Journal of Computational and Applied Mathematics 481 (2026) 117270

Contents lists available at ScienceDirect

Journal of Computational and Applied Mathematics

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

Numerical solution to the three-dimensional p-Laplace equation: Finite difference methods and biological applications

Matthias Dogbatsey [a], Zhan Chen [b], Yuanzhen Shao [a], Shan Zhao [a,*]

[a] Department of Mathematics, The University of Alabama, Tuscaloosa, 35487, AL, USA
[b] Department of Mathematical Sciences, Georgia Southern University, Statesboro, 30460, GA, United States

ARTICLE INFO

2000 MSC:
65n06
92C40

Keywords:
p-Laplace
Poisson-Boltzmann (PB) model
Pseudo-time methods
Alternating direction implicit (ADI)
Biomolecular solvation analysis

ABSTRACT

Solving the nonlinear and degenerate p-Laplace equation numerically is highly challenging, yet it has numerous applications across various fields. In this work, we propose and explore numerical schemes, including Picard's Iteration scheme, Crank-Nicolson (CN) method, and Alternating Direction Implicit (ADI) scheme, for solving the p-Laplace equation in $\mathbb{R}^3$ for $p \in [1, 2)$, aiming to identify the most effective numerical approaches within the framework of finite difference methods. In the existing literature, studies on the parabolic p-Laplace equation primarily consider the Cauchy problem with $p \geq 2$. Our study bridges this gap by examining the parabolic p-Laplace equation for $p \in [1, 2)$. To our knowledge, this is the first comprehensive study on the three-dimensional (3D) case. Our results demonstrate that the ADI scheme is the most efficient among the proposed 3D numerical schemes as it requires the least computational time while maintaining comparable accuracy in both analytical tests and solvation energy calculations. Moreover, we numerically verify that the solution of the pseudo-time parabolic p-Laplace equation converges to the solution of the corresponding highly nonlinear elliptic p-Laplace equation.

1. Introduction

The p-Laplace equation has been widely studied in the literature and finds various applications in many fields such as population modeling [1], image processing [2], modeling of sand piles [3], and biomolecular solvation energy calculations [4]. It can be formulated as follows

$$\begin{cases} \nabla \cdot (|\nabla S|^{p-2} \nabla S) = f & \text{in } \Omega \\ S = g & \text{on } \partial\Omega. \end{cases} \tag{1.1}$$

The p-Laplace equation involves a strong nonlinear term, making it degenerate for $p > 2$ and singular for $p < 2$ at the critical points ($\nabla S = 0$) [5]. Consequently, numerically solving such nonlinear and degenerate elliptic partial differential equations (PDEs) is quite challenging [6].

In fact, the p-Laplace equation has garnered significant attention in the numerical analysis community. To our knowledge, a large body of work has focused on finite element methods (FEM). This preference stems from the fact that the p-Laplace equation is naturally suited for FEM due to its divergence form. Previous work on numerical solutions for the p-Laplacian has been devoted to solving the variational formulation of the p-Laplace equation. For example, Luo and Teng [7] introduced an effective FEM combined with Newton's method for the 2D p-Laplace equation, providing a robust framework for such nonlinear problems. In [8], the authors

* Corresponding author.
E-mail address: szhao@ua.edu (S. Zhao).

https://doi.org/10.1016/j.cam.2025.117270
Received 27 May 2025; Received in revised form 24 November 2025
Available online 21 December 2025

provided a theoretical analysis of the Crank-Nicolson Galerkin finite element method for solving the p-Laplace equation for an open bounded subset in $\mathbb{R}^d$ with $d = 2, 3$. This analysis was again based on the variational formulation of the problem. For completeness, we mention other work in the literature on the FEM discretization of the p-Laplacian [9–12]. Moreover, numerical techniques beyond FEM, such as steepest descent algorithms, have also been employed. Huang et al. [13] introduced a preconditioned steepest descent method to minimize a regularized functional for $p > 2$, offering another avenue for solving p-Laplacian problems. Additionally, Loisel's study [14] explored novel numerical schemes leveraging the barrier method with Newton iteration, which further expanded the computational techniques available for tackling the p-Laplace equation.

Studies have shown that monotone schemes are essential for the convergence proof of fully nonlinear or degenerate elliptic equations. While monotone schemes are generally unavailable for most higher-order finite element methods, they can be achieved using certain finite difference schemes, such as wide stencil finite difference methods. For example, Oberman in [6] introduced monotone finite difference schemes for the normalized p-Laplacian ($\Delta_p^N u := |\nabla u|^{2-p}\Delta_p u$) and the normalized infinity Laplacian $\Delta_\infty^N u = |\nabla u|^{-2}\sum_{ij}^d u_{x_i}u_{x_j}u_{x_ix_j}$ in $\mathbb{R}^d$. Furthermore, Oberman proposed a semi-implicit scheme for solving the p-Laplacian in the absence of a source term. Similarly, Codenotti et al. in [15] presented a finite difference method for the normalized p-Laplacian based on the mean value properties of the operator for $p \geq 2$. Recently, del Teso et al. [16] proposed a monotone finite difference scheme for the p-Laplace equation, making significant progress by solving fully nonhomogeneous p-Laplace equations using finite differences in $\mathbb{R}^d$. They introduced two algorithms: one based on the Newton-Raphson method and another employing an explicit approach, highlighting alternative discretization strategies outside the finite element method (FEM) paradigm. For further details, we refer the reader to other works exploring finite difference schemes for the normalized p-Laplace equation, particularly those based on the mean value formula [17–19]. However, challenges remain in designing fast solvers for the p-Laplace equation within the finite difference framework. Newton's method is known to be effective for equations with convex and differentiable structures, and has recently been augmented with neural networks for solving nonlinear PDEs [20]. Nevertheless, Newton's method is not applicable to general infinity Laplace or p-Laplace equations, as the discrete equations in these cases can be non-differentiable. We take this opportunity to emphasize that the methods discussed in the works referenced in this paragraph primarily address the Cauchy problem and are restricted to cases where $p \geq 2$. Notably, earlier studies on the normalized p-Laplace operator are not directly applicable to the parabolic case. This limitation underscores the need for alternative approaches to address these challenges.

For $p = 1$, the p-Laplace operator is often referred to as the mean curvature operator. In the absence of a source term (i.e., $f = 0$), the problem reduces to finding a minimal surface. Various finite difference numerical schemes have been developed to address this problem. In [21], the authors introduced four finite difference schemes for computing the minimal molecular surface. These included two alternating direction implicit (ADI) schemes, a Crank-Nicolson scheme, and a semi-discrete scheme. Among these methods, one ADI scheme demonstrated particularly high efficiency. Building upon this work, Tian and Zhao [22] proposed an improved ADI scheme, which provided superior speed, efficiency, and stability for minimal molecular surface calculations. The accuracy and efficiency of the ADI finite difference methods have also been investigated for solving other PDEs, such as Keller-Segel equations [23], acoustic wave equations [24], reaction-diffusion equations [25], and integrodifferential equations [26,27]. However, to the best of our knowledge, no ADI scheme has been developed for solving the 3D p-Laplace equation.

The objectives of this work are threefold. First, three finite difference methods will be developed for solving the p-Laplace equation, by using either the relaxation or pseudo-time continuation approach for treating the nonlinear operator. By rewriting the p-Laplace operator in the divergence form, a Picard's Iteration (PI) relaxation scheme will be proposed, in which $|\nabla S|^{p-2}$ will be evaluated using S values in the previous iteration, and a large linear system involving 3D variables will be solved by an iterative algorithm in each relaxation step. Similarly, in pseudo-time approaches, a time derivative is added to the equation and $|\nabla S|^{p-2}$ is evaluated based on the solution from previous time step. To approximate the solution of the p-Laplace equation, the associated parabolic problem must be solved until a steady state is reached. In this work, two implicit time-stepping schemes are proposed for this purpose: the Crank-Nicolson (CN) and the Alternating Direction Implicit (ADI) methods.

The second goal of this work is to explore the most effective finite difference methods for solving 3D p-Laplace equations in terms of accuracy and efficiency. Previous numerical studies have mainly focused on two-dimensional (2D) problems and cases with $p \geq 2$, while this work focuses on 3D problems for $p \in [1, 2)$. In this study, a variety of finite difference schemes have been examined and validated using benchmark analytical solution as well as biological applications. Among these, the PI, CN, and ADI methods demonstrated superior performance and are therefore presented in detail herein. To the best of our knowledge, this represents the first comprehensive study of the 3D p-Laplace equation for $p \in [1, 2)$.

The third aim of this study is to investigate several key questions related to the biological application of the p-Laplace equation in molecular surface generation [4]. Recently, the p-Laplace equation has been integrated into the formulation of promising computational models for predicting ensemble average solvation energy (EASE) in [4], which incorporate thermodynamic fluctuations during the solvation process. An elliptic PDE system, consisting of a coupled p-Laplace equation and a generalized Poisson-Boltzmann equation, is derived through the first variation of the p-energy functional. It turns out that the following questions are critical to such applications, and will be addressed in this study: 1) Does the solution of pseudo-time parabolic p-Laplace equation converge to the corresponding highly nonlinear elliptic p-Laplace equation? 2) How can the challenges of singularity and blow-up in this specific computational domain be addressed? Specifically, we will numerically examine issues of convergence and regularization. 3) Do the energies predicted by the models converge as $p \to 1^+$? Moreover, the proposed schemes will be applied to real solvation energy calculations for molecules and proteins.

The rest of the paper is organized as follows. In Section 2, we present the numerical algorithms proposed for solving the p-Laplace equation. We provide details on the discretization scheme, the proposed algorithms, and their implementation. Numerical validations

are provided in Section 3. In particular, we show the convergence and efficiency of our proposed schemes numerically using two different functions with analytic solutions. In Section 4, we apply the schemes to biological systems.

2. Mathematical models and numerical algorithms

In this section, we discuss the discretization of the various numerical schemes used to study the p-Laplacian system when $p = p_m = \frac{2m}{2m-1}$, and m is an integer [4]. It is worth mentioning that the algorithms proposed in this paper are applicable for all values of p, particularly $p \in [1, 2)$.

2.1. Finite difference discretization

We turn our attention to the finite difference discretization of the p-Laplacian system presented in Eq. (1.1). In this study, we consider a three-dimensional (3D) cuboid domain for Ω. Without loss of generality, we assume uniform grid spacing h in all three spatial directions, such that $h = \Delta x = \Delta y = \Delta z$, and denote the number of grid points along each direction as N_x, N_y, and N_z, respectively. To streamline the discussion, we introduce the notation S^n_{ijk} to represent the solution at the grid point (x_i, y_j, z_k) and time step t_n such that $S^n_{ijk} = S(x_i, y_j, z_k, t_n)$.

First, consider

$$\Delta_{p_m} S = \nabla \cdot \left(|\nabla S|^{p_m - 2} \nabla S \right).$$

Let the coefficient β be defined as

$$\beta = |\nabla S|^{p_m - 2}.$$

To improve numerical stability and avoid division by zero, a small positive regularization parameter η is incorporated into the calculation of the gradient's norm [28]

$$\beta = |\nabla S|^{p_m - 2} = \left(S_x^2 + S_y^2 + S_z^2 + \eta \right)^{(p_m - 2)/2}.$$

The p-Laplace operator can then be written as:

$$\Delta_{p_m} S = \nabla \cdot (\beta \nabla S).$$

By treating the right hand side as a non-homogeneous diffusion process in the divergence form, we have:

$$\Delta_{p_m} S = \left[\frac{\partial}{\partial x} \left(\beta \frac{\partial S}{\partial x} \right) \right] + \left[\frac{\partial}{\partial y} \left(\beta \frac{\partial S}{\partial y} \right) \right] + \left[\frac{\partial}{\partial z} \left(\beta \frac{\partial S}{\partial z} \right) \right].$$

To carry out the discretization, we will use a uniform mesh over the computational domain Ω as mentioned earlier. Our discretization approach is similar to the one presented in [22], and it makes use of half-grid points such as $\beta_{i+\frac{1}{2},j,k} = \beta(x_{i+\frac{1}{2}}, y_j, z_k)$, where $x_{i+\frac{1}{2}} = x_i + \frac{1}{2}h$. Approximating the spatial derivatives through second-order central finite differences gives:

$$\begin{aligned}
\left[\frac{\partial}{\partial x} \left(\beta \frac{\partial S}{\partial x} \right) \right]_{i,j,k} &\approx \frac{1}{h} \left(\beta \frac{\partial S}{\partial x} \right)_{i+\frac{1}{2},j,k} - \frac{1}{h} \left(\beta \frac{\partial S}{\partial x} \right)_{i-\frac{1}{2},j,k} \\
&\approx \frac{1}{h} \beta_{i+\frac{1}{2},j,k} \left(\frac{1}{h} S_{i+1,j,k} - \frac{1}{h} S_{i,j,k} \right) - \frac{1}{h} \beta_{i-\frac{1}{2},j,k} \left(\frac{1}{h} S_{i,j,k} - \frac{1}{h} S_{i-1,j,k} \right) \\
&= \frac{1}{h^2} \left[\beta_{i+\frac{1}{2},j,k} \left(S_{i+1,j,k} - S_{i,j,k} \right) + \beta_{i-\frac{1}{2},j,k} \left(S_{i-1,j,k} - S_{i,j,k} \right) \right].
\end{aligned}$$

Therefore, we introduce the following finite difference discretization.

$$\left[\frac{\partial}{\partial x} \left(\beta \frac{\partial S}{\partial x} \right) \right]_{i,j,k} \approx \delta_{xx} S_{i,j,k} := \frac{1}{h^2} \left[\beta_{i+\frac{1}{2},j,k} \left(S_{i+1,j,k} - S_{i,j,k} \right) + \beta_{i-\frac{1}{2},j,k} \left(S_{i-1,j,k} - S_{i,j,k} \right) \right], \tag{2.1}$$

$$\left[\frac{\partial}{\partial y} \left(\beta \frac{\partial S}{\partial y} \right) \right]_{i,j,k} \approx \delta_{yy} S_{i,j,k} := \frac{1}{h^2} \left[\beta_{i,j+\frac{1}{2},k} \left(S_{i,j+1,k} - S_{i,j,k} \right) + \beta_{i,j-\frac{1}{2},k} \left(S_{i,j-1,k} - S_{i,j,k} \right) \right], \tag{2.2}$$

$$\left[\frac{\partial}{\partial z} \left(\beta \frac{\partial S}{\partial z} \right) \right]_{i,j,k} \approx \delta_{zz} S_{i,j,k} := \frac{1}{h^2} \left[\beta_{i,j,k+\frac{1}{2}} \left(S_{i,j,k+1} - S_{i,j,k} \right) + \beta_{i,j,k-\frac{1}{2}} \left(S_{i,j,k-1} - S_{i,j,k} \right) \right]. \tag{2.3}$$

The half-grid values, such as $\beta_{i+\frac{1}{2},j,k}$, are given by:

$$\beta_{i+\frac{1}{2},j,k} = \left[\left(S_x^2 + S_y^2 + S_z^2 + \eta \right) \right]^{(p_m - 2)/2}_{i+\frac{1}{2},j,k}.$$

The discretizations of $\beta_{i+\frac{1}{2},j,k}$ and other β related terms are given by

$$\left\{ S_x^2 \right\}_{i+\frac{1}{2},j,k} \approx \left(\frac{S_{i+1,j,k} - S_{i,j,k}}{h} \right)^2,$$

$$\left\{S_y^2\right\}_{i+\frac{1}{2},j,k} \approx \left(\frac{S_{i,j+1,k} - S_{i,j-1,k}}{4h} + \frac{S_{i+1,j+1,k} - S_{i+1,j-1,k}}{4h}\right)^2,$$

$$\left\{S_z^2\right\}_{i+\frac{1}{2},j,k} \approx \left(\frac{S_{i,j,k+1} - S_{i,j,k-1}}{4h} + \frac{S_{i+1,j,k+1} - S_{i+1,j,k-1}}{4h}\right)^2.$$

We note that the above discretization is different from what was presented in Eqs. (2.1)-(2.3) due to the half-grid shifting in one cartesian direction.

