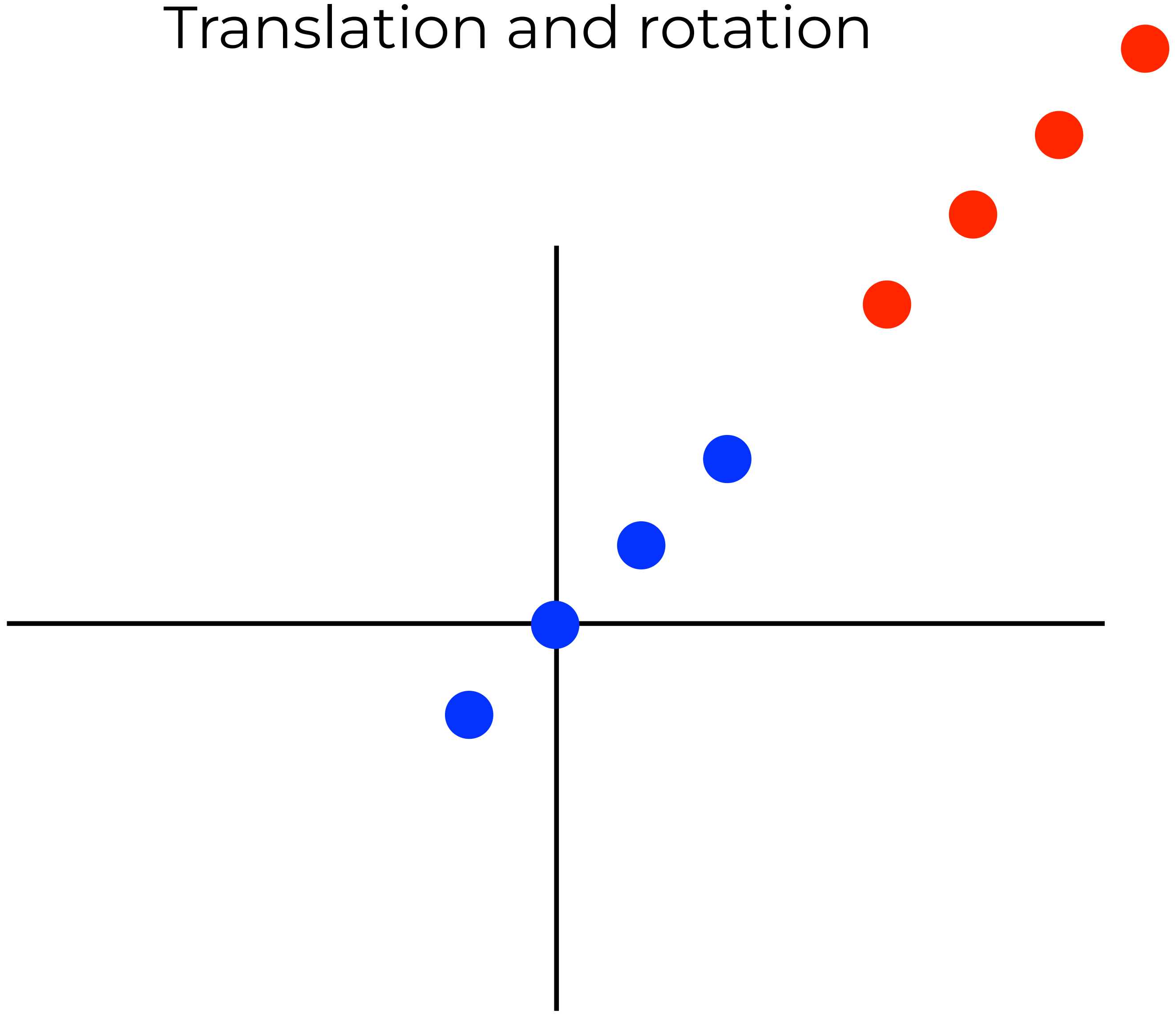
# Principal Component Analysis (PCA)

