THE JOURNAL OF PHYSICAL CHEMISTRY B
A JOURNAL OF THE AMERICAN CHEMICAL SOCIETY

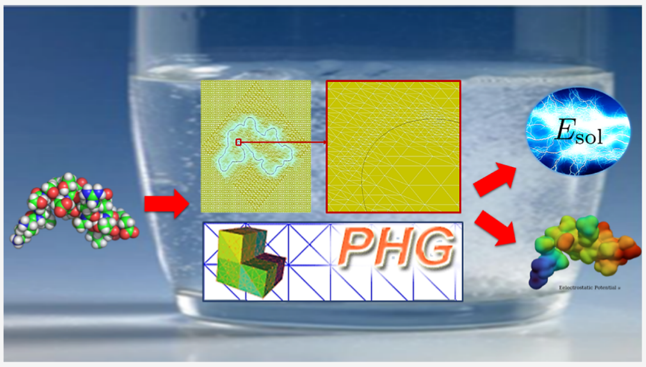
pubs.acs.org/JPCB Article

An Unfitted Finite Element Poisson−Boltzmann Solver with Automatic Resolving of Curved Molecular Surface

Ziyang Liu, Sheng Gui, Benzhuo Lu,* and Linbo Zhang*

Cite This: J. Phys. Chem. B 2024, 128, 6463−6475 Read Online

ACCESS | Metrics & More | Article Recommendations



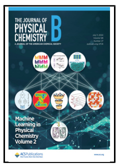
ABSTRACT: So far, the existing Poisson−Boltzmann (PB) solvers that accurately take into account the interface jump conditions need a pregenerated body-fitted mesh (molecular surface mesh). However, qualified biomolecular surface meshing and its implementation into numerical methods remains a challenging and laborious issue, which practically hinders the progress of further developments and applications of a bunch of numerical methods in this field. In addition, even with a molecular surface mesh, it is only a low-order approximation of the original curved surface. In this article, an interface-penalty finite element method (IPFEM), which is a typical unfitted finite element method, is proposed to solve the Poisson−Boltzmann equation (PBE) without requiring the user to generate a molecular surface mesh. The Gaussian molecular surface is used to represent the molecular surface and can be automatically resolved with a high-order approximation within our method. Theoretical convergence rates of the IPFEM for the linear PB equation have been provided and are well validated on a benchmark problem with an analytical solution (we also noticed from numerical examples that the IPFEM has similar convergence rates for the nonlinear PBE). Numerical results on a set of different-sized biomolecules demonstrate that the IPFEM is numerically stable and accurate in the calculation of biomolecular electrostatic solvation energy.

1. INTRODUCTION

The Poisson−Boltzmann equation (PBE) serves as a typical and effective implicit solvent model in the fields of biophysics and physical chemistry. Achieving improved efficiency, robustness, and ease of use of the PB solver has always been arousing many research interests (see our previous review[1]). A large number of methods have been developed and applied to solve the PBE.[2−16] DelPhi,[2] UHBD,[3] APBS,[4] GRASP,[5] AFMPB,[6−8] the PBEQ[9] module in CHARMM,[10] TABIPB,[11] MIBPB,[12] and SDPBS[13] are some examples of the successful PB solvers for computing biomolecular electrostatics. Due to the strong singularity and nonlinearity caused by its singular distribution, it is hard to obtain a numerical solution of the PBE. Many early works, like DelPhi,[2] UHBD,[3] PBEQ,[9] and APBS,[4] dealt with this difficulty by assigning singular charges to surrounding mesh points or other methods. These methods do not treat several essential features of the PBE, including the exact position of the molecular surface and the continuity condition of electric displacement on the molecular surface, which results in low accuracy of the surface potential. Although these errors have a relatively small effect on the calculation of electrostatic solvation energies and therefore these software are still widely used, the accurate surface potential distributions are still of interest to many researchers.[1] Based on a matched interface and boundary (MIB) method,[12] Zhou et al. successfully implemented the analytical molecular surface in their interface method for solving the PBE. There also exist a number of decomposition schemes for the PBE, like in refs[17,18] for finite difference methods and in refs[14,19,20] for finite element methods.

The finite element method (FEM), as detailed in the literature,[19,21−23] has effectively addressed some challenging numerical issues associated with the PBE, such as accurate treatments of the irregular geometry and interface conditions, charge singularity, and fast algorithms. Chern,[18] Xie,[14] and Chen[19] proposed several fundamental technical results, which made it possible to develop fast finite element algorithms. Holst et al.[19] proposed a number of fundamental technical results, including a priori pointwise bounds on solutions to the continuous and finite element discretized solutions to the PBE. Xie et al.[14,24] proposed a solution decomposition and minimization scheme, together with an analysis on solution existence and uniqueness. The FEM for PB advanced by Cortis

Received: March 22, 2024
Revised: May 6, 2024
Accepted: May 13, 2024
Published: July 1, 2024



and Friesner[25] makes use of a similar Galerkin formulation. Shestakov et al.[26] developed a nonlinear Poisson–Boltzmann solver based on the FEM, using Newton–Krylov iterations coupled with pseudo-transient continuation. Lu's group[27] developed a finite element solution for the PBE with a two-dimensional periodicity for membrane channel proteins, with different numerical treatments of the singular charge distributions in the channel protein.

To the best of our knowledge, all existing PB solvers (especially for FEMs) that accurately treat the interface jump conditions need a pregenerated body-fitted mesh, i.e., a molecular surface mesh. So far, for large biomolecular systems (e.g., see the review[1]), the creation of a mesh continues to pose a challenge for the implementation of FEM, primarily due to the highly irregular shapes of biomolecular systems. This task becomes more complex as it involves recognizing the irregular molecular surface and accurately describing it to resolve the molecular structures in greater detail. Furthermore, the molecular surface mesh is only a low-order approximation of the original curved surface. This fact is the main hurdle for the progress of implementation and application of a bunch of numerical methods relying on molecular surface meshing in this area. Triangular and tetrahedral meshing are most commonly adopted for generating an unstructured mesh.[28] Several programs have been developed to produce surface triangular meshes for biomolecules, but there are very few software tools available for directly generating tetrahedral meshes for biomolecules. A common approach for obtaining tetrahedral meshes of biomolecules involves first creating surface triangular meshes and then generating tetrahedral volume meshes based on these surface meshes. A widely used package, TetGen,[29] can generate a tetrahedral mesh based on a surface triangular mesh.

Our group has established a toolchain to create a qualified mesh for finite element/boundary element modeling of large molecular systems. This toolchain consists of the following key components: surface meshing, quality improvement, volume mesh generation, and membrane-protein mesh construction. First, a manifold triangular mesh of the Gaussian surface is generated using our program TMSmesh.[30−32] In the second step, the software package SMOPT is used[33] to improve the qualities of the surface triangular mesh and at the same time preserve the manifoldness. Finally, the package TetGen is used to generate a tetrahedral volume mesh, which is then used for 3D finite element simulations. More details can be found in the literature.[34,35] With this toolchain, we have successfully generated meshes for many protein systems and performed finite element simulations on them. However, efficiently generating high-quality biomolecular meshes via the entire toolchain involves a laborious process with a significant level of manual intervention.[33,36,37]

Thus, we aim to find a new strategy that does not rely on high-quality molecular surface meshing while ensuring the accuracy of the numerical results. From the analysis of the PBE (see, for example, ref 1), the electrostatic potential ϕ can be split into two components: $\phi = G + \phi^r$. G collects all singularities from the Delta distributions with available analytical formula, and ϕ^r is the solution of a typical elliptic interface problem. Over the past decades, various unfitted mesh methods, in which the interface is allowed to cross mesh elements, have been proposed for solving elliptic interface problems. Examples of such methods are the multiscale finite element methods,[38] the immersed interface method,[39] the immersed finite element methods,[40] the extended finite element methods,[41] and the penalty finite element methods.[42] Unfitted mesh methods involving penalty terms can be traced back to the penalty finite element methods proposed by Babuška.[43] Hansbo and Hansbo[44] presented an unfitted finite element method, in which through the introduction of a geometry-dependent average of flux at the interface, they established stable discretization and proved that the linear finite element scheme is nearly optimal in two dimensions. This approach has motivated many follow-up works, e.g., the cut finite element methods,[45] the unfitted finite element methods,[46] and the unfitted discontinuous Galerkin methods.[47] Wu and Xiao[42] proposed an *hp*-unfitted discontinuous Galerkin method for solving the linear elliptic interface problem, and Liu et al.[48] proved that, for the general interface problem in H^1, $\boldsymbol{H}(\text{curl})$, and $\boldsymbol{H}(\text{div})$, the method converges in broken H^1 norm at an optimal rate with respect to h and at a suboptimal rate with respect to p by a factor of p.

In this paper, we propose a robust parallel interface-penalty finite element method (IPFEM) for the PBE to calculate electrostatics in biomolecular systems without requiring molecular mesh generation. The optimal L^2 error estimate and energy norm error estimate of our IPFEM are obtained for the linear PBE. Numerical examples show optimal convergence of the proposed finite element method for a piecewise smooth solution. Our method also exhibits good performance for solving the PBE in molecular cases.

The contributions are listed as follows.

1. Our method only needs a simple tetrahedral mesh covering the whole computational domain without generating a molecular surface mesh.
2. The Gaussian molecular surface is approximated by the piecewise polynomial function, and we use high-order numerical quadratures of the interface to accurately treat the interface jump conditions.

The rest of the article is organized as follows. In Section 2, we introduce our methods, including the PB model and its regularization form in Section 2.1 and the discrete format of IPFEM in Section 2.2. The validation tests are provided to show the optimal convergence rates of our method in Section 3, as well as protein tests and their applications to solvation energy calculations. In Section 4 we conclude our paper. The job shell script of our method is presented in the Data Availability part.

2. METHODS

2.1. Mathematical Model. In this work, the PBE is used to study electrostatic interactions in biomolecular systems. Let $\Omega \in \mathbb{R}^3$ denote the calculation domain with a convex and Lipschitz-continuous boundary $\partial\Omega$ and $\Omega = \Omega_m \cup \Omega_s$, respectively.

The molecule for which we want to calculate the electrostatic potential is located in region Ω_m with a relative permittivity of ε_m. The region Ω_s contains N_c different species of ions with a relative permittivity of ε_s. Ω_m and Ω_s are separated by the interface (molecule surface) Γ. Figure 1 shows a 2D illustration of the computational domain of the PB model.

