



Research article

A Newton-like iterative method implemented in the DelPhi for solving the nonlinear Poisson-Boltzmann equation

Chuan Li^{1,*}, Mark McGowan², Emil Alexov³ and Shan Zhao²

¹ Department of Mathematics, West Chester University of Pennsylvania, West Chester, Pennsylvania 19383, USA

² Department of Mathematics, University of Alabama, Tuscaloosa, AL 35487, USA

³ Department of Physics and Astronomy, Clemson University, Clemson, South Carolina 29634, USA

* **Correspondence:** Email: cli@wcupa.edu; Tel: +16104361081; Fax: +16107380578.

Abstract: DelPhi is a popular scientific program which numerically solves the Poisson-Boltzmann equation (PBE) for electrostatic potentials and energies of biomolecules immersed in water via finite difference method. It is well known for its accuracy, reliability, flexibility, and efficiency. In this work, a new edition of DelPhi that uses a novel Newton-like method to solve the nonlinear PBE, in addition to the already implemented Successive Over Relaxation (SOR) algorithm, is introduced. Our tests on various examples have shown that this new method is superior to the SOR method in terms of stability when solving the nonlinear PBE, being able to converge even for problems involving very strong nonlinearity.

Keywords: DelPhi; Poisson-Boltzmann equation; electrostatics; Newton method; finite difference technique

1. Introduction

Electrostatic interaction is a major factor which is commonly taken into account when studying numerous biological phenomena [1,2], such as macromolecular binding and recognition [3–6], pH-dependent folding and binding [7–11], nonspecific ion binding [12–14], pKa calculations [15–17], and salt-dependent effects [18,19], etc.. Existing models of calculating electrostatic potentials and corresponding energies developed in the past couple of decades can be roughly classified into two

categories. Explicit solvent models treat mobile water and ions explicitly and thus capture all molecular details but are computationally costly in terms of CPU time and memory usage. Implicit solvent models, such as the Generalized Born [20] and Poisson-Boltzmann (PB) models [21–26], treat surrounding water as a continuum media, and can be solved with relatively low computational costs. Because of that, implicit solvent models are usually preferred when modeling electrostatics of macromolecules at genome-scale applications.

Among all existing implicit solvent models, the Poisson-Boltzmann equation (PBE), is one of the most popular models utilized by many researchers. A lot of efforts have been devoted to developing scientific software to solve the PBE. For instance, DelPhi [27,28] utilizes the finite difference and Successive Over Relaxation (SOR) methods to iteratively solve the PBE until a prescribed tolerance is satisfied, PBSA [29] adopts the Finite Volume/Periodic Conjugate Gradient (FV/PCG) and the Immersed Interface/Fast Fourier Transform (IIM/FFT) methods to solve the PBE, MIBPB [30] develops a unique matched interface and boundary (MIB) method to explicitly enforce the jump conditions on the interfaces (molecular surfaces) in the finite difference formulations, resulting in a method capable of capturing sharp jumps of the potentials at the molecular surfaces, APBS [31] is an adaptive PBE solver which solves the PBE by a specifically designed finite element method, and many others [32,33].

As one of the most popular PBE solvers, DelPhi has been continuously maintained and developed for improved performance. Many new features were added in DelPhi in recent years [34]. This work reports a newly developed Newton-like method which was introduced into DelPhi recently. This new method has been tested extensively, including some purposely created “crashing” cases with strong nonlinearity. In particular, this method has been shown to be incredibly stable and is capable of delivering reliable numerical results in all tested cases, making this newly developed method a valuable add-on to DelPhi for solving problems with strong nonlinearity.

The rest of this work is organized as follows. The PBE and the finite difference methods are presented in section 2. Benchmarks of selected examples are shown section 3 to numerically compare the two methods implemented in DelPhi, followed by Conclusions and Acknowledgements in sections 4 and 5, respectively.

2. Methods

2.1. The Poisson-Boltzmann equation (PBE)

The PBE [35] is an elliptic-type Partial Differential Equation (PDE) given by

$$\nabla \cdot (\varepsilon(x) \nabla \phi(x)) - \kappa(x)^2 \sinh(\phi(x)) = -4\pi\rho(x), \quad (1)$$

where $\phi(x)$ is the electrostatic potential, $\varepsilon(x)$ is a spatial dielectric function, $\kappa(x)$ is a modified Debye-Huckel parameter, and $\rho(x)$ is the charge distribution function. Equation (1) is usually referred to as the Nonlinear Poisson-Boltzmann Equation (NLPBE) due to the presence of the hyperbolic sine function, $\sinh(\phi(x))$, in Eq (1). If the potential $\phi(x)$ is known to be small, Eq (1) can be linearized by an approximation, $\sinh(\phi(x)) \approx \phi(x)$, yielding a simplified model

$$\nabla \cdot (\varepsilon(x) \nabla \phi(x)) - \kappa(x)^2 \phi(x) = -4\pi\rho(x), \quad (2)$$

commonly referred to as the Linearized Poisson-Boltzmann Equation (LPBE). It is known that exact solutions to Eqs (1) and (2) only exist for a few simplified cases [27]. In practice, they must be solved numerically via certain numerical treatments for real bio-objects due to their irregular shapes. The numerical approaches for handling the nonlinearity of the PBE can be classified into two categories. In the most commonly used approach, the PBE is discretized by using finite difference or finite element methods, resulting a nonlinear algebraic system. Then a nonlinear algebraic method, such as nonlinear relaxation method [36,37], nonlinear conjugate gradient method [38] or inexact Newton method [39], can be employed to solve the nonlinear system efficiently. A comprehensive assessment of various algebra-based nonlinear PBE solvers can be found in [40]. A pseudo-time approach has also been developed [41–43], in which a time-dependent PBE is introduced by adding a pseudo-time derivative, and the PBE solution is recovered by a steady-state integration. The pseudo-time approach is usually less efficient than the nonlinear algebraic approach, because a long-time integration is needed for the steady state. But the pseudo-time approach could be more stable, especially when an analytical treatment to the nonlinear term is applied [43]. The method proposed in this work belongs to the first category. In the following subsections, implemented numerical methods in the DelPhi program will be introduced.

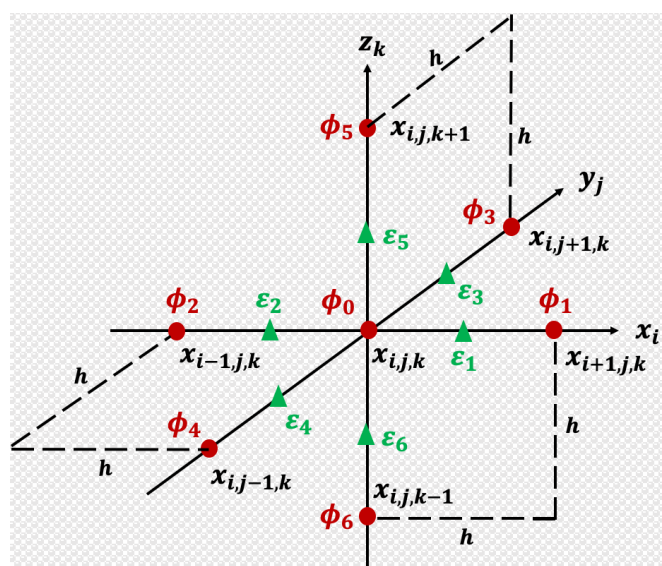


Figure 1. A graph demonstration of the numerical methods implemented in the DelPhi solver.

2.2. The successive over relaxation (SOR) method

DelPhi solves Eqs (1) and (2) in a cubic domain Ω containing the interested molecule. Boundary conditions are imposed on the six faces of Ω . Domain Ω is discretized by a uniform mesh size $h = \Delta x = \Delta y = \Delta z$ in all x -, y -, and z - directions. Approximations to the exact solutions of Eqs (1) and (2) are to be found at all grids.