Itthi Chatnuntawech

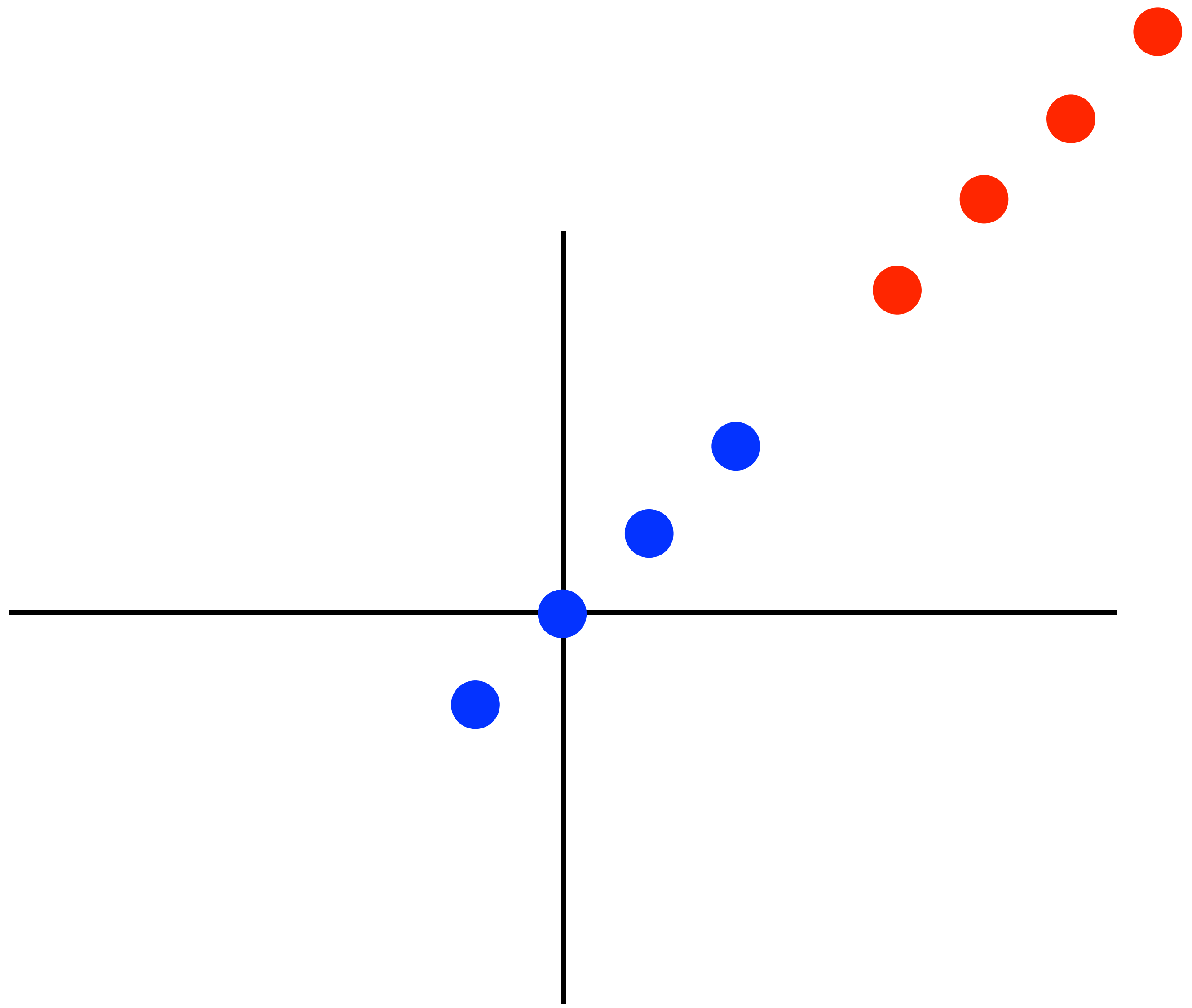
# Simple Transformations

Translation and rotation

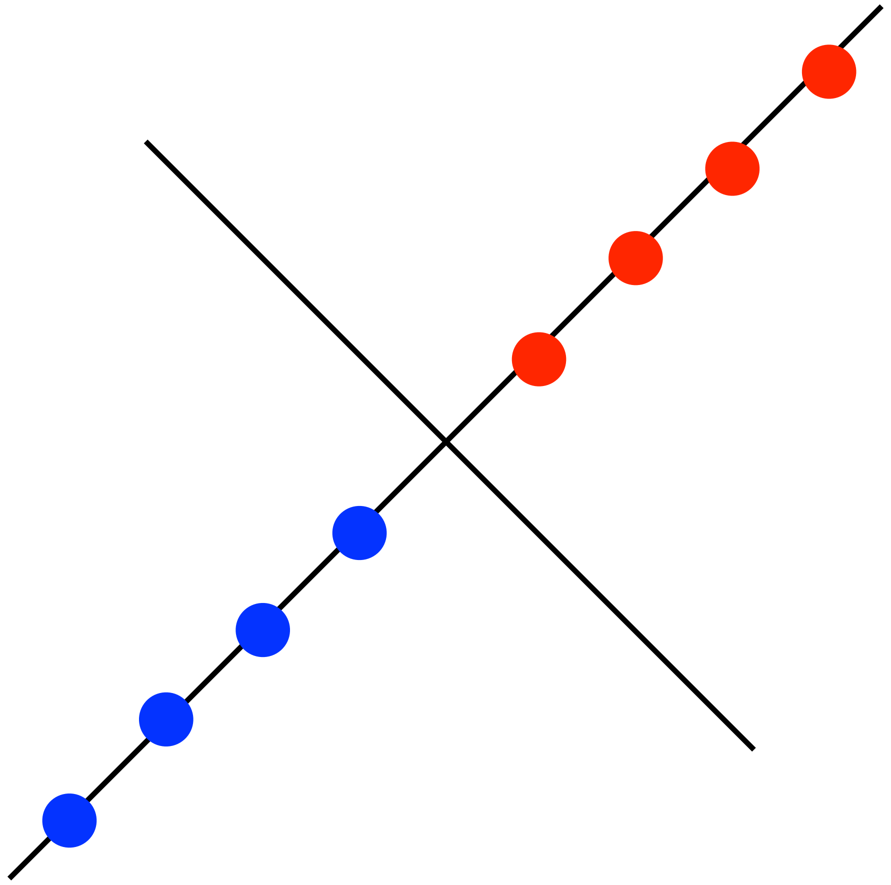


# Simple Transformations

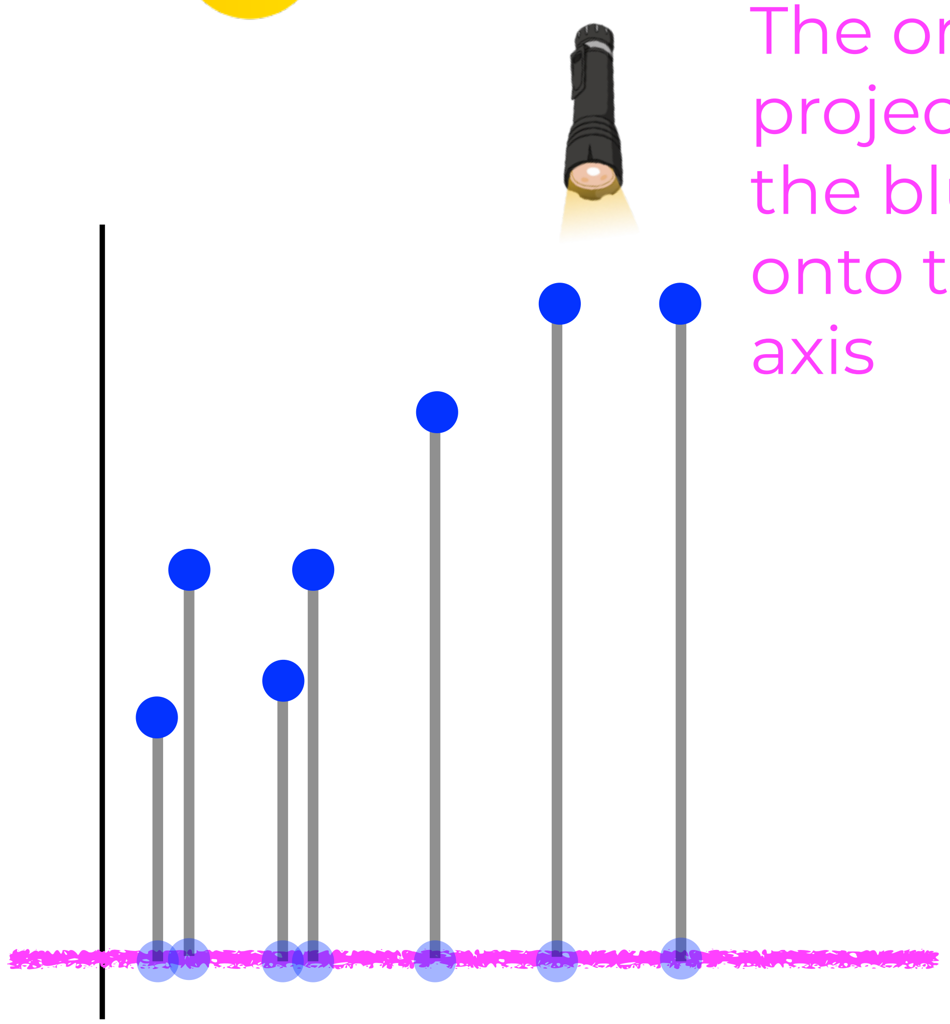
Original space



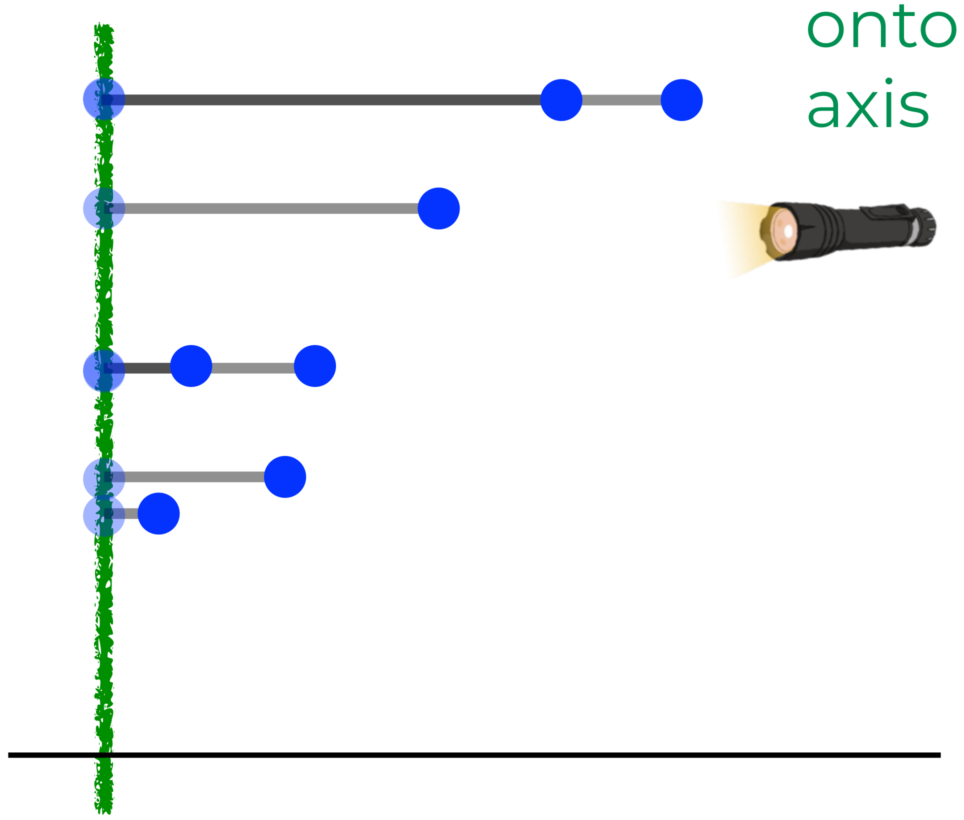
Transformed space



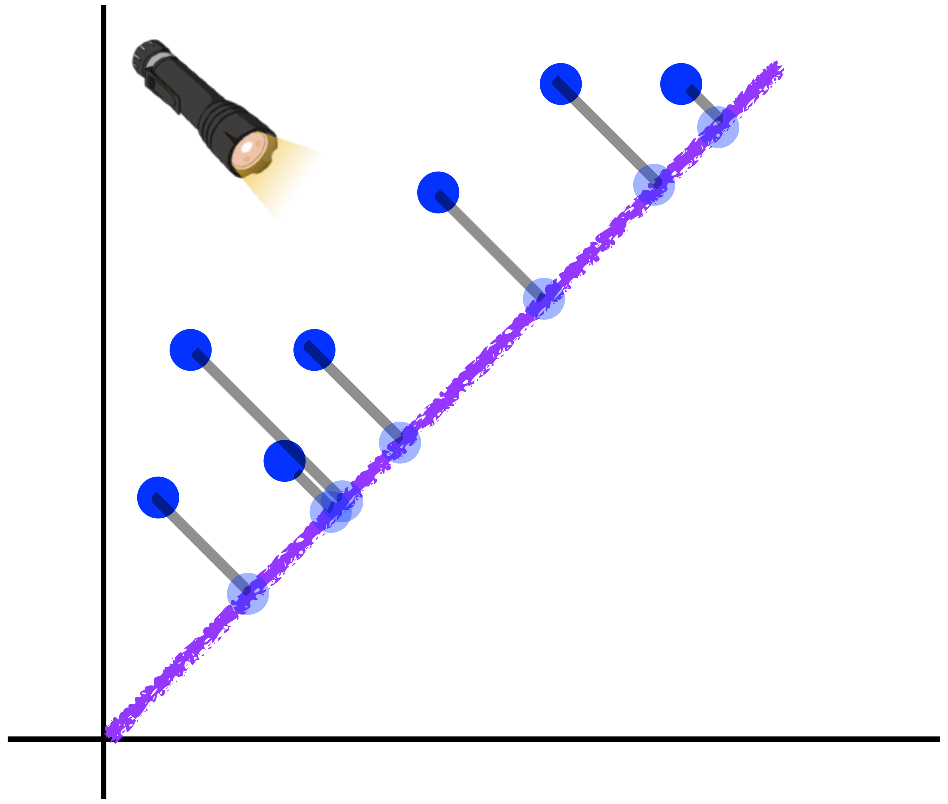
# Projection



The orthogonal projection of the blue points onto the pink axis

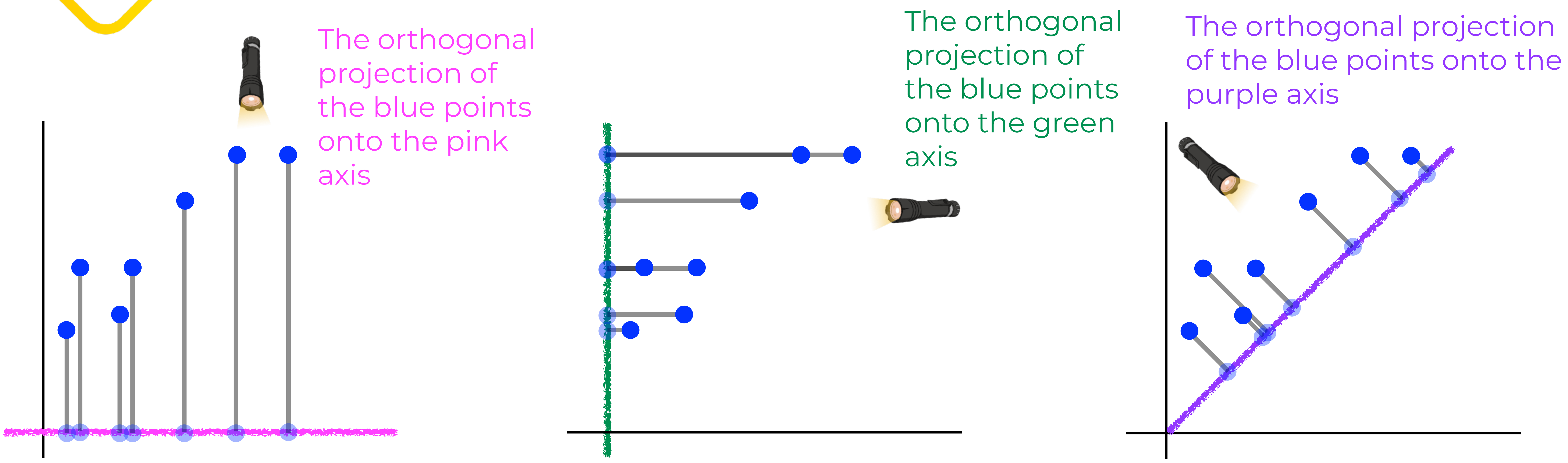


The orthogonal projection of the blue points onto the green axis

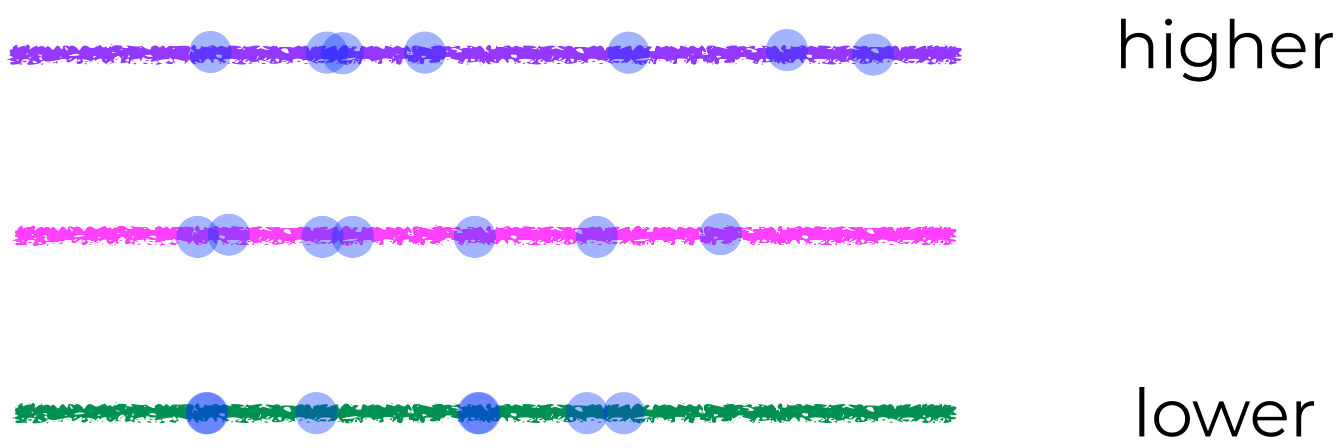


