EFFICIENT ALGORITHMS FOR A NONLOCAL DIELECTRIC MODEL FOR PROTEIN IN IONIC SOLVENT *

DEXUAN XIE†, YI JIANG† , AND L. RIDGWAY SCOTT ‡

Abstract. The nonlocal dielectric approach can significantly enhance the classical Poisson dielectric model by including polarization correlations among water molecules. In this paper, a nonlocal dielectric model for protein in ionic solvent is proposed and analyzed, alongside a new efficient numerical algorithm and program package for solving the model. In particular, by using solution splitting and reformulation techniques, it is shown that the solution of the nonlocal dielectric model is unique, and can be found from solving two well-defined partial differential systems and one Poisson-like boundary value problem. Consequently, the singular and computational difficulties caused by Dirac delta distributions and convolution terms are overcome. Furthermore, a nonlocal linearized Poisson-Boltzmann equation with uniform ionic size effect is proposed and numerically tested on three protein molecules with up to 6062 atoms by using a fast finite element solver from the FEniCS project. A nonlocal point charge Born model with a known analytical solution is also tested to validate the new algorithm and program package. Numerical results demonstrate the high performance of the program package, and confirm one advantage of the new algorithm in retaining a high order of accuracy of finite element approximations.

Key words. nonlocal dielectric continuum model, electrostatic potential, Poisson-Boltzmann equation, finite element solvers, FEniCS project, protein simulation

AMS subject classifications. 35Q92, 92-08 65N30 92C40

1. Introduction.

Calculation of electrostatic potential energy for proteins in ionic solvent is a fundamental task in the simulation study of the structure and biological function of proteins, catalytic activity, and ligand association [17, 25, 31]. One commonly used mathematical model for estimating the electrostatic potential function in an ionic solvent is the Poisson-Boltzmann equation (PBE) [2, 13, 16, 22, 23, 27, 37]. But, due to the polarization correlations among water molecules and ionic size effect [21], the PBE model may not work well in some important bioengineering applications (such as ion channel studies and rational drug design). To reflect the polarization correlations among water molecules, several nonlocal dielectric models have been developed for a wide range of dielectric materials and dipolar liquids in the last thirty years [4, 5, 6, 7, 8, 10, 19, 20, 28, 30]. Recent progress in the development of fast numerical algorithms has sharply reduced the complexity of solving a nonlocal dielectric model [15, 33, 35, 36], making it possible for a nonlocal dielectric model to be applied to large scale biomolecular simulations.

However, the study of a nonlocal model has been limited to the case of pure water solvent so far due to modeling and algorithmic complications. Most significantly,

*This work was partially supported by the National Science Foundation, USA, through grants DMS-0921004, 1226259, and 1226019.

†Department of Mathematical Sciences. University of Wisconsin-Milwaukee, Milwaukee, Wisconsin, USA, 53201-0413 (dxie@uwm.edu)

‡ Department of Computer Science, University of Chicago, Chicago, IL 60637

none of the current ionic models incorporate nonlocal dielectric effects. As the first step toward the direction of changing this situation, in this paper, we propose a nonlocal dielectric model for protein in ionic solvent (see (2.2)) by assuming that ionic concentration functions are given. For clarity, we call it the nonlocal Poisson dielectric model since it includes the classic Poisson dielectric model as a special case.

Because of its simplicity, the nonlocal Poisson dielectric model makes us focus on the study of two fundamental issues related to nonlocal dielectric modeling: One is how to deal with the solution singularity caused by Dirac delta distributions while the other one is how to simplify the computational complexity of a nonlocal model with convolution terms. In our recent work, the second issue was successfully tackled with a novel reformulation of a nonlocal model as a partial differential equation (PDE) system in the case of pure water [35], but how to deal with the case of protein in ionic solvent is still an open problem. Through studying these two important issues, we intend to strengthen the theoretical and numerical basis as required for our future work — the development of advanced nonlocal dielectric models based on the approach of a constrained electrostatic free energy minimization. In such a minimization process, the nonlocal Poisson dielectric model is set as one of the constraints. Hence, the results reported in this paper will be particularly valuable in the future work.

In particular, to overcome the singular difficulties caused by Dirac delta distributions, we propose and prove a solution splitting formula such that the solution of the nonlocal Poisson dielectric model can be found through solving two nonlocal models without any singular difficulty (see Theorem 3.1). The computing difficulties caused by convolution terms are then overcome through a novel reformulation of each related nonlocal model as a PDE system (see (4.4), (4.6), and (4.8)) or a Poisson-like boundary value problem (see (4.10)). Next, these PDE systems and Poisson-like boundary value problem are reformulated as variational problems without involving any surface integral defined on the interface between the protein and the solvent regions. In this way, the difficulty of analyzing and programming surface integrals is avoided. Furthermore, these PDE problems are theoretically proved to have unique solutions (see Theorem 5.1). Consequently, a fast numerical algorithm is derived for solving the nonlocal Poisson dielectric model.

Clearly, different selections of ionic concentration functions may generate different nonlocal dielectric models from the nonlocal Poisson dielectric model. In this paper, we construct ionic concentration functions using the solution of the local linearized PBE (LPBE) model with uniform ionic size effect [22]. Here all ions and water molecules are assumed to have a uniform volume to reflect ionic size effects, which are important in many biological applications. With such particular ionic concentration functions, the nonlocal Poisson dielectric model can be regarded as *a nonlocal LPBE model with uniform ionic size effect*, since it is an extension of the local LPBE model. This particular nonlocal model is also a good one for us to study the performance of our fast numerical algorithm for a protein in ionic solvent.

We programmed our fast algorithm for solving the nonlocal LPBE model as a Python program package based on the finite element program library `DOLFIN` [1]. To generate a tetrahedral finite element mesh for a given protein molecule in a PQR file, we adapted and converted the mesh generation program package `GAMer` [38] as a Python module of our program package using the software development tool `SWIG` (*http://www.swig.org*). As a result, the mesh data generated from `GAMer` can be directly used to construct a mesh object of `DOLFIN`, and stored in XML format – a format recognized in `DOLFIN`. We also wrote a Fortran program for computing the mesh values of the singular function G (see (3.2)) of our solution splitting formula and its gradient ∇G, and converted it into a Python module using the Fortran to Python interface generator `f2py` [26] (*http://cens.ioc.ee/projects/f2py2e/*). With such a Python module, we constructed finite element interpolation functions for G and ∇G in a special way to significantly speed up the assembling process of a linear form involving G or ∇G in a finite element approximation.

To validate our new algorithm and program package, we made numerical experiments on a nonlocal point charge Born model, whose analytical solution is available in [36, Pages 182-183]. As comparison, we also solved this Born model in a customary way in which the Dirac delta distribution was approximated by using `DOLFIN`'s numerical delta function `PointSource`. In numerical tests, we solved this nonlocal Born model approximately by linear, quadratic, and cubic finite element methods, respectively. Numerical results not only validate our algorithm and program package but also confirm one advantage of our new algorithm in retaining a higher order of accuracy of a finite element solution.

We then did numerical tests on three proteins with the number of atoms up to 6062 in ionic solvent containing two different kinds of ions – sodium (N_a^+) and chloride (Cl^-) ions. All the numerical experiments were made on one 2.3 GHz Intel Core i7 MacBook Pro with 8 GB 1600 MHz DDR3 memory. In the tests of protein BPTI (892 atoms and 69,982 mesh vertices), a linear finite element solution of the nonlocal LPBE model was found in 0.84 minutes only, which is the total CPU time starting from reading a protein structure PQR file and including the time spent on the finite element mesh generation. For a protein with 6062 atoms on a mesh with 158,908 vertices, the total CPU time was 4.1 minutes. These numerical results demonstrate the high efficiency of our new algorithm and program package.

The remaining part of the paper is arranged as follows. In Section 2, we introduce the nonlocal Poisson dielectric model. In Section 3, we present the solution splitting formula. In Section 4. we reformulate each related nonlocal model into a PDE system. In Section 5, we obtain the weak form of each PDE system and prove its solution existence and uniqueness. Finally, the fast algorithm, the nonlocal LPBE model, the program package, and the numerical results are presented in Section 6.

2. The nonlocal Poisson dielectric model. Let the whole space $\mathbb{R}^3$ be split into two disjoint open domains, D_p and D_s, satisfying that D_p is surrounded by D_s,

and $\mathbb{R}^3 = D_p \cup D_s \cup \Gamma$. Here Γ is the interface between D_p and D_s, D_p hosts a protein molecule with n_p atoms, and D_s contains the ionic solvent with n different types of ions. As an implicit solvent approach [27, 32], D_p and D_s are treated as two different dielectric continuum media with two different dielectric constants, ϵ_p and ϵ_s, satisfying $\epsilon_s > \epsilon_p > 0$, respectively. We also denote by ϵ_∞ the permittivity factor for water in the limit of high frequency [34], and by ϵ_0 the permittivity of the vacuum.

When a molecular structure of a protein and a concentration function $c_i(\mathbf{r})$ of ionic species i are given, the charge density function ρ can be estimated by

$$\rho(\mathbf{r}) = \sum_{j=1}^{n_p} Q_j \delta_{\mathbf{r}_j} + \sum_{i=1}^{n} q_i c_i(\mathbf{r}), \quad \mathbf{r} \in \mathbb{R}^3, \tag{2.1}$$

where each c_i has been extended as zero in the protein domain D_p, Q_j and $\mathbf{r}_j$ are the charge and position of atom j of the protein molecule, respectively, $\delta_{\mathbf{r}_j}$ is the Dirac delta distribution [29], and q_i is the charge of ionic species i.

