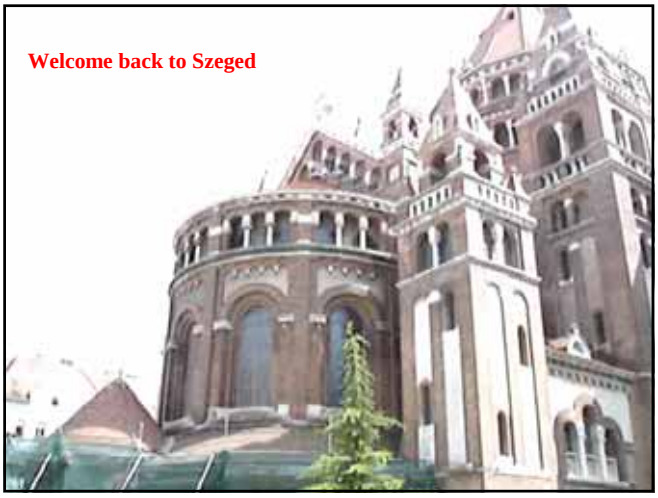
SSIP 2007 Szeged July 13th

Methods for the analysis of a series of image in time.

Andrew Todd-Pokropek
University College London,
A.Todd@ucl.ac.uk

Welcome back to Szeged



Outline

- Examples of temporal series 3D-nD
- Reductions of dimensionality
  - PCA based methods
    - Active shape models
  - Factor analysis based methods
  - ICA based methods
- Change detection (Kalman filtering)
- Conclusions and the future

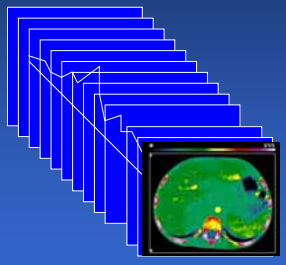
Handling of temporal data

Special class of 3-D data processing

- Looking for change
- Looking for (derived) function

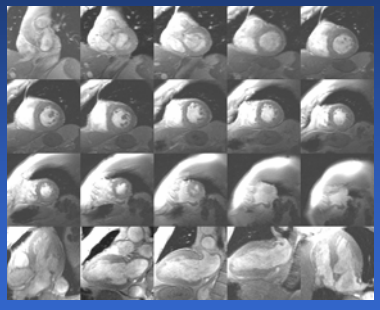
Consider set of time curves for every pixel

- Dual curve/image data set

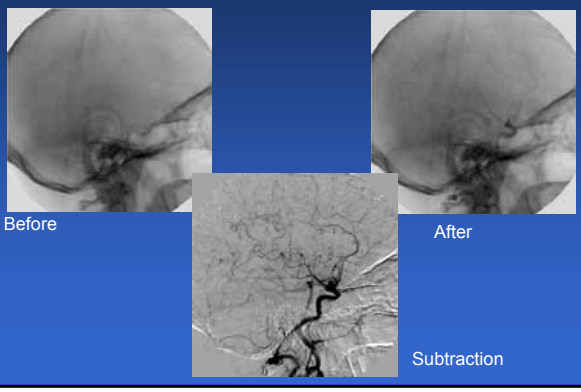


Weather Satellites

4D series

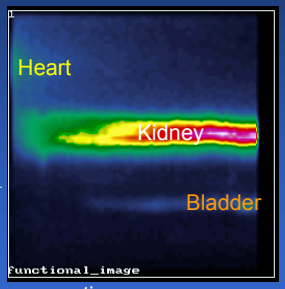


Looking for change



Data compression/ projection

- Removing redundancy
- Reducing dimensionality
- Projection against
  - time (summation)
  - y (vertical axis)
  - oblique
- Constraints are required (a priori information)



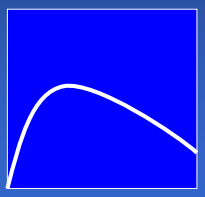
Function fitting

- For cyclical function
  - $A[i,j] = \Sigma_k \; C[i,j,k] \; \cos(\omega k)$
  - $B[i,j] = \Sigma_k \; C[i,j,k] \; \sin(\omega k)$

  - $AMP[i,j] = \text{sqrt}\,(A[i,j]^2 + B[i,j]^2)$
  - $PHASE[i,j] = \tan_{-1} (B[i,j] / A[i,j])$
- First Fourier component

Functional Images

- Image of a derived function
  - Rate of increase/ decrease
  - Time to max
  - Variance image
- Example Kalman filtering
  - Estimating current values
  - And statistical model



