

Chapter Two Numerical Solution of Systems of Linear Equations

(MA6624)

In this chapter, we consider the linear system

$$Ax = b.$$

where $A = (a_{ij})_{i,j=1}^n$ is an $n \times n$ matrix and b is an n -dimensional vector. We always assume that A is nonsingular, *i.e.*, $\det(A) \neq 0$.

2.1 Gaussian elimination method

Example 1: Solve the system $Ax = b$ where

$$A = \begin{bmatrix} 2 & -4 & -2 \\ -2 & 8 & -2 \\ -1 & -3 & 10 \end{bmatrix} \quad b = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

Use row-operation on the augmented matrix

$$\begin{aligned} [A, b] &= \begin{bmatrix} 2 & -4 & -2 & 1 \\ -2 & 8 & -2 & 1 \\ -1 & -3 & 10 & 1 \end{bmatrix} \xrightarrow{\substack{R_2 : R_2 - \frac{-2}{2}R_1 \\ R_3 : R_3 - \frac{-1}{2}R_1}} \begin{bmatrix} 2 & -4 & -2 & 1 \\ 0 & 4 & -4 & 2 \\ 0 & -5 & 9 & 3/2 \end{bmatrix} \\ &\xrightarrow{R_3 : R_3 - \frac{-5}{4}R_2} \begin{bmatrix} 2 & -4 & -2 & 1 \\ 0 & 4 & -4 & 2 \\ 0 & 0 & 4 & 4 \end{bmatrix} \end{aligned}$$

to get the upper triangular matrix and then, use *back substitution* to get a solution

$$x_3 = 1 \quad x_2 = 3/2 \quad x_1 = 9/2$$

Example 2: Solve the system $Ax = b$ where

$$A = \begin{bmatrix} 0 & 1 & 3 \\ 1 & -1 & 1 \\ 3 & -1 & 10 \end{bmatrix} \quad b = \begin{bmatrix} 4 \\ 1 \\ 12 \end{bmatrix}$$

Augmented matrix

$$\begin{aligned} [A, b] &= \begin{bmatrix} 0 & 1 & 3 & 4 \\ 1 & -1 & 1 & 1 \\ 3 & -1 & 10 & 12 \end{bmatrix} \xrightarrow{R_1 \leftrightarrow R_2} \begin{bmatrix} 1 & -1 & 1 & 1 \\ 0 & 1 & 3 & 4 \\ 3 & -1 & 10 & 12 \end{bmatrix} \\ &\xrightarrow{R_3 : R_3 - 3R_1} \begin{bmatrix} 1 & -1 & 1 & 1 \\ 0 & 1 & 3 & 4 \\ 0 & 2 & 7 & 9 \end{bmatrix} \xrightarrow{R_3 : R_3 - 2R_2} \begin{bmatrix} 1 & -1 & 1 & 1 \\ 0 & 1 & 3 & 4 \\ 0 & 0 & 1 & 1 \end{bmatrix} \end{aligned}$$

and then, by using back substitution,

$$x_3 = 1 \quad x_2 = 1 \quad x_1 = 1$$

There are three row operations:

- (i) interchange two rows.
- (ii) multiply a row by a constant.
- (iii) add the row i multiplying by a constant in the row j .

If the matrix A is nonsingular, we always get the exact solution of a linear system by using these row operations and back substitution. This is so-called Gaussian elimination. Computers can follow the routine.

Let $A = (a_{ij})$. In general, there are $n - 1$ stages for Gaussian elimination, with $A^{(1)} := A$, $b^{(1)} := b$, and finishing with the upper triangular system $A^{(n)}x = b^{(n)}$.

At the k th stage, we have converted the original system to $A^{(k)}x = b^{(k)}$, where

$$A^{(k)} = \begin{bmatrix} A_{11}^{(k)} & A_{12}^{(k)} \\ 0 & A_{22}^{(k)} \end{bmatrix}$$

with $A_{11}^{(k)} \in R^{(k-1) \times (k-1)}$ upper triangular. The purpose of the k th stage of the elimination is to zero the elements below the diagonal in the k th column of $A^{(k)}$. This is accomplished by the operations

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

where the multipliers $m_{ik} = a_{ik}^{(k)} / a_{kk}^{(k)}$, $i = k + 1 : n$. At the end of the $(n - 1)$ th stage, we have the upper triangular system $A^{(n)}x = b^{(n)}$, which is solved by back substitution.

Remarks

- (i) For an $n \times n$ matrix, Gaussian elimination requires $\frac{1}{3}n^3$ operations.
- (ii) In general, it requires $O(n^2)$ memory.
- (iii) Stability.

There are two problems with the method as described. First, there is break-down with division by zero if $a_{kk}^{(k)} = 0$. Second, if we are working in finite precision, the round off error is too large to obtain the exact solution.

