2.29 Numerical Fluid Mechanics
Spring 2015 – **Lecture 5**

Review of Lecture 4

Reference: Chapra and Canale, Chapters 5 and 6

- Roots of nonlinear equations
  - Bracketing Methods
    - Example: Heron's formula
    - Bisection and False Position
  - "Open" Methods
    - Fixed-point Iteration (General method or Picard Iteration)
      - Examples, Convergence Criteria
      - Order of Convergence

$$x_{n+1} = g(x_n) \quad \text{or}$$
$$x_{n+1} = x_n - h(x_n) f(x_n)$$

$$\lim_{n\to\infty} \frac{|e_{n+1}|}{|e_n|^p} = C$$

    - Newton-Raphson
      - Convergence speed and examples

$$x_{n+1} = x_n - \frac{1}{f'(x_n)} f(x_n)$$

    - Secant Method
      - Examples, Convergence and efficiency
    - Extension of Newton-Raphson to systems of nonlinear equations
  - Roots of Polynomial (all real/complex roots)
    - Open methods (applications of the above for complex numbers) and Special Methods (e.g. Muller's and Bairstow's methods)
- Systems of Linear Equations
  - Motivations and Plans
  - Direct Methods

TODAY's Lecture: Systems of Linear Equations

- Direct Methods
  - Cramer's Rule
  - Gauss Elimination
    - Algorithm
    - Numerical implementation and stability
      - Partial Pivoting
      - Equilibration
      - Full Pivoting
      - Well suited for dense matrices
      - Issues: round-off, cost, does not vectorize/parallelize well
    - Special cases, Multiple right hand sides, Operation count
  - LU decomposition/factorization
  - Error Analysis for Linear Systems
      - Condition Number
  - Special Matrices: Tri-diagonal systems
- Iterative Methods
  - Jacobi's method
  - Gauss-Seidel iteration
  - Convergence

Reading Assignment

- Chapters 9 and 10 of "Chapra and Canale, Numerical Methods for Engineers, 2006/2010/204."
  - Any chapter on "Solving linear systems of equations" in references on CFD that we provided. For example: chapter 5 of "J. H. Ferziger and M. Peric, Computational Methods for Fluid Dynamics. Springer, NY, 3rd edition, 2002"

Direct Methods for Small Systems: Determinants and Cramer's Rule

Linear System of Equations:

$$\begin{array}{ccccccc} a_{11}x_1 & a_{12}x_2 & \cdot & \cdot & a_{1n}x_n & = & b_1 \\ a_{21}x_1 & a_{22}x_2 & \cdot & \cdot & a_{2n}x_n & = & b_2 \\ \cdot & \cdot & \cdot & \cdot & \cdot & = & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & = & \cdot \\ a_{n1}x_1 & \cdot & \cdot & \cdot & a_{nn}x_n & = & b_n \end{array}$$

Recall, for a 2 by 2 matrix, the determinant is:

$$D = \begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix} = a_{11}\,a_{22} - a_{21}\,a_{12}$$

Recall, for a 3 by 3 matrix, the determinant is:

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

Direct Methods for small systems: Determinants and Cramer's Rule

Cramer's rule:

"Each unknown x_i in a system of linear algebraic equations can be expressed as a fraction of two determinants:

- Denominator is determinant D
- Numerator is D but with column i replaced by $\boldsymbol{b}$"

$$x_i = \frac{\begin{array}{c} \qquad\qquad \text{i}^{\text{th}}\ \text{column} \qquad\qquad \\ \begin{vmatrix} a_{11} & b_1 & a_{1n} \\ & b_2 & \\ & & \\ a_{n1} & b_n & a_{nn} \end{vmatrix} \end{array}}{D}$$

Example: Cramer's Rule, n=2

$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$$

$$\begin{aligned} D &= a_{11}a_{22} - a_{21}a_{12} \\ D_1 &= b_1 a_{22} - b_2 a_{12} \\ D_2 &= b_2 a_{11} - b_1 a_{21} \end{aligned}$$

$$x_1 = \frac{D_1}{D} = \frac{b_1 a_{22} - b_2 a_{12}}{a_{11}a_{22} - a_{21}a_{12}}$$

$$x_2 = \frac{D_2}{D} = \frac{b_2 a_{11} - b_1 a_{21}}{a_{11}a_{22} - a_{21}a_{12}}$$

Numerical case:

$$\begin{bmatrix} 0.01 & -1.0 \\ 1.0 & 0.01 \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \end{Bmatrix} = \begin{Bmatrix} 1.0 \\ 1.0 \end{Bmatrix}$$

$$\begin{aligned} x_1 &= \frac{1.0\cdot 0.01 - 1.0\cdot(-1.0)}{0.01\cdot 0.01 - 1.0\cdot(-1.0)} &= \quad 1.0099 \\ x_2 &= \frac{1.0\cdot 0.01 - 1.0\cdot 1.0}{0.01\cdot 0.01 - 1.0\cdot(-1.0)} &= -0.9899 \end{aligned}$$

Cramer's rule becomes impractical for n>3:
The number of operations is of *O*(n!)

