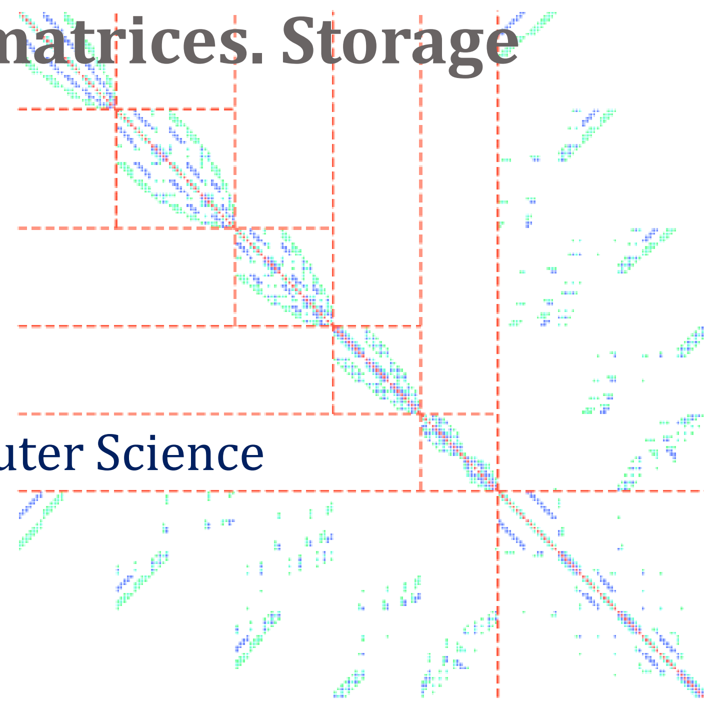
Numerical Methods of Linear Algebra for Sparse Matrices

