Rank

- The ***rank*** of a matrix is the number of leading ones in the reduced row echelon form.

Eg The following matrix

$$\begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ 6 & 7 & 8 & 9 & 10 \\ 11 & 12 & 13 & 14 & 15 \end{bmatrix}$$

Has a reduced row echelon form of

$$\begin{bmatrix} 1 & 0 & -1 & -2 & -3 \\ 0 & 1 & 2 & 3 & 4 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

The rank of matrix is 2

Comparing Iterative and reduction methods

- When are iterative methods (Gauss-Seidel) useful? A major advantage of Gauss-Seidel is that roundoff errors are not given a chance to "accumulate," as they are in Gaussian elimination and the Gauss Jordan Method, because each iteration essentially creates a new approximation to the solution. The only roundoff error that we need to consider with Gauss-Seidel method is the error involved in the most recent step.

Comparing Iterative and reduction methods

- Also, in many applications, the coefficient matrix for a given system contains a large number of zeroes (sparse matrix). When a linear system has a sparse matrix, each equation in the system may involve very few variables. If so, each step of the Gauss-Seidel process is relatively easy. However, neither the Gauss-Jordan Method nor Gaussian elimination would be very attractive in such a case because the cumulative effect of many row operations would tend to replace the zero coefficients with nonzero numbers. But even if the coefficient matrix is not sparse, Gauss-Seidel methods often give more accurate answers when large matrices are involved because fewer arithmetic operations are performed overall.

Comparing Iterative and reduction methods

- On the other hand, when Gauss-Seidel method take an extremely large number of steps to stabilize or do not stabilize at all (absence of diagonal dominance), it is much better to use the Gauss-Jordan Method or Gaussian elimination.

Ill-conditioned matrix

- System of equations, in which a very small change in a coefficient leads to a very large change in the solution set, are called

  ill-conditioned systems.

- Ill- conditioned or Nearly singular matrix—an invertible matrix that can become singular if some of its entries are changed ever so slightly. In this case, row reduction may produce fewer than n pivot positions, as a result of roundoff error. Also, roundoff error can sometimes make a singular matrix appear to be invertible.

Ill-conditioned matrix

Consider the following equations(the extra digits are in the ninth significant place)

$$x_1 + 2x_2 = 3$$
$$1.000\ 000\ 01x_1 + 2x_2 = 3.000\ 000\ 01$$

The solution is $x_1 = 1$, $x_2 = 1$. A computer has more trouble. If it represents real numbers to eight significant places, called single precision, then it will represent the second equation internally as

