Progress In Electromagnetics Research M, Vol. 13, 173–188, 2010

ARTIFICIAL NEURAL NETWORKS APPROACH IN MICROWAVE FILTER TUNING

J. J. Michalski

TeleMobile Electronics Ltd.
Pomeranian Science and Technology Park
Gdynia, Poland

Abstract—This paper presents a novel method of cavity filter tuning with the usage of an artificial neural network (ANN). The proposed method does not require information on the filter topology, and the filter is treated as a black box. In order to illustrate the concept, a feed-forward, multi-layer, non-linear artificial neural network with back propagation is applied. The method for preparing, learning and testing vectors consisting of sampled detuned scattering characteristics and corresponding tuning screw deviations is proposed. To collect the training vectors, the machine, an intelligent automatic filter tuning tool integrated with a vector network analyzer, has been built. The ANN was trained on the basis of samples obtained from a properly tuned filter. It has been proved that the usage of multidimensional approximation ability of an ANN makes it possible to map the characteristic of a detuned filter reflection in individual screw errors. Finally, after the ANN learning process, the tuning experiment on 6 and 11-cavity filters has been preformed, proving a very high efficiency of the presented method.

1. INTRODUCTION

At this juncture, after assembling a filter on factory lines there is the necessity of tuning each filter separately. This process is manual work done by an operator, who checks the appropriate parameters of a filter, e.g., the scattering parameters. In order to reach the specification goals, the adjustment of tunable elements has to be executed. In the majority of cases, operators are not microwave engineers, and they compare current (detuned) characteristics with the goal (ideal) characteristics. Based on their experience, the engineers set tuning

Corresponding author: J. J. Michalski (jerzy.michalski@telemobile.pl).

screw combinations on the position that eventually meets technical requirements. It can take a few hours for an unskilled operator to tune, e.g., 6-cavity filter. An engineer gains experience in the course of tuning. After a few months, (having correctly tuned hundreds of filters), the tuning time is decreased to about 20–30 minutes. The global growth of telecommunications systems has a significant influence on the number of produced microwave filters. The market demands the search for new solutions in the filter production process that is low-cost, short term and of high specification parameters. One way to minimize labor costs is to automate the production line.

Numerous different methods concerning filter tuning algorithms were presented in previous publications. In [1], filter tuning in time domain is shown. The described method requires a correctly tuned filter serving as a template and a skilled operator. In [2], a machine learning system is proposed, which employs techniques for pattern recognition and adaptive signal processing. In this method, a skilled operator is still indispensable. The computer-controlled method of tuning microwave filters proposed in [3]. In this work, an approximate filter network model investigating the effects of input/output couplings is used. The automatic tuning for three-pole resonator filter is presented. The novel approach, for filter tuning, is shown in paper [4]. In this paper, an algorithm based on fuzzy logic is introduced, proving that such a method can be helpful in the identification of the tuning elements being the source of the detuning. Interesting tool ROBO-CAT, for automated tuning, was presented in [11]. In this case the algorithms are based on coupling matrix extraction method, time domain and phase cloning. The new approach based on the direct search methods was presented in [5]. It has been proved that the described methods are very effective, but require many screw changes, which is not recommended due to passive inter modulation (PIM) problems.

This paper presents a new approach based on an artificial neural network (ANN). The ANN has been designed in such a way that for a detuned reflection characteristic, at its input, it generates the errors for each screw separately, at the output. As a result, of minimizing errors at the ANN output, the tuning of the filter is executed. The tuning is finished after a single iteration on all tuning elements.

In the paper, the general concept of the algorithm is presented. The main ideas of preparing an ANN that acts as a mapper of detuned characteristics to corresponding screw errors are shown. The physical implementation of the concept and the experimental results, for two filters with different number of cavities, are demonstrated. The conclusion is focused on the summary of the presented method describing strong and weak aspects of the presented approach.

