# Mathematical Foundations of Machine Learning
# Assignment 1

NAME ; EZE, NNAMDI SHEDRACK
ANDREW ID : SEZE

28th September, 2025

## Question 1

### Given Matrices

$$A = \begin{bmatrix} 2 & 4 & 1 \\ -3 & 0 & 5 \end{bmatrix}, \quad B = \begin{bmatrix} 1 & -1 \\ 2 & 3 \\ 0 & 4 \end{bmatrix}$$

### (a) Explain why the product AB is defined, state the dimensions of AB, and compute AB explicity

ANSWER: The product of AB is defined if the number of columns in A equals the number of rows in B.

Since A is 2×3 and B is 3×2 then the columns in A is 3 and the rows in B is also 3 , therefore the product AB is defined.

,

AB will have dimensions 2 x 2

$$AB = \begin{bmatrix} 2 & 4 & 1 \\ -3 & 0 & 5 \end{bmatrix} \begin{bmatrix} 1 & -1 \\ 2 & 3 \\ 0 & 4 \end{bmatrix}$$

$$\begin{aligned}
AB_{11} &= (2)(1) + (4)(2) + (1)(0) = (2) + (8) + (0) = 10 \\
AB_{12} &= (2)(-1) + (4)(3) + (1)(4) = (-2) + (12) + (4) = 14 \\
AB_{21} &= (-3)(1) + (0)(2) + (5)(0) = (-3) + (0) + (0) = -3 \\
AB_{22} &= (-3)(-1) + (0)(3) + (5)(4) = (3) + (0) + (20) = 23
\end{aligned}$$

Hence

$$AB = \begin{bmatrix} AB_{11} & AB_{12} \\ AB_{21} & AB_{22} \end{bmatrix} = \begin{bmatrix} 10 & 14 \\ -3 & 23 \end{bmatrix} = 2 \times 2$$

## (b) Determine whether the product BA is defined. If it is, compute BA and state its dimensions

Since B has 2 columns and 3 rows while A has 2 rows and 3 columns, then BA is defined. The dimensions of BA will be 3×3.

$$BA = \begin{bmatrix} 1 & -1 \\ 2 & 3 \\ 0 & 4 \end{bmatrix} \begin{bmatrix} 2 & 4 & 1 \\ -3 & 0 & 5 \end{bmatrix}$$

$$\begin{aligned}
BA_{11} &= 12 + (-1)(-3) = 2 + 3 = 5 \\
BA_{12} &= 14 + (-1)0 = 4 + 0 = 4 \\
BA_{13} &= 11 + (-1)5 = 1 - 5 = -4 \\
BA_{21} &= 22 + 3(-3) = 4 - 9 = -5 \\
BA_{22} &= 24 + 30 = 8 + 0 = 8 \\
BA_{23} &= 21 + 35 = 2 + 15 = 17 \\
BA_{31} &= 02 + 4(-3) = -12 \\
BA_{32} &= 04 + 40 = 0 \\
BA_{33} &= 01 + 45 = 20
\end{aligned}$$

Hence

$$BA = \begin{bmatrix} BA_{11} & BA_{12} & BA_{13} \\ BA_{21} & BA_{22} & BA_{23} \\ BA_{31} & BA_{32} & BA_{33} \end{bmatrix} = \begin{bmatrix} 5 & 4 & -4 \\ -5 & 8 & 17 \\ -12 & 0 & 20 \end{bmatrix} = 3 \times 3$$

## (c) Find the transpose $\mathbf{A^T}$

$$A = \begin{bmatrix} 2 & 4 & 1 \\ -3 & 0 & 5 \end{bmatrix} \quad A^T = \begin{bmatrix} 2 & -3 \\ 4 & 0 \\ 1 & 5 \end{bmatrix}$$

A is a 3 × 2 matrix dimensions

## (d) Compute $A^T B^T$

$$A^T = \begin{bmatrix} 2 & -3 \\ 4 & 0 \\ 1 & 5 \end{bmatrix}, \quad B = \begin{bmatrix} 1 & -1 \\ 2 & 3 \\ 0 & 4 \end{bmatrix}, \quad B^T = \begin{bmatrix} 1 & 2 & 0 \\ -1 & 3 & 4 \end{bmatrix}$$

Since the number of columns in $A^T$ (2) is equal to the number of rows in $B^T$, then $A^T B^T$ is defined.

$$
\begin{aligned}
AB_{11} &= (2)(1) + (-3)(-1) = (2) + (3) = 5 \\
AB_{12} &= (2)(2) + (-3)(3) = (4) + (-9) = -5 \\
AB_{13} &= (2)(0) + (-3)(4) = (0) + (-12) = -12 \\
AB_{21} &= (4)(1) + (0)(-1) = (4) + (0) = 4 \\
AB_{22} &= (4)(2) + (0)(3) = (8) + (0) = 8 \\
AB_{23} &= (4)(0) + (0)(4) = (0) + (0) = 0 \\
AB_{31} &= (1)(1) + (5)(-1) = (1) + (-5) = -4 \\
AB_{32} &= (1)(2) + (5)(3) = (2) + (15) = 17 \\
AB_{33} &= (1)(0) + (5)(4) = (0) + (20) = 20
\end{aligned}
$$

Finally

$$
A^T B^T = \begin{bmatrix} A^T B^T_{11} & A^T B^T_{12} & A^T B^T_{13} \\ A^T B^T_{21} & A^T B^T_{22} & A^T B^T_{23} \\ A^T B^T_{31} & A^T B^T_{32} & A^T B^T_{33} \end{bmatrix} = \begin{bmatrix} 5 & -5 & -12 \\ 4 & 8 & 0 \\ -4 & 17 & 20 \end{bmatrix} = 3 \times 3 \text{ matrix dimensions}
$$

# Question 2

A simple neural network layer applies a linear transformation from the two-dimensional input vector

$$X = \left[x1, x2\right]^T$$

to the three-dimensional output vector

$$y = \left[y1, y2, y3\right]^T$$

according to the equations:

$$\begin{aligned} y1 &= 2x1 + 3x2 \\ y2 &= 4x1 - x2 \\ y3 &= -x1 + 5x2 \end{aligned}$$

## (a) Write this transformation in the matrix-vector form $Y = AX$ by giving matrix A explicitly

### Solution

Extract coefficients from each equation:

y1 = 2x1 + 3x2, hence coefficient of x = 2 and 3
y2 = 4x1 - x2, hence coefficient of x = 4 and -1
y3 = -x1 + 5x2, hence coefficient of x = -1 and 5

Next we arrange the coefficients as rows in a matrix A:

$$A = \begin{bmatrix} 2 & 3 \\ 4 & -1 \\ -1 & 5 \end{bmatrix}$$

Then the transformation $y = AX$ becomes:

$$y = \begin{bmatrix} 2 & 3 \\ 4 & -1 \\ -1 & 5 \end{bmatrix} \begin{bmatrix} x1 \\ x2 \end{bmatrix}$$

## (b) State the dimensions of matrix A and vectors x and y

- $y = \begin{bmatrix} y1 \\ y2 \\ y3 \end{bmatrix}$, hence the dimensions of y = 3 × 1 (3 rows 1 column)
- $x = \begin{bmatrix} x1 \\ x2 \end{bmatrix}$, hence dimension of x = 2 × 1 (2 rows 1 column)

- $A = \begin{bmatrix} 2 & 3 \\ 4 & -1 \\ -1 & 5 \end{bmatrix}$, hence dimensions of A = 3 × 2 (3 rows and 2 columns)

### (c) Compute the numerical output y when the input is $x = \begin{bmatrix} 1 \\ -2 \end{bmatrix}$

Solution:

Recall $y = AX$ and

$$A = \begin{bmatrix} 2 & 3 \\ 4 & -1 \\ -1 & 5 \end{bmatrix}, \quad x = \begin{bmatrix} 1 \\ -2 \end{bmatrix}$$

Using dot product:

$$\begin{aligned} y1 &= (2)(1) + 3(-2) = 2 - 6 = -4 \\ y2 &= (4)(1) + (-1)(-2) = 4 + 2 = 6 \\ y3 &= (-1)(1) + 5(-2) = -1 - 10 = -11 \end{aligned}$$

Hence

$$y = \begin{bmatrix} -4 \\ 6 \\ -11 \end{bmatrix}$$

# Question 3

For each of the systems of linear equations in the unknowns $x_1$, $x_2$, $x_3$ given below, solve the system completely and show all steps of your chosen method (either Gaussian elimination, substitution, or another elimination method). Determine whether it has solutions, unique solution, no solutions, or infinitely many solutions. If the system has infinitely many solutions, give the general solution and one particular solution.