Example 3 Solve the following linear system by Gaussian elimination method in a 4-digit computer.

$$\begin{aligned} 0.0030x_1 + 59.14x_2 &= 59.17 \\ 5.291x_1 - 6.130x_2 &= 46.78 \end{aligned}$$

Augmented matrix

$$\begin{aligned} [A, b] &= \begin{bmatrix} 0.3000E - 2 & 0.5914E + 2 & 0.5917E + 2 \\ 0.5291E + 1 & -0.6130E + 1 & 0.4678E + 2 \end{bmatrix} \\ &\xrightarrow{R_2 : R_2 - \left(\frac{0.5291E1}{0.3000E - 2} \right) R_1} \begin{bmatrix} 0.3000E - 2 & 0.5914E2 & 0.5917E2 \\ 0 & -0.1043E6 & -0.1044E6 \end{bmatrix} \end{aligned}$$

By back substitution,

$$x_2 = 0.1001E1 \quad x_1 = -0.1000E2$$

while the exact solution is

$$x_2 = 0.1000E1 \quad x_1 = 0.1000E2$$

Obviously, these row operations enlarge the round off error. If we interchange the rows of augmented matrix firstly, and use Gaussian elimination, we obtain

$$\left[\begin{array}{ccc} 0.5291E1 & -0.6130E1 & 0.4678E2 \\ 0.3000E-2 & 0.5914E2 & 0.5917E2 \end{array} \right] \implies \left[\begin{array}{ccc} 0.5291E1 & -0.6130E1 & 0.4678E2 \\ 0 & 0.5914E2 & 0.59147E2 \end{array} \right]$$

By back substitution, we have

$$x_2 = 1 \quad x_1 = 10.0$$

This technique is called *Pivoting*. All computer codes based on Gaussian elimination need to do PIVOTING, *i.e.*, interchange two rows when a_{ii} is too small compared with other entries.

The pivoting is not needed if

* A is symmetric positive definite, or

* A is diagonally dominant

Partial pivoting (row pivoting, maximal column pivoting)

At the start of the k th stage, the k th and r th rows are interchanged, where

$$|a_{rk}^{(k)}| = \max_{k \leq i \leq n} |a_{ik}^{(k)}|$$

In computer, we only keep the index vector K . Assuming the initial

$$K = (1, 2, \dots, n)^T,$$

one will update the K . See the example above.

Scaled partial pivoting

Let

$$s_i^{(k)} = \max_{k \leq j \leq n} |a_{ij}^{(k)}|.$$

For the k th column $a_{ik}^{(k)}$, we choose p such that

$$\frac{a_{pk}^{(k)}}{s_p^{(k)}} = \max_{k \leq i \leq n} \frac{|a_{ik}^{(k)}|}{s_i^{(k)}},$$

and interchange the p th row and the k th row.

Example 4

$$30.00x_1 + 591400x_2 = 591700$$

$$5.291x_1 - 6.130x_2 = 46.78$$

$$s_1^{(1)} = \max \{|30.00|, |591400|\} = 591400$$

$$s_2^{(1)} = \max \{|5.291|, |-6.130|\} = 6.130$$

then $\frac{|a_{11}^{(1)}|}{s_1^{(1)}} = 0.5073 \times 10^{-4}$, $\frac{|a_{21}^{(1)}|}{s_2^{(1)}} = 0.8631$, Thus, we have $p = 2$.

Complete pivoting(total pivoting)

At the start of the k th stage, row k and r and columns k and s are interchanged, where

$$|a_{rs}^{(k)}| = \max_{k \leq i, j \leq n} |a_{ij}^{(k)}|$$

Example 5:

$$A = \begin{bmatrix} 2 & 3 & -6 \\ 1 & -6 & 8 \\ 3 & -2 & 8 \end{bmatrix}$$

step 1: $\max_{1 \leq i, j \leq 3} \{|a_{ij}^{(1)}|\} = |a_{23}^{(1)}| = 8$

$$\begin{aligned} \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 2 & 3 & -6 \\ 1 & -6 & 8 \\ 3 & -2 & 8 \end{bmatrix} \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} &= \begin{bmatrix} 8 & -6 & 1 \\ -6 & 3 & 2 \\ 8 & -2 & 3 \end{bmatrix} \\ &\rightarrow \begin{bmatrix} 8 & -6 & 1 \\ 0 & -\frac{3}{2} & \frac{11}{4} \\ 0 & 4 & 2 \end{bmatrix} \end{aligned}$$

step 2: $\max_{2 \leq i, j \leq 3} \{|a_{ij}^{(2)}|\} = |a_{32}^{(2)}| = 4$

$$\rightarrow \begin{bmatrix} 8 & -6 & 1 \\ 0 & 4 & 2 \\ 0 & -\frac{3}{2} & \frac{11}{4} \end{bmatrix} \rightarrow \begin{bmatrix} 8 & -6 & 1 \\ 0 & 4 & 2 \\ 0 & 0 & \frac{7}{2} \end{bmatrix}$$

Remarks:

(i) There are three kinds pivoting. Usually, we use scaled pivoting, since the complete pivoting is complicated for programming and the complete pivoting requires $O(n^3)$ comparisons in total, compared with $O(n^2)$ for partial pivoting. Because of the searching overhead, and because partial pivoting works so well, complete pivoting is used only in special situations.

(ii) Program structure

2.2 LU decomposition

We used row operations to reduce augmented matrices to upper triangle. Mathematically, a row operation is equivalent to left multiplication by an elementary matrix.

