A New Algorithm using Independent Components for Classification and Prediction of High Dimensional Data

Subhajit Chakrabarty[1][a] and Haim Levkowitz[2]
Louisiana State University Shreveport, LA, U.S.A.
University of Massachusetts Lowell, MA, U.S.A.

Keywords: High Dimension, Independent Component Analysis, Principal Component Analysis, Clustering, Classification, Dimension Reduction, Stability.

Abstract: Dimensionality reduction of high-dimensional data is often desirable, in particular where data analysis includes visualization – an ever more common scenario nowadays. Principal Component Analysis, and more recently Independent Component Analysis (ICA) are among the most common approaches. ICA may output components that are redundant. Interpretation of such groups of independent components may be achieved through application to tasks such as classification, regression, and visualization. One major problem is that grouping of independent components for high-dimensional time series is difficult. Our objective is to provide a comparative analysis using independent components for given grouping and prediction tasks related to high-dimensional time series. Our contribution is that we have developed a novel semi-supervised procedure for classification. This also provides consistency to the overall ICA result. We have conducted a comparative performance analysis for classification and prediction tasks on time series. This research has a broader impact on all kinds of ICA applied in several domains, including bio-medical sensors (such as electroencephalogram), astronomy, financial time series, environment and remote sensing.

1 INTRODUCTION

Independent Component Analysis (ICA) and Principal Component Analysis (PCA) are powerful methods for separation of multiple components (sources) from mixed signals that are high dimensional. PCA can separate the mixtures into components that will be orthogonal to each other, but it may not lead to the true sources, such as audio. PCA is able to find directions of maximum variance. Independent Component Analysis can perform this separation better because it finds directions most aligned with the data. The independent components need not be orthogonal to each other, unlike principal components. Another benefit of ICA is that it considers the true statistical independence (independence of all higher moments), while PCA considers independence only up to second moments. Thus, in PCA we maximize variance while in ICA we maximize cumulants (kurtosis) or likelihood (or mutual information or entropy). Hence, the estimated sources are uncorrelated in PCA while they are statistically independent in ICA. However, the ICA procedure includes pre-whitening (making the covariance as the identity matrix), which could be performed by PCA or other methods. Therefore, PCA and ICA are related to each other. Overall, with high-dimensional data in which dimension reduction is desirable, ICA is potentially superior to Principal Component Analysis (PCA). This leads us to the possibility of better applications of ICA for analysis of high-dimensional data, such as classification, clustering, and prediction.

ICA may output components that are redundant. For example, when we perform ICA with 100 components on electro-encephalogram (EEG) data, just about a dozen components are recognizable by the trained human expert. Further, multiple runs of the ICA over the same input signals provide different estimates of components – the estimates are unstable. Resampling has been attempted (Meinecke et. al., 2002) but is not effective in tackling instability (Chakrabarty and Levkowitz, 2019). Clustering of the independent components is important in this context. Interpretation of such groups of independent components may be achieved through application to

[a] https://orcid.org/0000-0003-0818-3190

Chakrabarty, S. and Levkowitz, H.
A New Algorithm using Independent Components for Classification and Prediction of High Dimensional Data.
DOI: 10.5220/0009148602650272
In *Proceedings of the 15th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2020) - Volume 3: IVAPP*, pages 265-272
ISBN: 978-989-758-402-2; ISSN: 2184-4321

tasks, such as classification or regression (Chakrabarty and Levkowitz, 2019).

The problem is that grouping independent components for high-dimensional time series is difficult. The objective is to provide a comparative analysis of grouping independent components for a given prediction and classification task with high-dimensional time series.

The contribution of this work is with respect to the classification task in which we have developed a novel semi-supervised procedure for classification. We demonstrate our results by using an array of the fourth cumulants. ICA has problems of inconsistency – when we perform ICA multiple times, we get different results. Our method performs ICA several times in order to provide consistency to the overall ICA result.