The orthogonal projection of the blue points onto the purple axis

# Projection

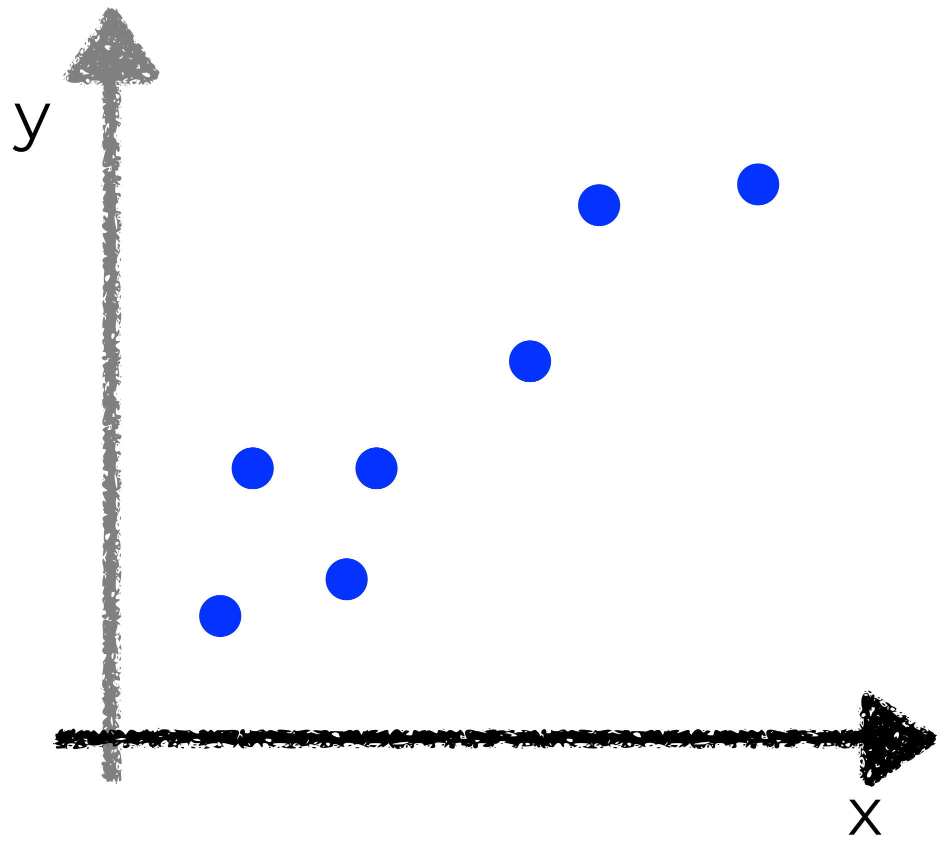


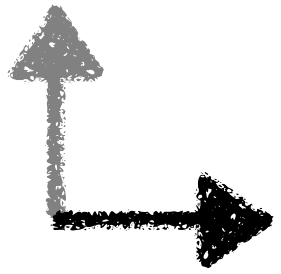

How spread out the projected points are



# Principal Component Analysis (PCA)

**Original space**

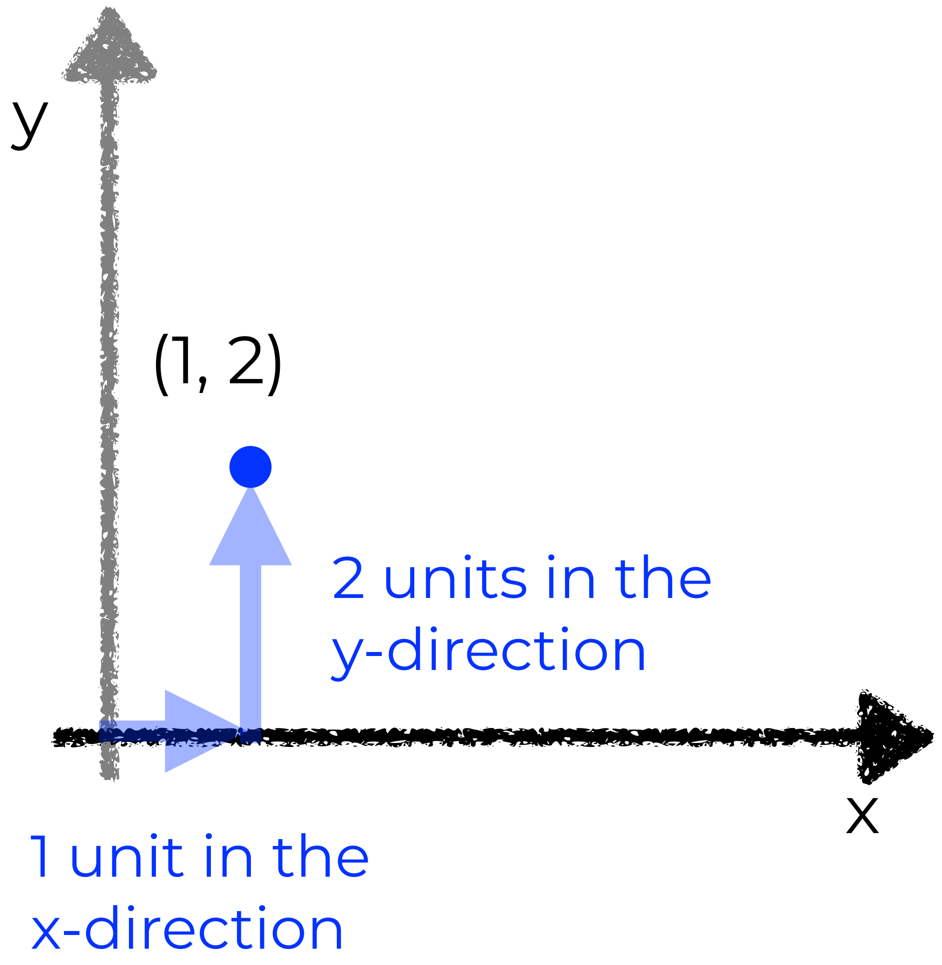


(x, y)

How much you move in the x-direction

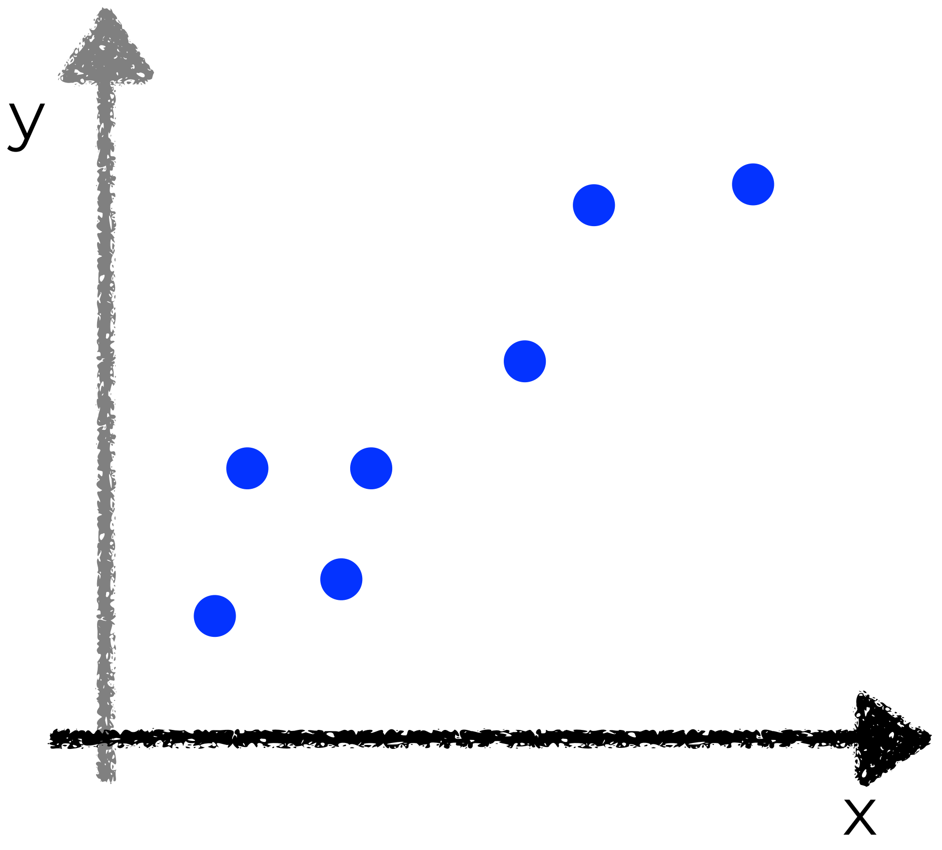
How much you move in the y-direction

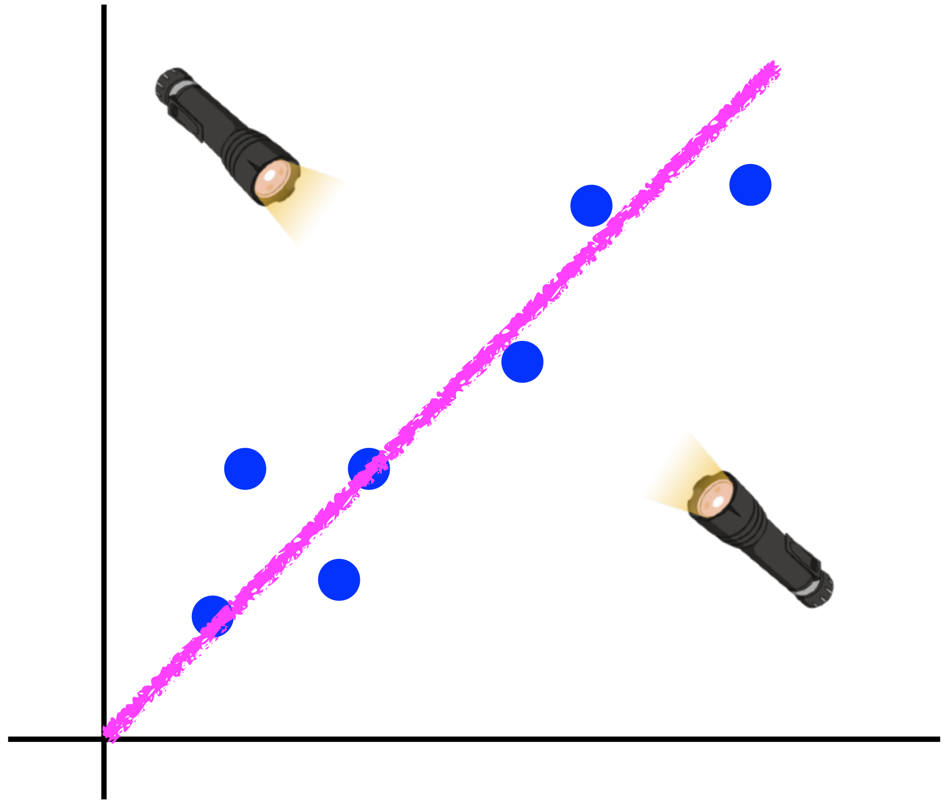


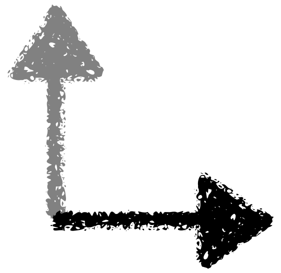
Brain Code Camp

# Principal Component Analysis (PCA)

**Original space**





(x, y)

