Principal Component Analysis

Principal component analysis

Standard dimensionality reduction technique:

- commonly used in data science and machine learning;
- uncovers lower dimensional patterns in high dimension data.

At its core, PCA is actually a data transformation technique. However, we can frequently ignore some of the transformed dimensions so PCA is often considered a dimension reduction technique.

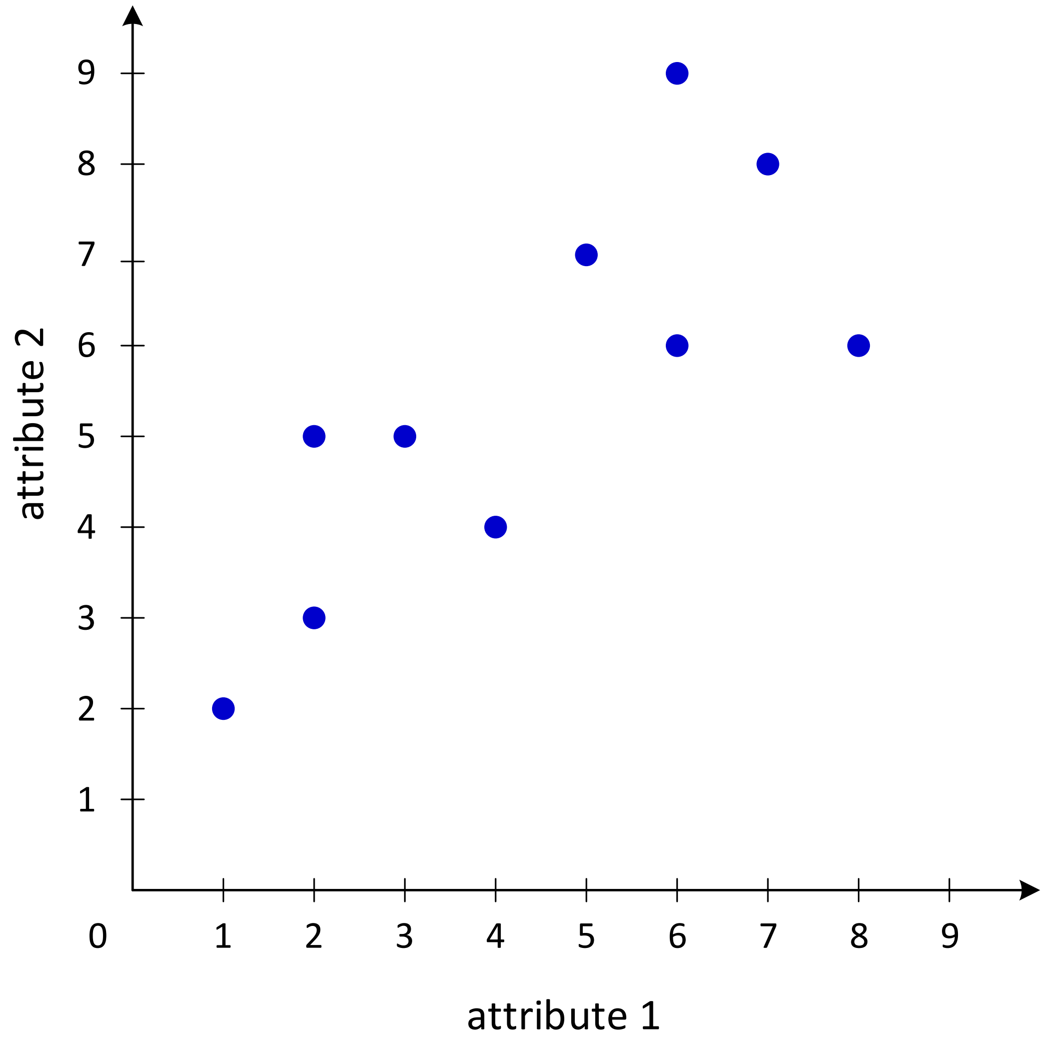
Usually applied to data with many attributes - some correlated with one another. We will start with an example having just two attributes.

Principal component analysis - example

Suppose we have a data set with two attributes or variables.

We wish to understand how these variables contribute to the data, which is useful in understanding the data etc.

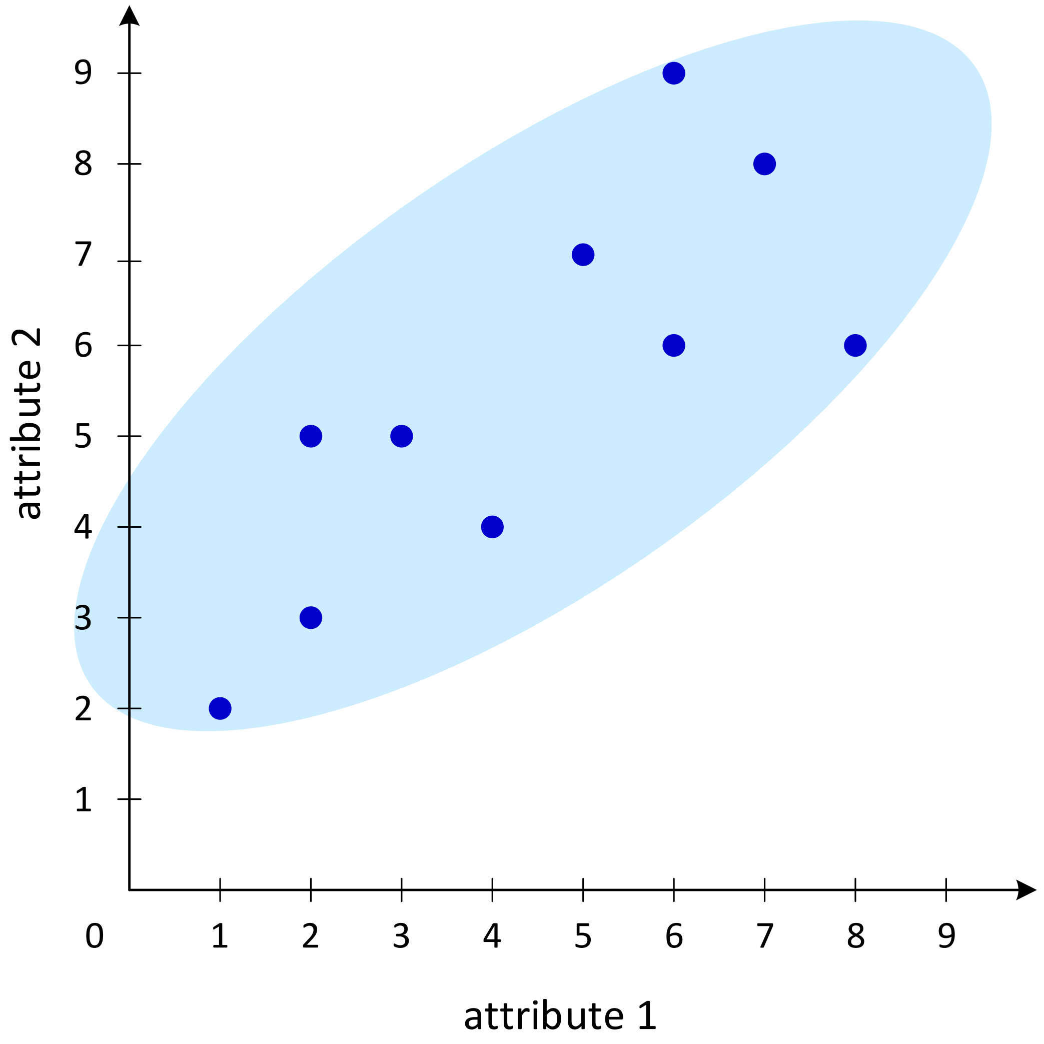
How could we describe these data?



Principal component analysis - example

How could we describe these data?

We might say that the attributes are positively correlated.

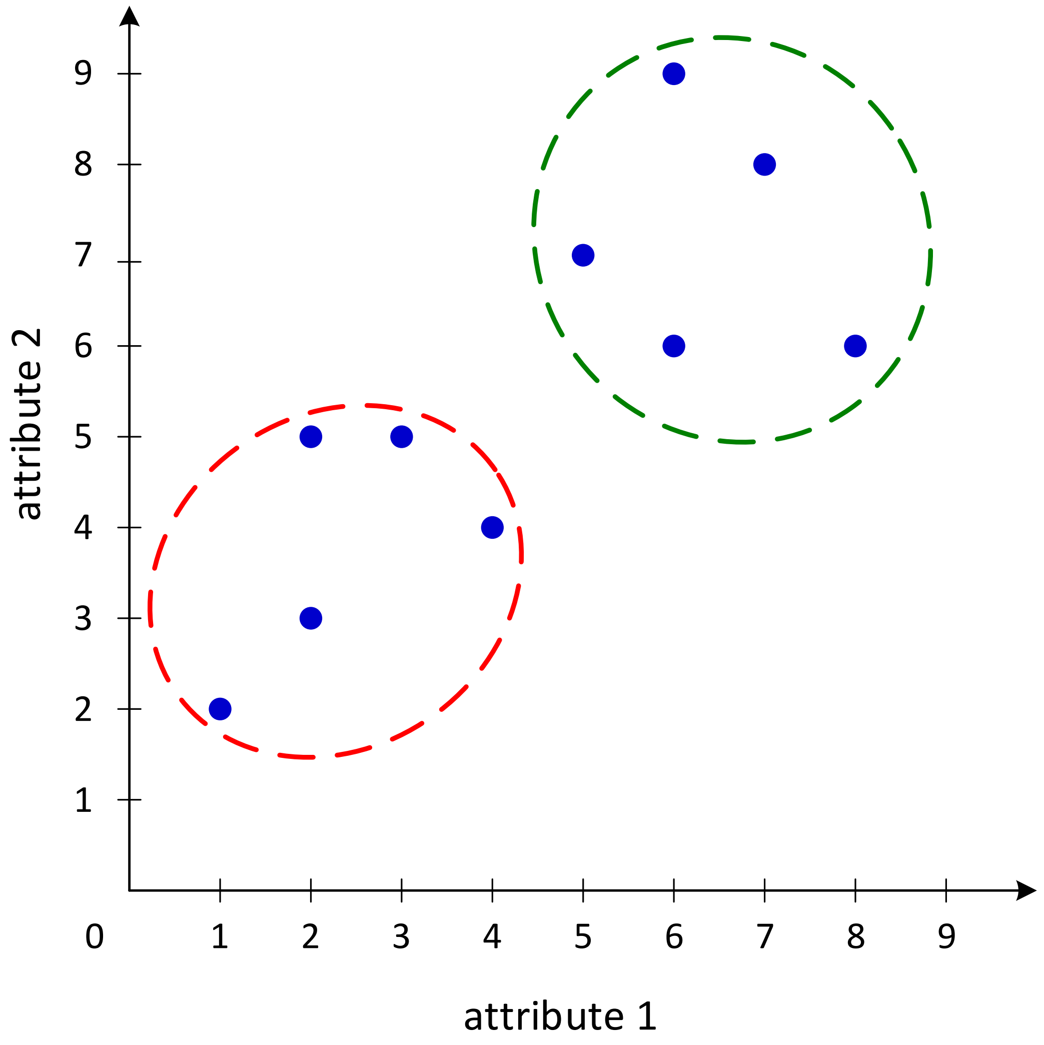


Principal component analysis - example

How could we describe the data?

We might say that the attributes are positively correlated.

We might identify two clusters in the data.



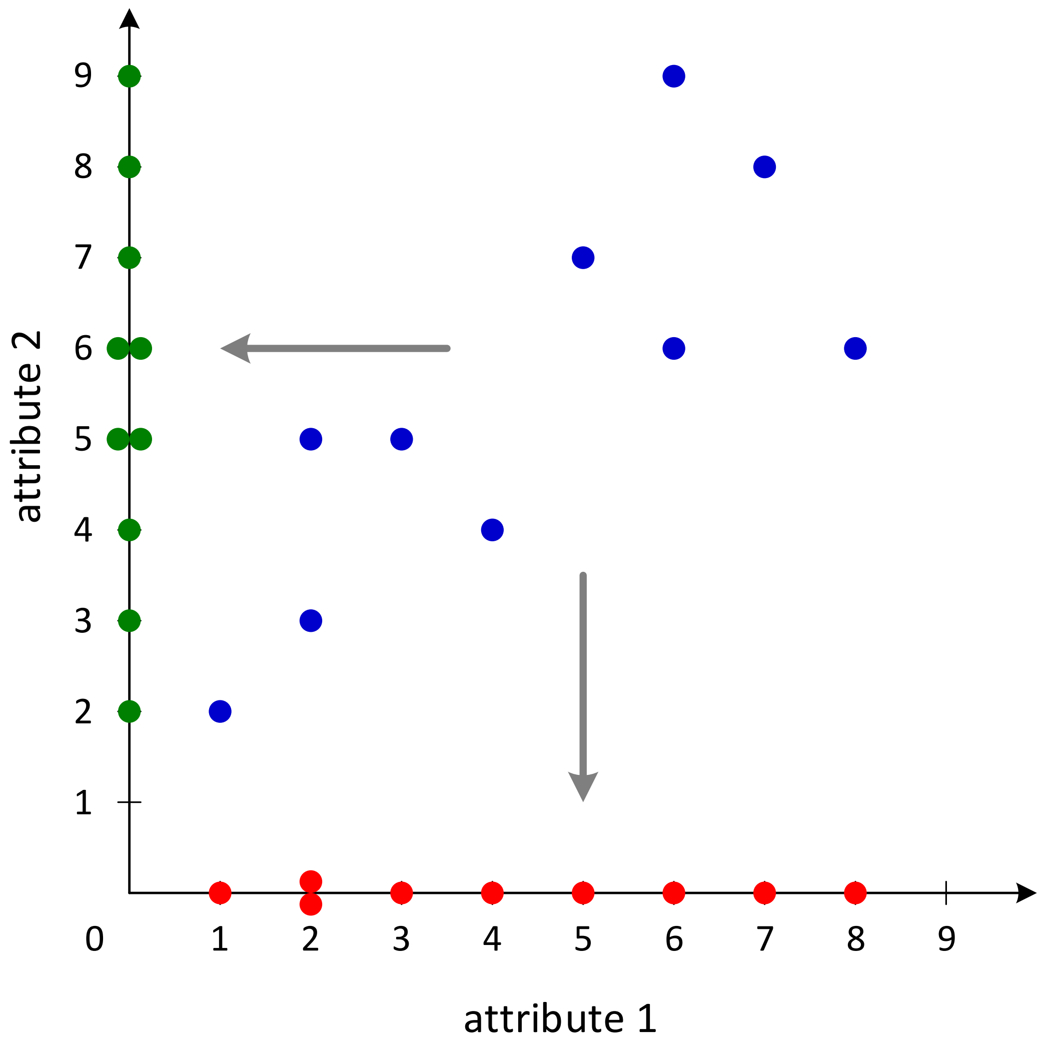
Principal component analysis - example

How could we describe the data?

We might say that the attributes are positively correlated.

We might identify two clusters in the data.

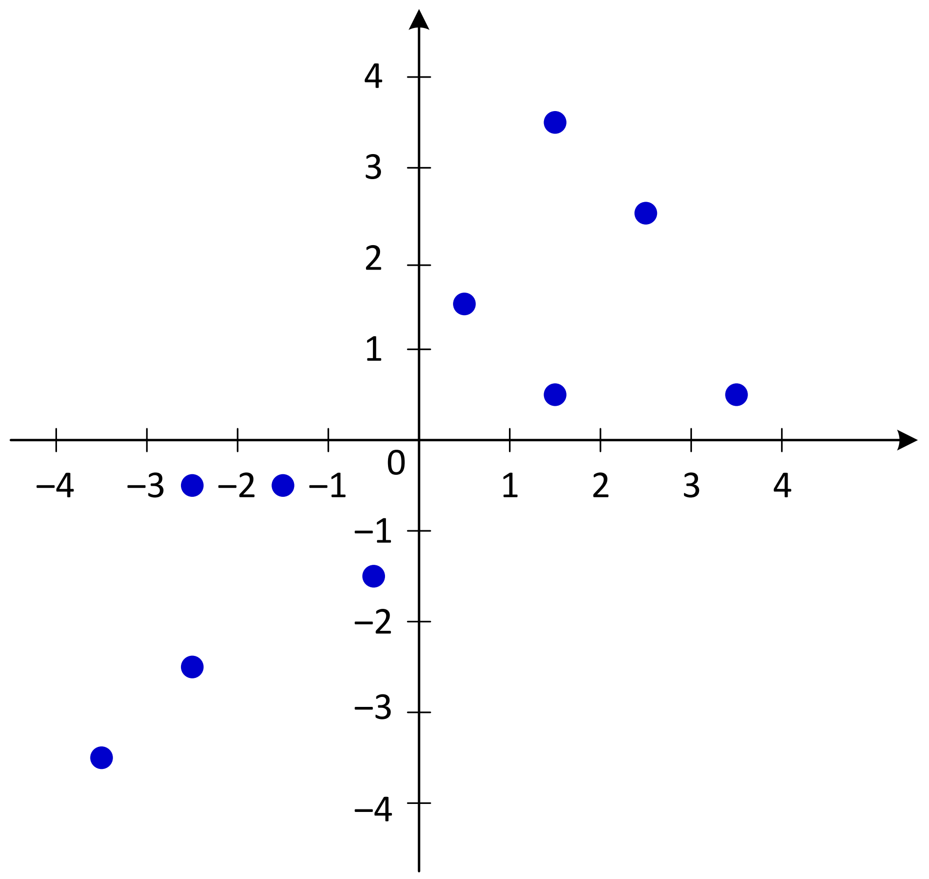
Note that the clusters are not identifiable if we consider just attribute 1 values alone or just attribute 2 values alone.



Principal component analysis - example

We wish to **transform the data** so that its **properties are more evident**.

First **we shift the data down** and to the left **to centre it on the origin**.

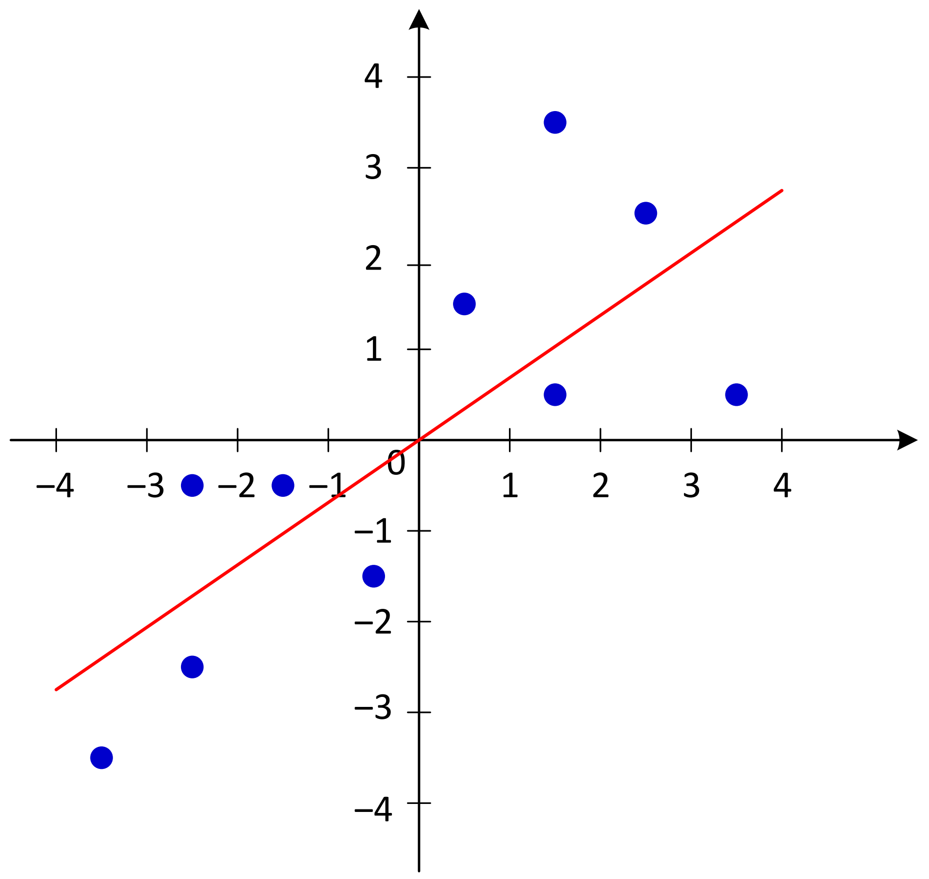


Principal component analysis - example

We wish to transform the data so that its properties are more evident.