2. GENERAL CONCEPT

Let us denote tuned filter characteristic by $S_0 \in R^M$ and the corresponding normalized tuning element positions by $Z_0 \in R^M$. The relation between tuning element positions Z and filter frequency characteristic S is a function dependence $S = f(Z)$. On the whole, for complex high order filters, the function f is very difficult or even impossible to be defined in an analytical form. In the presented concept, we will construct an operator A: $S \rightarrow \Delta Z$, where ΔZ is the normalized increment of the tuning element positions. The A operator, for each detuned filter frequency characteristic S, will return the normalized increment of the tuning elements ΔZ. After applying ΔZ, on tuning element positions Z, we get a tuned filter with S_0 characteristic. Thus, the increment ΔZ will satisfy $S_0 = f(Z + \Delta Z)$. In the process of construction of the operator A, the pairs of vectors $(S_x, \Delta Z_x)$ will be used, where $\Delta Z_x = Z_0 - Z_x$. The increment ΔZ_x is defined as a difference between screw positions of the correctly tuned filter Z_0 and the screw positions for the detuned filter Z_x, whereas S_x is the corresponding frequency characteristic of detuned filter. Randomly collected vector pairs $(S_x, \Delta Z_x)$ will be used as training vectors for the artificial neural network. Algorithms based on the artificial neural networks require training process, before they are used to perform the tasks they are designed for (like a human gains experience in the course of tuning more filters). In the training process, both the learning vectors and testing vectors are used in order to optimize ANN. The ability of the ANN to do a specified task is measured as a learning error and generalization error. The learning error is defined as the ability to generate the desired outputs for the given input samples, used during the training process of the ANN. Generalization is the ability of the ANN to answer correctly for the inputs not used during the training process. This feature makes the ANN universal multidimensional approximator [6–8]. In the described method, the sampled detuned reflection characteristics of S_{11} will be used as the input vectors. The corresponding outputs will be the normalized screws deviations ΔZ_x (Fig. 1).

Once S characteristic is presented on the ANN input, the screws' deviations ΔZ are generated on its output. After the correction the screws' positions the filter is tuned.

In order to train the ANN, the training vectors must be prepared. The screw deviations can be presented as points in multidimensional space, in a given distance from the center of the system of the

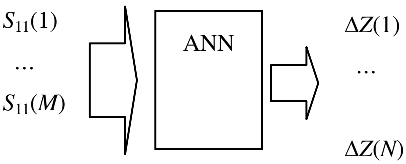

Figure 1. ANN as a mapper of detuned scattering characteristics to screw errors.

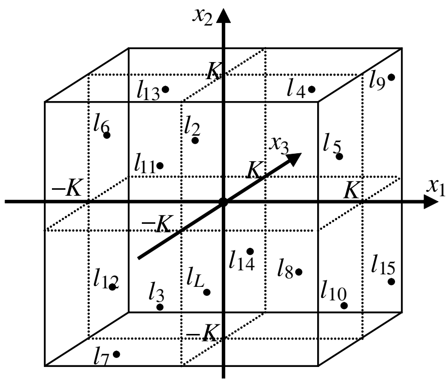

Figure 2. Sampled 3-dimensional search space with the training vectors l_x.

coordinates. The center of the system represents the tuned filter, where the screw's deviations equal zero. All points in this multidimensional space define the search space. In order to prepare the training vectors for the ANN training, the search space must be sampled. From the practical point of view, the numbers of training elements in the training sets must be limited. Nevertheless, the learning elements number must be high enough to ensure the generalization error on a satisfactory level. The sampled search space dimension is determined by the tuning elements number N. The maximum screw deviation is assumed as $\pm K$. Consequently, the search space edge length is defined as $2K$. In Fig. 2, the 3-dimensional search space with the edge $2K$ is presented. This shows the situation, where a 3 tuning elements filter is taken into account. The l_x are the points in the search space, representing detuned filter screws values, during the detuning of the filter. In order to sample the search space a discreet grid of the search space, has been introduced. Here, the one screw-adjustment increment $1u$ as a grid resolution needs to be defined.

