GENERAL ASSEMBLY

INTRO TO DATA SCIENCE

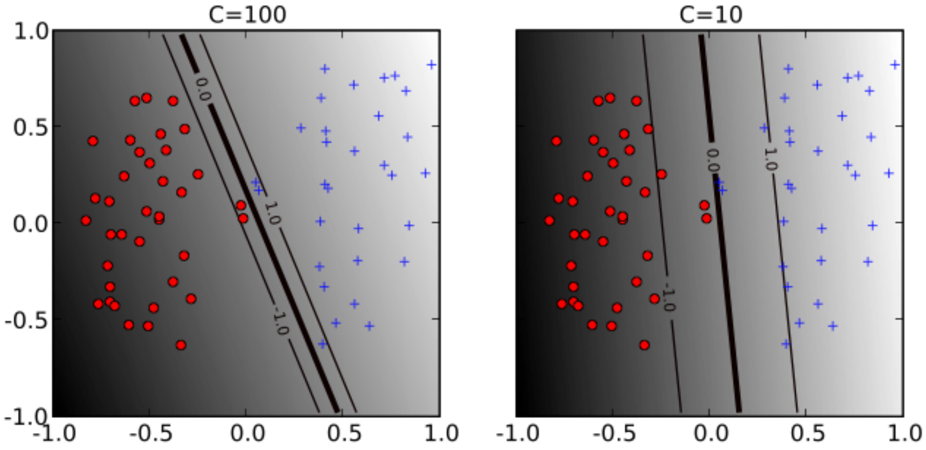
DIMENSIONALITY REDUCTION

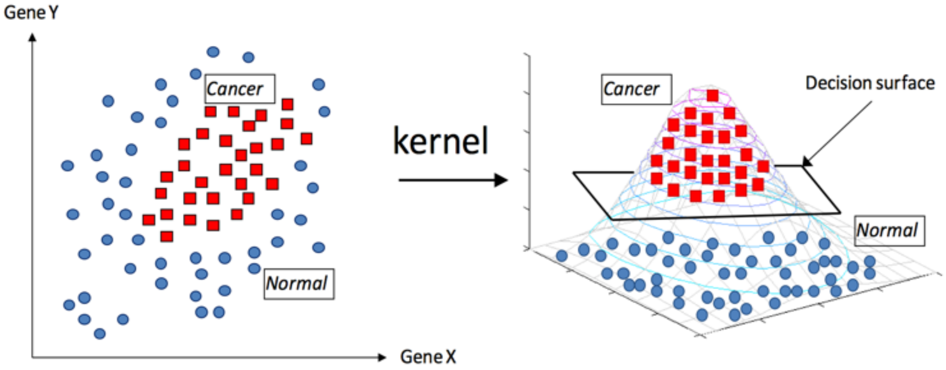
I. SUPPORT VECTOR MACHINES (SVM)

II. SLACK VARIABLES

III. NONLINEAR CLASSIFICATION WITH KERNELS

IV. LAB ON SVM'S IN PYTHON





QUESTIONS?

WHAT WAS THE MOST INTERESTING THING YOU LEARNT?

WHAT WAS THE HARDEST TO GRASP?

I. DIMENSIONALITY REDUCTION
II. PRINCIPAL COMPONENTS ANALYSIS
III. SINGULAR VALUE DECOMPOSITION

EXERCISE:
IV. DIMENSIONALITY REDUCTION IN SCIKIT-LEARN

DIMENSIONALITY REDUCTION

Q: What is dimensionality reduction?

Q: What is dimensionality reduction?
A: A set of techniques for **reducing the size** (in terms of features, records, and/or bytes) of the dataset under examination.

GENERAL IDEA: dataset as a **matrix**
=> decompose the matrix into **simpler, meaningful pieces.**

Dimensionality reduction is frequently performed as a pre-processing step before another learning algorithm is applied.

	Continuous	Categorical
Supervised	???	???
Unsupervised	???	???

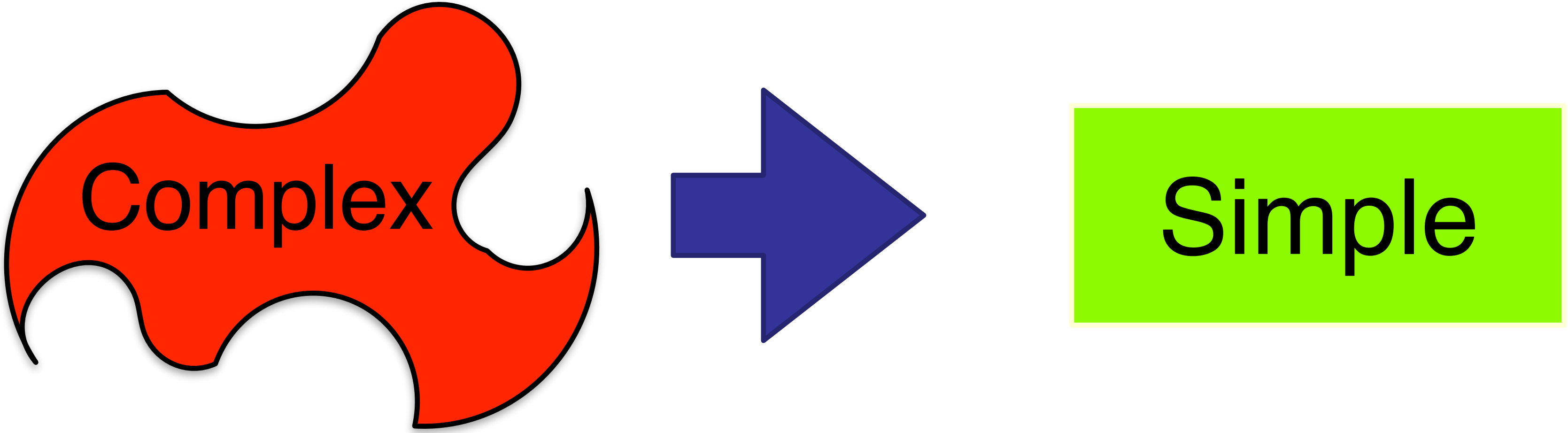
	Continuous	Categorical
Supervised	regression	classification
Unsupervised	dimension reduction	clustering

Q: Reasons to apply dimensionality reduction?

Q: Reasons to apply dimensionality reduction?

- too many features to manage in our dataset
- useless or misleading features (e.g., if the relationships are actually simpler than they appear)
- want to project data on 2D plane

data representation　　　　　　　　data representation



retain as much of the signal in our data as possible

Look at our data "from another angle"…

Q: What is the goal of dimensionality reduction?

- reduce computational expense
- reduce susceptibility to overfitting
- reduce noise in the dataset
- enhance our intuition

Q: How is dimensionality reduction performed?

Q: How is dimensionality reduction performed?
A: There are two approaches: feature selection and feature extraction.

Q: How is dimensionality reduction performed?
A: There are two approaches: feature selection and feature extraction.

feature selection – selecting a subset of features using an external criterion or the learning algo accuracy itself

feature extraction – mapping the features to a lower dimensional space

Feature selection is important, but typically when people say dimensionality reduction, they are referring to *feature extraction*.

Feature selection is important, but typically when people say dimensionality reduction, they are referring to *feature extraction*.

The goal of feature extraction is to create a new set of coordinates that *simplify the representation* of the data.

Example: features that are related to each other

Ideally, we would like to eliminate this redundancy and consolidate the number of variables we’re looking at.

If these relationships are *linear*, then we can use well-established techniques like PCA/SVD.

Example: features that are related to each other

- house dataset
- titanic dataset
- ?

HOW MIGHT WE SIMPLIFY THIS HARMONIC OSCILLATOR?

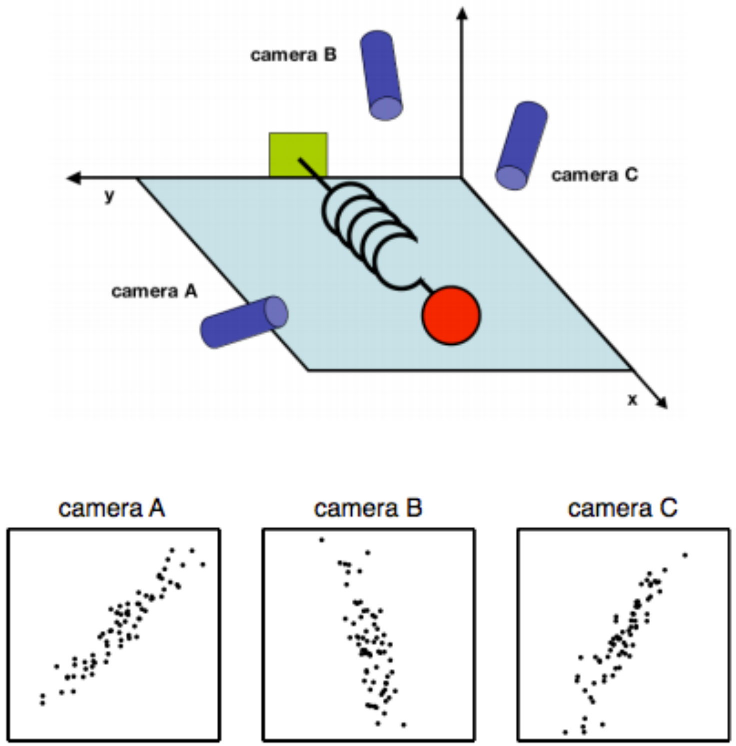


FIG. 1 A toy example. The position of a ball attached to an oscillating spring is recorded using three cameras A, B and C. The position of the ball tracked by each camera is depicted in each panel below.

source: http://www.snl.salk.edu/~shlens/pca.pdf

Large number of features => curse of dimensionality

Namely, the sample size needed to accurately estimate a random variable taking values in a d-dimensional feature space grows exponentially with d (almost).

