Appendix C Numerical linear algebra background

Last update on 2022-05-04 17:22

Table of contents

Matrix structure and algorithm complexity

Solving linear equations with factored matrices

LU, Cholesky, LDL^T factorization

Block elimination and matrix inversion lemma

Solving underdetermined equations

Matrix structure and algorithm complexity

Solving linear equations with factored matrices

LU, Cholesky, LDL^T factorization

Block elimination and matrix inversion lemma

Solving underdetermined equations

Complexity via flop count

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

- 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 with $x, y \in \mathbb{R}^n$

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

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

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

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

- $mp(2n-1)$ flops ($\approx 2mnp$ if n is 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

Matrix structure and algorithm complexity

Solving linear equations with factored matrices

LU, Cholesky, LDL^T factorization

Block elimination and matrix inversion lemma

Solving underdetermined equations

Linear equations that are easy to solve

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

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

lower triangular ($a_{ij} = 0$ if $j > i$) $\quad$ n^2 flops via forward substitution

$$\begin{aligned}
x_1 &= b_1/a_{11} \\
x_2 &= (b_2 - a_{21}x_1)/a_{22} \\
&\vdots \\
x_n &= (b_n - a_{n1}x_1 - \cdots - a_{n,n-1}x_{n-1})/a_{nn}
\end{aligned}$$

upper triangular ($a_{ij} = 0$ if $j < i$) $\quad$ 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 $\pi = (\pi_1, \pi_2, \ldots, \pi_n)$ is a permutation of $(1, 2, \ldots, n)$

- interpretation: $Ax = (x_{\pi_1}, x_{\pi_2}, \ldots, x_{\pi_n})$
- satisfies $A^{-1} = A^T$, hence cost of solving $Ax = b$ is 0 flops

Factor-solve method for solving $Ax = b$

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

$$A = A_1 A_2 \dots A_k$$

  where A_i diagonal, upper or lower triangular, etc.

- compute $x = A^{-1}b = A_k^{-1} \dots A_2^{-1} A_1^{-1} b$ 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 usually dominates cost of solve

- 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

Matrix structure and algorithm complexity

Solving linear equations with factored matrices

LU, Cholesky, LDL^T factorization

Block elimination and matrix inversion lemma

Solving underdetermined equations

assume that $A \in \mathbb{R}^{n \times n}$ is nonsingular

LU factorization

$$A = PLU$$

where P permutation matrix, L lower triangular, U upper triangular

$$\text{cost} = (2/3)n^3 \text{ flops}$$

solving linear equations by LU factorization

---

given a system of linear equations $Ax = b$ with A nonsingular

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

$$\text{total cost} = (2/3)n^3 + 2n^2 \approx (2/3)n^3$$

assume further that A is sparse

sparse LU factorization

$$A = P_1 L U P_2$$

- adding permutation matrix P_2 offers possibility of sparser L and U
- P_1 and P_2 chosen (heuristically) to yield sparse L and 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, and sparsity pattern

Cholesky factorization

assume that $A \in \mathbb{S}^n_{++}$

Cholesky factorization

$$A = LL^T$$

where L lower triangular

$$\text{cost} = (1/3)n^3 \text{ flops}$$

solving linear equations by Cholesky factorization

---

given a system of linear equations $Ax = b$ with $A \in \mathbb{S}^n_{++}$

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

---

$$\text{total cost} = (1/3)n^3 + 2n^2 \approx (1/3)n^3$$

assume further that A is sparse

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 depends only 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, and sparsity pattern

assume that $A \in \mathbb{S}^n$ is nonsingular

LDL^T factorization

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

where P permutation matrix, L lower triangular, D block diagonal with nonsingular 1×1 or 2×2 diagonal blocks

$$\text{cost} = (1/3)n^3 \text{ flops}$$

- cost of solving system of linear equations $Ax = b$ by LDL^T factorization

$$(1/3)n^3 + 2n^2 + cn \approx (1/3)n^3$$

- for sparse A, can choose P to yield sparse L, with cost much less than $(1/3)n^3$

Matrix structure and algorithm complexity

Solving linear equations with factored matrices

LU, Cholesky, LDL^T factorization

Block elimination and matrix inversion lemma

Solving underdetermined equations

Equations with structured subblocks

assume the system of linear equations $Ax = b$ can be written in the block form

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

where vabiables $x_1 \in \mathbb{R}^{n_1}$ and $x_2 \in \mathbb{R}^{n_2}$; blocks $A_{ij} \in \mathbb{R}^{n_i \times n_j}$

- if A_{11} is nonsingular, can eliminate x_1 by

$$x_1 = A_{11}^{-1}(b_1 - A_{12}x_2)$$

- to compute x_2, solve the reduced equation

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

- the matrix

$$S = A_{22} - A_{21}A_{11}^{-1}A_{12}$$

is called the **Schur complement** of A_{11} in A; S is nonsingular iff A is nonsingular

solving linear equations by block elimination

---

given a system of linear equations with A and 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 $\bar{b} = b_2 - A_{21}A_{11}^{-1}b_1$.
3. Determine x_2 by solving $Sx_2 = \bar{b}$.
4. Determine x_1 by solving $A_{11}x_1 = b_1 - A_{12}x_2$.

---

dominant terms in flop count

- step 1: $f + n_2 s$ (f is cost of factoring A_{11}; s is cost of solve step)
- step 2: $2n_2^2 n_1$ (cost dominated by product of A_{21} and $A_{11}^{-1}A_{12}$)
- step 3: $(2/3)n_2^3$ (LU factorization)
- step 4: neglected (A_{11} already factored in step 1)

$$\text{total cost} \approx f + n_2 s + 2n_2^2 n_1 + (2/3)n_2^3$$

- for general A_{11}, standard methods give $f = (2/3)n_1^3$ and $s = 2n_1^2$

$$\text{total cost} \approx (2/3)n_1^3 + 2n_1^2 n_2 + 2n_2^2 n_1 + (2/3)n_2^3 = (2/3)(n_1 + n_2)^3$$

- for structured A_{11}, could be much smaller, e.g., if A_{11} diagonal, $f = 0$ and $s = n_1$

$$\text{total cost} \approx 2n_2^2 n_1 + (2/3)n_2^3$$

Structured matrix plus low rank term

assume $A \in \mathbb{R}^{n \times n}$, $B \in \mathbb{R}^{n \times p}$, $C \in \mathbb{R}^{p \times n}$, consider

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

- write equivalently as

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

- apply block elimination, first solve

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

- then solve

$$Ax = b - By$$

Matrix inversion lemma

- **matrix inversion lemma** if A and $A+BC$ are nonsingular, then

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

- particularly useful when A has structure, and p small (BC low rank)

example A diagonal

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

$$\text{cost} \approx (2/3)n^3 + 2pn^2$$

- method 2: first solve $(I + CA^{-1}B)y = CA^{-1}b$, then solve $Ax = b - By$

$$\text{cost} \approx 2p^2n + (2/3)p^3$$

dominated by solving for y

Matrix structure and algorithm complexity

Solving linear equations with factored matrices

LU, Cholesky, LDL^T factorization

Block elimination and matrix inversion lemma

Solving underdetermined equations

Underdetermined linear equations

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

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

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