Machine Learning

Unsupervised Learning

Topics of previous lectures

- ✓ Ingredients of Machine Learning
- ✓ Classification Basics, Basic Linear Classifier
- ✓ K-Nearest Neighbours and Naive Bayes Classifier
- ✓ Linear and Quadratic Discriminant Analysis
- ✓ Support Vector Machines (SVM)
- ✓ Decision Trees
- ✓ Ensemble Methods (Bagging, Weighted Voting, Stacking)
- ✓ Regression Methods
- ✓ Evaluation and Scoring of Classifiers
- ✓ Ensemble Methods (Boosting)
- ✓ Clustering (Hierarchical, K-Means, K-Medoids, DBSCAN)

Topics of today's lecture

- Dimensionality Reduction
- Principal Component Analysis (PCA)
- t-Distributed Stochastic Neighbor Embedding (t-SNE)

Motivation for Dimensionality Reduction

- Working directly with high-dimensional data, such as images, comes with some difficulties

Motivation for Dimensionality Reduction

- Working directly with high-dimensional data, such as images, comes with some difficulties
- High-dimensional data is often overcomplete, i.e., many dimensions are redundant and can be explained by a combination of other dimensions.

Motivation for Dimensionality Reduction

- Working directly with high-dimensional data, such as images, comes with some difficulties
- High-dimensional data is often overcomplete, i.e., many dimensions are redundant and can be explained by a combination of other dimensions.
- Dimensions in high-dimensional data are often correlated so that the data possesses an intrinsic lower-dimensional structure.

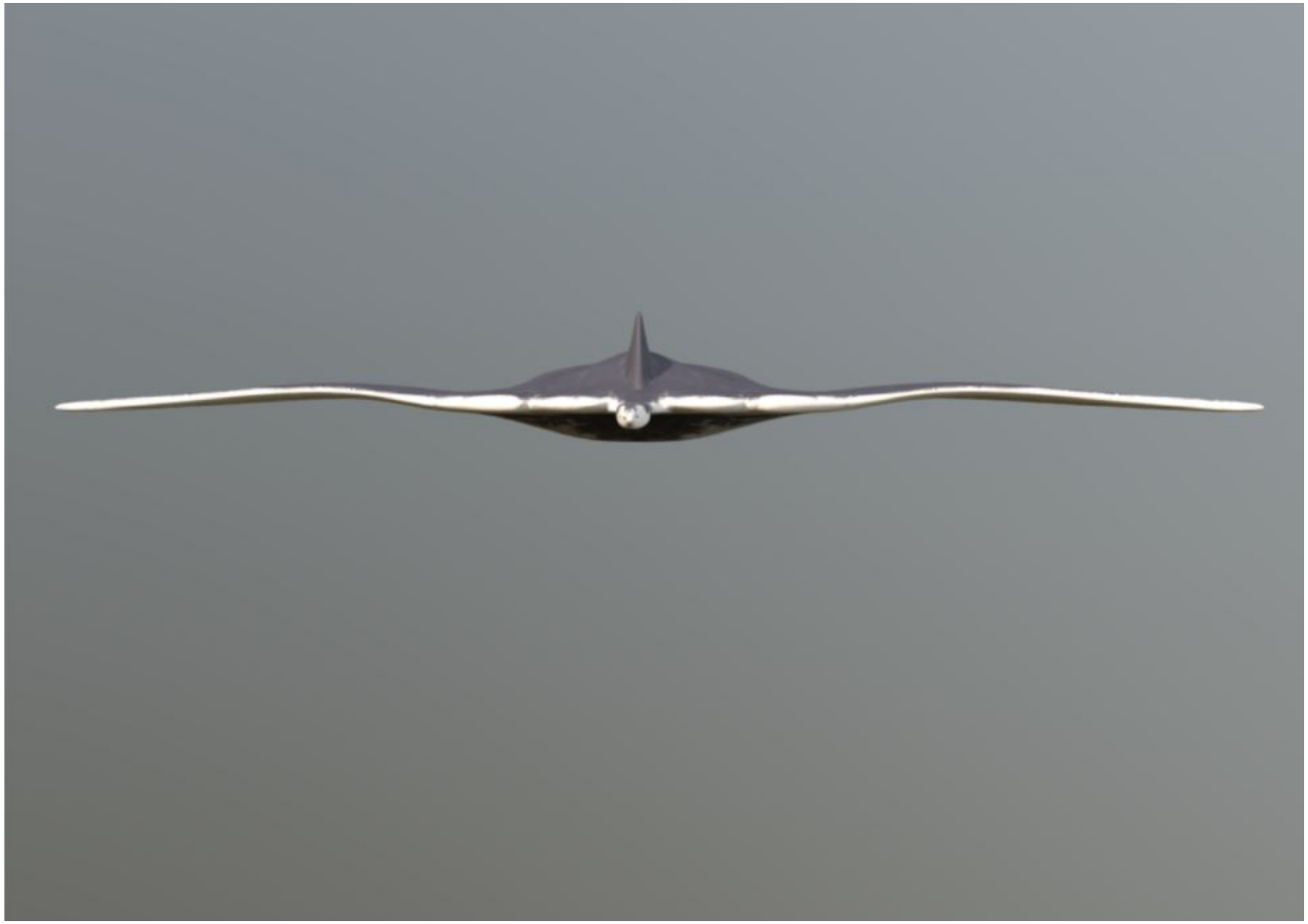
Motivation for Dimensionality Reduction

- Working directly with high-dimensional data, such as images, comes with some difficulties
- High-dimensional data is often overcomplete, i.e., many dimensions are redundant and can be explained by a combination of other dimensions.
- Dimensions in high-dimensional data are often correlated so that the data possesses an intrinsic lower-dimensional structure.
- Dimensionality reduction exploits structure and correlation and allows us to work with a more compact representation of the data, ideally without losing much information.

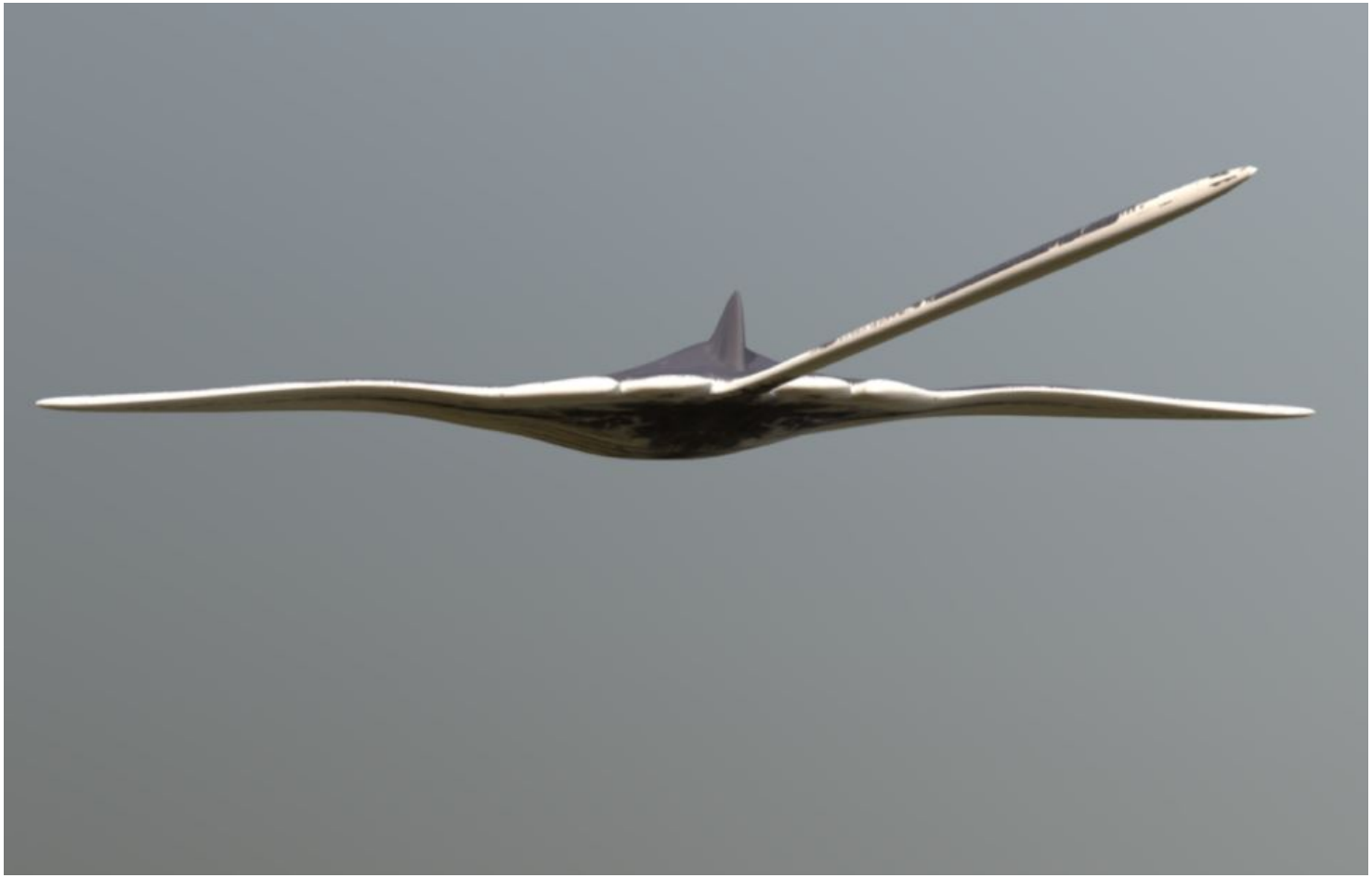
Motivation for Dimensionality Reduction

- Working directly with high-dimensional data, such as images, comes with some difficulties
- High-dimensional data is often overcomplete, i.e., many dimensions are redundant and can be explained by a combination of other dimensions.
- Dimensions in high-dimensional data are often correlated so that the data possesses an intrinsic lower-dimensional structure.
- Dimensionality reduction exploits structure and correlation and allows us to work with a more compact representation of the data, ideally without losing much information.
- It can also be useful to detect potential patterns in high dimensial data, by visualizing the first 2-3 projections.

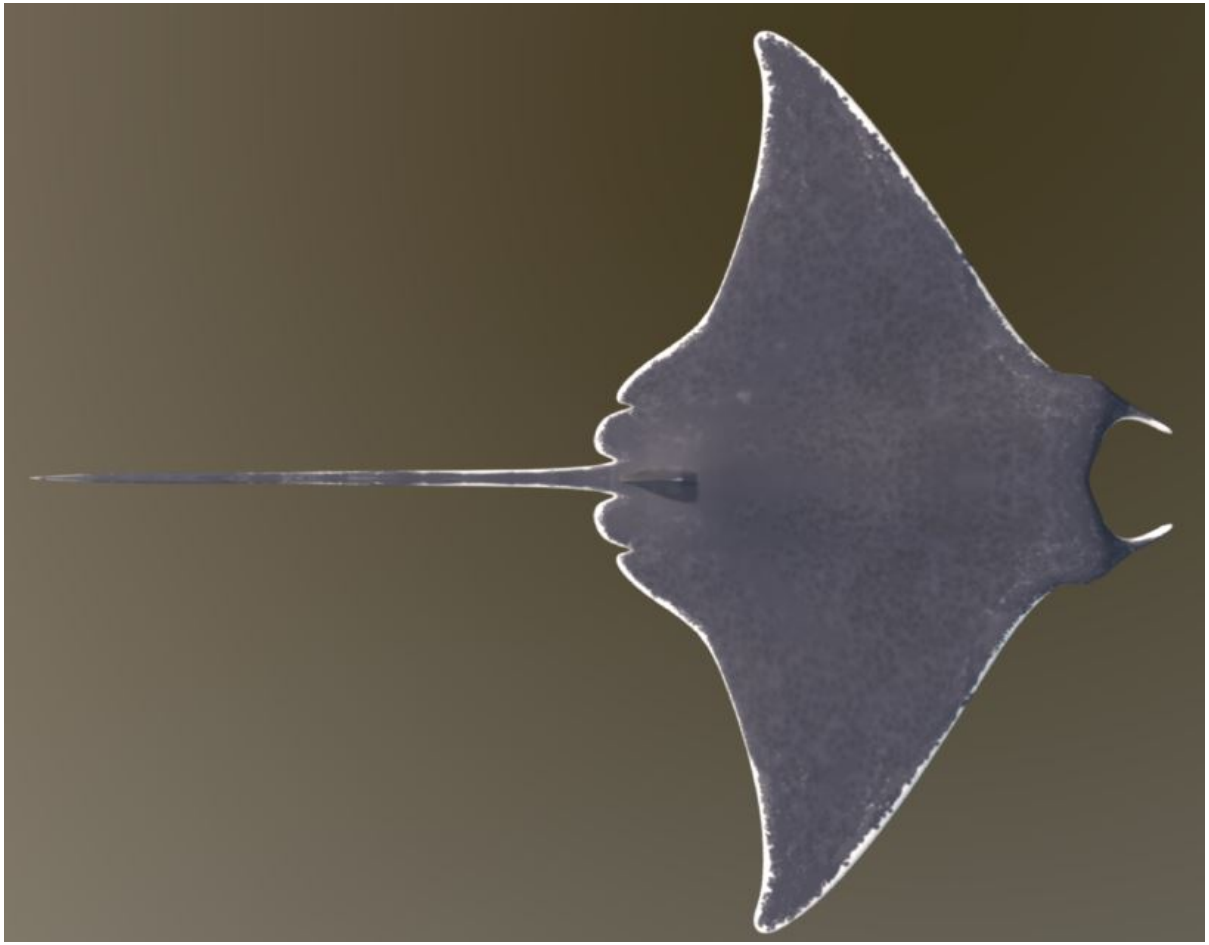
Motivation for Dimensionality Reduction



Motivation for Dimensionality Reduction

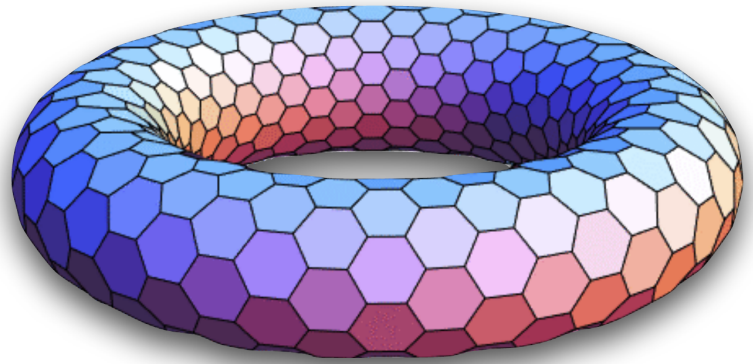


Motivation for Dimensionality Reduction



Principal Component Analysis

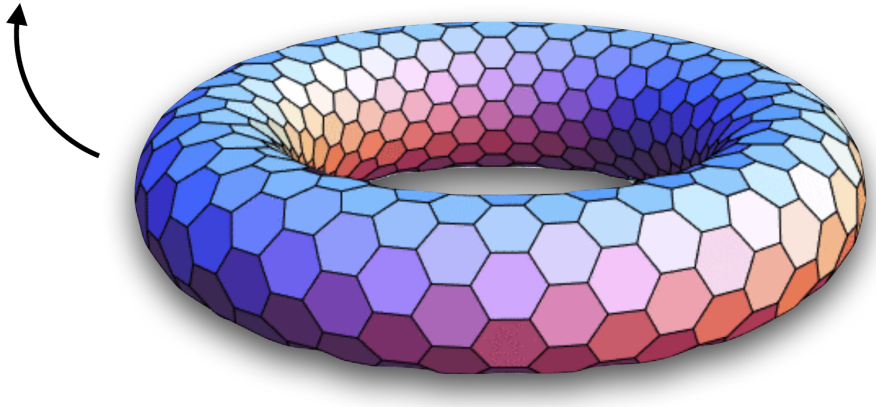
What is the problem with **high-dimensional** things?



Principal Component Analysis

What is the problem with **high-dimensional** things?

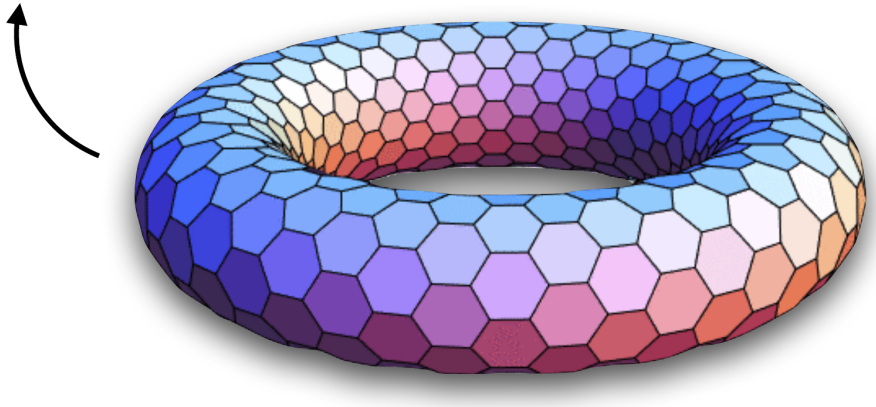
Hard to **visualise**



Principal Component Analysis

What is the problem with **high-dimensional** things?

Hard to **visualise**

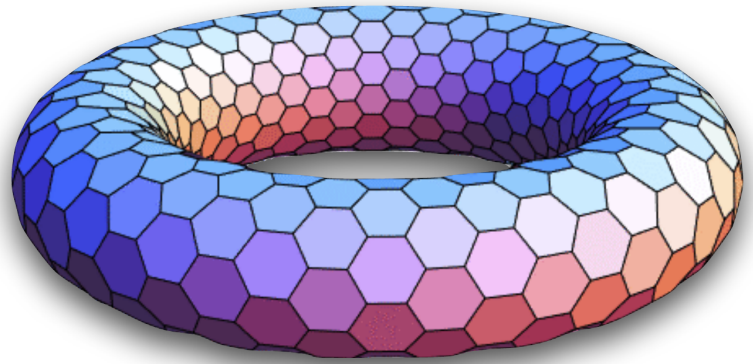


Principal Component Analysis

What is the problem with **high-dimensional** things?

Hard to **visualise**

Algorithms tend to **get slow**

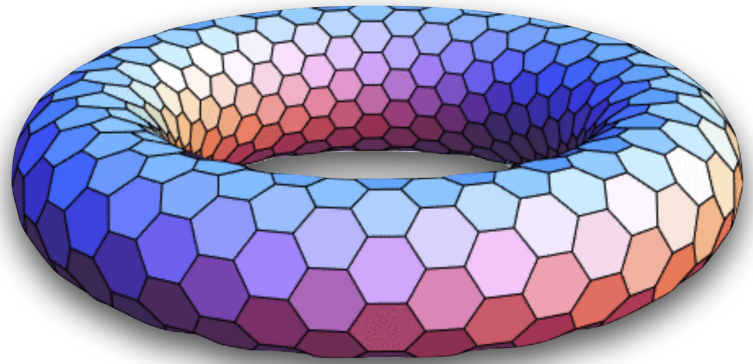


Principal Component Analysis

What is the problem with **high-dimensional** things?

Hard to **visualise**

Algorithms tend to **get slow**

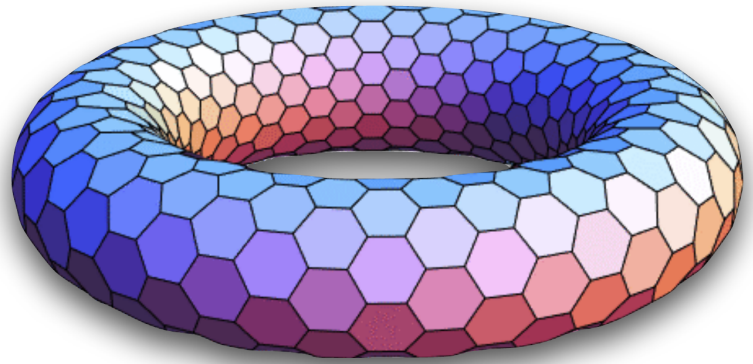


Principal Component Analysis

What is the problem with **high-dimensional** things?

Hard to **visualise**

Algorithms tend to **get slow**



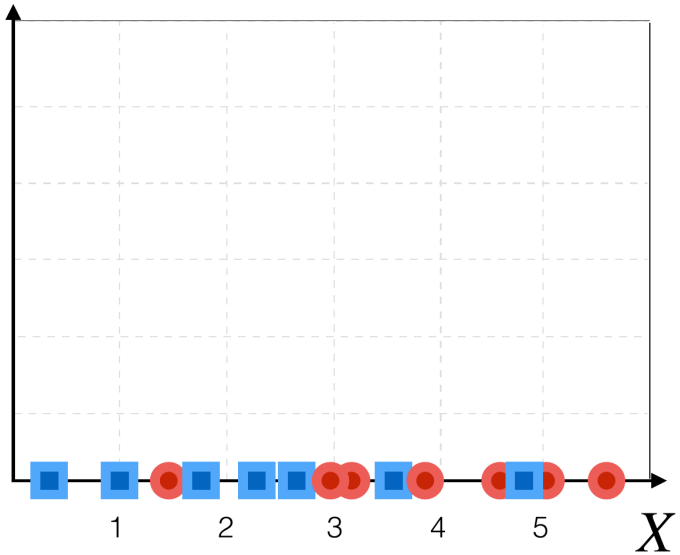
Methods trained on high-dimensional data suffer from the **curse of dimensionality**

Principal Component Analysis

What is the **curse** of **dimensionality**?

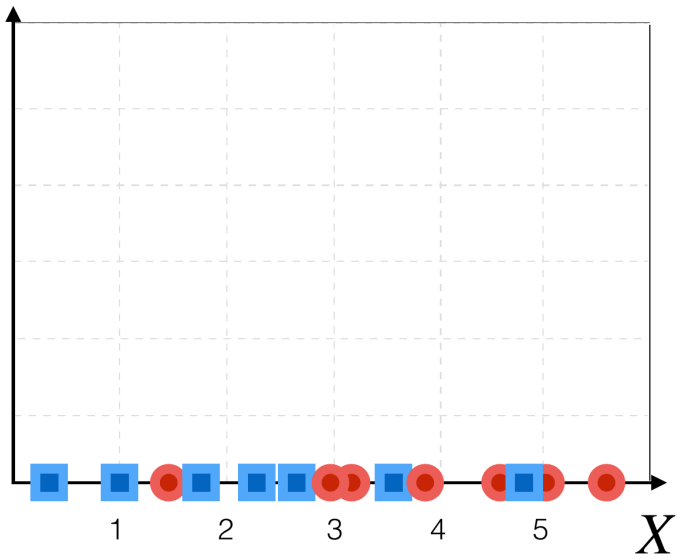
Principal Component Analysis

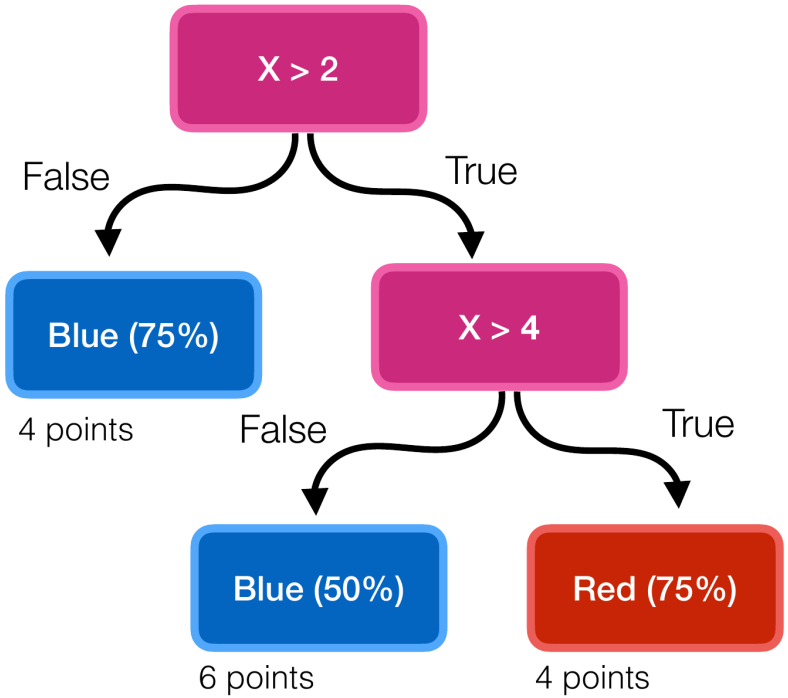
What is the **curse** of **dimensionality**?



Principal Component Analysis

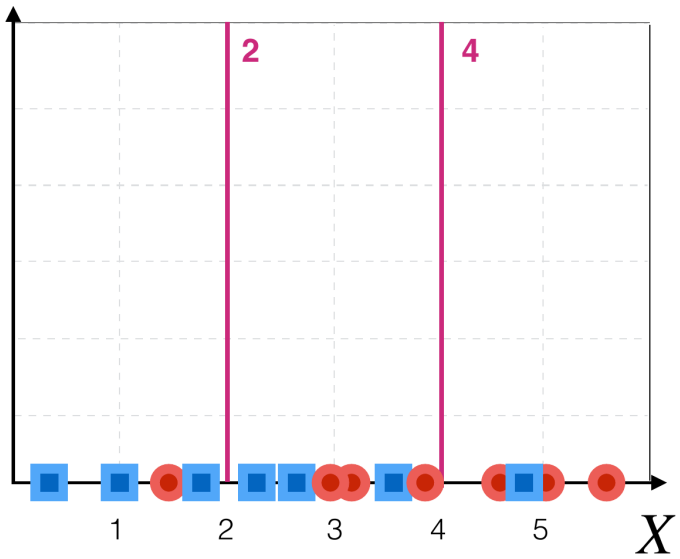
What is the **curse** of **dimensionality**?

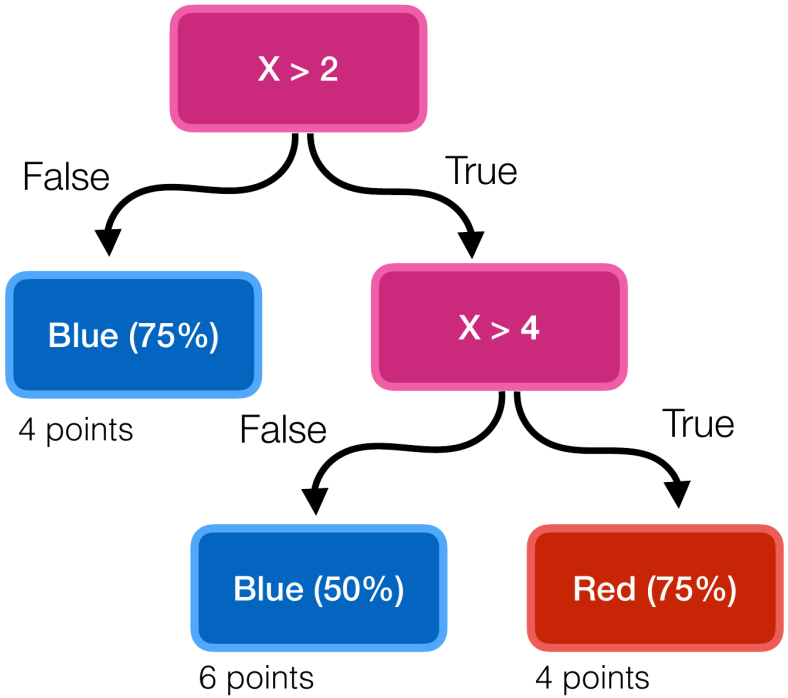




Principal Component Analysis

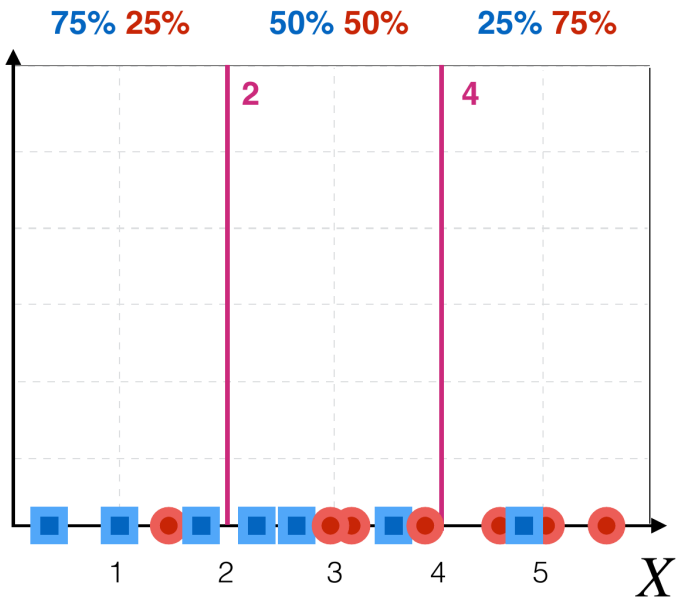
What is the **curse** of **dimensionality**?

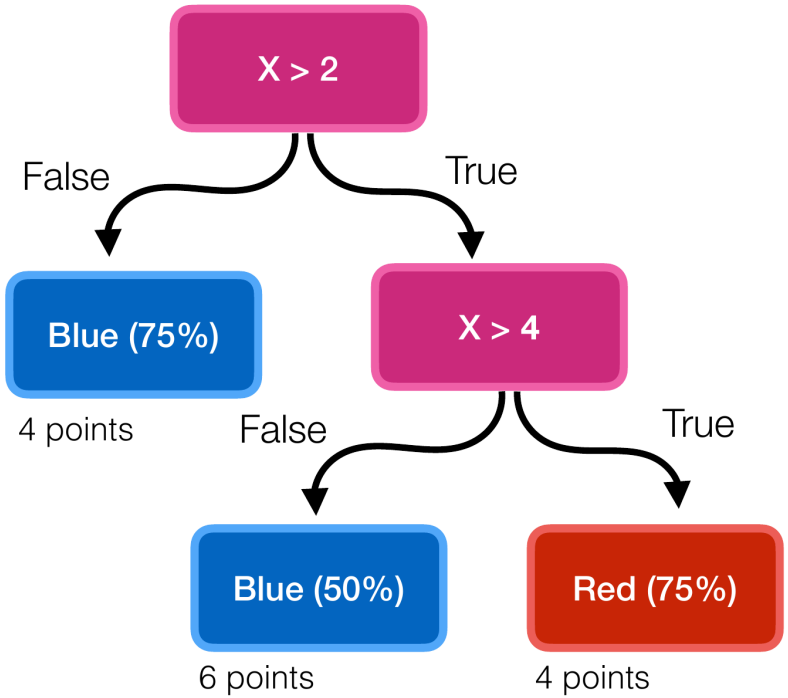




Principal Component Analysis

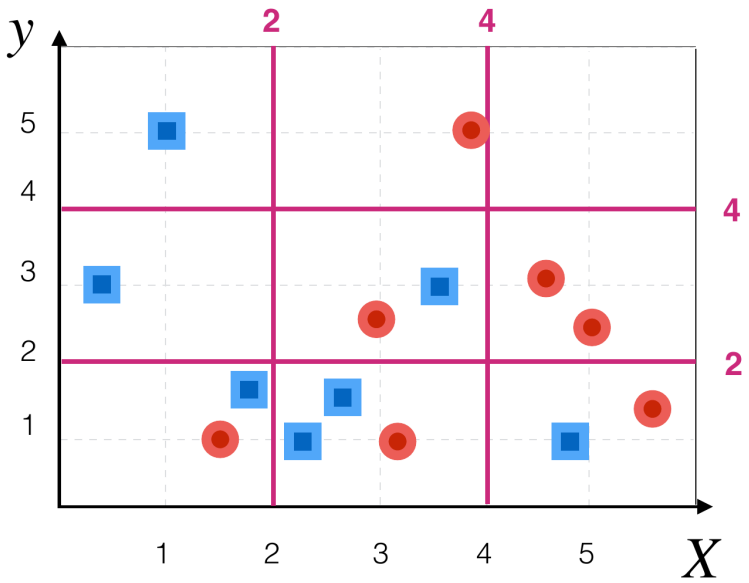
What is the **curse** of **dimensionality**?





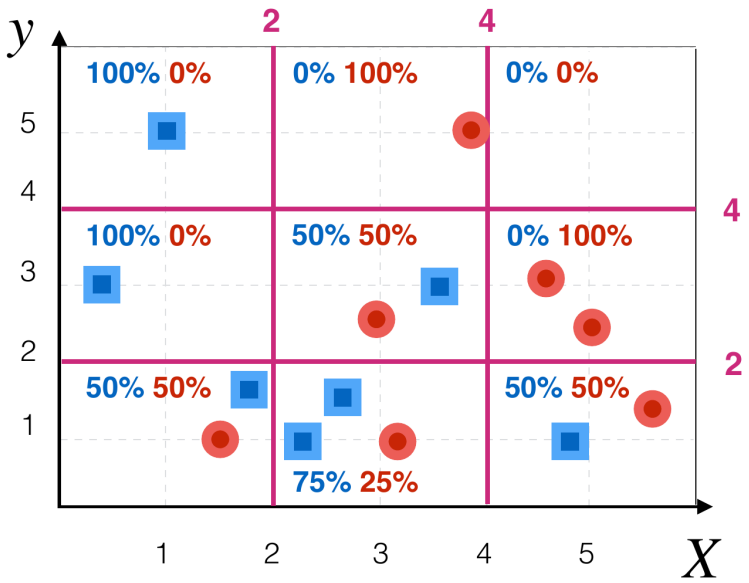
Principal Component Analysis

What is the **curse** of **dimensionality**?



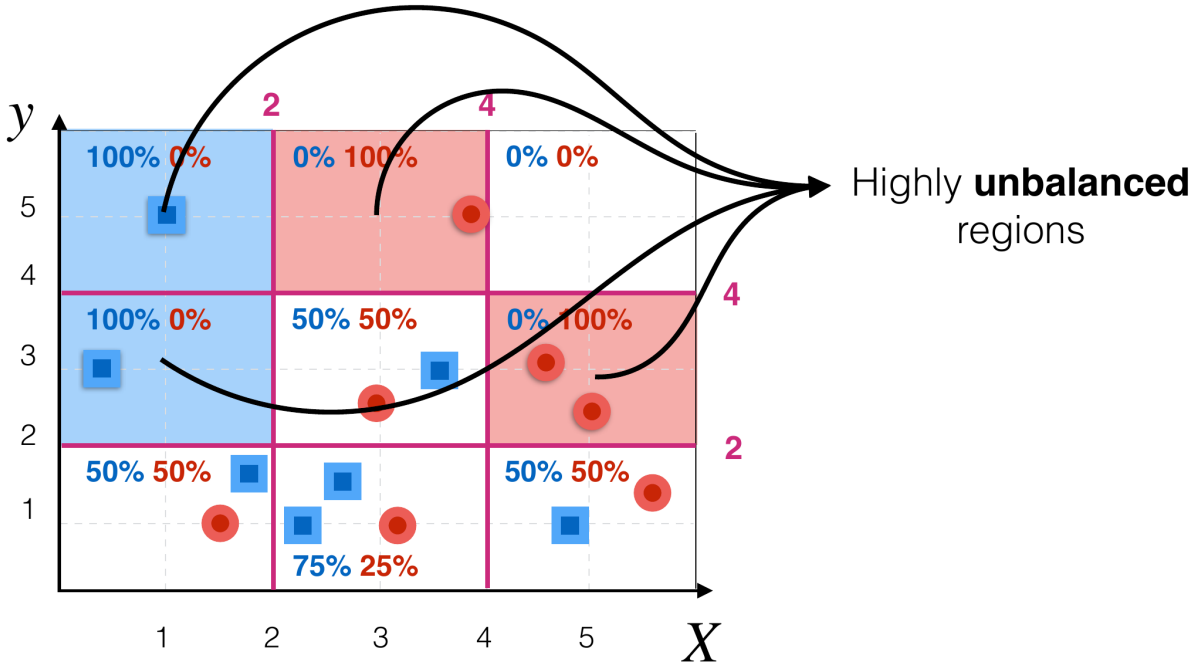
Principal Component Analysis

What is the **curse** of **dimensionality**?



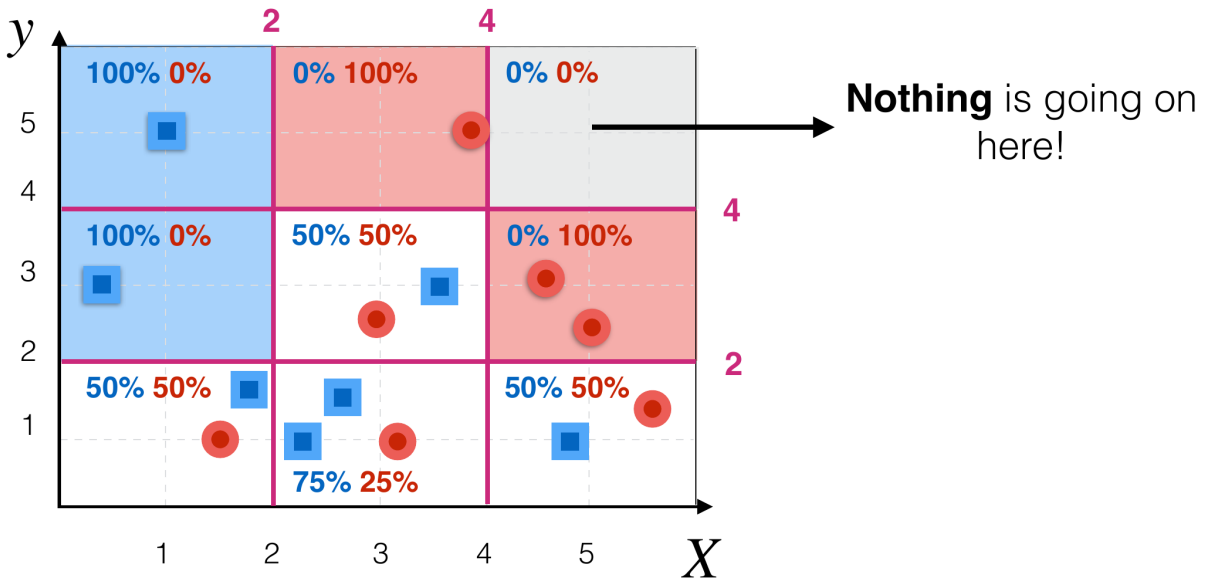
Principal Component Analysis

What is the **curse** of **dimensionality**?



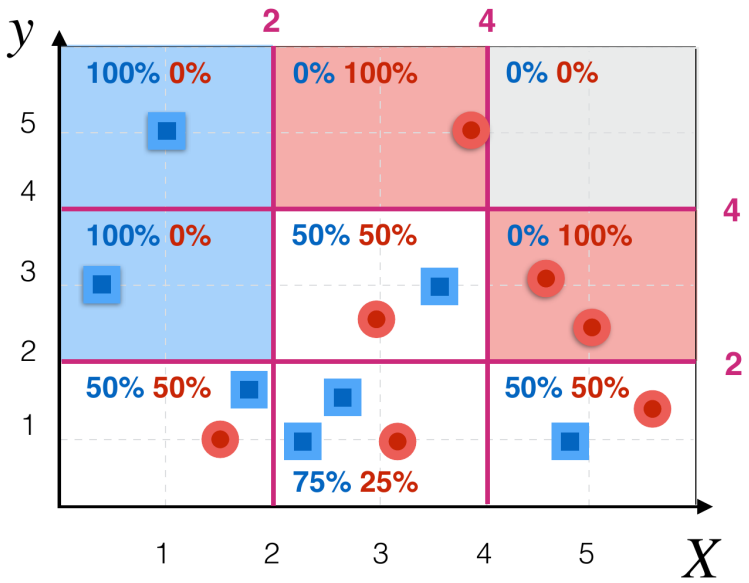
Principal Component Analysis

What is the **curse** of **dimensionality**?



Principal Component Analysis

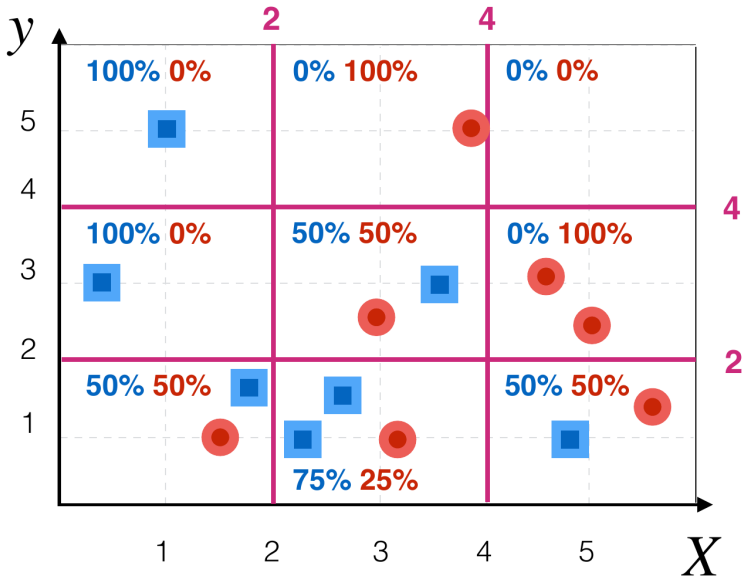
What is the **curse** of **dimensionality**?



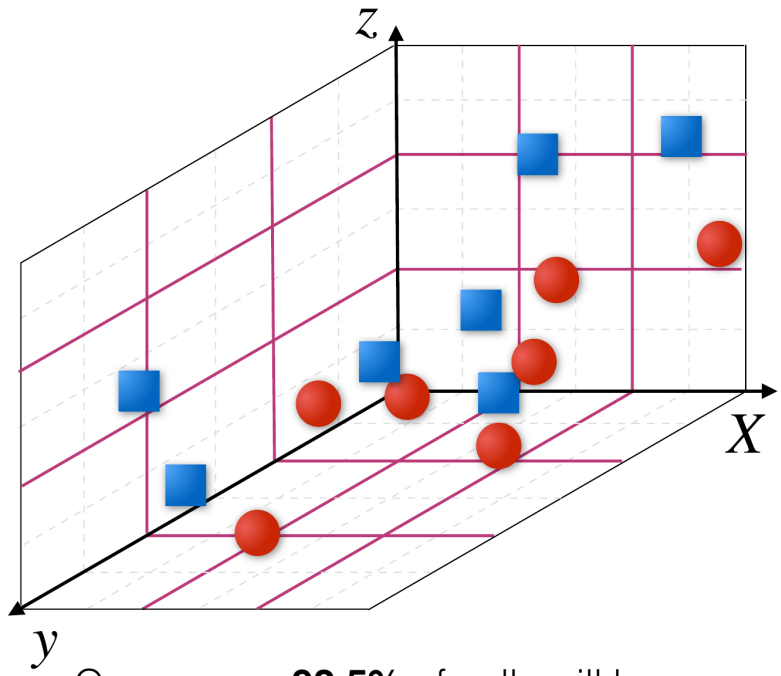
On average **55.5%** of cells will be either **empty** or **singletons**

Principal Component Analysis

What is the **curse** of **dimensionality**?



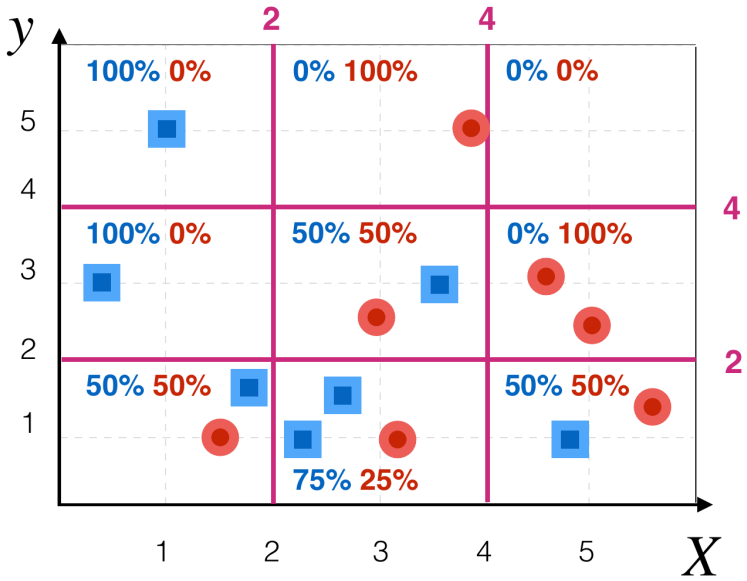
On average **55.5%** of cells will be either **empty** or **singletons**



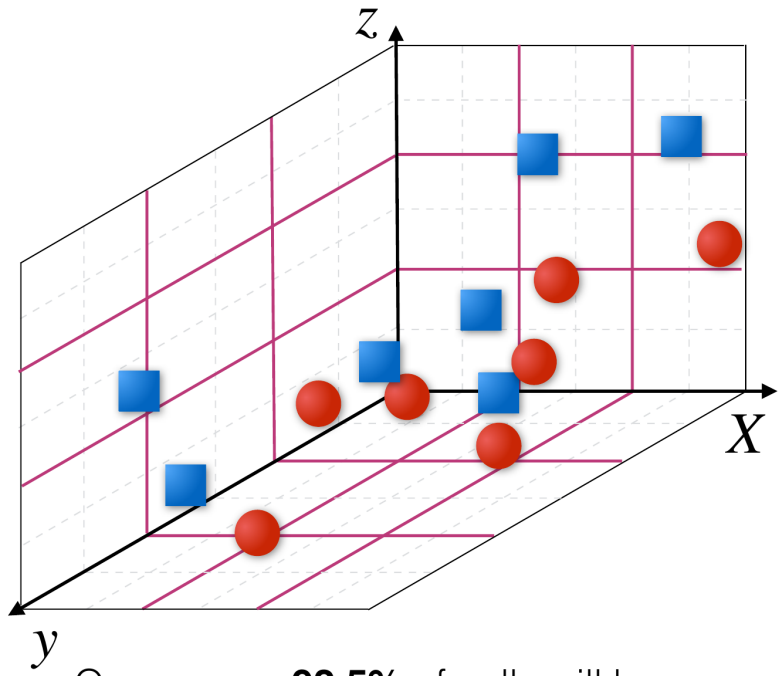
On average **92.5%** of cells will be either **empty** or **singletons**

What is the **curse** of **dimensionality**?

In order to keep **high-dimensional** space reasonably covered you need **a lot more data**



On average **55.5%** of cells will be either **empty** or **singletons**



On average **92.5%** of cells will be either **empty** or **singletons**

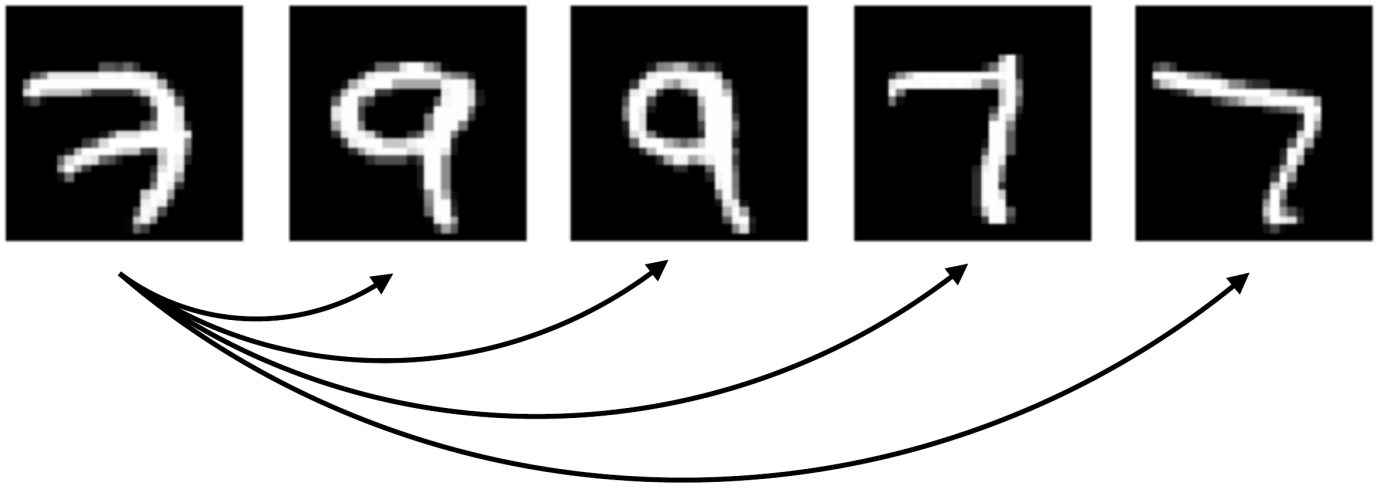
Principal Component Analysis

What is the **curse** of **dimensionality**?
(part **II**)

Principal Component Analysis

What is the **curse** of **dimensionality**?
(part **II**)

Distances become **similar** in high-dimensional space

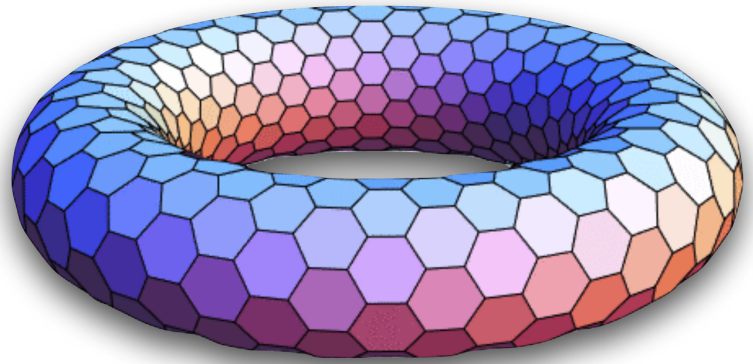


Principal Component Analysis

What is the problem with **high-dimensional** things?

Hard to **visualise**

Algorithms tend to **get slow**



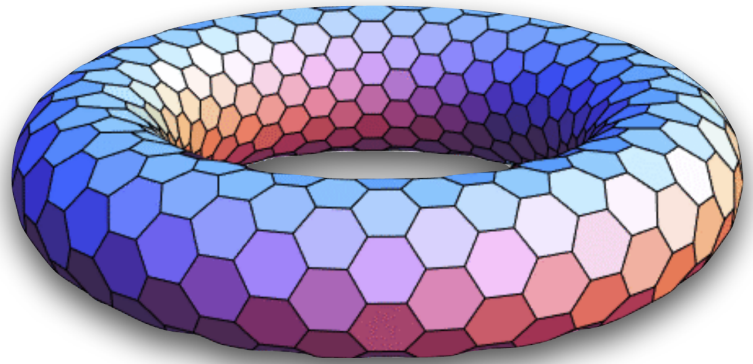
Methods trained on high-dimensional data suffer from the **curse of dimensionality**

Principal Component Analysis

What is the problem with **high-dimensional** things?

Hard to **visualise**

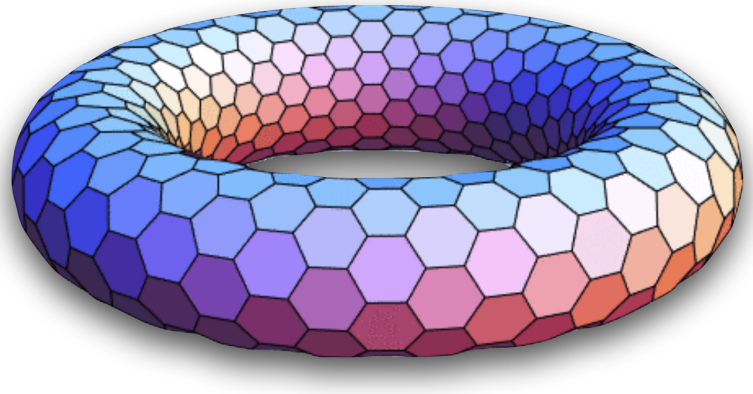
Algorithms tend to **get slow**



You need **more data** and **objects become closer** in high-dimensional space

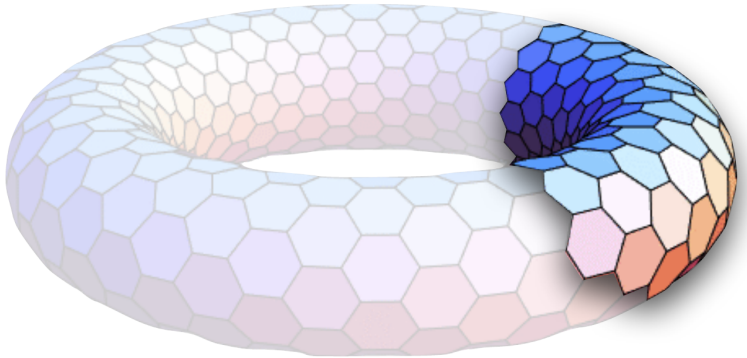
Principal Component Analysis

Feature **extraction** vs feature **elimination**



Principal Component Analysis

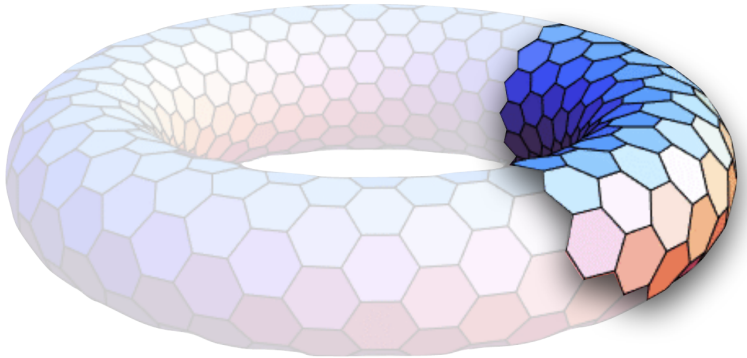
Feature **extraction** vs feature **elimination**



Keeping only **few** original **features**

Principal Component Analysis

Feature **extraction** vs feature **elimination**

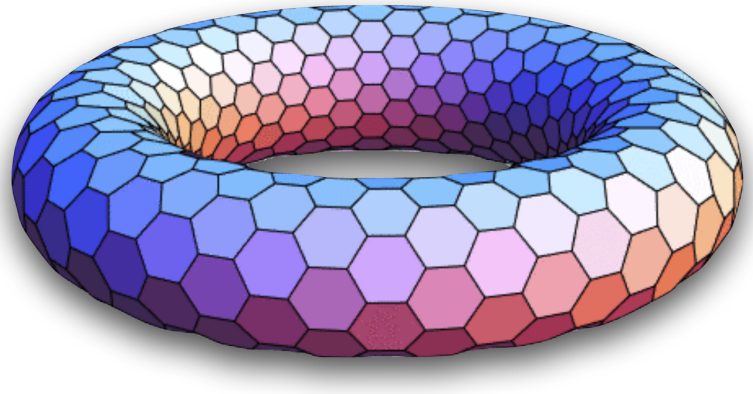


Remove
all **the rest**

Keeping only **few**
original **features**

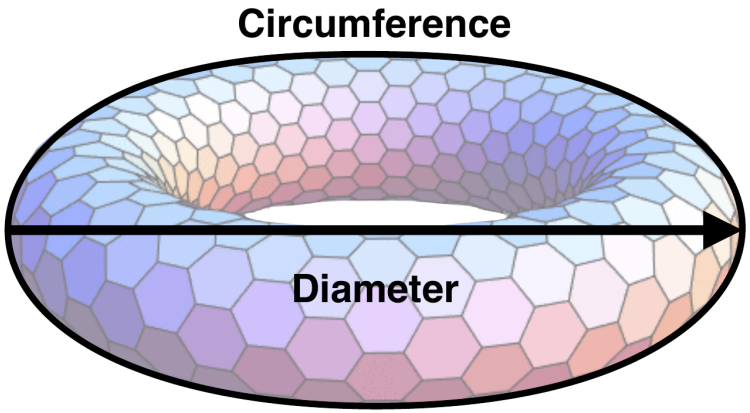
Principal Component Analysis

Feature **extraction** vs feature **elimination**



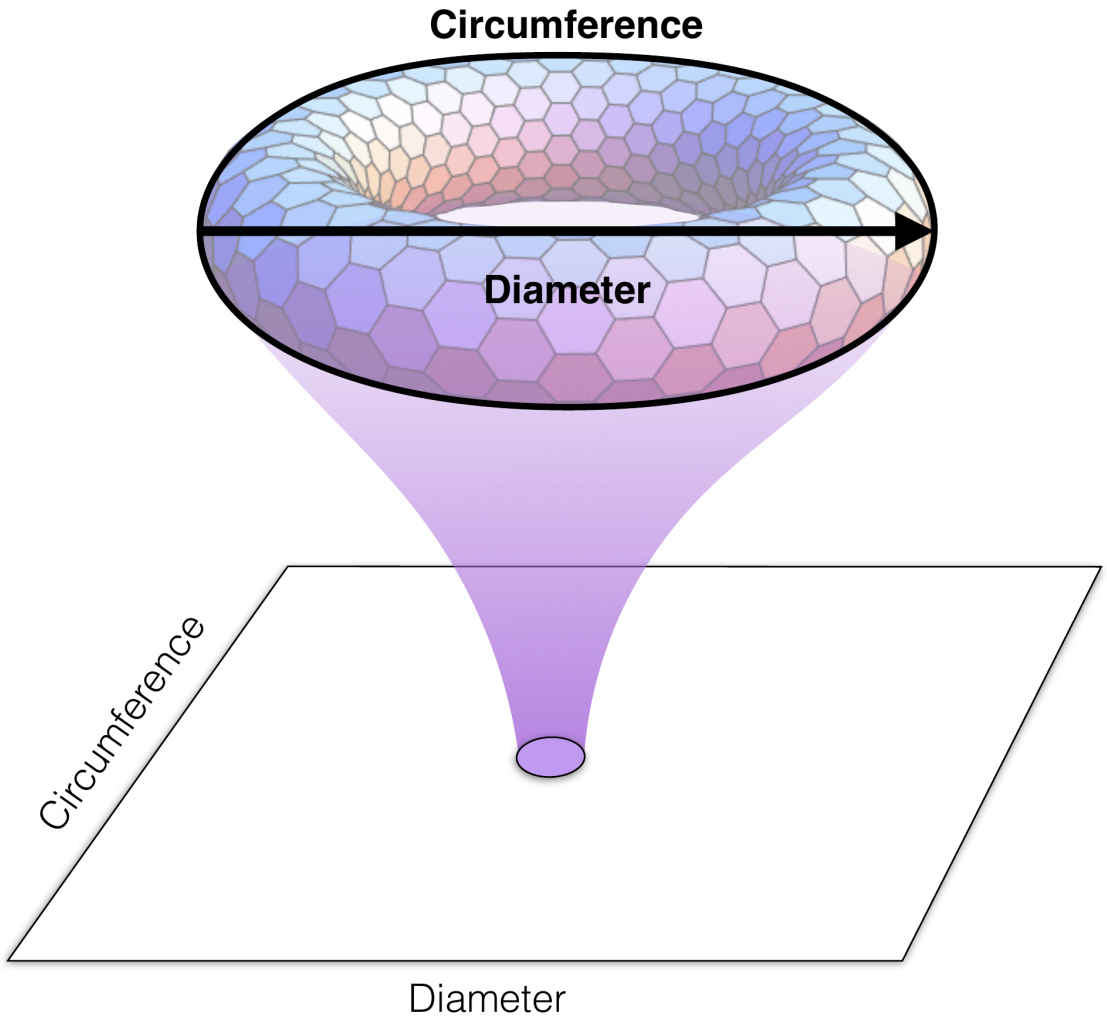
Principal Component Analysis

Feature **extraction** vs feature **elimination**

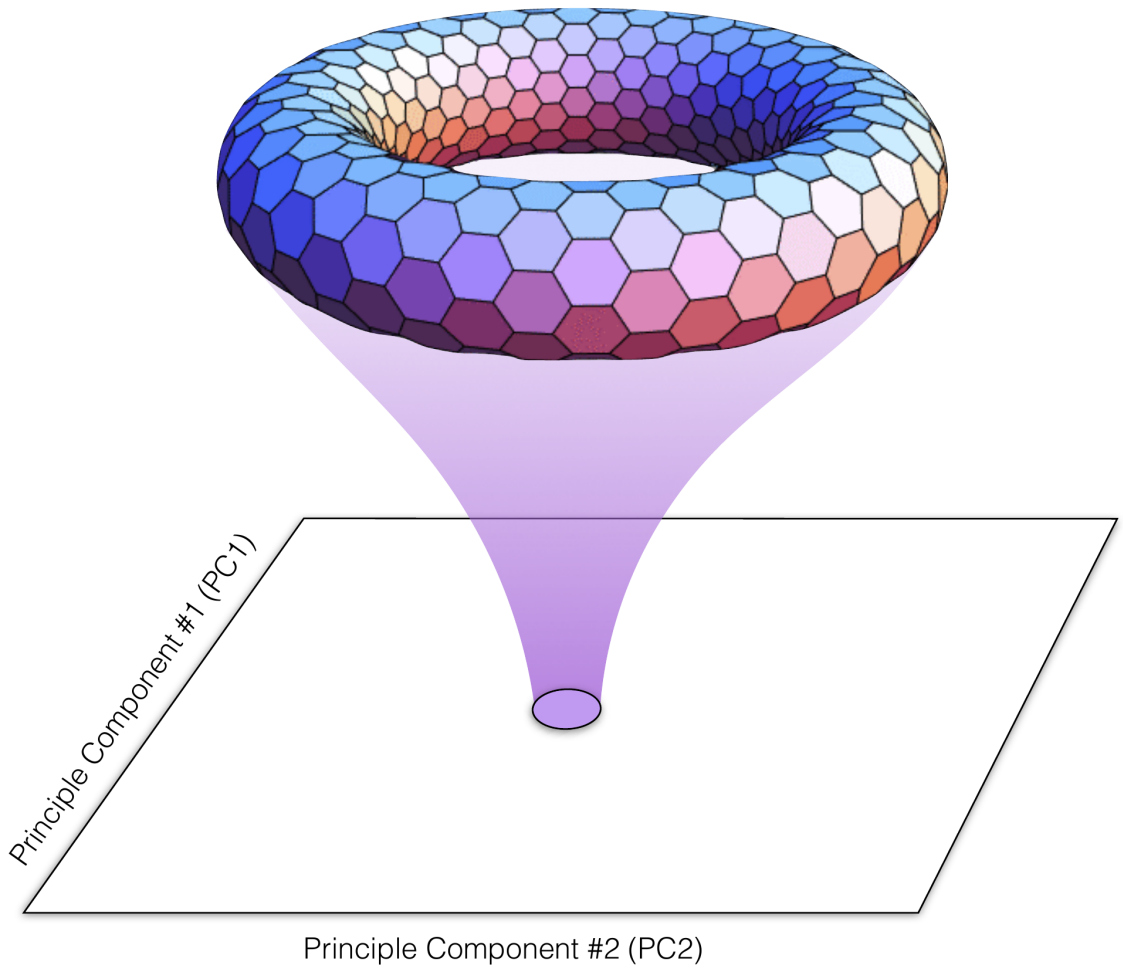


Principal Component Analysis

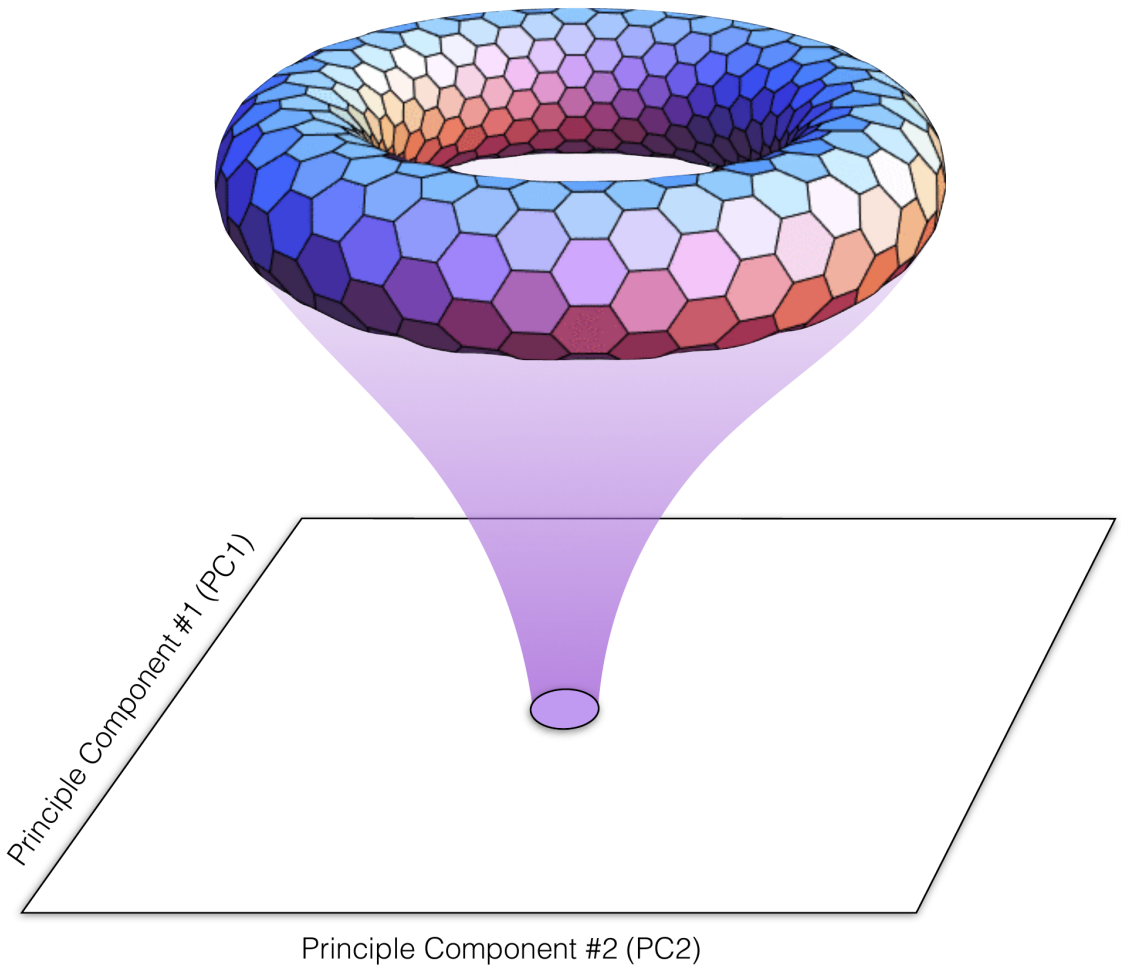
Feature **extraction** vs feature **elimination**



