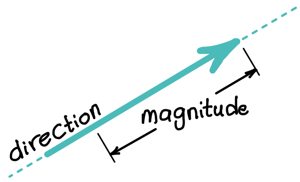
Day 1

What is Scaler?

A quantity that has only magnitude and no direction. It called scaler cause it scales the vector.

What is the Vector?

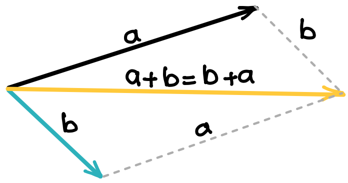


Physics perspective: arrows pointing in space.

- What defines vector is its length, and the direction it's pointing.

Computer science perspective : a list of numbers.

- Vectors are usually viewed by computers as an ordered



Mathematicians generalizes both perspective: a vector can be anything where there's a sensible notion of adding two vectors and multiplying vector by a number.

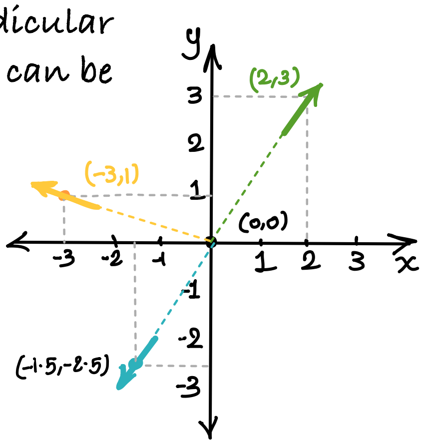
$$\underset{\text{scalar}}{x} \qquad \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \text{ or } \underset{\text{Vector}}{\begin{bmatrix} x_1 & x_2 \end{bmatrix}} \qquad \underset{\text{Matrix}}{\begin{bmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \\ x_{31} & x_{32} \end{bmatrix}} \qquad \underset{\text{Tensor}}{\begin{bmatrix} [x_1 \; x_2] & [x_1 \; x_2] \\ [x_1 \; x_2] & [x_1 \; x_2] \\ [x_1 \; x_2] & [x_1 \; x_2] \end{bmatrix}}$$

What is Coordinate System?

In geometry, it is a system that uses one or more numbers, or coordinates, to uniquely determine the position of the points.

- The coordinate of a point P: is defined as the distance from O to P depending on which side of the line P lies.

- In the plane (Cartesian coordinate system), two perpendicular lines are chosen, any vector from origin to the point that can be represented by two numbers.

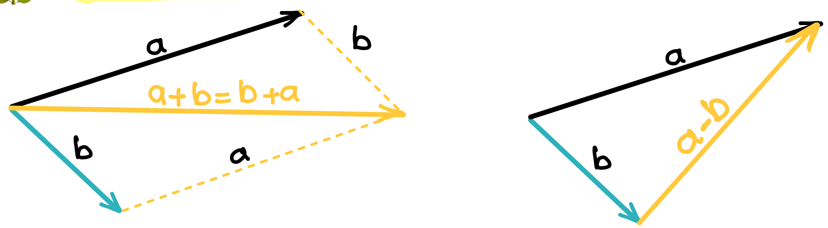


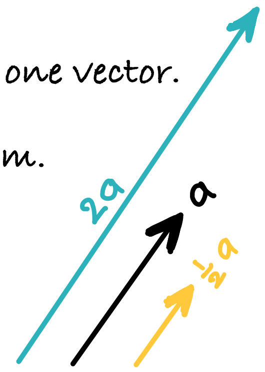
- In three dimensions, three orthogonal planes are chosen.

- This can be generalized to create n coordinates for any point in n-dimensional in Euclidean space. Any vector in this space can be represented by n numbers.

Vector Operations

- **Vector Scaling**: Multiply vector by scaler number
- **Vector addition**: combine two or more vectors together in one vector.
- **Vector substraction**: reverse vector direction then add them.





What is Linear Algebra?

- Linear algebra is the study of vectors and linear functions (linear transformation).
- Linear algebra allows **only** vector **addition** and **scalar multiplication**.

Vector Norms

L1 Norm (Manhattan Norm):

The L1 norm of a vector is the sum of the absolute values of its components.

Use Case: used in Lasso regression for feature selection since it tends to produce sparse models (many coefficients are zero)

$$|\mathbf{x}|_1 = \sum_{r=1}^{n} |x_r|. \qquad \|\mathbf{x}\|_1 = \sum_{i=1}^{N} |x_i| = |x_1| + |x_2| + \ldots + |x_N|$$

L2 Norm (Euclidean Norm):

A square root of the sum of the squared components of the vector.

- Geometric Interpretation: represents the length of the vector from the origin to the point in Euclidean space.
- The L2 norm is often referred to as the Euclidean Distance.
- The L2 norm finds the shortest path from point A to point B

$$||x||_2 = \sqrt{\sum_{i=1}^{n} |x_i|^2} \qquad |\mathbf{x}|_2 = |\mathbf{x}| = \sqrt{x_1^2 + x_2^2 + \ldots + x_n^2}\,.$$

🌻 **Use Cases:** - **Optimization:** The L2 norm is often used in loss functions

- **Evaluation:** Calculating the loss of a ML model by a single number that summarizes the performance.

- **Machine Learning:** The L2 norm is commonly used in Ridge regression (also known as Tikhonov regularization) to prevent overfitting by penalizing large coefficients in the model.

- **Magnitude in Multiple Dimensions:** the length of a data point in multiple dimensions is often calculated using the Euclidean norm (L2 norm), which is the square root of the sum of the squared components.

🌻 L_∞ Norm (Lmax Norm):

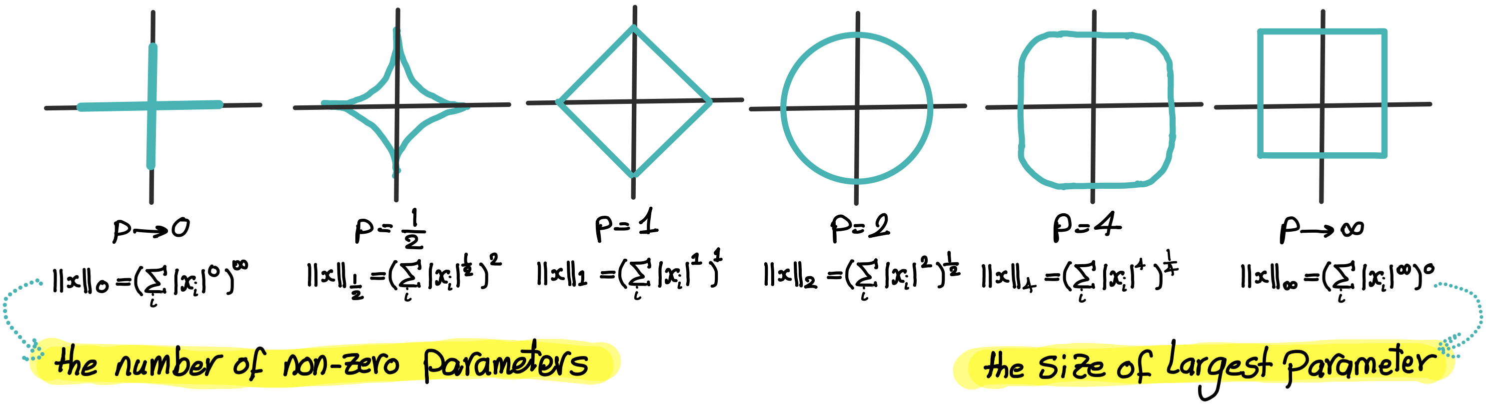
The maximum absolute value among the components of a vector. It measures the greatest distance in any coordinate direction.

$$|\mathbf{x}|_\infty = \max_i |x_i|.$$

🌻 **Use Case:** useful when you need to ensure that no single component dominates the vectorâs behavior, such as in certain optimization problems where the worst-case scenario is important.

