Asian Journal of Scientific Research

ISSN 1992-1454

Asian Journal of Scientific Research 7 (4): 460-471, 2014
ISSN 1992-1454 / DOI: 10.3923/ajsr.2014.460.471
© 2014 Asian Network for Scientific Information

High Performance Independent Component Analysis

[1]Jayasanthi Ranjith and [2]N.J.R. Muniraj
[1]Anna University, Tamil Nadu, India
[2]Tejaa Sakthi Institute of Technology for Women, Tamil Nadu, India

Corresponding Author: Jayasanthi Ranjith, Anna University, Tamil Nadu, India

ABSTRACT

Independent Component Analysis (ICA), a statistical signal processing technique, separates the independent source signals from their observed mixtures by maximizing the statistical independence of the components. Since the ICA algorithm is so complex to implement on the FPGA, implementation of this algorithm leads to excessive area and power consumption. This study presents FPGA implementation of a novel area and power efficient Fast Confluence Adaptive Independent Component Analysis (FCAICA) technique with reduced number of recursive iterations. This method occupies less area, less power and provides the high convergence speed. The reduction in area is achieved by hardware optimization and high convergence speed is achieved by a novel optimization scheme that adaptively changes the weight vector based on the kurtosis value. To increase the number precision and dynamic range of the signal, the Floating-point (FP) arithmetic units are used. To validate the performance of the proposed FCAICA, simulation and synthesis are performed with super-gaussian mixtures and experimental results are compared with FastICA and SFLO-ICA (Shuffled Frog Leap Optimization ICA). The proposed FCAICA processor separates the super-Gaussian signals with maximum operating frequency of 2.91 MHz.

Key words: Adaptive independent component analysis, blind source separation, contrast function optimization, field programmable gate array, floating point independent component analysis, very large scale integration

INTRODUCTION

ICA is one of the most commonly used algorithms in blind source separation. The term “blind” means that both the original independent sources and the way the sources were mixed are all unknown. Estimates of the source signals are found only from the observed signal mixtures. ICA recovers source signals from their mixtures by finding a linear transformation that maximizes the mutual independence or non-gaussianity of the mixtures regardless of the probability distribution. It plays an important role in a variety of signal processing, image processing techniques and communication networks. Though different ICA algorithms have been reported, the FastICA algorithm has been shown to have advantages in terms of convergence speed (Oja and Yuan, 2006). It measures the non-Gaussianity using kurtosis to find the independent sources from their mixtures (Hyvarinen *et al.*, 2001). Most ICA algorithms that are based on the Maximum Likelihood (ML) or the Maximization of Negentropy (MN) principle are equivalent when the demixing matrix is constrained to be unitary (Adali *et al.*, 2008). The most popular FastICA algorithm (Hyvarinen, 1999) uses the principle of maximization of negentropy and has the unitary constraint on the separating matrix. The simplest algorithm for maximizing the likelihood uses stochastic gradient methods. Nonlinear decorrelation algorithm has been proposed in order to reduce the computational overhead and to improve stability (Cichocki and Unbehauen, 1996). Another approach to ICA that is related to PCA is the non-linear method. Since learning rule uses higher order information in the learning when nonlinearities are introduced, this method indeed performs ICA, if the data is whitened. Algorithms for exactly maximizing the nonlinear PCA criteria are introduced in (Pham and Garat, 1997). Simple algorithms are derived from the one-unit contrast functions using the principle of stochastic gradient descent. Hebbian like learning rule is obtained by taking instantaneous gradient of contrast function with respect to w (Hyvarinen and Oja, 1998). Joint Approximate Diagonalization of Eigenmatrices (JADE) is based on the principle of computing several cumulant tensors. With low dimensional data, JADE is a competitive alternative to most popular FastICA algorithms. Other approaches include maximization of squared cumulants (Herrmann and Nandi, 2000) and fourth-order cumulant based methods (Nandi and Zarzoso, 1996). Fourth-Order Blind Identification (FOBI) method deals with the Eigen Value Decomposition (EVD) of the weighted correlation matrix (Cardoso, 1989). A Frequency-Domain method of Blind Source Separation (FD-BSS) is able to separate acoustic sources under highly reverberant challenging conditions (Nesta *et al.*, 2011). In Frequency-Domain BSS, the separation is generally performed by applying ICA at each frequency envelope. ICA is also done by entropy bound minimization (ICA-EBM) (Li and Adali, 2010).