Direct Methods for large dense systems
Gauss Elimination

- Main idea: "combine equations so as to eliminate unknowns systematically"
  - Solve for each unknown one by one
  - Back-substitute result in the original equations
  - Continue with the remaining unknowns

- General Gauss Elimination Algorithm
  i. Forward Elimination/Reduction to Upper Triangular Systems)
  ii. Back-Substitution

- Comments:
  - Well suited for dense matrices
  - Some modification of above simple algorithm needed to avoid division by zero and other pitfalls

Linear System of Equations

$$\begin{array}{ccccccc}
a_{11}x_1 & a_{12}x_2 & \cdot & \cdot & a_{1n}x_n & = & b_1 \\
a_{21}x_1 & a_{22}x_2 & \cdot & \cdot & a_{2n}x_n & = & b_2 \\
\cdot & \cdot & \cdot & \cdot & \cdot & = & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & = & \cdot \\
a_{n1}x_1 & \cdot & \cdot & \cdot & a_{nn}x_n & = & b_n
\end{array}$$

Gauss Elimination

Linear System of Equations

$$
\begin{array}{ccccccc}
a_{11}x_1 & a_{12}x_2 & \cdot & \cdot & a_{1n}x_n & = & b_1 \\
a_{21}x_1 & a_{22}x_2 & \cdot & \cdot & a_{2n}x_n & = & b_2 \\
\cdot & \cdot & \cdot & \cdot & \cdot & = & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & = & \cdot \\
a_{n1}x_1 & \cdot & \cdot & \cdot & a_{nn}x_n & = & b_n
\end{array}
$$

Reduction / Forward Elimination Step 0

$$a_{ij}^{(1)} = a_{ij}, \quad b_i^{(1)} = b_i$$

$$
\begin{array}{ccccccc}
a_{11}^{(1)}x_1 & a_{12}^{(1)}x_2 & \cdot & \cdot & a_{1n}^{(1)}x_n & = & b_1^{(1)} \\
a_{21}^{(1)}x_1 & a_{22}^{(1)}x_2 & \cdot & \cdot & a_{2n}^{(1)}x_n & = & b_2^{(1)} \\
\cdot & \cdot & \cdot & \cdot & \cdot & = & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & = & \cdot \\
a_{n1}^{(1)}x_1 & \cdot & \cdot & \cdot & a_{nn}^{(1)}x_n & = & b_n^{(1)}
\end{array}
$$

If a_{11} is non zero, we can eliminate x_1 from the remaining equations 2 to (n-1) by multiplying equation 1 with $\frac{a_{i1}}{a_{11}}$ and subtracting the result from equation i .

This leads to the following algorithm for “Step 1”:

Gauss Elimination

Reduction / Forward Elimination: Step 1

$$\left.\begin{aligned} m_{i1} &= \frac{a_{i1}^{(1)}}{a_{11}^{(1)}} \\ a_{ij}^{(2)} &= a_{ij}^{(1)} - m_{i1} a_{1j}^{(1)}, \quad j = 1, \cdots n \\ b_i^{(2)} &= b_i^{(1)} - m_{i1} b_1^{(1)} \end{aligned}\right\} i = 2, \cdots n$$

Subtract multiple of row 1 from rows 2 to n

a_{11} is called pivot element:

$$\begin{array}{c} \xrightarrow{\quad j \quad} \\ i \downarrow \begin{array}{ccccccc} a_{11}^{(1)} x_1 & a_{12}^{(1)} x_2 & \cdot & \cdot & a_{1n}^{(1)} x_n & = & b_1^{(1)} \\ 0 & a_{22}^{(2)} x_2 & \cdot & \cdot & a_{2n}^{(2)} x_n & = & b_2^{(2)} \\ \cdot & \cdot & \cdot & \cdot & \cdot & = & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & = & \cdot \\ 0 & a_{n2}^{(2)} x_2 & \cdot & \cdot & a_{nn}^{(2)} x_n & = & b_n^{(2)} \end{array} \end{array}$$

(is called Pivot equation for step 1)

Notes:
- Result of step 1: last (n-1) equations have (n-1) unknowns
- Pivot a_{11} needs to be non-zero

