On Descent Spectral CG algorithms for Training Recurrent Neural Networks

I.E. Livieris, D.G. Sotiropoulos and P. Pintelas

***Abstract*—In this paper, we evaluate the performance of descent conjugate gradient methods and we propose a new algorithm for training recurrent neural networks. The presented algorithm preserves the advantages of classical conjugate gradient methods while simultaneously avoids the usually inefficient restarts. Simulation results are also presented using three different recurrent neural network architectures in a variety of benchmarks.**

***Index Terms*—Recurrent neural networks, descent spectral conjugate gradient methods, sufficient descent property, performance profiles.**

I. Introduction

Recurrent neural networks (RNN) constitute an elegant way to provide better modeling accuracy compared to classical feedforward neural networks. It is well known that recurrent neural networks can be trained in a similar way with the feedforward neural networks [22], [23], such training requires a great deal of computation. More analytically, the training process can be formulated as the minimization of an error function $E(w)$ that depends on the connection weights w of the network. A traditional way to solve this problem is by an iterative gradient-based training algorithm which generates a sequence of weights $\{w\}_{k=0}^{\infty}$ using the recurrence

$$w_{k+1} = w_k + \eta_k d_k, \quad k = 0, 1, \dots \tag{1}$$

where k is the time step for a specific pattern, $w_0 \in \mathbb{R}^n$ is a given starting point, $\eta_k > 0$ is the learning rate and d_k is a descent search direction. In general, conjugate gradient methods constitute an elegant choice for efficiently training large neural networks due to their simplicity and their very low memory requirements since they do not require the evaluation of the Hessian matrix neither the impractical storage of an approximation of it.

Despite the theoretical and practical advantages of conjugate gradient methods, the main drawback of these methods is the use of restarting procedures in order to guarantee convergence [17]. Nevertheless, there is also a worry with restart algorithms that restarts may be triggered too often, abandoning the second-order derivative information; thus degrading the overall efficiency of the method [14]. Recently, Yu and Guan [24] proposed a new class of conjugate gradient methods which guarantee sufficient descent independent of the accuracy of the line search, avoiding thereby the usually inefficient restarting procedures.

I. E. Livieris is with the Department of Mathematics, University of Patras, Greece. E-mail: `livieris@upatras.gr`

D. G. Sotiropoulos is with the Department of Informatics, Ionian University, Greece. E-mail: `dgs@ionio.gr`

P. Pintelas is with the Department of Mathematics, University of Patras, Greece. E-mail: `pintelas@math.upatras.gr`

In this work, we study the performance of this class of methods and we propose a new conjugate gradient algorithm for training recurrent neural networks. The proposed algorithm preserves the advantages of classical conjugate gradient methods and simultaneously avoids the inefficient restarts.

The remainder of this paper is organized as follows. Section II presents a brief summary of conjugate gradient methods. In Section III are presented the new class of descent conjugate gradient methods and our proposed algorithm for training recurrent neural networks. Section IV, reports our experimental results and in Section V are presented our concluding remarks and our proposals for future research.

Notations. Throughout this paper $\|\cdot\|$ denotes the Euclidean norm, $\langle\cdot\rangle$ stands for the inner product and the gradient of the error function is indicated by $\nabla E(w^k) = g_k$.

II. Conjugate Gradient Methods

In conjugate gradient methods the basic idea for determining the search direction in Eq. (1) is the linear combination of the negative gradient vector at the current iteration with the previous search direction, namely

$$d_k = \begin{cases} -g_k, & \text{if } k = 0; \\ -g_k + \beta_k d_{k-1}, & \text{otherwise.} \end{cases} \tag{2}$$

In the literature, there have been proposed several choices for defining the scalar parameter β_k which give rise to distinct conjugate gradient methods. The most famous ones were proposed by Fletcher-Reeves [5] and Polak-Ribière [16]. However, probably the most efficient choice was introduced by Perry [15], that is

$$\beta_k^P = \frac{g_k^T(y_{k-1} - s_{k-1})}{y_{k-1}^T d_{k-1}} \tag{3}$$

where $s_{k-1} = w_k - w_{k-1}$ and $y_{k-1} = g_k - g_{k-1}$. The convergence analysis of these methods is usually based on mild conditions which refer to the Lipschitz and boundedness assumptions and is closely connected with the sufficient descent property

$$g_k^T d_k \le -c\|g_k\|^2 \tag{4}$$

where c is a fixed constant. Hager and Zhang [9] have presented an excellent survey on conjugate gradient methods, with special attention to their convergence properties.

