# Steady-State and Transient Heat Transfer problems in C++ framework

## Mech587 Project #1

Xiaoyu Mao

Oct. 3, 2022

THE UNIVERSITY OF BRITISH COLUMBIA

# Steady-State heat transfer problem: Definition

- Governing equation and boundary conditions

$$\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) = 0 \text{ in } (0,1) \times (0,1)$$

$$u = \begin{cases} y^4, & x = 0 \\ x^4, & y = 0 \\ 1 - 6y^2 + y^4, & x = 1 \\ 1 - 6x^2 + x^4, & y = 1 \end{cases} \quad \text{on}$$

$u = 1 - 6x^2 + x^4$

$u = y^4$

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0$$

$u = 1 - 6y^2 + y^4$

$u = x^4$

- The problem has the following exact solution:

$$u(x, y) = x^4 + y^4 - 6x^2y^2$$

- To simplify the notation, we denote:

$$\Delta u = \left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}\right) u, \qquad \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) = 0 \Rightarrow \Delta u = 0.$$

# Newton-Raphson iteration (1)

- The Newton-Raphson iteration is meant for non-linear root-finding problems:

$$f(x) = ax^5 + be^x + c\sin(x), \text{ solve } f(x) = 0$$

  - Start with initial guess $x_0$
  - Update the solution with $x_{k+1} = x_k + \delta x_k$
  - Hoping that $f(x_{k+1}) = 0$

- How to solve the increment $\delta x_k$ ?
  - Write in increment form:

$$f(x_{k+1}) = 0 \Rightarrow f(x_k + \delta x_k) = 0$$

  - Taylor Expansion:

$$f(x_k) + f'(x_k)\delta x_k = 0$$

  - Solve the increment:

$$f'(x_k)\delta x_k = -f(x_k)$$

  - Update solution: $x_{k+1} = x_k + \delta x_k$

# Newton-Raphson iteration (2)

- The Newton-Raphson method solve the problem in an iterative manner:
  - We start with an initial guess $u_0$
  - We update the guess with increment $u_1 = u_0 + \delta u_0$
  - Keep updating the solution $u_{k+1} = u_k + \delta u_k$
  - With the hope that $\Delta u_{k+1} = 0$

- How to get $\delta u_k$ and update the solution?
  - $\Delta u_{k+1} = 0 \Rightarrow \Delta(u_k + \delta u_k) = 0$
  - Problem is linear: $\Delta(u_k + \delta u_k) = \Delta u_k + \Delta(\delta u_k) = 0$
  - $\Delta(\delta u_k) = -\Delta u_k$, where $u_k$ is known and $\delta u_k$ is to be solved
  - Update the solution by $u_{k+1} = u_k + \delta u_k$ until $\|\delta u_k\|_2 < tol$

Note: The necessity of this method will be straight forward later when we handling the non-linear convection terms. We use it for linear problem here only for learning purpose.

# Matrix form with central difference method (1)

- Now we just need to solve:

$$\Delta(\delta u_k) = -\Delta u_k$$

- For true solution, $\Delta u = 0$. Since $u_k \neq u$, $\Delta u_k$ is a non-zero vector. We call $\Delta u_k$ as the <u>*Residual*</u>

$$\left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}\right)\delta u_k = -\left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}\right)u_k$$

- After discretization, the matrix form becomes:

$$\boldsymbol{A}\boldsymbol{\delta u_k} = -\boldsymbol{R}$$

- Where $\boldsymbol{A}$ is called the ***Jacobian matrix***, $\boldsymbol{\delta u_k}$ is the ***unknown vector***, $\boldsymbol{R}$ is the ***residual vector***.

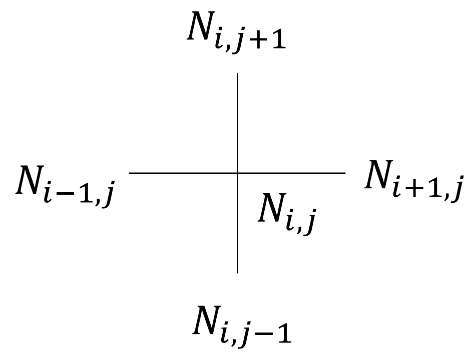
- Your task in this project is to construct $\boldsymbol{A}$ and $\boldsymbol{R}$ to solve for $\boldsymbol{\delta u_k}$

# Central difference approximation

- Remove subscript $k$ considering all the vectors are at $k$ iteration:

$$\Delta(\delta u_k) = -\Delta u_k \Rightarrow \left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}\right)\delta u = -\left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}\right)u$$

- To approximate the Laplacian operator $\Delta$:




  - Consider the central difference stencil:

    At Node $N_{i,j}$

$$\left(\frac{\partial^2}{\partial x^2}\delta u\right)_{i,j} \Rightarrow \frac{\delta u_{i-1,j} - 2\delta u_{i,j} + \delta u_{i+1,j}}{\Delta x^2}$$

$$\left(\frac{\partial^2}{\partial y^2}\delta u\right)_{i,j} \Rightarrow \frac{\delta u_{i,j-1} - 2\delta u_{i,j} + \delta u_{i,j+1}}{\Delta y^2}$$

