SINGULAR VALUE DECOMPOSITION (SVD)

&

PRINCIPAL COMPONENT ANALYSIS (PCA)

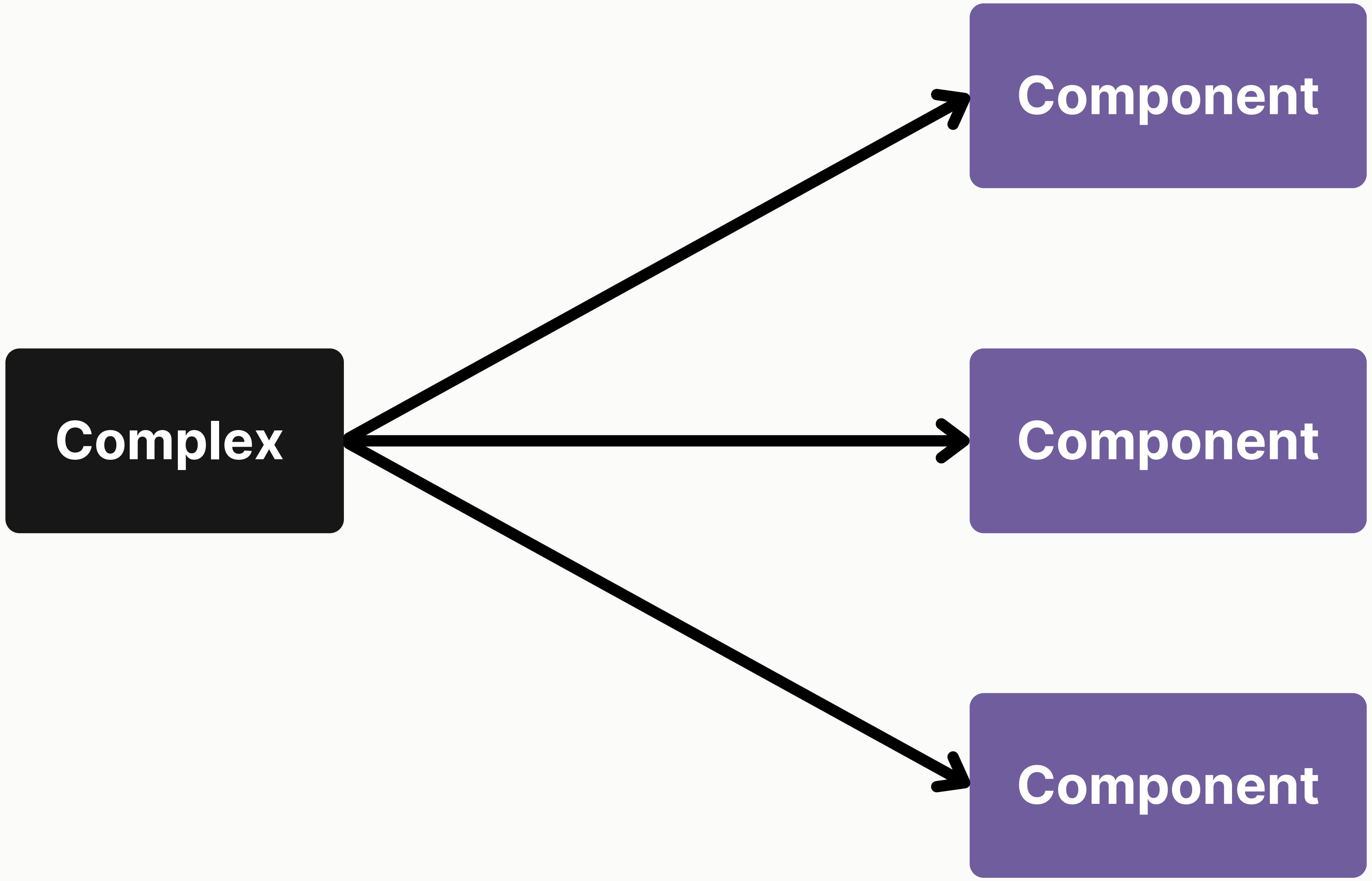
CS115

MỤC LỤC

1. Singular Value Decomposition (SVD)

1.1. What is SVD?

Idea



1.1. What is SVD?

Eigen decomposition

$$A = PDP^{-1}$$

1.1. What is SVD?

Eigen decomposition

Eigen Decomposition. → $$\boxed{A = PDP^{-1}}$$

1.1. What is SVD?

Eigen decomposition

Eigen Decomposition.

$$\boxed{A = PDP^{-1}}$$
$$AP = PD$$
$$Ap_i = Pd_i$$
$$Ap_i = p_i d_{ii}$$

1.1. What is SVD?

Eigen decomposition

Eigen Decomposition.

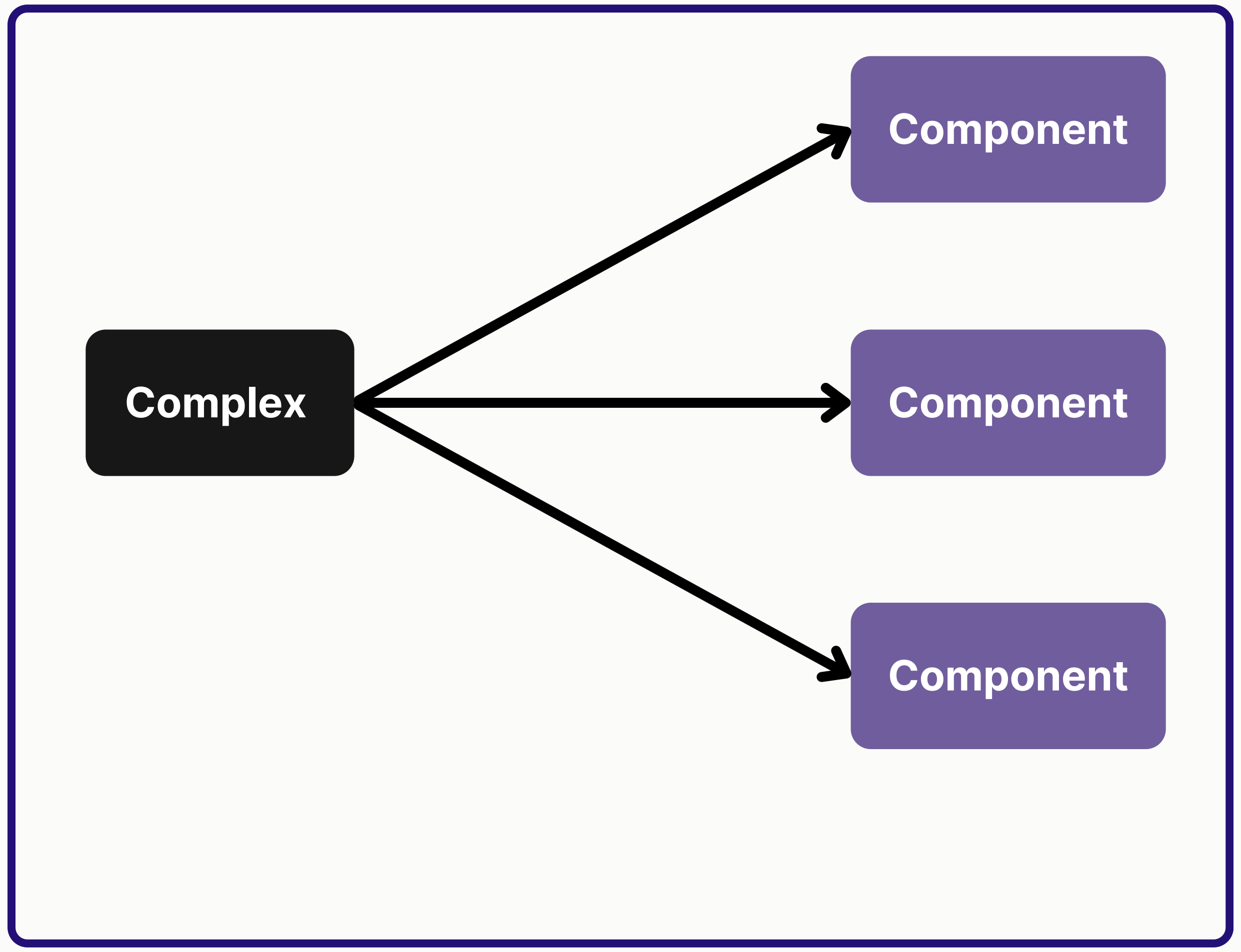
$$
\begin{aligned}
&\boxed{A = PDP^{-1}} \\
AP &= PD \\
Ap_i &= Pd_i \\
A\boxed{p_i} &= \boxed{p_i}\boxed{d_{ii}}
\end{aligned}
$$

