Journal of Computational Physics 241 (2013) 253–265

Contents lists available at SciVerse ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp

Parallel higher-order boundary integral electrostatics computation on molecular surfaces with curved triangulation

Weihua Geng *

Department of Mathematics, University of Alabama, Tuscaloosa, AL 35406, USA

ARTICLE INFO

Article history:
Received 30 September 2012
Received in revised form 16 January 2013
Accepted 22 January 2013
Available online 10 February 2013

Keywords:
Poisson–Boltzmann
Electrostatics
Boundary integral
Parallel computing
Message passing interface (MPI)

ABSTRACT

In this paper, we present a parallel higher-order boundary integral method to solve the linear Poisson–Boltzmann (PB) equation. In our method, a well-posed boundary integral formulation is used to ensure the fast convergence of Krylov subspace linear solver such as GMRES. The molecular surfaces are first discretized with flat triangles and then converted to curved triangles with the assistance of normal information at vertices. To maintain the desired accuracy, four-point Gauss–Radau quadratures are used on regular triangles and sixteen-point Gauss–Legendre quadratures together with regularization transformations are applied on singular triangles. To speed up our method, we take advantage of the embarrassingly parallel feature of boundary integral formulation, and parallelize the schemes with the message passing interface (MPI) implementation. Numerical tests show significantly improved accuracy and convergence of the proposed higher-order boundary integral Poisson–Boltzmann (HOBI-PB) solver compared with boundary integral PB solver using often-seen centroid collocation on flat triangles. The higher-order accuracy results achieved by present method are important to sensitive solvation analysis of biomolecules, particularly when accurate electrostatic surface potentials are critical in the molecular simulation. In addition, the higher-order boundary integral schemes presented here and their associated parallelization potentially can be applied to solving boundary integral equations in a general sense.

1. Introduction

Molecular modeling is a rising interdisciplinary approach on the study of structure, function and dynamics of molecules with biological significance [1]. Among interactions in molecular modeling, electrostatics are critical due to their ubiquitous existence. Meanwhile, electrostatics are expensive to compute as they are long-range and pairwise interactions. The Poisson–Boltzmann (PB) model is an effective approach to resolve the electrostatics including energy, potential and force of solvated biomolecules [2]. As an implicit solvent model, the PB model considers solvent effects with a mean field approximation, and models the mobile ions with the Boltzmann distribution. These implicit treatments of solvent surroundings make the PB model computationally more efficient compared with explicit solvent models, in which atomic details of solvent molecules and electrolytes are described. Recently, experimentalists also showed interests in using the PB model to provide references for newly released structures of biomolecules e.g. the neurotransmitter receptor Acetylcholine [3] and the Circadian clock complex CLOCK:BMAL1 [4].

In the PB model, as illustrated in Fig. 1(a), the computational domain $\mathbb{R}^3$ is divided into the solute (molecule) domain Ω_1 and the solvent domain Ω_2 by a closed molecular surface Γ such that $\mathbb{R}^3 = \Omega_1 \cup \Omega_2 \cup \Gamma$. As shown in Fig. 1(b), the molecular

* Tel.: +1 205 348 5302; fax: +1 205 348 7067.
E-mail address: wgeng@as.ua.edu

0021-9991/$ - see front matter © 2013 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.jcp.2013.01.029

surface Γ is formed by the traces of a spherical solvent probe rolling in contact with the van del Walls balls of the solute atoms [5,6]. The solvated molecule, which is located in domain Ω_1 with dielectric constant ε_1, is represented by a set of N_c point charges carrying Q_i charge in the units of e_c at positions $\mathbf{x}_i,\ i = 1,\ldots,N_c$. The exterior domain contains the solvent with dielectric constant ε_2, as well as mobile ions. For $\mathbf{x} = (x,y,z)$, the PB equation for the electrostatic potentials in each domain is derived from the Boltzmann distribution and Gauss' law and has the form

$$\nabla\cdot(\varepsilon_1(\mathbf{x})\nabla\phi_1(\mathbf{x})) = -\sum_{i=1}^{N_c} q_i\delta(\mathbf{x}-\mathbf{x}_i) \quad \text{in } \Omega_1, \tag{1}$$

$$\nabla\cdot(\varepsilon_2(\mathbf{x})\nabla\phi_2(\mathbf{x})) - \kappa^2\sinh\phi_2(\mathbf{x}) = 0 \quad \text{in } \Omega_2, \tag{2}$$

$$\phi_1(\mathbf{x}) = \phi_2(\mathbf{x}), \quad \varepsilon_1\frac{\partial\phi_1(\mathbf{x})}{\partial\nu} = \varepsilon_2\frac{\partial\phi_2(\mathbf{x})}{\partial\nu} \quad \text{on } \Gamma, \tag{3}$$

$$\lim_{|\mathbf{x}|\to\infty}\phi_2(\mathbf{x}) = 0, \tag{4}$$

where ϕ_1 and ϕ_2 are the electrostatic potentials in each domain, $q_i = e_cQ_i/k_BT,\ i = 1,\ldots,N_c$, e_c the electron charge, k_B the Boltzmann's constant, T the absolute temperature, δ the Dirac delta function, κ the Debye–Hückel parameter, and ν the unit outward normal on the interface Γ. We assume weak ionic strength in this context therefore the non-linear sinh function term can be approximated by its linearized term, resulting in the linear PB equation with $\sinh\phi_2(\mathbf{x})$ term replaced by $\phi_2(\mathbf{x})$ in Eq. (2).

The linear PB model is an elliptic equation defined on multiple domains with discontinuous coefficients across the domain interfaces. The PB equation has an analytical solution only for the simple geometries such as spheres [7] or rods [8]. For molecules with complex geometries, the PB equation can only be solved numerically, which is challenging due to the following numerical difficulties.

(1) The solutions to the PB equation, physically the electrostatic potentials, are not smooth across the interface as the continuities of both the potentials and the flux densities in Eq. (3) across the interface Γ are required to be satisfied.
(2) The complex geometry of the interface needs to be captured to maintain the accuracy of the potentials particularly on or near the interface.
(3) The partial charges carried by the individual atoms of the solute, modeled by the weighted summation of the Dirac delta functions, is hard to accurately discretize.
(4) The PB equation is defined on the entire $\mathbb{R}^3$ domain subject to boundary condition that the potentials approach zero at infinity thus a cutoff for 3D mesh-based methods is inevitable.

The wide application of the PB model as well as its associated numerical difficulties attracted attention from various computational science communities ranging from biophysics, biochemistry, mathematics, computer science, mechanical engineering as well as electrical engineering. Many numerical PB solvers were developed and they can be roughly but not completely divided into two categories: The 3D mesh-based finite difference/finite element methods [9–14]; and the boundary integral methods [15–24]. All these methods have their own advantages and disadvantages. For example, the PB solvers embedded in molecular modeling packages such as Dephi [9], CHARMM [10], AMBER [11], APBS [12] use standard seven-point finite difference with approximated approaches to bypass the numerical difficulties (1)–(4). Although arguably these

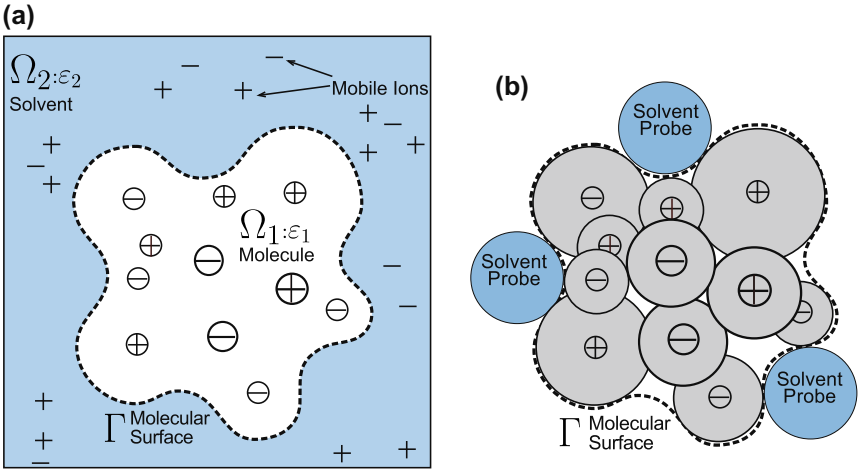

Fig. 1. (a) Poisson–Boltzmann (PB) model: domains Ω_1(molecule) and Ω_2(solvent) with different dielectric constants ε_1 and ε_2 respectively; (b) the molecular surface is formed by the trace of solvent probe in contact with the solute (molecule).

solvers have reduced accuracy, the efficient, robust and user-friendly features of these PB solvers brought their popularities among the bio-oriented community. An often overlooked drawback of these solvers is that they provide acceptable accuracy of resolved electrostatics potentials away from the interface but are unable to provide accurate solutions *near* or *on* the interfaces. Applications such as molecular simulation related to ion channels [25], cell membranes [26], and chromatin packing[27] require accurate electrostatic potentials and fields near or on the interface, thus call for developing higher-order methods to solve the PB equation. Three-dimensional mesh-based Interface methods such as Immersed Interface Methods (IIM) [13] and Matched Interface and Boundary Poisson–Boltzmann (MIBPB) solver [14] can significantly improve the accuracy by rigorously treating the numerical difficulties (1)–(3). However, these methods need to cope with numerical difficulty (4) and the complexities of the algorithms often reduce the efficiency.