We will propose three numerical schemes, categorized under two distinct approaches, for solving the p-Laplacian. The first approach employs an iterative method based on the Picard's Iteration (PI). In the second approach, we introduce a pseudo-time term into the p-Laplace equation, converting it into a time-dependent problem. We will develop two different time-stepping schemes to solve the pseudo-time p-Laplace equation, i.e., the Crank-Nicolson method and the Alternating Direction Implicit method.

2.2. An iterative method

We consider an iterative algorithm for solving the p-Laplacian with $S^n \to S$, as $n \to \infty$ using a stopping criterion $|S^{n+1} - S^n| < \text{TOL}$. This approach solves the p-Laplacian using a fixed-point iterative scheme. Given S^n, solve for S^{n+1} in:

$$\nabla \cdot \left(|\nabla S^n|^{p_m-2}\nabla S^{n+1}\right) = f. \tag{2.4}$$

By evaluating $|\nabla S^n|^{p_m-2}$ at the previous iteration step n, the relaxation used in Eq. (2.4) converts the nonlinear problem into a linear one in each iteration step.

2.2.1. Picard'S iteration

A Picard's Iteration method will be developed in this subsection. Picard's Iteration linearizes the p-Laplacian at each step, solving a sequence of linearized equations until convergence. To derive Picard's Iteration method, we adopt a two-step process akin to the implementation of Newton's method. Given S^n, we define $S^{n+1} := S^n + \delta S^n$ and solve $A\delta S = b$ where δS is the unknown at iteration step n.

From Eq. (2.4), we have:

$$\begin{aligned}
&\nabla \cdot \left(|\nabla S^n|^{p_m-2}\nabla(S^n + \delta S)\right) = f \\
\iff \quad &\nabla \cdot (\beta^n \nabla S^n) + \nabla \cdot (\beta^n \nabla \delta S) = f \\
\iff \quad &\nabla \cdot (\beta^n \nabla \delta S) = f - \nabla \cdot (\beta^n \nabla S^n).
\end{aligned}$$

Using the discretization discussed in Eqs. (2.1)-(2.3), we have:

$$\begin{aligned}
&\frac{1}{h^2}\Bigg[\beta^n_{i+\frac{1}{2},j,k}\left(\delta S_{i+1,j,k} - \delta S_{i,j,k}\right) + \beta^n_{i-\frac{1}{2},j,k}\left(\delta S_{i-1,j,k} - \delta S_{i,j,k}\right) + \beta^n_{i,j+\frac{1}{2},k}\left(\delta S_{i,j+1,k} - \delta S_{i,j,k}\right) + \\
&\quad \beta^n_{i,j-\frac{1}{2},k}\left(\delta S_{i,j-1,k} - \delta S_{i,j,k}\right) + \beta^n_{i,j,k+\frac{1}{2}}\left(\delta S_{i,j,k+1} - \delta S_{i,j,k}\right) + \beta^n_{i,j,k-\frac{1}{2}}\left(\delta S_{i,j,k-1} - \delta S_{i,j,k}\right)\Bigg] = f - \\
&\frac{1}{h^2}\Bigg[\beta^n_{i+\frac{1}{2},j,k}\left(S^n_{i+1,j,k} - S^n_{i,j,k}\right) + \beta^n_{i-\frac{1}{2},j,k}\left(S^n_{i-1,j,k} - S^n_{i,j,k}\right) + \beta^n_{i,j+\frac{1}{2},k}\left(S^n_{i,j+1,k} - S^n_{i,j,k}\right) + \\
&\quad \beta^n_{i,j-\frac{1}{2},k}\left(S^n_{i,j-1,k} - S^n_{i,j,k}\right) + \beta^n_{i,j,k+\frac{1}{2}}\left(S^n_{i,j,k+1} - S^n_{i,j,k}\right) + \beta^n_{i,j,k-\frac{1}{2}}\left(S^n_{i,j,k-1} - S^n_{i,j,k}\right)\Bigg].
\end{aligned}$$

We solve the system $A\delta S = b$ for δS and then add δS to S^n to get $S^{n+1} = S^n + \delta S$.

For the numerical implementation of Picard's Iteration, we utilize the biconjugate gradient (BiCG) iterative linear solver. The convergence of this scheme is controlled by two tolerance values. The first, known as the local tolerance, is used to manage the BiCG method, and we set it to a relatively large value of 10^{-1}. We tested a range of BiCG tolerances, from 10^0 to 10^{-10}, and found that a relatively loose tolerance of 10^{-1} is sufficient and efficient for the inner Picard's iteration. The second tolerance, referred to as the global tolerance, ensures that the iteration terminates when the difference between two successive solutions falls within the specified limit.

2.3. Pseudo-time methods

For the pseudo-time method, we solve the parabolic p-Laplace equation

$$\frac{\partial S}{\partial t} = \nabla \cdot \left(|\nabla S|^{p_m-2}\nabla S\right) - f \tag{2.5}$$

with an appropriate initial value until steady state. In [29], the authors noted that studies using finite difference method for the parabolic p-Laplace operator are uncommon in the literature. Their studies only considered the Cauchy problem and $p \geq 2$. Moreover,

previous studies on normalized p-Laplacian are not suitable for the parabolic case. To address this gap, we propose two numerical schemes for solving the general p-Laplace operator. The first approach we examine is the Crank-Nicolson method. The second is an Alternating Direction Implicit (ADI) method, adapted from the Crank-Nicolson scheme.

2.3.1. Crank-Nicolson (CN)

Using the Crank-Nicolson time integration at a general spatial node (x_i, y_j, z_k), we have

$$\frac{S^{n+1}-S^n}{\Delta t} = \frac{1}{2}\nabla\cdot\left(|\nabla S^n|^{p_m-2}\nabla S^n\right) + \frac{1}{2}\nabla\cdot\left(|\nabla S^n|^{p_m-2}\nabla S^{n+1}\right) - f^{n+\frac{1}{2}}$$

$$\iff \quad S^{n+1}-S^n = \frac{\Delta t}{2}\nabla\cdot\left(|\nabla S^n|^{p_m-2}\nabla S^n\right) + \frac{\Delta t}{2}\nabla\cdot\left(|\nabla S^n|^{p_m-2}\nabla S^{n+1}\right) - \Delta t f^{n+\frac{1}{2}}.$$

Rearranging, we have

$$S^{n+1} - \frac{\Delta t}{2}\nabla\cdot\left(|\nabla S^n|^{p_m-2}\nabla S^{n+1}\right) = S^n + \frac{\Delta t}{2}\nabla\cdot\left(|\nabla S^n|^{p_m-2}\nabla S^n\right) - \Delta t f^{n+\frac{1}{2}}. \tag{2.6}$$

Notice that the gradient norm at the left-hand side is evaluated at the previous time step, i.e., $|\nabla S^n|$. In dealing with the half-grid values such as $\beta^n_{i+\frac{1}{2},j,k} = \beta(x_{i+\frac{1}{2}}, y_j, z_k, t_n)$, we evaluate them at time step t_n instead of t_{n+1} because of the difficulties involved in solving 3D nonlinear systems. The consequence of such approximation is that it reduces the temporal convergence by half order. We note that for time-dependent problems, $f^{n+\frac{1}{2}}$ is calculated by taking the average of f at times t_n and t_{n+1}. However, for time-independent problems, we calculate $f^{n+\frac{1}{2}}$ based on the initial value.

Using the discretizations discussed in Eqs. (2.1)-(2.3), we have the CN scheme Eq. (2.6) given as:

$$\begin{aligned}
&\left\{1 + \frac{\Delta t}{2h^2}\left[\beta^n_{i+\frac{1}{2},j,k} + \beta^n_{i-\frac{1}{2},j,k} + \beta^n_{i,j+\frac{1}{2},k} + \beta^n_{i,j-\frac{1}{2},k} + \beta^n_{i,j,k+\frac{1}{2}} + \beta^n_{i,j,k-\frac{1}{2}}\right]\right\} S^{n+1}_{i,j,k} \\
&- \frac{\Delta t}{2h^2}\left\{\beta^n_{i+\frac{1}{2},j,k} S^{n+1}_{i+1,j,k} + \beta^n_{i-\frac{1}{2},j,k} S^{n+1}_{i-1,j,k} + \beta^n_{i,j+\frac{1}{2},k} S^{n+1}_{i,j+1,k} + \beta^n_{i,j-\frac{1}{2},k} S^{n+1}_{i,j-1,k} + \beta^n_{i,j,k+\frac{1}{2}} S^{n+1}_{i,j,k+1} + \beta^n_{i,j,k-\frac{1}{2}} S^{n+1}_{i,j,k-1}\right\} \\
&= S^n_{i,j,k} + \frac{\Delta t}{2h^2}\left[\beta^n_{i+\frac{1}{2},j,k}\left(S^n_{i+1,j,k} - S^n_{i,j,k}\right) + \beta^n_{i-\frac{1}{2},j,k}\left(S^n_{i-1,j,k} - S^n_{i,j,k}\right) + \beta^n_{i,j+\frac{1}{2},k}\left(S^n_{i,j+1,k} - S^n_{i,j,k}\right)\right. \\
&\left. + \beta^n_{i,j-\frac{1}{2},k}\left(S^n_{i,j-1,k} - S^n_{i,j,k}\right) + \beta^n_{i,j,k+\frac{1}{2}}\left(S^n_{i,j,k+1} - S^n_{i,j,k}\right) + \beta^n_{i,j,k-\frac{1}{2}}\left(S^n_{i,j,k-1} - S^n_{i,j,k}\right)\right] - \Delta t f^{n+\frac{1}{2}}.
\end{aligned}$$

In Appendix A, the unconditional stability of the Crank-Nicolson method has been proved for a 3D pseudo-time p-Laplace equation. In Section 3, we will numerically show that this method is second-order accurate in space and about 1.5-order accurate in time. Similar to Picard's method, we implement the Crank-Nicolson method using the BiCG iterative linear solver. To minimize errors, the convergence tolerance for the BiCG method is set to be a relatively small value. The convergence of the Crank-Nicolson method is determined by two criteria: the tolerance used in the BiCG method and a predefined stopping time.

2.3.2. Alternating direction implicit method (ADI)

The efficiency bottleneck of the CN method lies in the iterative solution of large linear systems in each time step. An efficient Alternating Direction Implicit (ADI) scheme is proposed in this work for solving the pseudo-time p-Laplace equation. To this end, the CN discretization is first rewritten into a divergence form

$$\frac{S^{n+1}-S^n}{\Delta t} = \left[\frac{\partial}{\partial x}\left(\beta\frac{\partial S}{\partial x}\right)\right]^{n+\frac{1}{2}} + \left[\frac{\partial}{\partial y}\left(\beta\frac{\partial S}{\partial y}\right)\right]^{n+\frac{1}{2}} + \left[\frac{\partial}{\partial z}\left(\beta\frac{\partial S}{\partial z}\right)\right]^{n+\frac{1}{2}} - f^{n+\frac{1}{2}},$$

where β is evaluated at time t_n to linearize the system. One can then apply the Douglas ADI scheme developed in [22] for the mean curvature flow to the present p-Laplacian

$$\begin{aligned}
\left(1 - \frac{\Delta t}{2}\delta_{xx}\right) S^*_{i,j,k} &= \left[1 + \frac{\Delta t}{2}\left(\delta_{xx} + 2\delta_{yy} + 2\delta_{zz}\right)\right] S^n_{i,j,k} - \Delta t f^{n+\frac{1}{2}}, \\
\left(1 - \frac{\Delta t}{2}\delta_{yy}\right) S^{**}_{i,j,k} &= S^*_{i,j,k} - \frac{\Delta t}{2}\delta_{yy} S^n_{i,j,k}, \\
\left(1 - \frac{\Delta t}{2}\delta_{zz}\right) S^{n+1}_{i,j,k} &= S^{**}_{i,j,k} - \frac{\Delta t}{2}\delta_{zz} S^n_{i,j,k}.
\end{aligned}$$

The operators δ_{xx}, δ_{yy}, and δ_{zz} are given by Eqs. (2.1), (2.2) and (2.3), respectively. As noted in [22], this method is second-order accurate in space and about 1.5-order accurate in time. The reduction in the temporal convergence order arises from approximating $\beta^{n+\frac{1}{2}}$ by β^n . The main advantage of the ADI method, as compared to the CN, is that it reduces the 3D linear system at each time

step to sets of independent one-dimensional (1D) subsystems of tridiagonal structures. Moreover, such 1D matrices can be efficiently solved by using the Thomas algorithm [30]. Since the ADI scheme can be interpreted as a high-order perturbation of the CN method [22], it preserves the property of unconditional stability, which will be numerically verified in Section 3.

For time-independent problems with analytic solutions, we begin by solving the Poisson equation $\Delta S = f$, where f is derived from the predetermined analytic solution S. The solution to the Poisson equation serves as the initial condition for the pseudo-time methods as well as Picard's method. For the purpose, we effectively set $p_m = 2$ in the nonlinear term $|\nabla S|^{p_m-2}$. We discretize this Poisson equation using central differences and numerically solve the resulting sparse linear system with the biconjugate gradient method to obtain the initial conditions necessary for solving the time-independent p-Laplace equation. This approach is consistent and can be applied to all three proposed numerical schemes in this paper. Moreover, we apply Dirichlet boundary conditions in all implementations.

It is worthwhile to note that these three algorithms are not the only approaches we have examined. We explored several other methods; however, these three were identified as the most effective ones in terms of accuracy and computational efficiency and were therefore selected for comparison and reporting. For instance, we also considered another CN method. In particular, we solve

$$\frac{\partial S}{\partial t} = |\nabla S|^{2-p_m}\left[\nabla \cdot \left(|\nabla S|^{p_m-2}\nabla S\right) - f\right] \tag{2.7}$$

until steady state using a CN scheme, which yielded results comparable to those from the two previously discussed methods, specifically Eqs. (2.4) and (2.5). However, we have chosen to omit these results due to the lower computational efficiency of this scheme compared to the first CN method proposed earlier. The reduced efficiency is primarily attributed to the cost of multiplication by the reciprocal term $|\nabla S|^{p_m-2}$.

