doi：10.13413/j.cnki.jdxblxb.2024391

求解 PNP 方程的虚单元两水平算法

毛万涛，阳 莺
（桂林电子科技大学 数学与计算科学学院，广西应用数学中心(GUET)，
广西高校数据分析与计算重点实验室，广西 桂林 541004）

摘要：针对稳态和含时两种 PNP(Poisson-Nernst-Planck)方程，提出一种基于虚单元离散的两水平算法. 该算法先利用线性虚单元解对 PNP 方程进行解耦和线性化，然后在二次虚单元空间上求解. 与 PNP 方程求解常用的 Gummel 算法相比，该算法能加快求解速度. 包含稳态和含时两种 PNP 方程的数值实验结果表明，与线性虚单元的 Gummel 算法相比，两水平算法精度更高，且在相当精度下，耗费 CPU 时间更少，效率更高.

关键词：Poisson-Nernst-Planck 方程；虚单元方法；两水平算法；Gummel 算法
中图分类号：O241.82 **文献标志码**：A **文章编号**：1671-5489(2025)04-1051-08

Virtual Element Two-Level Algorithm for Solving PNP Equations

MAO Wantao, YANG Ying
(*Guangxi Applied Mathematics Center (GUET), Guangxi Colleges and Universities Key Laboratory of Data Analysis and Computation, School of Mathematics & Computing Science, Guilin University of Electronic Technology, Guilin* 541004, *Guangxi Zhuang Autonomous Region, China*)

Abstract: We proposed a two-level algorithm based on the virtual element discretization for both steady-state and time-dependent PNP (Poisson-Nernst-Planck) equations. This algorithm first decoupled and linearized the PNP equations using the linear virtual element solution, and then solved them in the quadratic virtual element space. Compared with the commonly used Gummel algorithm for solving the PNP equations, this algorithm could accelerate the solving speed. Numerical experimental results, including both steady-state and time-dependent PNP equations, show that compared with the Gummel algorithm with the linear virtual element, the two-level algorithm has higher accuracy, and consumes less CPU time and is more efficient at comparable accuracy.

Keywords: Poisson-Nernst-Planck equations; virtual element method; two-level algorithm; Gummel algorithm

考虑一类由 Poisson 方程和 Nernst-Planck 方程耦合而成的 PNP 方程，该方程通常用于描述离子质量守恒和静电势扩散反应过程，在生物分子系统、半导体器件和电化学系统等领域应用广泛.

由于 PNP 方程自身有强耦合性和非线性性，通常很难寻求解析解，因此，有限差分法、有限元法和有限体积法等许多数值方法被应用于 PNP 方程的数值求解中. 虚单元法是在有限元法的基础上发展的一种新型偏微分方程数值离散技术，与传统有限元法相比，虚单元法有良好的网格适应性，已广

收稿日期：2024-10-08.

第一作者简介：毛万涛(2000—)，女，彝族，硕士研究生，从事偏微分方程数值解的研究，E-mail：2195795377@qq.com. **通信作者简介**：阳 莺(1976—)，女，汉族，博士，教授，从事偏微分方程数值解的研究，E-mail：yangying@lsec.cc.ac.cn.

基金项目：国家自然科学基金(批准号：12161026)、广西自然科学基金(批准号：2025GXNSFAA069683)和广西科技基地和人才专项(批准号：桂科 AD23026048).

泛应用于偏微分方程数值求解的多个领域. 文献[1]和文献[2]分别针对稳态和含时 PNP 方程给出了虚单元方法及相应的误差分析.

PNP 方程是一类非线性耦合方程组，一种常用的解耦和线性化方法是 Gummel 迭代法[3-5]. 文献[6]和文献[7]分别针对稳态和含时 PNP 方程提出了有限元两网格算法. 该算法先利用粗网格解对方程进行解耦和线性化，再在细网格上求解方程. 相较于传统的 Gummel 有限元算法，能加快求解过程. 然而，该算法依赖于粗细嵌套网格的构造，而在处理复杂实际问题时，嵌套网格的设计通常很困难，不利于应用. 文献[8]提出了一种有限元两水平算法，并将其应用到非对称不定的椭圆问题中. 该算法的基本思想是先利用低阶有限元解对问题进行对称化和正定化，然后在高阶有限元空间上求解对称正定问题. 由于两水平算法仅使用一套网格，因此与两网格算法相比更适用于实际计算.

本文针对 PNP 问题，采用虚单元法进行离散并结合两水平算法进行计算. 该算法的主要过程是：首先在线性虚单元空间上求解原问题获得线性元解，利用线性元解对方程进行解耦及线性化，然后在高阶虚单元空间上求解解耦后的方程，从而获得两水平解. 数值实验结果表明，对于稳态和含时 PNP 方程，与常用的线性虚单元的 Gummel 算法相比，两水平算法能达到更高的精度. 此外，在相当精度下，两水平算法所需的 CPU 时间显著低于 Gummel 算法，证明了两水平算法的有效性.