We have posed the following research questions.

1. Can ICA or PCA improve prediction performance over a baseline method, Auto Regressive Moving Average (ARMA), for some high-dimensional time series datasets?
2. Can ICA perform better than PCA classifying some high-dimensional time series datasets?

This paper is organized as follows. First, we present a very brief review of clustering methods and independent component analysis. Then, we mention the datasets used for this paper. We then present our algorithm and snippets of code to illustrate the implementation. We then discuss our comparative results, followed by our conclusions.

2 BACKGROUND

Classification and clustering are similar but not the same. The basic problem in clustering is: Given a set of data points, partition them into a set of groups that are as similar as possible (Aggarwal, 2014). Clustering is the art of finding groups in data (Kaufman & Rousseeuw, 2005). Clustering refers to grouping of data when the groups are unknown beforehand. In Classification, the groups (categories or classes) are known and there is a need to identify which group (category or class) each data belongs to (supervised learning). But in clustering, the groups are unravelled from the data (unsupervised learning). However, the same data may be tested for both classification and clustering: if we do not use the class information then it is clustering (otherwise classification).

Classification and clustering are broad terms for several methods and approaches. The broad approaches to clustering are named as Partition clustering, Hierarchical clustering, Density-based clustering, Grid-based clustering, Graph clustering, Time series clustering, Semi-supervised clustering, Spectral clustering, and Manifold clustering. Further, different domains may have different methods of clustering – such as, Document clustering, Stream clustering, Multimedia data clustering, and High-dimensional data clustering.

Partition clustering relocates points from one partition to another. The advantage is that the quality of clustering can be improved with iterative optimization (Berkhin, 2006). Typically, the number of partitions is pre-defined. So, if three clusters are known and information on, say, customer data is available, the data point of each customer can be relocated to obtain an optimal quality of clusters (to be validated). The relocation of points is performed over many iterations.

Hierarchical clustering recursively groups in a bottom-up (agglomerative) or top-town (divisive) manner. This does not require a user-defined number of clusters (Jain, Murty, & Flynn, 1999). So, each data point is grouped with a similar data point (based on a distance measure) and these smaller groups are grouped together to form larger groups recursively. From the top of this tree-like structure, one can see the required grouping at the desired level of grouping.

Density-based clustering basically groups based on a threshold density of points (Ester, Kriegel, Sander, & Xu, 1996). So, the adjacent data points in a particular cluster may have distances less than the threshold.

Grid-based clustering uses a grid for faster computation (Wang, Yang, & Muntz, 1997). Assuming that the data is uniform, it can be partitioned into a given number of cells, and the cells may be sorted according to their densities (this method overlaps with density-based method). Then, the partition centers can be identified. The challenge is to determine the grid.

Graph clustering uses the connectedness within sub-graphs to group them (Schaeffer, 2007). Thus, for example, if customer data can be represented in the form of a connected graph, such as a social media network of online customers, one can find clusters of these online customers, for example college students and working professionals (each grouped based on connectedness).

Time series clustering performs grouping of series having similar trends or similar shapes (Yi, et al., 2000; Liao, 2005). For example, customer data may represent monthly sales of several products over 15 years. Some product groups may have seasonality (e.g., selling more during winter) or may be selling together having a long-run relationship. Time series of electro-encephalograms of the scalp of epilepsy patients may be grouped as those with seizure and those without.

Semi-supervised clustering performs grouping by using information, such as labels for seeds (initializing), pairwise constraints, active learning, and user feedback (Chapelle, Scholkopf, & Zien, 2006). For example, when clusters in customer data are sought, some prior information, such as initial identification of few clusters (labels) or expert feedback on identification of clusters for particular points, could help in the effort.