Gauss Elimination

Reduction: Step k

Recursive repetition of step 1 for successively reduced set of (n-k) equations:

$$\left.\begin{aligned} m_{ik} &= \frac{a_{ik}^{(k)}}{a_{kk}^{(k)}} \\ a_{ij}^{(k+1)} &= a_{ij}^{(k)} - m_{ik} a_{kj}^{(k)}, \quad j = k, \cdots n \\ b_i^{(k+1)} &= b_i^{(k)} - m_{ik} b_k^{(k)} \end{aligned}\right\} \; i = k+1, \cdots n$$

The result after completion of step k is:

$$\begin{array}{cccccccc} a_{11}^{(1)} x_1 & a_{12}^{(1)} x_2 & \cdot & \cdot & a_{1n}^{(1)} x_n & = & b_1^{(1)} \\ 0 & a_{22}^{(2)} x_2 & \cdot & \cdot & a_{2n}^{(2)} x_n & = & b_2^{(2)} \\ 0 & \cdot & a_{kk}^{(k)} x_k & \cdot & \cdot & = & \cdot \\ 0 & \cdot & 0 & \cdot & \cdot & = & \cdot \\ 0 & \cdot & 0 & \cdot & a_{nn}^{(k+1)} x_n & = & b_n^{(k+1)} \end{array}$$

First non-zero element on row n: $a_{n,k+1}^{(k+1)} x_k$

Gauss Elimination

Reduction/Elimination: Step k

$$\left.\begin{aligned} m_{ik} &= \frac{a_{ik}^{(k)}}{a_{kk}^{(k)}} \\ a_{ij}^{(k+1)} &= a_{ij}^{(k)} - m_{ik} a_{kj}^{(k)}, \quad j = k, \cdots n \\ b_i^{(k+1)} &= b_i^{(k)} - m_{ik} b_k^{(k)} \end{aligned}\right\} \; i = k+1, \cdots n$$

Reduction: Step (n-1)

$$\begin{array}{cccccccc} a_{11}^{(1)} x_1 & a_{12}^{(1)} x_2 & \cdot & \cdot & & a_{1n}^{(1)} x_n & = & b_1^{(1)} \\ 0 & a_{22}^{(2)} x_2 & \cdot & \cdot & & a_{2n}^{(2)} x_n & = & b_2^{(2)} \\ 0 & \cdot & \cdot & \cdot & & \cdot & = & \cdot \\ 0 & \cdot & 0 & a_{n-1,n-1}^{(n-1)} x_{n-1} & & a_{n-1,n}^{(n-1)} x_n & = & b_{n-1}^{(n-1)} \\ 0 & \cdot & \cdot & 0 & & a_{nn}^{(n)} x_n & = & b_n^{(n)} \end{array}$$

Back-Substitution

$$\begin{aligned} x_n &= b_n^{(n)} / a_{nn}^{(n)} \\ x_{n-1} &= \left(b_{n-1}^{(n-1)} - a_{n-1,n}^{(n-1)} x_n \right) / a_{n-1,n-1}^{(n-1)} \\ & \cdot \quad \cdot \quad \cdot \\ & \cdot \quad \cdot \quad \cdot \\ x_k &= \left(b_k^{(k)} - \sum_{j=k+1}^{n} a_{kj}^{(k)} x_j \right) / a_{kk}^{(k)} \\ & \cdot \quad \cdot \quad \cdot \\ x_1 &= \left(b_1^{(1)} - \sum_{j=2}^{n} a_{1j}^{(1)} x_j \right) / a_{11}^{(1)} \end{aligned}$$

Result after step (n-1) is an Upper triangular system!

Gauss Elimination:
Number of Operations

Reduction/Elimination: Step k

$$\left.\begin{aligned} m_{ik} &= \frac{a_{ik}^{(k)}}{a_{kk}^{(k)}} \\ a_{ij}^{(k+1)} &= a_{ij}^{(k)} - m_{ik} a_{kj}^{(k)}, \quad j = k, \cdots n \\ b_i^{(k+1)} &= b_i^{(k)} - m_{ik} b_k^{(k)} \end{aligned}\right\} \; i = k+1, \cdots n$$

: n-k divisions

: 2 (n-k) (n-k+1) additions/multiplications

: 2 (n-k) additions/multiplications

For reduc., total number of ops: $\sum_{k=1}^{n-1} 3(n-k) + 2(n-k)\,(n-k+1) = \frac{3n(n-1)}{2} + \frac{2n(n^2-1)}{3} = O(\frac{2}{3}n^3)$

