IMPORTANT FIRST THIS NOTES

PCA (Principal Component Analysis) — 2D → 1D COMPLETE NOTES

WHAT IS PCA? (1 LINE)

PCA reduces dimensions by keeping maximum variance and removing redundancy.

STEP 0 : RAW DATA (2D)

Given dataset:

Point	x	y
A	2	4
B	4	6
C	6	8

Each point has 2 features → 2D data

STEP 1 : MEAN-CENTER THE DATA (MANDATORY)

Why mean-center?

- PCA studies variation around the center, not absolute position

- Covariance and eigenvectors assume centered data

Compute mean:

x_mean = 4

y_mean = 6

Subtract mean from each point:

Point	x - x_mean	y - y_mean
A	-2	-2
B	0	0
C	2	2

Now data center = (0, 0)

This step moves the data cloud, it does NOT rotate it.

STEP 2 : BUILD COVARIANCE MATRIX

Question:

How does the data spread around the center?

Variance of x:

Var(x) = [(-2)^2 + 0^2 + 2^2] / 2

Var(x) = (4 + 0 + 4) / 2

Var(x) = 4

Variance of y:

Var(y) = 4

Covariance:

Cov(x, y) = [(-2)(-2) + 0 + (2)(2)] / 2

Cov(x, y) = (4 + 0 + 4) / 2

Cov(x, y) = 4

Covariance Matrix:

Sigma =

[4   4

  4   4]

Meaning:

- Shape of data
- Direction of spread
- Relationship between features

STEP 3 : FIND EIGENVALUES AND EIGENVECTORS

Solve:

Sigma * v = lambda * v

Eigenvalues:

lambda1 = 8

lambda2 = 0

Eigenvectors:

v1 = (1, 1)

v2 = (1, -1)

Normalize eigenvectors:

PC1 = (1 / sqrt(2)) * (1, 1)

PC2 = (1 / sqrt(2)) * (1, -1)

These are the new rotated axes.

STEP 4 : SELECT PRINCIPAL COMPONENT

PCA rule:

Keep the eigenvector with the largest eigenvalue.

Component	Eigenvalue	Keep
PC1	8	YES
PC2	0	NO

PC1 captures maximum variance.

STEP 5 : PROJECT DATA (DOT PRODUCT)

This is the actual 2D → 1D step.

Projection formula:

z = X_centered * PC1

Project each point:

Point A (-2, -2):

z_A = (-2, -2) dot (1 / sqrt(2))(1, 1)

z_A = -4 / sqrt(2)

Point B (0, 0):

z_B = 0

Point C (2, 2):

z_C = 4 / sqrt(2)

STEP 6 : FINAL 1D DATA

Original (x, y) 1D value

(2, 4)	-4 / sqrt(2)
(4, 6)	0
(6, 8)	4 / sqrt(2)

Each 2D point → ONE number

Dimension reduced

Maximum variance preserved

PIPELINE — CONNECT EVERYTHING

Raw Data

↓

Mean-Center

↓

Covariance Matrix

↓

Eigenvalues and Eigenvectors

↓

Select PC1

↓

Dot Product (Projection)

↓

1D Value

FINAL INTUITION

- Eigenvector = new axis
- Eigenvalue = importance (variance)
- Dot product = shadow / projection
- PC1 kept, PC2 dropped
- PCA = Rotate → Project → Reduce

ONE-LINE EXAM ANSWER

PCA reduces 2D data to 1D by projecting mean-centered data onto the eigenvector of the covariance matrix corresponding to the largest eigenvalue.

MEMORY LOCK

Variance → Covariance

Direction → Eigenvector

Importance → Eigenvalue

Shadow → Dot product

Compression → PCA

🔹 CASE 1: VARIANCE (WHY IT LOOKS DIFFERENT)

Variance of x means:

```
Var(x) = Cov(x, x)
```

Now plug into the SAME formula:

```
Var(x) = (1 / (n - 1)) * sum( (xi - x_mean) * (xi - x_mean) )
```

That multiplication becomes a square:

```
Var(x) = (1 / (n - 1)) * sum( (xi - x_mean)^2 )
```

👉 **That's why variance has a square**
👉 **No new formula — same formula**

🔹 CASE 2: COVARIANCE (x, y)

Use the SAME formula, but with two different variables:

```
Cov(x, y) = (1 / (n - 1)) * sum( (xi - x_mean) * (yi - y_mean) )
```

Why the image writes A * B

Different authors use different symbols for the dot product:

A · B → Dot product (most common in mathematics)

