Big Data Computing

Master's Degree in Computer Science

2019-2020

Gabriele Tolomei

Department of Computer Science

Sapienza Università di Roma

tolomei@di.uniroma1.it

Sapienza
Università di Roma

Recap from Last Lecture(s)

- High-dimensional naïve representation (i.e., feature space) of text data
- Clustering high-dimensional data may be problematic
  - Due to the curse of dimensionality
- Many other data sources (e.g., images) share the same issue
- Good news: high-dimensionality is often not real
  - Due to the way in which we observe/collect data

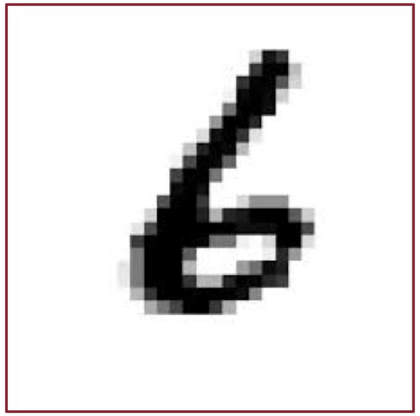
DIMENSIONALITY REDUCTION

Modeled vs. True Dimensionality

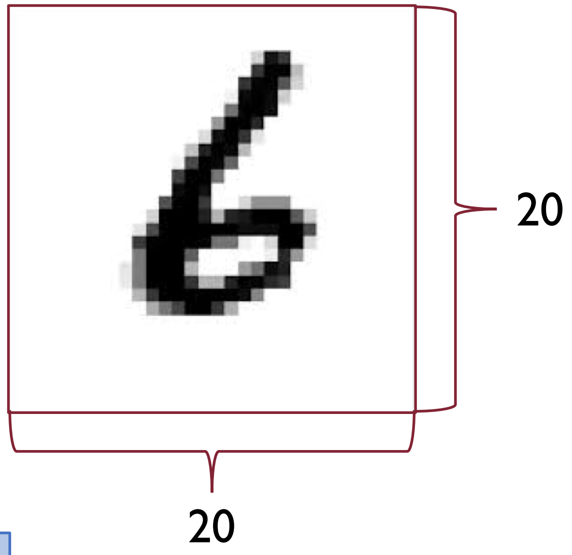
Example

Handwritten digit recognition

Modeled vs. True Dimensionality



Modeled vs. True Dimensionality

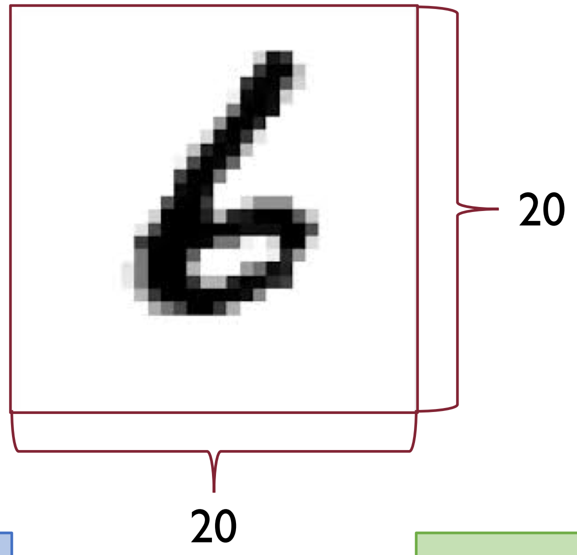


Modeled dimensionality

Each digit represented by **20x20** bitmap

400-dimensional binary vector

Modeled vs. True Dimensionality



Modeled dimensionality

Each digit represented by **20x20** bitmap

400-dimensional binary vector

True dimensionality

Actual digits just cover a tiny fraction of all this huge space

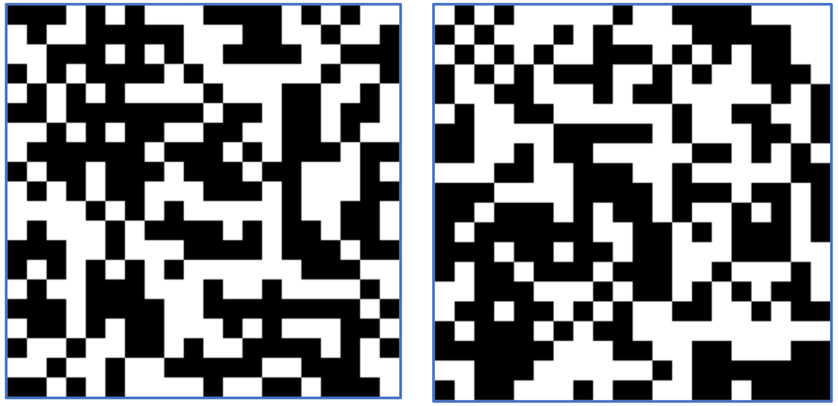
Small variations of the pen-stroke

Modeled vs. True Dimensionality

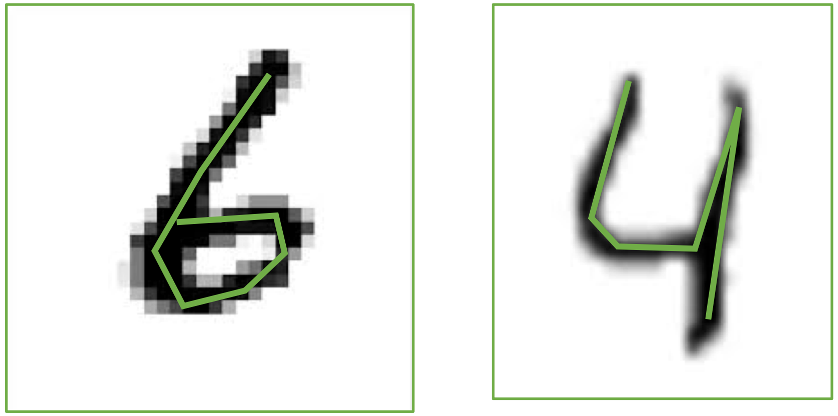
Random samples from 400-d space


Modeled vs. True Dimensionality

Random samples from 400-d space

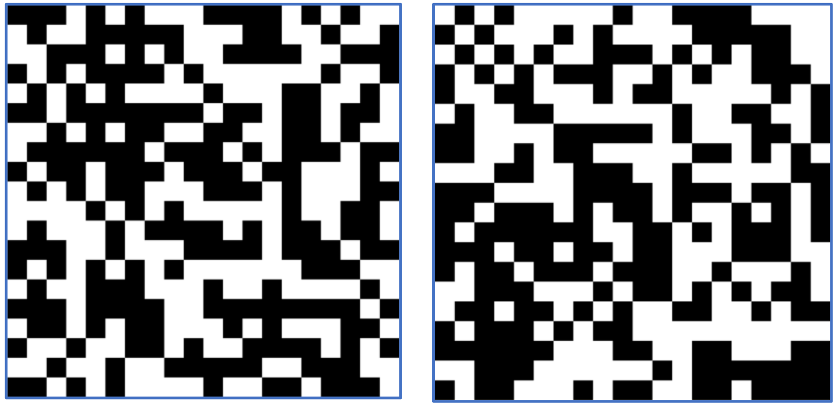


True digits living in a 400-d space

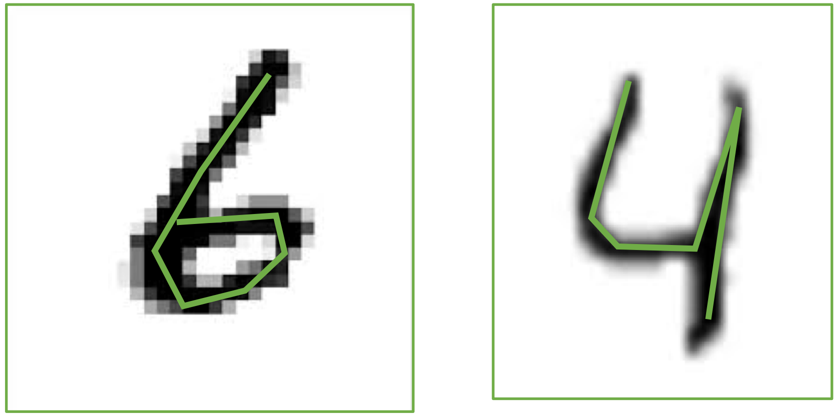


Modeled vs. True Dimensionality

Random samples from 400-d space



True digits living in a 400-d space



We model data (i.e., digits) as very high dimensional…

… In fact, they are not so

The Curse of Dimensionality

As dimensionality grows fewer examples in each region of the feature space (assuming # examples is constant)

The Curse of Dimensionality

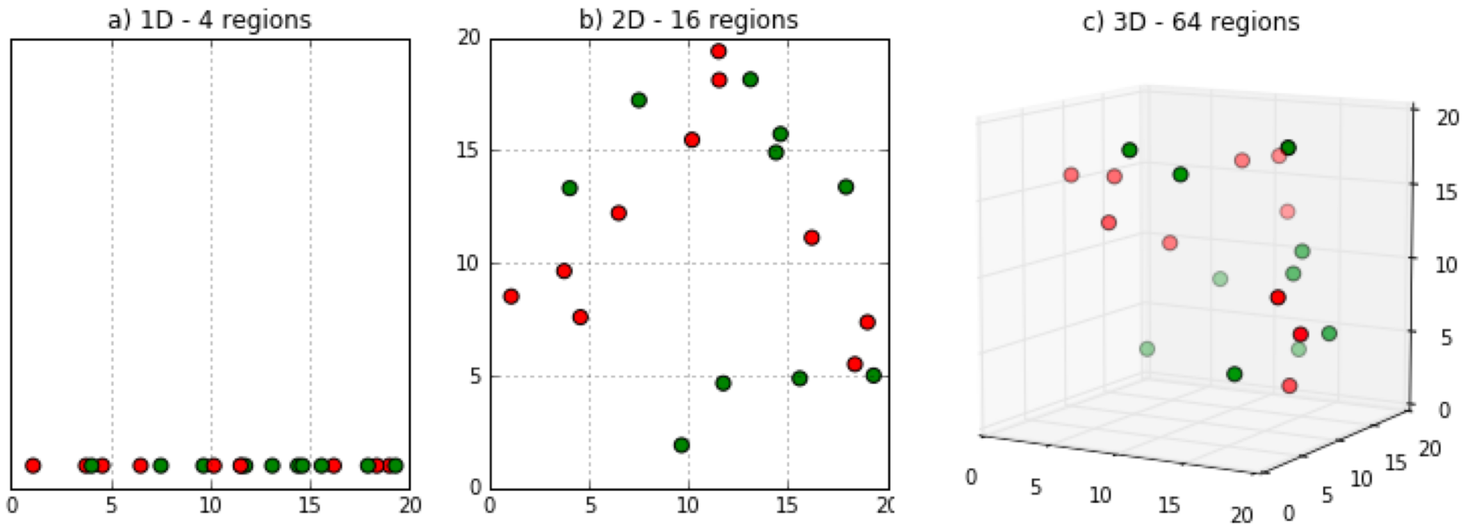
As dimensionality grows fewer examples in each region of the feature space (assuming # examples is constant)

Put it another way:
The number of examples must grow exponentially with dimensionality if we want to maintain the same "density"

The Curse of Dimensionality

As dimensionality grows fewer examples in each region of the feature space (assuming # examples is constant)

Put it another way:
The number of examples must grow exponentially with dimensionality if we want to maintain the same "density"



Dealing with High Dimensionality

3 possible approaches

Feature Engineering
(using domain knowledge)

e.g., SIFT in computer vision

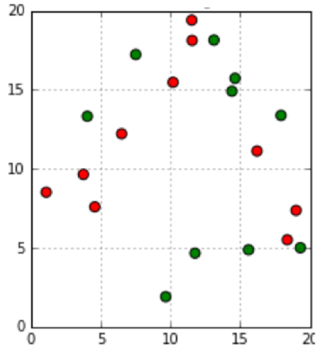
Dealing with High Dimensionality

3 possible approaches

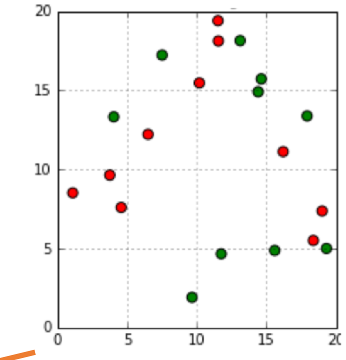
Feature Engineering
(using domain knowledge)

e.g., SIFT in computer vision

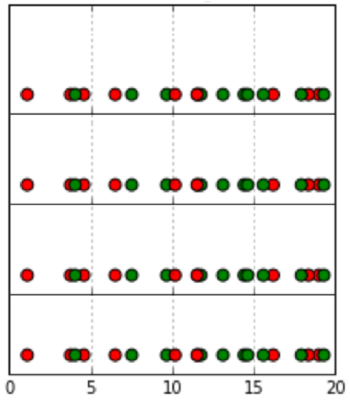
Making Assumptions



Dealing with High Dimensionality: Assumptions

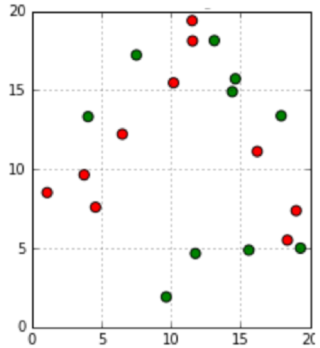


independence

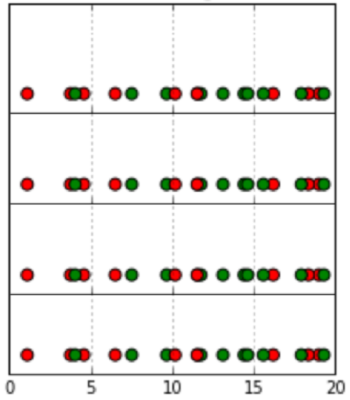


Count along each dimension separately

Dealing with High Dimensionality: Assumptions

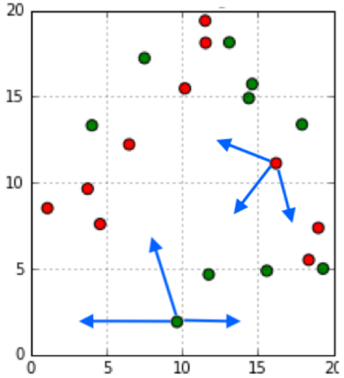


independence



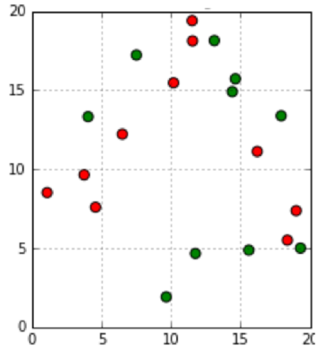
Count along each dimension separately

smoothness

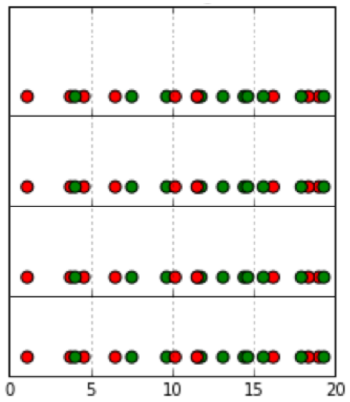


Propagate counts to neighboring regions

Dealing with High Dimensionality: Assumptions

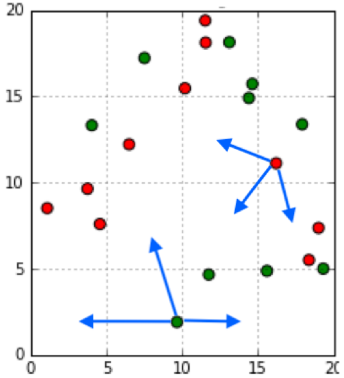


independence



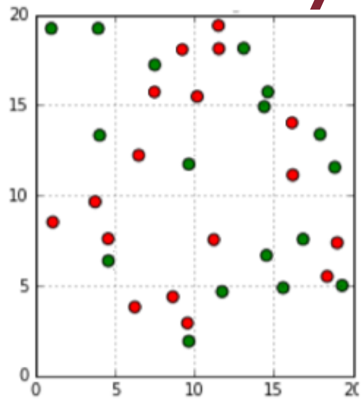
Count along each dimension separately

smoothness



Propagate counts to neighboring regions

simmetry



Invariance to the order of dimensions

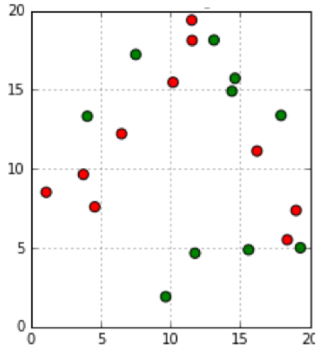
Dealing with High Dimensionality

3 possible approaches

Feature Engineering
(using domain knowledge)

e.g., SIFT in computer vision

Making Assumptions



Reduce Dimensionality

Create a new set of dimensions (i.e., variables)

Dimensionality Reduction

- A technique to unveil the actual (i.e., meaningful) dimensions of data
- A pre-processing step for representing data with fewer features
- Preserve as much "structure" of the data as possible
- Retained structure must be discriminative affecting data separability

"structure" here means **variance**

Dimensionality Reduction

2 main approaches

Dimensionality Reduction

2 main approaches

Feature Selection

Pick a subset of the original dimensions that are good predictors
(e.g., using information gain)

$x_1, x_2, \ldots, x_{j-1}, x_j, x_{j+1}, \ldots, x_{d-1}, x_d$

Dimensionality Reduction

2 main approaches

Feature Selection

Pick a subset of the original dimensions that are good predictors (e.g., using information gain)

$x_1, x_2, \ldots, x_{j-1}, x_j, x_{j+1}, \ldots, x_{d-1}, x_d$

Feature Extraction

Build a new set of $k < d$ dimensions as a (linear) combination of the originals

$e_1, e_2, \ldots, e_k$

$e_i = f(x_1, x_2, \ldots, x_d)$

Dimensionality Reduction

2 main approaches

Feature Selection

Pick a subset of the original dimensions that are good predictors (e.g., using information gain)

$x_1, x_2, \ldots, x_{j-1}, x_j, x_{j+1}, \ldots, x_{d-1}, x_d$

Feature Extraction

Build a new set of $k < d$ dimensions as a (linear) combination of the originals

$e_1, e_2, \ldots, e_k$

$e_i = f(x_1, x_2, \ldots, x_d)$

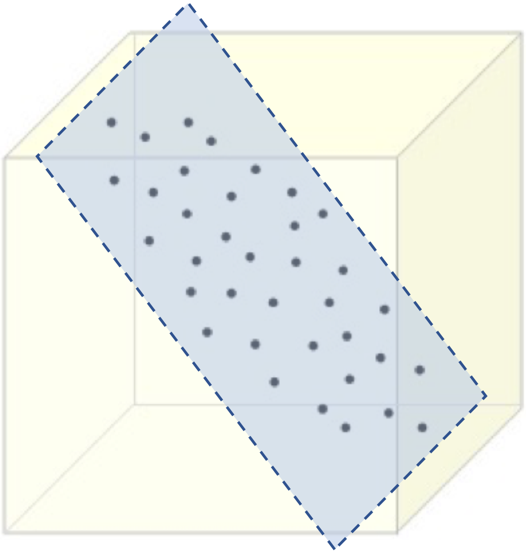
Principal Component Analysis (PCA)

Dimensionality reduction technique based on feature extraction

High-dimensional data is in fact embedded into some lower dimensional space

Example

A 3-d set of points embedded into a 2-d hyperplane

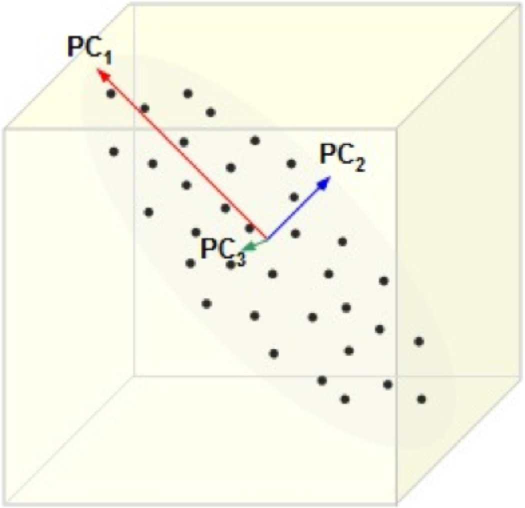


Principal Component Analysis (PCA)

PCA defines a set of principal components as follows:

- 1st: direction of the greatest variance of data
- 2nd: perpendicular to 1st and greatest variance of what's left
- … and so on until d

The top k < d components become the new dimensions



e.g., k = 2

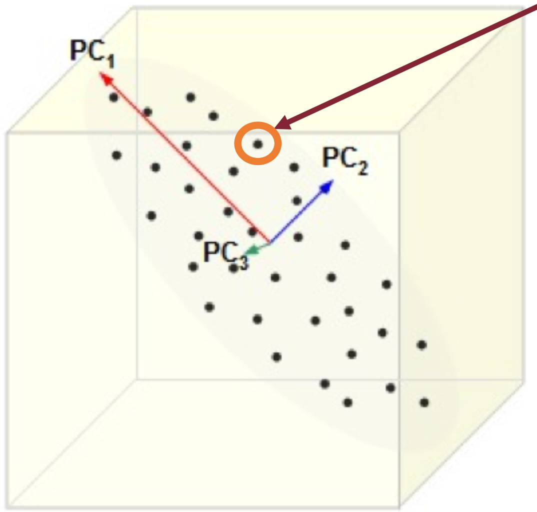
Principal Component Analysis (PCA)

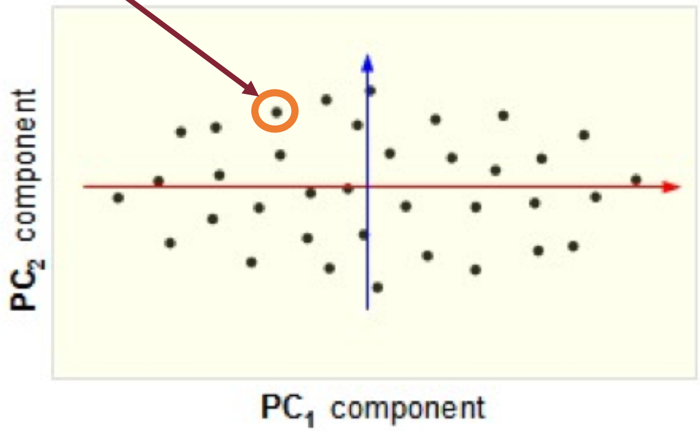
PC_1 and PC_2 are the top-2 principal components

Principal Component Analysis (PCA)

PC_1 and PC_2 are the top-2 principal components

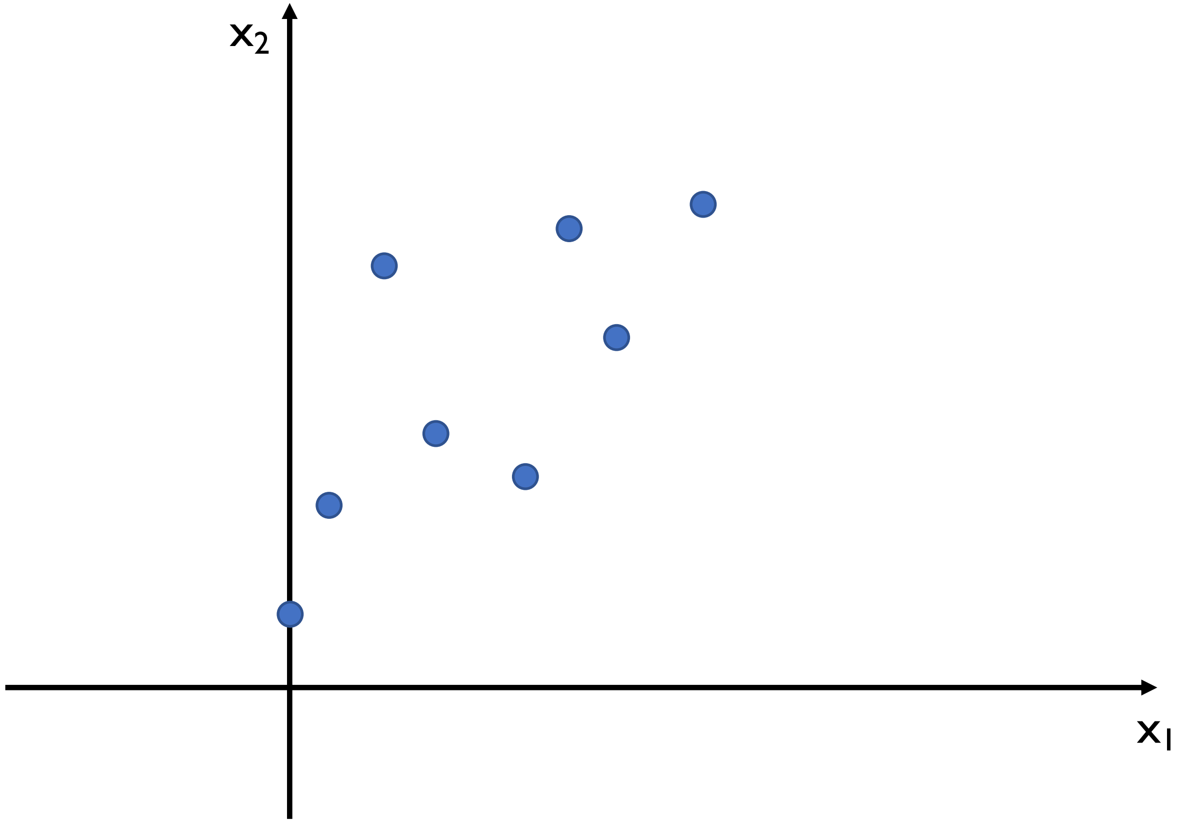
Change the coordinates of every point according to the new dimensions





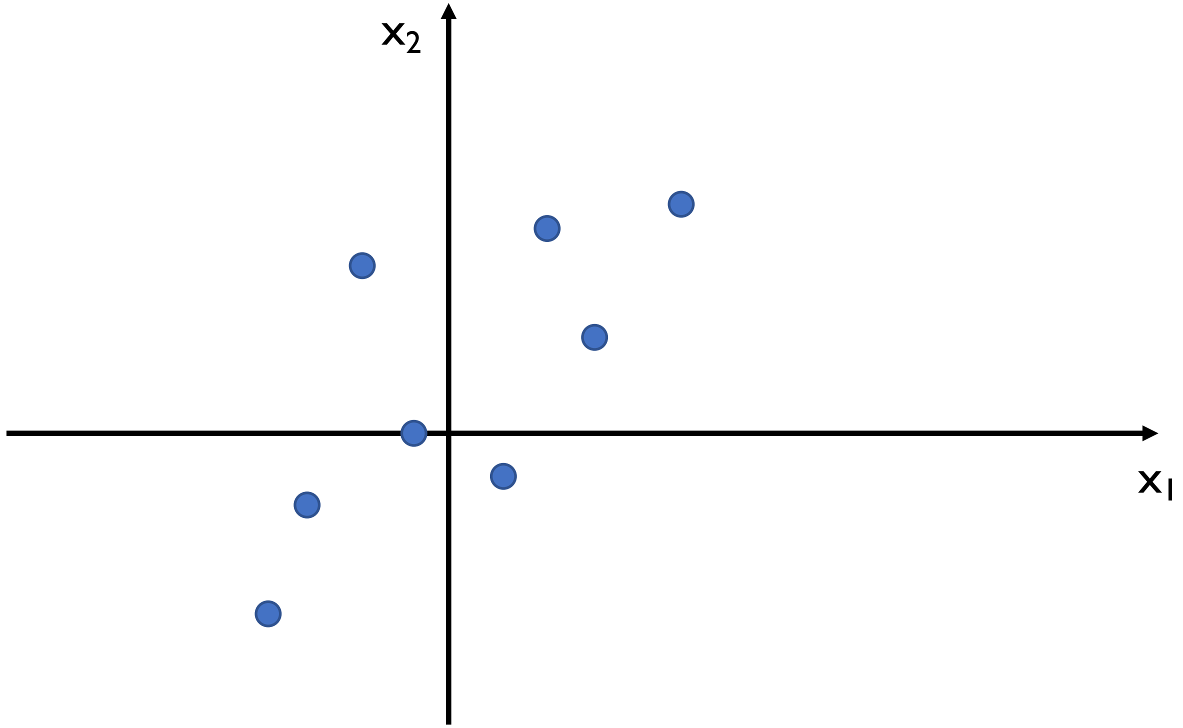
Why Do We Look for Greatest Variance?