Birgin and Martínez [2] incorporated the spectral gradient [20] in the conjugate gradient framework as follows

$$d_k = \begin{cases} -g_k, & \text{if } k = 0; \\ -\frac{1}{\delta_k} g_k + \beta_k d_{k-1}, & \text{otherwise,} \end{cases} \tag{5}$$

where δ_k is a scalar parameter usually defined as

$$\delta_k = \frac{\langle y_{k-1}, s_{k-1}\rangle}{\langle s_{k-1}, s_{k-1}\rangle} \tag{6}$$

which lies between the largest and the smallest eigenvalue of the average Hessian. The motivation for this selection constitutes in providing a two point approximation to the secant equation underlying Quasi-Newton methods [1]. Using a geometric interpretation for quadratic function minimization Birgin and Martínez suggested the following expression for defining the update parameter β_k in Eq. (5)

$$\beta^{SP} = \frac{g_k^T(y_{k-1} - \delta_k s_{k-1})}{\delta_k y_{k-1}^T d_{k-1}}. \tag{7}$$

Clearly, in case $\delta_k = 1$ this formula is reduced to the classical formula (3), introduced by Perry [15].

Unfortunately even with exact line search conjugate gradient methods, cannot guarantee to generate descent directions. Therefore, the use of a restarts are employed in order to guarantee convergence. A more sophisticated restarting criterion has been introduced by Birgin and Martínez [2] which consists of testing if the angle between the current direction and the gradient is not acute enough, namely

$$d_k^T g_k \le 10^{-3}\|d_k\|_2\|g_k\|_2.$$

In the literature there have been proposed several restarting criteria [3], [17], [21] with various performances.

III. Descent Conjugate Gradient Methods

In more recent works, Yu and Guan [24] proposed a new class of conjugate gradient methods which can guarantee the sufficient descent property Eq. (4) independent of the accuracy of the line search where the scalar parameter β_k is defined as

$$\beta_k^{DP} = \beta_k^P - \frac{C\|y_{k-1} - s_{k-1}\|^2}{(y_{k-1}^T d_{k-1})^2} g_k^T d_{k-1} \tag{8}$$

Parameter C essentially controls the relative weight between conjugacy and descent and in case $C \ge 1/4$ then all generated directions are descent directions.

Along this line, Yu et al. [25] motivated by the previous works [2], [8], [24], embedded the spectral gradient [1] in the descent conjugate methods framework

$$\beta_k^{DSP} = \beta_k^{SP} - \frac{C\|y_{k-1} - s_{k-1}\|^2}{\delta_k(y_{k-1}^T d_{k-1})^2} g_k^T d_{k-1} \tag{9}$$

Obviously, if $\delta_k = 1$ then the above formula will reduce to the corresponding descent conjugate gradient (8).

In order to ensure global convergence of the conjugate gradient methods for general nonconvex functions, it has been proposed to truncate β_k^{DSP} by restricting the lower bound of the update parameter of being nonnegative [6], [18]. However by restricting β_k^{DSP} to be nonnegative, results that the convergence speed may be reduced. Thus, Yu et al. [25] proposed two modified schemes in which the lower bound on β_k is dynamically adjusted, in order to make the lower bound small as the iterates converge. In particular, they proposed two variants for selecting the truncated update parameter β^{DSP^+}. The first variant of the truncated scheme uses the update parameter

$$\beta_k^{DSP_1^+} = \max\left\{\beta_k^{DSP}, -\frac{C\|y_{k-1} - \delta_k s_{k-1}\|^2}{\delta(y_{k-1}^T d_{k-1})^2}|g_k^T s_{k-1}|\right\} \tag{10}$$