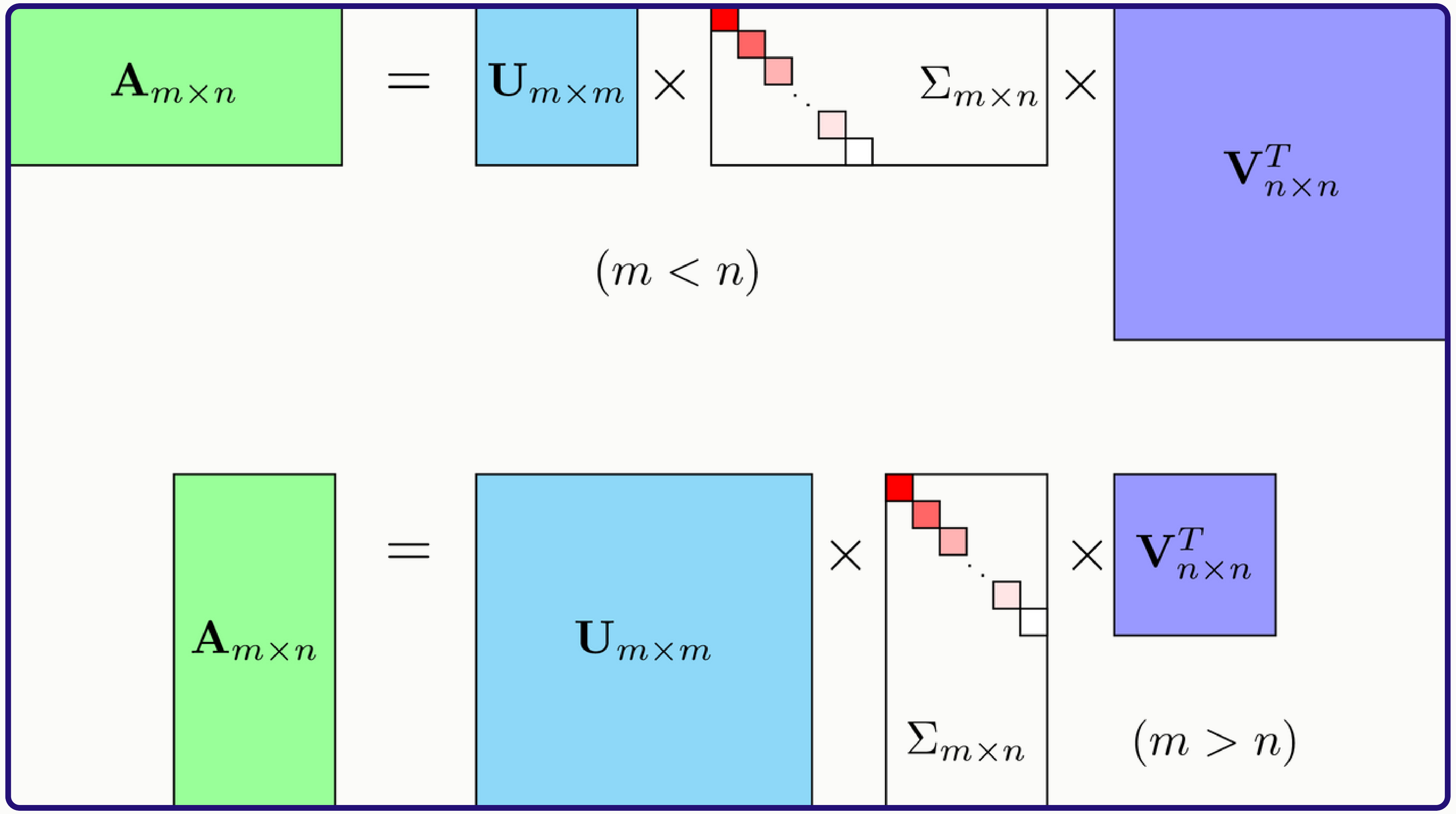
Eigen vector

Eigen value

1.1. What is SVD?

Singular Value Composition (SVD) ← Matrix factorization





idea

$$\mathbf{A}_{m\times n} = \mathbf{U}_{m\times m}\mathbf{\Sigma}_{m\times n}(\mathbf{V}_{n\times n})^T$$

1.1. What is SVD?

Singular Value Composition (SVD)

$$\boxed{\mathbf{A}_{m\times n} = \mathbf{U}_{m\times m}\mathbf{\Sigma}_{m\times n}(\mathbf{V}_{n\times n})^T}$$

$$\mathbf{A}\mathbf{A}^T = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T(\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T)^T = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T\mathbf{V}\mathbf{\Sigma}^T\mathbf{U}^T = \mathbf{U}\mathbf{\Sigma}\mathbf{\Sigma}^T\mathbf{U}^T = \mathbf{U}\mathbf{\Sigma}\mathbf{\Sigma}^T\mathbf{U}^{-1}$$

1.1. What is SVD?

Singular Value Composition (SVD)

$$\mathbf{A}_{m\times n} = \mathbf{U}_{m\times m}\mathbf{\Sigma}_{m\times n}(\mathbf{V}_{n\times n})^T$$

$$A = PDP^{-1}$$

$$Ap_i = p_i d_{ii}$$

$$\mathbf{AA}^T = \mathbf{U\Sigma V}^T(\mathbf{U\Sigma V}^T)^T = \mathbf{U\Sigma V}^T\mathbf{V\Sigma}^T\mathbf{U}^T = \mathbf{U\Sigma\Sigma}^T\mathbf{U}^T = \mathbf{U}\boxed{\mathbf{\Sigma\Sigma}^T}\mathbf{U}^{-1}$$

$$\begin{pmatrix} \lambda_1 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 \\ 0 & \lambda_2 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 \\ 0 & 0 & \lambda_3 & 0 & \cdots & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \lambda_4 & \cdots & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \ddots & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \cdots & \lambda_K & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & \lambda_{K+1} & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \cdots & \vdots & \vdots & \vdots & \lambda_N \end{pmatrix} \qquad \lambda_i = {\sigma_i}^2$$

$$\boxed{\sigma_1^2, \sigma_2^2 \ldots \sigma_m^2}$$

$$\boxed{\sigma_j}$$

Singular Value

1.1. What is SVD?

Singular Value Composition (SVD)

$$\boxed{\mathbf{A}_{m\times n} = \mathbf{U}_{m\times m}\mathbf{\Sigma}_{m\times n}(\mathbf{V}_{n\times n})^T}$$

column

left-singular vectors

$$\mathbf{A}\mathbf{A}^T = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T(\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T)^T = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T\mathbf{V}\mathbf{\Sigma}^T\mathbf{U}^T = \boxed{\mathbf{U}}\mathbf{\Sigma}\mathbf{\Sigma}^T\mathbf{U}^T = \mathbf{U}\mathbf{\Sigma}\mathbf{\Sigma}^T\mathbf{U}^{-1}$$

$$[\mathbf{e}_1 \quad \mathbf{e}_2 \; \ldots \mathbf{e}_N]$$

1.1. What is SVD?

Singular Value Composition (SVD)

$$\boxed{\mathbf{A}_{m\times n} = \mathbf{U}_{m\times m}\mathbf{\Sigma}_{m\times n}(\mathbf{V}_{n\times n})^T}$$

$$\mathbf{A}\mathbf{A}^T = \mathbf{U\Sigma V}^T(\mathbf{U\Sigma V}^T)^T = \mathbf{U\Sigma V}^T\mathbf{V\Sigma}^T\mathbf{U}^T = \boxed{\mathbf{U}}\mathbf{\Sigma\Sigma}^T\mathbf{U}^T = \mathbf{U}\boxed{\mathbf{\Sigma\Sigma}^T}\mathbf{U}^{-1}$$

