LECTURE 18

DIRECT SOLUTIONS TO LINEAR ALGEBRAIC SYSTEMS - CONTINUED

Ill-conditioning of Matrices

- There is no clear cut or precise definition of an ill-conditioned matrix.

Effects of ill-conditioning

- Roundoff error accrues in the calculations
- Can potentially result in very inaccurate solutions
- Small variation in matrix coefficients causes large variations in the solution

Detection of ill-conditioning in a matrix

- An inaccurate solution for X can satisfy an ill-conditioned matrix quite well!
  - Apply back substitution to check for ill-conditioning
    - Solve $\mathbf{A}X = B$ through Gauss or other direct method $\rightarrow X_{poor}$
    - Back substitute $\mathbf{A}X_{poor} \rightarrow B_{poor}$
    - Comparing we find that $B_{poor} \approx B$

- Back substitution is ***not*** a good detection technique.
- The effects of ill-conditioning are very subtle!

- Examine the inverse of matrix $\mathbf{A}$
  - If there are elements of $\mathbf{A}^{-1}$ which are many orders of magnitude larger than the original matrix, $\mathbf{A}$, then $\mathbf{A}$ is probably ill-conditioned
  - It is always best to normalize the rows of the original matrix such that the maximum magnitude is of order 1
  - Evaluate $\mathbf{A}^{-1}$ using the same method with which you are solving the system of equations. Now compute $\mathbf{A}^{-1}\mathbf{A}$ and compare the results to $\mathbf{I}$. If there's a significant deviation, then the presence of serious roundoff exists!
  - Compute $(\mathbf{A}^{-1})^{-1}$ using the same method with which you are solving the system of equations. This is a more severe test of roundoff since it is accumulated both in the original inversion and the re-inversion.

- Can also evaluate ill-conditioning by examining the ***normalized determinant***. The matrix may be ill-conditioned when:

$$\frac{det\mathbf{A}}{\sqrt{\sum_{i=1}^{N}\sum_{j=1}^{N} a_{ij}^2}} << 1$$

where

$$\text{Euclidean Norm of } \mathbf{A} \equiv \sqrt{\sum_{i=1}^{N}\sum_{i=1}^{N} a_{ij}^2}$$

- If the matrix $\mathbf{A}$ is ***diagonally dominant***, i.e. the absolute values of the diagonal terms $\geq$ the sum of the off-diagonal terms for each row, then the matrix is not ill-conditioned

$$|a_{ii}| \geq \sum_{\substack{j=1 \\ i \neq i}}^{N} |a_{ij}| \qquad i = 1, 2, \ldots, N$$

- Effects of ill-conditioning are most serious in large dense matrices (e.g. especially those obtained in such problems as curve fitting by least squares)
- Sparse banded matrices which result from Finite Difference and Finite Element methods are typically much better conditioned (i.e. can solve fairly large sets of equations without excessive roundoff error problems)
- Ways to overcome ill-conditioning
    - Make sure you pivot!
    - Use large word size (use double precision)
    - Can use error correction schemes to improve the accuracy of the answers
    - Use iterative methods

Factor Method (Cholesky Method)

- Problem with Gauss elimination
  - Right hand side "load" vector, $\boldsymbol{B}$, must be available at the time of matrix triangulation
  - If $\boldsymbol{B}$ is not available during the triangulation process, the entire triangulation process must be repeated!
  - Procedure is not well suited for solving problems in which $\boldsymbol{B}$ changes

$$\begin{array}{lcl} \mathbf{A}\boldsymbol{X} = \boldsymbol{B}_1 & \Rightarrow & O(N^3) + O(N^2) \text{ steps} \\ \mathbf{A}\boldsymbol{X} = \boldsymbol{B}_2 & \Rightarrow & O(N^3) + O(N^2) \text{ steps} \\ \vdots & & \vdots \\ \mathbf{A}\boldsymbol{X} = \boldsymbol{B}_R & \Rightarrow & O(N^3) + O(N^2) \text{ steps} \end{array}$$

  - Using Gauss elimination, $O(N^3 R)$ operations, where N = size of the system of equation and R = the number of different load vectors which must be solved for

- Concept of the factor method is to facilitate the solution of multiple right hand sides without having to go through a re-triangulation process for each $\boldsymbol{B}_r$

Factorization step

- Given $\mathbf{A}$, find $\mathbf{P}$ and $\mathbf{Q}$ such that

$$\mathbf{A} = \mathbf{PQ}$$

where $\mathbf{A}$, $\mathbf{P}$ and $\mathbf{Q}$ are all $N \times N$ matrices

- We note that $|\mathbf{A}| \neq 0 \Rightarrow |\mathbf{P}||\mathbf{Q}| \neq 0$ and therefore neither $\mathbf{P}$ nor $\mathbf{Q}$ can be singular

