Forecasting the MagnetoEncephaloGram (MEG) of Epileptic Patients Using Genetically Optimized Neural Networks

Adam V. Adamopoulos
Laboratory of Medical Physics, Medical School, Democritus University of Thrace, Creece.

Efstratios F. Georgopoulos
University of Patras, Dept. of Computer Engineering & Informatics, Patras 26500, Greece.

Spiridon D. Likothanassis
University of Patras, Dept. of Computer Engineering & Informatics, Patras 26500, Greece.

Photios A. Anninos
Laboratory of Medical Physics, Medical School, Democritus University of Thrace, Creece

Abstract

In this work MagnetoEncephaloGram (MEG) recordings of epileptic patients were analyzed using a hybrid neural networks training algorithm. This algorithm combines genetic algorithms and a training method based on the localized Extended Kalman Filter (EKF), in order to evolve the structure and train Multi-Layered Perceptrons (MLP) networks. Our goal is to examine the predictability of the MEG signal on a short and long predicting horizon. Numerous experiments were conducted giving highly successful results.

1 INTRODUCTION

A very interesting task in the field of signal analysis is the short and long term prediction of real world signals. In the present work is examined the predictability of MEG recordings of epileptic patients. MEG recordings were obtained using a Super-conductive QUantum Interference Device (SQUID) and were digitized with a sampling frequency of 256Hz using a 12-bit A/D Converter. SQUID is a very sensitive magnetometer, capable to detect and record the bio-magnetic fields produced in the human brain due to the generation of electrical micro-currents at neural cellular level (Anninos et. al. 1997). MEG data were provided by the Laboratory of Medical Physics of the Democritus University of Thrace, Greece, where a one-channel DC SQUID is operable. The MEG data were normalized in the interval [0,1] in order to be processed by the neural networks.

The problem of MEG predictability is faced using Multi-Layered neural networks which were evolved using a novel hybrid algorithm which combines Genetic Algorithms (GA's) and a training algorithm, based on the localized Extended Kalman Filter (EKF), known as Multiple Extended Kalman Algorithm (MEKA). The MEKA is described in detail in [Shah et. al., 1992]. The task of the proposed modified genetic algorithm is to evolve a population of MLP neural networks and find a near optimum network architecture (considering the number of inputs, the number of hidden units, etc.) that solves a specific problem, while, at the same time the Kalman training algorithm is used to train these networks. The novelty of this effort depends on, apart from the combination of evolution programs with the Kalman training algorithm, the capability of the proposed method to search, not only for the optimal number of hidden units, but also, for the number of inputs needed for the problem at hand. As far as we know this is the first attempt in the relevant literature to evolve simultaneously the numbers of hidden and input neurons of a network. The above method, but in a simpler version, was used successfully for system structure identification, using single layer neural networks in [Adamopoulos et. al., 1998] and for exchange rates forecasting in [Likothanassis et. al, 1998].

The rest of the paper is organized as follows. Section 2 describes the hybrid algorithm, while the numerical experiments are presented in section 3. Finally, section 4 discusses the concluding remarks.

2 THE HYBRID ALGORITHM

The proposed modified GA maintains a population of individuals (Neural Networks) for each generation, having random structure in the hidden region. The MEKA algorithm is employed for the training of each network for just one epoch. Performance is measured with the fitness function, which is a function of the MSE and the size (number of nodes) of the network. Then a new population is created, by selecting the more fit individuals according to their fitness (select step). Some members of the population undergo transformations by means of genetic operators to form the new individuals. We use a mutation operator that changes, randomly, the structure of the network in order to preserve diversity. Also, there is a crossover operator, which creates new individuals by combining parts from two individuals. After some number of iterations the program converges at a near-optimum solution. The steps of the algorithm are briefly described in the following:

Step1, Initialization: An initial population of randomly generated individuals (random number of inputs and hidden neurons) is created. Generally a large population size is preferable, but in our experiments we need to compromise with the computer limitations, so a population of fifty individuals was used in all of the conducted experiments. The connection weights are initialized to random values in [-1,1], using uniform probability distribution.

Step2, Selection: Selection is an essential operation in genetic algorithms; it constructs a new population with respect to the probability distribution based on fitness values of the individuals of the previous population. In our experiments a variation of the classic Roulette Wheel Selection Operator [Michalewicz, 1996] was used. In this variation we save the best ever individual in a place outside the population and in the selection operation we make sure that at least one copy of this individual will pass to the next generation (elitism).

