Accurate initialization of neural network weights by backpropagation of the desired response

Conference Paper · August 2003

DOI: 10.1109/IJCNN.2003.1223715 · Source: IEEE Xplore

6 authors, including:

Oscar Fontenla-Romero
University of A Coruña
93 PUBLICATIONS 958 CITATIONS
SEE PROFILE

Jose C Principe
University of Florida
1,085 PUBLICATIONS 18,677 CITATIONS
SEE PROFILE

Amparo Alonso-Betanzos
University of A Coruña
228 PUBLICATIONS 2,566 CITATIONS
SEE PROFILE

Enrique Castillo
Universidad de Cantabria
585 PUBLICATIONS 8,525 CITATIONS
SEE PROFILE

Accurate Initialization of Neural Network Weights by Backpropagation of the Desired Response

Deniz Erdogmus[1], Oscar Fontenla-Romero[2], Jose C. Principe[1],
Amparo Alonso-Betanzos[2], Enrique Castillo[3], Robert Jenssen[1]
[1] Electrical Engineering Department, University of Florida, Gainesville, FL 32611, USA
[2] Department of Computer Science, University of A Coruña, 15071 A Coruña, Spain
[3] Department of Applied Mathematics, University of Cantabria, 39005, Santander, Spain

Abstract – **Proper initialization of neural networks is critical for a successful training of its weights. Many methods have been proposed to achieve this, including heuristic least squares approaches. In this paper, inspired by these previous attempts to train (or initialize) neural networks, we formulate a mathematically sound algorithm based on backpropagating the desired output through the layers of a multilayer perceptron. The approach is accurate up to local first order approximations of the nonlinearities. It is shown to provide successful weight initialization for many data sets by Monte Carlo experiments.**

I. Introduction

Due to the nonlinear nature of neural networks, training requires the use of numerical nonlinear optimization techniques. Practically feasible training algorithms are usually susceptible to local optima and might require parameter fine-tuning. There are various approaches undertaken to find the optimal weights of a neural network. These include first- and second-order descent techniques, which are mainly variants of gradient [1], natural gradient [2], and Newton [3] optimization methods. Although higher order search techniques could speed up convergence at the cost of complexity, they are still vulnerable to local minima. Global search procedures, such as random perturbations [4], genetic algorithms [5], and simulated annealing [6] are not feasible for practical applications due to time constraints. Alternative approaches to help multilayer perceptrons (MLP) learn faster and better include *statistically proper* weight initialization [7,8], and approximate optimization through heuristic least squares application [9, 10]. Although there are many other references to list, we cannot go into such a detailed review of the state-of-the-art in MLP initialization and training, mainly due to the limited space available.

As mentioned, approximate least squares solutions have been previously proposed to initialize or train MLPs. However, these methods mostly relied on minimizing the mean square error (MSE) between the signal of an output neuron before the output nonlinearity and a modified desired output, which is exactly the actual desired output passed through the inverse of the nonlinearity. This approach, unfortunately does not consider the scaling effects of the nonlinearity slope on the propagation of the MSE through the nonlinearity. Nevertheless, they provided the invaluable inspiration for the work presented in this paper, which takes into account the effect of weights and nonlinearities on the propagation of MSE through the network. Specifically, we present an algorithm, which we named *backpropagation of the desired response* that can initialize the weights of an MLP to a point with very small MSE. This algorithm is an approximation of the nonlinear least squares problem with linear least squares and is accurate up to the first-order term in the Taylor series expansion. We considered including higher order terms in the expansion, but then the utility of linear least squares method is not possible.

In this paper, we first present two theoretical results that form the basis for the backpropagation of the desired response algorithm. Then, we provide the algorithm and demonstrate its performance with Monte Carlo experiments.

II. Theoretical Results

Notice that in the L-layer MLP architecture shown in Fig. 1 there are two parts that need to be investigated to achieve successful backpropagation of the desired output through the layers: linear weight matrix and neuron nonlinearity. For our algorithm, we require this nonlinearity to be invertible at every point in its range. We use the following notation to designate signals: the output of the l^{th} layer is $\mathbf{z}^l$ and $\mathbf{y}^l$ before and after the nonlinearity. The weight matrix and the bias vector of this layer are $\mathbf{W}^l$ and $\mathbf{b}^l$, respectively. The input vector is $\mathbf{x}$. The number of neurons in a layer is denoted by n_l and n_0 is the number of inputs. The training set consisting of N input-desired pairs is given in the form $(\mathbf{x}_t, \mathbf{d}_t^L)$. The backpropagated desired output for the l^{th} layer is denoted by $\mathbf{d}^l$ at the output of the nonlinearity and $\overline{\mathbf{d}}^l$ at the input of the nonlinearity.

