Journal of Computational and Applied Mathematics 307 (2016) 319–334

Contents lists available at ScienceDirect

Journal of Computational and Applied Mathematics

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

A new box iterative method for a class of nonlinear interface problems with application in solving Poisson–Boltzmann equation

Dexuan Xie *, Jinyong Ying

Department of Mathematical Sciences, University of Wisconsin-Milwaukee, Milwaukee, WI, 53201-0413, USA

HIGHLIGHTS

- Present a new box iterative method for a class of nonlinear interface problems.
- Obtain a new hybrid solver of Poisson–Boltzmann Equation (PBE) as application.
- Develop a new Newton-PCG-MG scheme for nonlinear boundary value problems.
- Obtain a simple series solution for Poisson ball test model with multiple charges.
- Validate the new PBE hybrid software and demonstrate its high performance.

ARTICLE INFO

Article history:
Received 18 July 2015
Received in revised form 5 January 2016

MSC:
92-08
65N30
65N06
65N55

Keywords:
Interface problems
Poisson–Boltzmann equation
Domain decomposition
Finite element methods
Finite difference methods
Electrostatics

ABSTRACT

In this paper, a new box iterative method for solving a class of nonlinear interface problems is proposed by intermixing linear and nonlinear boundary value problems based on a special seven-overlapped-boxes partition. It is then applied to the construction of a new finite element and finite difference hybrid scheme for solving the Poisson–Boltzmann equation (PBE) — a second order nonlinear elliptic interface problem for computing electrostatics of an ionic solvated protein. Furthermore, a modified Newton minimization algorithm accelerated by a multigrid preconditioned conjugate gradient method is presented to efficiently solve each involved nonlinear boundary value problem. In addition, the analytical solution of a Poisson dielectric test model with a spherical solute region containing multiple charges is expressed in a simple series of Legendre polynomials, resulting in a new PBE test model that works for a large number of point charges. The new PBE hybrid solver is programmed as a software package, and numerically validated on the new PBE test model with 892 point charges. It is also compared to a commonly used finite difference scheme in the accuracy of computing solution and electrostatic free energy for three proteins with up to 2124 atomic charges. Numerical results on six proteins demonstrate its high performance in comparison to the PBE finite element program package reported in Xie (2014).

1. Introduction

The classical alternating Schwarz method was introduced by Schwarz in [1] for the purpose of proving the solution existence and uniqueness of a Poisson boundary value problem in a domain that can be decomposed as the union of two

* Corresponding author.
E-mail address: dxie@uwm.edu (D. Xie).

http://dx.doi.org/10.1016/j.cam.2016.01.005
0377-0427/

Fig. 1. Roles of our seven-overlapped-boxes partition in the development of hybrid nonlinear interface solvers.

"simpler" domains. With the development of parallel computer architectures in 1980s, it was extensively re-studied as one important numerical technique for solving various boundary value problems, known as domain decomposition methods and preconditioners [2–5]. Its essential idea is to divide a complicated problem into simple subproblems to conquer the problem. In this paper, we use this idea to construct a new box iterative method for solving a class of nonlinear interface boundary value problem, which arises frequently from steady state heat diffusion problems with two different diffusion parameters and electrostatic problems with two different permittivity parameters. As one important application, this new nonlinear iterative method is used to construct a new finite element and finite difference hybrid scheme to solve the Poisson–Boltzmann equation (PBE)—a second order nonlinear elliptic interface problem with singular source terms. PBE has been widely applied to the calculation of electrostatics for protein in ionic solvent [6–11].

The finite element method is a natural choice to deal with a flux interface condition on a complex interface (e.g., a molecular surface in the case of PBE) [12–16]. But, because of using an interface fitted unstructured mesh, its implementation requires a large amount of extra computer memory to store mesh data and the nonzero entries of coefficient matrices. A system of finite element equations defined on an unstructured mesh also becomes much less efficient to solve than a system of finite difference equations defined on a Cartesian grid mesh. In fact, a Cartesian grid mesh has simple data structures, can be generated cheaply, and can lead to standard finite difference stencils. As such, it has been widely used to develop fast linear and nonlinear iterative schemes including geometric multigrid iterative schemes [17], multigrid preconditioned Krylov subspace methods [18], Newton multigrid methods [19], and multigrid preconditioned Newton Krylov methods [20]. To take advantages of these fast iterative solvers and to reduce the cost of mesh generation, immersed boundary/interface methods in finite difference formulation [21–25], virtual node methods [26], and immersed finite element methods [27] have been developed to solve linear interface problems based on uniform Cartesian grid meshes.

We recently proposed a special seven-overlapped-boxes partition to hybridize finite element and finite difference methods in the numerical solution of a linear interface problem [28]. As illustrated in Fig. 1, we can also use this special box partition to intermix a nonlinear problem with its linearized problem in the case of solving a nonlinear interface problem. This observation motivated us to develop the new box iterative method for solving the nonlinear interface problem. That is, we can restrict the nonlinear interface problem to a much smaller subdomain, the central box, reduce it to a nonlinear boundary value problem on each neighboring box, and then approximate it as a linear boundary value problem when the solution is small enough. Moreover, different numerical techniques can be applied to different boxes to turn the box iterative method into an efficient hybrid nonlinear solver.

As one important application, in this paper, we use this box iterative method to develop a new PBE hybrid solver to reduce the computing cost of a finite element solution decomposition PBE solver, called SDPB, reported in [29]. In SDPB, the PBE solution u is constructed as a sum of three functions G, Ψ, and $\tilde{\Phi}$ with G being a given function that collects all the singularity points of u, Ψ a solution of a linear interface problem, and $\tilde{\Phi}$ a solution of a nonlinear interface problem (see (4.3)). Thus, we can apply the new box iterative method to the calculation of Ψ and $\tilde{\Phi}$ to yield the new PBE hybrid solver (see Algorithm 4.1). While SDPBS is adopted to solve each nonlinear interface problem on the central box, we construct an efficient modified Newton minimization algorithm to solve a nonlinear boundary value problem on each neighboring box based on the finite difference approach (see Section 5). In particular, the nonlinear boundary value problem is shown to be equivalent to a nonlinear variational problem with a unique minimizer (see Theorem 5.1), and each Newton equation of the modified Newton minimization algorithm is reformulated from a variational form into a linear boundary value problem (see (5.7)), making it possible to calculate each Newton search direction by a fast finite difference solver—a multigrid V-cycle preconditioned conjugate gradient method (PCG-MG) developed in [28]. Together with a line search scheme for determining the steplength of each search direction, this modified Newton method, which will be called Newton-PCG-MG for clarity, can become globally convergent in the calculation of $\tilde{\Phi}$ on each neighboring box.

To validate a PBE solver, we construct a new PBE test model (see (6.1)) by using the analytical solution of a Poisson dielectric test model with a spherical solute region D_p containing multiple charges (see (6.2)). So far, the Born ball model [30], which is a Poisson dielectric test model with one central charge only, was employed to construct a PBE test model [29,31]. The Kirkwood's dielectric sphere model [32], which is a linearized PBE test model with a spherical D_p containing multiple point charges, was used to validate the matched interface and boundary PBE solver (MIBPB) [33], but the tests were done by using only six point charges due to the expensive cost of computing the analytical solution of the Kirkwood's model, which is given as a double series of associated Legendre polynomial P_n^m (i.e., a sum from $n = 0$ to ∞ and $m = -n$ to n; see [33, (A6), (A8) and (A11)]). Although the analytical solution of the Poisson test model can be followed from the Kirkwood's model as a special case, to reduce the computing cost, we recalculate it using different techniques, such as superposition principle and rotational symmetry mapping, and express the analytical solution as a simple series of Legendre polynomials P_n (see

Theorem 6.1). As a result, a validation test can be done with a large number of point charges on a mesh with a large number of mesh points.

We programmed our new PBE hybrid solver in `Python` and `Fortran` based on the software packages developed in [28,29]. In particular, we used the PCG-MG program, the box partition and mesh generation program, and the program for computing G and its gradient vector ∇G from [28], and adopted the software SDPBS from [29] to solve each nonlinear interface problem on the central box. Note that in SDPB, the PCG using the incomplete LU preconditioning (PCG-ILU) from the `PETSc` library [34] has been used to solve each system of finite element equations. We programmed our Newton-PCG-MG in `Fortran` without storing any mesh data or coefficient matrices of finite difference systems. We also programmed the new simple series solution of the Poisson test model in `Fortran` to quickly calculate its values on a large set of mesh points. Here a set of charge positions and numbers can be input directly from a PQR file of a protein, which simplifies the construction of a validation test with a large number of point charges. We converted the `Fortran` subroutines to `Python` modules by the Fortran-to-Python interface generator `f2py` (http://cens.ioc.ee/projects/f2py2e/).

We validated the new PBE solver on the new PBE test model with 892 point charges from a protein (4PTI) based on three nested meshes (see Fig. 3). In these tests, the relative errors of numerical solutions were found to reduce from 9.55×10^{-2} to 5.63×10^{-3} as mesh size h was reduced from 0.25 to 0.065, verifying that the new PBE solver has a second order of convergence rate in terms of h when it uses a linear finite element and a second order central finite difference approximation. Moreover, our `Fortran` subroutine for computing the analytical solution was found to be efficient. For example, it took only about 40 s to complete the calculation of 136,512 solution values for 892 atoms on one processor of our Mac Pro Workstation with the 3.7 GHz Quad-Core Intel Xeon E5 and 64 GB memory. Here the series solution was calculated approximately by only using its first 20 terms, which was found to be large enough for these tests since it resulted in a relative error less than $O(10^{-4})$ (see Fig. 4).

To demonstrate the performance of our new hybrid PBE program package, we made numerical tests for six proteins, and repeated these tests using SDPB. While the relative errors between the numerical solutions produced by the new PBE solver and SDPBS were found to be less than 10^{-7}, the total CPU time of SDPBS was reduced by about 58%–70% (see Table 4). These tests also demonstrated that our new hybrid box iterative method had a fast rate of convergence, which took only up to seven iterations to reduce the errors from $O(10^3)$ to $O(10^{-8})$ (see Fig. 6). In addition, they confirmed that our Newton-PCG-MG retained a quadratic rate of error reductions (see Fig. 7).