while the second one is based on a scheme that was first introduced by Hager and Zhang [8], namely

$$\beta_k^{DSP_2^+} = \max\left\{\beta_k^{DSP}, -\frac{1}{\delta_k\|d_{k-1}\|\min\{\xi, \|g_{k-1}\|\}}\right\} \tag{11}$$

where ξ is some positive constant. Notice that when $\|g_{k-1}\|$ tends to zero as k grows, $\beta_k^{DSP_2^+}$ is reduced to the descent spectral Perry (9), when $\|d_{k-1}\|$ is bounded [8].

In the following, we present a high level description of our proposed algorithm for training recurrent neural networks, based on descent spectral conjugate gradient methods.

```
Step 1: Initiate $w_0$, $0 < \sigma_1 < \sigma_2 < 1$, $Err$ and $\epsilon \to 0$;
    set $k = 0$.
Step 2: If $(E(w_k) < Err)$ or $(\|\nabla E(w_k)\|_2 < \epsilon)$ terminate.
Step 3: Compute the descent direction
    $d_k = \begin{cases} -\nabla E(w_k), & \text{if } k = 0; \\ -\frac{1}{\delta_k}\nabla E(w_k) + \beta_k d_{k-1}, & \text{otherwise} \end{cases}$
    where the update parameter $\beta_k$ is defined by one of (9), (10) or (11).
Step 4: Find a step length $\eta_k$ using the following line search. For $0 < \sigma_1 < \sigma_2 < 1$ at each iteration, choose
    the step length satisfying the strong Wolfe line search conditions
    $E(w_k + \eta_k d_k) - E(w_k) \le \sigma_1 \eta_k \langle\nabla E(w_k) d_k\rangle$
    $|\langle\nabla E(w_k + \eta_k d_k), d_k\rangle| \le -\sigma_2\langle\nabla E(w_k), d_k\rangle$
Step 5: Update the weights
    $w_{k+1} = w_k + \eta_k d_k$
Step 6: Set $k = k + 1$ and goto Step 2.
```

IV. Experimental Results

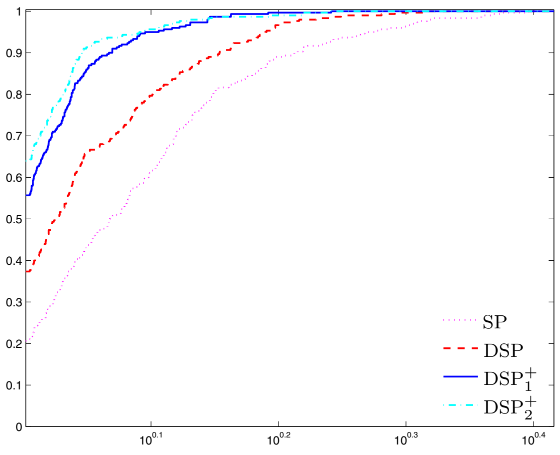
In this section we will present experimental results in order to evaluate the performance of the proposed conjugate gradient algorithm in three famous benchmarks acquired by the UCI Repository [12]. Subsequently, we briefly describe each problem and present the performance comparison between: the descent spectral Perry's conjugate gradient (DSP) with the spectral Perry's conjugate gradient (SP) of Birgin and Martínez [2]. Moreover we test the numerical performance of two different truncation strategies for the update parameter β_k. So we evaluate the two versions of the truncate spectral Perry conjugate gradient ($\mathtt{DSP}^+$): the first one ($\mathtt{DSP}_1^+$) uses the update parameter $\beta_k^{DSP_1^+}$ defined in Eq. (10) and the other one ($\mathtt{DSP}_2^+$) uses the update parameter $\beta_k^{DSP_2^+}$ defined in Eq. (11). We have chosen three classical first-order recurrent neural architectures:

- Feedforward Time Delay (FFTD),
- Nonlinear Autoregressive Network with Exogenous Inputs (NARX)
- Elman's Layered Recurrent Networks (ELRN)