A. Backpropagating Through a Nonlinearity

Consider a single-layer nonlinear network for which the output is obtained from $\mathbf{z} = \mathbf{W}\mathbf{x} + \mathbf{b}$ and $\mathbf{y} = \mathbf{f}(\mathbf{z})$, where $\mathbf{f}(.)$ is a vector-valued nonlinear function, invertible on its range. Assume that the objective is to minimize a weighted MSE cost function defined on the error between $\mathbf{y}$ and $\mathbf{d}$. Let $\mathbf{H}$ be

This work is partially supported by NSF grant ECS-9900394 and the Xunta de Galicia project PGIDT-01PXI10503PR.

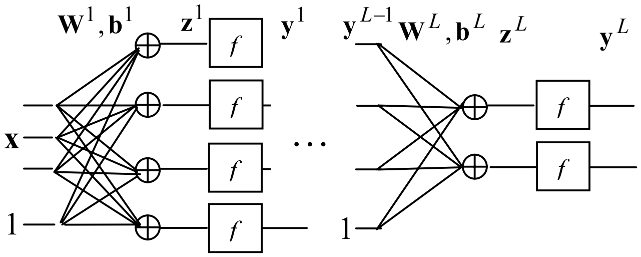


Fig. 1. MLP structure and variables

the weighting matrix. Then Lemma 1 describes the backpropagation of **d** through **f**(.).

Lemma 1. Let $\mathbf{d}, \mathbf{y}, \mathbf{z}, \overline{\mathbf{d}} \in \Re^n$ be the desired and actual outputs, $\mathbf{W} \in \Re^{nxm}$ and $\mathbf{b} \in \Re^{nx1}$ be the weight matrix and the bias vector, and $\mathbf{f}, \mathbf{f}^{-1}, \mathbf{f}' : \Re^n \rightarrow \Re^n$ be the nonlinearity, its inverse and its derivative. Then the following equivalence between two optimization problems is accurate up to the first order of Taylor series expansion.

$$\min_{\mathbf{W},\mathbf{b}} E[(\mathbf{d}-\mathbf{y})^T \mathbf{H}(\mathbf{d}-\mathbf{y})] \equiv \min_{\mathbf{W},\mathbf{b}} E[(\mathbf{f}'(\overline{\mathbf{d}}).*\overline{\boldsymbol{\varepsilon}})^T \mathbf{H}(\mathbf{f}'(\overline{\mathbf{d}}).*\overline{\boldsymbol{\varepsilon}})] \quad (1)$$

where '.*' denotes element-wise vector product, $\overline{\mathbf{d}} = \mathbf{f}^{-1}(\mathbf{d})$, and $\overline{\boldsymbol{\varepsilon}} = \overline{\mathbf{d}} - \mathbf{z}$.

Proof. Recall that **y=f**(**z**) and $\mathbf{d} = \mathbf{f}(\overline{\mathbf{d}})$. Substituting the first-order expansion $\mathbf{f}(\mathbf{z}) = \mathbf{f}(\overline{\mathbf{d}} - \overline{\boldsymbol{\varepsilon}}) \approx \mathbf{f}(\overline{\mathbf{d}}) - \mathbf{f}'(\overline{\mathbf{d}}).*\overline{\boldsymbol{\varepsilon}}$, we obtain the result. Due to space restrictions, we do not present this proof in detail. □

According to this lemma, when backpropagating the desired response through a nonlinearity, the sensitivity of the output error with respect to the slope of the nonlinearity at the operating point should be taken into consideration. Simply minimizing the MSE between the modified desired $\overline{\mathbf{d}}$ and **z** is not equivalent to minimizing the MSE between the **d** and **y**. Note that if $\mathrm{var}(\|\overline{\mathbf{d}}\|)$ is also small, then since the operating point of the nonlinearity is almost fixed for all samples, the $\mathbf{f}'$ terms become redundant. Previous applications of least squares to MLPs did not consider the variance of **d** and the correction scale factor based on the derivative of the nonlinearity at the operating point.

