DTS104TC
NUMERICAL METHODS

LECTURE 2

LONG HUANG

Xi'an Jiaotong-Liverpool University
西交利物浦大学

CONTENTS

- Summary of Root Finding Methods
- Linear Algebraic Equations
  - Gauss Elimination
  - LU Decomposition and Matrix Inversion

SUMMARY OF ROOT FINDING METHODS

Method	Type	Guesses	Convergence	Stability	Programming	Comments
Direct	Analytical	—	—	—		
Graphical	Visual	—	—	—	—	Imprecise
Bisection	Bracketing	2	Slow	Always	Easy	
False-position	Bracketing	2	Slow/medium	Always	Easy	
Modified FP	Bracketing	2	Medium	Always	Easy	
Fixed-point iteration	Open	1	Slow	Possibly divergent	Easy	
Newton-Raphson	Open	1	Fast	Possibly divergent	Easy	Requires evaluation of $f'(x)$

SUMMARY OF ROOT FINDING METHODS

Method	Type	Guesses	Convergence	Stability	Programming	Comments
Modified Newton-Raphson	Open	1	Fast (multiple), medium (single)	Possibly divergent	Easy	Requires evaluation of $f'(x)$ and $f''(x)$
Secant	Open	2	Medium/fast	Possibly divergent	Easy	Initial guesses do not have to bracket the root
Modified secant	Open	1	Medium/fast	Possibly divergent	Easy	
Brent	Hybrid	1 or 2	Medium	Always (for 2 guesses)	Moderate	Robust
Müller	Polynomials	2	Medium/fast	Possibly divergent	Moderate	
Bairstow	Polynomials	2	Fast	Possibly divergent	Moderate	

Gauss Elimination

LINEAR ALGEBRAIC EQUATIONS

- An equation of the form $ax + by + c = 0$ or equivalently $ax + by = -c$ is called a linear equation in x and y variables.
- $ax + by + cz = d$ is a linear equation in three variables, x, y, and z.
- Thus, a linear equation in n variables is,

$$a_1x_1 + a_2x_2 + \dots + a_nx_n = b$$

- A solution of such an equation consists of real numbers x_1, x_2, x_3, … , x_n. If you need to work more than one linear equation, a system of linear equations must be solved simultaneously.

NON-COMPUTER METHODS FOR SOLVING SYSTEMS OF EQUATIONS

- For small number of equations ($n \leq 3$) linear equations can be solved readily by simple techniques such as “method of elimination.”

- Linear algebra provides the tools to solve such systems of linear equations.

- Nowadays, easy access to computers makes the solution of large sets of linear algebraic equations possible and practical.

NON-COMPUTER METHODS FOR SOLVING SYSTEMS OF EQUATIONS

Solving Small Numbers of Equations

- There are many ways to solve a system of linear equations:

- Graphical method.
- Cramer's rule.
- Method of elimination.
- Computer methods.

For $n \leq 3$

GRAPHICAL METHOD

- For two equations:

$$a_{11}x_1 + a_{12}x_2 = b_1$$

$$a_{21}x_1 + a_{22}x_2 = b_2$$

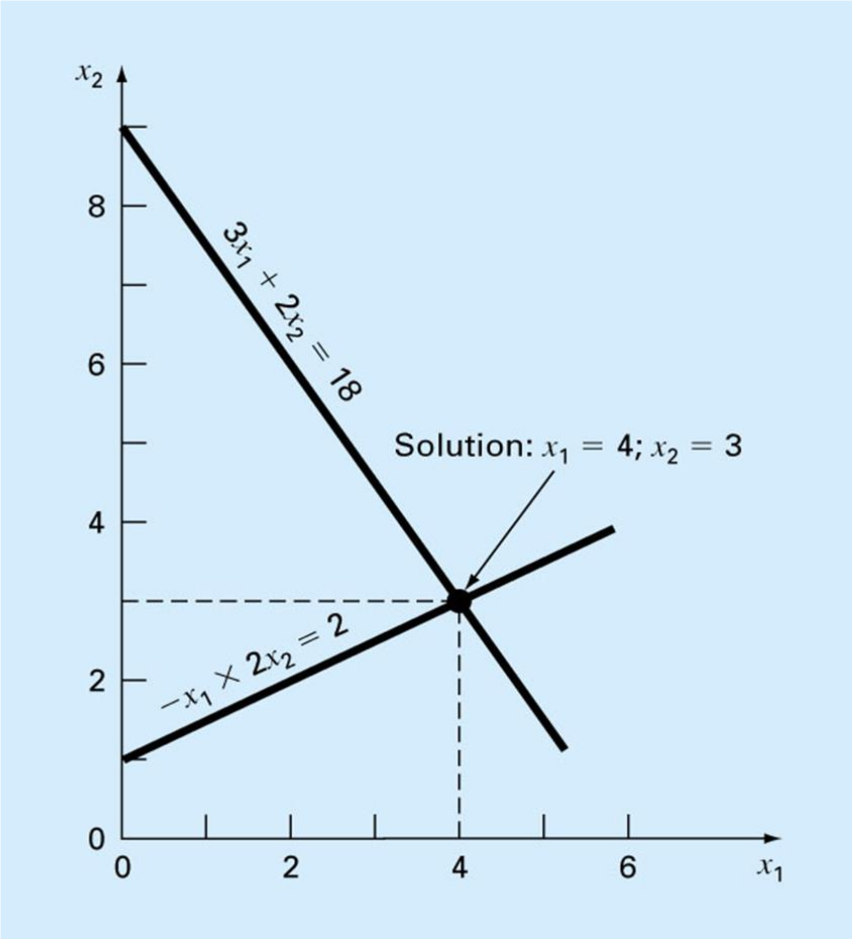
- Solve both equations for x_2:

$$x_2 = -\left(\frac{a_{11}}{a_{12}}\right)x_1 + \frac{b_1}{a_{12}} \quad \Rightarrow \quad x_2 = (\text{slope})x_1 + \text{intercept}$$

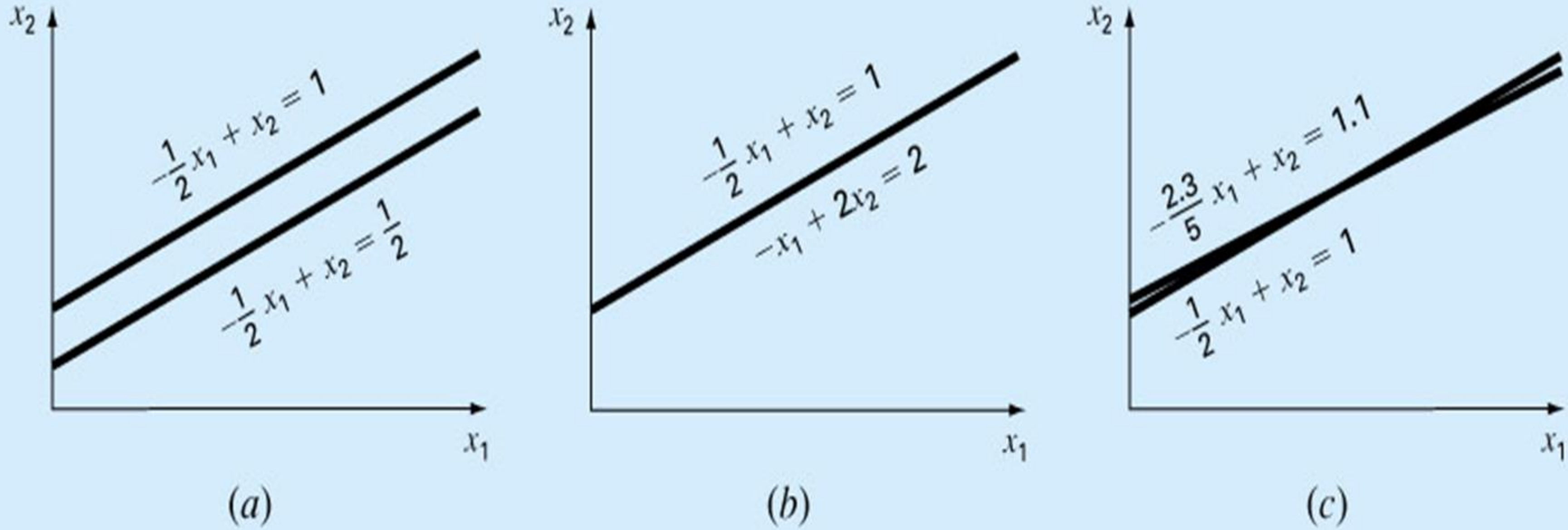
$$x_2 = -\left(\frac{a_{21}}{a_{22}}\right)x_1 + \frac{b_2}{a_{22}}$$

