计算物理作业 8

谢昀城 22307110070

2024 年 11 月 28 日

1 题目 1

1.1 题目描述

Consider the Poisson equation:

$$\nabla^2\phi(x,y) = -\frac{\rho(x,y)}{\varepsilon_0}$$

from electrostatics on a rectangular geometry with $x \in [0, L_x]$ and $y \in [0, L_y]$. Write a program that solves this equation using the relaxation method. Test your program with:

(a) $\rho(x,y) = 0$, $\phi(0,y) = \phi(L_x,y) = \phi(x,0) = 0$, $\phi(x,L_y) = 1\,\mathrm{V}$, $L_x = 1\,\mathrm{m}$, and $L_y = 1.5\,\mathrm{m}$;

(b) $\rho(x,y)/\varepsilon_0 = 1\,\mathrm{V/m^2}$, $\phi(0,y) = \phi(L_x,y) = \phi(x,0) = \phi(x,L_y) = 0$, and $L_x = L_y = 1\,\mathrm{m}$.

1.2 程序描述

在本程序中，我们使用 jacobi 算法求解方形区域上的泊松方程。程序将根据迭代前后函数值的 L_∞ 范数来判断是否收敛。

$$V_{i,j}^{n+1} = \frac{1}{2+2\alpha}\left[V_{i+1,j}^n + V_{i-1,j}^n + \alpha(V_{i,j+1}^n + V_{i,j-1}^n) + \Delta x^2 \rho_{i,j}\right], \quad n = 0,1,2,\ldots, \alpha = \frac{\Delta x^2}{\Delta y^2}$$

本程序源文件为 solvePoisson.py，在终端进入当前目录，使用命令 python -u solvePoisson.py 运行本程序。运行时请保证 Python 第三方库 Numpy,Matplotlib 已安装。程序开发环境为 Python3.12.3，可在 Python3.8 以上版本中运行。

1.3 伪代码

求解泊松方程伪代码:

Algorithm 1 PoissonSolver

Class PoissonSolver

Attributes: *upbound*, *downbound*, $leftbound$, *rightbound*, *rho*, Lx, Ly, Nx, Ny, dx, dy, ap, u, X, Y, *rhol*

Method

function InitGrid

$xl \leftarrow$ linspace(0, Lx, Nx)

$yl \leftarrow$ linspace(0, Ly, Ny)

$u \leftarrow$ zeros((Nx, Ny))

$X, Y \leftarrow$ meshgrid(xl, yl)

ApplyBoundary(u)

end function

function BoundaryPos

OUTPUT: bu, bd, bl, br (boolean 2D arrays)

$bu \leftarrow$ zeros((Nx, Ny), dtype=bool)

$bd \leftarrow$ zeros((Nx, Ny), dtype=bool)

$bl \leftarrow$ zeros((Nx, Ny), dtype=bool)

$br \leftarrow$ zeros((Nx, Ny), dtype=bool)

$bu[-1, :] \leftarrow 1$

$bd[1, :] \leftarrow 1$

$bl[:, 1] \leftarrow 1$

$br[:, -1] \leftarrow 1$

return bu, bd, bl, br

end function

function ApplyBoundary(u)

$bu, bd, bl, br \leftarrow$ BoundaryPos()

$u[bu] \leftarrow upbound(X[bu])$

$u[bd] \leftarrow downbound(X[bd])$

$u[bl] \leftarrow leftbound(Y[bl])$

$u[br] \leftarrow rightbound(Y[br])$

end function

function UpdateU(u)

$self.u \leftarrow u$

ApplyBoundary(u)

end function

function JacobiStep

OUTPUT: u (updated 2D array of u)

$u \leftarrow$ zerosLike(u)

$u[1 : -1, 1 : -1] \leftarrow$

$(u[: -2, 1 : -1] + u[2 :, 1 : -1] + ap \cdot (u[1 : -1, : -2] + u[1 : -1, 2 :]) + dx^2 \cdot rhol[1 : -1, 1 : -1])/2/(1 + ap)$

UpdateU(u)

return u

end function

function Jacobi($maxIter$, tol)

INPUT: $maxIter$ (integer), tol (float)

OUTPUT: u (2D array solution of Poisson equation)

$hisig \leftarrow$ copy(u)

for $i \leftarrow 0$ **To** $maxIter$ **do**

JacobiStep()

if max(abs($u - hisig$)) $< tol \cdot dx^2$ **then**

Print: "Jacobi converged after i iterations with relative tolerance tol"