Example: Reduce 2-dimensional data to 1-d



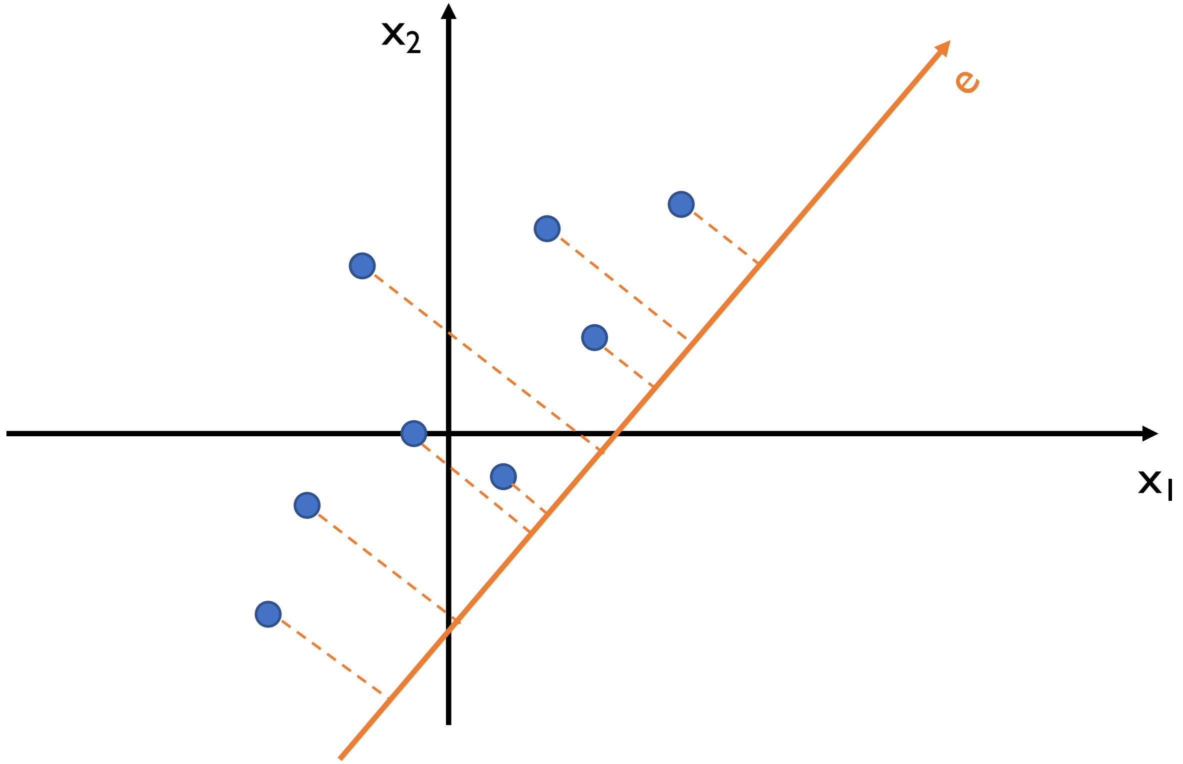
Why Do We Look for Greatest Variance?

First of all, let's center the points around the mean along x_1 and x_2



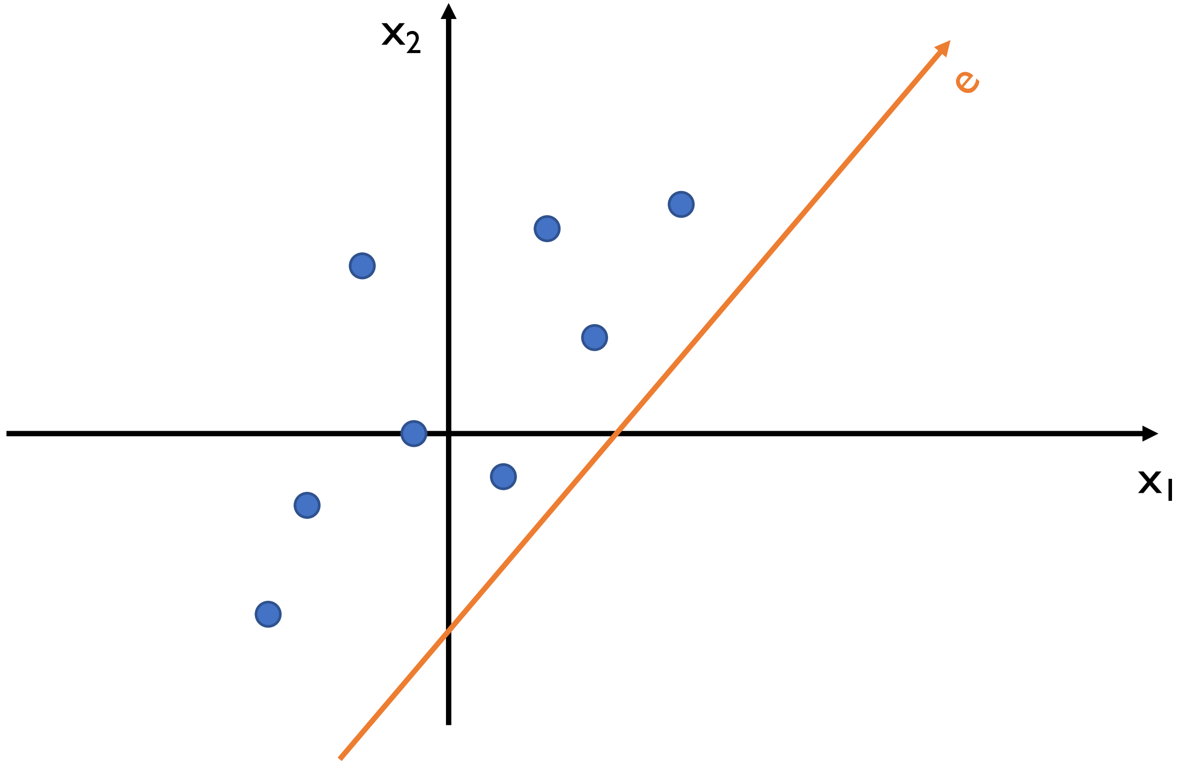
Why Do We Look for Greatest Variance?

Map, i.e., project (x_1, x_2) to a new single dimension axis e



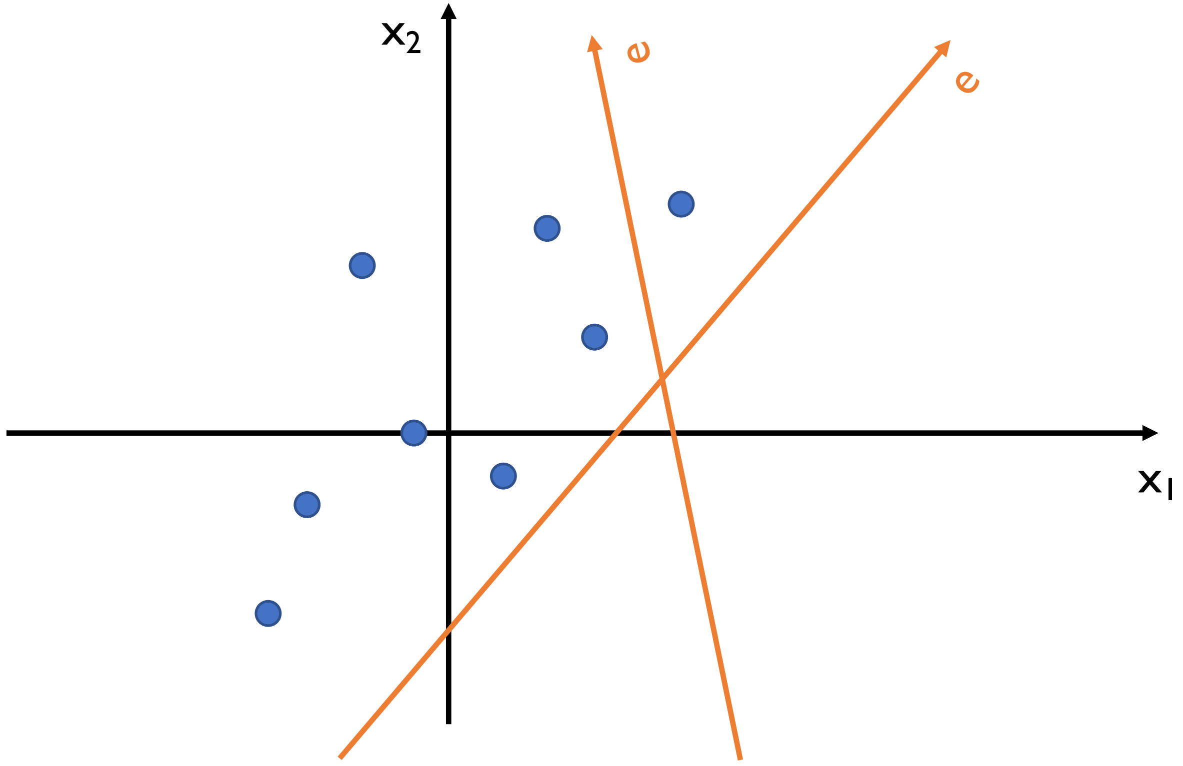
Why Do We Look for Greatest Variance?

Map, i.e., project (x_1, x_2) to a new single dimension axis e



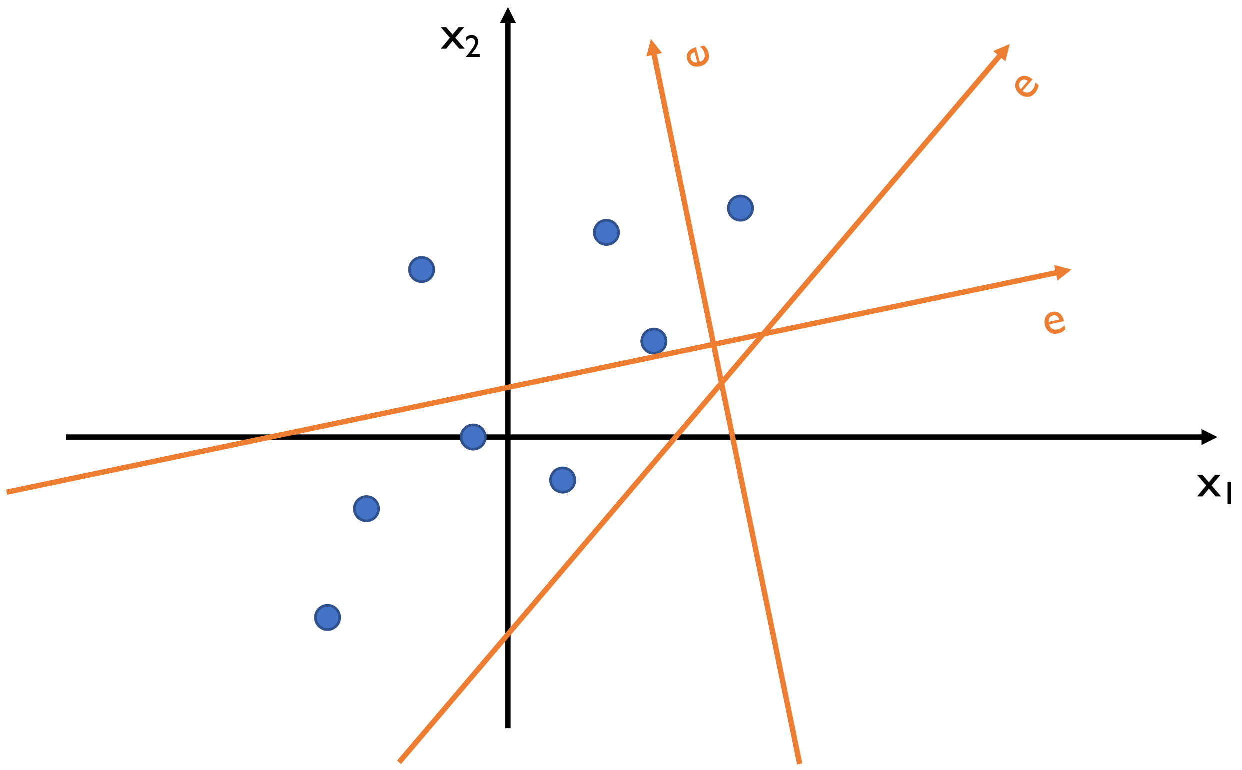
Why Do We Look for Greatest Variance?

Map, i.e., project (x_1, x_2) to a new single dimension axis e



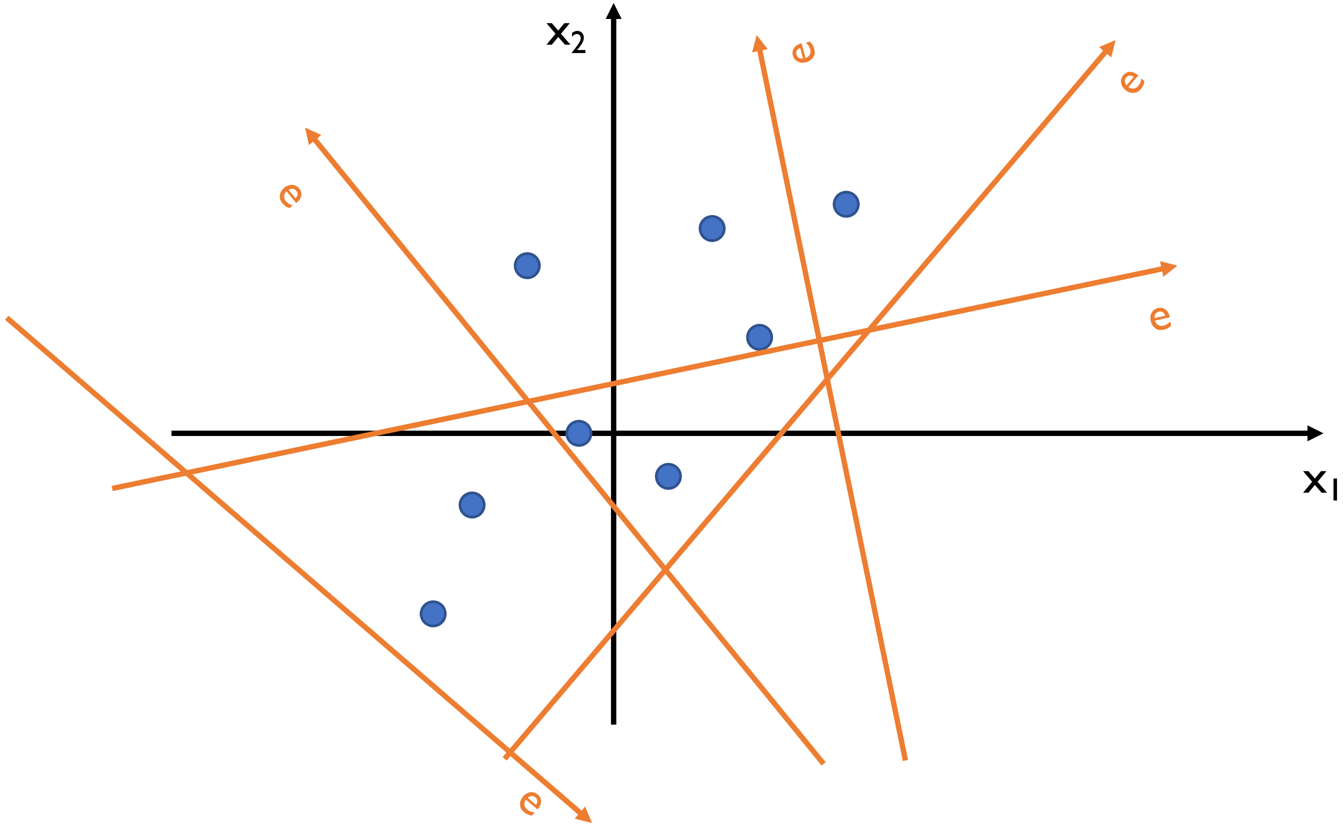
Why Do We Look for Greatest Variance?

Map, i.e., project (x_1, x_2) to a new single dimension axis e



Why Do We Look for Greatest Variance?

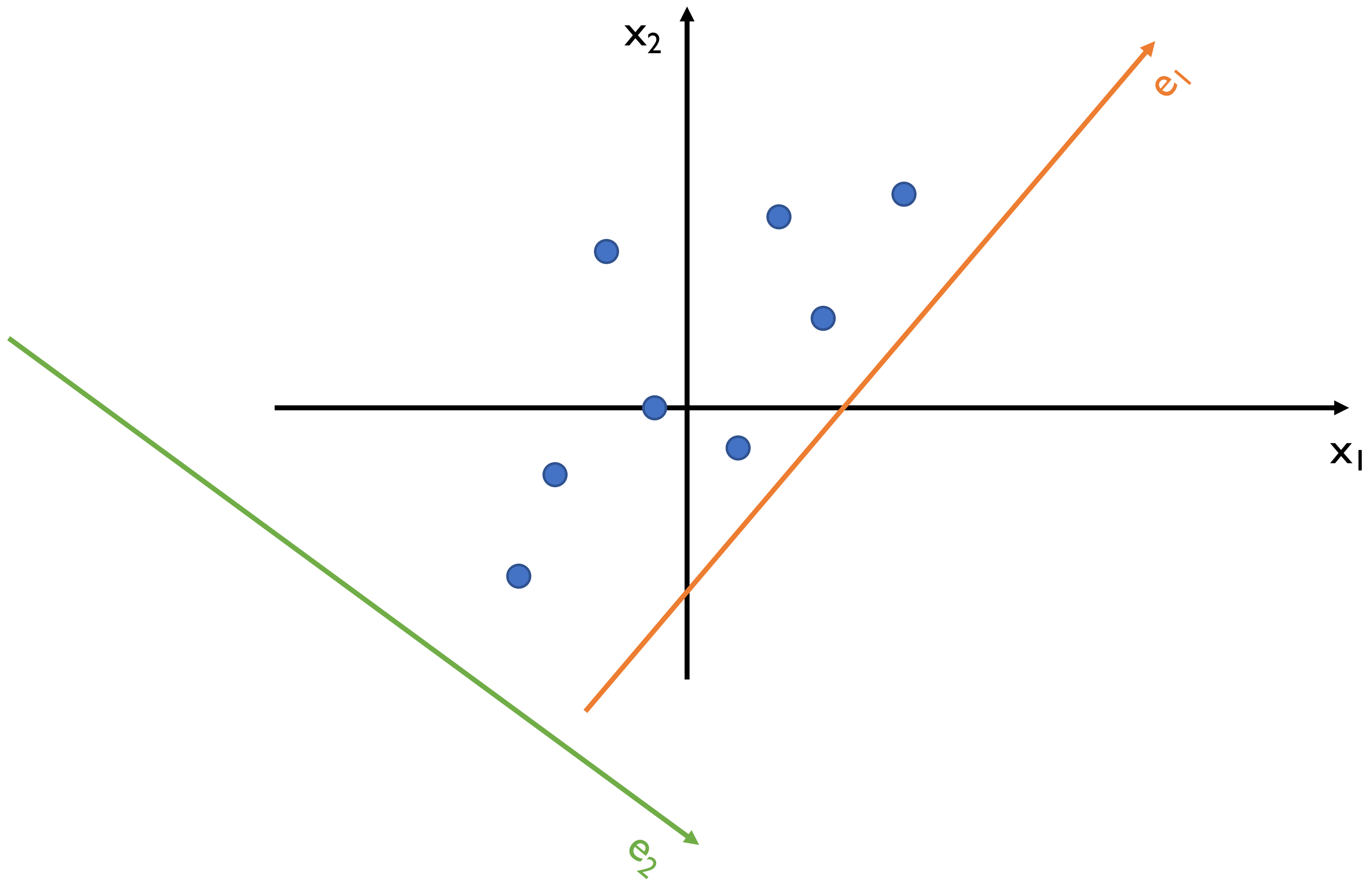
Map, i.e., project (x_1, x_2) to a new single dimension axis e



infinite many mappings from (x_1, x_2) to a new axis e

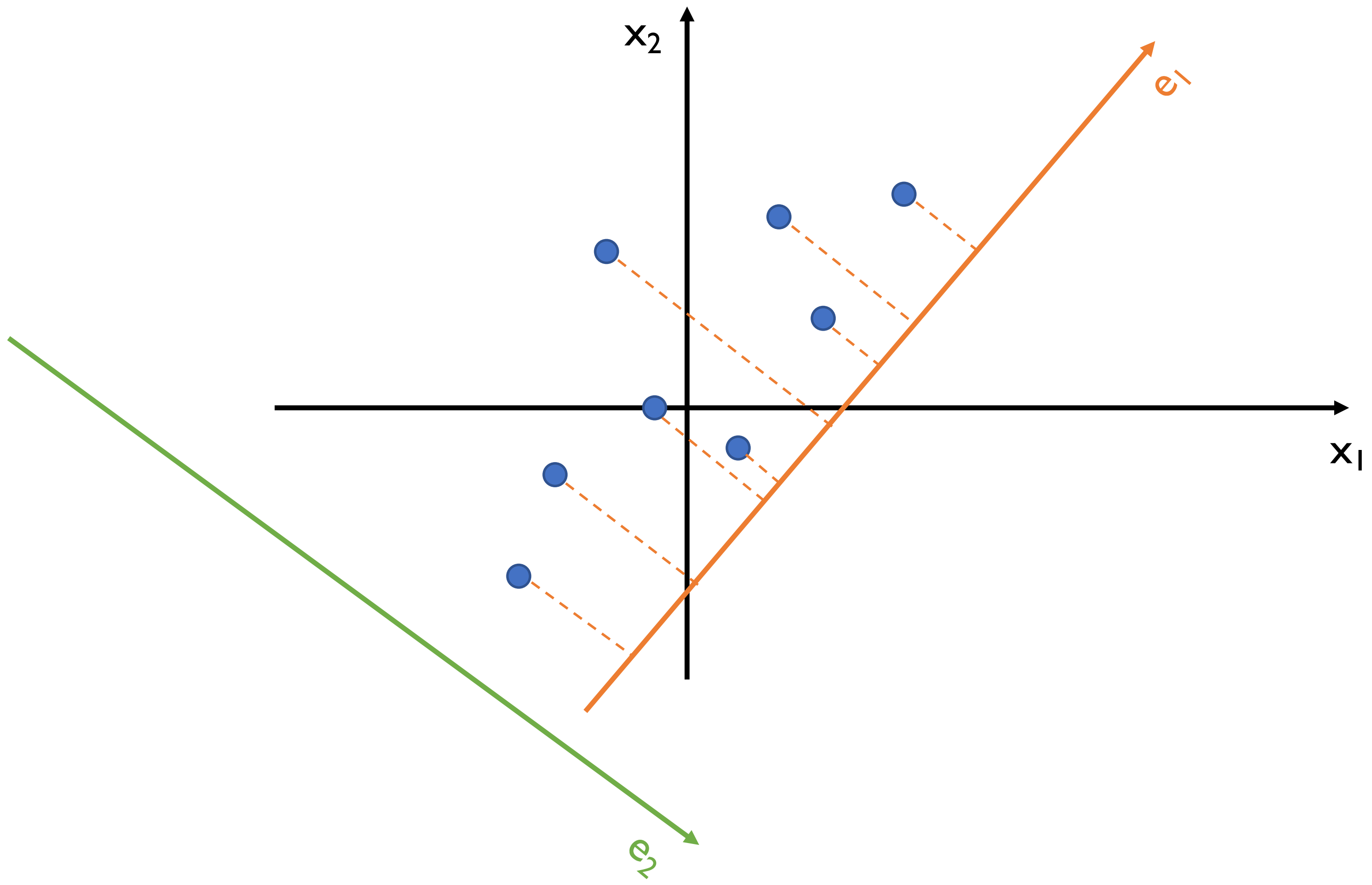
Why Do We Look for Greatest Variance?

