Sept 16, 2021

How much computation work does it take to make the LU factorization?

Count the loops...

```
rows,cols=size(A)

U=copy(A)
L=Matrix{Float16}(I,rows,cols)

for j=1:cols-1
    for i=j+1:rows
        alpha=U[i,j]/U[j,j]
        println("r$i <- r$i - $alpha * r$j")
        U[i,:]=U[i,:]-alpha*U[j,:]
        U[i,j]=0.0
        L[i,j]=alpha
    end
end
```

Suppose $A \in \mathbb{R}^{n \times n}$,

Then rows $= n$ and cols $= n$

The for j loop runs about n times

The for i loop runs about n times

Each row operation involves rows with n elements

The run time is about $n \times n \times n = O(n^3)$

But wait! the second loop starts at $j+1$ so it doesn't really perform n loops each time!

Summation formula...

$$S_n = 1+2+3+\dots+n$$

EXAMPLE

$$S_7 = 1+2+3+4+5+6+7$$
$$S_7 = 7+6+5+4+3+2+1$$
$$2S_7 = \underbrace{8+8+8+8+8+8+8}$$
$$2S_7 = 7 \times 8$$
$$S_7 = \frac{7 \times 8}{2}$$

In general: $S_n = \frac{n(n+1)}{2} = O(n^2)$

Therefore even the the second loop starts at $j+1$ the order of the number of loops is still $O(n^3)$

Conclusion: Making the LU factorization takes $O(n^3)$ number of operations...

What do we do with the LU factorization?

Idea: Solve $Ax=b$. How factor $A=LU$

$$\underbrace{LU}x=b$$

Substitute $y=Ux$

Get System: $\begin{cases} Ly=b \\ Ux=y \end{cases}$

$$y=\begin{bmatrix} y_1 \\ \vdots \\ y_5 \end{bmatrix}, \quad b=\begin{bmatrix} b_1 \\ \vdots \\ b_5 \end{bmatrix}$$

How to solve $Ly=b$... (similarly $Ux=y$)

$$L=\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 2 & 1 & 0 & 0 & 0 \\ 4 & 5 & 1 & 0 & 0 \\ 7 & 8 & 9 & 1 & 0 \\ 1 & 2 & 3 & 4 & 1 \end{bmatrix} \Rightarrow \begin{aligned} & y_1 = b_1 \\ & 2y_1 + y_2 = b_2 \\ & 4y_1 + 5y_2 + y_3 = b_3 \\ & 7y_1 + 8y_2 + 9y_3 + y_4 = b_4 \\ & y_1 + 2y_2 + 3y_3 + 4y_4 + y_5 = b_5 \end{aligned}$$

Does not take $O(n^3)$ operations to solve, but how many?

$y_1 = b_1$ — no computation

$y_2 = b_2 - 2y_1$ — 1 subtraction 1 multiplies

$y_3 = b_3 - 4y_1 - 5y_2$ — 2 subtractions 2 multiplies

$\vdots$

$y_5 = b_5 - 1y_1 - 2y_2 - 3y_3 - 4y_4$ — 4 subtraction, 4 multiplication

Total $1+2+3+4$ sub & mult.

In general $O(n^2)$ number of operations

What does $O(n^2)$ mean?

<u>Definition</u>: We say $f_n = O(n^2)$ if there is a constant M such that $|f_n| \le M n^2$ for all n sufficiently large...

Total work for solving $Ly=b$ for the $n \times n$ case is

$$T_n = 1+2+\cdots+(n-1) = \frac{(n-1)(n-1+1)}{2} = \frac{n(n-1)}{2}$$

$$= \frac{n^2-n}{2} = \frac{1}{2}n^2 - \frac{n}{2} \le \frac{1}{2}n^2$$

That means there is a constant M so $|T_n| \le M n^2$ for n large.

In fact $M = \frac{1}{2}$ works and any $n \ge 2$ is fine.

Therefore $\boxed{T_n = O(n^2)}$ ⇐ Just the notation...