B. Backpropagating Through Linear Weight Layer

Consider a linear network given by $\mathbf{z} = \mathbf{W}\mathbf{x} + \mathbf{b}$ and let $\overline{\mathbf{d}}$ be the desired output. In this scheme, we assume that the weights **W** and **b** are fixed already. The objective is to find the best input vector **x** that minimizes the output MSE. In the MLP context, the input vector will correspond to the output of the nonlinearity of the preceding layer. The result regarding the optimization of **x** in this situation is summarized in the following lemma.

Lemma 2. Let $\mathbf{d}, \mathbf{x} \in \Re^m, \overline{\mathbf{d}}, \mathbf{z} \in \Re^n$ be the desired signals and the corresponding output signals, $\mathbf{W} \in \Re^{nxm}$ and $\mathbf{b} \in \Re^{nx1}$ be fixed weights. Then the following equivalence between the optimization problems holds.

$$\begin{aligned} &\min_{\mathbf{x} \in D \subset \Re^{mx1}} E[(\overline{\mathbf{d}} - \mathbf{z})^T \mathbf{H}(\overline{\mathbf{d}} - \mathbf{z})] \\ &\equiv \min_{\mathbf{x} \in D \subset \Re^{mx1}} E[(\mathbf{d} - \mathbf{x})^T \mathbf{W}^T \mathbf{H}\mathbf{W}(\mathbf{d} - \mathbf{x})] \end{aligned} \quad (2)$$

where D is the set of allowed input values. In the MLP context, this set is determined by the output range of the nonlinearities in the network.

Proof. The proof of this result is very similar to the derivation of the least squares solution for a vector from an overdetermined (or underdetermined) system of linear equations. Due to space restrictions, we do not present this proof in detail. □

In the application of this lemma, two situations may occur: if $n \geq m$, then $\mathbf{d} = (\mathbf{W}^T\mathbf{H}\mathbf{W})^{-1}\mathbf{W}^T\mathbf{H}(\overline{\mathbf{d}} - \mathbf{b})$; if $n < m$ then the desired input **d** can be determined using QR factorization as the minimum norm solution to the underdetermined linear system of equations $\mathbf{W}\mathbf{d} + \mathbf{b} = \overline{\mathbf{d}}$ [11]. In both cases, in an MLP setting, given a desired signal $\overline{\mathbf{d}}^l$ for $\mathbf{z}^l$, we can determine $\mathbf{d}^{l-1}$ as the desired output for the preceding layer. output (after the nonlinearity) of the previous layer. The latter can then be backpropagated through the nonlinearity of layer l-1 as described in Lemma 1.

III. Optimizing The Weights Using Least Squares

Once the desired output is backpropagated through the layers, the weights of each layer can be optimized (approximately) using linear least squares. The following problem treats the optimization of the weights taking the two lemmas of the previous section into account.

Problem 1. Given a linear layer **z** = **Wx** + **b** with $\mathbf{W} \in \Re^{nxm}$ and $\mathbf{b} \in \Re^{nx1}$, the training data in the form of pairs, i.e., $(\mathbf{x}_s, \overline{\mathbf{d}}_s)$ $s = 1, \ldots, N$, and a matrix **G** as the weighting matrix for least squares. Define the error for every sample of the training data for each output of the network as

$$\overline{\boldsymbol{\varepsilon}}_{js} = \overline{\mathbf{d}}_{js} - \mathbf{z}_{js} \quad j = 1, \ldots, n \ , \quad s = 1, \ldots, N \quad (3)$$

where the outputs are evaluated using

$$\mathbf{z}_{js} = \mathbf{b}_j + \sum_{i=1}^{N} \mathbf{W}_{ji}\mathbf{x}_{is} \ , \quad j = 1, \ldots, n \ , \quad s = 1, \ldots, N \quad (4)$$

with $\mathbf{x}_{is}$ denoting the i^{th} entry of the input sample $\mathbf{x}_s$. The optimal weights for this layer of the MLP under consideration, according to the arguments in Lemmas 1 and 2 become the solution to the following minimization problem.

$$\min_{\mathbf{W},\mathbf{b}} J = \frac{1}{N} \sum_{s=1}^{N} \sum_{i=1}^{n} \sum_{j=1}^{n} \mathbf{G}_{ij} f'(\overline{\mathbf{d}}_{is}) f'(\overline{\mathbf{d}}_{js}) \overline{\boldsymbol{\varepsilon}}_{is} \overline{\boldsymbol{\varepsilon}}_{js} \quad (5)$$