Evolutionary computation techniques which are population search based optimization methods like genetic algorithms, Particle swarm optimization are used in ICA (Rojas *et al.*, 2004) (Palaniappan and Gupta, 2006). The only disadvantage of evolutionary computation based ICA technique is that it has heavy computational complexity. But with the advent of highly parallel processors and new technologies like VLSI, these methods provide competitive solutions to the problems. Fixed-point VLSI architecture was proposed for 2-Dimensional Kurtotic FastICA with reduced and optimized arithmetic units (Acharyya *et al.*, 2009). Implementation of ICA algorithm on a fixed point platform and floating point processor shows that the accuracy and speed of fixed point platform were found to be acceptable. In addition, the fixed point processor needs less space and consumes less power. But fixed point processor can handle only smaller range of values (Patil *et al.*, 2011). Due to the computation complexities and convergence rates, ICA is very time-consuming for high volume or high dimensional data set like hyperspectral images. In Parallel ICA (pICA), ICA module is partitioned into three temporally independent functional modules and each of them is synthesized individually. All these modules are developed for reuse and retargeting purpose. It provides optimal parallelism environment, a potential faster and real-time solution (Du and Qi, 2006). FPGA implementation of ICA in digital chip is reported with modular design concept in (Celik *et al.*, 2005) and with systolic architecture in (Jeong *et al.*, 2010). A mixed-signal VLSI system that operates on spatial and temporal differences of the acoustic field at very small aperture to separate and localize mixtures of traveling wave sources is presented in (Du *et al.*, 2007).

ICA techniques are mostly used to solve BSS problems. Speech-recognition technologies are not so popular though they are successful for clean speech signals. This is because of their poor performance in real-world noisy environment. FPGA implementation of 32-channel convolutive ICA chip is reported for real world signals (Kim *et al.*, 2003). Pipelined FastICA which can process the real time sequential mixed signal is reported with its FPGA implementation (Shyu *et al.*, 2008).

To analyse the convergence speed, two different ICA methods named as shuffled frog leap optimization based ICA and fast confluence adaptive ICA are proposed in floating point arithmetic in this study. The most commonly used FastICA algorithm that provides high convergence speed is also developed for comparison purpose. In order to enable the real-time ICA processing in VLSI and to speed up the computation, the ICA algorithms are written by hand coding HDL code. Various analog VLSI implementations of ICA also exist in the literature. Since digital adaptation offers the flexibility of reconfigurable ICA learning rules, digital implementations are common practice in this field. Though there is software that translates the high-level languages such as C code, MATLAB and even Simulink into HDL code, hand coding gives the optimized performance.

The originality of the proposed FCA-ICA is summarized as follows:

- The early determination of converging weight vector and demixing matrix reduces the number of operations required for convergence
- Convergent speed is improved by changing the weight vectors according to fitness value
- Floating point arithmetic improves the precision and dynamic range of the signals

BACKGROUND OF ICA

A long-standing problem in statistics and related areas is to find a suitable representation of multivariate data. Representation here means that data is transformed so that its hidden, essential structure is made more visible or accessible. Blind source separation is a problem of finding a linear representation of hidden data from the mixture in which the components are statistically independent. In practical situations, we cannot in general find a representation where the components are really independent but we can at least find components that are as independent as possible. Independent component analysis is a major task in signal processing to extract the source signals from the observed mixtures. The relationship between source signals S and observed mixtures X is given in matrix notation as in Eq. 1:

$$X = A\ S \tag{1}$$

A is a full rank matrix which is called mixing matrix. Under some assumptions, ICA solves the BSS problem by finding inverse linear transformation such that, it maximizes the statistical independence between the observed mixtures. In doing this, ICA finds unmixing matrix B. Then the estimate of the source signal (S_est) is found from Eq. 2:

$$S\_est = B\ X = S \tag{2}$$

ICA preprocessing: In order to simplify the ICA process, it is necessary to perform preprocessing before applying the mixtures to the ICA algorithm. The preprocessing of mixed signal involves finding the mixing matrix P. The first step in preprocessing is called centering.

Let N statistically independent sources be mixed through NxN nonsingular mixing matrix A so that we obtain the observed signal mixtures given by $x_1(t)$, $x_2(t)$, ...$x_n(t)$ which are the amplitudes of the recorded signals at time point t. For N = 2, representation of original source signals is given by $s_1(t)$, $s_2(t)$ and mixtures are $x_1(t)$ and $x_2(t)$. Centering consists of subtracting the mean from each observed mixtures $X_1(t)$ and $X_2(t)$ to produce zero mean outputs C_X_1 and C_X_2 as shown in Fig. 1.

