

## 9. Numerical linear algebra background

- matrix structure and algorithm complexity
- solving linear equations with factored matrices
- LU, Cholesky,  $LDL^T$  factorization
- block elimination and the matrix inversion lemma
- solving underdetermined equations

# Matrix structure and algorithm complexity

cost (execution time) of solving  $Ax = b$  with  $A \in \mathbf{R}^{n \times n}$

- for general methods, grows as  $n^3$
- less if  $A$  is structured (banded, sparse, Toeplitz, . . . )

## flop counts

- flop (floating-point operation): one addition, subtraction, multiplication, or division of two floating-point numbers
- to estimate complexity of an algorithm: express number of flops as a (polynomial) function of the problem dimensions, and simplify by keeping only the leading terms
- not an accurate predictor of computation time on modern computers
- useful as a rough estimate of complexity

## vector-vector operations ( $x, y \in \mathbf{R}^n$ )

- inner product  $x^T y$ :  $2n - 1$  flops (or  $2n$  if  $n$  is large)
- sum  $x + y$ , scalar multiplication  $\alpha x$ :  $n$  flops

## matrix-vector product $y = Ax$ with $A \in \mathbf{R}^{m \times n}$

- $m(2n - 1)$  flops (or  $2mn$  if  $n$  large)
- $2N$  if  $A$  is sparse with  $N$  nonzero elements
- $2p(n + m)$  if  $A$  is given as  $A = UV^T$ ,  $U \in \mathbf{R}^{m \times p}$ ,  $V \in \mathbf{R}^{n \times p}$

## matrix-matrix product $C = AB$ with $A \in \mathbf{R}^{m \times n}$ , $B \in \mathbf{R}^{n \times p}$

- $mp(2n - 1)$  flops (or  $2mnp$  if  $n$  large)
- less if  $A$  and/or  $B$  are sparse
- $(1/2)m(m + 1)(2n - 1) \approx m^2 n$  if  $m = p$  and  $C$  symmetric

# Linear equations that are easy to solve

**diagonal matrices** ( $a_{ij} = 0$  if  $i \neq j$ ):  $n$  flops

$$x = A^{-1}b = (b_1/a_{11}, \dots, b_n/a_{nn})$$

**lower triangular** ( $a_{ij} = 0$  if  $j > i$ ):  $n^2$  flops

$$x_1 := b_1/a_{11}$$

$$x_2 := (b_2 - a_{21}x_1)/a_{22}$$

$$x_3 := (b_3 - a_{31}x_1 - a_{32}x_2)/a_{33}$$

$\vdots$

$$x_n := (b_n - a_{n1}x_1 - a_{n2}x_2 - \dots - a_{n,n-1}x_{n-1})/a_{nn}$$

called forward substitution

**upper triangular** ( $a_{ij} = 0$  if  $j < i$ ):  $n^2$  flops via backward substitution

**orthogonal matrices:**  $A^{-1} = A^T$

- $2n^2$  flops to compute  $x = A^T b$  for general  $A$
- less with structure, *e.g.*, if  $A = I - 2uu^T$  with  $\|u\|_2 = 1$ , we can compute  $x = A^T b = b - 2(u^T b)u$  in  $4n$  flops

**permutation matrices:**

$$a_{ij} = \begin{cases} 1 & j = \pi_i \\ 0 & \text{otherwise} \end{cases}$$

where  $\pi = (\pi_1, \pi_2, \dots, \pi_n)$  is a permutation of  $(1, 2, \dots, n)$

- interpretation:  $Ax = (x_{\pi_1}, \dots, x_{\pi_n})$
- satisfies  $A^{-1} = A^T$ , hence cost of solving  $Ax = b$  is 0 flops

example:

$$A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}, \quad A^{-1} = A^T = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

## The factor-solve method for solving $Ax = b$

- factor  $A$  as a product of simple matrices (usually 2 or 3):

$$A = A_1 A_2 \cdots A_k$$

( $A_i$  diagonal, upper or lower triangular, etc)