  - Adding them up, the LHS of the equation can be discretized as:

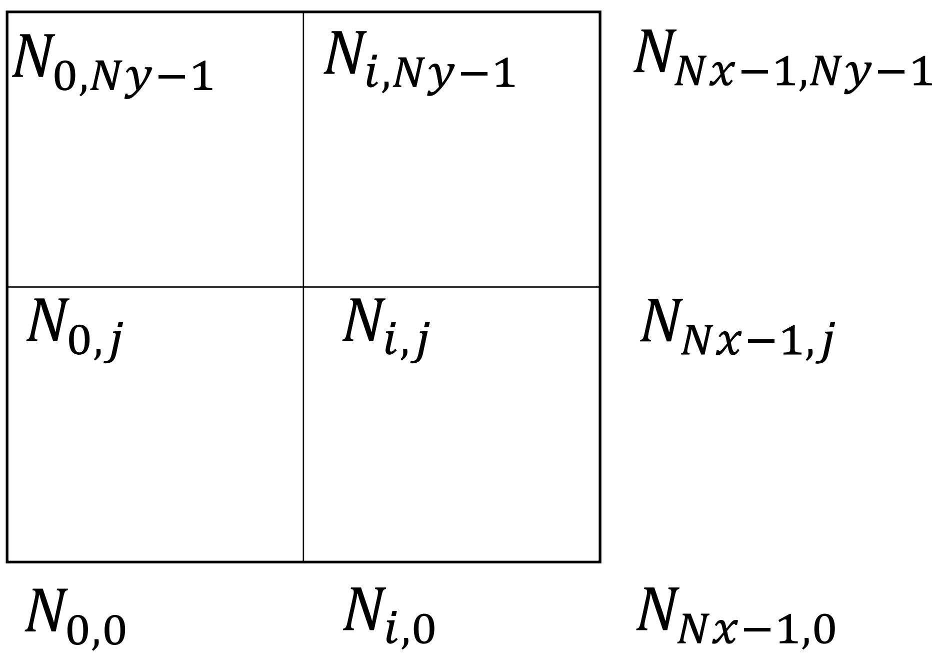
$$\left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}\right)_{i,j}\delta u = \frac{1}{\Delta x^2}\delta u_{i-1,j} + \frac{1}{\Delta y^2}\delta u_{i,j-1} + \left(-\frac{2}{\Delta x^2} - \frac{2}{\Delta y^2}\right)\delta u_{i,j} + \frac{1}{\Delta y^2}\delta u_{i,j+1} + \frac{1}{\Delta x^2}\delta u_{i+1,j}$$

  - Similarly, the RHS of the equation can be discretized as

$$\left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}\right)_{i,j} u = \frac{u_{i-1,j} - 2u_{i,j} + u_{i+1,j}}{\Delta x^2} + \frac{u_{i,j-1} - 2u_{i,j} + u_{i,j+1}}{\Delta y^2}$$

# Index of the mesh nodes

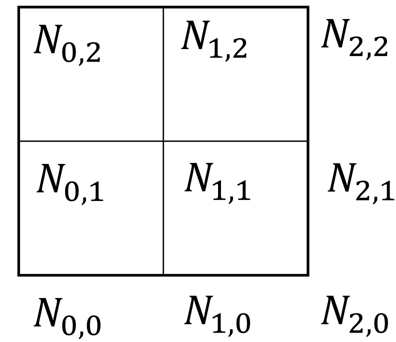
- The node indexes are
  - i = 1, 2,⋯, Nx in the X-direction,
  - j = 1, 2,⋯, Ny in the Y-direction
- Since the index in C++ starts from 0, we have :
  - Index i from 0 to Nx-1 in the X-direction
  - Index j from 0 to Ny-1 in the Y-direction



# Equation for interior node

- Remove subscript $k$ considering all the vectors are at $k$ iteration:

$$\Delta(\delta u_k) = -\Delta u_k \Rightarrow \left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}\right)\delta u = -\left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}\right)u$$




- Let's consider the simplest possible mesh at the RHS:
  with only one interior node: $N_{1,1}$, where we need to solve the PDE
  - With central difference method, the LHS of the PDE at Node $N_{1,1}$ can be approximated as:

$$\left(\left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}\right)\delta u\right)_{1,1} = \frac{1}{\Delta x^2}\delta u_{0,1} + \frac{1}{\Delta y^2}\delta u_{1,0} + \left(-\frac{2}{\Delta x^2} - \frac{2}{\Delta y^2}\right)\delta u_{1,1} + \frac{1}{\Delta y^2}\delta u_{1,2} + \frac{1}{\Delta x^2}\delta u_{2,1}$$

