PERIODICA POLYTECHNICA SER. EL. ENG. VOL. 41, NO. 1, PP. 53-62 (1997)

NEURAL CIRCUITS FOR SOLVING NONLINEAR PROGRAMMING PROBLEMS

József BÍRÓ*, Miklós BODA**, Zoltán KORONKAI*, Edith HALÁSZ*, András FARAGÓ*, Tamás HENK*, and Tibor TRÓN*

*High Speed Networks Laboratory
Department of Telecommunications and Telematics
Technical University of Budapest,
H-1521 Budapest, Hungary
E-mail: biro@ttt-atm.ttt.bme.hu
**Ellemtel Telecommunications Systems Laboratories, Sweden
E-mail: Miklos.Boda@eua.ericsson.se

Received: March 5, 1997

Abstract

This paper is concerned with neural networks which have the ability to solve linear and nonlinear constrained optimization tasks. After a short overview of such neural networks, we introduce useful extensions which make them capable of solving more general programming tasks, namely handling equality constraints more efficiently than in the known obvious ways, and obtaining global optimum with probability close to 1. We also refer to stability analysis worked out from both the circuit theory and optimization theory point of view. The simple simulation examples, one of which is presented in the paper, show that the extended networks are stable and converge to right solutions.

Keywords: neural networks, optimization, nonlinear programming.

1. Introduction

As it is well known, artificial neural networks (ANN) contain many simple processing elements connected with each other in some way. Two main properties make them attractive in several applications: the capability of 'learning' and the dramatically increased speed when used as parallel computation structures. Neural approaches of optimization problems have also been extensively studied by many researchers. Between the two main types of ANN architectures, feedforward and recurrent, the latter one, which serves the basis of our work, is more suitable for solving optimization tasks.

In recent years, with the rapid development of analogue VLSI technology, the interest in continuous systems and their analogue circuit realizations keeps growing. Simple 'analogue computers' have also been developed showing the remarkable results in this field. The analogue and mixed

(analogue-digital) circuits are important in the implementation of neural networks, especially of recurrent ones with continuous dynamics [1] [4] [5].

In the last decade there has been strong research activity in applying analogue circuits for solving mathematical programming tasks. The concept was originated by PYNE [2] who proposed an analogue electronic computer for solving linear programs. More recently, CHUA and LIN [3] developed a circuit to solve nonlinear programming problems. The main drawback of this circuit is that multiport transformers are required for realization and thus, cannot solve quadratic programs having negative semidefinite matrices. This problem has been resolved by Wilson, using negative floating resistors. Hopfield and Tank presented a neural network for linear programming in which the decision variable amplifiers have inverting outputs, as well, therefore, negative resistors are not needed [4]. Later it has turned out, on the basis of the work of KENNEDY and CHUA [5], that the Hopfield network can be viewed as a special case of the canonical programming circuit proposed by CHUA and LIN. Moreover, modification is required in penalizing the constraints in order to guarantee convergence to a feasible solution.

The significance of the works mentioned above lies in that the optimization problems appear in a common framework of neural paradigm and analogue circuit implementations (hence is the name neural circuits).

In this paper we introduce new results related to nonlinear programming neural networks. The attractive features of our proposed extensions are that they allow us handling equality constraints more efficiently than previous solutions and, at the same time, they increase the probability to reach global optimum.

The paper is organized as follows: The second part gives information on constrained optimization tasks and KENNEDY and CHUA'S canonical programming circuit [7] which serves as the basis of our work. The third section contains the essence of the article, the description of our extended neural networks, and arguments for the usefulness and feasibility of the new constructions based on analytical and simulation investigations. Finally, the paper is concluded and summarizes the possible prospective research directions.

