Principal Component Analysis (PCA)

and

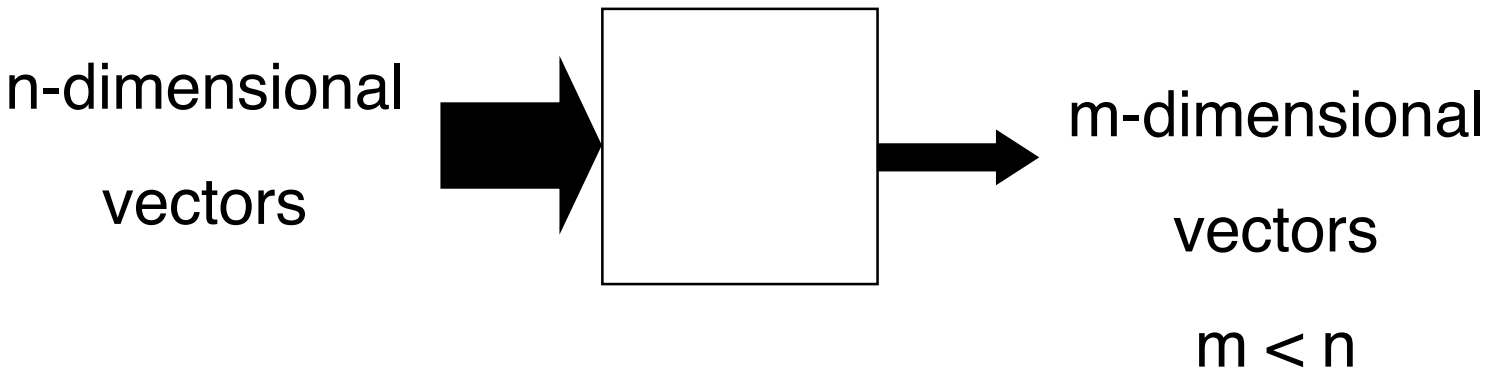
PCA Neural Networks: An Application of Unsupervised Hebbian Learning

with brief mention of Independent Components Analysis (ICA)

What is PCA?

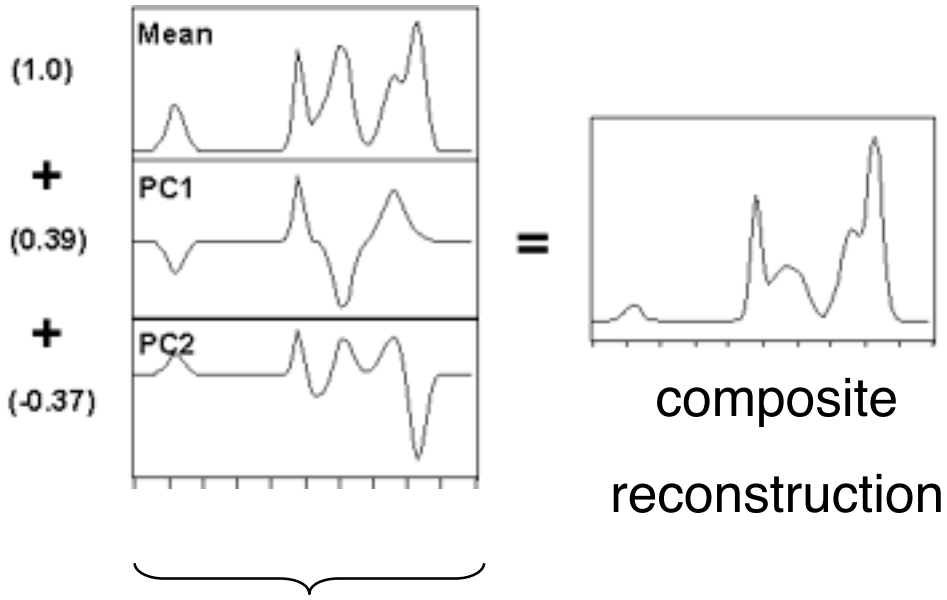
- A standard statistical technique for reducing dimensionality of data (without using neural approach)
- Purpose: Better understanding or communication of data; used a lot in the sciences to select the most important features
- In so reducing, we want to lose as little information as possible, given the before- and after- dimensions.
- Also known as Karhuenen-Loeve (K-L) transformation (Watanabe, 1969)

PCA



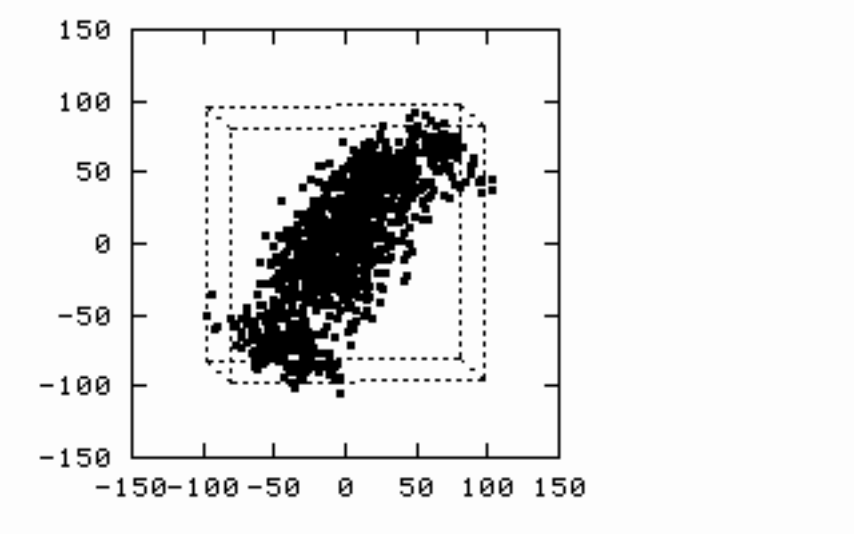
Principal Components Example

(http://www.galactic.com/algorithms/pca.htm)



Scientific Uses

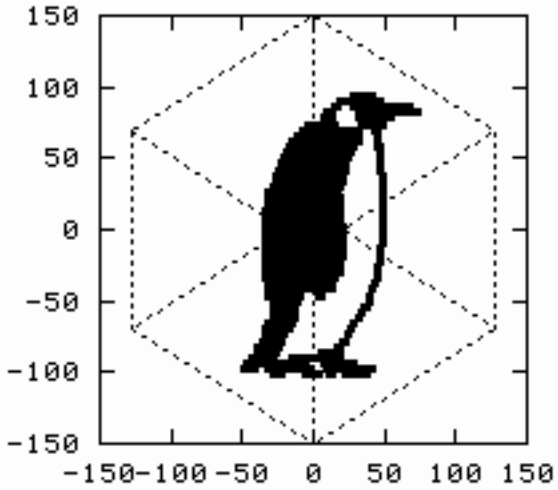
(cf. http://144.124.112.51/auj/scattering/demo/page4.active.html)



Transform coordinates to get a better understanding of the underlying phenomenon.