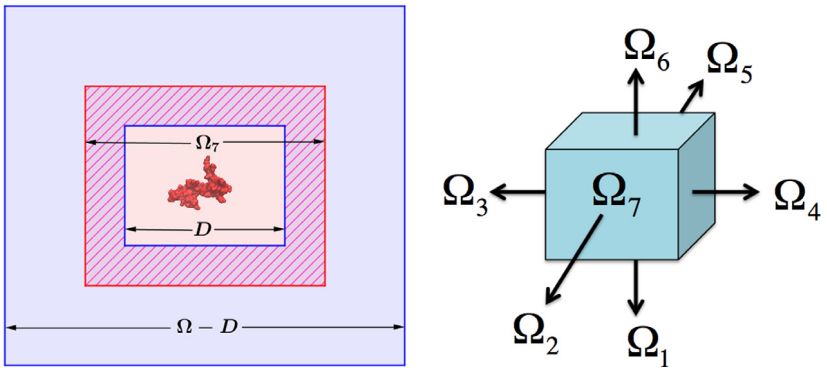
Finally, we compared the numerical accuracy of our hybrid solver with that of a traditional finite difference PBE scheme proposed in [35,36] in the numerical solution of Poisson test model (i.e., the PBE test model (6.1) with $\kappa = 0$) and the calculation of electrostatic solvation free energy (one important application of PBE). This finite difference scheme was commonly adopted to the popular PBE software packages such as DelPhi [37] and PBEQ [38]. Although the Poisson test model is a special case of PBE, its numerical solution involves the typical algorithm issues occurred in solving PBE. For example, how to deal with the flux interface condition, and how to overcome the solution singularity induced from the Dirac delta distributions. Since we know the analytical solution of Poisson test model, we can study these algorithm issues rigorously in terms of the relative errors of numerical solutions (see (7.1)) and predicted electrostatic solvation free energies (see (7.4)). To do so, we programmed the traditional finite difference scheme. Note that several improved finite difference PBE schemes were developed recently by the flux interface condition and solution decomposition techniques [39,40]. To mimic some convergence behaviors of these improved finite difference PBE schemes, we implemented our SDPBS finite element solver based on a uniform tetrahedral mesh. Comparison tests were done on these three solvers for three proteins with up to 2124 atomic charges. Test results (see Table 2) show that our hybrid solver has a much higher accuracy than the other two solvers in the calculation of numerical solutions and solvation free energy. Interestingly, the finite difference scheme was found to produce numerical solutions in a low accuracy, and not to guarantee any convergence. Gladly, it was found to have a satisfactory accuracy in the calculation of electrostatic solvation free energy. Hence, the finite difference scheme is still valuable in the application problems involving electrostatic solvation energies due to its simplicity and efficiency in implementation.

The rest of the paper is organized as follows. Section 2 defines the nonlinear interface problem. Section 3 describes the new nonlinear box iterative method. Section 4 presents the new PBE hybrid solver. Section 5 constructs the Newton-PCG-MG algorithm. Section 6 presents the new PBE test model and the new series solution of the Poisson test model. Finally, the new PBE program package and numerical results are reported in Section 7.

2. A class of nonlinear interface problems

Let Ω be a large rectangular box domain, and split into two subdomains, D_p and D_s, with D_p being surrounded by D_s and Γ denoting the interface between D_p and D_s. We consider a general nonlinear interface problem as follows:

$$\begin{cases} -\epsilon_p \Delta w(\mathbf{r}) = f_p(\mathbf{r}), & \mathbf{r} \in D_p, \\ -\epsilon_s \Delta w(\mathbf{r}) + \beta(w) = f_s(\mathbf{r}), & \mathbf{r} \in D_s, \\ w(\mathbf{s}^+) = w(\mathbf{s}^-), \quad \epsilon_s \dfrac{\partial w(\mathbf{s}^+)}{\partial \mathbf{n}(\mathbf{s})} = \epsilon_p \dfrac{\partial w(\mathbf{s}^-)}{\partial \mathbf{n}(\mathbf{s})} + \zeta(\mathbf{s}), & \mathbf{s} \in \Gamma, \\ w(\mathbf{s}) = g(\mathbf{s}), & \mathbf{s} \in \partial\Omega, \end{cases} \tag{2.1}$$



(a) Selection of D and Ω_7. (b) The ordering of seven boxes Ω_i.

Fig. 2. An illustration of cubic region D, central box Ω_7, and the ordering of seven overlapped boxes.

where ϵ_p and ϵ_s are two positive constants, f_p, f_s, ζ, and g are continuous functions, $\beta(w)$ denotes a nonlinear function of w, $\mathbf{n}(\mathbf{s})$ is the unit outward normal vector of D_p, $\Delta = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial z^2}$ is the Laplace operator for $\mathbf{r} = (x, y, z)$, $\frac{\partial w(\mathbf{s})}{\partial \mathbf{n}(\mathbf{s})} = \nabla w(\mathbf{s}) \cdot \mathbf{n}(\mathbf{s})$ with $\nabla = \left(\frac{\partial}{\partial x}, \frac{\partial}{\partial y}, \frac{\partial}{\partial z}\right)$, $w(\mathbf{s}^{\pm}) = \lim_{t \to 0^+} w(\mathbf{s} \pm t\mathbf{n})$, and $\frac{\partial w(\mathbf{s}^{\pm})}{\partial \mathbf{n}(\mathbf{s})} = \lim_{t \to 0^+} \frac{\partial w(\mathbf{s} \pm t\mathbf{n}(\mathbf{s}))}{\partial \mathbf{n}(\mathbf{s})}$.

We assume that the nonlinear interface problem (2.1) has two special properties:

P1. The solution $w(\mathbf{r}) \to 0$ as $|\mathbf{r}| \to \infty$.
P2. The coefficient function $\beta(w)$ has the Taylor expansion

$$\beta(w) = a_0(\mathbf{r}) + a_1(\mathbf{r})w + O(|w|^2) \quad \forall \mathbf{r} \in D_s, \tag{2.2}$$

where a_0 and a_1 are two continuous functions on D_s.

Clearly, because of P1, there exists a sufficiently large cubic box, D, such that

$$D_p \subset D \subset \Omega, \quad \text{and} \quad |w(\mathbf{r})| \ll 1 \quad \forall \mathbf{r} \in \Omega \setminus D. \tag{2.3}$$

With P2, we then can construct a linear boundary value problem on $\Omega \setminus D$ as follows:

$$\begin{cases} -\epsilon_s \Delta w(\mathbf{r}) + a_1(\mathbf{r})w = f_s(\mathbf{r}) - a_0(\mathbf{r}), & \mathbf{r} \in \Omega \setminus D, \\ w(\mathbf{s}) = q(\mathbf{s}), & \mathbf{s} \in \partial D, \\ w(\mathbf{s}) = g(\mathbf{s}), & \mathbf{s} \in \partial \Omega, \end{cases} \tag{2.4}$$

where ∂D denotes the boundary of D, and q is a boundary function on ∂D, which is usually unknown. Even so, the above linear problem can be valuable in the construction of an iterative scheme for solving the nonlinear interface problem (2.1) through a proper selection of q as what is done in the next section.

3. The new nonlinear iterative method

In this section, we construct a new box iterative method to solve the nonlinear interface problem (2.1) based on a special seven-overlapped-boxes partition of Ω proposed in [28]. For simplicity, we assume that Ω is a cubic box with side length L. We properly select a cubic region D, and partition Ω into seven overlapped boxes Ω_i such that

$$\text{(a) } \Omega_p \subset D \subset \Omega_7 \subset \Omega, \qquad \text{(b) } \Omega \setminus D = \bigcup_{j=1}^{6} \Omega_j, \qquad \text{(c) } \Omega_7 \setminus D = \bigcup_{j=1}^{6} \Omega_j \cap \Omega_7. \tag{3.1}$$

As illustrated in Fig. 2, Ω_7 is the central box, and $\Omega_7 \setminus D$ gives the overlapped part of Ω_7 with its six neighboring boxes Ω_i for $i = 1$ to 6. Note that the number of seven is the smallest number to satisfy the partition conditions of (3.1), and ordering the central box as the 7th box is to fully use the previous updates to solve an interface problem on Ω_7.

Let $W_i^{(k)}$ denote the kth iterate on box Ω_i, and an initial iterate $W_i^{(0)}$ be given. For $k = 0, 1, 2, \ldots$, we define the new box iterative method for solving (2.1) by

$$W_i^{(k+1)} = (1 - \omega)W_i^{(k)} + \omega W_{i,k} \quad \text{on } \Omega_i \text{ for } i = 1, 2, \ldots, 7, \tag{3.2}$$

where $\omega \in (1, 2)$ is an over-relaxation parameter, $W_{i,k}$ with $i = 1$ to 6 is a solution of the nonlinear boundary value problem on each neighboring box Ω_i:

$$\begin{cases} -\epsilon_s \Delta w(\mathbf{r}) + \beta(w) = f_s(\mathbf{r}) \quad \text{in } \Omega_i, \\ w(\mathbf{s}) = W_j^{(k)}(\mathbf{s}) \quad \text{on } \partial\Omega_i \cap \Omega_j \text{ when } \partial\Omega_i \cap \Omega_j \neq \varnothing \text{ for } j = i+1 \text{ to } 7, \\ w(\mathbf{s}) = W_j^{(k+1)}(\mathbf{s}) \quad \text{on } \partial\Omega_i \cap \Omega_j \text{ when } \partial\Omega_i \cap \Omega_j \neq \varnothing \text{ for } j = 1 \text{ to } i-1, \\ w(\mathbf{s}) = g(\mathbf{s}) \quad \text{on } \partial\Omega_i \cap \partial\Omega, \end{cases} \tag{3.3}$$

and $W_{7,k}$ is a solution of the nonlinear interface boundary value problem on the central box Ω_7:

$$\begin{cases} -\epsilon_p \Delta w(\mathbf{r}) = f_p(\mathbf{r}) \quad \text{in } D_p, \\ -\epsilon_s \Delta w(\mathbf{r}) + \beta(w) = f_s(\mathbf{r}) \quad \text{in } \Omega_7 \cap D_s, \\ w(\mathbf{s}^+) = w(\mathbf{s}^-), \qquad \epsilon_s \dfrac{\partial w(\mathbf{s}^+)}{\partial \mathbf{n}(\mathbf{s})} = \epsilon_p \dfrac{\partial w(\mathbf{s}^-)}{\partial \mathbf{n}(\mathbf{s})} + \zeta(\mathbf{s}) \quad \text{on } \Gamma, \\ w(\mathbf{s}) = W_i^{(k+1)}(\mathbf{s}) \quad \text{on } \partial\Omega_7 \cap \Omega_i \text{ for } i = 1 \text{ to } 6. \end{cases} \tag{3.4}$$

Here, $\partial\Omega_i$ is the boundary of Ω_i, and the updates $W_i^{(k+1)}$ from the six neighboring boxes have been used to update the boundary value condition on $\partial\Omega_7$.

