MLonNM

Athanasios Mattas, 2019 | atmattas@physics.auth.gr

MLonNM (Machine Learning on Numerical methods) is an ongoing project, investigating how ML can be utilized to accelerate numerical methods.

The **Successive Over Relaxation (SOR)** method on a 2D grid is selected as a first approach. SOR solves elliptic partial differential equations, like the Laplace or the Poisson's equation. For this project, the weak (variable-coefficient) form of the Poisson's equation[1] is used:

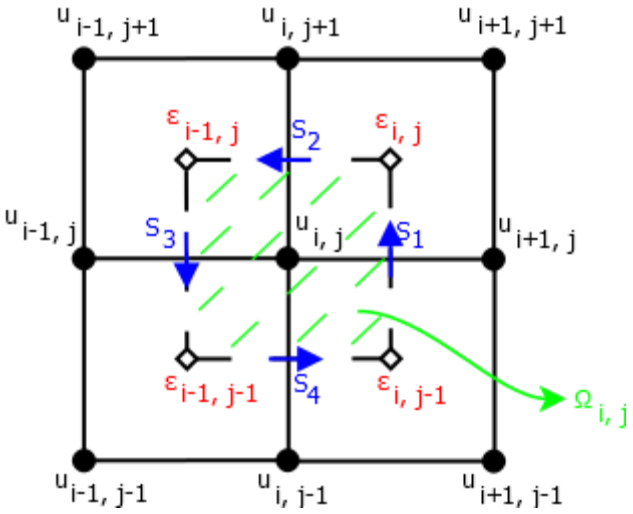
$$\boxed{\nabla[\varepsilon_r(\vec{r})\nabla u(\vec{r})] = f(\vec{r})}$$

where $\varepsilon_r(\vec{r})$ expresses the non-homogeneity of the medium (e.g. the relative permittivity at electrostatics as a function of the coordinates – the corresponding symbol is used for clarity) $u(\vec{r})$ is the function to be evaluated inside a domain space (the voltage potential V, staying with the electrostatics example) and $f(\vec{r})$ is the source (forcing) function (again, in electrostatics: $f(\vec{r}) = -\frac{\rho(\vec{r})}{\varepsilon_0}$, where ρ is the charge density function and ε_0 the permittivity of free space).

For convenience, ε_r populates a staggered grid:

$$\varepsilon_r^{i,j} = \varepsilon_r\left(x_{i+1/2}, y_{j+1/2}\right)$$

where *i, j* denote the node indexes at the *x* and *y* directions.



- basic mesh
- ◊ staggered (vertex centered) mesh

[1] Nagel, J. (2011) Solving the Generalized Poisson Equation Using the Finite-Difference Method (FDM). Retrieved from researchgate.net .

Iterative scheme built

$$\iint_{\Omega_{i,j}} \nabla[\varepsilon_r(\vec{r})\nabla u(\vec{r})]d\Omega = \iint_{\Omega_{i,j}} f(\vec{r})d\Omega$$

1. $\iint_{\Omega_{i,j}} f(\vec{r})d\Omega = h^2 f_{i,j}$

   where h is the discretization step (same at both directions) and $f_{i,j}$ is the average value of $f(\vec{r})$ in the area $\Omega_{i,j}$.

2. $\iint_{\Omega_{i,j}} \nabla[\varepsilon_r(\vec{r})\nabla u(\vec{r})]d\Omega \xrightarrow{Ostrogradsky-Gauss\ theorem} \int_{C_{i,j}} \varepsilon_r(\vec{r})\nabla u(\vec{r})d\hat{n}$

   where $C_{i,j}$ is the perimeter of the area $\Omega_{i,j}$ and $\hat{n}$ is the perpendicular to the area unit vector.