Example 6

$$\begin{aligned} A = \begin{bmatrix} 2 & -4 & -2 \\ -2 & 8 & -2 \\ -1 & -3 & 10 \end{bmatrix} &\xrightarrow{\substack{R_2 : R_2 - \frac{-2}{2}R_1 \\ R_3 : R_3 - \frac{-1}{2}R_1}} \begin{bmatrix} 2 & -4 & -2 \\ 0 & 4 & -4 \\ 0 & -5 & 9 \end{bmatrix} \\ &\xrightarrow{R_3 : R_3 - \frac{-5}{4}R_2} \begin{bmatrix} 2 & -4 & -2 \\ 0 & 4 & -4 \\ 0 & 0 & 4 \end{bmatrix} := U \end{aligned}$$

In matrix operation, we have

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 5/4 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1/2 & 0 & 1 \end{bmatrix} A = \begin{bmatrix} 2 & -4 & -2 \\ 0 & 4 & -4 \\ 0 & 0 & 4 \end{bmatrix} := U$$

or

$$M_3 M_2 M_1 A = U$$

where M_i is an elementary matrix and nonsingular. Then

$$A = M_1^{-1} M_2^{-1} M_3^{-1} U$$

and

$$M_3^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -5/4 & 1 \end{bmatrix} \quad M_2^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -1/2 & 0 & 1 \end{bmatrix} \quad M_1^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ -1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Let $L = M_1^{-1} M_2^{-1} M_3^{-1}$. We have

$$L = \begin{bmatrix} 1 & 0 & 0 \\ -1 & 1 & 0 \\ -1/2 & -5/4 & 1 \end{bmatrix}$$

and

$$A = LU$$

be a LU-decomposition of A , where U is upper triangular and L is a unit lower triangle.

If we have a LU decomposition of the matrix A , the system becomes:

$$LUx = b$$

Let $y = Ux$. We have

$$Ly = b$$

Linear solver with LU-decomposition is:

* Do the LU-decomposition $A = LU$

* Solve linear system

$$Ly = b$$

* Solve the system

$$Ux = y$$

Remarks

(i) LU -decomposition is a different form of the Gaussian elimination. The operation count is also $n^3/3$.

(ii) LU -decomposition is more efficient for several linear systems with the same coefficient matrix.

(iii) In computer codes, one usually uses LU decomposition.

(iv) When A is symmetric, we can write

$$U = D\tilde{U}$$

where D is diagonal and \tilde{U} is unit upper triangle. Then

$$A = LD\tilde{U}.$$

Since A is symmetric, we can prove that $\tilde{U} = L^T$ and the decomposition is rewritten by

$$A = LDL^T.$$

If A is symmetric semi-positive definite, $D = \text{diag}(d_1, d_2, \dots, d_n)$ with $d_i > 0$. Let $\tilde{D} = \text{diag}(\sqrt{d_1}, \sqrt{d_2}, \dots, \sqrt{d_n})$. $D = \tilde{D}^2$ and we have the cholesky decomposition

$$A = \tilde{L}\tilde{L}^T$$

where $\tilde{L} = L\tilde{D}$. The memory for Cholesky decomposition is $n^2/2$.

2.3 Tridiagonal linear system

A matrix A is tridiagonal if

$$A = \begin{bmatrix} d_1 & e_1 & 0 & & \\ c_2 & d_2 & e_2 & & \\ 0 & c_3 & d_3 & e_3 & \\ & \ddots & \ddots & \ddots & \\ & & & \ddots & \ddots & e_{n-1} \\ & & & & c_n & d_n \end{bmatrix}$$

We consider the linear system $Ax = b$ where A is tridiagonal, and $b = (b_1, b_2, \dots, b_n)^T$. Here we assume that A is diagonally dominant, *i.e.*,

$$|d_i| \geq |c_i| + |e_i|$$

Algorithm:

```

 $\alpha_1 = d_1$ 
 $v_1 = b_1/\alpha_1$ 
For i=2 to n, do
   $\gamma_{i-1} = e_{i-1}/\alpha_{i-1}$ 
   $\alpha_i = d_i - c_i\gamma_{i-1}$ 
   $v_i = (b_i - c_iv_{i-1})/\alpha_i$ 
end
 $x_n = v_n$ 
For i=1 to n-1, do
   $u_{n-i} = v_{n-i} - \gamma_{n-i}u_{n-i+1}$ 
end

```

Remarks:

- (i) The operation count for multiplications and divisions: $5n$.
- (ii) The memory required: $4n$.
- (iii) The condition: the leading principal submatrices are nonsingular.

2.4 Iterative methods

We consider

$$Ax = b.$$

An iterative method is a method to construct an infinite sequence which is convergent to the solution of linear system. Here we introduce two iterative methods. Let

$$A = D - L - U$$

where D , L and U are diagonal, strictly lower triangular and strictly upper triangular matrices, respectively.