Following what is done in (2.4), we linearize the nonlinear problem (3.3) to yield a linear boundary value problem on each neighboring box Ω_i for $i = 1$ to 6 as follows:

$$\begin{cases} -\epsilon_s \Delta w(\mathbf{r}) + a_1(\mathbf{r}) w(\mathbf{r}) = f_s(\mathbf{r}) - a_0(\mathbf{r}) \quad \text{in } \Omega_i, \\ w(\mathbf{s}) = W_j^{(k)}(\mathbf{s}) \quad \text{on } \partial\Omega_i \cap \Omega_j \text{ when } \partial\Omega_i \cap \Omega_j \neq \varnothing \text{ for } j = i+1 \text{ to } 7, \\ w(\mathbf{s}) = W_j^{(k+1)}(\mathbf{s}) \quad \text{on } \partial\Omega_i \cap \Omega_j \text{ when } \partial\Omega_i \cap \Omega_j \neq \varnothing \text{ for } j = 1 \text{ to } i-1, \\ w(\mathbf{s}) = g(\mathbf{s}) \quad \text{on } \partial\Omega_i \cap \partial\Omega. \end{cases} \tag{3.5}$$

To adaptively select the linear model (3.5) as a substitute for the nonlinear model (3.3), we propose the following Test rule 1.

Test rule 1 (*Selection of Linear Model* (3.5))**.** *Let $W_{i,a}$ be an average value of $W_7^{(k)}$ on the boundary $\partial\Omega_i \cap \Omega_7$. For a set of mesh points, $\{\mathbf{r}^j\}_{j=1}^{N_i}$, from $\partial\Omega_i \cap \Omega_7$, $W_{i,a}$ is calculated by*

$$W_{i,a} = \frac{1}{N_i} \sum_{j=1}^{N_i} |W_7^{(k)}(\mathbf{r}^j)| \quad \textit{for } i = 1, 2, \ldots, 6. \tag{3.6}$$

If $W_{i,a} < \eta$, $W_i^{(k+1)}$ is defined by the linear boundary value problem (3.5); otherwise, the nonlinear problem (3.3) is retained to define $W_i^{(k+1)}$. A default value of η is set as 0.1 in our implementation.

The new box iterative method is said to be convergent if it satisfies

$$\sqrt{\sum_{i=1}^{7} \|W_i^{(k+1)} - W_i^{(k)}\|^2} \leq \epsilon, \tag{3.7}$$

where $\|\cdot\|$ is the Euclidean norm, and ϵ is set as 10^{-7} by default.

For clarity, we summary our new box iterative method in Algorithm 3.1.

Algorithm 3.1 (*The New Box Iterative Method for Nonlinear Interface Problem* (2.1))**.** The new box iterative method defined in (3.2) can be implemented in the following five steps:

Step 1. Select a cubic region D and construct seven overlapped boxes, Ω_i for $i = 1, 2, \ldots, 7$, to satisfy (3.1).

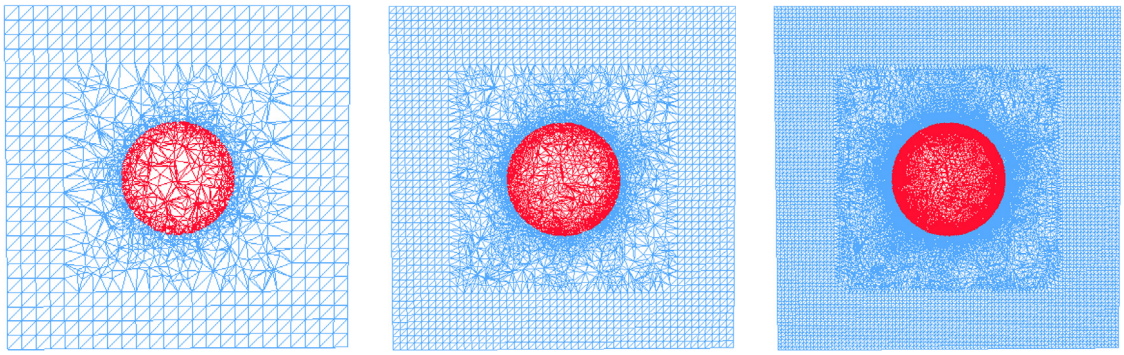
Step 2. Set $k = 0$ and select the initial iterate $W_7^{(0)} = 0$.

Step 3. Calculate $W_i^{(k+1)}$ for $i = 1$ to 6 according to Test rule 1.

Step 4. Solve the nonlinear interface problem (3.4) for $W_7^{(k+1)}$.

Step 5. Check convergence: If the termination rule (3.7) holds, output $\{W_i^{(k+1)}\}_{i=1}^7$ as a numerical solution of (2.1); otherwise, increase k by 1 and go back to Step 3.

Since our new box iterative method is a special alternating Schwarz method, its convergence can be followed directly from the standard domain decomposition theory [41,42]. Because of using only seven regular boxes, it has a fast rate of convergence. Its computing costs can be further reduced through properly selecting the over-relaxation parameter ω and the linearized model (3.5). Moreover, it can be implemented by hybrid techniques. For example, we can turn it into a finite



(a) $\Omega_{7,h}$ with $h = 0.25$. (b) $\Omega_{7,h}$ with $h = 0.125$. (c) $\Omega_{7,h}$ with $h = 0.0625$.

Fig. 3. Cross-section views (on the xy-plane) of the three nested meshes of the central box Ω_7 for validation tests on the new PBE hybrid solver. Here the solute region D_p is marked in red. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

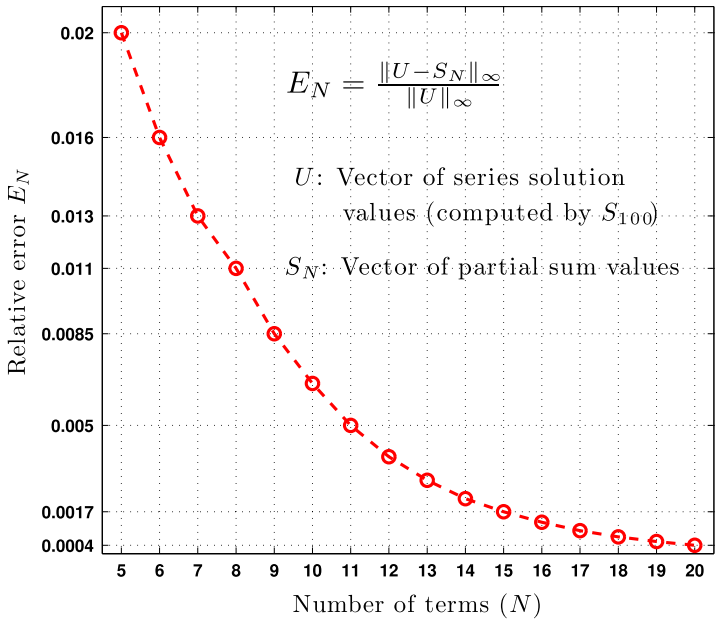


Fig. 4. Relative error E_N of the partial sum S_N of the series solution of the Poisson model (6.2) with 892 point charges on the mesh with $h = 0.25$ and 120,887 mesh points. Here S_N is a sum of the first N terms of the series solution of (6.5).

element and finite difference hybrid solver by selecting a fast finite element algorithm to solve each interface problem and a fast finite difference algorithm to solve each linear/nonlinear boundary value problem. In this case, the interface problem (3.4) can be reformulated as a nonlinear variational problem in the form

Find $w \in H^1(\Omega_7)$ subject to $w = W_i^{(k+1)}$ on $\partial\Omega_7 \cap \Omega_i$ for $i = 1$ to 6 such that

$$b(w, v) = l(v) \quad \forall v \in H_0^1(\Omega), \tag{3.8}$$

where $H^1(\Omega)$ and $H_0^1(\Omega)$ are two regular Sobolev function spaces [43], $b(w, v)$ is defined by

$$b(w, v) = \epsilon_p \int_{D_p} \nabla w \cdot \nabla v d\mathbf{r} + \epsilon_s \int_{D_s \cap \Omega_7} \nabla w \cdot \nabla v d\mathbf{r} + \int_{D_s \cap \Omega_7} \beta(w) v(\mathbf{r}) d\mathbf{r},$$

and $l(v)$ is a linear functional defined by

$$l(v) = \int_\Gamma \zeta(\mathbf{s}) v(\mathbf{s}) d\mathbf{s} + \int_{D_s \cap \Omega_7} f_s(\mathbf{r}) v(\mathbf{r}) d\mathbf{r} + \int_{D_p} f_p(\mathbf{r}) v(\mathbf{r}) d\mathbf{r}.$$

To simplify the data exchange between a finite element solver on the central box Ω_7 and a finite difference solver on each neighboring box, a hybrid mesh of Ω_7 is constructed (see Fig. 3 for example) such that the uniform tetrahedral mesh part (for the overlapped part $\Omega_7 \setminus D$) shares the same mesh points from the uniform finite difference meshes of the six neighboring boxes. Consequently, the data exchange can be carried out easily and efficiently.

4. Application in solving Poisson–Boltzmann equation

As one important application of Algorithm 3.1, in this section, we develop a new finite element and finite difference hybrid algorithm to solve a dimensionless PBE model as follows:

$$\begin{cases} -\epsilon_p \Delta \Phi(\mathbf{r}) = \alpha \sum_{j=1}^{n_p} z_j \delta(\mathbf{r} - \mathbf{r}_j), & \mathbf{r} \in D_p, \\ -\epsilon_s \Delta \Phi(\mathbf{r}) + \kappa^2 \sinh(\Phi(\mathbf{r})) = 0, & \mathbf{r} \in D_s, \\ \Phi(\mathbf{s}^+) = \Phi(\mathbf{s}^-), \quad \epsilon_s \dfrac{\partial \Phi(\mathbf{s}^+)}{\partial \mathbf{n}(\mathbf{s})} = \epsilon_p \dfrac{\partial \Phi(\mathbf{s}^-)}{\partial \mathbf{n}(\mathbf{s})}, & \mathbf{s} \in \Gamma, \\ \Phi(\mathbf{s}) = g(\mathbf{s}), & \mathbf{s} \in \partial\Omega, \end{cases} \tag{4.1}$$

where D_p hosts a protein molecule with n_p atoms, $\mathbf{r}_j$ and z_j are the position and charge number of the jth atom, α and κ are two physical constants, and $\delta(\mathbf{r} - \mathbf{r}_j)$ denotes the Dirac delta distribution at charge position $\mathbf{r}_j$. In this case, D_p and D_s are called the protein and solvent regions, respectively, which are treated as two different dielectric media, and ϵ_p and ϵ_s are two corresponding permittivity constants.