Solution. The minimization problem in (5) is quadratic in the weights, therefore, taking the gradient and equating to zero yields a system of linear equations. These equations are easily found to be ($l = 1,...,m$, $k = 1,...,n$)

$$\begin{aligned}&\sum_{i=1}^{n} b_i\gamma_{ik}\left[\sum_{s=1}^{N} f'(\bar{d}_{ks})f'(\bar{d}_{is})x_{ls}\right]\\&\quad+\sum_{p=1}^{m}\sum_{i=1}^{n} w_{ip}\gamma_{ik}\left[\sum_{s=1}^{N} f'(\bar{d}_{ks})f'(\bar{d}_{is})x_{ls}x_{ps}\right]\\&\qquad=\sum_{i=1}^{n}\gamma_{ik}\left[\sum_{s=1}^{N} f'(\bar{d}_{ks})f'(\bar{d}_{is})x_{ls}d_{is}\right]\\&\sum_{i=1}^{n} b_i\gamma_{ik}\left[\sum_{s=1}^{N} f'(\bar{d}_{ks})f'(\bar{d}_{is})\right]\\&\quad+\sum_{p=1}^{m}\sum_{i=1}^{n} w_{ip}\gamma_{ik}\left[\sum_{s=1}^{N} f'(\bar{d}_{ks})f'(\bar{d}_{is})x_{ps}\right]\\&\qquad=\sum_{i=1}^{n}\gamma_{ik}\left[\sum_{s=1}^{N} f'(\bar{d}_{ks})f'(\bar{d}_{is})d_{is}\right]\end{aligned} \tag{6}$$

The unknowns in this square system with $n \cdot m + n$ equations of (6) are the entries of **W** and **b**. This system of equations can easily be solved using a variety of computationally efficient approaches. The weight matrix **G** allows one to take into account the magnifying effect of the succeeding layers on the error of the specific layer. The derivatives of the nonlinearity, however, introduce the effect of the nonlinear layers on the propagation of the MSE through the layers.

IV. Optimization Algorithm for an MLP

The individual steps described in the preceding sections can be brought together to initialize the weights of an arbitrary size MLP in a very accurate fashion. In this section, we will consider the single hidden layer MLP case for simplicity. However, the described algorithm can easily be generalized to larger MLP topologies.

Initialization. Given training data in the form $(\mathbf{x}_s, \mathbf{d}_s^2)$, $s = 1,...,N$. Initialize the weights $\mathbf{W}^1$, $\mathbf{W}^2$, $\mathbf{b}^1$, $\mathbf{b}^2$ randomly. The superscripts '1' and '2' denote layer. Evaluate network outputs and store $\mathbf{z}_s^1$, $\mathbf{y}_s^1$, $\mathbf{z}_s^2$, $\mathbf{y}_s^2$ corresponding to $\mathbf{x}_s$. Set J_{opt} to the MSE between $\mathbf{y}_s^2$ and $\mathbf{d}_s^2$. Set $\mathbf{W}_{opt}^1 = \mathbf{W}^1$, $\mathbf{b}_{opt}^1 = \mathbf{b}^1$, $\mathbf{W}_{opt}^2 = \mathbf{W}^2$, $\mathbf{b}_{opt}^2 = \mathbf{b}^2$.

Step 1. Compute $\bar{\mathbf{d}}_s^2 = f^{-1}(\mathbf{d}_s^2)$, $\forall s$.

Step 2. Compute $\mathbf{d}_s^1 = \left(\mathbf{W}^{2T}\mathbf{W}^2\right)^{-1}\mathbf{W}^{2T}(\bar{\mathbf{d}}_s^2 - \mathbf{b}^2)$ (if overdetermined) or the minimum norm solution.

Step 3. Compute $\bar{\mathbf{d}}_s^1 = f^{-1}(\mathbf{d}_s^1)$, $\forall s$.

Step 4. Optimize $\mathbf{W}^1$ and $\mathbf{b}^1$ using (6). Since this is the first layer, the input **x** is the actual input of the MLP. The desired output is $\bar{\mathbf{d}}_s^1$ Optionally use $\mathbf{G} = \mathbf{W}^{2T}\mathbf{W}^2$ or $\mathbf{G} = \mathbf{I}$ (experimentally the latter gives better results).

Step 5. Evaluate $\mathbf{z}_s^1$, $\mathbf{y}_s^1$ using the new weights.

