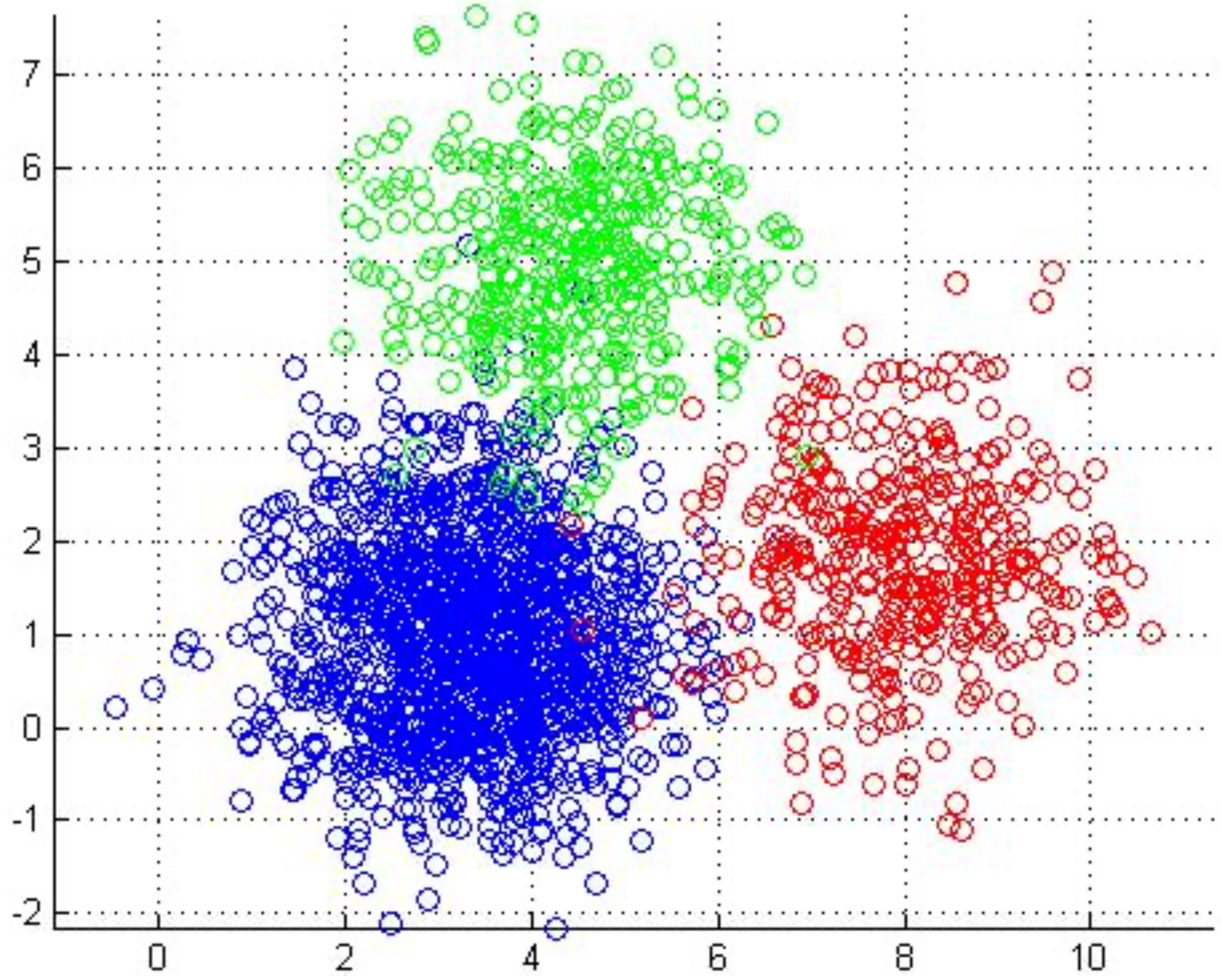
Dimensionality Reduction

EECE454 Intro. to Machine Learning Systems

Fall 2024

Recap

- **Unsupervised learning.** Discovering **useful stuctures** of the data, using the unlabeled dataset
    - K-Means Clustering
    - Gaussian Mixture Models
    - Dimensionality Reduction <— This week
    - Neural-net-based
        - Autoencoders
        - GANs
        - Diffusion models
        - Language Models



Dealing with high-dimensional data

- Many datasets are extremely high-dimensional in its raw form
- Suppose that you are an **ML engineer** at Google
    - Then, you'd need to learn from these datasets:

YouTube Shorts
1920 x 1080 x 3 colors x 60 fps x 60 seconds
= 22.4 billion pixels (per video)

Dealing with high-dimensional data

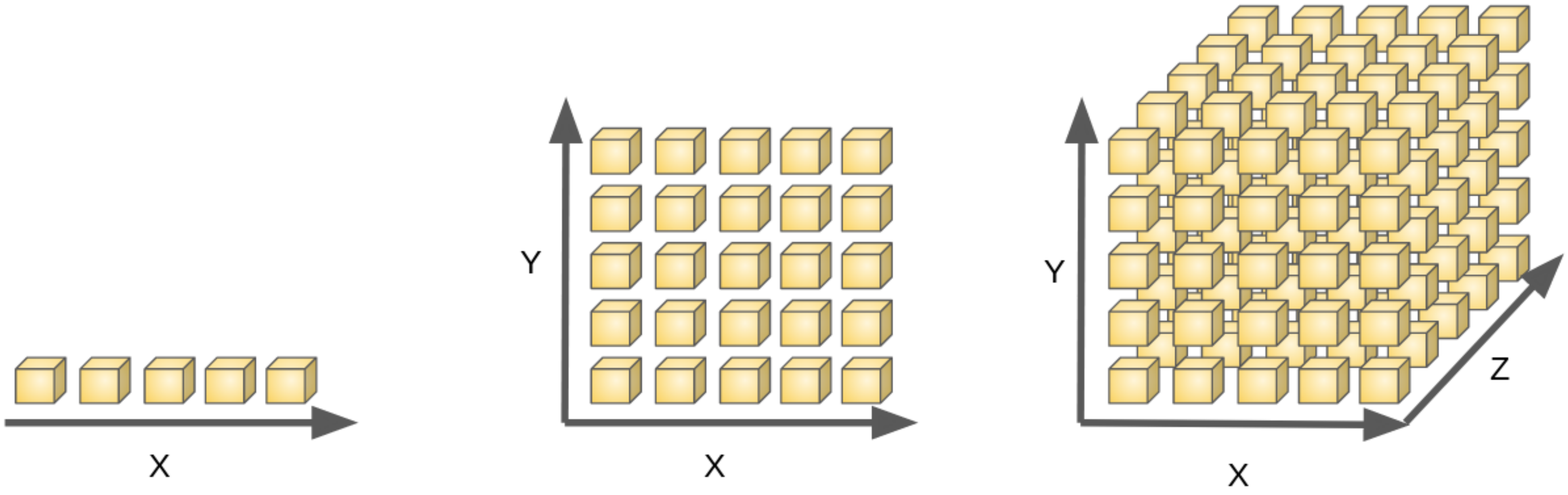
- Many datasets are extremely high-dimensional in its raw form
- Suppose that you are an **ML engineer** at Google
  - Then, you'd need to learn from these datasets:

YouTube Shorts
1920 x 1080 x 3 colors x 60 fps x 60 seconds
= 22.4 billion pixels (per video)

Gmail
1000s of words x sender info x receiver info x (images…)
= millions~billions real numbers (per mail)

Curse of dimensionality

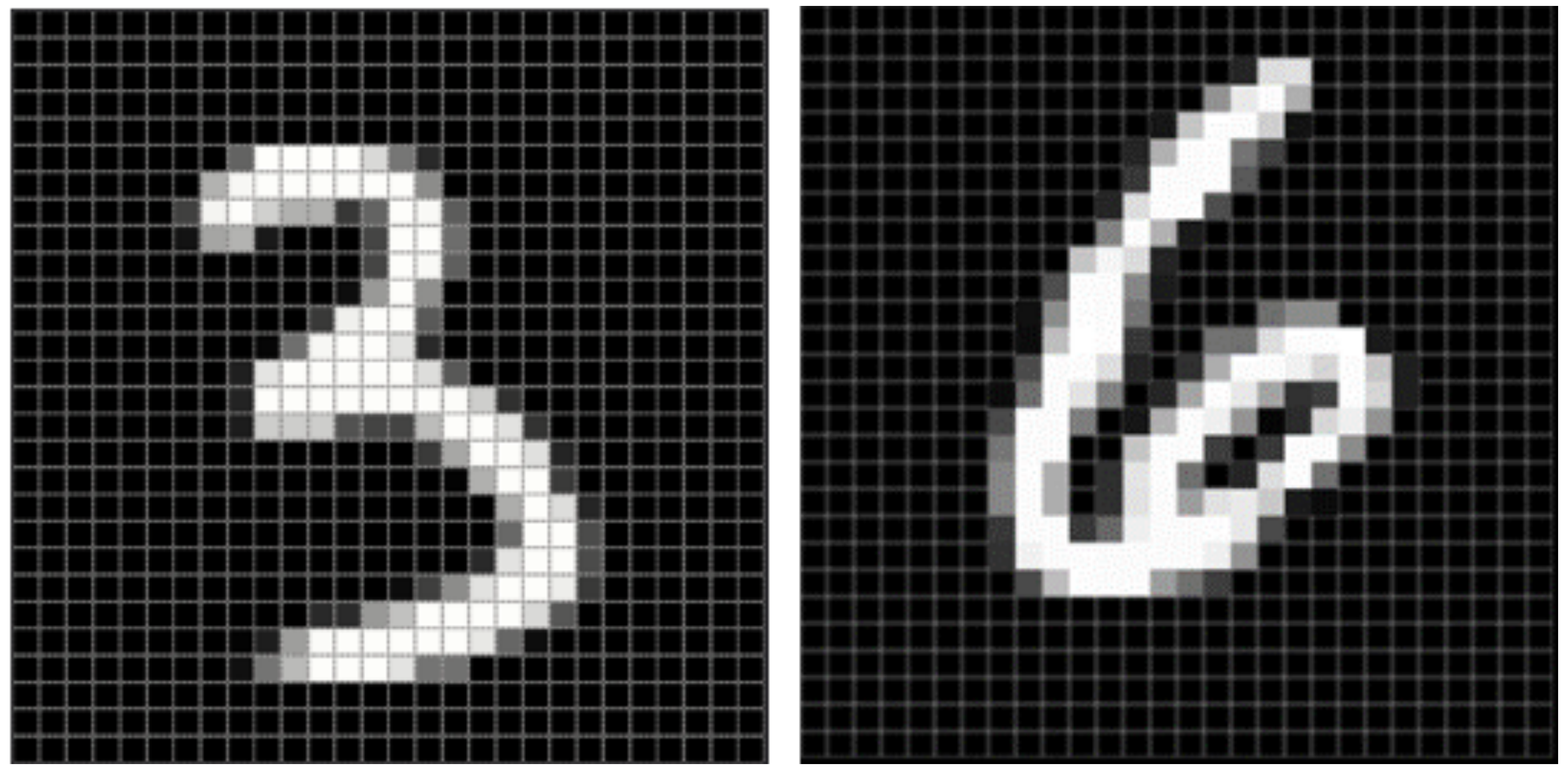
- Higher-dimensional data are nasty to do ML on
  - More computation
  - Higher chance of noise
  - Difficult to visualize (for human insight)
  - Difficult to find meaningful patterns



