PCA, Kernel PCA, ICA
Learning Representations.
Dimensionality Reduction

Lecture 9

Big & High-Dimensional Data

- High-Dimensions = Lot of Features
- Document classification

  Features per document =

  thousands of words/unigrams

  millions of bigrams

  contextual information

Surveys - Netflix

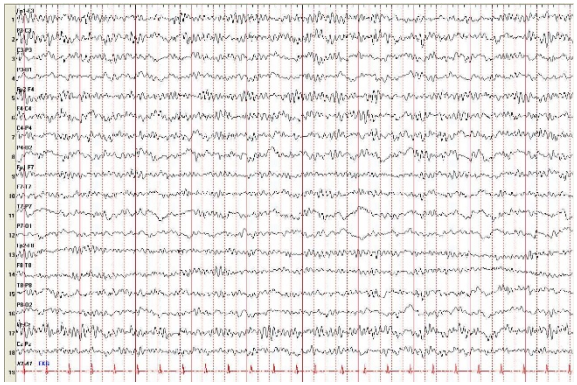
480189 users x 17770 movies

Big & High-Dimensional Data

- Brain signal/Imaging Data
  - EEG
  - fMRI

    Hundreds of locations

    and time points



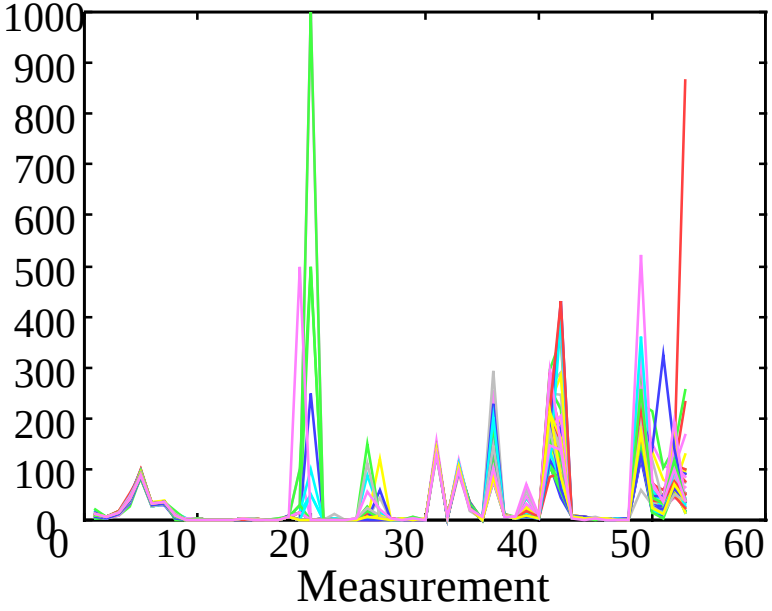
- High-dimensional image data

Data Presentation

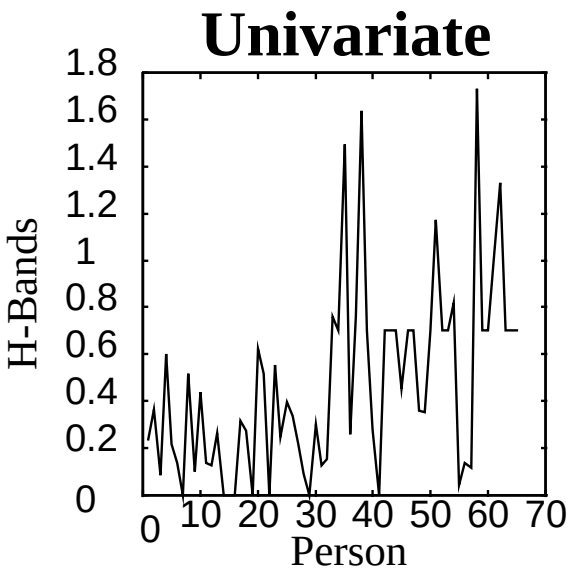
- Example: 53 Blood and urine measurements (wet chemistry) from 65 people (33 alcoholics, 32 non-alcoholics).
- Matrix Format

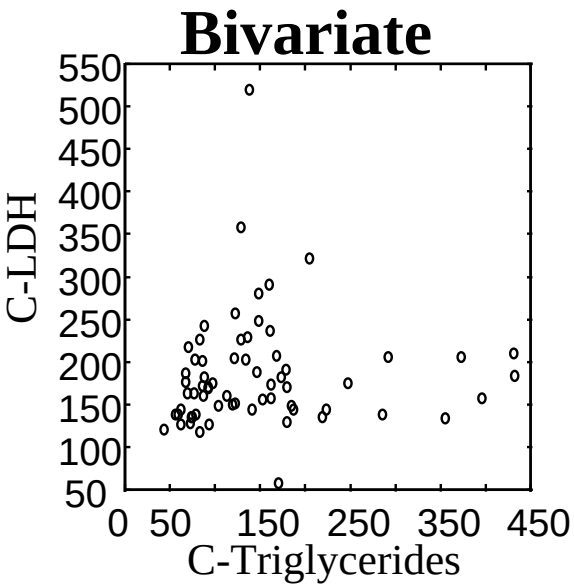
	H-WBC	H-RBC	H-Hgb	H-Hct	H-MCV	H-MCH	H-MCHC
A1	8.0000	4.8200	14.1000	41.0000	85.0000	29.0000	34.0000
A2	7.3000	5.0200	14.7000	43.0000	86.0000	29.0000	34.0000
A3	4.3000	4.4800	14.1000	41.0000	91.0000	32.0000	35.0000
A4	7.5000	4.4700	14.9000	45.0000	101.0000	33.0000	33.0000
A5	7.3000	5.5200	15.4000	46.0000	84.0000	28.0000	33.0000
A6	6.9000	4.8600	16.0000	47.0000	97.0000	33.0000	34.0000
A7	7.8000	4.6800	14.7000	43.0000	92.0000	31.0000	34.0000
A8	8.6000	4.8200	15.8000	42.0000	88.0000	33.0000	37.0000
A9	5.1000	4.7100	14.0000	43.0000	92.0000	30.0000	32.0000

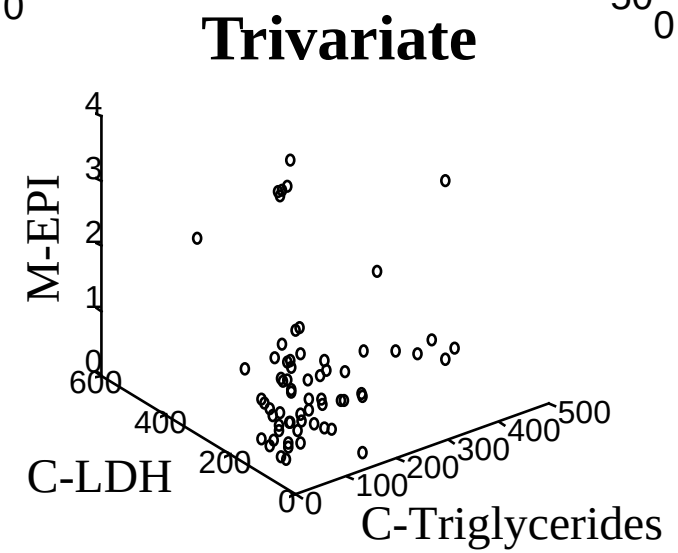
- Spectral Format



Data Presentation







Data Presentation

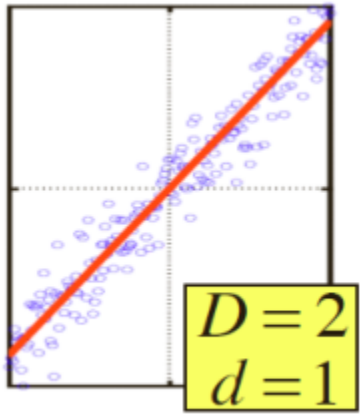
- Better presentation than ordinate axes?
- Do we need a 53 dimension space to view data?
- How to find the ‘best’ low dimension space that conveys maximum useful information?
- One answer: Find “Principal Components”

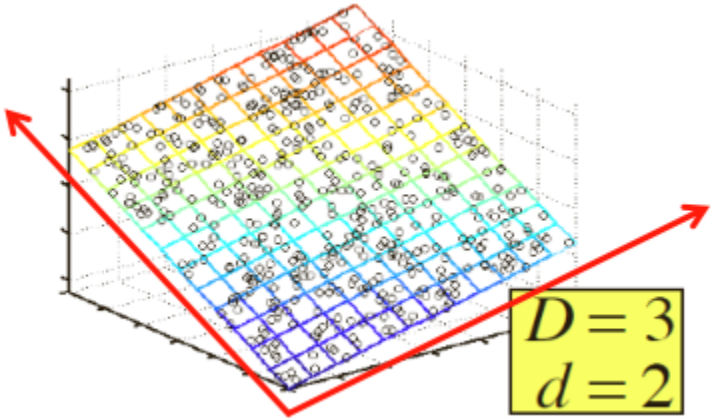
Learning Representations

- PCA, Kernel PCA, ICA: Powerful unsupervised learning techniques for extracting hidden (potentially lower dimensional) structure from high dimensional datasets.
- Useful for:
  - Visualization
  - More efficient use of resources(e.g., time, memory, communication)
  - Statistical: fewer dimensions → better generalization
  - Noise removal (improving data quality)
  - Further processing by machine learning algorithms

Principal Component Analysis (PCA)

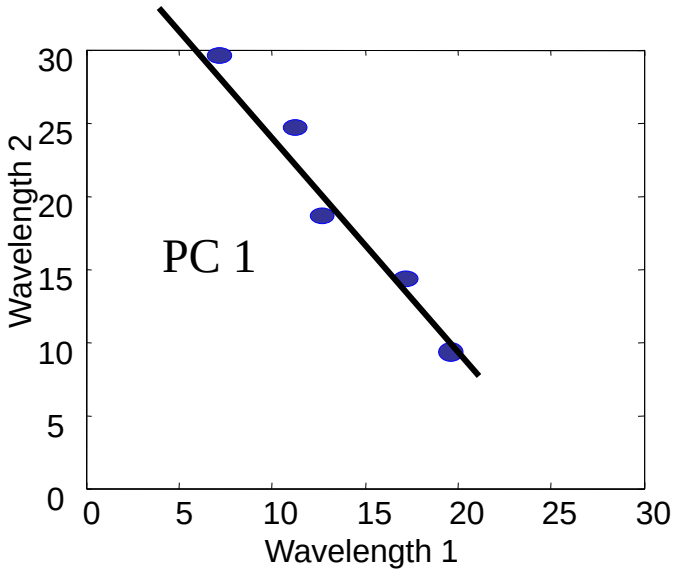
- What is PCA: Unsupervised technique for extracting variance structure from high dimensional datasets.
- PCA is an orthogonal projection or transformation of the data into a (possibly lower dimensional) subspace so that the variance of the projected data is maximized.

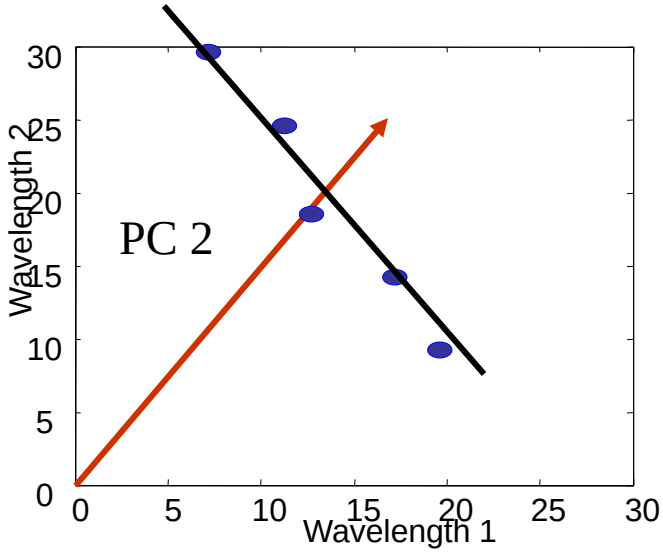




Principal Component Analysis (PCA)

- Principal Components (PC) are orthogonal directions that capture most of the variance in the data.
- First PC is direction of maximum variance
- Subsequent PCs are orthogonal to 1st PC and describe maximum residual variance

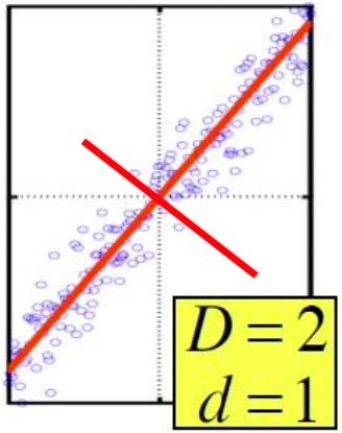




The Goal

We wish to explain/summarize the underlying variance-covariance structure of a large set of variables through a few linear combinations of these variables.

Principal Component Analysis (PCA)



Let $v_1, v_2, \ldots, v_d$ denote the d principal components.

$v_i \cdot v_j = 0,\ i \neq j$ and $v_i \cdot v_i = 1,\ i = j$

Assume data is centered (we extracted the sample mean).

Let $X = [x_1, x_2, \ldots, x_n]$ (columns are the datapoints)

Find vector that maximizes sample variance of projected data

$$\frac{1}{n}\sum_{i=1}^{n}(v^T x_i)^2 = v^T X X^T v$$

$$\max_v \ v^T X X^T v \quad s.t. \quad v^T v = 1$$

Wrap constraints into the objective function

Lagrangian: $\max_v v^T X X^T v - \lambda v^T v$

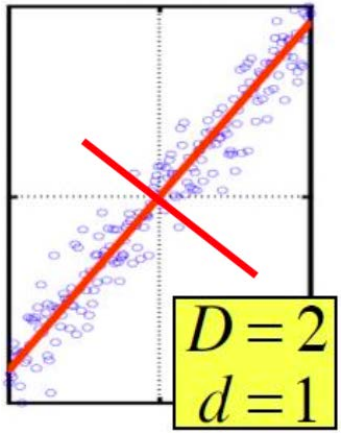
$$\partial/\partial v = 0 \qquad (XX^T - \lambda I)v = 0 \qquad \Rightarrow \boxed{(XX^T)v = \lambda v}$$

Principal Component Analysis (PCA)

$(XX^T)v = \lambda v$, so v (the first PC) is the eigenvector of sample correlation/covariance matrix XX^T

Sample variance of projection $v^T XX^T v = \lambda v^T v = \lambda$

Thus, the eigenvalue λ denotes the amount of variability captured along that dimension (aka amount of energy along that dimension).