Compared with 3D mesh-based methods, the boundary integral methods have many advantages.

(1) The solution is characterized solely in terms of surface distributions so there are fewer unknowns in comparison to methods that discretize the entire domain.
(2) The far-field boundary condition in Eq. (4) is exactly imposed.
(3) The surface geometry of Γ can be represented to high precision using appropriate boundary elements.
(4) The electrostatic potential at charge sites are accurately determined using exact analytical expressions.
(5) The continuity conditions in Eq. (3) are explicitly enforced.

Due to these advantages, boundary integral PB solvers have gained increased attention and we briefly review studies relating to the present work. In 1988, Zauhar et al. [28] introduced the boundary integral formulation by solving the Poisson equation for the induced surface charges. In 1990, Yoon and Lenhoff [29] formulated an ill-posed integral formulation of the PB equation. In 1991, Juffer et al. [15] reformulated the previous work to obtain a well-posed formulation, which were applied by most of the boundary integral PB solvers after that. The boundary integral methods can analytically circumvent the numerical difficulties (1)–(4), and accelerate the solver with fast algorithms such as fast multipole method (FMM) [16,17,19] and treecode [20]. These boundary integral PB solvers mostly applied centroid collocation methods on flat triangle and benchmark tests on spherical cavities with available analytical solutions show 0.5th order accuracy relative to number of elements [17,20], which left spaces for the more challenging problem of developing higher-order boundary integral PB solver.

In this paper, we present a more accurate boundary integral PB solver on curved triangles with higher-order quadratures and regularization of singularities. The rest of the paper is organized as follows. In Section 2, we provide our algorithms including the well-posed boundary integral formulation and the higher-order numerical schemes, followed by the MPI parallelization. In Section 3, we provide the numerical results, first on the spherical cavities with centered and eccentric partial changes whose analytical solutions are available and then on a protein (PDB: 1ajj) for the electrostatics solvation energy computation and the parallel efficiency. This paper ends with a section of concluding remarks.

2. Methods

We will use the well-posed boundary integral formulation from Juffer's work [15] together with high order quadrature [28] and singularities regularization by a transformation [30,31]. We modify and improve these methods as needed and we will explain the details in this section. One fact affecting the accuracy of boundary integral methods is the discretization of the surface. For the sphere, we use a non-uniformed triangular surface from MSMS [32] with radial projection to correct the truncated output, or a uniform icosahedral triangulation [33]. For biomolecules, we only use MSMS to generate the flat triangles. All flat triangles are then converted to curved triangles by applying the schemes introduced as follows. Through the paper, we call our higher-order boundary integral Poisson–Boltzmann solver as HOBI-PB solver.

2.1. Well-posed integral formulation

The differential PB equation in Eqs. (1) and (2) can be converted to boundary integral equation. By applying the fundamental solution of Poisson Eq. (1), G_0, in Ω_1 and the fundamental solution of PB Eq. (2), G_κ, in Ω_2, together with Green's second theorem, and cancel the normal derivative terms with interface jump conditions in Eq. (3), the coupled integral equations can be derived as [29]:

$$\phi_1(\mathbf{x}) = \int_\Gamma \left[G_0(\mathbf{x},\mathbf{y}) \frac{\partial \phi_1(\mathbf{y})}{\partial \nu_\mathbf{y}} - \frac{\partial G_0(\mathbf{x},\mathbf{y})}{\partial \nu_\mathbf{y}} \phi_1(\mathbf{y}) \right] dS_\mathbf{y} + \sum_{k=1}^{N_c} q_k G_0(\mathbf{x},\mathbf{y}_k), \quad \mathbf{x} \in \Omega_1, \tag{5}$$

$$\phi_2(\mathbf{x}) = \int_\Gamma \left[-G_\kappa(\mathbf{x},\mathbf{y}) \frac{\partial \phi_2(\mathbf{y})}{\partial \nu_\mathbf{y}} + \frac{\partial G_\kappa(\mathbf{x},\mathbf{y})}{\partial \nu_\mathbf{y}} \phi_2(\mathbf{y}) \right] dS_\mathbf{y}, \quad \mathbf{x} \in \Omega_2, \tag{6}$$

where $G_0(\mathbf{x},\mathbf{y})$ and $G_\kappa(\mathbf{x},\mathbf{y})$ are the Coulomb and screened Coulomb potentials,

$$G_0(\mathbf{x},\mathbf{y}) = \frac{1}{4\pi|\mathbf{x}-\mathbf{y}|}, \quad G_\kappa(\mathbf{x},\mathbf{y}) = \frac{e^{-\kappa|\mathbf{x}-\mathbf{y}|}}{4\pi|\mathbf{x}-\mathbf{y}|}. \tag{7}$$

However, straightforward discretization of Eqs. (5) and (6) yields a linear system which becomes ill-conditioned as the number of boundary elements increases [34]. Juffer et al. derived a well-posed boundary integral formulation by going through the differentiation of the single-layer and double-layer potentials [15]. The desired forms are:

$$\frac{1}{2}(1+\varepsilon)\phi_1(\mathbf{x}) = \int_\Gamma \left[K_1(\mathbf{x},\mathbf{y})\frac{\partial\phi_1(\mathbf{y})}{\partial v_\mathbf{y}} + K_2(\mathbf{x},\mathbf{y})\phi_1(\mathbf{y})\right] dS_\mathbf{y} + S_1(\mathbf{x}), \quad \mathbf{x}\in\Gamma, \tag{8}$$

$$\frac{1}{2}\left(1+\frac{1}{\varepsilon}\right)\frac{\partial\phi_1(\mathbf{x})}{\partial v_\mathbf{x}} = \int_\Gamma \left[K_3(\mathbf{x},\mathbf{y})\frac{\partial\phi_1(\mathbf{y})}{\partial v_\mathbf{y}} + K_4(\mathbf{x},\mathbf{y})\phi_1(\mathbf{y})\right] dS_\mathbf{y} + S_2(\mathbf{x}), \quad \mathbf{x}\in\Gamma, \tag{9}$$

with the notation

$$K_1(\mathbf{x},\mathbf{y}) = G_0(\mathbf{x},\mathbf{y}) - G_\kappa(\mathbf{x},\mathbf{y}), \quad K_2(\mathbf{x},\mathbf{y}) = \varepsilon\frac{\partial G_\kappa(\mathbf{x},\mathbf{y})}{\partial v_\mathbf{y}} - \frac{\partial G_0(\mathbf{x},\mathbf{y})}{\partial v_\mathbf{y}},$$

$$K_3(\mathbf{x},\mathbf{y}) = \frac{\partial G_0(\mathbf{x},\mathbf{y})}{\partial v_\mathbf{x}} - \frac{1}{\varepsilon}\frac{\partial G_\kappa(\mathbf{x},\mathbf{y})}{\partial v_\mathbf{x}}, \quad K_4(\mathbf{x},\mathbf{y}) = \frac{\partial^2 G_\kappa(\mathbf{x},\mathbf{y})}{\partial v_\mathbf{x}\partial v_\mathbf{y}} - \frac{\partial^2 G_0(\mathbf{x},\mathbf{y})}{\partial v_\mathbf{x}\partial v_\mathbf{y}}, \tag{10}$$

$$S_1(\mathbf{x}) = \sum_{k=1}^{N_c} q_k G_0(\mathbf{x},\mathbf{y}_k), \quad S_2(\mathbf{x}) = \sum_{k=1}^{N_c} q_k \frac{\partial G_0(\mathbf{x},\mathbf{y}_k)}{\partial v_\mathbf{x}} \tag{11}$$

and $\varepsilon = \varepsilon_1/\varepsilon_2$. Note this is the well-posed Fredholm second kind of integral equation which is also our choice in this paper.

In order to numerically solve the coupled Eqs. (8) and (9), we need to discretize the molecular surface Γ with high quality elements and implement the numerical integral with higher-order quadrature. We also need to treat the occurred singularities or near-singularities when $\mathbf{x}$ and $\mathbf{y}$ are equal or nearly equal in Kernels $K_{1,\ldots,4}$. These details are described in the following subsections.

