Journal of Computational Physics 275 (2014) 294–309

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

New solution decomposition and minimization schemes for Poisson–Boltzmann equation in calculation of biomolecular electrostatics

Dexuan Xie

Department of Mathematical Sciences, University of Wisconsin–Milwaukee, Milwaukee, WI 53211, USA

ARTICLE INFO

Article history:
Received 11 September 2013
Received in revised form 3 May 2014
Accepted 3 July 2014
Available online 16 July 2014

Keywords:
Poisson–Boltzmann equation
Finite element method
Variational minimization
Biomolecular electrostatics
FEniCS project

ABSTRACT

The Poisson–Boltzmann equation (PBE) is one widely-used implicit solvent continuum model in the calculation of electrostatic potential energy for biomolecules in ionic solvent, but its numerical solution remains a challenge due to its strong singularity and nonlinearity caused by its singular distribution source terms and exponential nonlinear terms. To effectively deal with such a challenge, in this paper, new solution decomposition and minimization schemes are proposed, together with a new PBE analysis on solution existence and uniqueness. Moreover, a PBE finite element program package is developed in Python based on the `FEniCS` program library and `GAMer`, a molecular surface and volumetric mesh generation program package. Numerical tests on proteins and a nonlinear Born ball model with an analytical solution validate the new solution decomposition and minimization schemes, and demonstrate the effectiveness and efficiency of the new PBE finite element program package.

1. Introduction

The Poisson–Boltzmann equation (PBE) is one widely-used implicit solvent continuum model in the calculation of electrostatic potential energy for biomolecule in ionic solvent [1–4]. However, its numerical solution is very challenging due to strong singularity and nonlinearity caused by its singular distribution and exponential nonlinear terms. In the past two decades, these challenges were addressed via typical numerical techniques (such as finite difference, finite element, and boundary element methods) and popular linear and nonlinear iterative methods (such as the successive over-relaxation method, the conjugate gradient method, the inexact-Newton method, the multigrid method, and numerical optimization methods) [4–14]. Several PBE program packages and web-based resources were developed, which include DelPhi [6,15], MEAD [16], APBS [17,18], PBE solver modules in the biomolecular modeling and simulation programs AMBER [19,20], CHARMM [21–23], and NAMD [17,24], making the PBE model a powerful simulation tool in the study of biomolecular structure, biological function, catalytic activity, ligand association, and rational drug design [1,25–27].

To further improve current PBE mathematical analysis, in this paper, we first present a novel PBE solution decomposition to split the PBE solution u into three parts within both the solute domain D_p and solvent domain D_s. These three parts, G, Ψ, and $\tilde{\Phi}$ (see Theorem 3.1), correspond to electrostatic contributions from the biomolecular charges, the boundary and interface conditions, and the ionic solvent charges, respectively. Here, G is a known function collecting all the singular points of u while both Ψ and $\tilde{\Phi}$ are twice differentiable in D_p and D_s. Hence, u can be found through calculating Ψ

E-mail address: dxie@uwm.edu.

http://dx.doi.org/10.1016/j.jcp.2014.07.012
0021-9991/

and $\tilde{\Phi}$ without involving any singular difficulty. Note that our solution decomposition differs from those in [28–31]. In the decomposition from [28], u was split within D_p only. In the decompositions from [29–31], Ψ and $\tilde{\Phi}$ were defined by elliptic boundary value problems with discontinuous coefficients, which had definitions only in the weak sense. Our Ψ and $\tilde{\Phi}$ are defined by elliptic interface problems, which are well defined in both strong and weak senses. All of the previous solution decompositions were performed only for a symmetric 1:1 ionic solution. In contrast, our solution decomposition works for a solvent containing any number of ionic species.

As an application of this PBE solution decomposition, we then construct new finite element solution decomposition and minimization schemes for solving PBE without any singular and "blow-up" difficulty. To do so, we begin with a review of the PBE model for a biomolecule (protein or nucleic acids) immersed in an ionic solvent containing n ionic species. In this concise review, the PBE model is clearly described in both SI (Systéke International) units and electrostatic units. Here, the values and units of all involved physical parameters are given for the convenience of study. We then show that the PBE model using either electrostatic or SI units can be transformed into the same dimensionless form (see (5)) when the length is measured in angstroms (Å). Hence, we only need to consider this dimensionless PBE model for the calculations of biomolecular electrostatics.

Our PBE solution decomposition and proof on PBE solution existence and uniqueness are presented for the dimensionless PBE model (see Theorems 3.1, 4.1, and 4.2). Note that a Lagrange finite element space can be a finite dimensional subspace of a Sobolev function space. Hence, a PBE finite element solution decomposition scheme (see Algorithm 1) can be directly followed from this novel PBE solution decomposition. As a finite element method, it includes the interface conditions of the PBE model naturally so that it can produce a numerical PBE solution with a higher numerical accuracy than a finite difference method.

Typically, a nonlinear boundary value problem is solved numerically as a system of nonlinear algebraic equations. To achieve a global convergence, an "artificial" merit function (see (39)) is often employed to yield a trust-region, line-search, or inexact Newton method [10,32–34]. Because of our PBE solution decomposition, $\tilde{\Phi}$ is found to be a unique solution of a variational minimization problem with a target functional J over a Sobolev function space, and the first and second derivatives of J are available (see Theorem 4.2). Hence, J is a "natural" merit function for us to use to develop an efficient Newton-type minimization algorithm for solving a nonlinear boundary value problem of $\tilde{\Phi}$. In our early work [14], we showed one minimization protocol for solving a system of PBE mortar finite element equations to be much more efficient and effective than a popular nonlinear iterative solver — a subspace trust region Newton method [32,33]. In this paper, we intend to extend this work to the case of a Lagrange finite element approximation to $\tilde{\Phi}$.

With our PBE solution decomposition, we propose a simple treatment to deal with a potential "blow-up" problem caused by PBE exponential nonlinear terms without affecting the accuracy of a PBE numerical solution. So far, we did not see any paper that addressed such a "blow-up" issue. We only encountered one treatment on the "blow-up" issue in a code survey of the program package APBS. In our simple treatment, we first construct a modified Newton bilinear form (see (40)) using a function truncation strategy. A new modified Newton minimization scheme is then developed through solving this modified Newton bilinear form by the preconditioned conjugate gradient (PCG) method with incomplete LU (ILU) preconditioning. Our function truncation strategy may not affect any accuracy of a PBE finite element solution since a possible modification to the target functional J or its derivatives happens only in the early stage of a minimum search process. To reflect a possible affect of a modified J to the new modified Newton minimization scheme, a special iteration test (see (41)) is added to make the modified Newton minimization scheme more robust. Eventually, our new modified Newton minimization scheme becomes a descent search method so that its convergence can be followed directly from the descent search minimization theory [34,35].

A combination of the PBE solution decomposition scheme with the modified Newton minimization scheme leads to a new effective PBE finite element solver. In this paper, we program it in Python as a new PBE finite element program package for a protein in a symmetric 1:1 ionic solvent based on the FEniCS finite element library [36] and a molecular surface and volumetric mesh generation program package, `GAMer` [37]. As a Python program package, our new PBE program package is easy to be used and portable on different computer operating systems. Due to the FEniCS finite element library, various finite element methods and various direct and iterative linear solvers become available for calculating Ψ and $\tilde{\Phi}$ numerically. We adapted `GAMer` as a Python module so that a tetrahedral mesh can be generated within our PBE program package to match the need of a FEniCS finite element solver. In addition, to speed up calculation, we wrote Fortran subroutines for computing the mesh node values of G, ∇G, and our modified hyperbolic functions (see (49)), and converted them as Python modules. All the related parameters from the PBE model, `DOLFIN`, and `GAMer` are collected into one parameter file, with which we can easily control solution accuracy and mesh quality. In this way, a protein file is the only input file for an implementation of our new PBE finite element program package.

With this new PBE finite element Python program package, we first made numerical tests on a nonlinear Born ball model with analytical solution using linear, quadratic, and cubic finite element methods. Numerical results validated the PBE solution decomposition scheme and our new PBE finite element program package. We then conducted numerical experiments on a protein suite with the number of atoms up to 6062 in a linear finite element method. Numerical results demonstrated the effectiveness and efficiency of the modified Newton minimization scheme and the high performance of our new PBE finite element program package. For example, in a test of protein represented in the PDB file 4PTI, the total computer CPU time was only about 31 seconds on one 2.3 GHz Intel Core i7 of a MacBook Pro, which included the time spent on the generation of a finite element mesh with 33 572 vertices and 191 372 tetrahedra.

Table 1
Parameters of the PBE model (1) in SI units.

Parameter	Value	Unit (abbr.)	Name
ϵ_0	$8.854187817 \times 10^{-12}$	Farad/meter (F/m)	Permittivity of vacuum
e_c	$1.602176565 \times 10^{-19}$	Coulomb (C)	Elementary charge
T	298.15	Kelvin (K)	Absolute temperature
k_B	$1.380648813 \times 10^{-23}$	Joule/Kelvin (J/K)	Boltzmann constant

Finally, to further improve the performance of our PBE program package, we can use the truncated Newton strategies [38–40] to construct a truncated modified Newton minimization scheme. Because the truncated Newton minimization method has been programmed in the truncated Newton Fortran program package TNPACK [38,39], one simple way to implement our PBE truncated modified Newton minimization scheme is to call TNPACK directly. We did so by converting TNPACK into a Python module. As required by the TNPACK usage, we wrote a driver file and the three required "callback functions" in Python for computing function values, gradient vectors, Hessian matrices, the Hessian matrix vector product, and preconditioners. Currently, we did not gain any significant performance improvement yet due to the low efficiency of callback functions in Python. Hence, such numerical tests were not reported in this paper since more studies are needed on the truncated modified Newton minimization scheme and program. We plan to do so in the future.

The remaining parts of the paper are outlined as follows. In Section 2, the PBE model is reviewed. In Section 3, the PBE solution decomposition scheme is described. In Section 4, a proof on the PBE solution existence and uniqueness is given. In Section 5, the new PBE solution decomposition and minimization schemes are presented. Finally, the new PBE finite element program package and numerical results are reported in Section 6.

2. The definition of the PBE model

Let Ω be a sufficiently large bounded domain satisfying that

$$\Omega = D_p \cup D_s \cup \Gamma,$$

where D_p is a solute region that hosts a biomolecule (e.g., protein) with n_p atoms, D_s is a solvent region surrounding D_p, which contains n different species of ions, and Γ is an interface between D_p and D_s. The position $\mathbf{r}_j$ and charge number z_j of the jth atom of the biomolecule and the charge number Z_i of ionic species i are given. Based on the continuum implicit solvent approach, both D_p and D_s are treated as continuum media with ϵ_p and ϵ_s being their dielectric constants (or called relative permittivity constants), respectively. It is common to set $\epsilon_p = 2$ and $\epsilon_s = 80$.