Following the standard finite difference formulation, Eq (1) is discretized, resulting in [27]

$$h \sum_{i=1}^6 \varepsilon_i \phi_i - h \sum_{i=1}^6 \varepsilon_i \phi_0 - \kappa^2 \sinh(\phi_0) h^3 + 4\pi q_0 = 0, \quad (3)$$

where ϕ_0 is the unknown potential at a grid $x_{i,j,k}$, q_0 is the charge assigned to the same grid $x_{i,j,k}$, $\phi_i, i = 1 \dots 6$ are unknown potentials at six closest adjacent grids, and $\varepsilon_i, i = 1 \dots 6$ are dielectric coefficients at six adjacent half grids. See Figure 1 for a demonstration. Equation (3) can be rewritten as an iteration updating formula

$$h \sum_{i=1}^6 \varepsilon_i \phi_i^n - h \sum_{i=1}^6 \varepsilon_i \phi_0^{n+1} - \kappa^2 \sinh(\phi_0^n) h^3 \frac{\phi_0^{n+1}}{\phi_0^n} + 4\pi q_0 = 0 \quad (4)$$

with the superscript $n = 0, 1, \dots$ indicating the number of iterations. Solving ϕ_0^{n+1} in terms of others in Eq (4) yields [35]

$$\phi_0^{n+1} = \left(\sum_{i=1}^6 \varepsilon_i \phi_i^n + \frac{4\pi q_0}{h} \right) / \left(\sum_{i=1}^6 \varepsilon_i + (\kappa h)^2 \frac{\sinh(\phi_0^n)}{\phi_0^n} \right) \quad (5)$$

to solve the NLPBE for the potential at grid $x_{i,j,k}$. In a similar fashion, one can obtain the formula [44]

$$\phi_0^{n+1} = \left(\sum_{i=1}^6 \varepsilon_i \phi_i^n + \frac{4\pi q_0}{h} \right) / \left(\sum_{i=1}^6 \varepsilon_i + (\kappa h)^2 \right) \quad (6)$$

to solve the LPBE for the potential at $x_{i,j,k}$. Provided a guessed value of ϕ_0^0 (usually called the initial value), the current (approximated) potential ϕ_i^n is evolved to the next (approximated) potential ϕ_0^{n+1} by either Eq (5) or Eq (6) in the n th step of an iteration process for $n = 1, 2, \dots$. This process is terminated until a prescribed criterion is satisfied.

In DelPhi, potentials ϕ_0^{n+1} and ϕ_0^n in Eqs (5) and (6) are used in the SOR method

$$\phi_0^{n+1} = \omega \phi_0^{n+1} + (1 - \omega) \phi_0^n \quad (7)$$

for improved efficiency or stability in the n th iteration as well. Here the relaxation parameter ω is selected to be $0 < \omega < 2$. When a value $0 < \omega < 1$ is used, the iteration process converges slower but more stably (under relaxation). When a value $1 < \omega < 2$ is used, the iteration process converges in a faster pace but could be less stable (over relaxation). DelPhi uses $\omega = 1$ as the default value, yielding a method commonly known as the Gauss-Seidel (GS) method. DelPhi users can either manually set the value of ω , or let the program automatically calculate the optimized values of ω , for either faster convergence rates or stronger stability.

Equation (5) provides a numerical formula to solve the NLPBE iteratively but its convergence rate is not fast enough to solve problems in three dimensions [44]. DelPhi utilizes a special technique to accelerate the convergent rate. To this end, Eq (1) is “linearized” and rewritten as

$$\nabla \cdot (\varepsilon(x) \nabla \phi(x)) - \kappa(x)^2 \phi(x) = -4\pi \rho(x) + \kappa(x)^2 (\sinh(\phi(x)) - \phi(x)), \quad (8)$$

where the nonlinear term, $\kappa(x)^2 (\sinh(\phi(x)) - \phi(x))$, acts as an “excess charge” added to the regular charge term on the right-hand side of Eq (8) [44]. When Eq (8) is discretized, a formula

$$\phi_0^{n+1} = \left(\sum_{i=1}^6 \varepsilon_i \phi_i^n + \frac{4\pi q_0}{h} - \frac{\chi(\kappa(x)^2 (\sinh(\phi_0^n) - \phi_0^n))}{h} \right) / (\sum_{i=1}^6 \varepsilon_i + (\kappa h)^2) \quad (9)$$

is derived in place of Eq (5) for solving the NLPBE. In Eq (9), the excess charge term is multiplied by a second relaxation (strength) parameter χ which is initially small, $\chi = 0.05$. This parameter is slowly increased as iteration moves forward until $\chi = 1$ is reached. Then “full” nonlinear iterations start with $\chi = 1$ along the way.

In DelPhi, Eqs (6) and (7) are coupled for solving the LPBE, and Eqs (7) and (9) are coupled for solving the NLPBE. The iteration process is terminated, for instance, when $|\phi_0^{n+1} - \phi_0^n| < \text{TOL}$ at all grids for a prescribed tolerance TOL. These methods, together with additional computational techniques, such as the “checkerboard” ordering, stripping, and contiguous memory mapping³⁵, have been proven to be able to effectively deliver accurate numerical solutions to the LPBE and NLPBE for many three-dimensional problems.

However, it is known that the aforementioned “excess charge” treatment is merely a computational technique which could lead to undesired divergences caused by potentials at grids in water passing certain threshold, the grid spacing, and other factors [44]. One such “bizarre” example in which the SOR method fails to converge is given in the next section. It calls for a new addition to DelPhi’s capabilities, namely a Newton-like method, primarily focusing on solving the NLPBE for problems with strong nonlinearity. This method is described in the next subsection.

2.3. A Newton-like (NWT) method

The NWT method was developed to improve the stability of the numerical procedure when solving the NLPBE for problems with strong nonlinearity. To this end, we reconsider the left-hand side of Eq (3) as a function of ϕ_0 and write

$$F(\phi_0) = h \sum_{i=1}^6 \varepsilon_i \phi_i - h \sum_{i=1}^6 \varepsilon_i \phi_0 - \kappa^2 \sinh(\phi_0) h^3 + 4\pi q_0. \quad (10)$$

In order to find the root(s) of the equation $F(\phi_0) = 0$ via the Newton’s algorithm, the derivate of $F(\phi_0)$ is calculated first

$$\frac{dF}{d\phi_0} = -h \sum_{i=1}^6 \varepsilon_i - h^3 \kappa^2 \cosh(\phi_0). \quad (11)$$

Then Eqs (10) and (11) are substituted in the Newton’s algorithm, yielding

$$\begin{aligned}
\phi_0^{n+1} &= \phi_0^n - \frac{F(\phi_0^n)}{\frac{dF}{d\phi_0}(\phi_0^n)} \\
&= \phi_0^n - \frac{h \sum_{i=1}^6 \varepsilon_i \phi_i^n - h \sum_{i=1}^6 \varepsilon_i \phi_0^n - \kappa^2 \sinh(\phi_0^n) h^3 + 4\pi q_0}{-h \sum_{i=1}^6 \varepsilon_i - h^3 \kappa^2 \cosh(\phi_0^n)} \\
&= \frac{\sum_{i=1}^6 \varepsilon_i \phi_i^n + \frac{4\pi q_0}{h} + (\kappa h)^2 (\phi_0^n \cosh(\phi_0^n) - \sinh(\phi_0^n))}{\sum_{i=1}^6 \varepsilon_i + (\kappa h)^2 \cosh(\phi_0^n)}.
\end{aligned} \tag{12}$$

Equation (12) can be treated as a new updating formula to evolve ϕ_0^n to ϕ_0^{n+1} . Moreover, one can see that there is no difficulty to couple Eqs (7) and (12) and embrace all techniques already implemented in DelPhi to solve the NPBE.

Following similar derivations, Eq (2) can be discretized as

$$h \sum_{i=1}^6 \varepsilon_i \phi_i - h \sum_{i=1}^6 \varepsilon_i \phi_0 - \kappa^2 h^3 \phi_0 + 4\pi q_0 = 0. \tag{13}$$

Defining

$$G(\phi_0) = h \sum_{i=1}^6 \varepsilon_i \phi_i - h \sum_{i=1}^6 \varepsilon_i \phi_0 - \kappa^2 h^3 \phi_0 + 4\pi q_0, \tag{14}$$

one can calculate $G'(\phi_0)$ as

$$\frac{dG}{d\phi_0} = -h \sum_{i=1}^6 \varepsilon_i - h^3 \kappa^2. \tag{15}$$

Substituting Eqs (14) and (15) in the Newton's algorithm yields

$$\begin{aligned}
\phi_0^{n+1} &= \phi_0^n - \frac{G(\phi_0^n)}{\frac{dG}{d\phi_0}(\phi_0^n)} \\
&= \phi_0^n - \frac{h \sum_{i=1}^6 \varepsilon_i \phi_i^n - h \sum_{i=1}^6 \varepsilon_i \phi_0^n - \kappa^2 h^3 \phi_0^n + 4\pi q_0}{-h \sum_{i=1}^6 \varepsilon_i - h^3 \kappa^2} \\
&= \frac{\sum_{i=1}^6 \varepsilon_i \phi_i^n + \frac{4\pi q_0}{h}}{\sum_{i=1}^6 \varepsilon_i + (\kappa h)^2}
\end{aligned} \tag{16}$$

for solving the LPBE.

