm{x}}_k)d\vec{x}=\frac{1}{2}\sum_{k=1}^{n_c}\phi_{\mathrm{rxn}}(\vec{\mathrm{x}}_k)q_k$$

where the difference between the electrostatic potential in the solvent and the air, also called the reaction field electrostatic potential, is $\phi_{\mathrm{rxn}}(\vec{x})=\phi_{\mathrm{sol}}(\vec{x})-\phi_{\mathrm{gas}}(\vec{x})$.

Because the ionic strength is zero in the air, the Poisson kernel $G_0(\vec{x},\vec{y})$ and Poisson-Boltzmann kernel $G_\kappa(\vec{x},\vec{y})$ are the same. At the same time, the dielectric constants inside and outside the molecule are the same: $\epsilon_E = \epsilon_I$, so the two surface integral terms in the equation (5.1) are zero. It indicates that the interior electrostatic potential in the air is

$$\phi_{\text{gas}}(\vec{x}) = \sum_{k=1}^{n_c} \frac{q_k}{\epsilon_I} G_0(\vec{x}, \vec{\text{x}}_k).$$

Therefore, the reaction field electrostatic potential $\phi_{\text{rxn}}(\vec{x})$ is the surface integral term of the electrostatic potential in the solvent $\phi_{\text{sol}}(\vec{x})$.

$$\phi_{\text{rxn}}(\vec{x}) = \int_\Gamma (\frac{\epsilon_E}{\epsilon_I} \frac{\partial G_\kappa(\vec{x},\vec{y})}{\partial \vec{n}^{(y)}} - \frac{\partial G_0(\vec{x},\vec{y})}{\partial \vec{n}^{(y)}}) \phi_{\text{sol}}(\vec{y}) d\vec{y} + \int_\Gamma (G_0(\vec{x},\vec{y}) - G_\kappa(\vec{x},\vec{y})) \frac{\partial \phi_{\text{sol}}(\vec{y})}{\partial \vec{n}^{(y)}} d\vec{y}$$

In Sharp's definition, the electrostatic free energy is actually composed of three terms and the above formulation is called the reaction field energy. In the planar case, the remaining two terms, the electrostatic stress and osmotic pressure terms, are canceled out.

Based on Sharp's definition, Gilson expressed the electrostatic force F_{pol} through a variational derivation of force expression from electrostatic free energy [22].

$$F_{pol} = -\int_\Omega \left(\rho_c(\vec{x}) E(\vec{x}) - \frac{1}{2} E(\vec{x})^2 \nabla \epsilon(\vec{x}) - k_B T \sum_i [c_i (e^{-z_i \phi(\vec{x})/k_B T} - 1) \nabla \lambda(\vec{x})] \right) d\vec{x} \tag{5.3}$$

where $\rho_c(\vec{x}) = 4\pi \sum_{k=1}^{n_c} q_k \delta(\vec{x} - \vec{\text{x}}_k)$ and the electric field is the gradient of the electrostatic potential, $E(\vec{x}) = -\nabla \phi(\vec{x})$.

The terms in the electrostatic force equation (5.3) are called the reaction field force, the dielectric boundary force, and the ionic boundary force.

- the reaction field force

$$F_{\text{rxn}} = -\int_\Omega \rho_c(\vec{x}) E(\vec{x}) d\vec{x} = -4\pi \sum_{k=1}^{n_c} q_k E(\vec{\text{x}}_k)$$

- the dielectric boundary force

$$F_{\text{db}} = \int_\Omega \frac{1}{2} E(\vec{x})^2 \nabla \epsilon(\vec{x}) d\vec{x}$$

- the ionic boundary force

$$F_{\text{ib}} = \int_\Omega k_B T \sum_i [c_i (e^{-z_i \phi(\vec{x})/k_B T} - 1) \nabla \lambda(\vec{x})] d\vec{x}$$

where $\lambda(\vec{x})$ is 1 outside Γ and 0 inside Γ.

For an atom of a molecule which doesn't form part of a dielectric or ionic boundary, the dielectric boundary force and ionic boundary force are zero and only the reaction field term is necessary $F_{pol} = F_{\text{rxn}}$. For an atom of a molecule which forms part of a dielectric or ionic boundary, also called a solvent-exposed atom, all three terms should be counted $F_{pol} = F_{\text{rxn}} + F_{\text{db}} + F_{\text{ib}}$.

In order to compute these different force terms, we have to approximate the electric field $E(\vec{x})$ at all atomic centers and molecular surface points. The electric field $\vec{x}$ is approximated using the gradient of interior and exterior electrostatic potential (5.1) and (5.2).

$$\begin{array}{ll}\nabla\phi(\vec{x}) & = \oint_\Gamma \left[\nabla G_0(\vec{x},\vec{y}) - \nabla G_\kappa(\vec{x},\vec{y})\right] \frac{\partial\phi}{\partial\vec{n}}(\vec{y})d\vec{y} \\ & + \oint_\Gamma \left[\epsilon\nabla\frac{\partial G_\kappa}{\partial\vec{n}}(\vec{x},\vec{y}) - \nabla\frac{\partial G_0}{\partial\vec{n}}(\vec{x},\vec{y})\right]\phi_I(\vec{y})d\vec{y} \\ & + \sum_{k=1}^{n_c}\frac{q_k}{\epsilon_I}\nabla_{\vec{x}}G_0(\vec{x},\vec{\mathrm{x}}_k)\end{array}$$

if $\vec{x}\in\Omega$ and

$$\begin{array}{ll}\epsilon\nabla\phi(\vec{x}) & = \oint_\Gamma \left[\nabla G_0(\vec{x},\vec{y}) - \nabla G_\kappa(\vec{x},\vec{y})\right] \frac{\partial\phi}{\partial\vec{n}}(\vec{y})d\vec{y} \\ & + \oint_\Gamma \left[\epsilon\nabla\frac{\partial G_\kappa}{\partial\vec{n}}(\vec{x},\vec{y}) - \nabla\frac{\partial G_0}{\partial\vec{n}}(\vec{x},\vec{y})\right]\phi(\vec{y})d\vec{y} \\ & + \sum_{k=1}^{n_c}\frac{q_k}{\epsilon_I}\nabla G_0(\vec{x},\vec{\mathrm{x}}_k)\end{array} .$$

if $\vec{x}\in\mathbb{R}^3-\Omega$.

The reaction field force can be computed using the electric field at these atomic centers. Nevertheless, we can not handle the dielectric boundary force and ionic boundary force so easily. In order to compute these two forces, we have to compute the $\nabla\epsilon(\vec{x})$ term in dielectric boundary force and the $\nabla\lambda(\vec{x})$ term in the ionic boundary force. In this paper, we used a distance-dependent model derived by Im to approximate the dielectric function $\epsilon(\vec{x})$ and the ionic boundary function $\lambda(\vec{x})$ [26]. The details of this model are described in the appendix.