Let's consider 2 different mappings e_1 and e_2



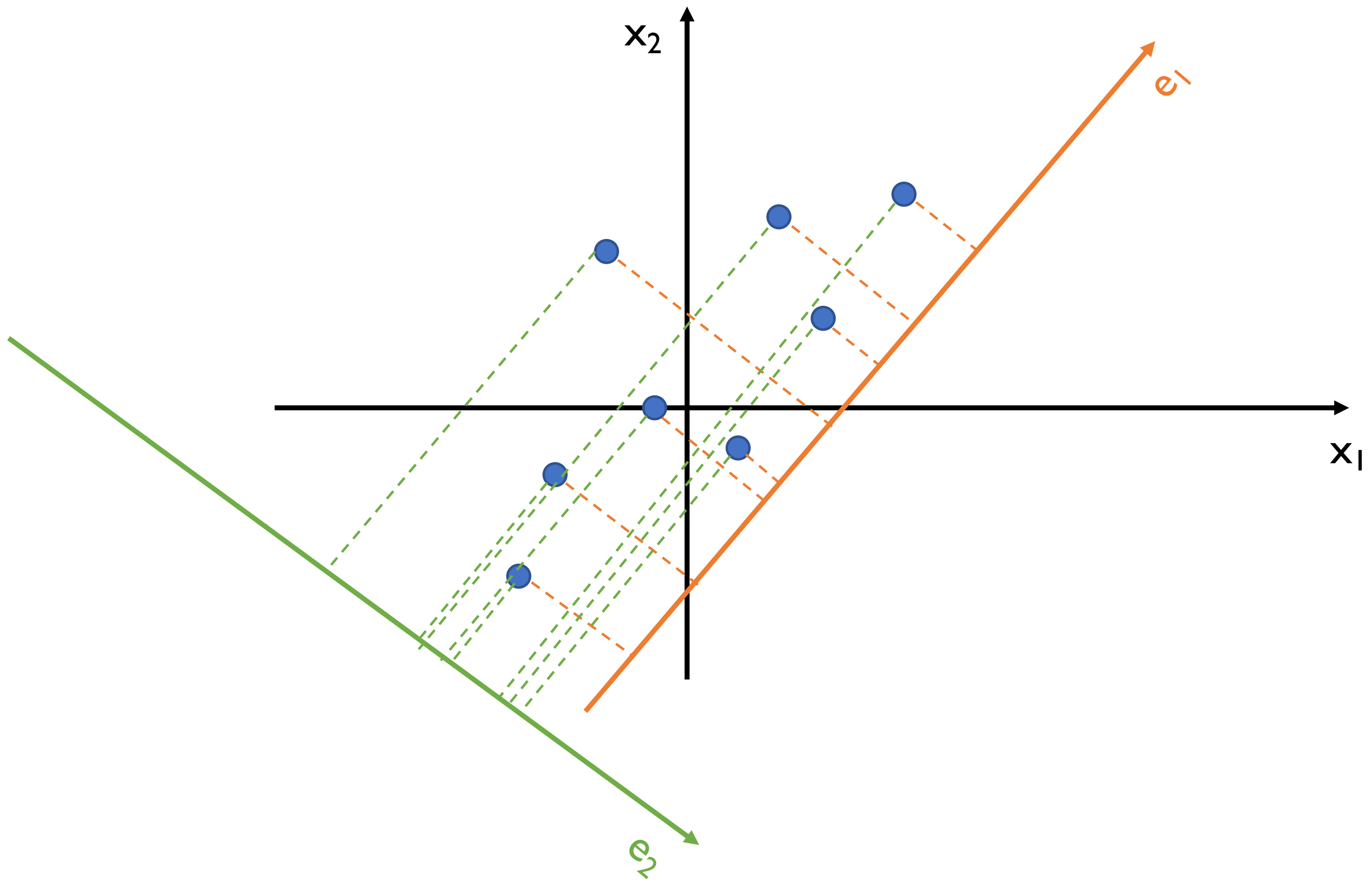
Why Do We Look for Greatest Variance?

Let's consider 2 different mappings e_1 and e_2



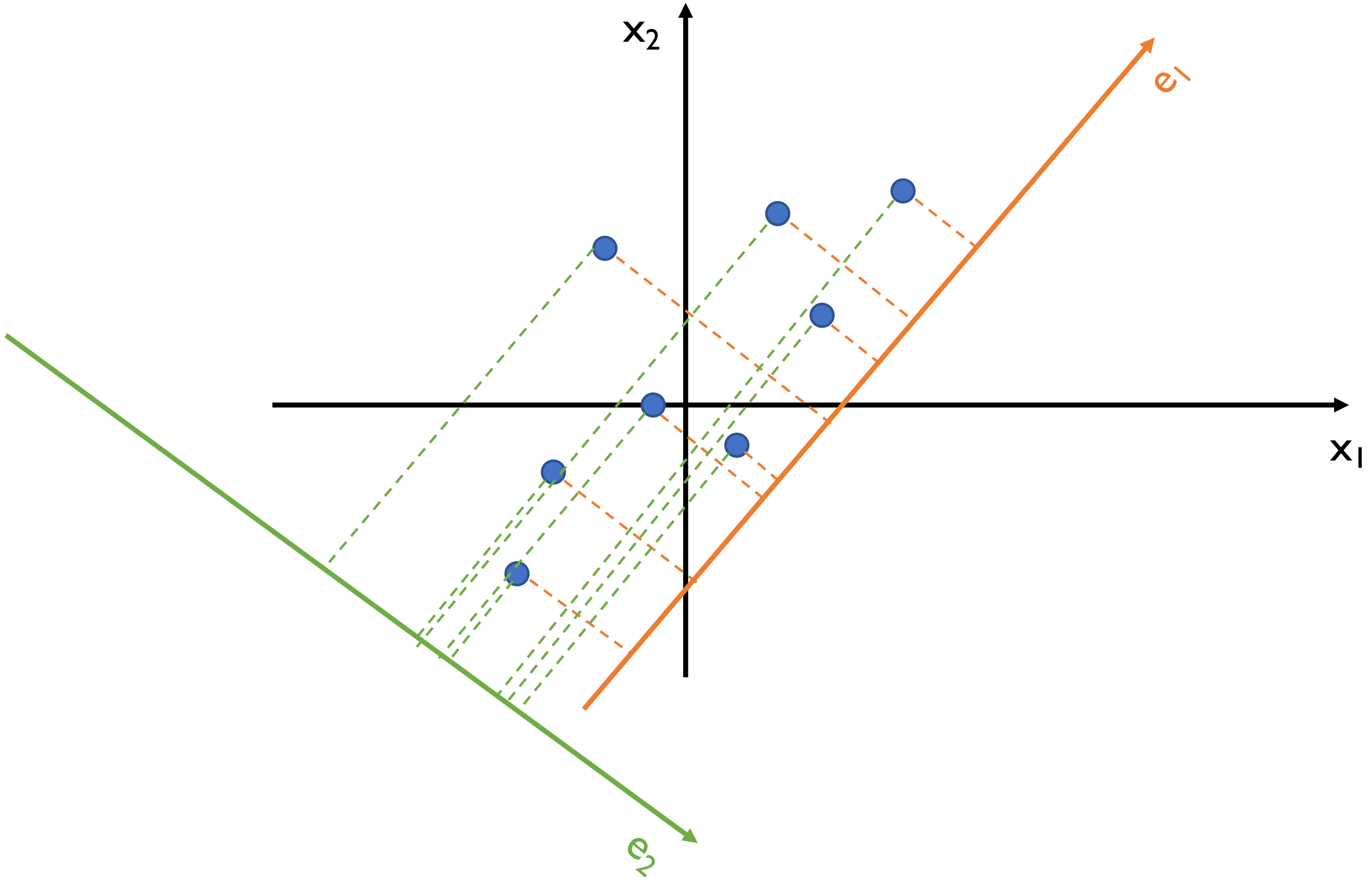
Why Do We Look for Greatest Variance?

Let's consider 2 different mappings e_1 and e_2



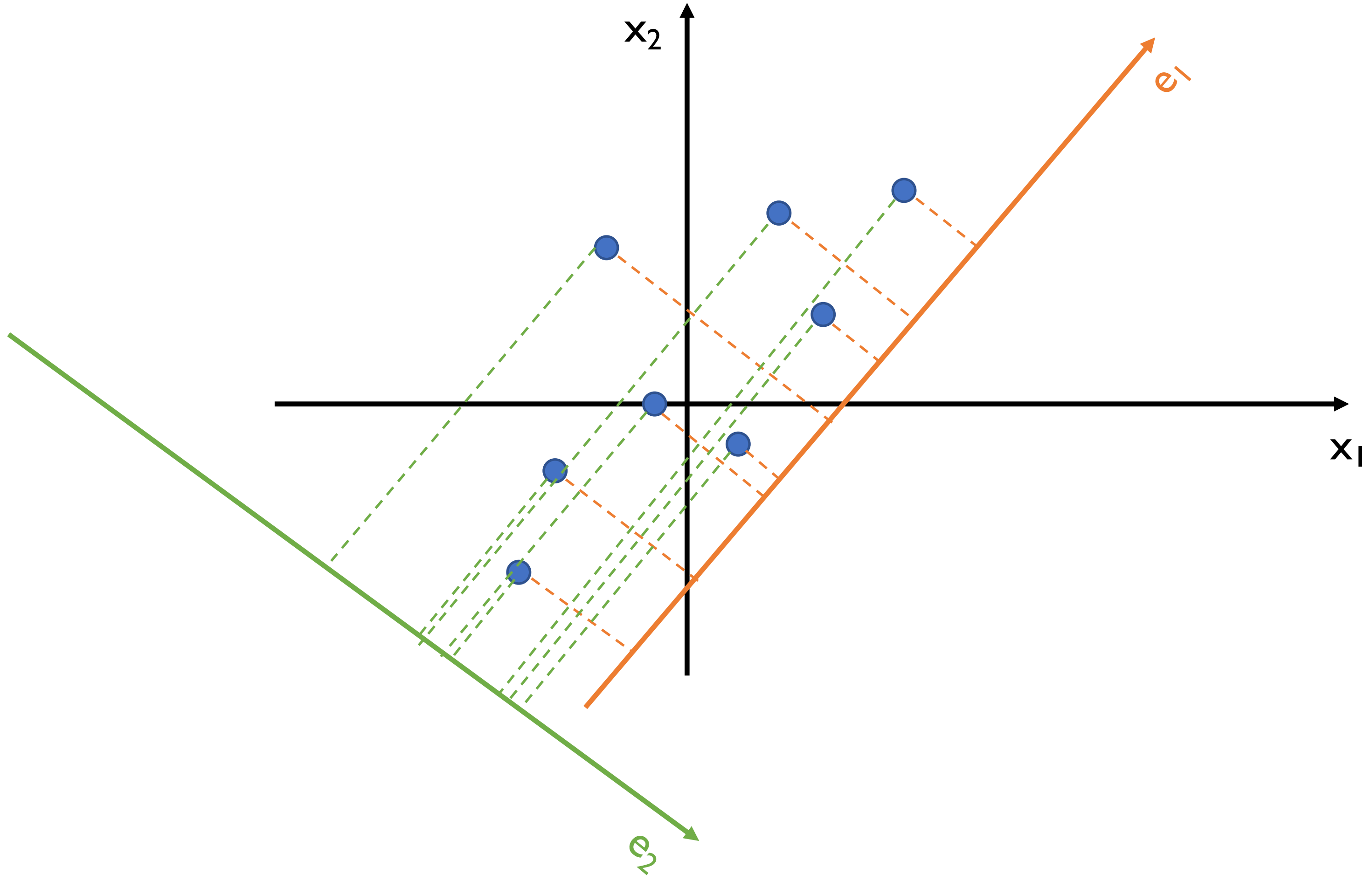
Why Do We Look for Greatest Variance?

Which one is better?



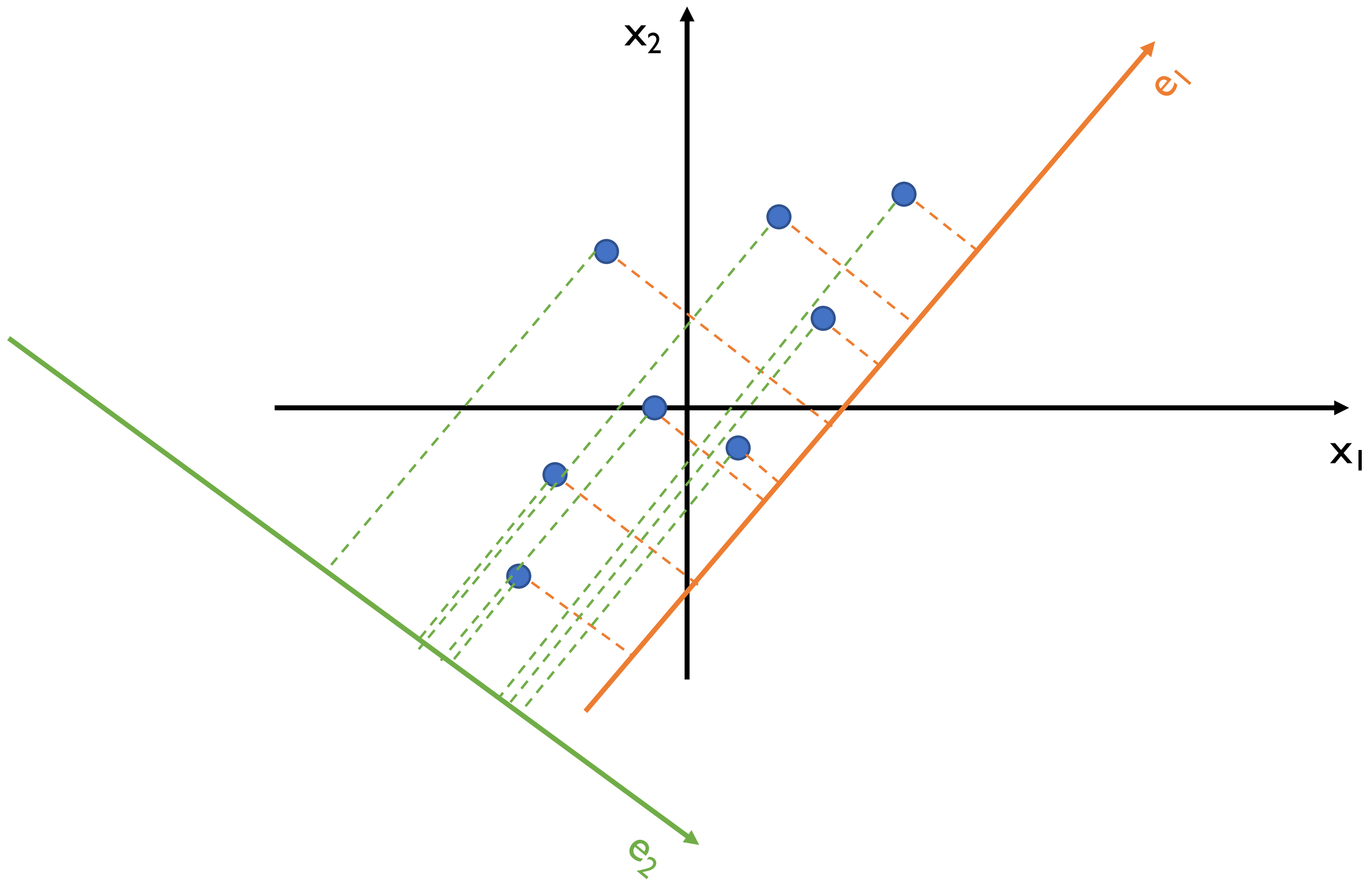
Why Do We Look for Greatest Variance?

Points projected onto e_1 look more **spread-out** than onto e_2



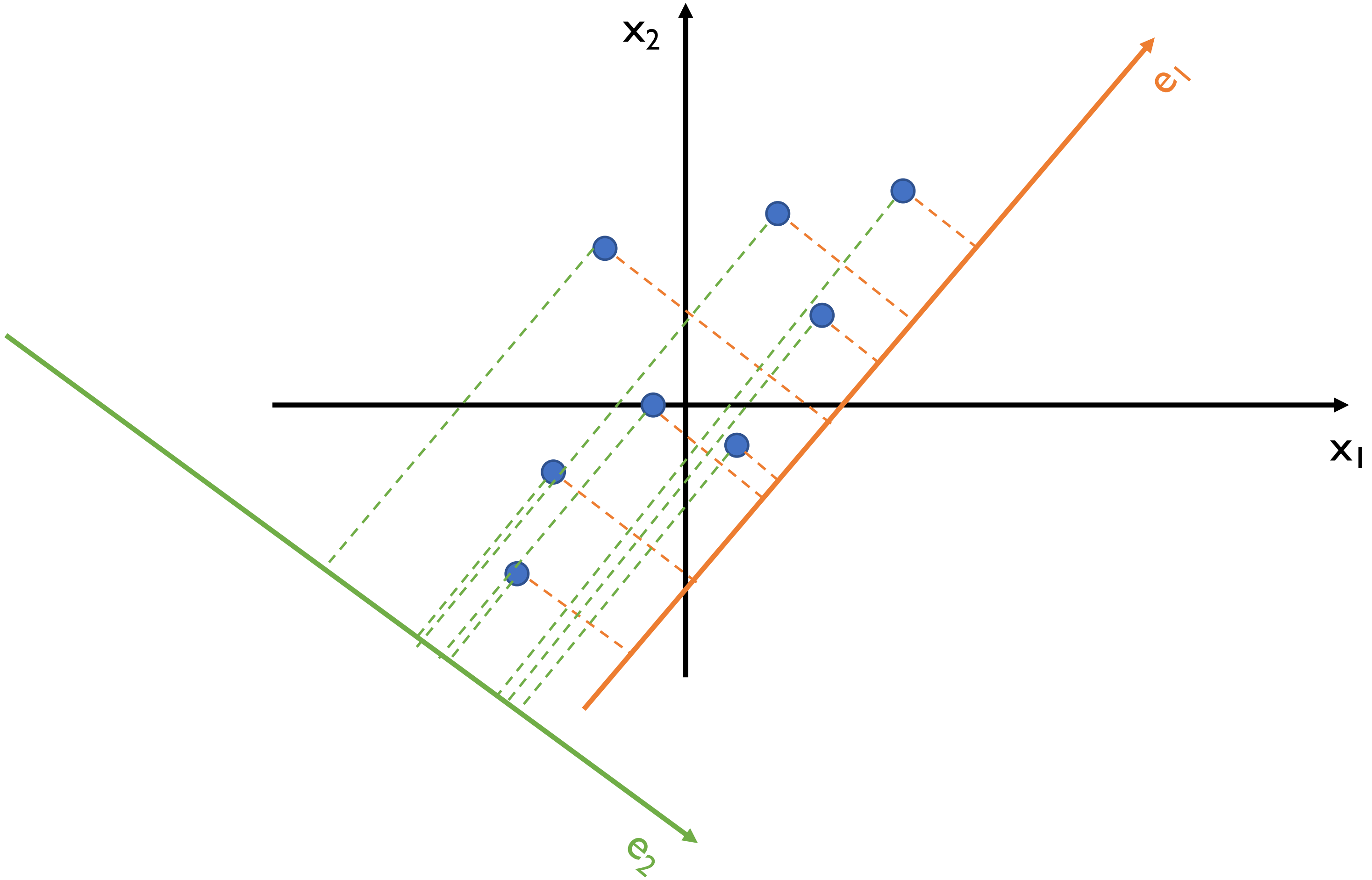
Why Do We Look for Greatest Variance?

The **variance** along e_1 is larger than along e_2



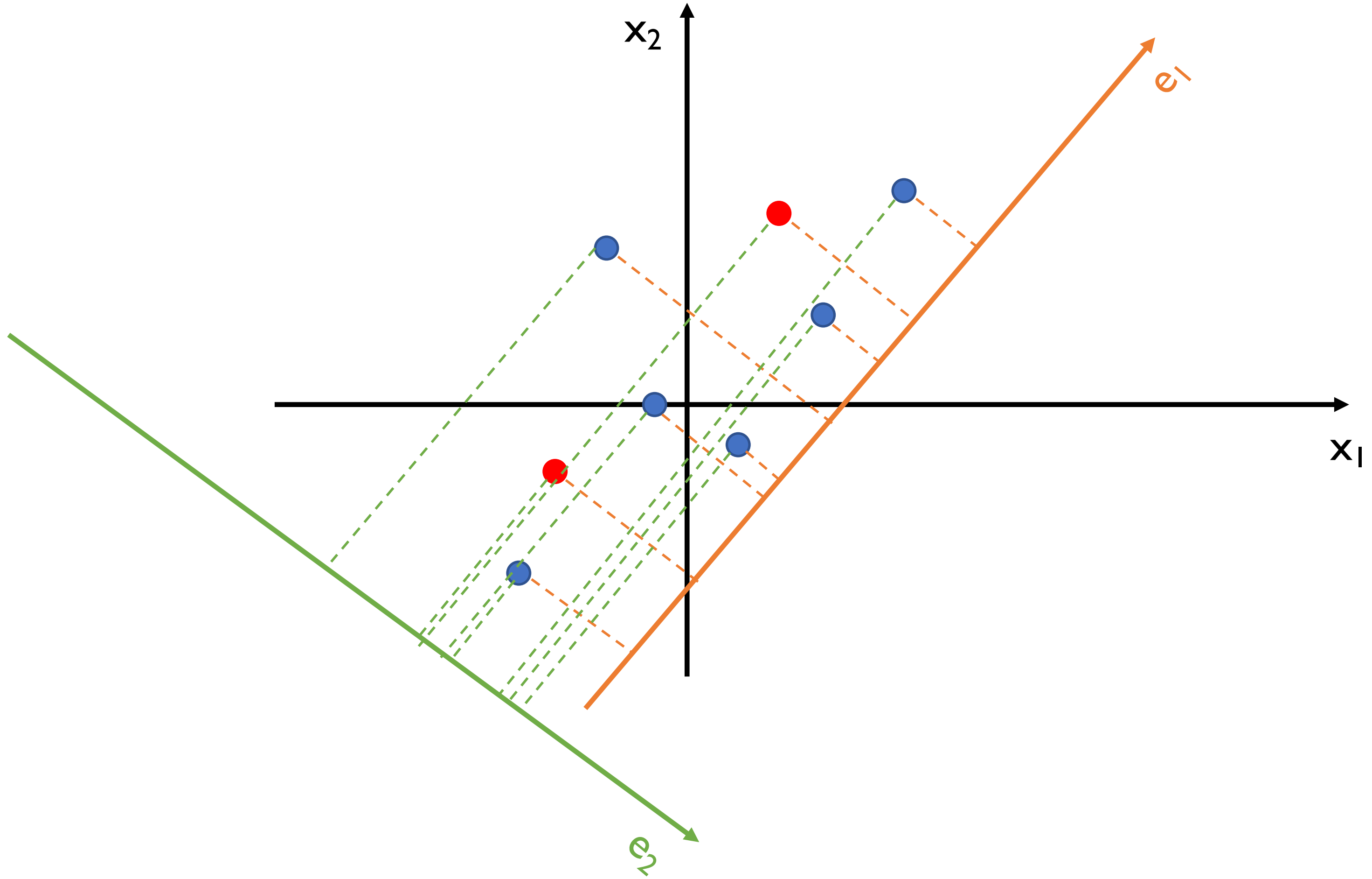
Why Do We Look for Greatest Variance?

Why is that good?