The PBE model[14] in SI (Systéke International) units is defined as

$$-\nabla\cdot(\varepsilon_m\varepsilon_0\nabla\phi(\vec{x})) = \sum_{i=1}^{N_m} z_i e_c \delta(\vec{x} - \vec{x}_i) \text{ in } \Omega_m$$

$$-\nabla\cdot(\varepsilon_s\varepsilon_0\nabla\phi(\vec{x})) + e_c\sum_{j=1}^{N_c} z_j^c c_j e^{-\frac{z_j^c e_c}{k_B T}\phi} = 0 \text{ in } \Omega_s \tag{1}$$

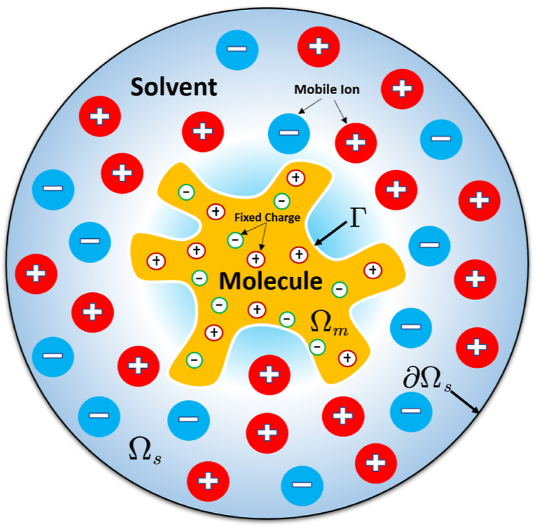


Figure 1. Illustration of computational domain Ω.

where $\phi(\vec{x})$ is the electric potential in volts, c_j is the bulk density of mobile ion species j with charge number z_j^c. N_m denotes the number of fixed charges within Ω_m, and z_i and $\vec{x}_i$, $i = 1, 2, ..., N_m$ denote the charge number and position of fixed charges within Ω_m, respectively. The values and units of physical parameters ε_0, e_c, k_B, and T are listed in Table 1.

Table 1. Parameters of the PBE Model in SI Units

parameter	name	value	unit (abbr.)
ε_0	vacuum permittivity	$8.854187817 \times 10^{-12}$	F/m
e_c	elementary charge	$1.602176565 \times 10^{-19}$	C
k_B	Boltzmann constant	$1.380648813 \times 10^{-23}$	J/K
T	absolute temperature	298.15	K

To simplify the presentation, we only consider a symmetric 1:1 salt in this paper, which means that $N_c = 2$, $Z_1^c = 1$, $Z_2^c = -1$, and $c_1 = c_2 = c_b$. So the PBE model (eq 1) can be simplified as

$$-\nabla\cdot(\varepsilon\varepsilon_0\nabla\phi(\vec{x})) + \bar{\kappa}^2(\vec{x})\left(\frac{k_BT}{e_c}\right)\sinh\left(\frac{e_c\phi(\vec{x})}{k_BT}\right) = \sum_{i=1}^{N_m} z_ie_c\delta(\vec{x}-\vec{x}_i) \text{ in } \Omega \tag{2}$$

where the dielectric coefficient ε is defined as $\varepsilon = \varepsilon_m$ in Ω_m and $\varepsilon = \varepsilon_s$ in Ω_s. $\bar{\kappa}$ is the modified Debye–Hückel parameter, $\bar{\kappa} = 0$ in the molecule region Ω_m, $\bar{\kappa} = \sqrt{\varepsilon_s\varepsilon_0}\,\kappa$ in the solution region Ω_s. κ is the Debye–Hückel parameter with $\kappa^2 = 2c_b\frac{e_c^2}{k_BT\varepsilon_s\varepsilon_0}$.

In fact, the PBE model (eq 2) is formally equivalent to a coupling of two equations for the electrostatic potential in different regions Ω_m and Ω_s through the molecule surface

$$-\nabla\cdot(\varepsilon_m\varepsilon_0\nabla\phi(\vec{x})) = \sum_{i=1}^{N_m} z_ie_c\delta(\vec{x}-\vec{x}_i) \text{ in } \Omega_m$$

$$-\nabla\cdot(\varepsilon_s\varepsilon_0\nabla\phi(\vec{x})) + \varepsilon_s\varepsilon_0\kappa^2\left(\frac{k_BT}{e_c}\right)\sinh\left(\frac{e_c\phi(\vec{x})}{k_BT}\right) = 0 \text{ in } \Omega_s$$

These two equations are coupled together through the boundary conditions on the interface (molecule surface) Γ

$$[\phi] = 0, \qquad \left[\varepsilon\frac{\partial\phi}{\partial\mathbf{n}_\Gamma}\right] = 0$$

where the jump $[v]$ of v on Γ is defined as $[v] = v|_{\Omega_m} - v|_{\Omega_s}$ and $\mathbf{n}_\Gamma$ is the unit normal vector on Γ, pointing from Ω_m to Ω_s.

We choose a widely used boundary condition $\phi|_{\partial\Omega} = \phi_D$ on $\partial\Omega$, which is determined by a known analytical solution to one of the simplifications of the linear PBE[49]

$$\phi_D = \left(\frac{e_c}{4\pi\varepsilon_s\varepsilon_0}\right)\sum_{i=1}^{N_m}\frac{z_ie^{-\kappa\|\vec{x}-\vec{x}_i\|}}{\|\vec{x}-\vec{x}_i\|} \tag{3}$$

Taking the first term in the series expansion $\sinh x = x + \frac{x^3}{3!} + \frac{x^5}{5!} + \cdots$ as an approximation of $\sinh x$, we can get the linear Poisson–Boltzmann equation (LPBE)

$$-\nabla\cdot(\varepsilon\varepsilon_0\nabla\phi(\vec{x})) + \bar{\kappa}^2(\vec{x})\phi(\vec{x}) = \sum_{i=1}^{N_m} z_ie_c\delta(\vec{x}-\vec{x}_i) \text{ in } \Omega \tag{4}$$

For computational convenience, the dimensionless equation is introduced. Defining $\beta = \frac{1}{k_BT}$ as the reciprocal of Boltzmann energy, the dimensionless PBE and boundary condition can be obtained for $u(\vec{x}) = e_c\beta\phi(\vec{x})$

$$-\nabla\cdot(\varepsilon\varepsilon_0\nabla u(\vec{x})) + \bar{\kappa}^2(\vec{x})\sinh(u(\vec{x})) = e_c^2\beta\sum_{i=1}^{N_m} z_i\delta(\vec{x}-\vec{x}_i) \text{ in } \Omega \tag{5}$$

Due to the presence of singular terms $\delta(\vec{x})$ in the PBE, its solution has singularities. In this study, a solution (potential) decomposition of the PBE is introduced to remove the potential singular part

$$u = u_r + G, \qquad G(\vec{x}) = \frac{\alpha}{4\pi\varepsilon_m}\sum_{i=1}^{N_m}\frac{z_i}{\|\vec{x}-\vec{x}_i\|} \text{ in } \Omega_m \tag{6}$$

with $-\nabla\cdot(\varepsilon_m\varepsilon_0\nabla G) = e_c^2\beta\sum_{i=1}^{N_m} z_i\delta(\vec{x}-\vec{x}_i)$, here $\alpha := \frac{e_c^2}{k_BT\varepsilon_0}$.

Substituting this decomposition (eq 6) into the dimensionless PBE (eq 5), we then obtain the regularized Poisson–Boltzmann equation (RPBE)

$$\begin{cases} -\nabla\cdot(\varepsilon\nabla u_r) + 2c_b\bar{\alpha}\sinh(u_r) = 0 \text{ in } \Omega \\ g_D = [u_r] = -G, \qquad g_N = \left[\varepsilon\frac{\partial u_r}{\partial\mathbf{n}}\right] = \varepsilon_m\frac{\partial G}{\partial\mathbf{n}} \text{ on } \Gamma \\ u_r = \frac{\alpha}{4\pi\varepsilon_s}\sum_{i=1}^{N_m} z_i\frac{e^{-\kappa\|\vec{x}-\vec{x}_i\|}}{\|\vec{x}-\vec{x}_i\|} := g \text{ on } \partial\Omega \end{cases} \tag{7}$$

where $\bar{\alpha} = 0$ in Ω_m and $\bar{\alpha} = \alpha$ in Ω_s.

Similarly, the regularized LPBE is as follows

$$\begin{cases} -\nabla\cdot(\varepsilon\nabla u_r) + 2c_b\bar{\alpha}u_r = 0 \text{ in } \Omega \\ g_D = [u_r] = -G, \qquad g_N = \left[\varepsilon\frac{\partial u_r}{\partial\mathbf{n}}\right] = \varepsilon_m\frac{\partial G}{\partial\mathbf{n}} \text{ on } \Gamma \\ u_r = \frac{\alpha}{4\pi\varepsilon_s}\sum_{i=1}^{N_m} z_i\frac{e^{-\kappa\|\vec{x}-\vec{x}_i\|}}{\|\vec{x}-\vec{x}_i\|} := g \text{ on } \partial\Omega \end{cases} \tag{8}$$

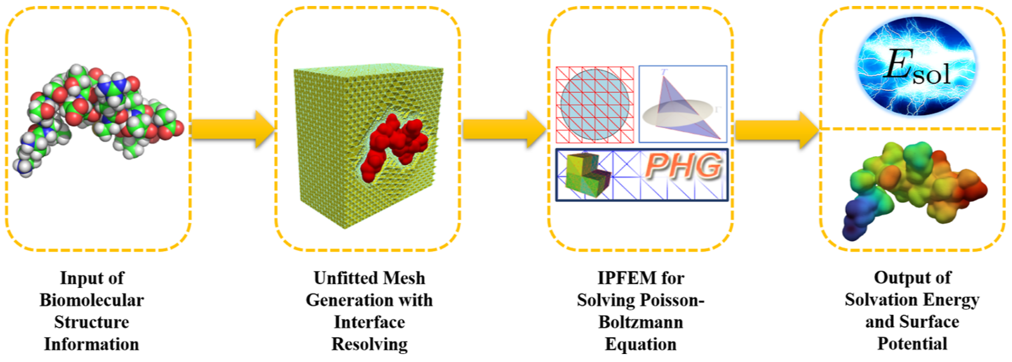

Esol
PHG
Input of Biomolecular Structure Information
Unfitted Mesh Generation with Interface Resolving
IPFEM for Solving Poisson-Boltzmann Equation
Output of Solvation Energy and Surface Potential

Figure 2. Pipeline of our algorithm.

2.2. IPFEM for the PBE. In this subsection, we present the algorithm for solving the PBE via the IPFEM. The IPFEM belongs to the class of unfitted finite element methods in which the finite element mesh (in our case a tetrahedral mesh) is not required to be fitted with the interface (in our case the molecular surface). In unfitted finite element methods, an element that intersects with the interface is called an interface element or cut element. In the case of a molecular surface, an interface element is divided into two parts by the molecular surface, one in Ω_m (the molecular part) and the other one in Ω_s (the solvent part). The basic idea of the IPFEM consists of using two sets of degrees of freedom in each interface element to define two finite element functions, which approximate the solution in the solvent part and the molecular part, and handling the interface conditions on the molecular surface by penalizing them in the finite element discretization. A formal and more detailed description of the IPFEM will be given in Section 2.2.2.

The input of our method is the PQR file, which includes coordinates of centers and radii of atoms. We first represent the molecular surface by using the Gaussian molecular surface 9 and project it into finite element space to present the level set function. After that we construct the IPFEM for the PBE and implement it using the open-source finite element toolbox Parallel Hierarchical Grid (PHG).[50] We present a parallel IPFEM of the Poisson–Boltzmann (IPFEMPB) solver. Figure 2 illustrates the flowchart of IPFEMPB.

2.2.1. Unfitted Mesh and Interface Resolution. There are various kinds of definitions for a molecular surface,[51] including the van der Waals surface (VDWs), the solvent-accessible surface (SAS),[52] the solvent-excluded surface (SES),[53] etc.