Principle Component Analysis

- We have N observations of M variables

$$\mathbf{Z} = \begin{bmatrix} Z_1 \\ Z_2 \\ Z_3 \\ Z_4 \\ . \\ . \\ . \\ Z_N \end{bmatrix} = \begin{bmatrix} z_{11} & z_{12} & z_{13} & z_{14} & \cdots\cdots & z_{1M} \\ z_{21} & z_{22} & z_{23} & z_{24} & \cdots\cdots & z_{2M} \\ z_{31} & z_{32} & z_{33} & z_{34} & \cdots\cdots & z_{3M} \\ z_{41} & z_{42} & z_{42} & z_{43} & \cdots\cdots & z_{4M} \\ . & & & & & \\ . & & & & & \\ . & & & & & \\ z_{N1} & z_{N2} & z_{N2} & \cdots\cdots & & z_{NM} \end{bmatrix}$$

Subtract the mean

$$Z_0 = \frac{1}{N} \sum_{i=1}^{N} Z_i$$

Form the covariance matrix

$$\mathbf{C}_z = \frac{1}{N} \sum_{j=1}^{M} Z_j^t Z_j = \frac{1}{N} \mathbf{Z}^t \mathbf{Z}$$

This is of size MxM

Diagonalise

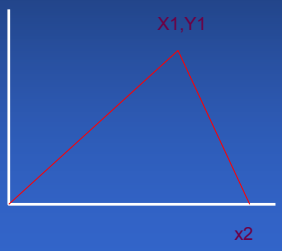
- Form the Eigenvectors and Eigenvalues, where the Eigenvalues or monotonically decreasing and the Eigenvectors are othogonal
- The Eigenvectors express the directionality of the principal axes
- The Eigenvalues express their 'strength' i.e. the amount of variance that is contained in the component

Active Shape Model
A simple example

- A set of triangles
- Characterised by 6 parameters
  - {x1,y1,x2,y2,x3,y3}
- Simpler description possible

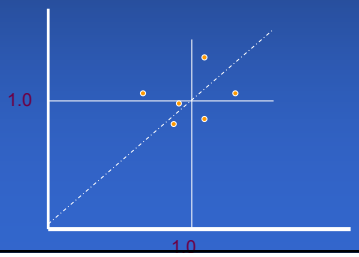
Triangle space

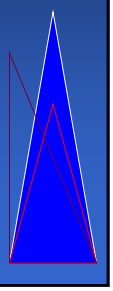
- Fix the origin
- Fix the x axis
- 3 parameters
- Also lengths of 3 sides



Normalise scale

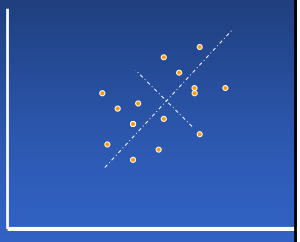
- Take ratios of lengths of 1 side v. other two
- Two dimensional space





Perform a PCA

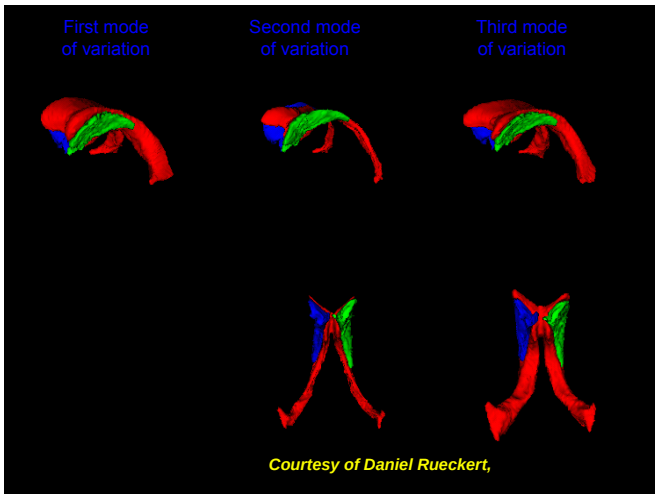
- The first mode of variation is the 1st Eigenvector
- The second mode of variation is the 2nd etc
- We can project each data example onto the corresponding axis



PCA in Matlab

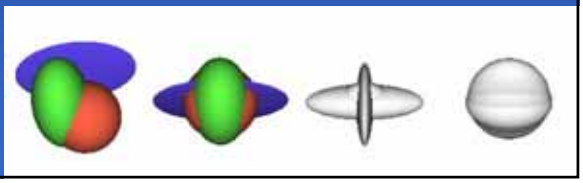
- PRINCOMP Principal Components Analysis.
- COEFF = PRINCOMP(X) performs principal components analysis on the N by-P
- data matrix X, and returns the principal component coefficients, also
- known as loadings. Rows of X correspond to observations, columns to
- variables. COEFF is a P-by-P matrix, each column containing coefficients
- for one principal component. The columns are in order of decreasing
- component variance.
-
- PRINCOMP centers X by subtracting off column means, but does not
- rescale the columns of X.