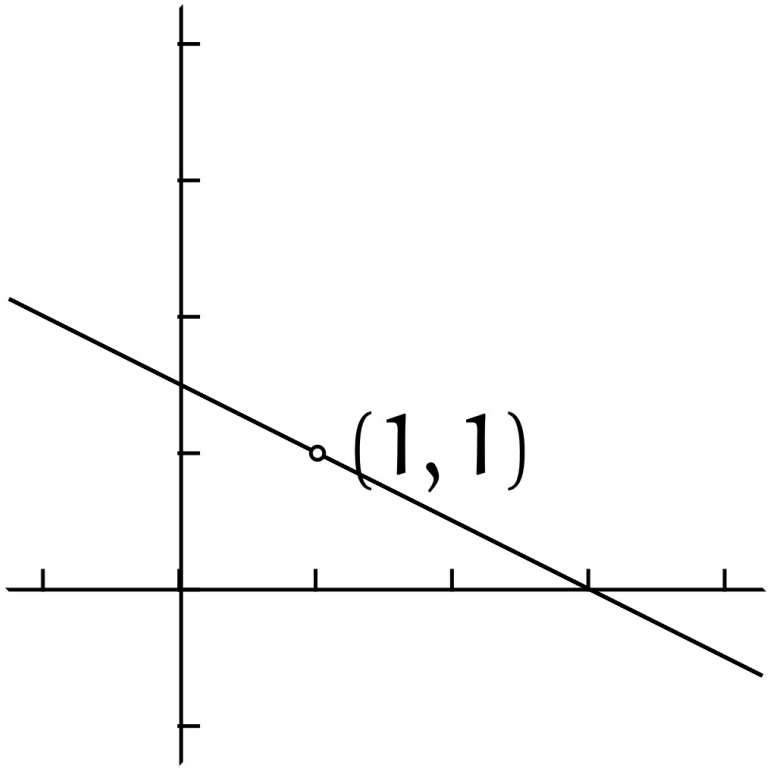
1.000 000 0 x_1 + 2 x_2 = 3.000 000 0,

losing the digits in the ninth place. Instead of reporting the correct solution, this computer will think that the two equations are equal and it will report that the system is singular.

Ill-conditioned matrix

Consider this graph of the system.

We cannot tell the two lines apart; this system is nearly singular in the sense that the two lines are nearly the same line. This gives the system the property that a small change in an equation can cause a large change in the solution.



Ill-conditioned matrix

- For instance, changing the 3.000 000 01 to 3.000 000 03 changes the intersection point from (1, 1) to (3, 0). The solution changes radically depending on the ninth digit, which explains why an eight-place computer has trouble. A problem that is very sensitive to inaccuracy or uncertainties in the input values is ill-conditioned
- The above example gives one way in which a system can be difficult to solve on a computer. It has the advantage that the picture of nearly-equal lines gives a memorable insight into one way for numerical difficulties to happen. Unfortunately this insight isn't useful when we wish to solve some large system. We typically will not understand the geometry of an arbitrary large system

Ill-conditioned matrix

Consider the following equations

$$0.001x_1 + x_2 = 1$$
$$x_1 - x_2 = 0$$

The second equation gives $x_1 = x_2$, so $x_1 = x_2 = 1/1.001$ and thus both variables have values that are just less than 1. A computer using two digits represents the system internally in this way (we will do this example in two-digit floating point arithmetic for clarity but inventing a similar one with eight or more digits is easy).

Ill-conditioned matrix

- $(1.0\times10^{-3})x_1 + (1.0\times10^{0})x_2 = (1.0\times10^{0})$
- $(1.0\times10^{0})x_1 - (1.0\times10^{0})x_2 = (0.0\times10^{0})$

The row reduction step −1000 R_1 + R_2 produces a second equation −1001 x_2 = −1000, which this computer rounds to two places as

$$(-1.0\times10^{3})x_2 = (-1.0\times10^{3})$$

The computer decides from the second equation that x_2 = 1 and with that it concludes from the first equation that x_1 = 0.

The x_2 value is close but the x_1 is incorrect

Ill-conditioned matrix

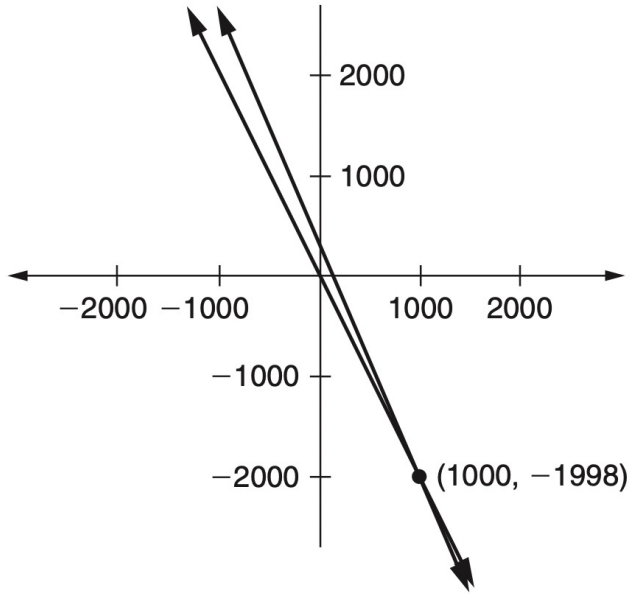
- Another cause of unreliable output is the computer's reliance on floating point arithmetic when the system-solving code leads to using leading entries that are small.
- An experienced programmer may respond by using double precision, which retains sixteen significant digits, or perhaps using some even larger size. This will indeed solve many problems. However, double precision has greater memory requirements and besides we can obviously tweak the above to give the same trouble in the seventeenth digit, so double precision isn't a panacea. We need a strategy to minimize numerical trouble as well as some guidance about how far we can trust the reported solutions.

