Automatic Artifact Removal (AAR) toolbox v1.3 (Release 09.12.2007) for MATLAB

Germán Gómez-Herrero
Tampere University of Technology

December 11, 2007

Abstract

This MATLAB toolbox integrates several state-of-the-art methods for automatic removal of artifacts in the electroencephalogram (EEG). The methods implemented so far are only for removal of ocular (EOG) and muscular (EMG) artifacts. EOG removal methods include regression techniques based on Least Mean Squares (LMS), Recursive Least Squares (RLS) and other adaptive algorithms. However, the core functionality of the toolbox is a general-purpose artifact removal procedure that consists on three steps. First, the EEG data is decomposed into several spatial components using Blind Source Separation (BSS). Second, a suitable criteria is used to automatically detect artifact-related components. Third, the EEG data is reconstructed using only non-artifactual components. The toolbox is designed so that the user can easily expand it by adding new BSS algorithms and new criteria for detecting artifactual components. Furthermore it can be easily integrated as a plug-in into EEGLAB, which is a very popular graphical toolbox for EEG analysis and visualization in MATLAB.

Contents

1 License and disclaimer

Rights, ownership, and copyright information related to each MATLAB file included in the AAR toolbox are individually stated in each file. This information can also be found in the document `readme.aar.txt`, which is included in the public release of this software. The current release is relatively stable and many bugs have been corrected. However, it is intented to be used for research and testing purposes only. No claims are made as to the validity of the methods or the correctedness of the toolbox code and documentation. Bugs and suggestions can be reported to `german.gomezherrero@tut.fi`.

2 Conventions

Names of files and folders as well as MATLAB commands are typeset in a `typewriter font`. Any word typeset in `typewriter font` and starting with the symbol `$` represents a non-literal string, i.e. a string whose value might be different for different operating systems, MATLAB versions, etc. For instance, `$MATLABROOT` denotes the folder where MATLAB is installed.

3 Installation

Note that the current release of the toolbox (v1.3, release 04.12.2007) has only been tested under MATLAB 7.4.0 (R2007a) and it might not work properly on older versions of MATLAB. Some features of the toolbox (e.g. the *emg_psd* criterion) require MATLAB's Signal Processing Toolbox v6.2 or newer. Such version of the Signal Processing Toolbox is usually included in MATLAB 7.0 (R14) and newer MATLAB releases.

The toolbox is installed following these steps:

1. Start MATLAB.
2. Download and install EEGLAB for MATLAB [6]. For information on this step please visit EEGLAB's homepage [1].
3. Download the most recent AAR release (`aarcode.zip`) from the Internet [2].
4. Create the folder `$EEGLABROOT\plugins\aar1.3` where `$EEGLABROOT` denotes the folder where EEGLAB was installed.

[1] `http://www.sccn.ucsd.edu/eeglab/`

[2] `http://www.cs.tut.fi/~gomezher/projects/eeg/aar/aarcode.zip`

Algorithm	**MATLAB files**	**Ref.**	**Strengths**	**Weaknesses**
LMS	`lms_regression` `pop_lms_regression`	[9]	Simplicity Stability	Slow conv.
RLS	`crls_regression` `pop_crls_regression`	[10]	Fast conv.	Unstability
Stable RLS	`scrls_regression` `pop_scrls_regression`	[14]	Fast conv. Stability	Comp. time

H^∞	`hinftv_regression`
	`pop_hinftv_regression`
	`hinfew_regression`
	`pop_hinfew_regression`
		[16]	Fast conv.
			Accuracy
				Unstability
				Comp. time

Table 1: The regression algorithms in a nutshell

5. Add folder `$EEGLABROOT\plugins\aar1.3` to Matlab's path.
6. Uncompress `aarcode.zip` into `$EEGLABROOT\plugins\aar1.3`.
7. From MATLAB's command window start EEGLAB by executing `eeglab`. Under the "Tools" menu there should be a sub-menu named "Automatic artifact removal" which contains the AAR functions.

4 EOG removal using regression

The current version of the toolbox (v1.3) includes four adaptive algorithms for EOG removal using one or more EOG regression channels. The implementations of those algorithms are not optimized for speed so you should expect a relatively large computation time. The first algorithm is based on classical Least Mean Squares (LMS) [9], which is very simple and quite stable if a small enough learning step is used. However, small learning steps lead to very slow convergence. The second algorithm is based on Recursive Least Squares (RLS) [9] which offers a much greater convergence speed at the cost of reduced numerical stability. The third algorithm is a numerically stable version of the RLS algorithm [14]. Note that the implementation of this latter algorithm is still very naive and inefficient so the required computation time can be very large. The two last algorithms are based on H^∞ principles [16, 17]. The most important information related to these regression algorithms is summarized in Table. 1.

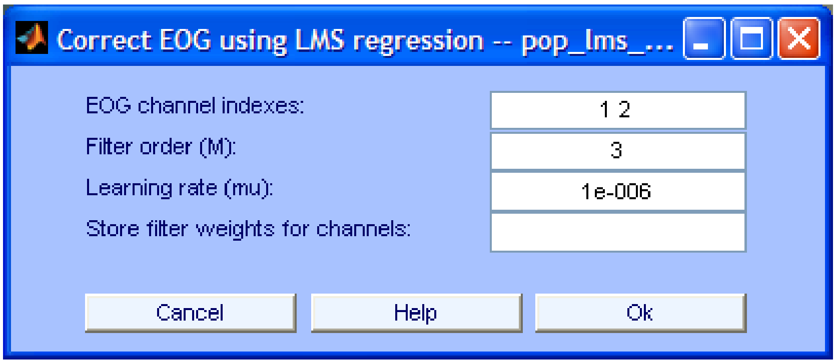

Figure 1: Interface window for EOG removal using LMS regression.

4.1 Least Mean Squares (LMS)

The graphical interface of the algorithm is shown in Fig. 1. The parameters that can be specified in that window are described below:

- *EOG channel indexes.* The indexes of the EEG channels to be used as reference (regression) channels. At least one channel has to be specified.
- *Filter order (M).* The number of taps of the adaptive filter. By increasing the order we might remove more EOG artifacts but we also increase the risk of overcorrection, i.e. removing useful information from the EEG. Also, high filter orders increase the computation time, slow down convergence and might lead to numerical unstability.
- *Learning rate.* Decreasing this parameter slows down convergence but makes the algorithm more stable and viceversa.
- *Store filter weights for channels.* In this field we can specify the indexes of the channels for which the values of the filter weights in each iteration should be tracked. If this field is left empty, tracking is not performed. Note that tracking the filter weights considerably increases the computation time and the memory requirements of the algorithm. The results of the tracking are stored in the field `.Hh` of the current EEGLAB structure. More specifically, the evolution of the i^{th} filter tap for the EEG channel corresponding to the j^{th} index specified in this field is stored in `EEG.Hh(i,j,:)`.

