Dimensionality Reduction

Dr Wajahat Hussain

Data Visualization

Eating in the UK (a 17D example)

Original example from Mark Richardson's class notes [Principal Component Analysis]

What if our data have way more than 3-dimensions? Like, **17** dimensions?! In the table is the average consumption of 17 types of food in grams per person per week for every country in the UK.

The table shows some interesting variations across different food types, but overall differences aren't so notable. Let's see if PCA can eliminate dimensions to emphasize how countries differ.

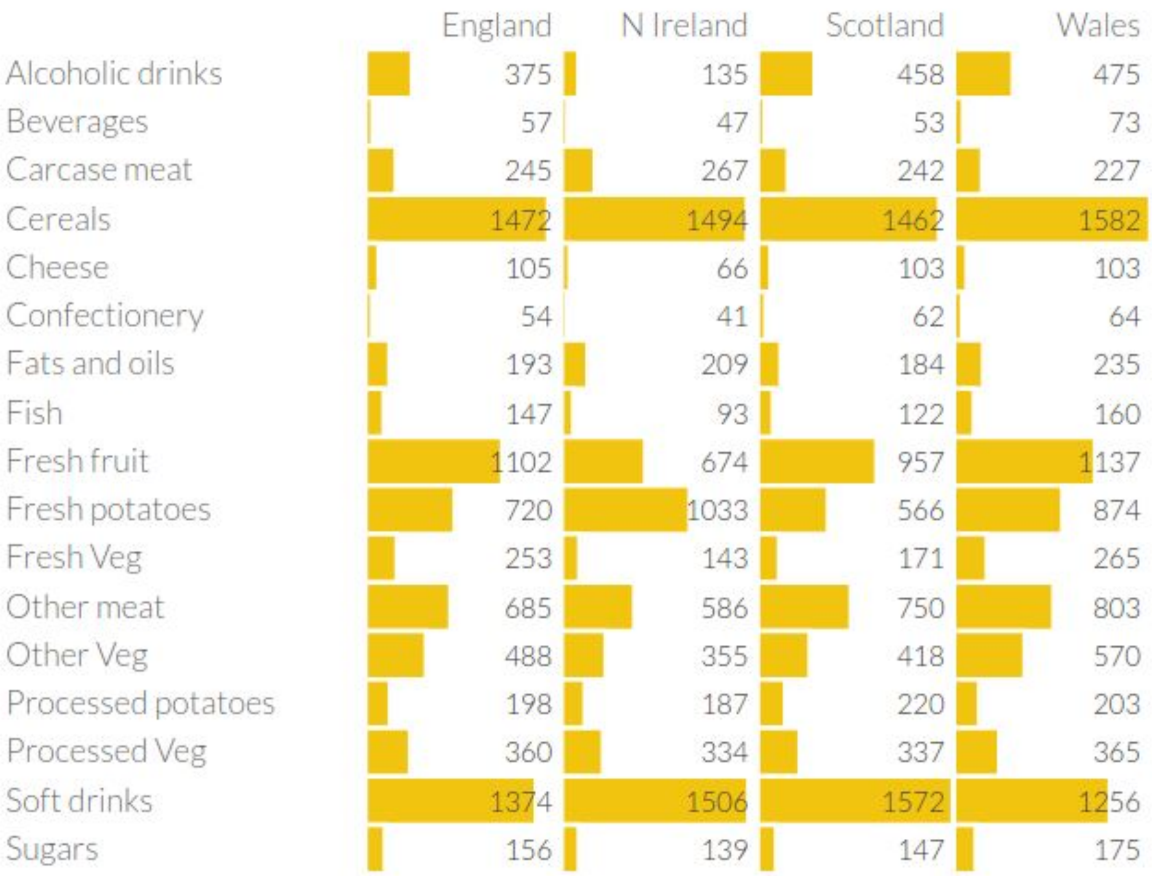
Which of England, Ireland, Scotland and Wales is behaving differently?

	England	N Ireland	Scotland	Wales
Alcoholic drinks	375	135	458	475
Beverages	57	47	53	73
Carcase meat	245	267	242	227
Cereals	1472	1494	1462	1582
Cheese	105	66	103	103
Confectionery	54	41	62	64
Fats and oils	193	209	184	235
Fish	147	93	122	160
Fresh fruit	1102	674	957	1137
Fresh potatoes	720	1033	566	874
Fresh Veg	253	143	171	265
Other meat	685	586	750	803
Other Veg	488	355	418	570
Processed potatoes	198	187	220	203
Processed Veg	360	334	337	365
Soft drinks	1374	1506	1572	1256
Sugars	156	139	147	175

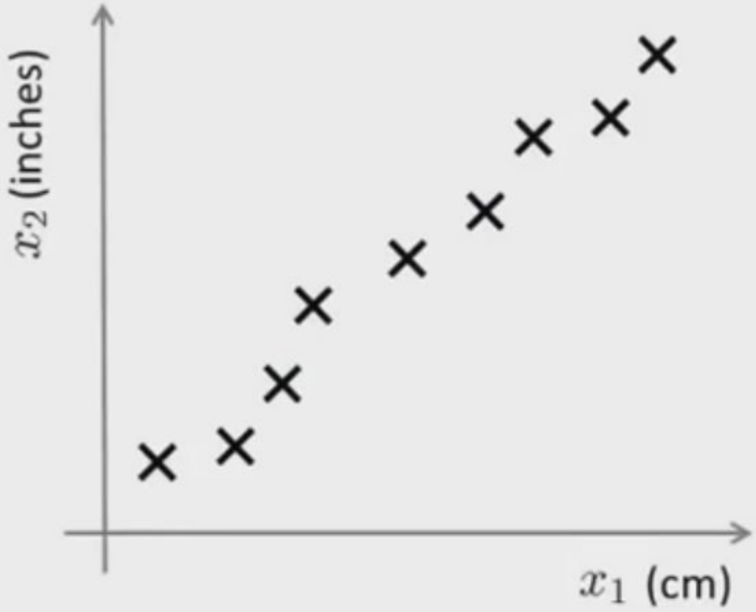
http://setosa.io/ev/principal-component-analysis/

Data Visualization

Which of England, Ireland, Scotland and Wales is behaving differently?

