ICA Project Report and Additional Work

By: Chris Bodden and SvyatVergun

ECE 532 – Fall 2014

CONTENTS

Contents 1

1. Introduction 1

2. ICA Overview 2

3. Summary of Lab 2

3.1 FastICA Algorithm and Applications 2

3.2 FastICA Iteration Implementation 2

4. Additional Work 3

4.1 ICA Implementation Using Projection Pursuit 3

4.2 ICA Implementation Using SVD 4

4.3 Overcomplete Blind Source Separation 5

References 7

1. INTRODUCTION

Independent component analysis (ICA) is a technique to decompose an observed signal, made up of several unknown source signals, into the original source signals or subcomponents. ICA requires that the subcomponents be statistically independent from one another (non-Gaussian).

In this lab we apply ICA towards the cocktail party problem, which involves separating individual speech measured in a room with many speakers. These signals are taken to be independent. More generally this problem is called blind source separation because we don't know the original source signals, only some mixing of the sources.

For parts of this lab we use the popular FastICA algorithm package that is freely available from: http://research.ics.aalto.fi/ica/fastica/.

In-Depth ICA Reading:

http://www.stat.ucla.edu/~yuille/courses/Stat161-261-Spring14/HyvO00-icatut.pdf

2. ICA OVERVIEW

Assume that we observe n different linear mixtures $x_1, \dots, x_n$ of the same n original independent source signals. For example, n original audio sources have been combined into n mixtures of the original source. This can be expressed as:

$$x_j = a_{j1}s_1 + a_{j2}s_2 + \cdots a_{jn}s_n \text{ ,for} j = 1 \dots n$$

This can also be expressed in Matrix notation. Let $\boldsymbol{x}$ be the vector whose elements are the mixed signals $x_1, \dots, x_n$, and let $\boldsymbol{s}$be the vector whose elements are the original source signals $s_1, \dots, s_n$. Let $\boldsymbol{A}$ be the square mixing matrix with elements $a_{ij}$for $i, j = 1 \dots n$. By convention, bold lower case letters indicate vectors and bold upper case letters indicate matrices. Using this notation, the mixing model above can be written as

$$\boldsymbol{x} = \boldsymbol{A}\boldsymbol{s}.$$

For ICA there are two unknowns (the mixing matrix, $\boldsymbol{A}$, and the original sources, $\boldsymbol{s}$) and only one known (the mixed signal, $\boldsymbol{x}$). We can solve for $\boldsymbol{A}$and$\boldsymbol{s}$by utilizing the assumption that the sources are statistically independent. This is an optimization problem that minimizes the measure of mutual information between the sources (components). Mutual information is defined as:

$$\sum_{c \in C} \sum_{d \in D} p(c,d) \log\left(\frac{p(c,d)}{p(c)p(d)}\right)$$

Where $p(c,d)$ is the joint probability of variables $c$and d, and $p(c)$ and $p(d)$ is the probability of c and d respectively. The sum is taken over the values of the possible c and d variables.

The FastICA algorithm implements this optimization and outputs the mixing matrix as well as the components.

3. SUMMARY OF LAB

3.1 FASTICA ALGORITHM AND APPLICATIONS

The FastICA package was downloaded and used on a warm-up exercise to separate sine and saw-tooth signals. ICA and PCA were compared for a synthetic bi-directional dataset. Separating Gaussian and non-Gaussian signals was also investigated.

3.2 FASTICA ITERATION IMPLEMENTATION

The FastICA fixed point iteration algorithm was implemented following Hyvärinen and Oja's implementation directions.

4. ADDITIONAL WORK

We have implemented ICA with several algorithms: a projection pursuit method as well as a SVD method.