Equation (16) is actually the same as Eq (6). That is, both SOR and NWT methods utilize the same numerical formula to solve the LPBE. Thus, it is expected that these two methods are equally accurate and efficient for solving the LPBE. An example is provided in the supplementary material to numerically verify that implementations of these two methods in DelPhi are indeed equally accurate and efficient. Therefore, we will concentrate on comparing their performance when different formulas are actually used to solve the NLPBE in the remaining of this work.

The novelty of the NWT method is two-fold. First, Eq (12) is derived by applying the Newton algorithm on discretized equations obtained from the original PBE, while other Newton-type PBE solvers in the literature, to our best knowledge, are obtained by applying the Newton's algorithm directly on the original PBE. Secondly, this NWT method is implemented in a way to inherit all unique computational techniques, except the "excess charge", already implemented in the DelPhi solver. One can view this new NWT method as a DelPhi-specialized Newton-like method which is not seen elsewhere.

2.4. Comparison

Three iteration formulas, Eqs (5), (9) and (12), have been presented in this section for solving the NLPBE. It will be interesting to compare them side by side and provide our understanding of these iteration formulas. In order to simplify the discussions, we assume that a mesh size h is fixed and $h \ll 1$. We focus on just one iteration step, the n th iteration, in which the potential ϕ_0^n at an arbitrary grid is evolved to ϕ_0^{n+1} by one of these three formulas. Moreover, we assume all potentials on the right-hand side of three equations all take on the same values in the n th iteration step. Noticing that the three formulas become identical at grids inside the molecule/protein because the modified Debye-Huckel parameter $\kappa(x) = 0$ in this case. Therefore, performance differences can only be observed at grids immersed in water. We thus limit our analysis to the solvent domain, where the potential function is smooth and bounded because no point charges locate there. Thus, in the water, it is reasonable to assume $\phi_i^n \approx \phi_0^n, i = 1, \dots, 6$ in these formulas for a small but fixed h . When being stable, these three formulas will converge to the same solution as n goes to infinity. Such a solution will be called the algebraic solution, which satisfies the finite difference discretization of the NLPBE, i.e., Eq (3).

We will investigate these three formulas in two aspects, i.e., compare their convergence rates and analyze their stabilities when the potential is large. Equation (5) is considered first. When ϕ_0^n is small, Eq (5) is reduced to Eq (6) by approximating $\sinh(\phi_0^n) \approx \phi_0^n$. In addition, by the assumption of $\phi_i^n \approx \phi_0^n, i = 1, \dots, 6$, Eq (5) can be viewed as a linear function, $\phi_0^{n+1} \approx a\phi_0^n + b$ for some constants a and b , where $0 < a < 1$. Thus, ϕ_0^{n+1} converges in a linear rate with respect to ϕ_0^n . When ϕ_0^n is large but still on track, the right-hand side of Eq (5) has a much large denominator than that of Eq (6) because $\frac{\sinh(\phi_0^n)}{\phi_0^n} \gg 1$. This drives Eq (6) to converge to the NLPBE potential.

Nevertheless, when ϕ_0^n is large and away from the limiting value, stability has to be analyzed. Assuming $\phi_i^n \approx \phi_0^n$ and neglecting constants, the dominate term of Eq (5) can be expressed as $C(\phi_0^n)^2 / \sinh(\phi_0^n)$ for some constant C . Because the denominator is much larger than the numerator, this iteration will not blow up and thus remains stable. In total, we view the series of potentials $\{\phi_0^{n+1}\}_{n=0,1,2,\dots}$ calculated by Eq (5) are stably converging to the algebraic solution of Eq (3). However, it is known that $\{\phi_0^{n+1}\}_{n=0,1,2,\dots}$ converges not quickly enough for solving three-dimensional problems [44].

The SOR method utilizing Eq (9) aims at making the iteration process converge in a faster pace. To see this, we rewrite Eq (9) as

$$\begin{aligned}
\phi_0^{n+1} &= \left(\sum_{i=1}^6 \varepsilon_i \phi_i^n + \frac{4\pi q_0}{h} - \frac{\chi(\kappa(x)^2(\sinh(\phi_0^n) - \phi_0^n))}{h} \right) / \left(\sum_{i=1}^6 \varepsilon_i + (\kappa h)^2 \right) \\
&= \frac{\sum_{i=1}^6 \varepsilon_i \phi_i^n + \frac{4\pi q_0}{h}}{\sum_{i=1}^6 \varepsilon_i + (\kappa h)^2} - \frac{\chi(\kappa(x)^2(\sinh(\phi_0^n) - \phi_0^n))/h}{\sum_{i=1}^6 \varepsilon_i + (\kappa h)^2},
\end{aligned} \tag{17}$$

where the first term on the right-hand side is the same as the right-hand side of Eq (6), and the second term can be viewed as a “correction” added to the first term for improved convergence rate. When ϕ_0^n is small, $\sinh(\phi_0^n) \approx \phi_0^n$ so that the correction term does not contribute much to ϕ_0^{n+1} . In this case Eq (9) converges in a similar rate as that of Eqs (5) and (6). When ϕ_0^n is large, $\sinh(\phi_0^n) - \phi_0^n \gg 1$ and $(\sinh(\phi_0^n) - \phi_0^n)/h$ is even larger provided $h \ll 1$ so that the correction term becomes a significant portion in ϕ_0^{n+1} and drives ϕ_0^{n+1} in an accelerated pace towards the algebraic solution of the discretized NLPBE. However, the correction term could also introduce additional issues. When $\phi_0^n \gg 1$, the value of $(\sinh(\phi_0^n) - \phi_0^n)/h$ could drive ϕ_0^{n+1} stride to be overshoot the solution. Assuming $\phi_i^n \approx \phi_0^n$ and neglecting constants, the dominate term of Eq (6) takes a form of $a \sinh(\phi_0^n) - b\phi_0^n$ for some constants a and b . Consequently, the potential could grow exponentially, and the whole iteration process quickly diverges. DelPhi utilizes a couple of relaxations parameters, ω in Eq (7) and χ in Eq (9), in order to pull ϕ_0^{n+1} back to the range of the solution in the overshoot situation. These relaxation techniques work in most situations, but there is no guarantee that they are always effective. For instance, one “crashing” example is demonstrated in the next section that the SOR method faces severe difficulties to converge.

Equation (9) has been proven to cope with most cases in practice and it has other advantages over Eq (5). First of all, it allows the same computational and programming techniques flawlessly shared between solving the LPBE and NLPBE. Secondly, the denominator on the right-hand side of Eq (9) is unchanged in all iterations so that it can be calculated once, saved and then reused in all iterations. It is very computationally economical. Third, Eq (9) can collaborate with other advanced techniques in DelPhi, resulting in one of the best PBE solvers in the world. Overall, we believe the SOR method implemented in DelPhi is an effective method to solve the NLPBE for three-dimensional problems.

The newly developed NWT method utilizes Eq (12) in order to maintain stability when solving the NLPBE for problems with strong nonlinearity, while it is still able to converge in a rate faster than the method using Eq (5). To see this, we rewrite Eq (12) as

$$\begin{aligned}
\phi_0^{n+1} &= \frac{\sum_{i=1}^6 \varepsilon_i \phi_i^n + \frac{4\pi q_0}{h} + (\kappa h)^2(\phi_0^n \cosh(\phi_0^n) - \sinh(\phi_0^n))}{\sum_{i=1}^6 \varepsilon_i + (\kappa h)^2 \cosh(\phi_0^n)} \\
&= \frac{\sum_{i=1}^6 \varepsilon_i \phi_i^n + \frac{4\pi q_0}{h}}{\sum_{i=1}^6 \varepsilon_i + (\kappa h)^2 \cosh(\phi_0^n)} + \frac{(\kappa h)^2(\phi_0^n \cosh(\phi_0^n) - \sinh(\phi_0^n))}{\sum_{i=1}^6 \varepsilon_i + (\kappa h)^2 \cosh(\phi_0^n)} \\
&= \frac{\sum_{i=1}^6 \varepsilon_i \phi_i^n + \frac{4\pi q_0}{h}}{\sum_{i=1}^6 \varepsilon_i + (\kappa h)^2 \cosh(\phi_0^n)} + \frac{(\kappa h)^2(\phi_0^n - \tanh(\phi_0^n))}{\sum_{i=1}^6 \varepsilon_i / \cosh(\phi_0^n) + (\kappa h)^2},
\end{aligned} \tag{18}$$

