

## Lecture 09-10

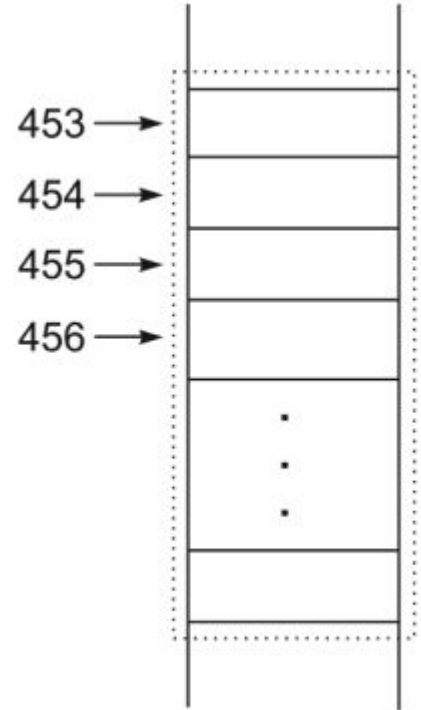
- Multidimensional Arrays

# Array: Definition

An array is

- ***finite*** – contains only a limited number of elements,
- ***indexed*** – all the elements are stored one-by-one in contiguous locations of the computer memory in a linear ordered fashion, and
- ***homogeneous*** – all the elements are of the same data type

collection (that's why a composite data structure) of data elements.



# Multidimensional Array

If only *more than one index* is required to reference all the elements in an array, such an array is termed as multi-dimensional array.

- In this course, we will mainly focus on two- and three- dimensional arrays.

## Two-Dimensional Array

Alternatively termed as Matrices are a collection of *homogeneous* elements where the elements are ordered in a number of rows and columns.

- E.g., An  $m \times n$  order matrix, where  $m$  is the number of rows and  $n$  is the number of columns.

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} & \cdots & a_{1n} \\ a_{21} & a_{22} & a_{23} & a_{24} & \cdots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & a_{m3} & a_{m4} & \cdots & a_{mn} \end{bmatrix}$$

- The subscripts of an arbitrary element  $a_{ij}$  denotes  $i^{th}$  row and  $j^{th}$  column.

# Memory (Physical) Representation of a 2-D Array

Like 1-D arrays, 2-D arrays are also stored in a contiguous memory locations. Conventionally, there are two ways to store elements of a 2-D array.

- **Row-major order**

Elements are stored row-wise one after another.

- **Column-major order**

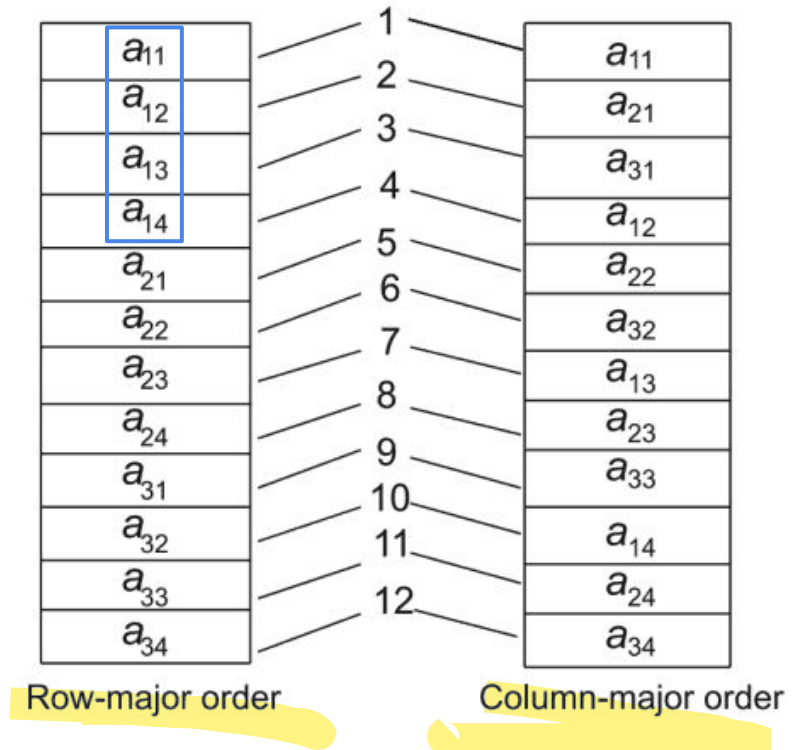
Elements are stored column-wise one after another.