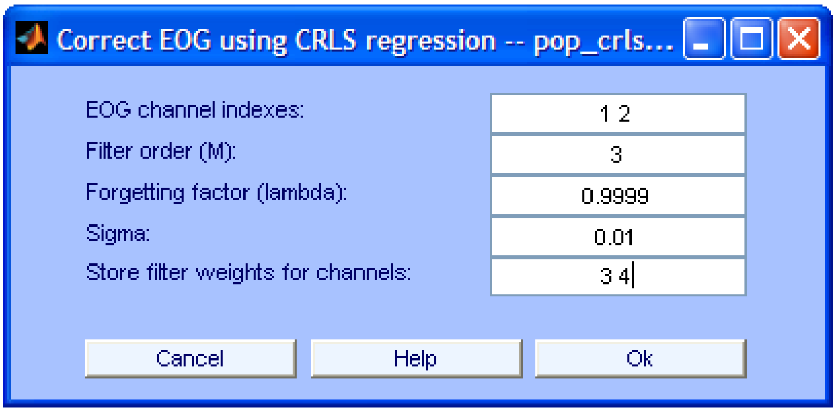

Figure 2: Interface window for EOG removal using the RLS algorithm.

4.2 Conventional Recursive Least Squares (CRLS)

The interface of this algorithm is shown in Fig. 2. Most parameters shown in that interface were already explained when we discussed the LMS algorithm. There are, however, two new parameters:

- *Forgetting factor (lambda)*. This parameter defines how fast the RLS algorithm should forget past data samples. If we set it to 1 the algorithm uses all available data samples to estimate the filter weights. As we decrease this value the contribution of past samples to the weight estimation decreases. This is useful in a non-stationary environment where the spatial pattern of ocular artifacts (and therefore the optimum filter weights) vary considerably in time. By contrary, in a stationary environment, discarding past samples leads to higher errors in the estimation of the optimum filter weights.

- *Sigma*. This parameter defines the initial state of the filter. Please read [10] for details.

4.3 Stable Recursive Least Squares (SRLS)

The algorithm RLS is well-known for its fast convergence but also for its numerical unstability. However, stability of the RLS algorithm can be guaranteed by imposing bounds on the relative precision of the computations performed in the different steps of the algorithm. This is explained in detail in [14]. The current implementation of this algorithm is very slow and therefore we recommend using the conventional RLS algorithm whenever possible.

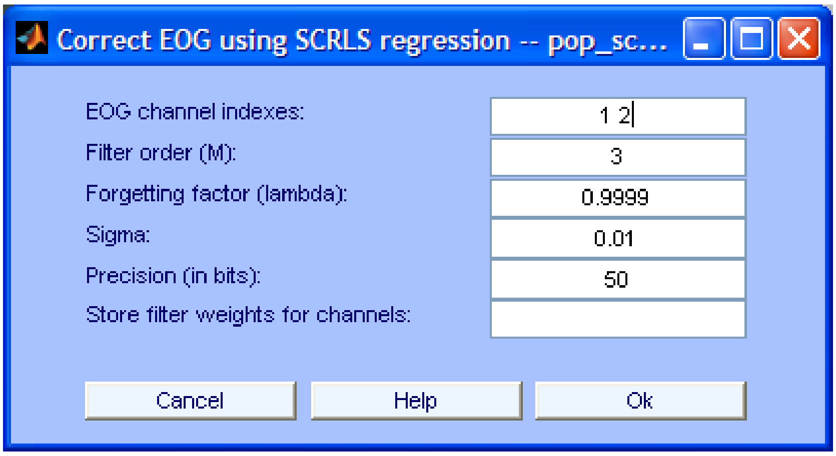

Figure 3: Interface window for EOG removal using the stable RLS algorithm.

Only if the conventional RLS algorithm becomes unstable it is worth trying its stable variant.

The graphical interface of the algorithm looks like Fig. 3. There, we can set up the precision of the computations in bits. Increasing the precision increases the accuracy of the filter weights estimates but also increases the risk of numerical unstability.

4.4 Algorithms based on the H^∞ norm

The toolbox includes two adaptive algorithms (*time varying* and *exponentially weighted*) based on the H^∞ principles for removing EOG artifacts using one or more reference EOG channels. The details of these two algorithms are described in [16], where it was found that H^∞-based algorithms clearly outperformed the LMS algorithm. The graphical interfaces of these algorithms are shown in Fig. 4 and Fig. 5. Their specific parameters are:

- *Distance at t=0 to optimal solution (eta).* This is a positive factor reflecting the a priori knowledge of how close the initial filter weights are to the optimal initial value. It corresponds to parameter $\mathbf{\Pi}_0$ in [16]. Smaller (resp. larger) values of this parameter are suitable when the initial filter weights are believed to be far (resp. close) from its optimal value. However, in most cases, the user does not need to change the default value. This is especially true for the current implementation, which does not let the user set the initial filter weights.

- *Speed of variation of filter coefficients (rho).* This is a positive factor reflecting a priori knowledge about how fast the optimal filter weights

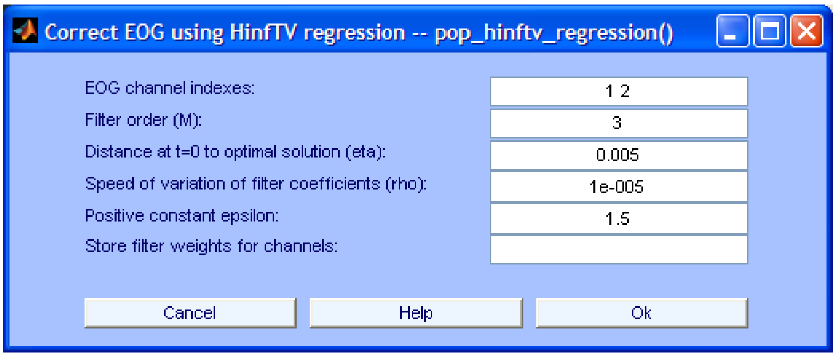

Figure 4: Interface window for EOG removal using the TV H^{∞} norm algorithm

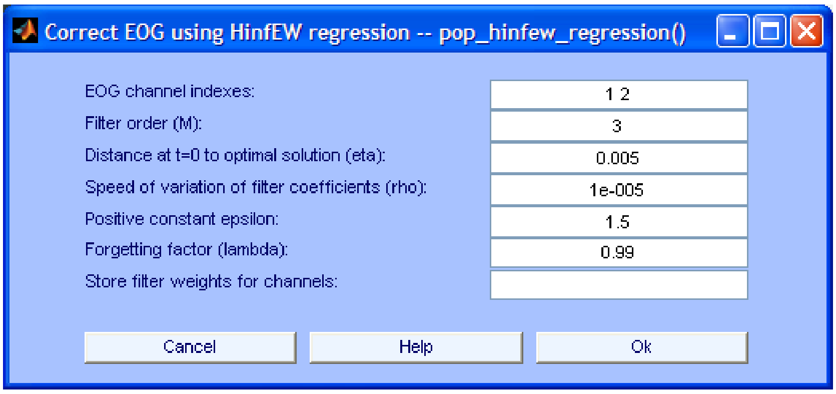

Figure 5: Interface window for EOG removal using the EW H^{∞} norm algorithm

vary with time. If the variation is believed to be slow (resp. fast) a larger (resp. smaller) value might be more appropriate.