## System (a)

$$\begin{aligned} x_1 + x_2 + x_3 &= 3 \\ x_1 - x_2 + 2x_3 &= 2 \\ 2x_1 + 0x_2 + 3x_3 &= 1 \end{aligned}$$

We start by writing the augmented matrix:

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

Next we perform EROs, starting with eliminating the entries below the first 1 so that the $R_{21}$ and $R_{31}$ becomes zeros. (Pivoting the first column)

Let $R_2 = R_2 - R_1$:

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

Let $R_3 = R_3 - 2R_1$:

$$\begin{bmatrix} 1 & 1 & 1 & | & 3 \\ 0 & -2 & 1 & | & -1 \\ 0 & -2 & 1 & | & -5 \end{bmatrix}$$

Next we need to pivot $R_{22}$ to get a Row Echelon Form of the equation

Let $R_3 = R_3 - R_2$:

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

The last row represents $0x_1 + 0x_2 + 0x_3 = -4$ which is impossible, therefore we can say this system has no solution and is considered an inconsistent system.

## System (b)

$$\begin{aligned} x_1 + x_2 + x_3 &= 3 \\ x_1 - x_2 + 2x_3 &= 2 \\ x_2 + x_3 &= 2 \end{aligned}$$

Again, as a first step, we form the augmented matrix:

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

Now we perform elementary row operations (EROs), we start by pivoting the first column

Let $R_2 = R_2 - R_1$:

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

Next we pivot column 2 at $R_{22}$

Let $R_3 = 2R_3 + R_2$:

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

Next we apply back substitution:

$$\begin{aligned} x_1 + x_2 + x_3 &= 3 \\ -2x_2 + x_3 &= -1 \\ 3x_3 &= 3 \end{aligned}$$

Hence:

$$3x_3 = 3 \Rightarrow x_3 = \frac{3}{3} = 1$$

In equation (ii): $-2x_2 + x_3 = -1$ and $x_3 = 1$
Hence:

$$\begin{aligned} -2x_2 + 1 &= -1 \\ -2x_2 &= -1 - 1 \\ -2x_2 &= -2 \\ x_2 &= \frac{-2}{-2} = 1 \end{aligned}$$

Similarly, in equation (i): $x_1 + x_2 + x_3 = 3$

Hence:

$$\begin{aligned}
x_1 + 1 + 1 &= 3 \\
x_1 &= 3 - 1 - 1 \\
x_1 &= 1
\end{aligned}$$

Finally:

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

This is unique solution.

## System (c)

$$\begin{aligned}
x_1 + x_2 + x_3 &= 3 \\
x_1 - x_2 + 2x_3 &= 2 \\
2x_1 + 0x_2 + 3x_3 &= 5
\end{aligned}$$

We form the augmented matrix as a first step:

$$\begin{bmatrix} 1 & 1 & 1 & | & 3 \\ 1 & -1 & 2 & | & 2 \\ 2 & 0 & 3 & | & 5 \end{bmatrix}$$

Next we perform EROs to pivot $R_{11}$, $R_{22}$, and $R_{33}$

Let $R_2 = R_2 - R_1$:

$$\begin{bmatrix} 1 & 1 & 1 & | & 3 \\ 0 & -2 & 1 & | & -1 \\ 2 & 0 & 3 & | & 5 \end{bmatrix}$$

Let $R_3 = R_3 - 2R_1$:

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

Let $R_3 = R_3 - R_2$:

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

The last row reduces completely to zero which means there are only two independent equations with three unknowns.

Let's assume a free variable $a$ at $x_3$, so $x_3 = a$

Hence:

$$-2x_2 + a = -1 \quad \Rightarrow \quad a = -1 + 2x_2 \quad \text{and} \quad 2x_2 = a + 1 \quad \Rightarrow \quad x_2 = \frac{a+1}{2}$$

Likewise:

$$x_1 + x_2 + a = 3 \quad \Rightarrow \quad x_1 = 3 - x_2 - a \quad \text{and} \quad x_2 = \frac{a+1}{2}$$

Multiply through by 2:

$$2x_1 = 6 - 2a - (a+1) = 6 - 2a - a - 1 = 5 - 3a \quad \Rightarrow \quad x_1 = \frac{5-3a}{2}$$

Hence:

$$x_1 = \frac{5-3a}{2}, \quad x_2 = \frac{a+1}{2}, \quad x_3 = a$$

If we assume $a = 1$, we get a particular solution:

$$x_1 = \frac{5-3(1)}{2} = \frac{2}{2} = 1, \quad x_2 = \frac{1+1}{2} = \frac{2}{2} = 1, \quad x_3 = a = 1$$

Therefore, the matrix is said to have infinitely many solutions since the free variable $a$ can take any value.

## (d) Determinant of A

**System (a):**

Recall:

$$A = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -1 & 2 \\ 2 & 0 & 3 \end{bmatrix} \quad b = \begin{bmatrix} 3 \\ 2 \\ 1 \end{bmatrix} \quad \text{and augment } M = \begin{bmatrix} 1 & 1 & 1 & | & 3 \\ 1 & -1 & 2 & | & 2 \\ 2 & 0 & 3 & | & 1 \end{bmatrix}$$

Compute determinant:

$$|A| = 1 \cdot ((-1)(3) - (0)(2)) - 1 \cdot ((1)(3) - (2)(2)) + 1 \cdot ((1)(0) - (2)(-1))$$

$$= 1 \cdot (-3) - 1 \cdot (3 - 4) + 1 \cdot (0 + 2) = -3 + 1 + 2 = 0$$

From our calculations:

$$\text{rank}(A) = 2, \quad \text{rank}(M) = 3$$

Since $\text{rank}(A) \neq \text{rank}(M)$, the system is inconsistent. This means we have 2 independent equations but 3 unknowns, hence no solution exists.

**System (b):**

$$A = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -1 & 2 \\ 0 & 1 & 1 \end{bmatrix} \quad b = \begin{bmatrix} 3 \\ 2 \\ 2 \end{bmatrix} \quad \text{and augment } M = \begin{bmatrix} 1 & 1 & 1 & | & 3 \\ 1 & -1 & 2 & | & 2 \\ 0 & 1 & 1 & | & 2 \end{bmatrix}$$

Compute determinant:

$$|A| = 1 \cdot ((-1)(1) - (2)(1)) - 1 \cdot ((1)(1) - (0)(2)) + 1 \cdot ((1)(0) - (0)(-1))$$

$$= 1 \cdot (-1 - 2) - 1 \cdot (1 - 0) + 1 \cdot (0 - 0) = -3 - 1 + 0 = -4$$

Since $|A| \neq 0$ and $\text{rank}(A) = \text{rank}(M) = 3$, the system is consistent and has a unique solution.

**System (c):**

$$A = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -1 & 2 \\ 2 & 0 & 3 \end{bmatrix}, \quad b = \begin{bmatrix} 3 \\ 2 \\ 5 \end{bmatrix} \quad \text{and augment } M = \begin{bmatrix} 1 & 1 & 1 & | & 3 \\ 1 & -1 & 2 & | & 2 \\ 2 & 0 & 3 & | & 5 \end{bmatrix}$$

Compute determinant:

$$|A| = 1 \cdot ((-1)(3) - (0)(2)) - 1 \cdot ((1)(3) - (2)(2)) + 1 \cdot ((1)(0) - (2)(-1))$$

$$= 1 \cdot (-3) - 1 \cdot (3 - 4) + 1 \cdot (0 + 2) = -3 + 1 + 2 = 0$$

Since $|A| = 0$ but $\text{rank}(A) = \text{rank}(M)$, the system has infinitely many solutions and is consistent.

Question 3(d): Computational Analysis For each system in (a), (b) and (c), use python NumPy to define the coefficient matrix A and the augmented matrix M. Compute the determinant of A to test for invertibility and the ranks of A and M to assess consistency via the Rouch´e–Capelli theorem. Use these results to justify your classification from part above. If a unique solution exists, compute it numerically using np.linalg.solve(A, b).