- Generalized Moore-Penrose inverse is PINV



Where to place the nodes

- Regular
- Maximum radius of curvature
- Minimal description length



Minimum description length

- Energy of the model (PCA)
- Energy of the description (points)
- Minimise total
- The two energies have to be in comparable units
- Add a 'lamda'
- Minimise that also

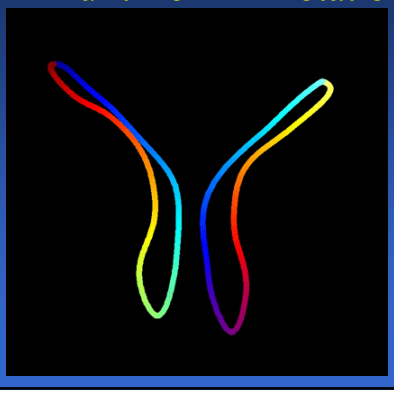
What is MDL?

- Transmission of the (quantized) dataset

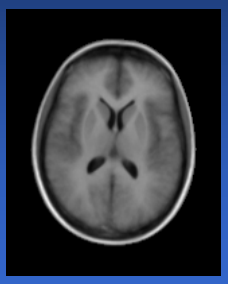
- Must have exact reconstruction of dataset.
- Optimal model ≡ shortest total message length.

$$\mathcal{L}_{total} = \mathcal{L}_{model\ parameters} + \mathcal{L}_{data\ encoded\ using\ model} + \mathcal{L}_{residual/unmodelled\ bits}$$

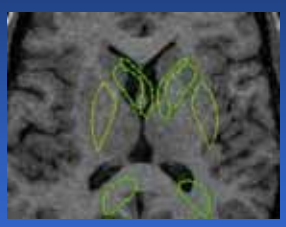
SSM Built from Annotation



Example from Cootes and Taylor



1st Mode of active appearance model



ASM model fitting

Bayesian Analysis

Bayes' theorem

$$\Pr(P \mid E) = \frac{\Pr(E \mid p)\Pr(P)}{\Pr(E)}$$

- Using the prior knowledge of shape to bias the boundary finding

$$\Pr(P_{map} \mid E) = \max \frac{\Pr(E \mid P)\Pr(P)}{\Pr(E)}$$

E is image object, P variables in template P_{map} is desired result

After serial inference

$$\Pr(P) = \prod_{i=1}^{N} \frac{1}{\sigma_i \sqrt{2\pi}} e^{-\frac{(p_i - m_i)^2}{2\sigma_i^2}}$$

$$E = P + noise$$

$$\Pr(E \mid p) = \prod_{A} \frac{1}{\sqrt{2\pi}\sigma_n} e^{-\frac{(E(x,y) - P(x,y))^2}{2\sigma_n^2}}$$

$$M(P) = \sum_{i=a}^{t} [-\frac{(P_i - m_i)^2}{2\sigma_i^2}] + \frac{1}{\sigma_n^2} \sum_{1}^{N} E(x(p,n), y(p,n))$$

m is mean, σ is SD, for N points (x,y) over A for whole image

Neighbour-Constrained Segmentation

(Yang, Staib, Duncan, IPMI03)

Observation:
- Neighbouring structures often have a consistent image location and shape ;
- Relative positions or shapes among neighbors can be modeled based on statistical information from a training set.

•Maximum A Posterior (MAP) framework:

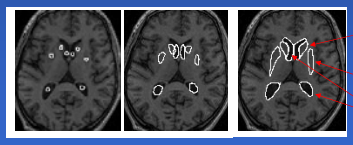
Assume image I has M objects of interest: $S_1, S_2, \ldots S_M$

$$\hat{S}_i = \arg\max_{S_i} p(S_1, S_2, \ldots, S_M / I)$$

$$= \arg\max_{S_i} p(I / S_1, S_2, \ldots, S_M) p(S_1, S_2, \ldots, S_M) \quad i = 1,2,\ldots,M$$

image gray level info

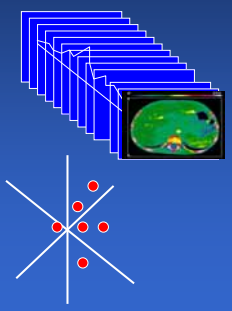
neighbour (shape + distance) prior info



Detection of 8 sub-cortical structures using neighbor priors

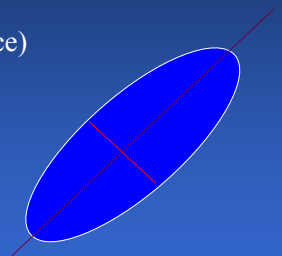
Model based approach

- Y = X + Є
  - Y is data matrix (m,n)
  - X is model
  - Є is error
- Decompose
  - X = F G'
    - F (m,k)
    - G (k,n)