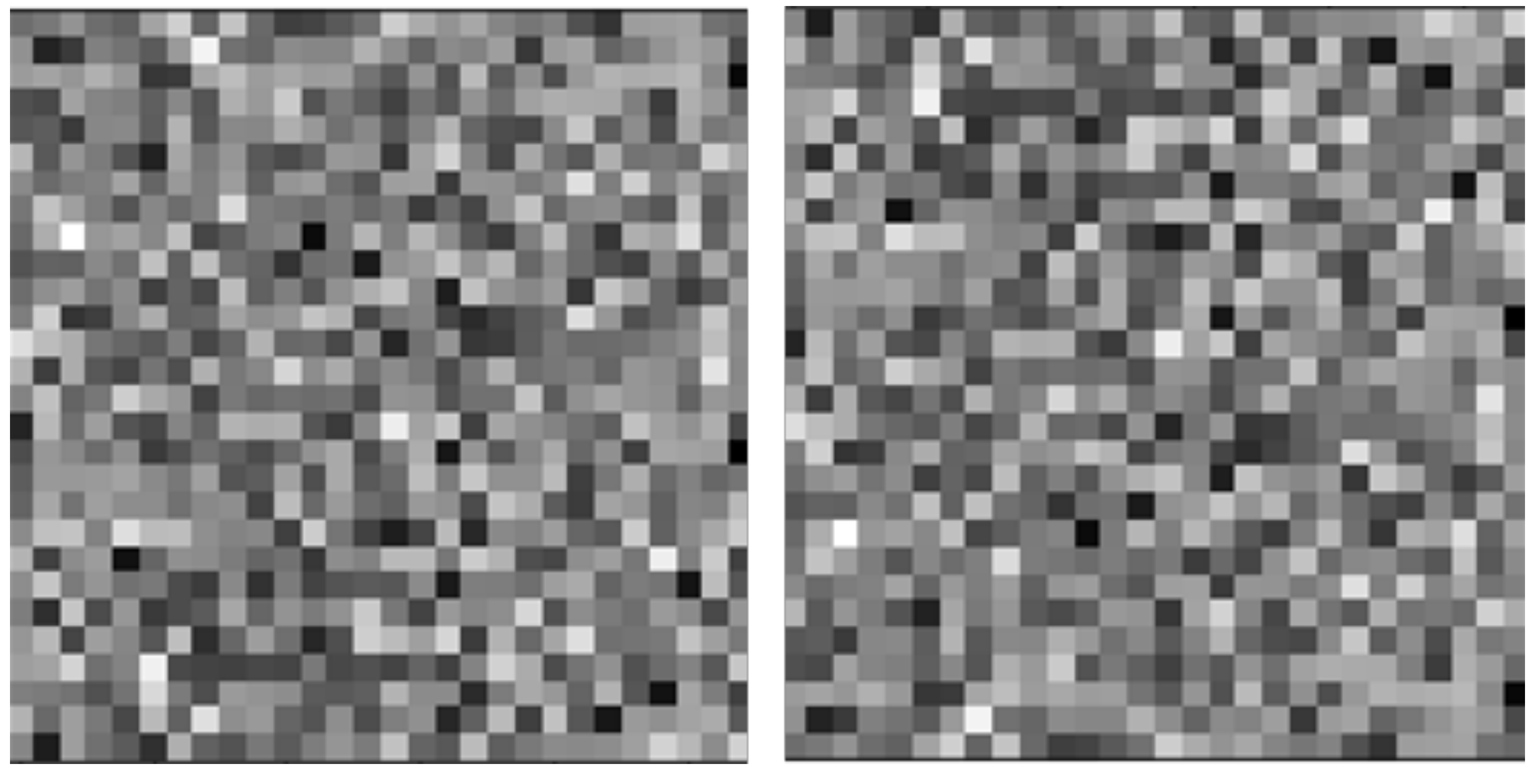
Dimensionality: Nominal vs. True

- But do we really need all dimensions?

  - Example. Handwritten digit recognition (MNIST, 28x28 image)



only looks like this

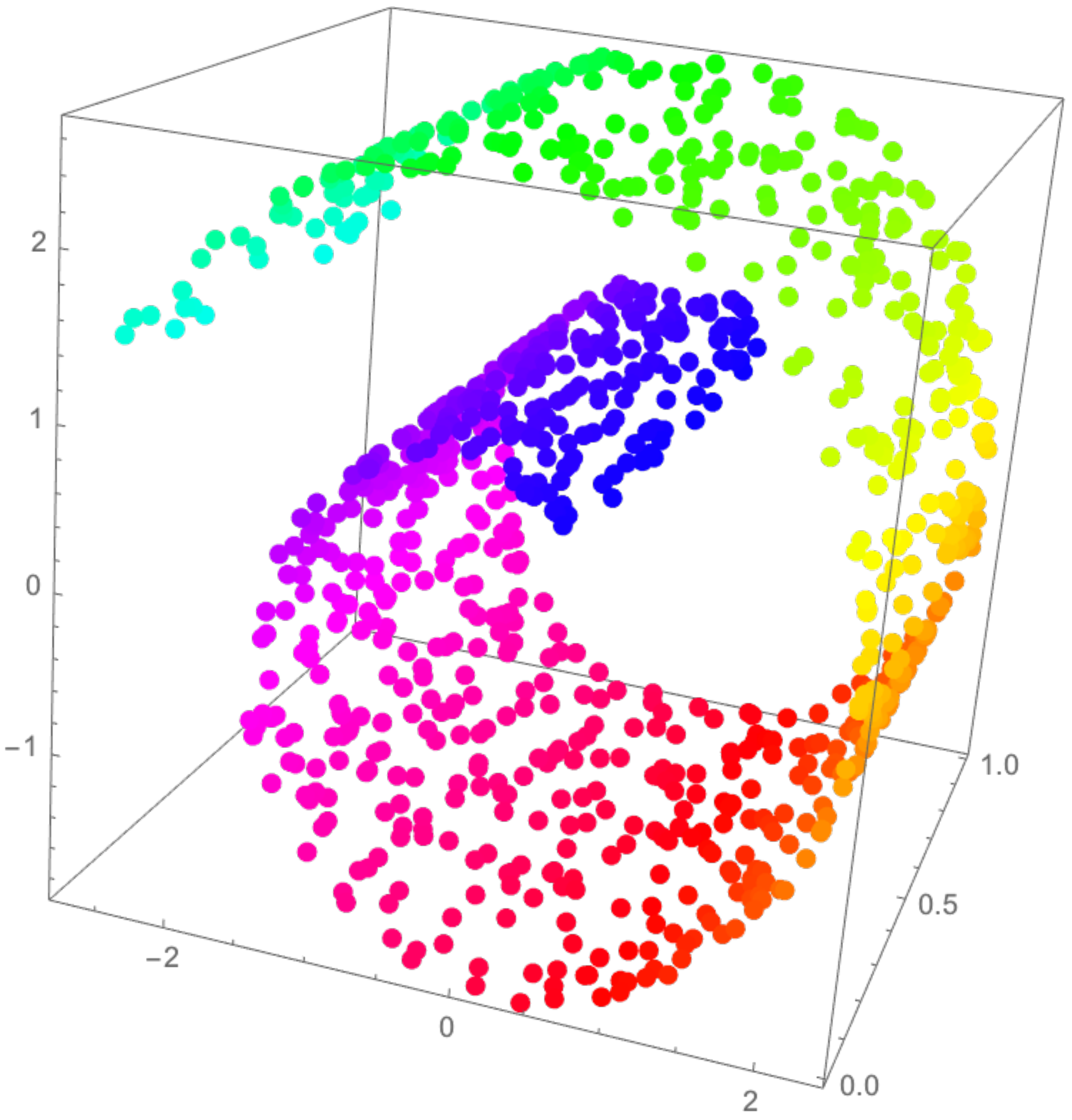


... and not like this

  - Thus, we may not need to fully utilize $\mathbb{R}^{28\times 28} = \mathbb{R}^{784}$

Dimensionality: Nominal vs. True

- **Hypothesis.**
  There exists some low-dim subspace (or submanifold) in the high-dimensional feature space, where the real data lies in

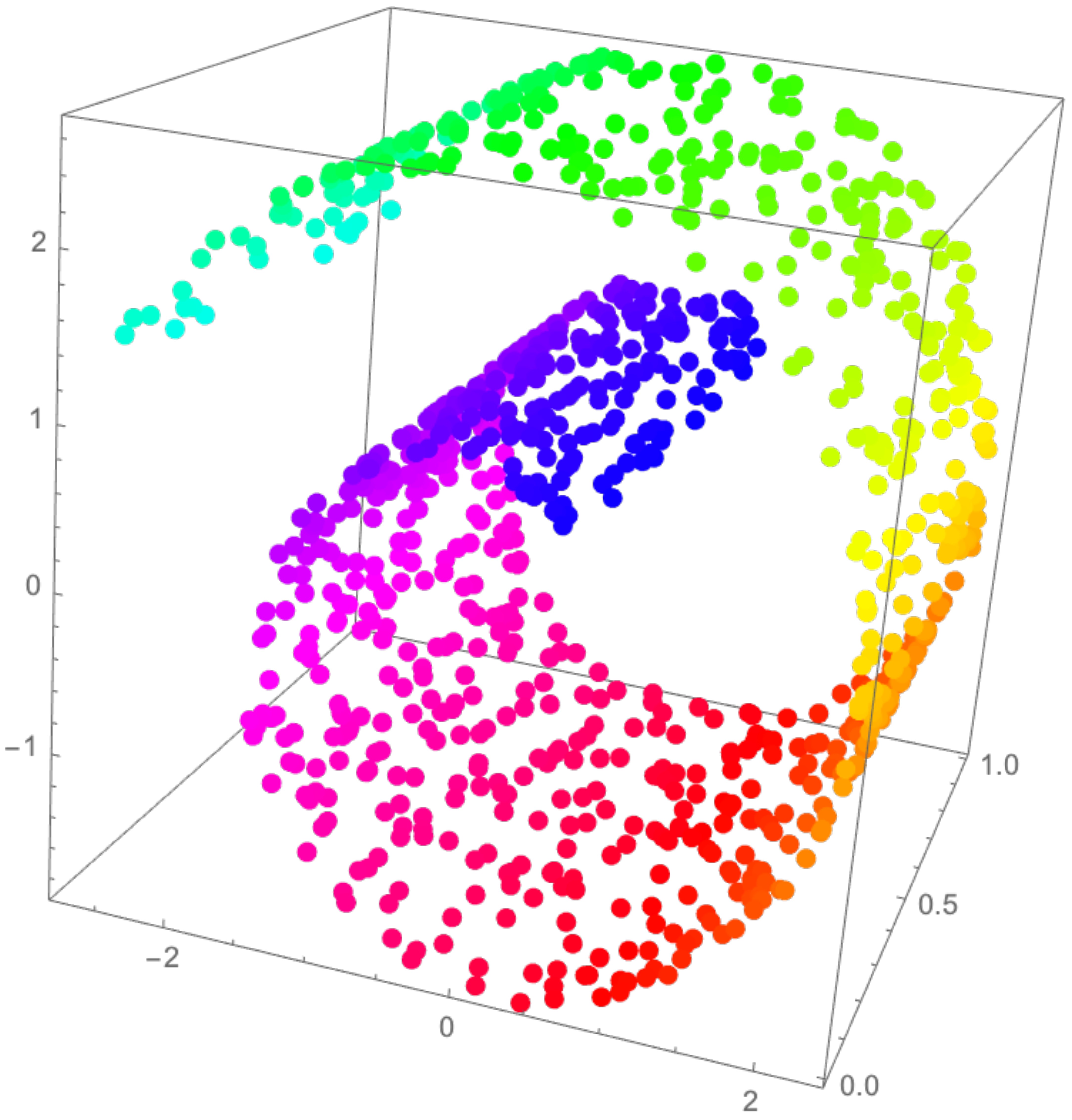


Dimensionality: Nominal vs. True

- **Hypothesis.**
  There exists some low-dim subspace (or submanifold)
  in the high-dimensional feature space,
  where the real data lies in

- **Dimensionality reduction**
  Use unlabeled data to find the right mapping
  from a high-dimensional to low-dimensional space

- Caveat. There could be some noises in each datum,
  which can make things tricky