- *Positive constant (epsilon)*. The definition of this positive constant can be found in [16]. In general, the default value should work well in most cases.

5 EOG and EMG removal using spatial filters

The toolbox implements a spatial filtering framework for removing different types of artifacts. This framework consists on three basic steps. First, the original EEG data is decomposed into a set of spatial components. Second, artifactual components are identified using a suitable automatic criterion.

Third, the EEG data is reconstructed by projecting back to the electrodes only the non-artifactual spatial components. Within this framework, many artifact removal algorithms can be defined by defining the way the spatial components are estimated and by defining the criterion used for identifying artifactual components. The default installation of the toolbox can decompose the EEG data into a set of spatial components using:

- *iWASOBI* [23]. This is a efficient version of the algorithm WASOBI [24, 20], which is an asymptotically optimal Blind Source Separation (BSS) algorithm for autoregressive (AR) sources. This algorithm is implemented in file `iwasobi.m`, which is owned by Dr. Tichavsky. See iWASOBI's web-page [19] for related license information and to get the latest version of `iwasobi.m`.

- *EFICA* [13]. This is an asymptotically efficient version of the well-known Independent Component Analysis (ICA) algorithm FastICA [11, 18]. Rights and ownership related to the MATLAB file `efica.m` implementing this algorithm are owned by Dr. Koldovsky. Please visit the EFICA download web-page [12] to get the most up to date version of `efica.m`.

- *MULTICOMBI* [22]. This is a BSS algorithm able to simultaneously separate non-Gaussian and time-correlated sources. This algorithm is implemented in file `multicombi.m`, which is owned by Dr. Tichavsky and Dr. Koldovsky. You can visit MULTICOMBI's web-page [21] to get the latest version of `multicombi.m`.

- *FCOMBI* [8]. FCOMBI is a computationally more efficient version of MULTICOMBI. However, FCOMBI is not as stable and reliable as MULTICOMBI.

- *SOBI* [2]. SOBI uses second order statistics to find spatial components that have non-zero time-delayed autocorrelations and zero time-delayed cross-correlations. The toolbox uses the implementation of SOBI included in EEGLAB.

- *RUNICA* [15]. This is the implementation of the ICA algorithm Infomax [1], which is included in the default installation of EEGLAB.

- *JADER* [4]. This is Cardoso's implementation of his well-known ICA algorithm JADE [3]. The algorithm is not included in the toolbox release but the toolbox is able to automatically detect it if it can be found in the MATLAB's path. JADE can be downloaded from ICA Central [4].

- *FastICA* [11, 18]. A very popular ICA algorithm that finds maximally non-Gaussian components. FastICA is not included in the toolbox release but it will be recognized by the toolbox if it can be found in the MATLAB's path. FastICA can be downloaded from the FastICA page at the Helsinki University of Technology [18].

- *BSSCCA*. Canonical Correlation Analysis (CCA), as defined in [5]. It projects the observed EEG data into maximally auto-correlated components.

- *PCA*. Decomposes the data into its principal components.

The criteria that are included in the current version of the toolbox are:

- *eog_fd*. This criterion was proposed in [7]. It marks as artifactual the components with smaller fractal dimension. Conceptually, components with low fractal dimensions are those who are composed of few low-frequency components. This is often the case of ocular activity and therefore this is a suitable criteria for detecting ocular (EOG) components.

- *eog_corr*. This criterion considers as EOG-related those components whose cross-correlation with any of the available reference EOG channels exceed certain threshold.

- *emg_psd*. This criterion considers to be EMG-related those components whose ratio of average power in the typical EEG and EMG bands is below certain threshold.

By default, the toolbox uses a combination of *iWASOBI* and the criterion *eog_fd* to automatically correct EOG artifacts in the EEG. This default combination can be called from the EEGLAB menu "Remove EOG using BSS", which opens a dialog like in Fig. 6. Using the graphical interface, the user can specify the length and shift between correlative analysis windows. A window shift of 1 second and a window length of 2 seconds means that the first analysis window (on which the BSS-based spatial filters will be obtained) will cover the time range 0-2 seconds, the second analysis window will correspond to the time range 1-3 seconds, the third window will cover the time 2-4 seconds and so on. Note that the optimum window temporal range would be that that covers enough data samples to learn the artifactual spatial components and as few as possible neural EEG components. Therefore, there is no easy and objective way of selecting the optimum window length. If our artifacts have relatively stable spatial patterns (e.g. EOG artifacts) a longer window length might be more appropriate. However, a too long window might cause removal of some EEG components due to the low spatial resolution of the EEG recordings. If our artifacts have relatively short duration (e.g. sudden EMG bursts), a short analysis window might be more appropriate.

Additionally, the user can pass extra parameters to the BSS algorithm and to the criterion. A parameter that is common to all BSS algorithms is `'eigratio'`, which determines the number of principal components that will be kept in the pre-processing PCA step which is performed before any BSS algorithm. The number of PCA components will be such that the ratio between the largest and smallest eigenvalue of the PCA-transformed data matrix is below the specified `'eigratio'`. A large `'eigratio'` will most likely not remove any principal component but might lead to numerical unstability of the BSS algorithm due to an ill-conditioned covariance matrix. By contrary, a small value of `'eigratio'` will cause some inaccuracies due to the removal of some principal components but will enforce the data covariance matrix to be well-conditioned. By default the `'eigratio'` is 1e6.

An important option that can be passed to all available rejection criteria is the `'range'` of components that should be removed. That range specifies the minimum and maximum number of components that are to be marked as artifactual in each analysis window. A range like `[5,5]` enforces the rejection of the 5 components ranked as most likely to be artifactual by the respective criterion. Indeed, increasing the number of rejected components allows for removing more EOG but increases the chances of removing also useful EEG activity of neural origin. If the user selects the criterion *eog_corr*, the index of at least a reference EOG channel must be provided through the graphical interface.