First we shift the data down and to the left to centre it on the origin.

Next we **obtain the direction** which shows the **greatest variability** the data.



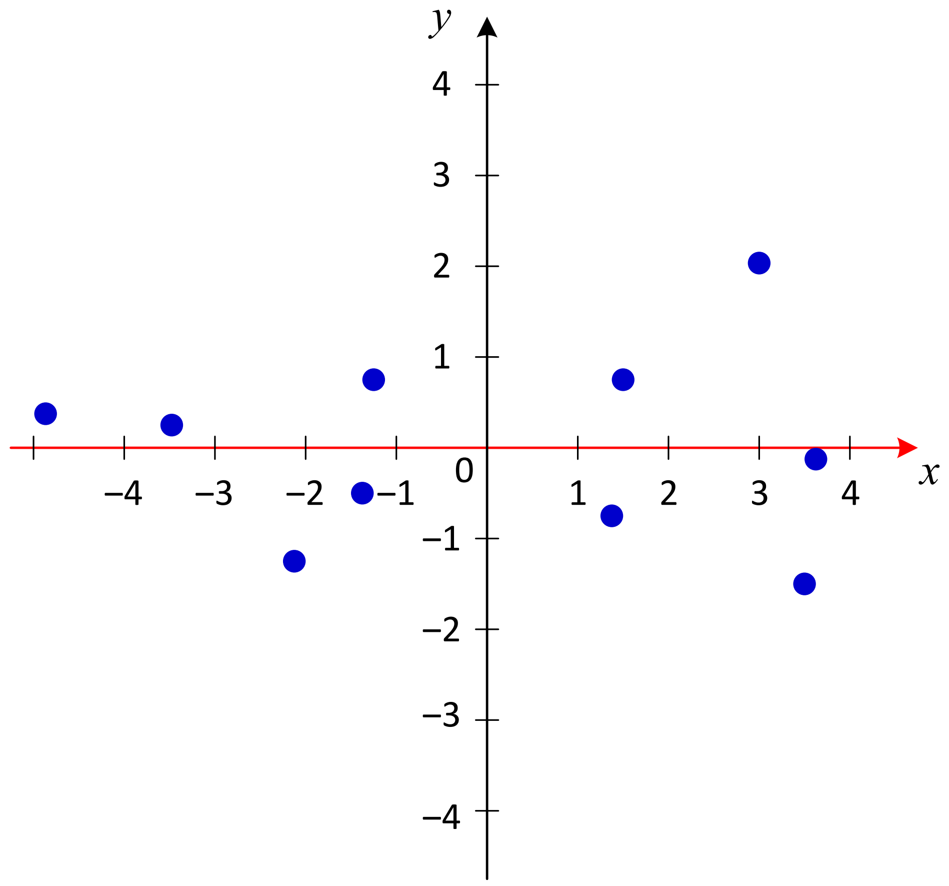
Principal component analysis - example

We wish to transform the data so that its properties are more evident.

First we shift the data down and to the left to centre it on the origin.

Next we obtain the direction which shows the greatest variability the data.

Finally we **transform the data** so that the **maximum variation aligns** to the **x-axis**.



Principal component analysis - example

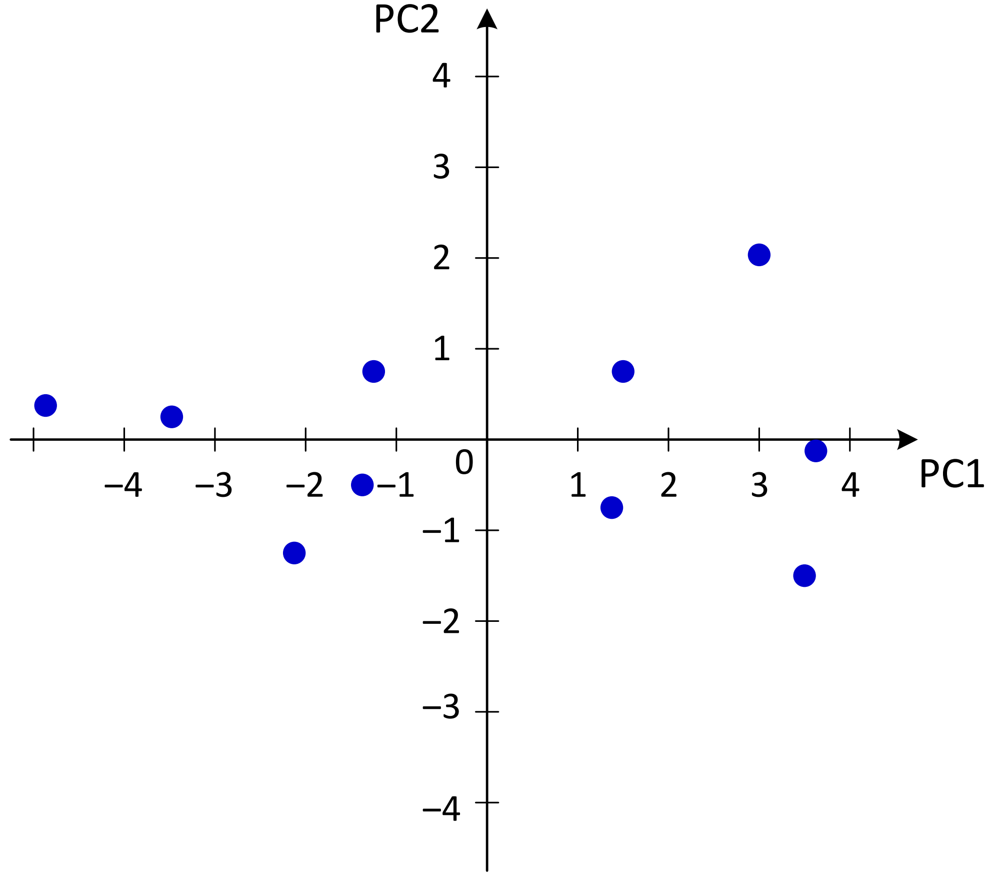
In the transformed data:

- x is principle component 1 (PC1)
- y is principal component 2 (PC2).

PC1 represents the maximum variability in the data.

In this case (as we will see) PC1 represents 89% of the variability in the data and PC2 represents 11%.

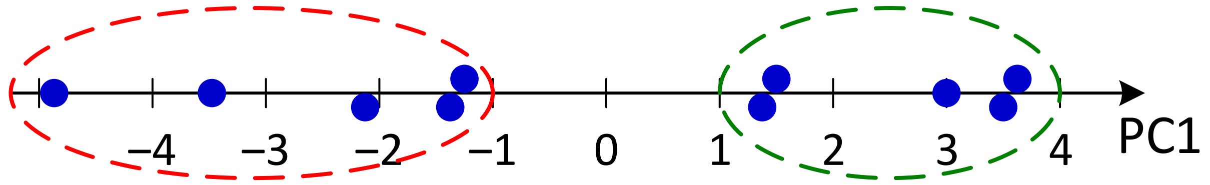
Thus, if we needed to use a single variable to represent the data, we would be advised to use PC1.



Principal component analysis - example

Thus, if we needed to use a single variable to represent the data, we would be advised to use PC1.

For example, just using PC1, we can still identify the two clusters in the data.



PCA – mathematics

We will work through (some of) the mathematics of PCA using our example data (shown in the table) and labelling attribute 1 as x and attribute 2 as y.

First calculate the mean of x and mean of y:

$$\bar{x} = \frac{1}{n}\sum x_i = \frac{1+2+3+\cdots+7+6}{10} = 4.4$$

$$\bar{y} = \frac{1}{n}\sum y_i = \frac{2+3+5+\cdots+8+6}{10} = 5.5$$

Attribute 1 (x)	Attribute 2 (y)
1	2
2	3
3	5
2	5
6	9
8	6
4	4
5	7
7	8
6	6

Centring the data

Summary: $\bar{x} = 4.4, \ \bar{y} = 5.5.$

To obtain the data centred at the origin, we replace each point (x_i, y_i) with $(x_i - \bar{x}, y_i - \bar{y})$.

Note that, for the centred data,

$$\bar{x} = 0 \text{ and } \bar{y} = 0.$$

x (centred)	y (centred)
–3.4	–3.5
–2.4	–2.5
–1.4	–0.5
–2.4	–0.5
1.6	3.5
3.6	0.5
–0.4	–1.5
0.6	1.5
2.6	2.5
1.6	0.5

Covariance matrix

Next we calculate the covariance matrix

$$\Sigma = \begin{pmatrix} \mathrm{var}(x) & \mathrm{cov}(x, y) \\ \mathrm{cov}(x, y) & \mathrm{var}(y) \end{pmatrix}$$

where var(x) is the variance of x

$$\mathrm{var}(x) = \frac{\sum(x_i - \bar{x})^2}{n-1} = 5.6$$

var(y) is the variance of y

$$\mathrm{var}(y) = \frac{\sum(y_i - \bar{y})^2}{n-1} = 4.722$$

Covariance matrix

and cov(x, y) is the covariance of x and y

$$\mathrm{cov}(x, y) = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{n-1} = 4.$$

In summary, the covariance matrix is

$$\Sigma = \begin{pmatrix} \mathrm{var}(x) & \mathrm{cov}(x, y) \\ \mathrm{cov}(x, y) & \mathrm{var}(y) \end{pmatrix} = \begin{pmatrix} 5.6 & 4 \\ 4 & 4.722 \end{pmatrix}$$

Note: it does not matter whether we use the original data or the centred data to calculate the covariance matrix; they give the same values.

Eigenvalues and eigenvectors

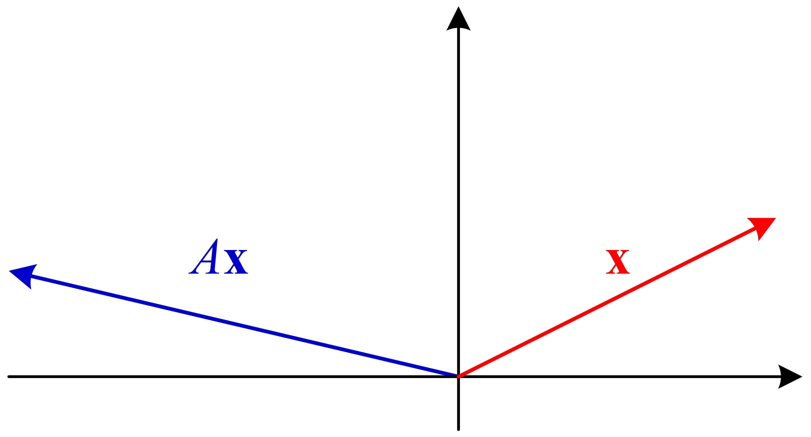
Next we need to find the eigenvalues λ and the eigenvectors $\mathbf{x}$ of the covariance matrix Σ. The eigenvectors give us the principal components.

First however we explain what are the eigenvalues and eigenvectors of a general (2 x 2) matrix $A = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}$.

Eigenvalues and eigenvectors

In general, multiplying a vector $\mathbf{x}$ by a (2 x 2) matrix A gives a new vector $A\mathbf{x}$ with a *different magnitude (size)* to $\mathbf{x}$ and pointing in a *different direction* to $\mathbf{x}$.

Ax pointing in different direction with different magnitude to vector x.



x – vector

A – 2x2 matrix

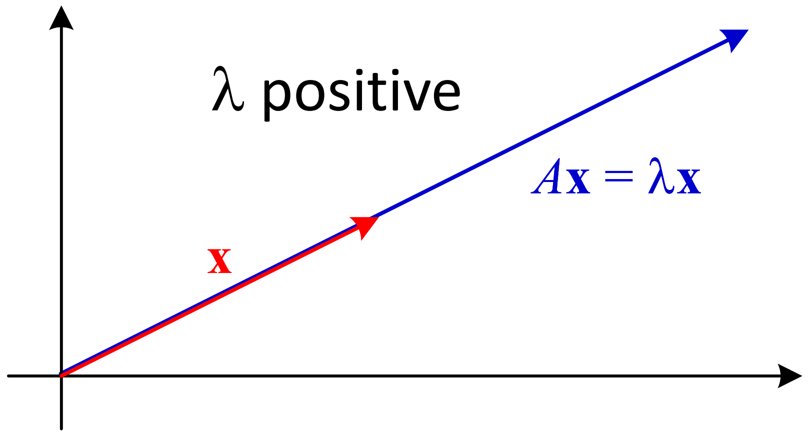
Eigenvalues and eigenvectors

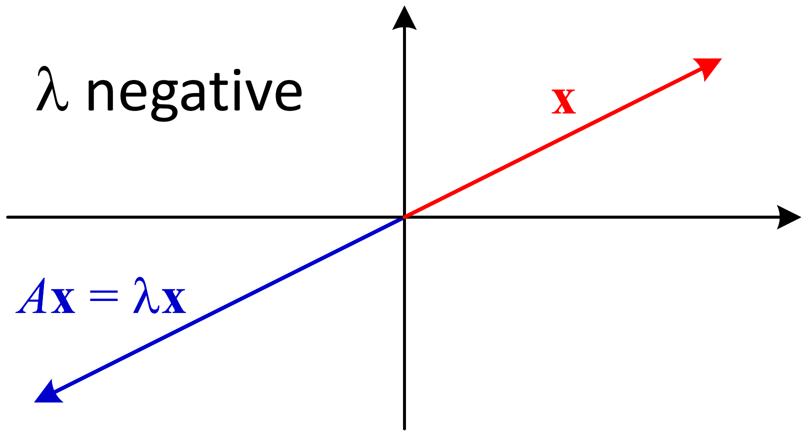
Sometimes there are **special vectors $\mathbf{x}$** where $A\mathbf{x}$ is in the *same direction* as $\mathbf{x}$. This means that

$$A\mathbf{x} = \lambda\mathbf{x}$$

so that $A\mathbf{x}$ is just a multiple (λ) of $\mathbf{x}$.

In this situation, λ is called an eigenvalue of A and $\mathbf{x}$ is a corresponding eigenvector.



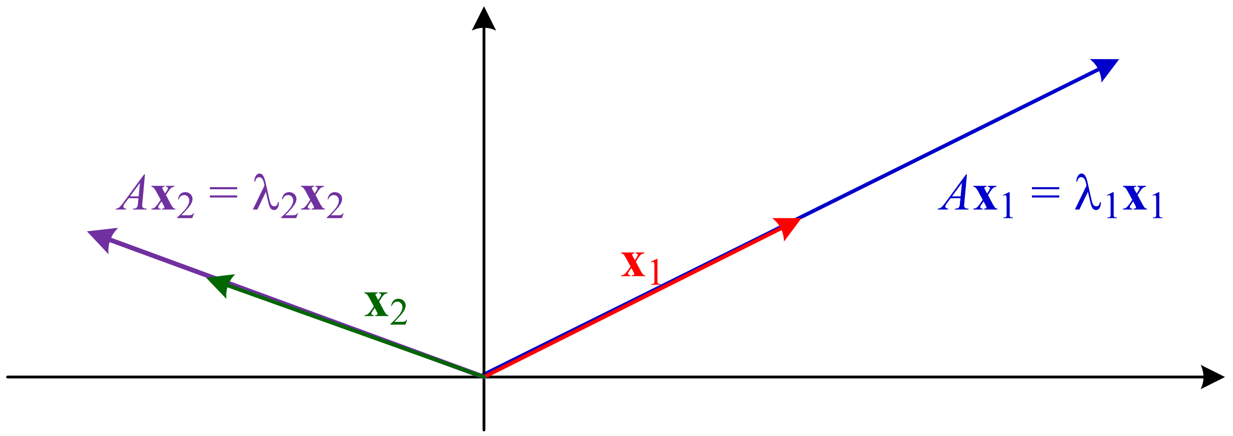


Eigenvalues and eigenvectors

Note that a multiple of an eigenvector $\mathbf{x}$ is also an eigenvector.

For example, if $A\mathbf{x} = \lambda\mathbf{x}$ **then** $A(2\mathbf{x}) = \lambda(2\mathbf{x})$ so $2\mathbf{x}$ is also an eigenvector.

For a 2 x 2 matrix A there will be (at most) two different eigenvalues, λ_1 and λ_2, and two different eigenvector directions.


$A\mathbf{x}_2 = \lambda_2\mathbf{x}_2$
$\mathbf{x}_2$
$\mathbf{x}_1$
$A\mathbf{x}_1 = \lambda_1\mathbf{x}_1$

Eigenvalues and eigenvectors of Σ

We need to find the eigenvalues and eigenvectors for our covariance matrix

$$\Sigma = \begin{pmatrix} 5.6 & 4 \\ 4 & 4.722 \end{pmatrix}.$$

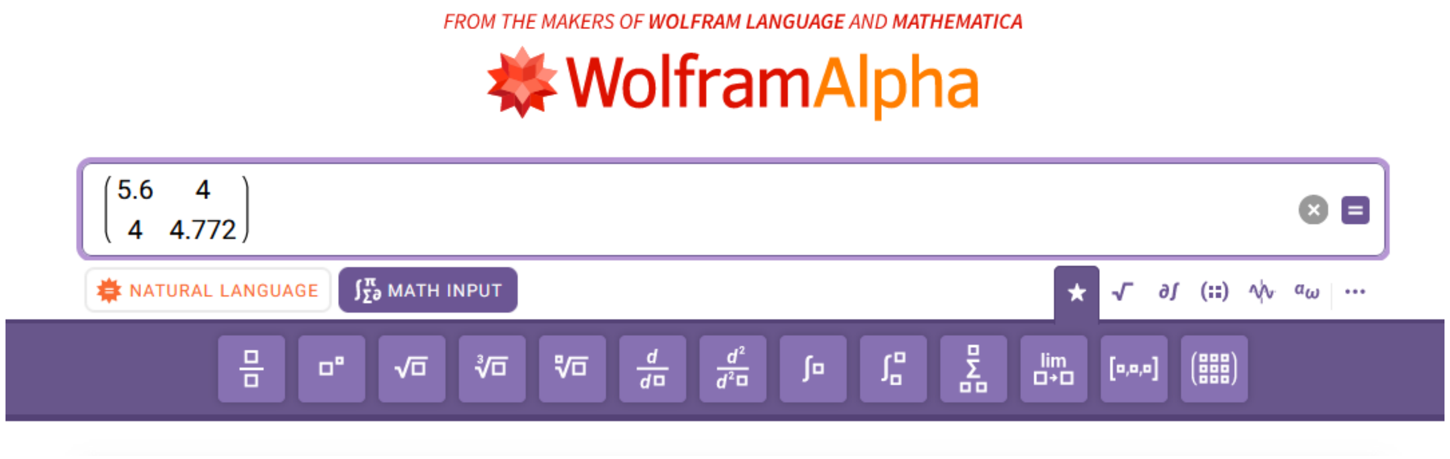
We can do this using numpy:

```
import numpy as np
from numpy.linalg import eig

a =  np.array([[5.6, 4],
               [4, 4.722]])
w,v=eig(a)
print('E-value:', w)
print('E-vector:', v)
```

Eigenvalues and eigenvectors of Σ

We could also do this using www.wolframalpha.com :



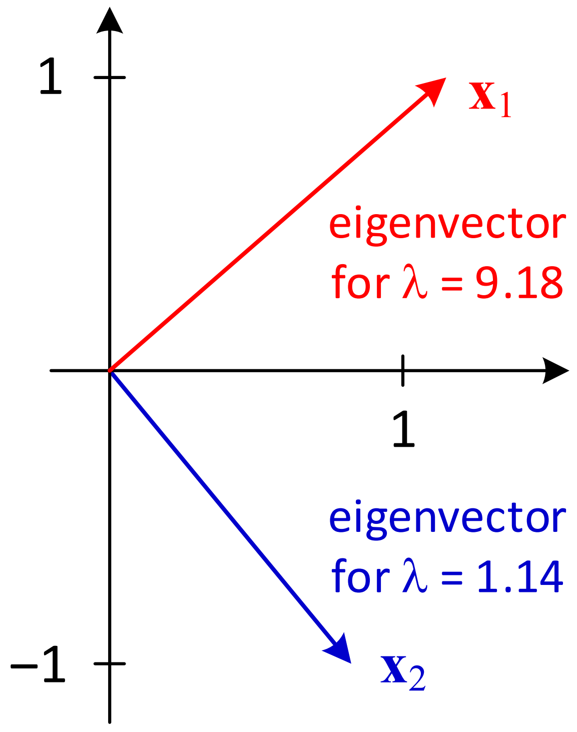
$$\begin{pmatrix} 5.6 & 4 \\ 4 & 4.772 \end{pmatrix}$$

Enter the matrix and hit return.