6. Implementation Details and Experimental Results. In this paper, we developed a platform of data structures and routines of a 3D boundary element solver, called PB-CFMM (Poisson-Boltzmann - curved fast multipole method). We implemented all the above methodologies for solving the PB electrostatic problem in PB-CFMM and it is callable from TexMol [6]. As we described in the above section, the input of the solver is the 3D atomic structure and the triangular mesh of the target molecule. Its properties including electrostatic potential, electrostatic free energy and forces are computed.

Here, PETSc (Portable, Extensible Toolkit for Scientific Computation) is used for the solution of the PB linear system [10]. It supports matrix-free Krylov iterative method (e.g. GMRES, CG) which do not require explicit storage of the matrix. The explicit matrix is replaced by a user-defined evaluation of matrix vector production. Here, we use kernel independent fast multipole method, KiFMM, to do linear-time evaluation [43].

The computational steps for the solution of the PB electrostatic problem are concluded in the following list.

Structure preparation Prepare structures for continuum electrostatic calculations using "PDB2PQR". The main task of "PDB2PQR" assigning charge and radius parameters to the atomic PDB structure [20]. Since many biomolecular structures in the Protein Data Bank do not contain hydrogen atoms and a fraction of heavy atoms, this software also checks and rebuilds those missing hydrogen and heavy atoms to biomolecular structures based on standard amino acid topologies.

Molecular surface extraction Extract the molecular surface from the level set computed through geometric flow evolution [5].

Triangular mesh generation Compute high-qualified linear triangular boundary elements using octree-based dual contouring method [45].

C^1 **A-spline modeling** Compute the cubic algebraic spline over the triangular elements.

Numerical solution Compute electrostatic potential by solving the PB equation using our boundary element solver "PB-CFMM" with the fast summation method using "KiFMM" [43].

- construct KiFMM models for PB kernels on the algebraic spline model,
- solve the linear system using GMRES iterative method with KiFMM.

Post-processing compute electrostatic free energy and forces using electrostatic potential.

In this paper, we solve the linear Poisson-Boltzmann system using the iterative method, GMRES with the initialization of electrostatic potential using the coulombic equation. The relative residual tolerance is 10^{-7} and number of Gaussian quadrature points per triangle is 7. We then gathered 71 sets of ligand-receptor protein complexes (ligand,receptor,ligand-receptor complex) from RCSB protein data bank (PDB). These are used for the evaluation of the PB electrostatic computation.

The first experiment is an analytical numerical error evaluation with a given potential function. This experiment is applied for understanding the reliability and efficiency of our PB solution. In the second experiment, we compute and compare real electrostatic results of these proteins between our boundary element solvers and DelPhi II finite difference solver [37, 33]. Then, we study the performance of our system by controlling different effective factors. All experiments are done on a linux machine with Dual Core AMD Opteron processor 280 with 4 GB memory. We discussed and analyze the experimental results in the following experiments.

6.1. Analytical numerical evaluation. In the first experiment, we evaluate the efficiency and accuracy of numerical computation of electrostatic potentials and their normal derivatives using regular or consistent PB boundary element solvers with fast matrix-vector product evaluation. The numerical test is done with the assumption that electrostatic potential is given as an exponential function $\tilde{\phi}(\vec{x}) = e^{-\|\vec{x}\|^2}$ and the normal derivative of potential as the normal derivative of this exponential function $\frac{\partial\tilde{\phi}(\vec{x})}{\partial\vec{n}^{(x)}} = -2e^{-\|\vec{x}\|^2}(\vec{x}\cdot\vec{n}^{(x)})$. We calculate $Q(\vec{x})$ and $R(\vec{x})$ on the vertices of the triangular meshes by evaluating the left hand sight of dBIEs (3.5) and (3.6).

$$\begin{aligned} Q(\vec{x}) &= \tfrac{1}{2}(1+\tfrac{\epsilon_E}{\epsilon_I})\tilde{\phi}(\vec{x}) + \int_\Gamma (\tfrac{\partial G_0(\vec{x},\vec{y})}{\partial\vec{n}^{(y)}} - \tfrac{\epsilon_E}{\epsilon_I}\tfrac{\partial G_\kappa(\vec{x},\vec{y})}{\partial\vec{n}^{(y)}})\tilde{\phi}(\vec{y})d\vec{y} \\ &\quad -\int_\Gamma (G_0(\vec{x},\vec{y}) - G_\kappa(\vec{x},\vec{y}))\tfrac{\partial\tilde{\phi}(\vec{y})}{\partial\vec{n}^{(y)}}d\vec{y} \\ R(\vec{x}) &= \tfrac{1}{2}(1+\tfrac{\epsilon_I}{\epsilon_E})\tfrac{\partial\phi(\vec{x})}{\partial\vec{n}^{(x)}} + \int_\Gamma (\tfrac{\partial^2 G_0(\vec{x},\vec{y})}{\partial\vec{n}^{(x)}\partial\vec{n}^{(y)}} - \tfrac{\partial^2 G_\kappa(\vec{x},\vec{y})}{\partial\vec{n}^{(x)}\partial\vec{n}^{(y)}})\phi(\vec{y})d\vec{y} \\ &\quad -\int_\Gamma (\tfrac{\partial G_0(\vec{x},\vec{y})}{\partial\vec{n}^{(x)}} - \tfrac{\epsilon_I}{\epsilon_E}\tfrac{\partial G_\kappa(\vec{x},\vec{y})}{\partial\vec{n}^{(x)}})\tfrac{\partial\phi(\vec{y})}{\partial\vec{n}^{(y)}}d\vec{y} \end{aligned}$$

We evaluate the electrostatic potential and its normal derivative on the vertices of triangulation computed using our boundary element solver by the relative errors

$$\frac{\sqrt{\sum_{i=1}^{P}|\phi(\vec{\mathrm{v}}_i) - \tilde{\phi}(\vec{\mathrm{v}}_i)|^2}}{\sqrt{\sum_{i=1}^{P}|\phi(\vec{\mathrm{v}}_i)|^2}}$$

and

$$\frac{\sqrt{\sum_{i=1}^{P}|\frac{\partial\phi}{\partial\vec{\mathrm{n}}_i}(\vec{\mathrm{v}}_i) - \frac{\partial\tilde{\phi}}{\partial\vec{\mathrm{n}}_i}(\vec{\mathrm{v}}_i)|^2}}{\sqrt{\sum_{i=1}^{P}|\frac{\partial\phi}{\partial\vec{\mathrm{n}}_i}(\vec{\mathrm{v}}_i)|^2}}.$$

# of A-patches	evaluation method	relative error (ϕ)	# of iterations	compute time (seconds)
2000	direct	6.380×10^{-7}	35.21	165.063
	KiFMM	6.379×10^{-7}	35.21	60.858
5000	direct	9.472×10^{-7}	40.88	1237.451
	KiFMM	1.309×10^{-6}	41.72	216.232
10000	direct	2.424×10^{-7}	46.71	5423.711
	KiFMM	2.635×10^{-7}	46.83	528.605
63444.81∗	KiFMM	4.678×10^{-7}	38.41	3012.344

TABLE 6.1

The results of analytical experiments computed using PB BEM solver with different number of A-patches for 213 *molecules; column 1 is the number of triangles (* is the average number of A-patches of the original triangular mesh of* 213 *molecular surfaces); column 2 is the type of evaluation method of matrix-vector product; column 3 is the average relative errors of potential ϕ and $\tilde{\phi}$; column 4 is the number of iterations for the convergence; column 5 is the computation time in seconds.*