# Memory (Physical) Representation of a 2-D Array

## Row-major Order

Consider a 2-D matrix A of an order of 3 x 4

$a_{11}$	$a_{12}$	$a_{13}$	$a_{14}$
$a_{21}$	$a_{22}$	$a_{23}$	$a_{24}$
$a_{31}$	$a_{32}$	$a_{33}$	$a_{34}$

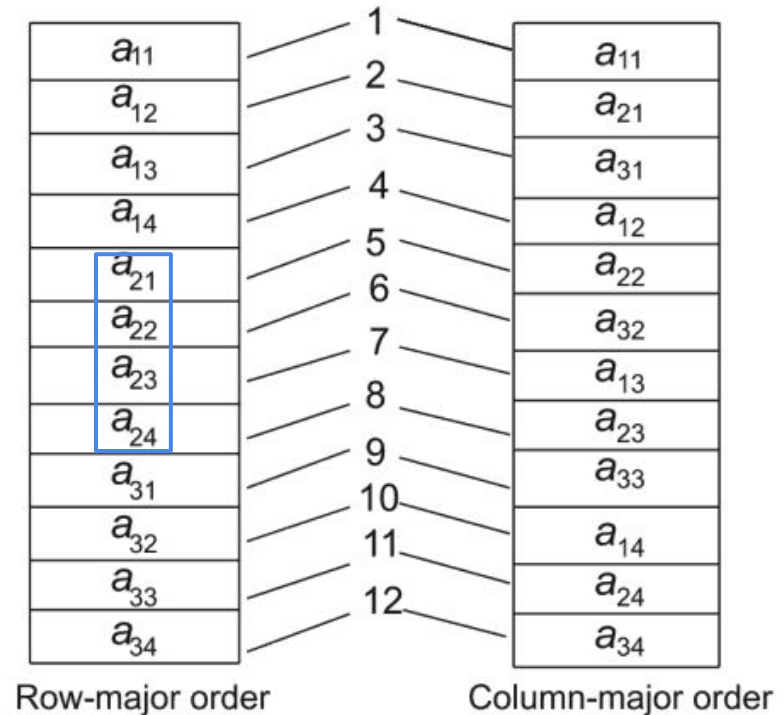
 $_{3 \times 4}$ 

# Memory (Physical) Representation of a 2-D Array

## Row-major Order

Consider a 2-D matrix A of an order of 3 x 4

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \end{bmatrix}_{3 \times 4}$$

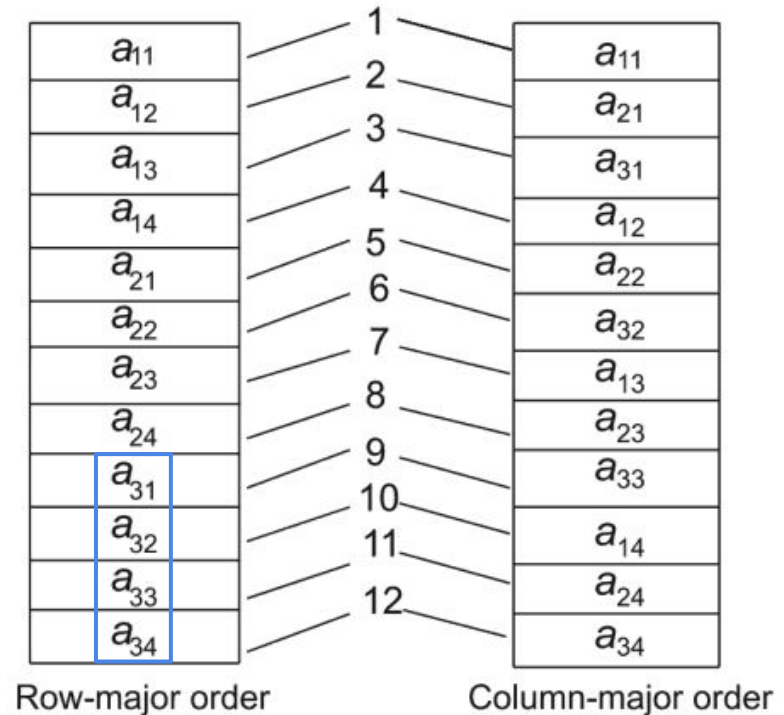


# Memory (Physical) Representation of a 2-D Array

## Row-major Order

Consider a 2-D matrix A of an order of 3 x 4

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \end{bmatrix}_{3 \times 4}$$