Spectral clustering uses the spectrum (eigenvalues) of the similarity matrix of the data to perform dimensionality reduction before clustering in fewer dimensions (Filiponne, Camastra, Masulli, & Rovatta, 2008). Sometimes, the dimensions are so many that one can only deal with subspaces (subspace clustering) though subspaces may be different among themselves. Dimensionality reduction is a better idea. This could be done with particular matrix operations in the process of spectral clustering.

Manifold clustering uses nonlinear dimensionality reduction (such as using Kernel Principal Component Analysis or Locally Linear Embedding) on the data before clustering in fewer dimensions (Roweis & Saul, 2000). Nonlinear dimensionality reduction manages the "curse of dimensionality" (data becomes increasingly sparse and creates new problems in high dimensions) to some extent.

The above are broad approaches. An overlap of these approaches is possible, as shown in the case of grid clustering. Other generalizations of the types of clustering approaches are possible – such as, those based on whether the underlying data representation is feature-based (vector of features) or graph-based (similarity graph between data points). K-means is an example of a feature-based approach, while spectral clustering is an example of a graph-based approach.

It is important to note that K-means is an optimization problem that cannot guarantee a global optimum solution. A major drawback of the K-means algorithm is that it is highly sensitive to the initial K-means. One popular option for initialization is to use random values in the partitions; another option is to draw from some distribution (e.g., normal).

There are many adaptations of the K-means algorithm, such as K-medians, K-medoids, Fuzzy C means, and K-modes. "Mean" has no meaning for categorical data. For example, the K-modes algorithm can work on categorical data. As compared with K-means, K-modes uses modes (frequencies of mismatches or a matching metric).

K-medoids is synonymous with Partitioning Around Medoids (PAM). Medoids are similar to means or centroids, but medoids will always be members of the data set. Medoids are commonly used when a mean or centroid cannot be defined, such as in graphs. This method starts from an initial set of medoids and iteratively replaces one of the medoids by one of the non-medoids if it improves the total distance of the resulting clustering.

Using a generalization of K-means, one can use a model-based clustering method called the Expectation-Maximization (EM) algorithm (Dempster, Laird, & Rubin, 1977). EM finds clusters by determining a mixture of Gaussians that fit a given set of observations. The parameters can be initialized randomly or by using the output of K-means. It has two steps, the Expectation step, in which the expected value of log likelihood is calculated, and the Maximization step, in which parameters maximizing the expected value are calculated and fed into the Expectation step iteratively.

Clustering algorithms can have variants that can be adapted to given circumstances / conditions. To this end, it is important to identify the underling nature of the data and to understand the underlying domain, particularly when it has high dimensions.

High dimensionality brings in a special kind of challenge called the "curse of dimensionality" (a term coined by Ricard E. Bellman), in which the data becomes increasingly sparse, and presents various problems – such as, global optimization difficulty increases exponentially, similarity measures such as L_P norm becomes less useful, and irrelevant attributes arise. There are two basic approaches in clustering high-dimensional data – projected clustering (Aggarwal, Procopiuc, Wolf, Yu, & Park, 1999) and subspace clustering (Agrawal, Gehrke, Gunopulos, & Raghavan, 1998). Projected clustering partitions the dataset in such a way that each point belongs to exactly one cluster by projecting on the attributes of the cluster. In subspace clustering a point may belong to more than one cluster (partial membership and overlaps are allowed). Subspace clustering finds all clusters in all subspaces. There are also hybrid approaches. It is important to note that there is no general solution to clustering on high dimensional data. So, some algorithms work on interesting subspaces, some try to build hierarchically, some try to optimize locally, and so on. Reducing dimensions is important for feature extraction and feature selection.

The concept of feature selection is slightly different from traditional feature extraction. In feature extraction, the features are projected onto a new space with lower dimensionality. Examples of feature extraction methods include Principal Component Analysis, Linear Discriminant Analysis, and Singular Value Decomposition. In feature selection, a small subset (variables) of features is selected that minimizes redundancy and maximizes relevance to the class label. Examples of feature selection methods include Information Gain, Relief, and Fischer Score. Feature extraction/selection is a very important step prior to tasks such as clustering or prediction. (Liu and Motoda, 2007; Liu and Yu, 2005; Chakrabarty, 2018).