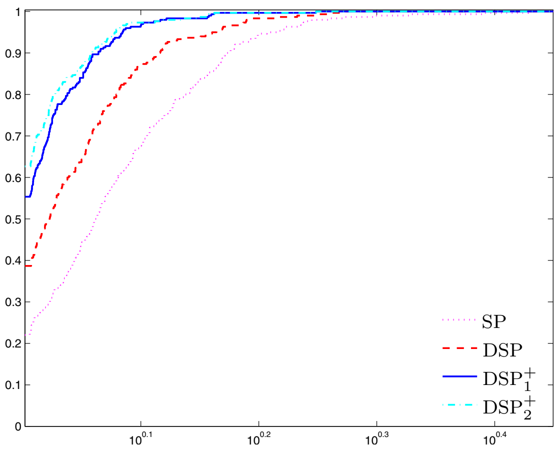
For all algorithms the heuristic parameters were set as $\sigma_1 = 10^{-4}$, $\sigma_2 = 0.5$ and $\xi = 0.01$ for all experiments [24]. All networks have received the same sequence of input patterns and the initial weights were initiated using the Nguyen-Widrow method [13]. For evaluating classification accuracy we have used the standard procedure called *k-fold cross-validation* [10]. All algorithms were evaluated using the performance profiles proposed by Dolan and Morè [4] for measuring their efficiency and robustness in terms of computational cost (i.e. function/gradient evaluations). The performance profile plots the fraction of simulations for which any given algorithm is within a factor of the best trainer. The horizontial axis of each plot shows the percentage of the simulations for which a method is the fastest (efficiency), while the vertical side gives the percentage of the simulations that the RNNs were successfully trained by each method (robustness). The reported performance profiles have been created using the `Libopt` environment [7], an excellent set of tools written in Perl. The implementation has been carried out using Matlab version 7, based on the SCG code of Birgin and Martínez [2].

A. SPECT Heart Problem

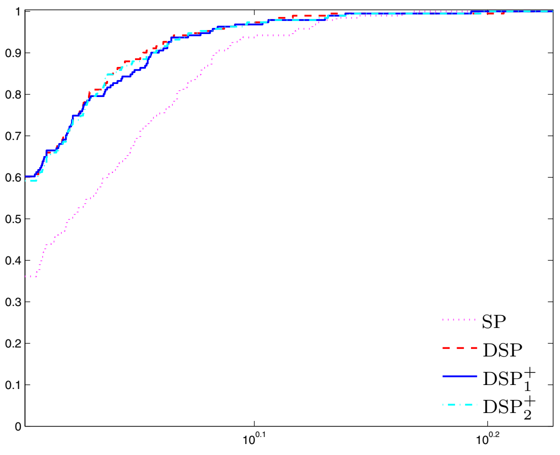
This data set contains data instances derived from cardiac Single Proton Emission Computed Tomography (SPECT) images from the University of Colorado [12]. The network architectures for this medical classification problem constitute of 1 hidden layer with 6 neurons and an output layer of 1 neuron. The termination criterion is set to $E \leq 0.1$ within the limit of 1000 epochs.



(a) Performance for FFTD



(b) Performance for ELRN



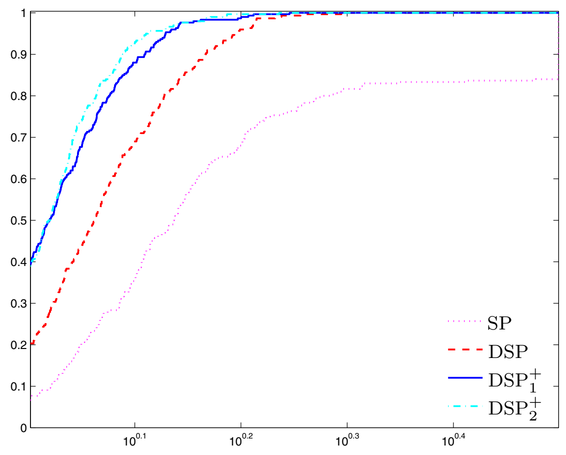
(c) Performance for NARX

Fig. 1. Log_{10} scaled performance profiles for the spect heart problem.