Graph representations of sparse matrices. Storage schemes for sparse matrices



Anna Nasedkina

Department of Mathematical Modeling

Institute of Mathematics, Mechanics and Computer Science

Southern Federal University

Structures and graph representations of sparse matrices

Types of sparse matrices

Graph representations

Permutations and reordering

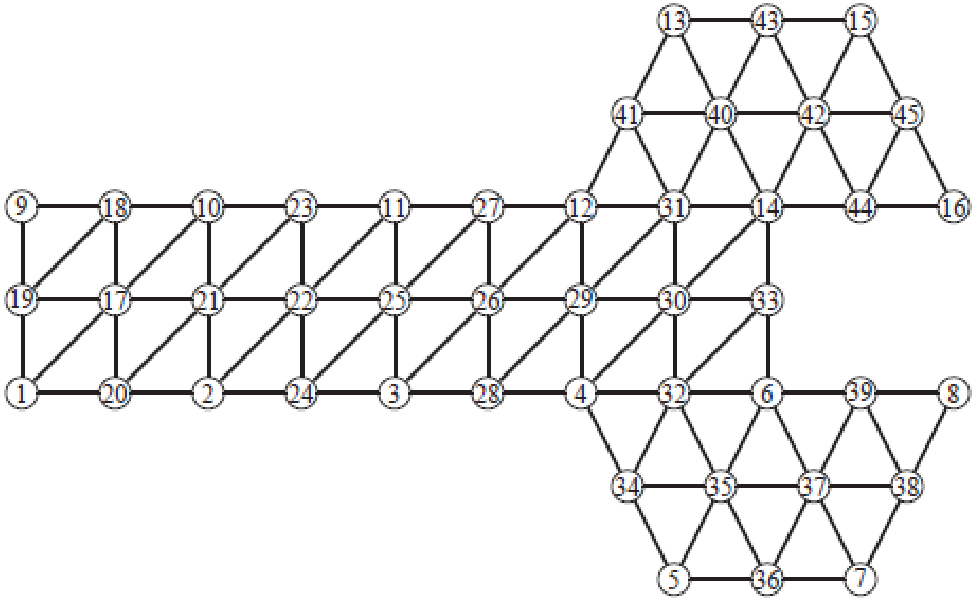
Definition of a sparse matrix

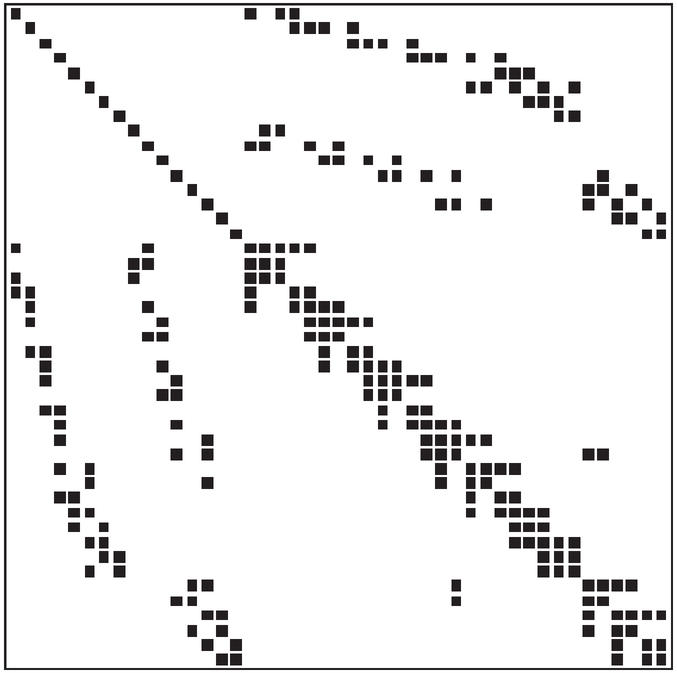
- A sparse matrix is a matrix which has very few nonzero elements.
- Example of sparse matrix: 64 elements, 52 zero elements and 12 nonzero elements (18%)

$$\begin{pmatrix} 1.0 & 0 & 5.0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3.0 & 0 & 0 & 0 & 0 & 11.0 & 0 \\ 0 & 0 & 0 & 0 & 9.0 & 0 & 0 & 0 \\ 0 & 0 & 6.0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 7.0 & 0 & 0 & 0 & 0 \\ 2.0 & 0 & 0 & 0 & 0 & 10.0 & 0 & 0 \\ 0 & 0 & 0 & 8.0 & 0 & 0 & 0 & 0 \\ 0 & 4.0 & 0 & 0 & 0 & 0 & 0 & 12.0 \end{pmatrix}$$

Types of sparse matrices

- *Structured*: nonzero entries form a regular pattern
- *Unstructured*: nonzero entries are located irregularly
- Example: final element grid and corresponding sparse matrix





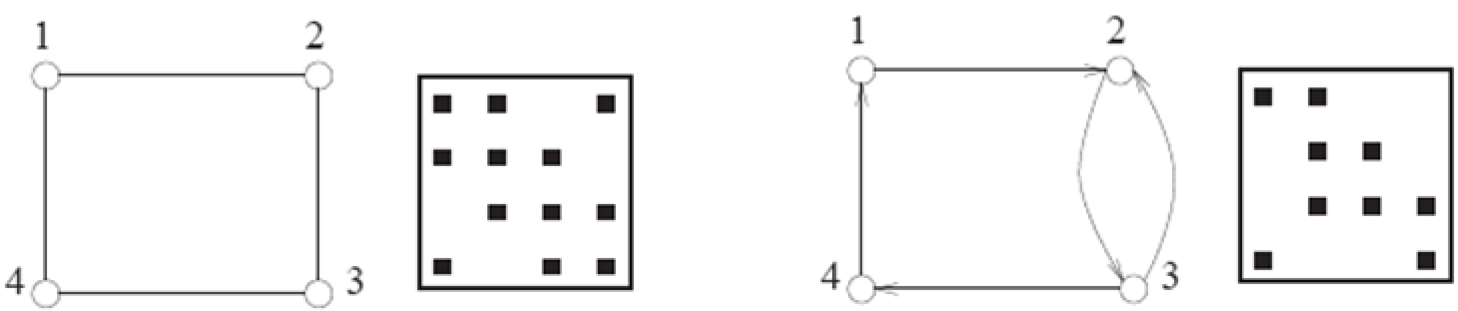
Graph representations of sparse matrices

- Vertices $V = \{v_1, v_2, \ldots, v_n\}$
- Edges $(v_i, v_j) \in E,\ E \subseteq V \times V$
- Adjacency graph $G = (V,E)$ for matrix $A \in C^{m \times n}$