Eigenvalues and eigenvectors of Σ

Either way, we should find:

- eigenvalues: 9.18 and 1.14 (subject to rounding)
- $\lambda_1 = 9.18$ has eigenvector $\begin{pmatrix} 1.12 \\ 1 \end{pmatrix}$
- $\lambda_2 = 1.14$ has eigenvector $\begin{pmatrix} 0.896 \\ -1 \end{pmatrix}$



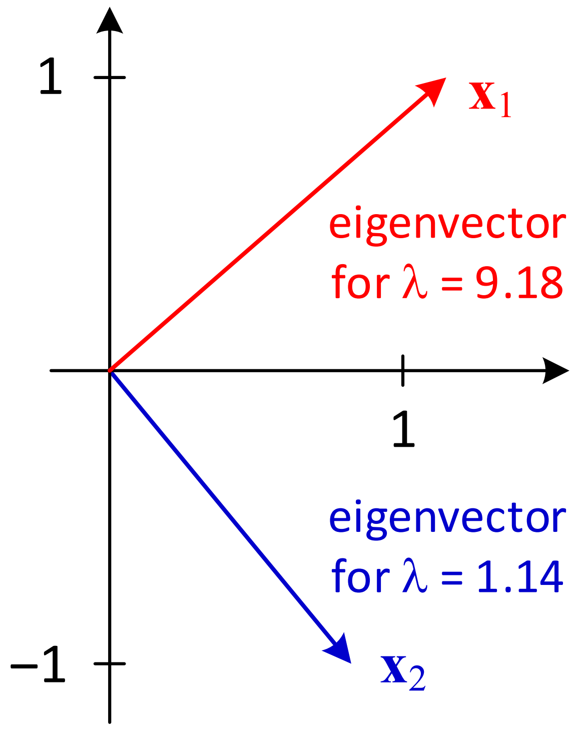
Eigenvalues and eigenvectors of Σ

Note that the eigenvectors are perpendicular (or orthogonal in mathematical terminology).

This is verified by showing their dot product is zero:

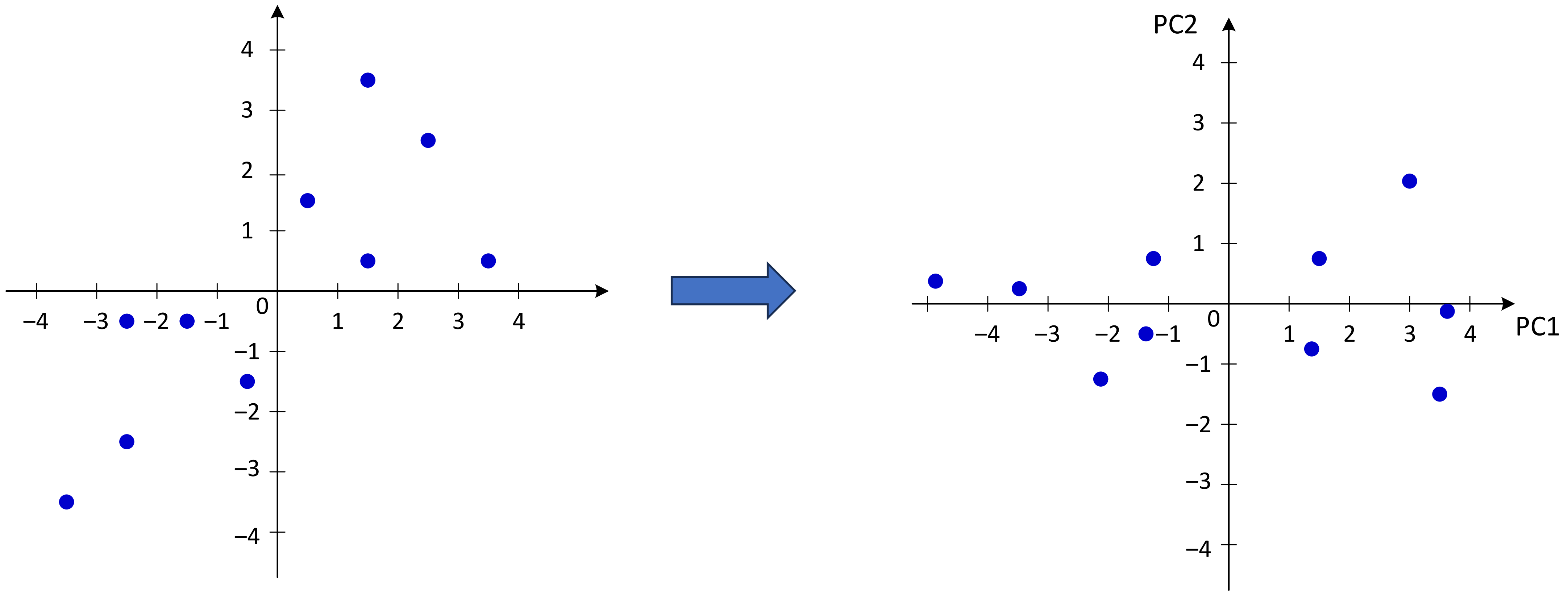
$$\begin{pmatrix}1.12\\1\end{pmatrix}\cdot\begin{pmatrix}0.896\\-1\end{pmatrix} = 1.12\times0.896 + 1\times(-1) = 0$$

It is *always* the case for any symmetric matrix, such as a covariance matrix, that the eigenvectors for different eigenvalues are perpendicular.


1
$\mathbf{x}_1$
eigenvector for λ = 9.18
1
eigenvector for λ = 1.14
−1
$\mathbf{x}_2$

Transforming the data

The final step is to transform the (centred) data.



Transforming the data

Firstly we normalise the eigenvectors to obtain unit vectors (vectors with length 1).

In general, to normalise a vector:

$$\begin{pmatrix} u \\ v \end{pmatrix} \xrightarrow{\text{normalise}} \begin{pmatrix} u/\sqrt{u^2+v^2} \\ v/\sqrt{u^2+v^2} \end{pmatrix}.$$

For our eigenvectors we have:

$$\begin{pmatrix} 1.12 \\ 1 \end{pmatrix} \xrightarrow{\text{normalise}} \begin{pmatrix} 0.745 \\ 0.667 \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} 0.896 \\ -1 \end{pmatrix} \xrightarrow{\text{normalise}} \begin{pmatrix} 0.667 \\ -0.745 \end{pmatrix}.$$

Transforming the data

To transform the data, form the matrix whose **columns are the normalised eigenvectors:**

$$(\text{eigenvector}(\lambda_1) \quad \text{eigenvector}(\lambda_2)) = \begin{pmatrix} 0.745 & 0.667 \\ 0.667 & -0.745 \end{pmatrix}.$$

Multiply the row vector $\mathbf{v}$ for each **centred data point by this matrix $\boldsymbol{A}$** by to give the (vector of the) transformed point $\mathbf{v}A$.

For example, the first centred data point $(-3.4, -3.5)$ is transformed to $(-4.87, 0.34)$ as follows:

$$(-3.4 \quad -3.5)\begin{pmatrix} 0.745 & 0.667 \\ 0.667 & -0.745 \end{pmatrix} = (-4.87 \quad 0.34)$$

Transforming the data

This can be done for all points in one go by multiplying the data matrix D by A as follows:

$$DA = \begin{pmatrix} -3.4 & -3.5 \\ -2.4 & -2.5 \\ -1.4 & -0.5 \\ -2.4 & -0.5 \\ 1.6 & 3.5 \\ 3.6 & 0.5 \\ -0.4 & -1.5 \\ 0.6 & 1.5 \\ 2.6 & 2.5 \\ 1.6 & 0.5 \end{pmatrix} \begin{pmatrix} 0.745 & 0.667 \\ 0.667 & -0.745 \end{pmatrix} = \begin{pmatrix} -4.87 & 0.34 \\ -3.46 & 0.26 \\ -1.38 & -0.56 \\ -2.12 & -1.23 \\ 3.53 & -1.54 \\ 3.01 & 2.03 \\ -1.30 & 0.85 \\ 1.48 & -0.72 \\ 3.60 & -0.13 \\ 1.53 & 0.70 \end{pmatrix}$$

Transforming the data

In summary:

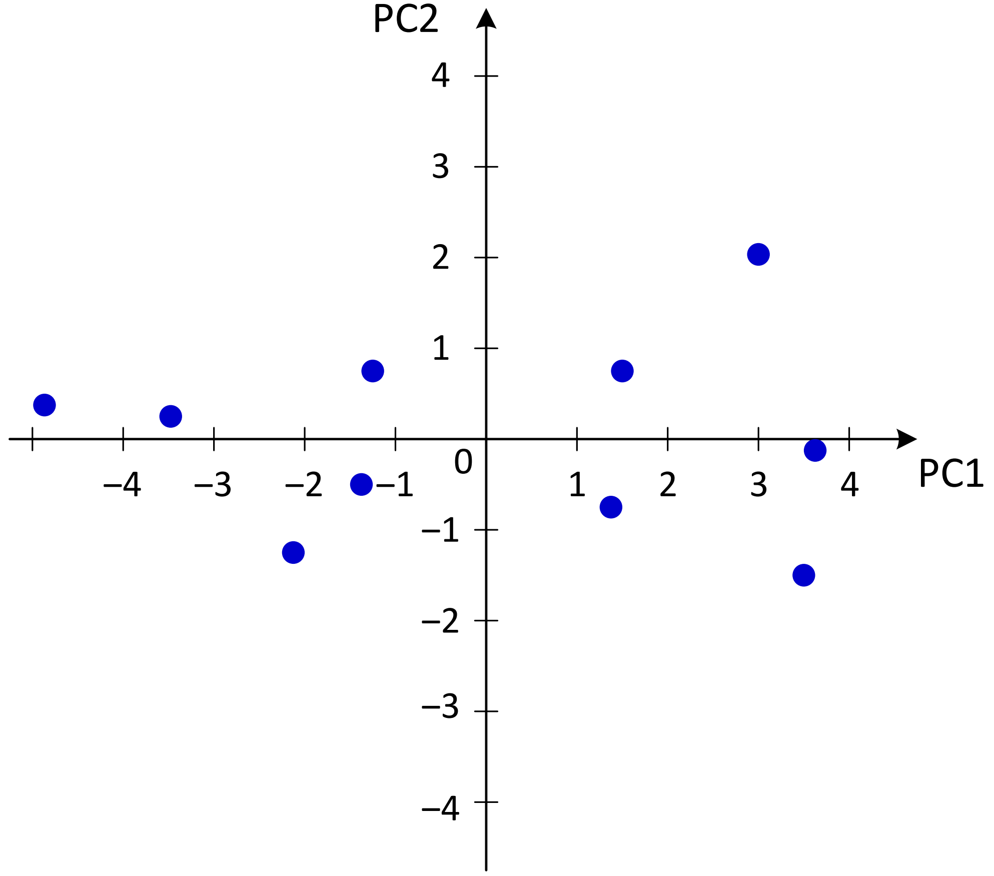
x (centred)	y (centred)		PC1	PC2
–3.4	–3.5	$\rightarrow$	–4.87	0.34
–2.4	–2.5	$\rightarrow$	–3.46	0.26
–1.4	–0.5	$\rightarrow$	–1.38	–0.56
–2.4	–0.5	$\rightarrow$	–2.12	–1.23
1.6	3.5	$\rightarrow$	3.53	–1.54
3.6	0.5	$\rightarrow$	3.01	2.03
–0.4	–1.5	$\rightarrow$	–1.30	0.85
0.6	1.5	$\rightarrow$	1.45	–0.72
2.6	2.5	$\rightarrow$	3.60	–0.13
1.6	0.5	$\rightarrow$	1.53	0.70

Transforming the data

Finally we can plot the transformed data.

PC1 is the first principal component and captures most of the variation of the data.

So if we needed to represent the data with a single variable only, PC1 is the best choice. As we have noted, the two clusters are visible using PC1 only when they were not visible using either of the original variables.



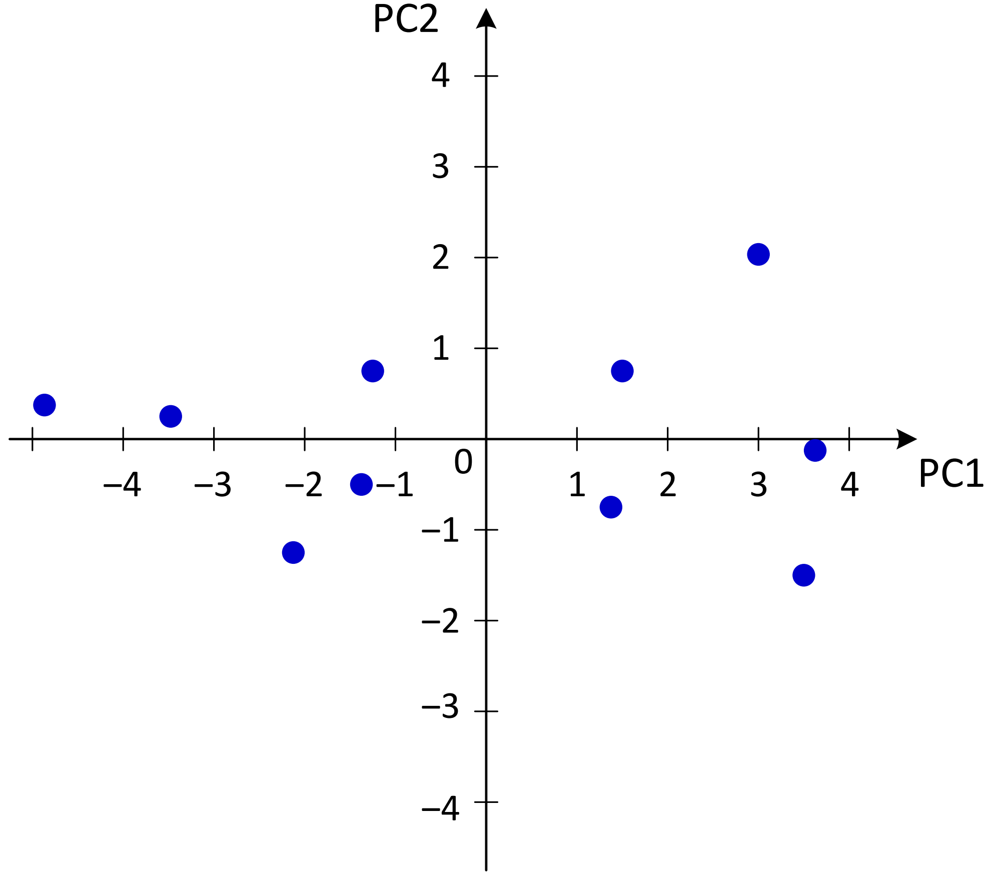
Transforming the data

If the eigenvalues are λ_1 and λ_2 (where $\lambda_1 > \lambda_2$) then

- PC1 contributes $\frac{\lambda_1}{\lambda_1+\lambda_2}$ of the variation
- PC2 contributes $\frac{\lambda_2}{\lambda_1+\lambda_2}$ of the variation.

In our case

- PC1 contributes $\frac{9.18}{10.32} = 89\%$ of the variation
- PC2 contributes $\frac{1.14}{10.32} = 11\%$ of the variation.



More attributes

The data in the previous example only had two attributes which is unlikely to be the case for real data which may have tens, hundreds or even thousands of attributes.

We can still perform the same data transformation to identify the principal components (PC1, PC2, PC3, etc.) and identify how much of the total variation of the data each principal component accounts for.

We can then **decide to retain only the number of principal components** that account for however much of the variation we are concerned about.

We will work through another example with 5 components (but not showing all the mathematical calculations!)

Example: 5 attributes

Consider the following data where the attributes are labelled x_1, x_2, x_3, x_4, x_5.

We cannot easily visualise these data as we would need 5 dimensions to draw their 'graph'.

We can, however, perform a similar analysis and identify the principal components.

$\boldsymbol{x_1}$	$\boldsymbol{x_2}$	$\boldsymbol{x_3}$	$\boldsymbol{x_4}$	$\boldsymbol{x_5}$
2	5	6	3	10
4	7	8	0	5
7	12	16	3	−1
9	15	17	−5	−8
2	6	7	4	9
1	7	2	6	7
6	9	11	1	1
5	11	11	1	0
8	15	15	−2	−7
6	13	12	0	−1

Centre the data

The first step is to centre the data by calculating the mean $\bar{x}_k$ of each attribute x_k and replacing each value $x_{i,k}$ with $x_{i,k} - \bar{x}_k$.

For x_1:

$$\bar{x}_1 = \frac{1}{n}\sum x_{i,1} = \frac{1}{10}(2+4+7+\cdots+8+6) = 5$$

so

$$\begin{array}{ccc} 2 & \rightarrow & 2-5=-3 \\ 4 & \rightarrow & 4-5=-1 \\ 7 & \rightarrow & 7-5=2 \\ \vdots & \vdots & \vdots \\ 8 & \rightarrow & 8-5=3 \\ 6 & \rightarrow & 6-5=1 \end{array}$$

x_1	x_2	x_3	x_4	x_5
2	5	6	3	10
4	7	8	0	5
7	12	16	3	−1
9	15	17	−5	−8
2	6	7	4	9
1	7	2	6	7
6	9	11	1	1
5	11	11	1	0
8	15	15	−2	−7
6	13	12	0	−1

Centre the data

Doing this for each attribute:

	x_1	x_2	x_3	x_4	x_5
	2	5	6	3	10
	4	7	8	0	5
	7	12	16	3	−1
	9	15	17	−5	−8
	2	6	7	4	9
	1	7	2	6	7
	6	9	11	1	1
	5	11	11	1	0
	8	15	15	−2	−7
	6	13	12	0	−1
mean	5	10	10.5	1.1	1.5

$\rightarrow$

x_1	x_2	x_3	x_4	x_5
−3	−5	−4.5	1.9	8.5
−1	−3	−2.5	-1.1	3.5
2	2	5.5	1.9	−2.5
4	5	6.5	−6.1	−9.5
−3	−4	−3.5	2.9	7.5
−4	−3	−8.5	4.9	5.5
1	−1	0.5	−0.1	−0.5
0	1	0.5	−0.1	−1.5
3	5	4.5	−3.1	−8.5
1	3	1.5	−1.1	−2.5

centred data

Covariance matrix

Next we calculate the covariance matrix

$$\Sigma = \begin{pmatrix} var(x_1) & cov(x_1, x_2) & cov(x_1, x_3) & cov(x_1, x_4) & cov(x_1, x_5) \\ cov(x_2, x_1) & var(x_2) & cov(x_2, x_3) & cov(x_2, x_4) & cov(x_2, x_5) \\ cov(x_3, x_1) & cov(x_3, x_2) & var(x_3) & cov(x_3, x_4) & cov(x_3, x_5) \\ cov(x_4, x_1) & cov(x_4, x_2) & cov(x_4, x_3) & var(x_4) & cov(x_4, x_5) \\ cov(x_5, x_1) & cov(x_5, x_2) & cov(x_5, x_3) & cov(x_5, x_4) & var(x_5) \end{pmatrix}$$

where $var(x_k) = \frac{1}{n-1} \sum (x_k - \bar{x}_k)^2$

and $cov(x_j, x_k) = \frac{1}{n-1} \sum (x_j - \bar{x}_j)\,(x_k - \bar{x}_k).$

Covariance matrix

Note that, since $var(x_j, x_k) = var(x_k, x_j)$, the covariance matrix is symmetric.

In our case

$$\Sigma = \begin{pmatrix} 7.33 & 9.22 & 12.56 & -7.11 & -16.11 \\ 9.22 & 13.78 & 15.56 & -8.67 & -22.56 \\ 12.56 & 15.56 & 22.94 & -11.39 & -26.50 \\ -7.11 & -8.67 & -11.39 & 9.88 & 15.94 \\ -16.11 & -22.56 & -26.50 & 15.94 & 38.72 \end{pmatrix}.$$

Eigenvalues

Using numpy or Wolfram Alpha, we find that the eigenvalues of Σ are

$$\lambda_1 = 85.28, \qquad \lambda_2 = 3.62, \qquad \lambda_3 = 3.29, \qquad \lambda_4 = 0.41, \qquad \lambda_5 = 0.05.$$

Percentage of total variation

Let $sum = \lambda_1 + \lambda_2 + \lambda_3 + \lambda_4 + \lambda_5 = 92.65.$

When we calculate the principal components, this means that

PC1 accounts for $\lambda_1/sum = \frac{85.28}{92.65} = 92.05\%$ of the total variation;

PC2 accounts for $\lambda_2/sum = \frac{3.62}{92.65} = 3.91\%$ of the total variation;

PC3 accounts for $\lambda_3/sum = \frac{3.29}{92.65} = 3.55\%$ of the total variation;

PC4 accounts for $\lambda_4/sum = \frac{0.41}{92.65} = 0.44\%$ of the total variation;

PC5 accounts for $\lambda_5/sum = \frac{0.05}{92.65} = 0.05\%$ of the total variation.