2.2. Curved triangles and higher-order quadratures

Given the positions and radii of all atoms of a molecule, a triangulation program e.g. MSMS [32] can generate a discretized surface with a set of N_f flat triangles, N_v nodes (vertices) and corresponding normal directions. To achieve higher-order accuracy, we apply the schemes described in [28] to convert these flat triangles to curved triangles under the new background of solving the well-posed integral PB equation in Eqs. (8) and (9). We also modify and improve these schemes to treat the singularities of kernels $K_{1,\ldots,4}$ in Eq. (10). To keep this paper in an integrated form, we start from restating the schemes in [28] to produce the curved triangles and higher-order quadratures for discretizing Eqs. (8) and (9).

We first replace the straight element edges as shown in Fig. 2(a) with curved arcs in Fig. 2(b) [28]. Let $\mathbf{x}(t)$ be the arc between two nodes, parameterized by the dimensionless variable t,

$$\mathbf{x}(t) = \mathbf{c}_0 + \mathbf{c}_1 t + \mathbf{c}_2 t^2 + \mathbf{c}_3 t^3, \tag{12}$$

where $\mathbf{c}_0, \mathbf{c}_1, \mathbf{c}_2, \mathbf{c}_3$ are vector constants (12 unknowns), which will be determined by a pair of connected nodes and associated unit normals (12 conditions). At any point on the curve, the normal can be found by $\mathbf{n}(t) = \mathrm{sgn}(t)\frac{\kappa(t)}{|(\kappa(t))|}$, where $\mathrm{sgn}(t) = \pm 1$ is chosen to keep a constant orientation of $\mathbf{n}(t)$ along the curve and $\kappa(t)$ is the curvature given as [28]:

$$\kappa(t) = \frac{1}{d\mathbf{x}/dt}\left[\frac{d^2\mathbf{x}}{dt^2} - \frac{(\frac{d\mathbf{x}}{dt}\cdot\frac{d^2\mathbf{x}}{dt^2})}{|d\mathbf{x}/dt|^2}\frac{d\mathbf{x}}{dt}\right]. \tag{13}$$

As shown in Fig. 2(c), we can use parameter $u \in [0,1]$ for the two curves starting from X_1 and ending at X_2 and X_3. Then for any given u, two points on the curves X_1X_2 and X_1X_3 are specified, say Y_1 and Y_2 with normal directions. By following the same procedure, we could find a curve connecting Y_1 and Y_2, using $v \in [0,1]$ as the parameter. The pair of parameters (u, v) will therefore correspond to a point on the curve element.

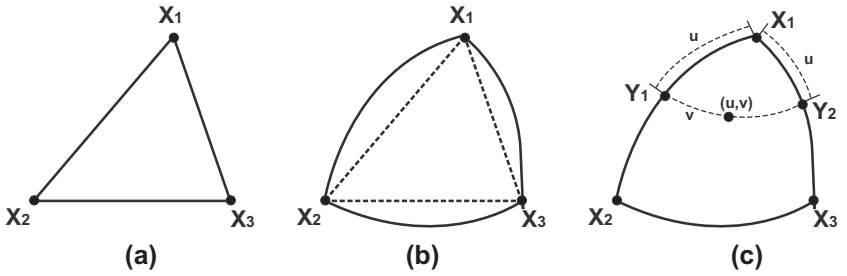

Fig. 2. Triangular surface elements: (a) a flat triangle (b) a curved triangle (c) curve linear coordinates (u, v) in the curved triangle.

In order to conveniently use quadrature rules, the integral is conducted in a unit right triangle. A mapping between the parameters $r, s \in [0,1]$ on the unit right triangle in Fig. 3(a) and parameters u, v on a unit square in Fig. 3(b) is constructed with the following transformation [28].

$$\begin{cases} u = (r+s), \quad v = s/(r+s), & \text{if } r+s \neq 0, \\ u = 0, \quad v = 0, & \text{if } r = s = 0, \end{cases} \tag{14}$$

Based on this, we can establish the mapping from point (r,s) on the unit right triangle in Fig. 3(a) to point $\mathbf{x}^{(j)}(u(r,s), v(r,s)) = \mathbf{x}^{(j)}(r,s)$ on the jth curved elements in Fig. 3(c) by the following steps:

(1) Given (r,s), find u, v through Eq. (14).
(2) Plug u as the parameter into the curve functions of X_1X_2 and X_1X_3 to locate Y_1 and Y_2 as shown in Fig. 2(c).
(3) Find the curve function of Y_1Y_2 and then plug v as the parameter to finally get $\mathbf{x}^{(j)}$ on the curved elements.

However, this is not an analytical function for efficient computations but a constructive procedure. As a remedy, a high-order 10-point interpolation scheme is used [28]. The brief idea is illustrated in Fig. (3) on the jth element of the triangulation.

(1) Pick 10 specified points (r_k, s_k) first in the right unit triangle for $k = 1, 2, \ldots, 10$.
(2) Find $u(r_k, s_k)$, $v(r_k, s_k)$ according to Eq. (14).
(3) Find $\mathbf{x}^j(u(r_k, s_k), v(r_k, s_k))$ and their normal directions on the curved element, which is parameterized by (u, v). Note points $1, 4, 9$ are already given with the flat triangles. With these three nodes, three trajectories can be found and used to find the positions and normal directions of points $2, 3, 5, 6, 7, 8$. Finally with points 5 and 6, the dashed trajectory can be formed to find point 10 and its normal direction.
(4) As required by the quadrature rules, interpolate any point on the curved element $\mathbf{x}^{(j)}(r,s)$ by the expression

$$\mathbf{x}^{(j)}(r,s) = \sum_{k=1}^{10} N_k(r,s)\mathbf{x}_k^{(j)}, \tag{15}$$

where $N_k(r,s)$ are Lagrangian interpolation polynomial for a 10-point element. See Table 1 of [28] for the expression of $N_k(r,s)$. Note Eq. (15) can also be used to find the partial derivative of $\mathbf{x}^{(j)}$ with respect to r and s, which is required for computing the Jacobian for transformation and the normal direction at $\mathbf{x}^{(j)}$.

Suppose now we will integrate function $f(\mathbf{x})$ on the jth curved element. The quadrature rules give the position of a set of points on the unit right triangle e.g. (r_m, s_m) and quadrature weights W_m for $m = 1, 2, \ldots, N$, where N is the number of quadrature points, and the integral can be evaluated as:

$$\int_{\triangle_j} f(\mathbf{x})dA = \sum_{m=1}^{N} f(\mathbf{x}(r_m, s_m)) \left| \frac{\partial \mathbf{x}(r_m, s_m)}{\partial r} \times \frac{\partial \mathbf{x}(r_m, s_m)}{\partial s} \right| W_m \tag{16}$$

Note the term $\frac{\partial \mathbf{x}(r_m,s_m)}{\partial r} \times \frac{\partial \mathbf{x}(r_m,s_m)}{\partial s}$ gives the normal direction of the point $\mathbf{x}^{(j)}(r,s)$ with parameters (r,s), and we will use this information to supply the required normal direction in Eqs. (8) and (9).

In this paper, we use $N = N_{GR}$ points (practically we choose $N_{GR} = 4$) in the Gauss–Radau quadrature [35]. For $i = 1, 2, \ldots, N_v$, the ith and the $(i + N_v)$th element of the discretized matrix–vector product **Au** are given as

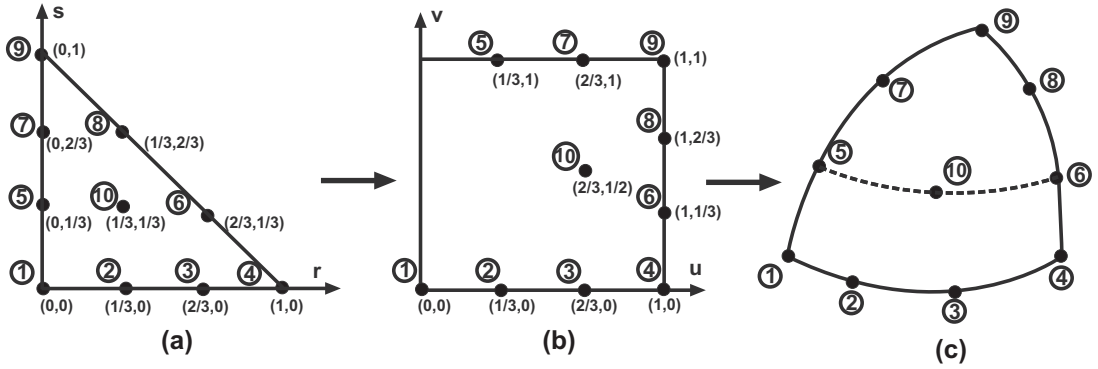

Fig. 3. Coordinates transformation.