$PC_1$

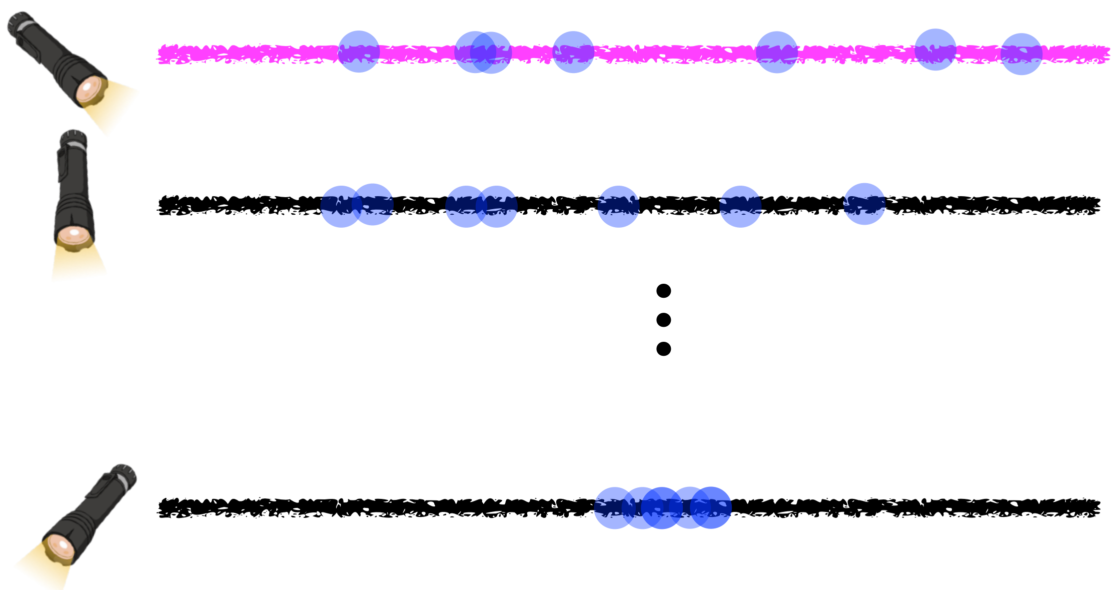
How much you move in the x-direction

How much you move in the y-direction

For d-dimensional data, PCA can find up to d directions (principal components or PCs)

1. Find the direction that maximizes the variance of the projected points. This direction can be described by a vector called the first principal component ($PC_1$)

How spread out the projected points are



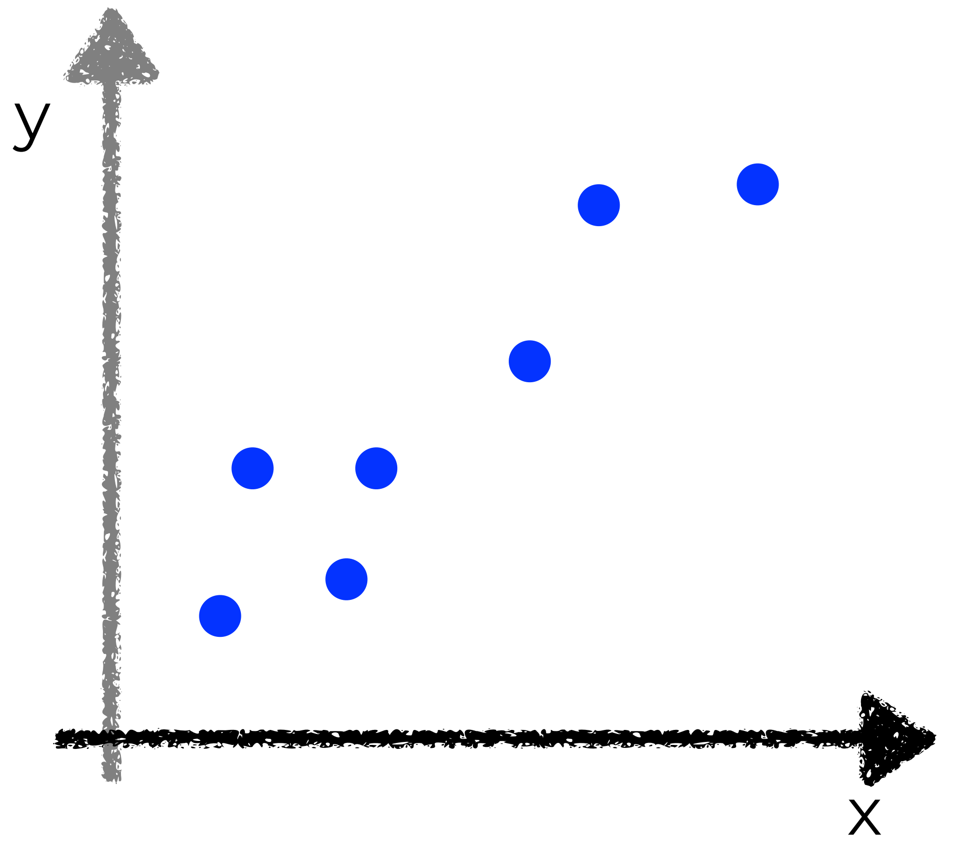
highest

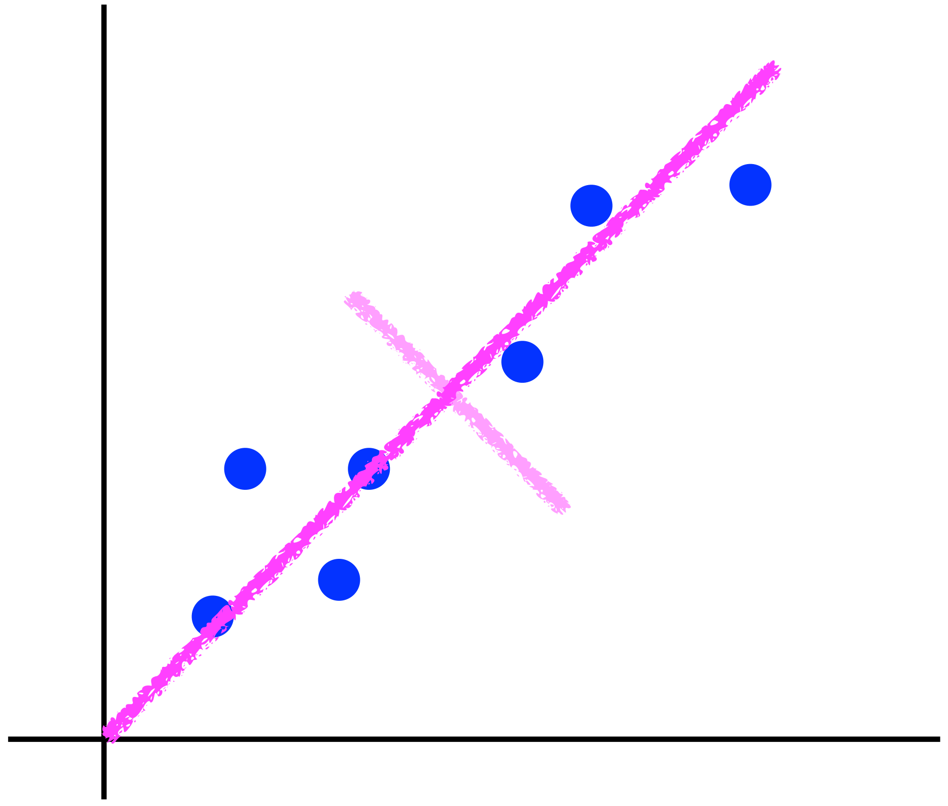
lowest

Brain Code Camp

# Principal Component Analysis (PCA)

**Original space**



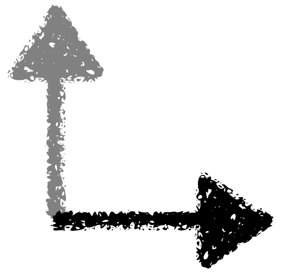



For d-dimensional data, PCA can find up to d directions (principal components or PCs)

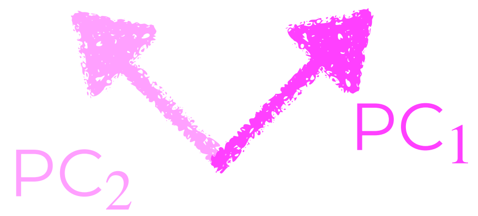
1. Find the direction that maximizes the variance of the projected points. This direction can be described by a vector called the first principal component ($PC_1$)

2. Find the next PC that
   - is orthogonal to the PCs already considered
   - maximizes the variance along the new direction

3. Repeat step 2 until you have the desired number of PCs



(x, y)




($PC_1$ score, $PC_2$ score)