1 预备知识

设 $\Omega\subset\mathbb{R}^2$ 是有界多边形区域. 本文采用 Sobolev 空间 $W^{k,p}(\Omega)$ 中的标准记号及相关的范数和半范数的定义，分别记为 $\|\cdot\|_{k,p,\Omega}=\|\cdot\|_{W^{k,p}(\Omega)}$ 和 $|\cdot|_{k,p,\Omega}$. 特别地，当 $p=2$ 时，记 $H^k(\Omega)=W^{k,2}(\Omega)$，$H_0^1(\Omega)=\{v\,|\,v\in H^1(\Omega);\ v|_{\partial\Omega}=0\}$，$(\cdot,\cdot)_\Omega$ 表示标准的 L^2 内积.

一类典型的含时 PNP 方程[7]为

$$\begin{cases}\partial_t p^i-\nabla\cdot(\nabla p^i+c_\lambda q^i p^i\nabla\phi)=F_i, & i=1,2,\ \text{在}\ \Omega_T\ \text{内},\\ -\Delta\phi-\sum\limits_{i=1}^{2}q^i p^i=f, & \text{在}\ \Omega_T\ \text{内},\end{cases}\tag{1}$$

其初边值条件为

$$\begin{cases}\phi=0, & \text{在}\ \partial\Omega_T\ \text{上},\\ p^i=0,\quad i=1,2, & \text{在}\ \partial\Omega_T\ \text{上},\\ p^i(x,0)=0,\quad i=1,2, & \text{在}\ \Omega\ \text{内},\end{cases}\tag{2}$$

其中：$\Omega_T=\Omega\times(0,T]$，$\partial\Omega_T=\partial\Omega\times(0,T]$；$\phi$ 为静电势，p^1,p^2 表示两种离子浓度；$q^1=1$，$q^2=-1$ 表示离子的带电量；c_λ 表示无量纲化后的常数；F_1,F_2,f 为反应源项.

方程(1)所对应的稳态 PNP 方程[9]为

$$\begin{cases}-\nabla\cdot(\nabla p^i+c_\lambda q^i p^i\nabla\phi)=F_i, & i=1,2,\ \text{在}\ \Omega\ \text{内},\\ -\Delta\phi-\sum\limits_{i=1}^{2}q^i p^i=f, & \text{在}\ \Omega\ \text{内}.\end{cases}\tag{3}$$

相应的 Dirichlet 边界条件为

$$\begin{cases}\phi=0, & \text{在}\ \partial\Omega\ \text{上},\\ p^i=0,\quad i=1,2, & \text{在}\ \partial\Omega\ \text{上}.\end{cases}\tag{4}$$

方程(1)-(2)的变分形式为：求 $p^i\in L^2(0,T;H_0^1(\Omega))(i=1,2)$ 及 $\phi\in L^2(0,T;H_0^1(\Omega))$，满足

$$(\partial_t p^i,v)+(\nabla p^i,\nabla v)+(c_\lambda q^i p^i\nabla\phi,\nabla v)=(F_i,v),\quad i=1,2,\quad\forall\, v\in H_0^1(\Omega),\tag{5}$$

$$(\nabla\phi,\nabla w)-\sum_{i=1}^{2}q^i(p^i,w)=(f,w),\qquad\forall\, w\in H_0^1(\Omega).\tag{6}$$

方程(3)-(4)的变分形式为：求 $p^i\in H_0^1(\Omega)(i=1,2)$ 及 $\phi\in H_0^1(\Omega)$，满足

$$(\nabla p^i,\nabla v)+(c_\lambda q^i p^i\nabla\phi,\nabla v)=(F_i,v),\quad i=1,2,\quad\forall\, v\in H_0^1(\Omega),\tag{7}$$

$$(\nabla\phi,\nabla w)-\sum_{i=1}^{2}q^i(p^i,w)=(f,w),\qquad\forall\, w\in H_0^1(\Omega).\tag{8}$$

2　虚单元离散及两水平算法

下面先给出虚单元空间及相关投影算子的定义，再以问题(1)-(2)为例，给出虚单元离散格式及相应的两水平算法. 问题(3)-(4)的离散格式及两水平算法可类似给出.

2.1　虚单元空间

假设 $\mathcal{T}^h=\{E\}$为 Ω 的网格剖分，满足如下假设[10]：对每个单元 $E\in\mathcal{T}^h$，存在常数 $\gamma>0$ 和 $c>0$，使得：

1) 单元 E 对半径大于等于 γh_E 的圆盘上的点都是星形区域；

2) 单元 E 任意两个顶点之间的距离大于等于 ch_E.

对 $\forall E\in\mathcal{T}^h$，首先引入空间 $\widetilde{Q}_h^k(E)$[11]：

$$\widetilde{Q}_h^k(E)=\{v\in H^1(E);\ v|_{\partial E}\in C^0(\partial E),\ v|_e\in\mathcal{P}_k(e),\ \forall e\in\partial E,\ \Delta v\in\mathcal{P}_k(E)\}.$$

定义椭圆投影算子 $\Pi_k^\nabla: H^1(E)\to\mathcal{P}_k(E)$，对 $\forall v\in H^1(E)$，满足

$$\begin{cases}(\nabla(\Pi_k^\nabla v-v),\nabla q)_{0,E}=0, & \forall q\in\mathcal{P}_k(E),\\ P_0(\Pi_k^\nabla v-v)=0,\end{cases}\tag{9}$$