Different from these definitions, the Gaussian surface[54] is defined as a level set from the Gaussian density maps as $\{\vec{x} \in \mathbb{R}^3, \varphi(\vec{x}) = C\}$ where the definition of the Gaussian density map is as follows

$$\varphi(\vec{x}) = \sum_{i=1}^{N_m} e^{-d(\|\vec{x}-\vec{x}_i\|^2 - r_i^2)} \tag{9}$$

here, $\vec{x}_i$ and r_i are the coordinate of center and the radius of the i-th atom, respectively. The parameter d is positive and controls the decay rate of the kernel functions. C is the isovalue parameter, and it controls the volume enclosed by the Gaussian molecular surface. These two parameters, d and C, can be chosen properly to make the Gaussian molecular surface approximate the SES, SAS, and VDW surfaces well.[55] The Gaussian surface and other mentioned surface types are all widely used in the community. Compared with the other definitions, the Gaussian surface has the following advantages.

- The Gaussian surface is smooth.
- The Gaussian surface provides a realistic representation of the electron density of a molecule as compared to other molecular surface definitions.[54]
- The Gaussian surface is well established[30,56–58] and has a wide range of applications in computational biology, such as docking problems,[59] molecular shape comparisons,[60] calculating SAS areas,[61] and the generalized Born models.[62]
- The Gaussian surface is currently suited to be handled within our finite element toolbox PHG.

In this paper, the Gaussian molecular surface is chosen to represent the molecular surface. We first read the PQR file to get the coordinates of centers and the radii of atoms and calculate the Gaussian density map by (eq 9). Meanwhile, a uniform tetrahedral mesh is constructed based on the size of the computation domain. Mesh elements near the Gaussian surface are locally refined to limit the relative curvature of the surface with respect to the size of the neighboring elements. The literature[63] provided a method to calculate the curvature of implicit surfaces defined by the level set function $\omega(x, y, z)$. The max curvature of the surface can be calculated by

$$\kappa_{\max} = \kappa_H + \sqrt{|\kappa_H^2 - \kappa_K|} \tag{10}$$

where $\kappa_H = \frac{\omega_{uu} + \omega_{vv}}{2\,|\omega_n|}$ represents the mean curvature and $\kappa_K = \frac{\omega_{uu}\omega_{vv} - \omega_{uv}^2}{|\omega_n|^2}$ represents the Gaussian curvature, $\boldsymbol{u}$ and $\boldsymbol{v}$ are arbitrary vectors defining the tangent plane to the surface and $|\omega_n|^2 = \omega_x^2 + \omega_y^2 + \omega_x^2$.

Then we can construct the level set function $L(\vec{x}) = \Pi_k(C - \varphi(\vec{x}))$, where Π_k denotes the kth-order finite element interpolation operator.[64] Clearly, $\vec{x} \in \Omega_m$, $L(\vec{x}) < 0$, $\vec{x} \in \Omega_s$, $L(\vec{x}) > 0$ and $\vec{x} \in \Gamma$, $L(\vec{x}) = 0$. Algorithm 1 illustrates the unfitted mesh generation process with interface resolving.

Figures 3 and 4 illustrate the final computational mesh, and Figure 5 illustrates a cut plane of the body-fitted mesh. Notice that the surface presented from the body-fitted mesh is a linear approximation of the Gaussian molecular surface while our method can choose a high-order finite element interpolation to make the interpolated molecular surface approximate the Gaussian molecular surface well.

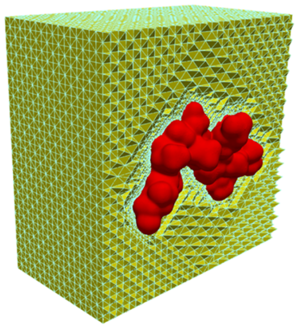


Figure 3. Illustration of interface resolving.

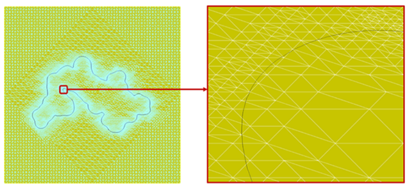


Figure 4. Cut plane of the computational mesh of IPFEM.

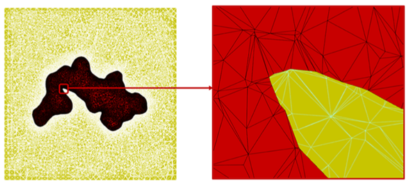


Figure 5. Cut plane of the body-fitted mesh.

Algorithm 1 Unfitted mesh generation with interface resolving

```
1: Given $PQR$ file, ctol $> 0$, an uniform tetrahedral mesh $\mathscr{T}_h$, the order $k$ of finite element interpolation operator, Gaussian parameters $C$ and $d$, mark set $M_{\text{mark}} = \emptyset$, flag = TRUE;
2: Read $PQR$ file and calculate $\varphi(\vec{x})$ by (9);
3: while flag $\neq$ TRUE do
4:     flag = FALSE, $M_{\text{mark}} = \emptyset$;
5:     Construct the finite element level set function $L(\vec{x}) = \Pi_{k,\mathscr{T}_h}(C - \varphi(\vec{x}))$;
6:     for element $T$ in $\mathscr{T}_h$ do
7:         if $T \cap \Gamma \neq \emptyset$ then
8:             Get the element size $h_T$ and the max curvature $\kappa_{\max}$ of $L(\vec{x})$ in $T$ by (10);
9:             if $\kappa_{\max} h_T \leq$ ctol then
10:                $M_{\text{mark}} = M_{\text{mark}} \cup \{T\}$, flag = TRUE;
11:            end if
12:        end if
13:    end for
14:    Refine all elements $T \in M_{\text{mark}}$ get $\mathscr{T}_h'$, $\mathscr{T}_h \leftarrow \mathscr{T}_h'$;
15: end while
```

2.2.2. Interface-Penalty Finite Element Method. This subsection introduces the IPFEM for the PBE. Let $\{\mathcal{T}_h\}$ be a family of conforming and shape regular partitions of the domain Ω into closed tetrahedra. For any $T \in \mathcal{T}_h$ we define, h_T :=diam(T), $h := \max_{T\in\mathcal{T}_h} h_T$ denote the size of element T and mesh $\mathcal{T}_h$, respectively. $\mathcal{I}_h$ represents the partition of the interface (molecular surface) Γ with

$$\mathcal{I}_h = \{e\colon e = T \cap \Gamma,\ |e| > 0, T \in \mathcal{T}_h\}$$

where $|e|$ is the area of e. For convenience, we denote $\Omega_1 = \Omega_m$ and $\Omega_2 = \Omega_s$. Figure 6 illustrates $\mathcal{I}_h$ with the red part. Let $T^e \in \mathcal{T}_h$ be one of the element(s) containing e and

$$T_j^e := T^e \cap \Omega_j,\ j = 1, 2$$

so we have

$$e \subset \partial T_1^e \cap \partial T_2^e,\ T_j^e \subset \bar{\Omega}_j,\ j = 1, 2$$

In addition, we define the set of boundary surfaces.

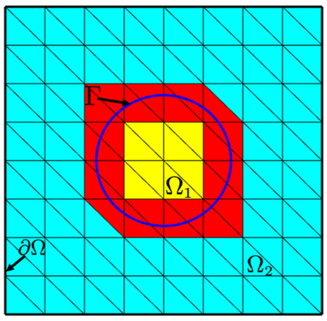


Figure 6. Illustration of interface elements $\mathcal{I}_h$ (the red part).

$$\mathcal{I}_D = \{e\colon e = \partial T \cap \partial\Omega,\ |e| > 0, T \in \mathcal{T}_h\}$$

We introduce the energy space as

$$V := \{v\colon v|_{\Omega_i} = v_i,\ \text{where } v_i \in H^1(\Omega),\ v_i|_{T\cap\Omega_i} \in H^2(T \cap \Omega_i),\ \forall\ T \in \mathcal{T}_h,\ i = 1, 2\} \tag{11}$$

and the continuous finite element space $U_h^p := \{v_h \in H^1(\Omega) : v_h|_T \in \mathcal{P}_p(T),\ \forall\ T \in \mathcal{T}_h\}$, where $\mathcal{P}_p(T)$ denotes the set of all polynomials of degrees less than or equal to p for any $T \in \mathcal{T}$. Our interface-penalty finite element approximation space V_h^p is defined as the space of piecewise continuous finite element functions

$$V_h^p := \{v_h\colon v_h|_{\Omega_i} \in U_h^p,\ i = 1, 2\} \tag{12}$$

That is, for any $v_h \in V_h^p$, the restriction of v_h onto Ω_i is a continuous piecewise polynomial for $i = 1, 2$. Defining the average $\{v\}$ of v on the interface Γ as $\{v\} := \frac{v|_{\Omega_1} + v|_{\Omega_2}}{2}$, testing the RPBE (eq 7) by any $v \in V$, using Green's formula and the identity $[uv] = [u]\{v\} + [v]\{u\}$, we obtain

$$\sum_{i=1}^{2} \int_{\Omega_i} \varepsilon\nabla u\cdot\nabla v - \sum_{e\in\mathcal{I}_D} \int_e \varepsilon\frac{\partial u}{\partial \mathbf{n}} v - \sum_{e\in\mathcal{I}_h} \int_e \left\{\varepsilon\frac{\partial u}{\partial \mathbf{n}}\right\}[v] + 2c_b\alpha\int_{\Omega_2} \sinh(u)v = \int_\Gamma g_N\{v\} \tag{13}$$

Define

$$a_h^0(u, v) := \sum_{i=1}^{2} \int_{\Omega_i} \varepsilon\nabla u\cdot\nabla v - \sum_{e\in\mathcal{I}_D} \int_e \varepsilon\frac{\partial u}{\partial \mathbf{n}} v - \sum_{e\in\mathcal{I}_h} \int_e \left\{\varepsilon\frac{\partial u}{\partial \mathbf{n}}\right\}[v]$$

and $f_h^0(v) := \int_\Gamma g_N\{v\}$. We refer to the symmetric interface penalty finite element method in the literature[42] and define the bilinear form $a_h(\cdot,\cdot)$ on $V \times V$

$$a_h(u, v) := a_h^0(u, v) - S(u, v) + J_0(u, v) + J_1(u, v) \tag{14}$$

where

$$S(u, v) := \sum_{e\in\mathcal{I}_D}\int_e \varepsilon\frac{\partial v}{\partial \mathbf{n}}u + \sum_{e\in\mathcal{I}_h}\int_e [u]\left\{\varepsilon\frac{\partial v}{\partial \mathbf{n}}\right\}$$

$$J_0(u, v) := \sum_{e\in\mathcal{I}_D}\frac{\gamma_0 p^2}{h_{T^e}}\int_e uv + \sum_{e\in\mathcal{I}_h}\frac{\gamma_0 p^2}{h_{T^e}}\int_e [u][v]$$

$$J_1(u, v) := \sum_{e\in\mathcal{I}_h}\frac{\gamma_1 h_{T^e}}{p^2}\int_e \left[\varepsilon\frac{\partial u}{\partial \mathbf{n}}\right]\left[\varepsilon\frac{\partial v}{\partial \mathbf{n}}\right]$$

and the linear form $f_h(\cdot)$ on V

$$f_h(u, v) := f_h^0(v) - S^f(v) + J_0^f(v) + J_1^f(v) \qquad (15)$$

where

$$S^f(v) := \int_{\mathcal{I}_D}\varepsilon\frac{\partial v}{\partial \mathbf{n}}g + \int_\Gamma g_D\left\{\varepsilon\frac{\partial v}{\partial \mathbf{n}}\right\}$$

$$J_0^f(v) := \sum_{e\in\mathcal{I}_h}\frac{\gamma_0 p^2}{h_{T^e}}\int_e g_D[v] + \sum_{e\in\mathcal{I}_D}\frac{\gamma_0 p^2}{h_{T^e}}\int_e gv$$

$$J_1^f(u, v) := \sum_{e\in\mathcal{I}_h}\frac{\gamma_1 h_{T^e}}{p^2}\int_e g_N\left[\varepsilon\frac{\partial v}{\partial \mathbf{n}}\right]$$

In (eq 14), $S(u, v)$ is to maintain the symmetry of the format, that is $a_h(u, v) = a_h(v, u)$, $J_0(u, v)$ and $J_1(u, v)$ are the interface penalty terms in which γ_0 and γ_1 are nonnegative numbers to be specified later. The tricks on dealing with the discontinuities are from the interior penalty discontinuous or continuous Galerkin methods.[65,66] $S^f(v)$, $J_0^f(v)$, and $J_1^f(v)$ correspond to $S(u, v)$, $J_0(u, v)$, and $J_1(u, v)$, respectively.