Scientific Uses

(cf. http://144.124.112.51/auj/scattering/demo/page4.active.html)



Transform coordinates to get a better understanding of the underlying phenomenon.

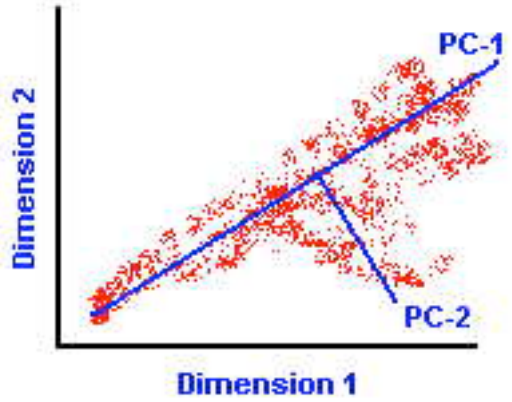
Comparison with Linear Regression

- Linear regression requires one to ***pre-identify*** dependent vs. independent variables.

- PCA does not.

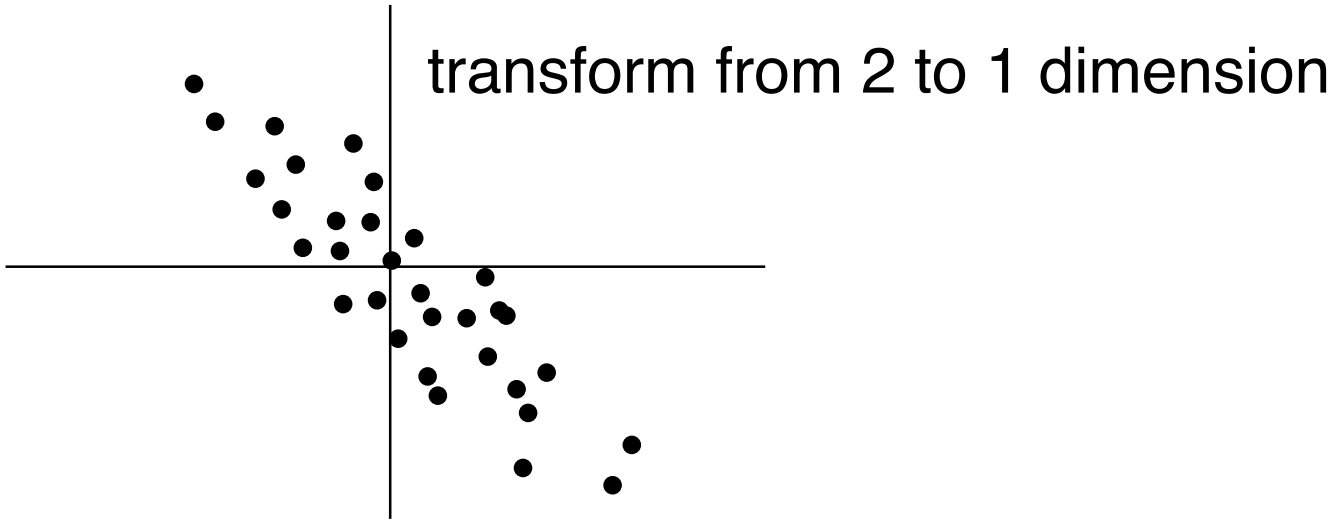
What is a “Principal Component”?

- If we were to plot the data, the **first principal component** would be the **predominant direction** in the data.
- The second principal component would be the second-most predominant direction, etc. up to the number of **dimensions** of the data points.



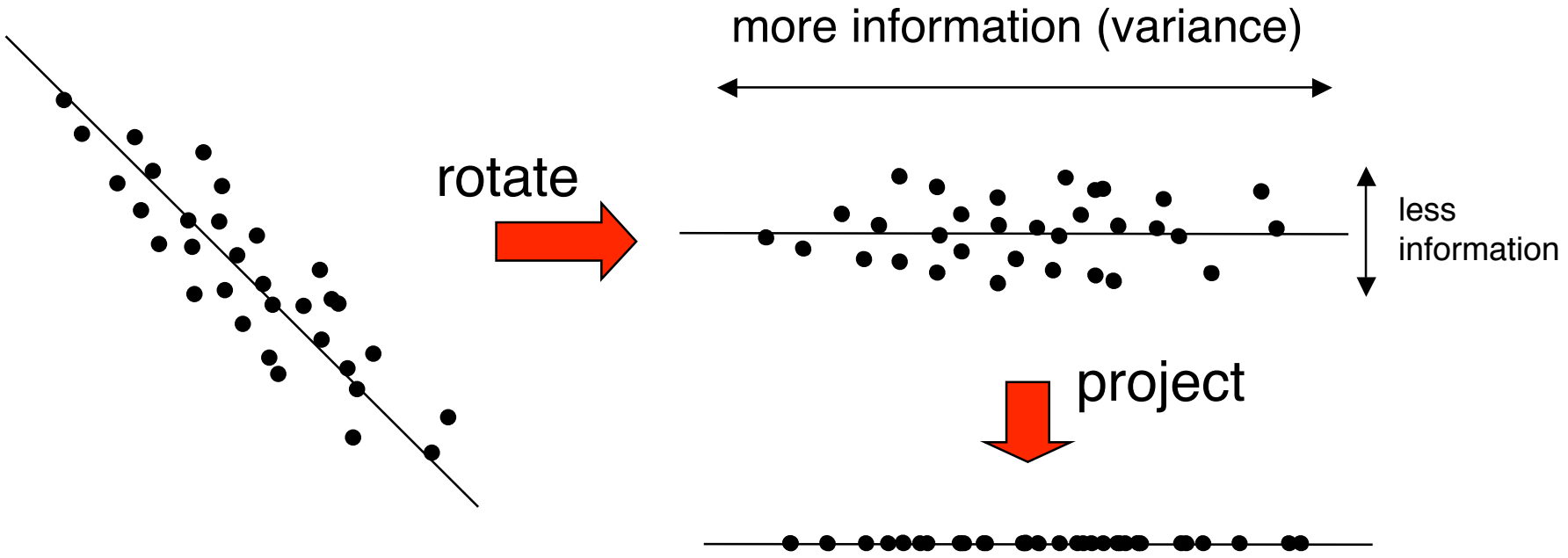
The main idea

- Transform the input data into fewer dimensions
- Preserve as much of the variance as possible



