

Efficient Initialization of Artificial Neural Network Weights for Electrical Component Models

Tuomo Kujanpää and Janne Roos

Helsinki University of Technology, Department of Electrical and Communications Engineering, Circuit Theory Laboratory
P.O.Box 3000, FI-02015 TKK, Finland
tuomo.kujanpaa@tkk.fi, janne@ct.tkk.fi

Abstract—Two methods for an initialization of Artificial Neural Network (ANN) weights are experimentally evaluated for electrical component modeling applications. Modification of the second method, based on special target training data scaling, is also presented. The methods are evaluated with respect to average ANN training error, ANN test error, and ANN training CPU time.

Keywords—artificial neural networks, multilayer perceptron, weight initialization, conjugate-gradient, circuit simulation, optimization, component modeling.

I. MOTIVATION

The modeling of RF/microwave components for Computer-Aided Design is facing new challenges because of increasing operation frequencies, circuit complexity, integration density, and decreasing time to market. Recently, it has been shown that Artificial Neural Networks (ANNs) offer solutions to urgent modeling problems encountered with conventional numerical methods (e.g., 3-D EM simulation) and empirical models. Fast and accurate models based on ANNs have been created for a wide range of components [1].

The crucial part in ANN-based modeling is ANN training, that is, optimization of ANN weights with given measurements or, say, 3-D EM simulation data. In [2] several ANN weight-initialization methods were compared mainly by means of classification problems. It was shown that the choice of an initialization method influences the convergence of the optimization and the optimal initial weights are, by some means, determined by the measurement/simulation data set. However, weight-initialization methods have not previously been systematically evaluated for electrical component modeling problems and the nature of the problems — the functions to be approximated — differs significantly from, e.g., classification problems with Boolean/discrete target/input values.

II. ARTIFICIAL NEURAL NETWORKS AND WEIGHT INITIALIZATION

The most widely used ANN in the field of RF/microwave component modeling is the Multi-Layer Perceptron (MLP) [1]. The three-layer MLP used in this work realizes the nonlinear mapping

$$\tilde{y}_l(\mathbf{x}, \mathbf{w}) = w_{l0}^3 + \sum_{j=1}^{N_h} w_{lj}^3 \tanh\left(w_{j0}^2 + \sum_{i=1}^{N_i} w_{ji}^2 x_i\right), \quad (1)$$

$$l = 1, 2, \dots, N_o,$$

where N_i , N_h , and N_o represent the number of inputs, hidden-layer neurons, and outputs, respectively; $\mathbf{x} = (x_1, x_2, \dots, x_{N_i})$, $\tilde{\mathbf{y}} = (\tilde{y}_1, \tilde{y}_2, \dots, \tilde{y}_{N_o})$, and \mathbf{w} represents ANN inputs, outputs, and weights, respectively. The function $\tanh()$ is called an Activation Function (AF).

Let $\mathbf{y} = \mathbf{y}(\mathbf{x})$ be an unknown, nonlinear, multidimensional function to be approximated by the MLP mapping (1): $\tilde{\mathbf{y}} = \tilde{\mathbf{y}}(\mathbf{x}, \mathbf{w})$. Let $\{(\mathbf{x}^k, \mathbf{y}^k), k = 1, 2, \dots, N_{tr}\}$ be an appropriate training set, N_{tr} being the number of samples, and the training-set inputs and outputs being scaled linearly in the range $[-1, 1]$. Furthermore, let us define the normalized training error as

$$E_{tr}(\mathbf{w}) = \sqrt{\frac{1}{N_{tr}N_o} \sum_{k=1}^{N_{tr}} \sum_{l=1}^{N_o} \left(\frac{\tilde{y}_l(\mathbf{x}^k, \mathbf{w}) - y_l^k}{2} \right)^2}. \quad (2)$$

The training of the ANN means minimizing $E_{tr}(\mathbf{w})$ with respect to the weights, \mathbf{w} , using a suitable optimization method — in this work, Hestenes–Stiefel conjugate-gradient with Error Back Propagation (EBP) [3]. The generalization capability of the trained ANN is evaluated by applying Eq. (2) to an independent test set, $\{(\mathbf{x}^k, \mathbf{y}^k), k = 1, 2, \dots, N_{te}\}$ to obtain $E_{te}(\mathbf{w})$.

The weight initialization tries to provide initial weight values close to the global minimum of $E_{tr}(\mathbf{w})$. There are several strategies for initializing the MLP weights, the most developed of which can also be regarded as training methods [4]. However, the most widely utilized strategy for ANN-based RF/microwave component modeling is, still, initializing the weights as random real numbers from a Uniform Distribution (UD) with fixed range. The weight-initialization methods evaluated in this work include: 1. random initialization from UD with fixed range [1], and 2. random initialization from UD with variable range and special input data scaling [5].

Utilizing the first method, one sets $w_{ji}, w_{lj} \in [-a, a]$, where, e.g., $a = 0.5$. This heuristic initialization tries to ensure the inner sum (v_j) of the AFs ($\tanh(v_j)$) in (1) to be such that it forces the AFs to operate in an approximately linear transition region determined by maximum curvature points $\max(\partial^2 \tanh(v_j)/\partial v_j^2)$. This would be desirable for the convergence of optimization because, when using EBP, $\partial E_{tr}^2/\partial \mathbf{w} \sim \partial \tanh(v_j)/\partial v_j$ and the latter has its maximum value in the transition region. However, the heuristic weight initialization does not take into account the standard deviation σ_{x_i} and therefore AFs may operate in saturation regions slowing down the optimization [5].

