

# Aprendizaje Automático

Based on material by Andrew Ng from Stanford University (Machine Learning course)

## Dimensionality Reduction



Máster en Bioinformática y Biología Computacional

# Dimensionality Reduction

Dimensionality Reduction algorithms:

- Map high-dimensional data to a lower dimension
- While preserving structure

They are used for:

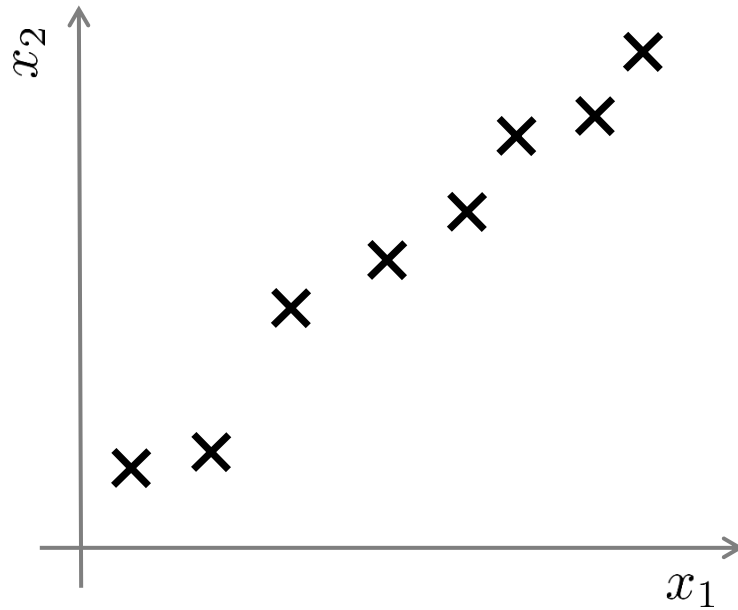


- Visualization
- Performance
- Curse of dimensionality

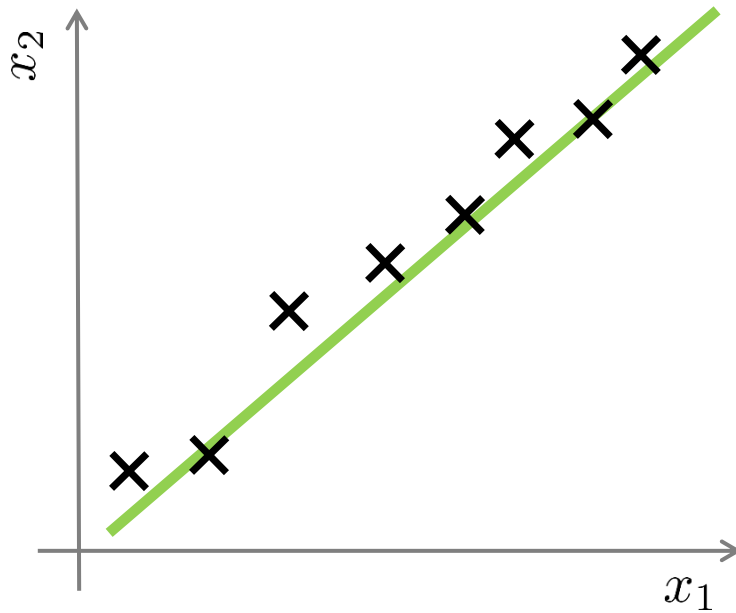
A ton of algorithms exist:

- t-SNE is specialised for visualization
- ... and has gained a lot of popularity

# From 2D to 1D

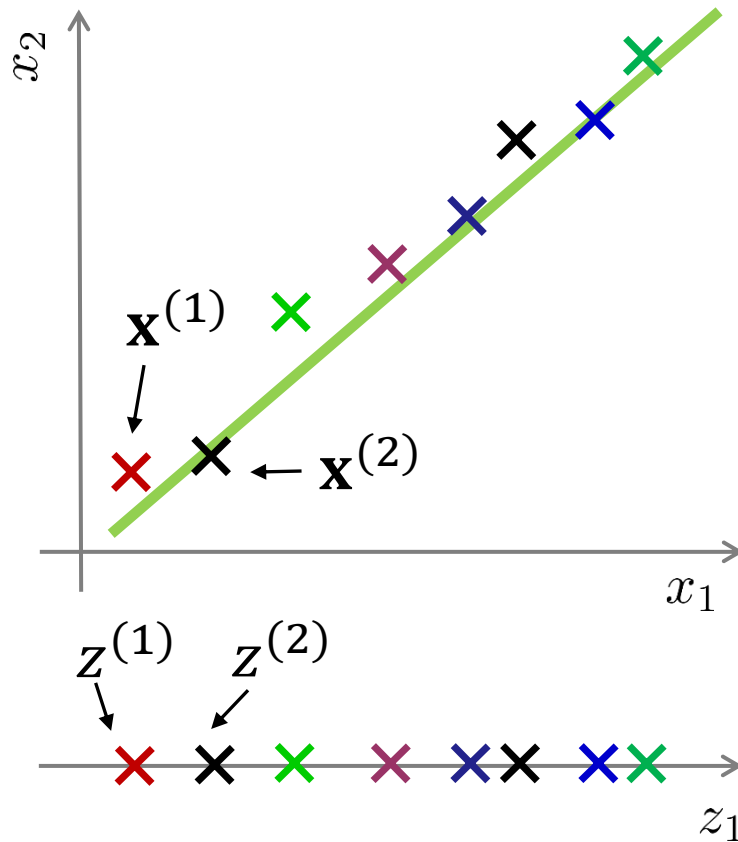


# From 2D to 1D



Reduce correlation of feats

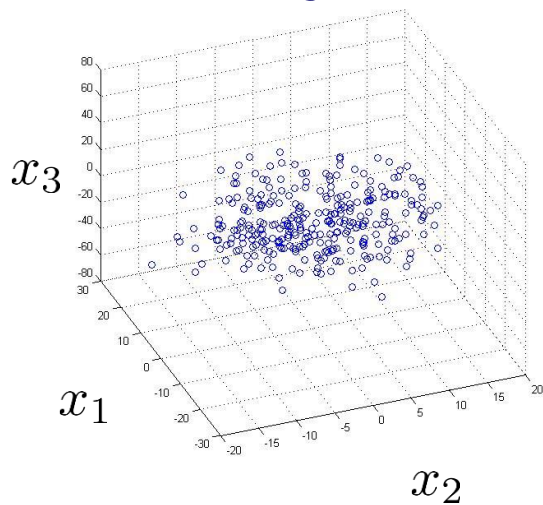
# From 2D to 1D



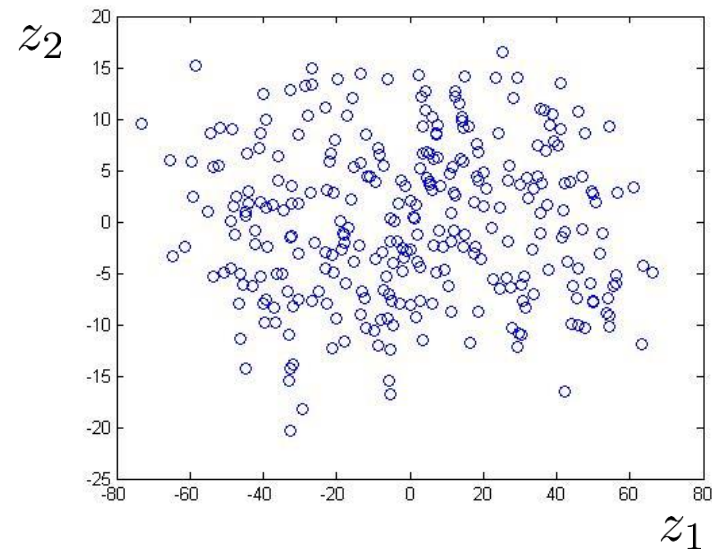
$$\begin{aligned}\mathbf{x}^{(1)} &\in \mathbb{R}^2 \rightarrow z^{(1)} \in \mathbb{R} \\ \mathbf{x}^{(2)} &\in \mathbb{R}^2 \rightarrow z^{(2)} \in \mathbb{R} \\ &\vdots \\ \mathbf{x}^{(N)} &\in \mathbb{R}^2 \rightarrow z^{(N)} \in \mathbb{R}\end{aligned}$$