  V – set of n unknowns, E – set of binary relations

  $a_{ij} \neq 0 \Leftrightarrow \exists (i, j):$ equation i includes unknown j
- Pattern of sparse matrix $P_A = \{(i, j) | a_{ij} \neq 0\}$

Patterns for undirected and directed graphs



Permutations and reordering

- Permutation of length n

  {1,2,...,n} ⇒ π={i1,i2,...,in}

- Row π-permutation

  $A_{\pi,*} = \{a_{\pi(i),j}\} \quad A \in C^{n\times m}; \quad i = 1,n; \quad \mathrm{j} = 1,\mathrm{m}$

- Column π-permutation

  $A_{*,\pi} = \{a_{i,\pi(j)}\} \quad A \in C^{n\times m}; \quad i = 1,n; \quad \mathrm{j} = 1,\mathrm{m}$

- Interchange matrix P_{ij} : identity matrix with interchanged rows i and j
- Permutation matrix P_{π} : the identity matrix with its rows (or columns) permuted

Permutations and reordering

- π-permutation

$$A_{\pi,*} = P_\pi A; \quad P_\pi = \{P_{i_n,j_n}, P_{i_{n-1},j_{n-1}}, \ldots. P_{i_1,j_1})$$

$$A_{*,\pi} = AQ_\pi; \quad Q_\pi = \{P_{i_1,j_1}, P_{i_2,j_2}, \ldots, P_{i_{n-1},j_{n-1}}, P_{i_n,j_n})$$

$$A_{*,\pi} = \{a_{i,\pi(j)}\} \quad A \in C^{n\times m}; \quad i = 1, n; \quad \mathrm{j} = 1, \mathrm{m}$$

- P_π and Q_π are unitary matrices

$$P_\pi Q_\pi = I; \quad Q_\pi = P_\pi^{-1}$$

Example of permutation

- Permutation $\pi = \{1,3,2,4\}$

$$A = \begin{pmatrix} a_{11} & 0 & a_{13} & 0 \\ 0 & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & 0 \\ 0 & a_{42} & 0 & a_{44} \end{pmatrix}$$

- Columns 2, 3 are permuted

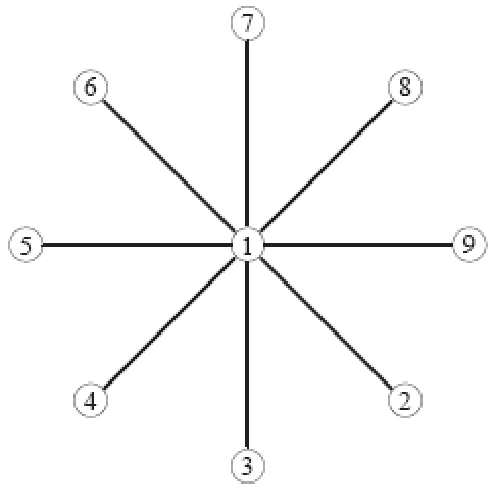
$$\begin{pmatrix} a_{11} & a_{13} & 0 & 0 \\ 0 & a_{23} & a_{22} & a_{24} \\ a_{31} & a_{33} & a_{32} & 0 \\ 0 & 0 & a_{42} & a_{44} \end{pmatrix}\begin{pmatrix} x_1 \\ x_3 \\ x_2 \\ x_4 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{pmatrix}$$

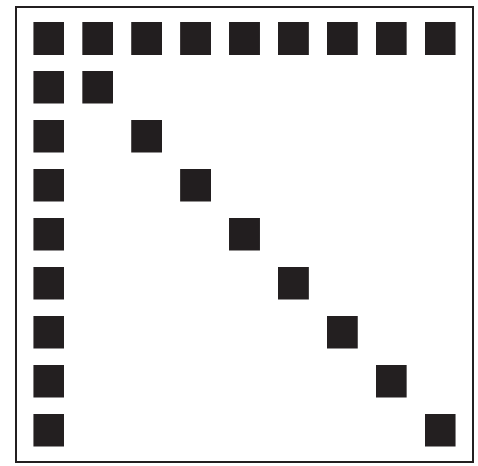
- Rows 2, 3 are permuted

$$\begin{pmatrix} a_{11} & a_{13} & 0 & 0 \\ a_{31} & a_{33} & a_{32} & 0 \\ 0 & a_{23} & a_{22} & a_{24} \\ 0 & 0 & a_{42} & a_{44} \end{pmatrix}\begin{pmatrix} x_1 \\ x_3 \\ x_2 \\ x_4 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_3 \\ b_2 \\ b_4 \end{pmatrix}$$