where the first term on the right-hand side is similar to that of Eq (17) with one additional $\cosh(\phi_0^n)$ in

the denominator, and the second term, which is still called the correction term, is new in the NWT method. When ϕ_0^n is small, the first term is practically the same as that of Eq (17) because $\cosh(\phi_0^n) \approx 1$, and the second term vanishes because $\tanh(\phi_0^n) \approx \phi_0^n$. In this case Eq (12) converges in a similar rate as that of Eqs (5), (6) and (9). When ϕ_0^n is large but still on track, $|\sinh(\phi_0^n)| \approx \cosh(\phi_0^n) \gg |\phi_0^n| > 1$ so that the first term on the right-hand side of Eq (18) is smaller than the right-hand side of Eq (5). Together with the second term, this will drive the potential convergent to the algebraic solution of the discretized NLPBE in a faster pace than that of Eq (5). When ϕ_0^n is large and far apart from the algebraic solution, the dominate term of Eq (12) behaves like $a\phi_0^n + b \tanh(\phi_0^n) + c\phi_0^n / \cosh(\phi_0^n)$ for some constants a , b , and c by assuming $\phi_i^n \approx \phi_0^n$ and neglecting constants. This iteration only grows linearly as ϕ_0^n increases. This is essentially why the NWT method is more stable than the SOR method.

In summary, we believe that Eq (5) could provide a stable method to solve the NLPBE. However, its relatively low convergence rate makes it unsuitable to solve the NLPBE for three-dimensional problems. The SOR method improves the convergence rate by an “exponential” correction term. This correction term allows the iterations progress in a fast pace, but it could lead to unexpected divergence for problems with high nonlinearity. The NWT method substitutes the correction term with a moderate one to balance the needs for both efficiency and stability, and we expect it to be a useful alternative of the SOR method in DelPhi to solve problems with high nonlinearity.

3. Results

Benchmarks are presented to compare the SOR and NWT methods in this section. Both methods have been implemented in DelPhi using the same computational and programming techniques. A wide selection of examples was tested, and three examples are chosen to demonstrate due to the limited length of this work.

Example 1. In the first example, we show that both methods are capable of producing close numerical approximations to the algebraic solution of the discretized NLPBE at a given mesh size h . To this end, a basic example of barnase-barstar complex (subfigure in Figure 2b) is borrowed from DelPhi’s online example repository <http://compbio.clemson.edu/delphi> and the NLPBE is solved for this complex. Two potential-dependent energies, the total grid energy G_g and the corrected reaction field (RXN) energy G_r , are used to compare the accuracy of the two methods.

The first series of benchmarks is conducted to show that both methods produce closer approximations as the mesh size h diminishes. In DelPhi the mesh size h is controlled by a parameter *scale*, defined to be the number of grids per angstrom. The mesh size h decreases as the scale increases. In this series of benchmarks, the tolerance is fixed, $\text{TOL} = 1.0\text{E}-4$, and the scale is varied from $\text{scale} = 0.5$ (29 grids per direction) to $\text{scale} = 5.0$ (293 grids per direction). Energies obtained by the SOR and NWT methods are denoted by G_g^{SOR} , G_r^{SOR} , G_g^{NWT} and G_r^{NWT} , respectively. Results are shown in Figure 2.

Obtained energies are shown in Figure 2a,b first. In both subfigures, it is noticed that energies obtained by the SOR method are always slightly larger than their comparative partners obtained by the NWT method at all tested scales. It shall be pointed out that it may not be the case for other molecules and proteins. It could just be caused by, for instance, the parameter values used in the tests, the initial values used for the iterations, and other factors. Nevertheless, close energies obtained by the two methods at all tested scales evidently demonstrate that they are converging to the exact energies. More detailed comparisons were performed and reported in Figure 2c,d. In these subfigures, the differences, defined as

$G_g^{SOR} - G_g^{NWT}$ and so on, and the relative differences, defined as $(G_g^{SOR} - G_g^{NWT})/G_g^{SOR} \times 100\%$ and so on, are shown. The differences, except those at a low scale = 1.0, are seen to approach to a value as small as < 5 KT as the scale increases (Figure 2c), while the relative differences, starting with an already low percentage $\approx 1.7\%$, consistently converge to zero as the scale increases (Figure 2d).

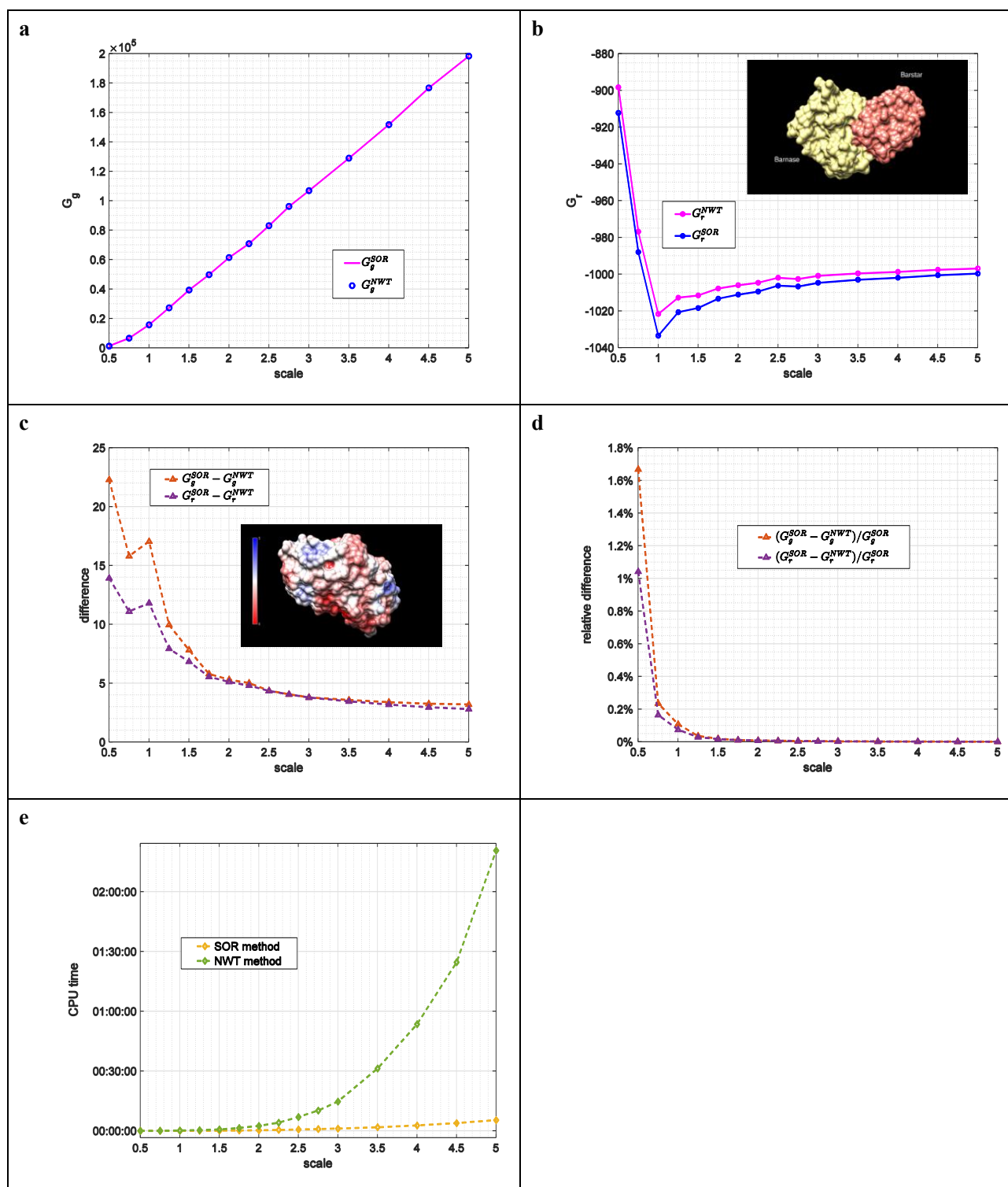


Figure 2. Benchmarks obtained by the SOR and NWT methods with a fixed tolerance $TOL = 1.0E-4$ and various scales in Example 1. (a) Grid energies. (b) Reaction field energies. (c) Differences. (d) Relative differences. (e) CPU time.

In the light that both methods acquire close approximations at all tested scales, and the approximations are getting closer as the scale increases, we conclude that both methods are obtaining close approximations to the same algebraic solutions of the NLPBE at all tested mesh sizes in this example.

