

Matrix Data Structures

Shusen Wang

Dense Matrix Data Structures

Dense Matrix Data Structure

- **Dense matrix:** most of the elements are non-zero.
- Dense matrix can be stored in a fixed-size array.

Array:

[illegible]

Dense Matrix Data Structure

Row-Major Order

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \\ a_{41} & a_{42} & a_{43} \end{bmatrix}$$

Column-Major Order

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \\ a_{41} & a_{42} & a_{43} \end{bmatrix}$$

Array:

[illegible]

Dense Matrix Data Structure

Row-Major Order

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \\ a_{41} & a_{42} & a_{43} \end{bmatrix}$$

Array:

a_{11}	a_{12}	a_{13}									
----------	----------	----------	--	--	--	--	--	--	--	--	--

Column-Major Order

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \\ a_{41} & a_{42} & a_{43} \end{bmatrix}$$

Dense Matrix Data Structure

Row-Major Order

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \\ a_{41} & a_{42} & a_{43} \end{bmatrix}$$

Column-Major Order

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \\ a_{41} & a_{42} & a_{43} \end{bmatrix}$$

Array:

a_{11}	a_{12}	a_{13}	a_{21}	a_{22}	a_{23}						
----------	----------	----------	----------	----------	----------	--	--	--	--	--	--

Dense Matrix Data Structure

Row-Major Order

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \\ a_{41} & a_{42} & a_{43} \end{bmatrix}$$

Column-Major Order

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \\ a_{41} & a_{42} & a_{43} \end{bmatrix}$$

Array:

a_{11}	a_{12}	a_{13}	a_{21}	a_{22}	a_{23}	a_{31}	a_{32}	a_{33}			
----------	----------	----------	----------	----------	----------	----------	----------	----------	--	--	--

Dense Matrix Data Structure

Row-Major Order

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \\ a_{41} & a_{42} & a_{43} \end{bmatrix}$$

Column-Major Order

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \\ a_{41} & a_{42} & a_{43} \end{bmatrix}$$

Array:

a_{11}	a_{12}	a_{13}	a_{21}	a_{22}	a_{23}	a_{31}	a_{32}	a_{33}	a_{41}	a_{42}	a_{43}
----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------

Dense Matrix Data Structure

Row-Major Order

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \\ a_{41} & a_{42} & a_{43} \end{bmatrix}$$

Column-Major Order

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \\ a_{41} & a_{42} & a_{43} \end{bmatrix}$$

Array:

a_{11}	a_{21}	a_{31}	a_{41}								
----------	----------	----------	----------	--	--	--	--	--	--	--	--

Dense Matrix Data Structure

Row-Major Order

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \\ a_{41} & a_{42} & a_{43} \end{bmatrix}$$

Column-Major Order

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \\ a_{41} & a_{42} & a_{43} \end{bmatrix}$$

Array:

a_{11}	a_{21}	a_{31}	a_{41}	a_{12}	a_{22}	a_{32}	a_{42}				
----------	----------	----------	----------	----------	----------	----------	----------	--	--	--	--

Dense Matrix Data Structure

Row-Major Order

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \\ a_{41} & a_{42} & a_{43} \end{bmatrix}$$

Column-Major Order

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \\ a_{41} & a_{42} & a_{43} \end{bmatrix}$$

Array:

a_{11}	a_{21}	a_{31}	a_{41}	a_{12}	a_{22}	a_{32}	a_{42}	a_{13}	a_{23}	a_{33}	a_{43}
----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------

Why does the layout matter?

Row-Major Order

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \\ a_{41} & a_{42} & a_{43} \end{bmatrix}$$

Column-Major Order

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \\ a_{41} & a_{42} & a_{43} \end{bmatrix}$$

Array:

a_{11}	a_{12}	a_{13}	a_{21}	a_{22}	a_{23}	a_{31}	a_{32}	a_{33}	a_{41}	a_{42}	a_{43}
----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------

Why does the layout matter?