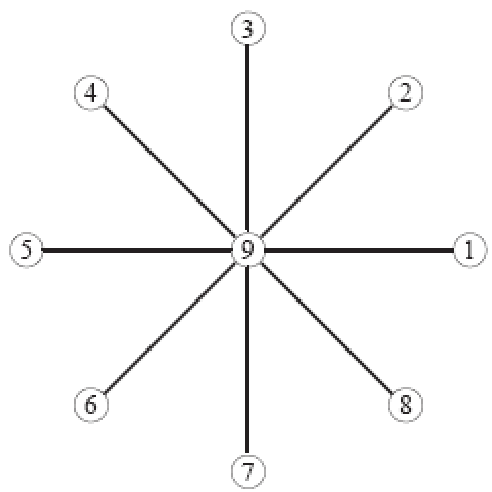
Relations with adjacency graph

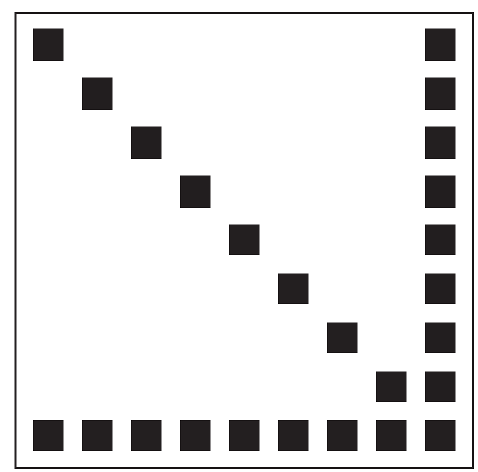
- A lot of fill-ins during Gaussian elimination



