Chemical Engineering 541

Numerical Methods

Direct Solution of Linear Systems

Outline

- Gauss Elimination
  - Pivoting
  - Scaling
  - Cost
- LU Decomposition
- Thomas Algorithm
- (Iterative Improvement)

Overview

- Two types of solutions to linear systems of equations

1. Direct Methods
   - "Analytic" solution of the system ("exact" solution)
   - Algorithms work to
     1) Reduce operations,
     2) Minimize roundoff error
2. Iterative Methods
   1. Approximate solutions to some accuracy

Direct Methods

- Gauss Elimination (GE), LU Decomposition (LU), Thomas Algorithm (TA), QR Decomposition (QR)
- Use when:

1) Small number of equations (~100 or less)
    - Cost scales as n^3, so minimize n
    - Roundoff errors increase with number of operations.
2) Dense Matricies
    - These usually arise in smaller problems anyway
3) Systems that are not diagonally dominant (DD)
    - Practical, large systems are DD $\quad |a_{ii}| \geq \sum_i |a_{ij}|$
4) Ill-conditioned systems
    - (still subject to roundoff error, but not iterative convergence)

Gauss Elimination

$$\begin{bmatrix} a & a & a \\ a & a & a \\ a & a & a \end{bmatrix} \cdot \begin{bmatrix} x \\ x \\ x \end{bmatrix} = \begin{bmatrix} b \\ b \\ b \end{bmatrix} \Rightarrow \begin{bmatrix} a' & a' & a' \\ 0 & a' & a' \\ 0 & 0 & a' \end{bmatrix} \cdot \begin{bmatrix} x \\ x \\ x \end{bmatrix} = \begin{bmatrix} b' \\ b' \\ b' \end{bmatrix}$$

1) Eliminate to upper triangular form
2) Back substitution to Solution.

Elimination

- Work through columns k=1,n-1
  - Put 0's in rows i=k+1 to n of column k
    - Subtract *multiples* of row k from row i
    - Operate on A, b (b as if elements were in A: augmented matrix)

G.E. Algorithm

Elimination

k=1 k=2 k=3

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix}$$

loop: k=1

$$R_2 = R_2 - \frac{R_1 * a_{21}}{a_{11}}$$

$$R_3 = R_3 - \frac{R_1 * a_{31}}{a_{11}}$$

$$R_4 = R_4 - \frac{R_1 * a_{41}}{a_{11}}$$

k=2

$$R_3 = R_3 - \frac{R_2 * a_{32}}{a_{22}}$$

$$R_4 = R_4 - \frac{R_2 * a_{42}}{a_{22}}$$

k=3

$$R_4 = R_4 - \frac{R_3 * a_{43}}{a_{33}}$$

```
loop over cols:          do k=1,n-1
    loop over rows:          do i=k+1,n
        find fac:                fac = a(i,k)/a(k,k)
                                 do j=k+1,n
        loop row elems:              a(i,j)=a(i,j)-fac*a(k,j)
                                 end
        do b:                    b(i) = b(i) - fac*b(k)
                             end
                         end
```

*Cost ~ 2/3*n*^{3}

G.E. Algorithm

Back Substitution

$$\begin{bmatrix} a_{11} & a'_{12} & a'_{13} & a'_{14} \\ 0 & a'_{22} & a'_{23} & a'_{24} \\ 0 & 0 & a'_{33} & a'_{34} \\ 0 & 0 & 0 & a'_{44} \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} b'_1 \\ b'_2 \\ b'_3 \\ b'_4 \end{bmatrix}$$

$$x_4 = \frac{b_4}{a_{44}} \qquad x_3 = \frac{b_3 - x_4 a_{34}}{a_{33}} \qquad x_2 = \frac{b_2 - x_4 a_{24} - x_3 a_{23}}{a_{22}} \qquad x_1 = \frac{b_1 - x_4 a_{14} - x_3 a_{13} - x_2 a_{12}}{a_{11}}$$

do it (for x_4)