  - With central difference method, the RHS of the PDE at Node $N_{1,1}$ can be approximated as:

$$-\left(\left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}\right)u\right)_{1,1} = -\left(\frac{u_{0,1} - 2u_{1,1} + u_{2,1}}{\Delta x^2} + \frac{u_{1,0} - 2u_{1,1} + u_{1,2}}{\Delta y^2}\right)$$

  - Therefore, the PDE at $N_{1,1}$ can be discretized as:

$$\frac{1}{\Delta x^2}\delta u_{0,1} + \frac{1}{\Delta y^2}\delta u_{1,0} + \left(-\frac{2}{\Delta x^2} - \frac{2}{\Delta y^2}\right)\delta u_{1,1} + \frac{1}{\Delta y^2}\delta u_{1,2} + \frac{1}{\Delta x^2}\delta u_{2,1}$$

# Equations for boundary nodes

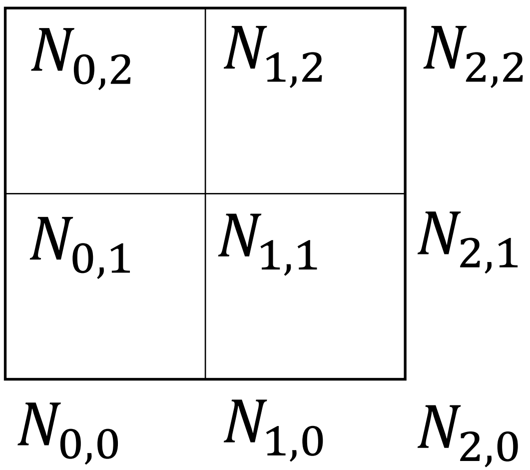
- Only Dirichlet boundary conditions are specified in this problem, which means that the function value at boundary nodes are directly given.

- Boundary conditions needs to be enforced at every time step as:

$$u_{\text{boundary}} = f(x, y, t)$$

- Since the values are known, we don't need to solve the PDE or update the solution. Therefore $\delta u_{\text{boundary}} = 0$. Considering all the boundary nodes, we have:

$$\begin{aligned}
\delta u_{0,0} &= 0 \\
\delta u_{0,1} &= 0 \\
\delta u_{0,2} &= 0 \\
\delta u_{1,0} &= 0 \\
\delta u_{1,2} &= 0 \\
\delta u_{2,0} &= 0 \\
\delta u_{2,1} &= 0 \\
\delta u_{2,2} &= 0
\end{aligned}$$



# Matrix form of the problem:

- Putting all equations together to form the linear system and denote them as $E_{i,j}$:

$$
\begin{cases}
\delta u_{0,0} = 0 & (E_{0,0}) \\
\delta u_{0,1} = 0 & (E_{0,1}) \\
\delta u_{0,2} = 0 & (E_{0,2}) \\
\delta u_{1,0} = 0 & (E_{1,0}) \\
\dfrac{1}{\Delta x^2}\delta u_{0,1} + \dfrac{1}{\Delta y^2}\delta u_{1,0} + \left(-\dfrac{2}{\Delta x^2} - \dfrac{2}{\Delta y^2}\right)\delta u_{1,1} + \dfrac{1}{\Delta y^2}\delta u_{1,2} + \dfrac{1}{\Delta x^2}\delta u_{2,1} = & \\
\qquad -\left(\dfrac{u_{0,1} - 2u_{1,1} + u_{2,1}}{\Delta x^2} + \dfrac{u_{1,0} - 2u_{1,1} + u_{1,2}}{\Delta y^2}\right) & (E_{1,1}) \\
\delta u_{1,2} = 0 & (E_{1,2}) \\
\delta u_{2,0} = 0 & (E_{2,0}) \\
\delta u_{2,1} = 0 & (E_{2,1}) \\
\delta u_{2,2} = 0 & (E_{2,2})
\end{cases}
$$

- Matrix form:

$$
\begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & \dfrac{1}{\Delta x^2} & 0 & \dfrac{1}{\Delta y^2} & -\dfrac{2}{\Delta x^2} - \dfrac{2}{\Delta y^2} & \dfrac{1}{\Delta y^2} & 0 & \dfrac{1}{\Delta x^2} & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
\delta u_{0,0} \\ \delta u_{0,1} \\ \delta u_{0,2} \\ \delta u_{1,0} \\ \delta u_{1,1} \\ \delta u_{1,2} \\ \delta u_{2,0} \\ \delta u_{2,1} \\ \delta u_{2,2}
\end{bmatrix}
= -
\begin{bmatrix}
0 \\ 0 \\ 0 \\ 0 \\ \left(\dfrac{u_{0,1} - 2u_{1,1} + u_{2,1}}{\Delta x^2} + \dfrac{u_{1,0} - 2u_{1,1} + u_{1,2}}{\Delta y^2}\right) \\ 0 \\ 0 \\ 0 \\ 0
\end{bmatrix}
\begin{array}{l}
(E_{0,0}) \\ (E_{0,1}) \\ (E_{0,2}) \\ (E_{1,0}) \\ (E_{1,1}) \\ (E_{1,2}) \\ (E_{2,0}) \\ (E_{2,1}) \\ (E_{2,2})
\end{array}
$$