# From 3D to 2D

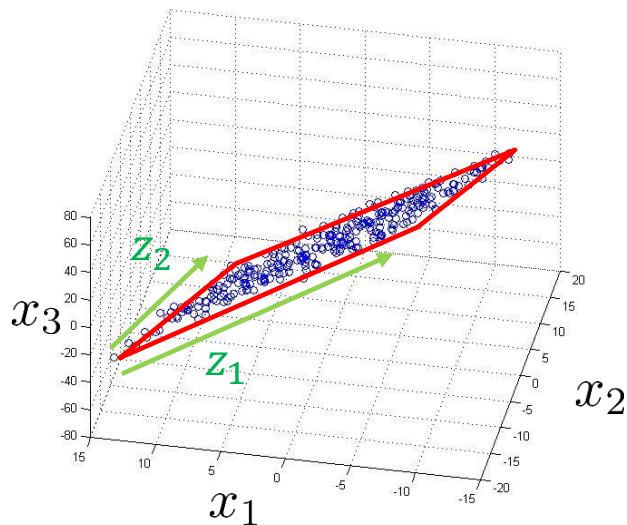
Original dataset



Reduce data from 3D to 2D



Projected dataset



$$\mathbf{x}^{(i)} \in \mathbb{R}^3 \rightarrow \mathbf{z}^{(i)} \in \mathbb{R}^2$$

# Data Visualization

	$x_1$	$x_2$	$x_3$	$x_4$	$x_5$	$x_6$	$x_{2000}$
Sub_ID	rs307377	rs7366653	rs41307846	rs3753242	rs35082957	rs34154371	...
HG00001	1	0	1	1	0	0	...
HG00002	0	0	1	1	1	0	...
HG00003	1	1	0	0	0	1	...
HG00004	0	0	0	1	0	1	...
HG00005	0	0	1	1	1	1	...
...							

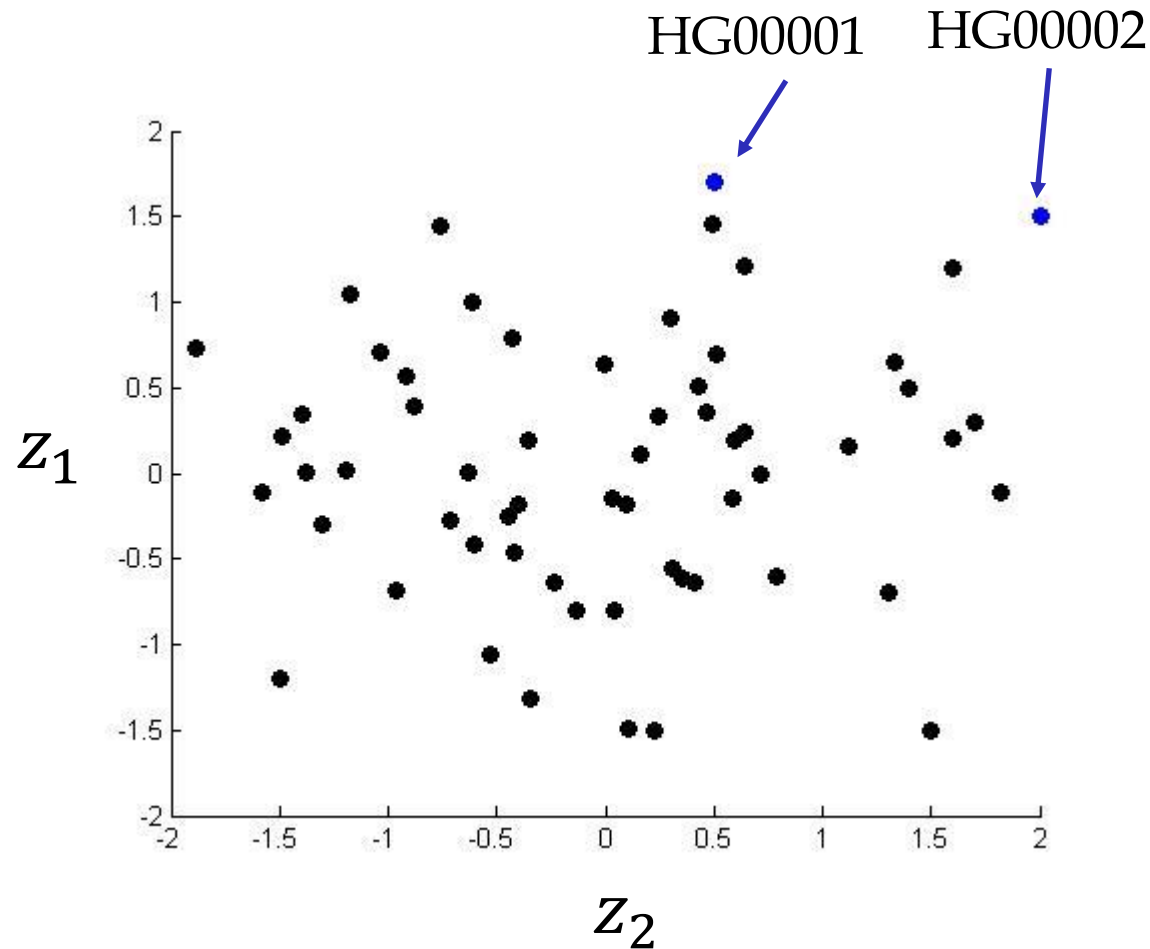
$$\mathbf{x}^{(i)} \in \mathbb{B}^{2000}$$

How can we understand our data better?  
How can we reduce from 2000D to 2D

# Data Visualization

$$\mathbf{z}^{(i)} \in \mathbb{R}^2$$

Sub_ID	$z_1$	$z_2$
HG00001	0.65	0.71
HG00002	0.43	2.43
HG00003	0.03	1.14
HG00004	5.40	2.11
HG00005	2.33	0.46
...		