Richardson iteration:

$$\begin{aligned} &\text{For a given initial guess } x^0 \\ x^{k+1} &= (I - \alpha A)x^k + \alpha b \end{aligned}$$

Jacobi iteration:

$$\begin{aligned} &\text{For a given initial guess } x^0 \\ Dx^{n+1} &= (L + U)x^n + b \end{aligned}$$

Gauss-Seidel iteration:

$$\begin{aligned} &\text{For a given initial guess } x^0 \\ (D + L)x^{n+1} &= Ux^n + b \end{aligned}$$

SOR iteration:

$$\begin{aligned} &\text{For a given initial guess } x^0 \\ (D - \omega L)x^{n+1} &= ((1 - \omega)D + \omega U)x^n + \omega b \end{aligned}$$

where ω is a positive number.

Example 7 For the system $Ax = b$, give the first two iterations for Jacobi and Gauss-Seidel methods, respectively, where

$$A = \begin{bmatrix} 10 & -1 & 2 & 0 \\ -1 & 11 & -1 & 3 \\ 2 & -1 & 10 & -1 \\ 0 & 3 & -1 & 8 \end{bmatrix} \quad b = \begin{bmatrix} 6 \\ 25 \\ -11 \\ 15 \end{bmatrix} \quad x^0 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

By the definition of Jacobi iteration and Gauss-Seidel iteration,

$$D = \begin{bmatrix} 10 & 0 & 0 & 0 \\ 0 & 11 & 0 & 0 \\ 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 8 \end{bmatrix} \quad L = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ -2 & 1 & 0 & 0 \\ 0 & -3 & 1 & 0 \end{bmatrix} \quad U = \begin{bmatrix} 0 & 1 & -2 & 0 \\ 0 & 0 & 1 & -3 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$Dx^1 = (L + U)x^0 + b$$

i.e.,

$$\begin{bmatrix} 10 & 0 & 0 & 0 \\ 0 & 11 & 0 & 0 \\ 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 8 \end{bmatrix} x^1 = \begin{bmatrix} 0 & 1 & -2 & 0 \\ 1 & 0 & 1 & -3 \\ -2 & 1 & 0 & 1 \\ 0 & -3 & 1 & 0 \end{bmatrix} x^0 + \begin{bmatrix} 6 \\ 25 \\ -11 \\ 15 \end{bmatrix} = \begin{bmatrix} 6 \\ 25 \\ -11 \\ 15 \end{bmatrix}$$

Then

$$x^1 = \begin{bmatrix} 0.6 \\ 2.2727 \\ -1.1 \\ 1.875 \end{bmatrix}$$

$$Dx^2 = (L + U)x^1 + b$$

i.e.

$$\begin{bmatrix} 10 & 0 & 0 & 0 \\ 0 & 11 & 0 & 0 \\ 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 8 \end{bmatrix} x^2 = \begin{bmatrix} 0 & 1 & -2 & 0 \\ 1 & 0 & 1 & -3 \\ -2 & 1 & 0 & 1 \\ 0 & -3 & 1 & 0 \end{bmatrix} x^1 + \begin{bmatrix} 6 \\ 25 \\ -11 \\ 15 \end{bmatrix} = \begin{bmatrix} 10.473 \\ 18.875 \\ -8.052 \\ 7.082 \end{bmatrix}$$

Then

$$x^2 = \begin{bmatrix} 1.0473 \\ 1.7159 \\ -0.852 \\ 0.8852 \end{bmatrix}$$

(ii) Gauss-Seidel

$$(D - L)x^1 = Ux^0 + b$$

i.e.

$$\begin{bmatrix} 10 & 0 & 0 & 0 \\ -1 & 11 & 0 & 0 \\ 2 & -1 & 10 & 0 \\ 0 & 3 & -1 & 8 \end{bmatrix} x^1 = \begin{bmatrix} 0 & 1 & -2 & 0 \\ 0 & 0 & 1 & -3 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} x^0 + \begin{bmatrix} 6 \\ 25 \\ -11 \\ 15 \end{bmatrix} = \begin{bmatrix} 6 \\ 25 \\ -11 \\ 15 \end{bmatrix}$$

We have

$$x^1 = \begin{bmatrix} 0.6 \\ 2.3172 \\ -0.9873 \\ 0.8789 \end{bmatrix}$$

Also

$$(D - L)x^2 = Ux^1 + b$$

i.e.

$$\begin{bmatrix} 10 & 0 & 0 & 0 \\ -1 & 11 & 0 & 0 \\ 2 & -1 & 10 & 0 \\ 0 & 3 & -1 & 8 \end{bmatrix} x^2 = \begin{bmatrix} 0 & 1 & -2 & 0 \\ 0 & 0 & 1 & -3 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} x^1 + \begin{bmatrix} 6 \\ 25 \\ -11 \\ 15 \end{bmatrix}$$

We have

$$x^1 = \begin{bmatrix} 1.030 \\ 2.037 \\ -1.014 \\ 0.9844 \end{bmatrix}$$

The exact solution of system is:

$$x = (1, 2, -1, 1)^T$$

Remarks These four iterative methods above are basic iterations. More popular one is a so-called Krylov subspace method, including GMRES and BiCG, particularly for non-symmetric matrices.

2.5 Norms of Vectors and Matrices

In order to estimate error $|x_n - x|$, we need to define a measure. When $|x_n - x|$ converges to zero, x_n converges to x .

