CE 601: Numerical Methods
Lecture 6

Course Coordinator:
Dr. Suresh A. Kartha,
Associate Professor,
Department of Civil Engineering,
IIT Guwahati.

Banded Matrices

- Mostly we have dealt now coefficient matrix. [A] having n X n elements
  - ✓ Most of the elements are non-zero.
  - ✓ Such matrices are dense.
- However for many engineering and scientific problems, the coefficient matrices may not be fully filled with non-zeroes. There may be many zeroes.

$$\begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{pmatrix}$$

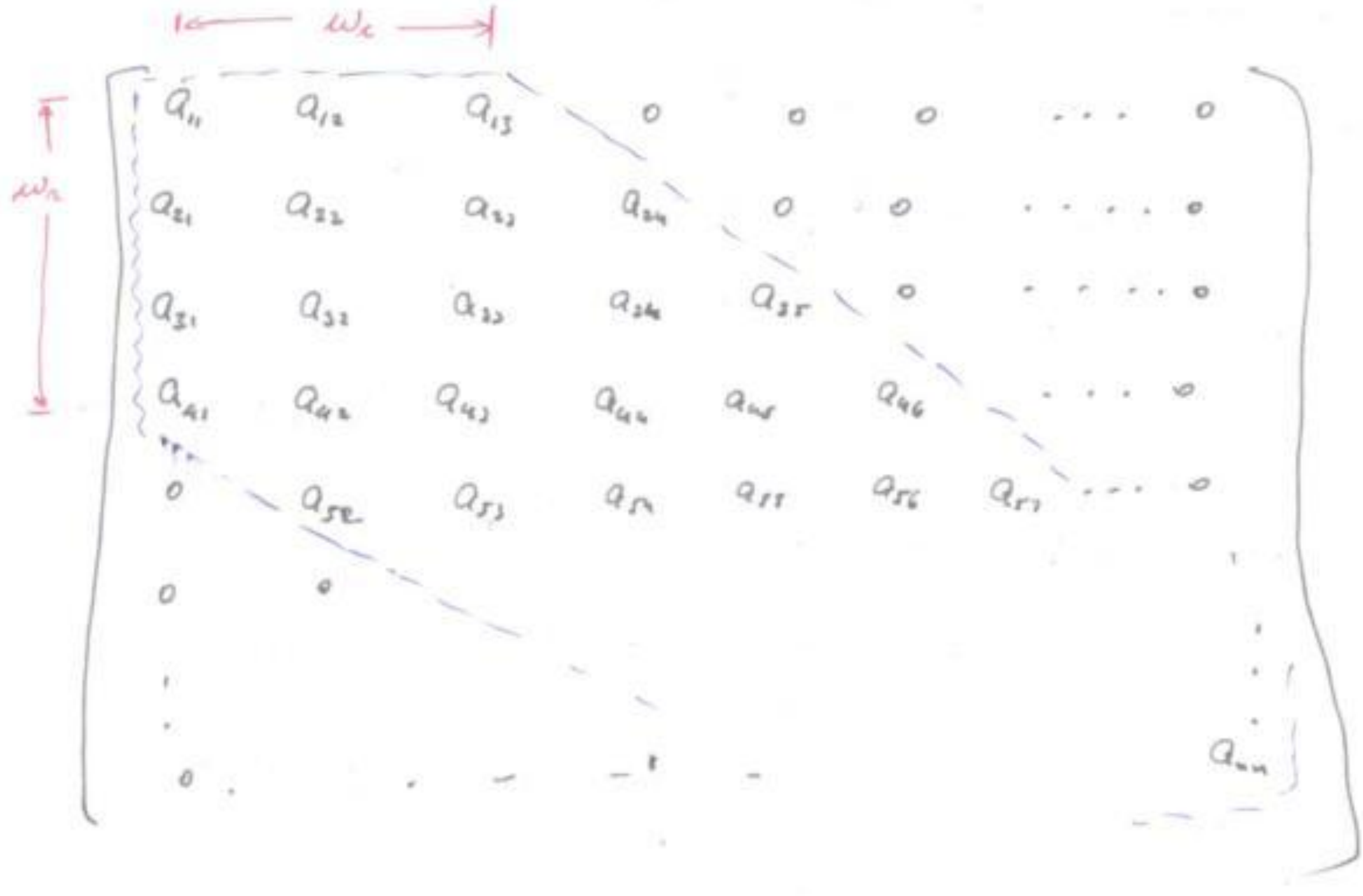
Dense Matrix

$$\begin{pmatrix} a_{11} & a_{12} & 0 & \cdots & 0 \\ a_{21} & a_{22} & a_{23} & \cdots & 0 \\ 0 & a_{32} & a_{33} & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & a_{n(n-1)} & a_{nn} \end{pmatrix}$$

Sparse Matrix

- If the coefficient matrix have many zero elements, there you can derive suitable methods to efficiently solve the system.
    - ❖ Recall in Gauss elimination, you performed (n-1) row operations.
    - ❖ If there are many zeroes, we can avoid unnecessary sub-steps in row operations.
- For dense matrices, you may go for Gauss elimination or LU decomposition.
- Banded matrices are sparse matrices that follow certain along diagonal elements. Mostly they are diagonally dominant.

- If your coefficient matrix in diagonally dominant or contains values along main diagonal and lines parallel to main diagonal.