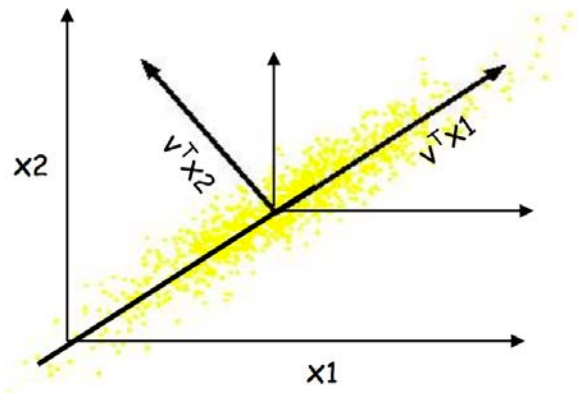
Eigenvalues $\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq \cdots$

- The 1st PC v_1 is the eigenvector of the sample covariance matrix XX^T associated with the largest eigenvalue
- The 2nd PC v_2 is the eigenvector of the sample covariance matrix XX^T associated with the second largest eigenvalue
- And so on …

Principal Component Analysis (PCA)

- So, the new axes are the eigenvectors of the matrix of sample correlations XX^T of the data.

- Transformed features are uncorrelated.



- Geometrically: centering followed by rotation.

  - Linear transformation

Key computation: eigendecomposition of XX^T (closely related to SVD of X).

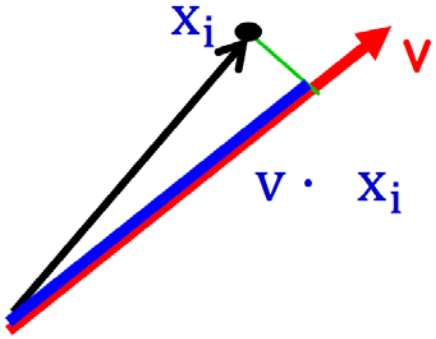
Two Interpretations

So far: Maximum Variance Subspace. PCA finds vectors v such that projections on to the vectors capture maximum variance in the data

$$\frac{1}{n}\sum_{i=1}^{n}(v^T x_i)^2 = v^T X X^T v$$

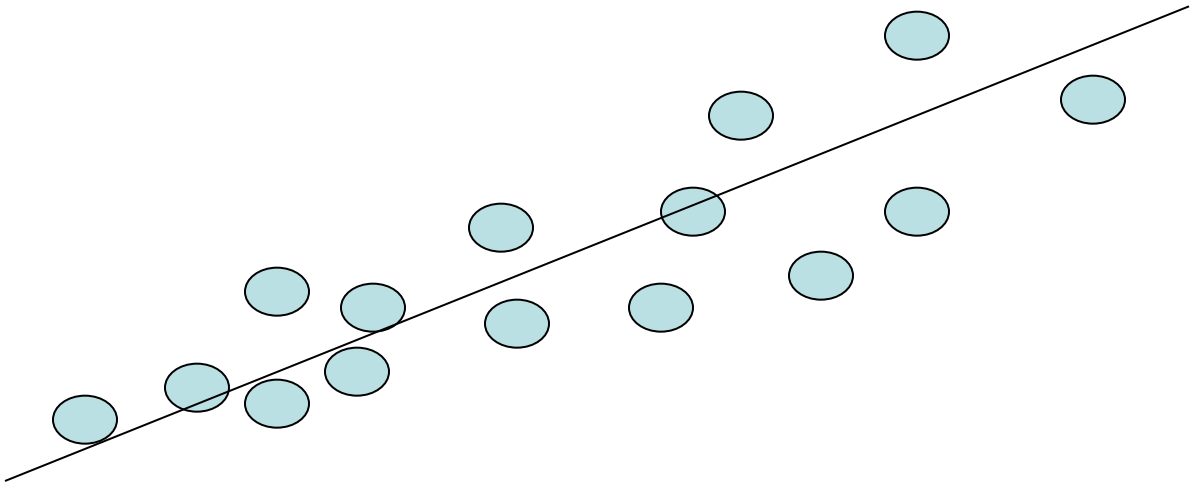
Alternative viewpoint: Minimum Reconstruction Error. PCA finds vectors v such that projection on to the vectors yields minimum MSE reconstruction

$$\frac{1}{n}\sum_{i=1}^{n}\|x_i - (v^T x_i)v\|^2$$



Algebraic Interpretation – 1D

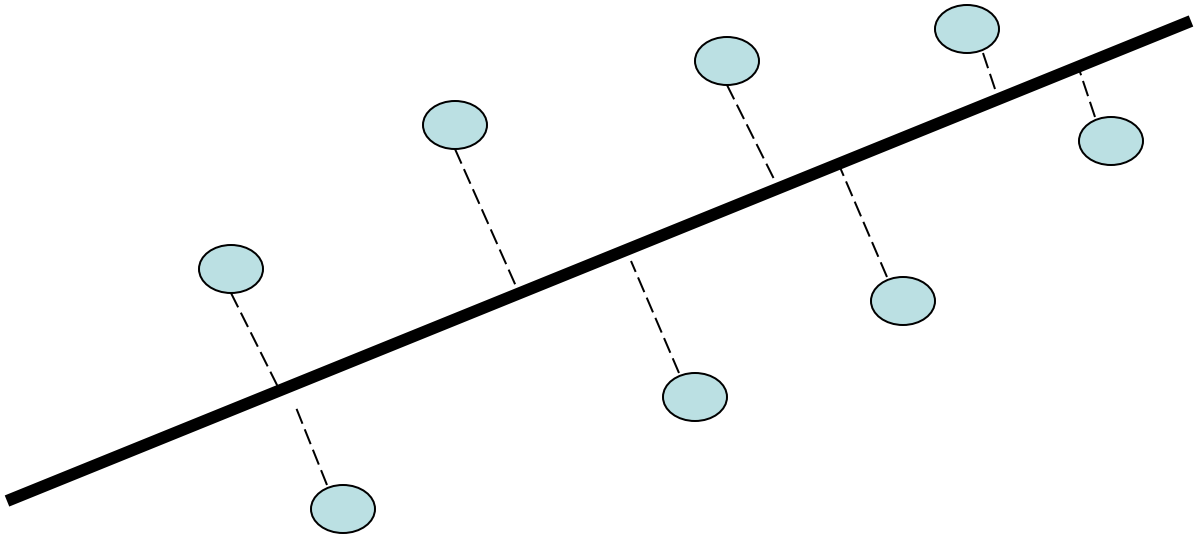
- Given m points in a n dimensional space, for large n, how does one project on to a 1 dimensional space?



- Choose a line that fits the data so the points are spread out well along the line

Algebraic Interpretation – 1D

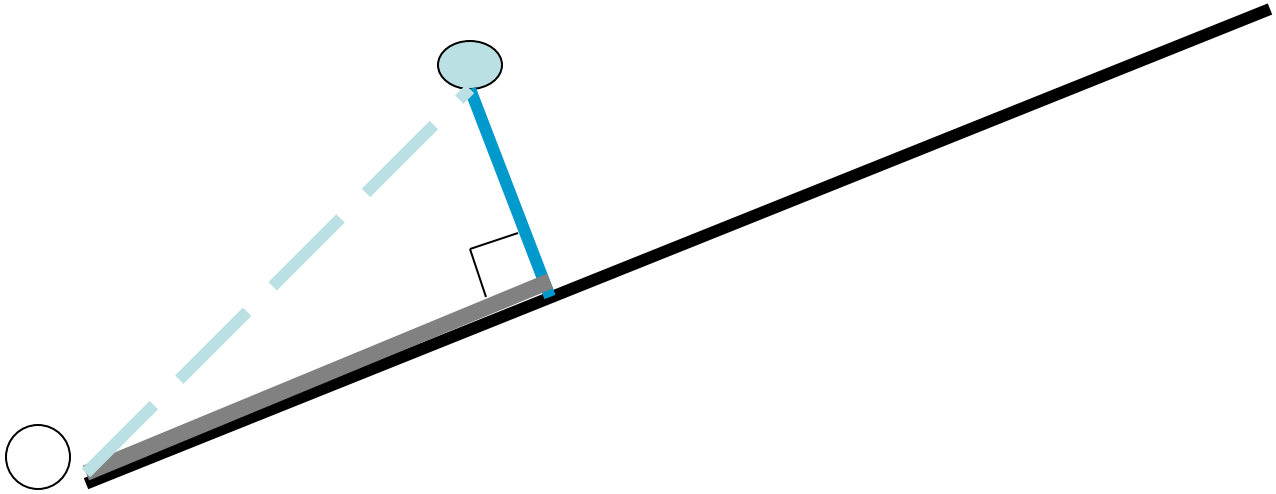
- Formally, minimize sum of squares of distances to the line.



- Why sum of squares? Because it allows fast minimization, assuming the line passes through 0

Algebraic Interpretation – 1D

- Minimizing sum of squares of distances to the line is the same as maximizing the sum of squares of the projections on that line, thanks to Pythagoras.



Why? Pythagorean Theorem

E.g., for the first component.

Maximum Variance Direction: 1st PC a vector v such that projection on to this vector capture maximum variance in the data (out of all possible one dimensional projections)

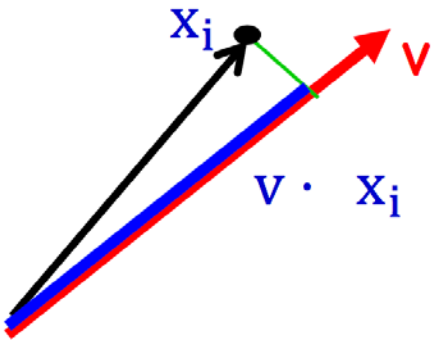
$$\frac{1}{n}\sum_{i=1}^{n}(v^T x_i)^2 = v^T X X^T v$$

Minimum Reconstruction Error: 1st PC a vector v such that projection on to this vector yields minimum MSE reconstruction

blue^2 + green^2 = black^2

black^2 is fixed (it's just the data)

So, maximizing blue^2 is equivalent to minimizing green^2



PCA: *General*

From k original variables: $x_1, x_2, \ldots, x_k$:

Produce k new variables: $y_1, y_2, \ldots, y_k$:

$y_1 = a_{11}x_1 + a_{12}x_2 + \ldots + a_{1k}x_k$

$y_2 = a_{21}x_1 + a_{22}x_2 + \ldots + a_{2k}x_k$

...

$y_k = a_{k1}x_1 + a_{k2}x_2 + \ldots + a_{kk}x_k$

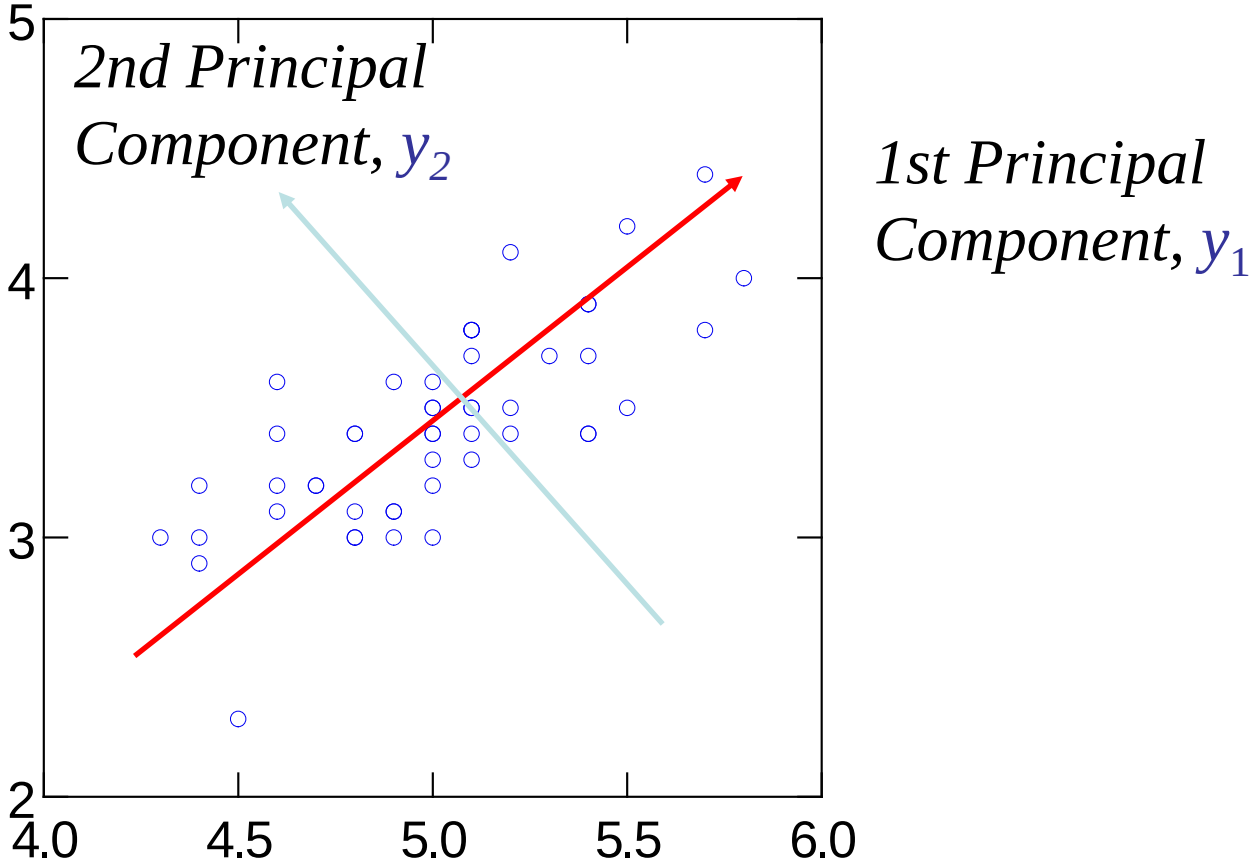
such that:

y_k's are uncorrelated (orthogonal)