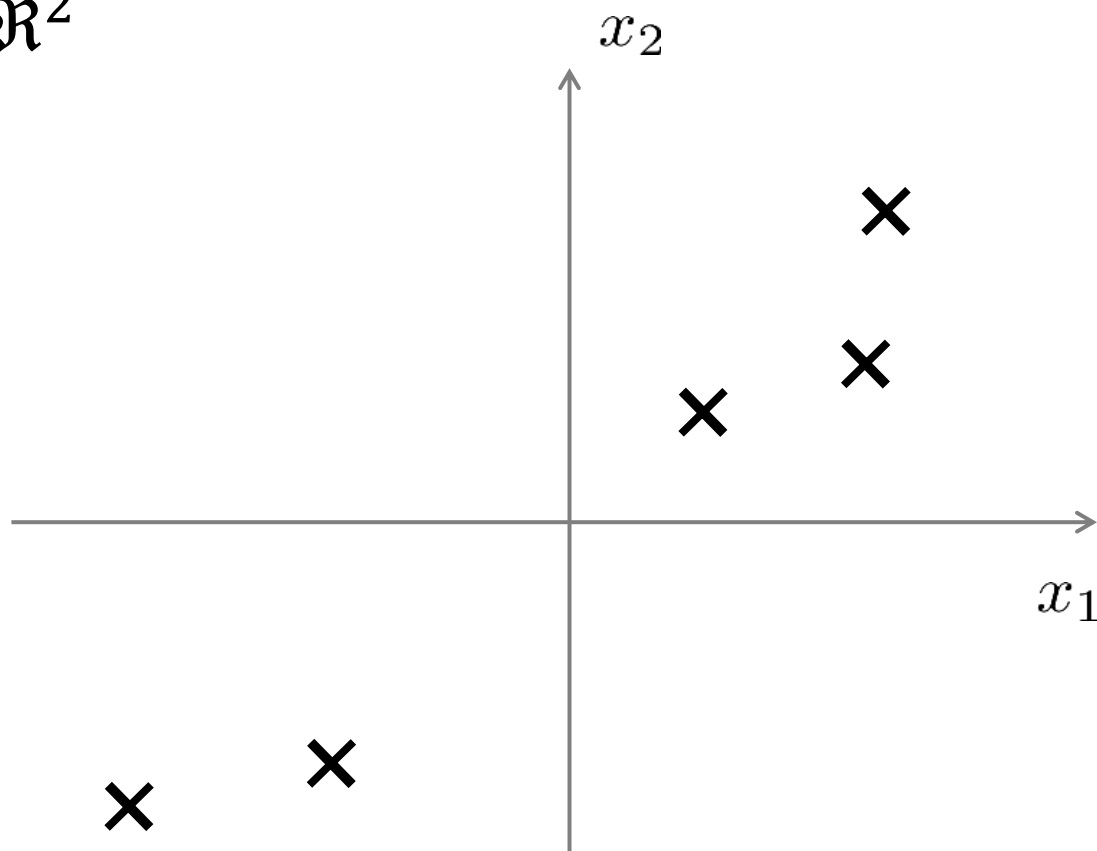


Drawback, the new axes do not have any physical meaning



# Principal Component Analysis: Formulation

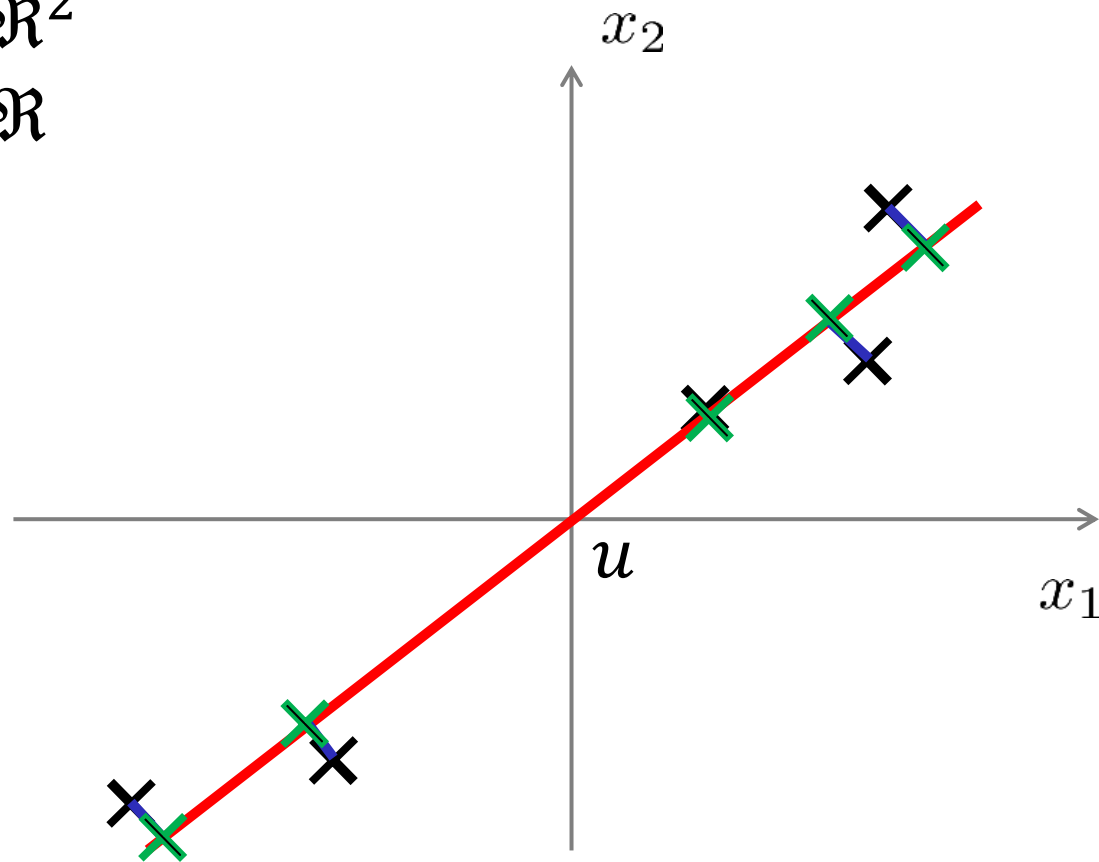
$$\mathbf{x}^{(i)} \in \mathbb{R}^2$$



# Principal Component Analysis: Formulation

$$\mathbf{x}^{(i)} \in \mathbb{R}^2$$

$$u^{(i)} \in \mathbb{R}$$

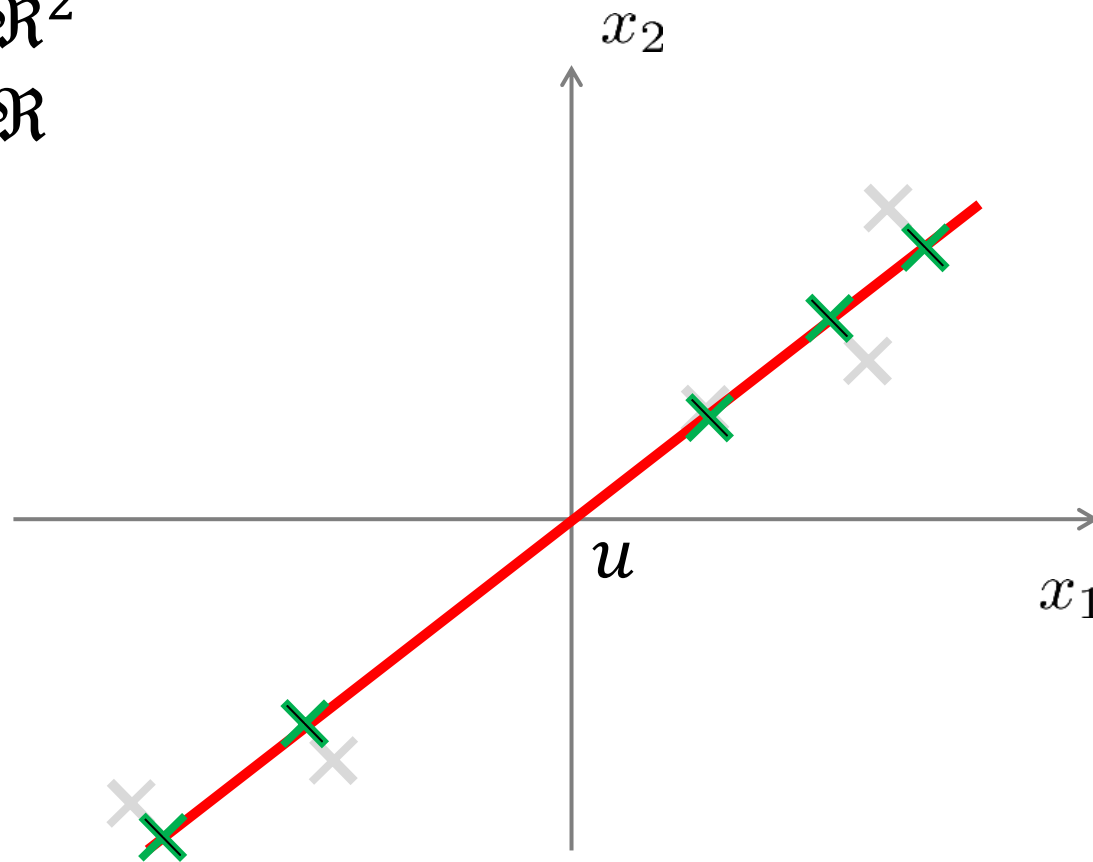


Projection to a line

# Principal Component Analysis: Formulation

$$\mathbf{x}^{(i)} \in \mathbb{R}^2$$

$$u^{(i)} \in \mathbb{R}$$

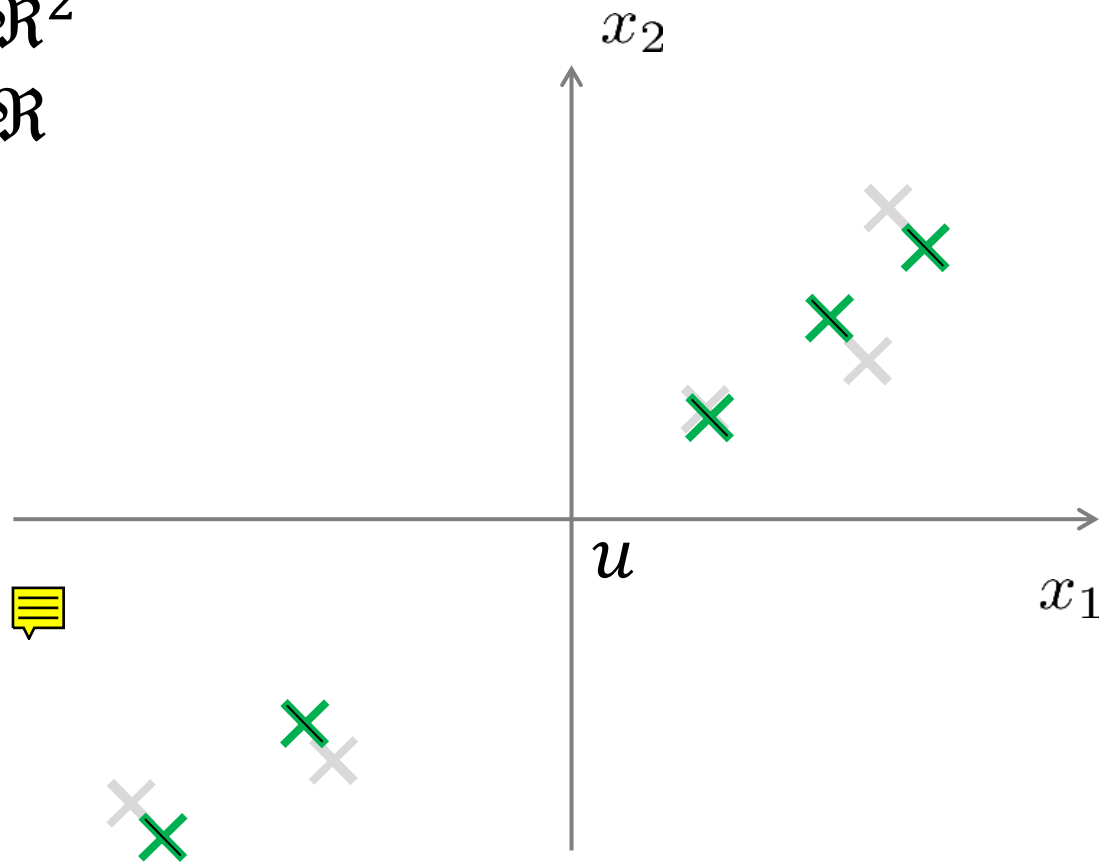


PCA tries to minimize the projection “error”