2.1. The PBE model in SI units

In the above notation, the PBE model in SI units is defined by

$$\begin{cases} -\epsilon_p \Delta\Phi(\mathbf{r}) = \dfrac{e_c}{\epsilon_0}\displaystyle\sum_{j=1}^{n_p} z_j \delta_{\mathbf{r}_j}, & \mathbf{r} \in D_p, \\ \epsilon_s \Delta\Phi(\mathbf{r}) + \dfrac{e_c}{\epsilon_0}\displaystyle\sum_{i=1}^{n} Z_i M_i e^{-\frac{Z_i e_c}{k_B T}\Phi} = 0, & \mathbf{r} \in D_s, \\ \Phi(\mathbf{s}^+) = \Phi(\mathbf{s}^-), \quad \varepsilon_s \dfrac{\partial\Phi(\mathbf{s}^+)}{\partial\mathbf{n}(\mathbf{s})} = \varepsilon_p \dfrac{\partial\Phi(\mathbf{s}^-)}{\partial\mathbf{n}(\mathbf{s})}, & \mathbf{s} \in \Gamma, \\ \Phi(\mathbf{s}) = g(\mathbf{s}), & \mathbf{s} \in \partial\Omega, \end{cases} \tag{1}$$

where Φ is the electrostatic potential function in volts, g is a given function, $\partial\Omega$ denotes the boundary of Ω, e_c is the electron charge, ϵ_0 is the permittivity of vacuum, M_i is an average number density (or called the bulk concentration) of the ith ionic species per cubic meters, k_B is the Boltzmann constant, T is the absolute temperature, $\mathbf{n}(\mathbf{s})$ denotes the unit outward normal vector of D_p, and $\delta_{\mathbf{r}_j}$ is the Dirac-delta distribution, which is defined by $\langle \delta_{\mathbf{r}_j}, v \rangle = v(\mathbf{r}_j)$ for any test function v [41].

The values and units of physical parameters ϵ_0, e_c, T, and k_B are listed in Table 1. See the NIST website http://physics.nist.gov/cuu/Constants/index.html for updates.

When Ω is large enough, it is common to set $g = 0$ since $\Phi(\mathbf{r}) \to 0$ as $|\mathbf{r}| \to \infty$. Other selections of g can be found in [2,4] for example.

2.2. The dimensionless PBE model

By Table 1 and the formula 1 V = 1 J/C, the unit of $\frac{e_c}{k_B T}$ is found to be 1/V. Thus, by setting

$$u = \frac{e_c}{k_B T}\Phi, \tag{2}$$

the PBE model (1) can be transformed as

$$\begin{cases} -\epsilon_p \Delta u(\mathbf{r}) = \dfrac{e_c^2}{\epsilon_0 k_B T} \displaystyle\sum_{j=1}^{n_p} z_j \delta_{\mathbf{r}_j}, & \mathbf{r} \in D_p, \\ \epsilon_s \Delta u(\mathbf{r}) + \dfrac{e_c^2}{\epsilon_0 k_B T} \displaystyle\sum_{i=1}^{n} Z_i M_i e^{-Z_i u} = 0, & \mathbf{r} \in D_s, \\ u(\mathbf{s}^+) = u(\mathbf{s}^-), \quad \varepsilon_s \dfrac{\partial u(\mathbf{s}^+)}{\partial \mathbf{n}(\mathbf{s})} = \varepsilon_p \dfrac{\partial u(\mathbf{s}^-)}{\partial \mathbf{n}(\mathbf{s})}, & \mathbf{s} \in \Gamma, \\ u(\mathbf{s}) = \hat{g}(\mathbf{s}), & \mathbf{s} \in \partial\Omega, \end{cases} \tag{3}$$

where $\hat{g} = \frac{e_c}{k_B T} g$. Clearly, both $\delta_{\mathbf{r}_j}$ and M_i have unit $1/\mathrm{m}^3$, Δu has unit $1/\mathrm{m}^2$, and $\frac{e_c^2}{\epsilon_0 k_B T}$ has unit m due to $\mathrm{C}^2/(\mathrm{FJ}) = 1$. Hence, the both sides of each equation of (3) have the same unit $1/\mathrm{m}^2$, which can be removed from the both sides of each equation to make (3) become dimensionless.

The solution u of the PBE model (3) is called the dimensionless electrostatic potential. When u is found, the electrostatic potential Φ can be simply said to have a value of u in units $k_B T/e_c$.

In biomolecular electrostatic calculation, the length is measured in angstrom (Å), and M_i is often given as a molar concentration: $M_i = \bar{c}_i$ mole/liter with $\bar{c}_i$ being a given nonnegative real number. Thus, $\Delta u(\mathbf{r})$ and $\delta_{\mathbf{r}_j}$ have the units Å^{-2} and Å^{-3}, respectively, and the total number of ions of the ith species per liter can be estimated as $N_A \bar{c}_i$ so that M_i is estimated by

$$M_i = N_A \bar{c}_i/\text{liter} = 10^3 N_A \bar{c}_i/\mathrm{m}^3 = 10^{-27} N_A \bar{c}_i/\text{Å}^3, \tag{4}$$

where $N_A = 6.02214129 \times 10^{23}$, which is the Avogadro number, and the formulas 1 liter $= 10^{-3}$ m^3, and 1 m $= 10^{10}$ Å have been used. It is common to set $\bar{c}_i = 0.1$ or 0.2.

Applying (4) to (3), and changing all the length units from meters to angstroms, we obtain the commonly-used PBE dimensionless model:

$$\begin{cases} -\epsilon_p \Delta u(\mathbf{r}) = \alpha \displaystyle\sum_{j=1}^{n_p} z_j \delta_{\mathbf{r}_j}, & \mathbf{r} \in D_p, \\ \epsilon_s \Delta u(\mathbf{r}) + \beta \displaystyle\sum_{i=1}^{n} Z_i \bar{c}_i e^{-Z_i u} = 0, & \mathbf{r} \in D_s, \\ u(\mathbf{s}^+) = u(\mathbf{s}^-), \quad \varepsilon_s \dfrac{\partial u(\mathbf{s}^+)}{\partial \mathbf{n}(\mathbf{s})} = \varepsilon_p \dfrac{\partial u(\mathbf{s}^-)}{\partial \mathbf{n}(\mathbf{s})}, & \mathbf{s} \in \Gamma, \\ u(\mathbf{s}) = \hat{g}(\mathbf{s}), & \mathbf{s} \in \partial\Omega, \end{cases} \tag{5}$$

where α and β are defined by

$$\alpha = \frac{10^{10} e_c^2}{\epsilon_0 k_B T} = \frac{2.09985253128 \times 10^6}{T}, \qquad \beta = \frac{10^{-17} N_A e_c^2}{\epsilon_0 k_B T} = \frac{1264.56086316}{T}. \tag{6}$$

Here the parameters values of Table 1 have been used, the length units of angstrom have been removed from the both sides of each equation of (5), and the unit of T has been canceled out in (6). It is common to set $T = 298.15$.

Similarly, the factor $e_c/(k_B T)$ can be estimated as

$$\frac{e_c}{k_B T} = \frac{1.602176565 \times 10^{-19}}{1.380648813 \times 10^{-23} T} = \frac{1.1604519193542375 \times 10^4}{T}. \tag{7}$$

In the case of pure water, we have $\bar{c}_i = 0$ for $i = 1, 2, \ldots, n$ so that (5) is reduced to the Poisson dielectric model for biomolecule in water:

$$\begin{cases} -\epsilon_p \Delta u(\mathbf{r}) = \alpha \displaystyle\sum_{j=1}^{n_p} z_j \delta_{\mathbf{r}_j}, & \mathbf{r} \in D_p, \\ \Delta u(\mathbf{r}) = 0, & \mathbf{r} \in D_s, \\ u(\mathbf{s}^+) = u(\mathbf{s}^-), \quad \varepsilon_s \dfrac{\partial u(\mathbf{s}^+)}{\partial \mathbf{n}(\mathbf{s})} = \varepsilon_p \dfrac{\partial u(\mathbf{s}^-)}{\partial \mathbf{n}(\mathbf{s})}, & \mathbf{s} \in \Gamma, \\ u(\mathbf{s}) = \hat{g}(\mathbf{s}), & \mathbf{s} \in \partial\Omega. \end{cases} \tag{8}$$

2.3. The PBE model for symmetric 1:1 ionic solution

In terms of $\bar{c}_i$, an ionic strength, I_s, of the solvent is defined by

$$I_s = \frac{1}{2}\sum_{i=1}^{n} \bar{c}_i Z_i^2. \tag{9}$$

In the case of a symmetric 1:1 ionic solution (e.g., the one containing sodium (N_a^+) and chloride (Cl^-) ions), $n=2$, $Z_1=1$, $Z_2=-1$, and $\bar{c}_1=\bar{c}_2=I_s$ so that the PBE model (5) can be simplified as

$$\begin{cases} -\epsilon_p \Delta u(\mathbf{r}) = \alpha \sum_{j=1}^{n_p} z_j \delta_{\mathbf{r}_j}, & \mathbf{r}\in D_p, \\ -\epsilon_s \Delta u(\mathbf{r}) + \kappa^2 \sinh(u) = 0, & \mathbf{r}\in D_s, \\ u(\mathbf{s}^+) = u(\mathbf{s}^-), \quad \varepsilon_s \dfrac{\partial u(\mathbf{s}^+)}{\partial \mathbf{n}(\mathbf{s})} = \varepsilon_p \dfrac{\partial u(\mathbf{s}^-)}{\partial \mathbf{n}(\mathbf{s})}, & \mathbf{s}\in \Gamma, \\ u(\mathbf{s}) = \hat{g}(\mathbf{s}), & \mathbf{s}\in \partial\Omega, \end{cases} \tag{10}$$

where $\kappa^2 = 2\beta I_s$. For $T=298.15$ and $I_s=0.1$, by (6), it is found that

$$\alpha = 7042.93990033, \qquad \kappa^2 = 0.8482715835384875. \tag{11}$$

In particular, when u satisfies the assumption

$$\left|u(\mathbf{r})\right| < 1 \quad \forall \mathbf{r}\in D_s, \tag{12}$$

the Taylor series of $\sinh(u)$ gives

$$\sinh\big(u(\mathbf{r})\big) = u(\mathbf{r}) + u(\mathbf{r})^3/6 + O\big(u(\mathbf{r})^5\big) \approx u(\mathbf{r}) \quad \forall \mathbf{r}\in D_s.$$

Thus, (10) is approximately as the linear PBE (LPBE) model:

$$\begin{cases} -\epsilon_p \Delta u(\mathbf{r}) = \alpha \sum_{j=1}^{n_p} z_j \delta_{\mathbf{r}_j}, & \mathbf{r}\in D_p, \\ -\epsilon_s \Delta u(\mathbf{r}) + \kappa^2 u(\mathbf{r}) = 0, & \mathbf{r}\in D_s, \\ u(\mathbf{s}^+) = u(\mathbf{s}^-), \quad \varepsilon_s \dfrac{\partial u(\mathbf{s}^+)}{\partial \mathbf{n}(\mathbf{s})} = \varepsilon_p \dfrac{\partial u(\mathbf{s}^-)}{\partial \mathbf{n}(\mathbf{s})}, & \mathbf{s}\in \Gamma, \\ u(\mathbf{s}) = \hat{g}(\mathbf{s}), & \mathbf{s}\in \partial\Omega. \end{cases} \tag{13}$$

A solution of the LPBE model can be selected as an initial guess to the solution of the PBE model (10).

