Computational Physics Lab

Homework 4

108000204
Yuan-Yen Peng
Dept. of Physics, NTHU
Hsinchu, Taiwan

December 29, 2022

1 Writing Assignments

1.1 Finite difference method for solving 2D Poisson equation

The 3×3 grids with grid space h and the source term $\rho(x, y)$ can be described as:

$$u_{xx} + u_{yy} = \rho(x, y)$$

where the subscripts mean second partial derivatives of the u with x and y, respectively. Besides, we annotate (x, y) with (i, j) where i and j are from $0 \sim (3 + 1)$ (including boundaries, which are $(i, j) = (i, 0),\ (i, 4),\ (0, j)$, and $(4, j)$), and implement the Euler method for derivative; then we can get:

$$\begin{aligned}
&\frac{\partial}{\partial x}\left(\frac{u_{i,j} - u_{i-1,j}}{h}\right) + \frac{\partial}{\partial y}\left(\frac{u_{i,j} - u_{i,j-1}}{h}\right) = \rho(x, y) \\
\Rightarrow &\frac{1}{h}\left(\frac{u_{i+1,j} - u_{i,j}}{h} - \frac{u_{i,j} - u_{i-1,j}}{h}\right) + \frac{1}{h}\left(\frac{u_{i,j+1} - u_{i,j}}{h} - \frac{u_{i,j} - u_{i,j-1}}{h}\right) = \rho(x, y) \\
\Rightarrow &\left(u_{i+1,j} - 2u_{i,j} + u_{i-1,j}\right) + \left(u_{i,j+1} - 2u_{i,j} + u_{i,j-1}\right) = \rho(x, y)h^2 \\
\Rightarrow &\ 4_{i,j} - u_{i-1,j} - u_{i+1,j} - u_{i,j-1} - u_{i,j+1} = \rho(x, y)h^2 \qquad \text{(I)}
\end{aligned}$$

In the wake of knowing the general form of the solution, we apply the boundary conditions; here boundaries are all zeros (i.e., $u_{0,j},\ u_{i,0},\ u_{4,j},$ and $u_{i,4} = 0$). Exploiting all i and j, we can get LHS of eq.(I) in the matrix form $\mathbf{A}$ and RHS in the vector form $\mathbf{b}$ with source term g_{ij} which represents the splitting of the source function $\rho(x, y)$:

$$\mathbf{A} = \begin{bmatrix}
\begin{bmatrix} 4 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 4 \end{bmatrix} & \begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{bmatrix} & \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \\
\begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{bmatrix} & \begin{bmatrix} 4 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 4 \end{bmatrix} & \begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{bmatrix} \\
\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} & \begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{bmatrix} & \begin{bmatrix} 4 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 4 \end{bmatrix}
\end{bmatrix}$$

and

$$\mathbf{b} = -h^2 \begin{bmatrix} g_{11} \\ g_{12} \\ g_{13} \\ \\ g_{21} \\ g_{22} \\ g_{23} \\ \\ g_{31} \\ g_{32} \\ g_{33} \end{bmatrix}$$

2 Programming Assignments

2.1 $\rho_{22} = 1$ and else are zeros

Homework's figure in the last problem is 4×4, but the question asks the 3×3 matrix. Thus, I generate the two results in Figure1, the left one uses the definition of the left bottom corner, and the other (right) implements the center of the grid as a benchmark.

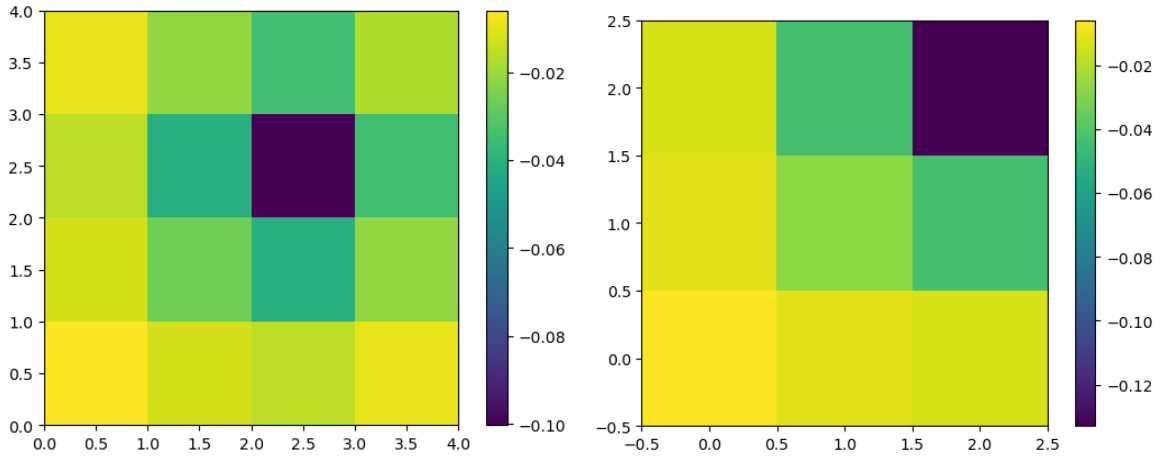

Figure 1: The left is the 4×4 grid; the right is the 3×3 grid. Both of them are with the source $\rho_{22} = 1$ and the others are zeros.

2.2 2D Poisson's equation with a given source with periodic boindary

In this subsection, we exploit the finite difference method with the sparse matrix to solve the equation:

$$\rho(x, y) = e^{-\frac{5}{4}r_1^2} + \frac{3}{2} \times e^{-r_2^2}$$

with the domain $\mathscr{D}\{[-5 < x < 5] \times [-5 < y < 5]\}$ in the 128×128 grid.

