Numerical linear algebra Ⅲ

Woojoo Lee

Now we will see how to estimate the complexity of an algorithm.

Our focus is to solve

$$Ax = b.$$

We will express the number of flops (floating-point operation) as a function of the problem dimensions, and simplify it by keeping the leading term only.

Remark) The flop count is a useful measure, but a rough estimate of complexity.

Vector operations

Let $x, y \in R^n$.

Q) How many flops are necessary to compute $x + y$?

Q) For a scalar α, how many flops are necessary to compute αx ?

Q) How many flops are necessary to compute $x^T y$?

Matrix-Vector operations

Let $x \in R^n$ and $A \in R^{m \times n}$ and $y = Ax$.

Q) How many flops are necessary to compute y ?

Q) Suppose that $m = n$.

- What if A is a diagonal matrix ?
- What if A is a lower triangular matrix ?
- What if A has many zeroes ?

Matrix-Matrix operations

Let $A \in R^{m \times n}$, $B \in R^{n \times p}$ and $C = AB$.

Q) How many flops are necessary to compute C ?

Mixed operations

Q) Consider ABx where x is an R^p vector.

Remark)The computaion time really depends on the order in which you multiply several matrices.

System of linear equations.

$$Ax = b$$

where A is $n \times n$ matrix and x and b are column vectors.
We assume that A is nonsingular, so the solution is unique.

Remark) # of flops of standard methods to solve $Ax = b \sim n^3$.

$\rightarrow$ When n is very large (Big-data era ?),
this is really time consuming !

$\rightarrow$ In many cases, A has a special structure.

Consider ”simple” problems first.

Q) Suppose that A is a diagonal matrix.
How many flops are necessary for computing $A^{-1}b$?

Remark) Proper permutations should precede other operations.

Q) Suppose that A is a lower-triangular matrix with nonzero diagonals.
How many flops are necessary for computing $A^{-1}b$?

(This method is known as forward substitution.)

Q) How about an upper traingular matrix ?

Q) Suppose that A is a lower-triangular matrix with nonzero diagonals and it has further structure: $a_{ij} = 0$ if $|i - j| \geq 2$. How many flops are necessary for computing $A^{-1}b$?

Q) Suppose that A is an orthogonal matrix.
How many flops are necessary for computing $A^{-1}b$?

Remark)) If there is a further structure on an orthogonal matrix A, we can reduce the computation time.
For example, consider $A = I - 2uu^T$ where $||u||_2 = 1$.
Q) How many flops are necessary for computing $A^{-1}b$?

In order to solve $Ax = b$, various matrix factorization methods

$$A = A_1 A_2 \cdots A_k$$

are useful in many applications. Why ?

The total number of flops consists of two parts:

1. Cost of factoring A (usually, larger !)
2. Cost of solving the k linear equations

Example

- When we solve $Ax_i = b_i$ for $i = 1, \cdots, m(>> 1)$, the total number of flops is similar to that of factoring A.

Cholesky factorization

For a symmetric positive definite A $(n \times n)$,

$$A = LL^T$$

where L is a lower triangular matrix, nonsingular and positive diagonals. In particular, L is called the Cholesky factor of A.

In general, the cost of computing Cholesky factorization is $n^3/3$.

To solve $Ax = b$ with Cholesky factorization, we need

1. Cholesky factorization $\sim n^3/3$ flops
2. Forward substitution $\sim n^2$ flops
3. Backward substitution $\sim n^2$ flops

Remark) In some computer softwares, L is defined to be an upper triangular matrix.

LU factorization (the standard method for $Ax = b$)

For every nonsingular matrix A $(n \times n)$,

$$A = LU$$

where $L(n \times n)$ is unit lower triangular matrix and $U(n \times n)$ is upper triangular and nonsingular. Note that some cases may require proper permutation steps.

The cost of computing the LU factorization of a dense matrix is $\frac{2}{3}n^3$. (if no structure in A is exploited)

Steps

1. LU factorization $\sim \frac{2}{3}n^3$ flops
2. Forward substitution $\sim n^2$ flops
3. Backward substitution $\sim n^2$ flops