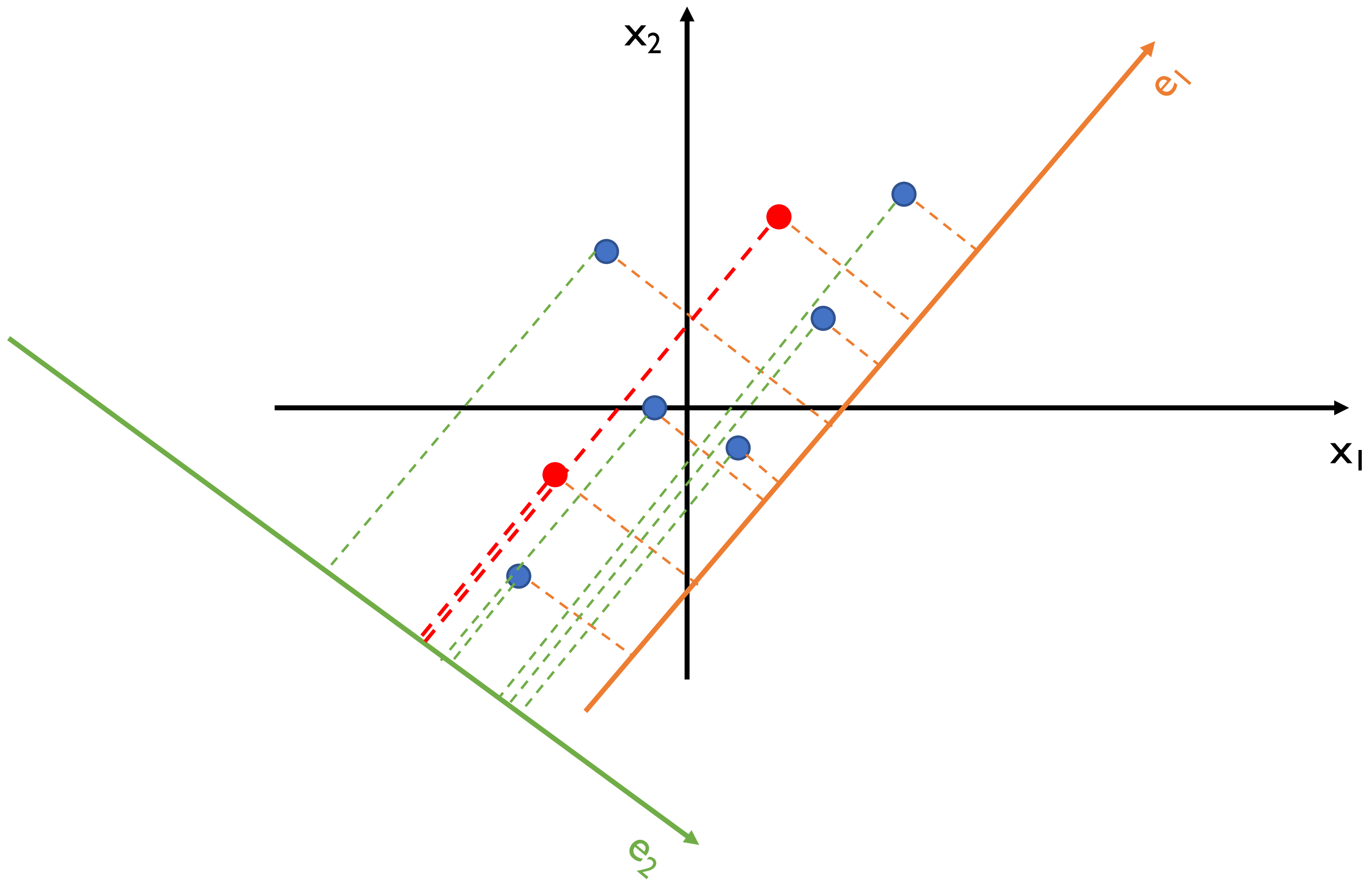
Why Do We Look for Greatest Variance?

Consider the 2 red points below



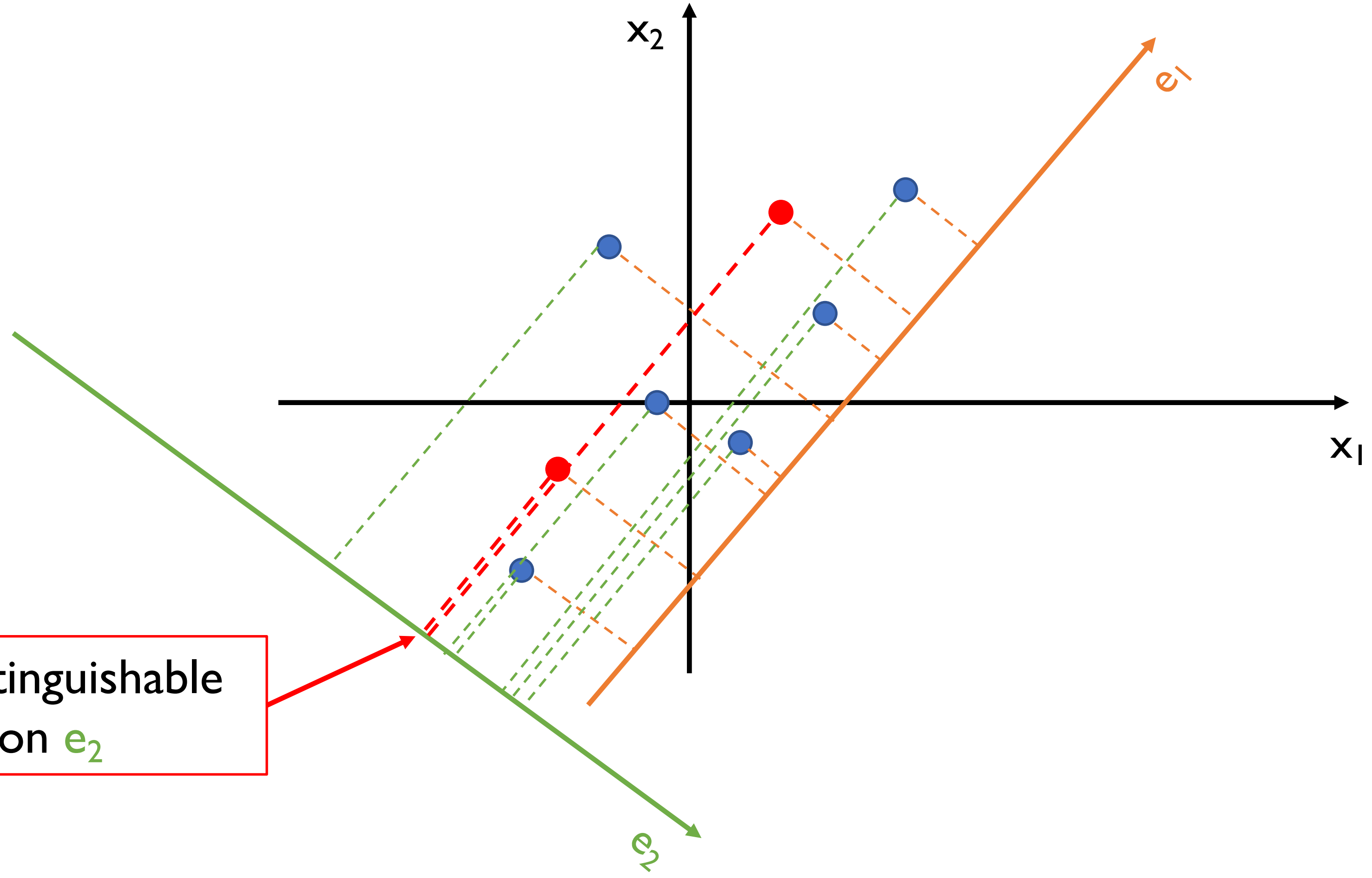
Why Do We Look for Greatest Variance?

On (x_1, x_2) far away from each other, end up close if projected onto e_2



Why Do We Look for Greatest Variance?

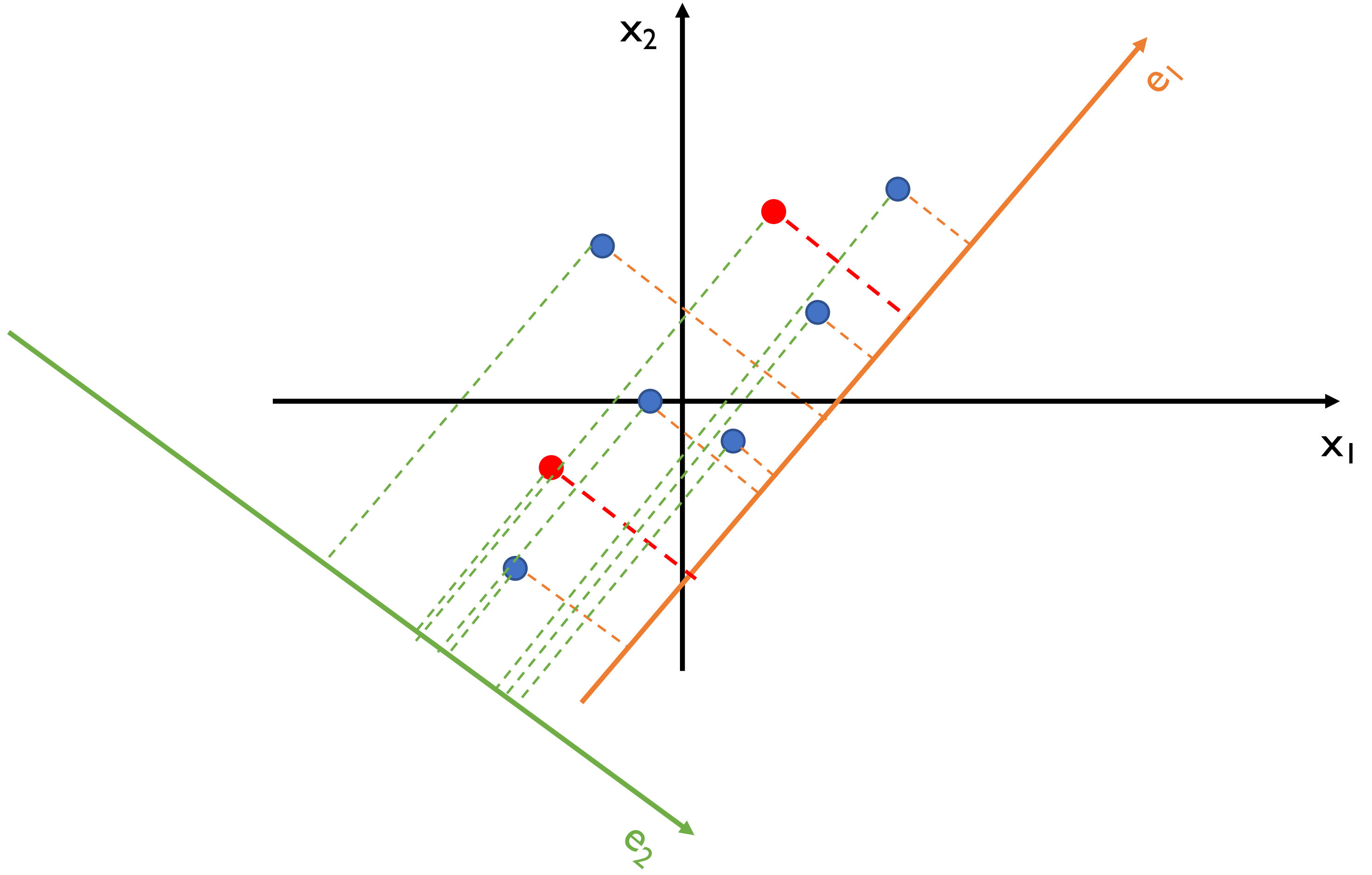
On (x_1, x_2) far away from each other, end up close if projected onto e_2



Almost indistinguishable points on e_2

Why Do We Look for Greatest Variance?

If projected onto e_1 they better preserve their distance



Why Do We Look for Greatest Variance?

- Intuitively, we want to minimize the chance that 2 points that are far in the original space end up close in the lower dimensional space

Why Do We Look for Greatest Variance?

- Intuitively, we want to minimize the chance that 2 points that are far in the original space end up close in the lower dimensional space
- Minimize distances between points as measured on (x_1, x_2) space and those measured on e

Why Do We Look for Greatest Variance?

- Intuitively, we want to minimize the chance that 2 points that are far in the original space end up close in the lower dimensional space
- Minimize distances between points as measured on (x_1, x_2) space and those measured on e

⇩

Solution

Pick e so as to **maximize variance** of projected data

Variance of a Random Variable

- The variance of a random variable X measures how far a set of (random) numbers are spread out from their mean value

Variance of a Random Variable

- The variance of a random variable X measures how far a set of (random) numbers are spread out from their mean value
- Formally, it is the expected value of the squared deviation from its mean

$$\mathrm{Var}(X) = \mathrm{E}[(X - \mu)^2]$$

$$\text{where } \mu = \mathrm{E}[X]$$

Covariance of Two Random Variables

- A measure of the joint variability of two random variables X and Y
  - Do X and Y increase/decrease together, or when one increases/decreases the other decreases/increases?

Covariance of Two Random Variables

- A measure of the joint variability of two random variables X and Y
  - Do X and Y increase/decrease together, or when one increases/decreases the other decreases/increases?
- Formally, it is the expected value of the product of their deviations from their individual means

$$\mathrm{Cov}(X, Y) = \mathrm{E}[(X - \mu_X)(Y - \mu_Y)]$$

$$\boxed{\mathrm{Cov}(X, X) = \mathrm{Var}(X)}$$

where $\mu_X = \mathrm{E}[X]$ and $\mu_Y = \mathrm{E}[Y]$

Covariance Matrix

- Given a random vector $\mathbf{X} = (X_1, \ldots, X_d)$ its covariance matrix K is a dxd square matrix with the covariance between each pair of elements

Covariance Matrix

- Given a random vector **X** = $(X_1, \ldots, X_d)$ its covariance matrix K is a dxd square matrix with the covariance between each pair of elements
- In the matrix diagonal there are variances, i.e., the covariance of each element with itself

$$\mathrm{K}[i, j] = \mathrm{Cov}(X_i, X_j)$$

Covariance Matrix of Original Dimensions

- The original set of dimensions is a random vector $\mathbf{X} = (X_1, \ldots, X_d)$

Covariance Matrix of Original Dimensions

- The original set of dimensions is a random vector $\mathbf{X} = (X_1, \ldots, X_d)$
- In our example, $d = 2$ and $\mathbf{X} = (X_1, X_2)$

Covariance Matrix of Original Dimensions

- The original set of dimensions is a random vector $\mathbf{X} = (X_1, \ldots, X_d)$
- In our example, d = 2 and $\mathbf{X} = (X_1, X_2)$
- The covariance matrix K is a 2-by-2 matrix

Covariance Matrix of Original Dimensions

- The original set of dimensions is a random vector $\mathbf{X} = (X_1, \ldots, X_d)$
- In our example, $d = 2$ and $\mathbf{X} = (X_1, X_2)$
- The covariance matrix K is a 2-by-2 matrix
- To ease the covariance computation, we center each data point at zero
  - Subtracting the mean of each attribute/dimension
  - The mean of each dimension becomes then 0

Covariance Matrix of Original Dimensions

Let n be the total number of data points: $\mathbf{x}_1, \ldots, \mathbf{x}_n$

Each data point is represented by a (x_1, x_2) pair

$\mathbf{x}_i = (x_{i,1}, x_{i,2})$

We associate 2 random variables X_1, X_2 to each dimension, and we compute:

$$\mu_1 = \mathrm{E}[X_1] = \frac{1}{n}\sum_{i=1}^{n} x_{i,1}$$

$$\mu_2 = \mathrm{E}[X_2] = \frac{1}{n}\sum_{i=1}^{n} x_{i,2}$$

$$\mathbf{x}_i = (x_{i,1} - \mu_1, x_{i,2} - \mu_2)$$

Covariance Matrix of Original Dimensions

Let us rewrite each data point $\mathbf{x}_i$ as follows:

$\mathbf{x}_i = (x'_{i,1}, x'_{i,2})$ where:

$x'_{i,1} = x_{i,1} - \mu_1; x'_{i,2} = x_{i,2} - \mu_2$

$$\mu_1^{\text{new}} = \mathrm{E}[X_1] = \frac{1}{n}\sum_{i=1}^{n} x'_{i,1} = \frac{1}{n}\sum_{i=1}^{n}(x_{i,1} - \mu_1)$$

$$\mu_2^{\text{new}} = \mathrm{E}[X_2] = \frac{1}{n}\sum_{i=1}^{n} x'_{i,2} = \frac{1}{n}\sum_{i=1}^{n}(x_{i,2} - \mu_2)$$

Covariance Matrix of Original Dimensions

$$\mu_1^{\text{new}} = \frac{1}{n}\sum_{i=1}^{n}(x_{i,1} - \mu_1) = \frac{1}{n}\Bigg(\underbrace{\sum_{i=1}^{n} x_{i,1}}_{n\mu_1} - \underbrace{\sum_{i=1}^{n} \mu_1}_{n\mu_1}\Bigg) = 0$$

$$\mu_2^{\text{new}} = \frac{1}{n}\sum_{i=1}^{n}(x_{i,2} - \mu_2) = \frac{1}{n}\Bigg(\underbrace{\sum_{i=1}^{n} x_{i,2}}_{n\mu_2} - \underbrace{\sum_{i=1}^{n} \mu_2}_{n\mu_2}\Bigg) = 0$$

0-mean

Covariance Matrix of Original Dimensions

Scaling data so as to have 0-mean on all dimensions
allow computing covariance much easily

$$\mathrm{Cov}(X_1, X_2) = \mathrm{E}[(X_1 - \underbrace{\mu_1^{\mathrm{new}}}_{=0})(X_2 - \underbrace{\mu_2^{\mathrm{new}}}_{=0})] = \mathrm{E}[X_1 X_2]$$

As a consequence, the covariance matrix is also easier to compute!

Covariance Matrix of Original Dimensions

Let's assume the following is our 2-by-2 covariance matrix

$$\begin{array}{cc} & \begin{array}{cc} x_1 & x_2 \end{array} \\ \begin{array}{c} x_1 \\ x_2 \end{array} & \begin{bmatrix} 2 & 4/5 \\ 4/5 & 3/5 \end{bmatrix} \end{array}$$

Covariance Matrix of Original Dimensions

Let's assume the following is our 2-by-2 covariance matrix

$$\begin{array}{cc} & \begin{array}{cc} x_1 & x_2 \end{array} \\ \begin{array}{c} x_1 \\ x_2 \end{array} & \begin{bmatrix} 2 & 4/5 \\ 4/5 & 3/5 \end{bmatrix} \end{array}$$

$$\boxed{\mathrm{Cov}(X_1, X_2) = \frac{1}{n} \sum_{i=1}^{n} x'_{i,1} * x'_{i,2}}$$

Covariance Matrix of Original Dimensions

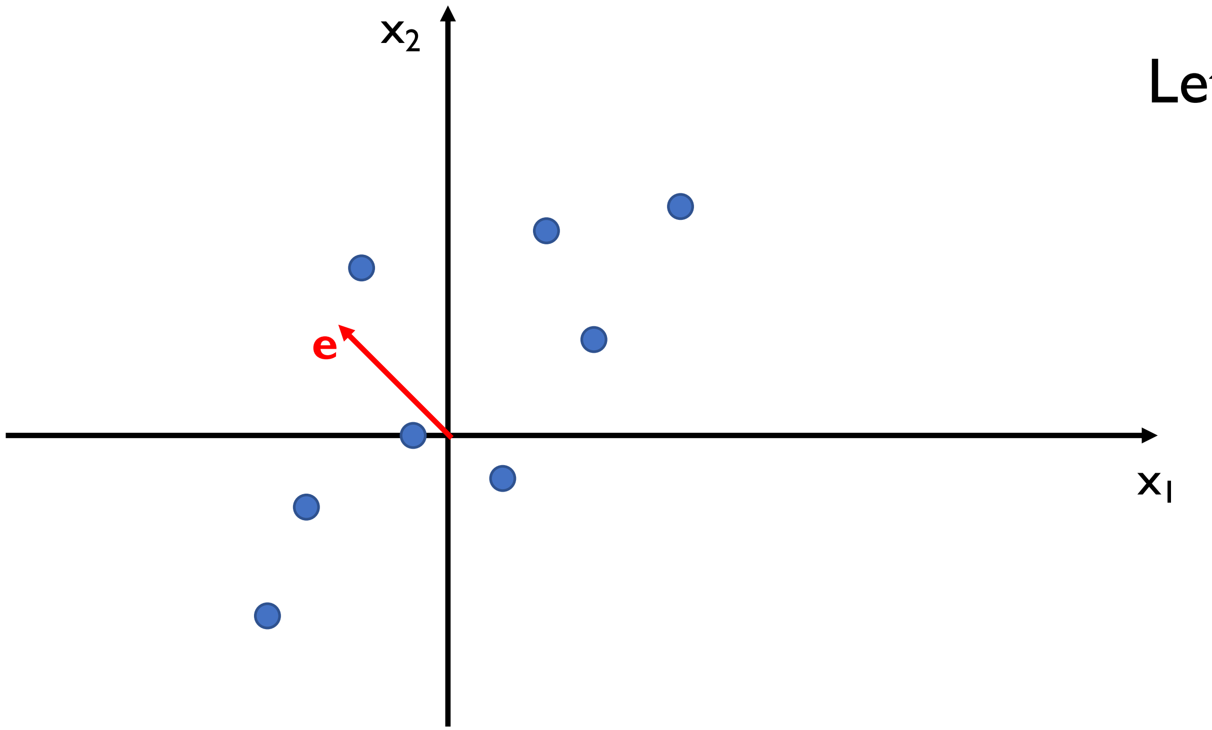
Let's assume the following is our 2-by-2 covariance matrix

$$\begin{array}{cc} & \begin{array}{cc} x_1 & x_2 \end{array} \\ \begin{array}{c} x_1 \\ x_2 \end{array} & \begin{bmatrix} 2 & 4/5 \\ 4/5 & 3/5 \end{bmatrix} \end{array}$$

$$\boxed{\mathrm{Cov}(X_1, X_2) = \frac{1}{n}\sum_{i=1}^{n} x'_{i,1} * x'_{i,2}}$$

$$\boxed{\mathrm{Cov}(X_2, X_2) = \mathrm{Var}(X_2) = \frac{1}{n}\sum_{i=1}^{n} (x'_{i,2})^2}$$

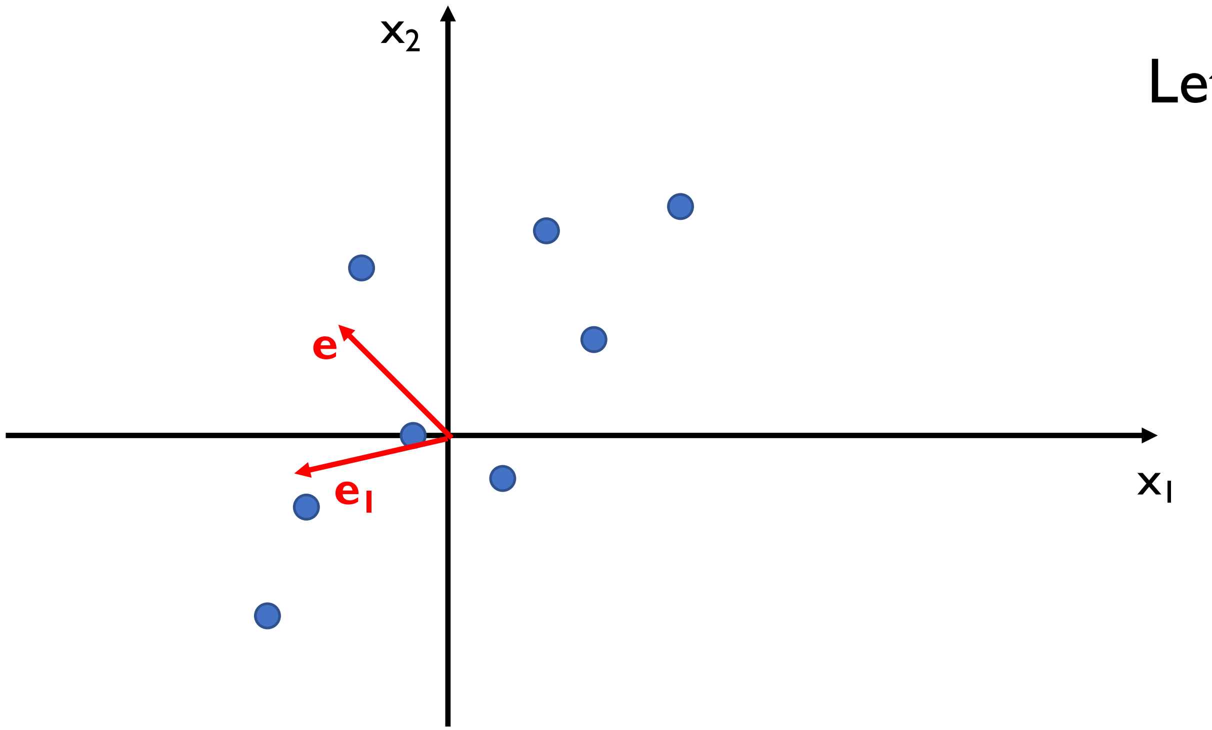
Covariance Matrix of Original Dimensions