GRAPHICAL METHOD

- Plot x_2 versus x_1 on rectilinear paper, the intersection of the lines present the solution.



VISUALIZATION OF SINGULAR & ILL-CONDITIONED SYSTEMS



Graphical depiction of singular and ill-conditioned systems: (a) no solution, (b) infinite solutions, and (c) ill-conditioned system where the slopes are so close that the point of intersection is difficult to detect visually.

DETERMINANTS AND CRAMER'S RULE

- Determinant can be illustrated for a set of three equations:

$$[A]\{x\} = \{B\}$$

- Where [*A*] is the coefficient matrix:

$$[A] = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

DETERMINANTS AND CRAMER'S RULE

- Assuming all matrices are square matrices, there is a number associated with each square matrix [*A*] called the determinant, *D*, of [*A*]. If [*A*] is order 1, then [*A*] has one element:

$$[A] = [a_{11}]$$

$$D = a_{11}$$

- For a square matrix of order 3, the ***minor*** of an element a_{ij} is the determinant of the matrix of order 2 by deleting row *i* and column *j* of [*A*].

DETERMINANTS AND CRAMER'S RULE

$$D = \begin{vmatrix} \cancel{a_{11}} & \cancel{a_{12}} & \cancel{a_{13}} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix}$$

$$D_{11} = \begin{vmatrix} a_{22} & a_{23} \\ a_{32} & a_{33} \end{vmatrix} = a_{22}\, a_{33} - a_{32}\, a_{23}$$

$$D_{12} = \begin{vmatrix} a_{21} & a_{23} \\ a_{31} & a_{33} \end{vmatrix} = a_{21}\, a_{33} - a_{31}\, a_{23}$$

$$D_{13} = \begin{vmatrix} a_{21} & a_{22} \\ a_{31} & a_{32} \end{vmatrix} = a_{21}\, a_{32} - a_{31}\, a_{22}$$

DETERMINANTS AND CRAMER'S RULE

$$D = a_{11}\begin{vmatrix} a_{22} & a_{23} \\ a_{32} & a_{33} \end{vmatrix} - a_{12}\begin{vmatrix} a_{21} & a_{23} \\ a_{31} & a_{33} \end{vmatrix} + a_{13}\begin{vmatrix} a_{21} & a_{22} \\ a_{31} & a_{32} \end{vmatrix}$$

- *Cramer's rule* expresses the solution of a systems of linear equations in terms of ratios of determinants of the array of coefficients of the equations. For example, x_1 would be computed as:

$$x_1 = \frac{\begin{vmatrix} b_1 & a_{12} & a_{13} \\ b_2 & a_{22} & a_{23} \\ b_3 & a_{32} & a_{33} \end{vmatrix}}{D}$$

METHOD OF ELIMINATION

- The basic strategy is to successively solve one of the equations of the set for one of the unknowns and to eliminate that variable from the remaining equations by substitution.

- The elimination of unknowns can be extended to systems with more than two or three equations; however, the method becomes extremely tedious to solve by hand.

NAIVE GAUSS ELIMINATION

Extension of *method of elimination* to large sets of equations by developing a systematic scheme or algorithm to eliminate unknowns and to back substitute.

As in the case of the solution of two equations, the technique for *n* equations consists of two phases:

- Forward elimination of unknowns.
- Back substitution.