How much you move in the x-direction

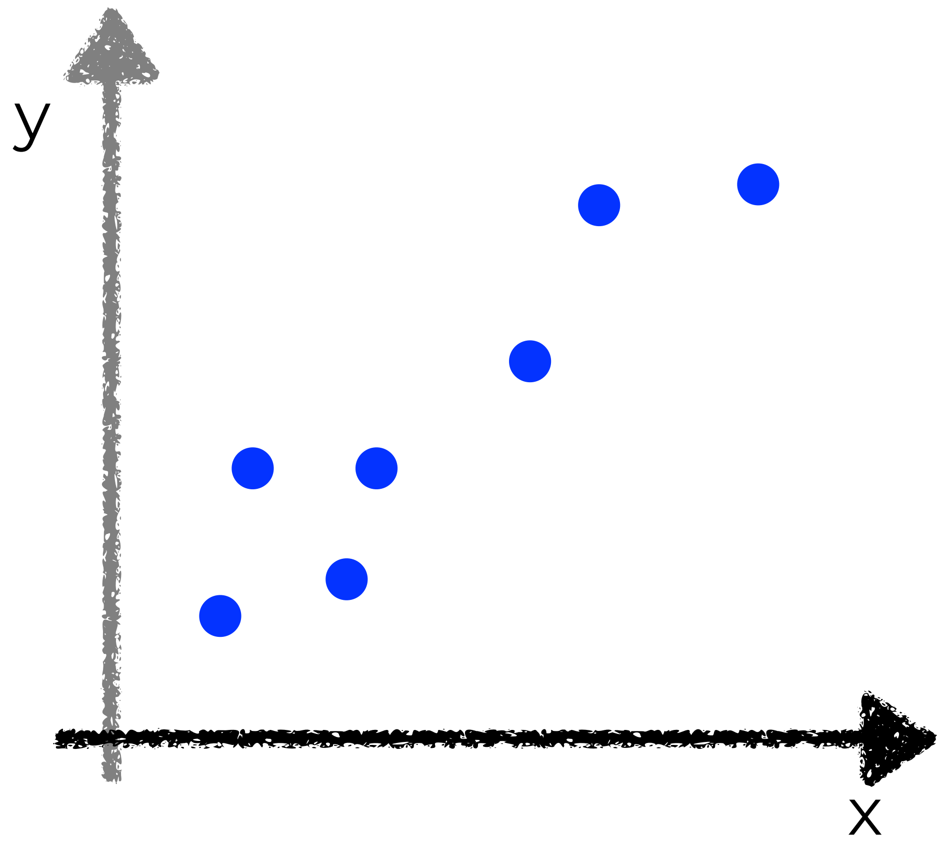
How much you move in the y-direction

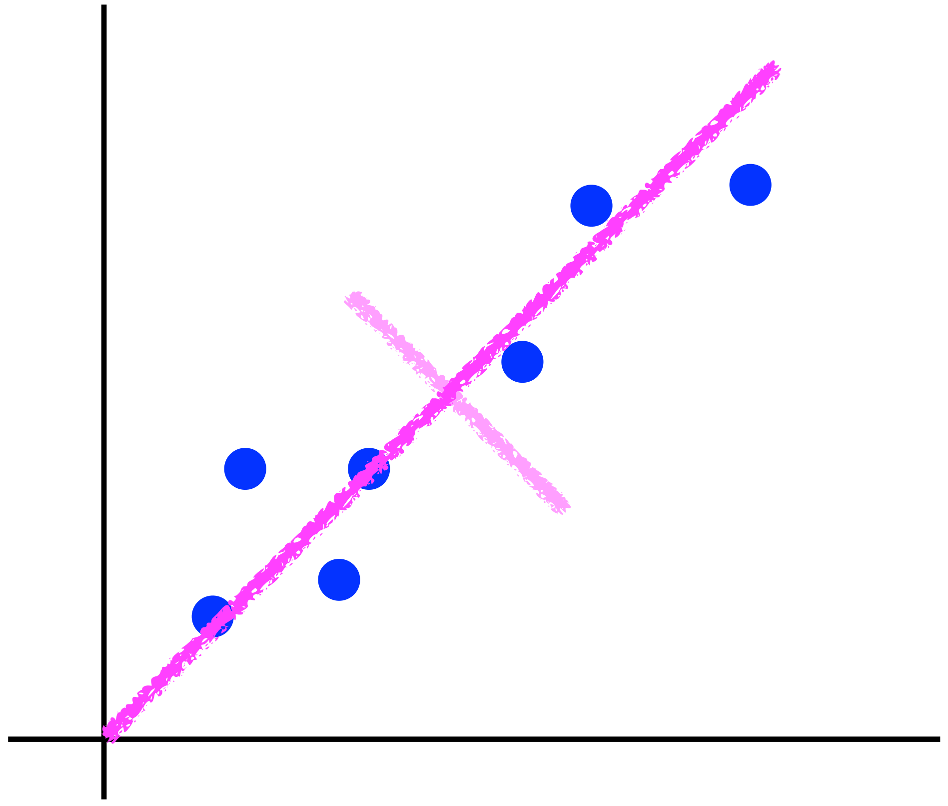
How much you move in the $PC_1$ direction

How much you move in the $PC_2$ direction

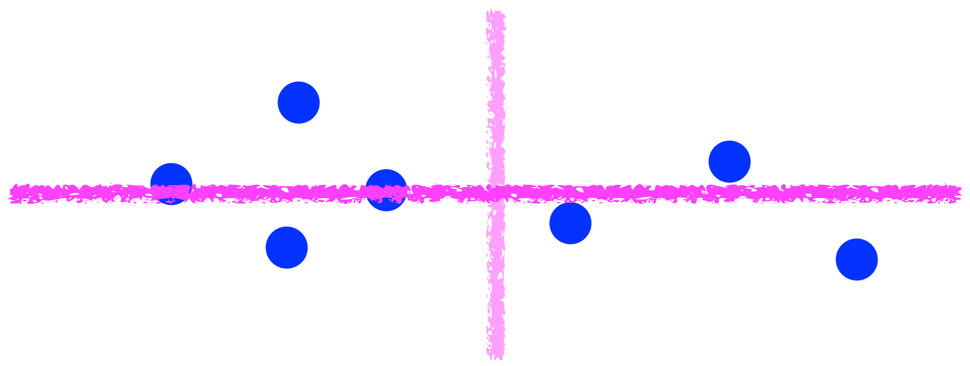
# Principal Component Analysis (PCA)

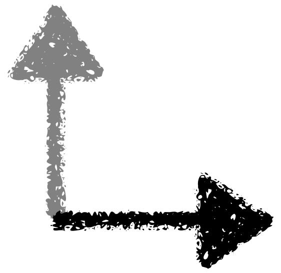
**Original space**



**PCA**

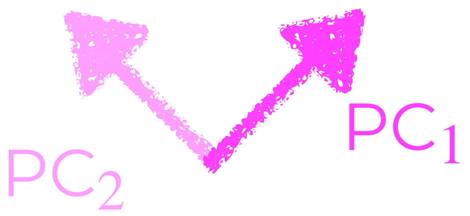




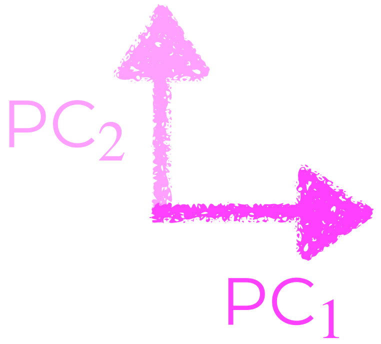
(x, y)

How much you move in the x-direction

How much you move in the y-direction




($PC_1$ score, $PC_2$ score)



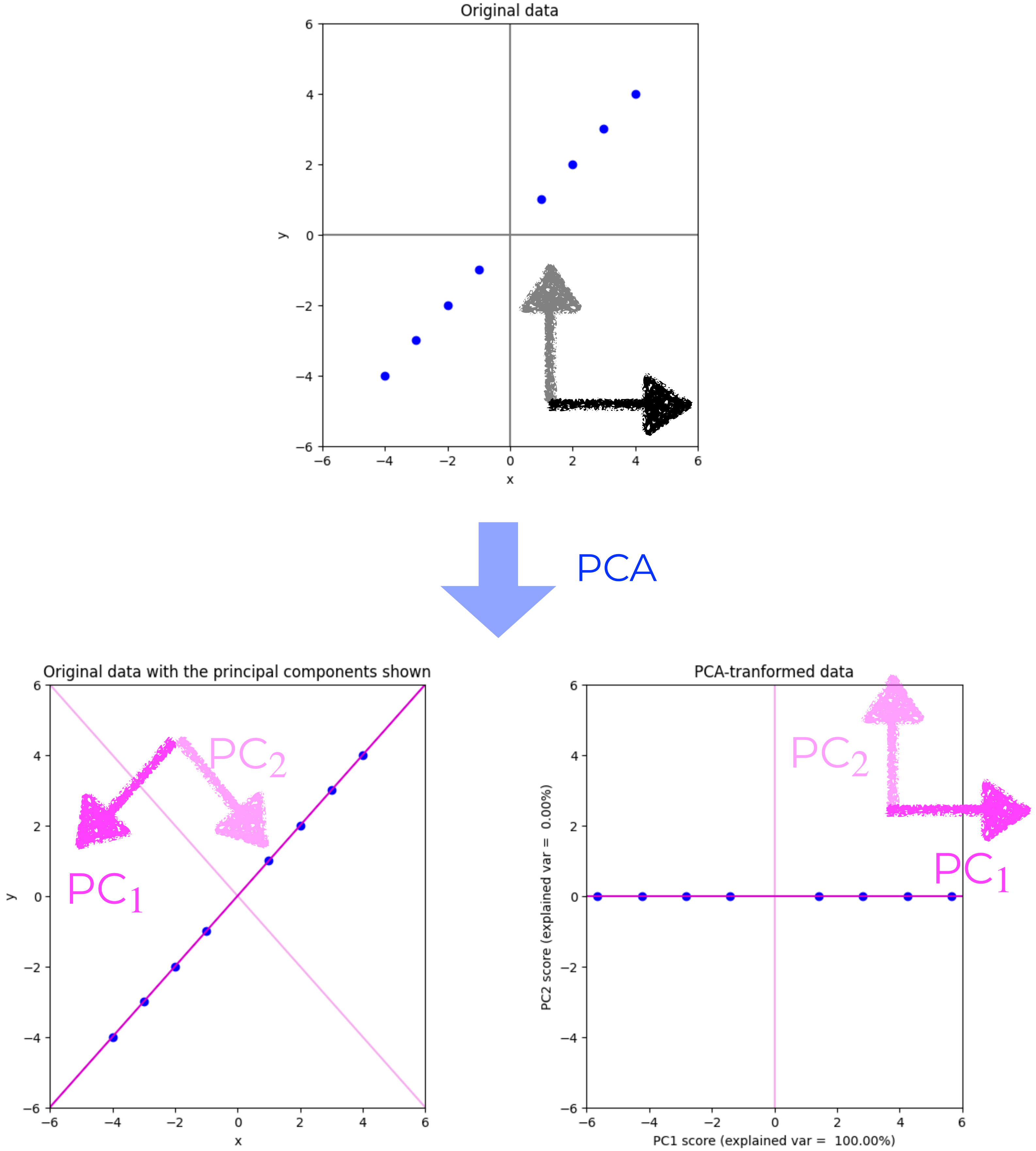

($PC_1$ score, $PC_2$ score)

How much you move along $PC_1$

How much you move along $PC_2$

## Original space

| | feature 1 | feature 2 |
|---|---|---|
| sample 1 | -4 | -4 |
| sample 2 | -3 | -3 |
| sample 3 | -2 | -2 |
| sample 4 | -1 | -1 |
| sample 5 | 1 | 1 |
| sample 6 | 2 | 2 |
| sample 7 | 3 | 3 |
| sample 8 | 4 | 4 |




## PCA

| $PC_1$ score | $PC_2$ score |
|---|---|
| 5.7 | 0 |
| 4.2 | 0 |
| 2.8 | 0 |
| 1.4 | 0 |
| -1.4 | 0 |
| -2.8 | 0 |
| -4.2 | 0 |
| -5.7 | 0 |

Brain Code Camp

# Principal Component Analysis (PCA)

## sklearn.decomposition.PCA

*class* `sklearn.decomposition.PCA`(*n_components=None, *, copy=True, whiten=False, svd_solver='auto', tol=0.0, iterated_power='auto', n_oversamples=10, power_iteration_normalizer='auto', random_state=None*) [source]

**Attributes:**

**components_ : *ndarray of shape (n_components, n_features)***

Principal axes in feature space, representing the directions of maximum variance in the data. Equivalently, the right singular vectors of the centered input data, parallel to its eigenvectors. The components are sorted by decreasing `explained_variance_`.

$PC_1, PC_2, \ldots$

**explained_variance_ : *ndarray of shape (n_components,)***

The amount of variance explained by each of the selected components. The variance estimation uses `n_samples - 1` degrees of freedom.

Equal to n_components largest eigenvalues of the covariance matrix of X.