Principle Component Analysis

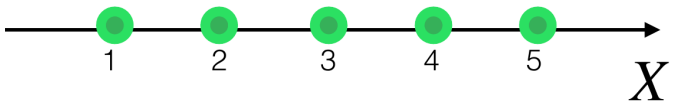


Principle **C**omponent **A**nalysis



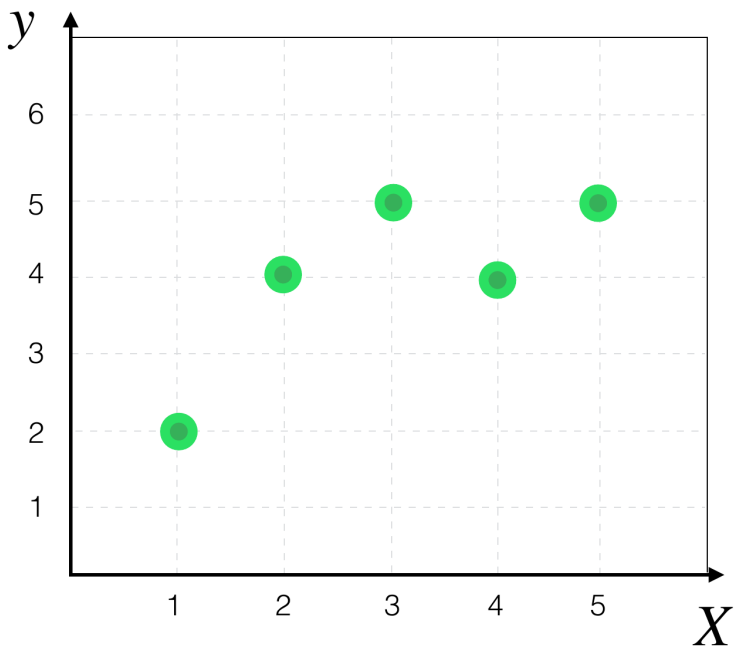
1-Dimensional data

x
1
2
3
4
5



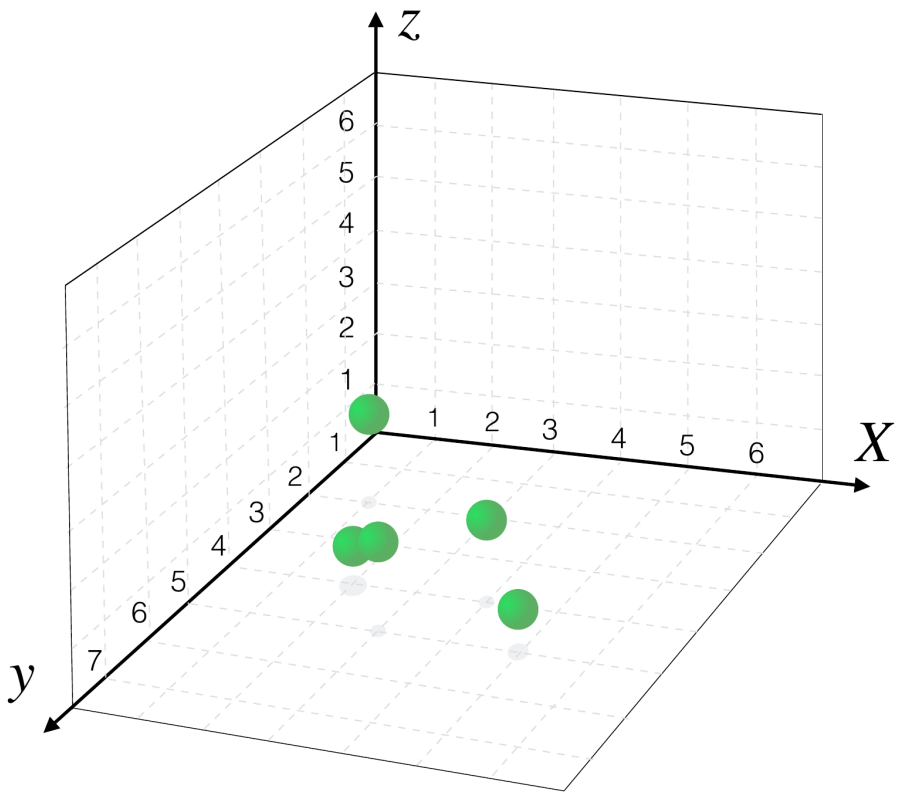
2-Dimensional data

x	y
1	2
2	4
3	5
4	4
5	5

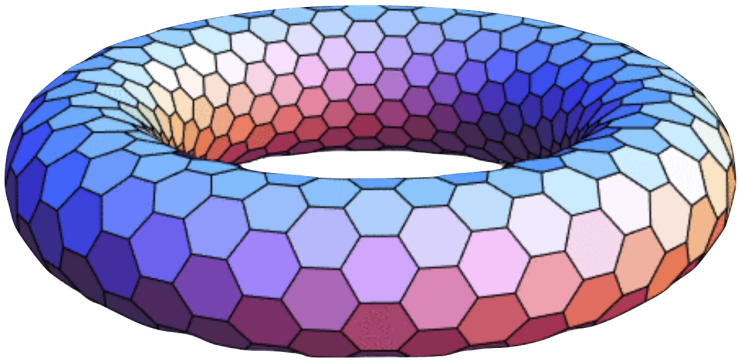


3-Dimensional data

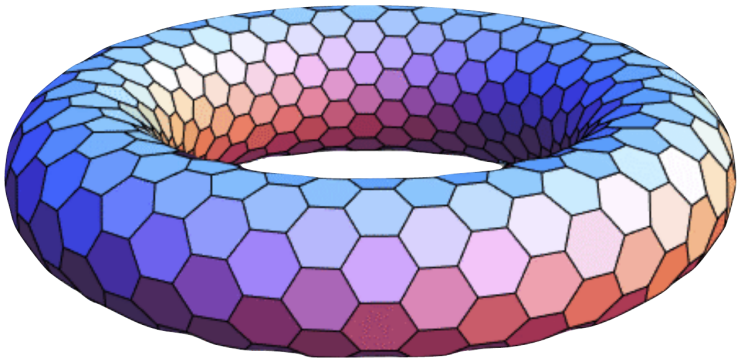
x	y	z
1	2	2
2	4	0.5
3	5	1
4	4	1
5	5	0.5



Principal Component Analysis



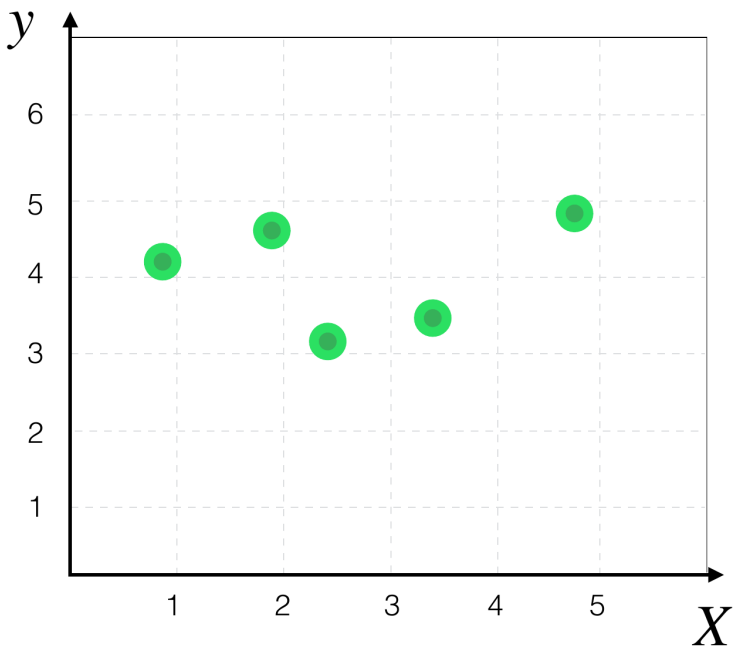
200-Dimensional data?



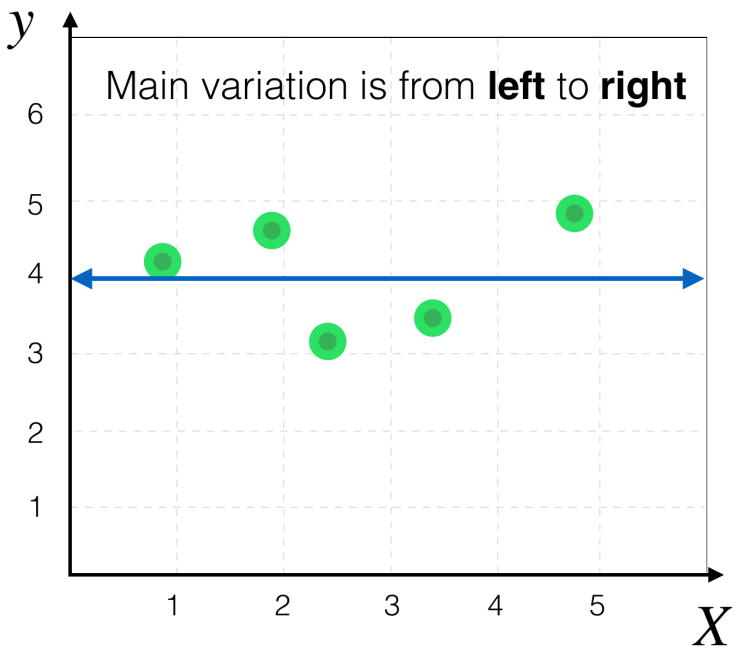
200-Dimensional data?

Are all of these dimensions **equally useful**?

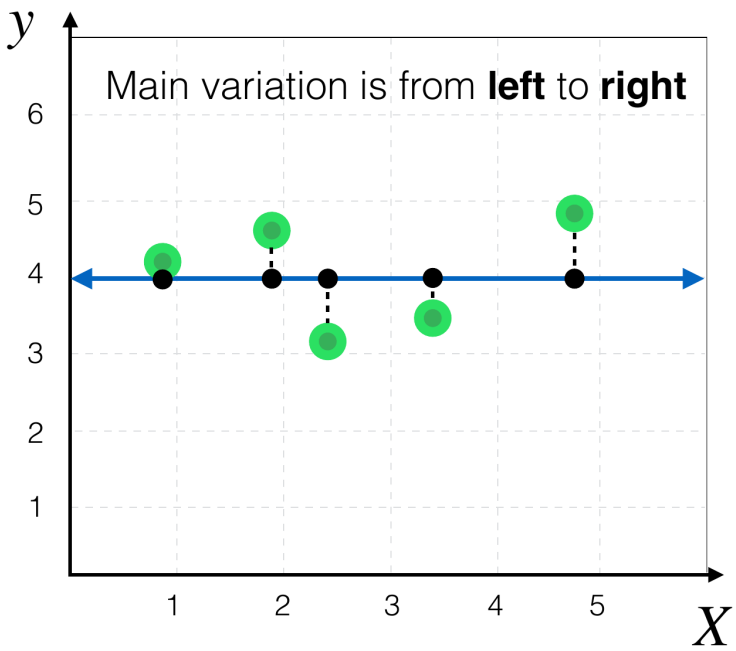
2-D example revisited



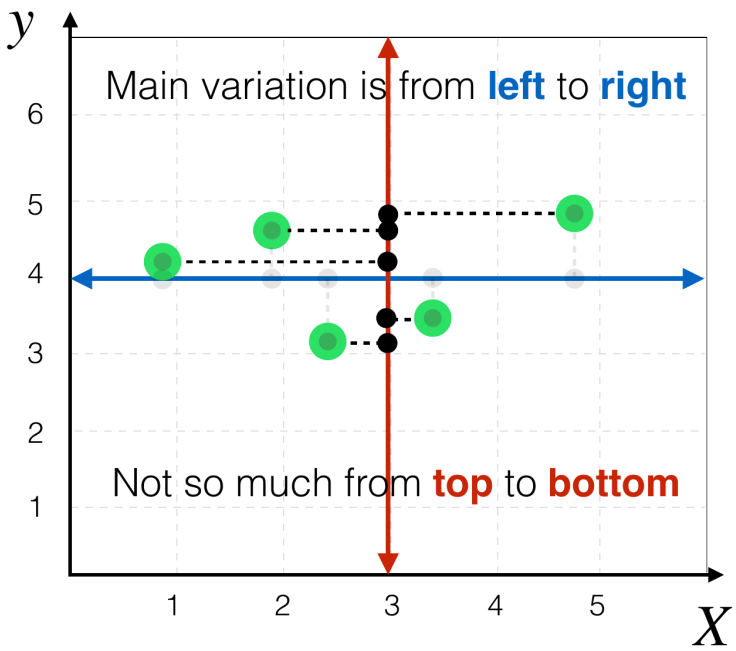
2-D example revisited



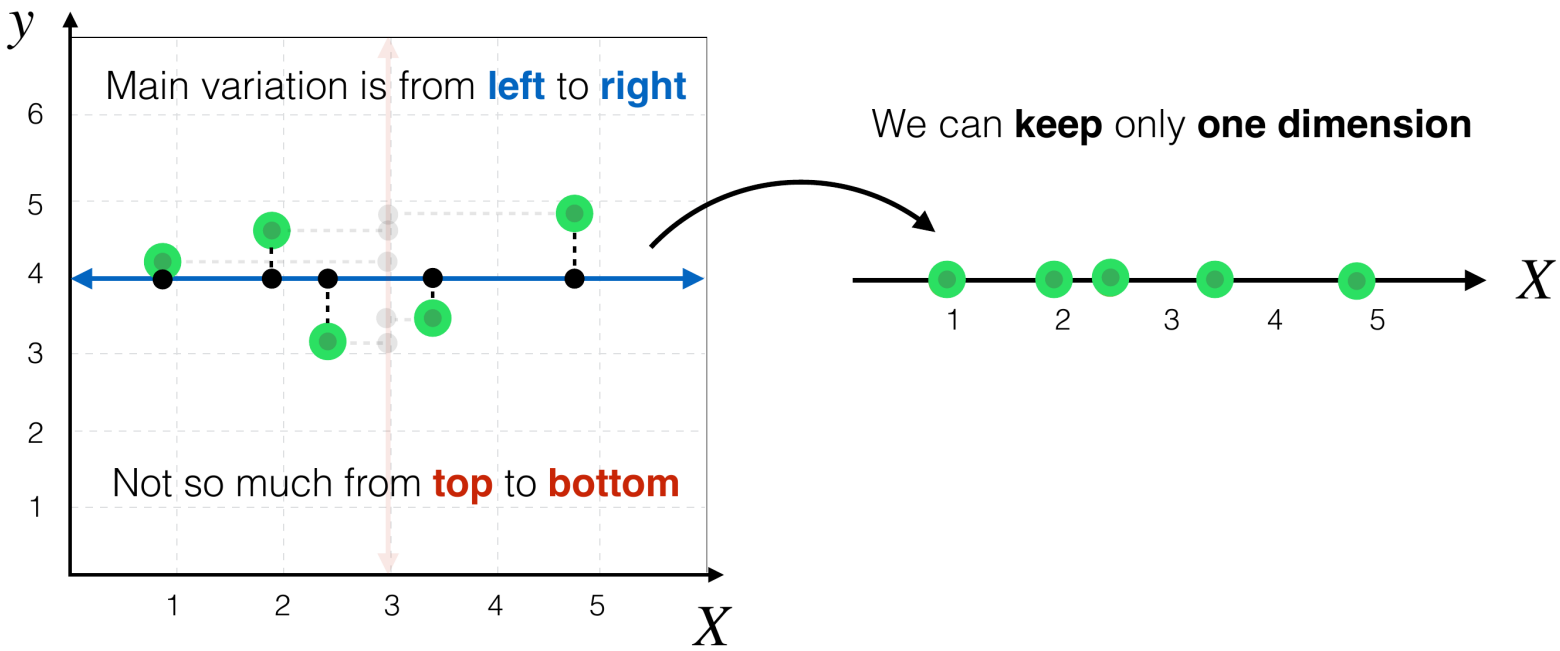
2-D example revisited



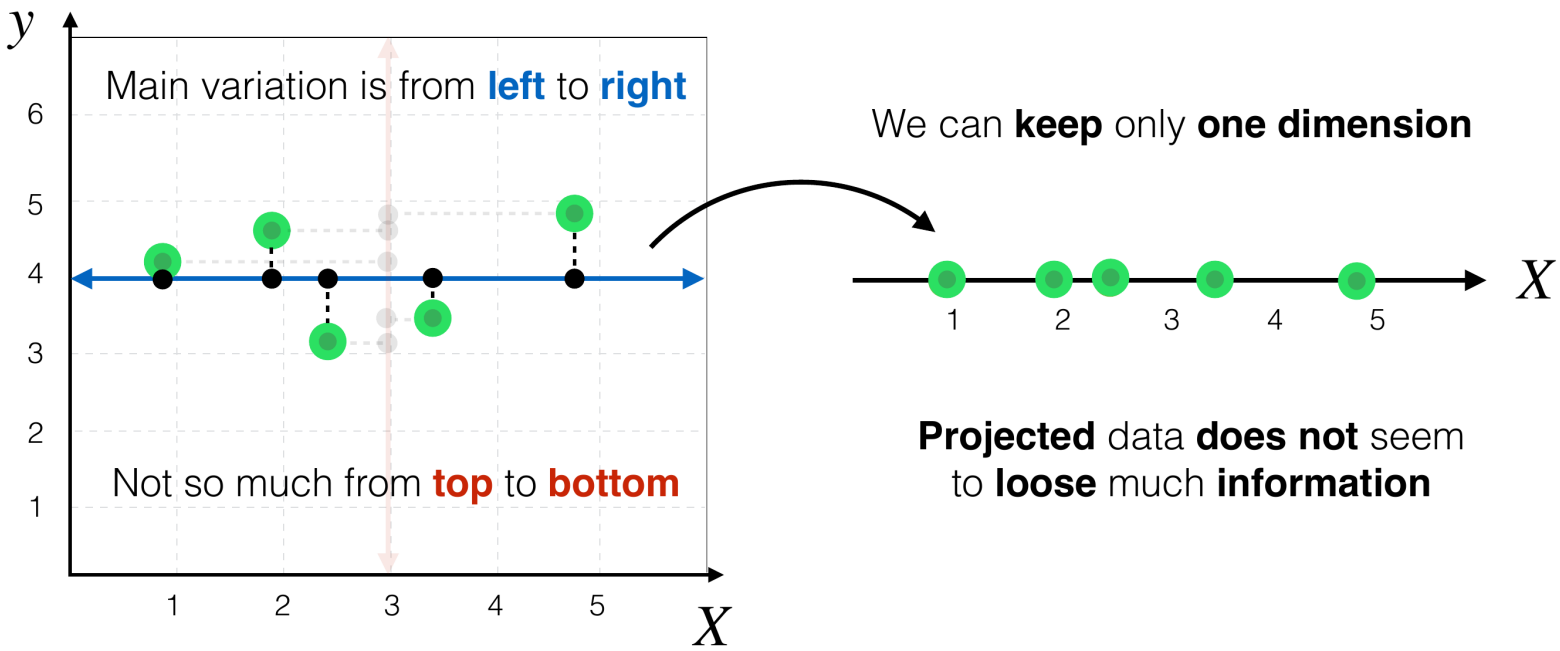
2-D example revisited



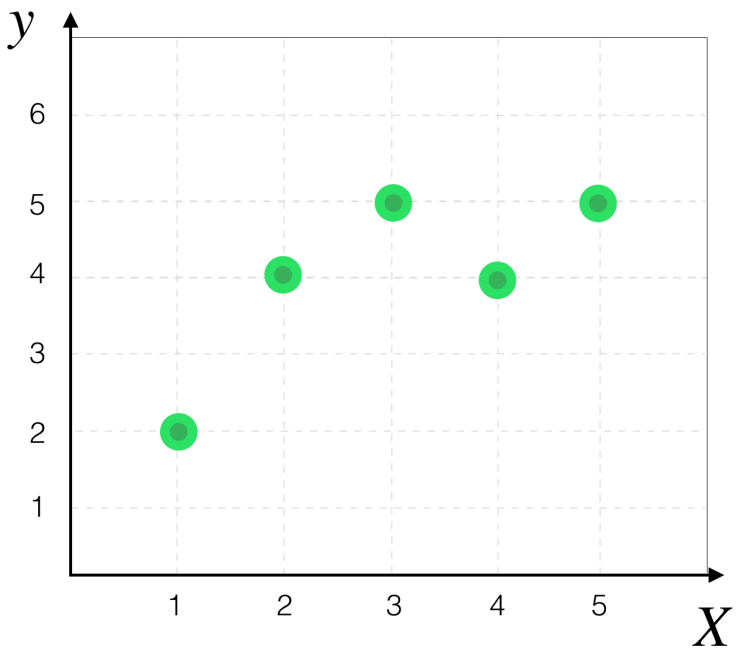
2-D example revisited



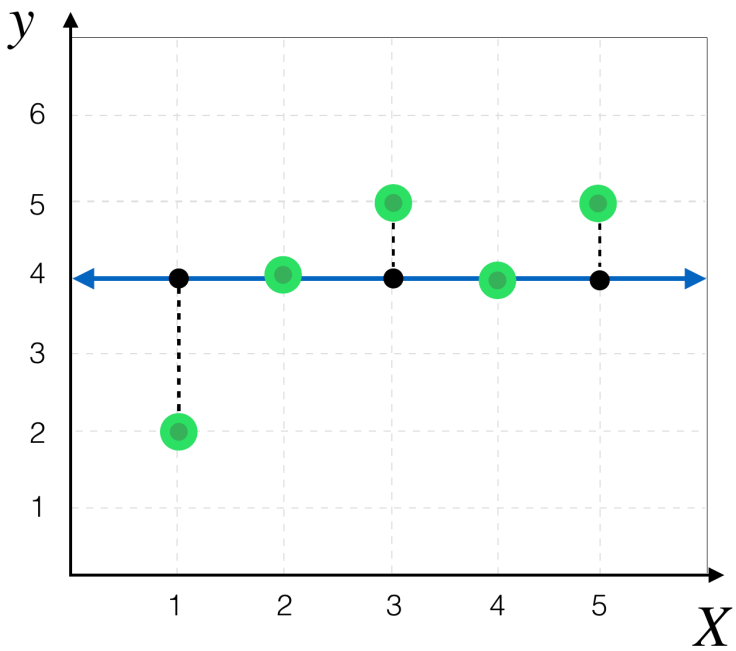
2-D example revisited



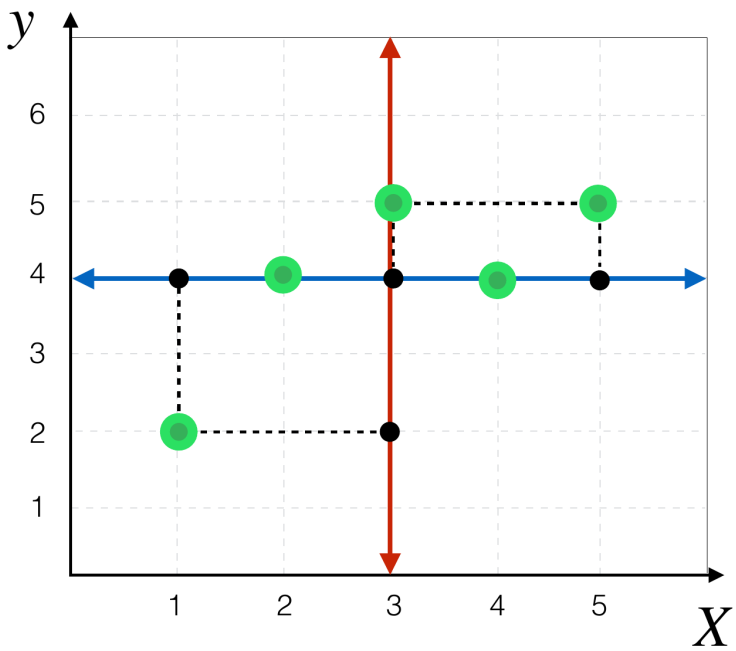
2-D example revisited



2-D example revisited

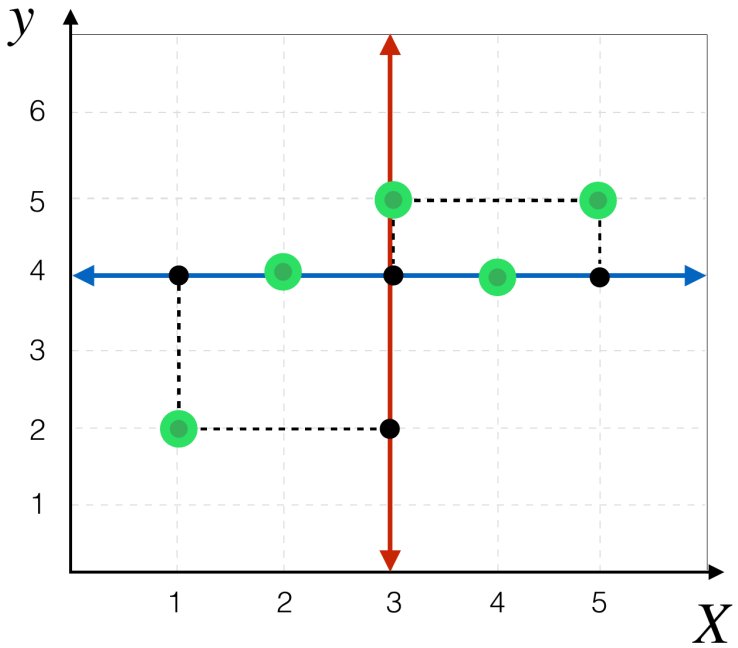


2-D example revisited

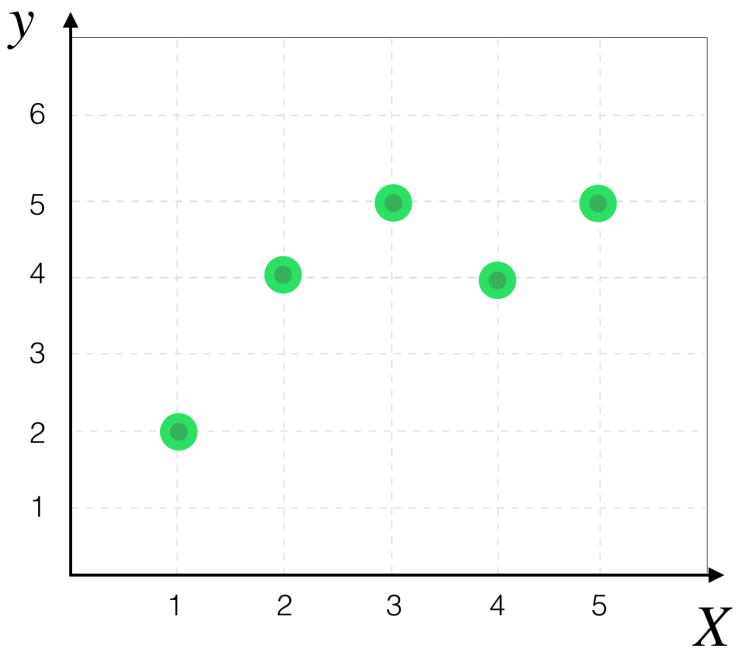


2-D example revisited

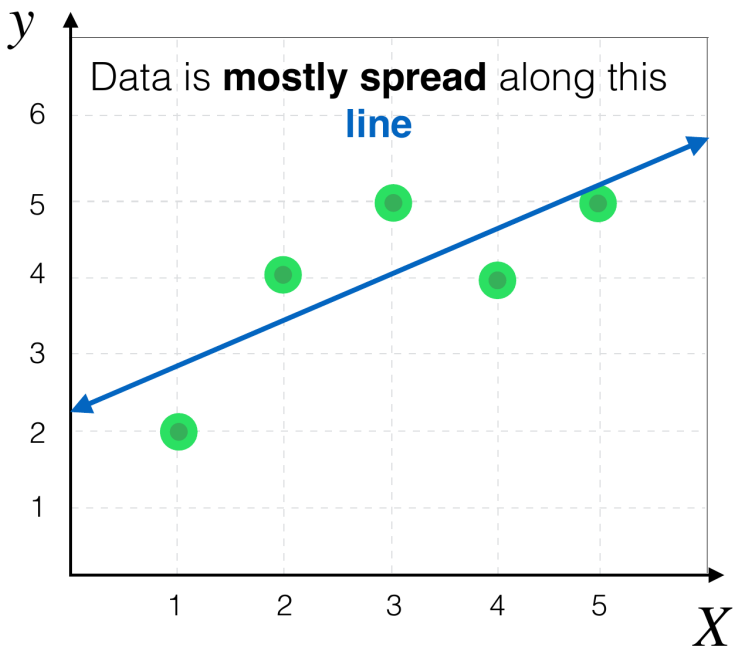
Data seem to be **spread more equally** along **X** and **y** axes



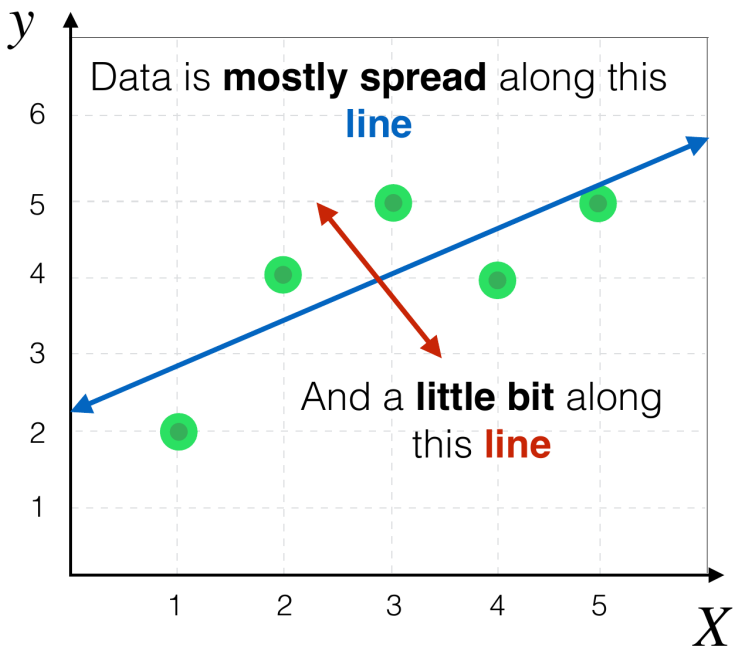
2-D example revisited



2-D example revisited

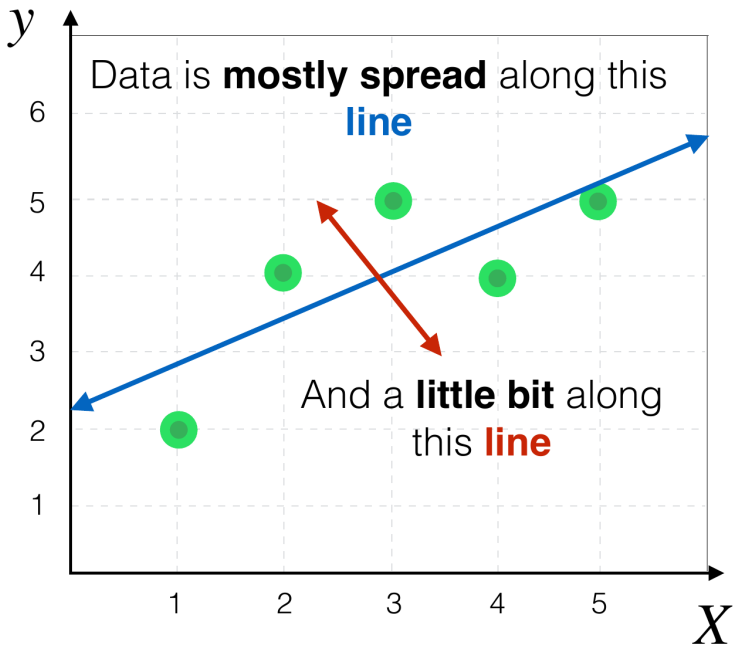


2-D example revisited



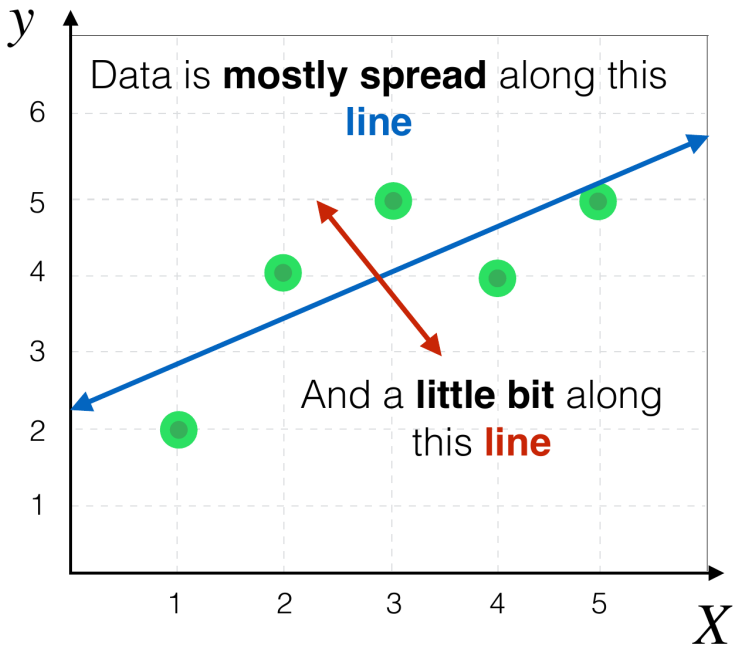
2-D example revisited

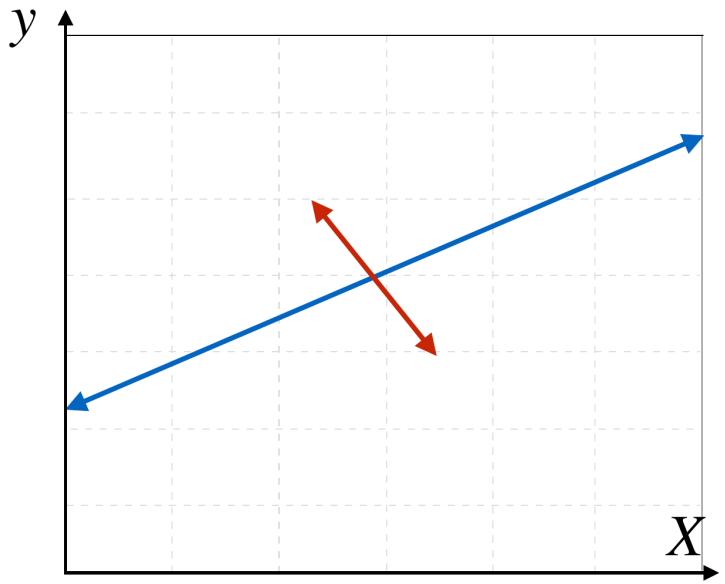
How about we make **new axes** from **these lines**?



2-D example revisited

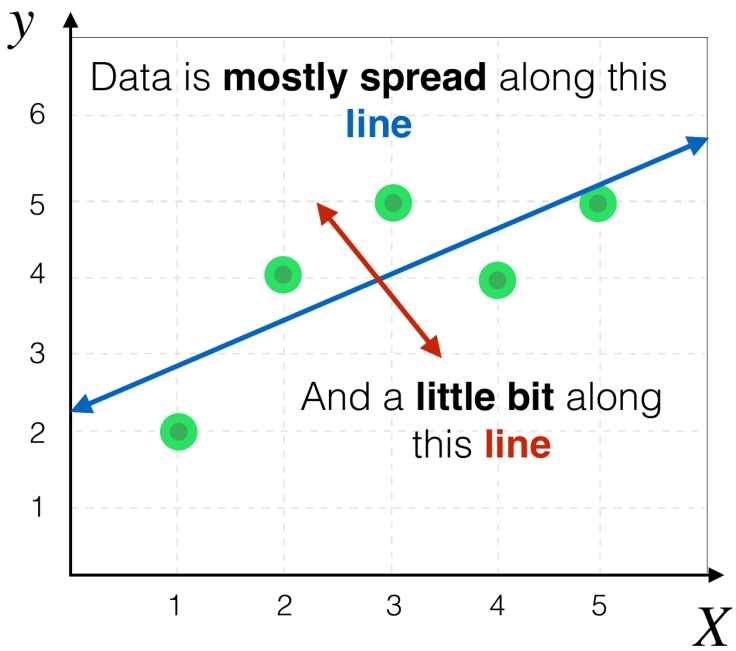
How about we make **new axes** from **these lines**?

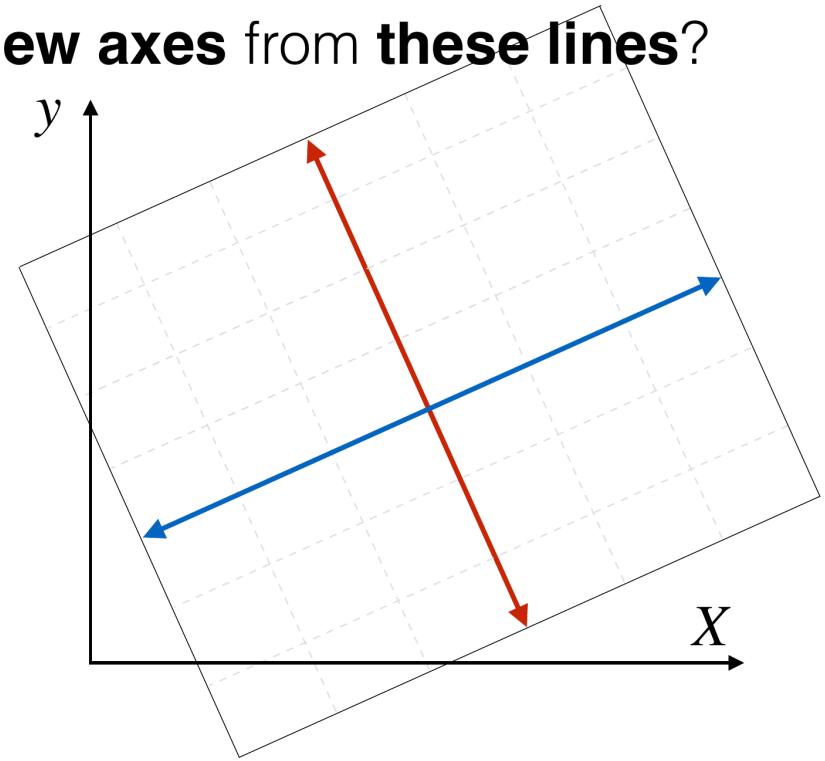




2-D example revisited

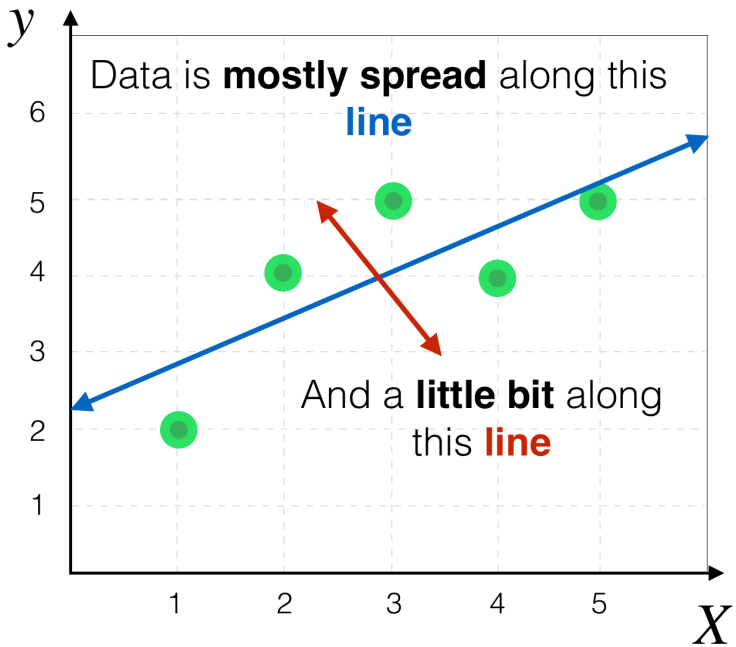
How about we make **new axes** from **these lines**?

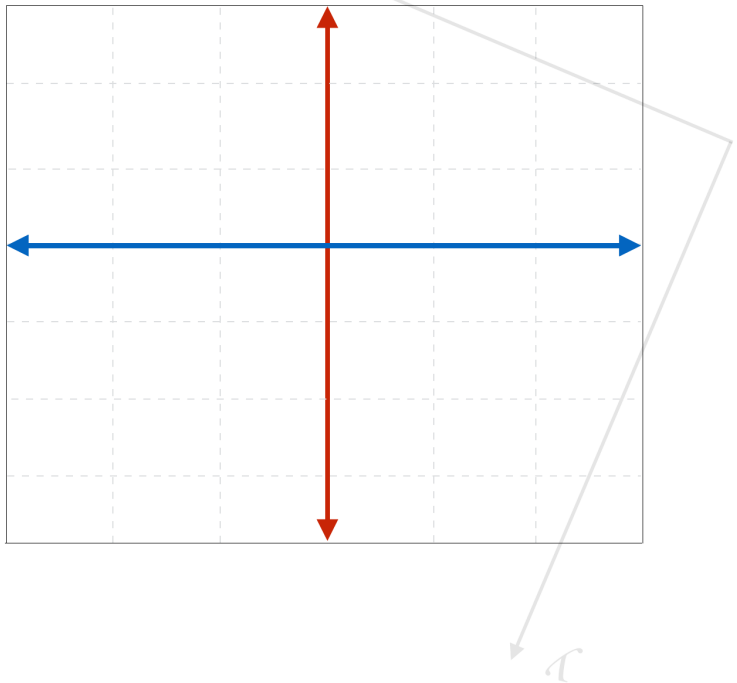




2-D example revisited

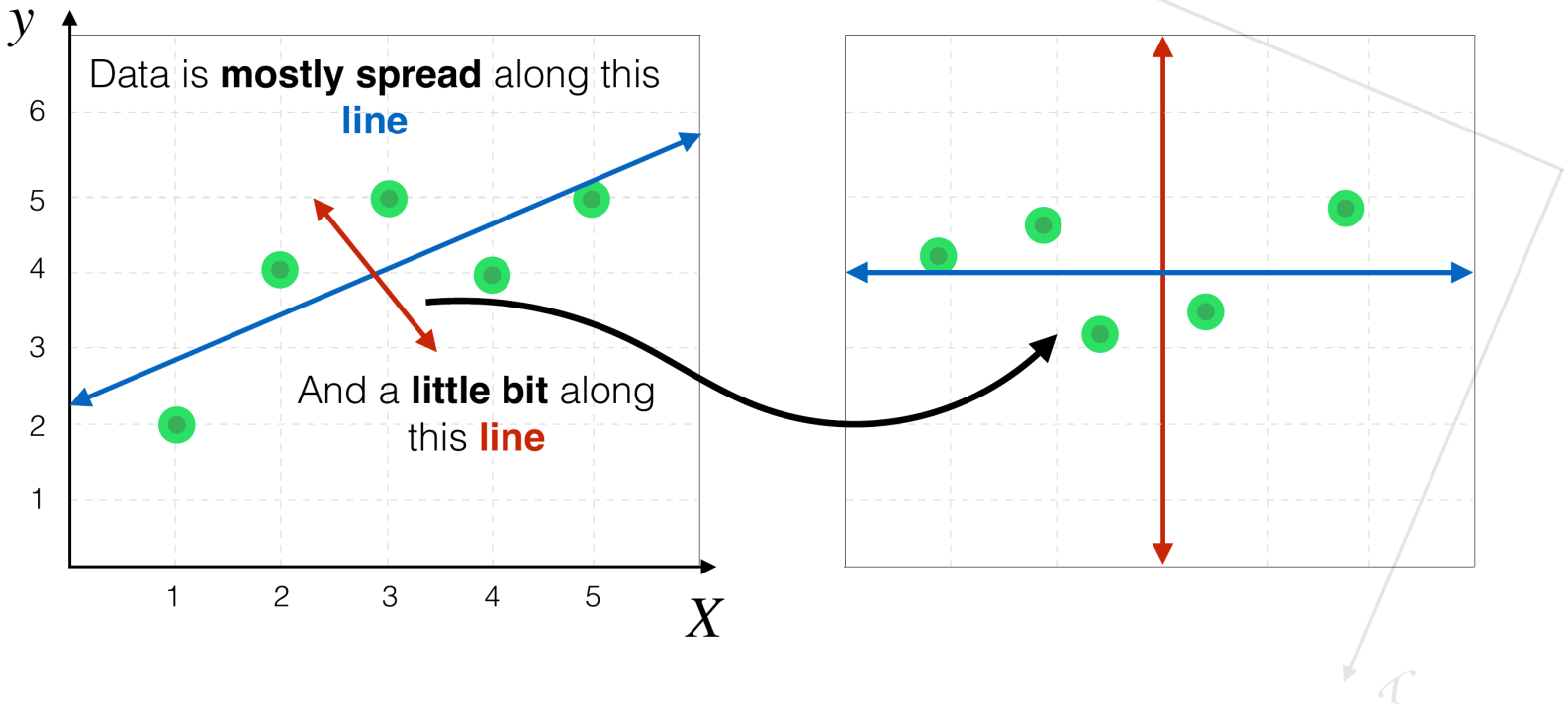
How about we make **new axes** from **these lines**?





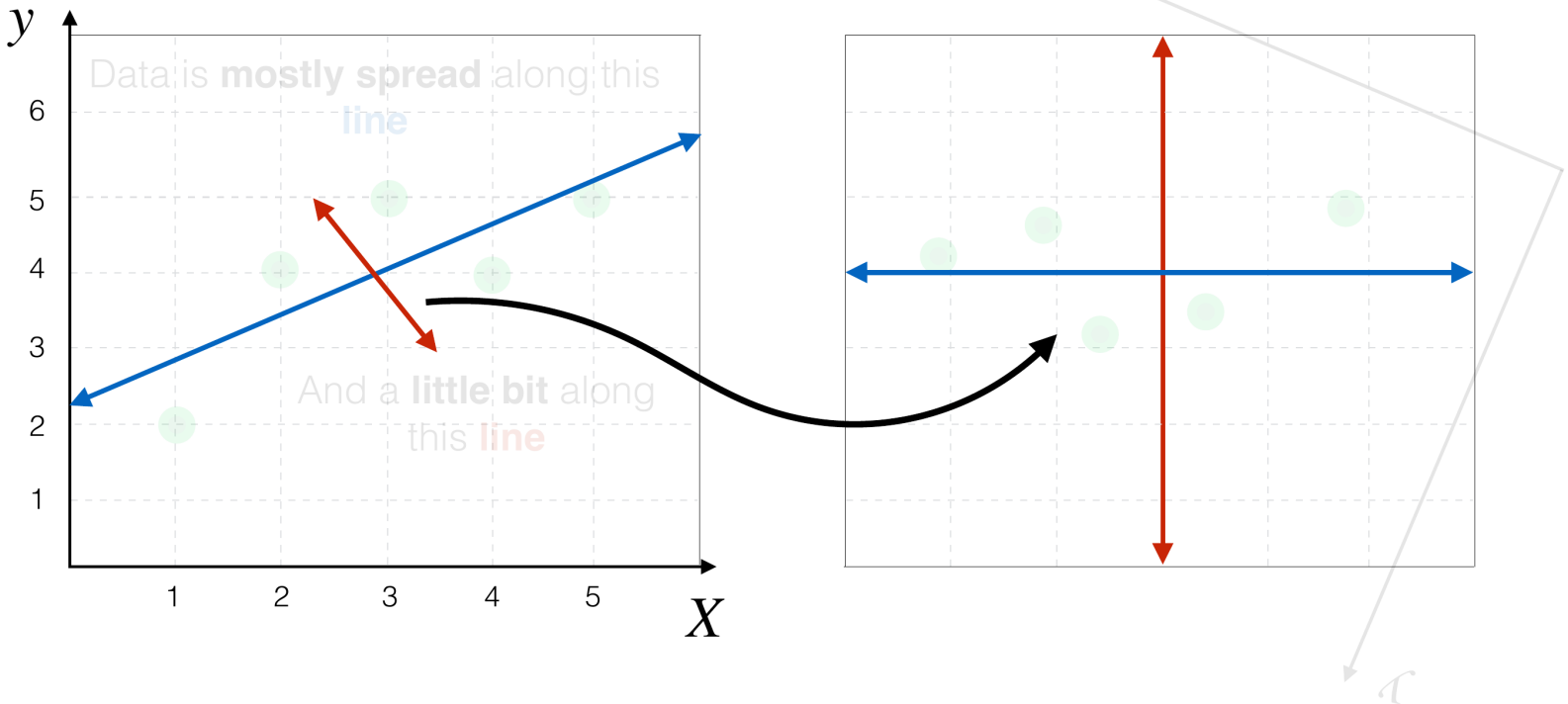
2-D example revisited

How about we make **new axes** from **these lines**?



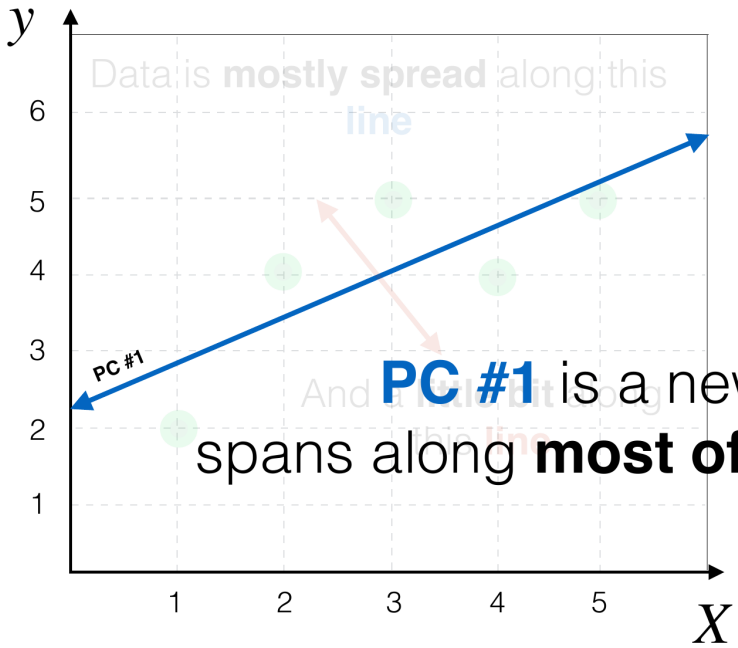
2-D example revisited

These **new axes** are called principle components



2-D example revisited

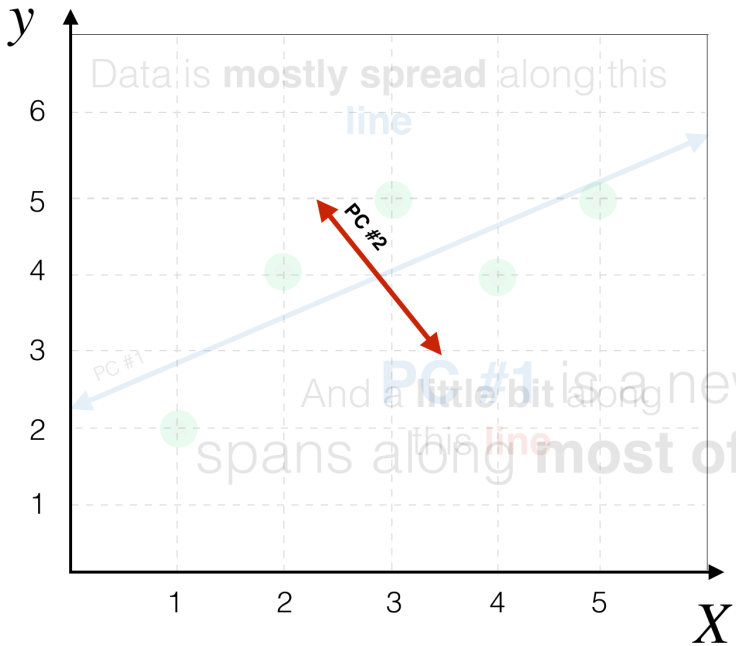
These **new axes** are called principle components

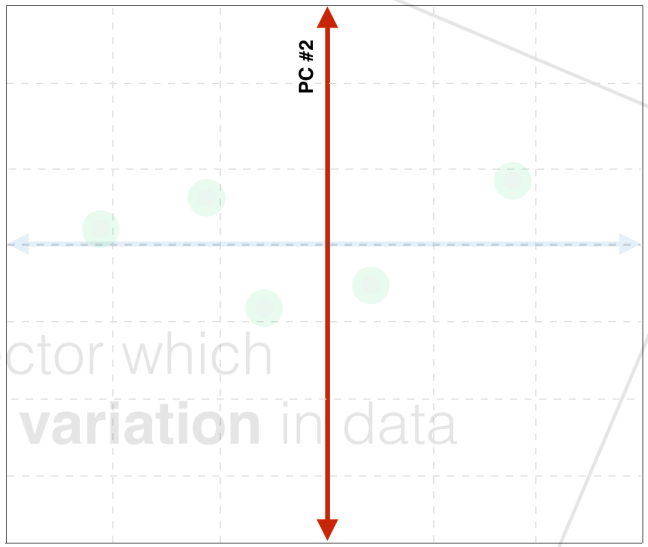


PC #1 is a new vector which spans along **most of the variation** in data

2-D example revisited

These **new axes** are called principle components

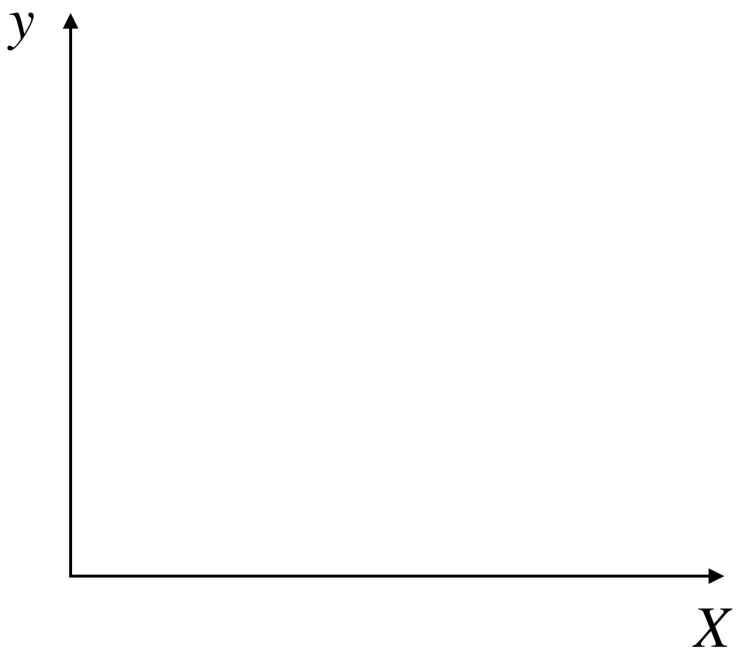


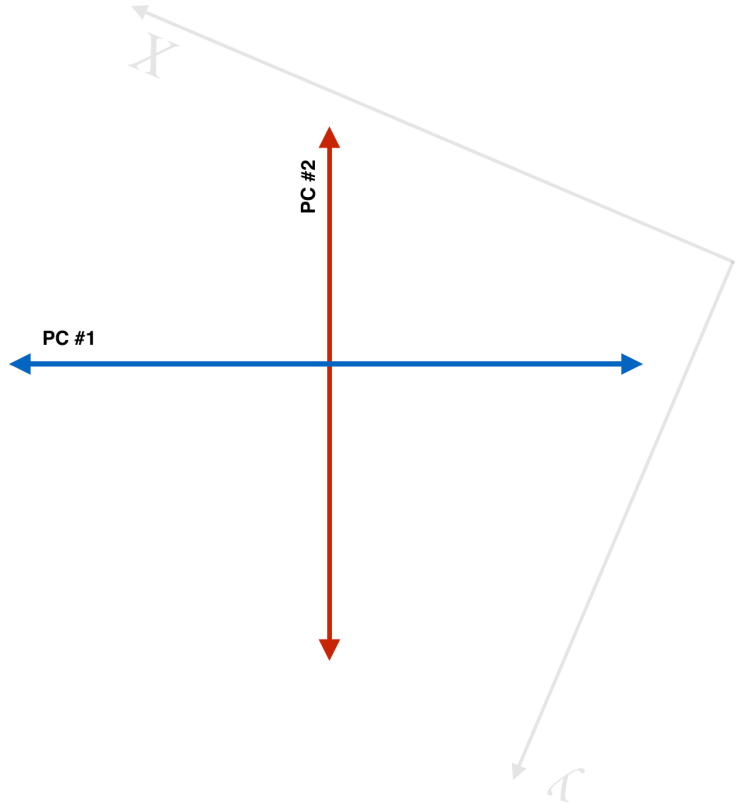


PC #2 is another new vector which spans along the direction of the second most variation

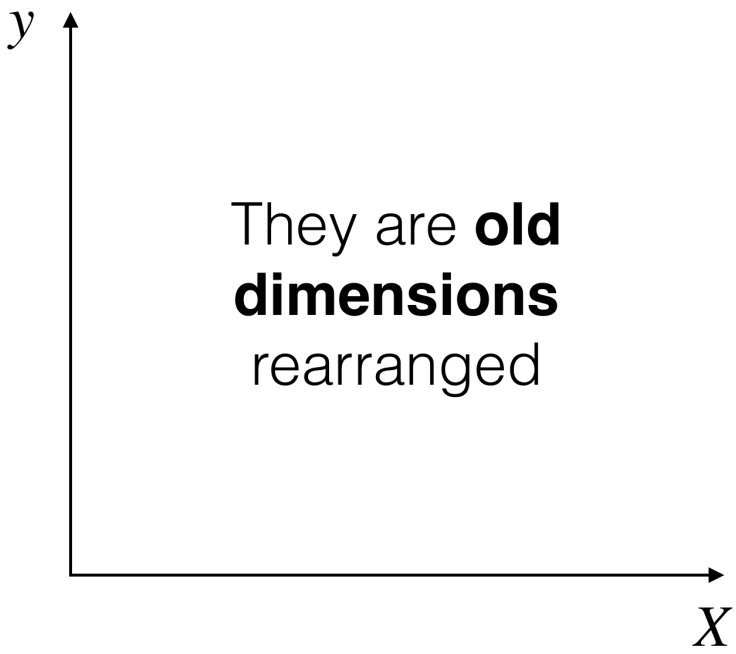
Principal Component Analysis

Principle components are **not additional axes/dimensions**

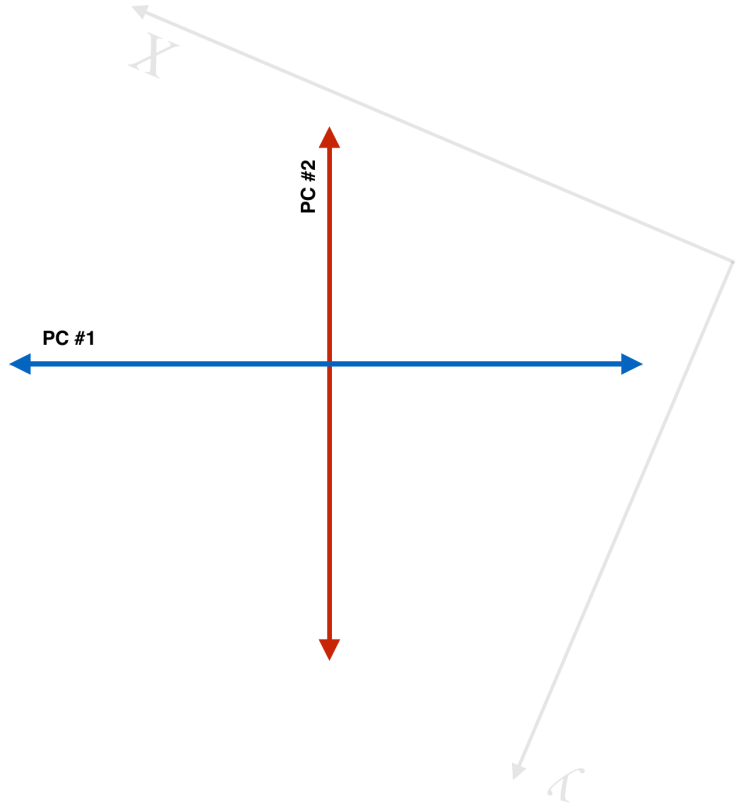




Principle components are **not additional axes/dimensions**

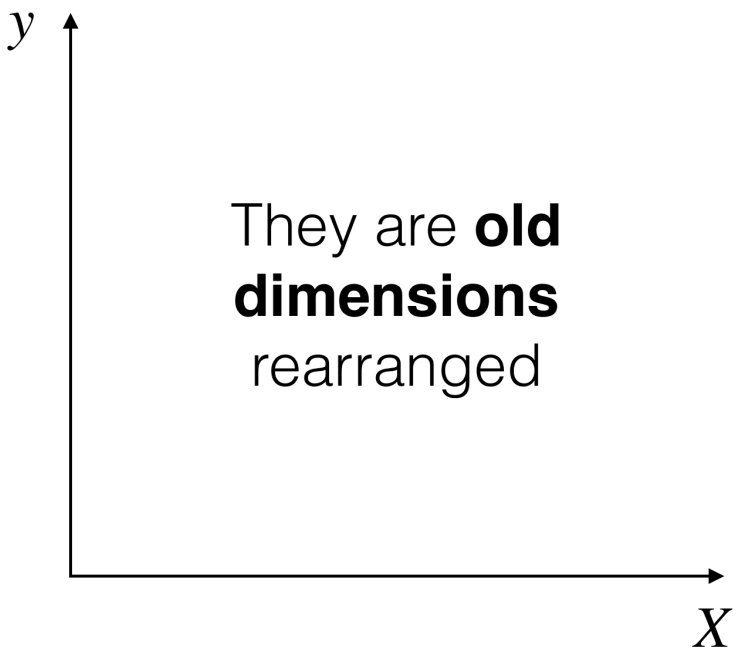


They are **old dimensions** rearranged

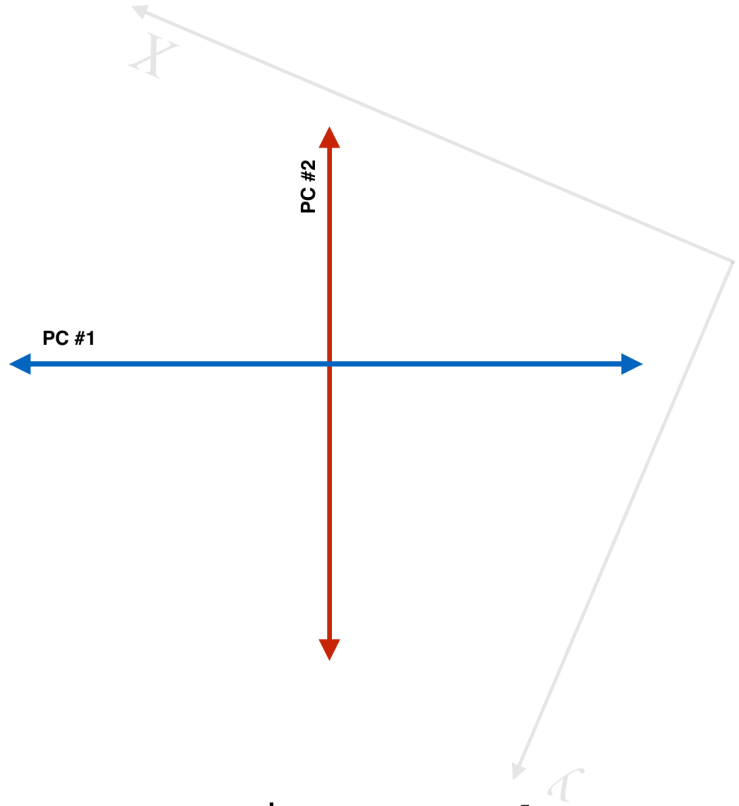


Principal Component Analysis

Principle components are **not additional axes/dimensions**

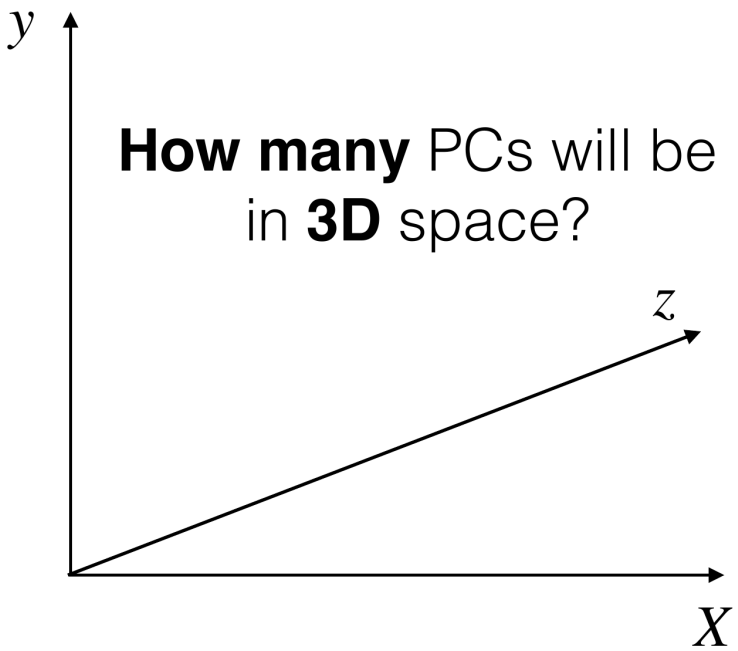


They are **old dimensions** rearranged



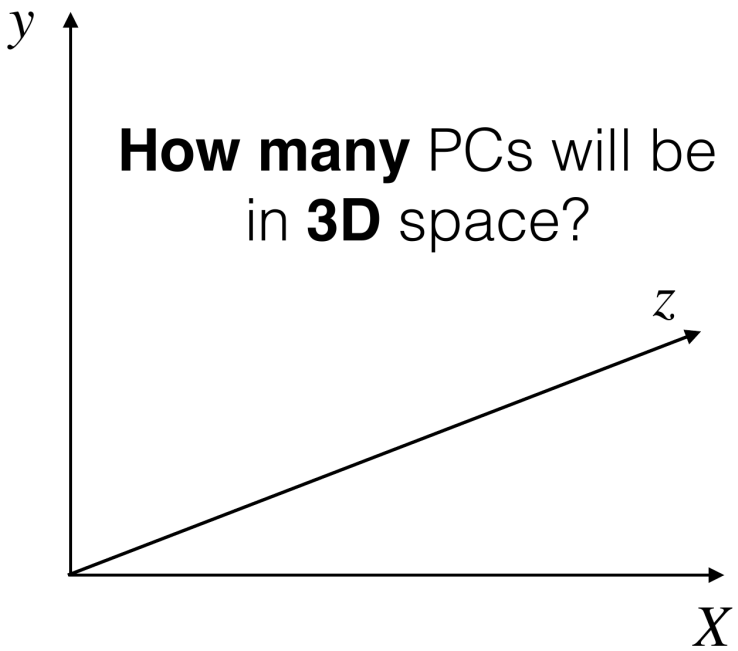
Such that the **first axis** now spans along **most variation**, the **second** the **second most variation** etc.