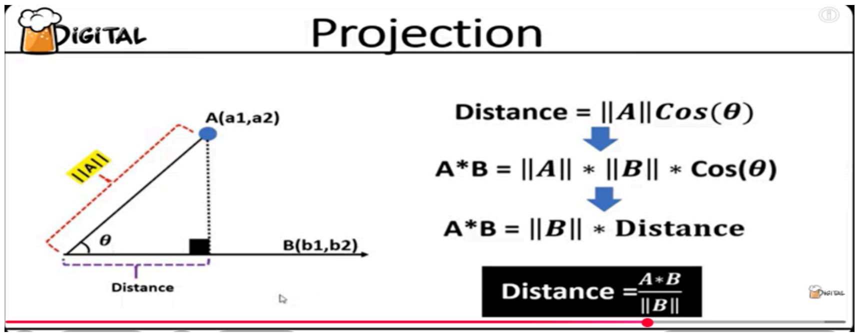
A * B → Dot product (used in some slides / videos)

$A^T B$ → Dot product (linear algebra / machine learning)

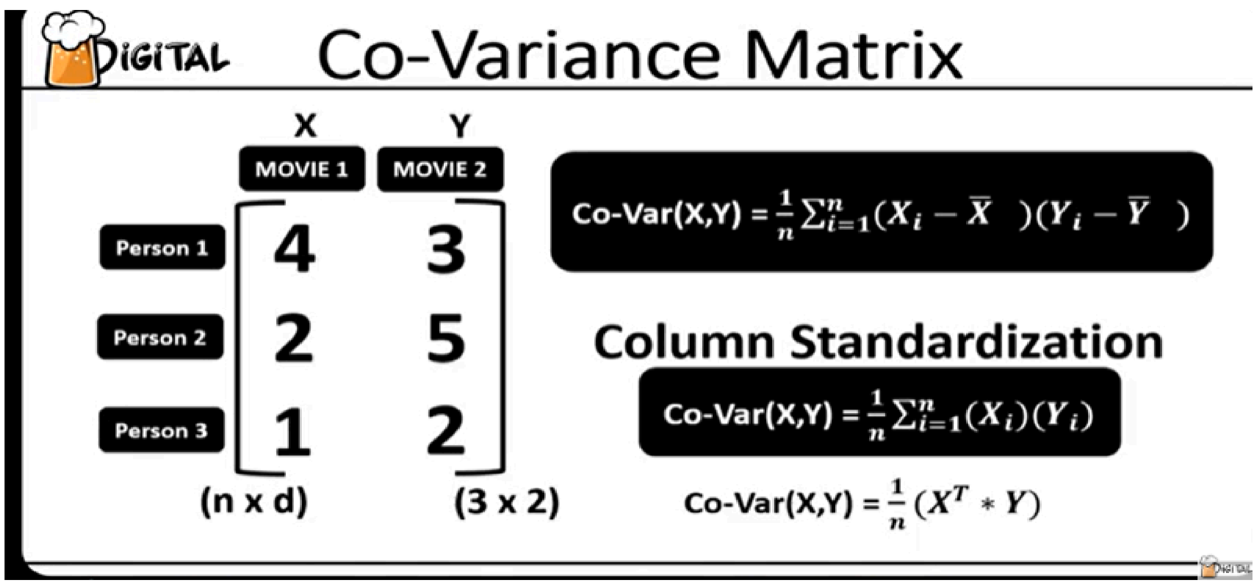
So in the image:

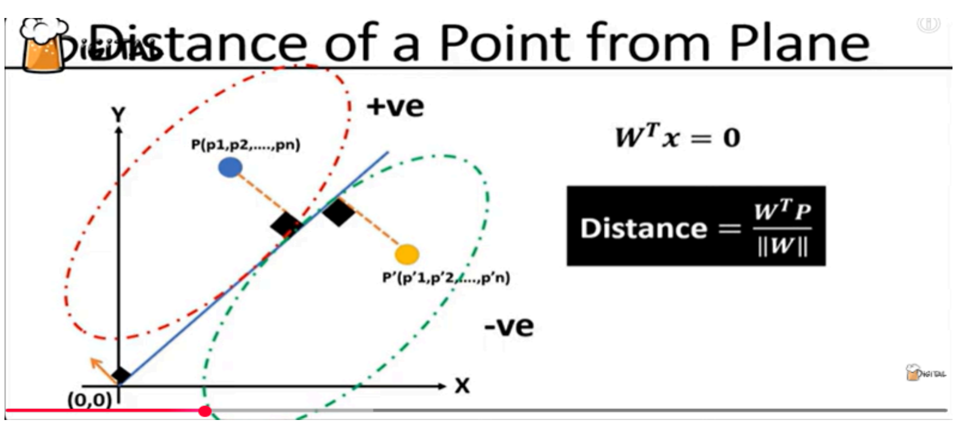
A * B = ||A|| ||B|| cos(θ)

This is exactly the definition of the dot product.