The fitness function that is used in the selection phase takes in mind the performance of the network on the test set and its size and has the following form:

$$\text{Fitness} = 1/(1 + \text{MRE} + \text{size\_par}*\text{MRE}*\text{SIZE}) \quad (1)$$

Where size_par is a parameter that controls the importance of size in the evaluation of fitness function. The objective is for size_par to take values that will lead to individuals with small sizes, maintaining though good forecasting ability. The term (size_par*MRE*SIZE) allows for the importance of the network's size to decrease accordingly to the decrease of MRE.

Step 3, Crossover: The crossoveroperator is applied to the new population. Generally it works as follows: it selects two parents and generates one or two offspring by recombining parts of them. The offspring take the place of their parents in the new population. In the proposed algorithm crossover operates precisely as follows:

Let assume that we have the two parents: I_1H_1O and I_2H_2O where I, H and O are the numbers of input, hidden and output nodes, respectively. Next we generate the random numbers:

i_1= a uniform random number in $[0,I_1]$,
i_2= a uniform random number in $[0,I_2]$,
h_1= a uniform random number in $[0,H_1]$,
h_2= a uniform random number in $[0,H_2]$.

Then we create a child with (i_1+i_2) input notes, (h_1+h_2) hidden nodes and O output nodes. If $(i_1+i_2) = 0$ then we set the number of input nodes to 1; if $(h_1+h_2) = 0$ we set the number of hidden nodes to 1. The weights of the child are initialized randomly in the same interval that was used in the initialization phase. The second child is created in the same manner.

Step4, Mutation: The mutation operator that was used, works as follows: it selects in random a neural network (individual) from the population and changes its number of inputs and/or its number of hidden neurons by adding or deleting a random number (selected uniformly from a given interval) of inputs and/or hidden neurons.

3. NUMERICAL EXPERIMENTS

In this section we present the results of the experiments that were conducted in order to evaluate the algorithm's performance. As stated in the introduction our initial intention was to test the prediction ability of the proposed algorithm for the case of MEG recordings of epileptic patients. Our work was split in two different directions. First, we examine the ability of the algorithm to produce networks that can predict accurately MEG recordings using different predicting horizons. Second, we tried to check how the choice of the size_param value influences the size of the produced networks for the problem of short term (next value – predicting horizon=1) prediction of MEG recordings.

In all the experiments we used the same parameter values (for comparison reasons) except of course of the size_param value for the second set of experiments. So, we used a population size of 50 neural networks, probability of crossover equal to 0.15, probability of input mutation equal to 0.2 and probability of hidden node mutation equal to 0.2. For the training of the networks we used 1024 data samples (corresponding to a four seconds epoch of the MEG) while for the testing we used 512 data samples (corresponding to a two seconds epoch of the MEG). The algorithm was left to run for 3000 generations.

In order to evaluate the forecasting capability of the produced networks we used three well-known error measures, the Normalized Root Mean

Squared Error (NRMSE), the Correlation Coefficient (CC) and the Mean Relative Error (MRE).

The NRMSE is calculated using the root-mean-squared-error (rmse) given by

$$\sigma_{\Delta}(T) = \left\langle \left[x_{pred}(t,T) - x_{act}(t,T) \right]^2 \right\rangle^{1/2} \qquad (2)$$

where T is the number of samples being predicted. The RMSE is normalized by the RMSE deviation of the data

producing the normalized error $NRMSE=\sigma_{\Delta}(T)/\sigma_x$. If NRMSE=0 then predictions are perfect; NRMSE=1 indicates that prediction is no better than taking x_{pred} equal to the x-mean.

$$\sigma_{\Delta} = \left\langle \left(x - \langle x \rangle \right)^2 \right\rangle^{1/2}$$

Prediction was also tested using the correlation coefficient (CC) between the actual and predicted series. The CC measures the ability of the predicted samples to follow the upward or downward jumps of the original series.

The MRE is given by the formulae:

$$MRE = \frac{1}{n} \sum_{i=1}^{n} \left| \frac{o_i - d_i}{d_i} \right| \qquad (3)$$

where o_i is the output of the network and d_i is the desired value when pattern *i* is presented, and *n* is the total number of patterns. MRE shows the percentage of the accuracy of predictions expressing it in a stricter way, since it focuses on the sample being predicted. Thus, we are able to estimate prediction error as a fraction of the actual value.

In the following Tables we can see the architecture (That has the form: inputs – hidden nodes - outputs) and the errors on the training set (Table 1) and on the test set (Table 2), of the best network, generated by the evolutionary algorithm for the case of MEG prediction using varying predicting horizon.