In addition, we have examined the performance of the Explicit Euler scheme in the context of both pseudo-time methods. Although these schemes function correctly and generate results comparable to the other methods, they impose strict stability conditions that significantly affect their efficiency. In particular, the Euler scheme for pseudo-time method 1 (see Eq. (2.5)) requires a stability condition of $\Delta t = \mathcal{O}(|\nabla S|h^2)$. Meanwhile, pseudo-time method 2 (Eq. (2.7)) necessitates $\Delta t = \mathcal{O}(h^2)$. Therefore, achieving convergence within the Euler scheme requires very small time steps Δt, which results in computational inefficiencies.

3. Numerical benchmark tests

In this section, we examine the accuracy, stability, and efficiency of the proposed schemes for solving the p-Laplace equation. We will consider two time-independent problems with analytic solutions. These tests are significant as they evaluate different aspects of the proposed methods. Benchmark test 1 demonstrates that the parabolic system asymptotically converges to the corresponding elliptic case, validating the consistency of different approaches in accuracy and stability. Benchmark test 2 explores a simplified system that is close to the transition region of the diffuse interface for the Poisson-Boltzmann equation, providing some insight into the behavior of different numerical methods in this critical regime, which will be further discussed in Section 4. For simplicity, the mesh sizes in the x, y, and z directions are set to be the same $h = \Delta x = \Delta y = \Delta z$. Numerical errors in both the L_∞ and L_2 norms are reported for the convergence tests in each example. Unless otherwise specified, we set $p_m = 1.2$ for all the tests in this section. The conclusions are the same for different $p_m \in [1, 2)$. We compute L_∞ and L_2 errors as

$$L_\infty = \max_{i,j,k} |S_{i,j,k} - (S_h)_{i,j,k}| \qquad \text{and} \qquad L_2 = \sqrt{\frac{\sum_{i,j,k} |S_{i,j,k} - (S_h)_{i,j,k}|^2}{N_2 + 1}} \tag{3.1}$$

respectively, where S_h is the numerical solution and $N_2 = N_x \times N_y \times N_z$.

All the experiments were performed on a single-core Dell PowerEdge R920 at The University of Alabama High-Performance Computing (UAHPC) facility (https://oit.ua.edu/services/research/), equipped with an Intel® Xeon® CPU E7-8891 v2 operating at 3.20 GHz clock speed.

3.1. Benchmark test 1: a time-independent analytical function

This test aims to demonstrate that the solution obtained from the pseudo-time methods converges to that of the p-Laplace equation. Specifically, we seek to show that the results from the ADI and CN schemes align with those obtained from Picard's Iteration. To this end, we consider the following function on the domain $[0, 2\pi]^3$:

$$S(x, y, z) = \sin(x)\sin(y)\sin(z). \tag{3.2}$$

By applying the p-Laplace operator with the regularization parameter η, the source term is derived as:

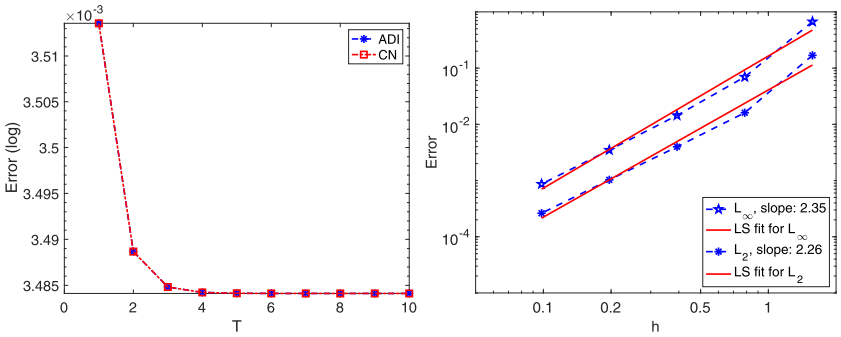
$$\begin{aligned} f(x, y, z; \eta) = -|\nabla S + \eta|^{p_m-4}\Big\{ & \sin(x)\sin(y)\sin(z)\Big[3\eta \\ & + (p_m+1)\Big(\big(\cos(x)\sin(y)\sin(z)\big)^2 + \big(\sin(x)\cos(y)\sin(z)\big)^2 + \big(\sin(x)\sin(y)\cos(z)\big)^2\Big) \\ & - 2(p_m-2)\Big(\big(\cos(x)\cos(y)\sin(z)\big)^2 + \big(\cos(x)\sin(y)\cos(z)\big)^2 + \big(\sin(x)\cos(y)\cos(z)\big)^2\Big)\Big]\Big\}. \end{aligned} \tag{3.3}$$

Table 1
Spatial accuracy of ADI, and CN for time-independent sine function.

h	ADI Scheme				CN Scheme			
	L_∞		L_2		L_∞		L_2	
	Error	Order	Error	Order	Error	Order	Error	Order
$\frac{\pi}{2}$	$6.69E-01$		$1.69E-01$		$6.69E-01$		$1.69E-01$	
$\frac{\pi}{4}$	$6.96E-02$	3.27	$1.61E-02$	3.39	$6.96E-02$	3.27	$1.61E-02$	3.39
$\frac{\pi}{8}$	$1.44E-02$	2.28	$3.98E-03$	2.01	$1.44E-02$	2.28	$3.98E-03$	2.01
$\frac{\pi}{16}$	$3.48E-03$	2.04	$1.03E-03$	1.95	$3.48E-03$	2.04	$1.03E-03$	1.95
$\frac{\pi}{32}$	$8.65E-04$	2.01	$2.62E-04$	1.97	$8.65E-04$	2.01	$2.62E-04$	1.97

Table 2
Spatial Convergence of the picard's iteration.

Iter	h	L_∞		L_2	
		Error	Order	Error	Order
25	$\frac{\pi}{2}$	$6.69128E-01$		$1.68604E-01$	
46	$\frac{\pi}{4}$	$6.95640E-02$	3.27	$1.60551E-02$	3.39
102	$\frac{\pi}{8}$	$1.43565E-02$	2.28	$3.97990E-03$	2.01
188	$\frac{\pi}{16}$	$3.48412E-03$	2.04	$1.02792E-03$	1.95
329	$\frac{\pi}{32}$	$8.65052E-04$	2.01	$2.62108E-04$	1.97



(a) Steady State Convergence (b) Spatial Convergence

Fig. 1. Convergence of the time-independent sine function.

In our numerical experiments for this function, we set the regularization parameter to $\eta = 1$. As stated before, convergence in Picard's iteration is controlled by two tolerance values: the first governs the convergence of the BiCG method at each iterative step, and the second determines when the overall scheme terminates, indicating convergence. We set the first tolerance to 10^{-1} and the second to 10^{-10}. For the Crank-Nicolson (CN) scheme, the simulation proceeds until a predefined stopping time. At each time step, the linear system generated by the CN scheme is solved iteratively with the BiCG method to a tolerance of 10^{-5}. Similarly, for the Alternating Direction Implicit (ADI) method, the scheme is advanced at each time step until the stopping time is reached.

We first demonstrate the steady-state convergence for the pseudo-time schemes using time step $\Delta t = 0.01$ and a mesh size of $h = \frac{\pi}{16}$. The results are presented in Fig. 1(a). We observe that both pseudo-time methods reach convergence at a stopping time of $T = 7$. This shows that the solution of the parabolic equation converges to that of the corresponding elliptic case.

Next, we analyze spatial accuracy while keeping the temporal parameters fixed. Specifically, we set the final time $T = 10$ and choose a time step of $\Delta t = 0.0005$ for both the ADI and CN schemes, resulting in a total of $20,000$ time steps. The results indicate that the ADI and CN schemes align with those obtained from Picard's Iteration. Moreover, the results shown in Tables 1 and 2 confirm that all methods achieve second-order accuracy in space. This is further illustrated in Fig. 1(b), where we plot the spatial accuracy results for the three schemes. It can be seen that when h is small, both methods exhibit second-order accuracy in space. We also note certain super-convergence in the first mesh refinement, i.e., the numerically detected rate is higher than two, approaching three. This may be because the errors of the first level are based on $h = \frac{\pi}{2} > 1$ and are thus very large. Consequently, a higher convergence rate is observed when h is reduced to $\frac{\pi}{4}$. Numerically, the detected order agrees with the theoretical one, only when a small enough h is used. Similar super-convergence can also be observed in the other spatial convergence tests in the rest of this paper.

Moreover, with $T = 10$ and $N = 97$ (corresponding to $h = \frac{\pi}{48}$), we analyze temporal convergence. The results shown in Fig. 2 demonstrate that both the ADI and CN schemes achieve excellent convergence even with a relatively large time step.

To determine the most effective proposed schemes balancing accuracy and efficiency, we examine the runtime of various algorithms with fixed time steps and mesh sizes. For this analysis, we set $\eta = 1$ and use a time step $\Delta t = 0.5$ for both the ADI and CN

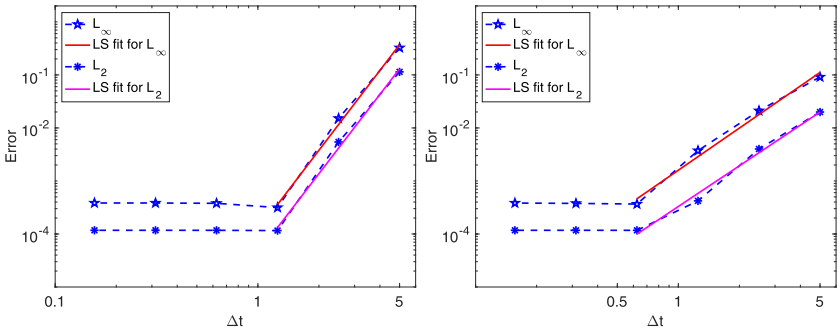


Fig. 2. Temporal Convergence of the time-independent sine function.

Table 3
CPU times of the proposed schemes for time-independent sine function with fixed mesh size and time step.

h	Δt	Scheme	CPU (s)	L_∞ Error	Relative CPU	Relative Error
		PI	0.8149	$3.4841E-03$		
$\frac{\pi}{16}$	0.5	ADI	0.3730	$3.4834E-03$	45.77 %	$1.6360E-04$
	0.5	CN	1.0629	$3.4760E-03$	130.43 %	$2.3277E-03$
		PI	10.4824	$8.6505E-04$		
$\frac{\pi}{32}$	0.5	ADI	4.3342	$8.6486E-04$	41.35 %	$2.2658E-04$
	0.5	CN	9.3854	$8.6668E-04$	89.53 %	$1.8820E-03$
		PI	53.0347	$3.8398E-04$		
$\frac{\pi}{48}$	0.5	ADI	14.7172	$3.8390E-04$	27.75 %	$2.3178E-04$
	0.5	CN	32.3314	$3.8912E-04$	60.96 %	$1.3378E-02$

methods. Using PI as a benchmark, we calculate the relative time savings for each scheme as shown in Table 3. As a result, the ADI method requires the least CPU time while still producing similarly accurate results. As a high-order perturbation of the CN method in time discretization, the ADI method is unconditionally stable for parabolic problems. The major advantage of the ADI method, as compared to CN or other implicit methods, is that it reduces the 3D linear system at each time step to sets of independent one-dimensional (1D) subsystems of tridiagonal structures, and such matrices can be efficiently solved by using the Thomas algorithm. Such features contribute to the high efficiency of the ADI method for solving the p-Laplace equation.

Finally, we examine the sensitivity of the numerical solution to the regularization parameter η by considering the ADI method. To this end, a new right-hand side source term $\tilde{f}(x, y, z)$ is generated by applying the p-Laplace operator without the regularization parameter η to the exact solution $S(x, y, z)$ given by Eq. (3.2). In particular, we use $||\nabla S||$ rather than $||\nabla S + \eta||$ in deriving $\tilde{f}(x, y, z)$. Note that $\tilde{f}(x, y, z)$ is different from the previous source $f(x, y, z; \eta)$ given by Eq. (3.3). The same ADI scheme is used to solve the system – its left-hand side matrix includes η to prevent singularities. Consequently, the exact solution $\tilde{S}(x, y, z)$ to this new test problem is different from $S(x, y, z)$ of Eq. (3.2). They become identical only when η goes to zero, i.e., $\lim_{\eta \to 0} \|\tilde{S}(x, y, z) - S(x, y, z)\| = 0$.

Computationally, since the analytical expression of $\tilde{S}(x, y, z)$ is unknown, we will validate the numerical approximation to $\tilde{S}(x, y, z)$ by comparing it with $S(x, y, z)$. We set $T = 10$, $\Delta t = 0.001$, and a tolerance 10^{-7} for the BiCG method when solving the Poisson equation to initialize the solution. The numerical differences by using different η values are reported in Table 4. It can be seen that when η is large, e.g. $\eta = 1$, the difference $\tilde{S}(x, y, z) - S(x, y, z)$ is large, and the norms remain constant as h goes to zero. On the other hand, when $\eta = 10^{-10}$, the differences $\tilde{S}(x, y, z) - S(x, y, z)$ are vanishing such that the difference becomes smaller when the mesh is refined in Table 4. Moreover, one can see that the numerical differences of $\eta = 10^{-7}$ are essentially the same as those of $\eta = 10^{-10}$. This suggests $\eta = 10^{-7}$ could be an ideal choice for the regularization parameter, which is large enough to avoid singularities, while producing similar results as $\eta = 10^{-10}$ for the tested h values. Indeed, Tian and Zhao in [22] demonstrated that for the Minimal Molecular Surface (MMS), the difference between the solutions for $\eta = 10^{-7}$ and $\eta = 10^{-10}$ is as small as 10^{-4}.

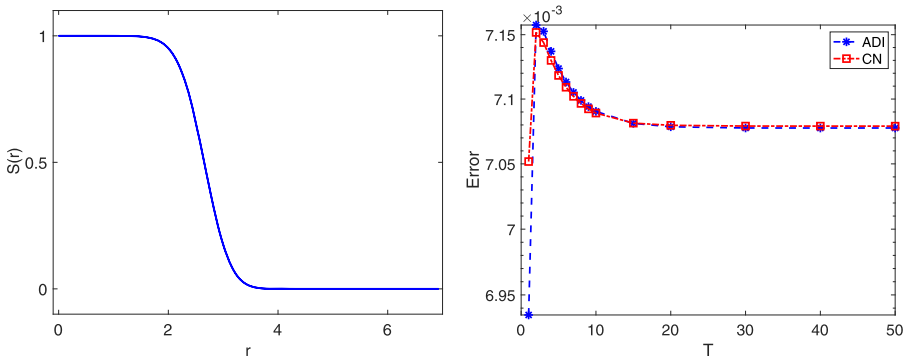
3.2. Benchmark test 2: A function with sharp transition region

This test aims to demonstrate that the proposed schemes are still effective in solving systems that closely resemble the sharp transition region for a diffuse interface problem. To that end, we consider a smooth function that behaves similarly to our target functions of diffuse interface profile with sharp changes around the interface transition region. In particular, we consider the analytic function given by:

$$S(x, y, z) = S(r) = a + b\text{sech}(\text{cr}^{1+\alpha}),$$

Table 4
Numerical difference between the time-independent sine function $S(x, y, z)$ and the ADI approximations of $\tilde{S}(x, y, z)$ by using different η values.