Jacobian matrix $\boldsymbol{A}$ Unknown vector $\boldsymbol{\delta u_k}$ Residual vector $-\boldsymbol{R}$

# Denotation for sparse matrix (2)

- Matrix form

$$
\begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & \frac{1}{\Delta x^2} & 0 & \frac{1}{\Delta y^2} & -\frac{2}{\Delta x^2}-\frac{2}{\Delta y^2} & \frac{1}{\Delta y^2} & 0 & \frac{1}{\Delta x^2} & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
\delta u_{0,0} \\ \delta u_{0,1} \\ \delta u_{0,2} \\ \delta u_{1,0} \\ \delta u_{1,1} \\ \delta u_{1,2} \\ \delta u_{2,0} \\ \delta u_{2,1} \\ \delta u_{2,2}
\end{bmatrix}
= -
\begin{bmatrix}
0 \\ 0 \\ 0 \\ 0 \\ \left(\frac{u_{0,1}-2u_{1,1}+u_{2,1}}{\Delta x^2}+\frac{u_{1,0}-2u_{1,1}+u_{1,2}}{\Delta y^2}\right) \\ 0 \\ 0 \\ 0 \\ 0
\end{bmatrix}
\begin{array}{c}
(E_{0,0}) \\ (E_{0,1}) \\ (E_{0,2}) \\ (E_{1,0}) \\ (E_{1,1}) \\ (E_{1,2}) \\ (E_{2,0}) \\ (E_{2,1}) \\ (E_{2,2})
\end{array}
$$

Jacobian matrix $\boldsymbol{A}$ Unknown vector $\boldsymbol{\delta u}$ Residual vector $-\boldsymbol{R}$

- To save memory, we don't want to store zeros. Therefore, we need to come up with projection from the node indexing to the position of the matrices and vectors for all the coefficients.

- Each linear equation takes one row in the matrix form.
  - Currently we are using column indexing, which means $E_{i,j} \rightarrow (\text{row of } (i \times Ny + (j+1)))$
  - For example, the equation $E_{1,1}$ at $N_{1,1}$ goes to $(\text{row of } (1 \times 3 + (1+1))) = (\text{row of } 5)$

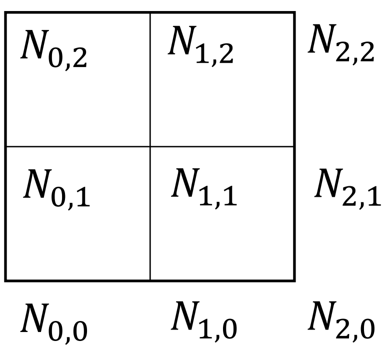
# Denotation of sparse matrix (1)

- We notice that there are lots of zeros in the Jacobian matrix and residual vector:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{\Delta x^2} & 0 & \frac{1}{\Delta y^2} & -\frac{2}{\Delta x^2}-\frac{2}{\Delta y^2} & \frac{1}{\Delta y^2} & 0 & \frac{1}{\Delta x^2} & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \delta u_{0,0} \\ \delta u_{0,1} \\ \delta u_{0,2} \\ \delta u_{1,0} \\ \delta u_{1,1} \\ \delta u_{1,2} \\ \delta u_{2,0} \\ \delta u_{2,1} \\ \delta u_{2,2} \end{bmatrix} = - \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \left(\frac{u_{0,1} - 2u_{1,1} + u_{2,1}}{\Delta x^2} + \frac{u_{1,0} - 2u_{1,1} + u_{1,2}}{\Delta y^2}\right) \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Jacobian matrix $\boldsymbol{A}$ Unknown vector $\boldsymbol{\delta u}$ Residual vector $-\boldsymbol{R}$

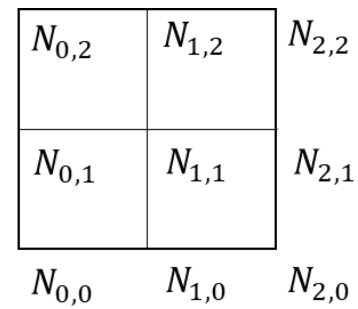
- Following the same column indexing, we have $\delta u_{i,j} \rightarrow \boldsymbol{\delta u}(i \times Ny + (j+1))$
  - For example, $\delta u_{1,1} \rightarrow \delta u(1 \times 3 + (1+1)) = \boldsymbol{\delta u}(5)$, which is the fifth entry of vector $\delta u$
- Similarly, $R_{i,j} \rightarrow R(i \times Ny + (j+1))$
  - For example, $R_{1,1} \rightarrow \boldsymbol{R}(1 \times 3 + (1+1)) = \boldsymbol{R}(5)$, which is the fifth entry of vector $\delta u$



