Poisson3D — A Fast 3D Poisson Solver

Yuan Cao and Weiguo Gao*

Poisson3D is a MATLAB code of a Fast Poisson Solver for the equation

$$\begin{cases} -\Delta u = f(x,y,z) \\ u|_{\partial\Omega} = b \end{cases}$$

where Ω is a cuboid area. Both a second-order scheme and a fourth-order scheme can be used. To use it, call the function

```
poisson3d(f,hx2,hy2,hz2,bx1,bx2,by1,by2,bz1,bz2,order)
```

where

- `f` is a 3D array of the values of the equation's right hand function on mesh points.
- `hx2`, `hy2` and `hz2` are the squares of the distance between two adjacent points in x,y,z directions.
- `bx1`, `bx2`, `by1`, `by2`, `bz1` and `bz2` are the corresponding boundary conditions of the equation.
- `order`=2 or 4 is the parameter for users to choose a scheme.

The 2nd-order scheme is the 7-points basic scheme which uses 3 points in each direction. If we denote the 3-point difference operator in one direction by δ_α^2, where α can be x, y or z ($(\delta_x^2 u)_{i,j,k} = \frac{u_{i+1,j,k}+u_{i-1,j,k}-2u_{i,j,k}}{h_x^2}$, for example), then the scheme is simply

$$-(\delta_x^2 + \delta_y^2 + \delta_z^2)u = f.$$

The 4th-order scheme is a compact scheme using 19 points:

$$-(\delta_x^2 + \delta_y^2 + \delta_z^2 + \frac{h_x^2 + h_y^2}{12}\delta_x^2\delta_y^2 + \frac{h_x^2 + h_z^2}{12}\delta_x^2\delta_z^2 + \frac{h_y^2 + h_z^2}{12}\delta_y^2\delta_z^2)u = f + \frac{h_x^2}{12}\delta_x^2 f + \frac{h_y^2}{12}\delta_y^2 f + \frac{h_z^2}{12}\delta_z^2 f.$$

To solve these two linear equations, the program first calls inverse discrete sine transform to the right-hand side in two directions, then solves the tridiagonal linear equations in the third direction, and finally uses discrete sine transform to get the solution. It chooses the longest direction to solve the tridiagonal linear equations to make the calculation faster.

Here are the functions in `poisson3d.m`:

- `poisson3d` - the main function which contains calls to subfunctions.

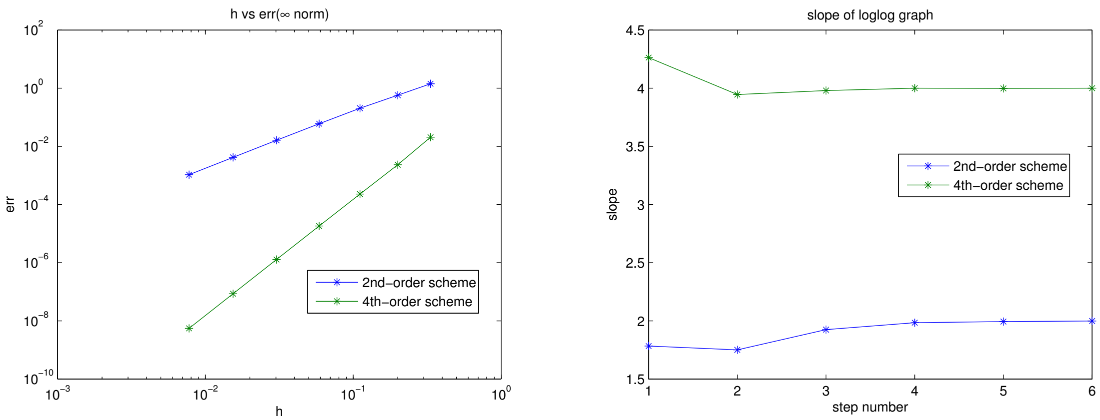
*School of Mathematical Sciences, Fudan University, Shangha 200433, China, email: wggao@fudan.edu.cn

- `my3dDST/my3diDST` - calls discrete sine transform/inverse discrete sine transform directly for a 3D array.
- `secdiff` - forms the modified right-hand side of the problem when the 4th-order scheme is resorted to.

Now we show two examples by setting

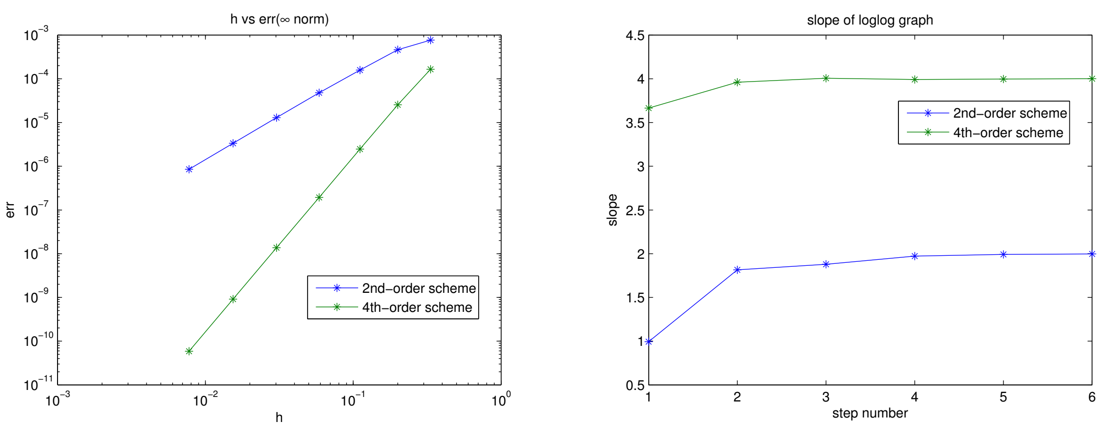
$$f = -14e^{x+2y+3z}, \qquad f = -\frac{2z(1-x+3y^2)}{(1+x+y^2)^3}$$

respectively. And the exact solutions are $u = e^{x+2y+3z}$ and $u = \dfrac{z}{1+x+y^2}$. To demonstrate



(a) (b)

Figure 1: $f = -14e^{x+2y+3z}$ with exact solution $u = e^{x+2y+3z}$



(a) (b)

Figure 2: $f = -\dfrac{2z(1-x+3y^2)}{(1+x+y^2)^3}$ with exact solution $u = \dfrac{z}{1+x+y^2}$

the convergence rate of two different schemes, we increase the mesh sizes n_x, n_y, n_z from 2 to 2^7. The relative errors with respect to the exact solutions are plotted in loglog form in Figures 1(a) and 2(a). The slopes of errors are shown in Figures 1(b) and 2(b).