- Bandwidth, $w = w_c + w_r - 1$

$$a_{ij} = \begin{cases} 0 & ; j \geq i + w_c \\ 0 & ; i \geq j + w_r \\ a_{ij} & ; \text{else} \end{cases}$$

- <u>Tridiagonal Matrix</u>

  - In a banded matrix of bandwidth *=3*

    $w_r = 2$, $w_c = 2$

    $[T]\{x\} = \{b\}$

- Thomas Algorithm for Tridiagonal Matrix

$$\begin{pmatrix} a_{11} & a_{12} & 0 & 0 & 0 \\ a_{21} & a_{22} & a_{23} & 0 & 0 \\ 0 & a_{32} & a_{33} & a_{11} & 0 \\ 0 & 0 & a_{43} & a_{44} & a_{45} \\ 0 & 0 & 0 & a_{54} & a_{55} \end{pmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{Bmatrix} = \begin{Bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \end{Bmatrix}$$

- Recall the Gauss elimination algorithm
  - ✓ In the first step, only 2nd row requires an operation
  - ✓ What does this operation do?

$$a_{2j}^{(1)} = a_{2j} - a_{21}/a_{11} \; a_{1j}$$

  - ✓ Due to this change in 2nd row:

$$\begin{bmatrix} 0 & a_{22} - a_{21}/a_{11} \; a_{12} & a_{23} & 0 & 0 \end{bmatrix}$$

i.e., only the diagonal element got changed.

- Similarly, At each step '*k*'
  - Only the $(k+1)^{th}$ row is modified.
  - Only the diagonal element $a_{(k+1)(k+1)}$ is modified.

$$a_{ii}^{(k)} = a_{ii}^{(k-1)} - (a_{i(j-1)}^{(k-1)} / a_{(i-1)(i-1)}^{(k-1)}) a_{(i-1)j}^{(k-1)}$$

$$b_{i}^{(k)} = b_{i}^{(k-1)} - (a_{i(j-1)}^{(k-1)} / a_{(i-1)(i-1)}^{(k-1)}) b_{i-1}^{(k-1)}; \text{ where } i = k+1, j = k+1$$

  - Therefore we can utilize this peculiarity for computational advantage.