# Principal Component Analysis: Formulation

$$\mathbf{x}^{(i)} \in \mathbb{R}^2$$

$$u^{(i)} \in \mathbb{R}$$

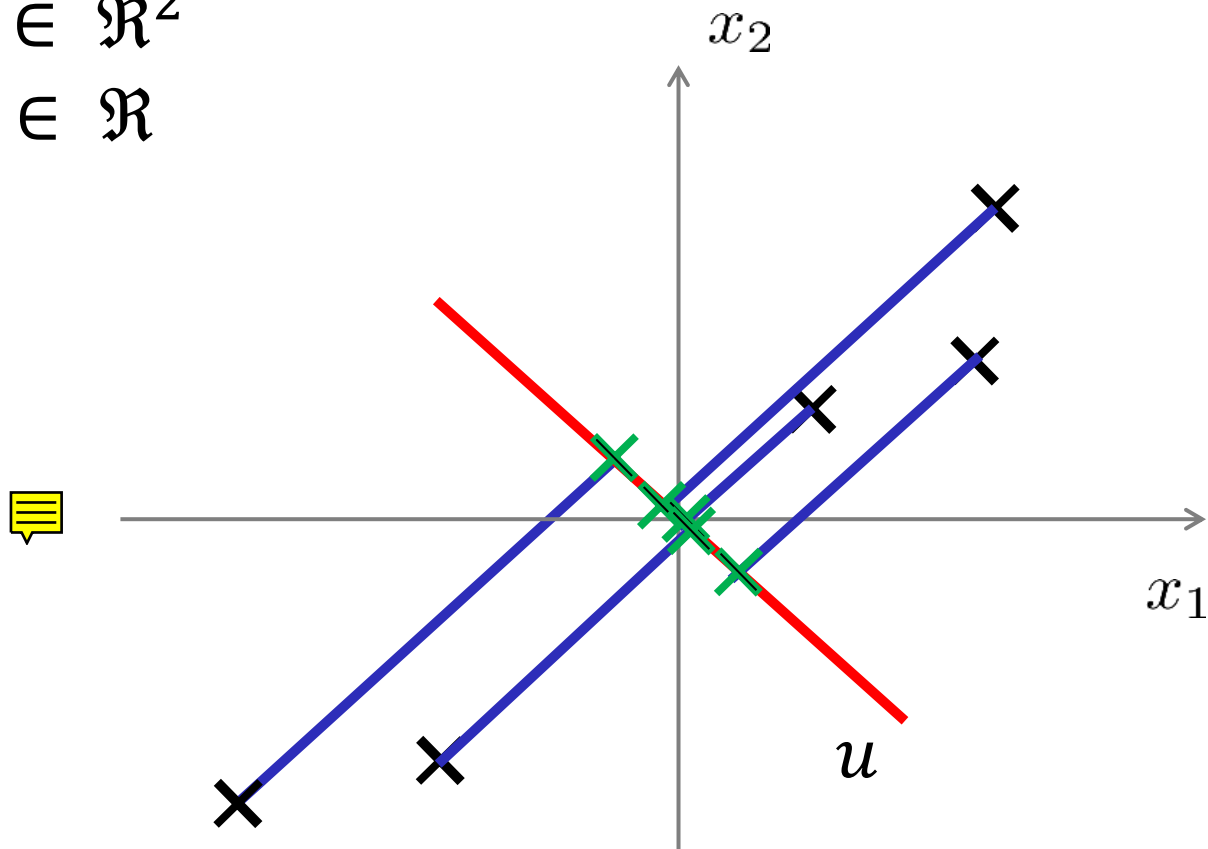


PCA tries to minimize the projection “error”

# Principal Component Analysis: Formulation

$$\mathbf{x}^{(i)} \in \mathbb{R}^2$$

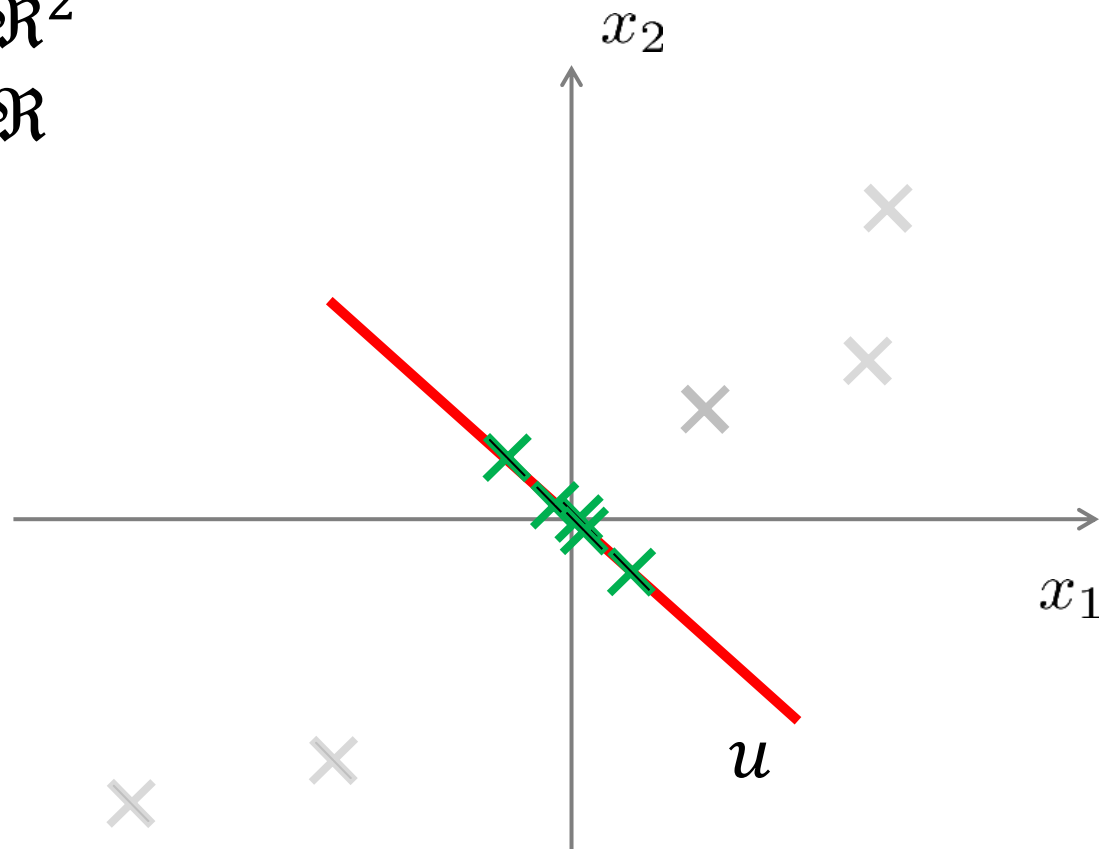
$$u^{(i)} \in \mathbb{R}$$



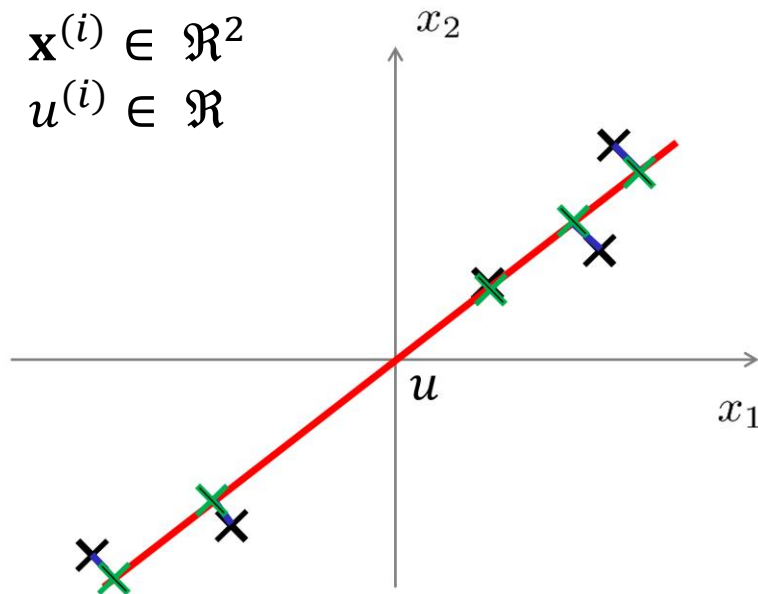
# Principal Component Analysis: Formulation

