Publication P3

J. Martikainen and S. J. Ovaska
"Fitness function approximation by neural networks in the optimization of MGP-FIR filters"
in
Proc. of the IEEE Mountain Workshop on Adaptive and Learning Systems
Logan, UT, 2006, pp. 231-236.

Fitness Function Approximation by Neural Networks in the Optimization of MGP-FIR Filters

Jarno Martikainen and Seppo J. Ovaska
Helsinki University of Technology
Institute of Intelligent Power Electronics
Espoo, FI-02015 Finland
E-mail: jkmartik@cc.hut.fi, ovaska@ieee.org

***Abstract*—In this paper we introduce a neural network based method for speeding up the fitness function calculations in a genetic algorithm (GA) -driven optimization process of Multiplicative General Parameter Finite Impulse Response (MGP-FIR) filters. In this case, calculating the fitness of a candidate solution is an extensive and time-consuming task. However, our results show that it is possible to approximate the fitness function components with neural networks up to sufficient degree, thus enabling the genetic algorithm to perform the fitness calculations considerably faster. This allows the algorithm to evaluate larger number of generations in a given time. Our results suggest that it is possible to decrease the approximation error of the neural network so that the NN-assisted GA eventually offers competitive performance compared to a reference GA.**

I. Introduction

In 50/60Hz power systems instrumentation, predictive lowpass and bandpass filters play a crucial role. These signal processing tasks are delay-constrained so that the distorted line voltages or currents should be filtered without delaying the fundamental frequency component. In addition, the line frequency can vary typically up to ±2%, so the filter should be able to adapt to the changing input frequency. For this purpose, Vainio et al. introduced the multiplicative general parameter (MGP) finite impulse response (FIR) filtering scheme in [1] and [2]. The aim of the MGP-FIR is to predict the signal value p steps ahead while simultaneously filtering out noise and harmonic components from the input signal.

Previously MGP-FIRs have been successfully designed [3, 4] using genetic algorithms [5, 6]. These computationally efficient filters are difficult to optimize using traditional methods, such as gradient descent, since no derivative information exists due to the discrete set of filter coefficients, i.e. [-1, 0, 1]. The optimization process, however, is still time-consuming, since the fitness of an individual is determined based on the results of applying the candidate filter to a set of test signals. In this paper, we introduce methods to speed up the GA-assisted MGP-FIR design process by means of neural networks and fitness function redefinition. Instead of three separate test signals, one for each frequency of 49 Hz, 50 Hz, and 51 Hz, to determine the fitness of an individual, only one of these test signals, the 50 Hz signal, is actually used and the calculated parameter values are fed to a neural network (NN) [7] to approximate the parameter values of the two other test signals. Also, the previously used fitness function is improved to better respond to the application's requirements.

Neural networks have been used before for fitness function calculations, for example, in evolving color recipes [8] and designing electric motors [9]. The results presented in this paper suggest that using neural networks for aiding the fitness function calculations helps to create a competitive algorithm for MGP-FIR basis filter optimization.

This paper is structured as follows. Section II describes the theory of MGP-FIRs. Section III explains the optimization schemes used in this paper. Section IV discusses the approximation capabilities of the neural network in this case. Section V contains results and Section VI the related discussion.

II. MGP-FIR

In a typical MGP-FIR, the filter output is computed as

$$y(n) = g_1(n)\sum_{k=0}^{N-1} h_1(k)x(n-k) + g_2(n)\sum_{k=0}^{N-1} h_2(k)x(n-k). \quad (1)$$

Where $g_1(n)$ and $g_2(n)$ present the adaptive MGP parameters, and $h_1(k)$ and $h_2(k)$ are the fixed coefficients of an FIR basis filter. Thus, the coefficients of the composite filter are $_1(k) = g_1(n)\ h_1(k)$, $k \in [0, 1,\ldots, N-1]$, for the first MGP, and, $_2(k) = g_2(n)\ h_2(k)$, $k \in [0, 1,\ldots, N-1]$, for the second MGP. An example of MGP-FIR with N=4 is shown in Fig. 1. Here N denotes the filter length. The adaptive coefficients, $g_1(n)$ and $g_2(n)$, are updated as follows

$$g_1(n+1) = g_1(n) + \mu e(n)\sum_{k=0}^{N-1} h_1(k)x(n-k) \quad (2)$$

$$g_2(n+1) = g_2(n) + \mu e(n)\sum_{k=0}^{N-1} h_2(k)x(n-k) \quad (3)$$

where μ is the adaptation gain factor and $e(n)$ is the prediction error between the filter output and the training signal, i.e.,

1-4244-0166-6/06/$20.00 ©2006 IEEE.