From PCA to factor analysis

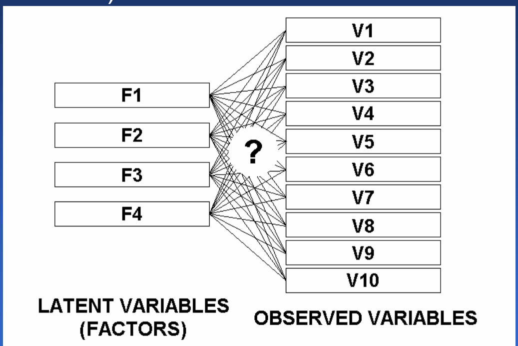
- F are Principal Axes
- G are Weights (variance)
- C (covariance matrix)
  - $C = (Y - y_m)' (Y - y_m)$



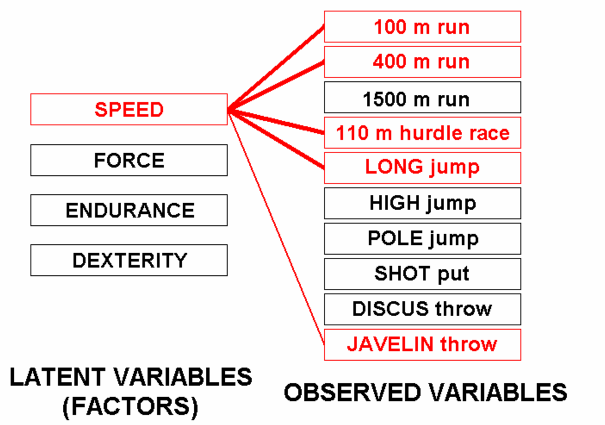
- Concept of factor analysis
- Factor analysis of image sequences
- Applications of principal component and factor analysis in image processing

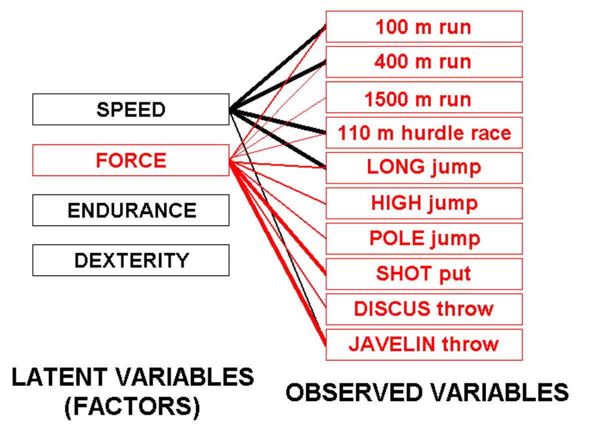
Acknowledgement Martin Samal

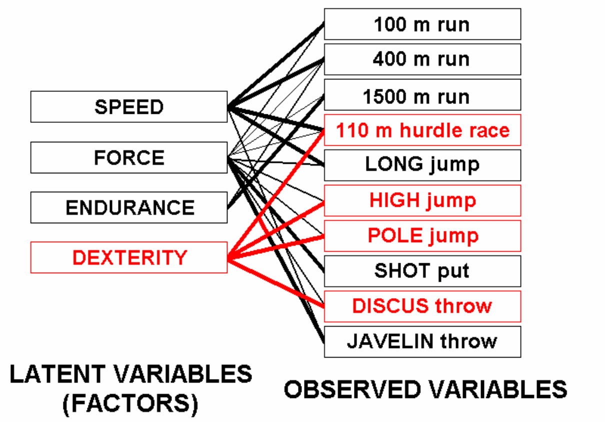
Observed variables are manifestations (functions) of latent variables called factors.