column

left-singular vectors

$$\boxed{\sigma_1^2, \sigma_2^2 \ldots \sigma_m^2}$$

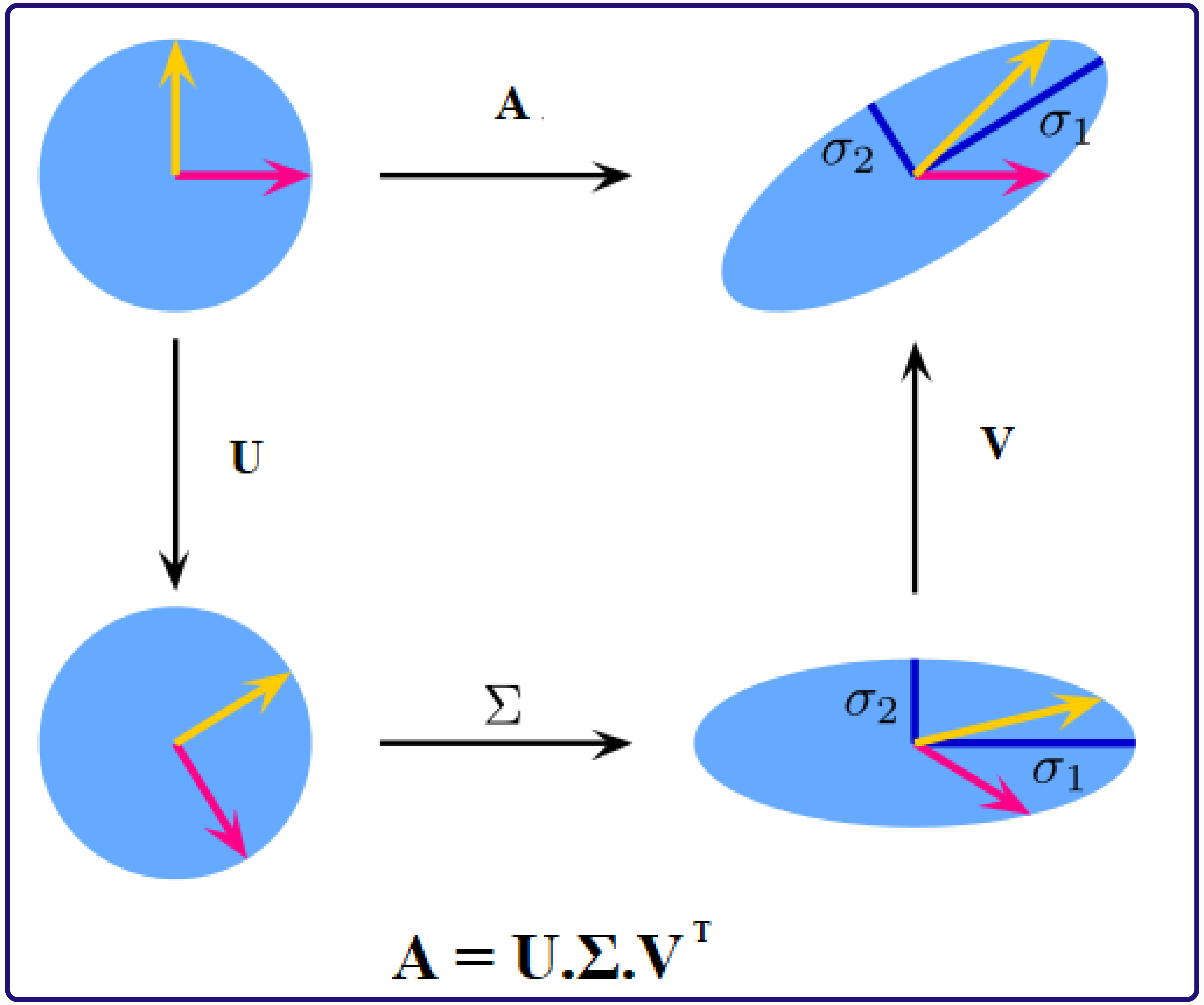
$$\boxed{\sigma_j}$$ Singular Value

right-singular vectors

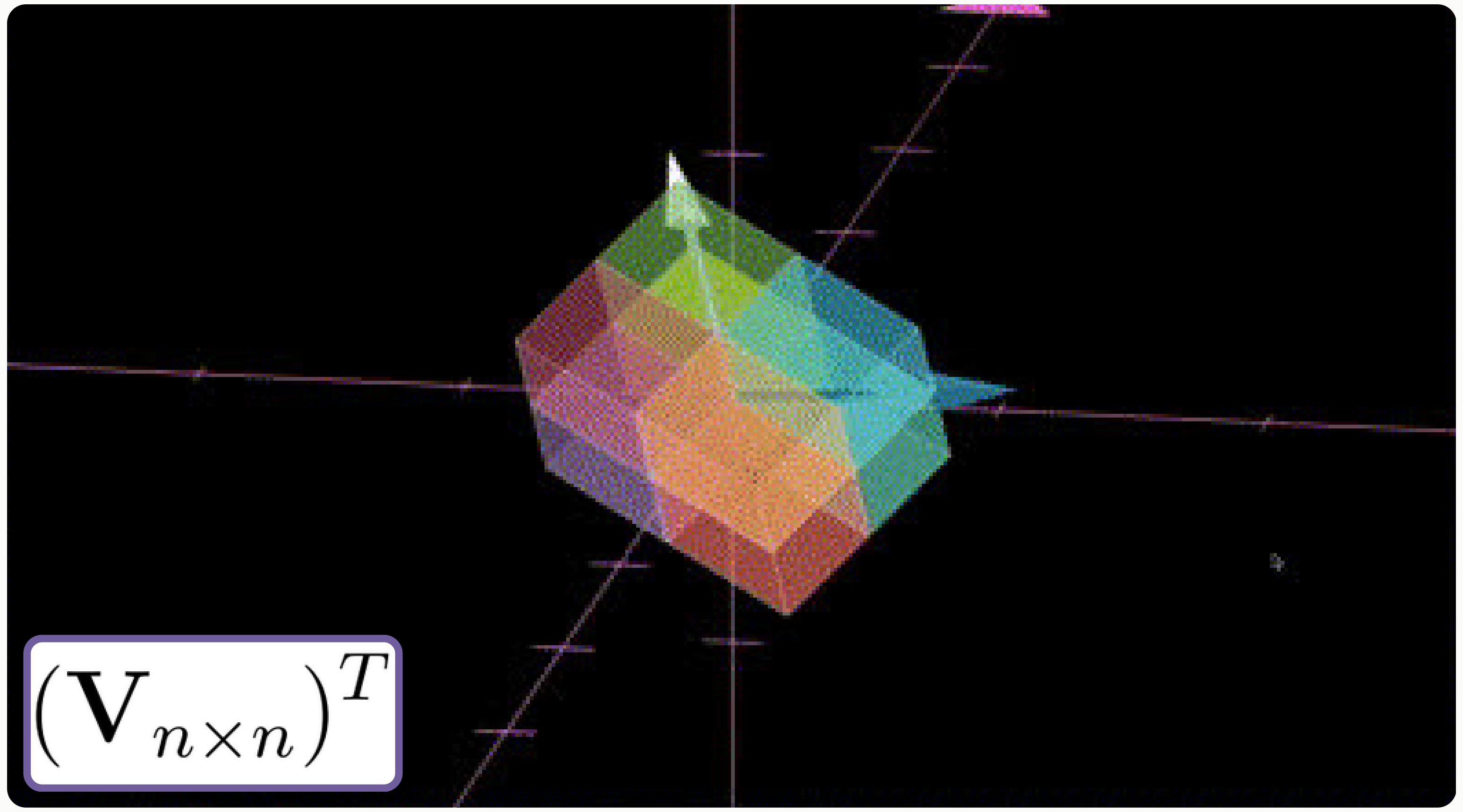
column

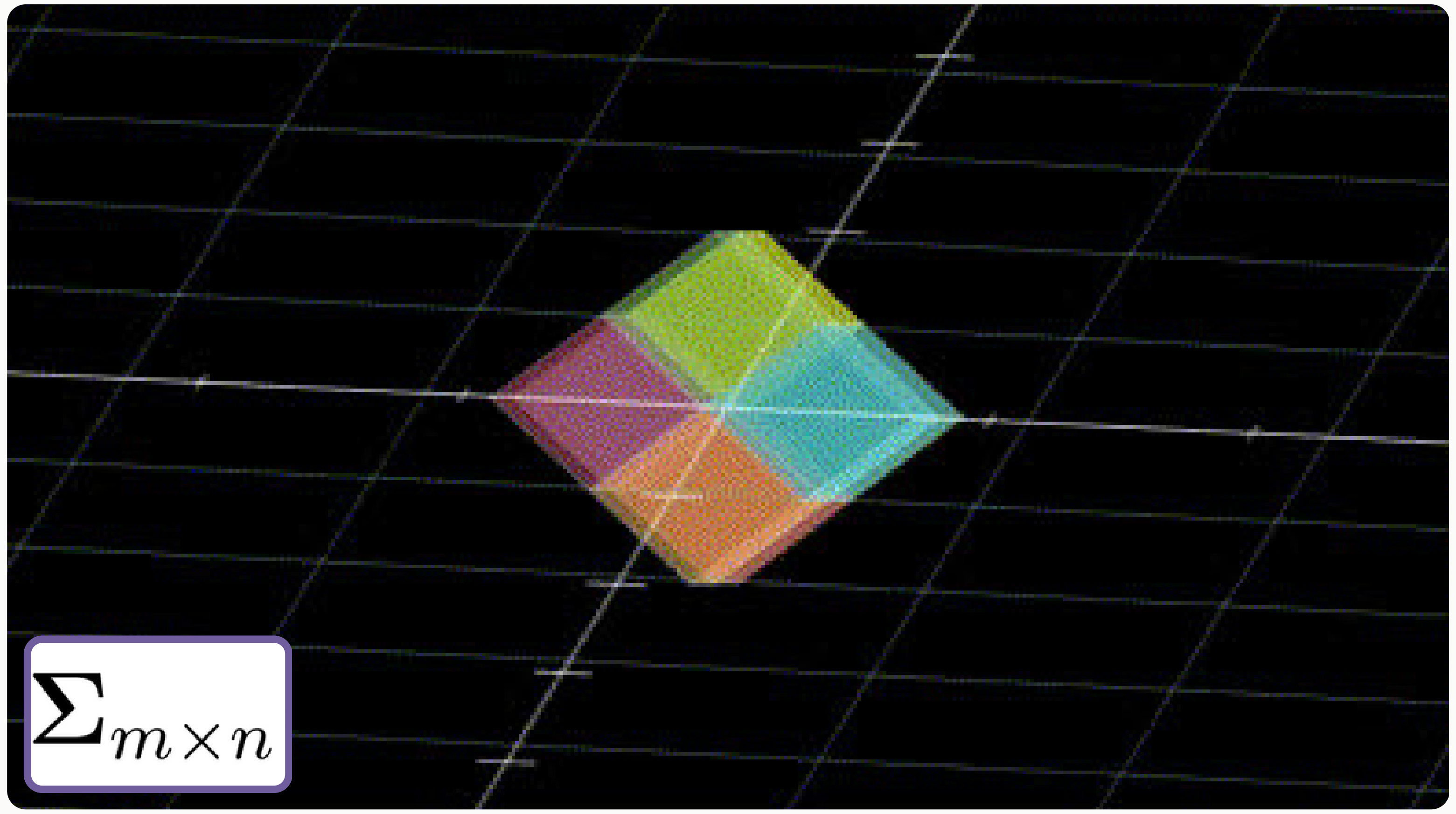
$$\mathbf{A}^T\mathbf{A} = (\mathbf{U\Sigma V}^T)^T\mathbf{U\Sigma V}^T = \mathbf{V\Sigma}^T\mathbf{U}^T\mathbf{U\Sigma V}^T = \boxed{\mathbf{V}}\mathbf{\Sigma}^T\mathbf{\Sigma V}^T = \mathbf{V\Sigma}^T\mathbf{\Sigma V}^{-1}$$

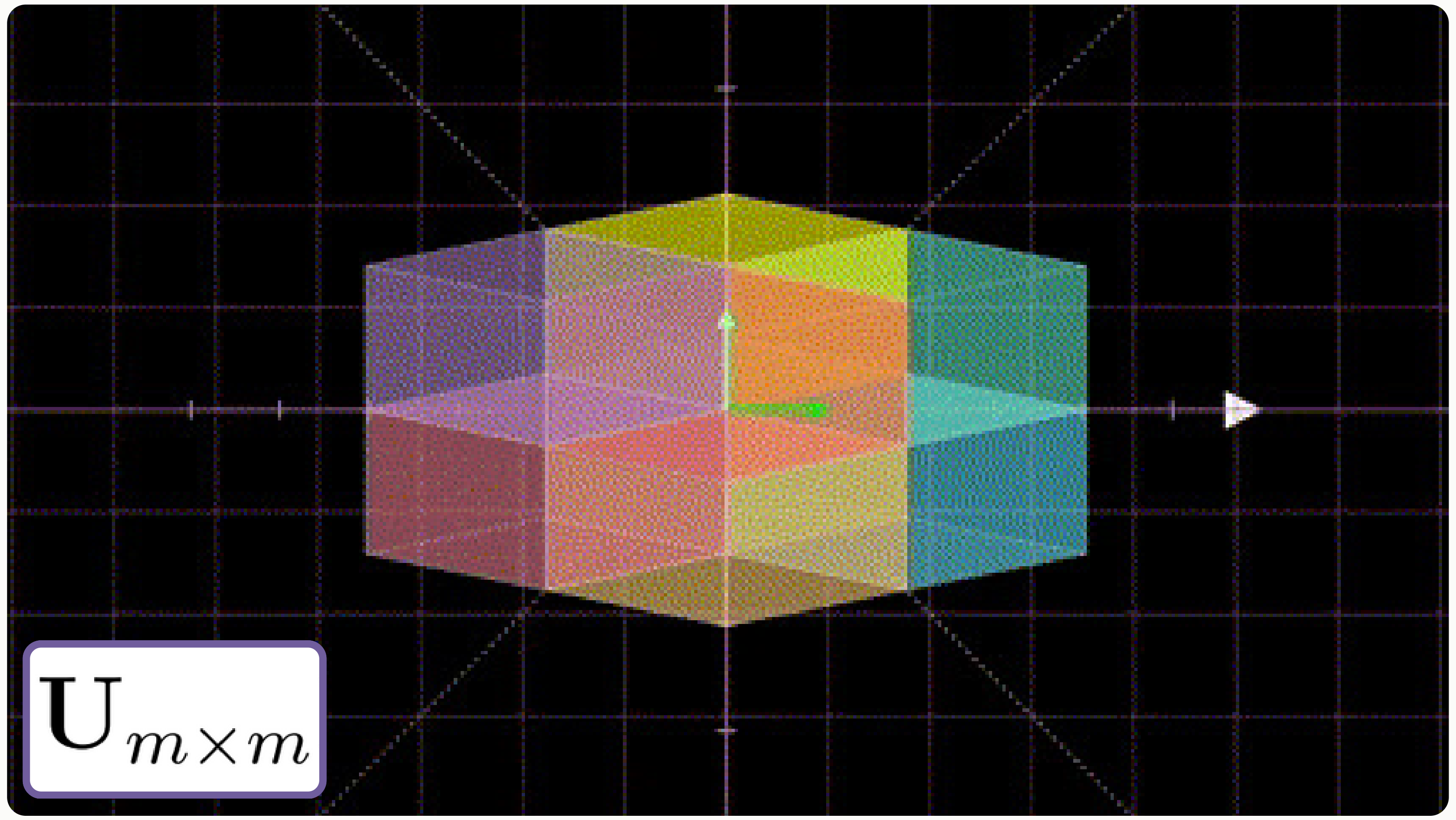
1.1. What is SVD?



$$\mathbf{A}_{m\times n} = \mathbf{U}_{m\times m}\mathbf{\Sigma}_{m\times n}(\mathbf{V}_{n\times n})^T$$







01

1.1. What is SVD?

Procedure

Step 1: Finding the eigenvalues

Step 2: Finding left/ right singular vectors

Step 3: Finding right/ left singular vectors

1.1. What is SVD?

Procedure

Step 1: Finding the eigenvalues and right singular vectors.

```
#Define svd function
def mySVD(A):
    m, n= A.shape

    #calculate eigen values and right singular vectors
    eigenvalues, V = np.linalg.eig(A.T @ A)
    eigenvalues = np.round(eigenvalues, decimals = 4)

    #sort the eigen values and right singular vectors descending
    idx = eigenvalues.argsort()[::-1]
    eigenvalues = (eigenvalues[idx])
    V = V[:,idx]
    VT = V.T
```

1.1. What is SVD?

Procedure

Step 2: Finding the singular values by square root eigen values.

```
#calc the singular values by square root eigen values
singularvalues = np.sqrt(eigenvalues)
```

Procedure

Step 3: Finding the right singular vectors.

```
#calc left singular vectors
U = (A @ V)[:, 0:m] / singularvalues[0:m]
U = U[:, :m]

S = np.diag(singularvalues)[0:m, :] #truncate the 0 values

return U, S, VT
```