2.4. The PBE model in electrostatic units

The PBE model in electrostatic units is also often used in electrostatic calculations [2,4], which is defined by

$$\begin{cases} -\epsilon_p \Delta \Phi(\mathbf{r}) = 4\pi e_c \sum_{j=1}^{n_p} z_j \delta_{\mathbf{r}_j}, & \mathbf{r}\in D_p, \\ \epsilon_s \Delta \Phi(\mathbf{r}) + 4\pi e_c \sum_{i=1}^{n} Z_i M_i e^{-\frac{Z_i e_c}{k_B T}\Phi} = 0, & \mathbf{r}\in D_s, \\ \Phi(\mathbf{s}^+) = \Phi(\mathbf{s}^-), \quad \varepsilon_s \dfrac{\partial \Phi(\mathbf{s}^+)}{\partial \mathbf{n}(\mathbf{s})} = \varepsilon_p \dfrac{\partial \Phi(\mathbf{s}^-)}{\partial \mathbf{n}(\mathbf{s})}, & \mathbf{s}\in \Gamma, \\ \Phi(\mathbf{s}) = g(\mathbf{s}), & \mathbf{s}\in \partial\Omega, \end{cases} \tag{14}$$

where the length is measured in centimeter (cm), e_c in the electrostatic unit esu (or called statcoulomb), T still in Kelvin (K), k_B in the energy unit erg/K, the potential Φ in esu/cm (or called statvolt), and the bulk concentration M_i can be estimated by

$$M_i = \frac{N_A}{10^3}\bar{c}_i/\mathrm{cm}^3$$

for an ionic concentration being given in $\bar{c}_i$ mole per liter.

By the unit conversion formula 1 erg = 1 esu^2/cm, $\frac{e_c}{k_B T}$ is found to have unit cm/esu. Thus, by (2), the PBE models (14) can also be transformed as a dimensionless form as follows:

$$\begin{cases} -\epsilon_p \Delta u(\mathbf{r}) = \alpha_1 \sum_{j=1}^{n_p} z_j \delta_{\mathbf{r}_j}, & \mathbf{r} \in D_p, \\ \epsilon_s \Delta u(\mathbf{r}) + \beta_1 \sum_{i=1}^{n} Z_i \bar{c}_i e^{-Z_i u} = 0, & \mathbf{r} \in D_s, \\ u(\mathbf{s}^+) = u(\mathbf{s}^-), \quad \varepsilon_s \dfrac{\partial u(\mathbf{s}^+)}{\partial \mathbf{n}(\mathbf{s})} = \varepsilon_p \dfrac{\partial u(\mathbf{s}^-)}{\partial \mathbf{n}(\mathbf{s})}, & \mathbf{s} \in \Gamma, \\ u(\mathbf{s}) = \hat{g}(\mathbf{s}), & \mathbf{s} \in \partial\Omega, \end{cases} \tag{15}$$

where α_1 and β_1 are defined by

$$\alpha_1 = \frac{4\pi e_c^2}{k_B T}, \qquad \beta_1 = \frac{4\pi N_A e_c^2}{10^3 k_B T}. \tag{16}$$

Here the parameters e_c and k_B have different values from the ones listed in Table 1 due to electrostatic units.

Although the expressions of α_1 and β_1 are different from those of α and β, changing length units to angstrom and using the unit conversion formulas esu^2/erg = 1 cm, 1 erg = 1 dyn cm, 1 cm = 10^8 Å, 1 m = 10^{10} Å, we can verify that

$$\alpha_1 = \alpha, \quad \text{and} \quad \beta_1 = \beta.$$

Hence, either (5) or (15) can be used to calculate the dimensionless electrostatic potential u. For clarity, we will only consider (5) in the remaining parts of this paper.

3. PBE solution decompositions

To overcome the singular difficulty caused by the distributions $\delta_{\mathbf{r}_j}$, a solution decomposition for the PBE model (5) is presented in Theorem 3.1.

Theorem 3.1. *Let u be the solution of the PBE model (5). Then u is decomposed as*

$$u(\mathbf{r}) = G(\mathbf{r}) + \Psi(\mathbf{r}) + \tilde{\Phi}(\mathbf{r}) \quad \forall \mathbf{r} \in \Omega, \tag{17}$$

where G is given by

$$G(\mathbf{r}) = \frac{\alpha}{4\pi\epsilon_p} \sum_{j=1}^{n_p} \frac{z_j}{|\mathbf{r} - \mathbf{r}_j|}, \tag{18}$$

Ψ is a solution of the linear interface problem

$$\begin{cases} \Delta\Psi(\mathbf{r}) = 0, & \mathbf{r} \in D_p \cup D_s, \\ \Psi(\mathbf{s}^+) = \Psi(\mathbf{s}^-), \quad \varepsilon_s \dfrac{\partial \Psi(\mathbf{s}^+)}{\partial \mathbf{n}(\mathbf{s})} = \varepsilon_p \dfrac{\partial \Psi(\mathbf{s}^-)}{\partial \mathbf{n}(\mathbf{s})} + (\epsilon_p - \epsilon_s) \dfrac{\partial G(\mathbf{s})}{\partial \mathbf{n}(\mathbf{s})}, & \mathbf{s} \in \Gamma, \\ \Psi(\mathbf{s}) = \hat{g}(\mathbf{s}) - G(\mathbf{s}), & \mathbf{s} \in \partial\Omega, \end{cases} \tag{19}$$

and $\tilde{\Phi}$ is a solution of the nonlinear interface problem

$$\begin{cases} \Delta\tilde{\Phi}(\mathbf{r}) = 0, & \mathbf{r} \in D_p, \\ \epsilon_s \Delta\tilde{\Phi}(\mathbf{r}) + \beta \sum_{i=1}^{n} Z_i \bar{c}_i w_i(\mathbf{r}) e^{-Z_i \tilde{\Phi}(\mathbf{r})} = 0, & \mathbf{r} \in D_s, \\ \tilde{\Phi}(\mathbf{s}^+) = \tilde{\Phi}(\mathbf{s}^-), \quad \varepsilon_s \dfrac{\partial \tilde{\Phi}(\mathbf{s}^+)}{\partial \mathbf{n}(\mathbf{s})} = \varepsilon_p \dfrac{\partial \tilde{\Phi}(\mathbf{s}^-)}{\partial \mathbf{n}(\mathbf{s})}, & \mathbf{s} \in \Gamma, \\ \tilde{\Phi}(\mathbf{s}) = 0, & \mathbf{s} \in \partial\Omega. \end{cases} \tag{20}$$

Here α and β are given in (6), and $w_i(\mathbf{r})$ and $\frac{\partial G(\mathbf{s})}{\partial \mathbf{n}(\mathbf{s})}$ are defined by

$$w_i(\mathbf{r}) = e^{-Z_i(\Psi(\mathbf{r}) + G(\mathbf{r}))}, \qquad \frac{\partial G(\mathbf{s})}{\partial \mathbf{n}(\mathbf{s})} = -\frac{\alpha}{4\pi\epsilon_p} \sum_{j=1}^{n_p} z_j \frac{(\mathbf{s} - \mathbf{r}_j) \cdot \mathbf{n}}{|\mathbf{s} - \mathbf{r}_j|^3}. \tag{21}$$

Proof. Since $\frac{1}{4\pi|\mathbf{r}-\mathbf{r}_j|}$ satisfies the equation $-\Delta G_j = \delta_{\mathbf{r}_j}$ [42, p. 111],

$$\Delta G = \frac{\alpha}{\epsilon_p}\sum_{j=1}^{n_p} z_j \Delta \frac{1}{4\pi|\mathbf{r}-\mathbf{r}_j|} = -\frac{\alpha}{\epsilon_p}\sum_{j=1}^{n_p} z_j \delta_{\mathbf{r}_j}.$$

For $\mathbf{r}\in D_p$, we have $\Delta\Psi = 0$ and $\Delta\tilde{\Phi} = 0$. Thus,

$$\Delta u = \Delta\Psi + \Delta\tilde{\Phi} + \Delta G = -\frac{\alpha}{\epsilon_p}\sum_{j=1}^{n_p} z_j \delta_{\mathbf{r}_j} \quad \text{in } D_p.$$

Similarly, when $\mathbf{r}\in D_s$, it can be easily verified that u satisfies the second equation of (3) since $\Delta\Psi = 0$, $\Delta G = 0$, and

$$w_i(\mathbf{r})e^{-Z_i\tilde{\Phi}(\mathbf{r})} = e^{-Z_i(G+\Psi+\tilde{\Phi})(\mathbf{r})} = e^{-Z_i u}.$$

On the interface Γ, both $G(\mathbf{s})$ and $\frac{\partial G(\mathbf{s})}{\partial \mathbf{n}(\mathbf{s})}$ are continuous while Ψ and $\tilde{\Phi}$ satisfy the interface conditions given in (19) and (20), respectively. Thus, the function u defined by (17) satisfies the interface conditions of (3) as shown below:

$$u(s^+) = \tilde{\Phi}(s^+) + \Psi(s^+) + G(s) = \tilde{\Phi}(s^-) + \Psi(s^-) + G(s) = u(s^-) \quad \forall \mathbf{s}\in\Gamma,$$

and

$$\begin{aligned}
\epsilon_p \frac{\partial u(s^-)}{\partial \mathbf{n}(\mathbf{s})} &= \epsilon_p \frac{\partial \tilde{\Phi}(s^-)}{\partial \mathbf{n}(\mathbf{s})} + \epsilon_p \frac{\partial \Psi(s^-)}{\partial \mathbf{n}(\mathbf{s})} + \epsilon_p \frac{\partial G(s)}{\partial \mathbf{n}(\mathbf{s})} \\
&= \epsilon_s \frac{\partial \tilde{\Phi}(s^+)}{\partial \mathbf{n}(\mathbf{s})} + \epsilon_s \frac{\partial \Psi(s^+)}{\partial \mathbf{n}(\mathbf{s})} - (\epsilon_p - \epsilon_s)\frac{\partial G(\mathbf{s})}{\partial \mathbf{n}(\mathbf{s})} + \epsilon_p \frac{\partial G(s)}{\partial \mathbf{n}(\mathbf{s})} \\
&= \epsilon_s \frac{\partial}{\partial \mathbf{n}(\mathbf{s})}(\tilde{\Phi} + \Psi + G)(s^+) = \epsilon_s \frac{\partial u(s^+)}{\partial \mathbf{n}(\mathbf{s})} \quad \forall \mathbf{s}\in\Gamma.
\end{aligned}$$

Finally, on the boundary $\partial\Omega$, it is obvious that

$$u(\mathbf{s}) = \tilde{\Phi}(\mathbf{s}) + \Psi(\mathbf{s}) + G(\mathbf{s}) = 0 + \big(\hat{g}(\mathbf{s}) - G(\mathbf{s})\big) + G(\mathbf{s}) = \hat{g}(\mathbf{s}) \quad \forall \mathbf{s}\in\partial\Omega.$$

This completes the proof of Theorem 3.1. □

In Physics, G, Ψ, and $\tilde{\Phi}$ correspond to the electrostatic contributions from atomic charges from the solute domain D_p, interface and boundary conditions, and ionic charges from the solvent domain D_s, respectively. Thus, the sum $U = \Psi + G$ gives the electrostatic potential for biomolecule in water (i.e., a solution of the Poisson dielectric model (8)) while the sum $u = U + \tilde{\Phi}$ gives the electrostatic potential for biomolecule in ionic solvent (i.e., a solution of (5)). Because G contains all the singular points of u, Ψ and $\tilde{\Phi}$ are expected to be well defined without any singularity.