	$\eta = 10^{0}$		$\eta = 10^{-4}$		$\eta = 10^{-7}$		$\eta = 10^{-10}$	
h	L_∞	L_2	L_∞	L_2	L_∞	L_2	L_∞	L_2
$\pi/4$	1.00E+00	2.52E-01	1.00E+00	2.52E-01	1.00E+00	2.52E-01	1.00E+00	2.52E-01
$\pi/8$	8.41E-01	2.26E-01	6.93E-01	1.58E-01	6.93E-01	1.58E-01	6.93E-01	1.58E-01
$\pi/16$	8.20E-01	2.48E-01	3.20E-01	5.82E-02	3.20E-01	5.78E-02	3.20E-01	5.78E-02
$\pi/32$	8.18E-01	2.60E-01	1.15E-01	1.58E-02	1.13E-01	1.52E-02	1.13E-01	1.52E-02
$\pi/32$	8.17E-01	2.67E-01	3.99E-02	4.47E-03	3.56E-02	3.59E-03	3.57E-02	3.59E-03



(a) S(r) (b) Steady state convergence

Fig. 3. (a) A visual representation of $S(r)$ based on the sech function. (b) A time history plot showing the steady-state convergence for the pseudo-time methods.

where $r = \sqrt{x^2 + y^2 + z^2}$. Setting $\alpha = 4$, $a = 0$, $b = 1$, and $c = 0.01$ ensures that the function is integrable. With these parameter choices and a domain $[-4, 4]^3$, the visual representation of S(r) is shown in Fig. 3(a). Such an S function characterizes the domain into three subdomains, a solute subdomain with $S \approx 1$, a solvent region with $S \approx 0$, and a transition band as the smooth solute-solvent boundary. We obtain the source term to be given by:

$$f(x, y, z; \eta) = (S_r^2 + \eta)^{\frac{p_m - 2}{2}} \left[\frac{2}{r} S_r + S_{rr} + (p_m - 2) \frac{S_r^2 S_{rr}}{S_r^2 + \eta} \right], \tag{3.4}$$

where

$$S_r = -bc(1+\alpha) r^{\alpha} \operatorname{sech}(\mathrm{cr}^{\alpha+1}) \tanh(\mathrm{cr}^{\alpha+1})$$

$$S_{rr} = -\alpha bc(1+\alpha) r^{\alpha-1} \operatorname{sech}(\mathrm{cr}^{\alpha+1}) \tanh(\mathrm{cr}^{\alpha+1}) + \mathrm{bc}^2(1+\alpha)^2 \mathrm{r}^{2\alpha} \operatorname{sech}(\mathrm{cr}^{\alpha+1}) \left(\tanh^2(\mathrm{cr}^{\alpha+1}) - \operatorname{sech}^2(\mathrm{cr}^{\alpha+1}) \right).$$

As mentioned earlier, for the initial condition, we solve the Poisson equation $\Delta S = f$ with $p_m = 2$ and use the solution as the initial condition for the p-Laplace equation. The initial condition is computed using the biconjugate gradient iterative algorithm with a tolerance of 10^{-5}. The regularization parameter is fixed at $\eta = 1$. For PI, the convergence is controlled by two tolerance levels: 10^{-1} for local convergence and 10^{-10} for global convergence. In the CN method, the iterative solver at each time step uses a tolerance of 10^{-5}.

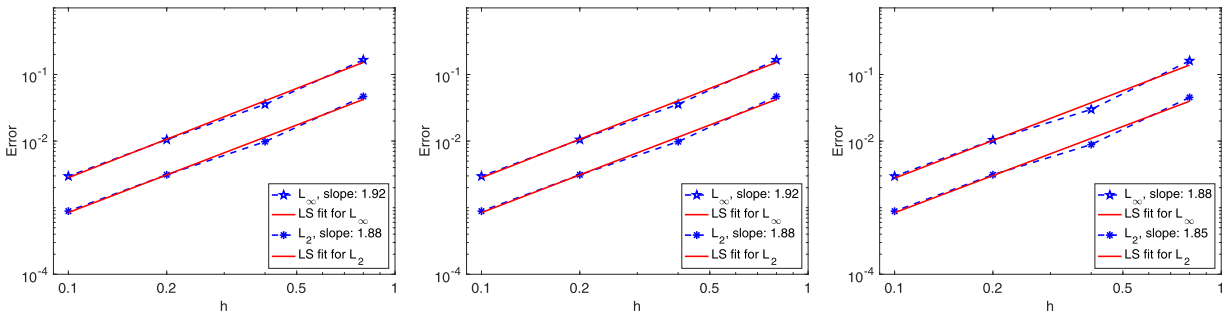
Here we carry out a series of similar experiments to study and validate the proposed numerical schemes for the new function involving fast changes around the transition region. First, we examine steady-state convergence for the pseudo-time schemes, using a time step size of $\Delta t = 0.01$ and $N = 50$. The results are shown in Fig. 3(b). At a stopping time of $T = 25$ which is relatively longer than that required for the sine function test, we observe that both pseudo-time methods (ADI and CN) are guaranteed to reach convergence.

Secondly, we demonstrate the spatial convergence and accuracy. For the study, the stopping time is fixed at $T = 50$ and a time step of $\Delta t = 0.05$ is used for both the ADI and CN schemes. From Fig. 4, it is observed that all three reported methods exhibit second-order accuracy in space.

Thirdly, we evaluate the computational efficiency of the algorithms by measuring the clock time required for their execution, given a fixed time step Δt and mesh size h. The CPU time and numerical solutions obtained from PI are used as references to compare both the ADI and CN schemes. Once again, the results in Table 5 demonstrate that the ADI scheme is the most efficient of the three, as it requires the least amount of time to produce outcomes with comparable accuracy.

Finally, we examine the dependence and sensitivity of the solution on the choice of η. This test is also conducted using the ADI scheme. As in the previous test considered in Table 4, the source term is modified by excluding η

$$\tilde{f}(x, y, z) = \left(S_r^2\right)^{\frac{p_m - 2}{2}} \left[\frac{2}{r} S_r + (p_m - 1) S_{rr} \right].$$



(a) PI (b) ADI (c) CN

Fig. 4. Spatial convergence results of proposed numerical schemes for time -independent sech function.

Table 5
CPU times of the proposed schemes to compute sech function with fixed mesh size and time step.

N	Δt	Scheme	CPU (s)	L_∞ Error	Relative CPU	Relative Error
		PI	0.3588	$2.1698E-02$		
20	1	ADI	0.2396	$2.1698E-02$	66.78 %	$2.30435E-05$
	1	CN	0.5406	$2.1698E-02$	150.69 %	$0.0000E+00$
		PI	6.0211	$5.6240E-03$		
40	1	ADI	2.8263	$5.6208E-03$	46.94 %	$5.7077E-04$
	1	CN	6.4319	$5.6213E-03$	106.82 %	$4.6942E-04$
		PI	84.0462	$1.3708E-03$		
80	1	ADI	30.5995	$1.3696E-03$	36.41 %	$8.6812E-04$
	0.5	CN	79.5750	$1.3685E-03$	94.68 %	$1.6560E-03$

Table 6
Numerical differences between the time-independent sech function $S(x, y, z)$ and the ADI approximations of $\tilde{S}(x, y, z)$ by using different η values.

	$\eta = 10^0$		$\eta = 10^{-4}$		$\eta = 10^{-7}$		$\eta = 10^{-10}$	
N	L_∞	L_2	L_∞	L_2	L_∞	L_2	L_∞	L_2
10	1.94E+00	5.34E-01	9.47E-02	2.49E-02	9.66E-02	2.54E-02	9.66E-02	2.54E-02
20	1.96E+00	5.68E-01	3.12E-02	9.57E-03	3.25E-02	9.24E-03	3.25E-02	9.24E-03
40	1.97E+00	5.90E-01	1.51E-02	5.05E-03	1.06E-02	3.15E-03	1.06E-02	3.13E-03
80	1.97E+00	6.01E-01	1.35E-02	3.92E-03	3.09E-03	9.29E-04	3.00E-03	8.96E-04

Table 7
Spatial convergence of the ADI scheme for the time-independent sech function by using different η values.

	$\eta = 10^{-4}$				$\eta = 10^{-7}$			
N	L_∞		L_2		L_∞		L_2	
	Error	Order	Error	Order	Error	Order	Error	Order
10	$1.37E-01$		$3.31E-02$		$1.66E-01$		$4.68E-02$	
20	$4.71E-02$	1.54	$1.28E-02$	1.38	$3.57E-02$	2.22	$9.80E-03$	2.26
40	$1.01E-02$	2.22	$2.85E-03$	2.17	$1.05E-02$	1.76	$3.12E-03$	1.65
80	$2.62E-03$	1.94	$7.91E-04$	1.85	$2.97E-03$	1.82	$8.87E-04$	1.81

The left-hand side matrix still contains η in the ADI algorithm. Again, denote the unknown analytical solution to the modified problem as $\tilde{S}(x, y, z)$. The numerical differences reported in Table 6 follow the same pattern as those in Table 4. When $\eta = 1$, the differences remain constant, while when $\eta = 10^{-10}$, the differences are vanishing. Again, the results demonstrate that selecting $\eta = 10^{-7}$ is sufficient both to prevent singularities and to ensure convergence to the desired results.

To further examine the impact of η on numerical convergence, we consider the original source $f(x, y, z; \eta)$ with η included. In Fig. 4, we see that the numerical order of the ADI scheme is two when $\eta = 1$. In Table 7, we examine the spatial orders for $\eta = 10^{-4}$ and $\eta = 10^{-7}$. It can be seen that the numerical convergence rates are slightly smaller than those in Fig. 4. But the ADI method still yields the second order convergence in space.

4. Biological applications

In this section, we present numerical experiments to evaluate the performance of the proposed schemes in solving real biological systems and assess their effectiveness in electrostatic analysis. Specifically, we aim to investigate the following: 1. Whether the solution of the pseudo-time parabolic equation converges to that of the original elliptic p-Laplacian operator in the context of solute-solvent interface profile and solvation energy calculations; 2. Whether the ADI scheme remains the optimal choice for protein energy calculations; 3. Whether solvation energies converge as p_m approaches one.

To address these questions, we first describe the setup of the initial and boundary conditions for protein simulations. The methodology is validated using a simple diatomic system to compute solvation energy-related non-electrostatic quantities, such as volume and surface area. Subsequently, we detail the solution of the Poisson-Boltzmann equation (PB) for electrostatic analysis and outline the approach used to calculate electrostatic energies. Finally, a set of protein systems is analyzed to demonstrate the overall performance and robustness of the proposed numerical schemes.

4.1. Initial and boundary conditions

Given a protein consisting of N_m atoms, we represent the atomic centers as $\mathbf{r}_i = (x_i, y_i, z_i)$, for $i = 1, \dots, N_m$, and denote the radius of the i-th atom as r_i. To ensure that the molecular domain Ω_m is adequately contained, we consider an enlarged domain Ω with a boundary $\partial\Omega$ sufficiently far from Ω_m, such that S is zero on $\partial\Omega$. Following [21,22,31], the initial value of S is set as a piecewise constant.

The domain enclosed by the solvent-accessible surface (SAS) is defined as:

$$D = \bigcup_{i=1}^{N_m} \left\{ \mathbf{r} : |\mathbf{r} - \mathbf{r}_i| < r_i + r_p \right\},$$

where r_p represents the probe radius. We solve for the values of $S(x, y, z, t)$ at node points located between the SAS and the Van der Waals (VdW) surface, which is defined similarly as the SAS with $r_p = 0$. The initial value of S is defined as:

$$S(x, y, z, 0) = \begin{cases} 1 & \text{if } (x, y, z) \in D, \\ 0 & \text{otherwise.} \end{cases}$$

To efficiently compute the solute-solvent surface, the Van der Waals spheres are protected during computation by introducing a control function at the preprocessing stage. This control function is applied at every time step. For further details, we refer readers to [21,31]. After sufficient time T, the solution typically reaches a steady state, $S(x, y, z, T)$.

The steady-state solution $S(x, y, z, T)$, obtained by solving the p-Laplace equation, serves as a diffuse interface profile [22]. Additionally, it produces a set of level surfaces, one of which can be extracted as an isosurface $S(x, y, z, T) = C$ to represent a sharp molecular surface. However, the numerical solution $S(x, y, z, T)$ often exhibits artifacts due to finite difference approximations on Cartesian grids. To mitigate these artifacts, we further integrate the p-Laplace equation for a short time duration ($t = 0.01$) without imposing the VdW constraints. This brief integration improves the smoothness of the solute-solvent surface while preserving its overall shape.

4.2. Numerical tests on diatomic system

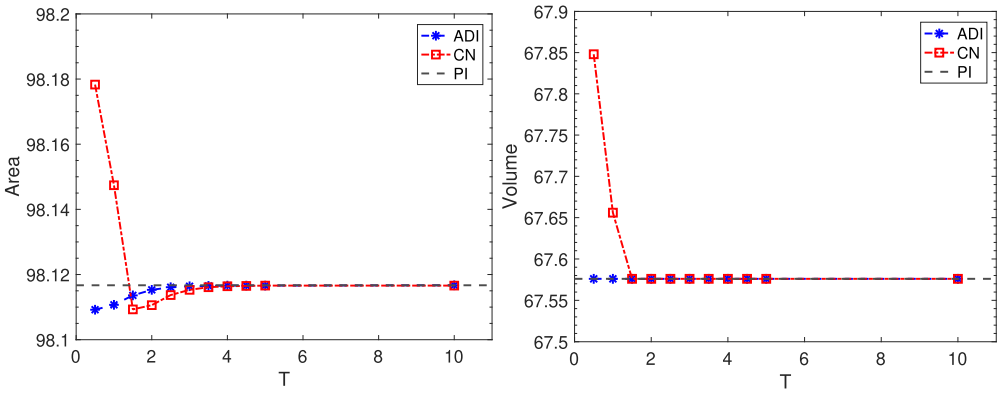
In this subsection, we consider a diatomic system with two atoms of equal radius, $r_1 = r_2 = 2\,\text{Å}$. The atomic centers are located at $\mathbf{r}_1 = (-2.2, 0, 0)$ and $\mathbf{r}_2 = (2.2, 0, 0)$. The computational domain is defined as $(x, y, z) \in [-6, 6] \times [-4, 4] \times [-4, 4]$. To ensure the uniqueness of the steady-state solution, we impose the same initial and boundary conditions for both the Crank-Nicolson (CN) and Alternating Direction Implicit (ADI) schemes when solving the p-Laplace equation. Similarly, these initial and boundary conditions are also applied in Picard's iteration.