Vector norms

Definition: A vector norm defined on R^n is a function, $\|\cdot\|$, from R^n to R with the following properties:

- (i) $\|x\| \geq 0$ for all $x \in R^n$,
- (ii) $\|x\| = 0$ if and only if $x = 0$,
- (iii) $\|\alpha x\| = |\alpha| \|x\|$ for all $\alpha \in C$ and $x \in R^n$,
- (iv) $\|x + y\| \leq \|x\| + \|y\|$ for all $x, y \in R^n$.

Several special norms of interest are:

l_1 norm (1-norm)

$$\|x\|_1 = \sum_{i=1}^n |x_i|$$

l_2 norm (2-norm, or Euclidean norm)

$$\|x\|_2 = \sqrt{x^T x} = \sqrt{\sum_{i=1}^n x_i^2} \quad (x \in R^n)$$

l_∞ norm

$$\|x\|_\infty = \max_j |x_j|$$

We can verify these four conditions in the above definition. For example,

(i) if $x \neq 0$, $\|x\|_2 > 0$.

(ii) if $x = 0$, $\|x\|_2 = 0$. If $\|x\|_2 = 0$, since $\|x\|_2 = \sqrt{\sum_{i=1}^n x_i^2} \Rightarrow x_i = 0$, i.e., $x = 0$.

(iii) $\|\alpha x\|_2 = \sqrt{\sum (\alpha x_i)^2} = \sqrt{\alpha^2 \sum x_i^2} = |\alpha| \|x\|_2$.

The fourth condition is the triangular inequality, which can be proved using the following Cauchy-Schwarz inequality.

Lemma (Cauchy–Bunyakovski-Schwarz inequality) Let x and y be two column vectors of length m , then

$$\sum_{i=1}^n x_i y_i \leq \|x\|_2 \|y\|_2$$

Proof: Here we consider the problem in the real field. We can assume that the vectors x and y are non-zero vectors, otherwise, both sides of the above inequality are just zero. Now, for any real λ ,

$$\|x - \lambda y\|_2^2 \geq 0.$$

The left hand side above is actually a quadratic polynomial of λ . We can find the minimum of the left hand side and the minimum should still be non-negative. Since

$$\|x - \lambda y\|_2^2 = (x^T - \lambda y^T)(x - \lambda y) = \|x\|_2^2 - \lambda x^T y - \lambda y^T x + \lambda^2 \|y\|_2^2,$$

the minimum of the above is reached at

$$\lambda = \frac{x^T y}{\|y\|_2^2}.$$

and therefore,

$$0 \leq \min \|x - \lambda y\|_2^2 = \|x\|_2^2 - \frac{(x^T y)^2}{\|y\|_2^2}.$$

This implies $\sum_{i=1}^n x_i y_i \leq \|x\|_2 \|y\|_2$. ■

Theorem 2.1 For any two vectors x and y , the 2-norm satisfies the triangular inequality:

$$\|x + y\|_2 \leq \|x\|_2 + \|y\|_2$$

Proof. Notice that

$$\|x + y\|_2^2 = (x^* + y^*)(x + y) = \|x\|_2^2 + x^* y + y^* x + \|y\|_2^2 \leq \|x\|_2^2 + |x^* y| + |y^* x| + \|y\|_2^2.$$

Now, we use the Cauchy-Schwarz inequalities:

$$|x^* y| \leq \|x\|_2 \|y\|_2, \text{ and } |y^* x| \leq \|x\|_2 \|y\|_2$$

Thus,

$$\|x + y\|_2^2 \leq \|x\|_2^2 + 2\|x\|_2 \|y\|_2 + \|y\|_2^2 = (\|x\|_2 + \|y\|_2)^2.$$

Therefore, $\|x + y\|_2 \leq \|x\|_2 + \|y\|_2$. ■

Matrix norms

Definition: A matrix norm on $R^{n \times m}$ is a function, $\|\cdot\|$, from $R^{n \times m}$ to R with the following properties:

- (i) $\|A\| \geq 0$ for all $A \in R^{n \times m}$,
- (ii) $\|A\| = 0$ if and only if $A = 0$,
- (iii) $\|\alpha A\| = |\alpha| \|A\|$ for all $\alpha \in C$ and $A \in R^{n \times m}$,
- (iv) $\|A + B\| \leq \|A\| + \|B\|$ for all $A, B \in R^{n \times m}$.

Consistent norm: A matrix norm $\|A\|$ on $R^{n \times m}$ is called consistent with a vector norm $\|\cdot\|_a$ on R^n and a vector norm $\|\cdot\|_b$ on R^m , if $\|Ax\|_a \leq \|A\| \|x\|_b$ for $\forall x \in R^m$ and $A \in R^{n \times m}$.

A class of consistent norms are defined by

$$\|A\| = \max_{x \in R^m} \frac{\|Ax\|}{\|x\|}, \text{ for } A \in R^{n \times m}$$

This matrix norm is dependent on the vector norm $\|\cdot\|$.