For the symmetric 1:1 ionic solution, the nonlinear interface problem (20) can be simplified as

$$\begin{cases}
\Delta\tilde{\Phi}(\mathbf{r}) = 0, & \mathbf{r}\in D_p, \\
-\epsilon_s\Delta\tilde{\Phi}(\mathbf{r}) + \kappa^2\sinh(\tilde{\Phi}+U) = 0, & \mathbf{r}\in D_s, \\
\tilde{\Phi}(\mathbf{s}^+) = \tilde{\Phi}(\mathbf{s}^-), \quad \varepsilon_s\dfrac{\partial\tilde{\Phi}(\mathbf{s}^+)}{\partial\mathbf{n}(\mathbf{s})} = \varepsilon_p\dfrac{\partial\tilde{\Phi}(\mathbf{s}^-)}{\partial\mathbf{n}(\mathbf{s})}, & \mathbf{s}\in\Gamma, \\
\tilde{\Phi}(\mathbf{s}) = 0, & \mathbf{s}\in\partial\Omega,
\end{cases} \tag{22}$$

where κ^2 is given in (11). Under the assumption (12), the above nonlinear interface problem can be linearized as

$$\begin{cases}
\Delta\tilde{\Phi}(\mathbf{r}) = 0, & \mathbf{r}\in D_p, \\
-\epsilon_s\Delta\tilde{\Phi}(\mathbf{r}) + \kappa^2\tilde{\Phi} = -\kappa^2 U, & \mathbf{r}\in D_s, \\
\tilde{\Phi}(\mathbf{s}^+) = \tilde{\Phi}(\mathbf{s}^-), \quad \varepsilon_s\dfrac{\partial\tilde{\Phi}(\mathbf{s}^+)}{\partial\mathbf{n}(\mathbf{s})} = \varepsilon_p\dfrac{\partial\tilde{\Phi}(\mathbf{s}^-)}{\partial\mathbf{n}(\mathbf{s})}, & \mathbf{s}\in\Gamma, \\
\tilde{\Phi}(\mathbf{s}) = 0, & \mathbf{s}\in\partial\Omega.
\end{cases} \tag{23}$$

Clearly, the sum of a solution of the above linear interface problem with U gives a LPBE solution decomposition since it can be easily verified to be a solution of the LPBE model (13).

4. PBE solution existence and uniqueness

Because of the solution decomposition (17), it only needs to consider (19) and (20) for the proof of the PBE solution existence and uniqueness. To do so, let $H^1(\Omega)$ denote the usual Sobolev function space and

$$H_0^1(\Omega) = \left\{ v \in H^1(\Omega) \;\middle|\; v = 0 \text{ on } \partial\Omega \right\}.$$

The norm of H^1 is defined by

$$\|v\|_{H^1(\Omega)} = \left(\|v\|_{L^2(\Omega)}^2 + \|\nabla v\|_{L^2(\Omega)}^2 \right)^{1/2},$$

where $\|v\|_{L^2(\Omega)} = (\int_\Omega |v(\mathbf{r})|^2 d\mathbf{r})^{1/2}$ is the norm of function space $L^2(\Omega)$ [43].

By the first Green's formula, (19) can be formulated into the weak form:

Find $\Psi \in H^1(\Omega)$ such that $\Psi = \hat{g} - G$ on $\partial\Omega$ and

$$a(\Psi, v) = (\epsilon_s - \epsilon_p) \int_\Gamma \frac{\partial G(\mathbf{s})}{\partial \mathbf{n}(\mathbf{s})} v(\mathbf{s}) d\mathbf{s} \quad \forall v \in H_0^1(\Omega), \tag{24}$$

where $a(u, v)$ is defined by

$$a(u, v) = \epsilon_p \int_{D_p} \nabla u \cdot \nabla v d\mathbf{r} + \epsilon_s \int_{D_s} \nabla u \cdot \nabla v d\mathbf{r}. \tag{25}$$

The solution existence and uniqueness of (19) is shown in Theorem 4.1.

Theorem 4.1. *Let $\mathcal{U} = H_0^1(\Omega) \cap H^2(D_p) \cap H^2(D_s)$ with the norm given by*

$$\|v\|_{\mathcal{U}} = \|v\|_{H^1(\Omega)} + \|v\|_{H^2(D_p)} + \|v\|_{H^2(D_s)}.$$

If the interface Γ is of class C^2, then the linear interface problem (19) has a unique weak solution $\Psi \in \mathcal{U}$ satisfying the weak form (24), and there exists a positive constant C such that

$$\|\Psi\|_{\mathcal{U}} \le C(\epsilon_s - \epsilon_p) \left\| \frac{\partial G}{\partial \mathbf{n}} \right\|_{H^{\frac{1}{2}}(\Gamma)}, \tag{26}$$

where $\frac{\partial G(\mathbf{s})}{\partial \mathbf{n}}$ is given in (21), and the norm $\|\cdot\|_{H^{\frac{1}{2}}(\Gamma)}$ is defined by

$$\|u\|_{H^{\frac{1}{2}}(\Gamma)} = \left(\|u\|_{L^2(\Gamma)}^2 + \int_\Gamma \int_\Gamma \frac{|u(\mathbf{s}) - u(\mathbf{s}')|^2}{|\mathbf{s} - \mathbf{s}'|^3} d\mathbf{s} d\mathbf{s}' \right)^{\frac{1}{2}}.$$

Proof. From (25) it can be seen that $a(u, v)$ is a bilinear functional on $H_0^1(\Omega)$. By $0 < \epsilon_p < \epsilon_s$ and Schwarz's inequality, the continuity of $a(u, v)$ can be shown as follows:

$$\begin{aligned} |a(u, v)| &\le (\epsilon_p + \epsilon_s) \int_\Omega |\nabla u \cdot \nabla v| d\mathbf{r} \\ &\le 2\epsilon_s \|\nabla u\|_{L^2(\Omega)} \|\nabla v\|_{L^2(\Omega)} \le 2\epsilon_s \|u\|_{H^1(\Omega)} \|v\|_{H^1(\Omega)}. \end{aligned} \tag{27}$$

By the Poincaré inequality [43, p. 62],

$$\begin{aligned} a(v, v) &= \epsilon_p \int_{D_p} |\nabla v|^2 d\mathbf{r} + \epsilon_s \int_{D_s} |\nabla v|^2 d\mathbf{r} \\ &\ge \epsilon_p \int_\Omega |\nabla v|^2 d\mathbf{r} \ge \frac{\epsilon_p}{C_1} \|v\|_{H^1(\Omega)}^2 \quad \forall v \in H_0^1(\Omega), \end{aligned} \tag{28}$$

where C_1 is a positive constant, from which it implies the coercivity of $a(u, v)$.

Since G is smooth in D_s, $\|\nabla G\|_{L^2(D_s)}$ is bounded. By the Green first identity and the equation $\Delta G = 0$ in D_s,

$$\begin{aligned} \int_\Gamma \frac{\partial G(\mathbf{s})}{\partial \mathbf{n}(\mathbf{s})} v(\mathbf{s}) d\mathbf{s} &= -\int_{D_s} \Delta G(\mathbf{r}) v(\mathbf{r}) d\mathbf{r} - \int_{D_s} \nabla G(\mathbf{r}) \cdot \nabla v(\mathbf{r}) d\mathbf{r} \\ &= -\int_{D_s} \nabla G(\mathbf{r}) \cdot \nabla v(\mathbf{r}) d\mathbf{r} \quad \forall v \in H_0^1(\Omega). \end{aligned} \tag{29}$$

Thus, by the above identity,

$$\left|\left\langle \frac{\partial G}{\partial \mathbf{n}}, v\right\rangle_{L^2(\Gamma)}\right| = \left|\int_{D_s} \nabla G \cdot \nabla v d\mathbf{r}\right| \le \|\nabla G\|_{L^2(D_s)} \|v\|_{H^1(\Omega)},$$

showing that $\langle \frac{\partial G}{\partial \mathbf{n}}, v\rangle_{L^2(\Gamma)}$ is a bounded linear functional on $H_0^1(\Omega)$. Hence, from the Lax–Milgram lemma [44] it implies that the weak form (24) has a unique solution in $H_0^1(\Omega)$.

Furthermore, from the smoothness of $\frac{\partial G(\mathbf{s})}{\partial \mathbf{n}(\mathbf{s})}$ on Γ it implies $\frac{\partial G}{\partial \mathbf{n}} \in H^{\frac{1}{2}}(\Gamma)$. By the regularity analysis of a general interface problem given in [45, Theorem 2.1] (or see [46,47]), it can be shown that $\Psi \in \mathcal{U}$, and satisfies (26). This completes the proof of Theorem 4.1. □

The surface integral $\int_\Gamma \frac{\partial G(\mathbf{s})}{\partial \mathbf{n}(\mathbf{s})} v(\mathbf{s}) d\mathbf{s}$ may become difficult to be evaluated for a complex molecular surface Γ. To avoid this difficulty, we can use (29) to reformulate (24) as a form involving volumetric integrals only:

$$a(\Psi, v) = (\epsilon_p - \epsilon_s) \int_{D_s} \nabla G(\mathbf{r}) \cdot \nabla v(\mathbf{r}) d\mathbf{r} \quad \forall v \in H_0^1(\Omega). \tag{30}$$

We next write the nonlinear interface problem (20) in the weak form: Find $\tilde{\Phi} \in \mathcal{H}$ such that

$$a(\tilde{\Phi}, v) - \beta \sum_{i=1}^{n} Z_i \bar{c}_i \int_{D_s} w_i(\mathbf{r}) e^{-Z_i \tilde{\Phi}(\mathbf{r})} v(\mathbf{r}) d\mathbf{r} = 0 \quad \forall v \in H_0^1(\Omega), \tag{31}$$

where $\mathcal{H}$ is a subset of $H_0^1(\Omega)$ defined by

$$\mathcal{H} = \left\{ v \in H_0^1(\Omega) \;\middle|\; \sum_{i=1}^{n} w_i e^{-Z_i v} \in L^2(D_s) \right\}. \tag{32}$$

We then introduce a variational minimization problem as follows: Find $\tilde{\Phi} \in \mathcal{H}$ such that

$$J(\tilde{\Phi}) = \inf_{v \in \mathcal{H}} J(v), \tag{33}$$

where $J : \mathcal{H} \to \mathbb{R}$ is a nonlinear functional defined by

$$J(v) = \frac{1}{2} a(v, v) + \beta \sum_{i=1}^{n} \bar{c}_i \int_{D_s} w_i(\mathbf{r}) e^{-Z_i v(\mathbf{r})} d\mathbf{r}. \tag{34}$$

Note that using the subset $\mathcal{H}$ is necessary since a function v of H^1 does not guarantee that each term $e^{-Z_i v} \in L^2(D_s)$.