Percentage of total variation

Depending on the application it might be sensible, for example, to consider:

- just PC1 which accounts for 92.05% of the total variation;
- PC1, PC2 and PC3 which, between them, account for 99.5% of the total variation.

Whatever the circumstances, it will be possible to discard PC4 and PC5.

It is in this way that PCA can be considered as a dimension reduction technique.

We now see how to calculate the principal components.

Principal components

To calculate the principal components we need eigenvectors $\mathbf{v}$ such that

$$\Sigma\mathbf{v} = \lambda\mathbf{v}$$

for each eigenvalue λ.

The principal components are the eigenvectors (for the eigenvalues in decreasing order).

We used the online calculator

https://www.wolframalpha.com/calculators/eigenvalue-calculator

will find the eigenvectors.

Normalised eigenvectors

When normalised this gave the following.

Eigenvalue	85.28	3.62	3.29	0.41	0.05
Eigenvector (normalised)	$\begin{pmatrix} -0.289 \\ -0.389 \\ -0.490 \\ 0.287 \\ 0.665 \end{pmatrix}$	$\begin{pmatrix} 0.180 \\ -0.276 \\ 0.814 \\ 0.129 \\ 0.461 \end{pmatrix}$	$\begin{pmatrix} 0.057 \\ -0.341 \\ -0.076 \\ -0.920 \\ 0.166 \end{pmatrix}$	$\begin{pmatrix} -0.381 \\ 0.758 \\ 0.116 \\ -0.231 \\ 0.462 \end{pmatrix}$	$\begin{pmatrix} 0.858 \\ 0.287 \\ -0.279 \\ 0.003 \\ 0.322 \end{pmatrix}$

Transforming the data

First construct the matrix A whose columns are the normalised eigenvectors:

$$A = \begin{pmatrix} -0.289 & 0.180 & 0.057 & -0.381 & 0.858 \\ -0.389 & -0.276 & -0.341 & 0.758 & 0.287 \\ -0.490 & 0.814 & -0.076 & 0.116 & -0.279 \\ 0.287 & 0.129 & -0.920 & -0.231 & 0.003 \\ 0.665 & 0.461 & 0.166 & 0.462 & 0.322 \end{pmatrix}.$$

Next multiply the data matrix of the centred data D by A to form DA.

Transforming the data

$$DA = \underset{\substack{\uparrow \\ \text{centred data}}}{\begin{pmatrix} -3 & -5 & -4.5 & 1.9 & 8.5 \\ -1 & -3 & -2.5 & -1.1 & 3.5 \\ 2 & 2 & 5.5 & 1.9 & -2.5 \\ 4 & 5 & 6.5 & -6.1 & -9.5 \\ -3 & -4 & -3.5 & 2.9 & 7.5 \\ -4 & -3 & -8.5 & 4.9 & 5.5 \\ 1 & -1 & 0.5 & -0.1 & -0.5 \\ 0 & 1 & 0.5 & -0.1 & -1.5 \\ 3 & 5 & 4.5 & -3.1 & -8.5 \\ 1 & 3 & 1.5 & -1.1 & -2.5 \end{pmatrix}} \begin{pmatrix} -0.289 & 0.180 & 0.057 & -0.381 & 0.858 \\ -0.389 & -0.276 & -0.341 & 0.758 & 0.287 \\ -0.490 & 0.814 & -0.076 & 0.116 & -0.279 \\ 0.287 & 0.129 & -0.920 & -0.231 & 0.003 \\ 0.665 & 0.461 & 0.166 & 0.462 & 0.322 \end{pmatrix} = \underset{\substack{\uparrow \\ \text{transformed data}}}{\begin{pmatrix} 11.22 & 1.34 & 1.54 & 0.32 & -0.01 \\ 4.69 & 0.08 & 2.75 & -0.31 & 0.10 \\ -5.17 & 3.38 & -3.15 & -0.20 & -0.04 \\ -14.36 & -0.54 & 2.07 & 0.03 & -0.02 \\ 9.96 & 1.55 & 0.04 & 0.50 & -0.32 \\ 11.55 & -3.64 & -2.16 & -0.32 & -0.14 \\ -0.51 & 0.62 & 0.37 & -1.29 & 0.27 \\ -1.66 & -0.57 & -0.54 & 0.15 & -0.34 \\ -11.56 & -1.50 & -0.43 & -0.05 & 0.01 \\ -4.17 & -0.72 & -0.48 & 1.16 & 0.49 \end{pmatrix}}$$

Transformed data

In summary, the transformed data is:

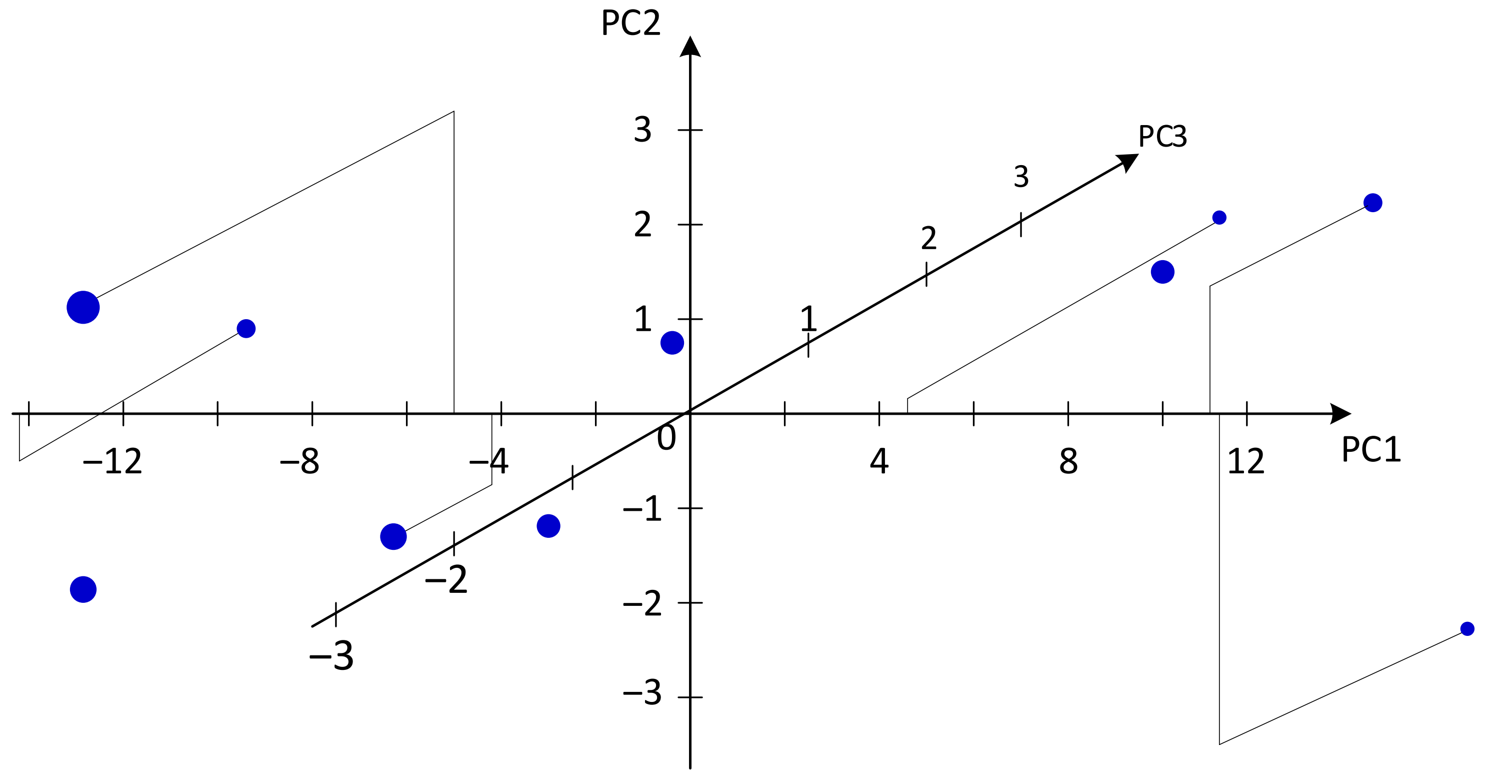
PC1	PC2	PC3	PC4	PC5
11.22	1.34	1.54	0.32	-0.01
4.69	0.08	2.75	-0.31	0.1
-5.17	3.38	-3.15	-0.2	-0.04
-14.36	-0.54	2.07	0.03	-0.02
9.96	1.55	0.04	0.5	-0.32
11.55	-3.64	-2.16	-0.32	-0.14
-0.51	0.62	0.37	-1.29	0.27
-1.66	-0.57	-0.54	0.15	-0.34
-11.56	-1.5	-0.43	-0.05	0.01
-4.17	-0.72	-0.48	1.16	0.49

Transformed data

Recall that PC1, PC2 and PC3 account for 99.5% of the variation in the data.

Here we attempt to draw the 3D graph showing PC1, PC2 and PC3 with points ‘further away’ shown smaller.

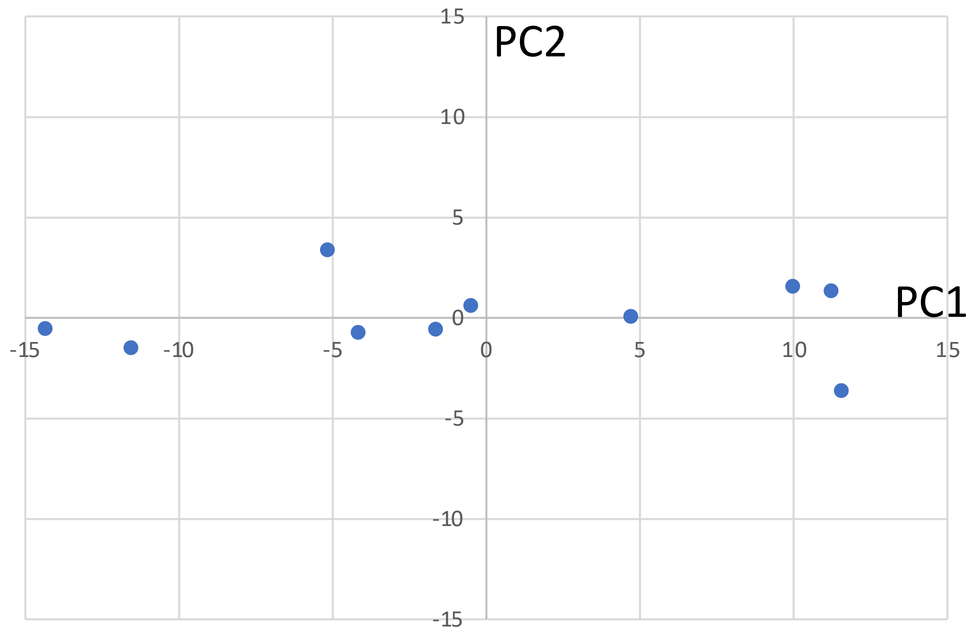
This has only limited success at visualising the data!

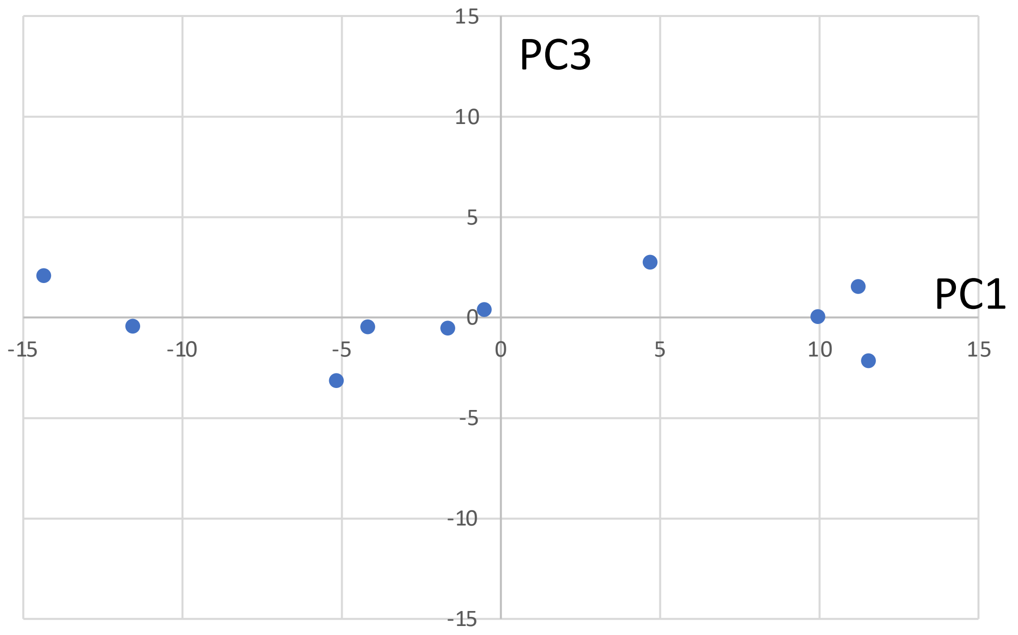


Transformed data

Recall that PC1, PC2 and PC3 account for 99.5% of the variation in the data.

Here we show 2D graphs of PC1 vs PC2 and PC1 vs PC3 which are clearer.





PCA in Python

```
from sklearn.decomposition import PCA

pca=PCA(n_componenets=10)

x_pca=pca.fit_transform(data)
```

Number of Principal Components

Application- Clustering Image Data using PCA

- Image data can also be clustered so that images are grouped together with similar characteristics
- Efficient ways of doing this is to use semantic information from the images (as features) and use these features to group images together based on the images.
- An algorithm to do this Principal Component Analysis which reduces dimensionality by selecting features to be used in machine learning.

Loading the image dataset

This dataset contains face images of celebrities downloaded from the internet. The dataset is skewed containing more pictures G.W.Bush and C.Powell. This dataset is used to train algorithms for facial recognition algorithms.

from sklearn.datasets **import** fetch_lwf_people

people = fetch_lfw_people(min_faces_per_person=70, resize=0.4)

Clustering Images using DBSCAN

Principal Component Analysis (PCA) is a technique which finds semantic features in images. The eigenvalues representation is used for the representation of the data to find interesting structure.

To use PCA we use

from **sklearn.decomposition** import **PCA**

pca=**PCA**(n_componenets=100)

x_pca=pca.fit_transform(people)

This is a semantic representation of the face images and not the raw pixels.

It will also make the computation faster

DBSCAN clustering

DBSCAN without defining epsilon and min_samples

```
dbscan =DBSCAN()
labels=dbscan.fit_predict(x_pca)
print(f"unique labels: {np.unique(labels)}")
```

Output

```
[-1]
```

Everything is labelled as noise!

```
dbscan =DBSCAN(min_samples=3, eps=15)
labels=dbscan.fit_predict(x_pca)
print(f"unique labels: {np.unique(labels)}")
```

Output

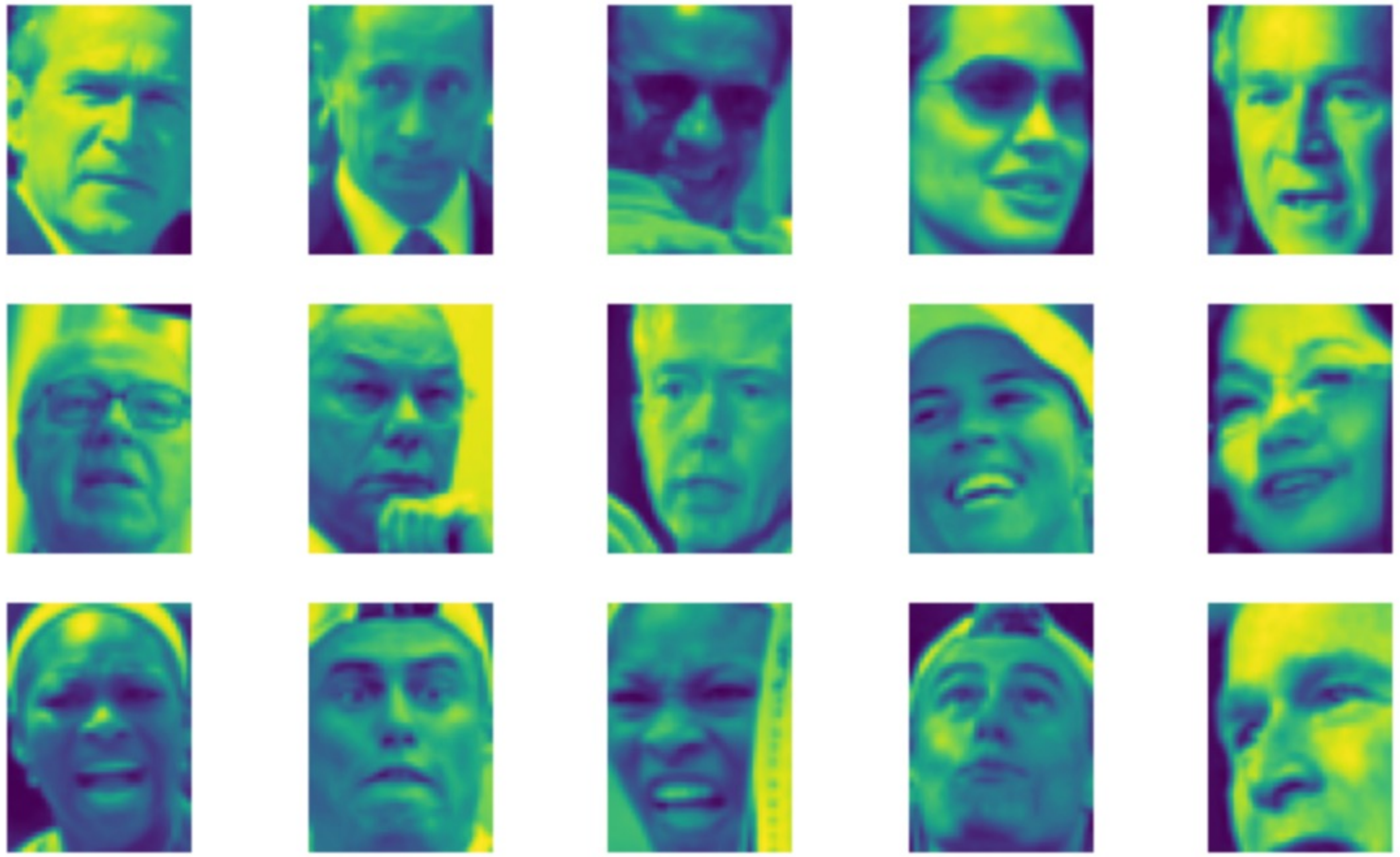
```
Unique labels [-1,0]
```

Using a much larger eps of 15 we get only a single cluster and noise points. We can use this result to find out what noise looks like compared to the rest of the data.