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = \begin{bmatrix} p_{11} & p_{12} & p_{13} \\ p_{21} & p_{22} & p_{23} \\ p_{31} & p_{32} & p_{33} \end{bmatrix} \bullet \begin{bmatrix} q_{11} & q_{12} & q_{13} \\ q_{21} & q_{22} & q_{23} \\ q_{31} & q_{32} & q_{33} \end{bmatrix}$$

- We can only have N^2 unknowns!
    - Define $\mathbf{P}$ as lower triangular
    - Define $\mathbf{Q}$ as upper triangular

- Now we have $N^2 + N$ unknowns

- Reduce the number of unknowns by selecting either
  - $p_{ii} = 1 \qquad i = 1, N \quad \Rightarrow$ Doolittle Method
  - $q_{ii} = 1 \qquad i = 1, N \quad \Rightarrow$ Crout Method

- Now we only have N^2 unknowns! We can solve for all unknown elements of **P** and **Q** by proceeding from left to right and top to bottom

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ p_{21} & 1 & 0 \\ p_{31} & p_{32} & 1 \end{bmatrix} \bullet \begin{bmatrix} q_{11} & q_{12} & q_{13} \\ 0 & q_{22} & q_{23} \\ 0 & 0 & q_{33} \end{bmatrix}$$

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = \begin{bmatrix} q_{11} & q_{12} & q_{13} \\ p_{21}q_{11} & p_{21}q_{12} + q_{22} & p_{21}q_{13} + q_{23} \\ p_{31}q_{11} & p_{31}q_{12} + p_{32}q_{22} & p_{31}q_{13} + p_{32}q_{23} + q_{33} \end{bmatrix}$$

- Factorization proceeds from left to right and then top to bottom as:
  - Red → current unknown being solved
  - Blue → unknown value ***already*** solved

$$a_{11} = q_{11}$$

$$a_{12} = q_{12}$$

$$a_{13} = q_{13}$$

$$a_{21} = p_{21}q_{11}$$

$$a_{22} = p_{21}q_{12} + q_{22}$$

$$a_{23} = p_{21}q_{13} + q_{23}$$

$$a_{31} = p_{31}q_{11}$$

$$a_{32} = p_{31}q_{12} + p_{32}q_{22}$$

$$a_{33} = p_{31}q_{13} + p_{32}q_{23} + q_{33}$$

- We can compute the elements of $\mathbf{P}$ and $\mathbf{Q}$ and store them directly into the old $\mathbf{A}$ matrix as the procedure progresses

$$\mathbf{A} = \mathbf{P}\mathbf{Q}$$

where

$\mathbf{P}$ = lower triangular matrix $\rightarrow$ rename $\mathbf{L}$

$\mathbf{Q}$ = upper triangular matrix $\rightarrow$ rename $\mathbf{U}$

- Therefore renaming these matrices

$$\mathbf{P} \rightarrow \mathbf{L}$$

$$\mathbf{Q} \rightarrow \mathbf{U}$$

we note that $\mathbf{A}$ has been factored as

$$\mathbf{A} = \mathbf{L}\mathbf{U}$$

- Now considering the equation to be solved

$$\mathbf{A}X = B$$

- However $\mathbf{A} = \mathbf{L}\mathbf{U}$ where $\mathbf{L}$ and $\mathbf{U}$ are known

$$(\mathbf{L}\mathbf{U})X = B$$

<u>Forward/backward substitution procedures to obtain a solution</u>

- Changing the order in which the product is formed

$$\mathbf{L}(\mathbf{U}X) = B$$

- Now let

$$Y = \mathbf{U}X$$

- Hence we have two systems of simultaneous equations

$$\mathbf{L}Y = B$$

$$\mathbf{U}X = Y$$

- Apply a forward substitution sweep to solve for $\boldsymbol{Y}$ for the system of equations

$$\mathbf{L}\boldsymbol{Y} = \boldsymbol{B}$$

- Apply a backward substitution sweep to solve for $\boldsymbol{X}$ for the system of equations

$$\mathbf{U}\boldsymbol{X} = \boldsymbol{Y}$$

Notes on Factorization Methods

- Procedure
  - Perform the factorization by solving for $\mathbf{L}$ and $\mathbf{U}$
  - Perform the sequential forward and backward substitution procedures to solve for $\boldsymbol{Y}$ and $\boldsymbol{X}$
- The factor method is very similar to Gauss elimination although the order in which the operations are carried out is somewhat different.
- Number of operations
  - $O(N^3)$ for $\mathbf{LU}$ decomposition (same as triangulation for Gauss)
  - $O(N^2)$ for forward/backward substitution (same as backward sweep for Gauss)

Advantages of* LU *factorization over Gauss Elimination

- Can solve for any load vector $\boldsymbol{B}$ at any time with $O(N^2)$ operations (other than triangulation which is done ***only once*** with $O(N^3)$ operations)
- Generally has somewhat smaller roundoff error

Example comparing costs

- If we are solving R systems of $N \times N$ equations in which the matrix $\mathbf{A}$ stays the same and only the vector $\boldsymbol{B}$ changes, compare the overall costs for Gauss elimination and **LU** factorization