Row-Major Order

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \\ a_{41} & a_{42} & a_{43} \end{bmatrix}$$

Column-Major Order

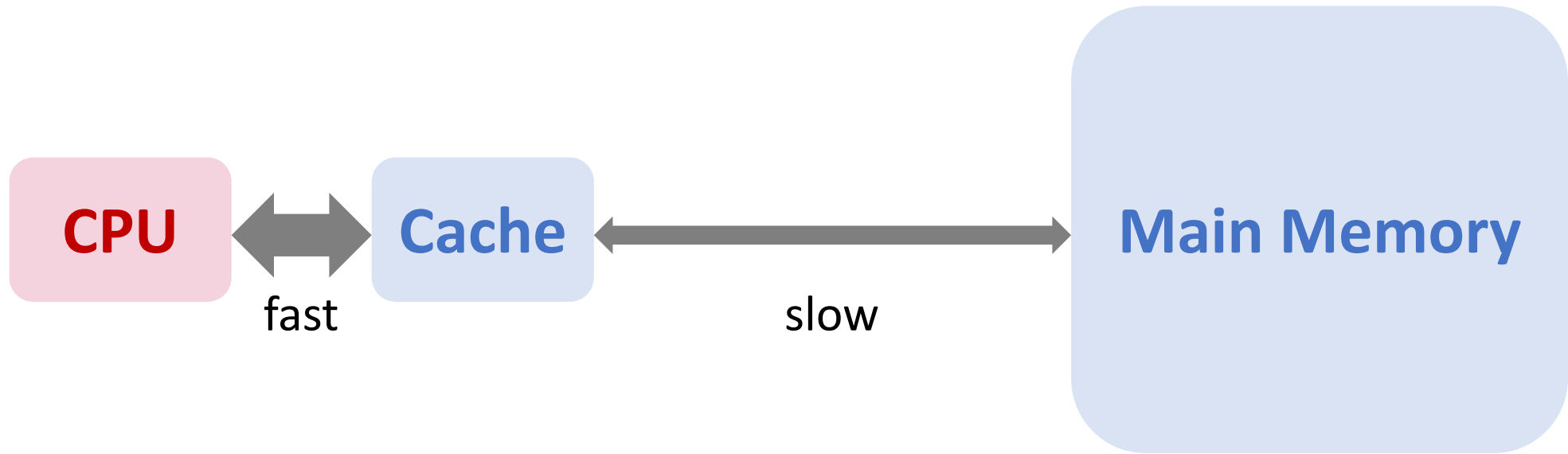
$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \\ a_{41} & a_{42} & a_{43} \end{bmatrix}$$

Getting a row is fast.

Array:

a_{11}	a_{12}	a_{13}	a_{21}	a_{22}	a_{23}	a_{31}	a_{32}	a_{33}	a_{41}	a_{42}	a_{43}
----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------

Why does the layout matter?



Array:

a_{11}	a_{12}	a_{13}	a_{21}	a_{22}	a_{23}	a_{31}	a_{32}	a_{33}	a_{41}	a_{42}	a_{43}
----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------

Why does the layout matter?

Row-Major Order

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \\ a_{41} & a_{42} & a_{43} \end{bmatrix}$$

Column-Major Order

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \\ a_{41} & a_{42} & a_{43} \end{bmatrix}$$

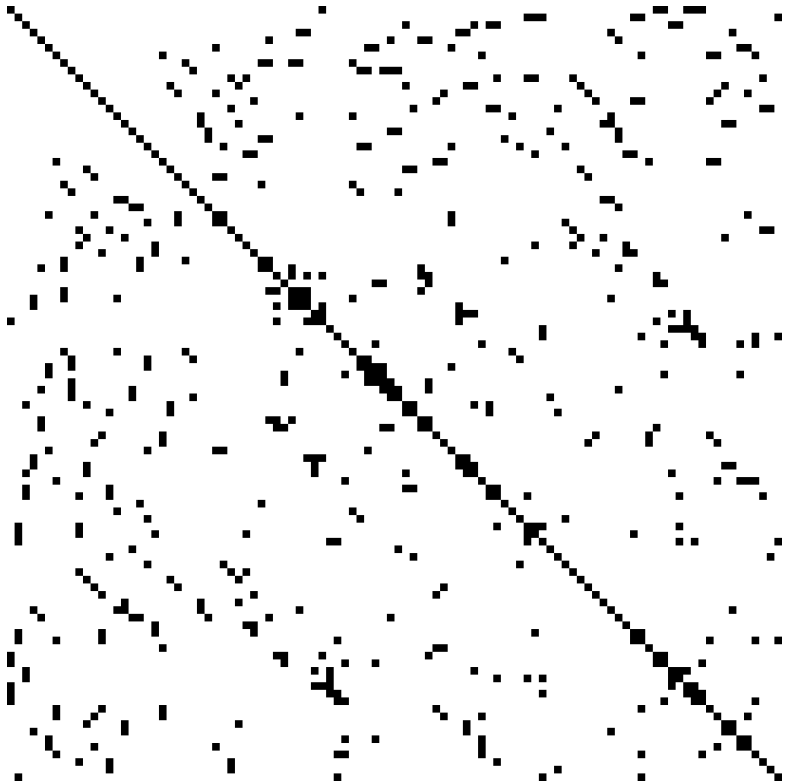
Getting a column is slow.