Step 6. Optimize $\mathbf{W}^2$ and $\mathbf{b}^2$ using (6). Since this is the second layer, the input **x** is the output of the previous layer, $\mathbf{y}_s^1$. The desired output is $\bar{\mathbf{d}}_s^2$.

Step 7. Evaluate $\mathbf{z}_s^2$, $\mathbf{y}_s^2$ using the new weights.

Step 8. Evaluate the new MSE and if $J<J_{opt}$, set $\mathbf{W}_{opt}^1 = \mathbf{W}^1$, $\mathbf{b}_{opt}^1 = \mathbf{b}^1$, $\mathbf{W}_{opt}^{21} = \mathbf{W}^2$, $\mathbf{b}_{opt}^2 = \mathbf{b}^2$.

Step 9. Go back to *Step 2* or stop.

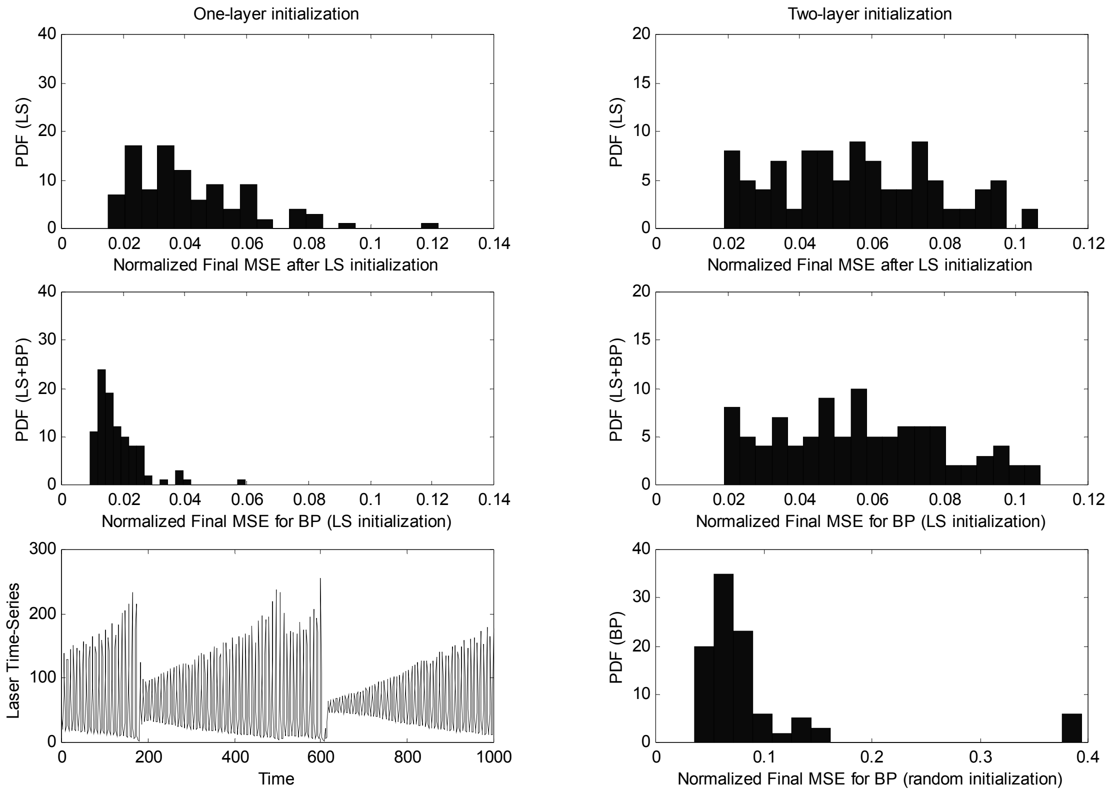
The algorithm above backpropagates the desired signal to the first layer and then optimizes the weights of the layers sweeping them from the first to the last. Alternatively, first the last layer weights may be optimized, then the desired signal can be backpropagated through that layer using the optimized values of the weights, and so on. Thus, in this alternative algorithm, the layers are optimized sweeping them from the last to the first. Simulations with the latter yield results similar to those obtained by the presented algorithm.

The algorithm is iterated a number of times (two to five). The weight values that correspond to the smallest MSE error are assigned as initial conditions to a standard backpropagation or some other optimization algorithm. Although determining the optimal weights requires using this hybrid approach, since the least squares approach yields approximate optimization, for some applications, the least squares initialization solution for the weights might yield satisfactory results. The loss in MSE, in the latter situation, is compensated for by the fast determination of these suboptimal solutions.