Noise (outlier) images

Outliers are the ones which
are not normal images. A common
trend with the outliers are the following:

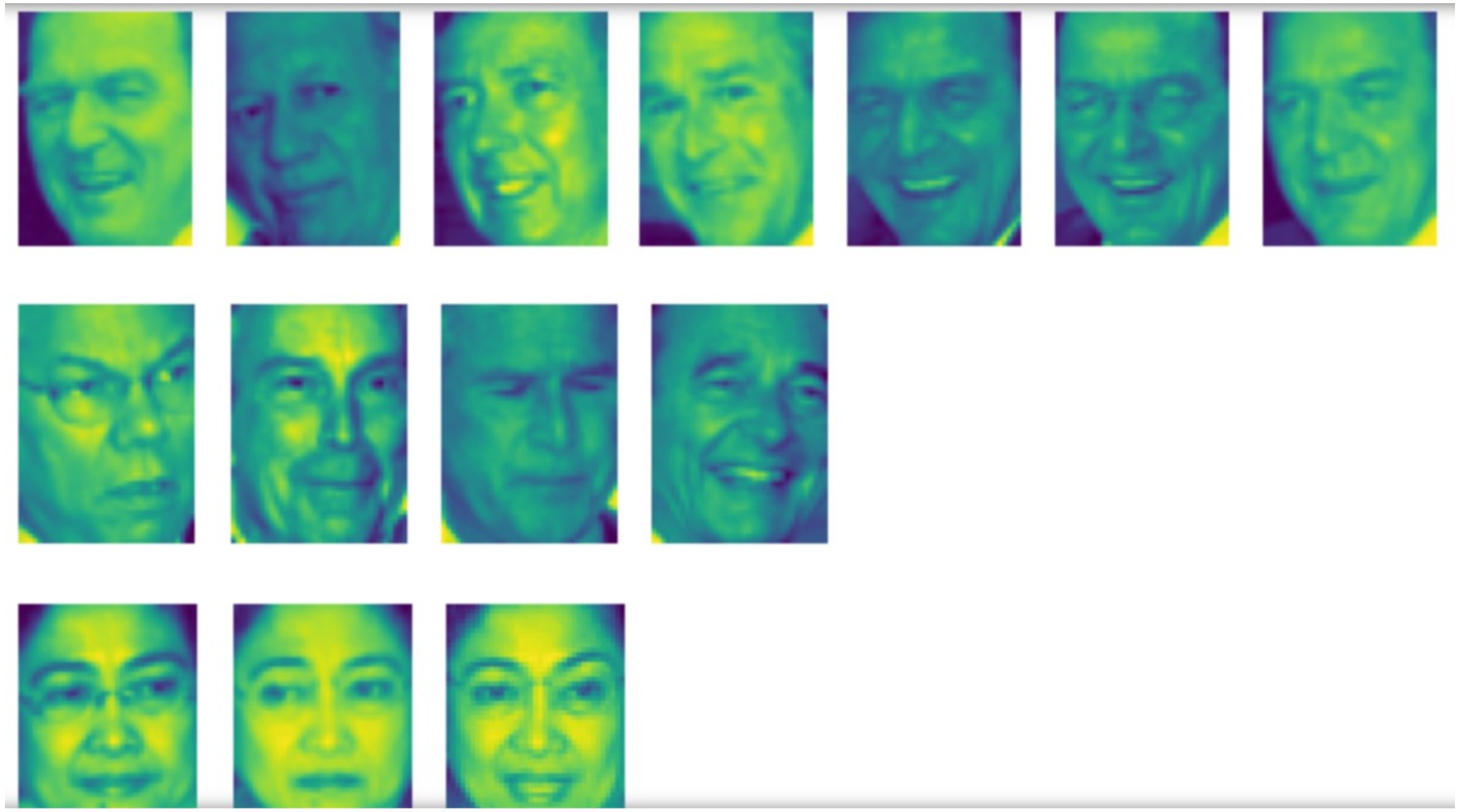
- Hands in front of the face
- Wearing a hat
- Wearing glasses
- Head in odd angles



- If we want to find more interesting clusters than just one large one, we need to set eps to smaller, somewhere between 15 and 0.5.

- The results for eps=7 are more interesting, with many small clusters. We can investigate this clusterings more closely by visualizing the points for each of the small clusters

Subset of the clusters



- Some of the clusters correspond to people with very distinct faces.
- Within each cluster the orientation of the face is quiet fixed, as well as the facial expressions.
- Some of the clusters contain faces but they share a similar orientation and expression
- This has been done manually but a much more automatic search approach would be needed to evaluate the parameters using the accuracy

Singular Value Decomposition

Singular value decomposition

Standard technique in data analysis.

Also used for data reduction.

SVD: ‘data matrix’

Suppose we have a ‘data matrix’ A which is an $n \times p$ matrix.

There are n rows each of which can be considered as a ‘data point’.

Each row has p columns so that each data point has p attributes.

$$A = \begin{pmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,p} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,p} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n,1} & x_{n,2} & \cdots & x_{n,p} \end{pmatrix}$$

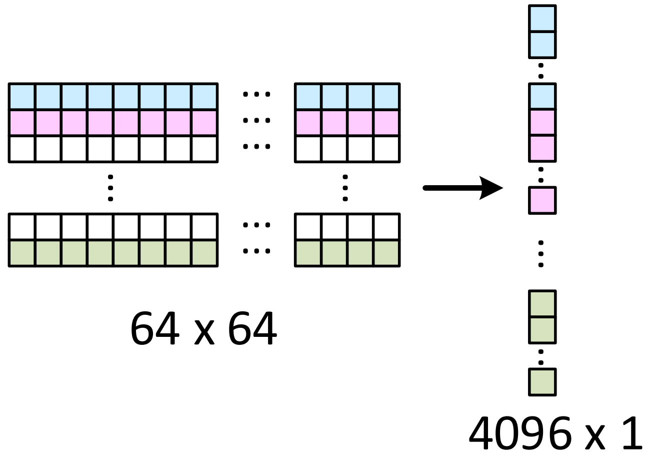
SVD: examples

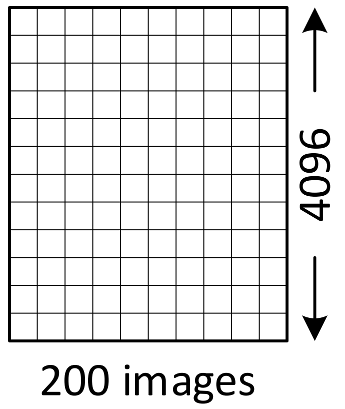
Some examples of possible data matrices.

Each column of A is a vector that represents an image of a person's face say.

For example, a small monochrome 64 x 64 pixel image could be represented by a column vector with 64 x 64 = 4096 entries where each entry represents the shade of the corresponding pixel.

We can then represent a dataset of 200 images as a 4096 x 200 matrix A.





SVD: examples

Suppose we wish to build a recommender system that recommends movies to users.

The system will be based on actual ratings of movies by users.

The data matrix A contains all the ratings of movies by users:

	Movie 1	Movie 2	Movie 3	…	Movie p
User 1	$R_{1,1}$	$R_{1,2}$	$R_{1,3}$	…	$R_{1,p}$
User 2	$R_{2,1}$	$R_{2,2}$	$R_{2,3}$	…	$R_{2,p}$
User 3	$R_{3,1}$	$R_{3,2}$	$R_{3,3}$	…	$R_{3,p}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$		$\vdots$
User n	$R_{n,1}$	$R_{n,2}$	$R_{n,3}$	…	$R_{n,p}$

SVD: overview

The basic structure of SVD is that we replace $A_{n\times p}$ (where $n \geq p$) with

$$A_{n\times p} = U_{n\times n}\,\Sigma_{n\times p}\,V_{p\times p}^{\mathrm{T}}$$

where the subscripts denote the dimensions of the matrices and V^{T} denotes the transpose of V.

Here

- U is orthogonal: $U^{\mathrm{T}}U = I_n$ (identity matrix)
- V is orthogonal: $V^{\mathrm{T}}V = I_p$ (identity matrix)
- Σ is ‘diagonal’ with entries $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_p$:

$$\Sigma = \begin{pmatrix} \sigma_1 & 0 & \cdots & 0 \\ 0 & \sigma_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sigma_p \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 \end{pmatrix}$$

Calculating U, V and Σ

Assume A is $n \times p$ where $n \geq p$.

- Find $A^T A$. This is a symmetric $p \times p$ matrix.
- Find the eigenvalues $\lambda_1, \lambda_2, \dots, \lambda_p$ of $A^T A$ where $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_p$.
- The diagonal entries of Σ are $\sigma_1 = \sqrt{\lambda_1},\ \sigma_2 = \sqrt{\lambda_2}, \dots,\ \sigma_p = \sqrt{\lambda_p}$ called the singular values.

$$\Sigma = \begin{pmatrix} \sigma_1 & 0 & \cdots & 0 \\ 0 & \sigma_2 & & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \cdots & \sigma_p \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 0 \end{pmatrix}$$

Calculating U, V and Σ

- Find the corresponding eigenvectors of $A^T A$.
- Normalise the eigenvectors: $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_p$ (so $\|\mathbf{v}_1\| = \|\mathbf{v}_2\| = \cdots = \|\mathbf{v}_p\| = 1$).
- These are the columns of the matrix V.

$$V = (\mathbf{v}_1 \quad \mathbf{v}_2 \quad \cdots \quad \mathbf{v}_p).$$

- The columns of the matrix V are orthogonal: $\mathbf{v}_i \cdot \mathbf{v}_j = 0$ (when $i \neq j$).
- This means that V is an orthogonal matrix: $V\, V^T = I_p$.

Calculating U, V and Σ

- The matrix AA^T is an $n \times n$ symmetric matrix.
- The eigenvalues of AA^T are $\lambda_1, \lambda_2, \dots, \lambda_p$ (same as $A^T A$) and 0.
- Corresponding normalised ($\|\mathbf{u}_k\| = 1$) eigenvectors are $\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_p, \mathbf{u}_{p+1}, \dots, \mathbf{u}_n$.
- Here $\mathbf{u}_{p+1}, \dots, \mathbf{u}_n$ are all eigenvectors corresponding to eigenvalue 0 (so $(AA^T)\mathrm{u}_{p+1} = \mathbf{0}$ etc.) and must be *chosen* to be orthogonal ($\mathbf{u}_k \cdot \mathbf{u}_l = 0$).
- The columns of U are $\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_p, \mathbf{u}_{p+1}, \dots, \mathbf{u}_n$:

$$U = \begin{pmatrix} \mathbf{u}_1 & \cdots & \mathbf{u}_p & \mathbf{u}_{p+1} & \cdots & \mathbf{u}_n \end{pmatrix}.$$

- The matrix U is also an orthogonal $n \times n$ matrix: $UU^T = I_n$.

Summary

Assume A is an $n \times p$ matrix where $n \geq p$. Then

$$A = U\Sigma V^T$$

where:

- Σ is diagonal followed by a block of zeros with entries $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_p$ called singular values.
- The singular values are the square roots of the eigenvalues of $A^T A$.
- U is orthogonal ($UU^T = I_n$) with columns eigenvectors of AA^T.
- V is orthogonal ($VV^T = I_p$) with columns eigenvectors of $A^T A$.

Example 1

Let $A = \begin{pmatrix} 1 & 1 \\ 2 & 1 \\ 4 & 5 \\ 5 & 3 \end{pmatrix}$. Then $A^T A = \begin{pmatrix} 1 & 2 & 4 & 5 \\ 1 & 1 & 5 & 3 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 2 & 1 \\ 4 & 5 \\ 5 & 3 \end{pmatrix} = \begin{pmatrix} 46 & 38 \\ 38 & 36 \end{pmatrix}$.

$A^T A$ has eigenvalues 79.3275 and 2.6725 (rounded) with corresponding normalised eigenvectors $\begin{pmatrix} 0.7518 \\ 0.6594 \end{pmatrix}$ and $\begin{pmatrix} -0.6594 \\ 0.7518 \end{pmatrix}$.

Hence $V = \begin{pmatrix} 0.7518 & -0.6594 \\ 0.65954 & 0.7518 \end{pmatrix}$.

Example 1

The singular values are $\sqrt{79.3275} = 8.9066$ and $\sqrt{2.6725} = 1.6348$.

Hence

$$\Sigma = \begin{pmatrix} 8.9066 & 0 \\ 0 & 1.6348 \\ 0 & 0 \\ 0 & 0 \end{pmatrix}.$$

Example 1

$$AA^T = \begin{pmatrix} 1 & 1 \\ 2 & 1 \\ 4 & 5 \\ 5 & 3 \end{pmatrix} \begin{pmatrix} 1 & 2 & 4 & 5 \\ 1 & 1 & 5 & 3 \end{pmatrix} = \begin{pmatrix} 2 & 3 & 9 & 8 \\ 3 & 5 & 13 & 13 \\ 9 & 13 & 41 & 35 \\ 8 & 13 & 35 & 34 \end{pmatrix}.$$

This has eigenvalues 79.3275, 2.67246 and 0 with corresponding normalised eigenvectors

$$\begin{pmatrix} 0.15844 \\ 0.24285 \\ 0.70781 \\ 0.64415 \end{pmatrix}, \begin{pmatrix} 0.05655 \\ -0.34680 \\ 0.68608 \\ -0.63704 \end{pmatrix}, \begin{pmatrix} -0.40825 \\ -0.81650 \\ 0 \\ 0.40825 \end{pmatrix}, \begin{pmatrix} -0.89724 \\ 0.39254 \\ 0.16823 \\ -0.11215 \end{pmatrix}.$$

Example 1

Note that the two eigenvectors corresponding to eigenvalue 0 are orthogonal

$$\begin{pmatrix} -0.40825 \\ -0.81650 \\ 0 \\ 0.40824 \end{pmatrix} \cdot \begin{pmatrix} -0.89724 \\ 0.39254 \\ 0.16823 \\ -0.11215 \end{pmatrix} = 0.366 - 0.320 + 0 - 0.046 = 0.$$

The matrix $U = \begin{pmatrix} 0.15844 & 0.05655 & -0.40825 & -0.89724 \\ 0.24285 & -0.34680 & -0.81650 & 0.39254 \\ 0.70781 & 0.68608 & 0 & 0.16823 \\ 0.64415 & -0.63704 & 0.40824 & -0.11215 \end{pmatrix}$

Example 1

We can check the equation $A = U\Sigma V^T$.

$U\Sigma V^T =$

$$\begin{pmatrix} 0.15844 & 0.05655 & -0.40825 & -0.89724 \\ 0.24285 & -0.34680 & -0.81650 & 0.39254 \\ 0.70781 & 0.68608 & 0 & 0.16823 \\ 0.64415 & -0.63704 & 0.40824 & -0.11215 \end{pmatrix} \begin{pmatrix} 8.9066 & 0 \\ 0 & 1.6348 \\ 0 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 0.7518 & 0.6594 \\ -0.65954 & 0.7518 \end{pmatrix} =$$

$$\begin{pmatrix} 1.4112 & 0.0924 \\ 2.1630 & -0.5669 \\ 6.3041 & 1.1216 \\ 5.7372 & -1.0414 \end{pmatrix} \begin{pmatrix} 0.7518 & 0.6594 \\ -0.65954 & 0.7518 \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 2 & 1 \\ 4 & 5 \\ 5 & 3 \end{pmatrix} = A \text{ (subject to rounding).}$$

SVD using numpy

The following code will calculate the SVD for A in our example.

```
import numpy as np

# Define the matrix A

A = np.array([[1, 1],
              [2, 1],
              [4, 5],
              [5, 3]])

# Compute the SVD

U, Sigma, Vt = np.linalg.svd(A)
```

SVD using numpy

And the following code will give the output.

```
print(“Matrix A:”)
print(A)
print(“\nU matrix:”)
print(U)
print(“\nSigma matrix:”)
print(np.diag(Sigma))
print(“\nVt matrix:”)
print(Vt)
```

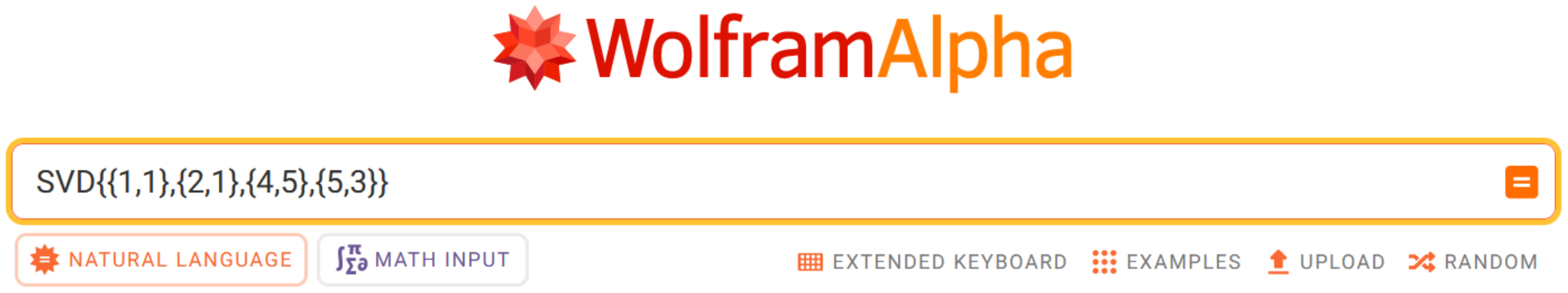
SVD using WolframAlpha

www.wolframalpha.com calculates various properties of matrices

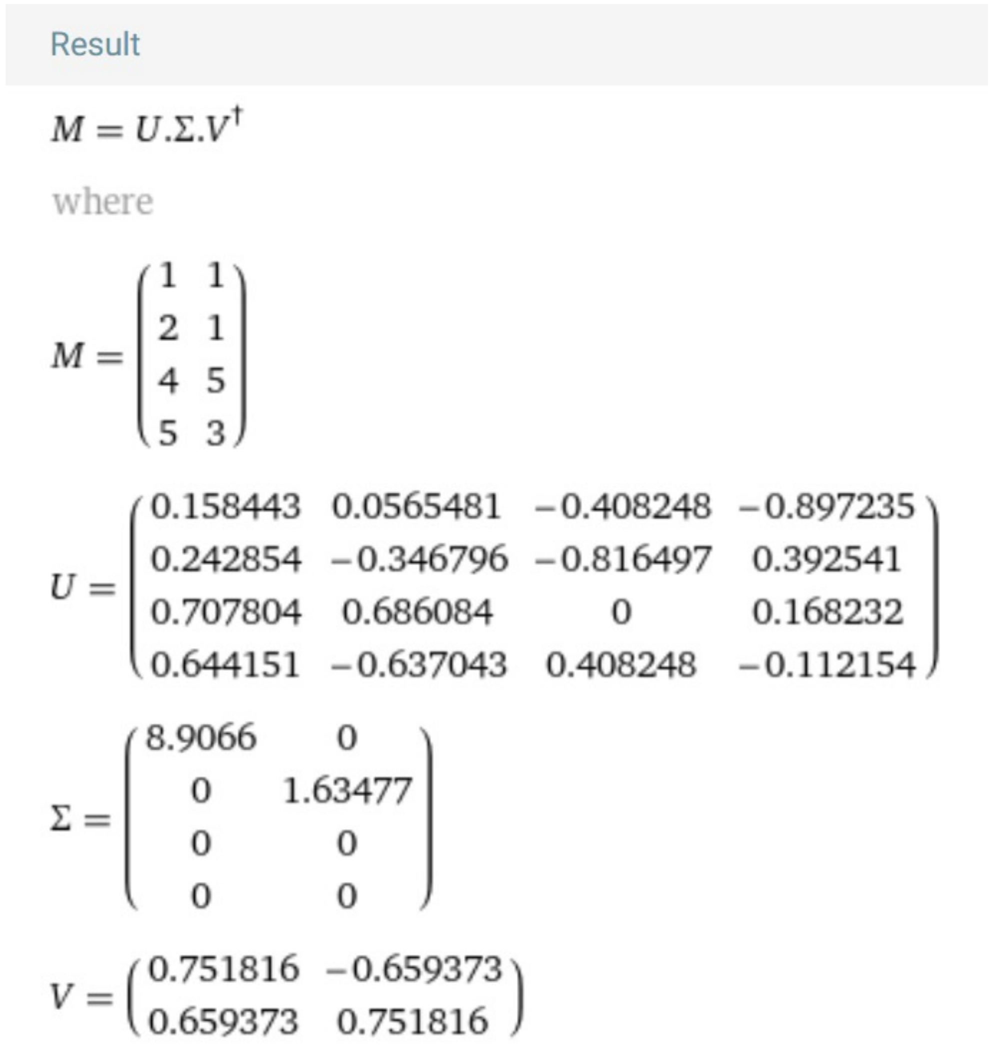
Enter

```
SVD{{1,1},{2,1},{4,5},{5,3}}
```

as shown below produces the output shown on the following slide.


SVD{{1,1},{2,1},{4,5},{5,3}}

SVD using WolframAlpha



Result

$$M = U.\Sigma.V^{\dagger}$$

where

$$M = \begin{pmatrix} 1 & 1 \\ 2 & 1 \\ 4 & 5 \\ 5 & 3 \end{pmatrix}$$

$$U = \begin{pmatrix} 0.158443 & 0.0565481 & -0.408248 & -0.897235 \\ 0.242854 & -0.346796 & -0.816497 & 0.392541 \\ 0.707804 & 0.686084 & 0 & 0.168232 \\ 0.644151 & -0.637043 & 0.408248 & -0.112154 \end{pmatrix}$$

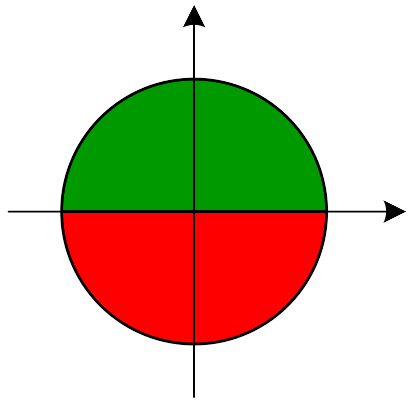
$$\Sigma = \begin{pmatrix} 8.9066 & 0 \\ 0 & 1.63477 \\ 0 & 0 \\ 0 & 0 \end{pmatrix}$$

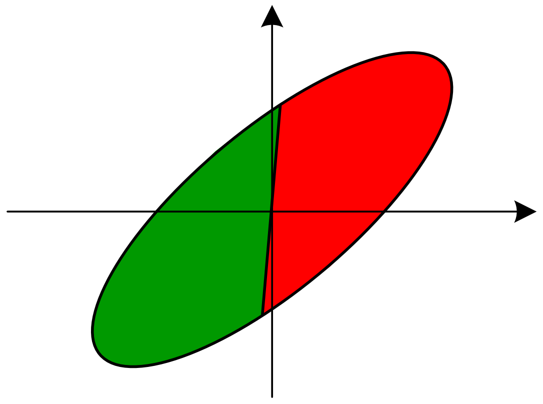
$$V = \begin{pmatrix} 0.751816 & -0.659373 \\ 0.659373 & 0.751816 \end{pmatrix}$$

This matches our calculations earlier.