Ill-conditioned matrix

Consider the following similar systems

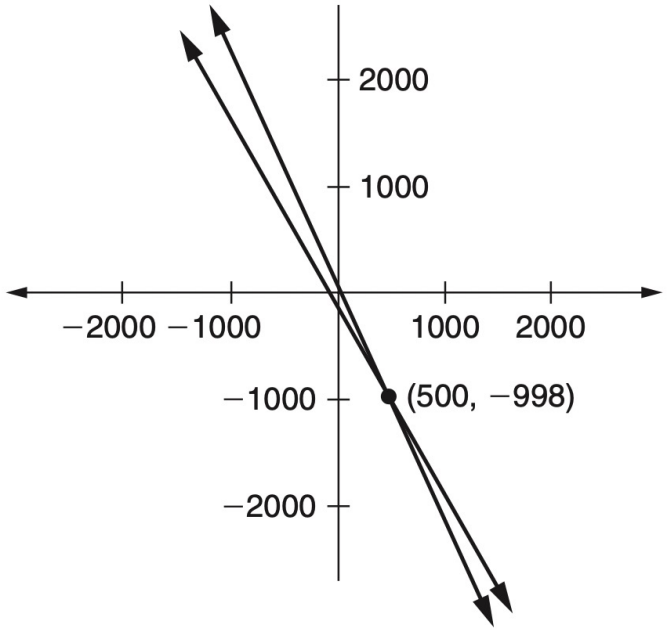
A:

$$2x_1 + x_2 = 2$$
$$2.005x_1 + x_2 = 7$$



B:

$$2x_1 + x_2 = 2$$
$$2.01x_1 + x_2 = 7$$



Ill-conditioned matrix

- Even though the coefficients of systems (A) and (B) are almost identical, the solutions to the systems are very different.
- Solution to (A) = (1000, -1998) and solution to (B) = (500, -998).
- In this case, there is a geometric way to see that these systems are ill-conditioned; the pair of lines in each system are almost parallel. Therefore, a small change in one line can move the point of intersection very far along the other line.

Ill-conditioned matrix

- Suppose the coefficients in system (A) had been obtained after a series of long calculations. A slight difference in the roundoff error of those calculations could have led to a very different final solution set. Thus, we need to be very careful when working with ill-conditioned systems. Special methods have been developed for recognizing ill-conditioned systems, and a technique known as iterative refinement is used when the coefficients are known only to a certain degree of accuracy.

Ill-conditioned matrix

- The degree of ill-conditioning of a matrix is measured by its *condition number.*
- Some matrix programs will compute a condition number for a square matrix. The larger the condition number, the closer the matrix is to being singular. The condition number of the identity matrix is 1.
- A singular matrix has an infinite condition number. In extreme cases, a matrix program may not be able to distinguish between a singular matrix and an ill-conditioned matrix.

Application in Machine Learning

- Why do we need these methods in machine learning?
  - Machine learning deals a lot with data
  - This data is at times represented in the form of equations
  - We need to solve these equations to arrive at a solution of our problem
  - Many applications require the inverse of matrices

Matrices Subspaces

Span of a set of Matrices

- Generally, then, if $A_1, A_2, \ldots A_k$ are matrices of the same size and

$c_1, c_2, \ldots c_k$ are scalars, we may form the ***linear combination***

$$c_1 A_1 + \ c_2 A_2, \ldots c_k A_k$$

We will refer to $c_1, c_2, \ldots c_k$ as the ***coefficients*** of the linear combination

Span of a set of Matrices

- The ***span*** of a set of matrices is the set of all linear combinations of the matrices.

