Proceedings of the 5th WSEAS International Conference on Signal Processing, Istanbul, Turkey, May 27-29, 2006 (pp238-242)

Automated BSS-Algorithm Performance Evaluation

MARINA CHARWATH,[1] IMKE HAHN,[1] SASCHA HAUKE,[1] MARTIN PYKA,[1] SLAWI STESNY,[1]
DIETMAR LAMMERS,[1] STEFFEN WACHENFELD[1]
and MARKUS BORSCHBACH[1&2]

[1]Dept. of Mathematics and Natural Science, Institute for Computer Science,
University of Münster, Einsteinstr. 62, D-48149, GERMANY
http://cs.uni-muenster.de/u/flyer/

[2]Chair of Distributed and Self-organizing Computer Systems,
University of Chemnitz, Strasse der Nationen 62, D-09107, GERMANY

Abstract: A first step to perform a competition of methods for source separation is the development of a testsuite that supports the development and evaluation of blind source separation (BSS) algorithms in a highly automated way. The concept of our testsuite is presented and it is shown how the testsuite can be used to apply BSS-algorithms to four standard sub-problems. To compare the performance of arbitrary algorithms on given problems the testsuite allows the integration of new algorithms and testing problems using well defined interfaces. A brief example is given by the integration of the FlexICA, EVD, EVD24 and the FastICA algorithm and our results achieved from automated tests and parameter optimizations.

Key-Words: Blind Signal Separation, Automated Evaluation.

1 Introduction

The development of new blind source separation algorithms is accompanied by the constant need of evaluating and comparing the newly created algorithms with existing ones. The comparison of algorithms in an objective manner requires test conditions - such as the test-problem itself and the performance measures - to be fixed. Furthermore, the quality of each algorithm depends on the algorithm's parameters and the performed steps of pre-processing and noise-addition, see [8], [4] for instant. To address this, we developed a MATLAB-based testsuite that allows an automated performance comparison of different algorithms under defineable testing conditions. Combinations of pre-processing steps such as PCA or high-/low-pass filtering and different noise models are automatically tested. For each combination the parameter range of the algorithms can be searched for optima. In order to determine optimum outcomes, different performance measures can be used. In the next section, we shortly present the concept of our testsuite and how it can be used for automated algorithm evaluation. In section 3 we demonstrate the evaluation and comparison of the FlexICA [5], EVD, EVD24 [6] and FastICA [6] algorithms. The section as well contains the achieved performance results of these algorithms for the four sub-problems:

1. Large scale problem
2. Incomplete or reduced set data problem
3. Very ill-conditioned problem
4. Noisy problem.

2 Testsuite for algorithm evaluation and comparison

The generation of test-data and the application, evaluation and comparison of algorithms is normally performed in different stages. Accordingly, our testsuite consists of multiple stages as shown in figure 1. At each stage of the testsuite, interfaces exist to allow for the addition of new signal mixing methods, new pre-processing steps, different noise models, sample reduction methods, signal separation algorithms and custom performance measures.
The different stages and the currently integrated modulesfunctionalities at each stage will be described briefly in the following.

Generation stage
The generation stage consists of signal mixing and noise addition modules. The signal mixing modules allow to mix any number of independent sources S according to the linear standard ICA model $X = AS$ [1] [7]. The testsuite supports any sources in MATLAB vector format and allows to synthetically generate sources or import *.wav* audio files using an integrated dataset creation tool.
The mixing matrix A can be explicitly given or randomly created. Furthermore, the use of Hilbert matrices for any dimension is possible. Other matrix creation methods can be added if needed. At the generation stage the

