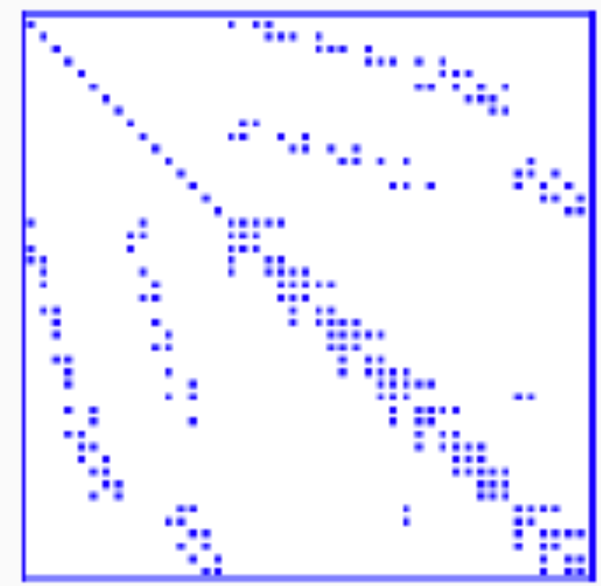
# Sparse Systems

# Sparse Matrices



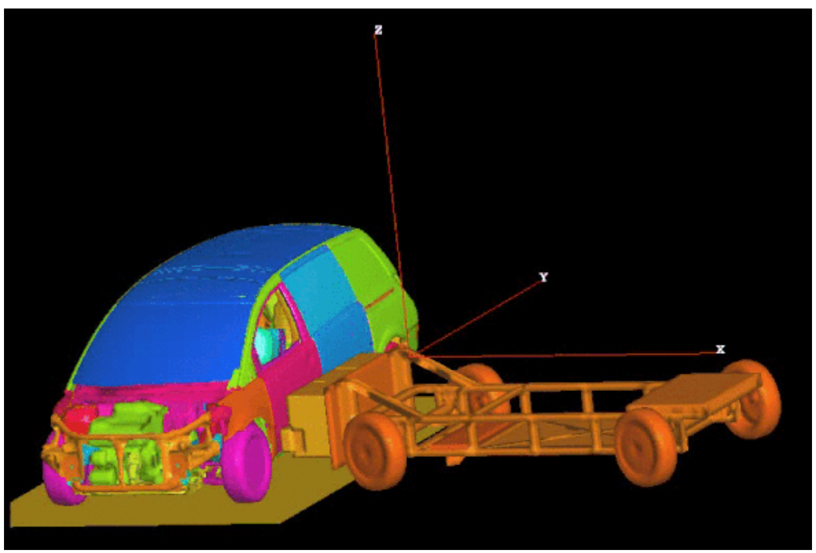
Some type of matrices contain many zeros.
Storing all those zero entries is wasteful!

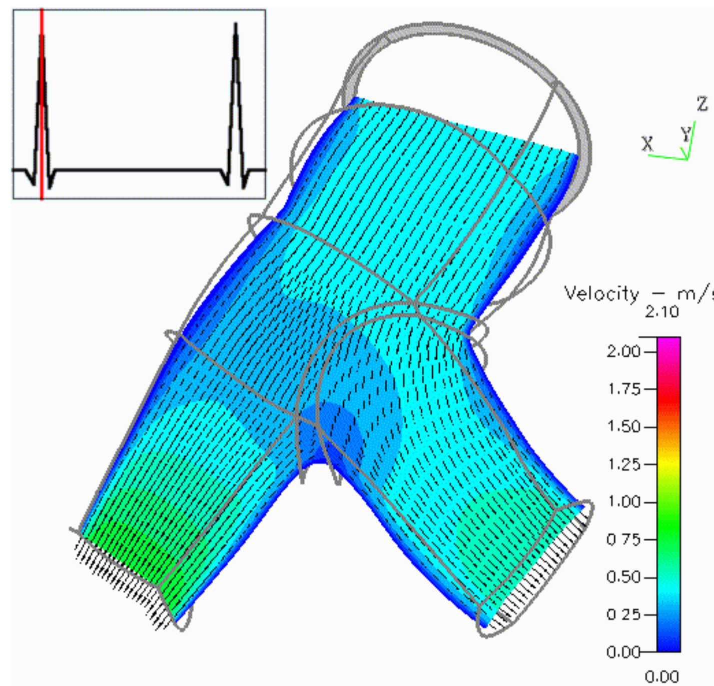
How can we efficiently store large matrices without storing tons of zeros?