Span of a set of Matrices

Let $A_1 = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$, $A_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$, and $A_3 = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$.

(a) Is $B = \begin{bmatrix} 1 & 4 \\ 2 & 1 \end{bmatrix}$ a linear combination of A_1, A_2, and A_3?

(b) Is $C = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$ a linear combination of A_1, A_2, and A_3?

Span of a set of Matrices

(a) We want to find scalars c_1, c_2, c_3 such that

$$c_1A_1 + c_2A_2 + c_3A_3 = B$$

$$c_1\begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} + c_2\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + c_3\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 4 \\ 2 & 1 \end{bmatrix}$$

This can be written as

$$\begin{aligned} c_2 + c_3 &= 1 \\ c_1 + c_3 &= 4 \\ -c_1 + c_3 &= 2 \\ c_2 + c_3 &= 1 \end{aligned}$$

Span of a set of Matrices

$c_1 \quad + \; c_3 = 4$

$-c_1 \quad\quad c_3 = 2$

Gives $c_1 = 1, \; c_3 = 3$

$c_2 \; + \; c_3 = 1$ gives $c_2 = -2$

B is a linear combination of A_1 , A_2 , A_3

i.e. $1\begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} - \; 2\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + 1\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 4 \\ 2 & 1 \end{bmatrix}$

Span of a set of Matrices

(b) We want to find scalars c_1, c_2, c_3 such that

$$c_1A_1 + c_2A_2 + c_3A_3 = C$$

$$c_1\begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} + c_2\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + c_3\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 4 \\ 2 & 1 \end{bmatrix}$$

This can be written as

$$\begin{aligned} c_2 + c_3 &= 1 \\ c_1 + c_3 &= 2 \\ -c_1 + c_3 &= 3 \\ c_2 + c_3 &= 4 \end{aligned}$$

Span of a set of Matrices

$$c_2 + c_3 = 1$$

$$c_2 + c_3 = 4$$

Gives, there is no solution for this system of equation.

C is not a linear combination of A_1 , A_2 , A_3

Subspace of R^n

- A **subspace** of R^n is any set H in R^n that has three properties:
  - The zero vector is in H.
  - For each **u** and **v** in H, the sum **u** + **v** is in H.
  - For each **u** in H and each scalar c, the vector c**u** is in H.

  - In words, a subspace is *closed* under addition and scalar multiplication.

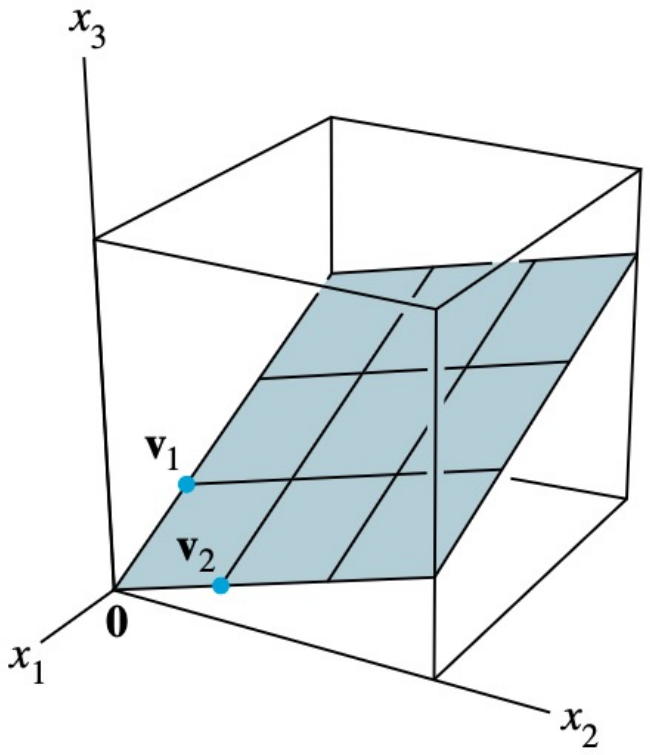


FIGURE 1

Span $\{\mathbf{v}_1, \mathbf{v}_2\}$ as a plane through the origin.