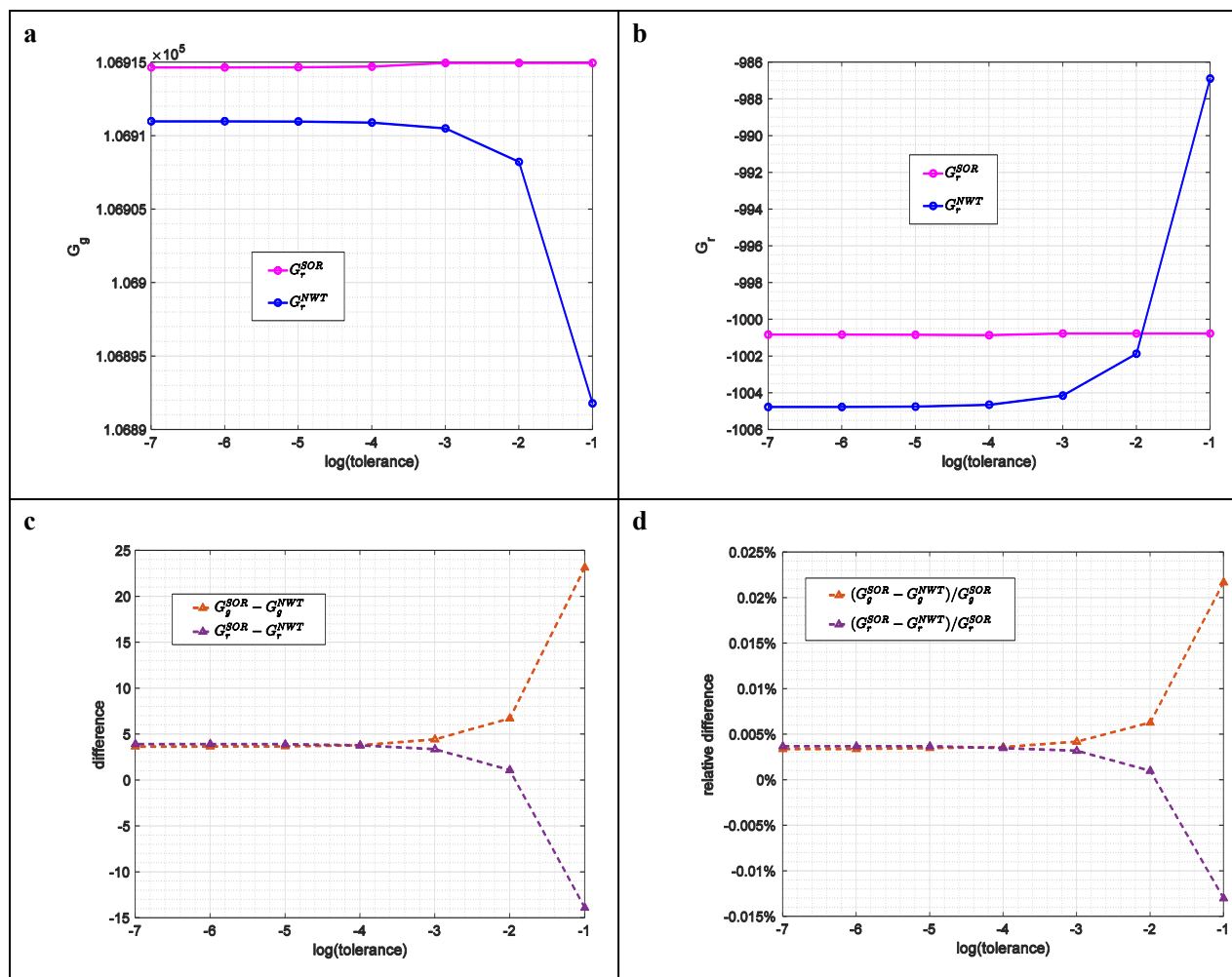


Figure 3. Benchmarks obtained by the SOR and NWT methods with a fixed scale = 2.0 and various tolerances in Example 1. (a) Grid energies. (b) Reaction field energies. (c) Differences. (d) Relative differences.

Corresponding execution time of the DelPhi program is demonstrated in Figure 2e to compare the efficiency of these two methods. One can see that the SOR method costs less time, and therefore is more efficient, at all tested scales. It is primarily due to two reasons. First, the SOR method starts off the nonlinear iterations with better initial values achieved by solving the LPBE for a few dozens of iterations. This numerical treatment significantly reduces the numbers of more costly nonlinear iterations. On the other hand, the NWT method merely uses the default initial values (zeros on all grids) without additional treatments. Secondly, by reusing the saved denominator, each iteration of the SOR method is computationally cheaper than that of the NWT method. As a consequence, the SOR method is found to be more computationally efficient than the NWT method for solving the NLPBE in this example, and it is believed to be the case for many other problems as well.

It has been shown that both methods are capable of achieving close approximations at all scales. We are also interested to see that how fast, in terms of the number of iterations, these two methods can achieve their best approximations at a given scale. To this end, the second series of benchmarks is performed by fixing the scale = 3.0 (175 grids per direction) and varying the tolerance from $1.0\text{E}-1$ to $1.0\text{E}-7$. It is naturally expected that both methods will take more iterations, and therefore produce more accurate energies, when smaller tolerance is used. The results shown in Figure 3 match our expectation. Semi-log plots (the horizontal axis is the logarithm of the tolerance) are used in Figure 3 that the scale decreases from the right to left.

Energies, G_g and G_r , obtained by the two methods are presented in Figure 3a,b. One can see that the SOR method shows its stunning efficiency in this example. The obtained curves for the SOR method (pink curves) are almost flat in both subfigures, implying that the SOR method achieves its best approximations without requiring many iterations. In contrast, the NWT method (blue curves) behaves differently: it takes less iterations and achieves coarser approximations when the tolerance is large, and it takes more iterations and obtains finer approximations as the tolerance is smaller. The observed different convergent trends of these two methods can be explained using Eqs (17) and (18). The SOR method tends to add a “big” correction in each iteration to pull ϕ_0^{n+1} into the range of its best approximation as quickly as possible, so that it does not take too many iterations to attain its best approximation despite which tolerance is actually used. On the contrary, the NWT method adds a moderate correction in each iteration so that it takes more iterations to attain its best, and the approximations are observed to gradually approach to the best as the tolerance decreases. Another important observation on Figure 3a,b is that G_r^{NWT} is larger than G_r^{SOR} at tolerance = $1.0\text{E}-1$ in Figure 3b. Thus, it is not true that the NWT method always obtains smaller energies.

Differences and relative differences are presented in Figure 3c,d. In both subfigures, all differences and relative differences are found to converge as the tolerance decreases. At the smallest tolerance = $1.0\text{E}-7$, the differences of G_g and G_r are found to be as close as < 5 KT in Figure 3c, and the relative differences are found to be as close as $< 0.005\%$ in Figure 3d. CPU times of this series of benchmarks are omitted because they are consistent to what shown in Figure 2e—the SOR method is more time consuming than the NWT method in all tested cases.

Example 2. Results obtained in the first example have provided some insights on the performance of the two methods. We continue to study these two methods for a blindly selected group of proteins. This group of proteins is composed of 15 dimers, and each of them consists of two monomers, namely monomer A and B. More energies, in addition to G_g and G_r , returned by DelPhi will be reported in this example. In particular, they will be used to calculate the binding energy, denoted by $\Delta G(\text{bind})$, in this example. Two approaches were suggested in the work [34] to calculate the binding energy. The first approach (approach 1) calculates the electrostatic component of the binding energy from the total nonlinear grid energies of the complex, monomer A and B by

$$\Delta G_1(\text{bind}) = G_g(\text{complex}) - G_g(A) - G_g(B), \quad (19)$$

and the second approach (approach 2) calculates the binding energy from partitioned energies by

$$\Delta G_2(\text{bind}) = \Delta G_r + \Delta G_p + \Delta G_o + \Delta G_i + \Delta G_c. \quad (20)$$

where G_g and G_r again denote the total grid energy and the corrected reaction field energy,

respectively, G_ρ denotes the $\frac{\rho}{\phi} * 2$ term in solution, G_0 denotes the osmotic pressure term, G_i denotes the direct ionic contribution inside the box, G_c denotes the Coulombic energy, and $\Delta G_{\blacksquare}$ with the subscript $\blacksquare = r, \rho, o, i, c$ denotes corresponding partitioned energy similar to that defined in Eq (19). Even though $\Delta G_1(\text{bind})$ and $\Delta G_2(\text{bind})$ are both used to approximate the exact binding energy $\Delta G(\text{bind})$, it has been pointed out in the work [34] that they are actually slightly different due to the fact that approach 1 does not fully cancel “artificial grid energy” arising from real charges partitioning onto the grids. Thus, $\Delta G_1(\text{bind})$ is always slightly larger than $\Delta G_2(\text{bind})$. Approach 2 via the energy partition technique does not have such issue so that it is recommended over approach 1.