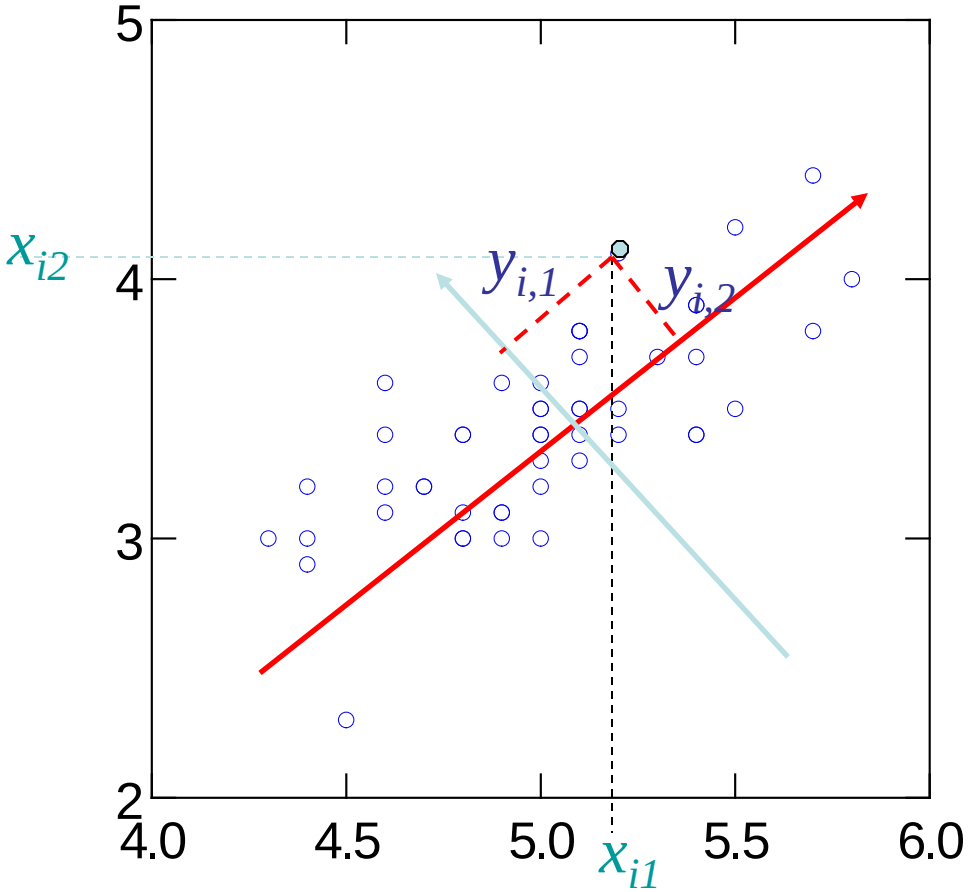
y_1 explains as much as possible of original variance in data set

y_2 explains as much as possible of remaining variance

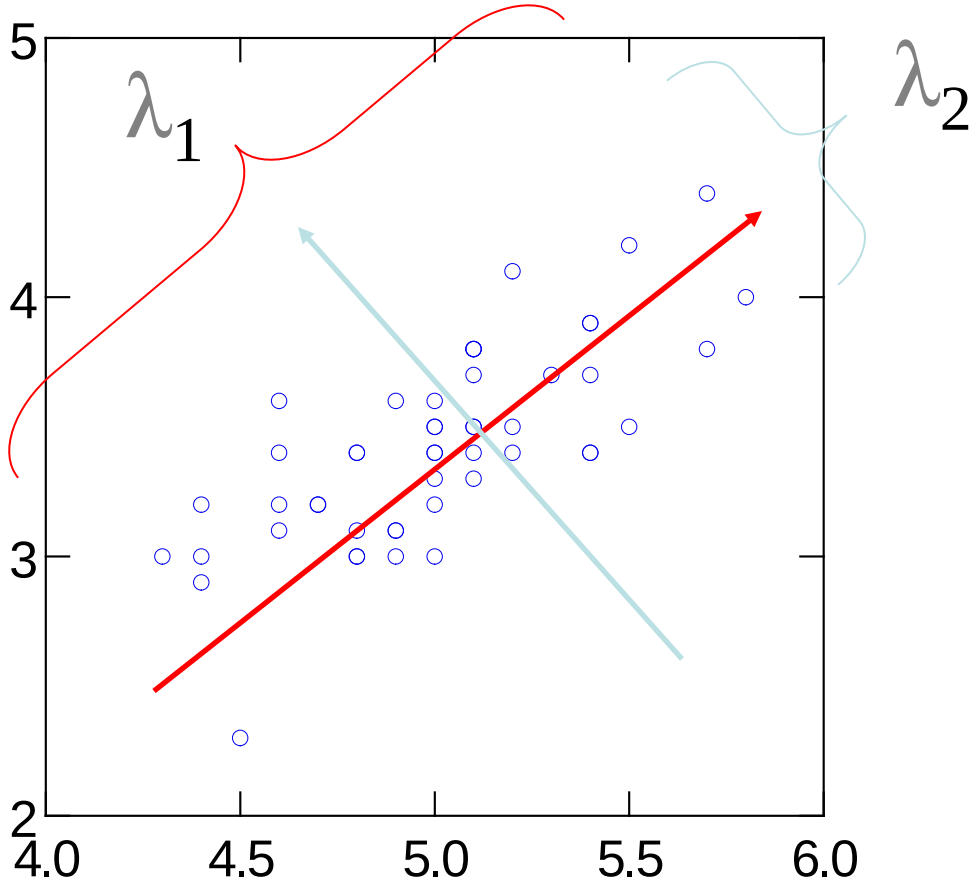
etc.


2nd Principal Component, y_2
1st Principal Component, y_1
5
4
3
2
4.0
4.5
5.0
5.5
6.0

PCA Scores

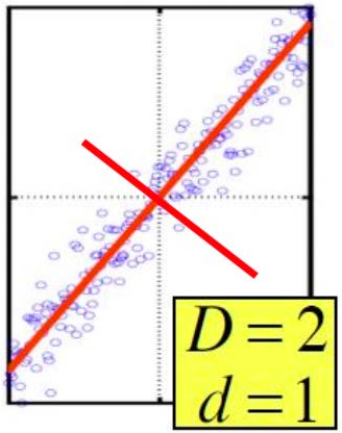


PCA Eigenvalues



Dimensionality Reduction using PCA

The eigenvalue λ denotes the amount of variability captured along that dimension (aka amount of energy along that dimension).



Zero eigenvalues indicate no variability along those directions => data lies exactly on a linear subspace

Only keep data projections onto principal components with non-zero eigenvalues, say $v_1, \ldots, v_k$, where $k = \text{rank}(XX^T)$

Original representation

Data point

$$x_i = (x_i^1, \ldots, x_i^D)$$

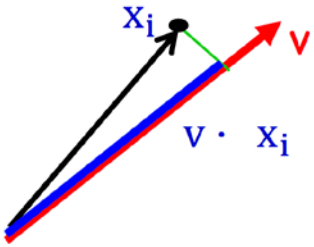
D-dimensioinal vector

Transformed representation

projection

$$(v_1 \cdot x^i, \ldots, v_d \cdot x^i)$$

d-dimensional vector

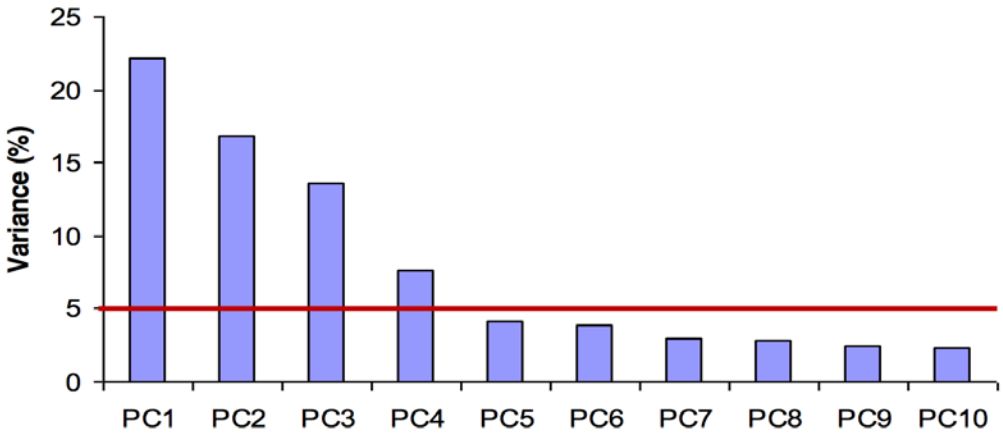


Dimensionality Reduction using PCA

In high-dimensional problems, data sometimes lies near a linear subspace, as noise introduces small variability

Only keep data projections onto principal components with large eigenvalues

Can *ignore* the components of smaller significance.



Might lose some info, but if eigenvalues are small, do not lose much

PCA Summary until now

- Rotates multivariate dataset into a new configuration which is easier to interpret

- Purposes
  - simplify data
  - look at relationships between variables
  - look at patterns of units

A 2D Numerical Example

PCA Example –STEP 1

- Subtract the mean

  from each of the data dimensions. All the x values have $\bar{x}$ subtracted and y values have $\bar{y}$ subtracted from them. This produces a data set whose mean is zero.

  Subtracting the mean makes variance and covariance calculation easier by simplifying their equations. The variance and co-variance values are not affected by the mean value.

PCA Example –STEP 1

DATA:

x	y
2.5	2.4
0.5	0.7
2.2	2.9
1.9	2.2
3.1	3.0
2.3	2.7
2	1.6
1	1.1
1.5	1.6
1.1	0.9

ZERO MEAN DATA:

x	y
.69	.49
-1.31	-1.21
.39	.99
.09	.29
1.29	1.09
.49	.79
.19	-.31
-.81	-.81
-.31	-.31
-.71	-1.01

PCA Example –STEP 1

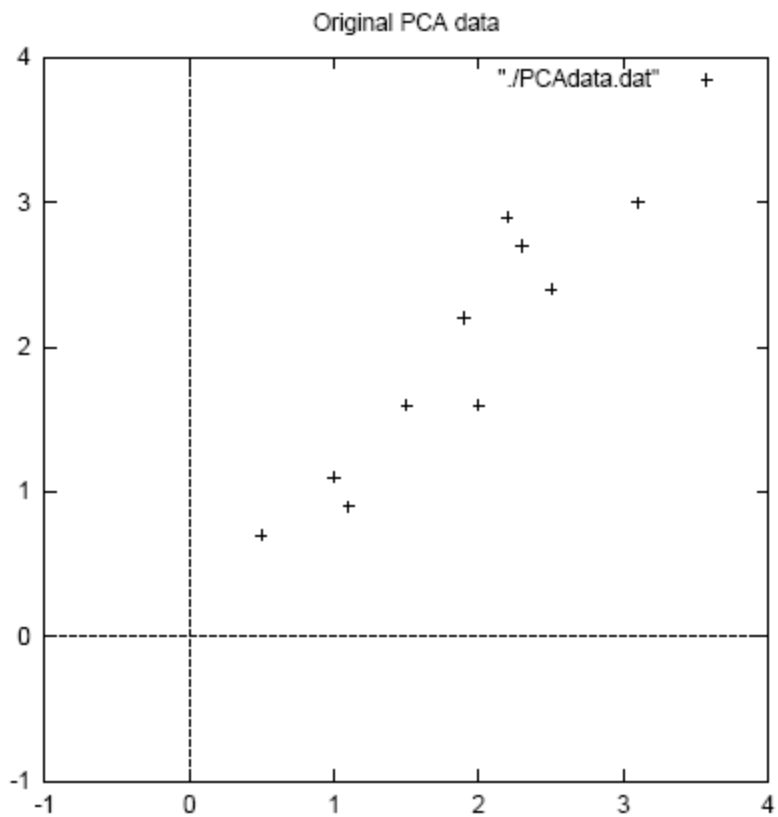


Figure 3.1: PCA example data, original data on the left, data with the means subtracted on the right, and a plot of the data

PCA Example –STEP 2

- Calculate the covariance matrix

$$\text{cov} = \begin{pmatrix} .616555556 & .615444444 \\ .615444444 & .716555556 \end{pmatrix}$$

- since the non-diagonal elements in this covariance matrix are positive, we should expect that both the x and y variable increase together.

PCA Example –STEP 3

- Calculate the eigenvectors and eigenvalues of the covariance matrix

$$\text{eigenvalues} = \begin{pmatrix} .0490833989 \\ 1.28402771 \end{pmatrix}$$

$$\text{eigenvectors} = \begin{pmatrix} -.735178656 & -.677873399 \\ .677873399 & -.735178656 \end{pmatrix}$$

PCA Example –STEP 3

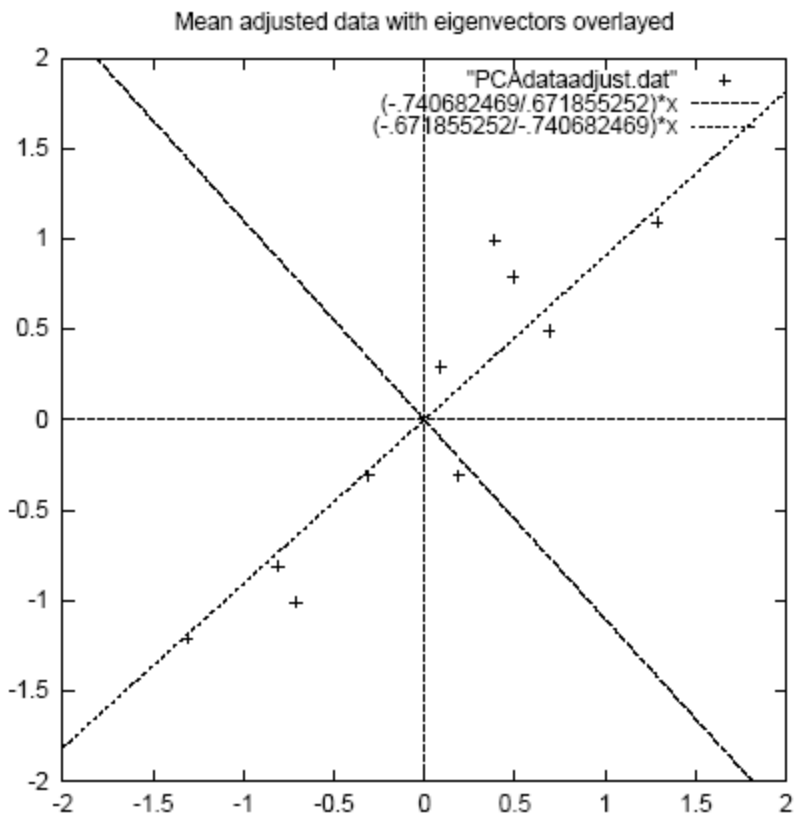


Figure 3.2: A plot of the normalised data (mean subtracted) with the eigenvectors of the covariance matrix overlayed on top.

