APPLICATION OF RECURRENT NEURAL NETWORK USING MATLAB SIMULINK IN MEDICINE

Raja Das

Madhu Sudan Reddy
VIT Unversity
Vellore, Tamil Nadu
India
rdasresearch@gmail.com

Abstract. In this paper, a recurrent neural network (RNN) for finding the solution of linear programming problems is presented with better, spontaneous and fast converging. To achieve optimality in accuracy and also in computational effort, an algorithm is also presented. This paper covers the MATLAB Simulink modeling and simulative confirmation of such a recurrent neural network. Modeling and simulative results validate the theoretical analysis and efficiency of the recurrent neural network for finding the solution for linear programming problem. An application RNN in medicine has been presented to show the performance of the recurrent neural network.

Keywords: linear programming problem, Recurrent Neural Network (RRN), MATLAB Simulink.

1. Introduction

Linear programming (LP) techniques are commonly used to find the solution of economic, military, industrial, and social problems. In the last 50 years, researchers have proposed various dynamic tools for solving LP problems. Pyne [1] proposed the dynamic systems approach to solve constrained optimization problems. The recent developments in this field have redefined the application of neural network by dynamic solvers [2, 3, 4].

In recent studies, many researches have been done in relation to the application of the technology to knowledge engineering such as Artificial Neural Network (ANN) to the engineering field. LP is a very traditional discipline of Operations Research. It continues to be the most active branch. LP has been widely applied practically in many sectors such as production, financial, human resources, governing and planning. In the present day scenario, it is possible to construct and solve linear programs with high-speed computers and multi-processors which was not feasible a couple of years ago.

In 1947 Dantzig [5] developed a method for finding the solution to linear programming problems which is the Simplex method. Brown and Koopmans [6] portrayed the first series of interior point method to solve linear programming problems. Karmakar [7] built an algorithm which appears to be more proficient

than the Simplex Method on some intricate real-world problems of scheduling, routing and planning. Conn [7] developed a substitute for solving LP problems by using unconstrained optimization procedure combined with penalty function methods. The first neural approach applied to LP problems was proposed and developed by Tank and Hopfield [2].

The simulation models are designed as a stand alone application using MATLAB Simulink [8]. Matlab Simulation Models have been widely used [9, 10]. The primary aim of this paper is to present a recurrent neural network for finding the solution of linear programming problems. Here I describe circuit implementation of proposed neural network using MATLAB Simulink. The vital advantage of the neural networks is their massive parallel processing ability and rapid convergence properties.

2. Linear programming

Linear Programming is the term used for defining a wide range of optimization problems in which the objective function to be minimized or maximized is linear in the unknown variables and the constraints are a combinations of linear equalities and inequalities. LP is one of the most widely applied techniques of operations research in business, industry and numerous other fields. The objective function may be profit, cost, production capacity or any other measure of effectiveness, which is to be obtained in the best possible or optimal manner. The constraints may be imposed by different resources such as market demand, production process and equipment, storege capacity, raw material availibilty, etc. First, the given problem must be presented in linear programming form. This requires defining the variables of the problem, establising inter-relationship between them and formulating the objective function and constraints. A model, which approximates as closely as possible to the given problem, is then to be developed. If some constraints happen to be non-linear, they are approximated to appropiate linear functions to fit the linear programming format.

3. Problem statement

Consider a standard form of Linear Programming Problems described as

$$\textbf{Minimize } C^T X \tag{1}$$

$$\textbf{Subject to } AX = B \tag{2}$$

$$X \geq 0, \tag{3}$$

where $X \in R^n$ is a column vector of decision variables, $C \in R^n$ and $B \in R^m$ are column vector of cost coefficient and right hand side parameters, respectively, $A \in R^{m \times n}$ is a constraint coefficient matrix, and the subscript denotes the transpose operator.

4. Solution to the LP Problem

In general, the LP problem can have four possible solution types:

1. Unique Solution: There is only one solution that satisfies all constraints, and the objective function reaches a minimum within the feasible region.
2. Nonunique Solution: There are several feasible solution where the objective function reaches a minimum.
3. An unbounded Solution: The objective function is not bounded in the feasible region and it approaches $-\infty$.
4. No feasible Solution: Constraint provided in (2) and (3) are too restrictive, and the set of feasible solution is empty.

Although theatrically valid, cases 3 and 4 appear rarely in engineering and scientific applications. Furthermore, they can be easily detected, and in further consideration of the LP problem we will assume that it is formulated in such a way that there exists at least one feasible solution

5. The neural network

An artificial Neural Network (ANN) is a dynamic system, consisting of highly interconnected and parallel non-linear processing elements, that is highly efficient in computation. In this paper, a recurrent neural network [7] with equilibrium points representing a solution of the constrained optimization problem has been developed. As introduced in Hopfield [2] these network are composed with feed back connection between nodes. In the standard case, the nodes are fully connected i.e., every node is connected to all other nodes, including itself.

