Sep 4, 2020

Dense vectors / matrices

① vectors αx $\quad x+y \quad x^T y = \sum_i x_i y_i$ $\quad O(n)$ BLAS 1

② Matrix-vector $y = Ax$ $\quad y_i = \sum_j A_{ij} x_j$ $\quad O(n^2)$ BLAS 2

③ Matrix-matrix $C = AB$ $\quad C_{ij} = \sum_k A_{ik} B_{kj}$ $\quad O(n^3)$ BLAS 3

Basic linear algebra subroutines

Lots of HPC implementations of BLAS

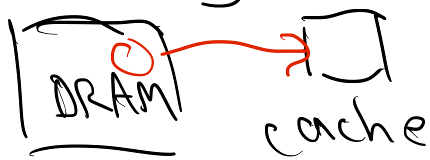
Ex: Intel MKL

Today: some basics on performance

Dense $\Rightarrow$ store as contiguous arrays

column major

Two main costs:
- arithmetic (flops = floating point operations)
- fetching data (communication)

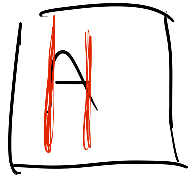


cost(comm) >> cost(flops)

$\hookrightarrow$ (1) spatial locality: faster to access memory sequentially

(2) temporal locality: re-use what's in cache

Example: mat-vec $y = Ax$ $A \in \mathbb{R}^{m \times n}$




```
y = 0
for i = 1:m
    for j = 1:n
        y_i += A_ij x_j
```

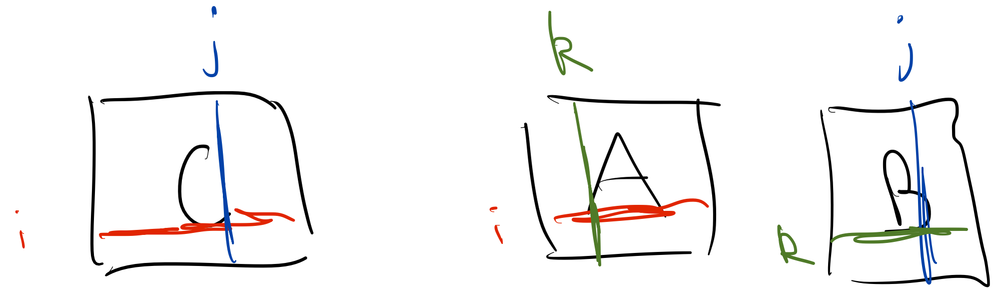
temporal locality

```
y = 0
for j = 1:n
    for i = 1:m
        y_i += A_ij x_j
```

BLAS	flops	data
1	$O(n)$	$O(n)$
2	$O(n^2)$	$O(n^2)$
3	$O(n^3)$	$O(n^2)$

major opportunity

$C = AB \qquad A, B \in \mathbb{R}^{n \times n}$



$C = 0$

for $i = 1:n$

  for $j = 1:n$

    for $k = 1:n$

      $C_{ij} \mathrel{+}= A_{ik} B_{kj}$ $\quad O(n)$

cache size $<<$ n

can't get good temporal locality for A, B

Key to BLAS 3 performance are blocking schemes

$$\begin{bmatrix} - & a_1^T & - \\ & \vdots & \\ - & a_n^T & - \end{bmatrix} \begin{bmatrix} | & & | \\ b_1 & \cdots & b_n \\ | & & | \end{bmatrix}$$

$$\begin{pmatrix} \vdots \end{pmatrix} \begin{pmatrix} - - - - \end{pmatrix} \qquad xy^T$$

$$(xy^T)_{ij} = x_i x_j$$

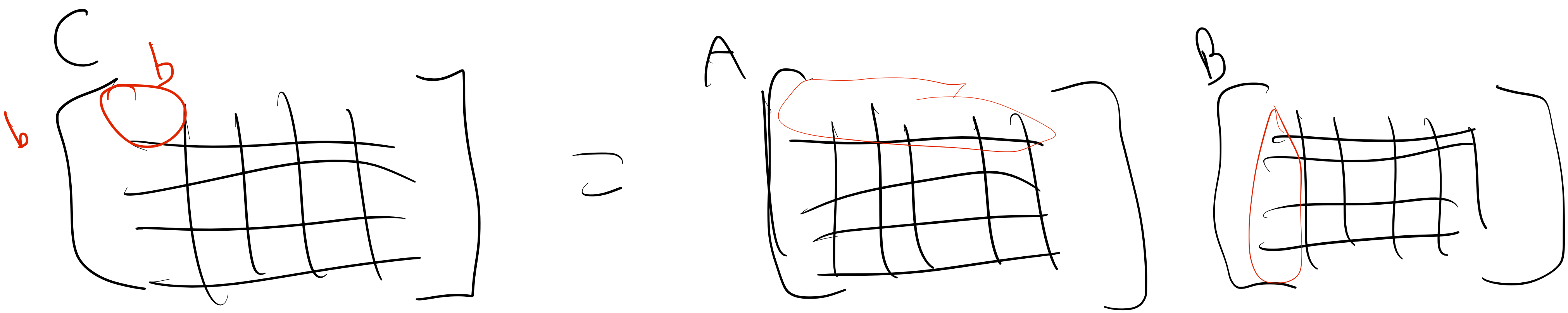
$a_i^T = A(i, :)$

$C_{ij} = a_i^T b_j$

Two-way blocking

$$\begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix} = {}^{n}\!/_{2}\overset{n/2}{\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}} \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix}$$