Obviously, solution u to the problem (eq 7) satisfies the following equation

$$a_h(u, v) + 2c_b\alpha\int_{\Omega_2}\sinh(u)v = f_h(v) \qquad \forall\ v\in V \qquad (16)$$

The interface-penalty finite element approximation to RPBE (eq 7) reads as follows: find $u_h \in V_h^p$, such that

$$a_h(u_h, v_h) + 2c_b\alpha\int_{\Omega_2}\sinh(u_h)v_h = f_h(v_h) \qquad \forall\ v_h\in V_h^p \qquad (17)$$

Similarly, the interface-penalty finite element approximation to LPBE (eq 8) reads as follows: find $u_h \in V_h^p$, such that

$$a_h(u_h, v_h) + 2c_b\alpha\int_{\Omega_2}u_h v_h = f_h(v_h) \qquad \forall\ v_h\in V_h^p \qquad (18)$$

Notice that in (eqs 14 and 15), we need to calculate both volume integrals in the subdomains and surface integrals on their common boundary (patch of Gaussian molecular surface). Our group[67] proposed a general-purpose, robust, and high-order numerical algorithm to handle these tasks, with the source code freely available in the recent distributions of PHG which can be downloaded at http://lsec.cc.ac.cn/phg/index_en.htm (see doc/quad-XFEM.pdf in the source code of PHG for related documentations).

We define the "energy" norm $|||v|||_{1,h}$ on the space V

$$|||v|||_{1,h} := \left(\sum_{i=1}^{2}\|\varepsilon^{1/2}\nabla v\|_{L^2(\Omega_i)}^2 + 2c_b\alpha\,\|v\|_{L^2(\Omega_2)}^2 + \sum_{e\in\mathcal{I}_h}\frac{\gamma_1 h_{T^e}}{p^2}\left\|\left[\varepsilon\frac{\partial v}{\partial \mathbf{n}}\right]\right\|_{L^2(e)} + \sum_{e\in\mathcal{I}_h}\frac{\gamma_0 p^2}{h_{T^e}}\|[v]\|_{L^2(e)}^2 + \sum_{e\in\mathcal{I}_D}\frac{\gamma_0 p^2}{h_{T^e}}\|v\|_{L^2(e)}^2 + \sum_{e\in\mathcal{I}_h}\frac{h_{T^e}}{\gamma_0 p^2}\left\|\left\{\varepsilon\frac{\partial v}{\partial \mathbf{n}}\right\}\right\|_{L^2(e)}^2 + \sum_{e\in\mathcal{I}_D}\frac{h_{T^e}}{\gamma_0 p^2}\left\|\varepsilon\frac{\partial v}{\partial \mathbf{n}}\right\|_{L^2(e)}^2\right)^{1/2}$$

For the linear PBE, our method has the following L^2 error and "energy" error estimates.

Theorem 1: let $s \geq 2$ be an integer and let $\mu = \min\{p + 1,s\}$. Let u and u_h be the solutions of eqs 8 and 18, respectively. Then there exists constants α_0, h_0, independent of h, p, and penalty parameters such that for $\gamma_1 > 0$ and $\gamma_0 \geq \alpha_0/\gamma_1$, the following error estimates hold

$$|||u - u_h|||_{1,h} \lesssim \gamma_0^{1/2}\frac{h^{\mu-1}}{p^{s-3/2}}\left(\sum_{i=1}^{2}\|u\|_{H^s(\Omega_i)}^2\right)^{1/2} \qquad \forall\ 0 < h \leq h_0$$

Theorem 2: under the conditions of Theorem 1, holds the following estimate

$$\|u - u_h\|_{L^2(\Omega)} \lesssim \gamma_0\frac{h^{\mu}}{p^{s-1}}\left(\sum_{i=1}^{2}\|u\|_{H^s(\Omega_i)}^2\right)^{1/2}, \qquad \forall\ 0 < h \leq h_0$$

The proofs of the above theoretical results can be obtained with similar analysis techniques as in the literature.[48] Note that γ_0 and γ_1 cannot take minor values in order to ensure the convergence and accuracy of the method according to Theorems 1 and 2, but we also found in numerical experiments that too large values for them can lead to difficulty or even failure in solving PBE. We usually choose $\gamma_0 = 10$ and $\gamma_1 = 1$ because they perform well in the majority of cases.

2.3. Nonlinear Iteration. A backtracking line search Newton-like method[68] is adopted to solve the above nonlinear eq 17. Let u^j be the approximation solution of u at the j-th Newton iteration, which could be expressed with basis functions of the finite element space V_h^p: $\{\Phi_i\}_{i=1}^n$. Referring to eq 17, we define a nonlinear vector function $F(x)$, whose t-th component is given as $F(u)_t = a_h(u, \Phi_t) + \int_{\Omega_m}\kappa\sinh(u)\Phi_t - f_h(\Phi_t)$. The Jacobian matrix of $F(u)$ is denoted with $F'(u)$, and the t-th, k-th element of which is described as $(F'(u))_{t,k} = a_h(\Phi_k, \Phi_t) + \int_{\Omega_m}\kappa\cosh(u)\Phi_k\Phi_t$, $t, k = 1, 2, ..., n$. Moreover, the merit function of $F(u)$ is the sum of squares, which is defined by $R(u) = \|F(u)\|_2^2 = \sum_{i=1}^{N}(F(u)_i)^2$. The framework of the backtracking line search Newton-like method is stated in Algorithm 2.

Algorithm 2 Backtracking line search Newton-like method

```
Given $h \in (0,1)$, $\rho \in (0,1)$, $\gamma_{\text{init}} > 0$, $\gamma_{\min} > 0$, $\eta_F$, $\eta_N$ and $j = 1$;
Solve the LPBE (18) to get $u^0$;
while $j \geq 0$ do
    Calculate a solution $p^j$ from the Newton equations: $F'(u^j)p^j = -F(u^j)$;
    Set $\gamma = \gamma_{\text{init}}$;
    while $R(u^j + \gamma p^j) > R(u^j) + h\gamma \nabla R(u^j)^{\top} p^j$ and $\gamma > \gamma_{\min}$ do
        Set $\gamma = \rho\gamma$;
    end while
    if $\frac{||u^j - u^{j-1}||_{L^2}}{||u^j||_{L^2}} \leq \eta_N$ and $R(u^j) \leq \eta_F$ then
        break;
    end if
    $j = j + 1$;
end while
```

3. RESULTS AND DISCUSSION

In this section, we carry out a suite of numerical examples to test the accuracy and robustness of our IPFEM and compare it with the boundary element method (BEM) and the standard finite element method (FEM). For simplicity, we set ε_m = 2.0, ε_s = 80.0, and T = 298.15 K, and unless otherwise specified, let c_b = 0.1 M, γ_0 = 10.0, and γ_1 = 1.0.

Our method is implemented using the open-source finite element toolbox Parallel Hierarchical Grid (PHG),[50] in which the integrals in the cut elements are computed using the corresponding numerical quadrature functions provided in PHG.[48] All numerical experiments were done on the high-performance computers of State Key Laboratory of Scientific and Engineering Computing, Chinese Academy of Sciences, which consist of dual Intel Gold 6140 CPU nodes (2 × 18 cores, 2.30 GHz) and a 100 Gbps EDR Infiniband network.

3.1. Validation Tests. A simple test (a sphere in a cube) is carried out in this subsection to check the accuracy of IPFEM. We assume that the solute region is an atom of radius R in the units of Å, and the only fixed charge q is located at the center of the sphere. Taking the center of the sphere as the coordinate origin, the LPBE (eq 8) has the following analytical solution

$$\begin{cases} \phi_m(r) = \dfrac{q}{4\pi\varepsilon_0 R}\left(\dfrac{1}{\varepsilon_s(1+\kappa R)} - \dfrac{1}{\varepsilon_m}\right) + \dfrac{q}{4\pi\varepsilon_0\varepsilon_m r}, \text{ in } \Omega_m \\ \phi_s(r) = \dfrac{q}{4\pi\varepsilon_0\varepsilon_s}\dfrac{e^{\kappa R}}{1+\kappa R}\dfrac{e^{-\kappa r}}{r}, \text{ in } \Omega_s \end{cases} \tag{19}$$

where $r = \sqrt{x^2 + y^2 + z^2}$ represents the distance to the center of the sphere.

Here, we select the calculation region as a cube Ω = [−2 Å, 2 Å]^3 and the solute region as a single atom sphere $\{x^2 + y^2 + z^2 \leq 1\}$. Ω is uniformly divided into several tetrahedra as the initial mesh for the calculation (see Figure 7).

We solve the example problem on five grids with elements that are bisected three times successively. Figure 8 shows the errors

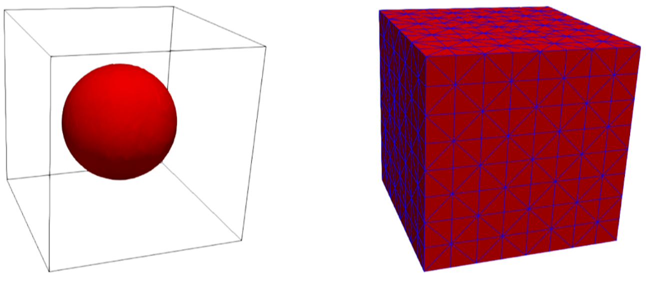


Figure 7. Calculation area for validation tests.

and convergence orders of IPFEM for the LPBE, where p is a parameter in V_h^p and N is the number of degrees of freedom. The L^2 error and H^1 error are defined as

$$||u - u_h||_{L^2} = \left(\int_\Omega |u - u_h|^2\right)^{1/2}$$

$$||u - u_h||_{H^1} = ||u - u_h||_{L^2} + ||\nabla(u - u_h)||_{L^2}$$

where u and u_h represent the analytical solution and the numerical solution, respectively.