Transformation

- Preserve as much of the variance as possible



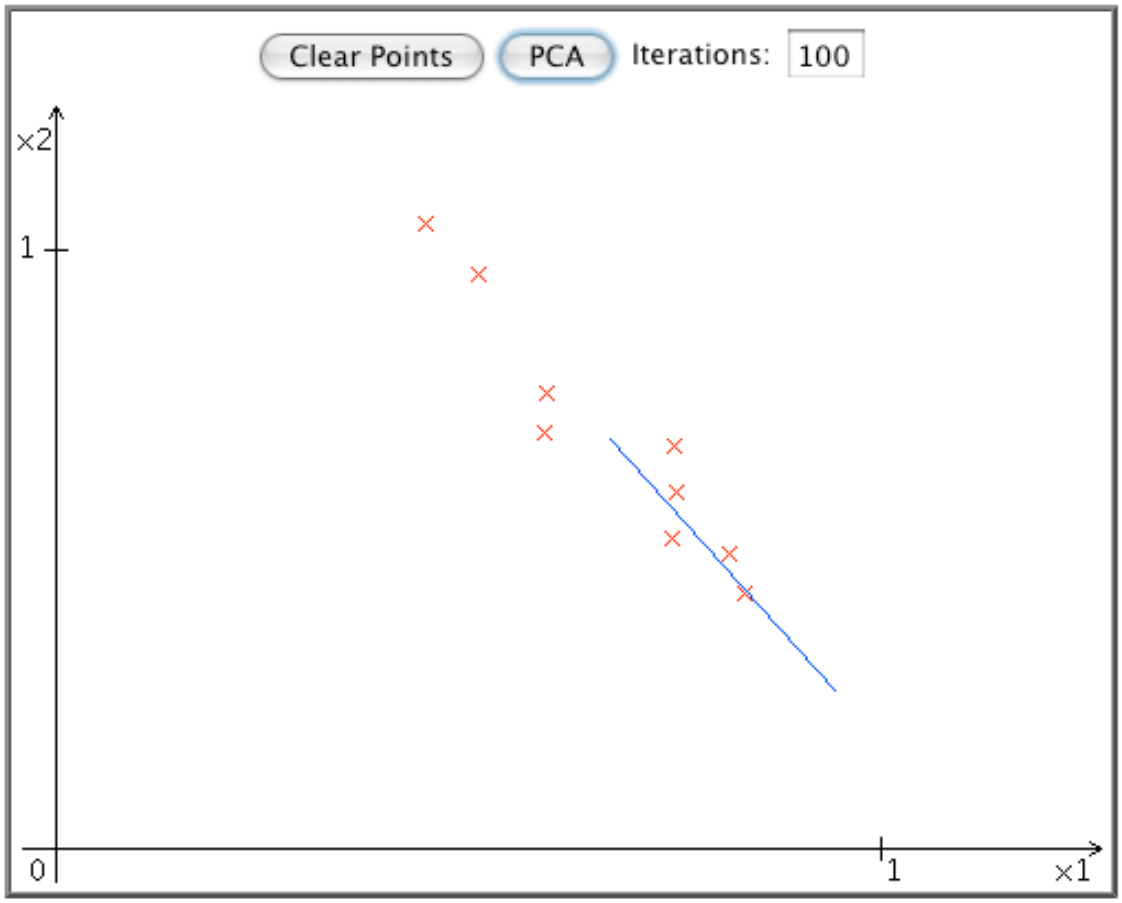
What is a Predominant Direction?

- This may seem subjective, but a mathematical definition can be provided.

- Let A be the matrix of data points:
  - Each point or observation is a column.
  - The rows correspond to the observed values of variables.

First P.C. Demo, you pick the points

http://diwww.epfl.ch/mantra/tutorial/english/pca/html/index.html



Linear Transformation

- The types of transformations shown are linear:
  - B = WA
    where
    - A is the matrix of data (each point as a column vector)
    - W is an m x n matrix
    - B is the transformed data matrix
- For analysis purposes, we want the columns of W to be orthonormal.
- Then $WW^T = I$ and the reconstruction of A from B is given by:
  - $A' = W^TB = W^TWX$.

Computing Principal Components

- PCA tries to minimize the expectation of reconstruction error:

  $E(\| A - A' \|^2) =$
  $E(\| A - A' \|^2) =$
  $E(tr((A - A')(A-A')^T)) =$
  $tr(E(AA')) - tr(WAA'W^T) =$
  $tr(R) - tr(WRW^T)$, where

  R is the covariance matrix E(AA')
- E is the expectation, i.e. average over the data points
- tr is the trace (sum of diagonal elements).

Computing Principal Components

- Want W such that tr(R) - tr(WRW^T) is **minimized**.
- For W orthonormal, W^T is also the **pseudo–inverse**, encountered in least-squares minimization.
- The eigenvectors of the covariance matrix R, in order of the largest to smallest eigenvalue, are the **principal components**.
- Construct matrix W as using the top so-many eigenvalues as rows.

Maximimzing Variance

- It turns out that minimizing the reconstruction error is equivalent to maximizing the variance among all possible transformations of a given dimension.