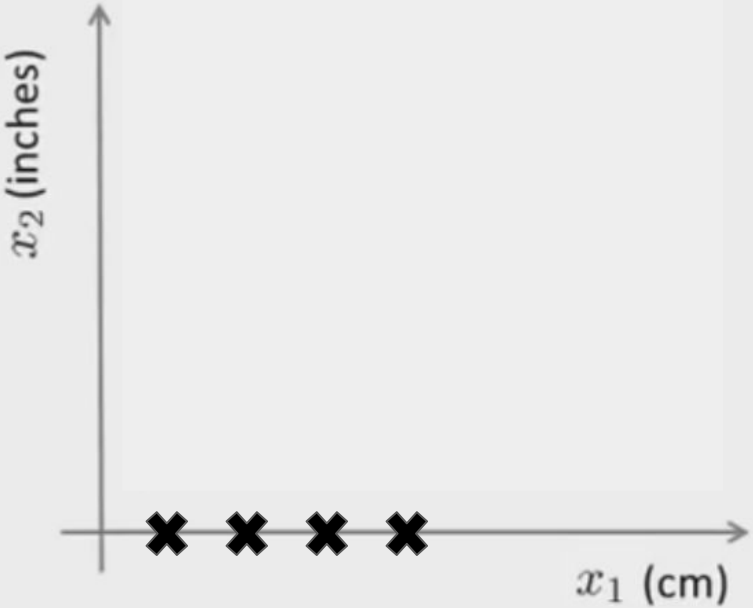


Data Compression



Reduce data from 2D to 1D

Data Compression

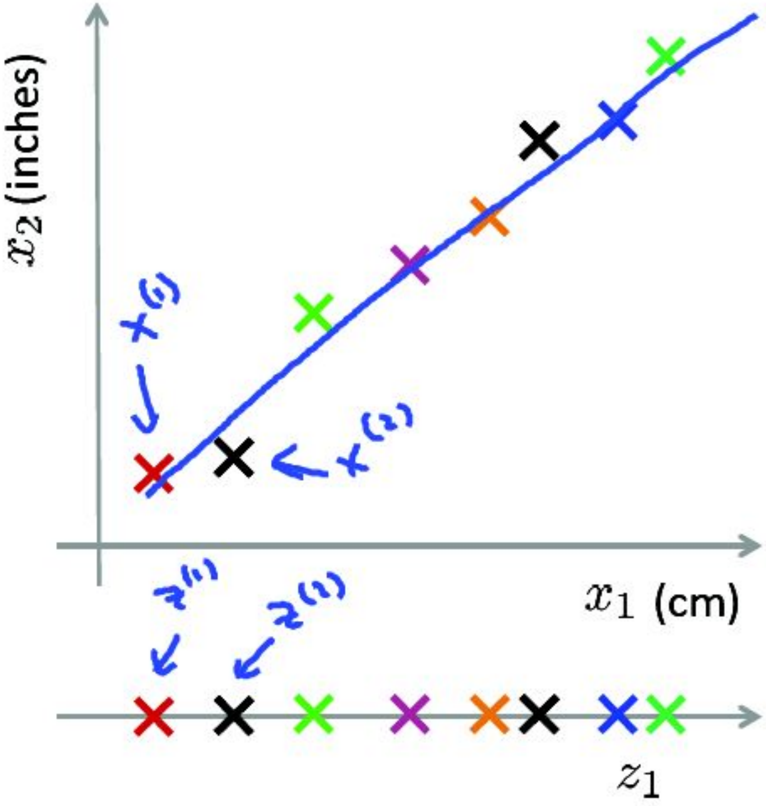


Reduce data from 2D to 1D

- **Is this data 2D? Or 1D?**

Andrew Ng

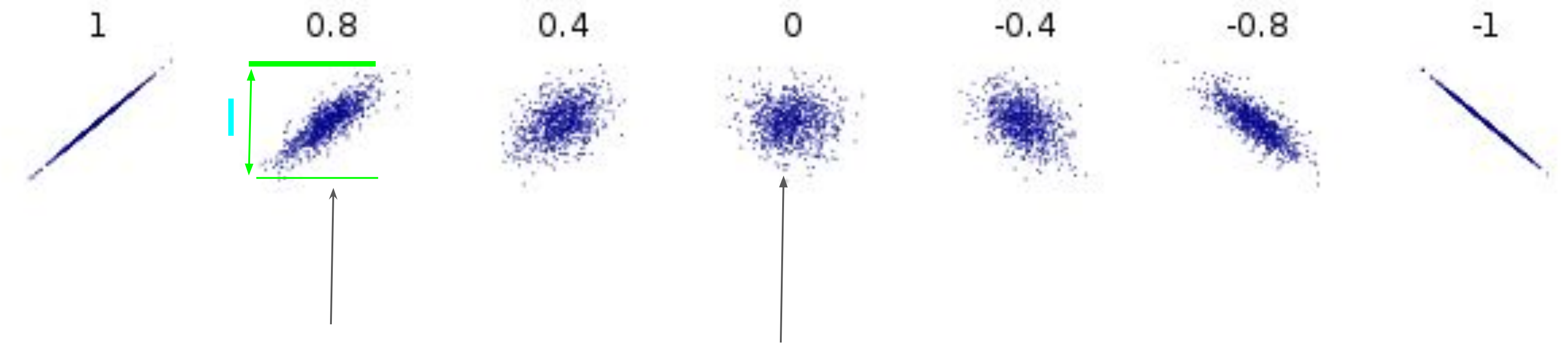
Data Compression



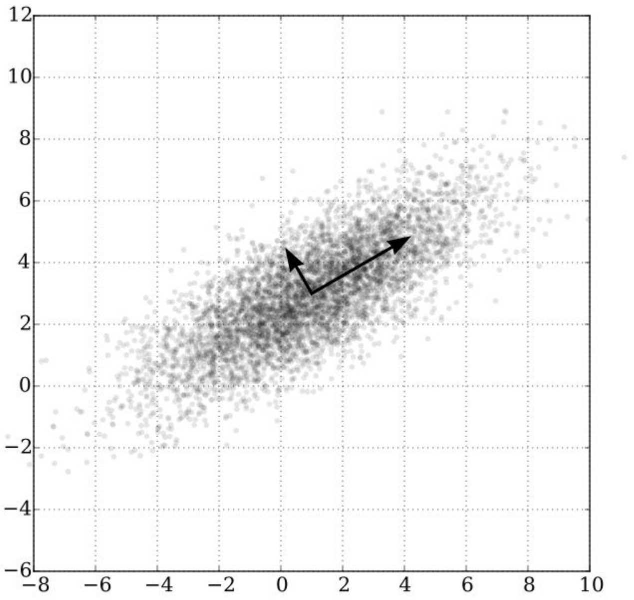
Reduce data from 2D to 1D

$$x^{(1)} \in \mathbb{R}^2 \rightarrow z^{(1)} \in \mathbb{R}$$
$$x^{(2)} \in \mathbb{R}^2 \rightarrow z^{(2)} \in \mathbb{R}$$
$$\vdots$$
$$x^{(m)} \in \mathbb{R}^2 \rightarrow z^{(m)} \in \mathbb{R}$$

Which data is easier to compress?



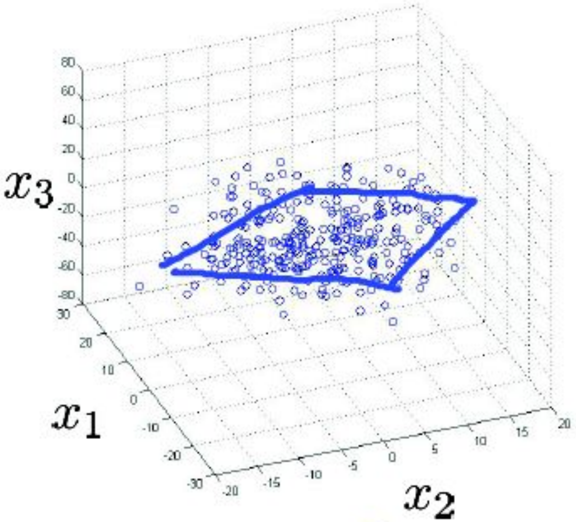
How to fit an ellipsoid in the given data?



