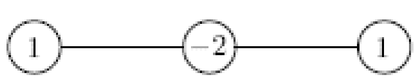
Lecture 9: Numerical Partial Differential Equations(Part 2)

Finite Difference Method to Solve Poisson's Equation

- Poisson's equation in 1D:

$$\begin{cases} -\dfrac{d^2u}{dx^2} = f(x),\ \ x \in (0,1) \\ \quad u(0) = u(1) = 0 \end{cases}.$$

- Spatial Discretization: $0 = x_0 < \cdots < x_M = 1$. Define $\Delta x = \frac{1}{M}$. Then $x_i = i\Delta x$.
- $\dfrac{d^2u(x_i)}{dx^2} \sim \dfrac{u(x_{i-1}) - 2u(x_i) + u(x_{i+1})}{\Delta x^2}$
- Stencil of finite difference approximation



- Finite difference equations: for $i = 1, \dots, M-1$

$$-u_{i-1} + 2u_i - u_{i+1} = \Delta x^2 f_i$$
$$u_0 = 0$$
$$u_M = 0$$

with $f_i = f(x_i)$

- Put into matrix equation format:

  Let $\boldsymbol{u} = (u_1, u_2, \dots, u_{M-1})^T, \boldsymbol{f} = (f_1, f_2, \dots, f_{M-1})^T$

$$A\boldsymbol{u} = \Delta x^2 \boldsymbol{f}$$

$$A = \begin{bmatrix} 2 & -1 & & & & & \\ -1 & 2 & -1 & & & & \\ & -1 & 2 & -1 & & & \\ & & \ddots & \ddots & \ddots & & \\ & & -1 & 2 & -1 & & \\ & & & -1 & 2 & -1 \\ & & & & -1 & 2 \end{bmatrix}$$

2D Poisson's Equation

Consider to solve

$$\begin{cases} -(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}) = f(x,y), \ (x,y) \in \Omega \\ \qquad u(x,y) = 0 \quad on \quad \partial\Omega \end{cases}$$

with Ω is rectangle $(0,1) \times (0,1)$ and $\partial\Omega$ is its boundary.

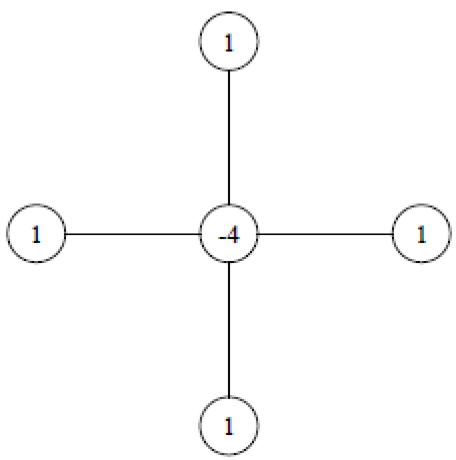
- Define $h = \frac{1}{M}$.
- Spatial Discretization: $0 = x_0 < \cdots < x_M = a$ with $x_i = ih$ and $0 = y_0 < \cdots < y_M = 1$ with $y_j = jh$.

Finite difference equation at grid point (i, j):

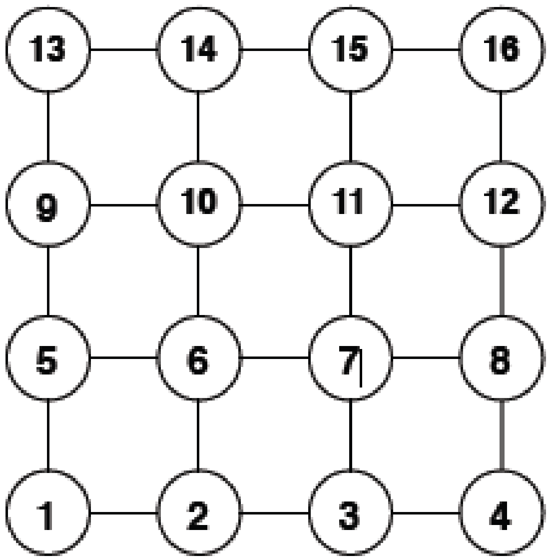
$$-\left(\frac{u_{i-1,j}-2u_{i,j}+u_{i+1,j}}{h^2}+\frac{u_{i,j-1}-2u_{i,j}+u_{i,j+1}}{h^2}\right)=f(x_i, y_j) \text{ or}$$