Principle components are **not additional axes/dimensions**

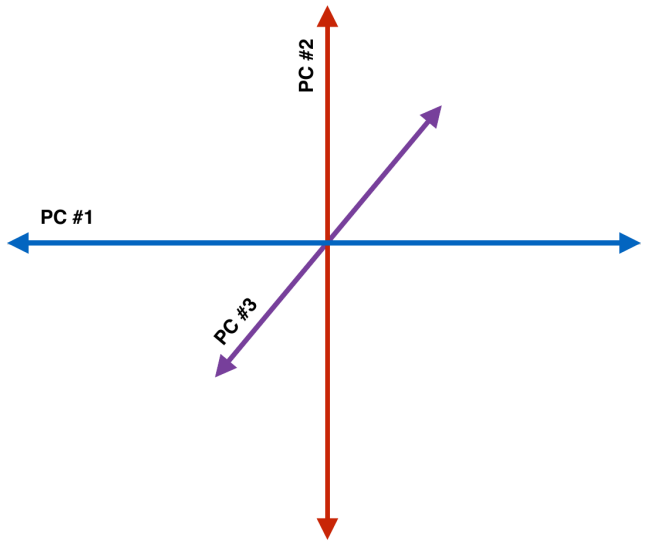


Principal Component Analysis

Principle components are **not additional axes/dimensions**



How many PCs will be in **3D** space?

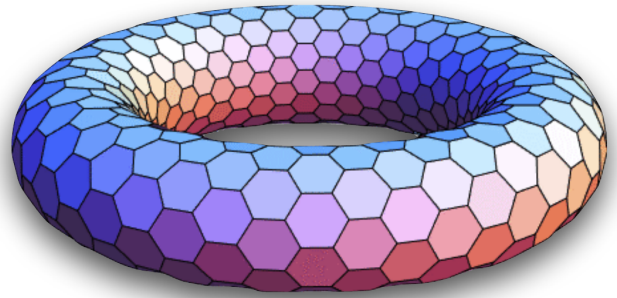


As many as there were **original dimensions**, hence **3** PCs

Principal Component Analysis

Principle components are **not additional axes/dimensions**

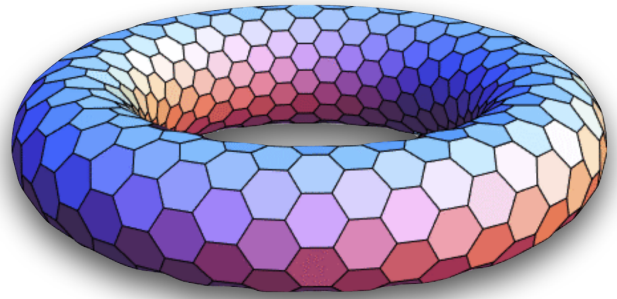
How many PCs will be formed in **200D** space?

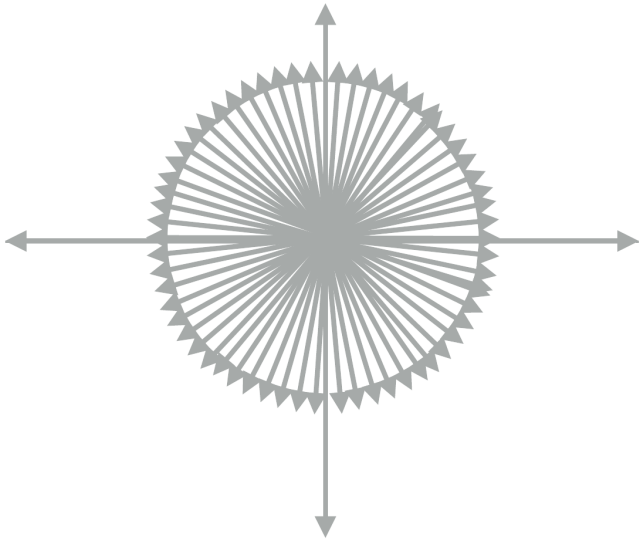


Principal Component Analysis

Principle components are **not additional axes/dimensions**

How many PCs will be formed in **200D** space?



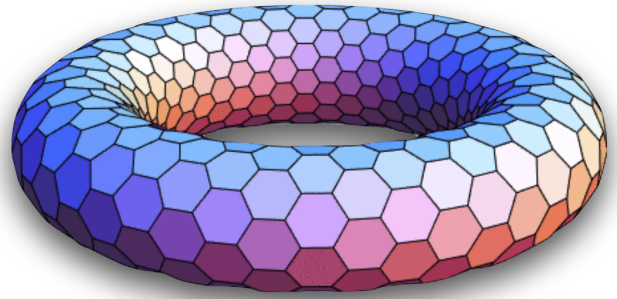


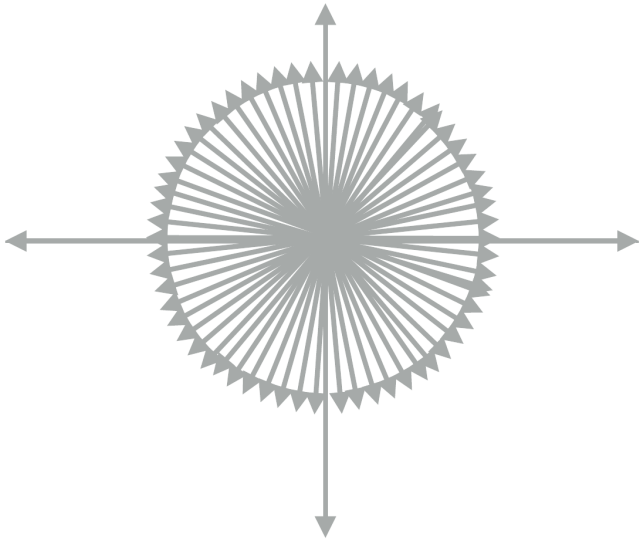
No exceptions, **200 PCs**

Principal Component Analysis

Principle components are **not additional axes/dimensions**

How many PCs will be formed in **200D** space?

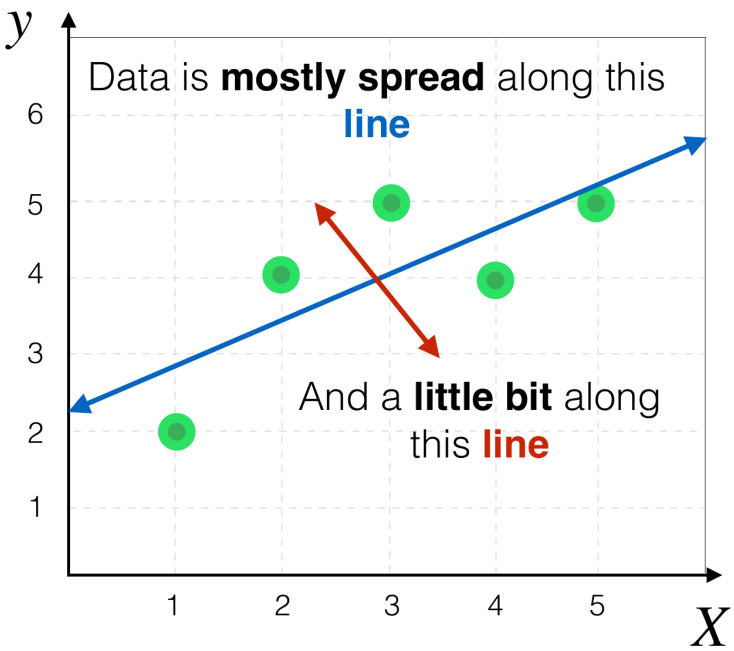


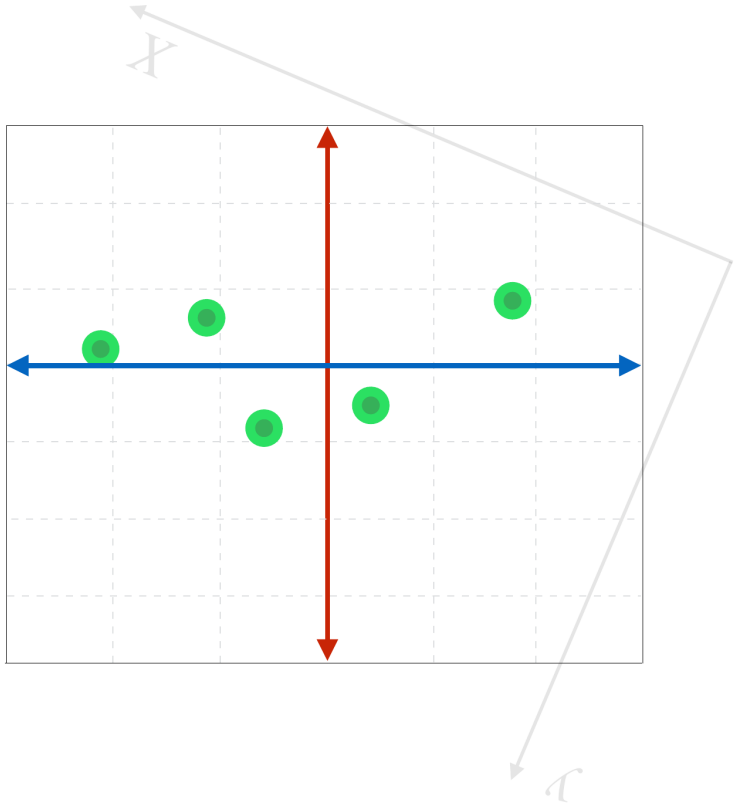


No exceptions, **200 PCs**

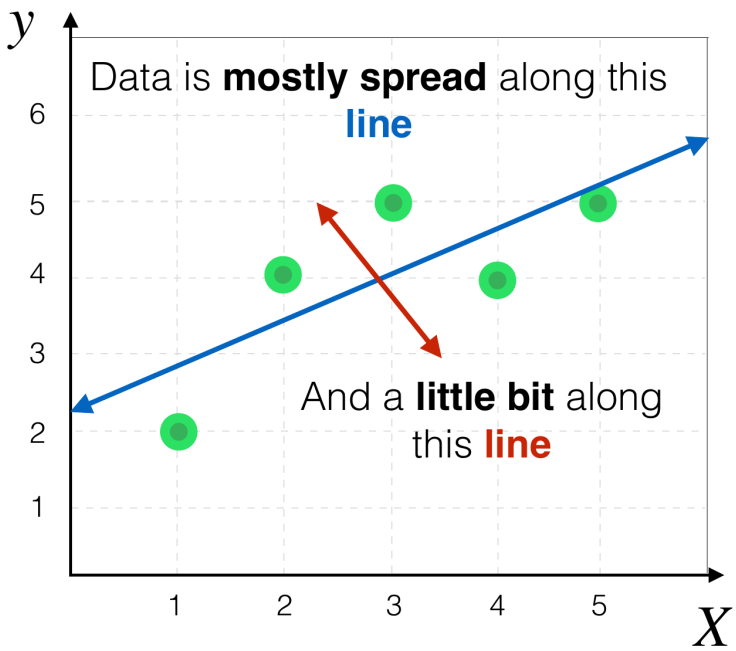
But what is **the benefit** of having PCs?

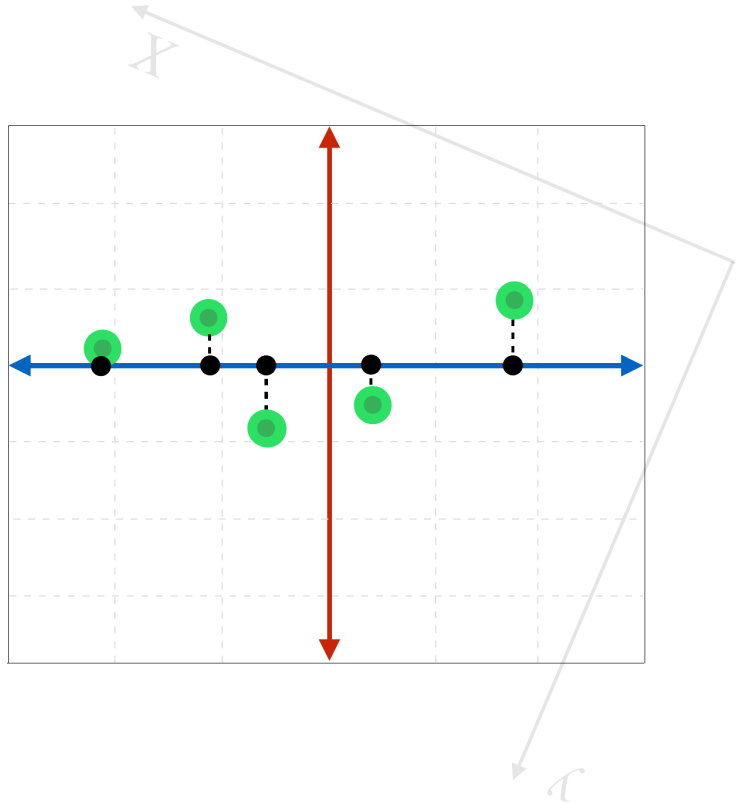
Principal Component Analysis



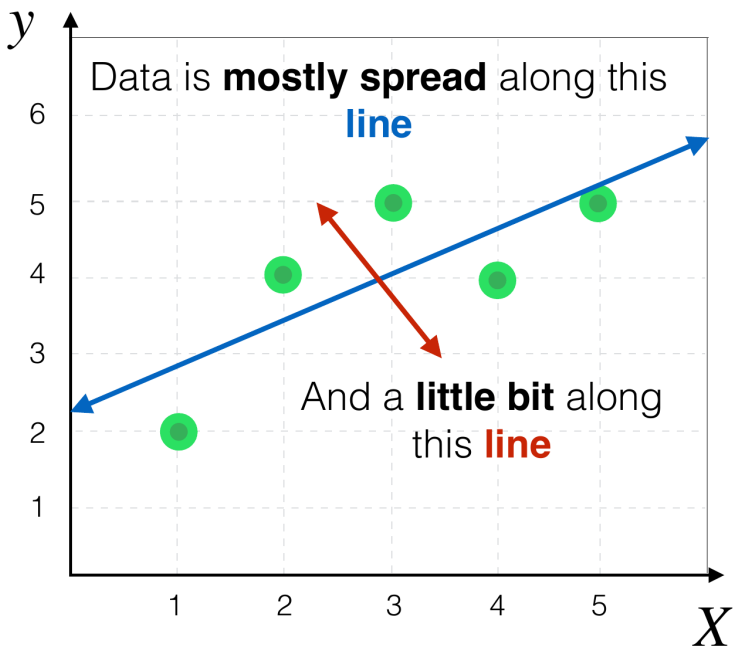


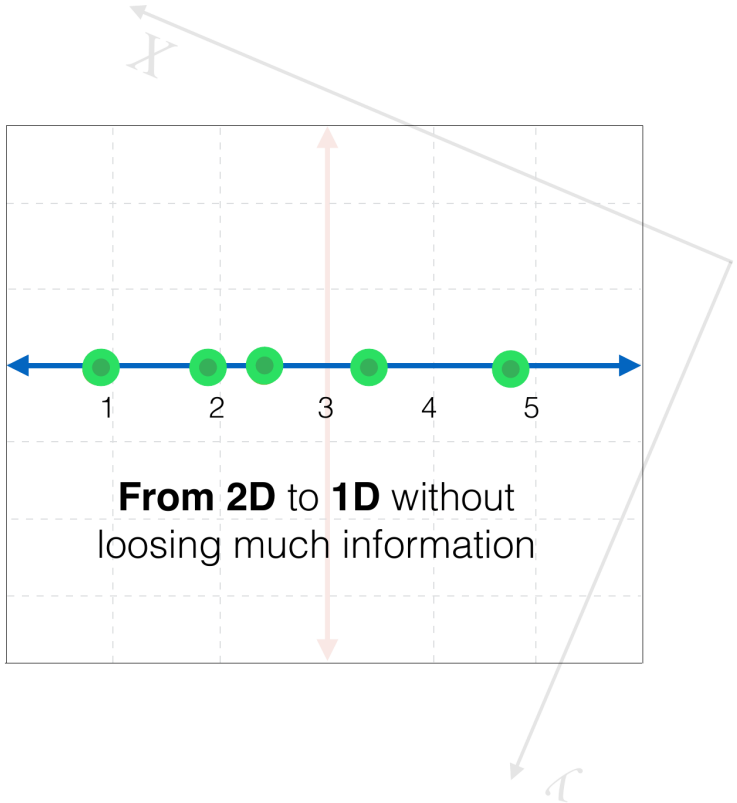
Principal Component Analysis





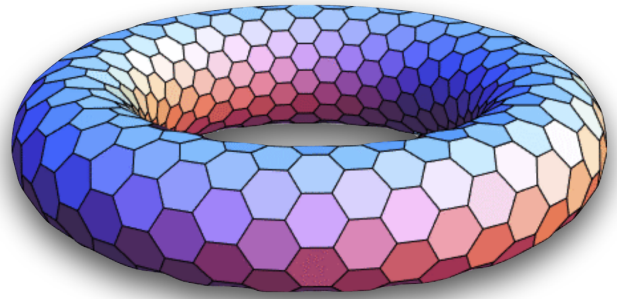
Principal Component Analysis

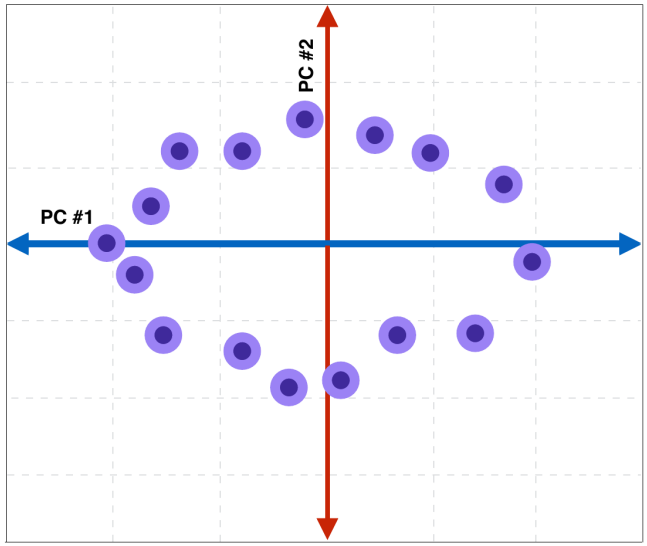




Principle components are **not additional axes/dimensions**

How many PCs will be formed in **200D** space?





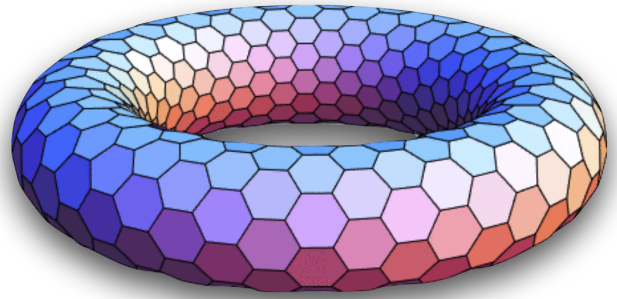
No exceptions, **200 PCs**

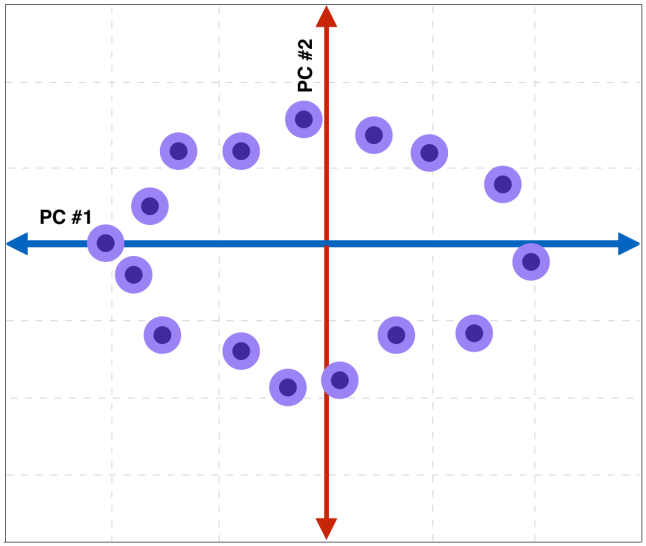
But what is **the benefit** of having PCs?

Principal Component Analysis

Principle components are **not additional axes/dimensions**

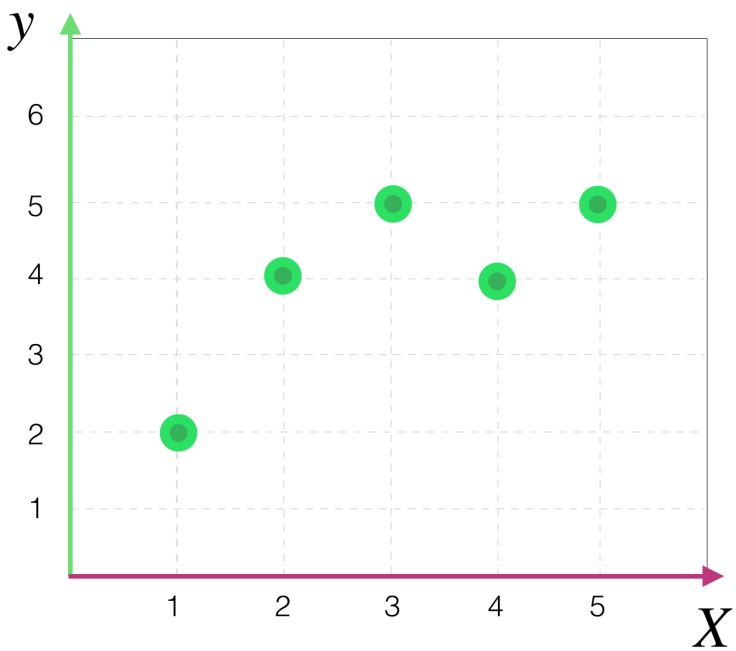
How many PCs will be formed in **200D** space?





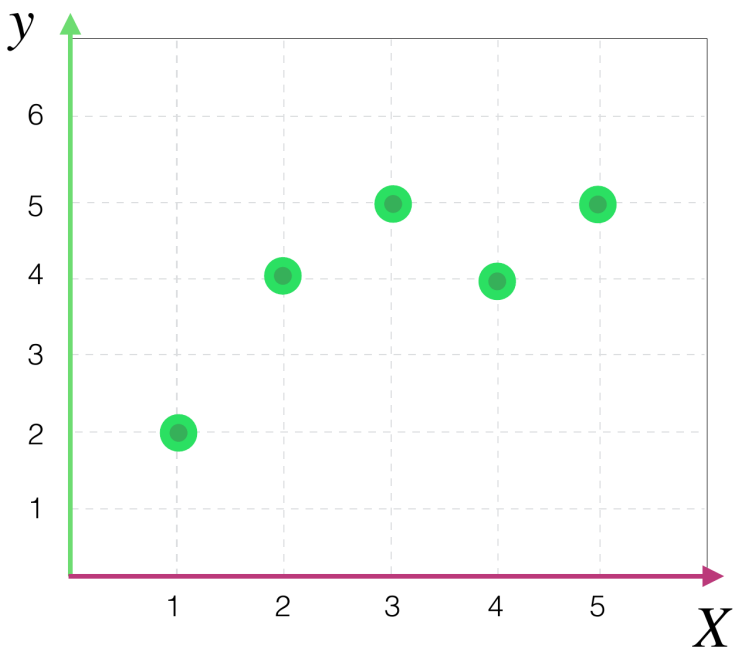
First few PCs would be **enough** to capture important information

Principal Component Analysis



x	y
1	2
2	4
3	5
4	4
5	5

Principal Component Analysis

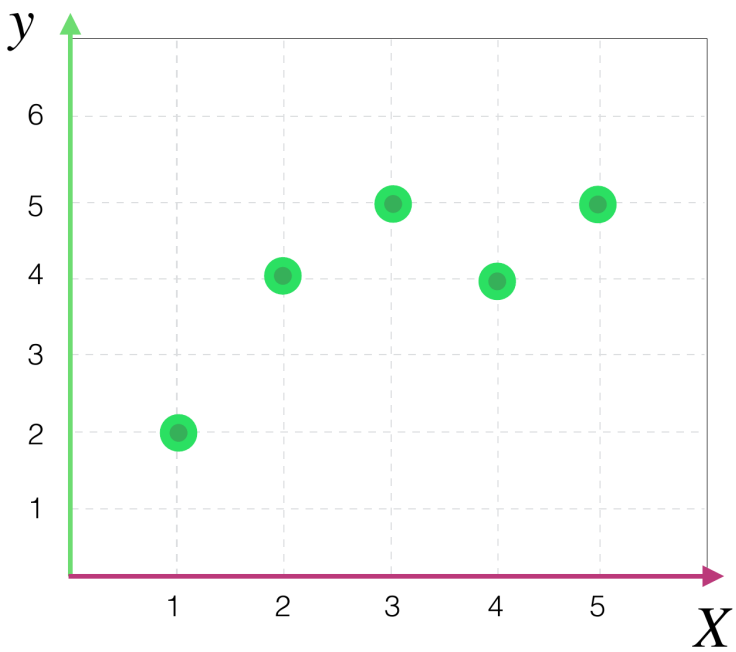


x	y
1	2
2	4
3	5
4	4
5	5

$\bar{x} = 3$

$\bar{y} = 4$

Principal Component Analysis



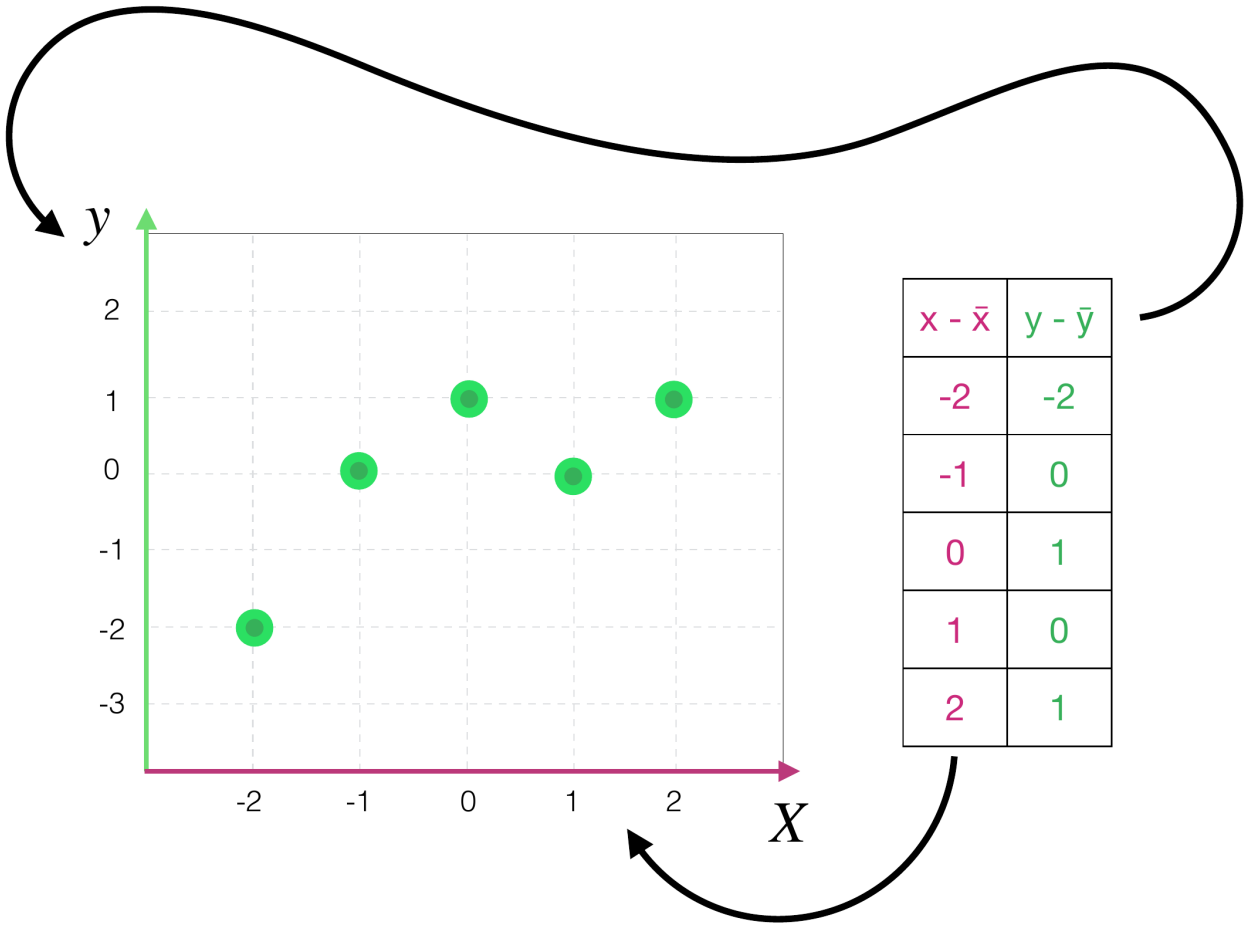
x	y
1	2
2	4
3	5
4	4
5	5

$\bar{x} = 3$

$\bar{y} = 4$

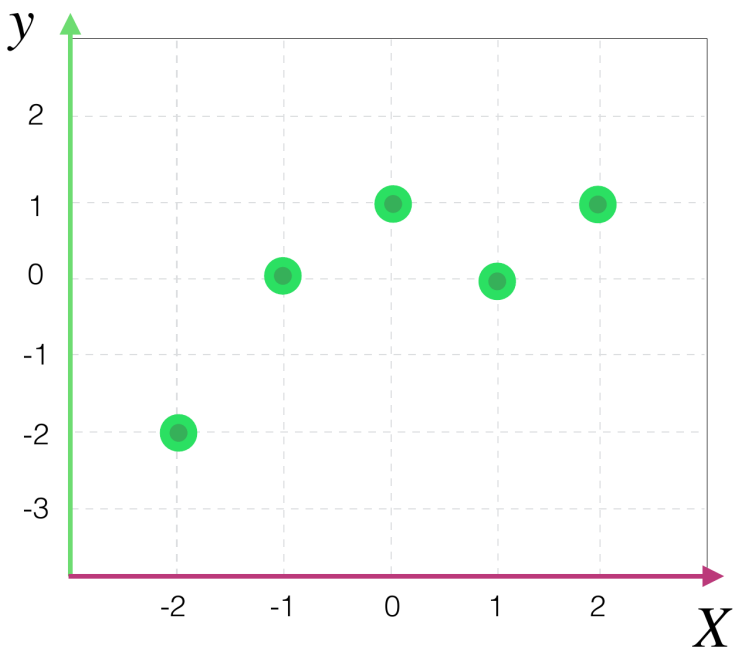
x - $\bar{x}$	y - $\bar{y}$
-2	-2
-1	0
0	1
1	0
2	1

Principal Component Analysis



x - x̄	y - ȳ
-2	-2
-1	0
0	1
1	0
2	1

Principal Component Analysis



Z

$x - \bar{x}$	$y - \bar{y}$
-2	-2
-1	0
0	1
1	0
2	1

Principal Component Analysis

Transpose the matrix of coordinates

Z

-2	-2
-1	0
0	1
1	0
2	1

Principal Component Analysis

Transpose the matrix of coordinates

Z

-2	-2
-1	0
0	1
1	0
2	1

What are the **dimensions** of the transposed matrix?

Principal Component Analysis

Transpose the matrix of coordinates

Z

-2	-2
-1	0
0	1
1	0
2	1

Z^T

Principal Component Analysis

Transpose the matrix of coordinates

Z

-2	-2
-1	0
0	1
1	0
2	1

Z^T

-2	-1	0	1	2
-2	0	1	0	1

Principal Component Analysis

$$Z^{\top} \times Z = S$$

Principal Component Analysis

Z^T

-2	-1	0	1	2
-2	0	1	0	1

$\times$

Z

-2	-2
-1	0
0	1
1	0
2	1

$=$

S

?	?
?	?

Principal Component Analysis

Z^T

-2	-1	0	1	2
-2	0	1	0	1

$\times$

Z

-2	-2
-1	0
0	1
1	0
2	1

$=$

S

10	6
6	6

Principal Component Analysis

Z^T

-2	-1	0	1	2
-2	0	1	0	1

$\times$

Z

-2	-2
-1	0
0	1
1	0
2	1

$=$

S

2.5	1.5
1.5	1.5

$\times 4$

Principal Component Analysis

Z^T

-2	-1	0	1	2
-2	0	1	0	1

$\times$

Z

-2	-2
-1	0
0	1
1	0
2	1

$=$

S

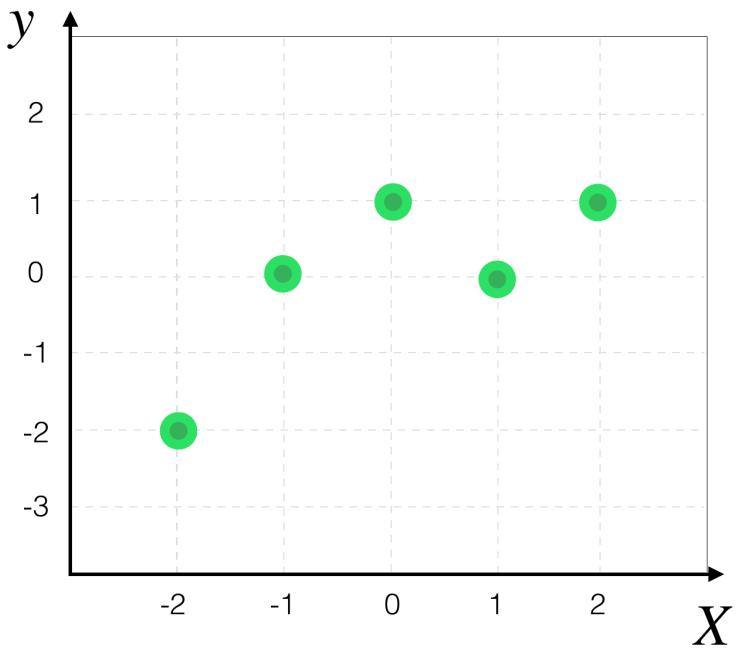
2.5	1.5
1.5	1.5

$\times 4$

Covariance matrix

Principal Component Analysis

How to **interpret** values in **covariance** matrix?



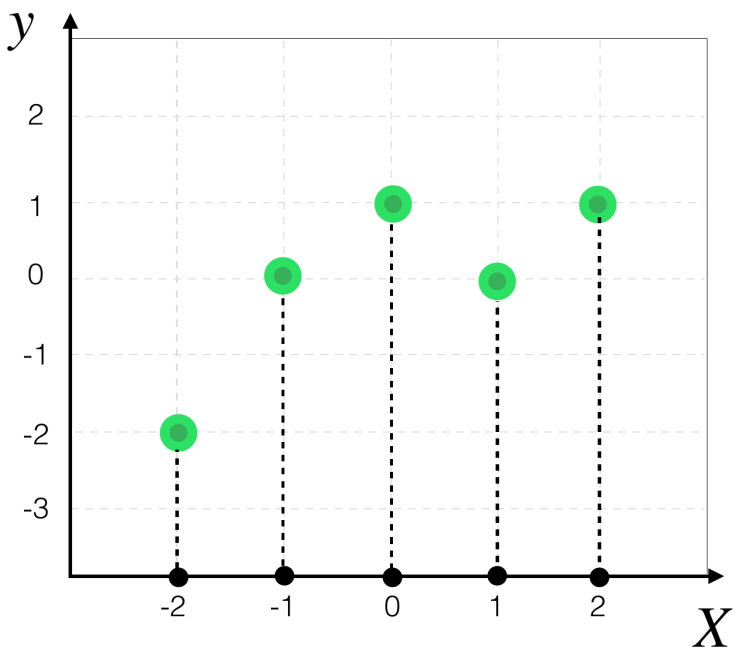
S

2.5	1.5
1.5	1.5

Covariance matrix

Principal Component Analysis

How to **interpret** values in **covariance** matrix?



S

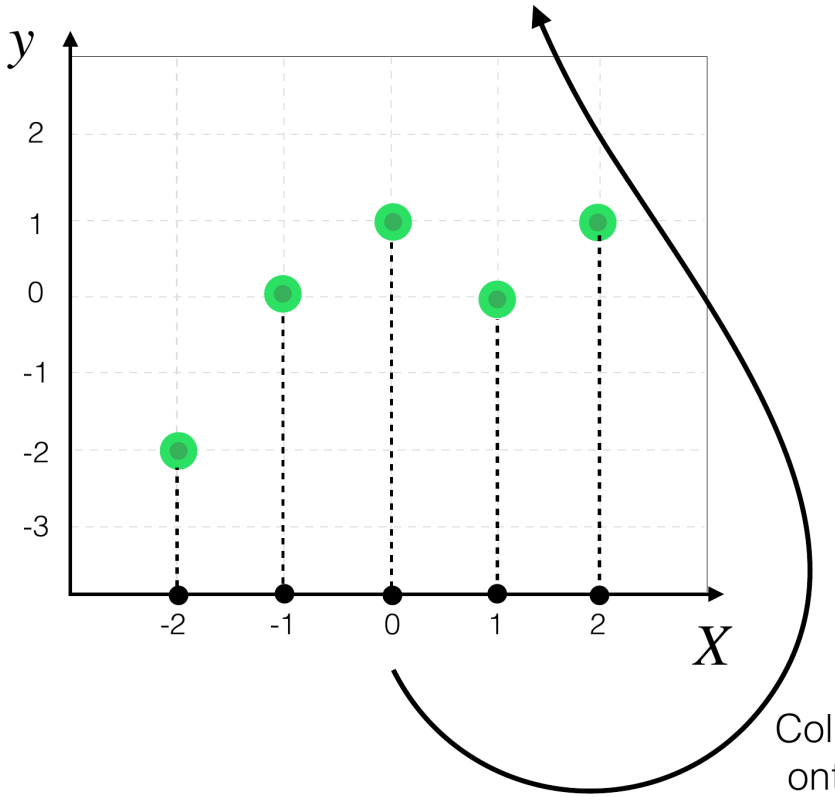
2.5	1.5
1.5	1.5

Covariance matrix

Principal Component Analysis

How to **interpret** values in **covariance** matrix?

[-2, -1, 0, 1, 2]



Collect all projected onto **X axis** values

S

2.5	1.5
1.5	1.5

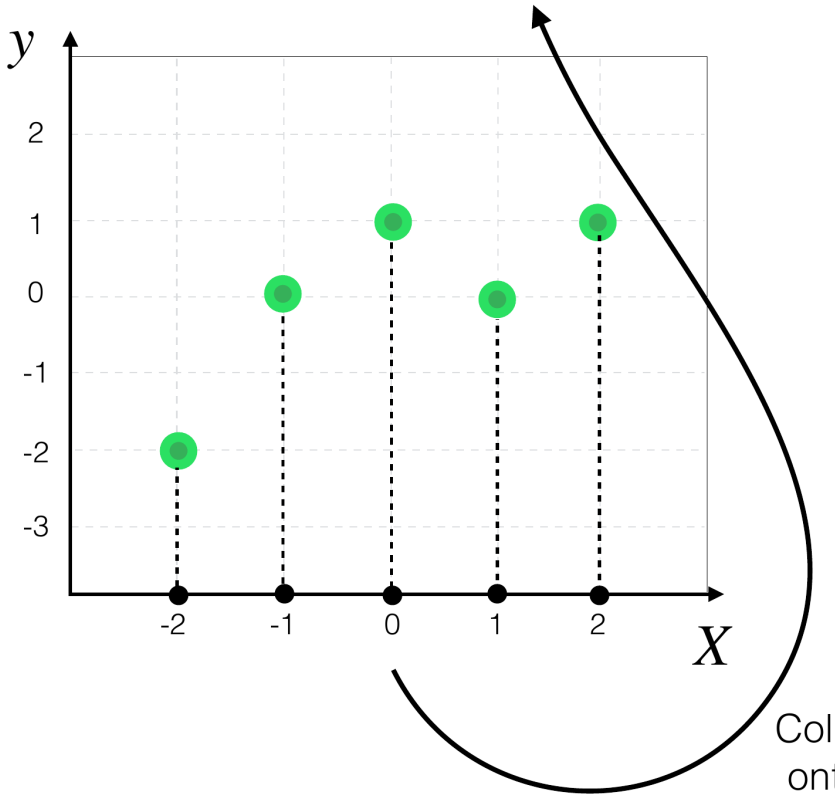
Covariance matrix

Principal Component Analysis

How to **interpret** values in **covariance** matrix?

[-2, -1, 0, 1, 2]

mean([-2, -1, 0, 1, 2]) = ?



Collect all projected onto **X axis** values

S

2.5	1.5
1.5	1.5

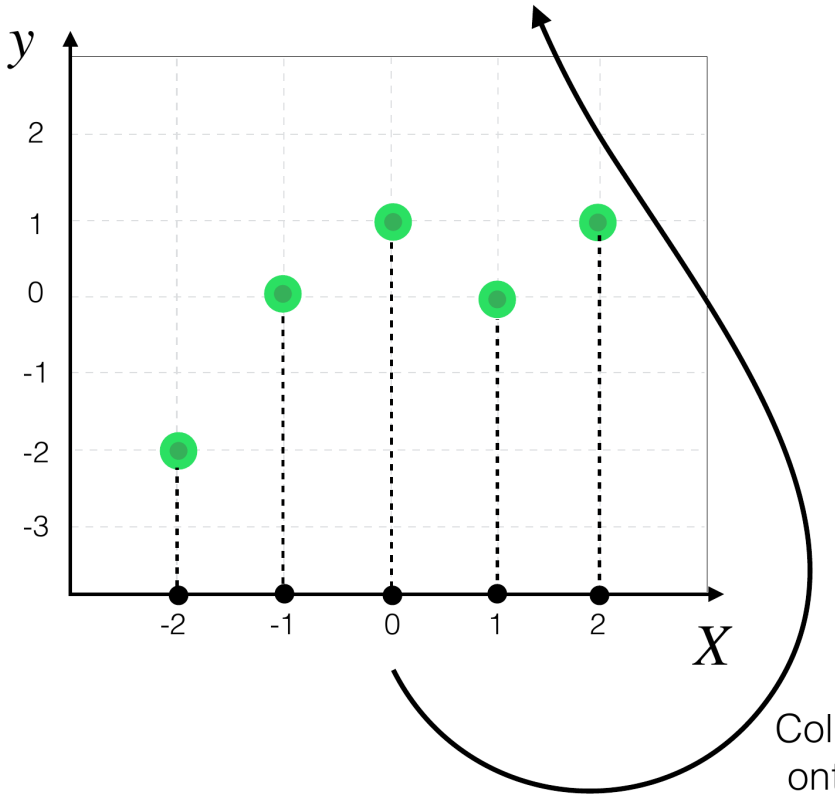
Covariance matrix

Principal Component Analysis

How to **interpret** values in **covariance** matrix?

[-2, -1, 0, 1, 2]

mean([-2, -1, 0, 1, 2]) = 0



S

2.5	1.5
1.5	1.5

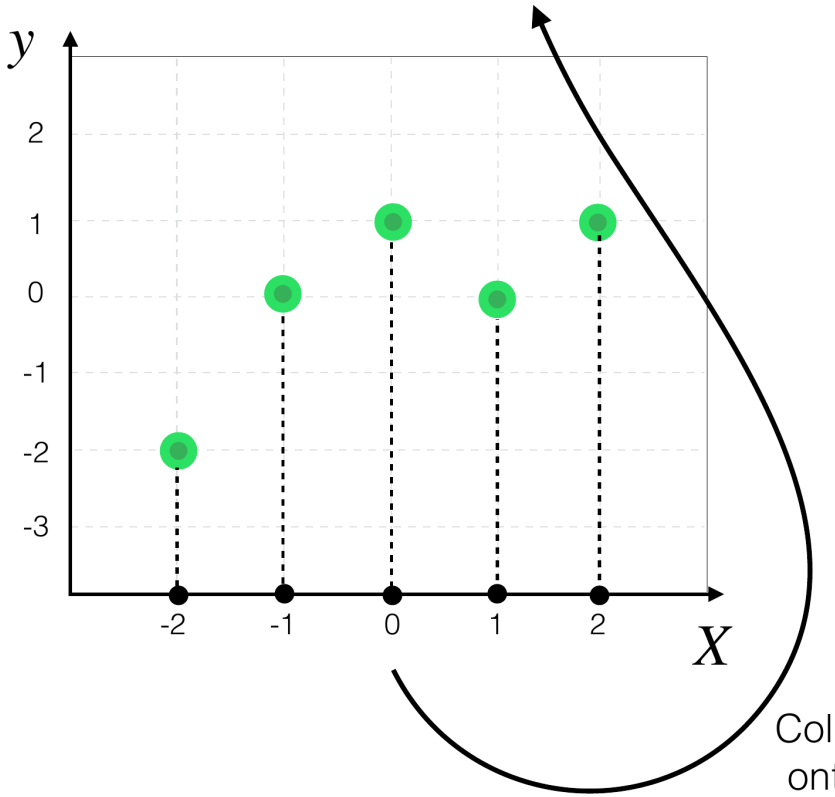
Covariance matrix

Collect all projected onto **X axis** values

Principal Component Analysis

How to **interpret** values in **covariance** matrix?

[-2, -1, 0, 1, 2] $\bar{\mathbf{x}} = 0$



Collect all projected onto **X axis** values

S

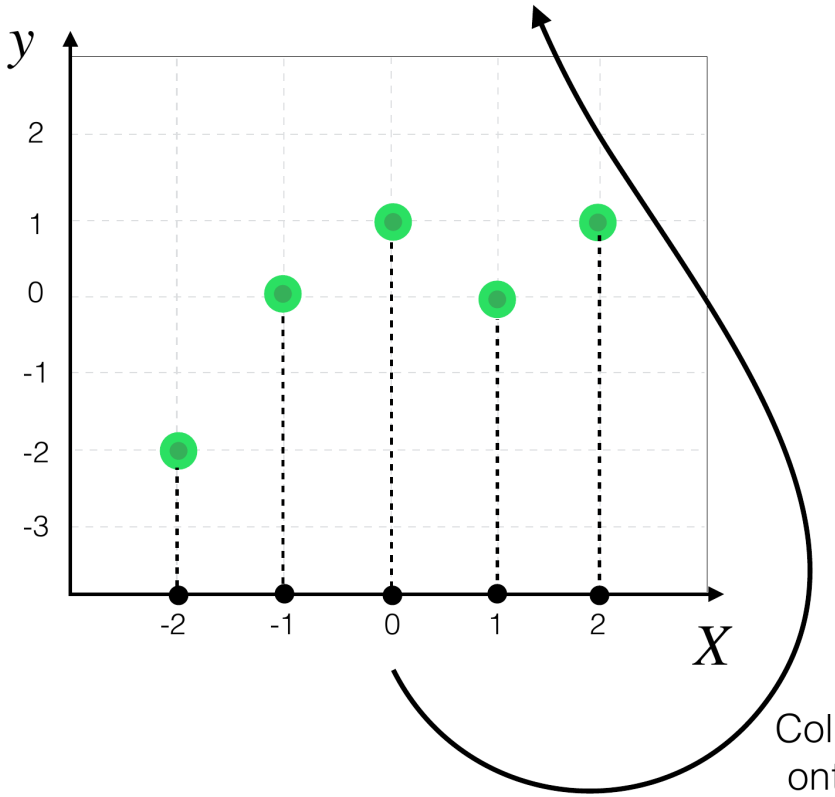
2.5	1.5
1.5	1.5

Covariance matrix

Principal Component Analysis

How to **interpret** values in **covariance** matrix?

[-2, -1, 0, 1, 2] $\quad \bar{\mathbf{x}} = 0 \quad \sigma = \dfrac{\sum (x_i - \bar{x})^2}{n-1}$



Collect all projected onto **X axis** values

S

2.5	1.5
1.5	1.5

Covariance matrix

Principal Component Analysis

How to **interpret** values in **covariance** matrix?