Break

end if

$hisig \leftarrow \text{copy}(u)$

else for

Print: "Jacobi did not converge after $maxIter$ iterations with relative tolerance tol"

end for

return u

end function

1.4 输入输出实例

对于本程序，运行后会生成图1为"solvePoisson.png" 于当前目录下, 分别为两种情况的求解结果。程序运行截图如图2所示。

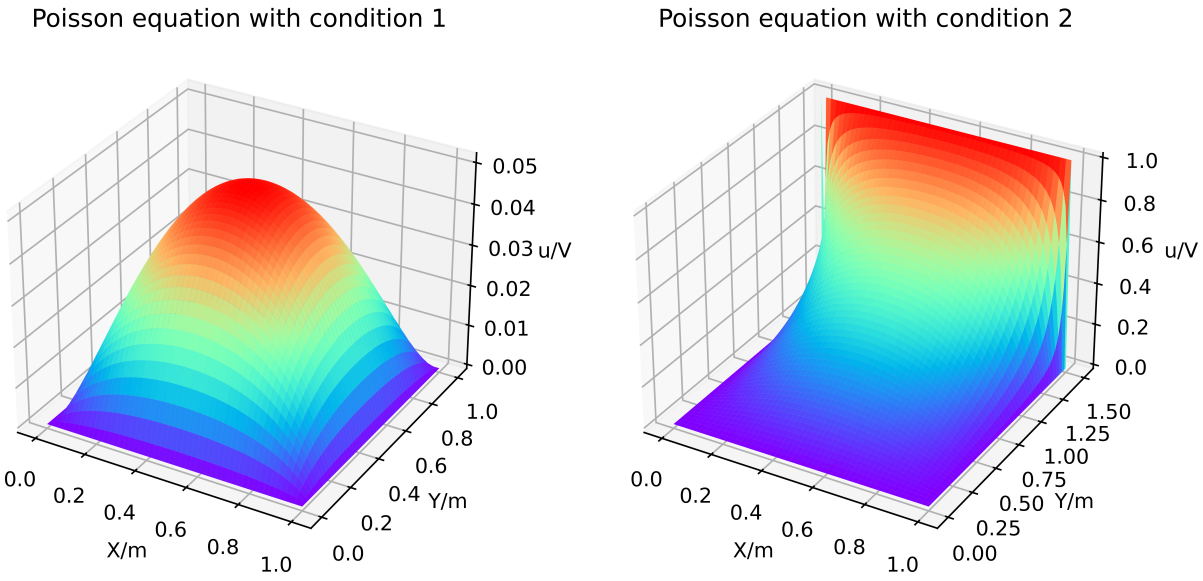

图 1: 泊松方程求解结果

```
(base) PS C:\Users\ASUS\Desktop\计算物理基础\hw8> python -u .\solvePoisson.py
Solving Poisson equation with Lx=1,Ly=1.5,Nx=100,Ny=100
Jacobi converged after 2718 iterations with relative tolerance 0.1
Solving Poisson equation with Lx=1,Ly=1.5,Nx=100,Ny=100
Jacobi converged after 6541 iterations with relative tolerance 0.1
```

图 2: 题目 1 程序运行截图

2 题目 2

2.1 题目描述

Solve the time-dependent Schrödinger equation using both the Crank‐Nicolson scheme and a stable explicit scheme. Consider the one-dimensional case and test it by applying it to the problem of a square well with a Gaussian initial state coming in from the left.

Hint: The Gaussian initial state could be expressed as:

$$\Psi(x,0)=\sqrt{\frac{1}{\pi}}\exp\left[ik_0x-\frac{(x-\xi_0)^2}{2}\right]$$

2.2 程序描述

本程序将使用 Crank-Nicolson 方法和稳定的显式方法求解一维含时薛定谔方程。为简便，我们取 $\hbar=m=1$

$$-\frac{1}{2}\frac{\partial^2\Psi(x,t)}{\partial x^2}+V(x)\Psi(x,t)=i\frac{\partial\Psi(x,t)}{\partial t}$$

其中

$$V(x)=\begin{cases}0, & -a\le x\le a\\ V_0, & x<-a \text{ 或} x>a\end{cases}$$

本题中我们取 $a=2.0, V_0=-1.0, \xi_0=-5.0, k_0=1.0$，并取空间区间为 $[-20,20]$，在边界处近似认为有边界条件 $\Psi(-20)=\Psi(20)=0$。

做离散化 $(\frac{\partial^2\Psi(x,t)}{\partial x^2})^n=\frac{\Psi_{i+1}^n-2\Psi_i^n+\Psi_{i-1}^n}{\Delta x^2}$,$(\frac{\partial\Psi(x,t)}{\partial t})^n=\frac{\Psi_{i+1}^n-\Psi_i^n}{\Delta t}$