1.2. SVD - How?

Some SVDs

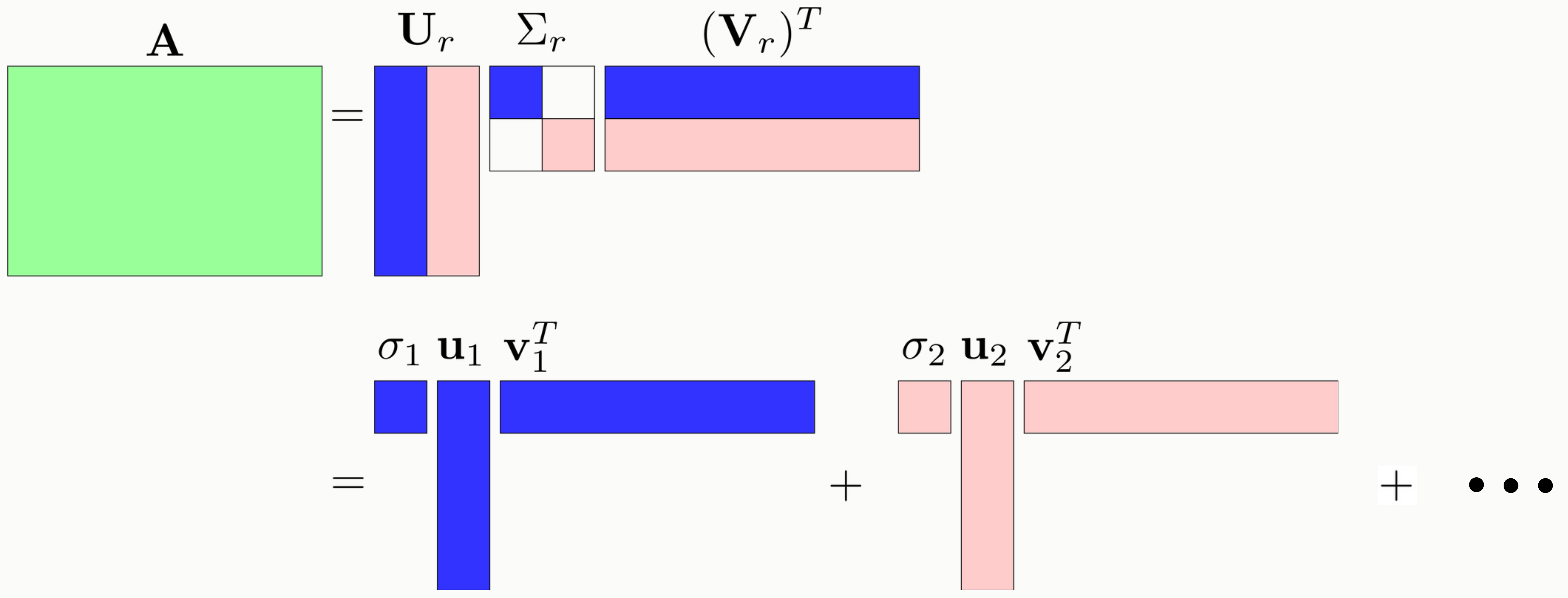
- Compact SVD
- Truncate SVD
- Best(Low) Rank ‘*k*’ Approximation

1.2. SVD - How?

SVDs -> Compact SVD

$$\boxed{\mathbf{A}_{m\times n} = \mathbf{U}_{m\times m}\mathbf{\Sigma}_{m\times n}(\mathbf{V}_{n\times n})^T}$$

$$\mathbf{A} = \mathbf{U}_r\Sigma_r(\mathbf{V}_r)^T = \sigma_1\mathbf{u}_1\mathbf{v}_1^T + \sigma_2\mathbf{u}_2\mathbf{v}_2^T + \cdots + \sigma_r\mathbf{u}_r\mathbf{v}_r^T$$

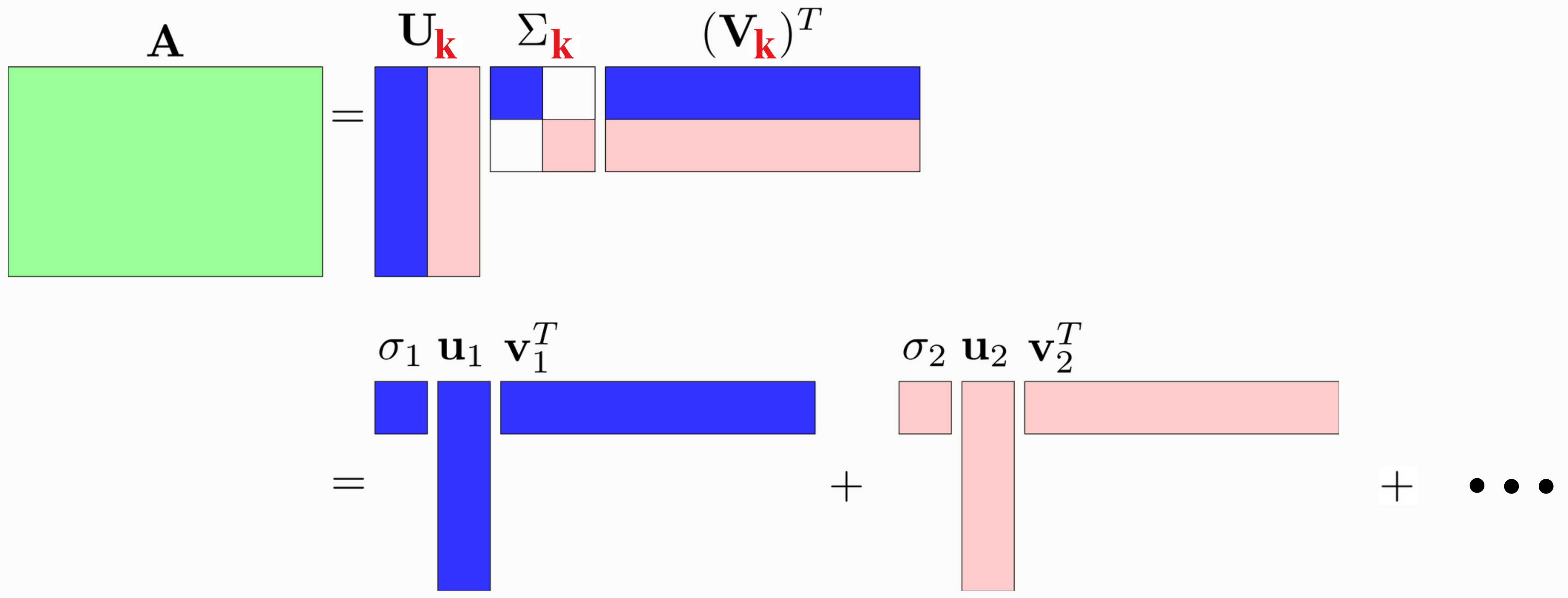


1.2. SVD - How?

SVDs -> Truncated SVD

$$\boxed{\mathbf{A}_{m\times n} = \mathbf{U}_{m\times m}\mathbf{\Sigma}_{m\times n}(\mathbf{V}_{n\times n})^T}$$

$$\mathbf{A} \approx \hat{\mathbf{A}} = U_k \Sigma_k V_k^T = \sigma_1 \mathbf{u}_1 \mathbf{v}_1^T + \sigma_2 \mathbf{u}_2 \mathbf{v}_2^T + \cdots + \sigma_k \mathbf{u}_k \mathbf{v}_k^T$$



1.2. SVD - How?

SVDs -> Truncated SVD

Theorem: $$\boxed{||\mathbf{A} - \mathbf{A}_k||_F^2 = \sum_{i=k+1}^{r} \sigma_i^2}$$

The error will equal to total square of the cut-off eigenvalues in truncated SVD.