$$\{\mathbf{Au}\}_i^r = \frac{1}{2}(1+\varepsilon)\phi_1(\mathbf{x}_i) - \sum_{j=1}^{N_i^r}\sum_{m=1}^{N_{GR}}\sum_{n=1}^{3} W_{j,m,n}\left[K_1(\mathbf{x}_i,\mathbf{y}_{j,m})\frac{\partial\phi_1(\mathbf{x}_{j,n})}{\partial v_{\mathbf{x}_{j,n}}} + K_2(\mathbf{x}_i,\mathbf{y}_{j,m})\phi_1(\mathbf{x}_{j,n})\right], \tag{17}$$

$$\{\mathbf{Au}\}_{i+N_v}^r = \frac{1}{2}\left(1+\frac{1}{\varepsilon}\right)\frac{\partial\phi_1(\mathbf{x}_i)}{\partial v_{\mathbf{x}_i}} - \sum_{j=1}^{N_i^r}\sum_{m=1}^{N_{GR}}\sum_{n=1}^{3} W_{j,m,n}\left[K_3(\mathbf{x}_i,\mathbf{y}_{j,m})\frac{\partial\phi_1(\mathbf{x}_{j,n})}{\partial v_{\mathbf{x}_{j,n}}} + K_4(\mathbf{x}_i,\mathbf{y}_{j,m})\phi_1(\mathbf{x}_{j,n})\right]. \tag{18}$$

In Eqs. (17) and (18), the superscript r of $\{\mathbf{Au}\}$ stands for regular triangle, $W_{j,m,n}$ contains the weights and coefficients associated with the quadrature, the transformation Jacobian, and the interpolation coefficients in Eq. (15). Note that N_i^r is the number of regular triangles associated with the ith vertex. In these equations, $\mathbf{x}_{j,n}$ are the same set of nodes as $\mathbf{x}_i$, but $\mathbf{y}_{j,m}$ are the quadrature points on the jth curved element, which are mapped from the predetermined points on the unit right triangle. This mismatch brings difficulty to apply Fast Multipole Method or treecode to accelerate the higher-order scheme. Studies about this issue will be proceeded in our future work.

2.3. Treatment of singularities

From Eqs. (17) and (18), we can see that when node i is one of the vertices of the jth curved triangular element, singularity or near singularity occurs at evaluating kernels $K_{1,\dots,4}$. In other words, $\mathbf{x}_i$ is equal to or nearly equal to $\mathbf{y}_{j,m}$. It can be shown that the singularities in kernels $K_{1,\dots,4}$ are in the order of $\mathcal{O}(\frac{1}{|\mathbf{x}_i-\mathbf{y}_{j,m}|})$ [15]. To treat these singularities, we use the tensor-product of Gauss–Legendre quadrature on a unit square, together with a transformation [30,31]. In this case the mapping from a unit square ($0 \leqslant x, y \leqslant 1$) to a unit right triangle ($0 \leqslant r, s \leqslant 1$ and $r+s \leqslant 1$), and to a curved triangle is constructed. The mapping that $r = (1-y)x$ and $s = yx$ for $0 \leqslant x, y \leqslant 0$ is used to remove the singularities in kernels $K_{1,\dots,4}$ when they appear at $(r,s) = (0,0)$ which indicates $x = 0$. To understand this, it can be seen that the Jacobian for the transformation from (r,s) to (x,y) is x, thus it can remove the $\mathcal{O}(\frac{1}{|\mathbf{x}_i-\mathbf{y}_{j,m}|})$ type of singularities.

In the treatment of singularities, if the number of quadrature points used in each direction of the Gauass-Lagendre quadrature is N_{GL}, the ith and $(i+N_v)$th element of the discretized matrix–vector product $\mathbf{Au}$ will in addition contain the following singular component

$$\{\mathbf{Au}\}_i^s = \frac{1}{2}(1+\varepsilon)\phi_1(\mathbf{x}_i) - \sum_{j=1}^{N_i^s}\sum_{m=1}^{(N_{GL})^2}\sum_{n=1}^{3} W_{j,m,n}\left[K_1(\mathbf{x}_i,\mathbf{y}_{j,m})\frac{\partial\phi_1(\mathbf{x}_{j,n})}{\partial v_{\mathbf{x}_{j,n}}} + K_2(\mathbf{x}_i,\mathbf{y}_{j,m})\phi_1(\mathbf{x}_{j,n})\right], \tag{19}$$

$$\{\mathbf{Au}\}_{i+N_v}^s = \frac{1}{2}\left(1+\frac{1}{\varepsilon}\right)\frac{\partial\phi_1(\mathbf{x}_i)}{\partial v_{\mathbf{x}_i}} - \sum_{j=1}^{N_i^s}\sum_{m=1}^{(N_{GL})^2}\sum_{n=1}^{3} W_{j,m,n}\left[K_3(\mathbf{x}_i,\mathbf{y}_{j,m})\frac{\partial\phi_1(\mathbf{x}_{j,n})}{\partial v_{\mathbf{x}_{j,n}}} + K_4(\mathbf{x}_i,\mathbf{y}_{j,m})\phi_1(\mathbf{x}_{j,n})\right]. \tag{20}$$

In Eqs. (19) and (20), the superscript s of $\{\mathbf{Au}\}$ stands for singular triangle, and $W_{j,m,n}$ contains the weights and coefficients associated with the quadrature, the transformation Jacobians (additionally contains the Jacobian of the $r = (1-y)x$ and $s = yx$ mapping), and the interpolation coefficients in Eq. (15). Note index j has the range up to N_i^s, which stands for the number of singular triangles associated with the ith vertex. Simulation shows this value is various for different vertices and it can be as big as 15. In this paper, we practically choose $N_{GL} = 4$ points in each direction to ensure desired accuracy [35].

2.4. Low-order scheme

We also briefly introduce the low-order boundary integral Poisson–Boltzmann (LOBI-PB) solver, which is used for comparison in this paper. In this low-order scheme, the flat triangle and centroid collocation are used, i.e. the quadrature point is located at the center of each triangle. This scheme also assumes that the potential and its normal derivative, as well as the kernel functions are uniform on each triangle. When singularity in kernels occurs, the contribution of this triangle in the integral is then simply removed. This scheme is in fact widely used in latest boundary integral methods in solving PB equations to provide convenience on incorporating fast algorithms such as FMM [34] and treecode [20]. For $i = 1, 2, \dots, N_f$, the ith and the $(i+N_f)$th element of the discretized matrix–vector product $\mathbf{Au}$ are given as

$$\{\mathbf{Au}\}_i = \frac{1}{2}(1+\varepsilon)\phi_1(\mathbf{x}_i) - \sum_{j=1, j\neq i}^{N_f} W_j\left[K_1(\mathbf{x}_i,\mathbf{x}_j)\frac{\partial\phi_1(\mathbf{x}_j)}{\partial v_{\mathbf{x}_j}} + K_2(\mathbf{x}_i,\mathbf{x}_j)\phi_1(\mathbf{x}_j)\right], \tag{21}$$

$$\{\mathbf{Au}\}_{i+N_f} = \frac{1}{2}\left(1+\frac{1}{\varepsilon}\right)\frac{\partial\phi_1(\mathbf{x}_i)}{\partial v_{\mathbf{x}_i}} - \sum_{j=1, j\neq i}^{N_f} W_j\left[K_3(\mathbf{x}_i,\mathbf{x}_j)\frac{\partial\phi_1(\mathbf{x}_j)}{\partial v_{\mathbf{x}_j}} + K_4(\mathbf{x}_i,\mathbf{x}_j)\phi_1(\mathbf{x}_j)\right]. \tag{22}$$

It worths mentioning that the unknowns of LOBI-PB is at the centroid of the triangle in the number of N_f while the unknowns of HOBI-PB is at the vertices of the triangle in the number of N_v.

Table 1
Pseudocode for parallel HOBI-PB solver using replicated data algorithm.