For real biological systems, solvation energy-related quantities of interest, such as volume, area, and electrostatic potential, are calculated for various tests and comparisons. In this subsection, a hypersurface is generated to compute the non-electrostatic quantities (volume and area) by varying the value of C within the range $[0.97, 0.99]$. For details on the computation of surface area and volume integrals, the reader is referred to [22] and the references therein.

Here, the hypersurface is specifically defined by $S(x, y, z, t) = 0.98$. The stopping time is set to $T = 2$, and the regularization parameter is fixed at $\eta = 10^{-7}$. For Picard's iteration, a tolerance of 10^{-1} is used for local convergence (biconjugate gradient, BiCG), and 10^{-10} is used for global convergence. In the Crank-Nicolson scheme, the BiCG tolerance is set to 10^{-5} for all tests. A probe radius of $r_p = 1.5\,\text{Å}$ is used. The results are presented for $p_m = 1.2$.

The first experiment examines steady-state convergence to verify that the original solution of the elliptic p-Laplace equation is accurately recovered by the steady-state solution of the parabolic equations when solving for the hypersurface. For this test, we set the time step to $\Delta t = 0.01$ and the mesh size to $h = 0.2\,\text{Å}$. We report the area and volume derived from the steady-state solution. As shown in Fig. 5, both the area and volume converge to the values obtained from the solution generated by Picard's method around $T = 2$.

Next, we assess the computational efficiency of the various schemes by measuring the time required for each to execute. Using Picard's Iteration (PI) as the benchmark, we compute the relative time savings for the ADI and CN schemes. The observations from

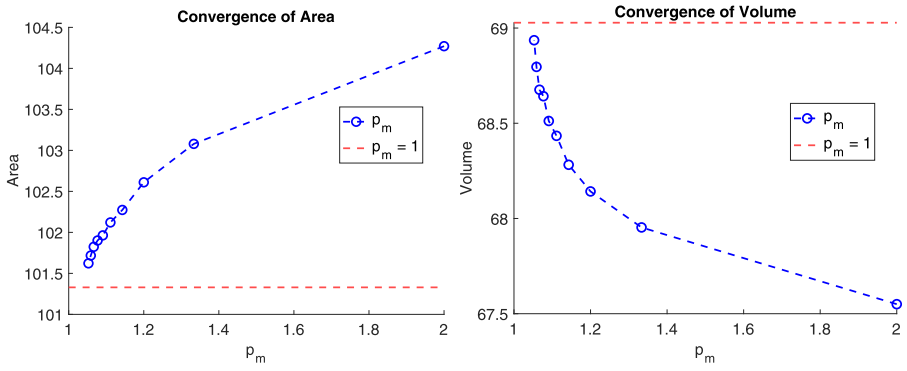


(a) Area (b) Volume

Fig. 5. The time evolution histories of area and volume for the diatom system.

Table 8
CPU times of the proposed schemes for the diatom system for fixed mesh size and time step.

h	Δt	Scheme	CPU (s)	Area	Relative CPU	Relative Area
		PI	7.1501	101.1903		
0.2	0.1	ADI	0.6605	101.2360	9.24 %	$4.5189E-04$
	0.1	CN	3.1463	101.1286	44.00 %	$6.0913E-04$
		PI	94.6258	102.6417		
0.1	0.1	ADI	5.9744	102.9743	6.31 %	$3.2412E-03$
	0.1	CN	47.2487	102.5532	49.93 %	$8.6215E-04$
		PI	1262.2621	102.7640		
0.05	0.02	ADI	225.3323	102.7366	17.85 %	$2.6743E-04$
	0.05	CN	1037.0249	102.7049	82.16 %	$5.7587E-04$



(a) Area (b) Volume

Fig. 6. p_m Convergence the diatom system.

Table 8 are consistent with those in Table 5, highlighting that, across various mesh sizes, the ADI scheme consistently outperforms in terms of both computational speed and accuracy.

The final test for this two-atom system examines convergence with respect to p_m. The goal is to demonstrate that as m increases (and p_m approaches 1), the solution converges to the one obtained for $p_m = 1$. Using the ADI scheme, we vary m from 1 to 10, with $p_m = \frac{2m}{2m-1}$. For these tests, the mesh size is fixed at $h = 0.1$ Å, and the time step is set to $\Delta t = 0.001$. Picard's iteration with $p_m = 1$ is used as the benchmark, and the simulations are performed with $\eta = 10^{-7}$. For $p_m = 1$, the area from Picard's Iteration (the reference solution) is 101.3294 and the volume is 69.028. It is observed from Fig. 6 that as p_m approaches 1, both the area and volume converge to the reference solution.

For this diatomic system, we also present visualizations of the molecular surfaces generated using the ADI scheme for various values of the parameter p_m. The visualizations were created using MATLAB; however, similar results can be achieved with other

(a) $p_m = 2.00$ (b) $p_m = 1.33$ (c) $p_m = 1.20$ (d) $p_m = 1.00$

Fig. 7. Comparison of isosurface generated for different p_m values for diatom system with isovalue of 0.97 and probe radius r_p = 1.5Å.

software tools, such as VMD. As illustrated in Fig. 7, the surface structures show noticeable differences depending on the choice of p_m. The convergence analysis in the previous section demonstrated that, as p_m approaches 1, the area and volume of the molecular surface converge to the reference values. The subtle variations in the molecular shape, as revealed by the visualizations in Fig. 7, likely contribute to the observed convergence in area and volume.

4.3. Poisson-Boltzmann equation (PBE) and electrostatic free energy

The Poisson-Boltzmann Equation (PBE) is a fundamental tool used to study the electrostatic interactions of solute macromolecules immersed in an aqueous solution. We consider a large domain $\Omega \subset \mathbb{R}^3$ that contains the solute molecule. This domain is subdivided into three regions: Ω_i: the **interior domain** occupied by the solute molecule, Ω_e: the **exterior domain** filled with the solvent, and Ω_i**: a transition layer** that serves as a smooth interface between Ω_i and Ω_e.

The transition layer Ω_t ensures a smooth change in the properties between the solute and solvent regions, avoiding sharp discontinuities. We denote the interface between the interior domain Ω_i and the transition layer Ω_t by Γ_i, while the interface between the transition layer Ω_t and the exterior domain Ω_e is denoted by Γ_e. The boundary of the entire domain Ω is referred to as $\partial\Omega$. To characterize the subdomains, we introduce a dimensionless smooth interface function $S(\mathbf{r})$. This function takes the value $S(\mathbf{r}) = 1$ inside Ω_i and $S(\mathbf{r}) = 0$ in Ω_e. Within the transition region Ω_t, $S(\mathbf{r})$ smoothly decays from 1 to 0 as $\mathbf{r}$ moves from the interior domain Ω_i to the exterior domain Ω_e. Based on this interface function, we define a smooth dimensionless dielectric function $\epsilon(\mathbf{r})$ over the entire domain Ω, which varies smoothly across the transition layer. This smooth variation of $\epsilon(\mathbf{r})$ ensures that the dielectric properties change continuously between the solute and solvent regions, which is crucial for achieving stable numerical solutions. For a detailed derivation of the dielectric function using $S(\mathbf{r})$, we refer the reader to [32–34].

For any point $\mathbf{r} \in \Omega$, the electrostatic interaction is governed by the nonlinear Poisson-Boltzmann equation in dimensionless form [33–35]:

$$-\nabla \cdot (\epsilon(\mathbf{r})\nabla u(\mathbf{r})) + (1 - S(\mathbf{r}))\kappa^2 \sinh(u(\mathbf{r})) = \rho(\mathbf{r}), \quad \mathbf{r} \in \Omega, \tag{4.1}$$

where $u(\mathbf{r}) = \frac{e_c \phi}{k_B T}$ is the dimensionless electrostatic potential. The singular source term $\rho(\mathbf{r})$ is defined as:

$$\rho(\mathbf{r}) = 4\pi \frac{e_c^2}{k_B T} \sum_{j=1}^{N_m} q_j \delta(\mathbf{r} - \mathbf{r}_j), \quad \mathbf{r} \in \Omega. \tag{4.2}$$

Here, e_c is the elementary charge, q_j is the partial charge of the jth atom located at position $\mathbf{r}_j$, and $\delta(\mathbf{r} - \mathbf{r}_j)$ is the Dirac delta function centered at $\mathbf{r}_j$. The Boltzmann constant is denoted by k_B, T represents the absolute temperature, and N_m is the total number of atoms in the solute molecule. The modified Debye-Hückel parameter κ is defined as $\kappa^2 = 8.486902807\,\text{Å}^{-2} I_s$, where I_s is the ionic strength of the solvent.

If we denote the dimensionless potential in the vacuum by v and the associated dimensionless dielectric function as ϵ_v, the Poisson equation governing the vacuum state can be derived as follows [33–35]:

$$-\nabla \cdot (\epsilon_v(\mathbf{r})\nabla v(\mathbf{r})) = \rho(\mathbf{r}), \quad \mathbf{r} \in \Omega. \tag{4.3}$$

We assume a sufficiently large computational domain Ω such that the dielectric function $\epsilon(\mathbf{r})$ becomes a constant ϵ_{out} on the outer boundary $\partial\Omega$. On this boundary, we impose a Dirichlet condition given by [36]:

$$b(\mathbf{r}; \epsilon_{out}) := \frac{e_c^2}{k_B T} \sum_{j=1}^{N_m} \frac{q_j e^{-|\mathbf{r}-\mathbf{r}_j|\sqrt{\frac{\kappa^2}{\epsilon_{out}}}}}{\epsilon_{out}|\mathbf{r} - \mathbf{r}_j|}, \quad \text{for } \mathbf{r} \in \partial\Omega. \tag{4.4}$$

In this work, we focus on the linearized version of the Poisson-Boltzmann Equation (LPBE). By approximating the hyperbolic sine function $\sinh(u)$ in (4.1) with u, the nonlinear equation simplifies to:

$$-\nabla \cdot (\epsilon(\mathbf{r})\nabla u(\mathbf{r})) + (1 - S(\mathbf{r}))\kappa^2 u(\mathbf{r}) = \rho(\mathbf{r}), \quad \mathbf{r} \in \Omega. \tag{4.5}$$

The electrostatic free energy quantifies the difference in polar solvation energy of a macromolecule when transitioning from a vacuum phase to an aqueous environment [35,37,38]. Once the potentials u and v are obtained, the electrostatic free energy can be

calculated using the definition $\Delta E = E[u] - E[v]$. The energy functional $E[u]$ is derived in [35] as:

$$E[u] = \frac{k_B T}{e_c}\frac{1}{2}\int_\Omega \sum_{j=1}^{N_m} q_j e_c \delta(\mathbf{r}-\mathbf{r}_j)u(\mathbf{r})\,d\mathbf{r} - \frac{1}{4\pi}\frac{(k_B T)^2}{e_c^2}\int_\Omega (1-S(\mathbf{r}))\kappa^2(\cosh(u(\mathbf{r}))-1)\,d\mathbf{r}$$
$$+ \frac{1}{8\pi}\frac{(k_B T)^2}{e_c^2}\int_\Omega (1-S(\mathbf{r}))\kappa^2\, u(\mathbf{r})\sinh(u(\mathbf{r}))\,d\mathbf{r} - \frac{1}{8\pi}\frac{(k_B T)^2}{e_c^2}\int_{\partial\Omega} \epsilon u\frac{\partial u}{\partial n}\,ds. \tag{4.6}$$

The corresponding energy functional $E[v]$ in the vacuum state is given by:

$$E[v] = \frac{k_B T}{e_c}\frac{1}{2}\int_\Omega \sum_{j=1}^{N_m} q_j e_c \delta(\mathbf{r}-\mathbf{r}_j)v(\mathbf{r})\,d\mathbf{r} - \frac{1}{8\pi}\frac{(k_B T)^2}{e_c^2}\int_{\partial\Omega} \epsilon_v v\frac{\partial v}{\partial n}\,ds. \tag{4.7}$$

The electrostatic free energy difference ΔE for the nonlinear Poisson-Boltzmann (NPB) model is then:

$$\Delta E = \frac{1}{2}k_B T\sum_{j=1}^{N_m} q_j(u(\mathbf{r}_j)-v(\mathbf{r}_j)) - \frac{1}{4\pi}\frac{(k_B T)^2}{e_c^2}\int_{\Omega_i^c}(1-S(\mathbf{r}))\kappa^2(\cosh(u(\mathbf{r}))-1)\,d\mathbf{r}$$
$$+ \frac{1}{8\pi}\frac{(k_B T)^2}{e_c^2}\int_{\Omega_i^c}(1-S(\mathbf{r}))\kappa^2\, u(\mathbf{r})\sinh(u(\mathbf{r}))\,d\mathbf{r} - \frac{1}{8\pi}\frac{(k_B T)^2}{e_c^2}\int_{\partial\Omega}\left(\epsilon u\frac{\partial u}{\partial n} - \epsilon_v v\frac{\partial v}{\partial n}\right)ds. \tag{4.8}$$

For the linearized Poisson-Boltzmann Equation, the electrostatic free energy simplifies to:

$$\Delta E = \frac{1}{2}k_B T\sum_{j=1}^{N_m} q_j(u(\mathbf{r}_j)-v(\mathbf{r}_j)) - \frac{1}{8\pi}\frac{(k_B T)^2}{e_c^2}\int_{\partial\Omega}\left(\epsilon u\frac{\partial u}{\partial n} - \epsilon_v v\frac{\partial v}{\partial n}\right)ds.$$

As noted in [35], the inclusion of the surface integral term does not influence the calculated electrostatic free energy (EFE) too much. By dropping such a term, the EFE for the linearized PBE is computed as:

$$\Delta E = \frac{1}{2}k_B T\sum_{j=1}^{N_m} q_j(u(\mathbf{r}_j)-v(\mathbf{r}_j)). \tag{4.9}$$

For a detailed derivation and discussion of the EFE, we refer the reader to [35].

4.4. Numerical tests on proteins

In this subsection, we evaluate the proposed numerical schemes using real protein systems. Specifically, we illustrate and compare their convergence, accuracy, stability, and efficiency in terms of electrostatic free energies in addition to surface areas and volumes. For the numerical tests presented here, we assume the following parameter values: the dielectric coefficients are set to $\epsilon_m = 1$ and $\epsilon_{out} = 80$, with an ionic strength of $I_s = 0.15$ M, where ϵ_m and ϵ_{out} represent the dielectric constants for the molecular solute and solvent medium, respectively.