.





COVARIANCE MATRIX — COMPLETE EXPLANATION (ALL IN ONE)

1️⃣ Start from the DATA

Assume we have n data points with two features:

$(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$

Example dataset:

i	x	y
1	2	4
2	4	6
3	6	8

2️⃣ Compute the MEANS (very important)

$\bar{x} = (2 + 4 + 6) / 3 = 4$

$\bar{y} = (4 + 6 + 8) / 3 = 6$

All covariance-matrix values come from deviations from the mean.

3️⃣ Variance of x → Var(x)

Formula:

$Var(x) = (1 / (n - 1)) \Sigma (x_i - \bar{x})^2$

Calculation:

$(2 - 4)^2 + (4 - 4)^2 + (6 - 4)^2$

$= 4 + 0 + 4$

$= 8$

$\mathrm{Var}(x) = 8 / 2 = 4$

Meaning:

Measures spread of x-values only.

4️⃣ Variance of y → Var(y)

Formula:

$\mathrm{Var}(y) = (1 / (n - 1)) \Sigma (y_i - \bar{y})^2$

Calculation:

$(4 - 6)^2 + (6 - 6)^2 + (8 - 6)^2$

$= 4 + 0 + 4$

$= 8$

$\mathrm{Var}(y) = 8 / 2 = 4$

Meaning:

Measures spread of y-values only.

5️⃣ Covariance → Cov(x, y)

Formula:

$\mathrm{Cov}(x, y) = (1 / (n - 1)) \Sigma (x_i - \bar{x})(y_i - \bar{y})$

Calculation:

$(2 - 4)(4 - 6) + (4 - 4)(6 - 6) + (6 - 4)(8 - 6)$

$= (-2)(-2) + 0 + (2)(2)$

$= 4 + 0 + 4$

$= 8$

$\mathrm{Cov}(x, y) = 8 / 2 = 4$

Meaning:

Measures how x and y change together.

6️⃣ Why $\mathrm{Cov}(x, y) = \mathrm{Cov}(y, x)$

$(x_i - \bar{x})(y_i - \bar{y}) = (y_i - \bar{y})(x_i - \bar{x})$

Hence:

$\mathrm{Cov}(x, y) = \mathrm{Cov}(y, x)$

7️⃣ Final Covariance Matrix

$$\Sigma = \begin{bmatrix} \mathrm{Var}(x) & \mathrm{Cov}(x, y) \\ \mathrm{Cov}(y, x) & \mathrm{Var}(y) \end{bmatrix}$$

$$\Sigma = \begin{bmatrix} 4 & 4 \\ 4 & 4 \end{bmatrix}$$

✔ Every value comes directly from raw data
✔ Only mean and deviations are used

8️⃣ Interpretation of Covariance Values

$\mathrm{Cov} > 0$ → Features increase together

Cov < 0 → One increases, other decreases

Cov ≈ 0 → No linear relationship

9️⃣ Big-Picture Summary

Matrix Entry	Meaning
Var(x)	Spread of x values
Var(y)	Spread of y values
Cov(x, y)	Relationship between x and y
Cov(y, x)	Same relationship (symmetric)

🔟 Why Covariance Matrix is Needed

- Captures spread in multiple dimensions
- Describes shape and orientation of data
- Basis for PCA
- Eigenvectors → principal directions
- Eigenvalues → variance along those directions

🧠 Memory Rules

Diagonal → variance (single feature)

Off-diagonal → covariance (feature interaction)

Variance = spread

```
Covariance = tilt
```

PRINCIPAL COMPONENT ANALYSIS (PCA)

ONE COMPLETE, CONNECTED EXPLANATION

(WHY → WHAT → HOW → MATH → NUMERICAL → USES)

1️⃣ WHY PCA EXISTS (REAL-LIFE MOTIVATION)

Suppose we collect house data with two features:

- House size

- Number of rooms

When plotted:

- Bigger houses usually have more rooms

- Data points form a slanted line, not a full 2D cloud

So we ask:

Do we really need two numbers to describe a house,

or can one number capture “how big the house is”?

📌 This question is the origin of PCA.

PCA exists because:

- Many features are correlated

- High-dimensional data is redundant

- Models become slow and overfit

Goal of PCA:

Replace many correlated features with fewer informative directions.

2️⃣ DATA AS VECTORS (HOW ML SEES DATA)

In machine learning:

- Each data point is a vector

- Each feature is a direction (axis)

Example:

$x = [\text{house size, number of rooms}]^T$

A vector represents:

- Direction → where the data point lies

- Length → magnitude

Thus, a dataset is a collection of vectors in space.

3️⃣ VARIANCE = INFORMATION (CORE IDEA)

PCA is based on one principle:

Directions with more variance contain more information.