(More precisely, the sample size grows exponentially with $l \leq d$, the dimension of the manifold *embedded* in the feature space).

Another way of characterizing this is to say that high-dimensional spaces are inherently sparse.

Another way of characterizing this is to say that high-dimensional spaces are inherently sparse.

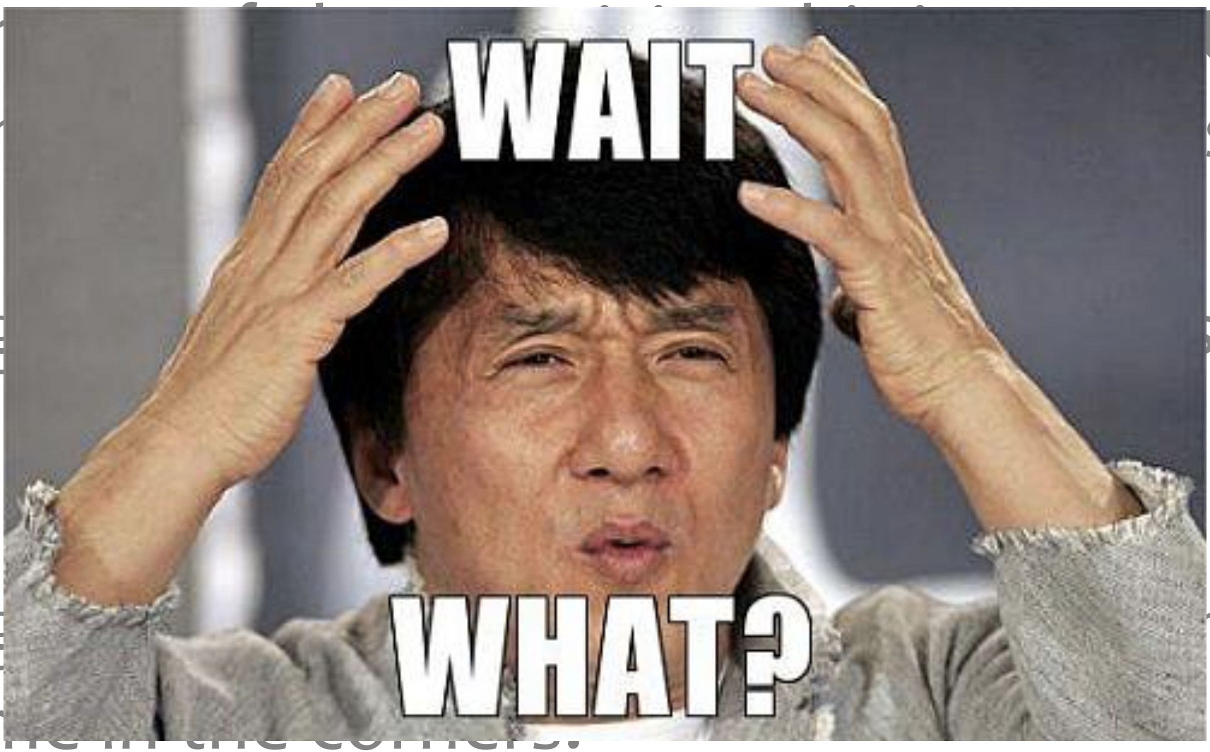
ex: A high-dimensional orange contains most of its volume in the rind!

ex: A high-dimensional hypercube contains most of its volume in the corners!

Anothe … that
high-di … se.

ex: A hi … st of its
volume

ex: A hi … most of
its volu



explained here: http://scipp.ucsc.edu/~haber/ph116A/volume_11.pdf

In either case, most of the points in the space are “far” from the center.

In either case, most of the points in the space are “far” from the center.

This illustrates the fact that local methods will break down in these circumstances (eg, in order to collect enough neighbors for a given point, you need to expand the radius of the neighborhood so far that locality is not preserved).

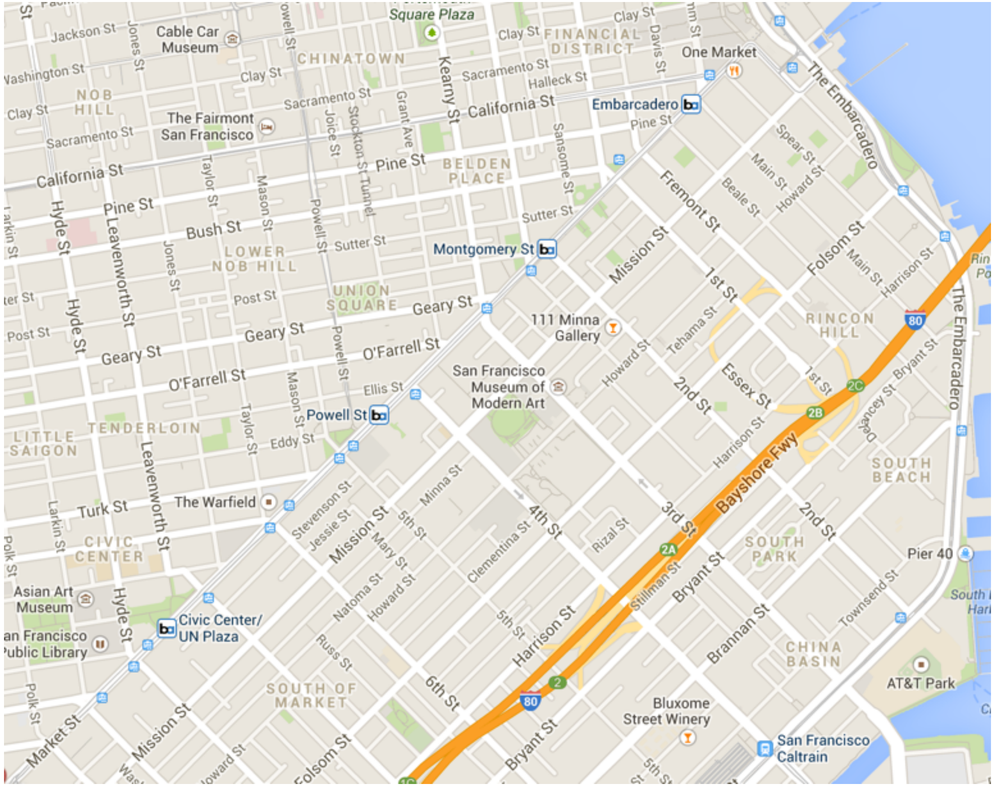
In either case, most of the points in the space are “far” from the center.

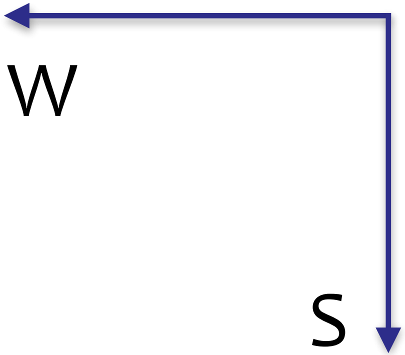
This illustrates the fact that local methods will break down in these circumstances (eg, in order to collect enough neighbors for a given point, you need to expand the radius of the neighborhood so far that locality is not preserved).

The bottom line is that high-dimensional spaces can be problematic.