The first step in a neural network implementation for solving the LP problem is to define an energy function that can be optimized in an unconstrained fashion. Therefore, the method of Lagrange multiplier is applied in the network. To accomplish this, the linear constraints and non negativity constraints are appended to the objective function in some convenient way. Commonly the constraints are incorporated as penalty terms that, when ever violated, increase the value of the energy function. One energy function that can be derived using the Lagrange multiplier method are defined as

$$E(X,Y) = C^T X + Y^T(AX - B) - \alpha Y^T Y \tag{4}$$

with $\alpha \geq 0$,$Y \in R^{m\times 1}$, and $X \geq 0$.

Applying the method steepest descent in discrete time, I compute the gradient of the energy function in (4) with respect to X and obtain

$$\frac{\partial E}{\partial X} = \frac{\partial}{\partial X}\left[C^T X + Y^T(AX - B) - \alpha Y^T Y\right]$$

and

$$\frac{\partial E}{\partial X} = C + A^T Y. \tag{5}$$

In a similar manner we have

$$\frac{\partial E}{\partial Y} = AX - B - \alpha Y \tag{6}$$

Based on (5) and (6), the node equation for the discrete-time network with n-neurons is given by

$$x_i(k+1) = \begin{cases} x_i(k) - \mu\big[c_i + \sum a_{ji} y_j(k)\big] & \text{if } x_i(k+1) > 0 \\ 0 & \text{if } x_i(k+1) \le 0 \end{cases} \tag{7}$$

and

$$y_j(k+1) = y_j(k) + \eta\big[z_j(k) - \alpha y_j(k)\big],$$

where $z_j(k) = AX - B$, $\alpha \ge 0$, and $\mu, \eta \ge 0$ are learning rate parameters.

Note that the update equations for the independent-variable vector in (7) guarantee that all the components remain nonnegative. A neural network architecture realization of this process is presented in Figure 1.

6. MATLAB Simulink

The Simulink toolbox is a useful software package to develop simulation models for recurrent neural network applications in the MATLAB Simulink environment. With its graphical user interface and extensive library, it provides researchers with a modern and interactive design tool build simulation models rapidly and easily. Simulink is an input/output device GUI block diagram simulator. It opens with the library browser and library browser is used to build simulation models. The library browser contains continuous system model elements, discontinuous system models elements, and list of math operation elements. In the library browser, Sink elements are used for displaying and Source elements are used for model source functions. Model elements are added by selecting the appropriate elements from the library browser and dragging them into model window to create the required model.

7. Application of RNN in Medicine

First, we give numerical example to demonstrate the solution of linear programming problem through the proposed recurrent neural network.

A patient in a hospital is required to have at least 84 units of drug A and 120 units of drug B each day. Each gram of substance M contains 10 units of drug A and 8 units of drug B, and each gram of substance N contains 2 units of drug A and 4 units of drug B. Now suppose that both M and N contain an

Table 1: Comparison of the simulation result with the exact solution

Variables	Neural Network Result	Exact Solution
x_1	3.85	4
x_2	22.24	22

undesirable drug C, 3 units per gram in M and 1 unit per gram in N. How many grams of substance M and N should be mixed to meet the minimum daily requirements at the same time minimize the intake of drug C? How many units of the undesirable drug C will be this mixture?

To form the mathematical model, we start by identifying the decision variables. Let x_1 be the number of grams of substance and x_2 be the number of substance N used. The objective is to minimize the intake of drug C. In terms of the decision variables, the objective function is $C = 3x_1 + x_2$ which gives the amount of the undesirable drug C in x_1 grams of M and x_2 grams of N. Considering the above constraint inequalities in terms of the decision variables x_1 and x_2 and including the objective function we obtain the following linear programming model.

$$\begin{aligned} \textbf{Minimize } & C = 3x_1 + x_2 \\ \textbf{Subject to } & \\ & 10x_1 + 2x_2 \leq 84 \\ & 8x_1 + 4x_2 \leq 120 \\ & x_1, x_2 \geq 0. \end{aligned}$$

From the problem statement it can be seen that the linear programming problem is not in the standard form. By adding surplus variables x_3 and x_4 , the linear programming problem can be transferred in the standard form as follows:

$$\begin{aligned} \textbf{Minimize } & C = 3x_1 + x_2 + 0x_3 + 0x_4 \\ \textbf{Subject to } & \\ & 10x_1 + 2x_2 - x_3 \leq 84 \\ & 8x_1 + 4x_2 - x_4 \leq 120 \\ & x_1, x_2, x_3, x_4 \geq 0. \end{aligned}$$

To solve this linear programming problem, the MATLAB Simulink model for recurrent neural network in Figure.1 is simulated. The parameters of the network were chosen as $\mu = 0.01$, $\eta = 0.01$. Zero initial conditions were assumed both for X and Y. The network converges in approximately 4,000 iterations.