🌻 Lp Norm (Generalized Norm):

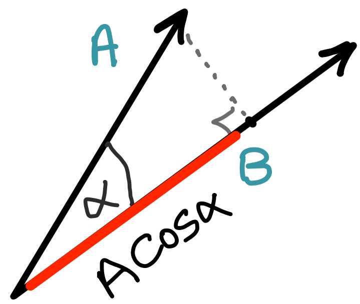
Generalizes the concept of L1 and L2 norms to any positive real number ð. It measures the distance by raising each component to the power of ð, summing them up, and then taking the p-th root.



$$|\mathbf{x}|_p \equiv \left(\sum_i |x_i|^p \right)^{1/p}.$$

What is Dot Product? (Inner product)

Geometric Representation



$|A|\,|B| \cos\alpha$

The dot product of two vectors measures how much they point in the same direction.

Multiply length of the projection of A ($A \cos\alpha$) on B and the length of B

🌻 If magnitude of B ($|B|$) $= 1$

. Dot product is the projection of A on B

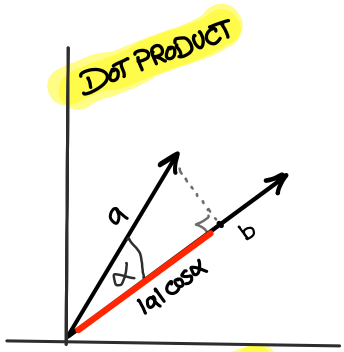
Algebraic Representation

This operation combines the vectors into a single scalar value.

$$A^T = \begin{bmatrix} a_1 & a_2 & a_3 \end{bmatrix} \quad B = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} \quad \begin{bmatrix} a_1 & a_2 & a_3 \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} = a_1 b_1 + a_2 b_2 + a_3 b_3 = A \cdot B$$

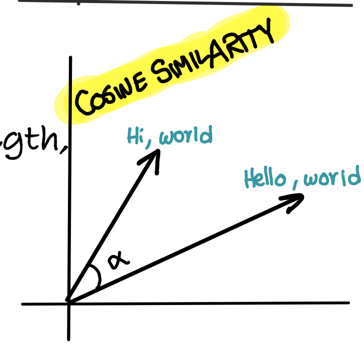
Data Science Applications

Similarity and Distance



The dot product is used to measure the similarity between two vectors. A higher dot product indicates greater similarity.

Cosine Similarity



A normalized dot product that accounts for vector length, making it particularly useful for comparing documents, images, or other high-dimensional data.

Linear Regression

Calculate the dot product of the feature vector and the weight vector.

Neural Networks

The dot product is fundamental to matrix multiplications in neural networks, used for calculating activations and gradients during training.

Principal Component Analysis (PCA)

The dot product is used to project data onto principal components, which are orthogonal directions with the most variance.

Practical part

```python
import numpy as np

# Create vectors
a = np.array([3, 4, 1])
b = np.array([1, 2, 3])

# Scale vector a by a scalar of 2
scalar = 2
scaled_a = scalar * a
print(f"Scaled Vector a: {scaled_a}")

# Add vector a and b
added_ab = a + b
print(f"Added Vector a and b: {added_ab}")

# Subtract vector b from vector a
subtracted_ab = a - b
print(f"Subtracted Vector b from a: {subtracted_ab}")

# Calculate L1 norm for vector a
l1_norm_a = np.linalg.norm(a, 1)
print(f"L1 Norm of vector a: {l1_norm_a}")

# Calculate L2 norm for vector a
l2_norm_a = np.linalg.norm(a)
print(f"L2 Norm of vector a: {l2_norm_a}")

# Calculate L∞ norm for vector a
linf_norm_a = np.linalg.norm(a, np.inf)
print(f"L∞ Norm of vector a: {linf_norm_a}")
```

Matrices

A matrix is a two-dimensional array that has a fixed number of rows and columns and contains a number at the intersection of each row and column.

- Matrices can operate on a vector and transform it to another vector.

$$\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} 3 \\ 2 \end{bmatrix} = \begin{bmatrix} 8 \\ 7 \end{bmatrix}$$

- Matrices can be seen as a linear function applied to a vector (Linear transformation).

$$T_M(V) = T_M\left(\begin{bmatrix} x \\ y \end{bmatrix}\right) = M \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}_{2\times2} \begin{bmatrix} x \\ y \end{bmatrix}_{2\times1} = \begin{bmatrix} ax+by \\ cx+dy \end{bmatrix}_{2\times1}$$

Transformation matrix

$$T_M(z, y) = (ax+by\ ,\ cz+dy)$$ Cartesian form

Operations on Matrices

Most deep-learning computational activities are done through basic matrix operations, such as multiplication, addition, subtraction, transposition... etc.

$$A_{m\times n} \in \mathbb{R}^{m\times n}$$

Addition of Two Matrices

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} + \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix} = \begin{bmatrix} 1+5 & 2+6 \\ 3+7 & 4+8 \end{bmatrix} = \begin{bmatrix} 6 & 8 \\ 10 & 12 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} - \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix} = \begin{bmatrix} 1-5 & 2-6 \\ 3-7 & 4-8 \end{bmatrix} = \begin{bmatrix} -4 & -4 \\ -4 & -4 \end{bmatrix}$$

Product of Two Matrices

$$A \in \mathbb{R}^{m\times n} \qquad B \in \mathbb{R}^{p\times q}$$

$$AB \neq BA \qquad p = m$$

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \times \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix} = \begin{bmatrix} 19 & 22 \\ 43 & 50 \end{bmatrix}$$

$$C_{11} = \begin{bmatrix} 1 & 2 \end{bmatrix}_{1\times2} \times \begin{bmatrix} 5 \\ 7 \end{bmatrix}_{2\times1} = 1\times5 + 2\times7 = 19$$

$$C_{12} = \begin{bmatrix} 1 & 2 \end{bmatrix}_{1\times2} \times \begin{bmatrix} 6 \\ 8 \end{bmatrix}_{2\times1} = 1\times6 + 2\times8 = 22$$

$$C_{21} = \begin{bmatrix} 3 & 4 \end{bmatrix}_{1\times2} \times \begin{bmatrix} 5 \\ 7 \end{bmatrix}_{2\times1} = 3\times5 + 4\times7 = 43$$

$$C_{22} = \begin{bmatrix} 3 & 4 \end{bmatrix}_{1\times2} \times \begin{bmatrix} 6 \\ 8 \end{bmatrix}_{2\times1} = 3\times6 + 4\times8 = 50$$

Transpose of a Matrix

Transposing the column vectors as row vectors.

$A \in \mathbb{R}^{m\times n} \longrightarrow A^T \in \mathbb{R}^{n\times m}$

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

$(AB)^T = A^T B^T$

$(AB)^{-1} = A^{-1} B^{-1}$

Matrix Rank:

The number of linearly independent rows (or columns) in the coefficient matrix (A).

Different Types of Matrices

$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

Identity Matrix (I)

$\begin{bmatrix} a & 0 & 0 & 0 \\ 0 & a & 0 & 0 \\ 0 & 0 & a & 0 \\ 0 & 0 & 0 & a \end{bmatrix}$

Scalar Matrix

$\begin{bmatrix} a & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & c & 0 \\ 0 & 0 & 0 & d \end{bmatrix}$

Diagonal Matrix

$\begin{bmatrix} 0 & 0 & 0 & 0 \\ a & 0 & 0 & 0 \\ b & d & 0 & 0 \\ c & e & f & 0 \end{bmatrix}$

Lower Triangular Matrix (L)

$\begin{bmatrix} 0 & a & c & f \\ 0 & 0 & b & e \\ 0 & 0 & 0 & d \\ 0 & 0 & 0 & 0 \end{bmatrix}$

Upper Triangular Matrix (U)

$\begin{bmatrix} x & a & c & f \\ a & y & b & e \\ b & d & z & d \\ c & e & f & w \end{bmatrix}$

Symmetric Matrix

Pivot

$\begin{bmatrix} a & c & f \\ 0 & b & e \\ 0 & 0 & d \end{bmatrix}$ $\begin{bmatrix} a & c & f \\ 0 & b & e \\ 0 & 0 & d \\ 0 & 0 & 0 \end{bmatrix}$ $\begin{bmatrix} a & b & c & d \\ 0 & e & f & g \\ 0 & 0 & h & i \end{bmatrix}$

Row Echelon Form (REF)

ال Pivot تحته كله أصفار

كل الأصفار تحت

$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ $\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$ $\begin{bmatrix} 1 & 0 & 0 & d \\ 0 & 1 & 0 & g \\ 0 & 0 & 1 & i \end{bmatrix}$

Reduced Row Echelon Form (RREF)

- لو Square matrix تطلع upper matrix
- لو Pivot عبارة عن [1] واللي فوقه وتحته صفر هنا أسميها Reduced Row Echelon Form (RREF)

- Symmetric Matrix saves memory in computation. You need to store only half of it.
- Symmetric Matrix saves memory in computation. You need to store only half of it.
- Symmetric Matrix can be decomposed into matrix of orthonormal eigenvectors and diagonal matrix of eigenvalues.