2. Constrained Optimization by Neural Networks

2.1 Constrained Optimization: Notation and Formulation

A constrained optimization problem can be formulated in the form:

$$\text{Minimize} f(x) \text{subject to} g_j(x) \geq 0,\ j = 1..p, \tag{2.1}$$

where $x \in \mathbf{R}^n, f : \mathbf{R}^n \to \mathbf{R}^p$ and $g : \mathbf{R}^n \to \mathbf{R}^p$ is a p dimensional vector valued function of n variables. We also assume that f and g are continuously differentiable functions. Using the penalty function method, the above task can be transformed into an unconstrained optimization:

$$\text{Minimize}\,\{f(x) + cP(x)\}, \tag{2.1a}$$

where c is a sufficiently large constant and P is the so-called penalty function. P is always a non-negative function and takes zero if and only if all constraints are satisfied. Frequently used penalty functions are

$$P_q(x) = \frac{1}{q}\sum_{j=1}^{p}[g_j^-(x)]^q g_j^-(x) = -\min(0, g_j(x)) \text{ and } \quad q > 0, \text{ integer} \tag{2.1b}$$

which satisfy the requirement that if violation occurs it should be positive, otherwise zero.

A more general optimization task involves equality constraints, in addition to the inequality ones. Let $h : \mathbf{R}^n \to \mathbf{R}^r$ denote the function in the set of equality constraints $(h(x) = 0)$. Then the problem can be formulated as:

Minimize $f(x)$

$$\text{subject to } g(x) \geq 0 \quad \text{and} \quad h(x) = 0. \tag{2.2}$$

In this case, the unconstrained cost function to be minimized is

$$\varphi(x) = f(x) + cP(x) + dQ(x), \tag{2.2a}$$

where Q is the function penalizing the violation of equality constraints. Q is usually quadratic but obtaining a more general set of allowed penalty functions, let Q be defined as:

$$Q_q(x) = \frac{1}{q}\sum_{j=1}^{r}[h_j(x)]^q(2\varepsilon(h_j(x)) - 1)^q, \quad q > 0, \text{ integer}, \tag{2.2b}$$

where ε is the Heaviside step function. The difference between the minimizers of the constrained and unconstrained problem can be made arbitrarily small (or zero at certain tasks with $q = 1$ parameters) by using sufficiently large c and d.

2.2 The Canonical Nonlinear Programming Neural Circuit

KENNEDY and CHUA have proposed a circuit called dynamical canonical nonlinear programming circuit which is intended to solve problems formulated in (2.1) [7]. The architecture can be seen in *Fig. 1* which shows the

circuit containing controlled voltage and current sources, nonlinear resistors and capacitances. The latter ones allow dynamical behaviour and their voltages represents the decision variables. The left-hand side generates the penalty functions for the corresponding constraints, while on the right-hand side the required derivatives appear. It is easy to show that the circuit can be regarded as a gradient system and it simulates an unconstrained optimization problem like (2.1a) having $P_2(x)$ penalty function [8], [9].

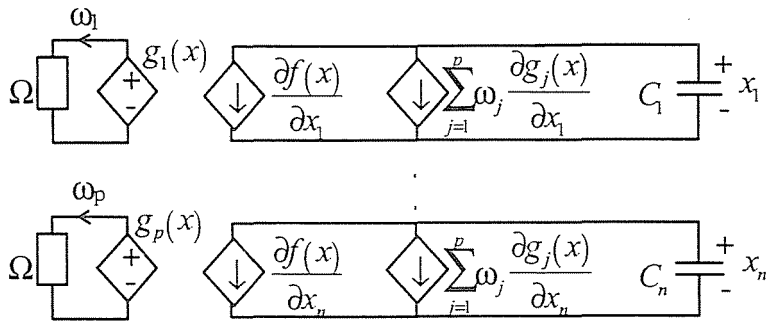

Fig. 1. The canonical nonlinear programming neural circuit

The stability analysis found in [7] shows that the network can converge to correct solutions under well-defined circumstances. However, there have been two problems remained to be solved. One of them is related to the efficient treatment of the tasks described in (3.2), the other one is the obvious drawback of gradient systems: the lack of global optimization. Both of them are attacked in the next section.