In [1]:

```
# Question 3 (d): Computational analysis using NumPy (determinant, ranks, classific

import numpy as nmp  # Import NumPy as 'nmp' per your preference
import pandas as pnd  # Import pandas as 'pnd' per your preference (not used; kept

EPS = 1e-12  # Numerical tolerance to interpret near-zero values as zero (floating-

def analyze_system(A_in, b_in, name="System"):  # Define a helper to analyze one sq
    A = nmp.array(A_in, dtype=float)  # Convert coefficient matrix to a float array
    b = nmp.array(b_in, dtype=float).reshape(-1, 1)  # Ensure RHS is a column vecto

    n_rows, n_cols = A.shape  # Unpack matrix dimensions (rows, cols)
    assert n_rows == n_cols, f"{name}: A must be square; got {n_rows}×{n_cols}"  #
    assert b.shape == (n_rows, 1), f"{name}: b must be shape ({n_rows}, 1); got {b.

    M = nmp.concatenate((A, b), axis=1)  # Build augmented matrix M = [A | b] by co

    detA = nmp.linalg.det(A)  # Compute determinant to test invertibility (non-zero
    rankA = nmp.linalg.matrix_rank(A)  # Compute rank of A (dimension of column spa
    rankM = nmp.linalg.matrix_rank(M)  # Compute rank of augmented matrix M = [A |

    if abs(detA) > EPS:  # Check if A is invertible using tolerance
        classification = "Unique solution"  # Non-singular matrix implies a unique
        x = nmp.linalg.solve(A, b.ravel())  # Solve Ax = b numerically (returns 1D
    else:  # Singular case: determinant is zero (within tolerance)
        if rankA != rankM:  # Different ranks imply inconsistency by Rouché–Capelli
            classification = "No solution (inconsistent)"  # Inconsistent system cl
            x = None  # No solution vector to return
        else:  # Equal ranks < n imply infinitely many solutions
            classification = "Infinitely many solutions"  # Under-determined but co
            x = None  # Not computing a parameterization here; classification is su

    print(f"{name}")  # Header line for readability
    print("A =")  # Label for coefficient matrix
    print(A)  # Display A
    print("b =")  # Label for RHS vector
    print(b.ravel())  # Display b as 1D array for readability
    print("Augmented M = [A | b] =")  # Label for augmented matrix
    print(M)  # Display M
    print(f"det(A) = {detA:.6g}")  # Print determinant in compact format
    print(f"rank(A) = {rankA}, rank(M) = {rankM}")  # Print ranks of A and M
    print(f"Classification: {classification}")  # Print final classification
    if x is not None:  # If a unique solution exists
        print("Unique solution x =")  # Label for solution vector
        print(x)  # Print the solution vector
    print()  # Blank line separation between systems
```

```
# Define systems (a), (b), (c)

# System (a):
#   x1 + x2 + x3 = 3
#   x1 − x2 + 2x3 = 2
#   2x1 + 3x3 = 1
A_a = nmp.array([[1,  1, 1],   # Coefficients of eqn 1
                 [1, -1, 2],   # Coefficients of eqn 2
                 [2,  0, 3]],  # Coefficients of eqn 3
                dtype=float)   # Use float dtype
b_a = nmp.array([3, 2, 1], dtype=float)  # Right-hand side values

# System (b):
#   x1 + x2 + x3 = 3
#   x1 − x2 + 2x3 = 2
#   x2 + x3 = 2
A_b = nmp.array([[1,  1, 1],   # Coefficients of eqn 1
                 [1, -1, 2],   # Coefficients of eqn 2
                 [0,  1, 1]],  # Coefficients of eqn 3
                dtype=float)   # Use float dtype
b_b = nmp.array([3, 2, 2], dtype=float)  # Right-hand side values

# System (c):
#   x1 + x2 + x3 = 3
#   x1 − x2 + 2x3 = 2
#   2x1 + 3x3 = 5
A_c = nmp.array([[1,  1, 1],   # Coefficients of eqn 1
                 [1, -1, 2],   # Coefficients of eqn 2
                 [2,  0, 3]],  # Coefficients of eqn 3
                dtype=float)   # Use float dtype
b_c = nmp.array([3, 2, 5], dtype=float)  # Right-hand side values

# Run analyses

analyze_system(A_a, b_a, name="System (a)")  # Analyze and print diagnostics for sy
analyze_system(A_b, b_b, name="System (b)")  # Analyze and print diagnostics for sy
analyze_system(A_c, b_c, name="System (c)")  # Analyze and print diagnostics for sy
```

```
System (a)
A =
[[ 1.  1.  1.]
 [ 1. -1.  2.]
 [ 2.  0.  3.]]
b =
[3. 2. 1.]
Augmented M = [A | b] =
[[ 1.  1.  1.  3.]
 [ 1. -1.  2.  2.]
 [ 2.  0.  3.  1.]]
det(A) = 0
rank(A) = 2, rank(M) = 3
Classification: No solution (inconsistent)

System (b)
A =
[[ 1.  1.  1.]
 [ 1. -1.  2.]
 [ 0.  1.  1.]]
b =
[3. 2. 2.]
Augmented M = [A | b] =
[[ 1.  1.  1.  3.]
 [ 1. -1.  2.  2.]
 [ 0.  1.  1.  2.]]
det(A) = -3
rank(A) = 3, rank(M) = 3
Classification: Unique solution
Unique solution x =
[1. 1. 1.]

System (c)
A =
[[ 1.  1.  1.]
 [ 1. -1.  2.]
 [ 2.  0.  3.]]
b =
[3. 2. 5.]
Augmented M = [A | b] =
[[ 1.  1.  1.  3.]
 [ 1. -1.  2.  2.]
 [ 2.  0.  3.  5.]]
det(A) = 0
rank(A) = 2, rank(M) = 2
Classification: Infinitely many solutions
```

# Question 4

CMU Africa is finalizing admissions for four master's programs:
- MS in Information Technology (IT) - MS in Electrical and Computer Engineering (ECE) - MS in Artificial Intelligence (AI) - MS in ECE–Advanced Studies

The admissions office must meet some constraints.

Let $x_1$, $x_2$, $x_3$, and $x_4$ represent the number of students admitted to IT, ECE, AI, and ECE–Advanced respectively.

## (a) Formulate the system of linear equations that models these four constraints and write the corresponding augmented matrix

**Solution:**

From the question:

$$\text{IT} = x_1, \quad \text{ECE} = x_2, \quad \text{AI} = x_3, \quad \text{ECE–Adv} = x_4$$

Total admitted student is capped at 10 students; hence total = 10:

$$x_1 + x_2 + x_3 + x_4 = 10 \quad \text{(i)}$$

Similarly; faculty supervision is capped at 23:

$$x_1 + 2x_2 + 3x_3 + 4x_4 = 23$$

Also; tutoring is capped at 17 units:

$$2x_1 + x_2 + 2x_3 + x_4 = 17$$

Finally; laboratory is capped at 9 seats:

$$x_1 + 0x_2 + x_3 + 2x_4 = 9$$

System matrix becomes:

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 \\ 2 & 1 & 2 & 1 \\ 1 & 0 & 1 & 9 \end{bmatrix}$$

Augmented matrix:

$$\left[\begin{array}{cccc|c} 1 & 1 & 1 & 1 & 10 \\ 1 & 2 & 3 & 4 & 23 \\ 2 & 1 & 2 & 1 & 17 \\ 1 & 0 & 1 & 2 & 9 \end{array}\right]$$

### (b) Use Gaussian elimination to reduce the matrix to row echelon form (REF) and Gauss–Jordan elimination to find the reduced row echelon form (RREF)

**Solution:**

Augment of the matrix:

$$\left[\begin{array}{cccc|c} 1 & 1 & 1 & 1 & 10 \\ 1 & 2 & 3 & 4 & 23 \\ 2 & 1 & 2 & 1 & 17 \\ 1 & 0 & 1 & 2 & 9 \end{array}\right]$$

Using Gaussian elimination to reduce to REF:

Let $R_2^i = R_2 - R_1$

$$\left[\begin{array}{cccc|c} 1 & 1 & 1 & 1 & 10 \\ 0 & 1 & 2 & 3 & 13 \\ 2 & 1 & 2 & 1 & 17 \\ 1 & 0 & 1 & 2 & 9 \end{array}\right]$$

Let $R_3^i = R_3 - 2R_1$

$$\left[\begin{array}{cccc|c} 1 & 1 & 1 & 1 & 10 \\ 0 & 1 & 2 & 3 & 13 \\ 0 & -1 & 0 & -1 & -3 \\ 1 & 0 & 1 & 2 & 9 \end{array}\right]$$

Let $R_4^i = R_4 - R_1$

$$\left[\begin{array}{cccc|c} 1 & 1 & 1 & 1 & 10 \\ 0 & 1 & 2 & 3 & 13 \\ 0 & -1 & 0 & -1 & -3 \\ 0 & -1 & 0 & 1 & -1 \end{array}\right]$$

Let $R_3^{ii} = R_3 + R_2$

$$\left[\begin{array}{cccc|c} 1 & 1 & 1 & 1 & 10 \\ 0 & 1 & 2 & 3 & 13 \\ 0 & 0 & 2 & 2 & 10 \\ 0 & -1 & 0 & 1 & -1 \end{array}\right]$$

Let $R_4^{ii} = R_4 + R_2$

$$\left[\begin{array}{cccc|c} 1 & 1 & 1 & 1 & 10 \\ 0 & 1 & 2 & 3 & 13 \\ 0 & 0 & 2 & 2 & 10 \\ 0 & 0 & 2 & 4 & 12 \end{array}\right]$$

Let $R_4^{iii} = R_4 - R_3$

$$\left[\begin{array}{cccc|c} 1 & 1 & 1 & 1 & 10 \\ 0 & 1 & 2 & 3 & 13 \\ 0 & 0 & 2 & 2 & 10 \\ 0 & 0 & 0 & 2 & 2 \end{array}\right]$$

Next we use Gauss–Jordan to reduce to RREF:

Let $R_4^{iv} = 1/2R_4$

$$\begin{bmatrix} 1 & 1 & 1 & 1 & | & 10 \\ 0 & 1 & 2 & 3 & | & 13 \\ 0 & 0 & 2 & 2 & | & 10 \\ 0 & 0 & 0 & 1 & | & 1 \end{bmatrix}$$

Let $R_1^i = R_1 - R_2$

$$\begin{bmatrix} 1 & 0 & -1 & -2 & | & -3 \\ 0 & 1 & 2 & 3 & | & 13 \\ 0 & 0 & 2 & 2 & | & 10 \\ 0 & 0 & 0 & 1 & | & 1 \end{bmatrix}$$

Let $R_3^{iii} = 1/2R_3$

$$\begin{bmatrix} 1 & 0 & -1 & -2 & | & -3 \\ 0 & 1 & 2 & 3 & | & 13 \\ 0 & 0 & 1 & 1 & | & 5 \\ 0 & 0 & 0 & 1 & | & 1 \end{bmatrix}$$

Let $R_2^{ii} = R_2 - 2R_3$

$$\begin{bmatrix} 1 & 0 & -1 & -2 & | & -3 \\ 0 & 1 & 0 & 1 & | & 3 \\ 0 & 0 & 1 & 1 & | & 5 \\ 0 & 0 & 0 & 1 & | & 1 \end{bmatrix}$$

Let $R_1^{ii} = R_1 + R_3$

$$\begin{bmatrix} 1 & 0 & 0 & -1 & | & 2 \\ 0 & 1 & 0 & 1 & | & 3 \\ 0 & 0 & 1 & 1 & | & 5 \\ 0 & 0 & 0 & 1 & | & 1 \end{bmatrix}$$

Let $R_3^{iv} = R_3 - R_4$

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

Let $R_1^{iii} = R_4 + R_1$

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

Let $R_2^{iii} = R_2 - R_4$

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

Final RREF:

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

## (c) Uniqueness and Interpretation

The solution to the matrix is given by:

$$x_1 = 3, \quad x_2 = 2, \quad x_3 = 4, \quad x_4 = 1$$

This means there is a unique solution to the problem facing the admission team

The result of our calculation means the admission team should admit students in this manner:

$x_1$ = IT, hence admit 3 IT students
$x_2$ = ECE, hence admit 2 ECE students
$x_3$ = AI, hence admit 4 AI students
$x_4$ = ECE–Advanced, hence admit 1 ECE–Advanced student

## (d) Using Numpy Library

Let:

$$A = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 \\ 2 & 1 & 2 & 1 \\ 1 & 0 & 1 & 2 \end{bmatrix}, \quad b = \begin{bmatrix} 10 \\ 23 \\ 17 \\ 9 \end{bmatrix}$$

Since $\det(A) \neq 0$ and $\text{rank}(A) = \text{rank}(M) = 4$, this means there is a unique solution.
Using ‘np.linalg.solve(A, b)‘ returns:

$$x_1 = 3, \quad x_2 = 2, \quad x_3 = 4, \quad x_4 = 1$$

Which corresponds to our calculation.

Question 4(d):Using python NumPy library and verify your manual solution. Define the coefficient matrix A and constant vector b for your system. Use np.linalg.solve(A, b) to compute the solution vector. Confirm the accuracy of your manual results by comparing them with this computed solution. Export your Python code as a PDF file with solution and submit

In [1]:

```
# Question 4(d): Verify admissions linear system with NumPy  # High-level descripti

import numpy as nmp  # Import NumPy with preferred alias 'nmp'

# Build coefficient matrix A and RHS vector b

A = nmp.array([  # Define the 4x4 coefficient matrix A
    [1, 1, 1, 1],   # Eq (1): total students: x1 + x2 + x3 + x4 = 10
    [1, 2, 3, 4],   # Eq (2): faculty units: 1*x1 + 2*x2 + 3*x3 + 4*x4 = 23
    [2, 1, 2, 1],   # Eq (3): tutoring hours: 2*x1 + 1*x2 + 2*x3 + 1*x4 = 17
    [1, 0, 1, 2]    # Eq (4): lab seats: 1*x1 + 0*x2 + 1*x3 + 2*x4 = 9
], dtype=float)  # Use float for stable linear algebra

b = nmp.array([10, 23, 17, 9], dtype=float)  # Define RHS vector b with the capacit

# Determinant, ranks, classification

detA = nmp.linalg.det(A)  # Compute determinant of A (nonzero => invertible => uniq
rankA = nmp.linalg.matrix_rank(A)  # Compute rank of A
M = nmp.concatenate((A, b.reshape(-1, 1)), axis=1)  # Form augmented matrix M = [A
rankM = nmp.linalg.matrix_rank(M)  # Compute rank of augmented matrix

# Solve (since A is expected to be invertible)

x = nmp.linalg.solve(A, b)  # Solve A x = b for the solution vector x

# Print results clearly

print("Coefficient matrix A:\n", A)  # Show A
print("\nRHS vector b:\n", b)  # Show b
print("\nAugmented matrix M = [A | b]:\n", M)  # Show augmented matrix

print(f"\nDeterminant det(A) = {detA:.6f}")  # Print determinant
print(f"rank(A) = {rankA}, rank(M) = {rankM}")  # Print ranks

classification = "Unique solution" if abs(detA) > 1e-12 else ("Consistent (infinite
print(f"Classification: {classification}")  # Show classification

print("\nSolution vector x (order: [x1, x2, x3, x4]):")  # Label for solution
print(x)  # Print numeric solution

# Sanity check: verify A @ x equals b

print("\nA @ x:\n", A @ x)  # Compute product A x
print("Allclose(A @ x, b)?", nmp.allclose(A @ x, b))  # Check numerical equality wi
```

```
Coefficient matrix A:
 [[1. 1. 1. 1.]
 [1. 2. 3. 4.]
 [2. 1. 2. 1.]
 [1. 0. 1. 2.]]

RHS vector b:
 [10. 23. 17.  9.]

Augmented matrix M = [A | b]:
 [[ 1.  1.  1.  1. 10.]
 [ 1.  2.  3.  4. 23.]
 [ 2.  1.  2.  1. 17.]
 [ 1.  0.  1.  2.  9.]]

Determinant det(A) = 4.000000
rank(A) = 4, rank(M) = 4
Classification: Unique solution

Solution vector x (order: [x1, x2, x3, x4]):
[3. 2. 4. 1.]

A @ x:
 [10. 23. 17.  9.]
Allclose(A @ x, b)? True
```

# Question 5

An overdetermined system arises in linear regression for house-price prediction when there are more data points than model parameters.