Example

- Data set:
  - (1.3, 3.2, 3.7)
  - (1.4, 2.8, 4.1)
  - (1.5, 3.1, 4.6)
  - (1.2, 2.9, 4.8)
  - (1.1, 3.0, 4.8)
- Component means:
  - (1.3, 3.0, 4.4)
- Mean-adjusted data set A:
  - (0, 0.2, -0.7)
  - (0.1, -0.2, -0.3)
  - (0.2, 0.1, 0.2)
  - (-0.1, -0.1, 0.4)
  - (-0.2, 0.0, 0.4)

Example (2)

- Covariance Matrix (A has 3 rows)
- S =

$$\begin{pmatrix} 0.0250 & 0.0025 & -0.0275 \\ 0.0025 & 0.0250 & -0.0250 \\ -0.0275 & -0.0250 & 0.2350 \end{pmatrix}$$

- Eigenvalues are $\gamma_1 = 0.2415$, $\gamma_2 = 0.0225$, $\gamma_3 = 0.0210$

Example (3)

- $\gamma_1 / (\gamma_1 + \gamma_2 + \gamma_3) = 0.847$, meaning that the first component alone captures 84.7% of the variance
- Eigenvectors corresponding to the two largest eigenvalues are:
  - (0.8232, 0.5419, 0.1691)
  - (0.5534, -0.8325, -0.0263)
- These two vectors would be the rows of the matrix mapping the training set into two dimensions.

Matlab cov Function

```
A = [0, 0.2, -0.7; 0.1, -0.2, -0.3; 0.2,  0.1,  0.2;
    -0.1, -0.1,  0.4; -0.2,  0.0,  0.4]

A =

         0    0.2000   -0.7000
    0.1000   -0.2000   -0.3000
    0.2000    0.1000    0.2000
   -0.1000   -0.1000    0.4000
   -0.2000         0    0.4000

>> R = cov(A)

R =

    0.0250    0.0025   -0.0275
    0.0025    0.0250   -0.0250
   -0.0275   -0.0250    0.2350
```

Matlab: pcacov function

```
>> [pc, latent, explained] = pcacov(R)

pc =

   -0.1265    0.5534    0.8232
   -0.1153   -0.8325    0.5419
    0.9852   -0.0263    0.1691

latent =

    0.2415
    0.0225
    0.0210

explained =

   84.7212
    7.9114
    7.3674
```

Corresponding eigenvectors by column (reversed?)

Eigenvalues

% Variance explained

Turtle Application
(D. Morrison, via Diamantaras & Kung)

- Characterize turtle shells based upon measured length(L), width(W), and height(H) (in mm).
- Covariance matrix for 24 female turtles:

	L	W	H
L	451.33	271.17	168.7
W	271.17	171.73	103.29
H	168.7	103.29	66.65

- Principal Components:

	1	2	3
L	0.8126	-0.5454	-0.2054
W	0.4955	0.8321	-0.2491
H	0.3068	0.1006	0.9465
variance	680.4	6.5	2.9

Turtle Application (D. Morrison)

- Principal Components:

	1	2	3
L	0.8126	-0.5454	-0.2054
W	0.4955	0.8321	-0.2491
H	0.3068	0.1006	0.9465
variance	680.4	6.5	2.9

- The first component accounts for 98.6% of the variance. So the following measure can be used:

$$Y_1 = 0.81L + 0.5W + 0.31H$$

Planets Example

(with the help of http://www.cs.mcgill.ca/~sqrt/dimr/dimreduction.html)

- Consider a 3-dimensional data set where the variables are the logarithms of
  - distance to the sun
  - equatorial diameter
  - density

Data set (prior to taking logs)

planet	distance	diameter	density	
Mercury	0.387	4878	5.42	
Venus	0.723	12104	5.25	
Earth	1.000	12756	5.52	
Mars	1.524	6787	3.94	
Jupiter	5.203	142800	1.314	
Saturn	9.539	120660	0.69	
Uranus	19.18	51118	1.29	
Neptune	30.06	49528	1.64	
Pluto	39.53	2300	2.03	

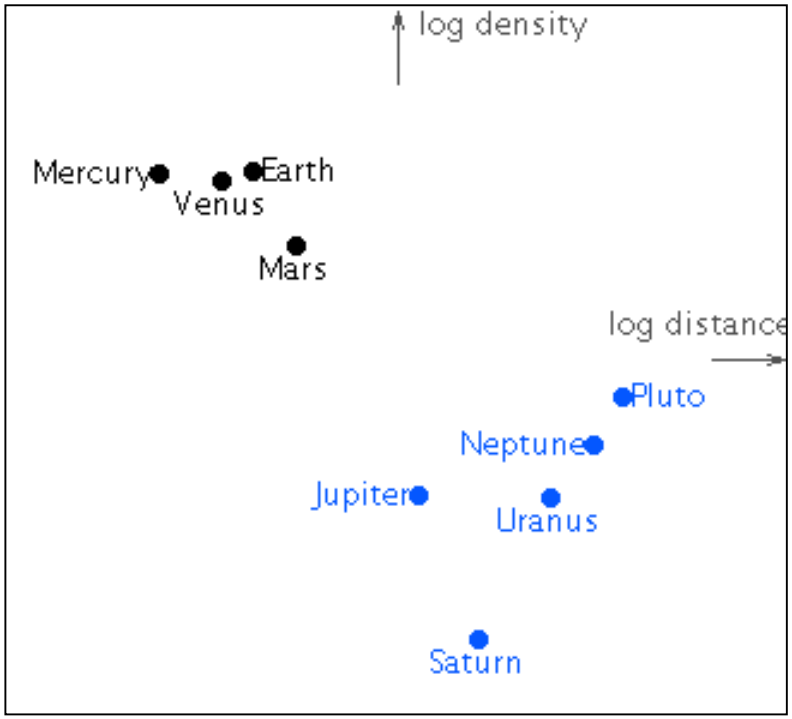
Projecting Data

- Obviously we can **project** a set of data points on fewer dimensions by **ignoring** certain columns.