When length is measured in angstroms (1angstrom $= 10^{-10}$ m), α and κ have the following values:

$$\alpha = \frac{10^{10} e_c^2}{\epsilon_0 k_B T}, \quad \text{and} \quad \kappa^2 = 2\frac{10^{-17} e_c^2 N_A I_s}{\epsilon_0 k_B T}, \tag{4.2}$$

where e_c, ϵ_0, k_B, N_A, and I_s are the electron charge, the permittivity of vacuum, the Boltzmann constant, the Avogadro number, and the ionic strength, respectively. Here I_s is measured in mole/liter. The solution Φ of PBE gives the electrostatic potential function (in units $k_B T/e_c$) of an electrostatic field caused by a biomolecule (such as a protein molecule) in a symmetric 1:1 ionic solvent (e.g., a salt solution with sodium (Na^+) and chloride (Cl^-) ions).

As shown in our previous work [29], the PBE solution Φ can be constructed by

$$\Phi = G + \Psi + \tilde{\Phi}, \tag{4.3}$$

where G is given by

$$G(\mathbf{r}) = \frac{\alpha}{4\pi\epsilon_p} \sum_{j=1}^{n_p} \frac{z_j}{|\mathbf{r} - \mathbf{r}_j|}, \tag{4.4}$$

Ψ is a solution of the linear interface boundary value problem

$$\begin{cases} \Delta \Psi(\mathbf{r}) = 0, & \mathbf{r} \in D_p \cup D_s, \\ \Psi(\mathbf{s}^+) = \Psi(\mathbf{s}^-), \quad \epsilon_s \dfrac{\partial \Psi(\mathbf{s}^+)}{\partial \mathbf{n}(\mathbf{s})} = \epsilon_p \dfrac{\partial \Psi(\mathbf{s}^-)}{\partial \mathbf{n}(\mathbf{s})} + (\epsilon_p - \epsilon_s) \dfrac{\partial G(\mathbf{s})}{\partial \mathbf{n}(\mathbf{s})}, & \mathbf{s} \in \Gamma, \\ \Psi(\mathbf{s}) = g(\mathbf{s}) - G(\mathbf{s}), & \mathbf{s} \in \partial\Omega, \end{cases} \tag{4.5}$$

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

$$\begin{cases} \Delta \tilde{\Phi}(\mathbf{r}) = 0, & \mathbf{r} \in D_p, \\ -\epsilon_s \Delta \tilde{\Phi}(\mathbf{r}) + \kappa^2 \sinh(\tilde{\Phi} + \Psi + G) = 0, & \mathbf{r} \in D_s, \\ \tilde{\Phi}(\mathbf{s}^+) = \tilde{\Phi}(\mathbf{s}^-), \quad \epsilon_s \dfrac{\partial \tilde{\Phi}(\mathbf{s}^+)}{\partial \mathbf{n}(\mathbf{s})} = \epsilon_p \dfrac{\partial \tilde{\Phi}(\mathbf{s}^-)}{\partial \mathbf{n}(\mathbf{s})}, & \mathbf{s} \in \Gamma, \\ \tilde{\Phi}(\mathbf{s}) = 0, & \mathbf{s} \in \partial\Omega. \end{cases} \tag{4.6}$$

Obviously, Algorithm 3.1 can be applied to the calculation of Ψ and $\tilde{\Phi}$, and the related nonlinear boundary value problem on a neighboring box Ω_i for $i = 1$ to 6 can be approximated by the linear boundary value problem:

$$\begin{cases} -\epsilon_s \Delta \tilde{\Phi}(\mathbf{r}) + \kappa^2 \tilde{\Phi}(\mathbf{r}) = -\kappa^2(\Psi + G), \quad \mathbf{r} \in \Omega_i, \\ \tilde{\Phi}(\mathbf{s}) = \tilde{\Phi}_j^{(k)}(\mathbf{s}) \quad \text{on } \partial\Omega_i \cap \Omega_j \text{ when } \partial\Omega_i \cap \Omega_j \neq \varnothing \text{ for } j = i + 1\text{–}7, \\ \tilde{\Phi}(\mathbf{s}) = \tilde{\Phi}_j^{(k+1)}(\mathbf{s}) \quad \text{on } \partial\Omega_i \cap \Omega_j \text{ when } \partial\Omega_i \cap \Omega_j \neq \varnothing \text{ for } j = 1 \text{ to } i - 1, \\ \tilde{\Phi}(\mathbf{s}) = 0 \quad \text{on } \partial\Omega_i \cap \partial\Omega \end{cases} \tag{4.7}$$

provided that $|\tilde{\Phi} + \Psi + G| \ll 1$ on the neighboring box Ω_i. Consequently, we obtain a new box iterative method for solving PBE as defined in Algorithm 4.1:

Algorithm 4.1 (*The New PBE Box Iterative Method*)**.** Let a diameter d of a ball that circumscribes D_p be given. The PBE solution Φ can be constructed in the following five steps:

Step 1. Set the boundary value function $g = 0$ (by default), select D and Ω as two cubic domains with side lengths being $2d$ and $10d$ (by default), respectively, and construct the seven overlapped boxes Ω_i for $i = 1, 2, \ldots, 7$ to satisfy (3.1).

Step 2. Calculate G on each box and ∇G on Ω_7.

Step 3. Solve the linear interface problem (4.5) for Ψ by Algorithm 3.1 using $f_p = 0, f_s = 0, \zeta(\mathbf{s}) = (\epsilon_p - \epsilon_s)\frac{\partial G(\mathbf{s})}{\partial \mathbf{n}(\mathbf{s})}$, and $\beta = 0$.

Step 4. Solve the nonlinear interface problem (4.6) for $\tilde{\Phi}$ by Algorithm 3.1 using $f_p = 0, f_s = 0, \beta = \kappa^2 \sinh(\tilde{\Phi} + \Psi + G)$, and $\zeta(\mathbf{s}) = 0$. Here, (3.5) is replaced by (4.7), and $W_{i,a}$ is computed by

$$W_{i,a} = \frac{1}{N_i}\sum_{j=1}^{N_i}\left|\tilde{\Phi}_7^{(k)}(\mathbf{r}^j) + \Psi(\mathbf{r}^j) + G(\mathbf{r}^j)\right| \quad \text{for } i = 1, 2, \ldots, 6.$$

Step 5. Construct Φ by the solution decomposition $\Phi = G + \Psi + \tilde{\Phi}$.

In the numerical implementation of Algorithm 4.1, we generate the interface Γ, a value of diameter d, an interface-matched tetrahedral mesh of Ω_7, and a uniform mesh of each neighboring box Ω_i from the mesh generation programs developed in [29,28]. We then adopt the software SDPBS developed in [29] to solve each nonlinear interface problem on Ω_7. Each linear boundary value problem on a neighboring box is solved numerically by the software PCG-MG developed in [28] such that the relative residual norm is less than 10^{-8}. To solve each nonlinear boundary value problem on a neighboring box, we develop a Newton-PCG-MG algorithm as described in the next section. In this way, from Algorithm 4.1 it yields a new finite element and finite difference hybrid PBE solver.

5. A new Newton-PCG-MG method

In this section, we present a new Newton-PCG-MG method for solving a nonlinear boundary value problem arising from Step 4 of Algorithm 4.1. For a general purpose, we consider the following nonlinear boundary value problem

$$\begin{cases} -b\Delta u(\mathbf{r}) + c\sinh(u+\varphi) = f(\mathbf{r}) & \text{in } \Omega, \\ u(\mathbf{s}) = g(\mathbf{s}) & \text{on } \partial\Omega, \end{cases} \tag{5.1}$$

where f, φ, and g are three given functions, b and c are two positive constants, and Ω is a rectangular box. The following theorem can be regarded as a theoretical base for us to construct Newton-PCG-MG.

Theorem 5.1. *The solution u of the nonlinear boundary value problem (5.1) is a unique minimizer of the nonlinear functional J on the function space $V = u + H_0^1(\Omega)$*

$$J(u) = \min\{J(v) \mid v \in V\}, \tag{5.2}$$

where $u \in C^2(\Omega)$, $u(\mathbf{s}) = g(s)$ for $\mathbf{s} \in \partial\Omega$, and J is defined by

$$J(v) = \frac{b}{2}\int_\Omega \nabla v \cdot \nabla v d\mathbf{r} + c\int_\Omega \cosh(v+\varphi)d\mathbf{r} - \int_\Omega f v d\mathbf{r} \quad \forall v \in V. \tag{5.3}$$

Here $C^2(\Omega)$ is a function space consisting of functions with continuous second derivatives, and $H_0^1(\Omega)$ is a regular Sobolev function space [43].

Proof. Let $J'(u)$ and $J''(u)$ denote the first and second Fréchet-derivative of J at u, which are linear and bilinear continuous functionals on $H_0^1(\Omega)$, respectively. They can be found as follows:

$$J'(u)v = b\int_\Omega \nabla u \cdot \nabla v d\mathbf{r} + c\int_\Omega \sinh(u+\varphi)v d\mathbf{r} - \int_\Omega f v d\mathbf{r} \quad \forall v \in H_0^1(\Omega),$$

and

$$J''(u)(p, v) = b\int_\Omega \nabla p \cdot \nabla v d\mathbf{r} + c\int_\Omega \cosh(u+\varphi)pv d\mathbf{r} \quad \forall p, v \in H_0^1(\Omega).$$

We then can obtain a second order Taylor expansion of J in the form

$$J(u+v) = J(u) + J'(u)v + \frac{1}{2}J''(\xi)(v, v) \quad \forall v \in H_0^1(\Omega), \tag{5.4}$$

where $\xi \in V$ is given between u and $u + v$.