3. Extensions of the Canonical Programming Neural Circuit

3.1 Handling Equality Constraints

The first modification is developed for handling equality constraints in order to make the canonical neural circuit capable of solving more general optimization task formulated in the form of:

$$\text{Minimize } f(x) \text{subject to } \quad g_j(x) \geq 0 \quad j = 1..p \quad \text{and} \quad h_i(x) = 0, i = 1..r. \tag{3.1}$$

Naturally, there are some obvious ways of taking into account the constraints $h_i(x) = 0$. For example, one of them is considering all equalities as

two inequalities, that is $h_i(x) = 0$ is equivalent to $h_i(x) \geq 0, h_i(x) \leq 0$. Another possibility is to use simple quadratic penalty terms for equalities and embedding them into $f(x)$ as can be seen in [12]. Hence, in this manner the problem can be transformed to that like in (2.1). Although these approaches are quite simple and applicable, they have some drawbacks, particularly in connection with the possible circuit realization. In the case of the first method the canonical circuit requires $2r$ additional elements, corresponding pairs of which are responsible for equality constraint fulfilment. It is a prodigal solution, because at most one of the two nonlinearities is active in each pair at any time. In addition, it involves another requirement, the two nonlinearities of each pair should work completely synchronously, otherwise it can occur that both of the inequalities $h_i(x) \geq 0, h_i(x) \leq 0$ are considered being violated leading to unnecessary oscillation and pathological behaviour in the circuit. The second approach should not even be preferred, because the neural circuit modified in this manner would lose an attractive feature, i.e. the topologically well-separated treatment of constraints, which exists in the original canonical neural circuit. Preserving this property is very important in obtaining an easily reconfigurable circuit.

In view of these facts, we suggest using special nonlinearities which help overcome the difficulties listed above. They are characterized by:

$$\Omega_{jq}(w) = c_j w^{q-1}(-\varepsilon(-w))^{q-2} \quad j = 1..p, \quad q > 0, \quad \text{integer}, \tag{3.2}$$

$$\Lambda_{iq}(w) = d_i w^{q-1}(2\varepsilon(w) - 1)^{q-2} \quad i = 1..r, \tag{3.3}$$

where $c_j, d_j \in \mathbf{R}$ and ε is the Heaviside step function. The *qth order* Q_{jq} and Λ_{iq} belong to the jth inequality and ith equality constraint, respectively. (*Omega* and Λ are conductance characteristics as it can be seen in *Fig. 2*). Therefore, in accordance with circuit dynamics, the time evolution of state variables x_k is determined by the following differential equation:

$$\frac{dx_k}{dt} = -\frac{1}{C_k}\left[\frac{\partial f(x)}{\partial x_k} + \sum_{j=1}^{p} c_j \omega_j \frac{\partial g_j(x)}{\partial x_k} + \sum_{i=1}^{r} d_i \lambda_i \frac{\partial h_i(x)}{\partial x_k}\right], \quad k = 1..n. \tag{3.4}$$

For establishing connection to the optimization task found in (2.4), let us define the unconstrained objective function $\varphi_{mod}(x)$ as:

$$\varphi_{\text{mod}}(x) = f(x) + P_q(x) + Q_q(x), \tag{3.5}$$

where $P_q(x)$ and $Q_q(x)$ are the *modified qth order* penalty functions:

$$P_q(x) = \frac{1}{q}\sum_{j=1}^{p} c_j [g_j(x)]^q \, (-\varepsilon(-g_j(x)))^q \quad q > 0, \text{ integer}, \tag{3.6}$$

$$Q_q(x) = \frac{1}{q} \sum_{j=1}^{r} d_j [h_j(x)]^q \left(2\varepsilon(h_j(x)) - 1\right)^q . \tag{3.7}$$

The following theorem makes the relation between the modified canonical circuit and the task in (2.4) more clear.

