Convergence Analysis of a Discrete-Time Recurrent Neural Network to Perform Quadratic Real Optimization with Bound Constraints

Maria José Pérez-Ilzarbe

Abstract—This paper presents a model of a discrete-time recurrent neural network designed to perform quadratic real optimization with bound constraints. The network iteratively improves the estimate of the solution, always maintaining it inside of the feasible region. Several neuron updating rules which assure global convergence of the net to the desired minimum have been obtained. Some of them also assure exponential convergence and maximize a lower bound for the convergence degree. Simulation results are presented to show the net performance.

Index Terms—Convergence of numerical methods, discrete-time systems, gradient methods, optimization methods, recurrent neural networks.

I. INTRODUCTION

The problem of quadratic real optimization with bound constraints appears frequently in many areas of science and engineering. It can be stated as the minimization over \( \mathbf{y}^T = (y_1, \ldots, y_n) \in \mathbb{R}^n \) of the quadratic cost function

\[
E = -\frac{1}{2} \mathbf{y}^T W \mathbf{y} - \mathbf{b}^T \mathbf{y}
\]

subject to the constraints

\[
down_i \leq y_i \leq up_i, \quad 1 \leq i \leq n
\]

where \( W \) is a negative definite \( n \times n \) matrix and \( \mathbf{b}, \mathbf{down}, \mathbf{up} \in \mathbb{R}^n \) are constant. This simple formulation includes a broad class of optimization problems. It must be remarked that, since \( E \) is strictly convex and the constraint region is a convex set, the constrained minimum is unique [1].

Quadratic real optimization with linear constraints is a difficult and time-intensive task. Hopfield-type networks have been used for solving it [3]–[7]. They essentially belong to the penalty function methods for constrained optimization [1], [2].

The final solutions attained with these methods satisfy the constraints only approximately, therefore they can be outside of the feasible region. Hopfield-type networks can also implement the augmented Lagrange multipliers method [8] which combines both Lagrange and penalty methods. The use of Lagrange multipliers makes the final solutions satisfy exactly the constraints without needing to send the penalty terms to infinity, whereas the use of penalty functions avoids some convergence problems of the standard Lagrange multipliers method. Finally, some other neural-network approaches have been proposed to perform quadratic real optimization with bound constraints in such a manner that all the points generated in the searching trajectory are inside the feasible region [9], [10]. The methods having this characteristic are known as primal methods [2]. Its use allows stopping the process before reaching the exact solution, and take the point attained, which satisfies the constraints, as an approximate solution to the problem.

The recurrent net proposed in the present work belongs to this last class of methods. It is formulated as a discrete-time system so, for computer simulations, it presents advantages over digital simulations of the continuous-time models given in [9] and [10]. In addition, it can be easily implemented in digital hardware. The main contribution of this work is the development of different neuron updating rules that guarantee global convergence of the discrete-time neural network to the desired solution, provided that matrix \( W \) fulfills some sufficient conditions. Some of these rules also assure exponential convergence, and certain network parameters can be properly chosen to maximize a lower bound for the convergence degree. All the schedules presented here are readily amenable to parallel implementation, unlike conventional methods for bound-constrained quadratic real optimization. The proposed optimization neural network has been used, with promising results, for solving two different problems: the problem of linear systems identification [11], which constitutes an unconstrained quadratic optimization problem, and the constrained quadratic optimization problem involved in the calculation of the control signal in predictive control [12]. In the last, the necessity of obtaining a feasible solution in the period of time between two sampling instants, as well as the strict requirement of not violating the constraints, make primal methods substantially appropriate for this application.

The organization of this paper is as follows. The proposed recurrent discrete neural-network is presented in Section II, and convergence of the net is studied in Section III (mathematical proofs are in the Appendix). Section IV shows some simulation results that illustrate the network behavior. In Section V, some practical aspects about the use of the network are discussed, and it is compared with the gradient projection method. Finally, Section VI concludes the main contributions of this paper.