2.1. Procedures for Building Training Vectors for an ANN

In order to prepare the training vectors, we propose the following procedures:

1. Use correctly tuned filter,
2. Read the tuned characteristic samples, $S_0(i)$; $i = 1, 2, \ldots M(M = 2{*}256$ points of complex S characteristic) and corresponding screw

positions $W_0(j)$; $j = 1, 2, \ldots N$ of tuned filter (the values of screws positions W, in contrast to positions Z, are not normalized, they are defined as multiple values of u). This screw positions will be used for calculation of screws differences $\Delta W(j)$ (2), for detuned filter, N is the number of the tuning elements of the filter.

3. Randomly detune a filter using the following formula

$$W(j) = W_0(j) + \Delta W(j) \tag{1}$$

where

$$\Delta W(j) = RND[2 * K] - K \tag{2}$$

K is maximum screw deviation as a multiple value of u, and $RND[X]$ is an operator returning random integer value from 0 to X.

4. Read the corresponding S_x of the detuned filter. This characteristic defines the input vector in the ANN learning process.
5. Store S_x as a sample of the input vector and normalized screws increment ΔZ_x, as a sample of the output vector, where

$$\Delta Z_x(j) = \Delta W_x(j)/(2K) + 0.5 \tag{3}$$

is a condition that performs the transformation of screws increments to the domain from 0 to 1.

6. Go to point 3, for creating new learning vector.

According to the above procedures, we obtain the vector pairs defining the relation $S \to \Delta Z_x$. The tuned screws position, either in the process of preparing the training vectors, or during the tuning process, is always the reference position, and this is the main idea of the presented concept. Using these vector pairs, in the ANN learning process, we create the ANN system generating the relative tuning screw errors ΔZ_x, for detuned characteristic S with the respect to positions of the tuned filter. With this definition, if the ANN will respond with zero vector ΔZ_x, the tuning goal is reached.

Following definitions of learning error (4) and generalization error (5), specify the mean error for the all screws. They were applied to check the ANN learning and generalization ability respectively

$$L = 2 * K * \sum_{a=1}^{l} \sum_{b=1}^{N} \left| W_{0L}^{a}(b) - W_{xL}^{a}(b) \right| / (l * N) \tag{4}$$

$$G = 2 * K * \sum_{a=1}^{t} \sum_{b=1}^{N} \left| W_{0T}^{a}(b) - W_{xT}^{a}(b) \right| / (t * N) \tag{5}$$

where, l — learning elements number, t — testing elements number, W_{0L}^{a} – screw positions of the tuned filter, W_{xL}^{a} — the screw positions of the detuned filter used in the process of collecting the learning samples.

Screw positions, of the tuned filter used in detuning process to collect the testing samples, are defined as W_{0T}^{a}. The screw positions, during collecting the testing vectors, are defined as W_{xT}^{a}. If we use the same filter for collecting the learning and testing samples then $W_{0L}^{a} = W_{xT}^{a}$.

From the practical point of view, in order to minimize the generalization error, the training vectors number l should be as high as possible. With the change of training vectors number, the change of the hidden layer neurons number should follow. The exact dependences between training vectors number and neurons number in a hidden layer are very difficult to define. The theoretical estimation can be done using Vapnik-Chervonenkis dimension [9], in practice this relation should be chosen experimentally.

2.2. Selection of Learning Elements Number and Edge Length of Search Space Used for Training Vectors Preparation

The generalization abilities of the ANN depend on many parameters, such as a quality and number of learning vectors, used in the learning process. More of the ANN outputs (more network weights); more learning vectors have to be sampled in the search space. In order to allow the ANN to generalize at a certain level, a proper number of learning elements have to be prepared. This number depends on the dimension N and edge length $2K$ of the search space. It has to be asserted that in order to keep the generalization error at a certain level at changing search space dimension N and/or search space edge $2K$, it is necessary to keep the density of learning samples at the same level. Assuming L_0 as a number of points (learning vectors), in N dimensional search space (N — tuning elements number), the density d_0 can be defined as follows:

$$d_0 = \frac{L_0}{V_0} \tag{6}$$

where

$$V_0 = (2K)^N \tag{7}$$

and V_0 is the volume of N dimensional space, $2K$ is the length of the edge of the N dimensional hypercube in the search space. Applying (7) in (6), the density of L_0 learning samples in N dimensional space can be expressed as follows:

$$d_0 = L_0(2K)^{-N} \tag{8}$$

The relation (8) shows the influence on search space sampling density with the change of the dimension of space (change of the tuning elements number N) and change of the length of the search space edge K.