Clearly, $J''(w)(v, v) > 0$ for any $w \in V$ and $v \neq 0$. Since u is a solution of (5.1), it is easy to get that $J'(u)v = 0$, which gives the weak form of (5.1) as follows:

Find $u \in V$ such that

$$b\int_{\Omega} \nabla u \nabla v d\mathbf{r} + c\int_{\Omega} \sinh(u+\varphi)v d\mathbf{r} = \int_{\Omega} f v d\mathbf{r} \quad \forall v \in H_0^1(\Omega).$$

Hence, from the Taylor expansion (5.4) it implies that

$$J(u) \leq J(v) \quad \forall v \in V,$$

which follows that u is a minimizer of J over the function space V. □

We now define the modified Newton minimization method for solving (5.2) by

$$u^{(k+1)} = u^{(k)} + \lambda_k p_k, \quad k = 0, 1, 2, \ldots, \tag{5.5}$$

where $u^{(0)}$ is a given initial guess, λ_k is a step length determined by a line search algorithm to satisfy the condition

$$J(u^{(k+1)}) \leq J(u^{(k)}), \quad \text{or} \quad \|J'(u^{(k+1)})\| \leq \|J'(u^{(k)})\|,$$

and p_k is a search direction satisfying the Newton bilinear variational form

$$\text{Find } p_k \in H_0^1(\Omega) \text{ such that} \quad J''(u^{(k)})(p_k, v) = -J'(u^{(k)})v \ \forall v \in H_0^1(\Omega). \tag{5.6}$$

Since $u \in C^2(\Omega) \cap H^1(\Omega)$, the above Newton variational form can be easily reformulated as a linear boundary value problem in the differential form

$$\begin{cases} -b\Delta p + c\cosh(u^{(k)}+\varphi)p = b\Delta u^{(k)} - c\sinh(u^{(k)}+\varphi) + f & \text{in } \Omega, \\ p(\mathbf{s}) = 0 & \text{on } \partial\Omega. \end{cases} \tag{5.7}$$

Hence, the modified Newton minimization method (5.5) can also be implemented by finite difference techniques.

In Algorithm 4.1, we approximate each Newton equation of (5.7) as a system of second order finite difference equations based on a uniform mesh of each neighboring box Ω_i for $i = 1$ to 6, which can be written in the matrix form

$$H(u_i^{(k)})P = F(u_i^{(k)}), \quad k = 0, 1, 2, \ldots, \tag{5.8}$$

where P is a vector with components being the numerical values of p at interior mesh points, $H(u_i^{(k)})$ is a coefficient matrix of the finite difference system, $F(u_i^{(k)})$ is a vector with each component being a second order finite difference approximation to the function at the right hand side of Eq. (5.7) at each interior mesh point, and $\varphi = G + \Psi$.

Clearly, $F(u_i) = 0$ gives a system of nonlinear finite difference equations, which approximates the nonlinear boundary value problem (5.1) on each neighboring box Ω_i for $i = 1$ to 6.

We modify the modified Newton minimization method (5.5) as a new Newton-PCG-MG algorithm for solving a nonlinear boundary value problem on each neighboring box Ω_i for $i = 1$ to 6 when each linear system of (5.8) is solved approximately by the PCG-MG program until its relative residual is less than 10^{-8}. Here the Newton-PCG-MG iterative sequence, $\{u_i^{(k+1)}\}$, is required to satisfy the convergence rule

$$\|F(u_i^{(k+1)})\| < \epsilon \quad \text{or} \quad \|u_i^{(k+1)} - u_i^{(k)}\| < \epsilon, \tag{5.9}$$

where ϵ is a convergence tolerance ($\epsilon = 10^{-7}$ by default).

6. A new PBE test model with multiple charges

To validate a PBE solver, in this section, we present a PBE test model as follows:

$$\begin{cases} -\epsilon_p \Delta\Phi(\mathbf{r}) = \alpha \sum_{j=1}^{n_p} z_j \delta(\mathbf{r} - \mathbf{r}_j), & \mathbf{r} \in D_p, \\ -\epsilon_s \Delta\Phi(\mathbf{r}) + \kappa^2 \sinh(\Phi) = \kappa^2 \sinh(u(\mathbf{r})), & \mathbf{r} \in D_s, \\ \Phi(\mathbf{s}^+) = \Phi(\mathbf{s}^-), \quad \epsilon_s \dfrac{\partial\Phi(\mathbf{s}^+)}{\partial\mathbf{n}(\mathbf{s})} = \epsilon_p \dfrac{\partial\Phi(\mathbf{s}^-)}{\partial\mathbf{n}(\mathbf{s})}, & \mathbf{s} \in \Gamma, \\ \Phi(\mathbf{s}) = u(\mathbf{s}) & \text{on } \partial\Omega, \end{cases} \tag{6.1}$$

where $D_p = \{\mathbf{r} \mid |\mathbf{r}| < a\}$, $\Gamma = \{\mathbf{r} \mid |\mathbf{r}| = a\}$, $D_s = \Omega \setminus (D_p \cup \Gamma)$, and u is a solution of the following Poisson test model

$$\begin{cases} -\epsilon_p \Delta u(\mathbf{r}) = \rho(\mathbf{r}) & \text{for } |\mathbf{r}| < a, \\ -\epsilon_s \Delta u(\mathbf{r}) = 0, & \text{for } |\mathbf{r}| > a, \\ u(\mathbf{s}^+) = u(\mathbf{s}^-), \quad \epsilon_s \dfrac{\partial u(\mathbf{s}^+)}{\partial \mathbf{n}(\mathbf{s})} = \epsilon_p \dfrac{\partial u(\mathbf{s}^-)}{\partial \mathbf{n}(\mathbf{s})} & \text{for } |\mathbf{s}| = a, \\ u(\mathbf{r}) \to 0 & \text{as } |\mathbf{r}| \to \infty, \end{cases} \tag{6.2}$$

where $\rho(\mathbf{r}) = \alpha \sum_{j=1}^{n_p} z_j \delta(\mathbf{r} - \mathbf{r}_j)$. Because of "excess" charge term added to the solvent region D_s, the two parameters α and κ lost their physical meaning. Hence, they can be selected as the parameters for controlling the solution range.

The solution u of the PBE test model is presented in the following theorem.

Theorem 6.1. *The analytical solution u of the Poisson test model (6.2) is given by*

$$u(\mathbf{r}) = \alpha \sum_{j=1}^{n_p} z_j u_j(\mathbf{r}), \tag{6.3}$$

where u_j has the analytical expression

$$u_j(\mathbf{r}) = \begin{cases} \dfrac{1}{4\pi\epsilon_p|\mathbf{r}|} + \dfrac{1}{4\pi a}\left(\dfrac{1}{\epsilon_s} - \dfrac{1}{\epsilon_p}\right), & \mathbf{r} \in D_p, \\ \dfrac{1}{4\pi\epsilon_s|\mathbf{r}|}, & \mathbf{r} \in D_s \end{cases} \quad \text{if } |\mathbf{r}_j| = 0 \text{ (i.e., a charge at the origin),} \tag{6.4}$$

and the series expression

$$u_j(\mathbf{r}) = \begin{cases} \dfrac{1}{4\pi\epsilon_p|\mathbf{r} - \mathbf{r}_j|} + \displaystyle\sum_{n=0}^{\infty} A_{j,n}|\mathbf{r}|^n P_n\left(\dfrac{\mathbf{r}_j \cdot \mathbf{r}}{|\mathbf{r}_j||\mathbf{r}|}\right), & \mathbf{r} \in D_p, \\ \displaystyle\sum_{n=0}^{\infty} B_{j,n} \dfrac{1}{|\mathbf{r}|^{n+1}} P_n\left(\dfrac{\mathbf{r}_j \cdot \mathbf{r}}{|\mathbf{r}_j||\mathbf{r}|}\right), & \mathbf{r} \in D_s \end{cases} \quad \text{if } |\mathbf{r}_j| \neq 0. \tag{6.5}$$

Here, P_n denotes the Legendre polynomial of degree n, and $A_{j,n}$ and $B_{j,n}$ are defined by

$$A_{j,n} = \frac{(\epsilon_p - \epsilon_s)(n+1)|\mathbf{r}_j|^n}{4\pi\epsilon_p a^{2n+1}[n\epsilon_p + (n+1)\epsilon_s]}, \qquad B_{j,n} = \frac{(2n+1)|\mathbf{r}_j|^n}{4\pi[n\epsilon_p + (n+1)\epsilon_s]}. \tag{6.6}$$

Proof. By the superposition principle, we can express the solution u of (6.2) in the form of (6.3) with u_j being the solution of the test model (6.2) using $\rho(\mathbf{r}) = \delta(\mathbf{r} - \mathbf{r}_j)$. Under the spherical coordinate system, $\mathbf{r}_j$ is expressed as

$$\mathbf{r}_j = (|\mathbf{r}_j| \sin\phi_j \cos\theta_j, |\mathbf{r}_j| \sin\phi_j \cos\theta_j, |\mathbf{r}_j| \cos\phi_j) \quad \text{for } \theta_j \in [0, 2\pi] \text{ and } \phi_j \in [0, \pi].$$

Let $\tilde{\mathbf{r}}_j = (0, 0, |\mathbf{r}_j|)$ and $\tilde{u}_j$ denote the solution of (6.2) using $\rho(\mathbf{r}) = \delta(\mathbf{r} - \tilde{\mathbf{r}}_j)$. We can find u_j by the formula

$$u_j(\mathbf{r}) = \tilde{u}_j(\mathcal{O}_j \mathbf{r}),$$

where $\mathcal{O}_j$ is a rotation operator that maps $\mathbf{r}_j$ to $\tilde{\mathbf{r}}_j$. Hence, the problem becomes to find $\tilde{u}_j(\mathbf{r}')$ with $\mathbf{r}' = \mathcal{O}_j \mathbf{r}$.

Since $\tilde{u}_j(\mathbf{r}')$ is rotationally symmetric, we can follow the procedure suggested in [44, Section 4.4] to find $\tilde{u}_j$ in the series expressions

$$\tilde{u}_j(\mathbf{r}') = \begin{cases} \dfrac{1}{4\pi\epsilon_p|\mathbf{r}' - \tilde{\mathbf{r}}_j|} + \displaystyle\sum_{n=0}^{\infty} A_{j,n}|\mathbf{r}'|^n P_n(\cos\langle\tilde{\mathbf{r}}_j, \mathbf{r}'\rangle), & \mathbf{r}' \in D_p, \\ \displaystyle\sum_{n=0}^{\infty} B_{j,n}|\mathbf{r}'|^{-n-1} P_n(\cos\langle\tilde{\mathbf{r}}_j, \mathbf{r}'\rangle), & \mathbf{r}' \in D_s, \end{cases} \tag{6.7}$$

where $\langle\tilde{\mathbf{r}}_j, \mathbf{r}'\rangle$ denotes the angle between $\tilde{\mathbf{r}}_j$ and $\mathbf{r}'$. Clearly, $\cos\langle\tilde{\mathbf{r}}_j, \mathbf{r}'\rangle$ can be calculated by

$$\cos\langle\tilde{\mathbf{r}}_j, \mathbf{r}'\rangle = \frac{\tilde{\mathbf{r}}_j \cdot \mathbf{r}'}{|\tilde{\mathbf{r}}_j||\mathbf{r}'|} \quad \text{for } j = 1, 2, \ldots, n_p.$$

Note that $|\mathbf{r}'| = |\mathbf{r}|$, $|\mathbf{r}' - \tilde{\mathbf{r}}_j| = |\mathbf{r} - \mathbf{r}_j|$, and $\langle\tilde{\mathbf{r}}_j, \mathbf{r}'\rangle = \langle\mathbf{r}_j, \mathbf{r}\rangle$. Hence, from (6.7) we can obtain u_j in the series expression (6.5). This completes the proof. □

Table 1
Validation test results on our new PBE hybrid solver for the PBE test model (6.1) with a unit ball D_p containing 892 point charges from a protein (4PTI). Here, Iter. denotes the number of iterations for the PBE hybrid solver satisfying the termination rule (3.7), and $\mathcal{E}_{h_j}$ is the relative error defined in (7.1).