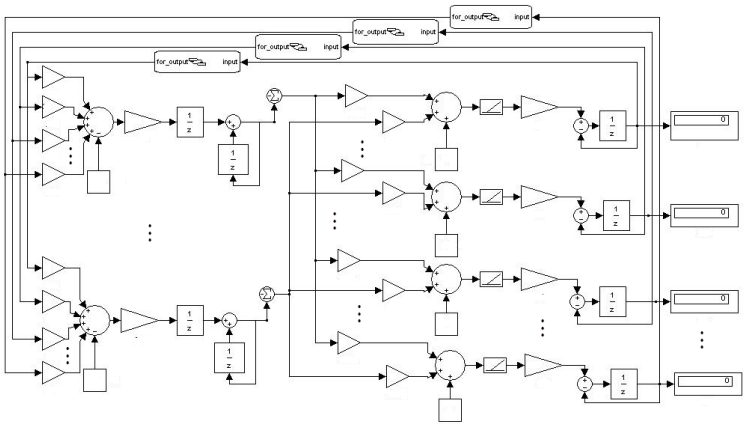

Figure 1: Recurrent Neural Network

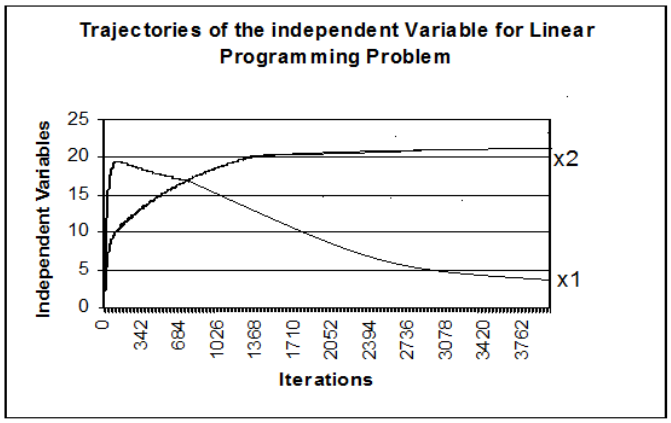

Figure 2: Recurrent Neural Network

8. Conclusion

We investigated in this paper the MATLAB Simulink modeling and simulative verification of such a recurrent neural network. Finding solution of linear programming problems through recurrent neural network approach is an interesting area of research. The energy function of the linear programming is defined in this paper. A circuit has been designed for the purpose. By using click-and-drag mouse operations in MATLAB Simulink environment, we could quickly model and simulate complicated dynamic systems. It has been concluded that recurrent neural network can be used for determining the solution of medicine problem. It converges to the exact solutions of the medicine Problem. Modeling and simulative results substantiate the theoretical analysis and efficacy of the recurrent neural network for solving the linear programming problem.

References

[1] I. B. Pyne, *Linear programming on an electronic analogue computer*, Trans. Amer. Inst.Eng., 75 (1956), 139-143.

[2] D. W. Tank and J. J. Hopfield, *Simple neural optimization network s: An A/D converter, signal decision circuit and a linear programming circuit*, IEEE Trans. Circ. Syst., vol. CAS-33 (1986), 533-541.

[3] M. P. Kennedy and L. O. Chua, 1988, *Neural networks for nonlinear programming*, IEEE Trans. Circ. Syst., vol. 35 (1988), 554-562.

[4] W. E. Lillo, M. H. Loh, S Hui and S. H. Zak, *On solving constrained optimization problems with neural networks: a penalty function method approach*, Tech. rep. TR-EE 91-43 (1991), Purdue Univ., W. Lafayette IN.

[5] G. B. Dantzig, *Linear Programming and Extensions*, Princeton NJ, Princeton Press, 1963.

[6] G. W. Brown and T. C. Koopmans, *Computation suggestions for maximizing a linear function subject to linear equalities in Activity Analysis of Production and Allocation*, T. Koopmans. Ed. New York, J. Wiley, 1951, 377-380.

[7] A. R. Conn, *Linear programming via a non differentialable penalty function*, SIAM J. Num. Anal., 13 (1976), 145-154.

[8] Book, *SIMULINK, Model-Based and System-Based Design, using simulink*, Math Works Inc., Natick, MA, 2000.

[9] C. D. Vournas, E. G. Potamianakis, C. Moors, and T. V. Cutsem, *An educational simulation tool for power system control and stability* IEEE Trans Power Syst., 19 (2004), 48-55.

[10] R. Patel, T. S. Bhatti, and D. P. Kothari, *MATLAB / Simulink-based transient stability analysis of a multimachine power system*, Int J Electr Eng Educ 39 (2003), 320-336.

Accepted: 21.01.2016