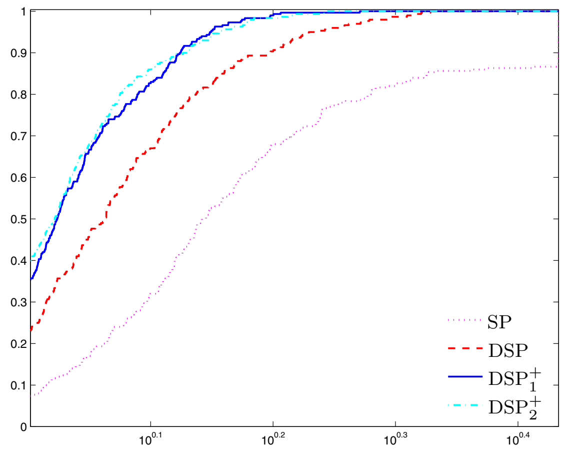
Figure 1 presents the performance profiles for each recurrent neural network architecture for the spect heart problem. `DSP` is much more robust and efficient the `SP` since the curve of the former lies above the curve of the latter. Furthermore, $\mathtt{DSP}_1^+$ and $\mathtt{DSP}_2^+$ exhibit the highest probability of being the optimal solver with the latter exhibiting slightly better performance.

B. Australian credit approval problem

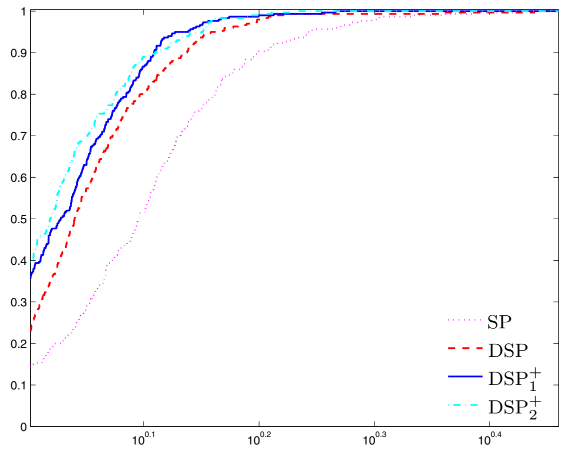
Australian Credit Approval dataset contains all the details about credit card applications. This dataset is interesting because the data varies and has mixture of attributes which is continuous, nominal with small numbers of values, and nominal with larger numbers of values. We have used neural networks with 2 hidden layers consist of 16 and 8 neurons and an output layer of 2 neurons [19]. The termination criterion is set to $E \leq 0.1$ within the limit of 1000 epochs and all networks were tested using 10-fold cross validation.



(a) Performance for FFTD



(b) Performance for ELRN



(c) Performance for NARX

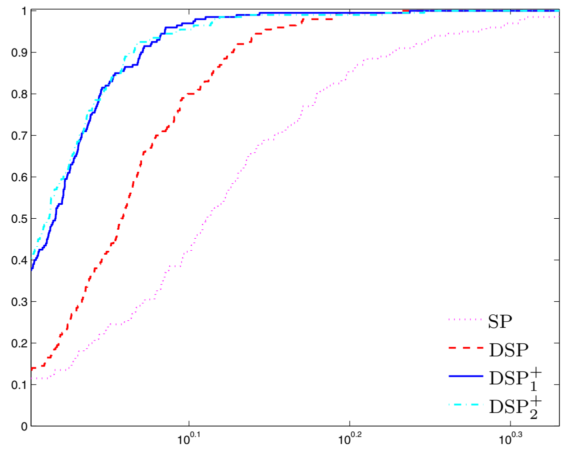
Fig. 2. Log_{10} scaled performance profiles for the Australian credit approval problem.