4.1 ICA IMPLEMENTATION USING PROJECTION PURSUIT

One way we attempted to implement ICA was using a method known as projection pursuit. Simply put this algorithm attempts to find the unmixing matrix $\boldsymbol{W} = \boldsymbol{A}^{-1}$ such that $\boldsymbol{W} * \boldsymbol{X} = \boldsymbol{S}$ and each signal in $\boldsymbol{S}$ is most independent from one another. In other words we are pursuing the projection matrix that maximizes independence of the sources. Formally:

$$\max_{\boldsymbol{W} \in \mathbb{R}^{mxm}} \boldsymbol{kurtosis}(\boldsymbol{W} \times \boldsymbol{X})$$

By maximizing the kurtosis the independence (non-Gaussianity) of the signals is maximized. Note that kurtosis is sensitive to outliers, so projection pursuit is not the ideal method for ICA.

Algorithm sketch:

1. Mean-center & whiten the mixed signals $\boldsymbol{X}$, call this $\boldsymbol{Z}$.
2. Make $\boldsymbol{Z}$ unit variance.
3. Find each column of $\boldsymbol{W}$ one at a time by randomly initializing $\boldsymbol{w_i}$ and then using gradient decent to maximize the kurtosis of $\boldsymbol{w_i} \times \boldsymbol{Z}$.
4. Remove the found signal ($\boldsymbol{w_i} \times \boldsymbol{Z}$) from $\boldsymbol{Z}$ and repeat for all i.

Code (*Note: this is not working because we decided to go with our FastICA implementation. This is close to finished though*):

```
function [ signals, W, A ] = projPursuit( data )
    % init variable
    [numSignals numSamples] = size(data);
    signals = nan(numSignals,numSamples);
    W = nan(numSignals,numSignals);

    % center and whitten data
    Z = data' * pca(data');

    % make unit variance
    for i = 1:numSignals
        Z(:,i) = Z(:,i) / std( Z(:,i) );
    end
    Z = Z';

    % kurtosis function (minus sign is to maximize)
    kurtNum = @(in) -moment(in,4);
    K = @(w) kurtNum(w * Z);

    % compute signals
    for i = 1:numSignals
        w0 = rand(1,numSignals); % random initial weight vector
        w = fminunc(K,w0, {'MaxIter', 100000}); % optimize w
        W(i,:) = w;
        signals(i,:) = w * Z;
```

```
        signals(i,:) = signals(i,:) / norm(signals(i,:));

        % subtract proj of signals(i,:) onto Z
        proj = ((Z'\signals(i,:)') * signals(i,:));
        Z = Z - proj;
    end

    A = inv(W);
end
```

4.2 ICA IMPLEMENTATION USING SVD

Reference:

J. Shlens, "A Tutorial on Independent Component Analysis," 7 April 2014. [Online]. Available: http://arxiv.org/pdf/1404.2986v1.pdf

Algorithm sketch:

1. Examine the covariance of the data $\boldsymbol{x}$ in order to calculate $\boldsymbol{U}$ ($\boldsymbol{E}$) and $\boldsymbol{S}$ ($\boldsymbol{D}$) (covariance: $\boldsymbol{x}\boldsymbol{x}^T = \boldsymbol{E}\boldsymbol{D}\boldsymbol{E}^T$).

   $\boldsymbol{W} = \boldsymbol{V}\boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{E}^T$, where $\boldsymbol{D}$ and $\boldsymbol{E}$ are the eigenvalues and eigenvectors (SVD singular values and singular vectors) of the covariance of the data $\boldsymbol{x}$**.**

2. Return to the assumption of independence of $\boldsymbol{s}$ to solve for $\boldsymbol{V}$ (in $\boldsymbol{W} = \boldsymbol{V}\boldsymbol{S}^{-1}\boldsymbol{U}^T$).

   $\boldsymbol{V} = argmin_{\boldsymbol{V}} \sum \boldsymbol{H}[(\boldsymbol{V}\boldsymbol{x}_{\boldsymbol{w}})_i]$, where $\boldsymbol{H}$ is the entropy. This is minimized using fourth-order blind identification from Cardoso (1989) (see code below).