Let's multiply our 2-by-2 covariance matrix K by a random vector **e** = (-1,1)

$$\underbrace{\begin{bmatrix} 2 & 4/5 \\ 4/5 & 3/5 \end{bmatrix}}_{K} \underbrace{\begin{bmatrix} -1 \\ 1 \end{bmatrix}}_{\mathbf{e}} =$$

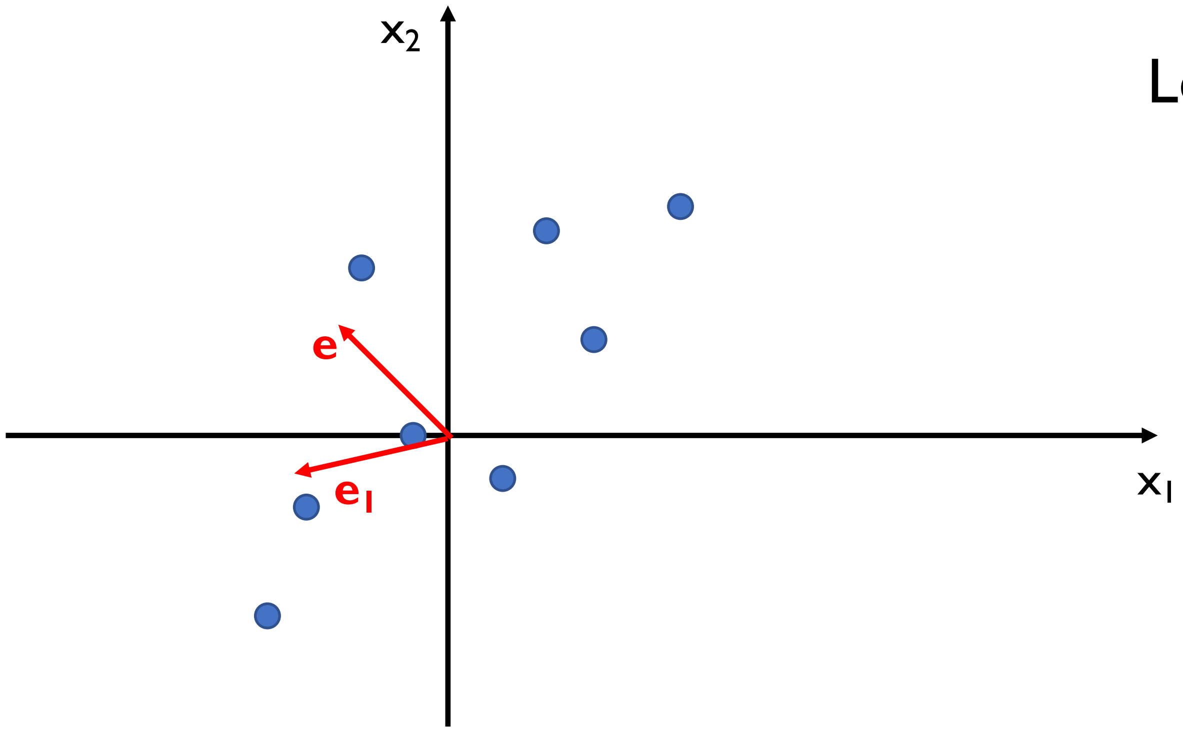
Covariance Matrix of Original Dimensions



Let's multiply our 2-by-2 covariance matrix K by a random vector **e** = (-1,1)

$$\underbrace{\begin{bmatrix} 2 & 4/5 \\ 4/5 & 3/5 \end{bmatrix}}_{K} \underbrace{\begin{bmatrix} -1 \\ 1 \end{bmatrix}}_{\mathbf{e}} = \underbrace{\begin{bmatrix} -6/5 \\ -1/5 \end{bmatrix}}_{\mathbf{e_1}}$$

Covariance Matrix of Original Dimensions

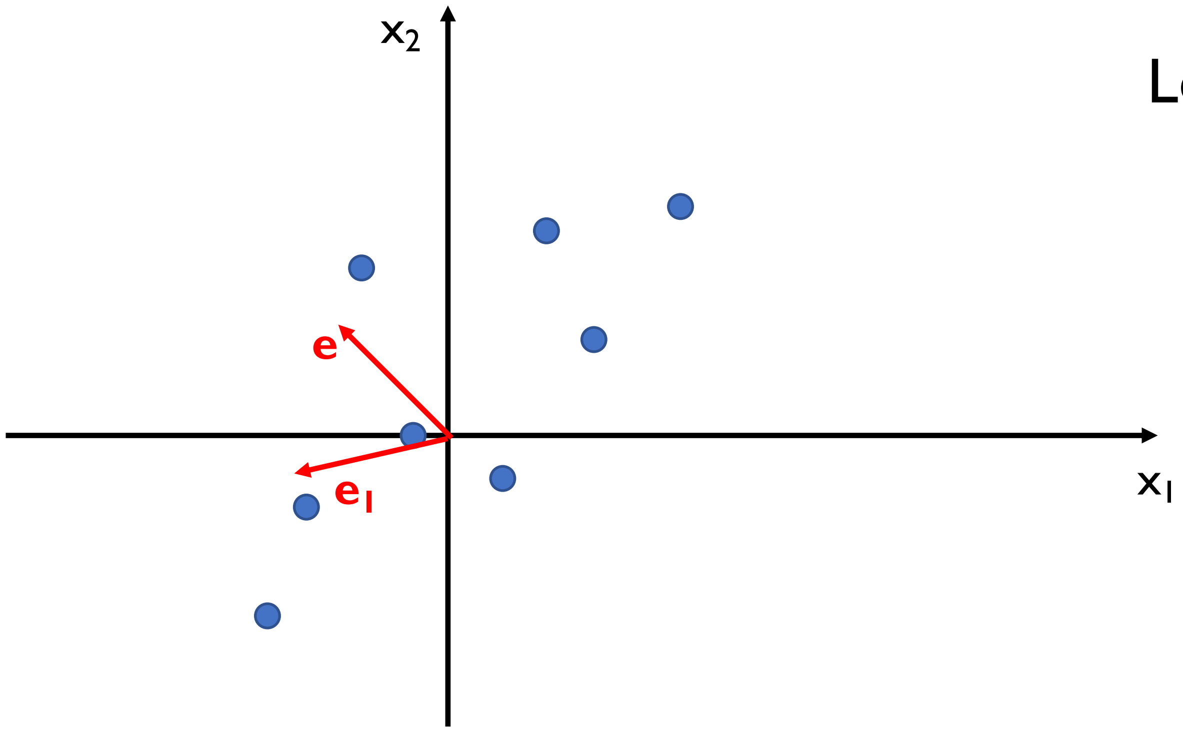


Let's multiply our 2-by-2 covariance matrix K by a random vector **e = (-1,1)**

$$\underbrace{\begin{bmatrix} 2 & 4/5 \\ 4/5 & 3/5 \end{bmatrix}}_{K} \underbrace{\begin{bmatrix} -1 \\ 1 \end{bmatrix}}_{\mathbf{e}} = \underbrace{\begin{bmatrix} -6/5 \\ -1/5 \end{bmatrix}}_{\mathbf{e_1}}$$

slope = 1/-1 = -1

Covariance Matrix of Original Dimensions



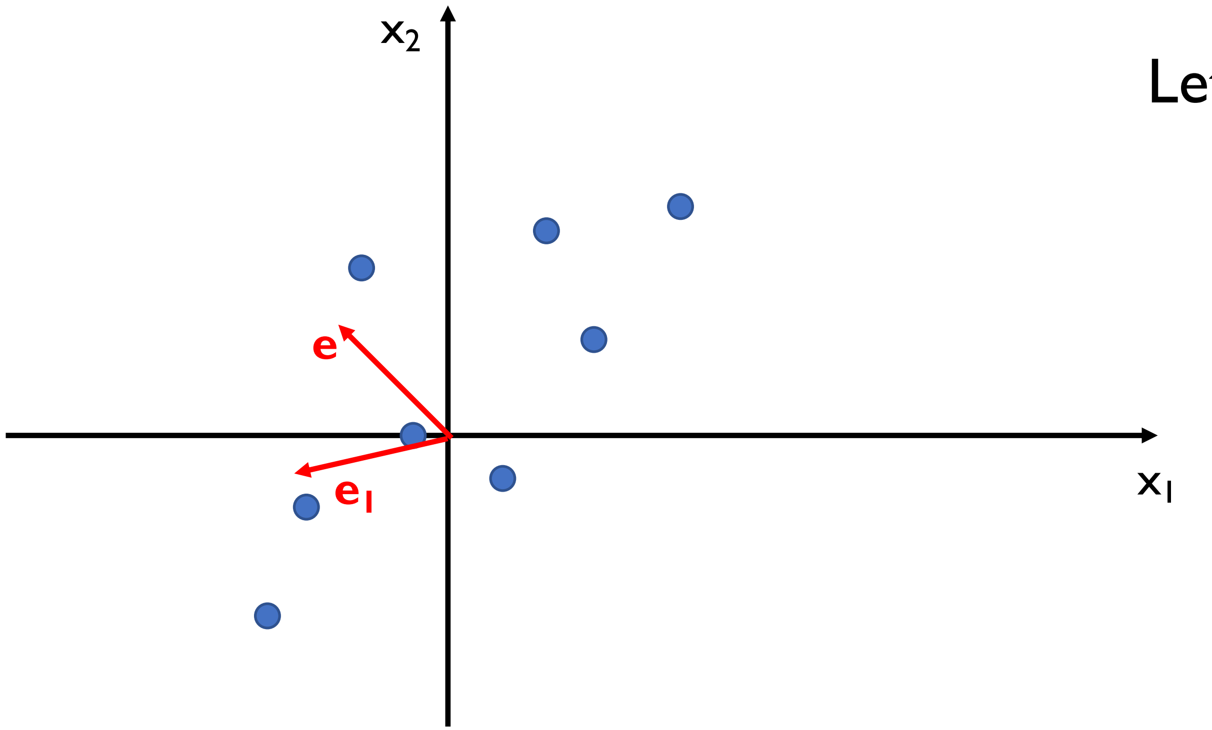
Let's multiply our 2-by-2 covariance matrix K by a random vector **e = (-1,1)**

$$\underbrace{\begin{bmatrix} 2 & 4/5 \\ 4/5 & 3/5 \end{bmatrix}}_{K} \underbrace{\begin{bmatrix} -1 \\ 1 \end{bmatrix}}_{\mathbf{e}} = \underbrace{\begin{bmatrix} -6/5 \\ -1/5 \end{bmatrix}}_{\mathbf{e_1}}$$

slope = 1/-1 = -1

new slope = -(1/5)/-(6/5) = 1/6

Covariance Matrix of Original Dimensions



Let's multiply our 2-by-2 covariance matrix K by a random vector **e = (-1,1)**

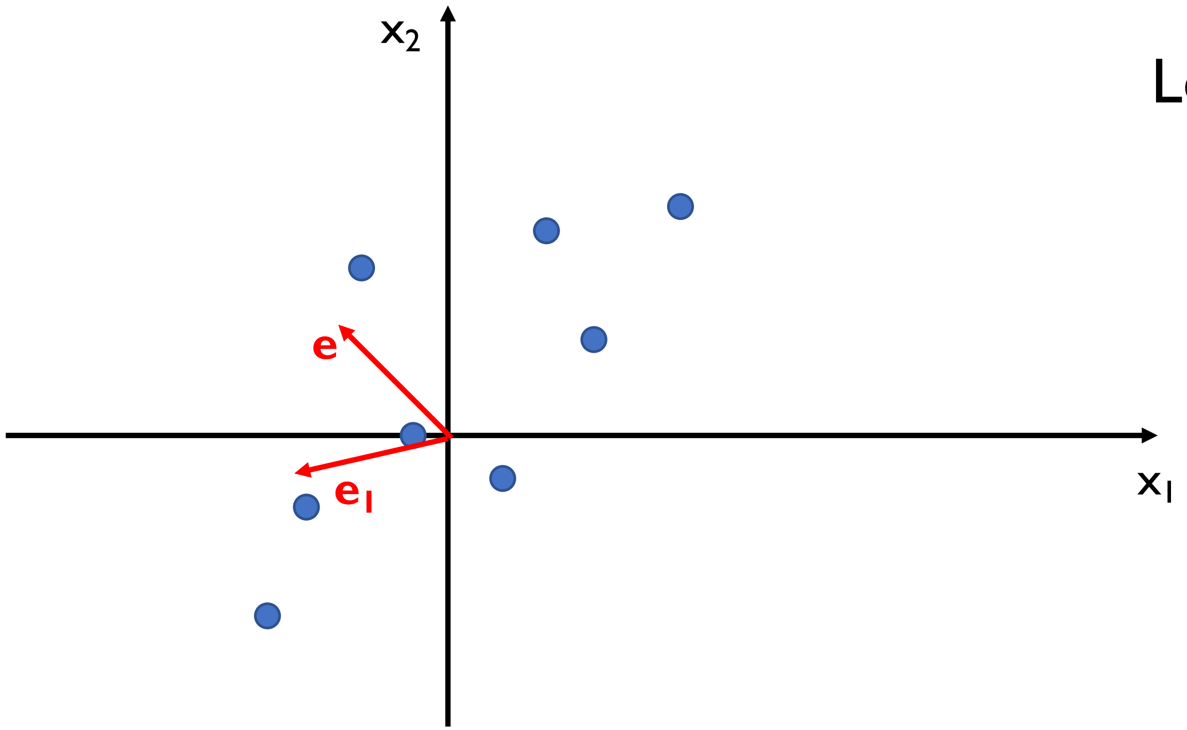
$$\underbrace{\begin{bmatrix} 2 & 4/5 \\ 4/5 & 3/5 \end{bmatrix}}_{K} \underbrace{\begin{bmatrix} -1 \\ 1 \end{bmatrix}}_{\mathbf{e}} = \underbrace{\begin{bmatrix} -6/5 \\ -1/5 \end{bmatrix}}_{\mathbf{e_1}}$$

slope = 1/-1 = -1

Turns towards the direction of the greatest variance

new slope = -(1/5)/-(6/5) = 1/6

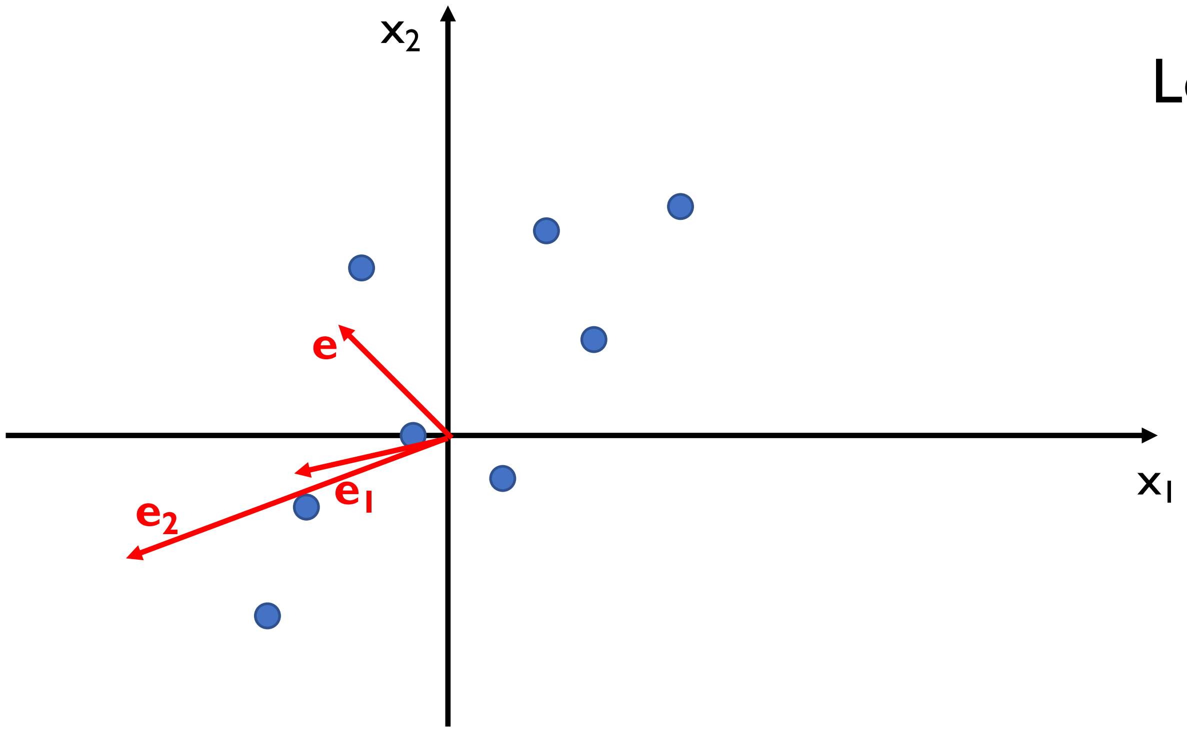
Covariance Matrix of Original Dimensions



Let's repeat the previous step multiplying the covariance matrix K by **e_1** = (-6/5,-1/5)

$$\underbrace{\begin{bmatrix} 2 & 4/5 \\ 4/5 & 3/5 \end{bmatrix}}_{K} \underbrace{\begin{bmatrix} -6/5 \\ -1/5 \end{bmatrix}}_{\mathbf{e_1}} =$$

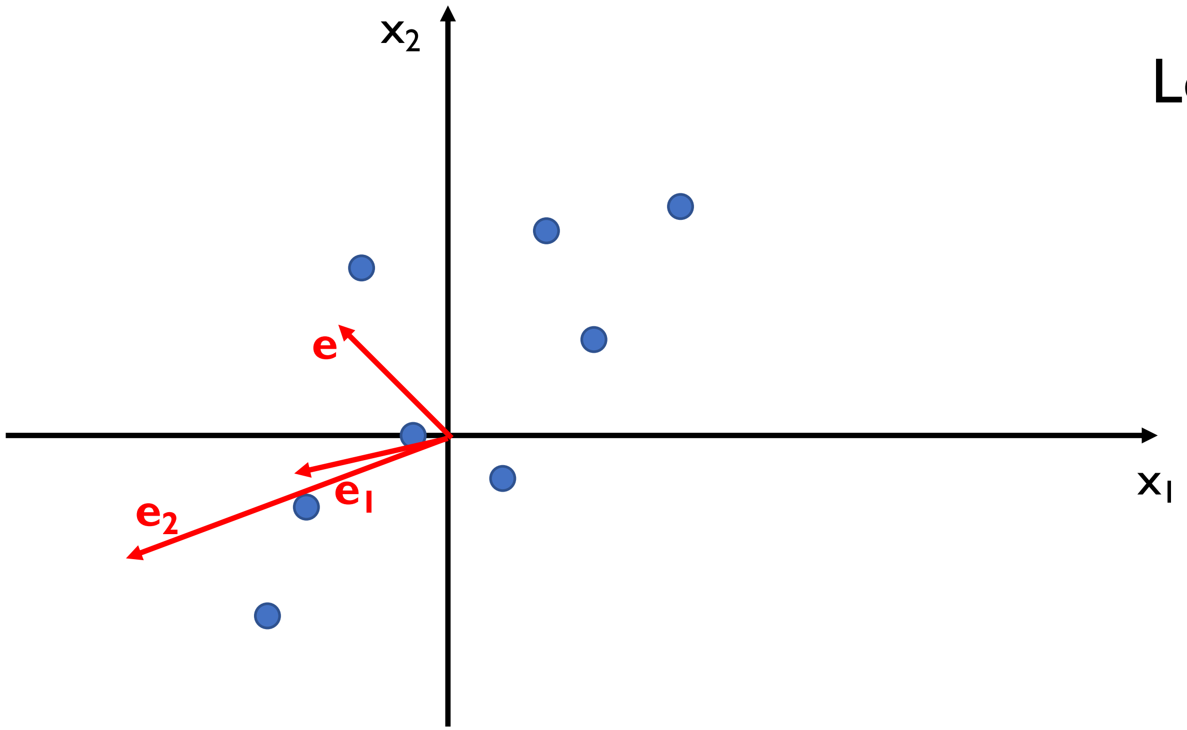
Covariance Matrix of Original Dimensions



Let's repeat the previous step multiplying the covariance matrix K by $\mathbf{e}_1$ = (-6/5,-1/5)

$$\underbrace{\begin{bmatrix} 2 & 4/5 \\ 4/5 & 3/5 \end{bmatrix}}_{K} \underbrace{\begin{bmatrix} -6/5 \\ -1/5 \end{bmatrix}}_{\mathbf{e_1}} = \underbrace{\begin{bmatrix} -64/25 \\ -27/25 \end{bmatrix}}_{\mathbf{e_2}}$$

Covariance Matrix of Original Dimensions



Let's repeat the previous step multiplying the covariance matrix K by **$\mathbf{e}_1$ = (-6/5,-1/5)**

$$\underbrace{\begin{bmatrix} 2 & 4/5 \\ 4/5 & 3/5 \end{bmatrix}}_{K} \underbrace{\begin{bmatrix} -6/5 \\ -1/5 \end{bmatrix}}_{\mathbf{e}_1} = \underbrace{\begin{bmatrix} -64/25 \\ -27/25 \end{bmatrix}}_{\mathbf{e}_2}$$

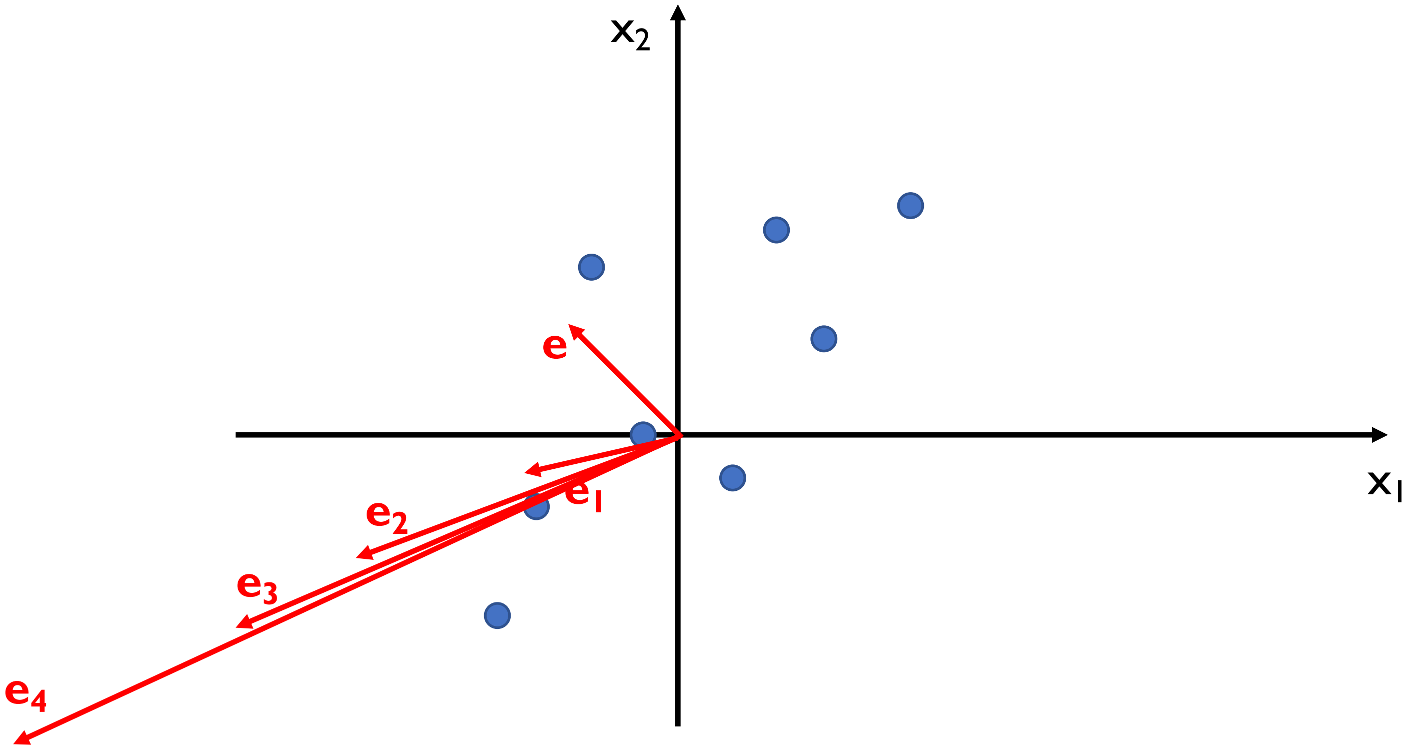
slope = 1/6

Turns towards the direction of the greatest variance

new slope = 27/64

Covariance Matrix of Original Dimensions

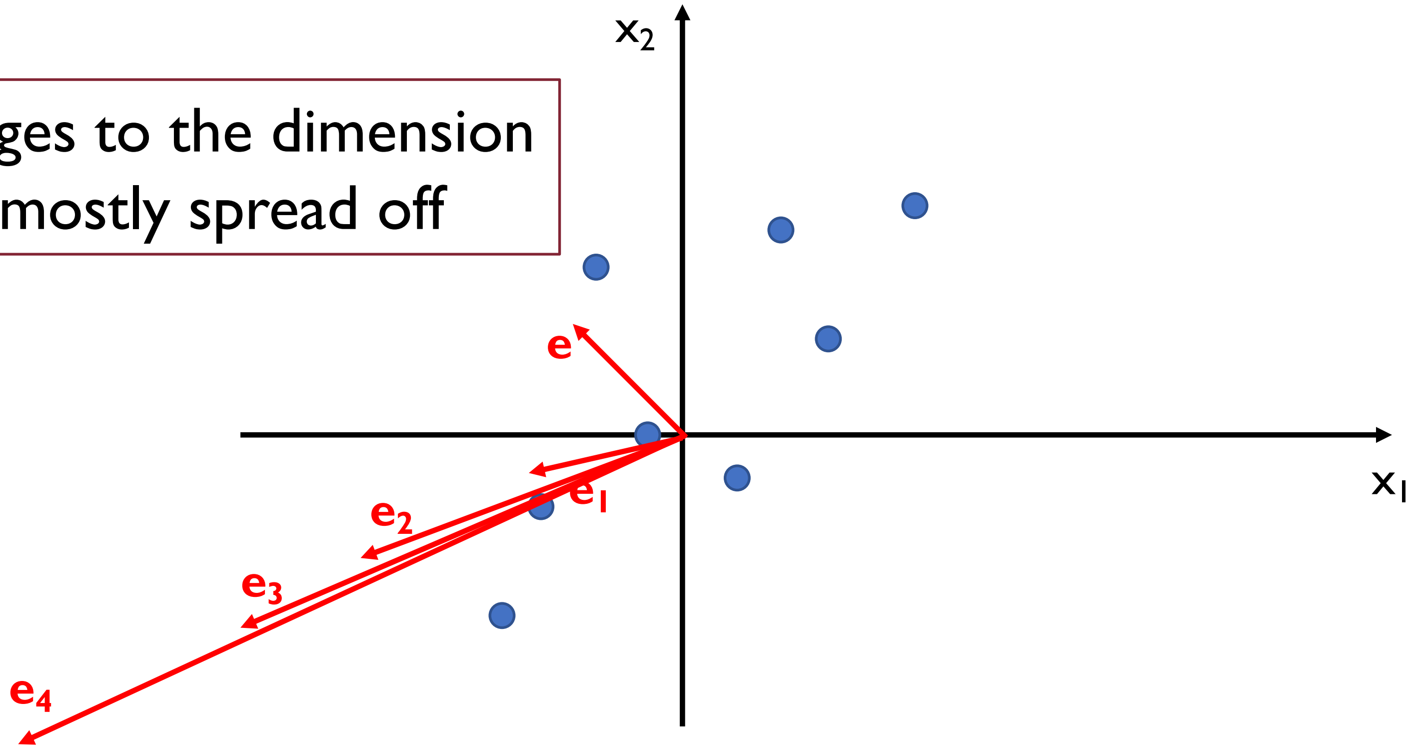
If we keep doing this the resulting vector is getting **longer** and **turns** towards the direction of the **largest variance**