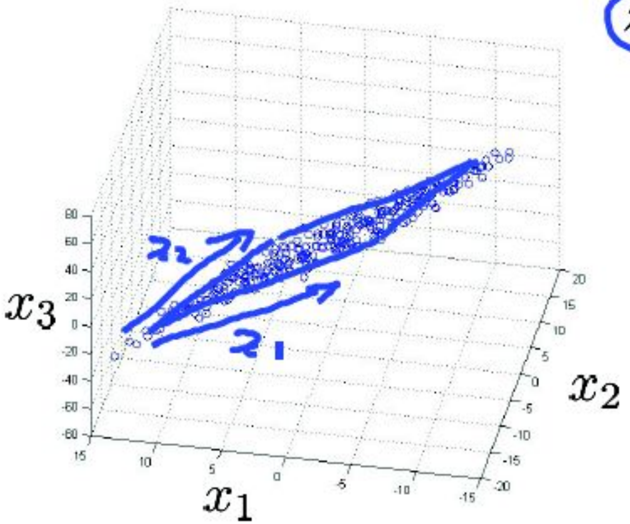
Data Compression

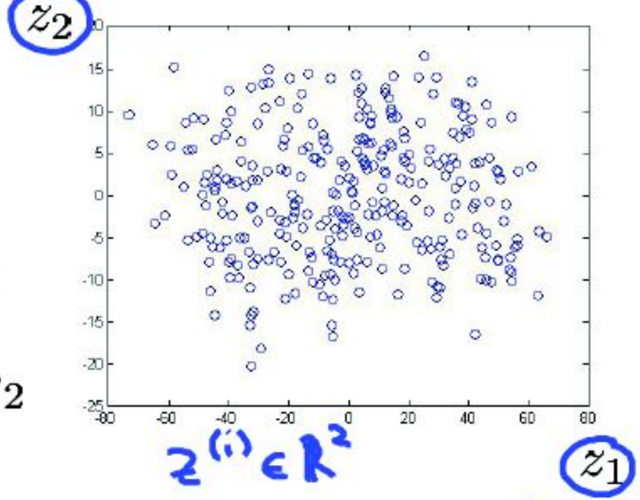
$1000D \rightarrow 100D$

Reduce data from 3D to 2D



$x^{(i)} \in \mathbb{R}^3$





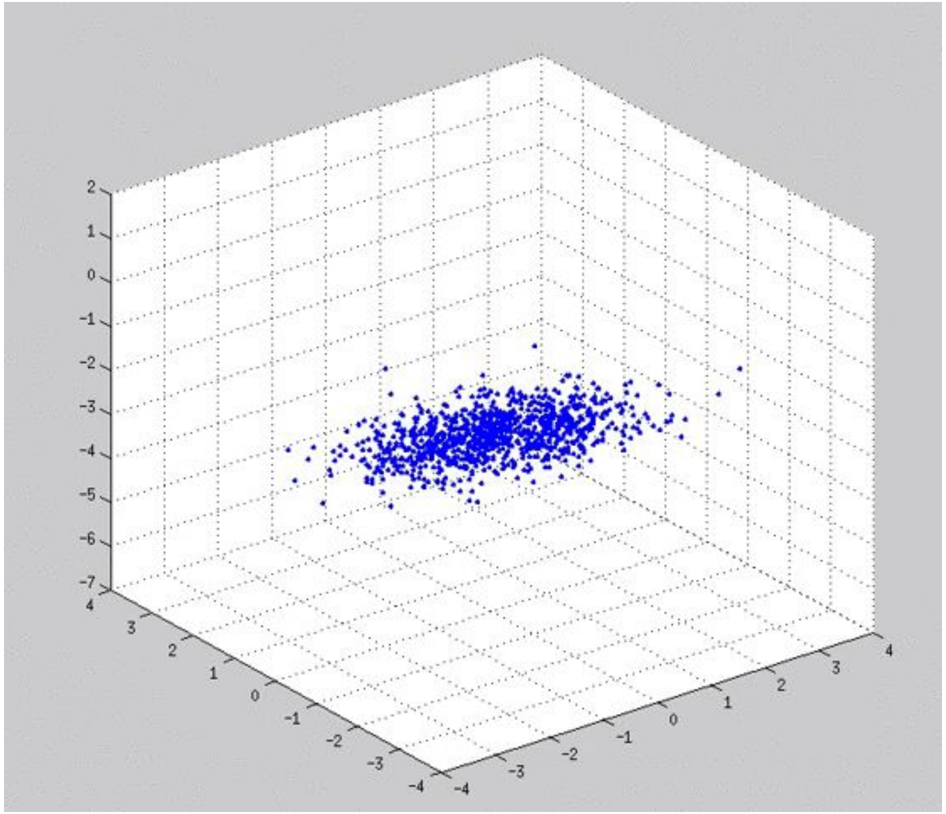
$z^{(i)} \in \mathbb{R}^2$

$$z = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} \qquad z^{(i)} = \begin{bmatrix} z_1^{(i)} \\ z_2^{(i)} \end{bmatrix}$$

Data Compression

10000 → 1000

Reduce data from 3D to 2D



Data Visualization

Eating in the UK (a 17D example)

Original example from Mark Richardson's class notes Principal Component Analysis

What if our data have way more than 3-dimensions? Like, **17** dimensions?! In the table is the average consumption of 17 types of food in grams per person per week for every country in the UK.

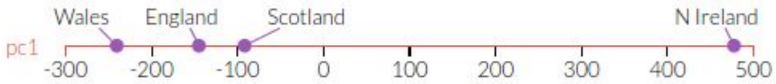
The table shows some interesting variations across different food types, but overall differences aren't so notable. Let's see if PCA can eliminate dimensions to emphasize how countries differ.

	England	N Ireland	Scotland	Wales
Alcoholic drinks	375	135	458	475
Beverages	57	47	53	73
Carcase meat	245	267	242	227
Cereals	1472	1494	1462	1582
Cheese	105	66	103	103
Confectionery	54	41	62	64
Fats and oils	193	209	184	235
Fish	147	93	122	160
Fresh fruit	1102	674	957	1137
Fresh potatoes	720	1033	566	874
Fresh Veg	253	143	171	265
Other meat	685	586	750	803
Other Veg	488	355	418	570
Processed potatoes	198	187	220	203
Processed Veg	360	334	337	365
Soft drinks	1374	1506	1572	1256
Sugars	156	139	147	175

Which of England, Ireland, Scotland and Wales is behaving differently?

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

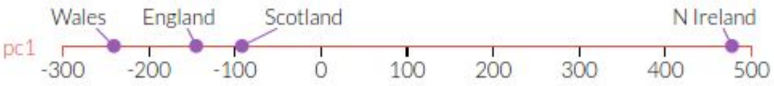
Data Visualization



	England	N Ireland	Scotland	Wales
Alcoholic drinks	375	135	458	475
Beverages	57	47	53	73
Carcase meat	245	267	242	227
Cereals	1472	1494	1462	1582
Cheese	105	66	103	103
Confectionery	54	41	62	64
Fats and oils	193	209	184	235
Fish	147	93	122	160
Fresh fruit	1102	674	957	1137
Fresh potatoes	720	1033	566	874
Fresh Veg	253	143	171	265
Other meat	685	586	750	803
Other Veg	488	355	418	570
Processed potatoes	198	187	220	203
Processed Veg	360	334	337	365
Soft drinks	1374	1506	1572	1256
Sugars	156	139	147	175

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

Data Visualization



Now, see the first principal component, we see Northern Ireland a major outlier. Once we go back and look at the data in the table, this makes sense: the Northern Irish eat way more grams of fresh potatoes and way fewer of fresh fruits, cheese, fish and alcoholic drinks.