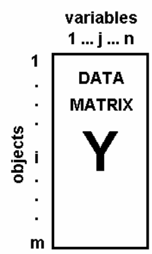


- Structure of observed data
- How many factors
- Factor structure (factor loadings)
- Interpretation of factors (new variables)
- Quantification of factors for individual objects / samples (factor scores)







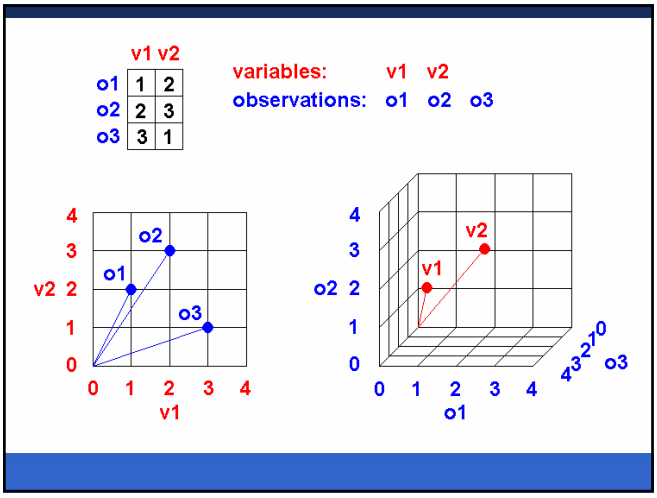


y_{ij} = observation of j-th variable on i-th object

vector interpretation:

(1) m vectors (objects) in n-dimensional space of variables

(2) n vectors (variables) in m-dimensional space of objects



The factor variables are also called

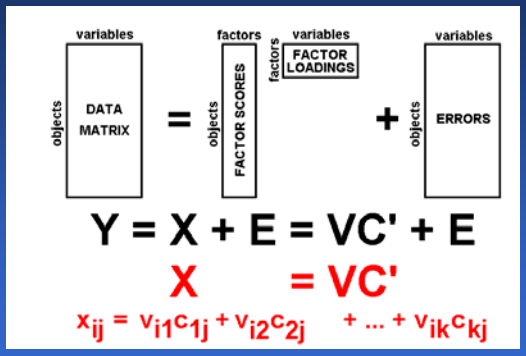
features

of the multivariate random signal,

and the vector space they form is called a

feature space.

Observed variables are linear combinations of factors.



Factor analysis is performed in 2 steps:

1) dimension of data is reduced (data are projected into a subspace of lower dimension)

   = factor extraction

2) base vectors of the subspace are rotated with respect to the data in order to allow useful interpretation

   = factor rotation

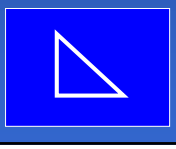
Considering empirical (visual, clinical) analysis of image sequence and omitting the abstract terms, the 2 steps can be interpreted as:

- select the most dissimilar images of a sequence (each is expected to display one dominant factor but still contains the structures of more than one factor), and

- subtract between them to remove residuals of other factors and display a single factor structure per image.

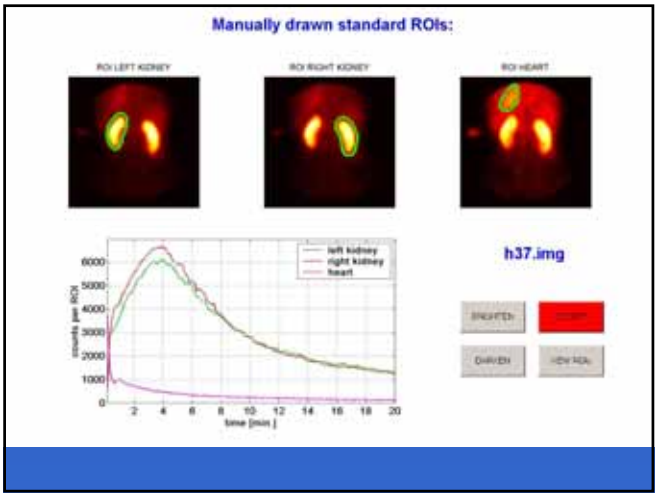
Oblique rotation

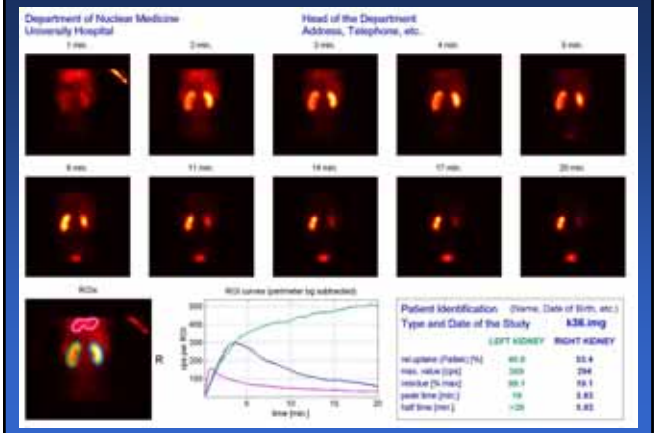
- PCA solution is orthogonal
- Make linear combinations
  - Oblique rotation
  - To satisfy constraint (positivity)
  - For example is higher dimensional space



