Dimensionality Reduction

Lecture 12

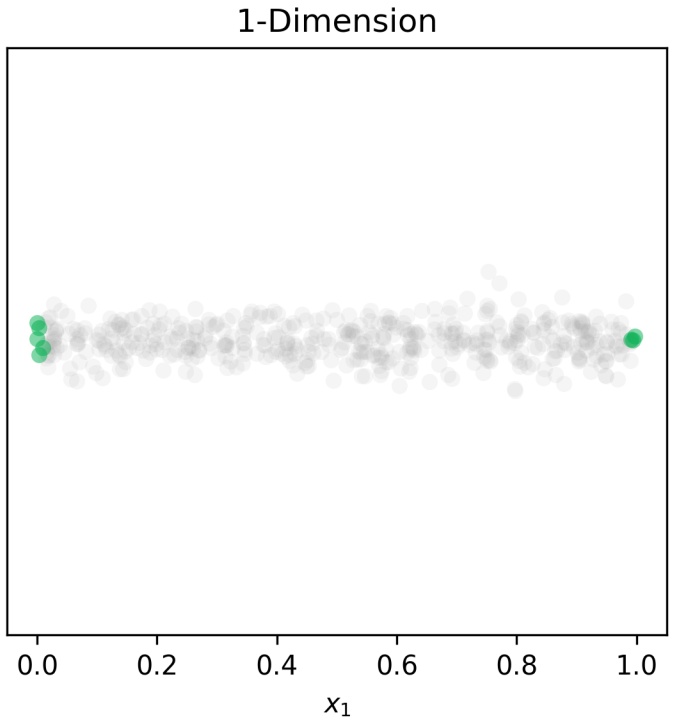
The Curse of Dimensionality

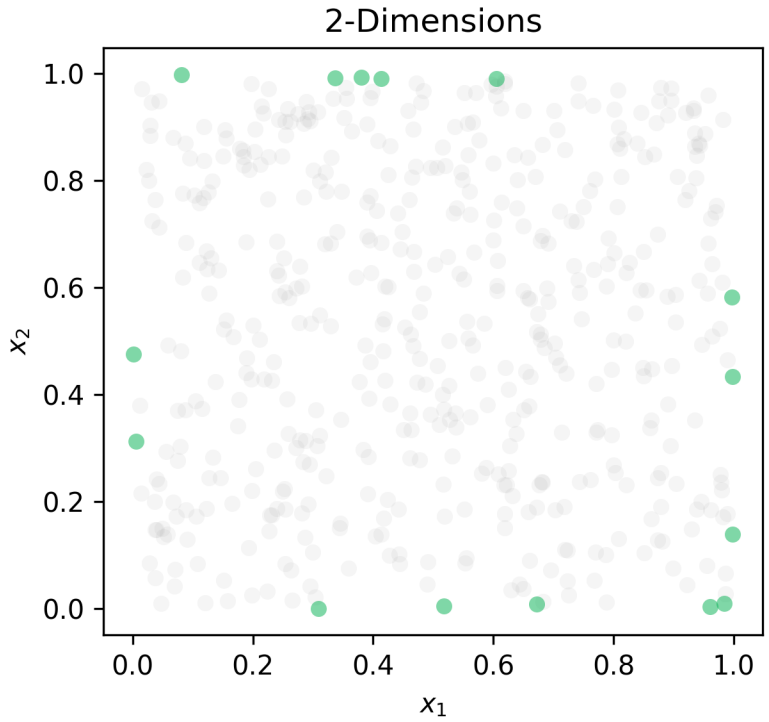
Challenge 1

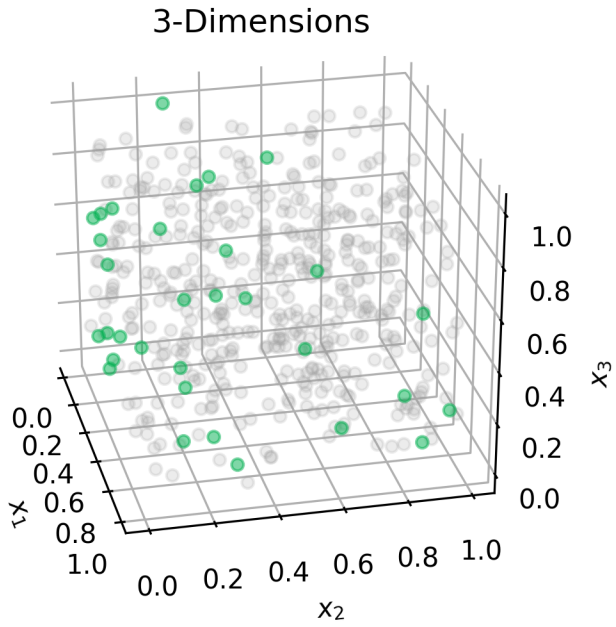
In high dimensions, data become sparse
(increasing the risk of overfitting)

Random data points in a unit hypercube…

- Data point is a distance < 0.01 units from the edge of a unit hypercube
- All other data







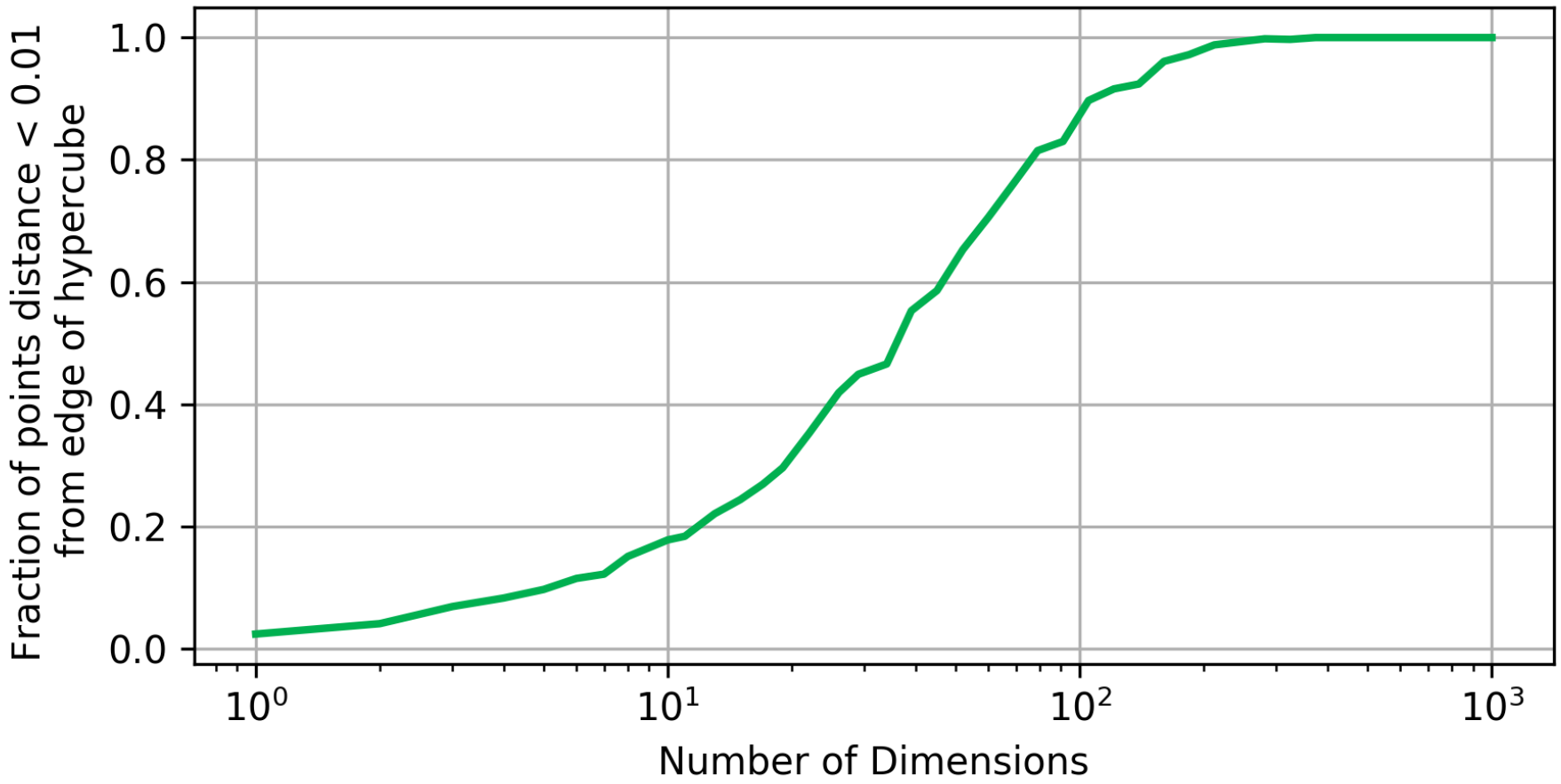
Fraction of edge data = $\frac{\text{●}}{\text{● + ●}}$