NAIVE GAUSS ELIMINATION

$$\left[\begin{array}{ccc|c} a_{11} & a_{12} & a_{13} & c_1 \\ a_{21} & a_{22} & a_{23} & c_2 \\ a_{31} & a_{32} & a_{33} & c_3 \end{array}\right]$$

$$\Downarrow$$

$$\left[\begin{array}{ccc|c} a_{11} & a_{12} & a_{13} & c_1 \\ & a'_{22} & a'_{23} & c'_2 \\ & & a''_{33} & c''_3 \end{array}\right]$$

Forward elimination

$$\Downarrow$$

$$x_3 = c''_3/a''_{33}$$

$$x_2 = (c'_2 - a'_{23}x_3)/a'_{22}$$

$$x_1 = (c_1 - a_{12}x_2 - a_{13}x_3)/a_{11}$$

Back substitution

PITFALLS OF ELIMINATION METHODS

- *Division by zero*. It is possible that during both elimination and back-substitution phases a division by zero can occur.
- *Round-off errors*.
- *Ill-conditioned systems*. Systems where small changes in coefficients result in large changes in the solution. Alternatively, it happens when two or more equations are nearly identical, resulting a wide ranges of answers to approximately satisfy the equations. Since round off errors can induce small changes in the coefficients, these changes can lead to large solution errors.

PITFALLS OF ELIMINATION METHODS

- *Singular systems*. When two equations are identical, we would loose one degree of freedom and be dealing with the impossible case of $n-1$ equations for n unknowns. For large sets of equations, it may not be obvious however. The fact that the determinant of a singular system is zero can be used and tested by computer algorithm after the elimination stage. If a zero diagonal element is created, calculation is terminated.

TECHNIQUES FOR IMPROVING SOLUTIONS

Use of more significant figures.

Pivoting. If a pivot element is zero, normalization step leads to division by zero. The same problem may arise, when the pivot element is close to zero. Problem can be avoided:

- *Partial pivoting*. Switching the rows so that the largest element is the pivot element.
- *Complete pivoting*. Searching for the largest element in all rows and columns then switching.

GAUSS-JORDAN

It is a variation of Gauss elimination. The major differences are:

- When an unknown is eliminated, it is eliminated from all other equations rather than just the subsequent ones.
- All rows are normalized by dividing them by their pivot elements.
- Elimination step results in an identity matrix.
- Consequently, it is not necessary to employ back substitution to obtain solution.

FLOPS: FLOATING POINT OPERATIONS

COUNTING ADDITIONS,

SUBTRACTIONS,

MULTIPLICATIONS AND DIVISIONS

FLOP COUNT FOR NAIVE GAUSS ELIMINATION

Forward elimination: $\dfrac{2n^3}{3}+O\left(n^2\right)$

- **Back substitution:** $n^2+O\left(n\right)$

- **TOTAL:**

$$\underbrace{\frac{2n^3}{3}+O\left(n^2\right)}_{\text{forward elimination}}+\underbrace{n^2+O\left(n\right)}_{\text{back substitution}} \xrightarrow{\text{as } n \text{ increases}} \frac{2n^3}{3}+O\left(n^2\right)$$

GAUSS-JORDAN

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & | \, b_1 \\ a_{21} & a_{22} & a_{23} & | \, b_2 \\ a_{31} & a_{32} & a_{33} & | \, b_3 \end{bmatrix}$$

$$\Downarrow$$

$$\begin{bmatrix} 1 & 0 & 0 & | \, b_1^{(n)} \\ 0 & 1 & 0 & | \, b_2^{(n)} \\ 0 & 0 & 1 & | \, b_3^{(n)} \end{bmatrix}$$

$$\Downarrow$$

$$\begin{bmatrix} x_1 & & & = b_1^{(n)} \\ & x_2 & & = b_2^{(n)} \\ & & x_3 & = b_3^{(n)} \end{bmatrix}$$

$$FLOPS \cong n^3 + O\left(n^2\right)$$

RECALL FOR GAUSS ELIMINATION,

$$FLOPS \cong \frac{2n^3}{3} + O\left(n^2\right)$$

- ***Therefore, Gauss Elimination is superior to Gauss-Jordan.***

LU Decomposition

LU DECOMPOSITION AND MATRIX INVERSION

- Provides an efficient way to compute matrix inverse by separating the time consuming elimination of the Matrix [A] from manipulations of the right-hand side {b}.