With $k = 0$, we got: $$\boxed{||\mathbf{A}||_F^2 = \sum_{i=1}^{r} \sigma_i^2}$$

$$\boxed{\frac{||\mathbf{A} - \mathbf{A}_k||_F^2}{||\mathbf{A}||_F^2} = \frac{\sum_{i=k+1}^{r} \sigma_i^2}{\sum_{j=1}^{r} \sigma_j^2}}$$

1.2. SVD - How?

SVDs -> Best(Low) Rank ‘k’ Approximation

$$\min_{\mathbf{A}} ||\mathbf{X} - \mathbf{A}||_F$$
$$\text{s.t. } \operatorname{rank}(\mathbf{A}) = K$$

$$\mathbf{A} \approx \hat{\mathbf{A}} = U_k \Sigma_k V_k^T = \sigma_1 \mathbf{u}_1 \mathbf{v}_1^T + \sigma_2 \mathbf{u}_2 \mathbf{v}_2^T + \cdots + \sigma_k \mathbf{u}_k \mathbf{v}_k^T$$

1.3. SVD - Summarise

SVD → **untangle data into independent components**

$$\boxed{\mathbf{A}_{m\times n} = \mathbf{U}_{m\times m}\mathbf{\Sigma}_{m\times n}(\mathbf{V}_{n\times n})^T}$$

$$\boxed{\mathbf{A} \approx \hat{\mathbf{A}} = U_k \Sigma_k V_k^T = \sigma_1 \mathbf{u}_1 \mathbf{v}_1^T + \sigma_2 \mathbf{u}_2 \mathbf{v}_2^T + \cdots + \sigma_k \mathbf{u}_k \mathbf{v}_k^T}$$

1.3. SVD - Summarise

SVD

untangle data into independent components

$$\boxed{\mathbf{A}_{m\times n} = \mathbf{U}_{m\times m}\mathbf{\Sigma}_{m\times n}(\mathbf{V}_{n\times n})^T}$$

$$\boxed{\mathbf{A} \approx \hat{\mathbf{A}} = U_k \Sigma_k V_k^T = \sigma_1 \mathbf{u}_1 \mathbf{v}_1^T + \sigma_2 \mathbf{u}_2 \mathbf{v}_2^T + \cdots + \sigma_k \mathbf{u}_k \mathbf{v}_k^T}$$

Pros:

- Simplifies data
- Removes noise
- Data compression
- ...

Cons:

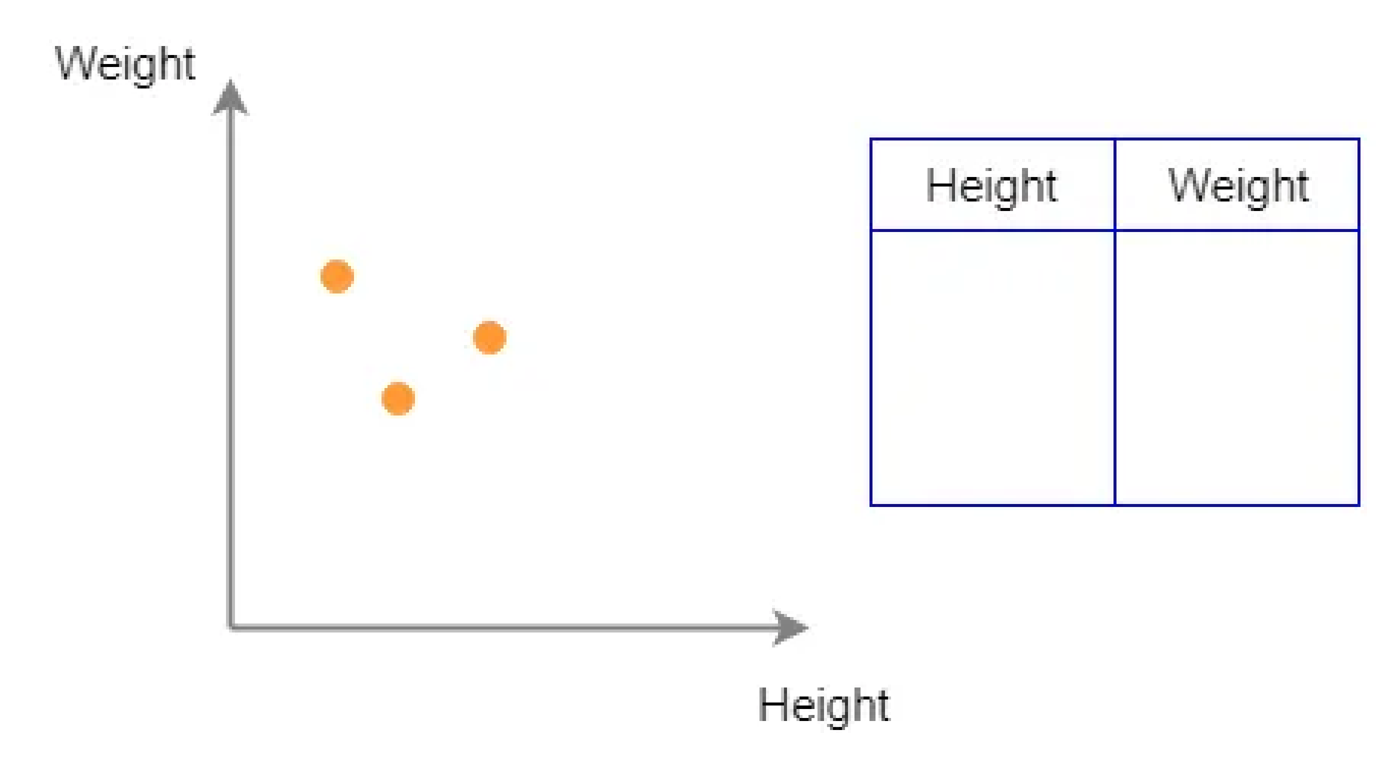
- Transformed data may be difficult to understand
- Limitations in non-linear relationships
- ...

2. Principle Component Analysis (PCA)

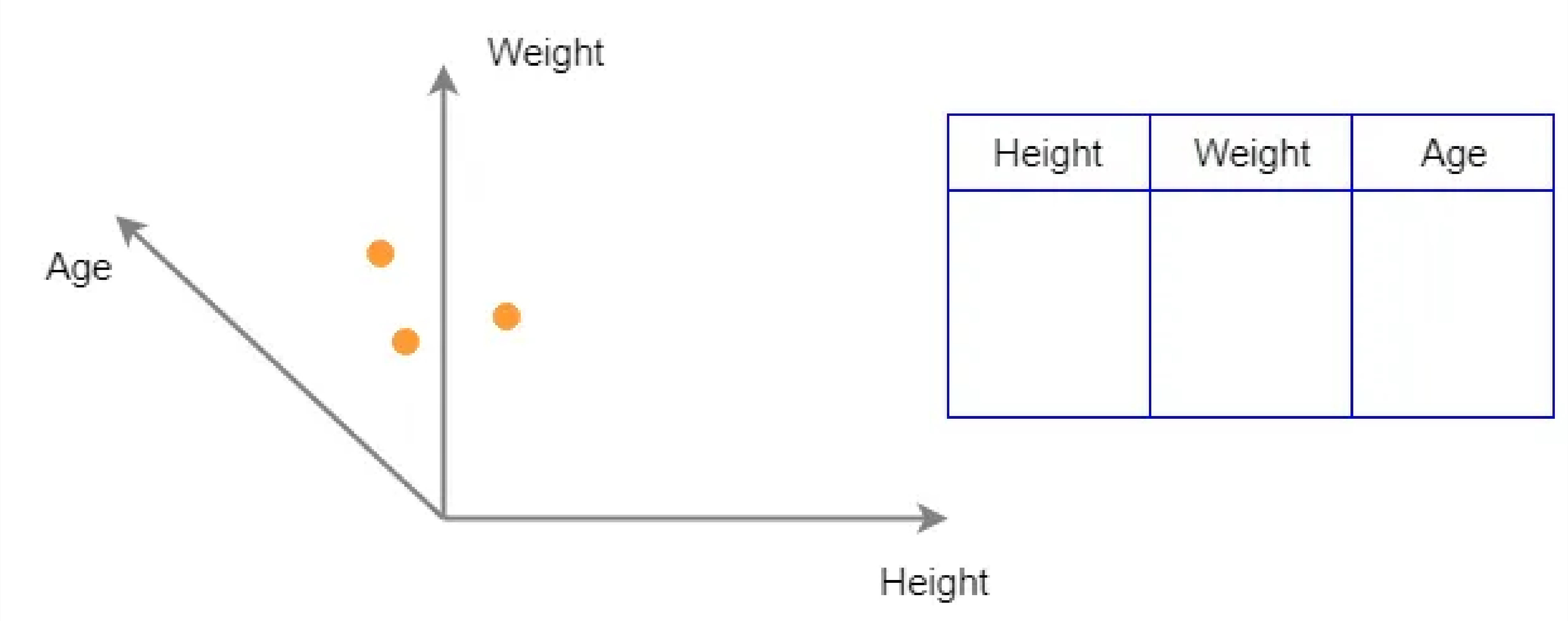
2.1. Why PCA?

Dataset's high dimensionality

few



a lot



→ **a lot of variables to consider.**

→ **high-dimensional data causes challenges**

2.1. Why PCA?

Dimensionality reduction

lose some percentage but

- less cost
- prevent overfitting
- data visualization
- ...

