MATLAB Simulink Modeling and Simulation of Recurrent Neural Network for Solving Linear Programming Problems

Raja Das*

*School of Advanced Sciences, VIT University, Vellore, Tamil Nadu, India

Abstract

In this paper, a recurrent neural network for solving linear programming problems is presented that is simpler, intuitive and fast converging. To achieve optimality in accuracy and also in computational effort, an algorithm is presented. We investigate in this paper the MATLAB Simulink modeling and simulative verification of such a recurrent neural network. Modeling and simulative results substantiate the theoretical analysis and efficacy of the recurrent neural network for solving the linear programming problem. A detailed example has been presented to demonstrate the performance of the recurrent neural network.

Keywords: Linear Programming Problem; Recurrent Neural Network; MATLAB Simulink

1. Introduction

Linear programming techniques are widely used to solve a number of military, economic, industrial, and social problems. In the last 50 years, researchers have proposed various dynamic solvers for solving linear programming problems. The dynamic systems approach to solve constrained optimization problems was first proposed by Pyne [1]. The recent developments in this field have redefined the application of neural network by dynamic solvers [2-4].

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. Linear Programming (LP), one of the oldest disciplines of Operations Research, 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. Now, with high-speed computers and multiprocessors it is possible to construct and solve linear programs that were impossible just a few years ago.


The simulation models are developed as a stand alone application using MATLAB Simulink [8]. Matlab Simulation Models have been widely used [9,10]. The primary objective 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 most important advantages of the neural networks are their massive parallel processing capacity and fast 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, storage capacity, raw material availability, etc. First, the given problem must be presented in linear programming form. This requires defining the variables of the problem, establishing 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 appropriate linear functions to fit the linear programming format.

3. Problem Statement

Consider a standard form of Linear Programming Problems described as

Minimize $C^T X$  \hspace{1cm} (1)  
Subject to $AX = B$  \hspace{1cm} (2)  
$X \geq 0$  \hspace{1cm} (3)

where $X \in \mathbb{R}^n$ is a column vector of decision variables, $C \in \mathbb{R}^n$ and $B \in \mathbb{R}^m$ are column vector of cost coefficients and right hand side parameters, respectively, $A \in \mathbb{R}^{m \times n}$ is a constraint coefficient matrix, and the subscript $T$ 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 theoretically 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 exist 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 feedback 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 node equation for the discrete-time network with n-neurons is given by

$$x_i(k+1) = \begin{cases} x_i(k) - \mu \left[ c_i + \sum a_{ij} y_j(k) \right] & \text{if } x_i(k+1) > 0 \\ 0 & \text{if } x_i(k+1) \leq 0 \end{cases}$$

and

$$y_j(k+1) = y_j(k) + \eta \left[ z_j(k) - a_j y_j(k) \right]$$

where

$$z_j(k) = AX - B$$
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 \( AX = B \) and non negativity constraints \( X \geq 0 \) 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

\[
E(X, Y) = C^T X + Y^T (AX - B) - \alpha Y^T Y
\]

with \( \alpha \geq 0, Y \in \mathbb{R}^{m \times 1} \), 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

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

and

\[
\frac{\partial E}{\partial X} = C + A^T Y = C + A^T (KZ + Y)
\]

where \( Z \in \mathbb{R}^{m \times 1} \) is defined as \( Z = AX - B \)

In a similar manner we have

\[
\frac{\partial E}{\partial Y} = AX - B - \alpha Y
\]

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. Numerical Example

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

Example 1. Consider the linear programming problem

\[
\text{Max } Z = 45x_1 + 80x_2
\]

Subject to

\[
\begin{align*}
5x_1 + 20x_2 & \leq 400 \\
10x_1 + 15x_2 & \leq 450 \\
x_1, x_2 & \geq 0
\end{align*}
\]
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:

$$\text{Max } Z = 45x_{1} + 80x_{2} + 0x_{3} + 0x_{4}$$

Subject to

$$5x_{1} + 20x_{2} + x_{3} = 400$$
$$10x_{1} + 15x_{3} + x_{4} = 450$$
$$x_{1}, x_{2}, x_{3}, x_{4} \geq 0$$

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.

$$X^{+} = [24, 14]^{T}, \text{ which is within the learning rate parameter accuracy from the exact solution } X = [23.34, 14.24]^{T}$$
Table 1 (Comparison of the simulation result with the exact solution of the example)

<table>
<thead>
<tr>
<th>Variables</th>
<th>Neural Network Result</th>
<th>Exact Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1$</td>
<td>23.34</td>
<td>24</td>
</tr>
<tr>
<td>$x_2$</td>
<td>14.24</td>
<td>14</td>
</tr>
</tbody>
</table>

The results are also given in Table 1 and they are very close to the exact solution. The trajectories of the neural network variables are plotted in Fig. 2.

![Trajectories of the independent Variable for Linear Programming Problem](image)

Fig. 2 Trajectories of the independent Variable for linear Programming Problem

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 linear programming problem. It converges to the exact solutions of the Problem. Modeling and simulative results substantiate the theoretical analysis and efficacy of the recurrent neural network for solving the linear programming problem. This model is very simple. The model can be extended to solve other optimization problems, including convex optimization and other nonlinear programming problems.
References