To estimate the electrostatic potential Φ induced by ρ, we define a nonlocal dielectric model for protein in ionic solvent as follows:

$$\begin{cases} -\nabla\cdot(\epsilon(\mathbf{r})\nabla\Phi(\mathbf{r}) + \kappa(\mathbf{r}) \int_{\mathbb{R}^3} Q_\lambda(\mathbf{r}-\mathbf{r}')\nabla\Phi(\mathbf{r}')\mathrm{d}\mathbf{r}') = \frac{1}{\epsilon_0}\rho(\mathbf{r}), \quad \mathbf{r} \in \mathbb{R}^3, \\ \Phi(\mathbf{r}) \to 0 \quad \text{as} \quad |\mathbf{r}| \to \infty, \end{cases} \tag{2.2}$$

subject to the interface conditions $\Phi(\mathbf{s}^+) = \Phi(\mathbf{s}^-)$ and

$$\epsilon_\infty \frac{\partial\Phi(\mathbf{s}^+)}{\partial\mathbf{n}(\mathbf{s})} + (\epsilon_s - \epsilon_\infty) \int_{\mathbb{R}^3} Q_\lambda(\mathbf{s}-\mathbf{r}')\nabla\Phi(\mathbf{r}')\mathrm{d}\mathbf{r}' \cdot \mathbf{n}(\mathbf{s}) = \epsilon_p \frac{\partial\Phi(\mathbf{s}^-)}{\partial\mathbf{n}(\mathbf{s})}, \quad \mathbf{s} \in \Gamma, \tag{2.3}$$

where $\varepsilon(\mathbf{r})$ and $\kappa(\mathbf{r})$ are defined by

$$\epsilon(\mathbf{r}) = \begin{cases} \epsilon_p, & \mathbf{r} \in D_p, \\ \epsilon_\infty, & \mathbf{r} \in D_s, \end{cases} \quad \kappa(\mathbf{r}) = \begin{cases} 0, & \mathbf{r} \in D_p, \\ \epsilon_s - \epsilon_\infty, & \mathbf{r} \in D_s, \end{cases} \tag{2.4}$$

$\mathbf{n}(\mathbf{s})$ denotes the unit outward normal vector of D_p, $Q_\lambda(\mathbf{r})$ is defined by

$$Q_\lambda(\mathbf{r}) = \frac{1}{4\pi\lambda^2|\mathbf{r}|} e^{-|\mathbf{r}|/\lambda}, \quad \mathbf{r} \neq 0, \tag{2.5}$$

$\frac{\partial\Phi(\mathbf{s})}{\partial\mathbf{n}(\mathbf{s})} = \nabla\Phi(\mathbf{s})\cdot\mathbf{n}(\mathbf{s})$, $\frac{\partial\Phi(\mathbf{s}^\pm)}{\partial\mathbf{n}(\mathbf{s})} = \lim_{t\to 0+} \frac{\partial\Phi(\mathbf{s}\pm t\mathbf{n}(\mathbf{s}))}{\partial\mathbf{n}(\mathbf{s})}$, and $\Phi(\mathbf{s}^\pm) = \lim_{t\to 0^+} \Phi(\mathbf{s}\pm t\mathbf{n})$.

Clearly, the nonlocal model (2.2) includes the classic local Poisson model and the nonlocal model for protein in pure water studied in [36] as two special cases. In fact, setting all $c_i = 0$ reduces (2.2) to the case of protein in pure water while setting $\epsilon_\infty = \epsilon_s$ reduces (2.2) to the classic local Poisson model

$$\begin{cases} -\nabla\cdot(\epsilon(\mathbf{r})\nabla\Phi(\mathbf{r})) = \frac{1}{\epsilon_0}\rho(\mathbf{r}), \quad \mathbf{r} \in \mathbb{R}^3, \\ \Phi(\mathbf{r}) \to 0 \quad \text{as } |\mathbf{r}| \to \infty, \end{cases} \tag{2.6}$$

subject to the interface conditions

$$\Phi(\mathbf{s}^-) = \Phi(\mathbf{s}^+), \quad \epsilon_p \frac{\partial\Phi(\mathbf{s}^-)}{\partial\mathbf{n}(\mathbf{s})} = \epsilon_s \frac{\partial\Phi(\mathbf{s}^+)}{\partial\mathbf{n}(\mathbf{s})}, \quad \mathbf{s} \in \Gamma. \tag{2.7}$$

Due to this reason, we often refer to the nonlocal model (2.2) as a nonlocal Poisson dielectric model.

One major difficulty in solving and analyzing the nonlocal model (2.2) comes from the integral term which mixes with derivatives. As in [35, 36], we can detach the derivatives from the integral term to simplify (2.2) as follows:

$$\begin{cases} -\epsilon_p \Delta\Phi(\mathbf{r}) = \frac{1}{\epsilon_0}\sum_{j=1}^{n_p} Q_j \delta_{\mathbf{r}_j}, & \mathbf{r} \in D_p, \\ -\epsilon_\infty \Delta\Phi(\mathbf{r}) + \frac{\epsilon_s-\epsilon_\infty}{\lambda^2}[\Phi(\mathbf{r}) - (\Phi * Q_\lambda)(\mathbf{r})] = \frac{1}{\epsilon_0}\sum_{i=1}^{n} q_i c_i(\mathbf{r}), & \mathbf{r} \in D_s, \\ \Phi(\mathbf{s}^-) = \Phi(\mathbf{s}^+), & \mathbf{s} \in \Gamma \\ \epsilon_p \frac{\partial\Phi(\mathbf{s}^-)}{\partial \mathbf{n}(\mathbf{s})} = \epsilon_\infty \frac{\partial\Phi(\mathbf{s}^+)}{\partial \mathbf{n}(\mathbf{s})} + (\epsilon_s - \epsilon_\infty)\frac{\partial(\Phi * Q_\lambda)(\mathbf{s})}{\partial \mathbf{n}(\mathbf{s})}, & \mathbf{s} \in \Gamma, \\ \Phi(\mathbf{r}) \to 0 \quad \text{as } |\mathbf{r}| \to \infty, \end{cases} \tag{2.8}$$

where $\Phi * Q_\lambda$ denotes the convolution of Q_λ with Φ, which is defined by

$$(\Phi * Q_\lambda)(\mathbf{r}) = \int_{\mathbb{R}^3} Q_\lambda(\mathbf{r}-\mathbf{r}')\Phi(\mathbf{r}')\mathrm{d}\mathbf{r}'.$$

This reformulation reduces the complexity of solving the nonlocal model (2.2) remarkably.

3. Solution splitting formulation. However, because of the Dirac delta distributions $\delta_{\mathbf{r}_j}$, the solution Φ of the nonlocal Poisson model (2.2) has singular points at the point charge positions $\mathbf{r}_j$ for $j = 1, 2, \ldots, n_p$, causing difficulties in the analysis and numerical solution of (2.2). To overcome such a difficulty, in this section, we introduce a solution splitting formula in the following theorem.

THEOREM 3.1. *If each c_i is a given function independent of Φ for $i = 1, 2, \ldots, n$, then the solution Φ of the nonlocal Poisson dielectric model* (2.8) *can be split as*

$$\Phi(\mathbf{r}) = \Psi(\mathbf{r}) + \tilde{\Phi}(\mathbf{r}) + G(\mathbf{r}), \qquad \mathbf{r} \in \mathbb{R}^3, \tag{3.1}$$

where G is defined by

$$G(\mathbf{r}) = \frac{1}{4\pi\epsilon_0\epsilon_p}\sum_{j=1}^{n_p}\frac{Q_j}{|\mathbf{r}-\mathbf{r}_j|} \quad \textit{with all } \mathbf{r}_j \in D_p, \tag{3.2}$$

Ψ satisfies the nonlocal interface problem

$$\begin{cases} \Delta\Psi(\mathbf{r}) = 0, & \mathbf{r} \in D_p, \\ -\epsilon_\infty \Delta\Psi(\mathbf{r}) + \alpha[\Psi(\mathbf{r}) - (\Psi * Q_\lambda)(\mathbf{r})] = -\alpha[G(\mathbf{r}) - (G * Q_\lambda)(\mathbf{r})], & \mathbf{r} \in D_s, \\ \Psi(\mathbf{s}^-) = \Psi(\mathbf{s}^+), & \mathbf{s} \in \Gamma, \\ \epsilon_p \frac{\partial\Psi(\mathbf{s}^-)}{\partial \mathbf{n}(\mathbf{s})} - \epsilon_\infty \frac{\partial\Psi(\mathbf{s}^+)}{\partial \mathbf{n}(\mathbf{s})} = (\epsilon_s - \epsilon_\infty)\frac{\partial(\Psi * Q_\lambda)(\mathbf{s})}{\partial \mathbf{n}(\mathbf{s})} + g(\mathbf{s}), & \mathbf{s} \in \Gamma, \\ \Psi(\mathbf{r}) \to 0 \quad \textit{as } |\mathbf{r}| \to \infty, \end{cases} \tag{3.3}$$

and $\tilde{\Phi}$ *satisfies the nonlocal interface problem*

$$\begin{cases} \Delta\tilde{\Phi}(\mathbf{r}) = 0, & \mathbf{r} \in D_p, \\ -\epsilon_\infty \Delta\tilde{\Phi}(\mathbf{r}) + \alpha[\tilde{\Phi}(\mathbf{r}) - (\tilde{\Phi} * Q_\lambda)(\mathbf{r})] = \frac{1}{\epsilon_0}\sum_{i=1}^{n} q_i c_i(\mathbf{r}), & \mathbf{r} \in D_s, \\ \tilde{\Phi}(\mathbf{s}^-) = \tilde{\Phi}(\mathbf{s}^+), & \mathbf{s} \in \Gamma, \\ \epsilon_p \frac{\partial\tilde{\Phi}(\mathbf{s}^-)}{\partial\mathbf{n}(\mathbf{s})} - \epsilon_\infty \frac{\partial\tilde{\Phi}(\mathbf{s}^+)}{\partial\mathbf{n}(\mathbf{s})} = (\epsilon_s - \epsilon_\infty)\frac{\partial(\tilde{\Phi} * Q_\lambda)(\mathbf{s})}{\partial\mathbf{n}(\mathbf{s})}, & \mathbf{s} \in \Gamma, \\ \tilde{\Phi}(\mathbf{r}) \to 0 \quad as \ \ |\mathbf{r}| \to \infty. \end{cases} \tag{3.4}$$

Here α *and* $g(\mathbf{s})$ *are given by*

$$\alpha = \frac{\epsilon_s - \epsilon_\infty}{\lambda^2}, \qquad g(\mathbf{s}) = (\epsilon_s - \epsilon_\infty)\frac{\partial(G * Q_\lambda)(\mathbf{s})}{\partial\mathbf{n}(\mathbf{s})} + (\epsilon_\infty - \epsilon_p)\frac{\partial G(\mathbf{s})}{\partial\mathbf{n}(\mathbf{s})}.$$

Proof. With the functions G, Ψ, and $\tilde{\Phi}$, given in (3.2), (3.3), and (3.4), we can directly verify that the function Φ defined by (3.1) satisfies (2.8).