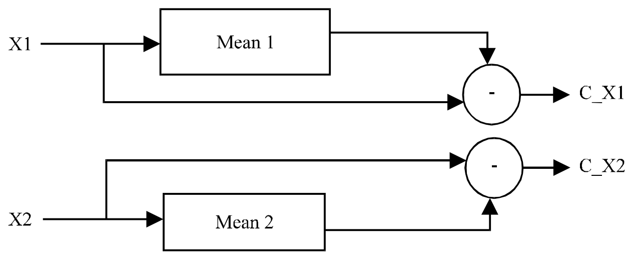

Fig. 1: Implementation of centering

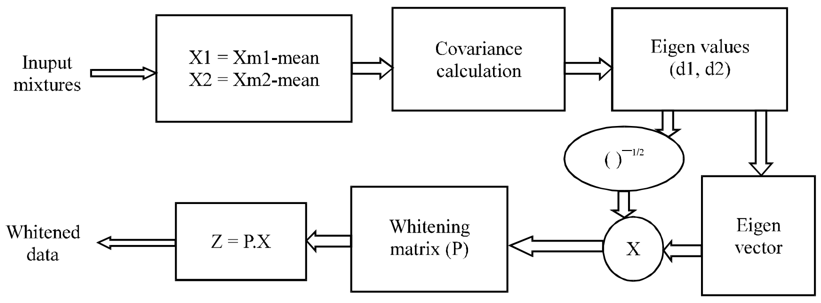

Fig. 2: Implementation of whitening

The second step is called whitening and consists in linear transformation of the centered mixtures, to obtain new vectors which are white. Figure 2 shows the whitening process. The components of a whitened vector are uncorrelated and their variances equals to unity. This means that the covariance matrix of whitened data is equal to the identity matrix. One way to perform whitening is to use Eigen Value Decomposition (EVD). The whitening matrix can be found by using Eq. 3:

$$P = ED^{-1/2}ET \tag{3}$$

where, E is the orthogonal matrix of eigenvector found from the covariance matrix E $\{XX^T\}$. D is the diagonal matrix of the eigenvalues associated with each eigenvector. The efficiency of ICA is based on the selection of cost functions, also called objective functions or contrast functions. The cost function in some way or other is a measure of independence (Adali *et al.*, 2008). Some measures of independence are kurtosis, negentropy and mutual information. Though there are different contrast functions, the most popular contrast function used in ICA is kurtosis.

FLOATING POINT ARITHMETIC

Based on the storage area available, there are two variants of floating point representation of a real number i.e., IEEE single-precision representation and IEEE double-precision representation. IEEE single precision format, that uses 32 bits, has been used for this proposed ICA algorithm. Format of 32 bit floating point representation is shown in Fig. 3.

The sign field S in Fig. 1 is used to specify the sign of the real number. Exponent field E is a 8 bit quantity represented by using a bias of 127. Bits 22 down to 0 in field M are used to store the binary representation of the floating point number. Since leading one in the mantissa is implicit,

S	E	M
31	30......23	22......0

Fig. 3: Thee to thirty two bit floating point format representation

it does not appear in the representation. Addition, subtraction, multiplication, division and square root operations are carried out following the appropriate algorithms of single precision IEEE 754 standard.

FAST-ICA

FastICA algorithm: Due to simplicity and fast convergence, FastICA is considered as one of the most popular solutions for linear ICA/BSS problem. The VLSI implementation of this algorithm involves the preprocessing and iteration scheme.

- **Iteration for one unit:** The FastICA algorithm for one unit estimates one row of the demixing matrix as a vector that is an extremum of contrast functions. FastICA is an iterative algorithm, derived from kurtosis based contrast function. Assuming Z as the whitened data vector and $w^T(k+1)$ as one of the rows of the separating matrix, estimation of $w^T(k+1)$ is done iteratively until convergence is achieved. The FastICA algorithm involves the following steps:

Step 1: Choose an initial random vector of unit norm (w_{old})
Step 2: Find norm of vectors and divide by corresponding norms
Step 3: Update the vector using whitened data vector Z to find w_{new}:

$$w_{new} \leftarrow E\{Z(w(k)Z^T)^3\}\text{-}3w(k)$$

Step 4: If $W_{new}\text{-}w_{old}<\varepsilon$ is not satisfied then go back to step 2. Where, ε is a convergence parameter ($\sim10^{-4}$) and w_{old} is the value of before it‘s replacement by the newly calculated value w_{new}

- **Fixed-point iteration for finding several ICs:** More than one independent components are estimated using deflationary approach one by one or estimated simultaneously by symmetric approach. In order to prevent that the algorithm estimates the same component more than one time, the orthogonalization is made using Eq. 4 and 5. This verification is done by subtracting the projections of all previously estimated vectors from the current estimate after every iteration step and before normalization

$$Wp \leftarrow (Wp\text{-}(Wp^T Wj)Wj \tag{4}$$

In the symmetric approach the iteration step is computed for all w_p and the matrix W is orthogonalized as:

$$w \leftarrow (ww^T) - \frac{1}{2}w \tag{5}$$

The results are obtained in FastICA following the deflationary approach.