Prediction is performed by regression methods and their variants. Regression is broadly of two types – linear regression and non-linear regression. Another way to see the variants of regression are: Logistic regression, Quantile regression, Ordinal regression, Poisson regression, Cox regression, Support vector regression, Partial least squares regression, Ridge regression, Lasso regression, ElasticNet, and Polynomial regression. It is also possible to perform prediction over components (PCA or ICA).

ICA comprises of several related algorithms and methods. The key groups of algorithms can be classified as higher order statistics (HOS) or second order statistics (SOS). SOS is also known as time-structure based. For sensor data, the main algorithms for ICA are FastICA (Hyvärinen and Oja, 1997), second order blind identification (SOBI) (Belouchrani et. al., 1997), extended information-maximization (InfoMax) (Lee et. al., 1999), adaptive mixture of independent component analyzers (AMICA) (Palmer et. al., 2011), algorithm for multiple unknown signals extraction (AMUSE) (Tong et. Al., 1990), joint approximate diagonalization of eigen-matrices (JADE) (Cardoso and Souloumiac, 1993; Miettinen et. al., 2017), and temporal decorrelation separation (TDSEP) (Ziehe and Muller, 1998).

Broadly, there may be several choices for the methods – based on objective, iterative procedure et cetera. Some of which are mentioned as follows.

Objective: Cumulant based; Maximum likelihood based.

Iterative procedure: Batch method; Adaptive method; Relative gradient.

Extraction of components: Iterative/deflationary; Joint diagonalization/symmetric/simultaneous extraction.

Non-stationarity: Quadratic and other methods.

Pre-whitening: PCA; ZCA.

Other algorithm variants: Subspace ICA; Bayesian approaches; Semi-blind approaches.

Excellent reviews of ICA can be found in (Comon and Jutten, 2010) and (Shi, 2011).

Clustering of independent components have been performed for the Icasso index (Himberg and Hyvärinen, 2003). But this used hierarchical clustering (and Euclidean distance). While hierarchical clustering is visually appealing, this may not be the best choice in high dimensions because of difficulties in selection of merge or spilt points, no backtracking, no object swapping between clusters and poor time complexity (does not scale well).

ICA has been performed for recovering missing signal data segments, stock market prediction, and financial time series. However, time series present their own challenges. For example, variables may be dependent on their own values in the previous period, called auto-regression. The mean and variance may change over time, called non-stationary. The baseline methods are the Auto-Regressive Moving Average (ARMA) and Auto-Regressive Integrated Moving Average (ARIMA). These may be univariate or multivariate. Multivariate ARMA/ARIMA models have not been explored in the context of ICA literature.

3 METHODS

3.1 Datasets

As our enquiry involves dimension reduction, we would prefer a high-dimensional dataset that is openly available. For the classification task involving time series, we have used the Epilepsy Seizure Recognition dataset (archive.ics.uci.edu/ml/datasets/ Epileptic+Seizure+Recognition) from the open UCI Machine Learning Repository. For the prediction task with time series, the UCI Machine Learning Repository provides us the Istanbul Stock Exchange dataset (https://archive.ics.uci.edu/ml/datasets/ ISTANBUL+STOCK+EXCHANGE#).

The Epilepsy Seizure Recognition dataset has 11,500 rows, each row containing 178 data points for 1 second (columns) and the last column represents the label y {1,2,3,4,5}. All subjects falling in Classes 2, 3, 4, and 5 are subjects who did not have an epileptic seizure. Only subjects in Class 1 have had an epileptic seizure. The dataset is unbalanced if we consider binary classification of seizure. There are 2,300 rows for seizure. We take 2,300 non-seizure rows from the already-shuffled dataset. So, our balanced dataset for binary classification is a 4,600 by 178 matrix.