[-2, -1, 0, 1, 2] $\bar{x} = 0$

$$\sigma = \frac{\sum (x_i - \bar{x})^2}{n - 1}$$

S

Covariance matrix

Collect all projected
onto **X axis** values

Principal Component Analysis

Variance $\longrightarrow \sigma = \dfrac{\sum (x_i - \bar{x})^2}{n-1}$

Covariance matrix

Principal Component Analysis

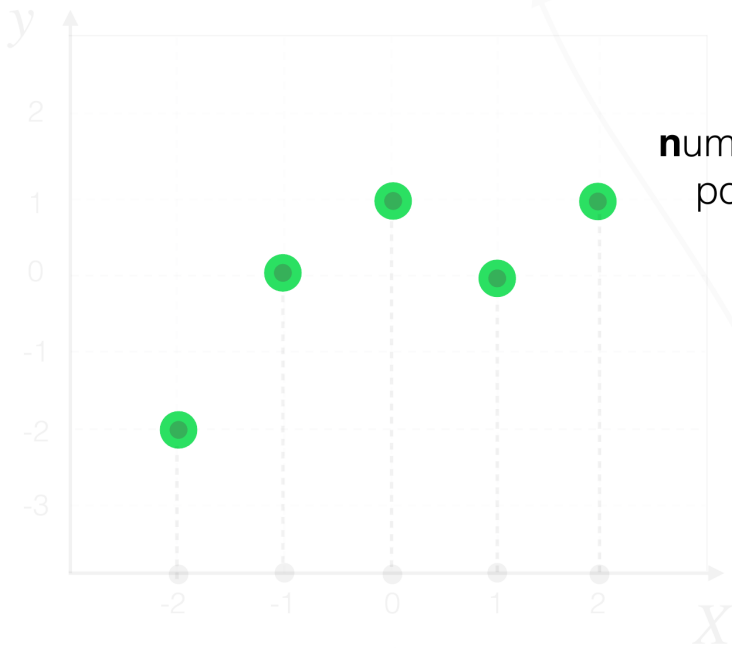
Variance $\longrightarrow \sigma = \dfrac{\sum (x_i - \bar{x})^2}{n - 1}$

number of points

Principal Component Analysis

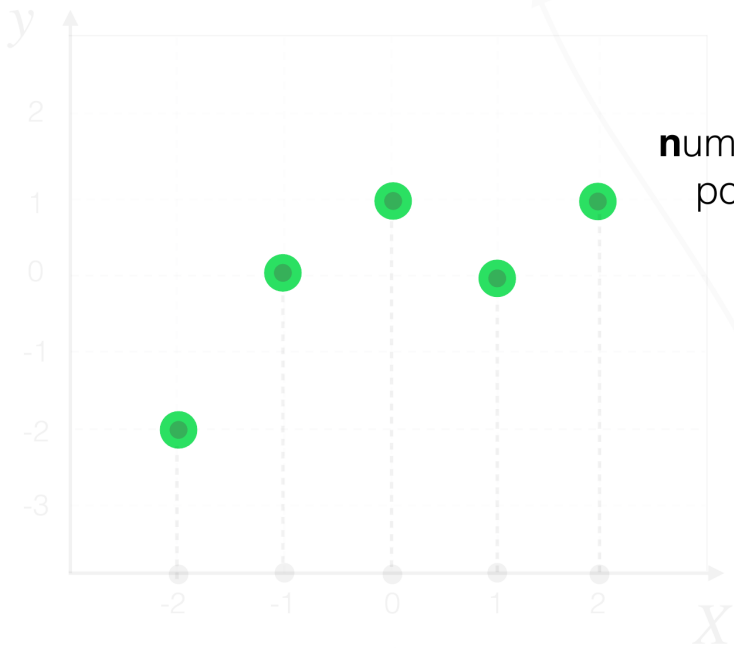
Variance $\longrightarrow \sigma = \dfrac{\sum (x_i - \bar{x})^2}{5 - 1}$

number of points



Principal Component Analysis

Variance $\longrightarrow \sigma = \dfrac{\sum (x_i - \bar{x})^2}{4}$



number of points

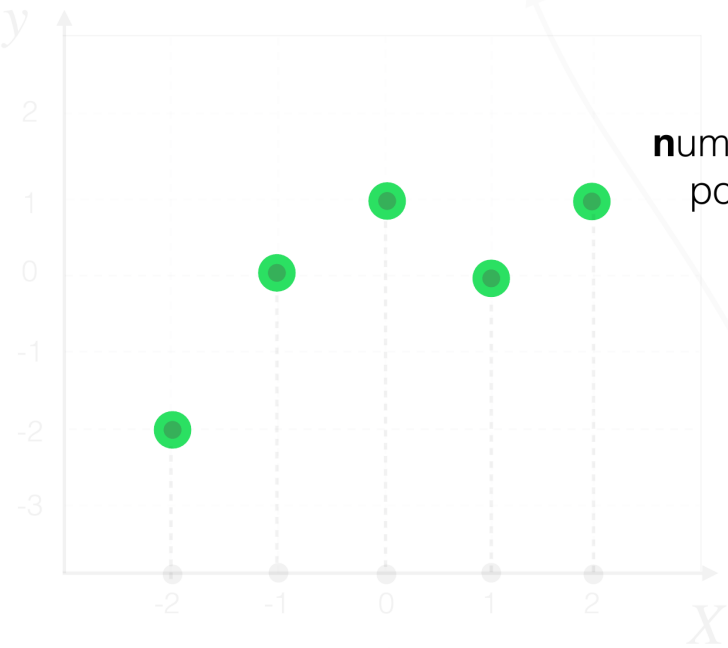
Principal Component Analysis

mean of all points

$$\textbf{Variance} \longrightarrow \sigma = \frac{\sum (x_i - \bar{x})^2}{4}$$

$\bar{\mathbf{x}} = 0$

number of points



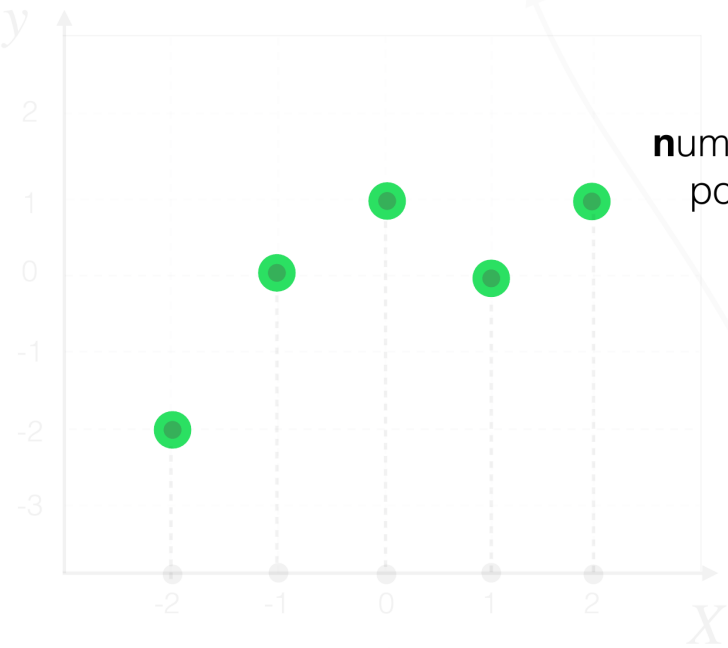
Covariance matrix

Principal Component Analysis

mean of all points

$$\textbf{Variance} \longrightarrow \sigma = \frac{\sum (x_i - 0)^2}{4} \qquad \bar{\mathbf{x}} = 0$$

number of points

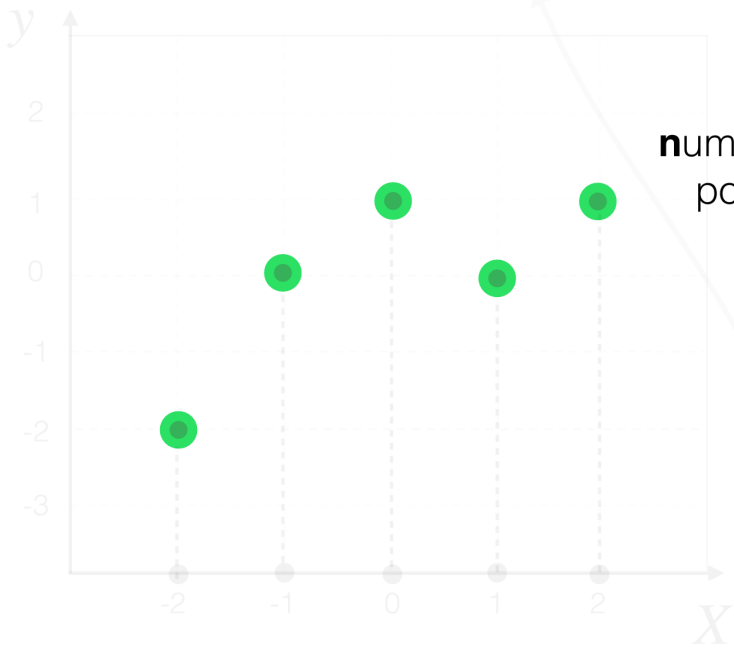


Covariance matrix

Principal Component Analysis

$\bar{\mathbf{x}} = 0$ **m**ean of all points

$$\textbf{Variance} \longrightarrow \sigma = \frac{\sum (x_i)^2}{4}$$



number of points

Covariance matrix

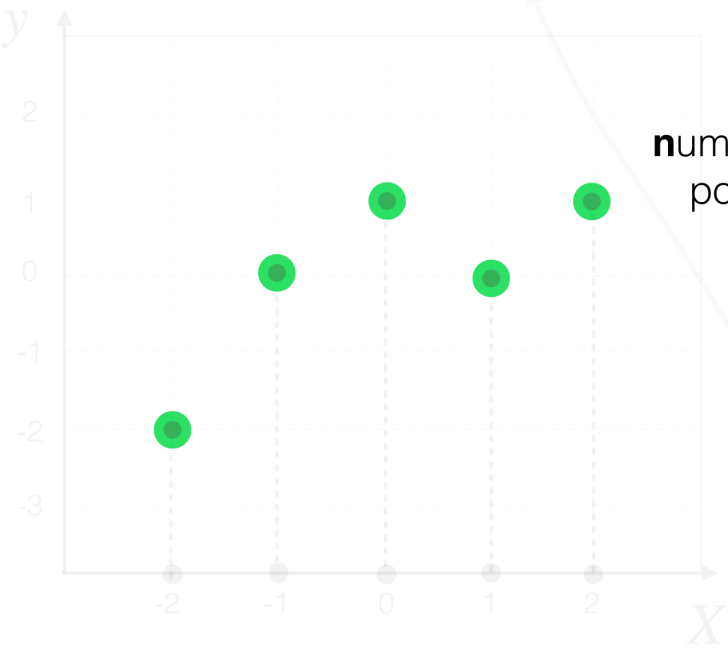
Principal Component Analysis

value of each point

$\bar{\mathbf{x}} = 0$ **m**ean of all points

$$\textbf{Variance} \longrightarrow \sigma = \frac{\sum (x_i)^2}{4}$$

number of points



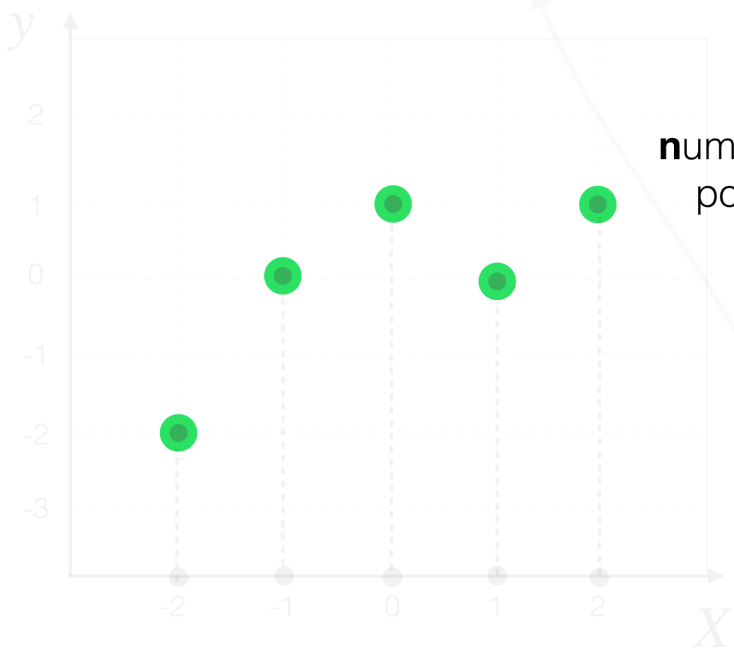
Covariance matrix

Principal Component Analysis

[-2, -1, 0, 1, 2] **value** of each point

$\bar{\mathbf{x}} = 0$ **m**ean of all points

Variance $\longrightarrow \sigma = \dfrac{\sum (x_i)^2}{4}$



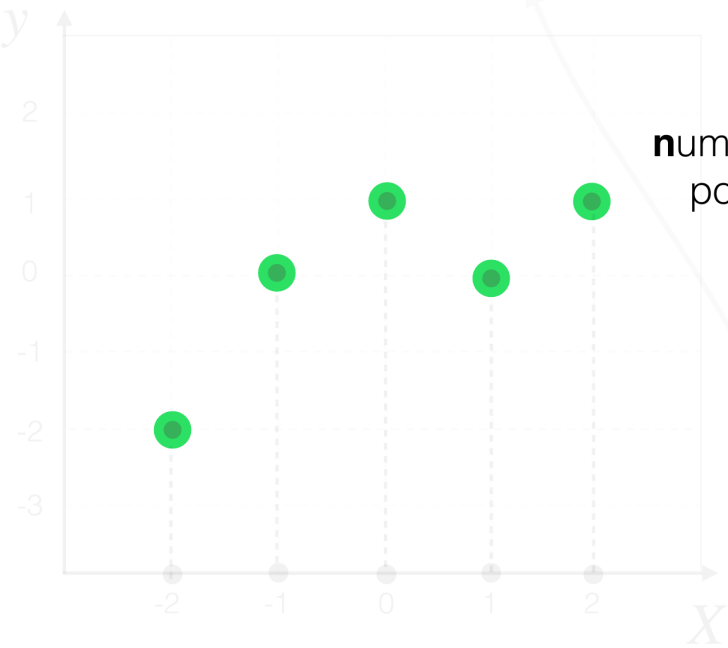
number of points

Covariance matrix

Principal Component Analysis

[-2, -1, 0, 1, 2] **value** of each point → $\bar{\mathbf{x}} = 0$ **m**ean of all points

Variance → $\sigma = \frac{(-2)^2 + (-1)^2 + (0)^2 + (1)^2 + (2)^2}{4}$



number of points

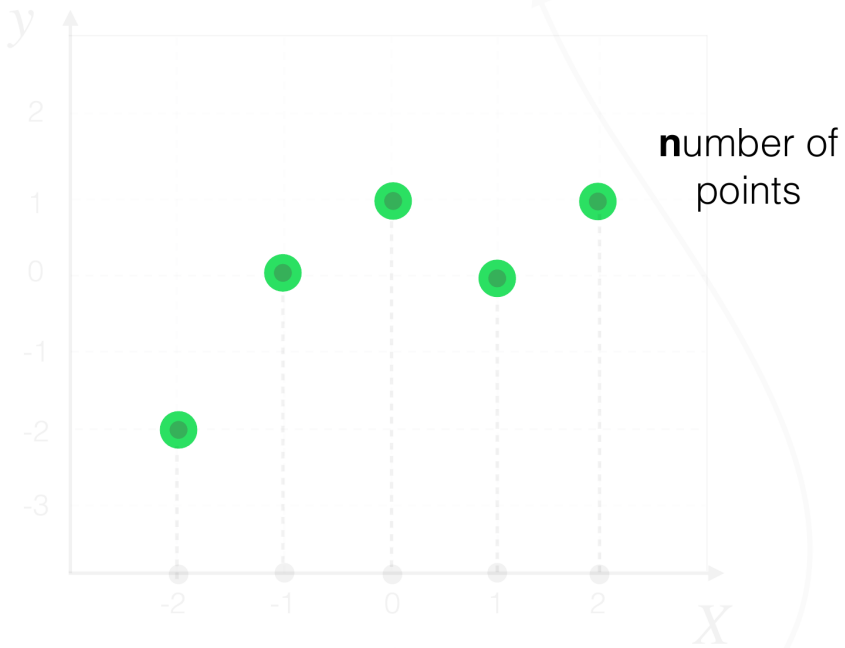
Covariance matrix

Principal Component Analysis

[-2, -1, 0, 1, 2] **value** of each point

$\bar{\mathbf{x}} = 0$ **m**ean of all points

$$\textbf{Variance} \longrightarrow \sigma = \frac{10}{4}$$



number of points

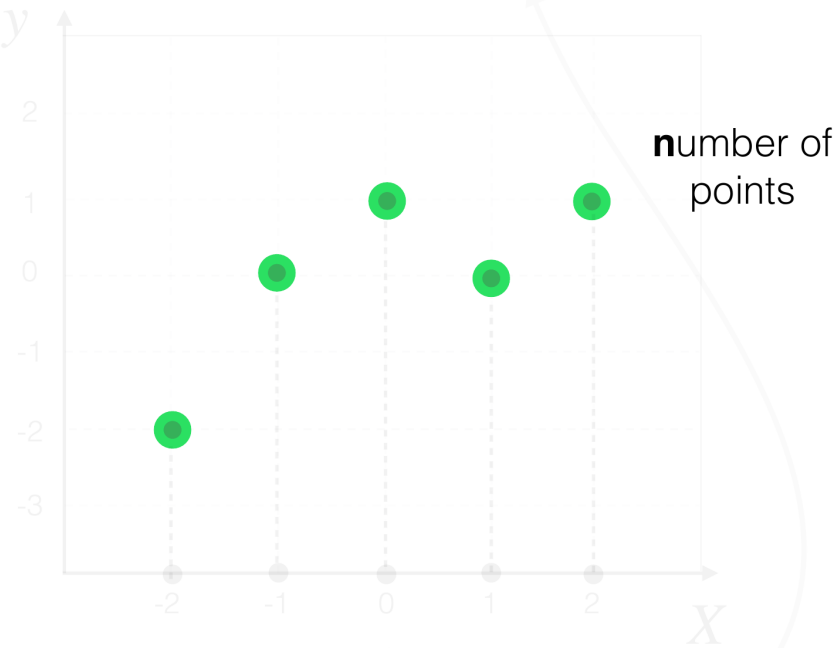
Covariance matrix

Principal Component Analysis

[-2, -1, 0, 1, 2] **value** of each point

$\bar{\mathbf{x}} = 0$ **m**ean of all points

Variance $\longrightarrow \sigma = 2.5$



number of points

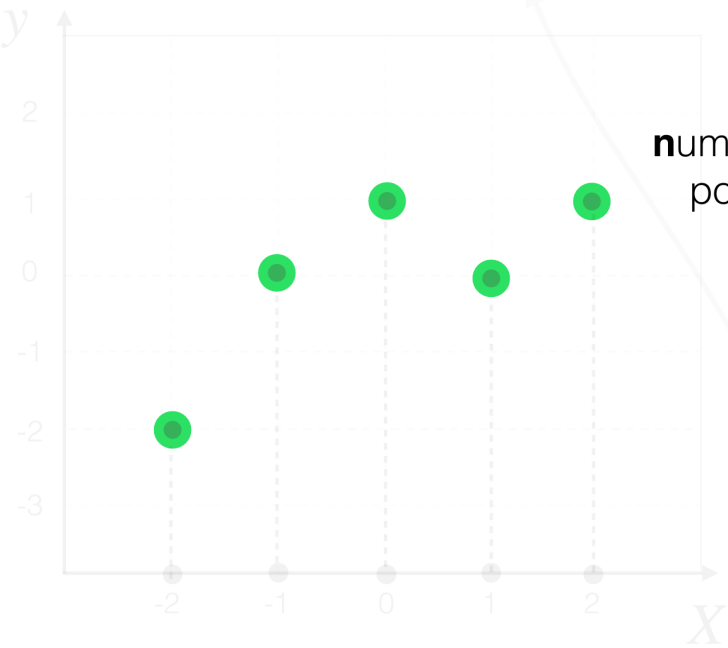
Covariance matrix

Principal Component Analysis

[-2, -1, 0, 1, 2] **value** of each point

$\bar{\mathbf{x}} = 0$ **m**ean of all points

Variance $\longrightarrow \sigma = 2.5$



number of points

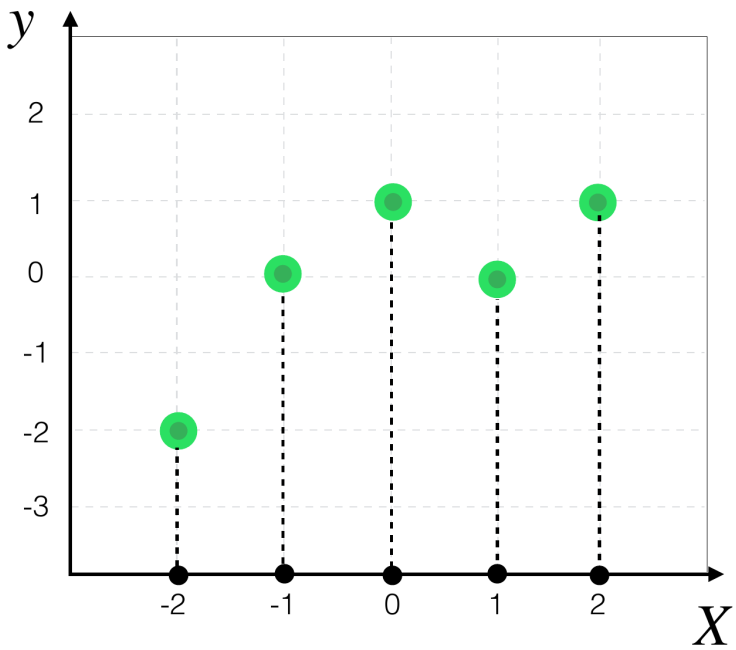
$$\sigma = \frac{\sum (x_i - \bar{x})^2}{n - 1}$$

Variance is an expected value of the squared deviation from the mean

Principal Component Analysis

How to **interpret** values in **covariance** matrix?

[-2, -1, 0, 1, 2] $\bar{\mathbf{x}} = 0$ $\sigma = 2.5$



S

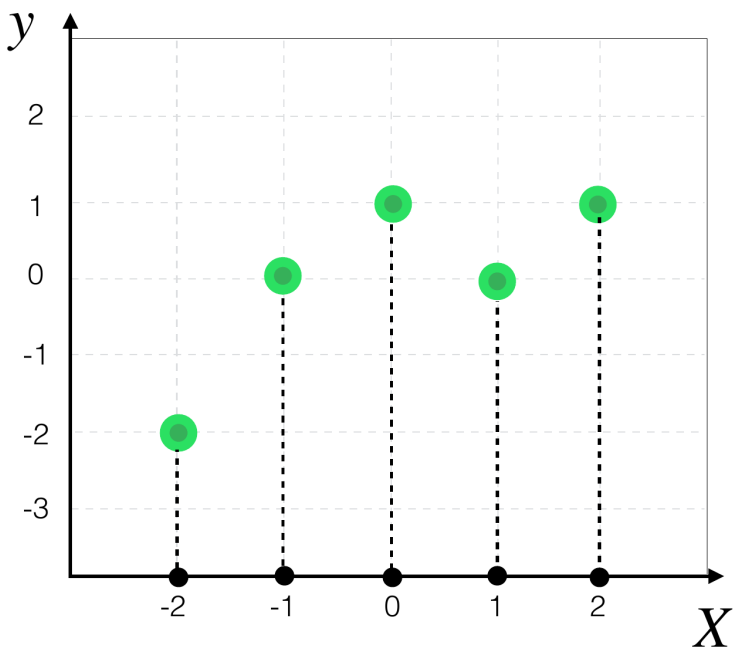
2.5	1.5
1.5	1.5

Covariance matrix

Principal Component Analysis

How to **interpret** values in **covariance** matrix?

[-2, -1, 0, 1, 2] $\bar{\mathbf{x}} = 0$ $\sigma = 2.5$



Variance along **first** axis

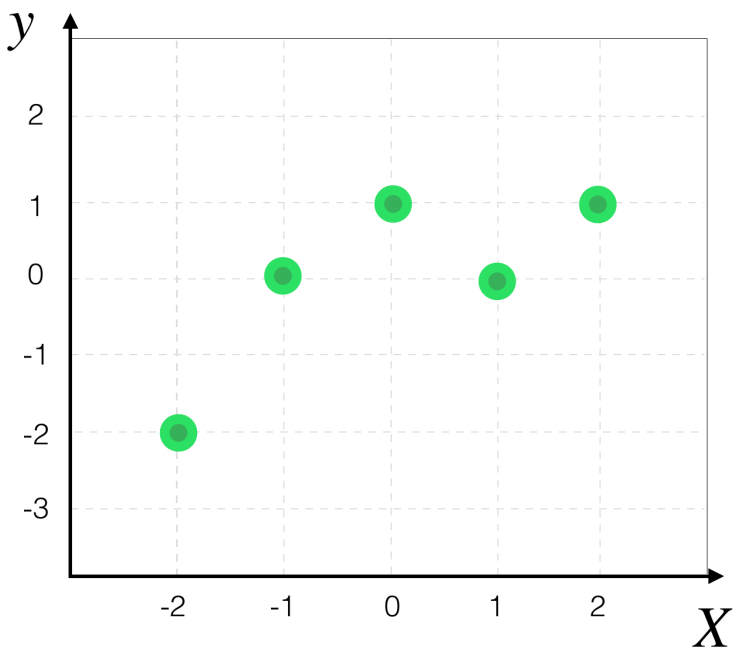
S

2.5	1.5
1.5	1.5

Covariance matrix

Principal Component Analysis

How to **interpret** values in **covariance** matrix?



Variance along **first** axis

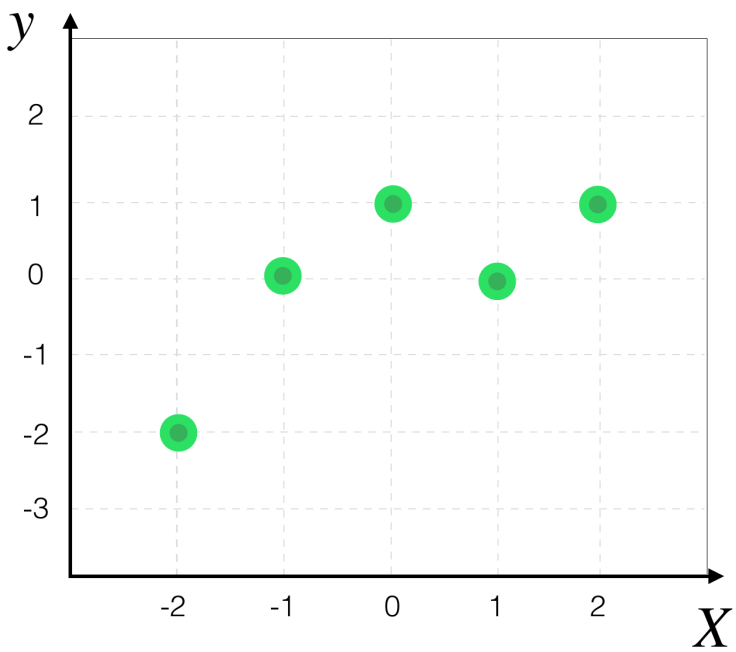
S

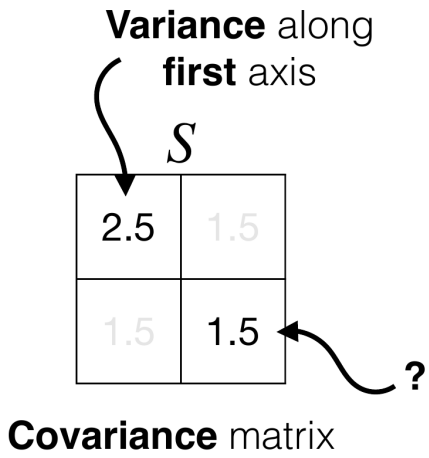
2.5	1.5
1.5	1.5

Covariance matrix

Principal Component Analysis

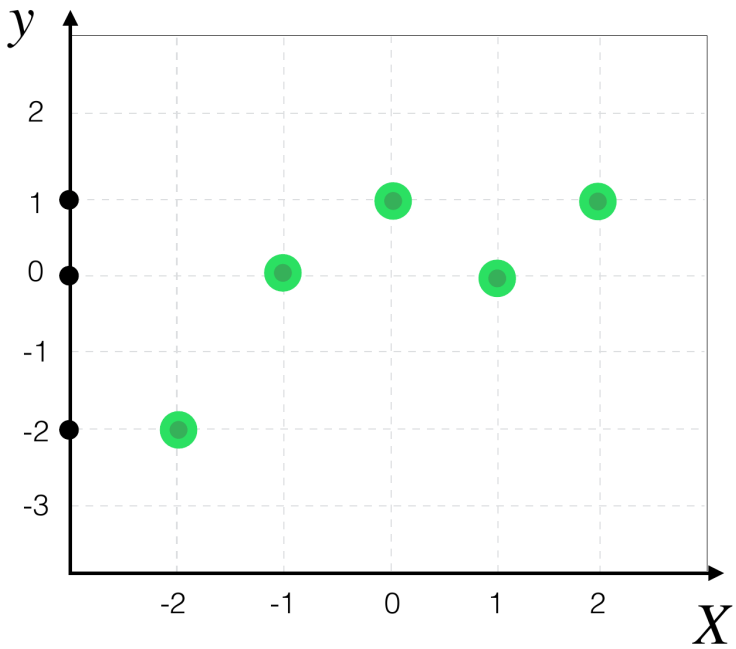
How to **interpret** values in **covariance** matrix?





Principal Component Analysis

How to **interpret** values in **covariance** matrix?



Variance along **first** axis

S

2.5	1.5
1.5	1.5

Variance along **second** axis

Covariance matrix

Principal Component Analysis

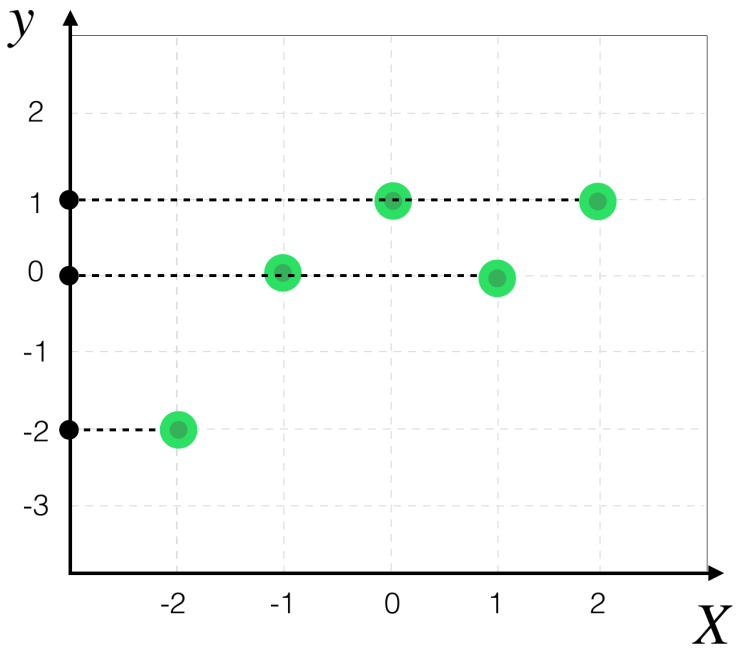
How to **interpret** values in **covariance** matrix?

value of each point

number of points

mean of all points

$$\sigma = \frac{\sum (y_i - \bar{y})^2}{n - 1}$$



Variance along **first** axis

S

2.5	1.5
1.5	1.5

Variance along **second** axis

Covariance matrix

Principal Component Analysis

How to **interpret** values in **covariance** matrix?

value of each point

[-2, 0, 0, 1, 1]

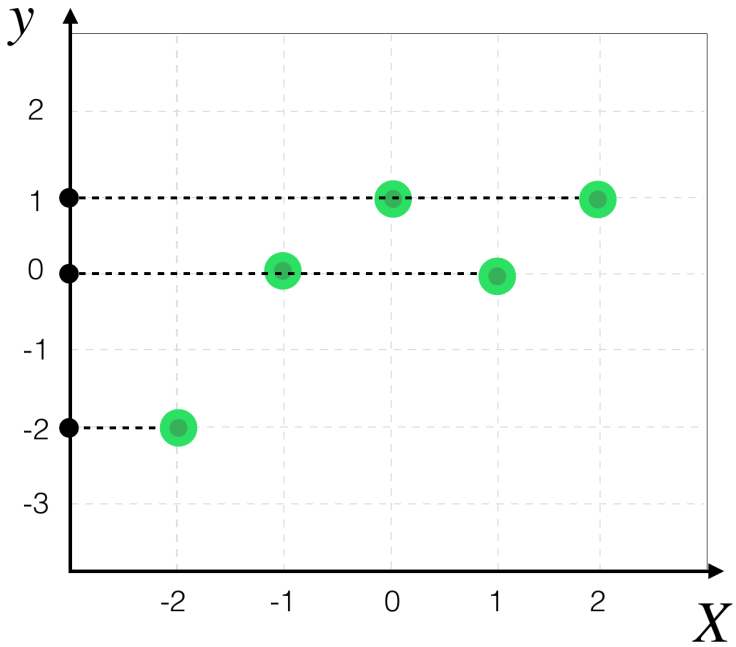
number of points

5

mean of all points

$\bar{\mathbf{y}} = 0$

$$\sigma = \frac{\sum (y_i - \bar{y})^2}{n - 1}$$



Variance along **first** axis

S

2.5	1.5
1.5	1.5

Variance along **second** axis

Covariance matrix

Principal Component Analysis

How to **interpret** values in **covariance** matrix?

value of each point

[-2, 0, 0, 1, 1]

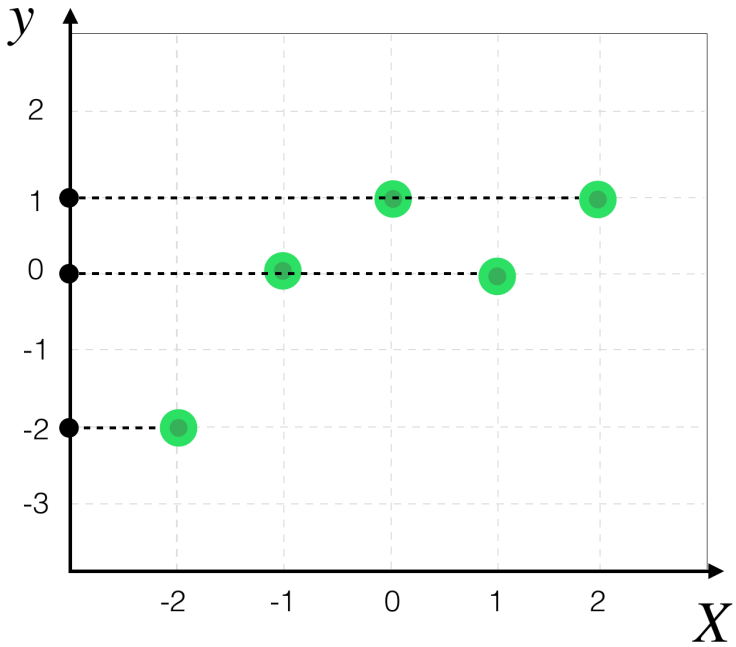
number of points

5

mean of all points

$\bar{\mathbf{y}} = 0$

$$\sigma = \frac{6}{4}$$



Variance along **first** axis

S

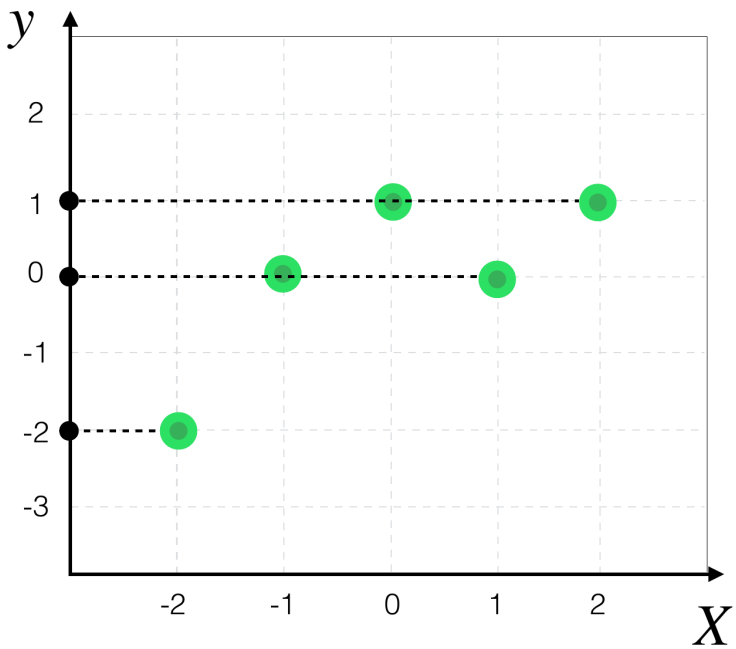
2.5	1.5
1.5	1.5

Variance along **second** axis

Covariance matrix

Principal Component Analysis

How to **interpret** values in **covariance** matrix?



Variance along **first** axis

S

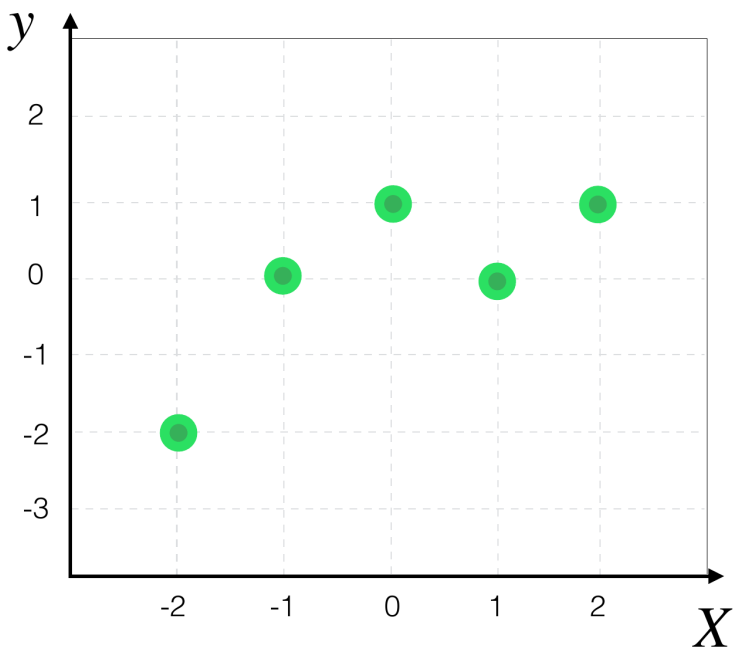
2.5	1.5
1.5	1.5

Variance along **second** axis

Covariance matrix

Principal Component Analysis

How to **interpret** values in **covariance** matrix?



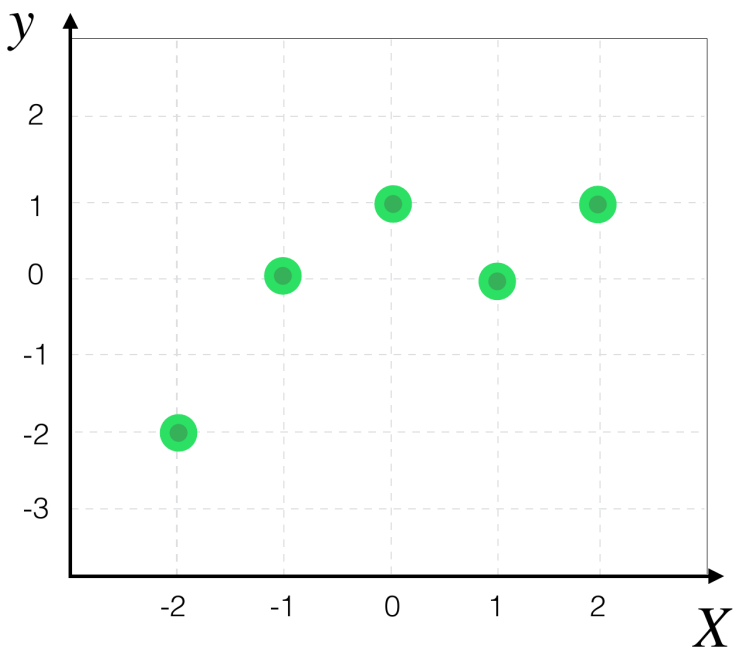
S

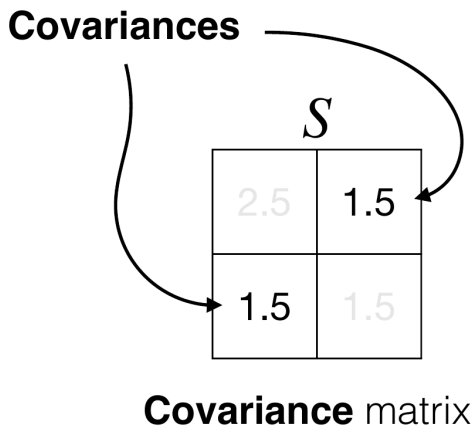
2.5	1.5
1.5	1.5

Covariance matrix

Principal Component Analysis

How to **interpret** values in **covariance** matrix?

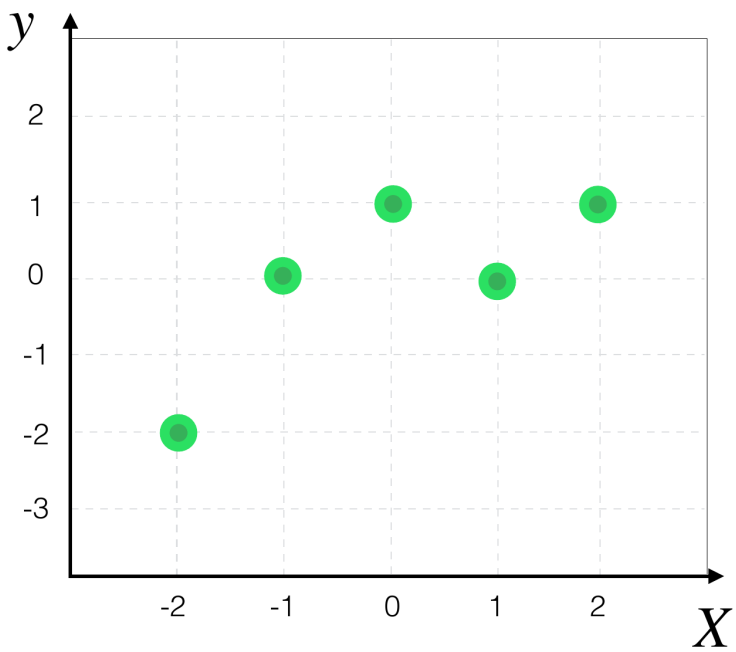


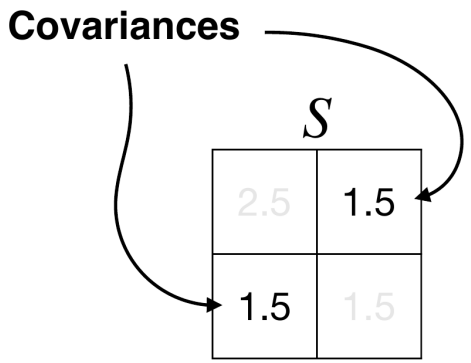


Covariance matrix

Principal Component Analysis

How to **interpret** values in **covariance** matrix?

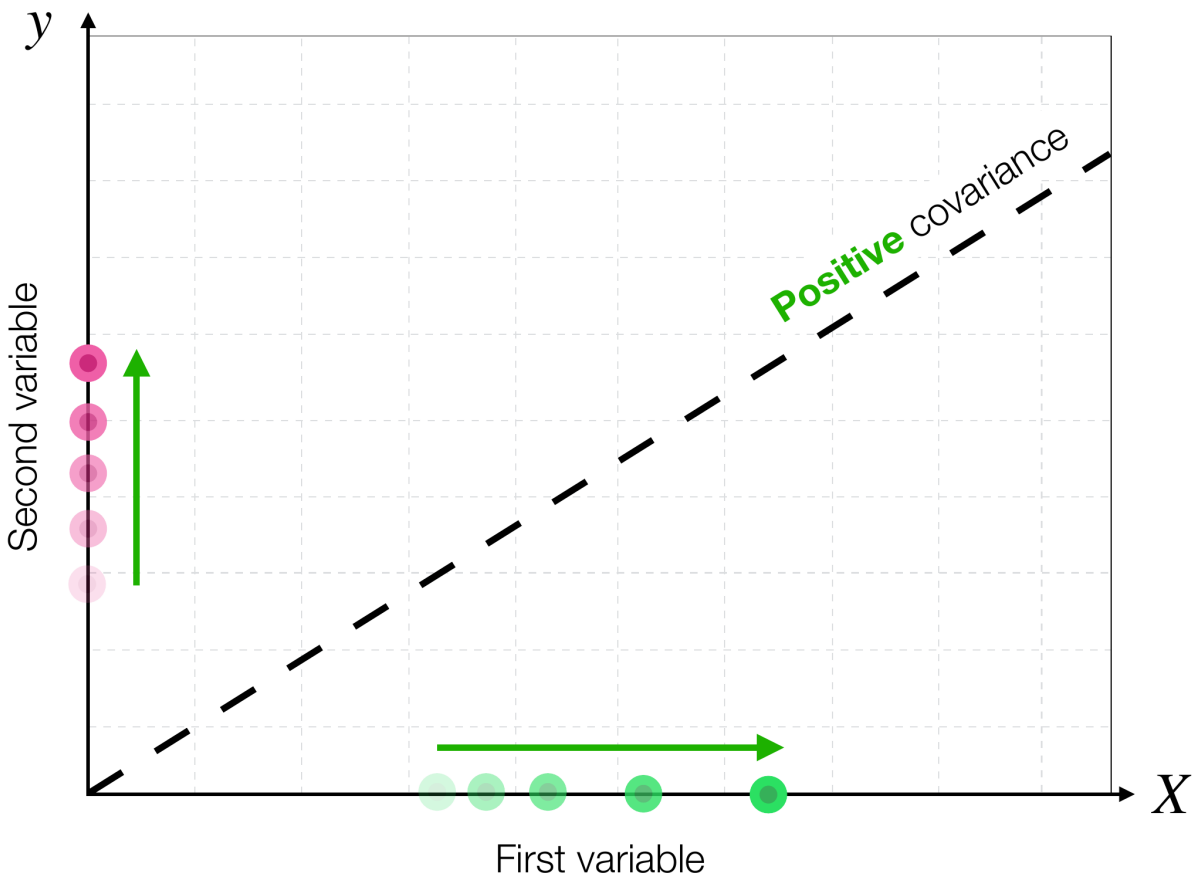




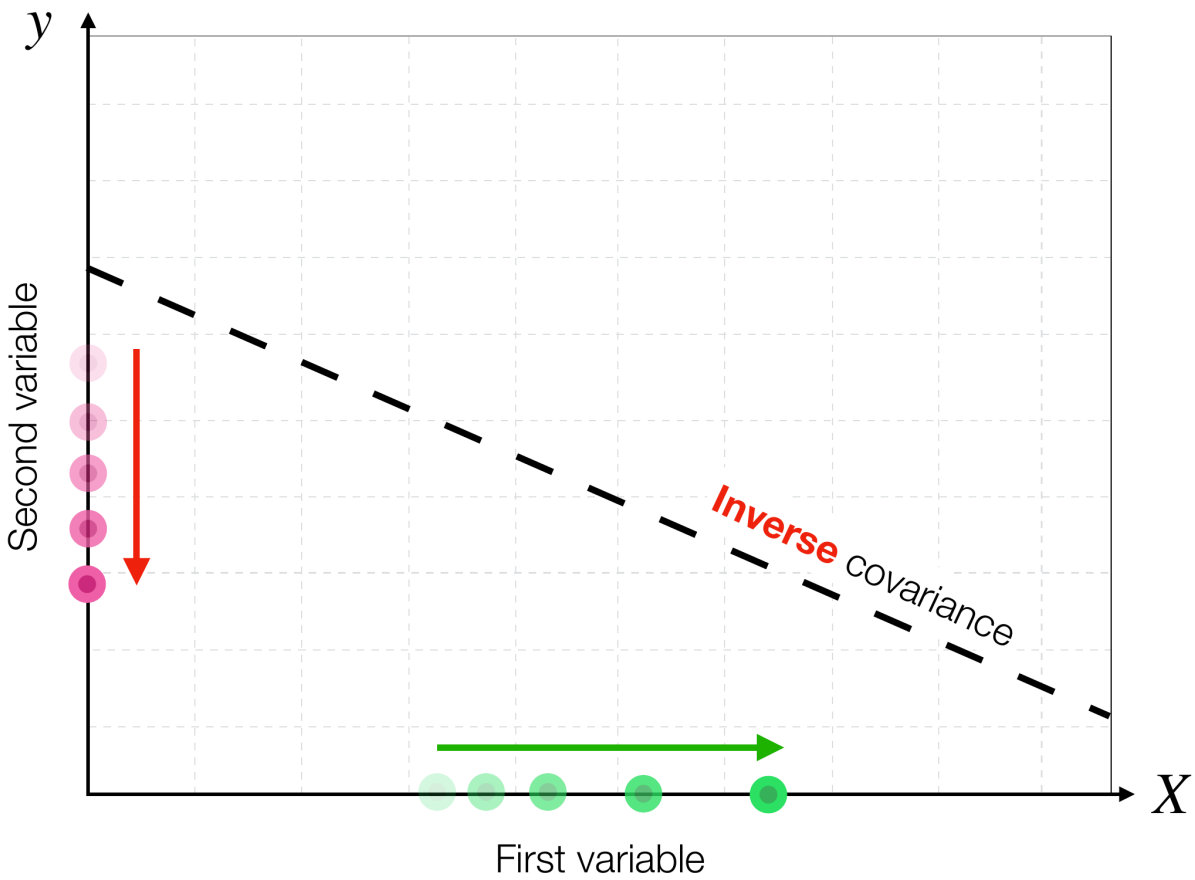
Covariance matrix

Covariance indicates how two variables are related. A **positive** covariance means the variables are **positively related**, while a **negative** covariance means the variables are **inversely related**.

Principal Component Analysis

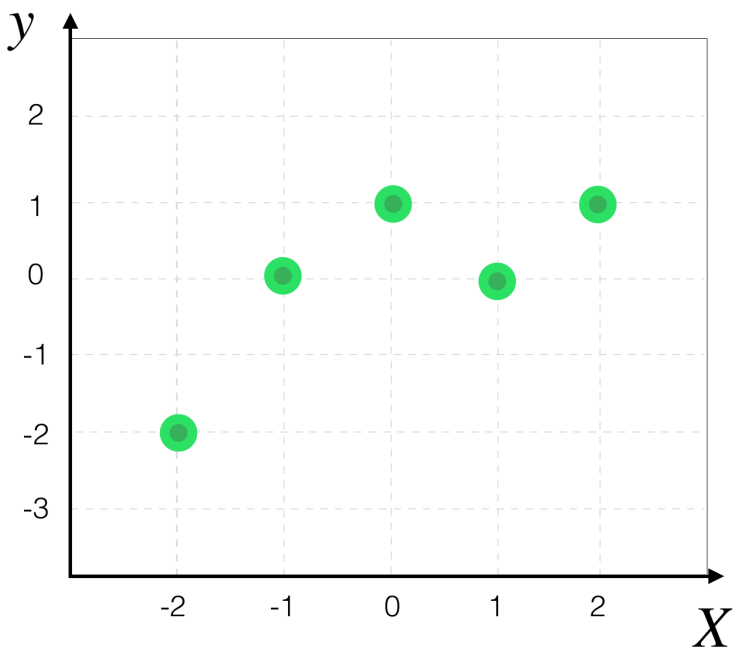


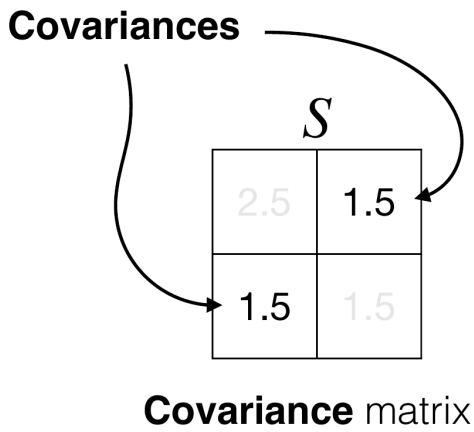
Principal Component Analysis



Principal Component Analysis

How to **interpret** values in **covariance** matrix?





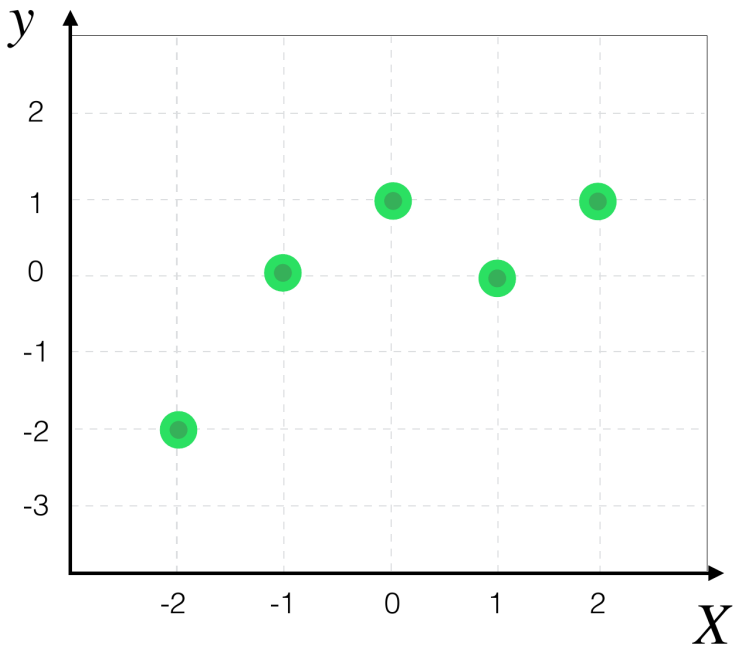
Covariance matrix

Principal Component Analysis

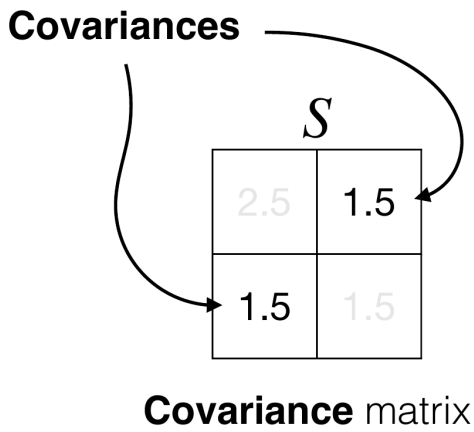
How to **interpret** values in **covariance** matrix?

[-2, -1, 0, 1, 2] [-2, 0, 0, 1, 1]

$\bar{\mathbf{x}} = 0$ $\bar{\mathbf{y}} = 0$



$$cov(x, y) = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{n - 1}$$



Covariance matrix

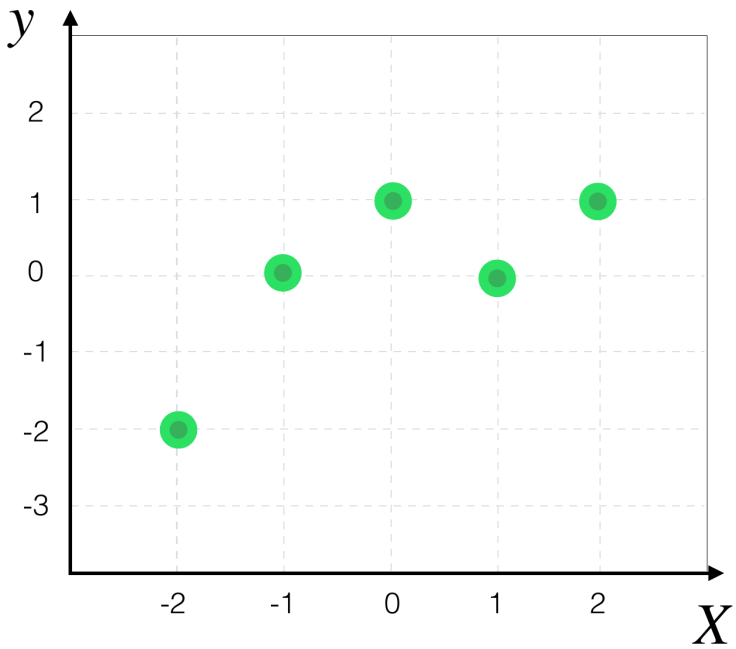
Principal Component Analysis

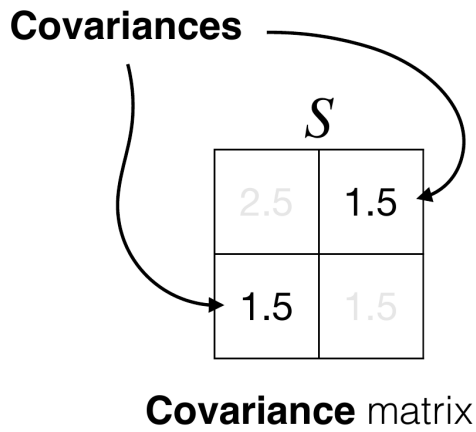
How to **interpret** values in **covariance** matrix?

[-2, -1, 0, 1, 2] [-2, 0, 0, 1, 1]

$\bar{\mathbf{x}} = 0$ $\bar{\mathbf{y}} = 0$

$$cov(x, y) = \frac{6}{4}$$





Covariance matrix

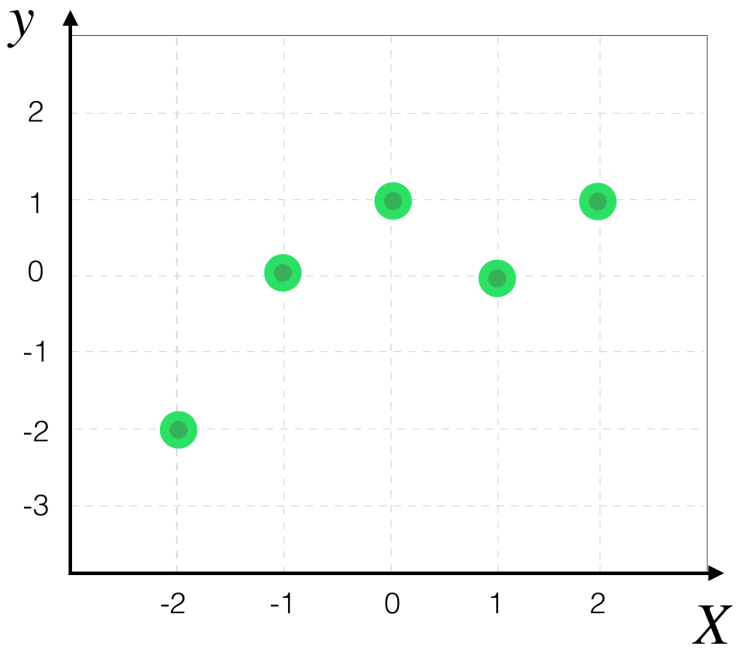
Principal Component Analysis

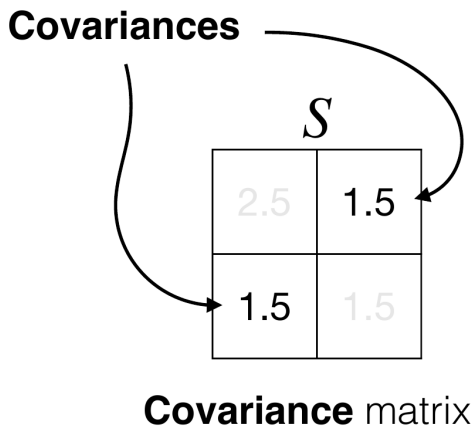
How to **interpret** values in **covariance** matrix?

[-2, -1, 0, 1, 2] [-2, 0, 0, 1, 1]

$\bar{\mathbf{x}} = 0$ $\bar{\mathbf{y}} = 0$

$$cov(x, y) = 1.5$$

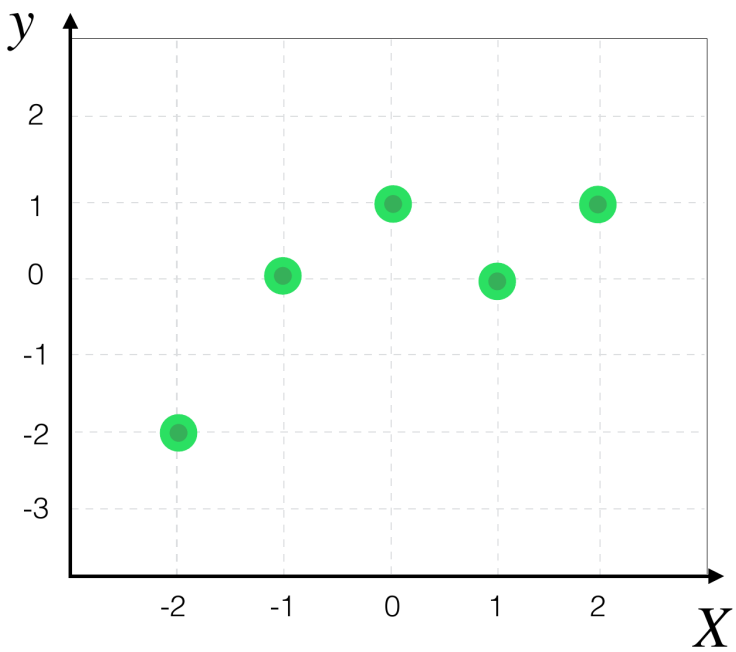




Covariance matrix

Principal Component Analysis

How to **interpret** values in **covariance** matrix?



S

2.5	1.5
1.5	1.5

Covariance matrix

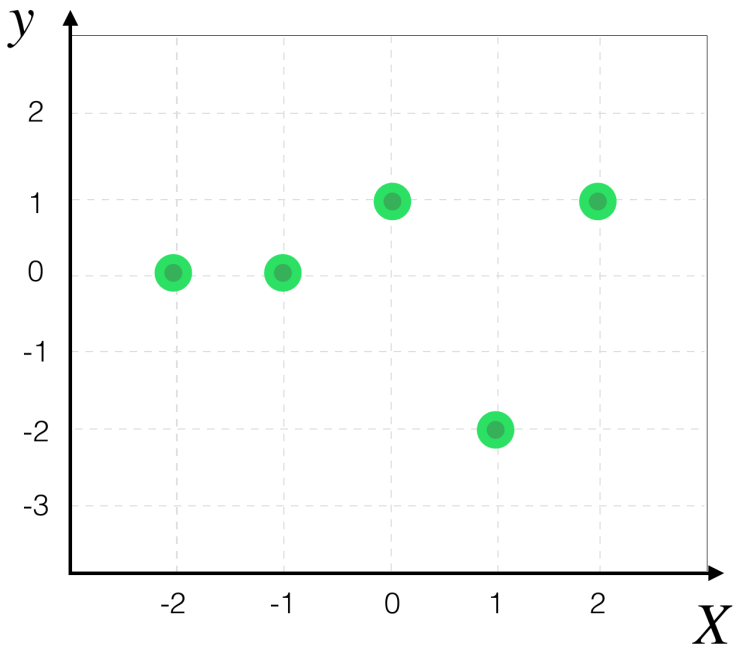
Principal Component Analysis

How to **interpret** values in **covariance** matrix?

[-2, -1, 0, 1, 2] [0, 0, 1,-2, 1]

$\bar{\mathbf{x}} = ?$ $\bar{\mathbf{y}} = ?$

$$cov(x, y) = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{n - 1}$$



S

2.5	?
?	1.5

Covariance matrix

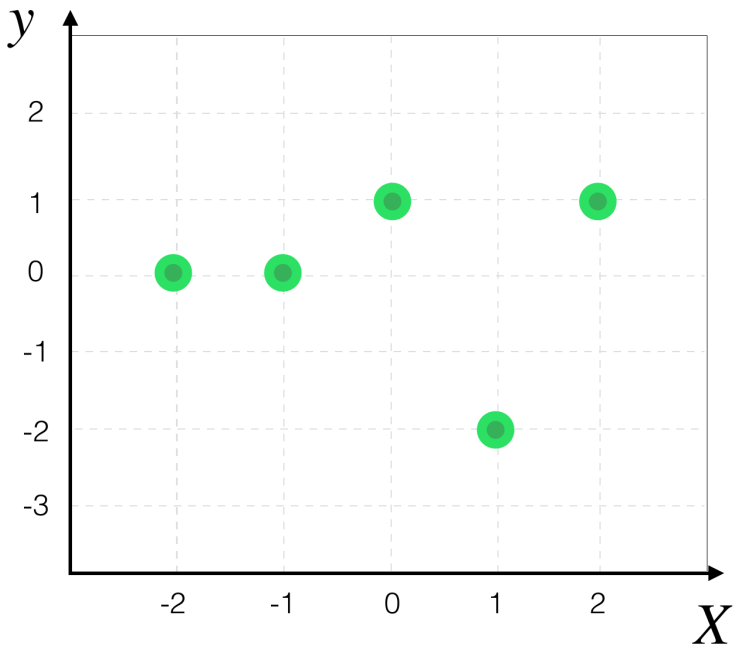
Principal Component Analysis

How to **interpret** values in **covariance** matrix?

[-2, -1, 0, 1, 2] [0, 0, 1,-2, 1]

$\bar{\mathbf{x}} = 0$ $\bar{\mathbf{y}} = 0$

$$cov(x, y) = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{n - 1}$$



S

2.5	?
?	1.5

Covariance matrix

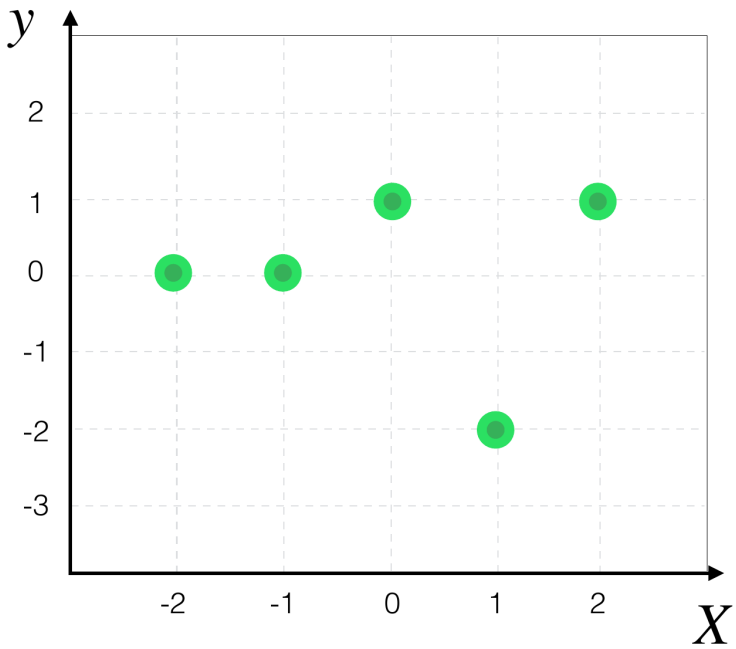
Principal Component Analysis

How to **interpret** values in **covariance** matrix?

[-2, -1, 0, 1, 2] [0, 0, 1,-2, 1]

$\bar{\mathbf{x}} = 0$ $\bar{\mathbf{y}} = 0$

$$cov(x, y) = \frac{\sum (x_i)(y_i)}{4}$$



S

2.5	?
?	1.5

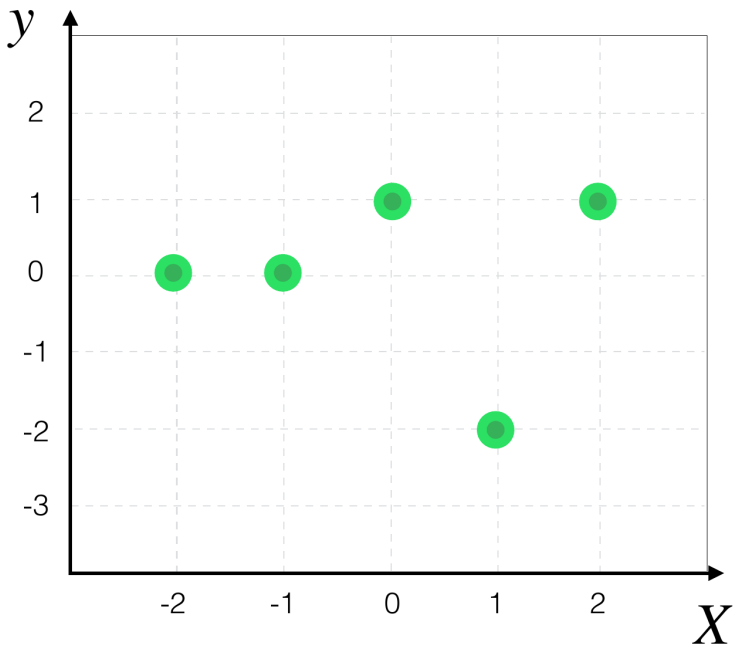
Covariance matrix

Principal Component Analysis

How to **interpret** values in **covariance** matrix?