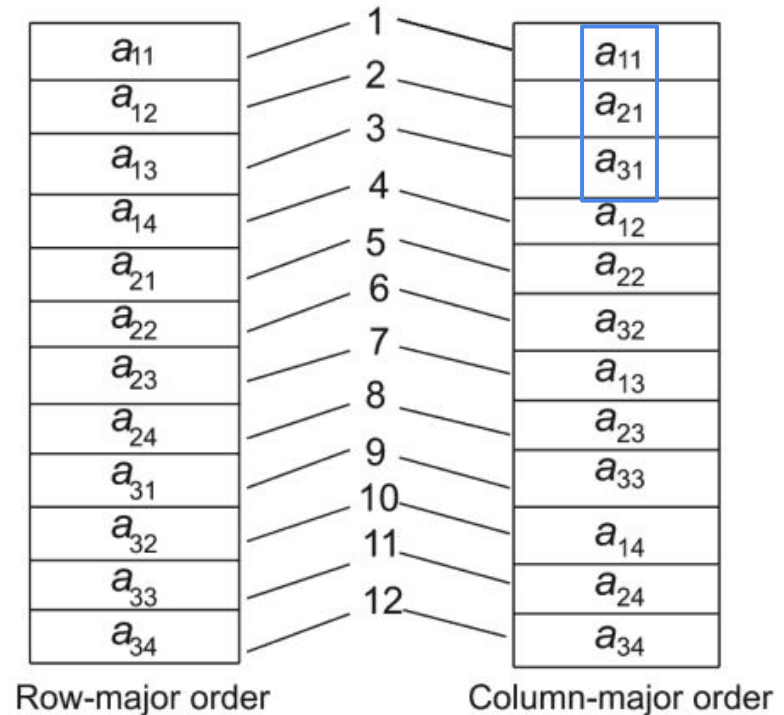


# Memory (Physical) Representation of a 2-D Array

## Column-major Order

Consider a 2-D matrix A of an order of 3 x 4

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \end{bmatrix}_{3 \times 4}$$

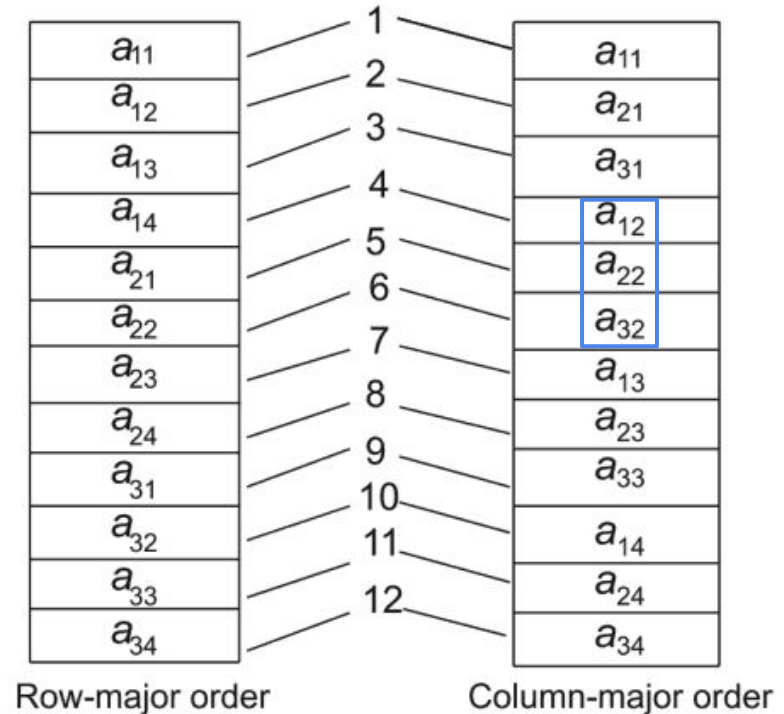


# Memory (Physical) Representation of a 2-D Array

## Column-major Order

Consider a 2-D matrix A of an order of 3 x 4

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \end{bmatrix}_{3 \times 4}$$

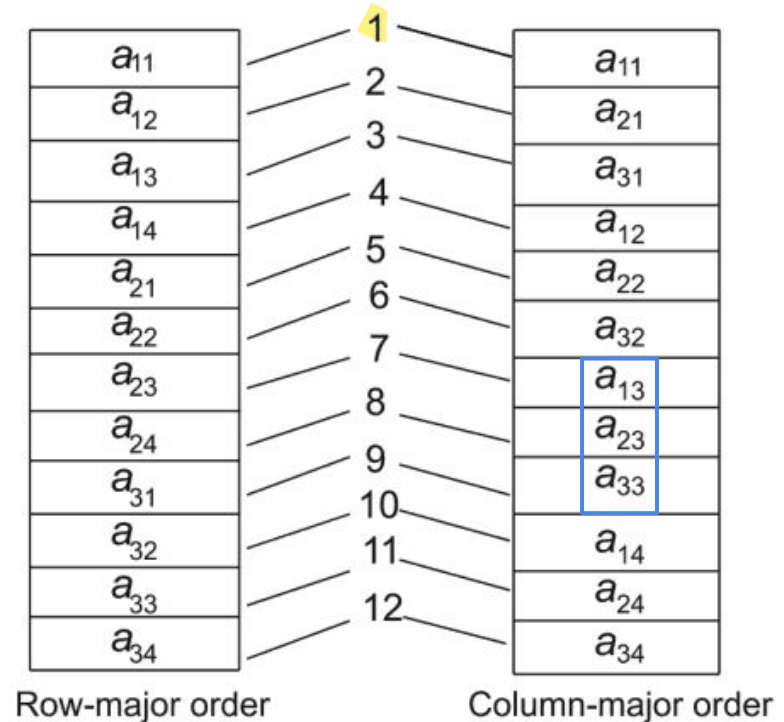


# Memory (Physical) Representation of a 2-D Array

## Column-major Order

Consider a 2-D matrix A of an order of 3 x 4

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \end{bmatrix}_{3 \times 4}$$