The solution existence and uniqueness of (20) is presented in Theorem 4.2.

Theorem 4.2. *The nonlinear interface problem (20) has a unique weak solution $\tilde{\Phi} \in \mathcal{H} \cap H^2(D_p) \cap H^2(D_s)$ defined by the weak form (31), which is also a unique minimizer of the variational problem (33).*

Proof. We first show that the variational problem (33) has a unique minimizer. Similar to what is done in [48, p. 2549], it can be shown that $\mathcal{H}$ is a nonempty closed convex subset of $H_0^1(\Omega)$. The first and second Gâteaux derivatives of J at $u \in \mathcal{H}$, $J'(u)$ and $J''(u)$, which are continuous linear and bilinear functionals on H_0^1 and $H_0^1 \times H_0^1$, respectively, can be found as follows:

$$J'(u)v = a(u, v) - \beta \sum_{i=1}^{n} Z_i \bar{c}_i \int_{D_s} w_i e^{-Z_i u} v d\mathbf{r} \quad \forall v \in H_0^1(\Omega), \tag{35}$$

and

$$J''(u)(v, w) = a(v, w) + \beta \sum_{i=1}^{n} Z_i^2 \bar{c}_i \int_{D_s} w_i e^{-Z_i u} w v d\mathbf{r} \quad \forall v, w \in H_0^1(\Omega). \tag{36}$$

Since $w_i(\mathbf{r}) > 0$ on D_s, with (28), we get that

$$J(v) \ge \frac{\epsilon_p}{2C} \|v\|_{H^1(\Omega)}^2 \quad \forall v \in \mathcal{H},$$

and for any $u \in \mathcal{H}$,

$$J''(u)(v,v) \geq \frac{\epsilon_p}{C}\|v\|^2_{H^1(\Omega)} \quad \forall v \in H^1_0(\Omega),$$

from which it implies that $J(v) \to \infty$ as $\|v\|^2_{H^1(\Omega)} \to \infty$, and $J''(u)(v,v) > 0$ for all nonzero $v \in H^1_0(\Omega)$. Hence, J is coercive and strictly convex on $\mathcal{H}$. J is continuous on $\mathcal{H}$ since it is twice differentiable; thus, it is lower semicontinuous on $\mathcal{H}$. Therefore, from [49, Proposition 1.2, p. 35] it implies that (33) has a unique minimizer $\tilde{\Phi} \in \mathcal{H}$.

Clearly, from (35) it can imply that the minimizer $\tilde{\Phi}$ is a unique solution of the nonlinear weak form (31), and $\tilde{\Phi} \in \mathcal{H} \cap H^2(D_p) \cap H^2(D_s)$ when $\tilde{\Phi}$ satisfies the nonlinear interface problem (20). This completes the proof of Theorem 4.2. □

5. New PBE solution decomposition and minimization schemes

Let $\mathcal{M}$ be a Lagrange finite element space defined on a tetrahedral mesh of domain Ω such that $\mathcal{M}$ is a finite dimensional subspace of $H^1(\Omega)$, and each function of $\mathcal{M}$ is continuous. Set

$$\mathcal{M}_0 = \{v \in \mathcal{M} \mid v = 0 \text{ on } \partial\Omega\},$$

which is a subspace of $H^1_0(\Omega)$ and a subspace of $\mathcal{H}$ too. Thus, according to Theorems 3.1, 4.1, and 4.2, we can construct a solution decomposition scheme for calculating a PBE finite element solution in Algorithm 1.

Algorithm 1 *(PBE finite element solution decomposition scheme).* Let u be a finite element solution of the PBE model (5) on the Lagrange finite element space $\mathcal{M}$. Then, it can be calculated in four steps:

Step 1. Calculate G via (18) and $\nabla G(\mathbf{r}) = -\frac{\alpha}{4\pi\epsilon_p}\sum_{j=1}^{n_p} z_j \frac{\mathbf{r}-\mathbf{r}_j}{|\mathbf{r}-\mathbf{r}_j|^3}$ on $\mathcal{M}$.

Step 2. Find a finite element solution Ψ of the linear problem (19) on $\mathcal{M}$ such that $\Psi = \hat{g} - G$ on $\partial\Omega$, and

$$a(\Psi, v) = (\epsilon_p - \epsilon_s)\int_{D_s} \nabla G(\mathbf{r}) \cdot \nabla v(\mathbf{r}) d\mathbf{r} \quad \forall v \in \mathcal{M}_0. \tag{37}$$

Step 3. Find a finite element solution $\tilde{\Phi}$ of the nonlinear interface problem (20) on $\mathcal{M}_0$ by solving

$$J(\tilde{\Phi}) = \min_{v \in \mathcal{M}_0} J(v). \tag{38}$$

Step 4. Construct u by the solution decomposition:

$$u(\mathbf{r}) = \tilde{\Phi}(\mathbf{r}) + \Psi(\mathbf{r}) + G(\mathbf{r}) \quad \forall \mathbf{r} \in \Omega.$$

In Step 3, $\tilde{\Phi}$ can also be found directly from solving the nonlinear weak form (31) on $\mathcal{M}_0$. Typical nonlinear iterative methods for solving a system of nonlinear algebraic equations (such as a subspace trust-region Newton method [32,33] and a line search inexact Newton method [34]) can be applied to the numerical solution of (31). To achieve a global convergence, a merit function is usually set in the form

$$f(U) = \frac{1}{2}\sum_{i=1}^{n}\big[F_i(U)\big]^2, \tag{39}$$

for a nonlinear algebraic system in the form $F(U) = 0$. Here $F(U) = (F_1(U), F_2(U), \ldots, F_n(U))^t$, $U \in \mathbb{R}^n$, and $F_i : \mathbb{R}^n \to \mathbb{R}$ is a multivariable function [34]. Because of the equivalence between (31) and (33), the functional J has become "a natural merit function". Hence, a minimization scheme is selected to find $\tilde{\Phi}$ in Step 3 of Algorithm 1 since using a natural merit function can lead to a more effective and efficient nonlinear iterative solver than using an artificial merit function of (39) [14,34].

In particular, a modified Newton minimization scheme for solving (38) is constructed in Algorithm 2.

Algorithm 2 *(Modified Newton minimization scheme).* Let $\tilde{\Phi}^{(k)}$ denote the kth iterate of the modified Newton minimization scheme for solving (38) and an initial iterate, $\tilde{\Phi}^{(0)}$, be given. For $k = 0, 1, 2, \ldots$, the $(k+1)$th iterate $\tilde{\Phi}^{(k+1)}$ is defined by the following steps:

Step 1. The "blow-up" test: If $(-Z_i)(\Psi(\mathbf{r}) + G(\mathbf{r}) + \tilde{\Phi}^{(k)}(\mathbf{r})) > \tau$ for $\mathbf{r} \in D_s$, truncate the value of $e^{-Z_i(\Psi+G+v)}$ as e^{τ}. Here τ is a truncation parameter ($\tau = 85$ by default), and the modifications of J, J', and J'' are denoted by $\bar{J}$, $\bar{J}'$, and $\bar{J}''$, respectively.

Step 2. Find a descent search direction $p_k \in \mathcal{M}_0$ from solving the modified Newton bilinear variational form

$$\bar{J}''\big(\tilde{\Phi}^{(k)}\big)(p_k, v) = -\bar{J}'\big(\tilde{\Phi}^{(k)}\big)v \quad \forall v \in \mathcal{M}_0. \tag{40}$$

Step 3. Set steplength $\lambda_k = 1$.
Step 4. The iteration test: Accept λ_k and p_k and go to Step 6 if

$$\bar{J}(\tilde{\Phi}^{(k)} + \lambda_k p_k) \le \bar{J}(\tilde{\Phi}^{(k)}) \quad \text{or} \quad \|\bar{J}'(\tilde{\Phi}^{(k)} + \lambda_k p_k)\| \le \|\bar{J}'(\tilde{\Phi}^{(k)})\|. \tag{41}$$

Step 5. Determine another value of λ_k by a line search algorithm to satisfy (41).
Step 6. Define the update $\tilde{\Phi}^{(k+1)}$ by

$$\tilde{\Phi}^{(k+1)} = \tilde{\Phi}^{(k)} + \lambda_k p_k. \tag{42}$$

Step 7. The convergence test: Stop the iteration provided that

$$\|\bar{J}'(\tilde{\Phi}^{(k+1)})\| < \epsilon \quad \text{or} \quad \|\tilde{\Phi}^{(k+1)} - \tilde{\Phi}^{(k)}\| < \epsilon, \tag{43}$$

where ϵ is a convergence tolerance ($\epsilon = 10^{-12}$ by default).

Two choices of $\tilde{\Phi}^{(0)}$ are suggested for Algorithm 2. One is to set $\tilde{\Phi}^{(0)} = 0$, and the other one is to set $\tilde{\Phi}^{(0)}$ as a solution of the linear interface problem (23). These two choices lead to the solutions of the Poisson dielectric model (8) for protein in water and the LPBE model (13), respectively. Thus, they may be two good choices for Algorithm 2.

A search direction p_k is said to be a descent search direction if it satisfies the condition

$$\bar{J}'(\tilde{\Phi}^{(k)})p_k < 0. \tag{44}$$

In this paper, the PCG with the ILU preconditioning is considered to generate a descent search direction p_k in Step 2. That is, we stop the PCG iteration if its current iterate fails to satisfy the condition (44), and output the previous PCG iterate as p_k. In case the failure happens at the first PCG iterate, we set p_k to be a steepest descent direction.

To further reduce the costs of numerical calculations, truncated Newton strategies [38–40] can be employed in Step 2, along with a line search algorithm given in [50] to determine a steplength λ_k in Step 5 satisfying the Wolfe conditions

$$\bar{J}(\tilde{\Phi}^{(k)} + \lambda_k p_k) \le \bar{J}(\tilde{\Phi}^{(k)}) + \alpha\lambda_k \bar{J}'(\tilde{\Phi}^{(k)})p_k, \qquad \bar{J}'(\tilde{\Phi}^{(k)} + \lambda_k p_k)p_k \ge \beta \bar{J}'(\tilde{\Phi}^{(k)})p_k,$$

where $\alpha \in (0, \frac{1}{2})$, $\beta \in (\frac{1}{2}, 1)$. With such modifications, we can obtain a truncated modified Newton minimization method for solving (38).

The "Blow-up" Test of Step 1 is necessary to avoid a possible overflow problem caused by the exponential terms of J, J', and J'', since a large positive value of $Z_i(\Psi(\mathbf{r}) + G(\mathbf{r}) + \tilde{\Phi}^{(k)}(\mathbf{r}))$ may occur in D_s in a minimizer search process. With a sufficiently large τ, the test can be passed by all $\tilde{\Phi}^{(k)}$ near the minimizer. Thus, it should not affect the accuracy of a finite element solution.