We can find that our method has L^2 and H^1 norm convergence rates, which validates the theoretical results.

Similarly, we tested the convergence of IPFEM for the nonlinear RPBE (eq 7) in the same way. A test analytical solution for the electrostatic potential is defined as follows

$$u(\vec{x}) = \begin{cases} x^2 + xy + z, \text{ in } \Omega_m \\ \sin(x + 2y + 3z), \text{ in } \Omega_s \end{cases} \tag{20}$$

The source term and the corresponding boundary conditions can be calculated accordingly.

The results are displayed in Figure 9, which shows that the relative errors decrease as the time of refinement increases, verifying the accuracy and applicability of IPFEM to the three-dimensional nonlinear Poisson−Boltzmann equation (NPBE).

3.2. Protein Tests. In this subsection, to further validate the effectiveness of our method, we apply IPFEM to seven proteins with PDB names/IDs ADP, 1V4Z, 1A36, 3LOD, 1RMP, 1BL8, and AChE and compare our results with the BEM simulations for LPBE and the standard FEM simulations for nonlinear RPBE. High-quality molecular surface meshes are a prerequisite for using BEM and FEM in the implicit solvent model. However, the surface mesh generated by TMSmesh 2.0 can not be directly used for BEM simulation or generation of body-fitted volume meshes due to singular triangles with tiny angles or very short edges. We use our group's SMOPT software to improve the quality of these meshes. The improved mesh is used in the BEM[6−8] simulations and the generation of corresponding surface-conforming volume meshes that are required in the FEM[69] simulations. The BEM software used is a publicly available PB solver, AFMPB. In FEM simulations, the corresponding volume meshes are generated by TetGen.

The PDB files of these proteins were downloaded from the PDB website (https://www.rcsb.org/) and then converted to PQR files by the tool PDB2PQR[70] with AMBER force field. These protein molecules have 39, 266, 1402, 2246, 3478, 5892, and 8280 atoms with net charges $-3e_c$, $-1e_c$, $-42e_c$, $-5e_c$, $-8e_c$, $+8e_c$, and $-9e_c$, respectively. Figure 10 shows their molecular structures.

For protein tests, we typically use the solvation energy to validate the accuracy of computational results. In the implicit solvent continuum dielectric approach, the electrostatic solvation energy E_{sol} is commonly estimated in the units of kilocalories per mole (kcal/mol) by

$$E_{\text{sol}} = \frac{N_A}{4184}\frac{k_B T}{2e_c}\int_\Omega \rho_f(\vec{x})(u_{\text{sol}} - u_{\text{ref}})(\vec{x})\, \mathrm{d}\vec{x} \tag{21}$$

where u_{sol} and u_{ref} denote the electrostatic potential functions in the solvent and reference states, respectively. ρ_f denotes the fixed charge density function. By using the decomposition (eq 6) technique in our method, the electrostatic solvation energy is equivalently calculated as

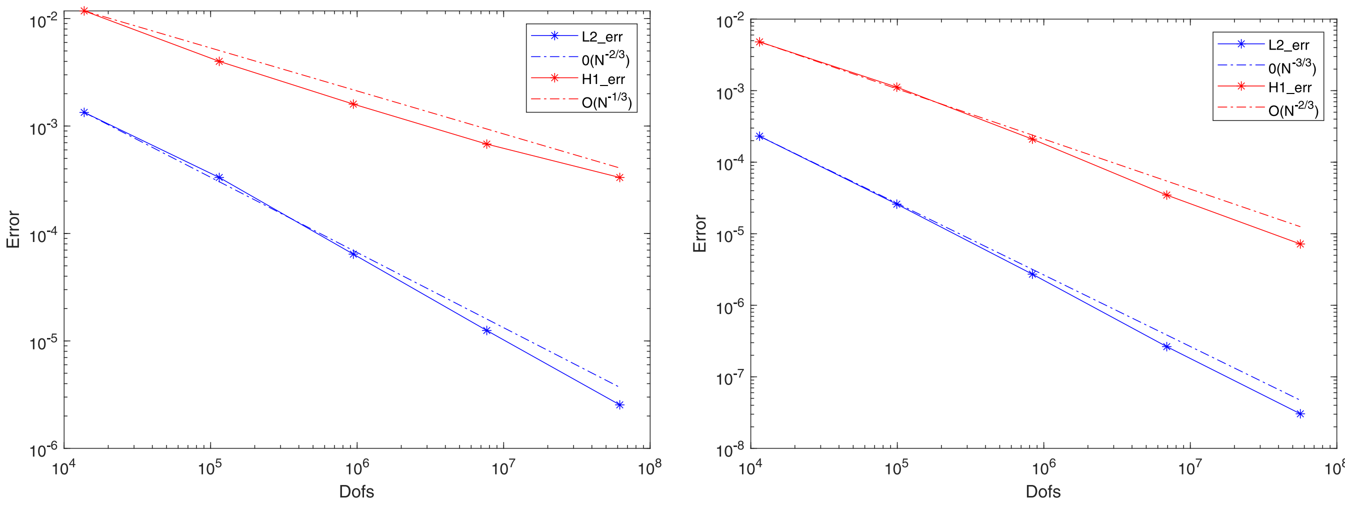


Figure 8. Errors and convergence orders of IPFEM for the LPBE (left: $p = 1$, right: $p = 2$).

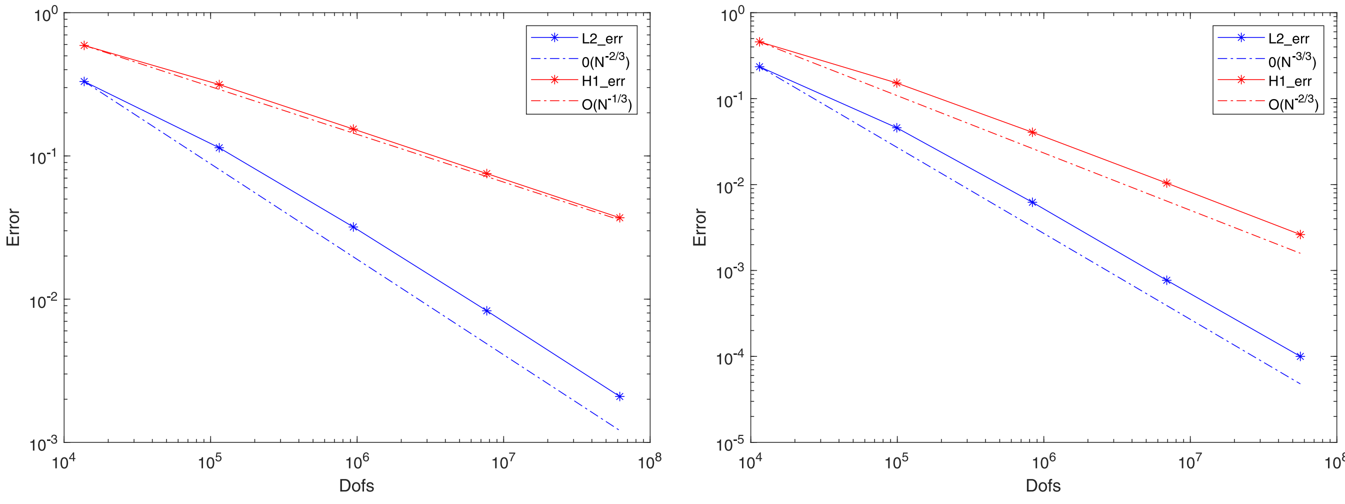


Figure 9. Errors and convergence orders of IPFEM for the RPBE (left: $p = 1$, right: $p = 2$).

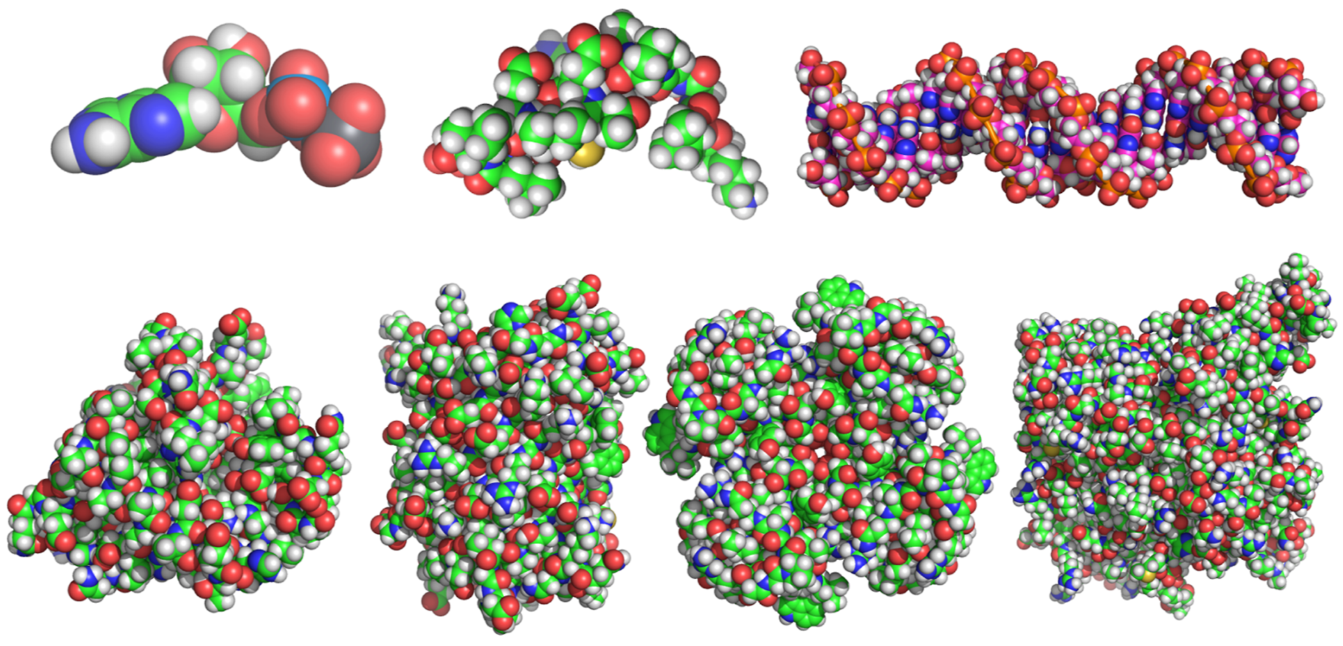


Figure 10. Molecular structure (left to right, up to down): ADP, 1V4Z, 1A36, 3LOD, 1RMP, 1BL8, and AChE.