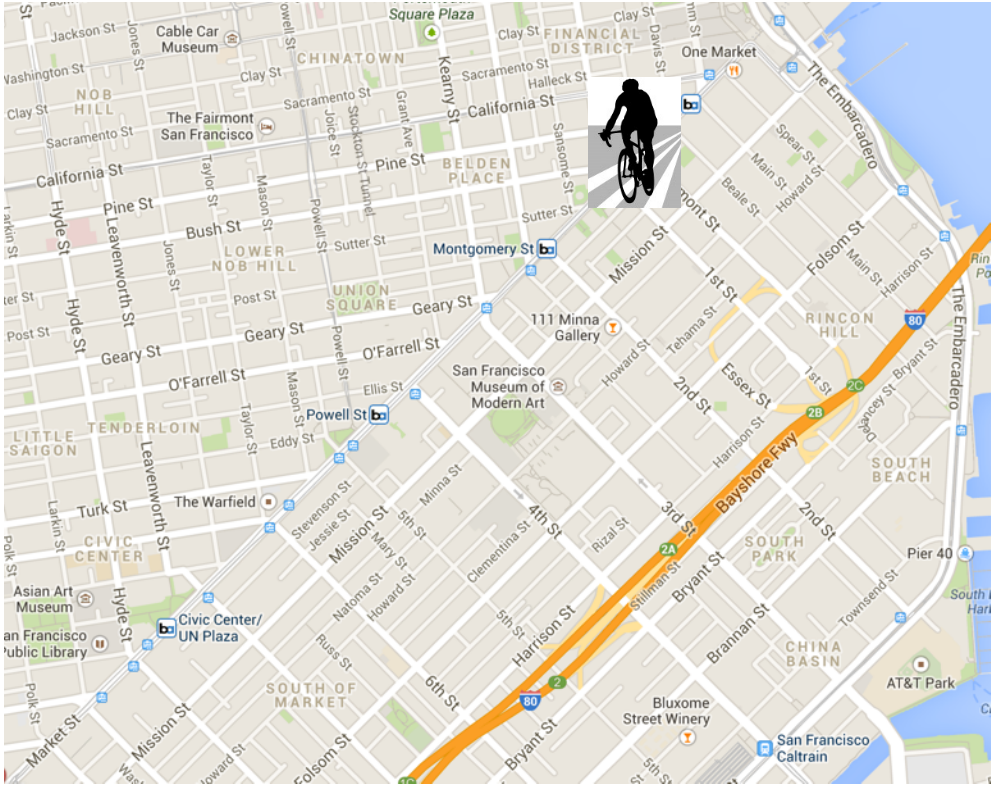
Okay, so how about a more intuitive example of dimensionality reduction than a harmonic oscillator? That would be great, mmmmkayyy?