EMG correction can be performed by selecting the EEGLAB menu "remove EMG using BSS", which opens an interface window as in Fig. 7. By default, the BSS algorithm used is based on CCA as in [5]. Currently, the only automatic criterion that can be used for detecting EMG components is *emg_psd*. The option `'range'` allows to reject a fixed number of components in each analysis window, as explained above. Another important parameter of the *emg_psd* criterion is the ratio of average power in the typical EEG band and in the typical EMG band. For instance, the option `'ratio',10` makes the criterion to mark as EMG-related only those components having less than 10 times more average power in the EEG band than in the EMG band. The boundary between EEG and EMG bands can be specified using the parameter `'femg'`. The sampling rate of the data also needs to be specified using the parameter `'fs'`. The user can also specify with the parameter `estimator`, the type of spectral estimator that will be used to estimate the power in the EEG and EMG bands. By default the estimator used is a

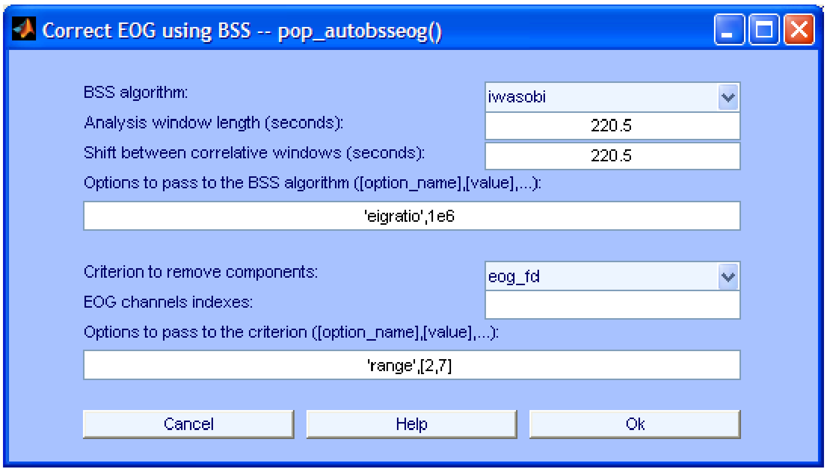

Figure 6: Interface window for EOG removal using spatial filters.

Hamming-windowed Welch periodogram with segment length equal to the analysis window length. You can check the help of the MATLAB function implementing the different criteria and BSS algorithms for more details on the available options.

6 Correction examples

In this section we show few correction examples obtained with a long-term EEG recording from a patient suffering Mesial Temporal Lobe Epilepsy. The data was provided by our collaborators at the Kaholieke Universiteit Leuven (Belgium) and was collected from 21 scalp electrodes placed according to the international 10-20 System with addition electrodes T1 and T2 on the temporal region. The sampling frequency was 250 Hz and an average reference montage was used. The dataset did not have EOG channels and therefore regression-based techniques were not suitable for removing EOG artifacts. Thus, we used the default BSS-based EOG removal algorithm included in the toolbox to suppress as much EOG activity as possible. Subsequently, we used the EMG correction algorithm to remove EMG artifacts. It is important to notice when using any of the BSS-based algorithms included in the toolbox that the corrected dataset will be rank deficient. Also, the order in which the algorithms are applied is important and the results differ depending on the order in which EOG and EMG artifacts are removed. In general, we have observed that slightly better results are obtained when removing EOG artifacts first.

In Fig. 8 we show a frame of original EEG spanning from second 275

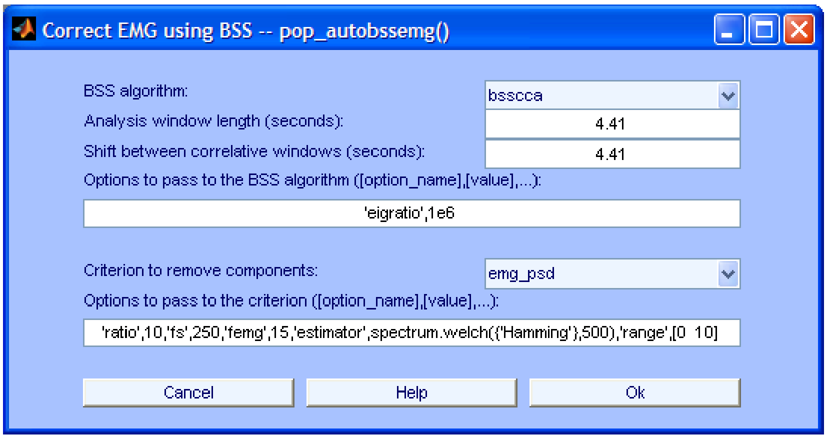

Figure 7: Interface window for EMG removal using spatial filters.

to second 290. Observe that this EEG frame contains just few blinks and almost no EMG artifacts. In Fig. 9 we show the result of the automatic EOG correction algorithm. Notice how the blinks were almost perfectly removed while the clean EEG was not significantly altered. In Fig. 10 we show the result of applying EMG correction algorithm on the EOG corrected dataset. We can observed that Fig. 9 and Fig. 10 are almost identical, which is a desirable result since there was very little EMG activity in this EEG frame.

In Fig 11 we show the EEG frame that covers the temporal range from second 320 to second 335, which corresponds to the first clinical signs of a seizure. This frame is heavily contaminated by EMG artifacts and contains also some ocular artifacts. In Fig 12 is the result of removing the EOG artifacts and in Fig. 13 after removing also the EMG artifacts.

In Fig. 14 appears the EEG from second 380 to second 395. This EEG frame contains little EOG activity but is considerably distorted by several EMG bursts. In Fig. 15 is shown the output of the EOG correction algorithm. As can be observed, the algorithm did not modified significantly the data as was expected. In Fig. 16 appears the results of removing both EOG and EMG artifacts. Notice that most EMG artifacts were removed, while the sharp quality ictal theta activity was preserved.

7 Version history

- **Release 09-12-2007, version 1.3:** minor changes.
  - SOBI is again the default algorithm for BSS-based EOG correction.

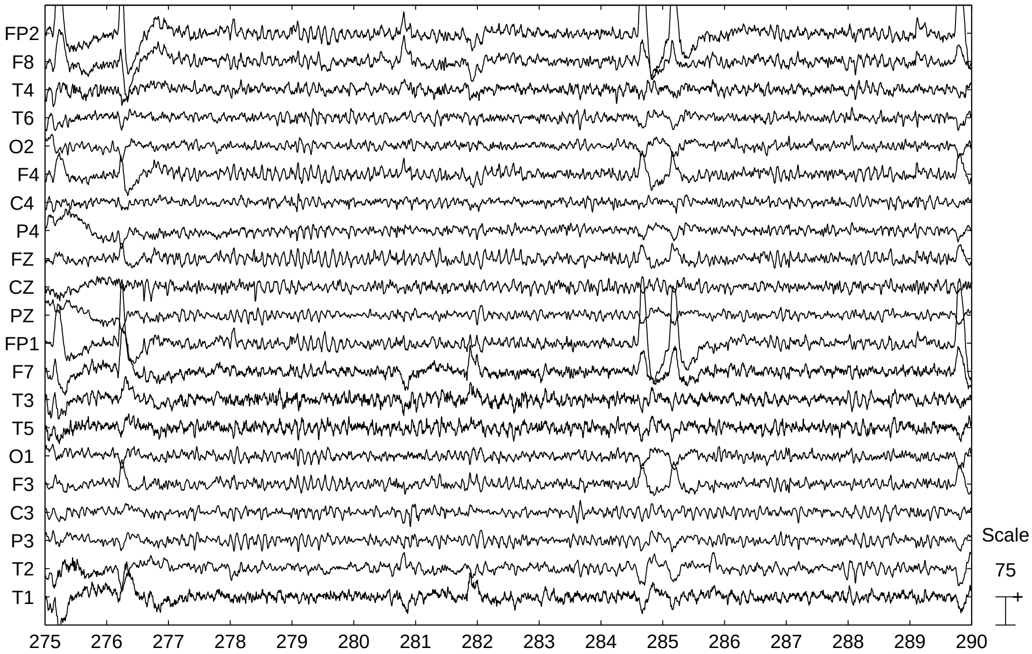

Figure 8: Original EEG frame.