# Denotation for sparse matrix (3)

- We have five coefficients for the central difference stencil in $E_{i,j}$ when $N_{i,j}$ is an interior node. Considering $E_{1,1}$ as an example:

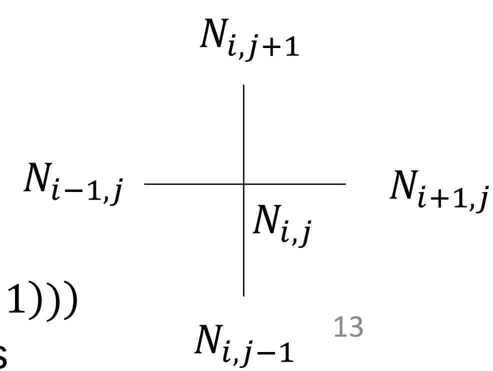
$$
\begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & \color{red}{\frac{1}{\Delta x^2}} & 0 & \color{red}{\frac{1}{\Delta y^2}} & \color{red}{-\frac{2}{\Delta x^2}-\frac{2}{\Delta y^2}} & \color{red}{\frac{1}{\Delta y^2}} & 0 & \color{red}{\frac{1}{\Delta x^2}} & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
\delta u_{0,0} \\ \delta u_{0,1} \\ \delta u_{0,2} \\ \delta u_{1,0} \\ \delta u_{1,1} \\ \delta u_{1,2} \\ \delta u_{2,0} \\ \delta u_{2,1} \\ \delta u_{2,2}
\end{bmatrix}
= -
\begin{bmatrix}
0 \\ 0 \\ 0 \\ 0 \\ \left(\frac{u_{0,1}-2u_{1,1}+u_{2,1}}{\Delta x^2}+\frac{u_{1,0}-2u_{1,1}+u_{1,2}}{\Delta y^2}\right) \\ 0 \\ 0 \\ 0 \\ 0
\end{bmatrix}
\begin{array}{l}
(E_{0,0}) \\ (E_{0,1}) \\ (E_{0,2}) \\ (E_{1,0}) \\ (E_{1,1}) \\ (E_{1,2}) \\ (E_{2,0}) \\ (E_{2,1}) \\ (E_{2,2})
\end{array}
$$




- The highlighted line is essentially:

$$
\underbrace{\color{red}{\frac{1}{\Delta x^2}}}_{\color{red}{M(1,1,0)}}\delta u_{0,1} + \underbrace{\color{red}{\frac{1}{\Delta y^2}}}_{\color{red}{M(1,1,1)}}\delta u_{1,0} + \underbrace{\color{red}{\left(-\frac{2}{\Delta x^2}-\frac{2}{\Delta y^2}\right)}}_{\color{red}{M(1,1,2)}}\delta u_{1,1} + \underbrace{\color{red}{\frac{1}{\Delta y^2}}}_{\color{red}{M(1,1,3)}}\delta u_{1,2} + \underbrace{\color{red}{\frac{1}{\Delta x^2}}}_{\color{red}{M(1,1,4)}}\delta u_{2,1}
$$

- We denote them as $M(i,j,0), M(i,j,1), \cdots, M(i,j,4)$, where
  - $M(i,j,0)$ for $N_{i-1,j}$, multiplied with $\delta u_{i-1,j}$
  - $M(i,j,1)$ for $N_{i,j-1}$, multiplied with $\delta u_{i,j-1}$
  - $M(i,j,2)$ for $N_{i,j}$, multiplied with $\delta u_{i,j}$
  - $M(i,j,3)$ for $N_{i,j+1}$, multiplied with $\delta u_{i,j+1}$
  - $M(i,j,4)$ for $N_{i+1,j}$, multiplied with $\delta u_{i+1,j}$
- Belonging to $E_{i,j}$ , they are all arranged at $\boldsymbol{A}(\text{row of } (i \times Ny + (j+1)))$
- The rest is just to determine the column number of the coefficients



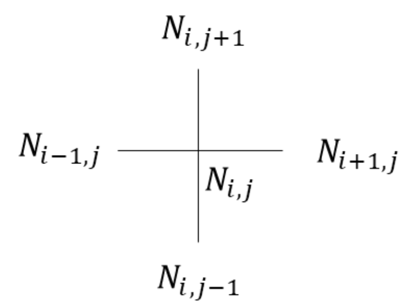
# Denotation for sparse matrix (4)