Consider the dataset with features (size in square feet, number of bedrooms) and corresponding prices:

- (1000, 2) $\to$ \$200K - (1500, 3) $\to$ \$300K - (1200, 2) $\to$ \$240K - (2000, 4) $\to$ \$400K - (1800, 3) $\to$ \$350K

Model: Price $= w_0 + w_1 \cdot \text{size} + w_2 \cdot \text{bedrooms}$ where $w_0$ is the bias term and $x_0 = 1$ for all observations.

## (a) Form the overdetermined system $Xw = y$ where $X$ is a $5 \times 3$ matrix that includes the bias column

**Solution:**

The matrix $X$ is:

$$X = \begin{bmatrix} 1 & 1000 & 2 \\ 1 & 1500 & 3 \\ 1 & 1200 & 2 \\ 1 & 2000 & 4 \\ 1 & 1800 & 3 \end{bmatrix} \quad y = \begin{bmatrix} 200 \\ 300 \\ 240 \\ 400 \\ 350 \end{bmatrix}$$

## (b) Because the system is overdetermined it will generally have no exact solution, compute the ranks of $X$ and of the augmented matrix $[X|y]$ to confirm whether the system is inconsistent

**Solution:**

To find the rank of $X$, we need to reduce $X$ to REF since rank $X$ is the number of non-zero rows in the REF.

Initial matrix:

$$X = \begin{pmatrix} 1 & 1000 & 2 \\ 1 & 1500 & 3 \\ 1 & 1200 & 2 \\ 1 & 2000 & 4 \\ 1 & 1800 & 3 \end{pmatrix}$$

We start by pivoting $R_{11}$

Let $R_2 = R_2 - R_1$:

$$\begin{pmatrix} 1 & 1000 & 2 \\ 0 & 500 & 1 \\ 1 & 1200 & 2 \\ 1 & 2000 & 4 \\ 1 & 1800 & 3 \end{pmatrix}$$

Let $R_3 = R_3 - R_1$:

$$\begin{pmatrix} 1 & 1000 & 2 \\ 0 & 500 & 1 \\ 0 & 200 & 0 \\ 1 & 2000 & 4 \\ 1 & 1800 & 3 \end{pmatrix}$$

Let $R_4 = R_4 - R_1$:

$$\begin{pmatrix} 1 & 1000 & 2 \\ 0 & 500 & 1 \\ 0 & 200 & 0 \\ 0 & 1000 & 2 \\ 1 & 1800 & 3 \end{pmatrix}$$

Let $R_5 = R_5 - R_1$:

$$\begin{pmatrix} 1 & 1000 & 2 \\ 0 & 500 & 1 \\ 0 & 200 & 0 \\ 0 & 1000 & 2 \\ 0 & 800 & 1 \end{pmatrix}$$

Next we pivot $X_{22}$:

Let $R_3' = 5R_3 - 2R_2$:

$$\begin{pmatrix} 1 & 1000 & 2 \\ 0 & 500 & 1 \\ 0 & 0 & -2 \\ 0 & 1000 & 2 \\ 0 & 800 & 1 \end{pmatrix}$$

Let $R_4' = R_4 - 2R_2$:

$$\begin{pmatrix} 1 & 1000 & 2 \\ 0 & 500 & 1 \\ 0 & 0 & -2 \\ 0 & 0 & 0 \\ 0 & 800 & 1 \end{pmatrix}$$

Let $R_5' = 5R_5 - 8R_2$:

$$\begin{pmatrix} 1 & 1000 & 2 \\ 0 & 500 & 1 \\ 0 & 0 & -2 \\ 0 & 0 & 0 \\ 0 & 0 & -3 \end{pmatrix}$$

Pivot $X_{33}$:

Let $R_5'' = 2R_5 - 3R_3$:

$$\begin{pmatrix} 1 & 1000 & 2 \\ 0 & 500 & 1 \\ 0 & 0 & -2 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

Final REF matrix:

$$\begin{pmatrix} 1 & 1000 & 2 \\ 0 & 500 & 1 \\ 0 & 0 & -2 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

Since the number of non-zero rows in REF is 3:

$$\text{Rank}(X) = 3$$

Now we compute the REF of the augmented matrix $[X|y]$:

## (b) The rank of the augmented matrix $[X|y]$

Initial augmented matrix:

$$\left[\begin{array}{ccc|c} 1 & 1000 & 2 & 200 \\ 1 & 1500 & 3 & 300 \\ 1 & 1200 & 2 & 240 \\ 1 & 2000 & 4 & 400 \\ 1 & 1800 & 3 & 350 \end{array}\right]$$

Again, rank $[X|y]$ is the number of non-zero rows in the REF.

Let $R_2 = R_2 - R_1$:

$$\left[\begin{array}{ccc|c} 1 & 1000 & 2 & 200 \\ 0 & 500 & 1 & 100 \\ 1 & 1200 & 2 & 240 \\ 1 & 2000 & 4 & 400 \\ 1 & 1800 & 3 & 350 \end{array}\right]$$

Let $R_3 = R_3 - R_1$:

$$\left[\begin{array}{ccc|c} 1 & 1000 & 2 & 200 \\ 0 & 500 & 1 & 100 \\ 0 & 200 & 0 & 40 \\ 1 & 2000 & 4 & 400 \\ 1 & 1800 & 3 & 350 \end{array}\right]$$

Let $R_4 = R_4 - R_1$:

$$\left[\begin{array}{ccc|c} 1 & 1000 & 2 & 200 \\ 0 & 500 & 1 & 100 \\ 0 & 200 & 0 & 40 \\ 0 & 1000 & 2 & 200 \\ 1 & 1800 & 3 & 350 \end{array}\right]$$

Let $R_5 = R_5 - R_1$:

$$\left[\begin{array}{ccc|c} 1 & 1000 & 2 & 200 \\ 0 & 500 & 1 & 100 \\ 0 & 200 & 0 & 40 \\ 0 & 1000 & 2 & 200 \\ 0 & 800 & 1 & 150 \end{array}\right]$$

Pivot $R_{22}$: Let $R_3' = 5R_3 - 2R_2$:

$$\left[\begin{array}{ccc|c} 1 & 1000 & 2 & 200 \\ 0 & 500 & 1 & 100 \\ 0 & 1000 & -2 & 0 \\ 0 & 1000 & 2 & 200 \\ 0 & 800 & 1 & 150 \end{array}\right]$$

Let $R_4' = R_4 - 2R_2$:

$$\left[\begin{array}{ccc|c} 1 & 1000 & 2 & 200 \\ 0 & 500 & 1 & 100 \\ 0 & 0 & -2 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 800 & 1 & 150 \end{array}\right]$$

Let $R_5' = 5R_5 - 8R_2$:

$$\left[\begin{array}{ccc|c} 1 & 1000 & 2 & 200 \\ 0 & 500 & 1 & 100 \\ 0 & 0 & -2 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & -3 & -50 \end{array}\right]$$

Finally, pivot $r_{33}$, Let $R_5'' = 2R_5 - 3R_2$:

$$\left[\begin{array}{ccc|c} 1 & 1000 & 2 & 200 \\ 0 & 500 & 1 & 100 \\ 0 & 0 & -2 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -100 \end{array}\right]$$

The number of non-zero rows is 4:

$$\text{Rank}([X|y]) = 4$$

Since $\text{Rank}([X|y]) > \text{Rank}(X)$, the system has no exact solution and is inconsistent.