- In the tridiagonal matrix
- ✓ the n X n matrix can be stored as n X 3 matrix with no zeroes.

$$[A]=\begin{pmatrix} a_{11} & a_{12} & 0 & 0 & 0 \\ a_{21} & a_{22} & a_{23} & 0 & 0 \\ 0 & a_{32} & a_{33} & a_{11} & 0 \\ 0 & 0 & a_{43} & a_{44} & a_{45} \\ 0 & 0 & 0 & a_{54} & a_{55} \end{pmatrix} \Rightarrow [A']=\begin{pmatrix} - & a'_{12} & a'_{13} \\ a'_{21} & a'_{22} & a'_{23} \\ a'_{31} & a'_{32} & a'_{33} \\ a'_{41} & a'_{42} & a'_{43} \\ a'_{51} & a'_{52} & - \end{pmatrix}$$

- ✓ The computation will be (using Gauss elimination principle) performed in such a way that first column is eliminated.

$$a'_{12} = a'_{12}$$

$$a'_{i2} = a'_{i2} - \; a'_{i1} / a'_{(i-1)2} \;\; a'_{(i-1)3}; i = 2, 3, 4, \ldots, n.$$

$$\rightarrow \{2 \times (n-1) \text{ operations}\}$$

The vector $\{b\}$

$$b_1 = b_1$$

$$b_i = b_i - \; a'_{i1} / a'_{(i-1)2} \;\; b_{i-1}; i = 2, 3, 4, \ldots, n$$

$$\rightarrow \{2 \times (n-1) \text{ operations}\}$$

- ✓ The multiplying factor at each step is

$$a'_{i1} / a'_{(i-1)2}$$

$$\rightarrow \{1 \times (n-1) \text{ operations}\}$$

- Use back-substitution,

$$x_n = b_n / a'_{n2}; \quad \rightarrow \{1 \text{ operations}\}$$

$$x_i = \frac{b_i - a'_{i3}\, x_{i+1}}{a'_{i2}}; i = n-1, n-2, ..., 2, 1.$$

$$\rightarrow \{3 \times (n-1) \text{ operations}\}$$

- From this algorithm, you can see the no. of operations involved = 8(n-1)+1 = 8n-7.

- <u>Example</u>

you can solve using the algorithm described above:

$$\begin{pmatrix} -2 & 1 & 0 & 0 \\ 1 & -4 & 1 & 0 \\ 0 & 1 & -4 & 1 \\ 0 & 0 & 1 & -2 \end{pmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{Bmatrix} = \begin{Bmatrix} 3 \\ 1 \\ 2 \\ -2 \end{Bmatrix}$$

$$[A'] = \begin{pmatrix} - & -2 & 1 \\ 1 & -4 & 1 \\ 1 & -4 & 1 \\ 1 & -2 & - \end{pmatrix}$$

$$a'^{(1)}_{i2} = a'_{i2} - \left(a'_{i1} / a'_{(i-1)2} \right) a'_{(i-1)3}; i = 2, 3, 4$$

$$b^{(1)}_{i} = b_i - \left(a'_{i1} / a'_{(i-1)2} \right) b_{i-1}; i = 2, 3, 4$$

Also perform back-substitution,

$$x_4 = b_4 \, / \, a'_{i2}$$

$$x_i = \frac{b_i - a'_{i2} \, x_{i+1}}{a'_{i2}}$$

i.e., $a'_{12} = a'_{13} = -2;$ $\quad b_1 = 3$

$a'_{23} = -4 - (1/-2) \times 1 = -3.5;$ $\quad b_2 = 1 - (1/-2) \times 3 = 2.5$

$a'_{32} = -4 - (1/-3.5) \times 1 = -3.7143;$ $\quad b_3 = 2 - (1/-3.5) \times 2.5 = 2.7143$

$a'_{42} = -2 - (1/-3.7143) \times 1 = -1.7308;$ $\quad b_4 = -2 - (1/-3.7143) \times 2.7143 = -1.2692$

Back-substitution

$$x_4 = b_4 / a'_{42} = -1.2692 / -1.7308 = 0.7333$$

$$x_3 = (b_3 - a'_{33} x_4) / a'_{32}$$

$$= (2.7143 - 1 \times 0.7333) / -3.7143$$

$$= -0.5333$$

Now, complete these.