考虑:

$$\theta(-\frac{1}{2}\frac{\partial^2\Psi}{\partial x^2}+V\Psi)^n+(1-\theta)(-\frac{1}{2}\frac{\partial^2\Psi}{\partial x^2}+V\Psi)^{n+1}=c$$

得到差分格式：

$$\Psi^{n+1}=(2iI-\theta V-\alpha\theta L)^{-1}(2iI+(1-\theta)V+(1-\theta)\alpha L)\Psi^n$$

其中 $\alpha=\frac{\Delta t}{\Delta x^2}, V_{ii}=V(x_i)$

$$L=\begin{pmatrix}2&-1&0&0&\cdots&0\\-1&2&-1&0&\cdots&0\\0&-1&2&-1&\cdots&0\\0&0&-1&2&\cdots&0\\\vdots&\vdots&\vdots&\vdots&\ddots&\vdots\\0&0&0&0&\cdots&2\end{pmatrix}$$

取 $\theta=\frac{1}{2}$，为 Crank‐Nicolson 格式，取 $\theta=0$ 为 Explicit 格式。

若取 $(\frac{\partial\Psi(x,t)}{\partial t})^n=\frac{\Psi_{i+1}^{n+1}-\Psi_i^{n-1}}{\Delta t}$, 考虑:

$$(-\frac{1}{2}\frac{\partial^2\Psi}{\partial x^2}+V\Psi)^n=(\frac{\partial\Psi(x,t)}{\partial t})^n$$

则得到 Stable Explicit 格式：

$$\Psi^n=\Psi^{n-1}-i\alpha L\psi^n-2iVdt\Psi^n$$

本程序源文件为 solveSchrodinger.py，在终端进入当前目录，使用命令 python -u solveSchrodinger.py 运行本程序。使用 python -u solveSchrodinger.py –test_convergence SE 或 python -u solveSchrodinger.py –test_convergence CN 运行收敛性测试。运行时请保证 Python 第三方库 Numpy,Matplotlib 已安装。程序开发环境为 Python3.12.3，可在 Python3.8 以上版本中运行。

2.3 伪代码

Crank-Nicolson 和 Stable Explicit 格式求解伪代码

Algorithm 2 SchrodingerSolver

Class $SchrodingerSolver(Lx, Nx, V, dt)$ ▷ Solve Schrodinger equation

$V, Lx, Nx, dx, dt, xl, u, Bmat, Vmat, Acn$ ▷ Class attributes

Method

function INITGRID

$xl \leftarrow \text{linspace}(-Lx, Lx, Nx)$

$u \leftarrow \text{zerosLike}(xl, \text{dtype=complex})$

$\text{applyBoundary}(u)$

end function

function APPLYBOUNDARY(u)

$u[0], u[-1] \leftarrow 0.0 + 0.0j$

end function

function SETINITIALCONDITION(u0)

$u \leftarrow u0(xl)$

$\text{applyBoundary}(u)$

end function

function UPDATEU(u)

$u \leftarrow u$

$\text{applyBoundary}(u)$

end function

function INITCRANKNICOLSONMAT(theta)

INPUT: *theta* (float, default 0.5)

$ap \leftarrow \frac{dt}{dx^2}$

$B \leftarrow \text{zeros}(Nx, Nx)$

$V \leftarrow \text{zeros}(Nx, Nx)$

$I \leftarrow \text{eye}(Nx)$

for $i \leftarrow 0$ **To** $Nx - 1$ **do**

$B[i, i] \leftarrow 2$

$V[i, i] \leftarrow V(xl[i]) \cdot dt$

else for $i < Nx - 1$

$B[i, i + 1] \leftarrow -1$

$B[i + 1, i] \leftarrow -1$

end for

$A \leftarrow \text{inv}(I \cdot 2j - 2 \cdot theta \cdot V - theta \cdot ap \cdot B)$ @
$(I \cdot 2j + 2 \cdot (1 - theta) \cdot V + (1 - theta) \cdot ap \cdot B)$
$Acn \leftarrow A$
end function
function CRANKNICOLSON(maxStep)
INPUT: $maxStep$ (integer, number of iterations)
$ul \leftarrow$ emptyList()
for $i \leftarrow 0$ **To** $maxStep - 1$ **do**
$phi2 \leftarrow \text{abs}(u)^2$
append(ul, u)
updateU$(Acn \cdot u)$
end for
return array(ul)
end function
function STABLEEXPLICIT(maxStep)
INPUT: $maxStep$ (integer, number of iterations)
$ap \leftarrow \frac{dt}{dx^2}$
initCrankNicolsonMat$(\theta = 0.5)$
$ul \leftarrow [u]$
updateU$(Acn \cdot u)$
append(ul, u)
for $i \leftarrow 0$ **To** $maxStep - 1$ **do**
$phi2 \leftarrow \text{abs}(u)^2$
append(ul, u)
updateU$(ul[-2] - 1j \cdot ap \cdot Bmat \cdot u - 2j \cdot dt \cdot Vmat \cdot u)$
end for
return array(ul)
end function