*New in version 0.18.*

$\lambda_1, \lambda_2, \ldots$

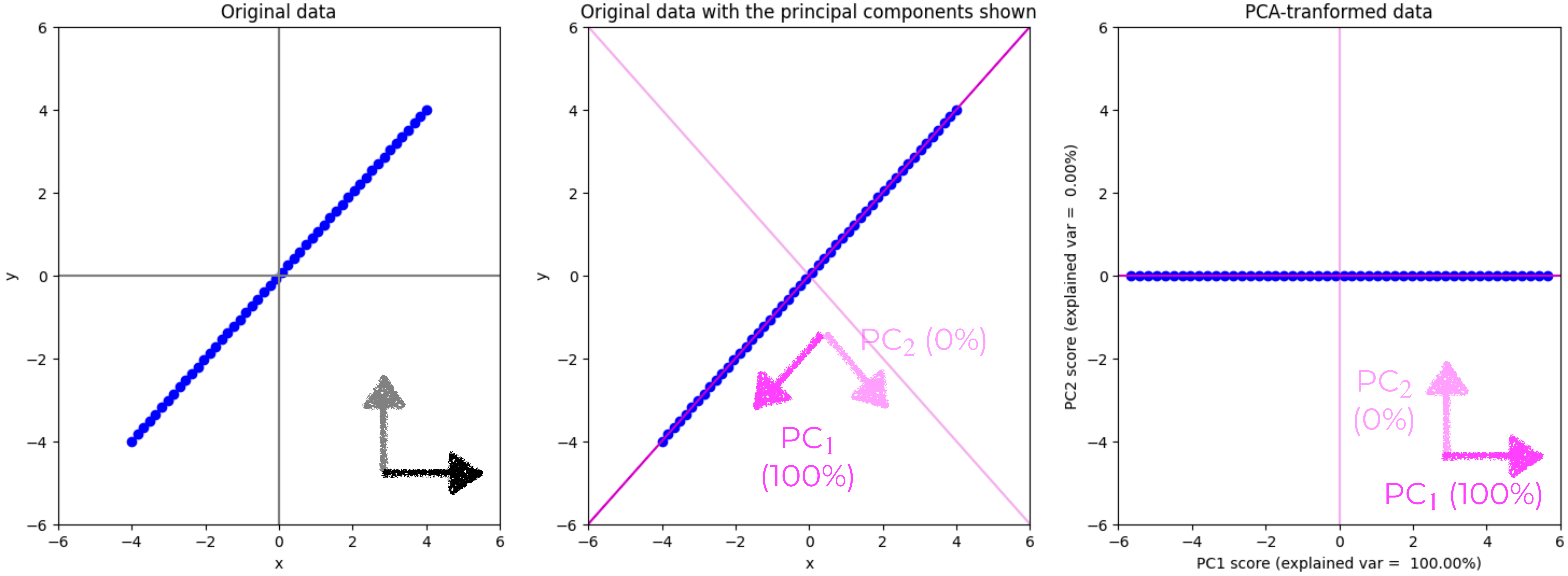
**explained_variance_ratio_ : *ndarray of shape (n_components,)***

Percentage of variance explained by each of the selected components.

If `n_components` is not set then all components are stored and the sum of the ratios is equal to 1.0.

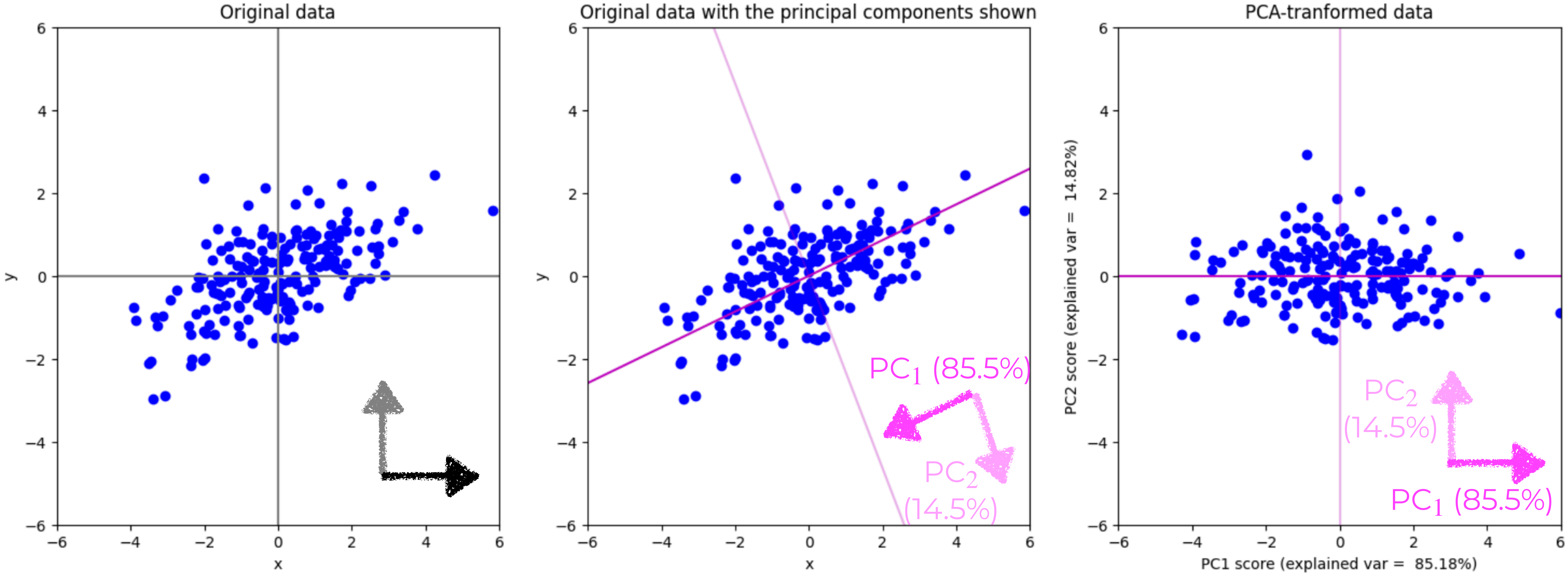
ex. 0.8, 0.2, ...

# PCA in 2D

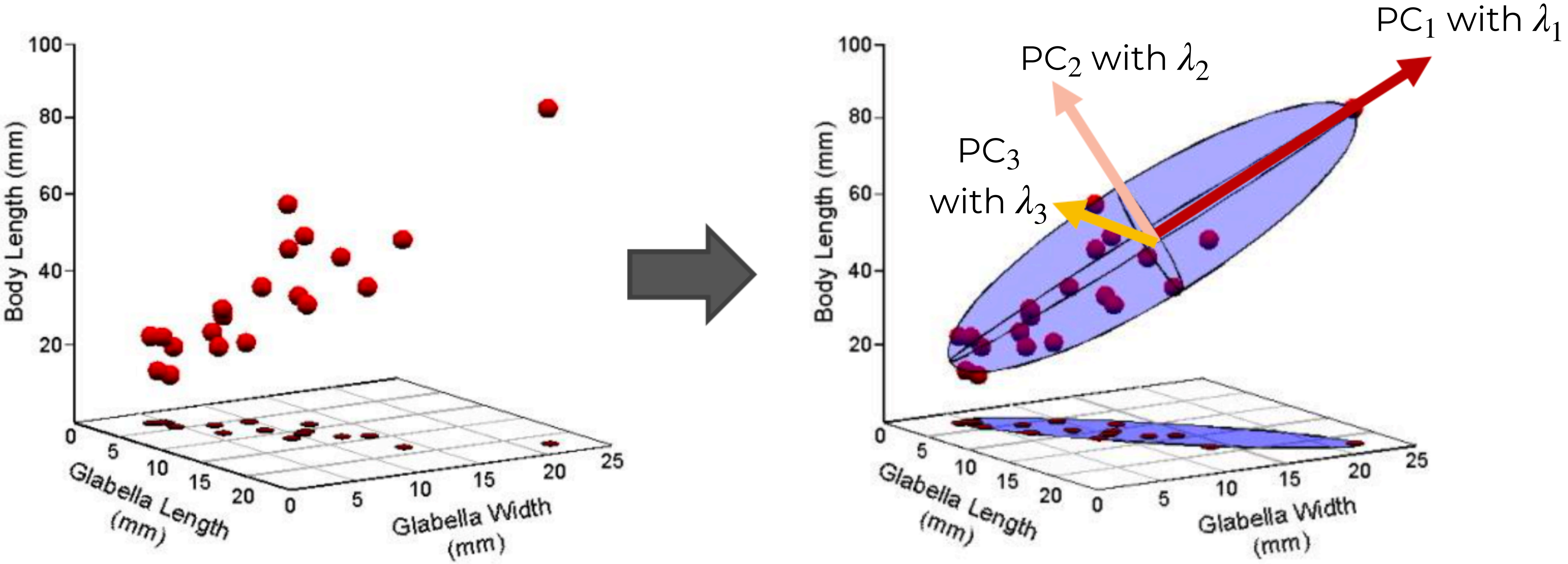


Original data
y
x
Original data with the principal components shown
y
x
PC2 (0%)
PC1
(100%)
PCA-tranformed data
PC2 score (explained var = 0.00%)
PC1 score (explained var = 100.00%)
PC2
(0%)
PC1 (100%)

# PCA in 2D


Original data
y
x
Original data with the principal components shown
y
x
PC1 (85.5%)
PC2 (14.5%)
PCA-tranformed data
PC2 score (explained var = 14.82%)
PC1 score (explained var = 85.18%)
PC2 (14.5%)
PC1 (85.5%)