其中

$$P_0(v)=\begin{cases}\dfrac{1}{N^V}\displaystyle\sum_{i=1}^{N^V}v(V_i), & k=1,\\ \dfrac{1}{|E|}\displaystyle\int_E v\,\mathrm{d}\boldsymbol{x}, & k\geqslant 2.\end{cases}$$

定义 L^2 投影算子 Π_k^0：$L^2(E)\to\mathcal{P}_k(E)$，对 $\forall v\in L^2(E)$，满足

$$(v-\Pi_k^0 v,q)_{0,E}=0,\qquad \forall q\in\mathcal{P}_k(E).\tag{10}$$

定义局部虚单元空间为

$$Q_h^k(E)=\{v_h\in\widetilde{Q}_h^k(E);\ (v_h-\Pi_k^\nabla v_h,q)_E=0,\ \forall q\in(\mathcal{P}_k/\mathcal{P}_{k-2}(E))\},$$

其中 $\mathcal{P}_k/\mathcal{P}_{k-2}(E)$表示 $\mathcal{P}_k(E)$中在 $L^2(E)$内积下与 $\mathcal{P}_{k-2}(E)$正交的多项式. 利用上述局部虚单元空间给出整体虚单元空间的定义为 $Q_h^k=\{v_h\in H^1(\Omega);\ v_h|_E\in Q_h^k(E),\ \forall E\in\mathcal{T}^h\}$.

2.2　虚单元离散形式

对 $\forall u,v,\phi,u^1,u^2\in H_0^1(\Omega)$，定义如下形式：

$$a(u,v)=(\nabla u,\nabla v),\quad b_i(u,\phi,v)=(q^i u\nabla\phi,\nabla v),\quad \tilde{b}(u^1,u^2,v)=\Big(-\sum_{i=1}^{2}q^i u^i,v\Big).$$

设 $S_m^E(\cdot,\cdot)$和 $S_a^E(\cdot,\cdot)$为定义在 $Q_h^k(E)\times Q_h^k(E)$上的对称双线性形式[2]，满足：

$$\alpha_1(v_h,v_h)_E\leqslant S_m^E(v_h,v_h)\leqslant\alpha_2(v_h,v_h)_E,\quad \forall v_h\in Q_h^k(E),\quad \Pi_k^0 v_h=0,$$

$$\beta_1 a^E(v_h,v_h)\leqslant S_a^E(v_h,v_h)\leqslant\beta_2 a^E(v_h,v_h),\quad \forall v_h\in Q_h^k(E),\quad \Pi_k^\nabla v_h=0,$$

其中 $\alpha_i,\beta_i(i=1,2)$为正常数.

对 $\forall u_h,v_h,\phi_h,u_h^1,u_h^2\in Q_h^k$，定义

$$m_h(u_h,v_h):=\sum_E\Big(\int_E[\Pi_k^0 u_h]\cdot[\Pi_k^0 v_h]\mathrm{d}x+S_m^E((I-\Pi_k^0)u_h,(I-\Pi_k^0)v_h)_E\Big),$$

$$a_h(u_h,v_h):=\sum_E\Big(\int_E[\Pi_{k-1}^0\nabla u_h]\cdot[\Pi_{k-1}^0\nabla v_h]\mathrm{d}x+S_a^E((I-\Pi_k^\nabla)\,u_h,(I-\Pi_k^\nabla)v_h)\Big),$$

$$\tilde{a}_h(u_h,v_h):=\sum_E\Big(\int_E[\nabla\Pi_k^\nabla u_h]\cdot[\nabla\Pi_k^\nabla v_h]\mathrm{d}x+S_a^E((I-\Pi_k^\nabla)\,u_h,(I-\Pi_k^\nabla)v_h)\Big),$$

$$b_{i,h}(u_h,\phi_h,v_h):=\sum_E\int_E q^i[\Pi_{k-1}^0 u_h][\Pi_{k-1}^0\nabla\phi_h]\cdot[\Pi_{k-1}^0\nabla v_h]\mathrm{d}x,\quad i=1,2,$$

$$\tilde{b}_h(u_h^1,u_h^2,v_h):=-\sum_E\int_E\Big[\Pi_{k-1}^0\Big(\sum_{i=1}^{2}q^i u_h^i\Big)\Big][\Pi_{k-1}^0 v_h]\mathrm{d}x,$$

$$(F_{i,h},v):=\sum_E\int_E F_i\Pi_{k-1}^0 v_h\,\mathrm{d}x,\quad i=1,2,\quad (f_h,v):=\sum_E\int_E f\Pi_{k-1}^0 v_h\,\mathrm{d}x.$$

于是方程(5)-(6)对应的虚单元半离散格式为：求 $p_h^i\in L^2(0,T,Q_h^k)(i=1,2)$ 及 $\phi_h\in L^2(0,T,Q_h^k)$，满足：

$$\begin{cases} m_h(p_{h,t}^i,v_h)+a_h(p_h^i,v_h)+c_\lambda b_{i,h}(p_h^i,\phi_h,v_h)=(F_{i,h},v_h),\quad i=1,2,\quad \forall v_h\in Q_h^k,\quad t\in(0,T),\\ \tilde{a}_h(\phi_h,w_h)+\tilde{b}_h(p_h^1,p_h^2,w_h)=(f_h,w_h),\quad \forall w_h\in Q_h^k,\quad t\in(0,T). \end{cases}\tag{11}$$