$x(n)$–y(n–p), p being the prediction step. The MGP-FIR has two adaptive parameters to adapt only to the phase and amplitude of the principal frequency. More degrees of freedom would allow the filter to adapt also to undesired properties, such as the harmonic frequencies.

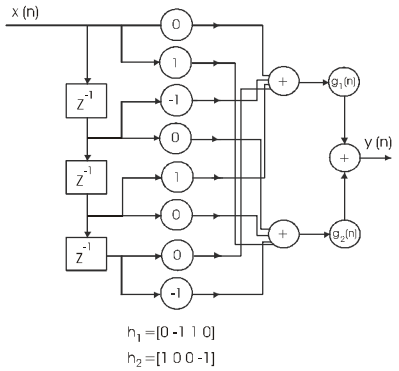

Fig. 1. An example of MGP implementation, where N=4. Signal values $x(n-1)$ and $x(n-2)$ are connected to the first MGP and values $x(n)$ and $x(n-3)$ are connected to the second MGP with filter coefficients -1, 1, 1, and -1, respectively

The basic idea of MGP-FIR filters is that all the samples of input delay line should be connected either to the first or to the second MGP, and no value should be left unused. Computational efficiency of these particular MGP filters arises from the fact that the filter coefficients are either -1, 0, or 1. Thus the number of multiplications in filtering operations is radically reduced compared to a normal filtering operation using more general coefficient values. In this paper, the length of the filter studied was 40.

III. Optimization Schemes

A. The Reference Genetic Algorithm

A standard GA reference was used to optimize the basis filter. The GA used in this paper operates as follows:

1. Create an initial population of 80 individuals.

2. Calculate the fitness of each individual and sort the population in descending order based on the fitness value.

3. Perform mating using single point crossover so that the best individual mates with the second best, the third with fourth and so on. Thus, the 40 best solutions create totally 40 offspring

4. Create the population for the next generation taking all the 40 offspring and using fitness-proportional roulette wheel selection for selecting 40 of the parents.

5. Select each solution for mutation with the probability of 0.05. If a solution is mutated, only a single gene is subjected to mutation.

The fitness function is expressed as

$$fitness = \frac{1 \cdot 10^6}{a \cdot (ITAE49 + ITAE50 + ITAE51)} \tag{4}$$

where

$$a = \max(h49/10 + NG49, h50/10 + NG50, h51/10 + NG51) \tag{5}$$

and

$$ITAE = \sum_{n=1}^{M} n \cdot e_f(n) \,. \tag{6}$$

e_f is the error when comparing the filtered test signal to a pure signal, i.e., a signal containing only the principal frequency component. There are separate test and pure signals for three frequencies, 49 Hz, 50 Hz, and 51 Hz. The test signals contain the fundamental frequency component with the amplitude of 1, odd harmonics from the third to the 15th with amplitudes of 0.1 and uniformly distributed white noise with amplitude of 0.04. In addition

$$NG(n) = \sum_{k=0}^{N-1} \left[g_1(n) \cdot h_1(k) \right]^2 + \sum_{k=0}^{N-1} \left[g_2(n) \cdot h_2(k) \right]^2 . \tag{7}$$

and $h49$, $h50$, and $h51$ correspond to the amplitude of the third harmonic in the filtered test signal. The structure of the fitness functions guides the GA to minimize the amplification of harmonic frequencies of the input signal.