Factor Analysis in Matlab

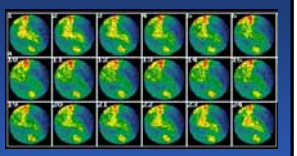
- **Factor Analysis Toolbox**
- The Factor Analysis Toolbox provides a family of specialized MATLAB functions for factor analysis techniques. It is designed to help you learn the principles of target factor analysis and to provide the capabilities necessary for tackling real research and modelling problems. Factor analysis organizes chemical data into matrices so that it can be processed to create calibrations or extract useful information. This makes MATLAB an ideal environment for factor analysis. The Factor Analysis Toolbox provides the functions that enable you to quickly and easily explore your data with factor analysis techniques. This allows you to concentrate on the chemistry while maintaining confidence in the math.

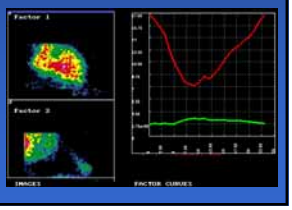




Factor analysis: an example

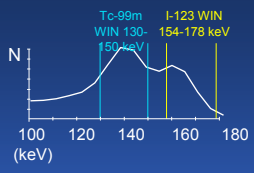
- Decomposition into principal component
- Oblique rotation (based on constraints)
- Display of images (eigenfuctions) and curves (eigenvalues)
- Segmentation, model fitting and quantitation

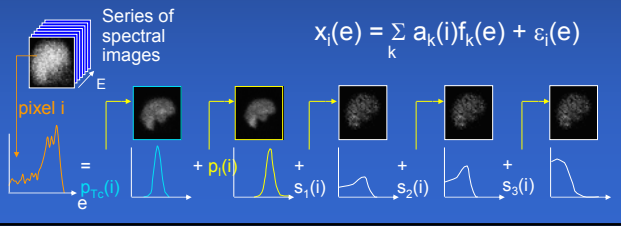




Spectral Analysis

Ack I. Buvat



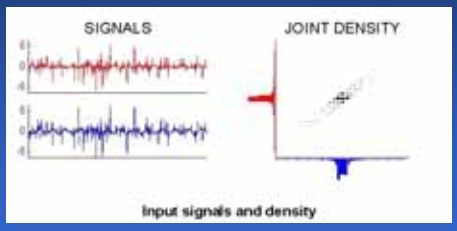


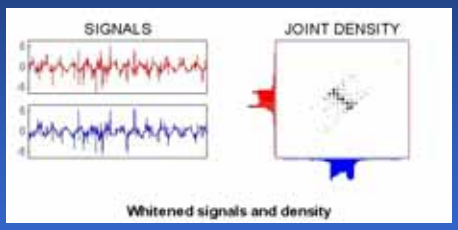
$$x_i(e) = \sum_k a_k(i) f_k(e) + \varepsilon_i(e)$$

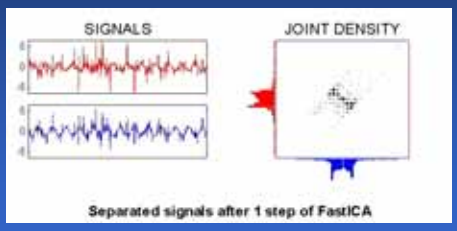
Independent Component Analysis

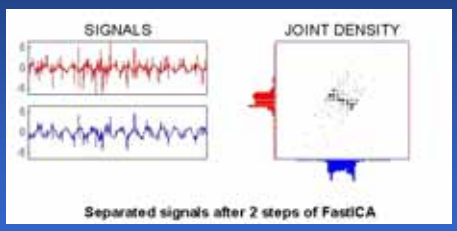
- Cocktail party problem
- Assume signals strictly independent
- Prewhitening
- Components not ordered.