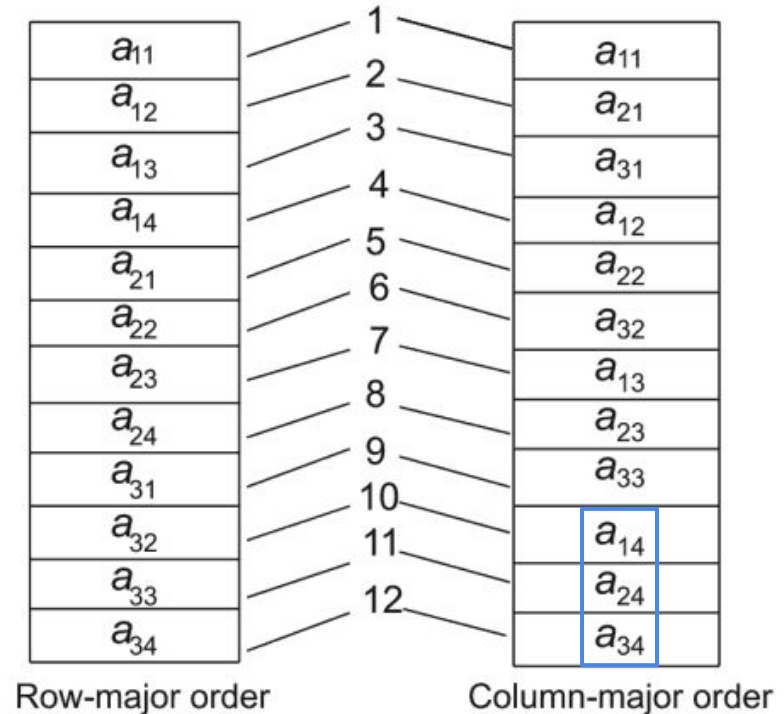


# Memory (Physical) Representation of a 2-D Array

## Column-major Order

Consider a 2-D matrix A of an order of 3 x 4

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \end{bmatrix}_{3 \times 4}$$



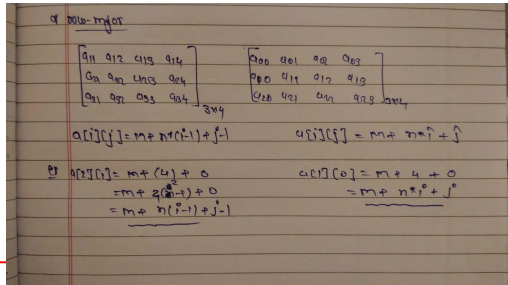
# Two-Dimensional Array: Logical View to Physical Representation

Take small example and calculate

- Logically, a matrix appears as two-dimensional, however, physically, it is stored in a linear fashion.
- So, in order to map from logical view to physical structure, we need an indexing formula.
- Row-major Order**
  - Assuming a matrix of the order of  $m \times n$  and the base address is  $M$ , address of an element  $a_{ij}$  can be obtained as:

Address ( $A[i][j]$ ) = storing all the elements in the first  $(i - 1)$  rows + the number of elements in the  $i^{\text{th}}$  row up to the  $j^{\text{th}}$  column

$$= M + (i - 1) * n + (j - 1)$$



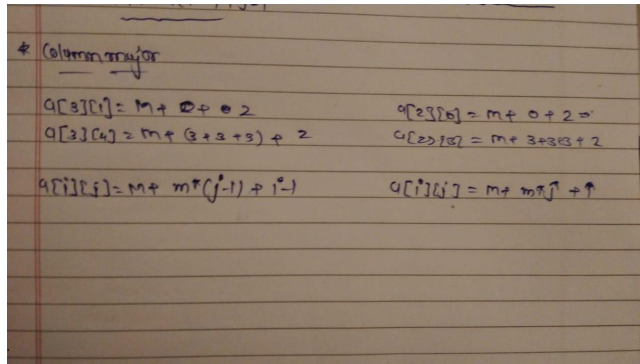
# Two-Dimensional Array: Logical View to Physical Representation

- Column-major Order

- Assuming a matrix of the order of  $m \times n$  and the base address is  $M$ , address of an element  $a_{ij}$  can be obtained as:

Address ( $A[i][j]$ ) = storing all the elements in the first  $(j - 1)$  columns + the number of elements in the  $j^{\text{th}}$  column up to the  $i^{\text{th}}$  row

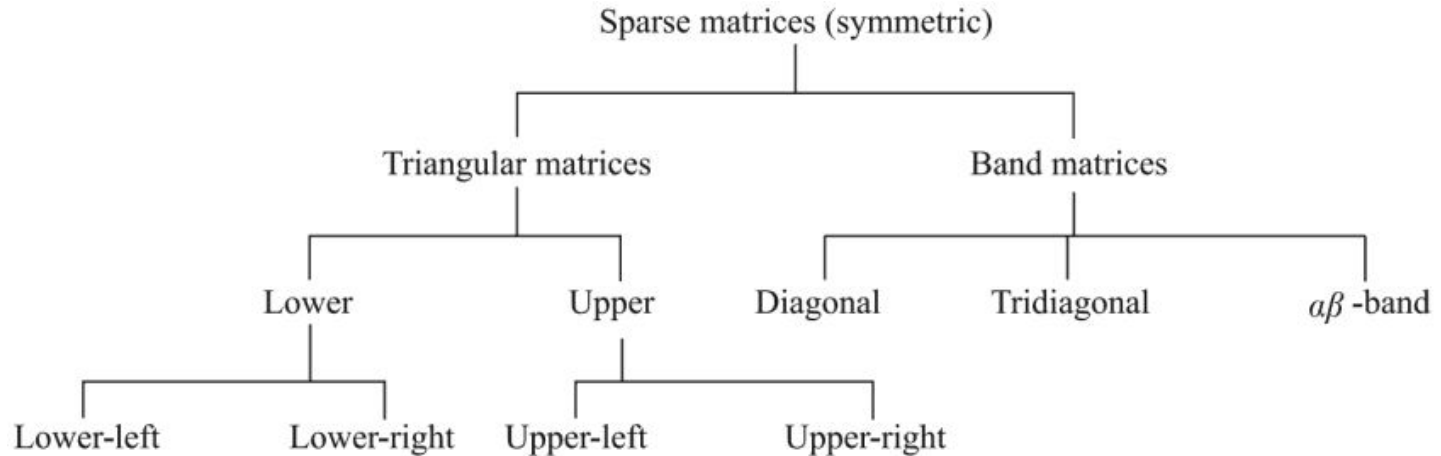
$$= M + (j - 1) * m + (i - 1)$$



# Sparse Matrices

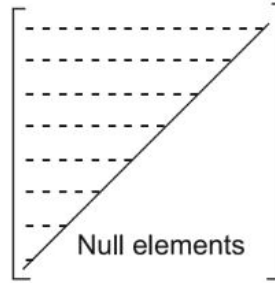
A sparse matrix is a 2-D array where most of the elements have the value **null**.

- Some known sparse matrices are characterized as follows.

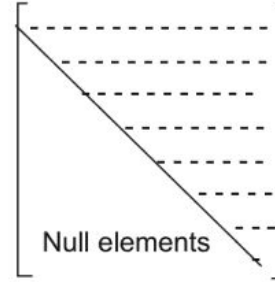


# Sparse Matrices: Triangular Matrices

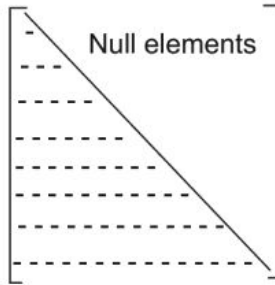
Some known sparse matrices are characterized as follows.



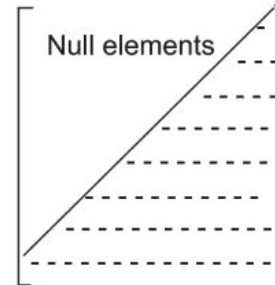
Upper-left triangular



Upper-right triangular



Lower-left triangular



Lower-right triangular



# Sparse Matrices

A sparse matrix is a 2-D array where most of the elements have the value **null**.

- Storing **null** values is the wastage of memory
- Need to come up with a technique to store non-**null** elements only.
  - One approach is to use Linked-list, we will study this topic in the coming weeks.
  - But, what about arrays?