2.4 输入输出实例

本程序中使用 python -u solveSchrodinger.py 运行后将会生成图3至图6于当前目录下, 分别为不同参数下 (α 从 0.06 到 0.51),Crank-Nicolson，Stable Explicit,Explicit 的求解结果。程序运行截图如图7所示。

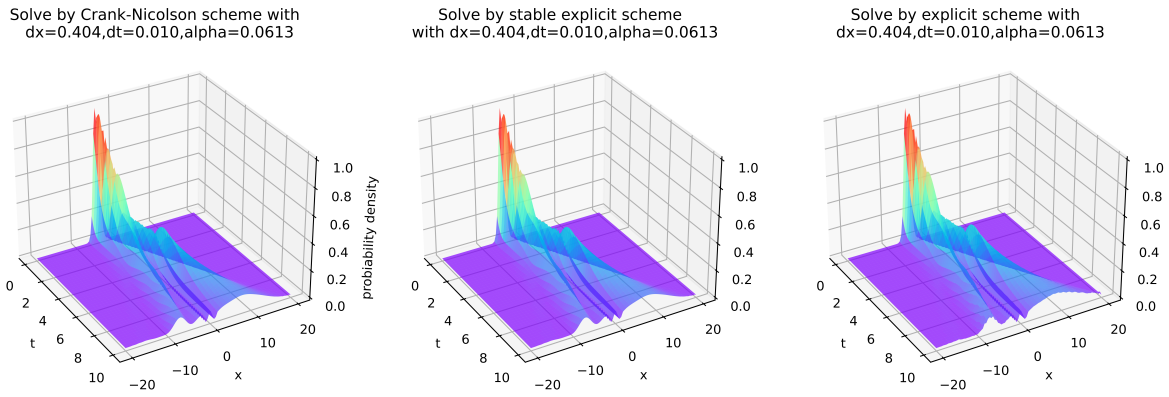

图 3: 求解结果 $\alpha = 0.0613$

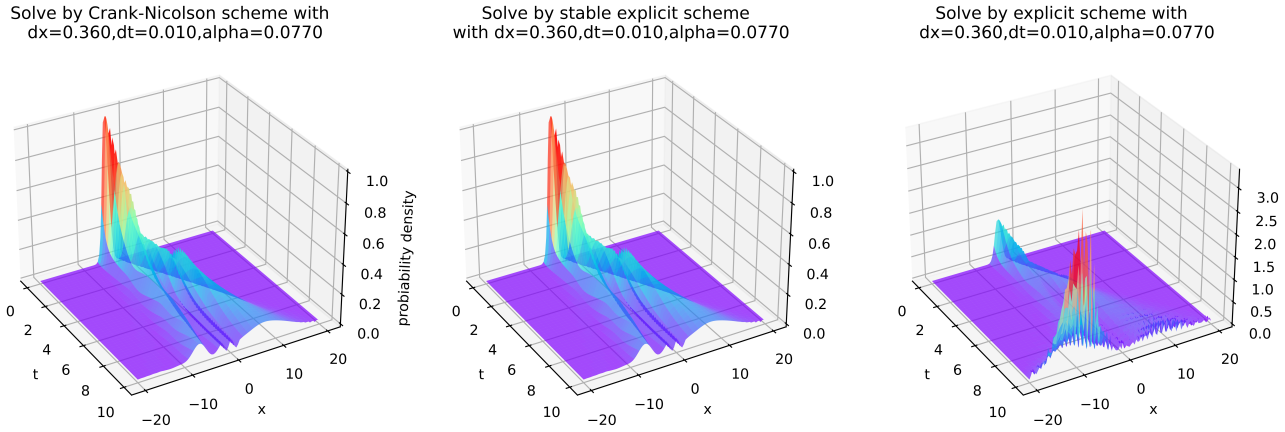

图 4: 求解结果 $\alpha = 0.0770$

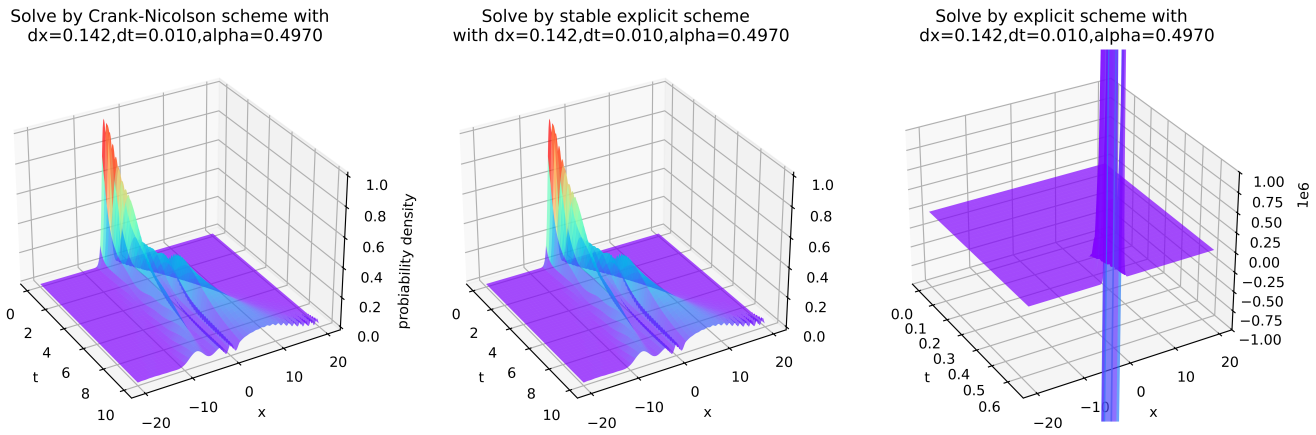

图 5: 求解结果 $\alpha = 0.4970$

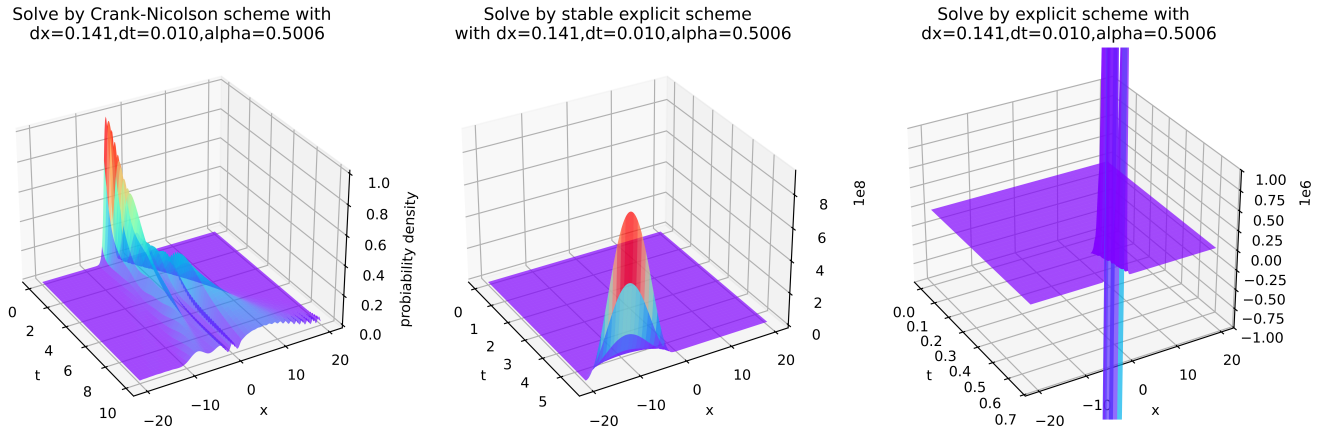

图 6: 求解结果 $\alpha = 0.5006$

```
(base) PS C:\Users\ASUS\Desktop\计算物理基础\hw8> python -u .\solveSchrodinger.py --test_convergence SE --test_convergence CN
Solving Schrodinger equation with square well potential with Lx=20.0,Nx=100,dt=0.01,maxstep=1000
Save figure to solvebyCrankNicolson_Nx=100_dt=0.010.png
Solving Schrodinger equation with square well potential with Lx=20.0,Nx=112,dt=0.01,maxstep=1000
Save figure to solvebyCrankNicolson_Nx=112_dt=0.010.png
Solving Schrodinger equation with square well potential with Lx=20.0,Nx=283,dt=0.01,maxstep=1000
Crank_Nicolson Method :Iteration stopped at step 66 due to overflow or invalid values.
Save figure to solvebyCrankNicolson_Nx=283_dt=0.010.png
Solving Schrodinger equation with square well potential with Lx=20.0,Nx=284,dt=0.01,maxstep=1000