Mesh size h	Number of mesh points		In the l_2 norm		In the l_∞ norm		Iter.
	On Ω	On Ω_7	$\mathcal{E}_{h_j}$	$\mathcal{E}_{h_j}/\mathcal{E}_{h_{j+1}}$	$\mathcal{E}_{h_j}$	$\mathcal{E}_{h_j}/\mathcal{E}_{h_{j+1}}$	
$h_1 = 0.25$	120 887	18 863	9.55×10^{-2}		1.79×10^{-2}		8
$h_2 = h_1/2\ (0.125)$	940 247	145 223	2.29×10^{-2}	4.170	5.38×10^{-3}	3.33	8
$h_3 = h_2/2\ (0.0625)$	7 412 989	1 136 605	5.63×10^{-3}	4.067	1.81×10^{-3}	2.97	8

7. Program package and numerical results

We programmed Algorithm 4.1 and the Newton-PCG-MG scheme in `Python` and `Fortran` based on the program packages we developed in [28,29]. In particular, we adopted the PBE finite element package SDPBS from [29] to solve each nonlinear interface problem on the central box Ω_7, and reused the PCG-MG program, the programs for generating the seven-boxes partition and the interface-fitted unstructured mesh of Ω_7, and the program for calculating G and ∇G from [28]. We also programmed the analytical solution of the Poisson test model (6.2) in `Fortran`. The `Fortran` programs were converted to `Python` modules via the Fortran-to-Python interface generator `f2py` to directly use in Python programs. The usage of our new PBE program package is similar to that of SDPB, see [29] for details.

For simplicity, we did all the numerical tests with $\epsilon_p = 2, \epsilon_s = 80, T = 298.15, I_s = 0.1$, which gave

$$\alpha \approx 7042.94001, \qquad \kappa^2 \approx 0.84827,$$

according to the expressions of (4.2), and the default values of other parameters. The tests were done on one processor of our Mac Pro Workstation with the 3.7 GHz Quad-Core Intel Xeon E5 and 64 GB memory.

7.1. Validation tests

To validate our new PBE hybrid solver, we made numerical experiments on the PBE test model (6.1) using $\Omega = (-6, 6)^3$, $a = 1, D = (-2, 2)^3$, $\Omega_7 = (-3, 3)^3$, $\alpha = 1$, and 892 charges coming from a protein with the PDB ID 4PTI. The atomic positions $\mathbf{r}_j$ were rescaled to the unit ball D_p such that $|\mathbf{r}_j| \le 0.8$ for $j = 1, 2, \ldots, 892$. The relaxation parameter ω was set as 1.275 for computing Ψ and 1.225 for computing $\tilde{\Phi}$. We constructed three nested uniform meshes with mesh sizes $h = 0.25$, 0.125, and 0.0625 for each neighboring box, and three nested finite element meshes of the central box Ω_7, which have 18 863, 145 223, and 1 136 605 mesh points, respectively. As illustrated in Fig. 3, each mesh of Ω_7 consists of an unstructured mesh of D to fit the interface, and a uniform mesh of $\Omega_7 \setminus D$ (the overlapped part of Ω_7 with its six neighboring boxes) with the same mesh size as the one of a neighboring box finite difference mesh to simplify the data exchanges between Ω_7 and its neighboring boxes. By a "regular" subdivision (i.e., edge midpoints are connected by new edges), we constructed the three nested tetrahedral meshes of Ω_7 with the largest diameters of tetrahedra being 0.7974, 0.4883, and 0.2809, respectively.

In the tests, we calculated the series solution using its partial sum S_N, which is a sum of the first N terms of the series solution given in (6.5). We used $N = 20$ since it was found to be large enough for these validation tests as indicated in Fig. 4. Here, as an example, we displayed the relative errors of S_N on the mesh with $h = 0.25$ from $N = 5, 6, \ldots, 20$. The series solution vector U was calculated by using $N = 100$, which was accurate enough for these tests according to our numerical experiments. From this figure it can also been seen that our simple series expression had a geometric rate of convergence.

We calculated the relative error, $\mathcal{E}_h$, by the formula

$$\mathcal{E}_h = \frac{\|U - U_h\|}{\|U\|}, \tag{7.1}$$

where U and U_h denote the two vectors of series solution and numerical solution values of the PBE test model (6.1) at interior mesh points, respectively. In these validation tests, we calculated the relative error using both l_2 and l_∞ vector norms. The results were reported in Table 1.

From Table 1 it can be seen that the relative error in the l_2 vector norm was reduced to one fourth when the mesh size h was decreased by half, which showed that our new PBE box iterative method had a second order of convergence rate in terms of h. In the case of l_∞ norm, the numerical convergence order was about 1.6. These error orders matched well with the finite element theory [43, Page 217]. The total number of iterations was eight for these three nested meshes, implying that the new box iterative method had a fast rate of convergence independent of mesh size h. These numerical tests well validated our new PBE hybrid solver and its program package.

7.2. Comparison tests with other solvers

To simplify the comparison of our PBE hybrid scheme with a PBE finite difference scheme proposed in [35], we considered the Poisson test model, which is defined by (6.1) with $\kappa = 0$. We programmed this finite difference scheme in `Fortran`

Table 2
A comparison of our new hybrid solver using an interface-fitted tetrahedral mesh with the finite difference and finite element solvers using a uniform Cartesian mesh Ω_h in the case of solving the Poisson test model: (6.1) with $\kappa = 0$, $\Omega = (-2, 2)^3$, $D_p = \{\mathbf{r} \mid |\mathbf{r}| < 1\}$, $\alpha = 7042.94$, $\epsilon_p = 2$, $\epsilon_s = 80$, and point charges from three proteins (1D3X, 1AZQ, 1TC3). Here, $\mathcal{E}_h$ is defined in (7.1) in the case of l_2 norm, and $\mathcal{E}_h^{solv}$ is defined in (7.4). We also calculated $\mathcal{E}_h$ by (7.2) for the two finite element solvers.

Finite difference solver on Ω_h				Finite element solver on Ω_h			Our hybrid solver				
h	N_h	$\mathcal{E}_h$ in l_2 norm	$\mathcal{E}_h^{solv}$	$\mathcal{E}_h$		$\mathcal{E}_h^{solv}$	$h_{\max}$	N_h	$\mathcal{E}_h$		$\mathcal{E}_h^{solv}$
				l_2 norm	l_∞ norm				l_2 norm	l_∞ norm	
Protein (1D3X): 756 atoms and analytical solvation free energy $\triangle E = -36115.4$ kcal/mol											
0.16	15 625	0.327	0.012	0.251	0.291	0.165	1.11	5 577	0.1093	0.0307	0.0184
0.083	117 649	0.302	0.008	0.141	0.177	0.083	0.63	44 948	0.0263	0.0091	0.0047
0.042	912 673	0.174	0.004	0.070	0.097	0.038	0.34	358 719	0.0069	0.0028	0.0012
Protein (1AZQ): 1603 atoms and analytical solvation free energy $\triangle E = -7090.24$ kcal/mol											
0.16	15 625	0.219	0.0216	0.195	0.317	0.243	1.11	5 577	0.0966	0.0495	0.0264
0.083	117 649	0.286	0.0126	0.102	0.173	0.116	0.63	44 948	0.0230	0.0154	0.0063
0.042	912 673	0.239	0.0067	0.049	0.089	0.053	0.34	358 719	0.0058	0.0046	0.0018
Protein (1TC3): 2124 atoms and analytical solvation free energy $\triangle E = -101\,769$ kcal/mol											
0.16	15 625	0.142	0.0087	0.226	0.325	0.171	1.11	5 577	0.0991	0.0345	0.0190
0.083	117 649	0.086	0.0085	0.120	0.161	0.083	0.63	44 948	0.0238	0.0099	0.0049
0.042	912 673	0.103	0.0046	0.058	0.078	0.038	0.34	358 719	0.0060	0.0027	0.0013

based on a uniform Cartesian grid mesh. That is, the Poisson test model was treated as a second-order elliptic boundary value problem with the jump coefficient functions, and then discretized by using the seven-point finite difference stencil without considering any interface condition. Furthermore, each Dirac-delta functional $\delta(\mathbf{r} - \mathbf{r}_j)$ was approximated as a piecewise linear interpolation function, $\delta_h(\mathbf{r})$, [36,45]. We solved each finite difference linear system by our PCG-MG scheme. We also implemented our SDPBS finite element solver based on a uniform tetrahedral mesh. This program was intended to mimic a PBE finite difference scheme whose construction involved the interface conditions and solution decomposition techniques.

We made numerical experiments on these three solvers for the Poisson test model using $\Omega = (-2, 2)^3$, $a = 1$, $\alpha = 7042.94$, and the point charges from three proteins (with PDB ID 1D3X, 1AZQ, and 1TC3), which have 756, 1603, and 2124 atoms, respectively. The relative error $\mathcal{E}_h$ were calculated by formula (7.1) in the l_2 norm for the three solvers. We further calculated the relative errors for the two finite element solvers using the formula:

$$\mathcal{E}_h = \frac{\|\Psi - \Psi_h\|_\infty}{\|\Psi\|_\infty} \quad \text{with } \Psi = U - G, \tag{7.2}$$

since G is given in (4.4) analytically, and each numerical solution of the Poisson model only involves the calculation of Ψ. In (7.2), Ψ and Ψ_h are the two vectors of series solution and numerical solution values of (4.5) at the interior mesh points of a mesh of domain Ω, respectively.

