Klöckner

Lecture 7   Numerical Analysis   2/13/18

Last time:

Intro to Numerical Linear Algebra:

- Computational costs of direct methods:

  Mat-vec: $O(n^2)$

  Mat-mat: $O(n^3)$

  Solve $A\vec{x} = b$ via Gaussian Elim: $O(n^3)$

- Pivoting in Gaussian Elim.

  - Row vs. column:
  - To insure that the algorithm doesn't divide by zero.

Today: LU factorization & pivoting

Another Way to think of Gaussian Elimination : LU factorization:

- Each row operation corresponds to multiplication by a lower triangular matrix.

Ex: $A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 10 \end{pmatrix}$

$$\underbrace{\begin{pmatrix} 1 & 0 & 0 \\ -4 & 1 & 0 \\ -7 & 0 & 1 \end{pmatrix}}_{L_1} \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix} = \begin{pmatrix} 1 & 2 & 3 \\ 0 & -3 & -6 \\ 0 & -6 & -21 \end{pmatrix}$$

$$\underbrace{\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -2 & 1 \end{pmatrix}}_{L_2} \begin{pmatrix} 1 & 2 & 3 \\ 0 & -3 & -6 \\ 0 & -6 & -21 \end{pmatrix} = \underbrace{\begin{pmatrix} 1 & 2 & 3 \\ 0 & -3 & -6 \\ 0 & 0 & -9 \end{pmatrix}}_{U.}$$

So $L_2 L_1 A = U$

$A = (L_2 L_1)^{-1} U$

$= \underbrace{L_1^{-1} L_2^{-1}}\, U$

These "undo" the row operations in L_1, L_2

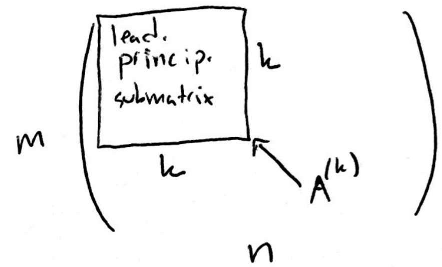
$L_1^{-1} = \begin{pmatrix} 1 & 0 & 0 \\ 4 & 1 & 0 \\ 7 & 0 & 1 \end{pmatrix}$

$L_2^{-1} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 2 & 1 \end{pmatrix}$

$L_1^{-1} L_2^{-1} = \begin{pmatrix} 1 & 0 & 0 \\ 4 & 1 & 0 \\ 7 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -2 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 4 & 1 & 0 \\ 7 & 2 & 1 \end{pmatrix}$

Existence of the LU factorization

Def: The leading principal submatrix of order k is the $k \times k$ contiguous submatrix of A whose $(1,1)$ element ~~is~~ corresponds to the $(1,1)$ element of A:

$A =$ 

Thm: If $A^{(k)}$ is invertible for all $k = 2, \ldots, n-1$, then $A = LU$ where L is lower-triangular with 1's on the diagonal and U is upper-triangular.

Proof: By induction.

Case: $n = 2$. $A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$, $L = \begin{pmatrix} 1 & 0 \\ m & 1 \end{pmatrix}$

$U = \begin{pmatrix} u & v \\ 0 & w \end{pmatrix}$

then $A = \begin{pmatrix} A^{(k)} & \vec{b} \\ \vec{c}^T & d \end{pmatrix}$

$L = \begin{pmatrix} L^{(k)} & 0 \\ \vec{m}^T & 1 \end{pmatrix}$ $\quad U = \begin{pmatrix} U^{(k)} & \vec{v} \\ 0 & w \end{pmatrix}$

Gaussian Elimination & LU factorization are equivalent.