$$\mathbf{x}^{(i)} \in \mathbb{R}^2$$

$$u^{(i)} \in \mathbb{R}$$



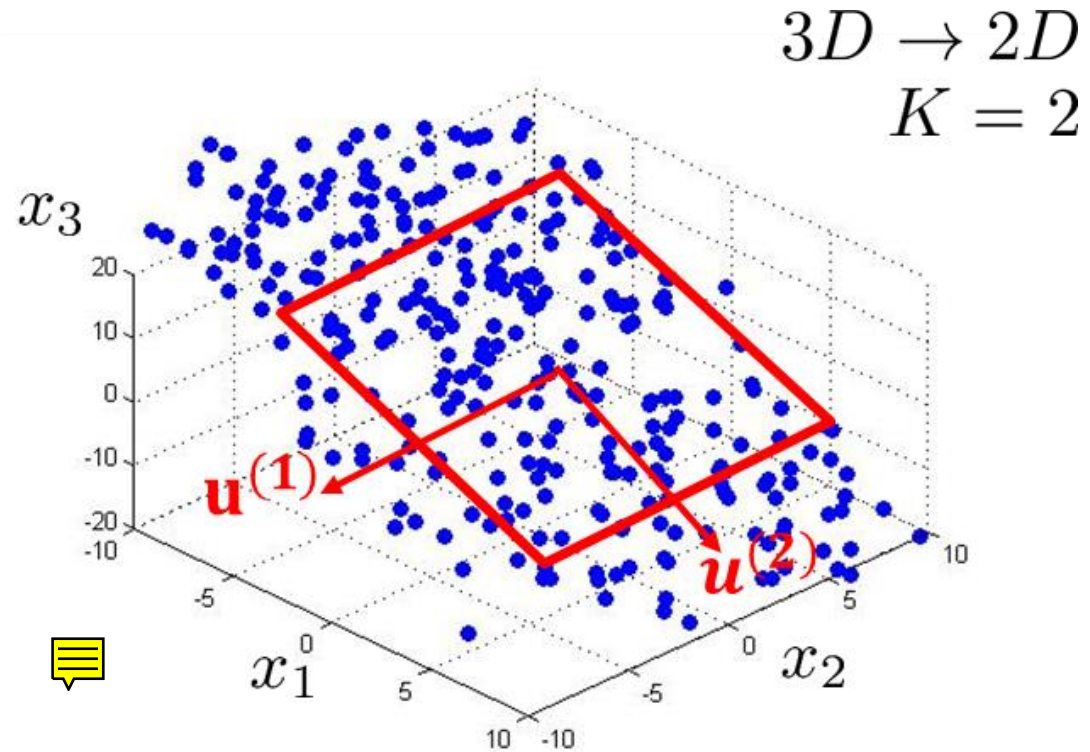
# Principal Component Analysis: Formulation



- Reduce from 2-D to 1-D:  
Find a direction (a vector  $u^{(1)} \in \mathbb{R}$ ) onto which to project the data so as to minimize the projection error.

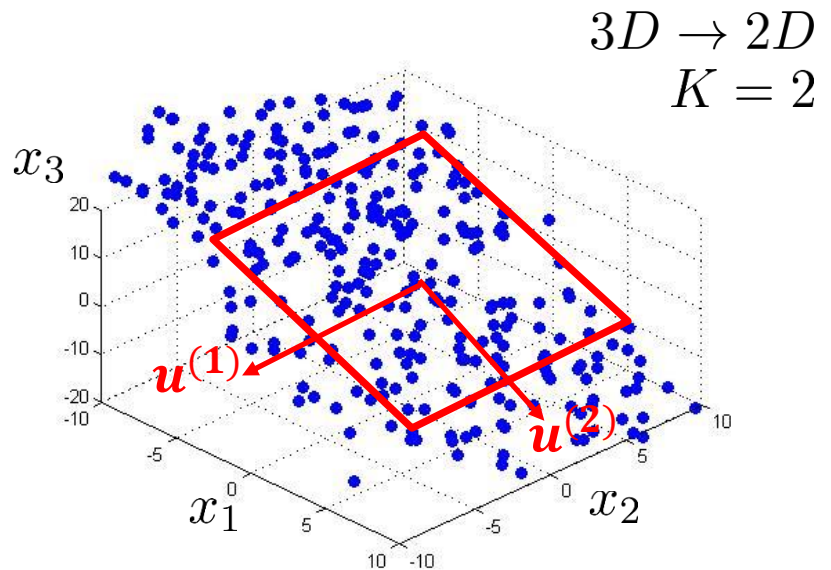
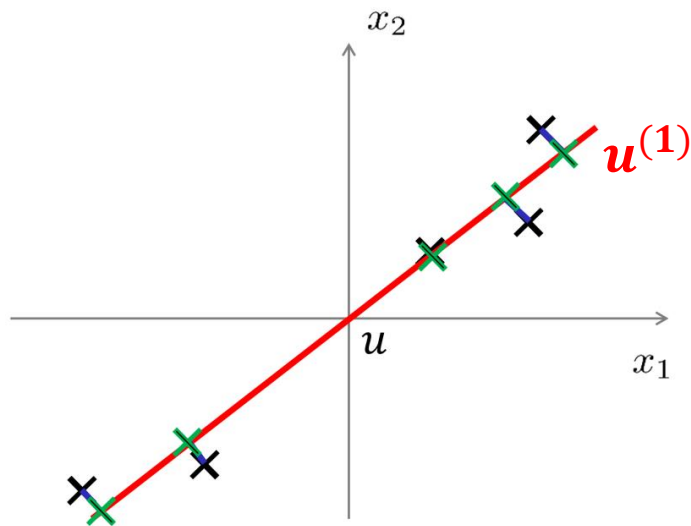
# Principal Component Analysis: Formulation

- Reduce from  $n$ -dimension to  $k$ -dimension: Find  $k$  vectors  $u^{(1)}, u^{(2)}, u^{(3)}, \dots, u^{(k)}$  onto which to project the data, so as to minimize the projection error.





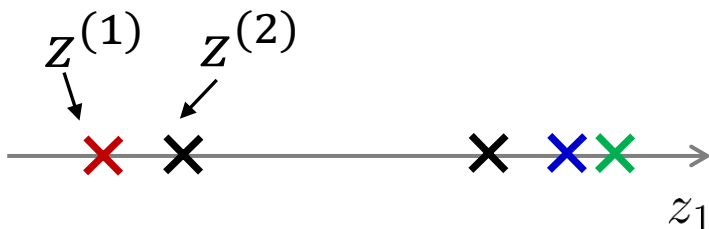
# PCA: Algorithm



$3D \rightarrow 2D$   
 $K = 2$

Reduce data from 2D to 1D

$$\mathbf{x}^{(i)} \in \mathbb{R}^2 \rightarrow z^{(i)} \in \mathbb{R}$$



Reduce data from 3D to 2D

$$\mathbf{x}^{(i)} \in \mathbb{R}^3 \rightarrow \mathbf{z}^{(i)} \in \mathbb{R}^2$$

$$\mathbf{z} = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}$$

# Principal Component Analysis: Algorithm



- Training set:  $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \dots, \mathbf{x}^{(M)}$
- Data Preprocessing (feature scaling/mean normalization):

$$\mu_j = \frac{1}{M} \sum_{i=1}^M x_j^{(i)}$$

- If different features on different scales (e.g.,  $x_1$  = area,  $x_2$  = growth rate), scale features to have comparable range of values.

$$x_j^{(i)} = \frac{x_j^{(i)} - \mu_j}{\sigma_j}$$

- Where, the standard deviation:


$$\sigma_j = \sqrt{\frac{1}{M} \sum_{i=1}^M (x_j^{(i)} - \mu_j)^2}$$

# Principal Component Analysis: Algorithm

- Compute “covariance matrix” of  $\mathbf{x}$  :

$$\Sigma = \frac{1}{M} \sum_{i=1}^M (\mathbf{x}^{(i)}) (\mathbf{x}^{(i)})^T \in \mathbb{R}^{n \times n}$$