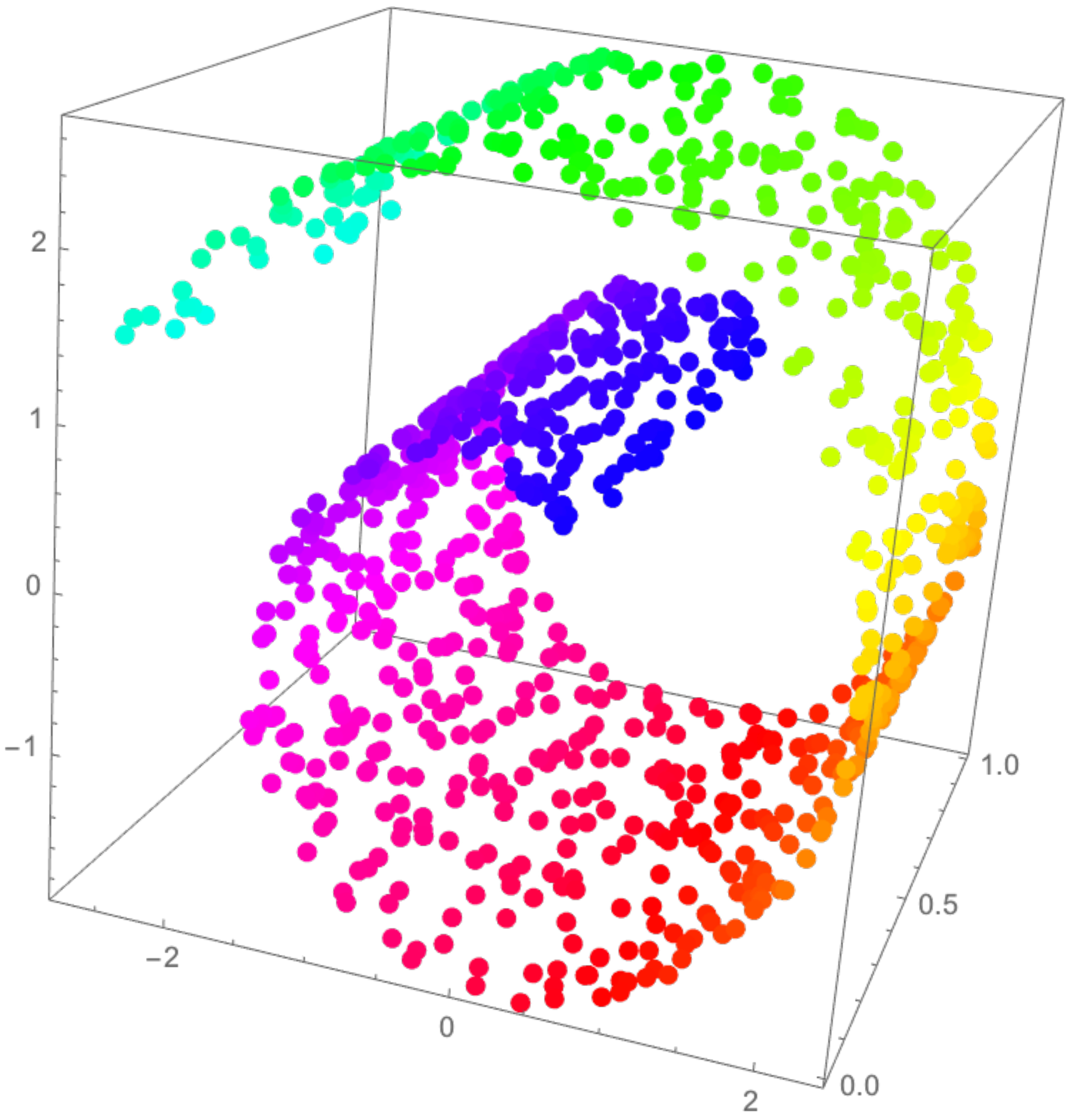
Dimensionality: Nominal vs. True

- **Hypothesis.**
  There exists some low-dim subspace (or submanifold)
  in the high-dimensional feature space,
  where the real data lies in

- **Dimensionality reduction**
  Use unlabeled data to find the right mapping
  from a high-dimensional to low-dimensional space

- Caveat. There could be some noises in each datum,
  which can make things tricky

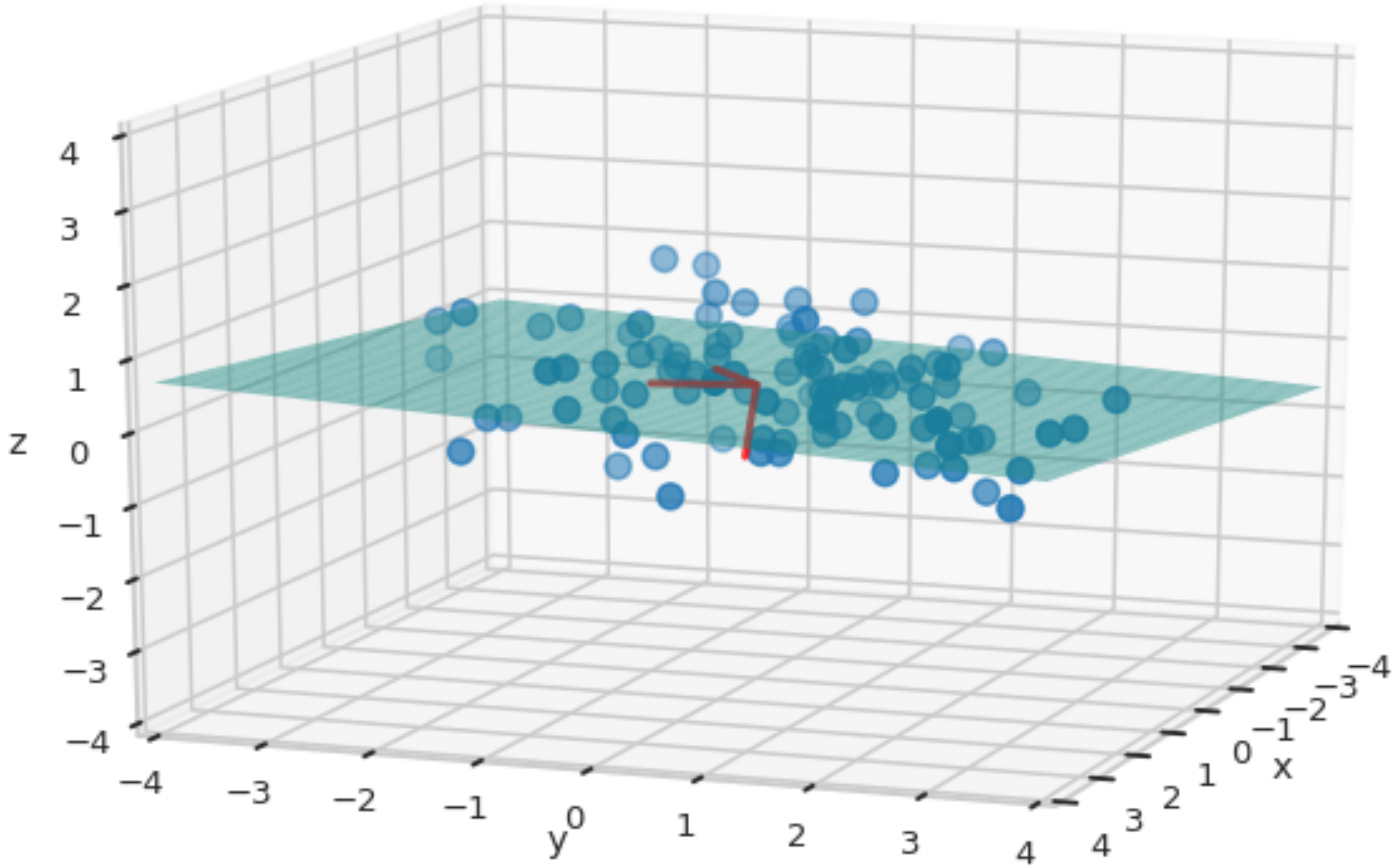
- **Today.** Look at a linear case, called PCA.



Principal component analysis

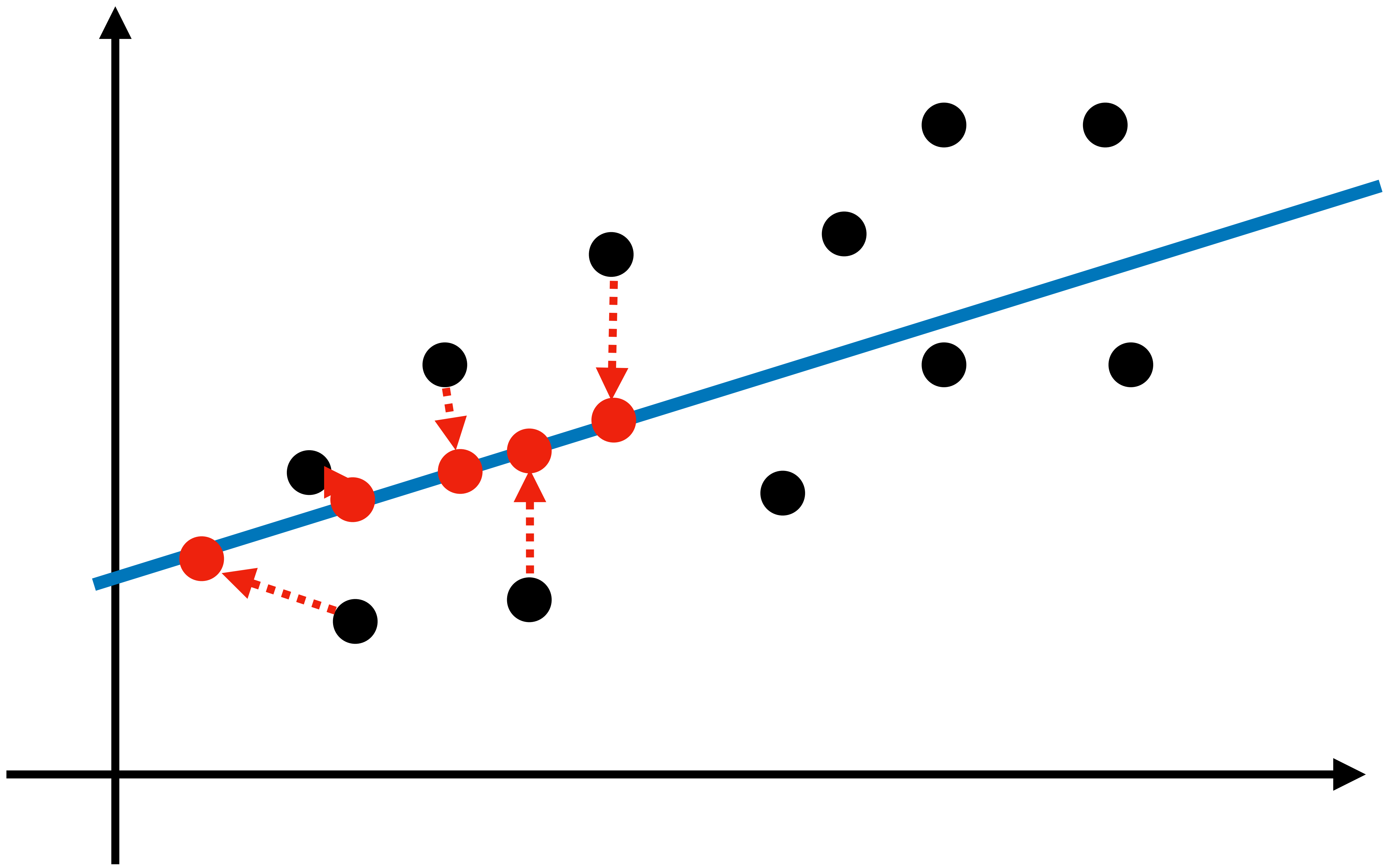
Overview

- Dimensionality reduction, using an **affine subspace** of the original space
  - Invented by Karl Pearson (1909)
  - Many aliases, e.g., Karhunen-Loève Transform