As stated above one of the directions of our research was to examine how the choice of size_param value influences the architecture (size) of the generated network. So we ran the algorithm for several size_param values, keeping the other parameters constant, for the problem of MEG prediction using predicting horizon equal to 1. The results are exhibited in the following tables for the case of training set (Table 3) and test set (Table 4).

The following Figures depict the performance of the generated networks, at the end of the algorithm's run, on the test set and how the MRE and the architecture of the best network are changing through the generations.

Table 1. MEG forecasting with varying Predicting Horizon - Errors on the Training Set

Predicting Horizon	**Architecture (I-H-O)**	**NRMSE**	**C.C.**	**MSE**	**RMAE**
1	4-8-1	0.2608	0.9654	0.0004736	0.0359
2	4-3-1	0.4130	0.9141	0.0012	0.0561
3	4-15-1	0.4486	0.8961	0.0014	0.0614
4	5-2-1	0.5681	0.8327	0.0022	0.0747
5	4-6-1	0.6291	0.7911	0.0027	0.0846

Table 2. MEG forecasting with varying Predicting Horizon - Errors on the Test Set

Predicting Horizon	**Architecture (I-H-O)**	**NRMSE**	**C.C.**	**MSE**	**RMAE**
1	4-8-1	0.2001	0.9798	0.000401	0.0403
2	4-3-1	0.3064	0.9525	0.000937	0.0627
3	4-15-1	0.3654	0.9310	0.0013	0.0740
4	5-2-1	0.4299	0.9054	0.0018	0.0855
5	4-6-1	0.4990	0.8719	0.0025	0.1025

Table 3. MEG forecasting with Horizon =1 - Errors on the Training Set

Size Parameter	**Architecture (I-H-O)**	**NRMSE**	**C.C.**	**MSE $\times 10^{-4}$**	**RMAE**
0.004	4-9-1	0.2540	0.9674	4.4915	0.0350
0.008	3-3-1	0.2913	0.9581	5.9108	0.0386
0.012	4-5-1	0.2564	0.9672	4.5789	0.0357
0.016	3-2-1	0.2967	0.9557	6.1316	0.0411
0.026	3-3-1	0.2705	0.9656	5.0886	0.0369
0.035	3-4-1	0.2655	0.9645	4.9101	0.0361

Table 4. MEG forecasting with Horizon =1 - Errors on the Test Set

Size Parameter	**Architecture (I-H-O)**	**NRMSE**	**C.C.**	**MSE $\times 10^{-4}$**	**MRE**
0.004	4-9-1	0.1971	0.9805	3.8921	0.0403
0.008	3-3-1	0.2189	0.9757	4.8351	0.0438
0.012	4-5-1	0.2063	0.9786	4.2656	0.0434
0.016	3-2-1	0.2309	0.9733	5.3838	0.0466
0.026	3-3-1	0.2177	0.9765	4.7830	0.0446
0.035	3-4-1	0.2111	0.9775	4.4978	0.0429