	England	N Ireland	Scotland	Wales
Alcoholic drinks	375	135	458	475
Beverages	57	47	53	73
Carcase meat	245	267	242	227
Cereals	1472	1494	1462	1582
Cheese	105	66	103	103
Confectionery	54	41	62	64
Fats and oils	193	209	184	235
Fish	147	93	122	160
Fresh fruit	1102	674	957	1137
Fresh potatoes	720	1033	566	874
Fresh Veg	253	143	171	265
Other meat	685	586	750	803
Other Veg	488	355	418	570
Processed potatoes	198	187	220	203
Processed Veg	360	334	337	365
Soft drinks	1374	1506	1572	1256
Sugars	156	139	147	175

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

Data Visualization

$x \in \mathbb{R}^{50}$ $x^{(i)} \in \mathbb{R}^{50}$

Country	x_1 GDP (trillions of US$)	x_2 Per capita GDP (thousands of intl. $)	x_3 Human Develop-ment Index	x_4 Life expectancy	x_5 Poverty Index (Gini as percentage)	x_6 Mean household income (thousands of US$)	…
Canada	1.577	39.17	0.908	80.7	32.6	67.293	…
China	5.878	7.54	0.687	73	46.9	10.22	…
India	1.632	3.41	0.547	64.7	36.8	0.735	…
Russia	1.48	19.84	0.755	65.5	39.9	0.72	…
Singapore	0.223	56.69	0.866	80	42.5	67.1	…
USA	14.527	46.86	0.91	78.3	40.8	84.3	…
…	…	…	…	…	…	…	

[resources from en.wikipedia.org]

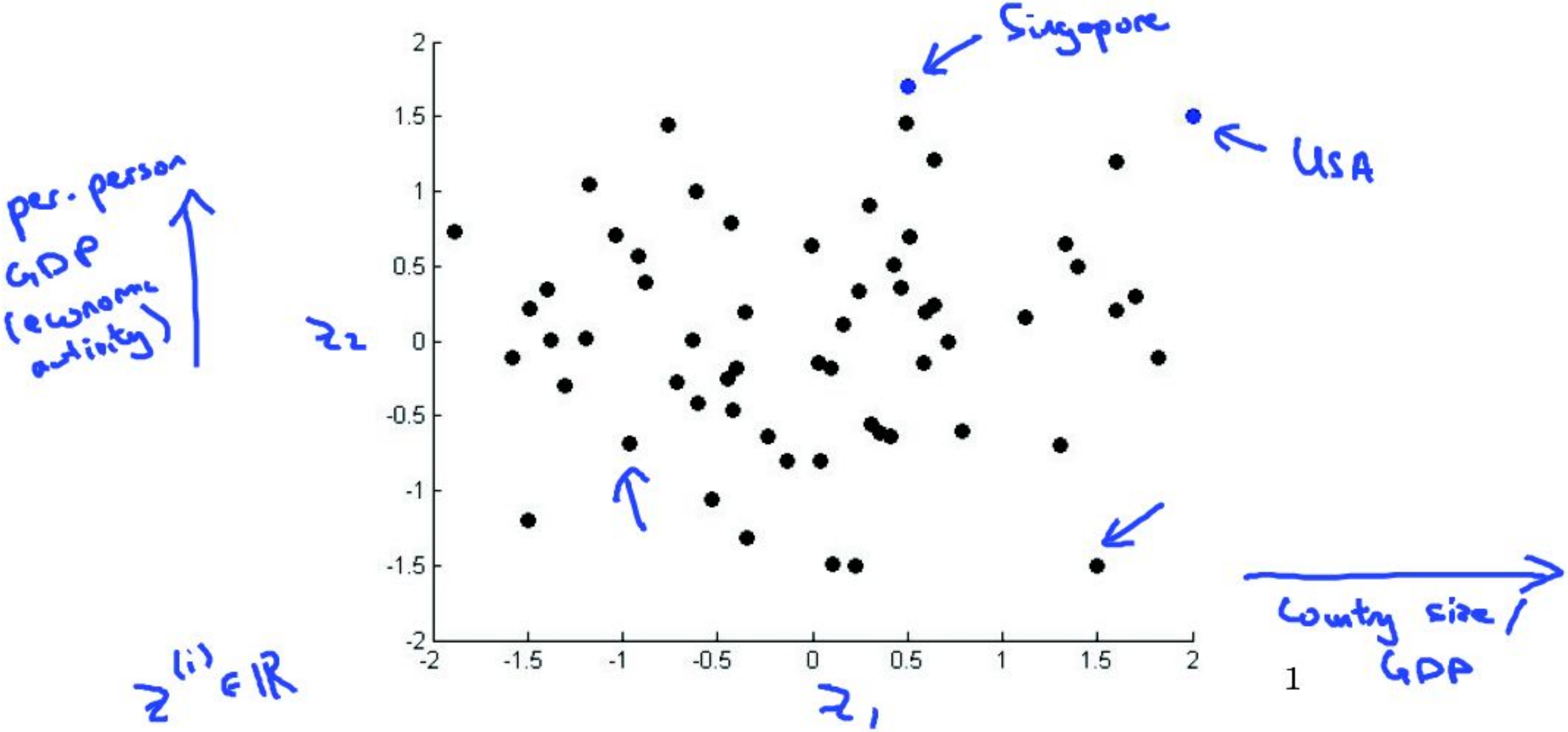
Data Visualization

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

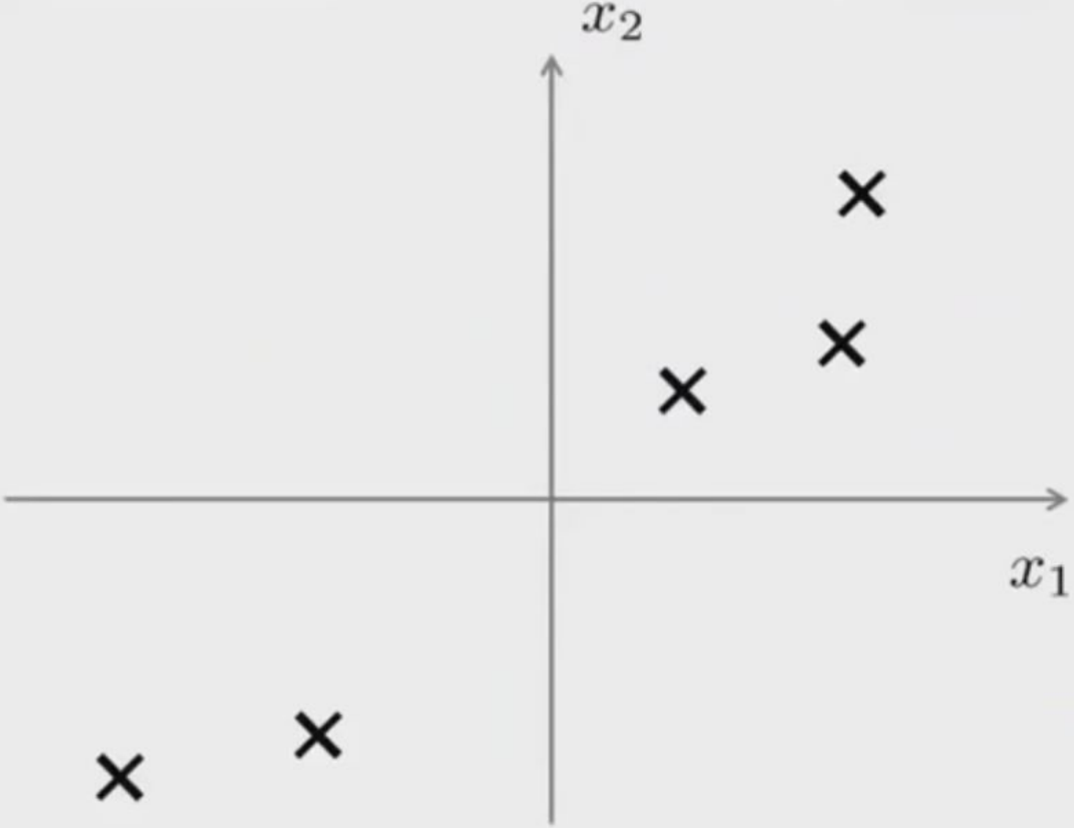
Country	z_1	z_2
Canada	1.6	1.2
China	1.7	0.3
India	1.6	0.2
Russia	1.4	0.5
Singapore	0.5	1.7
USA	2	1.5
…	…	…