Table 2. Solvation Energies Obtained with IPFEM and AFMPB for the Linear PBE (Unit: kcal/mol)

c_b (M)	protein	number of atoms	electrical charge	$E_{\text{sol}}^{\text{AFMPB}}$	$E_{\text{sol}}^{\text{IPFEM}}$	relative difference (%)
0.05	ADP	39	$-3e_c$	−253.80	−252.25	0.61
	1V4Z	266	$-1e_c$	−252.52	−251.40	0.45
	1A36	1402	$-42e_c$	−8049.01	−8067.72	0.23
	3LOD	2246	$-5e_c$	−982.71	−988.67	0.61
	1RMP	3478	$-8e_c$	−1775.13	−1803.71	1.61
	1BL8	5892	$+8e_c$	−2450.38	−2448.11	0.09
	AChE	8280	$-9e_c$	−4046.86	−4059.55	0.32
0.10	ADP	39	$-3e_c$	−254.16	−252.61	0.44
	1V4Z	266	$-1e_c$	−253.71	−251.63	0.83
	1A36	1402	$-42e_c$	−8066.19	−8078.28	0.15
	3LOD	2246	$-5e_c$	−984.16	−990.09	0.60
	1RMP	3478	$-8e_c$	−1776.69	−1805.26	1.61
	1BL8	5892	$+8e_c$	−2454.90	−2449.11	0.23
	AChE	8280	$-9e_c$	−4050.09	−4060.05	0.25
0.15	ADP	39	$-3e_c$	−254.41	−252.88	0.60
	1V4Z	266	$-1e_c$	−253.92	−251.63	0.83
	1A36	1402	$-42e_c$	−8075.43	−8081.75	0.08
	3LOD	2246	$-5e_c$	−985.18	−991.15	0.61
	1RMP	3478	$-8e_c$	−1777.81	−1806.38	1.60
	1BL8	5892	$+8e_c$	−2458.44	−2449.88	0.34
	AChE	8280	$-9e_c$	−4052.46	−4060.38	0.20

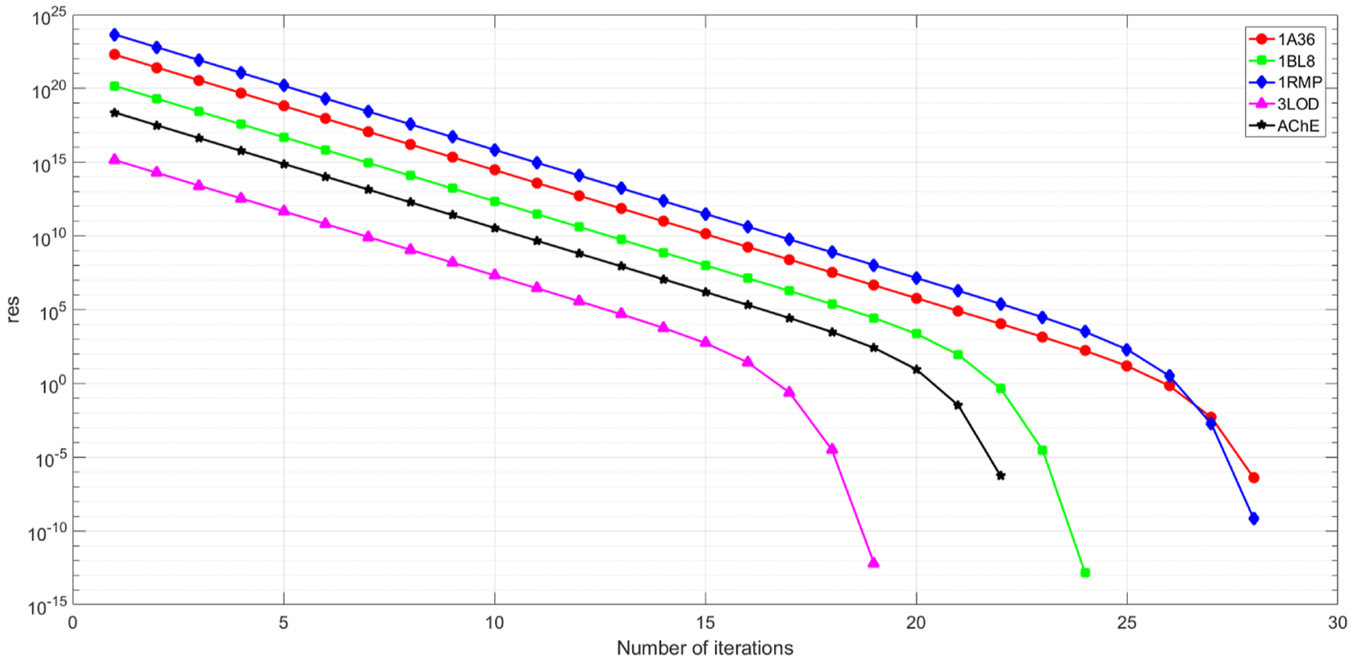


Figure 11. Iteration errors $R(u)$ of Algorithm 2.

$$E_{\text{sol}} = \frac{N_A}{4184} \cdot \frac{k_B T}{2e_c} \sum_{i=1}^{N_m} z_i u_r(\vec{x}_i) \tag{22}$$

First we compare the solvation energies obtained using the IPFEM method for a set of molecules at different ionic concentrations (0.05, 0.1, and 0.15 M) with the BEM solver AFMPB. Obviously, Table 2 shows that IPFEM can produce converged results without a body-fitted mesh, and the calculated solvation energies are close to the results obtained with AFMPB, validating the accuracy of our method for solving LPBE.

We also test the performance of IPFEM for RBPE (eq 7), and Figure 11 reports the convergence process of our Newton iterative Algorithm 2 for solving the RPBE (eq 17) for biomolecules. From this figure, it can be seen that our method stably converges within 30 iterations for general cases. Figure 12, produced with ParaView (https://www.paraview.org/), shows the computed nonlinear PB electrostatic potentials mapped on the molecular surface.

To validate the accuracy of our method for solving nonlinear RPBE, we compare the solvation energies obtained using the IPFEM method for a set of molecules at different ionic concentrations (0.05, 0.1, and 0.15 M) with the results of standard FEM. Table 3 shows the results.

3.3. Performance. The performance of our method is discussed in this subsection. First, we tested the parallel performance of our IPFEM solver using proteins ADP and

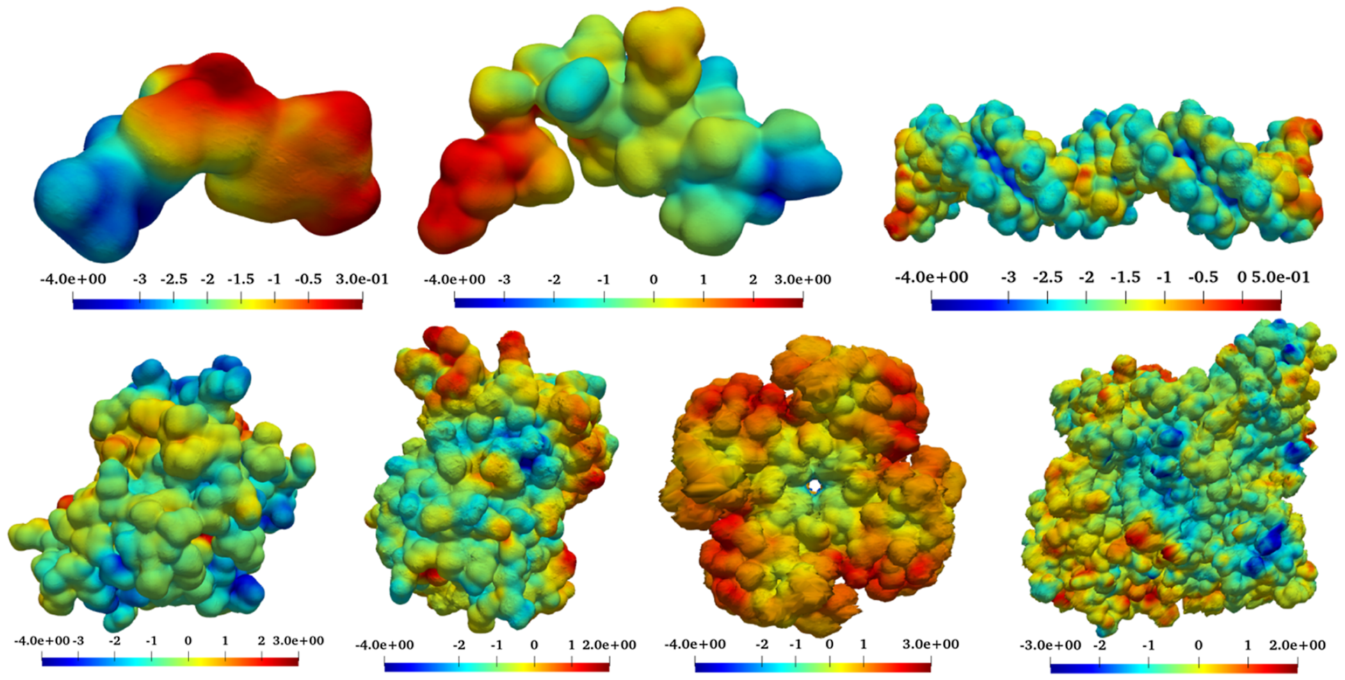

-4.0e+00 -3 -2.5 -2 -1.5 -1 -0.5 3.0e-01
-4.0e+00 -3 -2 -1 0 1 2 3.0e+00
-4.0e+00 -3 -2.5 -2 -1.5 -1 -0.5 0 5.0e-01
-4.0e+00 -3 -2 -1 0 1 2 3.0e+00
-4.0e+00 -2 -1 0 1 2.0e+00
-4.0e+00 -2 -1 0 1 3.0e+00
-3.0e+00 -2 -1 0 1 2.0e+00

Figure 12. Electrostatic potential ϕ (unit: kcal/mole) of molecules (left to right, up to down): ADP, 1V4Z, 1A36, 3LOD, 1RMP, 1BL8, and AChE.

Table 3. Solvation Energies Obtained with IPFEM and the Standard FEM for the Nonlinear PBE (Unit: kcal/mol)

c_b (M)	protein	number of atoms	electrical charge	$E_{\text{sol}}^{\text{FEM}}$	$E_{\text{sol}}^{\text{IPFEM}}$	relative difference (%)
0.05	ADP	39	$-3e_c$	−255.82	−254.64	0.46
	1V4Z	266	$-1e_c$	−253.39	−253.42	0.01
	1A36	1402	$-42e_c$	−8045.87	−8099.56	0.66
	3LOD	2246	$-5e_c$	−1006.43	−992.37	1.39
	1RMP	3478	$-8e_c$	−1789.80	−1810.39	1.15
	1BL8	5892	$+8e_c$	−2452.94	−2455.59	0.11
	AChE	8280	$-9e_c$	−3953.76	−3928.74	0.63
0.10	ADP	39	$-3e_c$	−255.99	−255.06	0.36
	1V4Z	266	$-1e_c$	−253.45	−253.26	0.07
	1A36	1402	$-42e_c$	−8047.97	−8107.21	0.73
	3LOD	2246	$-5e_c$	−1006.64	−994.06	1.25
	1RMP	3478	$-8e_c$	−1790.03	−1812.18	1.23
	1BL8	5892	$+8e_c$	−2453.52	−2456.87	0.13
	AChE	8280	$-9e_c$	−3953.96	−3929.57	0.63
0.15	ADP	39	$-3e_c$	−256.07	−255.32	0.29
	1V4Z	266	$-1e_c$	−253.49	−254.03	0.18
	1A36	1402	$-42e_c$	−8048.58	−8111.52	0.78
	3LOD	2246	$-5e_c$	−1006.74	−995.16	1.15
	1RMP	3478	$-8e_c$	−1790.13	−1813.33	1.29
	1BL8	5892	$+8e_c$	−2453.52	−2457.04	0.14
	AChE	8280	$-9e_c$	−3954.03	−3930.11	0.60