Use: $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$ and $\sum_{i=1}^{n} i^2 = \frac{n(n+1)(2n+1)}{6}$

Back-Substitution

$$x_k = \left(b_k^{(k)} - \sum_{j=k+1}^{n} a_{kj}^{(k)} x_j \right) / a_{kk}^{(k)}$$

: (n-k-1)+(n-k)+2=2(n-k) +1 additions/multiplications

Hence, total number of ops is: $1 + \sum_{k=1}^{n-1} \big(2(n-k)+1\big) = 1 + (n-1)(n+1) = n^2$ $\quad (\sum_{i=1}^{n} i = \frac{n(n+1)}{2})$

(the first 1 before the sum is for x_n)

Grand total number of ops is $O(\frac{2}{3}n^3) = O(n^3)$ **:**

- Grows rapidly with n
- Most ops occur in elimination step

Gauss Elimination:
Issues and Pitfalls to be addressed

- Division by zero:
  - Pivot elements $a_{k,k}^{(k)}$ must be non-zero and should not be close to zero
- Round-off errors
  - Due to recursive computations and so error propagation
  - Important when large number of equations are solved
  - Always substitute solution found back into original equations
  - Scaling of variables can be used
- Ill-conditioned systems
  - Occurs when one or more equations are nearly identical
  - If determinant of normalized system matrix $\mathbf{A}$ is close to zero, system will be ill-conditioned (in general, if $\mathbf{A}$ is not well conditioned)
  - Determinant can be computed using Gauss Elimination
    - Since forward-elimination consists of simple scaling and addition of equations, the determinant is the product of diagonal elements of the Upper Triangular System

Gauss Elimination: Pivoting

Reduction
Step k

$$\left.\begin{aligned} m_{ik} &= \frac{a_{ik}^{(k)}}{a_{kk}^{(k)}} \\ a_{ij}^{(k+1)} &= a_{ij}^{(k)} - m_{ik}a_{kj}^{(k)}, \quad j = k, \cdots n \\ b_i^{(k+1)} &= b_i^{(k)} - m_{ik}b_k^{(k)} \end{aligned}\right\} i = 2, \cdots n$$

Pivot Elements

$$a_{11}^{(1)}, a_{22}^{(2)}, \ldots, a_{nn}^{(n)}$$

Required at each step!

$$\boxed{a_{kk}^{(k)} \neq 0}$$

Partial Pivoting by Columns

$$\begin{array}{l} \\ \\ \\ \text{Row k} \\ \updownarrow \\ \text{Row i} \\ \\ \\ \end{array} \begin{bmatrix} \times & & & & & & \\ 0 & \times & & & & & \\ \cdot & \cdot & \times & & & & \\ \cdot & & 0 & \boxed{0 & \times & \times & \times} \\ \cdot & & \cdot & \times & & & \\ \cdot & & \cdot & \boxed{\times} & & & \\ \cdot & & \cdot & \times & & & \\ 0 & \cdot & 0 & \times & & & \end{bmatrix} \cdot \bar{\mathbf{x}} = \begin{Bmatrix} \times \\ \cdot \\ \cdot \\ \boxed{\times} \\ \cdot \\ \boxed{\cdot} \\ \cdot \\ \times \end{Bmatrix}$$

Gauss Elimination: Pivoting

Reduction Step k

$$
\left.
\begin{aligned}
m_{ik} &= \frac{a_{ik}^{(k)}}{a_{kk}^{(k)}} \\
a_{ij}^{(k+1)} &= a_{ij}^{(k)} - m_{ik} a_{kj}^{(k)}, \quad j = k, \cdots n \\
b_i^{(k+1)} &= b_i^{(k)} - m_{ik} b_k^{(k)}
\end{aligned}
\right\} \; i = 2, \cdots n
$$

Partial Pivoting by Columns:
i.e. pivot is chosen with each column

New Row k

New Row i

$$
\begin{bmatrix}
\times & & & & & & \\
0 & \times & & & & & \\
\cdot & \cdot & \times & & & & \\
\cdot & & 0 & \times & \times & \times & \times \\
\cdot & & \cdot & \times & & & \\
\cdot & & \cdot & 0 & & & \\
\cdot & & \cdot & \times & & & \\
0 & \cdot & 0 & \times & & &
\end{bmatrix}
\cdot \bar{\mathbf{x}} =
\begin{Bmatrix}
\times \\ \cdot \\ \cdot \\ \times \\ \cdot \\ \cdot \\ \cdot \\ \times
\end{Bmatrix}
$$

Pivot Elements

$$a_{11}^{(1)}, a_{22}^{(2)}, \ldots, a_{nn}^{(n)}$$

Required at each step!