Three special matrix norms: $\|\cdot\|_2$, $\|\cdot\|_1$ and $\|\cdot\|_\infty$

$$\|A\|_2 = \max_{x \in R^m} \frac{\|Ax\|_2}{\|x\|_2}$$

$$\|A\|_1 = \max_{x \in R^m} \frac{\|Ax\|_1}{\|x\|_1}$$

$$\|A\|_\infty = \max_{x \in R^m} \frac{\|Ax\|_\infty}{\|x\|_\infty}$$

Theorem 2.2 Let $A \in R^{n \times m}$, then

$$\|A\|_1 = \max_j \sum_{i=1}^n |a_{ij}|$$

$$\|A\|_2 = \max_i \sqrt{\lambda_i(A^T A)}$$

$$\|A\|_\infty = \max_i \sum_{j=1}^m |a_{ij}|$$

where $\lambda_i(A^T A)$ denotes an eigenvalue of $A^T A$. *Proof:* We only consider $\|A\|_2$.

$$\|A\|_2 = \max_{x \in R^m} \frac{\|Ax\|_2}{\|x\|_2} = \max_{x \in R^m} \frac{\sqrt{x^T A^T A x}}{\sqrt{x^T x}}$$

Since $A^T A$ is symmetric, there are m orthogonal eigenvectors v_j . Assume

$$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_m > 0$$

be eigenvalues of $A^T A$ and $v_i, i = 1, 2, \dots, m$, be the corresponding eigenvectors, where

$$A^T A v_j = \lambda_j v_j, \quad v_i^T v_j = \delta_{ij}.$$

For any $x \in R^n$, there exist $\alpha_i, i = 1, 2, \dots, m$, such that

$$x = \alpha_1 v_1 + \alpha_2 v_2 + \cdots + \alpha_m v_m.$$

Thus,

$$A^T A x = A^T A (\alpha_1 v_1 + \alpha_2 v_2 + \cdots + \alpha_m v_m) = \alpha_1 \lambda_1 v_1 + \alpha_2 \lambda_2 v_2 + \cdots + \alpha_m \lambda_m v_m.$$

Further,

$$\begin{aligned} x^T A^T A x &= (\alpha_1 v_1 + \alpha_2 v_2 + \cdots + \alpha_m v_m)^T (\alpha_1 \lambda_1 v_1 + \alpha_2 \lambda_2 v_2 + \cdots + \alpha_m \lambda_m v_m) \\ &= \lambda_1 \alpha_1^2 + \lambda_2 \alpha_2^2 + \cdots + \lambda_m \alpha_m^2, \end{aligned}$$

and

$$x^T x = \alpha_1^2 + \alpha_2^2 + \cdots + \alpha_m^2.$$

For any $x \in R^m$,

$$\frac{x^T A^T A x}{x^T x} = \frac{\lambda_1 \alpha_1^2 + \lambda_2 \alpha_2^2 + \cdots + \lambda_m \alpha_m^2}{\alpha_1^2 + \alpha_2^2 + \cdots + \alpha_m^2} \leq \frac{\lambda_1 (\alpha_1^2 + \alpha_2^2 + \cdots + \alpha_m^2)}{\alpha_1^2 + \alpha_2^2 + \cdots + \alpha_m^2} = \lambda_1.$$

Let $x = v_1$,

$$\frac{v_1^T A^T A v_1}{v_1^T v_1} = \lambda_1$$

Then,

$$\max_{x \in R^m} \frac{\|Ax\|_2}{\|x\|_2} = \sqrt{\lambda_1} = \max_{1 \leq j \leq m} \sqrt{\lambda_j(A^T A)}.$$

Theorem 2.3 Let $A \in R^{n \times n}$ and $\rho(A) = \max_i |\lambda_i(A)|$ define the spectral radius of the matrix A . Then for any consistent norm,

$$\rho(A) \leq \|A\|.$$

Proof. Let λ and v be an eigenvalue and its eigenvector of A ,

$$Av = \lambda v$$

We have

$$|\lambda| \|v\| = \|\lambda v\| = \|Av\| \leq \|A\| \|v\|$$

which leads to

$$|\lambda| \leq \|A\|.$$

We have completed the proof. ■

Finally we consider the perturbation of a linear system Let

$$Ax = b$$

be a given linear system, where A is invertible. Let \tilde{b} be a perturbed vector of b and the perturbed system be given by

$$A\tilde{x} = \tilde{b}.$$

We need to estimate the relative error of this system. Since A is invertible,

$$\begin{aligned} \|x - \tilde{x}\| &= \|A^{-1}b - A^{-1}\tilde{b}\| \leq \|A^{-1}\| \|b - \tilde{b}\| \\ \|A^{-1}\| \|Ax\| \frac{\|b - \tilde{b}\|}{\|b\|} &\leq \|A^{-1}\| \|A\| \|x\| \frac{\|b - \tilde{b}\|}{\|b\|} \end{aligned}$$

and therefore, the relative error is given by

$$\frac{\|x - \tilde{x}\|}{\|x\|} \leq \kappa(A) \frac{\|b - \tilde{b}\|}{\|b\|}$$

where

$$\kappa(A) = \|A^{-1}\| \|A\|$$

defines a condition number of the matrix A . When κ is very large, the matrix is called *ill-conditioning*. In this case, the solution of system is very sensitive to the small change in the vector b .