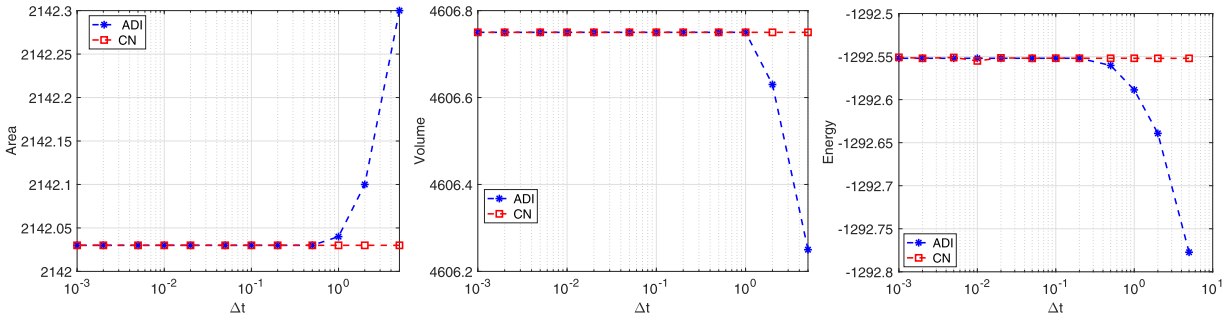
To compute the electrostatic free energies, we use the schemes proposed in this work to generate the solute-solvent interface profile and subsequently solve the linearized Poisson-Boltzmann equation as formulated in (4.5). The electrostatic solvation free energy is then obtained using Equation (4.9). We use a probe radius of $r_p = 1.5$ Å. Surface areas and volumes are computed from the hypersurface extracted at $S = 0.98$, based on the solutions of the p-Laplace equation. To carry out the tests, we fix $\eta = 10^{-7}$ and $p_m = 1.2$. For Picard's iteration, we used a tolerance of 10^{-1} for local convergence (BiCG) and 10^{-7} for global convergence. For the Crank-Nicolson (CN) scheme, the local tolerance was set to 10^{-5}.

4.4.1. Tests on one protein - **1ajj**

Stability, convergence and accuracy of pseudo-time schemes: We begin by conducting tests on the protein **1ajj** (PDB ID: **1ajj**), which has 519 atoms. We investigate the stability, convergence and accuracy of pseudo-time schemes. First, we demonstrate numerically that the proposed pseudo-time methods are unconditionally stable. To illustrate this, we consider a range of Δt values while fixing the mesh size at $h = 0.5$ Å: $\Delta t \in \{0.001, 0.002, 0.005, 0.01, 0.02, 0.05, 0.1, 0.2, 0.5, 1, 2, 5\}$. Meanwhile, we set the stopping time at $T = 10^4\Delta t$ to ensure sufficient accumulation of results. We plot the results in Fig. 8. It is clear that both ADI and CN methods exhibit unconditional stability.

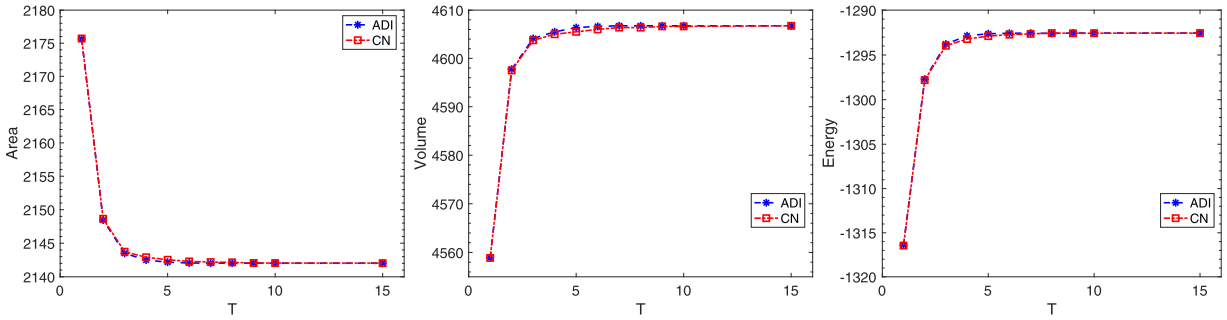
Next, we demonstrate that the pseudo-time methods (ADI and CN) ensure convergence of the solution. For this test, the time step is set to $\Delta t = 0.01$ and the mesh size to $h = 0.5$ Å. As illustrated in Fig. 9, the resulting physical quantities based on the solution profiles, such as the area, volume, and electrostatic solvation energy, converge around $T = 9$.

Furthermore, we evaluate the accuracy of the pseudo-time schemes using the protein **1ajj** as a test case. For this experiment, the stopping time is set to $T = 10$ and the mesh size to $h = 0.25$ Å. The results are summarized in Table 9. The solution obtained from Picard's Iteration scheme at the same mesh size serves as the reference. Specifically, the calculated electrostatic free energy (EFE) for the protein **1ajj** is -1165.817782, and the computed area is 2168.5779. For each method, we report the area, the EFE and its error relative to the reference energy, as well as the CPU time. The results show that ADI and CN schemes produce very similar EFEs and areas to those obtained by the PI method even with different time step sizes.



(a) Area (b) Volume (c) Energy

Fig. 8. Stability tests for the protein PDB ID: **1ajj**.



(a) Area (b) Volume (c) Energy

Fig. 9. Time history of the area, enclosed volume of the molecular surfaces and the associated electronic energy generated by ADI and CN for the protein **1ajj**.

Table 9
Accuracy tests for **1ajj**.

	ADI Scheme				CN Scheme			
Δt	Area	EFE	Error	CPU(s)	Area	EFE	Error	CPU (s)
1	2454.2201	−2078.3524	78.27 %	15.3077	2276.5471	−1459.4565	25.19 %	64.2473
0.5	2417.9697	−1240.9387	6.44 %	30.7787	2228.7401	−1168.2137	0.21 %	95.3386
0.2	2245.4193	−1168.2107	0.21 %	77.3099	2169.1088	−1166.1068	0.02 %	146.9369
0.1	2170.3411	−1165.8710	0.00 %	154.6337	2168.8474	−1165.9567	0.01 %	225.7803
0.05	2168.6895	−1165.9397	0.01 %	286.6408	2168.9464	−1166.0043	0.02 %	380.8611
0.01	2168.5779	−1165.8175	0.00 %	1531.6434	2168.6860	−1165.8758	0.00 %	1638.1257

Computational Efficiency: It is important to evaluate whether the ADI scheme remains the most effective choice for protein systems in terms of accuracy and efficiency. To this end, we analyze the computational time required by various algorithms for fixed time steps and mesh sizes, using the protein **1ajj** as a test case. Picard's Iteration (PI) is employed as the benchmark to calculate the relative time savings achieved by each scheme, as shown in Table 10. The results indicate that the ADI scheme consistently produces fast and accurate results, even for relatively smaller step sizes.

Convergence in p_m: We aim to demonstrate that as m increases, the solution converges to that obtained with $p_m = 1$. To achieve this, we employ the ADI scheme with parameters $T = 10$, $\Delta t = 0.01$, and $h = 0.5$ Å, using Picard's Iteration with $p_m = 1$ as the benchmark. The test is conducted with $\eta = 10^{-7}$.

For $p_m = 1$, Picard's Iteration produces an area of 2031.34874 Å^2, a volume of 4824.000 Å^3, and an EFE of −1238.39802 kcal/mol. As shown in Fig. 10, the results indicate that as p_m approaches 1, the area, volume, and EFE converge to the reference values.

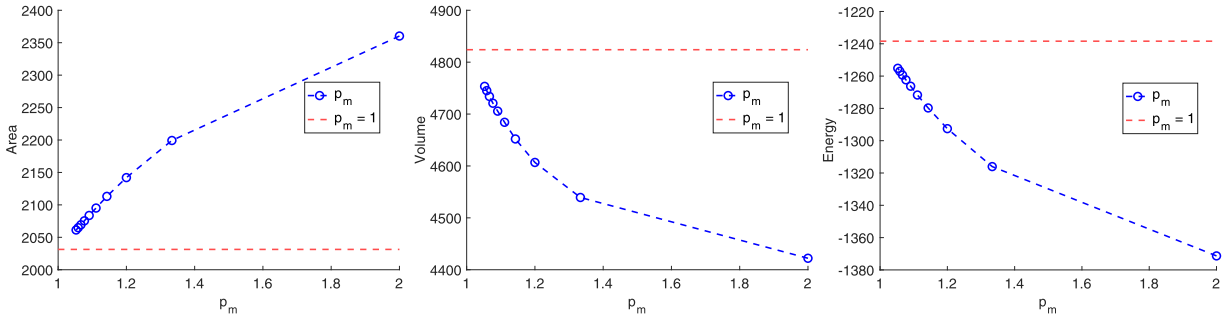
We finally examine the impact of η on the protein **1ajj**. To this end, the ADI method is tested with $T = 10$, $\Delta t = 0.01$, $h = 0.5$, and $p_m = 1.2$. It can be seen from Table 11 that all physical quantities produced by $\eta = 10^{-7}$ are essentially the same as those of $\eta = 10^{-10}$. Therefore, the use of $\eta = 10^{-7}$ does not affect the physical results, while providing an effective regularization to prevent singularities.

4.4.2. Tests on set of 23 proteins

We conclude this section by analyzing a set of 23 proteins commonly tested in the literature for solvation models. The study is conducted using the linearized Poisson-Boltzmann (LPB) model with a fixed mesh size of $h = 0.5$ Å. These proteins vary in size, with the number of atoms ranging from 519 to 2809.

Table 10
CPU times of the proposed schemes for **1ajj** for fixed mesh size and time step.

h	Δt	Scheme	CPU (s)	Area	Relative CPU	Relative Area
		PI	2.1887	2094.2412		
0.8	0.5	ADI	0.4821	2097.1080	22.02 %	$1.3689E-03$
	0.5	CN	1.2132	2094.1346	55.43 %	$5.0871E-05$
		PI	29.6355	2156.0691		
0.4	0.1	ADI	19.3334	2156.2817	65.24 %	$9.8597E-05$
	0.1	CN	31.9769	2156.1031	107.90 %	$1.5753E-05$
		PI	350.2450	2170.3989		
0.05	0.05	ADI	290.2882	2170.6267	82.88 %	$1.0497E-04$
	0.05	CN	484.7998	2173.3230	138.42 %	$1.3473E-03$



(a) Area (b) Volume (c) Electrostatic free Energy

Fig. 10. Convergence for the protein **1ajj** when $p_m \to 1$.

Table 11
Impact of η to the ADI scheme for **1ajj**.

η	Area	Volume	EFE
10^{0}	2350.24812	4426.500	−1369.16684
10^{-4}	2146.00351	4600.625	−1297.10954
10^{-7}	2142.03056	4606.750	−1292.55208
10^{-10}	2142.01852	4606.750	−1292.49457

For the pseudo-time schemes, the stopping time is set to $T = 10$, with a time step size of $\Delta t = 0.2$. The regularization parameter is fixed at $\eta = 10^{-7}$, and $p_m = 1.2$ is used. In the Crank-Nicolson scheme, the local tolerance is set to 10^{-5}, while for Picard's Iteration, the local convergence tolerance for the BiCG linear iterative solver is 10^{-1}, with a global tolerance of 10^{-8}. As shown in Table 12, the three proposed schemes produce very similar electrostatic free energies. For interesting comparisons, energies computed from other surface representations, such as Gaussian Convolution Surfaces (GCS), and Minimal Molecular Surface (MMS), are also listed. Specifically, MMS is calculated with $p_m = 1$ in the p-Laplace equation. For details of the implementation of GCS, we refer the reader to [34] and the references therein. The trilinear method [35] is utilized in the implementation.

To assess the computational efficiency of the three proposed schemes, we report the CPU time required (in seconds) to solve the p-Laplace equation for each protein. These results are summarized in Fig. 11. Notably, the reported CPU times pertain exclusively to computing the surface profile function. The time spent on initial numerical setup, post-surface computations, and solving the LPB equation is excluded, as these processes are consistent across all three methods.

In Fig. 11, the CPU time is plotted against the number of atoms N_m, revealing an approximately linear relationship. The slopes of the lines in the figure indicate the computational complexity of the schemes. Specifically, the CPU time follows $\mathcal{O}(N^r)$, where r represents the slope of the line. It is evident that the ADI scheme remains the most efficient choice among the three, being approximately twice as fast as the Crank-Nicolson scheme and Picard's Iteration.

Electrostatic Potentials on different surface representations: To analyze the differences in electrostatic free energies arising from various surface representations, we present the electrostatic potentials on the surfaces of proteins **1sh1** and **451c** (PDB IDs). These potentials were calculated using three numerical schemes: the ADI scheme with $p_m = 1.2$, the GCS method, and the MMS representation with $p_m = 1$. The results are visualized in Figs. 12 and 13.

The electrostatic potentials were mapped onto isosurfaces generated by each scheme, providing a visual representation of the potential distribution on the computed hypersurfaces. This approach facilitates a direct comparison of the differences between the methods. In all cases, the potentials were scaled linearly to the interval $[-3, 3]$. The computations were performed using the following parameters: grid spacing $h = 0.25$ Å, time step size $\Delta t = 0.01$, stopping time $T = 10$, and isovalue $S = 0.97$.