$$\boxed{a_{kk}^{(k)} \neq 0}$$

Two Solutions:

A. **Partial Pivoting**
   i. Search for largest available coefficient in column below pivot element
   ii. Switch rows k and i

B. **Complete Pivoting**
   i. As for Partial, but search both rows and columns
   ii. Rarely done since column re-ordering changes order of x's, hence more complex code

Gauss Elimination: Pivoting Example
(for division by zero but also reduces round-off errors)

Example, n=2

$$\begin{bmatrix} 0.01 & -1.0 \\ 1.0 & 0.01 \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \end{Bmatrix} = \begin{Bmatrix} 1.0 \\ 1.0 \end{Bmatrix}$$

Cramer's Rule - Exact

$$x_1 = \frac{1.0 \cdot 0.01 - 1.0 \cdot (-1.0)}{0.01 \cdot 0.01 - 1.0 \cdot (-1.0)} = 1.0099$$

$$x_2 = \frac{1.0 \cdot 0.01 - 1.0 \cdot 1.0}{0.01 \cdot 0.01 - 1.0 \cdot (-1.0)} = -0.9899$$

Relatively close to zero

Direct Gaussian Elimination, <u>no pivoting</u>

$$\begin{bmatrix} 0.01 & -1.0 \\ 1.0 & 0.01 \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \end{Bmatrix} = \begin{Bmatrix} 1.0 \\ 1.0 \end{Bmatrix} \Rightarrow \begin{cases} x_1 = 1.01 \\ x_2 = -0.99 \end{cases}$$

2-digit Arithmetic

100% error

1% error

$$\begin{aligned} m_{21} &= 100 \\ a_{21}^{(2)} &= 0 \\ a_{22}^{(2)} &= \boxed{0.01} + 10\boxed{0} \simeq 100 \\ b_2^{(2)} &= \boxed{1} - 10\boxed{0} \simeq -100 \\ x_2 &= \boxed{-1} \\ x_1 &= (1.0 - 1.0)/0.01 = \boxed{0} \end{aligned}$$

tbt.m

```
n=2
a = [ [0.01 1.0]' [-1.0 0.01]']
b= [1 1]'
r=a^(-1) * b
x=[0 0];
m21=a(2,1)/a(1,1);
a(2,1)=0;
a(2,2)  = radd(a(2,2),-m21*a(1,2),n);
b(2)    = radd(b(2),-m21*b(1),n);
x(2)    = b(2)/a(2,2);
x(1)    = (radd(b(1), -a(1,2)*x(2),n))/a(1,1);
x'
```

Gauss Elimination: Pivoting Example
(for division by zero but also reduces round-off errors)

Example, n=2

$$\begin{bmatrix} 0.01 & -1.0 \\ 1.0 & 0.01 \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \end{Bmatrix} = \begin{Bmatrix} 1.0 \\ 1.0 \end{Bmatrix}$$

Cramer's Rule - Exact

$$x_1 = \frac{1.0 \cdot 0.01 - 1.0 \cdot (-1.0)}{0.01 \cdot 0.01 - 1.0 \cdot (-1.0)} = 1.0099$$

$$x_2 = \frac{1.0 \cdot 0.01 - 1.0 \cdot 1.0}{0.01 \cdot 0.01 - 1.0 \cdot (-1.0)} = -0.9899$$

Partial Pivoting

Interchange Rows

$$\begin{bmatrix} 1.0 & 0.01 \\ 0.01 & -1.0 \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \end{Bmatrix} = \begin{Bmatrix} 1.0 \\ 1.0 \end{Bmatrix}$$

2-digit Arithmetic

$$\begin{aligned} m_{21} &= 0.01 \\ a_{22}^{(2)} &= -1 - 0.0\boxed{001} \simeq -1.0 \\ b_2^{(2)} &= 1 - 0.0\boxed{1} \simeq 1.0 \\ x_2 &= \boxed{-1} \\ x_1 &= 1 + 0.01 \simeq \boxed{1.0} \end{aligned}$$

1% error

1% error

See tbt2.m

Notes on coding:

- Pivoting can be done in function/subroutine
- Most codes don't exchange rows, but rather keep track of pivot rows (store info in "pointer" vector)

Gauss Elimination: Equation Scaling Example
(normalizes determinant, also reduces round-off errors)

Example, n=2

$$\begin{bmatrix} 0.01 & -1.0 \\ 1.0 & 0.01 \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \end{Bmatrix} = \begin{Bmatrix} 1.0 \\ 1.0 \end{Bmatrix}$$