- **Sparse matrices** (vague definition): matrix with few non-zero entries.
- For practical purposes: an $m \times n$ matrix is sparse if it has $O(\min(m, n))$ non-zero entries.
- This means roughly a constant number of non-zero entries per row and column.
- Another definition: "matrices that allow special techniques to take advantage of the large number of zero elements" (J. Wilkinson)

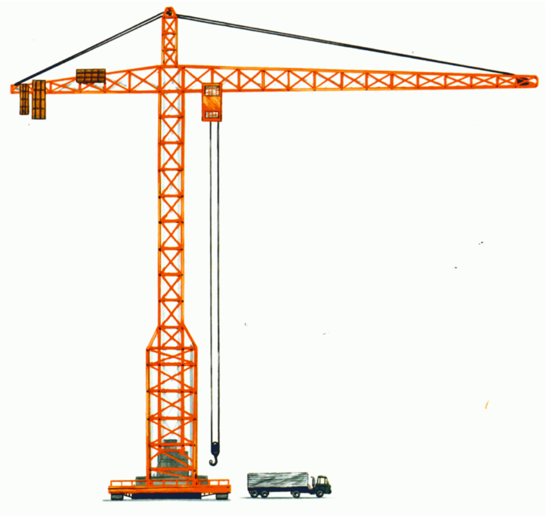
# Sparse Matrices: Goals

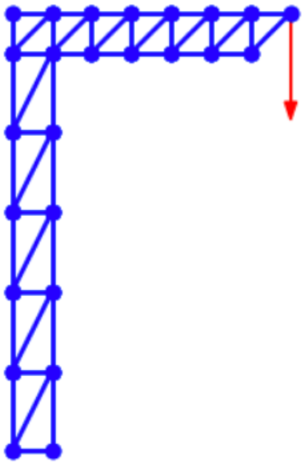
- **Perform standard matrix computations economically, i.e., without storing the zeros of the matrix.**
- For typical Finite Element and Finite Difference matrices, the number of non-zero entries is $O(n)$

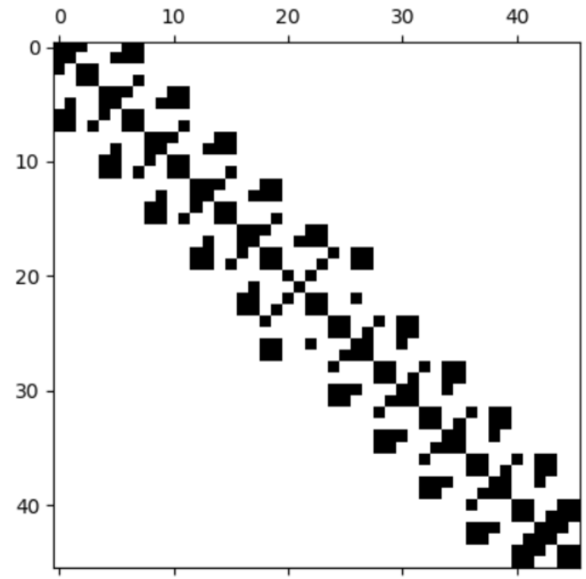


# Sparse Matrices: MP example







# Sparse Matrices

**EXAMPLE:**

Number of operations required to add two square dense matrices:

$$O(n^2)$$

Number of operations required to add two sparse matrices $\mathbf{A}$ and $\mathbf{B}$:

$$O(\mathrm{nnz}(\mathbf{A}) + \mathrm{nnz}(\mathbf{B}))$$

where $\mathrm{nnz}(\mathbf{X}) =$ number of non-zero elements of a matrix $\mathbf{X}$

