Journal of Computational and Applied Mathematics 330 (2018) 456–474

Contents lists available at ScienceDirect

Journal of Computational and Applied Mathematics

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

On developing stable finite element methods for pseudo-time simulation of biomolecular electrostatics

Weishan Deng [a,c], Jin Xu [a,*], Shan Zhao [b]

[a] Laboratory of Parallel Software and Computational Science, Institute of Software, Chinese Academy of Sciences, Beijing, 100190, China
[b] Department of Mathematics, University of Alabama, Tuscaloosa, AL 35487, USA
[c] University of Chinese Academy of Sciences, Beijing, 100049, China

ARTICLE INFO

Article history:
Received 24 March 2017
Received in revised form 9 June 2017

Keywords:
Poisson–Boltzmann equation
Regularized Poisson–Boltzmann equation
Pseudo-time approach
Finite element method
Molecular surface
Electrostatic free energy

ABSTRACT

The Poisson–Boltzmann Equation (PBE) is a widely used implicit solvent model for the electrostatic analysis of solvated biomolecules. To address the exponential nonlinearity of the PBE, a pseudo-time approach has been developed in the literature, which completely suppresses the nonlinear instability through an analytic integration in a time splitting framework. This work aims to develop novel Finite Element Methods (FEMs) in this pseudo-time framework for solving the PBE. Two treatments to the singular charge sources are investigated, one directly applies the definition of the delta function in the variational formulation and the other avoids numerical approximation of the delta function by using a regularization formulation. To apply the proposed FEMs for both PBE and regularized PBE in real protein systems, a new tetrahedral mesh generator based on the minimal molecular surface definition is developed. With a body-fitted mesh, the proposed pseudo-time FEM solvers are more accurate than the existing pseudo-time finite difference solvers. Moreover, based on the implicit Euler time integration, the proposed FEMs are unconditionally stable for solvated proteins with source singularities and non-smooth potentials, so that they could be more efficient than the existing pseudo-time discontinuous Galerkin method based on the explicit Euler time stepping. Due to the unconditional stability, the proposed pseudo-time algorithms are free of blow-up or overflow issues, without resorting to any thresholding technique. Numerical experiments of several benchmark examples and free energy calculations of protein systems are conducted to validate the stability, accuracy, and robustness of the proposed PBE solvers.

1. Introduction

The Poisson–Boltzmann (PB) equation is commonly used in biomolecular simulations to describe the electrostatic interactions and ionic density distributions of solvated biomolecules at the equilibrium state [1,2]. By treating the macromolecule and its aqueous surroundings as continuum, the PB equation combines the classical Gauss's law in electrodynamics with the fundamental Boltzmann distribution in statistical thermodynamics. Mathematically, the PB model takes the form of a nonlinear elliptic equation on multiple domains with discontinuous dielectric coefficients across the molecular surface or solute–solvent interface [3,4]. A brief history of PB model can be found in [5]. The PB methodology and applications in biomolecular modeling are summarized briefly in [2,6].

* Corresponding author.
E-mail addresses: weishan2012@iscas.ac.cn (W. Deng), xu_jin@iscas.ac.cn (J. Xu), szhao@ua.edu (S. Zhao).

http://dx.doi.org/10.1016/j.cam.2017.09.004
0377-0427/

The PB equation admits analytic solutions only for a few simple shapes [7], so that for simulating biomolecules with complex geometries, numerical solution is indispensable. Driven by practical needs, tremendous advances have been made in developing PB numerical solvers over the last few decades, giving rise to a variety of fast computational approaches, including finite difference methods [8–13], finite element methods [14–18], boundary element methods [19–23], and discontinuous Galerkin methods [24–26]. Moreover, many numerical algorithms for the PB equation have been incorporated into popular molecular simulation software packages or online web servers, such as DelPhi [8,11], UHBD [27], CHARMM [10], APBS [28], AMBER [12], MIBPB [29], and SDPBS [30]. Even though a great success has been achieved in the aforementioned studies, the numerical solution of the PB equation remains to be challenging, due to the consideration of discontinuous coefficients, complex geometry of protein structures, singular source terms, infinite domain, and strong non-linearity [31].

The non-linearity associated with the PB equation is of exponential form, and is critical when the underlying electrostatic potential is strong. An improper numerical treatment of the PB non-linearity could introduce instability or ill-conditioning in the computation. Two different approaches have been developed in the literature for handling the PB non-linearity. In a usual approach [32], the PB equation is discretized into a nonlinear algebra system, which is then solved by nonlinear algebraic solvers, including nonlinear relaxation methods [10,11], nonlinear conjugate gradient method [9], and inexact Newton method [33].

The other approach is based on the pseudo-time solution of the PB equation [34–36]. In such solutions, a time dependent PB equation is constructed by introducing a pseudo-time derivative. A long time integration is carried out, so that the resulting steady state solution converges to the solution of the original nonlinear boundary value system. Nevertheless, by using an explicit or semi-implicit time integration scheme [34–36], the nonlinear term has to be linearized or evaluated at a previous time, which undermines both stability and efficiency.

Recently, a novel nonlinear treatment in the pseudo-time approach has been developed [37,38] to completely suppress the nonlinear instability. The success lies in an analytic integration of the nonlinear term, after the time-dependent PB equation is split into linear and nonlinear subsystems. Being free of blow-up or overflow problems which may be encountered in the nonlinear algebraic methods [39], this tailored nonlinear treatment finally makes the pseudo-time solution a viable approach for electrostatic analysis of biomolecules [37,38,40].

In particular, both pseudo-time finite difference [37,38,40] and discontinuous Galerkin (DG) [26] methods have been developed in this framework. Using implicit time stepping schemes, the finite difference methods are very efficient for large protein systems, but are quite inaccurate. Based on a body-fitted mesh, the DG variational formulation offers great accuracy and flexibility in handling complex geometries and nonsmooth potentials. However, by integrating with an explicit Euler scheme, the pseudo-time DG method [26] is quite inefficient in steady state simulations.

The objective of this work is to develop stable finite element methods (FEMs) for solving the PB equation. To this end, Nodal-Based Finite Element Method (NBFEM) [41–44] will be introduced for pseudo-time simulation of biomolecular electrostatics, by overcoming the difficulties associated with the existing pseudo-time approaches [26,37,38,40]. In the framework of continuous Galerkin (CG) weak formulations, the implicit Euler time discretization is constructed, giving rise to more stable and efficient pseudo-time FEMs for solving the PB equation. Moreover, rigorous treatment of solute–solvent interface will be adopted by using unstructured tetrahedra meshes, so that the numerical accuracy could be greatly improved, compared with the finite difference results [37,38,40]. Furthermore, significant effort will be devoted to attack other challenging features of the PB equation. In particular, we will investigate and compare different treatments for singular partial charges. A new tetrahedral mesh generator based on the minimal molecular surface [3,4] will be developed, so that the proposed pseudo-time FEMs can be applied to protein structures downloaded from the protein data bank.

It is well known that the singular charge sources of the PB model, in terms of a sum of Dirac delta functions, not only impose great difficulty in numerical analysis [16], but also introduce a large approximation error in 3D mesh based algorithms [31]. The most commonly used technique in the PB literature is using a trilinear approximation to distribute a singular source to the vertices of the cube or element containing the source point [8]. In a FEM variational form, one can apply the definition of the delta function so that a singular source can be evaluated through the trial functions [17]. Being cheap and efficient, these treatments may not guarantee second order accuracy [13], because the potential function changes rapidly near the singular poles. A more elegant way to handle singular charges is using regularization approaches [13,16,18,39,45,46]. In these approaches, the potential function is decomposed into a singular component plus one or two other components. The singular Coulomb potential satisfies a Poisson equation with the same PB singular sources, and can be analytically solved in terms of Green's functions. After removing the singular component, the other potential components satisfy elliptic equations with a higher regularity so that their discretization by general numerical methods becomes more accurate [31]. In the present study, both the trial function method [17] and a regularization approach [39] will be studied in the proposed pseudo-time FEMs.

In the PB model, a molecular surface is needed to distinguish the molecular domain from the solvent domain. The classical molecular surfaces are hard sphere models including Van der Waals (VdW) surface, solvent-accessible surface [47], and solvent-excluded surface [48]. On the other hand, the soft sphere models, such as Gaussian surface [49] and minimal molecular surface (MMS) [3,4], are more suitable for the FEM tetrahedral mesh generation, because these surfaces are free of geometrical singularities and the underlying level set functions allow simpler algebraic calculations in generating volumetric meshes [18,50,51]. The MMS will be employed in this work, which is defined as the unique surface that is of the smallest area and encloses all VdW balls. However, in the previous MMS generation [52], only Cartesian grid type volumetric meshes have been considered for finite difference PB solvers.

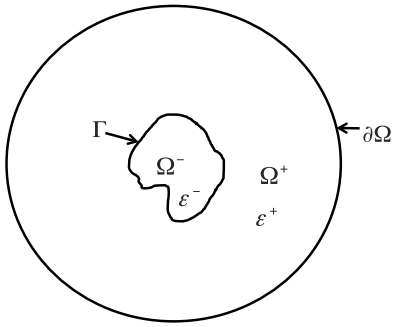

Fig. 1. Computational domain in 3D.

In this paper, a new MMS mesh generator will be developed for the use of FEMs. In particular, we will first employ the Computational Geometry Algorithms Library (CGAL) [53] to construct a surface triangulation of the MMS. Then, a 3D Delaunay tetrahedral mesh can be constructed by using the TetGen [54] for both molecule and solvent domains. By employing these state of the art tools, the generated MMS meshes are of high quality for biomolecular simulations, and ensure the optimal convergence of the FEMs.

The rest of this paper is organized as follows: The details of the proposed FEMs will be discussed in Section 2. Both regularized and original PB equations will be studied. The pseudo-time weak formulations with time splitting will be constructed, and the implicit Euler time discretization and FEM spatial discretization will be established. Numerical validations through benchmark examples with analytic solutions will be considered in Section 3. Stability and accuracy of the proposed pseudo-time FEMs will be examined. In Section 4, the free energy calculations of a one-atom model with reference values will be conducted. The applications of the developed FEMs to some real proteins will also be considered. Finally, this paper ends with a conclusion.

2. Theory and algorithm

2.1. Poisson–Boltzmann equation

In the electrostatic analysis of a solvated biomolecule, the macromolecule is naturally assumed to be immersed in an aqueous solvent environment with mobile ions. Computationally, we consider a large enough sphere containing the solute protein, and denote this sphere as the computational domain Ω. The minimal molecular surface (MMS) as defined in [3,4] is assumed to divide Ω into the inside closed solute domain Ω^- and the outside solvent domain Ω^+ such that $\Omega = \Omega^- \cup \Omega^+$, $\Omega^- \cap \Omega^+ = \emptyset$ and $\Gamma = \partial\Omega^-$. A diagram illustrating the solute and solvent domains is given in Fig. 1.

