FIS Tutorial 1
Matrix Storage Schemes

Matrix Storage Schemes

$$A = \begin{pmatrix} 1 & 0 & 4 & 0 \\ 0 & 2 & 2 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 4 & 2 \end{pmatrix} = (a_{ij})_{i,j=1,\dots,4}$$

$a_{1,3}$

Coordinate Format

correction

$I = (1, 1, 2, 2, 3, 4, 4)$ ← Row Indices

$J = (1, 3, 2, 3, 3, 3, 4)$ ← Column –"–

$V = (1, 4, 2, 2, 1, 4, 2)$ ← values

entries: $3 \cdot nz$ ← # nonzero entries in A

Compressed Sparse Row (CSR) Format

correction

$IA = (1, 3, 5, 6, 8)$ ← Row pointers

point to location in J, V where a new row starts

$J = (\boxed{1, 3}, \boxed{2, 3}, \boxed{3}, \boxed{3, 4})$ ← Column index

$V = (1, 4, 2, 2, 1, 4, 2)$ ← value

$2 \cdot nz + n + 1$ entries

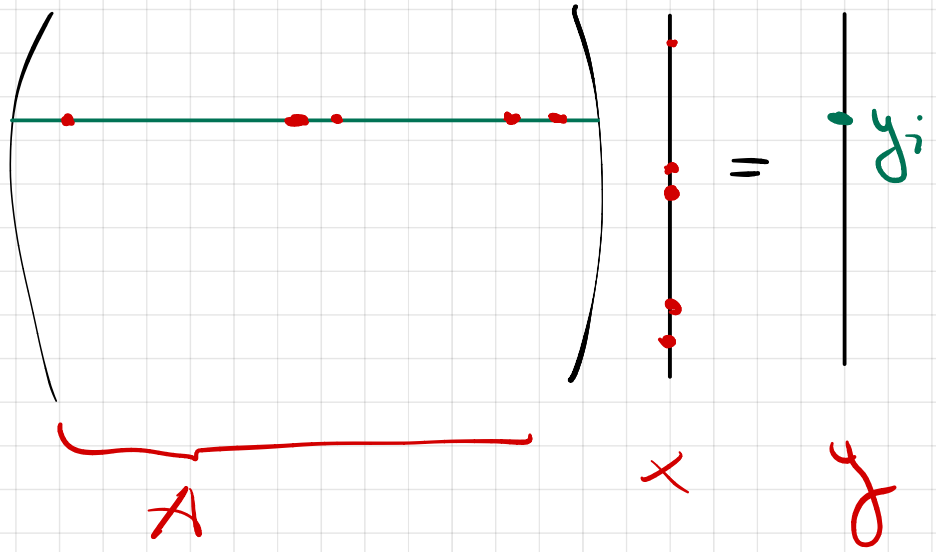
Matrix-Vector Product (CSR Format)

$A \in \mathbb{R}^{n \times n}$

$$(y_1, \dots, y_n)^T$$

$$Ax = y \in \mathbb{R}^n$$

$$y_i = \sum_{j=1}^{n} a_{ij} x_j$$




```
for i = 1, ..., n do
    i1 = IA(i)
    i2 = IA(i+1) - 1
    y(i) = DOT PROD (V(i1:i2), X(J(i1:i2)))
end for
```