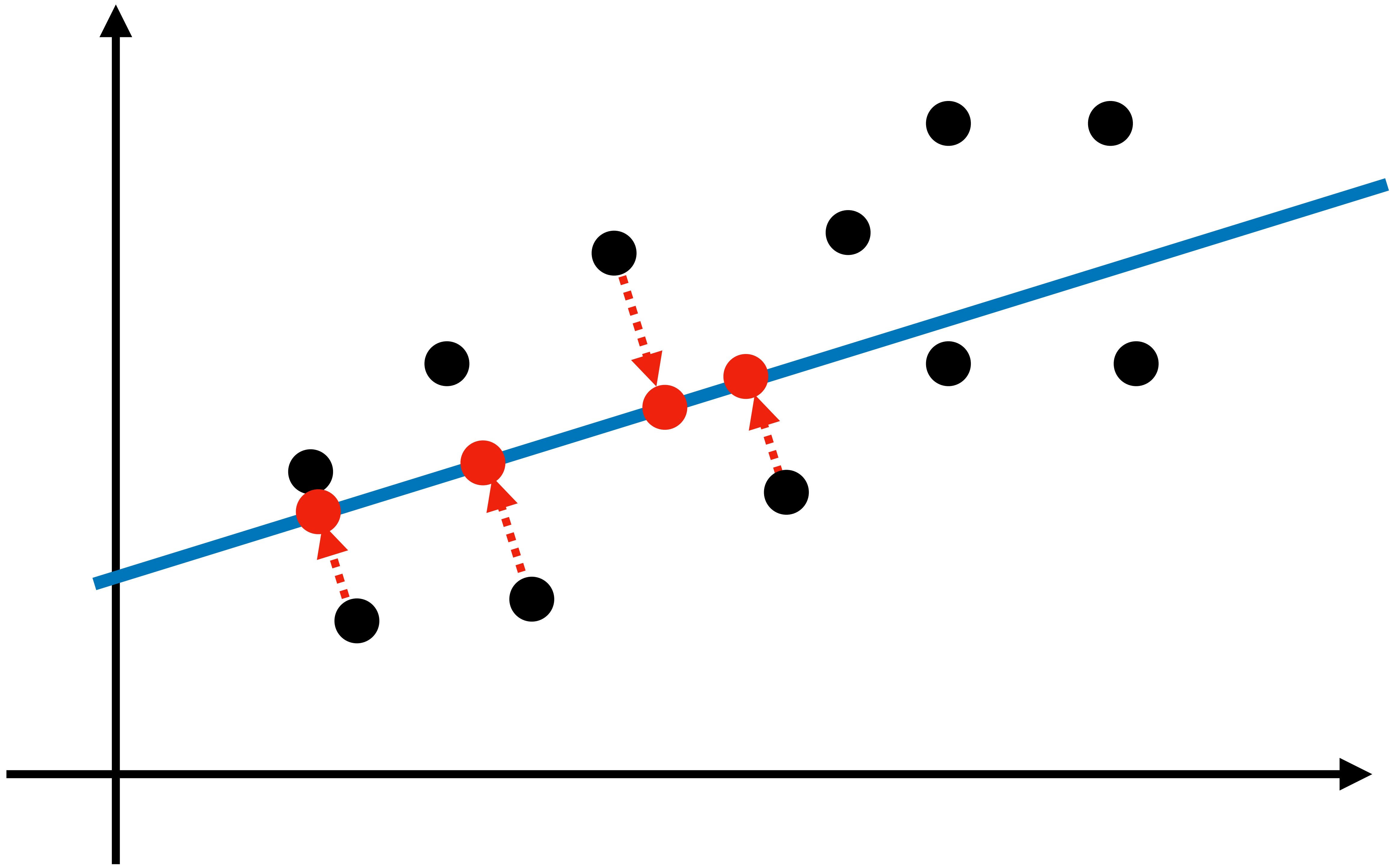
Motivational example

- Suppose that we are given a 2D dataset.
- **Goal.** Find a nice **1D subspace** and a **mapping**, s. t. the mapped data has **nice properties**



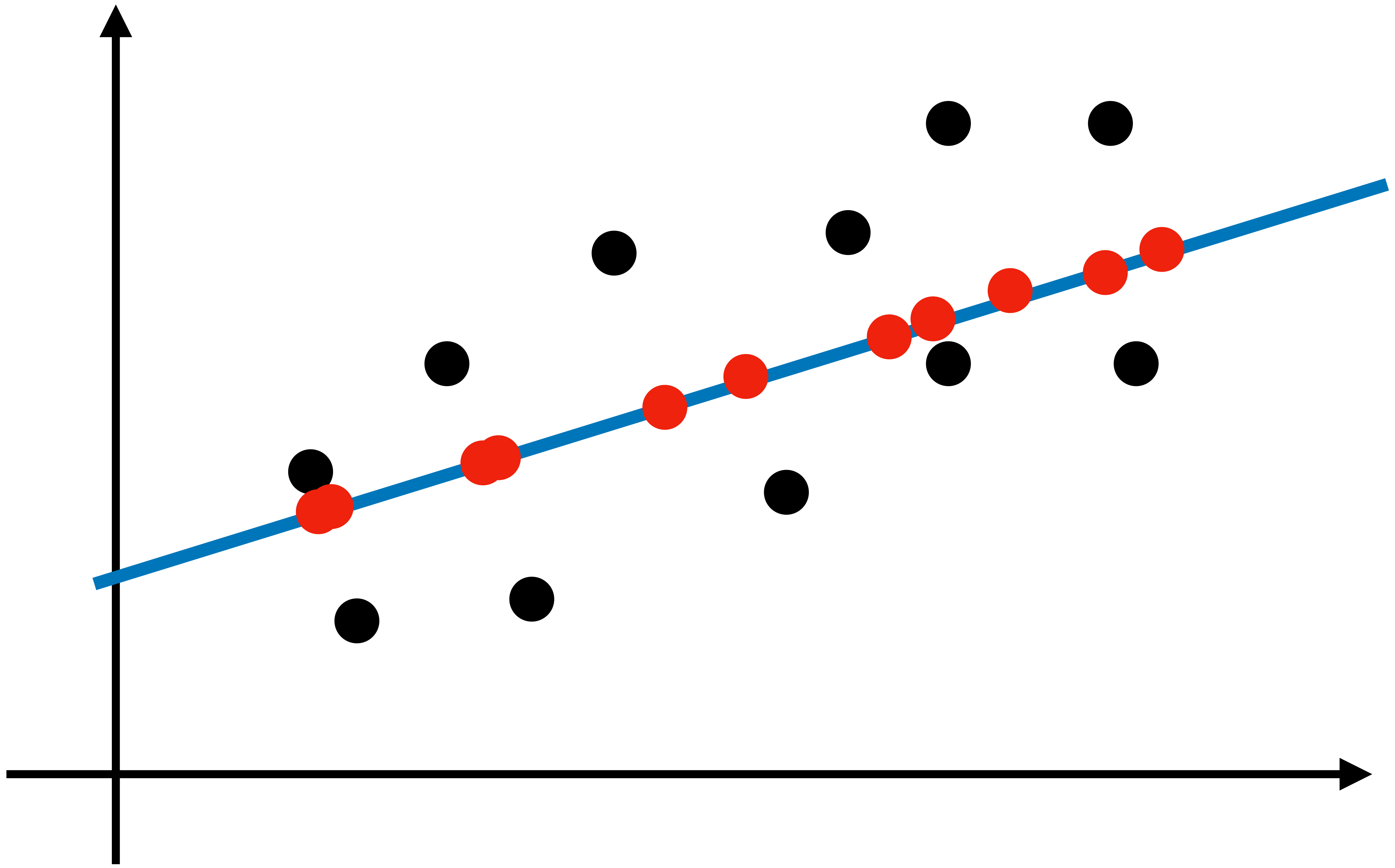
Motivational example

- Suppose that we are given a 2D dataset.
- **Goal.** Find a nice **1D subspace** and a **mapping**, s. t. the mapped data has **nice properties**
    - Confine the mapping to be an **orthogonal projection** —> Only about determining **subspaces**



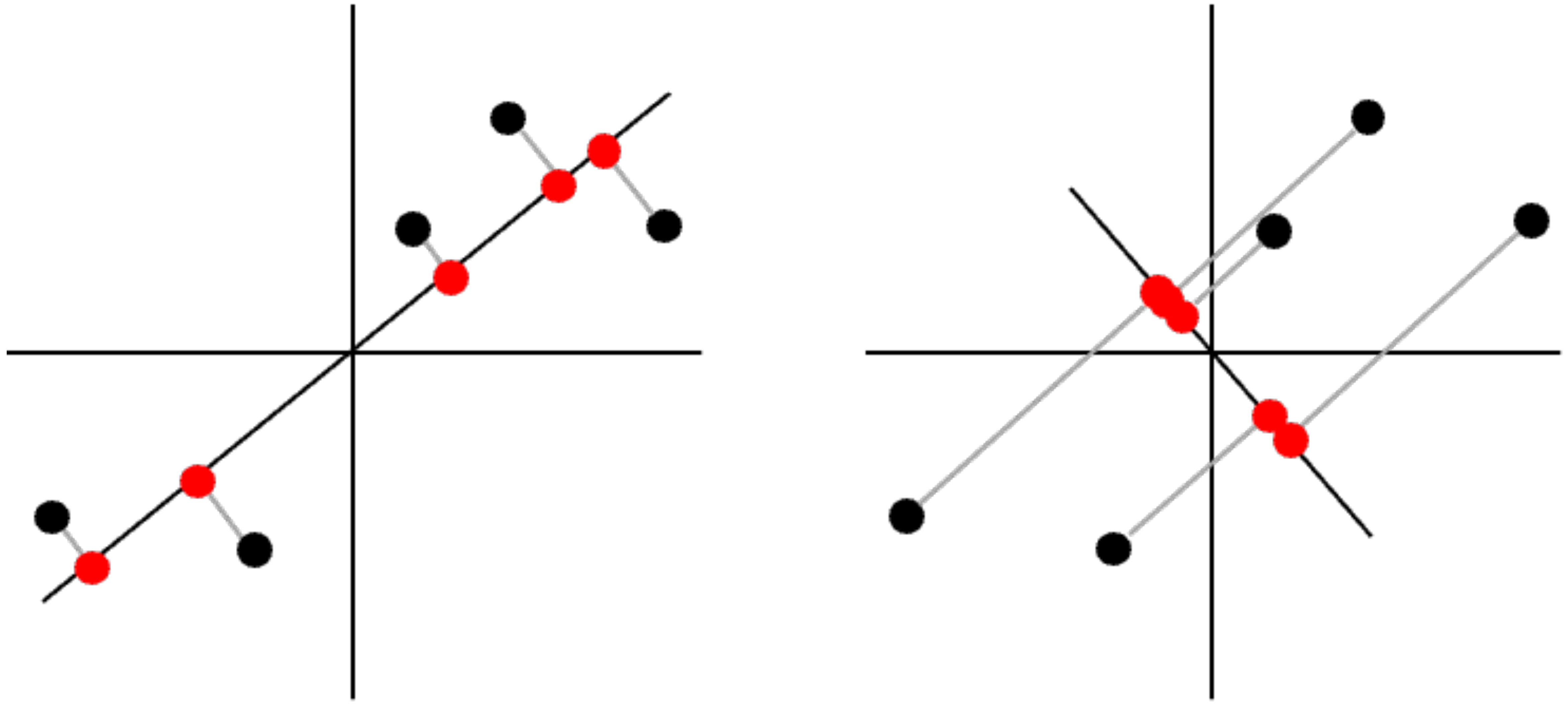
Motivational example

- Suppose that we are given a 2D dataset.
- **Goal.** Find a nice **1D subspace** and a **mapping**, s. t. the mapped data has **nice properties**
  - Confine the mapping to be an **orthogonal projection** —> Only about determining **subspaces**
- **Goal (restated).** Find a nice **1D subspace** that the projected data has **nice properties**