# PCA in 3D



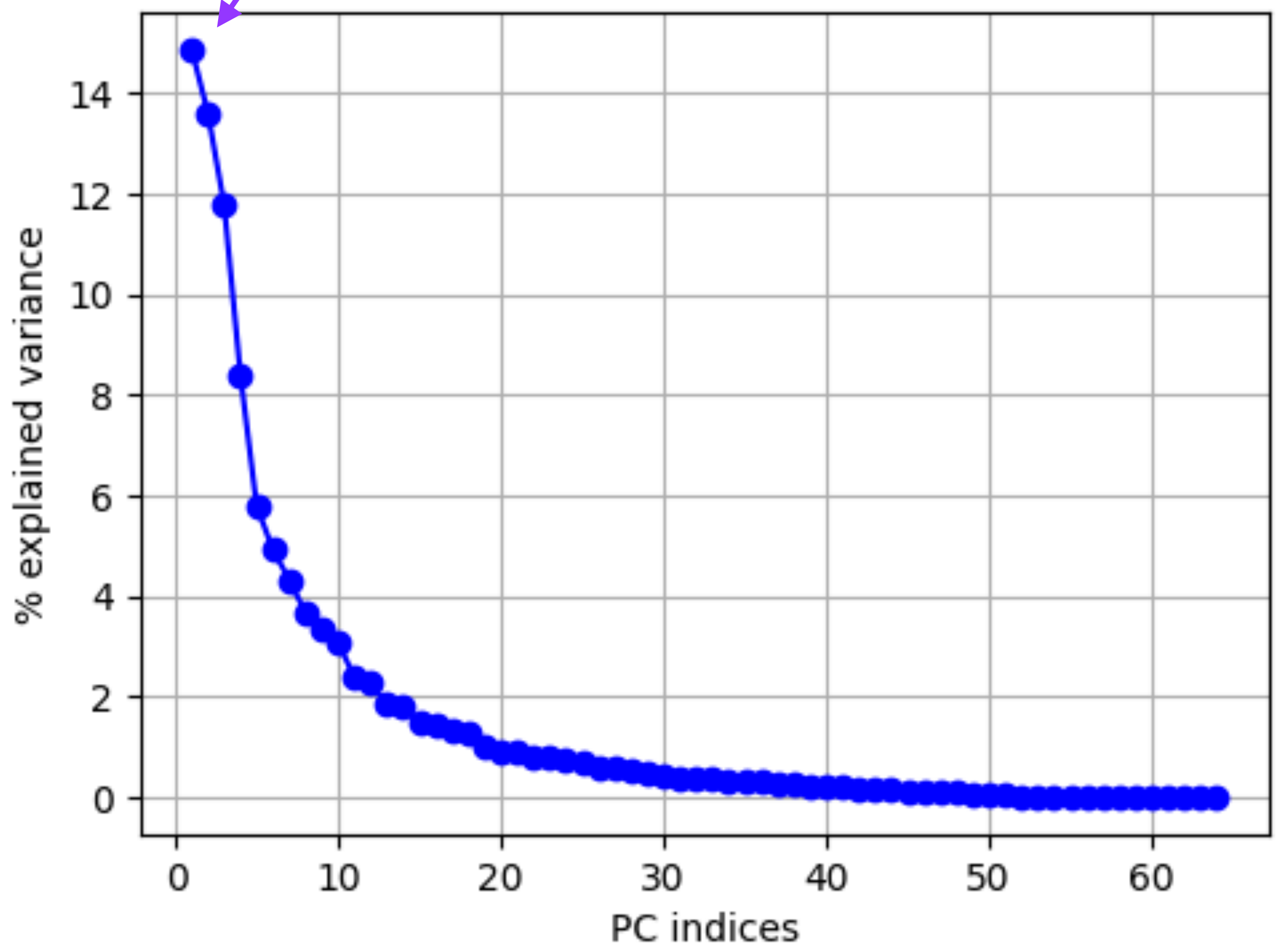

Source: the paleontological association

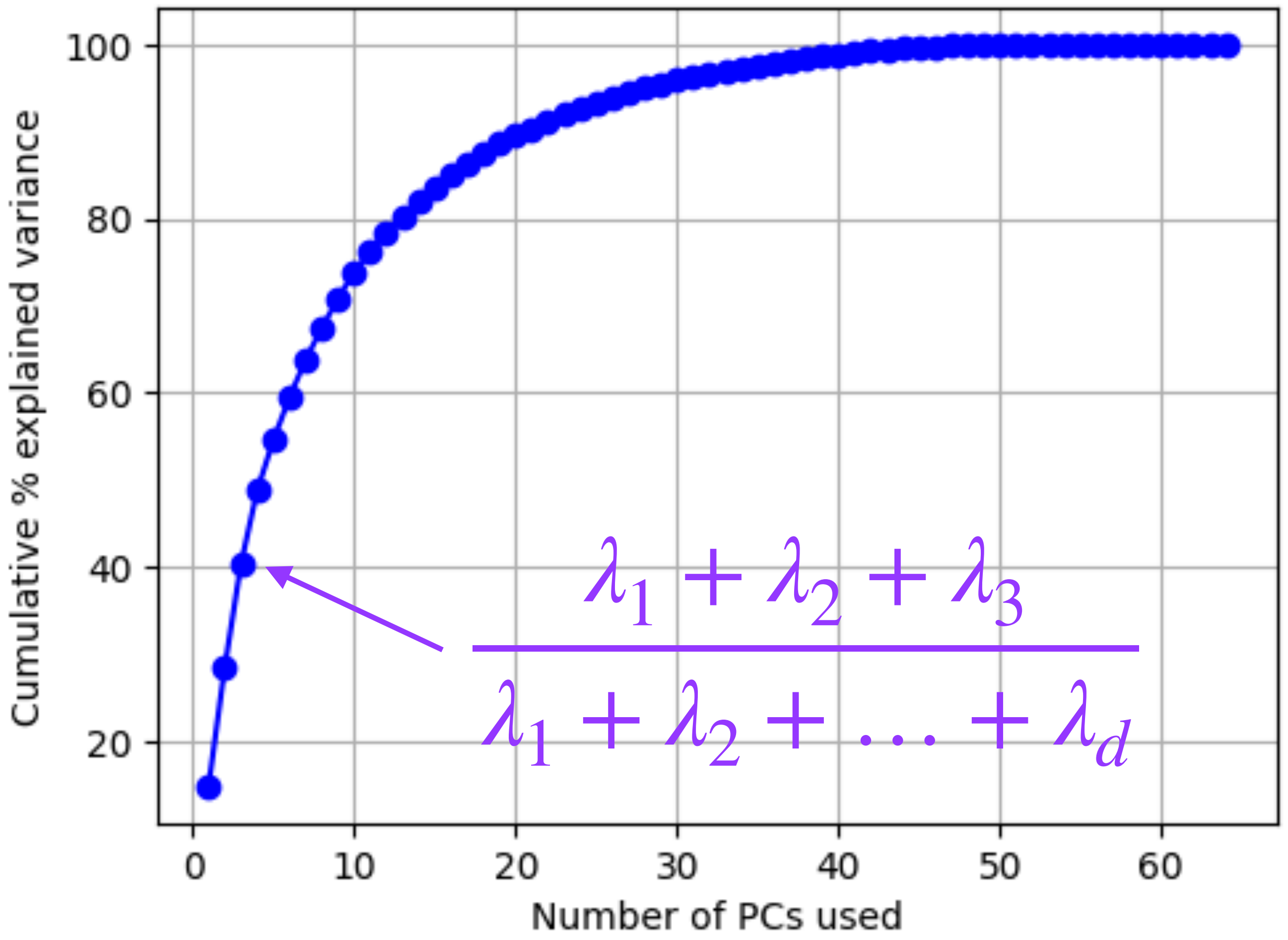
# PCA in High-dimensional Space

**d-dimensional space** d principal components (PCs) $\mathrm{PC}_1,\ \mathrm{PC}_2\ ,\ldots,\ \mathrm{PC}_d$

$$\frac{\lambda_1}{\lambda_1 + \lambda_2 + \ldots + \lambda_d}$$

explained variances $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_d$

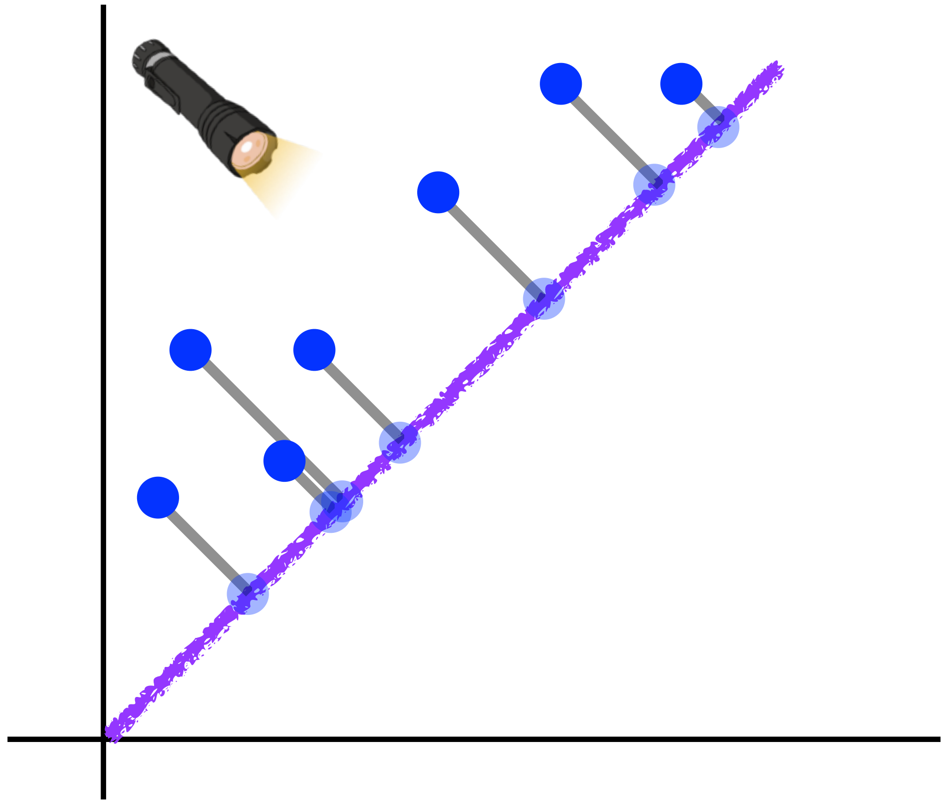



$$\frac{\lambda_1 + \lambda_2 + \lambda_3}{\lambda_1 + \lambda_2 + \ldots + \lambda_d}$$