0.016

0.030

0.064

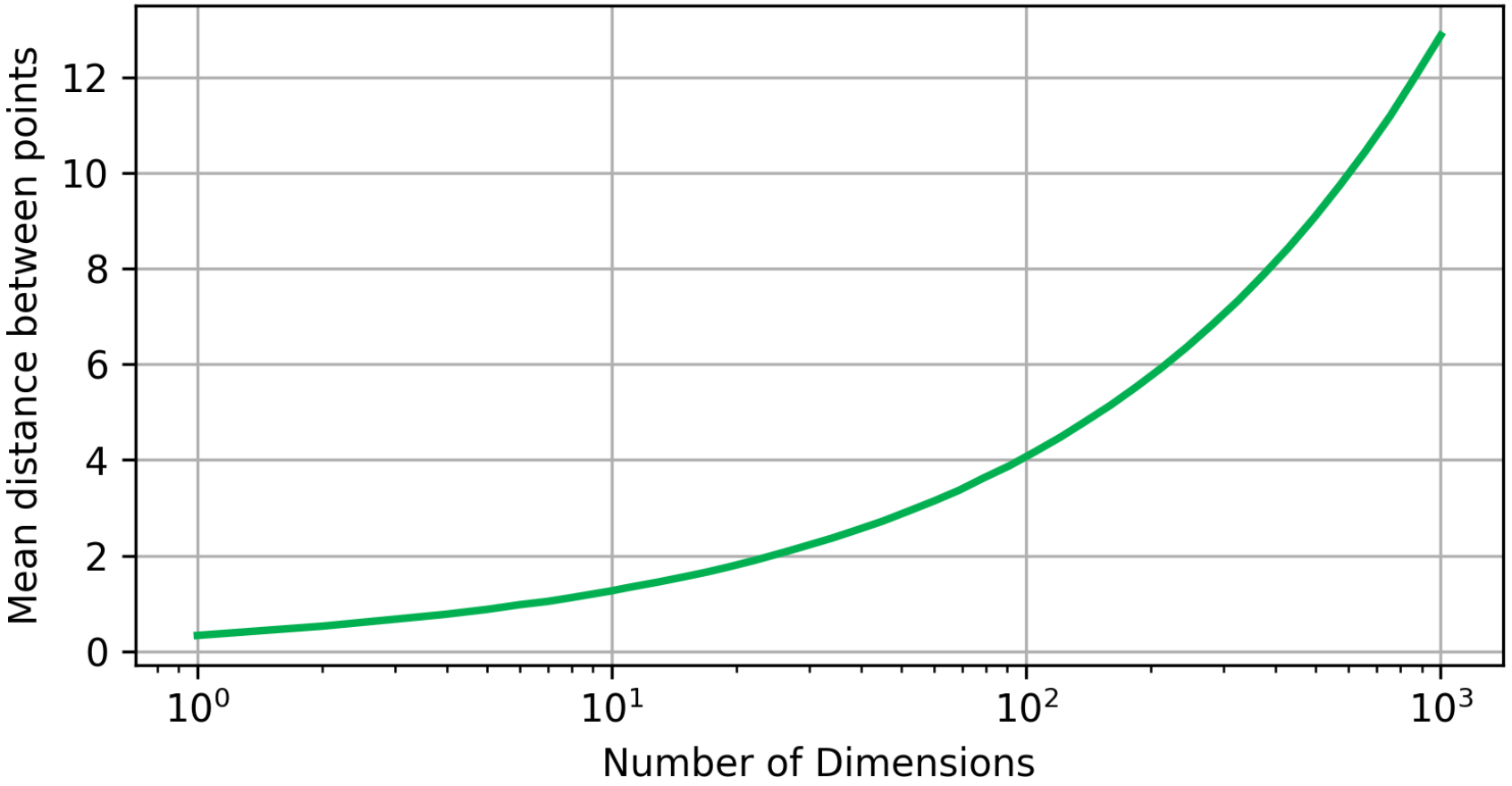
In high dimensions…



…nearly all of the high dimensional space is **far away from the center**

Note: figures constructed using 1,000 random points

In high dimensions…



…data become sparse

Note: figures constructed using 1,000 random points

Challenge 2

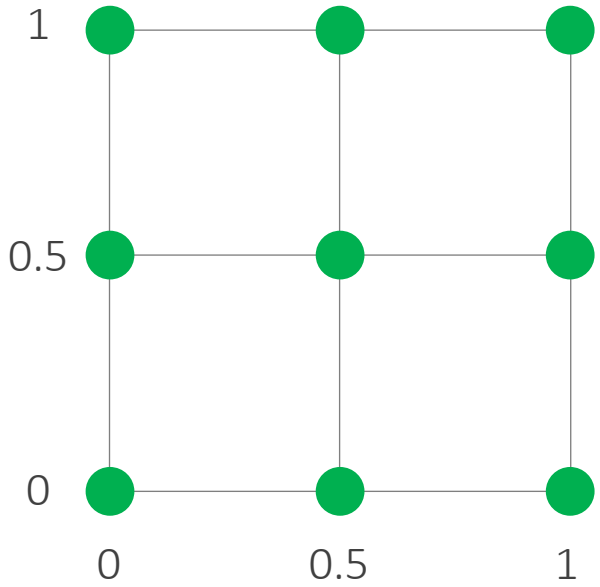
Much more data are needed for sampling higher dimensional spaces

Sample a unit hypercube on a grid spaced at intervals of 0.5

D = 1

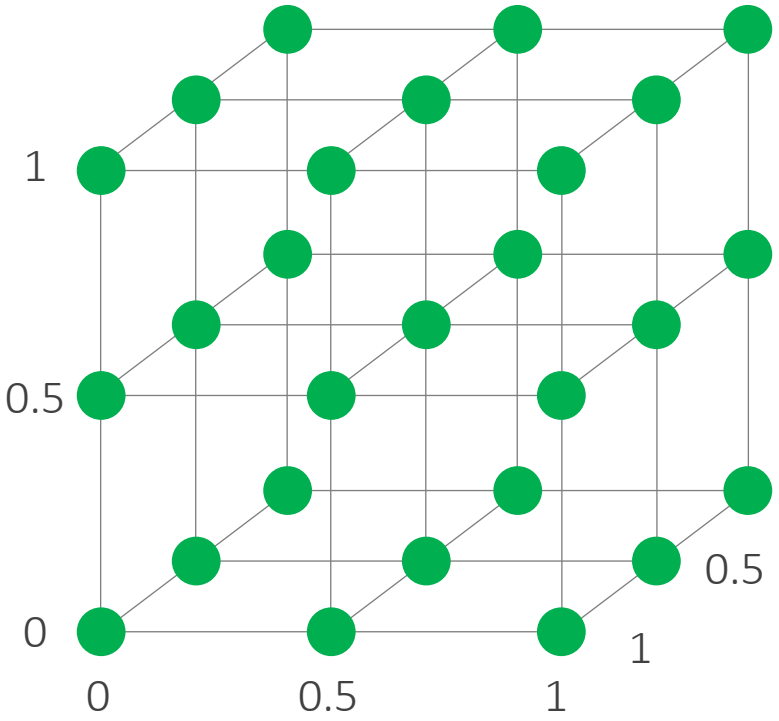
N = 3

D = 2



N = 9

D = 3



N = 27

Dim (D)	Samples (N)
4	81
5	243
10	59,049
100	5.2×10^{47}
300	**1.4×10^{143}**

1 million Googles, each with 20 exabytes of data would only be 10^{25} bytes

…it takes more data to learn in high dimensional spaces

Dimensionality Reduction

Benefits:

Simplified computation
Reduced redundancy of features
Improved numerical stability due to removed correlations

Popular approach:

Principal Components Analysis (PCA)

Other approaches:

Linear Discriminant Analysis (LDA) [supervised]
Non-negative Matrix Factorization (NMF)
Multidimensional scaling (MDS)
Autoencoder
t-distributed Stochastic Neighbor Embedding (t-SNE)
Uniform Manifold Approximation and Projection (UMAP)

PCA

Before you begin: Normalize the data!

For each feature, subtract the mean and divide by the standard deviation

Normalized feature — Raw, unscaled feature

$$x = \frac{x' - \overline{x'}}{\sigma_{x'}}$$

columns = features

$$\mathbf{X} = \begin{bmatrix} x_{11} & \cdots & x_{1D} \\ \vdots & \ddots & \vdots \\ x_{N1} & \cdots & x_{ND} \end{bmatrix}$$

rows = observations

We normalize each of the columns

Principal components analysis

a.k.a.
Karhunen–Loève Transform
Proper orthogonal decomposition
Hotelling transform

Transform the data from a high dimensional space to a lower dimensional subspace, while minimizing the projection error

$$\mathbf{A} \qquad \mathbf{x} \qquad \rightarrow \qquad \mathbf{z}$$

$$\mathbb{R}^{M \times D} \qquad \mathbb{R}^{D} \qquad \mathbb{R}^{M} \qquad M < D$$

$$\begin{bmatrix} u_{11} & \cdots & u_{1D} \\ \vdots & \ddots & \vdots \\ u_{M1} & \cdots & u_{MD} \end{bmatrix} \qquad \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_D \end{bmatrix} \qquad \begin{bmatrix} z_1 \\ z_2 \\ z_3 \\ \vdots \\ z_M \end{bmatrix}$$

linear transformation matrix
(this is what we want to find through PCA)

sample of data in original D-dimensional space
(this is one of N observations)

Transformed data in M-dimensional (lower dimensional) subspace

Principal components analysis

$$A$$

$$\begin{bmatrix} u_{11} & \cdots & u_{1D} \\ \vdots & \ddots & \vdots \\ u_{M1} & \cdots & u_{MD} \end{bmatrix} = \begin{bmatrix} - \ \mathbf{u}_1^T \ - \\ \vdots \\ - \ \mathbf{u}_M^T \ - \end{bmatrix}$$

linear transformation matrix

Each $\mathbf{u}_i$ represents a unit vector

The i^{th} principal component:

$$z_i = \mathbf{u}_i^T \mathbf{x}$$

scalar — unit vector

Since only direction matters, we assume the $\mathbf{u}_i$ are unit vectors

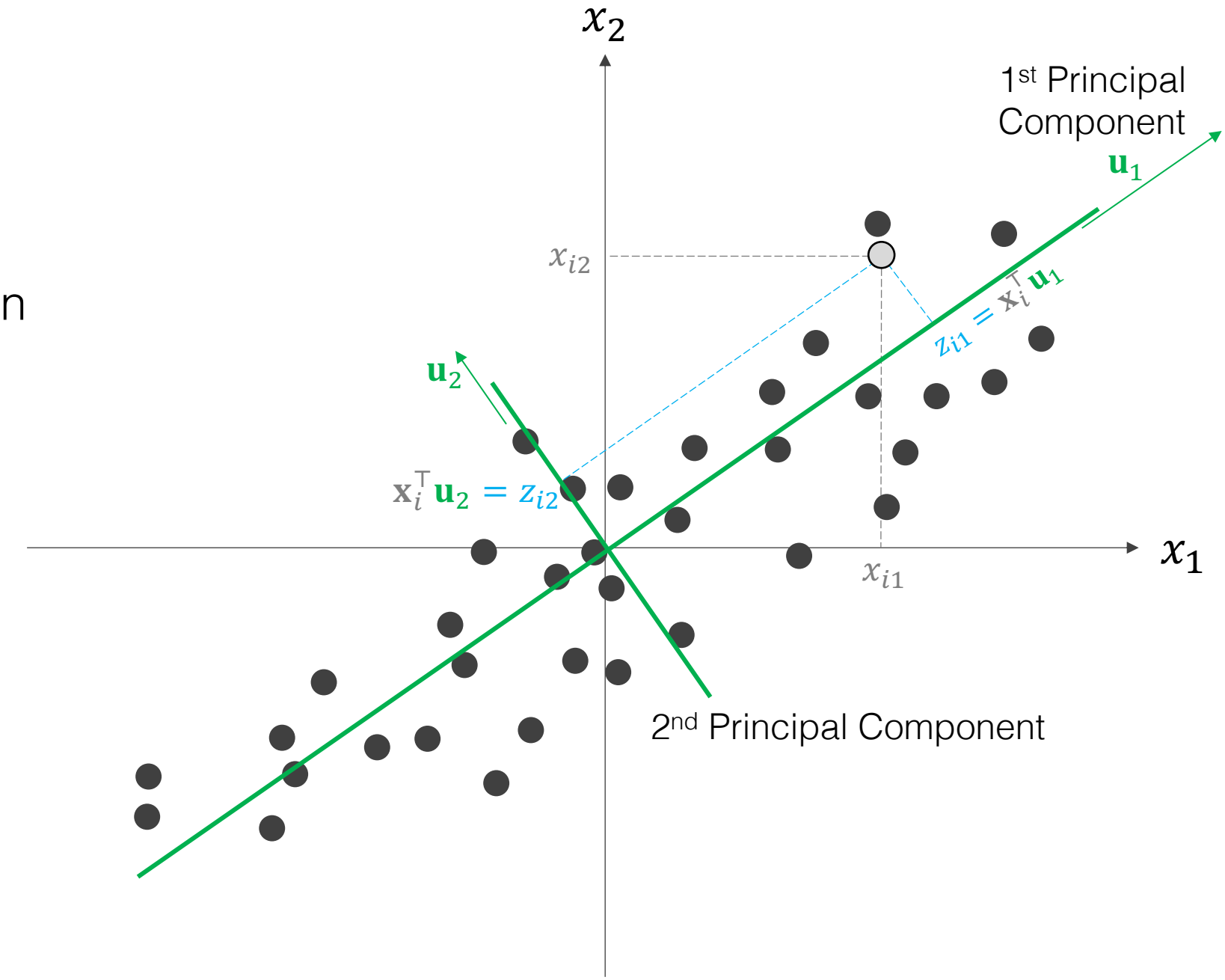
$$\mathbf{u}_i^T \mathbf{u}_i = 1$$

Principal Components

Maximum variance formulation

Length of a projection onto a unit vector:

$$z_{i1} = \mathbf{x}_i^\top \mathbf{u}_1$$



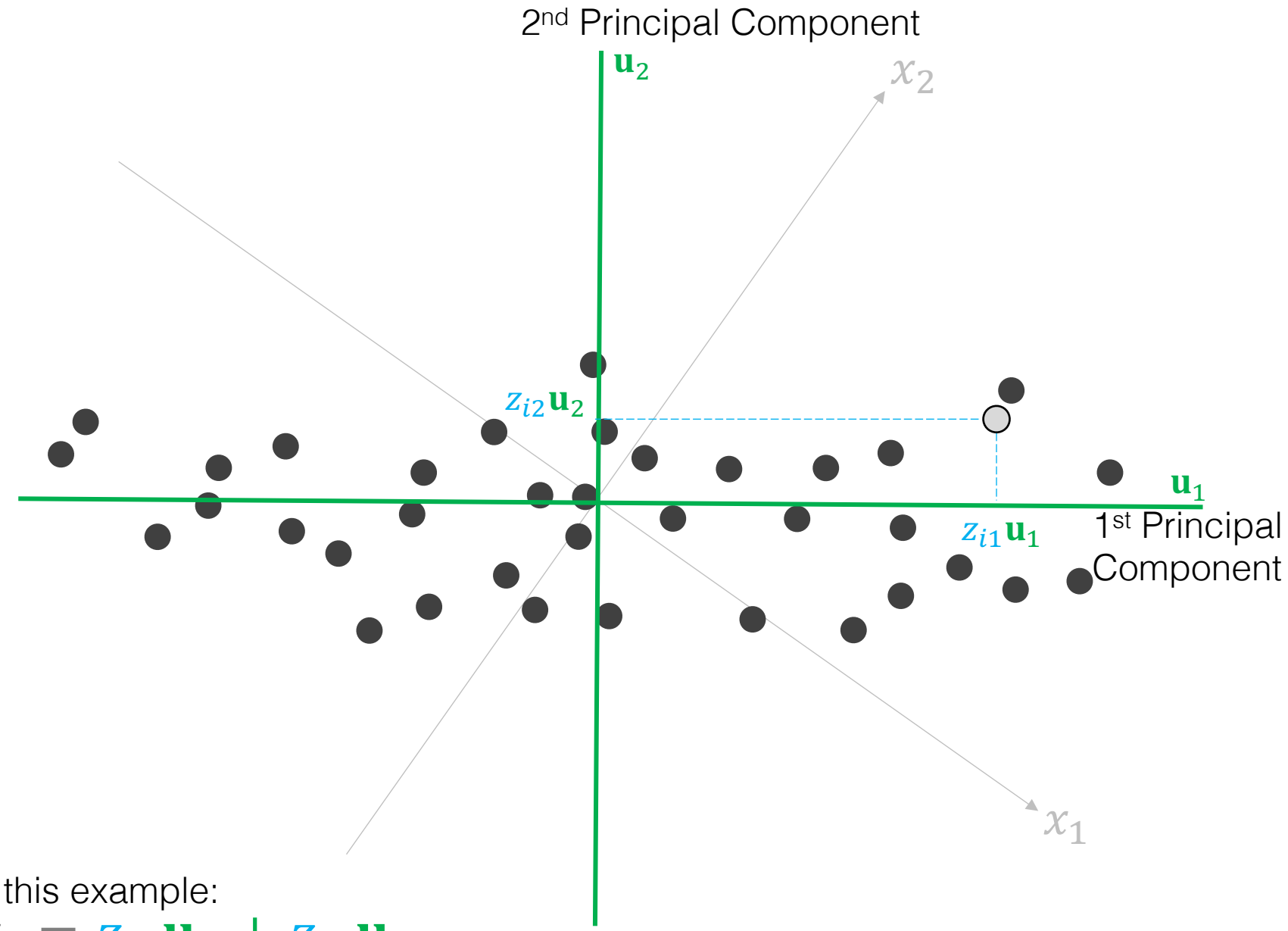
$$\mathbf{x}_i = \begin{bmatrix} x_{i1} \\ x_{i2} \end{bmatrix}$$

Reprojected Data onto Principal Components

Any point $\boldsymbol{x}_i$ can be represented as a combination of the principle components

$$\mathbf{x}_i = \sum_{j=1}^{D} z_{ij}\mathbf{u}_j$$

The $\mathbf{u}_j$'s are an orthogonal basis for the space $\mathbb{R}^D$



In this example:

$$\mathbf{x}_i = z_{i1}\mathbf{u}_1 + z_{i2}\mathbf{u}_2$$

Approximating data with principal components

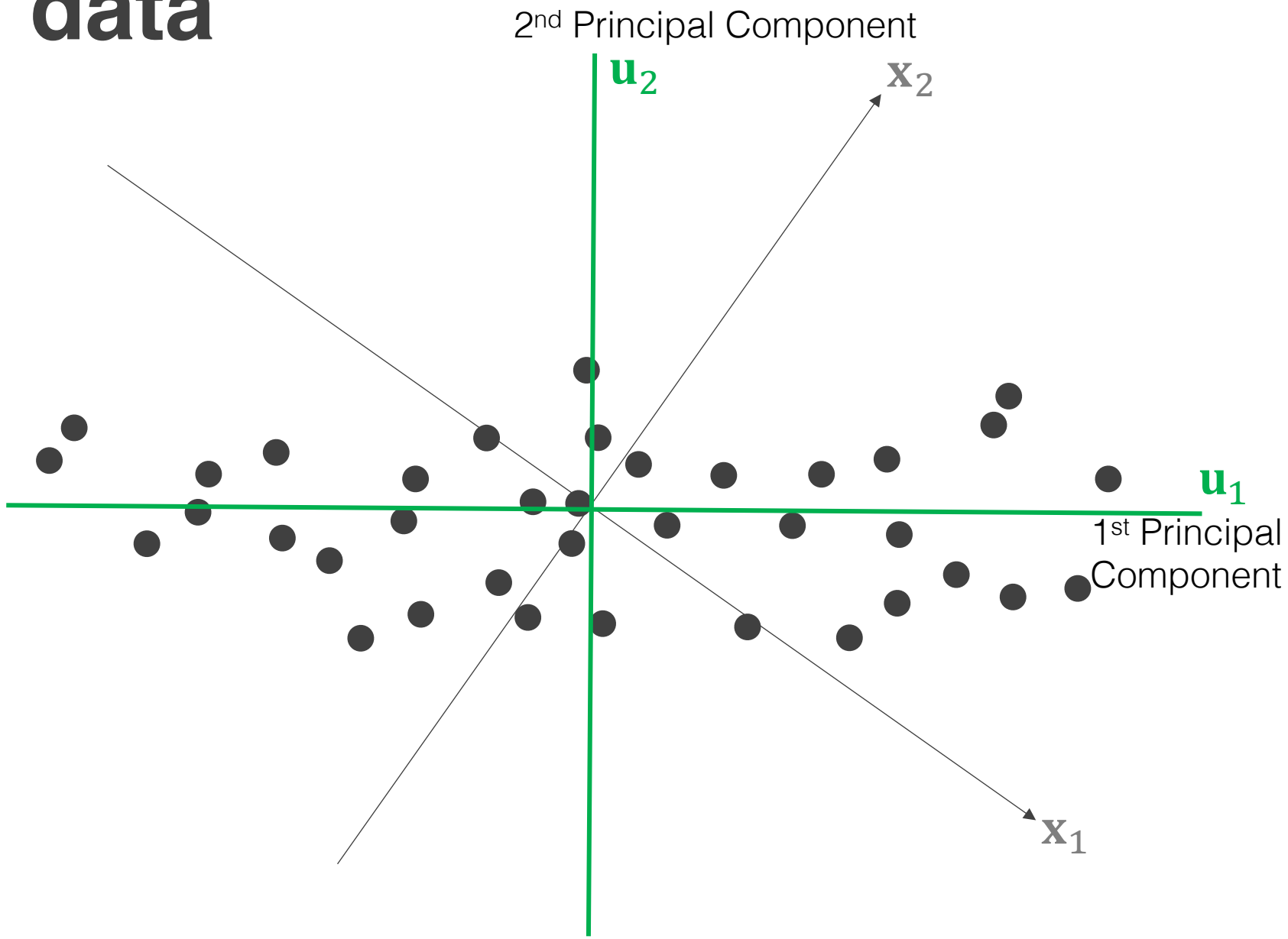
scalar projection

principal component direction

$$\mathbf{x}_j = \sum_{i=1}^{D} \left(\mathbf{x}_j^T \mathbf{u}_i\right) \mathbf{u}_i$$