Cramer's Rule - Exact

$$x_1 = \frac{1.0 \cdot 0.01 - 1.0 \cdot (-1.0)}{0.01 \cdot 0.01 - 1.0 \cdot (-1.0)} = 1.0099$$

$$x_2 = \frac{1.0 \cdot 0.01 - 1.0 \cdot 1.0}{0.01 \cdot 0.01 - 1.0 \cdot (-1.0)} = -0.9899$$

Multiply Equation 1 by 200:

this solves division by 0, but eqns. not scaled anymore!

$$\begin{bmatrix} 2.0 & -200 \\ 1.0 & 0.01 \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \end{Bmatrix} = \begin{Bmatrix} 200.0 \\ 1.0 \end{Bmatrix} \Rightarrow \begin{cases} x_1 = 1.01 \\ x_2 = -0.99 \end{cases}$$

2-digit Arithmetic

$$\begin{aligned} m_{21} &= 0.5 \\ a_{21}^{(2)} &= 0 \\ a_{22}^{(2)} &= 0.01 + 100 \simeq 100 \\ b_2^{(2)} &= 1 - 0.5 \cdot 200 \simeq -100 \\ x_2 &= \boxed{-1} \\ x_1 &= (200 - 200)/2 = \boxed{0} \end{aligned}$$

100% error

1% error

See tbt3.m

Equations must be normalized for partial pivoting to ensure stability

This **Equilibration** is made by normalizing the matrix to unit norm

Row-based Infinity-norm Normalization

$$||a_{ij}||_\infty = \max_j |a_{ij}| \simeq 1, \quad i = 1, \dots n$$

Row-based 2-norm Normalization

$$||a_{ij}||_2 = \sum_{j=1}^{n} a_{ij}^2 \simeq 1, \quad i = 1, \dots n$$

Examples of Matrix Norms

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

"Maximum Column Sum"

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

"Maximum Row Sum"

$$\|A\|_F = \left(\sum_{i=1}^{m} \sum_{j=1}^{n} |a_{ij}|^2 \right)^{1/2}$$

"The Frobenius norm" (also called Euclidean norm)", which for matrices differs from:

$$\|A\|_2 = \sqrt{\lambda_{\max}(A^* A)}$$

"The l-2 norm" (also called spectral norm)

Gauss Elimination: Full Pivoting Example

(also reduces round-off errors)

Pivoting searches both rows and columns

Start from system where
eq. 1 multiplied by 200:

Example, n=2

pivot chosen
within each row,
across all columns

$$\begin{bmatrix} 2.0 & -200 \\ 1.0 & 0.01 \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \end{Bmatrix} = \begin{Bmatrix} 200.0 \\ 1.0 \end{Bmatrix}$$

Cramer's Rule - Exact

$$x_1 = \frac{1.0 \cdot 0.01 - 1.0 \cdot (-1.0)}{0.01 \cdot 0.01 - 1.0 \cdot (-1.0)} = 1.0099$$

$$x_2 = \frac{1.0 \cdot 0.01 - 1.0 \cdot 1.0}{0.01 \cdot 0.01 - 1.0 \cdot (-1.0)} = -0.9899$$

Interchange Unknowns

$$x_1 = \tilde{x}_2$$

$$x_2 = \tilde{x}_1$$

$$\begin{bmatrix} -200 & 2.0 \\ 0.01 & 1.0 \end{bmatrix} \begin{Bmatrix} \tilde{x}_1 \\ \tilde{x}_2 \end{Bmatrix} = \begin{Bmatrix} 200.0 \\ 1.0 \end{Bmatrix} \Rightarrow \begin{Bmatrix} \tilde{x}_1 = -0.99 \\ \tilde{x}_2 = 1.01 \end{Bmatrix}$$

2-digit Arithmetic

$$\begin{aligned} m_{21} &= -0.00005 \\ a_{21}^{(2)} &= 0 \\ a_{22}^{(2)} &= 0.01 + 1.0 \simeq 1.0 \\ b_2^{(2)} &= 1 + 0.01 \simeq 1 \\ \tilde{x}_2 &\simeq \boxed{1} \\ \tilde{x}_1 &= (200 - 2)/(-200) \simeq \boxed{-1} \end{aligned}$$

1% error

Full Pivoting

Find largest numerical value in eligible rows and columns, and interchange

Affects ordering of unknowns (hence rarely done)

Gauss Elimination

Numerical Stability

- Partial Pivoting
  - Equilibrate system of equations (Normalize or scale variables)
  - Pivoting within columns
  - Simple book-keeping
    - Solution vector in original order
- Full Pivoting
  - Does not necessarily require equilibration
  - Pivoting within both row and columns
  - More complex book-keeping
    - Solution vector re-ordered