**(c) Derive the normal equations $X^TXw = X^Ty$ by hand. Solve for $w$ using Gaussian elimination on the resulting 3×3 system, showing all workings as you reduce to Row Echelon Form (REF) and then to Reduced Row Echelon Form (RREF)**

**Solution**

$$X = \begin{bmatrix} 1 & 1000 & 2 \\ 1 & 1500 & 3 \\ 1 & 1200 & 2 \\ 1 & 2000 & 4 \\ 1 & 800 & 3 \end{bmatrix} \quad X^T = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1000 & 1500 & 1200 & 2000 & 1800 \\ 2 & 3 & 2 & 4 & 3 \end{bmatrix}$$

$X^TX$ is defined since the number of columns in $X^T$ is the same as the number of rows in $X$:

$$X^TX = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1000 & 1500 & 1200 & 2000 & 1800 \\ 2 & 3 & 2 & 4 & 3 \end{bmatrix} \begin{bmatrix} 1 & 1000 & 2 \\ 1 & 1500 & 3 \\ 1 & 1200 & 2 \\ 1 & 2000 & 4 \\ 1 & 800 & 3 \end{bmatrix}$$

$$\begin{aligned}
X^TX_{11} &= 1 \times 1 + 1 \times 1 + 1 \times 1 + 1 \times 1 + 1 \times 1 = 5 \\
X^TX_{12} &= 1 \times 1000 + 1 \times 1500 + 1 \times 1200 + 1 \times 2000 + 1 \times 1800 = 7500 \\
X^TX_{13} &= 1 \times 2 + 1 \times 3 + 1 \times 2 + 1 \times 4 + 1 \times 3 = 14 \\
X^TX_{21} &= 1000 \times 1 + 1500 \times 1 + 1200 \times 1 + 2000 \times 1 + 1800 \times 1 = 7500 \\
X^TX_{22} &= 1000 \times 1000 + 1500 \times 1500 + 1200 \times 1200 + 2000^2 + 1800^2 = 11,930,000 \\
X^TX_{23} &= 1000 \times 2 + 1500 \times 3 + 1200 \times 2 + 2000 \times 4 + 1800 \times 3 = 22,300 \\
X^TX_{31} &= 2 \times 1 + 3 \times 1 + 2 \times 1 + 4 \times 1 + 3 \times 1 = 14 \\
X^TX_{32} &= 2 \times 1000 + 3 \times 1500 + 2 \times 1200 + 4 \times 2000 + 3 \times 1800 = 22,300 \\
X^TX_{33} &= 2^2 + 3^2 + 2^2 + 4^2 + 3^2 = 42
\end{aligned}$$

Finally,

$$X^TX = \begin{bmatrix} X^TX_{11} & X^TX_{12} & X^TX_{13} \\ X^TX_{21} & X^TX_{22} & X^TX_{23} \\ X^TX_{31} & X^TX_{32} & X^TX_{33} \end{bmatrix} = \begin{bmatrix} 5 & 7500 & 14 \\ 7500 & 11,930,000 & 22,300 \\ 14 & 22,300 & 42 \end{bmatrix} \quad (3 \times 3 \text{ matrix})$$

$$X^TY = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1000 & 1500 & 1200 & 2000 & 1800 \\ 2 & 3 & 2 & 4 & 3 \end{bmatrix} \begin{bmatrix} 200 \\ 300 \\ 240 \\ 400 \\ 350 \end{bmatrix}$$

$X^TY_1 = 1\times 200 + 1\times 300 + 1\times 240 + 1\times 400 + 1\times 350 = 1,490$
$X^TY_2 = 1000\times 200 + 1500\times 300 + 1200\times 240 + 2000\times 400 + 1800\times 350 = 2,368,000$
$X^TY_3 = 2\times 200 + 3\times 300 + 2\times 240 + 4\times 400 + 3\times 350 = 4,430$

$$X^TY = \begin{bmatrix} 1490 \\ 2,368,000 \\ 4,430 \end{bmatrix}$$

The augmented system: $(X^TX \mid X^TY)$

$$= \begin{bmatrix} 5 & 7500 & 14 & | & 1,490 \\ 7500 & 11,930,000 & 22,300 & | & 2,368,000 \\ 14 & 22,300 & 42 & | & 4,430 \end{bmatrix}$$

Let's find the REF:

$$\text{Pivot } R_1 \to R_2 = \tfrac{1}{100}R_2 \to \begin{bmatrix} 5 & 7500 & 14 & | & 1,490 \\ 75 & 119,300 & 223 & | & 23,680 \\ 14 & 22,300 & 42 & | & 4,430 \end{bmatrix}$$

$$R_2' = R_2 - 15R_1 = \begin{bmatrix} 5 & 7500 & 14 & | & 1,490 \\ 0 & 6,800 & 13 & | & 1,330 \\ 14 & 22,300 & 42 & | & 4,430 \end{bmatrix}$$

$$R_3 = 5R_3 - 14R_1 \to \begin{bmatrix} 5 & 7500 & 14 & | & 1,490 \\ 0 & 6,800 & 13 & | & 1,330 \\ 0 & 6,500 & 14 & | & 1,290 \end{bmatrix}$$

$$R_3' = R_3 \times 6800 - 6500R_2 \to \begin{bmatrix} 5 & 7500 & 14 & | & 1,490 \\ 0 & 6,800 & 13 & | & 1,330 \\ 0 & 0 & 10,700 & | & 127,000 \end{bmatrix}$$

Next we compute RREF:

$$R_3'' = \frac{1}{100}R_3 \to \begin{bmatrix} 5 & 7500 & 14 & | & 1,490 \\ 0 & 6,800 & 13 & | & 1,330 \\ 0 & 0 & 107 & | & 1,270 \end{bmatrix}$$

$$R_1 = \frac{1}{5}R_1 \to \begin{bmatrix} 1 & 1500 & \frac{14}{5} & | & 298 \\ 0 & 6,800 & 13 & | & 1,330 \\ 0 & 0 & 107 & | & 1,270 \end{bmatrix}$$

Make the second pivot 1 as well:

$$R_2 = \frac{1}{6800}R_2 \to \begin{bmatrix} 1 & 1500 & \frac{14}{5} & | & 298 \\ 0 & 1 & \frac{13}{6800} & | & \frac{1330}{6800} \\ 0 & 0 & 107 & | & 1,270 \end{bmatrix}$$

$$R_3 = \frac{1}{107}R_3 \rightarrow \begin{bmatrix} 1 & 1500 & \frac{14}{5} & | & 298 \\ 0 & 1 & \frac{13}{6800} & | & \frac{1330}{6800} \\ 0 & 0 & 1 & | & \frac{1270}{107} \end{bmatrix}$$

$$R_1 = R_1 - 1500R_2 \rightarrow \begin{bmatrix} 1 & 0 & -\frac{23}{340} & | & \frac{157}{34} \\ 0 & 1 & \frac{13}{6800} & | & \frac{1330}{6800} \\ 0 & 0 & 1 & | & \frac{1270}{107} \end{bmatrix}$$

$$R_1 = R_1 + \frac{23}{340}R_3 \rightarrow \begin{bmatrix} 1 & 0 & 0 & | & \frac{580}{107} \\ 0 & 1 & \frac{13}{6800} & | & \frac{133}{680} \\ 0 & 0 & 1 & | & \frac{1270}{107} \end{bmatrix}$$

$$R_2 = R_2 - \frac{13}{6800}R_3 \rightarrow \begin{bmatrix} 1 & 0 & 0 & | & \frac{580}{107} \\ 0 & 1 & 0 & | & \frac{37}{214} \\ 0 & 0 & 1 & | & \frac{1270}{107} \end{bmatrix}$$

$$R_2 \rightarrow \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{vmatrix} 580/107 \\ 37/214 \\ 1270/107 \end{vmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{vmatrix} 5.42 \\ 0.173 \\ 11.87 \end{vmatrix}$$

Hence RREF is gotten. Next we solve for $w_0, w_1$ and $w_2$:

$$w_0 = \frac{580}{107}, \quad w_1 = \frac{37}{214}, \quad w_2 = \frac{1270}{107}$$

$$w = [w_0, w_1, w_2]^T = \begin{bmatrix} 5.42 \\ 0.173 \\ 11.87 \end{bmatrix} \quad \begin{aligned} w_0 &= \text{bias term in } \$x \\ w_1 &= \text{Price increase by } \$173 \text{ per square foot} \\ w_2 &= \text{Price increase by } \$11.87\text{k per bedroom} \end{aligned}$$

**d.** Interpret the solution geometrically as the orthogonal projection of $y$ onto the column space of $X$. Explain how this projection corresponds to minimizing the squared error in machine learning and compute the residual error $\|y - Xw\|_2$. Finally verify your result using Python.