1A36 as examples. The initial tetrahedral mesh contained 163840 elements, and the computational mesh obtained after Algorithm 1 contained 352086 elements for protein ADP and 5698923 elements for protein 1A36. Table 4 illustrates the total time (refine mesh according to Algorithm 1 and solve LPBE) and the parallel efficiency. For protein ADP, the parallel efficiency can be maintained at around 70% at 256 processes. After that the parallel efficiency drops significantly; since the size of the problem is not large enough, the reduction in computation time by increasing the parallel size cannot cover the increase in communication overheads. For protein 1A36, the parallel efficiency of the algorithm can still be maintained above

Table 4. Parallel Efficiency of the IPFEM Solver (Protein ADP)

MPI processes	ADP		1A36	
	T^{tol} (s)	efficiency	T^{tol} (s)	Efficiency
1	430			
4	112	0.96		
16	32	0.84	18080	
64	8.6	0.78	4185	1.08
256	2.43	0.69	1228	0.92
1024	1.17	0.36	336.4	0.84

Table 5. Comparison of CPU Time and Memory Usage between IPFEM and FEM

protein	cores		elements		CPU time (s)		memory usage (GB)	
	IPFEM	FEM	IPFEM	FEM	IPFEM	FEM	IPFEM	FEM
ADP	36	36	457,437	400,984	8.9	1.23	0.5	0.3
1V4Z	144	144	3,068,854	3,551,137	18.0	4.35	2.06	1.56
1A36	360	360	9,554,178	15,176,252	52.6	7.27	6.06	6.10
3LOD	540	540	16,070,603	23,095,787	85.2	14.36	9.52	8.64

80% at 1024 processes, which demonstrates the excellent parallel scalability of our solver.

In Table 5, we report the computational costs to obtain the numerical solutions of RPBE for proteins ADP, 1V4Z, 1A36, and 3LOD using IPFEM and FEM (the meshes are provided for FEM). Here, both methods adopt backtracking line search Newton-like method (Algorithm 2) to solve the nonlinear equation, and each involved linear system in the test is solved by the GMRES with the additive Schwarz preconditioner. We count the CPU time of solving one Newton-like iteration step and the maximum memory usage for the entire solving process. Though the computation time with IPFEM is about 4−8 times more than with traditional FEM for comparable precisions, we think it is acceptable for practical applications. Moreover, the time for generating interface-fitted meshes, which are only required by traditional FEM was not counted in the comparisons. The most time-consuming part for the IPFEM is the numerical quadrature in the interface elements, which is a topic of our future studies.

4. CONCLUSIONS

The traditional method of the PBE in biomolecules-solution system needs biomolecular surface meshing, which is complex and tedious. In this work, we have proposed an IPFEM for solving the PBE. This method does not require molecular mesh generation and adopts techniques from the standard interior penalty discontinuous Galerkin method to handle the transmission conditions across the interface. To depict the Gaussian molecular surface, we employ interpolation of the Gaussian density map within the finite element space to characterize the level set function. Numerical examples have also been provided to illustrate the performance of our method. Validation tests are given to check the accuracy of IPFEM. We noticed that IPFEM has a convergence rate for the NPBE similar to that for the linear PBE in numerical examples. Our method also exhibits good performance for solving the PBE on molecular cases. In the future, the theoretical convergence of IPFEM for the NPBE will be studied, and the method will be further developed and applied to more complex models (e.g., Poisson−Nernst−Planck equations, SMPB equations,[71,72] etc.) and biosystems (like membrane-channel protein systems). Based on PHG, we also consider extending our method to other molecular surfaces (e.g., VDWs, SAS, SES, etc.).

APPENDIX

In the following, we provide an example shell script for executing the IPFEMPB package that can be found at https://github.com/bzlu-Group/IPFEMPB.

```
mpirun -np 32 ./IPFEMPB \
        -epsilon_solute 2.0 \
        -epsilon_solvent 80.0 \
        -concentration 0.10 \
        -temp 298.15 \
        -pqr ADP.pqr
```

where mpirun -np 32 ./IPFEMPB denotes running IPFEMPB using 32 MPI processes, -epsilon_solute denotes the relative permittivity of the solute molecule, -epsilon_solvent denotes the relative permittivity of the solvent, -concentration denotes the salt concentration with unit mol/L, -temp denotes the temperature with unit K, and -pqr denotes the PQR file.

The output file which contained electrostatic solvation energy and other information will be generated in the working directory. We also provide a tool for visualization of the molecular surface potential and more details can be found from the readme file of the uploaded package.

■ ASSOCIATED CONTENT

Data Availability Statement

The data that support the findings of this study are available within the article. The source code is available from https://github.com/bzlu-Group/IPFEMPB.

■ AUTHOR INFORMATION

Corresponding Authors

Benzhuo Lu − *ICMSEC, LSEC, NCMIS, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing 100190, China; School of Mathematical Sciences, University of Chinese Academy of Sciences, Beijing 100049, China;* orcid.org/0000-0003-4159-1532; Email: bzlu@lsec.cc.ac.cn

Linbo Zhang − *ICMSEC, LSEC, NCMIS, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing 100190, China; School of Mathematical Sciences, University of Chinese Academy of Sciences, Beijing 100049, China*; Email: zlb@lsec.cc.ac.cn

Authors

Ziyang Liu − *ICMSEC, LSEC, NCMIS, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing 100190, China; School of Mathematical Sciences, University of Chinese Academy of Sciences, Beijing 100049, China;* orcid.org/0009-0003-6995-8620

Sheng Gui − *Key Laboratory of Systems and Control, Institute of Systems Science, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing 100190, China; School of Mathematical Sciences, University of Chinese Academy of Sciences, Beijing 100049, China;* orcid.org/0000-0003-0739-9221

Complete contact information is available at:
https://pubs.acs.org/10.1021/acs.jpcb.4c01894

Notes

The authors declare no competing financial interest.

■ ACKNOWLEDGMENTS

Z.L. and L.Z.'s work was supported by National Key R&D Program of China (Grant numbers 2020YFA0711900, 2020YFA0711904). S.G. and B.L.'s work was supported by the Strategic Priority Research Program of Chinese Academy of Sciences (Grant number XDB0500000) and the National Natural Science Foundation of China (Grant numbers 12371413 and 22073110). All authors' work was supported by State Key Laboratory of Scientific and Engineering Computing (LSEC) and National Center for Mathematics and Interdisciplinary Sciences of Chinese Academy of Sciences (NCMIS). The authors thank our group student Minrui Lv for her help.

■ REFERENCES

(1) Lu, B.; Zhou, Y.; Holst, M.; McCammon, J. Recent progress in numerical methods for the Poisson-Boltzmann equation in biophysical applications. *Commun. Comput. Phys.* **2008**, *3*, 973−1009. https://www.global-sci.org/intro/article_detail/cicp/7885.html

(2) Klapper, I.; Hagstrom, R.; Fine, R.; Sharp, K.; Honig, B. Focusing of electric fields in the active site of Cu-Zn superoxide dismutase: Effects of ionic strength and amino-acid modification. *Proteins: Struct., Funct., Bioinf.* **1986**, *1*, 47−59.

(3) Madura, J. D.; Briggs, J. M.; Wade, R. C.; Davis, M. E.; Luty, B. A.; Ilin, A.; Antosiewicz, J.; Gilson, M. K.; Bagheri, B.; Scott, L. R.; et al. Electrostatics and diffusion of molecules in solution: simulations with the University of Houston Brownian dynamics program. *Comput. Phys. Commun.* **1995**, *91*, 57−95.

(4) Baker, N. A.; Sept, D.; Joseph, S.; Holst, M. J.; McCammon, J. A. Electrostatics of nanosystems: Application to microtubules and the ribosome. *Proc. Natl. Acad. Sci. U.S.A.* **2001**, *98*, 10037−10041.

(5) Petrey, D.; Honig, B. *Macromolecular Crystallography, Part D; Methods in Enzymology*; Academic Press, 2003; Vol. *374*; pp 492−509.

(6) Lu, B.; Cheng, X.; Huang, J.; McCammon, J. A. AFMPB: an adaptive fast multipole Poisson−Boltzmann solver for calculating electrostatics in biomolecular systems. *Comput. Phys. Commun.* **2010**, *181*, 1150−1160.

(7) Zhang, B.; Peng, B.; Huang, J.; Pitsianis, N. P.; Sun, X.; Lu, B. Parallel AFMPB solver with automatic surface meshing for calculation of molecular solvation free energy. *Comput. Phys. Commun.* **2015**, *190*, 173−181.

(8) Zhang, B.; DeBuhr, J.; Niedzielski, D.; Mayolo, S.; Lu, B.; Sterling, T. DASHMM accelerated adaptive fast multipole Poisson-Boltzmann solver on distributed memory architecture. *Commun. Comput. Phys.* **2019**, *25*, 1235−1258.

(9) Im, W.; Beglov, D.; Roux, B. Continuum solvation model: Computation of electrostatic forces from numerical solutions to the Poisson-Boltzmann equation. *Comput. Phys. Commun.* **1998**, *111*, 59−75.

(10) Brooks, B. R.; Bruccoleri, R. E.; Olafson, B. D.; States, D. J.; Swaminathan, S.; Karplus, M. CHARMM: A program for macromolecular energy, minimization, and dynamics calculations. *J. Comput. Chem.* **1983**, *4*, 187−217.

(11) Geng, W.; Krasny, R. A treecode-accelerated boundary integral Poisson-Boltzmann solver for electrostatics of solvated biomolecules. *J. Comput. Phys.* **2013**, *247*, 62−78.

(12) Zhou, Y.; Feig, M.; Wei, G.-W. Highly accurate biomolecular electrostatics in continuum dielectric environments. *J. Comput. Chem.* **2008**, *29*, 87−97.

(13) Jiang, Y.; Xie, Y.; Ying, J.; Xie, D.; Yu, Z. SDPBS web server for calculation of electrostatics of ionic solvated biomolecules. *Comput. Math. Biophys.* **2015**, *3*, 179−196.

(14) Xie, D. New solution decomposition and minimization schemes for Poisson−Boltzmann equation in calculation of biomolecular electrostatics. *J. Comput. Phys.* **2014**, *275*, 294−309.

(15) Xie, Y.; Ying, J.; Xie, D. SMPBS: Web server for computing biomolecular electrostatics using finite element solvers of size modified Poisson-Boltzmann equation. *J. Comput. Chem.* **2017**, *38*, 541−552.

(16) Ying, J.; Xie, D. A new finite element and finite difference hybrid method for computing electrostatics of ionic solvated biomolecule. *J. Comput. Phys.* **2015**, *298*, 636−651.

(17) Zhou, Z.; Payne, P.; Vasquez, M.; Kuhn, N.; Levitt, M. Finite-difference solution of the Poisson−Boltzmann equation: Complete elimination of self-energy. *J. Comput. Chem.* **1996**, *17*, 1344−1351.