Theorem 1 : *The modified canonical neural circuit with qth order Λ_q and Q_q nonlinearities solves the unconstrained mathematical programming task having modified qth order penalty terms P_q and Q_q.*

Proof :

The energy function of the modified canonical neural circuit is:

$$L(x) = f(x) + \sum_{j=1}^{p} \int_0^{g_j(x)} \Omega_{jq}(w)dw + \sum_{j=1}^{r} \int_0^{h_j(x)} \Lambda_{jq}(w)dw. \tag{3.8}$$

Introducing the variables

$$\omega_j = \Omega(g_j(x)) \quad \text{and} \quad \lambda_j = \Lambda(h_j(x)) \tag{3.9}$$

referring to the conductance characteristics of the applied nonlinearities, the time derivative of $L(x)$ can be expressed as:

$$\frac{\partial L(x)}{\partial t} = \sum_{k=1}^{n} \frac{\partial f(x)}{\partial x_k} \frac{\partial x_k}{\partial t} + \sum_{j=1}^{p} \sum_{k=1}^{n} c_j \omega_j \frac{\partial g_j(x)}{\partial x_k} \frac{\partial x_k}{\partial t} + \sum_{j=1}^{r} \sum_{k=1}^{n} d_j \lambda_j \frac{\partial h_j(x)}{\partial x_k} \frac{\partial x_k}{\partial t}. \tag{3.10}$$

Substituting governing *Eq.* (3.4) into (3.10), we obtain the following more concise form:

$$\sum_{k=1}^{n} \frac{\partial x_k}{\partial t} \left[-C_k \frac{\partial x_k}{\partial t} \right] \tag{3.11}$$

which is evidently less than or equal to zero. Consequently, the system is stable in Lyapunov sense meaning that the dynamical equation (2.7) continuously decreases the energy function $L(x)$ until it reaches its stable equilibrium point.

As the second step, we show that $L(x)$ as a Lyapunov function is equivalent to the objective function $\varphi_{\text{mod}}(x)$. Considering the following identities:

$$\int w^{q-1} \left(2\varepsilon(w) - 1\right)^{q-2} dw = \frac{1}{q} w^q \left(2\varepsilon(w) - 1\right)^q , \tag{3.12}$$

$$\int w^{q-1} \left(-\varepsilon(-w)\right)^{q-2} dw = \frac{1}{q} w^q \left(-\varepsilon(-w)\right)^q \tag{3.13}$$

and substituting them into (3.8) we get the required formula of $\varphi(x)$.

3.2 A Stochastic Version of the Modified Canonical Circuit

It is well known that finding the global optimum cannot be guaranteed using a pure gradient system. Regarding neural networks, numerous methods have been developed for finding global optimum mainly based on simulated annealing (SA), mean field annealing [9], [10] and the continuous versions of mean field techniques [14], [15], [16].

In [9] WONG has developed a neural circuit which is based on a diffusion machine using a deterministic approach of SA. KESIDIS has showed that this network, in principle, can be used for analogue optimization [18]. It is also known that SA can be extended to continuous variable cases based on the Langevin algorithm. That allows us to apply the concept in the modified canonical circuit meaning that additional current generators are inserted in parallel with the previous ones on the right-hand side (see *Fig. 2*) which generate uncorrelated gaussian noises whose amplitudes are gradually being reduced as the time evolves.