Reduce data from 50D to 2D

Data Visualization



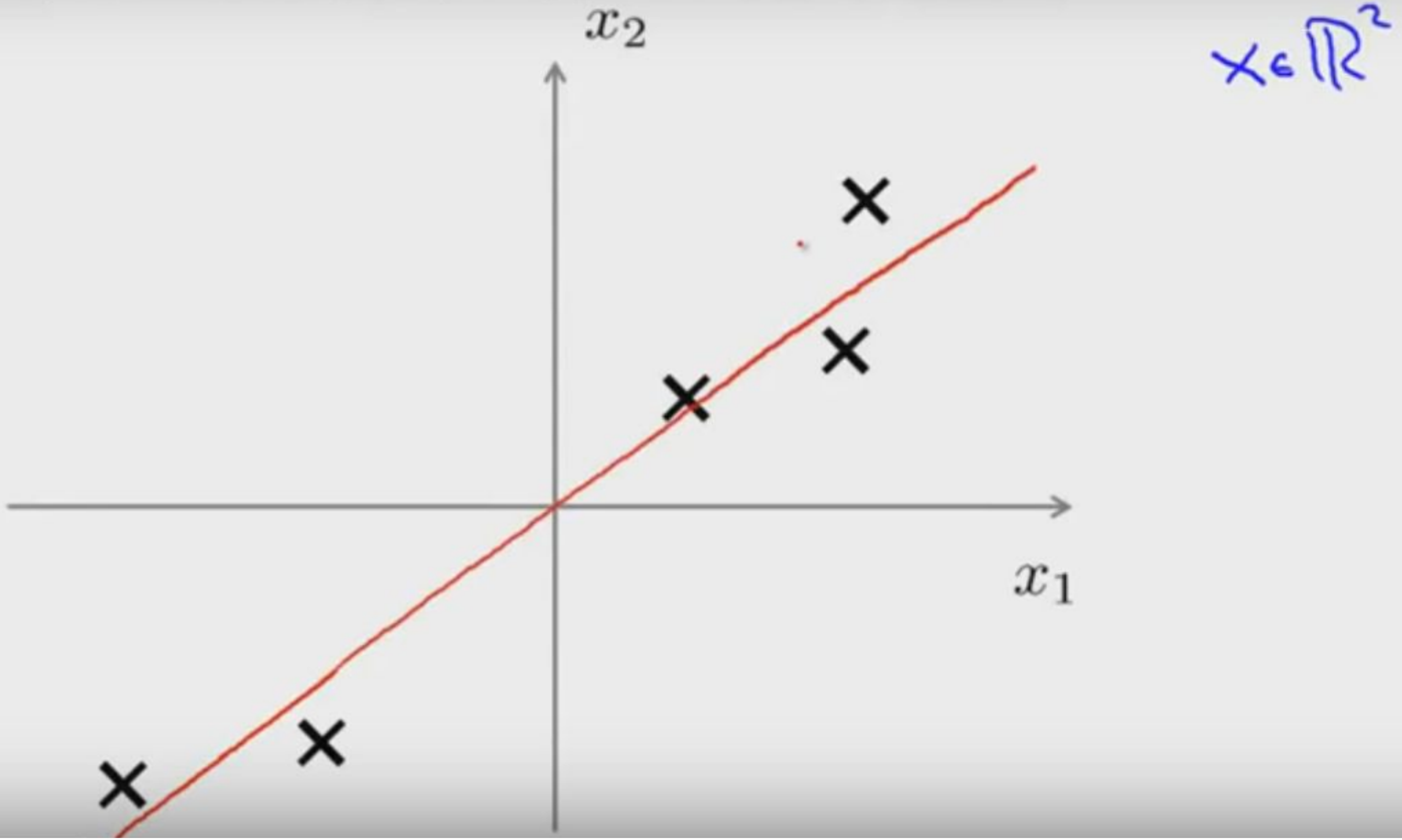
Principal Component Analysis (PCA) problem formulation