Clearly, the modifications of J at $\tilde{\Phi}^{(k)}$ may be different from that at the update $\tilde{\Phi}^{(k+1)}$. This may cause $\bar{J}$ to be ascent even though J is descent. The Iteration Test of Step 4 reflects this case, which allows the iteration to be continued in this case if the gradient norm $\|\bar{J}'(\tilde{\Phi}^{(k)})\|$ is still reduced. When $\tilde{\Phi}^{(k)}$ is near the minimizer, we should have $\bar{J} = J$. Hence, the modified Newton minimization scheme eventually becomes a descent search method. Consequently, its convergence can be followed directly from the descent search minimization theory [34,35].

6. Program package and numerical results

A combination of Algorithm 1 with Algorithm 2 yields a new PBE finite element solver. As initial numerical studies, in this section, we only numerically test it for a protein immersed in a symmetric 1:1 ionic solution. In this case, the PBE model is given in (10), whose finite element solution can be constructed from Algorithm 1 with a finite element variation form of (22) being followed from (31) as follows: Find $\tilde{\Phi} \in \mathcal{M}_0$ such that

$$a(\tilde{\Phi}, v) + \kappa^2 \int_{D_s} \sinh(\tilde{\Phi} + U)v d\mathbf{r} = 0 \quad \forall v \in \mathcal{M}_0, \tag{45}$$

and $\bar{J}$, $\bar{J}'$, and $\bar{J}''$ can be found in the forms

$$\bar{J}(v) = \frac{1}{2}a(v, v) + \kappa^2 \int_{D_s} \widehat{\cosh}(v + U)d\mathbf{r} \quad \forall v \in \mathcal{M}_0, \tag{46}$$

$$\bar{J}'(\tilde{\Phi})v = a(\tilde{\Phi}, v) + \kappa^2 \int_{D_s} \widehat{\sinh}(\tilde{\Phi} + U)v d\mathbf{r} \quad \forall v \in \mathcal{M}_0, \tag{47}$$

$$\bar{J}''(\tilde{\Phi})(v, w) = a(v, w) + \kappa^2 \int_{D_s} \widehat{\cosh}(\tilde{\Phi} + U)v w d\mathbf{r} \quad \forall v, w \in \mathcal{M}_0, \tag{48}$$

where $\widehat{\sinh}$ and $\widehat{\cosh}$ denote modified hyperbolic sine and modified hyperbolic cosine, respectively, as defined by

$$\widehat{\sinh}(u)=\begin{cases}\sinh(u) & \text{if } |u|<\tau,\\ \sinh(\tau) & \text{if } u\geq\tau,\\ -\sinh(\tau) & \text{if } u\leq-\tau,\end{cases}\qquad \widehat{\cosh}(u)=\begin{cases}\cosh(u) & \text{if } |u|<\tau,\\ \cosh(\tau) & \text{if } |u|\geq\tau.\end{cases}\tag{49}$$

We developed a finite element program package for solving the PBE model (10) in Python based on the FEniCS finite element program library [36] and `GAMer` [37]. To speedup calculation, we wrote Fortran subroutines for calculating the mesh node values of G, ∇G, $\widehat{\sinh}(\tilde{\Phi}^{(k)}+\Psi+G)$, and $\widehat{\cosh}(\tilde{\Phi}^{(k)}+\Psi+G)$, and converted them as Python modules by the Fortran-to-Python interface generator `f2py` [51] (http://cens.ioc.ee/projects/f2py2e/). In addition, a Python parameter file is provided to control parameters from `DOLFIN` and `GAMer` such that a PQR file of protein is the only input file for our new PBE program package. Here, a PQR file is generated by the program tool PDB2PQR (http://www.poissonboltzmann.org/pdb2pqr/) [52] from a PDB file of a protein, which can be downloaded from the Protein Data Bank (PDB) (http://www.rcsb.org/), to get all the required data (such as atomic coordinates, charges, radii, and the hydrogen atoms missed in the PDB file) for the numerical solution of PBE.

To check the convergence rule (43), we calculated the following gradient norm, $\|\bar{J}'(v)\|$, at each iterate (i.e., $v=\tilde{\Phi}^{(k)}$) of our modified Newton minimization scheme:

$$\left\|\bar{J}'(v)\right\|=\sqrt{\sum_{j=1}^{N}\left[\bar{J}'(v)\phi_j\right]^2}\quad\text{for } v\in\mathcal{M}_0,\tag{50}$$

where N is the number of mesh nodes $\{\mathbf{r}^j\}$, and ϕ_j is a Lagrange interpolation polynomial of degree $m\geq 1$ satisfying that $\phi_j(\mathbf{r}^i)=0$ for $i\neq j$ and 1 for $i=j$.

In all the numerical experiments, we used $\epsilon_p=2$, $\epsilon_s=80$, the values of α and κ given in (11), $\tau=85$ for the "blow-up" test, and $\epsilon=10^{-12}$ for the convergence test of (43). Each involved system of linear algebraic equations was solved by PCG with ILU preconditioning until the relative residue error and the absolute residue error were less than 10^{-8}. All the tests were done on one 2.3 GHz Intel Core i7 of MacBook Pro with 8 GB 1600 MHz memory.

6.1. Numerical tests on a nonlinear Born ball model with analytical solution

To validate the solution decomposition scheme and our PBE program package, we made numerical tests on the following nonlinear Born ball model:

$$\begin{cases}-\epsilon_p\Delta u=\alpha z\delta & \text{in } D_p,\\ -\epsilon_s\Delta u+\kappa^2\sinh(u)=f & \text{in } D_s,\\ u(\mathbf{s}^+)=u(\mathbf{s}^-),\qquad \epsilon_s\dfrac{\partial u(\mathbf{s}^+)}{\partial\mathbf{n}}=\epsilon_p\dfrac{\partial u(\mathbf{s}^-)}{\partial\mathbf{n}} & \text{on } \Gamma,\\ u(\mathbf{s})=\hat{g}(\mathbf{s}) & \text{on } \partial\Omega,\end{cases}\tag{51}$$

where z is a charge number, α and κ^2 are given in (11), $D_p=\{\mathbf{r}\mid|\mathbf{r}|<a\}$, $D_s=\{\mathbf{r}\mid a<|\mathbf{r}|<A\}$, $\Gamma=\{\mathbf{r}\mid|\mathbf{r}|=a\}$, $\partial\Omega=\{\mathbf{r}\mid|\mathbf{r}|=A\}$, $\hat{g}(\mathbf{s})=\frac{\alpha z}{4\pi\epsilon_s|\mathbf{s}|}$, $f(\mathbf{r})=\kappa^2\sinh(\frac{\alpha z}{4\pi\epsilon_s|\mathbf{r}|})$, and the analytical solution u is given by

$$u(\mathbf{r})=\begin{cases}\frac{\alpha z}{4\pi a}\left(\frac{1}{\epsilon_s}-\frac{1}{\epsilon_p}\right)+\frac{\alpha z}{4\pi\epsilon_p|\mathbf{r}|} & \text{in } D_p,\\ \frac{\alpha z}{4\pi\epsilon_s|\mathbf{r}|} & \text{in } D_s.\end{cases}\tag{52}$$

In the numerical tests, we set $a=1$, $A=10$, and $z=1$, and used a tetrahedral mesh with 2955 vertices and 17 357 tetrahedra (see Fig. 1). The mesh excludes the origin point as a mesh node to avoid the singularity of $G(\mathbf{r})$ at the origin. To reflect the case of the Born model, we modified our program package to subtract the term $\int_{D_s}f(\mathbf{r})v d\mathbf{r}$ from the expressions (47) and (46) of $\bar{J}'$ and $\bar{J}$, respectively. An initial guess $\tilde{\Phi}^{(0)}=0$ was used in each test.

The numerical results are reported in Table 2. A comparison of the finite element solution u_h with the analytical solution u is displayed in Fig. 2. From Table 2 it can be seen that the accuracy of a finite element solution can be improved significantly as the order m of the finite element method is increased from 1 to 3. As shown in Fig. 2, the finite element solution ($m=2$) matches the analytical solution very well. These numerical results well validated the PBE solution decomposition scheme and our PBE finite element program package.

The numerical results of Table 2 confirmed the efficiency of our modified Newton minimization scheme and PBE program package too. For example, a finite element solution was found in 5 iterations and about 8 seconds for a system of nonlinear quadratic finite element equations defined on a mesh with 23 522 mesh nodes. It is also interesting to see that the starting gradient norm is reduced sharply when the finite element order m is increased from 1 to 2, resulting in a sharp reduction of the total number of iterations from 29 to 5.

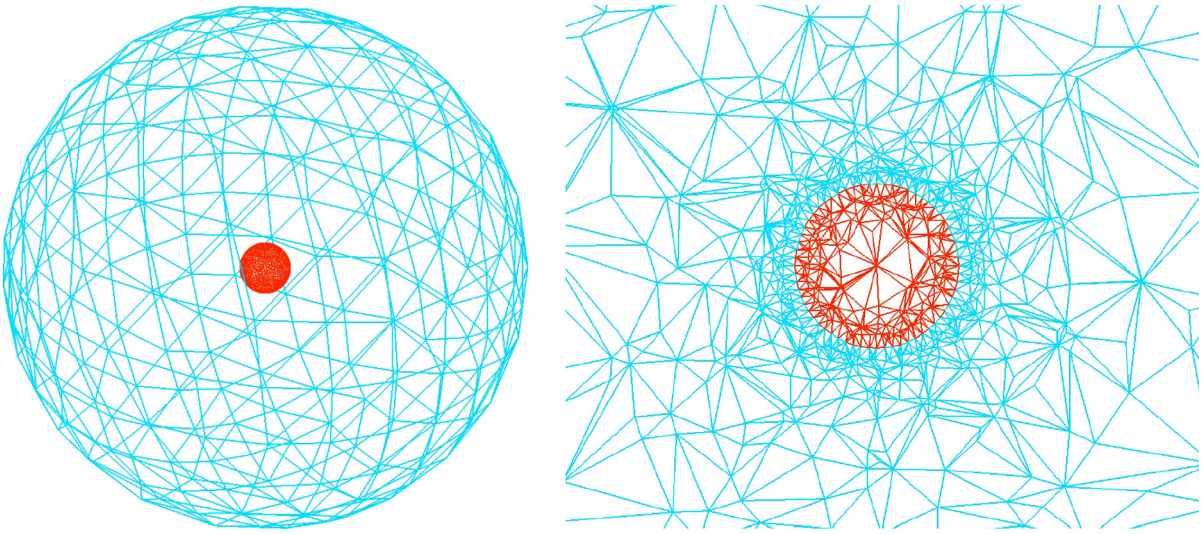

Fig. 1. A mesh partition of the spheric domain Ω with $a = 1$ and $A = 10$. Here the meshes of D_p and D_s are marked in red and cyan, respectively. A part of cross section of the mesh is displayed in the right figure. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 2
Performance of our PBE finite element program package for solving the nonlinear Born ball model (51) with $a = 1$, $A = 10$, and $z = 1$. Here u and u_h denote the analytical and finite element solutions, respectively, and the numbers in parentheses are the number of mesh nodes of the mth order finite element method (FEM).