Subspace Example 1

- Example 1 If v_1 and v_2 are in R^n and H = Span$\{v_1, v_2\}$, then H is a subspace of R^n .
- To verify this statement, note that the zero vector is in H (because 0**v** + 0**u** is a linear combination of **u** and **v**).

Now take two arbitrary vectors in H , say,

$\boldsymbol{u} = s_1 v_1 + s_2 v_2 \; and \; \boldsymbol{v} = t_1 v_1 + \; t_2 v_2$

Then

$u + v = (s_1 + t_1) v_1 + (s_2 + \; t_2) v_2$

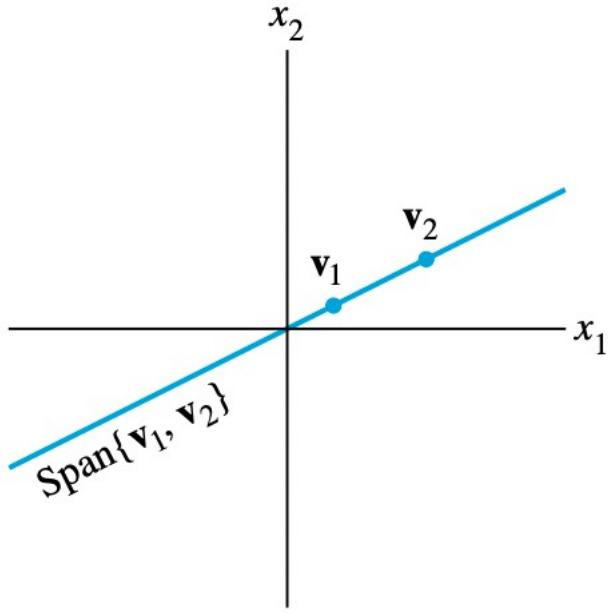
which shows that **u** + **v** is a linear combination of v_1 and v_2 and hence is in H .

Also, for any scalar c, the vector c**u** is in H , because

$\boldsymbol{cu} = c(s_1 v_1 + s_2 v_2)$ = $(cs_1) v_1 + (\mathrm{c}s_2) v_2)$

Subspace Example 2

- If v_1 is not zero and if v_2 is a multiple of v_1 , then v_1 and v_2 simply span a *line* through the origin. So a line through the origin is another example of a subspace.



$\mathbf{v}_1 \neq \mathbf{0}$, $\mathbf{v}_2 = k\mathbf{v}_1$.

Subspace Example 3

- A line L *not* through the origin is *not* a subspace, because it does not contain the origin, as required. Also, Fig. 2 shows that L is not closed under addition or scalar multiplication.