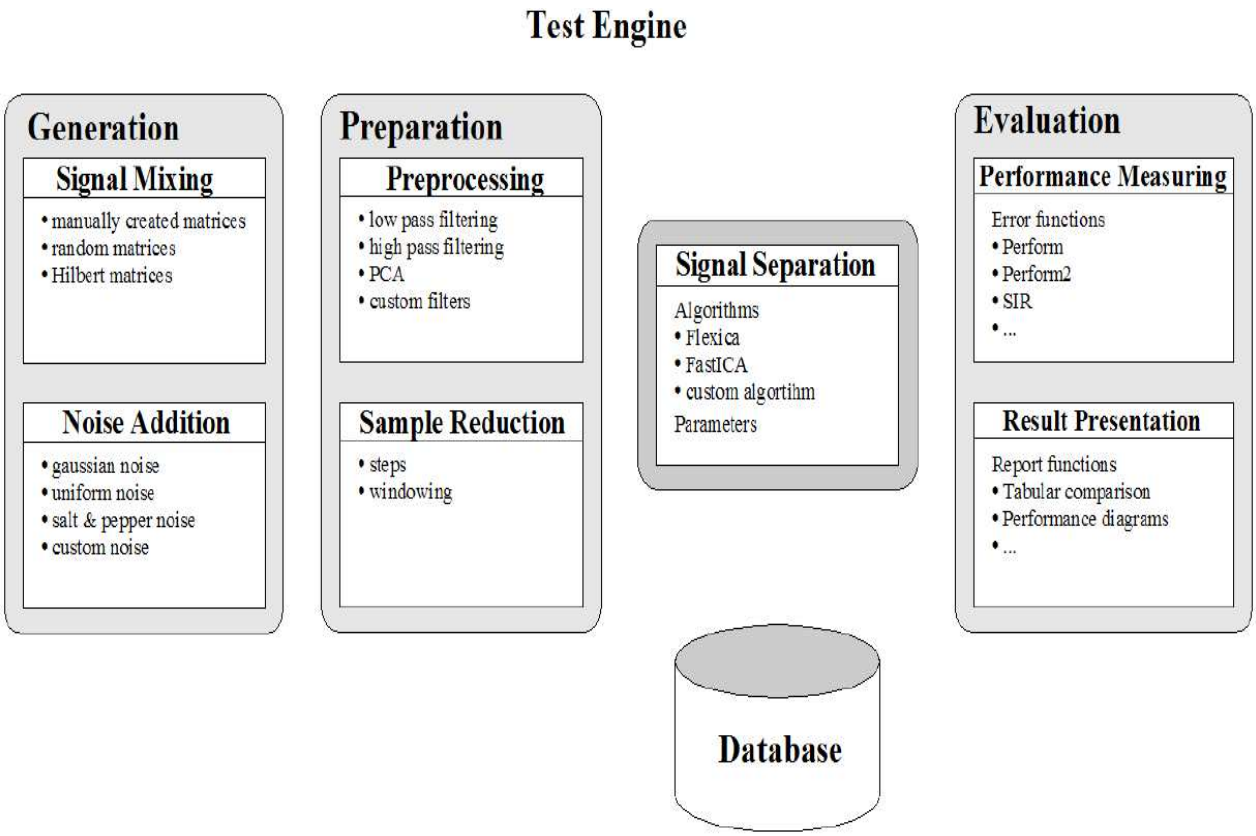

Fig. 1: Stages of algorithm evaluation and comparison

addition of noise can be performed so that the mixing model is extended to $X = AS + N$. The added noise N is generated according to a noise model which is currently selectable from Gaussian-noise, Uniform-noise and Salt-and-Pepper-noise. For each noise model, the number of affected samples can be adjusted and a target signal to noise ratio (SNR) can be specified in *dB*. Other noise models can easily be added if wanted.

Preparation stage
The preparation stage offers signal pre-processing and sample reduction functionalities. Before executing the main signal separation algorithms the application of pre-processing steps such as principal component analysis (PCA) and signal filtering is possible. In addition to the currently implemented low- and high-pass filters custom filters can be easily integrated. A windowing technique allows for limiting the amount of considered data samples. Only samples inside the window are used for blind signal separation.

Separation stage
Within the separation stage the integrated blind source separation algorithms are applied to the generated and prepared data sets. The algorithms can be called with different combinations of their individually required parameters. The possibility to call the algorithms with varied parameters allows for systematical parameter testing and optimization. Newly developed algorithms can be added in an easy and comfortable way. We provide a tutorial for developers on how to integrate new algorithms into the application [9].