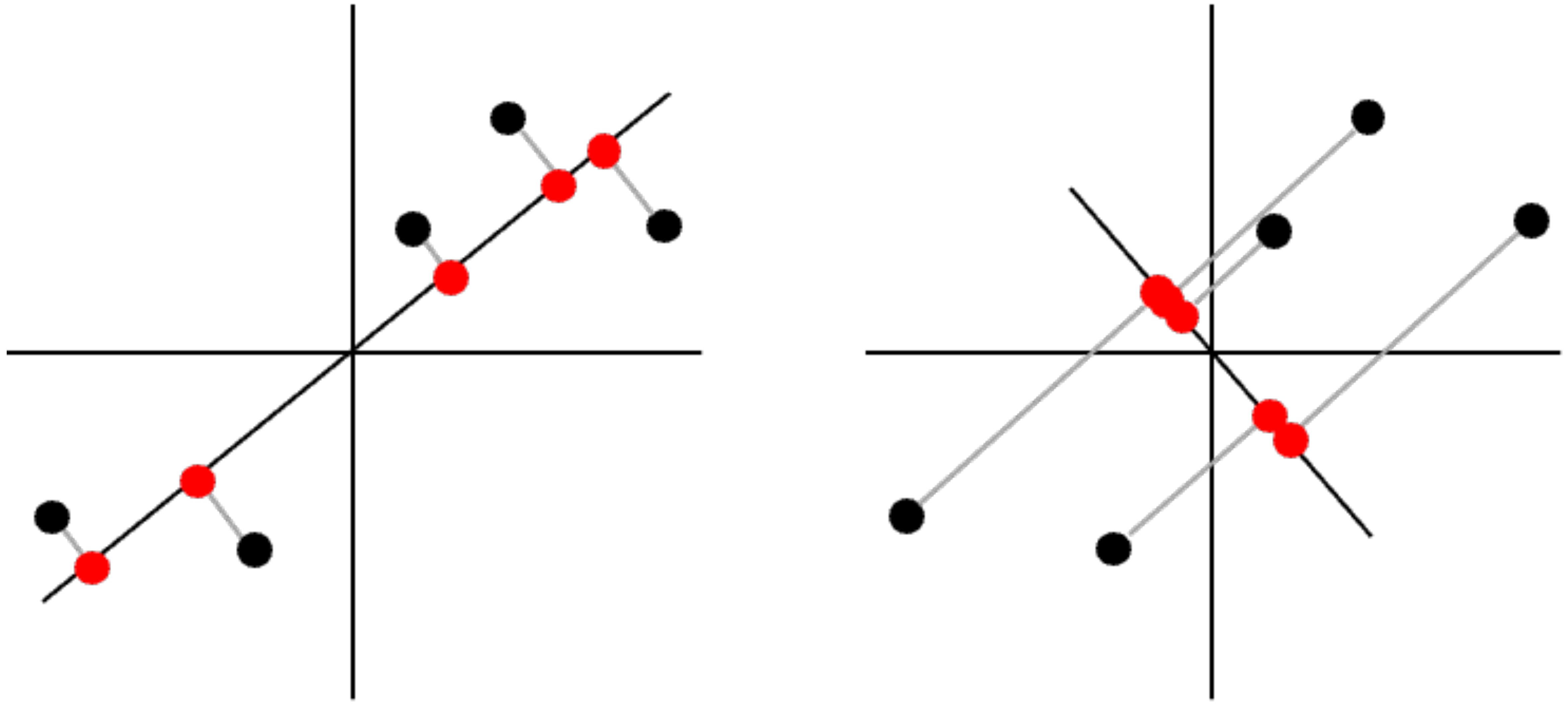
Motivational example

- **Idea.** Suppose that we want to preserve some information as much as possible
  - Question. Which projection contains more information?



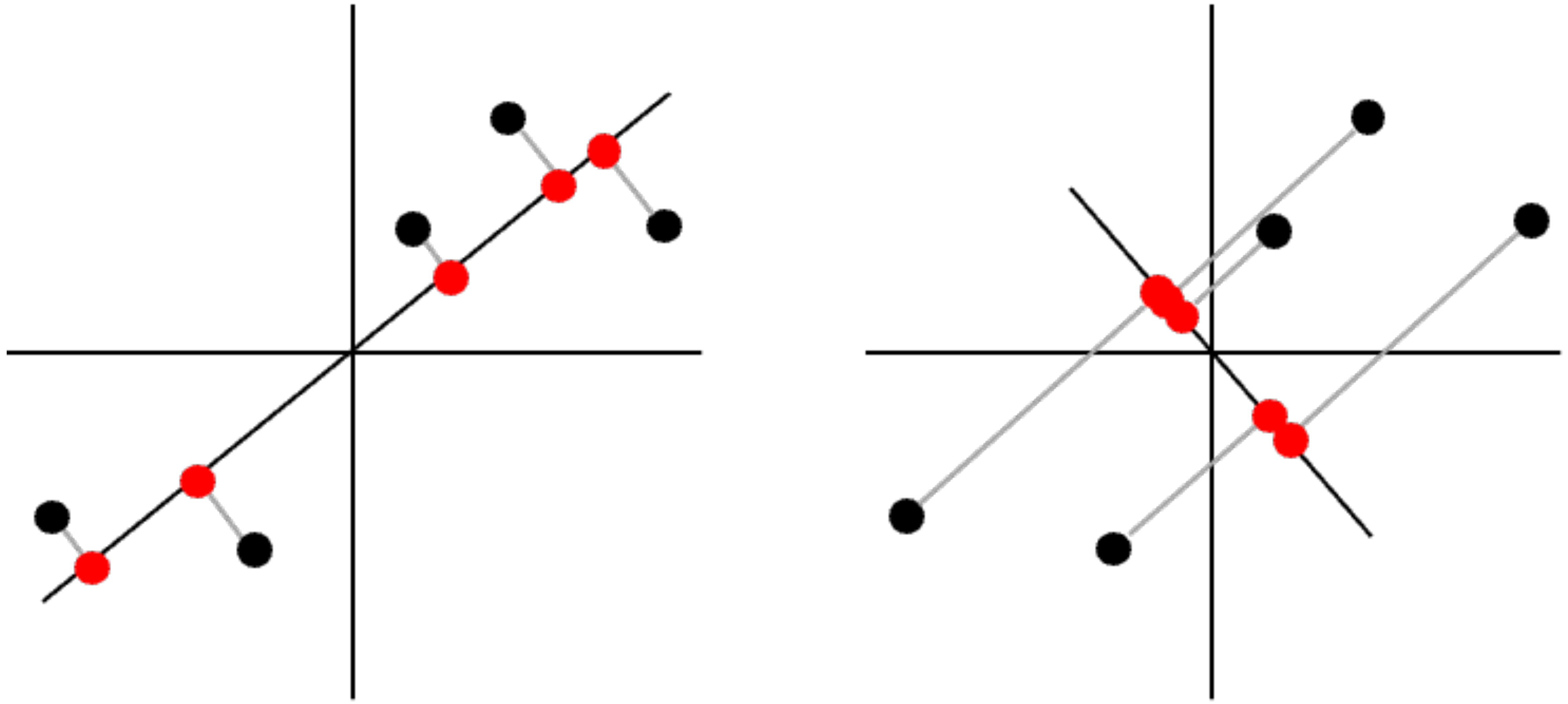
Motivational example

- **Idea.** Suppose that we want to preserve some information as much as possible
  - Question. Which projection contains more information?
  - Answer. Left, for two reasons.
    - (A) Projected points are more spread, thus not ignoring the differences between points
    - (B) Original points (●) are closer to their projections (●)



Motivational example

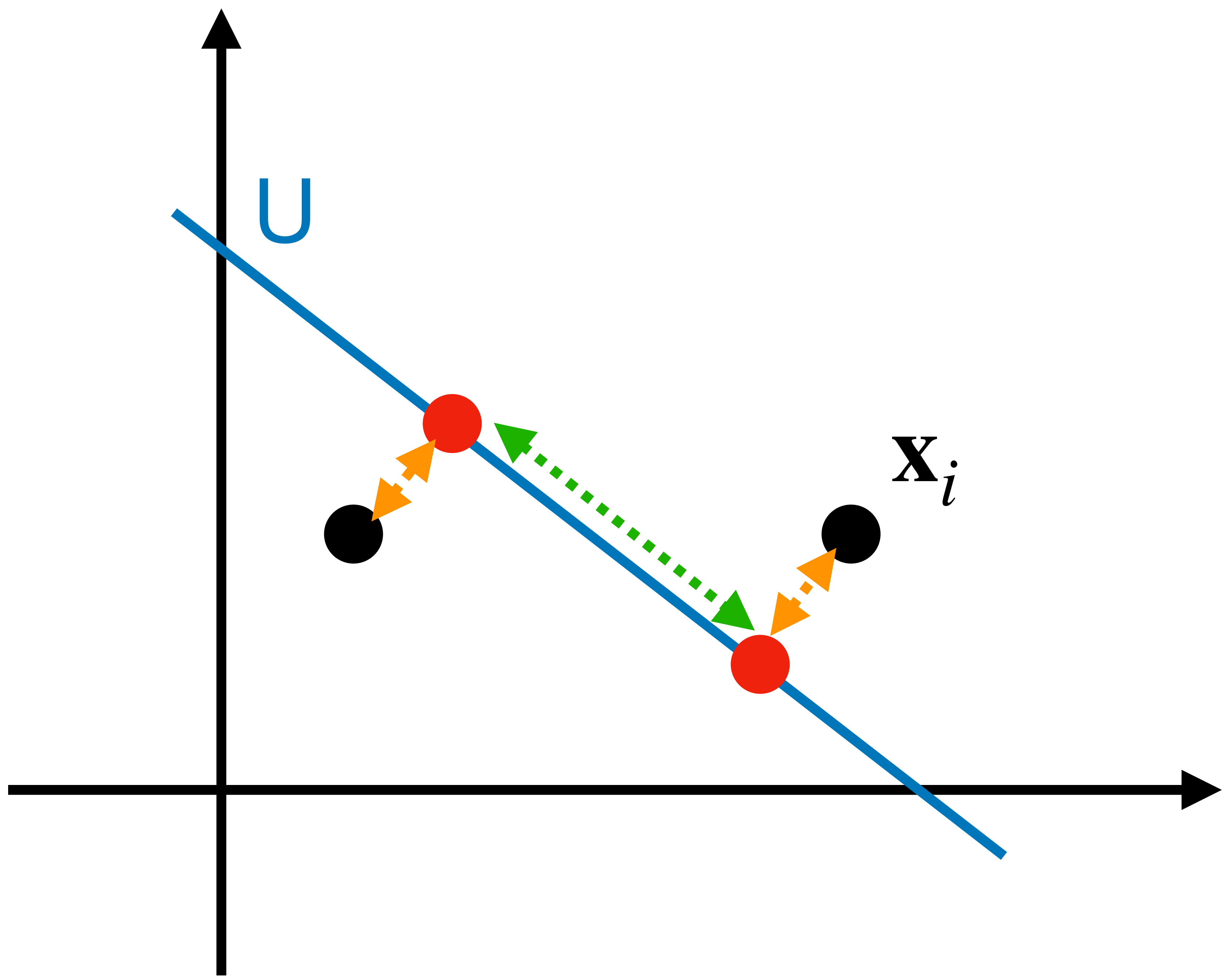
- **Idea.** Suppose that we want to preserve some information as much as possible
  - Question. Which projection contains more information?
  - Answer. Left, for two reasons.
    - (A) Projected points are more spread, thus not ignoring the differences between points
    - (B) Original points (●) are closer to their projections (●)
      - Note. We will see later that (A) is equivalent to (B)!