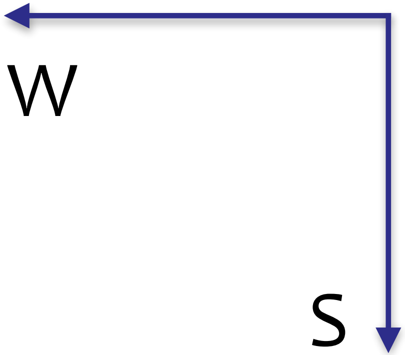
INTUITIVE EXAMPLE - BIKING DOWN MARKET STREET



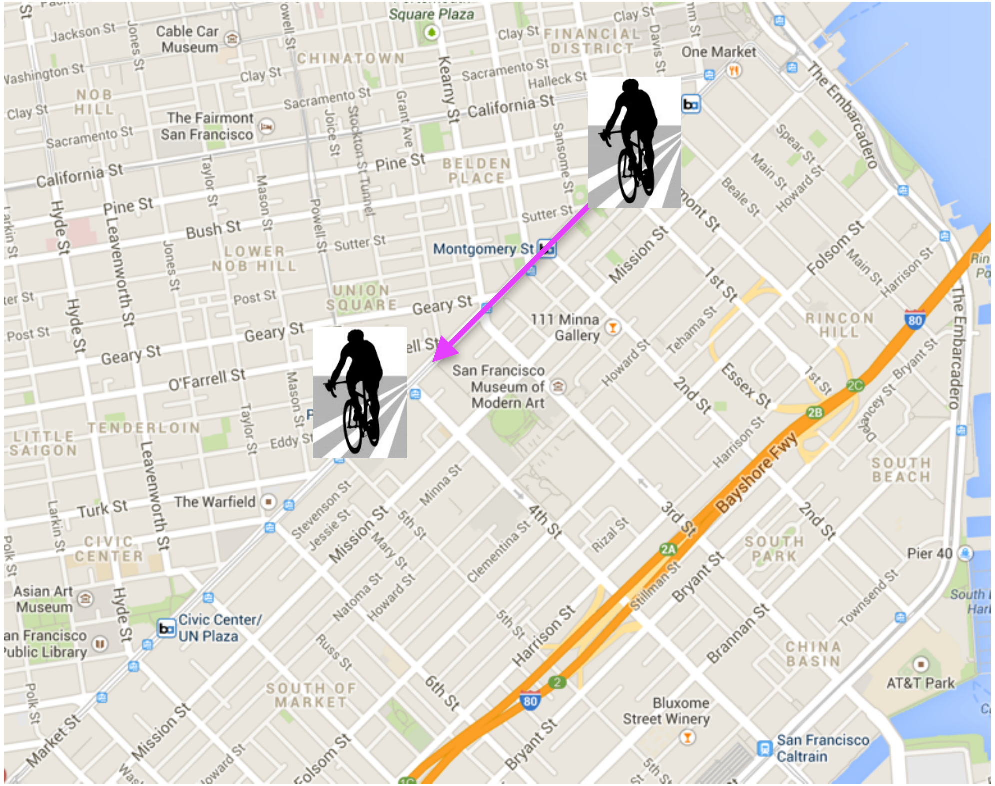


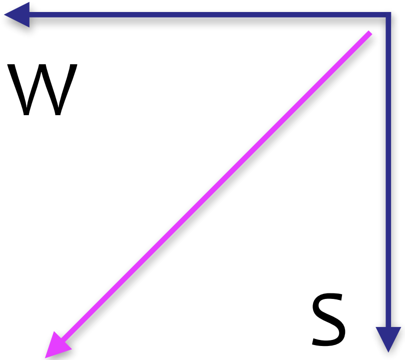
INTUITIVE EXAMPLE - BIKING DOWN MARKET STREET



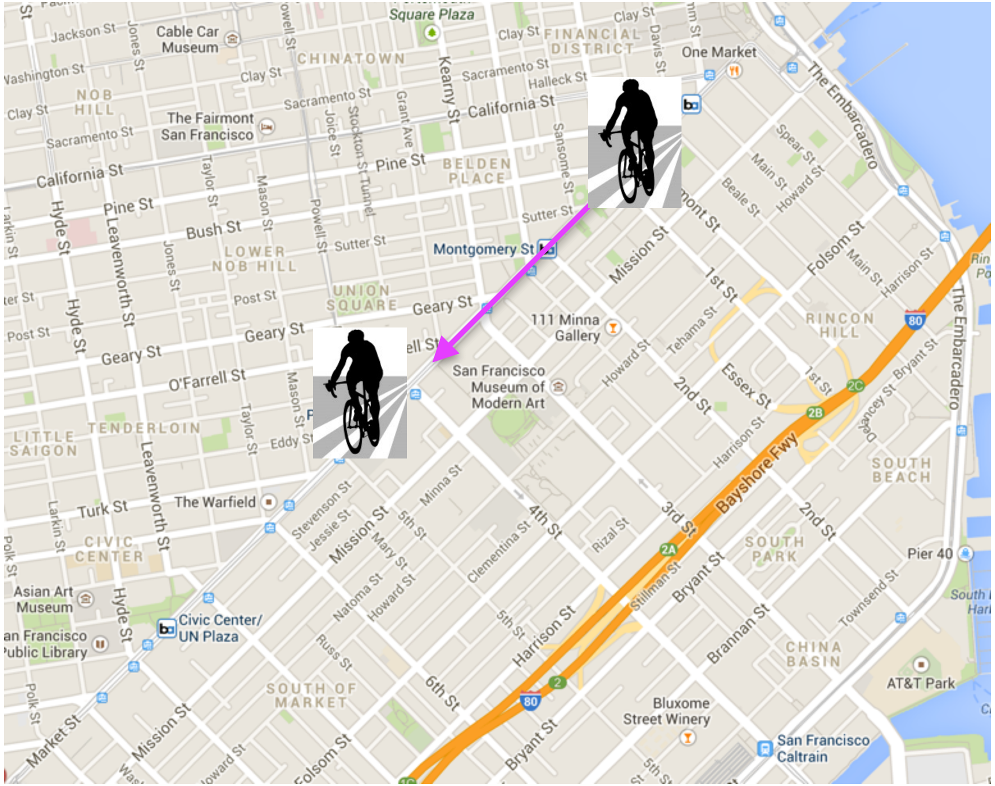


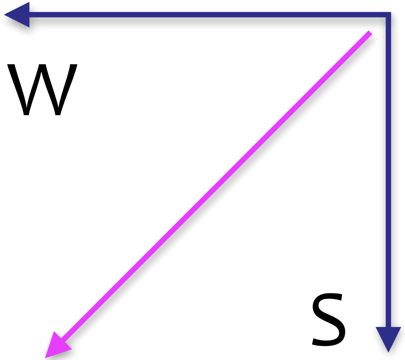
INTUITIVE EXAMPLE - BIKING DOWN MARKET STREET





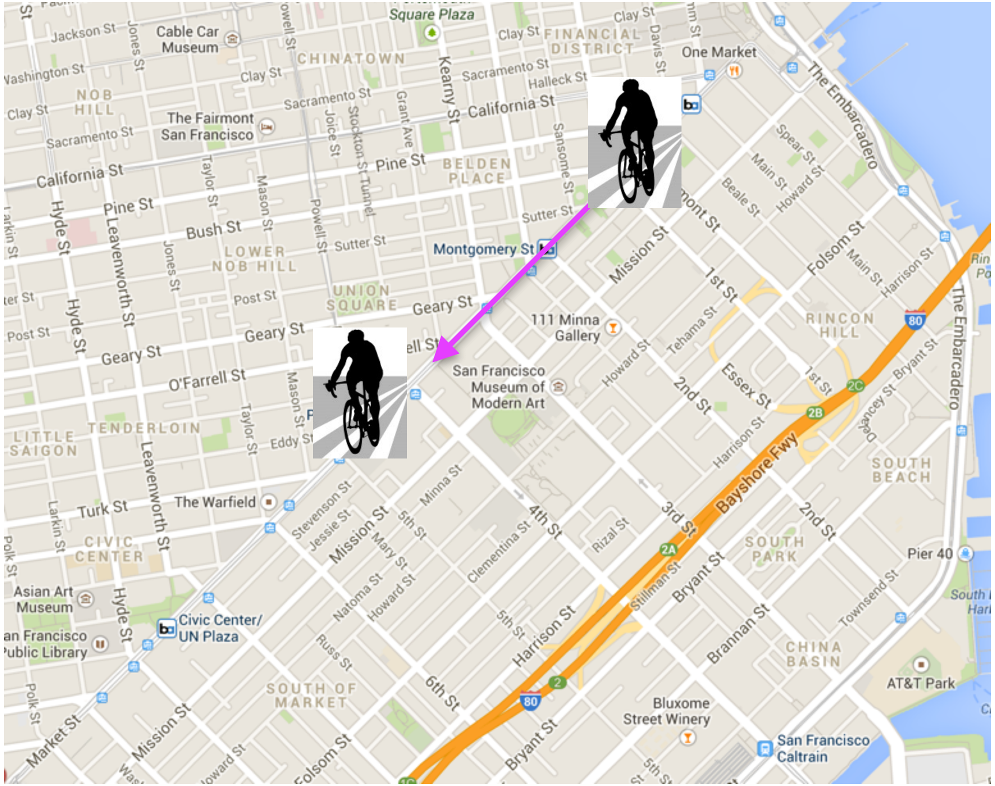
INTUITIVE EXAMPLE - BIKING DOWN MARKET STREET

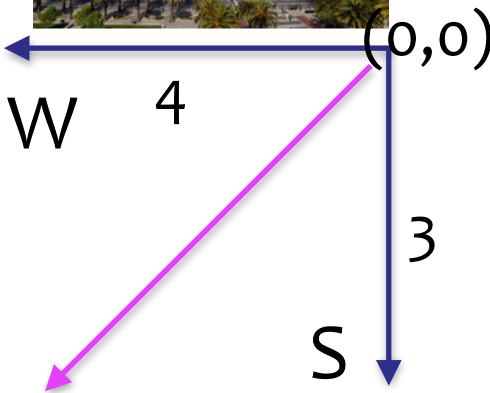




How many dimensions do we need to specify the position of this bike?

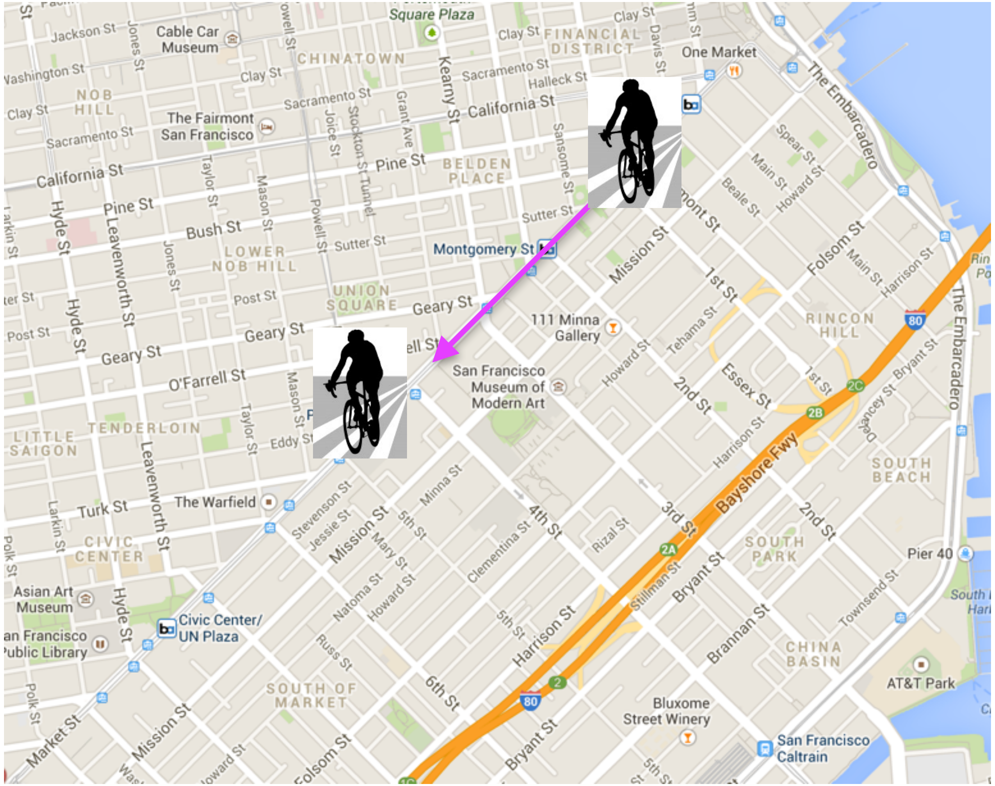
INTUITIVE EXAMPLE - BIKING DOWN MARKET STREET

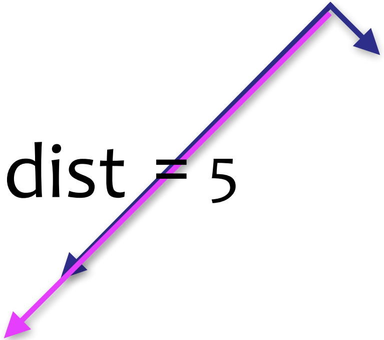




Yep, two. But could we represent the biker’s position with fewer dimensions? How?

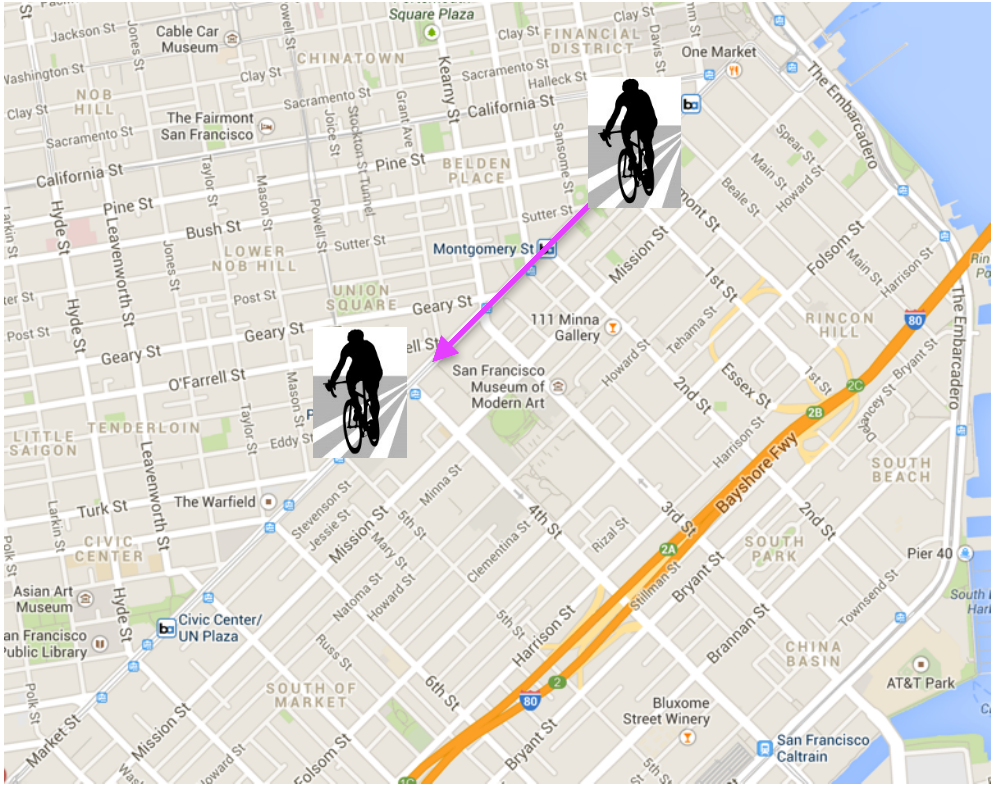
INTUITIVE EXAMPLE - BIKING DOWN MARKET STREET

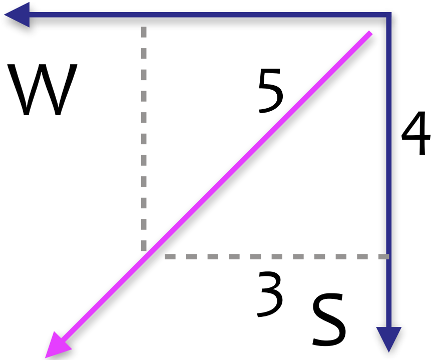




What if we just used distance down Market St.?

INTUITIVE EXAMPLE - BIKING DOWN MARKET STREET





Of course, we can always map back to the original coordinate system!