Crank_Nicolson Method :Iteration stopped at step 67 due to overflow or invalid values.
Stable Explicit Method:Iteration stopped at step 559 due to overflow or invalid values.
Save figure to solvebyCrankNicolson_Nx=284_dt=0.010.png

Test convergence of stable explicit method with parameters:Lx=20.0,dt=0.01,Vr=120.0,apb=0.01,ape=1.2,x0=5.0,k0=-1.0,maxstep=50,umax=100.0,NV=40,Na=40
This will take a few minutes


Test convergence of Crank-Nicolson method with parameters:theta=0.0,Lx=20.0,dt=0.01,Vr=150.0,apb=0.01,ape=1.2,x0=5.0,k0=-1.0,maxstep=50,umax=100.0,NV=40,Na=40
This will take a few minutes


Test convergence of Crank-Nicolson method with parameters:theta=0.1,Lx=20.0,dt=0.01,Vr=150.0,apb=0.01,ape=1.2,x0=5.0,k0=-1.0,maxstep=50,umax=100.0,NV=40,Na=40
This will take a few minutes


Test convergence of Crank-Nicolson method with parameters:theta=0.3,Lx=20.0,dt=0.01,Vr=150.0,apb=0.01,ape=1.2,x0=5.0,k0=-1.0,maxstep=50,umax=100.0,NV=40,Na=40
This will take a few minutes


Test convergence of Crank-Nicolson method with parameters:theta=0.5,Lx=20.0,dt=0.01,Vr=150.0,apb=0.01,ape=1.2,x0=5.0,k0=-1.0,maxstep=50,umax=100.0,NV=40,Na=40
This will take a few minutes
```