Partial Pivoting is simplest and most common

Neither method guarantees stability due to large number of recursive computations (round-off error)

Gauss Elimination:
Effect of variable transform (variable scaling)

Example, n=2

$$\begin{bmatrix} 0.01 & -1.0 \\ 1.0 & 0.01 \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \end{Bmatrix} = \begin{Bmatrix} 1.0 \\ 1.0 \end{Bmatrix}$$

Cramer's Rule - Exact

$$\begin{aligned} x_1 &= \frac{1.0 \cdot 0.01 - 1.0 \cdot (-1.0)}{0.01 \cdot 0.01 - 1.0 \cdot (-1.0)} = \quad 1.0099 \\ x_2 &= \frac{1.0 \cdot 0.01 - 1.0 \cdot 1.0}{0.01 \cdot 0.01 - 1.0 \cdot (-1.0)} = -0.9899 \end{aligned}$$

Variable Transformation

$$\begin{aligned} x_1 &= \tilde{x}_1 \\ x_2 &= 0.01 \cdot \tilde{x}_2 \end{aligned}$$

See tbt4.m

$$\begin{bmatrix} 1.0 & -1.0 \\ 1.0 & 0.0001 \end{bmatrix} \begin{Bmatrix} \tilde{x}_1 \\ \tilde{x}_2 \end{Bmatrix} = \begin{Bmatrix} 100.0 \\ 1.0 \end{Bmatrix} \Rightarrow \begin{cases} \tilde{x}_1 = 1.01 \\ \tilde{x}_2 = -99 \end{cases}$$

2-digit Arithmetic

1% error

100% error

$$\begin{aligned} m_{21} &= 1.0 \\ a_{21}^{(2)} &= 0 \\ a_{22}^{(2)} &= 0.0001 + 1.0 \simeq 1.0 \\ b_2^{(2)} &= 1 - 100 \simeq -100 \\ \tilde{x}_2 &= \boxed{-100} \\ \tilde{x}_1 &= 100 - 100 = \boxed{0} \end{aligned}$$

Systems of Linear Equations
Gauss Elimination

How to Ensure Numerical Stability

- System of equations must be well conditioned
  - Investigate condition number
    - Tricky, because it can require matrix inversion (as we will see)
  - Consistent with physics
    - e.g. don't couple domains that are physically uncoupled
  - Consistent units
    - e.g. don't mix meter and μm in unknowns
  - Dimensionless unknowns
    - Normalize all unknowns consistently
- Equilibration and Partial Pivoting, or Full Pivoting

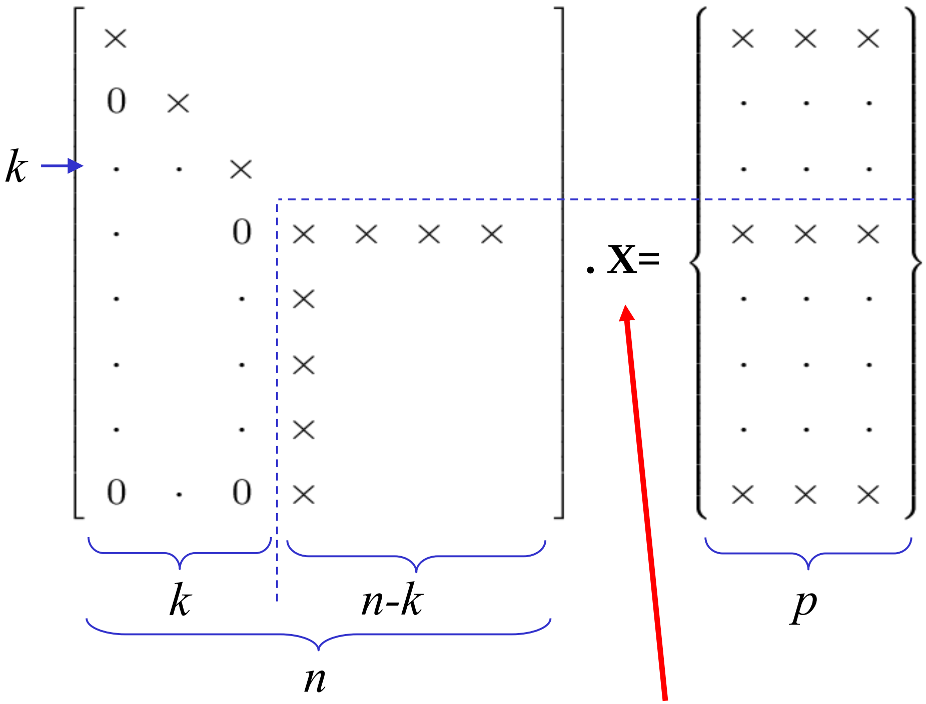
Special Applications of Gauss Elimination