System of linear equations:

$$A \cdot X = B$$

$$2x + y = 100$$
$$x + 2y = 100$$

Linear algebra make it easy

$$\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 100 \\ 100 \end{bmatrix}$$

$$x\begin{bmatrix} 2 \\ 1 \end{bmatrix} + y\begin{bmatrix} 1 \\ 2 \end{bmatrix} = \begin{bmatrix} 100 \\ 100 \end{bmatrix}$$

So we can find X by

$$X = A^{-1}B$$

How to find the solution of the linear system

$$\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$$ Coefficient matrix

- **More Equations Than Unknowns ($m > n$):**

  Usually no solutions (unless redundant equations exist).

- **Fewer Equations Than Unknowns ($m < n$):**

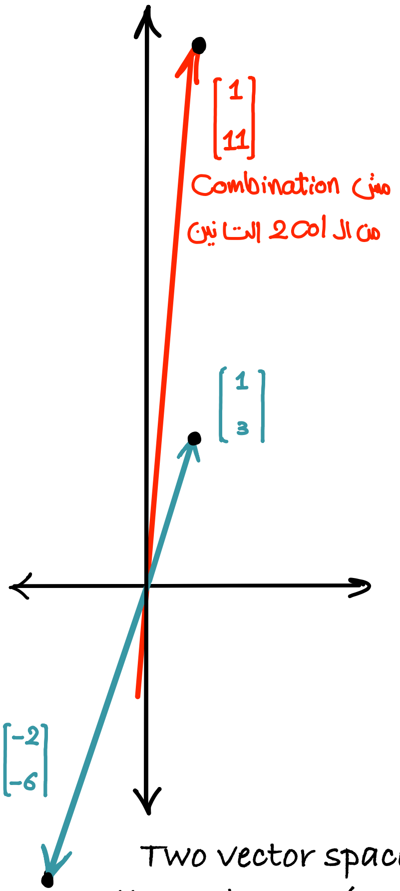
  Usually infinite solutions.

- **Equal Number of Equations and Unknowns ($m = n$):**

  Could have a unique solution, no solution, or infinite solutions.

$$x - 2y = 1$$
$$3x - 6y = 11$$
$$0y = 8$$

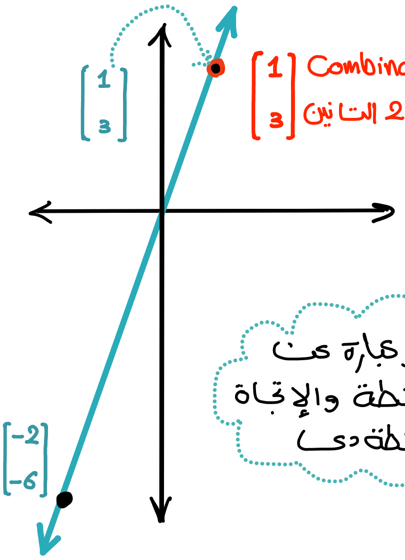
impossible solution



Two vector space is "collapsed" to a single line.

$$x - 2y = 1$$
$$3x - 6y = 3$$
$$0y = 0$$

infinity solution



All vector space is "collapsed" to a single line.

الـ Vector هو عبارة عن
أحداثيات نقطة والإتجاه
من origin للنقطة دى

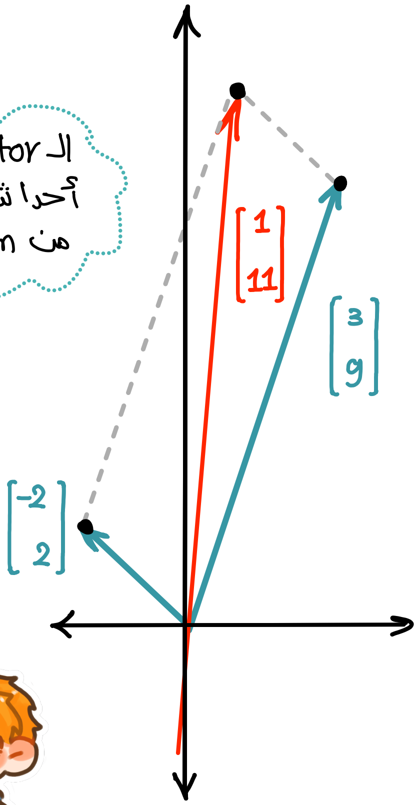
$$x - 2y = 1$$
$$3x + 2y = 11$$

$$x = 3$$
$$y = 1$$

One solution

$$x\begin{bmatrix}1\\3\end{bmatrix} + y\begin{bmatrix}-2\\2\end{bmatrix} = \begin{bmatrix}1\\11\end{bmatrix}$$

$$3\begin{bmatrix}1\\3\end{bmatrix} + 1\begin{bmatrix}-2\\2\end{bmatrix} = \begin{bmatrix}1\\11\end{bmatrix}$$



Those equations are linear

If the vector space collapses to a single line and the right-hand side of your equations (red line) is a vector not on that line.

There's no way to create it as a combination of your columns (no solution)

But if the right-hand side of your equations (red line) is a vector on that line.

All points in the vector can a be combination of the columns (infinity solution)

Understanding Solutions in Linear Equations in Data Science

Redundant features (columns) lead to no solutions because the equations represent parallel lines.

Redundant observations (rows) lead to infinite solutions because the equations represent the same line.

Solve a System of Linear Equations

Gaussian Elimination is a systematic method for solving systems of linear equations.

$$
\begin{aligned}
x - 2y &= 1 \qquad \times 3 \\
3x + 2y &= 11 \qquad -
\end{aligned}
$$

1. Present the system as a matrix equation

Pivot

entry to eliminate

$$
\begin{bmatrix} 1 & -2 \\ 3 & 2 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 1 \\ 11 \end{bmatrix}
$$

2. Perform row operations: make the coefficient matrix (A) into row echelon form.

$$
\begin{bmatrix} 1 & -2 \\ 0 & -8 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 1 \\ 8 \end{bmatrix}
$$

3. Back Substitution

Solve for y: $8y = 8 \quad => \quad y = 1$

Substitute $y = 1$ into the first equation: $x - 2(1) = 1 \quad => \quad x = 3$

Solution: $x = 3$, $y = 1$

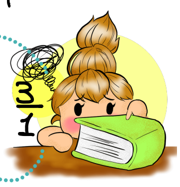
Pivots

Nonzero entries in each row are called pivots. Pivots must be nonzero to perform the division required for elimination.

Multipliers

To eliminate a variable, we use a multiplier, which is the ratio of the coefficient to be eliminated to the pivot.

$$\frac{\text{معامل إلى عاوزة أحذفه}}{\text{قيمة ال Pivot}} = \frac{3}{1}$$



Singular Systems

If a zero appears in a pivot position, the system may be singular, meaning it has no solution or infinitely many solutions.

Row Exchange انجنبها بـ

Day 2

What is vector space?

A set of vectors with operations of addition and scalar multiplication

Start from origin to certain Cartesian values as the following examples.

$$x = [5] \in \mathbb{R}^1 \qquad x = \begin{bmatrix} 1 \\ 3 \end{bmatrix} \in \mathbb{R}^2 \qquad x = \begin{bmatrix} 3 \\ 0 \\ 4 \end{bmatrix} \in \mathbb{R}^3$$

What is Linear combination?

weighted sum of vectors.