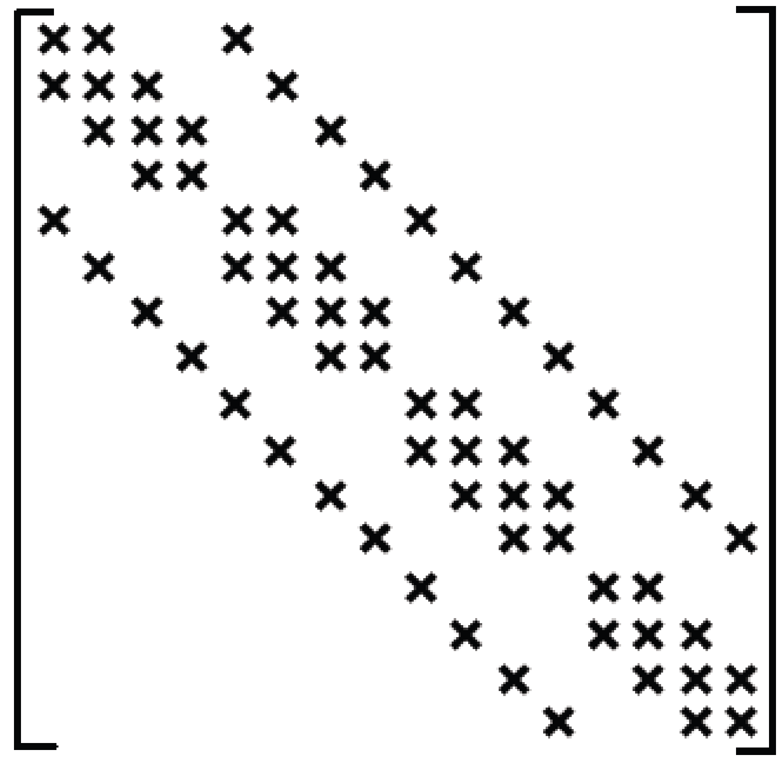
$$-u_{i,j-1}-u_{i-1,j}+4u_{i,j}-u_{i+1,j}-u_{i,j+1}=h^2 f(x_i, y_j)$$

- Five-point stencil of the finite difference approximation



Natural Row-Wise Ordering



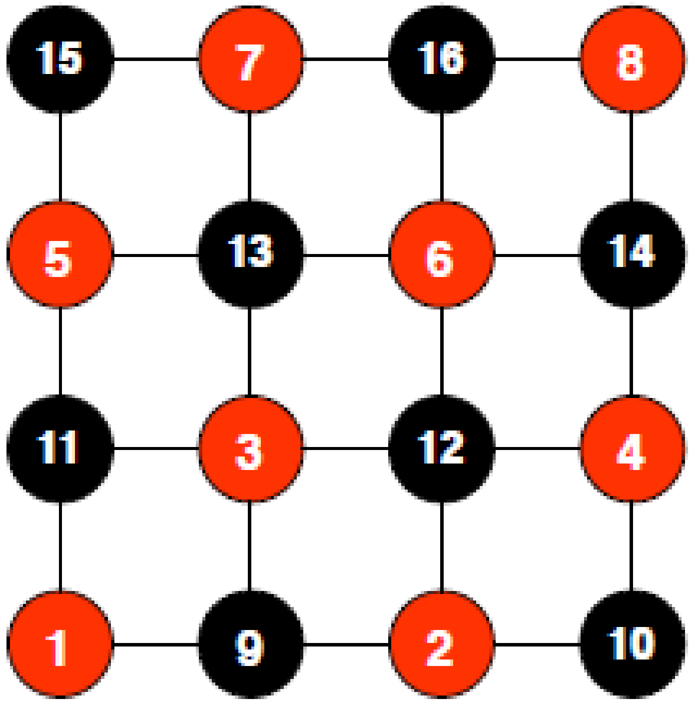


$$u_{ij}^{(k+1)} = (1-w)u_{ij}^{(k)} + w/4\left(h^2 f_{ij} + u_{i-1,j}^{(k+1)} + u_{i,j-1}^{(k+1)} + u_{i+1,j}^{(k)} + u_{i,j+1}^{(k)}\right)$$

This is completely sequential.