The Poisson–Boltzmann (PB) equation governing the electrostatic interactions between the charged protein and the mobile ions in the water can be written in the dimensionless form [31,38]

$$\begin{cases} -\nabla\cdot(\epsilon\nabla u) + \kappa^2 \sinh(u) = \rho & in \quad \Omega, \\ u^- = u^+, \quad \epsilon^- \dfrac{\partial u^-}{\partial n} = \epsilon^+ \dfrac{\partial u^+}{\partial n} & on \quad \Gamma, \\ u = g(\vec{x}) & on \quad \partial\Omega, \end{cases} \tag{1}$$

where u is the dimensionless electrostatic potential, g is a boundary data, and ρ is a singular source

$$\rho(\vec{x}) = \frac{4\pi e_c^2}{k_B T} \sum_{j=1}^{N_m} q_j \delta(\vec{x} - \vec{x}_j). \tag{2}$$

Here e_c is the unit of fundamental charge, q_j is the partial charge value on the jth atom of the solute located at $\vec{x}_j = (x_j, y_j, z_j)$, k_B is the Boltzmann constant, and T is the temperature. With the Dirac delta functions, the source term ρ is singular at the centers of a total of N_m atoms.

In the PB equation (1), the dielectric coefficient ϵ is assumed to be a piece-wise constant such that $\epsilon = \epsilon^-$ for $\vec{x} \in \Omega^-$ and $\epsilon = \epsilon^+$ for $\vec{x} \in \Omega^+$. The water domain usually has a much higher dielectric value than the protein. The modified Debye–Huckel parameter κ is also assumed to be a piece-wise constant with a vanishing value inside Ω^-, i.e., $\kappa^- = 0$. In the water, we have [7]

$$(\kappa^+)^2 = \left(\frac{2N_A e_c^2}{1000 k_B T}\right) I_s, \quad \text{for } \vec{x} \in \Omega^+, \tag{3}$$

where N_A is the Avogadro's number and I_s is the ionic strength in the unit of mole. Numerically, when $T = 298$ K, we have $\frac{e_c^2}{k_B T} = 332.06364/0.592183$ Å, $(\kappa^+)^2 = 8.486902807$ Å$^{-2} I_s$ in Ω^+ [7]. The PB equation varies on the choice of units, and in the present setting we utilized the dimensionless form presented in [33]. The dimensionless electrostatic potential u can be conveniently converted to the electrostatic potential of units kcal/mol/e_c by multiplying the constant 0.592183 subject to room temperature ($T = 298$ K) [7].

Two interface jump conditions are defined across Γ in Eq. (1), where superscript '$-$' means the limiting value approaching Γ from the inside and '$+$' from the outside for u and its derivative, and $\vec{n}$ is the outward normal direction of the solute–solvent boundary Γ. On the outer boundary $\partial\Omega$, a Dirichlet boundary condition can be assumed

$$u(\vec{x}) = g(\vec{x}) = \frac{e_c^2}{k_B T} \sum_{i=1}^{N_m} \frac{q_i}{\epsilon^+ |\vec{x} - \vec{x}_i|} \mathrm{e}^{-\kappa^+ |\vec{x} - \vec{x}_i| / \sqrt{\epsilon^+}}. \tag{4}$$

Eq. (4) is actually a linear superposition of damped Coulomb's law for a series of N_m partial charges q_i at positions $\vec{x}_i$. It is obvious that $\lim_{|\vec{x}| \to \infty} u(\vec{x}) = 0$, which means the physical condition is satisfied at the infinity. When $\partial\Omega$ is sufficiently distanced from the macromolecule, (4) provides a fairly accurate approximation to the solution, so that our numerical computation can be conducted on a finite domain Ω.

2.2. Pseudo-time approach for the PB equation

As an indirect method, the pseudo-time approach solves the PB equation by introducing a pseudo-time derivative [34–36]

$$\frac{\partial u(\vec{x}, t)}{\partial t} = \nabla \cdot (\epsilon \nabla u(\vec{x}, t)) - \kappa^2 \sinh(u(\vec{x}, t)) + \rho(\vec{x}), \quad \text{in } \Omega, \tag{5}$$

with the same interface condition in Eq. (1) and boundary condition (4). By using a trivial initial value $u(\vec{x}, 0) = 0$, one numerically integrates (5) for a sufficiently long time period to stable state. The solution to the original nonlinear PB equation is essentially recovered by the steady state solution of the pseudo-time dependent process (5). We note that the sign on the right-hand side of Eq. (5) is the reverse of that of Eq. (1) to ensure the numerical stability [37]. Care has to be exercised in the time discretization of (5). The explicit or semi-implicit time stepping schemes are known to be inefficient [34–36], while an implicit treatment of the nonlinear term $\sinh(\cdot)$ is non-trivial.

2.2.1. Time splitting scheme for pseudo-time PBE

The pseudo-time method finally becomes a viable approach for solving the PB equation, when an analytic treatment for the PB non-linearity is developed in [37,38]. Consider a uniform grid in time with the time increment Δt. We denote $u^n := u^n(\vec{x}) = u(\vec{x}, t_n)$. The time stepping of (5) over the time interval $[t_n, t_{n+1}]$ can be carried out in two stages in a first order time splitting method [37,38]

$$\frac{\partial w}{\partial t} = -\kappa^2 \sinh(w), \quad \text{with } w^n = u^n, \quad t \in [t_n, t_{n+1}], \tag{6}$$

$$\frac{\partial v}{\partial t} = \nabla \cdot (\epsilon \nabla v) + \rho, \quad \text{with } v^n = w^{n+1}, \quad t \in [t_n, t_{n+1}]. \tag{7}$$

We then set $u^{n+1} = v^{n+1}$. Here the time independent interface condition (1) and boundary condition (4) should be held for both subsystems. However, since the first Eq. (6) is free of spatial derivatives, one does not need to enforce these conditions for its solution. In the present work, we will ensure the solution of the second Eq. (7) satisfying the required interface and boundary conditions. In this manner, the overall numerical solution u^{n+1} will also satisfy these conditions, and it differs from the direct solution of (5) by an error on the order one, i.e., $O(\Delta t)$. A second order time splitting has also been developed in [37,38], by dividing the process into three stages with the nonlinear sub-system (6) being repeated twice.

2.2.2. Analytic solution for the nonlinear term

The solution of the nonlinear sub-system (6) is conducted analytically [37,38]

$$w^{n+1}(\vec{x}) = \ln\left(\frac{\cosh(\frac{1}{2}\kappa^2 \Delta t) + \exp(-w^n(\vec{x})) \sinh(\frac{1}{2}\kappa^2 \Delta t)}{\exp(-w^n(\vec{x})) \cosh(\frac{1}{2}\kappa^2 \Delta t) + \sinh(\frac{1}{2}\kappa^2 \Delta t)} \right). \tag{8}$$

Note that (8) is valid for all $\vec{x} \in \Omega$. In particular, we have $\kappa^- = 0$ for $\vec{x} \in \Omega^-$ so that $w^{n+1}(\vec{x}) = w^n(\vec{x})$ in the solute region. It is known [31] that the exponential nonlinear term of the PB equation could grow without bound when the potential value is large. Without an implicit treatment of $\sinh(\cdot)$, an extremely small Δt has to be used to ensure stability in early pseudo-time methods [34–36]. Similarly, a blow-up issue has also been noticed in the inexact Newton's method [39]. A thresholding procedure has to be applied to avoid large potential values in the minimum search process, to prevent numerical overflow [39]. We note that these instability or blow-up issues are completely bypassed in the present analytic treatment (8).

2.2.3. Weak formulation for the second order derivative and source terms

In the framework of the first order time splitting scheme [37,38], spatial discretization needs merely for a linear diffusion equation

$$\frac{\partial u}{\partial t} = \nabla \cdot (\epsilon \nabla u) + \rho, \quad \text{in} \quad \Omega, \tag{9}$$

subject to the same interface condition (1) and boundary condition (4). Note that we have abused the notation by taking the unknown function as $u(x, t)$ in (9) for simplicity. Otherwise, (9) is just (7).

A nodal-based finite element method (FEM) is developed in this subsection for solving the Time-Dependent Poisson–Boltzmann (TDPB) equation. The resulting TDPB-FEM will be more accurate than the pseudo-time finite difference methods [37,38,40], by using a tetrahedral mesh body-fitted to the minimal molecular surface (MMS). On the other hand, as an implicit scheme, the TDPB-FEM will be more stable and efficient than the explicit pseudo-time discontinuous Galerkin (DG) method [26].

Consider an approximation of domain Ω by a tetrahedral mesh Ω^h, which is composed of K non-overlapping tetrahedron

$$\Omega \approx \Omega^h = \bigcup_{k=1}^{K} D_k. \tag{10}$$

Within each of the elements, D_k, we assume that the solution, u_h is represented as

$$u(\vec{x}, t) \approx u_h(\vec{x}, t) = \sum_{i=1}^{N} u_i(t) L_i(\vec{x}) \in V_h = \bigoplus_{k=1}^{k} P_n(D_k) \tag{11}$$

where $u_i(t)$ is the solution at N grid points, $\vec{x} \in D_k$. $P_n(D_k)$ is the n-th order polynomial space on D_k, and union of the polynomial spaces on all elements. Here $L_i(\vec{x})$ is the three-dimensional Lagrange polynomial associated with the N grid nodes, and it belongs to the space of n-th order three dimensional polynomials, $P_n(D_k)$. The nodes are distributed at vertices, along edges, on faces as well as in the interior of the elements. Given the location of a node $\vec{x}_j$, the efficient and accurate construction of the Lagrange polynomial and other details about the nodal-based FEM can be found in [43].

For a test function $\forall \phi_i \in P_n^3(\Omega^h)$, the weak formulation of (9) reads

$$\int_\Omega \frac{\partial u}{\partial t} \phi_i dv = \int_\Omega \nabla \cdot (\epsilon \nabla u) \phi_i dv + \int_\Omega \rho \phi_i dv. \tag{12}$$

Denote $\vec{n}$ as the normal direction of Γ, pointing from Ω^- to Ω^+, we have

$$\begin{aligned} \int_\Omega \frac{\partial u}{\partial t} \phi_i dv = &\int_\Gamma \vec{n} \cdot (\epsilon^- \nabla u^-) \phi_i ds - \int_{\Omega^-} \epsilon \nabla u \cdot \nabla \phi_i dv + \int_\Omega \rho \phi_i dv \\ &- \int_\Gamma \vec{n} \cdot (\epsilon^+ \nabla u^+) \phi_i ds - \int_{\Omega^+} \epsilon \nabla u \cdot \nabla \phi_i dv, \end{aligned} \tag{13}$$

subject to the boundary condition (4) on $\partial\Omega$. On the interface Γ between Ω^- and Ω^+, the jump condition in Eq. (1) suggests

$$\int_\Gamma (\vec{n} \cdot (\epsilon^- \nabla u^- - \epsilon^+ \nabla u^+)) \phi_i ds = 0. \tag{14}$$

Thus, we have simply

$$\int_\Omega \frac{\partial u}{\partial t} \phi_i dv + \int_\Omega \epsilon \nabla u \cdot \nabla \phi_i dv = \int_\Omega \rho \phi_i dv \tag{15}$$

subject to the boundary condition (4) on $\partial\Omega$.