Covariance Matrix of Original Dimensions

If we keep doing this the resulting vector is getting **longer** and **turns** towards the direction of the **largest variance**

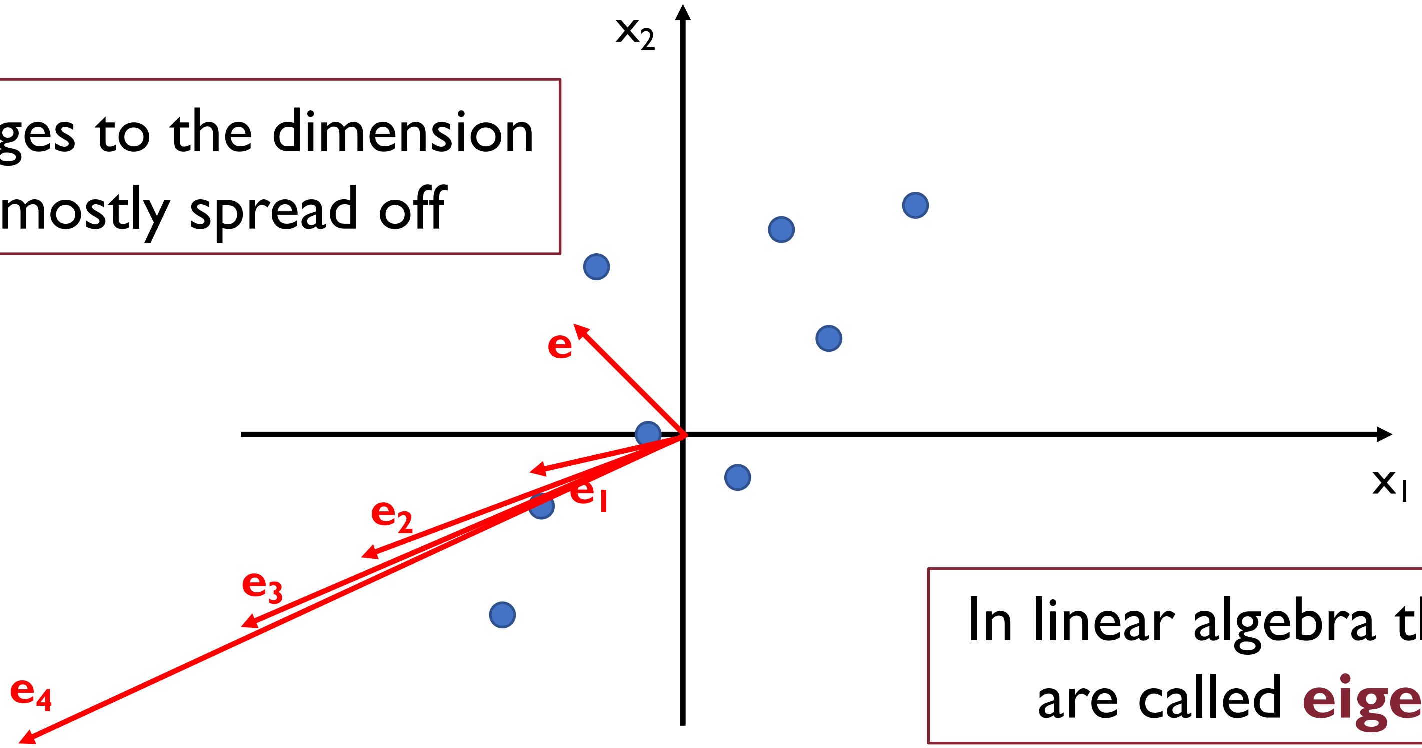
The slope converges to the dimension where data is mostly spread off



Covariance Matrix of Original Dimensions

If we keep doing this the resulting vector is getting **longer** and **turns** towards the direction of the **largest variance**

The slope converges to the dimension where data is mostly spread off



In linear algebra those vectors are called **eigenvectors**

Eigenvectors of the Covariance Matrix

$$\boxed{K\mathbf{e} = \lambda\mathbf{e}}$$

eigenvector

eigenvalue

When you multiply a matrix by an **eigenvector** $\mathbf{e}$ the resulting vector does not change its direction, but it is only scaled by a factor λ (**eigenvalue**)

Eigenvectors of the Covariance Matrix

$$\boxed{K\mathbf{e} = \lambda\mathbf{e}}$$

eigenvector

eigenvalue

When you multiply a matrix by an **eigenvector** **e** the resulting vector does not change its direction, but it is only scaled by a factor λ (**eigenvalue**)

Principal Components

eigenvectors of the covariance matrix with the **largest** eigenvalues

How Do We Compute Eigenvectors?

Remember that we want to solve for **e** the following:

$$K\mathbf{e} = \lambda\mathbf{e}$$

We can rewrite the system of equations above as:

$$K\mathbf{e} - \lambda\mathbf{e} = 0 \Rightarrow (K - \lambda I)\mathbf{e} = 0$$

I is the identity matrix

How Do We Compute Eigenvectors?

We therefore resort to solve the following **homogeneous system**:

$$(K - \lambda I)\mathbf{e} = 0$$

How Do We Compute Eigenvectors?

We therefore resort to solve the following **homogeneous system**:

$$(K - \lambda I)\mathbf{e} = 0$$

Any homogeneous system always has a **trivial solution**, i.e., the zero vector **e = 0**

How Do We Compute Eigenvectors?

We therefore resort to solve the following **homogeneous system**:

$$(K - \lambda I)\mathbf{e} = 0$$

Any homogeneous system always has a **trivial solution**, i.e., the zero vector **e = 0**

The only way for the homogeneous system above to have a **non-trivial** solution is for its matrix (K – λI) to be **non-invertible**, otherwise:

$$\cancel{(K - \lambda I)}\cancel{(K - \lambda I)^{-1}}\boxed{\mathbf{e} = 0}\cancel{(K - \lambda I)^{-1}}$$

How Do We Compute Eigenvectors?

A square matrix is **invertible** iff its determinant is **not** equal to 0

How Do We Compute Eigenvectors?

A square matrix is **invertible** iff its determinant is **not** equal to 0

If the determinant of the matrix $(K - \lambda I)$ is equal to 0, it is **non-invertible**

How Do We Compute Eigenvectors?

A square matrix is **invertible** iff its determinant is **not** equal to 0

⇓

If the determinant of the matrix $(K - \lambda I)$ is equal to 0, it is **non-invertible**

⇓

The corresponding homogeneous system will have a **non-trivial** solution

How Do We Compute Eigenvectors?

1. Find the eigenvalues by solving for λ: det(K – λI) = 0

$$\det\left(\underbrace{\begin{bmatrix} 2-\lambda & 4/5 \\ 4/5 & 3/5-\lambda \end{bmatrix}}_{K-\lambda I}\right) = 0$$

How Do We Compute Eigenvectors?

1. Find the eigenvalues by solving for λ: det(K – λI) = 0

$$\det\left(\underbrace{\begin{bmatrix} 2-\lambda & 4/5 \\ 4/5 & 3/5-\lambda \end{bmatrix}}_{K-\lambda I}\right) = 0$$

$$(2-\lambda)(3/5-\lambda)-(4/5)(4/5) = \lambda^2-13/5\lambda+14/25$$

$$\boxed{\lambda^2 - 13/5\lambda + 14/25 = 0}$$ **characteristic equation** of K

How Do We Compute Eigenvectors?

1. Find the eigenvalues by solving for λ: det(K – λI) = 0

$$\det\left(\underbrace{\begin{bmatrix} 2-\lambda & 4/5 \\ 4/5 & 3/5-\lambda \end{bmatrix}}_{K-\lambda I}\right) = 0$$

$$(2-\lambda)(3/5-\lambda)-(4/5)(4/5) = \lambda^2-13/5\lambda+14/25$$

$$\boxed{\lambda^2 - 13/5\lambda + 14/25 = 0}$$ **characteristic equation** of K

$$\boxed{\lambda_1 = \frac{13+\sqrt{113}}{10} \approx 2.36;} \quad \boxed{\lambda_2 = \frac{13-\sqrt{113}}{10} \approx 0.24}$$

How Do We Compute Eigenvectors?

2. Plug each eigenvalue in to find the corresponding eigenvector

$$\underbrace{\begin{bmatrix} 2 & 4/5 \\ 4/5 & 3/5 \end{bmatrix}}_{K} \underbrace{\begin{bmatrix} e_{1,1} \\ e_{1,2} \end{bmatrix}}_{\mathbf{e_1}} = \lambda_1 \underbrace{\begin{bmatrix} e_{1,1} \\ e_{1,2} \end{bmatrix}}_{\mathbf{e_1}}$$

$$\underbrace{\begin{bmatrix} 2 & 4/5 \\ 4/5 & 3/5 \end{bmatrix}}_{K} \underbrace{\begin{bmatrix} e_{2,1} \\ e_{2,2} \end{bmatrix}}_{\mathbf{e_2}} = \lambda_2 \underbrace{\begin{bmatrix} e_{2,1} \\ e_{2,2} \end{bmatrix}}_{\mathbf{e_2}}$$

How Do We Compute Eigenvectors?

Let's see what happens for λ_1

$$\begin{cases} 2e_{1,1} + 4/5e_{1,2} = \frac{13+\sqrt{113}}{10}e_{1,1} \\ 4/5e_{1,1} + 3/5e_{1,2} = \frac{13+\sqrt{113}}{10}e_{1,2} \end{cases}$$

How Do We Compute Eigenvectors?

Let's see what happens for λ_1

$$\begin{cases} 2e_{1,1} + 4/5e_{1,2} = \frac{13+\sqrt{113}}{10}e_{1,1} \\ 4/5e_{1,1} + 3/5e_{1,2} = \frac{13+\sqrt{113}}{10}e_{1,2} \end{cases}$$

Just replacing $\lambda_1 \sim 2.36$

$$\begin{cases} 2e_{1,1} + 0.8e_{1,2} = 2.36e_{1,1} \\ 0.8e_{1,1} + 0.6e_{1,2} = 2.36e_{1,2} \end{cases}$$

How Do We Compute Eigenvectors?

Let's see what happens for λ_1

$$\begin{cases} 2e_{1,1} + 4/5e_{1,2} = \frac{13+\sqrt{113}}{10}e_{1,1} \\ 4/5e_{1,1} + 3/5e_{1,2} = \frac{13+\sqrt{113}}{10}e_{1,2} \end{cases}$$

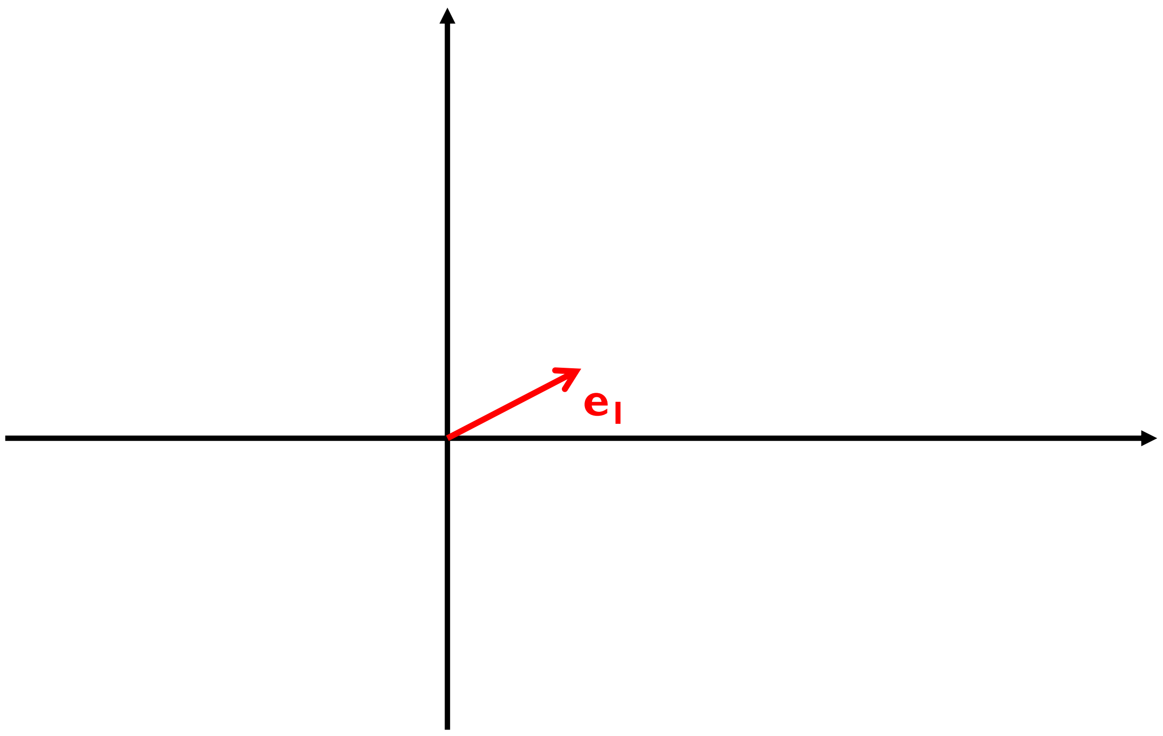
Just replacing $\lambda_1 \sim 2.36$

$$\begin{cases} 2e_{1,1} + 0.8e_{1,2} = 2.36e_{1,1} \\ 0.8e_{1,1} + 0.6e_{1,2} = 2.36e_{1,2} \end{cases} \Rightarrow \boxed{e_{1,1} \approx 2.2e_{1,2}}$$

The system has infinte many solutions

How Do We Compute Eigenvectors?

Any vector which satisfies the relationship above works!

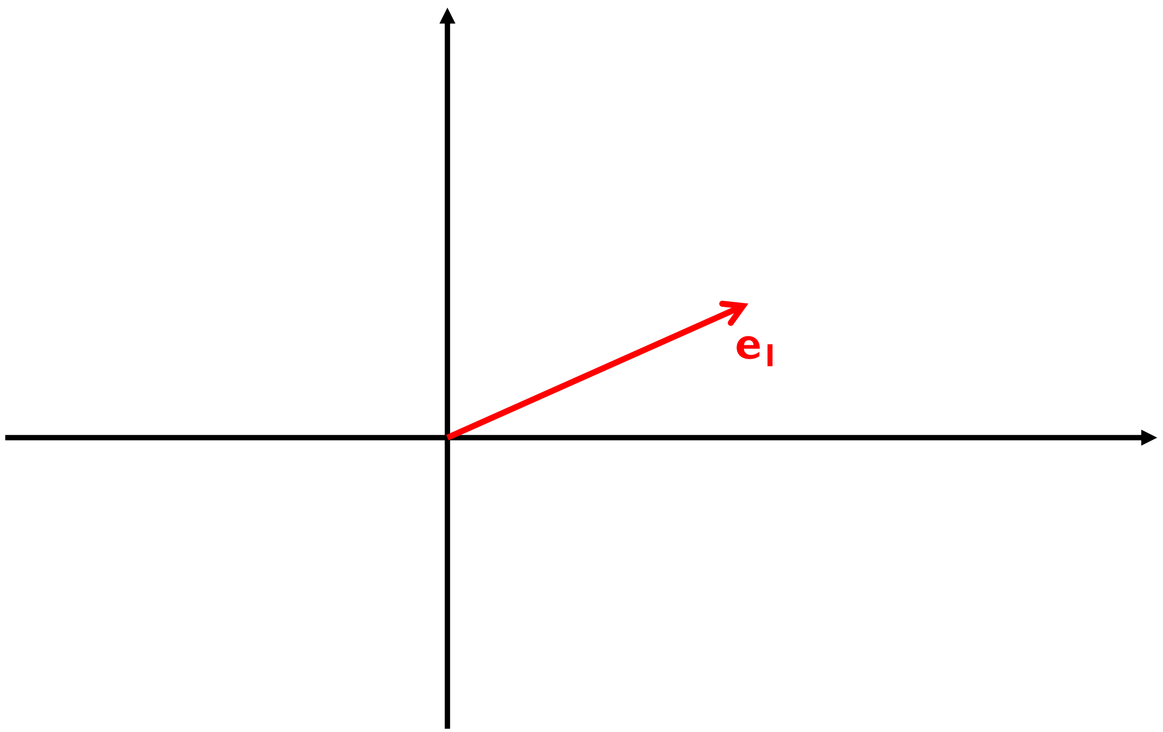


$$\begin{bmatrix} 1.1 \\ 0.5 \end{bmatrix}$$

$$e_{1,1} \approx 2.2 e_{1,2}$$

How Do We Compute Eigenvectors?

Any vector which satisfies the relationship above works!

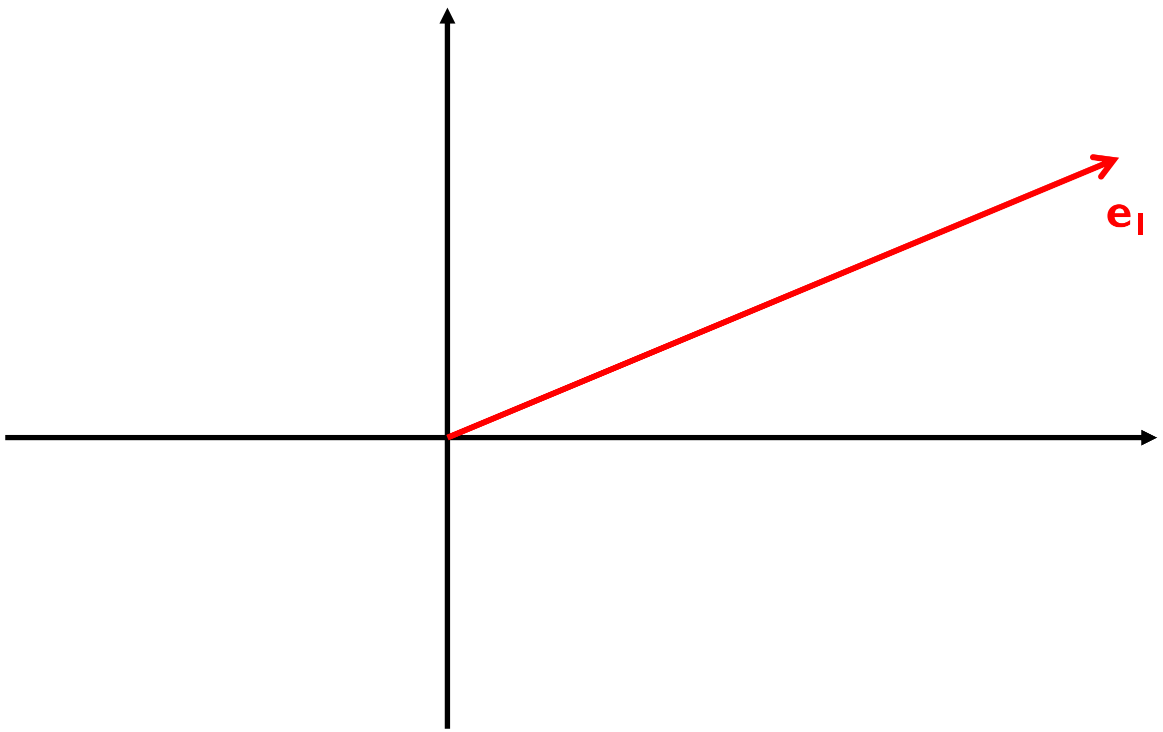


$$\begin{bmatrix} 2.2 \\ 1 \end{bmatrix}$$

$$e_{1,1} \approx 2.2 e_{1,2}$$

How Do We Compute Eigenvectors?

Any vector which satisfies the relationship above works!

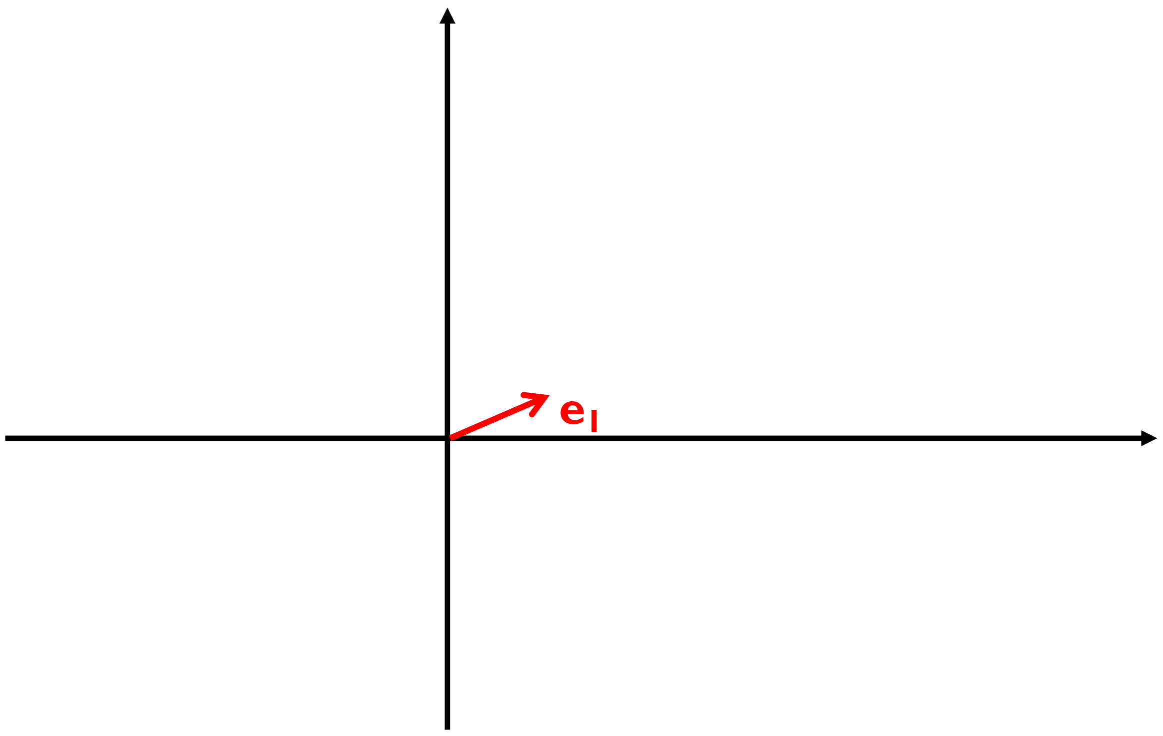


$$\begin{bmatrix} 4.4 \\ 2 \end{bmatrix}$$

$$e_{1,1} \approx 2.2 e_{1,2}$$

How Do We Compute Eigenvectors?