- *Gauss elimination*, in which the forward elimination comprises the bulk of the computational effort, can be implemented as an *LU* decomposition.

LU DECOMPOSITION

The system, $[A]\{x\}=\{b\}$, can be decomposed into two matrices

$$[L]: \text{lower triangular matrix}$$

$$[U]: \text{ upper triangular matrix}$$

- such that

$$[L][U]=[A]$$

$$[L][U]\{x\}=\{b\}$$

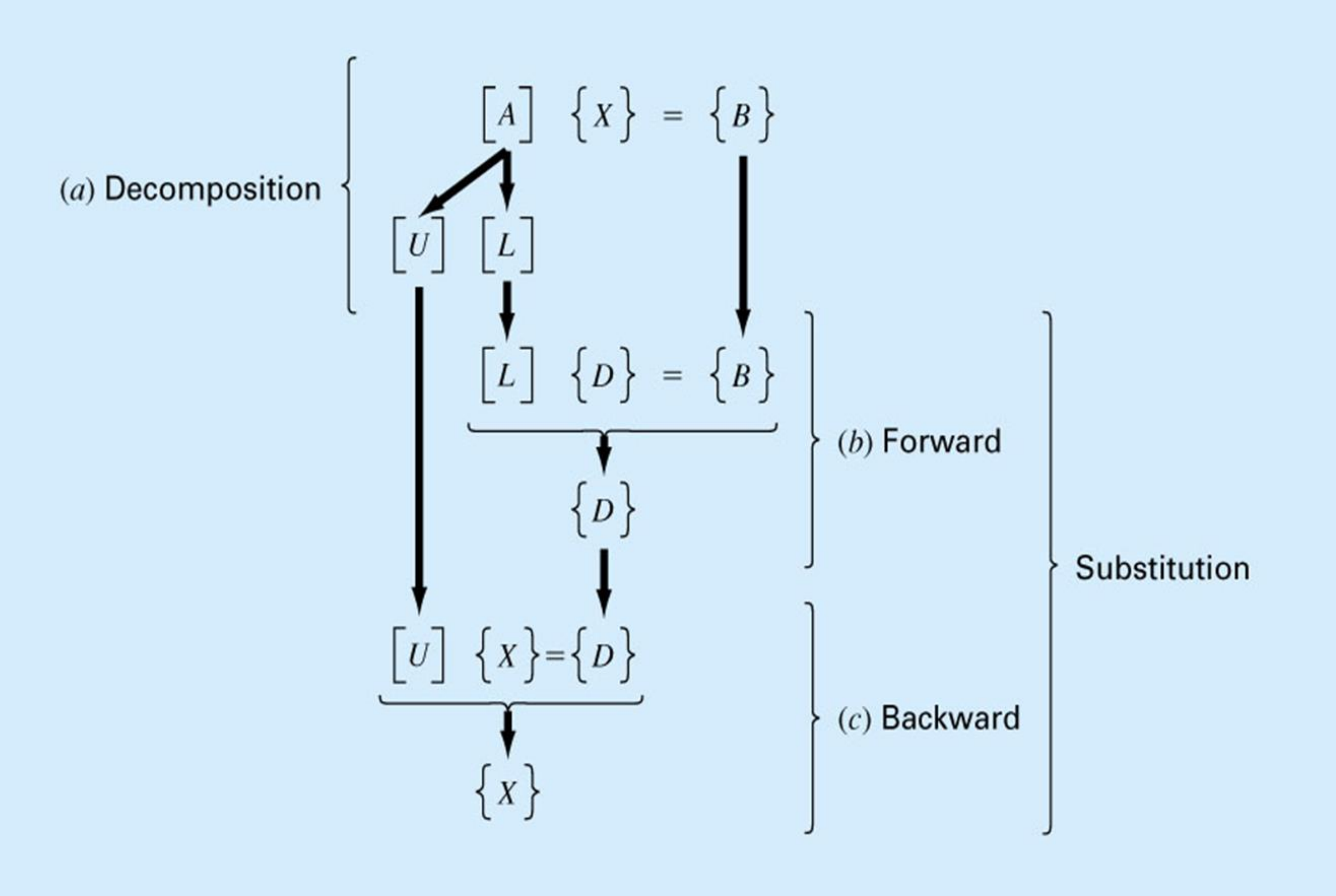
- Set up the following systems

$$[L]\{d\}=\{b\}$$

$$[U]\{x\}=\{d\}$$

- Step 1: $[L]\{d\}=\{b\}$ is used to generate an intermediate vector $\{d\}$ by forward substitution.
- Step 2: $[U]\{x\}=\{d\}$ is used to get the answer, $\{x\}$, by back substitution.