The second method forces the AFs, defined as $\tanh()$ in [5], too, to operate in the transition region. This is due to a special input data scaling, with $\bar{x}_i = 0$ and $\sigma_{x_i} = 1$. The AFs are forced to operate in the transition region, with $w_{ij} \in [-a, a]$, where $a = \sqrt{3/N_i}$ [5]. The weights of the output layer neurons are initialized such that $w_{lj} \in [-\sqrt{3/N_h}, \sqrt{3/N_h}]$.

When one utilizes method 2 and approximates the transition regions of AFs as straight lines going through the origin with slope 1, the maximum curvature points are $(-1, -1)$ and $(1, 1)$ for the AFs defined in [5]. The distribution parameters of the MLP outputs (\bar{y}_l) are then $\bar{y}_l = 0$ and $\sigma_{\bar{y}_l} = 1$ as for MLP inputs x_i . A hypothesis to be tested is presented: if one scales the target training data, y_l , such as $\bar{y}_l = 0$ and $\sigma_{y_l} = 1$, the convergence of optimization will be improved. *The hypothesis will be tested in the near future evaluation of the methods.*

III. EXPERIMENTAL SETUP

In the evaluation, we had seven modeling problems: 1. approximation of a modulated sinusoidal function, 2. the same problem with additive normal-distributed noise, 3. MEMS gas-damper behavior, 4. rounded-stripline-bend parallel capacitance and series inductance vs. device geometries, 5. JFET DC characteristics, 6. spiral-inductor S-parameters vs. geometries, and 7. gate and drain currents of MESFET vs. bias current and temperature. The corresponding seven appropriately sized MLPs (N_i , N_h , N_o), the resulting number of ANN weights, i.e., optimization variables (N_w), the number of training-set samples (N_{tr}), and the resulting number of optimization goals ($N_g = N_{tr}N_o$) are shown in Table I.

TABLE I
MODELING-PROBLEM CHARACTERIZATION

problem	N_i	N_h	N_o	N_{tr}	N_w	N_g
1	1	5	1	20	16	20
2	1	5	1	20	16	20
3	3	10	1	40	51	40
4	3	10	2	50	62	100
5	2	10	3	306	63	918
6	5	15	5	486	170	2430
7	3	15	2	37597	92	75194

For each problem and weight initialization method — method 1 with $a = 0.1, 0.5, 1.0$ and method 2 — the number of optimization cycles was set at four different values depending on the problem. Then MLP was trained 30 times, and E_{tr} , E_{te} , and CPU time noted. The results obtained were averaged over all runs at each value of the optimization cycles. A total number of 3360 runs were carried out by semi-automatic scripts using APLAC 8.1 ANNModelGenerator [6] on an Ia64 HP Server rx5670 with a 1.3 GHz processor and 4 Gbyte memory.

IV. ANALYSIS OF RESULTS AND CONCLUSIONS

A set of representative results is shown in Figs. 1 and 2. According to the evaluation, there is no significant

training error performance difference between the methods (Fig. 1). Also, the CPU time performance of the methods was similar and almost linear as a function of optimization cycles. Instead, the test-error performance of method 2 is much better on the average than the one of method 1 with different a values (Fig. 2). The evaluation executed provides motivation for a further research of the methods and a deeper analysis of the results.

In the near future, we will test the hypothesis presented for the modification of the method 2. Also, the training/optimization will be done in hundred-step increments, MLPs for different modeling problems will be trained more than 30 times, and, for the deeper analysis, the standard deviations of E_{tr} , E_{te} and CPU time will be calculated and plotted.

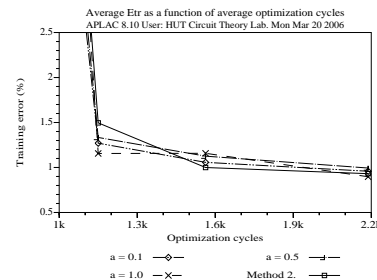


Fig. 1. Average training error vs. average optimization cycles.

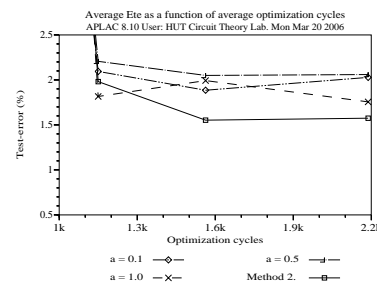


Fig. 2. Average test error vs. average optimization cycles.

ACKNOWLEDGMENT

This work was funded by Nokia Corporation and AWR-APLAC Corporation through TEKES/ELMO/MOSAICS project (grants 2078/31/03 and 2440/31/03).

REFERENCES

- [1] Q. J. Zhang and K. C. Gupta, *Neural Networks for RF and Microwave Design*, Artech House Inc., 2000.
- [2] G. Thimm and E. Fiesler, "High-order and multilayer perceptron initialization," *IEEE Trans. on Neural Networks*, vol. 8, no. 2, pp. 349–359, 1997.
- [3] T. Kujanpää, J. Roos, and M. Honkala, "Experimental comparison of optimization methods in ANN training," *Proceedings of PRIME 2005*, vol. 2, pp. 430–433, 2005.
- [4] D. Erdogmus, O. Fontenla-Romero, J. C. Principe, and A. Alonso-Betanzos, "Linear-least-squares initialization of multilayer perceptrons through backpropagation of the desired response," *IEEE Trans. on Neural Networks*, vol. 16, no. 2, pp. 325–337, 2005.
- [5] S. Haykin, *Neural Networks: A Comprehensive Foundation*, Prentice Hall, 1999.
- [6] *APLAC 8.1 Manuals*, AWR-APLAC Corporation, 2005.