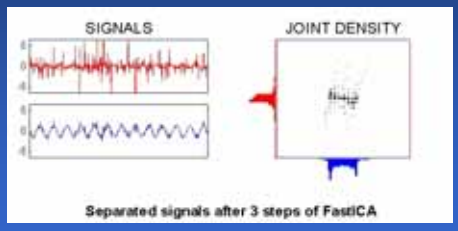
Example

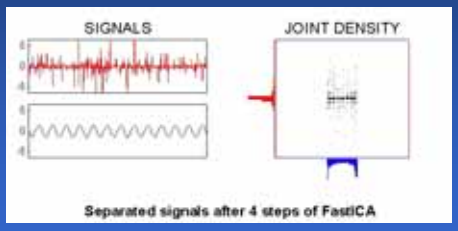

SIGNALS
JOINT DENSITY
Input signals and density

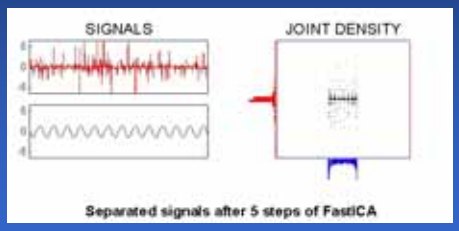

SIGNALS
JOINT DENSITY
Whitened signals and density


SIGNALS
JOINT DENSITY
Separated signals after 1 step of FastICA

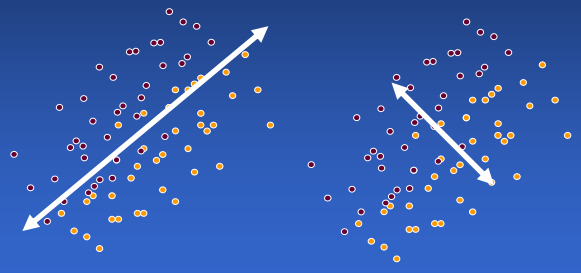

SIGNALS
JOINT DENSITY
Separated signals after 2 steps of FastICA


SIGNALS
JOINT DENSITY
Separated signals after 3 steps of FastICA


SIGNALS
JOINT DENSITY
Separated signals after 4 steps of FastICA



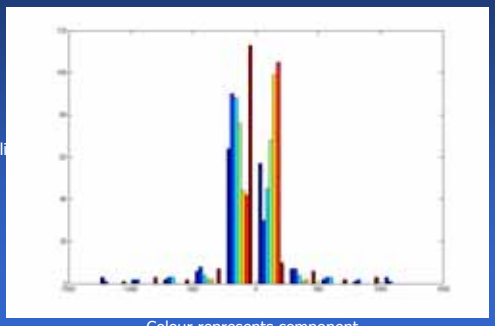
PCA v. ICA



How ICA works

- Originally from signal processing
- Three algorithms:
  1. Fast ICA
  2. InfoMax
  3. JADE
- To obtain 'vectors'

Ordering ICA



Assumptions

- x is n-dimensional random variable
- We wish to find n-dimensional s=f(x)

i.e. s=Wx where W is to be determined

And s_i are as statistically as independent as possible by maximising some function $F(s_1,\ldots s_n)$ that measures independence

All the independent componants s_i must be non-Gaussian

Often pre-whitening is used.

Constrast Functions

- Negentropy $J(y) = H(y_{gauss}) - H(y)$
  - Where y_{gauss} is a Gaussian random variable with the same covariance matrix as y.
- Network Entropy (Infomax) is equivalent to max. likely estimation -> mutual information
- Mutual information = H(x,y)-H(x)-H(y)

Fast ICA

- $w(k)=E(xg(w(k-1)^t x)-E(g'w(k-1)^t x)w(k-1)$
  - W is the weight vector for iteration k
  - g is the derivative if the function G used in the general contrast function
  - where $J_{G(y)} = |E_y\{G(y))-Ey\{G(v)\}|^p$
  - E is expectation wrt Gaussian random variable y, p=1,2,
  - J measures the non-normality
  - Examples of G are: log cosh a1 u or $\exp(-a2u^2/2)$

FASTICA in MATLAB

```
function [Out1, Out2, Out3] = fastica(mixedsig, varargin)
%FASTICA - Fast Independent Component Analysis
%
% FastICA for Matlab 7.x and 6.x
% Version 2.5, October 19 2005
% Copyright (c) Hugo Gävert, Jarmo Hurri, Jaakko Särelä, and Aapo Hyvärinen.
%
% FASTICA(mixedsig) estimates the independent components from given
% multidimensional signals. Each row of matrix mixedsig is one
% observed signal.  FASTICA uses Hyvarinen's fixed-point algorithm,
% see http://www.cis.hut.fi/projects/ica/fastica/. Output from the
% function depends on the number output arguments:
```