Principal Component Analysis

- Let us be a little more formal:
  - Suppose that we have a dataset $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^d$
  - Goal. Find a k-dimensional subspace U of $\mathbb{R}^d$ such that:
    - (A) The projection has the **maximum variance**:

$$\max_{\mathsf{U}} \mathrm{Var}(\pi_{\mathsf{U}}(\mathbf{x}_1), \ldots, \pi_{\mathsf{U}}(\mathbf{x}_n))$$



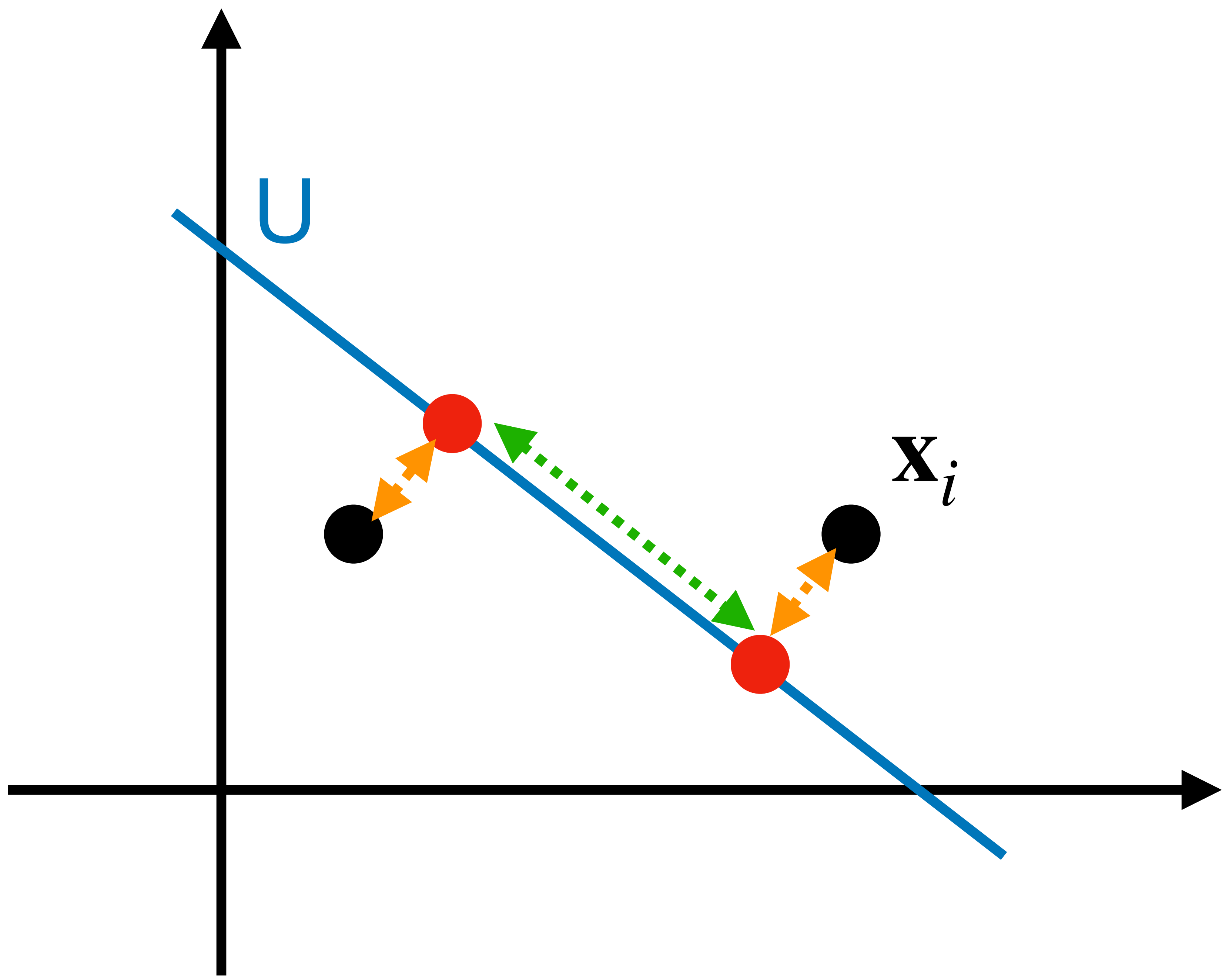
Principal Component Analysis

- Let us be a little more formal:
  - Suppose that we have a dataset $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^d$
  - Goal. Find a k-dimensional subspace U of $\mathbb{R}^d$ such that:
    - (A) The projection has the **maximum variance**:

$$\max_{\mathsf{U}} \mathrm{Var}(\pi_{\mathsf{U}}(\mathbf{x}_1), \ldots, \pi_{\mathsf{U}}(\mathbf{x}_n))$$

    - (B) The **distortion** from projection is **minimized**:

$$\min_{\mathsf{U}} \sum_{i=1}^{n} \|\mathbf{x}_i - \pi_{\mathsf{U}}(\mathbf{x}_i)\|_2^2$$

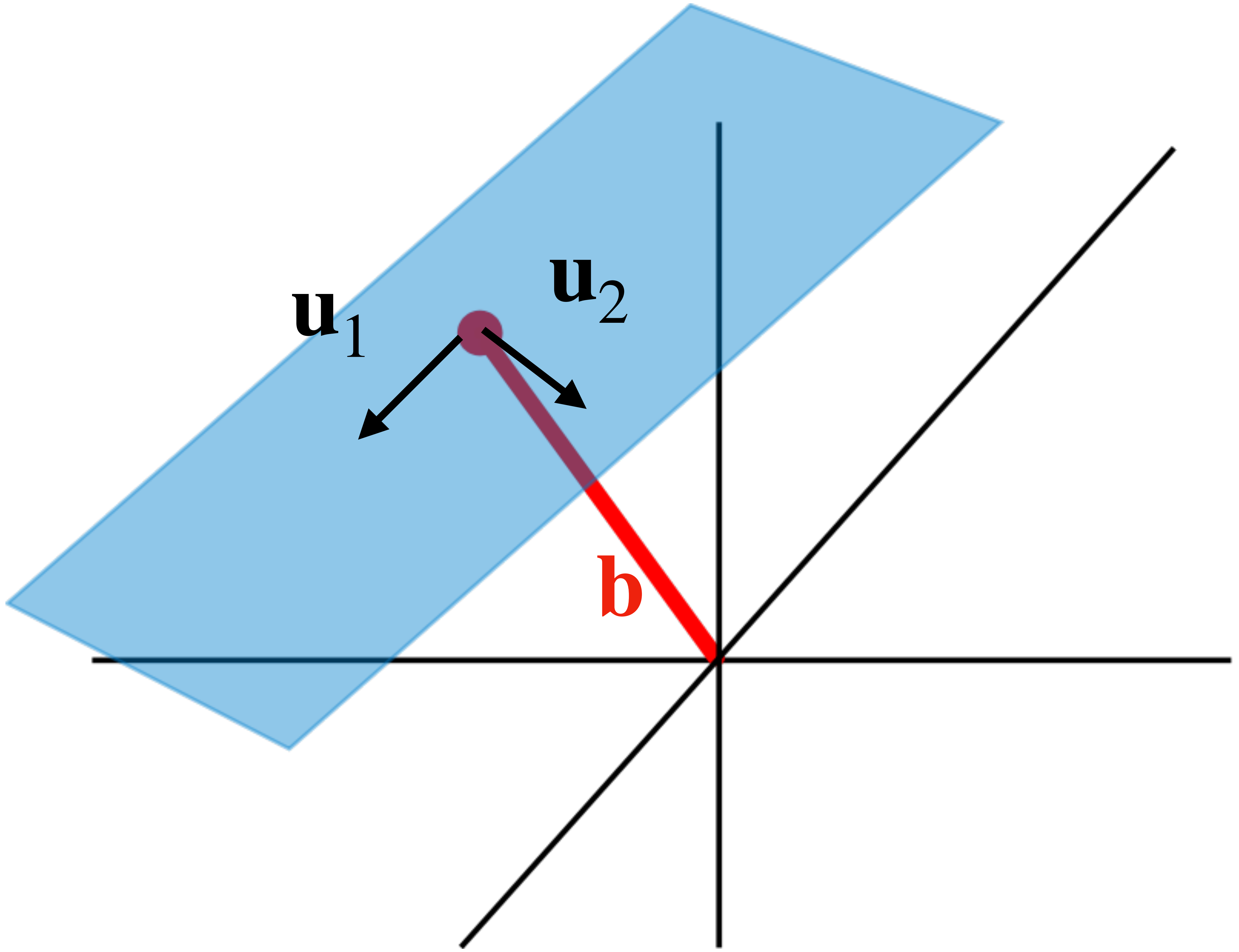


PCA: Formalisms

Formalism

- A **k-dimensional affine subspace** $\mathsf{U} \subset \mathbb{R}^d$ can be characterized by:
  - its orthonormal bases $\mathbf{u}_1, \ldots, \mathbf{u}_k \in \mathbb{R}^d$
  - an orthogonal bias $\mathbf{b} \in \mathbb{R}^d$

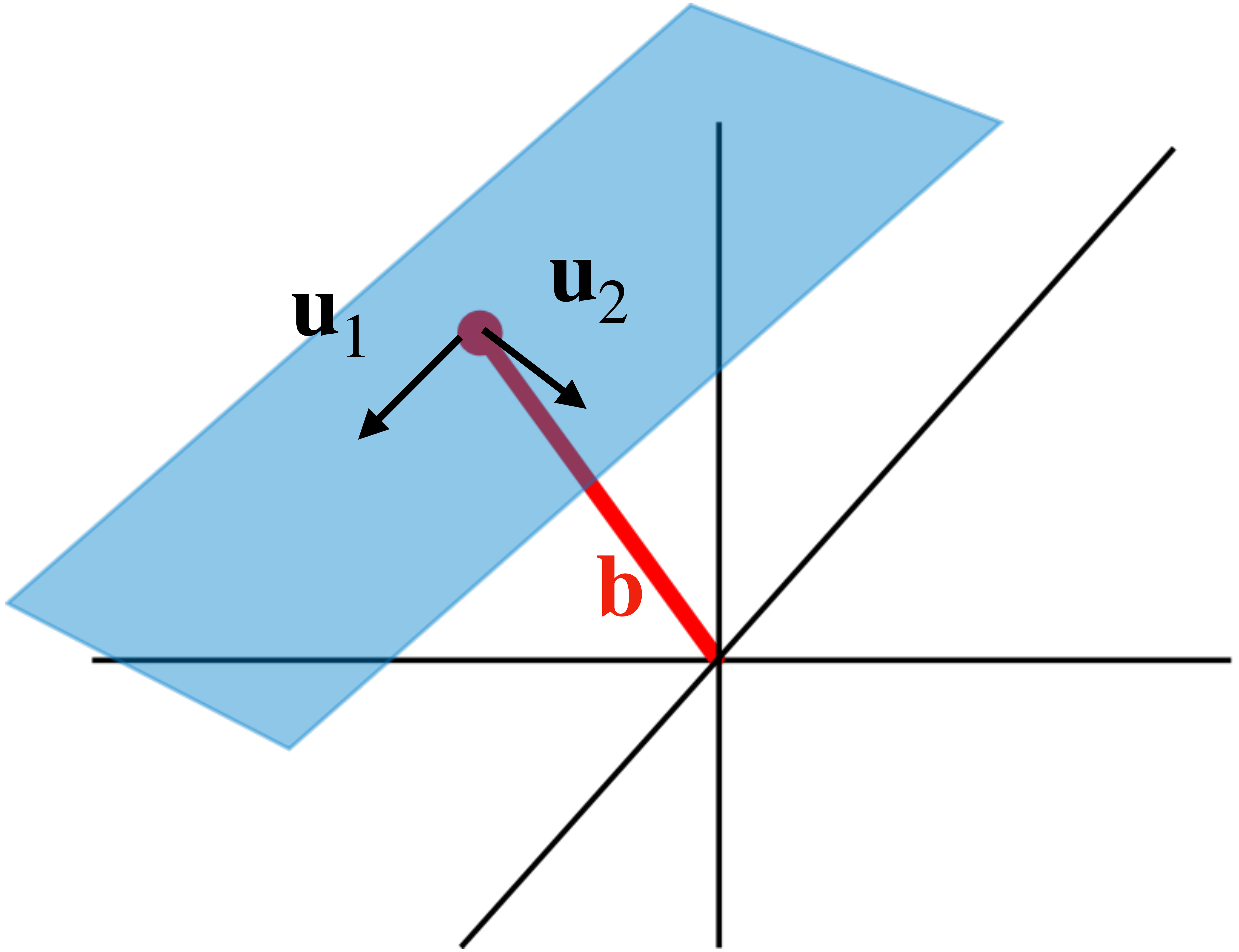
$$\mathsf{U} = \{a_1\mathbf{u}_1 + \cdots + a_k\mathbf{u}_k + \mathbf{b} \ : \ a_i \in \mathbb{R}\}$$