Since it is well known that $-\Delta\frac{1}{4\pi|\mathbf{r}-\mathbf{r}_j|} = \delta_{\mathbf{r}_j}$ (see [24, page 111], for example), the definition of G in (3.2) gives

$$-\epsilon_p\Delta G = \frac{1}{\epsilon_0}\sum_{j=1}^{n_p} Q_j \Delta\left(-\frac{1}{4\pi|\mathbf{r}-\mathbf{r}_j|}\right) = \frac{1}{\epsilon_0}\sum_{j=1}^{n_p} Q_j\delta_{\mathbf{r}_j}.$$

Thus, together with (3.1) and the first equations of (3.3) and (3.4), we get the first equation of (2.8) as shown below:

$$-\epsilon_p\Delta\Phi(\mathbf{r}) = -\epsilon_p\Delta(\tilde{\Phi}(\mathbf{r}) + \Psi(\mathbf{r}) + G(\mathbf{r})) = -\epsilon_p\Delta G(\mathbf{r}) = \frac{1}{\epsilon_0}\sum_{j=1}^{n_p} Q_j\delta_{\mathbf{r}_j}, \quad \mathbf{r} \in D_p.$$

Obviously, $\Delta G(\mathbf{r}) = 0$ for $\mathbf{r} \in D_s$. With (3.1) and the second equations of (3.3) and (3.4), we can obtain the second equation of (2.8) as shown below:

$$\begin{aligned} &-\epsilon_\infty\Delta\Phi(\mathbf{r}) + \alpha[\Phi(\mathbf{r}) - (\Phi * Q_\lambda)(\mathbf{r})] \\ =& -\epsilon_\infty\Delta\Big[\tilde{\Phi}(\mathbf{r}) + \Psi(\mathbf{r}) + G(\mathbf{r})\Big] + \alpha\Big[\tilde{\Phi}(\mathbf{r}) + \Psi(\mathbf{r}) + G(\mathbf{r}) \\ &-(\tilde{\Phi} * Q_\lambda)(\mathbf{r}) - (\Psi * Q_\lambda)(\mathbf{r}) - (G * Q_\lambda)(\mathbf{r})\Big] \\ =& -\epsilon_\infty\Delta\tilde{\Phi}(\mathbf{r}) + \alpha[\tilde{\Phi}(\mathbf{r}) - (\tilde{\Phi} * Q_\lambda)(\mathbf{r})] - \epsilon_\infty\Delta\Psi(\mathbf{r}) \\ &+\alpha[\Psi(\mathbf{r}) - (\Psi * Q_\lambda)(\mathbf{r})] + \alpha[G(\mathbf{r}) - (G * Q_\lambda)(\mathbf{r})] \\ =& -\epsilon_\infty\Delta\tilde{\Phi}(\mathbf{r}) + \alpha[\tilde{\Phi}(\mathbf{r}) - (\tilde{\Phi} * Q_\lambda)(\mathbf{r})] = \frac{1}{\epsilon_0}\sum_{i=1}^{n} q_i c_i(\mathbf{r}), \quad \mathbf{r} \in D_s. \end{aligned}$$

Clearly, $G(\mathbf{r}) \to 0$ as $|\mathbf{r}| \to \infty$. Combining this fact with the boundary conditions of (3.3) and (3.4), we obtain the boundary condition of (2.8):

$$\Phi(\mathbf{r}) = \tilde{\Phi}(\mathbf{r}) + \Psi(\mathbf{r}) + G(\mathbf{r}) \to 0 \qquad \text{as } |\mathbf{r}| \to \infty.$$

Since G is smooth on the interface Γ, the first interface condition of (2.8) follows from the first interface conditions of (3.3) and (3.4). Finally, by the second interface conditions of (3.3) and (3.4), the second interface condition of (2.8) is verified as follows:

$$\begin{aligned}
&\epsilon_p \frac{\partial \Phi(\mathbf{s}^-)}{\partial \mathbf{n}(\mathbf{s})} = \epsilon_p \frac{\partial \tilde{\Phi}(\mathbf{s}^-)}{\partial \mathbf{n}(\mathbf{s})} + \epsilon_p \frac{\partial \Psi(\mathbf{s}^-)}{\partial \mathbf{n}(\mathbf{s})} + \epsilon_p \frac{\partial G(\mathbf{s})}{\partial \mathbf{n}(\mathbf{s})} \\
=&\epsilon_\infty \frac{\partial \tilde{\Phi}(\mathbf{s}^+)}{\partial \mathbf{n}(\mathbf{s})} + (\epsilon_s - \epsilon_\infty) \frac{\partial (\tilde{\Phi} * Q_\lambda)(\mathbf{s})}{\partial \mathbf{n}(\mathbf{s})} + \epsilon_\infty \frac{\partial \Psi(\mathbf{s}^+)}{\partial \mathbf{n}(\mathbf{s})} + (\epsilon_s - \epsilon_\infty) \frac{\partial (\Psi * Q_\lambda)(\mathbf{s})}{\partial \mathbf{n}(\mathbf{s})} \\
&+ (\epsilon_s - \epsilon_\infty) \frac{\partial (G * Q_\lambda)(\mathbf{s})}{\partial \mathbf{n}(\mathbf{s})} + (\epsilon_\infty - \epsilon_p) \frac{\partial G(\mathbf{s})}{\partial \mathbf{n}(\mathbf{s})} + \epsilon_p \frac{\partial G(\mathbf{s})}{\partial \mathbf{n}(\mathbf{s})}, \\
=&\epsilon_\infty \frac{\partial}{\partial \mathbf{n}(\mathbf{s})} \big[\Psi(\mathbf{s}^+) + \tilde{\Phi}(\mathbf{s}^+) + G(\mathbf{s}) \big] \\
&+ (\epsilon_s - \epsilon_\infty) \frac{\partial}{\partial \mathbf{n}(\mathbf{s})} \big[(\Psi * Q_\lambda)(\mathbf{s}) + (\tilde{\Phi} * Q_\lambda)(\mathbf{s}) + (G * Q_\lambda)(\mathbf{s}) \big] \\
=&\epsilon_\infty \frac{\partial \Phi(\mathbf{s}^+)}{\partial \mathbf{n}(\mathbf{s})} + (\epsilon_s - \epsilon_\infty) \frac{\partial (\Phi * Q_\lambda)(\mathbf{s})}{\partial \mathbf{n}(\mathbf{s})}.
\end{aligned}$$

This completes the proof. □

Because of Theorem 3.1, the solution Φ of (2.8) can now be found through solving the nonlocal interface problems (3.3) and (3.4), which do not involve any singular Dirac delta distributions. In this way, the singular difficulties caused by Dirac delta distributions have been avoided completely in the solution of the nonlocal model (2.8).

4. Reformulation of PDE systems. To further reduce the complexity of a nonlocal model, in this section, we reformulate the nonlocal interface problems (2.8), (3.3), and (3.4) as three PDE systems without any convolution calculation. x Let u, u_1, and u_2 denote the convolutions $\Phi * Q_\lambda$, $\Psi * Q_\lambda$, and $\tilde{\Phi} * Q_\lambda$, respectively. As in [36], we can find that they satisfy the following three equations:

$$-\lambda^2 \Delta u(\mathbf{r}) + u(\mathbf{r}) - \Phi(\mathbf{r}) = 0, \quad \mathbf{r} \in R^3, \tag{4.1}$$

$$-\lambda^2 \Delta u_1(\mathbf{r}) + u_1(\mathbf{r}) - \Psi(\mathbf{r}) = 0, \quad \mathbf{r} \in R^3, \tag{4.2}$$

$$-\lambda^2 \Delta u_2(\mathbf{r}) + u_2(\mathbf{r}) - \tilde{\Phi}(\mathbf{r}) = 0, \quad \mathbf{r} \in R^3. \tag{4.3}$$

We then combine them with the nonlocal problems (2.8), (3.3), and (3.4) to yield three systems of PDEs as listed below.

The first system for solving (Φ, u) is given by

$$\begin{cases}
-\epsilon_p \Delta \Phi(\mathbf{r}) = \frac{1}{\epsilon_0} \sum_{j=1}^{n_p} Q_j \delta_{\mathbf{r}_j}, & \mathbf{r} \in D_p, \\
-\epsilon_\infty \Delta \Phi(\mathbf{r}) + \alpha[\Phi(\mathbf{r}) - u(\mathbf{r})] = \frac{1}{\epsilon_0} \sum_{i=1}^{n} q_i c_i(\mathbf{r}), & \mathbf{r} \in D_s, \\
-\lambda^2 \Delta u(\mathbf{r}) + u(\mathbf{r}) - \Phi(\mathbf{r}) = 0, & \mathbf{r} \in \mathbb{R}^3, \\
\Phi(\mathbf{r}) \to 0, \quad u(\mathbf{r}) \to 0 & \text{as } |\mathbf{r}| \to \infty,
\end{cases} \tag{4.4}$$

subject to the interface conditions

$$\Phi(\mathbf{s}^-) = \Phi(\mathbf{s}^+), \quad \epsilon_p \frac{\partial \Phi(\mathbf{s}^-)}{\partial \mathbf{n}(\mathbf{s})} = \epsilon_\infty \frac{\partial \Phi(\mathbf{s}^+)}{\partial \mathbf{n}(\mathbf{s})} + (\epsilon_s - \epsilon_\infty) \frac{\partial u(\mathbf{s})}{\partial \mathbf{n}(\mathbf{s})}, \quad \mathbf{s} \in \Gamma. \tag{4.5}$$

The second system for solving (Ψ, u_1) is given by

$$\begin{cases} \Delta \Psi(\mathbf{r}) = 0, & \mathbf{r} \in D_p, \\ -\epsilon_\infty \Delta \Psi(\mathbf{r}) + \alpha[\Psi(\mathbf{r}) - u_1(\mathbf{r})] = -\alpha[G(\mathbf{r}) - u_0(\mathbf{r})], & \mathbf{r} \in D_s, \\ -\lambda^2 \Delta u_1(\mathbf{r}) + u_1(\mathbf{r}) - \Psi(\mathbf{r}) = 0, & \mathbf{r} \in \mathbb{R}^3, \\ \Psi(\mathbf{r}) \to 0, \quad u_1(\mathbf{r}) \to 0 & \text{as } |\mathbf{r}| \to \infty, \end{cases} \tag{4.6}$$

subject to the interface conditions

$$\begin{aligned} \Psi(\mathbf{s}^-) = \Psi(\mathbf{s}^+), \quad \epsilon_p \frac{\partial \Psi(\mathbf{s}^-)}{\partial \mathbf{n}(\mathbf{s})} &= \epsilon_\infty \frac{\partial \Psi(\mathbf{s}^+)}{\partial \mathbf{n}(\mathbf{s})} + (\epsilon_s - \epsilon_\infty) \frac{\partial u_1(\mathbf{s})}{\partial \mathbf{n}(\mathbf{s})} \\ &+ (\epsilon_s - \epsilon_\infty) \frac{\partial u_0(\mathbf{s})}{\partial \mathbf{n}(\mathbf{s})} + (\epsilon_\infty - \epsilon_p) \frac{\partial G(\mathbf{s})}{\partial \mathbf{n}(\mathbf{s})} \qquad \forall\, \mathbf{s} \in \Gamma. \end{aligned} \tag{4.7}$$