V. Case Studies

In this section, we present the results of Monte Carlo initialization and training experiments performed using the procedure described in the preceding sections. In these experiments, we used three data sets: the laser time-series [12], the Dow Jones Closing Index [12], and realistic engine manifold pressure-temperature dynamics data [13]. The first two data sets will be utilized in the single-step prediction framework, whereas, the last one will be considered as a nonlinear system identification problem. In this system identification problem, the input is the throttle angle that controls the amount of air flowing into the manifold. The system states are the internal manifold temperature and pressure.

For these three data sets, we have employed the following networks respectively: TDNN(3,11,1) for the laser data, TDNN(5,7,1) for the Dow Jones data, and MLP(4,5,1)



(a) (b)

Figure 2. Histograms of final MSE values for the laser-series.

for system identification. In this notation, the first value denotes the number of inputs, the second value denotes the number of processing elements (PE) in the hidden layer and the last value denotes the number of outputs of the MLP-type neural network. In the system identification example, the four inputs of the MLP are the current and the previous values of the input and the output (manifold pressure) of the system. In all examples, PEs have sigmoid nonlinearities (*arctan*).

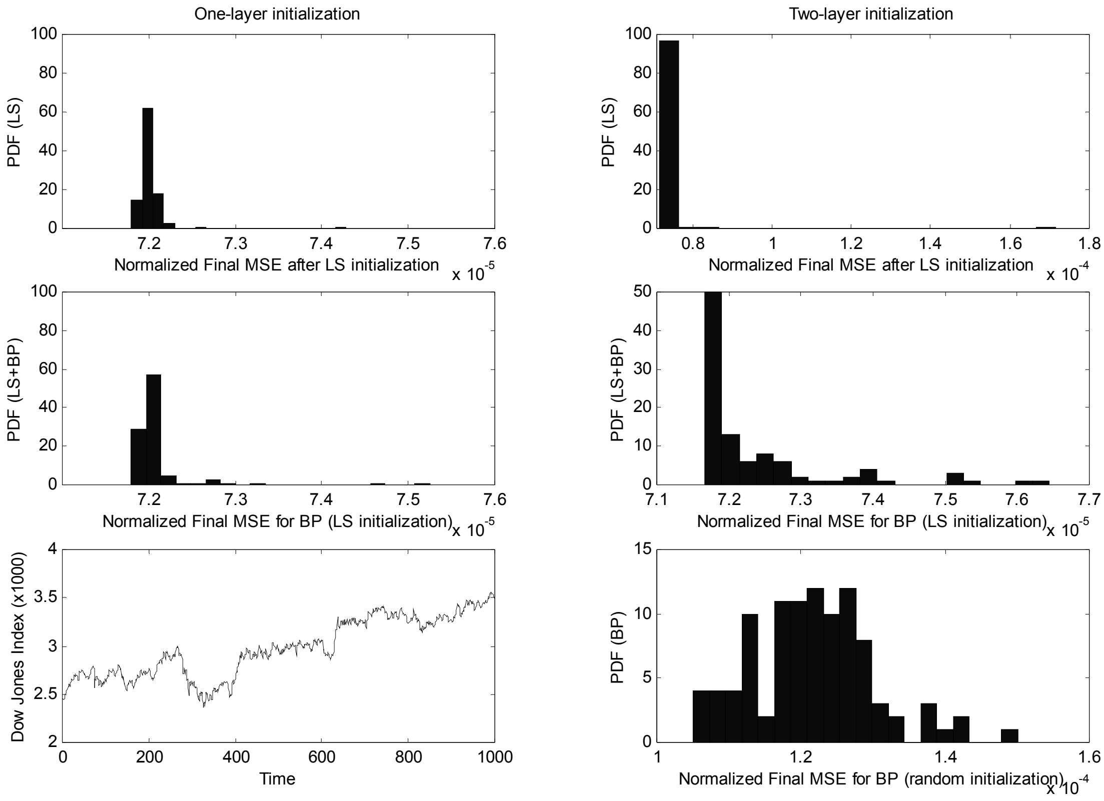
A total of five different approaches are taken in the training of all networks in all three examples. These are listed below and in the rest of the paper they will be addressed by the designated letter codes.