Let the vectors $v_1, v_2, v_3, \ldots, v_n$ be vectors in $\mathbb{R}^n$

$c_1, c_2, c_3, \ldots, c_n$ Scaler (commonly called the "weights").

Then the vector b, where $b = c_1 v_1 + c_2 v_2 + c_3 v_3 + \cdots + c_n v_n$ called a linear combination of $v_1, v_2, v_3, \ldots, v_n$

$$\begin{bmatrix} 2 \\ 5 \\ 3 \end{bmatrix} = c_1 \begin{bmatrix} 3 \\ 0 \\ 4 \end{bmatrix} + c_2 \begin{bmatrix} 2 \\ 2 \\ 1 \end{bmatrix} + c_3 \begin{bmatrix} 3 \\ 1 \\ 4 \end{bmatrix}$$

Vector Span

Scaler

$$c_1 \underbrace{\begin{bmatrix} 3 \\ 0 \\ 4 \end{bmatrix}}_{v_1} + c_2 \underbrace{\begin{bmatrix} 2 \\ 2 \\ 1 \end{bmatrix}}_{v_2}$$

All linear combination of the vectors. v_1, v_2 in form $c_1 v_1 + c_2 v_2$

If we have a vector $v_1 = [1 \ 2 \ 3]^T$, we can span only one dimension in the three-dimensional space because each vector would be of the form $a_1 v_1$

If we have vectors $v_1 = [1 \ 2 \ 3]^T$, $v_2 = [5 \ 9 \ 7]^T$, Span(v1, v2) is the linear combination of v_1 and v_2 with the form $a_1 v_1 + a_2 v_2$ that lies in the plane of the two-dimension plane in the three dimension space.

If we need 3D span, we need 3 vectors with 3 dimensions

Vectors MUST be independent

This means that we can't express any vector as a linear combination of others.

$$a_1 v_1 + a_2 v_2 + a_3 v_3 = 0$$

v_1, v_2, v_3 are independent

iff a_1, a_2, a_3 has values and not zero

Example:

Show whether v_1, v_2. Show weather or not the vector $v_3 \in \text{Span}(v_1, v_2)$.

$$c_1 v_1 + c_2 v_2 = v_3 \qquad v_1 = \begin{bmatrix} 2 \\ 3 \end{bmatrix} \quad v_2 = \begin{bmatrix} 1 \\ 2 \end{bmatrix} \quad v_3 = \begin{bmatrix} 19 \\ 3 \end{bmatrix}$$

If we found values for c_1 and c_2 for this equation so vector $v_3 \in \text{Span}(v_1, v_2)$.

$$c_1 \begin{bmatrix} 2 \\ 3 \end{bmatrix} + c_2 \begin{bmatrix} 1 \\ 2 \end{bmatrix} = \begin{bmatrix} 19 \\ 3 \end{bmatrix} \qquad \begin{bmatrix} 2 & 1 \\ 3 & 2 \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} = \begin{bmatrix} 19 \\ 3 \end{bmatrix} \qquad c_1 = 35 \quad c_2 = -51$$

Therefore, we have found a set of scalars c1, c2 which satisfy our condition, and therefore, $(19,3) \in \text{Span}(V)$ since $35(2,3) - 51(1,2) = (19,3)$.

Matrix Rank

Number of pivots (or non-zero rows) after Gaussian elimination (r).

Free Colums

$$A = \begin{bmatrix} 1 & 1 & 2 & 4 \\ 1 & 2 & 2 & 5 \\ 1 & 3 & 2 & 6 \end{bmatrix} \xrightarrow{\text{Gassian Elimination}} B = \begin{bmatrix} 1 & 0 & 2 & 3 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \qquad \text{Rank}(r) = 2$$

Application in image compression: the rank can be used to reduce the number of columns in a matrix representation of an image, thus reducing the amount of data needed to store it.

$$\underset{4\times5}{\begin{bmatrix} 1 & 1 & 2 & 4 & 2 \\ 2 & 1 & 3 & 5 & 4 \\ 1 & 1 & 2 & 4 & 2 \\ 0 & 1 & 1 & 2 & 0 \end{bmatrix}} \xrightarrow{\text{Decomposed to}} \underset{4\times2}{\begin{bmatrix} 1 & 1 \\ 2 & 1 \\ 1 & 1 \\ 0 & 1 \end{bmatrix}} \underset{2\times4}{\begin{bmatrix} 1 & 0 & 1 & 1 & 2 \\ 0 & 1 & 1 & 3 & 0 \end{bmatrix}}$$

20 elements $\xrightarrow{\text{Less storage}}$ 18 elements

Rank = no. of Piviots = no. of indp. Vectors

Matrix Determinant

Scaler value calculated from the a square matrix.

$v_1 = \begin{bmatrix} 2 \\ 3 \end{bmatrix} \quad v_2 = \begin{bmatrix} 1 \\ 2 \end{bmatrix} \dashrightarrow A = \begin{bmatrix} 2 & 1 \\ 3 & 2 \end{bmatrix}$

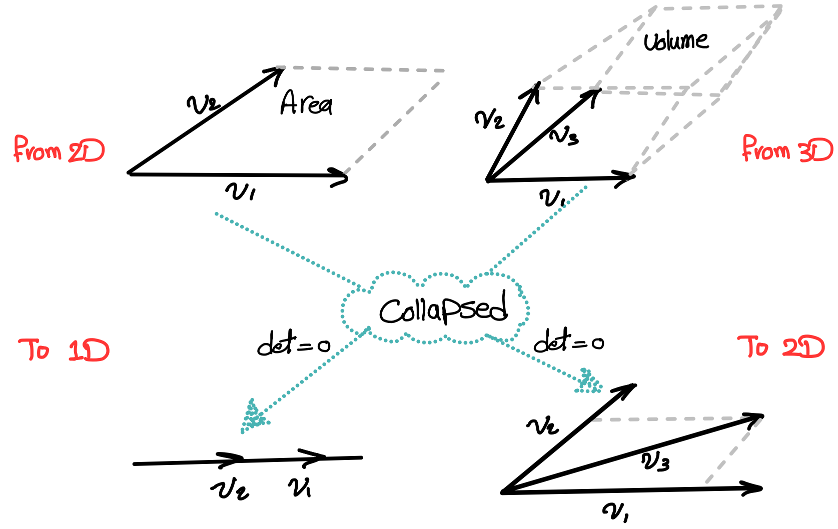
- $\det(A) \neq 0$ — v_1, v_2 are independent
- $\det(A) < 0$ — Fliping the space
- $\det(A) = 0$ — Collapsing the space

Transformation can't be restored

We can't find A^{-1}

in 2D Area $= 0$

in 3D Volume $= 0$



Vector Basis

A set of linearly independent vectors that span the full space.

Standard bases:

i = [1, 0]: This vector points one unit to the right along the x-axis.
j = [0, 1]: This vector points one unit upwards along the y-axis.

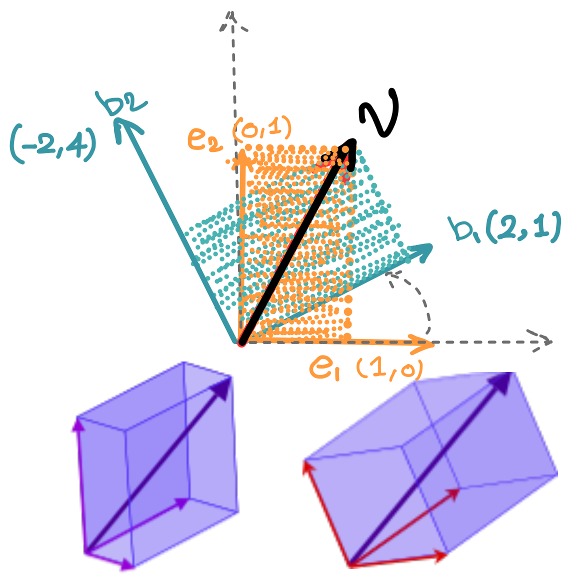
Rewriting vectors in a vector space in terms of a different set of basis elements. This allows you to move between different coordinate systems or to simplify calculations.