The third system for solving $(\tilde{\Phi}, u_2)$ is given by

$$\begin{cases} \Delta \tilde{\Phi}(\mathbf{r}) = 0, & \mathbf{r} \in D_p, \\ -\epsilon_\infty \Delta \tilde{\Phi}(\mathbf{r}) + \alpha[\tilde{\Phi}(\mathbf{r}) - u_2(\mathbf{r})] = \frac{1}{\epsilon_0} \sum\limits_{i=1}^{n} q_i c_i(\mathbf{r}), & \mathbf{r} \in D_s, \\ -\lambda^2 \Delta u_2(\mathbf{r}) + u_2(\mathbf{r}) - \tilde{\Phi}(\mathbf{r}) = 0, & \mathbf{r} \in \mathbb{R}^3, \\ \tilde{\Phi}(\mathbf{r}) \to 0, \quad u_2(\mathbf{r}) \to 0 & \text{as } |\mathbf{r}| \to \infty, \end{cases} \tag{4.8}$$

subject to the interface conditions

$$\tilde{\Phi}(\mathbf{s}^-) = \tilde{\Phi}(\mathbf{s}^+), \quad \epsilon_p \frac{\partial \tilde{\Phi}(\mathbf{s}^-)}{\partial \mathbf{n}(\mathbf{s})} = \epsilon_\infty \frac{\partial \tilde{\Phi}(\mathbf{s}^+)}{\partial \mathbf{n}(\mathbf{s})} + (\epsilon_s - \epsilon_\infty) \frac{\partial u_2(\mathbf{s})}{\partial \mathbf{n}(\mathbf{s})} \quad \forall\, \mathbf{s} \in \Gamma. \tag{4.9}$$

Here $\alpha = (\epsilon_s - \epsilon_\infty)/\lambda^2$, $u_0 = G * Q_\lambda$, and $\frac{\partial G(\mathbf{s})}{\partial \mathbf{n}(\mathbf{s})}$ can be found as

$$\frac{\partial G(\mathbf{s})}{\partial \mathbf{n}(\mathbf{s})} = -\frac{1}{4\pi\epsilon_0\epsilon_p} \sum_{j=1}^{n_p} Q_j \frac{(\mathbf{s} - \mathbf{r}_j) \cdot \mathbf{n}}{|\mathbf{s} - \mathbf{r}_j|^3}.$$

Furthermore, we find that u_0 can be calculated as a solution of the following boundary value problem

$$\begin{cases} -\lambda^2 \Delta u_0(\mathbf{r}) + u_0(\mathbf{r}) = G(\mathbf{r}) & \text{in } \mathbb{R}^3, \\ u_0(\mathbf{r}) \to 0 & \text{as } |\mathbf{r}| \to \infty. \end{cases} \tag{4.10}$$

5. Solution Existence and Uniqueness. In this section, we consider the solution existence and uniqueness for (4.4), (4.6), (4.8), and (4.10). For simplicity, we assume them to be approximated as boundary value problems with the homogeneous Dirichlet boundary conditions

$$\Phi(\mathbf{s}) = 0, \quad \tilde{\Phi}(\mathbf{s}) = 0, \quad \Psi(\mathbf{s}) = 0, \quad u_i(\mathbf{s}) = 0, \quad \mathbf{s} \in \partial\Omega, \tag{5.1}$$

where $i = 0, 1, 2$, $\partial\Omega$ denotes the boundary of a sufficiently large spherical domain, Ω, satisfying $D_p \subset \Omega$ and $D_s = \Omega \setminus (D_p \cup \Gamma)$. The case of inhomogeneous boundary value problems can be constructed by several numerical techniques (see [16, 23] for examples) and then treated by standard variation techniques (see [9, pages 137 - 138], for example). Hence, considering a homogeneous Dirichlet boundary condition here does not lose any generality.

We start with the derivation of the variational formulation of (4.4) over Ω. We define the function spaces $\mathcal{V}_p$ as

$$\mathcal{V}_p = W_p^1(\Omega) \times W_p^1(\Omega),$$

where $W_p^1(\Omega)$ denotes the usual Sobolev function space [9]. Here we will take p and p' to be dual indices, such that $(1/p) + (1/p') = 1$. We choose $1 < p < 3/2$ in order to ensure that the solution $(\Phi, u) \in \mathcal{V}_p$. This also ensures that $3 < p' < \infty$ so that test functions in $\mathcal{V}_{p'}$ are continuous. (Note that working with Sobolev spaces with $p \neq 2$ is technical, so we actually do most of our work with the conventional case $p = 2$ based on our solution splitting approach.) We write $W_{p,0}^1(\Omega)$ for the functions in $W_p^1(\Omega)$ that vanish on the boundary of Ω. For any test function vector $\underset{\sim}{v} = (v_1, v_2) \in \mathcal{V}_{p'}$, we multiply the first and second equations of (4.4) with v_1, integrate them, and then add them together to get

$$\begin{aligned} &-\epsilon_p \int_{D_p} \Delta\Phi(\mathbf{r}) v_1(\mathbf{r}) \mathrm{d}\mathbf{r} - \epsilon_\infty \int_{D_s} \Delta\Phi(\mathbf{r}) v_1(\mathbf{r}) \mathrm{d}\mathbf{r} + \alpha \int_{D_s} (\Phi(\mathbf{r}) - u(\mathbf{r})) v_1(\mathbf{r}) \mathrm{d}\mathbf{r} \\ =&\frac{1}{\epsilon_0} \int_\Omega \rho(\mathbf{r}) v_1(\mathbf{r}) \mathrm{d}\mathbf{r}. \end{aligned}$$

Applying the Green's first identity to the first two integrals, we can simplify the above expression as

$$\begin{aligned} &\int_\Omega \epsilon(\mathbf{r}) \nabla\Phi(\mathbf{r}) \cdot \nabla v_1(\mathbf{r}) \mathrm{d}\mathbf{r} - \int_\Gamma \left[\epsilon_p \frac{\partial \Phi(\mathbf{s}^-)}{\partial \mathbf{n}(\mathbf{s})} - \epsilon_\infty \frac{\partial \Phi(\mathbf{s}^+)}{\partial \mathbf{n}(\mathbf{s})} \right] v_1(\mathbf{s}) \mathrm{d}\mathbf{s} \\ &+\alpha \int_{D_s} (\Phi(\mathbf{r}) - u(\mathbf{r})) v_1(\mathbf{r}) \mathrm{d}\mathbf{r} = \frac{1}{\epsilon_0} \int_\Omega \rho v_1 \mathrm{d}\mathbf{r}. \end{aligned}$$

Substituting the interface conditions of (4.5) to the above expression gives

$$\begin{aligned} &\int_\Omega \epsilon(\mathbf{r}) \nabla\Phi(\mathbf{r}) \cdot \nabla v_1(\mathbf{r}) \mathrm{d}\mathbf{r} + \alpha \int_{D_s} (\Phi(\mathbf{r}) - u(\mathbf{r})) v_1(\mathbf{r}) \mathrm{d}\mathbf{r} \\ &-(\epsilon_s - \epsilon_\infty) \int_\Gamma \frac{\partial u(\mathbf{s})}{\partial \mathbf{n}(\mathbf{s})} v_1(\mathbf{s}) \mathrm{d}\mathbf{s} = \frac{1}{\epsilon_0} \int_\Omega \rho(\mathbf{r}) v_1(\mathbf{r}) \mathrm{d}\mathbf{r} \quad \forall v_1 \in W_{p',0}^1(\Omega). \end{aligned} \tag{5.2}$$

Note that the interface Γ is a complex molecular surface of a protein in our application, which may cause difficulties in a calculation of the surface integral $\int_\Gamma \frac{\partial u(\mathbf{s})}{\partial \mathbf{n}(\mathbf{s})} v_1(\mathbf{s}) \mathrm{d}\mathbf{s}$. To avoid such difficulties, we use the Green's first identity and the third equation of

(4.4) to reformulate the surface integral as

$$\begin{aligned}\int_\Gamma \frac{\partial u(\mathbf{s})}{\partial \mathbf{n}(\mathbf{s})} v_1(\mathbf{s})\mathrm{d}\mathbf{s} &= -\int_{D_s} \Delta u(\mathbf{r}) v_1(\mathbf{r})\mathrm{d}\mathbf{r} - \int_{D_s} \nabla u(\mathbf{r})\cdot\nabla v_1(\mathbf{r})\mathrm{d}\mathbf{r} \\ =&\frac{1}{\lambda^2}\int_{D_s}(\Phi(\mathbf{r})-u(\mathbf{r}))v_1(\mathbf{r})\mathrm{d}\mathbf{r} - \int_{D_s}\nabla u(\mathbf{r})\cdot\nabla v_1(\mathbf{r})\mathrm{d}\mathbf{r}.\end{aligned} \tag{5.3}$$

By the above expression, (5.2) becomes the one without any surface integral term:

$$\int_\Omega \epsilon(\mathbf{r})\nabla\Phi(\mathbf{r})\cdot\nabla v_1(\mathbf{r})\mathrm{d}\mathbf{r} + (\epsilon_s-\epsilon_\infty)\int_{D_s}\nabla u(\mathbf{r})\cdot\nabla v_1(\mathbf{r})\mathrm{d}\mathbf{r} = \frac{1}{\epsilon_0}\int_\Omega \rho(\mathbf{r})v_1(\mathbf{r})\mathrm{d}\mathbf{r}. \tag{5.4}$$

The variational formulation of the third equation of (4.4) can be easily obtained in the form

$$\lambda^2\int_\Omega \nabla u(\mathbf{r})\cdot\nabla v_2(\mathbf{r})\mathrm{d}\mathbf{r} + \int_\Omega (u(\mathbf{r})-\Phi(\mathbf{r}))v_2(\mathbf{r})\mathrm{d}\mathbf{r} = 0 \quad \forall v_2 \in W^1_{p',0}(\Omega). \tag{5.5}$$