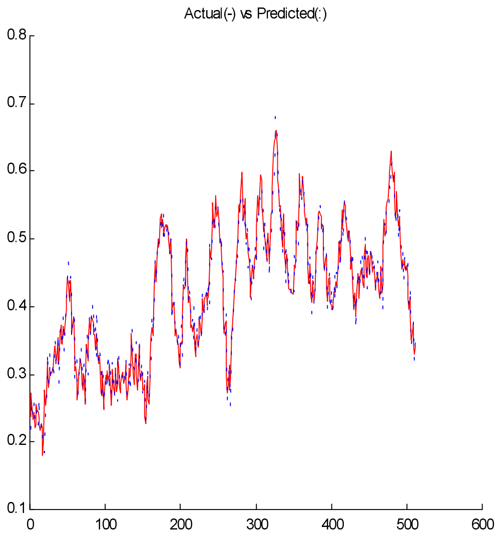

Figure 1. MEG forecasting with Horizon =1

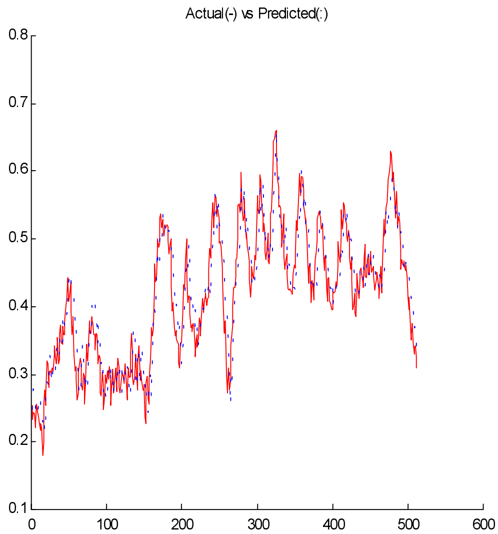

Figure 3. MEG Forecasting with Horizon=3

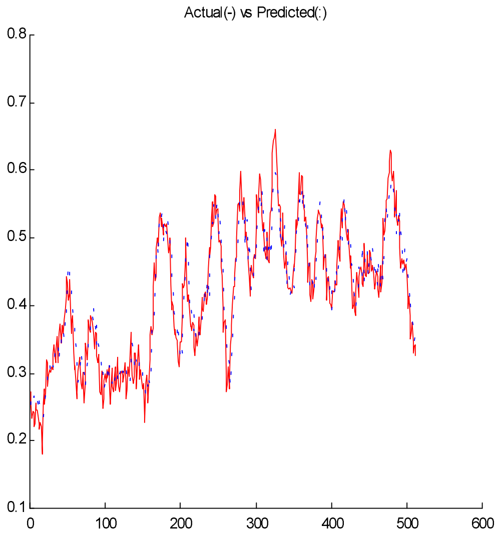

Figure 2. MEG forecasting with Horizon 2

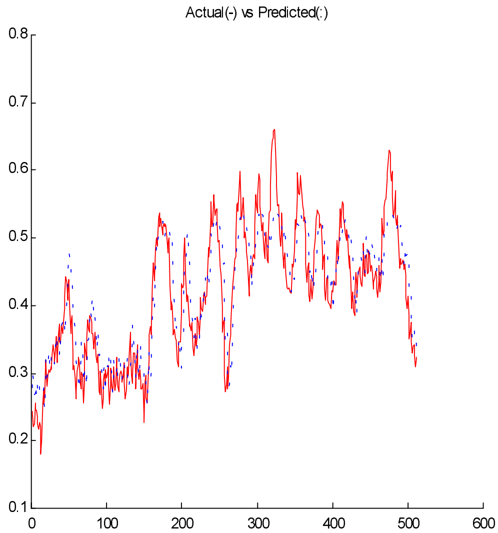

Figure 4. MEG forecasting Horizon=4

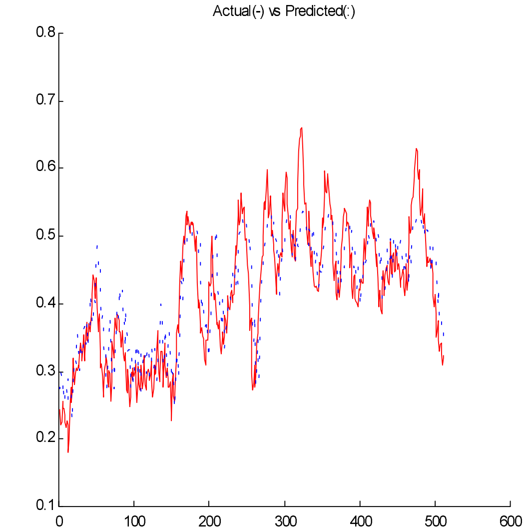

Figure 5. MEG forecasting with Horizon=5

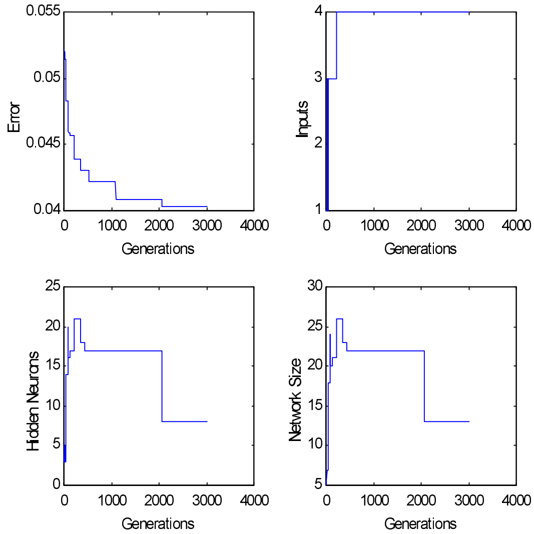

Figure 6. MEG forecasting with Horizon=1

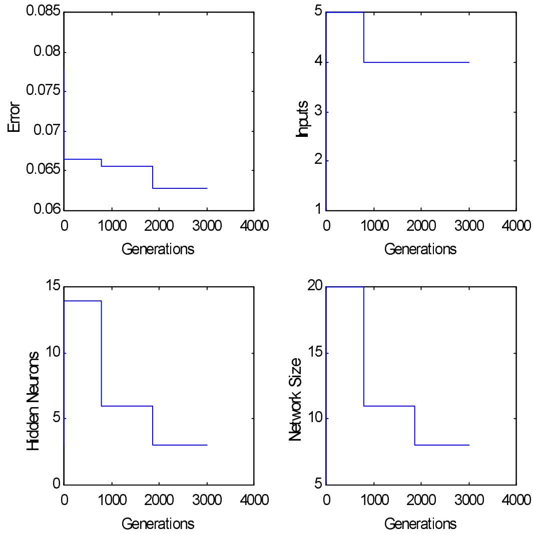

Figure 7. MEG forecasting with Horizon=2

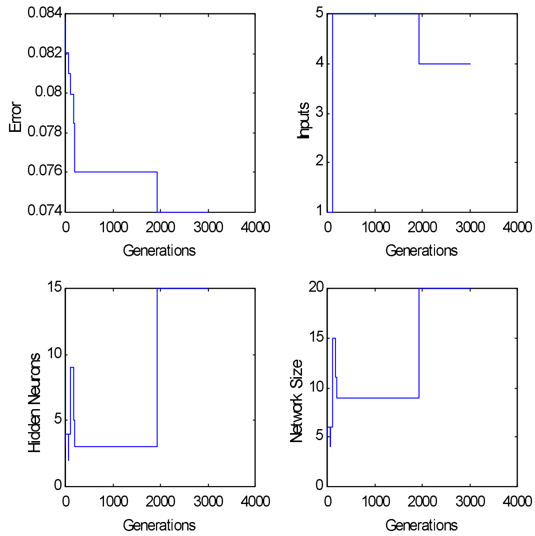

Figure 8. MEG forecasting with Horizon=3

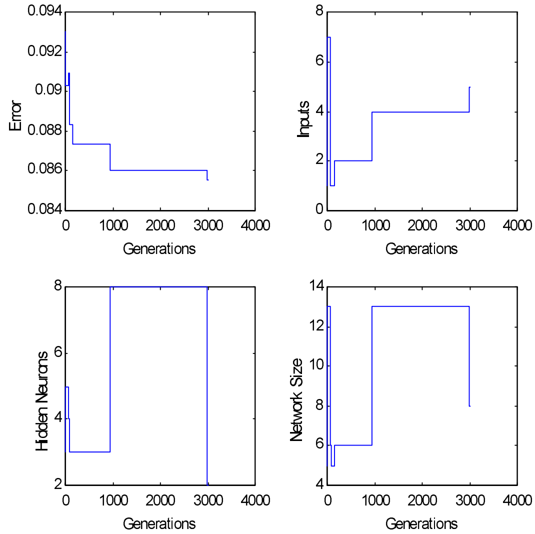

Figure 9. MEG forecasting with Horizon=4

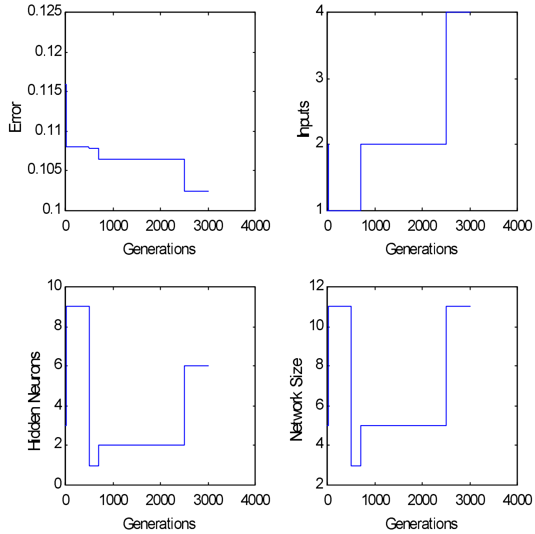

Figure 10. MEG forecasting with Horizon=5

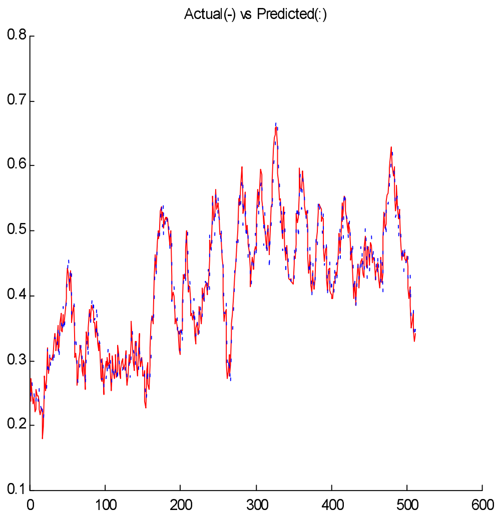

Figure 11. MEG forecasting with Horizon =1 and Size_Param = 0.004

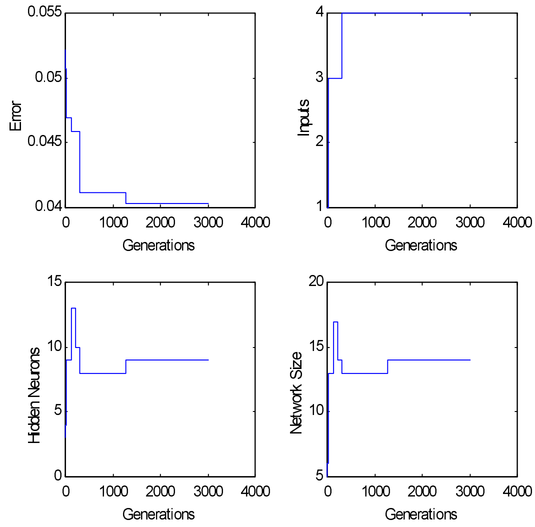

Figure 12. MEG forecasting with Horizon =1 and Size_Param =0.004

4. CONCLUSIONS

In the present work was examined the ability of evolutionary neural networks to predict the MEG of epileptic patients. The utilized algorithm combined a genetic algorithm and a training method based on the localized Extended Kalman Filter applied on Multi-Layer-Perceptrons neural networks. In addition to the MEG predictability, the method was used to investigate the influence of the architecture (size) of the network on its performance.