FEM order m	$m = 1$ (2955)	$m = 2$ (23 522)	$m = 3$ (79 059)
$\|u - u_h\|_{L^2(\Omega)}$	2.3897	1.816×10^{-1}	4.168×10^{-2}
Start $\|\bar{J}'(\tilde{\Phi}^{(0)})\|$	7.573×10^{8}	1.479×10^{1}	1.691×10^{-1}
Final $\|\bar{J}'(\tilde{\Phi}^{(k)})\|$	3.275×10^{-8}	6.066×10^{-9}	6.036×10^{-9}
Total iteration number for solving (38)	29	5	4
Total CPU time (sec.)	2.18	7.77	68.53

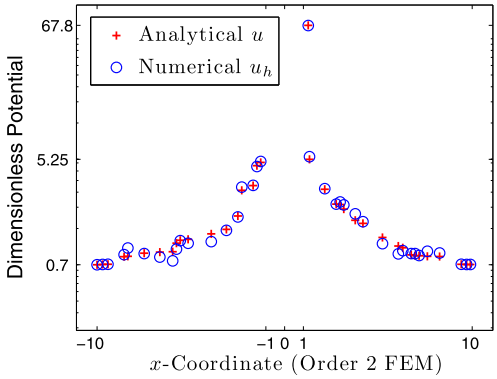

Fig. 2. A comparison of the quadratic finite element solution (in blue circle) with the analytical solution (in red plus) for the nonlinear Born ball model (51) with $a = 1$ and $A = 10$. The nodes are selected by considering their x-coordinates in a step-length of 0.2 with their magnitudes decreasing on $[-10, 0]$ and increasing on $[0, 10]$. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

6.2. Numerical tests on proteins

To demonstrate the effectiveness and efficiency of our PBE finite element program package, numerical experiments were made on four proteins represented in the PDB files 4PTI, 1CID, and 2AQ5 and the PQR file FAS2.pqr. The linear interface problem (22) was selected to generate initial iterates $\tilde{\Phi}^{(0)}$. By using `GAMer`, the domains Ω for these four proteins were generated as the spherical balls with radii 461.90, 463.72, 749.90, and 720.16 Å and centers $(15.29, 20.81, 4.60)$, $(-1.57, 0.65, 24.67)$, $(-9.76, 38.18, 32.80)$, and $(14.81, 53.14, 21.21)$, respectively, together with the protein domains D_p and the tetrahedral meshes with the number N of vertices being 33 572, 389 26, 84 415, and 107 400.

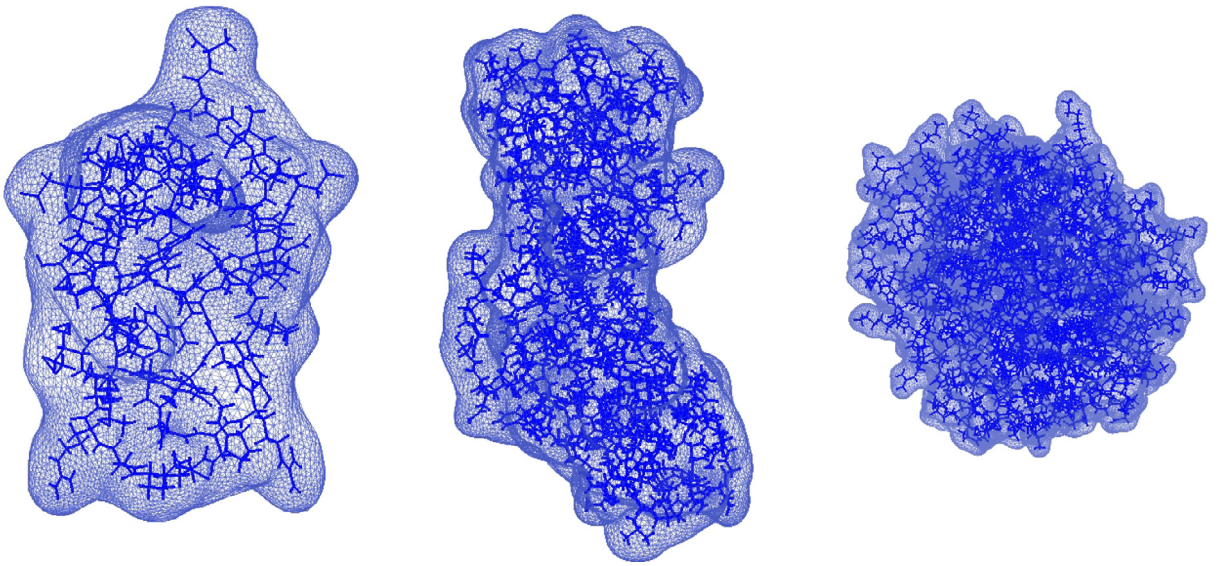

Fig. 3. Protein domains D_p and their surface meshes generated by GAMer for proteins represented in 4PTI, 1CID, and 2AQ5 PDB files. Here the molecular structures of the proteins are represented in solid lines.

Table 3
Performance of our modified Newton minimization scheme for solving the variational minimization problem (38) for four proteins represented in 4PTI ($N = 33\,572$, $n_p = 892$), FAS2 ($N = 389\,26$, $n_p = 906$), 1CID ($N = 84\,415$, $n_p = 2783$), and 2AQ5 ($N = 107\,400$, $n_p = 6024$). Here the CPU time is measured in seconds.

Protein	Start $\bar{J}(\tilde{\Phi}^{(0)})$	Final $\bar{J}(\tilde{\Phi}^{(k)})$	Start $\|\bar{J}'(\tilde{\Phi}^{(0)})\|$	Final $\|\bar{J}'(\tilde{\Phi}^{(k)})\|$	Total iterations	CPU time
4PTI	3.78×10^9	3.52×10^8	9.69×10^5	1.89×10^{-6}	18	11.17
FAS2	3.59×10^8	3.55×10^8	1.56×10^5	2.10×10^{-6}	15	11.39
1CID	1.56×10^9	1.49×10^9	5.32×10^6	2.73×10^{-6}	24	48.59
2AQ5	4.44×10^9	1.32×10^9	1.40×10^9	1.31×10^{-6}	33	90.45

Fig. 3 displays three protein domains as examples to show that D_p has a very complex shape, whose boundary gives the interface Γ, making a finite element mesh generation to be a challenging task. From the figure it can also be seen that a three-dimensional molecular structure of each protein was wrapped well by D_p, showing that the protein domains D_p generated from GAMer are satisfactory for our numerical experiments.

Table 3 reports the performance of our modified Newton minimization scheme for solving the variational minimization problem (38) for the four proteins. In these tests, the convergence tolerance of (43) was set as $\epsilon = 10^{-12}$. From this table it can be seen that our modified Newton minimization scheme worked effectively and efficiently. For example, in the case of 4PTI, it took only about 11 seconds to find a minimizer of (38) on a mesh with 38 572 vertices. Even for a protein molecule (2AQ5) with 6062 atoms on a mesh with 107 400 vertices and 665 297 tetrahedra, a minimum solution was found in only 90.45 seconds while the gradient norm was reduced from 1.40×10^9 to 1.31×10^{-6} in 33 iterations.

Fig. 4 displays the convergence behavior of our modified Newton minimization scheme for the four protein cases. From Fig. 4 we see that the gradient norm $\|\bar{J}'(\tilde{\Phi}^{(k)})\|$, which is defined in (50), the solution difference $\|\tilde{\Phi}^{(k)} - \tilde{\Phi}^{(k+1)}\|$, and the relative error of the target functional of the variational minimization problem (38), which is defined as $|\bar{J}(\tilde{\Phi}^{(k+1)}) - \bar{J}(\tilde{\Phi}^{(k)})|/|\bar{J}(\tilde{\Phi}^{(k+1)})|$, were reduced quickly, numerically confirming the convergence of our modified Newton minimization scheme. Here the differences $\|\tilde{\Phi}^{(k)} - \tilde{\Phi}^{(k+1)}\|$ were almost zero at the last iteration due to that the last two vectors of $-\bar{J}'(\tilde{\Phi}^{(k)})$ (the right hand side term of the Newton equation (40)) were almost identical. They were plotted approximately as $O(10^{-14})$ to be displayed in the logarithmic scale.

Table 4 reports the performance of our finite element program package in terms of CPU time. Here the total time includes the CPU time spent on data input, data output, and mesh generation. From this table it can be seen that all the linear finite element equations were efficiently solved by the PCG using the ILU preconditioning, which only took about 3 to 11 seconds. However, the CPU time spent on mesh generation almost took a half of the total time. It was increased from about 12 seconds to 106 seconds when the number of atoms of a protein was increased from 892 to 6024. This indicates that mesh generation is the bottleneck in the finite element solution of the PBE model.

In the future, we will carry out further studies on our PBE solution decomposition and minimization schemes, and develop more efficient linear and nonlinear iterative solvers for solving (19) and (38). We will also program the truncated

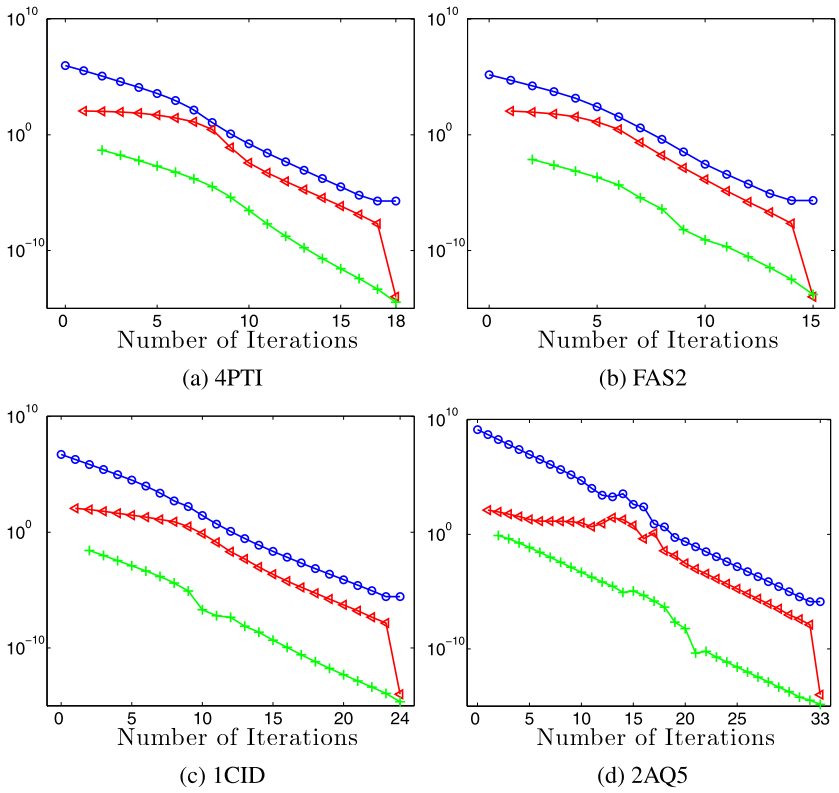

Fig. 4. Convergences of the modified Newton minimization scheme for solving the variational minimization problem (38) in terms of the gradient norm $\|\bar{J}'(\tilde{\Phi}^{(k)})\|$ (in blue circles), the difference $\|\tilde{\Phi}^{(k)} - \tilde{\Phi}^{(k+1)}\|$ (in red triangles), and the relative error $|\bar{J}(\tilde{\Phi}^{(k+1)}) - \bar{J}(\tilde{\Phi}^{(k)})|/|\bar{J}(\tilde{\Phi}^{(k+1)})|$ (in green plus). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 4
Performance (in CPU time in seconds) of our PBE finite element program package on one 2.3 GHz Intel Core i7 processor of a MacBook Pro.