Manuscript received January 15, 1997; revised January 19, 1998. This work was supported in part by the Comisión Interministerial de Ciencia y Tecnología (CICYT) under Grant TAP96-0834, and by the Gobierno de Navarra under Grant OF 55796.

The author is with Departamento de Automática y Computación, Universidad Pública de Navarra, Campus de Arrosadia s/n, Pamplona, 31006 Navarra, Spain.

Publisher Item Identifier S 1045-9227(98)08345-3.
II. The Neural-Network Model

The proposed neural-network consists of a single layer of \( n \) neurons interconnected with symmetric weights that correspond to the elements of the symmetric matrix \( W \) in (1). Each neuron has an external input \( b_i \) which is the corresponding element of the vector \( b \) in (1). The outputs of the \( n \) neurons are the estimated values of the optimization variables \( y_1, \ldots, y_n \). The dynamics of this network can be written as follows:

\[
\sigma(k+1) = g(y(k) + \Delta(k)) = g(y(k) + C(Wg(y(k) + b)) = g(y(k) + C(Wg(y(k) + b)) = g(y(k) + f(\sigma(k+1)))
\]

where \( C \) is a constant diagonal matrix: \( c_{ij} = c_i \delta_{ij} \) and \( f \) is a saturation function which performs the constraints given by (2), i.e.,

\[
\tilde{f}(\sigma) = (f_1(\sigma_1), f_2(\sigma_2), \ldots, f_n(\sigma_n))
\]

with

\[
\begin{align*}
f_i(\sigma_i) &= \sigma_i & \text{if } \sigma_i \leq \sigma_i \leq \sigma_i \\
f_i(\sigma_i) &= \text{down}_i & \text{if } \sigma_i < \text{down}_i \\
f_i(\sigma_i) &= \text{up}_i & \text{if } \sigma_i > \text{up}_i
\end{align*}
\]

In this model synchronous (parallel) neuron updating is assumed. It can be observed that \( \tilde{u} = \tilde{u} = -\frac{\partial E}{\partial \sigma} \), so the network can be included in the family of \textit{gradient methods}. It is also interesting to remark that the network output vector \( y = f(\sigma) \) always remains inside the feasible region defined by the constraints, so the network belongs to the general class of \textit{primal methods} for constrained optimization.

The proposed network can be viewed as a discrete-time continuous state Hopfield network, in a generalized formulation which includes the saturated-linear as neuron transfer function. It has been proved in [13] that “almost every” network belonging to this class converges, when iterating in parallel, to an equilibrium point or to a cycle of length two. On the other hand, the model given by (3) and (4) formally resembles the brain-state-in-a-box model (BSB) [14], [15], although the standard formulation of this last corresponds to \( f_i(\sigma_i) = \sigma_i \) for all \( i \), and \( c_i = \alpha \) for all \( i \). More deep differences, however, exist between the present work and previously published works dealing with the BSB. These differences come from the fact that the BSB is used as associative memory, so one is interested in designing a network with multiple equilibrium points each of which corresponds to a vector to be stored. Thus, stability studies of this model focus on ensuring local stability of each equilibrium point so that a network trajectory starting at an initial state within the basin of attraction of this equilibrium point converges to it [16]–[19]. On the contrary, since we are interested in solving a minimization problem having a unique minimum, our goal is to design a network with a unique equilibrium point which corresponds to such minimum. This equilibrium point must be a globally stable attractor state, such that the network reaches this state independently of the initial conditions, avoiding the convergence to two-length cycles.

It can be easily proved that the network given by (3) and (4) converges to the constrained minimum of (1) provided that the matrix \( C \) is appropriately chosen.

Let us first obtain the conditions for the network to have only one equilibrium point corresponding to the minimum of \( E \) subject to the constraints given in (2). For a point \( \hat{y}^* \) to correspond to this constrained minimum it is a necessary and sufficient condition that it satisfies the Kuhn–Tucker optimality conditions [1] given by

\[
\begin{align*}
\frac{\partial E}{\partial y_i} &= 0 & \text{if } y_i^* \leq \text{down}_i \\
\frac{\partial E}{\partial y_i} &\geq 0 & \text{if } y_i^* \in \text{(down}_i, \text{up}_i) \\
\frac{\partial E}{\partial y_i} &\leq 0 & \text{if } y_i^* = \text{up}_i \\
\end{align*}
\]

for all \( i \).