- **Complex Systems**
  - Replace all numbers by complex ones, or,
  - Re-write system of n complex equations into $2n$ real equations
- **Nonlinear Systems of equations**
  - Newton-Raphson: 1st order term kept, use 1st order derivatives
  - Secant Method: Replace 1st order derivatives with finite-difference
  - In both cases, at each iteration, this leads to a linear system, which can be solved by Gauss Elimination (if full system)
- **Gauss-Jordan**: variation of Gauss Elimination
  - Elimination
    - Eliminates each unknown completely (both below and above the pivot row) at each step
    - Normalizes all rows by their pivot
  - Elimination leads to diagonal unitary matrix (identity): no back-substitution needed
  - Number of Ops: about 50% more expensive than Gauss Elimination ($n^3/2$ vs. $n^3/3$ multiplications/divisions)

Gauss Elimination: Multiple Right-hand Sides

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

Reduction
Step k



$\mathbf{X}$ is a $[n\ x\ p]$ matrix

Total Computation Count = ?

Reduction: Nr

Back Substitution: Nb

If $n >> p$, we expect Nr >> Nb

But, if $n \sim p$? (next slide)

Gauss Elimination: Multiple Right-hand Sides
Number of Ops

Reduction/Elimination: Step k

$$\left.\begin{aligned} m_{ik} &= \frac{a_{ik}^{(k)}}{a_{kk}^{(k)}} \\ a_{ij}^{(k+1)} &= a_{ij}^{(k)} - m_{ik} a_{kj}^{(k)}, \quad j = k, \cdots n \\ b_i^{(k+1)} &= b_i^{(k)} - m_{ik} b_k^{(k)} \end{aligned}\right\} i = k+1, \cdots n$$

: n-k divisions

: 2 (n-k) (n-k+1) additions/multiplications

: 2 (n-k) p additions/multiplication

p equations as this one

For reduction, the number of ops is: $\sum_{k=1}^{n-1} (2p+1)(n-k) + 2(n-k)*(n-k+1) =$

$$(2p+1)\frac{n(n-1)}{2} + \frac{2n(n^2-1)}{3} = O(n^3 + pn^2)$$

Back-Substitution

$$x_k = \left(b_k^{(k)} - \sum_{j=k+1}^{n} a_{kj}^{(k)} x_j \right) / a_{kk}^{(k)}$$

: p * ((n-k-1)+(n-k)+2)= p * (2(n-k) +1) add./mul./div.

Number of ops for back-substitution: $p + p\sum_{k=1}^{n-1} 2(n-k)+1 = p + p(n-1)(n+1) = p\,n^2$

(the first *p* before the sum is for the evaluations of the *p* $x_{n's}$)

Grand total number of ops is $O(n^3 + p\,n^2)$: note, extra reduction/elimination only for RHS

Gauss Elimination: Multiple Right-hand Sides
Number of Ops, Cont'd

Reduction
at end of step k

$$\begin{array}{c} \\ \\ k \rightarrow \\ \\ \\ \\ \\ \\ \end{array} \underbrace{\underbrace{\begin{matrix} \times & & \\ 0 & \times & \\ \cdot & \cdot & \times \\ \cdot & & 0 \\ \cdot & & \cdot \\ \cdot & & \cdot \\ \cdot & & \cdot \\ 0 & \cdot & 0 \end{matrix}}_{k} \; \underbrace{\begin{matrix} & & & \\ & & & \\ & & & \\ \times & \times & \times & \times \\ \times & & & \\ \times & & & \\ \times & & & \\ \times & & & \end{matrix}}_{n-k}}_{n} \cdot \bar{\mathbf{x}} = \underbrace{\begin{Bmatrix} \times & \times & \times \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ \times & \times & \times \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ \times & \times & \times \end{Bmatrix}}_{p}$$

i. Repeating reduction/elimination of A for each RHS would be inefficient if p >>>

ii. However, if RHS is result of iterations and unknown a priori, it may seem one needs to redo the Reduction each time

$\mathbf{A}\,\mathbf{x}_1 = \mathbf{b}_1$, $\mathbf{A}\,\mathbf{x}_2 = \mathbf{b}_2$, etc, where vector $\mathbf{b}_2$ is a function of $\mathbf{x}_1$, etc

=> LU Factorization / Decomposition of A