图 7: 题目 2 程序运行截图

2.5 讨论

根据图3至图6，我们可以看到，Crank-Nicolson 格式始终保持稳定，在大约 $\alpha > 0.0613$ 后,Explicit 格式开始不稳定 (图3-4)，这说明对于普通的 Explicit 格式对于求解含时薛定谔方程并不适合。

另外我们注意到在 $\alpha > 0.5$ 后，Stable Explici 格式也开始不稳定 (5-6)。这与 PPT 上给出的参考文献 (J. Chem. Phys. 68, 2794 (1978)) 中的描述不符。其中作者宣称:"*The stability of the proposed explicit scheme is seen to be governed by the same criterion as the Crank-NichOlson method.*"(图10)。我检查了其中对于稳定性的推导，发现其存在错误，正确过程如下:

采用 von Neumann 分析，令 $\psi_j^n = \xi^n e^{ijK}$, 带入差分格式得到:

$$\xi^2 - 2i(\alpha coskx - \alpha - V_j\Delta t)\xi = 1 = 0$$

解为:

$$\xi_{1,2} = -i[\alpha coskx - \alpha - V_j\Delta t] \pm \sqrt{-(\alpha coskx - \alpha - V_j\Delta t)^2 + 1}$$

论文中认为 $|\xi_{1,2}| = 1$，因此该差分格式是恒稳定的。

但事实上，$|\xi_{1,2}| = 1$ 要求第二项为实数，即 $-(\alpha coskx - \alpha - V_j\Delta t)^2 + 1 = 1 - \gamma^2 < 0 >= 0$。若 $1 - \gamma^2 < 0$，则 $\xi_{1,2} = -i(\gamma \pm \sqrt{\gamma^2 - 1}),\gamma$. 这时 $|\xi_2| = \gamma + \sqrt{\gamma^2 - 1} > 0$ 格式不稳定。

因此，Stable Explicit 格式稳定的条件应是:

$$(\alpha coskx - \alpha - V_j\Delta t)^2 < 1 \quad for \quad any \quad V_j \quad and \quad kx$$

由于 $\alpha > 0, coskx\epsilon[-1, 1]$, 对有限方势阱:$V_j = V_0 or 0$, 我们可以得到稳定的区域为由以下不等式围成的区域:

$$\begin{cases} -1 < 2\alpha + V_0 dt < 1, \\ -1 < V_0 dt < 1, \\ 0 < \alpha < 1 \end{cases}$$

我们改变 V_0 和 α 测试在不同参数下求解的稳定性，得到稳定区域如图8, 可以看到其与上式给出的范围一致。

我们也测试了不同 θ 下，类 Crack-Nicolson 格式的稳定性，得到不同 θ 稳定区域如图9。可以看到随着 θ 增大，稳定的参数区域逐渐增大，直到 $\theta = 0.5$ 时，对任何参数都稳定。