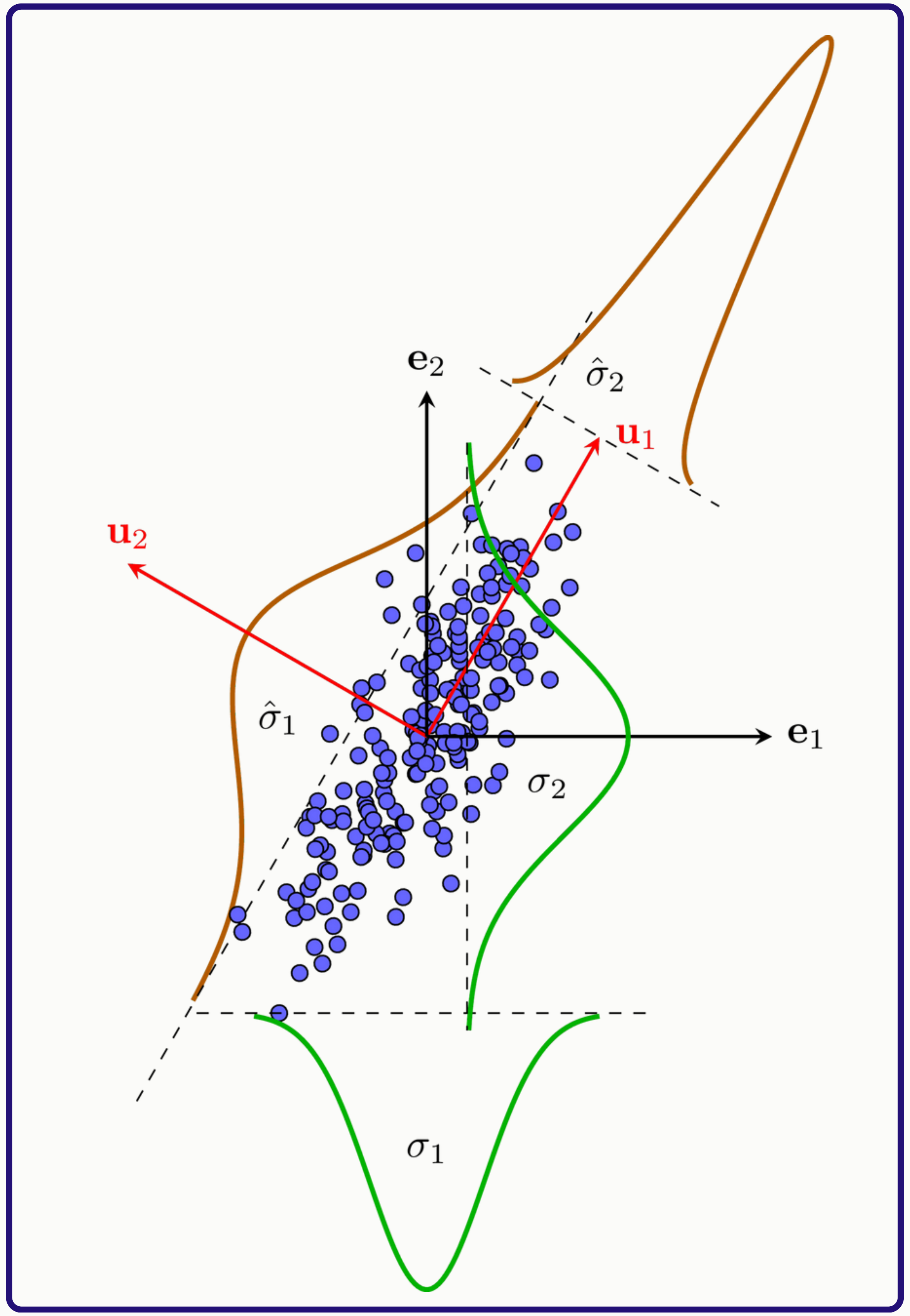
2.2. What is PCA?

Idea

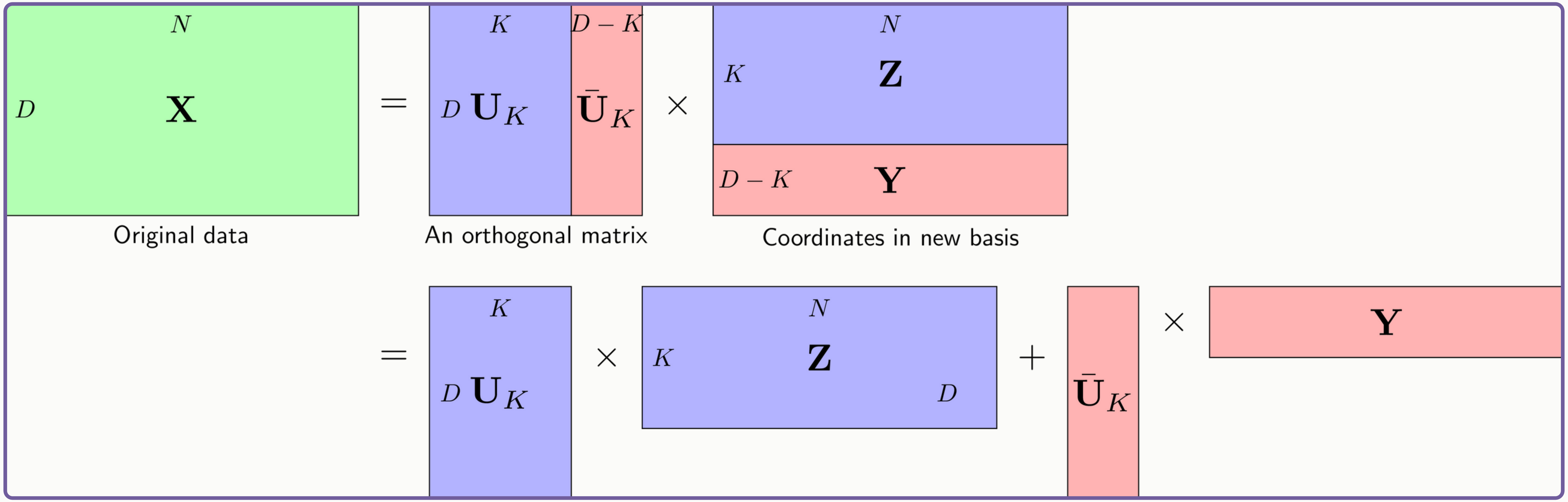
D $\longrightarrow$ **K**

(K<D)

- The idea of PCA is plotting the data into a **new basis system**
- The importance of the components is **significantly different**
  -> ignoring the **least important** component