下面给出方程(5)-(6)的虚单元全离散格式. 先将时间区间等距剖分为 $0=t^0<t^1<\cdots<t^N=T$，其中时间步长 $\tau=T/N$，$t^n=n\tau$，$n\in\mathbb{Z}$. 对 $\forall u_h\in Q_h^k$，记

$$u_h^n=u_h(\cdot,t^n),$$

则方程(5)-(6)对应的向后 Euler 全离散格式为：寻找 $p_h^{i,n}(i=1,2)$ 及 $\phi_h^n\in Q_h^k$，使得

$$\begin{cases} m_h\left(\dfrac{p_h^{i,n}-p_h^{i,n-1}}{\tau},v_h\right)+a_h(p_h^{i,n},v_h)+c_\lambda b_{i,h}(p_h^{i,n},\phi_h^n,v_h)=\\ \qquad (F_{i,h}^n,v_h),\quad i=1,2,\quad \forall v_h\in Q_h^k,\quad n=1,2,\cdots,N,\\ \tilde{a}_h(\phi_h^n,w_h)+\tilde{b}_h(p_h^{1,n},p_h^{2,n},w_h)=(f_h^n,w_h),\quad \forall w_h\in Q_h^k,\quad n=1,2,\cdots,N. \end{cases}\tag{12}$$

2.3 虚单元两水平算法

含时 PNP 方程的全离散格式(12)是一个耦合的非线性系统，通常采用 Gummel 迭代法对其进行解耦和线性化. 参考文献[2]，可给出式(12)的基于线性虚单元(即 $k=1$)的 Gummel 算法.

算法 1 Gummel 虚单元算法($k=1$).

输入：迭代初始值$(p_h^{1,0},p_h^{2,0},\phi_h^0)$，时间步长 τ，时间 t^n，停机标准 ϵ；

输出：$(p_h^{1,N},p_h^{2,N},\phi_h^N)$；

步骤 1) 当 $n\geqslant 1$ 时，令 $p_h^{i,n,0}=p_h^{i,n-1}$，$i=1,2$，$\phi_h^{n,0}=\phi_h^{n-1}$；

步骤 2) 对 $l\geqslant 1$，求$(p_h^{i,n,l},\phi_h^{n,l})\in[Q_h^1]^3$，$i=1,2$，使得对任意的 $v_h,\omega_h\in Q_h^1$，有

$$\frac{1}{\tau}(p_h^{i,n,l},v_h)+(\nabla p_h^{i,n,l},\nabla v_h)+(c_\lambda q^i p_h^{i,n,l}\nabla\phi_h^{n,l-1},\nabla v_h)=(F_i^n,v_h)+\frac{1}{\tau}(p_h^{i,n-1},v_h),\qquad i=1,2,$$

$$(\nabla\phi_h^{n,l},\nabla\omega_h)-\sum_{i=1}^2 q^i(p_h^{i,n,l},\omega_h)=(f^n,\omega_h);$$

步骤 3) 对给定的误差容限 ϵ，当

$$\|p_h^{1,n,l}-p_h^{1,n,l-1}\|+\|p_h^{2,n,l}-p_h^{2,n,l-1}\|+\|\phi_h^{n,l}-\phi_h^{n,l-1}\|\leqslant\epsilon$$

时即停止迭代，并令$(p_h^{1,n},p_h^{2,n},\phi_h^n)=(p_h^{1,n,l},p_h^{2,n,l},\phi_h^{n,l})$；否则，令 $l\leftarrow l+1$ 并返回步骤 2)继续非线性迭代；

步骤 4) 当 $n+1=N$ 时停止，否则令 $n\leftarrow n+1$ 并返回步骤 1).

对于稳态 PNP 方程离散系统的 Gummel 算法可参考文献[1]. 下面给出本文的主要算法，即虚单元两水平算法. 该算法的主要步骤是：首先在线性虚单元空间上求解原问题，然后利用线性元解对方程进行解耦及线性化，进而在二次虚单元空间上求解解耦后的方程得到两水平解. 方程(12)对应的虚单元两水平算法如下.

算法 2 虚单元两水平算法.

输入：迭代初始值$(p_h^{1,0},p_h^{2,0},\phi_h^0)$，时间步长 τ，时间 t^n；

输出：$(p_{h^*}^{1,N},p_{h^*}^{2,N},\phi_{h^*}^N)$；

步骤 1) 对 $n\geqslant 1$，已知 $p_h^{i,n-1}\in Q_h^1$，$i=1,2$，求$(p_h^{i,n},\phi_h^n)\in[Q_h^1]^3$，$i=1,2$，使得对 $\forall v_h,\omega_h\in Q_h^1$，满足

$$\frac{1}{\tau}(p_h^{i,n},v_h)+(\nabla p_h^{i,n},\nabla v_h)+(c_\lambda q^i p_h^{i,n}\nabla\phi_h^n,\nabla v_h)=(F_i^n,v_h)+\frac{1}{\tau}(p_h^{i,n-1},v_h),\quad i=1,2,$$

$$(\nabla\phi_h^n,\nabla\omega_h)-\sum_{i=1}^2 q^i(p_h^{i,n},\omega_h)=(f^n,\omega_h);$$