$\vec{v} = a_1 \hat{e}_1 + a_2 \hat{e}_2 = 3\hat{e}_1 + 4\hat{e}_2 \dashrightarrow \begin{bmatrix} 3 \\ 4 \end{bmatrix}$

express $\vec{v}$ by $\rightarrow \underbrace{\begin{bmatrix} 2 \\ 1 \end{bmatrix}, \begin{bmatrix} -2 \\ 4 \end{bmatrix}}_{\text{Transformation Matrix}}$

$B = \begin{bmatrix} 2 & -2 \\ 1 & 4 \end{bmatrix}$

New bases



$$a_1\hat{e}_1 + a_2\hat{e}_2 = b_1\hat{b}_1 + b_2\hat{b}_2$$

$$3\hat{e}_1 + 4\hat{e}_2 = ?\,\hat{b}_1 + ?\,\hat{b}_2$$

$$V = B \cdot V'$$

$$\begin{bmatrix} 3 \\ 4 \end{bmatrix} = \begin{bmatrix} 2 & -2 \\ 1 & 4 \end{bmatrix}^{-1} \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$$

$$\begin{bmatrix} 2 & -2 \\ 1 & 4 \end{bmatrix} \begin{bmatrix} 3 \\ 4 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$$

$$\begin{bmatrix} 0.4 & 0.2 \\ -0.1 & 0.2 \end{bmatrix} \begin{bmatrix} 3 \\ 4 \end{bmatrix} = \begin{bmatrix} 2 \\ 0.5 \end{bmatrix}$$

$$B^{-1} = \begin{bmatrix} 2 & -2 \\ 1 & 4 \end{bmatrix}^{-1} = \frac{1}{\det(B)} \begin{bmatrix} 4 & 2 \\ -1 & 2 \end{bmatrix}$$

$$\therefore \det(B) = 4 \times 2 - (-1 \times 2) = 10$$

$$B^{-1} = \frac{1}{10} \begin{bmatrix} 4 & 2 \\ -1 & 2 \end{bmatrix} = \begin{bmatrix} 0.4 & 0.2 \\ -0.1 & 0.2 \end{bmatrix}$$

$?\,\hat{b}_1 + ?\,\hat{b}_2 \rightarrow 2\hat{b}_1 + 0.5\hat{b}_2$

← عملت Transformation ال vectors $\hat{e}_1, \hat{e}_2$ ل vectors $\hat{b}_1, \hat{b}_2$

← كأني بجيب ال $\vec{v}$ vector في space مختلف عن طريق ال vectors span ال space دا

← Projection ← الطول في المتجه الجديد ← $\hat{b}_1$ unit vector of . $\|V\|$

$$\text{length} = \frac{\vec{v} \cdot \vec{b}_1}{\|b_1\|} \quad , \quad \text{unit vector of } \hat{b}_1 = \frac{1}{\|b_1\|}$$

$$\underbrace{b_1}_{\text{length}} = \frac{\vec{v} \cdot \vec{b}_1}{\|b_1\|} \frac{1}{\|b_1\|}$$

Projection of $\vec{v}$ on new vector $\vec{b}_1$

Changing bases formula

$$B^{-1} v B = v'$$

new bases

أحداثيات ال vector القديم

عشان أكون بكتب ال v بدلالة ال space الجديد B

أحداثيات ال vector الجديد

This is for orthogonal bases e1, e2
Called (linear transformation)

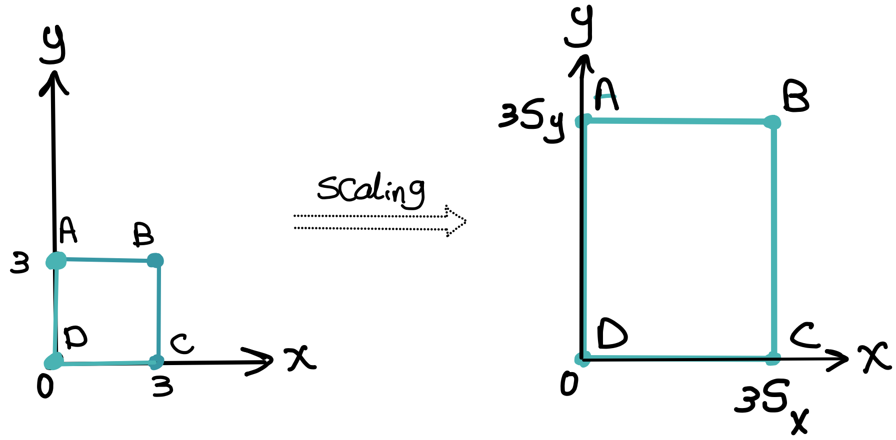
Linear Transformation

mapping space to a new space using function. Mathematically it's a matrix that maps vectors from space to another one.

- Lines stay lines No curves or bending!
- Origin stays doesn't move.
- Grid is preserved : Think of it like stretching, shrinking, rotating, or reflecting the grid without distorting the overall shape.

$$\underbrace{T}_{\text{Transformation matrix}}\ \underbrace{V}_{\text{matrix}} = \underbrace{V_t}_{\text{matrix after Transformation}}$$