# of A-patches	numerical method	relative error (ϕ)	relative error ($\frac{\partial \phi}{\partial \vec{n}}$)	# of iterations	compute time (s)
2000	regular	6.379×10^{-7}	1.442×10^{-3}	35.21	60.858
	consistent	5.208×10^{-7}	1.533×10^{-7}	36.17	62.388
5000	regular	1.309×10^{-6}	9.454×10^{-4}	41.72	216.232
	consistent	9.081×10^{-7}	5.900×10^{-7}	45.83	258.544
10000	regular	2.635×10^{-7}	2.850×10^{-3}	46.83	528.605
	consistent	2.769×10^{-7}	3.669×10^{-7}	45.82	492.002
63444.81∗	regular	4.678×10^{-7}	1.498×10^{-3}	38.41	3012.344
	consistent	4.921×10^{-7}	4.944×10^{-7}	39.09	3107.15

TABLE 6.2

The results of analytical experiments computed using PB BEM solver with different number of A-patches for 213 *proteins; column 1 is the number of triangles (* is the average number of A-patches of the original triangular mesh of* 213 *molecular surfaces); column 2 is the type of numerical solution of boundary element method; column 3 is the average relative errors of potential ϕ and $\tilde{\phi}$; column 4 is the average relative errors of normal derivative of potential $\frac{\partial \phi}{\partial \vec{n}}$ and $\frac{\partial \tilde{\phi}}{\partial \vec{n}}$; column 5 is the number of iterations for the convergence; column 6 is the computation time in seconds.*

Table 6.1 shows the average relative error of potential and compute time of the evaluations of whole proteins.

In this experiment, we observe that our fast boundary element solver is much more efficient than the direct solver because fast multipole methods are linear-time algorithms with high accuracy. With triangular meshes in different resolutions, small relative errors of KiFMM indicate that our fast multipole method works well in solving the PB linear system.

On the other hand, the normal derivative of potential on the molecular surface is taken as unknown in the original derivative boundary integral equations. In our paper, we used the parametric formulation of the algebraic spline model to derive the normal derivative of potential. Here, we compute the potential and normal derivative of potential using regular or consistent numerical methods and compare the relative errors and computation time in Table 6.2.

The relative errors of potential are similar in both numerical solutions but those of the normal derivative of potential are not. The normal derivative of potential

numerical method	solver name	# of grids/ # of A-patches	G_{pol} $(kcal/mol)$	relative error	compute time (seconds)
FDM	Delphi II	65^3	-82.943	2.523%	11.39
FDM	Delphi II	129^3	-82.228	1.642%	95.35
FDM	Delphi II	193^3	-82.144	1.244%	286.65
BEM	PB-CFMM	1436	-80.926	0.032%	4.63

TABLE 6.3

PB electrostatic free energy of a unit sphere with single charge computed using different numerical method; column 1 is the numerical method; column 2 is the name of the solver; column 3 is the number of grids for FDM and number of A-patches for BEM; column 4 is the electrostatic free energy G_{pol} $(kcal/mol)$; column 5 is the relative error of electrostatic free energy G_{pol}. As a reference, the exact electrostatic free energy is -80.9 $kcal/mol$ with the interior and exterior dielectric constatnt 2 and 80; column 6 is computational time in seconds.

numerical method	BEM	BEM	BEM	BEM	FDM
# of A-patches/grids	2000	5000	10000	63444.81*	193^3
avg. # of iterations	35.21	41.72	37.14	28.86	-
max # of iterations	91	98	93	84	-
min # of iterations	13	15	19	12	-
avg. compute time (s)	60.86	216.23	418.96	1506.18	408.66
max compute time (s)	165.6	553.69	901.91	7578.36	2705.42
min compute time (s)	19.99	56.88	153.87	221.77	69.11
avg. compute time per iter (s)	2.19	6.71	13.40	61.46	-
avg. correlation of G_{pol}	0.852	0.927	0.948	0.960	-

TABLE 6.4

The statistics of the experiments including the average, maximum and minimum of the number of iterations, compute time, compute time per iterations and the correlation with Delphi FDM(193^3 grids) of our BEM with different number of A-patches for 213 molecules (71 sets of ligands, receptors and ligand-receptor complexes). (the average number of A-patches of the original triangular mesh of 213 molecular surfaces)*

computed using the parametric formulation is more accurate than that computed using the regular solution. This indicates that the relation between potential and normal derivative of potential is accurate and consistent when we used our parametric formulation (C.2).

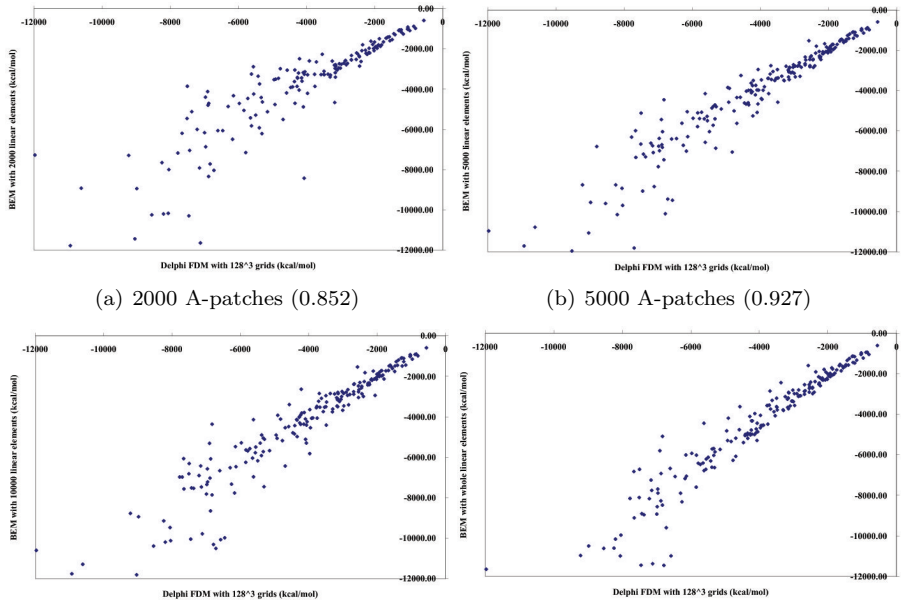
6.2. Poisson-Boltzmann electrostatic solvation free energies.

6.2.1. A unit sphere with single positive charge. Only in some ideal cases, we can derive the electrostatic free energy analytically from the PB equation. To test the correctness of the PB solver, we compute the electrostatic free energy for a unit sphere with $+1e$ single charge placed at its center and the results are shown in Table 6.3. In this ideal case, the electrostatic free energy is -80.9 $kcal/mol$ with the interior and exterior dielectric constants 2 and 80. We can see that our BEM solution is more accurate and efficient than FDM in this case. The relative error of G_{pol} computed using BEM is much lower than that of FDM with any grid size. BEM also costs less computational time than FDM.