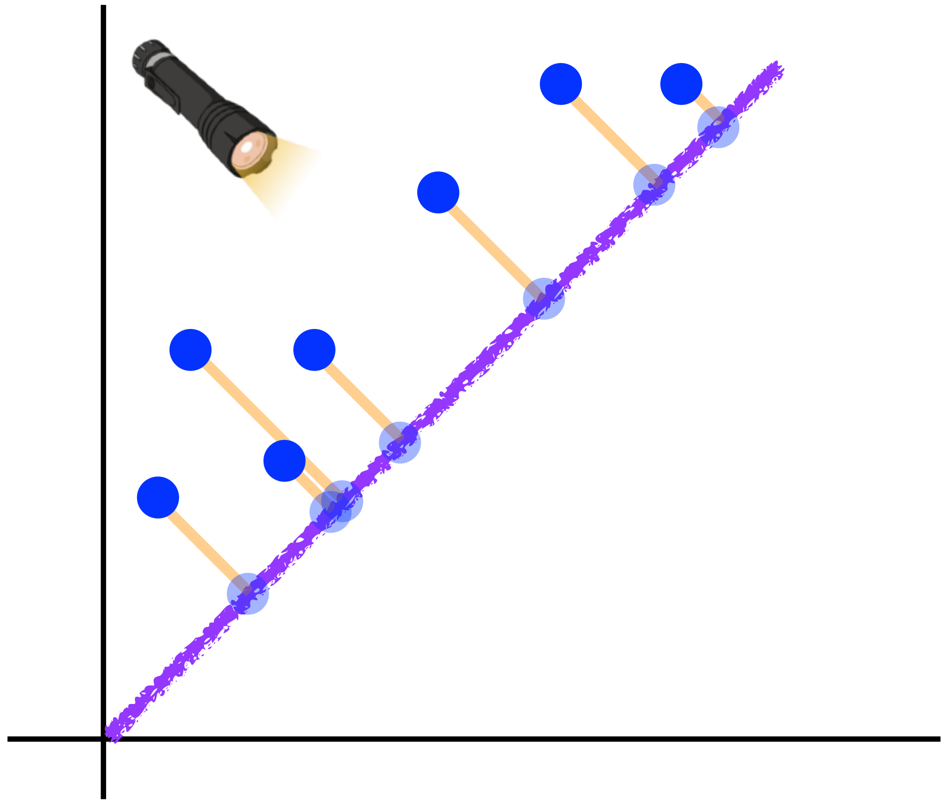
# Two Common Definitions of PCA

## Maximum variance formulation

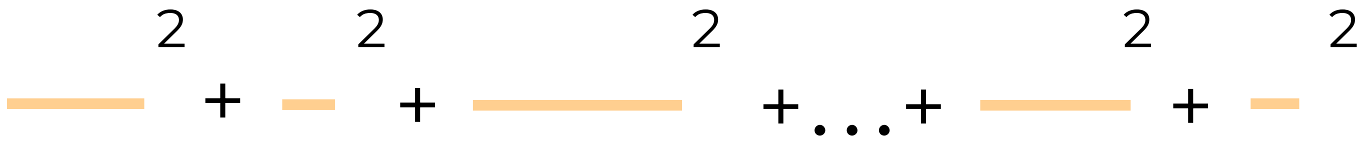


Maximizes the variance of the projected points

## Minimum-error formulation



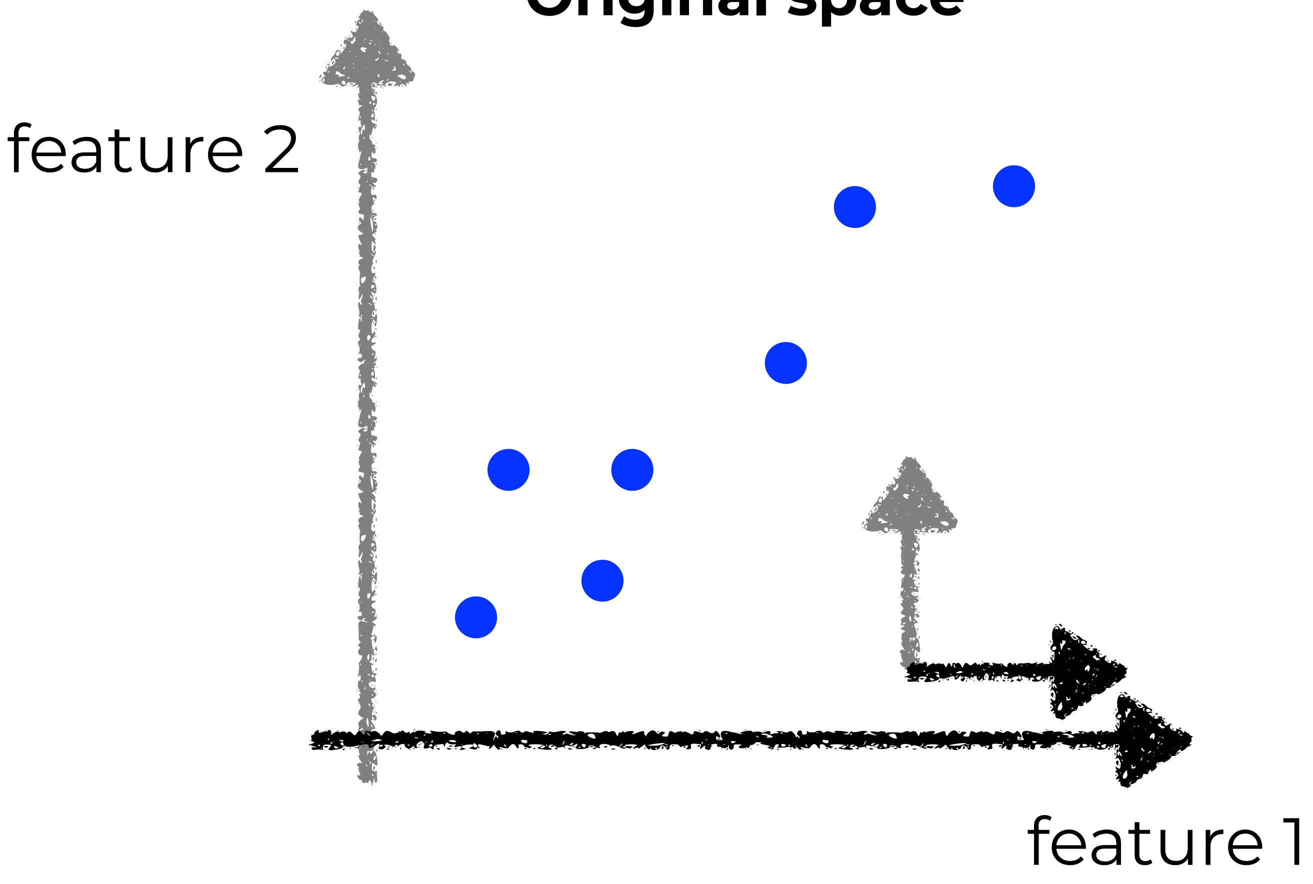
Minimizes the sum-of-squares of the projection errors

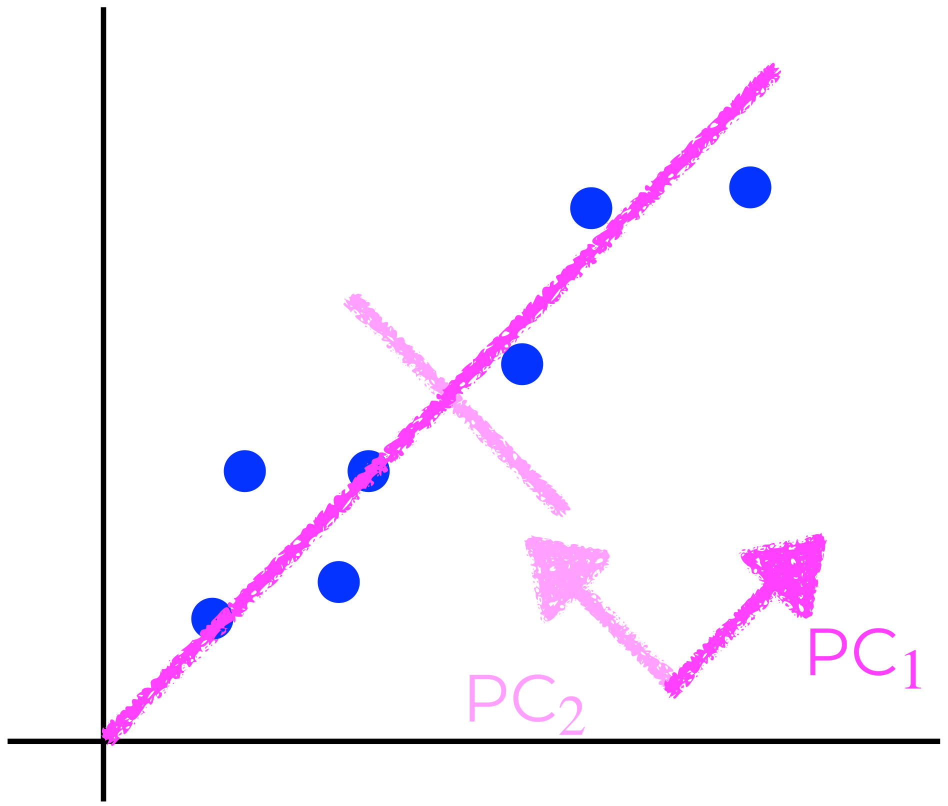


# Pros and Cons of PCA

- Each PC can be interpreted by looking at how it can be written as a linear combination of the original axes

**Original space**

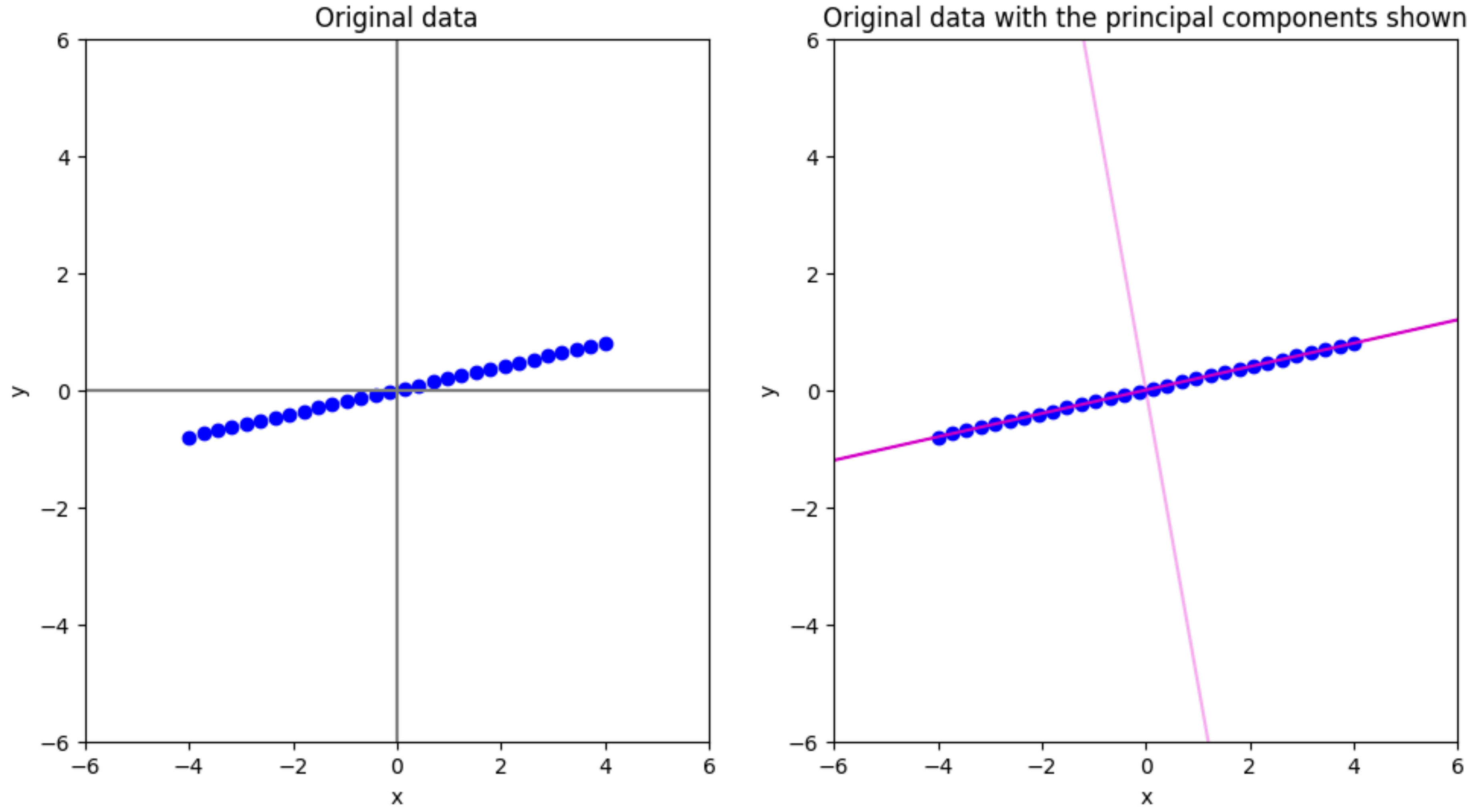



↗ $PC_1$ = 0.707 → + 0.707 ↑

# Pros and Cons of PCA

- Each PC can be interpreted by looking at how it can be written as a linear combination of the original axes



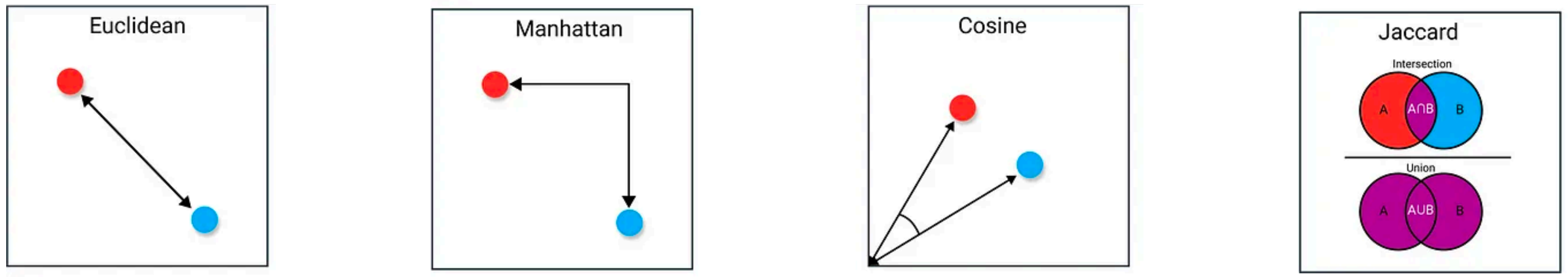

$PC_1$ = 0.98 + 0.20

# Pros and Cons of PCA

- Each PC can be interpreted by looking at how it can be written as a linear combination of the original axes

$$PC_1 = 0.98 \rightarrow + 0.20 \uparrow$$

- Highly correlated features tend to be grouped into the same PC
  - It can be used as a preprocessing step
    - Decorrelate the variables (the data have zero mean and unit variance)
    - Perform dimensionality reduction
  - We need to be careful with our interpretation
- PCA strictly preserves the Euclidean distance between data points, which may not be ideal in many situations




source: https://towardsdatascience.com/9-distance-measures-in-data-science-918109d069fa