- Implementation. If we have the  $\mathbf{X}$  matrix defined as:

$$\mathbf{X} = \begin{bmatrix} - & (\mathbf{x}^{(1)})^T & - \\ & \vdots & \\ - & (\mathbf{x}^{(M)})^T & - \end{bmatrix} \in \mathbb{R}^{M \times n} \rightarrow \Sigma = \left(\frac{1}{M}\right) \times \mathbf{X}^T \times \mathbf{X}$$


# Principal Component Analysis: Algorithm

- Compute “eigenvalues” of matrix  $\Sigma$ :



$$\boldsymbol{\lambda} = \begin{bmatrix} \lambda_1 \\ \vdots \\ \lambda_n \end{bmatrix} \in \mathbb{R}^{n \times 1}$$

- Compute the “eigenvectors” of matrix  $\Sigma$ :

$$\mathbf{U} = \begin{bmatrix} | & & | \\ \mathbf{u}^{(1)} & \dots & \mathbf{u}^{(n)} \\ | & & | \end{bmatrix} \in \mathbb{R}^{n \times n}$$

# Principal Component Analysis: Algorithm

- Order “eigenvectors” according to its “eigenvalues” from higher to lower values. It means higher significance to lower significance

$$\mathbf{U} = \begin{bmatrix} | & & | \\ \mathbf{u}^{(1)} & \dots & \mathbf{u}^{(n)} \\ | & & | \end{bmatrix} \in \mathbb{R}^{n \times n}$$
$$\boldsymbol{\lambda}' = \begin{bmatrix} \lambda'_1 \\ \vdots \\ \lambda'_n \end{bmatrix} \in \mathbb{R}^{n \times 1} \quad \lambda'_1 > \lambda'_2 > \dots > \lambda'_n \quad \Downarrow$$
$$\mathbf{U}' = \begin{bmatrix} | & & | \\ (\mathbf{u}')^{(1)} & \dots & (\mathbf{u}')^{(n)} \\ | & & | \end{bmatrix} \in \mathbb{R}^{n \times n}$$

# Principal Component Analysis: Algorithm

- Select the components with higher significance:

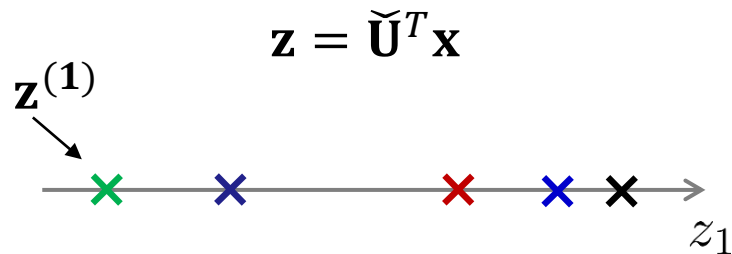
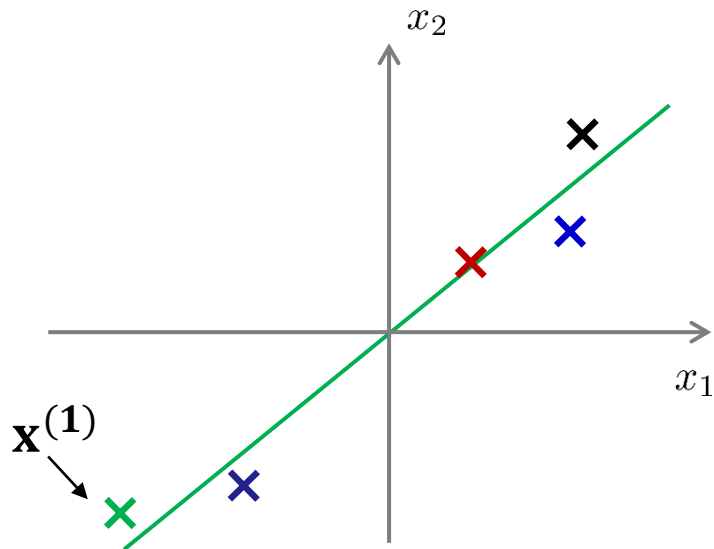
$$\mathbf{U}' = \left[ \begin{array}{c|c|c} | & & | \\ (\mathbf{u}')^{(1)} & \dots & (\mathbf{u}')^{(n)} \\ | & & | \end{array} \right] \in \mathfrak{R}^{n \times n}$$

$$\mathbf{x} \in \mathfrak{R}^n \rightarrow \mathbf{z} \in \mathfrak{R}^k$$

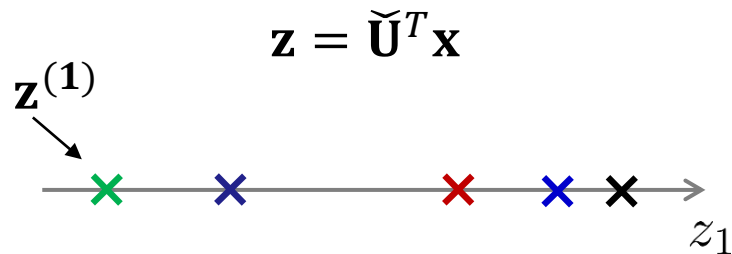
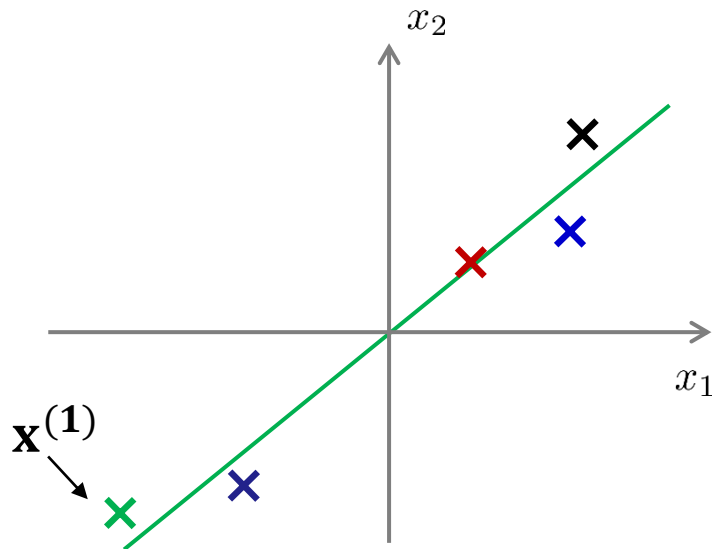
$$\mathbf{z} = \underbrace{\left[ \begin{array}{c|c|c} | & & | \\ (\mathbf{u}')^{(1)} & \dots & (\mathbf{u}')^{(k)} \\ | & & | \end{array} \right]}_{n \times k}^T \underbrace{\mathbf{x}}_{n \times 1} = \underbrace{\left[ \begin{array}{c|c|c} - & ((\mathbf{u}')^{(1)})^T & - \\ & \vdots & \\ - & ((\mathbf{u}')^{(k)})^T & - \end{array} \right]}_{k \times n} \underbrace{\mathbf{z}}_{n \times 1} = \underbrace{\mathbf{x}}_{k \times 1} \in \mathfrak{R}^k$$