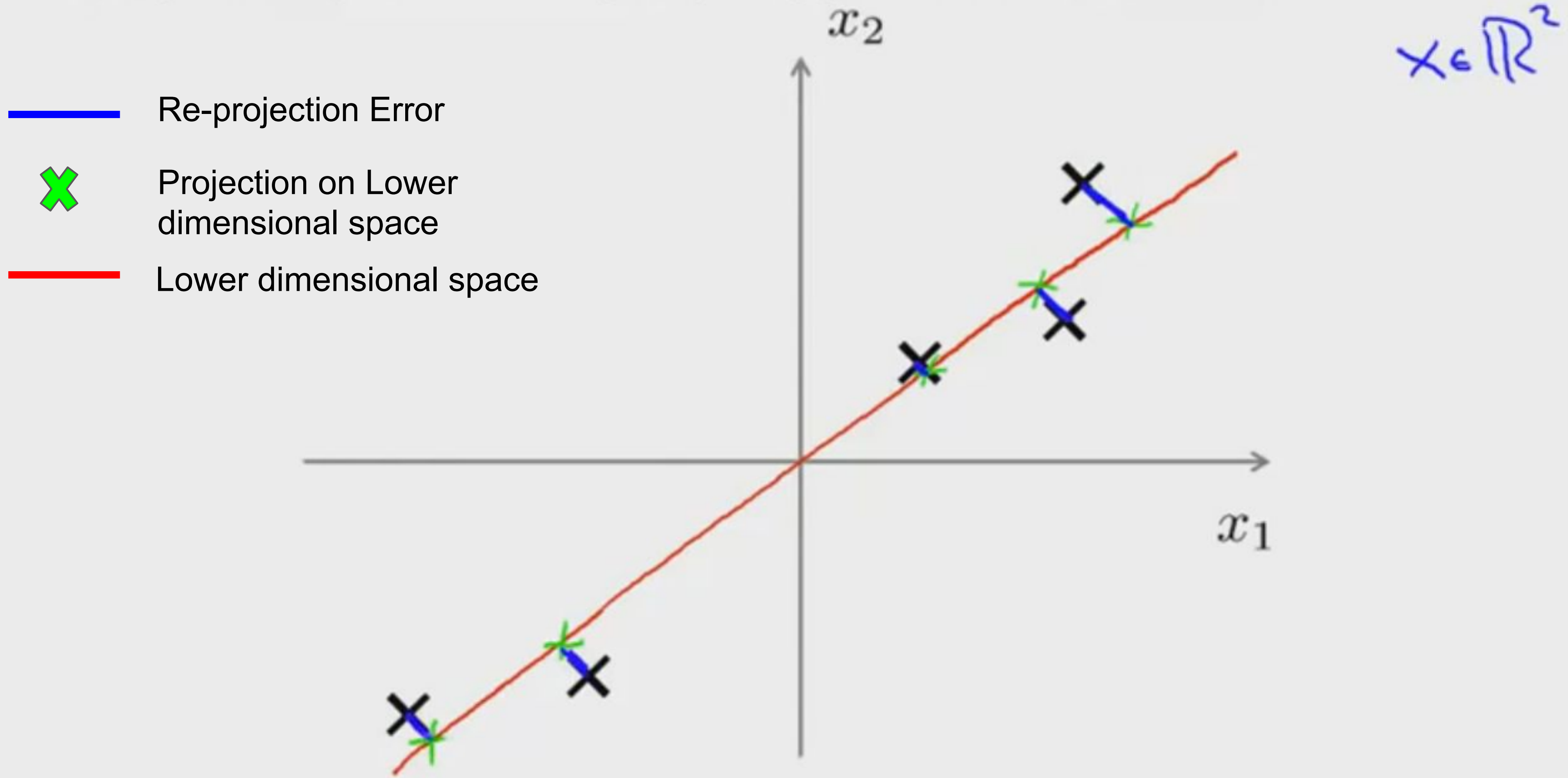
$x \in \mathbb{R}^2$

Andrew Ng

Principal Component Analysis (PCA) problem formulation



Principal Component Analysis (PCA) problem formulation



Andrew Ng

Principal Component Analysis (PCA) problem formulation

$x \in \mathbb{R}^2$

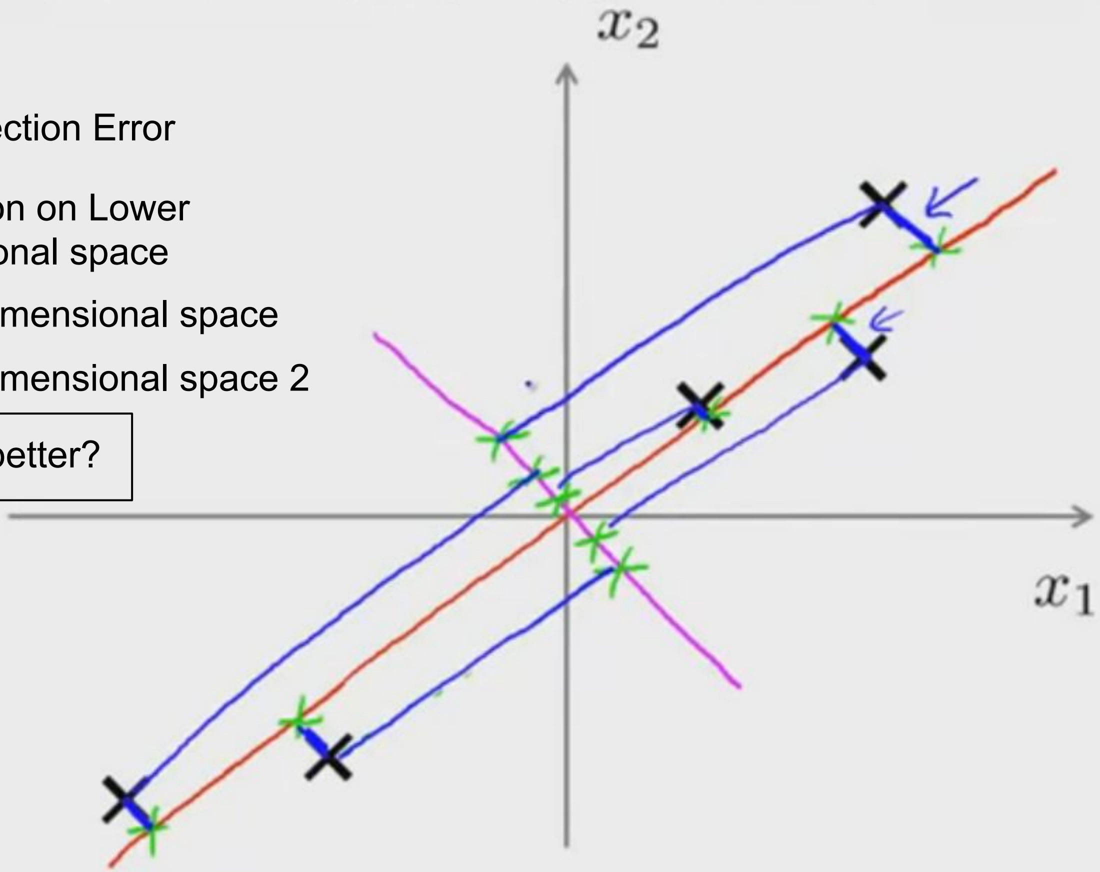
Re-projection Error

Projection on Lower dimensional space

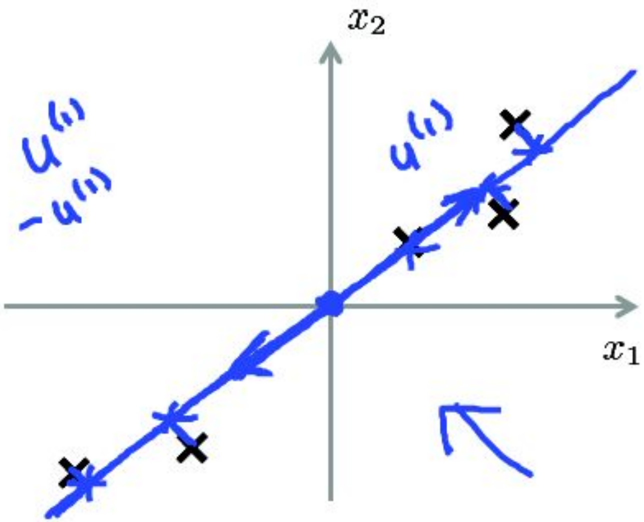
Lower dimensional space

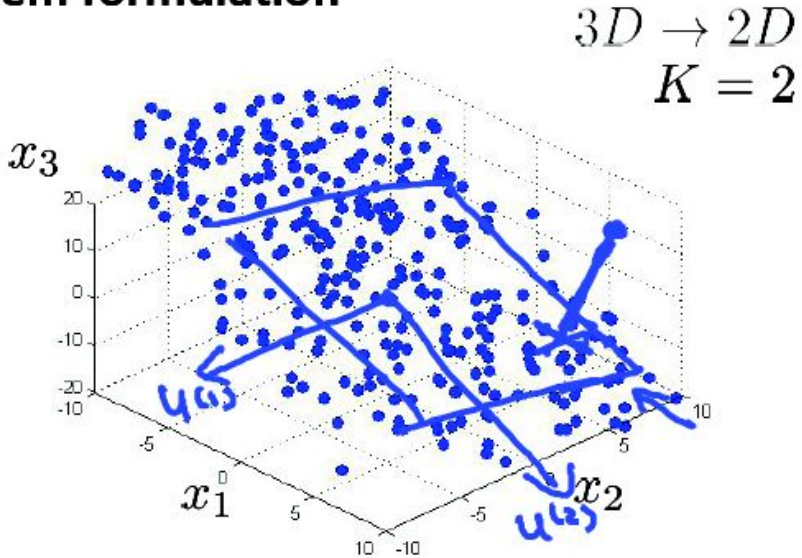
Lower dimensional space 2

Magenta or Red is better?



Principal Component Analysis (PCA) problem formulation



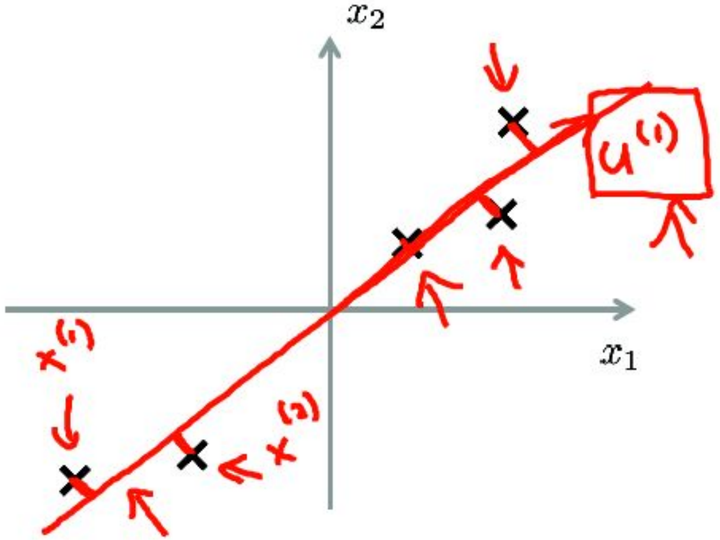


Reduce from 2-dimension to 1-dimension: Find a direction (a vector $u^{(1)} \in \mathbb{R}^n$) onto which to project the data so as to minimize the projection error.