- Matrix form

$$
\begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0\\
0 & \frac{1}{\Delta x^2} & 0 & \frac{1}{\Delta y^2} & -\frac{2}{\Delta x^2}-\frac{2}{\Delta y^2} & \frac{1}{\Delta y^2} & 0 & \frac{1}{\Delta x^2} & 0\\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
\delta u_{0,0}\\ \delta u_{0,1}\\ \delta u_{0,2}\\ \delta u_{1,0}\\ \delta u_{1,1}\\ \delta u_{1,2}\\ \delta u_{2,0}\\ \delta u_{2,1}\\ \delta u_{2,2}
\end{bmatrix}
= -
\begin{array}{c}
\begin{bmatrix}
0\\ 0\\ 0\\ 0\\ \left(\frac{u_{0,1}-2u_{1,1}+u_{2,1}}{\Delta x^2}+\frac{u_{1,0}-2u_{1,1}+u_{1,2}}{\Delta y^2}\right)\\ 0\\ 0\\ 0\\ 0
\end{bmatrix}
\end{array}
\begin{array}{c}
(E_{0,0})\\ (E_{0,1})\\ (E_{0,2})\\ (E_{1,0})\\ (E_{1,1})\\ (E_{1,2})\\ (E_{2,0})\\ (E_{2,1})\\ (E_{2,2})
\end{array}
$$

Jacobian matrix $\boldsymbol{A}$ Unknown vector $\boldsymbol{\delta u}$ Residual vector $-\boldsymbol{R}$

- We have five coefficients for the central difference stencils in $E_{i,j}$:
  - We denote them as $M(i,j,0), M(i,j,1), \cdots, M(i,j,4)$
    Belong to $E_{i,j}$, they are all arranged at $\boldsymbol{A}(\text{row of } (i \times Ny + (j+1)))$, where
    - $M(i,j,0)$ for $N_{i-1,j}$, multiplied with $\delta u_{i-1,j}$
    - $M(i,j,1)$ for $N_{i,j-1}$, multiplied with $\delta u_{i,j-1}$
    - $M(i,j,2)$ for $N_{i,j}$, multipliedwith $\delta u_{i,j}$
    - $M(i,j,3)$ for $N_{i,j+1}$, multiplied with $\delta u_{i,j+1}$
    - $M(i,j,4)$ for $N_{i+1,j}$, multiplied with $\delta u_{i+1,j}$




  - Taking $M(i,j,0)$ as an example, to multiply with $\delta u_{i-1,j}$,
    it's column number of $M(i,j,0)$ should be the same with the row number of $\delta u_{i-1,j}$,
    which is: $\boldsymbol{A}(\text{column of } ((i-1) \times Ny + (j+1)))$
  - For example, $M(1,1,0)$ is allocated at $\boldsymbol{A}(\text{column of } ((1-1) \times Ny + (1+1))) =$
    $\boldsymbol{A}(\text{column of } 2)$ as highlighted.

# Denotation for sparse matrix (5)

- We notice that there are a lot of zeros in the Jacobian matrix and residual vector:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{\Delta x^2} & 0 & \frac{1}{\Delta y^2} & -\frac{2}{\Delta x^2}-\frac{2}{\Delta y^2} & \frac{1}{\Delta y^2} & 0 & \frac{1}{\Delta x^2} & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \delta u_{0,0} \\ \delta u_{0,1} \\ \delta u_{0,2} \\ \delta u_{1,0} \\ \delta u_{1,1} \\ \delta u_{1,2} \\ \delta u_{2,0} \\ \delta u_{2,1} \\ \delta u_{2,2} \end{bmatrix} = - \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \left( \frac{u_{0,1} - 2u_{1,1} + u_{2,1}}{\Delta x^2} + \frac{u_{1,0} - 2u_{1,1} + u_{1,2}}{\Delta y^2} \right) \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Jacobian matrix $\boldsymbol{A}$ Unknown vector $\boldsymbol{\delta u}$ Residual vector $-\boldsymbol{R}$

- To summaries, we have the following denotation:
  - Unknown vector entry at $N_{i,j}$: $\delta u_{i,j} \rightarrow \delta u(i \times Ny + (j+1))$
  - Residual vector at $N_{i,j}$ : $R_{i,j} \rightarrow R(i \times Ny + (j+1))$
  - Jacobian matrix:
    - $M(i,j,0)$ for $N_{i-1,j} \rightarrow A(\text{row of } (i \times Ny + (j+1)), \text{column of } ((i-1) \times Ny + (j+1)))$
    - $M(i,j,1)$ for $N_{i,j-1} \rightarrow A(\text{row of } (i \times Ny + (j+1)), \text{column of } (i \times Ny + ((j-1)+1)))$
    - $M(i,j,2)$ for $N_{i,j} \rightarrow A(\text{row of } (i \times Ny + (j+1)), \text{column of } (i \times Ny + (j+1)))$
    - $M(i,j,3)$ for $N_{i,j+1} \rightarrow A(\text{row of } (i \times Ny + (j+1)), \text{column of } (i \times Ny + ((j+1)+1)))$
    - $M(i,j,4)$ for $N_{i+1,j} \rightarrow A(\text{row of } (i \times Ny + (j+1)), \text{column of } ((i+1) \times Ny + (j+1)))$

# Transient heat transfer problem: Definition

- The governing equation of the problem is:

$$\frac{\partial u}{\partial t} = \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) \text{ in } (0,1) \times (0,1)$$