Adding (5.4) to (5.5) and setting $\underset{\sim}{\phi} = (\Phi, u)$, we obtain the variational formulation of (4.4) as follows:

$$a(\underset{\sim}{\phi}, \underset{\sim}{v}) = l(\underset{\sim}{v}) \qquad \forall \underset{\sim}{v} \in W^1_{p',0}(\Omega)\times W^1_{p',0}(\Omega), \tag{5.6}$$

where $a(\underset{\sim}{\phi}, \underset{\sim}{v})$ and $l(\underset{\sim}{v})$ are bilinear and linear forms as defined by

$$\begin{aligned}a(\underset{\sim}{\phi}, \underset{\sim}{v}) = &\int_\Omega \epsilon(\mathbf{r})\nabla\Phi(\mathbf{r})\cdot\nabla v_1(\mathbf{r})\mathrm{d}\mathbf{r} + (\epsilon_s-\epsilon_\infty)\int_{D_s}\nabla u(\mathbf{r})\cdot\nabla v_1(\mathbf{r})\mathrm{d}\mathbf{r} \\ &+\lambda^2\int_\Omega \nabla u(\mathbf{r})\cdot\nabla v_2(\mathbf{r})\mathrm{d}\mathbf{r} + \int_\Omega (u(\mathbf{r})-\Phi(\mathbf{r}))v_2(\mathbf{r})\mathrm{d}\mathbf{r}\end{aligned} \tag{5.7}$$

and

$$l(\underset{\sim}{v}) = \frac{1}{\epsilon_0}\left[\sum_{j=1}^{n_p} Q_j v_1(\mathbf{r}_j) + \sum_{i=1}^{n} q_i \int_{D_s} c_i(\mathbf{r})v_1(\mathbf{r})\mathrm{d}\mathbf{r}\right]. \tag{5.8}$$

Here the definition (2.1) of ρ has been applied to the derivation of $l(\underset{\sim}{v})$. The first summation term of (5.8) is clearly well defined since v_1 is continuous in D_p.

Similarly, we set that $\underset{\sim}{\phi}_1 = (\Psi, u_1)$, $\underset{\sim}{\phi}_2 = (\tilde{\Phi}, u_2)$, and $\mathcal{V} = H^1_0(\Omega)\times H^1_0(\Omega)$, and then obtain the variational formulations of (4.6) and (4.8) as follows:

$$\text{find } \underset{\sim}{\phi}_1 \in \mathcal{V} \text{ such that} \quad a(\underset{\sim}{\phi}_1, \underset{\sim}{v}) = \ell_1(\underset{\sim}{v}) \qquad \forall \underset{\sim}{v} \in \mathcal{V}, \tag{5.9}$$

and

$$\text{find } \underset{\sim}{\phi}_2 \in \mathcal{V} \text{ such that} \quad a(\underset{\sim}{\phi}_2, \underset{\sim}{v}) = \ell_2(\underset{\sim}{v}) \qquad \forall \underset{\sim}{v} \in \mathcal{V}, \tag{5.10}$$

where $\ell_1(\underset{\sim}{v})$ and $\ell_2(\underset{\sim}{v})$ are two linear forms as defined by

$$\ell_1(\underset{\sim}{v}) = (\epsilon_\infty-\epsilon_s)\int_{D_s}\nabla u_0(\mathbf{r})\cdot\nabla v_1(\mathbf{r})\mathrm{d}\mathbf{r} + (\epsilon_p-\epsilon_\infty)\int_{D_s}\nabla G(\mathbf{r})\cdot\nabla v_1(\mathbf{r})\mathrm{d}\mathbf{r} \tag{5.11}$$

and

$$\ell_2(\underset{\sim}{v}) = \frac{1}{\epsilon_0}\sum_{i=1}^{n} q_i \int_{D_s} c_i(\mathbf{r}) v_1(\mathbf{r}) \mathrm{d}\mathbf{r}. \tag{5.12}$$

Obviously, the variational problem of (4.10) with a homogenous Dirichlet boundary condition can be obtained in the form

$$\text{find } u_0 \in H_0^1(\Omega) \text{ such that} \quad a_0(u_0, w) = \ell_0(w) \qquad \forall w \in H_0^1(\Omega), \tag{5.13}$$

where ℓ_0 and a_0 are linear and bilinear forms as defined by

$$\ell_0(w) = \int_\Omega G(\mathbf{r}) w(\mathbf{r}) \mathrm{d}\mathbf{r}, \quad a_0(u_0, w) = \lambda^2 \int_\Omega \nabla u_0 \cdot \nabla w \mathrm{d}\mathbf{r} + \int_\Omega u_0 w \mathrm{d}\mathbf{r}. \tag{5.14}$$

We show the solution existence and uniqueness in the following theorem.

THEOREM 5.1. *Let the interface Γ be of class C^2, and $c_i \in L^2(\Omega)$ for $i = 1, 2, \cdots, n$. If λ is sufficiently large, then the weak forms* (5.6), (5.9), (5.10), *and* (5.13) *have unique solutions* (Φ, u), (Ψ, u_1), $(\tilde{\Phi}, u_2)$, *and* u_0. *Moreover, the solution* (Φ, u) *of* (5.6) *can be calculated by the solution splitting formulas*

$$\Phi(\mathbf{r}) = \Psi(\mathbf{r}) + \tilde{\Phi}(\mathbf{r}) + G(\mathbf{r}) \qquad \forall \mathbf{r} \in \Omega, \tag{5.15}$$

and

$$u(\mathbf{r}) = u_0(\mathbf{r}) + u_1(\mathbf{r}) + u_2(\mathbf{r}) \qquad \forall \mathbf{r} \in \Omega, \tag{5.16}$$

where G is given in (3.2).

Proof. It suffices to show that the form $a(\underset{\sim}{\phi}, \underset{\sim}{v})$ in (5.7) is coercive and that the forms ℓ_0, ℓ_1, and ℓ_2 are linear bounded functionals in $H_0^1(\Omega)$ (i.e., ℓ_i in the dual space $H_0^1(\Omega)'$). First we prove coercivity.

Let $\epsilon_{\min} = \min\{\epsilon_\infty, \epsilon_p\}$, $\underset{\sim}{v} = (v_1, v_2)$, and $|v|$ be the seminorm of v defined by

$$|v| = \left(\int_\Omega |\nabla v(\mathbf{r})|^2 \, d\mathbf{r} \right)^{1/2},$$

which is a norm on $H_0^1(\Omega)$ (see (5.18) below). Then

$$\begin{aligned} a(\underset{\sim}{v}, \underset{\sim}{v}) &\geq \epsilon_{\min} |v_1|^2 + \lambda^2 |v_2|^2 \\ &\quad + (\epsilon_s - \epsilon_\infty) \int_{D_s} \nabla v_1 \cdot \nabla v_2 \, d\mathbf{r} - \int_\Omega v_1 \, v_2 \, d\mathbf{r}. \end{aligned} \tag{5.17}$$

By the Cauchy-Schwarz inequality,

$$\left| \int_{D_s} \nabla v_1 \cdot \nabla v_2 \, d\mathbf{r} \right| \leq \frac{1}{2} \left(\delta |v_1|^2 + \frac{1}{\delta} |v_2|^2 \right)$$

for any $\delta > 0$. Picking $\delta = \epsilon_{\min}/(\epsilon_s - \epsilon_\infty)$, we find

$$(\epsilon_s - \epsilon_\infty) \left| \int_{D_s} \nabla v_1 \cdot \nabla v_2 \, d\mathbf{r} \right| \leq \frac{1}{2} \left(\epsilon_{\min} |v_1|^2 + \lambda^2 |v_2|^2 \right)$$

for any $\lambda \geq (\epsilon_s - \epsilon_\infty)/\sqrt{\epsilon_{\min}}$. From (5.17), we conclude that

$$a(\underset{\sim}{v},\ \underset{\sim}{v}) \geq \frac{1}{2}\left(\epsilon_{\min}|v_1|^2 + \lambda^2|v_2|^2\right) - \int_\Omega v_1\, v_2\, d\mathbf{r}.$$

By Poincaré's inequality,

$$\int_\Omega v^2\, d\mathbf{r} \leq C_\Omega\, |v|^2 \tag{5.18}$$

for all $v \in H_0^1(\Omega)$. Combining this with the Cauchy-Schwarz inequality,

$$\left|\int_\Omega v_1\, v_2\, d\mathbf{r}\right| \leq \frac{1}{2}\left(\delta C_\Omega|v_1|^2 + \frac{C_\Omega}{\delta}|v_2|^2\right)$$

for any $\delta > 0$. Choosing $\delta = \epsilon_{\min}/2C_\Omega$, we find

$$\left|\int_\Omega v_1\, v_2\, d\mathbf{r}\right| \leq \frac{1}{4}\left(\epsilon_{\min}|v_1|^2 + \frac{4\,C_\Omega^2}{\epsilon_{\min}}|v_2|^2\right).$$

Thus we conclude that

$$a(\underset{\sim}{v},\ \underset{\sim}{v}) \geq \frac{1}{4}\left(\min\{\epsilon_\infty, \epsilon_p\}|v_1|^2 + \lambda^2|v_2|^2\right) \quad \text{for} \quad \lambda \geq \frac{\max\{\epsilon_s - \epsilon_\infty, 2C_\Omega\}}{\sqrt{\min\{\epsilon_\infty, \epsilon_p\}}},$$

where $v_1, v_2 \in H_0^1(\Omega)$. This completes the proof of coercivity.

Now we discuss the boundedness of the forms ℓ_i, $i = 0, 1, 2$. The only term of concern occurs in the last expression in ℓ_1. But we are assuming that the points $\mathbf{r}_j$ are in the complement of the closure of the set D_s (that is, in the interior of the protein domain D_p), and so G is actually a smooth, bounded function on the closure of D_s. The terms involving G itself in ℓ_0 and the first integral in the definition of ℓ_1 may require some care in implementation. However, $G \in L^p(\Omega)$ for $p < 3$, and indeed $\nabla G \in L^p(\Omega)$ for $p < 3/2$. Thus all of the forms ℓ_i are well defined on $H_0^1(\Omega)$. This completes the proof of solution existence and uniqueness for (5.9), (5.10), and (5.13).