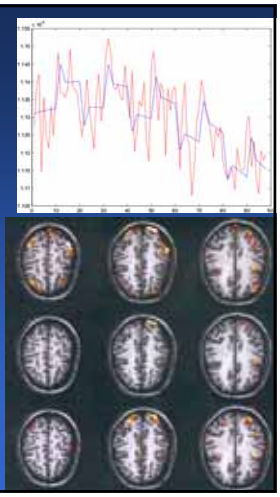
More ICA

- **ICALAB** can be useful in the following tasks:
- 1. Blind Source Separation (BSS), Sequential Blind Sources Extraction (BSE),
- 2. Reduction of redundancy (Book, Chapter 3),
- 3. Decomposition of multi-variable signals into independent components (Chapters 6-8),
- 4. Spatio-temporal decorrelation of correlated signals (Chapter 4),
- 5. Extraction and removal of undesirable artifacts and interference by applying deflation (see Chapters 1 and 4),
- 6. Removal of noise or "cleaning" the raw sensor data,
- 7. Extraction of features and patterns,
- 8. Comparison of the performance of various algorithms for Independent Component Analysis (ICA) and Blind Source Separation (BSS),
- 9. Monte Carlo analysis

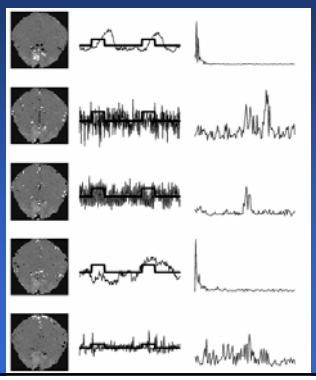
- http://www.tsi.enst.fr/icacentral/algos.html

Applications

- Signal analysis – example very noisy signal such as evoked potentials
- Image analysis – example fMRI



ICA of fMRI



Ack Calhoun

Kalman Filtering

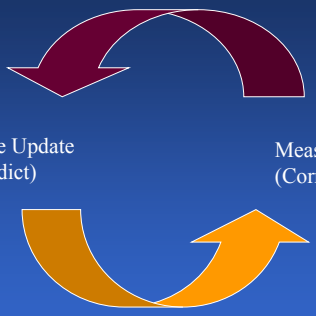
- Problem
  - To estimate the state of $x \in \Re^n$
  - where
    $$x_k = Ax_{k-1} + Bu_{k-1} + w_{k-1}$$
  - With a measurement $z \in \Re^n$

  that is
    $$z_k = Hx_k + v_k$$

  Random variables w_k and v_k are process and measurement noise

  Q is noise covariance and R is measurement noise covariance

  A relates previous step to current step (state transition matrix), B is optional, H (measurement matrix) relates to changes in measurements



$$e_k^- = x_k - \hat{x}_k^-$$

a priori error from the knowledge of previous values

$$e_k = x_k - \hat{x}_k$$

a posteriori error given measurement z_k

Update equations

- Filter time update

$$\hat{x}_k^- = A\hat{x}_{k-1}^- + Bu_{k-1}$$ Project the state ahead

$$P_k^- = AP_{k-1}A^T + Q$$ Project the error covariance

- Filter measurement update

$$K_k = P_k^- H^T (HP_k^- H^T + R)^{-1}$$ Compute the Kalman gain

$$\hat{x}_k^- = \hat{x}_k^- + K_k(z_k - H\hat{x}_k^-)$$ Update estimates with measurements

$$P_k = (I - K_k H)P_k^-$$ Update error covariance

^ **indicates a posteriori estimate** ⁻ **indicates a priori estimate**

Kalman filtering in Matlab

- **KALMAN Continuous- or discrete-time Kalman estimator.**
- **[KEST,L,P] = KALMAN(SYS,QN,RN,NN) designs a Kalman estimator KEST for the continuous- or discrete-time plant with state-space model SYS. For a continuous-time model**

  x = Ax + Bu + Gw {State equation}
  y = Cx + Du + Hw + v {Measurements}

- **with known inputs u, process noise w, measurement noise v, and noise covariances**

  E{ww'} = QN, E{vv'} = RN, E{wv'} = NN,

- **the estimator KEST has input [u;y] and generates the optimal estimates y_e,x_e of y,x by:**

  x_e = Ax_e + Bu + L(y - Cx_e - Du)

  |y_e| = | C | x_e + | D | u
  |x_e| | I | | 0 |

Some references

- PCA: in Wikipaedia, notes By Phillipe, etc Batchelor
- Factor analysis: notes by Martin Samal, book by Gorsuch etc
- ICA: Survey by Aapo Hyvarinen
- Kalman: Welch and Bishop An interduction fo the Kalman filter

Applications- Signals and Images

Standing on the shoulders of giants

Working in multidisciplinary teams

General Conclusions

- Ensure it is a good problem
- Acquire high quality data (as far as possible)
- Validate
- Evaluate
- Adapt