步骤 2) 已知 $p_{h^*}^{i,n-1}\in Q_h^2$，$i=1,2$，利用步骤 1)中所得的 ϕ_h^n，求$(p_{h^*}^{i,n},\phi_{h^*}^n)\in[Q_h^2]^3$，$i=1,2$，使得对 $\forall v_h,\omega_h\in Q_h^2$，满足

$$\frac{1}{\tau}(p_{h^*}^{i,n},v_h)+(\nabla p_{h^*}^{i,n},\nabla v_h)+(c_\lambda q^i p_{h^*}^{i,n}\nabla\phi_h^n,\nabla v_h)=(F_i^n,v_h)+\frac{1}{\tau}(p_{h^*}^{i,n-1},v_h),\quad i=1,2,$$

$$(\nabla\phi_{h^*}^n,\nabla\omega_h)-\sum_{i=1}^{2}q^i(p_{h^*}^{i,n},\omega_h)=(f^n,\omega_h);$$

步骤 3) 当 $n+1=N$ 时停止，否则令 $n\leftarrow n+1$ 并返回步骤 1).

对稳态 PNP 方程(3)的离散系统对应的虚单元两水平算法也可仿照算法 2 给出. 此外，算法 2 的误差估计可参考文献[7]中的两网格算法类似给出，其收敛阶通常比算法 1 高一阶.

3　数值实验

下面将虚单元两水平算法分别应用于二维稳态和含时 PNP 方程. 采用 Fortran90 语言编写计算程序，数值实验操作环境为个人电脑，配置为 CPU-1.80 GHz(AMD (R) Ryzen (TM) 7-4800U)，RAM-16 GB. 分别给出算法 1 的线性元解以及算法 2 的两水平解的误差结果，通过比较在达到相当精度时线性元解和两水平解所需的 CPU 时间，验证虚单元两水平算法的有效性.

数值结果中 H^1 和 L^2 模误差分别定义为

$$e_{H^1}=\sqrt{\sum_{E\in\mathcal{T}^h}\|\nabla(u-\Pi_k^\nabla u_h)\|_{0,E}^2},\qquad e_{L^2}=\sqrt{\sum_{E\in\mathcal{T}^h}\|u-\Pi_k^\nabla u_h\|_{0,E}^2},$$

其中 u 表示 ϕ，p^1 和 p^2 的精确解，u_h 表示虚单元解. 选取如下的收敛阶计算公式：

$$\text{收敛阶}=\frac{\log(e_{H^1,i+1}/e_{H^1,i})}{\log(h_{i+1}/h_i)},\qquad i=1,2,\cdots,$$

其中 h_i 表示网格尺寸，$e_{H^1,i}$ 表示在尺寸 h_i 的网格剖分下的 e_{H^1}，L^2 模误差的收敛阶可类似计算. 记 $|E|$ 为单元 E 的面积，网格尺寸定义为 $h_E:=\sqrt{|E|}$.

3.1　稳态 PNP 模型

考虑如下二维稳态 PNP 方程[9]：

$$\begin{cases}-\nabla\cdot(\nabla p^i+c_\lambda q^i p^i\nabla\phi)=F_i, & i=1,2,\ \text{在 }\Omega\text{ 内},\\ -\Delta\phi-\sum_{i=1}^{2}q^i p^i=f, & \text{在 }\Omega\text{ 内},\end{cases}\tag{13}$$

相应的边界条件为

$$\begin{cases}\phi=g_\phi, & \text{在 }\partial\Omega\text{ 上},\\ p^i=g_{p^i}, & i=1,2,\ \text{在 }\partial\Omega\text{ 上}.\end{cases}\tag{14}$$

右端 $F_i(i=1,2)$，f 及边界条件由如下定义的真解给出：

$$\begin{cases}\phi=\cos(\pi x)\cos(\pi y),\\ p^1=3\pi^2\left(1+\frac{1}{2}\cos(\pi x)\cos(\pi y)\right),\\ p^2=3\pi^2\left(1-\frac{1}{2}\cos(\pi x)\cos(\pi y)\right).\end{cases}$$

注意到方程(13)是半导体器件中的 PNP 方程经过无量纲化和规范化后的二维形式[9]. 原始求解域为 $\Omega=[-L/2,L/2]^2$，其中 L 表示半导体器件的尺寸. 经过无量纲化和规范化处理后，方程(13)的计算域变为 $\Omega=[-1/2,1/2]^2$，$c_\lambda=0.179L^2$. 若 $L=1$ nm，则 $c_\lambda=0.179$. 因此，对方程(13)的计算，选取 $\Omega=[-1/2,1/2]^2$，$c_\lambda=0.179$.

在计算中，采用一致三角形网格剖分. 表 1～表 3 分别列出了利用算法 1 和算法 2 求解方程(13)-(14)的误差结果. 表 1 和表 2 的结果表明：算法 1 的线性元解的 H^1 模误差达到 1 阶收敛阶；算法 2 的两水平解的 H^1 模误差达到 2 阶收敛阶. 说明两水平算法获得了更高的精度(这是因为两水平算法利用二次虚单元求解方程，因而精度更高). 此外，在达到相当精度的情况下，算法 2 所需的 CPU 时间显著低于算法 1. 例如，由表 1 和表 2 可见：当电势的线性元解达到精度 $\phi_h=5.46\times10^{-2}$

($h=1/64$)时，所需的 CPU 时间为 15.66 s；当电势的两水平解达到更高的精度 $\phi_{h^*}=3.26\times10^{-2}$ ($h=1/8$)时，所需的 CPU 时间仅为 1.20 s. 可见，算法 2 能显著提高求解效率，证明了虚单元两水平算法的有效性.