- eigenvectors are plotted as diagonal dotted lines on the plot.
- Note they are perpendicular to each other.
- Note one of the eigenvectors goes through the middle of the points, like drawing a line of best fit.
- The second eigenvector gives us the other, less important, pattern in the data, that all the points follow the main line, but are off to the side of the main line by some amount.

PCA Example –STEP 4

- Reduce dimensionality and form *feature vector*

  the eigenvector with the *highest* eigenvalue is the *principle component* of the data set.

  In our example, the eigenvector with the larges eigenvalue was the one that pointed down the middle of the data.

  Once eigenvectors are found from the covariance matrix, the next step is to order them by eigenvalue, highest to lowest. This gives you the components in order of significance.

PCA Example –STEP 4

Now, if you like, you can decide to *ignore* the components of lesser significance.

You do lose some information, but if the eigenvalues are small, you don't lose much

- n dimensions in your data
- calculate n eigenvectors and eigenvalues
- choose only the first p eigenvectors
- final data set has only p dimensions.

PCA Example –STEP 4

- Feature Vector

$$\text{FeatureVector} = (\text{eig}_1\ \text{eig}_2\ \text{eig}_3\ \ldots\ \text{eig}_n)$$

We can either form a feature vector with both of the eigenvectors:

$$\begin{pmatrix} -.677873399 & -.735178656 \\ -.735178656 & .677873399 \end{pmatrix}$$

or, we can choose to leave out the smaller, less significant component and only have a single column:

$$\begin{pmatrix} -.677873399 \\ -.735178656 \end{pmatrix}$$

PCA Example –STEP 5

- Deriving the new data

FinalData = RowFeatureVector x RowZeroMeanData

RowFeatureVector is the matrix with the eigenvectors in the columns *transposed* so that the eigenvectors are now in the rows, with the most significant eigenvector at the top

RowZeroMeanData is the mean-adjusted data *transposed*, ie. the data items are in each column, with each row holding a separate dimension.

PCA Example –STEP 5

<u>FinalData transpose:</u>
<u>dimensions along columns</u>

x	y
-.827970186	-.175115307
1.77758033	.142857227
-.992197494	.384374989
-.274210416	.130417207
-1.67580142	-.209498461
-.912949103	.175282444
.0991094375	-.349824698
1.14457216	.0464172582
.438046137	.0177646297
1.22382056	-.162675287

PCA Example –STEP 5

http://kybele.psych.cornell.edu/~edelman/Psych-465-Spring-2003/PCA-tutorial.pdf

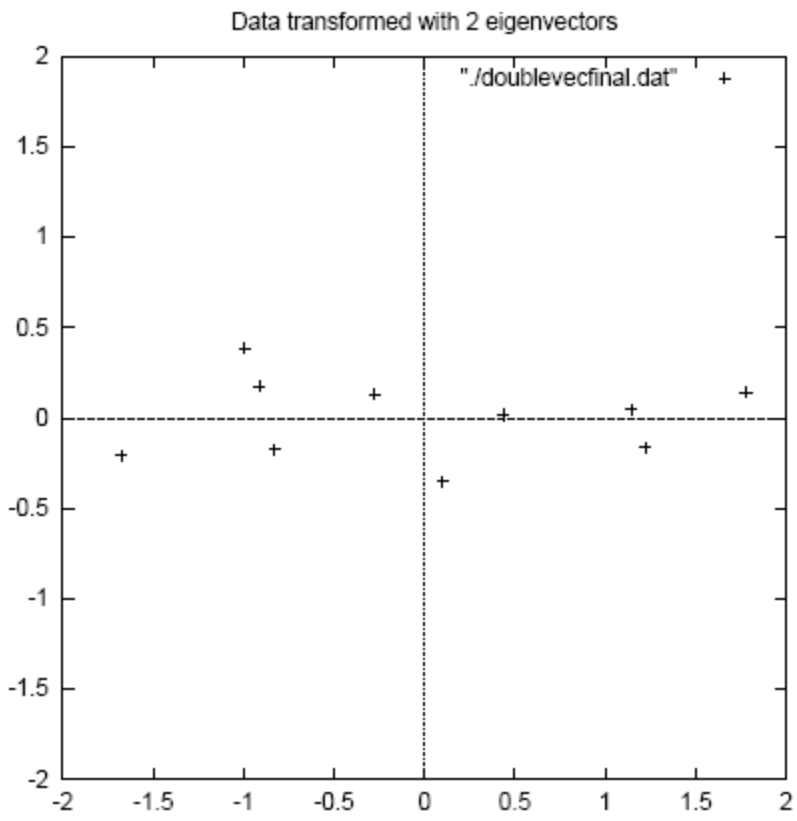


Figure 3.3: The table of data by applying the PCA analysis using both eigenvectors, and a plot of the new data points.

Reconstruction of original Data

- If we reduced the dimensionality, obviously, when reconstructing the data we would lose those dimensions we chose to discard. In our example let us assume that we considered only the x dimension…

Reconstruction of original Data

http://kybele.psych.cornell.edu/~edelman/Psych-465-Spring-2003/PCA-tutorial.pdf

x
-.827970186
1.77758033
-.992197494
-.274210416
-1.67580142
-.912949103
.0991094375
1.14457216
.438046137
1.22382056

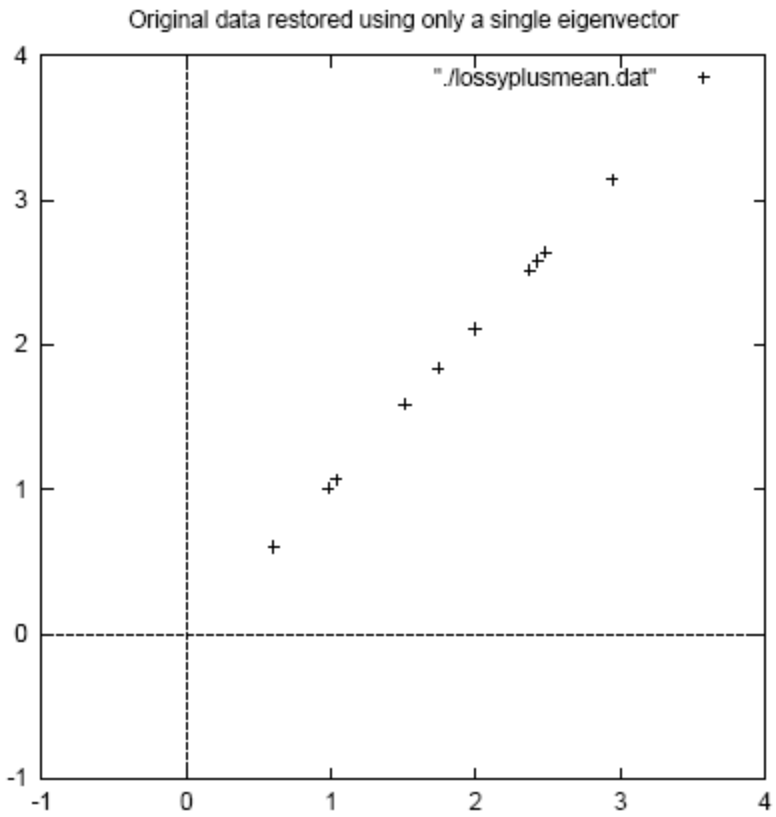


Figure 3.5: The reconstruction from the data that was derived using only a single eigenvector

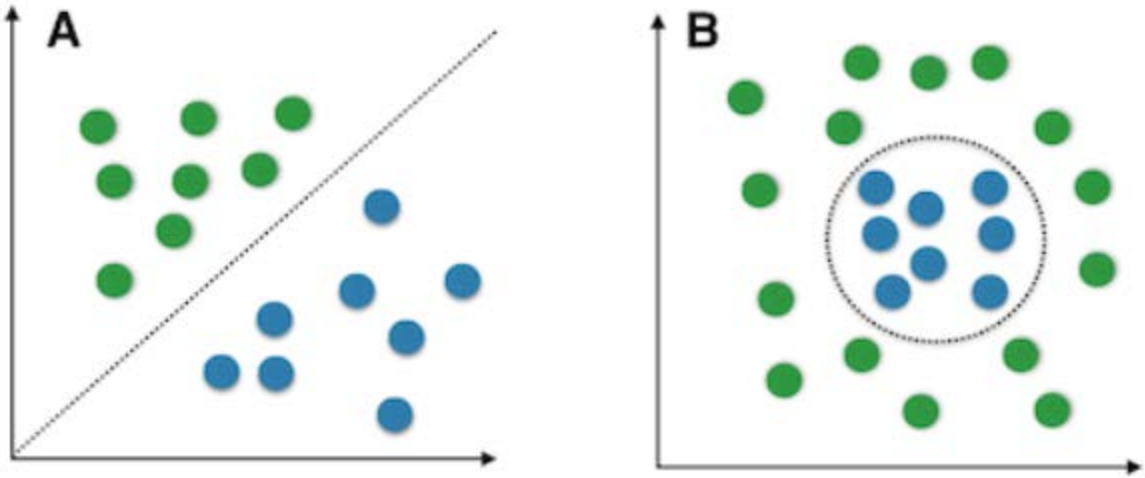
PCA Discussion

- Strengths
  - Eigenvector method
  - No tuning of the parameters
  - No local optima
- Weaknesses
  - Limited to second order statistics
  - Limited to linear projections

Kernel PCA (Kernel Principal Component Analysis)

Kernel PCA

- The “classic” PCA approach described above is a linear projection technique that works well if the data is linearly separable. However, in the case of linearly inseparable data, a nonlinear technique is required if the task is to reduce the dimensionality of a dataset.



Kernel functions and the kernel trick

- The basic idea to deal with linearly inseparable data is to project it onto a higher dimensional space where it becomes linearly separable. Let us call this nonlinear mapping function ϕ so that the mapping of a sample x can be written as $x \rightarrow \phi(x)$, which is called “kernel function.”
- Now, the term “kernel” describes a function that calculates the dot product of the images of the samples x under ϕ.

$$\kappa(\mathbf{x_i}, \mathbf{x_j}) = \phi(\mathbf{x_i})\phi(\mathbf{x_j})^T$$

Kernel functions

- In other words, the function ϕ maps the original d-dimensional features into a larger, k-dimensional feature space by creating nononlinear combinations of the original features.
- For example, if x consists of 2 features:

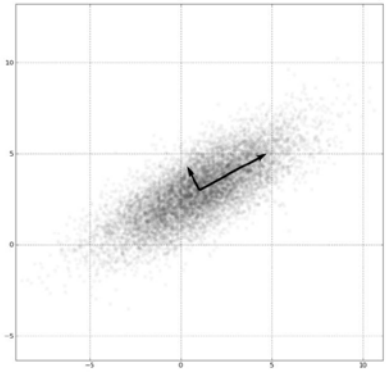
$$\mathbf{x} = \begin{bmatrix} x_1 & x_2 \end{bmatrix}^T \qquad \mathbf{x} \in I\!R^d$$

$$\Downarrow \phi$$

$$\mathbf{x}' = \begin{bmatrix} x_1 & x_2 & x_1 x_2 & x_1^2 & x_1 x_2^3 & \ldots \end{bmatrix}^T \qquad \mathbf{x} \in I\!R^k (k >> d)$$

Properties of PCA

- Given a set of n centered observations $x_i \epsilon R^D$, 1st PC is the direction that maximizes the variance



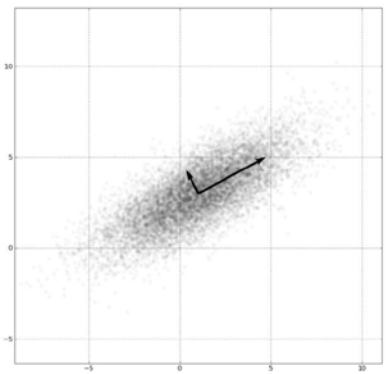
$$X = (x_1, x_2, .., x_n)$$

$$v_1 = argmax_{\|v\|=1} \frac{1}{n} \sum_i (v^T x_i)^2$$

$$= argmax_{\|v\|=1} \frac{1}{n} v^T X X^T v$$

- Covariance matrix $C = \frac{1}{n} X X^T$
- v_1 can be found by solving the eigenvalue problem:
  - $Cv_1 = \lambda v_1$ (of maximum λ)

Properties of PCA



- Given a set of n centered observations $x_i \epsilon R^D$, 1st PC is the direction that maximizes the varianc

$$X = (x_1, x_2, .., x_n)$$

$$v_1 = argmax_{\|v\|=1} \frac{1}{n} \sum_i (v^T x_i)^2$$

$$= argmax_{\|v\|=1} \frac{1}{n} v^T X X^T v$$

- Covariance matrix $C = \frac{1}{n} X X^T$ is a DxD matrix
  - the (i,j) entry of XX^T is the correlation of the i-th coordinate ofexamples with jth coordinate of examples
- To use kernels, need to use the inner-product matrix $X^T X$.

Alternative expression for PCA

- The principal component lies in the span of the data

$$v_1 = \sum_i \alpha_k x_i = X\alpha$$

- Plug this in we have

$$Cv_1 = \frac{1}{n} X X^T X \alpha = \lambda X \alpha$$

- Now, left-multiply the LHS and RHS by X^T.

$$\frac{1}{n} X^T X X^T X \alpha = \lambda X^T X \alpha$$

Only depends on the inner product matrix

Kernel PCA

- **Key Idea**: Replace inner product matrix by kernel matrix
  - PCA: $\frac{1}{n}X^TXX^TX\alpha = \lambda X^TX\alpha$
  - Let $K = [K(x^i, x^j)]_{ij}$ be the matrix of all dot-products in the ϕ-space.
  - Kernel PCA: replace "X^TX" with K.
    $\frac{1}{n}KK\alpha = \lambda K\alpha$,orequivalently, $\frac{1}{n}K\alpha = \lambda\alpha$
- **Key computation**: form an n by n kernel matrix K, and then perform eigen-decomposition on K.