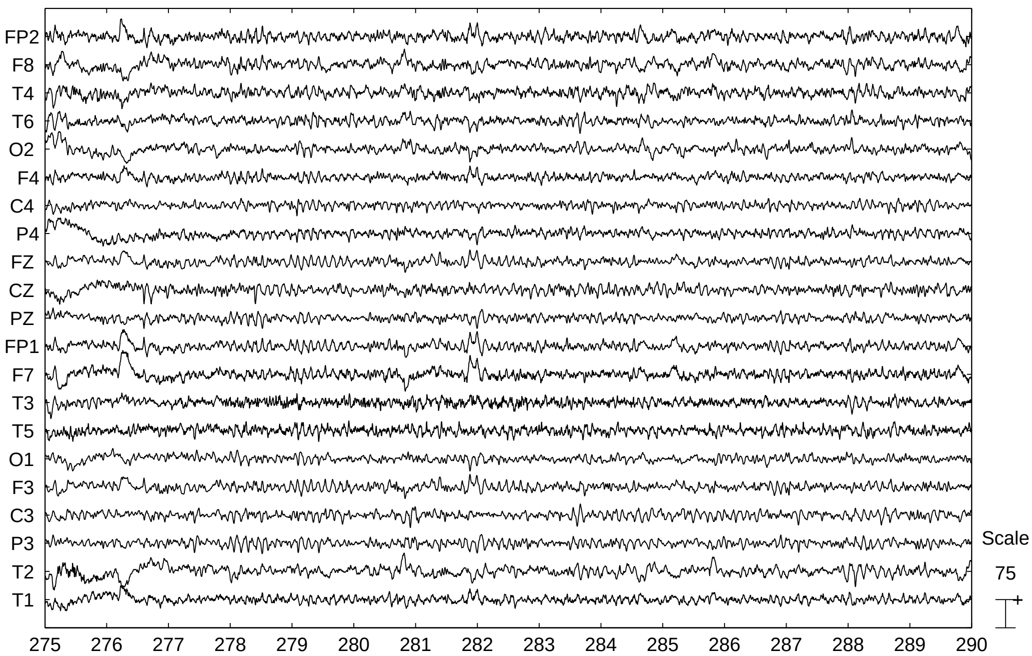

Figure 9: EOG corrected frame.

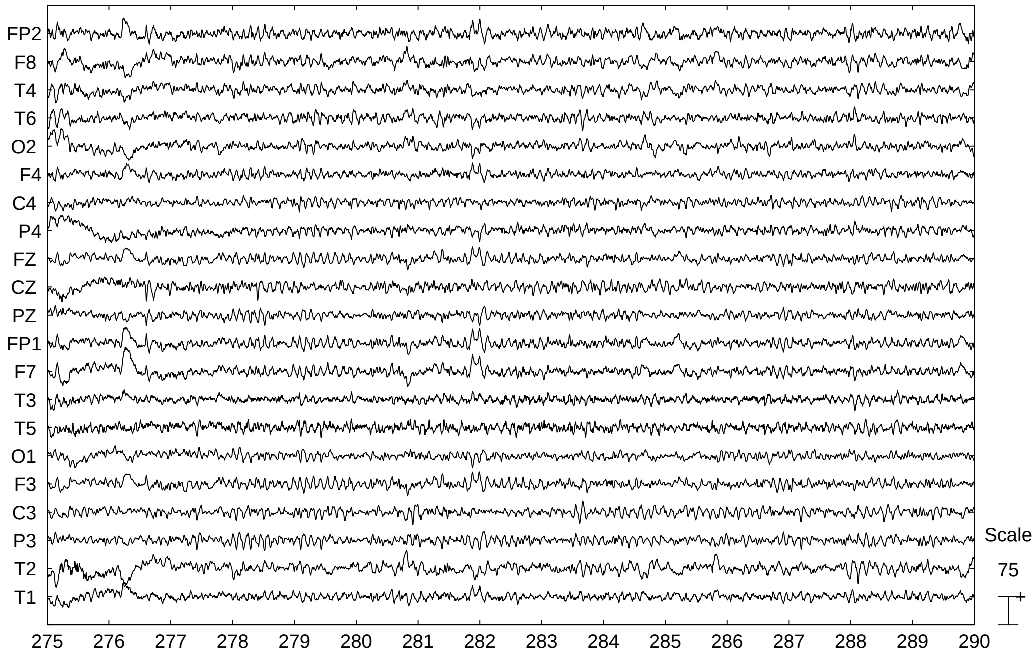

Figure 10: EOG and EMG corrected frame.

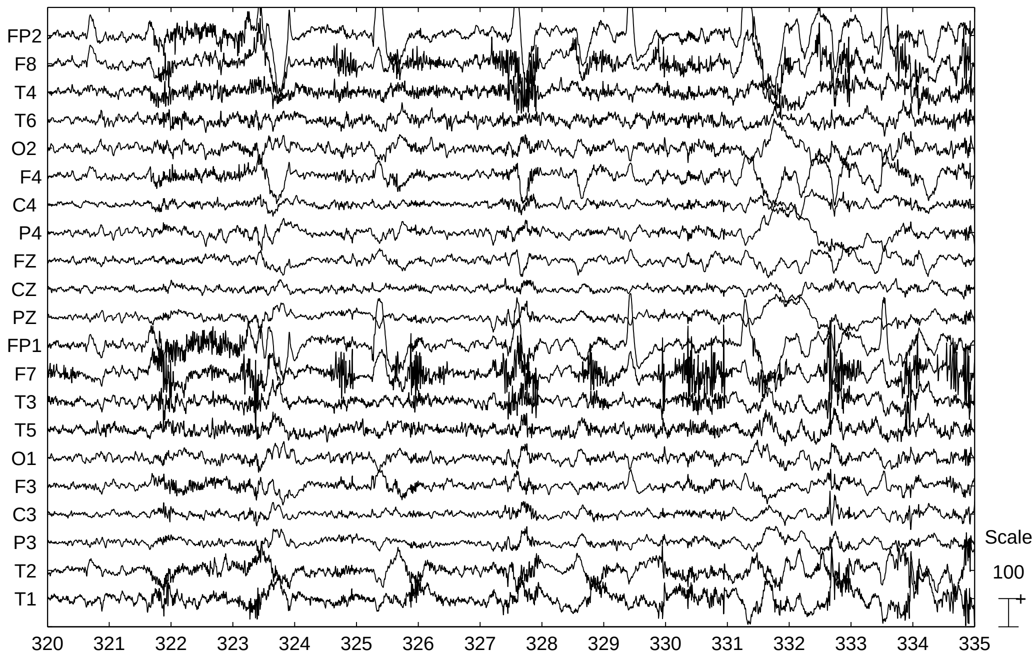

Figure 11: Original EEG frame.