In Summary:

- It takes $O(n^3)$ operations to factor $A = LU$.
- But only $O(n^2)$ operations to solve $Ly = b$ and $Ux = y$

When I was editing the notes from last time I noticed a surprising amount of difference between LU and the original matrix A.

```
julia> L*U
4×4 Matrix{Float64}:
 0.459076  0.201025  0.192312  0.464086
 0.783656  0.544129  0.45053   0.0603414
 0.12721   0.916119  0.27272   0.535355
 0.432177  0.22033   0.315932  0.373066

julia> A
4×4 Matrix{Float64}:
 0.459076  0.201025  0.192312  0.464086
 0.783558  0.544086  0.450489  0.0602417
 0.127198  0.916158  0.272742  0.535181
 0.432221  0.220349  0.315929  0.373361
```

Looks good...sort of...

wait! to only 3 significant digits???

rounding error!

Matrix norm: How to compare two matrices and measure the degree to which they are different.

Question what is $\|A\|$?

measure of the size of A

Question what is $\| A - LU \|$?

measure of the error in the LU factorization...

Engineers make it simple:

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ \vdots & & \ddots & \vdots \\ a_{m1} & \cdots & \cdots & a_{mn} \end{bmatrix} \in \mathbb{R}^{m \times n}$$

Aside: ① What is the determinant.

Back to matrix norm

I moved this discussion of determinants to the end of the lecture notes, so it doesn't break the flow...

Engineering Idea for Matrix norm (also called the Frobenius norm)

$$\|A\|_F = \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{n} |a_{ij}|^2}$$

Write an F subscript to keep track of which norm is which

Treats a matrix and the entries in it as one big vector written is a weird way...

This is simple and has of physics but it lack some mathematical optimality...

Here is an example, that we didn't do in class for this engineering norm...

EXAMPLE

norm of the identity matrix ...

$$A = I = \begin{bmatrix} 1 & & & & 0 \\ & 1 & & & \\ & & 1 & & \\ & & & \ddots & \\ 0 & & & & 1 \end{bmatrix} \in \mathbb{R}^{n \times n}$$

add up sum of the squares at the diagonal.

$$\|A\|_F = \sqrt{\sum_{i=1}^{n} \sum_{j=1}^{n} |a_{ij}|^2} = \sqrt{n}$$

That's easy! But what sort of mathematical optimality does the Frobenius norm lack?

From a mathematical point of view ... all we want to do is an estimate like this

$$\|Ax\| \leq \|A\| \, \|x\|$$

vector norm

matrix norm..

vector norm

Idea: define $\|A\|$ to be the smallest number such that this inequality holds for all vectors x...

$$\|A\| = \max \left\{ \frac{\|Ax\|}{\|x\|} : x \neq 0 \right\}.$$

$$= \max \{ \|Ax\| : \|x\| = 1 \}.$$

to compute this, find the largest eigenvalue of the matrix A^TA.

Why the Engineering matrix norm is not optimal....

$$\|Ax\| \leq \|A\| \, \|x\|$$

What if $A = I$, $\|x\| = \|Ix\| \leq \|I\| \, \|x\|$

which means the optimal value of $\|I\| = 1$. ideally the norm of the identity should always be 1

But as computed in the example added earlier $\|I\|_F = \sqrt{n} > 1$

Frobenius

This is the discussion about determinants we had in the middle of class.

Since a determinant is not a type of matrix norm...I've put this at the end of the lecture notes, for reference...

Aside: (1) What is the determinant.

(2) How do you compute it.

(3) Is it the product of the eigenvalues.

(2) $\det(A) = \det(LU) = \det(L)\det(U)$

$$L = \begin{bmatrix} 1 & 0 & 0 & 0 \\ * & 1 & 0 & 0 \\ * & * & 1 & 0 \\ * & * & * & 1 \end{bmatrix} \qquad \det(L) = 1$$

$$U = \begin{bmatrix} 7 & * & * & * \\ 0 & -2 & * & * \\ 0 & 0 & 3 & * \\ 0 & 0 & 0 & 4 \end{bmatrix} \qquad \det(U) = 7(-2)(3)(4)$$

(3) Eigenvalue decomposition $A = SDS^{-1}$

$$\det(A) = \det(SDS^{-1}) = \det(S)\det(D)\det(S^{-1})$$

$$= \cancel{\det(S)}\det(D)\frac{1}{\cancel{\det S}} = \det(D).$$