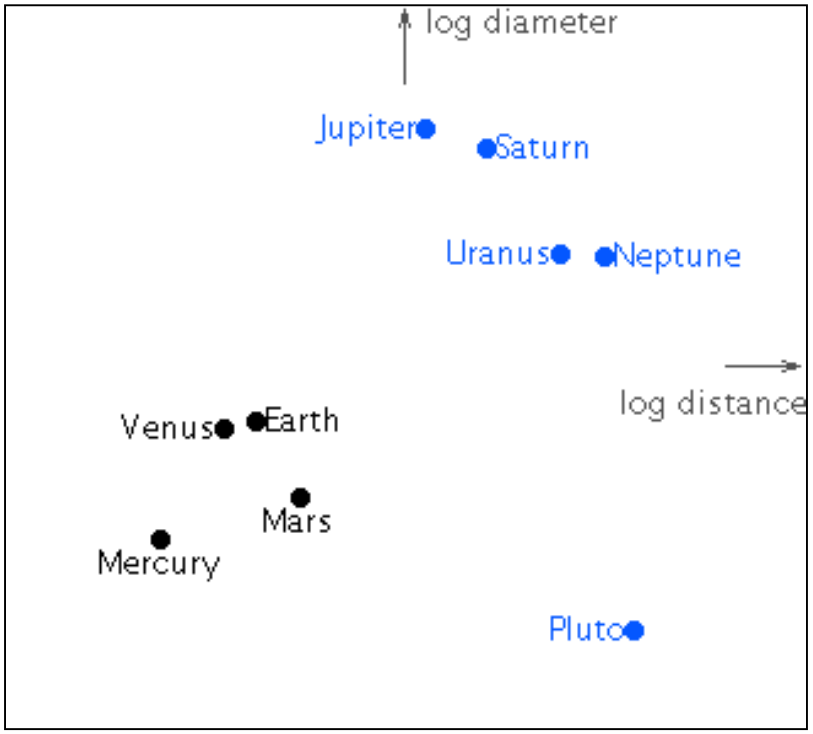
- Projection is a special case of a linear transformation.

Data Projections

- In two dimensions, we can plot any one variable against another, e.g.

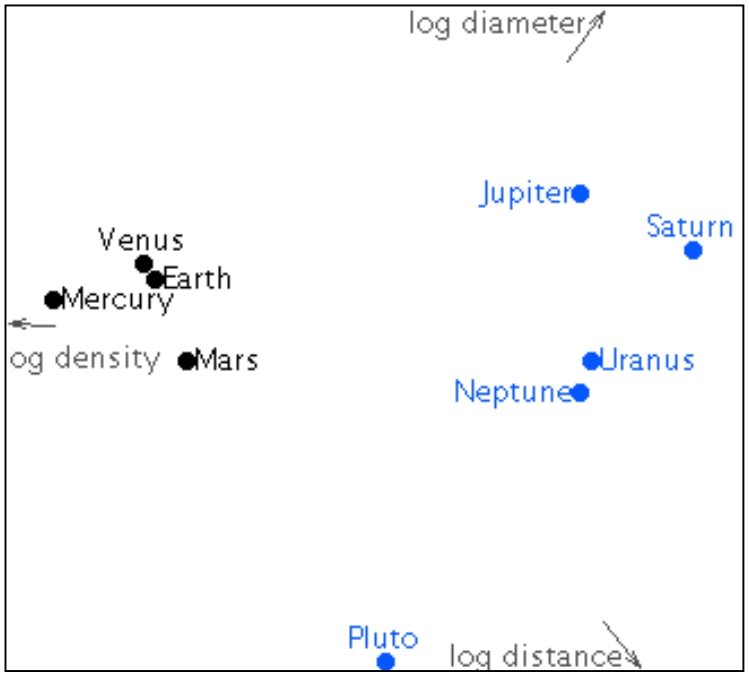


or



Maximizing Variance

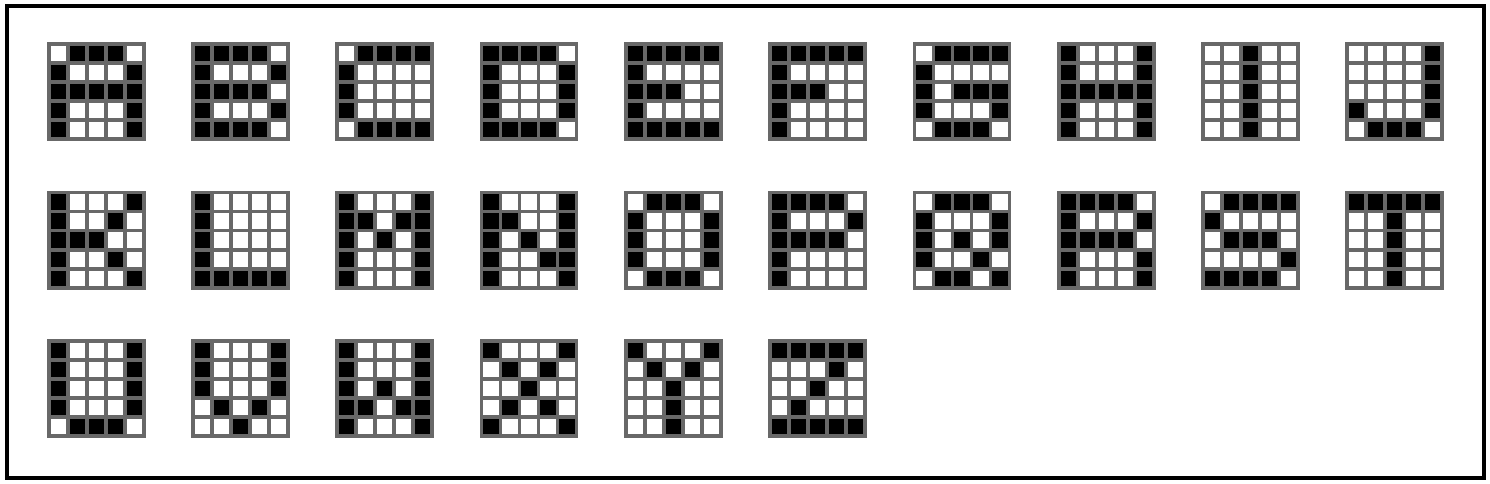
- Transforming using first two principal components preserves more of the variance (summing variances in each dimension) in the projection than does projecting on any 2 variables:



Another Example

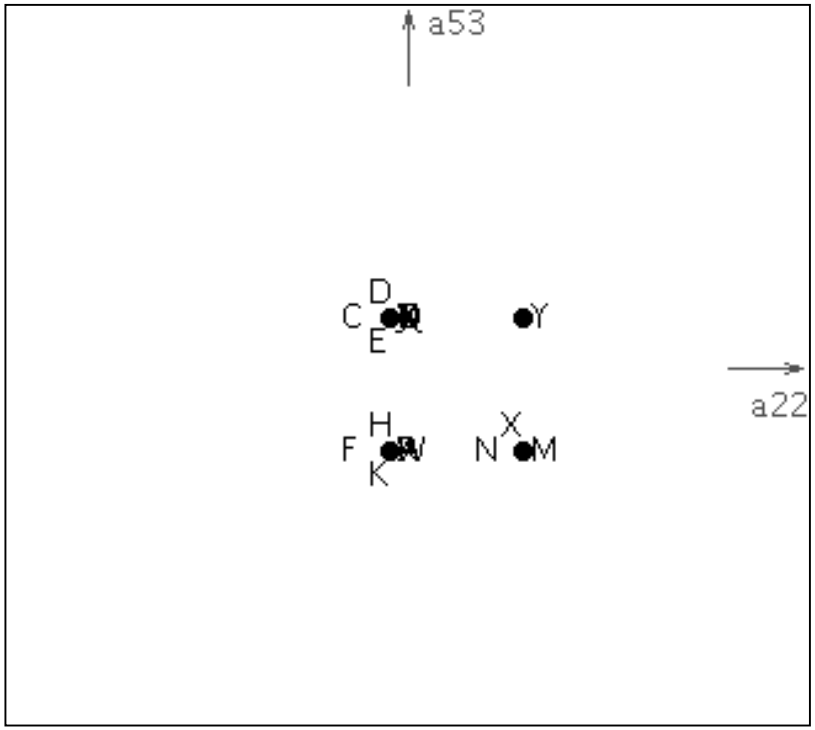
(also with the help of http://www.cs.mcgill.ca/~sqrt/dimr/dimreduction.html)

- A 25-dimensional data set:



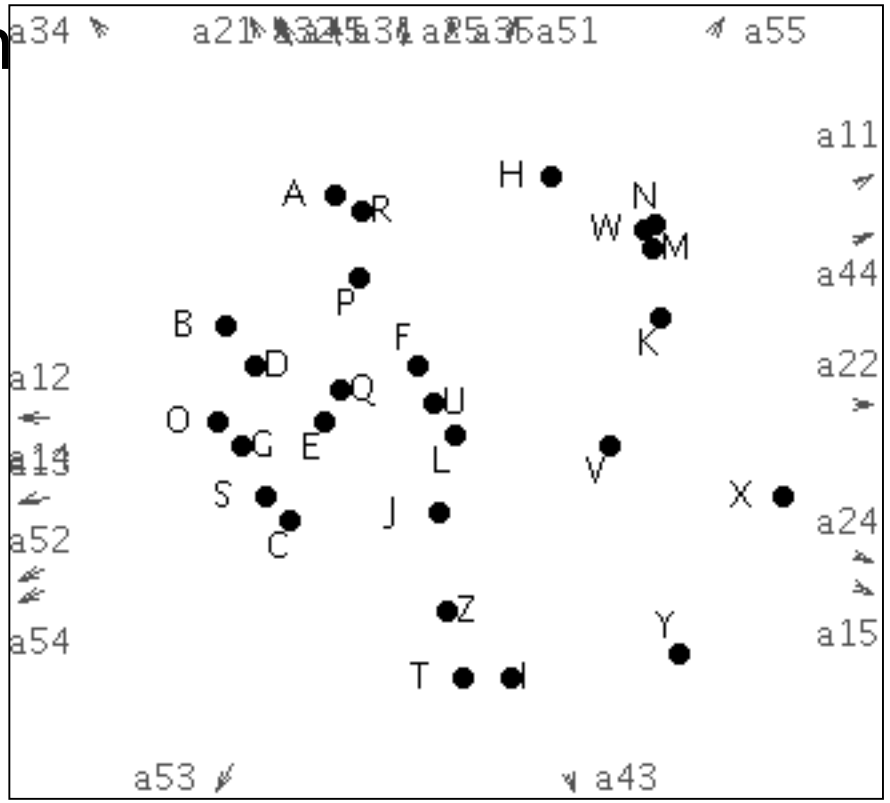
Projected Data

- Projecting on 2 variables does not help much in discriminating the points, e.g.