Loop with inner loop in the numerator (for x_3, x_2, x_1)

```
x(n) = b(n)/a(n,n)       (do first)
do i=n-1,1,-1            (bottom up)
    x(i) = b(i)
    do j=i+1,n           (loop numrtr)
        x(i) = x(i) - x(j)*a(i,j)
    end
    x(i) = x(i) / a(i,i)
end
```

Cost ~ n^2

Pivoting

- G.E. fails for 0 on the pivot (diagonal)
  - this causes a division by zero
- Avoid by interchanging rows
  - partial pivoting
- Also use for numerical precision to minimize roundoff errors
  - Don't divide small #'s:
    - R_2-R_1*a_{21}/a_{11}.
    - If a_{11} small → R_2-big → lose precision.
    - Swap rows to get largest element in the pivot position

See Chapra and Canale 6th edition, Example 9.9

- Algorithmically, don't actually swap rows, use and index array:
  - pos = (1 2 3 4) → (4 2 3 1) A[i,j] → A[p[i],j]
  - Avoid recopying memory, easy to program
- Partial pivoting cost ~n^2, full pivoting cost ~n^3

Scaling

- Scaling is needed to select the pivot elements
  - To compare rows (pivot element) need rows to be of the same "size"
  - Divide each possible pivot by the max row element.
    - Scale each row to have a max of one
  - Use scaling *just* to select pivot

$$\begin{matrix} R_1 \\ R_2 \end{matrix} \begin{bmatrix} 5 & 100 \\ 1 & 1 \end{bmatrix} \rightarrow \begin{matrix} 5 \\ 1 \end{matrix} \rightarrow \begin{matrix} 5/100 \\ 1/1 \end{matrix} \rightarrow \begin{matrix} 0.05 \\ 1 \end{matrix} \rightarrow \begin{bmatrix} 0.05 & 1 \\ 1 & 1 \end{bmatrix}$$

At first sight, use R_1, Scaling → use R_2

LU Decomposition

- Closely related to G.E.
- G.E. produces LU decomposition implicitly
- Useful if multiple b vectors are to be evaluated
- $Ax=b \rightarrow LUx=b \rightarrow L(Ux)=b \rightarrow Ly=b$
  - Forward substitution to get y (easy)
  - Solve $Ux=y$ for x with back substitution (easy)

LU Algorithm

$$\begin{matrix} R_1: \\ R_2: \\ R_3: \end{matrix} \begin{bmatrix} 1 & 0 & 0 \\ l_{21} & 1 & 0 \\ l_{31} & l_{32} & 1 \end{bmatrix} \begin{bmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

$$\begin{array}{llll} R_1: & u_{11} = a_{11} & u_{12} = a_{12} & u_{13} = a_{13} \\ R_2: & l_{21}u_{11} = a_{21} & l_{21}u_{12} + u_{22} = a_{22} & l_{21}u_{13} + u_{23} = a_{23} \\ R_3: & l_{31}u_{11} = a_{31} & l_{31}u_{12} + l_{32}u_{22} = a_{32} & l_{31}u_{13} + l_{32}u_{23} + u_{33} = a_{33} \end{array}$$

$$\begin{array}{llll} R_1: & u_{11} = a_{11} & u_{12} = a_{12} & u_{13} = a_{13} \\ R_2: & l_{21} = \frac{a_{21}}{u_{11}} & u_{22} = a_{22} - l_{21}u_{12} & u_{23} = a_{23} - l_{21}u_{13} \\ R_3: & l_{31} = \frac{a_{31}}{u_{11}} & l_{32} = \frac{a_{32} - l_{31}u_{12}}{u_{22}} & u_{33} = a_{33} - l_{31}u_{13} - l_{32}u_{23} \end{array}$$

- l's are G.E. Factors
- u's are from G.E. row operations
- Algorithm works "in place": LU occupies same space as A
- G.E. Alg. changes:
  - don't consider b on factorization
  - Add in l_{ij} replacement