The solution splitting formulas (5.15) and (5.16) can follow from Theorem 3.1 and the definitions of u and u_i for $i = 0, 1, 2$.

Finally, the proof of existence of Φ in $W_p^1(\Omega)$ for $1 < p < 3/2$ follows from the fact that $G \in W_p^1(\Omega)$ and $H^1(\Omega) \subset W_p^1(\Omega)$, together with the corresponding spaces satisfying boundary conditions. Uniqueness corresponds to the case $G = 0$ for which the arguments in $H^1(\Omega)$ suffice. ∎

6. A fast algorithm and numerical examples. According to Theorem 5.1, in this section, we construct a fast algorithm for solving the nonlocal Poisson dielectric model (5.6). The outline of our new algorithm is presented in Algorithm 1. Here all the variational problems (5.9), (5.10), and (5.13) are given in the standard form so that they can be easily programmed for their numerical solutions based on the finite element program library `DOLFIN` [1].

Algorithm 1. *Let c_i be given for $i = 1, 2, \ldots, n$. The solution Φ of the variational problem* (5.6) *(i.e., the weak solution of the nonlocal model* (2.2)*) is calculated in the following steps:*

Step 1. Solve the Poisson-like equation (5.13) *for* u_0.
Step 2. Solve the nonlocal Poisson-like interface problem (5.9) *for* Ψ.
Step 3. Solve the nonlocal Poisson-like interface problem (5.10) *for* $\tilde{\Phi}$.
Step 4. Find the weak solution Φ *by the solution splitting formula* (5.15).

To test our new algorithm for a protein in an ionic solvent, which contains only sodium (N_a^+) and chloride (Cl^-) ions (i.e., $n = 2$, $q_1 = e_c$, and $q_2 = -e_c$), we set the ionic concentration functions c_1 and c_2 by the following expressions:

$$c_i(\mathbf{r}) = \frac{Me^{-\beta q_i w(\mathbf{r})}}{1 + 2\tau M \cosh(\beta e_c w(\mathbf{r}))}, \qquad \mathbf{r} \in D_s, \quad i = 1, 2, \tag{6.1}$$

where τ denotes the volume occupied by each ion or each water molecule, M denotes an average concentration for both sodium and chloride ions, e_c denots the elementary charge, $\beta = 1/(k_B T)$ with k_B being the Boltzmann constant and T being the absolute temperature, and w is a given function satisfying a classical LPBE with uniform size effect [22], which is defined as follows:

$$\begin{cases} -\nabla \cdot (\epsilon_p \nabla w(\mathbf{r})) = \dfrac{1}{\epsilon_0} \displaystyle\sum_{i=1}^{n_p} Q_i \delta_{\mathbf{r}_i}, & \mathbf{r} \in D_p, \\ -\nabla \cdot (\epsilon_s \nabla w(\mathbf{r})) + \dfrac{2M\beta e_c^2}{\epsilon_0(1 + 2\tau M)} w(\mathbf{r}) = 0, & \mathbf{r} \in D_s, \\ w(\mathbf{s}) = 0, & \mathbf{s} \in \partial\Omega. \end{cases} \tag{6.2}$$

In (6.1), c_i satisfies the volume constraint condition $c_0 + c_1 + c_2 = 1/\tau$. Here c_0 denotes the concentration function of water molecules, from which it implies that each c_i is a bounded positive function. With such a selection of c_i, the nonlocal Poisson model (2.2) can be regarded as *a nonlocal LPBE model*.

We implemented Algorithm 1 for solving the nonlocal LPBE model as a Python program package based on the finite element library `DOLFIN` [1]. To generate a tetrahedral finite element mesh for a protein represented in a PQR file, we adapted and converted the molecular surface and volumetric mesh generation program package `GAMer` [38] to a Python module by using `SWIG` (*http://www.swig.org*). The mesh data can be generated from `GAMer` within our package and then used to produce a `DOLFIN` mesh object and a domain partition for D_p and D_s. It is also saved in the XML format – one mesh format recognized by `DOLFIN`. In our tests, each linear algebraic system arising from a finite element approximation to the weak form (5.9), (5.10), or (5.13) was solved numerically by using a linear iterative solver GMRES with an incomplete LU preconditioning from the `PETSc` library [3], which is a part of `DOLFIN`. Here the two iterative termination parameters, *relative_tolerance* and *absolute_tolerance*, were set as 10^{-10}.

Our package contains a special construction of finite element interpolation functions for G and ∇G. It is known that the interpolant G_h of G has the expression

$$G_h(\mathbf{r}) = \sum_{i=1}^{N} G(\mathbf{r}^i)\xi_i(\mathbf{r}) \quad \text{with} \quad G(\mathbf{r}^i) = \frac{1}{4\pi\epsilon_0\epsilon_p} \sum_{j=1}^{n_p} \frac{Q_j}{|\mathbf{r}^i - \mathbf{r}_j|} \quad \forall \mathbf{r} \in \Omega, \tag{6.3}$$

where ξ_i denotes a base function of a finite element function space V at the ith mesh node $\mathbf{r}^i$ of a mesh, and N is the total number of mesh nodes. To speed up the calculation of mesh values $\{G(\mathbf{r}^i)\}_{i=1}^N$, which requires Nn_p multiplications, we wrote a Fortran program for computing the mesh values of G and ∇G, and converted it as a Python function via `f2py` [26]. We then constructed G_h directly as follows:

1. Set $G_h \in V$ via `DOLFIN`'s statement: $G_h = \texttt{Function}(V)$.
2. Use Fortran program to compute $G(\mathbf{r}^i)$, and return these values as array G_A.
3. Update G_h via `DOLFIN`'s statement: $G_h.vector()[:] = G_A$.

The interpolant of ∇G can be constructed similarly.

All the numerical tests were made on one 2.3 GHz Intel Core i7 processor of a MacBook Pro with 8 GB 1600 MHz DDR3 memory.

Remark: *One commonly used way to construct G_h in `DOLFIN` is to use the statement*

$$G_h = interpolate(G, V),$$

where G can be an object of `DOLFIN`'s complex user-defined JIT-compiled `Expression` with a C++ program section for calculating the mesh values of G [1]. However, in our numerical tests, it was found to be much less efficient than our special construction.

6.1. One ionic Born ball in water solvent. To validate our program package and demonstrate the advantage of Algorithm 1 in improving the accuracy of numerical solutions, we made numerical experiments on a nonlocal point charge Born model with a known analytical solution [36, Page 181]. Here an ion with a point charge q in pure water is modeled as a sphere with radius $a < 1$, and the point charge is located at the center of the sphere. Thus, we have that $\rho = q\delta$, $D_p = \{\mathbf{r} \in \mathbb{R}^3 \mid |\mathbf{r}| < a\}$, and all $c_i = 0$, from which it implies that $(\tilde{\Phi}, u_2) = (0, 0)$. The analytical solution (Φ, u) of such a nonlocal Poisson dielectric model (4.4) is given in [36, Pages 182-183], from which we find the analytical expressions of u_0, Ψ, and u_1 as follows:

$$u_0(\mathbf{r}) = \frac{q}{4\pi\epsilon_0\epsilon_p|\mathbf{r}|}(1 - e^{-|\mathbf{r}|/\lambda}),$$

$$\Psi(\mathbf{r}) = \begin{cases} \frac{q}{4\pi a\epsilon_0\epsilon_p\epsilon_s}\left[\epsilon_p - \epsilon_s - (\epsilon_\infty - \epsilon_s)b_1\right], & |\mathbf{r}| < a, \\ \frac{q}{4\pi\epsilon_0\epsilon_p\epsilon_s|\mathbf{r}|}\left[\epsilon_p - \epsilon_s - (\epsilon_\infty - \epsilon_s)b_1 e^{\mu(a-|\mathbf{r}|)}\right], & |\mathbf{r}| > a, \end{cases} \tag{6.4}$$

and

$$u_1(\mathbf{r}) = \begin{cases} \frac{q}{4\pi\epsilon_0\epsilon_p}\left(\frac{b_2}{|\mathbf{r}|}\sinh\frac{|\mathbf{r}|}{\lambda} + \frac{\epsilon_p - \epsilon_s - (\epsilon_\infty - \epsilon_s)b_1}{a\epsilon_s}\right), & |\mathbf{r}| \le a, \\ \frac{q}{4\pi\epsilon_0\epsilon_p\epsilon_s|\mathbf{r}|}\left(\epsilon_p - \epsilon_s - \epsilon_\infty b_1 e^{\mu(a-|\mathbf{r}|)} + \epsilon_s e^{-\frac{|\mathbf{r}|}{\lambda}}\right), & |\mathbf{r}| > a, \end{cases} \tag{6.5}$$

where μ, b_1, and b_2 are three constants as given by

$$\mu = \frac{1}{\lambda}\sqrt{\frac{\epsilon_s}{\epsilon_\infty}}, \quad b_1 = \frac{[a\epsilon_s + \lambda(\epsilon_p - \epsilon_s)\sinh\frac{a}{\lambda}]}{[a\sqrt{\epsilon_\infty\epsilon_s} + \lambda(\epsilon_\infty - \epsilon_s)]\sinh\frac{a}{\lambda} + a\epsilon_s\cosh\frac{a}{\lambda}},$$

and

$$b_2 = \frac{[a\sqrt{\epsilon_\infty \epsilon_s} + \lambda(\epsilon_\infty - \epsilon_s) - a\epsilon_s]e^{-\frac{a}{\lambda}} + \lambda(\epsilon_s - \epsilon_p)}{[a\sqrt{\epsilon_\infty \epsilon_s} + \lambda(\epsilon_\infty - \epsilon_s)]\sinh\frac{a}{\lambda} + a\epsilon_s \cosh\frac{a}{\lambda}}.$$

Using the above analytical solutions, we set Ω to be the unit ball with the origin as the center, and constructed "accurate" inhomogenous Dirichlet boundary conditions on the spherical surface $\partial\Omega = \{\mathbf{s} \in \mathbb{R}^3 \mid |\mathbf{s}| = 1\}$.

In the numerical tests, we set $a = 0.1$, $q = 1$, $\epsilon_0 = 1$, $\epsilon_p = 2$, $\epsilon_s = 78.54$, $\epsilon_\infty = 1.8$, and $\lambda = 15$. A nonuniform tetrahedral mesh of the unit spheric domain Ω that contains domain D_p as a small ball with radius $a = 0.1$ was generated by using `GAMer` [38], which contains 2955 vertices and 17357 tetrahedra. The smallest and largest grid sizes are 0.015 and 0.585, respectively. Here the origin point was excluded as a vertex of the mesh to avoid the singular point of function $G(\mathbf{r}) = q/(4\pi\epsilon_0\epsilon_p|\mathbf{r}|)$.