6.2.2. A list of ligand-receptor complexes. Using PB BEM solver, we compute electrostatic free energy for all proteins in the list of ligand-receptor complexes. In Table 6.4, we show the statistics of the PB computation using our BEM solution.

We compute the average, maximum and minimum iteration number and compute time from the results of all 213 proteins (71×3). The average iteration number is smaller than 40 and not related to the number of A-patches. The computational time includes the time of solving surface and per-atom electrostatic potential and computing electrostatic free energy. The evaluation time per iteration is linearly proportional to the number of A-patches since KiFMM is a linear time solver of fast matrix-vector product.

Meanwhile, we also observe the influence of the mesh quality to the convergence speed of iterative solution. We use average aspect ratio (twice of the ratio of the incircle radius to the circumcircle radius of a triangle) of a mesh to measure the quality of the mesh. After we compute an initial triangular mesh for the molecular surface of a protein, we applied a geometric flow algorithm to improve the quality of the mesh [46]. We observe that the averge aspect ratio of a mesh goes from 0.326 to 0.430 after improving the mesh quality using geometric flow algorithm. At the same time, the average number of iterations goes from 43.41 to 28.86. It indicates that better mesh quality will lead to faster convergence speed. The correlation of our



(a) 2000 A-patches (0.852)

(b) 5000 A-patches (0.927)

(c) 10000 A-patches (0.948)

(d) 63444.81* A-patches (0.960)

FIG. 6.1. *The comparison of electrostatic free energy (kcal/mol) of 213 proteins (71 sets of ligand-receptor complexes) between BEM and FDM with* 193^3 *grids with the correlation in parentheses.*

BEM solver to Delphi II FDM solver with 193^3 grids are shown in the figures 6.1. Each point in the chart indicates PB electrostatic solvation free energy of a protein (ligand, receptor or their complex) computed using BEM or FDM. According to the value of correlation, we found that the more patches we used, the higher a correlation we obtained.

6.3. Poisson-Boltzmann electrostatic potential. We also compute the real PB electrostatic potential for all the proteins. In Figure 6.3, we show the electrostatic potential on the molecular surface of an example in the protein list. PB electrostatic

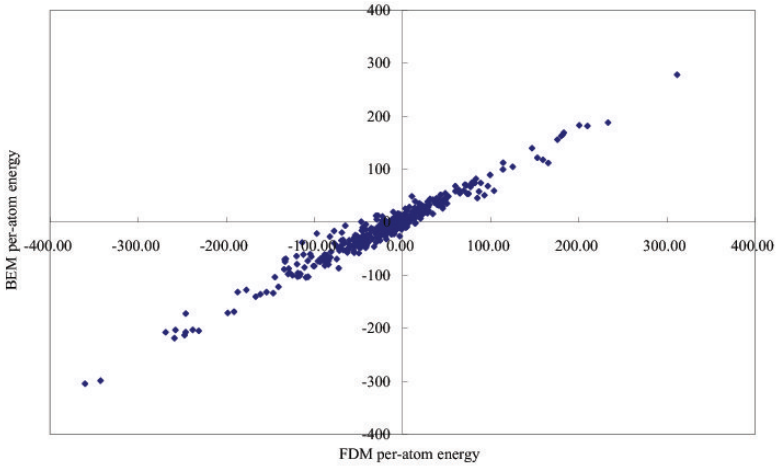
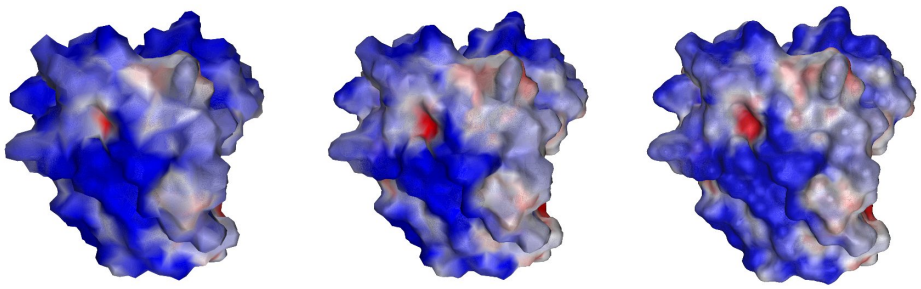
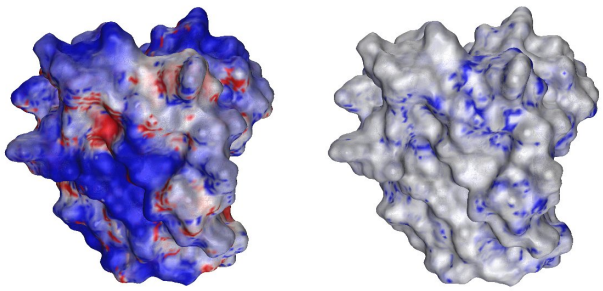


FIG. 6.2. *The comparison of per-atom energy (KJ/mol) of Bovine Chymotrypsinogen*A (PDB id: 1CGI) between FDM with* 193^3 *grids and BEM with* 10000 *A-patches. Each point indicates the electrostatic solvation free energy of an atom.*

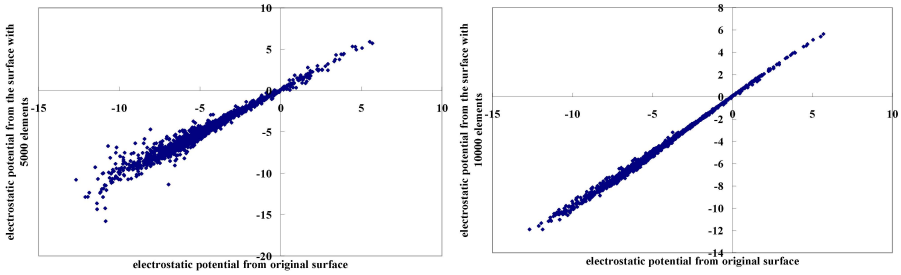


(a) 5000 A-patches (b) 10000 A-patches (c) 54592 A-patches



(d) FDM with 193^3 grids (e) difference between BEM and FDM

FIG. 6.3. *The PB electrostatic potential on molecular surface of nuclear transport factor 2 (PDB id: 1A2K) with different resolutions. (e) shows the difference of PB electrostatic potential between BEM and FDM. The color is going from red (potential of* $-3.8\ k_bT/e_c$*) to blue (potential of* $+3.8\ k_bT/e_c$*).*



(a) 5000 A-patches (0.895) (b) 10000 A-patches (0.998)

FIG. 6.4. *The comparison of electrostatic potential between the molecular surface of Bovine Chymotrypsinogen*A (PDB id: 1CGI) with different resolutions (the correlation in parentheses) where the average number of A-patches of the original surface is* 63444.81.

numerical method	solver name	# of grids/ # of A-patches	inverse discretization length scale (1.0/Å)	correlation (ϕ)
FDM	Delphi II	65^3	0.333	0.965
FDM	Delphi II	129^3	0.667	0.977
FDM	Delphi II	193^3	1.000	-
BEM	PB-CFMM	5000	0.367	0.944
BEM	PB-CFMM	10000	0.732	0.968
BEM	PB-CFMM	63444.81∗	5.301	0.981