EXAMPLE: 1D HARMONIC OSCILLATOR

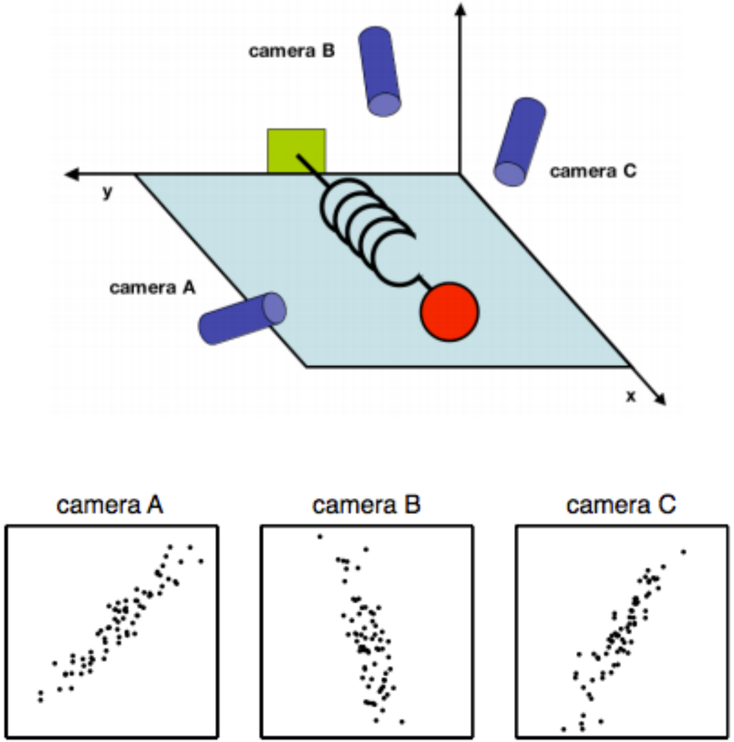
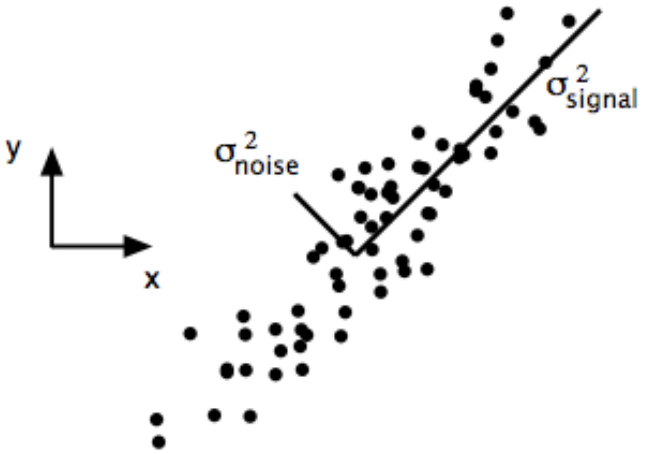


FIG. 1 A toy example. The position of a ball attached to an oscillating spring is recorded using three cameras A, B and C. The position of the ball tracked by each camera is depicted in each panel below.

source: http://www.snl.salk.edu/~shlens/pca.pdf



$$SNR = \frac{\sigma^2_{signal}}{\sigma^2_{noise}}.$$

FIG. 2 Simulated data of (x, y) for camera A. The signal and noise variances σ^2_{signal} and σ^2_{noise} are graphically represented by the two lines subtending the cloud of data. Note that the largest direction of variance does not lie along the basis of the recording (x_A, y_A) but rather along the best-fit line.

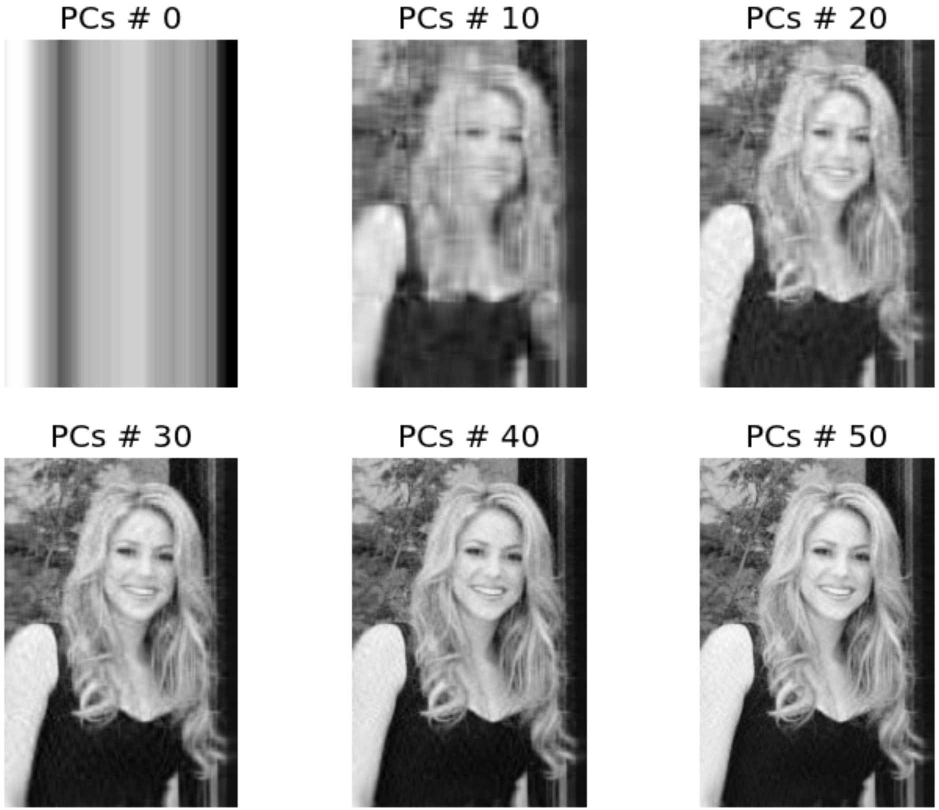
source: http://www.snl.salk.edu/~shlens/pca.pdf

Q: What are some applications of dimensionality reduction?

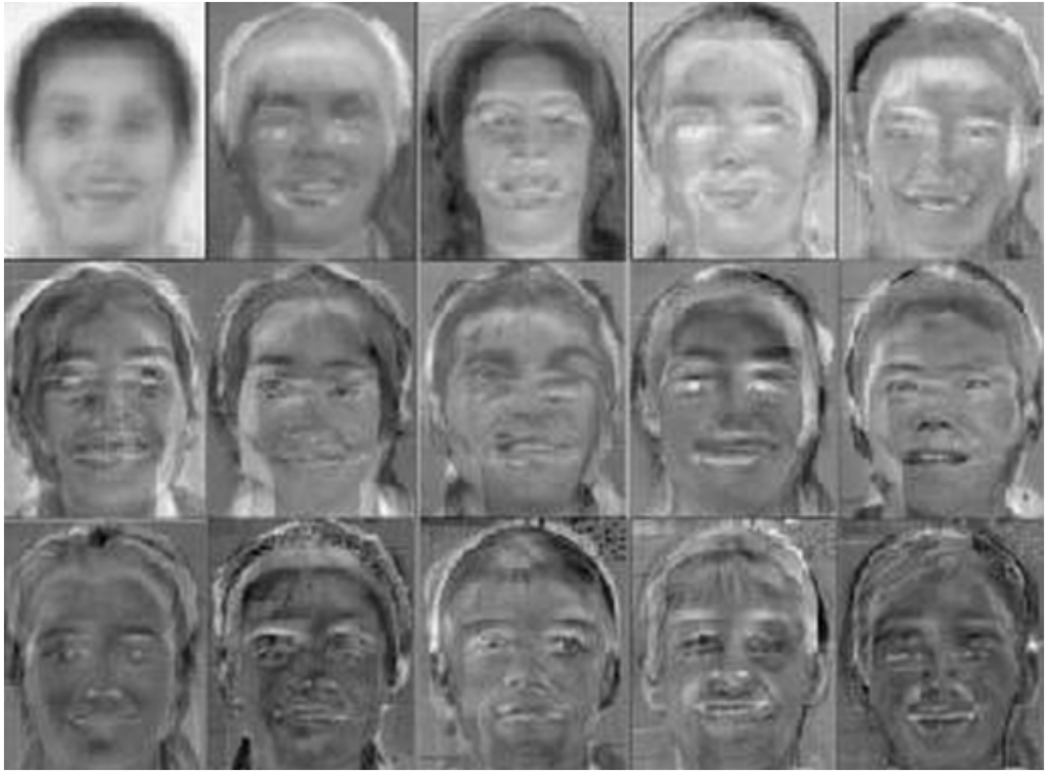
Q: What are some applications of dimensionality reduction?

- topic models (document clustering)
- image recognition/computer vision
- bioinformatics (microarray analysis)
- speech recognition
- astronomy (spectral data analysis)
- recommender systems

DIMENSIONALITY REDUCTION



source: http://glowingpython.blogspot.it/2011/07/pca-and-image-compression-with-numpy.html



Your turn:

Why use Dimensionality Reduction?

PRINCIPAL COMPONENT ANALYSIS (PCA)

Principal component analysis is a dimension reduction technique that can be used on a matrix of any dimensions.

resources on eigenvectors:
https://en.wikipedia.org/wiki/Eigendecomposition_of_a_matrix
http://setosa.io/ev/eigenvectors-and-eigenvalues/

Principal component analysis is a dimension reduction technique that can be used on a matrix of any dimensions.