LU DECOMPOSITION



FLOPS FOR LU DECOMPOSITION

- Requires the same total FLOPS as for *Gauss elimination*.
- Saves computing time by separating time-consuming elimination step from the manipulations of the right hand side.
- Provides efficient means to compute the matrix inverse.

SYSTEM CONDITION OVERVIEW

- The coefficient matrix of some systems of linear algebraic systems are intrinsically *ill-conditioned*.

- This means that a small change in the constant coefficients results in a large change in the solution.

- A *condition number*, defined in more advanced courses, is used to measure the degree of ill-conditioning of the matrix.

CUMBERSOME WAYS TO DETECT SYSTEM CONDITION

THE BASIC IDEA: Matrix inversion exposes ill-conditioning

1. Scale the matrix of coefficients, [*A*], so that the largest element in each row is 1. If there are elements of $[A]^{-1}$ that are several orders of magnitude greater than one, it is likely that the system is ill-conditioned.

2. Multiply the inverse by the original coefficient matrix and assess whether the result is close to the identity matrix. If not, it indicates ill conditioning.

3. Invert the inverted matrix and assess whether the result is sufficiently close to the original coefficient matrix. If not, it again indicates that the system is ill-conditioned.

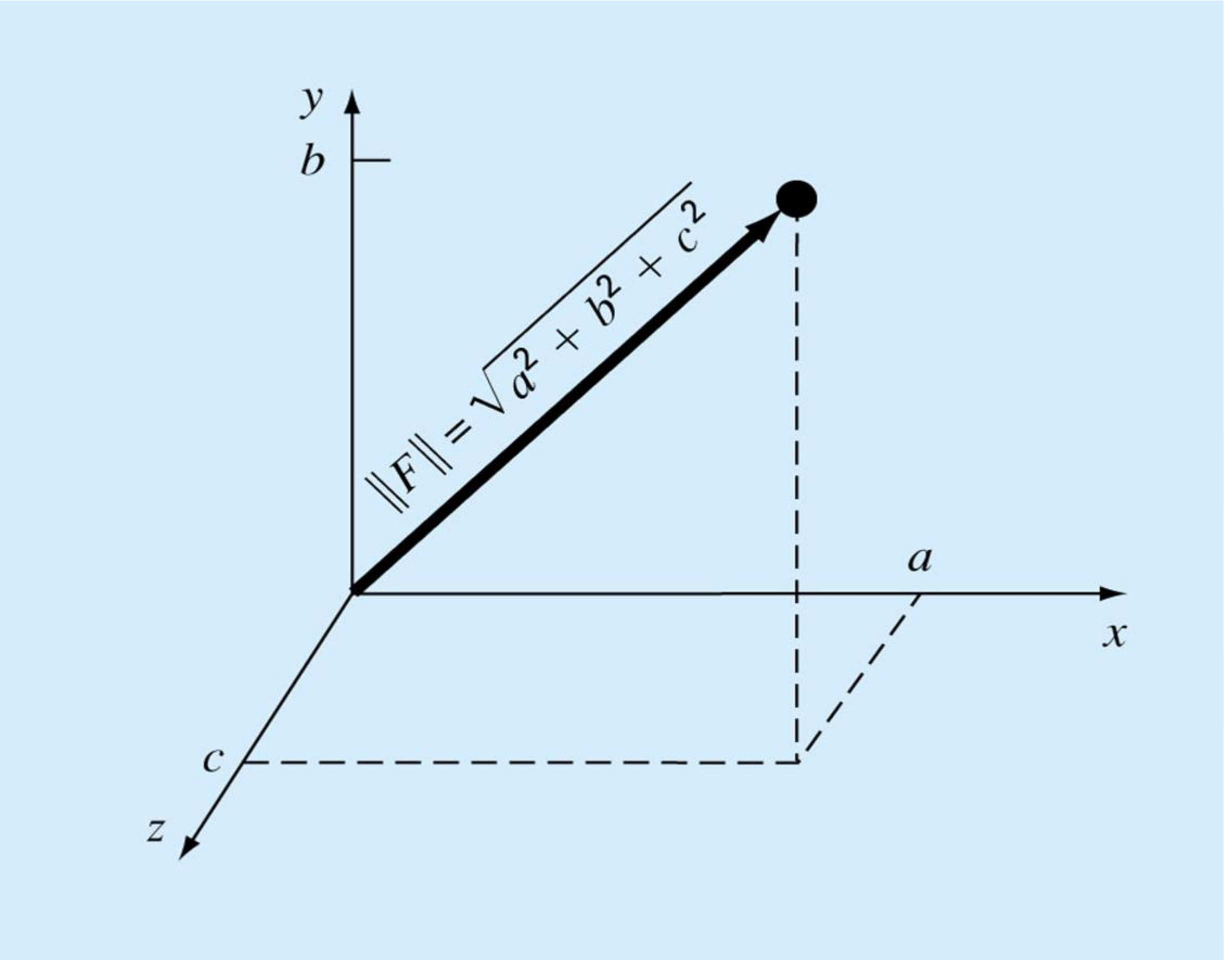
ERROR ANALYSIS AND SYSTEM CONDITION

- Inverse of a matrix provides a means to test whether systems are ill-conditioned.

Vector and Matrix Norms

- *Norm* is a real-valued function that provides a measure of size or “length” of vectors and matrices. Norms are useful in studying the error behavior of algorithms.