- Gauss Elimination costs

$$\text{Triangulation Cost} = R\,[O(N^3)]$$

$$\text{Back Substitution Cost} = R[O(N^2)]$$

$$\text{Total Cost} = R[O(N^3) + O(N^2)]$$

$$\text{Total Cost for Large } N \cong R\;O(N^3)$$

- **LU** factorization costs

$$\mathbf{LU} \text{ Factorization Cost} = [O(N^3)]$$

$$\text{Back/Forward Substitution Cost} = R[O(N^2)]$$

$$\text{Total Cost} = [O(N^3) + R\ O(N^2)]$$

$$\text{Total Cost for } R \gg N \cong R\ O(N^2)$$

- Considering some typical values for N and R

N	R	Gauss Elim.	LU Factorization	Ratio of Costs
N	R	$O(RN^3)$	$O(RN^2)$	$O(N)$
1,000	5,000	5×10^{12}	5×10^{9}	1,000
10^5	10^6	10^{21}	10^{16}	100,000

- We can also implement **LU** factorization (decomposition) in banded mode and the savings compared to banded Gauss elimination would be $O(M)$ (where M = band-width)

Other Factorization Methods for Symmetrical Matrices

- Solve $\mathbf{A}\boldsymbol{X} = \boldsymbol{B}$ assuming that $\mathbf{A}$ is symmetrical, i.e

$$\mathbf{A}^T = \mathbf{A} \quad \text{or} \quad a_{ij} = a_{ji}$$

Cholesky Square Root Method

- Requires that $\mathbf{A}$ is symetrical ($\mathbf{A}^T = \mathbf{A}$) and positive definite ($\mathbf{U}^T\mathbf{A}\mathbf{U} > c$ where $\mathbf{U}$ = any vector and c is a positive number.

- First step is to decompose the matrix $\mathbf{A}$

$$\mathbf{A} = \mathbf{L}\mathbf{L}^T \quad \text{or} \quad \mathbf{A} = \mathbf{U}^T\mathbf{U}$$

- Diagonal terms on $\mathbf{L}$ or $\mathbf{U}$ don't equal unity

- We note that **A** contains 6 independent entries and that **L** contains 6 unknowns and therefore we can perform the factorization

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \qquad \mathbf{L} = \begin{bmatrix} l_{11} & 0 & 0 \\ l_{21} & l_{22} & 0 \\ l_{31} & l_{32} & l_{33} \end{bmatrix}$$

- Substituting the factored form of the matrix and changing the order in which products are taken

$$\mathbf{L}(\mathbf{L}^T \boldsymbol{X}) = \boldsymbol{B}$$

- Let $\mathbf{L}^T \boldsymbol{X} = \boldsymbol{Y}$ and substitute

- Now sequentially solve

$$\mathbf{L}\boldsymbol{Y} = \boldsymbol{B} \quad \text{by forward substitution}$$

$$\mathbf{L}^T \boldsymbol{X} = \boldsymbol{Y} \quad \text{by backward substitution}$$

$\mathbf{LDL}^T$ *Method*

- Decompose $\mathbf{A} = \mathbf{LDL}^T$
  - Where $\mathbf{L}$ is a lower triangular matrix
  - Where $\mathbf{D}$ = diagonal matrix
- Set the diagonal terms of $\mathbf{L}$ to unity
- Solving for the elements of $\mathbf{L}$ and $\mathbf{D}$

$$\mathbf{A} = \mathbf{LDL}^T$$

- Substituting and changing the order in which the products are formed

$$\mathbf{L}(\mathbf{DL}^T \boldsymbol{X}) = \boldsymbol{B}$$

- Now let

$$\mathbf{DL}^T \boldsymbol{X} = \boldsymbol{Y}$$

- Now sequentially solve

$$\mathbf{L}\boldsymbol{Y} = \boldsymbol{B} \quad \text{by forward substitution}$$

$$\mathbf{L}^T\boldsymbol{X} = \mathbf{D}^{-1}\boldsymbol{Y} \quad \text{by backward substitution}$$

- Note that $\mathbf{D}^{-1}$ = 1/diagonal terms and are easily computed

Computation of the Determinant for Factorization Methods

- Notes
  - If $\mathbf{A} = \mathbf{L}\mathbf{U} \Rightarrow |\mathbf{A}| = |\mathbf{L}||\mathbf{U}|$
  - The determinant of a triangularized matrix equals the product of the diagonal terms
- In case of $\mathbf{L}\mathbf{U}$ factorization

$$|\mathbf{A}| = \prod_{i=1}^{N} l_{ii} \qquad \text{(Crout)}$$

$$|\mathbf{A}| = \prod_{i=1}^{N} u_{ii} \qquad \text{(Doolittle)}$$

- In case of $\mathbf{L}\mathbf{D}\mathbf{L}^T$ decomposition

$$|\mathbf{A}| = |\mathbf{D}| = \prod_{i=1}^{N} d_{ii}$$

- For Gauss Elimination, simply keep a running product of the ***pivot elements***.