By convention, we restrict to $||\mathbf{e_1}|| = 1$

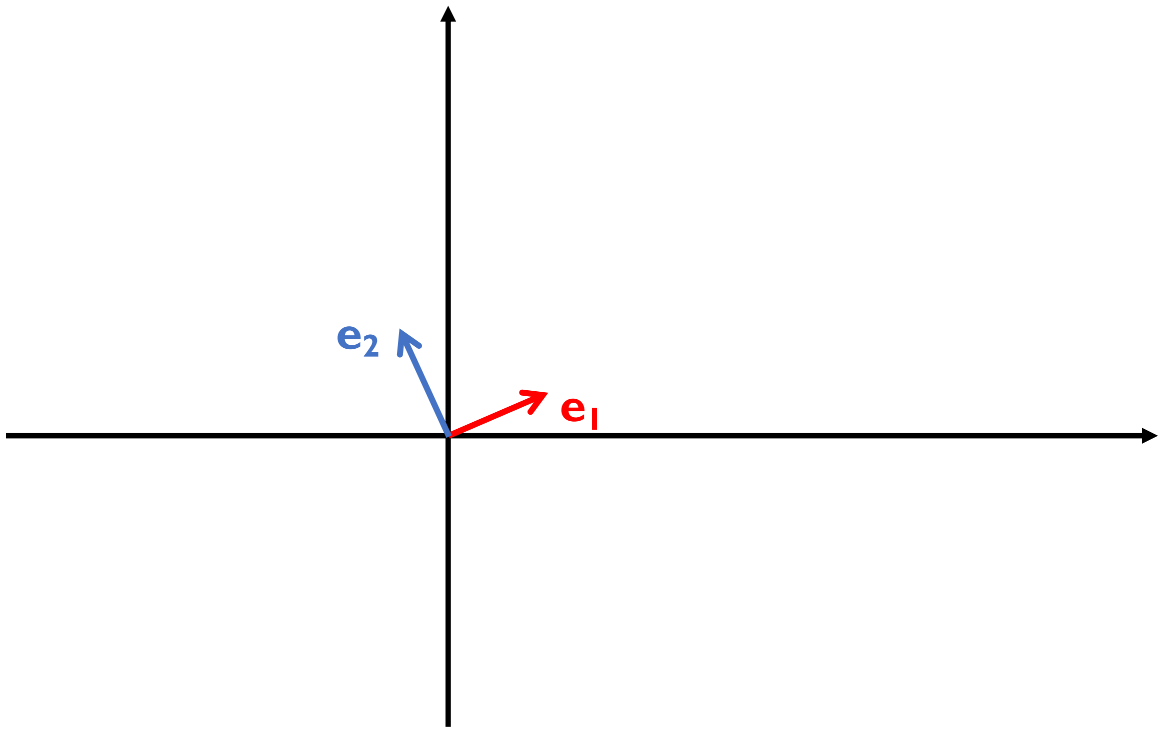


$$\begin{bmatrix} 0.91 \\ 0.41 \end{bmatrix}$$

$$e_{1,1} \approx 2.2 e_{1,2}$$

How Do We Compute Eigenvectors?

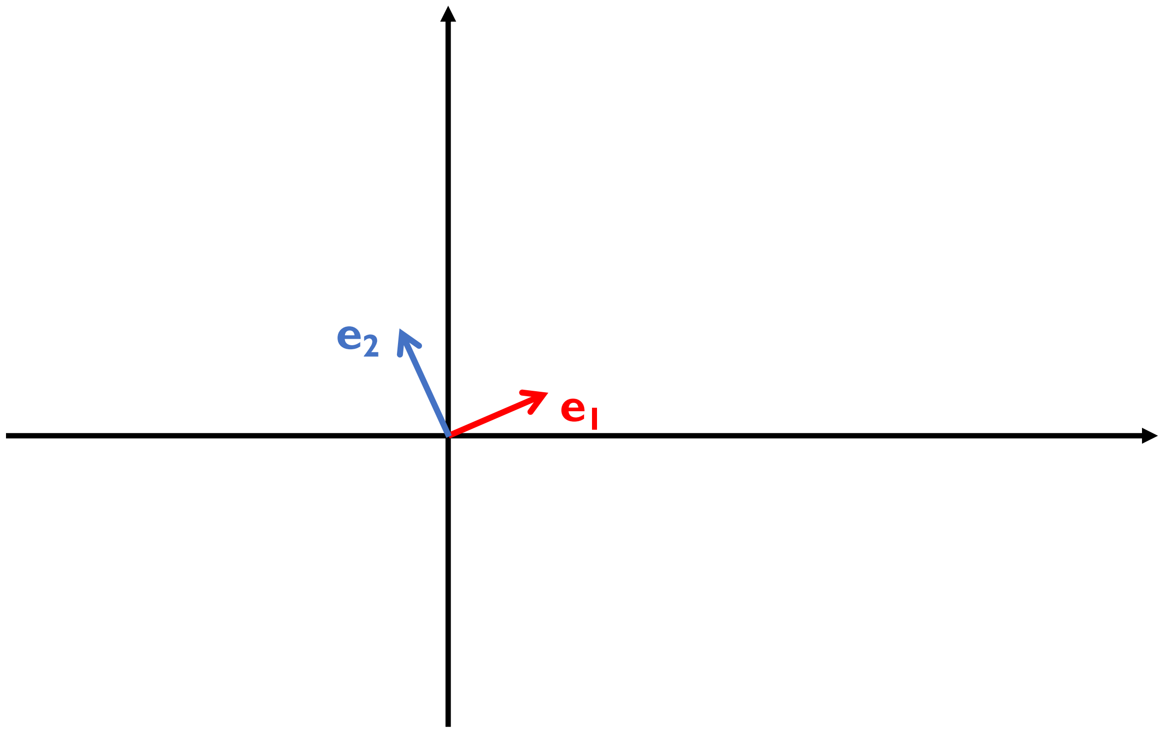
The second eigenvector **e_2** can be found by plugging in the smaller eigenvalue λ_2



How Do We Compute Eigenvectors?

The second eigenvector $\mathbf{e}_2$ can be found by plugging in the smaller eigenvalue λ_2

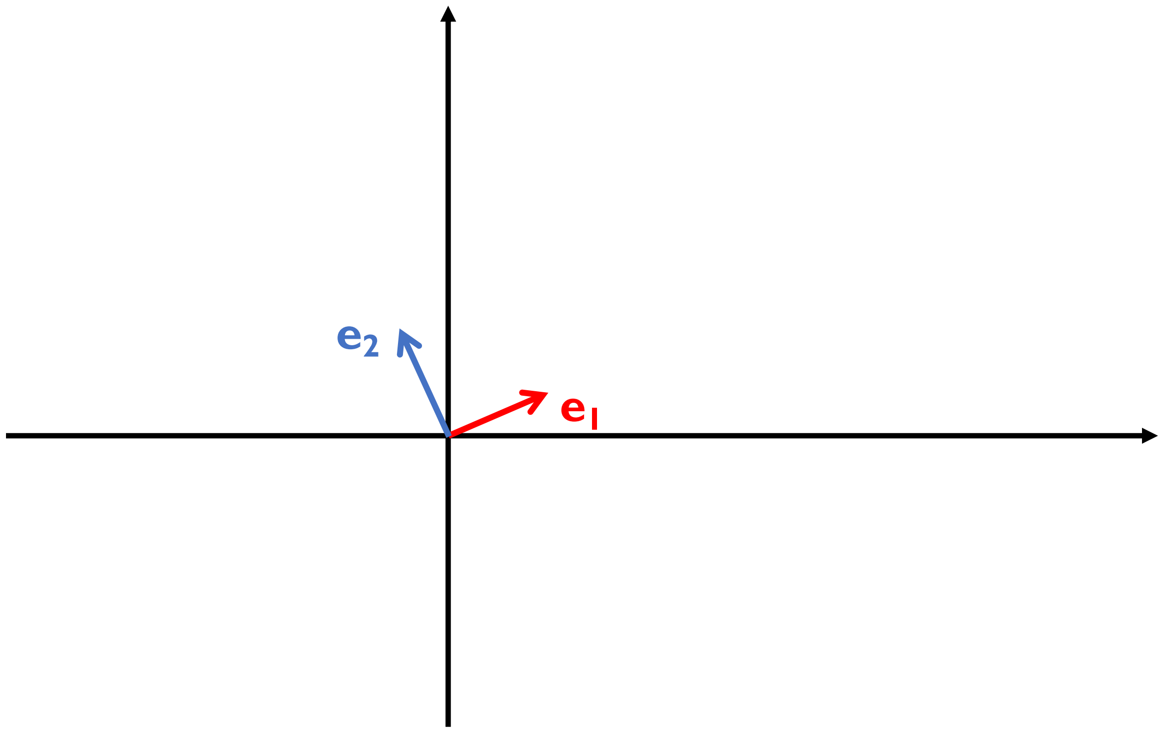
This is just orthogonal to the previously found $\mathbf{e}_1$



How Do We Compute Eigenvectors?

The second eigenvector $\mathbf{e}_2$ can be found by plugging in the smaller eigenvalue λ_2

This is just orthogonal to the previously found $\mathbf{e}_1$



$\mathbf{e}_1$ and $\mathbf{e}_2$ are the new coordinate system replacing the original $\mathbf{x}_1$ and $\mathbf{x}_2$

$$\mathbf{e}_1 = \begin{bmatrix} 0.91 \\ 0.41 \end{bmatrix} \mathbf{e}_2 = \begin{bmatrix} -0.41 \\ 0.91 \end{bmatrix}$$

Principal Components

$$\mathbf{e}_1 = \begin{bmatrix} 0.91 \\ 0.41 \end{bmatrix} \quad \mathbf{e}_2 = \begin{bmatrix} -0.41 \\ 0.91 \end{bmatrix}$$

$\mathbf{e}_1$ is the 1st principal component as it is the eigenvector corresponding to the largest eigenvalue

$\mathbf{e}_2$ is the 2nd principal component as it is the eigenvector corresponding to the smallest eigenvalue

Projecting to New Dimensions: 2-d Case

- $\mathbf{e}_1$ and $\mathbf{e}_2$ identify our new coordinate system (principal components)

Projecting to New Dimensions: 2-d Case

- $\mathbf{e}_1$ and $\mathbf{e}_2$ identify our new coordinate system (principal components)
- Both of them are 2-dimensional vectors

Projecting to New Dimensions: 2-d Case

- $\mathbf{e_1}$ and $\mathbf{e_2}$ identify our new coordinate system (principal components)
- Both of them are 2-dimensional vectors
- Let $\mathbf{x} = (x_1, x_2)$ be a point (i.e., a vector) in the original $(\mathbf{x_1}, \mathbf{x_2})$-space

Projecting to New Dimensions: 2-d Case

- $\mathbf{e}_1$ and $\mathbf{e}_2$ identify our new coordinate system (principal components)
- Both of them are 2-dimensional vectors
- Let $\mathbf{x} = (x_1, x_2)$ be a point (i.e., a vector) in the original $(\mathbf{x}_1, \mathbf{x}_2)$-space

⇩

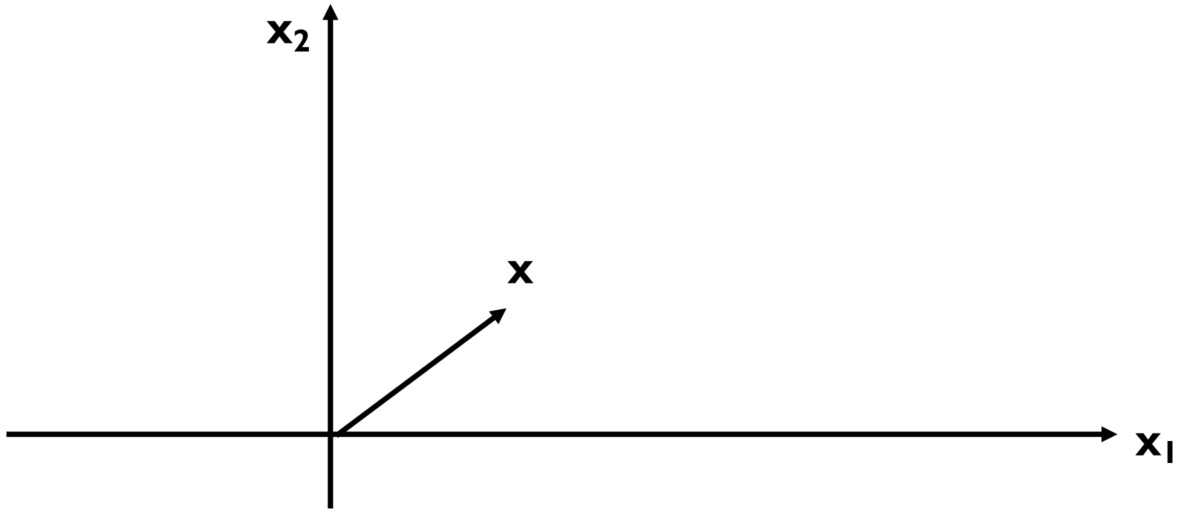
Goal

We want to represent $\mathbf{x}$ in the new $(\mathbf{e}_1, \mathbf{e}_2)$-coordinate system

Projecting to New Dimensions: 2-d Case

1. Center **x** around the mean of each dimension

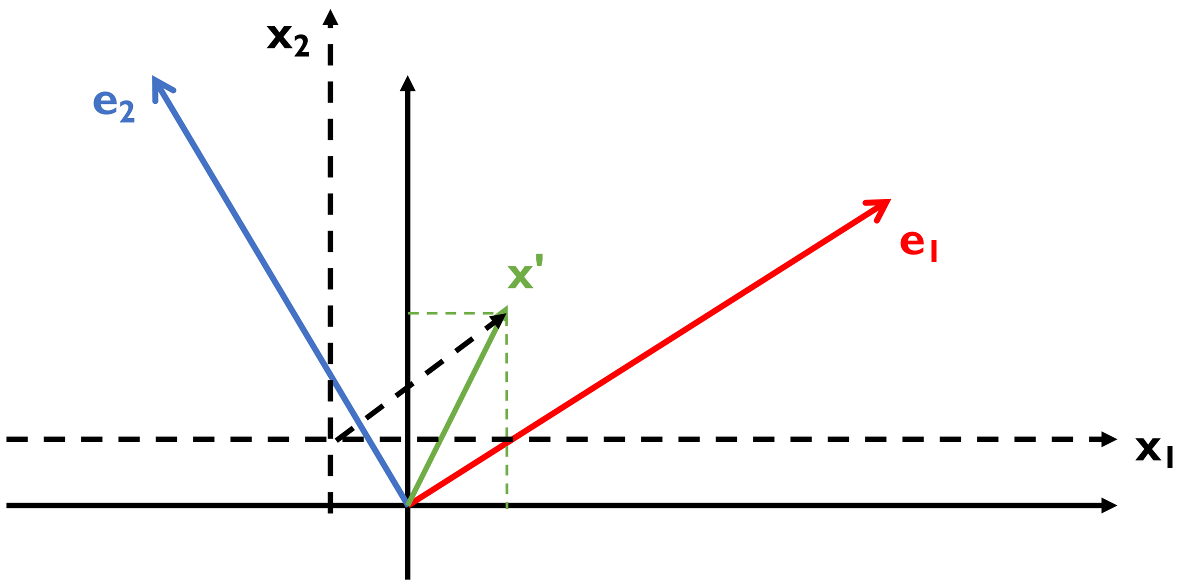
$$\mathbf{x}' = \mathbf{x} - \boldsymbol{\mu} = (x_1 - \mu_1, x_2 - \mu_2)$$



Projecting to New Dimensions: 2-d Case

1. Center **x** around the mean of each dimension

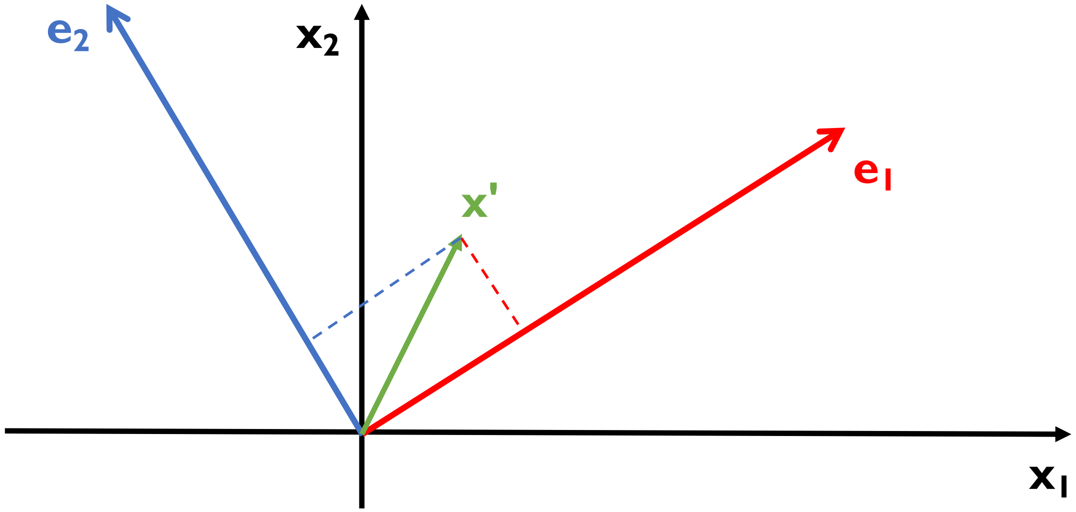
$$\mathbf{x}' = \mathbf{x} - \boldsymbol{\mu} = (x_1 - \mu_1, x_2 - \mu_2)$$



Projecting to New Dimensions: 2-d Case

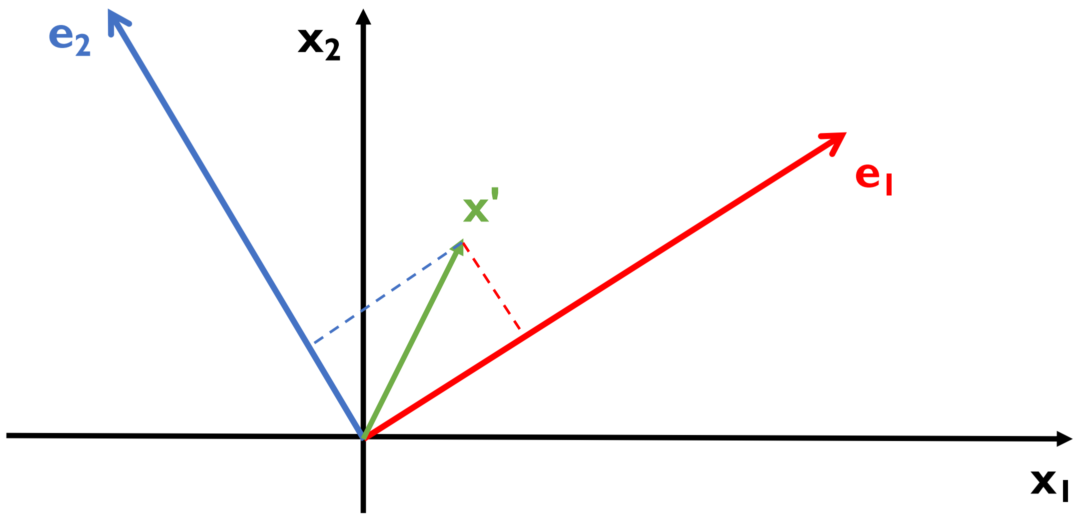
2. Project **x'** on each dimension **e_1** and **e_2**

$$\mathbf{x}' = \underbrace{(x'_1, x'_2)}_{\text{coordinates of } \mathbf{x}' \text{ in the } (\mathbf{e}_1, \mathbf{e}_2)\text{-space}} = (\mathbf{x}'^T\mathbf{e}_1, \mathbf{x}'^T\mathbf{e}_2)$$



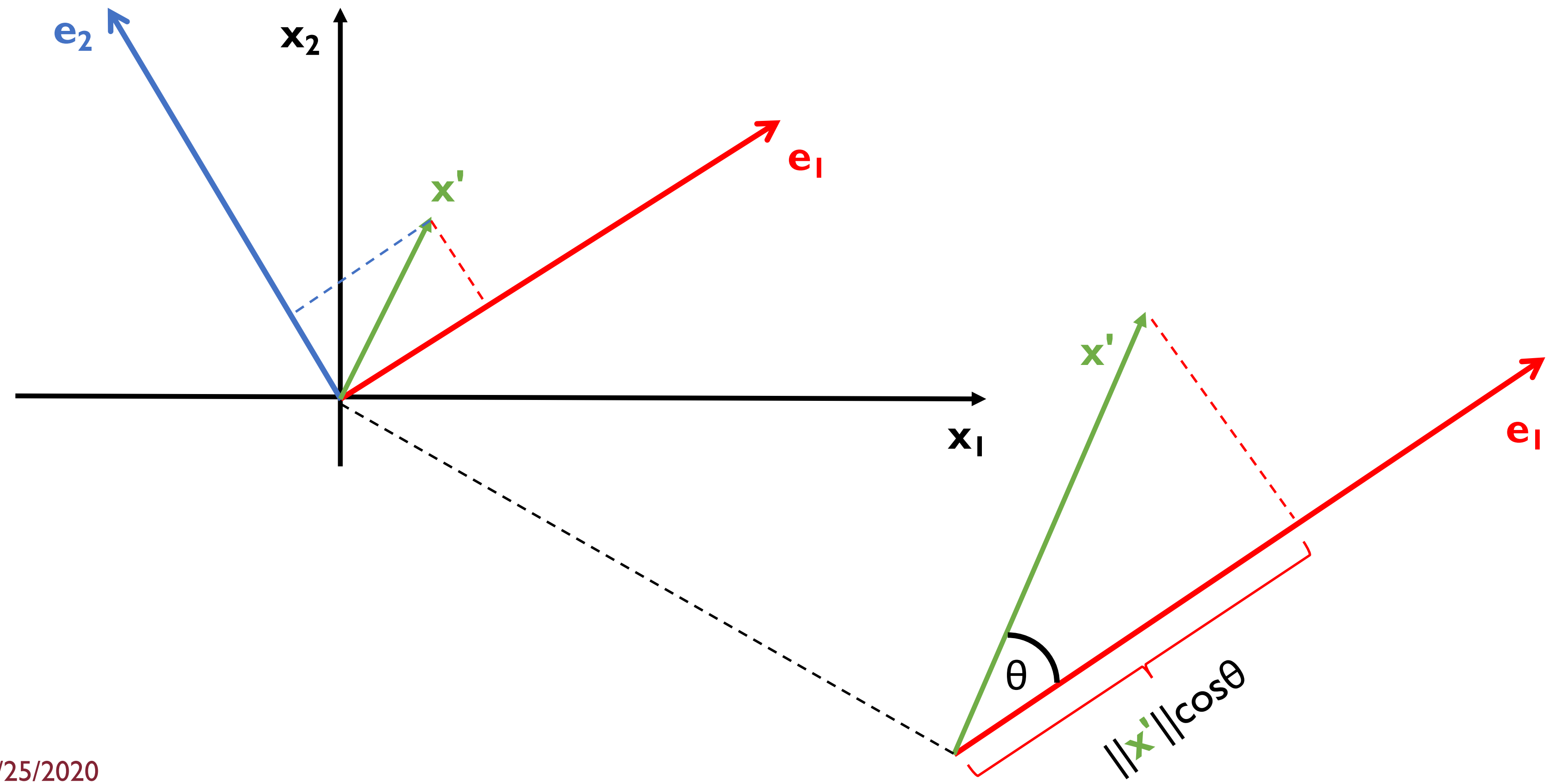
Projecting to New Dimensions: 2-d Case

Why the dot product?



Projecting to New Dimensions: 2-d Case

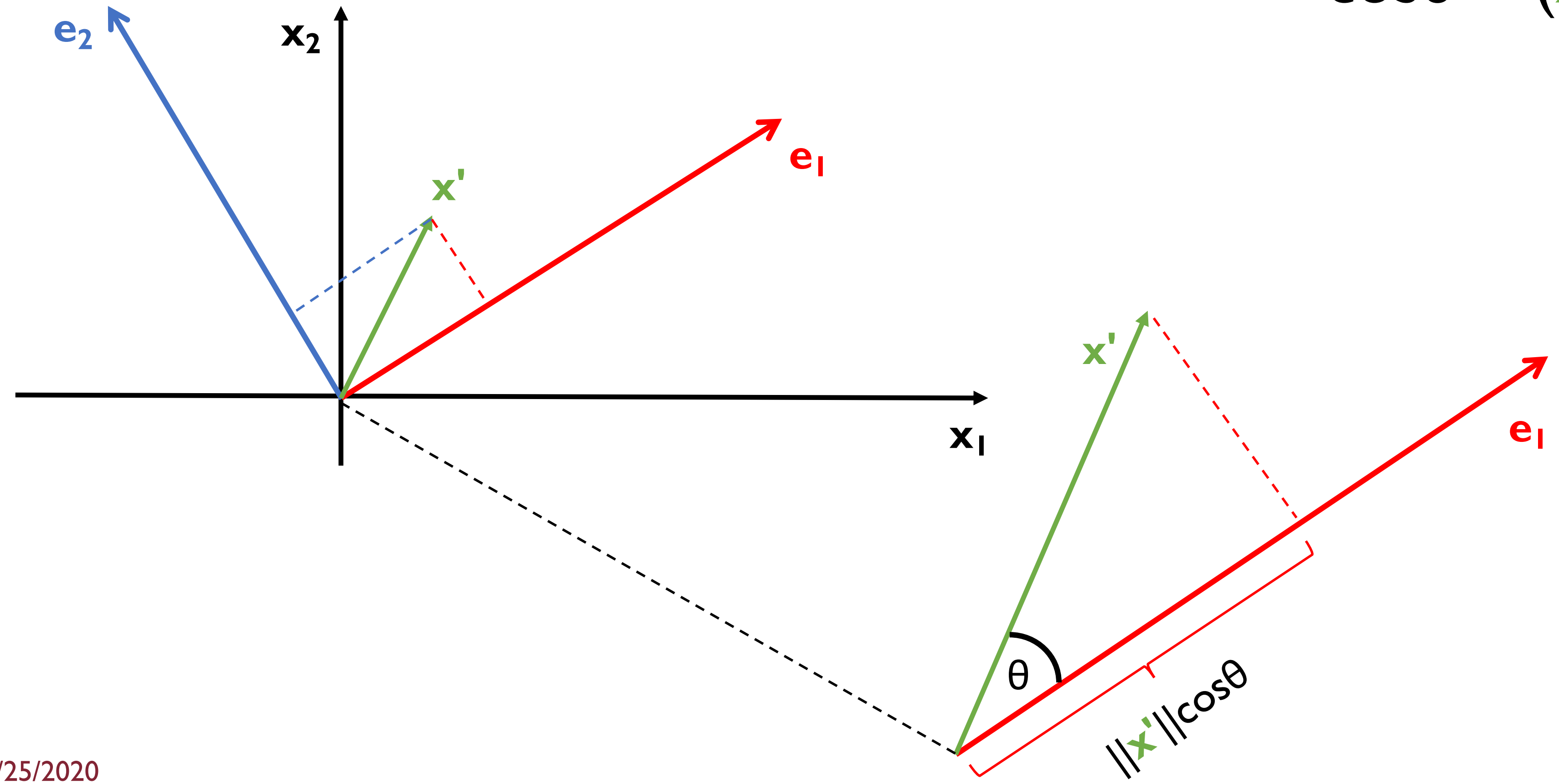
Why the dot product?