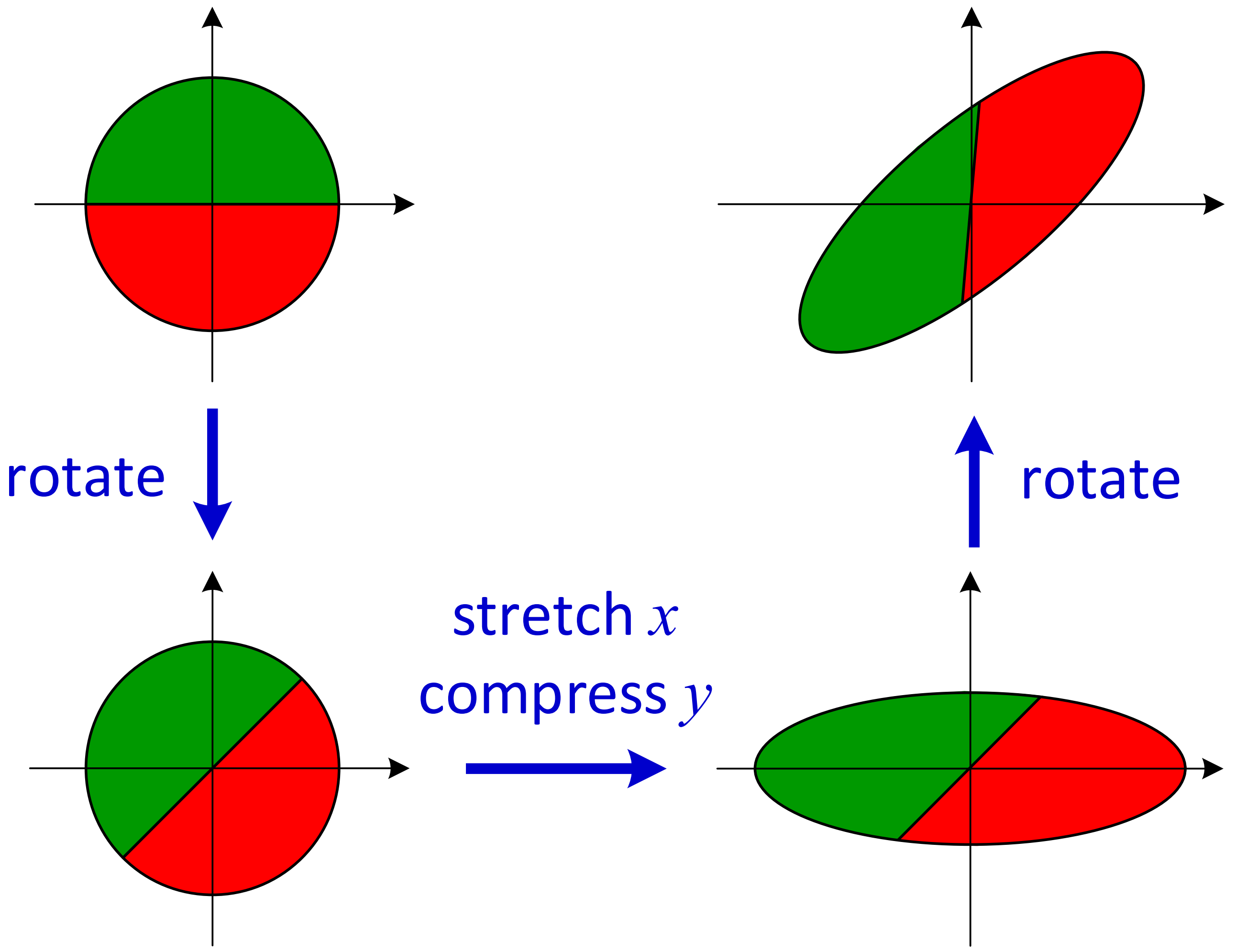
Intuition – transformations of the plane

Challenge: transform the image on the left to the image on the right using only rotations about the origin and horizontal and vertical stretches or compressions.





Intuition – transformations of the plane



Intuition – transformations of the plane

Any linear transformation of the plane $\mathbb{R}^2$ – one that can be represented by matrix multiplication – (that preserves clockwise/anticlockwise) can be represented by

- a rotation followed by
- stretching or compression in x and y directions followed by
- a rotation.

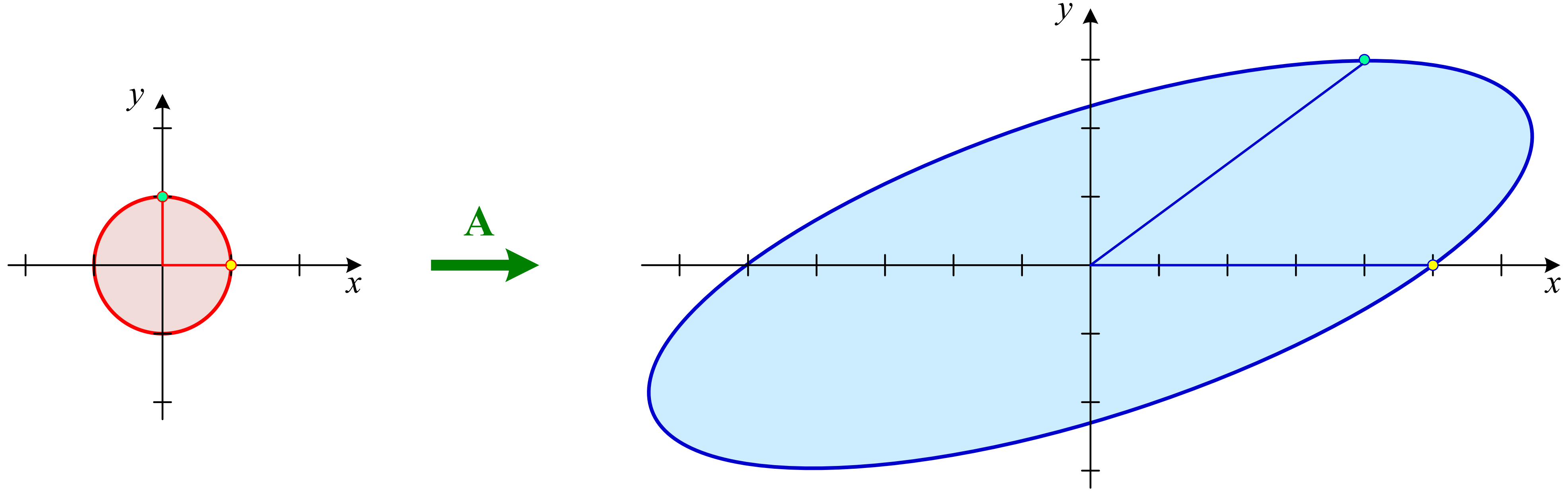
In matrix terms:

$$A = \underset{\substack{\nearrow \\ \text{rotation} \\ \text{matrix}}}{U} \quad \underset{\substack{\uparrow \\ \text{stretch/} \\ \text{compress in} \\ x \text{ and } y \\ \text{directions}}}{\Sigma} \quad \underset{\substack{\nwarrow \\ \text{rotation} \\ \text{matrix}}}{V^T}$$

Intuition – transformations of the plane

Example: $A = \begin{pmatrix} 5 & 4 \\ 0 & 3 \end{pmatrix}$. This defines a mapping $\mathbb{R}^2 \to \mathbb{R}^2$ that sends the (red) unit circle to the (blue) ellipse.

Note: $\begin{pmatrix} 1 \\ 0 \end{pmatrix} \longmapsto \begin{pmatrix} 5 \\ 0 \end{pmatrix}$ and $\begin{pmatrix} 0 \\ 1 \end{pmatrix} \longmapsto \begin{pmatrix} 4 \\ 3 \end{pmatrix}$



Intuition – transformations of the plane

The SVD of A is

$$A = U\Sigma V^T = \begin{pmatrix} {}^{3}\!/_{\sqrt{10}} & {}^{-1}\!/_{\sqrt{10}} \\ {}^{1}\!/_{\sqrt{10}} & {}^{3}\!/_{\sqrt{10}} \end{pmatrix} \begin{pmatrix} 3\sqrt{5} & 0 \\ 0 & \sqrt{5} \end{pmatrix} \begin{pmatrix} {}^{1}\!/_{\sqrt{2}} & {}^{1}\!/_{\sqrt{2}} \\ {}^{-1}\!/_{\sqrt{2}} & {}^{1}\!/_{\sqrt{2}} \end{pmatrix} = \begin{pmatrix} 5 & 4 \\ 0 & 3 \end{pmatrix}.$$

Intuition – transformations of the plane

The SVD of A is

$$A = U\Sigma V^T = \begin{pmatrix} 3/\sqrt{10} & -1/\sqrt{10} \\ 1/\sqrt{10} & 3/\sqrt{10} \end{pmatrix} \begin{pmatrix} 3\sqrt{5} & 0 \\ 0 & \sqrt{5} \end{pmatrix} \begin{pmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ -1/\sqrt{2} & 1/\sqrt{2} \end{pmatrix} = \begin{pmatrix} 5 & 4 \\ 0 & 3 \end{pmatrix}.$$

- V^T is a rotation by -45° (45 degrees clockwise)

Intuition – transformations of the plane

The SVD of A is

$$A = U\Sigma V^T = \begin{pmatrix} 3/\sqrt{10} & -1/\sqrt{10} \\ 1/\sqrt{10} & 3/\sqrt{10} \end{pmatrix} \begin{pmatrix} 3\sqrt{5} & 0 \\ 0 & \sqrt{5} \end{pmatrix} \begin{pmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ -1/\sqrt{2} & 1/\sqrt{2} \end{pmatrix} = \begin{pmatrix} 5 & 4 \\ 0 & 3 \end{pmatrix}.$$

- V^T is a rotation by -45° (45 degrees clockwise)
- Σ is a stretch of $3\sqrt{5} = 6.71$ in the x direction and a stretch of $\sqrt{5} = 2.24$ in the y direction

Intuition – transformations of the plane

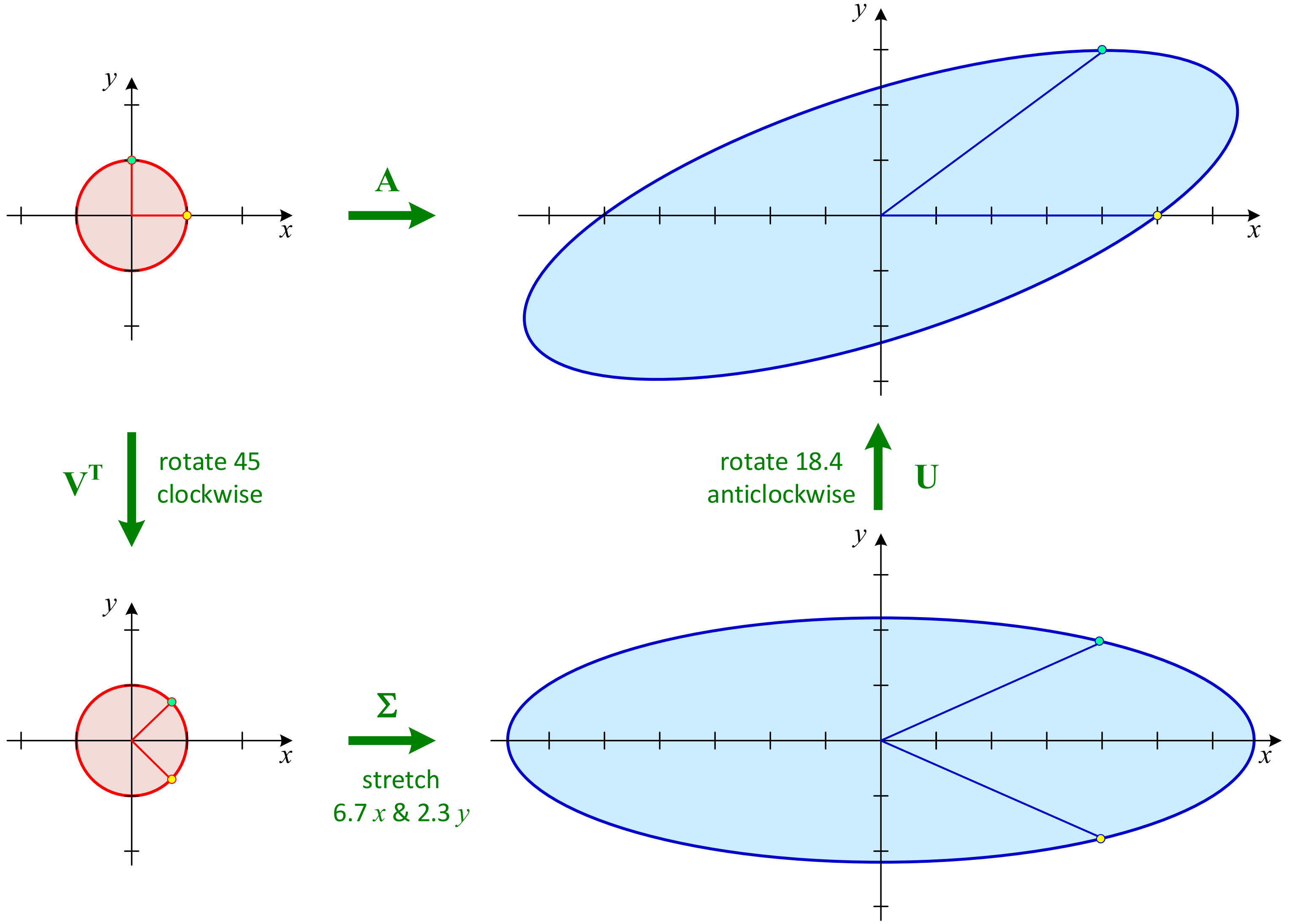
The SVD of A is

$$A = U\Sigma V^T = \begin{pmatrix} 3/\sqrt{10} & -1/\sqrt{10} \\ 1/\sqrt{10} & 3/\sqrt{10} \end{pmatrix} \begin{pmatrix} 3\sqrt{5} & 0 \\ 0 & \sqrt{5} \end{pmatrix} \begin{pmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ -1/\sqrt{2} & 1/\sqrt{2} \end{pmatrix} = \begin{pmatrix} 5 & 4 \\ 0 & 3 \end{pmatrix}.$$

- V^T is a rotation by -45° (45 degrees clockwise)
- Σ is a stretch of $3\sqrt{5} = 6.71$ in the x direction and a stretch of $\sqrt{5} = 2.24$ in the y direction
- U is a rotation of $\tan^{-1}\frac{1}{3} = 18.4^\circ$ (anticlockwise).

Intuition – transformations of the plane

- V^T is a rotation by 45° clockwise
- Σ is stretch of 6.71 in x direction and stretch of 2.24 in y direction
- U is a rotation of 18.4° anticlockwise.



Example 2

For a larger example, let $A = \begin{pmatrix} 1 & 2 & 0 \\ 2 & 3 & 1 \\ 4 & 2 & 1 \\ 8 & 5 & 2 \\ 7 & 5 & 2 \\ 4 & 6 & 3 \\ 6 & 4 & 1 \end{pmatrix}$.

(We will not show the details of the calculations.)

The singular values are: $\sigma_1 = 17.69$, $\sigma_2 = 3.3028$, $\sigma_3 = 1.07515$.

Example 2

Hence

$$\Sigma = \begin{pmatrix} 17.69 & 0 & 0 \\ 0 & 3.3028 & 0 \\ 0 & 0 & 1.07515 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}.$$

Example 2

Also

$$U = \begin{pmatrix} 0.111248 & 0.21065 & -0.745839 & -0.561951 & 0.204591 & 0.16911 & 0.0269986 \\ 0.201801 & 0.329185 & -0.253063 & 0.187317 & -0.865578 & 0.018268 & -0.0471735 \\ 0.253791 & -0.264536 & 0.263846 & -0.65561 & -0.259674 & -0.50472 & -0.209759 \\ 0.498638 & -0.130709 & 0.0683791 & 0 & 0 & 0 & 0.854166 \\ 0.541683 & -0.326068 & 0.125975 & 0 & 0 & 0.665479 & -0.376201 \\ 0.417009 & 0.769258 & 0.330771 & 0 & 0.299018 & -0.111174 & -0.152201 \\ 0.408085 & -0.249244 & -0.424397 & 0.468293 & 0.228198 & -0.510983 & -0.242394 \end{pmatrix}$$

Again the columns/eigenvectors in blue correspond to the eigenvalue 0.

Example 2

Finally

$$V = \begin{pmatrix} 0.761476 & -0.645229 & 0.0619242 \\ 0.60325 & 0.670482 & -0.431906 \\ 0.237159 & 0.366242 & 0.89979 \end{pmatrix}$$

Example 2

By direct calculation, we can verify that

$$U\Sigma V^T = \begin{pmatrix} 0.111248 & 0.21065 & -0.745839 & -0.561951 & 0.204591 & 0.16911 & 0.0269986 \\ 0.201801 & 0.329185 & -0.253063 & 0.187317 & -0.865578 & 0.018268 & -0.0471735 \\ 0.253791 & -0.264536 & 0.263846 & -0.65561 & -0.259674 & -0.50472 & -0.209759 \\ 0.498638 & -0.130709 & 0.0683791 & 0 & 0 & 0 & 0.854166 \\ 0.541683 & -0.326068 & 0.125975 & 0 & 0 & 0.665479 & -0.376201 \\ 0.417009 & 0.769258 & 0.330771 & 0 & 0.299018 & -0.111174 & -0.152201 \\ 0.408085 & -0.249244 & -0.424397 & 0.468293 & 0.228198 & -0.510983 & -0.242394 \end{pmatrix} \begin{pmatrix} 17.69 & 0 & 0 \\ 0 & 3.3028 & 0 \\ 0 & 0 & 1.07515 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} 0.761476 & 0.60325 & 0.237159 \\ -0.645229 & 0.670482 & 0.366242 \\ 0.0619242 & -0.431906 & 0.89979 \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 2 & 0 \\ 2 & 3 & 1 \\ 4 & 2 & 1 \\ 8 & 5 & 2 \\ 7 & 5 & 2 \\ 4 & 6 & 3 \\ 6 & 4 & 1 \end{pmatrix} = A$$

Example 2

Note that we can ignore

- the $n-p$ columns of U corresponding to eigenvalue 0 and
- the last $n-p$ rows of zeros in Σ

as these do not contribute to the product $U\Sigma$.

So we can replace U and Σ with

$$\bar{U} = \begin{pmatrix} 0.111248 & 0.21065 & -0.745839 \\ 0.201801 & 0.329185 & -0.253063 \\ 0.253791 & -0.264536 & 0.263846 \\ 0.498638 & -0.130709 & 0.0683791 \\ 0.541683 & -0.326068 & 0.125975 \\ 0.417009 & 0.769258 & 0.330771 \\ 0.408085 & -0.249244 & -0.424397 \end{pmatrix} \text{ and } \bar{\Sigma} = \begin{pmatrix} 17.69 & 0 & 0 \\ 0 & 3.3028 & 0 \\ 0 & 0 & 1.07515 \end{pmatrix}.$$

Example 2

Then

$\overline{U}\overline{\Sigma}V^T$

$$= \begin{pmatrix} 0.111248 & 0.21065 & -0.745839 \\ 0.201801 & 0.329185 & -0.253063 \\ 0.253791 & -0.264536 & 0.263846 \\ 0.498638 & -0.130709 & 0.0683791 \\ 0.541683 & -0.326068 & 0.125975 \\ 0.417009 & 0.769258 & 0.330771 \\ 0.408085 & -0.249244 & -0.424397 \end{pmatrix} \begin{pmatrix} 17.69 & 0 & 0 \\ 0 & 3.3028 & 0 \\ 0 & 0 & 1.07515 \end{pmatrix} \begin{pmatrix} 0.761476 & -0.645229 & 0.0619242 \\ 0.60325 & 0.670482 & -0.431906 \\ 0.237159 & 0.366242 & 0.89979 \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 2 & 0 \\ 2 & 3 & 1 \\ 4 & 2 & 1 \\ 7 & 5 & 2 \\ 8 & 5 & 2 \\ 4 & 6 & 3 \\ 6 & 4 & 1 \end{pmatrix} = A$$

SVD with ‘reduced matrices’

In $\overline{U}\overline{\Sigma}V^T = A$:

- A is an $n \times p$ matrix
- $\overline{U}$ is an $n \times p$ matrix
- $\overline{\Sigma}$ is a $p \times p$ matrix (with singular values on the diagonal)
- V is a $p \times p$ matrix.