Evaluation stage
The evaluation stage consists of performance measuring and result representation functionalities. The performance of algorithms is being determined according to several measures, such as signal-to-interference ratio (SIR) or intersymbol interference (ISI, called Perform in the test-suite) [2]. Further performance measures can be added as required. Results are being presented primarily in a tabular form, with other representations, such as box plots [3], optionally available. As with the other stages, further methods of presenting the data can be integrated if needed. All test results can automatically be written to a SQL database during the test. This allows for data analysis with powerful external tools.

Test Engine
The test engine controls all settings and parameters passed

to the modules during the test cycle.This helps to find optimal settings and parameters in a systematic fashion. Several steps in determining the algorithms' best performance are automated. These include progression of noise intensity and sample reduction.
Noise progression is done by successively increasing the signal-to-noise ratio (SNR) and per cent amount of noise of a mixture up to a maximum. Sample reduction is done by gradually decreasing the size of the window used by the algorithm. This shows how the algorithms' performance is affected by random noise or the amount of data available, respectively. Noise progression and sample reduction are mutually exclusive. Automation of parameter values are also integrated into the test engine.
Results can be stored in a SQL database, as mentioned before, as well as in *.mat* files for further performance analysis during the test-cycle and between the steps. This allows for the reduction of signal generation and modification to a single time and reusing the generated and preparated data with different algorithms. This feature has been implemented using the MATLAB database toolbox.
Finally, the test engine will generate a result report showing the pertinent data. This includes options to visualize the results, such as data sample plots of source, mixed and unmixed data. Additionally, boxplots of performance measures for the different algorithms are supplied. This is done in order to improve general comparability of different algorithms.

3 Evaluation results

Using the developed testsuite, the following four algorithms have been tested:

- The Flexica algorithm developed by S. Choi, A. Cichocki and S. Amari [5]
- The EVD algorithm developed by P. Georgiev and A. Cichocki [6]
- The EVD24 algorithm developed by P. Georgiev and A. Cichocki [6]
- The FastICA algorithm developed by J. Hurri, H. Gvert, J. Srel, and A. Hyvrinen [6]

All of these have been applied to the four subproblems. The results the algorithms yielded will be presented in the following sections. A general observation was that high pass filtering yielded better results than unfiltered or low pass filtered data. Therefore, all datasets have been high pass filtered before applying the respective algorithms. All synthetic datasets have been created using the testsuite's integrated data generation module. The figures given in the next sections represent the mean values of the SIR over at least 100 runs using randomly generated mixing matrices. This was done to guarantee stable results in a Monte Carlo fashion.

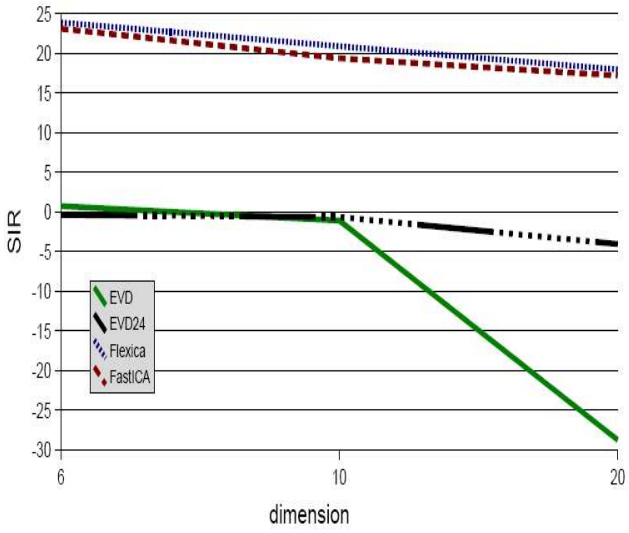

Fig. 2: SIR global performance index for the large scale sub-problem 1 (set 2) using synthetic data