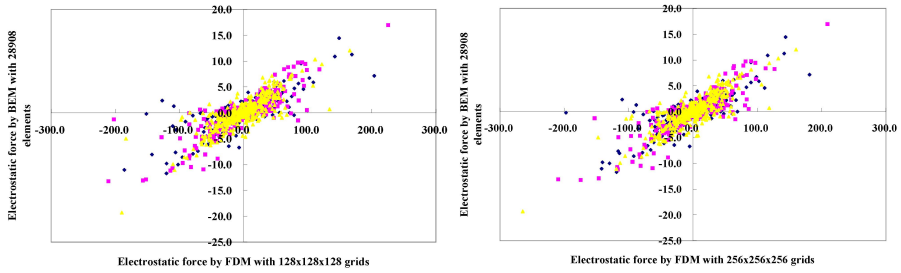
TABLE 6.5

Average experimental results of PB electrostatic potential computation for 213 proteins (71 sets of ligand-receptor complexes); column 1 is the numerical method; column 2 is the name of the solver; column 3 is the number of grids for FDM and number of A-patches for BEM; column 4 is the inverse discretization length scale of each grid or A-patches; column 5 is the correlation of electrostatic potential to FDM with 193^3 *grids.*

potential is computed with different numbers of A-patches. The color of the surface represents the electrostatic potential on the molecular surface, going from red (potential of $-3.8\ k_bT/e_c$) to blue (potential of $+3.8\ k_bT/e_c$) and white is neutral potential. The distribution of electrostatic potential computed using the triangular A-spline models with different resolution are almost the same. The same results can be observed in Figure 6.4 which represents the different of electrostatic potential of a protein (PDB id: 1CGI) computed using A-spline models with different resolutions. The number of A-patches of its original surface is 54592. The correlation of the results computed from the original surface and decimated surface with 10000 A-patches is up to 0.998. It indicates that we can get a similar result using only 1/5 of A-patches. However, if we just use 5000 A-patches, they are not enough to represent the details of the molecular surface and the correlation becomes 0.895.

Figures 6.3 (c) and (d) show the surface electrostatic potential computed using our BEM solver and finite different solver, Delphi II. The distributions of their electrostatic potential are roughly the same. We then compute the difference between them, shown in Figure 6.3 (e). Blue color represents the magnitude of the difference of surface electrostatic potential. We can observe that the large difference occurs only in some small regions. In Table 6.5, we compute electrostatic potential at the points of 65^3 grids using BEM or FDM with different resolutions and compare the results by their correlation to the electrostatic potential computed by FDM with 193^3 grids.

The inverse discretization length scale in the table is the average edge length of triangulation for BEM and distance between grid points for FDM. We can observe that electrostatic potential computed using BEM and FDM is highly correlated.



(a) BEM vs FDM with 128^3 grids (0.8108,0.8449,0.8198)

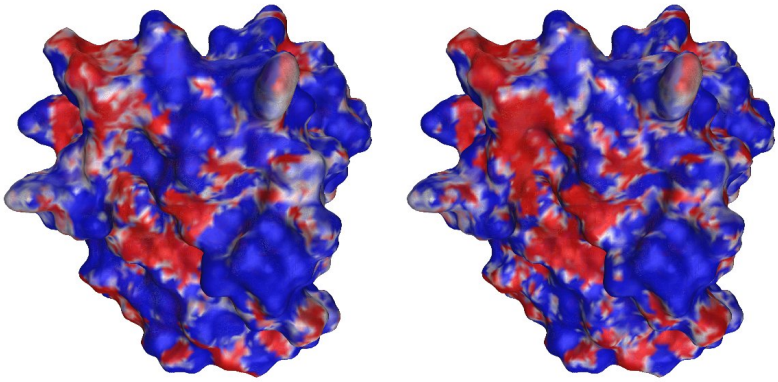
(b) BEM vs FDM with 256^3 grids (0.8118,0.8629,0.8317)

FIG. 6.5. *The relation of per-atom electrostatic force (kcal/mol·Å) of Bovine Chymotrypsinogen*A (PDB id: 1CGI) computed using BEM or FDM where blue,pink,yellow dots indicate x,y,z-dimensional values of forces; (a) BEM with* 28908 *A-patches vs FDM with* 193^3 *grids, the correlations at* x, y, z *dimensions are (0.8108,0.8449,0.8198); (b) BEM with* 28908 *A-patches vs FDM with* 256^3 *grids, the correlations at* x, y, z *dimensions are (0.8118,0.8629,0.8317).*

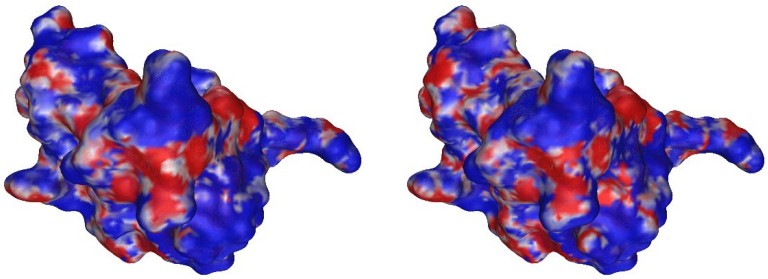
6.4. Poisson-Boltzmann Electrostatic forces. Electrostatic force computation depends on the accurate evaluation of the gradient of electrostatic potential. It requires a very stable electrostatic potential computation. For FDM, we approximate the gradient of electrostatic potential at any specific point based on the electrostatic potential computed on each grid points. On the other hand, for BEM, we can compute the gradient of electrostatic potential at a point using potential computed along three different directions. We can observe the correlation of electrostatic forces between BEM and FDM of an example in Figure 6.5. The correlation becomes higher when the number of grids in FDM increases. It indicates that the electrostatic forces computed using FDM may converge to that computed using BEM. In Figure 6.6, we show PB electrostatic forces of two protein examples (PDB id: 1A2K and 1CGI). The color of the molecular surface represents the inner product of the electrostatic forces and the unit surface normals. The outward force gives a positive inner product and negative otherwise. The color is going from blue (≥ 3.8 $kcal/mol$·Å) to red (≤ -3.8 $kcal/mol$·Å). We can see that the distribution of inward and outward forces computed using BEM and FDM are almost the same.

The electrostatic force computation depends on the accurate computation of the gradient of electrostatic potential and the approximation of the dielectric function and ionic boundary. In this part, we found that if we used the fast multipole method to compute the integrals of three electrostatic force terms, the numerical error will be amplified. Therefore, we still used direct computation to deal with force computation. On the other hand, in both BEM and FDM solutions, we used Im's volume exclusion function to approximate the derivatives of the dielectric function and ionic boundary function in F_{db} and F_{ib}. This approximate function is used for computing the $\nabla\epsilon(\vec{x})$ term in dielectric boundary force and the $\nabla\lambda(\vec{x})$ term in the ionic boundary force.