# Popular Storage Structures

| | | | |
|---|---|---|---|
| **DNS** | Dense | **ELL** | Ellpack-Itpack |
| **BND** | Linpack Banded | **DIA** | Diagonal |
| **COO** | Coordinate | **BSR** | Block Sparse Row |
| **CSR** | Compressed Sparse Row | **SSK** | Symmetric Skyline |
| **CSC** | Compressed Sparse Column | **BSR** | Nonsymmetric Skyline |
| **MSR** | Modified CSR | **JAD** | Jagged Diagonal |
| **LIL** | Linked List | | |

note: CSR = CRS, CCS = CSC, SSK = SKS in some references

We will focus on COO and CSR!

# Dense (DNS)

$$A = \begin{bmatrix} 0. & 1.9 & 0. & -5.2 \\ 0.3 & 0. & 9.1 & 0. \\ 4.4 & 5.8 & 3.6 & 0. \\ 0. & 0. & 7.2 & 2.7 \end{bmatrix}$$

$$Ashape = (nrow, ncol)$$

$$A_{dense} = [\underbrace{0. \quad 1.9 \quad 0. \quad -5.2}_{\text{Row 0}} \quad \underbrace{0.3 \quad 0. \quad 9.1 \quad 0.}_{\text{Row 1}} \quad \underbrace{4.4 \quad 5.8 \quad 3.6 \quad 0.}_{\text{Row 2}} \quad \underbrace{0. \quad 0. \quad 7.2 \quad 2.7}_{\text{Row 3}}]$$

- Simple
- Row-wise
- Easy blocked formats
- Stores all the zeros

# Coordinate (COO)

$$A = \begin{bmatrix} 0. & 1.9 & 0. & -5.2 \\ 0.3 & 0. & 9.1 & 0. \\ 4.4 & 5.8 & 3.6 & 0. \\ 0. & 0. & 7.2 & 2.7 \end{bmatrix}$$

$$data = \begin{bmatrix} 1.9 & -5.2 & 0.3 & 9.1 & 4.4 & 5.8 & 3.6 & 7.2 & 2.7 \end{bmatrix}$$

$$row = \begin{bmatrix} 0 & 0 & 1 & 1 & 2 & 2 & 2 & 3 & 3 \end{bmatrix}$$

$$col = \begin{bmatrix} 1 & 3 & 0 & 2 & 0 & 1 & 2 & 2 & 3 \end{bmatrix}$$

- Simple
- Does not store the zero elements
- Not sorted
- *row* and *col*: array of integers
- *data*: array of doubles

# Iclicker question

$$A = \begin{bmatrix} 1 & 0 & 0 & 2 & 0 \\ 3 & 4 & 0 & 5 & 0 \\ 6 & 0 & 7 & 8 & 9 \\ 0 & 0 & 10 & 11 & 0 \\ 0 & 0 & 0 & 0 & 12 \end{bmatrix}$$

$$\begin{array}{lcl} data & = & [\ 12.0\ \ 9.0\ \ 7.0\ \ 5.0\ \ 1.0\ \ 2.0\ \ 11.0\ \ 3.0\ \ 6.0\ \ 4.0\ \ 8.0\ \ 10.0\ ] \\ row & = & [\ 4\ \ 2\ \ 2\ \ 1\ \ 0\ \ 0\ \ 3\ \ 1\ \ 2\ \ 1\ \ 2\ \ 3\ ] \\ col & = & [\ 4\ \ 4\ \ 2\ \ 3\ \ 0\ \ 3\ \ 3\ \ 0\ \ 0\ \ 1\ \ 3\ \ 2\ ] \end{array}$$

How many integers are stored in COO format
($\boldsymbol{A}$ has dimensions $n{\times}n$)?

A) $nnz$
B) $n$
C) $2\,nnz$
D) $n^2$
E) $2\,n$

# Iclicker question

## Representing a Sparse Matrix in Coordinate (COO) Form

1 point

Consider the following matrix:

$$A = \begin{bmatrix} 0 & 0 & 1.3 \\ -1.5 & 0.2 & 0 \\ 5 & 0 & 0 \\ 0 & 0.3 & 3 \\ 0 & 0 & 0 \end{bmatrix}$$

Suppose we store one row index (a 32-bit integer), one column index (a 32-bit integer), and one data value (a 64-bit float) for each non-zero entry in $A$. How many bytes in total are stored? Please note that 1 byte is equal to 8 bits.