B. Neural Network-Assisted Genetic Algorithm

Evaluating such a fitness function is time-consuming and in order to speed up the fitness calculations the fitness function was rewritten as:

$$fitness = \frac{1 \cdot 10^6}{b \cdot \left(\hat{ITAE}49 + ITAE50 + \hat{ITAE}51 \right)} \,. \tag{8}$$

The $ITAE$ term is now calculated only for the 50 Hz test signal and this value is used to approximate the $ITAE$ values for 49 Hz and 51 Hz signals using a neural network. This way, we need to filter only one third of the test signals available, namely the 50 Hz test signal. Moreover, b is expressed as

$$b = \max(h49/10 + \hat{NG}49, h50/10 + NG50, h51/10 + \hat{NG}51). \tag{9}$$

In *b*, the third harmonic gains are calculated for all the three test frequencies. However, only the noise gain for the 50 Hz test signal needs to be calculated, since noise gains for 49 and 51 Hz test signals were calculated using g_1 and g_2 approximated by a multilayer perceptron network. This neural network consisted of a single hidden layer and 5 hidden neurons. Figure 2 shows a box plot of the neural network's performance using different number of hidden neurons. Clearly, by using 5 hidden neurons the median value (marked by a vertical line) as well as the variance of the results were better than with other number of hidden neurons. These results were calculated using averages of 25 runs. In Fig. 2 + denotes an outlier value.

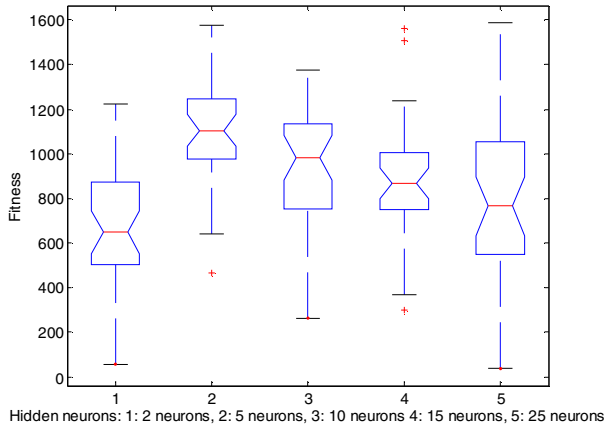

Fig. 2. The effect of the number of hidden neurons on the NN-assisted GA performance.

As inputs the network takes g_1, g_2, and *ITAE* after the 50 Hz test signal has been processed. The outputs of the network are the approximated values of g_1, g_2, and *ITAE* for the 49 Hz and 51 Hz test signals. Thus, the neural network approach aims at reducing the computational time required to evaluate individual's fitness by a theoretical two thirds.

Both the standard GA as well as the neural network enhanced GA operate equally up to the 10th generation. By this time, the NN-GA has collected 5 training and 3 validation samples per generation, i.e., 50 and 30 individuals, respectively. Early stopping rule is used in training of the neural network [7]. Figure 3 shows the effect of *transition point*, i.e., the point after which the fitness function is calculated using the assistance of the neural network.

The task of choosing the transition point is a trade-off between accuracy and computing time: the longer we collect the training and validation data the more likely the network is to produce accurate results. However, the sooner the neural network assisted fitness calculation is implemented the more generations the GA is capable of going through during the rest of the given time. Eventually, 10 generations was chosen to be the transition point. Although the network performs similarly using 10 and 20 generations as the transition point, 10 generations was chosen because the algorithm with this parameter value is capable of evaluating a larger number of generations than using 20 as the transition point. Table I summarizes the average number of generations evaluated by the algorithms using different transition point.

TABLE I. AVERAGE NUMBER OF GENERATIONS EVALUATED DURING A SINGLE 300-SECOND RUN USING A DIFFERENT TRANSITION POINT.

Transition point (generations)	Number of Generations
5	122
10	112
20	96
50	68
Reference GA (no NN involved)	66

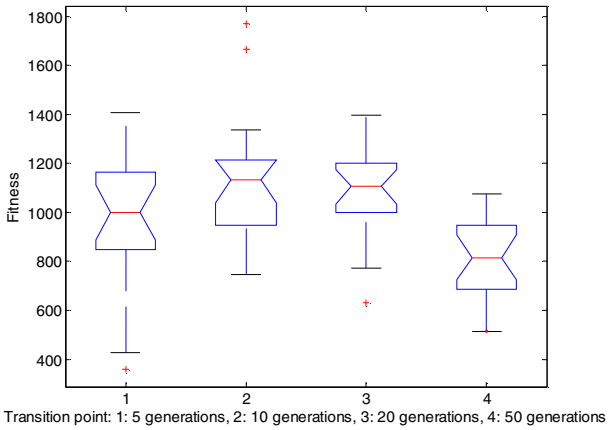

Fig. 3. The effect of the transition point on the NN-assisted GA performance.

Figure 4 shows the principles of the fitness calculations of the reference GA and the NN-assisted GA.