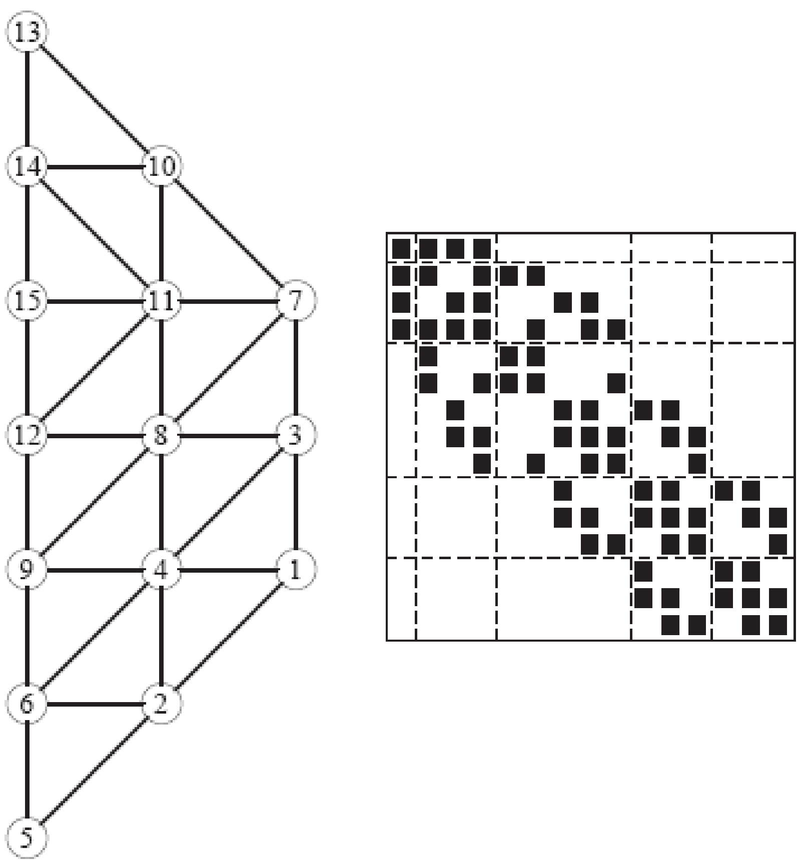


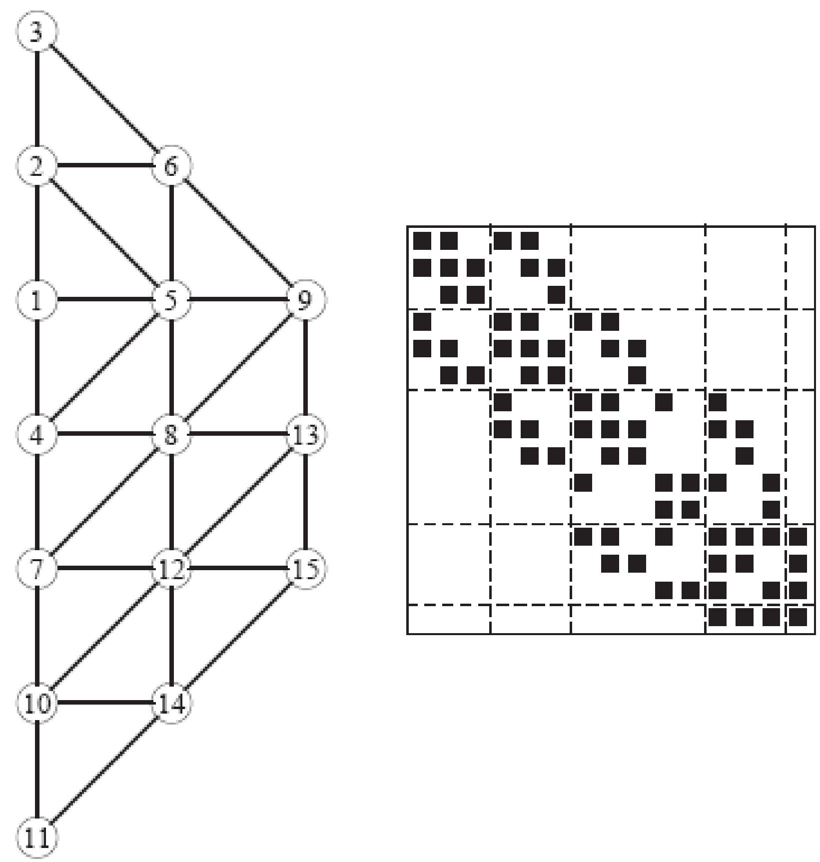
- No fill-ins during Gaussian elimination





Examples of reordering: standard and reverse Cuthill-McKee





Storage schemes and algorithms of matrix-by-vector multiplication for sparse matrices

Coordinate format

Compressed sparse row format (CRS)

Compressed sparse column format (CRC)

Modified sparse row format (MSR)

Modified sparse column format (MSC)

Diagonal format (DIAG)

Ellpack-Itpack

Coordinate format (COO)

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

AA	12.	9.	7.	5.	1.	2.	11.	3.	6.	4.	8.	10.	Nz
JR	5	3	3	2	1	1	4	2	3	2	3	4	Nz
JC	5	5	3	4	1	4	4	1	1	2	4	3	Nz

Nz – number of nonzero elements, n – number of rows

AA – nonzero entries

JR – row indices

JC – column indices

Compressed Sparse Row (CSR)

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

$$A \in C^{m \times n}$$

Nz=12
m=5
n=5

R1: 2(1) R2: 3(3) R3: 4(6) R4: 2(10) R5: 1(12)

AA	1.	2.	3.	4.	5.	6.	7.	8.	9.	10.	11.	12.	Nz
JA	1	4	1	2	4	1	3	4	5	3	4	5	Nz
IA	1	3	6	10	12	13	m+1						

IA(m+1)=IA(1)+Nz

Nz – number of nonzero elements, m – number of rows
AA – nonzero entries by rows,
JA – column indices; **IA** – pointers to the descriptions of rows
Description of i-th row: from IA(i) to IA(i+1)−1
Number of nonzero elements in i-th row: IA(i+1)−IA(i)

Compressed Sparse Column (CSC)

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

$$A \in C^{m \times n}$$

Nz=12
m=5
n=5

	C1: 3(<u>1</u>)			C2: 1(<u>4</u>)	C3: 2(<u>5</u>)		C4: 4(<u>7</u>)				C5: 2(<u>11</u>)		
AA	1.	3.	6.	4.	7.	10.	2.	5.	8.	11.	9.	12.	Nz
JA	1	2	3	2	3	4	1	2	3	4	3	5	Nz
IA	1	4	5	7	11	13	n+1						