- We take a manufactured solution which satisfies the above equation:

$$u(x, y, 0) = -\exp(-100(x^2 + y^2))$$

- Consider the Trapezoidal rule for temporal discretization:

$$\frac{\boldsymbol{I}}{\Delta t}(\boldsymbol{u}^{n+1} - \boldsymbol{u}^n) = 0.5\boldsymbol{A}\boldsymbol{u}^{n+1} + 0.5\boldsymbol{A}\boldsymbol{u}^n$$

where $\boldsymbol{I}$ is an identity matrix, $\boldsymbol{A}$ is the same matrix we derived previously for central difference approximation of the diffusion operator $\Delta$

# Newton-Raphson iteration(1)

- Given the governing equation, we want to solve $\boldsymbol{u}^{n+1}$ :

$$\frac{\boldsymbol{I}}{\Delta t}(\boldsymbol{u}^{n+1}-\boldsymbol{u}^{n})=0.5\boldsymbol{A}\boldsymbol{u}^{n+1}+0.5\boldsymbol{A}\boldsymbol{u}^{n}$$

- Rearrange the equation:

$$\left(\frac{\boldsymbol{I}}{\Delta t}-0.5\boldsymbol{A}\right)\boldsymbol{u}^{n+1}=\left(\frac{\boldsymbol{I}}{\Delta t}+0.5\boldsymbol{A}\right)\boldsymbol{u}^{n}$$

- With known value of $\boldsymbol{u}^{n}$, we denote $\boldsymbol{b}=\left(\frac{\boldsymbol{I}}{\Delta t}+0.5\boldsymbol{A}\right)\boldsymbol{u}^{n}$. Therefore we have:

$$\left(\frac{\boldsymbol{I}}{\Delta t}-0.5\boldsymbol{A}\right)\boldsymbol{u}^{n+1}=\boldsymbol{b}$$

- Rewrite this with Newton-Raphson iteration:

$$\left(\frac{\boldsymbol{I}}{\Delta t}-0.5\boldsymbol{A}\right)(\boldsymbol{u}_k^{n+1}+\delta\boldsymbol{u}_k^{n+1})=\boldsymbol{b}$$

where $k$ denotes the number of iterations.

# Newton-Raphson iteration (2)

- As derived in the last page:

$$\left(\frac{\boldsymbol{I}}{\Delta t} - 0.5\boldsymbol{A}\right)(\boldsymbol{u}_k^{n+1} + \delta \boldsymbol{u}_k^{n+1}) = \boldsymbol{b}$$

- The update can be calculated as:

$$\left(\frac{\boldsymbol{I}}{\Delta t} - 0.5\boldsymbol{A}\right)(\delta \boldsymbol{u}_k^{n+1}) = \boldsymbol{b} - \left(\frac{\boldsymbol{I}}{\Delta t} - 0.5\boldsymbol{A}\right)\boldsymbol{u}_k^{n+1}$$

- For linear problem, the solution converges in one single iteration. This can be derived as follows:

$$\boldsymbol{M}(\boldsymbol{x}_0 + \boldsymbol{\delta x}_0) = \boldsymbol{b}$$

$$\boldsymbol{\delta x}_0 = \boldsymbol{M}^{-1}\boldsymbol{b} - \boldsymbol{x}_0$$

$$\boldsymbol{x}_1 = \boldsymbol{x}_0 + \boldsymbol{\delta x}_0 = \boldsymbol{M}^{-1}\boldsymbol{b}$$

After updating the solution, we substitute $\boldsymbol{x}_1$ in to the governing equation, we find:

$$\boldsymbol{M}\boldsymbol{x}_1 = \boldsymbol{M}\boldsymbol{M}^{-1}\boldsymbol{b} = \boldsymbol{b}$$

Therefore $\boldsymbol{x}_1$ is the exact solution.
However, this is not the case for nonlinear problems $\boldsymbol{M}(\boldsymbol{u}_k)\boldsymbol{u}_k = \boldsymbol{b}(\boldsymbol{u}_k)$.

# Newton-Raphson iteration (3)

- With the convergence in one iteration, we have:

  Initial guess as solution at previous time step: $\boldsymbol{u}_0^{n+1} = \boldsymbol{u}^n$

  Ture solution after one iteration: $\boldsymbol{u}_1^{n+1} = \boldsymbol{u}_0^{n+1} + \delta\boldsymbol{u}_0^{n+1} = \boldsymbol{u}^{n+1}$

- Substitute into the first iteration:

$$\left(\frac{\boldsymbol{I}}{\Delta t} - 0.5\boldsymbol{A}\right)(\delta\boldsymbol{u}_0^{n+1}) = \boldsymbol{b} - \left(\frac{\boldsymbol{I}}{\Delta t} - 0.5\boldsymbol{A}\right)\boldsymbol{u}_0^{n+1}$$