The implicit Euler scheme is employed for the time discretization

$$\int_\Omega \frac{u^{n+1} - u^n}{\Delta t} \phi_i dv + \int_\Omega \epsilon \nabla \phi_i \cdot \nabla u^{n+1} dv = \int_\Omega \rho \phi_i dv. \tag{16}$$

$$\frac{1}{\Delta t} \int_\Omega u^{n+1} \phi_i dv + \int_\Omega \epsilon \nabla \phi_i \cdot \nabla u^{n+1} dv = \frac{1}{\Delta t} \int_\Omega u^n \phi_i dv + \int_\Omega \rho \phi_i dv. \tag{17}$$

By using the basis expansion (11), the stiffness and mass matrices of the TDPB-FEM can be calculated as

$$A_{ij} = \int_\Omega \epsilon \nabla \phi_j \cdot \nabla \phi_i dv, \qquad M_{ij} = \int_\Omega \phi_j \cdot \phi_i dv. \tag{18}$$

Following [17], the singular source ρ in (17) will be discretized based on the definition of the delta function in the TDPB-FEM

$$F_i = \int_\Omega \rho \phi_i dv = \int_\Omega \frac{4\pi e_c^2}{k_B T} \sum_{j=1}^{N_m} q_j \delta(\vec{x} - \vec{x}_j) \phi_i(\vec{x}) dv = \frac{4\pi e_c^2}{k_B T} \sum_{j=1}^{N_m} q_j \phi_i(\vec{x}_j). \tag{19}$$

Assembling the matrix coefficients and source terms for $i, j = 1, \ldots, N$, where N is the total unknown nodal values, we have the variational formulation of the TDPB-FEM

$$(\Delta t A + M)u^{n+1} = Mu^n + \Delta t F. \tag{20}$$

In this paper, only piece-wise linear basis will be considered. Consequently, the nodal-based FEM is simply the usual linear Lagrange FEM. The boundary condition (4) will be implemented as in the usual nodal based FEMs [43].

2.3. Pseudo-time approach for the regularized PB equation

In the TDPB-FEM presented above, the singular source ρ is calculated analytically in (19), according to the definition of the delta function. Such an approximation seems to be better than the usual treatment in the finite difference methods [8], where a trilinear approximation is carried out to distribute a point charge to the vertices of the element containing this charge. However, since the trial function $\phi_i(x)$ is usually a low order polynomial that has nonzero values over a few neighboring tetrahedron, the effect of (19) is actually similar to that of the finite difference treatment, i.e., each partial charge is just numerically accounted for by using a few coefficients of F. Therefore, in view of the rapid change of the potential function induced by the delta function near the singular pole, the direct singular source approximations often undermines the overall spatial accuracy [13]. To overcome this difficulty, we will introduce another FEM based on a regularization approach for treating the singular charges [39] in this subsection. Our new FEM will solve a time-dependent regularized Poisson–Boltzmann (TDRPB) equation, and is thus termed as TDRPB-FEM.

2.3.1. Regularization scheme for singular sources

Like other regularization approaches [13,18,46], the regularization method introduced by Xie [39] decomposes the electrostatic potential u into three parts $u = u^s + u^h + u^r$, where u^s, u^h, and u^r are singular, harmonic and regular components, respectively. These three components satisfy different partial differential equations (PDEs) so that the sum u still satisfies the original PB equation (1).

The singular potential satisfies a Poisson's equation on the entire space R^3,

$$\begin{cases} -\epsilon^- \Delta u^s = \dfrac{4\pi e_c^2}{k_B T} \displaystyle\sum_{j=1}^{N_m} q_j \delta(\vec{x} - \vec{x}_j), & in \quad R^3, \\ u^s(\vec{x}) = 0, & as \quad |\vec{x}| \to \infty. \end{cases} \tag{21}$$

The solution is simply Green's function $u^s(\vec{x}) = G(\vec{x})$ [39], where

$$G(\vec{x}) = \frac{e_c^2}{k_B T \epsilon^-} \sum_{j=1}^{N_m} \frac{q_j}{|\vec{x} - \vec{x}_j|}. \tag{22}$$

The harmonic potential satisfies a Poisson's equation with two interface conditions [39] ,

$$\begin{cases} -\epsilon^- \Delta u^h = 0, & in \quad \Omega, \\ (u^h)^+ = (u^h)^- & on \quad \Gamma, \\ \epsilon^+ \left(\dfrac{\partial u^h}{\partial n}\right)^+ = \epsilon^- \left(\dfrac{\partial u^h}{\partial n}\right)^- + (\epsilon^- - \epsilon^+) \dfrac{\partial G}{\partial n} & on \quad \Gamma, \\ u^h = g(\vec{x}) - G(\vec{x}), & on \quad \partial\Omega, \end{cases} \tag{23}$$

where the superscripts denote the limiting values on two sides of the interface. The regular potential is governed by a regularized Poisson–Boltzmann (RPB) equation [39]

$$\begin{cases} -\nabla \cdot (\epsilon \nabla u^r) + \kappa^2 \sinh(u^s + u^h + u^r) = 0, & in \quad \Omega, \\ (u^r)^+ = (u^r)^-, \quad \epsilon^+ \left(\dfrac{\partial u^r}{\partial n}\right)^+ = \epsilon^- \left(\dfrac{\partial u^r}{\partial n}\right)^- & on \quad \Gamma, \\ u^r = 0, & on \quad \partial\Omega, \end{cases} \tag{24}$$

with homogeneous interface and boundary conditions. Both harmonic and regular potentials need to be solved numerically.

2.3.2. Weak formulation for harmonic potential

In the present study, a FEM similar to [39] will be constructed to solve the harmonic component u^h. Since ϵ^- is a constant in (23), one actually solves an elliptic interface with $\Delta u^h = 0$. Testing by ϕ_i, we have

$$\int_\Omega \nabla \cdot (\nabla u^h) \phi_i dv = 0 \quad \text{in } \Omega. \tag{25}$$

We will consider the solute domain Ω^- and solvent domain Ω^+ separately. In Ω^-, we first multiply (25) by ϵ^-. Then the integration by parts gives rise to

$$\int_{\Omega^-} \epsilon^- \nabla\phi_i \cdot \nabla u^h dv = \int_\Gamma \vec{n} \cdot (\epsilon^- \nabla u^h \phi_i) ds. \tag{26}$$

Similarly, we multiply (25) by ϵ^+ in Ω^+,

$$\int_{\Omega^+} \epsilon^+ \nabla\phi_i \cdot \nabla u^h dv = \int_\Gamma -\vec{n} \cdot (\epsilon^+ \nabla u^h \phi_i) ds. \tag{27}$$

Note that the flux term on $\partial\Omega$ is dropped in (27), because the Dirichlet boundary condition $u^h = g(\vec{x}) - G(\vec{x})$ will be enforced on $\partial\Omega$ instead. By adding (26) with (27) and applying the jump condition of (23), one attains

$$\int_\Omega \epsilon \nabla\phi_i \cdot \nabla u^h dv = \int_\Gamma (\epsilon^+ - \epsilon^-) \frac{\partial G}{\partial n} \phi_i ds. \tag{28}$$

The gradient of Green's function is analytically available [39], so that the surface integral in (28) can be calculated accurately. Denote the calculated right-hand side coefficient as F_i^h. The discretized equation becomes $Au^h = F^h$, where the matrix A is the same as that in the TDPB-FEM.

2.3.3. Weak formulation for regular potential

A new FEM is proposed to solve the regular component u^r in our pseudo-time framework [37,38]. It is noted that the RPB equation (24) is free of sources, but it still retains the same non-linearity term. To fully control the nonlinear instability, a time-dependent regularized Poisson–Boltzmann (TDRPB) equation is introduced

$$\frac{\partial u^r}{\partial t} = \nabla \cdot (\epsilon \nabla u^r) - \kappa^2 \sinh(u^s + u^h + u^r), \quad \text{in } \Omega, \tag{29}$$

with $u^r(\vec{x}, t)$ satisfying the same interface and boundary conditions given in (24). Denoting $u^r(\vec{x}, t_n)$ as $(u^r)^n$, a first order time splitting scheme is similarly employed

$$\frac{\partial w}{\partial t} = -\kappa^2 \sinh(u^s + u^h + w), \quad \text{with } w^n = (u^r)^n, \quad t \in [t_n, t_{n+1}], \tag{30}$$

$$\frac{\partial u}{\partial t} = \nabla \cdot (\epsilon \nabla u), \quad \text{with } u^n = w^{n+1}, \quad t \in [t_n, t_{n+1}]. \tag{31}$$

We then set $(u^r)^{n+1} = u^{n+1}$. Theoretically, both w and u in (30) and (31) should satisfy the interface and boundary conditions for u^r in (24). Computationally, the nonlinear subsystem (30) is analytically integrated without enforcing these conditions

$$w^{n+1}(\vec{x}) = \ln\left(\frac{\cosh(\frac{\kappa^2\Delta t}{2}) + \exp(-u^s(\vec{x}) - u^h(\vec{x}) - w^n(\vec{x}))\sinh(\frac{\kappa^2\Delta t}{2})}{\exp(-u^s(\vec{x}) - u^h(\vec{x}) - w^n(\vec{x}))\cosh(\frac{\kappa^2\Delta t}{2}) + \sinh(\frac{\kappa^2\Delta t}{2})}\right) - u^s(\vec{x}) - u^h(\vec{x}), \quad \text{for } \vec{x} \in \Omega. \tag{32}$$

Note that in (32) $w^{n+1} = w^n$ in Ω^- since $\kappa^- = 0$. One then numerically solves the source-free heat Eq. (31) for u with the interface and boundary conditions of (24). This guarantees the overall numerical solution u^r to satisfy these conditions asymptotically, i.e., on the order of $O(\Delta t)$.

The weak formulation of the heat Eq. (31) can be derived similarly. With vanishing boundary and interface data, we have

$$\int_\Omega \frac{\partial u}{\partial t} \phi_i dv + \int_\Omega \epsilon \nabla\phi_i \cdot \nabla u dv = 0. \tag{33}$$

By using the implicit Euler scheme for time discretization, one attains

$$\frac{1}{\Delta t} \int_\Omega u^{n+1} \phi_i dv + \int_\Omega \epsilon \nabla\phi_i \cdot \nabla u^{n+1} dv = \frac{1}{\Delta t} \int_\Omega u^n \phi_i dv. \tag{34}$$

The full discretization is thus given as $(\Delta t A + M)u^{n+1} = Mu^n$, where A and M are defined as in (20). Being free of singular sources, this equation can be evaluated accurately in the computation.

2.4. Triangle and tetrahedral mesh generation

A new tetrahedral mesh generator is developed in this work based on the minimal molecular surface (MMS) [3,4]. The proposed procedure amounts to three major steps, i.e., prepare a protein structure, construct a triangular mesh representing the MMS, and generate a tetrahedral mesh for the entire computational domain. A brief description of these three steps is given below.