**Solution**

Recall that

$$X = \begin{bmatrix} 1 & 1000 & 2 \\ 1 & 1500 & 3 \\ 1 & 1200 & 2 \\ 1 & 2000 & 4 \\ 1 & 1800 & 3 \end{bmatrix} \quad \textit{and} \quad w = \begin{bmatrix} 5.42 \\ 0.173 \\ 11.87 \end{bmatrix}$$

Also $\hat{y}_i = w_0 + w_1 \cdot \text{size} + w_2 \cdot \text{bedroom}$, hence the least squared regression model is given as:

$\hat{y}_i = 5.42 + 0.173 \cdot \text{size} + 11.87 \cdot \text{bedroom}$

Therefore:

$$\begin{aligned}
(1000, 2) : \hat{y}_1 &= 5.42 + 0.173 \times 1000 + 11.87 \times 2 = 202.159 \\
(1500, 3) : \hat{y}_2 &= 5.42 + 0.173 \times 1500 + 11.87 \times 3 = 300.528 \\
(1200, 2) : \hat{y}_3 &= 5.42 + 0.173 \times 1200 + 11.87 \times 2 = 236.759 \\
(2000, 4) : \hat{y}_4 &= 5.42 + 0.173 \times 2000 + 11.87 \times 4 = 398.897 \\
(1800, 3) : \hat{y}_5 &= 5.42 + 0.173 \times 1800 + 11.87 \times 3 = 352.428
\end{aligned}$$

Hence

$$\hat{y} = [202.159, 300.528, 236.759, 398.897, 352.428]$$

Residual error $r = y - \hat{y}$, and $y = [200, 300, 240, 400, 350]$

$$r = [200, 300, 240, 400, 350] - [202.159, 300.528, 236.759, 398.897, 352.428]$$

$$r = [200 - 202.159, 300 - 300.528, 240 - 236.759, 400 - 398.897, 350 - 352.428]$$

$$r = [-2.159, -0.528, 3.241, 1.103, -2.428]$$

Next we compute squared norm $\|r\|_2^2$:

$$\begin{aligned}
\|r\|_2^2 &= (-2.159)^2 + (-0.528)^2 + (3.241)^2 + (1.103)^2 + (-2.428)^2 \\
&= 4.661 + 0.279 + 10.504 + 1.217 + 5.895 \\
&= 22.556
\end{aligned}$$

From the Python program, the squared norm is 22.43, which is similar to what I calculated; the slight difference is due to approximation errors. This small residual norm indicates a good fit of the model on the data. In machine learning, the loss function is usually the least squared error, which is given by

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 = \frac{1}{n} \|y - Xw\|^2$$

By minimizing this loss we are basically finding the projection of $y$ onto the column space of $X$, because the projection gives the closest vector $Xw$ to $y$ in Euclidean distance. This is done by orthogonally projecting $y$ onto the column space of $X$.

Question 5(d): (d) Interpret the solution geometrically as the orthogonal projection of y onto the column space of X. Explain how this projection corresponds to minimizing the squared error in machine learning, and compute the residual vector ‖y − Xw‖2. Finally, verify your result using Python.

In [1]:
```
# Question 5: Verify Least Squares Solution using Python
# -------------------------------------------------------
# This code performs the following steps:
# 1) Builds X and y from the dataset.
# 2) Computes X^T X and X^T y.
# 3) Solves (X^T X) w = X^T y using manual Gaussian elimination.
# 4) Computes predictions, residuals, and residual norm.

import numpy as nmp  # Use NumPy for array handling

# Step 1: Build X and y

X = nmp.array([
    [1, 1000, 2],
    [1, 1500, 3],
    [1, 1200, 2],
    [1, 2000, 4],
    [1, 1800, 3]
], dtype=float)

y = nmp.array([200, 300, 240, 400, 350], dtype=float).reshape(-1, 1)

# Step 2: Compute X^T X and X^T y

XT_X = X.T @ X
XT_y = X.T @ y

print("X^T X:\n", XT_X)
print("\nX^T y:\n", XT_y)

# Step 3: Manual Gaussian Elimination

def gaussian_elimination(A, b):
    A = A.astype(float).copy()
    b = b.astype(float).copy()
    n = len(b)

    # Forward elimination
    for i in range(n):
        # Normalize pivot row
        pivot = A[i, i]
        A[i] = A[i] / pivot
        b[i] = b[i] / pivot

        # Eliminate below
        for j in range(i + 1, n):
            factor = A[j, i]
            A[j] -= factor * A[i]
```

```python
            b[j] -= factor * b[i]

    # Back substitution
    for i in range(n - 1, -1, -1):
        for j in range(i):
            factor = A[j, i]
            A[j] -= factor * A[i]
            b[j] -= factor * b[i]

    return b

w = gaussian_elimination(XT_X, XT_y)
print("\nSolution vector w (bias, size weight, bedrooms weight):\n", w)

# Step 4: Predictions and Residuals

y_pred = X @ w
residual = y - y_pred
residual_norm_sq = float((residual.T @ residual)[0, 0])

print("\nPredicted y:\n", y_pred)
print("\nResiduals:\n", residual)
print("\nSquared residual norm ||y - Xw||^2:", residual_norm_sq)
```

```
X^T X:
 [[5.000e+00 7.500e+03 1.400e+01]
 [7.500e+03 1.193e+07 2.230e+04]
 [1.400e+01 2.230e+04 4.200e+01]]

X^T y:
 [[1.490e+03]
 [2.368e+06]
 [4.430e+03]]

Solution vector w (bias, size weight, bedrooms weight):
 [[ 5.42056075]
 [ 0.1728972 ]
 [11.86915888]]

Predicted y:
 [[202.05607477]
 [300.37383178]
 [236.63551402]
 [398.69158879]
 [352.24299065]]

Residuals:
 [[-2.05607477]
 [-0.37383178]
 [ 3.36448598]
 [ 1.30841121]
 [-2.24299065]]

Squared residual norm ||y - Xw||^2: 22.429906542056138
```

**(6) Consider the vectors in $\mathbb{R}^4$ that arise from a clustering position problem:**

$$u = [1, 2, 3, 4], \quad v = [2, 3, 4, 5]^T, \quad \omega = [0, 1, -1, 2]^T$$

**a.** compute the L1, L2, and L$\infty$ norms of the vectors. Normalize $u$ and $v$ to unit vectors with respect to the L2 norm.

**Solution**

Let's compute the norms:

$$||u||_1 = \sum |u| = |1| + |2| + |3| + |4| = 1 + 2 + 3 + 4 = 10$$

$$||u||_2 = \sqrt{1^2 + 2^2 + 3^2 + 4^2} = \sqrt{30} = 5.48$$

$$||u||_\infty = \max\{1, 2, 3, 4\} = 4$$

for $v$:

$$||v||_1 = |2| + |3| + |4| + |5| = 2 + 3 + 4 + 5 = 14$$

$$||v||_2 = \sqrt{2^2 + 3^2 + 4^2 + 5^2} = \sqrt{54} = 7.35$$

$$||v||_\infty = \max\{2, 3, 4, 5\} = 5$$

for $w$:

$$||w||_1 = |0| + |1| + |-1| + |2| = 0 + 1 + 1 + 2 = 4$$

$$||w||_2 = \sqrt{0^2 + 1^2 + (-1)^2 + 2^2} = \sqrt{6} = 2.45$$

$$||w||_\infty = \max\{|0|, |1|, |-1|, |2|\} = 2$$

Next we normalize $u, v$ under $L_2$:

$$\hat{u} = \frac{u}{||u||_2} = \frac{[1, 2, 3, 4]}{5.48} = \begin{bmatrix} 0.183, & 0.365, & 0.548, & 0.730 \end{bmatrix}$$

$$\hat{v} = \frac{v}{||v||_2} = \frac{[2, 3, 4, 5]}{7.35} = [0.272, 0.408, 0.566, 0.680]$$

**b.** compute all Pairwise Euclidean distances among $U, V$ and $W$. Set up a system of equations for a point $x \in \mathbb{R}^4$ satisfying $||x - v||_2 = ||x - u||_2 = ||x - w||_2$ (that is a point equidistant from all three.)