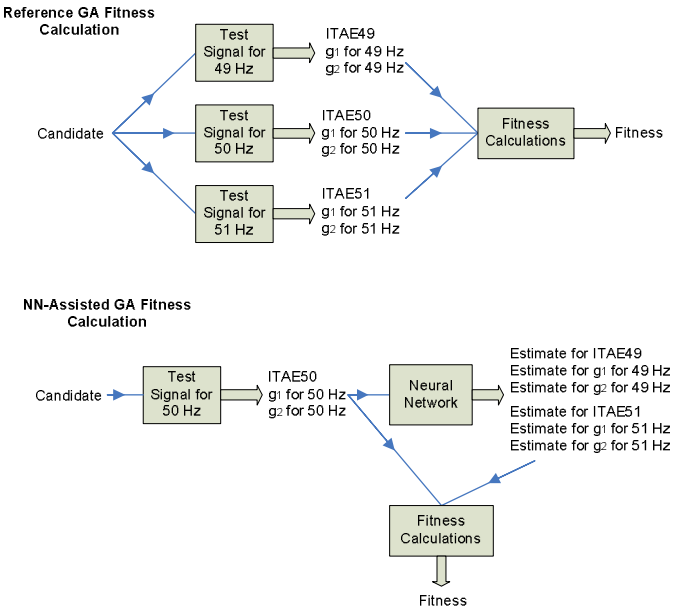

Fig. 4. The principles of the reference GA and the NN-assisted GA fitness calculations.

IV. Approximation Capabilities of the Neural Network

Using neural network to model different components of the fitness function is a trade off between speed and accuracy. When calculating the fitness using neural network, we are not concerned how the network eventually maps the true fitness values as long as the fitness-based order of the candidate solutions remains close to the true order. To tackle the inaccuracy of the fitness order based on the simulated values, a roulette wheel selection was used. This kind of selection scheme enables also less-fit individuals to be chosen for the next generation. In theory, it is possible that a low-level simulated fitness would actually be a high-level true-value fitness. In the following the outputs of the neural networks are compared to the true values. The results are calculated based on the averages of 100 runs.

Figure 5 shows the NN-assisted and real fitness values per generation. The simulated value follows closely the real value at the beginning, but eventually the difference increases. This is likely caused by the fact that due to the evolution process the parameter values enter such regions that were not included in the original training set and thus it is difficult for the NN to approximate the rest of the parameter values precisely.

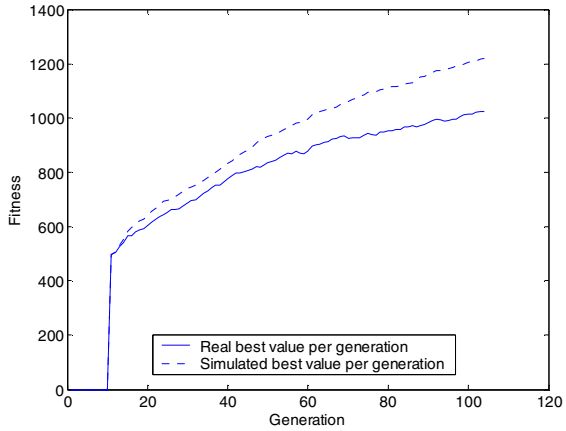

Fig. 5. NN-assisted and real fitness values per generation

Figures 6-9 show the real and simulation results of the MGP values, i.e., g_1 and g_2, for 49 Hz and 51 Hz signals. Similarly to the overall fitness per generation, the simulated and real MGP values are close to each other in the early generations of the run but separate later on. Again, this can be due to the incapability of the original training set to accurately present the whole parameter space confronted during the optimization process.

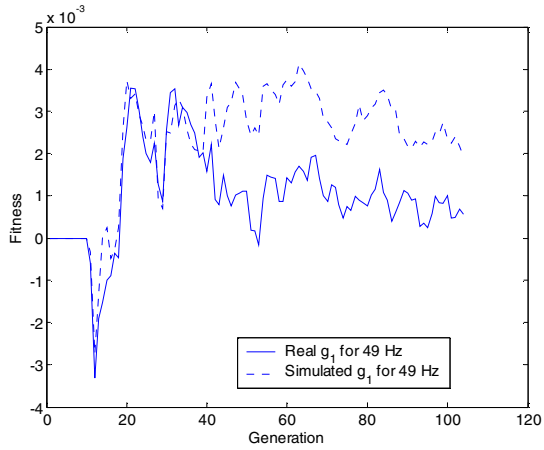

Fig. 6. NN-assisted and real values for g_1 at 49 Hz.