For the particular case of the cost function \( E \) given in (1) and the constraints given in (2) these conditions can be written as follows:

\[
-(u_i^* + b_i) = - \left[ \sum_{j=1}^{n} w_{ij} y_j^* + b_i \right] \\
\begin{align*}
\geq 0 & \text{ if } y_i^* = \text{down}_i \\
= 0 & \text{ if } y_i^* \in \text{(down}_i, \text{up}_i) \\
\leq 0 & \text{ if } y_i^* = \text{up}_i
\end{align*}
\]

for all \( i \). Let us now look at (3) and (4) of the neural network. For a point \( \tilde{y} = (y_1, \ldots, y_n) \) to be an equilibrium point of the network, the following must hold for all \( i \):

\[
o_i = y_i + \Delta_i = y_i + c_i(u_i + b_i) \quad \text{with} \quad y_i = f_i(\sigma_i),
\]

For an element of \( \tilde{y} \) inside the constraint range \( y_i = \text{down}_i \) becomes \( c_i(u_i + b_i) = 0 \). For an element of \( \tilde{y} \) at the lower bound of the constraint range \( y_i = \text{up}_i \) \( c_i(u_i + b_i) \leq 0 \). For these expressions to coincide with the corresponding Kuhn–Tucker conditions it is necessary and sufficient that \( c_i > 0 \) for all \( i \).

The arguments of the above paragraphs prove that the necessary and sufficient condition for the described network to have the desired minimum as its unique equilibrium point is \( c_i > 0 \) \( \forall i \). In the following section, some sufficient conditions for the network convergence to this equilibrium point will be obtained.

III. Convergence Analysis

First, two different sufficient conditions for exponential convergence of the net will be developed.

Let \( \tilde{y}(k) \) be the point attained at instant \( k \) by a trajectory starting at point \( \tilde{y}(0) \). Taking into account that a trajectory starting at the equilibrium point \( \tilde{y}^* \) always remains at this point, the following expression is obtained:

\[
||\tilde{y}(k) - \tilde{y}^*|| = ||\tilde{f}(\sigma(k)) - \tilde{f}(\sigma^*)|| \\
\leq ||\tilde{f}(\sigma(k) - \sigma^*)|| \\
= ||(I + CW)(\tilde{y}(k-1) - \tilde{y}^*)|| \\
\leq ||I + CW|| \cdot ||(\tilde{y}(k-1) - \tilde{y}^*)||
\]

\[
||\tilde{y}(k) - \tilde{y}^*|| \\
\leq ||(I + CW)(\tilde{y}(k-1) - \tilde{y}^*)|| \\
\leq ||I + CW|| \cdot ||(\tilde{y}(k-1) - \tilde{y}^*)||
\]
where $I$ represents the identity matrix, and it has been used the fact that the saturation function $f$ is a contraction: $||f(v_1) - f(v_2)|| \leq ||v_1 - v_2||$ for all $v_1, v_2 \in \mathbb{R}^n$.

By applying (8) iteratively we obtain $||y'(k) - y'^*|| \leq ||I + CW||^k||y'(0) - y'^*||$. Thus, a sufficient condition for exponential convergence of the net is

$$||I + CW|| = r < 1,$$  \hspace{1cm} (9)

If condition (9) holds, the value $r$ is an upper bound for the convergence ratio of the net. Equating $r = \exp(-\eta)$ we can write $||y'(k) - y'^*|| \leq ||y'(0) - y'^*|| \exp(-\eta k)$, and a lower bound for the net convergence degree is obtained from

$$\eta = -\ln r.$$  \hspace{1cm} (10)

It can be proved that (9) and, consequently, exponential convergence of the net can be assured by appropriately choosing constants $c_i$, provided that matrix $W$ fulfills some conditions. The corresponding results are summarized in the following, and the calculations can be found in the Appendix.