In Figure 2 are presented the performance profiles for the card problem investigating the efficiency and robustness of each training method. $\texttt{DSP}_1^+$ and $\texttt{DSP}_2^+$ exhibit the best performance regarding all recurrent neural network architectures. Moreover, the interpretation in Figure 2 implies that the sufficient descent property led to a significant improvement of DSP, $\texttt{DSP}_1^+$ and $\texttt{DSP}_2^+$ over the SP method since the curves of the former lie above the curve of the latter.

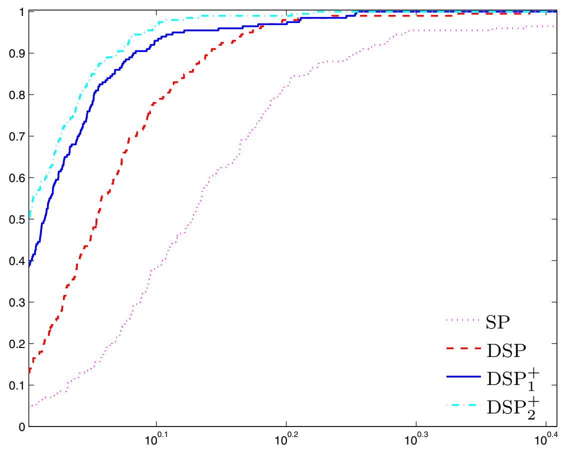
C. Escherichia coli Problem

This problem is based on a drastically imbalanced data set of 336 patterns and concerns the classification of the *E. coli* protein localization patterns into eight localization sites. E. coli, being a prokaryotic gram-negative bacterium, is an important component of the biosphere. Three major and distinctive types of proteins are characterized in E. coli: enzymes, transporters and egulators. The largest number of genes encodes enzymes (34%) (this should include all the cytoplasm proteins) followed by the genes for transport functions and the genes for regulatory process (11.5%) [11]. The network architectures constitute of 1 hidden layer with 16 neurons and an output layer of 8 neurons. The stopping criterion is set to $E \leq 0.02$ within the limit of 2000 epochs. In all simulations the neural networks were tested using 4-fold cross-validation.

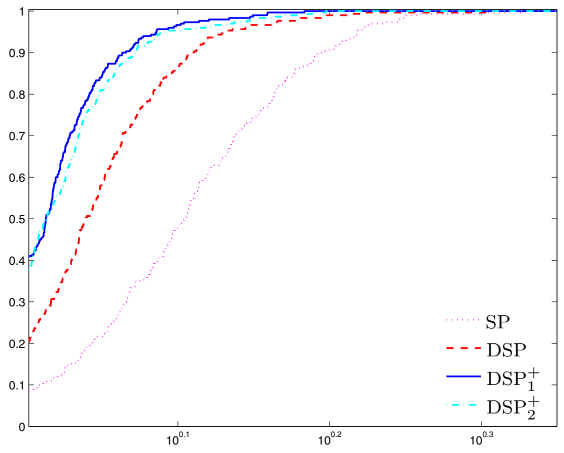
Figure 3 illustrates the performance profiles for the ecoli problem investigating the performance of each training method. DSP is much more efficient and robust than SP regarding all recurrent neural network architectures. Furthermore, $\texttt{DSP}_1^+$ and $\texttt{DSP}_2^+$ perform similarly and exhibit the best performance since they present the highest probability of being the optimal solver.



(a) Performance for FFTD



(b) Performance for ELRN



(c) Performance for NARX

Fig. 3. Log_{10} scaled performance profiles for the escherichia coli problem.

V. Conclusions

In this work, we evaluate the performance of the proposed conjugate gradient algorithm for training recurrent neural networks. From the rigorous analysis of the simulation results is strongly demonstrated the sufficient descent property led to a significant improvement of the DSP method over SP method which provides faster, more stable and reliable convergence.

Since RNNs have the ability to deal with time varying input and output through their own natural temporal operation, our future work will be concentrated on evaluating our proposed algorithm on time varying large datasets, such as time series prediction.

References

[1] J. Barzilai and J.M. Borwein. Two point step size gradient methods. *IMA Journal of Numerical Analysis*, 8:141–148, 1988.