1	On main processor
2	Read protein data
3	Call MSMS to generate triangulation
4	Copy protein data and triangulation to all other processors
5	On each processor
6	Locally compute sources terms for each assigned vertex
7	Locally convert flat triangles to curved triangles
8	For each assigned triangle, locally compute and store quadrature information
9	For each assigned vertex, locally store quadrature of associated singular triangles
10	Copy result to all other processors
11	Set initial guess for GMRES iteration
12	Compute assigned segment of matrix–vector
13	Copy result to all other processors
14	On main processor
15	Test for GMRES convergence
16	If no, go to step 12 for next iteration
17	If yes, go to step 18
18	On each processor
19	Compute assigned segment of electrostatic solvation energy
20	Copy result to main processor
21	On main processor
22	Add segments of electrostatic solvation energy and output result

2.5. Electrostatic solvation energy formulation

The electrostatic solvation energy is computed by

$$E_{\mathrm{sol}} = \frac{1}{2}\sum_{k=1}^{N_c} q_k \phi_{\mathrm{reac}}(\mathbf{x}_k) = \frac{1}{2}\sum_{k=1}^{N_c} q_k \int_\Gamma \left[K_1(\mathbf{x}_k, \mathbf{y}) \frac{\partial \phi_1(\mathbf{y})}{\partial \nu_{\mathbf{y}}} + K_2(\mathbf{x}_k, \mathbf{y}) \phi_1(\mathbf{y}) \right] dS_{\mathbf{y}}, \tag{23}$$

where $\phi_{\mathrm{reac}}(\mathbf{x}_k) = \phi_1(\mathbf{x}_k) - S_1(\mathbf{x}_k)$, whose formulation is the integral part of Eq. (23), is the reaction potential at the kth solute atom. The electrostatic solvation energy, which can be regarded as the atomistic charge weighted average of the reaction potential ϕ_{reac}, can effectively characterize the accuracy of a PB solver.

2.6. MPI based parallel implementation

The HIBO-PB solver uses the boundary integral formulation, which can be conveniently parallelized. The majority of the CPU time is taken by the following routines.

(1) Compute the source term in Eqs. (8) and (9).
(2) Convert flat triangles to curved triangles.
(3) Compute and store the Gauss–Radau quadrature information for each curved triangle.
(4) Compute and store the Gauss–Legendre quadrature information for singular triangles associated with each vertex.
(5) Perform matrix–vector product as in Eqs. (17)–(20) on each GMRES iteration.
(6) Compute electrostatic solvation energy with Gauss–Radau quadrature.

Among all these routines, routine (5) is the most expensive one and it will be repeatedly computed on each GMRES iteration. We parallelize all these routines to maximize the parallel efficiency. Table 1 provides the Pseudocode for MPI based parallel implementation.

3. Results

In this section, we present numerical results. We first solve the PB equation on spherical cavities with a centered charge and with an eccentric change at different locations. The analytical solutions in terms of a closed form (for centered charge) and in terms of spherical harmonics (for multiple eccentric charges) are available for these tests [7]. To demonstrate the higher accuracy obtained by higher-order boundary integral Poission–Boltzmann (HOBI-PB) solver, we compare the numerical results with APBS [12], MIBPB[14], and the lower-order boundary integral Poisson–Boltzmann (LOBI-PB) solver. APBS uses straight-forward finite difference scheme. MIBPB is a 2nd order interface method repeatedly using local interpolation to capture interface jump conditions [36,37] and applying a Dirichlet-to-Neumann mapping to transform the singular charges to interface jump conditions [38]. LOBI-PB discretizes the integral equations with flat triangles and performs the numerical integral with centroid collocation as explained in the previous section.

We then solve the PB equation on a test protein (PDB: 1ajj), which is a lipprotein receptor with 37 residues and 519 atoms. We report the electrostatic solvation energy results for this protein computed from both HOBI-PB and LOBI-PB to demonstrate the improved accuracy achieved by the higher-order integral schemes. All algorithms are written in Fortran 90/95 and compiled with GNU Fortran with flag -O3. The serial simulations are performed with a single CPU (Intel (R) Xeon (R) CPU E5440 @ 2.83 GHz with 2 G Memory) on an 8-core workstation. MPI parallel simulations are conducted on the DMC cluster (Intel (R) Xeon (R) CPU E5520 @ 2.27 GHz with 1.5 G memory for each core) at Alabama Supercomputing Center. Before seeing the numerical results, we define order and errors.

3.1. Order and errors

In this paper, we report the relative errors defined as

$$e_\phi = \frac{\max\limits_{i=1,\dots,N} |\phi^{num}(x_i) - \phi^{exa}(x_i)|}{\max\limits_{i=1,\dots,N} |\phi^{exa}(x_i)|}, \tag{24}$$

where N is the number of unknowns. Note N is the number of vertices for HOBI-PB, the number of triangular elements for LOBI-PB, and the number of close-to-surface mesh points (irregular points) for APBS and MIBPB. The notation ϕ^{num} represents numerically solved surface potentials and ϕ^{exa} denotes the analytical solutions obtained by Kirkwood's spherical harmonic expansion [7]. The discretization of APBS and MIBPB are on the Cartesian grid with mesh size h. The discretization of HOBI-PB and LOBI-PB are on the molecular surface with density d, number of vertices per Å^2.

The numerical order of accuracy is computed with

$$\texttt{order} = \log_{\frac{\texttt{coarse\_mesh}}{\texttt{fine\_mesh}}} \frac{\texttt{coarse\_error}}{\texttt{fine\_error}} \tag{25}$$

following the convention of numerical analysis, where "mesh" refers to h for finite difference methods or density d for boundary integral methods, both at coarse and fine levels.

3.2. On a spherical cavity with one centered charge

We first solve the linear Poisson–Boltzmann equation on a spherical cavity with radius 2 Å and a centered change $1e_c$ submerged in water with zero ionic strength. We report the electrostatic solvation energy E_{sol} and surface potential errors e_ϕ computed with all above-mentioned methods in Table 2. The results of APBS and MIBPB are from reference [38].

From Table 2, we can see that APBS provides acceptable value in electrostatic solvation energy compared with the true value 81.98 kcal/mol since only the potential at the center of the spherical cavity is required to compute the electrostatic solvation energy. However, the surface potentials computed by APBS method show large errors. This is due to the approximation on interface conditions and singular charges of standard finite difference methods.

MIBPB uses a more sophisticated finite difference scheme. The rigorous treatment on interface conditions and singular charges significantly improves accuracy. The electrostatic solvation energy is nearly perfect even at coarse grid and the surface potential is very accurate with solid 2nd order convergence pattern relative to mesh-size h.

LOBI-PB gives the electrostatic solvation energy in the same accurate level as APBS but produces much more accurate surface potentials than APBS does. These surface potentials converge at the 0.5th order relative to density d.

HOBI-PB is obviously a more accurate method. It shows 1.5th order of convergence relative to density d as reflected from surface potential errors and its accuracy is even better than results from MIBPB.

These results demonstrate HOBI-PB is the most accurate PB solver among its peers. According to our knowledge, for this classic benchmark test, there is no other PB solvers can achieve the same level of accuracy as HOBI-PB does.

3.3. A spherical cavity with one eccentric charge

We further investigate the performance of HOBI-PB and LOBI-PB on a spherical cavity with radius 1 Å and a $1e_c$ eccentric charge at different locations. The errors shown are in terms of surface potential e_u as previously defined. The Debye–Hückel

Table 2
Accuracy comparison of different PB solvers on a spherical cavity (radius = 2 Å, $q(0,0,0) = 1e_c,\ \varepsilon = 80,\ \kappa = 0$); h the mesh size; d the number of vertices per Å^2.

h	APBS			MIBPB			d	HOBI-PB			LOBI-PB		
	E_{sol}	e_ϕ	ord.	E_{sol}	e_ϕ	ord.		E_{sol}	e_ϕ	ord.	E_{sol}	e_ϕ	ord.
1	-83.44	1.94e + 0		-81.95	1.24e−2		5	-81.98	1.45e−4		-83.41	4.07e−4	
0.5	-85.85	1.31e + 0	0.6	-81.98	1.91e−3	2.7	10	-81.98	5.26e−5	1.5	-83.13	2.55e−4	0.7
0.2	-82.58	5.76e−1	0.9	-81.98	3.87e−4	1.7	20	-81.98	1.52e−5	1.8	-82.82	1.82e−4	0.5
0.1	-82.27	2.94e−1	1.0	-81.98	1.07e−4	1.9	40	-81.98	6.13e−6	1.3	-82.60	1.27e−4	0.5
0.05	-82.03	1.49e−1	1.0	-81.98	2.31e−5	2.2	80	-81.98	1.85e−6	1.7	-82.43	8.58e−5	0.6

parameter κ is set to 1. The charge moves from the center of the sphere toward to the surface and we test the performance of the PB solvers in response to the change of locations. The closer the charge is to the surface, the more variant the induced charges on the surface appear to be. This interesting phenomenon draws attention from many researchers, e.g. the peak separation method by Juffer et al. [15] and the image method by Deng et al. [39]. In addition to that, we also try to investigate if the quality of the triangulation will affect the accuracy. To this end, we report the results computed on the triangular surfaces generated by MSMS [32] and by a uniform icosahedral triangulation routine (ico) [33]. MSMS generates triangles in various shapes and sizes, while ico generates uniform and equilateral triangles with fixed numbers of triangles such as $20, 80, 320$, etc. The results are plotted in Fig. 4, and we have the following observation and discussion.