Sparse matrices

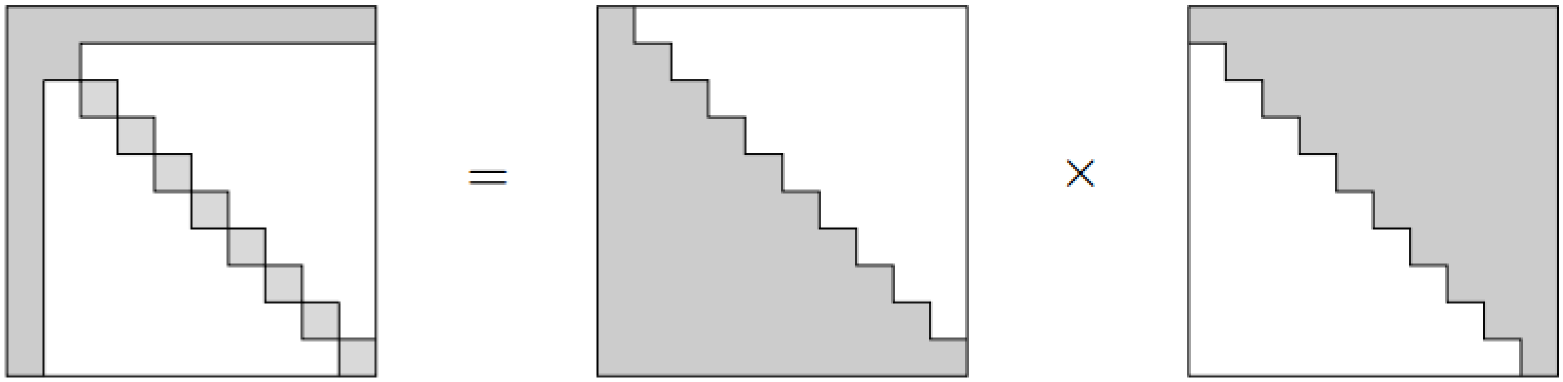
A matrix is sparse if most of elements are zero. Otherwise, it is called dense.

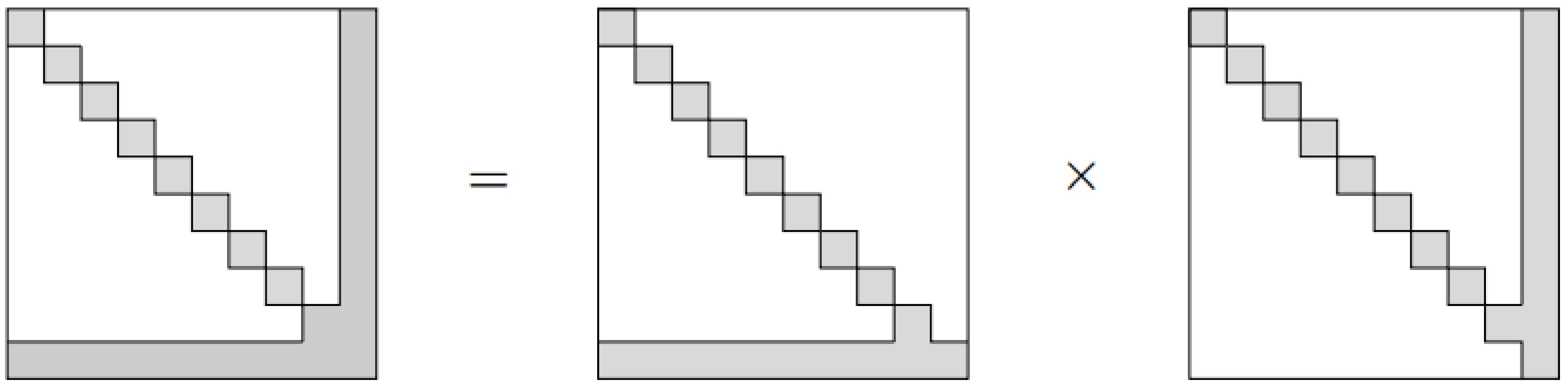
A useful R package to deal with sparse matrix computations

1. Koenker, R. and Ng, P. (2003) SparseM: A Sparse Matrix Package for R
2. as.matrix.csr
3. A set of commonly used linear algebra operations: t,chol,solve etc
4. Check examples showing benefits from sparse matrix computations

Cholesky facorization of sparse matries

1. If A is sparse, L is often sparse. But, not always.
2. If L is sparse, the number of flops for Cholesky factorization is much less than $n^3/3$.
3. Using sparsity pattern is important for wise computation.





Therefore, we should consider permuting the rows and columns of A before implementing Cholesky factorization.

Block elimination method
Step 1) Eliminating a subset of the variables.
Step 2) Solving a smaller system of linear equations for the remaining variables.

For $Ax = b$, we partition x into two subvectors:

$$x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

where $x_1 \in R^{n_1}$ and $x_2 \in R^{n_2}$.

$\rightarrow$

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

where $A_{11} \in R^{n_1 \times n_1}$ and $A_{22} \in R^{n_2 \times n_2}$.

Suppose that A_{11} is invertible.
Q) Express x_1 in terms of x_2 firstly and find x_2.

Example - exploiting structure in other blocks

1)

$$\begin{pmatrix} A_{11} & 0 \\ A_{21} & A_{22} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}$$

2) A_{11} is diagonal.

3) For sparse $A \in R^{n\times n}, B \in R^{n\times p}$ and $C \in R^{p\times n}$ (but $A + BC$ is dense),

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

where A is nonsingular.