2.3. Extending the Edge Length of the Search Space

If the search space edge is longer, the filter is more detuned during the preparation of the training vectors, and on the basis of these samples, more detuned filters can be successfully tuned. Below is presented the way of changing the number of learning samples with the change of search space edge in order to get the learning vectors density fixed. Using the definition of the training samples density (6) and volume of the search space defined as (7), we define the dependence on a new number of samples

$$L = L_0 (2k)^N \tag{9}$$

where $k = \frac{K_n}{K_0}$ and K_n is a new half of the search space edge length. Analyzing this dependence, we can conclude that using the samples obtained from the search space with longer edge $2K$, it allows tuning more detuned filters, but it requires more samples to keep density at the same level.

2.4. Extending the Search Space Dimension

Below there are the considerations on how to keep the density of learning vectors fixed d_0 with the change of search space dimension N. Assuming L_0 — as number of samples in N_0 dimensional search space, it is possible to define the density of samples as a d_0 (7). Extending the search space dimension $N = N_0 + 1$, implies growth of a number of samples in the search space with the following formula

$$L = L_0 2K \tag{10}$$

In order to keep the search space sampling density fixed, during the extension of the search space dimension by n as $N = N_0 + n$, the new number of learning vectors must be sampled

$$L = L_0 (2K)^n \tag{11}$$

The more tuning elements a filter has, the more learning vectors have to be sampled in the search space. The new number of learning elements L is an exponential function dependence of the additional tuning screws number n.

3. IMPLEMENTATION OF THE CONCEPT

In order to verify the presented concept, the experiments have been performed. The device being tuned is a 17 cavity diplexer. In general, the diplexer is a passive device, which implements the frequency domain multiplexing. The diplexer topology is depicted above in Fig. 3. The cavities denoted by RX_n represent the RX filter, and TX_n represent TX filter of the diplexer respectively. The RX filter has one cross-coupling between the cavities: #2–#5, and TX filter has two cross-couplings: between the cavities: #2–#5 and #5–#8. All couplings and cross-couplings are fixed.

The two filters are multiplexed onto a third common port at an antenna. In our experiment, the RX filter consists of 6 cavities and TX filter consists of 11 cavities (Fig. 3). In the experiments below, we will tune cavities only, and all the couplings are fixed. For collecting the learning vectors, the intelligent automatic filter tuning tool [10] was used. Block diagram of the tool is presented in Fig. 4, and its physical implementation in Fig. 5.

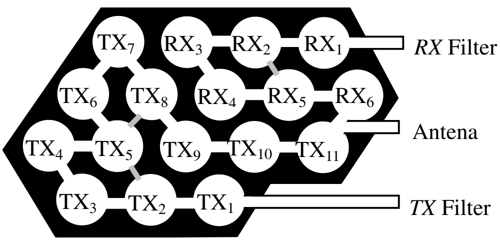

Figure 3. Diplexer used in the experiment.

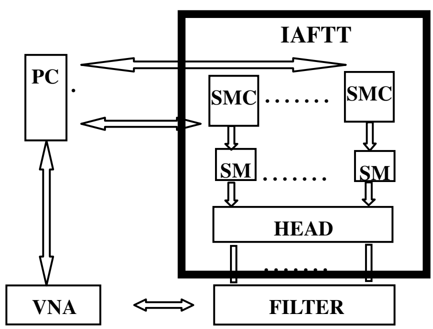

Figure 4. Tuning environment [10] — block diagram (PC — The computer used for ANN processing and reading characteristics from VNA and stepper motors control, SMC — stepper motor controller, SM — stepper motor, HEAD — interconnects stepper motors and filter screws, VNA — vector network analyzer, FILTER — microwave filter).