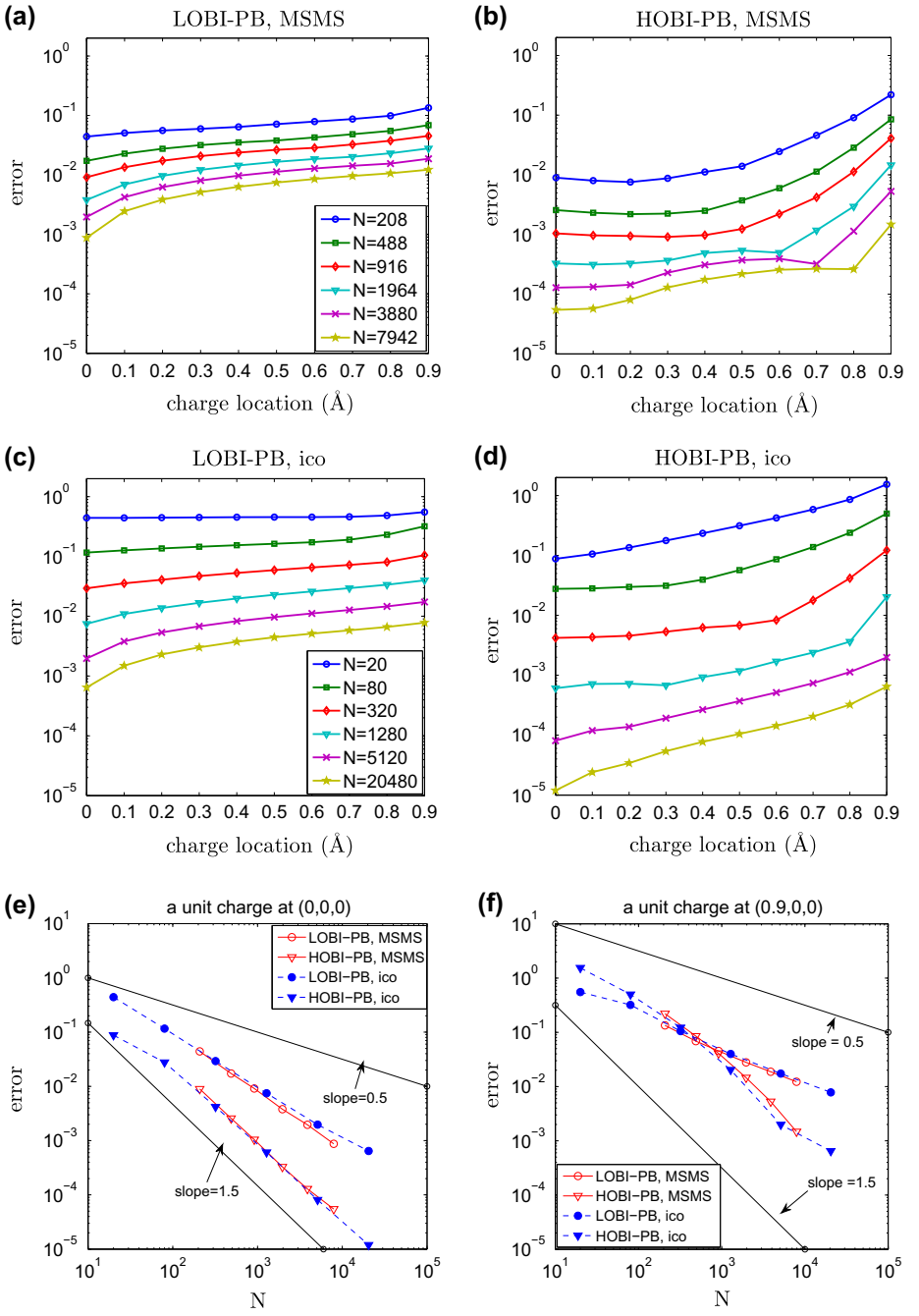

Fig. 4. relative surface potential errors (e_ϕ) on a spherical cavity with radius 1 Å and eccentric unit charge, $\kappa = 1$, and $\varepsilon = 80$: (a) LOBI-PB, MSMS; (b) HOBI-PB, MSMS; (c) LOBI-PB, ico; (d) HOBI-PB, ico; (e) error vs. elements # N, a charge located at (0,0,0); (f) error vs. elements # N, a charge located at (0.9,0,0).

(1) The results of HOBI-PB in Fig. 4(b)–(d) show significant improvement in accuracy compared with results of LOBI-PB in Fig. 4(a)–(c). For HOBI-PB, the errors are smaller in general and the difference between two different meshes are larger, indicating better accuracy and higher-order convergence.
(2) LOBI-PB, due to its simplicity in algorithm, shows more consistent convergence pattern. HOBI-PB, however affected by the quadrature rule and singularity removal schemes, shows some irregular patterns.
(3) We also observe that when the charge is located close to the interface e.g. at (0.9,0,0), for coarse mesh, LOBI-PB shows better accuracy than that from HOBI-PB. This is due to the fact that the close-to-boundary charge brings high variation of the induced charges on surface and multiple quadrature points selected in HOBI-PB amplify the variations. In LOBI-PB, there is only one quadrature point in each triangle at the centroid thus the scheme is less sensitive to the induced charges.
(4) The quality of triangular surface will slightly affect the convergence of the boundary integral PB solvers. The results from icosahedral triangulation routines show more uniform convergence pattern than that from MSMS.
(5) We further draw the error-vs-element plots for one charge located at (0,0,0) in Fig. 4(e) and (0.9,0,0) in Fig. 4(f) for both LOBI-PB (circle) and HOBI-PB (triangle) solvers on both MSMS (red, empty marker, solid line) and icosahedron (blue, filled marker, dashed line) triangles. By observing one color at a time (one kind of mesh at a time), we can see the pattern in Fig. 4(e) is uniform and HOBI-PB shows better accuracy (smaller y values) and faster convergence (larger slope). The pattern in Fig. 4(f) is tangled. By observing the slope, we can still see that HOBI-PB converges faster than LOBI-PB generally. However, the error of the HOBI-PB is bigger than LOBI-PB for coarse mesh due to the interaction between the induced charges on surface and the singular charge near the surface, as explained in observation and discussion (3). In practice, the partial charges are at least 1–2 Å away from the molecular surfaces therefore the slightly deteriorated pattern observed here when charges are close to the surfaces will unlikely happen.

In short, the numerical results of HOBI-PB and some other reference PB solvers on the spherical cavities compared with available analytical solutions quantitatively demonstrate the achieved better accuracy and higher-order convergence of HIBO-PB. The complexity of the higher-order schemes for quadrature and regularization introduces only minor instabilities but gain significantly improved accuracy and convergence. Next we use HOBI-PB and LOBI-PB to solve PB equation and compute electrostatic solvation energy on a protein.

3.4. Computing electrostatic solvation energy on protein 1ajj

The ultimate goal of HOBI-PB is to provide accurate electrostatic potentials for solvated biomolecules. With this solver, we solve the PB equation and compute the electrostatic solvation energy for many small-middle sized proteins. Here we take protein 1ajj with triangulation at different resolutions as an example. The coordinates and partial charges of the protein are obtained from CHARMM22 force field [40]. The numerical results show that HOBI-PB solves the PB equation with significantly improved accuracy compared with what LOBI-PB does.

The numerical results for computing electrostatic solvation energy for protein 1ajj are reported in Table 3. To produce these results, we discretize the molecular surface of protein 1ajj at different densities as seen in the first column of the table. Different densities result in different numbers of triangular elements which are listed in the second column of the table, characterizing the dimension of problem. In the third column, we report electrostatic solvation energies. We can see these values are very close to each other at different resolutions and they converge to a value near about -1138.49 kcal/mol. In column 4, we report the CPU time and it increases at the order of $\mathcal{O}(N^2)$. This reveals currently the most critical limitation of HOBI-PB as the $\mathcal{O}(N^2)$ computational cost eventually will make the HOBI-PB prohibitively expensive. To alleviate the pain, we take advantage of the convenient parallelization of boundary integral formulation, and we will see the parallelization performance next. Memory uses are shown in column 5 and we see a $\mathcal{O}(N)$ pattern, which is advantageous compared with the 3D mesh-based methods whose memory uses increase at $\mathcal{O}(N^2)$ or even $\mathcal{O}(N^3)$, where N is the number of unknowns. The last column is the number of iterations, and these stable results attribute to the well-posed integral formulation in Eqs. (8) and (9).