Projecting to New Dimensions: 2-d Case

Why the dot product?

$$\cos\theta = (\mathbf{x}'\mathbf{e}_1)/||\mathbf{x}'||||\mathbf{e}_1||$$

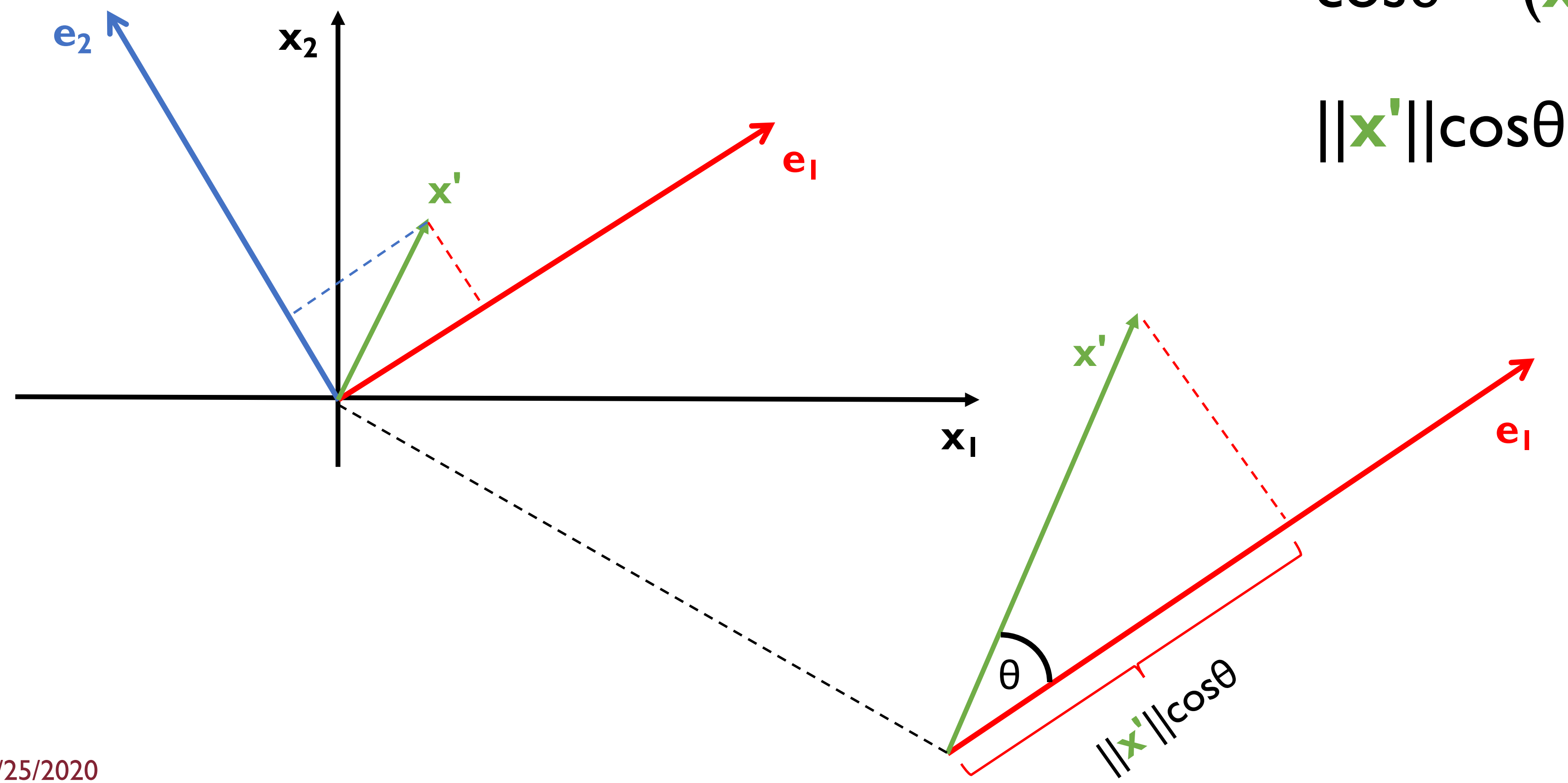


Projecting to New Dimensions: 2-d Case

Why the dot product?

$$\cos\theta = (\mathbf{x}'\mathbf{e}_1)/||\mathbf{x}'||\,||\mathbf{e}_1||$$

$$||\mathbf{x}'||\cos\theta = \mathbf{x}'\mathbf{e}_1/||\mathbf{e}_1||$$



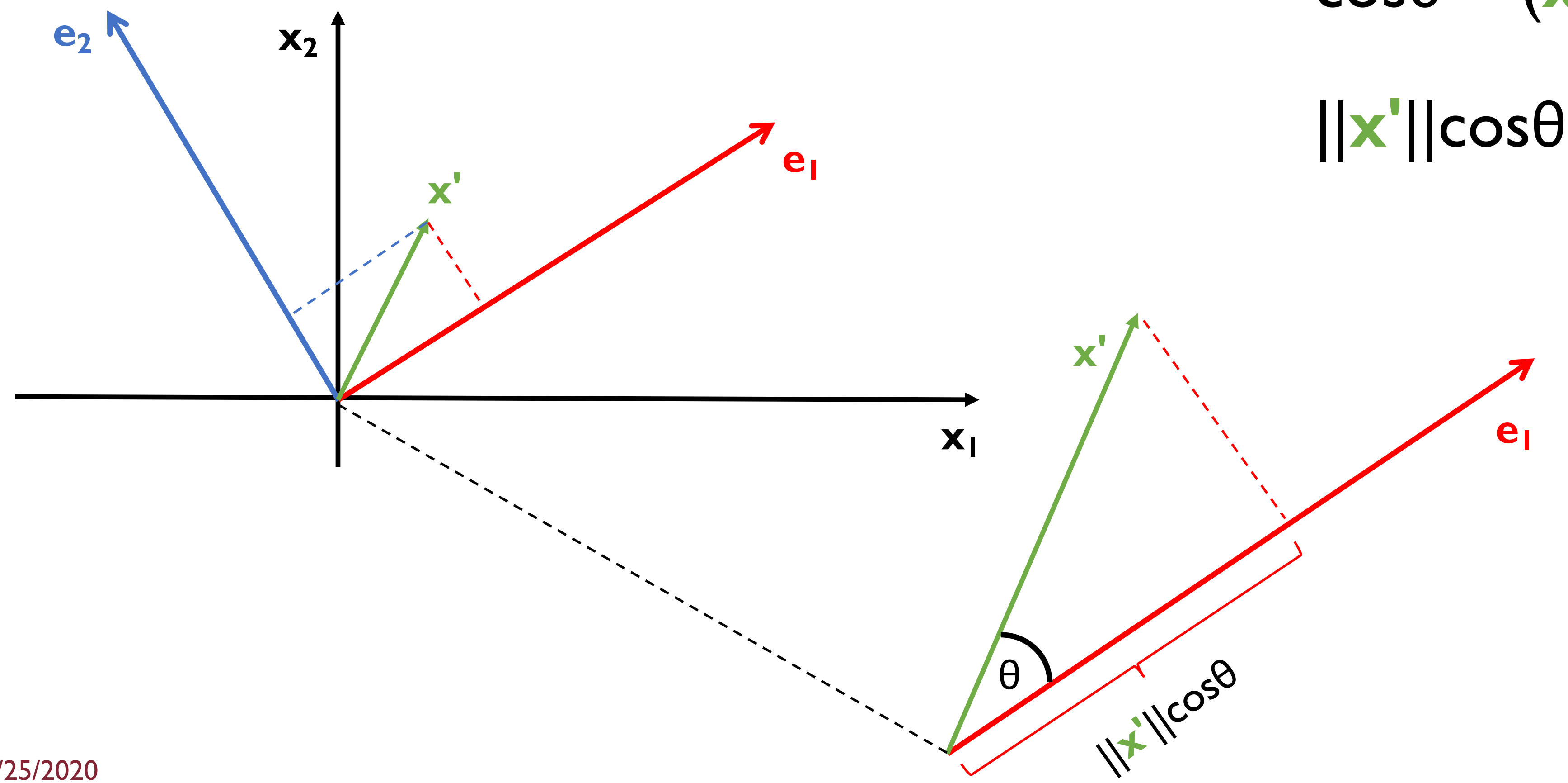
Projecting to New Dimensions: 2-d Case

Why the dot product?

$\cos\theta = (x'e_1)/||x'||\,||e_1||$

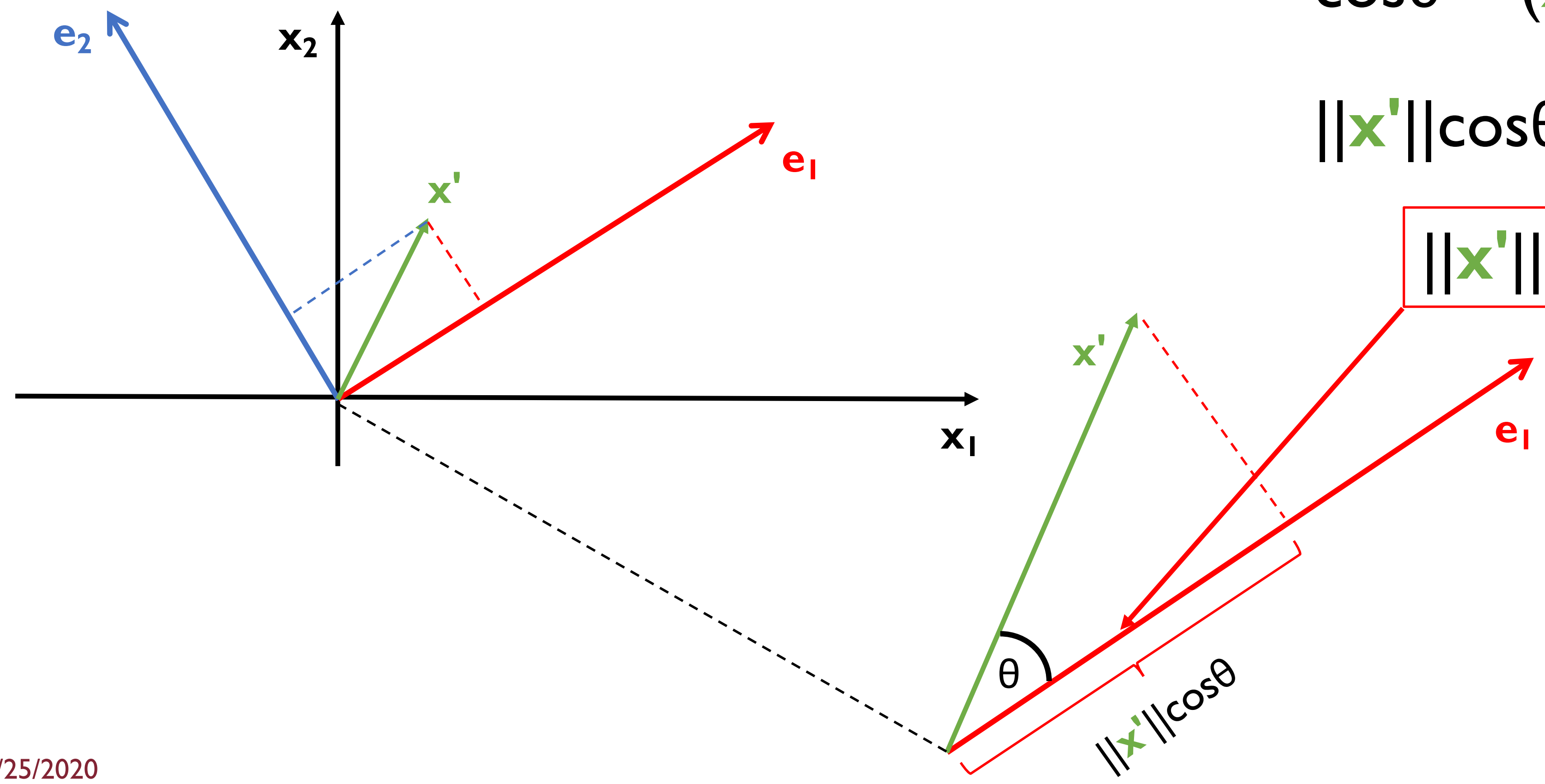
$||x'||\cos\theta = x'e_1/||e_1||$

$||e_1|| = 1$



Projecting to New Dimensions: 2-d Case

Why the dot product?



$$\cos\theta = (\mathbf{x}'\mathbf{e}_1)/\|\mathbf{x}'\|\|\mathbf{e}_1\|$$

$$\|\mathbf{x}'\|\cos\theta = \mathbf{x}'\mathbf{e}_1/\|\mathbf{e}_1\|$$

$$\boxed{\|\mathbf{x}'\|\cos\theta = \mathbf{x}'\mathbf{e}_1}$$

$$\boxed{\|\mathbf{e}_1\| = 1}$$

Projecting to New Dimensions: 2-d Case

The new coordinates of the original data point **x** according to the eigenvectors $\mathbf{e_1}$ and $\mathbf{e_2}$ are as follows:

$$\mathbf{x}' = \begin{bmatrix} x'_1 \\ x'_2 \end{bmatrix} = \begin{bmatrix} \mathbf{x}'^T \mathbf{e}_1 \\ \mathbf{x}'^T \mathbf{e}_2 \end{bmatrix} = \begin{bmatrix} (x_1 - \mu_1)e_{1,1} + (x_2 - \mu_2)e_{1,2} \\ (x_1 - \mu_1)e_{2,1} + (x_2 - \mu_2)e_{2,2} \end{bmatrix}$$

Projecting to New Dimensions: *d*-dimensions

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \end{bmatrix}$$

Original d-dimensional data point

Projecting to New Dimensions: *d*-dimensions

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \end{bmatrix}$$

Original d-dimensional data point

$$\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_k$$
$$k \ll d, \;\; \mathbf{e}_i \in \mathbb{R}^d$$

k << *d*
principal components

Projecting to New Dimensions: *d*-dimensions

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \end{bmatrix}$$

Original d-dimensional data point

$$\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_k$$
$$k \ll d, \ \mathbf{e}_i \in \mathbb{R}^d$$

k << *d*
principal components

1. Mean centering

$$\mathbf{x}' = \mathbf{x} - \boldsymbol{\mu} = \begin{bmatrix} x_1 - \mu_1 \\ x_2 - \mu_2 \\ \vdots \\ x_d - \mu_d \end{bmatrix}$$

Projecting to New Dimensions: *d*-dimensions

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \end{bmatrix}$$

Original d-dimensional data point

$$\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_k$$
$$k \ll d, \ \mathbf{e}_i \in \mathbb{R}^d$$

k << d
principal components

2. Projection to principal components

$$\mathbf{x}' = \begin{bmatrix} x'_1 \\ x'_2 \\ \vdots \\ x'_k \end{bmatrix} = \begin{bmatrix} (\mathbf{x}-\boldsymbol{\mu})^T\mathbf{e}_1 \\ (\mathbf{x}-\boldsymbol{\mu})^T\mathbf{e}_2 \\ \vdots \\ (\mathbf{x}-\boldsymbol{\mu})^T\mathbf{e}_k \end{bmatrix} = \begin{bmatrix} (x_1-\mu_1)e_{1,1} + (x_2-\mu_2)e_{1,2} + \ldots + (x_d-\mu_d)e_{1,d} \\ (x_1-\mu_1)e_{2,1} + (x_2-\mu_2)e_{2,2} + \ldots + (x_d-\mu_d)e_{2,d} \\ \vdots \\ (x_1-\mu_1)e_{k,1} + (x_2-\mu_2)e_{k,2} + \ldots + (x_d-\mu_d)e_{k,d} \end{bmatrix}$$

Why Eigenvector with the Largest Eigenvalue

- We have only "visually proved" that eigenvector **e** turns towards the direction of the greatest variance

Why Eigenvector with the Largest Eigenvalue

- We have only "visually proved" that eigenvector **e** turns towards the direction of the greatest variance
- To actually prove it, we need to show that **e** maximizes the variance among all possible projections

Why Eigenvector with the Largest Eigenvalue

- We have only "visually proved" that eigenvector **e** turns towards the direction of the greatest variance
- To actually prove it, we need to show that **e** maximizes the variance among all possible projections
- Moreover, we pick the eigenvector with the **largest eigenvalue** λ because λ is exactly the variance of the data along that eigenvector

Why Eigenvector with the Largest Eigenvalue

- We have only "visually proved" that eigenvector **e** turns towards the direction of the greatest variance
- To actually prove it, we need to show that **e** maximizes the variance among all possible projections
- Moreover, we pick the eigenvector with the **largest eigenvalue** λ because λ is exactly the variance of the data along that eigenvector

More details available here:
https://github.com/gtolomei/big-data-computing/raw/master/extra/Notes_on_Principal_Component_Analysis.pdf

How Many Dimensions?

- In a d-dimensional space we have $\mathbf{e}_1, \ldots, \mathbf{e}_d$ length-1 eigenvectors

How Many Dimensions?

- In a d-dimensional space we have $\mathbf{e}_1, \ldots, \mathbf{e}_d$ length-1 eigenvectors
- We want to select k dimensions (k << d)

How Many Dimensions?

- In a d-dimensional space we have $\mathbf{e}_1, \ldots, \mathbf{e}_d$ length-1 eigenvectors
- We want to select k dimensions (k << d)
- We know that eigenvalue λ_i is the variance along $\mathbf{e}_i$ and the overall variance of the data is given by the sum of all the eigenvalues

How Many Dimensions?

- In a d-dimensional space we have $\mathbf{e}_1, \ldots, \mathbf{e}_d$ length-1 eigenvectors
- We want to select k dimensions (k << d)
- We know that eigenvalue λ_i is the variance along $\mathbf{e}_i$ and the overall variance of the data is given by the sum of all the eigenvalues

⇓

Pick the subset of k eigenvectors that "explain" the most variance

How Many Dimensions?

1. Sort eigenvectors by eigenvalues such that $\lambda_1 \geqslant \lambda_2 \geqslant \ldots \geqslant \lambda_d$

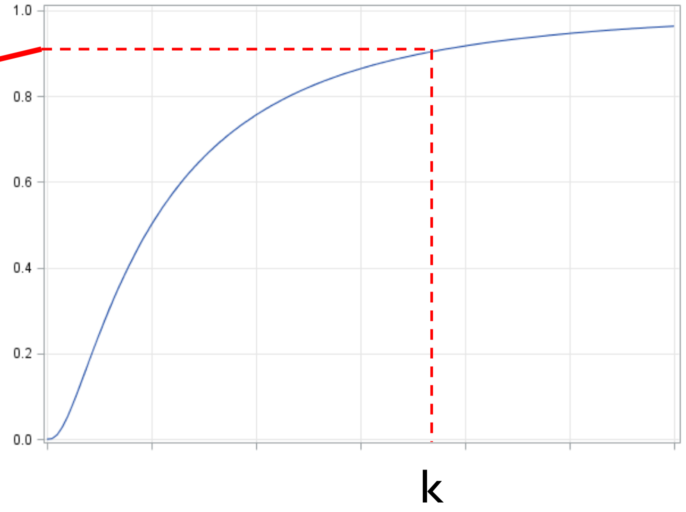
How Many Dimensions?

1. Sort eigenvectors by eigenvalues such that $\lambda_1 \geqslant \lambda_2 \geqslant \ldots \geqslant \lambda_d$

2. Pick the first k eigenvectors that explain *x*% of the total variance

$$\frac{\sum_{i=1}^{k} \lambda_i}{\sum_{i=1}^{d} \lambda_i} \leq 1$$

e.g., *x* = 90÷95%



Practical Issues of PCA

- Covariance and variance are highly sensitive to large values

Practical Issues of PCA

- Covariance and variance are highly sensitive to large values
- If one dimension takes on extremely large values w.r.t. other dimensions the former will be considered as the principal component

Practical Issues of PCA

- Covariance and variance are highly sensitive to large values
- If one dimension takes on extremely large values w.r.t. other dimensions the former will be considered as the principal component

Solution

Normalize each dimension to 0-mean and 1-std-deviation

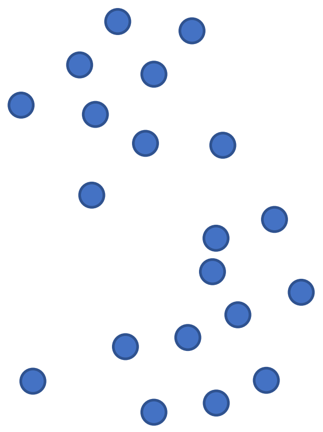
$$z = \frac{x - \mu}{\sigma}$$

Practical Issues of PCA

- PCA assumes the projection subspace is linear, i.e., an hyperplane:
  - 1-d → straight line, 2-d → flat surface, …

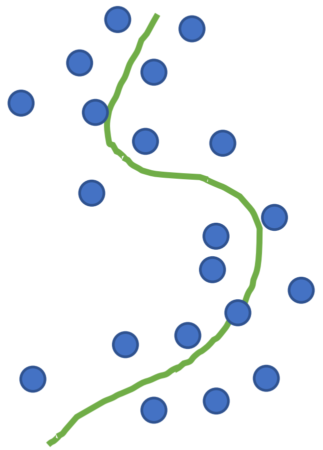
Practical Issues of PCA

- PCA assumes the projection subspace is linear, i.e., an hyperplane:
  - 1-d → straight line, 2-d → flat surface, …
- If data live in a low-dimensional but not linear space (i.e., manifold), PCA can still be applied but may not work nicely



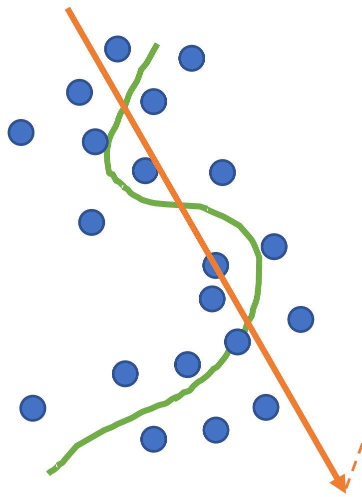
Practical Issues of PCA

- PCA assumes the projection subspace is linear, i.e., an hyperplane:
  - 1-d → straight line, 2-d → flat surface, …
- If data live in a low-dimensional but not linear space (i.e., manifold), PCA can still be applied but may not work nicely



Practical Issues of PCA

- PCA assumes the projection subspace is linear, i.e., an hyperplane:
  - 1-d → straight line, 2-d → flat surface, …
- If data live in a low-dimensional but not linear space (i.e., manifold), PCA can still be applied but may not work nicely



PCA will find a straight line and will not mimic non-linearity

Take-Home Message of Today

- PCA is a dimensionality reduction technique which allows representing high-dimensional data into low-dimensional linear subspace
  - The original space is rotated to make dimensions uncorrelated (i.e., independent)

Take-Home Message of Today

- PCA is a dimensionality reduction technique which allows representing high-dimensional data into low-dimensional linear subspace
  - The original space is rotated to make dimensions uncorrelated (i.e., independent)
- Reduced size of data means faster processing and smaller storage

Take-Home Message of Today

- PCA is a dimensionality reduction technique which allows representing high-dimensional data into low-dimensional linear subspace
  - The original space is rotated to make dimensions uncorrelated (i.e., independent)
- Reduced size of data means faster processing and smaller storage
- PCA can be very expensive for many very large-scale applications

Take-Home Message of Today

- PCA is a dimensionality reduction technique which allows representing high-dimensional data into low-dimensional linear subspace
  - The original space is rotated to make dimensions uncorrelated (i.e., independent)
- Reduced size of data means faster processing and smaller storage
- PCA can be very expensive for many very large-scale applications
- If data do not live on a linear subspace PCA may not work well