表 3 展示了稳态 PNP 方程两水平解的 L^2 模误差，其收敛阶达到三阶. 考虑到在文献[1]中已给出算法 1 的 L^2 模误差的二阶收敛结果，且对比分析结果与 H^1 模误差相似，因此本文不再列出. 图 1 为求解稳态 PNP 方程得到的线性元解和两水平解总 CPU 时间和 H^1 模误差的关系曲线. 由图 1 可见，算法 2 的计算效率远高于算法 1.

表 1 稳态 PNP 方程的精确解与线性元解的 H^1 模误差

Table 1 H^1 norm errors between exact solution and linear element solution for steady-state PNP equations

h	$\|\phi-\phi_h\|_1$	收敛阶	$\|p^1-p_h^1\|_1$	收敛阶	$\|p^2-p_h^2\|_1$	收敛阶	t_{CPU}/s
1/4	1.00×10^{0}	—	1.24×10^{1}	—	1.24×10^{1}	—	0.03
1/8	4.59×10^{-1}	1.13	6.40×10^{0}	0.96	6.39×10^{0}	0.96	0.20
1/16	2.21×10^{-1}	1.05	3.22×10^{0}	0.99	3.22×10^{0}	0.99	0.83
1/32	1.09×10^{-1}	1.02	1.61×10^{0}	1.00	1.61×10^{0}	1.00	3.42
1/64	5.46×10^{-2}	1.00	8.07×10^{-1}	1.00	8.07×10^{-1}	1.00	15.66

表 2 稳态 PNP 方程的精确解与两水平解的 H^1 模误差

Table 2 H^1 norm errors between exact solution and two-level solution for steady-state PNP equations

h	$\|\phi-\phi_{h^*}\|_1$	收敛阶	$\|p^1-p_h^{1*}\|_1$	收敛阶	$\|p^2-p_h^{2*}\|_1$	收敛阶	t_{CPU}/s
1/4	1.24×10^{-1}	—	1.84×10^{0}	—	1.82×10^{0}	—	0.30
1/8	3.26×10^{-2}	1.93	4.82×10^{-1}	1.94	4.80×10^{-1}	1.92	1.20
1/16	8.29×10^{-3}	1.97	1.23×10^{-1}	1.97	1.23×10^{-1}	1.97	5.06
1/32	2.08×10^{-3}	1.99	3.09×10^{-2}	1.99	3.09×10^{-2}	1.99	24.72
1/64	5.22×10^{-4}	2.00	7.73×10^{-3}	2.00	7.73×10^{-3}	2.00	134.39

表 3 稳态 PNP 方程的精确解与两水平解的 L^2 模误差

Table 3 L^2 norm errors between exact solution and two-level solution for steady-state PNP equations

h	$\|\phi-\phi_{h^*}\|_0$	收敛阶	$\|p^1-p_h^{1*}\|_0$	收敛阶	$\|p^2-p_h^{2*}\|_0$	收敛阶	t_{CPU}/s
1/4	5.17×10^{-3}	—	6.20×10^{-2}	—	5.94×10^{-2}	—	0.30
1/8	5.65×10^{-4}	3.19	7.79×10^{-3}	2.99	7.70×10^{-3}	2.95	1.20
1/16	6.74×10^{-5}	3.07	9.81×10^{-4}	2.99	9.78×10^{-4}	2.98	5.06
1/32	8.30×10^{-6}	3.02	1.23×10^{-4}	3.00	1.23×10^{-4}	2.99	24.72
1/64	1.06×10^{-6}	2.97	1.56×10^{-5}	2.98	1.55×10^{-5}	2.98	134.39

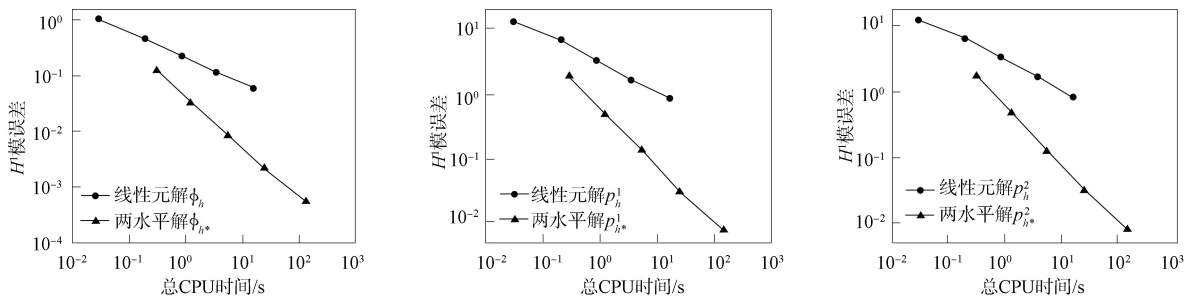

图 1 稳态 PNP 方程的总 CPU 时间和 H^1 模误差关系曲线

Fig. 1 Relationship curves between total CPU time and H^1 norm errors for steady-state PNP equations