We next plot the solvation energies of HOBI-PB (solid circle) from Table 3 on Fig. 5 together with the solvation energies computed with LOBI-PB (empty circle) for the same protein at different resolutions. The plot shows results from LOBI-PB

Table 3
Testing results for protein 1ajj. HOBI-PB results showing electrostatic solvation energy E_{sol}, CPU time, memory usage; number of GMRES iterations (it.).

d	# of ele. N	E_{sol} (kcal/mol)	CPU (s)	Memory (Mbyte)	# of it.
1	6027	-1168.87	115	47	10
2	9198	-1152.42	266	70	10
4	17278	-1145.50	867	129	9
8	32386	-1140.60	3114	238	9
16	66558	-1139.23	17790	523	13
32	132028	-1138.38	56039	759	10
64	270680	-1138.49	254198	2116	11

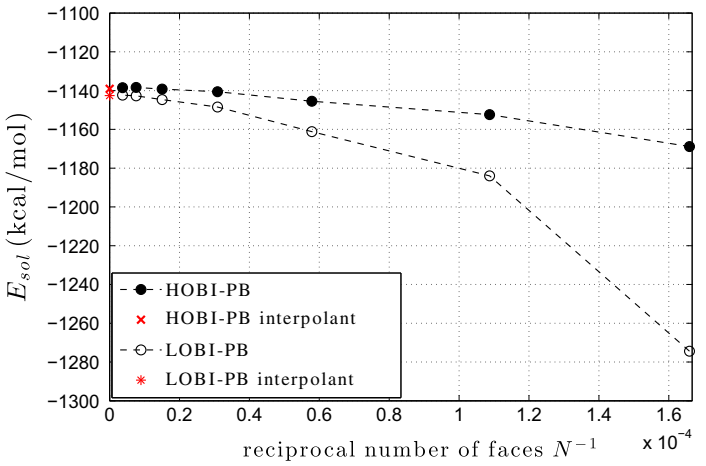

Fig. 5. Comparison of electrostatic solvation energy computed with HOBI-PB and LOBI-PB for protein 1ajj.

Table 4
MPI parallel performance for computing electrostatic solvation energy on protein 1ajj; p is number of CPUs, T_p is the time using p CPUs, T_1/pT_p is the parallel efficiency.

p	$N = 132028$		$N = 66558$		$N = 32386$	
	$T_p(s)$	T_1/pT_p	$T_p(s)$	T_1/pT_p	$T_p(s)$	T_1/pT_p
1	50457	100.0%	17755	100.0%	3060	100.0%
2	24740	102.0%	8790	101.0%	1528	100.1%
4	12877	98.0%	4381	101.3%	777	98.5%
8	6398	98.6%	2190	101.3%	406	94.3%
16	3321	95.0%	1090	101.8%	194	98.8%
32	1677	94.0%	571	97.2%	106	90.6%
64	860	91.7%	306	90.8%	62	77.4%

converge to that of HOBI-PB. By using cubic interpolation, we can see both methods eventually converge toward almost the same values (the red "x" for HOBI-PB and the red "$*$" for LOBI-PB). The advantage of HOBI-PB is that it can achieve high accuracy even at very coarse mesh. For example, the first solid point from right, which corresponds to electrostatic solvation energy of -1168.87 kcal/mol at density $d = 1$, is only 30 kcal/mol different from the interpolated true value at about -1139.09 kcal/mol. Similar patterns are observed on our other tests on different proteins.

We finally provided the MPI based parallel performance in terms of parallel efficiency of HOBI-PB for protein 1ajj at different meshes in Table 4. Due to the limitation of the resources, our results are generated with up to 64 CPUs. The maximum memories for each CPU in MPI implementation is about 1.5 G therefore for protein (1ajj) we solve the PB equation with maximum $d = 32$ requiring 759 M memory (note $d = 64$ requires more than 2 G memory per core from Table 3). From Table 4, we can see the CPU time is substantially reduced when the code is run in parallel on high performance computers. For example, for $d = 32$ with 132,028 elements, the serial work requires more than half of a day (50,457 s), the parallel work with 64 CPUs produce results in about 15 min (860 s). We see high parallel efficiency from the $\frac{T_1}{pT_p}$ column. When the dimension of the problem is sufficiently large (e.g. $N = 66,558$ or $N = 132,028$), the parallel efficiency with 64 CPUs is higher than 90%. We observed occasionally the interesting larger-than-one parallel efficiencies and those could be explained by the traffic fluctuation on the cluster or possibly the argument mentioned in [41].

In summary, the HOBI-PB solver solves the PB equation accurately on both spherical cavities and real biomolecules. The surface potential and electrostatic solvation energy computed with the solver is accurate, fast-convergent and stable. The parallelization of the solver is easy to implement and the parallel efficiency is attractively high.

4. Conclusions

This paper describes the schemes of a higher-order boundary integral Poisson–Boltzmann (HOBI-PB) solver. This solver discretizes the molecular surface with curved triangles, and performs numerical integral with four-point Gauss–Radau quadratures on regular triangles and with sixteen-point Gauss–Legendre quadratures on singular triangles. The singularities are regularized with a coordinate transformation. The numerical tests on spherical cavities show that HOBI-PB can achieve 1.5th order convergence of on surface potentials relative to boundary elements and these computed surface potentials are very accurate even at coarse mesh. In addition, the order of convergence does not compromise when the electric charge is

off-center or even closed to the surface. The numerical tests of HOBI-PB on biomolecules show much improved accuracy compared with results from the popular LOBI-PB solvers. The accurate surface potentials are of vital importance to molecular modeling that are sensitive to electrostatics near or on the molecular surface. To improve the efficiency of HOBI-PB, we also developed its MPI based parallel version. The numerical results demonstrate very encouraging parallel efficiency, e.g. above 90% when up to 64 CPUs work concurrently.

HOBI-PB achieves higher accuracy at the price of more complex algorithms. The limitation of the HOBI-PB is mainly at the problem dimension it can treat, which is subject to the available computing resources. For example, for the accessibility to up 64 CPUs each with 1.5 G memory per core at Alabama Supercomputing Center, the problem size HOBI-PB can handle for a reasonable long waiting time (<15 min) is about 150,000 elements. Considering the high accuracy, we can use fairly small density $1 \leqslant d \leqslant 5$, then PB equations on proteins with hundreds of residues can be conveniently solved. More computing resources will bring better performance on bigger problems. The rapid updating of computing power will definitely make HOBI-PB more and more capable.

There are many spaces in which HOBI-PB can be improved and extended. For example, we are looking for better triangulation programs for the molecular surfaces [42–44]. The currently adopted MSMS only provides 3 digits accuracy in vertices, positions and normal directions (we radially project vertices for spheres). In addition, the adoption of fast algorithms such as FMM [45] and treecode [46] to HOBI-PB, although considerably challenging due to the complexity of HOBI-PB schemes, is under our consideration. A more challenging problem is the application of HOBI-PB to molecular dynamics [47,48], where the PB equation will be solved at every time sampling. Furthermore, the higher-order schemes applied in HOBI-PB has the potential to solve other integral equations such as the integral forms of Helmholtz equation [49] and Maxwell Equations [50]. For these challenges, the application of the quadrature rules and the treatment of singularity will be similar, however, new challenges such as obtaining the well-posedness of the integral formulation and applying the fast algorithms need further investigation.

Acknowledgments

The authors thank Robert Krasny, Yongcheng Zhou, Benzhuo Lu and Peijun Li for helpful discussions. The work was supported by NSF Grant DMS-0915057, University of Alabama new faculty startup fund and Alabama Supercomputer Center.