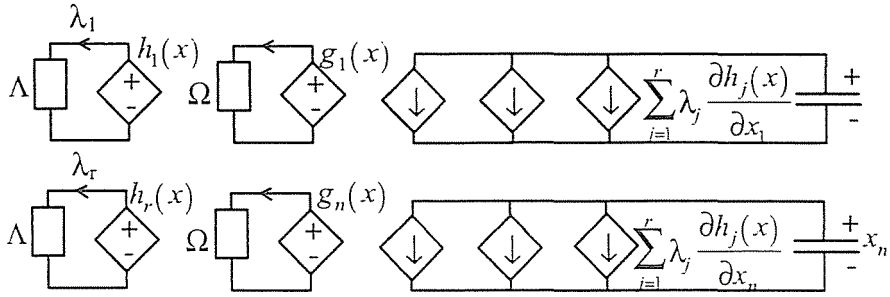

Fig. 2. The modified nonlinear programming circuit

In this case the network dynamics can be derived on the basis of Ito-type stochastic differential equations appeared in [9]:

$$\frac{\partial x_k(t)}{\partial t} = -C_k^{-1}\frac{\partial L(x)}{\partial x_k} + \sqrt{2T}\eta_k(t), \quad k = 1..n, \tag{3.14}$$

where T is the artificial temperature decreasing by logarithmic rule.

The above-mentioned approach is also feasible to circuit implementations because there exist efficient methods for providing independent gaussian noise sources developed for analogue VLSI neural circuits [14].

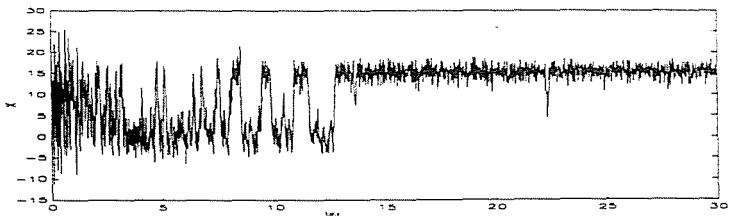

Fig. 3. The output of the stochastic neural circuit

3.3 On Stability and Simulation Results

Regarding the modified canonical circuit (without noise sources) we can also use the co-content function known from circuit theory as a Lyapunov function in the form of:

$$L(x) = f(x) + \sum_{j=1}^{p} \int_{0}^{g_j(x)} \Omega_q(w)dw + \sum_{j=1}^{r} \int_{0}^{h_j(x)} \Lambda_q(w)dw. \tag{3.15}$$

The obvious implication of Theorem 1 is that this function is strictly decreasing during the relaxation of the circuit, so according to the Lyapunov stability theorem the network converges to an equilibrium point at which $L(x)$ takes zero value. It is also right that $L(x)$ is equivalent to the objective function found in (3.5). Therefore, the state variable x at equilibrium state is a local minimizer of $\varphi_{\text{mod}}(x)$.

In the case of the stochastic version the Lyapunov technique is no longer applicable for stability. Instead, we have to use properties of the corresponding Ito type stochastic differential equation from which the network dynamics is derived. The η_k noises must be generated by independent Gaussian sources in order to make x_k stationary at a given temperature. If the T schedule is slow sufficiently the system goes through quasi-stationary states ensuring the stability and reaching global optimum with high probability.

The computer simulations based on the approximation of the differential equations demonstrated that the modified canonical circuit, according to the analytical results, does not produce large oscillations or chaotic behaviour, but converges to points close enough to the right solution. The stochastic version also works satisfactorily. In this case there are some parameters which have to be tuned mainly based on experience. In *Fig. 3* we

can see an example showing the behaviour of one output of the stochastic modified canonical circuit. During the operation x spends more and more time in the neighbourhood of 0 and 15, respectively, then finally it 'gets stuck' at 15, the global optimum.

4. Conclusion

In this paper we have introduced a modified canonical circuit and its stochastic version for handling equality constraints efficiently and obtaining global optimization. The analytical investigations and computer simulations show the usefulness of these networks. In the framework the possibility of using some kind of hardware annealing and the more rigorous analysis of the stochastic modified canonical circuit are not elaborated in detail, and thus, they form good starting points for future research.

Acknowledgement

This research was performed as part of a joint project between Ellemtel Telecommunication Systems Laboratories and the Department of Telecommunications and Telematics, Technical University of Budapest. The authors are grateful for the continuous support of Harald Brandt and Géza Gordos.