Array:

a_{11}	a_{12}	a_{13}	a_{21}	a_{22}	a_{23}	a_{31}	a_{32}	a_{33}	a_{41}	a_{42}	a_{43}
----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------

Sparse Matrix Data Structures

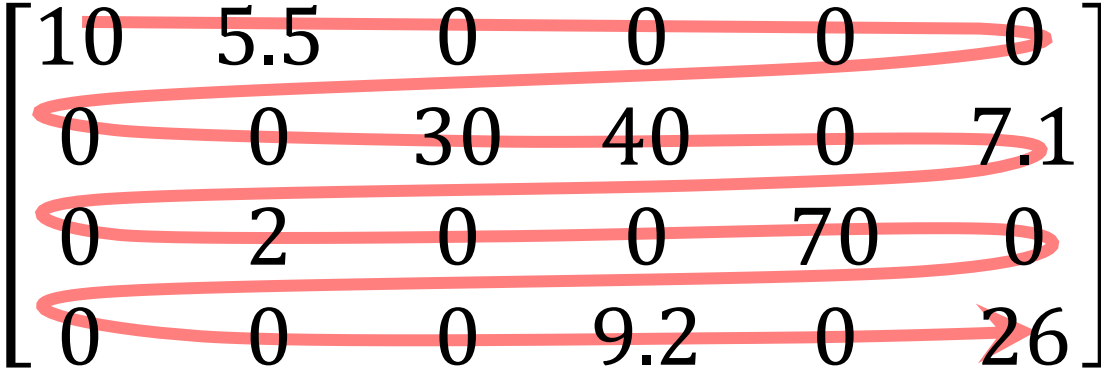
Sparse Matrix



Example of sparse matrix

- **Sparse matrix:** A matrix in which most elements are zeros.
- **Question:** How to store a sparse matrix?
- **Bad solution:** As a dense matrix.
- **Good solution:** Storing only the nonzero elements and their indices.

Compressed Sparse Row (CSR)

$$\mathbf{A} = \begin{bmatrix} 10 & 5.5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 30 & 40 & 0 & 7.1 \\ 0 & 2 & 0 & 0 & 70 & 0 \\ 0 & 0 & 0 & 9.2 & 0 & 26 \end{bmatrix}$$
The matrix A is a 4x6 matrix. Red ovals are drawn around the non-zero elements in each row. Row 1: 10, 5.5, 0, 0, 0, 0. Row 2: 0, 0, 30, 40, 0, 7.1. Row 3: 0, 2, 0, 0, 70, 0. Row 4: 0, 0, 0, 9.2, 0, 26. The ovals are horizontal and slightly curved, connecting the non-zero values in each row.

Compressed Sparse Row (CSR)

$$A = \begin{bmatrix} 10 & 5.5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 30 & 40 & 0 & 7.1 \\ 0 & 2 & 0 & 0 & 70 & 0 \\ 0 & 0 & 0 & 9.2 & 0 & 26 \end{bmatrix}$$

Value:

10	5.5	30	40	7.1	2	70	9.2	26
----	-----	----	----	-----	---	----	-----	----

Compressed Sparse Row (CSR)

$$\mathbf{A} = \begin{bmatrix} 10 & 5.5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 30 & 40 & 0 & 7.1 \\ 0 & 2 & 0 & 0 & 70 & 0 \\ 0 & 0 & 0 & 9.2 & 0 & 26 \end{bmatrix}$$

Value:

10	5.5	30	40	7.1	2	70	9.2	26
----	-----	----	----	-----	---	----	-----	----

$\text{nnz}(\mathbf{A})$

Compressed Sparse Row (CSR)

$$\mathbf{A} = \begin{bmatrix} 10 & 5.5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 30 & 40 & 0 & 7.1 \\ 0 & 2 & 0 & 0 & 70 & 0 \\ 0 & 0 & 0 & 9.2 & 0 & 26 \end{bmatrix}$$

Value:

10	5.5	30	40	7.1	2	70	9.2	26
----	-----	----	----	-----	---	----	-----	----

Row Index:

1	1	2	2	2	3	3	4	4
---	---	---	---	---	---	---	---	---

Col Index:

1	2	3	4	6	2	5	4	6
---	---	---	---	---	---	---	---	---

Compressed Sparse Row (CSR)

$$\mathbf{A} = \begin{bmatrix} 10 & 5.5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 30 & 40 & 0 & 7.1 \\ 0 & 2 & 0 & 0 & 70 & 0 \\ 0 & 0 & 0 & 9.2 & 0 & 26 \end{bmatrix}$$

Value:

10	5.5	30	40	7.1	2	70	9.2	26
----	-----	----	----	-----	---	----	-----	----

Row Index:

1	1	2	2	2	3	3	4	4
---	---	---	---	---	---	---	---	---

Col Index:

1	2	3	4	6	2	5	4	6
---	---	---	---	---	---	---	---	---

How to slice a row?

CSR Matrix: $A = \begin{bmatrix} 10 & 5.5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 30 & 40 & 0 & 7.1 \\ 0 & 2 & 0 & 0 & 70 & 0 \\ 0 & 0 & 0 & 9.2 & 0 & 26 \end{bmatrix}$

Value:

10	5.5	30	40	7.1	2	70	9.2	26
----	-----	----	----	-----	---	----	-----	----

Row Index:

1	1	2	2	2	3	3	4	4
---	---	---	---	---	---	---	---	---

Col Index:

1	2	3	4	6	2	5	4	6
---	---	---	---	---	---	---	---	---

How to slice a row?

CSR Matrix: $A = \begin{bmatrix} 10 & 5.5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 30 & 40 & 0 & 7.1 \\ 0 & 2 & 0 & 0 & 70 & 0 \\ 0 & 0 & 0 & 9.2 & 0 & 26 \end{bmatrix}$

Value:

10	5.5	30	40	7.1	2	70	9.2	26
----	-----	----	----	-----	---	----	-----	----

Row Index:

1	1	2	2	2	3	3	4	4
---	---	---	---	---	---	---	---	---

Col Index:

1	2	3	4	6	2	5	4	6
---	---	---	---	---	---	---	---	---

How to slice a row?

CSR Matrix: $A = \begin{bmatrix} 10 & 5.5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 30 & 40 & 0 & 7.1 \\ 0 & 2 & 0 & 0 & 70 & 0 \\ 0 & 0 & 0 & 9.2 & 0 & 26 \end{bmatrix}$

Getting a row is fast.

Value:

10	5.5	30	40	7.1	2	70	9.2	26
----	-----	----	----	-----	---	----	-----	----

Row Index:

1	1	2	2	2	3	3	4	4
---	---	---	---	---	---	---	---	---

Col Index:

1	2	3	4	6	2	5	4	6
---	---	---	---	---	---	---	---	---

How to slice a column?

CSR Matrix: $A = \begin{bmatrix} 10 & 5.5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 30 & 40 & 0 & 7.1 \\ 0 & 2 & 0 & 0 & 70 & 0 \\ 0 & 0 & 0 & 9.2 & 0 & 26 \end{bmatrix}$

Value:

10	5.5	30	40	7.1	2	70	9.2	26
----	-----	----	----	-----	---	----	-----	----

Row Index:

1	1	2	2	2	3	3	4	4
---	---	---	---	---	---	---	---	---

Col Index:

1	2	3	4	6	2	5	4	6
---	---	---	---	---	---	---	---	---

How to slice a column?

CSR Matrix: $A = \begin{bmatrix} 10 & 5.5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 30 & 40 & 0 & 7.1 \\ 0 & 2 & 0 & 0 & 70 & 0 \\ 0 & 0 & 0 & 9.2 & 0 & 26 \end{bmatrix}$

Value:

10	5.5	30	40	7.1	2	70	9.2	26
----	-----	----	----	-----	---	----	-----	----

Row Index:

1	1	2	2	2	3	3	4	4
---	---	---	---	---	---	---	---	---

Col Index:

1	2	3	4	6	2	5	4	6
---	---	---	---	---	---	---	---	---



How to slice a column?