$\tilde{\mathbf{U}}^T = \mathbf{U}'^T$  reduced

# Reconstruction from compressed representation



# Reconstruction from compressed representation



$$\mathbf{z} \in \mathbb{R} \rightarrow \mathbf{x} \in \mathbb{R}^2$$

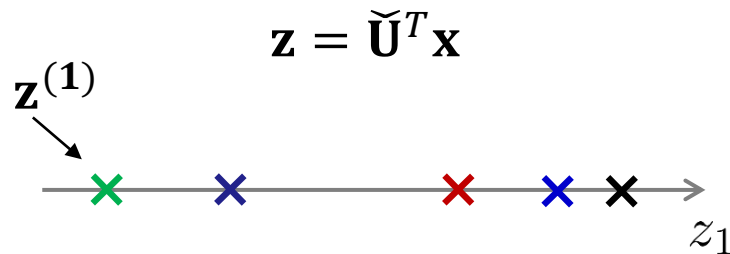
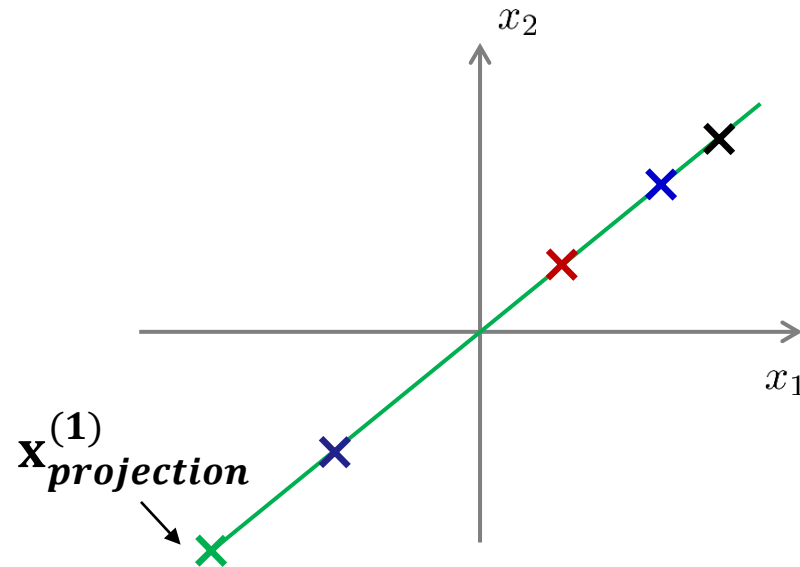
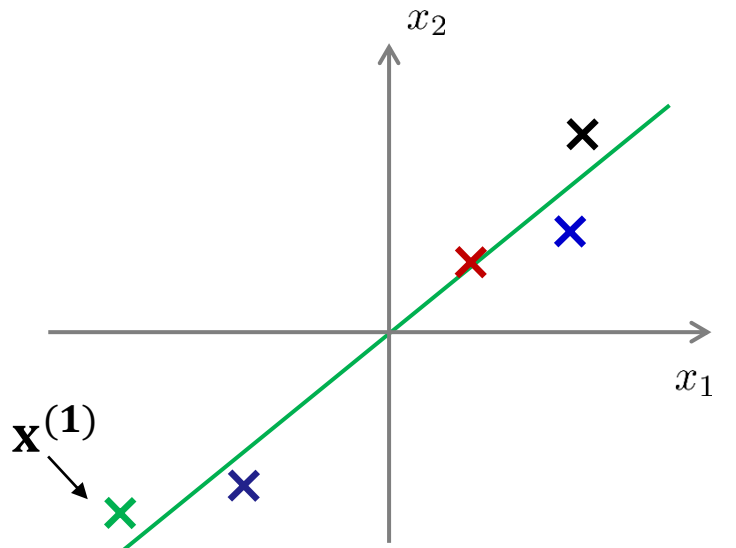


$$\mathbf{x}_{\text{projection}} = \tilde{\mathbf{U}} \mathbf{z} \approx \mathbf{x}$$

$n \times 1$        $n \times k$        $k \times 1$



# Reconstruction from compressed representation




$$\mathbf{z} \in \mathbb{R} \rightarrow \mathbf{x} \in \mathbb{R}^2$$

$$\mathbf{x}_{projection} = \tilde{\mathbf{U}} \mathbf{z} \approx \mathbf{x}$$

$\swarrow \quad \quad \quad \nwarrow \quad \quad \quad \swarrow$   
 $n \times 1 \quad \quad n \times k \quad \quad k \times 1$

# Choosing the number of principal components

-  Average squared projection error =  $\frac{1}{M} \sum_{i=1}^M \underbrace{\left\| \mathbf{x}^{(i)} - \mathbf{x}_{projection}^{(i)} \right\|^2}_{\text{Variance between data and projections}}$   
(what PCA minimizes)
- Total variation in the data =  $\frac{1}{M} \sum_{i=1}^M \left\| \mathbf{x}^{(i)} \right\|^2$   
(the average length squared of the examples)
- Typically, choose  $k$  to be smallest value so that:

$$\frac{\frac{1}{M} \sum_{i=1}^M \left\| \mathbf{x}^{(i)} - \mathbf{x}_{projection}^{(i)} \right\|^2}{\frac{1}{M} \sum_{i=1}^M \left\| \mathbf{x}^{(i)} \right\|^2} \leq 0.01$$

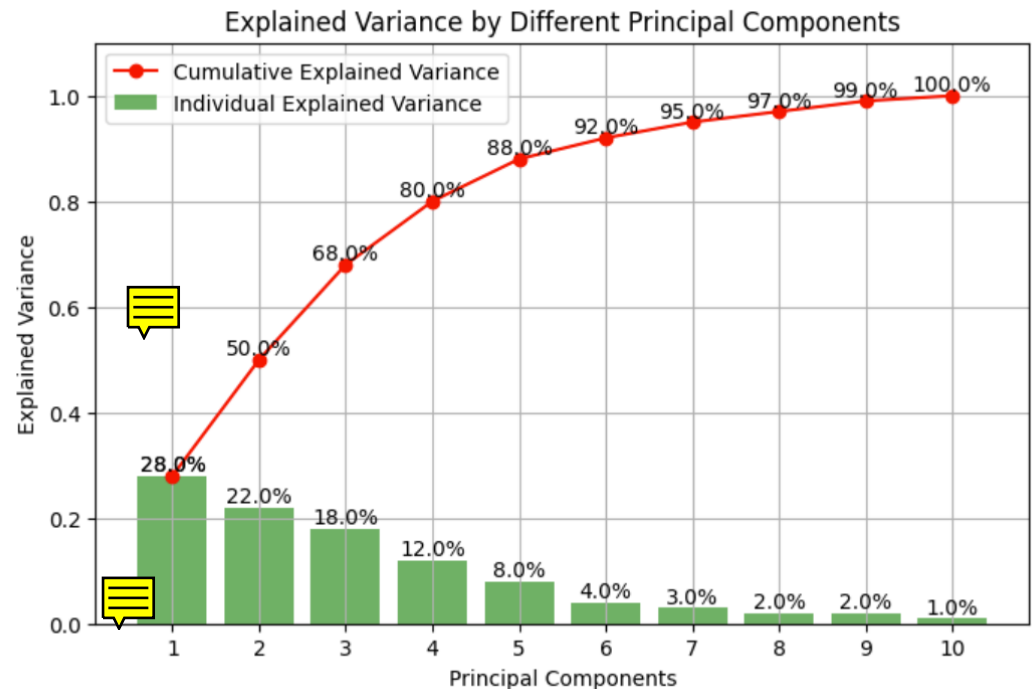
“99% of variance is retained”

# Choosing the number of principal components

- Using the eigenvalues to estimate the variance retained in each component:

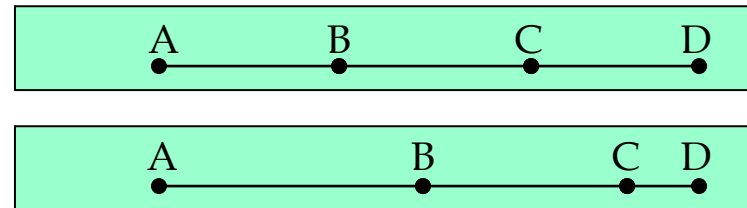
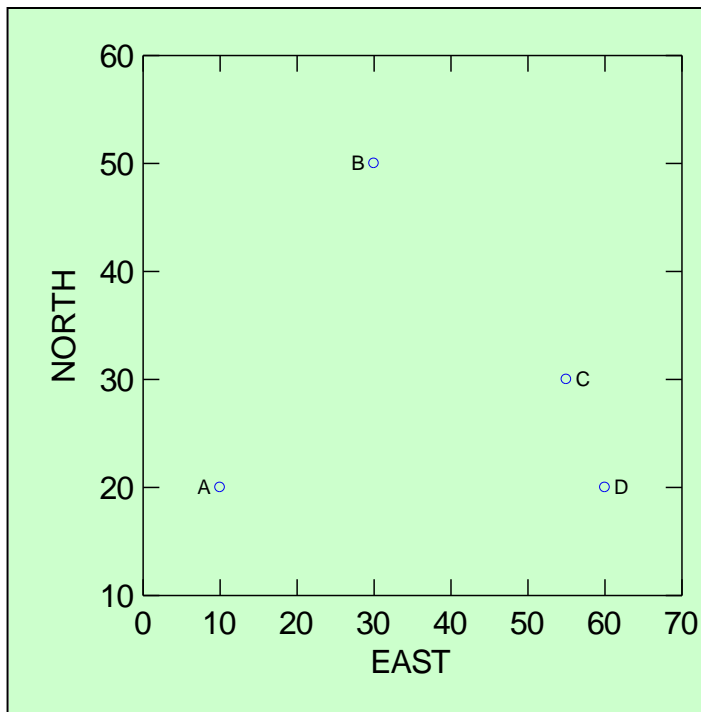
- Variance explained by the component  $j = \frac{\lambda_j}{\sum_{i=1}^n \lambda_i}$

- Plot the cumulative variance ratio:



# Dimensionality Reduction: Other Methods

- **Multidimensional Scaling (MDS)**: define low-dimension space that preserves the distance between cases in original high-dimension space.