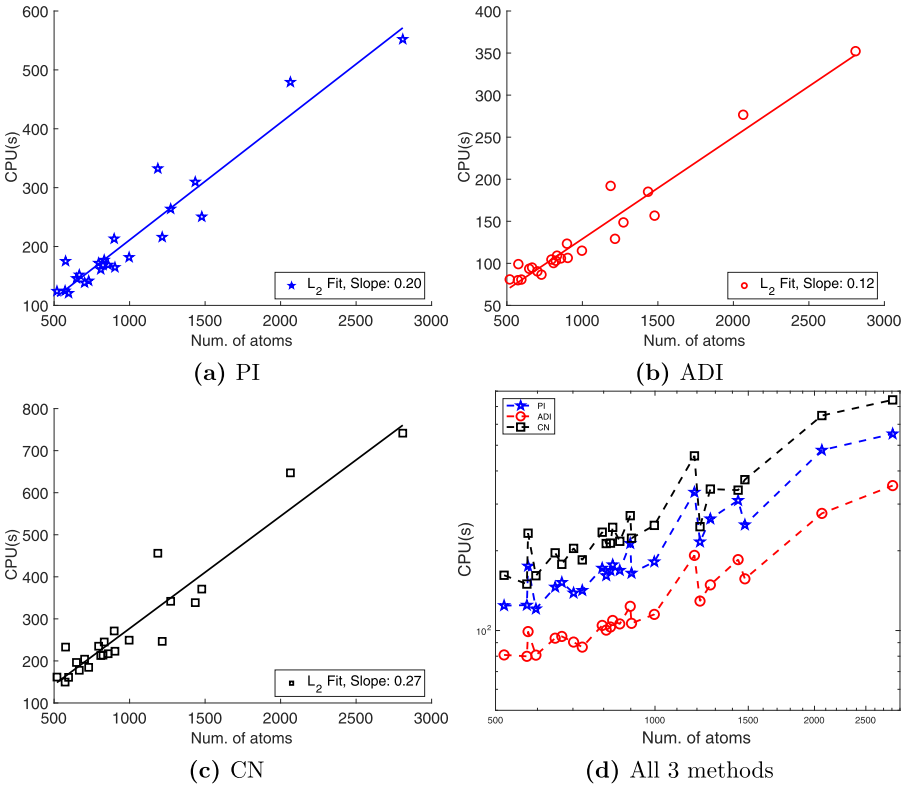
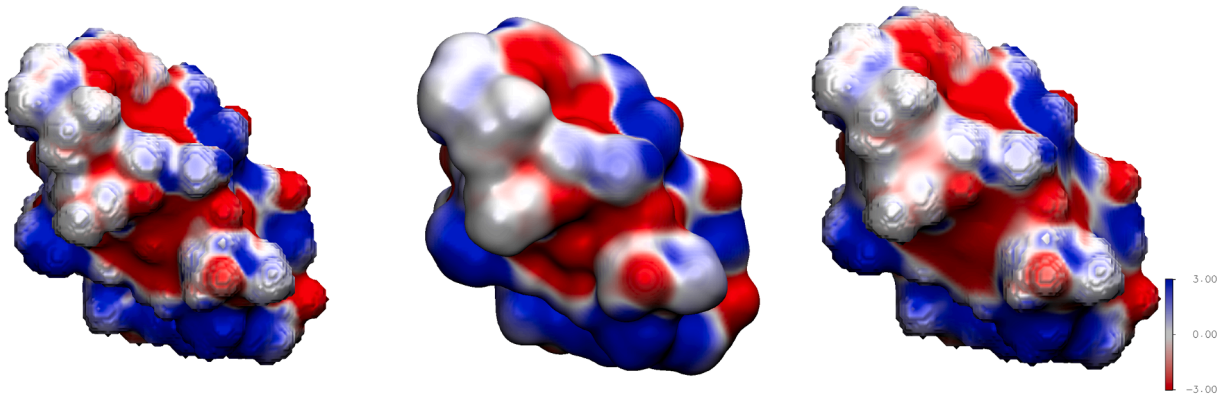


Fig. 11. CPU time in seconds for the set of 23 proteins for $h = 0.25$ and $\Delta t = 0.2$.

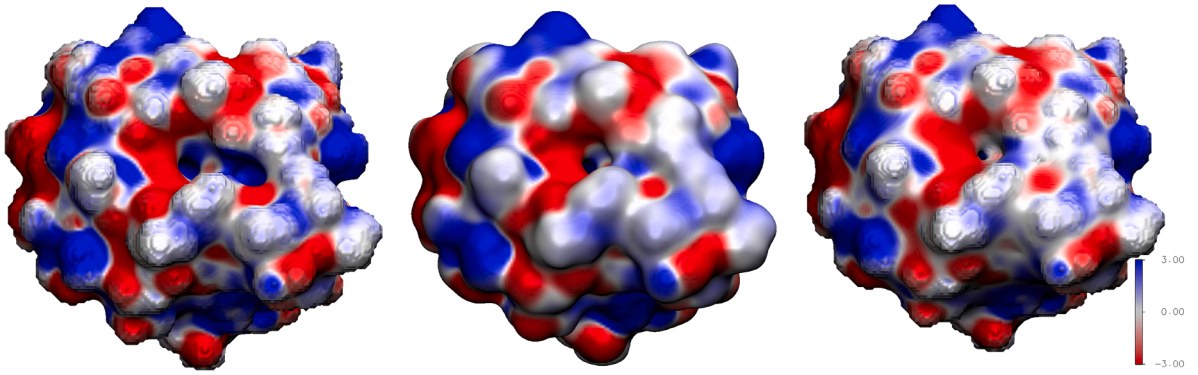
Table 12
Electrostatic free energy for set of 23 proteins in kcal/mol with $h = 0.25$Å, $\Delta t = 0.2$. For ADI, PI, CN and MMS schemes, isovalue of 0.98 was used. Propagation time was set to 10 in steady state integration.

PDB ID	Atom number	Electrostatic free energy				
				p-Laplacian ($p = 1.2$)		
		MMS	GCS	PI	ADI	CN
1ajj	519	−1098.67	−873.78	−1121.47	−1131.95	−1123.30
1bbl	576	−984.22	−707.24	−1003.06	−1013.76	−1004.94
1bor	832	−825.09	−594.62	−844.30	−854.53	−845.98
1bpi	898	−1536.58	−974.47	−1322.23	−1330.89	−1324.20
1cbn	648	−282.00	−161.91	−296.10	−301.67	−297.38
1fca	729	−1204.06	−972.95	−1219.51	−1227.12	−1220.94
1frd	1478	−2851.07	−2349.46	−2884.98	−2905.27	−2888.17
1fxd	824	−3327.68	−2978.79	−3348.50	−3359.11	−3350.43
1hpt	858	−791.26	−490.90	−815.96	−828.97	−818.14
1mbg	903	−1366.29	−1066.10	−1381.90	−1389.39	−1383.26
1neq	1187	−1698.38	−1231.50	−1729.08	−1753.73	−1731.90
1ptq	795	−843.05	−582.07	−867.57	−879.60	−870.01
1r69	997	−1063.51	−759.45	−1092.75	−1104.60	−1094.91
1sh1	702	−694.68	−471.45	−720.11	−737.31	−722.09
1svr	1435	−1663.30	−1154.57	−1710.09	−1731.21	−1713.98
1uxc	809	−1160.34	−811.64	−1177.77	−1188.11	−1179.76
1vii	596	−892.64	−636.29	−908.24	−919.69	−909.70
2erl	573	−964.15	−754.00	−974.77	−980.80	−975.71
2pde	667	−733.80	−557.24	−763.65	−781.40	−766.23
451c	1216	−966.27	−625.63	−995.24	−1016.38	−998.27
1a2s	1272	−1901.46	−1527.73	−1927.11	−1942.28	−1929.49
1a63	2065	−2316.31	−1624.72	−2371.20	−2403.19	−2376.07
1a7m	2809	−2063.77	−1510.07	−2115.82	−2146.21	−2121.05



(a) ADI ($p_m = 1.2$) **(b)** GCS **(c)** MMS ($p_m = 1.0$)

Fig. 12. Potentials mapped onto molecular surface after solving the LPBE with $h = 0.25$, $\Delta t = 0.01$ and $T = 10$. Visualization of the protein (PDB ID: **1sh1**) 12(a)–(c). Potentials were scaled linearly into the interval $[-3, 3]$. For ADI, $p_m = 1.2$ was used. Surfaces were generated using an isovalue of 0.97.



(a) ADI ($p_m = 1.2$) **(b)** GCS **(c)** MMS ($p_m = 1.0$)

Fig. 13. Electrostatic surface potential of the protein (PDB ID: **451c**) mapped onto three surfaces. 13(a): surface generated by ADI ($p_m = 1.2$), 13(b): surface generated by the GCS, and 13(c): surface generated by MMS. Potentials were scaled linearly into the interval $[-3, 3]$ and $h = 0.25$, $\Delta t = 0.01$, $T = 10$, and isovalue set at 0.97.

Upon visual inspection, the potential maps generated by the ADI and MMS schemes exhibit a high degree of similarity, resulting in comparable electrostatic free energies, as shown in Table 12. In contrast, the GCS method produces a distinctly different surface representation, characterized by a noticeably "fatter" appearance. These differences suggest that the GCS method may capture features that are not as apparent in the ADI and MMS schemes.

5. Conclusion

The widely studied p-Laplace equation has numerous applications across various fields. Due to its strong nonlinearity (being degenerate for $p > 2$ and singular for $p < 2$ at the critical points ($\nabla S = 0$)) solving the nonlinear and degenerate p-Laplace equation numerically is highly challenging. As a result, developing efficient solvers for this equation has garnered significant attention in the numerical analysis community. Several numerical techniques have been employed, including Finite Element Methods (FEM), Finite Difference Methods (FDM), and steepest descent algorithms. Notably, for fully nonlinear or degenerate elliptic equations, certain finite difference schemes have proven particularly effective.

In this work, we proposed and explored numerical schemes for solving the p-Laplace equation in $\mathbb{R}^3$ for $p \in [1, 2)$, aiming to identify the most effective numerical approaches within the framework of finite difference methods. It is noteworthy that previous numerical experiments primarily focused on dimensions $d \leq 2$. To our knowledge, this is the first comprehensive study on the 3D case. Three numerical schemes –Picard iteration, an Alternating Direction Implicit (ADI) method, and the Crank-Nicolson (CN) method– emerged as the most promising and were therefore selected for comparison and discussion. These numerical schemes have been validated using benchmark analytical solutions. In particular, our results demonstrate that the ADI scheme is the most efficient among the three, requiring the least computational time while maintaining comparable accuracy in both analytical tests and solvation energy

calculations. The stability of the CN method has been proved in this work. As a high-order perturbation to the CN scheme, the ADI method is also found to be unconditionally stable. For a well-posed linear PDE, the convergence of a numerical scheme can usually be guaranteed based on its consistency and stability. However, because the p-Laplace equation is highly nonlinear, the convergence analysis of the proposed schemes is not straightforward, and will be explored in the future.

Moreover, we numerically verified that the solution of pseudo-time parabolic p-Laplace equation converges to the solution of the corresponding highly nonlinear elliptic p-Laplace equation. In the literature, studies on the parabolic p-Laplace equation primarily consider the Cauchy problem with $p \geq 2$. Additionally, previous investigations on the normalized p-Laplace equation are not well suited for the parabolic case. Our study bridges this gap by examining the parabolic p-Laplace equation for $p \in [1, 2)$. To address the challenges of singularity and blow-up within the computational domain, we introduced a small regularization parameter. This approach effectively prevents singularities while ensuring that the original solution is accurately approximated.

After making tangible progress on effective numerical schemes for solving the p-Laplace equation, we are now prepared to investigate the aforementioned coupled, highly nonlinear PDEs in protein systems. For this purpose, the challenges arising from the coupling of highly nonlinear PDEs, the geometric complexity of real protein systems, and the parameterization schemes will be carefully examined in the near future.

Data availability

Data will be made available on request.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This research was partially supported by the National Science Foundation under grants DMS-2512104 (Zhao, Shao), DMS-2306991 (Shao, Zhao), DMS-2306992 (Chen), DMS-2110914 (Zhao), and a CARSCA grant from the University of Alabama (Shao).

Appendix A. Stability analysis of the Crank-Nicolson method

In this appendix, we will use the Gershgorin Circle Theorem to prove the unconditional stability of the Crank-Nicolson scheme for a pseudo-time parabolic p-Laplace equation in 3D:

$$\begin{cases} \partial_t S - \nabla \cdot (|\nabla S|^{p-2} \nabla S) = f & \text{in} \quad \Omega \times [0, T], \\ S = g & \text{on} \quad \partial\Omega \times [0, T], \\ S|_{t=0} = S^0 & \text{in} \quad \Omega, \end{cases} \tag{A.1}$$

for any $T > 0$, where f and g are smooth and bounded functions defined on $\Omega \subseteq \mathbb{R}^3$.

We use the Crank-Nicolson method to discretize (A.1) as follows:

$$\begin{cases} \dfrac{S^{n+1} - S^n}{\Delta t} - \nabla \cdot \left(|\nabla S^n|^{p-2} \nabla \dfrac{S^{n+1} + S^n}{2} \right) = f & \text{in} \quad \Omega, \\ S^{n+1} = g & \text{on} \quad \partial\Omega. \end{cases} \tag{A.2}$$

By introducing a new variable $u^n = S^n - g$, we can recast $(A.2)_1$ into:

$$u^{n+1} - \frac{\Delta t}{2} \nabla \cdot \left(|\nabla S^n|^{p-2} \nabla u^{n+1} \right) = u^n + \frac{\Delta t}{2} \nabla \cdot \left(|\nabla S^n|^{p-2} \nabla u^n \right) + \Delta t f + \Delta t \nabla \cdot \left(|\nabla S^n|^{p-2} \nabla g \right). \tag{A.3}$$

Observe that $u^{n+1} = 0$ on $\partial\Omega$.

Define I to be the set of all interior mesh points and $N = |I|$. Note that $\beta = (|\nabla S|^2 + \eta)^{(p-2)/2}$, where a regularizing parameter η has been added to avoid dividing by zero. Then $\nabla \cdot \left(|\nabla S^n|^{p-2} \nabla \phi \right)$ can be discretized into

$$\begin{aligned} &- \frac{1}{h^2} \left(\beta^n_{i+\frac{1}{2},j,k} + \beta^n_{i-\frac{1}{2},j,k} + \beta^n_{i,j+\frac{1}{2},k} + \beta^n_{i,j-\frac{1}{2},k} + \beta^n_{i,j,k+\frac{1}{2}} + \beta^n_{i,j,k+\frac{1}{2}} \right) \phi_{i,j,k} \\ &+ \frac{1}{h^2} \left(\beta^n_{i+\frac{1}{2},j,k} \phi_{i+1,j,k} + \beta^n_{i-\frac{1}{2},j,k} \phi_{i-1,j,k} + \beta^n_{i,j+\frac{1}{2},k} \phi_{i,j+1,k} + \beta^n_{i,j-\frac{1}{2},k} \phi_{i,j-1,k} + \beta^n_{i,j,k+\frac{1}{2}} \phi_{i,j,k+1} + \beta^n_{i,j,k-\frac{1}{2}} \phi_{i,j,k-1} \right) \end{aligned}$$

for $(i, j, k) \in I$. This motivates us to define the discrete linear operator $\mathbf{D}_n := \mathbf{D}(S^n) \in \mathbb{R}^{N \times N}$ as follows:

$$(\mathbf{D}_n \phi)_{i,j,k} = \frac{1}{2h^2} \Big(\alpha_{i+1,j,k} \beta^n_{i+\frac{1}{2},j,k} \phi_{i+1,j,k} + \alpha_{i-1,j,k} \beta^n_{i-\frac{1}{2},j,k} \phi_{i-1,j,k} + \alpha_{i,j+1,k} \beta^n_{i,j+\frac{1}{2},k} \phi_{i,j+1,k}$$

$$+\alpha_{i,j-1,k}\beta^n_{i,j-\frac{1}{2},k}\phi_{i,j-1,k}+\alpha_{i,j,k+1}\beta^n_{i,j,k+\frac{1}{2}}\phi_{i,j,k+1}+\alpha_{i,j,k-1}\beta^n_{i,j,k-\frac{1}{2}}\phi_{i,j,k-1}\Bigg)$$
$$-\frac{1}{2h^2}\Big(\beta^n_{i+\frac{1}{2},j,k}+\beta^n_{i-\frac{1}{2},j,k}+\beta^n_{i,j+\frac{1}{2},k}+\beta^n_{i,j-\frac{1}{2},k}+\beta^n_{i,j,k+\frac{1}{2}}+\beta^n_{i,j,k+\frac{1}{2}}\Big)\phi_{i,j,k},\quad (i,j,k)\in I,$$

where

$$\alpha_{i,j,k}=\begin{cases}0, & (x_i,y_j,z_k)\in\partial\Omega\\ 1, & \text{otherwise.}\end{cases}$$

Here we view ϕ as a column vector in $\mathbb{R}^N$. For a boundary node (x_i,y_j,z_k), $\phi_{i,j,k}$ is not part of the vector ϕ. However, its value is arbitrary in the above definition. In addition, we set $\mathbf{A}_n=\mathbf{I}-\Delta t\mathbf{D}_n$, and $\mathbf{B}_n=\mathbf{I}+\Delta t\mathbf{D}_n$.