Some authors describe SVD in terms of matrices of these dimensions for $\overline{U}$ and $\overline{\Sigma}$.

Data reduction

Note that $\overline{U}\overline{\Sigma}V^T$ gives the original (data) matrix A exactly (subject to rounding).

However we can remove some of the rows/columns corresponding to smaller singular values and obtain and approximation to the original matrix A

$$\widehat{U}\widehat{\Sigma}\widehat{V}^T \approx A.$$

Data reduction

In example 2, we had singular values $\sigma_1 = 17.69$, $\sigma_2 = 3.3028$, $\sigma_3 = 1.0751$.

To use only σ_1 and σ_2 and ignore σ_3:

- delete the last column of $\bar{\Sigma}$ to give $\hat{\Sigma} = \begin{pmatrix} 17.69 & 0 \\ 0 & 3.3028 \end{pmatrix}$
- delete the last column of V to give $\hat{V} = \begin{pmatrix} 0.761476 & -0.645229 \\ 0.60325 & 0.670482 \\ 0.237159 & 0.366242 \end{pmatrix}$ so

$$\hat{V}^T = \begin{pmatrix} 0.761476 & 0.60325 & 0.237159 \\ -0.645229 & 0.670482 & 0.2366242 \end{pmatrix}.$$

Data reduction

- Also delete the last column of $\overline{U}$ to give

$$\widehat{U} = \begin{pmatrix} 0.111248 & 0.21065 \\ 0.201801 & 0.329185 \\ 0.253791 & -0.264536 \\ 0.498638 & -0.130709 \\ 0.541683 & -0.326068 \\ 0.417009 & 0.769528 \\ 0.408085 & -0.249224 \end{pmatrix}$$

Data reduction

Then

$$
\begin{aligned}
&\hat{U}\hat{\Sigma}\hat{V}^T \\
&= \begin{pmatrix} 0.111248 & 0.21065 \\ 0.201801 & 0.329185 \\ 0.253791 & -0.264536 \\ 0.498638 & -0.130709 \\ 0.541683 & -0.326068 \\ 0.417009 & 0.769528 \\ 0.408085 & -0.249224 \end{pmatrix} \begin{pmatrix} 17.69 & 0 \\ 0 & 3.3028 \end{pmatrix} \begin{pmatrix} 0.761476 & 0.60325 & 0.237159 \\ -0.645229 & 0.670482 & 0.2366242 \end{pmatrix} \\
&= \begin{pmatrix} 1.05 & 1.65 & 0.72 \\ 2.02 & 2.88 & 1.24 \\ 3.98 & 2.12 & 0.74 \\ 7.00 & 5.03 & 1.93 \\ 7.99 & 5.06 & 1.88 \\ 3.98 & 6.15 & 2.68 \\ 6.03 & 3.80 & 1.41 \end{pmatrix}.
\end{aligned}
$$

Data reduction

Thus $\hat{U}\hat{\Sigma}\hat{V}^T = \begin{pmatrix} 1.05 & 1.65 & 0.72 \\ 2.02 & 2.88 & 1.24 \\ 3.98 & 2.12 & 0.74 \\ 7.00 & 5.03 & 1.93 \\ 7.99 & 5.06 & 1.88 \\ 3.98 & 6.15 & 2.68 \\ 6.03 & 3.80 & 1.41 \end{pmatrix}$ approximates the original data matrix

$$A = \begin{pmatrix} 1 & 2 & 0 \\ 2 & 3 & 1 \\ 4 & 2 & 1 \\ 8 & 5 & 2 \\ 7 & 5 & 2 \\ 4 & 6 & 3 \\ 6 & 4 & 1 \end{pmatrix}.$$

Example 3

To illustrate with a slightly larger example, consider the 9×7 data matrix

$$A = \begin{pmatrix} 1 & 2 & 1 & 3 & 0 & 1 & 0 \\ 3 & 2 & 1 & 1 & 0 & 0 & 1 \\ 4 & 2 & 1 & 5 & 1 & 0 & 0 \\ 7 & 5 & 2 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 7 & 9 & 4 \\ 1 & 0 & 2 & 0 & 5 & 3 & 1 \\ 0 & 0 & 0 & 1 & 3 & 8 & 5 \\ 0 & 2 & 0 & 0 & 6 & 8 & 3 \\ 0 & 1 & 0 & 0 & 6 & 8 & 2 \end{pmatrix}.$$

There are seven singular values, in order,

$$21.9641, 11.4939, 4.60771, 4.03631, 2.39344, 1.70588, 1.14254.$$

Example 3

In the SVD equation $A = U\Sigma V^T$:

- U is a 9×9 orthogonal matrix
- Σ is a 9×7 matrix with singular values as diagonal entries and two rows of zeros
- V is a 7×7 orthogonal matrix.

Suppose we choose to use just the first three of the seven singular values:

$$21.9641, 11.4939, 4.60771.$$

In the resulting approximation $A \approx \hat{U}\hat{\Sigma}\hat{V}^T$, $\hat{U}$ is 9×3, $\hat{\Sigma}$ is 3×3 and $\hat{V}$ is 7×3.

Example 3

In fact

$$\hat{U}\hat{\Sigma}\hat{V}^T = \begin{pmatrix} 1.81 & 1.41 & 0.46 & 2.66 & -0.05 & 0.87 & 0.79 \\ 2.6 & 1.98 & 0.87 & 1.37 & 0.37 & 0.21 & 0.12 \\ 3.86 & 2.92 & 1.1 & 4.09 & -0.38 & 0.59 & 0.78 \\ 6.5 & 4.89 & 2.31 & 1.75 & 1.07 & -0.56 & -0.55 \\ -0.04 & 1 & 0.25 & -0.04 & 6.79 & 9.24 & 3.79 \\ 1.08 & 1.28 & 0.61 & -0.89 & 3.67 & 3.78 & 1.22 \\ -0.71 & 0.25 & -0.25 & 1.98 & 4.23 & 7.62 & 3.68 \\ 0.45 & 1.23 & 0.4 & -0.02 & 6.03 & 7.96 & 3.22 \\ 0.11 & 0.94 & 0.29 & -0.31 & 5.85 & 7.67 & 3.06 \end{pmatrix}.$$

This is reasonable approximation to the original data matrix A given the much smaller matrices used.

Example 3

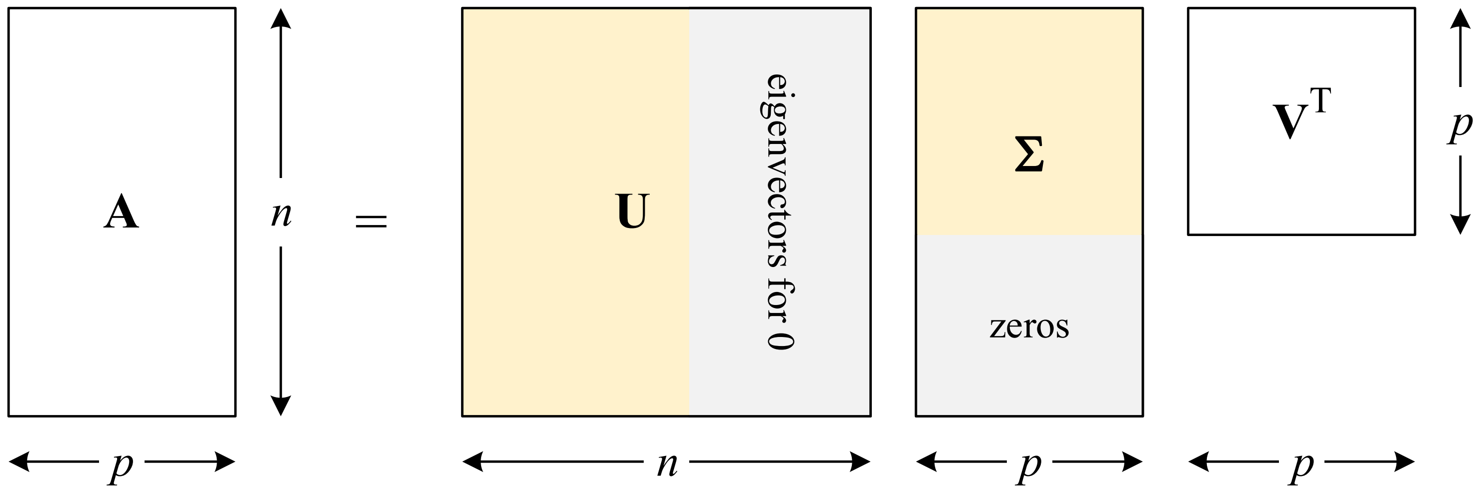
This is reasonable approximation to the original data matrix A.

$$\begin{pmatrix} 1.81 & 1.41 & 0.46 & 2.66 & -0.05 & 0.87 & 0.79 \\ 2.6 & 1.98 & 0.87 & 1.37 & 0.37 & 0.21 & 0.12 \\ 3.86 & 2.92 & 1.1 & 4.09 & -0.38 & 0.59 & 0.78 \\ 6.5 & 4.89 & 2.31 & 1.75 & 1.07 & -0.56 & -0.55 \\ -0.04 & 1 & 0.25 & -0.04 & 6.79 & 9.24 & 3.79 \\ 1.08 & 1.28 & 0.61 & -0.89 & 3.67 & 3.78 & 1.22 \\ -0.71 & 0.25 & -0.25 & 1.98 & 4.23 & 7.62 & 3.68 \\ 0.45 & 1.23 & 0.4 & -0.02 & 6.03 & 7.96 & 3.22 \\ 0.11 & 0.94 & 0.29 & -0.31 & 5.85 & 7.67 & 3.06 \end{pmatrix} \approx \begin{pmatrix} 1 & 2 & 1 & 3 & 0 & 1 & 0 \\ 3 & 2 & 1 & 1 & 0 & 0 & 1 \\ 4 & 2 & 1 & 5 & 1 & 0 & 0 \\ 7 & 5 & 2 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 7 & 9 & 4 \\ 1 & 0 & 2 & 0 & 5 & 3 & 1 \\ 0 & 0 & 0 & 1 & 3 & 8 & 5 \\ 0 & 2 & 0 & 0 & 6 & 8 & 3 \\ 0 & 1 & 0 & 0 & 6 & 8 & 2 \end{pmatrix}$$

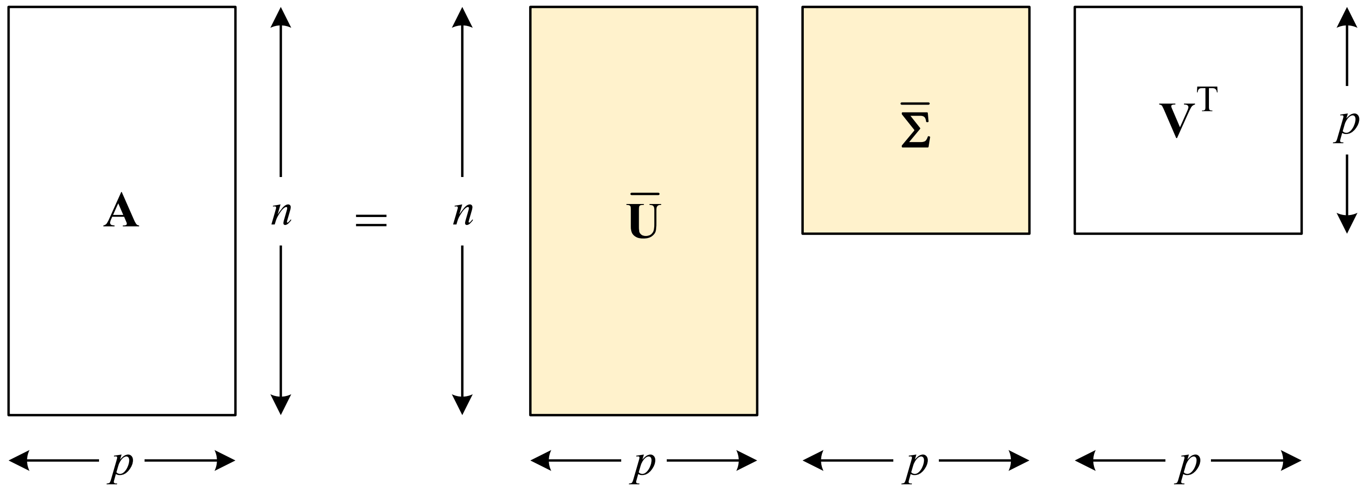
In fact, given the ‘size’ of matrices used (formally the matrix ‘rank’), this is the best possible approximation. In cases where the ‘omitted’ singular values are very small, the approximation would be even better.

Data reduction

We can picture the data reduction as follows.

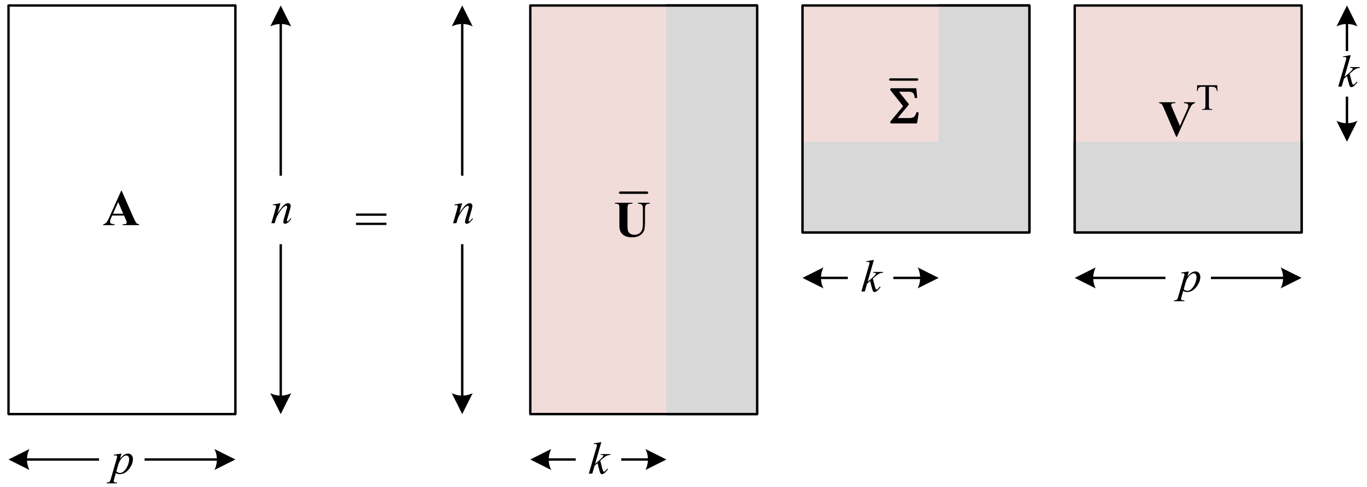


Removing the zero rows of Σ and the zero eigenvectors of U:

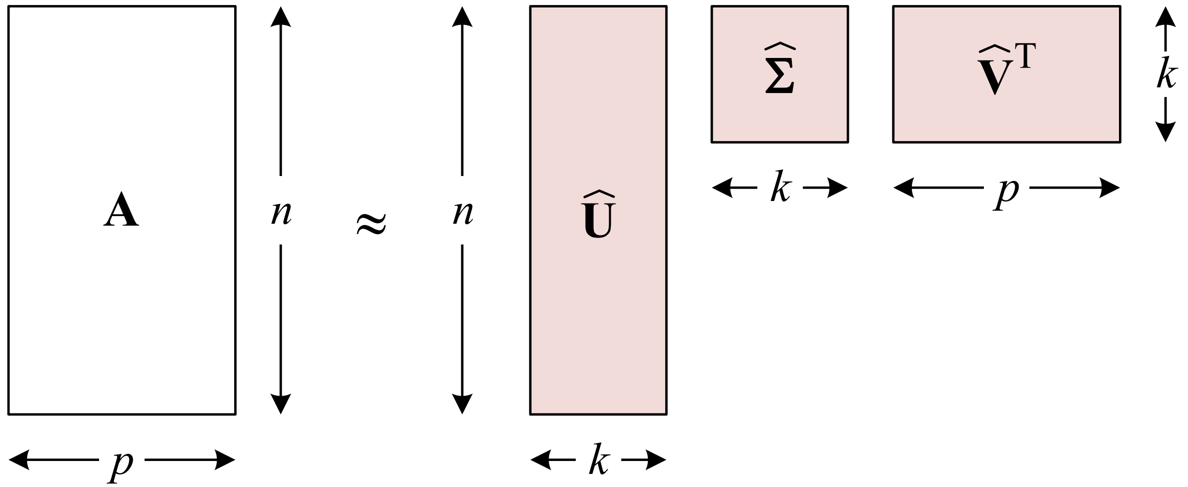


Data reduction

Selecting only k of the singular values.



Approximate A.