Red-Black Ordering

- Color the alternate grid points in each dimension red or black



R/B SOR

- First iterates on red points by

$$u_{ij}^{(k+1)} = (1-w)u_{ij}^{(k)} + w/4\left(h^2 f_{ij} + u_{i-1,j}^{(k)} + u_{i,j-1}^{(k)} + u_{i+1,j}^{(k)} + u_{i,j+1}^{(k)}\right)$$

- Then iterates on black points by

$$u_{ij}^{(k+1)} = (1-w)u_{ij}^{(k)} + w/4\left(h^2 f_{ij} + u_{i-1,j}^{(k+1)} + u_{i,j-1}^{(k+1)} + u_{i+1,j}^{(k+1)} + u_{i,j+1}^{(k+1)}\right)$$

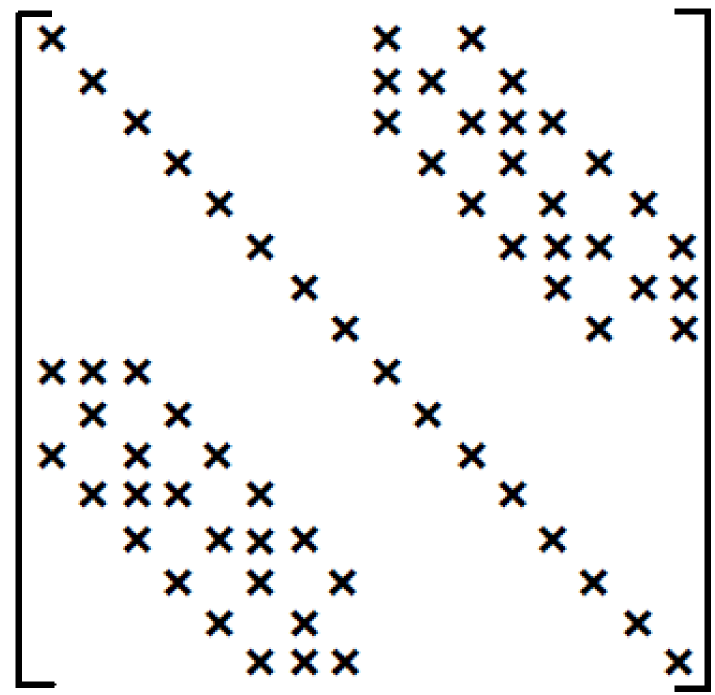
- R/B SOR can be implemented in parallel on the same color grid points.
- The renumbering of the matrix A changes the iteration formula.

For the example just shown:

$$A = \begin{bmatrix} D_r & -C \\ -C^T & D_b \end{bmatrix}$$

Diagonal matrices $D_r = D_b = 4I_8$.

$$C = \begin{pmatrix} 1 & & 1 & & & & & \\ 1 & 1 & & 1 & & & & \\ 1 & & 1 & & 1 & 1 & & \\ & 1 & & 1 & & 1 & & \\ & & 1 & & 1 & & 1 & \\ & & & 1 & 1 & 1 & & 1 \\ & & & & 1 & & 1 & 1 \\ & & & & & 1 & & 1 \end{pmatrix}$$