This procedure produces a **new basis** (a new coordinate system), each of whose components retain as much of the variance in the original data as possible.

resources on eigenvectors:
https://en.wikipedia.org/wiki/Eigendecomposition_of_a_matrix
http://setosa.io/ev/eigenvectors-and-eigenvalues/

Principal component analysis is a dimension reduction technique that can be used on a matrix of any dimensions.

This procedure produces a **new basis** (a new coordinate system), each of whose components retain as much variance from the original data as possible.

The PCA of a matrix *A* boils down to the **eigenvalue decomposition** of the **covariance matrix of *A*.**

resources on eigenvectors:
https://en.wikipedia.org/wiki/Eigendecomposition_of_a_matrix
http://setosa.io/ev/eigenvectors-and-eigenvalues/

what is variance?

$$s^2 = \frac{\sum_{i=1}^{n}(X_i - \bar{X})^2}{(n-1)}$$

We can think of variance as the average distance from a point in a data set to the mean of that data set.

In other words, it is a measure of the *spread* of the data. Recall that standard deviation is the square root of variance.

what is covariance?

Standard deviation and **variance** only operate on 1 dimension, so that you could only calculate the standard deviation for each dimension of the data set *independently* of the other dimensions.

That said, it is useful to have a similar measure to find out how much the dimensions vary from the mean ***with respect to each other***.

This variance with respect to each other is called **covariance**.

covariance is a measure of how much two random variables change together

Variance:

$$s^2 = \frac{\sum_{i=1}^{n}(X_i - \bar{X})^2}{(n-1)} \qquad var(X) = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(X_i - \bar{X})}{(n-1)}$$

Covariance:

$$cov(X,Y) = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{(n-1)}$$

covariance is a measure of how much two random variables change together

Variance:

$$s^2 = \frac{\sum_{i=1}^{n}(X_i - \bar{X})^2}{(n-1)} \qquad var(X) = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(X_i - \bar{X})}{(n-1)}$$

Covariance:

$$\boxed{cov(X,Y) = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{(n-1)}}$$

The covariance matrix C of a matrix A is always square:

$$\mathrm{C} = \begin{bmatrix} \mathrm{E}[(X_1-\mu_1)(X_1-\mu_1)] & \mathrm{E}[(X_1-\mu_1)(X_2-\mu_2)] & \cdots & \mathrm{E}[(X_1-\mu_1)(X_n-\mu_n)] \\ \mathrm{E}[(X_2-\mu_2)(X_1-\mu_1)] & \mathrm{E}[(X_2-\mu_2)(X_2-\mu_2)] & \cdots & \mathrm{E}[(X_2-\mu_2)(X_n-\mu_n)] \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{E}[(X_n-\mu_n)(X_1-\mu_1)] & \mathrm{E}[(X_n-\mu_n)(X_2-\mu_2)] & \cdots & \mathrm{E}[(X_n-\mu_n)(X_n-\mu_n)] \end{bmatrix}.$$

off-diagonal elements C_{ij} give the *covariance* between X_i, X_j $(i \neq j)$

diagonal elements C_{ii} give the *variance* of X_i

The *eigenvalue decomposition* of a square matrix A is given by:

$$A = Q \Lambda Q^{-1}$$

The *eigenvalue decomposition* of a square matrix A is given by:

$$A = Q \Lambda Q^{-1}$$

The columns of Q are the eigenvectors of A, and the values in Λ are the associated eigenvalues of A.

The *eigenvalue decomposition* of a square matrix A is given by:

$$A = Q \Lambda Q^{-1}$$

The columns of Q are the eigenvectors of A, and the values in Λ are the associated eigenvalues of A.

For an eigenvector v of A and its eigenvalue λ, we have the important relation:

$$Av = \lambda v$$

The *eigenvalue decomposition* of a square matrix A is given by:

$$A = Q \Lambda Q^{-1}$$

The columns of Q are the eigenvectors of A, and the
are the associated eigenvalues of A.

NOTE

This relationship *defines* what it means to be an eigenvector of A.

For an eigenvector v of A and its eigenvalue λ, we have the important relation:

$$Av = \lambda v$$

The eigenvectors form a basis of the vector space on which A acts (e.g., they are orthogonal).

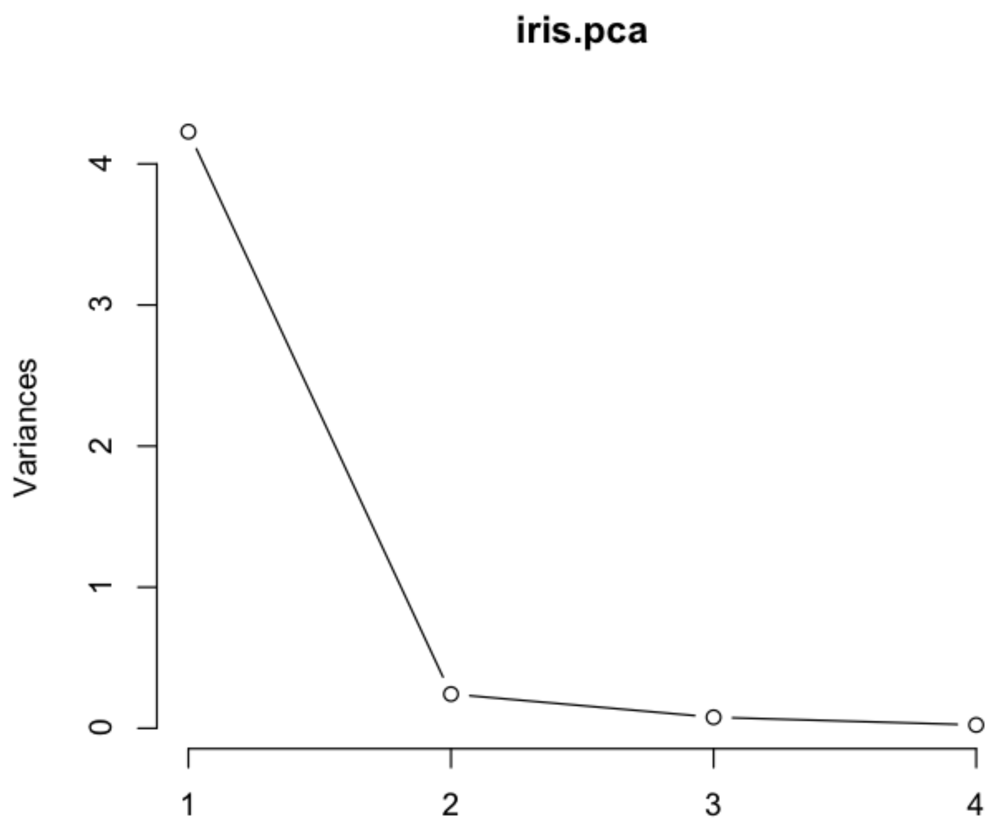
The eigenvectors form a basis of the vector space on which A acts (e.g., they are orthogonal).

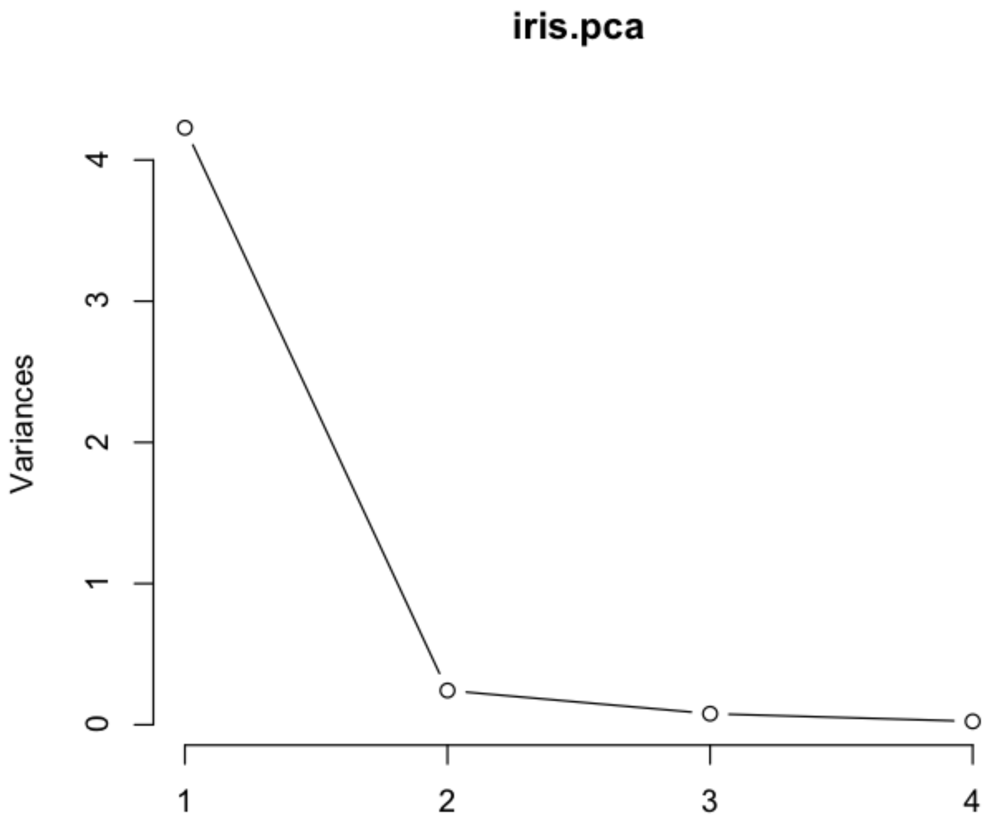
Furthermore the basis elements are ordered by their eigenvalues (from largest to smallest), and these eigenvalues represent the amount of variance explained by each basis element.