SFLO-ICA

In this ICA method, contrast function optimization is performed based on SFLO for improving the optimality and convergence performance. Mutation operator introduced in SFLO Algorithm avoids the solution from getting trapped in local minima. It converges better in lesser time when compared to other optimization algorithms. In this algorithm, initial weight vectors for estimating the demixing matrix are assumed as frogs and updated by step 3 of the algorithm. Then fitness value is calculated and sorting is done according to fitness value. Based on the fitness values, total population is partitioned into q groups (memeplexes) of p frogs,that search independently. In this process, the first frog goes to the first memeplex, the second frog goes to the second memeplex, pth goes to the pth memeplex and frog p+1 goes back to the first memeplex and so on. In each memeplex, the frogs with the best and the worst fitnesses are identified as Xb and Xw, respectively. Also, the frog with the most qualified fitness level among all the memeplexes is identified as Xg. Then improvement is done to improve only the frog with the worst fitness according to step 8. If this process produces a better solution, it replaces the worst frog. Otherwise, a new population is randomly generated to replace that population. This process continues for a specific number of iterations (I_{max1}). Then all memeplexes are combined and sorted. Then mutation operation is included using (11) to avoid local minima. If the current iteration number reaches (I_{max2}), the search procedure is stopped; otherwise it goes to step 5. The last Xg is the solution of the problem.

Floating point Iteration: Estimation of $w^T(k+1)$ is done iteratively with following steps until a convergence is achieved:

(1) Choose initial population of 'n' frogs (weights) at random
(2) Find norm of pair of frogs and divide by corresponding norms
(3) Update the frogs by the equation $w(k+1) \leftarrow E\{Z(w(k)Z^T)^3\}-3w(k)$
(4) Calculate the fitness value from $f = w(k+1)-w(k)$
(5) Sort the initial population based on the fitness values with decreasing manner
(6) Partition the sorted population into p memeplexes of q frogs
(7) Select the best frog (Xb), worst frog (Xw) in each memeplex and globally best frog (Xg)
(8) Update the position of Xw using $Xw\,(new) = Xw_{(old)}+C$. where, $C = rand().(Xb-Xw)$
(9) If it produces better solution, older frog is replaced by updated frog and this process continues for a specific number of iterations (Imax1). Otherwise a new frog is randomly generated to replace Xw and algorithm goes to step 2
(10) All memeplexes are combined and sorted again
(11) Apply mutation

$$X^i_{mut} = X^i_{rand}+rand(.)(X^i_b-X^i_{rand})+rand(.)(X^i_g-X^i_{rand})$$

(12) If the current iteration number reaches Imax2, the search procedure is stopped or it goes to step 5
(13) The last X g is the solution of the problem

Here, X^i_{rand} is a randomly generated vector, Nmem is the number of memeplexs, $i = 1,2,\ldots\ldots Nmem$, rand () is random number between (0 and 1) and ε is a convergence parameter ($\sim 10^{-4}$). With the above steps, deflationary othogonalization is made to find second independent component.

NOVEL LOW POWER FAST CONFLUENCE ADAPTIVE ICA

Though the above algorithm which is based on MSFO improves the optimality performance, it suffers from computational complexity due to large number of iterative calculations in floating point iteration scheme. For reducing the number of manipulations and for improving the performance of ICA algorithm in terms of convergence speed and power, adaptive optimization of contrast function is proposed in floating point arithmetic. Here, initial weight vectors for estimating the demixing matrix B in (2), are assumed as frogs and are described as memetic vectors. This algorithm computes new weights (frogs) from the initial weights in adaptive manner based on fitness value.

Floating point iteration for one unit: Having done the preprocessing to whiten the mixed signal,this algorithms is used to find the independent components. The proposed fast confluence adaptive ICA algorithm for one unit estimates one row of the demixing matrix. Updation of weights continues in iterative manner with following steps until a convergence is achieved.