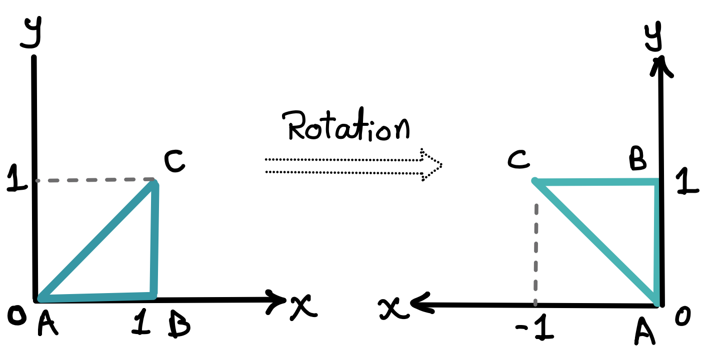
Scaling transformation matrix



$$\begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix}$$

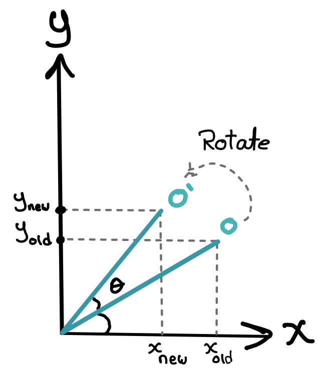
Scaling matrix

Rotation transformation matrix

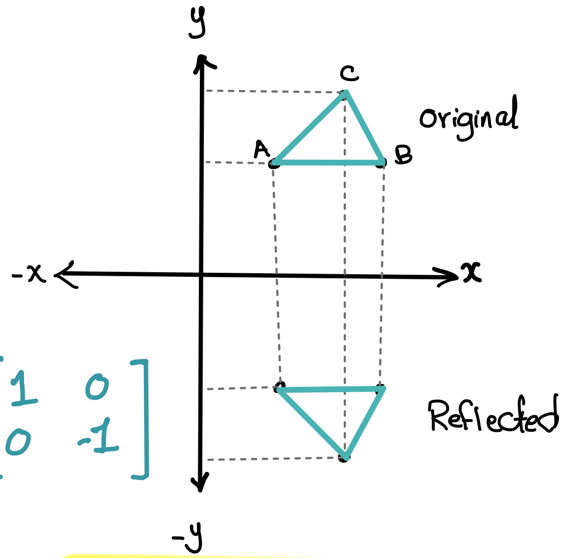


$$\begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

Rotation matrix

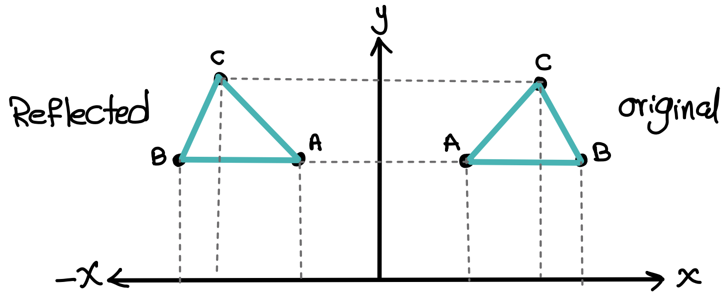


Reflecting transformation matrix



$$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$

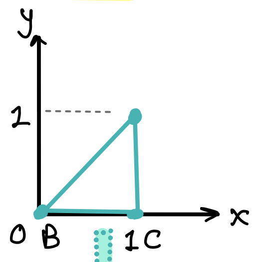
Reflection along x-axis

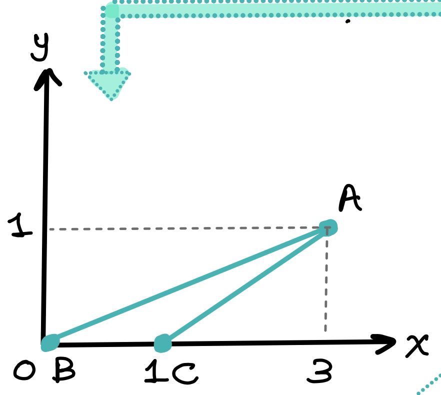


$$\begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}$$

Reflection along y-axis

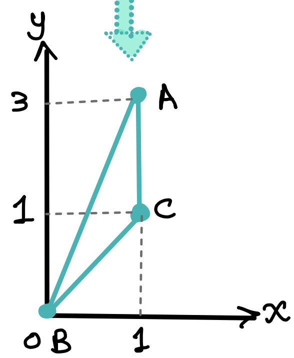
Rotation transformation matrix





$$\begin{bmatrix} 1 & S_x \\ 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 \\ S_y & 1 \end{bmatrix}$$



Shearing in x-axis

Shearing in y-axis

🌻 Importance in Data Science and AI:

In data science, features are often represented as vectors. The choice of basis for these vectors directly influences how your algorithms understand the data.

1. Different bases highlight different aspects of the data.

2. Simplify and reduce the dimensionality of data: PCA and SVD use changing basis to find new, more informative features. This process often involves rotating and scaling the original data to highlight the most important directions of variation by remove less important features.

3. Text Features Extraction: Techniques like Word2Vec and GloVe learn embeddings by changing the basis to capture semantic relationships between words.

4. Image Features Extraction: Changing the basis of an image matrix can be used to extract features like edges, corners, or textures, which are important for tasks like object recognition.

5. Image Processing: Linear transformations can rotate, scale, or shear images while preserving the overall structure.

Non-Orthonormal transformation

Non-orthonormal spaces are common in data science and AI. These spaces are where the basis vectors are not perpendicular to each other.

So, we first convert matrix v to an orthogonal space by changing bases as explained, then multiply the new matrix by the transformation matrix.

Computationally expensive

$$B^{-1}TBV = V'$$

new transformation matrix

تتغير ال bases كأني بغير ال Grid لكن ال origin وال Vector نفسة ثابت

ال Transformation بغير ال Vector نفسة

use B^T instead of B^{-1} in orthonormal

Orthogonalization

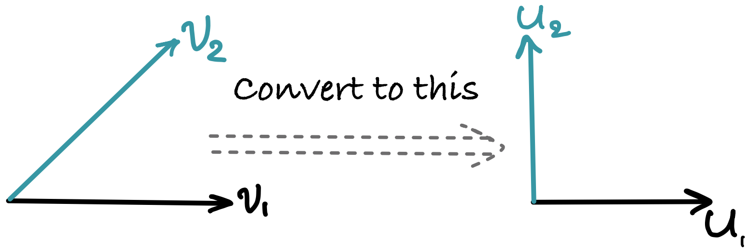
Gram- Schmidt process

$$u_1 = v_1$$

$$u_2 = v_2 - \text{Proj}_{u_1}(v_2)$$

$$u_3 = v_3 - [\text{Proj}_{u_1}(v_3) + \text{Proj}_{u_2}(v_3)]$$

$$u_k = v_k - \sum_{j=1}^{k-1} \text{Proj}_{u_j}(v_k)$$



$$\text{Proj}_{u_j}(v_k) = \frac{v_k \cdot u_j}{u_j\, u_j}\, u_j$$

$$e_k = \frac{u_k}{\|u_k\|}$$

For orthonormal matrix

$$E = [e_1 \quad e_2 \quad \dots \quad e_k] \qquad E = Q \qquad Q^T Q = I \qquad Q^{-1} = Q^T$$

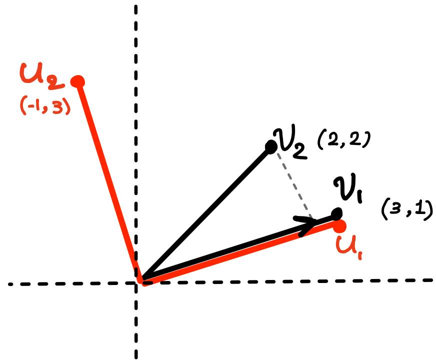
Steps: -

1- Transform r from B space to E space using $\rightarrow r_E = E^{-1} r_B \rightarrow r_E = E^T r_B$

2- Apply transformation $\rightarrow r'_E = E^T T_E r_B$

3- Back to B space $\rightarrow r'_B = E^T T_E r_B E$

Example:



$$v_1 = \begin{bmatrix} 3 \\ 1 \end{bmatrix} \qquad v_2 = \begin{bmatrix} 2 \\ 2 \end{bmatrix}$$

$$u_1 = v_1 = \begin{bmatrix} 3 \\ 1 \end{bmatrix}$$

$$u_2 = v_2 - \text{proj}_{u_1}(v_2) = \begin{bmatrix} 3 \\ 1 \end{bmatrix} - \text{proj}_{\left[\begin{smallmatrix} 3 \\ 1 \end{smallmatrix}\right]} \begin{bmatrix} 2 \\ 2 \end{bmatrix}$$

$$= \begin{bmatrix} 3 \\ 1 \end{bmatrix} - \frac{\begin{bmatrix} 3 \\ 1 \end{bmatrix}\begin{bmatrix} 2 \\ 2 \end{bmatrix}}{\begin{bmatrix} 3 \\ 1 \end{bmatrix}\begin{bmatrix} 3 \\ 1 \end{bmatrix}} \begin{bmatrix} 2 \\ 2 \end{bmatrix} = \begin{bmatrix} 3 \\ 1 \end{bmatrix} - \frac{8}{10}\begin{bmatrix} 2 \\ 2 \end{bmatrix} = \begin{bmatrix} -2/5 \\ 6/5 \end{bmatrix}$$

$$e_1 = \frac{u_1}{\|u_1\|} = \frac{1}{\sqrt{10}}\begin{bmatrix} 3 \\ 1 \end{bmatrix} \qquad e_2 = \frac{u_2}{\|u_2\|} = \frac{1}{\sqrt{\frac{40}{25}}}\begin{bmatrix} -2/5 \\ 6/5 \end{bmatrix} = \frac{1}{\sqrt{10}}\begin{bmatrix} -1 \\ 3 \end{bmatrix}$$

$$E = [e_1 \quad e_2] = \frac{1}{\sqrt{10}}\begin{bmatrix} 3 & -1 \\ 1 & 3 \end{bmatrix}$$

Dot product for $\hat{u}_1, \hat{u}_2 = 0$

QR decomposition.

QR decomposition separates a matrix into a product of an orthogonal matrix and an upper triangular matrix.:

$$A = QU \qquad U = Q^{-1}A$$

R: An upper triangular matrix.

Q: An orthogonal matrix. Its columns are perpendicular and unit length.