1) **Condition 1 for Exponential Convergence and Corresponding Neuron Updating Rule:** If matrix $W$ is negative definite and fulfills the condition

$$\sum_{j \neq i} |w_{ij}| > 0 \quad \forall i,$$  \hspace{1cm} (11)

which is named to be strongly row diagonal dominant, then the neural network is exponentially convergent if each $c_i$ is chosen according to

$$0 < c_i < \frac{2}{\sum_j |w_{ij}|}.$$  \hspace{1cm} (12)

By using the maximum absolute row sum as matrix norm to calculate $||I + CW||$ (see the Appendix), the minimum upper bound of the convergence ratio, i.e., the maximum lower bound of the convergence degree is reached for

$$c_i = \frac{1}{|w_{ii}|}.$$  \hspace{1cm} (13)

If the rule given in (12) is used, a lower bound for the network convergence degree is

$$\eta = -\ln r_1 \quad \text{with} \quad r_1 = \max_i \left( \frac{\sum_{j \neq i} |w_{ij}|}{|w_{ii}|} \right).$$  \hspace{1cm} (14)

2) **Condition 2 for Exponential Convergence and Corresponding Neuron Updating Rule:** If matrix $W$ is negative definite and fulfills the condition

$$\sum_{i, j} \left( \frac{\sum_{j \neq i} |w_{ij}|^2}{\sum_j |w_{ij}|^2} \right) < 1$$  \hspace{1cm} (15)

then the neural network is exponentially convergent if the following value of $c_i$ is used:

$$c_i = \frac{|w_{ii}|}{\sum_j |w_{ij}|^2}.$$  \hspace{1cm} (16)

This rule has been obtained by using, as matrix norm to calculate $||I + CW||$, the square root of the sum of squared elements (see the Appendix). If the rule given in (15) is used a lower bound for the network convergence degree is

$$\eta = -\ln r_2 \quad \text{with} \quad r_2 = \sqrt{\sum_i \sum_j |w_{ij}|^2}.$$  \hspace{1cm} (17)

The two exponential convergence conditions stated above allow to test easily if it is possible to assure a priori that exponential convergence will be reached (these are sufficient, not necessary conditions). In addition, formulas (12) and (15) give the constant factors $c_i$ that maximize a lower bound for the convergence degree, which can be calculated with (13) and (16).

For problems having a matrix $W$ which does not fulfill neither condition (10) nor condition (14), exponential convergence cannot be guaranteed a priori. For such cases, a schedule which assures global convergence with any symmetric matrix $W$ has been developed. In order to do it, an updating rule has been obtained which assures that $E$ is a Lyapunov function for the neural system given in (3) and (4). The proof can be found in the Appendix. The result can be stated as follows.

3) **Updating Rule for Global Convergence:** If matrix $W$ is symmetric, global convergence of the network can be assured by taking

$$0 < c_i \leq \frac{2}{\eta |w|_{\text{max}}}, \quad \forall i$$  \hspace{1cm} (18)

where $|w|_{\text{max}}$ is the maximum of the matrix elements absolute values.

**IV. SIMULATION RESULTS**

To illustrate the performance of the proposed neural network, the following problems will be studied.

**Example 1:** Obtain the optimum solution $\bar{y}'$ of

$$\min \{E(y') : -20 \leq y_k \leq 20 \quad \forall i\}$$  \hspace{1cm} (19)

where matrix $W$ and vector $\bar{b}$ in expression (1) are

$$W = \begin{bmatrix} -0.18 & -0.648 & -0.288 \\ -0.648 & -2.88 & -0.72 \\ -0.288 & -0.72 & -0.72 \end{bmatrix}$$

and

$$\bar{b} = \begin{bmatrix} -0.4 \\ -0.2 \\ -0.3 \end{bmatrix}.$$  \hspace{1cm} (20)