The Istanbul Stock Exchange dataset is organized about working days in the Istanbul Stock Exchange. The attributes are stock exchange returns for the Istanbul stock exchange national 100 index, the Standard & Poor's 500 return index, the Stock market return index of Germany, the Stock market return index of the UK, the Stock market return index of Japan, the Stock market return index of Brazil, the MSCI European index, and the MSCI emerging markets index. The dataset has 536 rows (time) and 10 columns.

We do not have generalization claims that would apply to all datasets. We have simply tested if we can improve prediction and classification performance on the given datasets and the potential of ICA; but we have developed a new algorithm.

3.2 Methods

For prediction, we performed the following.

1. Baseline Auto Regression Moving Average (ARMA)
2. Multivariate ARMA on PCA
3. Multivariate ARMA on ICA

For classification, we performed the following.

1. PCA followed by k-means clustering
2. ICA followed by k-means clustering
3. ICA followed by Partitioning Around Medoids (PAM) clustering
4. ICA followed by semi-supervised learning and classification (with our own algorithm).

The clustering was validated through external means (Xiong and Li, 2014) – labels are available. So, for k-means and PAM, we used an unsupervised method for a classification task because labels were available.

The programming environment was R. The important libraries used were fastica, stats, cluster, caret, marima, and their dependencies. Our source code and dataset will be made freely available for reproducibility.

3.3 Novel Algorithm

Our novel algorithmic procedure was in the classification task. It is as follows.

1. Perform ICA *n* times for reduced dimension.

If we take three independent components out of 178 columns, the result is a 4,600 by 3 matrix for each ICA. The number of iterations of ICA, *n*, can be empirically estimated to get stable (asymptotic) results. We can choose 20; more is better.

2. For each ICA, calculate the fourth cumulant of each independent component.

So, we have 4,600 such values per ICA iteration.

3. Find the maximum fourth cumulant over the iterations.

So, this will be a vector of size 4,600.

4. Partition the rows based on the maximum fourth cumulant by learning a threshold. Some other statistic could also be used.

This can easily be performed by observing the change in accuracy or F1 (from confusion matrix) with a change in the single threshold parameter.

The fourth cumulants are calculated as follows.

```
cumulant_4 <- function(estimated)
{
cum4 <- vector('numeric')

for (i in 1:ncol(estimated))
{
TS <- estimated[,i]#time-series column-
wise
cum4 <- c(cum4,(mean(TS^4)-
4*mean(TS)*mean(TS^3)-
(3*mean(TS^2)^2)+12*(mean(TS)^2)*mean(T
S^2)-6*mean(TS)^4))
}

return(cum4)
}
```

In the above code, the name of the function is 'cumulant_4' and it takes in a matrix, called 'estimated', as its parameter. Inside the function, 'cum4' is a numeric vector that is first initialized as blank. The 'for' loop runs from 1 to the number of columns in the matrix named 'estimated'. 'TS' is the column-wise timeseries. Next, the statistical formula of fourth cumulant is calculated within the loop and returned after the loop.

The application of the array of fourth cumulants to iterations of ICA is performed as follows.

```
library(fastICA)

iterations_ica <-
as.integer(readline(prompt="Enter
number of iterations of ICA: ")) #20

number_components <-
as.integer(readline(prompt="Enter
number of ICA components: ")) #5

c4 <- matrix(,nrow = iterations_ica,
ncol = rowNumbers)#for fourth cumulants

compICA <- matrix(, nrow = rowNumbers,
ncol = number_components)

for(itr in 1:iterations_ica)
{
```

```
res1 <-
fastICA(RawData,number_components)

compICA <- matrix(res1$S, nrow =
rowNumbers, ncol = number_components)

source("cumulant_4.R")
c4[itr,] <- cumulant_4(t(compICA))
}
```

The library fastICA in R is used – it has the fastICA function to perform ICA. We are performing ICA several times: this number is input in 'iterations_ica', The number of ICA components is input in 'number_components'. 'c4' is initialized as a matrix that holds the fourth cumulants over several iterations for all rows. 'res1' holds the results of ICA. 'compICA' holds the sources (ICA components), which is returned by 'res1$S'. Then our function, cumulant_4, is called, and its return value populates 'c4' for each iteration in the 'for' loop.