The orthogonal matrix 'Q' is usually obtained through orthogonalization. Once you have Q, you can find U by

Solving Linear Equations

QR decomposition is helpful for solving systems of linear equations in the form $AX = b$

Rewrite the equation as $QUX = b$

Since 'Q' is orthogonal, its inverse is simply its transpose $Q^{-1} = Q^T$

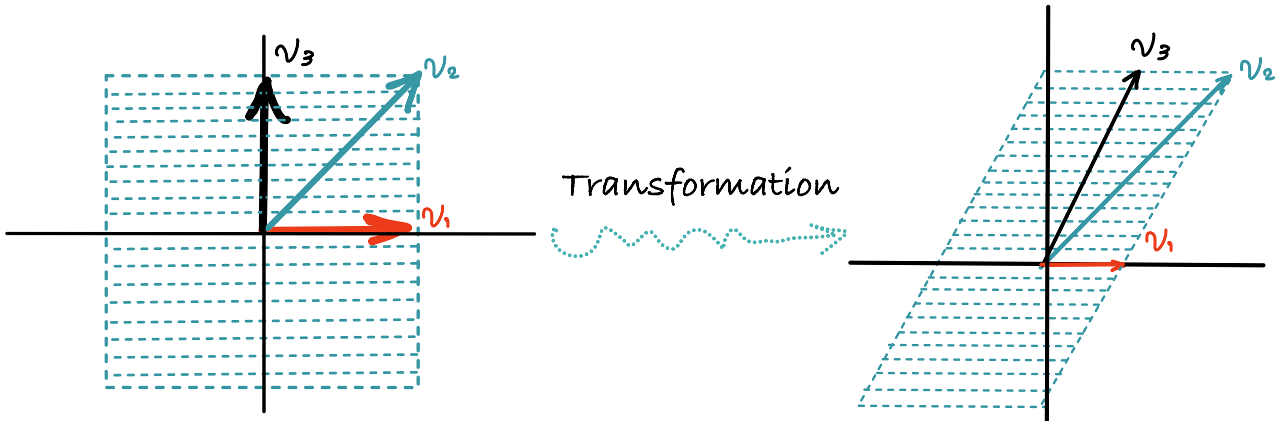
This gives you $UX = Q^T b$ which can be solved directly using back substitution (a simpler method for upper triangular matrices).

Day 3

Eigenvectors and Eigenvalues

Eigenvectors represent directions that are "unchanged" (except for scaling) by a linear transformation.

Eigenvalues measure how much the eigenvectors are scaled by the transformation.



v_1 v_2 Eigenvectors ⇢ Their direction doesn't change after transfernation

Calculating Eigenvectors and Eigenvalues

$$Ax = \lambda x$$

$$Ax - \lambda x = 0$$

$$(A - \lambda I)\,x = 0$$

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

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

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

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

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

$$-4 - \lambda + 4\lambda + \lambda^2 - 6 = 0$$

$$\lambda^2 + 3\lambda - 10 = 0$$

$$\lambda = 2 \qquad \lambda = -5$$

Eigenvalues

$\lambda = 2$

$$\begin{bmatrix} -1 & 2 \\ 3 & -2 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = 0$$

$$-v_1 - 2v_2 = 0$$

$$3v_1 + 2v_2 = 0$$

$$v_1 = 2 \qquad v_2 = 1$$

$\lambda = -5$

$$\begin{bmatrix} -6 & 2 \\ 3 & 1 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = 0$$

$$-6v_1 + v_2 = 0$$

$$3v_1 + 2v_2 = 0$$

$$v_1 = 1 \qquad v_2 = 3$$

Eigenvectors

$$\begin{bmatrix} 2 \\ 1 \end{bmatrix} \qquad \begin{bmatrix} 1 \\ 3 \end{bmatrix}$$

Eigen-Decomposition (Diagonalization)

Convert matrix (t) to mother more simple form includes diagonal matrix

$$T = C D C^{-1}$$

- **T**: The original transformation matrix.
- **C**: A matrix whose columns are the eigenvectors of 'T' (the eigen basis).
- **D**: A diagonal matrix containing the eigenvalues of 'T' along its diagonal.
- **C^{-1}**: The inverse of the eigenvector matrix 'C'.

Example:

$$T = \begin{bmatrix} 1 & 1 \\ 0 & 2 \end{bmatrix}$$

Find T^2

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

Most Computationally efficient method

$$T = C D C^{-1} \qquad T^2 = C D^2 C^{-1}$$

Solve to find eignvectors and eignvalues we found that

$$\lambda = 1 \qquad \lambda = 2 \qquad \begin{bmatrix} 1 \\ 1 \end{bmatrix} \qquad \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

$$C = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \longrightarrow C^{-1} = \begin{bmatrix} 1 & -1 \\ 0 & 1 \end{bmatrix}$$

$$D = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix} \longrightarrow D^2 = \begin{bmatrix} 1 & 0 \\ 0 & 4 \end{bmatrix}$$

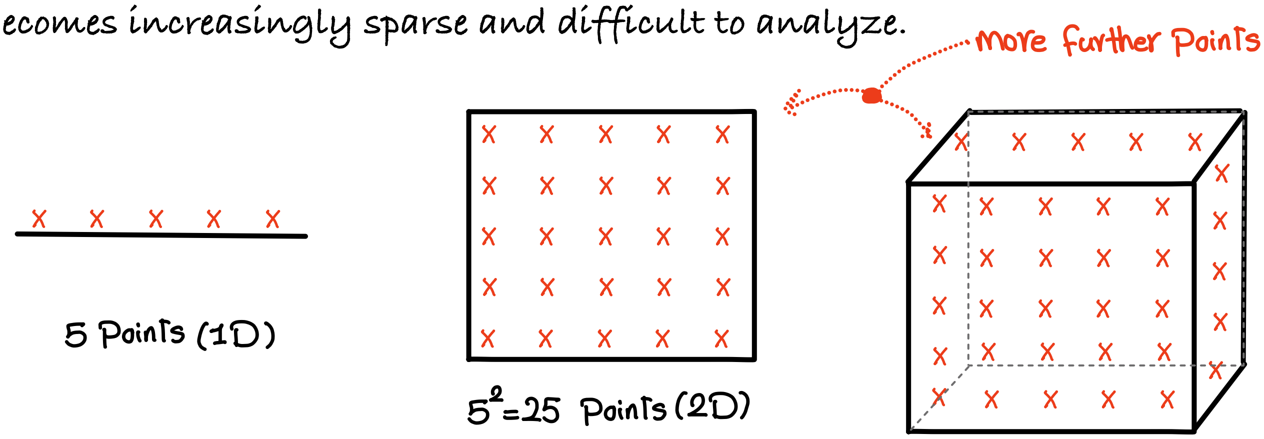
$$T^2 = C D^2 C^{-1}$$

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

Same Result

Curse of Dimensionality

When the number of features (dimensions) in a dataset increases, the data becomes increasingly sparse and difficult to analyze.



5 Points (1D)

$5^2 = 25$ Points (2D)

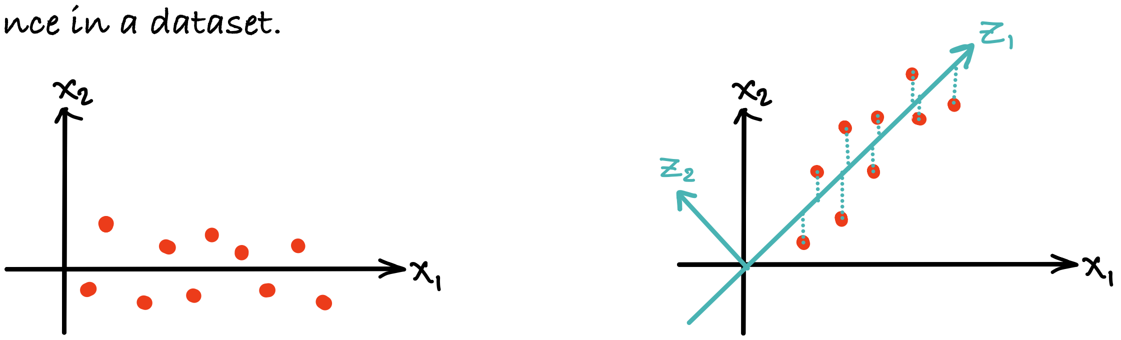
$5^3 = 125$ Points (3D)

Principal Component Analysis (PCA)

A crucial dimensionality reduction technique used in data science and AI.

أعبر عنها بـ dim أقل

PCA aims to find a new set of axes (called principal components) that capture the most variance in a dataset.