3.2 含时 PNP 模型

考虑问题(1)及初边值条件：

$$\begin{cases} \phi=g_\phi, & \text{在 } \partial\Omega \text{ 上}, t\in(0,T], \\ p^i=g_{p^i}, \quad i=1,2, & \text{在 } \partial\Omega \text{ 上}, t\in(0,T], \\ p^i(x,0)=0, \quad i=1,2, & \text{在 } \Omega \text{ 内}. \end{cases} \tag{15}$$

计算域 Ω，c_λ 的取法与 3.1 节相同. 取时间 $T=0.25$，设置网格比为 $\tau=h^2$，右端 $F_i(i=1,2)$，f 及边界

条件可由下列定义的真解获得：

$$\begin{cases} \phi=(1-\mathrm{e}^{-t})\cos(\pi x)\cos(\pi y), \\ p^1=3\pi^2\sin(t)\left(1+\dfrac{1}{2}\cos(\pi x)\cos(\pi y)\right), \\ p^2=3\pi^2\sin(2t)\left(1-\dfrac{1}{2}\cos(\pi x)\cos(\pi y)\right). \end{cases}$$

分别用算法 1 和算法 2 求解含时 PNP 方程(1). 表 4～表 6 分别列出了其在最后一个时间层 $t=0.25$ 上的线性元解和两水平解的误差结果. 图 2 为含时 PNP 方程的总 CPU 时间和 H^1 模误差关系曲线.

表 4　含时 PNP 方程的精确解与线性元解的 H^1 模误差

Table 4　H^1 norm errors between exact solution and linear element solution for time-dependent PNP equations

h	$\|\phi-\phi_h\|_1$	收敛阶	$\|p^1-p_h^1\|_1$	收敛阶	$\|p^2-p_h^2\|_1$	收敛阶	t_{CPU}/s
1/4	2.70×10^{-1}	—	3.08×10^{0}	—	5.96×10^{0}	—	0.16
1/8	1.11×10^{-1}	1.28	1.58×10^{0}	0.96	3.07×10^{0}	0.96	1.75
1/16	5.03×10^{-2}	1.14	7.97×10^{-1}	0.99	1.54×10^{0}	0.99	22.55
1/32	2.44×10^{-2}	1.04	3.99×10^{-1}	1.00	7.73×10^{-1}	1.00	276.97
1/64	1.21×10^{-2}	1.01	2.00×10^{-1}	1.00	3.87×10^{-1}	1.00	4 446.38

表 5　含时 PNP 方程的精确解与两水平解的 H^1 模误差

Table 5　H^1 norm errors between exact solution and two-level solution for time-dependent PNP equations

h	$\|\phi-\phi_{h^*}\|_1$	收敛阶	$\|p^1-p_h^{1*}\|_1$	收敛阶	$\|p^2-p_h^{2*}\|_1$	收敛阶	t_{CPU}/s
1/4	2.77×10^{-2}	—	4.75×10^{-1}	—	9.30×10^{-1}	—	0.41
1/8	7.33×10^{-3}	1.92	1.22×10^{-1}	1.96	2.40×10^{-1}	1.95	4.05
1/16	1.87×10^{-3}	1.97	3.08×10^{-2}	1.99	6.05×10^{-2}	1.99	48.86
1/32	4.72×10^{-4}	1.99	7.72×10^{-3}	2.00	1.52×10^{-2}	1.99	576.52
1/64	1.18×10^{-4}	2.00	1.93×10^{-3}	2.00	3.79×10^{-3}	2.00	10 353.48

表 6　含时 PNP 方程的精确解与两水平解的 L^2 模误差

Table 6　L^2 norm errors between exact solution and two-level solution for time-dependent PNP equations

h	$\|\phi-\phi_{h^*}\|_0$	收敛阶	$\|p^1-p_h^{1*}\|_0$	收敛阶	$\|p^2-p_h^{2*}\|_0$	收敛阶	t_{CPU}/s
1/4	9.10×10^{-4}	—	2.17×10^{-2}	—	3.72×10^{-2}	—	0.41
1/8	2.99×10^{-4}	1.61	3.67×10^{-3}	2.56	9.64×10^{-3}	1.95	4.05
1/16	8.46×10^{-5}	1.82	7.62×10^{-4}	2.27	2.46×10^{-3}	1.97	48.86
1/32	2.31×10^{-5}	1.87	1.80×10^{-4}	2.08	6.17×10^{-4}	2.00	576.52
1/64	5.60×10^{-6}	2.04	4.41×10^{-5}	2.03	1.55×10^{-4}	1.99	10 353.48