Since $\mathbf{x}_j = \sum_{j=1}^{D} z_{ji} \mathbf{u}_i$

$$z_{ji} = \mathbf{x}_j^\top \mathbf{u}_i$$



PCA

We want to **maximize the variance** of the **projected data**

Goal

Let's start by finding the **unit vector in the direction of greatest variation** in the dataset

Here the magnitude is unimportant, but the direction matters

We seek to project each point $\mathbf{x}_i$ onto a unit PC vector. $z_i = \mathbf{u}_1^T \mathbf{x}_i$

PCA: Compute the variance of the transformed data

Mean of the data:

$$\bar{\mathbf{x}} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{x}_i \qquad \mathbf{x}_i \ [D\times 1]$$

The projected mean of the data:

$$\bar{z} = \mathbf{u}_1^T\,\bar{\mathbf{x}}$$

We can compute the (projected) variance of the data as:

$$\underset{\text{[scalar]}}{\sigma_z^2} = \frac{1}{N}\sum_{i=1}^{N}(z_i - \bar{z})^2$$

$$\sigma_z^2 = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{u}_1^T\mathbf{x}_i - \mathbf{u}_1^T\bar{\mathbf{x}})^2$$

The magnitude z_i of our data $\mathbf{x}_i$ projected length on the unit vector $\mathbf{u}_1$ is:

$$z_i = \mathbf{u}_1^T\,\mathbf{x}_i$$

PCA: Compute the variance of the transformed data

We can compute the variance as:

$$\sigma_z^2 = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{u}_1^T\mathbf{x}_i - \mathbf{u}_1^T\bar{\mathbf{x}})^2$$

$$= \frac{1}{N}\sum_{i=1}^{N}\mathbf{u}_1^T(\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^T\mathbf{u}_1$$

$$= \mathbf{u}_1^T\boldsymbol{\Sigma}\mathbf{u}_1$$

Variance of the projected data

Define:

$$\boldsymbol{\Sigma} = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^T$$

Covariance matrix of our data

Covariance matrix

$$\mathbf{x}_i = \begin{bmatrix} x_{i1} \\ x_{i2} \\ \vdots \\ x_{iD} \end{bmatrix}$$

Vector of observation i

x_{ij}: Observation index, Predictor index

$$\mathbf{X} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1D} \\ x_{21} & x_{22} & \cdots & x_{2D} \\ \vdots & \vdots & \ddots & \vdots \\ x_{N1} & x_{N2} & \cdots & x_{ND} \end{bmatrix}$$

Observations (↓) × Predictors (→)

$$\boldsymbol{\Sigma} = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^T \rightarrow [D \times D]$$

$$[D \times 1][1 \times D]$$

$$\boldsymbol{\Sigma} = \begin{bmatrix} \Sigma_{11} & \Sigma_{12} & \cdots & \Sigma_{1D} \\ \Sigma_{21} & \Sigma_{22} & \cdots & \Sigma_{2D} \\ \vdots & \vdots & \ddots & \vdots \\ \Sigma_{D1} & \Sigma_{D2} & \cdots & \Sigma_{DD} \end{bmatrix}$$

$$\Sigma_{jk} = \frac{1}{N}\sum_{i=1}^{N}(x_{ij} - \bar{x}_j)(x_{ik} - \bar{x}_k)$$

$$= \mathrm{cov}(X_j, X_k)$$

$$= E[(X_j - \mu_j)(X_k - \mu_k)]$$

$$\boldsymbol{\Sigma} = \begin{bmatrix} \sigma_1^2 & \Sigma_{12} & \cdots & \Sigma_{1D} \\ \Sigma_{21} & \sigma_2^2 & \cdots & \Sigma_{2D} \\ \vdots & \vdots & \ddots & \vdots \\ \Sigma_{D1} & \Sigma_{D2} & \cdots & \sigma_D^2 \end{bmatrix}$$

$$\sigma_j^2 = \frac{1}{N}\sum_{i=1}^{N}(x_{ij} - \bar{x}_j)^2$$

$$= E[(X_j - \mu_j)^2]$$

Mean of each predictor

If $\mu_j = 0$ for all j

This will be the case IF the data are standardized

$$\Sigma_{jk} = \frac{1}{N}\sum_{i=1}^{N} x_{ij}x_{ik}$$

$$= \frac{1}{N}\mathbf{x}_j^T\mathbf{x}_k$$

$$= E[X_jX_k]$$

$$\boldsymbol{\Sigma} = \frac{1}{N}\mathbf{X}^{\mathrm{T}}\mathbf{X}$$

Covariance matrix properties

$$\boldsymbol{\Sigma} = \begin{bmatrix} \sigma_1^2 & \Sigma_{12} & \cdots & \Sigma_{1D} \\ \Sigma_{21} & \sigma_2^2 & \cdots & \Sigma_{2D} \\ \vdots & \vdots & \ddots & \vdots \\ \Sigma_{D1} & \Sigma_{D2} & \cdots & \sigma_D^2 \end{bmatrix}$$

Positive semidefinite ($\mathbf{v}^T\boldsymbol{\Sigma}\mathbf{v} \geq 0$ for all $\mathbf{v}$) and symmetric ($\boldsymbol{\Sigma} = \boldsymbol{\Sigma}^{\mathrm{T}}$)

All eigenvalues are non-negative

Eigenvectors are orthogonal

If the features (predictors), $x_1, x_2, \ldots, x_D$ are independent, $\boldsymbol{\Sigma}$ is diagonal because $\mathbf{cov}(X_j, X_k) = 0$ if $j \neq k$

PCA: Maximize the variance

We want to **maximize variance** $\sigma_z^2 = \mathbf{u}_1^T \mathbf{\Sigma} \mathbf{u}_1$

subject to $\mathbf{u}_1^T \mathbf{u}_1 = 1$ (unit vectors)

We can use **Lagrange multipliers**:

Maximize $f(x)$

subject to the constraint $g(x)$

We maximize this: $L(x, \lambda) = f(x) - \lambda g(x)$

$$f(\mathbf{x}, \mathbf{u}_i) = \mathbf{u}_1^T \mathbf{\Sigma} \mathbf{u}_1$$

$$g(\mathbf{x}, \mathbf{u}_i) = \mathbf{u}_1^T \mathbf{u}_1 - 1 = 0$$

For our case: $L(\mathbf{x}, \mathbf{u_1}, \lambda) = \mathbf{u}_1^T \mathbf{\Sigma} \mathbf{u}_1 - \lambda(\mathbf{u}_1^T \mathbf{u}_1 - 1)$

We take the derivative and set it equal to zero

PCA

$$L(\mathbf{x}, \mathbf{u_1}, \lambda) = \mathbf{u}_1^T \mathbf{\Sigma} \mathbf{u}_1 - \lambda(\mathbf{u}_1^T \mathbf{u}_1 - 1)$$

We take the derivative with respect to $\mathbf{u}_1$ and set it equal to zero

$$\frac{\partial L}{\partial \mathbf{u}_1} = \frac{\partial}{\partial \mathbf{u}_1} \mathbf{u}_1^T \mathbf{\Sigma} \mathbf{u}_1 - \frac{\partial}{\partial \mathbf{u}_1} \lambda(\mathbf{u}_1^T \mathbf{u}_1 - 1)$$

$$= 2\mathbf{\Sigma} \mathbf{u}_1 - 2\lambda \mathbf{u}_1 = 0 \quad \text{(since } \Sigma \text{ is symmetric)}$$

$$\mathbf{\Sigma} \mathbf{u}_1 = \lambda \mathbf{u}_1 \quad \rightarrow \quad \mathbf{u}_1 \text{ is an eigenvector of the covariance matrix } \mathbf{\Sigma}\text{, and } \lambda \text{ is an eigenvalue}$$

How do we know which eigenvector to use as the first principal component?

Eigenanalysis and PCA

Eigenvector Demo:

http://setosa.io/ev/eigenvectors-and-eigenvalues/

PCA Demo:

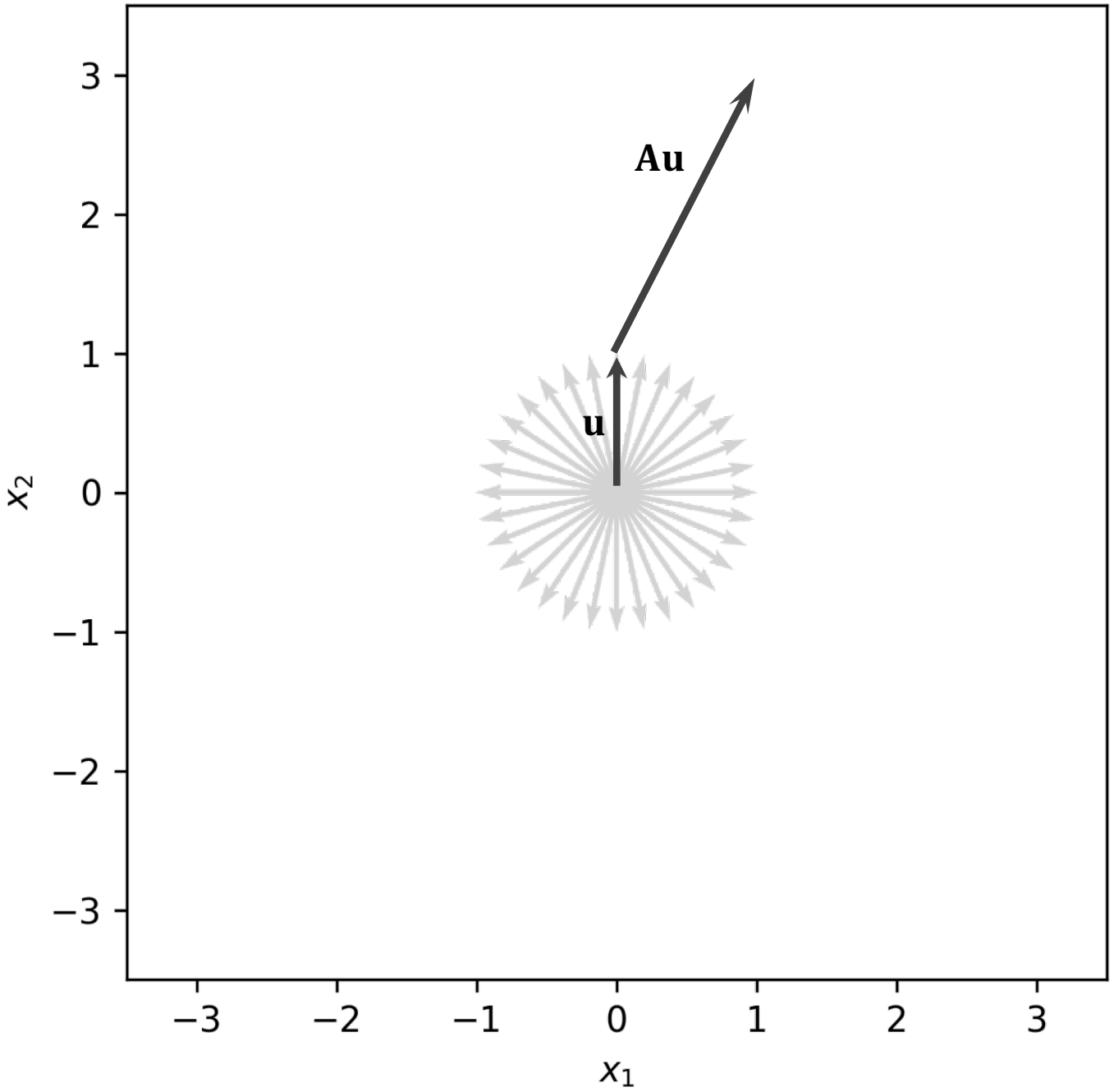
http://setosa.io/ev/principal-component-analysis/

Eigenvalues/Eigenvectors

$$\mathbf{Au} = \lambda\mathbf{u}$$

$$\mathbf{A} = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$$

$$\mathbf{u} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \qquad \mathbf{Au} = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 2 \\ 1 \end{bmatrix}$$

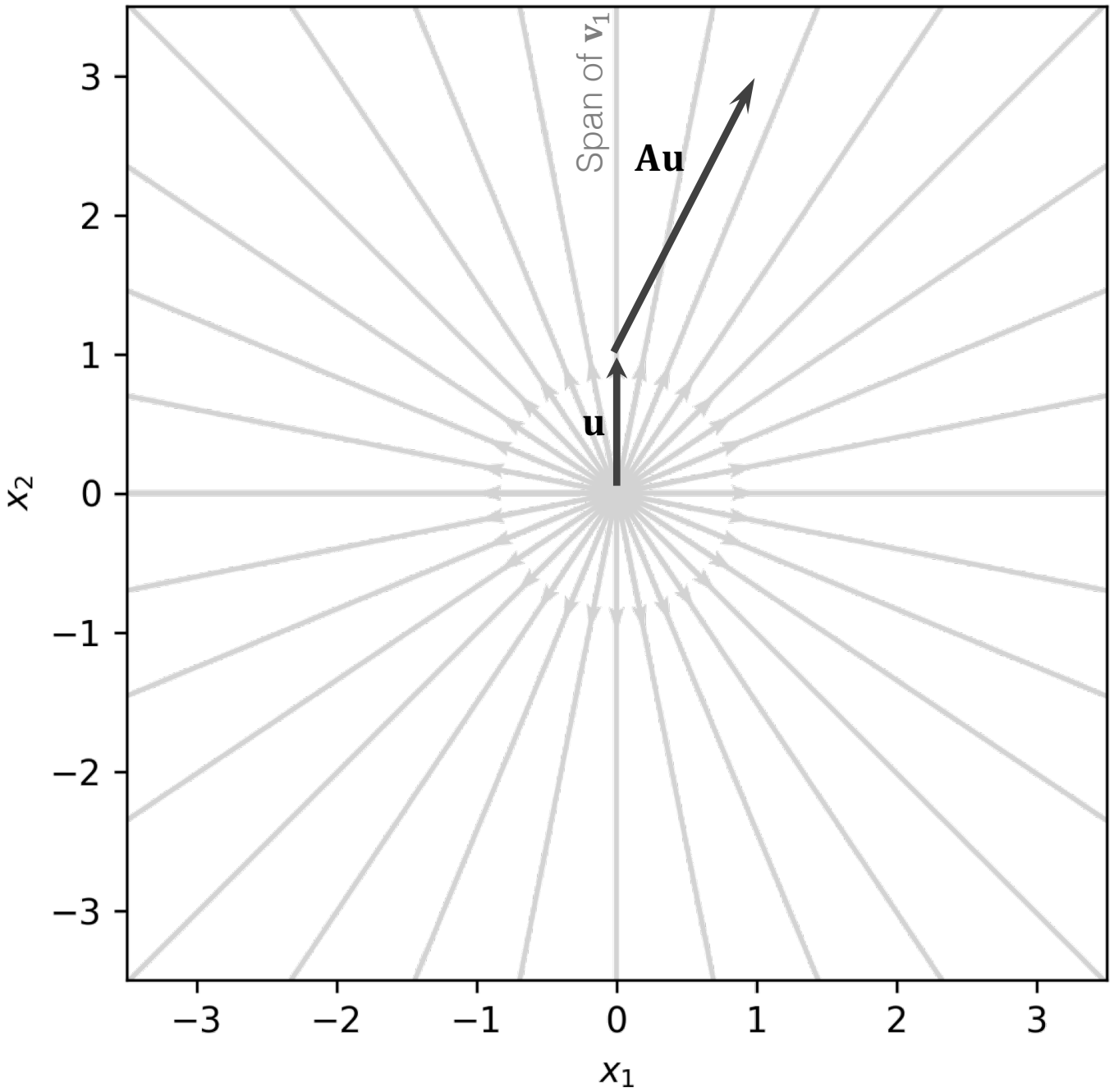


Eigenvalues/Eigenvectors

$$\mathbf{Au} = \lambda\mathbf{u}$$

$$\mathbf{A} = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$$

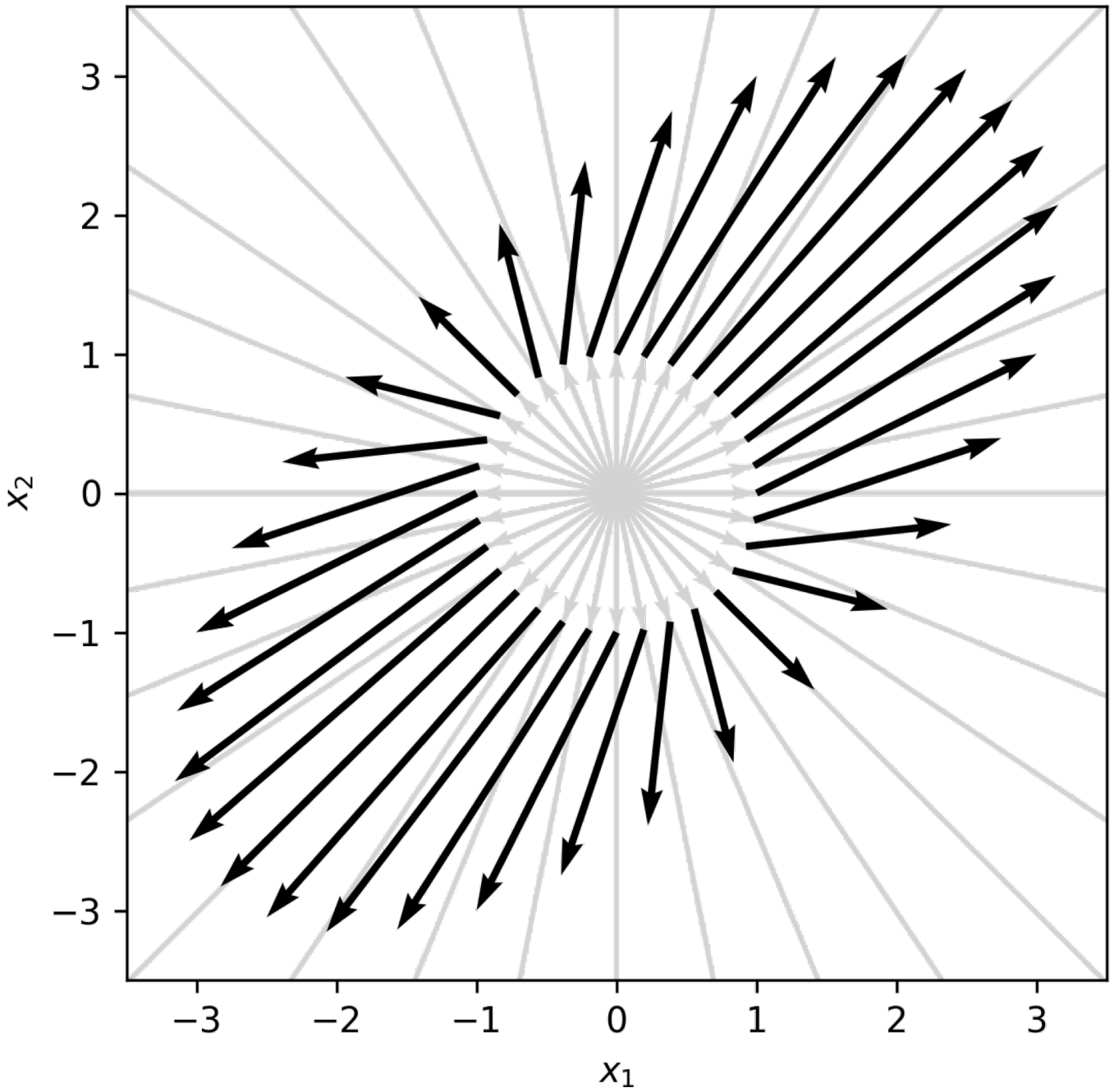
$$\mathbf{u} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \qquad \mathbf{Au} = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 2 \\ 1 \end{bmatrix}$$