الـ x_2 Feature قيمة في Range ديق وبالـ var صغير... فهيكون

تأثيرها على الناتج ضعيف جداً ← يمكن حذفها

هنا في Var كبير في x_1, x_2

طب ما أغير الـ Bases لـ z_1, z_2

واحد منهم في اتجاه أكبر Var والتاني

عمودي عليه ← z_1, z_2 eignvectors وبالتالي أقدر أحذف z_2 لما غيرت الـ Bases معناه

أني عملت Transformation وبالتالي هاضرب الـ Data Matrix في الـ Bases الجديدة

وديت الـ Data لـ Space مختلف من خلاله أقدر أعبر عنها بـ dim أقل

PCA Steps:

1. Standardize the data to avoid bias in the covariance.

$$x_{new} = \frac{x - \mu}{\sigma}$$

2. Calculate the covariance matrix that measures the joint variability between pairs of features.

$$Var(x) = \frac{\sum (x_i - x)^2}{n-1}$$

0 in std data

$$Cov(x,y) = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{n-1} = \frac{X^T X}{n-1}$$

Dot Product Normalized data

$$\begin{array}{c c} & \begin{array}{ccc} x & y & z \end{array} \\ \begin{array}{c} x \\ y \\ z \end{array} & \begin{bmatrix} var(x) & cov(x,y) & cov(x,z) \\ cov(x,y) & var(y) & cov(y,z) \\ cov(x,z) & cov(y,z) & var(z) \end{bmatrix} \end{array}$$

3. Calculate eigenvalues and eigenvectors

$\lambda =$ 2.51579324, 1.0652885, 0.39388704, 0.02503121

E =

e1	e2	e3	e4
0.161960	-0.917059	-0.307071	0.196162
-0.524048	0.206922	-0.817319	0.120610
-0.585896	-0.320539	0.188250	-0.720099
-0.596547	-0.115935	0.449733	0.654547

4. Select principal components by choose the k eigenvectors with the largest eigenvalues. These eigenvectors will represent the 'k' most important directions of variance in the data.

large variance ⇒ more info

e1	e2	e3	e4
0.161960	-0.917059	-0.307071	0.196162
-0.524048	0.206922	-0.817319	0.120610
-0.585896	-0.320539	0.188250	-0.720099
-0.596547	-0.115935	0.449733	0.654547

5. Transform the original data: Project the original data onto the selected principal components (eigenvectors), creating a new, lower-dimensional representation.

f1	f2	f3	f4
-1.118034	-0.707107	0.000000	0.291386
0.372678	1.414214	1.936492	1.748315
-1.118034	0.707107	-0.645497	-0.194257
0.372678	0.000000	-0.645497	-1.165543
1.490712	-1.414214	-0.645497	-0.679900

@

e1	e2
0.161960	-0.917059
-0.524048	0.206922
-0.585896	-0.320539
-0.596547	-0.115935

=

PC1	PC2
-0.015656	0.845205
2.858292	-0.872549
0.057557	1.401047
-1.133854	0.000267
-1.766338	-1.373970

Singular Value Decomposition (SVD)

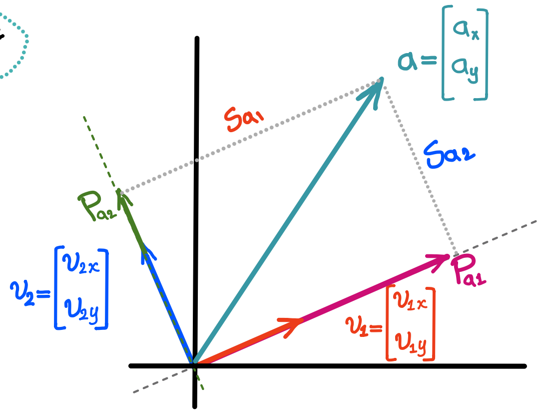
Problem:

When dealing with large datasets with many features, calculating the covariance matrix (needed for eigen-decomposition) becomes computationally expensive.

Any vector can be expressed in terms of:

أقدر أمثل الـVector عن طريق إحداثيات الـ Orthogonal Basis

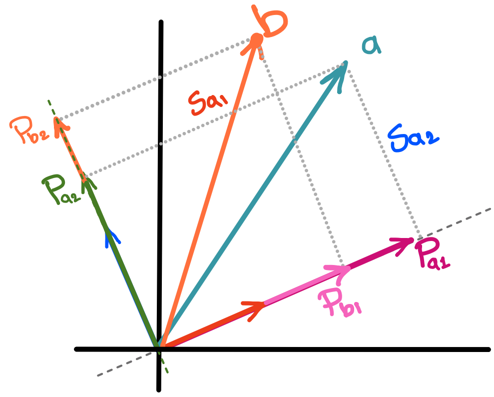
$$\vec{a} = S_{a_1} v_1 + S_{a_2} v_2$$



أقدر أمثل الـVector عن طريق Projection on the Orthogonal Basis

$$P_{a_1} = \begin{bmatrix} a_x & a_y \end{bmatrix} \begin{bmatrix} v_{1x} \\ v_{1y} \end{bmatrix}$$ Projection of vector a on. v_1

$$P_{a_2} = \begin{bmatrix} a_x & a_y \end{bmatrix} \begin{bmatrix} v_{2x} \\ v_{2y} \end{bmatrix}$$ Projection of vector a on. v_2



$$a^T \cdot V = \begin{bmatrix} a_x & a_y \end{bmatrix} \begin{bmatrix} v_{1x} & v_{2x} \\ v_{1y} & v_{2y} \end{bmatrix} = \begin{bmatrix} S_{a_1} \\ S_{a_2} \end{bmatrix}$$

$$A \cdot V = \begin{bmatrix} a_x & a_y \\ b_x & b_y \end{bmatrix} \begin{bmatrix} v_{1x} & v_{2x} \\ v_{1y} & v_{2y} \end{bmatrix} = \begin{bmatrix} S_{a_1} & S_{b_1} \\ S_{a_2} & S_{b_2} \end{bmatrix}$$

$$A \cdot V = \underbrace{\begin{bmatrix} a_x & a_y & \cdots \\ b_x & b_y & \cdots \\ \vdots & \vdots & \end{bmatrix}}_{n \times d} \underbrace{\begin{bmatrix} v_{1x} & v_{2x} & \cdots \\ v_{1y} & v_{2y} & \cdots \\ \vdots & \vdots & \end{bmatrix}}_{d \times d} = \underbrace{\begin{bmatrix} S_{a_1} & S_{b_1} & \cdots \\ S_{a_2} & S_{b_2} & \cdots \\ \vdots & \vdots & \end{bmatrix}}_{n \times d}$$

n: no. of Points — d: no. of Dim.

$$\boxed{A \cdot V = S}$$

matrix of points (A) — Decomposion matrix (V) — Length of projections (basis) (S)

$$A = S V^{-1} \dashrightarrow A = S V^{T}$$

$$S = U \Sigma$$

Unit vector

Vector Length

$$U = \begin{bmatrix} \frac{S_{a_1}}{\sigma_1} & \frac{S_{b_1}}{\sigma_1} \\ \frac{S_{a_2}}{\sigma_2} & \frac{S_{b_2}}{\sigma_2} \end{bmatrix} \qquad \Sigma = \begin{bmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \end{bmatrix}$$

Normalized input data

$$A = U \Sigma V^{T}$$

Relation between PCA and SVD

$$(A^T B)^T = B^T A$$

$$U^T = U^{-1}$$

if orthogonal

$$\text{Covar} = \frac{A^T A}{n-1} = \frac{(U \Sigma V^T)^T U \Sigma V^T}{n-1} = \frac{(V \Sigma^T \cancel{U^T}) \cancel{U} \Sigma V^T}{n-1}$$

$$= \frac{V \Sigma^T \Sigma V^T}{n-1} = V \frac{\Sigma^T \Sigma}{n-1} V^T$$

$$\begin{bmatrix} \sigma_1^2 & 0 \\ 0 & \sigma_2^2 \end{bmatrix} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} = \Lambda$$

lamda = eignvalues

عارفين أن هي ال Covariance
Std- input data ال Dot Product

Vectors الأساسية بتاعت الداتة basis
eignvectors

Singularvalues Σ ⇢ Eignvalues

Singularvectors V ⇢ Eignvector