Example Let A be orthogonal. Then

$$\kappa(A) = \|A\|_2 \|A^{-1}\|_2 = 1.$$

2.6 Convergence analysis. Many iterative methods can be given in the form of

$$x^{k+1} = Mx^k + b \quad (1)$$

for a given x^0 . We have

$$M = I - \alpha A \quad M = D^{-1}(L + U) \quad M = (D + L)^{-1}U$$

for Richardson, Jacobi and G-S, respectively.

Theorem 2.4 The iteration in (2.1) is convergent for any initial x^0 if and only if

$$\rho(M) < 1$$

where $\lambda_i(M)$ denotes an eigenvalue of M .

Proof. Here we assume that M is diagonalizable (non-diagonalizable case is given in assignment).

If $\max_i \lambda_i(M) < 1$, the matrix $I - M$ is nonsingular. Let \bar{x} be a solution of the linear system

$$(I - M)x = b.$$

Then,

$$x^{k+1} - \bar{x} = M(x^k - \bar{x}) = \dots = M^{k+1}(x^0 - \bar{x})$$

and therefore,

$$\|x^{k+1} - \bar{x}\| \leq \|M^{k+1}\| \|x^0 - \bar{x}\|.$$

where $\|\cdot\|$ defines a consistent norm. Since M is diagonalizable, there exists a nonsingular matrix Q such that

$$M = Q^{-1}\Lambda Q$$

where $\Lambda = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_n)$ and λ_i is an eigenvalue of M . Moreover,

$$M^{k+1} = (Q^{-1}\Lambda Q)^{k+1} = Q^{-1}\Lambda^{k+1}Q.$$

Since $\|\Lambda^{k+1}\| \rightarrow 0$,

$$\|M^{k+1}\| \rightarrow 0 \quad \text{as } k \rightarrow \infty$$

and we have

$$\|x^{k+1} - \bar{x}\| \rightarrow 0 \quad \text{as } k \rightarrow \infty.$$

If the iteration is convergent for any initial x^0 , we assume that

$$x^k \rightarrow \tilde{x}.$$

From (2.1), letting $k \rightarrow \infty$, we have

$$\tilde{x} = M\tilde{x} + b$$

and

$$x^{k+1} - \tilde{x} = M(x^k - \tilde{x}).$$

Let $x^0 - \tilde{x}$ be an eigenvector of M corresponding to the eigenvalue λ_i , *i.e.*,

$$M(x^0 - \tilde{x}) = \lambda_i(x^0 - \tilde{x}).$$

We have

$$x^{k+1} - \tilde{x} = M^{k+1}(x^0 - \tilde{x}) = \lambda_i^{k+1}(x^0 - \tilde{x})$$

and

$$\|x^{k+1} - \tilde{x}\| = |\lambda_i^{k+1}| \|x^0 - \tilde{x}\| \rightarrow 0 \quad \text{as } k \rightarrow \infty$$

which implies that $|\lambda_i| < 1$. ■

Theorem 2.5

(i) If $\lambda_i(A) > 0$, the Richardson's iteration with $0 < \alpha < 1/\max \lambda_i(A)$ converges to the solution of the system $Ax = b$.

(ii) If the matrix A is row strictly diagonally dominant, Jacobi and G-S iterations converge to the solution of $Ax = b$.

Proof. (i) When $0 < \alpha < 1/\max \lambda_i(A)$,

$$\rho(M) = \max |1 - \alpha \lambda_i(A)| < 1.$$

The iteration is convergent. Let $k \rightarrow \infty$. The iteration converges to the solution of

$$x = Mx + b$$

which is equivalent to $Ax = b$.

(ii) Let $M = (m_{ij})$. For Jacobi iteration, we have $M = I - D^{-1}A$ and

$$m_{ij} = -a_{ij}/a_{ii}, \quad i \neq j \quad a_{ii} = 0.$$

When A is strictly diagonally dominant,

$$\|M\|_\infty = \max_i \sum_{j \neq i} \frac{|a_{ij}|}{|a_{ii}|} < 1$$

and

$$\rho(M) \leq \|M\|_\infty < 1.$$

By Theorem 2.4, the Jacobi iteration is convergent for any initial x^0 .

For G-S iteration, we consider the following eigenvalue problem

$$Mv = \lambda v$$

or equivalently

$$Uv = \lambda(D - L)v$$

which implies that

$$-\sum_{j=i+1}^n a_{ij}v_j = \lambda \sum_{j=1}^i a_{ij}v_j$$

It follows that

$$\lambda a_{ii}v_i = - \sum_{j=i+1}^n a_{ij}v_j + \lambda \sum_{j=1}^{i-1} a_{ij}v_j$$

Since v is an eigenvector, $v \neq 0$. Let

$$|v_l| = \max_i |v_i|$$

and consider the i -th equation of the above system

$$\lambda a_{ii}v_i = - \sum_{j=l+1}^n a_{ij}v_j + \lambda \sum_{j=1}^{l-1} a_{ij}v_j.$$