Eigenvalues/Eigenvectors

$$\mathbf{Au} = \lambda\mathbf{u}$$

$$\mathbf{A} = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$$



Eigenvalues/Eigenvectors

$$\mathbf{Au} = \lambda\mathbf{u}$$

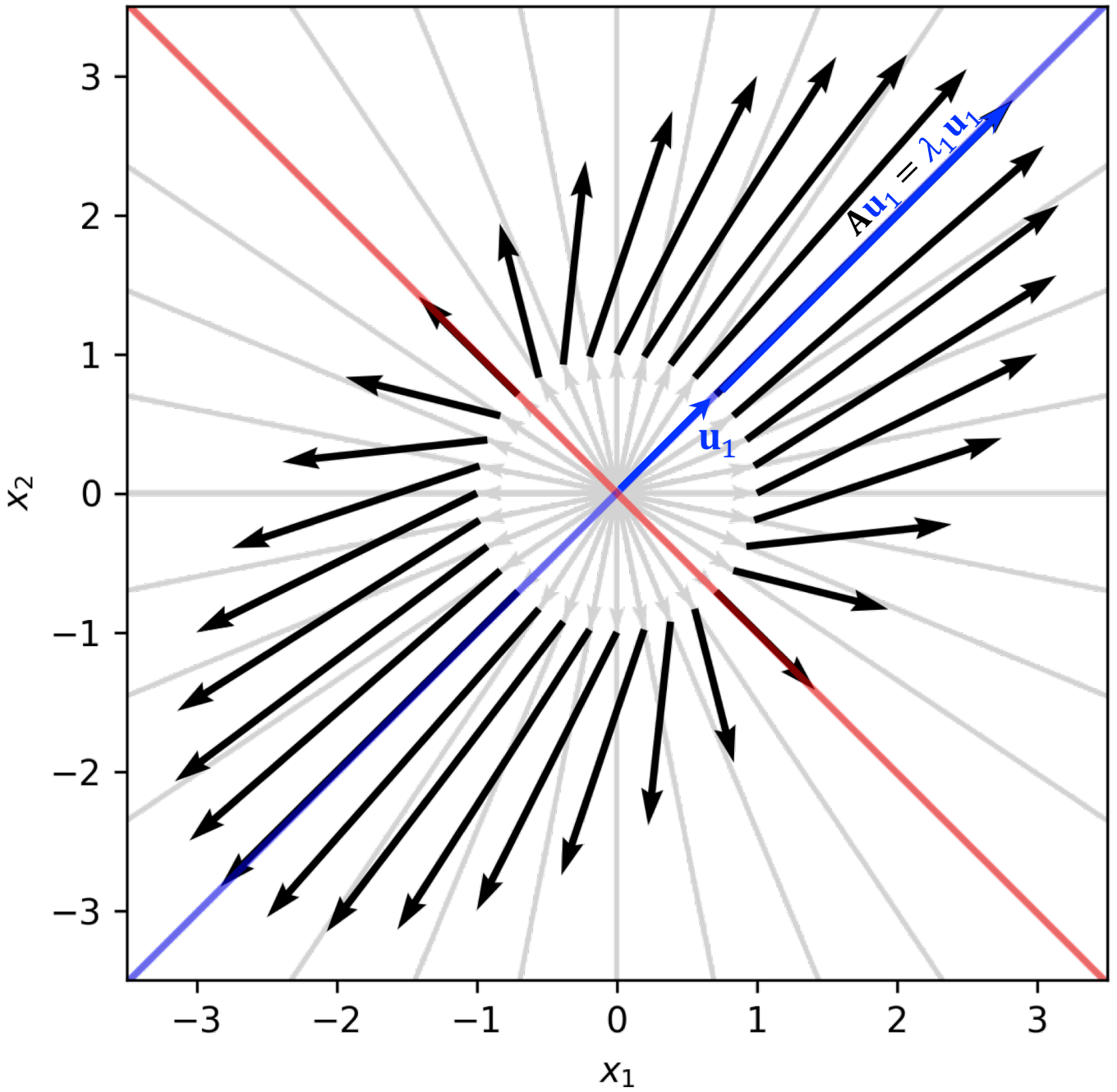
$$\mathbf{A} = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$$

$$\mathbf{u}_1 = \begin{bmatrix} 0.707 \\ 0.707 \end{bmatrix} \qquad \lambda_1 = 3$$

$$\mathbf{Au}_1 = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} 0.707 \\ 0.707 \end{bmatrix}$$

$$= \begin{bmatrix} 2.12 \\ 2.12 \end{bmatrix}$$

$$= \lambda_1\mathbf{u}_1 = 3 \begin{bmatrix} 0.707 \\ 0.707 \end{bmatrix}$$



PCA

Since we want to maximize the variance in the projected features:

We want to maximize: $\sigma_z^2 = \mathbf{u}_1^T \mathbf{\Sigma} \mathbf{u}_1$

To do that this must be true: (shown on last slide) $\mathbf{\Sigma u}_1 = \lambda \mathbf{u}_1$

So we can write: $\sigma_z^2 = \mathbf{u}_1^T \lambda \mathbf{u}_1 = \lambda \mathbf{u}_1^T \mathbf{u}_1 = \lambda$

Variance corresponding to our first principle component

Therefore we choose as our first principle component the eigenvector that corresponds to the **largest eigenvalue**

The first PC will account for the most variance, the second PC to the second most, etc.

PCA: Variance explained

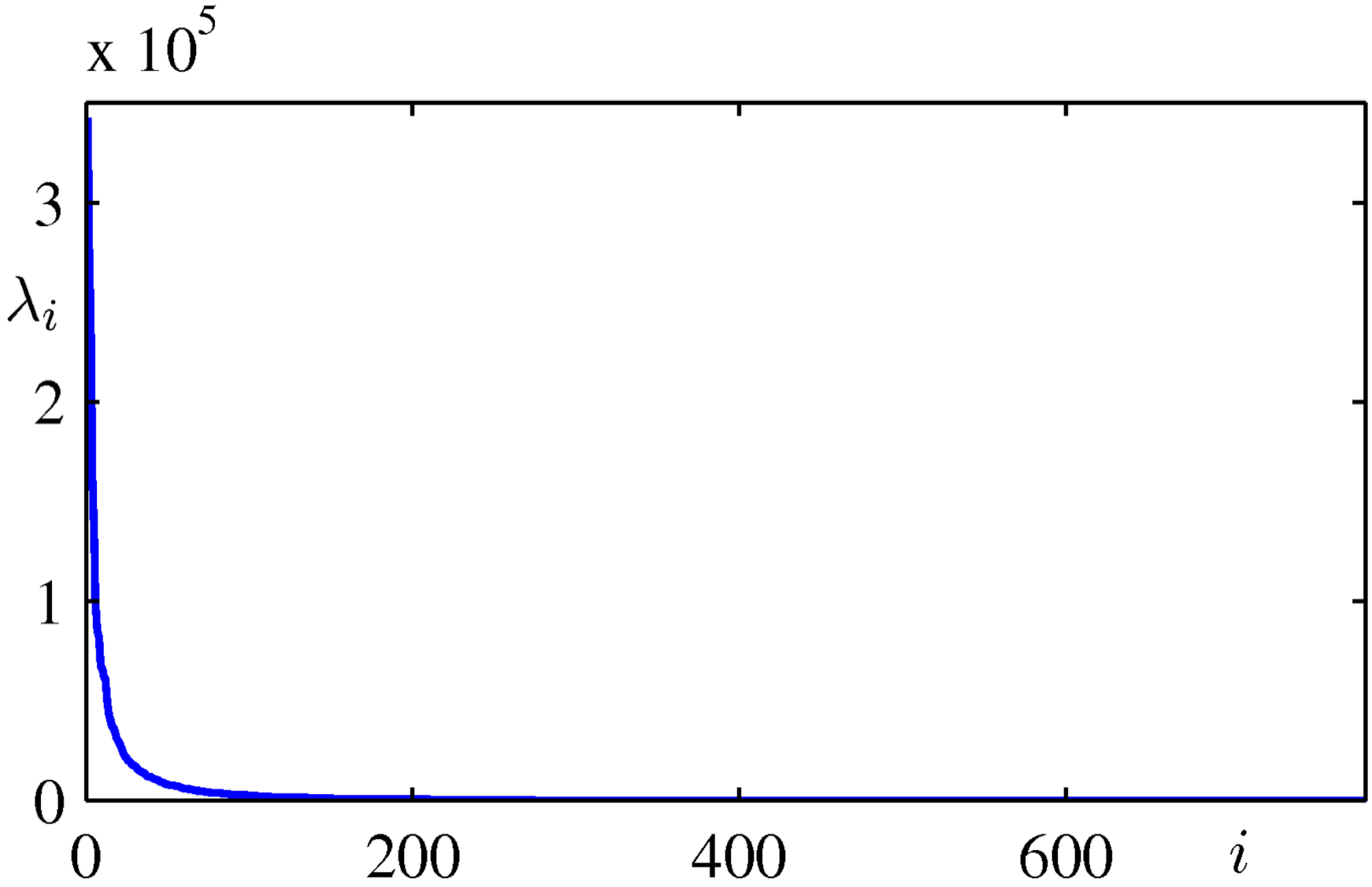
The **fraction of variance explained** $= \frac{\sum_{i=1}^{M} \lambda_i}{\sum_{i=1}^{D} \lambda_i}$