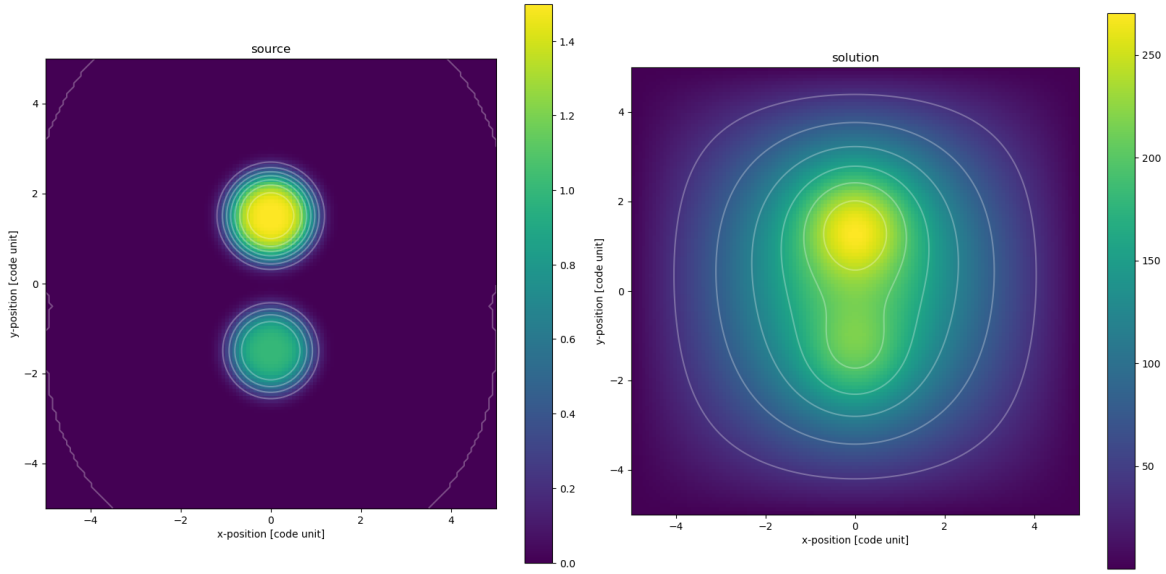

Figure 2: The left is the source function with the scheme with periodic boundary; the right is the solution of the corresponding potential for this Poisson's equation also with the periodic boundary exploiting the sparse matrix method. Moreover, this "periodic boundary" will be updated in each run during finite difference algorithm execution.

2.3 Error convergence comparison between different algorithms

We utilize three different methods learned in the lecture to investigate the error convergence of this Poisson's equation. The first method we used is the Jacobi method and the second is the Gauss-Seidel method, and the last method is the successive over-relaxation method with $w = 1.2,\ 1, 5,$ and 2.0. However, in SOR (successive over-relaxation method), it might be "diverge", so we plot two schemes to research the convergence rate, one is all converge and the other is one of them diverge, please see in Figure3.

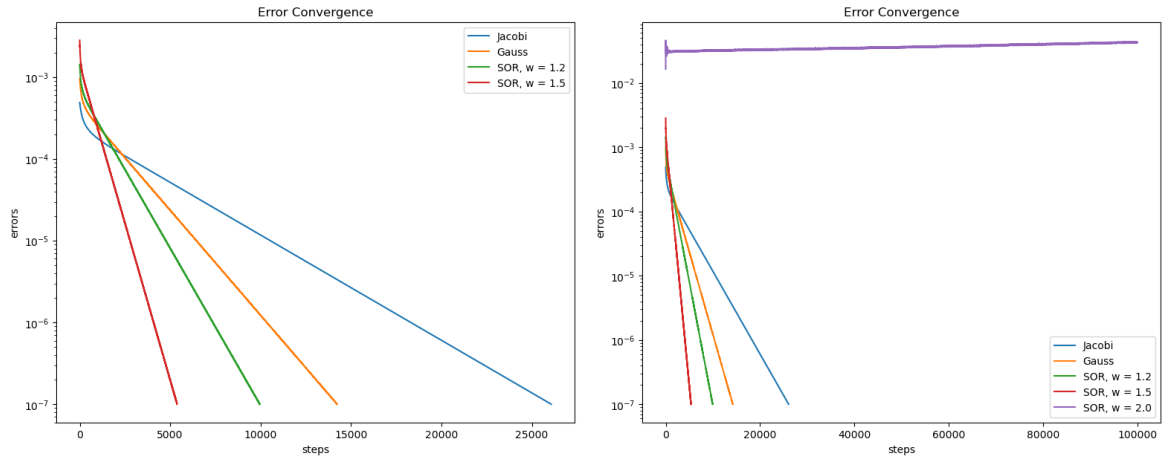

Figure 3: These two figures have the log-scale y-axis; normal scale x-axis. The left is all of methods are converged; the right scheme is the one diverge situation ($w = 2.0$).

It is manifestly that the Jacobi method is the slowest one and then is the Gauss-Seidel method, and the fastest is the SOR method. In this scenario, although SOR is the fastest when $w = 1.5$, it will diverge with $w = 2.0$ (Figure3-right)! The fastest (SOR with $w = 1.5$) is approximately 4.5 times faster than the slowest (Jacobi) as the error tolerence is $\sim 10^{-6}$.

3 Codes

All the codes are transferred from jupyterlab or python codes; hence, if you want to re-run them, please see the source code in the attached files or my GitHub repository:

<https://github.com/gary20000915/Comphyslab-HW4.git>

3.1 `particles.py`
