Flops of Some Algos

Wan

November 8, 2023

Back-Substitution, operation count

ALGORITHM : 1 . *Back-Substitution algorithm*

For $i = n : -1 : 1$ *do:*
  $t := b_i$
  For $j = i + 1 : n$ *do*
    $t := t - a_{ij} x_j$
  End
  $x_i = t / a_{ii}$
End

$$\left.\begin{array}{l} \text{For } j = i+1 : n \text{ do} \\ \quad t := t - a_{ij}x_j \\ \text{End} \end{array}\right\} \begin{array}{l} t := b_i - (a_{i,i+1:n}, x_{i+1:n}) \\ \quad = b_i - \text{an inner product} \end{array}$$

- We must require that each $a_{ii} \neq 0$
- Operation count?

Forward-Substitution, operation count

Gaussian Elimination

✍3 Exact operation count for GE.

Solution:

$$
\begin{aligned}
T &= \sum_{k=1}^{n-1} \sum_{i=k+1}^{n} (2(n-k)+3) \\
&= \sum_{k=1}^{n-1} (2(n-k)+3)(n-k) \\
&= 2\sum_{k=1}^{n-1}(n-k)^2 + 3\sum_{k=1}^{n-1}(n-k) \\
&= 2\sum_{j=1}^{n-1} j^2 + 3\sum_{j=1}^{n-1} j
\end{aligned}
$$

In the last step we made a change of variables $j = n-k$. Now we know that $\sum_{k=1}^{n} k^2 = n(n+1)(2n+1)/6$ and $\sum_{k=1}^{n} k = n(n+1)/2$ and so

$$
\begin{aligned}
T &= 2\frac{(n-1)(n)(2n-1)}{6} + 3 \times \frac{n(n-1)}{2} \\
&= .... \\
&= n(n-1)\left(\frac{2n}{3} + \frac{7}{6}\right) \qquad (1)
\end{aligned}
$$

What is the operation count (leading term only) for solving the linear system $Ax = b$ with Gaussian elimination without pivoting?
$\frac{1}{2} * n^3$

What happens when partial pivoting is used?

Gauss-Jordan Elimination

✍8 Operation count for Gauss-Jordan. Order of the cost? How does it compare with Gaussian Elimination?

Solution: From the notes:

$$\begin{aligned}
T &= \sum_{k=1}^{n-1} \sum_{i=1}^{n-1} [2(n-k)+3)] = \sum_{k=1}^{n-1} (n-1)[2(n-k)+3] \\
&= (n-1) \sum_{j=1}^{n-1} [2j+3] \\
&= (n-1)\left[n(n-1)+3(n-1)\right] \\
&= (n-1)^2(n+3) = (n-1)^3 + 4(n-1)^2
\end{aligned}$$

The bottom line is that the cost is $\approx n^3$ which is 50% more expensive than GE. This additional cost is not worth it in spite of the simplicity

of the algorithm. For this Gauss-Jordan is seldom used in practice. $\square$

LU decomposition

$\frac{2}{3} * n^3$.

An easy transformation ($j = n - i + 1$) easily shows that

$$\begin{aligned}
\boxed{\sum_{i=1}^{n} 2(n-i)(n-i+1)} &= 2\sum_{j=0}^{n-1} j(j+1) \\
&= 2\sum_{j=0}^{n-1} (j^2 + j) \\
&= 2\left(\frac{1}{3}n^3 - \frac{1}{3}n\right)
\end{aligned}$$

Then, since we are interested only in the asymptotic behaviour, we drop the $\frac{2}{3}n$ part, and what's left is $\boxed{\frac{2}{3}n^3.}$

Use LU to solve linear sys, cost

✍4 Practical use: Show how to use the LU factorization to solve linear systems with the same matrix $\boldsymbol{A}$ and different $\boldsymbol{b}$'s.

Solution: If we have the LU factorization $\boldsymbol{A} = \boldsymbol{L}\boldsymbol{U}$ available then we can solve the linear system $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ by writing it as

$$\boldsymbol{L}\underbrace{(\boldsymbol{U}\boldsymbol{x})}_{\boldsymbol{y}} = \boldsymbol{b}$$

So we solve for $\boldsymbol{y}$: $\boldsymbol{L}\boldsymbol{y} = \boldsymbol{b}$ then once $\boldsymbol{y}$ is computed we solve for $\boldsymbol{x}$: $\boldsymbol{U}\boldsymbol{x} = \boldsymbol{y}$. This involves two triangular solves at the cost of $\boldsymbol{n}^2$ each instead of the $\boldsymbol{O}(\boldsymbol{n}^3)$ cost of redoing everything with Gaussian elimination.□

Cholesky decomposition

Gram-Schmidt

✍3 Cost of Gram-Schmidt?

Solution: Step j of the algorithm costs : $(j-1) \times 2m$ operations for line 3, + $(j-1) \times 2m$ operations for loop in line 4 + $3m$ operations in Lines 7 and 8 together. Total for step j = $c_j = (4j-1)m$. Total over the n columns = $T(n) = (2n^2 + n)m \approx \boxed{2n^2m.}$

Note: this is linear in m (number of rows) and quadratic in n (number of columns).

QR decomposition

✍5 What is the cost of solving a linear system with the QR factorization?

Solution: According to the previous question we have a cost of $2n^3$ for the factorization (since $m = n$), to which we need to add the cost of solving a triangular solve $O(n^2)$ and the cost for computing $Q^T b$ which is again $O(n^2)$. In the end the cost is dominated by the QR factorization which is $\boxed{2n^3.}$ This is 3 times more expensive than GE.

Comparison

1, Gauss-Jordan is 50 percent more expensive than GE. This additional cost is not worth it in spite of the simplicity of the algorithm. For this Gauss-Jordan is seldom used in practice. If gauss takes 60 secs to complete then gauss-jordan needs $60 * (1 + 0.5) = 60 + 30 = 90$ secs to complete.

2, True or false: "Computing the LU factorization of matrix A involves more arithmetic operations than solving the system Ax = b by Gaussian."
False. The number of arithmetic operations of LU and Gauss is identical. (just LU involves additional memory to store the factors - but these are not floating point operations).

3, QR is 3 times more expensive than GE(LU decomposition). If gauss takes 60 secs to complete then gauss-jordan needs $3 * 60 = 180$ secs to complete.