[2] E.G. Birgin and J.M. Martínez. A spectral conjugate gradient method for unconstrained optimization. *Applied Mathematics and Optimization*, 43:117–128, 1999.

[3] A. Buckley and A. LeNir. QN-like variable storage conjugate gradients. *Mathematical Programming*, 27:155–175, 1983.

[4] E. Dolan and J.J. Moré. Benchmarking optimization software with performance profiles. *Mathematical Programming*, 91:201–213, 2002.

[5] R. Fletcher and C.M. Reeves. Function minimization by conjugate gradients. *Computer Journal*, 7:149–154, 1964.

[6] J.C. Gilbert and J. Nocedal. Global convergence properties of conjugate gradient methods for optimization. *SIAM Journal of Optimization*, 2(1):21–42, 1992.

[7] J.Ch. Gilbert and X. Jonsson. LIBOPT - an environment for testing solvers on heterogeneous collections of problems. *Submitted to ACM Transactions on Mathematical Software*, January 2009.

[8] W.W. Hager and H. Zhang. A new conjugate gradient method with guaranteed descent and an efficient line search. *SIAM of Journal Optimization*, 16:170–192, 2005.

[9] W.W. Hager and H. Zhang. A survey of nonlinear conjugate gradient methods. *Pacific of Journal Optimization*, 2:35–58, 2006.

[10] R. Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *International Joint Conference on Artificial Intelligence*, pages 223–228. AAAI Press and MIT Press, 1995.

[11] P. Liang, B. Labedan, and M. Riley. Physiological genomics of escherichia coli protein families. *Physiological Genomics*, 9:15–26, 2002.

[12] P.M. Murphy and D.W. Aha. UCI repository of machine learning databases. Irvine, CA: University of California, Department of Information and Computer Science, 1994.

[13] D. Nguyen and B. Widrow. Improving the learning speed of 2-layer neural network by choosing initial values of adaptive weights. *Biological Cybernetics*, 59:71–113, 1990.

[14] J. Nocedal. Theory of algorithms for unconstrained optimization. *Acta Numerica*, 1:199–242, 1992.

[15] A. Perry. A modified conjugate gradient algorithm. *Operational Research*, 26:1073–1078, 1978.

[16] E. Polak and G. Ribière. Note sur la convergence de methods de directions conjuguees. *Revue Francais d'Informatique et de Recherche Operationnelle*, 16:35–43, 1969.

[17] M.J.D. Powell. Restart procedures for the conjugate gradient method. *Mathematical Programming*, 12:241–254, 1977.

[18] M.J.D. Powell. Convergence properties of algorithms for nonlinear optimization. *SIAM Review*, 28:487–500, 1986.

[19] L. Prechelt. PROBEN1-A set of benchmarks and benchmarking rules for neural network training algorithms. Technical Report 21/94, Fakultt fr Informatik, University of Karlsruhe, 1994.

[20] M. Raydan. The Barzilai and Borwein gradient method for the large scale unconstrained minimization problem. *SIAM Journal of Optimization*, 7:26–33, 2007.

[21] D.F. Shanno and K.H. Phua. Minimization of unconstrained multivariate functions. *ACM Transactions on Mathematical Software*, 2:87–94, 1976.

[22] P.J. Werbos. Backpropagation through time: What it does and how to do it. In *Proceedings of the IEEE*, volume 78, pages 1550–1560, 1990.

[23] R.J. Williams and D. Zipser. A learning algorithm for continually running fully recurrent neural networks. *Neural Computation*, 1:270–280, 1989.

[24] G. Yu and L. Guan. New descent nonlinear conjugate gradient methods for large scale unconstrained optimization. Technical report, Department of Scientific Computation and Computer Application, Sun Yat-Sen University, P.R. China, 2005.

[25] G. Yu, L. Guan, and W. Chen. Spectral conjugate gradient methods with sufficient descent property for large-scale unconstrained optimization. *Optimization Methods and Software*, 23(2):275–293, 2008.