In the large number of numerical experiments done considering the task of investigating the ability of MEG prediction, different predicting horizons were used. The results of these experiments (some representative of them are listed on Tables 1 and 2) indicated that the smaller the value of the predicting horizon a better prediction of the MEG signal is obtained. Especially for the cases of values of predicting horizon equal to 1 and 2, the prediction can be considered very satisfactory. For larger values of the predicting horizon (4, 5) the error of prediction is larger and a disability of the algorithm to follow the strong peaks of the original MEG timeseries is observed. This is pictorially presented in Figs. 4 and 5 which correspond to the predicted signal for predicted horizons 4 and 5.

On the other hand, considering the second task of this work which referred to the investigation of the influence of the size of the produced networks the obtained results are summarized on Tables 3 and 4. The results shown on that tables indicated that the smaller the size_par the larger and more complicated the structure of the produced networks, which in accordance to the previous results led to more accurate prediction of the signal.

In conclusion, the results of this work indicated that the MEG signals of epileptic patients are predictable by the method presented above. The method presented here is of general purpose and can be used in a wide variety of signals. In future work we will attempt to use this method on different types of biomagnetic signals such as the magnetoencephalogram (MEG) and the magnetocardiogram (MCG) of fetuses as well as adult subjects.

References

Adamopoulos, A., Georgopoulos E., Manioudakis, G. and Likothanassis, S. "An Evolutionary Method for System Structure Identification Using Neural Networks" Neural Computation '98.

Anninos, P. Jacobson, J. Tsagas, N. Adamopoulos, A. (1997). Spatiotemporal Stationarity of Epileptic Focal Activity Evaluated by Analyzing MagnetoEncephaloGraphic (MEG) data and the Theoretical Implications. Panminerva Med. 39, 189-201.

Likothanassis, S. D., Georgopoulos, E. F. and Manioudakis, G., "Currency Forecasting Using Genetically Optimized Neural Networks", HERCMA Athens September 1998.

Michalewicz, Z., "Genetic Algorithms + Data Structures = Evolution Programs", Springer-Verlag, 1996.

Shah, S., Palmieri, F. and Datum, M., "Optimal Filtering Algorithms for Fast Learning in Feed-Forward Neural Networks", *Neural Networks*, Vol. 5, pp. 779-787, 1992.