7. Conclusion. In this paper, we introduce a complete pipeline to solve the linearized Poisson-Boltzmann equation and compute electrostatic potential, energy and forces for biomolecules. The boundary element method is used and the derivation



(a) BEM with 54592 A-patches (b) FDM with 256^3 grids



(c) BEM with 29108 A-patches (d) FDM with 256^3 grids

FIG. 6.6. *The inner product of unit normal vector and PB electrostatic forces on the molecular surface of nuclear transport factor 2 (PDB id: 1A2K) and Bovine Chymotrypsinogen*A (PDB id: 1CGI) with different resolutions. The color is going from blue ($\geq$ 3.8 kcal/mol·Å) to red (≤ -3.8 kcal/mol·Å).*

of boundary integral equations and their numerical treatment are presented. Unlike the original boundary integral equations derived from the Poisson-Boltzmann equation which take normal derivatives of potential as unknowns in linear system, we derive the parametric formulation of the normal derivative of potential based on an algebraic spline model. In the analytical experiment, we observe that our solution gives more accurate normal derivatives of potential than the original solution. In addition, we also compare our numerical results including electrostatic free energy, potential and forces with the state of the art finite difference solution, Delphi II. All the results show that our boundary element solution is an accurate and efficient technique to solve Poisson-Boltzmann electrostatic problems. For the purpose of biological simulation, we developed a visualization application to observe the electrostatic potential and forces on the surfaces of molecules.

8. Acknowledgements. This research was supported in part by NSF grant CNS-0540033 and NIH contracts R01-EB00487, R01-GM074258, R01-GM07308. Sincere thanks to members of CVC for their help in maintaining the software environ-

ment in particular for the use of TexMol (http://cvcweb.ices.utexas.edu/software). Authors also thank to Prof. Lexing Ying of the Dept. of Mathematics for the use of KiFMM (kernel independent fast multipole method). Curved boundary element PB implementation based on KiFMM is called PB-CFMM and its front end is TexMol.

REFERENCES

[1] A.H.Juffer, E.F.F. Botta, B.A.M. van Keulen, A. van der Ploeg, and H.J.C Berensen, *The electric potential of a macromolecule in a solvent: a fundamental approach*, J Chemical Physics, 97 (1991), pp. 144–171.

[2] M. D. Altman, J. P. Bardhan, J. K. White, and B. Tidor, *Accurate solution of multi-region continuum biomolecule electrostatic problems using the linearized poisson-boltzmann equation with curved boundary elements*, J. Comput Chem., 30 (2009), pp. 132–153.

[3] A.Nicholls and B.Honig, *A rapid finite difference algorithm, utilizing successive over-relaxation to solve the Poisson-Boltzmann equation*, J. Comput Chem., 12 (1991), pp. 435–445.

[4] C. Bajaj and G. Xu, *A-splines: Local interpolation and approximation using g_k- continuous piecewise real algebraic curves*, Computer Aided Geometric Design, 16 (1999), pp. 557–578.

[5] ———, *Smooth shell construction with mixed prism fat surfaces*, Geometric Modeling Computing Supplement, 14 (2001), pp. 19–35.

[6] C. L. Bajaj, P. Djeu, V. Siddavanahalli, and A. Thane, *Texmol: Interactive visual exploration of large flexible multi-component molecular complexes*, in Proceedings of the Annual IEEE Visualization Conference'04, Austin, Texas, October 2004, pp. 243–250.

[7] C. L. Bajaj, G. Xu, and Q. Zhang, *Higher-order level-set method and its application in biomolecule surfaces construction*, Journal of Computer Science and Technology, 23 (2008), pp. 1026–1036.

[8] ———, *A fast variational method for the construction of resolution adaptive c^2-smooth molecular surfaces*, Comput. Methods Appl. Mech. Engrg., (2009), p. to appear.

[9] N. Baker, M. Holst, and F. Wang, *Adaptive multilevel finite element solution of the Poisson-Boltzmann equation II: refinement at solvent accessible surfaces in biomolecular systems*, J. Comput Chem., 21 (2000), pp. 1343–1352.

[10] Satish Balay, Kris Buschelman, Victor Eijkhout, William D. Gropp, Dinesh Kaushik, Matthew G. Knepley, Lois Curfman McInnes, Barry F. Smith, and Hong Zhang, *PETSc users manual*, Tech. Report ANL-95/11 - Revision 2.1.5, Argonne National Laboratory, 2004.

[11] R. Bharadwaj, A. Windemuth, S. Sridharan, B. Honig, and A. Nicholls, *The fast multipole boundary element method for molecular electrostatics: An optimal approach for large systems*, J. Comput Chem., 16 (1995), p. 898.

[12] A. J. Bordner and G. A. Huber, *Boundary element solution of the linear Poisson-Boltzmann equation and a multipole method for the rapid calculation of forces on macromolecules in solution*, J. Comput Chem., 24 (2003), pp. 353–367.

[13] A. H. Boschitsch and M. O. Fenley, *Hybrid boundary element and finite difference method for solving the nonlinear Poisson-Boltzmann equation*, J. Comput Chem., 25 (2003), pp. 935–955.

[14] A. H. Boschitsch, M. O. Fenley, and Huan-Xiang Zhou, *Fast boundary element method for the linear Poisson-Boltzmann equation*, J. Phys. Chem., 106 (2002), pp. 2741–2754.

[15] W.R. Bowen and A.O. Sharif, *Adaptive finite element solution of the nonlinear Poisson-Boltzmann equation: A charged spherical particle at various distances from a charge cylindrical pore in a charged planar surface*, Journal of Colloid and Interface Science, 187 (1997), pp. 363–374.

[16] T. E. Cheatham and P. A. Kollman, *Molecular dynamics simulation of nucleic acid*, Annu. Rev. Phys. Chem., 51 (2000), pp. 435–471.

[17] W. D. Cornell, P. Cieplak, C. I. Bayly, I. R. Gould, K. M. Merz Jr, D. M. Ferguson, D. C. Spellmeyer, T. Fox, J. W. Caldwell, and P. Kollman, *A second generation force field for the simulation of proteins, nucleic acids, and organic molecules*, J. Am. Chem. Soc, 117 (1995), pp. 5179–5197.

[18] C.M. Cortis and R.A. Friesner, *An automatic three-dimensional finite element mesh generation system for the Poisson-Boltzmann equation*, J. Comput Chem., 18 (1997), pp. 1570–1590.