# Memory Representation of a Lower Triangular Matrix

## Row-major Order

Assuming the base address is **M**, address of an element  $a_{ij}$ ,  $1 \leq i, j \leq n$ , can be obtained as:

Address ( $A[i][j]$ ) = storing all the elements in the first  $(i - 1)$  rows +  
the number of elements in the  $i^{\text{th}}$  row up to the  $j^{\text{th}}$  column

$$= M + \frac{i(i-1)}{2} + j - 1$$

$$\begin{bmatrix} a_{11} & & & & \\ a_{21} & a_{22} & & & \\ a_{31} & a_{32} & a_{33} & & \\ \vdots & \vdots & \vdots & & \\ a_{n1} & a_{n2} & a_{n3} & \cdots & a_{nn} \end{bmatrix}_{n \times n}$$

# Memory Representation of a Lower Triangular Matrix

## Column-major Order

Assuming the base address is **M**, address of an element  $a_{ij}$ ,  $1 \leq i, j \leq n$ , can be obtained as:

Address ( $A[i][j]$ ) = storing all the elements in the first  $(j - 1)$  columns +  
the number of elements up to the  $i^{\text{th}}$  row in the  $j^{\text{th}}$  column

$$= M + (j - 1) \times \left( n - \frac{j}{2} \right) + i - 1$$

$$\begin{bmatrix} a_{11} & & & & \\ a_{21} & a_{22} & & & \\ a_{31} & a_{32} & a_{33} & & \\ \vdots & \vdots & \vdots & & \\ a_{n1} & a_{n2} & a_{n3} & \cdots & a_{nn} \end{bmatrix}_{n \times n}$$

Handwritten derivation of the address formula for a lower triangular matrix in column-major order:

$$\begin{aligned} \text{Address of } a_{ij} &= M + (n-1) + (n-2) + \dots + (n-j+1) + (i-j) \\ &= M + \left[ (n-1) + (n-2) + \dots + (n-j+1) \right] + (i-j) \\ &= M + \left[ \frac{(n-1 + n-j+1) \times (j-1)}{2} \right] + (i-j) \\ &= M + \left[ \frac{(2n-j) \times (j-1)}{2} \right] + (i-j) \\ &= M + (n-j) \times (j-1) + (i-j) \\ &= M + (n-j) \times (j-1) + (i-1) \end{aligned}$$

# Memory Representation of a Lower Triangular Matrix

## Column-major Order

Assuming the base address is **M**, address of an element  $a_{ij}$ ,  $1 \leq i, j \leq n$ , can be obtained as:

$$\begin{bmatrix} a_{11} & & & & \\ a_{21} & a_{22} & & & \\ a_{31} & a_{32} & a_{33} & & \\ \vdots & \vdots & \vdots & & \\ a_{n1} & a_{n2} & a_{n3} & \cdots & a_{nn} \end{bmatrix}_{n \times n}$$

$$\begin{aligned} \text{Address (A[i][j])} &= \text{Number of elements up to the } a_{ij} \text{ element} \\ &= \text{Total number of elements in the first } j-1 \text{ columns} \\ &\quad + \text{number of elements up to the } i\text{th row in the } j\text{th column} \\ &= [n + (n-1) + (n-2) + \cdots + (n-j+2)] + (i-j+1) \\ &= \{n \times (j-1) - [1 + 2 + 3 + \cdots + (j-2) + (j-1)] + i\} \\ &= n \times (j-1) - \frac{j(j-1)}{2} + i \\ &= (j-1) \times \left( n - \frac{j}{2} \right) + i \end{aligned}$$

# Memory Representation of an Upper Triangular Matrix

## Row-major Order

Assuming the base address is **M**, address of an element  $a_{ij}$ ,  $1 \leq i, j \leq n$ , can be obtained as:

Address ( $A[i][j]$ ) = storing all the elements in the first  $(i - 1)$  rows +  
the number of elements in the  $i^{\text{th}}$  row up to the  $j^{\text{th}}$  column

$$= M + (i - 1) \times \left( n - \frac{i}{2} \right) + j - 1$$

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ & a_{22} & a_{23} & \cdots & a_{2n} \\ & & a_{33} & \cdots & a_{3n} \\ & & & \ddots & \\ & & & & a_{nn} \end{bmatrix}_{n \times n}$$

# Memory Representation of a Lower Triangular Matrix

## Column-major Order

Assuming the base address is **M**, address of an element  $a_{ij}$ ,  $1 \leq i, j \leq n$ , can be obtained as:

Address ( $A[i][j]$ ) = storing all the elements in the first  $(j - 1)$  columns +  
the number of elements up to the  $i^{\text{th}}$  row in the  $j^{\text{th}}$  column

$$= M + \frac{j(j-1)}{2} + i - 1$$

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ & a_{22} & a_{23} & \cdots & a_{2n} \\ & & a_{33} & \cdots & a_{3n} \\ & & & \ddots & \\ & & & & a_{nn} \end{bmatrix}_{n \times n}$$

# Memory Representation of an Upper Triangular Matrix

## Row-major Order

Assuming the base address is **M**, address of an element  $a_{ij}$ ,  $1 \leq i, j \leq n$ , can be obtained as:

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ & a_{22} & a_{23} & \cdots & a_{2n} \\ & & a_{33} & \cdots & a_{3n} \\ & & & \ddots & \\ & & & & a_{nn} \end{bmatrix}_{n \times n}$$

$$\begin{aligned} \text{Address (A[i][j])} &:= \text{Number of elements up to the } a_{ij} \text{ element} \\ &= \text{Total number of elements in the first } (i - 1) \text{ rows} \\ &\quad + \text{number of elements up to the } j\text{th column in the } i\text{th row} \\ &= n + (n - 1) + (n - 2) + \cdots + (n - i + 2) + (j - i + 1) \\ &= n \times (i - 1) - [1 + 2 + 3 + \cdots + (i - 2) + (i - 1)] + j \\ &= n \times (i - 1) - \frac{i(i - 1)}{2} + i \\ &= (i - 1) \times \left( n - \frac{i}{2} \right) + j \end{aligned}$$

# Memory Representation of a Lower Triangular Matrix

## Column-major Order

Assuming the base address is **M**, address of an element  $a_{ij}$ ,  $1 \leq i, j \leq n$ , can be obtained as:

Address ( $A[i][j]$ ) = Number of elements up to the  $a_{ij}$  element

= Total number of elements in the first  $(j - 1)$  columns  
+ number of elements up to the  $i$ th row in the  $j$ th column

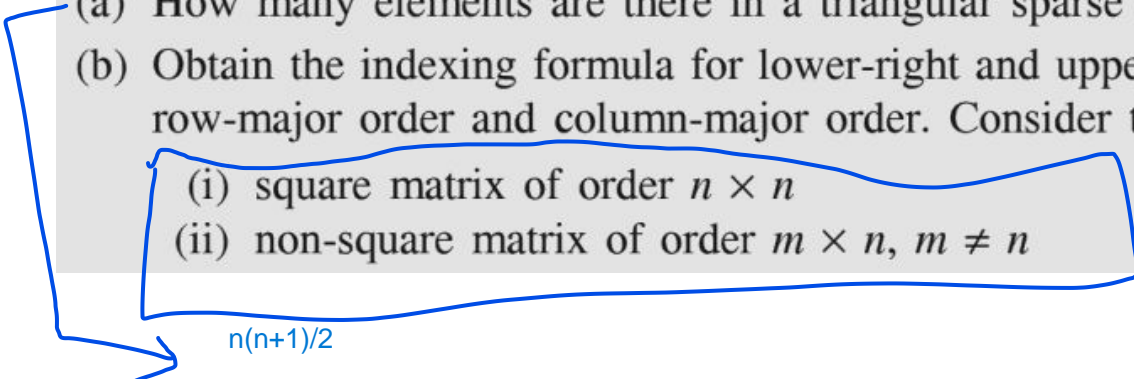
$$= [1 + 2 + 3 + \dots + (j - 1)] + i$$

$$= \frac{j(j-1)}{2} + i$$

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ & a_{22} & a_{23} & \cdots & a_{2n} \\ & & a_{33} & \cdots & a_{3n} \\ & & & \ddots & \\ & & & & a_{nn} \end{bmatrix}_{n \times n}$$



# Exercises

- 
- (a) How many elements are there in a triangular sparse matrix of order  $n \times n$ ?
- (b) Obtain the indexing formula for lower-right and upper-left triangular matrices using row-major order and column-major order. Consider the cases of
- (i) square matrix of order  $n \times n$
  - (ii) non-square matrix of order  $m \times n, m \neq n$

baki

$n(n+1)/2$

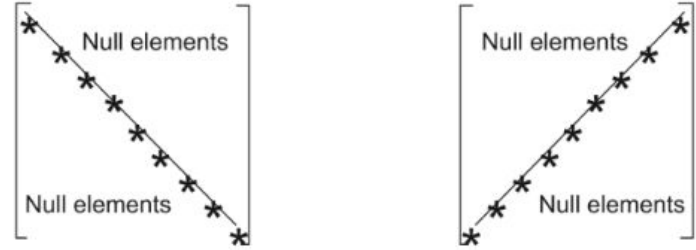
# Sparse Matrices: Band Matrices

Consider an  $n \times n$  matrix  $A=(a_{i,j})$ . If all matrix elements are zero outside a diagonally bordered **band** whose range is determined by constants  $\alpha$  and  $\beta$ , such that

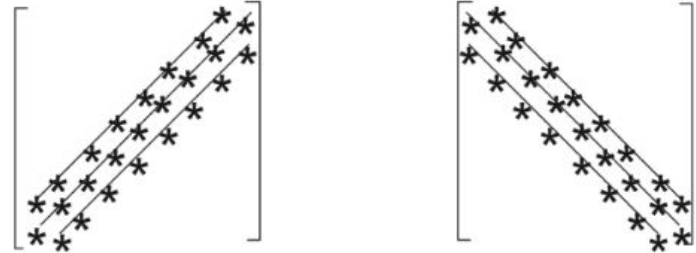
$$a_{i,j} = 0 \quad \text{if } j < i - \beta \text{ or } j > i + \alpha; \quad \alpha, \beta \geq 0.$$

then the quantities  $\alpha$  and  $\beta$  are called *upper bandwidth* and *lower bandwidth* respectively.