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

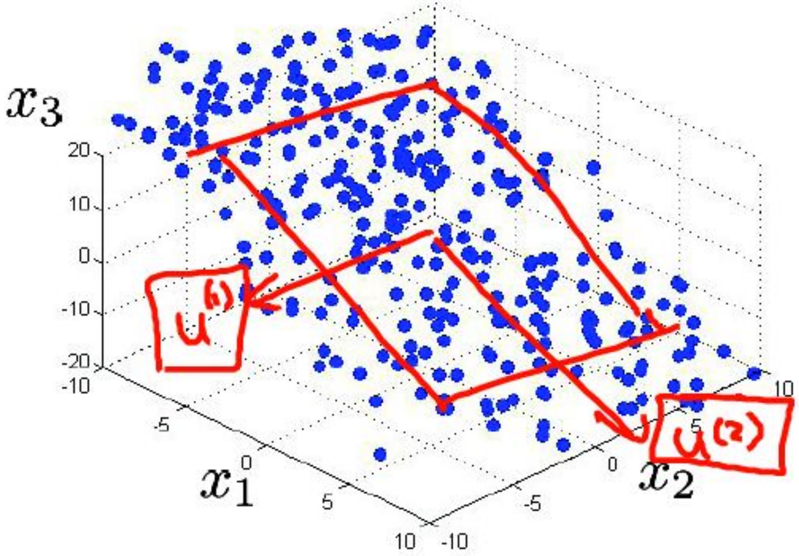
Andrew Ng

Principal Component Analysis (PCA) algorithm



Reduce data from 2D to 1D

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



Reduce data from 3D to 2D

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

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

Andrew Ng

Data preprocessing

Training set: $x^{(1)}, x^{(2)}, \ldots, x^{(m)}$ ←

Preprocessing (feature scaling/mean normalization):

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

Replace each $x_j^{(i)}$ with $x_j - \mu_j$.

If different features on different scales (e.g., $x_1 =$ size of house, $x_2 =$ number of bedrooms), scale features to have comparable range of values.

$$x_j^{(i)} \leftarrow \frac{x_j^{(i)} - \mu_j}{s_j}$$

Andrew Ng

Principal Component Analysis (PCA) algorithm

Reduce data from n-dimensions to k-dimensions

Compute "covariance matrix":

$$\Sigma = \frac{1}{m}\sum_{i=1}^{m} \underbrace{\underbrace{(x^{(i)})}_{n\times 1}\underbrace{(x^{(i)})^T}_{1\times n}}_{n\times n}$$

Sigma

Compute "eigenvectors" of matrix Σ :

→ Singular value decomposition

```
[U,S,V] = svd(Sigma);
```

eig(Sigma)

n×n matrix.

$$U = \begin{bmatrix} | & | & | & & | \\ u^{(1)} & u^{(2)} & u^{(3)} & \cdots & u^{(n)} \\ | & | & | & & | \end{bmatrix}$$

$\underbrace{u^{(1)}\ u^{(2)}\ u^{(3)}}_{k}$

$U \in \mathbb{R}^{n\times n}$

$u^{(1)}, \ldots, u^{(k)}$

Singular Value Decomposition (SVD)

$$\mathbf{M} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^{T}$$

- $\mathbf{U}$ is an $m \times m$ unitary matrix
- $\boldsymbol{\Sigma}$ is a diagonal $m \times n$ matrix with nonnegative real numbers on the diagonal,
- $\mathbf{V}$ is an $n \times n$ unitary matrix
- $\mathbf{V}^{T}$ is the transpose of $\mathbf{V}$

- What is a unitary matrix?
- $U^{T} U = I$, $V^{T} V = I$,

Example: SVD

$$\mathbf{M} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^{\mathbf{T}}$$

$$\mathbf{M} = \begin{bmatrix} 1 & 0 & 0 & 0 & 2 \\ 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 \end{bmatrix}$$

$$\mathbf{U} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

$$\boldsymbol{\Sigma} = \begin{bmatrix} 2 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & \sqrt{5} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\mathbf{V}^* = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ \sqrt{0.2} & 0 & 0 & 0 & \sqrt{0.8} \\ 0 & 0 & 0 & 1 & 0 \\ -\sqrt{0.8} & 0 & 0 & 0 & \sqrt{0.2} \end{bmatrix}$$

$$\mathbf{U}\mathbf{U}^{\mathbf{T}} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1 \\ 1 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \mathbf{I}_4$$

$$\mathbf{V}\mathbf{V}^{\mathbf{T}} = \begin{bmatrix} 0 & 0 & \sqrt{0.2} & 0 & -\sqrt{0.8} \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & \sqrt{0.8} & 0 & \sqrt{0.2} \end{bmatrix} \cdot \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ \sqrt{0.2} & 0 & 0 & 0 & \sqrt{0.8} \\ 0 & 0 & 0 & 1 & 0 \\ -\sqrt{0.8} & 0 & 0 & 0 & \sqrt{0.2} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} = \mathbf{I}_5$$

Example: SVD

$$\mathbf{M} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^{\mathsf{T}}$$

```
A = [2 0 2; 0 1 0; 0 0 0]
```

```
A = 3x3

     2     0     2
     0     1     0
     0     0     0
```

```
[U,S,V] = svd(A)
```

```
U = 3x3

     1     0     0
     0     1     0
     0     0     1

S = 3x3

    2.8284         0         0
         0    1.0000         0
         0         0         0

V = 3x3

    0.7071         0   -0.7071
         0    1.0000         0
    0.7071         0    0.7071
```

Principal Component Analysis (PCA) algorithm

From `[U,S,V] = svd(Sigma)`, we get:

$$\rightarrow U = \underbrace{\left[\begin{array}{cccc} | & | & & | \\ u^{(1)} & u^{(2)} & \ldots & u^{(n)} \\ | & | & & | \end{array}\right]}_{k} \in \mathbb{R}^{n \times n}$$

$$x \in \mathbb{R}^n \longrightarrow z \in \mathbb{R}^k$$

$$z^{(i)} = \underbrace{\left[\begin{array}{cccc} | & | & & | \\ u^{(1)} & u^{(2)} & \ldots & u^{(k)} \\ | & | & & | \end{array}\right]}_{\substack{n \times k \\ U_{reduce}}}^{T} x^{(i)} = \underbrace{\underbrace{\left[\begin{array}{c} — (u^{(1)})^T — \\ \vdots \\ — (u^{(k)})^T — \end{array}\right]}_{k \times n} \underbrace{x^{(i)}}_{n \times 1}}_{k \times 1}$$

$$z \in \mathbb{R}^k$$

Andrew Ng

Principal Component Analysis (PCA) algorithm summary

- After mean normalization (ensure every feature has zero mean) and optionally feature scaling:

$$\text{Sigma} = \frac{1}{m}\sum_{i=1}^{m}(x^{(i)})(x^{(i)})^T$$

$$\underline{X} = \begin{bmatrix} — & x^{(1)T} & — \\ & \vdots & \\ — & x^{(m)T} & — \end{bmatrix}$$