The discrete atomic structure of a protein is prepared as in the references [29,38]. For a given protein, we first download the PDB file from the protein data bank (http://www.rcsb.org/). Then the PDB2PQR software (http://www.Poissonboltzmann.

org/) is employed for preparing the molecular structure. This can be done either through the PDB2PQR online server or through downloading and executing a Python package. In particular, the CHARMM22 force field is employed to derive partial charges at atomic sites and atomic Van der Waals radii in angstroms. In the obtained all-atom description, one then knows the center, radius, and partial charge of every atom.

To numerically define the MMS, the fast alternating direction implicit (ADI) algorithm introduced in [52] is employed to solve the 3D mean curvature flow PDE with atom constraints. This defines a discrete level set function $S(x, y, z)$, whose grid values are determined numerically over a Cartesian uniform mesh. For the protein and solvent domain, we have $S(x, y, z)$ greater and less than the zero, respectively. With a little processing, the function $S(x, y, z)$ can be utilized to determine the dielectric profile over the volumetric mesh for solving the PB equation by the finite difference method [36]. However, no package has ever been developed for generating a triangular mesh representing the MMS. In the previous MMS applications, the isosurface generation of the zero level set $S(x, y, z) = 0$ is always through some visualization software. Even though there may be a triangular mesh underlying the displayed MMS, it is inconvenient to output such a mesh for numerical usage. Moreover, such a mesh is for visualization purpose, and thus may not suit our needs for FEM computations.

In this work, we have developed a C++ package to generate the triangular mesh of the MMS. Our package reads the centers and radii of all atoms of the protein, and computes $S(x, y, z)$ using the ADI algorithm. Then, the 3D surface mesh generation package [53] of the Computational Geometry Algorithms Library (CGAL) (http://www.cgal.org/) is employed for isosurface generation. The constructed MMS surface is saved in the Object File Format (OFF). The minimal degree for all angles of mesh facets is set to be 30°. A fine enough grid resolution is employed in both the solution of mean curvature flow equation and setting upper bounds for radius and distance in the CGAL subroutine. This gives rise to a high quality triangular mesh representing the MMS, which is smooth and closed.

Finally, a 3D Delaunay tetrahedral mesh is generated based on the MMS triangular mesh, by calling the TetGen package [54] (http://wias-berlin.de/software/tetgen/). A large enough sphere is constructed and triangularized to bound the protein. The TetGen is executed by reading both the MMS and spherical boundary. By enforcing the boundary conformity, the triangulation of the MMS remains to be the same in this process. We also mark the molecule and solvent domains by assigning different region attributes. In our computations, the build-in quality constraints of the TetGen are utilized to refine the mesh and improve the quality of the tetrahedral mesh, and the main controlling parameter is the maximum volume of a tetrahedron V_{max}. It is believed that based on such a high quality tetrahedral mesh, the optimal convergence rates of the proposed FEMs could be preserved in dealing with the geometrically complicated protein systems.

In order to simulate large and complex protein, parallel computing is necessary. In the pseudo-time FEMs, one needs to solve a large linear system at each time step. For this purpose, a parallel conjugate gradient method has been implemented with Message Passing Interface (MPI) library for parallelization. The mesh generated from the TetGen is first partitioned by the METIS software, which could provide a optimal partition according to some graph theory. Then each processor saves variables in each sub-domain, and communicates with other processors to complete the computation. Up to thousands of processors could be utilized in this parallel algebraic solver, and some results will be shown for real protein later.

2.5. Free energy calculations

The solvation energy or free energy of solvation is the energy released when the macromolecule or solute in free space is dissolved in solvent [37]. In the present content, we will only focus on the polar part of the solvation and compute the electrostatic free energy. Physically, the free energy is defined as the difference between the total electrostatic energy of the protein in the vacuum and in the solvent [29]

$$\Delta G = G - G_0 = \frac{k_B T}{2e_c} \int_\Omega \sum_{j=1}^{N_m} q_j \delta(\vec{x} - \vec{x}_j)(u - u_0) dv \tag{35}$$

where u and u_0 is the electrostatic potential, respectively, in the solvent and in the vacuum.

The calculations of the electrostatic free energy in the regularized and non-regularized approaches are different. In the non-regularized approach, besides solving the PB equation for u, we need to solve a Poisson equation in the vacuum for u_0, in which the same source ρ is assumed and we take $\epsilon^+ = 1$ in vacuum. Without introducing a pseudo-time, this linear Poisson equation can be simply solved by the continuous FEM method. The numerical details are neglected here. After solving the PB and Poisson equations, we know the nodal values of u and u_0 on all elements. Applying the definition of the delta function, we have

$$\begin{aligned} \Delta G &= \frac{k_B T}{2e_c} \int_\Omega \sum_{j=1}^{N_m} q_j \delta(\vec{x} - \vec{x}_j)(u(\vec{x}) - u_0(\vec{x})) dv \\ &= \frac{k_B T}{2e_c} \sum_{j=1}^{N_m} q_j (u(\vec{x}_j) - u_0(\vec{x}_j)). \end{aligned} \tag{36}$$

Numerically, a trilinear interpolation is carried out for every singular charge. In particular, for each charge q_j, we find the tetrahedral element containing the charge point $\vec{x}_j$. Then, four nodal values of u on this element are employed to interpolate

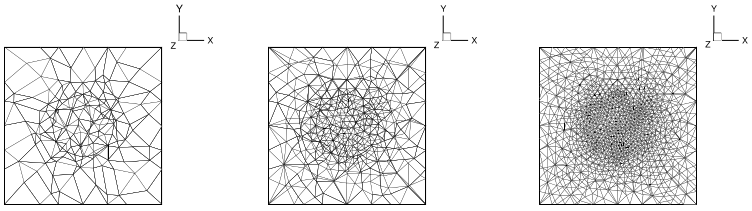

Fig. 2. Cross-section views of the three nested tetrahedral meshes used for a single charge ball.

the $u(\vec{x}_j)$ value. It is commonly believed [31] that the errors of u and u_0 near the singular charges will somehow be canceled out in (36), because the same mesh and charge interpolation are used.

For the regularized approach, one does not need to solve a Poisson equation in vacuum if $\epsilon^- = 1$. In fact, the singular component u^s simply gives us u_0. Thus, the electrostatic free energy can be simply calculated as

$$\begin{aligned} \Delta G &= \frac{k_B T}{2e_c} \int_\Omega \sum_{j=1}^{N_m} q_j \delta(\vec{x} - \vec{x}_j)(u^h(\vec{x}) + u^r(\vec{x}))dv \\ &= \frac{k_B T}{2e_c} \sum_{j=1}^{N_m} q_j (u^h(\vec{x}_j) + u^r(\vec{x}_j)) \end{aligned} \tag{37}$$

where u^h and u^r are, respectively, harmonic and regular component. The same trilinear interpolation can be conducted for calculating $u^h(\vec{x}_j)$ and $u^r(\vec{x}_j)$ values. Note that without the singular component, u^h and u^r are finite and continuous near the singular charges.

3. Benchmarks

In this section, we will consider two examples with analytic solutions to verify the stability and accuracy of the proposed methods using FEM. In order to make sure that the steady state is reached, we usually carry out the time integration until $t \geq T_s$, where T_s is a large enough stopping time. We will numerically explore a suitable value of T_s in our computations. In all computations, the conjugate gradient method is employed to solve all linear systems, and numerical errors in a suitable norm will be reported. All simulations are compiled with mpich-3.2/bin/mpicxx and run on the clusters in the Laboratory of Parallel Software and Computational Science at Institute of Software, Chinese Academy of Science.

3.1. A single charge ball with analytic solution

First an analytic solution on a nonlinear Born ball model [39] is used to benchmark the new methods. Consider the 3D nonlinear PB equation in the dimensionless form given in Eq. (1), where ρ has following forms

$$\rho = \begin{cases} \dfrac{4\pi e_c^2}{k_B T} \delta(\vec{x}) & |\vec{x}| \leq R \\ \kappa^2 \sinh\left(\dfrac{e_c^2}{k_B T \epsilon^+ |\vec{x}|}\right) & |\vec{x}| > R. \end{cases} \tag{38}$$

The analytic solution of Eq. (1) is following

$$u = \begin{cases} \dfrac{e_c^2}{k_B T R}\left(\dfrac{1}{\epsilon^+} - \dfrac{1}{\epsilon^-}\right) + \dfrac{e_c^2}{k_B T \epsilon^- |\vec{x}|} & |\vec{x}| \leq R \\ \dfrac{e_c^2}{k_B T \epsilon^+ |\vec{x}|} & |\vec{x}| > R. \end{cases} \tag{39}$$

The model parameters for this single charge ball are chosen as: $R = 1$ Å, $\kappa^2 = 0.8486902807$ Å^{-2}, $\frac{4\pi e_c^2}{k_B T} = \frac{4.0 * \pi * 332.06364}{0.5921830} = 7046.52914351$ Å, $\epsilon^- = 1$, and $\epsilon^+ = 80$. Three grid levels are considered for a domain $\Omega = [-2.5, 2.5] \times [-2.5, 2.5] \times [-2.5, 2.5]$. The number of elements is 1169, 9352, and 74 816, respectively, for grid level 1, 2, and 3. A cross-section view of these meshes is displayed in Fig. 2.

3.1.1. Non-regularized method

We first consider the TDPB-FEM. For a given mesh level and Δt, denote the numerical solution as u_h. The relative error $L_{2,re}$ is defined as

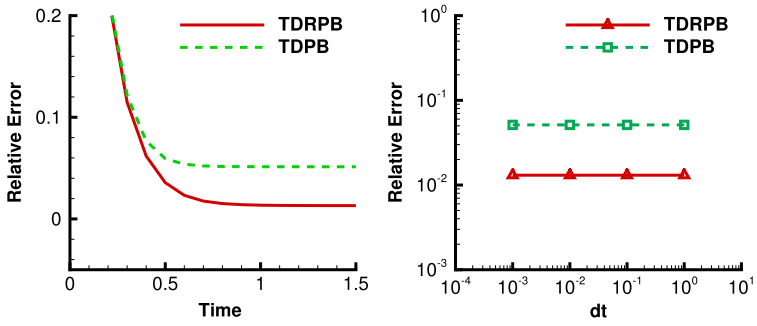

Fig. 3. Time history of TDPB and TDRPB (left) and relative errors for different time steps (right) for a single charge ball.

Table 1
Spatial convergence of the TDPB and TDRPB methods for a single charge ball.