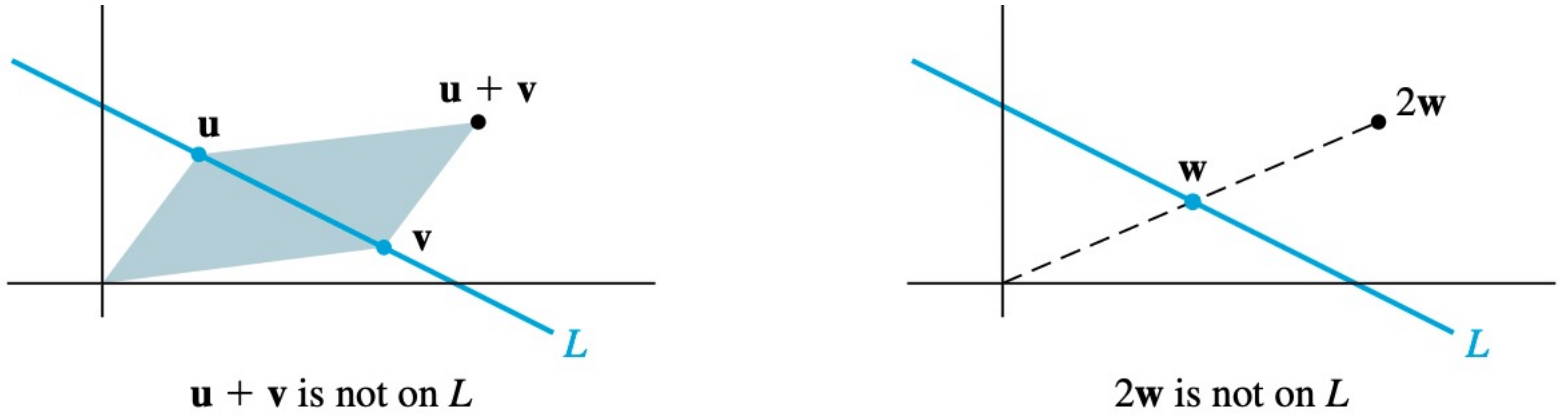


FIGURE 2

Subspace Example 4

Let $v_1 = \begin{bmatrix} 2 \\ 3 \\ -5 \end{bmatrix}$, $v_2 = \begin{bmatrix} -4 \\ -5 \\ 8 \end{bmatrix}$, and w = $\begin{bmatrix} 8 \\ 2 \\ -9 \end{bmatrix}$. Determine if w is in the subspace of R^3 generated by v_1 and v_2.

The vector w is in the subspace generated by v_1 and v_2 if and only if the vector equation $x_1 v_1 + x_2 v_2$ = w is consistent.

$$[\, v_1 \quad v_2 \quad w] = \begin{bmatrix} 2 & -4 & 8 \\ 3 & -5 & 2 \\ -5 & 8 & -9 \end{bmatrix}$$

Subspace Example 4

Replace R_2 by R_2 - (3/2) R_1 and R_3 by R_3 + (5/2) R_1

$$\begin{bmatrix} 2 & -4 & 8 \\ 0 & 1 & -10 \\ 0 & -2 & 11 \end{bmatrix}$$

Replace R_3 by R_3 + 2 R_2

$$\begin{bmatrix} 2 & -4 & 8 \\ 0 & 1 & -10 \\ 0 & 0 & -9 \end{bmatrix}$$

The row operations below show that w is not in the subspace generated by v_1 and v_2.

Subspace Example 5

Let $v_1 = \begin{bmatrix} 1 \\ -2 \\ 4 \\ 3 \end{bmatrix}$, $v_2 = \begin{bmatrix} 4 \\ -7 \\ 9 \\ 7 \end{bmatrix}$, $v_3 = \begin{bmatrix} 5 \\ -8 \\ 6 \\ 5 \end{bmatrix}$ and u = $\begin{bmatrix} -4 \\ 10 \\ -7 \\ -5 \end{bmatrix}$. Determine if u is in the subspace of R^3 generated by v_1 , v_2 and v_3.

The vector w is in the subspace generated by v_1 , v_2 and v_3 if and only if the vector equation $x_1v_1 + x_2v_2 + x_3v_3$ = u is consistent

$$[\, v_1 \quad v_2 \quad v_3 \quad u] = \begin{bmatrix} 1 & 4 & 5 & -4 \\ -2 & -7 & -8 & 10 \\ 4 & 9 & 6 & -7 \\ 3 & 7 & 5 & -5 \end{bmatrix}$$

Subspace Example 5

Replace R_2 by $R_2 + 2R_1$, R_3 by $R_3 - 4R_1$ and R_4 by $R_4 - 3R_1$

$$\begin{bmatrix} 1 & 4 & 5 & -4 \\ 0 & 1 & 2 & 2 \\ 0 & -7 & -14 & 9 \\ 0 & -5 & -10 & 7 \end{bmatrix}$$

Replace R_3 by $R_3 + 7R_2$

$$\begin{bmatrix} 1 & 4 & 5 & -4 \\ 0 & 1 & 2 & 2 \\ 0 & 0 & 0 & 23 \\ 0 & 0 & 0 & 17 \end{bmatrix}$$

The row operations below show that u is not in the subspace generated by v_1 $v_2$and v_3.