RBF Kernel PCA

Gaussian radial basis function (RBF)

1. Computation of the kernel (similarity) matrix

   Calculate $$\kappa(\mathbf{x_i}, \mathbf{x_j}) = exp\left(-\gamma \left\|\mathbf{x_i} - \mathbf{x_j}\right\|_2^2\right)$$

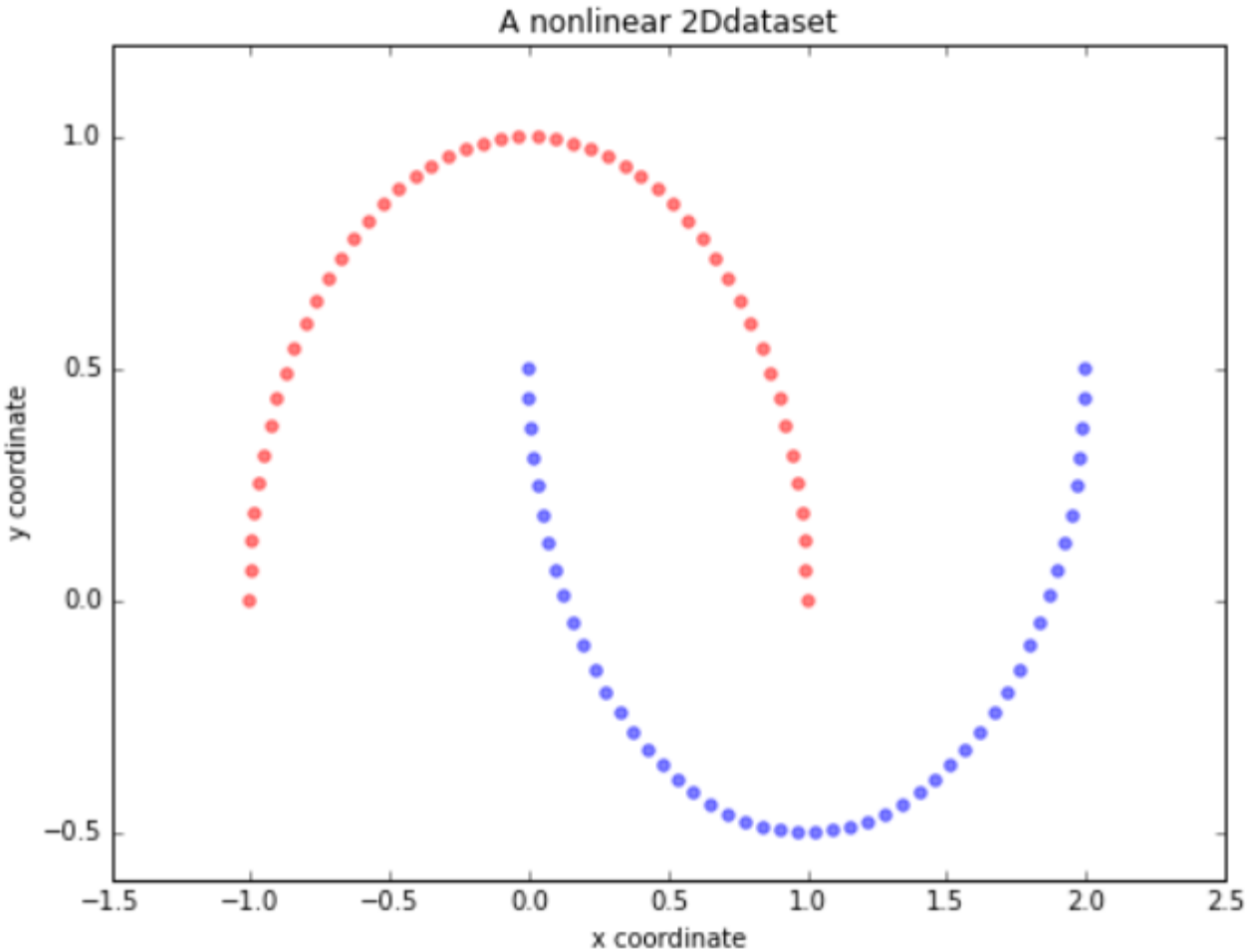
for every pair of points. E.g., if we have a dataset of 100 samples, this step would result in a symmetric 100x100 kernel matrix.

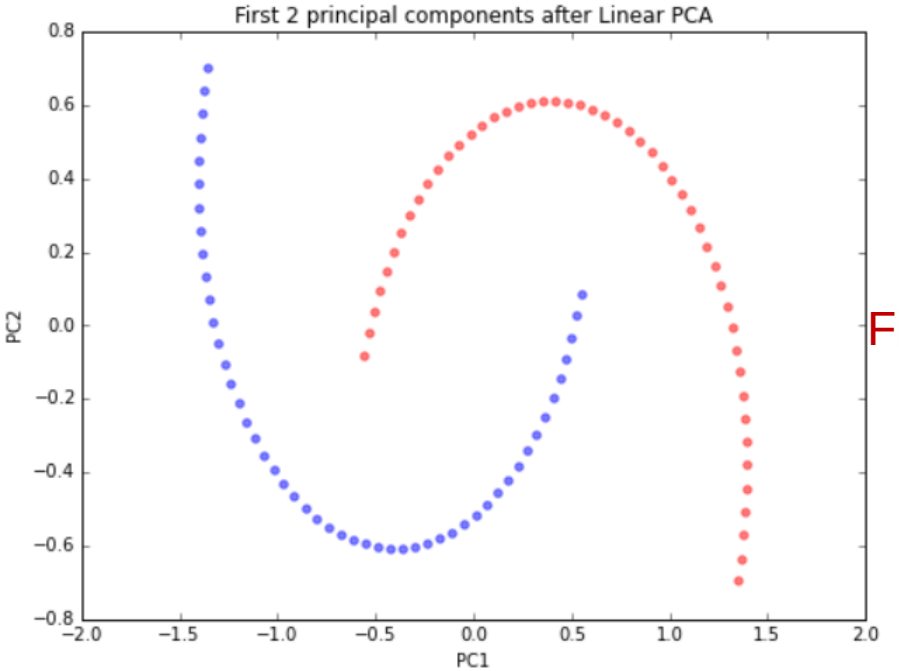
2. Eigendecomposition of the kernel matrix.

Obtain the eigenvectors of the centered kernel matrix that correspond to the largest eigenvalues. Those eigenvectors are the data points already projected onto the respective principal components.