The point corresponding to the unconstrained minimum is $\bar{y}' = [-61.81; 10.35; 13.96]$. It lies outside the feasible region so, does not correspond to the desired constrained minimum. The optimum solution for the constrained problem is

$$\bar{y}' = \begin{bmatrix} -20 \\ 3.38 \\ 4.204 \end{bmatrix}.$$  \hspace{1cm} (21)
To improve numerical stability and convergence speed a **preconditioning** technique will be used. It consists on modifying the system equations in order to obtain a system matrix $W'$ having a **condition number** near unity [10], [20], [21]. The **condition number** $k(M)$ of a matrix $M$ is the ratio of the largest to the smallest eigenvalue of $M$. It is known that the larger the condition number of a system matrix, the more susceptible is the system to numerical errors [21]. So, it is considered an optimal **preconditioning** technique the one which achieves the minimum possible **condition number**.

A near optimal **preconditioning** is made by substituting matrix $W$ by $W' = BWB$, vector $\tilde{b}$ by $\tilde{b}' = B\tilde{b}$ and the bound constraints down and up by down' = $B^{-1}\text{down}$ and up' = $B^{-1}\text{up}$. Matrix $B$ is called **preconditioner** and it is a diagonal matrix with $b_{ii} = \beta_i = K/\sqrt{|w_{ii}|}$ [20]. The minimum $y'^*\text{ of the function } E' = -\frac{1}{2}y'^T W' y' - \tilde{b}'^T \tilde{y}'$ and the minimum $y'^*\text{ of the original cost function } E$ are related by $y'^* = B^{-1}\tilde{y}'$. The value of constant $K$ has no effect on the convergence speed of our network and $K = 1$ has been taken. The resulting **preconditioned** matrix $W'$ is

$$W' = \begin{bmatrix} -1.0000 & -0.9000 & -0.8000 \\ -0.9000 & -1.0000 & -0.5000 \\ -0.8000 & -0.5000 & -1.0000 \end{bmatrix}. $$

Applying the global convergence updating rule, constant values $c_i = 2/|u'|_{\max} = \frac{1}{3}$ for all $i$ have been taken. In Fig. 1(a) it is shown the time evolution of the optimization variables for this case, taking as starting point $\tilde{y}'(0) = [20; 20; 20]$. The network spends less than 50 iterations to reach the solution.

On the other hand, matrix $W'$ does not fulfill neither condition (10) nor condition (14) for exponential convergence. These conditions are, however, sufficient, not necessary ones. Let us test the network performance with the corresponding neuron updating rules. The same starting point as before will be used in the two cases. The first exponential convergence updating rule (12) gives $c_i = 1/|w_{ii}'| = 1$ for all $i$. The result obtained can be seen in Fig. 1(b). The network does not converge to a fixed point, but it stagnates at a two-length cycle: it remains oscillating between [20; 8.713; 16.2] and [20; 8.62; 17.13]. The second exponential convergence updating rule (15) gives $c_1 = 1/1.7$, $c_2 = 1/1.4$, and $c_3 = 1/1.3$. The result obtained can be seen in Fig. 1(c). The network converges and, actually, comparing with the global convergence schedule, it takes less iterations to reach the solution with this exponential convergence schedule. This good behavior, however, can not be assured a priori.

**Example 2:** Obtain the optimum solution $\tilde{y}^*$ of

$$\min\{E(\tilde{y}): -15 \leq y_i \leq 15 \text{ \quad } \forall i\} \text{ (19)}$$

where matrix $W$ and vector $\tilde{b}$ in (1) are

$$W = \begin{bmatrix} -39.6 & -3.96 \\ -3.96 & -0.53 \end{bmatrix}$$

and

$$\tilde{b} = \begin{bmatrix} -574.2 \\ -55.4 \end{bmatrix}.$$
Fig. 2. Time evolution of the estimates of the optimization variables for example 2: (a) result obtained with the first exponential convergence updating rule and (b) result obtained with the second exponential convergence updating rule.