- compute  $x = A^{-1}b = A_k^{-1} \cdots A_2^{-1} A_1^{-1} b$  by solving  $k$  'easy' equations

$$A_1 x_1 = b, \quad A_2 x_2 = x_1, \quad \dots, \quad A_k x = x_{k-1}$$

cost of factorization step usually dominates cost of solve step

**equations with multiple righthand sides**

$$Ax_1 = b_1, \quad Ax_2 = b_2, \quad \dots, \quad Ax_m = b_m$$

cost: one factorization plus  $m$  solves

# LU factorization

every nonsingular matrix  $A$  can be factored as

$$A = PLU$$

with  $P$  a permutation matrix,  $L$  lower triangular,  $U$  upper triangular

cost:  $(2/3)n^3$  flops

---

*Solving linear equations by LU factorization.*

**given** a set of linear equations  $Ax = b$ , with  $A$  nonsingular.

1. *LU factorization.* Factor  $A$  as  $A = PLU$  ( $(2/3)n^3$  flops).
  2. *Permutation.* Solve  $Pz_1 = b$  (0 flops).
  3. *Forward substitution.* Solve  $Lz_2 = z_1$  ( $n^2$  flops).
  4. *Backward substitution.* Solve  $Ux = z_2$  ( $n^2$  flops).
- 

cost:  $(2/3)n^3 + 2n^2 \approx (2/3)n^3$  for large  $n$

## sparse LU factorization

$$A = P_1 L U P_2$$

- adding permutation matrix  $P_2$  offers possibility of sparser  $L, U$  (hence, cheaper factor and solve steps)
- $P_1$  and  $P_2$  chosen (heuristically) to yield sparse  $L, U$
- choice of  $P_1$  and  $P_2$  depends on sparsity pattern and values of  $A$
- cost is usually much less than  $(2/3)n^3$ ; exact value depends in a complicated way on  $n$ , number of zeros in  $A$ , sparsity pattern

# Cholesky factorization

every positive definite  $A$  can be factored as

$$A = LL^T$$

with  $L$  lower triangular

cost:  $(1/3)n^3$  flops

---

*Solving linear equations by Cholesky factorization.*

**given** a set of linear equations  $Ax = b$ , with  $A \in \mathbf{S}_{++}^n$ .

1. *Cholesky factorization.* Factor  $A$  as  $A = LL^T$  ( $(1/3)n^3$  flops).
2. *Forward substitution.* Solve  $Lz_1 = b$  ( $n^2$  flops).
3. *Backward substitution.* Solve  $L^T x = z_1$  ( $n^2$  flops).

---

cost:  $(1/3)n^3 + 2n^2 \approx (1/3)n^3$  for large  $n$

## sparse Cholesky factorization

$$A = PLL^T P^T$$

- adding permutation matrix  $P$  offers possibility of sparser  $L$
- $P$  chosen (heuristically) to yield sparse  $L$
- choice of  $P$  only depends on sparsity pattern of  $A$  (unlike sparse LU)
- cost is usually much less than  $(1/3)n^3$ ; exact value depends in a complicated way on  $n$ , number of zeros in  $A$ , sparsity pattern

# LDL<sup>T</sup> factorization

every nonsingular symmetric matrix  $A$  can be factored as

$$A = PLDL^T P^T$$

with  $P$  a permutation matrix,  $L$  lower triangular,  $D$  block diagonal with  $1 \times 1$  or  $2 \times 2$  diagonal blocks

cost:  $(1/3)n^3$

- cost of solving symmetric sets of linear equations by LDL<sup>T</sup> factorization:  
 $(1/3)n^3 + 2n^2 \approx (1/3)n^3$  for large  $n$
- for sparse  $A$ , can choose  $P$  to yield sparse  $L$ ; cost  $\ll (1/3)n^3$