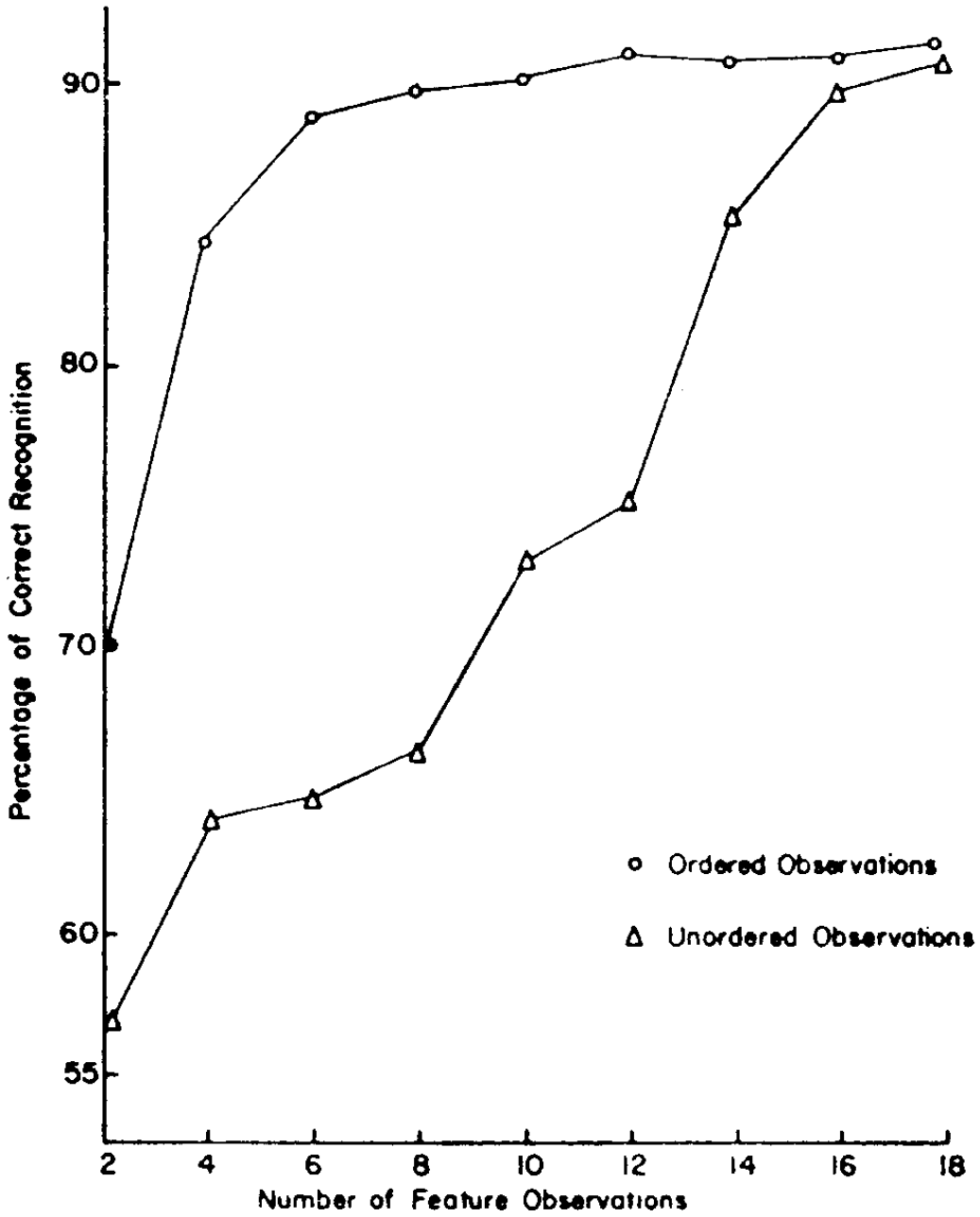
Projected Data

- Projecting on the first two principal components achieves the maximum variation in two dimension



Application: Handwritten Character recognition (K.S. Fu, 1968)

- Attempt recognition based upon **18 radial measurements**, spaced at 20 degree increments, of characters (240 samples).
- Recognition rate was computed vs. the number of measurements used (2, 4, 6, … out of 18) in no particular order.
- The same computation was done with the measurements **ordered by principal components**.
- The next slide compares the two approaches.


90
80
70
60
55
Percentage of Correct Recognition
2
4
6
8
10
12
14
16
18
Number of Feature Observations
Ordered Observations
Unordered Observations

Eigenfaces Example

- Principal components of a raster image
- cf. http://vismod.media.mit.edu/vismod/demos/facerec/basic.html

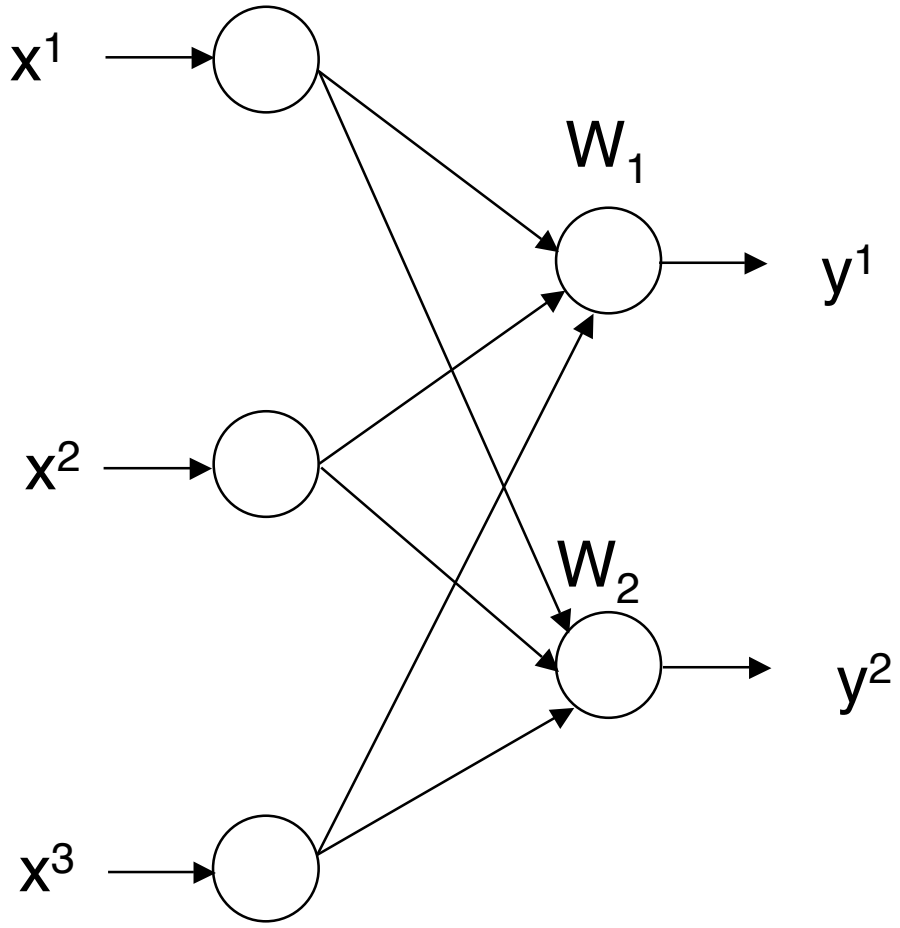
Singular Value Decomposition: Another approach to PCA

- The singular-value decomposition (SVD) of A is:
  $A = USV^T$
- where:
  - The columns of U are the eigenvectors of $A\,A^T$.
  - The columns of V are the eigenvectors of $A^T\,A$.
  - S is pseudo-diagonal.

What is a PCA Network?

- The matrix W of PCA can be interpreted as a 1-layer neural network.
- Each row of W corresponds to **weights** of a neuron.
- Training is by unsupervised Hebbian learning.
- The user pre-selects the output variables.