Formalism

- A k-**dimensional affine subspace** $\mathsf{U} \subset \mathbb{R}^d$ can be characterized by:
    - its orthonormal bases $\mathbf{u}_1, \ldots, \mathbf{u}_k \in \mathbb{R}^d$
    - an orthogonal bias $\mathbf{b} \in \mathbb{R}^d$

$$\mathsf{U} = \{a_1\mathbf{u}_1 + \cdots + a_k\mathbf{u}_k + \mathbf{b} \;:\; a_i \in \mathbb{R}\}$$



- Any element can be represented as:
    - a d-dimensional vector $\mathbf{u} \in \mathsf{U}$
    - a k-dimensional quantity

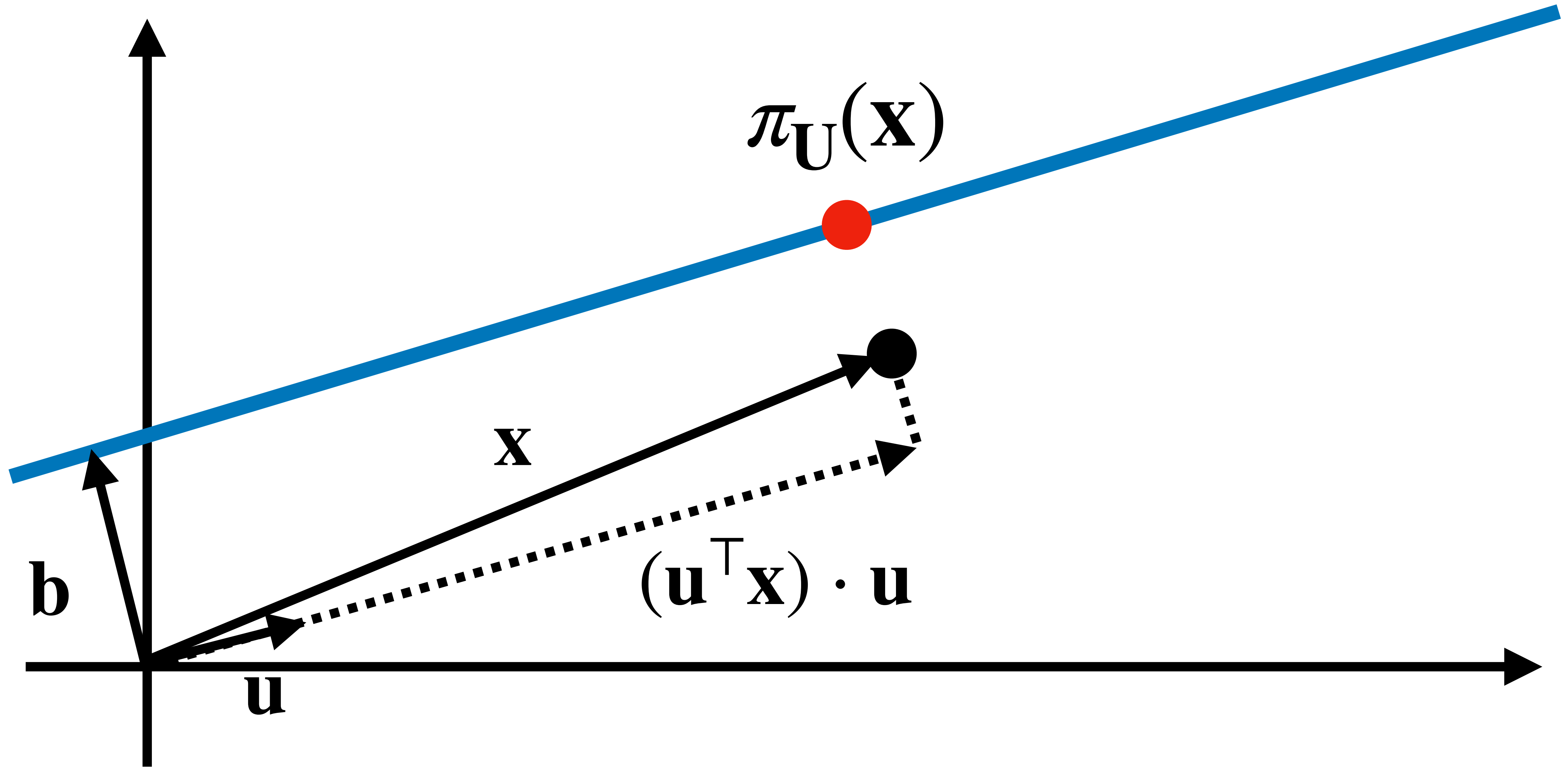
$$(a_1, a_2, \ldots, a_k)$$

Formalism

- A **projection** of a vector $\mathbf{x} \in \mathbb{R}^d$ to the affine subspace U is

$$\pi_{\mathsf{U}}(\mathbf{x}) = \sum_{i=1}^{k} (\mathbf{u}_i^\top \mathbf{x}) \cdot \mathbf{u}_i + \mathbf{b}$$

  - d-dimensional, with an alternative representation $\mathbf{a} = (\mathbf{u}_1^\top \mathbf{x}, \ldots, \mathbf{u}_k^\top \mathbf{x}) \in \mathbb{R}^k$



Formalism

- A **projection** of a vector $\mathbf{x} \in \mathbb{R}^d$ to the affine subspace U is

$$\pi_{\mathsf{U}}(\mathbf{x}) = \sum_{i=1}^{k} (\mathbf{u}_i^\top \mathbf{x}) \cdot \mathbf{u}_i + \mathbf{b}$$

  - d-dimensional, with an alternative representation $\mathbf{a} = (\mathbf{u}_1^\top \mathbf{x}, \ldots, \mathbf{u}_k^\top \mathbf{x}) \in \mathbb{R}^k$

- The projection admits a matrix form:

$$\pi_{\mathsf{U}}(\mathbf{x}) = \left(\sum_{i=1}^{k} \mathbf{u}_i \mathbf{u}_i^\top \right) \mathbf{x} + \mathbf{b}$$

a $d \times d$ matrix with the rank k

$$=: \mathbf{U}\mathbf{x} + \mathbf{b}$$

  - The projection matrix $\mathbf{U}$ satisfies (1) $\mathbf{U}^\top = \mathbf{U}$
(2) $\mathbf{U}^\top\mathbf{U} = \mathbf{U}$

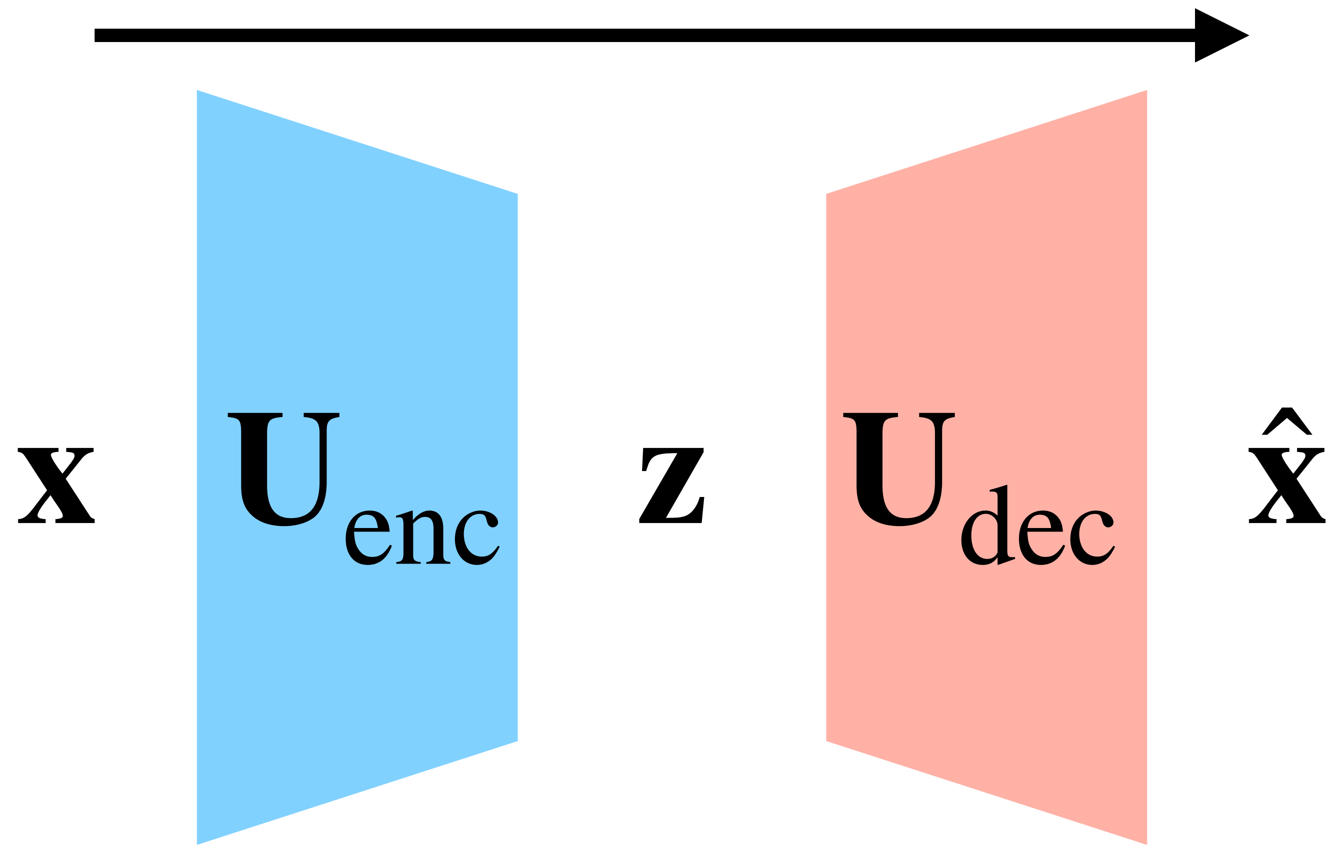
Formalism

- In a sense, projection consists of two operations
  - Compression (or encoding, $\mathbb{R}^d \to \mathbb{R}^k$)

$$\mathbf{z} = \mathbf{U}_{\text{enc}}\mathbf{x}, \qquad \text{where} \quad \mathbf{U}_{\text{enc}} = \begin{bmatrix} \leftarrow & \mathbf{u}_1^\top & \rightarrow \\ & \cdots & \\ \leftarrow & \mathbf{u}_k^\top & \rightarrow \end{bmatrix} \in \mathbb{R}^{k\times d}$$