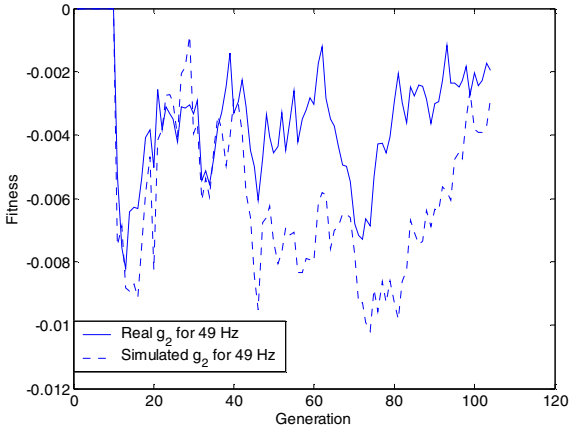

Fig 7. NN-assisted and real values for g_2 at 49 Hz.

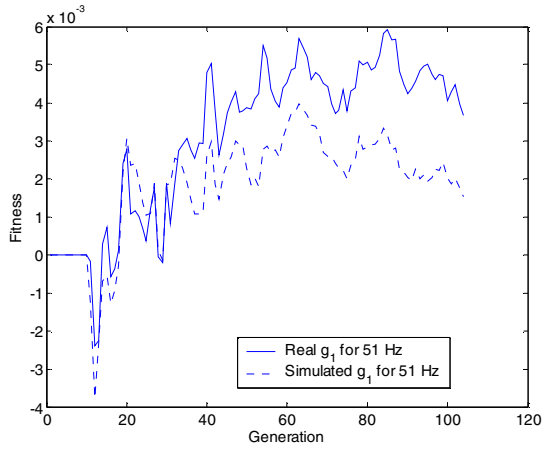

Fig. 8. NN-assisted and real values for g_1 at 51 Hz.

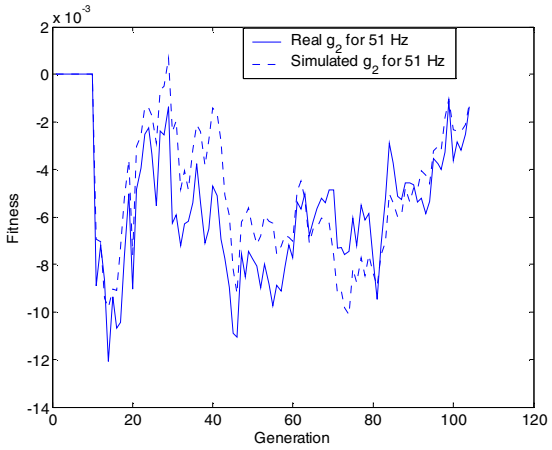

Fig. 9. NN-assisted and real values for g_2 at 51 Hz.

Figures 10 and 11 present real and simulated *ITAE* parameters for 49 Hz and 50 Hz signals. The NN seems to approximate the *ITAE* value for 49 Hz signal well, whereas for the 51 Hz the real and simulated seem to diverge towards the end.

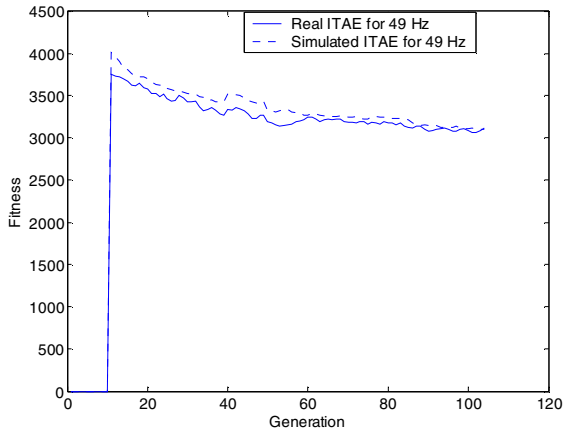

Fig. 10. NN-assisted and real values for *ITAE* at 49 Hz.