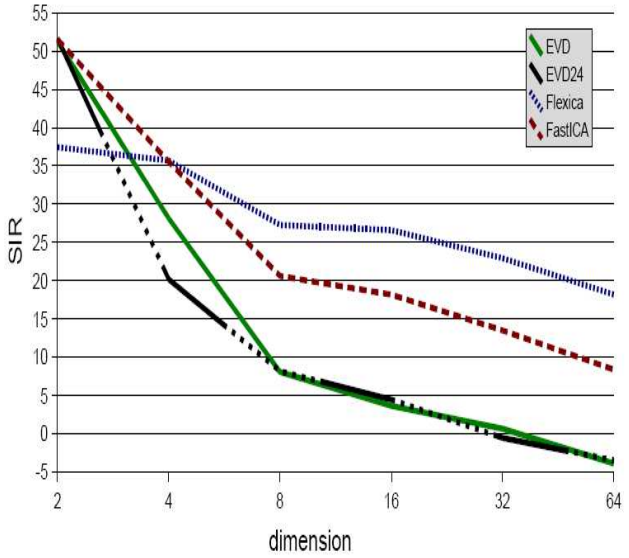

Fig. 3: SIR global performance index for the large scale sub-problem 1 (set 2) using real world speech data

3.1 Large scale problem

The first sub-problem deals with the algorithms' performance given an increasing dimension of the data set. The data sets used for testing contain an equal number of sub- and super-Gaussian sources, e.g. for dimension 6 the data set would contain 3 sources of each type. With increasing dimension the SIR drops, as illustrated in figure 2 for synthetic data and figure 3 for real world speech data. It can be

noticed that the two EVD variants fare poorly compared to FastICA and Flexica. For synthetic data they are basically unable to separate the mixtures. FastICA and Flexica on the other hand are performing better. Up to 20 sources can be separated without the SIR falling below 15 *dB*.
For real world data (see figure 3) the algorithms perform better than for the synthetic sources. This can probably be attributed to the super-Gaussian nature of the signals. Flexica yields the best performance, being able to separate up to 64 different sources.

3.1.1 Incomplete or reduced set data problem

The second sub-problem is concerned with the reduction of samples available to the algorithms. Algorithmic performance in this task can be determined using the test engine's sample reduction feature.The datasets are being reduced in length successively, the algorithms attempt to estimate the inverse mixing matrix using fewer samples each step. This is done fully automated by our testsuite.
Figure 4 shows the performance of three algorithms trying to separate 50 synthetically generated sources of decreasing sample count. FastICA was not able to complete this task in a stable way and was therefore not included. The EVD variants were not able to successfully seperate any of the sources, and even the Flexica algorithm never reached a SIR $>$ 15 *dB*, even at full sample count of 5000. The

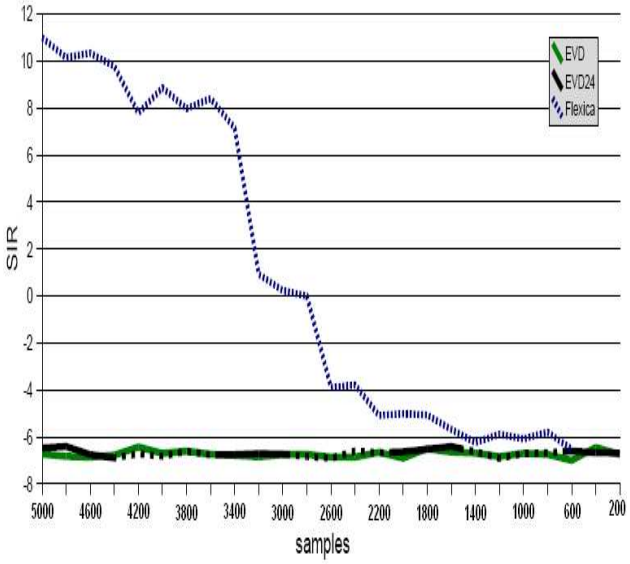

Fig. 4: SIR global performance index for the incomplete/reduced sub-problem 2 (set 1) using synthetic data

performance for real world data is somewhat better for the Flexica algorithm, as displayed in figure 5. Even with only 3200 available samples it still reaches a SIR of approximately 15 *dB*. EVD and EVD24 show unusual behavior, as their performance improves significantly beginning at 2800 samples. The cause of this has yet to be determined.