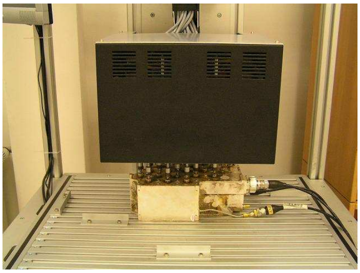

Figure 5. The physical implementation of the tuning environment [10].

This machine can change all filter screws' positions simultaneously. Screw positions are controlled from a PC, and appropriate scattering characteristic is read from the vector network analyzer. The minimal possible angle of the screw change is 1.8°, but for this project it is defined one screw-adjustment increment u equal to 18° of the minimum screw change.

$$1u = 18^\circ \tag{12}$$

In practice the value of $1u$ depends on tuning elements sensitivity and should be chosen experimentally.

3.1. Choosing the ANN Architecture

In order to verify the presented concept, a 3 layered feed-forward (FF) network architecture (input, hidden and output layer) was chosen. The number of neurons for input and output layers is determined by the number of the samples of the reflection characteristics, and the number of tuning elements respectively. The first input layer has 512 neurons (256 complex frequency points). The hidden layer has 50 neurons. The output layer has 6 neurons for RX filter (6 tuning elements), and 11 for TX filter (11 tuning elements). The number of neurons, in the hidden layer, must be chosen experimentally as a compromise between learning time, and learning/generalization ability of the network. For ANN learning the classical error back propagation with momentum method is used. All presented experiment results were obtained using FF network. The concept was also verified using the Radial Basis Function ANN and Fuzzy Logic System (FLS) with subtractive clustering. Since the results obtained for investigated systems were similar, we decided to publish only these for ANN FF.

3.2. Influence of the Learning Vectors Number on Learning and Generalization Errors Level

The investigation has been carried out to verify how many learning elements are necessary to achieve learning and generalization on low enough level errors. During detuning the learning and testing vectors were collected with the maximum screws deviation, assumed as $K = 20u$, which reflects the screw change of the angle of $+-360$ (deg). For each filter (RX, TX) 2000 learning vectors: $\{S_L^l, \Delta Z_L^l\}$, $l = 1, 2, \ldots, 2000$ were collected. The next 100 testing vectors are collected to check the generalization ability during the learning process: $\{S_T^t, \Delta Z_T^t\}$, $t = 1, 2, \ldots, 100$. In the experiment, we will use RX filter of diplexer. In the first step, the ANN was learned using only 25 learning elements, and in the following steps, the number of learning elements was increased up to 1000 elements. The 100 testing vectors in each step were the same.

Figure 6 shows the learning error, and Fig. 7 shows the generalization error of the ANN, number of learning elements l as a parameter. In the case where more learning vectors l are used, to prepare the training examples, the generalization error is decreasing. It can be concluded that using 500 or more learning samples, the learned ANN will answer at the same level, for both samples used in the learning process, and the samples not used for the network learning.

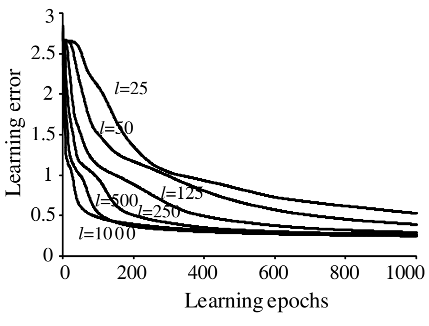

Figure 6. Learning error L in a function of learning epochs. Number of used learning elements l as a parameter.

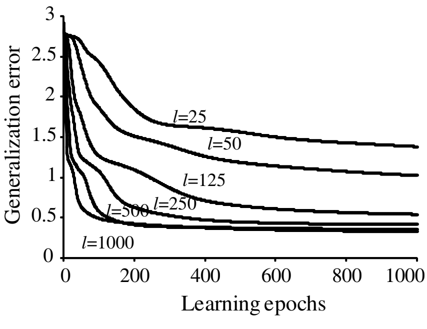

Figure 7. Generalization error G in a function of learning epochs. Number of used learning elements l as a parameter.