Code:

```
% mean center data
means = mean(data');
fori = 1:numSignals
X(i,:) = data(i,:) - means(i);
end

% plot mean centered
figure
scatter(X(1,:),X(2,:));
title('mean-centered data');

% whitten data
    M = X * X'; % covariance
    [E,D] = eig(M);
    d = diag(D);
    d = real(d.^-.5);
    D = diag(d);
    Q = D * E'; % whittening matrix
```

```
    Z = Q * X;

% plot whittened
figure
scatter(Z(1,:),Z(2,:));
title('whittened, mean-centered data');

% calculate rotation matrix
% method: FOURTH-ORDER BLIND IDENTIFICATION
% http://perso.telecom
      paristech.fr/~Cardoso/Papers.PDF/icassp89.pdf
% |Z|^2 * Z * Z'
% also see: http://arxiv.org/abs/1404.2986
Z_squared = Z.*Z;
sum_Z_squared = sum(Z_squared,1); %|Z|^2
repeated_sum = repmat(sum_Z_squared,numSignals,1) .* Z %|Z|^2 * Z
cov_matrix = repeated_sum*Z';% |Z|^2 * Z * Z'
    [V,~,~] = svd(cov_matrix); % V is the rotation for the
whittening matrix

% compute W, A, and original signals
    W = V * Q; % unmixing matrix
signals = W * data; % unmix signals
    A = inv(W); % A = W ^-1
    A = A / norm(A); % normalize A
```

4.3 OVERCOMPLETE BLIND SOURCE SEPARATION

Reference:

> M. S. Pedersen, D. Wang, U. Kjems and J. Larsen, "Overcomplete Blind Source Separation by Combining ICA and Binary Time-Frequency Masking," in Machine Learning for Signal Processing, 2005 IEEE Workshop on, Mystic, CT, 2005.

In our original proposal we had considered focusing more on ICA as a tool for blind source separation than on the actual mechanics of ICA. After the initial round of feedback we decided to change our approach and focus more on ICA than blind source separation. However, in the lab we do explore an ideal example of blind source separation, when the number of signals is equal to the number of source (meaning A and $W=A^{-1}$ are square matrices).

But what happens if A is not square, meaning there are more speech sources than microphones? The method presented in this paper is designed to handle this case. While we did not get to in depth into coding this method, it works as follows:

Algorithm sketch (for 2 mixed signals and more than 2 sources):

1. Use ICA to decompose the mixed signals into the 2 most independent pseudo-sources.
2. Use a Short-Time Fourier Transform to transform these signals into the time-frequency domain.
3. Based on a threshold, create a binary mask for each of the 2 signals.

4. Apply each binary mask to each of the original mixed signals (in the time-frequency domain).
5. Convert the signal back by using the Inverse Short-Time Fourier Transform
6. Use a stopping criteria to determine if one of the signals is a source signal. If so, add the signal to the list of source signals. Otherwise, repeat the process on the masked signals.

REFERENCES

[1] A. Hyvärinen and E. Oja, "Independent Component Analysis: Algorithms and Applications," [Online]. Available: http://www.cs.helsinki.fi/u/ahyvarin/papers/NN00new.pdf.

[2] J. Shlens, "A Tutorial on Principal Component Analysis," 7 April 2014. [Online]. Available: http://arxiv.org/pdf/1404.1100.pdf.

[3] J. Shlens, "A Tutorial on Independent Component Analysis," 14 April 2014. [Online]. Available: http://arxiv.org/pdf/1404.2986.pdf.

[4] S. Ikeda, "Some Trials of Blind Source Separation," [Online]. Available: http://www.ism.ac.jp/~shiro/research/blindsep.html.

[5] J. Särelä, P. Hoyer and E. Bingham, "Cocktail Party Problem," [Online]. Available: http://research.ics.aalto.fi/ica/cocktail/cocktail_en.cgi.

[6] J.-F. Cardoso, "Source separation using higher order moments," in *Acoustics, Speech, and Signal Processing, 1989. ICASSP-89., 1989 International Conference on*, 1989.

[7] M. S. Pedersen, D. Wang, U. Kjems and J. Larsen, "Overcomplete Blind Source Separation by Combining ICA and Binary Time-Frequency Masking," in *Machine Learning for Signal Processing, 2005 IEEE Workshop on*, Mystic, CT, 2005.