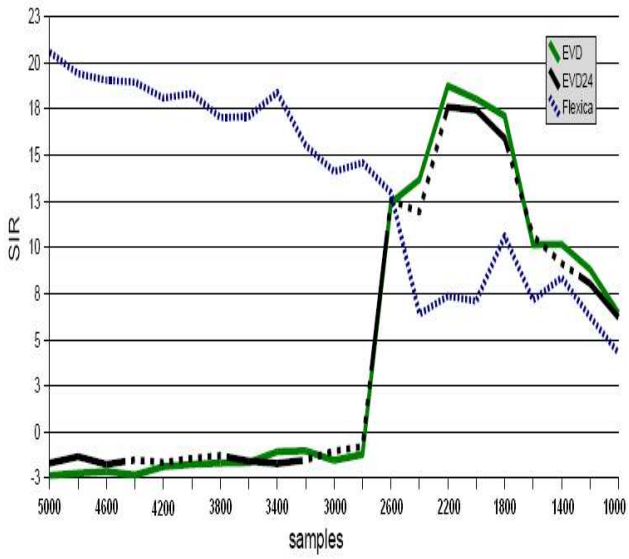

Fig. 5: SIR global performance index for the incomplete/reduced sub-problem 2 (set 2) using real world speech data

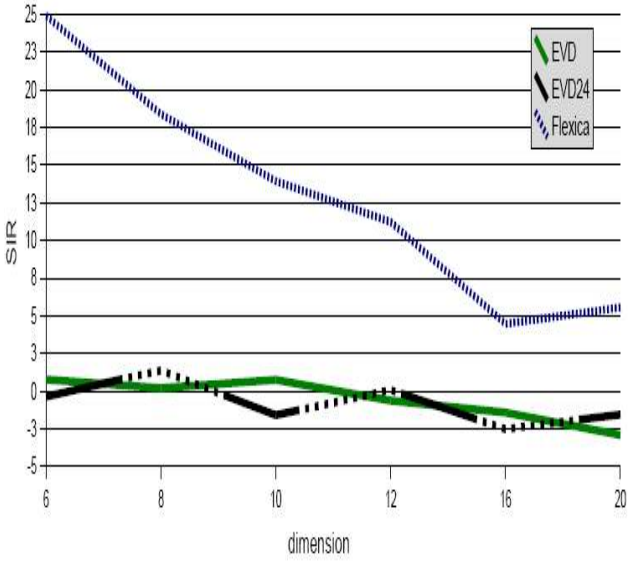

Fig. 6: SIR global performance index for the very ill conditioned sub-problem 3 (set 1) using synthetic data

3.1.2 Very ill-conditioned problem

Sub-problem three uses Hilbert matrices in the mixing process. Due to the nearly singular nature of these matrices, inversion becomes more difficult. This is mirrored by the results given in figure 6. Among the tested algorithms, only Flexica manages to provide acceptable results.
For low dimensions below ten sources, the SIR is higher than the required 15 *dB*. While this is significantly more than for the EVD variants, it still is not a truly satisfactory performance. Overall these results are congruent with those of the previous tests and clearly show the inabilty of EVD to deal with datasets comprised of both sub- and

super-Gaussian sources.

3.1.3 Noisy problem

The fourth and final sub-problem adds successively increasing noise levels in the mixing process. This has been achieved by using the noise progression module provided by the test engine. This simplified testing the algorithms. For this sub-problem, only the performance of the Flexica algorithm was of interest as the EVD methods did not manage to separate the mixtures. Even Flexica's performance was not remarkable, not reaching the required SIR of 15 *dB* with synthetic data at any noise level.