One important application of PBE is to predict the electrostatic solvation free energy, $\triangle E$, of a protein in a solvent. According to the solution decomposition (4.3), we can estimate $\triangle E$ by the formula

$$\triangle E = \frac{N_A}{4184} \cdot \frac{k_B T}{2} \sum_{j=1}^{n_p} z_j \left[\Psi(\mathbf{r}_j) + \tilde{\Phi}(\mathbf{r}_j)\right] \quad \text{in kilocalorie per mole (kcal/mol)}. \tag{7.3}$$

For the Poisson test model, we have $\tilde{\Phi} = 0$, and can find the analytical value of $\triangle E$ from the series solution (6.3). Hence, the relative error, $\mathcal{E}_h^{solv}$, of a numerical solvation free energy value, $\triangle E_h$, can be calculated by the formula

$$\mathcal{E}_h^{solv} = \frac{|\triangle E_h - \triangle E|}{|\triangle E|}. \tag{7.4}$$

In these tests, an initial guess of zero was used, the iteration was terminated when the relative residue norm was less than 10^{-5}, the series solutions and solvation free energies were calculated by using the first 20 terms, each finite difference system was solved by our PCG-MG method, and each finite element system was solved by the PCG-ILU from scientific library PETSc. The numerical results were reported in Table 2. Here, Ω_h denotes a uniform mesh of Ω, N_h is the total number of mesh points, and $h_{\max}$ denotes the maximum of all the diameters of tetrahedra for an unstructured tetrahedral mesh of Ω.

From Table 2 we can see that the finite difference solver had a much lower accuracy than our hybrid solver in the calculation of numerical solutions. Even on a small mesh with only 5577 mesh points, our finite element solver produced more accurate numerical solutions than the finite difference method on a large mesh of 912,673 mesh points. This implies that our hybrid solver can have better performance than the finite difference method in both CPU time and computer memory if the same numerical accuracy is required for the both solvers. Gladly, from Table 2 we also see that the finite difference solver had a satisfactory accuracy in the calculation of electrostatic solvation free energy. Hence, it can be valuable for the application problems that involve solvation free energies due to its simplicity and efficiency in implementation.

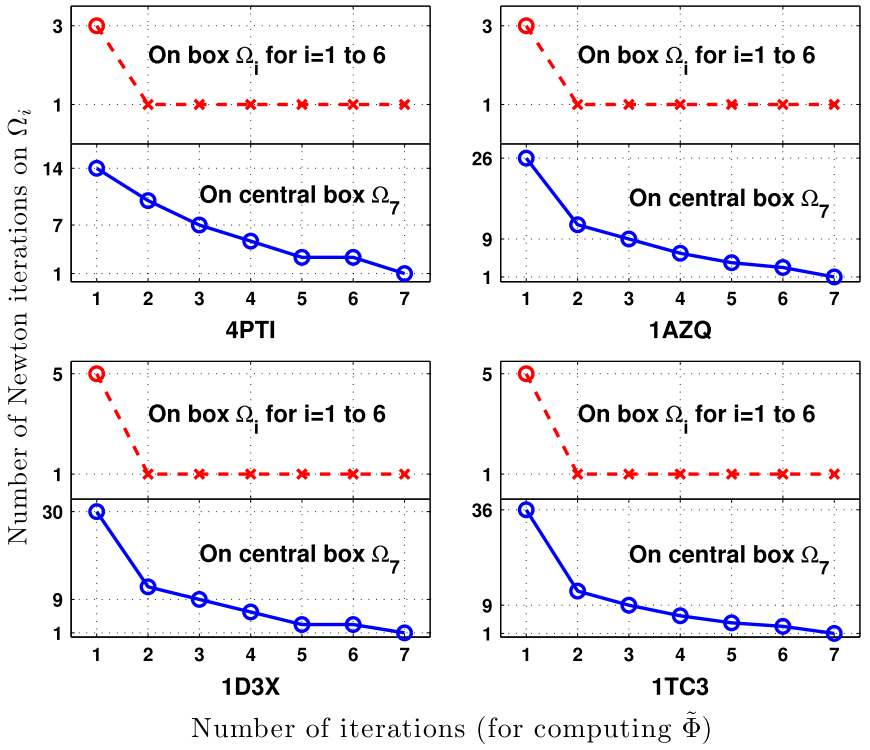


Fig. 5. Numbers of Newton-PCG-MG (dash line) and Newton-PCG-ILU (solid line) iterations for each iteration of the new box iterative method in computing $\tilde{\Phi}$. Here cross indicates that the nonlinear problem has been substituted to the linear problem (4.7).

Table 3
Some protein and mesh data used in the numerical tests. Here, n_p is the number of atoms, N_{Ω_7} and N_Ω denote the numbers of mesh points on Ω_7 and Ω, respectively, and $\hat{\rho}$ is the percentage defined by $\hat{\rho} = 100N_{\Omega_7}/N_\Omega\%$.

PDB ID	n_p	Cube $D = (a_1, b_1; a_2, b_2; a_3, b_3)$	N_{Ω_7}	N_Ω	$\hat{\rho}$
1CBN	642	(−28.3, 46.9; −27.9, 47.2; −30.4, 44.7)	51 133	546 637	9.4%
1SVR	1433	(−50.9, 53.3; −56.5, 47.7; −50.6, 53.6)	78 149	573 653	13.6%
4PTI	892	(−30.9, 61.5; −25.4, 67.0; −41.7, 50.7)	62 199	557 703	11.2%
1AZQ	1603	(−51.5, 70.9; −50.2, 72.2; −49.8, 72.6)	91 864	587 368	15.6%
1D3X	756	(−42.0, 41.5; −42.5, 41.1; −41.3, 42.3)	59 297	554 801	10.7%
1TC3	2124	(−71.0, 91.6; 50.8, 213.4; −48.8, 113.8)	79 320	574 824	13.8%

7.3. Performance tests for proteins

We made numerical tests on six proteins to compare the performance of our new PBE hybrid solver with that of SDPB. Here D and Ω were constructed from Step 1 of Algorithm 4.1 by using the default values. For clarity, we listed the dimensions of D and some basic information of these proteins and meshes in Table 3. In these tests, the relaxation parameter ω was set as 1.215 for computing Ψ and 1.015 for computing $\tilde{\Phi}$. Numerical results were reported in Figs. 5, 6, and 7 and Table 4.

Fig. 5 reports the numbers of iterations within each box in solving the nonlinear interface problem on Ω_7 by Newton-PCG-ILU and the nonlinear boundary value problem on each neighboring box by Newton-PCG-MG. Here the nonlinear boundary value problems had been substituted to its linearized problem after the first iteration in all the six neighboring boxes. From this figure it can also be seen that the iteration number of Newton-PCG-ILU was reduced significantly after the first iteration, indicating that the properties of the nonlinear interface problem of Ω_7 was improved as its boundary value function became a better approximation to the solution $\tilde{\Phi}$. In other words, the iterates from the neighboring boxes mainly played a role to generate a "good" boundary value function for the nonlinear interface problem on Ω_7.

Fig. 6 displays the convergence of our new box iterative method (3.2) for solving the nonlinear interface problem (4.6) for $\tilde{\Phi}$. Here, an initial iterate of zero was used, and the convergence was controlled by the test rule (3.7) with $\epsilon = 10^{-7}$. From the figure we can see that the errors were reduced by more than tenth per iteration, showing that the new box iterative method had a fast rate of convergence.

Fig. 7 shows that our Newton-PCG-MG method had a quadratic rate of convergence. These test results came from box Ω_1 for four proteins represented in the PDB IDs 1D3X and 1TC3. Here, an initial iterate of zero was used, and the convergence was controlled by the test rule (5.9) with $\epsilon = 10^{-7}$.

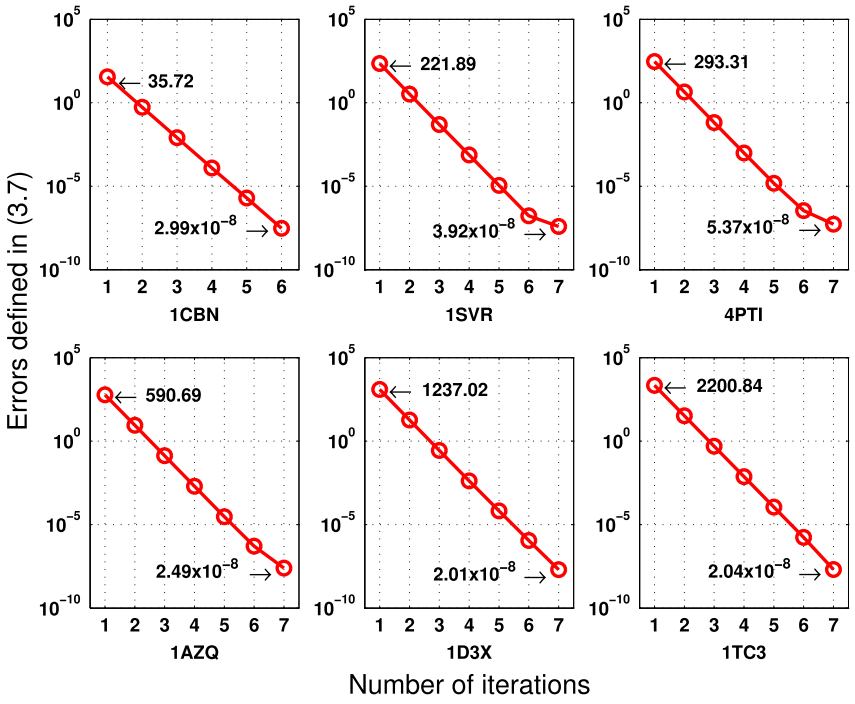


Fig. 6. Convergence of the new box iterative method (3.2) for solving the nonlinear interface problem (4.6) for $\tilde{\Phi}$ with the iteration termination rule (3.7). Here 1CBN, 1SVR, 4PTI, 1AZQ, 1D3X, and 1TC3 are the PDB IDs of the six proteins.