3.3. Influence of the Tuning Elements Number on Learning and Generalization Errors Level

In order to check to what extent the presented approach is effective, the process of tuning of both RX and TX filter of diplexer is performed. The ANN architecture is the same as in the previous consideration, 512 input layer neurons, 50 hidden layer neurons and 6 (10) output layer neurons for RX and TX filters respectively. The number of the input vectors for both RX and TX filters is 2000 for each filter. The first 1000 are sampled in the search space of the length $K = 10u$, and the following 1000 with $K = 20u$. The testing set has 100 elements, (50 with $K = 10u$, and 50 with $K = 20u$). The search space has been sampled randomly. The learning and generalization error curves for RX, and TX filters are presented in Fig. 8. Having analyzed the errors' curves, it can be concluded that for an RX filter both errors are at the same level (both curves are undistinguishable), which implies, that the number of samples gathered in the search space is high enough for the ANN training. Observing the errors for a TX filter, it can be noticed that the generalization error is at a higher level than the learning error, which means that the ANN generalization abilities are at a lower level in comparison with the ANN for an RX filter. The main reason of the mentioned difference is the low number of learning vectors sampled in the search space. The TX filter has 11 tuning elements and RX has 6 tuning elements. In order to achieve better generalization abilities of the TX filter, the number of learning vectors should be higher, according to dependence (11). In practice, for each filter type, the training vectors number should be chosen experimentally by successive adding the new vectors and verifying the tuning results.

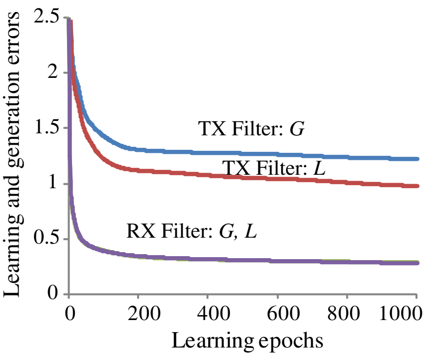

Figure 8. Learning L and generalization G errors for RX and TX filters.

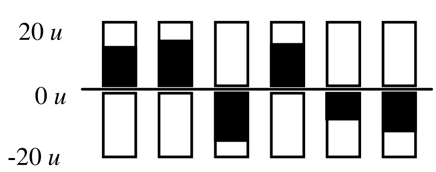

Figure 9. Graphical ANN response of screw errors for a detuned filter characteristic.

3.4. ANN Tuning Experiments

As a result of applying detuned reflection characteristic $S_{11}(i)$ on the ANN input, the network will respond on its output $\Delta Z(j)$. Considering the reverse form of (3), screw errors $\Delta W(j)(u)$ could be determined. Fig. 9 presents graphically the errors for all 6 screws of the detuned RX filter. The graphical form of tuning elements deviation is very helpful and intuitive in the tuning process. The tuning goal is to obtain the zero response on the ANN output.

The tuning process of a filter is the minimization of the error on the screws, successively one by one. In this process, in one tuning step, we will change only one screw, minimizing the error generated by the ANN for this screw exclusively.

Having settled the last screw, into a proper position, we get tuned filter characteristic S_0 and the tuning process is finished. Tables 1

Table 1. Screw deviations in $[u]$ generated by ANN during the tuning steps for RX filter.