M = dimensionality of the subspace

D = dimensionality of the original data space

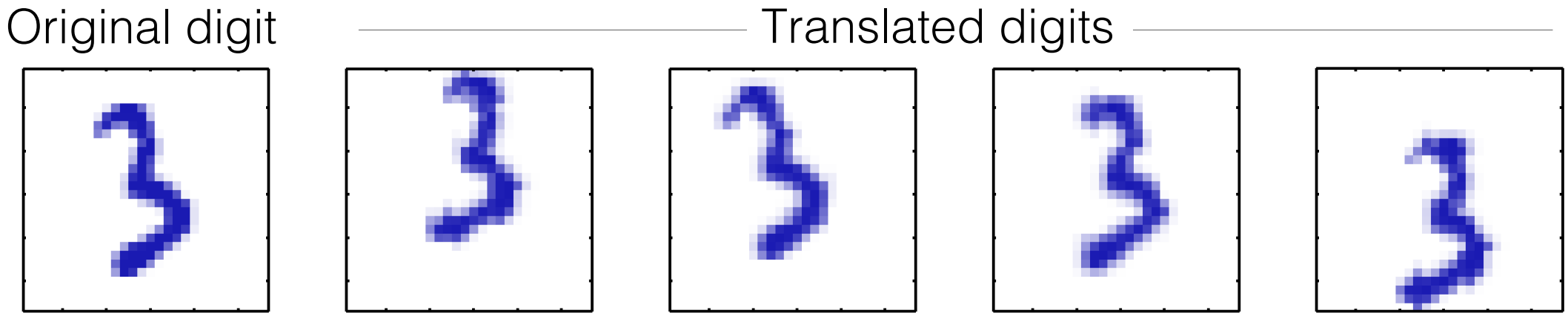
The more principle components included, the more of the variance will be represented in the projected data

Eigenvalues by principal component i



Bishop, Pattern Recognition, 2006

Example: translated digits



Types of translation:
1. Horizontal translation
2. Vertical translation
3. Rotation

Original digits: 64 x 64 pixels

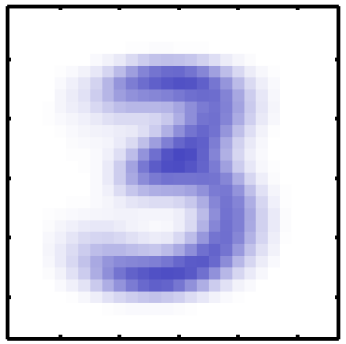
New size: 100 x 100 pixels

Bishop, Pattern Recognition, 2006

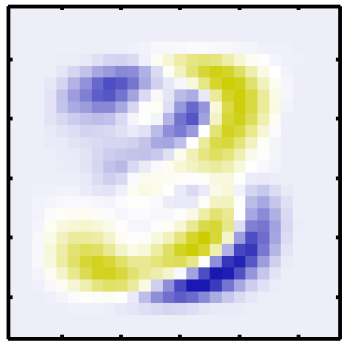
Example: translated digits

Examples of first four principle component eigenvectors and eigenvalues:

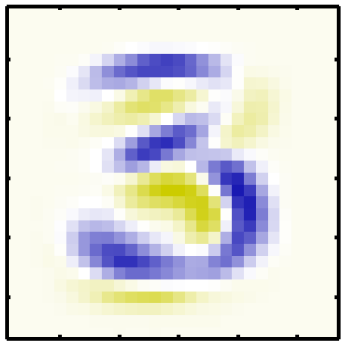
Mean



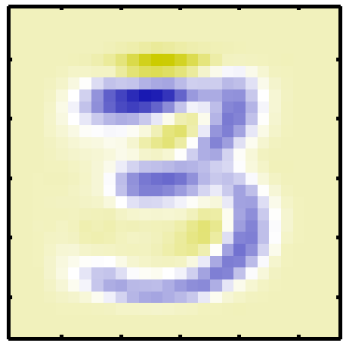
$\lambda_1 = 3.4 \cdot 10^5$



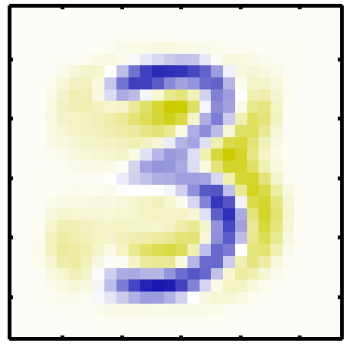
$\lambda_2 = 2.8 \cdot 10^5$



$\lambda_3 = 2.4 \cdot 10^5$



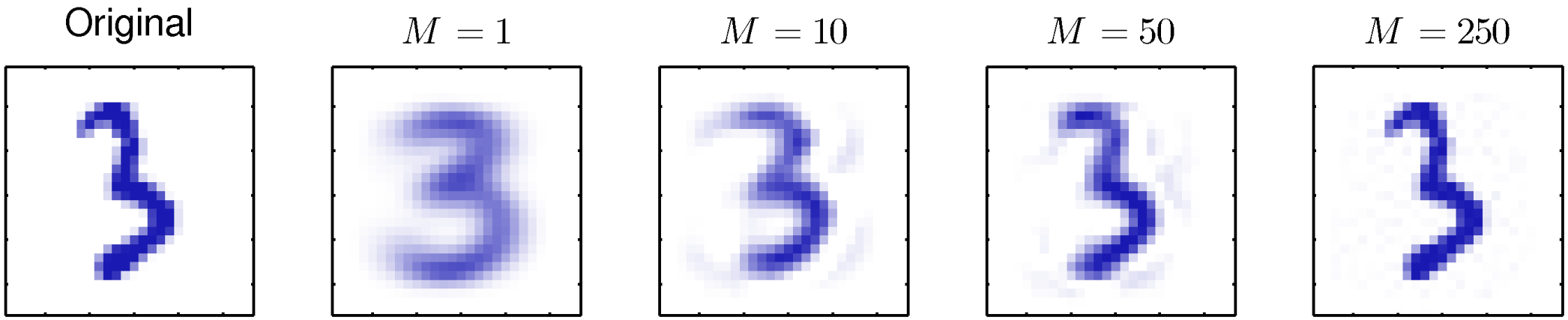
$\lambda_4 = 1.6 \cdot 10^5$



Bishop, Pattern Recognition, 2006

Example: translated digits

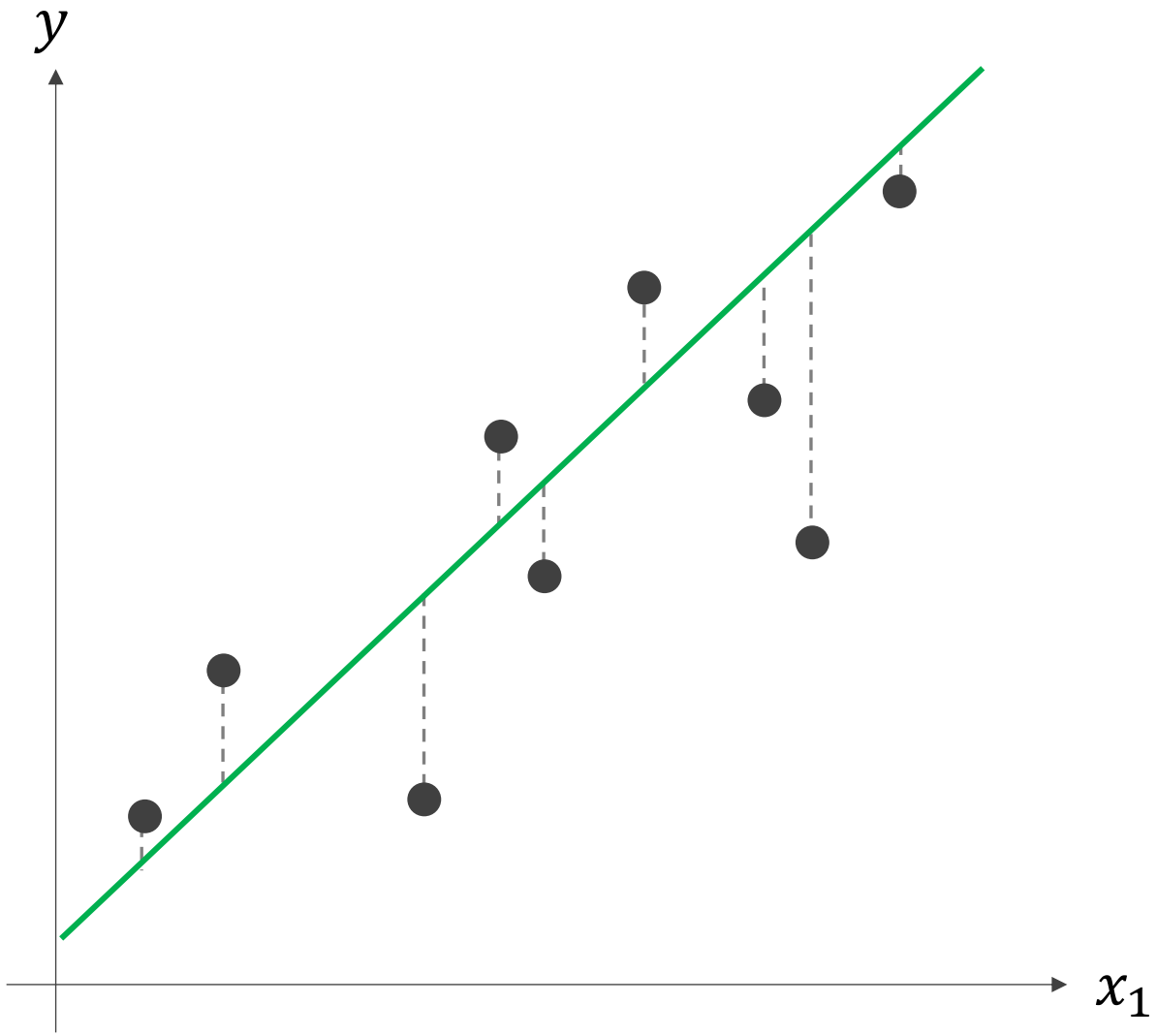
Reconstructed examples using different numbers of principal components:



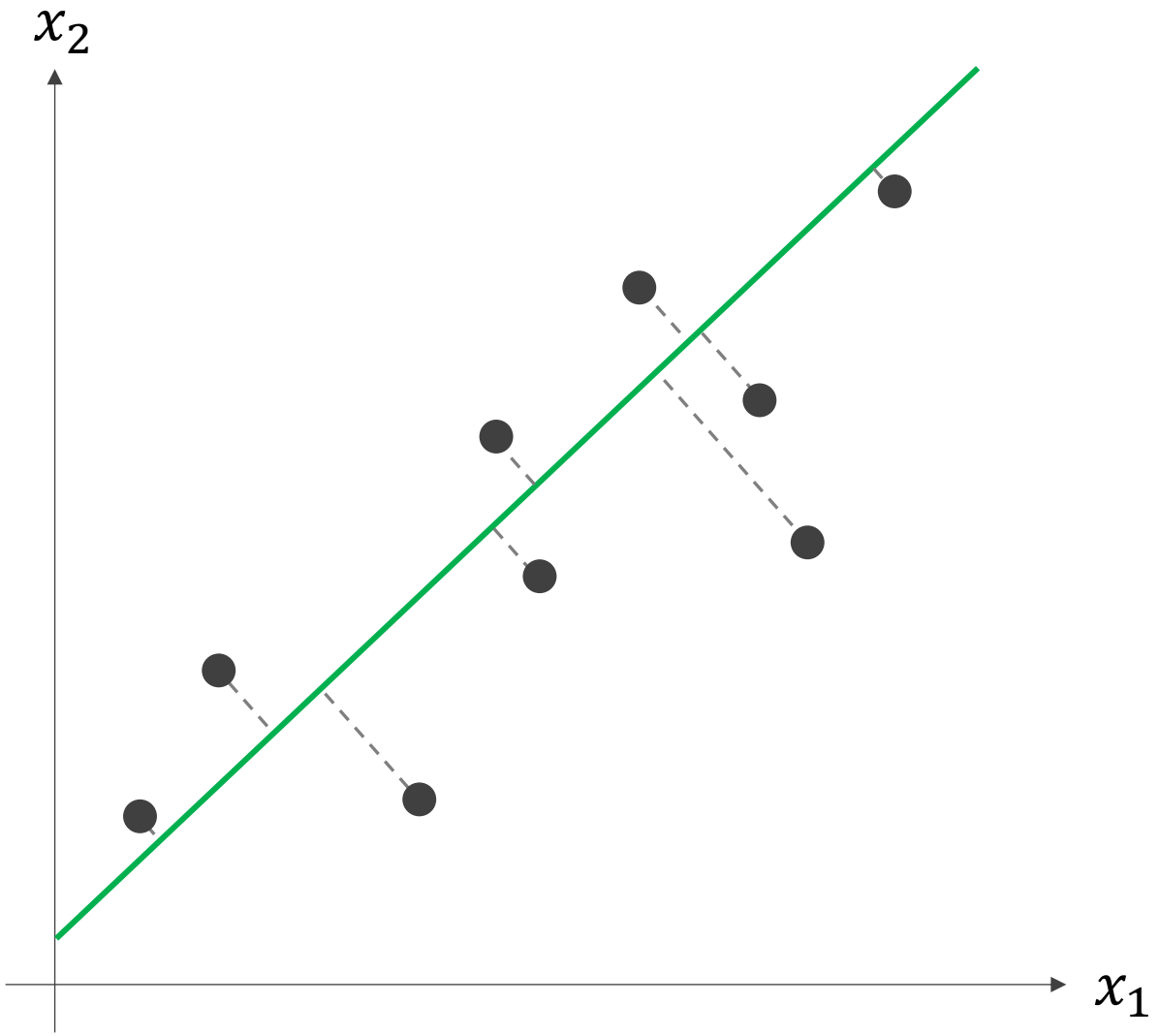
Bishop, Pattern Recognition, 2006

Relationship between the objective for least squares and PCA

Linear regression



First principal component



Extracting principal components

size | example

0 **Goal**: reduce the dimensionality of our data from D to M, where $M < D$

columns = features (D)

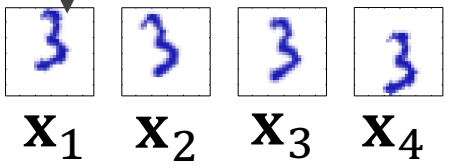
$$\mathbf{X} = \begin{bmatrix} x_{11} & \cdots & x_{1D} \\ \vdots & \ddots & \vdots \\ x_{N1} & \cdots & x_{ND} \end{bmatrix}$$ rows = observations (N) $[N \times D]$

1 Normalize each feature to mean zero and a standard deviation of 1

Each observation as a vector:

$\mathbf{x}_i \quad i = 1, \ldots, D$ $[D \times 1]$

Each pixel represents a feature



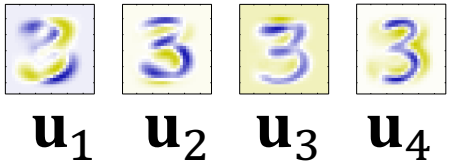
$\mathbf{x}_1$ $\mathbf{x}_2$ $\mathbf{x}_3$ $\mathbf{x}_4$

2 Determine the principal components

Calculate the eigenvectors and eigenvalues of the data covariance matrix, $\boldsymbol{\Sigma}$

Eigenvectors in descending order of their eigenvalues are the principal components

$\mathbf{u}_i$ eigenvectors / principal components $[D \times 1]$



$\mathbf{u}_1$ $\mathbf{u}_2$ $\mathbf{u}_3$ $\mathbf{u}_4$

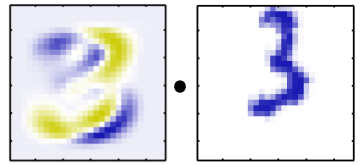
λ_i eigenvalues (how much of the variance is explained) [scalar]

$i = 1, \ldots, D$

3 Project the data features on the principal components

$$z_{ij} = \mathbf{u}_j^T \mathbf{x}_i \qquad \begin{matrix} j = 1, \ldots, D \\ i = 1, \ldots, N \end{matrix}$$ [scalar]

$\mathbf{u}_1 \cdot \mathbf{x}_1 = z_{11}$



$= 2.43$

$[D \times 1]$ $[D \times 1]$ [scalar]

4 Keep the top M principal components to reduce into a lower dimension

$$\mathbf{A} = [\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_M] \quad [D \times M]$$

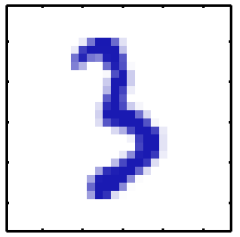
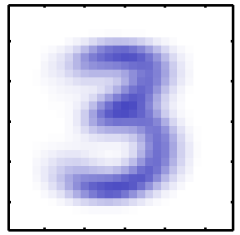
$$\mathbf{z}_i = \mathbf{A}^T \mathbf{x}_i \qquad i = 1, \ldots, N$$

Images from Bishop, Pattern Recognition, 2006

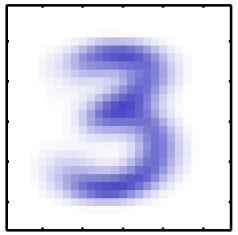
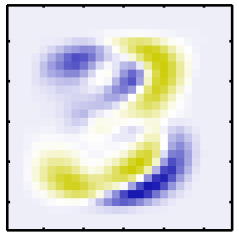
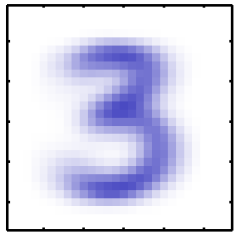
Reconstructing our data from principal components

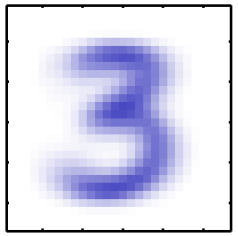
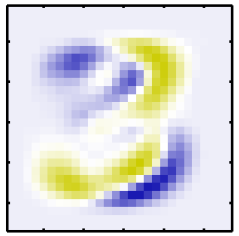
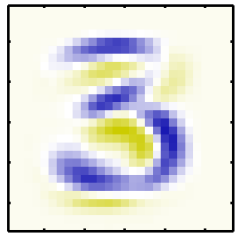
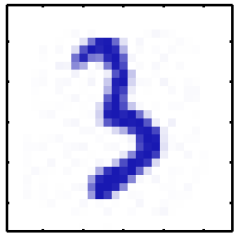
Sum the product of our projected data, $\mathbf{z}_i$, and our principle components

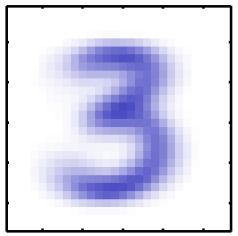
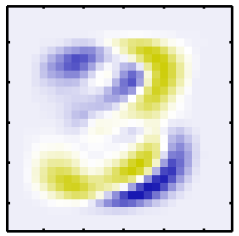
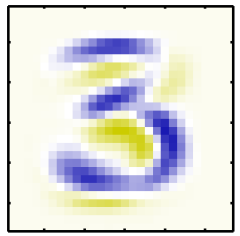
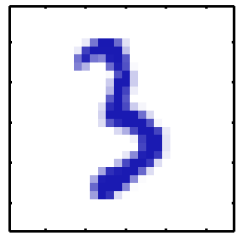
$$\hat{\mathbf{x}}_i = \sum_{j=1}^{M} z_{ij}\mathbf{u}_j$$

Example: the i[th] observation: $\mathbf{x}_i =$  $\bar{\mathbf{x}} =$ 

PCA-projected data: $\mathbf{z}_i = [z_{i1}, z_{i2}, \ldots, z_{iM}]$

$M = 1$ $\hat{\mathbf{x}}_i =$ $\bar{\mathbf{x}}$  $+z_{i1} \cdot$ $\mathbf{u}_1$  $=$ 

$M = 250$ $\hat{\mathbf{x}}_i =$ $\bar{\mathbf{x}}$  $+z_{i1} \cdot$ $\mathbf{u}_1$  $+z_{i2} \cdot$ $\mathbf{u}_2$  $+ \sum_{j=3}^{250} z_{ij}\mathbf{u}_j =$ 

$M = 10{,}000$ $\hat{\mathbf{x}}_i =$ $\bar{\mathbf{x}}$  $+z_{i1} \cdot$ $\mathbf{u}_1$  $+z_{i2} \cdot$ $\mathbf{u}_2$  $+ \sum_{j=3}^{10{,}000} z_{ij}\mathbf{u}_j =$ 

(perfect reconstruction)

Images from Bishop, Pattern Recognition, 2006

Why PCA?

- Dimensionality reduction
- Feature extraction
- Data visualization
- Reducing feature correlation
- Lossy data compression

Other dimensionality reduction techniques

- Kernel PCA
- Random projections
- Multidimensional scaling
- Locality sensitive hashing
- Autoencoders
- Isomap
- t-SNE
- UMAP

e.g. Manifold Learning