Variance (1D):

$Var(x) = (1/n)\ \Sigma\ (x_i - \bar{x})^2$

Interpretation:

- High variance → important structure

- Low variance → noise or redundancy

📌 PCA searches for directions of maximum variance.

4️⃣ ROTATING THE AXES (KEY INTUITION)

Imagine rotating the coordinate axes:

- Data points remain fixed

- Only the viewing direction changes

At one special rotation:

- Data appears most stretched

- Variance is maximized

📌 PCA finds this optimal rotation.

The resulting direction is a principal component.

5️⃣ WHY THE COVARIANCE MATRIX IS NEEDED

To understand spread in multiple dimensions, we need:

- Variance of each feature

- Relationship between features

This is captured by the covariance matrix.

For two features:

Σ = [[Var(x), Cov(x,y)],

[Cov(y,x), Var(y)]]

Where:

$Cov(x,y) = (1/(n-1)) \Sigma (x_i - \bar{x})(y_i - \bar{y})$

Meaning:

- Cov > 0 → features increase together

- Cov < 0 → one increases, other decreases

- Cov ≈ 0 → no linear relation

📌 Covariance matrix describes the shape and orientation of data.

6️⃣ EIGENVALUES & EIGENVECTORS (WHY THEY APPEAR)

For a square matrix A:

$A v = \lambda v$

Where:

- A = covariance matrix

- v = eigenvector

- λ = eigenvalue

In PCA:

- Eigenvector → direction in which data spreads naturally

- Eigenvalue → amount of variance along that direction

📌 Covariance matrix acts as a transformation.

📌 Eigenvectors are directions unchanged by that transformation.

📌 These directions are the natural axes of the data.

7️⃣ WHY EIGENVECTORS ARE PRINCIPAL COMPONENTS

From the covariance matrix:

- Each eigenvector gives a direction

- Its eigenvalue gives importance (variance)

Ordering:

- Largest eigenvalue → PC1

- Second largest → PC2

- And so on

📌 PCA keeps top eigenvectors and discards the rest.

8️⃣ PCA ALGORITHM (LOGIC + MATH)

Step 1: Standardize the data

$z = (x - \mu) / \sigma$

Reason: PCA is variance-based; scale matters.

Step 2: Compute covariance matrix

$\Sigma = (1/(n-1))\, X^T X$

Step 3: Compute eigenvalues & eigenvectors

Solve:

$|\Sigma - \lambda I| = 0$

Step 4: Sort eigenvalues

$\lambda_1 \geq \lambda_2 \geq \ldots$

Step 5: Choose top k components

$(\Sigma_{i=1}^{k} \lambda_i) / (\Sigma_{j=1}^{d} \lambda_j) \geq 95\%$

Step 6: Project data

$Z = XW$

Where:

- W = matrix of selected eigenvectors

- Z = reduced dataset

📌 Projection = shadow of data onto principal directions.

---

9 COMPLETE NUMERICAL EXAMPLE (2D → 1D)

Dataset:

X = [[2, 1],
 [3, 2],
 [4, 3]]

Mean:

$\bar{x} = (3, 2)$

Centered data:

X′ = [[−1, −1],
 [0, 0],
 [1, 1]]

Covariance matrix:

Σ = [[1, 1],
 [1, 1]]

Eigenvalues:

$\lambda_1 = 2$, $\lambda_2 = 0$

Eigenvectors:

$v_1 = (1/\sqrt{2})(1, 1)$

$v_2 = (1/\sqrt{2})(1, -1)$

Choose PC1 since it captures 100% variance.

Projection:

$Z = X'v_1$

Result: 1D representation.

🔟 GEOMETRIC INTERPRETATION

PCA rotates the coordinate system so that new axes align with directions of maximum variance. These axes are orthogonal eigenvectors of the covariance matrix. Projecting data onto the top axes reduces dimensionality while preserving maximum information.

1️⃣1️⃣ WHEN & WHERE PCA IS USED

WHEN:

- High-dimensional data

- Correlated features

- Noise reduction

- Visualization

WHERE:

- Image compression

- Face recognition

- Genomics

- NLP embeddings

- Recommendation systems

1️⃣2️⃣ PCA VS FEATURE SELECTION

PCA:

- Creates new features

- Linear combinations

- Less interpretable

Feature Selection:

- Chooses existing features

- Subset of original

- More interpretabll

1️⃣3️⃣ ONE-LINE INTERNAL WORKING

PCA diagonalizes the covariance matrix to find orthogonal directions of maximum variance and projects data onto them.

1️⃣4️⃣ FINAL MEMORY BLOCK

Eigenvector → direction

Eigenvalue → importance

Covariance matrix → data shape

Projection → compression

PCA → rotate → keep long axes → drop short ones