We have immediately

$$|\lambda| |a_{ii}| |v_i| \leq \sum_{j=l+1}^n |a_{ij}| |v_j| + |\lambda| \sum_{j=1}^{l-1} |a_{ij}| |v_j|$$

and moreover,

$$|\lambda| |a_{ii}| \leq \sum_{j=l+1}^n |a_{ij}| \frac{|v_j|}{|v_i|} + |\lambda| \sum_{j=1}^{l-1} |a_{ij}| \frac{|v_j|}{|v_i|} \leq \sum_{j=l+1}^n |a_{ij}| + |\lambda| \sum_{j=1}^{l-1} |a_{ij}|$$

Solving the above inequality for $|\lambda|$ gives

$$|\lambda| < 1. \quad \blacksquare$$

Theorem 2.6

(i) If A is a Z -matrix ($a_{ij} \leq 0$ for $i \neq j$ and $a_{ii} > 0$), then only one of the following statements holds

- (a) $0 \leq \rho(M_J) < \rho(M_{GS}) < 1$.
- (b) $1 < \rho(M_J) < \rho(M_{GS})$.
- (c) $\rho(M_J) = \rho(M_{GS}) = 0$.
- (d) $\rho(M_J) = \rho(M_{GS}) = 1$.

(ii) If A is symmetric positive definite, then the SOR iteration is convergent when $0 < \omega < 2$.

2.7 Matrix eigenvalues

Let A be an $n \times n$ matrix. Let λ be a scalar. If the equation

$$Ax = \lambda x$$

has a nontrivial solution (i.e., $x \neq 0$), then the λ is an eigenvalue of A and the nonzero solution x is an eigenvector of A corresponding to the eigenvalue λ .

The condition that the above system has a nontrivial solution is

$$\det(A - \lambda I) = 0.$$

In linear algebra, we often solve the algebraic equation $\det(A - \lambda I) = 0$ to get eigenvalues and then, solve the system $Ax = \lambda x$ for corresponding eigenvectors. We have studied these problems carefully in linear algebra courses. Some main features are given below:

- An $n \times n$ matrix has n eigenvalues, including multiplicity.
- A symmetric matrix has n orthogonal eigenvectors and its eigenvalues are real.
- Some matrices may not have n linearly independent eigenvectors.
- The eigenvalues of a symmetric positive definite matrix are positive.

In this section, we only introduce "Power Method" which is designed to compute the dominant eigenvalue and the corresponding eigenvector. Here we assume that for a given matrix A

- there is a single eigenvalue of maximum modulus.
- there is a linearly independent set of n eigenvectors.

Then we can label these n eigenvalues by λ_j and assume that

$$|\lambda_1| > |\lambda_2| \geq \dots \geq |\lambda_n|$$

and

$$Av_j = \lambda_j v_j, \quad j = 1, 2, \dots, n.$$

Now we choose an initial guess $x^{(0)}$ and define a sequence by

$$x^{(k)} = A^k x^{(0)}, \quad k = 1, 2, \dots$$

Since A has n linearly independent eigenvectors, v_j , $j = 1, 2, \dots, n$. Then

$$x^{(0)} = \sum_{j=1}^n \alpha_j v_j$$

and

$$\begin{aligned} x^{(k)} &= A^k x^{(0)} = A^k \sum_{j=1}^n \alpha_j v_j = \sum_{j=1}^n \alpha_j \lambda_j^k v_j \\ &= \lambda_1^k \left(\alpha_1 v_1 + \alpha_2 \left(\frac{\lambda_2}{\lambda_1} \right)^k v_2 + \dots + \alpha_n \left(\frac{\lambda_n}{\lambda_1} \right)^k v_n \right) \end{aligned}$$

Since $|\lambda_1| > |\lambda_j|$ for $j = 2, \dots, n$, we see that $(\lambda_j/\lambda_1)^k \rightarrow 0$ as $k \rightarrow \infty$ and the vector $x^{(k)}/\lambda_1^k \rightarrow \alpha_1 v_1$. In practical computation, we can choose a component, such as the j -th component and let

$$r_k = \frac{x_j^{(k+1)}}{x_j^{(k)}} = \lambda_1 \left(\frac{\alpha_1 v_{1j} + o\left(\left(\frac{\lambda_2}{\lambda_1}\right)^k\right)}{\alpha_1 v_{1j} + o\left(\left(\frac{\lambda_2}{\lambda_1}\right)^k\right)} \right) \rightarrow \lambda_1$$

Example Use the power method on the following matrix and initial vector

$$A = \begin{bmatrix} 6 & 5 & -5 \\ 2 & 6 & -2 \\ 2 & 5 & -1 \end{bmatrix} \quad x^{(0)} = \begin{bmatrix} -1 \\ 1 \\ 1 \end{bmatrix}.$$

By using the power method, we obtain

$$\begin{aligned}x^{(0)} &= (-1, 1, 1)^T, \\x^{(1)} &= (-1, 0.3333, 0.3333)^T, \quad r_0 = 2.0 \\x^{(2)} &= (-1, -0.1111, -0.1111)^T, \quad r_1 = -2.0 \\x^{(3)} &= (-1, -0.4074, -0.4074)^T, \quad r_2 = 22.0 \\x^{(4)} &= (-1, -0.6049, -0.6049)^T, \quad r_3 = 8.809 \\&\dots\dots \\x^{(28)} &= (-1, -1, -1)^T, \quad r_{27} = 6.00007\end{aligned}$$