VECTOR AND MATRIX NORMS



Graphical depiction of a vector $\lfloor F \rfloor = \lfloor a \; b \; c \rfloor$ in Euclidean space.

EUCLIDIAN NORMS

$$\|X\|_e = \sqrt{\sum_{i=1}^{n} x_i^2}$$

Euclidian Norm

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

Frobenius Norm

ROW SUM NORM

$$\|X\|_\infty = \max_{1 \le i \le n} |x_i|$$

Maximum magnitude norm

$$\|A\|_\infty = \max_{1 \le i \le n} \sum_{j=1}^{n} |a_{ij}|$$

Row-sum norm

COLUMN SUM NORM

$$\left\| X \right\|_1 = \sum_{i=1}^{n} \left| x_i \right|$$

Sum norm

$$\left\| A \right\|_1 = \max_{1 \le j \le n} \sum_{i=1}^{n} \left| a_{ij} \right|$$

Column-sum norm

SPECTRAL NORM

$$\|A\|_2 = (\mu_{\max})^{1/2}$$

Largest eigenvalue of $[A]^T[A]$.

Spectral Norm is the minimum norm.

THE CONDITION NUMBER

$$\mathrm{Cond}[A] = \|A\| \cdot \|A^{-1}\|$$

$$\frac{\|\Delta X\|}{\|X\|} \leq \mathrm{Cond}[A] \frac{\|\Delta A\|}{\|A\|}$$

$$\boldsymbol{c} = \mathbf{Log}_{10}\left(\mathbf{Cond}[\boldsymbol{A}]\right)$$

Significant digits that are suspect.

CONDITION NUMBER SUMMARY

$$\frac{\|\Delta X\|}{\|X\|} \leq Cond\left[A\right]\frac{\|\Delta A\|}{\|A\|}$$

- That is, the relative error of the norm of the computed solution can be as large as the relative error of the norm of the coefficients of [*A*] multiplied by the condition number.
- For example, if the coefficients of [*A*] are known to *t*-digit precision $\left(\text{rounding errors } \sim 10^{-t}\right)$ and $\text{Cond}\left[A\right] = 10^{c}$, the solution {*x*} may be valid to only *t* – *c* digits (rounding errors $\sim 10^{c-t}$).