[19] ———, *Numerical solution of the Poisson-Boltzmann equation using tetrahedral finite element*

methods, J. Comput Chem., 18 (1997), pp. 1591–1608.
[20] T. Dolinsky, J. Nielsen, A. McCammon, and N. Baker, *Pdb2pqr: an automated pipeline for the setup of poissonboltzmann electrostatics calculations*, Nucleic Acids Research, 32 (2004), pp. 665–667.
[21] D. Eisenberg and A. D. Mclachlan, *Solvation energy in protein folding and binding*, Nature (London), 319 (1986), pp. 199–203.
[22] M. K. Gilson, M. E. Davis, B. A. Luty, and J.A. McCammon, *Computation of electrostatic forces on solvated molecules using the poisson-boltzmann equation*, J. Phys. Chem., 97 (1993), pp. 3591–600.
[23] R. B. Hermann, *Theory of hydrophobic bonding. ii. correlation of hydrocarbon solubility in water with solvent cavity surface area*, J. Phys. Chem., 76 (1972), pp. 2754–2759.
[24] M. Holst, *Multilevel Methods for the Poisson-Boltzmann Equation*, ph.d. thesis, University of Illinois at Urbana-Champaign, 1993.
[25] M. Holst, N. Baker, and F. Wang, *Adaptive multilevel finite element solution of the Poisson-Boltzmann equation I: algorithm and examples*, J. Comput Chem., 21 (2000), pp. 1319–1342.
[26] W. Im, D. Beglov, and B. Roux, *Continuum solvation model: computation of electrostatic forces from numerical solutions to Poisson-Boltzmann equation*, Computer Physics Communications, 111 (1997), pp. 59–75.
[27] S. S. Kuo, M. D. Altman, J. P. Bardhan, B. Tidor, and J. K. White, *Fast methods for simulation of biomolecule electrostatics*, IEEE/ACM international conference on Computer-aided design, (2002), pp. 466–473.
[28] B. Lee and F.M. Richards, *The interpretation of protein structures: estimation of static accessibility*, J. Mol. Biol., 55(3) (1971), pp. 379–400.
[29] M. Levitt, M. Hirshberg, R. Sharon, and V. Daggett, *Potential energy function and parameters for simulations of the molecular dynamics of proteins and nucleic acids in solution*, Comp. Phys. Comm., 91 (1995), pp. 215–231.
[30] J. Liang and S. Subramaniam, *Computation of molecular electrostatics with boundary element methods*, Biophysical Journal, 73 (1997), pp. 1830–1841.
[31] B. Z. Lu, X. Cheng, J. Huang, and J. A. McCammon, *Order n algorithm for computation of electrostatic interactions in biomolecular systems*, Proc. Natl. Acad. Sci., 103 (2006), pp. 19314–19319.
[32] B. Z. Lu, D. Q. Zhang, and J. A. McCammon, *Computation of electrostatic forces between solvated molecules determined by the Poisson-Boltzmann equation using a boundary element method*, J Chemical Physics, 122(21) (2005), p. 214102.
[33] A. V. Morozov, T. Kortemme, and D. Baker, *Evaluation of models of electrostatic interations in proteins*, J. Phys. Chem. B, 107 (2003), pp. 2075–2090.
[34] T. J. Perun and C. L. Propst, *Computer-Aided Drug Design: Methods and Applications*, Informa Healthcare, 1989.
[35] A Sayyed-Ahmad, K. Tuncay, and P.J. Ortoleva, *Efficient solution technique for solving the Poisson-Boltzmann equation*, J. Comput Chem., 25 (2004), pp. 1068–1074.
[36] K. Sharp, *Incorporating solvent and ion screening into molecular dynamics using the finite-difference poisson-boltzmann method*, J. Comput Chem., 12 (1991), pp. 454–468.
[37] K. Sharp and B Honig, *Applications of the finite defference Poisson-Boltzman method to proteins and nucleic acids*, Structure and Methods: DNA Protein Complexes and Proteins, 2 (1990), pp. 211–214.
[38] K. A. Sharp and B. Honig, *Calculating total electrostatic energies with the nonlinear poisson-boltzmann equation*, J. Phys. Chem., 94 (1990), pp. 7684–7692.
[39] T. Simonson and A. T. Bruenger, *Solvation free energies estimated from macroscopic continuum theory: An accuracy assessment*, J. Phys. Chem., 98 (1994), pp. 4683 – 4694.
[40] Y. N. Vorobjev, J. A. Grant, and H. A. Scheraga, *A combined iterative and boundary element approach for solution of the nonlinear Poisson-Boltzmann equation*, J. Am. Chem. Soc, 114 (1992), pp. 3189–3196.
[41] Y. N. Vorobjev and H. A. Scheraga, *A fast adaptive multigrid boundary element method for macromolecular electrostatic computations in a solvent*, J. Comput Chem., 18 (1996), pp. 569–583.
[42] J. Wagoner and N. A. Baker, *Solvation forces on biomolecular structures: A comparison of explicit solvent and poisson-boltzmann models*, J. Comput Chem., 25 (2004), pp. 1623–1629.
[43] L. Ying, G. Biros, and D. Zorin, *A kernel-independent adaptive fast multipole method in two and three dimensions*, Journal of Computational Physics, 196(2) (2004), pp. 591–626.
[44] R.J. Zauhar and R.S. Morgan, *A new method for computing the macromolecular electric*

potential, J. Mol. Biol., 186 (1985), pp. 815–820.
[45] Yongjie Zhang, Guoliang Xu, and Chandrajit Bajaj, *Quality meshing of implicit solvation models of biomolecular structures*, Computer Aided Geometric Design, 23 (2006), pp. 510–530.
[46] Y. Zhang, G. Xu, and C. Bajaj, *Quality meshing of implicit solvation models of biomolecular structures*, Computer Aided Geometric Design, (2006), p. in press.
[47] W. Zhao, G. Xu, and C. Bajaj, *An algebraic spline model of molecular surfaces*, Proceedings of the ACM symposium on Solid and physical modeling, (2007), pp. 297–302.
[48] H. Zhou, *Boundary element solution of macromolecular electrostatics: interaction energy between two proteins*, Biophys. J., 65 (1993), pp. 955–963.

Appendix A. The Jacobian of an Algebraic patch.

The details of the derivation of numerical quadrature technique using Algebraic spline model are in the manuscript [47]. They derived the Jacobian of an A-patch as $J(\bar{\Gamma}_j) = \sqrt{EG - F^2}$ where

$$\begin{aligned} E &= (\tfrac{\partial x_1}{\partial b_1})^2 + (\tfrac{\partial x_2}{\partial b_1})^2 + (\tfrac{\partial x_3}{\partial b_1})^2 \\ F &= (\tfrac{\partial x_1}{\partial b_1})(\tfrac{\partial x_1}{\partial b_2}) + (\tfrac{\partial x_2}{\partial b_1})(\tfrac{\partial x_2}{\partial b_2}) + (\tfrac{\partial x_3}{\partial b_1})(\tfrac{\partial x_3}{\partial b_2}) \\ G &= (\tfrac{\partial x_1}{\partial b_2})^2 + (\tfrac{\partial x_2}{\partial b_2})^2 + (\tfrac{\partial z_3}{\partial b_2})^2 \end{aligned}$$

Appendix B. Distance-dependent dielectric model. Im et al. defined $\epsilon(\vec{x})$ and $\lambda(\vec{x})$ by a volume exclusion function [26].

$$\epsilon(\vec{x}) = \epsilon_I + (\epsilon_E - \epsilon_I) H(\vec{x}; \{\vec{\mathrm{x}}_k\}),$$

and

$$\lambda(\vec{x}) = H(\vec{x}; \{\vec{\mathrm{x}}_k\}),$$

where the volume exclusion function is defined by the atomic centers $\{\vec{\mathrm{x}}_k\}_{k=1}^{n_c}$,

$$H(\vec{x}; \{\vec{\mathrm{x}}_k\}) = \prod_{k=1}^{n_c} H_k(\|\vec{x} - \vec{\mathrm{x}}_k\|),$$

and

$$H_k(r) = \begin{cases} 0, & r \le r_k - w, \\ -\frac{(r-r_k+w)^3}{4w^3} + \frac{3(r-r_k+w)^2}{4w^2}, & r_k - w < r < r_k + w, \\ 1, & r \ge r_k + w. \end{cases}$$

The gradient of $\epsilon(\vec{x})$ and $\lambda(\vec{x})$ is then derived from he derivative of this volume exclusion function.

$$H_k'(r) = -\frac{3}{4w^3}(r - r_k + w)^2 + \frac{3}{2w^2}(r - r_k + w).$$

Appendix C. Parametrization of normal derivative of electrostatic potential.