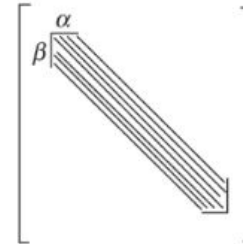
- What if  $\alpha = \beta = 0$  and if  $\alpha = \beta = 1$ ?



Diagonal matrices



Tridiagonal matrices



$\alpha\beta$ -band matrix

# Memory Representation of Tridiagonal Matrix

## Row-major Order

Assuming the base address is **M**, address of an element  $a_{ij}$ ,  $1 \leq i, j \leq n$ , can be obtained as:

Address ( $A[i][j]$ ) = Number of elements up to  $a_{ij}$  element

$$\begin{bmatrix} a_{11} & a_{12} & & & & \\ & a_{21} & a_{22} & a_{23} & & \\ & & a_{32} & a_{33} & a_{34} & \\ & & & a_{43} & a_{44} & a_{45} \\ & & & & \vdots & \\ & & & & & \vdots \\ & & & & & & a_{(n-1)(n-2)} & a_{(n-1)(n-1)} & a_{(n-1)n} \\ & & & & & & & a_{n(n-1)} & a_{nn} \end{bmatrix}$$

= Total number of elements in the first  $(i - 1)$  rows +  
the number of elements up to the  $j^{\text{th}}$  column in the  $i^{\text{th}}$  row

$$= \{2 + [3 + 3 + \dots + \text{up to } (i - 2) \text{ terms}]\} + (j - i + 2)$$

$$= 2 + (i - 2) \times 3 + j - (i - 2)$$

$$= 2 + 2 \times (i - 2) + j$$

$$= M + 2 \times (i - 2) + j + 1$$

# Memory Representation of Tridiagonal Matrix

## Column-major Order

Assuming the base address is **M**, address of an element  $a_{ij}$ ,  $1 \leq i, j \leq n$ , can be obtained as:

Address ( $A[i][j]$ ) = Number of elements up to  $a_{ij}$  element

$$\begin{bmatrix} a_{11} & a_{12} & & & & \\ & a_{22} & a_{23} & & & \\ & & a_{33} & a_{34} & & \\ & & & a_{44} & a_{45} & \\ & & & \vdots & & \\ & & & & \vdots & \\ & & & & & a_{(n-1)(n-2)} & a_{(n-1)(n-1)} & a_{(n-1)n} \\ & & & & & & a_{n(n-1)} & a_{nn} \end{bmatrix}$$

= Total number of elements in the first  $(j - 1)$  columns +  
the number of elements up to the  $i^{\text{th}}$  row in the  $j^{\text{th}}$  column

$$= \{2 + [3 + 3 + \dots + \text{up to } (j - 2) \text{ terms}]\} + (i - j + 2)$$

$$= 2 + (j - 2) \times 3 + i - (j - 2)$$

$$= 2 + 2 \times (j - 2) + i$$

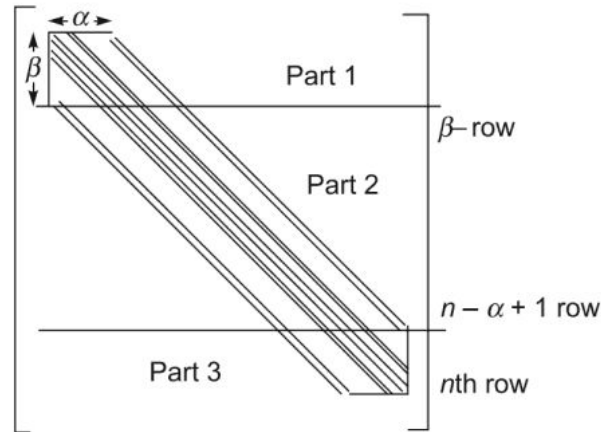
$$= \mathbf{M + 2 \times (j - 2) + i + 1}$$

Note that the formula for row- and column- major order is symmetric and one can be obtained from the other by interchanging  $i$  and  $j$ .

# Exercise

What will be the indexing formula for the  $\alpha\beta$ -band matrix using both the schemes?

Hint: there will be  $(\alpha - 1)$  and  $(\beta - 1)$  sub-diagonals above and below the main diagonal respectively. As there will be three possible arrangements so the elements in each arrangement can be referred by three indexing formulae corresponding to the three parts shown below.



## Next Lecture

- Multidimensional Arrays Contd...