## Equations with structured sub-blocks

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} \quad (1)$$

- variables  $x_1 \in \mathbf{R}^{n_1}$ ,  $x_2 \in \mathbf{R}^{n_2}$ ; blocks  $A_{ij} \in \mathbf{R}^{n_i \times n_j}$
- if  $A_{11}$  is nonsingular, can eliminate  $x_1$ :  $x_1 = A_{11}^{-1}(b_1 - A_{12}x_2)$ ; to compute  $x_2$ , solve

$$(A_{22} - A_{21}A_{11}^{-1}A_{12})x_2 = b_2 - A_{21}A_{11}^{-1}b_1$$

---

*Solving linear equations by block elimination.*

**given** a nonsingular set of linear equations (1), with  $A_{11}$  nonsingular.

1. Form  $A_{11}^{-1}A_{12}$  and  $A_{11}^{-1}b_1$ .
2. Form  $S = A_{22} - A_{21}A_{11}^{-1}A_{12}$  and  $\tilde{b} = b_2 - A_{21}A_{11}^{-1}b_1$ .
3. Determine  $x_2$  by solving  $Sx_2 = \tilde{b}$ .
4. Determine  $x_1$  by solving  $A_{11}x_1 = b_1 - A_{12}x_2$ .

## dominant terms in flop count

- step 1:  $f + n_2s$  ( $f$  is cost of factoring  $A_{11}$ ;  $s$  is cost of solve step)
- step 2:  $2n_2^2n_1$  (cost dominated by product of  $A_{21}$  and  $A_{11}^{-1}A_{12}$ )
- step 3:  $(2/3)n_2^3$

total:  $f + n_2s + 2n_2^2n_1 + (2/3)n_2^3$

## examples

- general  $A_{11}$  ( $f = (2/3)n_1^3$ ,  $s = 2n_1^2$ ): no gain over standard method

$$\text{\#flops} = (2/3)n_1^3 + 2n_1^2n_2 + 2n_2^2n_1 + (2/3)n_2^3 = (2/3)(n_1 + n_2)^3$$

- block elimination is useful for structured  $A_{11}$  ( $f \ll n_1^3$ )

for example, diagonal ( $f = 0$ ,  $s = n_1$ ):  $\text{\#flops} \approx 2n_2^2n_1 + (2/3)n_2^3$

## Structured matrix plus low rank term

$$(A + BC)x = b$$

- $A \in \mathbf{R}^{n \times n}$ ,  $B \in \mathbf{R}^{n \times p}$ ,  $C \in \mathbf{R}^{p \times n}$
- assume  $A$  has structure ( $Ax = b$  easy to solve)

first write as

$$\begin{bmatrix} A & B \\ C & -I \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} b \\ 0 \end{bmatrix}$$

now apply block elimination: solve

$$(I + CA^{-1}B)y = CA^{-1}b,$$

then solve  $Ax = b - By$

this proves the **matrix inversion lemma**: if  $A$  and  $A + BC$  nonsingular,

$$(A + BC)^{-1} = A^{-1} - A^{-1}B(I + CA^{-1}B)^{-1}CA^{-1}$$

**example:**  $A$  diagonal,  $B, C$  dense

- method 1: form  $D = A + BC$ , then solve  $Dx = b$

cost:  $(2/3)n^3 + 2pn^2$

- method 2 (via matrix inversion lemma): solve

$$(I + CA^{-1}B)y = CA^{-1}b, \quad (2)$$

then compute  $x = A^{-1}b - A^{-1}By$

total cost is dominated by (2):  $2p^2n + (2/3)p^3$  (*i.e.*, linear in  $n$ )

# Underdetermined linear equations

if  $A \in \mathbf{R}^{p \times n}$  with  $p < n$ ,  $\mathbf{rank} A = p$ ,

$$\{x \mid Ax = b\} = \{Fz + \hat{x} \mid z \in \mathbf{R}^{n-p}\}$$

- $\hat{x}$  is (any) particular solution
- columns of  $F \in \mathbf{R}^{n \times (n-p)}$  span nullspace of  $A$
- there exist several numerical methods for computing  $F$   
(QR factorization, rectangular LU factorization, . . .)