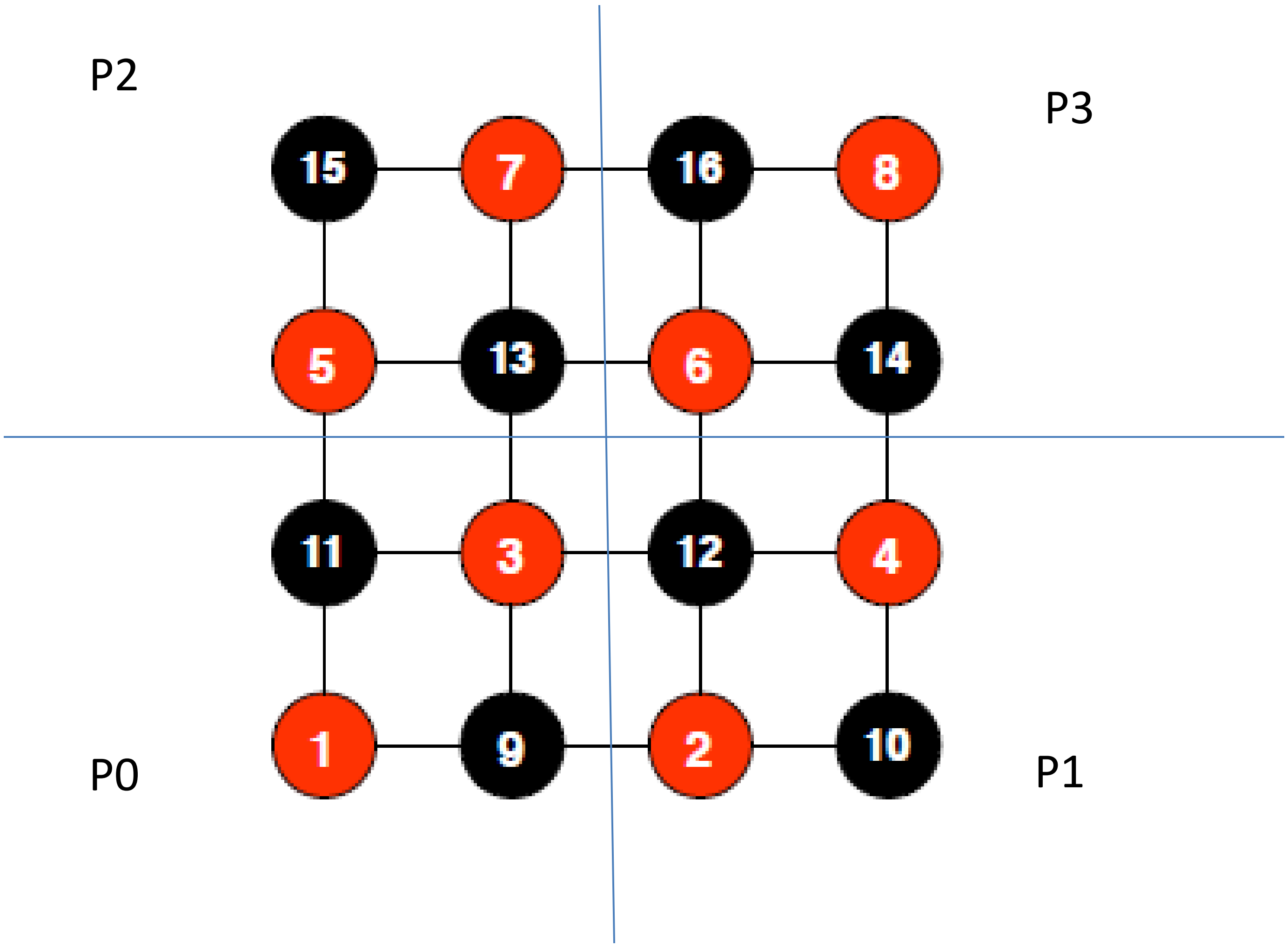
- Using GS:

$$\begin{bmatrix} D_r & 0 \\ -C^T & D_b \end{bmatrix} \begin{bmatrix} \boldsymbol{u}_r^{(k+1)} \\ \boldsymbol{u}_b^{(k+1)} \end{bmatrix} = \begin{bmatrix} 0 & C \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{u}_r^{(k)} \\ \boldsymbol{u}_b^{(k)} \end{bmatrix} + h^2 \boldsymbol{f}$$

Here $\boldsymbol{u}_r = (u_1, u_2, u_3, u_4, \ldots, u_8)^T$

$$\boldsymbol{u}_b = (u_9, u_{10}, u_{11}, u_{12}, \ldots, u_{16})^T$$

Parallel R/B SOR



Algorithm

While error > TOL, do:

- Compute all red-points
- Send/Recv values of the red-points at the boarder of the subdomain to neighboring processes
- Compute all black-points
- Send/Recv values of the black-points at the boarder of the subdomain to neighboring processes

  Compute residual error

Endwhile

References

- L. Adams and J.M. Ortega. A Multi-Color SOR Method for Parallel Computation. ICASE 82-9. 1982.
- L. Adams and H.F. Jordan. Is SOR Color-Blind? *SIAM J. Sci. Stat. Comput.* 7(2):490-506, 1986
- D. Xie and L. Adams. New Parallel SOR Method by Domain Partitioning. *SIAM J. Sci. Comput.* 20(6), 1999.