Mesh Data			TDPB	TDRPB
Grid	#Vertices	#Tetrahedra	$L_{2,re}$	$L_{2,re}$
Level 1	238	1,169	2.23e−1	2.53e−1
Level 2	1,698	9,352	6.21e−2	4.58e−2
Level 3	12,963	74,816	5.13e−2	1.31e−2

$$L_{2,re} = \sqrt{\frac{\int_\Omega |u_h(\vec{x}) - u(\vec{x})|^2 dv}{\int_\Omega |u(\vec{x})|^2 dv}}. \tag{40}$$

The TDPB-FEM is found to be stable for all Δt values. Based on the mesh level 3, the $L_{2,re}$ errors of the TDPB method by using different Δt are depicted in the right figure of Fig. 3. Since the error does not depend on Δt for this simple example, we set $\Delta t = 0.1$ for other tests. In the left figure of Fig. 3, the time history of the TDPB method is shown, and the steady state is quickly achieved. By taking $\Delta t = 0.1$, the spatial relative errors $L_{2,re}$ of the TDPB method are listed in Table 1. It can be seen that the relative error $L_{2,re}$ has a worrisome convergence when mesh is refined. The worrisome convergent behavior is because the solution u blows up at the origin due to the charge singularity.

3.1.2. Regularized method

In the TDRPB-FEM, the solution is decomposed into three components $u = u^s + u^h + u^r$. The numerical difficulty of the TDPB method can be simply avoided, because the singular component u^s is not numerically computed in the TDRPB method. Instead, it is analytically given by Green's function. Thus, in calculating the numerical errors, we can exclude the singular component and just access the other two components.

For the present problem with a single charge ball, the analytic solution can be split as a sum of Green's function G and $\bar{u}$, i.e., $u = \bar{u} + G$ where $\bar{u}$ and G have the following forms

$$\bar{u} = \begin{cases} \dfrac{e_c^2}{k_B T R}\left(\dfrac{1}{\epsilon^+} - \dfrac{1}{\epsilon^-}\right) & |\vec{x}| \le R \\ \dfrac{e_c^2}{k_B T |\vec{x}|}\left(\dfrac{1}{\epsilon^+} - \dfrac{1}{\epsilon^-}\right) & |\vec{x}| > R \end{cases} \tag{41}$$

$$G = \frac{e_c^2}{k_B T \epsilon^- |\vec{x}|} \quad in \quad \Omega. \tag{42}$$

In our TDRPB approach, u^s is G given in Eq. (42), while u^h and u^r are computed numerically. In particular, u^h is the solution of the linear interface problem (23), in which $\frac{\partial G}{\partial n} = -\frac{e_c^2}{k_B T \epsilon^- |\vec{x}|^2}$ *on* Γ, and $u^h = \frac{e_c^2}{k_B T |\vec{x}|}\left(\frac{1}{\epsilon^+} - \frac{1}{\epsilon^-}\right)$ *on* $\partial\Omega$. Without introducing a pseudo-time, this linear interface Poisson equation can be simply solved by the continuous FEM. And u^r is the solution of the nonlinear interface problem given in Eq. (24), which is rewritten as the following

$$\begin{cases} -\epsilon^- \Delta u^r = 0 & in \quad \Omega^-, \\ -\epsilon^+ \Delta u^r + \kappa^2 \sinh(u^s + u^h + u^r) = \kappa^2 \sinh\left(\dfrac{e_c^2}{k_B T \epsilon^+ |\vec{x}|}\right) & in \quad \Omega^+, \\ (u^r)^- = (u^r)^+, \quad \epsilon^- \dfrac{\partial (u^r)^-}{\partial n} = \epsilon^+ \dfrac{\partial (u^r)^+}{\partial n} & on \quad \Gamma, \\ u^r = 0 & on \quad \partial\Omega. \end{cases} \tag{43}$$

And this nonlinear interface problem can be solved by a pseudo-time FEM.

Without the singular component, the relative error $L_{2,re}$ is calculated as

$$L_{2,re} = \sqrt{\frac{\int_\Omega |u^h(\vec{x}) + u^r(\vec{x}) - \bar{u}(\vec{x})|^2 dv}{\int_\Omega |u(\vec{x})|^2 dv}}. \tag{44}$$

Based on the mesh level 3, the relative errors of the TDRPB-FEM are plotted against different Δt in the right figure of Fig. 3. The TDRPB method is stable for all Δt values. As shown in the left figure of Fig. 3, the steady state is obviously reached by the TDRPB method. By using $\Delta t = 0.1$, the spatial errors of the TDRPB method are reported in Table 1. It can also be seen that the proposed TDRPB method converges in a second order manner in terms of the relative error $L_{2,re}$.

3.1.3. Comparison of regularized and non-regularized methods

The consideration of the singular component $u^s = G$ makes a difference in the TDPB and TDRPB methods. Without approximating it numerically, the TDRPB method achieves a fast convergence in space, while the spatial convergence of the TDPB method is ruined by the singular charge. This justifies the development of various regularized methods for solving the PB equation [13,16,18,39,45,46]. Moreover, we note that the proposed pseudo-time approach works well in both TDPB and TDRPB methods. The steady state solutions are reached quickly in both cases, and a large Δt is permissible, due to the unconditional stability.

3.2. On a large ball with multiple charges

Next, an analytic solution for multiple charges introduced in [55] is used to validate the proposed TDPB and TDRPB methods further. A protein (PDB ID 2LZX) with 488 atoms is considered. After loading the PDB data, we know the centers and partial charges of these 488 atoms. Then a sphere with a radius a is employed to bound all atoms. This sphere is denoted as our solute region Ω^- with a dielectric constant ϵ^-. Then, we choose $\Omega = \Omega^- \cup \Omega^+$ being a larger sphere of radius A containing Ω^-, and let $\epsilon = \epsilon^+$ in Ω^+.

Consider the 3D nonlinear PB equation in the dimensionless form given in Eq. (1) with a source term ρ

$$\rho = \begin{cases} \sum_{i=1}^{N_m} q_i \delta(\vec{x} - \vec{x}_i) & in \quad \Omega^- \\ f_s(\vec{x}) + \kappa^2 \sinh(u) & in \quad \Omega^+ \end{cases} \tag{45}$$

where q_i and $\vec{x}_i$ are partial charges and atom centers

$$\begin{aligned} f_s(\vec{x}) = \frac{(\epsilon^- - \epsilon^+)}{4\pi a^2 \epsilon^-} \sum_{i=1}^{N_m} q_i \Bigg[& \cos\left(\frac{|\vec{x}|^2}{a^2} - 1\right) \frac{7|\vec{x}|^2 - 5\vec{x}\cdot\vec{x}_i}{|\vec{x} - \vec{x}_i|^3} \\ & - \sin\left(\frac{|\vec{x}|^2}{a^2} - 1\right) \frac{2|\vec{x}|^2 \left(|\vec{x}|^2 - \vec{x}\cdot\vec{x}_i\right)}{a^2 |\vec{x} - \vec{x}_i|^3} \\ & - 6\cos\left(\frac{|\vec{x}|^2}{a^2} - 1\right) \frac{\left(|\vec{x}|^2 - \vec{x}\cdot\vec{x}_i\right)^2}{|\vec{x} - \vec{x}_i|^5} \Bigg] \qquad in \quad \Omega^+. \end{aligned} \tag{46}$$

Here u is the analytic solution with the decomposition $u = \bar{u} + G$, where

$$\bar{u}(\vec{x}) = \begin{cases} 0 & in \quad \Omega^-, \\ \frac{(\epsilon^+ - \epsilon^-)}{8\pi\epsilon^-\epsilon^+} \sin\left(\frac{|\vec{x}|^2}{a^2} - 1\right) \sum_{i=1}^{N_m} \frac{q_i(\vec{x} - \vec{x}_i)\cdot\vec{x}}{|\vec{x} - \vec{x}_i|^3} & in \quad \Omega^+. \end{cases} \tag{47}$$

$$G = \frac{1}{4\pi\epsilon^-} \sum_{i=1}^{N_m} \frac{q_i}{|\vec{x} - \vec{x}_i|}, \quad in \quad \Omega. \tag{48}$$

The boundary data is $g(\vec{x}) = G(\vec{x}) + \bar{u}(\vec{x})$ on $\partial\Omega$.

The model parameters are chosen as: $a = 17, A = 34, \kappa^2 = 0.8486902807, \epsilon^- = 2.0$, and $\epsilon^+ = 78.54$. Three grid levels are considered for a boundary sphere of radius A. The number of elements is 1703, 13,978, and 110,218, respectively, for grid level 1, 2, and 3. One cross-section view of each mesh is displayed in Fig. 4.

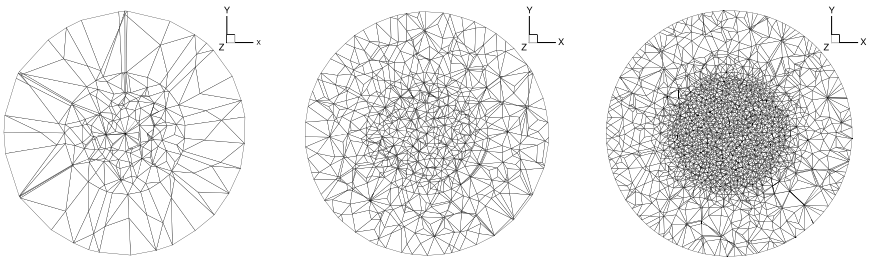

Fig. 4. Meshes used for a ball with many charges.

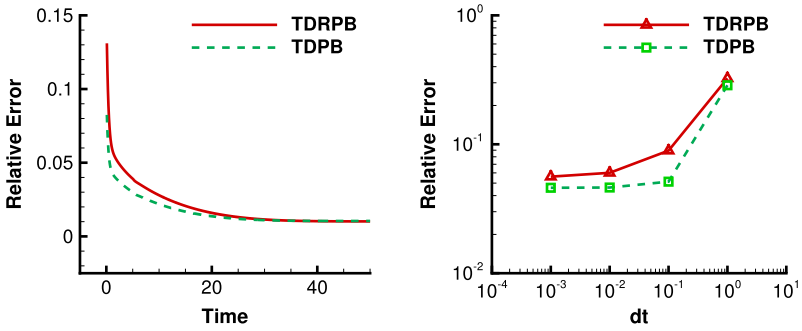

Fig. 5. Time history of TDPB and TDPRB (left) and relative errors for different time steps (right) for a ball with many charges.

Table 2
Spatial order of convergence for a ball with many charges.

