ECE 4960

Spring 2017

Lecture 9

LU Decomposition
(Also Relation to Gaussian Elimination)

Edwin C. Kan
School of Electrical and Computer Engineering
Cornell University

Gaussian Elimination vs. LU Decomposition

Gaussian elimination	LU decomposition
$4x_1 + 4x_2 + 2x_3 = 2$ $4x_1 + 5x_2 + 3x_3 = 3$ $2x_1 + 3x_2 + 3x_3 = 5$	$A = \begin{pmatrix} 4 & 4 & 2 \\ 4 & 5 & 3 \\ 2 & 3 & 3 \end{pmatrix}$; $b = \begin{pmatrix} 2 \\ 3 \\ 5 \end{pmatrix}$
Eliminate x_1 from 2nd and 3rd rows (a_{11} is called the **pivot**): $4x_1 + 4x_2 + 2x_3 = 2$ $x_2 + x_3 = 1$ $x_2 + 2x_3 = 4$	$M_1 = \begin{pmatrix} 1 & 0 & 0 \\ -1 & 1 & 0 \\ -0.5 & 0 & 1 \end{pmatrix}$ $M_1A = \begin{pmatrix} 4 & 4 & 2 \\ 0 & 1 & 1 \\ 0 & 1 & 2 \end{pmatrix}$ $M_1b = \begin{pmatrix} 2 \\ 1 \\ 4 \end{pmatrix}$
Eliminate x_2 from 3rd row (a_{22} is chosen as the pivot): $4x_1 + 4x_2 + 2x_3 = 2$ $x_2 + x_3 = 1$ $x_3 = 3$	$M_2 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -1 & 1 \end{pmatrix}$ $M_2M_1A = \begin{pmatrix} 4 & 4 & 2 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix}$ $M_2M_1b = \begin{pmatrix} 2 \\ 1 \\ 3 \end{pmatrix}$

$M_2M_1Ax = M_2M_1b$; $A = LU$

$U = M_2M_1A$ is an upper triangular matrix

M_1, M_2 and $L = M_1^{-1}M_2^{-1}$ are lower triangular matrices

Triangular Matrix Properties

- All upper U and lower L triangular matrices with **unit diagonals** have product and inversion rules,
  - Product of upper triangular matrices will be an upper triangular matrix;
  - Inverse of an upper triangular matrix will be an upper triangular matrix.
- The triangular matrix can be readily solved by backward substitution with $O(n^2)$ computational cost.

Symbolic LU Factorization

- If we just know the symbolic position of the non-zero elements, we can still perform LU factorization, i.e., we will know the structure of the M_i and $M_iM_{i-1}A$ matrices without their element values.
- The computing sequence and the movement/fill-in of the LU factorization will be entirely known.
- Actual element values will affect the results (for sure), and the precision (choice of the pivot element, or **pivoting**)
- In circuit simulation, once the circuit topology is fixed, this sparse structure will not change, and all following DC and transient simulation can share the first *symbolic LU factorization*.
- For finite-difference and finite-element simulation, once the gridding and discretization of the geometry are fixed, we can perform one symbolic LU factorization and use it to organize simulation in all following stages.

First Objective: Minimal Fill-ins

- The sparse matrix structure can change during the Gaussian elimination, or equivalently, LU decomposition.
- For both memory and computing concerns, we hope to keep these "fill-ins" to a minimal by reordering the rows or columns (i.e., row or column permutation).

$$\begin{pmatrix} X & X & X & 0 \\ X & X & 0 & 0 \\ 0 & 0 & X & X \\ X & 0 & 0 & X \end{pmatrix} \Rightarrow \begin{pmatrix} X & X & X & 0 \\ 0 & X & \otimes & 0 \\ 0 & 0 & X & X \\ 0 & \otimes & \otimes & X \end{pmatrix}$$

Three fill-ins

$$\begin{pmatrix} X & X & 0 & 0 \\ X & X & X & 0 \\ 0 & 0 & X & X \\ X & 0 & 0 & X \end{pmatrix} \Rightarrow \begin{pmatrix} X & X & 0 & 0 \\ 0 & X & X & 0 \\ 0 & 0 & X & X \\ 0 & \otimes & 0 & X \end{pmatrix}$$

One fill-in

Row and Column Permutation

- Row permutation is the change of the order of the objective functions, which does not change the property of *A* except that the determinant is multiplied by –1.
- Row permutation does change the **sparse structure** significantly (but not the total number of nonzero elements) and can affect the number of fill-ins for subsequent operations and hence the memory usage and total computational time.
- Similarly, column permutation can tidy up the matrix so that the pivot (the element chosen to zero out the lower elements in the same column) is always on the diagonal. We just need to keep the solution vector indices correspondingly.

$Ax = b$ Solution from LU Factorization

$$A = LU;\ Ax = b$$

$$LUx = b;$$

$$Ux = L^{-1}b \text{ or } x = U^{-1}L^{-1}b$$

1. **Symbolic LU factorization** to determine the order of computing for minimal fill-in and best anticipated pivoting (described in the next section): O(n^3)
2. **Numerical calculation of *L* and *U* matrices**: O(nz), where nz is the total number of nonzero elements during the LU factorization.
3. **Backward substitution** (one for *L* and one for *U*) to obtain the solution: O($n \times nz$).

LU Factorization Achieves…

- Make $Ux = v$. We can solve $Lv = b$ and $Ux = v$ by backward substitution in $O(n^2)$ time.
- L^{-1} and U^{-1} are also triangular matrices, and can be readily computed IF necessary (they may have a lot of fill-ins, so this is not typically done in view of memory usage).
- The **eigenvalues** of L and U are their respective diagonals (hence the spectral radii of L and U, which are the largest magnitude of the diagonals, are known).
- The matrix determinant of L and U can be readily determined:

$$Det(L) = \prod_{i=1}^{n} L_{ii}; \qquad Det(U) = \prod_{i=1}^{n} U_{ii}$$

Choices in LU Factorization

- L and U ($n^2 + n$) have more degree of freedoms than A (n^2)
- Further constraint is needed to obtain specific L and U .

1. L has a unit diagonal: **Doolittle** LU factorization.
2. U has a unit diagonal: **Crout** LU factorization.
3. constrain $L_{ii} = U_{ii}$: **Choleski** LU factorization.

Equivalent Criteria for $Ax = b$ Solution Existence

- A has a rull rank of n.
- A can be inverted (A^{-1} exists).
- A is non-singular or non-degenerate.
- A can have successful LU factorization with no zero eigenvalues in L and U (precision will be subject to the matrix condition number).
- The matrix determinant is not zero: Det(A) $\neq 0$.
- The eigenvalues of A are not zero (but they can have the repeated nonzero elements), and A can be diagonalized into the Jordon block structure..

Hacker Practice

Use the full matrix format and only row permutation (easier to implement in a short time, and a good check for your sparse matrixlater on), perform the minimal fill-in algorithm for choosing the sequence of pivoting to solve:

$$Ax = b; \quad \begin{pmatrix} 1 & 2 & 0 & 0 & 3 \\ 4 & 5 & 6 & 0 & 0 \\ 0 & 7 & 8 & 0 & 9 \\ 0 & 0 & 0 & 10 & 0 \\ 11 & 0 & 0 & 0 & 12 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{pmatrix} = \begin{pmatrix} 5 \\ 4 \\ 3 \\ 2 \\ 1 \end{pmatrix}$$

Use the backward substitution to check your answer. For an interesting process, change your data structure to row-compressed format. However, your choices of pivoting should be the same, and your LU matrix identical.