CSR Matrix: $A = \begin{bmatrix} 10 & 5.5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 30 & 40 & 0 & 7.1 \\ 0 & 2 & 0 & 0 & 70 & 0 \\ 0 & 0 & 0 & 9.2 & 0 & 26 \end{bmatrix}$

Getting a column is slow.

Value:

10	5.5	30	40	7.1	2	70	9.2	26
----	-----	----	----	-----	---	----	-----	----

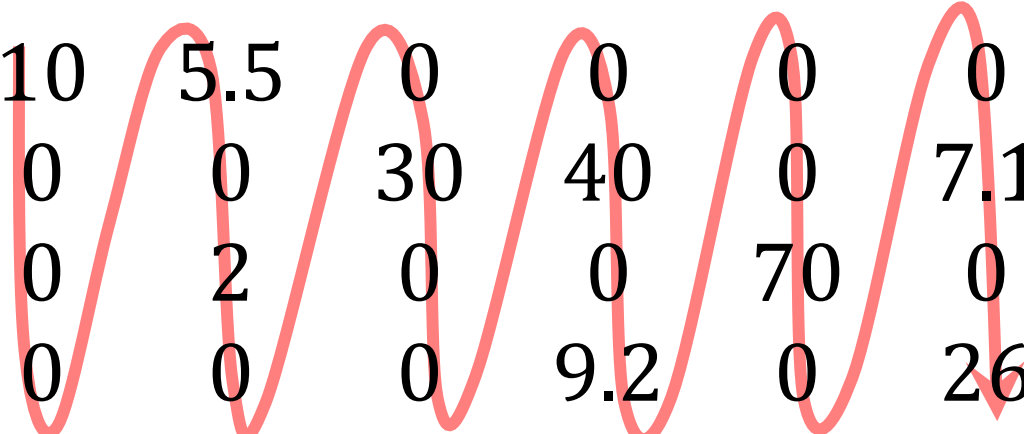
Row Index:

1	1	2	2	2	3	3	4	4
---	---	---	---	---	---	---	---	---

Col Index:

1	2	3	4	6	2	5	4	6
---	---	---	---	---	---	---	---	---

Compressed Sparse Column (CSC)

$$\mathbf{A} = \begin{bmatrix} 10 & 5.5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 30 & 40 & 0 & 7.1 \\ 0 & 2 & 0 & 0 & 70 & 0 \\ 0 & 0 & 0 & 9.2 & 0 & 26 \end{bmatrix}$$
A red wavy line is drawn across the matrix, highlighting the non-zero elements. It starts at the first element (10), goes down to the second (0), up to the third (0), down to the fourth (0), up to the fifth (0), down to the sixth (0), up to the seventh (0), down to the eighth (0), up to the ninth (0), down to the tenth (0), up to the eleventh (0), down to the twelfth (0), up to the thirteenth (0), down to the fourteenth (0), up to the fifteenth (0), down to the sixteenth (0), up to the seventeenth (0), down to the eighteenth (0), up to the nineteenth (0), down to the twentieth (0), up to the twenty-first (0), down to the twenty-second (0), up to the twenty-third (0), down to the twenty-fourth (0), up to the twenty-fifth (0), down to the twenty-sixth (0), up to the twenty-seventh (0), down to the twenty-eighth (0), up to the twenty-ninth (0), down to the thirtieth (0), up to the thirty-first (0), down to the thirty-second (0), up to the thirty-third (0), down to the thirty-fourth (0), up to the thirty-fifth (0), down to the thirty-sixth (0), up to the thirty-seventh (0), down to the thirty-eighth (0), up to the thirty-ninth (0), down to the fortieth (0), up to the forty-first (0), down to the forty-second (0), up to the forty-third (0), down to the forty-fourth (0), up to the forty-fifth (0), down to the forty-sixth (0), up to the forty-seventh (0), down to the forty-eighth (0), up to the forty-ninth (0), down to the fiftieth (0), up to the fifty-first (0), down to the fifty-second (0), up to the fifty-third (0), down to the fifty-fourth (0), up to the fifty-fifth (0), down to the fifty-sixth (0), up to the fifty-seventh (0), down to the fifty-eighth (0), up to the fifty-ninth (0), down to the sixtieth (0), up to the sixty-first (0), down to the sixty-second (0), up to the sixty-third (0), down to the sixty-fourth (0), up to the sixty-fifth (0), down to the sixty-sixth (0), up to the sixty-seventh (0), down to the sixty-eighth (0), up to the sixty-ninth (0), down to the seventieth (0), up to the seventy-first (0), down to the seventy-second (0), up to the seventy-third (0), down to the seventy-fourth (0), up to the seventy-fifth (0), down to the seventy-sixth (0), up to the seventy-seventh (0), down to the seventy-eighth (0), up to the seventy-ninth (0), down to the eightieth (0), up to the eighty-first (0), down to the eighty-second (0), up to the eighty-third (0), down to the eighty-fourth (0), up to the eighty-fifth (0), down to the eighty-sixth (0), up to the eighty-seventh (0), down to the eighty-eighth (0), up to the eighty-ninth (0), down to the ninetieth (0), up to the ninety-first (0), down to the ninety-second (0), up to the ninety-third (0), down to the ninety-fourth (0), up to the ninety-fifth (0), down to the ninety-sixth (0), up to the ninety-seventh (0), down to the ninety-eighth (0), up to the ninety-ninth (0), down to the one hundredth (0).