2.2. What is PCA?

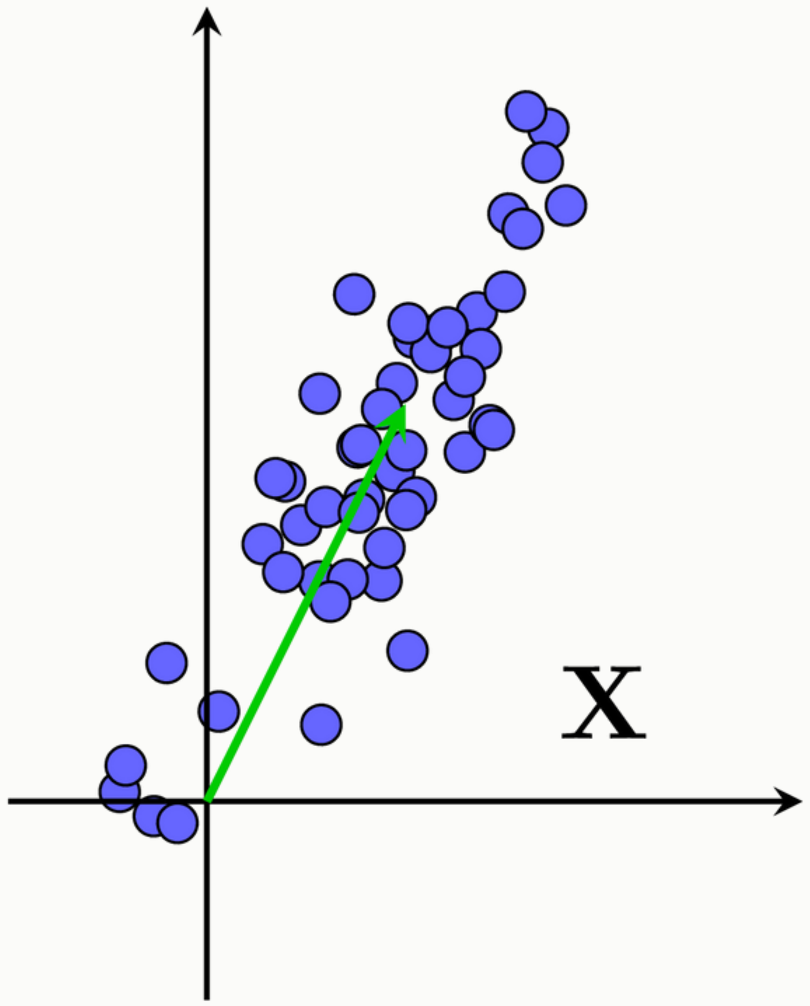


$$\mathbf{X} = \mathbf{U}_K\mathbf{Z} + \bar{\mathbf{U}}_K\mathbf{Y}$$

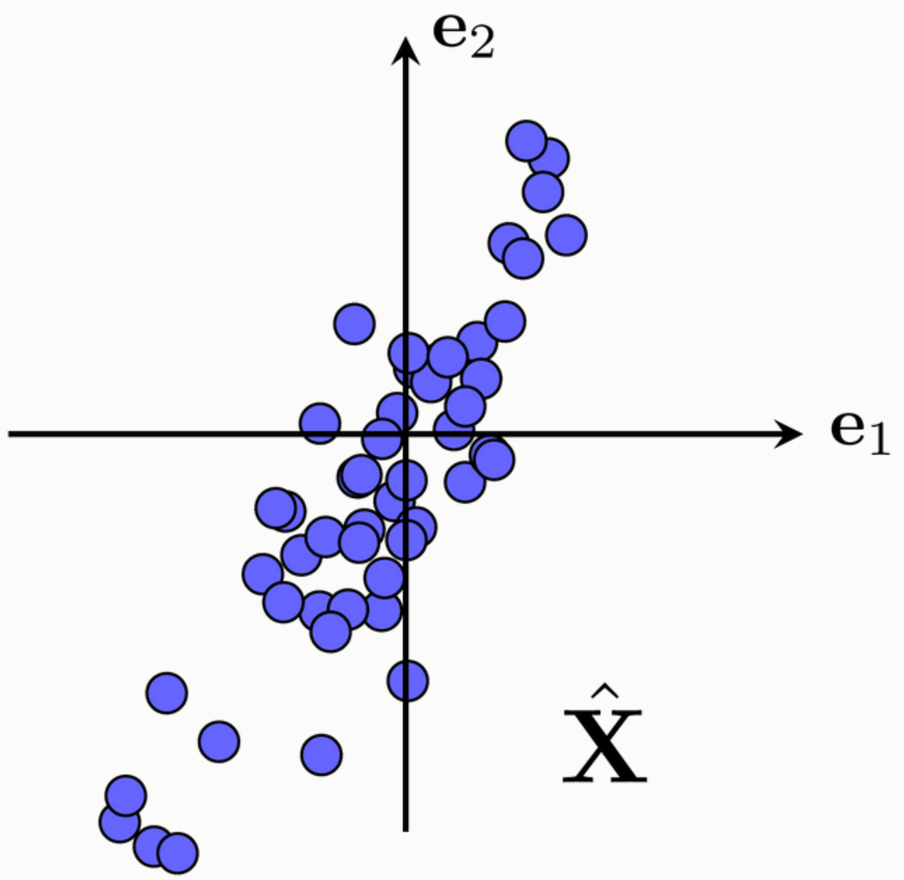
$$\mathbf{X} \approx \tilde{\mathbf{X}} = \mathbf{U}_K\mathbf{Z} + \bar{\mathbf{U}}_K\bar{\mathbf{U}}_K^T\bar{\mathbf{x}}\mathbf{1}^T$$

Procedure

1. Find mean vector



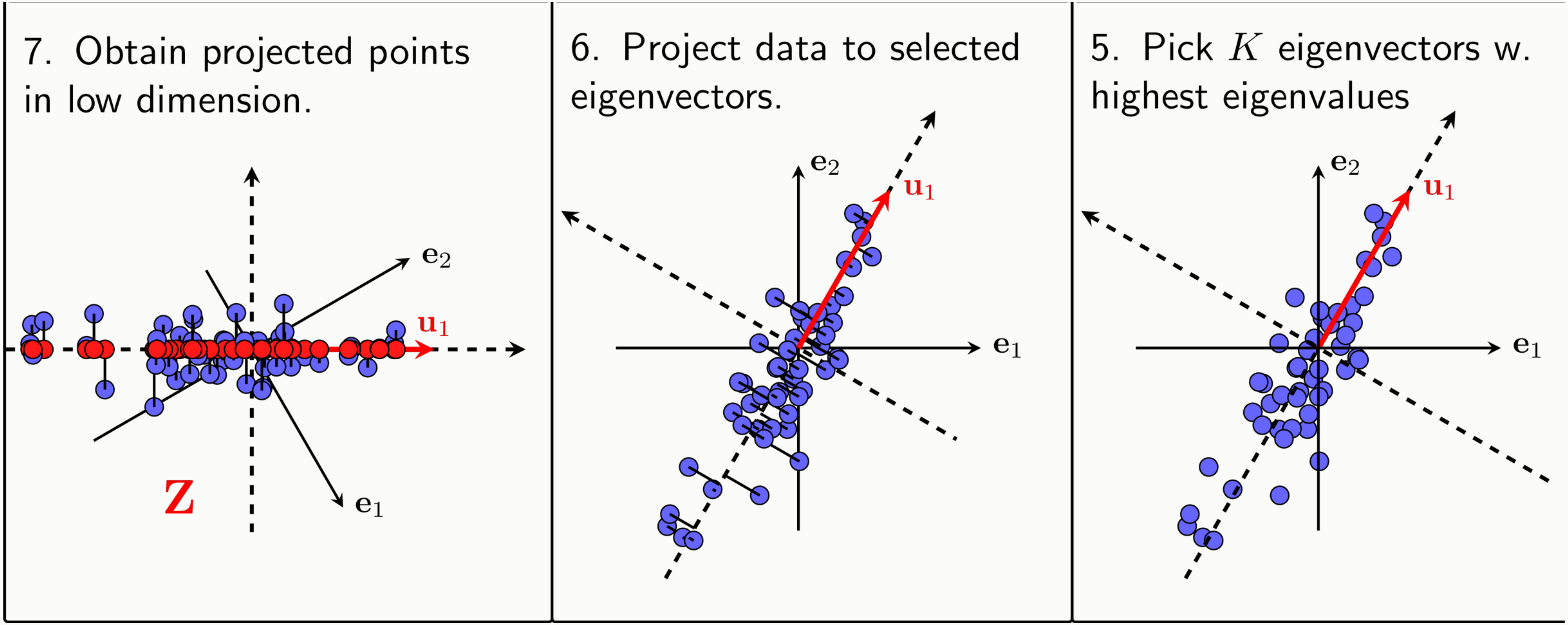
2. Subtract mean



3. Compute covariance matrix:
$\mathbf{S} = \frac{1}{N}\hat{\mathbf{X}}\hat{\mathbf{X}}^T$

4. Computer eigenvalues and eigenvectors of $\mathbf{S}$:
$(\lambda_1, \mathbf{u}_1), \ldots, (\lambda_D, \mathbf{u}_D)$
Remember the orthonormality of $\mathbf{u}_i$.

Procedure


7. Obtain projected points in low dimension.
$\mathbf{e}_2$
$\mathbf{u}_1$
$\mathbf{Z}$
$\mathbf{e}_1$
6. Project data to selected eigenvectors.
$\mathbf{e}_2$
$\mathbf{u}_1$
$\mathbf{e}_1$
5. Pick K eigenvectors w. highest eigenvalues
$\mathbf{e}_2$
$\mathbf{u}_1$
$\mathbf{e}_1$

2.4. PCA & SVD, any relationship?

PCA

- **PCA transforms data**

$$\mathbf{U}_K, \mathbf{Z} = \min_{\mathbf{U}_K, \mathbf{Z}} ||\mathbf{X} - \mathbf{U}_K \mathbf{Z}||_F$$

$$\text{s.t.:} \quad \mathbf{U}_K^T \mathbf{U}_K = \mathbf{I}_K$$

$$\mathbf{X} \approx \tilde{\mathbf{X}} = \mathbf{U}_K \mathbf{Z}$$

SVD

- **SVD: Decompose matrix**

$$\min_{\mathbf{A}} ||\mathbf{X} - \mathbf{A}||_F$$

$$\text{s.t. } \operatorname{rank}(\mathbf{A}) = K$$

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

2.5. PCA - Summarise

PCA → new basis system

PCA → covariance matrix

PCA → truncated SVD

Pros:

- Prevent overfitting
- Improve visualization
- Improve performance
- ...

Cons:

- Information Loss
- Scaling data
- Same variance
- ...

3. Implement in Python

THAT'S IT