  - Reconstruction (or decoding, $\mathbb{R}^k \to \mathbb{R}^d$)

$$\hat{\mathbf{x}} = \mathbf{U}_{\text{dec}}\mathbf{z} + \mathbf{b}, \qquad \text{where} \quad \mathbf{U}_{\text{dec}} = \mathbf{U}_{\text{enc}}^\top \in \mathbb{R}^{d\times k}$$



PCA: Variance maximization

Variance Maximization

- For PCA, we want to find a nice $\mathbf{U}$ such that

$$\max_{\mathbf{U}} \ \mathrm{Var}\Big(\mathbf{U}\mathbf{x}_1 + \mathbf{b}, \ldots, \mathbf{U}\mathbf{x}_n + \mathbf{b}\Big)$$

Variance Maximization

- For PCA, we want to find a nice $\mathbf{U}$ such that

$$\max_{\mathbf{U}} \ \mathrm{Var}\Big(\mathbf{U}\mathbf{x}_1 + \mathbf{b}, \ldots, \mathbf{U}\mathbf{x}_n + \mathbf{b}\Big)$$

- As the constant term does not affect variance, this is equal to

$$\max_{\mathbf{U}} \ \mathrm{Var}\big(\mathbf{U}\mathbf{x}_1, \ldots, \mathbf{U}\mathbf{x}_n\big)$$

Variance Maximization

- For PCA, we want to find a nice $\mathbf{U}$ such that

$$\max_{\mathbf{U}} \ \mathrm{Var}\Big(\mathbf{U}\mathbf{x}_1 + \mathbf{b}, \ldots, \mathbf{U}\mathbf{x}_n + \mathbf{b}\Big)$$

- As the constant term does not affect variance, this is equal to

$$\max_{\mathbf{U}} \ \mathrm{Var}\big(\mathbf{U}\mathbf{x}_1, \ldots, \mathbf{U}\mathbf{x}_n\big)$$

- Let $\bar{\mathbf{x}}$ be the mean of $\{\mathbf{x}_i\}_{i=1}^n$. Then, the variance can be written as:

$$\begin{aligned}\mathrm{Var}\big(\mathbf{U}\mathbf{x}_1, \ldots, \mathbf{U}\mathbf{x}_n\big) &= \frac{1}{n}\sum_{i=1}^n \|\mathbf{U}(\mathbf{x}_i - \bar{\mathbf{x}})\|_2^2 = \frac{1}{n}\sum_{i=1}^n (\mathbf{x}_i - \bar{\mathbf{x}})^\top \mathbf{U}^\top \mathbf{U}(\mathbf{x}_i - \bar{\mathbf{x}}) \\ &= \frac{1}{n}\sum_{i=1}^n (\mathbf{x}_i - \bar{\mathbf{x}})^\top \mathbf{U}(\mathbf{x}_i - \bar{\mathbf{x}})\end{aligned}$$

Variance Maximization

$$\max_{\mathbf{U}} \frac{1}{n} \sum_{i=1}^{n} (\mathbf{x}_i - \bar{\mathbf{x}})^\top \mathbf{U} (\mathbf{x}_i - \bar{\mathbf{x}})$$

- By the definition of $\mathbf{U}$, we can re-write the above as

$$\max_{\mathbf{U}} \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{k} (\mathbf{x}_i - \bar{\mathbf{x}})^\top \mathbf{u}_j \mathbf{u}_j^\top (\mathbf{x}_i - \bar{\mathbf{x}})$$

$$= \max_{\mathbf{U}} \sum_{j=1}^{k} \mathbf{u}_j^\top \left(\frac{1}{n} \sum_{i=1}^{n} (\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^\top \right) \mathbf{u}_j$$

= sample covariance matrix $\mathbf{S}$

(positive-semidefinite)

Variance Maximization

$$\max_{\mathbf{U}} \frac{1}{n} \sum_{i=1}^{n} (\mathbf{x}_i - \bar{\mathbf{x}})^\top \mathbf{U} (\mathbf{x}_i - \bar{\mathbf{x}})$$

- By the definition of $\mathbf{U}$, we can re-write the above as

$$\max_{\mathbf{U}} \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{k} (\mathbf{x}_i - \bar{\mathbf{x}})^\top \mathbf{u}_j \mathbf{u}_j^\top (\mathbf{x}_i - \bar{\mathbf{x}})$$

$$= \max_{\mathbf{U}} \sum_{j=1}^{k} \mathbf{u}_j^\top \left(\frac{1}{n} \sum_{i=1}^{n} (\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^\top \right) \mathbf{u}_j$$

- Thus, PCA is about solving the **constrained quadratic optimization**

$$\max_{\mathbf{u}_1, \ldots, \mathbf{u}_k} \sum_{j=1}^{k} \mathbf{u}_j^\top \mathbf{S} \mathbf{u}_j, \qquad \text{subject to} \quad \mathbf{u}_i^\top \mathbf{u}_j = \begin{cases} 1 & \cdots \quad i = j \\ 0 & \cdots \quad i \neq j \end{cases}$$

Solving the quadratic problem

$$\max_{\mathbf{u}_1,\ldots,\mathbf{u}_k} \sum_{j=1}^{k} \mathbf{u}_j^\top \mathbf{S}\mathbf{u}_j, \qquad \text{subject to} \quad \mathbf{u}_i^\top \mathbf{u}_j = \mathbf{1}\{i = j\}$$

- How do we solve this problem?

Solving the quadratic problem

$$\max_{\mathbf{u}_1,\ldots,\mathbf{u}_k} \sum_{j=1}^{k} \mathbf{u}_j^\top \mathbf{S}\mathbf{u}_j, \qquad \text{subject to} \quad \mathbf{u}_i^\top \mathbf{u}_j = \mathbf{1}\{i = j\}$$

- How do we solve this problem?

- **Strategy.** Perform greedy optimization

  - Select a nice $\mathbf{u}_1$ that maximizes $\mathbf{u}_1^\top \mathbf{S}\mathbf{u}_1$, subject to $\mathbf{u}_1^\top \mathbf{u}_1 = 1$

  - Select a nice $\mathbf{u}_2$ that maximizes $\mathbf{u}_2^\top \mathbf{S}\mathbf{u}_2$, subject to $\mathbf{u}_2^\top \mathbf{u}_2 = 1$ and $\mathbf{u}_2^\top \mathbf{u}_1 = 0$

  - ...

Solving the quadratic problem

- Let us take a look at the first step: determining $\mathbf{u}_1$

$$\max_{\mathbf{u}} \mathbf{u}^\top \mathbf{S}\mathbf{u}, \qquad \text{subject to} \quad \mathbf{u}^\top \mathbf{u} = 1$$

Solving the quadratic problem

- Let us take a look at the first step: determining $\mathbf{u}_1$

$$\max_{\mathbf{u}} \mathbf{u}^\top \mathbf{S}\mathbf{u}, \qquad \text{subject to} \quad \mathbf{u}^\top \mathbf{u} = 1$$

- To solve this, consider the Lagrangian relaxation

$$\max_{\mathbf{u}} \mathbf{u}^\top \mathbf{S}\mathbf{u} + \alpha(1 - \mathbf{u}^\top \mathbf{u})$$

  - The critical point is where $\mathbf{S}\mathbf{u} = \alpha\mathbf{u}$ holds, i.e., eigenvectors.

Solving the quadratic problem

- Let us take a look at the first step: determining $\mathbf{u}_1$

$$\max_{\mathbf{u}} \mathbf{u}^\top \mathbf{S}\mathbf{u}, \qquad \text{subject to} \quad \mathbf{u}^\top \mathbf{u} = 1$$

  - To solve this, consider the Lagrangian relaxation

$$\max_{\mathbf{u}} \mathbf{u}^\top \mathbf{S}\mathbf{u} + \alpha(1 - \mathbf{u}^\top \mathbf{u})$$

    - The critical point is where $\mathbf{S}\mathbf{u} = \alpha\mathbf{u}$ holds, i.e., eigenvectors.

    - Choosing the principal component (eigenvector with the largest eigenvalue) maximizes the value of $\mathbf{u}^\top \mathbf{S}\mathbf{u}$

Solving the quadratic problem

- Next, try to determine $\mathbf{u}_2$

$$\max_{\mathbf{u}} \mathbf{u}^\top \mathbf{S} \mathbf{u}, \qquad \text{subject to} \quad \mathbf{u}^\top \mathbf{u} = 1, \mathbf{u}^\top \mathbf{u}_1 = 0$$

Solving the quadratic problem

- Next, try to determine $\mathbf{u}_2$

$$\max_{\mathbf{u}} \mathbf{u}^\top \mathbf{S}\mathbf{u}, \qquad \text{subject to} \quad \mathbf{u}^\top \mathbf{u} = 1, \mathbf{u}^\top \mathbf{u}_1 = 0$$

- The Lagrangian becomes

$$\mathbf{u}^\top \mathbf{S}\mathbf{u} + \alpha(1 - \mathbf{u}^\top \mathbf{u}) - \beta(\mathbf{u}^\top \mathbf{u}_1)$$

- The critical point condition is

$$\mathbf{S}\mathbf{u} = \alpha\mathbf{u} + \frac{\beta}{2}\mathbf{u}_1$$

Solving the quadratic problem

- Next, try to determine $\mathbf{u}_2$

$$\max_{\mathbf{u}} \mathbf{u}^\top \mathbf{S}\mathbf{u}, \qquad \text{subject to} \quad \mathbf{u}^\top \mathbf{u} = 1, \mathbf{u}^\top \mathbf{u}_1 = 0$$

  - The Lagrangian becomes

$$\mathbf{u}^\top \mathbf{S}\mathbf{u} + \alpha(1 - \mathbf{u}^\top \mathbf{u}) - \beta(\mathbf{u}^\top \mathbf{u}_1)$$

  - The critical point condition is

$$\mathbf{S}\mathbf{u} = \alpha \mathbf{u} + \frac{\beta}{2}\mathbf{u}_1$$

  - Multiplying $\mathbf{u}_1^\top$ on both sides, we get

$$\underbrace{\mathbf{u}_1^\top \mathbf{S}\mathbf{u}}_{=0} = \alpha \underbrace{\mathbf{u}_1 \mathbf{u}}_{=0} + \frac{\beta}{2}$$

  - and thus we get $\beta = 0$

Solving the quadratic problem

- Using $\beta = 0$, our Lagrangian becomes

$$\mathbf{u}^\top \mathbf{S}\mathbf{u} + \alpha(1 - \mathbf{u}^\top \mathbf{u})$$

  with the critical point condition

$$\mathbf{S}\mathbf{u} = \alpha \mathbf{u}$$

- Thus, our solution should be selecting the eigenvector with 2nd largest eigenvalue

Solving the quadratic problem

- Using $\beta = 0$, our Lagrangian becomes

$$\mathbf{u}^\top \mathbf{S}\mathbf{u} + \alpha(1 - \mathbf{u}^\top \mathbf{u})$$

with the critical point condition

$$\mathbf{S}\mathbf{u} = \alpha \mathbf{u}$$

- Thus, our solution should be selecting the eigenvector with 2nd largest eigenvalue

- Repeat this procedure, and get top-k principal components of the sample covariance matrix as our bases $\mathbf{u}_1, \ldots, \mathbf{u}_k$.
  - Can be done by performing SVD on the data matrix

$$\mathbf{X} = [\mathbf{x}_1 - \bar{\mathbf{x}} \mid \cdots \mid \mathbf{x}_n - \bar{\mathbf{x}}] = \mathbf{U}\Sigma\mathbf{V}^\top$$

  and selecting the columns of $\mathbf{U}$ for top-k singular values.

Wrapping up

- **Today**
    - Dimensionality reduction
    - Principal component analysis
        - Basic maths on projection
        - PCA as Variance maximization
            - Solved in a greedy manner
- **Next class.**
    - PCA continued
        - PCA as distortion minimization
        - Applications and Limitations
        - Modern versions

Cheers