Protein	Find mesh	Find G & ∇G	Solver (37) for Ψ	Solve (38) for $\tilde{\Phi}$	Total time
4PTI	11.969	0.678	2.735	11.17	31.173
FAS2	16.03	0.7987	3.272	11.398	37.071
1CID	25.785	5.1563	7.658	48.594	99.269
2AQ5	106.372	14.212	10.693	90.456	238.315

modified Newton minimization scheme in an efficient way to further improve the performance of our finite element program package.

Acknowledgements

This work was partially supported by the National Science Foundation, USA, through grant DMS-1226259, and the UWM Research Growth Initiative.

References

[1] B. Honig, A. Nicholls, Classical electrostatics in biology and chemistry, Science 268 (1995) 1144–1149.
[2] M. Holst, The Poisson–Boltzmann equation: analysis and multilevel numerical solution, online, 1994.
[3] 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.
[4] B. Lu, Y. Zhou, M. Holst, J. McCammon, Recent progress in numerical methods for the Poisson–Boltzmann equation in biophysical applications, Commun. Comput. Phys. 3 (5) (2008) 973–1009.
[5] M.E. Davis, J.A. McCammon, Solving the finite difference linearized Poisson–Boltzmann equation: a comparison of relaxation and conjugate gradient methods, J. Comput. Chem. 10 (1989) 386–391.
[6] 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.
[7] B. Roux, T. Simonson, Implicit solvent models, Biophys. Chem. 78 (1999) 1–20.

[8] N.A. Baker, D. Sept, S. Joseph, M. Holst, J.A. McCammon, Electrostatics of nanosystems: application to microtubules and the ribosome, Proc. Natl. Acad. Sci. USA 98 (18) (2001) 10037–10041.
[9] 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.
[10] M. Holst, F. Saied, Numerical solution of the nonlinear Poisson–Boltzmann equation: developing more robust and efficient methods, J. Comput. Chem. 16 (1995) 337–364.
[11] N. Baker, M. Holst, F. Wang, Adaptive multilevel finite element solution of the Poisson–Boltzmann equation II. Refinement at solvent-accessible surfaces in biomolecular systems, J. Comput. Chem. 21 (15) (2000) 1343–1352.
[12] W. Geng, R. Krasny, A treecode-accelerated boundary integral Poisson–Boltzmann solver for electrostatics of solvated biomolecules, J. Comput. Phys. 247 (2013) 62–78.
[13] B. Lu, X. Cheng, J.A. McCammon, New-version-fast-multipole-method accelerated electrostatic interactions in biomolecular systems, J. Comput. Phys. 226 (2007) 1348–1366.
[14] D. Xie, S. Zhou, A new minimization protocol for solving nonlinear Poisson–Boltzmann mortar finite element equation, BIT Numer. Math. 47 (2007) 853–871.
[15] N. Smith, S. Witham, S. Sarkar, J. Zhang, L. Li, C. Li, E. Alexov, DelPhi web server v2: incorporating atomic-style geometrical figures into the computational protocol, Bioinformatics 28 (12) (2012) 1655–1657.
[16] D. Bashford, An object-oriented programming suite for electrostatic effects in biological molecules an experience report on the mead project, in: Scientific Computing in Object-Oriented Parallel Environments, Springer, 1997, pp. 233–240.
[17] N. Baker, D. Sept, M. Holst, J.A. McCammon, The adaptive multilevel finite element solution of the Poisson–Boltzmann equation on massively parallel computers, IBM J. Res. Dev. 45 (2001) 427–438.
[18] S. Unni, Y. Huang, R.M. Hanson, M. Tobias, S. Krishnan, W.W. Li, J.E. Nielsen, N.A. Baker, Web servers and services for electrostatics calculations with APBS and PDB2PQR, J. Comput. Chem. 32 (7) (2011) 1488–1491.
[19] M.-J. Hsieh, R. Luo, Physical scoring function based on amber force field and Poisson–Boltzmann implicit solvent for protein structure prediction, Proteins, Struct. Funct. Bioinform. 56 (3) (2004) 475–486.
[20] D.A. Case, T.E. Cheatham, T. Darden, H. Gohlke, R. Luo, K.M. Merz, A. Onufriev, C. Simmerling, B. Wang, R.J. Woods, The amber biomolecular simulation programs, J. Comput. Chem. 26 (16) (2005) 1668–1688.
[21] W. Im, D. Beglov, B. Roux, Continuum solvation model: electrostatic forces from numerical solutions to the Poisson–Boltzmann equation, Comput. Phys. Commun. 111 (1998) 59–75.
[22] B.R. Brooks, R.E. Bruccoleri, B.D. Olafson, D.J. States, S. Swaminathan, M. Karplus, CHARMM: a program for macromolecular energy, minimization, and dynamics calculations, J. Comput. Chem. 4 (1983) 187–217.
[23] S. Jo, M. Vargyas, J. Vasko-Szedlar, B. Roux, W. Im, PBEQ-solver for online visualization of electrostatic potential of biomolecules, Nucleic Acids Res. 36 (suppl. 2) (2008) W270–W275.
[24] J.C. Phillips, R. Braun, W. Wang, J. Gumbart, E. Tajkhorshid, E. Villa, C. Chipot, R.D. Skeel, L. Kalé, K. Schulten, Scalable molecular dynamics with NAMD, J. Comput. Chem. 26 (16) (2005) 1781–1802, http://dx.doi.org/10.1002/jcc.20289.
[25] M. Neves-Petersen, S. Petersen, Protein electrostatics: a review of the equations and methods used to model electrostatic equations in biomolecules – applications in biotechnology, Biotechnol. Annu. Rev. 9 (2003) 315–395.
[26] M. Perutz, Electrostatic effects in proteins, Science 201 (1978) 1187–1191.
[27] C.L. Vizcarra, S.L. Mayo, Electrostatics in computational protein design, Curr. Opin. Chem. Biol. 9 (2005) 622–626.
[28] I. Chern, J.-G. Liu, W.-C. Wang, Accurate evaluation of electrostatics for macromolecules in solution, Methods Appl. Anal. 10 (2) (2003) 309–328.
[29] L. Chen, M.J. Holst, J. Xu, The finite element approximation of the nonlinear Poisson–Boltzmann equation, SIAM J. Numer. Anal. 45 (6) (2007) 2298–2320.
[30] 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. 17 (11) (1996) 1344–1351.
[31] Y. Zhou, M. Holst, J.A. McCammon, A nonlinear elasticity model of macromolecular conformational change induced by electrostatic forces, J. Math. Anal. Appl. 340 (1) (2008) 135–164.
[32] T. Coleman, Y. Li, An interior, trust region approach for nonlinear minimization subject to bounds, SIAM J. Optim. 6 (1996) 418–445.
[33] T. Coleman, Y. Li, On the convergence of reflective Newton methods for large-scale nonlinear minimization subject to bounds, Math. Program. 67 (1994) 189–224.
[34] J. Nocedal, S. Wright, Numerical Optimization, 2nd edition, Springer Series in Operations Research and Financial Engineering, Springer, New York, 2006.
[35] J.E. Dennis Jr., R.B. Schnabel, Numerical Methods for Unconstrained Optimization and Nonlinear Equations, Classics in Applied Mathematics, vol. 16, SIAM, Philadelphia, PA, 1996.
[36] A. Logg, G.N. Wells, J. Hake, DOLFIN: a C++/Python finite element library, in: Automated Solution of Differential Equations by the Finite Element Method, in: Lect. Notes Comput. Sci. Eng., vol. 84, Springer, Heidelberg, 2012, pp. 173–225, Ch. 10.
[37] Z. Yu, M. Holst, Y. Cheng, J. McCammon, Feature-preserving adaptive mesh generation for molecular shape modeling and simulation, J. Mol. Graph. Model. 26 (8) (2008) 1370–1380.
[38] T. Schlick, A. Fogelson, TNPACK — a truncated Newton minimization package for large-scale problems: I. Algorithm and usage, ACM Trans. Math. Softw. 14 (1992) 46–70.
[39] D. Xie, T. Schlick, Remark on Algorithm 702 – the updated truncated Newton minimization package, ACM Trans. Math. Softw. 25 (1) (1999) 108–122.
[40] D. Xie, T. Schlick, Efficient implementation of the truncated Newton method for large-scale chemistry applications, SIAM J. Optim. 10 (1) (1999) 132–154.
[41] W. Rudin, Functional Analysis, 2nd edition, McGraw-Hill, New York, 1991.
[42] R.C. McOwen, Partial Differential Equations: Methods and Applications, 2nd edition, Prentice Hall, Upper Saddle River, NJ, 2003.
[43] S. Brenner, L. Scott, The Mathematical Theory of Finite Element Methods, 3rd edition, Springer-Verlag, New York, 2008.
[44] L. Debnath, P. Mikusiński, Hilbert Spaces with Applications, Academic Press, 2005.
[45] Z. Chen, J. Zou, Finite element methods and their convergence for elliptic and parabolic interface problems, Numer. Math. 79 (1998) 175–202.
[46] J. Bramble, J. King, A finite element method for interface problems in domains with smooth boundaries and interfaces, Adv. Comput. Math. 6 (1) (1996) 109–138.
[47] G. Savaré, Regularity results for elliptic equations in Lipschitz domains, J. Funct. Anal. 152 (1) (1998) 176–201.
[48] B. Li, Minimization of electrostatic free energy and the Poisson–Boltzmann equation for molecular solvation with implicit solvent, SIAM J. Math. Anal. 40 (6) (2009) 2536–2566.
[49] I. Ekeland, R. Témam, Convex Analysis and Variational Problems, SIAM, Philadelphia, 1999.
[50] J.J. Moré, D.J. Thuente, Line search algorithms with guaranteed sufficient decrease, ACM Trans. Math. Softw. 20 (1994) 286–307.
[51] P. Peterson, F2PY: a tool for connecting Fortran and Python programs, Int. J. Comput. Sci. Eng. 4 (4) (2009) 296–305.
[52] T. Dolinsky, J. Nielsen, J. McCammon, N. Baker, PDB2PQR: an automated pipeline for the setup of Poisson–Boltzmann electrostatics calculations, Nucleic Acids Res. 32 (suppl. 2) (2004) W665.