Mesh data			TDPB		TDRPB	
Grid	#Vertices	#Tetrahedra	L_∞	L_2	L_∞	L_2
Level 1	325	1,703	3.64e−3	5.36e−1	4.54e−3	5.99e−1
Level 2	2,366	13,978	8.08e−4	9.96e−2	1.26e−3	1.02e−1
Level 3	17,975	110,218	4.75e−4	4.62e−2	5.81e−4	4.84e−2

3.2.1. Non-regularized method

If we just solve the present problem by applying the TDPB-FEM to the PB equation, the results will also have worrisome convergence in errors, similar to the previous single charge ball test. On the other hand, as discussed above that in free energy calculations by the non-regularized methods, besides solving the PB equation for a solution u_{pb}, one needs to solve a Poisson equation in the vacuum for a solution u_{ps}. By using the same spatial discretization, it is well known [31] that the errors of u_{pb} and u_{ps} caused by the singular charges will somehow be canceled out in free energy calculations. Therefore, in order to fairly evaluate the performance of the proposed TDPB-FEM, we will apply it to both the PB equation (1) and a Poisson's equation $-\epsilon^- \Delta u_{ps} = \sum_{i=1}^{N_m} q_i \delta(\vec{x} - \vec{x}_i) \quad in \quad \Omega$ with boundary condition $g(\vec{x}) = \frac{1}{4\pi\epsilon^-} \sum_{i=1}^{N_m} \frac{q_i}{|\vec{x}-\vec{x}_i|}$ on $\partial\Omega$. Consequently, the numerical error of the TDPB method is evaluated as

$$\begin{cases} L_\infty = \max\limits_{\vec{x}\in\Omega} |u_{pb}(\vec{x}) - u_{ps}(\vec{x}) - \bar{u}(\vec{x})| \\ L_2 = \sqrt{\int_\Omega |u_{pb}(\vec{x}) - u_{ps}(\vec{x}) - \bar{u}(\vec{x})|^2 dv}. \end{cases} \tag{49}$$

The strategy of error cancellation works decently for the TDPB method. By using $\Delta t = 0.1$, the spatial errors are listed in Table 2. For a more challenging problem with 488 charges, the TDPB method now provides a fairly fast convergence in both L_∞ and L_2 norms. The TDPB errors against different Δt are shown in Fig. 5. It can be seen that $\Delta t = 0.1$ is enough for the TDPB method as a smaller Δt will not improve the accuracy. As more complicated analytic solution is constructed in Eq. (47), a longer stopping time T_s is required for the present problem to reach the steady state, as shown in the left figure of Fig. 5.

3.2.2. Regularized method

For the TDRPB-FEM, we simply take u^s as G given in Eq. (48), and numerically approximate u^h and u^r. In particular, u^h is the solution of the linear interface problem (23), in which $\frac{\partial G}{\partial n} = -\frac{1}{4\pi\epsilon^-} \sum_{i=1}^{N_m} \frac{q_i(\vec{x}-\vec{x}_i)\cdot\vec{n}}{|\vec{x}-\vec{x}_i|^3}$, and $u^h = \bar{u}$ on $\partial\Omega$. Without introducing

a pseudo-time, this linear interface Poisson equation can be simply solved by the continuous FEM. Here u^r is the solution of the nonlinear interface problem (24), which is rewritten as the following

$$\begin{cases} -\epsilon^- \Delta u^r = 0 & in \quad \Omega^-, \\ -\epsilon^+ \Delta u^r + \kappa^2 \sinh(u^s + u^h + u^r) = f_s(\vec{x}) + \kappa^2 \sinh(\bar{u} + G) & in \quad \Omega^+, \\ (u^r)^- = (u^r)^+, \quad \epsilon^- \dfrac{\partial (u^r)^-}{\partial n} = \epsilon^+ \dfrac{\partial (u^r)^+}{\partial n} & on \quad \Gamma, \\ u^r = 0 & on \quad \partial\Omega. \end{cases} \tag{50}$$

This problem can be solved by a pseudo-time FEM.

The errors of the TDRPB method are defined as in (44). Based on the mesh level 3, the relative errors of the TDRPB-FEM are plotted against different Δt in the right figure of Fig. 5. Based on these results, we have chosen $\Delta t = 0.01$ for the TDRPB in spatial convergence test. The spatial convergence rate of the TDRPB is found to be comparable with that of the TDPB in Table 2. As more complicated analytic solution is constructed in Eq. (47), a longer stopping time T_s is also required for the TDRPB to achieve steady state for this multiple charges simulation, and such a T_s would be the same as that of the TDPB.

3.2.3. Comparison of regularized and non-regularized methods

A new error cancellation strategy is adopted in this example for the TDPB method, by solving an additional Poisson's equation to approximate Green's function. This strategy can be regarded as mimicking a regularized method numerically by canceling nearly blow-up values on nodal points near singular charges. Boosted by this strategy, the TDPB method yields a similar convergence rate in space as the TDRPB method. Moreover, both methods have increased errors when using larger time step, and this is different to the previous test, which probably due to the complexity of multiple charges. Using same Δt, the TDPB method seems to have a higher temporal accuracy. Again, both TDPB and TDRPB methods are unconditionally stable. Because of complicated analytic solution, the steady state is reached at a later time than the previous test with a single charge. The regularized method needs to solve an interface problem, which adds more complexity in the implementation, while the present TDPB method needs to solve a Poisson's equation additionally. Thus, for each time step, a similar execution time is needed in both methods. However, since the TDRPB method has to use a much smaller Δt, the speed of non-regularized method is much faster than the regularized method.

4. Applications

In this section, we further examine the proposed TDPB-FEM and TDRPB-FEM by studying free energies of a one-atom model and several real protein systems. The protein structure preparation and formulas for calculating free energies in both regularized and non-regularized approaches have been discussed in Section 2.

4.1. Free energy for a single charge ball model

We start our test with a single charge q_0 at the center of a sphere with radius a. The free energy of such a dielectric sphere is analytically known [13]

$$\Delta G = -\frac{q_0^2}{2a}\left(\frac{1}{\epsilon^-} - \frac{1}{\epsilon^+}\right). \tag{51}$$

It is customary to report ΔG in the unit of kcal/mol. For this one atom model with q_0 and $\vec{x}_0$ = (0,0,0), one needs to solve the PB equation (1) with ρ being given as

$$\rho = \frac{4\pi {e_c}^2}{k_B T} q_0 \delta(\vec{x}). \tag{52}$$

In this example, we consider a unit charge $q_0 = 1$ and take $a = 2$ Å. The other model parameters are the same as those in the test of single charge ball. The relative error in the free energy computation will be reported in each case

$$E_{re} = \left| \frac{\Delta G_h - \Delta G}{\Delta G} \right|, \tag{53}$$

where ΔG_h is a numerically calculated free energy and $\Delta G = -81.9782$ kcal/mol is the analytic free energy.

4.1.1. Non-regularized method free energy calculation

In the non-regularized approach, besides solving the PB equation (1) for u, a Poisson's Eq. (21) is also solved for u_0, where $g(\vec{x}) = \frac{{e_c}^2 q_0}{K_B T \epsilon^- |\vec{x}|}$ on $\partial\Omega$. Then the numerical free energy ΔG_h is calculated according to (36), and the relative error is computed using (53). Based on the mesh level 3, the change of relative error with respective to different Δt is examined in the right figure in Fig. 6. In the following, we take $\Delta t = 0.1$ to study the space convergence of the TDPB-FEM in Table 3. It can be seen that more than second order has been achieved in space for the relative error.

Table 3
Spatial convergence for free energy of a single charge ball.

Mesh data			TDPB		TDRPB	
Grid	#Vertices	#Tetrahedra	ΔG_h	E_{re}	ΔG_h	E_{re}
Level 1	238	1,169	−76.51	6.67e−2	−80.13	2.26e−2
Level 2	1,698	9,352	−80.76	1.49e−2	−80.94	1.27e−2
Level 3	12,963	74,816	−81.81	1.99e−3	−81.71	3.23e−3

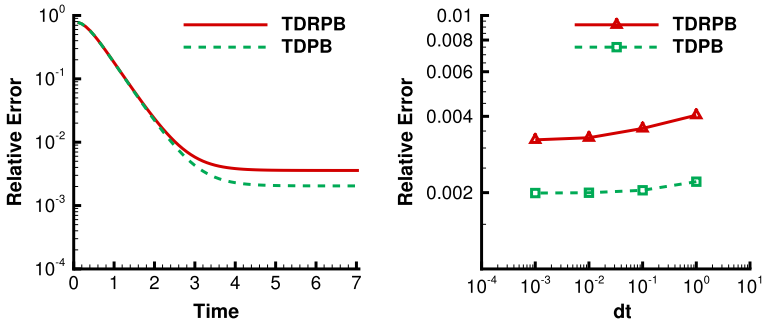

Fig. 6. Time history of the TDPB and TDRPB (left) and relative errors for different time steps (right) for free energy calculation of one atom.

4.1.2. Regularized method free energy calculation

In the regularized approach, we need to calculate u^h and u^r. Here u^h is the solution of the linear interface problem (23), in which $\frac{\partial G}{\partial n} = -\frac{e_c^2 q_0}{\epsilon^- K_B T |\vec{x}|^2}$, and $u^h(\vec{x}) = \frac{e_c^2}{K_B T} \cdot \frac{q_0}{\epsilon^+ |\vec{x}|} exp(-|\vec{x}|\sqrt{\frac{\kappa^2}{\epsilon^+}}) - \frac{e_c^2}{K_B T} \cdot \frac{q_0}{\epsilon^- |\vec{x}|}$ on $\partial\Omega$. And u^r is the solution of the nonlinear interface problem (24). Then the numerical free energy ΔG_h is calculated according to (37), and the relative error is computed using (53). The right figure in Fig. 6 shows that the errors converge at $\Delta t = 0.01$. Thus, we set $\Delta t = 0.01$ in the TDRPB method for all grid levels. We found that the spatial order of the TDRPB method is about two in Table 3.

4.1.3. Comparison of TDPB and TDRPB for free energy on single charge ball

From above, both TDPB and TDRPB methods achieve second order convergence rate in space. In fact, the TDPB-FEM provides an even better result in free energy calculation, which means that an error cancellation has been taken advantage of in the non-regularized methods for free energy computations. The time histories of both methods are monotonic before the steady state is reached, as shown in the left panel of Fig. 6. The TDPB method also performs better in temporal convergence.

4.2. Computation of free energy on real proteins

In this section, we apply the proposed TDPB-FEM and TDRPB-FEM to real proteins. We first compute the free energy of a protein to verify the stability and convergence of the proposed FEMs. Particular attention is paid to investigate the impact of numerical parameters to the calculated free energy values. In particular, we will explore suitable values for the TetGen parameter V_{max}, time step Δt, and the radius of domain Ω. With the recommended parameter values, free energies of several real proteins will be reported. In all numerical tests, we take $\kappa^2 = 8.486902807$, $\epsilon^- = 1$, and $\epsilon^+ = 80$. The protein for parameter calibration is 2LZX with 488 atoms, which was downloaded from the Protein Data Bank. As required by the proposed FEMs, we converted the PDB file of 2LZX to a PQR file using the software tool PDB2PQR. The developed MMS mesh generator based on the TetGen was used to construct meshes for a boundary sphere, see Fig. 7.

4.2.1. Stability

We first explore the stability of the proposed FEMs for real proteins. For this purpose, a large $\Delta t = 1$ and two different initial fields are studied. One initial function is taken as the boundary function (4), but numerically evaluated for all nodal values in Ω, except at the atom centers. The other initial field is obtained by multiplying the first one by a factor of 100,000. We then carry out the time integration until $T_s = 10,000$ for both TDPB and TDRPB methods. In Fig. 8, line Init1 shows the convergence by using the first initial function, while line Init2 is for the scaled one. It can be seen from Fig. 8 that for both TDPB and TDRPB methods, the same steady state is achieved by using different initial values. This demonstrates the robustness of the pseudo-time approach for solving the PB equation. Moreover, this figure also indicates that both TDPB and TDRPB methods are unconditionally stable. Given the fact that one initial function has been scaled by 100,000 times, this means that the pseudo-time approach is free of the blow-up issue, which has been reported in other FEM studies [39].