The eigenvectors form a basis of the vector space on which A acts (e.g., they are orthogonal).

Furthermore the basis elements are ordered by their eigenvalues (from largest to smallest), and these eigenvalues represent the amount of variance explained by each basis element.

This can be visualized in a scree plot, which shows the amount of variance explained by each basis vector.


iris.pca
Variances
0
1
2
3
4
1
2
3
4



NOTE

Looking at this plot also gives you an idea of how many principal components to keep.

Apply the *elbow test*: keep only those pc's that appear to the left of the elbow in the graph.

1. **Linearity** – The change in basis is a linear projection

2. **Large variances have important structure** – e.g. large signal-to-noise ratio. In other words, we assume that principal components with larger associated variances are signal, while those with lower variances represent noise. NOTE: this is a strong (and not always correct) assumption!

3. **The principal components are orthogonal** – A simplification that makes PCA soluble with linear algebra matrix decomposition techniques

Your turn:

Find the python command to calculate eigenvectors and eigenvalues of a matrix…

SINGULAR VALUE DECOMPOSITION (SVD)

Eigenvalues and eigenvectors exist for a SQUARE matrix.

What if I have a rectangular matrix?

Consider a matrix M *with* n *rows and* d *features.*

Consider a matrix M *with* n *rows and* d *features.*

The **singular value decomposition** *of* A *is given by:*

$$\underset{(m \times n)}{\mathbf{M}} = \underset{(m \times r)}{\mathbf{U}} \; \underset{(r \times r)}{\Sigma} \; \underset{(r \times n)}{\mathbf{V}^{\mathrm{T}}}$$

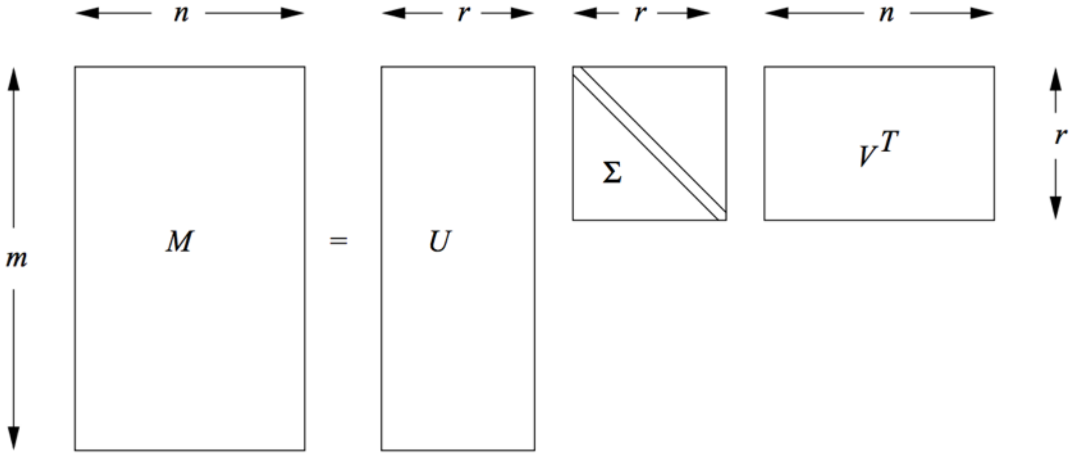
st. U, V *are* **orthogonal** *matrices and* Σ *is a* **diagonal** *matrix.*

→ $\mathrm{U}\mathrm{U}^{\mathrm{T}} = \mathrm{I}_{\mathrm{n}},\ \mathrm{V}\mathrm{V}^{\mathrm{T}} = \mathrm{I}_{\mathrm{d}}$ → $\Sigma_{\mathrm{ij}} = 0 \ (\mathrm{i} \neq \mathrm{j})$

The **singular value decomposition** *of* M *is given by:*

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

(m x n) (m x r) (r x r) (r x n)



source: http://infolab.stanford.edu/~ullman/mmds/ch11.pdf

Ratings of movies by users:

	Matrix	Alien	Star Wars	Casablanca	Titanic
Joe	1	1	1	0	0
Jim	3	3	3	0	0
John	4	4	4	0	0
Jack	5	5	5	0	0
Jill	0	0	0	4	4
Jenny	0	0	0	5	5
Jane	0	0	0	2	2

source: http://infolab.stanford.edu/~ullman/mmds/ch11.pdf

Ratings of movies by users:

	Matrix	Alien	Star Wars	Casablanca	Titanic
Joe	1	1	1	0	0
Jim	3	3	3	0	0
John	4	4	4	0	0
Jack	5	5	5	0	0
Jill	0	0	0	4	4
Jenny	0	0	0	5	5
Jane	0	0	0	2	2

there are two "concepts" underlying the movies:

science-fiction and romance

source: http://infolab.stanford.edu/~ullman/mmds/ch11.pdf

SINGULAR VALUE DECOMPOSITION – EXAMPLE

Ratings of movies by users:

	Matrix	Alien	Star Wars	Casablanca	Titanic
Joe	1	1	1	0	0
Jim	3	3	3	0	0
John	4	4	4	0	0
Jack	5	5	5	0	0
Jill	0	0	0	4	4
Jenny	0	0	0	5	5
Jane	0	0	0	2	2

All the boys rate only science-fiction
All the girls rate only romance

source: http://infolab.stanford.edu/~ullman/mmds/ch11.pdf

Ratings of movies by users:

	Matrix	Alien	Star Wars	Casablanca	Titanic
Joe	1	1	1	0	0
Jim	3	3	3	0	0
John	4	4	4	0	0
Jack	5	5	5	0	0
Jill	0	0	0	4	4
Jenny	0	0	0	5	5
Jane	0	0	0	2	2

$$\underbrace{\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 0 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 0 & 0 & 2 & 2 \end{bmatrix}}_{M} = \underbrace{\begin{bmatrix} .14 & 0 \\ .42 & 0 \\ .56 & 0 \\ .70 & 0 \\ 0 & .60 \\ 0 & .75 \\ 0 & .30 \end{bmatrix}}_{U} \underbrace{\begin{bmatrix} 12.4 & 0 \\ 0 & 9.5 \end{bmatrix}}_{\Sigma} \underbrace{\begin{bmatrix} .58 & .58 & .58 & 0 & 0 \\ 0 & 0 & 0 & .71 & .71 \end{bmatrix}}_{V^{\mathrm{T}}}$$

source: http://infolab.stanford.edu/~ullman/mmds/ch11.pdf

SINGULAR VALUE DECOMPOSITION – EXAMPLE

	Matrix	Alien	Star Wars	Casablanca	Titanic
Joe	1	1	1	0	0
Jim	3	3	3	0	0
John	4	4	4	0	0
Jack	5	5	5	0	0
Jill	0	0	0	4	4
Jenny	0	0	0	5	5
Jane	0	0	0	2	2

$$\underbrace{\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 0 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 0 & 0 & 2 & 2 \end{bmatrix}}_{M} = \underbrace{\begin{bmatrix} .14 & 0 \\ .42 & 0 \\ .56 & 0 \\ .70 & 0 \\ 0 & .60 \\ 0 & .75 \\ 0 & .30 \end{bmatrix}}_{U} \underbrace{\begin{bmatrix} 12.4 & 0 \\ 0 & 9.5 \end{bmatrix}}_{\Sigma} \underbrace{\begin{bmatrix} .58 & .58 & .58 & 0 & 0 \\ 0 & 0 & 0 & .71 & .71 \end{bmatrix}}_{V^{\mathrm{T}}}$$

M: people -> movies

U: people -> concepts

V: concepts -> movies

Σ: the strength of each of the concepts

source: http://infolab.stanford.edu/~ullman/mmds/ch11.pdf

SINGULAR VALUE DECOMPOSITION – A MORE REALISTIC EXAMPLE

	Matrix	Alien	Star Wars	Casablanca	Titanic
Joe	1	1	1	0	0
Jim	3	3	3	0	0
John	4	4	4	0	0
Jack	5	5	5	0	0
Jill	0	0	0	4	4
Jenny	0	0	0	5	5
Jane	0	0	0	2	2

$$\underbrace{\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{bmatrix}}_{M'} =$$

$$\underbrace{\begin{bmatrix} .13 & .02 & -.01 \\ .41 & .07 & -.03 \\ .55 & .09 & -.04 \\ .68 & .11 & -.05 \\ .15 & -.59 & .65 \\ .07 & -.73 & -.67 \\ .07 & -.29 & .32 \end{bmatrix}}_{U} \underbrace{\begin{bmatrix} 12.4 & 0 & 0 \\ 0 & 9.5 & 0 \\ 0 & 0 & 1.3 \end{bmatrix}}_{\Sigma} \underbrace{\begin{bmatrix} .56 & .59 & .56 & .09 & .09 \\ .12 & -.02 & .12 & -.69 & -.69 \\ .40 & -.80 & .40 & .09 & .09 \end{bmatrix}}_{V^{\mathrm{T}}}$$

source: http://infolab.stanford.edu/~ullman/mmds/ch11.pdf

SINGULAR VALUE DECOMPOSITION - EXAMPLE

How to reduce dimensions?