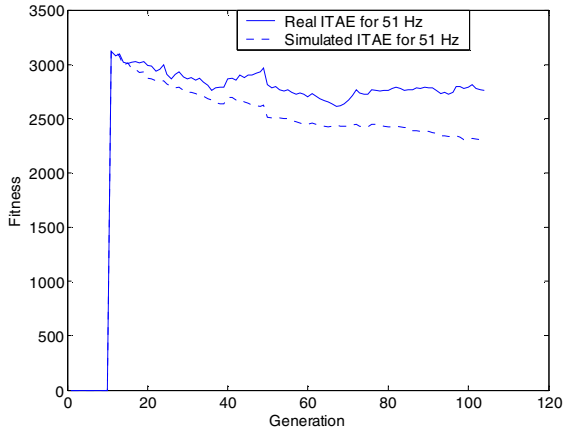

Fig. 11. NN-assisted and real values for *ITAE* at 51 Hz.

Obviously, based on the results, the approximation accuracy of the neural network decreases as the evolution proceeds. To cope with this problem, the training of the network several times during the evolution with new training sets was experimented. These experiments produced results quite similar to that of the NN-assisted GA trained only once and no dramatic improvement in the performance was observed.

Also, the components of the fitness function were approximated using separate neural networks for MGPs for 49 Hz, MGPs for 51 Hz and the *ITAE* parameters. Using these separate networks for different components of the fitness function produced poor results. However, embedding all the components to the same network seems to bind the approximated values together so that no large approximation errors occur. Approximating the parameter values individually using single NN for each could produce better accuracy, but the advantage is lost in more time-consuming calculations.

V. Results

Figures 12 and 13 show the box plots for the averages of 50 individual 300 and 600-second runs. It is clearly visible that the median values are higher in the NN-assisted GA than when using the reference GA. The results of the algorithms should be subjected to a more thorough statistical inspection like the scheme including multiple hypothesis testing and bootstrap resampling [10] to get a more reliable evaluation of the differences between the two algorithms. This kind of scheme, however, requires a lot of data to be collected and in this case a computational time of weeks and it is thus not feasible.

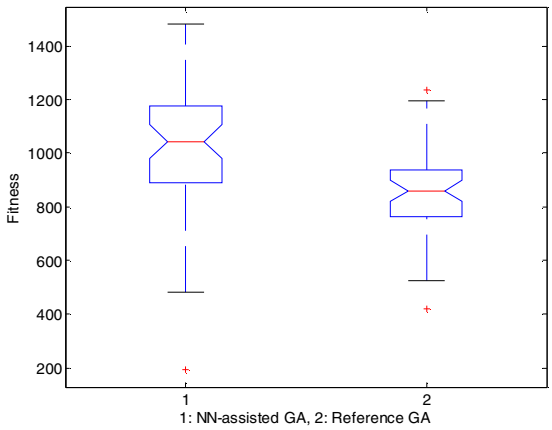

Fig. 12. Box plots for the NN-assisted GA and the reference GA for 50 individual 300-second runs.

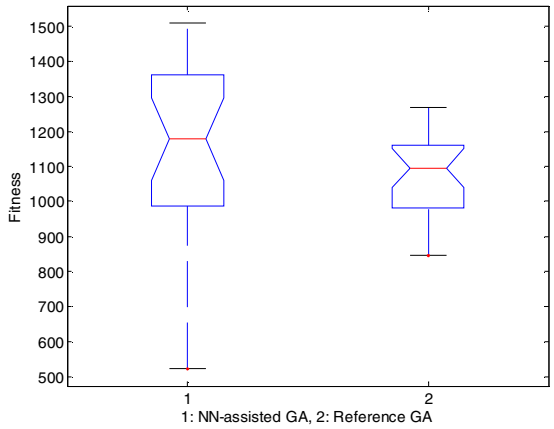

Fig. 13. Box plots for the NN-assisted GA and the reference GA for 50 600-second runs.

These MGP filters are intended for suppressing the harmonics in the input signal. Tables II and III show the performance of filters with median fitness values produced by the different algorithms. The filter performance is expressed as the total harmonic distortion (THD). In table II, THDs are given for filters that after 100 individual 300-second runs have the median fitness values of 1043 and 858 for the NN-assisted GA and the reference GA, respectively.

TABLE II. AVERAGE THD FOR A 300-SECOND RUN.