[-2, -1, 0, 1, 2] [0, 0, 1,-2, 1]

$\bar{\mathbf{x}} = 0$ $\bar{\mathbf{y}} = 0$



$$cov(x, y) = \frac{(-2)(0) + (-1)(0) + (0)(1) + (1)(-2) + (2)(1)}{4}$$

S

2.5	?
?	1.5

Covariance matrix

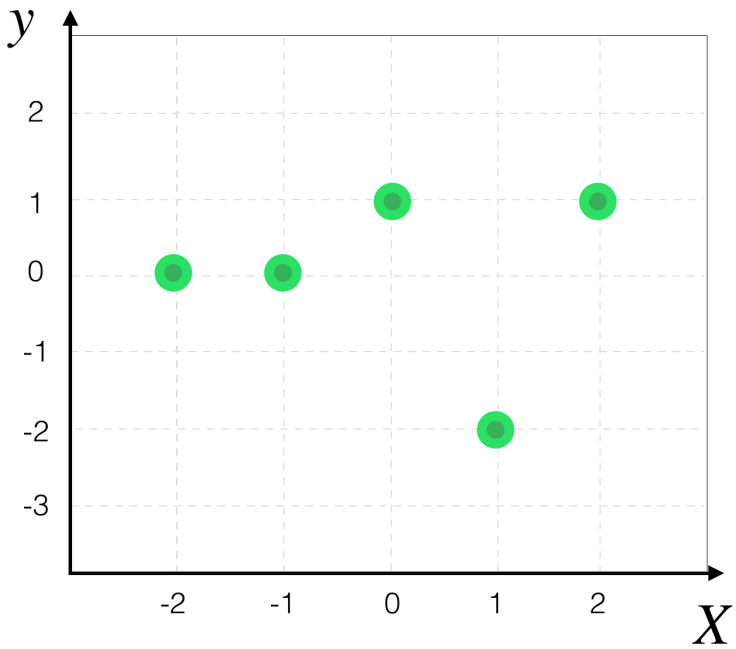
Principal Component Analysis

How to **interpret** values in **covariance** matrix?

[-2, -1, 0, 1, 2] [0, 0, 1,-2, 1]

$\bar{\mathbf{x}} = 0$ $\bar{\mathbf{y}} = 0$

$$cov(x, y) = \frac{0}{4} = 0$$



$$S$$

2.5	?
?	1.5

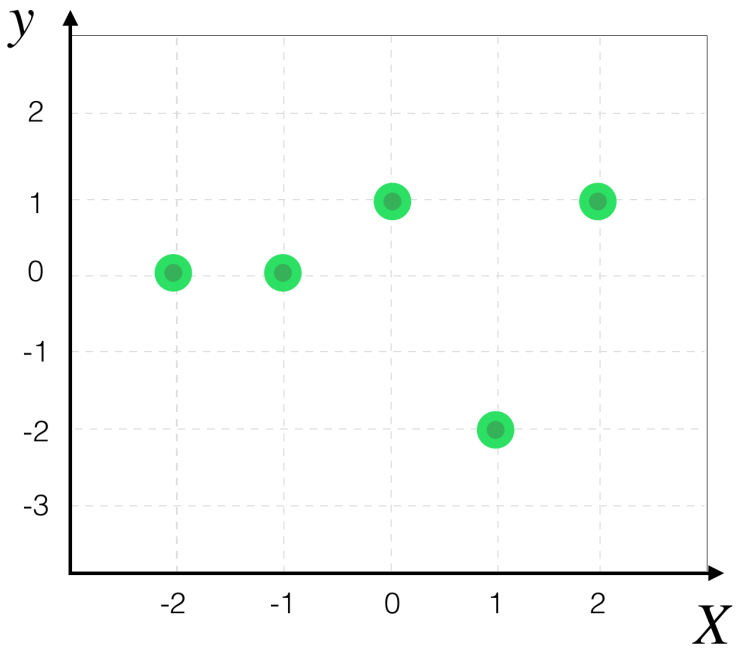
Covariance matrix

Principal Component Analysis

How to **interpret** values in **covariance** matrix?

[-2, -1, 0, 1, 2] [0, 0, 1,-2, 1]

$\bar{\mathbf{x}} = 0$ $\bar{\mathbf{y}} = 0$



$$cov(x, y) = \frac{0}{4} = 0$$

S

2.5	0
0	1.5

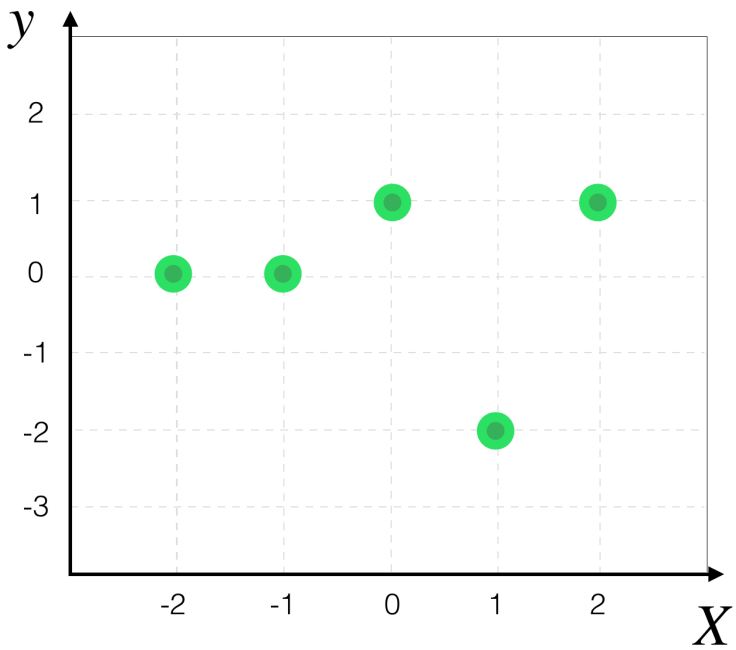
Covariance matrix

Principal Component Analysis

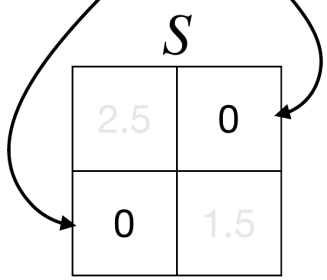
How to **interpret** values in **covariance** matrix?

[-2, -1, 0, 1, 2] [0, 0, 1,-2, 1]

$\bar{\mathbf{x}} = 0$ $\bar{\mathbf{y}} = 0$



$$cov(x, y) = \frac{0}{4} = 0$$

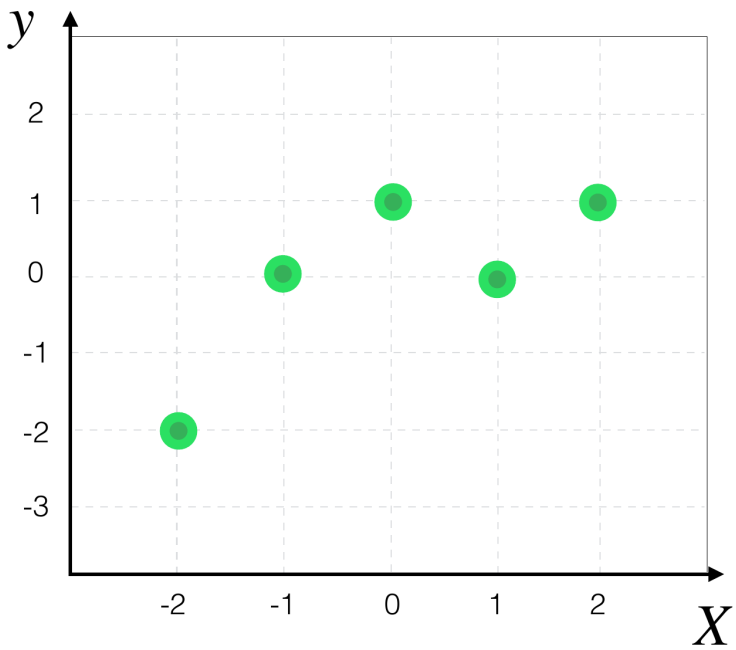


Covariance matrix

Covariance **0** means that there is **no relationship** between two variables. Knowing something about the value of one does not say anything about the value of the other.

Principal Component Analysis

How to **interpret** values in **covariance** matrix?



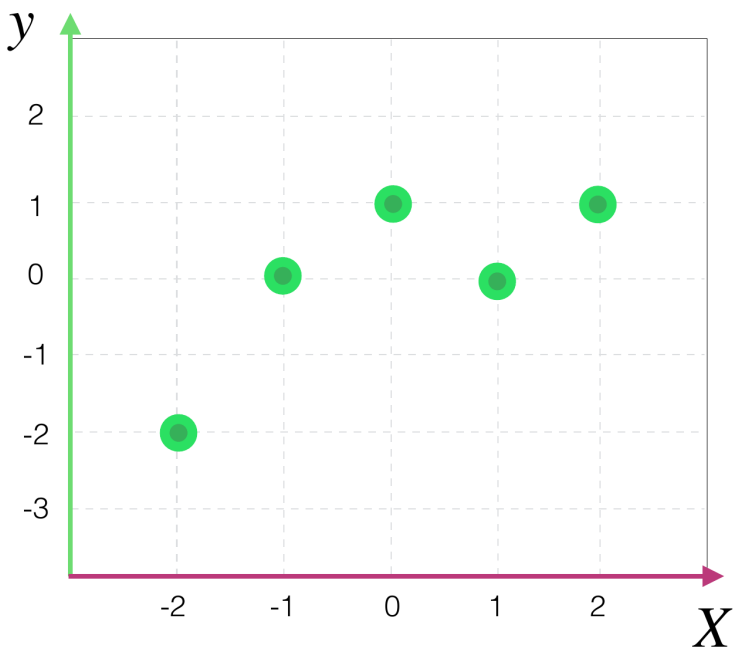
S

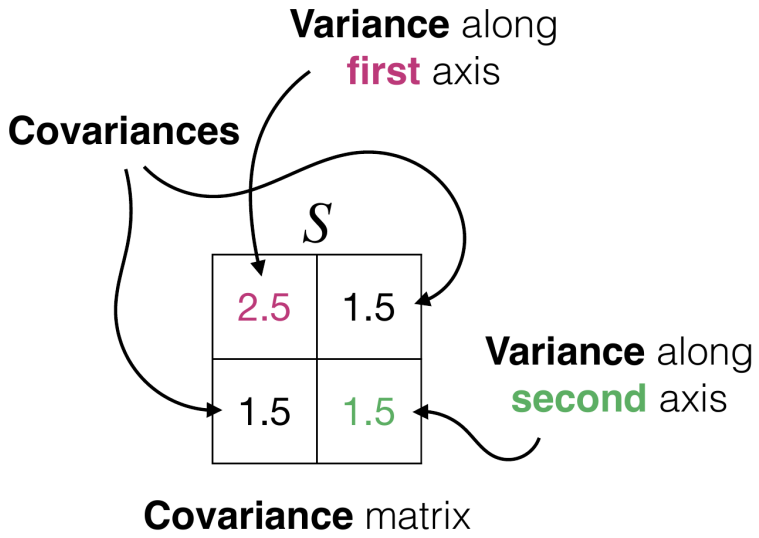
2.5	1.5
1.5	1.5

Covariance matrix

Principal Component Analysis

How to **interpret** values in **covariance** matrix?





Principal Component Analysis

Z^T

-2	-1	0	1	2
-2	0	1	0	1

$\times$

Z

-2	-2
-1	0
0	1
1	0
2	1

$=$

S

2.5	1.5
1.5	1.5

$\times 4$

Principal Component Analysis

$$S = \begin{bmatrix} 2.5 & 1.5 \\ 1.5 & 1.5 \end{bmatrix} = PDP^{\top}$$

Principal Component Analysis

$$S = PDP^{\top}$$

$$\begin{bmatrix} 2.5 & 1.5 \\ 1.5 & 1.5 \end{bmatrix} = \begin{bmatrix} 2.5 & 1.5 \\ 1.5 & 1.5 \end{bmatrix}$$

For an example: https://www.scss.tcd.ie/Rozenn.Dahyot/CS1BA1/SolutionEigen.pdf

Principal Component Analysis

$$
\underset{S}{\begin{bmatrix} 2.5 & 1.5 \\ 1.5 & 1.5 \end{bmatrix}} = \underset{PD}{\begin{bmatrix} -2.9 & 0.24 \\ -2.1 & -0.33 \end{bmatrix}} \times \underset{P^{\top}}{\begin{bmatrix} -0.81 & -0.58 \\ 0.58 & -0.81 \end{bmatrix}}
$$

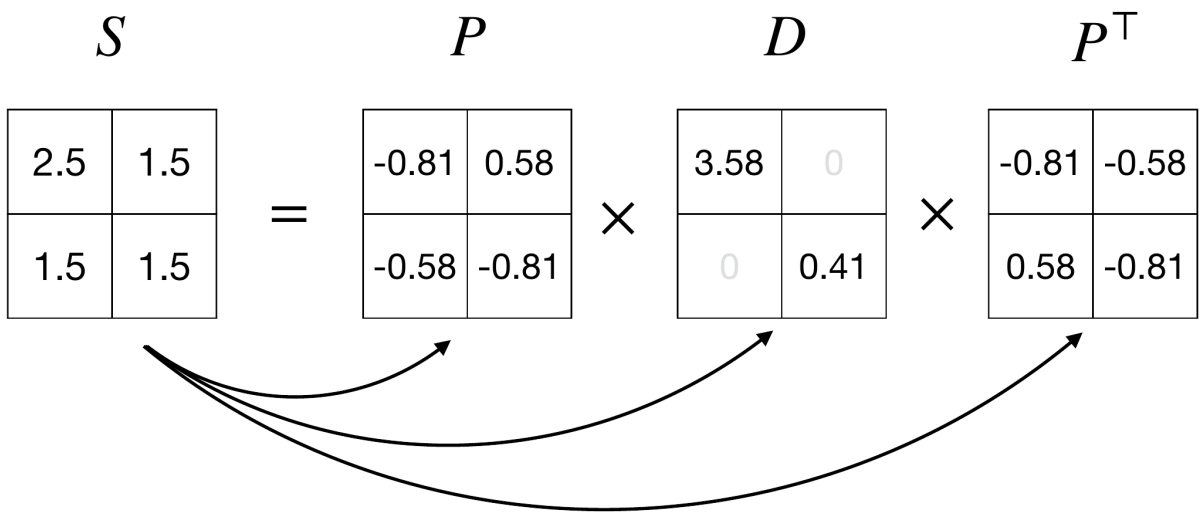
For an example: https://www.scss.tcd.ie/Rozenn.Dahyot/CS1BA1/SolutionEigen.pdf

Principal Component Analysis

$$
\underset{S}{\begin{bmatrix} 2.5 & 1.5 \\ 1.5 & 1.5 \end{bmatrix}} = \underset{P}{\begin{bmatrix} -0.81 & 0.58 \\ -0.58 & -0.81 \end{bmatrix}} \times \underset{D}{\begin{bmatrix} 3.58 & 0 \\ 0 & 0.41 \end{bmatrix}} \times \underset{P^{\top}}{\begin{bmatrix} -0.81 & -0.58 \\ 0.58 & -0.81 \end{bmatrix}}
$$

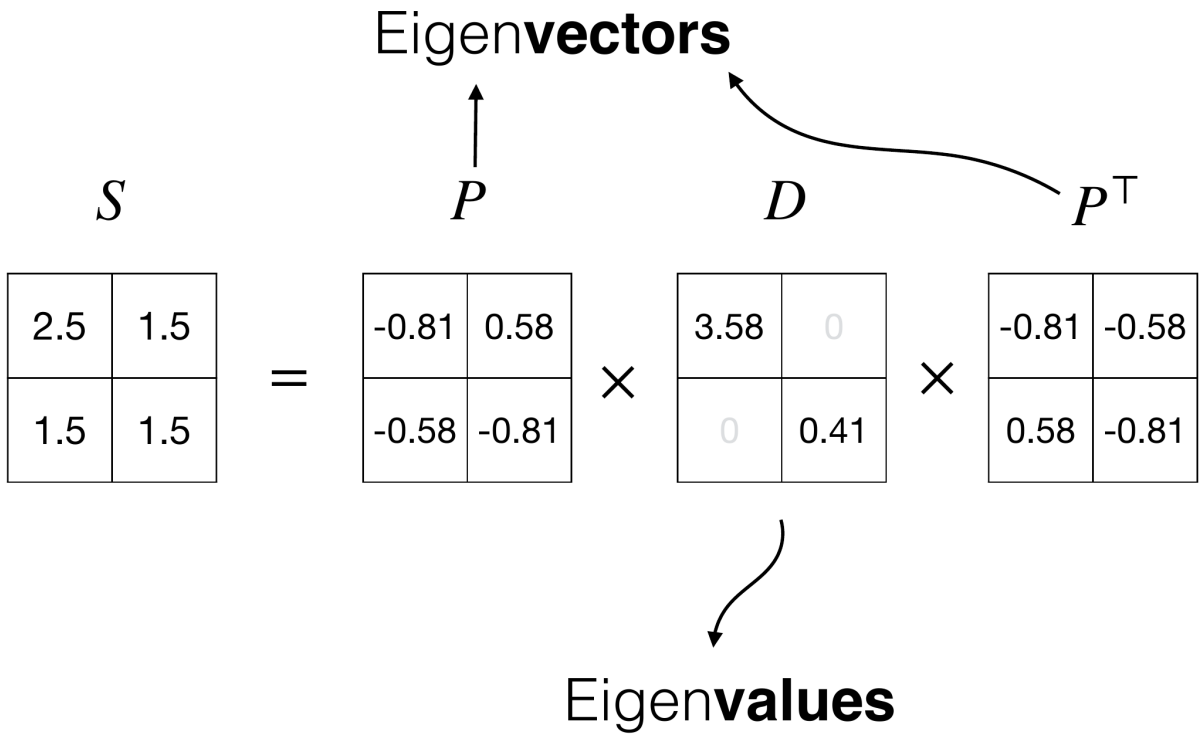
For an example: https://www.scss.tcd.ie/Rozenn.Dahyot/CS1BA1/SolutionEigen.pdf

Eigen**decomposition**



For an example: https://www.scss.tcd.ie/Rozenn.Dahyot/CS1BA1/SolutionEigen.pdf

Eigen**decomposition**

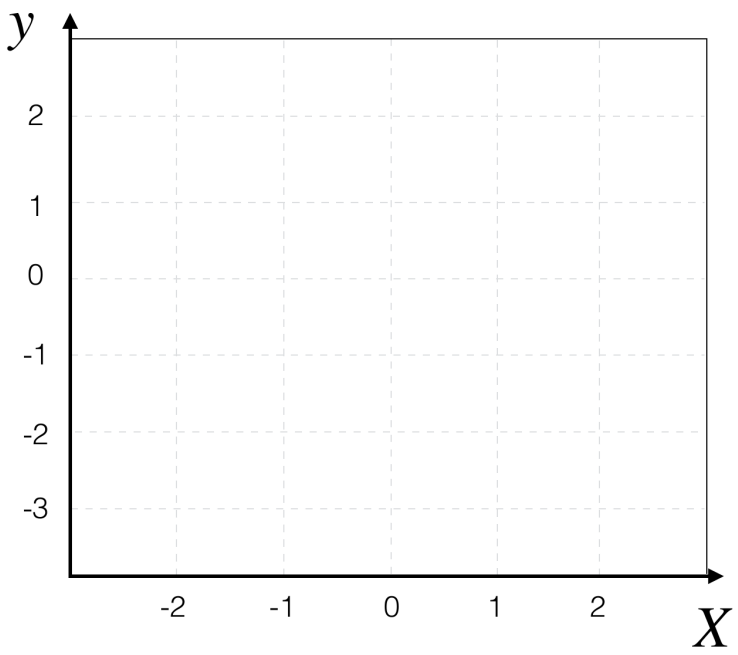


For an example: https://www.scss.tcd.ie/Rozenn.Dahyot/CS1BA1/SolutionEigen.pdf

Principal Component Analysis

Eigen**vectors**

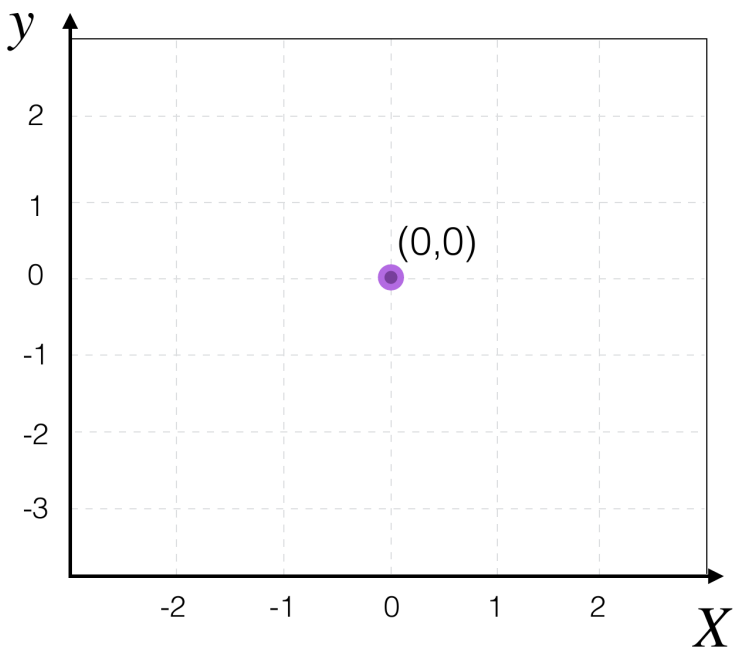
-0.81	0.58
-0.58	-0.81



Principal Component Analysis

Eigen**vectors**

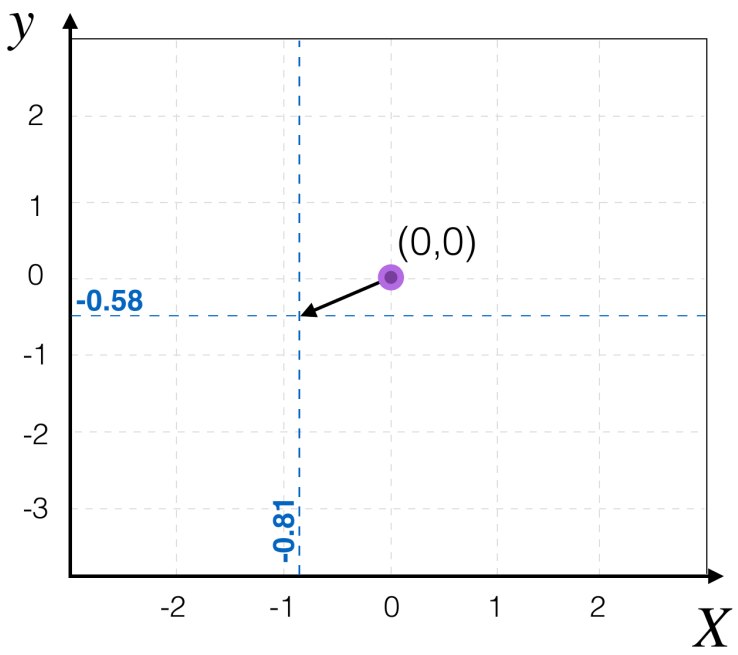
-0.81	0.58
-0.58	-0.81



Principal Component Analysis

Eigen**vectors**

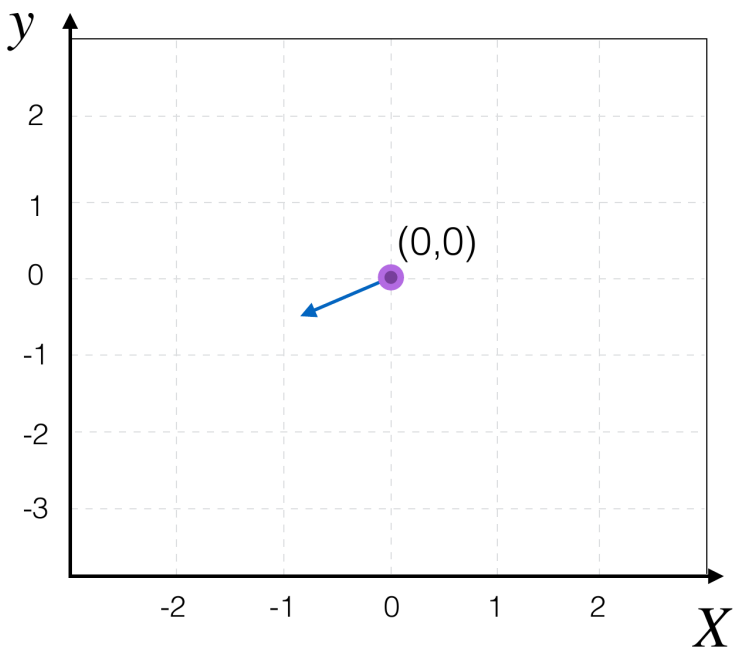
-0.81	0.58
-0.58	-0.81



Principal Component Analysis

Eigen**vectors**

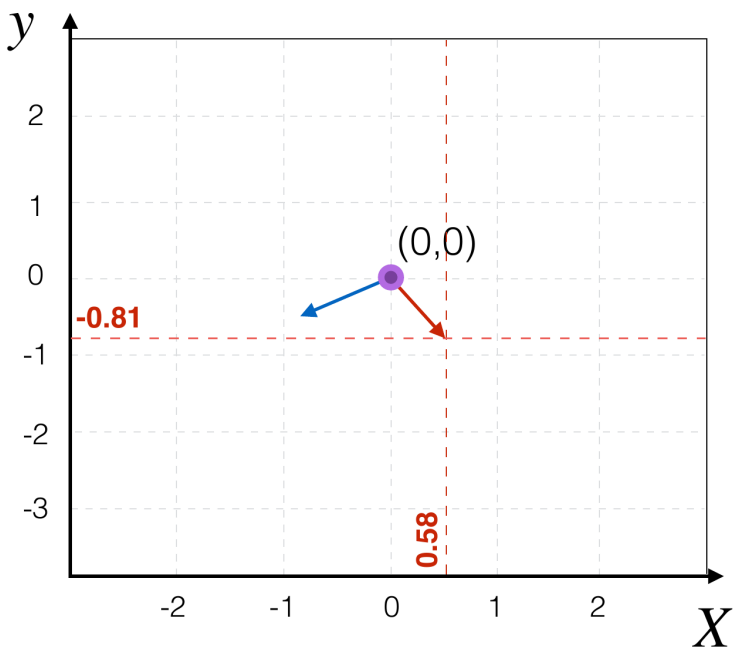
-0.81	0.58
-0.58	-0.81



Principal Component Analysis

Eigen**vectors**

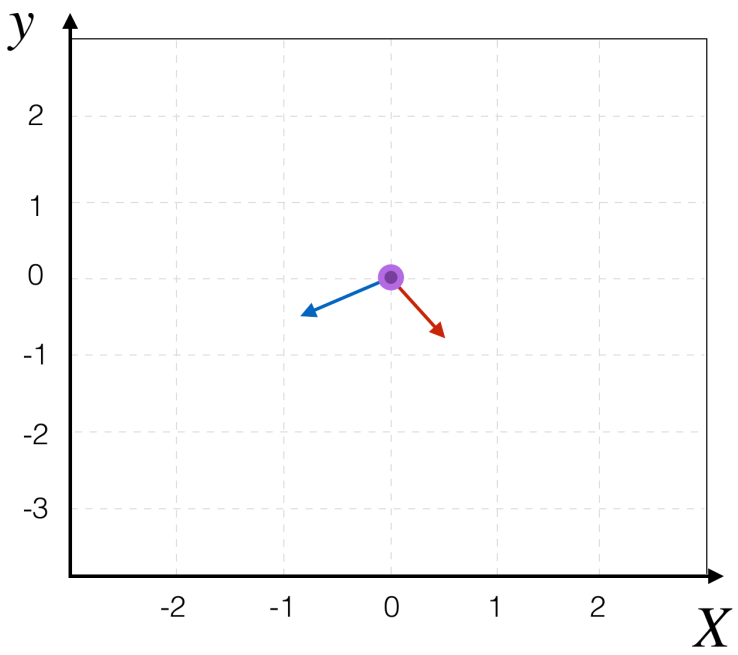
-0.81	0.58
-0.58	-0.81



Principal Component Analysis

Eigen**vectors**

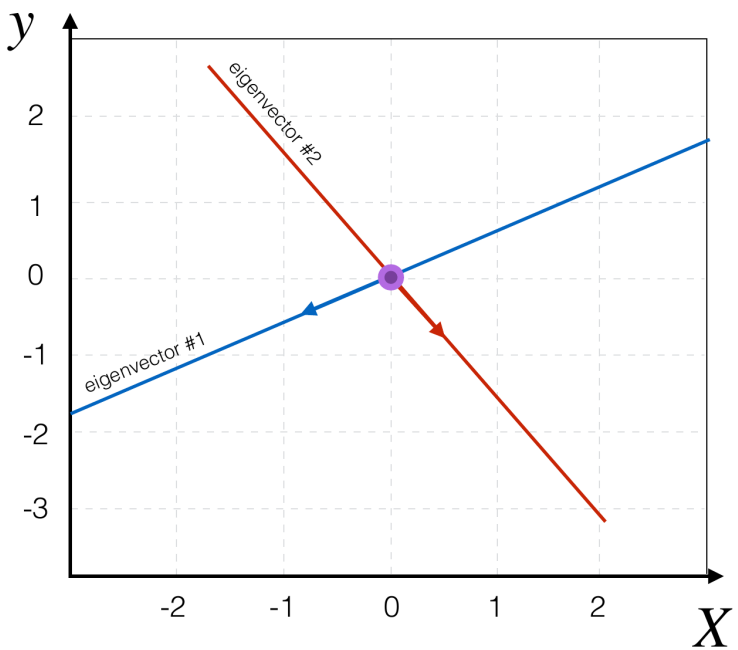
-0.81	0.58
-0.58	-0.81



Principal Component Analysis

Eigen**vectors**

-0.81	0.58
-0.58	-0.81

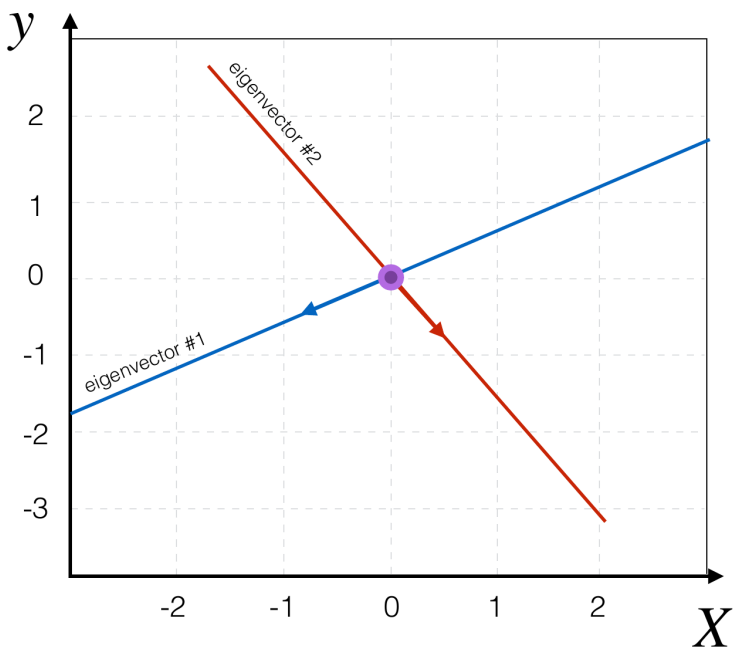


Old coordinate system

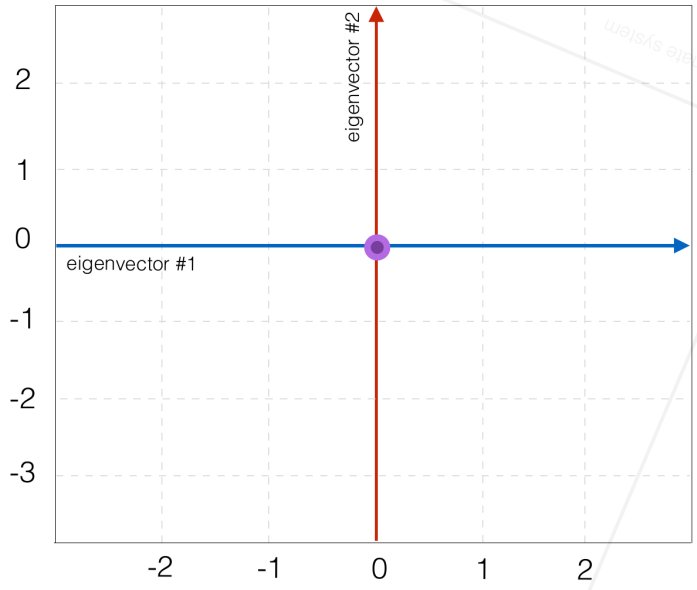
Principal Component Analysis

Eigen**vectors**

-0.81	0.58
-0.58	-0.81



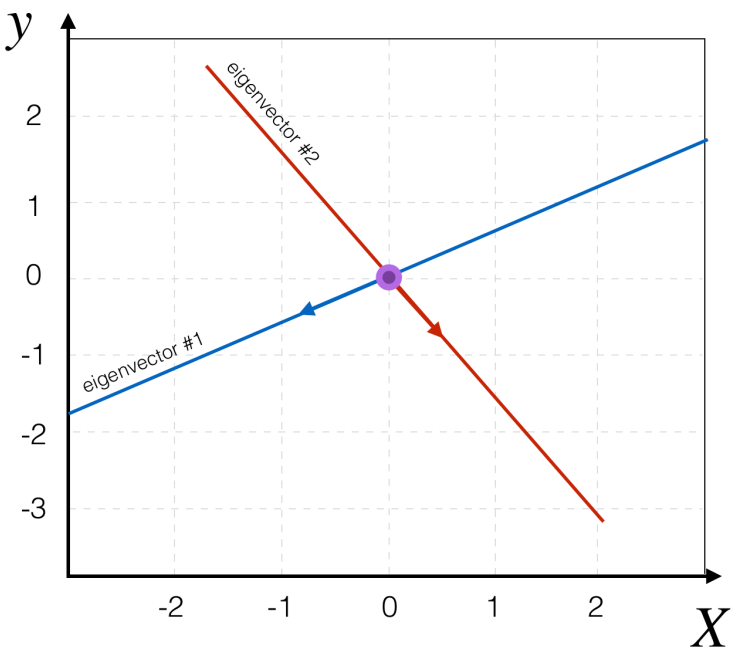
Old coordinate system



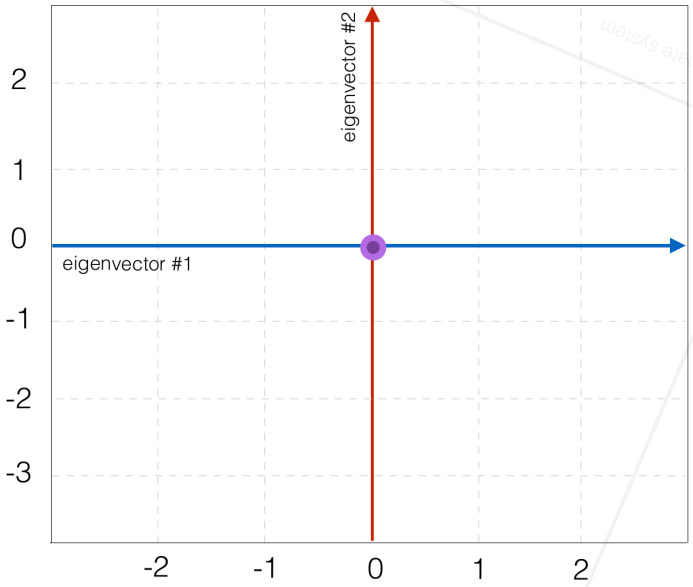
New coordinate system

Principal Component Analysis

Eigen**vectors**		transpose	Eigen**vectors**^{T}	
-0.81	0.58	=	-0.81	-0.58
-0.58	-0.81		0.58	-0.81