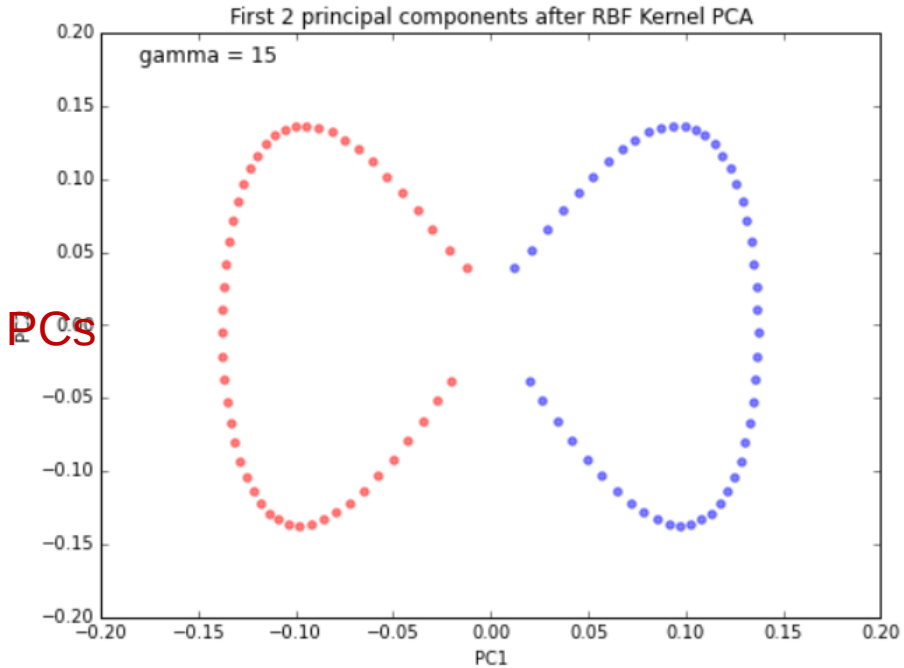
Examples of RBF Kernel PCA

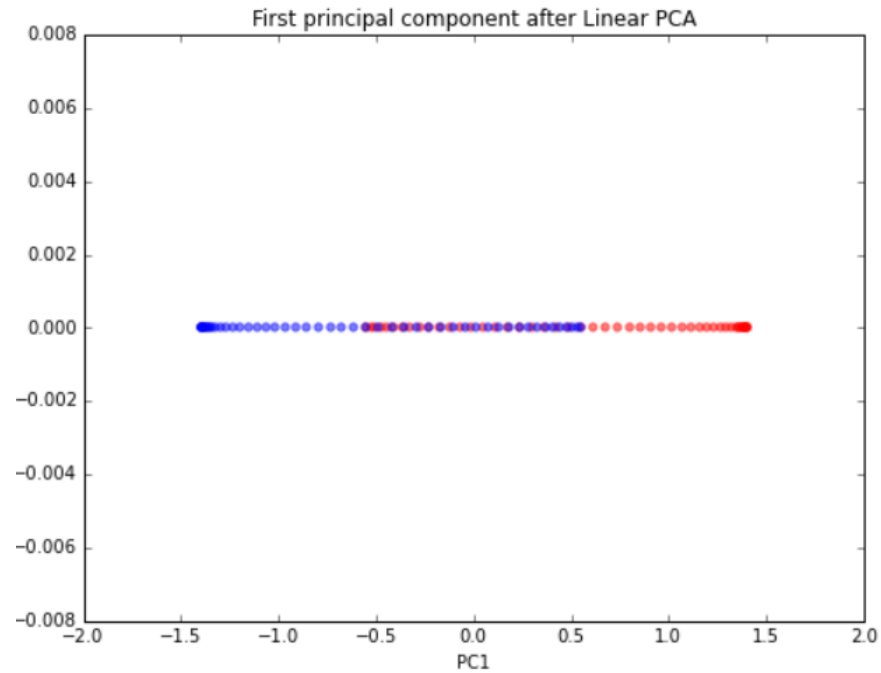
- Half-moon shapes



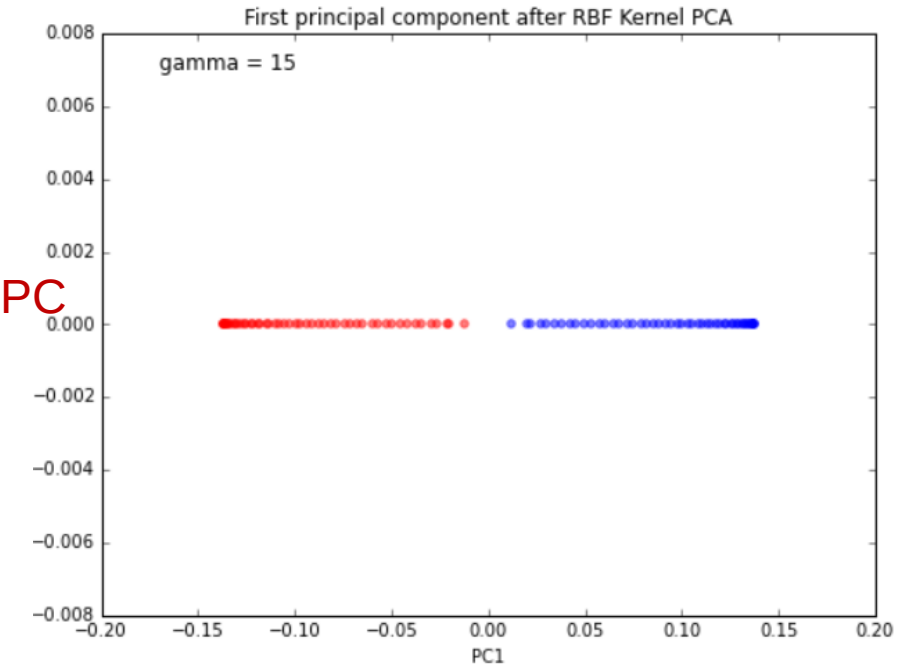


First 2 PCs





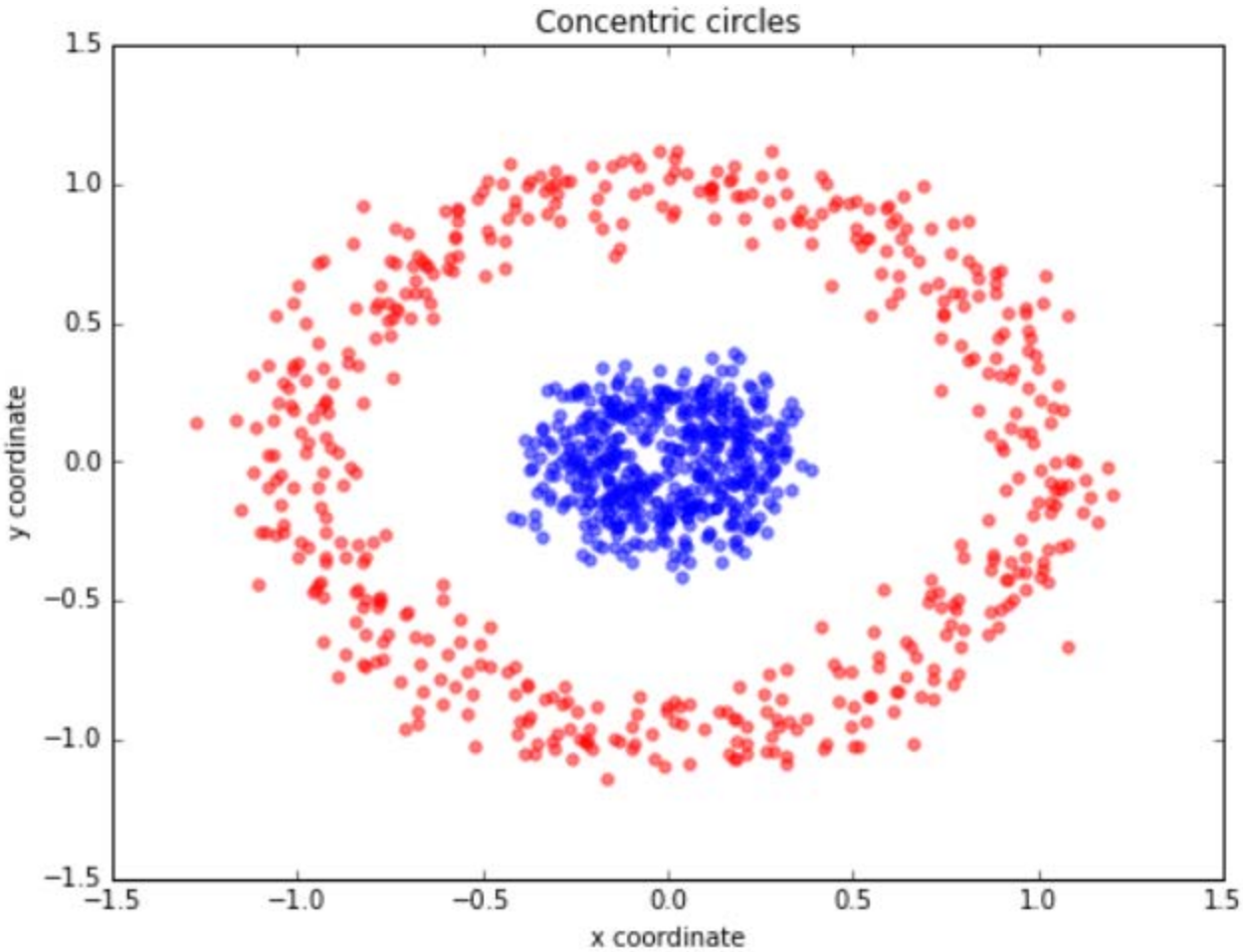
First PC

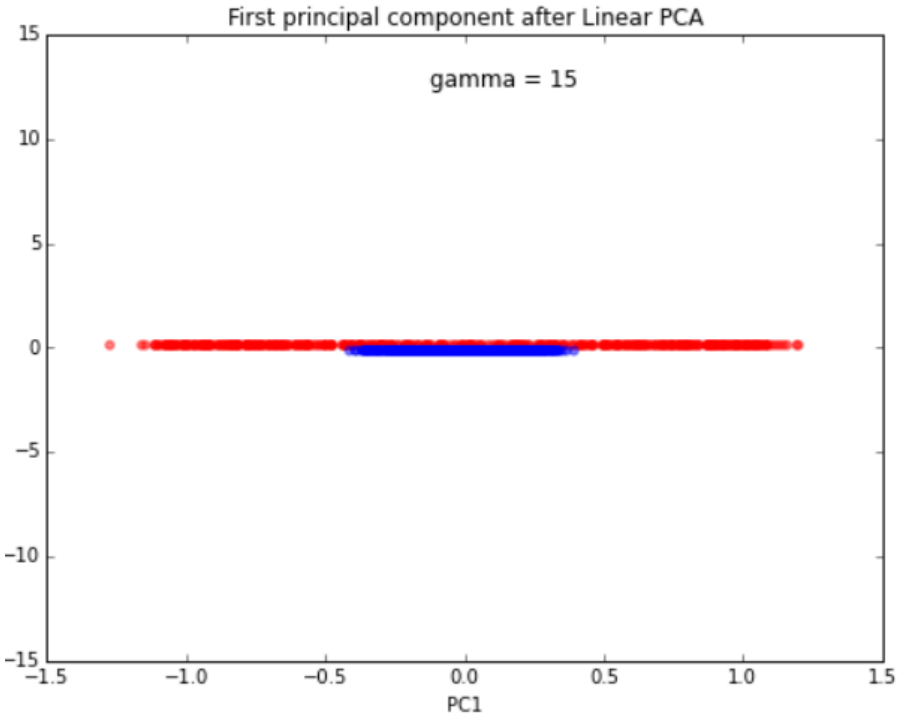


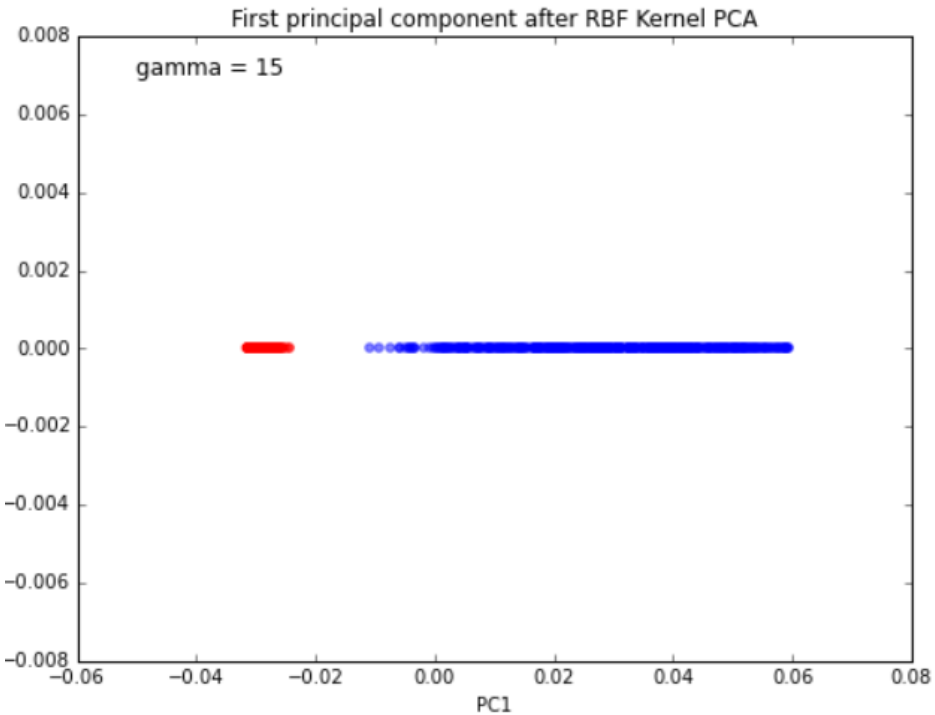
Linear PCA

RBF Kernel PCA

Concentric circles



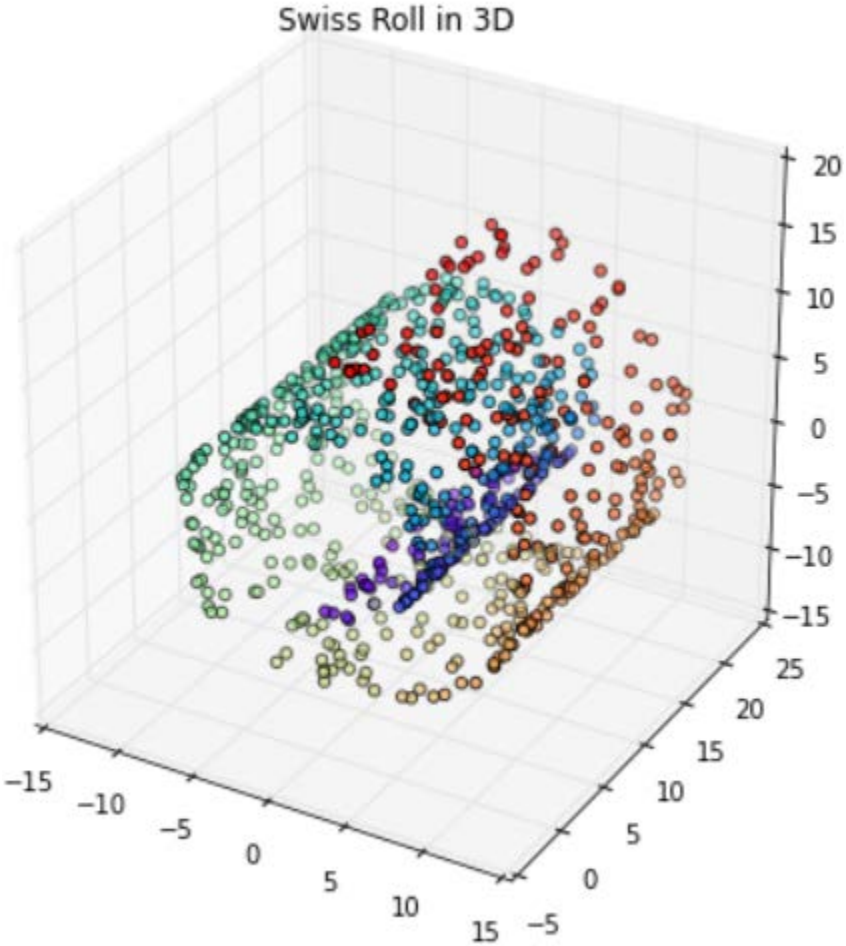




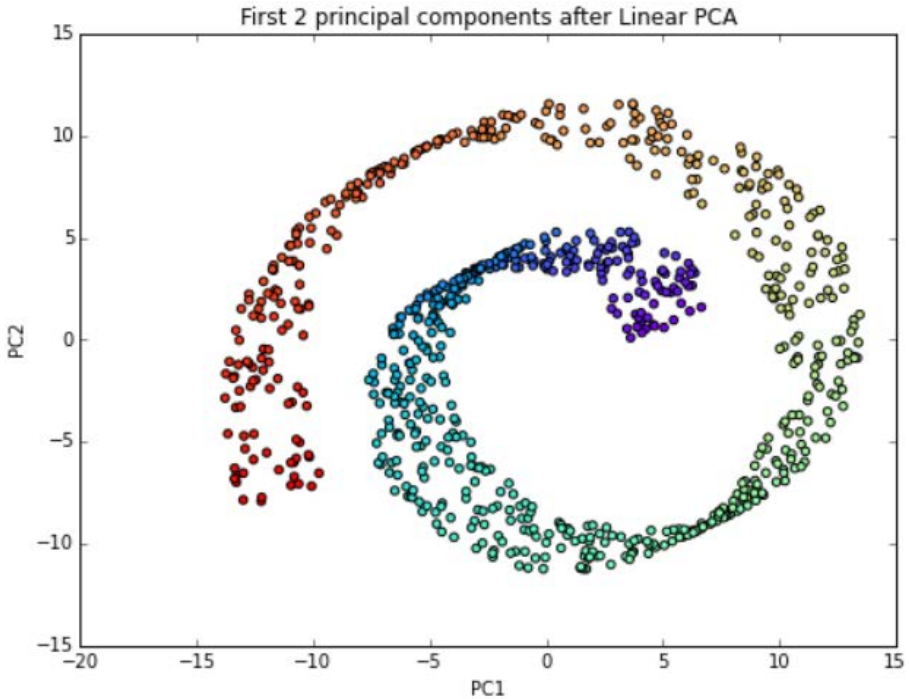
Linear PCA First PC RBF Kernel PCA

- The results obtained via the linear PCA approach does not produce a subspace where the 2 classes are linearly well separate
- This 1-dimensional subspace obtained via Gaussian RBF kernel PCA looks much better in terms of linear class separation

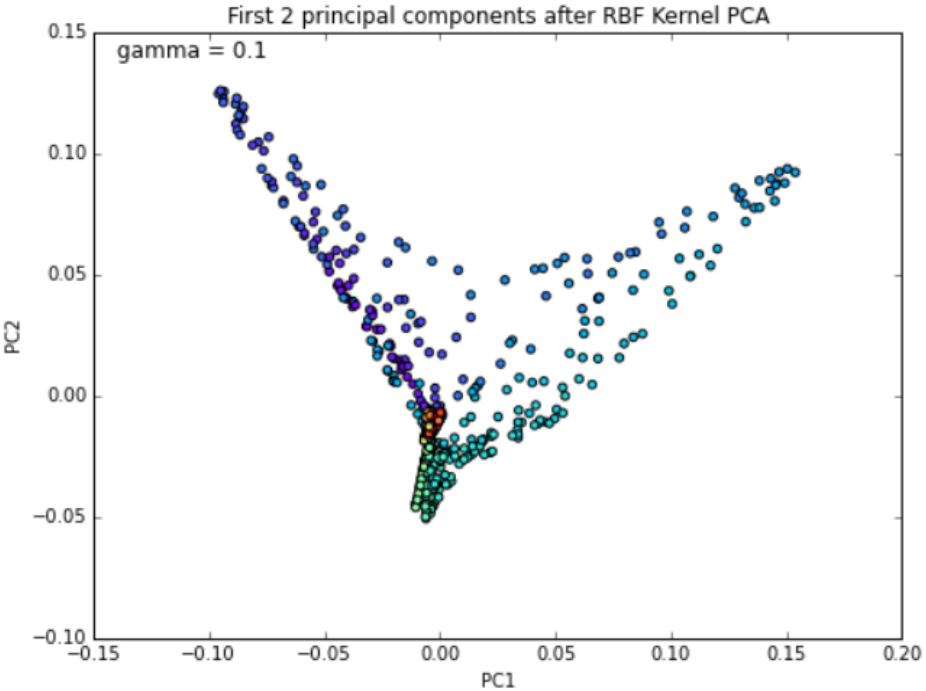
Swiss roll



First 2 PCs



Linear PCA



RBF Kernel PCA

PCA implemented in Python

- scikit-learn
  - PCA
  - KernelPCA

Independent Component Analysis

Overview

- The Problem
- Definition of ICA
- Restrictions
- Ways to solve ICA
  - NonGaussianity
  - Mutual Information
  - Maximum Likelihood
- Fast ICA algorithm
- Simulations
- Conclusion

The Problem

- Cocktail Problem
  - Several Sources
  - Several Sensors
  - Ex: Humans hear mixed signal, but able to unmix signals and concentrate on a sole source
- Recover source signals given only mixtures
  - No prior knowledge of sources or mixing matrix
- aka Blind Source Separation (BSS)

Assumptions

- Source signals are statistically independent
  - Knowing the value of one of the components does not give any information about the others
- ICs have nongaussian distributions
  - Initial distributions unknown
  - At most one Gaussian source
- Recovered sources can be permutated and scaled

Definition of ICA

- Observe N linear mixtures $x_1,\ldots,x_n$ of n independent components

$$x_j = a_{j1}s_1 + a_{j2}s_2 + \ldots + a_{jn}s_n, \text{ for all } j$$

  - a_j is the column of the mixing matrix A
  - Assume each mixture x_j and each IC s_k is a random variable
  - Time difference between mixes dropped
- Independent components are latent variables
  - Cannot be directly observed

Definition of ICA

- ICA Mixture model: $x=As$
  - A is mixing matrix; s is matrix of source signals
- Goal
  - Find some matrix W, so that

$$s = Wx$$

  - W = inverse of A

Definition: Independence

- Two functions independent if

$$E\{h_1(y_1)h_2(y_2)\} = E\{h_1(y_1)\}\, E\{h_2(y_2)\}$$

  - If variables are independent, they are uncorrelated
- Uncorrelated variables
  - Defined: $E\{y_1 y_2\} = E\{y_1\}\, E\{y_2\} = 0$
  - Uncorrelation doesn't equal independence
  - Ex: (0,1),(0,-1),(1,0),(-1,0)
  - $E\{y_1^2 y_2^2\} = 0 \neq ¼ = E\{y_1^2\}\, E\{y_2^2\}$
- ICA has to prove independence

ICA restrictions

- Cannot determine variances
  - s and A are unknown
  - Scalar multipliers on s could be canceled out by a divisor on A
  - Multiplier could even be -1
- Cannot determine order
  - Order of terms can changed.