Drop Low Singular Values -> eliminate corresponding rows of U and V

$$\underbrace{\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{bmatrix}}_{M'} =$$

$$\underbrace{\begin{bmatrix} .13 & .02 & -.01 \\ .41 & .07 & -.03 \\ .55 & .09 & -.04 \\ .68 & .11 & -.05 \\ .15 & -.59 & .65 \\ .07 & -.73 & -.67 \\ .07 & -.29 & .32 \end{bmatrix}}_{U} \underbrace{\begin{bmatrix} 12.4 & 0 & 0 \\ 0 & 9.5 & 0 \\ 0 & 0 & 1.3 \end{bmatrix}}_{\Sigma} \underbrace{\begin{bmatrix} .56 & .59 & .56 & .09 & .09 \\ .12 & -.02 & .12 & -.69 & -.69 \\ .40 & -.80 & .40 & .09 & .09 \end{bmatrix}}_{V^{\mathrm{T}}}$$

$$\Rightarrow \underbrace{\begin{bmatrix} 12.4 & 0 \\ 0 & 9.5 \end{bmatrix}}_{\Sigma}$$

source: http://infolab.stanford.edu/~ullman/mmds/ch11.pdf

How to reduce dimensions?

<u>Drop Low Singular Values</u>

$$\underset{M'}{\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{bmatrix}} =$$

$$\underset{U}{\begin{bmatrix} .13 & .02 & -.01 \\ .41 & .07 & -.03 \\ .55 & .09 & -.04 \\ .68 & .11 & -.05 \\ .15 & -.59 & .65 \\ .07 & -.73 & -.67 \\ .07 & -.29 & .32 \end{bmatrix}} \underset{\Sigma}{\begin{bmatrix} 12.4 & 0 & 0 \\ 0 & 9.5 & 0 \\ 0 & 0 & 1.3 \end{bmatrix}} \underset{V^{\mathrm{T}}}{\begin{bmatrix} .56 & .59 & .56 & .09 & .09 \\ .12 & -.02 & .12 & -.69 & -.69 \\ .40 & -.80 & .40 & .09 & .09 \end{bmatrix}}$$

$$\begin{bmatrix} .13 & .02 \\ .41 & .07 \\ .55 & .09 \\ .68 & .11 \\ .15 & -.59 \\ .07 & -.73 \\ .07 & -.29 \end{bmatrix} \begin{bmatrix} 12.4 & 0 \\ 0 & 9.5 \end{bmatrix} \begin{bmatrix} .56 & .59 & .56 & .09 & .09 \\ .12 & -.02 & .12 & -.69 & -.69 \end{bmatrix}$$

$$= \begin{bmatrix} 0.93 & 0.95 & 0.93 & .014 & .014 \\ 2.93 & 2.99 & 2.93 & .000 & .000 \\ 3.92 & 4.01 & 3.92 & .026 & .026 \\ 4.84 & 4.96 & 4.84 & .040 & .040 \\ 0.37 & 1.21 & 0.37 & 4.04 & 4.04 \\ 0.35 & 0.65 & 0.35 & 4.87 & 4.87 \\ 0.16 & 0.57 & 0.16 & 1.98 & 1.98 \end{bmatrix}$$

source: http://infolab.stanford.edu/~ullman/mmds/ch11.pdf

For a general SVD, the columns of U are the eigenvectors of AA^T, and the columns of V are the eigenvectors of A^TA.

Also, the singular values of A are the square roots of the eigenvalues of AA^T and A^TA.

Q: How do you interpret the SVD?

A: Recall that given a set of n points in d-dimensional space (e.g., a matrix A), we want to find the best $k < d$ dimensional subspace to represent the data.

OTHER METHODS

Whereas PCA and SVD create new coordinates by transform the old coordinates, factor analysis requires new coordinates to be specified externally.

Whereas PCA and SVD create new coordinates by transform the old coordinates, factor analysis requires new coordinates to be specified externally.

These new coordinates are associated with *hidden* or *latent* features that we think our data depends on.

Whereas PCA and SVD create new coordinates by transform the old coordinates, factor analysis requires new coordinates to be specified externally.

These new coordinates are associated with *hidden* or *latent* features that we think our data depends on.

The old coordinates are then modeled as linear combinations of the latent features.

For example, consider a dataset that represents the results of a decathalon (rows = participants, columns = events, entries = times).

For example, consider a dataset that represents the results of a decathalon (rows = participants, columns = events, entries = times).

Though this dataset contains 10 features X_i, we may be interested in modeling these features as functions of *latent variables* such as the speed and strength of the participants:

$$X_i = \lambda_1 f_1 + \lambda_2 f_2 + \varepsilon$$

For example, consider a dataset that represents the results of a decathalon (rows = participants, columns = events, entries = times).

Though this dataset contains 10 features X_i, we may be interested in modeling these features as functions of *latent variables* such as the speed and strength of the participants:

$$X_i = \lambda_1 f_1 + \lambda_2 f_2 + \varepsilon$$

This would allow us to analyze the data in a more fundamental way.

SVD, PCA, and factor analysis are all linear techniques (eg, we use a linear transformation to embed the in a lower-dimensional space).

However, sometimes linear techniques are not sufficient.

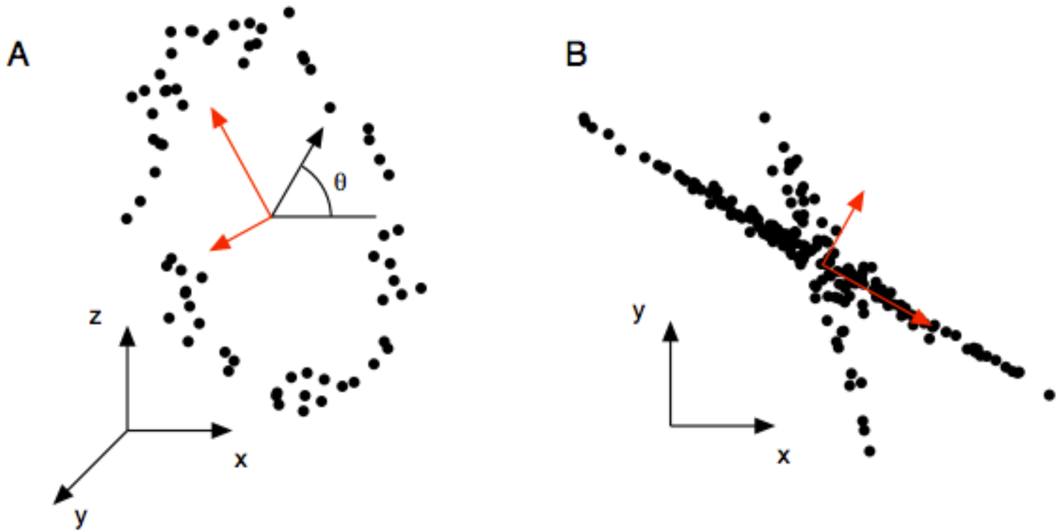


FIG. 6 Example of when PCA fails (red lines). (a) Tracking a person on a ferris wheel (black dots). All dynamics can be described by the phase of the wheel θ, a non-linear combination of the naive basis. (b) In this example data set, non-Gaussian distributed data and non-orthogonal axes causes PCA to fail. The axes with the largest variance do not correspond to the appropriate answer.

Some methods for nonlinear dimensional reduction (or *manifold learning*) include:

multidimensional scaling: low-dim embedding that preserves pairwise distances

locally linear embedding: approximates local structure of data (nbd preserving embedding)

Some methods for nonlinear dimensional reduction (or *manifold learning*) include:

kernel PCA: exploits PCA dependence on inner product (same logic as SVM)

isomap: nonlinear dim reduction via MDS using geodesic (surface-bound) distances

In any case, the key difficulties with dimensionality reduction are time/space complexity, randomness (eg different results for different runs), and selecting the number of dimensions in the lower-dim subspace.

In any case, the key difficulties with dimensionality reduction are time/space complexity, randomness (eg different results for different runs), and selecting the number of dimensions in the lower-dim subspace.

Furthermore, there’s an obvious (bias/variance) tradeoff between the number of subspace dimensions and the size of approximation error.

Exercise in pairs:

- Eigenfaces - RandomizedPCA
- Non-negative components - NMF
- Independent components - FastICA
- Sparse comp. - MiniBatchSparsePCA
- MiniBatchDictionaryLearning
- Cluster centers - MiniBatchKMeans
- Factor Analysis components - FA

http://scikit-learn.org/stable/auto_examples/decomposition/plot_faces_decomposition.html