Old coordinate system

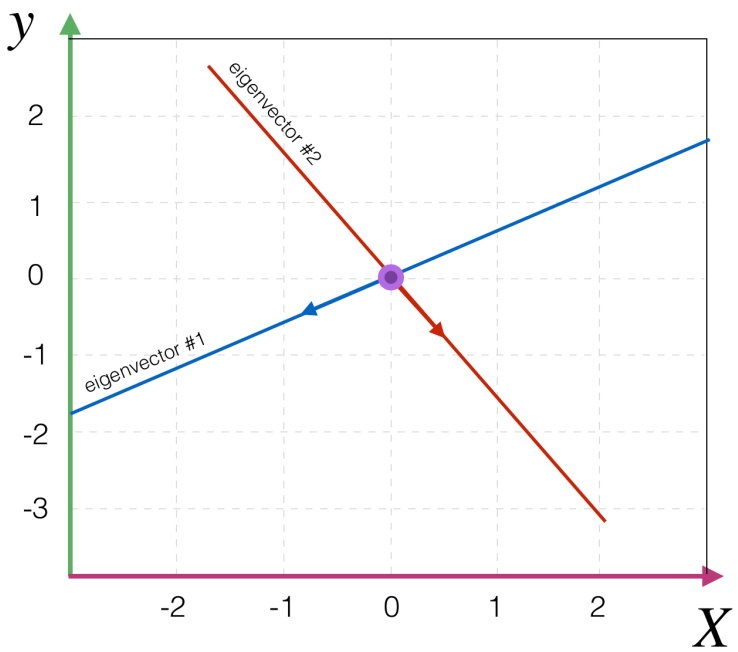


New coordinate system

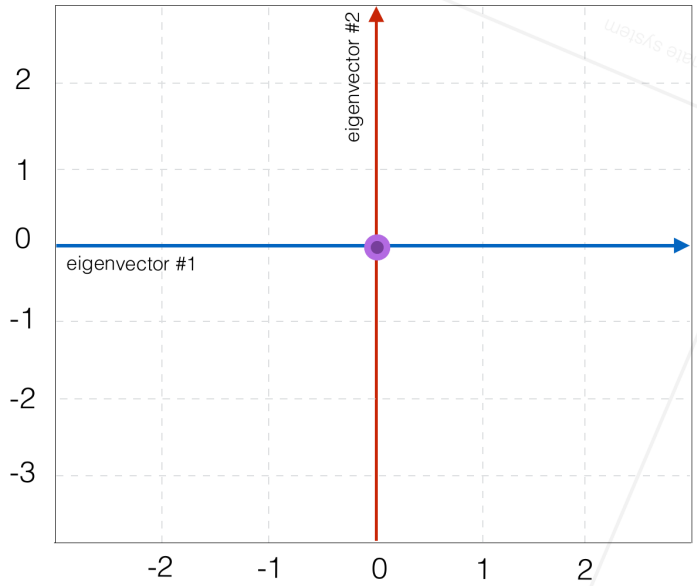
Principal Component Analysis

Eigen**vectors**$^{\mathsf{T}}$

-0.81	-0.58
0.58	-0.81



Old coordinate system



New coordinate system

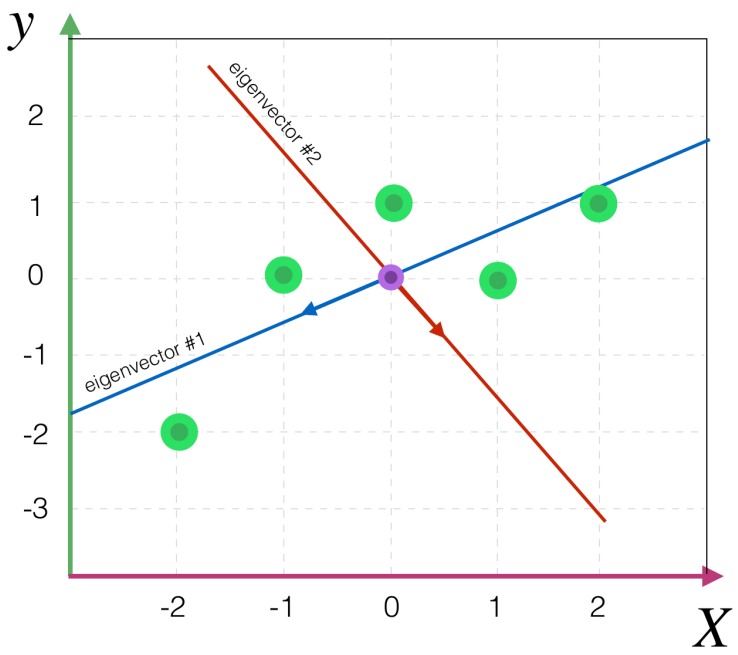
Principal Component Analysis

Eigen**vectors**$^{\mathsf{T}}$

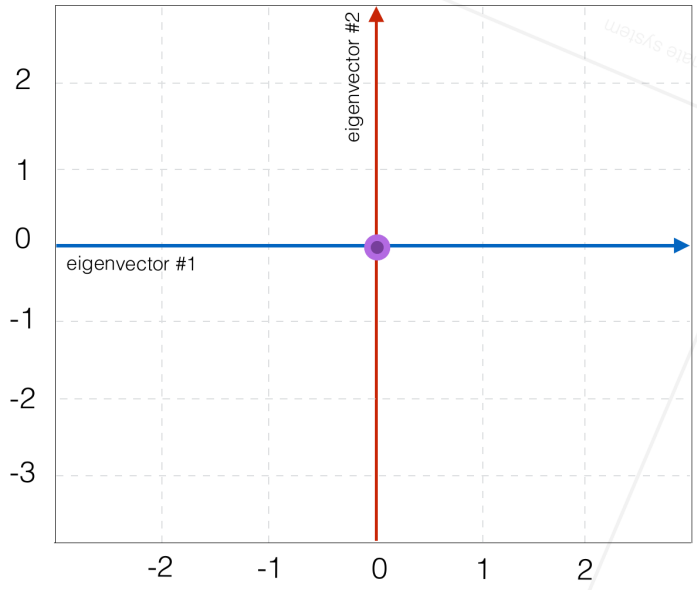
-0.81	-0.58
0.58	-0.81

Z^{T}

-2	-1	0	1	2
-2	0	1	0	1



Old coordinate system



New coordinate system

Principal Component Analysis

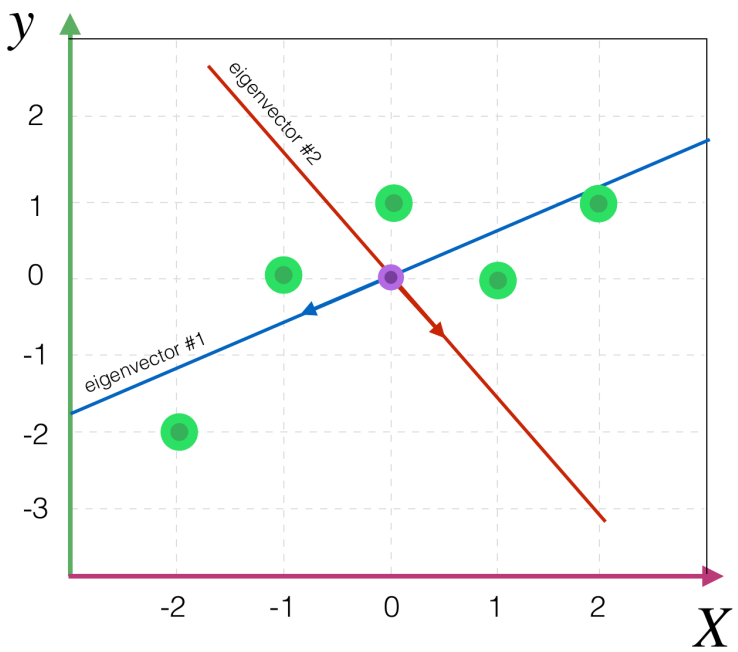
Eigen**vectors**$^{\mathsf{T}}$

-0.81	-0.58
0.58	-0.81

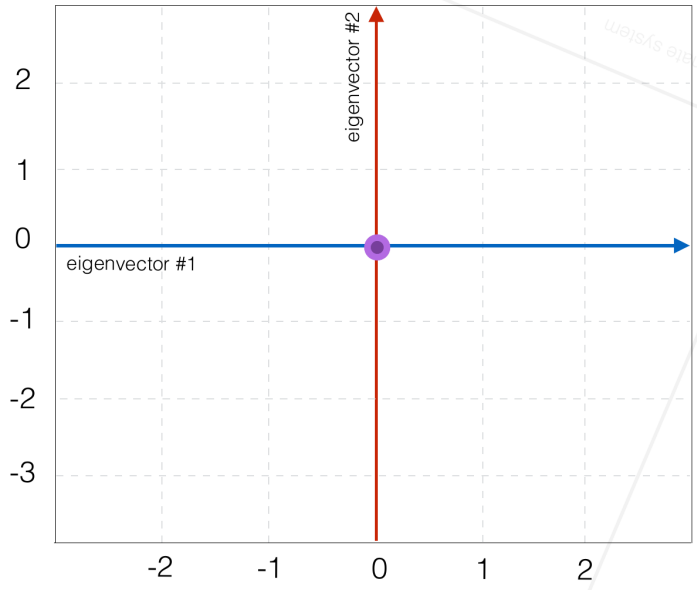
$\times$

Z^{T}

-2	-1	0	1	2
-2	0	1	0	1



Old coordinate system



New coordinate system

Principal Component Analysis

Eigen**vectors**$^{\mathsf{T}}$

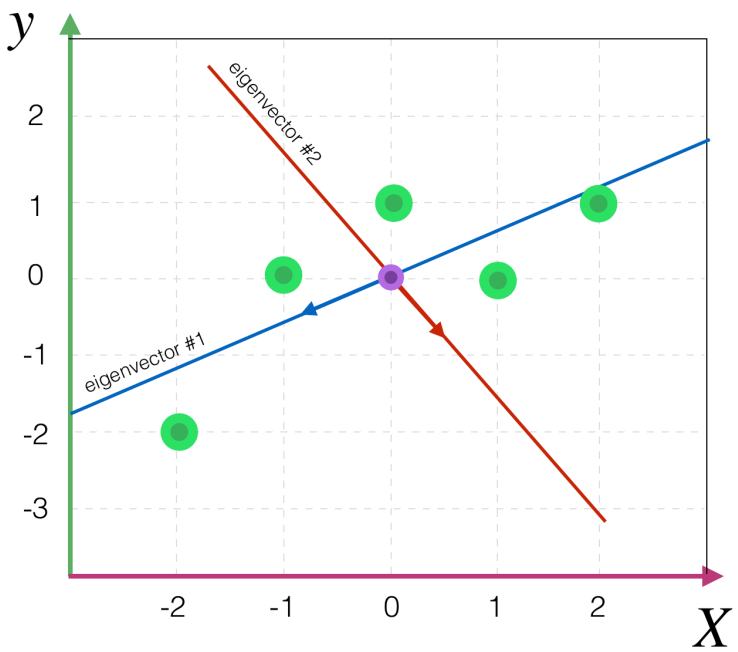
-0.81	-0.58
0.58	-0.81

$\times$

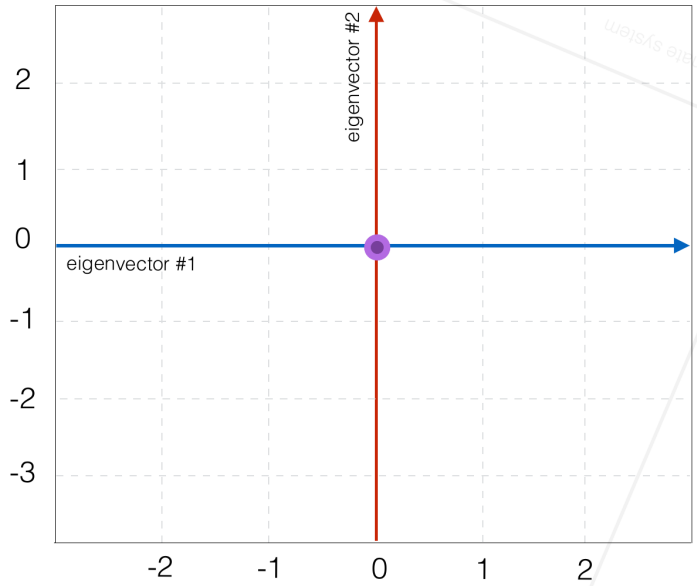
Z^{T}

-2	-1	0	1	2
-2	0	1	0	1

$=$



Old coordinate system



New coordinate system

Principal Component Analysis

Eigen**vectors**$^{\top}$

-0.81	-0.58
0.58	-0.81

$\times$

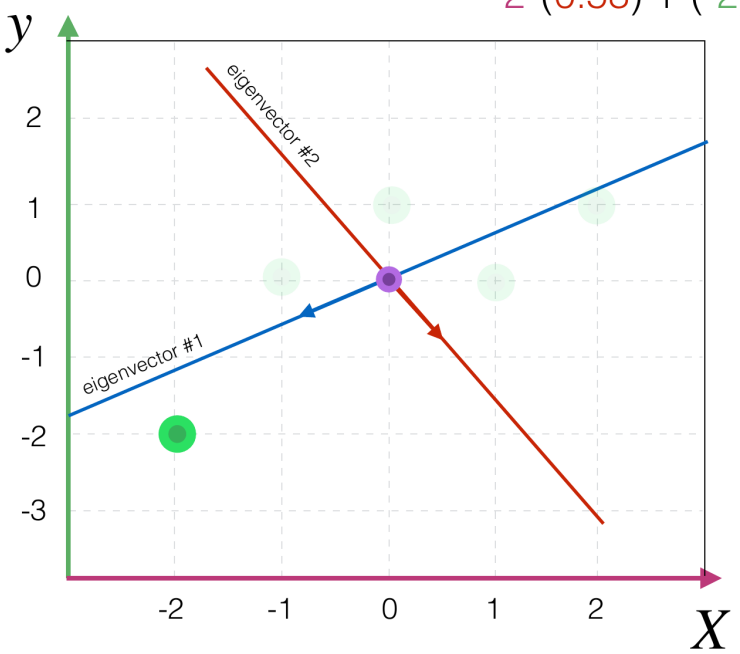
$Z^{\top}$

-2	-1	0	1	2
-2	0	1	0	1

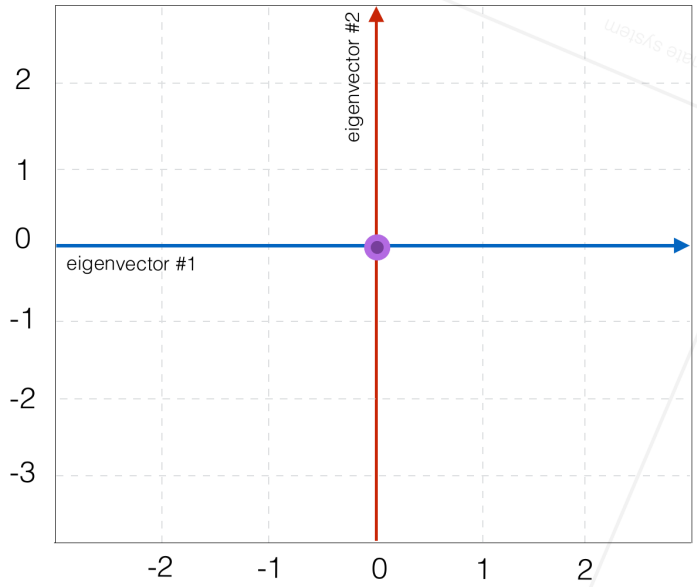
$=$

-2*(-0.81) + (-2)*(-0.58) = **2.78**

-2*(0.58) + (-2)*(-0.81) = **0.46**



Old coordinate system



New coordinate system

Principal Component Analysis

Eigen**vectors**$^{\top}$

-0.81	-0.58
0.58	-0.81

$\times$

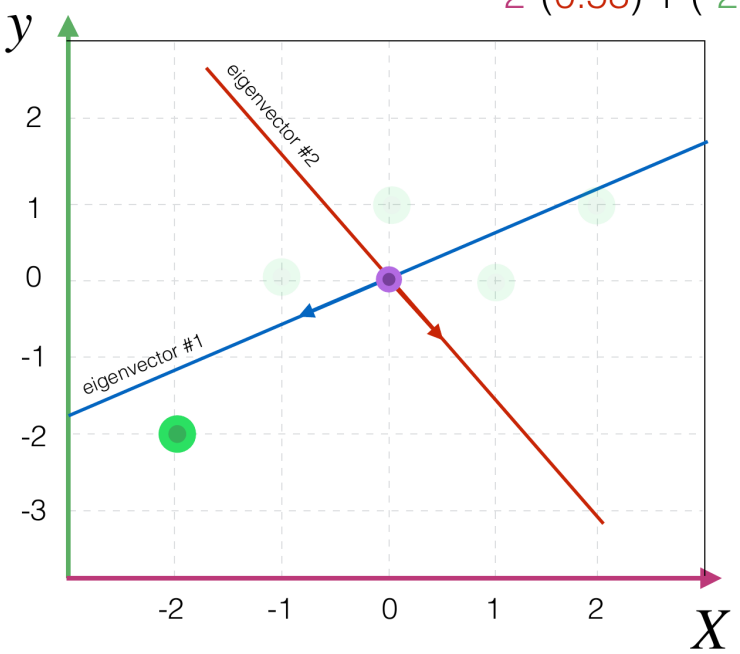
$Z^{\top}$

-2	-1	0	1	2
-2	0	1	0	1

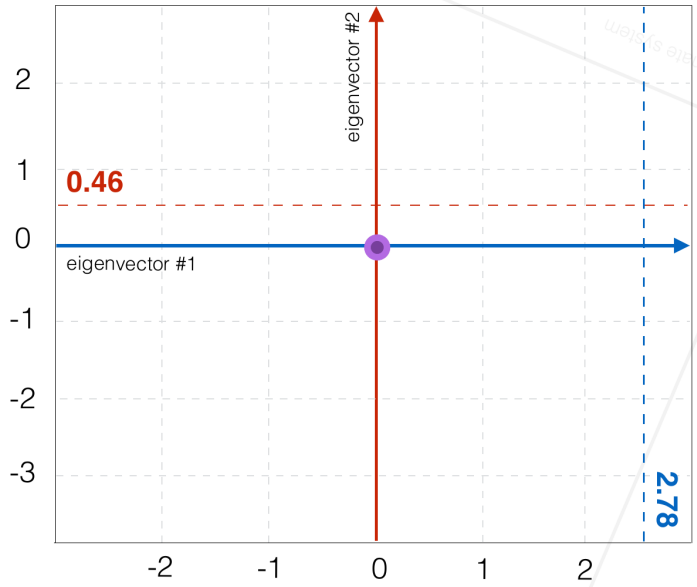
$=$

-2*(-0.81) + (-2)*(-0.58) = **2.78**

-2*(0.58) + (-2)*(-0.81) = **0.46**

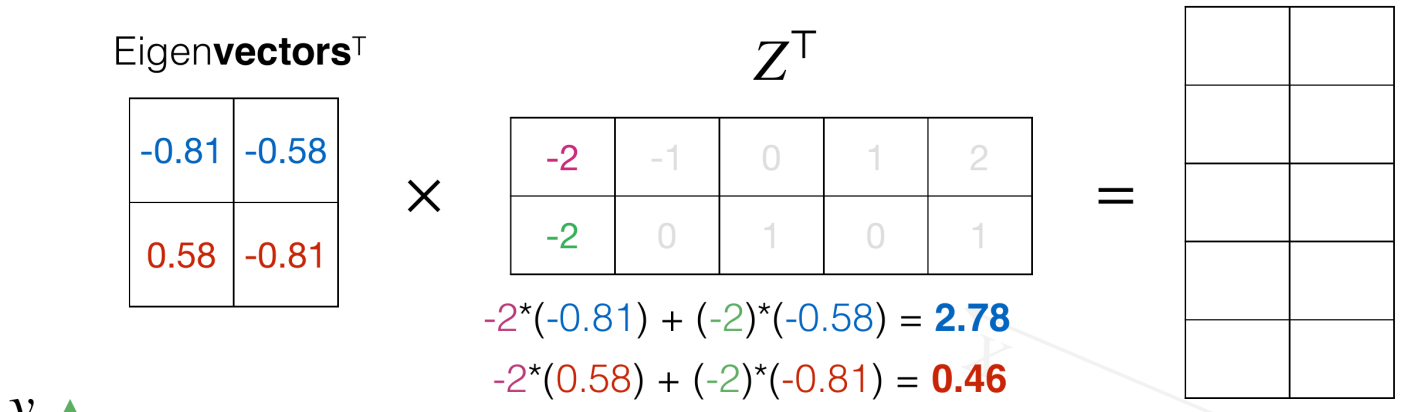


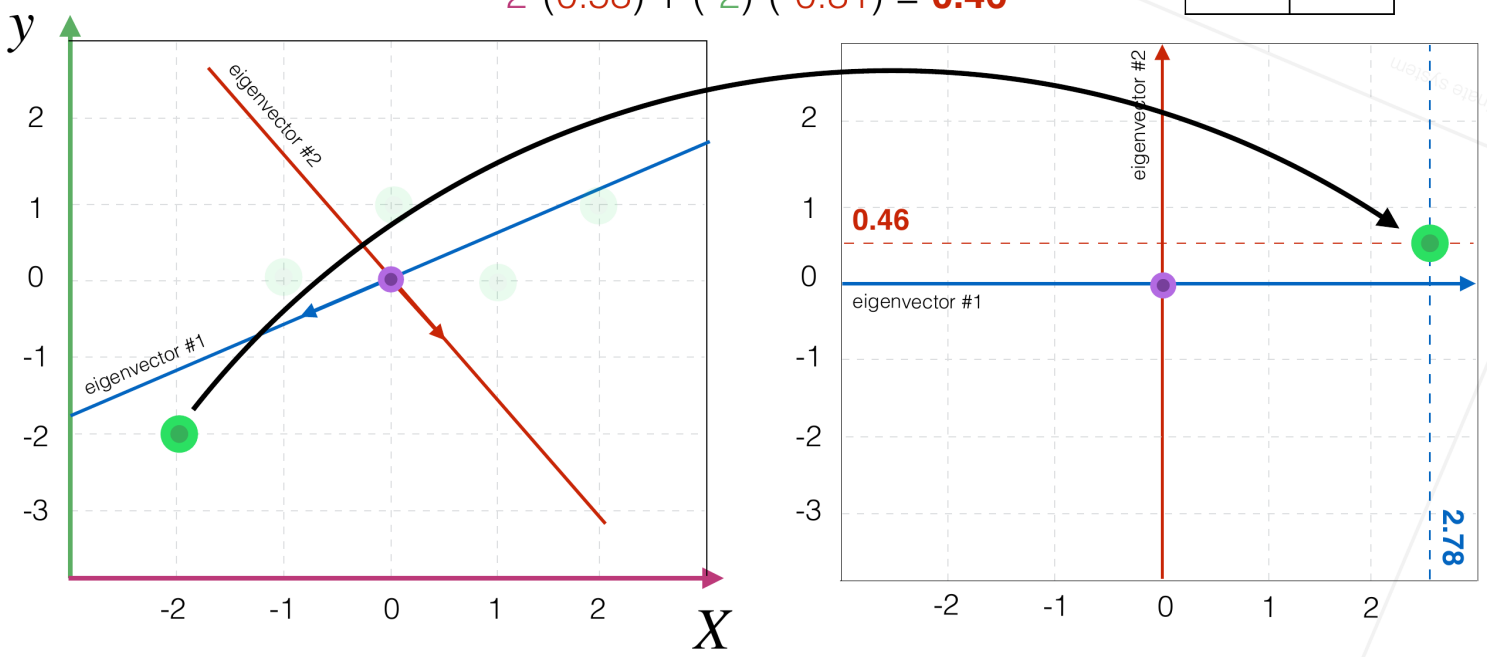
Old coordinate system



New coordinate system

Principal Component Analysis





Old coordinate system

New coordinate system

Principal Component Analysis

Eigen**vectors**$^{\mathsf{T}}$

-0.81	-0.58
0.58	-0.81

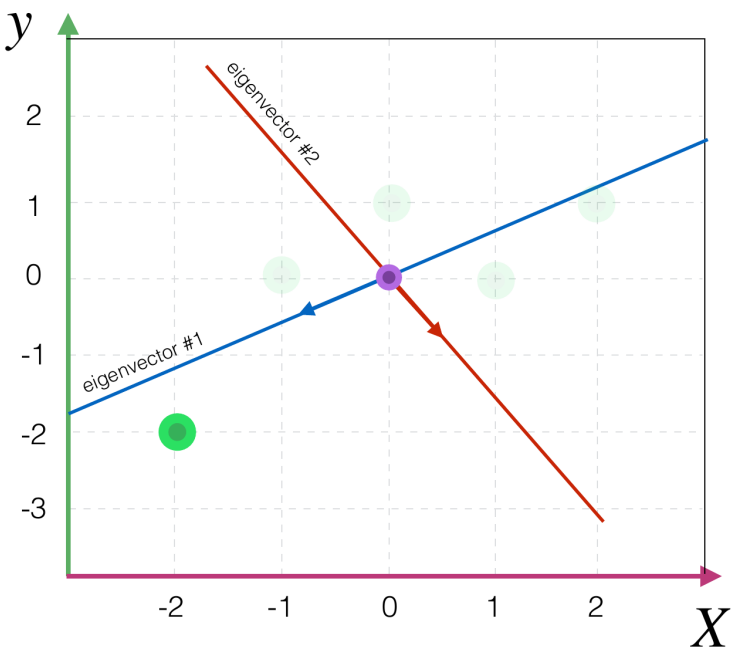
$\times$

Z^{T}

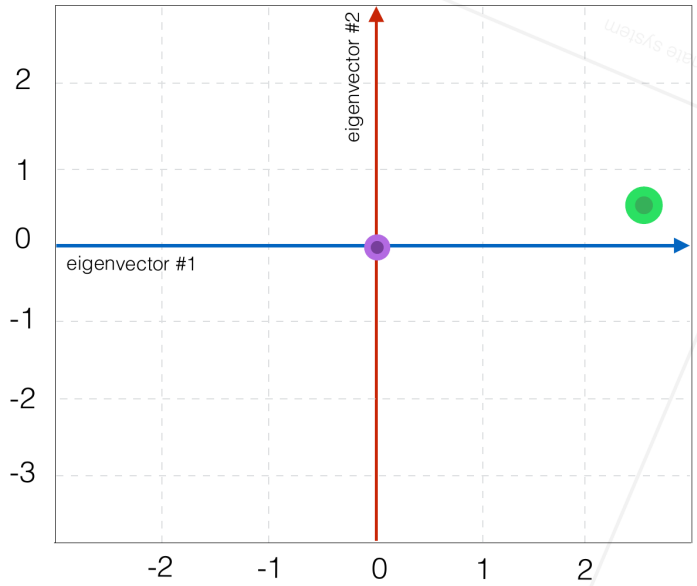
-2	-1	0	1	2
-2	0	1	0	1

$=$

2.78	0.46



Old coordinate system



New coordinate system

Principal Component Analysis

Eigen**vectors**$^{\top}$

-0.81	-0.58
0.58	-0.81

$\times$

$Z^{\top}$

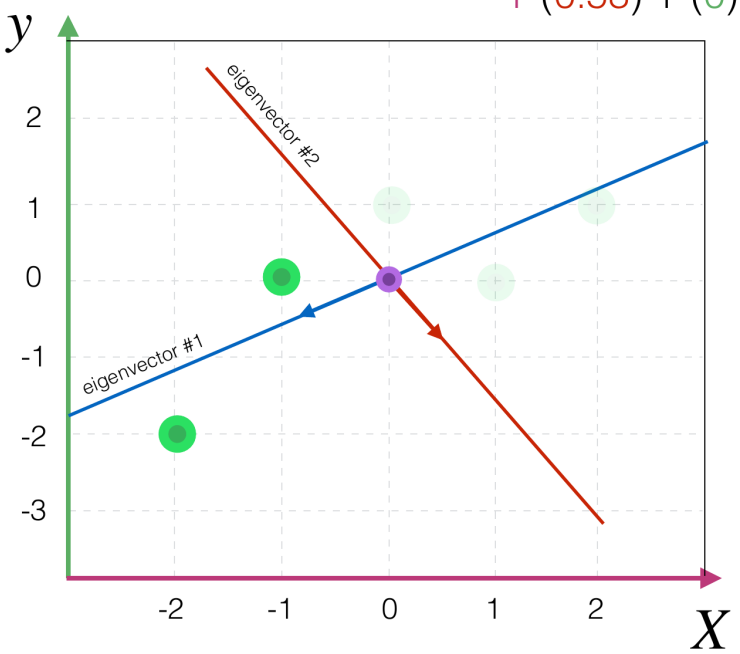
-2	-1	0	1	2
-2	0	1	0	1

$=$

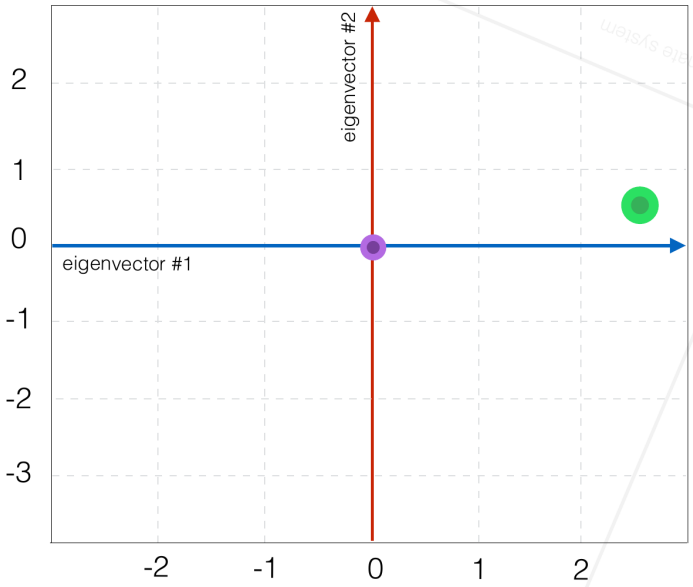
2.78	0.46

-1*(-0.81) + (0)*(-0.58) = **0.81**

-1*(0.58) + (0)*(-0.81) = **-0.58**



Old coordinate system



New coordinate system

Principal Component Analysis

Eigen**vectors**$^{\top}$

-0.81	-0.58
0.58	-0.81

$\times$

$Z^{\top}$

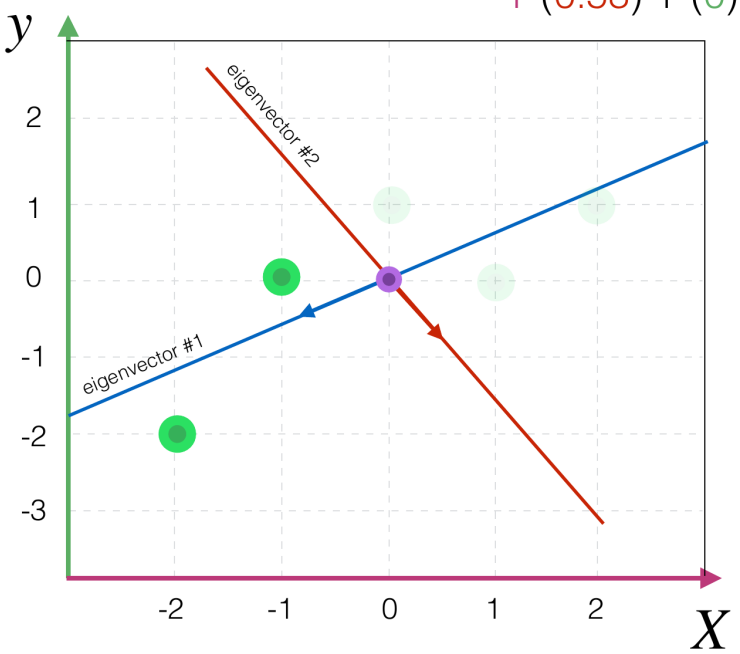
-2	-1	0	1	2
-2	0	1	0	1

$=$

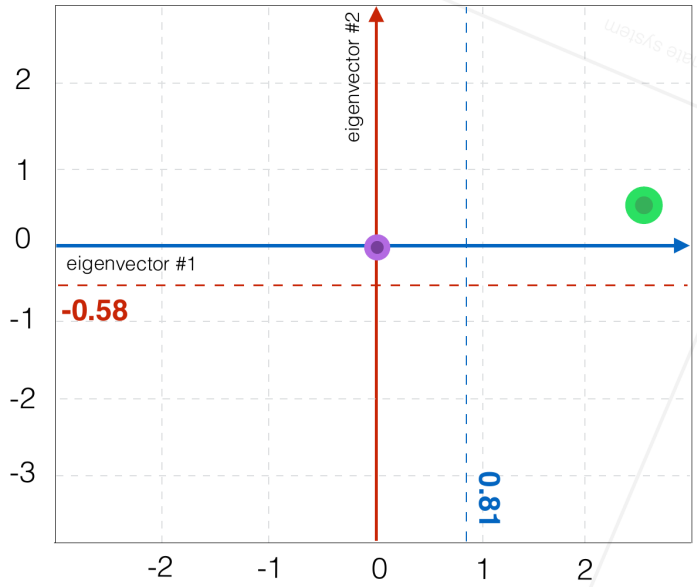
2.78	0.46

-1*(-0.81) + (0)*(-0.58) = **0.81**

-1*(0.58) + (0)*(-0.81) = **-0.58**



Old coordinate system



New coordinate system

Principal Component Analysis

Eigen**vectors**$^{\top}$

-0.81	-0.58
0.58	-0.81

$\times$

$Z^{\top}$

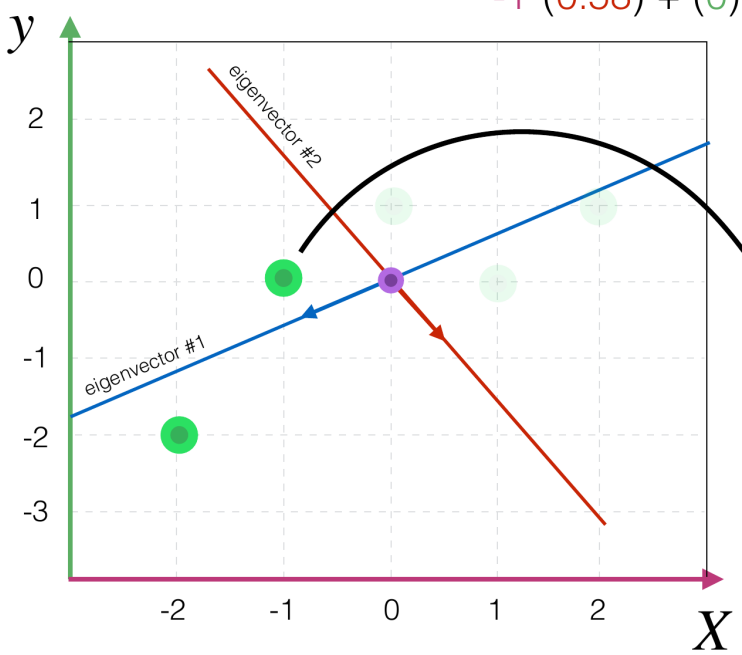
-2	-1	0	1	2
-2	0	1	0	1

$=$

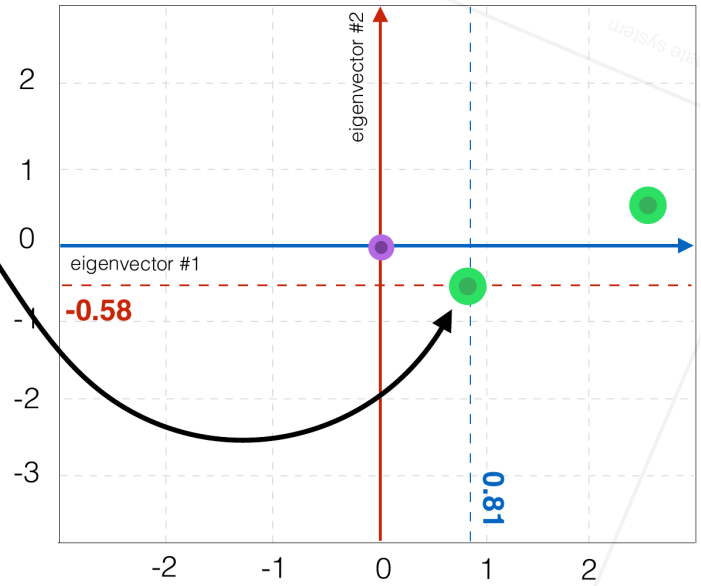
2.78	0.46
0.81	-0.58

-1*(-0.81) + (0)*(-0.58) = **0.81**

-1*(0.58) + (0)*(-0.81) = **-0.58**



Old coordinate system



New coordinate system

Principal Component Analysis

Eigen**vectors**$^{\mathsf{T}}$

-0.81	-0.58
0.58	-0.81

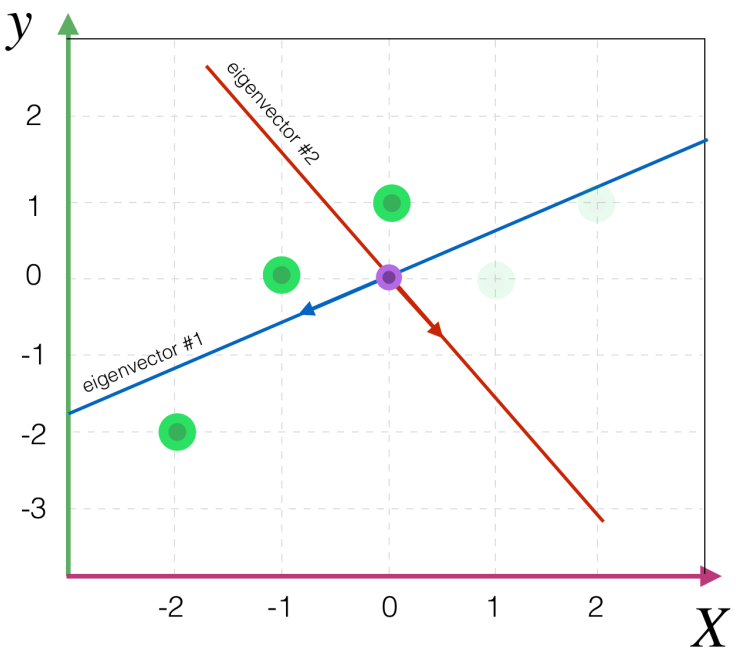
$\times$

Z^{T}

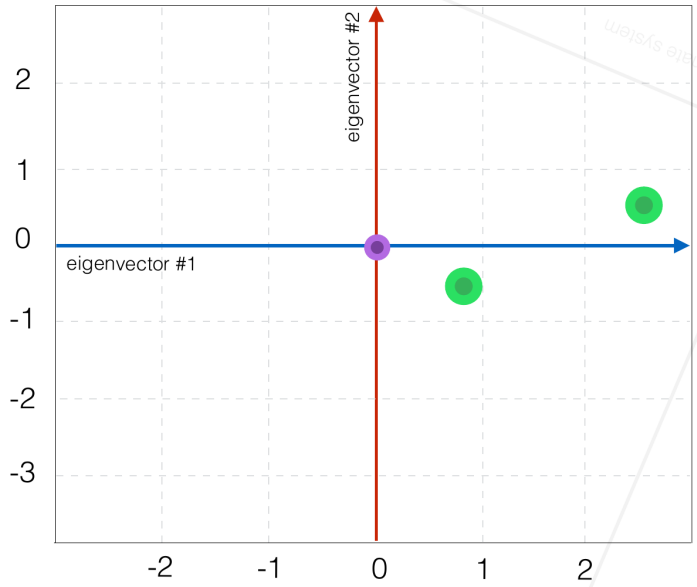
-2	-1	0	1	2
-2	0	1	0	1

$=$

2.78	0.46
0.81	-0.58



Old coordinate system



New coordinate system

Principal Component Analysis

Eigen**vectors**$^{\top}$

-0.81	-0.58
0.58	-0.81

$\times$

$Z^{\top}$

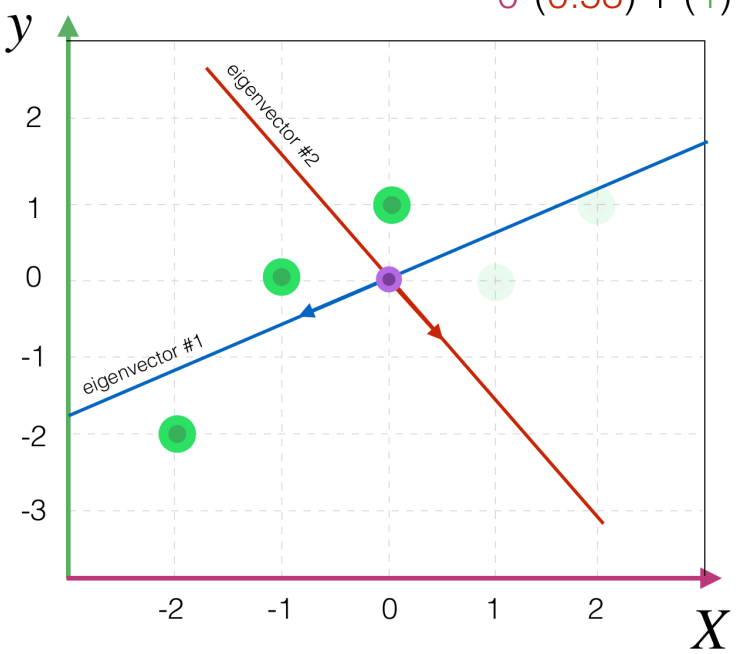
-2	-1	0	1	2
-2	0	1	0	1

$=$

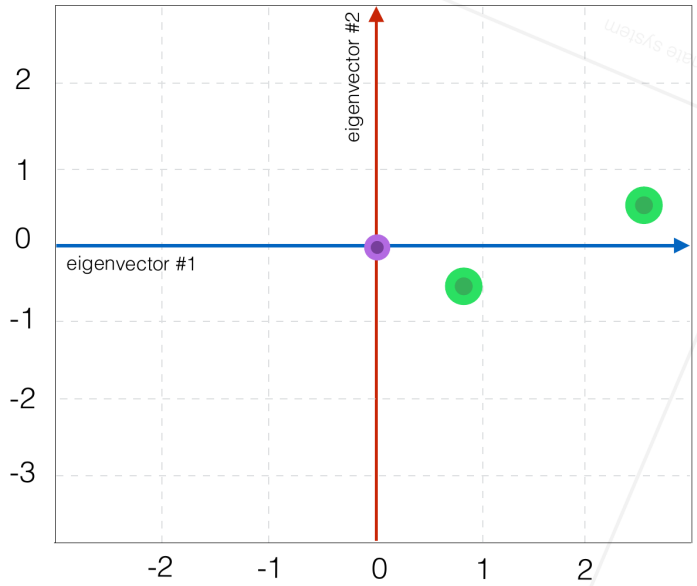
2.78	0.46
0.81	-0.58

0*(-0.81) + (1)*(-0.58) = **-0.58**

0*(0.58) + (1)*(-0.81) = **-0.81**



Old coordinate system



New coordinate system

Principal Component Analysis

Eigen**vectors**$^{\top}$

-0.81	-0.58
0.58	-0.81

$\times$

$Z^{\top}$

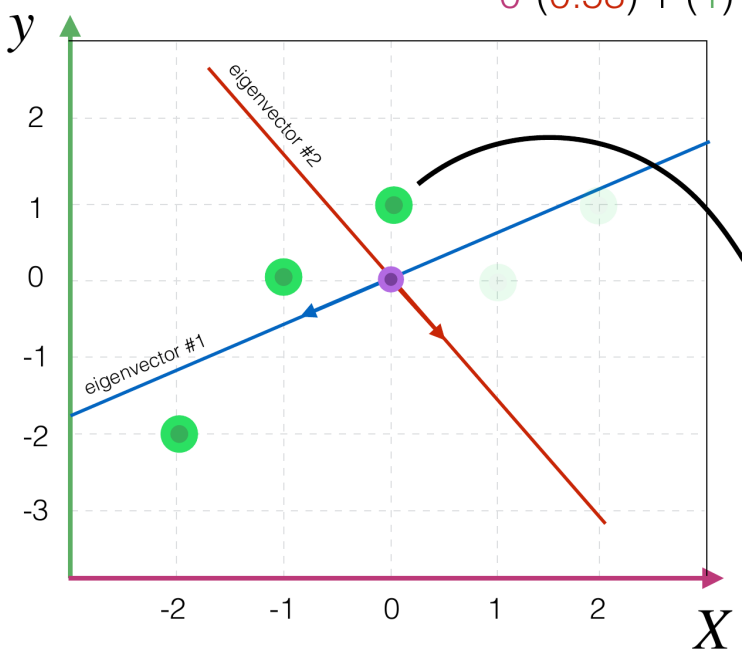
-2	-1	0	1	2
-2	0	1	0	1

$=$

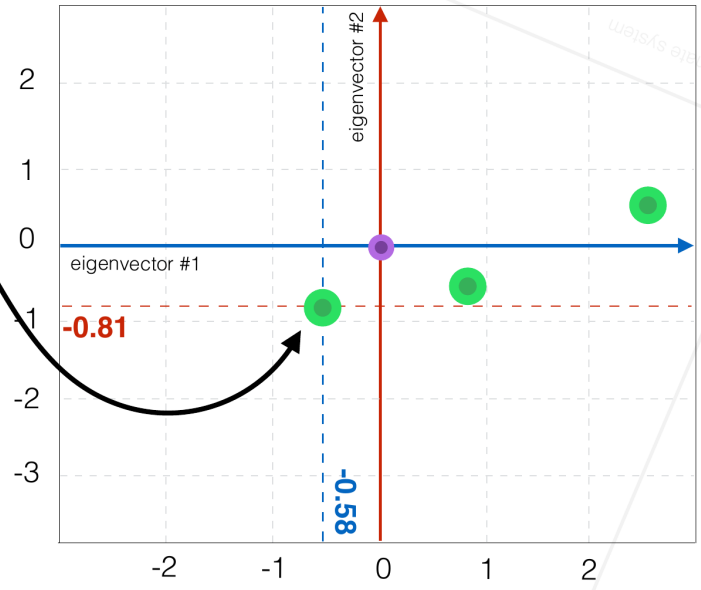
2.78	0.46
0.81	-0.58
-0.58	-0.81

0*(-0.81) + (1)*(-0.58) = **-0.58**

0*(0.58) + (1)*(-0.81) = **-0.81**



Old coordinate system



New coordinate system

Principal Component Analysis

Eigen**vectors**$^{\mathsf{T}}$

-0.81	-0.58
0.58	-0.81

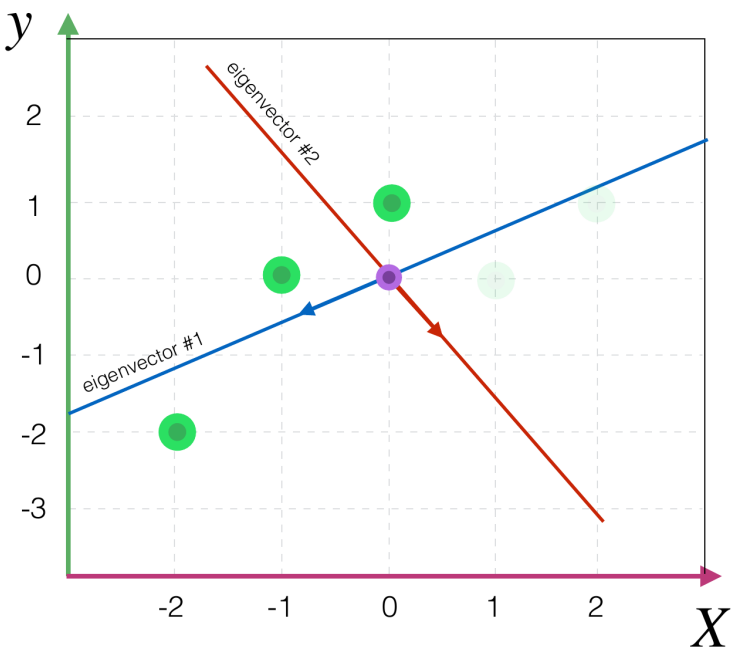
$\times$

Z^{T}

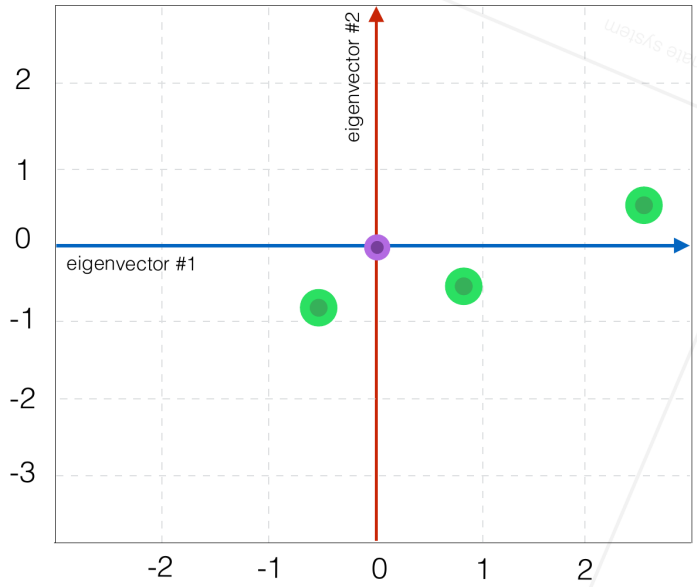
-2	-1	0	1	2
-2	0	1	0	1

$=$

2.78	0.46
0.81	-0.58
-0.58	-0.81



Old coordinate system



New coordinate system

Principal Component Analysis

Eigen**vectors**$^{\mathsf{T}}$

-0.81	-0.58
0.58	-0.81

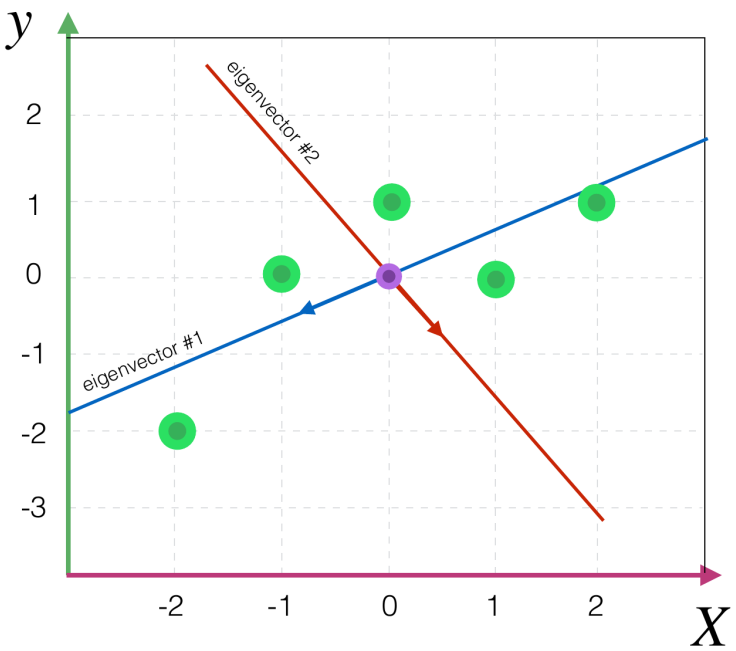
$\times$

Z^{T}

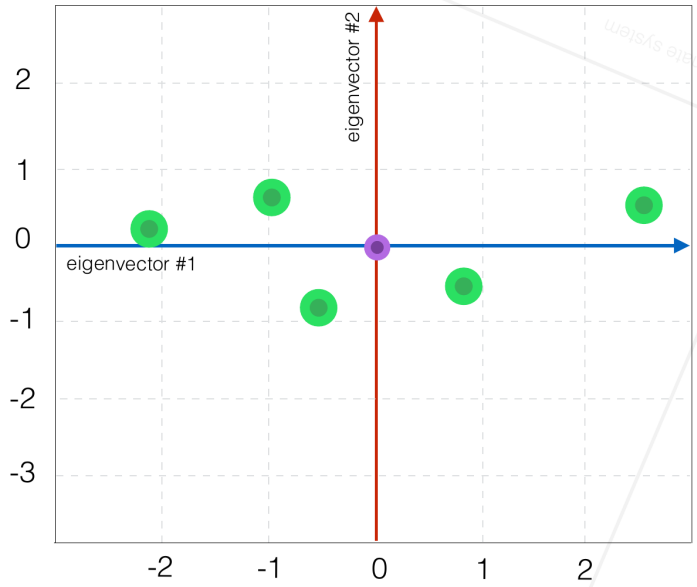
-2	-1	0	1	2
-2	0	1	0	1

$=$

2.78	0.46
0.81	-0.58
-0.58	-0.81
-0.81	0.58
-2.2	0.35



Old coordinate system



New coordinate system

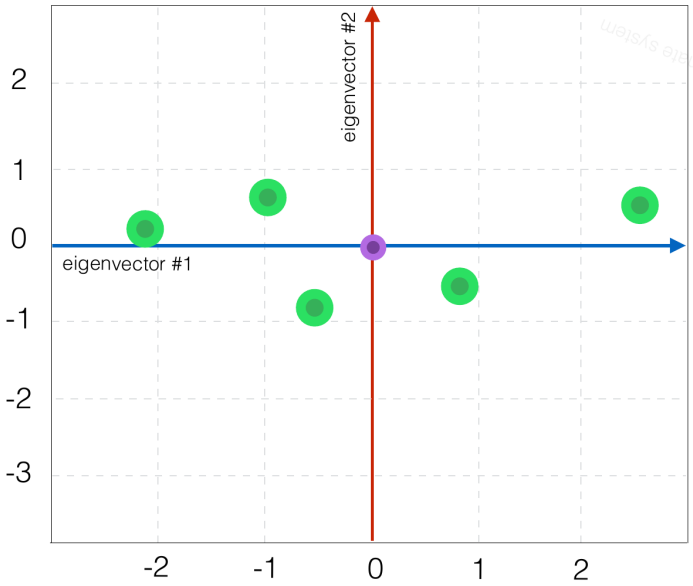
Principal Component Analysis

These eigenvectors are called **P**rinciple **C**omponents

2.78	0.46
0.81	-0.58
-0.58	-0.81
-0.81	0.58
-2.2	0.35

eigenvector #1 is called **PC1**

eigenvector #2 is called **PC2**



New coordinate system

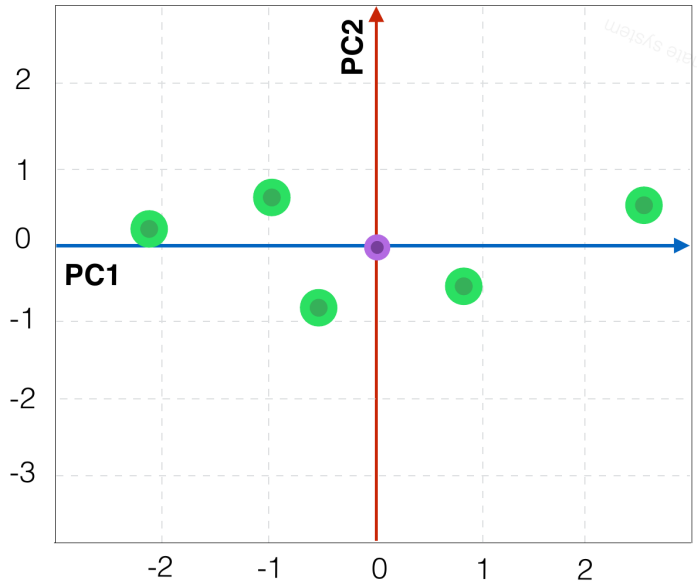
Principal Component Analysis

These eigenvectors are called **P**rinciple **C**omponents

2.78	0.46
0.81	-0.58
-0.58	-0.81
-0.81	0.58
-2.2	0.35

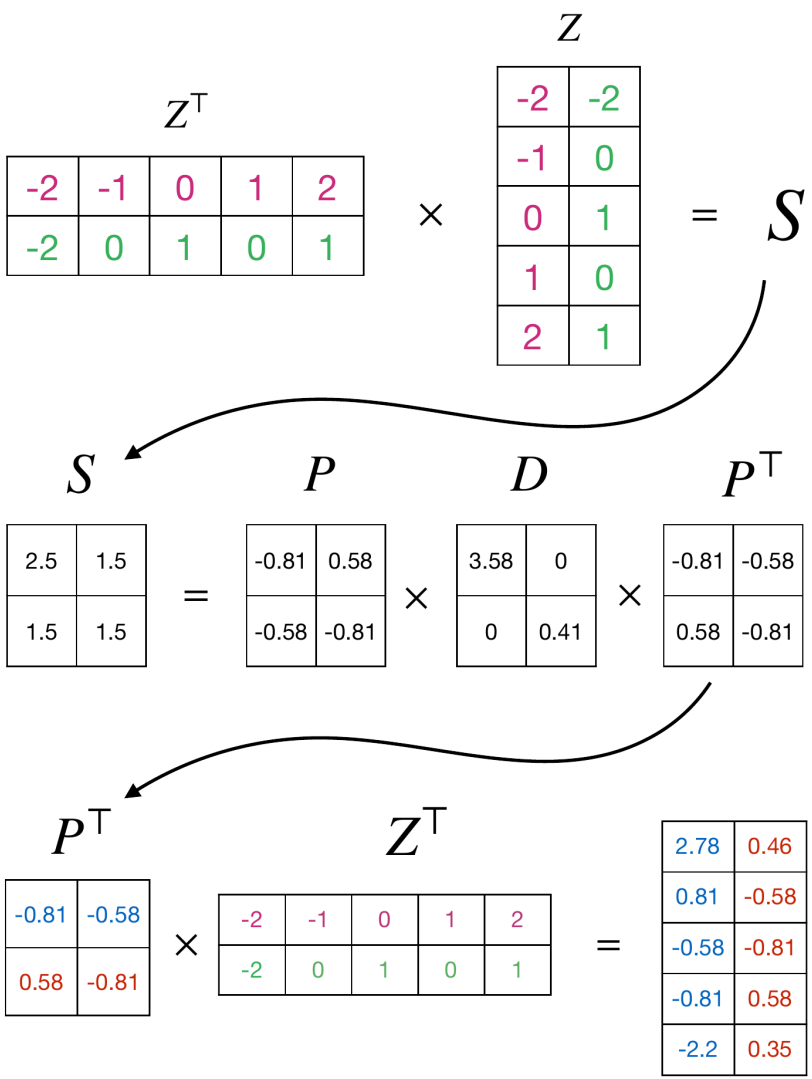
eigenvector #1 is called **PC1**

eigenvector #2 is called **PC2**



New coordinate system

Principal Component Analysis



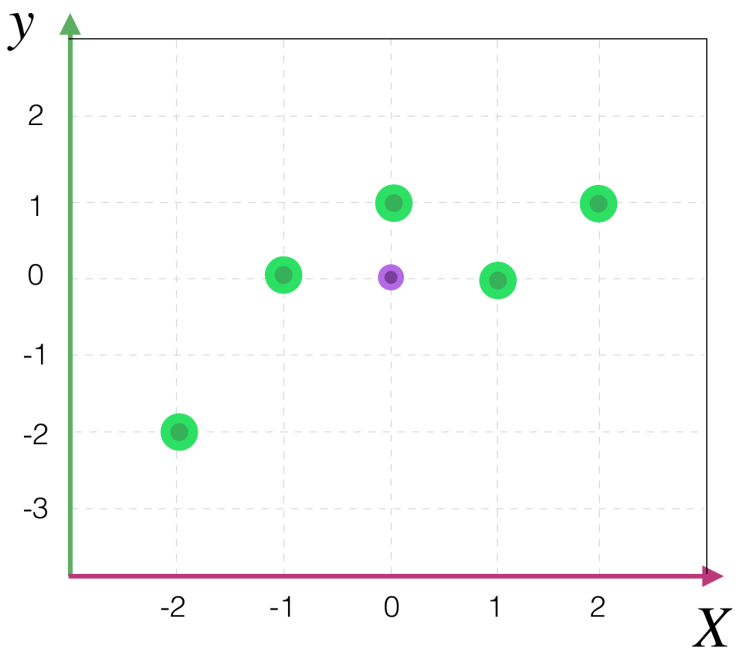
Compute **covariance** matrix

Perform **eigen**decomposition

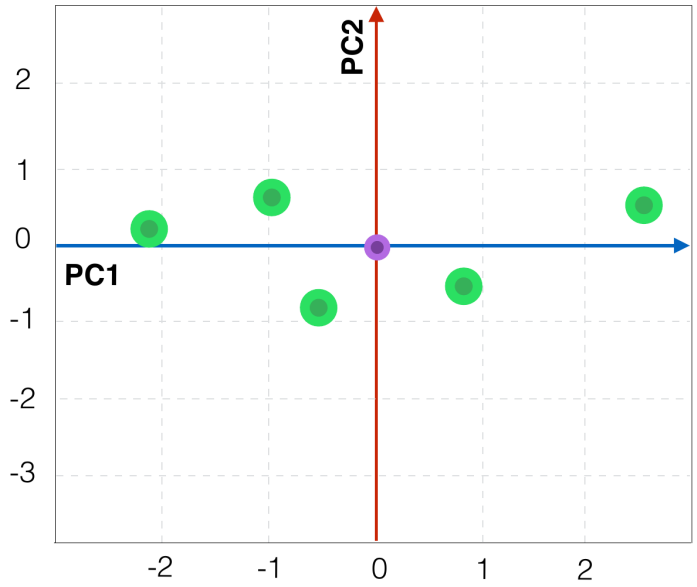
Compute new **coordinates**

Principal Component Analysis

Do you still remember what was it all about?



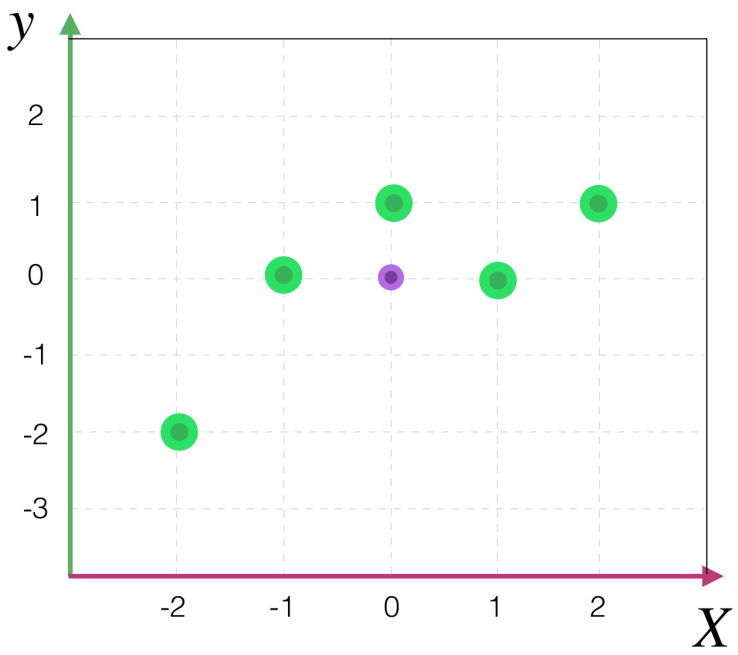
Old coordinate system



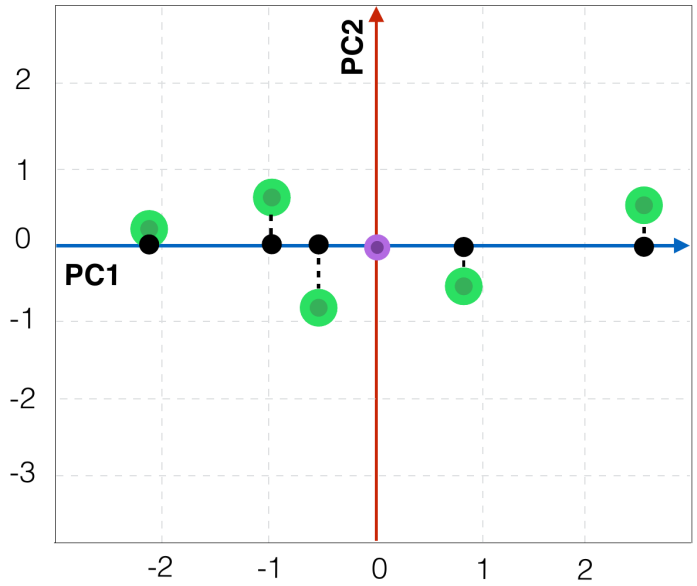
New coordinate system

Principal Component Analysis

Do you still remember what was it all about?
We want to **reduce the dimensionality**!



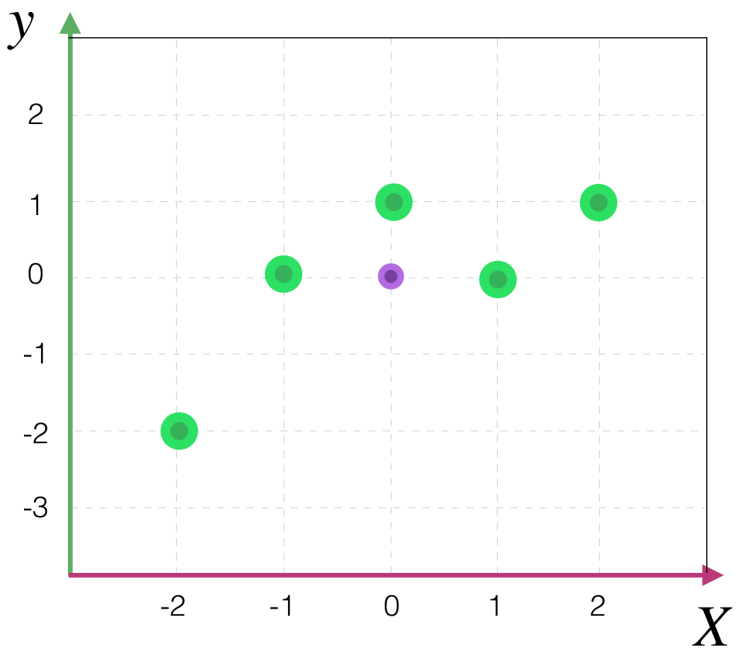
Old coordinate system



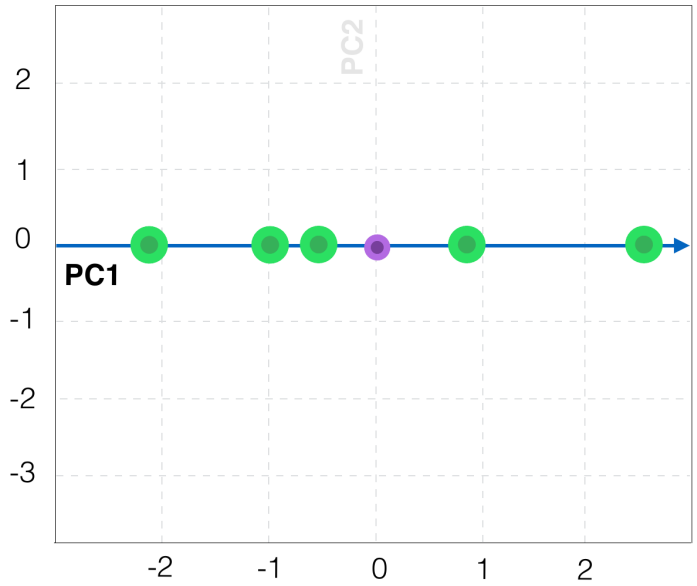
New coordinate system

Principal Component Analysis

Do you still remember what was it all about?
We want to **reduce the dimensionality**!



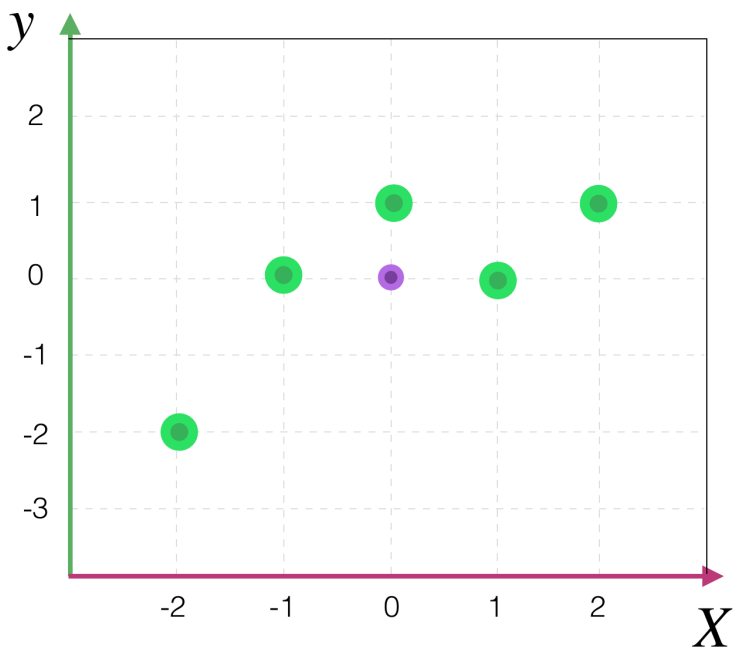
Old coordinate system



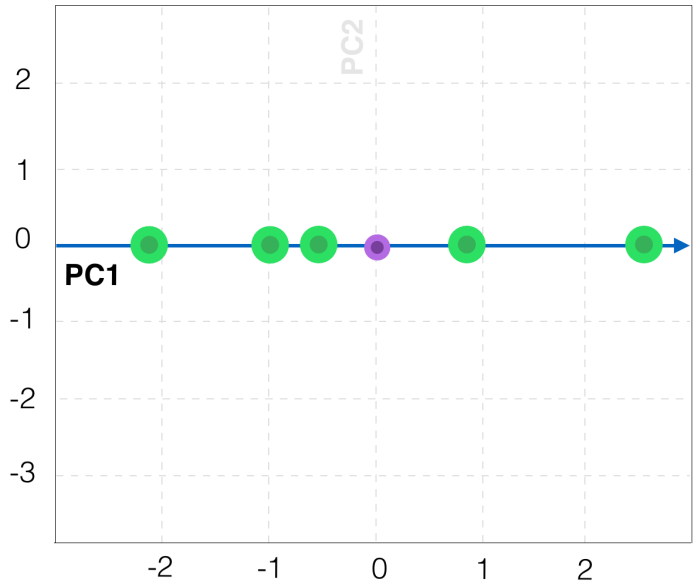
New coordinate system

Principal Component Analysis

We can **ignore** the **second eigenvector**
because it does not contain much information



Old coordinate system

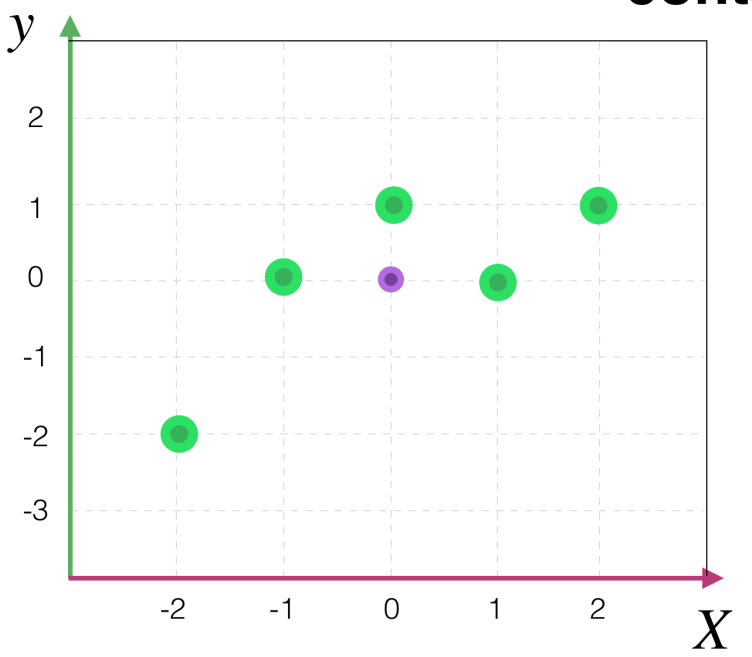


New coordinate system

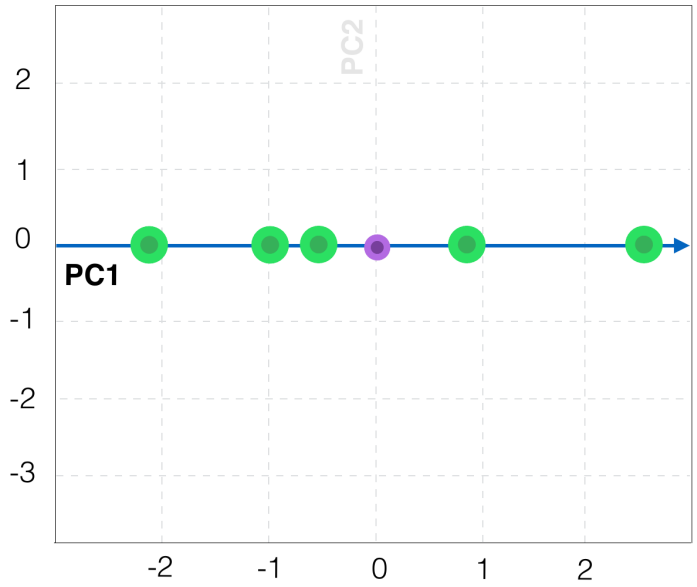
Principal Component Analysis

We can **ignore** the **second eigenvector** because it does not contain much information

How much information **the second eigenvector contains**?