GE, LU Relationship

GE equations

$$\begin{matrix} R_1 \\ R_2 \\ R_3 \end{matrix} \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

$$R_2 = R_2 - \left(\frac{a_{21}}{a_{11}}\right) R_1$$

$$R_3 = R_3 - \left(\frac{a_{31}}{a_{11}}\right) R_1$$

$$R_3 = R_3 - \left(\frac{a_{32}}{a_{22}}\right) R_2$$

a_{11}	a_{12}	a_{13}
$a_{21} = 0$	$a'_{22} \leftarrow a_{22} - \left(\frac{a_{21}}{a_{11}}\right) a_{12}$	$a'_{23} \leftarrow a_{23} - \left(\frac{a_{21}}{a_{11}}\right) a_{13}$
$a_{31} = 0$	$a'_{32} \leftarrow a_{32} - \left(\frac{a_{31}}{a_{11}}\right) a_{12}$ $a'_{32} = 0$	$a'_{33} \leftarrow a_{33} - \left(\frac{a_{31}}{a_{11}}\right) a_{13}$ $a''_{33} \leftarrow a'_{33} - \left(\frac{a'_{32}}{a'_{22}}\right) a'_{23}$

GE → LU equations

u_{11}	u_{12}	u_{13}
l_{21}	$u_{22} = a_{22} - \underbrace{\left(\frac{a_{21}}{a_{11}}\right)}_{l_{21}} u_{12}$	$a'_{33} = a_{33} - \underbrace{\left(\frac{a_{31}}{a_{11}}\right)}_{l_{31}} u_{13}$
l_{31}	$a'_{32} = a_{32} - \underbrace{\left(\frac{a_{21}}{a_{11}}\right)}_{l_{31}} u_{12}$ l_{32}	$a'_{33} = a_{33} - \underbrace{\left(\frac{a_{21}}{a_{11}}\right)}_{l_{31}} u_{13}$ $u_{33} = a'_{33} - \underbrace{\left(\frac{a'_{32}}{a'_{22}}\right)}_{l_{32}} u_{23}$

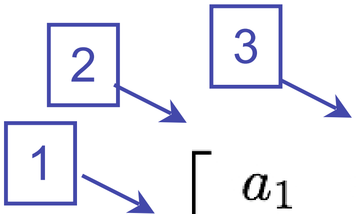
Recall

$$l_{32} = \frac{a_{32} - l_{31}u_{12}}{u_{22}} \qquad u_{33} = a_{33} - l_{31}u_{13} - l_{32}u_{23}$$

Thomas Algorithm

- G.E. applied to a tridiagonal system
  - TDMA
- Common in Differential Equations
  - Especially diffusion problems
- No pivoting
  - pivoting would destroy the structure
  - no need to pivot in practice due to natural structure.
- Cost ~ n
  - Very fast algorithm

Thomas Algorithm

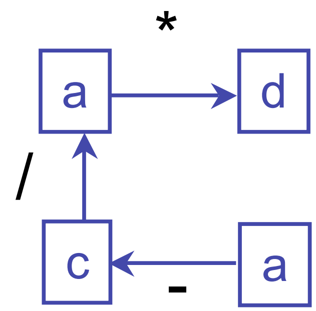


$$\begin{bmatrix} a_1 & d_1 & 0 & 0 & 0 \\ c_2 & a_2 & d_2 & 0 & 0 \\ 0 & c_3 & a_3 & d_3 & 0 \\ 0 & 0 & c_4 & a_4 & d_4 \\ 0 & 0 & 0 & c_5 & a_5 \end{bmatrix}$$

$$a_1 = a_1$$

$$a_2 = a_2 - \frac{c_2}{a_1} d_1$$

$$a_3 = a_3 - \frac{c_3}{a_2} d_2$$



- Store 3, 1D arrays
- G.E. → array *c* becomes zeros, so ignore
- G.E. → array *d* not changes cause zeros above, so ignore
- Just change array *a*

Elimination

$$a_i = a_i - \frac{c_i}{a_{i-1}} d_{i-1} \qquad i = 2, n$$

$$b_i = b_i - \frac{c_i}{a_{i-1}} b_{i-1} \qquad i = 2, n$$

Back Substitution

$$x_i = \frac{b_i}{a_i} \qquad i = n$$

$$x_i = \frac{b_i - d_i x_{i+1}}{a}{}_i \qquad i = n-1, 1, -1$$