References

1. ROSKA, T. – CHUA, L. O.: The CNN Universal Machine: an Analogic Array Computer *IEEE Trans. on Circuit and Systems*, Vol. CAS-40, pp. 163–173, March, 1993.
2. PYNE, I. B.: 'Linear Programming on an Analogue Computer', *Trans. Am. Ins. Ele. Eng.*, Vol. 75, pp. 139–143, May, 1956.
3. CHUA, L. O. – LIN, G.: Nonlinear Programming without Computation, *IEEE Trans. on Circuits and Systems*, Vol. CAS-31, No. 2, February, 1984.
4. HOPFIELD, J. J. – TANK, D. W.: Neural' Computation on Decisions Optimization Problems, *Biological Cybern.*, Vol. 52, pp. 141–152, 1985.
5. TANK, D. W. – HOPFIELD, J. J.: Simple 'Neural' Optimization Networks: An A/D Converter, Signal Decision Circuit and a Linear Programming Circuit", *IEEE Trans. on Circuits and Systems*, Vol. CAS-33, No. 5, May, 1986.
6. KENNEDY, M. P. – CHUA, L. O.: Unifying the Tank and Hopfield Linear Programming Network and the Canonical Nonlinear Programming Circuit of Chua and Lin, *IEEE Trans. on Circuits and Systems*, Vol. CAS-34, No. 2, pp. 210–214, February, 1987.
7. KENNEDY, M. P. – CHUA, L. O.: Neural Networks for Nonlinear Programming", *IEEE Trans. on Circuits and Systems*, Vol. CAS-35, No 5, pp. 554–562, May, 1988.
8. LILLO, W. E. et. al.: On Solving Constrained Optimization Problems with Neural Networks: A Penalty Method Approach", *IEEE Trans. on Neural Networks*, Vol. 4, No. 6, pp. 931–940, November, 1993.

9. MAA, C.: Linear and Quadratic Programming Neural Networks Analysis, *IEEE Trans. on Neural Networks*, Vol. 3, No. 4, pp. 580–594, July, 1992.
10. WONG, E.: Stochastic Neural Networks, *Algorithmica*, Vol. 6, No. 3, pp. 467–478, 1991.
11. PETERSON, C. –SÖDERBERG, B. : A New Method for Mapping Optimization Problems onto Neural Networks, *Int. Journal of Neural Systems*, Vol. 1, No. 1, pp. 3–22, 1989.
12. WHITTLE, P.: Optimization under Constraints, Wiley-Interscience, 1971.
13. KARMANOV, V. G. Mathematical Programming, Mir Publishers, Moscow, 1989.
14. J. ALSPECTOR et al.: A VLSI-Efficient Technique for Generating Multiple Uncorrelated Noise Sources and Its Application to Stochastic Neural Networks, *IEEE Trans. on Circuits and Systems*, Vol. 38, No. 1, pp. 109–123, January, 1991.
15. LEE, B. W. – SHAU, B. J.: Hardware Annealing in Electronic Neural Networks, *IEEE Trans. on Circuits and Systems*, Vol. 38, No. 1, pp. 134–137, January, 1991.
16. LEE, B. W. – SHAU, B. J.: Paralleled Hardware Annealing for Optimal Solutions on Electronic Neural Networks, *IEEE Trans. on Circuits and Systems*, Vol. CAS-40, No. 4, pp. 588–598, July, 1993.
17. ROMEO, F. – SANGIOVANNI-VINCENTELLI, A.: A Theoretical Framework for Simulated Annealing, *Algorithmica*, Vol. 6, No. 3, pp. 302–345, 1991.
18. KEISIDIS, G.: Analog Optimization with Wong's Stochastic Neural Network, *IEEE Trans. on Neural Networks*, Vol. 6, No. 1, pp. 258–260, January, 1995.