We first show that binding energies calculated via those returned by DelPhi in solving the NLPBE via the SOR and NWT methods are close. To this end, binding energies calculated by both approaches are demonstrated for one dimer, *Ifle*. The NLPBE is solved by the SOR and NWT methods with a fixed tolerance = 1.0E−4 and various scales. Calculated binding energies are denoted by $\Delta G_1^{\text{SOR}}(\text{bind})$, $\Delta G_2^{\text{SOR}}(\text{bind})$, $\Delta G_1^{\text{NWT}}(\text{bind})$, and $\Delta G_2^{\text{NWT}}(\text{bind})$, respectively, and demonstrated in Figure 4. A couple of observations can be made on Figure 4. First of all, the two binding energies have the same trend as those obtained by the SOR method that $\Delta G_1^{\text{NWT}}(\text{bind})$ (solid green curve) is always slightly larger than $\Delta G_2^{\text{NWT}}(\text{bind})$ (dashed brown curve) at all tested scales. It matches the statements in the work [34]. Secondly, one can see that $\Delta G_1^{\text{SOR}}(\text{bind})$ and $\Delta G_1^{\text{NWT}}(\text{bind})$ (two solid curves) converge to $\Delta G_1(\text{bind})$, while $\Delta G_2^{\text{SOR}}(\text{bind})$ and $\Delta G_2^{\text{NWT}}(\text{bind})$ (two dashed curves) converge to $\Delta G_2(\text{bind})$, as the scale increases. It is also interesting to point out another important observation, which is not shown in Figure 4. In the benchmarks of dimer *Ifle*, we observed that the SOR method is faster than the NWT method in most tested cases. However, there are a few cases in which the NWT method uses the default $\omega = 1.0$ and converges without any issue, while the SOR method needs a smaller relaxation parameter, $\omega = 0.5$, in order to converge. When it occurs, the SOR method takes significantly more iterations and becomes slower than the NWT method.

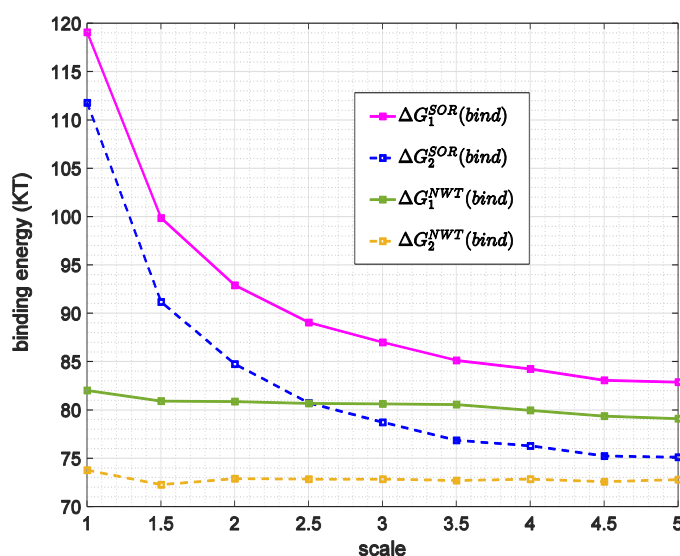


Figure 4. Binding energies obtained by solving the NLPBE via the SOR and NWT methods with a fixed tolerance TOL = 1.0E−4 and various scales for dimer *Ifle* in Example 2.

The next series of benchmarks was performed to calculate the binding energies at a fixed scale = 2.0 (the most commonly used scale in practice) for all 15 dimers. Results are presented in Figure 5

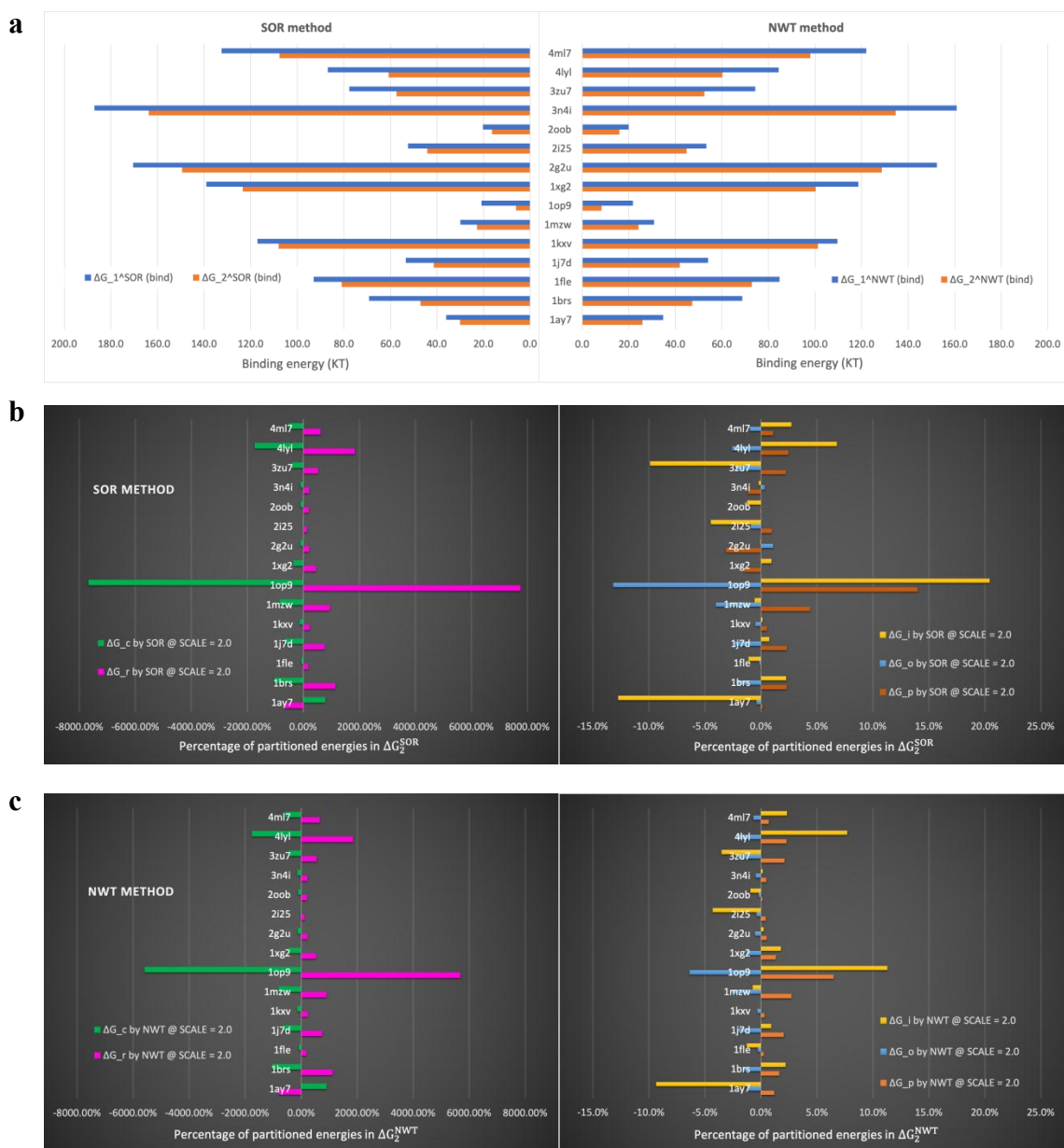


Figure 5. Binding energies and partitioned energies obtained on 15 dimes in Example 2. (a) Binding energies obtained by the SOR method (left panel) and the NWT method (right panel). (b) Percentages of partitioned energies in the binding energy $\Delta G_2^{SOR}(\text{bind})$. (c) Percentages of partitioned energies in the binding energy $\Delta G_2^{NWT}(\text{bind})$. Partitioned energies with large magnitudes are shown on the left panel and the remaining energies are shown on the right panel in Figure 5b,c.