(18) Chern, I.-L.; Liu, J.-g.; Wang, W.-C. Accurate evaluation of electrostatics for macromolecules in solution. *Methods Appl. Anal.* **2003**, *10*, 309−328.

(19) Chen, L.; Holst, M. J.; Xu, J. The finite element approximation of the nonlinear Poisson−Boltzmann equation. *SIAM J. Numer. Anal.* **2007**, *45*, 2298−2320.

(20) Holst, M.; McCammon, J. A.; Yu, Z.; Zhou, Y.; Zhu, Y. Adaptive finite element modeling techniques for the Poisson-Boltzmann equation. *Commun. Comput. Phys.* **2012**, *11*, 179−214.

(21) Holst, M. Adaptive numerical treatment of elliptic systems on manifolds. *Adv. Comput. Math.* **2001**, *15*, 139−191.

(22) Holst, M.; Baker, N.; Wang, F. Adaptive multilevel finite element solution of the Poisson−Boltzmann equation I. Algorithms and examples. *J. Comput. Chem.* **2000**, *21*, 1319−1342.

(23) Baker, N.; Holst, M.; Wang, F. Adaptive multilevel finite element solution of the Poisson−Boltzmann equation II. Refinement at solvent-accessible surfaces in biomolecular systems. *J. Comput. Chem.* **2000**, *21*, 1343−1352.

(24) Li, J.; Xie, D. A new linear Poisson-Boltzmann equation and finite element solver by solution decomposition approach. *Commun. Math. Sci.* **2015**, *13*, 315−325.

(25) Cortis, C. M.; Friesner, R. A. Numerical solution of the Poisson−Boltzmann equation using tetrahedral finite-element meshes. *J. Comput. Chem.* **1997**, *18*, 1591−1608.

(26) Shestakov, A.; Milovich, J.; Noy, A. Solution of the nonlinear Poisson−Boltzmann equation using pseudo-transient continuation and the finite element method. *J. Colloid Interface Sci.* **2002**, *247*, 62−79.

(27) Ji, N.; Liu, T.; Xu, J.; Shen, L. Q.; Lu, B. A finite element solution of lateral periodic Poisson−Boltzmann model for membrane channel proteins. *Int. J. Mol. Sci.* **2018**, *19*, 695.

(28) Owen, S. J. A survey of unstructured mesh generation technology. *International Meshing Roundtable Conference*, 1998; Vol. *239*, p 15.

(29) Si, H. TetGen, a Delaunay-based quality tetrahedral mesh generator. *ACM Trans. Math. Software* **2015**, *41*, 1−36.

(30) Chen, M.; Lu, B. TMSmesh: A robust method for molecular surface mesh generation using a trace technique. *J. Chem. Theory Comput.* **2011**, *7*, 203−212.

(31) Chen, M.; Tu, B.; Lu, B. Triangulated manifold meshing method preserving molecular surface topology. *J. Mol. Gra. Modell.* **2012**, *38*, 411−418.

(32) Liu, T.; Chen, M.; Lu, B. Efficient and qualified mesh generation for Gaussian molecular surface using adaptive partition and piecewise polynomial approximation. *SIAM J. Sci. Comput.* **2018**, *40*, B507−B527.

(33) Liu, T.; Chen, M.; Song, Y.; Li, H.; Lu, B. Quality improvement of surface triangular mesh using a modified Laplacian smoothing approach avoiding intersection. *PLoS One* **2017**, *12*, No. e0184206.

(34) Liu, T.; Bai, S.; Tu, B.; Chen, M.; Lu, B. Membrane-channel protein system mesh construction for finite element simulations. *Comput. Math. Biophys.* **2015**, *3*, 128−139.

(35) Chao, Z.; Gui, S.; Lu, B.; Xie, D. Efficient generation of membrane and solvent tetrahedral meshes for ion channel finite element calculation. *Int. J. Numer. Anal. Model.* **2022**, *19*, 887−906.

(36) Khan, D.; Yan, D.-M.; Gui, S.; Lu, B.; Zhang, X. Molecular surface remeshing with local region refinement. *Int. J. Mol. Sci.* **2018**, *19*, 1383.

(37) Khan, D.; Gui, S.; Cheng, Z. Molecular surface mesh smoothing with subdivision. *Lect. Notes Comput. Sci.* **2024**, *14496*, 236−248.

(38) Chu, C.-C.; Graham, I.; Hou, T.-Y. A new multiscale finite element method for high-contrast elliptic interface problems. *Math. Comput.* **2010**, *79*, 1915−1955.

(39) LeVeque, R. J.; Li, Z. The immersed interface method for elliptic equations with discontinuous coefficients and singular sources. *SIAM J. Numer. Anal.* **1994**, *31*, 1019−1044.

(40) Li, Z. The immersed interface method using a finite element formulation. *Appl. Numer. Math.* **1998**, *27*, 253−267.

(41) Moës, N.; Dolbow, J.; Belytschko, T. A finite element method for crack growth without remeshing. *Int. J. Numer. Methods Eng.* **1999**, *46*, 131−150.

(42) Wu, H.; Xiao, Y. An unfitted hp-interface penalty finite element method for elliptic interface problems. *J. Comput. Math.* **2019**, *37*, 316−339.

(43) Babuška, I. The finite element method for elliptic equations with discontinuous coefficients. *Computing* **1970**, *5*, 207−213.

(44) Hansbo, A.; Hansbo, P. An unfitted finite element method, based on Nitsche's method, for elliptic interface problems. *Comput. Methods Appl. Mech. Eng.* **2002**, *191*, 5537−5552.

(45) Burman, E.; Hansbo, P.; Larson, M. A cut finite element method with boundary value correction. *Math. Comput.* **2017**, *87*, 633−657.

(46) Xiao, Y.; Xu, J.; Wang, F. High-order extended finite element methods for solving interface problems. *Comput. Methods Appl. Mech. Eng.* **2020**, *364*, 112964.

(47) Massjung, R. An unfitted discontinuous Galerkin method applied to elliptic interface problems. *SIAM J. Numer. Anal.* **2012**, *50*, 3134−3162.

(48) Liu, H.; Zhang, L.; Zhang, X.; Zheng, W. Interface-penalty finite element methods for interface problems in H1, H (curl), and H (div). *Comput. Methods Appl. Mech. Eng.* **2020**, *367*, 113137.

(49) Lamm, G. The Poisson−Boltzmann equation. *Rev. Comput. Chem.* **2003**, *19*, 147−365.

(50) Zhang, L.-B. A parallel algorithm for adaptive local refinement of tetrahedral meshes using bisection. *Numer. Math. Theory Methods Appl.* **2009**, *2*, 65−89.

(51) Gui, S.; Khan, D.; Wang, Q.; Yan, D.; Lu, B. Frontiers in biomolecular mesh generation and molecular visualization systems. *Visual Comput. Ind. Biomed. Art* **2018**, *1*, 7−13.

(52) Lee, B.; Richards, F. M. The interpretation of protein structures: estimation of static accessibility. *J. Mol. Biol.* **1971**, *55*, 379−400.

(53) Richards, F. M. Areas, volumes, packing, and protein structure. *Annu. Rev. Biophys. Bioeng.* **1977**, *6*, 151−176.

(54) Duncan, B. S.; Olson, A. J. Shape analysis of molecular surfaces. *Biopolymers* **1993**, *33*, 231−238.

(55) Liu, T.; Chen, M.; Lu, B. Parameterization for molecular Gaussian surface and a comparison study of surface mesh generation. *J. Mol. Model.* **2015**, *21*, 113−114.

(56) Liao, T.; Zhang, Y.; Kekenes-Huskey, P. M.; Cheng, Y.; Michailova, A.; McCulloch, A. D.; Holst, M.; McCammon, J. A. Multi-core CPU or GPU-accelerated Multiscale Modeling for Biomolecular Complexes. *Comput. Math. Biophys.* **2013**, *1*, 164−179.

(57) Yu, Z.; Holst, M. J.; Andrew McCammon, J. High-fidelity geometric modeling for biomedical applications. *Finite Elem. Anal. Des.* **2008**, *44*, 715−723.

(58) Zhang, Y.; Xu, G.; Bajaj, C. Quality meshing of implicit solvation models of biomolecular structures. *Comput. Aided Geomet. Des.* **2006**, *23*, 510−530.

(59) McGann, M.; Almond, H.; Nicholls, A.; Grant, J.; Brown, F. Gaussian docking functions. *Biopolymers* **2003**, *68*, 76−90.

(60) Grant, J.; Gallardo, M.; Pickup, B. A fast method of molecular shape comparison: A simple application of a Gaussian description of molecular shape. *J. Comput. Chem.* **1996**, *17*, 1653−1666.

(61) Weiser, J.; Shenkin, P.; Still, W. Optimization of Gaussian surface calculations and extension to solvent-accessible surface areas. *J. Comput. Chem.* **1999**, *20*, 688−703.

(62) Yu, Z.; Jacobson, M.; Friesner, R. What role do surfaces play in GB models? A new-generation of surface-generalized Born model based on a novel Gaussian surface for biomolecules. *J. Comput. Chem.* **2006**, *27*, 72−89.

(63) Albin, E.; Knikker, R.; Xin, S.; Paschereit, C. O.; d'Angelo, Y. Computational assessment of curvatures and principal directions of implicit surfaces from 3D scalar data. *Math. Methods Curves Surfaces* **2017**, *10521*, 1−22.

(64) Adams, R. A.; Fournier, J. J. *Sobolev Spaces*; Academic Press, 2003.

(65) Arnold, D. N. An interior penalty finite element method with discontinuous elements. *SIAM J. Numer. Anal.* **1982**, *19*, 742−760.

(66) Burman, E.; Ern, A. Continuous interior penalty hp-finite element methods for advection and advection-diffusion equations. *Math. Comput.* **2007**, *76*, 1119−1140.

(67) Cui, T.; Leng, W.; Liu, H.; Zhang, L.; Zheng, W. High-order numerical quadratures in a tetrahedron with an implicitly defined curved interface. *ACM Trans. Math. Software* **2020**, *46*, 1−18.

(68) Zhang, Q.; Gui, S.; Li, H.; Lu, B. Model reduction-based initialization methods for solving the Poisson-Nernst-Plank equations in three-dimensional ion channel simulations. *J. Comput. Phys.* **2020**, *419*, 109627.

(69) Leng, W.; Cui, T.; Lin, D.; Zheng, W.; Zhang, L.; Lu, B. The toolbox PHG and its applications. *Sci. Sin. Inform.* **2016**, *46*, 1442−1464.

(70) Dolinsky, T. J.; Nielsen, J. E.; McCammon, J. A.; Baker, N. A. PDB2PQR: an automated pipeline for the setup of Poisson−Boltzmann electrostatics calculations. *Nucleic Acids Res.* **2004**, *32*, 665−667.

(71) Dou, W.; Chen, M.; Zhou, S. Fast iterative method for local steric Poisson−Boltzmann theories in biomolecular solvation. *Comput. Phys. Commun.* **2023**, *291*, 108808.

(72) Zhou, S.; Wang, Z.; Li, B. Mean-field description of ionic size effects with nonuniform ionic sizes: A numerical approach. *Phys. Rev. E: Stat., Nonlinear, Soft Matter Phys.* **2011**, *84*, 021901.