Compressed Sparse Column (CSC) Format

$$IA = (1, 2, 3, 7, 8)$$

$$J = (\boxed{1}, \boxed{2}, \boxed{1, 2, 3, 4}, \boxed{4})$$

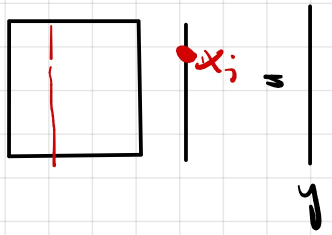
$$V = (1, 2, 4, 2, 1, 4, 2)$$

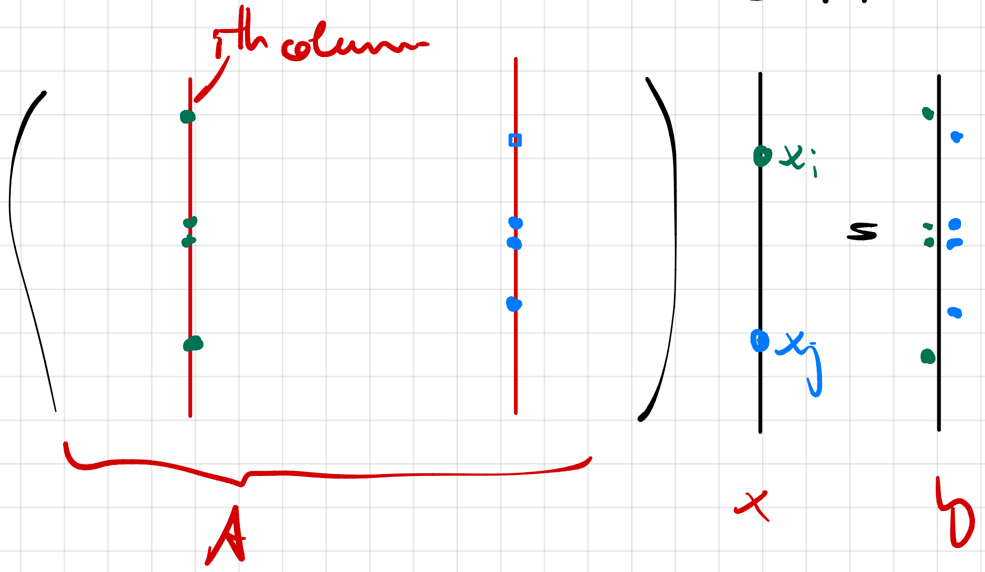
$$A = \begin{array}{cccc} & & a_{1,3} & \\ \end{array}$$
$$A = \begin{pmatrix} 1 & 0 & 4 & 0 \\ 0 & 2 & 2 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 4 & 2 \end{pmatrix}$$

Matrix-Vector Product (CSC Format)

$$A \in \mathbb{R}^{n \times n} \qquad A = (a_1 | \cdots | a_n)$$

$$Ax = y \qquad y = \sum_{i=1}^{n} x_i a_i$$





Algorithm:

```
y(1:n) = 0
for i = 1, ..., n do
    i1 = IA(i)
    i2 = IA(i+1) - 1
    y(J(i1:i2)) += V(i1:i2) · x(i)
end for
```

$a += b$

$a \leftarrow a + b$

Modified CSR (MSR)

correction

column indices

$$JM = (\underbrace{\overset{n+2 \text{ always}}{6}, 7, 8, 8, 9}_{n+1}, 3, 3, 3)$$

$$VM = (\underbrace{1, 2, 1, 2}_{n}, \cancel{6}, \boxed{4}, \boxed{2}, \boxed{4})$$

$$A = \begin{array}{cccc} & & a_{1,3} & \\ \end{array} \begin{pmatrix} 1 & 0 & 4 & 0 \\ 0 & 2 & 2 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 4 & 2 \end{pmatrix}$$

$2(nz + nd + 1)$ entries

#zeros on diagonal

Matrix-Vector Product

diagonal of A off-diagonal entries of A

$$Ax = y \iff (D + \tilde{A})x = y$$

$$= \underbrace{Dx}_{y_D} + \underbrace{\tilde{A}x}_{\text{CSR-like algorithm}} = y$$

$(y_D)_i = a_{ii} \cdot x_i$

If A symmetric

(strict) lower triangle

(strict) upper triangle

$$Ax = y \iff Dx + \underset{\underset{U^T}{\parallel}}{L}x + Ux = y$$

Store only one triangle, not the whole matrix!

Use CSR-like algo for the triangle that you store, CSC algo for the other one (i.e. the transpose)