In Figure 5a, the SOR-generated binding energies ($\Delta G_1^{SOR}(\text{bind})$ and $\Delta G_2^{SOR}(\text{bind})$) are demonstrated on the left panel, and the NWT-generated binding energies ($\Delta G_1^{NWT}(\text{bind})$ and

$\Delta G_2^{\text{NWT}}(\text{bind})$) are demonstrated on the right panel. By comparing each blue bar to its paired orange bar on both panels, one can see that the binding energies obtained by approach 1 are always larger than those obtained by approach 2 for all 15 proteins. It is the case for both SOR and NWT methods. Next, by comparing bars in the same color for each dimer on the left and right panels, one can see visible differences on the SOR- and NWT- generated binding energies. However, given the experiences achieved for dimer *Ile*, it is reasonable to expect that these differences are going to diminish if a larger scale is used.

We are interested in seeing how much each individual partitioned energy contributes in the calculated binding energies. Taking $\Delta G_2(\text{bind})$ calculated by Eq (20) in approach 2 as an example, the percentages of partitioned energies in the binding energy, defined as $\Delta G_{\square}/\Delta G_2(\text{bind}) \times 100\%$, are shown in Figure 5b for the SOR method, and Figure 5c for the NWT method, respectively. Percentages of two partitioned energies, ΔG_r and ΔG_c , are found to be significantly larger than those of other partitioned energies. Therefore, they are presented on the left panel and others are presented on the right panel in both Figure 5b,c. In these subfigures, one can see that ΔG_r and ΔG_c are always in opposite signs for all 15 dimers, and their sum, $\Delta G_r + \Delta G_c$, contributes more than 90% of $\Delta G_2(\text{bind})$, while the sum of the remaining three, $\Delta G_p + \Delta G_o + \Delta G_i$, contributes less than 10% of $\Delta G_2(\text{bind})$, for all 15 dimers. Moreover, by comparing corresponding energies, it is easy to see that the two methods, SOR and NWT, not only produce similar binding energy $\Delta G_2(\text{bind})$ as a sum of 5 partitioned energies, but also produce similar individual partitioned energy. These partitioned energies, except the partitioned Coulombic energy ΔG_c , all depend on the potentials calculated via the SOR and NWT methods. It suggests that the two methods indeed produce close potentials for all 15 dimers.

Above experiments at scale = 2.0 were repeated at a doubled scale, scale = 4.0, and the differences shown in Figure 5 are found to be consistently smaller for all 15 dimers. It evidently shows that one can confidently relies on the energies produced by DelPhi using either method when the iteration process converges at the end. Moreover, we have observed more cases in which the SOR method requires smaller relaxation parameter to converge, while the NWT method has no such issue at all, in the cases tested at scale = 4.0. It inspires us to perform more tests to examine the stability of the two methods.

Example 3. It has been observed in Example 2 that the SOR method may require smaller relaxation parameter in order to successfully converge in some cases, while the NWT method never has such issue. Out of abundance of caution, a “crashing” example is purposely created and examined to numerically verify that the NWT method is still able to converge even in some rare and extreme scenarios before we claim that the NWT is a strongly stable method for solving the NLPBE.

This example was tested with a fixed tolerance, $\text{TOL} = 1.0\text{E}-4$, and numerous scales ranging from 1.0 to 5.0. This example is believed to be “bizarre” that the iteration process of the SOR method can never be terminated by meeting the desired tolerance at all tested scales. The iteration process is hindered only after a few iterations when the differences of calculated potentials in two successive iterations are large at some grids, causing the SOR method relentlessly seek for smaller relaxation parameter ω to reduce these differences before moving forward to the next iteration. This effort repeats many times in each of the first several iterations and prevents the iterations progress properly towards the end. As a consequence, the SOR method fails to produce any energies in this example after waiting for a long time.

It is a completely different story for the NWT method. The NWT method merely uses the default $\omega = 1.0$ and converges successfully in all tested cases. Energies produced by DelPhi running the NWT method are presented by a semi-log plot (the vertical axis is the logarithms of the absolute values of the energies) in Figure 6. One can see that all energies behave normally without any unanticipated outcomes.

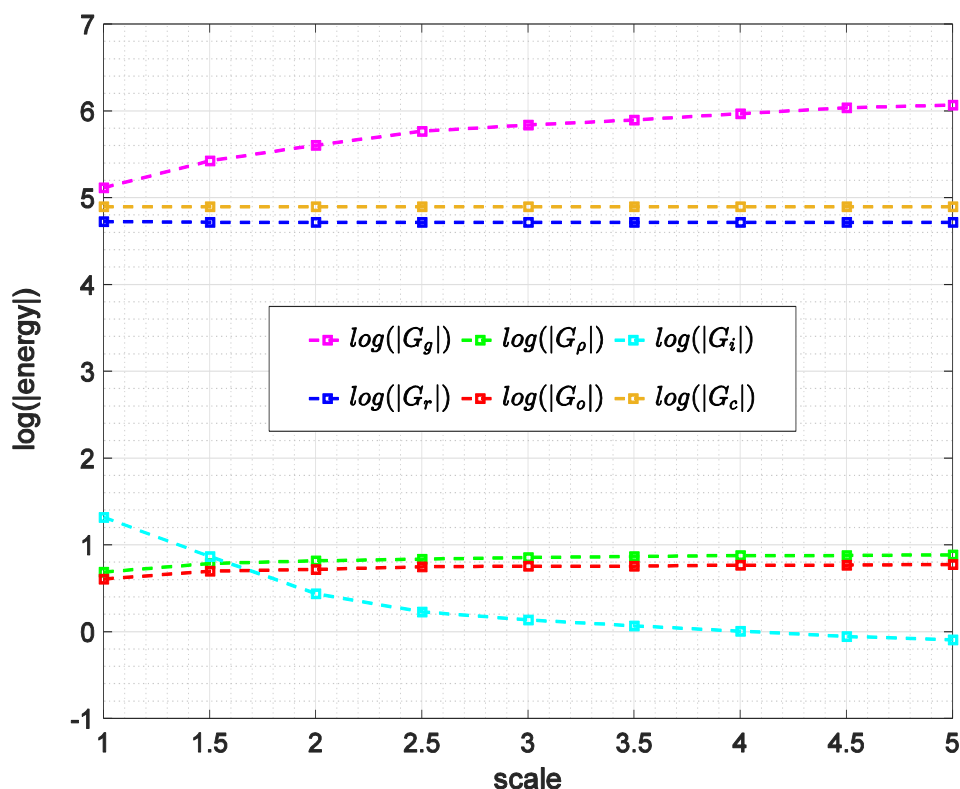


Figure 6. DelPhi returned energies obtained by solving the NLPBE via the NWT method with a fixed tolerance = $1.0\text{E}-4$ in Example 3.

Additional examples beside Example 3 have been tested as well and we have not seen one case that the NWT method fails to converge. The experiences we earned make us confidently claim that the newly developed NWT method is a reliable alternative to solve the NLPBE for problems with high nonlinearity. Meanwhile, bearing in mind that the SOR method is still more efficient in many cases, the SOR method is still recommended to solve the LPBE/NLPBE when no divergence issue takes place. In the cases that the SOR method has troubles to converge, one can immediately observe in DelPhi's outputs that the iteration stops progressing forward, the relaxation parameter becomes smaller, and the calculated tolerances get larger. It will be enough to tell that the SOR method is having troubles to converge, and it is advised to stop the program and switch to the NWT method.

4. Discussions and conclusions

In this work, a newly developed Newton-like method is proposed. It has been implemented in the DelPhi program to solve the PBE for electrostatic potentials. It has been demonstrated that the NWT method is relatively slower, equally accurate, and more stable compared to the SOR method

for solving the NLPBE. The merits of the new NWT method make it a valuable add-on to the DelPhi program. The NWT method is recommended to the computational molecular society to solve the NLPBE for problems with strong nonlinearity when other solvers have trouble to converge and deliver reliable solutions. Developments to improve the efficiency of the NWT method will be carried out and reported in the future.

Acknowledgments

The research of E.A. and C.L was supported by a grant from NIH, grant number R01GM093937. The research of S.Z. was supported in part by National Science Foundation under grant DMS-1812930.

Conflict of interest

The authors declare that there is no conflict of interest regarding the publication of this article.