If, for example, we consider the threshold as 0.000001 (though the threshold is learned), a simple way of classification based on the threshold is as follows, for illustration. The value ICA components may be negative because ICA does not consider sign. So, we use absolute values. As per our algorithm, we take the maximum of the fourth cumulants ('cum4_20_max'). This is because we want to discover the most non-gaussian value. We compare this value with the threshold and perform the binary partition, populating the classes in 'cum4_20_class'.

```
cum4_20_max <- c()
cum4_20_class <- c()

for (i in 1:4600)
{
cum4_20_max <- c(cum4_20_max,
max(abs(cum4[i,])))

if(abs(cum4_20_max[i]) > 0.000001){
cum4_20_class <- c(cum4_20_class,1)
}
else
{
cum4_20_class <- c(cum4_20_class,2)
}
}
```

4 RESULTS

The results for prediction of one step are given in Table 1.

Observe that in high-dimensional data, such as ours, baseline ARMA has been outperformed by component-based methods in one-step prediction. When working with components, ICA performed better than PCA in prediction. However, we are careful not to generalize our claims about prediction for all kinds of datasets; it may not be true for low-dimensional datasets or highly sparse datasets.

Table 1: One-step prediction.

	Mean Square Error
Baseline ARMA	0.0001056352
Multivariate ARMA on PCA	0.0001067322
Multivariate ARMA on ICA	0.0000965637

The results for prediction of ten steps are given at in Table 2.

Table 2: Ten-step prediction.

	Mean Square Error
Baseline ARMA	0.0008024023
ARMA on PCA	0.0008043217
ARMA on ICA	0.0007754946

The Confusion Matrix revealed the classification performance given in Table 3.

Table 3: Classification performance.

	Overall Accuracy	F1 score
PCA followed by kmeans	0.3692469	0.3542
ICA followed by kmeans	0.4170153	0.4294
ICA followed by PAM	0.6188982	0.7641
ICA followed by our algorithm	0.8019526	0.8901

PCA may not have performed well in classification because the data is time series (has a temporal dimension). ICA performed better than PCA in classification, following the theory that ICA finds true statistical independence rather than working on covariance only as in PCA.

PAM finds the medoids, the series that actually exist, rather than means that are not actual observations. Another benefit is that it is not affected by extremes. This method performed better than k-means in classification.

Importantly, our semi-supervised method works the best among these options for classification because we are able to tune (a single parameter) based on the value of the fourth cumulant. Theoretically this is sound because ICA works on this principle too. In real life, the 'labels' in the dataset for classification were provided by medical experts who may have applied some threshold in their mind while partitioning the data. Therefore, learning this threshold and basing it in line with theory is a good idea. Hence, it is reasoned that our algorithm performed well in classification.

5 CONCLUSIONS

High-dimensional data require dimensional reduction techniques for which PCA is usually considered suitable. ICA has not been much used for time series data. If we are required to perform classification tasks on high-dimensional data, we would need to perform dimensional reduction first. We found evidence that ICA can indeed provide better classification than PCA. One of our contributions is that we have found that a careful choice of the clustering algorithm (PAM instead of k-means) also leads to better performance. Our most important contribution is that we have developed a new algorithm that works on semi-supervised learning. We have applied it on multiple ICAs for more stable results. The new algorithm has provided the best classification performance. The limitation of this work is that we do not generalize to all kinds of datasets. Datasets that are in low dimensions and have many columns that are highly sparse may not yield good results using ICA. On the overall, this work provides an additional method that uses ICA, and may work very well on high-dimensional datasets. Future work may explore many more types of datasets for possible generalization, though our work provides good indications of better performance in higher dimensions. Dimension reduction is very important to visualization of high-dimensional data, so it is possible that future work may consider using similar approach to improve visualization.