ICA restrictions

- At most 1 Gaussian source
  - x_1 and x_2 Gaussian, uncorrelated, and unit variance

$$p(x_1, x_2) = \frac{1}{2\pi} \exp\left(-\frac{x_1^2 + x_2^2}{2} \right)$$

  - Density function is completely symmetric
    - Does not contain info on direction of the columns of the mixing matrix A.

ICA estimation

- Nongaussianity estimates independent
  - Estimation of $y = w^T x$
  - let $z = A^T w$, so $y = w^T A s = z^T s$
  - y is a linear combination of s_i, therefore $z^T s$ is more gaussian than any of s_i
  - $z^T s$ becomes least gaussian when it is equal to one of the s_i
  - $w^T x = z^T s$ equals an independent component
- Maximizing nongaussianity of $w^T x$ gives us one of the independent components
  - Maximizing nongaussianity by measuring nongaussiantiy
  - Minimizing mutual information
  - Maximum Likelihood

Measuring nongaussianity

- Kurtosis
  - Fourth order cumulant
  - Classical measure of nongaussianity
  - $\mathrm{kurt}(y) = E\{y^4\} - 3(E\{y^2\})^2$
  - For gaussian y, fourth moment = $3(E\{y^2\})^2$
    - Kurtosis for gaussian random variables is 0
  - Con – not a robust measure of nongaussianity
    - Sensitive to outliers

Measuring nongaussianity

- Entropy (H): degree of information that an observation gives

$$H(Y) = -\sum_i P(Y = a_i) \log P(Y = a_i)$$

  – A Gaussian variable has the largest entropy among all random variables of equal variance

- Negentropy J
  – Based on the information theoretic quantity of differential entropy

$$J(Y) = H\left(y_{gauss}\right) - H\left(y\right)$$

  – Computationally difficult

Negentropy approximations

- Classical method using higher-order moments

$$J(y) = \frac{1}{12} E\left\{y^3\right\}^2 + \frac{1}{48} kurt(y)^2$$

  – Validity is limited to nonrobustness of kurtosis

Negentropy approximations

- Hyvärinen 1998b: maximum-entropy principle

$$J(y) \propto \left[E\{G(y)\} - E\{G(v)\}\right]^2$$

  - G is some contrast function
  - v is a Gaussian variable of zero mean and unit variance
  - Taking $G(y) = y^4$ makes the equation kurtosis based approximation

Negentropy approximations

- Instead of kurtosis function, choose a contrast function G that doesn't grow too fast

$$G_1(u) = \frac{1}{a_1} \log \cosh a_1 u, \; G_2(u) = -\exp\left(-u^2/2\right)$$

Where $1 \le a_1 \le 2$

Minimizing mutual information

- Mutual information I is defined as

$$I(y_1, y_2, \ldots, y_m) = \sum_{i=1}^{m} H(y_i) - H(y)$$

  - Measure of the dependence between random variables
  - I = 0 if variables are statistically independent
  - Equivalent to maximizing negentropy

Maximum Likelihood Estimation

- Closely related to infomax principle
- Infomax (Bell and Sejnowski, 1995)
  - Maximizing the output entropy of a neural network with non-linear outputs
  - Densities of ICs must be estimated properly
  - If estimation is wrong ML will give wrong results

Fast ICA

- Preprocessing
- Fast ICA algorithm
  - Maximize non gaussianity
- Unmixing signals

Fast ICA: Preprocessing

- Centering
  - Subtract its mean vector to make x a zero-mean variable
  - ICA algorithm does not need to estimate the mean
  - Estimate mean vector of s by $A^{-1}m$, where m is the mean the subtracted mean

Fast ICA: Preprocessing

- Whitening
  - Transform x so that its components are uncorrelated and their variances equal unity

$$E\left\{\tilde{x}\tilde{x}^T\right\} = I$$

the covariance matrix of $\tilde{x}$ equals the identity matrix

  - Use eigen-value decomposition (EVD) of the covariance matrix E

$$\tilde{x} = ED^{-1/2}E^T x$$

    - D is the diagonal matrix of its eigenvalues
    - E is the orthogonal matrix of eigenvectors

Fast ICA: Preprocessing

- Whitening
  - Transforms the mixing matrix into Ã.

$$\tilde{\boldsymbol{x}} = \boldsymbol{E}\boldsymbol{D}^{-1/2}\boldsymbol{E}^{\boldsymbol{T}}\boldsymbol{A}\boldsymbol{s} = \tilde{\boldsymbol{A}}\boldsymbol{s}$$

  - Makes Ã orthogonal
    - Lessons the amount of parameters that have to be estimated from n^2 to n(n-1)/2
    - In large dimensions an orthogonal matrix contains approximately ½ the number of parameters

Fast ICA Algorithm

- One-unit (component) version

1. Choose an initial weight vector w.
2. Let $w^+ = E\{xg(w^T x)\} - E\{g'(w^T x)\}w$
   - Derivatives of contrast functions G

     $g_1(u) = \tanh(a_1 u)$,

     $g_2(u) = u \exp(-u^2/2)$
3. $w = w^+/||w^+||$. (Normalization step)
4. If not converged go back to 2

   -converged if $\text{norm}(w_{new} - w_{old}) > \xi$ or $\text{norm}(w_{old} - w_{new}) > \xi$

   - ξ typically around 0.0001

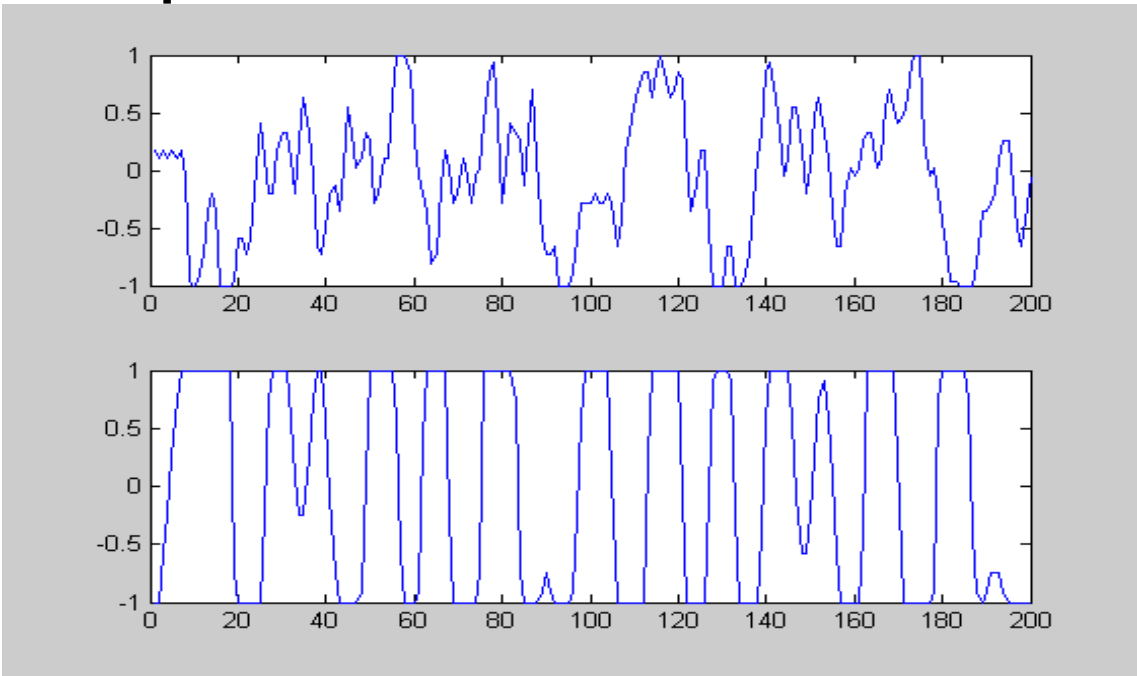
Fast ICA Algorithm

- Several unit algorithm
  - Define B as mixing matrix and B′ as a matrix whose columns are the previously found columns of B
  - Add projection step before step 3
  - Step 3 becomes

3. Let $w(k) = w(k) - B'B'^{T}w(k)$. $w = w^{+}/||w^{+}||$

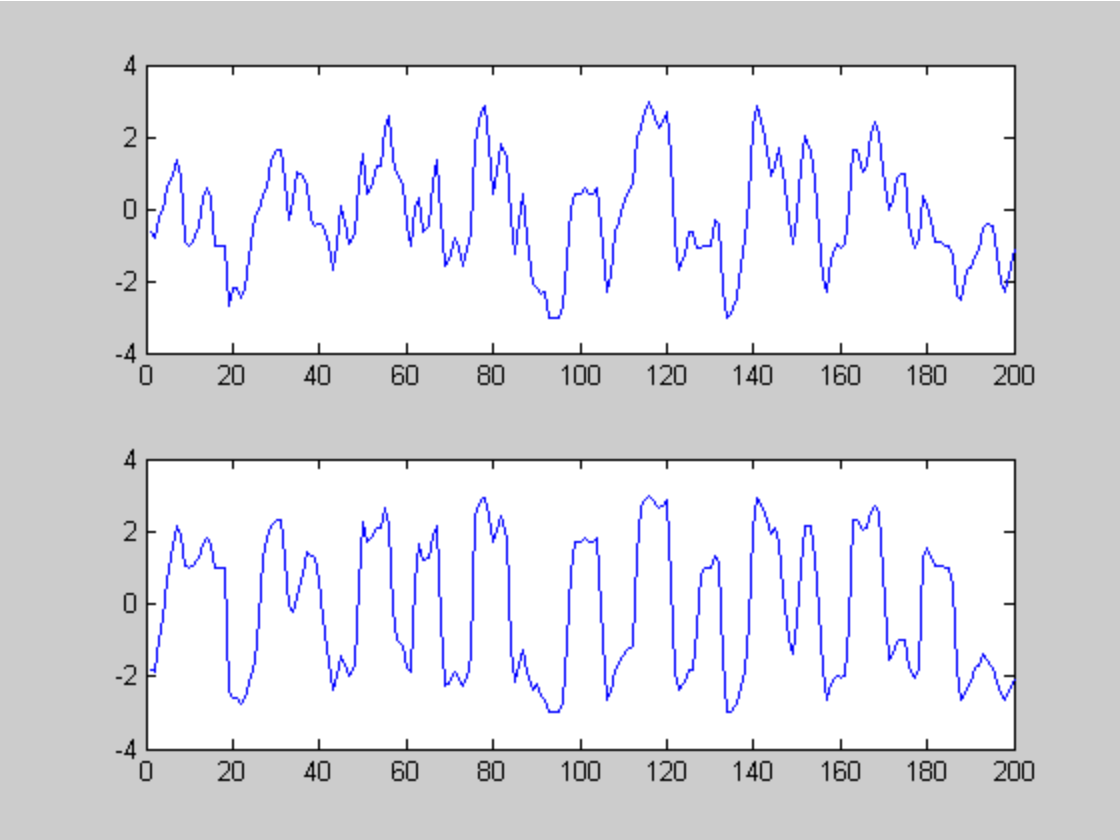
Simple Simulation

- Separation of 2 components
- Figure 1: Two independent non gaussian wav samples



Simple Simulation

- Figure 2: Mixed signals



Simple Simulation

- Recovered signals vs original signals

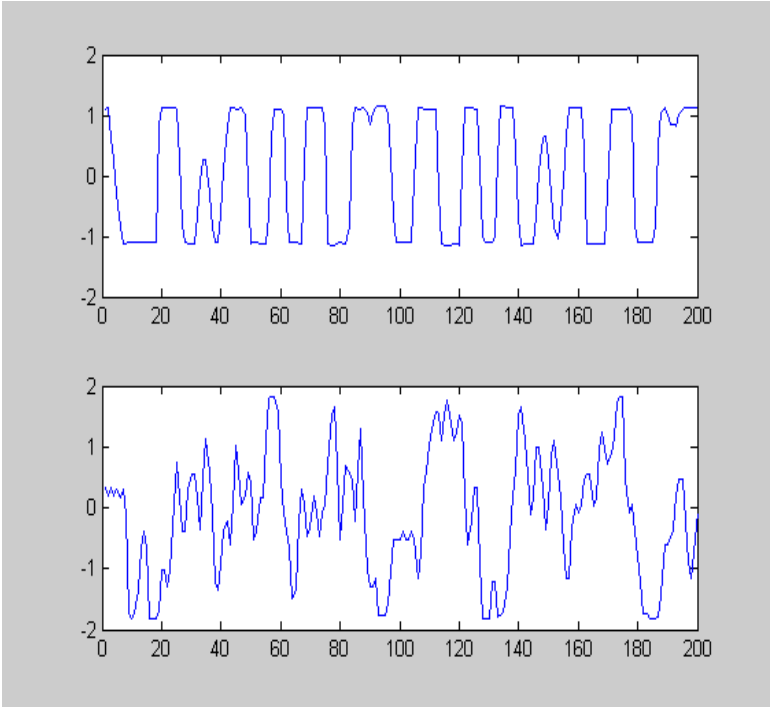


Figure 3: Recovered signals

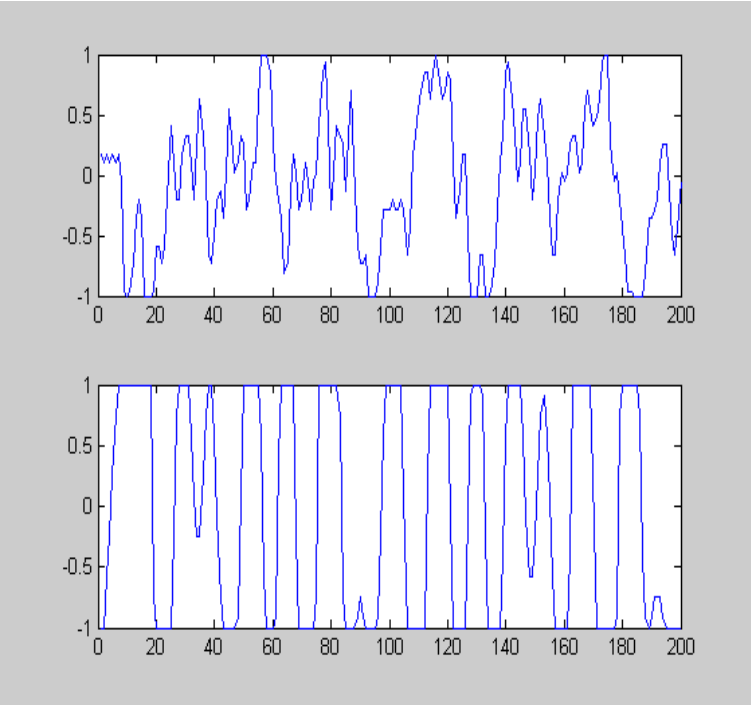


Figure 4: Original signals

Simulation Results

- IC 1 recovered in 6 steps and IC 2 recovered in 2 steps
- Retested with 20000 samples
  - Requires approximately the same number of steps