As comparison, we also solved the variational problem (5.6) directly without using our solution splitting formula. Here the `DOLFIN`'s function `PointSource` was used to deal with the difficulty caused by the Dirac delta distribution. In this case, the jth component of the right-hand side vector of a finite element algebraic linear system is given in the form $\frac{q}{\epsilon_0}\phi_{j,h}(0,0,0)$, where $\phi_{j,h}$ denotes the j-th finite element basis function, implying that the right-hand side vector has at most four nonzero entries, since only one tetrahedra contains the origin point $(0,0,0)$. Clearly, an approximation error has been produced in this `PointSource` treatment since the distribution source term $\delta_{\mathbf{r}_j}(v)$ is approximated within a finite element space spanned linearly by the base functions $\phi_{j,h}$, lowering the accuracy of a finite element solution.

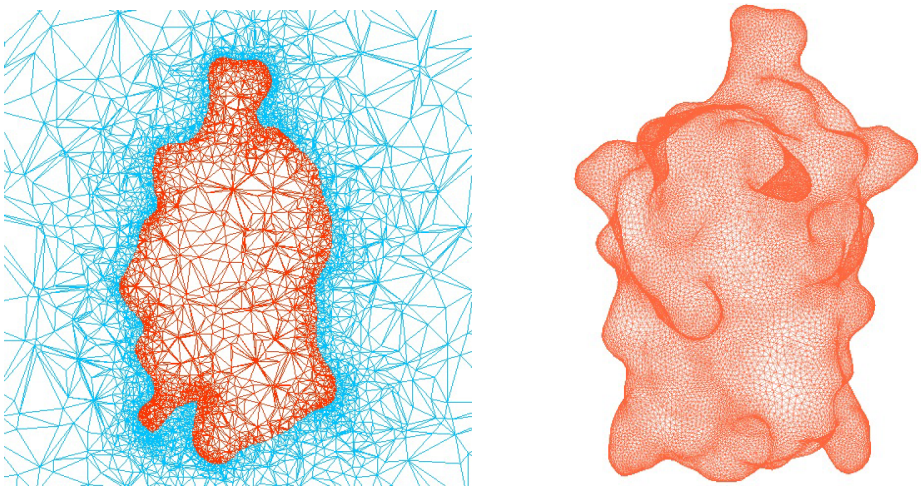
Table 6.1: Comparison of the accuracy of the finite element solution Φ_h calculated via Algorithm 1 with that calculated via `PointSource`. The numbers in parentheses are the number of mesh nodes in the kth order finite element method (FEM).

FEM Order k	Via Algorithm 1		Via `PointSource`	
	$\|\Phi - \Phi_h\|$	$\|\Phi - \Phi_h\|/\|\Phi\|$	$\|\Phi - \Phi_h\|$	$\|\Phi - \Phi_h\|/\|\Phi\|$
1 (2955)	5.849×10^{-4}	4.611×10^{-3}	1.106×10^{-2}	8.718×10^{-2}
2 (23522)	4.760×10^{-5}	3.752×10^{-4}	5.643×10^{-3}	4.448×10^{-2}
3 (79059)	8.250×10^{-6}	6.503×10^{-5}	4.204×10^{-3}	3.314×10^{-2}

The numerical results are reported in Table 6.1. Here the absolute error $\|\Phi - \Phi_h\|$ and relative error $\|\Phi - \Phi_h\|/\|\Phi\|$ are defined by

$$\|\Phi - \Phi_h\| = \sqrt{\int_\Omega |\Phi(\mathbf{r}) - \Phi_h(\mathbf{r})|^2 \mathrm{d}\mathbf{r}}, \quad \frac{\|\Phi - \Phi_h\|}{\|\Phi\|} = \sqrt{\frac{\int_\Omega |\Phi(\mathbf{r}) - \Phi_h(\mathbf{r})|^2 \mathrm{d}\mathbf{r}}{\int_\Omega |\Phi(\mathbf{r})|^2 \mathrm{d}\mathbf{r}}},$$

where Φ_h denotes the finite element solution. From Table 6.1 it can be seen that the finite element solution calculated via Algorithm 1 has a higher order of accuracy than the one calculated directly via `PointSource`. The numerical results not only validate



(a) A mesh cross section around BPTI (b) A molecular surface mesh of BPTI

Fig. 6.1: Two views of the tetrahedral finite element mesh with 69,982 vertices generated by `GAMer` [38] for protein BPTI represented in the PDB file 4PTI.

our program package but also confirm a well-known finite element theoretical result — the accuracy degree increases with the increment of the degree of the finite element approximation (see [9, Theorem 5.4.8, page 137] for example).

The reason why Algorithm 1 can lead to a higher solution accuracy can be explained as follows. In Algorithm 1, we only computed Ψ_h, and then set $\Phi_h = \Psi_h + G$. Since $\Phi = \Psi + G$, the finite element error $\Phi - \Phi_h$ is actually equal to $\Psi - \Psi_h$:

$$\Phi - \Phi_h = (\Psi + G) - (\Psi_h + G) = \Psi - \Psi_h.$$

Hence, the effect of the singular part G on the accuracy of numerical solution Φ_h has been completely eliminated within Algorithm 1.

6.2. Performance for protein in ionic solvent. To demonstrate the performance of our program package, we made numerical tests on three proteins represented in the PDB files 4PTI,1CID, and 2AQ5. The PDB files can be downloaded from the Protein Data Bank (PDB) (*http://www.rcsb.org/*). The protein represented in the PDB file 4PTI is also referred to as BPTI (Bovine Pancreatic Trypsin Inhibitor). Using the program package PDB2PQR (http://www.poissonboltzmann.org/pdb2pqr/) [11], we converted each PDB file into a PQR file, which contain atomic coordinates, charges, radii, and the hydrogen atoms missed in the PDB file. Each PQR file was then used as an input file for our Python program package. A finite element mesh was made by using `GAMer` for a spherical domain Ω containing a protein. The protein and mesh data are listed in Table 6.2. As a mesh demonstration, two views of the finite element mesh around protein BPTI are displayed in Figure 6.1.

Table 6.2: Protein and mesh data used in numerical tests.

Protein	#Atoms	Spherical domain radius	#vertices	#tetrahedra
4PTI	892	436Å	69,982	433,124
1CID	2783	711Å	116,591	721,815
2AQ5	6062	697Å	158,908	986,334

Table 6.3: Performance of our program package for solving the nonlocal LPBE model for three proteins represented in the PDB files 4PTI, 1CID, and 2AQ5.

	CPU time (seconds)			Percentage out of total time		
	4PTI	1CID	2AQ5	4PTI	1CID	2AQ5
Solve (5.13) for u_0	10.72	19.77	28.30	21.18%	16.96%	11.51%
Solve (5.9) for Ψ	9.59	17.61	27.56	18.96%	15.10%	11.22%
Solve (5.10) for $\tilde{\Phi}$	9.92	18.67	40.90	19.61%	16.01%	16.64%
Calculate c_1 & c_2	2.17	3.82	6.42	4.28%	3.28%	2.61%
Find G_h & $(\nabla G)_h$	1.40	7.02	20.58	2.76%	6.02%	8.37%
Generate mesh	16.60	49.45	121.62	32.81%	42.40%	49.48%
Total time (minutes)	0.84	1.94	4.10			

In numerical tests, we set $\tau = 150$ Å^3, $M = 0.1$ *Mol/L*, and $T = 298.15$ *K*. The other constants $\epsilon_p, \epsilon_s, \epsilon_\infty$, and λ were the same as for the Born ball case. The Dirichlet boundary conditions were set for $\tilde{\Phi}$ and Ψ as follows:

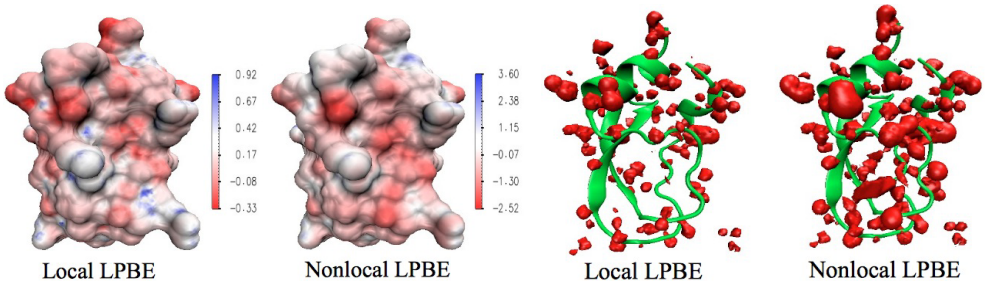
$$\tilde{\Phi}(\mathbf{s}) = 0, \quad \Psi(\mathbf{s}) = -G(\mathbf{s}) \quad \forall \mathbf{s} \in \partial\Omega.$$

Such settings guarantee $\Phi(\mathbf{s}) = 0$ on $\partial\Omega$. They are made because the values of G on $\partial\Omega$ are reduced much slowly than that of Φ as $|\mathbf{s}| \to \infty$. They are still too large to be ignored for the spherical domains Ω given in Table 6.2.

Figure 6.2 compares the solution Φ of the nonlocal LPBE model (i.e., (2.2) with c_i being given in (6.1)) with the solution $w(\mathbf{r})$ of the local LPBE model defined in (6.2) in two different views. Here both Φ and w were generated from our Python program for protein BPTI. From it we can see that the solution of the nonlocal LPBE model is significantly different from that of the local LPBE model.

Furthermore, Figure 6.3 display the finite element solutions of the nonlocal LPBE model for BPTI, which were generated from the three different meshes with 44572, 69982, and 106299 vertices, respectively. From Figure 6.3 it can be seen that the finite element solutions display similar patterns on the molecular surface, showing a good example of convergence of our new numerical scheme defined in Algorithm 1.

Table 6.3 reports the performance of our program package for proteins represented



(a) On the molecular surface of BPTI (b) Contour surfaces of solutions for BPTI

Fig. 6.2: A comparison of the solution Φ of the nonlocal LPBE model with the solution w of the local LPBE model (6.2) for protein BPTI represented in the PDB file 4PTI based on the mesh with 69,982 vertices. In Figure (b), the two contour surfaces (in red) are defined by equations $w(\mathbf{r}) = -1.5V$ and $\Phi(\mathbf{r}) = -1.5V$, respectively, and the backbone of BPTI is represented in green. The figures were made by using VMD [18].