$$C_{ij} = \sum_{k=1}^{2} A_{ik} B_{kj} \qquad C_{11} = A_{11}B_{11} + A_{12}B_{21}$$



$3b^2 \approx$ cache size

lots of work to make this fast!

Rank

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

range(A) $\iff$ "col space of A" $\iff$ $\{Ax \mid x \in \mathbb{R}^n\}$

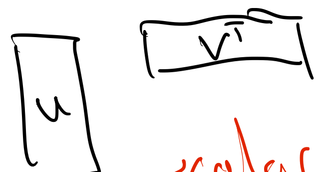
$$\text{rank}(A) = \dim(\text{range}(A))$$

"full rank" $\iff$ $\text{rank}(A) = n$

null(A) $\iff$ "null space of A" $\iff$ $\{z \in \mathbb{R}^n \mid Az = 0\}$

rank-nullity theorem: $\text{rank}(A) + \dim(\text{null}(A)) = n$

Example: $A = uv^T$



$A_{ij} = u_i v_j$

$$\text{range}(A) = \{Ax\} = \{u\overbrace{v^T x}^{\text{scalar}}\} = \{\alpha u \mid \alpha \in \mathbb{R}\}$$

$$\Rightarrow \text{rank}(uv^T) = 1$$

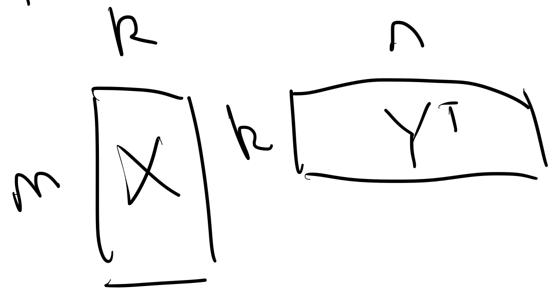
$\dim(\text{null}(A)) = n - 1$ $\quad A = uv^T$

Structured matrices

Low-rank

$A \in \mathbb{R}^{m \times n}$ $\quad \text{rank}(A) = k < n$

$A = XY^T$



starting w/ X, Y

mat-vec: Az

① form A $\quad O(mkn)$

② $y = Az$ $\quad O(mn)$

$O(mkn)$

mat-vec: $X(Y^T z)$

① $w = Y^T z$

② $y = Xw$

$O(kn + mk)$

Nonsingular

$A \in \mathbb{R}^{n \times n}$ $\quad \text{rank}(A) = n$ full rank

$b \Rightarrow \exists\, x : \; Ax = b \qquad x = A^{-1}b$

expansion of b in cols of A

not nonsingular $\Rightarrow$ singular

$A(A^{-1}b) = (AA^{-1})b = Ib = b$

Orthogonal $\quad Q \in \mathbb{R}^{n \times n} \quad Q^T Q = I \quad (Q^{-1} = Q^T)$

$$q_i^T q_j = \begin{cases} 1 & i = j \\ 0 & o/w \end{cases}$$

Symmetric $\quad A \in \mathbb{R}^{n \times n} \quad A = A^T \quad (A^T)_{ij} = A_{ji}$

$$\boxed{x} \xrightarrow{T} \boxed{x^T}$$

$$(AB)^T = B^T A^T$$

$$(A^{-1})^T = (A^T)^{-1}$$

$$A_{ij} = A_{ji}$$

1/2 storage

Sparse

Diagonal $\begin{pmatrix} \ddots & & 0 \\ & \ddots & \\ 0 & & \ddots \end{pmatrix}$ $D_{ij} = 0 \quad i \neq j$

Identity: $\begin{pmatrix} 1 & \cdots & 0 \\ 0 & \cdots & 1 \end{pmatrix}$

Tridiagonal $\begin{pmatrix} \ddots & \ddots & 0 \\ \ddots & \ddots & \ddots \\ 0 & \ddots & \ddots \end{pmatrix}$ $T_{ij} = 0$ if $i \notin \{j-1, j, j+1\}$

Permutation $P_{ij} \in \{0, 1\}$ each row/col has exactly one 1

$AP \qquad P^T P = I$ (orthogonal)