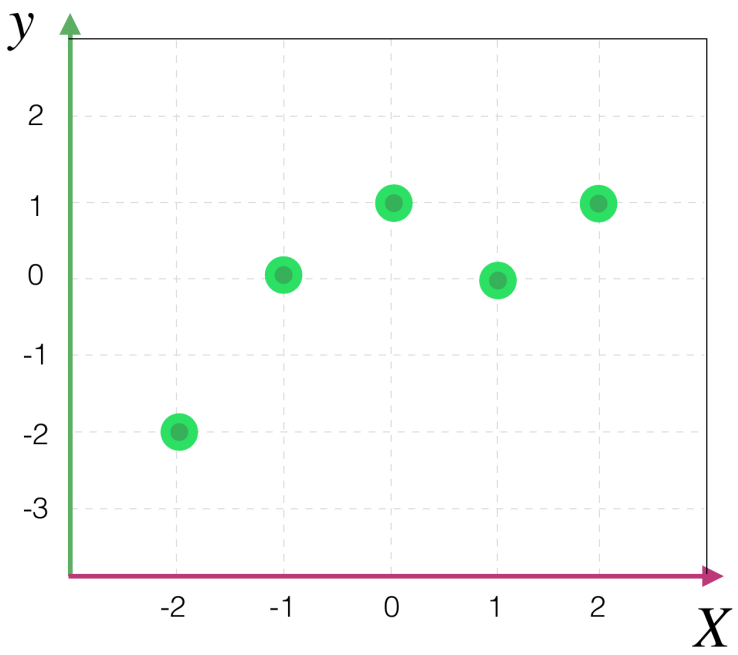


Old coordinate system

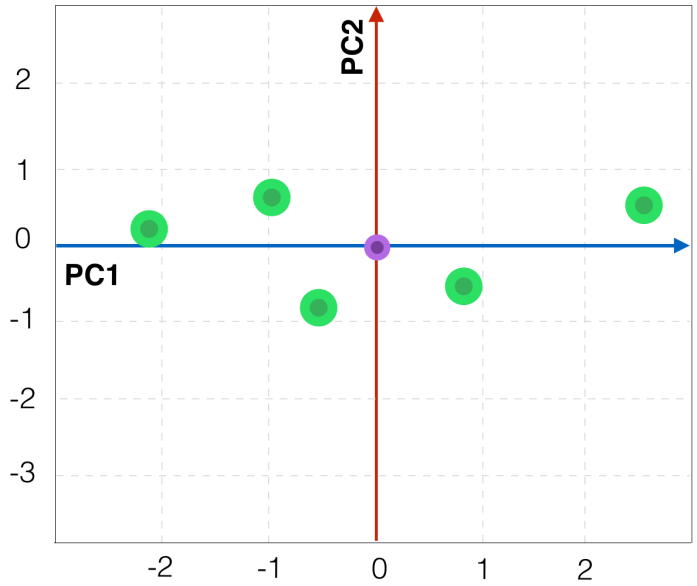


New coordinate system

Principal Component Analysis



Old coordinate system



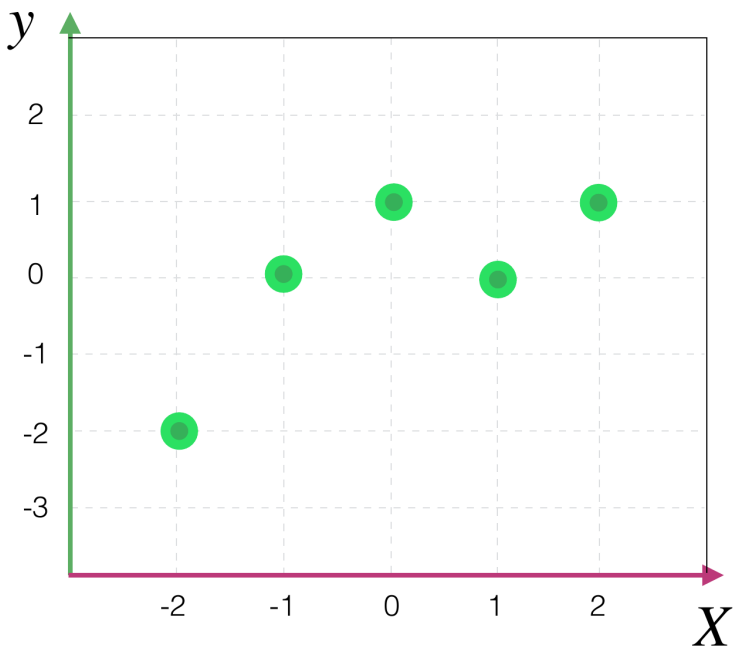
New coordinate system

Principal Component Analysis

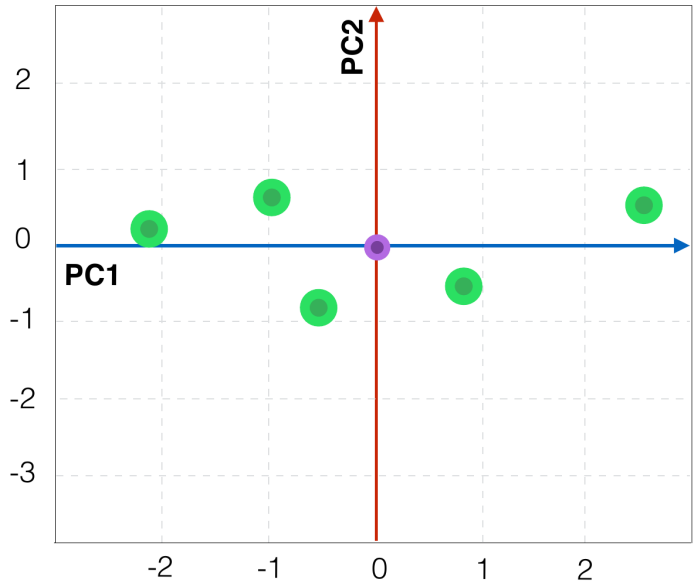
D

3.58	0
0	0.42

Eigen**values**



Old coordinate system



New coordinate system

Principal Component Analysis

S

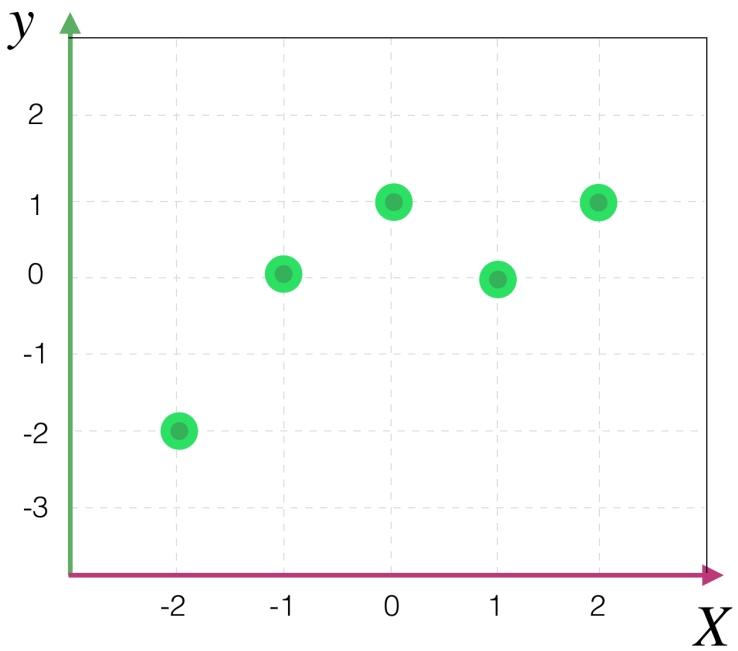
2.5	1.5
1.5	1.5

Covariance matrix

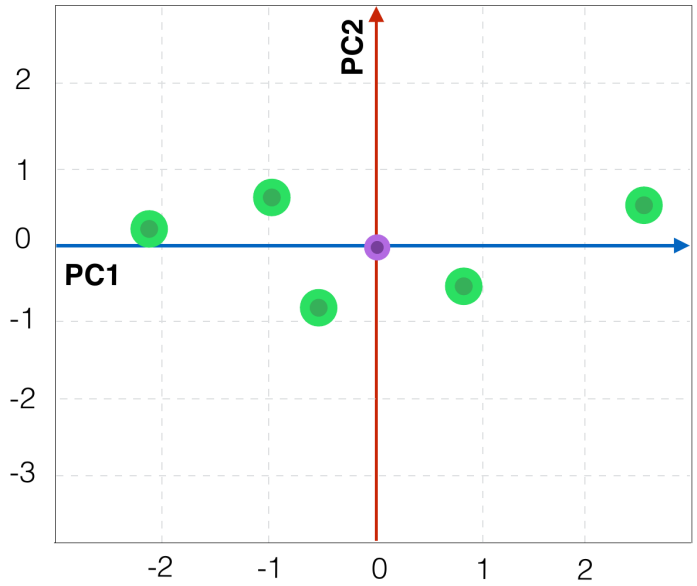
D

3.58	0
0	0.42

Eigen**values**

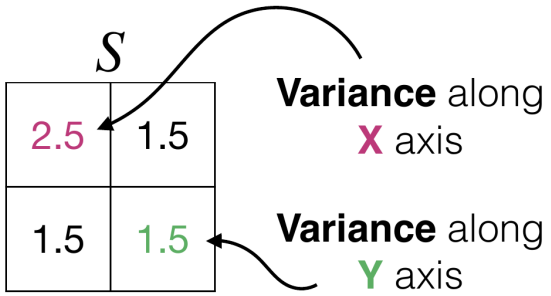


Old coordinate system



New coordinate system

Principal Component Analysis

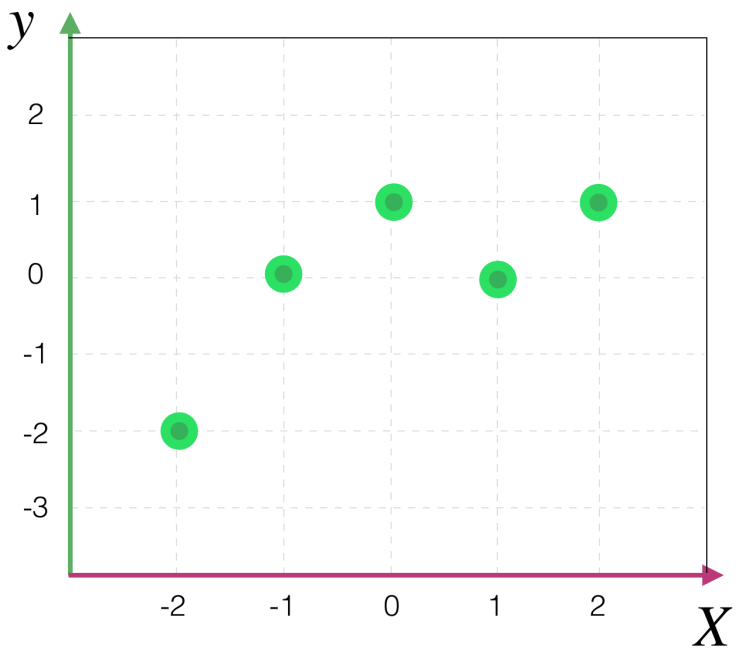


Covariance matrix

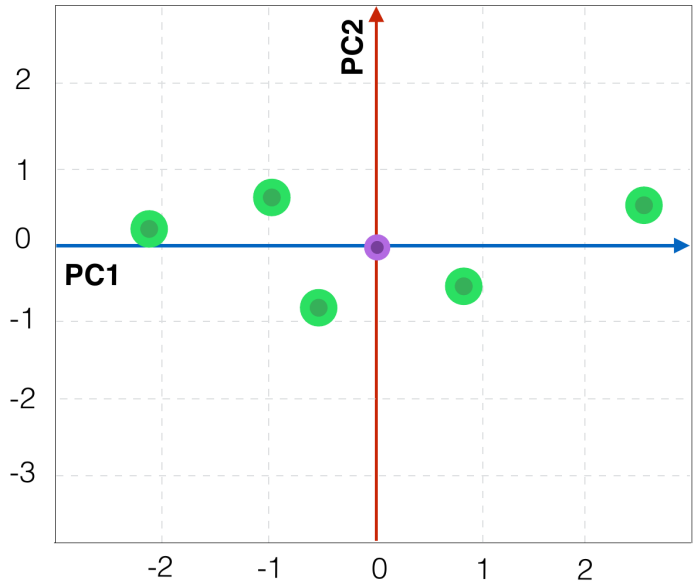
D

3.58	0
0	0.42

Eigen**values**

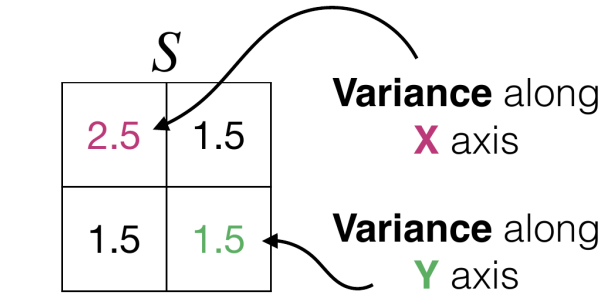


Old coordinate system



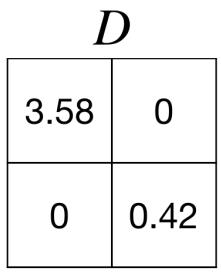
New coordinate system

Principal Component Analysis



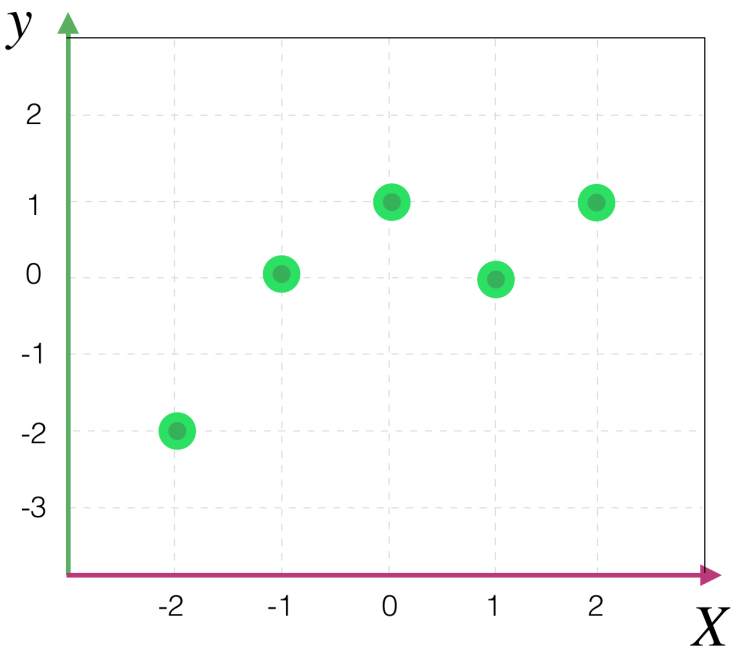
Covariance matrix

Variance along **?**

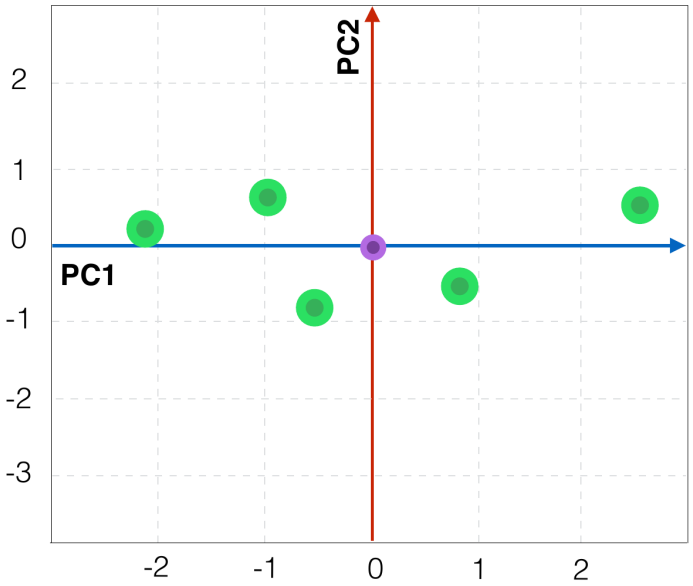


Variance along **?**

Eigen**values**

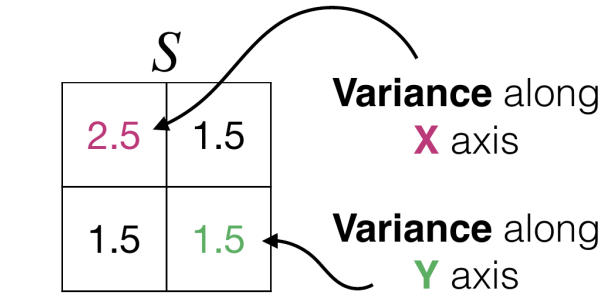


Old coordinate system



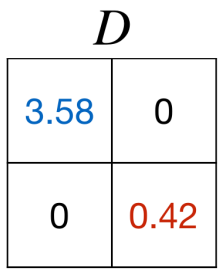
New coordinate system

Principal Component Analysis



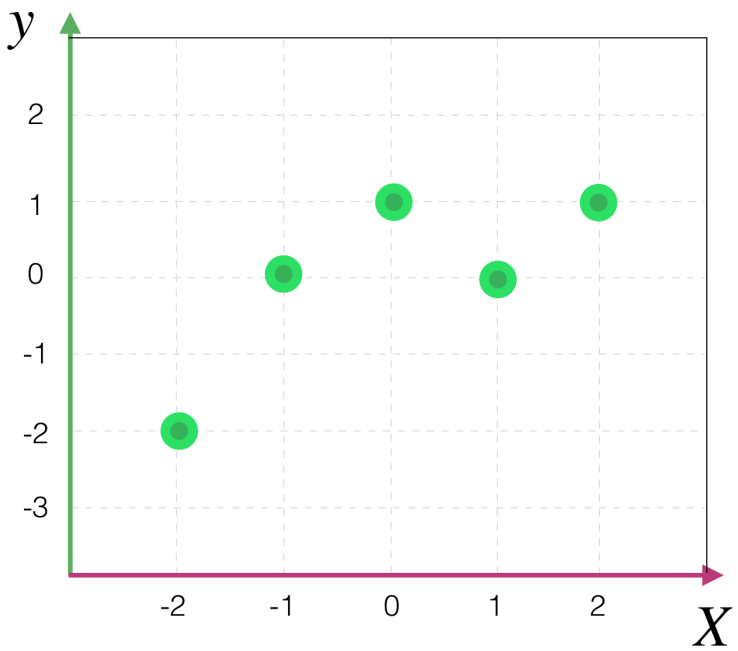
Covariance matrix

Variance along **PC1**

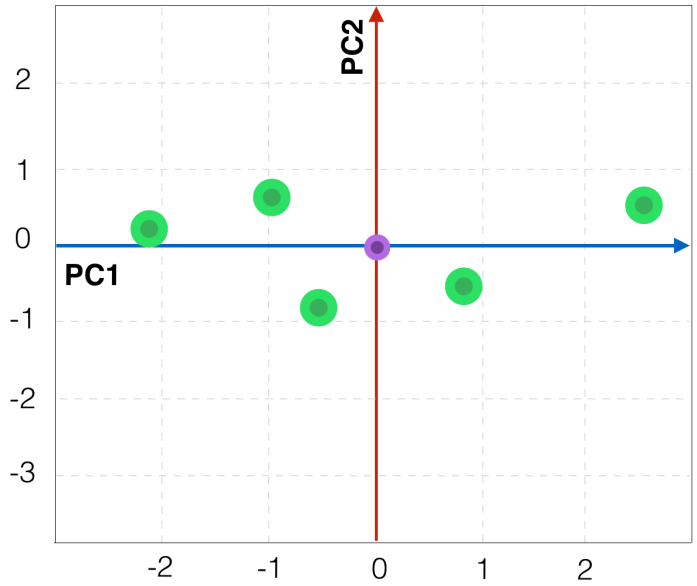


Variance along **PC2**

Eigen**values**

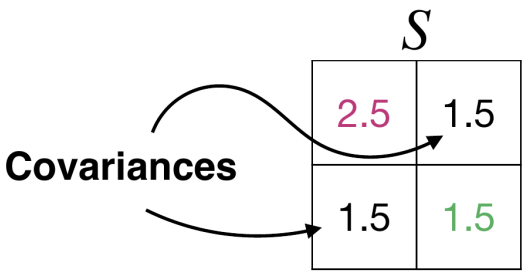


Old coordinate system

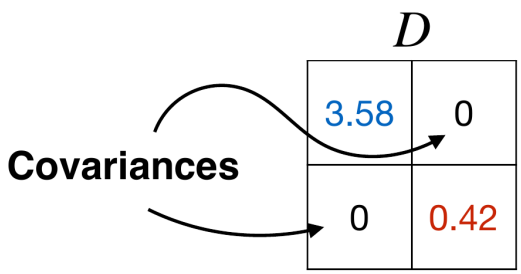


New coordinate system

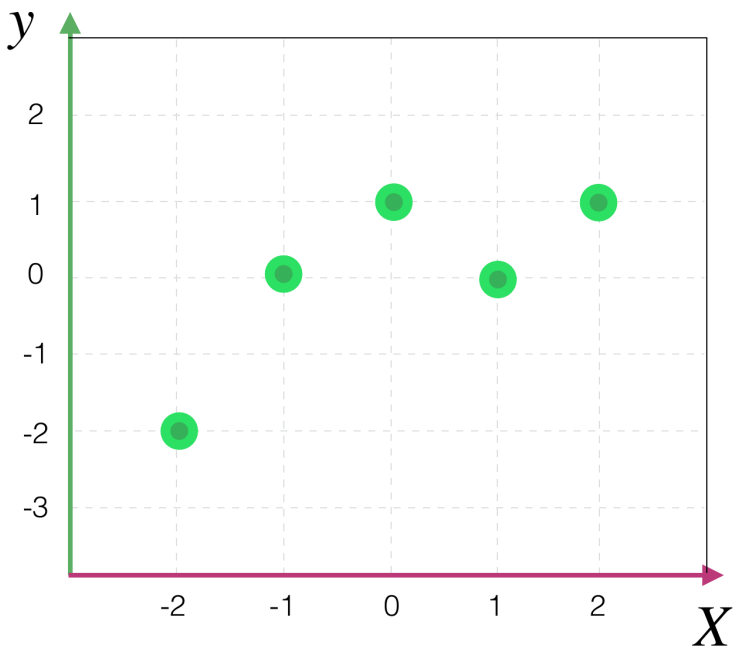
Principal Component Analysis



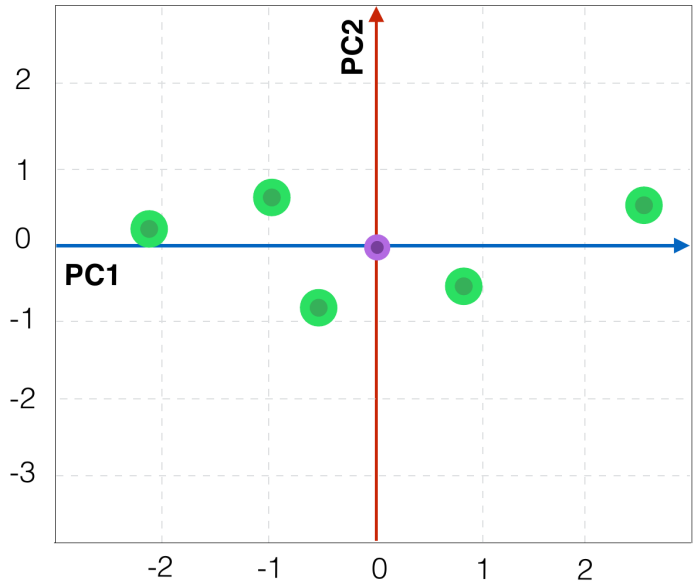
Covariance matrix



Eigen**values**



Old coordinate system



New coordinate system

Principal Component Analysis

S

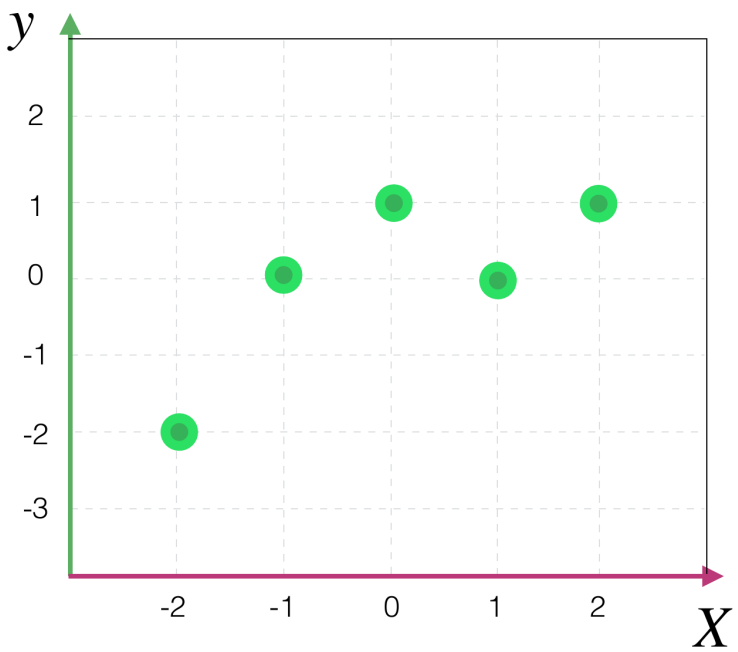
2.5	1.5
1.5	1.5

Covariance matrix

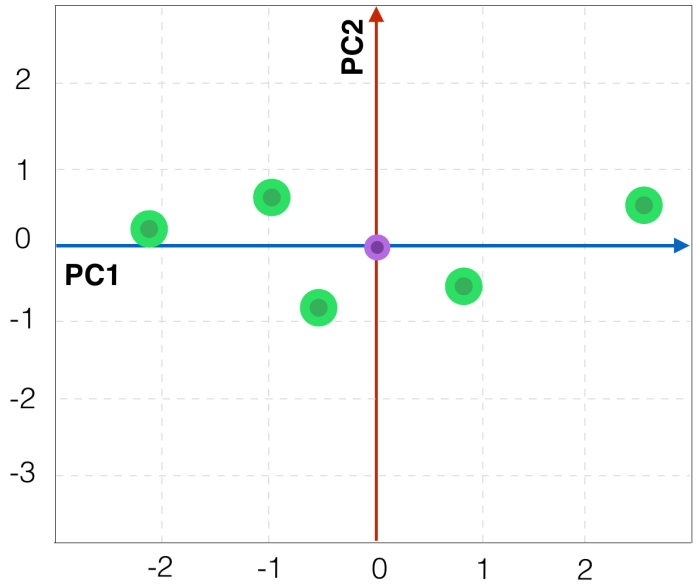
D

3.58	0
0	0.42

New **covariance** matrix



Old coordinate system



New coordinate system

Principal Component Analysis

S

2.5	1.5
1.5	1.5

Covariance matrix

D

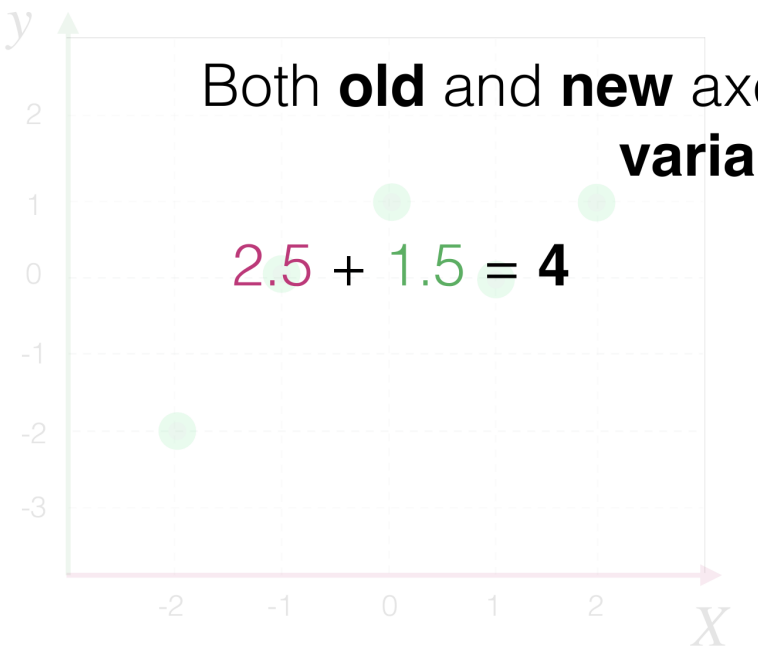
3.58	0
0	0.42

New **covariance** matrix

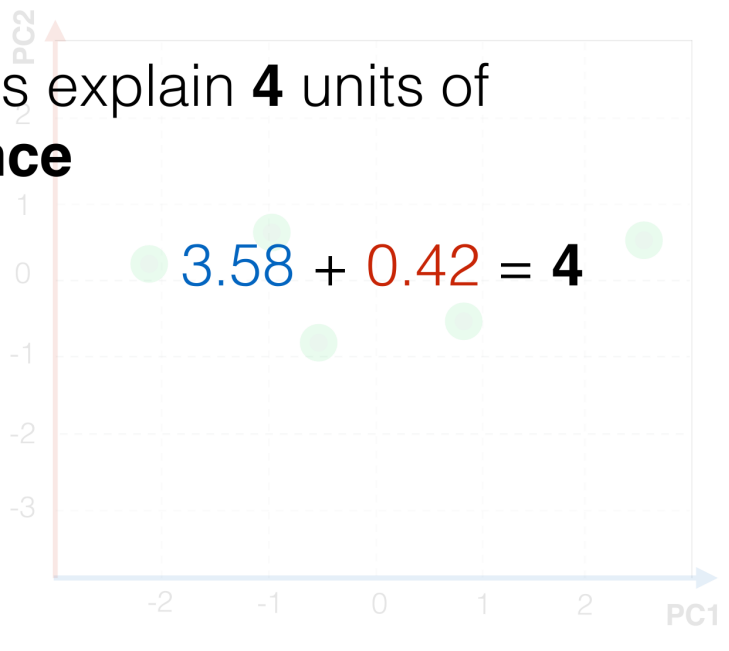
Both **old** and **new** axes explain **4** units of **variance**

$2.5 + 1.5 = \mathbf{4}$

$3.58 + 0.42 = \mathbf{4}$



Old coordinate system



New coordinate system

Principal Component Analysis

S

2.5	1.5
1.5	1.5

Covariance matrix

D

3.58	0
0	0.42

New **covariance** matrix

Both **old** and **new** axes explain **4** units of **variance**

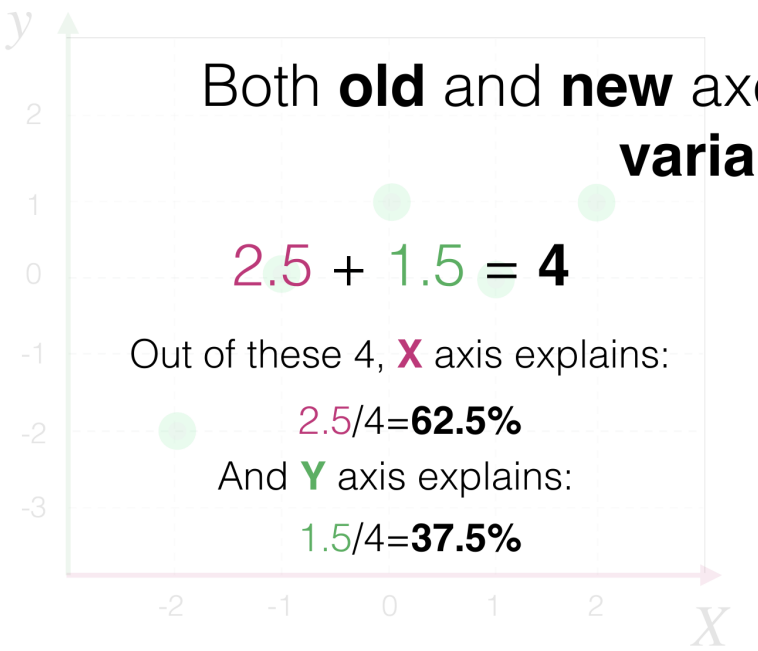
2.5 + 1.5 = **4**

Out of these 4, **X** axis explains:

2.5/4=**62.5%**

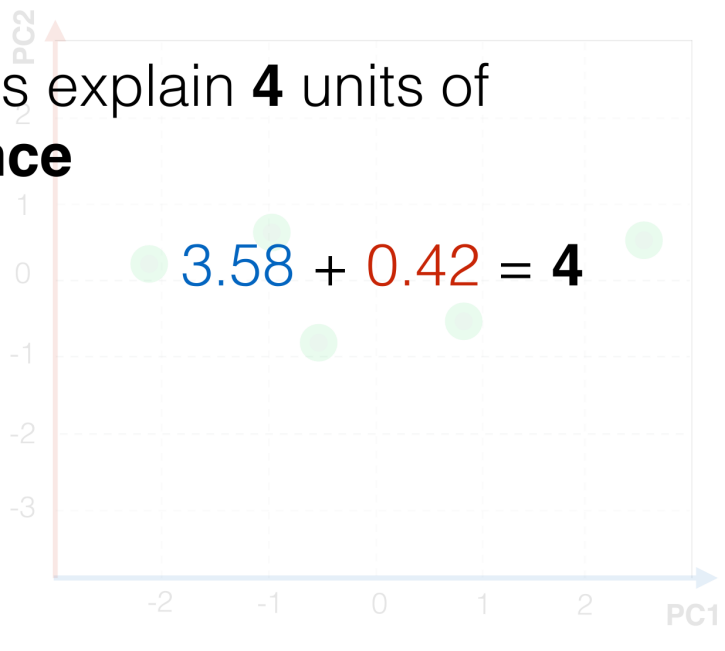
And **Y** axis explains:

1.5/4=**37.5%**



Old coordinate system

3.58 + 0.42 = **4**



New coordinate system

Principal Component Analysis

S

2.5	1.5
1.5	1.5

Covariance matrix

D

3.58	0
0	0.42

New **covariance** matrix

Both **old** and **new** axes explain **4** units of **variance**

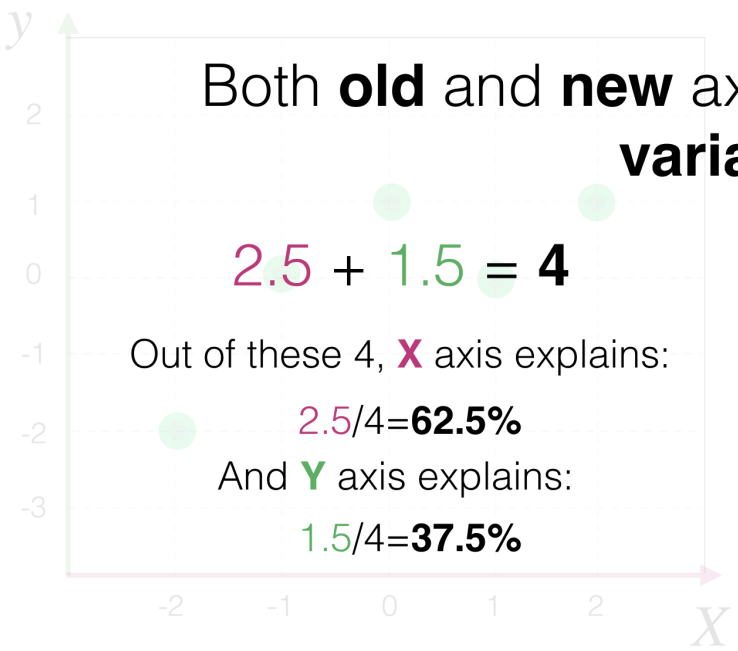
2.5 + 1.5 = **4**

Out of these 4, **X** axis explains:

2.5/4=**62.5%**

And **Y** axis explains:

1.5/4=**37.5%**



Old coordinate system

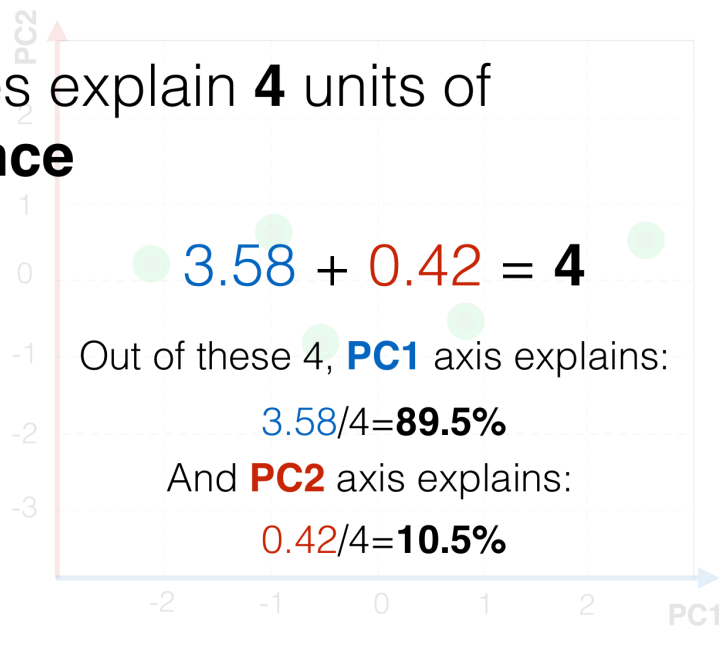
3.58 + 0.42 = **4**

Out of these 4, **PC1** axis explains:

3.58/4=**89.5%**

And **PC2** axis explains:

0.42/4=**10.5%**

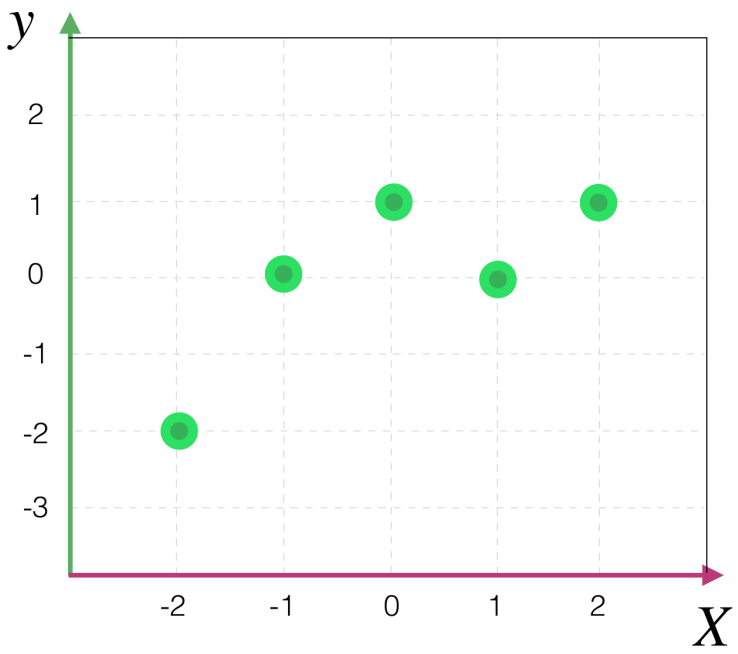


New coordinate system

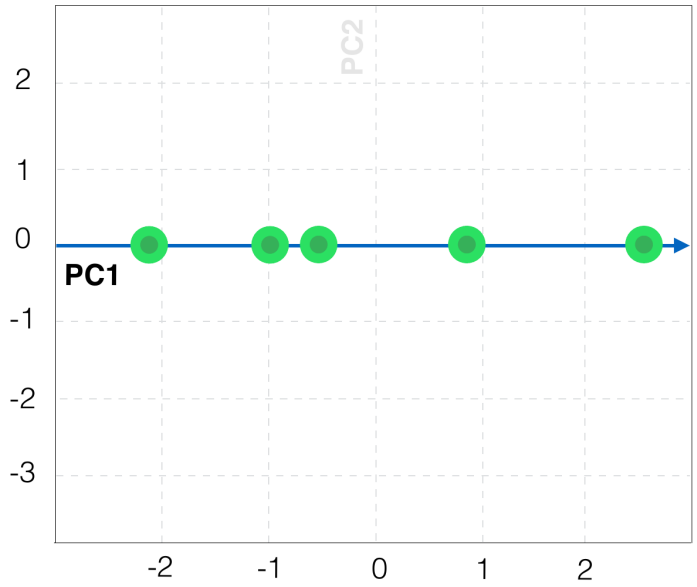
Principal Component Analysis

We can **ignore** the **second PC** because it does not contain much information

How much information **the second PC contains**?



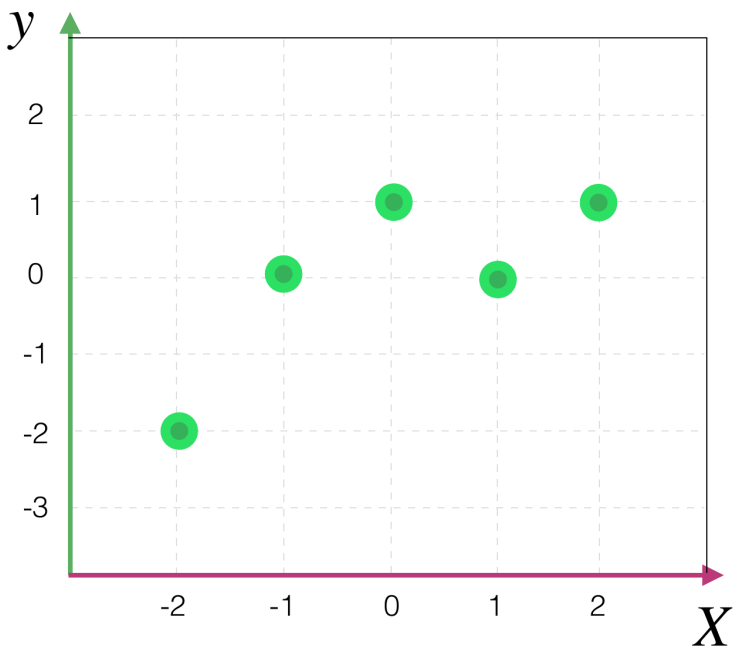
Old coordinate system



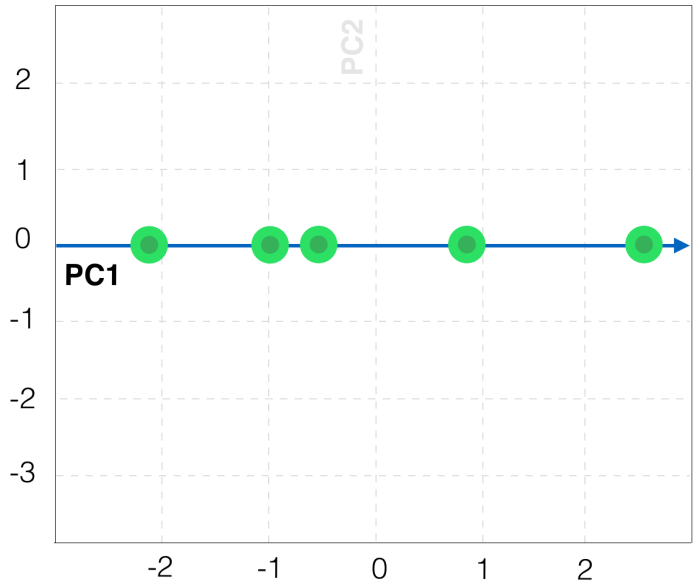
New coordinate system

Principal Component Analysis

We can **ignore** the **second PC** because it explains only **10.5%** of variation



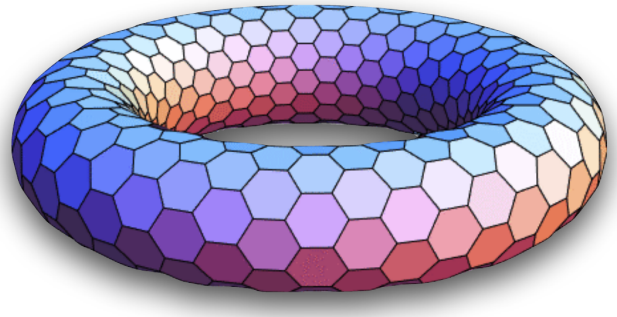
Old coordinate system

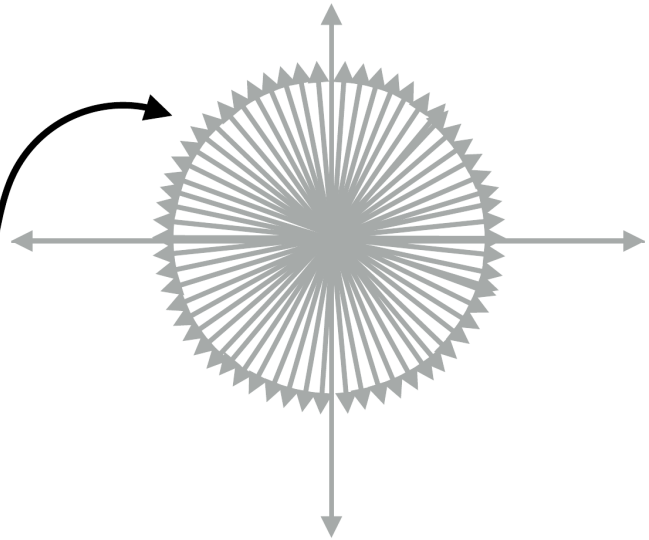


New coordinate system

Principal Component Analysis

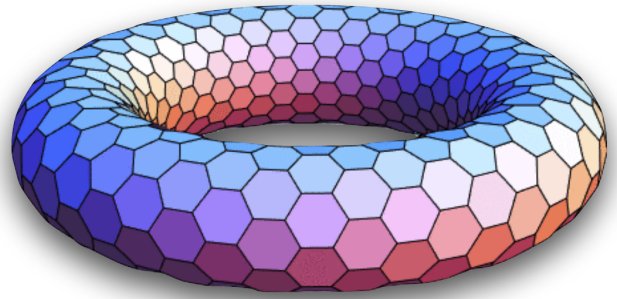
How many PCs will be formed in **200D** space?

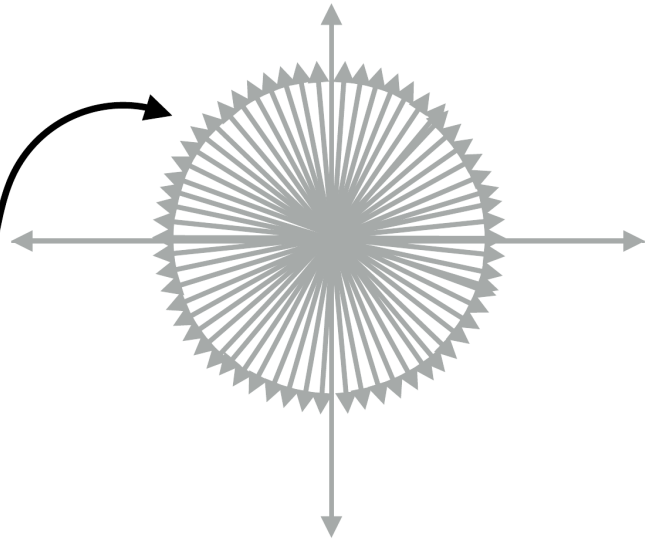




Principal Component Analysis

How many PCs will be formed in **200D** space?

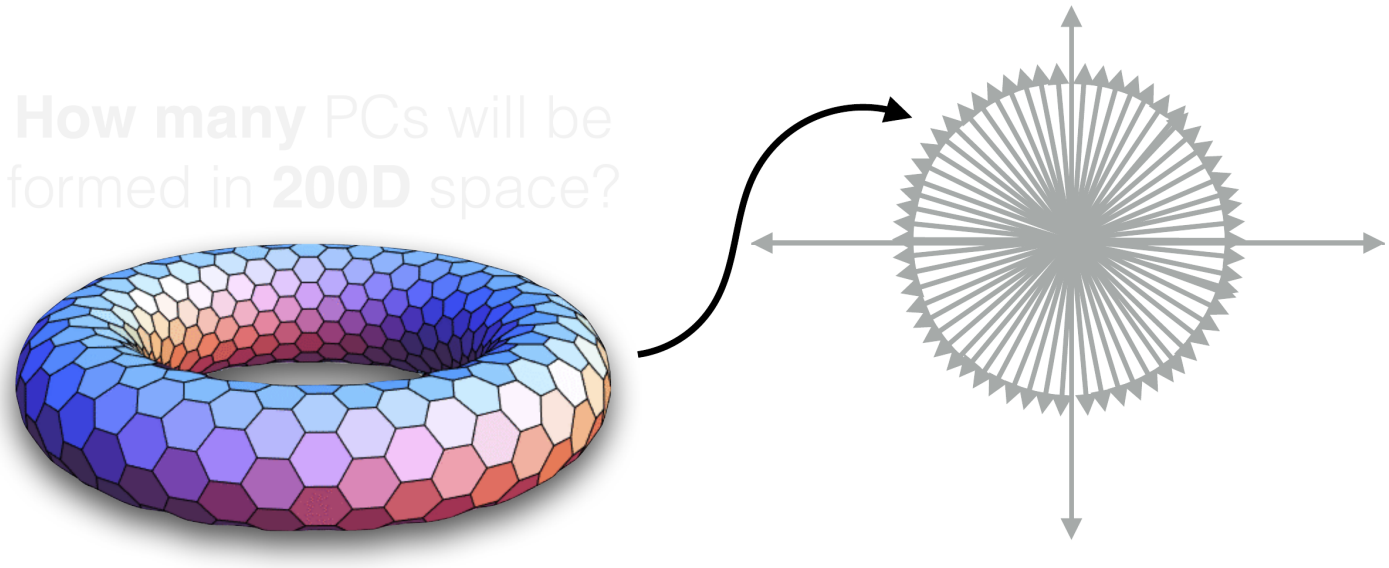




No exceptions, **200 PCs**

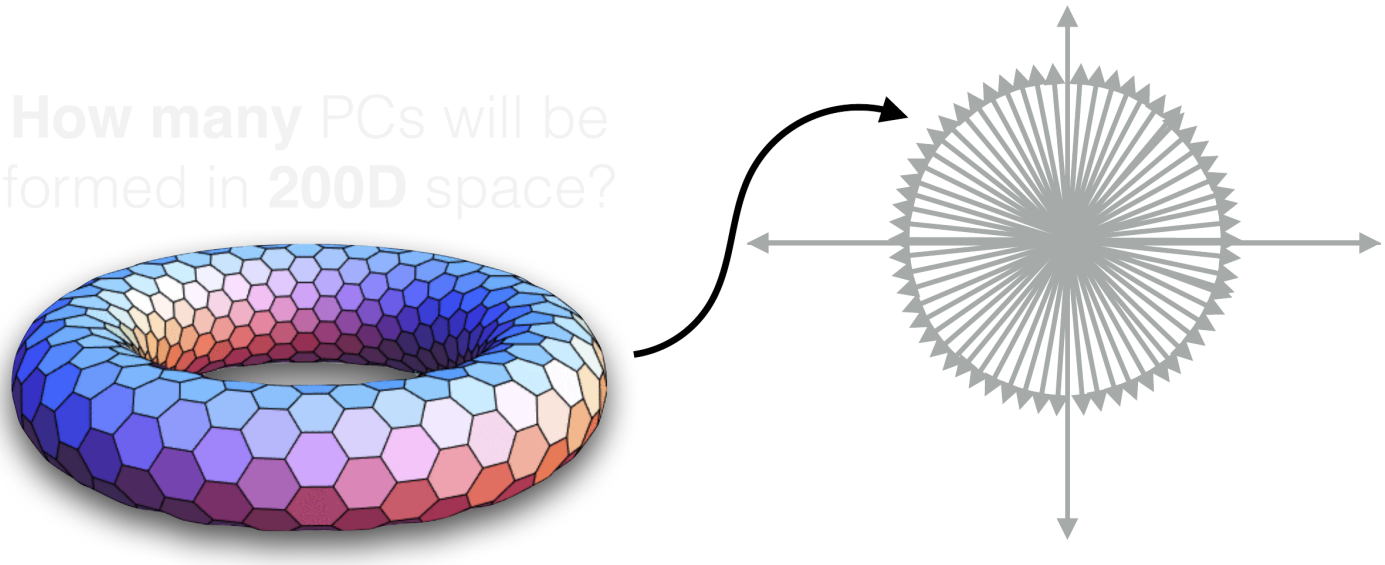
Principal Component Analysis

How many PCs will be formed in **200D** space?



No exceptions, **200 PCs**

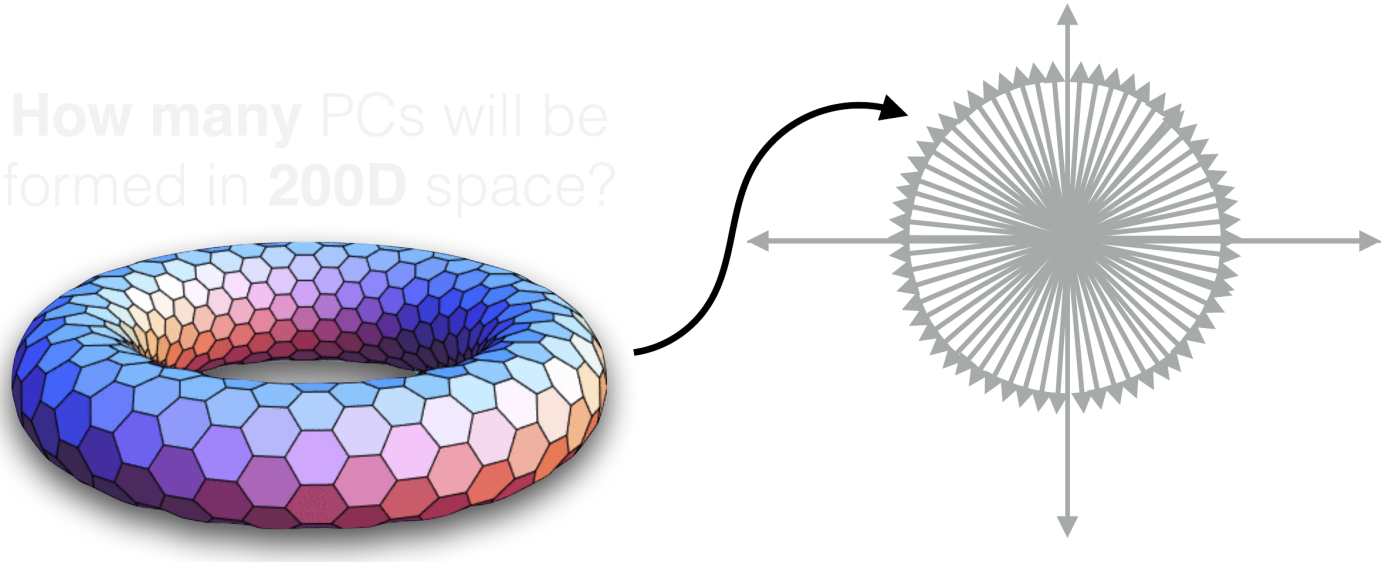
Principal Component Analysis



How many PCs should we keep?

Principal Component Analysis

Variance explained is a good criteria for choosing the total number of PCs to keep

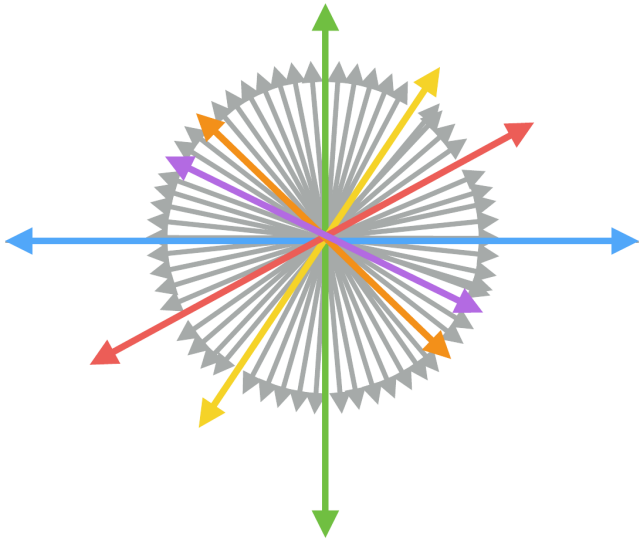


How many PCs should we keep?

Principal Component Analysis

Variance explained is a good criteria for choosing the total number of PCs to keep

You should keep as many PCs as it takes to explain **90%** of **total variance**

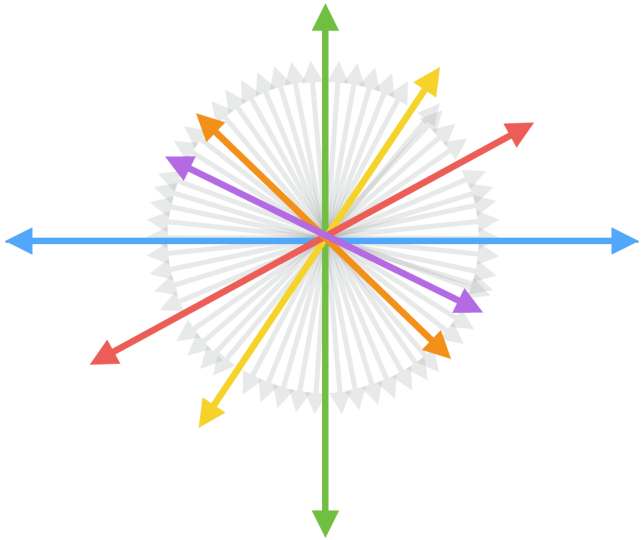


How many PCs should we keep?

Principal Component Analysis

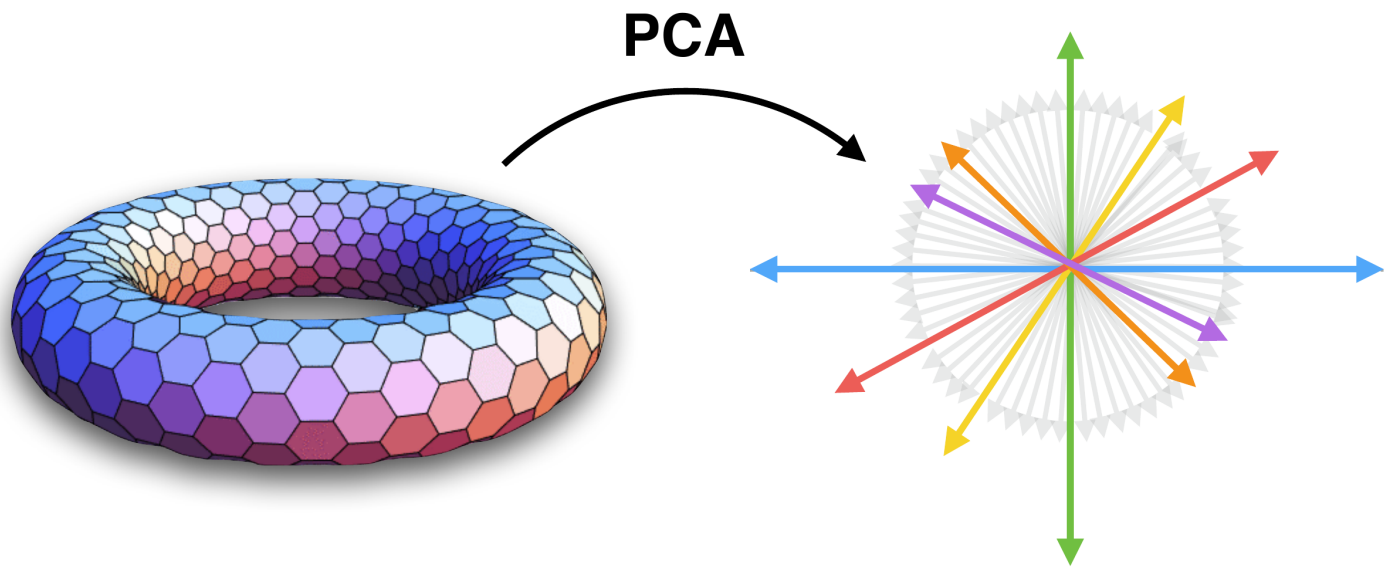
Variance explained is a good criteria for choosing the total number of PCs to keep

You should keep as many PCs as it takes to explain **90%** of **total variance**



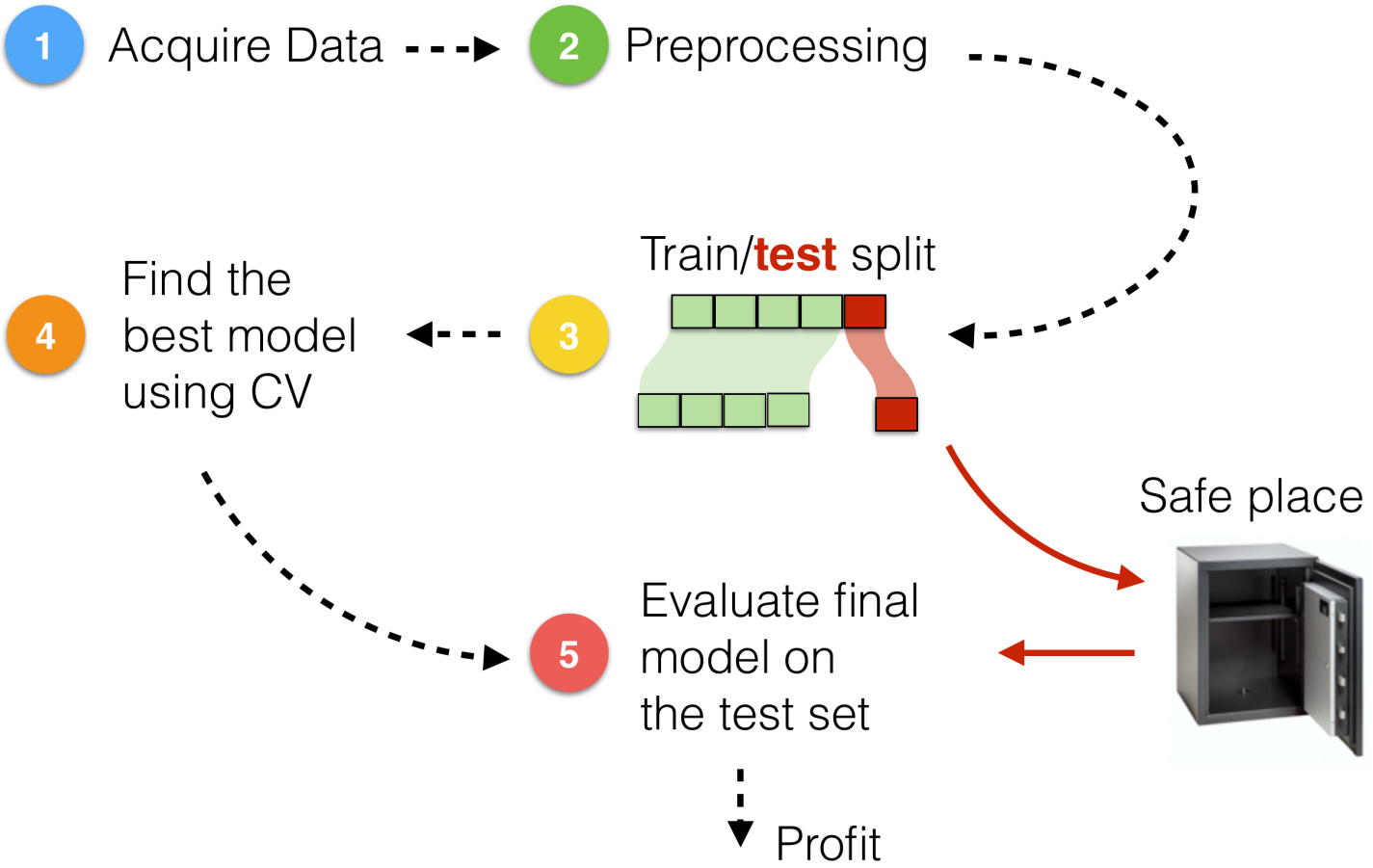
How many PCs should we keep?

Principle Component Analysis (**PCA**)



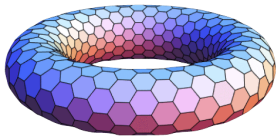
Can be used as part of supervised learning pipeline

Supervised Learning pipeline



Supervised Learning pipeline

1



200D raw data

2 Preprocessing

Train/**test** split

3

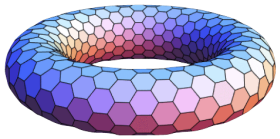
4 Find the best model using CV

Safe place

5 Evaluate final model on the test set

Profit

Supervised Learning pipeline

1  ---▸ 2 Normalisation (subtract mean)