(1) Choose initial frogs (wi) of 'N' numbers at random
(2) Find norm of frogs and divide by corresponding norms
(3) Update all the N frogs by using the equation

$$w(k+1) \leftarrow E\{Z(w(k)Z^T)^3\}-3w(k)$$

(4) Calculate the fitness value f = {w(k+1)-w(k)} and sort the frogs according to the fitness values
(5) Divide the N frogs into M groups (N = 2*M) with 2 frogs in each group. The division is done in such a way that 1st frog goes to 1st group, 2nd frog goes to 2nd group and continuous up to M frogs. Then (M+1)th frog goes to 1st group and so on
(6) In each group, determine the best and worst individuals. Update the worst frogs using step 3
(7) If $\{w(k+1)-w(k)<\varepsilon$ is not satisfied, then go back to step 2 by adaptively taking a new 'wi' lesser than that of worst frog where ε is a convergence parameter ($\sim 10^{-4}$)
(8) When $\{w(k+1)-w(k)<\varepsilon$ is satisfied, move to next group of frogs and repeat from step 6 until iteration limit is reached
(9) Then the two vectors with good fitness value can be used as row vectors of demixing matrix. With the above steps, deflationary othogonalization is then made to find second independent component

RESULTS AND DISCUSSION

For the verification of the validity and performance of the FastICA and the two proposed ICA algorithms, two different super-gaussian signal mixtures are taken and applied to the algorithms. The input signal to these ICA algorithms are speech and mike signals mixed with artificial mixing matrix A. The mixing matrix is a full rank matrix of 2 rows and 2 columns. The experiment was carried out first, for small-sized problem with 256 samples. Because the defined algorithm must be capable of efficiently solving the real-world sized instances, another experiment is carried out for large-sized problem with 3000 samples each.

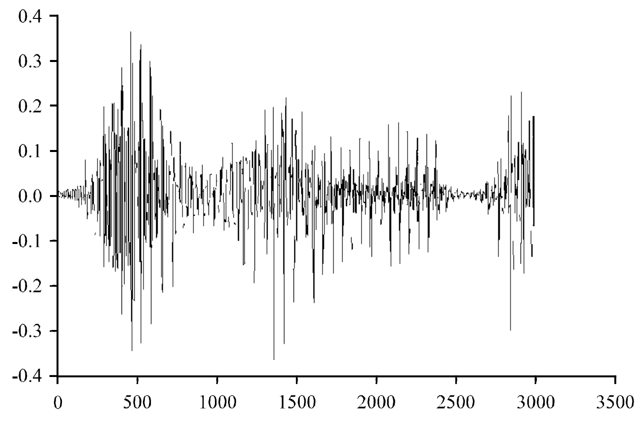

Fig. 4: Speech signal 1

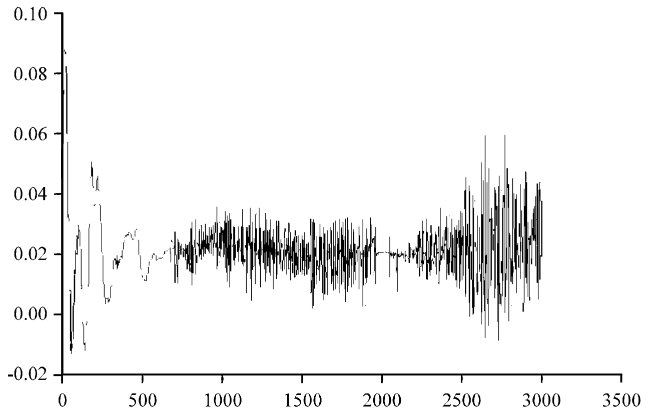

Fig. 5: Speech signal 2

Parameter setting: For fine-tuning this category of problem to real world applications, extensive experiments were carried out with parameters. The parameters set are:

- Number of weight vectors is 2 since the dimension of the demixing matrix is 2
- Maximum number of iteration for each weight vector is 100

Results of supergaussian mixture: The original supergaussian speech signals applied to the ICA process are shown in Fig. 4 and 5, respectively. Supergaussian signals have kurtosis value greater than zero. It is less than zero for sub gaussian signals. When kurtosis is zero,the signal is gaussian for which ICA cannot be applied. Most of the real world signals are supergaussian in nature. Examples are speech signal, train sound, car noise etc. sine waves, sawtooth waves are examples of subgaussian signals. The super-gaussian mixture signals are shown in Fig. 6 and 7, respectively. Figure 8 and 9 shows the independent components obtained through proposed systems.