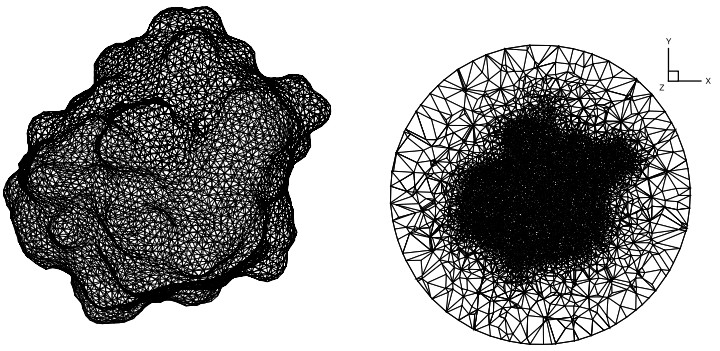

Fig. 7. The mesh for 2lzx.

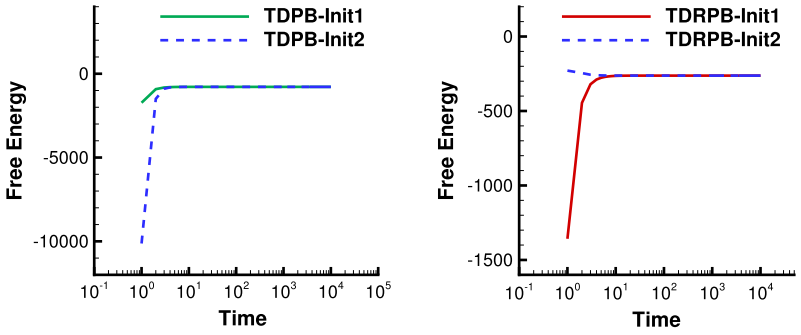

Fig. 8. Stability test with different initial functions for the TDPB (left) and TDRPB (right).

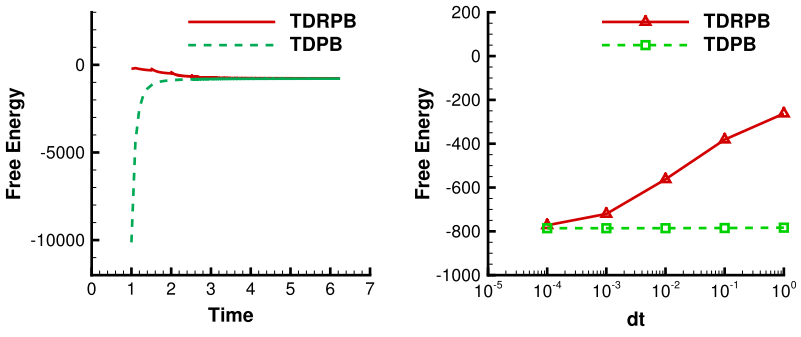

Fig. 9. Left: time history of the TDPB and TDRPB with adaptive Δt; Right: Free energy for different time steps.

4.2.2. Temporal accuracy

Based on a fixed mesh, the free energies of TDPB and TDRPB methods by using different Δt are depicted in the right figure of Fig. 9. It can be seen that $\Delta t = 0.1$ is enough for the TDPB method as a smaller Δt will not improve the accuracy of free energy. While for the TDRPB method, a very small $\Delta t = 0.0001$ has to be used to guarantee the accuracy. This implies a very expensive computation for the TDRPB method. In order to save the CPU time, we have designed an adaptive Δt scheme, by gradually refining Δt to be smaller and smaller. Initially, $\Delta t = 0.1$ is used. The refinement happens whenever Δt times time steps equal 0.5, then Δt becomes one-tenth smaller. But if $\Delta t \leq 0.0001$, no further refinement will be conducted and the computation continues until $T_s = 5$ is reached. As depicted in the left figure of Fig. 9, the calculated free energy based on this adaptive scheme is very close to that by using a fixed $\Delta t = 0.0001$. In some sense, the adaptive scheme just provides a better initial function for the smaller Δt computation so that the steady state can be reached in a quicker manner.

4.2.3. Spatial convergence

Based on fixed $\Delta t = 0.0001$ for the TDRPB and $\Delta t = 0.1$ for TDPB, three grid levels are considered for Ω being a sphere with radius 20 to validate the spatial convergence. The number of elements is 64 776, 570 021, and 1 082 863, respectively,

Table 4
Spatial convergence in calculating the free energy ΔG for the 2lzx.

Mesh data				ΔG	
Grid	Radius	#Vertices	#Tetrahedra	TDPB	TDRPB
Level 1	20	10,243	64,776	−785.24	−772.23
Level 2	20	89,240	570,021	−672.92	−660.53
Level 3	20	169,042	1,082,863	−662.55	−653.46

Table 5
Impact of domain size to free energy computation for 2lzx.

Radius	#Vertices	#Tetrahedra	Δt	Steps	ΔG
20	68,148	434,286	0.1	50	−672.92
30	68,998	440,084	0.1	50	−673.59

Table 6
Comparison of free energies.

Protein	Name	1ajj	1ucs	1v05	1ng6
	#atoms	513	997	1456	2405
ΔG	MIBPB	−1125.98	−622.38	−1271.88	−2404.28
	SDPBS	−1245.37	−749.92	−1399.02	−2691.51
	TDPB	−1379.84	−630.80	−1346.83	−2493.75

for grid level 1, 2, and 3. As shown in Table 4, both TDPB and TDRPB results show certain convergence pattern. Moreover, the converged free energies by the TDPB and TDRPB methods generally agree each other. The mesh level 3, generated by taking $V_{max} = 0.015$ Å^3 in TetGen, gives a satisfactory results for both FEMs.

4.2.4. Spherical domain size

To save computation in electrostatic analysis, we further explore an economic size of computational domain by studying the protein 2lzx. In generating the triangular mesh of the MMS by the CGAL, we found that the radius $a = 16.05$ Å gives the smallest bounding sphere, that completely encloses the MMS of the 2lzx. Based on the radius a, two different sizes are examined for a spherical domain with a radius A, i.e., $A = 20$ Å and $A = 30$ Å. By considering the TDPB-FEM, the free energies calculated by using different radii are listed in Table 5. The difference in the energies is found to be negligible. Thus, domain Ω is chosen as a sphere with radius $A = 20$ Å in other simulations of 2lzx.

4.2.5. Computation on real proteins

Based on the above exploration, suitable values of A, V_{max} and Δt are found for the computation of real proteins. For the radius of the spherical domain Ω, we set $A = a + 4$, where a corresponds to the smallest bounding sphere that packs the MMS surface of the protein. A larger spherical domain Ω will not improve the accuracy of free energy. For mesh size, we set $V_{max} = 0.015$ Å^3 for mesh level 3, since a smaller mesh size consumes more computational time and the solutions with this mesh size are acceptable. For Δt, we use $\Delta t = 0.1$ in TDPB-FEM. However, since Δt of the TDRPB is one thousand times smaller than that of the TDPB, we do not apply it for the computation of real proteins. Based on the above parameters, we computed free energies of several real proteins (see Table 6) and compared them to other popular solvers, including MIBPB [29] and SDPBS [30]. For both MIBPB and SDPBS, the solvent excluded surface (SES) is utilized, while the present TDPB uses a MMS. Albeit the difference in molecular surface definition, the free energies of the TDPB are comparable to those of MIBPB and SDPBS.

The CPU times of TDPB-FEM are listed in Table 7. Here, both the execution times for mesh generation and numerical solution of the PB equation are reported. In particular, the mesh generation consists of three parts, and is computed sequentially. MMS time is for generating surface triangle mesh on the surface of the protein; TETGEN time is for generating the tetrahedral mesh in the protein and solvent; METIS time is for partition the mesh for different processors. The TDPB-FEM is calculated in parallel, based on 64 cores.

5. Conclusion

In this paper, several methods for solving the PBE based on FEM have been developed in a pseudo-time approach for biomolecular electrostatic simulations. These methods adopt pseudo-time approach, time splitting, fully implicit scheme, analytic solution of nonlinear term, non-regularization and regularization techniques. Moreover, a new tetrahedral mesh generator based on the minimal molecular surface definition has been developed for real protein systems. With such a body-fitted mesh, the proposed pseudo-time methods shall be more accurate than the existing pseudo-time finite difference methods.

Table 7
CPU time in seconds for mesh generation and TDPB computation. A parallel computation is conducted for the TDPB-FEM using 64 cores.

Protein	Name	1ajj	1ucs	1v05	1ng6
	#atoms	513	997	1456	2405
Mesh	Radius A	22.43	24.08	33.34	39.20
	#Vertices	45,942	92,198	140,802	227,529
	#Tetrahedra	291,407	588,804	899,799	1,456,431
	MMS time	41.79	52.39	93.30	190.56
	Tetgen time	5.53	11.07	16.99	31.91
	Metis time	6.72	14.65	24.26	50.85
TDPB	#cores	64	64	64	64
	Time	1.90	2.53	4.06	5.83

The pseudo-time approach is an indirect way to solve PBE in the literature, whose performance can be greatly enhanced by using time splitting methods. Furthermore, by using the implicit Euler time integration, the proposed FEMs are found to be unconditionally stable in all numerical tests. It can suppress the exponential non-linearity of the PBE very well. This allows us to use a large Δt, which is impossible in the existing pseudo-time discontinuous Galerkin method based on the explicit Euler time stepping.

Two treatments of the singular charge sources were investigated. In the non-regularized method, one directly applies the definition of the delta function on the mesh, while in the regularized method, the Green's function is introduced to avoid the difficulty of expressing delta function on the mesh.

For solving the PB equation only, the non-regularized method is inaccurate. Nevertheless, in calculating electrostatic free energy, a systematic error cancellation can be adopted, so that the non-regularized method becomes comparable to or even outperforms the regularized method.

Numerical studies of several benchmarks and free energy calculations of protein systems are conducted to validate the stability, accuracy, and robustness of the proposed PBE methods,

i.e., the regularized FEM and non-regularized FEM with error cancellation. In particular, these two methods have been compared in detail in terms of speed, stability and rate of convergence for the potential and free energy of one single charge sphere, a big sphere with multiple charges, and real proteins. From these comparisons, the advantages and shortcomings of these two methods are listed in the following:

- The implementation of non-regularized method is easier than that of the regularized method.
- In algebraic computations of the linear substep, the two methods have similar convergent speeds.
- Enhanced with the error cancellation, the approximation of the delta function in the non-regularized method is as accurate as that of the regularized method for free energy calculations, while the regularized method is still more accurate in approximating the potential function alone, particularly near the singular points.
- Both methods are unconditionally stable.
- The regularized method used in this work invokes a large temporal error in pseudo-time simulations so that its efficiency is greatly impaired in steady state computations. Consequently, the non-regularized method is much more efficient.