200D raw data

Train/**test** split

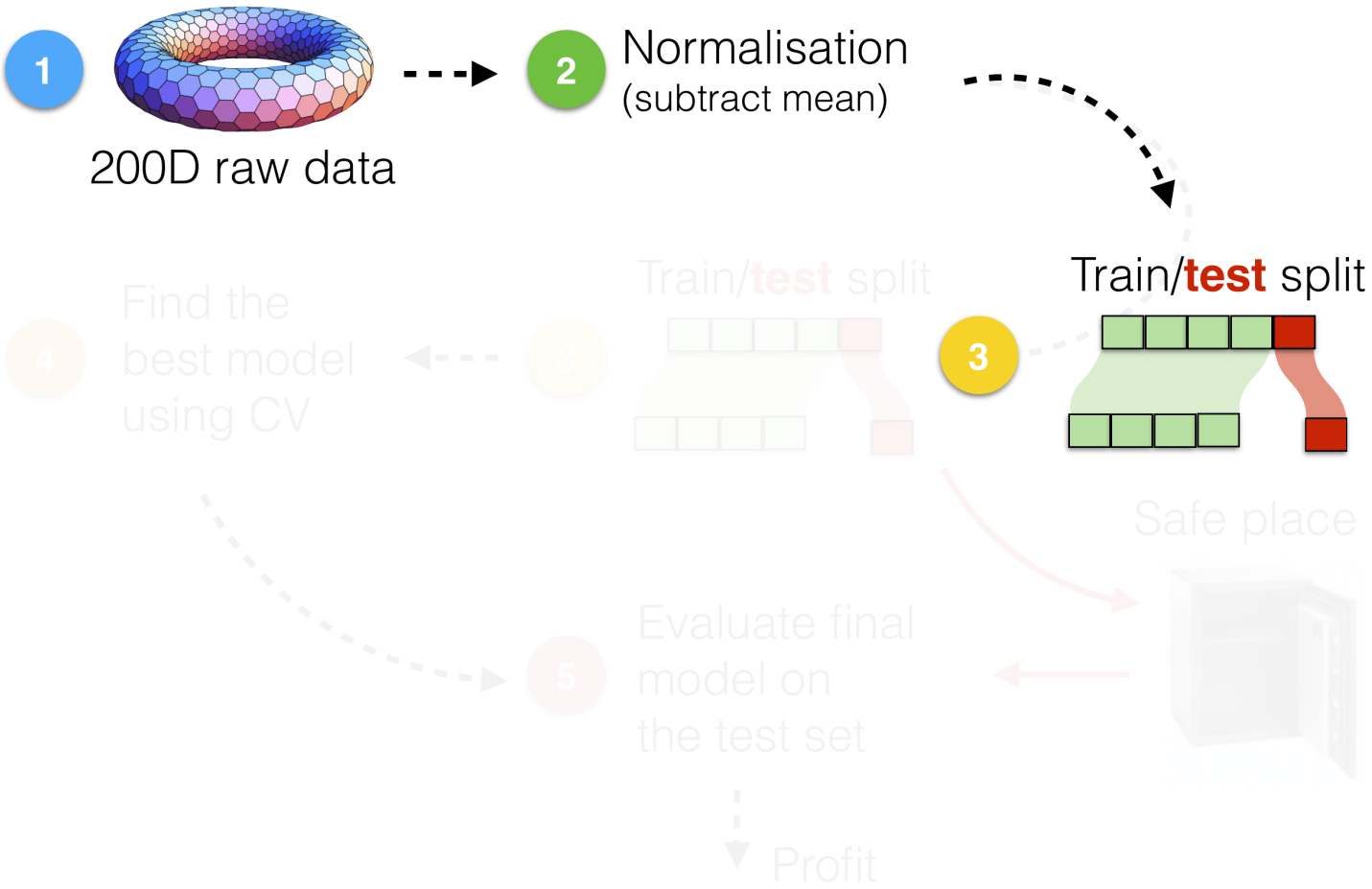
Safe place

Find the best model using CV

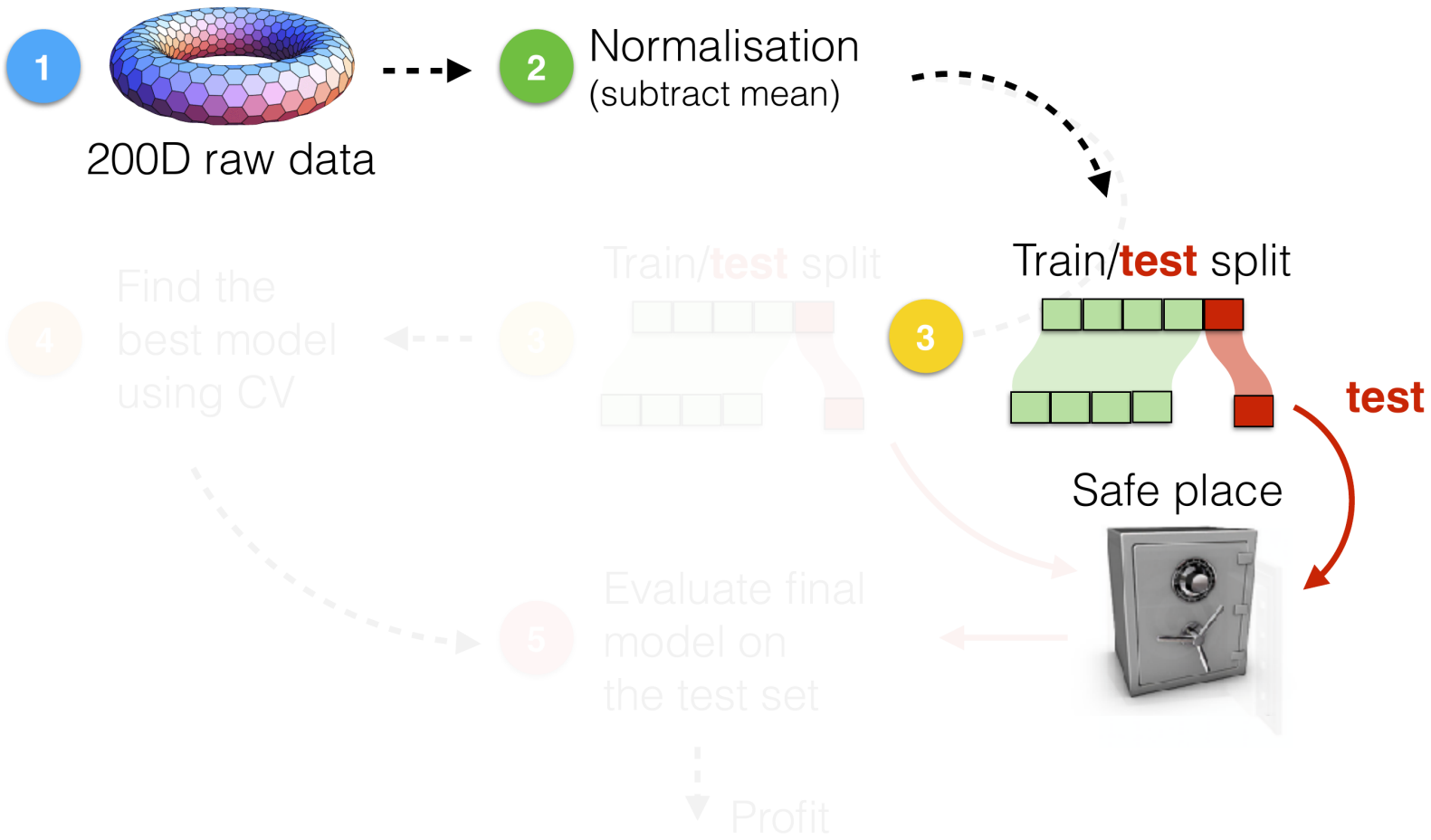
Evaluate final model on the test set

Profit

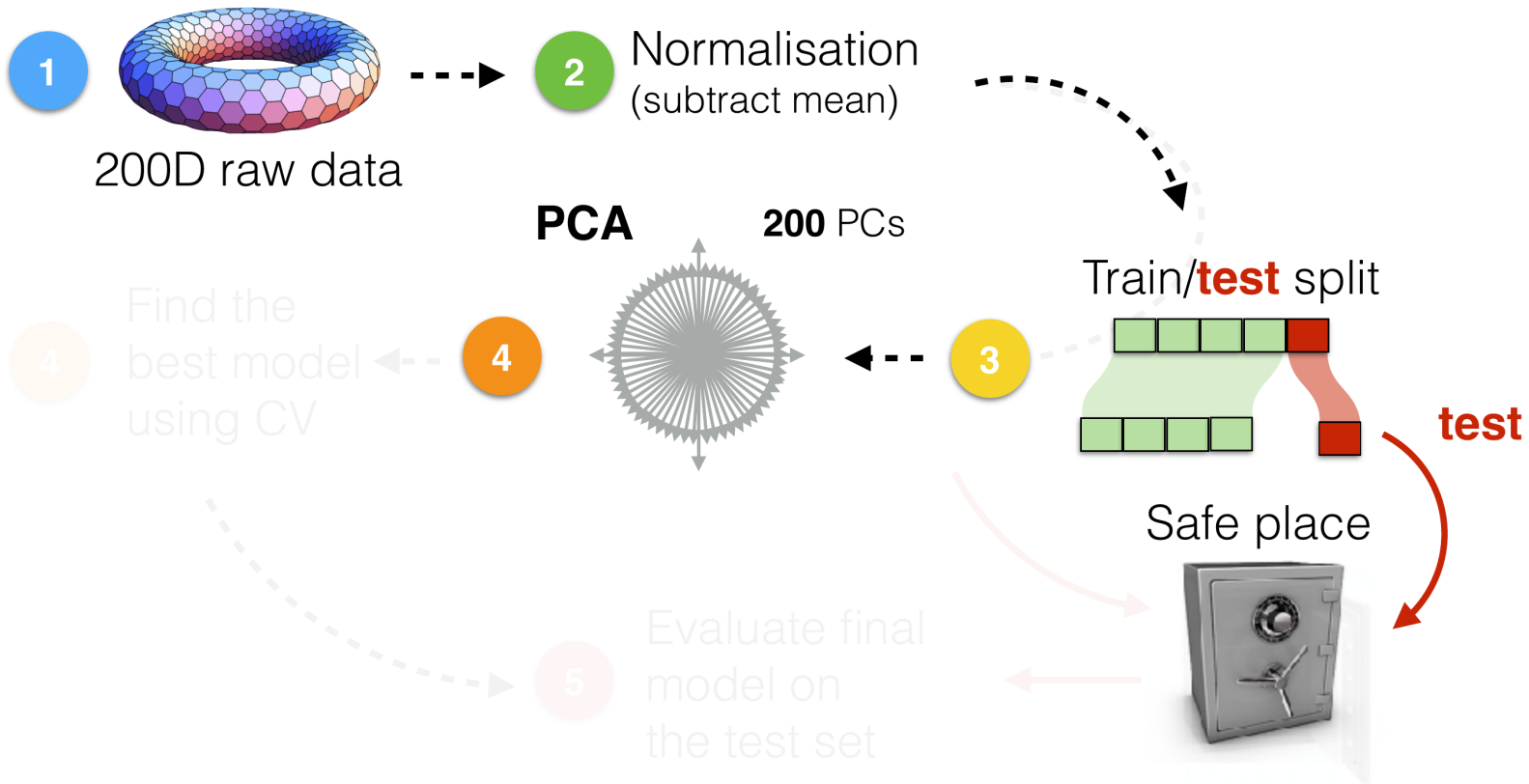
Supervised Learning pipeline



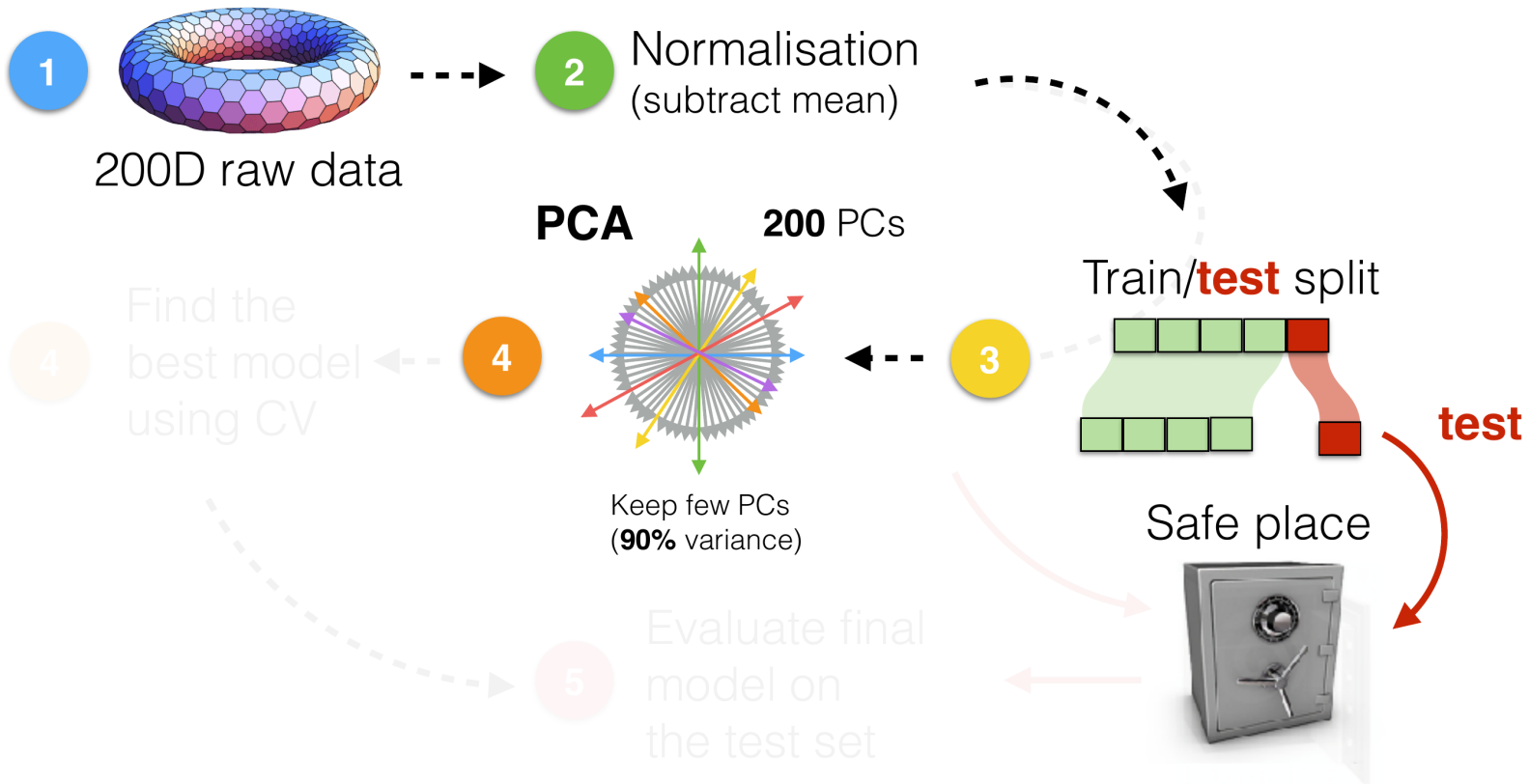
Supervised Learning pipeline



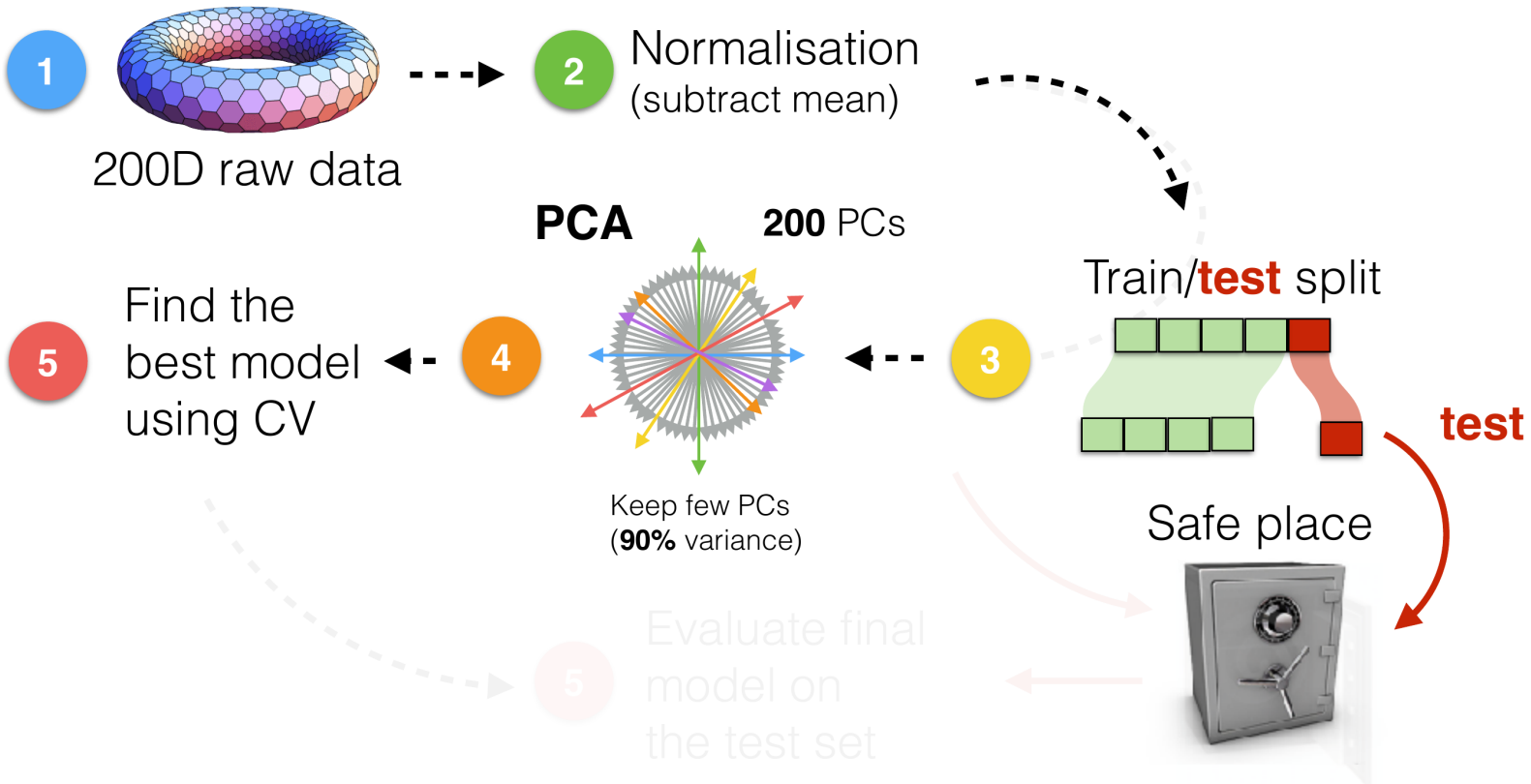
Supervised Learning pipeline



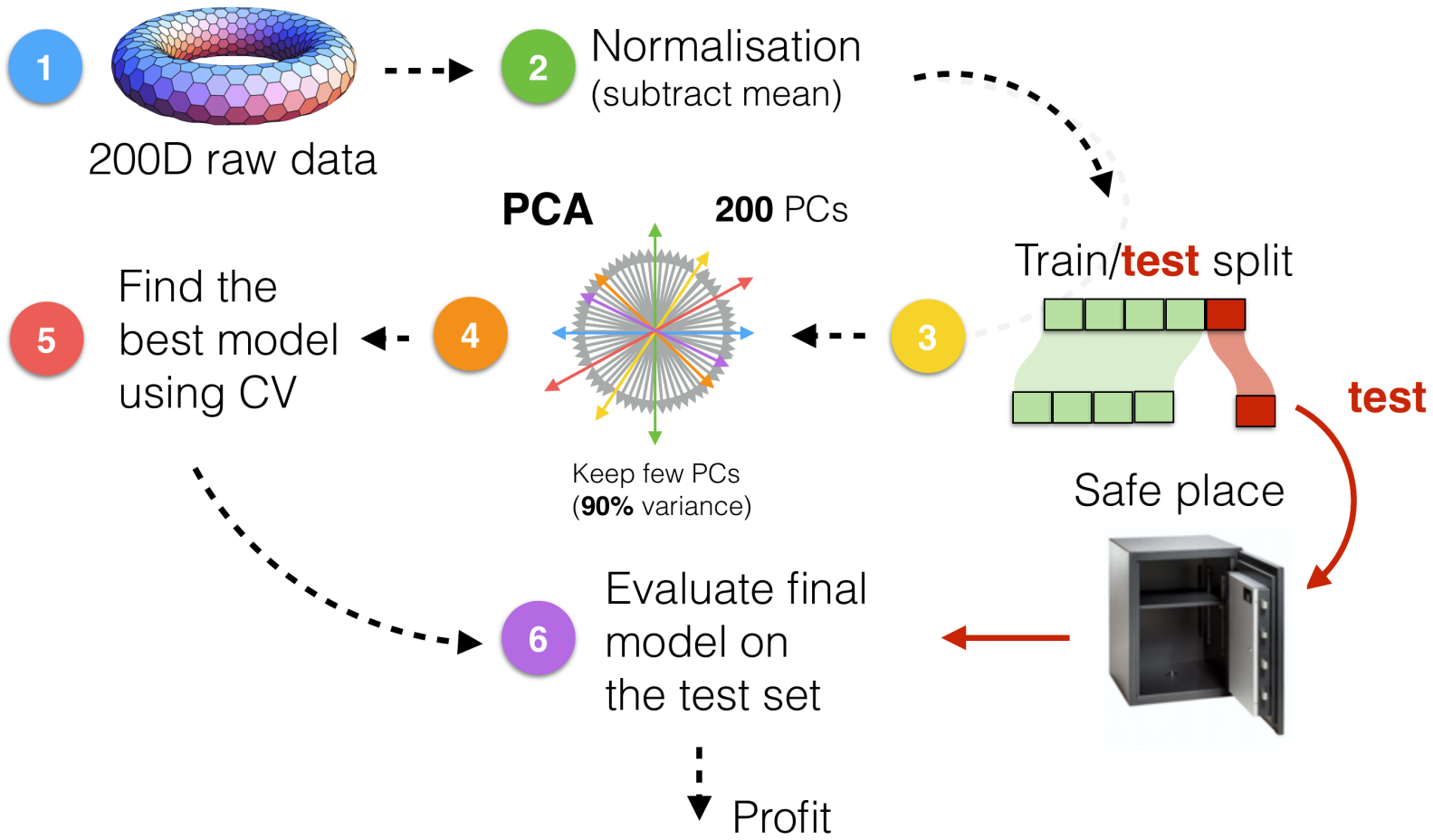
Supervised Learning pipeline



Supervised Learning pipeline

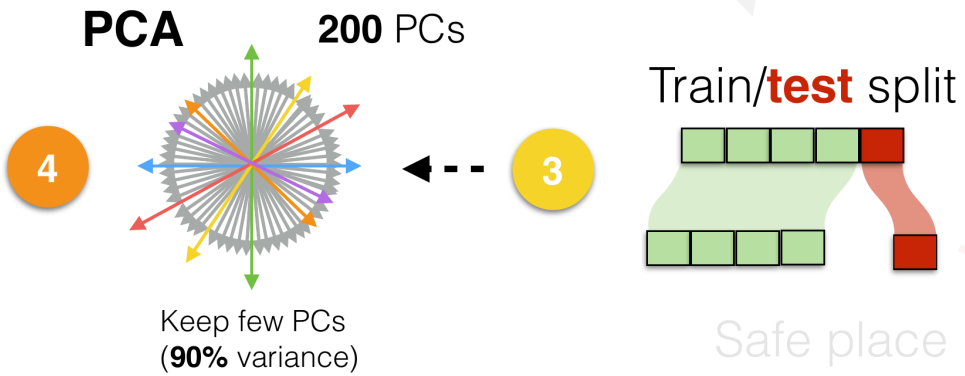


Supervised Learning pipeline



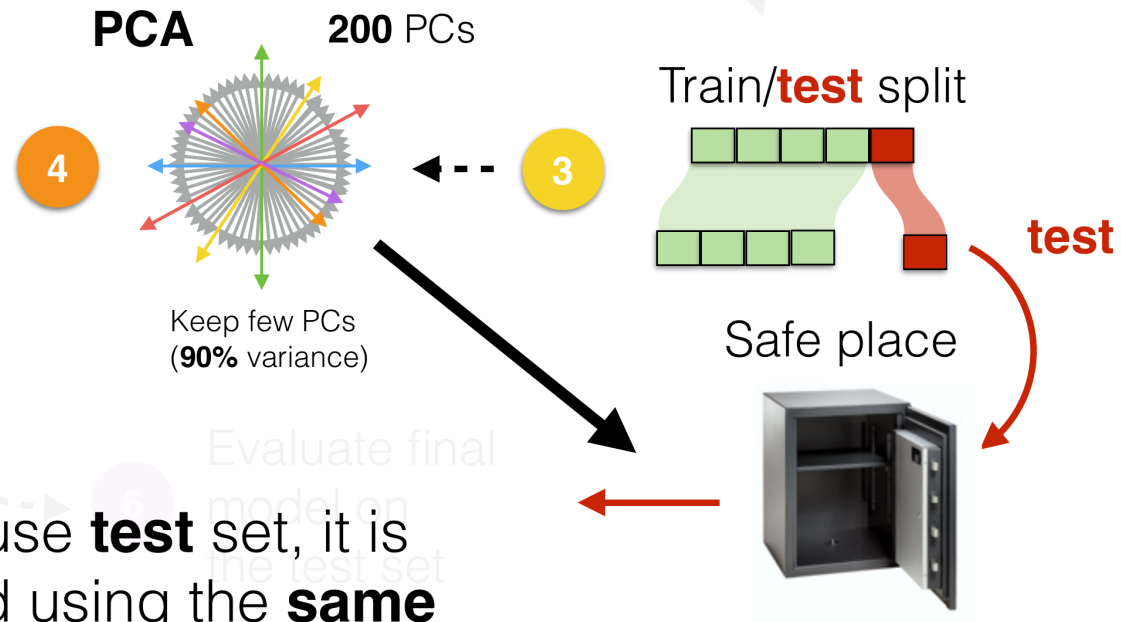
Supervised Learning pipeline

Here, **PCA** is performed only on **training data**



Supervised Learning pipeline

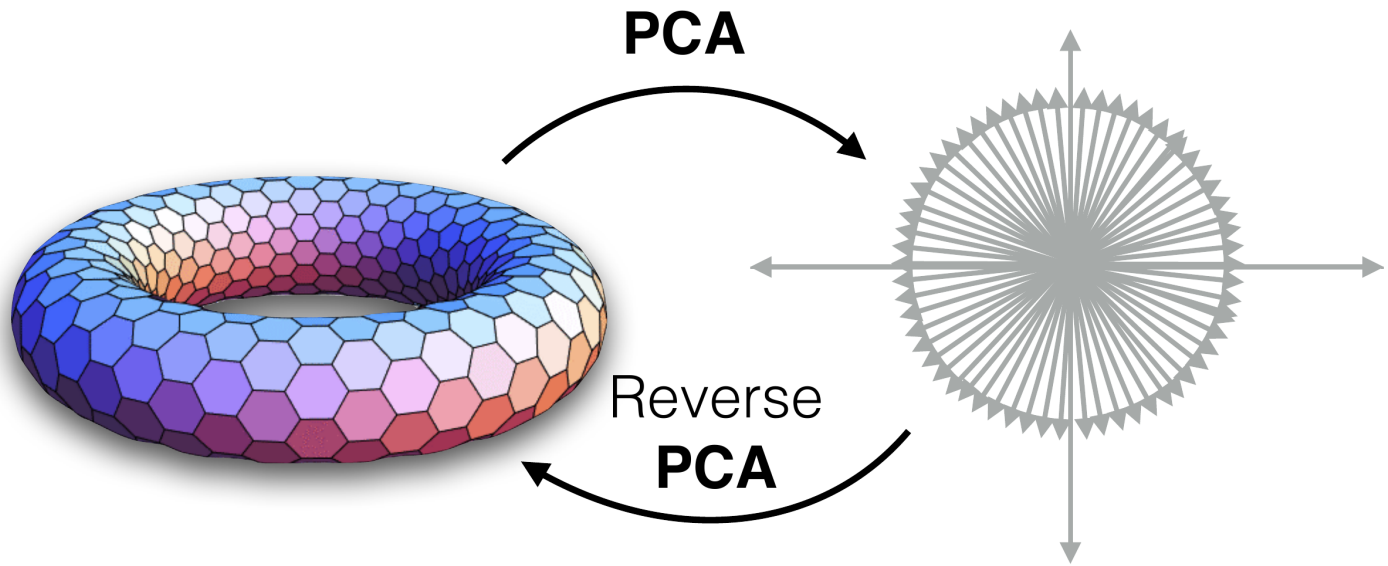
Here, **PCA** is performed only on **training data**



When we use **test** set, it is transformed using the **same eigenvectors** as **training** data

Principal Component Analysis

PCA has an "**undo**" button



You can **recover** the **original features** back!

PCA has an "**undo**" button

Conventional PCA

$$eigenvectors \times Z^{\top}_{original} = Z_{transformed}$$

Reversed PCA

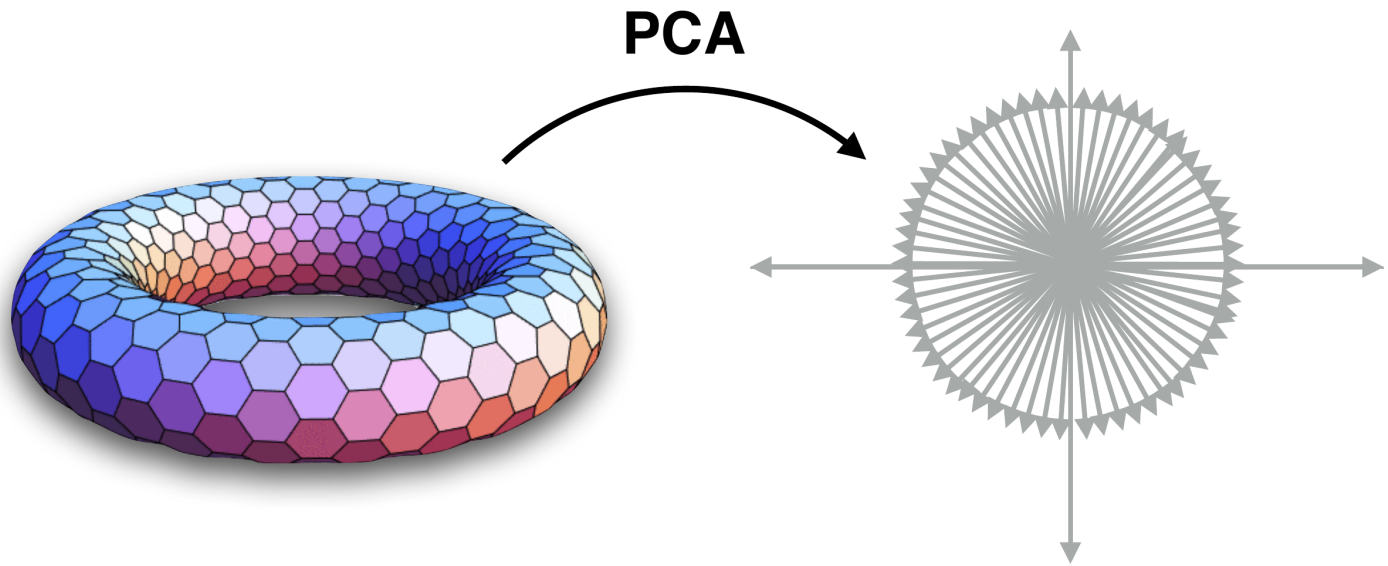
$$eigenvectors^{\top} \times Z_{transformed} = Z_{original}$$

You can **recover** the **original features** back!

Convince yourself: http://www.cs.otago.ac.nz/cosc453/student_tutorials/principal_components.pdf

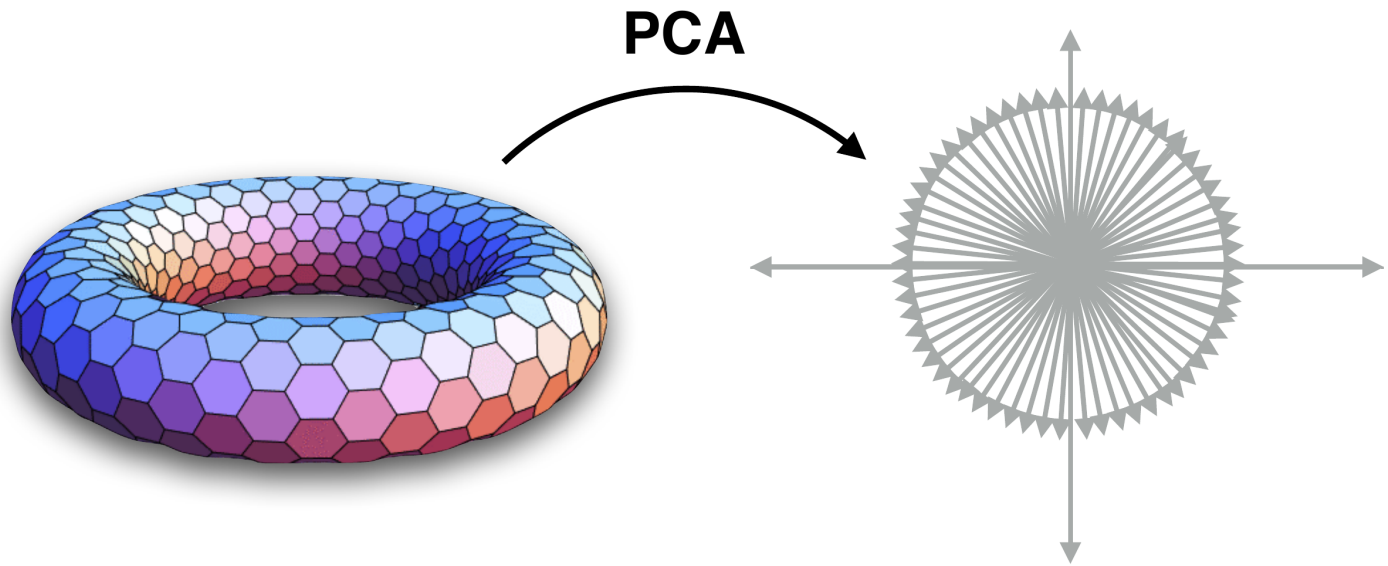
Principal Component Analysis

Principle components **are linear combinations** of **original features**



Principal Component Analysis

Principle components **are linear combinations** of **original features**



So if you predict anything based on PCs, the **meaning** of original features **is not preserved** after the transformation

Problem Setting of PCA

- In PCA, we are interested in finding projections $\mathbf{x}'_n$ of data points $\mathbf{x}_n$ that are as similar to the original data points as possible, but which have a significantly lower intrinsic dimensionality.

Problem Setting of PCA

- In PCA, we are interested in finding projections $\mathbf{x}'_n$ of data points $\mathbf{x}_n$ that are as similar to the original data points as possible, but which have a significantly lower intrinsic dimensionality.
- Suppose we have a dataset $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\} \in \mathbb{R}^{N \times D}$, where $\mathbf{x}_n \in \mathbb{R}^D$, $\mathbb{E}(\mathbf{X}) = 0$ and the sample covariance of $\mathbf{X}$ is

$$S = \frac{1}{N}\mathbf{X}^T\mathbf{X} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n\mathbf{x}_n^T$$

Problem Setting of PCA

- In PCA, we are interested in finding projections $\mathbf{x}'_n$ of data points $\mathbf{x}_n$ that are as similar to the original data points as possible, but which have a significantly lower intrinsic dimensionality.
- Suppose we have a dataset $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\} \in \mathbb{R}^{N \times D}$, where $\mathbf{x}_n \in \mathbb{R}^D$, $\mathbb{E}(\mathbf{X}) = 0$ and the sample covariance of $\mathbf{X}$ is

$$S = \frac{1}{N}\mathbf{X}^T\mathbf{X} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n\mathbf{x}_n^T$$

- We assume there exists a low-dimensional compressed representation (code) of $\mathbf{x}_n$

$$\mathbf{z}_n = \mathbf{B}^T\mathbf{x}_n \in \mathbb{R}^M,$$

where $\mathbf{B} = [\mathbf{b}_1, \ldots, \mathbf{b}_M] \in \mathbb{R}^{D \times M}$ is the projection matrix.

Problem Setting of PCA

- In PCA, we are interested in finding projections $\mathbf{x}'_n$ of data points $\mathbf{x}_n$ that are as similar to the original data points as possible, but which have a significantly lower intrinsic dimensionality.
- Suppose we have a dataset $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\} \in \mathbb{R}^{N \times D}$, where $\mathbf{x}_n \in \mathbb{R}^D$, $\mathbb{E}(\mathbf{X}) = 0$ and the sample covariance of $\mathbf{X}$ is

$$S = \frac{1}{N}\mathbf{X}^T\mathbf{X} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n\mathbf{x}_n^T$$

- We assume there exists a low-dimensional compressed representation (code) of $\mathbf{x}_n$

$$\mathbf{z}_n = \mathbf{B}^T\mathbf{x}_n \in \mathbb{R}^M,$$

  where $\mathbf{B} = [\mathbf{b}_1, \ldots, \mathbf{b}_M] \in \mathbb{R}^{D \times M}$ is the projection matrix.
- We further assume that the columns of $\mathbf{B}$ are orthonormal, so that $\mathbf{b}_i^T\mathbf{b}_j = 0$ iff $i \neq j$ and $\mathbf{b}_i^T\mathbf{b}_i = 1$

Problem Setting of PCA

- In PCA, we are interested in finding projections $\mathbf{x}'_n$ of data points $\mathbf{x}_n$ that are as similar to the original data points as possible, but which have a significantly lower intrinsic dimensionality.
- Suppose we have a dataset $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\} \in \mathbb{R}^{N \times D}$, where $\mathbf{x}_n \in \mathbb{R}^D$, $\mathbb{E}(\mathbf{X}) = 0$ and the sample covariance of $\mathbf{X}$ is

$$S = \frac{1}{N}\mathbf{X}^T\mathbf{X} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n\mathbf{x}_n^T$$

- We assume there exists a low-dimensional compressed representation (code) of $\mathbf{x}_n$

$$\mathbf{z}_n = \mathbf{B}^T\mathbf{x}_n \in \mathbb{R}^M,$$

  where $\mathbf{B} = [\mathbf{b}_1, \ldots, \mathbf{b}_M] \in \mathbb{R}^{D \times M}$ is the projection matrix.
- We further assume that the columns of $\mathbf{B}$ are orthonormal, so that $\mathbf{b}_i^T\mathbf{b}_j = 0$ iff $i \neq j$ and $\mathbf{b}_i^T\mathbf{b}_i = 1$
- We seek an M-dimensional subspace $U \subseteq \mathbb{R}^D$, $dim(U) = M < D$ onto which we project the data

Problem Setting of PCA

- In PCA, we are interested in finding projections $\mathbf{x}'_n$ of data points $\mathbf{x}_n$ that are as similar to the original data points as possible, but which have a significantly lower intrinsic dimensionality.
- Suppose we have a dataset $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\} \in \mathbb{R}^{N \times D}$, where $\mathbf{x}_n \in \mathbb{R}^D$, $\mathbb{E}(\mathbf{X}) = 0$ and the sample covariance of $\mathbf{X}$ is

$$S = \frac{1}{N}\mathbf{X}^T\mathbf{X} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n\mathbf{x}_n^T$$

- We assume there exists a low-dimensional compressed representation (code) of $\mathbf{x}_n$

$$\mathbf{z}_n = \mathbf{B}^T\mathbf{x}_n \in \mathbb{R}^M,$$

  where $\mathbf{B} = [\mathbf{b}_1, \ldots, \mathbf{b}_M] \in \mathbb{R}^{D \times M}$ is the projection matrix.
- We further assume that the columns of $\mathbf{B}$ are orthonormal, so that $\mathbf{b}_i^T\mathbf{b}_j = 0$ iff $i \neq j$ and $\mathbf{b}_i^T\mathbf{b}_i = 1$
- We seek an M-dimensional subspace $U \subseteq \mathbb{R}^D$, $dim(U) = M < D$ onto which we project the data
- We denote the projected data as $\mathbf{x}'_n \in U$ and their coordinates by $\mathbf{z}_n$

Problem Setting of PCA

- We can describe the information contained in the data by looking at the spread of the data and measure it with **variance**

Problem Setting of PCA

- We can describe the information contained in the data by looking at the spread of the data and measure it with **variance**
- In this setting, retaining most information after data compression is equivalent to capturing the largest amount of variance in the low-dimensional representation

Problem Setting of PCA

- We can describe the information contained in the data by looking at the spread of the data and measure it with **variance**
- In this setting, retaining most information after data compression is equivalent to capturing the largest amount of variance in the low-dimensional representation
- We maximize the variance of the low-dimensional code using a sequential approach.

Problem Setting of PCA

- We can describe the information contained in the data by looking at the spread of the data and measure it with **variance**
- In this setting, retaining most information after data compression is equivalent to capturing the largest amount of variance in the low-dimensional representation
- We maximize the variance of the low-dimensional code using a sequential approach.
- We start by seeking a single vector $\mathbf{b}_1 \in \mathbb{R}^D$ that maximizes variance of the projected data

$$\mathbb{V}ar(z_1) = \frac{1}{N}\sum_{n=1}^{N} z_{1n}^2 = \frac{1}{N}\sum_{n=1}^{N}(\mathbf{b}_1^T\mathbf{x}_n)^2 =$$

Problem Setting of PCA

- We can describe the information contained in the data by looking at the spread of the data and measure it with **variance**
- In this setting, retaining most information after data compression is equivalent to capturing the largest amount of variance in the low-dimensional representation
- We maximize the variance of the low-dimensional code using a sequential approach.
- We start by seeking a single vector $\mathbf{b}_1 \in \mathbb{R}^D$ that maximizes variance of the projected data

$$\mathbb{V}ar(z_1) = \frac{1}{N}\sum_{n=1}^{N} z_{1n}^2 = \frac{1}{N}\sum_{n=1}^{N}(\mathbf{b}_1^T\mathbf{x}_n)^2 =$$

$$= \frac{1}{N}\sum_{n=1}^{N}\mathbf{b}_1^T\mathbf{x}_n\mathbf{x}_n^T\mathbf{b}_1 = \mathbf{b}_1^T\Big(\frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n\mathbf{x}_n^T\Big)\mathbf{b}_1 = \mathbf{b}_1^T S\mathbf{b}_1,$$

where S is the sample covariance matrix.

Problem Setting of PCA

- We can describe the information contained in the data by looking at the spread of the data and measure it with **variance**
- In this setting, retaining most information after data compression is equivalent to capturing the largest amount of variance in the low-dimensional representation
- We maximize the variance of the low-dimensional code using a sequential approach.
- We start by seeking a single vector $\mathbf{b}_1 \in \mathbb{R}^D$ that maximizes variance of the projected data

$$\mathbb{V}ar(z_1) = \frac{1}{N}\sum_{n=1}^{N} z_{1n}^2 = \frac{1}{N}\sum_{n=1}^{N}(\mathbf{b}_1^T\mathbf{x}_n)^2 =$$

$$= \frac{1}{N}\sum_{n=1}^{N}\mathbf{b}_1^T\mathbf{x}_n\mathbf{x}_n^T\mathbf{b}_1 = \mathbf{b}_1^T\Big(\frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n\mathbf{x}_n^T\Big)\mathbf{b}_1 = \mathbf{b}_1^T S\mathbf{b}_1,$$

where S is the sample covariance matrix.

- Note that if we would not put a unit vector contraint on $\mathbf{b}_1$, the variance would increase for longer-length $\mathbf{b}_1$.

Mathematical Derivations

We need to solve the following constrained optimization problem

$$\max_{\mathbf{b}_1} \mathbf{b}_1^T S \mathbf{b}_1$$

$$\text{subject to } \|\mathbf{b}_1\|^2 = 1$$

Mathematical Derivations

We need to solve the following constrained optimization problem

$$\max_{\mathbf{b}_1} \mathbf{b}_1^T S \mathbf{b}_1$$

$$\text{subject to } \|\mathbf{b}_1\|^2 = 1$$

The Lagragian will be

$$\Lambda(\mathbf{b}_1, \lambda_1) = \mathbf{b}_1^T S \mathbf{b}_1 + \lambda_1 (1 - \mathbf{b}_1^T \mathbf{b}_1)$$

Mathematical Derivations

We need to solve the following constrained optimization problem

$$\max_{\mathbf{b}_1} \mathbf{b}_1^T S \mathbf{b}_1$$

$$\text{subject to } \|\mathbf{b}_1\|^2 = 1$$

The Lagragian will be

$$\Lambda(\mathbf{b}_1, \lambda_1) = \mathbf{b}_1^T S \mathbf{b}_1 + \lambda_1 (1 - \mathbf{b}_1^T \mathbf{b}_1)$$

Finding the partial derivatives w.r.t. $\mathbf{b}_1$ and λ_1

$$\frac{\partial \Lambda}{\partial \mathbf{b}_1} = 2\mathbf{b}_1^T S - 2\lambda_1 \mathbf{b}_1^T, \frac{\partial \Lambda}{\partial \lambda_1} = 1 - \mathbf{b}_1^T \mathbf{b}_1$$

Mathematical Derivations

We need to solve the following constrained optimization problem

$$\max_{\mathbf{b}_1} \mathbf{b}_1^T S \mathbf{b}_1$$

$$\text{subject to } \|\mathbf{b}_1\|^2 = 1$$

The Lagragian will be

$$\Lambda(\mathbf{b}_1, \lambda_1) = \mathbf{b}_1^T S \mathbf{b}_1 + \lambda_1 (1 - \mathbf{b}_1^T \mathbf{b}_1)$$

Finding the partial derivatives w.r.t. $\mathbf{b}_1$ and λ_1

$$\frac{\partial \Lambda}{\partial \mathbf{b}_1} = 2\mathbf{b}_1^T S - 2\lambda_1 \mathbf{b}_1^T, \frac{\partial \Lambda}{\partial \lambda_1} = 1 - \mathbf{b}_1^T \mathbf{b}_1$$

Setting these partial derivatives to 0 gives us

$$S\mathbf{b}_1 = \lambda_1 \mathbf{b}_1$$
$$\mathbf{b}_1^T \mathbf{b}_1 = 1$$

Does this look familiar?

Mathematical Derivations

$$S\mathbf{b}_1 = \lambda_1 \mathbf{b}_1$$
$$\mathbf{b}_1^T \mathbf{b}_1 = 1$$

$\mathbf{b}_1$ is an eigenvector of the data covariance matrix S, and the Lagrange multiplier λ_1 plays the role of the corresponding eigenvalue.

Mathematical Derivations

$$S\mathbf{b}_1 = \lambda_1 \mathbf{b}_1$$
$$\mathbf{b}_1^T \mathbf{b}_1 = 1$$

$\mathbf{b}_1$ is an eigenvector of the data covariance matrix S, and the Lagrange multiplier λ_1 plays the role of the corresponding eigenvalue.
The objective can be re-written as

$$\mathbb{V}ar(z_1) = \mathbf{b}_1^T S \mathbf{b}_1 = \lambda_1 \mathbf{b}_1^T \mathbf{b}_1 = \lambda_1,$$

the variance of the data projected onto a one-dimensional subspace equals the eigenvalue that is associated with the basis vector $\mathbf{b}_1$ that spans this subspace.

Mathematical Derivations

$$S\mathbf{b}_1 = \lambda_1 \mathbf{b}_1$$
$$\mathbf{b}_1^T \mathbf{b}_1 = 1$$

$\mathbf{b}_1$ is an eigenvector of the data covariance matrix S, and the Lagrange multiplier λ_1 plays the role of the corresponding eigenvalue.
The objective can be re-written as

$$\mathbb{V}ar(z_1) = \mathbf{b}_1^T S \mathbf{b}_1 = \lambda_1 \mathbf{b}_1^T \mathbf{b}_1 = \lambda_1,$$

the variance of the data projected onto a one-dimensional subspace equals the eigenvalue that is associated with the basis vector $\mathbf{b}_1$ that spans this subspace.
To obtain the projection of $\mathbf{x}_n$ on the obtained subspace, we can use the following formula

$$\mathbf{x}'_n = \mathbf{b}_1 z_{1n} = \mathbf{b}_1 \mathbf{b}_1^T \mathbf{x}_n \in \mathbb{R}^D$$

Mathematical Derivations

- Assume we have found the first $m-1$ principal components as the $m-1$ eigenvectors of S that are associated with the largest $m-1$ eigenvalues.

Mathematical Derivations

- Assume we have found the first $m-1$ principal components as the $m-1$ eigenvectors of S that are associated with the largest $m-1$ eigenvalues.
- The m-th principal component can be found by subtracting the effect of the first $m-1$ principal components $\mathbf{b}_1, \ldots, \mathbf{b}_{m-1}$ from the data, by trying to find principal components that compress the remaining information.

$$\hat{\mathbf{X}} = \mathbf{X} - \sum_{i=1}^{m-1} \mathbf{b}_i \mathbf{b}_i^T \mathbf{X} = \mathbf{X} - \mathbf{B}_{m-1}\mathbf{X},$$

where $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_N] \in \mathbb{R}^{D \times N}$ and $\mathbf{B}_{m-1} = \sum_{i=1}^{m-1} \mathbf{b}_i \mathbf{b}_i^T$ is the projection matrix onto the subspace spanned by $\mathbf{b}_1, \ldots, \mathbf{b}_{m-1}$.

Mathematical Derivations

- Assume we have found the first $m-1$ principal components as the $m-1$ eigenvectors of S that are associated with the largest $m-1$ eigenvalues.
- The m-th principal component can be found by subtracting the effect of the first $m-1$ principal components $\mathbf{b}_1, \ldots, \mathbf{b}_{m-1}$ from the data, by trying to find principal components that compress the remaining information.

$$\hat{\mathbf{X}} = \mathbf{X} - \sum_{i=1}^{m-1} \mathbf{b}_i \mathbf{b}_i^T \mathbf{X} = \mathbf{X} - \mathbf{B}_{m-1}\mathbf{X},$$

where $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_N] \in \mathbb{R}^{D \times N}$ and $\mathbf{B}_{m-1} = \sum_{i=1}^{m-1} \mathbf{b}_i \mathbf{b}_i^T$ is the projection matrix onto the subspace spanned by $\mathbf{b}_1, \ldots, \mathbf{b}_{m-1}$.

- The m-th PC can be found by maximizing the variance

$$\mathbb{V}ar(z_m) = \frac{1}{N}\sum_{n=1}^{N} z_{mn}^2 = \frac{1}{N}\sum_{n=1}^{N} (\mathbf{b}_m^T \hat{\mathbf{x}_n})^2 = \mathbf{b}_m^T \hat{S} \mathbf{b}_m$$

$$\text{subject to } \|\mathbf{b}_m\|^2 = 1$$

where $\hat{S}$ is the covariance matrix of $\hat{\mathbf{X}}$

Mathematical Derivations

- Assume we have found the first $m-1$ principal components as the $m-1$ eigenvectors of S that are associated with the largest $m-1$ eigenvalues.
- The m-th principal component can be found by subtracting the effect of the first $m-1$ principal components $\mathbf{b}_1, \ldots, \mathbf{b}_{m-1}$ from the data, by trying to find principal components that compress the remaining information.

$$\hat{\mathbf{X}} = \mathbf{X} - \sum_{i=1}^{m-1} \mathbf{b}_i \mathbf{b}_i^T \mathbf{X} = \mathbf{X} - \mathbf{B}_{m-1}\mathbf{X},$$

  where $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_N] \in \mathbb{R}^{D \times N}$ and $\mathbf{B}_{m-1} = \sum_{i=1}^{m-1} \mathbf{b}_i \mathbf{b}_i^T$ is the projection matrix onto the subspace spanned by $\mathbf{b}_1, \ldots, \mathbf{b}_{m-1}$.
- The m-th PC can be found by maximizing the variance

$$\mathbb{V}ar(z_m) = \frac{1}{N}\sum_{n=1}^{N} z_{mn}^2 = \frac{1}{N}\sum_{n=1}^{N} (\mathbf{b}_m^T \hat{\mathbf{x}_n})^2 = \mathbf{b}_m^T \hat{S} \mathbf{b}_m$$

$$\text{subject to } \|\mathbf{b}_m\|^2 = 1$$

  where $\hat{S}$ is the covariance matrix of $\hat{\mathbf{X}}$
- $\mathbf{b}_m$ is also an eigenvector of S and $\mathbb{V}ar(z_m) = \mathbf{b}_m^T S \mathbf{b}_m = \lambda_m$ is the m-th largest eigenvalue of S

Summary of PCA

- We can reduce the dimensionality of our data, by using the M eigenvectors of the covariance matrix S associated with the M largest eigenvalues.

Summary of PCA

- We can reduce the dimensionality of our data, by using the M eigenvectors of the covariance matrix S associated with the M largest eigenvalues.
- The projection matrix $\mathbf{B}$ consists of the eigenvectors of S
- The amount of variance PCA captured with the first M principal components is

$$\sum_{i=1}^{M} \lambda_m$$

Summary of PCA

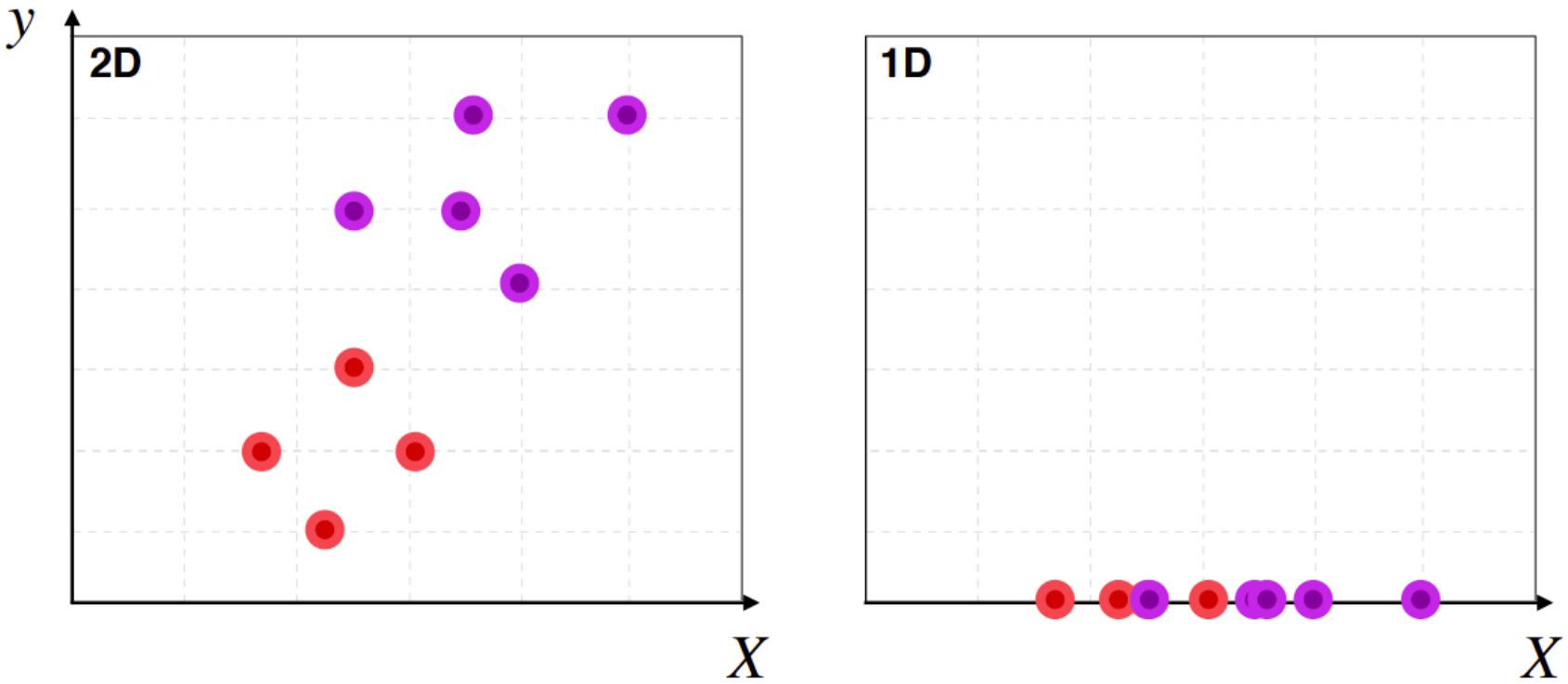
- We can reduce the dimensionality of our data, by using the M eigenvectors of the covariance matrix S associated with the M largest eigenvalues.
- The projection matrix $\mathbf{B}$ consists of the eigenvectors of S
- The amount of variance PCA captured with the first M principal components is

$$\sum_{i=1}^{M} \lambda_m$$

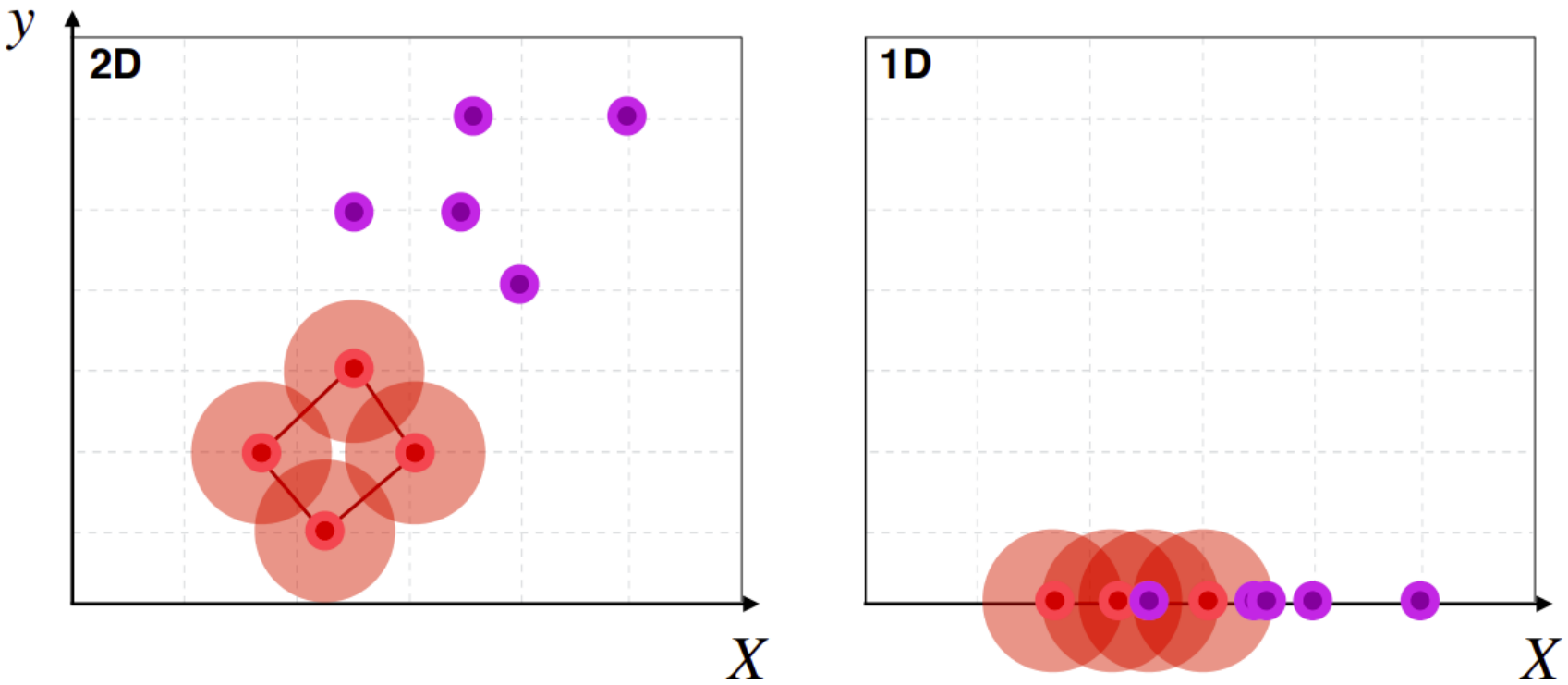
- The variance lost by data compression via PCA is

$$\sum_{i=M+1}^{D} \lambda_m$$

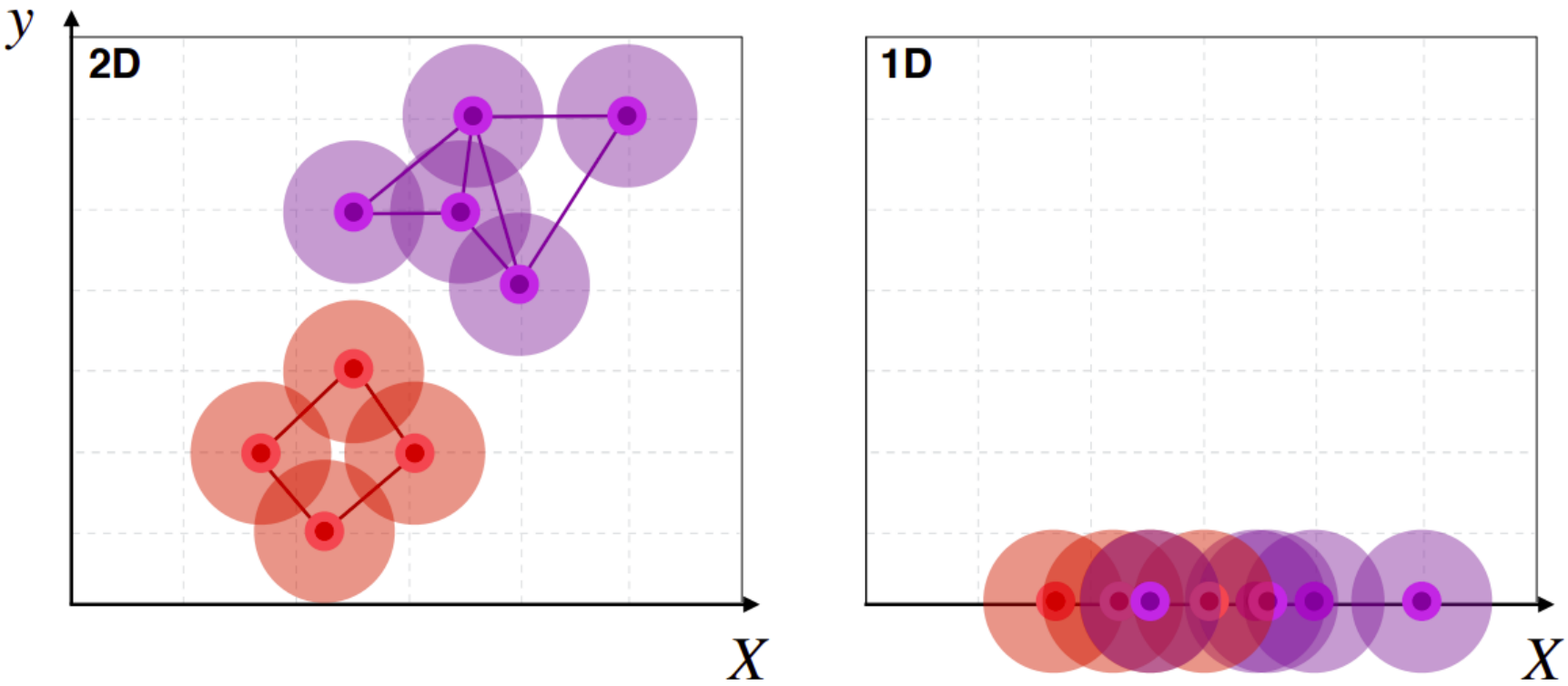
Intuition behind t-SNE



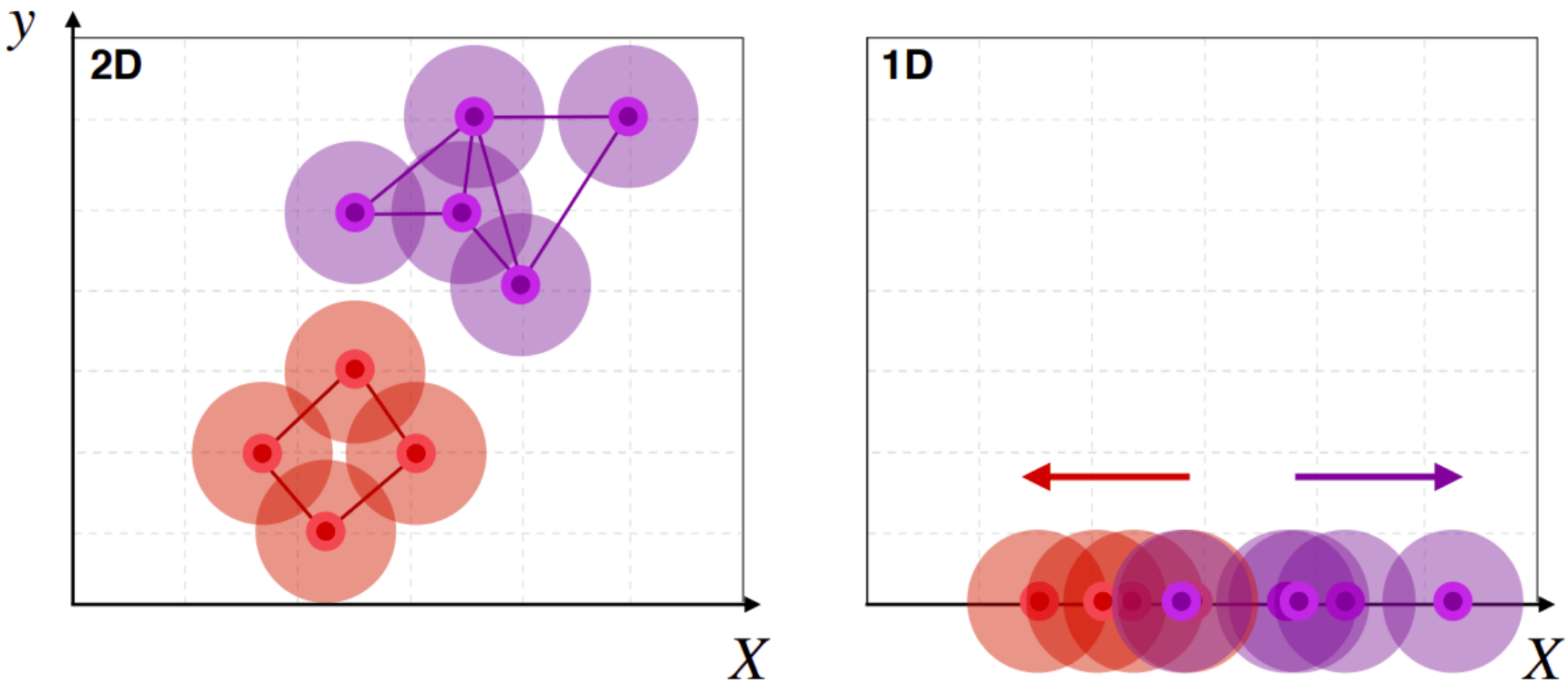
Intuition behind t-SNE



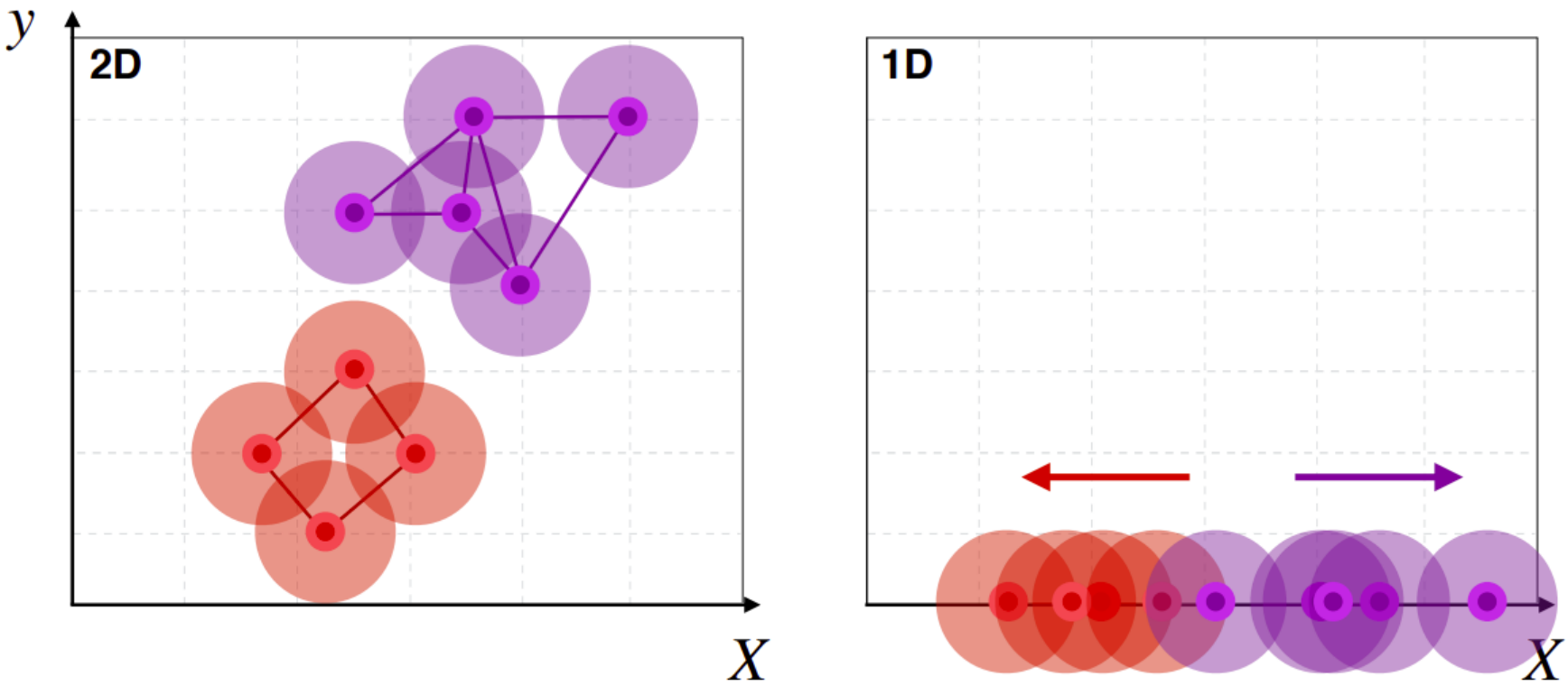
Intuition behind t-SNE



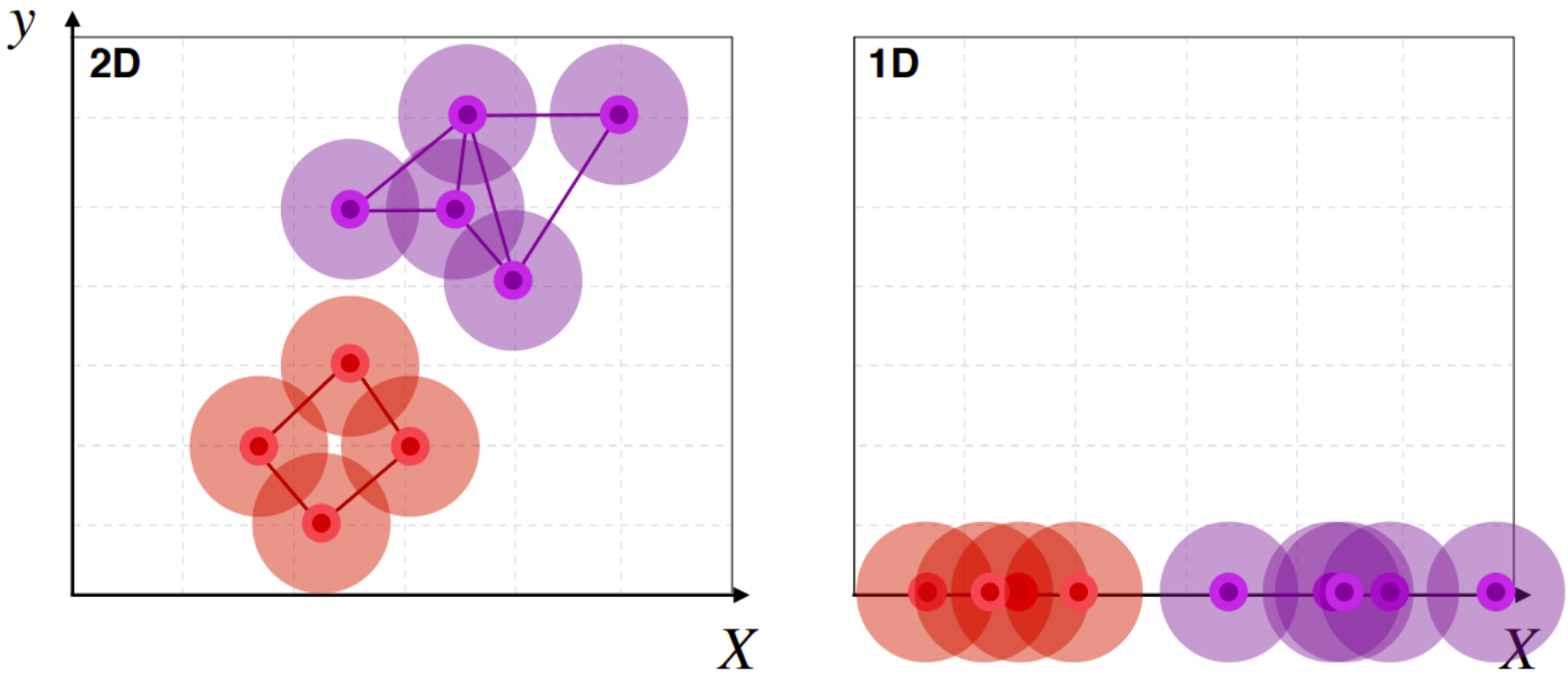
Intuition behind t-SNE



Intuition behind t-SNE

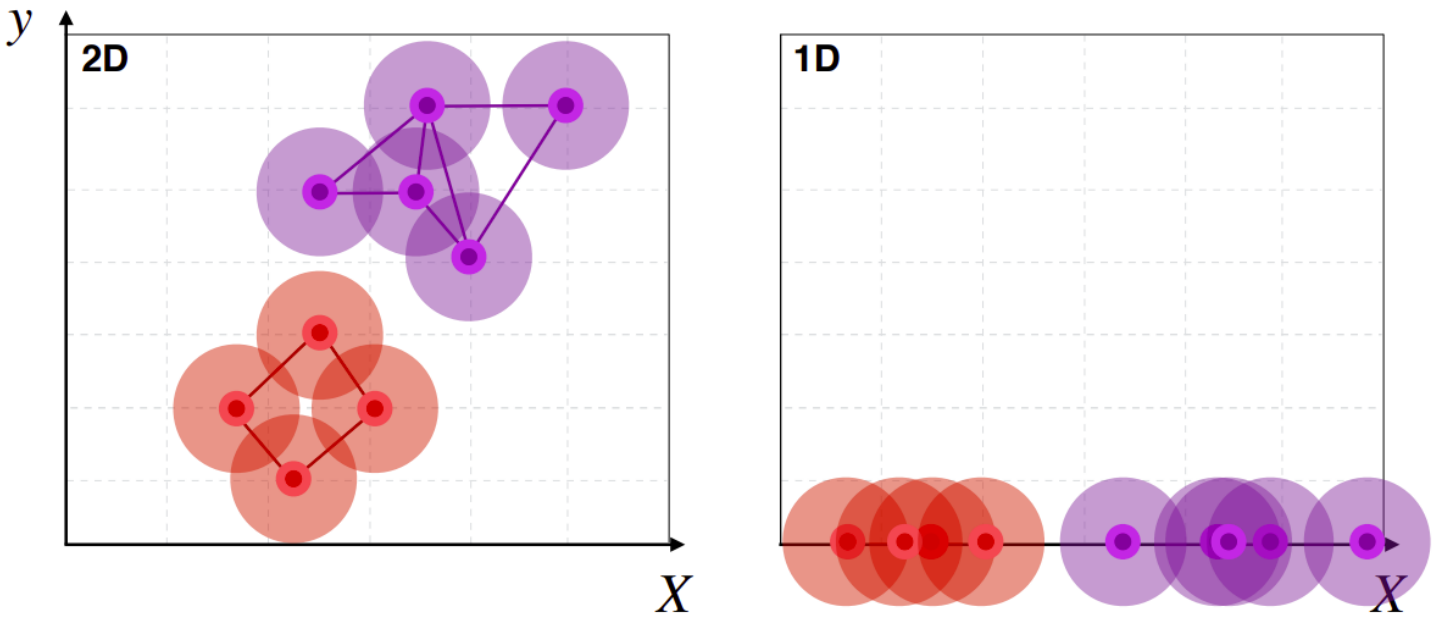


Intuition behind t-SNE



Intuition behind t-SNE

t-SNE iteratively tries to make distances in low-dimensional space to be similar to distances in high-dimensional space



What have we learned today?

- ✓ Dimensionality Reduction
- ✓ Principal Component Analysis
- ✓ t-Distributed Stochastic Neighbor Embedding (t-SNE)