Simulation and Implementation results of ICA algorithms: All the algorithms are written in VHDL and implemented on FPGA ALTERA using Quartus 10.0 Tool. The simulation results are

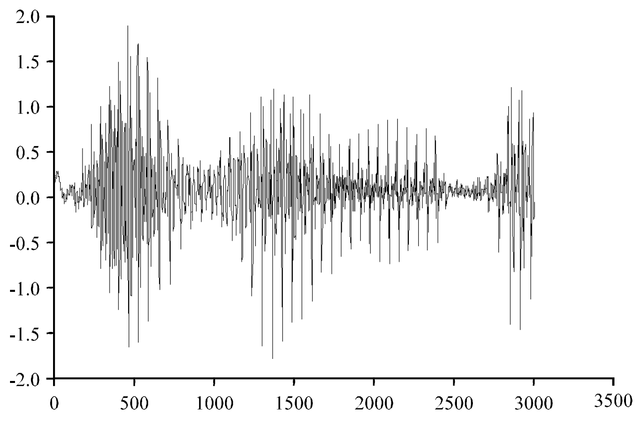

Fig. 6: Mixture of speech signal 1

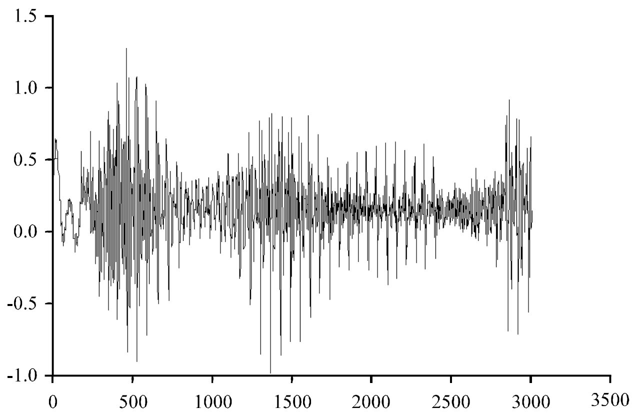

Fig. 7: Mixture of speech signal 2

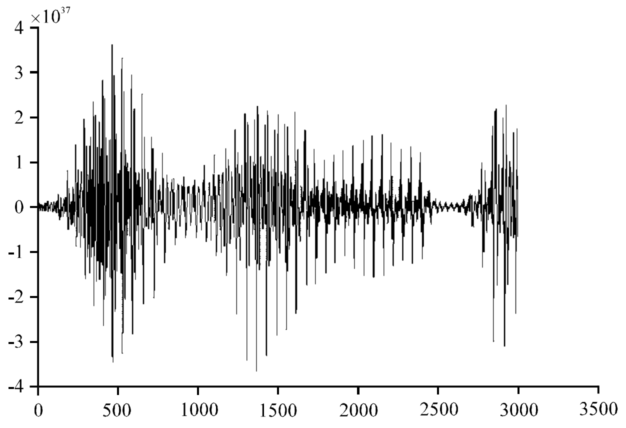

Fig. 8: Recovered speech signal

obtained from Modelsim 10.0c Tool. Table 1 compares the performance of FastICA, SFLO-ICA and FCA-ICA in terms of area, power and maximum operating frequency and convergence speed T. Here, T represents the time taken for each of the algorithms to reach convergence.