For a Gaussian quadrature point $\vec{y}_{jm}$ on the patch $\bar{\Gamma}_j$, the parametrization of electrostatic potential can be derived using the equation (4.5). Its formulation is

written as follows (C.1).

$$\begin{aligned}\phi(\vec{y}_{jm}) &= b_{m1}((1-\lambda_{jm})\phi(\vec{v}_{j1})+\lambda_{jm}\phi(\vec{v}_{j1}+\vec{n}_{j1}))\\ &+b_{m2}((1-\lambda_{jm})\phi(\vec{v}_{j2})+\lambda_{jm}\phi(\vec{v}_{j2}+\vec{n}_{j2}))\\ &+b_{m3}((1-\lambda_{jm})\phi(\vec{v}_{j3})+\lambda_{jm}\phi(\vec{v}_{j3}+\vec{n}_{j3}))\\ &= \begin{bmatrix} b_{m1}(1-\lambda_{jm})\\ b_{m2}(1-\lambda_{jm})\\ b_{m3}(1-\lambda_{jm})\\ b_{m1}\lambda_{jm}\\ b_{m2}\lambda_{jm}\\ b_{m3}\lambda_{jm}\end{bmatrix}^T \begin{bmatrix}\phi(\vec{v}_{j1})\\ \phi(\vec{v}_{j2})\\ \phi(\vec{v}_{j3})\\ \phi(\vec{v}_{j1}+\vec{n}_{j1})\\ \phi(\vec{v}_{j2}+\vec{n}_{j2})\\ \phi(\vec{v}_{j3}+\vec{n}_{j3})\end{bmatrix} = \begin{bmatrix} B_1^{jm}\\ B_2^{jm}\\ B_3^{jm}\\ B_4^{jm}\\ B_5^{jm}\\ B_6^{jm}\end{bmatrix}^T \begin{bmatrix}\phi(\vec{v}_{j1})\\ \phi(\vec{v}_{j2})\\ \phi(\vec{v}_{j3})\\ \phi(\vec{v}_{j1}+\vec{n}_{j1})\\ \phi(\vec{v}_{j2}+\vec{n}_{j2})\\ \phi(\vec{v}_{j3}+\vec{n}_{j3})\end{bmatrix}\end{aligned} \tag{C.1}$$

where $\vec{v}_{j1}$, $\vec{v}_{j2}$ and $\vec{v}_{j3}$ are the vertices of the patch $\bar{\Gamma}_j$ and $\vec{n}_{j1}$, $\vec{n}_{j2}$ and $\vec{n}_{j3}$ are their unit normal vectors.

Then, the normal derivative of electrostatic potential in Equation (4.3) can be computed using the parametric formulations (4.4) and (4.6), so that we can write it in the following formulation (C.2).

$$\frac{\partial\phi}{\partial\vec{n}}(\vec{y}_{jm}) = \begin{bmatrix} C_1^{jm}\\ C_2^{jm}\\ C_3^{jm}\\ C_4^{jm}\\ C_5^{jm}\\ C_6^{jm}\end{bmatrix}^T \begin{bmatrix}\phi(\vec{v}_{j1})\\ \phi(\vec{v}_{j2})\\ \phi(\vec{v}_{j3})\\ \phi(\vec{v}_{j1}+\vec{n}_{j1})\\ \phi(\vec{v}_{j2}+\vec{n}_{j2})\\ \phi(\vec{v}_{j3}+\vec{n}_{j3})\end{bmatrix} \tag{C.2}$$

We include the parametric representation of electrostatic potential (C.1) and the normal derivative of electrostatic potential (C.2) in the following matrix form C.3.

$$\begin{bmatrix}\phi(\vec{y}^{11})\\ \phi(\vec{y}^{12})\\ \vdots\\ \phi(\vec{y}_{jm})\\ \vdots\\ \phi(\vec{y}^{LM})\\ \frac{\partial\phi(\vec{y}^{11})}{\partial\vec{n}_{11}^{(y)}}\\ \frac{\partial\phi(\vec{y}^{12})}{\partial\vec{n}_{12}^{(y)}}\\ \vdots\\ \frac{\partial\phi(\vec{y}_{jm})}{\partial\vec{n}_{jm}^{(y)}}\\ \vdots\\ \frac{\partial\phi(\vec{y}^{LM})}{\partial\vec{n}_{LM}^{(y)}}\end{bmatrix} = \begin{bmatrix} B^{11} & 0 & \cdots & 0 & \hat{B}^{11} & 0 & \cdots & 0\\ 0 & B^{22} & \cdots & \vdots & 0 & \hat{B}^{22} & \cdots & 0\\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \cdots & B^{LM} & 0 & 0 & \cdots & \hat{B}^{LM}\\ C^{11} & 0 & \cdots & 0 & \hat{C}^{11} & 0 & \cdots & 0\\ 0 & C^{22} & \cdots & \vdots & 0 & \hat{C}^{22} & \cdots & 0\\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \cdots & C^{LM} & 0 & 0 & \cdots & \hat{C}^{LM}\end{bmatrix} \begin{bmatrix}\phi(\vec{v}_{11})\\ \phi(\vec{v}_{12})\\ \phi(\vec{v}_{13})\\ \vdots\\ \phi(\vec{v}_{L1})\\ \phi(\vec{v}_{L2})\\ \phi(\vec{v}_{L3})\\ \phi(\vec{v}_{11}+\vec{n}_{11})\\ \phi(\vec{v}_{12}+\vec{n}_{12})\\ \phi(\vec{v}_{13}+\vec{n}_{13})\\ \vdots\\ \phi(\vec{v}_{L1}+\vec{n}_{L1})\\ \phi(\vec{v}_{L2}+\vec{n}_{L2})\\ \phi(\vec{v}_{L3}+\vec{n}_{L3})\end{bmatrix} \tag{C.3}$$

where $B^{jm} = (B_1^{jm}, B_2^{jm}, B_3^{jm})$, $\hat{B}^{jm} = (B_4^{jm}, B_5^{jm}, B_6^{jm})$, $C^{jm} = (C_1^{jm}, C_2^{jm}, C_3^{jm})$ and $\hat{C}^{jm} = (C_4^{jm}, C_5^{jm}, C_6^{jm})$.