In summary, the proposed methods have been proven to be stable and accurate to solve the PBE for electrostatic analysis of real proteins with complex geometries. In the future, the methods will be further improved in performance and efficiency for more complex biomolecular simulations.

Acknowledgments

We thank Benzhuo Lu in the Institute of Computational Mathematics, Chinese Academy of Science (CAS) for useful suggestions. The research of Zhao was supported in part by National Science Foundation Grant DMS-1318898 and the University of Alabama Research Stimulation Program (RSP) award. This research of Xu was support in part by One-Hundred-Talented Program in Chinese Academy of Science.

References

[1] B. Honig, A. Nicholls, Classical electrostatics in biology and chemistry, Science 268 (1995) 1144–1149.
[2] N.A. Baker, Improving implicit implicit solvent simulations: a Poisson-centric view, Curr. Opin. Struct. Biol. 15 (2005) 137–143.
[3] P.W. Bates, G.W. Wei, S. Zhao, Minimal molecular surfaces and their applications, J. Comput. Chem. 29 (2008) 380–391.
[4] P.W. Bates, Z. Chen, Y.H. Sun, G.W. Wei, S. Zhao, Geometric and potential driving formation and evolution of biomolecular surfaces, J. Math. Biol. 59 (2009) 193–231.
[5] F. Fogolari, A. Brigo, H. Molinari, The Poisson-Boltzmann equation for biomolecular electrostatics: a tool for structural biology, J. Mol. Recognit. 15 (6) (2002) 377–392.
[6] P. Koehl, Electrostatics calculations: latest methodological advances, Curr. Opin. Struct. Biol. 16 (2) (2006) 142–151.

[7] M.J. Holst, Multilevel Methods for the Poisson-Boltzmann Equation, University of Illinois at Urbana-Champaign, 1993 (Ph.D. thesis).
[8] A. Nicholls, B. Honig, A rapid finite difference algorithm, utilizing successive over-relaxation to solve the Poisson-Boltzmann equation, J. Comput. Chem. 12 (1991) 435–445.
[9] B.A. Luty, M.E. Davis, J.A. McCammon, Solving the finite-difference nonlinear Poisson-Boltzmann equation, J. Comput. Chem. 13 (1992) 1114–1118.
[10] W. Im, D. Beglov, B. Roux, Continuum solvation model: Computation of electrostatic forces from numerical solutions to the Poisson-Boltzmann equation, Comput. Phys. Comm. 111 (1998) 59–75.
[11] 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.
[12] R. Luo, L. David, M.K. Gilson, Accelerated Poisson-Boltzmann calculations for static and dynamic systems, J. Comput. Chem. 23 (2002) 1244–1253.
[13] W. Geng, S. Yu, G.W. Wei, Treatment of charge singularities in implicit solvent models, J. Chem. Phys. 127 (2007) 114106.
[14] C.M. Cortis, R.A. Friesner, Numerical solution of the Poisson-Boltzmann equation using tetrahedral finite element meshes, J. Comput. Chem. 18 (1997) 1591–1608.
[15] M. Holst, N. Baker, F. Wang, Adaptive multilevel finite element solution of the Poisson-Boltzmann equation I: algorithms and examples, J. Comput. Chem. 21 (2000) 1319–1342.
[16] L. Chen, M.J. Holst, J. Xu, The finite element approximation of the nonlinear Poisson-Boltzmann equation, SIAM J. Numer. Anal. 45 (2007) 2298–2320.
[17] D. Xie, S. Zhou, A new minimization protocol for solving nonlinear Poisson-Boltzmann mortar finite element equation, BIT 47 (2007) 853–871.
[18] M. Holst, J.A. McCammon, Z. Yu, Y.C. Zhou, Y. Zhu, Adaptive finite element modeling techniques for the Poisson-Boltzmann equation I, Commun. Comput. Phys. 11 (2012) 179–214.
[19] 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.
[20] A.H. Boschitsch, M.O. Fenley, H.-X. Zhou, Fast boundary element method for the linear Poisson-Boltzmann equation, J. Phys. Chem. B 106 (2002) 2741–2754.
[21] B.Z. Lu, X.L. Cheng, J.F. Huang, J.A. McCammon, Order N algorithm for computation of electrostatic interactions in biomolecular systems, Proc. Natl. Acad. Sci. USA 103 (2006) 19314–19319.
[22] 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. Comm. 182 (2011) 1272–1283.
[23] W. Geng, R. Krasny, A treecode-accelerated boundary integral Poisson-Boltzmann solver for continuum electrostatics of solvated biomolecules, J. Comput. Phys. 247 (2013) 62–87.
[24] L. Bedin, P.R. Bosing, Discontinuous Galerkin method for the linear Poisson-Boltzmann equation, Int. J. Appl. Math. 26 (2013) 713–726.
[25] P. Yin, Y. Huang, H. Liu, An iterative discontinuous Galerkin method for solving the nonlinear Poisson-Boltzmann equation, Commun. Comput. Phys. 16 (2014) 491–515.
[26] W. Deng, X. Zhufu, J. Xu, S. Zhao, A new discontinuous Galerkin method for the nonlinear Poisson-Boltzmann equation, Appl. Math. Lett. 257 (2015) 1000–1021.
[27] J.D. Madura, J.M. Briggs, R.C. Wade, M.E. Davis, B.A. Luty, A. Ilin, J. Antosiewicz, M.K. Gilson, B. Bagheri, L.R. Scott, J.A. McCammon, Electrostatic and diffusion of molecules in solution - simulations with the University-of-Houston brownian dynamics progrom, Comput. Phys. Comm. 91 (1995) 57–95.
[28] N.A. Baker, D. Sept, S. Joseph, M.J. Holst, J.A. McCammon, Electrostatic of nanosystems: application to microtubes and ribosome, Proc. Natl. Acad. Sci. USA 98 (2001) 10037–10041.
[29] D. Chen, Z. Chen, C. Chen, W. Geng, G.W. Wei, MIBPB: a software package for electrostatic analysis, J. Comput. Chem. 32 (2011) 756–770.
[30] Y. Jiang, Y. Xie, J. Ying, D. Xie, Z. Yu, SDPBS web server for calculation of electrostatics of ionic solvated biomolecules, Mol. Based Math. Biol. 3 (2015) 179–196.
[31] B.Z. Lu, Y.C. Zhou, M.J. Holst, J.A. McCammon, Recent progress in numerical methods for the Poisson-Boltzmann equation in biophysical applications, Commun. Comput. Phys. 3 (2008) 973–1009.
[32] Q. Cai, M.J. Hsieh, J. Wang, R. Luo, Performance of nonlinear finite-difference Poisson-Boltzmann solvers, J. Chem. Theory Comput. 6 (2010) 203–211.
[33] M. Holst, F. Saied, Numerical solution of the nonlinear Poisson-Boltzmann equation: developing more robust and efficient methods, J. Comput. Chem. 16 (3) (1995) 337–364.
[34] A.I. Shestakov, J.L. Milovich, A. Noy, Solution of the nonlinear Poisson-Boltzmann equation using pseudo-transient continuation and the finite element method, J. Colloid Interface Sci. 247 (2002) 62–79.
[35] A. Sayyed-Ahmad, K. Tuncay, P.J. Ortoleva, Efficient solution technique for solving the Poisson-Boltzmann equation, J. Comput. Chem. 25 (2004) 1068–1074.
[36] S. Zhao, Pseudo-time coupled nonlinear models for biomolecular surface representation and solvation analysis, Int. J. Numer. Method Biomed. Eng. 27 (2011) 1964–1981.
[37] S. Zhao, Operator splitting ADI schemes for pseudo-time coupled nonlinear solvation simulations, J. Comput. Phys. 257 (2014) 1000–1021.
[38] W. Geng, S. Zhao, Fully implicit ADI schemes for solving the nonlinear Poisson-Boltzmann equation, Mol. Based Math. Bio. 1 (2013) 109–123.
[39] D. Xie, New solution decomposition and minimization schemes for Poisson-Boltzmann equation in calculation of biomolecular electrostatics, J. Comput. Phys. 275 (2014) 294–309.
[40] L. Wilson, S. Zhao, Unconditionally stable time splitting methods for the electrostatic analysis of solvated biomolecules, Int. J. Numer. Anal. Model. 13 (2016) 852–878.
[41] M.O. Deville, P.F. Fischer, E.H. Mund, High-Order Methods for Incompressible Fluid Flow, Cambridge University Press, Cambridge, 2002.
[42] M. Dubiner, Spectral methods on triangles and other domains, J. Sci. Comput. 6 (1991) 345.
[43] J.S. Hesthaven, T. Warburton, Nodal Discontinuous Galerkin Methods: Algorithms, Analysis and Applications, Springer, New York, 2008.
[44] G.E. Karniadakis, S.J. Sherwin, Spectral/Hp Element Methods for CFD, Oxford University Press, London, 1999.
[45] Z. Zhou, P. Payne, M. Vasquez, N. Kuhn, M. Levitt, Finite-difference solution of the Poisson-Boltzmann equation: complete elimination of self-energy, J. Comput. Chem. 11 (1996) 1344–1351.
[46] I.-L. Chern, J.-G. Liu, W.-C. Wang, Accurate evaluation of electrostatics for macromolecules in solution, Methods Appl. Anal. 10 (2003) 309–328.
[47] B. Lee, F.M. Richards, Interpretation of protein structures: estimation of static accessibility, J. Mol. Biol. 55 (1973) 379–400.
[48] F.M. Richards, Areas, volumes, packing and protein structure, Annu. Rev. Biophys. Bioeng. 6 (1977) 151–176.
[49] J. Grant, B. Pickup, A Gaussian description of molecular shape, J. Phys. Chem. 99 (1995) 3503–3510.

[50] Z. Yu, M.J. Holst, Y. Cheng, J.A. McCammon, Feature-preserving adaptive mesh generation for molecular shape modeling and simulation, J. Mol. Graphics. Modell. 26 (2008) 1370–1380.
[51] M. Chen, B. Lu, TMSmesh: A robust method for molecular surface mesh generation using a trace technique, J. Chem. Theory Comput. 7 (2011) 203–212.
[52] W. Tian, S. Zhao, A fast alternating direction implicit algorithm for geometric flow equations in biomolecular surface generation, Int. J. Numer. Method Biomed. Eng. 30 (2014) 490–516.
[53] L. Rineau, M. Yvinec, 3D surface mesh generation, in: CGAL User and Reference Manual, 4.8.1 ed., CGAL Editorial Board, 2016.
[54] S. Hang, . Tetgen, A Delaunay-based quality tetrahedral mesh generator, ACM Trans. Math. Softw. 41 (2) (2015) 36 Article 11.
[55] Y. Jiang, J. Ying, D. Xie, A Poisson-Boltzmann equation test model for protein in spherical solute region and its applications, Mol. Based Math. Bio. 2 (2014) 86–97.