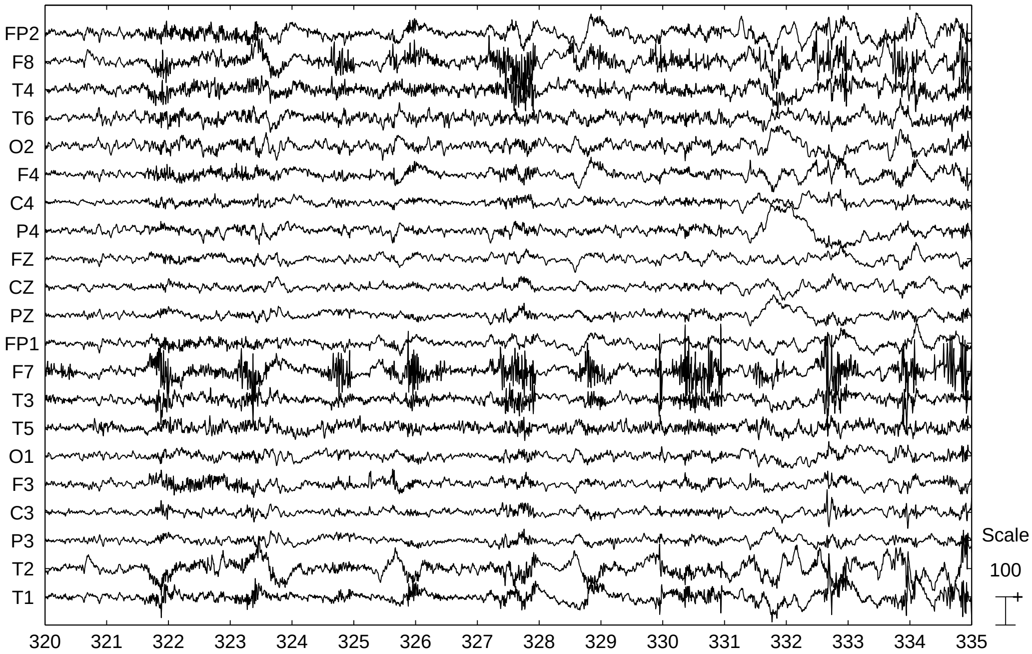

Figure 12: EOG corrected frame.

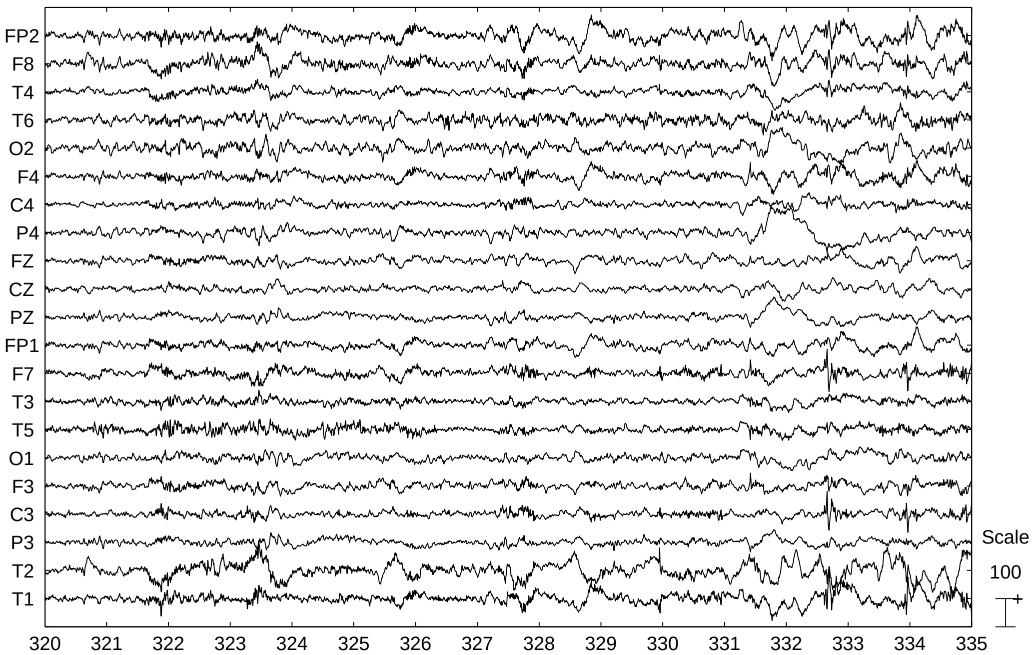

Figure 13: EOG and EMG corrected frame.

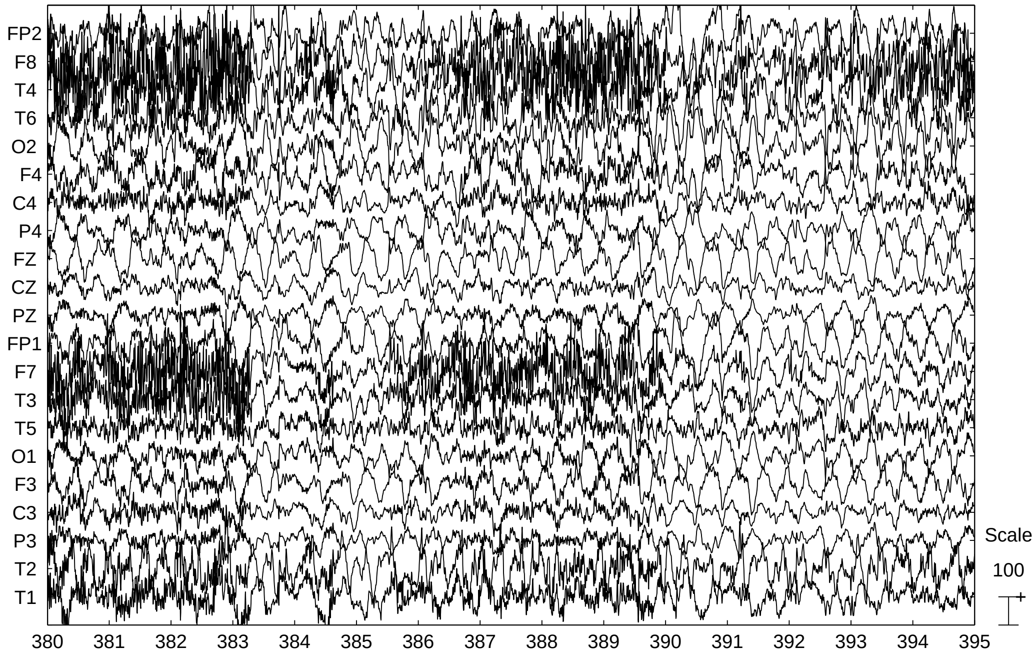

Figure 14: Original EEG frame.