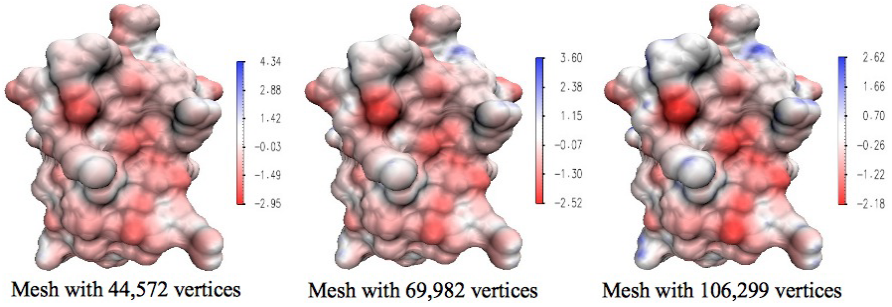


Fig. 6.3: Finite element solutions of the nonlocal LPBE models generated from three different tetrahedral meshes for protein BPTI.

in the PDB files 4PTI, 1CID, and 2AQ5. From it we can see that each related nonlocal problem, (5.13), (5.9), (5.10), or (6.2), can be efficiently solved by the iterative solver GMRES from the `PETSc` library [3]. For example, for a protein molecule (PDB ID 2AQ5) with 6062 atoms on a mesh with 158,908 vertices, a linear finite element solution of a system of two PDEs was found in about 41 seconds while the nonlocal LPBE model was solved in about 4 minutes. Furthermore, numerical tests show that our direct construction of interpolants G_h and $(\nabla G)_h$ is very efficient. For example, as shown in Table 6.3, constructing both G_h and $(\nabla G)_h$ took only 1.41 seconds in the case of 4PTI. These numerical results demonstrate a high efficiency of our program package in solving the nonlocal LPBE model.

The data in Tables 6.2 and 6.3 provide information regarding the scaling of our algorithms and implementation. In terms of total execution time, the scaling with respect to protein size (number of atoms) is actually sublinear. On the other hand,

the dependence of time on mesh size (either vertices or tetrahedra) is quadratic.

From Table 6.3 it can be seen that mesh generation is still the bottleneck in our application. We also noticed that the percentage of time for computing the interpolants G_h and $(\nabla G)_h$ might be further increased when the number n_p of atoms becomes huge. Hence, we may need a more efficient mesh generator for our Python package and study a variant of the fast multipole or tree code method [12, 14] to accelerate the calculation of G_h and $(\nabla G)_h$. Finally, since FEniCS executes in parallel and uses PETSc in part to achieve scalability, we plan to develop a version of our program package suitable for large scale parallel computation in the future.

Acknowledgements. The authors thank the anonymous referees' valuable comments.

REFERENCES

[1] G. N. Wells A. Logg and J. Hake, *DOLFIN: a C++/Python finite element library, in Automated Solution of Differential Equations by the Finite Element Method*, Lect. Notes Comput. Sci. Eng., 84, Springer, 2012, pp. 173–225.

[2] N.A. Baker, *Improving implicit solvent simulations: a Poisson-centric view*, Curr. Opin. Struct. Biol. **15** (2005), 137–143.

[3] S. Balay, J. Brown, K. Buschelman, W.D. Gropp, D. Kaushik, M.G. Knepley, L.C. McInnes, B.F. Smith, and H. Zhang, *PETSc Web page*, 2012, http://www.mcs.anl.gov/petsc.

[4] M.V. Basilevsky and D.F. Parsons, *An advanced continuum medium model for treating solvation effects: Nonlocal electrostatics with a cavity*, J. Chem. Phys. **105** (1996), no. 9, 3734–3746.

[5] ———, *Nonlocal continuum solvation model with exponential susceptibility kernels*, J. Chem. Phys. **108** (1998), 9107–9113.

[6] ———, *Nonlocal continuum solvation model with oscillating susceptibility kernels: A nonrigid cavity model*, J. Chem. Phys. **108** (1998), 9114–9123.

[7] V. Basilevsky and G.N. Chuev, *Nonlocal solvation theories*, in Continuum Solvation Models in Chemical Physics: From Theory to Applications (Benedetta Mennucci and Roberto Cammi, eds.), Wiley, 2008, pp. 94–109.

[8] P.A. Bopp, A.A. Kornyshev, and G. Sutmann, *Static nonlocal dielectric function of liquid water*, Phys. Rev. Lett. **76** (1996), 1280–1283.

[9] S.C. Brenner and L.R. Scott, *The mathematical theory of finite element methods*, third ed., Springer-Verlag, New York, 2008.

[10] R. R. Dogonadze, E. Kálmán, A. A. Kornyshev, and J. Ulstrup, *The chemical physics of solvation, Part A: Theory of solvation*, Elsevier Science, Amsterdam, 1985.

[11] T.J. Dolinsky, J.E. Nielsen, J.A. McCammon, and N.A. Baker, *PDB2PQR: an automated pipeline for the setup of Poisson–Boltzmann electrostatics calculations*, Nucl. Acids Res. **32** (2004), no. suppl 2, W665.

[12] Z.H. Duan and R. Krasny, *An adaptive treecode for computing nonbonded potential energy in classical molecular systems*, J. Comput. Chem. **22** (2000), no. 2, 184–195.

[13] F. Fogolari, A. Brigo, and H. Molinari, *The Poisson-Boltzmann equation for biomolecular electrostatics: a tool for structural biology*, J. Mol. Recogn. **15** (2002), no. 6, 377–392.

[14] L. Greengard and V. Rokhlin, *A new version of the fast multipole method for the Laplace equation in three dimensions*, Acta Numer. **6** (1997), 229–269.

[15] A. Hildebrandt, R. Blossey, S. Rjasanow, O. Kohlbacher, and H.-P. Lenhof, *Novel formulation of nonlocal electrostatics*, Phys. Rev. Lett. **93** (2004), no. 10, 108104.

[16] M.J. Holst, *The Poisson-Boltzmann equation: Analysis and multilevel numerical solution*, http://scicomp.ucsd.edu/~mholst/pubs/dist/Hols94d.pdf (1994).

[17] B. Honig and A. Nicholls, *Classical electrostatics in biology and chemistry*, Science, **268** (1995), 1144–1149.

[18] W. Humphrey, A. Dalke, and K. Schulten, *VMD: Visual molecular dynamics*, J. Mol. Graph. **14** (1996), no. 1, 33–38.

[19] A.A. Kornyshev and G. Sutmann, *Nonlocal dielectric saturation in liquid water*, Phys. Rev. Lett. **79** (1997), 3435–3438.

[20] ———, *Nonlocal dielectric function of water: How strong are the effects of intramolecular charge form factors?*, J. Mol. Liq. **82** (1999), 151–160.

[21] V. Kralj-Iglič and A. Iglič, *A simple statistical mechanical approach to the free energy of the electric double layer including the excluded volume effect*, J. Phys. II **6** (1996), no. 4, 477–491.

[22] B. Li, *Minimization of electrostatic free energy and the Poisson-Boltzmann equation for molecular solvation with implicit solvent*, SIAM J. Math. Anal. **40** (2009), no. 6, 2536–2566.

[23] B.Z. Lu, Y.C. Zhou, M.J. Holst, and J.A. McCammon, *Recent progress in numerical methods for the Poisson-Boltzmann equation in biophysical applications*, Commun. Comput. Phys. **3** (2008), no. 5, 973–1009.

[24] R. C. McOwen, *Partial differential equations: Methods and applications*, 2nd ed., Prentice Hall, Upper Saddle River, NJ, 2003.

[25] M.T. Neves-Petersen and S.B. Petersen, *Protein electrostatics: A review of the equations and methods used to model electrostatic equations in biomolecules – Applications in biotechnology*, Biotech. Annu. Rev. **9** (2003), 315–395.

[26] P. Peterson, *F2py: a tool for connecting fortran and Python programs*, Internat. J. Comput. Sci. Engrg. **4** (2009), no. 4, 296–305.

[27] B. Roux and T. Simonson, *Implicit solvent models*, Biophys. Chem. **78** (1999), 1–20.

[28] A. Rubinstein and S. Sherman, *Influence of the solvent structure on the electrostatic interactions in proteins*, Biophys. J. **87** (2004), no. 3, 1544–1557.

[29] W. Rudin, *Functional analysis*, 2nd ed., McGraw-Hill, New York, 1991.

[30] L.R. Scott, M. Boland, K. Rogale, and A. Fernández, *Continuum equations for dielectric response to macro-molecular assemblies at the nano scale*, J. Phys. A, **37** (2004), 9791–9803.

[31] C. L Vizcarra and S. L Mayo, *Electrostatics in computational protein design*, Curr. Opin. Chem. Biol. **9** (2005), 622–626.

[32] Y. N. Vorobjev, *Advances in implicit models of water solvent to compute conformational free energy and molecular dynamics of proteins at constant pH*, Adv. Protein. Chem. Struct. Biol. **85** (2011), 281–322.

[33] S. Weggler, V. Rutka, and A. Hildebrandt, *A new numerical method for nonlocal electrostatics in biomolecular simulations*, J. Comput. Phys. **229** (2010), no. 11, 4059 – 4074.

[34] G.J. Wilson, R.K. Chan, D.W. Davidson, and E. Whalley, *Dielectric properties of ices II, III, V, and VI*, J. Chem. Phys. **43** (1965), 2384–2391.

[35] D. Xie, Y. Jiang, P. Brune, and L. Ridgway Scott, *A fast solver for a nonlocal dielectric continuum model*, SIAM J. Sci. Comput. **34** (2012), no. 2, B107–B126.

[36] D. Xie and H. W. Volkmer, *A modified nonlocal continuum electrostatic model for protein in water and its analytical solutions for ionic Born models*, Commun. Comput. Phys. **13** (2013), no. 1, 174–194.

[37] D. Xie and S. Zhou, *A new minimization protocol for solving nonlinear Poisson-Boltzmann mortar finite element equation*, BIT Num. Math. **47** (2007), 853–871.

[38] Z. Yu, M.J. Holst, Y. Cheng, and J.A. McCammon, *Feature-preserving adaptive mesh generation for molecular shape modeling and simulation*, J. Mol. Graph. Model. **26** (2008), no. 8, 1370–1380.