Harmonic	Amplitude	NN-GA	Reference GA
1st	1	1	1
3rd	0.1	0.055	0.047
5th	0.1	0.031	0.054
7th	0.1	0.018	0.021
9th	0.1	0.009	0.024
11th	0.1	0.012	0.016
13th	0.1	0.017	0.015
15th	0.1	0.021	0.015
THD %	26.46	7.26	8.28

In Table III THDs are given for filters that after 100 individual 600-second runs have the median fitness values of 1178 and 1097 for the NN-assisted GA and the reference GA, respectively.

TABLE III. AVERAGE THD FOR A 600-SECOND RUN.

Harmonic	Amplitude	NN-GA	Reference GA
1st	1	1	1
3rd	0.1	0.059	0.028
5th	0.1	0.030	0.0030
7th	0.1	0.018	0.025
9th	0.1	0.016	0.023
11th	0.1	0.010	0.016
13th	0.1	0.008	0.015
15th	0.1	0.011	0.034
THD %	26.46	7.24	5.99

All the featured filters in Tables 2 and 3 are capable of reducing the THD value of the test signal considerably. In table 3, the THD value of the reference GA is lower than that of the NN-assisted GA although the fitness value of the former is lower. This is caused by the fact that the THD value is actually not part of the fitness function (4) or (8), rather the fitness function consists of other related components.

VI. Discussion and Conclusions

In this paper we have shown how to efficiently model parts of the fitness function calculations of an MGP-FIR basis filter optimization process. Using this method the fitness function calculations are made faster, but this is not without a cost. The accuracy of the NN-approximated fitness function contains approximation error that may affect the final output of the optimization process. However, the approximation error is sufficiently small to enable correct enough ordering of the candidate solutions during the GA optimization process. This way the NN-assisted GA can take advantage of the additional generations run due to the time saved in the fitness function calculations. The resulting algorithm offers competitive performance when compared to conventional GA.

Acknowledgment

This research work was funded by the Academy of Finland under Grant 214144.

References

[1] O. Vainio, S. J. Ovaska, and M. Pöllä, "Adaptive filtering using multiplicative general parameters for zero-crossing detection," *IEEE Transactions on Industrial Electronics*, vol. 50, no. 6, 2003, pp. 1340-1342.

[2] S. J. Ovaska and O. Vainio, "Evolutionary-programming-based optimization of reduced-rank adaptive filters for reference generation in active power filters," *IEEE Transactions on Industrial Electronics*, vol. 51, no. 4, 2004, pp. 910-916.

[3] J. Martikainen and S. J. Ovaska, "Designing multiplicative general parameter filters using adaptive genetic algorithms," in *Proc. of the Genetic and Evolutionary Computation Conference*, Seattle, WA, 2004, pp. 1162-1176.

[4] J. Martikainen and S. J. Ovaska, "Designing multiplicative general parameter filters using multipopulation genetic algorithm," in *Proc. of the 6th Nordic Signal Processing Symposium*, Espoo, Finland, 2004, pp. 25-28.

[5] T. Bäck, *Evolutionary Algorithms in Theory and Practice*. New York, NY: Oxford University Press, 1996.

[6] D. B. Fogel, *Evolutionary Computation: Toward a New Philosophy of Machine Intelligence*. Piscataway, NJ: IEEE Press, 2000.

[7] S. Haykin, *Neural Networks: A Comprehensive Foundation*. 2nd edition. Upper Saddle River, NJ: Prentice Hall PTR, 1998.

[8] E. Mizutani, H. Takagi, D. M. Auslander, and J.-S. R. Jang, "Evolving color recipes," *IEEE Transactions on Systems, Man and Cybernetics, Part C,* vol. 30, no. 4, 2000, pp. 537-550.

[9] B. Dongjin, K. Dowan, J. Hyun-kyo, H. Song-yop, and S. K. Chang, "Determination of induction motor parameters by using neural network based on FEM results," *IEEE Transactions on Magnetics*, vol. 33, no. 2, 1997, pp. 1924-1927.

[10] D. Shilane, J. Martikainen, S. Dudoit, and S. J. Ovaska, "A general framework for statistical performance comparison of evolutionary computation algorithms," In *Proc. of the IASTED International Conference on Artificial Intelligence and Applications*, Innsbruck, Austria, 2006, pp. 7-12.