A) 56 bytes
B) 72 bytes
C) 96 bytes
D) 120 bytes
E) 144 bytes

# Compressed Sparse Row (CSR)

$$A = \begin{bmatrix} 1 & 0 & 0 & 2 & 0 \\ 3 & 4 & 0 & 5 & 0 \\ 6 & 0 & 7 & 8 & 9 \\ 0 & 0 & 10 & 11 & 0 \\ 0 & 0 & 0 & 0 & 12 \end{bmatrix}$$

$$data = [\ \underbrace{\overset{\mathbf{0}}{1.0}\ \ \overset{1}{2.0}}_{\text{Row 0}}\ \ \underbrace{\overset{\mathbf{2}}{3.0}\ \ \overset{3}{4.0}\ \ \overset{4}{5.0}}_{\text{Row 1}}\ \ \underbrace{\overset{\mathbf{5}}{6.0}\ \ \overset{6}{7.0}\ \ \overset{7}{8.0}\ \ \overset{8}{9.0}}_{\text{Row 2}}\ \ \underbrace{\overset{\mathbf{9}}{10.0}\ \ \overset{10}{11.0}}_{\text{Row 3}}\ \ \underbrace{\overset{\mathbf{11}}{12.0}}_{\text{Row 4}}\ ]$$

$$col = [\ 0\ \ 3\ \ 0\ \ 1\ \ 3\ \ 0\ \ 2\ \ 3\ \ 4\ \ 2\ \ 3\ \ 4\ ]$$

$$rowptr = [\ \underbrace{0\ \ 2\ \ 5\ \ 9\ \ 11}_{\text{Row offsets}}\ \ \underbrace{12}_{nnz(\boldsymbol{A})}\ ]$$

# Compressed Sparse Row (CSR)

$$A = \begin{bmatrix} 1 & 0 & 0 & 2 & 0 \\ 3 & 4 & 0 & 5 & 0 \\ 6 & 0 & 7 & 8 & 9 \\ 0 & 0 & 10 & 11 & 0 \\ 0 & 0 & 0 & 0 & 12 \end{bmatrix}$$

$$\begin{array}{lcl} data & = & [\ 1.0 \quad 2.0 \quad 3.0 \quad 4.0 \quad 5.0 \quad 6.0 \quad 7.0 \quad 8.0 \quad 9.0 \quad 10.0 \quad 11.0 \quad 12.0\ ] \\ col & = & [\ 0 \quad 3 \quad 0 \quad 1 \quad 3 \quad 0 \quad 2 \quad 3 \quad 4 \quad 2 \quad 3 \quad 4\ ] \\ rowptr & = & [\ 0 \quad 2 \quad 5 \quad 9 \quad 11 \quad 12\ ] \end{array}$$

- Does not store the zero elements
- Fast arithmetic operations between sparse matrices, and fast matrix-vector product
- *col*: contain the column indices (array of $nnz$ integers)
- *data*: contain the non-zero elements (array of $nnz$ doubles)
- *rowptr*: contain the row offset (array of $n + 1$ integers)

# Example - CSR format

$$A = \begin{bmatrix} 0. & 1.9 & 0. & -5.2 \\ 0. & 0. & 0. & 0. \\ 4.4 & 5.8 & 3.6 & 0. \\ 0. & 0. & 7.2 & 2.7 \end{bmatrix}$$

$$data = \begin{bmatrix} 1.9 & -5.2 & 4.4 & 5.8 & 3.6 & 7.2 & 2.7 \end{bmatrix}$$

$$rowptr = \begin{bmatrix} 0 & 2 & 2 & 5 & 7 \end{bmatrix}$$

$$col = \begin{bmatrix} 1 & 3 & 0 & 1 & 2 & 2 & 3 \end{bmatrix}$$