The problem is

\[ f^p = \begin{bmatrix} -15 \\ 7.547 \end{bmatrix}. \]

The \textit{preconditioned} matrix is

\[ W^p = \begin{bmatrix} -1 & -0.8644 \\ -0.8644 & -1 \end{bmatrix}. \]

It fulfills the two conditions for exponential convergence giving, for the first one, a lower bound for the convergence degree $\eta_1 = 0.15$ and, for the second one, $\eta_2 = 0.078$. In fact, it is easy to prove that a $2 \times 2$ symmetric negative definite matrix \textit{preconditioned} with the technique explained above always fulfills these two conditions. In addition, the lower bound $\eta_1$ for the convergence degree obtained from condition 1 for exponential convergence is always greater than the lower bound $\eta_2$ obtained from condition (2).

The first exponential convergence updating rule gives $c_1 = \sigma_1 = 1$ and the corresponding result can be seen in Fig. 2(a). The second exponential convergence updating rule gives $c_1 = \sigma_2 = 0.5364$ and the corresponding result can be seen in Fig. 2(b). In both cases the point $\bar{y}^p(0) = [15; 15]$ has been taken as starting point. From these figures we see that, as expected, the quickest convergence has been achieved with the first rule. With this rule the network needs only ten iterations to reach the solution. It is illustrative to show a spatial representation of the trajectories followed by the network output vector. Those corresponding to Fig. 2(a) and (b) are represented, in the \textit{preconditioned} space, in Fig. 3(a) and (b), respectively. In these figures the contours of the \textit{preconditioned} cost function $E^p$ and the limits of the constraint region are also plotted. It can be noted that the two updating schedules produce very different trajectories. On the other hand, the result shown in Figs. 2(b) and 3(b) is qualitatively similar to that obtained by computer simulations of the continuous-time model presented in [10] when a small sampling time is used to discretize the model.

With respect to the global convergence updating rule, expression (17) gives $c^g \leq 2/n\|u\|_{\text{max}} = 1$. The best result is
attained with the maximum value $c_i = 1$ which, in this case
(for $n = 2$) coincides with the first exponential convergence
updating rule given in (12).

V. DISCUSSION

In general, in order to optimally use the neural network for
minimizing a particular cost function, the following procedure
is recommended. First to precondition the problem and test if
$W^T$ fulfills the conditions for exponential convergence. If it
does, use the updating rule which corresponds to the greatest
value of $\eta_i$. It must be remarked that for $\eta_i$ $\geq 3$ neither the
relation $\eta^*_i > \eta_2$ nor the converse $\eta_2 > \eta^*_i$ are generally true.
Thus, the two exponential convergence conditions have to be
tested. If $W^T$ does not fulfill such conditions, use the global
convergence updating rule.

With respect to the preconditioning technique, highly recom-
manded when a conventional gradient method is to be used
to improve numerical stability and convergence speed, it is also
useful for some of the neuron updating schedules developed.
However, it is easy to see that using the first exponential
convergence rule given in (12) without preconditioning is
equivalent to first make preconditioning and then using this
rule. So, when this updating schedule is used conditioning is
made implicitly.

On the other hand, it has been said that the neural network
presented can be included in the general class of primal methods
for constrained real optimization. Let us compare the neural-net behavior with that of the well known gradient projection
method, which belongs to the same class.

The searching trajectories are significantly different for both
methods. When the current point is at the interior of the
constraint region, the gradient projection follows the minus
gradient direction. With the neural network presented this only
happens when all the $c_i$ have the same value. When the current
point is at a boundary face of the constraint region, the gradient projection explores it following the direction of the projection
of the minus gradient onto such face. With the neural network,
in the same situation the searching is not confined to the
boundary.

From the programming point of view, the implementation
of the neural network algorithm is simpler than that of a
conventional primal method for the same application. For example, each iteration of the gradient projection method
involves several steps [2]. First to calculate the projection of the
cost function gradient onto the subspace defined by the active
constraints at the current point (for bound constraints, the projection calculation can be made simply by
making zero the corresponding gradient components). If this
projection is not null, minimize the cost function from the
current point along this direction inside the constraint region
(this step constitutes itself an optimization problem which
involves several calculations [2]). If this projection is null,
calculate the Lagrange multipliers corresponding to the active
constraints. If all the Lagrange multipliers are nonnegative, the
minimum has been found. If not all the Lagrange multipliers
are nonnegative, eliminate from the set of active constraints
the one corresponding to the most negative multiplier and go
to the first step.