Compressed Sparse Column (CSC)

$$A = \begin{bmatrix} 10 & 5.5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 30 & 40 & 0 & 7.1 \\ 0 & 2 & 0 & 0 & 70 & 0 \\ 0 & 0 & 0 & 9.2 & 0 & 26 \end{bmatrix}$$

Value:

10	5.5	2	30	40	9.2	70	7.1	26
----	-----	---	----	----	-----	----	-----	----

Compressed Sparse Column (CSC)

$$\mathbf{A} = \begin{bmatrix} 10 & 5.5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 30 & 40 & 0 & 7.1 \\ 0 & 2 & 0 & 0 & 70 & 0 \\ 0 & 0 & 0 & 9.2 & 0 & 26 \end{bmatrix}$$

Value:

10	5.5	2	30	40	9.2	70	7.1	26
----	-----	---	----	----	-----	----	-----	----

Row Index:

1	1	3	2	2	4	3	2	4
---	---	---	---	---	---	---	---	---

Col Index:

1	2	2	3	4	4	5	6	6
---	---	---	---	---	---	---	---	---

Memory Cost

- 8 Bytes for a double-precision floating-point number.
- 4 Bytes for a long integer.
- Memory cost (Bytes) of CSR or CSC:

$$(8 + 4 + 4) \cdot \text{nnz}(\mathbf{A}) = 16 \cdot \text{nnz}(\mathbf{A}).$$

- Memory cost (Bytes) of an $m \times n$ dense matrix:

$$8mn.$$

- If over 50% elements are zeros, then CSR and CSC save memory.

Questions

From CSR to dense matrix

Value:

9	8.2	29	2	3.1	5	2	1.5	7	10
---	-----	----	---	-----	---	---	-----	---	----

Row Index:

1	1	1	1	2	2	3	4	4	4
---	---	---	---	---	---	---	---	---	---

Col Index:

2	4	5	6	1	2	2	3	4	6
---	---	---	---	---	---	---	---	---	---

Convert the CSR matrix to dense matrix:

$$\mathbf{A} = \begin{bmatrix} ? & ? & ? & ? & ? & ? \\ ? & ? & ? & ? & ? & ? \\ ? & ? & ? & ? & ? & ? \\ ? & ? & ? & ? & ? & ? \end{bmatrix}$$

Matrix L1 Norm

Value:	3	2	1	7	4	3	5	1	2
Row Index:	1	1	2	2	2	3	3	4	4
Col Index:	1	2	3	4	6	2	5	4	6

- The 4×6 matrix \mathbf{A} is stored as CSR matrix (in the above).
- **Question:** What is the ℓ_1 -norm of \mathbf{A} ?
- **Hint:** The matrix ℓ_1 -norm is $\|\mathbf{A}\|_1 = \sum_{i=1}^4 \sum_{j=1}^6 |a_{ij}|$.

Thank You!