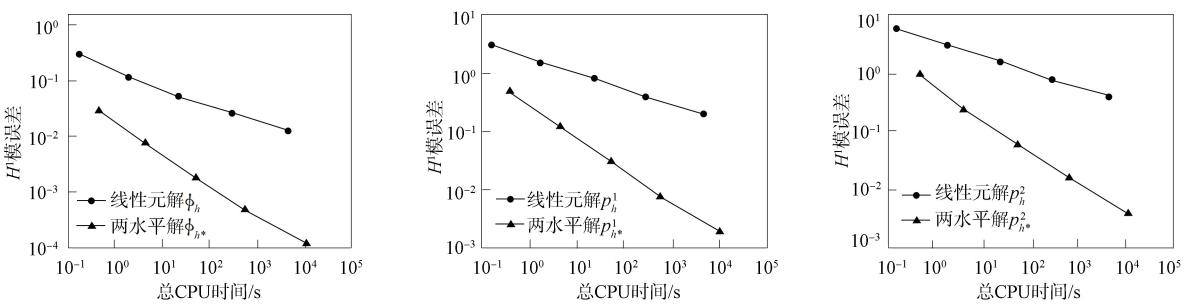

图 2　含时 PNP 方程的总 CPU 时间和 H^1 模误差关系曲线

Fig. 2　Relationship curves between total CPU time and H^1 norm errors for time-dependent PNP equations

由表 4 和表 5 及图 2 可见，算法 2 比算法 1 的精度更高，且在相当精度下，两水平解所需 CPU 时间远小于线性元解所需的 CPU 时间. 如当电势的线性元解 $\phi_h=1.21\times10^{-2}(h=1/64)$时，所需 CPU 时间为 4 446.38 s，而当电势的两水平解在达到更高的精度 $\phi_{h^*}=7.33\times10^{-3}(h=1/8)$时，所需 CPU 时间仅为 4.05 s. 表明了两水平算法求解含时 PNP 方程(1)的有效性.

表6结果表明，两水平解的 L^2 模误差收敛阶为二阶(若将网格比调整为 $\tau=h^3$，L^2 模误差能达到更高的精度，其收敛阶达到三阶). 与线性元解的 L^2 模误差[2]相比，算法2所需的CPU时间更少，效率更高. 该问题的 L^2 模误差需要更高的网格比才能达到三阶，这可能与该问题的特殊性有关(这是半导体器件中的一个BenchMark模型)，具体原因需进一步探讨.

综上所述，针对稳态和含时两种PNP方程，本文提出了一种虚单元两水平算法. 该算法先利用线性虚单元解对方程进行解耦和线性化，然后在二次虚单元空间上求解解耦后的问题. 数值实验结果表明，在达到相当的数值精度下，两水平算法所需的CPU时间显著低于常用的线性虚单元的Gummel算法，表明了两水平算法求解PNP方程的有效性. 由于两水平算法只需要生成一套网格，因此对实际问题有很强的适用性.

参 考 文 献

[1] 刘杨. 二维稳态Poisson-Nernst-Planck方程的一种虚单元方法[D]. 湘潭：湘潭大学，2020.(LIU Y. A Virtual Element Method for Two-Dimensional Steady-State Poisson-Nernst-Planck Equations [D]. Xiangtan: Xiangtan University, 2020.)

[2] YANG Y, LIU Y, LIU Y, et al. Error Analysis of Virtual Element Methods for the Time-Dependent Poisson-Nernst-Planck Equations [J]. Journal of Computational Mathematics, 2025, 43(3): 731-770.

[3] BURGER M, PINNAU R. A Globally Convergent Gummel Map for Optimal Dopant Profiling [J]. Mathematical Models and Methods in Applied Sciences, 2009, 19(5): 769-786.

[4] JEROME J W, BROSOWSKI B. Evolution Systems in Semiconductor Device Modeling: A Cyclic Uncoupled Line Analysis for the Gummel Map [J]. Mathematical Methods in the Applied Sciences, 1987, 9(1): 455-492.

[5] LU B Z, ZHOU Y C. Poisson-Nernst-Planck Equations for Simulating Biomolecular Diffusion-Reaction Processes Ⅱ: Size Effects on Ionic Distributions and Diffusion-Reaction Rates [J]. Biophysical Journal, 2011, 100(10): 2475-2485.

[6] YANG Y, LU B Z, XIE Y. A Decoupling Two-Grid Method for the Steady-State Poisson-Nernst-Planck Equations [J]. Journal of Computational Mathematics, 2019, 37(4): 556-578.

[7] SHEN R G, SHU S, YANG Y, et al. A Decoupling Two-Grid Method for the Time-Dependent Poisson-Nernst-Planck Equations [J]. Numerical Algorithms, 2020, 83(4): 1613-1651.

[8] TANG M, XING X Q, YANG Y, et al. Iterative Two-Level Algorithm for Nonsymmetric or Indefinite Elliptic Problems [J]. Applied Mathematics Letters, 2023, 140: 108594-1-108594-5.

[9] 倪宇晖，阳莺. 一类Poisson-Nernst-Planck方程的边平均有限元计算[J]. 吉林大学学报(理学版)，2022，60(1)：73-78.(NI Y H, YANG Y. Edge-Averaged Finite Element Calculation for a Class of Poisson-Nernst-Planck Equations [J]. Journal of Jilin University (Science Edition), 2022, 60(1): 73-78.)

[10] BEIRÃO DA VEIGA L, BREZZI F, MARINI L D, et al. Virtual Element Method for General Second-Order Elliptic Problems on Polygonal Meshes [J]. Mathematical Models and Methods in Applied Sciences, 2016, 26(4): 729-750.

[11] BEIRÃO DA VEIGA L, BREZZI F, MARINI L D, et al. The Hitchhiker's Guide to the Virtual Element Method [J]. Mathematical Models and Methods in Applied Sciences, 2014, 24(8): 1541-1573.

(责任编辑：赵立芹)