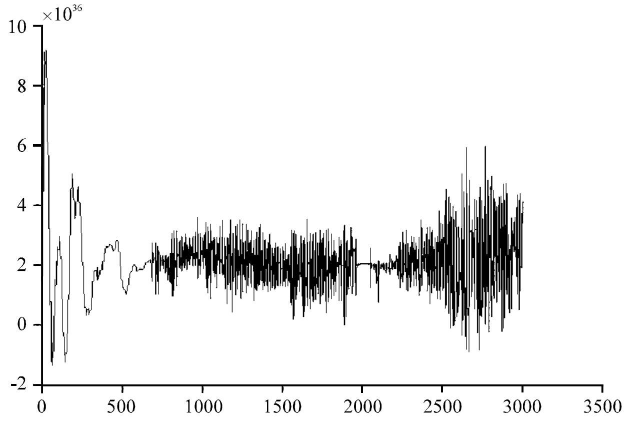

Fig. 9: Recovered mike signal

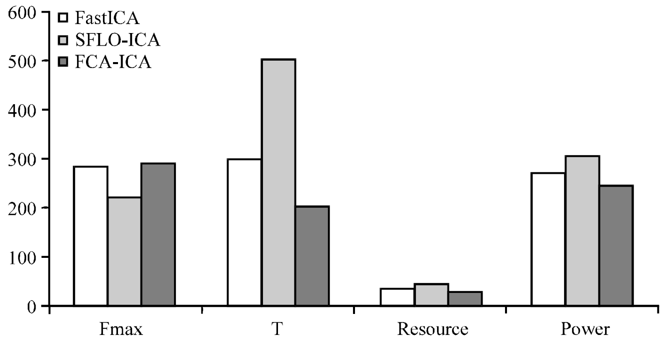

Fig. 10: Performance comparison of ICA algorithms

Table 1: Comparison of ICA algorithms

Parameters	FastICA	SFLO-ICA	FCA-ICA
Resource utilization (%)	34.0	43.0	27.0
Power dissipation (mW)	270.76	307.27	246.94
F_{max}	2.85	2.21	2.91

Implementation of FastICA algorithm, SFLO-ICA and FCA-ICA algorithm shows that the achievable speed is 2.85, 2.21 and 2.91 MHz for FastICA, SFLO-ICA and FCA-ICA, respectively. This is the maximum value of frequency at which the circuit can be clocked. Figure 10 shows the plot of area, Fmax, power and convergence time. The FCA-ICA based extraction of components from their mixtures consumes lesser area, lesser power and provides faster convergence compared to the other two. Further investigations of design implementation in FPGA for power, area and timing performance shows that, area and power increases whereas time reduces in the order of FCA-ICA, FastICA and SFLO-ICA.

Convergence analysis: The convergence analysis is done from the simulation results obtained from Modelsim 10.0c Tool. Convergence speed T represents the time taken for each of the algorithms to reach convergence. It is 300, 500 and 200 ps for Fast ICA, SFLO-ICA and FCA-ICA, respectively. It shows that FCAICA converge faster than FastICA and SFLO-ICA due to increased search space and adaptivity. Figure 11 shows the convergence plot.

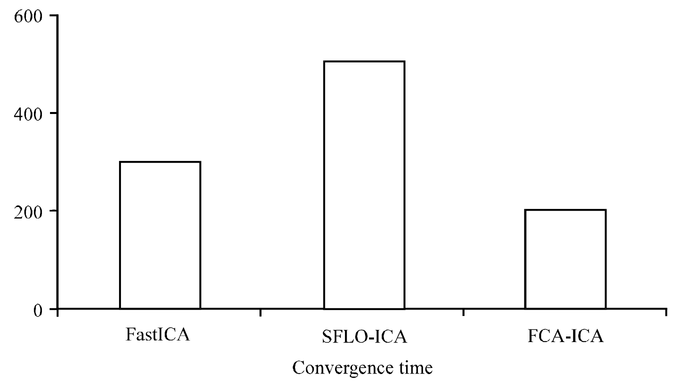

Fig. 11: Convergence time of ICA algorithms

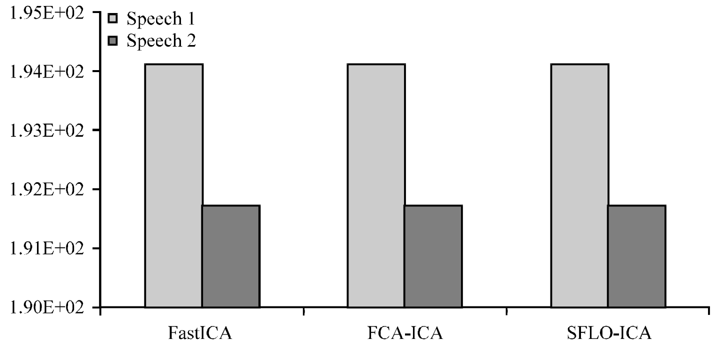

Fig. 12: SNR for supergaussian mixtures

SNR analysis: Figure 12 shows that all the ICA algorithms have same Signal to Noise (SNR) for a particular signal. Though SNR is same, FCAICA performance is better than FastICA and in terms of convergence speed, operating frequency, area and power.

CONCLUSION

In this study, new time-domain approaches to estimate the independent components from their observed supergaussian mixtures have been presented. Use of modularity and hierarchy simplifies the design, reduces the area, power and speeds up the convergence process of ICA. The usage of optimization algorithm enables finding optimal solution. Floating point manipulations enable increased input signal range. The peculiarity of the resulting system is the capability of providing faster convergence with improved precision. Further research includes the application of the proposed method for other signals, such as EEG, spread spectrum signals and images under poor SNR circumstances. Further improvement is possible by employing this technique with sources more than two.