Tuning step	1	2	3	4	5	6	7
Screw 1	−17	0	0	0	0	0	0
Screw 2	−15	−15	0	0	0	0	0
Screw 3	13	13	18	0	0	0	0
Screw 4	−13	−12	−12	−12	0	0	0
Screw 5	−6	−4	−3	−3	−4	0	0
Screw 6	16	15	15	15	15	15	0

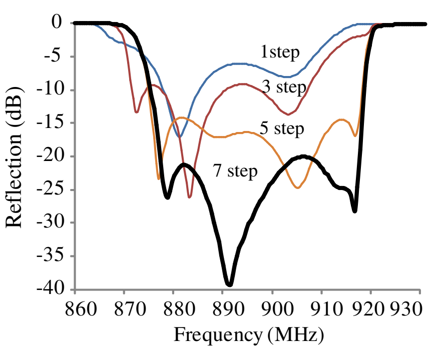

Figure 10. Reflection characteristics during the tuning steps for RX filter.

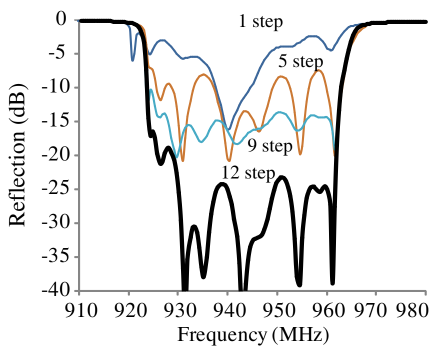

Figure 11. Reflection characteristics during the tuning steps for TX filter.

and 2 show the screw errors of the tuning elements generated on the ANN output in successive tuning steps, for an RX and a TX filter respectively. The corresponding S_{11} characteristics are presented in Figs. 10 and 11. The thin line characteristics are obtained during the tuning steps.

The bold characteristic is obtained after the last step of tuning, and it reflects the situation where the ANN responds with no errors on its output. During the tuning, the order of the screws is of no significance and can be randomly chosen. As a result of the final

Table 2. Screw deviations in $[u]$ generated by ANN during the tuning steps for TX filter.

Tuning step	1	2	3	4	5	6	7	8	9	10	11	12
Screw 1	−10	−8	−5	−8	−9	−10	−10	−10	−10	−9	−8	0
Screw 2	5	6	6	4	1	−2	2	3	3	3	0	0
Screw 3	−14	−16	−15	−16	−15	−15	−15	−15	−15	0	0	0
Screw 4	−10	−4	6	−1	−1	−2	0	2	0	0	0	0
Screw 5	12	12	13	13	12	12	12	0	0	0	0	0
Screw 6	14	13	13	10	12	12	0	0	0	0	0	0
Screw 7	−10	−10	−10	−10	−9	0	0	0	0	0	0	0
Screw 8	−14	−14	−14	−14	0	0	0	0	0	0	0	0
Screw 9	−15	−15	−17	0	0	0	0	0	0	0	0	0
Screw 10	−16	−14	0	0	0	0	0	0	0	0	0	0
Screw 11	8	0	0	0	0	0	0	0	0	0	0	0

tuning characteristic, it can be concluded that the obtained results are significantly effective. The mean manual tuning time for an RX filter (6 tuning elements) is about one minute and for a TX filter (11 tuning elements) — two minutes.

Sometimes, after certain tuning element correction, for example, screw #5 there is a need to readjust a little some screws set before, e.g., #1, #2, #3 or #4, if ANN generates non-zero values on the outputs associated with these screws. Very often, we have the situation during tuning that, for example after tuning the screw #5 the ANN proposes for the screw #4 to correct the value by $+\Delta z$, and after tuning the next screw #6, the ANN generates for the screw #4 back the value $-\Delta z$. It has been investigated that very often after adjusting screw $\#R$ there is no need to readjust screws $\# < R$, even if there are non-zero values generated by ANN on these outputs. Considering this, the tuning algorithm can work in two options. In the first option, ANN generates always the outputs for all tuning elements regardless of the currently tuned screw. In the second option, algorithm masks the ANN outputs $\# < R$, if the screw R is considered for tuning. The Tables 1 and 2 show the situation for the second option of the algorithm.