References

[1] T. Schlick, Molecular Modeling and Simulation: An Interdisciplinary Guide, Springer, New York, 2002.
[2] N.A. Baker, Improving implicit solvent simulations: a Poisson-centric view, Current Opin. Struct. Biol. 15 (2005) 137–143.
[3] A. Miyazawa, Y. Fujiyoshi, N. Unwin, Structure and gating mechanism of the acetylcholine receptor pore, Nature 423 (2003) 949–955.
[4] N. Huang, Y. Chelliah, Y. Shan, C.A. Taylor, S.H. Yoo, C. Partch, C.B. Green, H. Zhang, J.S. Takahashi, Crystal structure of the heterodimeric CLOCK:BMAL1 transcriptional activator complex, Science 337 (2012) 189–194.
[5] F.M. Richards, Areas, volumes, packing and protein structure, Annu. Rev. Biophys. Bioeng. 6 (1977) 151–176.
[6] M.L. Connolly, Molecular surface triangulation, J. Appl. Crystallogr. 18 (1985) 499–505.
[7] J.G. Kirkwood, Theory of solution of molecules containing widely separated charges with special application to Zwitterions, J. Chem. Phys. 7 (1934) 351–361.
[8] M. Holst, F. Saied, Multigrid solution of the Poisson–Boltzmann equation, J. Comput. Chem. 14 (1993) 105–113.
[9] W. Rocchia, E. Alexov, B. Honig, Extending the applicability of the nonlinear Poisson–Boltzmann equation: multiple dielectric constants and multivalent ions, J. Phys. Chem. B 105 (2001) 6507–6514.
[10] W. Im, D. Beglov, B. Roux, Continuum solvation model: computation of electrostatic forces from numerical solutions to the Poisson–Boltzmann equation, Comput. Phys. Commun. 111 (1998) 59–75.
[11] R. Luo, L. David, M.K. Gilson, Accelerated Poisson–Boltzmann calculations for static and dynamic systems, J. Comput. Chem. 23 (2002) 1244–1253.
[12] N.A. Baker, D. Sept, M.J. Holst, J.A. McCammon, The adaptive multilevel finite element solution of the Poisson–Boltzmann equation on massively parallel computers, IBM J. Res. Dev. 45 (2001) 427–438.
[13] Z.H. Qiao, Z.L. Li, T. Tang, A finite difference scheme for solving the nonlinear Poisson–Boltzmann equation modeling charged spheres, J. Comput. Math. 24 (2006) 252–264.
[14] D. Chen, Z. Chen, C.J. Chen, W.H. Geng, G.W. Wei, MIBPB: a software package for electrostatic analysis, J. Comput. Chem. 32 (2011) 756–770.
[15] A. Juffer, E. Botta, B. van Keulen, A. van der Ploeg, H. Berendsen, The electric potential of a macromolecule in a solvent: a fundamental approach, J. Comput. Phys. 97 (1991) 144–171.
[16] A. Boschitsch, M. Fenley, H.-X. Zhou, Fast boundary element method for the linear Poisson–Boltzmann equation, J. Phys. Chem. B 106 (2002) 2741–2754.
[17] B.Z. Lu, X.L. Cheng, J.F. Huang, J.A. McCammon, Order N algorithm for computation of electrostatic interactions in biomolecular systems, PNAS 103 (2006) 19314–19319.
[18] M.D. Altman, J.P. Bardhan, J.K. White, 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) 132–153.
[19] R. Yokota, J.P. Bardhan, M.G. Knepley, L.A. Barba, T. Hamada, Biomolecular electrostatics using a fast multipole BEM on up to 512 GPUS and a billion unknowns, Comput. Phys. Commun. 182 (2011) 1272–1283.
[20] W.H. Geng, R. Krasny, A treecode-accelerated boundary integral Poisson–Boltzmann solver for electrostatics of solvated biomolecules, J. Comput. Phys., submitted for publication.
[21] C. Bajaj, S.-C. Chen, A. Rand, An efficient higher-order fast multipole boundary element solution for Poisson–Boltzmann-based molecular electrostatics, SIAM J. Sci. Comput. 33 (2011) 826–848.
[22] A. Bordner, G. 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) 353–367.
[23] J. Liang, S. Subramaniam, Computation of molecular electrostatics with boundary element methods, Biophys. J. 73 (1997) 1830–1841.
[24] Y.N. Vorobjev, H.A. Scheraga, A fast adaptive multigrid boundary element method for macromolecular electrostatic computations in a solvent, J. Comput. Chem. 18 (1997) 569–583.

[25] N.A. Baker, D. Sept, S. Joseph, M.J. Holst, J.A. McCammon, Electrostatics of nanosystems: application to microtubules and the ribosome, PNAS 98 (2001) 10037–10041.
[26] K.E. Forsten, R.E. Kozack, D.A. Lauffenburger, S. Subramaniam, Numerical solution of the nonlinear Poisson–Boltzmann equation for a membrane-electrolyte system, J. Phys. Chem. 98 (1994) 5580–5586.
[27] D.A. Beard, T. Schlick, Modeling salt-mediated electrostatics of macromolecules: the discrete surface charge optimization algorithm and its application to the nucleosome, Biopolymers 58 (2001) 106–115.
[28] R.J. Zauhar, R.S. Morgan, The rigorous computation of the molecular electric potential, J. Comput. Chem. 9 (1988) 171–187.
[29] B.J. Yong, A.M. Lenhoff, A boundary element method for molecular electrostatics with electrolyte effects, J. Comput. Chem. 11 (1990) 1080–1086.
[30] K.E. Atkinson, User's Guide to a Boundary Element Package for Solving Integral Equations on Piecewise Smooth Surfaces, Department of Mathematics, University of Iwoa, 1998. <http://www.cs.uiowa.edu/atkinson/bie.html>.
[31] C. Schwab, W. Wendland, On numerical cubatures of singular surface integrals in boundary element methods, Numer. Math. 62 (1992) 343–369.
[32] M.F. Sanner, A.J. Olson, J.C. Spehner, Reduced surface: an efficient way to compute molecular surfaces, Biopolymers 38 (1996) 305–320.
[33] J.R. Baumgardner, P.O. Frederickson, Icosahedral discretization of the two-sphere, SIAM J. Numer. Anal. 22 (1985) 1107–1115.
[34] B.Z. Lu, X.L. Cheng, J.A. McCammon, New-version-fast-multipole-method accelerated electrostatic calculations in biomolecular systems, J. Comput. Phys. 226 (2007) 1348–1366.
[35] J.E. Akin, Application and Implementation of Finite Element Methods, Academic Press Inc., London, 1982.
[36] Y.C. Zhou, M. Feig, G.W. Wei, Highly accurate biomolecular electrostatics in continuum dielectric environments, J. Comput. Chem. 29 (2008) 87–97.
[37] S.N. Yu, W.H. Geng, G.W. Wei, Treatment of geometric singularities in the implicit solvent models, J. Chem. Phys. 126 (2007) 244108.
[38] W.H. Geng, S.N. Yu, G.W. Wei, Treatment of charge singularities in the implicit solvent models, J. Chem. Phys. 128 (2007) 114106.
[39] S.Z. Deng, W. Cai, Extending the fast multipole method for charges inside a dielectric sphere in an ionic solvent: high-order image approximations for reaction fields, J. Comput. Phys. 227 (2007) 1246–1266.
[40] A.D. MacKerell Jr., D. Bashford, M. Bellott, J.D. Dunbrack, M.J. Evanseck, M.J. Field, S. Fischer, J. Gao, H. Guo, S. Ha, D. Joseph-McCarthy, L. Kuchnir, K. Kuczera, F.T.K. Lau, C. Mattos, S. Michnick, T. Ngo, D.T. Nguyen, B. Prodhom, W.E. Reiher, B. Roux, M. Schlenkrich, J.C. Smith, R. Stote, J. Straub, M. Watanabe, J. Wiorkiewicz-Kuczera, D. Yin, M. Karplus, All-atom empirical potential for molecular modeling and dynamics studies of proteins, J. Phys. Chem. 102 (1998) 3586–3616.
[41] D. Parkinson, Parallel efficiency can be greater than unity, Parallel Comput. 3 (3) (1986) 261–262.
[42] P. Bates, G.W. Wei, S. Zhao, Minimal molecular surfaces and their applications, J. Comput. Chem. 29 (2008) 380–391.
[43] M.X. Chen, B.Z. Lu, TMSmesh: a robust method for molecular surface mesh generation using a trace technique, J. Chem. Theory Comput. 7 (2011) 203–212.
[44] D. Xu, Y. Zhang, Generating triangulated macromolecular surfaces by Euclidean distance transform, PLoS ONE 4 (12) e8140. http://dx.doi.org/10.1371/journal.pone.0008140.
[45] L. Greengard, J. Huang, A new version of the fast multipole method for screened Coulomb interactions in three dimensions, J. Comput. Phys. 180 (2002) 642–658.
[46] P.J. Li, R. Krasny, H. Johnston, A Cartesian treecode for screened Coulomb particle interactions, J. Comput. Phys. 228 (2009) 3858–3868.
[47] W.H. Geng, G.W. Wei, Multiscale molecular dynamics using the matched interface and boundary method, J. Comput. Phys. 230 (2011) 435–457.
[48] Q. Lu, R. Luo, A Poisson–Boltzmann dynamics method with nonperiodic boundary condition, J. Chem. Phys. 119 (2003) 11035–11047.
[49] S. Amini, N.D. Maines, Preconditioned Krylov subspace methods for boundary element solution of the Helmholtz equation, Int. J. Numer. Methods Eng. 41 (1998) 875–898.
[50] A. Buffa, M. Costabel, C. Schwab, Boundary element methods for Maxwells equations on non-smooth domains, Numer. Math. 92 (2002) 679–710.