By using these notations, (A.3) can be recasted into the following matrix form:

$$\mathbf{A}_n u^{n+1}=\mathbf{B}_n u^n+\Delta t f+2\Delta t\mathbf{D}_n g,$$

or equivalently,

$$u^{n+1}=\mathbf{M}_n u^n+\Delta t\big[\mathbf{A}_n^{-1}f+2\mathbf{A}_n^{-1}\mathbf{D}_n g\big]. \tag{A.4}$$

Here we identity u^n, f, g as column matrices in $\mathbb{R}^N$. Let $\mathbf{M}_n=(\mathbf{I}-\Delta t\mathbf{D}_n)^{-1}(\mathbf{I}+\Delta t\mathbf{D}_n)=\mathbf{A}_n^{-1}\mathbf{B}_n$ be the amplification matrix. Note that $\mathbf{D}_n$ is symmetric. Since $\mathbf{A}_n^{-1}$ commutes with $\mathbf{A}_n=\mathbf{I}-\Delta t\mathbf{D}_n=2\mathbf{I}-\mathbf{B}_n$, we see that $\mathbf{A}_n^{-1}$ and $\mathbf{B}_n$ commutes. Consequently, $\mathbf{M}_n$ is also symmetric.

We define the discrete L_h^2-norm as

$$\|v\|_2:=\|v\|_{L_h^2}=\left(h^3\sum_{(i,j,k)\in I}|v(x_i,y_j,z_k)|^2\right)^{1/2}.$$

In addition, because $\beta=(|\nabla S|^2+\eta)^{(p-2)/2}$ and $p\in(1,2)$. One readily sees that there exists some constant $C{=}C(\eta)>0$ such that $\beta\le C(\eta)$. In particular, this implies that for any $v\in L_h^2$,

$$\|\mathbf{D}_n v\|_2\le\frac{1}{h^2}\tilde{C}(\eta)\|v\|_2 \tag{A.5}$$

for some constant $\tilde{C}(\eta)$ independent of n and N.

The matrix $\mathbf{D}_n=(d_{\ell,m})_{1\le\ell,m\le N}$ satisfies

$$d_{\ell,\ell}=-\frac{1}{2h^2}\Big[\beta_{i-\frac{1}{2},j,k}+\beta_{i+\frac{1}{2},j,k}+\beta_{i,j-\frac{1}{2},k}+\beta_{i,j+\frac{1}{2},k}+\beta_{i,j,k-\frac{1}{2}}+\beta_{i,j,k+\frac{1}{2}}\Big]<0$$

and

$$|d_{\ell,\ell}|\ge\sum_{\substack{m=1\\ m\ne\ell}}^{N}|d_{\ell,m}|=:R_\ell.$$

Since $\mathbf{D}_n$ is diagonally dominant with $d_{\ell,\ell}$ real and negative, we infer that the spectrum of $\mathbf{D}_n$ is nonpositive

$$\sigma(\mathbf{D})\subset(-\infty,0]. \tag{A.6}$$

Indeed, since any disk centered at $(d_{\ell,\ell},0)$ with radius $R_\ell\le|d_{\ell,\ell}|$ lies in the left half of the complex plane, Eq. (A.6) immediately follows from the Gershgorin Circle Theorem and the symmetry of $\mathbf{D}_n$.

Direct computations show that any eigenvalue $\lambda_{\mathbf{M}_n}$ of $\mathbf{M}_n$ is of the form:

$$\lambda_{\mathbf{M}_n}=\frac{1+\Delta t\lambda}{1-\Delta t\lambda}\quad\text{for some }\lambda\in\sigma(\mathbf{D}_n).$$

(A.6) implies that the spectra of $\mathbf{A}_n^{-1}$ and $\mathbf{M}_n$ satisfy

$$\sigma(\mathbf{A}^{-1})\subset(0,1]\quad\text{and}\quad\sigma(\mathbf{M}_n)\subset[-1,1]. \tag{A.7}$$

Utilizing the recursive relation (A.4), we obtain the expression

$$u^{n+1}=\prod_{l=0}^{n}\mathbf{M}_l u^0+\Delta t\left(\mathbf{A}_n^{-1}+\mathbf{M}_n\mathbf{A}_{n-1}^{-1}+\cdots+\prod_{l=1}^{n}\mathbf{M}_l\mathbf{A}_0^{-1}\right)f+2\Delta t\left(\mathbf{A}_n^{-1}\mathbf{D}_n+\mathbf{M}_n\mathbf{A}_{n-1}^{-1}\mathbf{D}_{n-1}+\cdots+\prod_{l=1}^{n}\mathbf{M}_l\mathbf{A}_0^{-1}\mathbf{D}_0\right)g. \tag{A.8}$$

Here

$$\prod_{k=i}^{j}\mathbf{M}_k=\mathbf{M}_j\mathbf{M}_{j-1}\cdots\mathbf{M}_i,\quad i<j.$$

In view of (A.7), we have

$$\left\| \prod_{l=0}^{n} \mathbf{M}_l u^0 \right\|_2 \leq \|u^0\|_2. \tag{A.9}$$

By virtue of (A.5) and (A.7), we can infer that

$$\left\| \Delta t \left(\mathbf{A}_n^{-1}\mathbf{D}_n + \mathbf{M}_n\mathbf{A}_{n-1}^{-1}\mathbf{D}_{n-1} + \cdots + \prod_{l=1}^{n} \mathbf{M}_l\mathbf{A}_0^{-1}\mathbf{D}_0 \right) g \right\|_2 \leq \Delta t \sum_{l=0}^{n} \|\mathbf{D}_l g\|_2 \leq \frac{1}{h^2}\Delta t(n+1)\tilde{C}(\eta)\|g\|_2 \leq \frac{T}{h^2}\tilde{C}(\eta)\|g\|_2. \tag{A.10}$$

Similarly, it holds that

$$\left\| \Delta t \left(\mathbf{A}_n^{-1} + \mathbf{M}_n\mathbf{A}_{n-1}^{-1} + \cdots + \prod_{l=1}^{n} \mathbf{M}_l\mathbf{A}_0^{-1} \right) f \right\|_2 \leq T\tilde{C}(\eta)\|f\|_2. \tag{A.11}$$

Combining (A.9)-(A.11), we arrive at the final estimate for S^n:

$$\|S^n\|_2 \leq \|S^0\|_2 + \left[2 + \frac{T}{h^2}\tilde{C}(\eta) \right] \|g\|_2 + T\tilde{C}(\eta)\|f\|_2. \tag{A.12}$$

Remark. The presence of the constant $\tilde{C}(\eta)$ in (A.12) is not surprising. Because in the proof of the unconditional stability of the Crank-Nicolson method (when applied to the heat equation $\partial_t u - \kappa\Delta u = f$), the corresponding bound depends on the diffusion coefficient κ. In (A.1), the bound $\eta^{(p-2)/2}$ plays the same role as κ.

References

[1] S. Oruganti, J. Shi, R. Shivaji, Diffusive logistic equation with constant yield harvesting, i: steady states, Trans. Am. Math. Soc. 354 (9) (2002) 3601–3619.
[2] F. Catté, P.-L. Lions, J.-M. Morel, T. Coll, Image selective smoothing and edge detection by nonlinear diffusion, SIAM J. Numer. Anal. 29 (1) (1992) 182–193.
[3] F. Andreu, J.M. Mazón, J.D. Rossi, J. Toledo, A nonlocal p-laplacian evolution equation with nonhomogeneous dirichlet boundary conditions, SIAM J. Math. Anal. 40 (5) (2009) 1815–1851.
[4] Z. Chen, S. Zhao, Y. Shao, A p-laplacian approach and its analysis for the calculation of ensemble average solvation energy, Commun. Inf. Syst. 24 (3) (2024) 275–311.
[5] P. Lindqvist, Notes on the p-Laplace equation, 161, University of Jyväskylä, 2017.
[6] A.M. Oberman, Finite difference methods for the infinity laplace and p-laplace equations, J. Comput. Appl. Math. 254 (2013) 65–80.
[7] Z. Luo, F. Teng, An effective finite element newton method for 2D p-laplace equation with particular initial iterative function, J. Inequalit. Appl. 2016 (2016) 1–24.
[8] M.D. Haggar, M. Mbehou, A. Njifenjou, Crank-Nicolson finite element methods for nonlocal problems with p-laplace-type operator, J. Math. Comput. Sci. 33 (2024) 57–70.
[9] J.W. Barrett, W.B. Liu, Finite element approximation of the p-laplacian, Math. Comput. 61 (204) (1993) 523–537.
[10] R. Ferreira, A. De Pablo, Numerical blow-up for the p-laplacian equation with a source, Comput. Methods Appl. Math. 5 (2) (2005) 137–154.
[11] R. Glowinski, A. Marroco, Sur l'approximation, par éléments finis d'ordre un, et la résolution, par pénalisation-dualité d'une classe de problèmes de dirichlet non linéaires, Revue française d'automatique, informatique, recherche opérationnelle. Analyse numérique 9 (R2) (1975) 41–76.
[12] K.Y. Kim, Error estimates for a mixed finite volume method for the p-laplacian problem, Numer. Math. 101 (1) (2005) 121–142.
[13] Y.Q. Huang, R. Li, W. Liu, Preconditioned descent algorithms for p-laplacian, J. Sci. Comput. 32 (2007) 343–371.
[14] S. Loisel, Efficient algorithms for solving the p-laplacian in polynomial time, Numer. Math. 146 (2) (2020) 369–400.
[15] L. Codenotti, M. Lewicka, J. Manfredi, Discrete approximations to the double-obstacle problem and optimal stopping of tug-of-war games, Trans. Am. Math. Soc. 369 (10) (2017) 7387–7403.
[16] F. del Teso, E. Lindgren, A finite difference method for the variational p-laplacian, J. Sci. Comput. 90 (1) (2022) 67.
[17] F. del Teso, J. Endal, E.R. Jakobsen, Robust numerical methods for nonlocal (and local) equations of porous medium type. part II: schemes and experiments, SIAM J. Numer. Anal. 56 (6) (2018) 3611–3647.
[18] F. del Teso, J.J. Manfredi, M. Parviainen, Convergence of dynamic programming principles for the p-laplacian, Adv. Calculus Variat. 15 (2) (2022) 191–212.
[19] M. Falcone, S.F. Vita, T. Giorgi, R.G. Smits, A semi-lagrangian scheme for the game p-laplacian via p-averaging, Appl. Numer. Math. 73 (2013) 63–80.
[20] Q. Chen, W. Hao, Randomized Newton's method for solving differential equations based on the neural network discretization, J. Sci. Comput. 92 (2) (2022) 49.
[21] P.W. Bates, Z. Chen, Y. Sun, G.-W. Wei, S. Zhao, Geometric and potential driving formation and evolution of biomolecular surfaces, J. Math. Biol. 59 (2009) 193–231.
[22] W. Tian, S. Zhao, A fast alternating direction implicit algorithm for geometric flow equations in biomolecular surface generation, Int. J. Numer. Method Biomed. Eng. 30 (4) (2014) 490–516.
[23] Y. Lu, C.-A. Chen, X. Li, C. Liu, Finite difference approximation with ADI scheme for two-Dimensional keller-Segel equations, Commun. Comput. Phys. 35 (5) (2024) 1352–1386.
[24] K. Li, W. Liao, Y. Lin, A compact high order alternating direction implicit method for three-dimensional acoustic wave equation with variable coefficient, J. Comput. Appl. Math. 361 (2019) 113–129.
[25] M. He, W. Liao, A compact ADI finite difference method for 2D reaction-diffusion equations with variable diffusion coefficients, J. Comput. Appl. Math. 436 (2024) 115400.
[26] T. Guo, O. Nikan, Z. Avazzadeh, W. Qiu, Efficient alternating direction implicit numerical approaches for multi-dimensional distributed-order fractional integro differential problems, Comput. Appl. Math. 41 (6) (2022) 236.
[27] M. Liu, T. Guo, M.A. Zaky, A.S. Hendy, An accurate second-order ADI scheme for three-dimensional tempered evolution problems arising in heat conduction with memory, Appl. Numer. Math. 204 (2024) 111–129.
[28] S. Osher, R. Fedkiw, Level set methods and dynamic implicit surfaces, Springer, Berlin. 2003. https://doi.org/https://doi.org/10.1007/b98879
[29] F. del Teso, E. Lindgren, Finite difference schemes for the parabolic p-laplace equation, SeMA J. 80 (4) (2023) 527–547.
[30] J.C. Strikwerda, Finite difference schemes and partial differential equations, SIAM, 2004.
[31] P.W. Bates, G.-W. Wei, S. Zhao, Minimal molecular surfaces and their applications, J. Comput. Chem. 29 (3) (2008) 380–391.
[32] T. Hazra, S. Ahmed Ullah, S. Wang, E. Alexov, S. Zhao, A super-gaussian poisson–Boltzmann model for electrostatic free energy calculation: smooth dielectric distribution for protein cavities and in both water and vacuum states, J. Math. Biol. 79 (2019) 631–672.
[33] S. Wang, E. Alexov, S. Zhao, On regularization of charge singularities in solving the poisson-Boltzmann equation with a smooth solute-solvent boundary, Math. Biosci. Eng.: MBE 18 (2) (2021) 1370.

[34] S. Wang, Y. Shao, E. Alexov, S. Zhao, A regularization approach for solving the super-gaussian poisson-Boltzmann model with heterogeneous dielectric functions, J. Comput. Phys. 464 (2022) 111340.
[35] S. Zhao, I.E. Ijaodoro, M. McGowan, E. Alexov, Calculation of electrostatic free energy for the nonlinear poisson-Boltzmann model based on the dimensionless potential, J. Comput. Phys. 497 (2024) 112634.
[36] M.J. Holst, F. Saied, Numerical solution of the nonlinear poisson–Boltzmann equation: developing more robust and efficient methods, J. Comput. Chem. 16 (3) (1995) 337–364.
[37] N.A. Baker, Improving implicit solvent simulations: a poisson-centric view, Curr. Opin. Struct. Biol. 15 (2) (2005) 137–143.
[38] B. Honig, A. Nicholls, Classical electrostatics in biology and chemistry, Science 268 (5214) (1995) 1144–1149.