   $$\int_{C_{i,j}} \varepsilon_r(\vec{r})\nabla u(\vec{r})d\hat{n} = \int_{C_{i,j}} \varepsilon_r(x,y)\left[\frac{\partial}{\partial \mathrm{x}}u(x,y)\hat{\imath} + \frac{\partial}{\partial \mathrm{y}}u(x,y)\hat{\jmath}\right]d\hat{n} = S_1 + S_2 + S_3 + S_4$$

   $$S_1 = \int_{-h/2}^{h/2} \varepsilon_r(x,y)\left[\frac{\partial}{\partial \mathrm{x}}u(x,y)\hat{\imath} + \frac{\partial}{\partial \mathrm{y}}u(x,y)\hat{\jmath}\right]\hat{\imath}dy = \int_{-h/2}^{h/2} \varepsilon_r(x,y)\frac{\partial}{\partial \mathrm{x}}u(x,y)dy = \frac{\partial}{\partial \mathrm{x}}u(x,y)h\varepsilon_{i,j}$$

   - $\frac{\partial}{\partial \mathrm{x}}u(x,y)$ is constant across the contour line and gets out of the integral and with forward differences:
     $$\frac{\partial}{\partial \mathrm{x}}u(x,y) = \frac{u_{i+1,j} - u_{i,j}}{h}$$
   - $\varepsilon_{i,j}$ is the average "relative permittivity" across the contour line. It can be expressed as the mean between the nodes:
     $$\overline{\varepsilon_{i,j}} = \frac{\varepsilon_{i+1/2,j+1/2} + \varepsilon_{i-1/2,j-1/2}}{2}$$
     or, using the staggered mesh:
     $$\overline{\varepsilon_{i,j}} = \frac{\varepsilon_{i,j} + \varepsilon_{i,j-1}}{2}$$

   So,

   $$S_1 = h\frac{\varepsilon_{i,j} + \varepsilon_{i,j-1}}{2}\frac{u_{i+1,j} - u_{i,j}}{h} = \frac{1}{2}\left(\varepsilon_{i,j} + \varepsilon_{i,j-1}\right)\left(u_{i+1,j} - u_{i,j}\right)$$

   Likewise,

   $$S_2 = \frac{1}{2}\left(\varepsilon_{i,j} + \varepsilon_{i-1,j}\right)\left(u_{i,j+1} - u_{i,j}\right)$$

   $$S_3 = \frac{1}{2}\left(\varepsilon_{i-1,j} + \varepsilon_{i-1,j-1}\right)\left(u_{i-1,j} - u_{i,j}\right)$$

   $$S_4 = \frac{1}{2}\left(\varepsilon_{i,j-1} + \varepsilon_{i-1,j-1}\right)\left(u_{i,j-1} - u_{i,j}\right)$$

Defining the factors of the 5-points scheme:

$$a_1 = \frac{1}{2}\left(\varepsilon_{i,j} + \varepsilon_{i,j-1}\right)$$

$$a_2 = \frac{1}{2}\left(\varepsilon_{i,j} + \varepsilon_{i-1,j}\right)$$

$$a_3 = \frac{1}{2}\left(\varepsilon_{i-1,j} + \varepsilon_{i-1,j-1}\right)$$

$$a_4 = \frac{1}{2}\left(\varepsilon_{i,j-1} + \varepsilon_{i-1,j-1}\right)$$

$$a_0 = \varepsilon_{i,j} + \varepsilon_{i-1,j} + \varepsilon_{i,j-1} + \varepsilon_{i-1,j-1}$$

At last, equating 1 and 2:

$$S_1 + S_2 + S_3 + S_4 = h^2 f_{i,j}$$

$$-a_0 u_{i,j} + a_1 u_{i+1,j} + a_2 u_{i,j+1} + a_3 u_{i-1,j} + a_4 u_{i,j-1} = h^2 f_{i,j}$$

$$\boxed{u_{i,j}^{n+1} = \frac{1}{a_0}\begin{Bmatrix} & a_2 & \\ a_3 & 0 & a_1 \\ & a_4 & \end{Bmatrix} u_{i,j}^{n} - h^2 f_{i,j}^{n}}$$ Liebmann method

$$\boxed{u_{i,j}^{n+1} = u_{i,j}^{n} + \frac{\omega}{a_0}\begin{Bmatrix} & a_2 & \\ a_3 & -a_0 & a_1 \\ & a_4 & \end{Bmatrix} u_{i,j}^{n} - h^2 f_{i,j}^{n}}$$ SOR method

where

- n and n+1 denote current and next time-step
- ω (omega) is the relaxation factor (0-2)