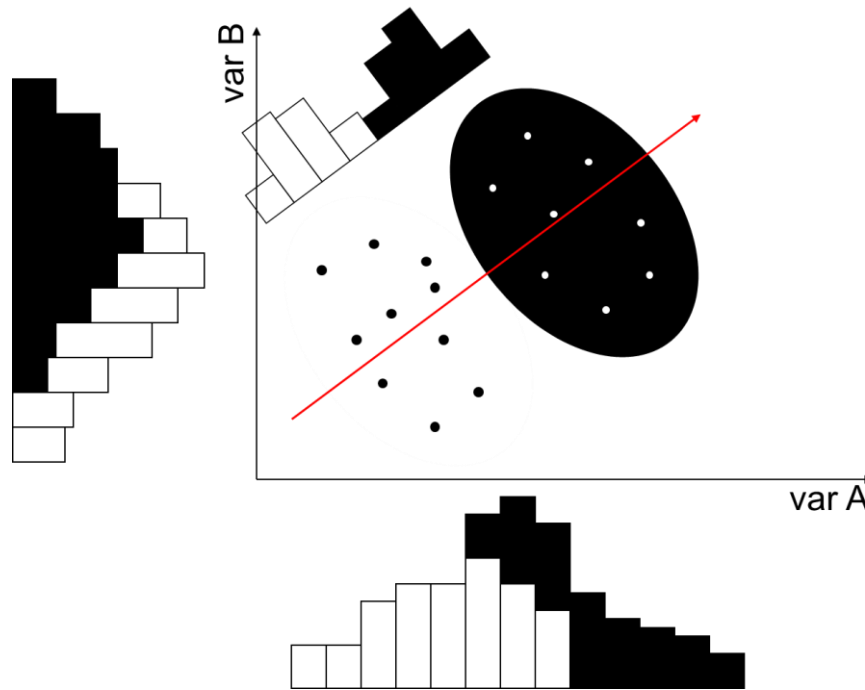


'non-metric' MDS

'metric' MDS

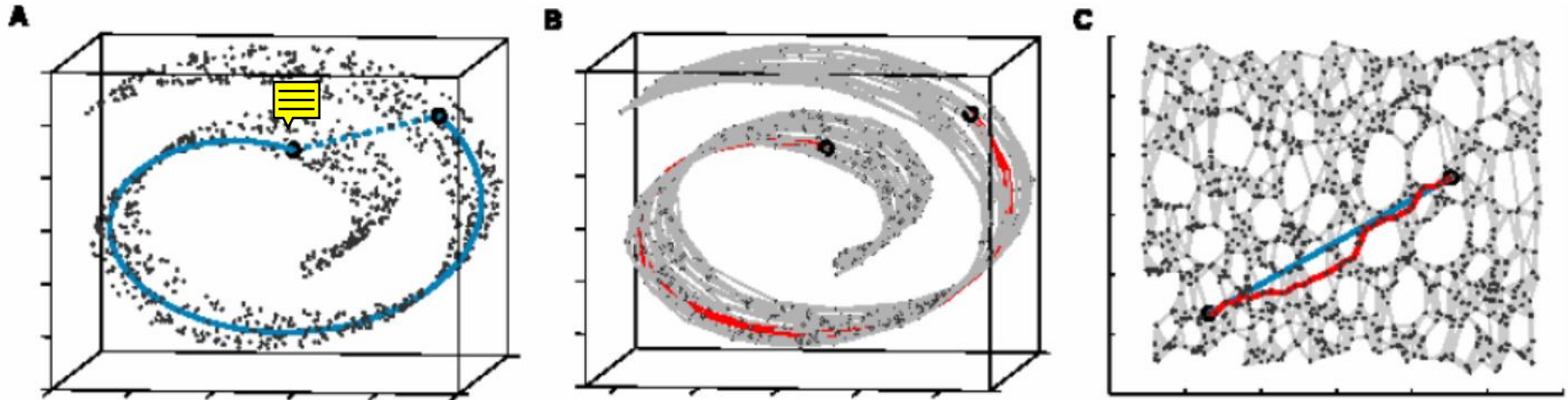
# Dimensionality Reduction: Other Methods

- **Discriminant Analysis (e.g., LDA):** calculate a function that maximizes the ability to discriminate among 2 or more groups.



# Dimensionality Reduction: Other Methods

- **Manifold Learning (e.g., Isomap):** discover low dimensional representations in locally Euclidean smooth manifolds.



# t-SNE

- t-SNE reduces dimensionality while preserving local similarity, has been build heuristically, and is commonly used to visualize representations.



Geoffrey Hinton

Emeritus Prof. Comp Sci, U.Toronto & Engineering Fellow, Google  
Dirección de correo verificada de cs.toronto.edu - [Página principal](#)

[machine learning](#) [psychology](#) [artificial intelligence](#) [cognitive science](#) [computer science](#)

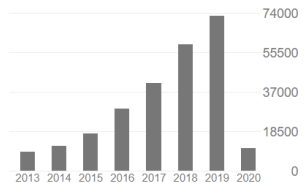


TÍTULO	CITADO POR	AÑO
<a href="#">Learning internal representations by error-propagation</a> DE Rumelhart, GE Hinton, RJ Williams Parallel Distributed Processing: Explorations in the Microstructure of ...	65178 *	1986
<a href="#">Imagenet classification with deep convolutional neural networks</a> A Krizhevsky, I Sutskever, GE Hinton Advances in neural information processing systems, 1097-1105	57920	2012
<a href="#">Learning representations by back-propagating errors</a> DE Rumelhart, GE Hinton, RJ Williams Nature 323, 533-536	44542 *	1986
<a href="#">Learning internal representations by error propagation</a> DE Rumelhart, GE Hinton, RJ Williams CALIFORNIA UNIV SAN DIEGO LA JOLLA INST FOR	26340	1985
<a href="#">Deep learning</a> Y LeCun, Y Bengio, G Hinton nature 521 (7553), 436-444	23304	2015
<a href="#">Learning representations by back-propagating errors</a> DE Rumelhart, GE Hinton, RJ Williams nature 323 (6088), 533-536	20220	1986
<a href="#">Dropout: a simple way to prevent neural networks from overfitting</a> N Srivastava, G Hinton, A Krizhevsky, I Sutskever, R Salakhutdinov The journal of machine learning research 15 (1), 1929-1958	18131	2014
<a href="#">A fast learning algorithm for deep belief nets</a> GE Hinton, S Osindero, YW Teh Neural computation 18 (7), 1527-1554	12473	2006
<a href="#">Visualizing data using t-SNE</a> L van der Maaten, G Hinton Journal of Machine Learning Research 9 (Nov), 2579-2605	12051	2008
<a href="#">Reducing the dimensionality of data with neural networks</a> GE Hinton, RR Salakhutdinov science 313 (5786), 504-507	11566	2006

Citado por

[VER TODO](#)

	Total	Desde 2015
Citas	353082	232746
Índice h	157	117
Índice i10	397	284



Coautores

	James L. McClelland Professor of Psychology, Stanfor...	>
	Terrence Sejnowski Francis Crick Professor, Saik Ins...	>
	George E. Dahl Google Brain	>
	Abdelrahman Mohamed Research scientist, Facebook AI ...	>
	Vinod Nair Research Scientist, DeepMind	>
	Radford Neal Emeritus Professor, Dept. of Stat...	>
	Sidney Fels Professor of Electrical and Comp...	>
	Chris Williams Professor of Machine Learning, ...	>
	David C. Plaut Professor of Psychology, Carneg...	>

# AAAI 2020

## Invited Speakers



**Geoffrey Hinton**  
U. of Toronto



**Yann LeCun**  
Facebook and NYU



**Yoshua Bengio**  
U. de Montreal



**Aude Billard**  
EPFL

Turing Award Winner Event!



**Henry Kautz**  
NSF and U. Rochester  
AAAI/IAAI Robert S.  
Engelmore Award  
Lecture



**Dawn Song**  
UC Berkeley  
AAAI/IAAI Joint  
Invited Talk



**Susan Athey**  
Stanford U.



**Stuart Russell**  
UC Berkeley



## History Panel: Advancing AI by Playing Games



*Moderator:*  
**Amy Greenwald**  
Brown University



**Michael Bowling**  
U. of Alberta



**Murray Campbell**  
IBM Research



**Gary Kasparov**  
Former world  
champion in chess



**Hiroaki Kitano**  
Sony Computer Science  
Laboratories, Inc.



**David Silver**  
DeepMind

*The Previous AAAI History Panel: Expert Systems (AAAI'17 in San Francisco)*

Educator Award Lecture  
(yesterday ☺)



**Benjamin  
Shapiro**



**Abigail  
Zimmerman--  
Niefield**

EAAI Invited Talk



**David Cox**  
IAAI Invited Talk





# Aprendizaje Automático

Based on material by Andrew Ng from Stanford University (Machine Learning course)

## Dimensionality Reduction



Máster en Bioinformática y Biología Computacional