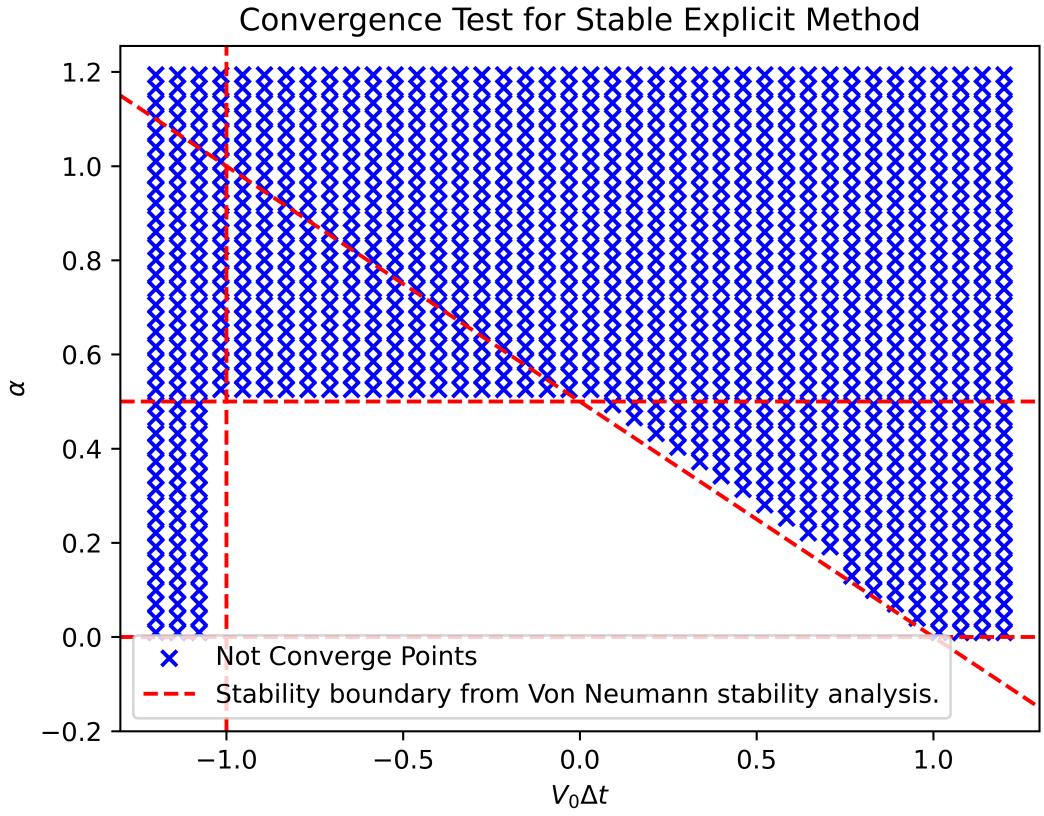

图 8: 不同 $\alpha, V_0\Delta t$Stable Explicit 格式的稳定性

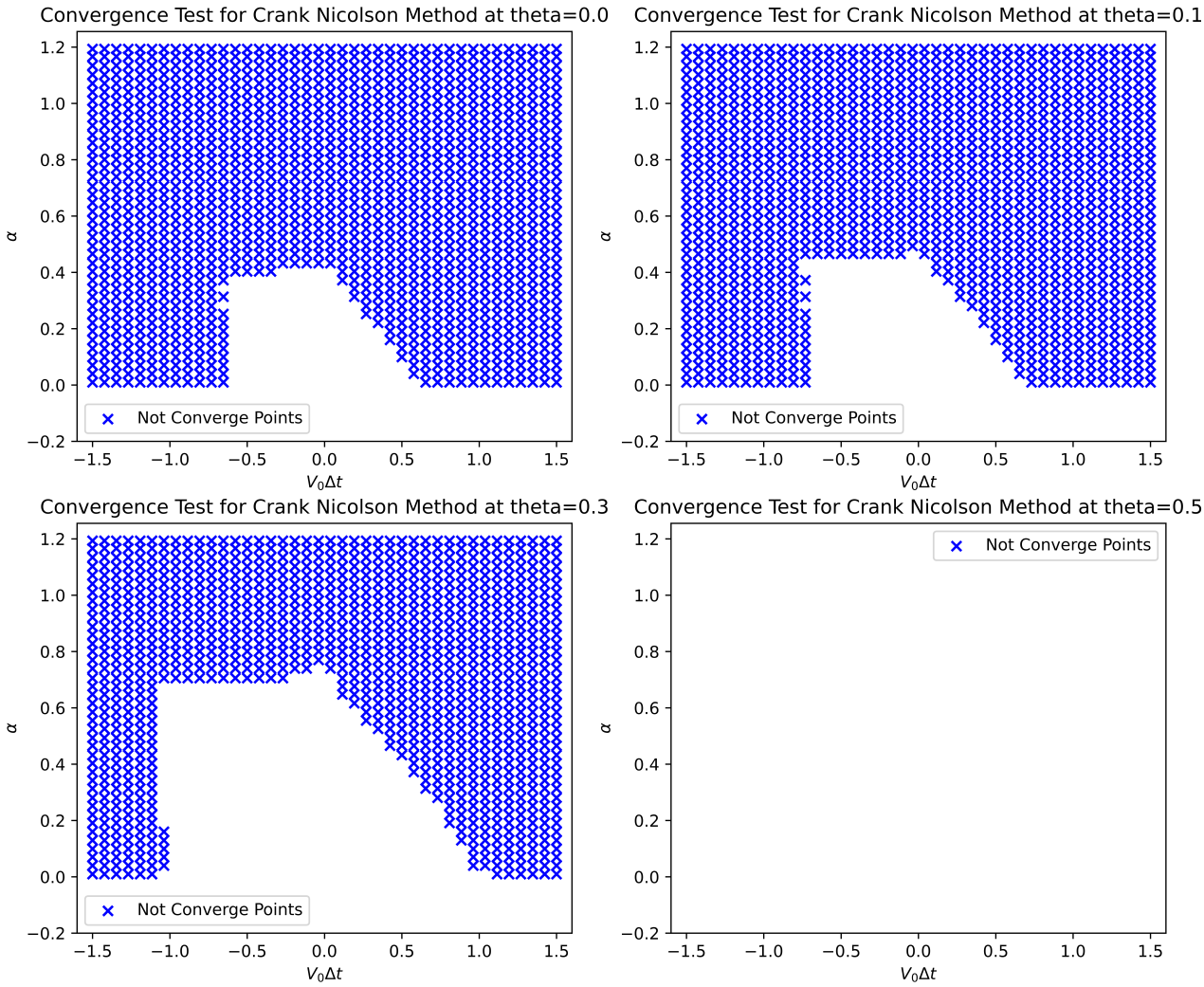

图 9: 不同 $\theta, \alpha, V_0\Delta t$ 类 Crank Nicolson 格式的稳定性

The stability analysis of this scheme along the same steps as above yields the following equation for the growth factor g

$$g^2 + 2i[2\alpha(1 - \cos q\Delta x) + V_j\Delta t]g - 1 = 0 \ . \qquad (3.5)$$

Two growth factors g_1 and g_2 are obtained from the quadratic equation above as

$$g_{1,2} = -i[2\alpha(1 - \cos q\Delta x) + V_j\Delta t] \pm \{1 - [2\alpha(1 - \cos q\Delta x) + V_j\Delta t]^2\}^{1/2} \ . \qquad (3.6)$$

Consequently, just as for the implicit Crank–Nicholson method, one has for both g_1 and g_2

$$|g_1|^2 = |g_2|^2 = 1 \ . \qquad (3.7)$$

The stability of the proposed explicit scheme is seen to be governed by the same criterion as the Crank–Nicholson method. As for the accuracy, Taylor series

图 10: 求解结果,α

3 题目 3

3.1 题目描述

Prove the stability condition of the explicit scheme of the 1D wave equation by performing Von Neumann stability analysis:

$$\frac{\partial^2 u}{\partial t^2} = c^2 \frac{\partial^2 u}{\partial x^2}.$$

If $\frac{c\Delta t}{\Delta x} \leq 1$, the explicit scheme is stable.

3.2 证明

做离散化：

$$\left(\frac{\partial^2 u(x,t)}{\partial x^2}\right)^n = \frac{u_{j+1}^n - 2u_j^n + u_{j-1}^n}{\Delta x^2}, \quad \left(\frac{\partial^2 u(x,t)}{\partial t^2}\right)^n = \frac{u_j^{n+1} - 2u_j^n + u_j^{n-1}}{\Delta t^2}$$

得到：

$$u_j^{n+1} = 2u_j^n - u_j^{n-1} + \alpha^2\left(u_{j+1}^n - 2u_j^n + u_{j-1}^n\right),$$

其中，$\alpha = \frac{c^2\Delta t^2}{\Delta x^2}$。

采用 von Neumann 分析，令 $\psi_j^n = \xi^n e^{ijK}$, 带入差分格式得到:

$$\xi^2 - 2\beta\xi + 1 = 0, \beta = 1 + \alpha^2(coskx - 1)$$

解为:$\xi_{1,2} = \beta \pm \sqrt{\beta^2 - 1}$

当 $\beta > 1$，$|\xi_2| = \beta + \sqrt{\beta^2 - 1} > 1$, 当 $\beta < -1$,$|\xi_1| = -\beta + \sqrt{\beta^2 - 1} > 1$，不稳定。当 $\beta^2 - 1 < 0$,$|\xi_{1,2}|^2 = 1$, 差分格式稳定。

因此要求 $\beta = 1 + \alpha^2(coskx - 1)\epsilon[-1, 1]$。而 $coskx\epsilon[-1, 1]$, 因此 $\alpha^2\epsilon[0, 1]$, 即 $\frac{c\Delta t}{\Delta x} \leq 1$ 时，差分格式稳定。