REFERENCES

Aggarwal, C. C., 2014. An Introduction to Cluster Analysis. In C. C. Aggarwal, & C. K. Reddy, *Data Clustering: Algorithms and Applications* (pp. 1- 27). Boca Raton, FL: CRC Press.

Aggarwal, C. C., Procopiuc, C. M., Wolf, J. L., Yu, P. S., & Park, J. S., 1999. Fast Algorithms for Projected Clustering. *Proceedings of ACM International Conference on Management of Data (SIGMOD)*, (pp. 61-72). Philadelphia, PA.

Aggarwal, C., Han, J., Wang, J., & Yu, P., 2003. A Framework for Clustering Evolving Data Streams. *VLDB Conference*.

Agrawal, R., Gehrke, J., Gunopulos, D., & Raghavan, P., 1998. Automatic Subspace Clustering of High Dimensional Data. *Proceedings of ACM International Conference on Management of Data (SIGMOD)*, (pp. 94-105). Seattle, WA.

Alelyani, S., Tang, J., & Liu, H., 2014. Feature Selection for Clustering: A Review. In C. C. Aggarwal, & C. K. Reddy, *Data Clustering - Algorithms and Applications* (pp. 29-60). Boca Raton, FL: CRC Press.

Belouchrani, A., et al., 1997. A Blind Source Separation Technique Using Second-Order Statistics. *IEEE Transactions on Signal Processing*, vol. 45, no. 2, pp. 434–44, doi:10.1109/78.554307.

Berkhin, P., 2006. A Survey of Clustering Data Mining Techniques. In J. Kogan, C. Nicholas, & M. Teoulle, *Grouping Multidimensional Data* (pp. 27-71). Berlin Heidelberg: Springer.

Cardoso, J.-F. & Souloumiac, A., 1993. Blind beamforming for non Gaussian signals. *IEE Proceedings-F*, 140, 362–370.

Chakrabarty, S. & Levkowitz, H., 2019. Denoising and stability using Independent Component Analysis in high dimensions – visual inspection still required. in *23rd International Conference Information Visualisation*, Paris, 2019.

Chakrabarty, S. & Levkowitz, H., 2019. A New Index for Measuring Inconsistencies in Independent Component Analysis Using Multi-sensor Data. In: Luo Y. (eds) Cooperative Design, Visualization, and Engineering. CDVE 2019. *Lecture Notes in Computer Science*, vol 11792. Springer, Cham.

Chakrabarty, S., 2018. Clustering Methods in Business Intelligence. in *Global Business Intelligence*, J. M. Munoz, Ed., New York, Routledge, pp. 37-50.

Chapelle, O., Scholkopf, B., & Zien, A., 2006. *Semi-Supervised Learning*. MIT Press.

Comon, P. & Jutten, C., 2010. *Handbook of Blind Source Separation*, Burlington, MA: Academic Press.

Dempster, A. P., Laird, N. M., & Rubin, D. B., 1977. Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society*, 39(1), 1-38.

Ester, M., Kriegel, H. P., Sander, J., & Xu, X., 1996. A Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. *ACM KDD Conference*, (pp. 226-231).

Filiponne, M., Camastra, F., Masulli, F., & Rovatta, S. 2008. A Survey of Kernel and Spectral Methods for Clustering. *Pattern Recognition*, 41(1), 176-190.

Himberg, J. & Hyvärinen, A., 2003. Icasso: software for investigating the reliability of ICA estimates by clustering and visualization. in In *Proc. 2003 IEEE Workshop on Neural Networks for Signal Processing* (NNSP2003), Toulouse, France.