REFERENCES

Acharyya, A., K. Maharatna, J. Sun, B.M. Al-Hashimi and S.R. Gunn, 2009. Hardware efficient fixed-point VLSI architecture for 2D Kurtotic FastICA. Proceedings of the 19th IEEE European Conference on Circuit Theory and Design, August 23-27, 2009, Antalya, Turkey, pp: 165-168.

Adali, T., H. Li, M. Novey and J. Cardoso, 2008. Complex ICA using nonlinear functions. IEEE Trans. Signal Process., 56: 4536-4544.

Cardoso, J.F., 1989. Source separation using higher order moments. Proceedings of the International Conference on Acoustics, Speech and Signal Processing, May 23-26, 1989, Glasgow, UK., pp: 2109-2112.

Celik, A., M. Stanacevic and G. Cauwenberghs, 2005. Gradient flow independent component analysis in micropower VLSI. Adv. Neural Inform. Process. Syst., 18: 187-194.

Cichocki, A. and R. Unbehauen, 1996. Robust neural networks with on-line learning for blind identification and blind separation of sources. IEEE Trans. Circuits Syst. I: Fundam. Theory Appl., 43: 894-906.

Du, H. and H. Qi, 2006. A reconfigurable FPGA system for parallel independent component analysis. EURASIP J. Embedded Syst., 10.1155/ES/2006/23025

Du, H., H. Qi and X. Wang, 2007. Comparative study of VLSI solutions to independent component analysis. IEEE Trans. Ind. Electron., 54: 548-558.

Herrmann, F. and A.K. Nandi, 2000. Maximisation of squared cumulants for blind source separation. Electron. Lett., 36: 1664-1665.

Hyvarinen, A. and E. Oja, 1998. Independent component analysis by general nonlinear Hebbian-like learning rules. Signal Process., 64: 301-303.

Hyvarinen, A., 1999. Fast and robust fixed-point algorithms for independent component analysis. IEEE Trans. Neural Networks, 10: 626-634.

Hyvarinen, A., J. Karhunen and E. Oja, 2001. Independent Component Analysis. Wiley, New York.

Jeong, H., Y. Kim and H.J. Jang, 2010. Adaptive parallel computation for blind source separation with systolic architecture. Intell. Inform. Manage., 2: 46-52.

Kim, C.M., H.M. Park, T. Kim, Y.K. Choi and S.Y. Lee, 2003. FPGA implementation of ICA algorithm for blind signal separation and adaptive noise canceling. IEEE Trans. Neural Networks, 14: 1038-1046.

Li, X.L. and T. Adali, 2010. Independent component analysis by entropy bound minimization. IEEE Trans. Signal Process., 58: 5151-5164.

Nandi, A.K. and V. Zarzoso, 1996. Fourth-order cumulant based blind source separation. IEEE Signal Process. Lett., 3: 312-314.

Nesta, F., P. Svaizer and M. Omologo, 2011. Convolutive BSS of short mixtures by ICA recursively regularized across frequencies. IEEE Trans. Audio Speech Lang. Process., 19: 624-639.

Oja, E. and Z. Yuan, 2006. The FastICA algorithm revisited: Convergence analysis. IEEE Trans. Neural Networks, 17: 1370-1381.

Palaniappan, R. and C. Gupta, 2006. Genetic algorithm based independent component analysis to separate noise from electrocardiogram signals. Proceedings of the IEEE International Conference on Engineering of Intelligent Systems, April 22-23, 2006, Islamabad, Pakistan, pp: 1-5.

Patil, D., N. Das and A. Routray, 2011. Implementation of Fast-ICA: A performance based comparison between floating point and fixed point DSP platform. Meas. Sci. Rev., 11: 118-124.

Pham, D.T. and P. Garat, 1997. Blind separation of mixture of independent sources through a quasi-maximum likelihood approach. IEEE Trans. Signal Process., 45: 1712-1725.

Rojas, F., C.G. Puntonet, M. Rodriguez-Alvarez, I. Rojas and R. Martin-Clemente, 2004. Blind Source separation in post-nonlinear mixtures using competitive learning, simulated annealing and a genetic algorithm. IEEE Trans. Syst. Man Cybernat. Part C: Appl. Rev., 34: 407-416.

Shyu, K.K., M.H. Lee, Y.T. Wu and P.L. Lee, 2008. Implementation of pipelined FastICA on FPGA for real-time blind source separation. IEEE Trans. Neural Networks, 19: 958-970.