- Backpropagation with random initial weights (BP).
- Initialize second layer only using Steps 5-7 of the least squares algorithm (LS1). Iterate once.
- Initialize both layers using the least squares algorithm in its entirety (LS2). Iterate three times.
- Use LS1 to initialize second layer and run BP starting with random weights for first layer and LS1-initialized weights for second layer (LS1+BP).
- Use LS2 to initialize all the weights and run BP starting with LS2-initialized weights (LS2+BP).

For the three data sets, we have iterated BP for 1000, 2000, and 200 epochs, respectively. In contrast, for LS+BP approaches, the BP step was iterated 250, 500, and 50 epochs only. For all backpropagation updates, MATLAB®'s Neural Network Toolbox was utilized. The numbers of epochs mentioned above that are required for convergence was determined experimentally beforehand.

The results for laser time series prediction are summarized in the histograms given in Fig. 2. In the 100 Monte Carlo experiments, LS1 and LS2 initialization schemes achieved low normalized MSE levels as seen in subfigures a1 and b1 (MSE is normalized by dividing with the power of the desired signal). Further training with backpropagation resulted in an improvement in MSE in the LS1+BP approach, but it did not change MSE much in LS2+BP (see a2 and b2). Training with BP, on the other hand, in general resulted in higher MSE values either due to slow convergence or local minima. Notice that the least squares algorithm has a much smaller computational complexity compared to backpropagation, yet it still achieves very small MSE levels.

The results of the Dow Jones series prediction are summarized in Fig. 3. Similarly, LS1 and LS2 initialization schemes achieved very small MSE levels and further training with backpropagation (LS+BP) did not improve MSE significantly. At the end of the preset number of iterations,



(a) (b)

Figure 3. Histograms of final MSE values for the Dow-Jones-series.

the MSE levels of BP were much larger than those obtained with methods that used LS initialization.

We have seen the advantage of using LS1 and LS2 initialization in MLP training in the first two examples. Performance-wise, we did not observe great differences between these two LS approaches, however. In this last example, we see a possible benefit of using LS2 over LS1. The results of the engine-dynamics-identification example are shown in Fig. 4. Notice that LS1 achieves an MSE around $5\text{x}10^{-2}$ (subfigure a1), while LS2 yields an MSE on the order of 10^{-5} (subfigure b1). In both cases, further training using backpropagation does not improve MSE significantly. The BP approach was trapped in the same local minimum as LS1.