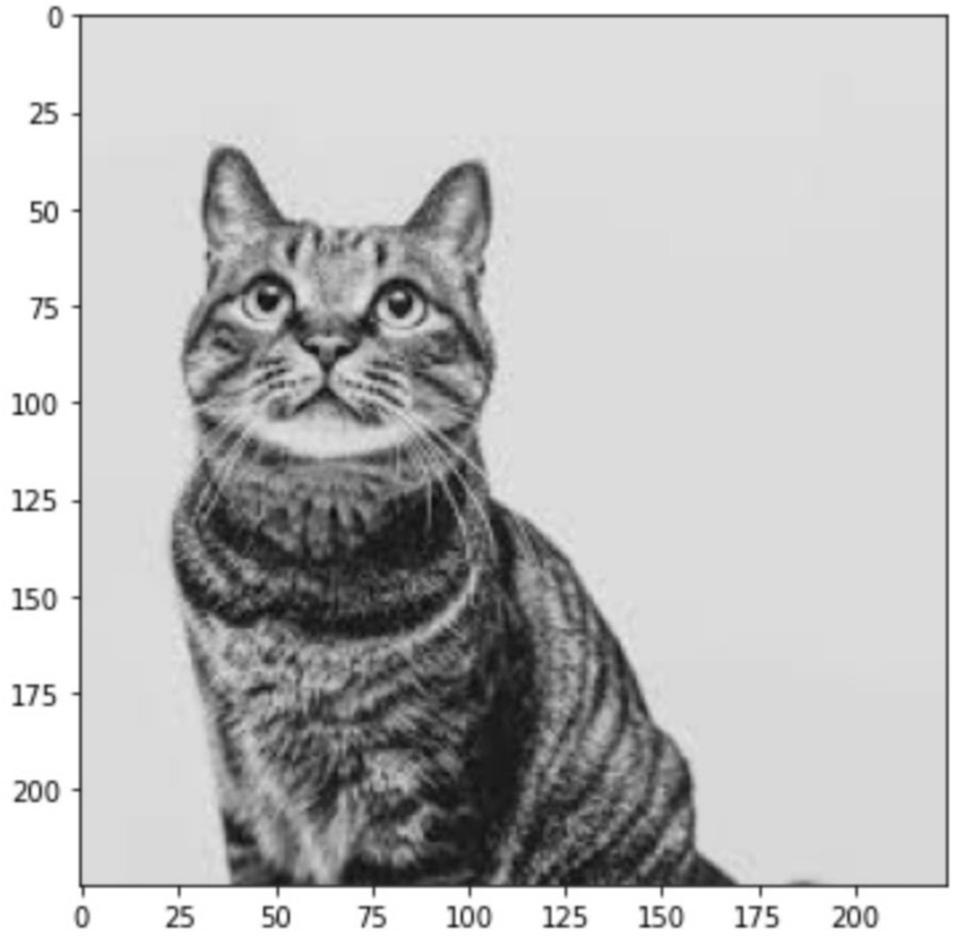


$\mathbf{A}$
n
$\approx$
n
$\widehat{\mathbf{U}}$
$\widehat{\boldsymbol{\Sigma}}$
$\widehat{\mathbf{V}}^{\mathrm{T}}$
k
k
p
p
k

Application - Image compression using SVD

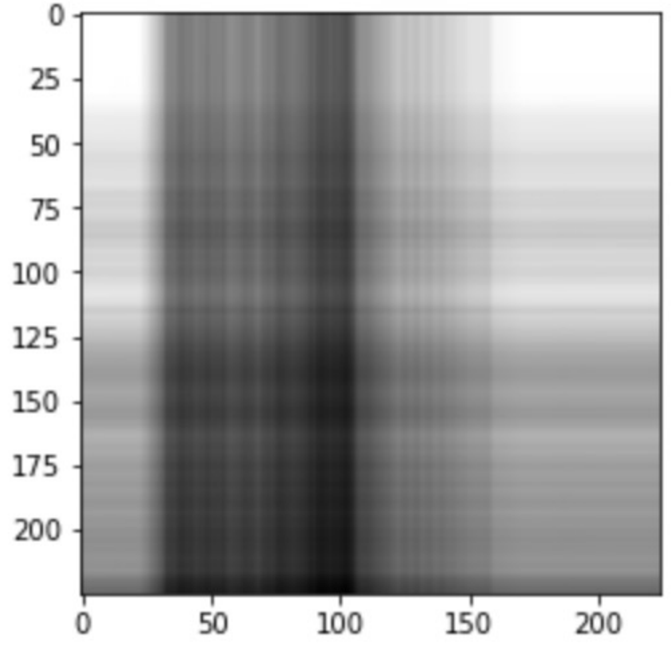
```
import matplotlib.pyplot as plt
import numpy as np
from PIL import Image

img = Image.open('cat.jpg’)
imgmat = np.matrix(imgmat)
```

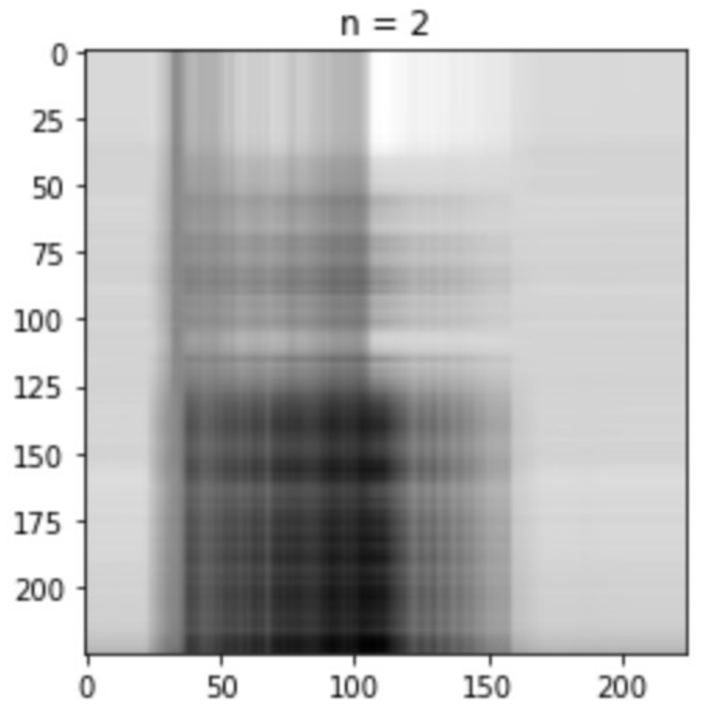


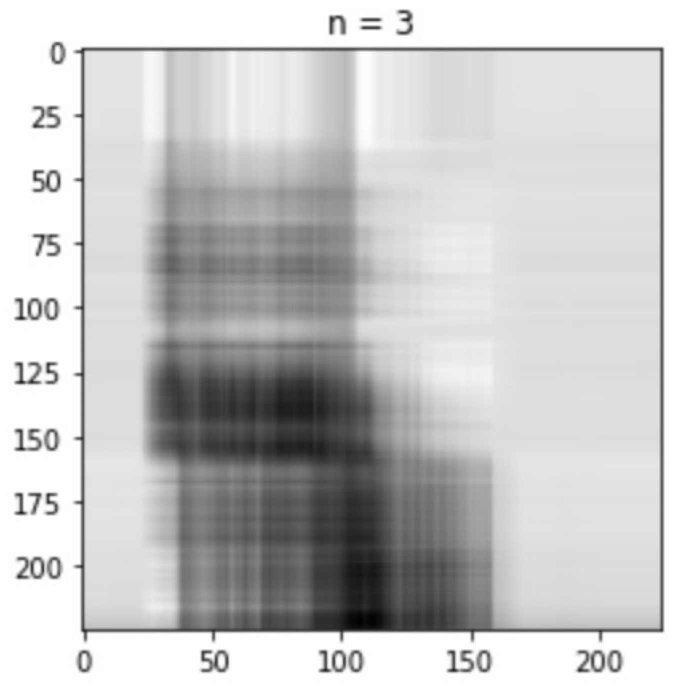
```
U, sigma, V = np.linalg.svd(imgmat)

reconstimg = np.matrix(U[:, :1]) * np.diag(sigma[:1]) * np.matrix(V[:1, :])
plt.imshow(reconstimg, cmap='gray');
```

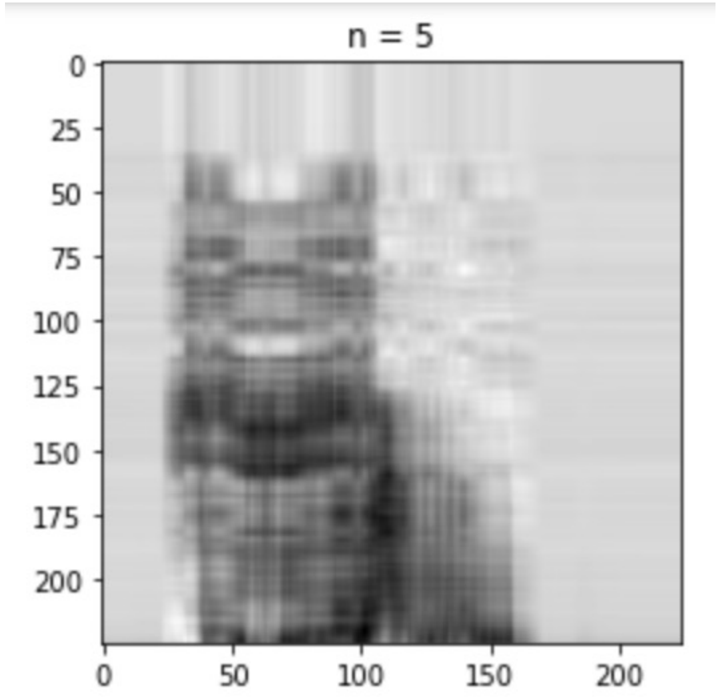


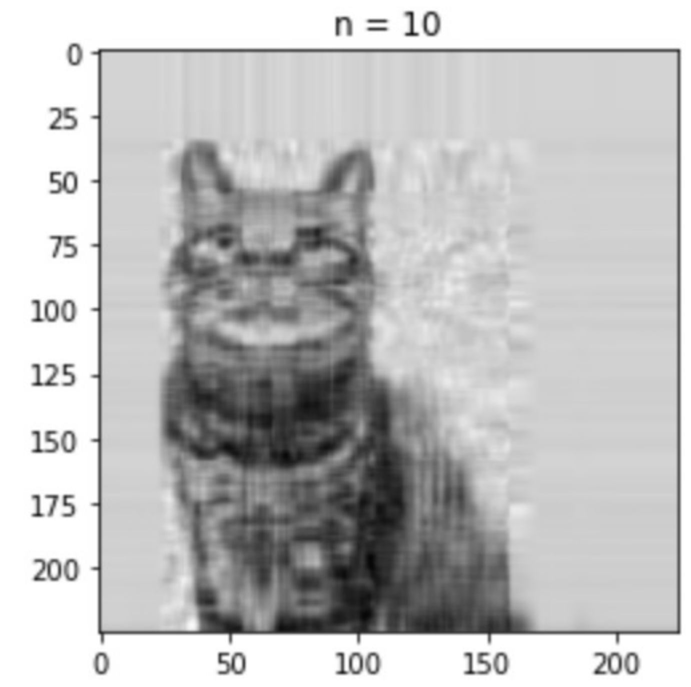
```
for i in range(2, 4):
    reconstimg = np.matrix(U[:, :i]) * np.diag(sigma[:i]) * np.matrix(V[:i, :])
    plt.imshow(reconstimg, cmap='gray')
    title = "n = %s" % i
    plt.title(title)
    plt.show()
```

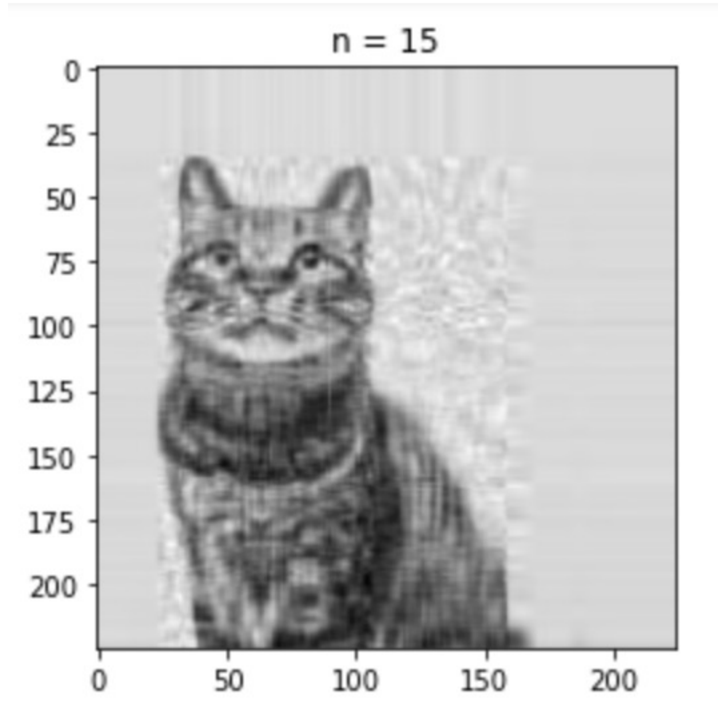


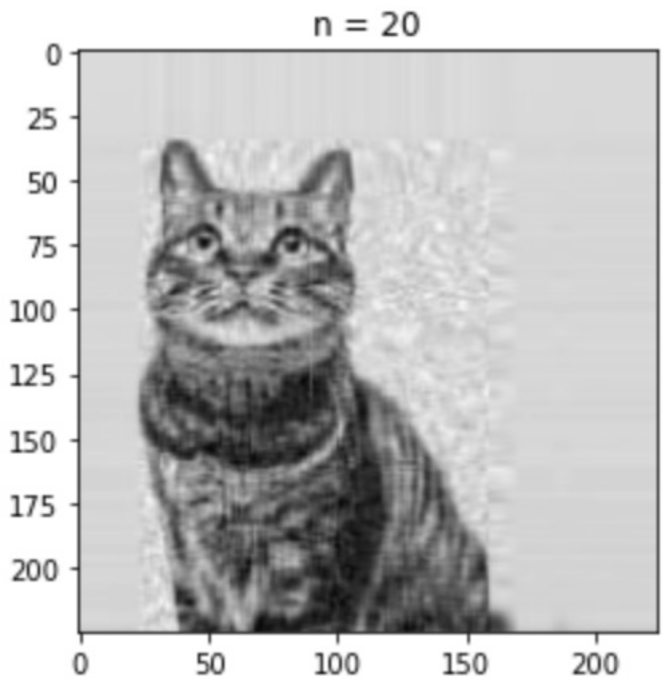


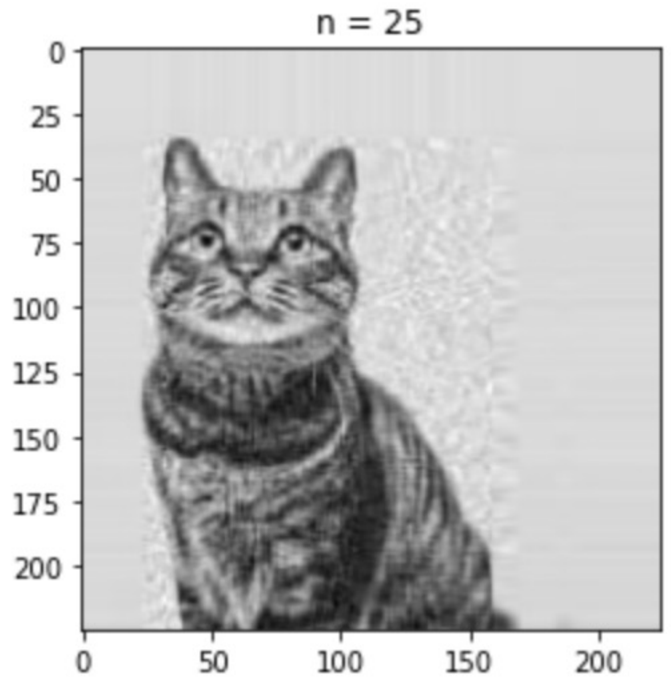
```
for i in range(5, 46, 5):
    reconstimg = np.matrix(U[:, :i]) * np.diag(sigma[:i]) * np.matrix(V[:i, :])
    plt.imshow(reconstimg, cmap='gray')
    title = "n = %s" % i
    plt.title(title)
    plt.show()
```

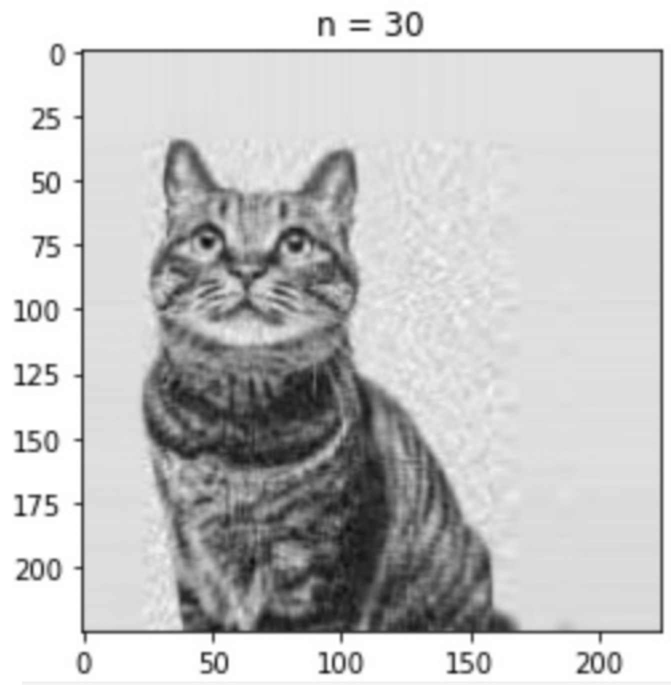


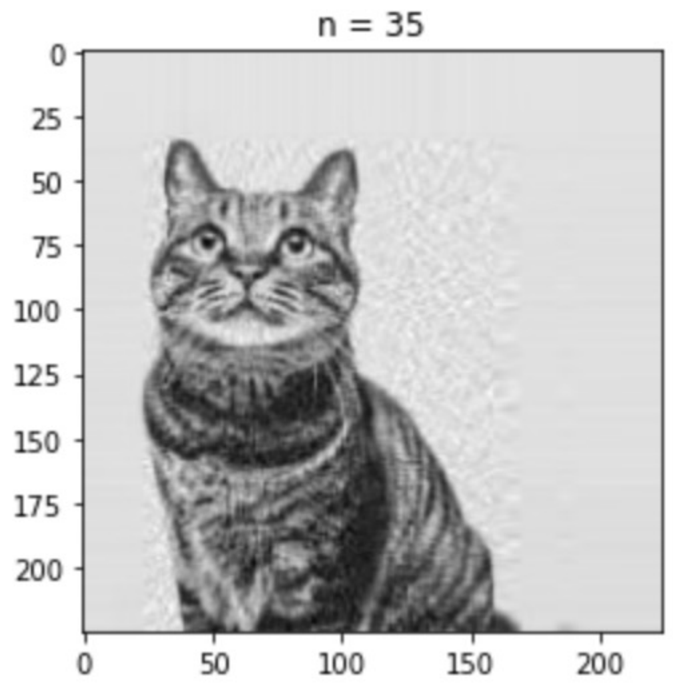


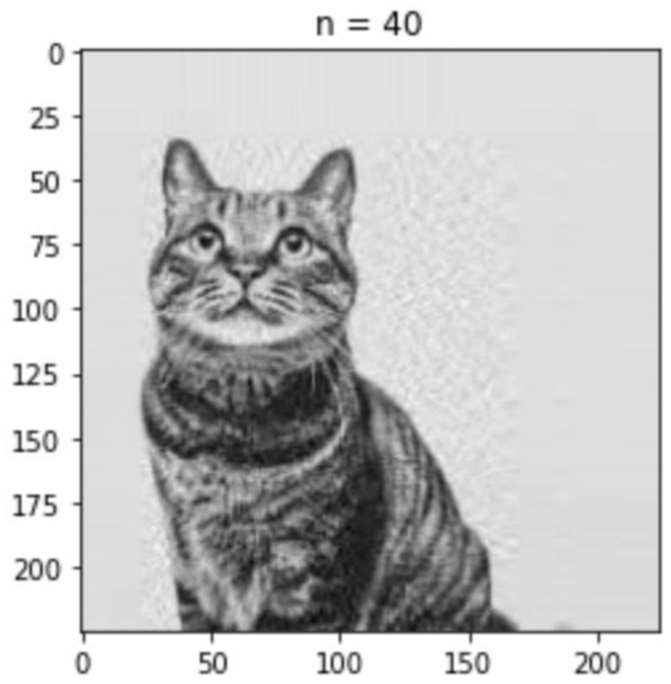


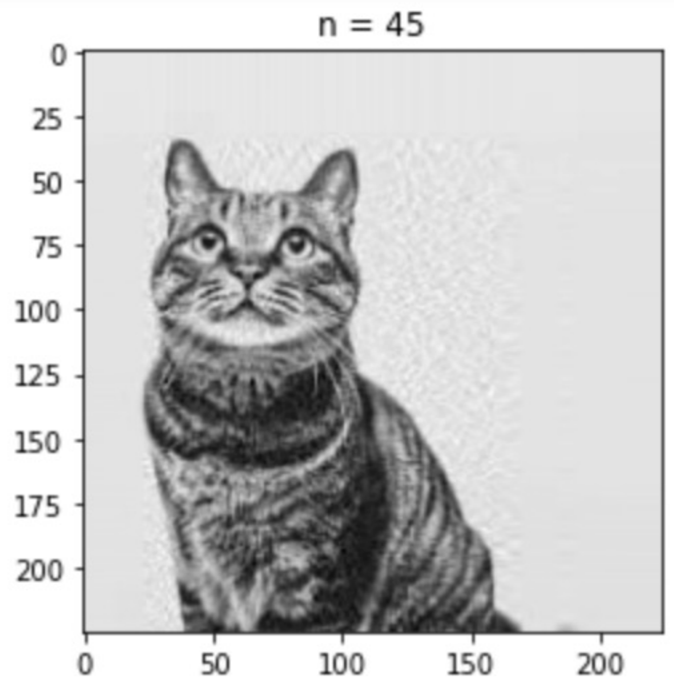

n = 20


n = 25


n = 30


n = 35


n = 40


n = 45

Application: recommender systems

In a recommender systems application, let A represent users to movies ratings.

	Movie 1	Movie 2	Movie 3	…	Movie p
User 1	$R_{1,1}$	$R_{1,2}$	$R_{1,3}$	…	$R_{1,p}$
User 2	$R_{2,1}$	$R_{2,2}$	$R_{2,3}$	…	$R_{2,p}$
User 3	$R_{3,1}$	$R_{3,2}$	$R_{3,3}$	…	$R_{3,p}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$		$\vdots$
User n	$R_{n,1}$	$R_{n,2}$	$R_{n,3}$	…	$R_{n,p}$

Here $R_{i,j}$ is the rating that user i gives to movie j.

Application: recommender systems

We can think of movies as having genres or concepts or styles that users implicitly rate when rating the movies.

Then in the SVD representation $A = U\Sigma V^T$:

- U represents user to concept similarities
- Σ represents the strength of each concept
- V represents movie to concept similarities (so that V^T represents concept to user similarities).

The approximation from selecting only some singular values corresponds to choosing only the ‘most significant’ concepts.