Documentation of the Poisson solver

January 21, 2018

The Poisson equation is solved by directly solving a linear equation set for each z-plane.

$$\nabla_{\perp}^{2}\left(n\phi+p_{i}\right)=w$$

Let's take one of the z-planes for example.
We first implement fft to w in y direction to get

$$\begin{pmatrix} w_{x_1 ky_1} & w_{x_1 ky_2} & \cdots & w_{x_1 ky_{ny}} \\ w_{x_2 ky_1} & w_{x_2 ky_2} & \cdots & w_{x_2 ky_{ny}} \\ \vdots & \vdots & \ddots & \vdots \\ w_{x_{nx} ky_1} & w_{x_{nx} ky_2} & \cdots & w_{x_{nx} ky_{ny}} \end{pmatrix}$$

then reshape it into

$$\begin{pmatrix} W_{ky_1} \\ W_{ky_2} \\ \vdots \\ W_{ky_{ny}} \end{pmatrix}$$

where

$$W_{ky_i}=\begin{pmatrix} w_{x_1 ky_i} \\ w_{x_2 ky_i} \\ \vdots \\ w_{x_n ky_i} \end{pmatrix}$$

the variable `nphi_plus_p` is used to denote $\phi+p_i$, `lap` to denote the Laplacian operator casted to matrix. So that the linear equation set can be expressed here as

$$\begin{pmatrix} Lap_{ky_1} & & & \\ & Lap_{ky_2} & & \\ & & \ddots & \\ & & & Lap_{ky_{ny}} \end{pmatrix}\begin{pmatrix} Nphi\_plus\_p_{ky_1} \\ Nphi\_plus\_p_{ky_2} \\ \vdots \\ Nphi\_plus\_p_{ky_{ny}} \end{pmatrix}=\begin{pmatrix} W_{ky_1} \\ W_{ky_2} \\ \vdots \\ W_{ky_{ny}} \end{pmatrix}$$

The Laplacian is given by

$$\begin{aligned} \nabla_{\perp}^{2} &= \left(\frac{\partial}{\partial x}+\theta\hat{s}\frac{\partial}{\partial y}\right)^{2}+\frac{\partial^{2}}{\partial y^{2}} \\ &= \frac{\partial^{2}}{\partial x^{2}}+\theta\hat{s}\frac{\partial}{\partial x}\frac{\partial}{\partial y}+\frac{\partial^{2}}{\partial y^{2}} \end{aligned}$$

after the y-direction fft, the Laplacian goes to

$$\nabla_{\perp}^{2}=\frac{\partial^{2}}{\partial x^{2}}+\theta\hat{s}\mathrm{i}k_{y}\frac{\partial}{\partial x}-\left(1+(\theta\hat{s})^{2}\right)k_{y}^{2}\equiv lap1+lap2+lap3$$

In this program we use $\phi=0$ at the right boundary and floating boundary condition at the left boundary.

For a given z-plane, $\theta\hat{s}$ is constant. k_y is known once the gird is given. For a given k_y, we have

$$Lap_{ky_i} = \frac{1}{\Delta x^2}\begin{pmatrix} -1 & 1 & & & \\ 1 & -2 & 1 & & \\ & 1 & -2 & \ddots & \\ & & \ddots & \ddots & 1 \\ & & & 1 & -2 \end{pmatrix} + \frac{\theta\hat{s}}{2\Delta x} iky_i \begin{pmatrix} -1 & 1 & & & \\ -1 & 0 & 1 & & \\ & -1 & 0 & \ddots & \\ & & \ddots & \ddots & 1 \\ & & & -1 & 0 \end{pmatrix} - \left(1 + (\theta\hat{s})^2\right) ky_i^2 \begin{pmatrix} 1 & & & & \\ & 1 & & & \\ & & 1 & & \\ & & & 1 & \\ & & & & 1 \end{pmatrix}$$

together with

$$Lap_{ky_i}\begin{pmatrix} nphi\_plus\_p_{x_1 ky_i} \\ nphi\_plus\_p_{x_2 ky_i} \\ \vdots \\ nphi\_plus\_p_{x_{nx-1} ky_i} \\ nphi\_plus\_p_{x_{nx} ky_i} \end{pmatrix} = \begin{pmatrix} w_{x_1 ky_i} \\ w_{x_2 ky_i} \\ \vdots \\ w_{x_{nx-1} ky_i} \\ \left(w_{x_{nx}} - \frac{1}{\Delta x^2} p_{right}\right)_{ky_i} \end{pmatrix}$$

where p_{right} is p at the right boundary, fetched at runtime.