3→2 PCA Network

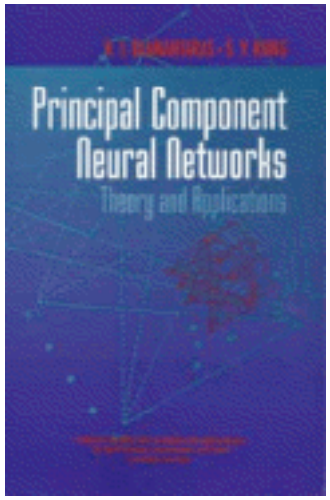


PCA Learning Rules

- $\Delta W = \eta\, y_i\, x_i^T - K_i\, W$, where
  - x_i is an input vector
  - y_i is the corresponding output vector
  - $y_i\, x_i^T$ is the Hebbian component
  - K_i is **one of the following** (purpose: weight decay):
    - 0 pure Hebbian
    - $y_i\, y_i^T$ (Williams' rule, 1985)
    - $3D(y_i\, y_i^T) + 2L(y_i\, y_i^T)$ (Oja and Karhunen rule, 1985)
    - $L(y_i\, y_i^T)$ (Sanger's rule, 1989)

  where D(M) maps the **diagonal** entries of M to themselves, and other entries to 0, and
- L(M) maps entries below the main diagonal to themselves, and other entries to 0.

Other Variations Exist: Book on PCA NN



Diamantaras, K.I. / Kung, S.Y.

Principal Component Neural Networks
Theory and Applications

1996. 256 pages.
ISBN 0-471-05436-4 John Wiley & Sons

The coverage in this book is extensive. It gives several additional learning algorithms, including ones with **lateral** connections as well as feed-forward ones.

Other means of learning for dimensionality reduction

- Self-Organizing Map
  (# of dimensions in superstructure = reduced dimension)
- LVQ
  (# of neurons = reduced dimension)
- Projection pursuit, another statistical technique

PCA Shortcomings

- The PCA user must say how many target dimensions to use.
- PCA only finds linear sub-spaces.
- It works best if the individual components are Gaussian-distributed.
- ICA (e.g. Bell & Senjowski) does not rely on such a distribution.

Independent Component Analysis (ICA)

- Also known as Blind Source Separation.
- Proposed for neuromimetic hardware in 1983 by Herault and Jutten.
- ICA seeks to transform to components that are **statistically independent**.
- Two variables x, y are *statistically independent* iff

  $$P(x\ y) = P(x)P(y).$$

  Equivalently,

  $$E\{g(x)h(y)\} - E\{g(x)\}E\{h(y)\} = 0$$

  where g and h are any functions.

ICA

- Suppose we have two distinct mixtures of two signals that are statistically independent.
- We wish to transform the mixtures to (signals as close as possible to) the original signals.

ICA Demo

- http://www.cnl.salk.edu/~tewon/Blind/blind_audio.html

Neural ICA Algorithms

- Herault-Jutten: local updates

$$B = (I+S)^{-1}$$

$$S_{k+1} = S_k + \beta_k g(y_k) h(y_k^T)$$

$g = t$, $h = t^3$; g = hardlim, h = tansig

- Bell and Sejnowski: information theory

$$B_{k+1} = B_k + \beta_k [B_k^{-T} + z_k x_k^T]$$

$$z(i) = \partial/\partial u(i)\ \partial u(i)/\partial y(i)$$

$u = f(Bx)$; f = tansig, etc.

Amari's Recurrent Neural ICA

- Fully recurrent neural network with self-inhibitory connections.

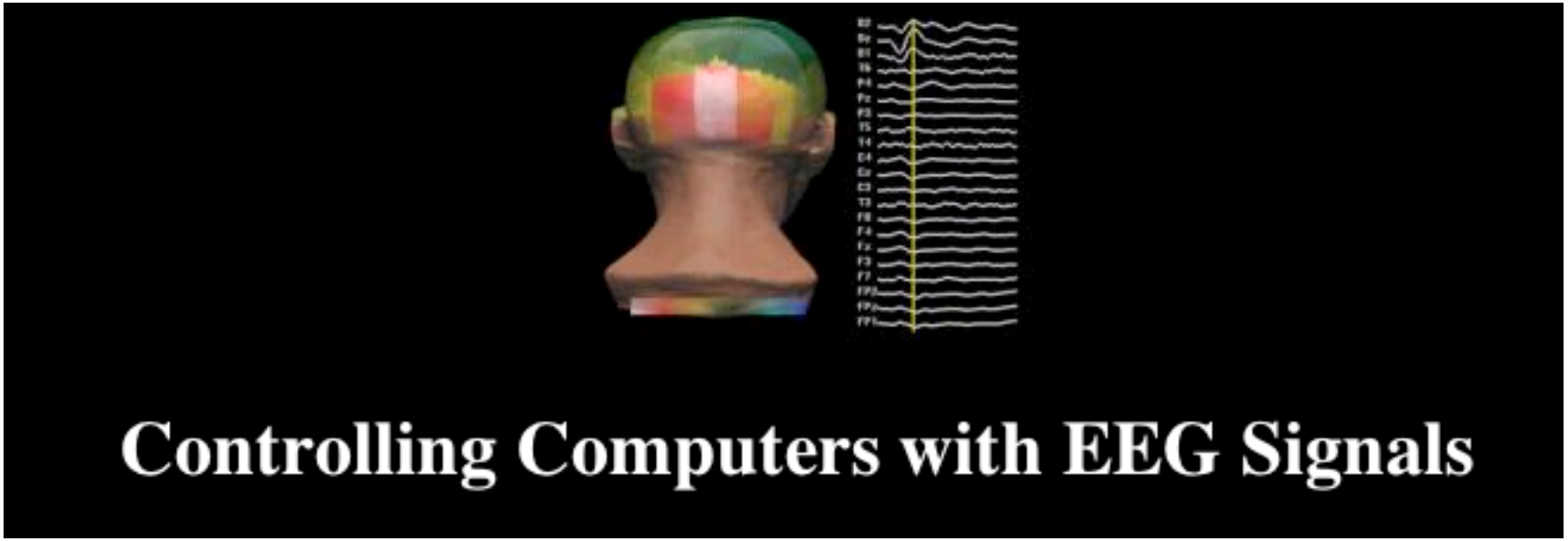
$$\tau_i \frac{dy_i}{dt} + y_i = x_i(t) - \sum_{i=1}^{n} \hat{w}_{ij}(t) y_j,$$

$$y(t) = (I + \widehat{W}(t))^{-1} x(t),$$

$$y(t) = x(t) - \widehat{W}(t) y(t - \tau),$$

$$\frac{d\widehat{W}}{dt} = -\mu(t)[I + \widehat{W}][\Lambda - f(y(t))g^T(y(t))].$$

Culpepper/Keller Project using: http://www.cs.hmc.edu/~bjc/research/



ICA was used to achieve a separation among 12 EEG channels