4. CONCLUSION

The novel filter tuning method, based on the artificial neural networks, has been proposed. One of the most important features of presented approach is that the filter is treated as a black box, no information on the filter topology is necessary in the algorithm customization. The way of preparing input and output vectors, for the ANN training process, has been described. It has been proved and demonstrated that using multidimensional approximation ability of the ANN, it is possible to map the detuned filter scattering characteristics to the corresponding screw errors, creating a very efficient tool for tuning. It has been suggested that the density of the sampled learning vectors, in the search space, influences the ANN generalization error most significantly. The dependences on the learning vectors density with the change of the search space dimension and with the edge length of the search space have been defined. Using this presented method, the mean time of manual tuning of 11 cavities filter takes about 2 minutes, instead of 20–30 minutes by an experienced tuner. With the usage of the introduced intelligent automatic filter tuning tool [10], where all tuning elements can be changed simultaneously, it can take even less than 5 seconds. However, mean time is about 20 seconds. Tuning time of the whole diplexer (both RX and TX filters) is about 1 minute. The performance of alternative solution [11] is about 30 minutes for the diplexer of similar complexity.

Performing the tuning experiments, with large volume of filters, it has been experienced that although ANN responded with zero output after tuning, almost all filters required the small fine tuning. It has been observed that the algorithm works best if the cavities screws are taken in the process of tuning, and the couplings screws are pretuned.

The ANN used in these experiments was learned based on the samples from one filter only of the considered type (RX/TX). This presented method was verified on many filters of the same type. It has been observed that based on such "poor" samples not all other filters were properly tuned. This limitation is the result of filter differences and relates to all tuning methods that can be found in publications. To avoid this disadvantage, in our method, learning samples from more than one filter should be collected. It has been proved that the ANN gains experience if more filters take part in preparing learning samples.

ACKNOWLEDGMENT

The author wishes to thank the team from TeleMobile Electronics, Gdynia, Poland for their generous support, and engagement in the projects of mechanical and electrical aspects of intelligent automatic filter tuning tool [10] as well as Dr. Wlodzimierz Zieniutycz for helpful discussion.

REFERENCES

1. "Simplified filter tuning using time domain," HP Application, Santa Rosa, CA, Note 1287-8, 1999.
2. Mirzai, A. R., C. F. N. Cowan, and T. M. Crawford, "Intelligent alignment of waveguide filters using a machine learning approach," *IEEE Trans. Microwave Theory & Tech.*, Vol. 37, No. 1, 166–173, January 1989.
3. Harscher, P. and R. Vahldieck, "Automated computer-controlled tuning of waveguide filters using adaptive network models," *IEEE Trans. Microwave Theory & Tech.*, Vol. 49, No. 11, 2125–2130, November 2001.
4. Miraftab, V. and R. R. Mansour, "Computer-aided tuning of microwave filters using fuzzy logic," *IEEE Trans. Microwave Theory & Tech.*, Vol. 50, No. 12, 2781–2788, December 2002.
5. Cegielski, T. and J. Michalski, "Heuristic methods for automated microwave filter tuning," *Proceedings of XVII International Conference on Microwave, Radar and Wireless Communications*

MIKON-2008, Vol. 3, 647–650, Poland, Wroclaw, May 19–21, 2008.

6. Amari, S. L., "Mathematical theory of neural learning," *New Generation Computing*, Vol. 8, 281–294, 1991.
7. Hornik, K., M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, Vol. 2, 359–366, 1989.
8. Haykin, S., *Neural Networks: A Comprehensive Foundation*, Prentice Hall, Upper Ssaddle River, NJ, 1999.
9. Vapnik, V. N. and A. Chervonenkis, "On the uniform convergence of relative frequencies of events to their probabilities," *Theory of Probability Appl.*, Vol. 16, 264–280, 1971.
10. Intelligent automatic filter tuning tool (IAFTT) is registered with "Priority certificate #2516" (Hannover Cebit 2007), and patent pending "European Patent Application No. P382895 assigned by Polish National Patent Office". More information on www.trimsolution.eu.
11. Yu, M. and W. C. Tang, "A fully automated filter tuning robots for wireless base station diplexers, workshop computer aided filter tuning," *IEEE Int. Microwave Symposium*, Philadelphia, June 8–13, 2003.