The algorithm outlined in the above paragraph is more
complex than the neural model presented which only involves,
at each iteration, first to calculate the gradient, then to add to
the previous estimate of each variable the result of multiplying
the corresponding gradient component by the corresponding
constant $c_i$; and finally to apply the saturation function. This
simplicity makes it appropriate to be implemented in digital
hardware. In addition, parallel implementation is direct.

VI. CONCLUSIONS

In this paper, a discrete-time recurrent neural network which
solves the problem of quadratic real optimization with bound
constraints, as stated in (1) and (2), has been developed. A
general formulation of this network is given in (3) and (4). It is a gradient method and can be classified as a primal method
for constrained optimization, i.e., the approximate solutions
attained by the algorithm during the searching process are all
feasible solutions.

The necessary and sufficient condition for this network to
have only one equilibrium point which corresponds to the
constrained minimum of (1) is $c_i > 0$ for all $i$. Two neuron
updating rules, given in (12) and (15), have been developed
that assure exponential convergence of the network, provided
that matrix $W$ fulfills certain conditions given, respectively, in
(10) and (14). For each one of these rules, a lower bound of
the network convergence degree can be calculated by using,
respectively, expressions (13) and (16). Moreover, an updating
schedule, given in (17), that guarantees global convergence of
the network for any symmetric matrix has been obtained.

In the neural schedules developed, all the neurons are
updated synchronously, and only the $i$th gradient component
is used to calculate the estimated value of the $i$th optimization
variable at each iteration. So, a direct parallel implementation is
possible for the neural network, whereas parallelization of the
gradient projection or other conventional methods for
quadratic constrained real optimization is not straightforward.
In addition, since it is a simple discrete-time neural system,
it can be easily implemented digitally. The neuron updating
rules and convergence results presented will hold for such
digital implementations.

APPENDIX

PROOFS OF CONVERGENCE

Proof of Condition 1 for Exponential Convergence: The
norm definition $||A|| = \max_i \sum_j |a_{ij}|$ is used to calculate
$N = ||I + CW||$, thus obtaining $N = \max_i (\sum_j ||(I +
CW)_{ij}||) = \max_i (N_i)$ with

$$
N_i = \begin{cases}
1 + c_i \sum_j |w_{ij}| & \text{if } c_i \leq 0 \\
1 - c_i \left( \sum_j |w_{ij}| - \sum_{j \neq i} |w_{ij}| \right) & \text{if } 0 < c_i < \frac{1}{|w_{ii}|} \\
c_i \sum_j |w_{ij}| - 1 & \text{if } c_i \geq \frac{1}{|w_{ii}|}
\end{cases}
$$

(20)
where the fact that $w_{ij} < 0$ for all $i$ because $W$ is negative definite has been used. Two different situations have to be distinguished. If $|w_{ij}| - \sum_{j \neq i} |w_{ij}| \leq 0$ the corresponding term $N_i$ in the matrix norm is always greater or equal to one, independently of the value of $c_i$. However, if $|w_{ij}| - \sum_{j \neq i} |w_{ij}| > 0$ there exists a range of values of $c_i$, $0 < c_i < 1/2$, for which the corresponding $N_i$ is less than one. The minimum value of $N_i$ is reached for $c_i = 1/|w_{ij}|$, and this minimum is $(N_i)_{\text{min}} = \sum_{j \neq i} |w_{ij}|/|w_{ij}|$. These results correspond to expressions (10)–(13).