We get:

$$\left(\frac{\boldsymbol{I}}{\Delta t} - 0.5\boldsymbol{A}\right)(\delta\boldsymbol{u}^{n+1}) = \boldsymbol{b} - \left(\frac{\boldsymbol{I}}{\Delta t} - 0.5\boldsymbol{A}\right)\boldsymbol{u}^n$$

$$\boldsymbol{u}^{n+1} = \boldsymbol{u}^n + \delta\boldsymbol{u}^{n+1}$$

Substituting $\boldsymbol{b} = \left(\frac{\boldsymbol{I}}{\Delta t} + 0.5\boldsymbol{A}\right)\boldsymbol{u}^n$,we finally get the linear matrix form:

$$\left(\frac{\boldsymbol{I}}{\Delta t} - 0.5\boldsymbol{A}\right)(\delta\boldsymbol{u}^{n+1}) = \boldsymbol{A}\boldsymbol{u}^n$$

# Your tasks

- To summarize, the linear systems we need to solve are:
  - Steady-state heat transfer

$$\boxed{\boldsymbol{A}}\,\delta\boldsymbol{u}_k = -\boxed{\boldsymbol{A}\boldsymbol{u}_k}, \qquad \boldsymbol{u}_{k+1} = \boldsymbol{u}_k + \delta\boldsymbol{u}_k$$

  - Transient heat transfer

$$\boxed{\left(\frac{\boldsymbol{I}}{\Delta t} - 0.5\boldsymbol{A}\right)}(\delta\boldsymbol{u}^{n+1}) = \boxed{\boldsymbol{A}\boldsymbol{u}^n}, \qquad \boldsymbol{u}^{n+1} = \boldsymbol{u}^n + \delta\boldsymbol{u}^{n+1}$$

- The matrices and vectors in the red boxes are to be formed in the code

- In the code, the matrices and vectors are initialized as random values. You need to change them to correct values.

- To be more specific, complete functions in Laplace.cc:
  - computeMatrix for $\boldsymbol{A}$ in steady-state heat transfer
  - computeTransientMatrix for $\left(\frac{\boldsymbol{I}}{\Delta t} - 0.5\boldsymbol{A}\right)$ in transient heat transfer
  - computeDiffusion for $\boldsymbol{A}\boldsymbol{u}_k$ as residual vector
  - applyBc for boundary conditions

# On assigning values to matrices and vectors

- Taking steady-state heat transfer problem as an example:

- For row of matrices $\boldsymbol{A}$ corresponding to point $u_{i,j}$ :

Formulation: $$\frac{\delta u_{0,1} - 2\delta u_{1,1} + \delta u_{2,1}}{\Delta x^2} + \frac{\delta u_{1,0} - 2\delta u_{1,1} + \delta u_{1,2}}{\Delta y^2}$$

Coefficients: $$\begin{bmatrix} \cdots & \frac{1}{\Delta x^2} & \cdots & \frac{1}{\Delta y^2} & -\frac{2}{\Delta x^2} - \frac{2}{\Delta y^2} & \frac{1}{\Delta y^2} & \cdots & \frac{1}{\Delta x^2} & \cdots \end{bmatrix}$$

Corresponding Variables: $\delta u_{i-1,j}$ $\delta u_{i,j-1}$ $\delta u_{i,j}$ $\delta u_{i,j+1}$ $\delta u_{i+1,j}$

Coefficients can be accessed: $\mathrm{M(i,j,0)}$ $\mathrm{M(i,j,1)}$ $\mathrm{M(i,j,2)}$ $\mathrm{M(i,j,3)}$ $\mathrm{M(i,j,4)}$

- For Solution $u_{i,j}$, increment $\delta u_{i,j}$ and $R_{i,j}$:
  - $u_{i,j}$ can be accessed through u(i,j) in the function
  - $\delta u_{i,j}$ can be accessed through du(i,j) in the function
  - $R_{i,j}$ can be accessed through R(i,j) in the function

# Files in the code

- Input: Laplace.in contains 3 numbers of: Nx, Ny, dt
- Source code: Base.h, Base.cc, Laplace.cc
- Output and reference data:
  - SteadyA.dat, SteadyA_ref.dat.: $\boldsymbol{A}$ matrix for the steady state problem in the format of:
    i, j, M(i,j,0), M(i,j,1), M(i,j,2), M(i,j,3), M(i,j,4),
    and reference values for mesh of $33 \times 33$
  - SteadyR0.dat, SteadyR0_ref.dat: initial residual vector in the format of:
    i, j, R(i,j)
    and reference values for mesh of $33 \times 33$
  - SteadySolError.dat, TransSolError.dat: error compare to exact solution in the format of:
    total grid number, L2norm of the error, L infinity norm of the error
    Instead of overwriting, adding a new line for the current run
  - Ue.vtk, steadyPhi.vtk, unsteadyPhi.vtk : exact, steady-state and final unsteady solution which can be visualized through paraview.
    Open .vtk as a text, you can find 3 columns matrix as coordinates, and 1 column vector as solution.
  - Run time is typed on the terminal