Gaussian Simulation

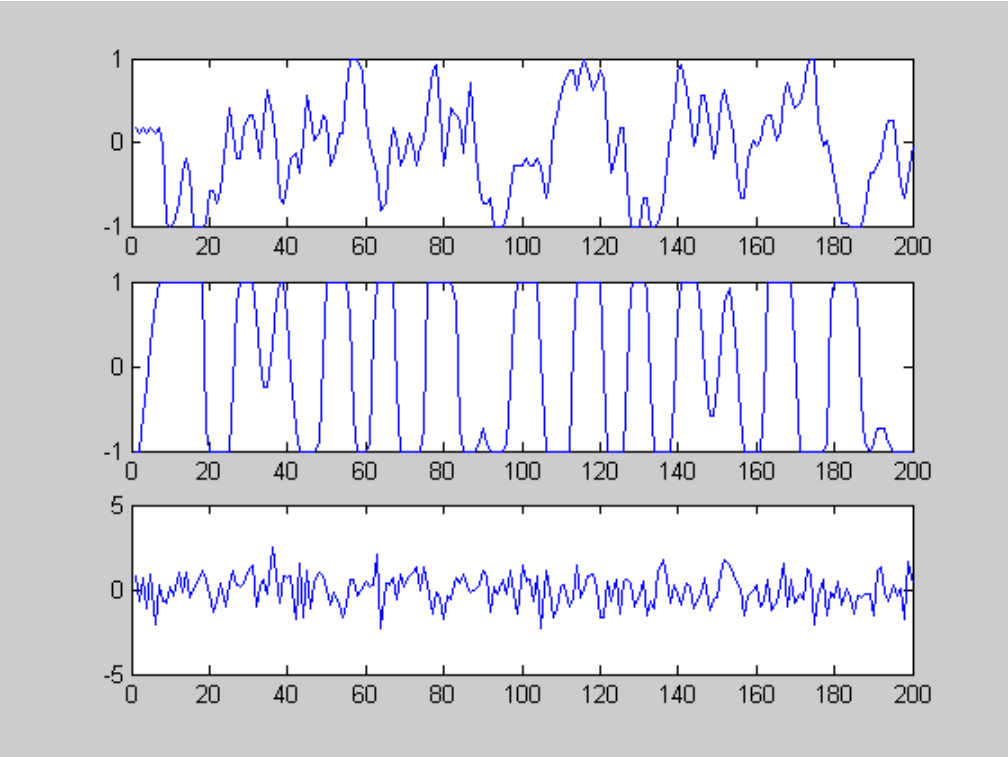


Figure 5: 2 wav samples and noise signal

Gaussian Simulation

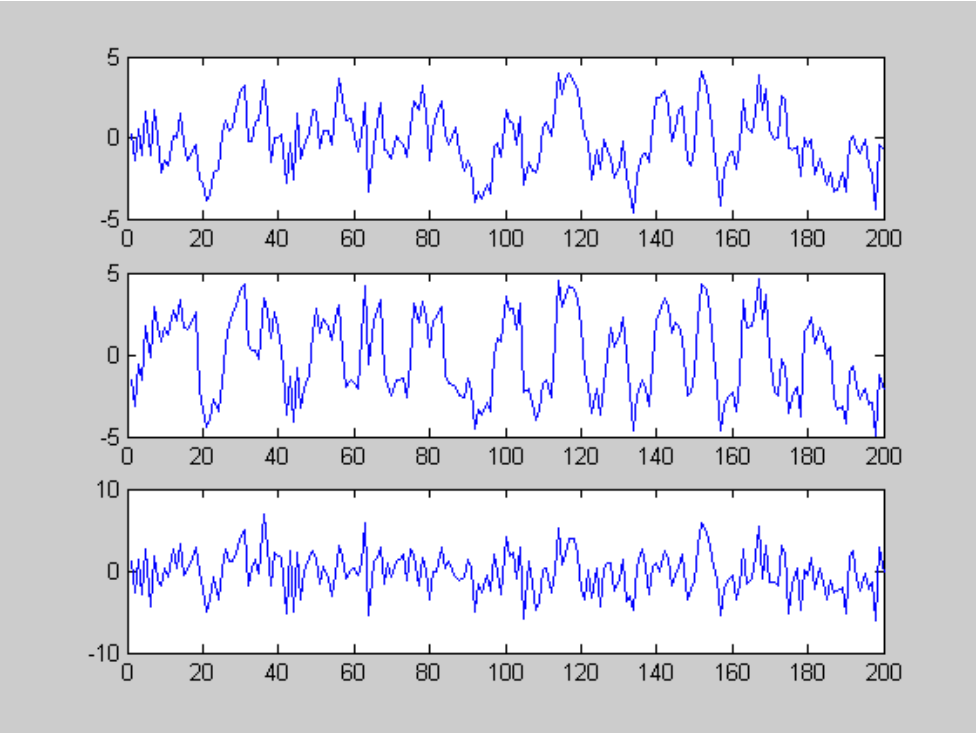


Figure 6: 3 mixed signals

Gaussian Simulation

- Comparison of recovered signals vs original signals

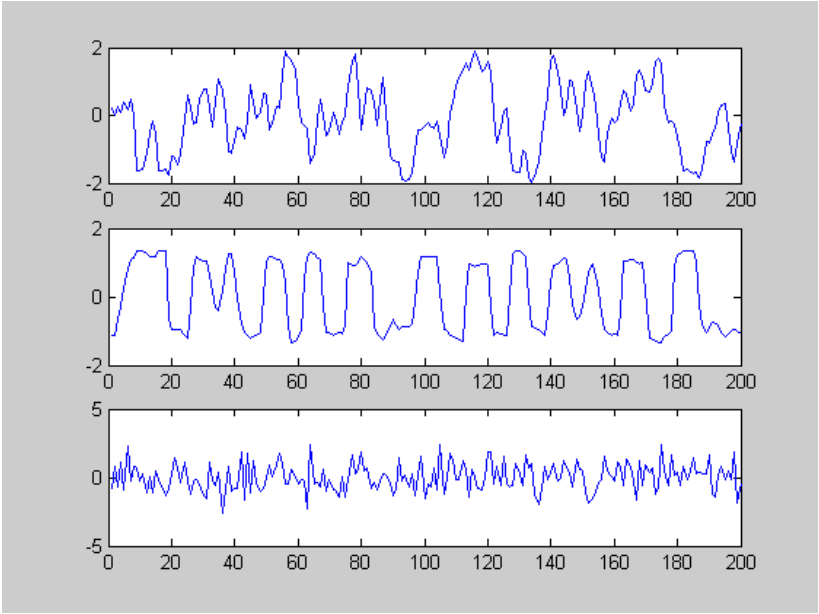


Figure 7: Recovered signals

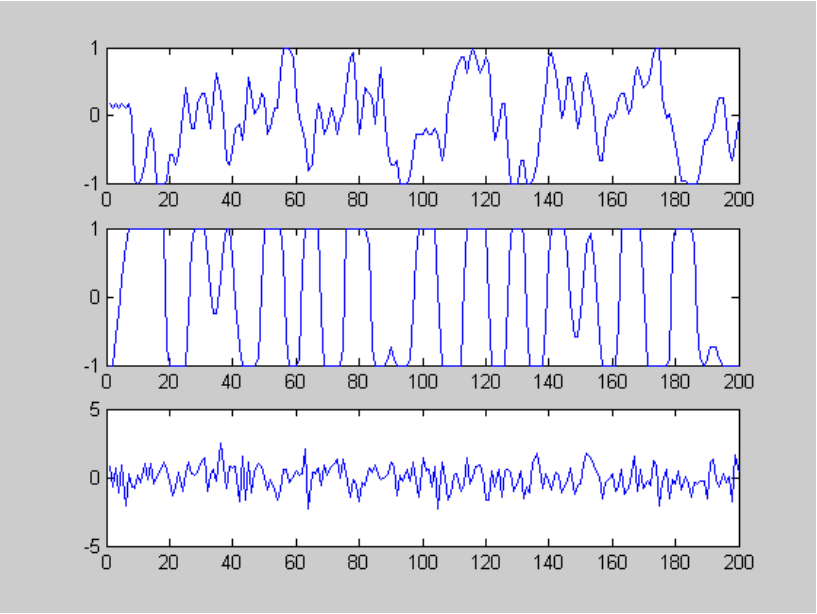


Figure 8: Original signal

Gaussian Simulation 2:

- Tried with 2 gaussian components

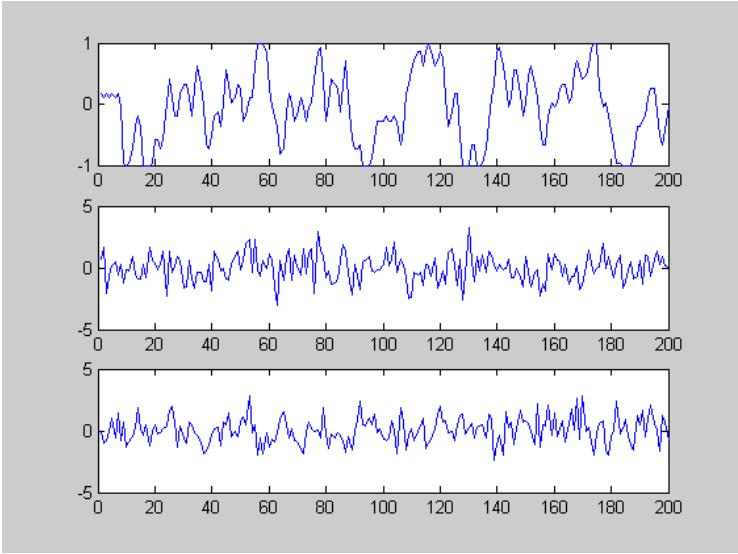


Figure 10: Original signals

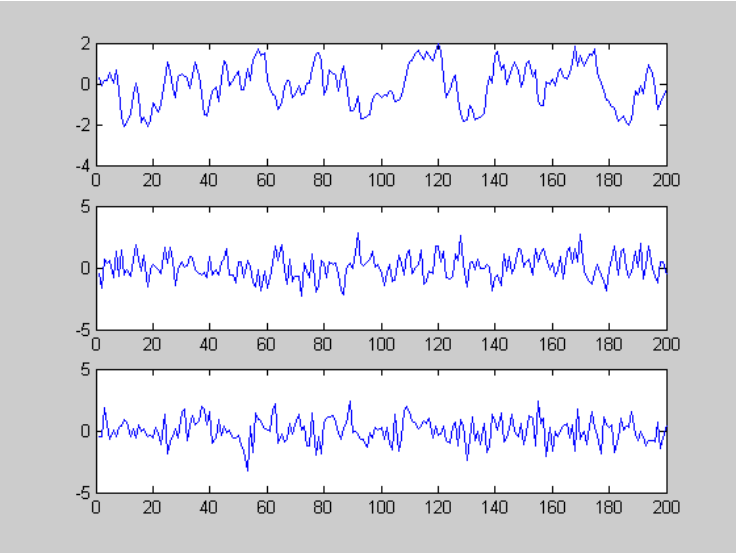


Figure 11: Recovered signals

  – Components were not estimated properly due to more than one Gaussian component

Conclusion

- Fast ICA properties
  - No step size, unlike gradient based ICA algorithms
  - Finds any non-Gaussian distribution using any non linear g contrast function.
  - Components can be estimated one by one
- Other Applications
  - Separation of Artifacts in image data
  - Find hidden factors in financial data
  - Reduce noise in natural images
  - Medical signal processing – fMRI, ECG, EEG (Mackeig)

References

- [1] Aapo Hyvärinen and Erkki Oja, Independent Component Analysis: Algorithms and Applications. Neural Networks Research Centre Helsinki University of Technology; *Neural Networks,* 13 (4-5): 411-430, 2000
- [2] Aapo Hyvärinen and Erkki Oja, A Fast Fixed-Point Algorithm for Independent Component Analysis. Helsinki University of Technology Laboratory of Computer and Information Science, Neural Computation, 9:1483:1492, 1997
- [3] Anthony J. Bell and Terrence J. Sejnowski, The ‘Independent Components’ of Natural Scenes are Edeg Filters. Howard Hughes Medical Institute Computational Neurobiology Laboratory
- [4] Te-Won Lee, Mark Girolami, Terrence J. Sejnowski, Independent Component Analysis Using and Extended Infomax Algorithm for Mixed Subgaussian and Supergaussian Sources. 1997
- [5] Antti Leino, Independent Component Analysis An Overview. 2004
- [6] Erik G. Learned-Miller, John W. Fisher III, ICA Using Spacings Estimates of Entropy Journal of Machine Learning Research 4 (2003) 1271-1295.