Easy to alter GE code to save LU (exercise):

Then to solve $A\vec{x}=\vec{b}$, $(LU\vec{x}=\vec{b})$

(1) solve $L\vec{y}=\vec{b}$ ← Forward substitution

(2) solve $U\vec{x}=\vec{y}$ ← Backward substitution

Operation Count? (For the L solve)

$$\begin{pmatrix} l_{11} & & & \\ l_{21} & l_{22} & & \\ \vdots & & \ddots & \\ & & & l_{nn} \end{pmatrix} \vec{y} = \vec{b}$$

$y_1 = b_1 / l_{11}$ ← are all 1's in our case

$y_2 = (b_2 - l_{21} y_1) / l_{22}$

$\vdots$

```
do i = 1, n          <- not included in op. count.
  y_i = b_i
  do j = 1, i-1
     y_i = y_i - l_ij * y_j
  end do
end do
```

$$\text{Operat: } \sum_{i=1}^{n} \sum_{j=1}^{i-1} 2 = 2 \sum_{i=1}^{n} (i-1)$$

$$= 2 \sum_{1}^{n-1} k = 2 \cdot \frac{n(n-1)}{2} = n^2 + O(n)$$

This is <u>fast</u>

Pivoting

This algorithm for $A = LU$ may fail if a pivot is equal to 0, or if it is very small.

Thm For any $A \in \mathbb{R}^{n \times n}$, $\exists$ P, L, U s.t. $PA = LU$.

Solution: Interchange rows.

Ex: $A = \begin{pmatrix} 1 & 2 & 2 \\ 2 & 4 & 4 \\ 4 & 8 & 3 \end{pmatrix}$ is obviously singular

Ex: What goes wrong with

$$A: \begin{pmatrix} 10^{-20} & 1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$$

Exact solution: $\vec{x} \approx \begin{pmatrix} 1 \\ 1 \end{pmatrix}$ $\quad x_1 = 1 + \frac{1}{10^{20}-1}$, $\quad x_2 = 1 - \frac{1}{10^{20}-1}$

With our Gaussian elimination procedure: (in double prec. arith.)

$$\left(\begin{array}{cc|c} 10^{-20} & 1 & 1 \\ 1 & 1 & 2 \end{array}\right) \sim \left(\begin{array}{cc|c} 10^{-20} & 1 & 1 \\ 0 & -10^{20} & -10^{20} \end{array}\right)$$

$\Rightarrow$ $x_2 = 1$

$x_1 = \frac{1 - 1}{10^{-20}} = 0$ $\quad$ (or maybe $\frac{\epsilon}{10^{-20}} \sim 10^4$)

$\boxed{\text{Program this}}$

The problem is completely avoided by interchanging:

$$\left(\begin{array}{cc|c} 1 & 1 & 2 \\ 10^{-20} & 1 & 1 \end{array}\right) \sim \left(\begin{array}{cc|c} 1 & 1 & 2 \\ 0 & 1 & 1 \end{array}\right)$$

Interchanging with largest element in column: Partial Pivoting

Op. Count. - no cost for pivoting

- $O(n^2)$ cost for searching.

Full pivoting search column and row (requires re-ordering of RHS as well)

In this case (part. piv.): $A = PLU$

P: permutation matrix ~~swapping~~ swapping rows.

Special Case Symmetric Positive Definite matrices

No pivoting is necessary → one of the earliest results in the field of numerical ~~anal~~ analysis.

At the time: Solve $A^* A \vec{x} = A^* \vec{b}$ instead (not a good idea).

~~8/19~~

~~8/19~~ 7-6

If A is . sym. pos. def. ~~then~~ then

$$A = LU$$

$$= A^t = U^t L^t$$

Can we make $L = U^t$, $L^t = U$? Yes. Scale things differently.

&

<u>Cholesky Decomposition</u>: $A = L^t L$ for A s.p.d.

$$L = \begin{pmatrix} l_{11} & & & \\ l_{21} & l_{22} & & \\ \vdots & \vdots & \ddots & \\ \vdots & & & l_{nn} \end{pmatrix}$$

$$L L^t = \begin{pmatrix} l_{11} & & \\ l_{21} & l_{22} & \\ \vdots & & \ddots \end{pmatrix} \begin{pmatrix} l_{11} & l_{21} & \cdots \\ 0 & l_{22} & \\ & & \ddots \end{pmatrix}$$

$$= \begin{pmatrix} l_{11} & & \\ l_{21} l_{11} & \ddots & \\ l_{31} l_{11} & & \end{pmatrix}$$

$$\Rightarrow l_{11} = a_{11}$$

$$l_{21} = a_{21}/l_{11}$$

$$\vdots$$