IA(n+1)=IA(1)+Nz

Nz – number of nonzero elements, n – number of columns
AA – nonzero entries by columns,
JA – row indices; **IA** – pointers to the descriptions of columns
Description of i-th column: from IA(i) to IA(i+1)−1
Number of nonzero elements in i-th column: IA(i+1)−IA(i)

Modified Sparse Row (MSR)

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

$A \in C^{n\times n}$

Nz=12
n=5

Nondiagonal elements

Diagonal elements

n+1

R1: 1(<u>7</u>) R2: 2(<u>8</u>) R3: 3(<u>10</u>) R4: 1(<u>13</u>)

AA	1.	4.	7.	11.	12.	*	2.	3.	5.	6.	8.	9.	10.	Nz+1
JA	7	8	10	13	14	14	4	1	4	1	4	5	3	Nz+1

From 1 to n+1:
pointers to rows

From n+2 to Nz+1:
column indices

Nz – number of nonzero elements, n – size of matrix

AA – nonzero entries: main diagonal and nondiagonal elements by rows

JA – pointers to rows and column indices

Modified Sparse Column (MSC)

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

$A \in C^{n \times n}$

Nz=12
n=5

n+1

Diagonal elements | Nondiagonal elements

AA	1. 4. 7. 11. 12. *	?	Nz+1
JA	?	?	Nz+1

From 1 to n+1:
pointers to columns

From n+2 to Nz+1:
row indices

Nz – number of nonzero elements, n – size of matrix

AA – nonzero entries: main diagonal and nondiagonal elements by columns

JA – pointers to columns and row indices

Diagonal format (DIAG)

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

$$A \in C^{n \times n}$$

Nd=3
n=5

$$\mathtt{DIAG} = \boxed{\begin{array}{ccc} * & 1. & 2. \\ 3. & 4. & 5. \\ 6. & 7. & 8. \\ 9. & 10. & * \\ 11 & 12. & * \end{array}}$$

Main diagonal

$$\mathtt{IOFF} = \boxed{-1 \quad 0 \quad 2}$$

DIAG(i,j)←a(i,i+IOFF(j))

Nd – number of diagonals, n – size of matrix
DIAG – 2D array [1..n,1..Nd], its columns contain diagonals of the matrix
IOFF – array [1..Nd], contains offsets of diagonals with respect to the main diagonal

Ellpack-Itpack format

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

$$A \in C^{n \times m}$$

Nmax=3
n=5

$$\text{COEF} = \boxed{\begin{matrix} 1. & 2. & 0. \\ 3. & 4. & 5. \\ 6. & 7. & 8. \\ 9. & 10. & 0. \\ 11 & 12. & 0. \end{matrix}}$$

Unused positions

$$\text{JCOEF} = \boxed{\begin{matrix} 1 & 3 & 1 \\ 1 & 2 & 4 \\ 2 & 3 & 5 \\ 3 & 4 & 4 \\ 4 & 5 & 5 \end{matrix}}$$

Row numbers

Nmax – maximal number of nozero elements per row, n – number of rows
COEFF – 2D array [1..n,1..Nmax], its rows contain nonzero entries by rows
JCOEFF – 2D array [1..n,1..Nmax], its rows contain column positions of nonzero entries

Algorithms for matrix-by-vector multiplication

- **CSR format**

  N – number of rows, Ax=z

  IA – pointers of rows, JA – column indices

```
for i=1:N
z(i)=0
for j=IA(i):IA(i+1)-1
z(i)=z(i)+x(JA(j))*AA(j)
end
end
```

Algorithms for matrix-by-vector multiplication (continue)

- **CSC format**

  N – number of rows, M – number of columns, Ax=z
  IA – pointers of columns, JA – row indices

```
for i=1:N
z(i)=0
end
for j=1:M
for i=IA(j):IA(j+1)-1
z(JA(i))=z(JA(i))+x(j)*AA(i)
end
end
```

Summary

- Sparse matrix can be represented by its adjacency graph
- Permutations and reordering are used to reduce fill-ins in Direct solution methods
- Storage schemes for sparse matrices are aimed at representing only the nonzero elements
- The matrix-by-vector product is an important operation required in almost all iterative solution methods