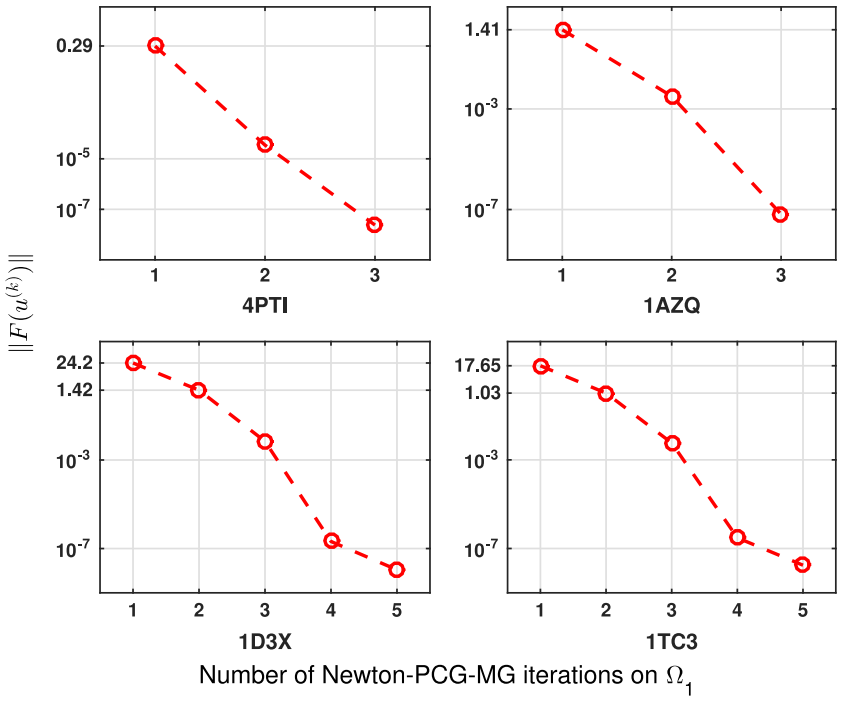


Fig. 7. Convergence of our Newton-PCG-MG method for solving a nonlinear boundary value problem on box Ω_1 with the iteration termination rule (5.9). Here, F is defined in (5.8), and $\|F(u^{(k)})\|$ is the Euclidean vector norm at the kth iterate $u^{(k)}$ on Ω_1.

Table 4 compares the performance of the new PBE hybrid solver with that of the PBE finite element software SDPBS in the calculation of PBE component functions G, Ψ and $\tilde{\Phi}$ as well as in the total CPU time. Here, the total time excluded mesh generation time, and the total time speedup S_p is defined as a ratio of the total time costed by SDPBS to the one by our new PBE hybrid solver. In these tests, the relative errors between the numerical solutions by the new PBE hybrid solver and

Table 4
A comparison of the performance of our new PBE hybrid solver (Hybrid) with that of the PBE finite element solver SDPBS from [29] in CPU time (in seconds).

PDB ID	Find G & ∇G		Find Ψ		Find $\tilde{\Phi}$		Total time		Time speed up S_p
	Hybrid	SDPBS	Hybrid	SDPBS	Hybrid	SDPBS	Hybrid	SDPBS	
1CBN	2.72	5.28	3.90	13.71	23.34	74.67	30.53	94.54	3.10
1SVR	7.59	15.01	7.76	15.27	56.98	123.66	73.39	154.81	2.11
4PTI	4.48	9.05	5.71	15.54	36.01	85.33	46.96	110.77	2.36
1AZQ	8.88	17.28	9.09	17.27	81.27	175.14	100.43	210.60	2.10
1D3X	3.76	7.65	5.51	15.31	48.79	170.23	58.73	194.05	3.30
1TC3	11.58	22.34	8.21	16.6	77.03	236.09	98.11	275.92	2.81

SDPBS were found to be less than 10^{-7}. From Table 4 we can see that our new PBE hybrid solver speeded up the total CPU time of SDPBS up to 3 times. We also noted that a smaller value of the ratio $\hat{\rho}$ might result in a larger speedup S_p due to the fact that the Newton-PCG-MG method on the six neighboring boxes is more efficient than the Newton-PCG-ILU method used in the central box Ω_7. Hence, the performance can be further improved through a proper reduction of the ratio $\hat{\rho}$.

Acknowledgment

This work was partially supported by the National Science Foundation, USA, through grant DMS-1226259.

References

[1] H.A. Schwarz, Ueber einige abbildungsaufgaben, J. Reine Angew. Math. 70 (1869) 105–120.
[2] P.-L. Lions, On the Schwarz alternating method. I, in: First International Symposium on Domain Decomposition Methods for Partial Differential Equations, Paris, France, 1988, pp. 1–42.
[3] T.P.A. Mathew, Domain Decomposition Methods for the Numerical Solution of Partial Differential Equations, in: Lecture Notes in Computational Science and Engineering, vol. 61, Springer Verlag, 2008.
[4] A. Quarteroni, A. Valli, Domain Decomposition Methods for Partial Differential Equations, Oxford University Press, 1999.
[5] A. Toselli, O. Widlund, Domain Decomposition Methods: Algorithms and Theory, Vol. 3, Springer, 2005.
[6] 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.
[7] B. Honig, A. Nicholls, Classical electrostatics in biology and chemistry, Science 268 (1995) 1144–1149.
[8] 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.
[9] 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.
[10] P. Ren, J. Chun, D.G. Thomas, M.J. Schnieders, M. Marucho, J. Zhang, N.A. Baker, Biomolecular electrostatics and solvation: A computational perspective, Q. Rev. Biophys. 45 (04) (2012) 427–491.
[11] L. Xiao, C. Wang, R. Luo, Recent progress in adapting Poisson–Boltzmann methods to molecular simulations, J. Theoret. Comput. Chem. 13 (03) (2014) 1430001.
[12] I. Babuška, The finite element method for elliptic equations with discontinuous coefficients, Computing 5 (1970) 207–213.
[13] 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.
[14] Z. Chen, J. Zou, Finite element methods and their convergence for elliptic and parabolic interface problems, Numer. Math. 79 (1998) 175–202.
[15] B.P. Lamichhane, B.I. Wohlmuth, Mortar finite elements for interface problems, Computing 72 (2004) 333–348.
[16] D. Xie, S. Zhou, A new minimization protocol for solving nonlinear Poisson–Boltzmann mortar finite element equation, BIT 47 (2007) 853–871.
[17] A. Brandt, Multi-level adaptive solutions to boundary value problems, Math. Comp. 31 (138) (1977) 333–390.
[18] M.A. Olshanskii, E.E. Tyrtyshnikov, Iterative Methods for Linear Systems: Theory and Applications, SIAM, 2014.
[19] P. Deuflhard, Newton Methods for Nonlinear Problems: Affine Invariance and Adaptive Algorithms, Vol. 35, Springer Science & Business Media, 2011.
[20] D. Knoll, W. Rider, A multigrid preconditioned Newton–Krylov method, SIAM J. Sci. Comput. 21 (2) (1999) 691–710.
[21] L. Mu, J. Wang, G. Wei, X. Ye, S. Zhao, Weak Galerkin methods for second order elliptic interface problems, J. Comput. Phys. 250 (2013) 106–125.
[22] Z. Li, K. Ito, The Immersed Interface Method: Numerical Solutions of PDEs Involving Interfaces and Irregular Domains, SIAM, Philadelphia, 2006.
[23] C. Peskin, The immersed boundary method, Acta Numer. 11 (1) (2002) 479–517.
[24] K. Xia, G.-W. Wei, A Galerkin formulation of the MIB method for three dimensional elliptic interface problems, Comput. Math. Appl. 68 (7) (2014) 719–745.
[25] K. Xia, M. Zhan, G.-W. Wei, MIB Galerkin method for elliptic interface problems, J. Comput. Appl. Math. 272 (2014) 195–220.
[26] J.L. Hellrung, L. Wang, E. Sifakis, J.M. Teran, A second order virtual node method for elliptic problems with interfaces and irregular domains in three dimensions, J. Comput. Phys. 231 (4) (2012) 2015–2048.
[27] T. Lin, Y. Lin, X. Zhang, Partially penalized immersed finite element methods for elliptic interface problems, SIAM J. Numer. Anal. 53 (2) (2015) 1121–1144.
[28] J. Ying, D. Xie, A new finite element and finite difference hybrid method for computing electrostatics of ionic solvated biomolecule, J. Comput. Phys. 298 (2015) 636–651.
[29] D. Xie, New solution decomposition and minimization schemes for Poisson–Boltzmann equation in calculation of biomolecular electrostatics, J. Comput. Phys. 275 (2014) 294–309.
[30] M.Z. Born, Volumen und hydratationswärme der ionen, Z. Phys. 1 (1) (1920) 45–48.
[31] M. Holst, J.A. McCammon, Z. Yu, Y. Zhou, Y. Zhu, Adaptive finite element modeling techniques for the Poisson–Boltzmann equation, Commun. Comput. Phys. 11 (1) (2012) 179–214.
[32] J.G. Kirkwood, Theory of solutions of molecules containing widely separated charges with special application to zwitterions, J. Chem. Phys. 2 (1934) 351.
[33] W. Geng, S. Yu, G. Wei, Treatment of charge singularities in implicit solvent models, J. Chem. Phys. 127 (11) (2007) 114106.
[34] S. Balay, J. Brown, K. Buschelman, W. Gropp, D. Kaushik, M. Knepley, L. McInnes, B. Smith, H. Zhang, PETSc Web page, 2012. http://www.mcs.anl.gov/petsc.
[35] A. Nicholls, B. Honig, A rapid finite difference algorithm, utilizing successive over-relaxation to solve the Poisson–Boltzmann equation, J. Comput. Chem. 12 (1991) 435–445.

[36] I. Klapper, R. Hagstrom, R. Fine, K. Sharp, B. Honig, Focusing of electric fields in the active site of Cu-Zn superoxide dismutase: Effects of ionic strength and amino-acid modification, Proteins: Struct. Funct. Bioinform. 1 (1) (1986) 47–59.
[37] 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.
[38] 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.
[39] W. Geng, S. Yu, G. Wei, Treatment of charge singularities in implicit solvent models, J. Chem. Phys. 127 (11) (2007) 114106.
[40] C. Wang, J. Wang, Q. Cai, Z. Li, H.-K. Zhao, R. Luo, Exploring accurate Poisson–Boltzmann methods for biomolecular simulations, Comput. Theoret. Chem. 1024 (2013) 34–44.
[41] J. Xu, Iterative methods by space decomposition and subspace correction, SIAM Rev. 34 (4) (1992) 581–613.
[42] J. Xu, J. Zou, Some nonoverlapping domain decomposition methods, SIAM Rev. 40 (4) (1998) 857–914.
[43] S.C. Brenner, L.R. Scott, The Mathematical Theory of Finite Element Methods, third ed., Springer-Verlag, 2008.
[44] J.D. Jackson, Classical Electrodynamics, third ed., Wiley, New York, 1999.
[45] Y. Jiang, J. Ying, D. Xie, A Poisson–Boltzmann equation test model for protein in spherical solute region and its applications, Mol. Based Math. Biol. 2 (2014) 86–97. open Access.