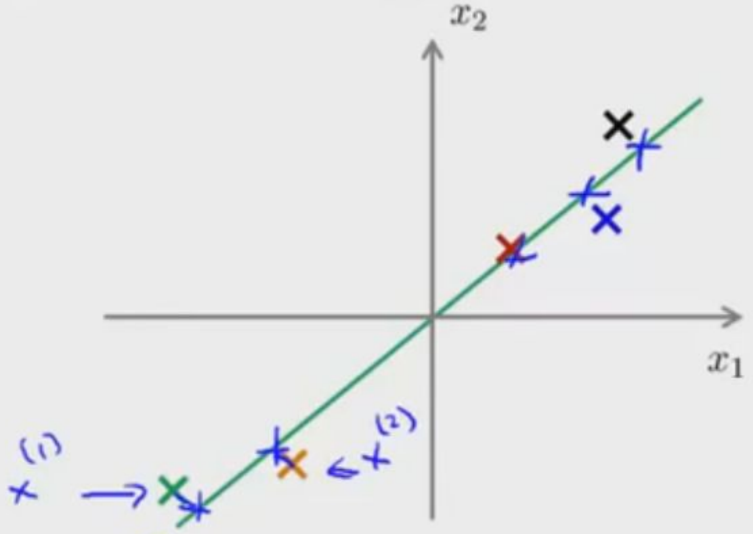
$\boxed{\text{Sigma} = (1/m) * X' * X;}$

```
[U,S,V] = svd(Sigma);
Ureduce = U(:,1:k);
z = Ureduce'*x;
```

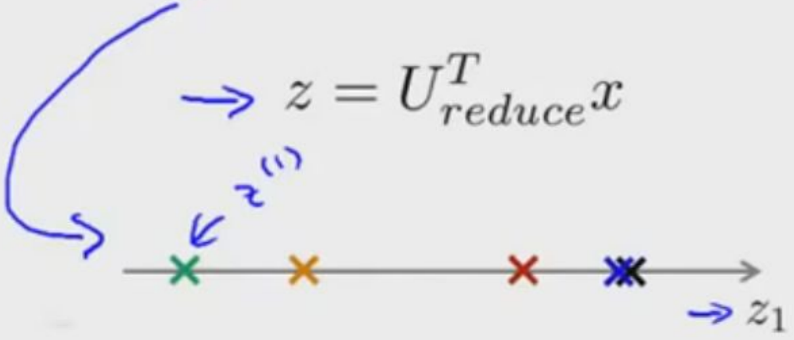
$x \in \mathbb{R}^n$ $\cancel{x_0 = 1}$

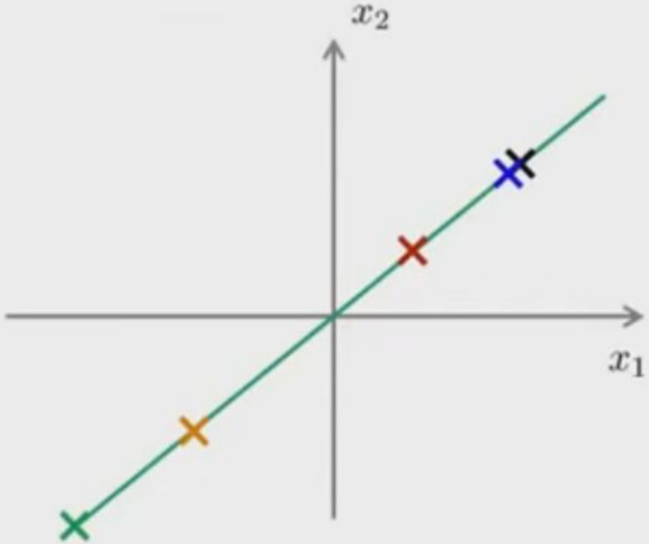
Andrew Ng

Reconstruction from compressed representation



$$\rightarrow z = U_{reduce}^T x$$





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

$$\approx x$$

$$\underbrace{x_{approx}}_{\mathbb{R}^n} = \underbrace{\underbrace{U_{reduce}}_{n \times k} \cdot \underbrace{z}_{k \times 1}}_{n \times 1}$$

Choosing k (number of principal components)

Average squared projection error: $\frac{1}{m}\sum_{i=1}^{m}\|x^{(i)} - x_{approx}^{(i)}\|^2$

Total variation in the data: $\frac{1}{m}\sum_{i=1}^{m}\|x^{(i)}\|^2$

Typically, choose k to be smallest value so that

$$\frac{\frac{1}{m}\sum_{i=1}^{m}\|x^{(i)} - x_{approx}^{(i)}\|^2}{\frac{1}{m}\sum_{i=1}^{m}\|x^{(i)}\|^2} \leq 0.01 \qquad (1\%)$$

"99% of variance is retained"

http://mathworld.wolfram.com/VectorNorm.html

Andrew Ng

$$|\mathbf{x}|_p \equiv \left(\sum_i |x_i|^p\right)^{1/p}.$$

$$|\mathbf{x}|_2 = |\mathbf{x}| = \sqrt{x_1^2 + x_2^2 + \ldots + x_n^2}.$$

This and other types of vector norms are summarized in the following table, together with the value of the norm for the example vector $\mathbf{v} = (1, 2, 3)$.

name	symbol	value	approx.
L^1-norm	$\lvert\mathbf{x}\rvert_1$	6	6.000
L^2-norm	$\lvert\mathbf{x}\rvert_2$	$\sqrt{14}$	3.742
L^3-norm	$\lvert\mathbf{x}\rvert_3$	$6^{2/3}$	3.302
L^4-norm	$\lvert\mathbf{x}\rvert_4$	$2^{1/4}\sqrt{7}$	3.146
L^∞-norm	$\lvert\mathbf{x}\rvert_\infty$	3	3.000

http://mathworld.wolfram.com/VectorNorm.html

Choosing k (number of principal components)

Algorithm:

Try PCA with $\cancel{k=1}$ $\cancel{k=2}$ $\cancel{k=3}$ $k=4$ $\vdots$

Compute $U_{reduce}, z^{(1)}, z^{(2)}, \ldots, z^{(m)}, x^{(1)}_{approx}, \ldots, x^{(m)}_{approx}$

Check if

$$\frac{\frac{1}{m}\sum_{i=1}^{m}\|x^{(i)} - x^{(i)}_{approx}\|^2}{\frac{1}{m}\sum_{i=1}^{m}\|x^{(i)}\|^2} \leq 0.01?$$

$k = 17$

```
→ [U,S,V] = svd(Sigma)
```

$$S = \begin{bmatrix} s_{11} & & & 0 \\ & s_{22} & & \\ & & s_{33} & \\ & & & \ddots \\ 0 & & & & s_{nn} \end{bmatrix}$$

Choosing k (number of principal components)

```
[U,S,V] = svd(Sigma)
```

Pick smallest value of k for which

$$\boxed{\frac{\sum_{i=1}^{k} S_{ii}}{\sum_{i=1}^{m} S_{ii}}} \geq 0.99$$

k=100

(99% of variance retained)

Application of PCA

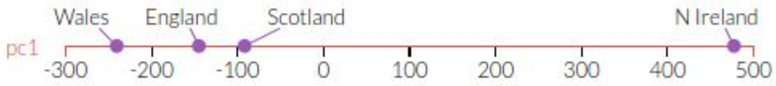
- Compression
  - Reduce memory/disk needed to store data
  - Speed up learning algorithm ←

Choose k by % of variance retain

- Visualization

$k=2$ or $k=3$

Data Visualization



	England	N Ireland	Scotland	Wales
Alcoholic drinks	375	135	458	475
Beverages	57	47	53	73
Carcase meat	245	267	242	227
Cereals	1472	1494	1462	1582
Cheese	105	66	103	103
Confectionery	54	41	62	64
Fats and oils	193	209	184	235
Fish	147	93	122	160
Fresh fruit	1102	674	957	1137
Fresh potatoes	720	1033	566	874
Fresh Veg	253	143	171	265
Other meat	685	586	750	803
Other Veg	488	355	418	570
Processed potatoes	198	187	220	203
Processed Veg	360	334	337	365
Soft drinks	1374	1506	1572	1256
Sugars	156	139	147	175

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