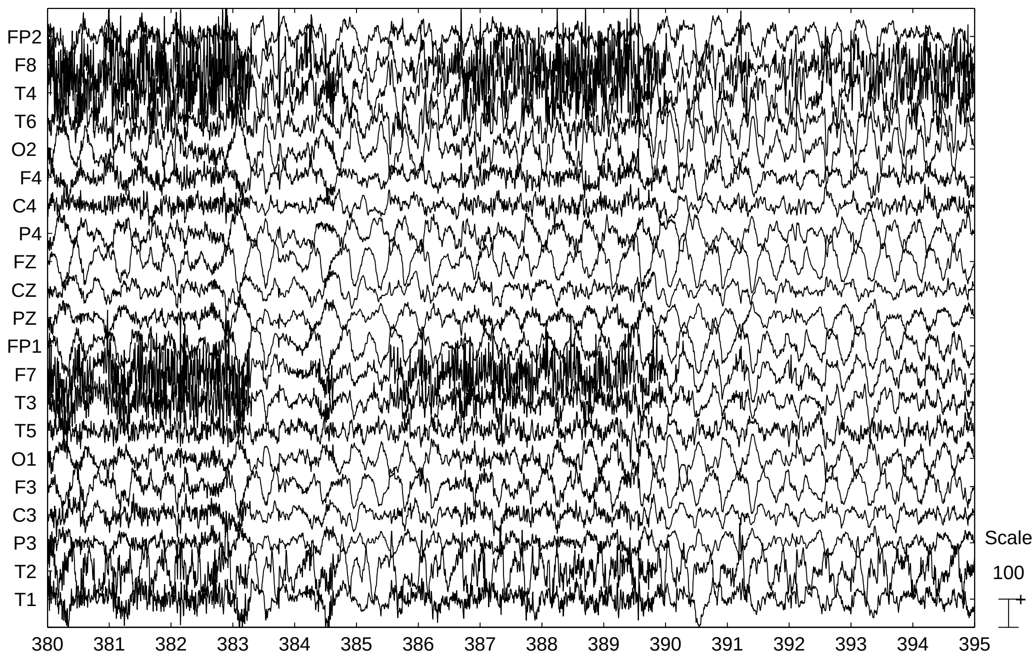

Figure 15: EOG corrected frame.

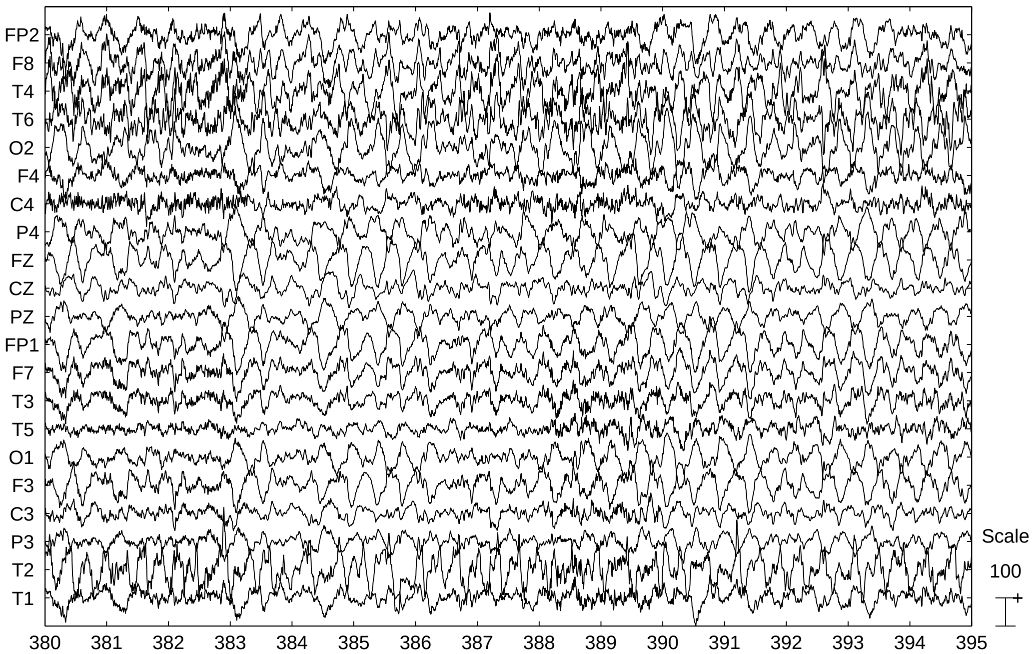

Figure 16: EOG and EMG corrected frame.

  - Documentation has been updated and correction examples have been included.

- **Release 04-12-2007, version 1.3:** minor bugs and few major bugs corrected.

  - A few minor and major bugs were still left in the BSS interface functions and have now been corrected.
  - The interface functions use now `pinv` instead of `inv` to avoid numerically unstable results when the data covariance matrix is close to singular. This numerical unstability might be the reason for the small differences that were observed when running the artifact correction methods under different MATLAB versions and under different operating systems.
  - The default window length in the automatic EMG correction method is now twice as long as it was before.

- **Release 03-12-2007, version 1.3:** major bugs corrected.

  - Several major bugs have been corrected in the functions that act as interface to the BSS algorithms.

  - Parameters passed to the BSS algorithms were ignored in previous releases due to a bug in function autobss.m. This has been corrected.

- **Release 29-11-2007, version 1.3:** minor bugs corrected.
  - Minor bugs corrected.

- **Release 28-11-2007, version 1.3:** major bugs corrected.
  - A major bug in function emg_psd.m has been corrected. This bug was causing the emg_psd.m criterion to incorrectly detect the EMG-related components.
  - Minor update of function pop_autobssemg.m to adapt to the new version of function emg_psd.m
  - The criterion emg_psd.m now accepts a new parameter that allows the user to select the spectral estimator to be used to compute the EEG average power and the EMG average power of each estimated component.
  - A compatibility issue related to function emg_psd.m has been solved. Previously to this change, the criterion emg_psd was producing different results when using MATLAB's Signal Processing Toolbox 6.6 and previous when using older versions of the Signal Processing Toolbox. Now it produces "almost" the same results.
  - Several other minor bugs corrected.

- **Release 07-11-2007, version 1.3:** minor update.
  - pop_autobssemg now automatically passes the sampling rate to emg_psd.m
  - First draft of the toolbox documentation.

- **Release 31-10-2007, version 1.3:** minor update.
  - Minor bugs corrected.
  - EFICA v1.9 was updated with EFICA v2.0 (implemented by Z. Koldovský).