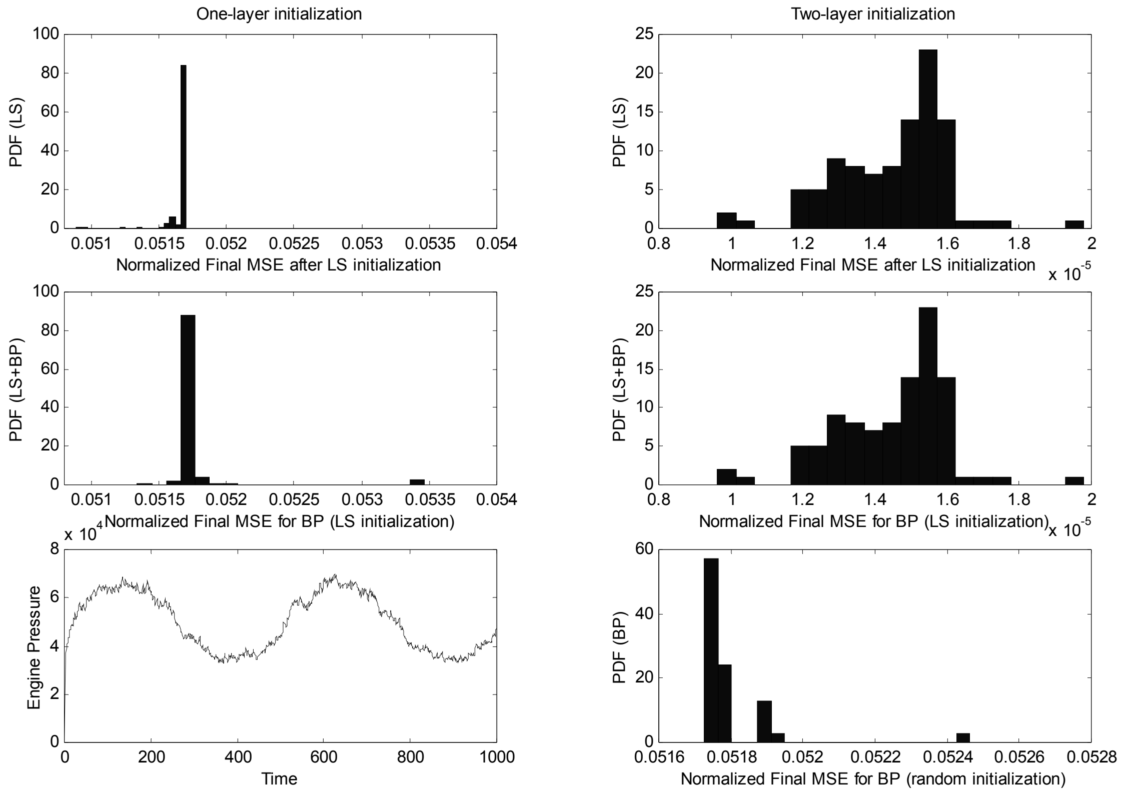
VI. Conclusions

The training speed and accuracy of neural networks can be improved drastically by proper initialization of the weights before a conventional nonlinear optimization tool is employed. In this paper, we have investigated a previously studied initialization scheme, namely least squares, in a mathematically rigorous manner. Previous work using this methodology often ignored the effect of the network nonlinearities on the propagation of the MSE through the layers of the network. Based on the theoretical results that are presented here, we have determined an algorithm to accurately initialize the weights of an MLP to a suboptimal solution, which yields a very small MSE. This algorithm is named as *backpropagation of the desired response*, due to the procedure actually prescribing how to propagate the desired output to the internal layers of the MLP. Then each layer of weights can be (almost) optimally trained by solving a linear system of equations, which correspond to finding the linear least squares solution for this layer of weights.

Although we have focused on the initialization aspect of this least squares algorithm, in many practical problems, such as real-time adaptive control using neural network models and controllers, the solutions offered by the proposed algorithm could be sufficiently accurate. This was demonstrated by a nonlinear system identification problem example, in which an MLP was trained to approximate a realistic engine manifold model accurately.

References

[1] D.E. Rumelhart, G.E. Hinton, R.J. Williams, "Learning Representations of Back-Propagation Errors," Nature, vol. 323, pp.533-536, 1986.



(a) (b)

Figure 4. Histograms of final MSE values for engine-dynamics-identification.

[2] S. Amari, "Natural Gradient Works Efficiently in Learning," Neural Computation, vol.10, pp.251-276, 1998.

[3] C.M. Bishop, "Exact Calculation of the Hessian Matrix for the Multilayer Perceptron," Neural Computation, vol. 4, no. 4, pp.494-501, 1992.

[4] M.A. Styblinski, T.S. Tang, "Experiments in Nonconvex Optimization: Stochastic Approximation with function Smoothing and Simulated Annealing," Neural Networks, vol. 3, pp.467-483, 1990.

[5] S. Bengio, Y. Bengio, J. Cloutier, "Use of Genetic Programming for the Search of a New Learning Rule for Neural Networks," Proceedings of the First IEEE World Congress on Computational Intelligence and Evolutionary Computation, pp.324-327, 1994.

[6] V.W. Porto, D.B. Fogel, "Alternative Neural Network Training Methods [Active Sonar Processing]," IEEE Expert, vol. 10, no. 3, pp. 16-22, 1995.

[7] D. Nguyen, B. Widrow, "Improving the Learning Speed of 2-layer Neural Networks by Choosing Initial Values of the Adaptive Weights," Proceedings of International Joint Conference on Neural Networks, vol. 3, pp.21-26, 1990.

[8] G.P. Drago, S. Ridella, "Statistically Controlled Activation Weight Initialization (SCAWI)," IEEE Transactions on Neural Networks, vol. 3, pp. 899-905, 1992.

[9] Y.F. Yam, T.W.S. Chow, "A New Method in Determining the Initial Weights of Feedforward Neural Networks," Neurocomputing, vol. 16, no. 1, pp.23-32, 1997.

[10] F. Biegler-Konig, F. Barnmann, "A Learning Algorithm for Multilayered Neural Networks Based on Linear Least Squares Problems," Neural Networks, vol. 6, pp. 127-131, 1993.

[11] G. Golub, C.V. Loan, *Matrix Computation*, John Hopkins University Press, Baltimore, MD, 1993.

[12] http://www.kernel-machines.org/

[13] J.D. Powell, N.P. Fekete, C-F. Chang, "Observer-Based Air-Fuel Ratio Control," IEEE Control Systems Magazine, vol. 18, no. 5, pp. 72-83, Oct. 1998.