Hyvärinen, A. & Oja, E., 1997. A Fast Fixed-Point Algorithm for Independent Component Analysis. *Neural Computation*, vol. 9, pp. 1483-1492.

Hyvärinen, A. & Oja, E., 2000. Independent Component Analysis: Algorithms and Applications. *Neural Networks*, vol. 13, no. 4-5, pp. 411-430.

Jain, A. K., Murty, M. N., & Flynn, P. J., 1999. Data Clustering: A Review. *ACM Computing Surveys* (CSUR), 31(3), 264-323.

Kaufman, L., & Rousseeuw, P. J., 2005. *Finding Groups in Data - An Introduction to Cluster Analysis*. Hboken, New Jersey: John Wiley & Sons, Inc.

Lee, T. W., Girolami, V. and Sejnowski, T. J., 1999. Independent Component Analysis Using an Extended Infomax Algorithm for Mixed Sub-Gaussian and Super-Gaussian Sources. *Neural Computation*, vol. 11, no. 2, pp. 417-441.

Liao, T., 2005. Clustering of Time Series Data - A Survey. *Pattern Recognition*, 38(11), 1857-1874.

Liu, H., & Motoda, H., 2007. *Computational Methods of Feature Selection*. Boca Raton, FL: CRC Press.

Liu, H., & Yu, L., 2005. Towards integrating feature selection algorithms for classification and clustering. *Knowledge and Data Engineering, IEEE Transactions on*, 17(4), 502.

MacQueen, J. B., 1967. Some Methods for classification and Analysis of Multivariate Observations. J. B. MacQueen: "Some Methods for classification and *Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability* (pp. 281-297). Berkeley: University of California Press.

Meinecke, F., Ziehe, A., Kawanabe, M. and Müller, K.-R., 2002. A Resampling Approach to Estimate the Stability of One-Dimensional or Multidimensional Independent Components. *IEEE Transactions on Biomedical Engineering*, vol. 49, no. 12.

Miettinen, J., Nordhausen, K. and Taskinen, S., 2017. Blind Source Separation Based on Joint Diagonalization in R: The Packages JADE and BSSasymp. *Journal of Statistical Software*, vol. 76, pp. 1-31.

Palmer, J. A., Kreutz-delgado, K. and Makeig, S., 2011. AMICA: An Adaptive Mixture of Independent Component Analyzers with Shared Components. [Online]. Available: https://sccn.ucsd.edu/~jason/amica_a.pdf.

Roweis, S. T., & Saul, L. K., 2000. Nonlinear Dimensionality Reduction by Locally Linear Embedding. *Science*, 290(5500), 2323-2326.

Schaeffer, S., 2007. Graph Clustering. *Computer Science Review*, 1(1), 27-64.

Shi, X., 2011. *Blind Signal Processing, Shanghai*: Springer Jiao Tong University Press.

Tong, L., Soon, V., Huang, Y. and Liu, R., 1990. AMUSE: a new blind identification algorithm. in *IEEE International Symposium on Circuits and Systems*.

Wang, F., & Sun, J., 2012. Distance Metric Learning in Data Mining. *SDM Conference* (Tutorial).

Wang, W., Yang, J., & Muntz, R., 1997. Sting: A Statistical Information Grid Approach to Spatial Data Mining. *VLDB Conference*.

Xiong, H., & Li, Z., 2014. Clustering Validation Measures. In C. C. Aggarwal, & C. K. Reddy, *Data Clustering - Algorithms and Applications* (pp. 571-605). Boca Raton, FL: CRC Press.

Yi, B. K., Sidiropoulos, N. D., Johnson, T., Jagadish, H., Faloutsos, C., & Biliris, A., 2000. Online Data Mining for Co-evolving Time Sequences. *ICDE Conference*.

Ziehe, A. & Muller, K.-R., 1998. TDSEP - an e□cient algorithm for blind separation using time structure. in *International Conference on Artificial Neural Networks*, ICANN98, Skovde, Sweden.