- **Release 29-10-2007, version 1.3:** iWASOBI, EFICA, COMBI, FCOMBI added.
  - Few minor bugs corrected.

- Slightly improved documentation.
- A sample EEG dataset has been included in the release.
- Added several new algorithms for BSS: iWASOBI, EFICA, COMBI, FCOMBI.

- **Release 01-07-2006, version 1.2:**, Regression-based and PCA-based EOG correction added.
  - Automatic EOG correction using Least Mean Squares (LMS) adaptive filtering. This is implemented in `lms_regression.m`.
  - Automatic EOG correction using the conventional Recursive Least Squares (RLS) adaptive algorithm. This is implemented in function `crls_regression.m`.
  - Automatic EOG correction using an stable version of the RLS algorithm. This is implemented in `scrls_regression.m`.
  - Automatic EOG correction using two H-infinity regression methods. These are implemented in functions `hinftv_regression.m` and `hinfew_regression.m`.
  - Automatic EOG correction using Principal Component Analysis (PCA).
  - Some bugs corrected.

- **Release 10-04-2006, version 1.1:** EMG correction included.
  - Automatic correction of EMG artifacts using canonical correlation analysis. This is implemented in functions bsscca.m and `emg_psd.m`.
  - Function BSS has been modified to separate the actual BSS algorithm from the components selection criteria.
  - Interfaces to BSSCCA, FastICA, JADE and RUNICA (implementation of Infomax included in the EEGLAB toolbox) have been included.
  - Ill-conditioning of the data covariance matrix is now handled properly.
  - Some minor bugs corrected.

- **Release 27-02-2006, version 1.0:** original release.
  - Automatic correction of EOG artifacts using Blind Source Separation (BSS).
  - Only an interface to SOBI is provided.

8 Acknowledgments

This work was supported by the EU project No. 043309 (Global Approach to Brain Activity: From Cognition to Disease) and by the Academy of Finland, project No. 213462 (Finnish Centre of Excellence program 2006-2011). Special thanks go to Dr. Ranjit Thuraisingham from the University of Technology at Sydney (Australia) for carefully testing the toolbox, reporting many bugs and helping me in improving the toolbox.

References

[1] A. J. Bell and T. J. Sejnowski. An information maximisation approach to blind separation and blind deconvolution. *Neural Comp.*, 7(6):1129–1159, 1995.

[2] A. Belouchrani, K. Abed Meraim, J.-F. Cardoso, and E. Moulines. A blind source separation technique based on second order statistics. *IEEE T. on Signal Processing*, 45(2):434–444, 1997.

[3] J.-F. Cardoso. High-order contrasts for Independent Component Analysis. *Neural Comput.*, pages 157–192, 1999.

[4] J.-F. Cardoso. Implementation in MATLAB/Octave for the ica of real-valued data. `http://www.tsi.enst.fr/~cardoso/Algo/Jade/jadeR.m`, 2005.

[5] W. De Clercq, A. Vergult, B. Vanrumste, W. Van Paesschen, and S. Van Huffel. Canonical correlation analysis applied to remove muscle artifacts from the electroencephalogram. *Biomedical Engineering, IEEE Transactions on*, 53(12):2583–2587, Nov. 2006.

[6] A. Delorme and S. Makeig. EEGLAB: an open source toolbox for analysis of single-trial EEG dynamics including independent component analysis. *J. Neurosci. Meth.*, 134:9–21, 2004.

[7] G. Gómez-Herrero, W. De Clercq, H. Anwar, Egiazarian K. Kara, O., S. Van Huffel, , and W. Van Paesschen. Automatic removal of ocular artifacts in the eeg without a reference eog channel. In *Proc. NORSIG 2006, Reykjavik, Iceland*, pages 130–133, 2006.

[8] Germán Gómez-Herrero, Z. Koldovský, P. Tichavský, and K. Egiazarian. A fast algorithm for blind separation of non-Gaussian and time-correlated signals. In *Proc. EUSIPCO 2007, Poznan, Poland*, 2007.

[9] S. Haykin. *Adaptive filter theory (3rd ed.)*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1996.

[10] P. He, G. Wilson, and C. Russell. Removal of ocular artifacts from electro-encephalogram by adaptive filtering. *Med. Biol. Eng. Comp.*, 42:407–412, 2004.

[11] A. Hyvärinen. Fast and robust fixed-point algorithms for Independent Component Analysis. *IEEE T. on Neural Networks*, 10(3):626–634, 1999.

[12] Z. Koldovský. Matlab implementation of EFICA. `http://itakura.kes.tul.cz/zbynek/efica.htm`, 2007.

[13] Z. Koldovský, P. Tichavský, and E. Oja. Efficient variant of algorithm fastica for independent component analysis attaining the cramer-rao lower bound. *IEEE T. Neural Networks*, 17(5):1265–1277, 2006.

[14] A. P. Liavas and P. A. Regalia. On the numerical stability and accuracy of the conventional recursive least squares algorithm, 1999.

[15] S. Makeig, A. J. Bell, Jung T. P., and T. J. Sejnowski. Independent component analysis of electroencephalographic data. *Adv. Neural Inf. Process. Syst.*, 8:145–151, 1996.

[16] S. Puthusserypady and T. Ratnarajah. H^∞ adaptive filters for eye blink artifact minimization from electroencephalogram. *IEEE Signal Proc. Let.*, 12(12):816–819, 2005.

[17] S. Puthusserypady and T. Ratnarajah. Robust adaptive techniques for minimization of eog artefacts from eeg signals. *Signal Process.*, 86(9):2351–2363, 2006.

[18] The FastICA team at Helsinki University of Technology. The FastICA MATLAB package.

[19] P. Tichavský. Matlab implementation of iWASOBI. `http://si.utia.cas.cz/downloadPT.htm`, 2007.

[20] P. Tichavský, E. Doron, and A. Yeredor. A computationally affordable implementation of an asymptotically optimal BSS algorithm for AR sources. In *Proc. EUSIPCO'06, Florence, Italy*, 2006.

[21] P. Tichavský and Z. Koldovský. Matlab implementation of MULTI-COMBI. `http://si.utia.cas.cz/downloadPT.htm`, 2007.

[22] P. Tichavský, Z. Koldovský, A. Yeredor, G. Gómez-Herrero, and E. Doron. A hybrid technique for blind separation of non-Gaussian and time-correlated sources using a multicomponent approach. *To appear in IEEE Trans. Neural Networks*, 2008.

[23] P. Tichavský, A. Yeredor, and J. Nielsen. A fast approximate joint diagonalization algorithm using a criterion with a block diagonal weight matrix. In *Submitted to ICASSP 2008*, 2007.

[24] A. Yeredor. Blind separation of Gaussian sources via second-order statistics with asymptotically optimal weighting. *IEEE Signal Processing Letters*, 7(7):197–200, 2000.