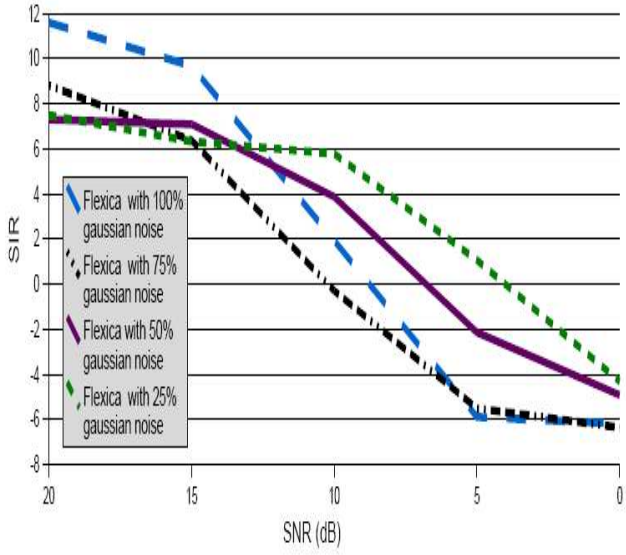

Fig. 7: SIR global performance index for the noisy sub-problem 4 (set 1) using synthetic data

With speech data, Flexica's performance is somewhat better as can be seen in figure 8. It remains satisfactory up to a SNR level of 5 *dB* with 50% noise. This could be attributed to the super-Gaussian nature of the speech signals. As witnessed in all other sub-problems as well, both EVD algorithms are unfit to address the tasks given.

4 Conclusion

We have shown that our testsuite allows for the systematic and automated evaluation of blind source separation algorithms and supports the development of new algorithms by providing an easy way of standardized comparison. For the given test problems we have determined an optimal combination of preprocessing steps and corresponding optimal parameters for selected known algorithms. The systematic search for optimal parameters may allow using the full potential of existing and new algorithms.

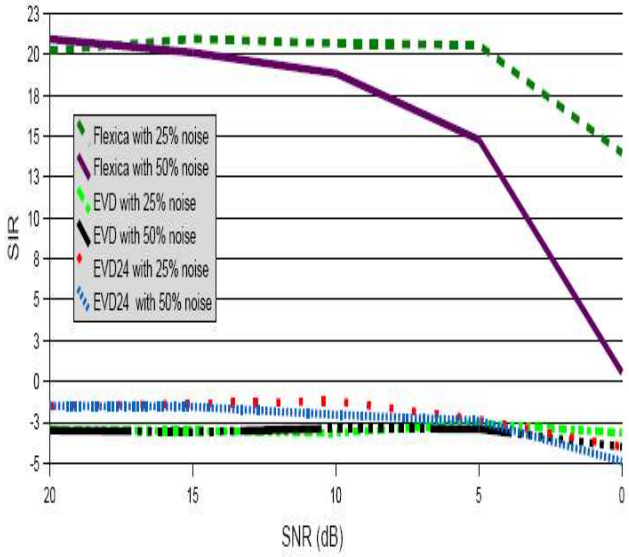

Fig. 8: SIR global performance index for the sub-problem 4 (set 2) using real world data with 25% and 50% uniform noise

References

[1] M. Borschbach and M. Schulte, Performance Analysis of Learning Rules for the Blind Separation of Magnetencephalography Signals, *Proc. of ICA'99, First Int. Workshop on Independent Component Analysis and Signal Separation*, 1999, pp. 341-346.

[2] D. Schobben, K. Torkkola and P. Smaragdis, Evaluation Blind Signal Separation Methods, *Proc. of ICA'99, First Int. Workshop on Independent Component Analysis and Signal Separation*, 1999, pp. 261-266.

[3] X. Giannakopoulos, J. Karhunen and E. Oja, An experimental comparison of neural algorithms for independent component analysis and blind separation, *Int. J. of Neural Systems*, Vol. 9, No. 2, 1999.

[4] V. Calhoun and T. Adali and J. Larsen and D. Miller and S. Douglas, *Proceedings of IEEE Machine Learning for Signal Processing Workshop XV*, 2005.

[5] A. Cichocki, S.-I. Amari, *Adaptive Blind Signal and Image Processing*, Wiley, 2002.

[6] A. Hyvärinen, J. Karhunen, E. Oja, *Independent Component Analysis*, Wiley, 2001.

[7] C. Hagen, *Neural Networks for statistical data analysis* (in German), Shaker, 1997.

[8] X. Giannakopoulos, *Comparison of Adaptive Independent Component Analysis Algorithms*, Helsinki University: Master Thesis, 1998.

[9] *http://cs.uni-muenster.de/ICA/* ;-last date of access 03.03.2006.