References

1. G. A. Cisneros, M. Karttunen, P. Ren, C. Sagui, Classical electrostatics for biomolecular simulations, *Chem. Rev.*, **114** (2014), 779–814.
2. B. Honig, A. Nicholls, Classical electrostatics in biology and chemistry, *Science*, **268** (1995), 1144–1149.
3. Z. Zhang, S. Witham, E. Alexov, On the role of electrostatics in protein–protein interactions, *Phys. Biol.*, **8** (2011), 035001.
4. J. Batra, A. Szabó, T. R. Caulfield, A. S. Soares, M. Sahin-Tóth, E. S. Radisky, Long-range electrostatic complementarity governs substrate recognition by human chymotrypsin C, a key regulator of digestive enzyme activation, *J. Biol. Chem.* **288** (2013), 9848–9859.
5. H. Ikeuchi, Y. M. Ahn, T. Otokawa, B. Watanabe, L. Hegazy, J. Hiratake, et al., A sulfoximine-based inhibitor of human asparagine synthetase kills L-asparaginase-resistant leukemia cells, *Bioorg. Med. Chem.*, **20** (2012), 5915–5927.
6. X. Huang, F. Dong, H. X. Zhou, Electrostatic recognition and induced fit in the κ -PVIIA toxin binding to Shaker potassium channel, *J. Am. Chem. Soc.*, **127** (2005), 6836–6849.
7. E. Alexov, Numerical calculations of the pH of maximal protein stability: The effect of the sequence composition and three-dimensional structure, *Eur. J. Biochem.*, **271** (2004), 173–185.
8. A. Isvoran, C. Craescu, E. Alexov, Electrostatic control of the overall shape of calmodulin: numerical calculations, *Eur. Biophys. J.*, **36** (2007), 225–237.
9. R. C. Mitra, Z. Zhang, E. Alexov, In silico modeling of pH-optimum of protein–protein binding, *Proteins: Struct., Funct., Bioinf.*, **79** (2011), 925–936.
10. A. V. Onufriev, E. Alexov, Protonation and pK changes in protein–ligand binding, *Q. Rev. Biophys.*, **46** (2013), 181–209.
11. K. Talley, E. Alexov, On the pH - optimum of activity and stability of proteins, *Proteins: Struct., Funct., Bioinf.*, **78** (2010), 2699–2706.
12. M. Petukh, T. Kimmet, E. Alexov, BION web server: predicting non-specifically bound surface ions, *Bioinformatics*, **29** (2013), 805–806.

13. M. Petukh, M. Zhang, E. Alexov, Statistical investigation of surface bound ions and further development of BION server to include p H and salt dependence, *J. Comput. Chem.*, **36** (2015), 2381–2393.
14. M. Petukh, M. Zhenirovskyy, C. Li, L. Li, L. Wang, E. Alexov, Predicting nonspecific ion binding using DelPhi, *Biophys. J.*, **102** (2012), 2885–2893.
15. E. Alexov, E. L. Mehler, N. Baker, A. M. Baptista, Y. Huang, F. Milletti, et al., Progress in the prediction of pKa values in proteins, *Proteins: Struct., Funct., Bioinf.*, **79** (2011), 3260–3275.
16. R. E. Georgescu, E. G. Alexov, M. R. Gunner, Combining conformational flexibility and continuum electrostatics for calculating pKas in proteins, *Biophys. J.*, **83** (2002), 1731–1748.
17. M. R. Gunner, N. A. Baker, Continuum electrostatics approaches to calculating pKas and Ems in proteins, *Methods Enzymol.*, **578** (2016), 1–20.
18. C. Bertonati, B. Honig, E. Alexov, Poisson-Boltzmann calculations of nonspecific salt effects on protein-protein binding free energies, *Biophys. J.*, **92** (2007), 1891–1899.
19. J. H. Bredenbergh, C. Russo, M. O. Fenley, Salt-mediated electrostatics in the association of TATA binding proteins to DNA: a combined molecular mechanics/Poisson-Boltzmann study, *Biophys. J.*, **94** (2008), 4634–4645.
20. A. Ghosh, C. S. Rapp, R. A. Friesner, Generalized Born model based on a surface integral formulation, *J. Phys. Chem. B*, **102** (1998), 10983–10990.
21. P. Grochowski, J. Trylska, Continuum molecular electrostatics, salt effects, and counterion binding—a review of the Poisson–Boltzmann theory and its modifications, *Biopolym.: Orig. Res. Biomol.*, **89** (2008), 93–113.
22. N. A. Baker, Poisson–Boltzmann methods for biomolecular electrostatics, *Methods Enzymol.*, **383** (2004), 94–118.
23. L. Xiao, J. Diao, D. A. Greene, J. Wang, R. Luo, A continuum Poisson–Boltzmann model for membrane channel proteins, *J. Chem. Theory Comput.*, **13** (2017), 3398–3412.
24. C. Li, L. Li, M. Petukh, E. Alexov, Progress in developing Poisson-Boltzmann equation solvers, *Comput. Math. Biophys.*, **1** (2013), 42–62.
25. J. Mongan, C. Simmerling, J. A. McCammon, D. A. Case, A. Onufriev, Generalized Born model with a simple, robust molecular volume correction, *J. Chem. Theory Comput.*, **3** (2007), 156–169.
26. M. Feig, A. Onufriev, M. S. Lee, W. Im, D. A. Case, C. L. Brooks III, Performance comparison of generalized born and Poisson methods in the calculation of electrostatic solvation energies for protein structures, *J. Comput. Chem.*, **25** (2004), 265–284.
27. L. Li, C. Li, S. Sarkar, J. Zhang, S. Witham, Z. Zhang, et al., DelPhi: A comprehensive suite for DelPhi software and associated resources, *BMC Biophys.*, **5** (2012), 9.
28. W. Rocchia, S. Sridharan, A. Nicholls, E. Alexov, A. Chiabrera, B. Honig, Rapid grid - based construction of the molecular surface and the use of induced surface charge to calculate reaction field energies: Applications to the molecular systems and geometric objects, *J. Comput. Chem.*, **23** (2002), 128–137.
29. W. M. Botello-Smith, X. Liu, Q. Cai, Z. Li, H. Zhao, R. Luo, Numerical Poisson–Boltzmann model for continuum membrane systems, *Chem. Phys. Lett.*, **555** (2013), 274–281.
30. Y. Zhou, S. Zhao, M. Feig, G. W. Wei, High order matched interface and boundary method for elliptic equations with discontinuous coefficients and singular sources, *J. Comput. Phys.*, **213** (2006), 1–30.

31. E. Jurrus, D. Engel, K. Star, K. Monson, J. Brandi, L. E. Felberg, et al., Improvements to the APBS biomolecular solvation software suite, *Protein Sci.*, **27** (2018), 112–128.
32. A. H. Boschitsch, M. O. Fenley, A new outer boundary formulation and energy corrections for the nonlinear Poisson–Boltzmann equation, *J. Comput. Chem.*, **28** (2007), 909–921.
33. A. H. Boschitsch, M. O. Fenley, Hybrid boundary element and finite difference method for solving the nonlinear Poisson–Boltzmann equation, *J. Comput. Chem.*, **25** (2004), 935–955.
34. C. Li, Z. Jia, A. Chakravorty, S. Pahari, Y. Peng, S. Basu, et al., DelPhi Suite: New Developments and Review of Functionalities, *J. Comput. Chem.*, **40** (2019), 2502–2508.
35. 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., Bioinf.*, **1** (1986), 47–59.
36. W. Im, D. Beglov, B. Roux, Continuum solvation model: computation of electrostatic forces from numerical solutions to the Poisson-Boltzmann equation, *Comput. Phys. Commun.*, **111** (1998), 59–75.
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. B. A. Luty, M. E. Davis, J. A. McCammon, Solving the finite - difference non - linear Poisson–Boltzmann equation, *J. Comput. Chem.*, **13** (1992), 1114–1118.
39. M. J. Holst, F. Saied, Numerical solution of the nonlinear Poisson–Boltzmann equation: developing more robust and efficient methods, *J. Comput. Chem.*, **16** (1995), 337–364.
40. Q. Cai, M. J. Hsieh, J. Wang, R. Luo, Performance of nonlinear finite-difference Poisson–Boltzmann solvers, *J. Chem. Theory Comput.*, **6** (2010), 203–211.
41. A. Shestakov, J. Milovich, A. Noy, Solution of the nonlinear Poisson–Boltzmann equation using pseudo-transient continuation and the finite element method, *J. Colloid Interface Sci.*, **247** (2002), 62–79.
42. A. Sayyed–Ahmad, K. Tuncay, P. J. Ortoleva, Efficient solution technique for solving the Poisson–Boltzmann equation, *J. Comput. Chem.*, **25** (2004), 1068–1074.
43. W. Geng, S. Zhao, Fully implicit ADI schemes for solving the nonlinear Poisson-Boltzmann equation, *Comput. Math. Biophys.*, **1** (2013), 109–123.
44. 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.



AIMS Press

©2020 the Author(s), licensee AIMS Press. This is an open access article distributed under the terms of the Creative Commons Attribution License (<http://creativecommons.org/licenses/by/4.0>)