**Solution**

Pairwise euclidean distance is given by

$$d(u,v) = \sqrt{(u_1 - v_1)^2 + (u_2 - v_2)^2 + \cdots + (u_n - v_n)^2}$$

$$d(u,v) = \sqrt{(1-2)^2 + (2-3)^2 + (3-4)^2 + (4-5)^2} = \sqrt{1+1+1+1} = \sqrt{4} = 2$$

$$d(u,w) = \sqrt{(u_1 - w_1)^2 + (u_2 - w_2)^2 + \cdots + (u_n - w_n)^2}$$

$$= \sqrt{(1-0)^2 + (2-1)^2 + (3-(-1))^2 + (4-2)^2} = \sqrt{1^2 + 1^2 + 4^2 + 2^2} = \sqrt{22} = 4.69$$

$$d(v,w) = \sqrt{(v_1 - w_1)^2 + (v_2 - w_2)^2 + \cdots + (v_n - w_n)^2}$$

$$= \sqrt{(2-0)^2 + (3-1)^2 + (4-(-1))^2 + (5-2)^2} = \sqrt{2^2 + 2^2 + 5^2 + 3^2} = \sqrt{42} = 6.48$$

**Equidistant point** $x = (x_1, x_2, x_3, x_4)$
Where

$$||x-u||_2 = ||x-v||_2 = ||x-w||_2$$

where

$$u = [1,2,3,4], \quad v = [2,3,4,5], \quad w = [0,1,-1,2]$$

Let's square and subtract $||x-u||^2 - ||x-v||^2 = 0$ and

$$||x-u||^2 - ||x-w||^2 = 0$$

$$||x-u||^2 - ||x-v||^2 = (x_1-1)^2 - (x_1-2)^2 + (x_2-2)^2 - (x_2-3)^2 + (x_3-3)^2 - (x_3-4)^2 + (x_4-4)^2 - (x_4-5)^2$$

Recall that $(a-b)^2 - (a-c)^2 = 2(c-b)a + (b^2 - c^2)$ where $a = x$, $b = u$, $c = v$
Hence:

$$(x_1 - 1)^2 - (x_1 - 2)^2 = 2(2-1)x_1 + (1^2 - 2^2)$$
$$(x_2 - 2)^2 - (x_2 - 3)^2 = 2(3-2)x_2 + (2^2 - 3^2)$$
$$(x_3 - 3)^2 - (x_3 - 4)^2 = 2(4-3)x_3 + (3^2 - 4^2)$$
$$(x_4 - 4)^2 - (x_4 - 5)^2 = 2(5-4)x_4 + (4^2 - 5^2)$$

$$\Rightarrow 2x_1 + 1 - 4 + 2x_2 + 4 - 9 + 2x_3 + 9 - 16 + 2x_4 + 16 - 25 = 0$$

$$= 2x_1 - 3 + 2x_2 - 5 + 2x_3 - 7 + 2x_4 - 9 = 0$$

collect the x terms:

$$2x_1 + 2x_2 + 2x_3 + 2x_4 = 0 + 3 + 5 + 7 + 9$$
$$2(x_1 + x_2 + x_3 + x_4) = 24$$
$$x_1 + x_2 + x_3 + x_4 = \frac{24}{2} = 12$$

second difference is calculated similarly:

$$||x - u||^2 - ||x - w||^2 = 0$$

$$\Rightarrow (x_1-1)^2-(x_1-0)^2+(x_2-2)^2-(x_2-1)^2+(x_3-3)^2-(x_3-(-1))^2+(x_4-4)^2-(x_4-2)^2 = 0$$

Again, recall $(a - b)^2 - (a - c)^2 = 2(c - b)a + (b^2 - c^2)$

$$\Rightarrow (x_1 - 1)^2 - (x_1 - 0)^2 = 2(0 - 1)x_1 + (1^2 - 0^2)$$
$$(x_2 - 2)^2 - (x_2 - 1)^2 = 2(1 - 2)x_2 + (2^2 - 1^2)$$
$$(x_3 - 3)^2 - (x_3 + 1)^2 = 2(-1 - 3)x_3 + (3^2 - (-1)^2)$$
$$(x_4 - 4)^2 - (x_4 - 2)^2 = 2(2 - 4)x_4 + (4^2 - 2^2)$$

$$\Rightarrow -2x_1 + 1 - 2x_2 + 3 - 8x_3 + 8 - 4x_4 + 12 = 0$$

collect the x terms again:

$$-2x_1 - 2x_2 - 8x_3 - 4x_4 = 0 - 1 - 3 - 8 - 12$$
$$-2(x_1 + x_2 + 4x_3 + 2x_4) = -24$$
$$x_1 + x_2 + 4x_3 + 2x_4 = 12$$

In a linear system:

$$x_1 + x_2 + x_3 + x_4 = 12 \quad (1)$$
$$x_1 + x_2 + 4x_3 + 2x_4 = 12 \quad (2)$$

If we subtract (1) from (2):

$$3x_3 + x_4 = 0$$

which means $3x_3 = -x_4$ and

$$x_3 = -\frac{1}{3}x_4$$

and $x_4 = -3x_3$.

Substituting back into (1):

$$x_1 + x_2 - \frac{1}{3}x_4 + x_4 = 12$$

$$x_1 + x_2 + \frac{2}{3}x_4 = 12$$

$$x_1 = 12 - x_2 - \frac{2}{3}x_4$$

Hence

$$x_1 = 12 - x_2 - \frac{2}{3}x_4$$

$$x_2 = 12 - x_1 - \frac{2}{3}x_4$$

$$x_3 = -\frac{1}{3}x_4$$

$$x_4 = -3x_3$$

General solution (X) =

$$\begin{bmatrix} 12 - x_2 - \frac{2}{3}x_4 \\ 12 - x_1 - \frac{2}{3}x_4 \\ -\frac{1}{3}x_4 \\ -3x_3 \end{bmatrix}$$

**6d.** In the context of machine learning, explain how such equidistant points can be related to decision boundaries in k-means clustering or support-Vector machines. Discuss why high dimensional spaces cause norms to behave differently (the "curse of dimensionality") and how it affects distance based algorithms.

**Solution**

1. In k-means clustering, the algorithm assigns each point to the cluster whose centroid is the closest. The decision boundary is the line where two clusters are equally close. This means that if a point is equidistant from two centroids then it lies exactly on the decision boundary of those two clusters.

2. In support vector machines (SVMs), the decision boundary is the hyperplane that separate two classes. Points that are equidistant from this hyperplane are on the margin. These points are what defines the boundary

3. High dimensional spaces and the curse of dimensionality: distances behave strangely in high dimensional planes because all points start to look almost equally far from each other. This means the difference between the nearest and farthest neighbour becomes very small. This happens because when you add more dimensions, the space grows so fast that the data points spread out and norms such as euclidean distances become meaningless.

## References


1. "NumPy Matrix Multiplication — DigitalOcean." Accessed: Sept. 28, 2025. [Online]. Available: https://www.digitalocean.com/community/tutorials/numpy-matrix-multiplication

2. M. P. Deisenroth, A. A. Faisal, and C. S. Ong, "Mathematics for Machine Learning".

3. "Matrix Multiplication in NumPy - GeeksforGeeks." Accessed: Sept. 28, 2025. [Online]. Available: https://www.geeksforgeeks.org/python/matrix-multiplication-in-numpy/

4. "Loss Functions in Deep Learning," GeeksforGeeks. Accessed: Sept. 28, 2025. [Online]. Available: https://www.geeksforgeeks.org/deep-learning/loss-functions-in-deep-learning/

5 "Curse of Dimensionality: Challenges & Solutions in High-Dimensional Data." Accessed: Sept. 28, 2025. [Online]. Available: https://www.statology.org/curse-of-dimensionality-challenges-solutions-high-dimensional-data/

6 "How SVM constructs boundaries? - GeeksforGeeks." Accessed: Sept. 28, 2025. [Online]. Available: https://www.geeksforgeeks.org/machine-learning/how-svm-constructs-boundaries/

7 "K-Means Clustering Algorithm." Accessed: Sept. 28, 2025. [Online]. Available: https://www.analyticsvidhya.com/blog/2019/08/comprehensive-guide-k-means-clustering/