**Proof of Condition 2 for Exponential Convergence:** By using the norm definition $||A|| = (\sum_i \sum_j a_{ij}^2)^{1/2}$ to calculate $N^2 = ||I + CW||^2$ the following expression is obtained:

$$N^2 = \sum_{i,j} c_i^2 w_{ij}^2 + \sum_i (1 + c_i w_{ii})^2. \quad (21)$$

By differentiating $N^2$ with respect to $c_i$ it is easy to calculate the value $c_i$ which minimizes it and the corresponding $N^2_{\text{min}}$:

$$c_i = \frac{|w_{ii}|}{\sum_j |w_{ij}|^2}$$

and

$$N^2_{\text{min}} = \sum_i \sum_j |w_{ij}|^2$$

where the fact that $w_{ii} < 0$ for all $i$ because $W$ is negative definite has been used. In order to assure exponential convergence, $N^2_{\text{min}} < 1$ is imposed, thus obtaining the second sufficient condition given in (14). The expressions for $c_i$ and $N^2_{\text{min}}$ given above correspond, respectively, to (15) and (16).

**Proof of Global Convergence:** It will be proved that $E$ is a Lyapunov function of the network.

If the matrix $W$ is symmetric the change produced in the value of $E$ at each iteration is given by

$$\Delta E(k+1) = -\sum_{i=1}^n (u_i(k) + b_i) \Delta_i(k)$$

$$- \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n w_{ij} \Delta_i(k) \Delta_j(k)$$

$$= [I] + [II] \quad (23)$$

where $\Delta_i(k) = y_i(k+1) - y_i(k)$. Assuming that $c_i > 0 \Rightarrow \text{sign}(\Delta_i(k)) = \text{sign}(u_i(k) + b_i)$ and taking into account that the nature of the constraints given in (4) assures that $\text{sign}(\Delta_i(k)) = \text{sign}(\Delta_i(k))$, the first term in the right-hand side is negative. Therefore, to obtain a negative variation in the value of the function $E$ it is sufficient to assure that the absolute value of the second term is smaller than the absolute value of the first one. Making use first of Schwartz’s inequality and then of Chebyshev’s inequality [22] it is possible to obtain an upper bound for the absolute value of $[II]$

$$|[II]| \leq \frac{|v|_{\text{max}}}{2} \left( \sum_{i=1}^n \Delta_i(k)^2 \right) \leq \frac{|v|_{\text{max}}}{2} \left( \sum_{i=1}^n (u_i(k) + b_i) \Delta_i(k) \right)^2 \quad (24)$$

where $|v|_{\text{max}}$ is the maximum of the matrix elements absolute values. Then to fulfill the condition $|[E]| \leq |[I]|$ it is sufficient that

$$\frac{|v|_{\text{max}}}{2} \sum_{i=1}^n (u_i(k) + b_i) \Delta_i(k) \leq \sum_{i=1}^n (u_i(k) + b_i) \Delta_i(k). \quad (25)$$

A simple way to assure it is to make each term in the summation of the left-hand side of (25) smaller than the corresponding term in the summation of the right-hand side. Taking into account that $\text{sign}(\Delta_i(k)) = \text{sign}(u_i(k) + b_i)$, it can be reached if

$$|\Delta_i(k)| \leq \frac{2}{n|v|_{\text{max}}} |u_i(k) + b_i|, \quad \forall i. \quad (26)$$

Finally, the nature of the constraints given in (4) assures that $|\Delta_i(k)| \leq |\Delta_i(k)|$. With all that the sufficient condition for global stability given in (17) can be easily obtained.

**REFERENCES**


María José Pérez-Ilzarbe was born in Pamplona, Spain, in 1961. She received the B.S. degree in physics from the Universidad de Zaragoza in 1984. During 1985 and 1986, she worked in the field of photovoltaic solar energy, at the Instituto de Energías Renovables (Centro de Investigaciones Energéticas Medioambientales y Tecnológicas). From 1987 to 1990, she was a predoctoral research student at the Instituto de Óptica (Consejo Superior de Investigaciones Científicas), where she worked in digital image processing, and received the Ph.D. degree in physics from the Universidad Complutense de Madrid in 1990.

In 1991, she joined the Universidad de Zaragoza and was an Assistant Lecturer in automatics at the Departamento de Ingeniería Eléctrica e Informática. In 1992, she joined the Universidad Pública de Navarra, where she presently is a Senior Lecturer in automatics at the Departamento de Automática y Computación. Her research interests include neural networks, fuzzy systems, control systems, signal and image processing, and artificial vision.