Caspian Journal of Mathematical Sciences (CJMS)
University of Mazandaran, Iran
http://cjms.journals.umz.ac.ir
ISSN: 1735-0611

CJMS. **4**(2)(2015), 205-213

A Recurrent Neural Network Model for Solving Linear Semidefinite Programming

A. Malek [1], S. M. Mirhosseini Alizamini [2] and G. Ahmadi [3]
[1] Department of Applied Mathematics, Faculty of Mathemetical Sciences,Tarbiat Modares University, Tehran, Iran
[2], [3] Department of Mathematics, Payame Noor University (PNU), P.O. Box 19395-3697,Tehran, Iran

Abstract. In this paper we solve a wide rang of Semidefinite Programming (SDP) Problem by using Recurrent Neural Networks (RNNs). SDP is an important numerical tool for analysis and synthesis in systems and control theory. First we reformulate the problem to a linear programming problem, second we reformulate it to a first order system of ordinary differential equations. Then a recurrent neural network model is proposed to compute related primal and dual solutions simultaneously. Illustrative examples are included to demonstrate the validity and applicability of the technique.

Keywords: Semidefinite programming; Primal-dual problems; Recurrent Neural Networks.

2000 Mathematics subject classification: 90C22; Secondary: 92B20.

1. Introduction

A semidefinite programming problem is a generalization of linear programming and has various applications in system and control theory and combinatorial optimization. Due to its many applications in control theory, robust optimization, combinatorial optimization and eigenvalue

[1] Corresponding author: mirhosseini@phd.pnu.ac.ir
Received: 30 January 2013
Revised: 01 November 2015
Accepted: 10 November 2015

optimization, semidefinite programming had been in widespread use. In the resent years development of efficient algorithms brought it into the realm of tractability [1,3,6]. Today it is one of the basic modeling and optimization tools along with linear and quadratic programming. A very good overview of the applications is provided by Guoyin et al. [3]. So far, a significant number of reports has been devoted to generalizing the interior point method to semidefinite programming [8]. Yashita et al. [5], Ghami et al. [2], presented primal-dual interior algorithms for semidefinite programming. This paper presents a new recurrent neural network for solving linear semidefinite programming problems. The mentioned model is simpler and more intuitive than existing models and converges very fast to the exact primal and dual solutions. The model is based on a nonlinear dynamical system and has an interesting economic interpretation. Frankly, we concentrate here on generalizing the primal-dual method and neural network model of linear programming towards semidefinite programming [4,7].

The paper is organized as follows. In Section 2, we review some basic notations from linear algebra and fundamental properties of the cone of positive semidefinite matrices. Semidefinite programs and their duals are introduced in this section. In Section 3, we define the network dynamics of the new method for primal-dual problems. In Section 4, the numerical examples are simulated to show the reasonableness of our theory and demonstrate the performance of our network. Finally, we end this paper with conclusions in Section 5.

2. Preliminaries

2-1. Semidefinite programming

In this section, we present some notations and preliminary lemmas that will be used in the proofs of the main results.

Let $\mathcal{S}^n$ denote the vector space of real symmetric $n \times n$ matrices. Denote the dimension of this space by [3]

$$n^{\bar{2}} = \frac{n(n+1)}{2}.$$

The standard inner product on $\mathcal{S}^n$ is

$$A \bullet B = trace(AB) = \sum_{i,j} A_{ij} B_{ij}.$$

By $X \succeq 0$ ($X \succ 0$), where $X \in \mathcal{S}^n$, we mean that X is positive semidefinite (positive definite). It is well known that

$$X \in \mathcal{S}^n,\ X \succeq 0, \quad if \quad h^T X h \geq 0,\ \forall h \in \mathbb{R}^n.$$

The Primal Semidefinite Programming (PSDP) problem is [3]

$$
\begin{aligned}
(PSDP) \quad & min \quad C \bullet X \\
& s.t \quad A_i \bullet X = b_i, \quad i = 1, \ldots, m \\
& \qquad X \succeq 0, \ X \in \mathcal{S}^n,
\end{aligned} \tag{2.1}
$$

where $C \in \mathcal{S}^n, \ X \in \mathcal{S}^n, \ A_i \in \mathcal{S}^n, \ i = 1, \ldots, m, and\, b = [b_1, b_2, \ldots, b_m] \in \mathbb{R}^m$.

The dual problem for PSDP is in the form

$$
\begin{aligned}
(DSDP) \quad & max \quad \sum_{i=1}^{m} y_i b_i \\
& s.t \quad \sum_{i=1}^{m} y_i A_i + Z = C, \\
& \qquad Z \succeq 0, \ Z \in \mathcal{S}^n,
\end{aligned} \tag{2.2}
$$

where y is known as multiplier and Z is the dual slack variable.

Lemma 2.1. There exists a primal feasible point $X \succ 0$, and a dual feasible point (y, Z) with $Z \succ 0$.

Lemma 2.2. The matrices $A_i, i = 1, \ldots, m$, are linearly independent, i.e. they span an m-dimensional linear space in $\mathcal{S}^n$.

Theorem 2.3. (Weak Duality) If X and (y,Z) are feasible in PSDP and DSDP problems, respectively, then

$$C \bullet X - b^T y = X \bullet Z \geq 0.$$

Proof : We find

$$C \bullet X - b^T y = (\sum_{i=1}^{m} y_i A_i + Z) \bullet X - b^T y = \sum_{i=1}^{m} (A_i \bullet X) y_i + Z \bullet X - b^T y = X \bullet Z$$

Moreover, since X is positive semidefinite, it has a square root $X^{\frac{1}{2}}$, thus

$$X \bullet Z = trace(XZ) = trace(X^{\frac{1}{2}} X^{\frac{1}{2}} Z) = trace(X^{\frac{1}{2}} Z X^{\frac{1}{2}}) \geq 0.$$

■

Using (2.1) and (2.2), first we transform the problem PSDP into a linear programming (P) as follows:

$$(P) \quad min \quad \sum_{k=1}^{2n} c_k x_k \tag{2.3}$$
$$s.t \quad \sum_{k=1}^{2n} a_{ik} x_k = b_i, \quad i = 1, \ldots, m$$

where

$$c_k = \{c_{ii}, k = 1, \ldots, n \ ; \ 2c_{ij}, k = n+1, \ldots, 2n\},$$
$$x_k = \{x_{ii}, k = 1, \ldots, n \ ; \ x_{ij}, k = n+1, \ldots, 2n\},$$
$$a_{ik} = \{a_{ii}, k = 1, \ldots, n \ ; \ 2a_{ij}, k = n+1, \ldots, 2n\},$$

$for \ \ i = 1, \ldots, n, j = 1, \ldots, n, \ i < j.$

The dual problem of problem (P) is as follows:

$$(D) \quad max \quad \sum_{i=1}^{m} y_i b_i \tag{2.4}$$
$$s.t \quad \sum_{k=1}^{2n} y_i a_{ik} \leq c_k$$

Consider the following constrained primal and dual linear programming problems associated with the problems (2.3) and (2.4), repectively

$$(P) \quad min \quad z(\mathcal{X}) = \mathcal{C}^T \mathcal{X} \tag{2.5}$$
$$s.t. \quad \mathcal{A}\mathcal{X} = b,$$

the dual formulation is

$$(D) \quad max \quad h(\mathcal{Y}) = b^t \mathcal{Y} \tag{2.6}$$
$$s.t \quad \mathcal{A}^T \mathcal{Y} \leq \mathcal{C}.$$

3. Primal-dual solution for the SDP using RNN

In this section we use the penalty function method to construct a recurrent neural network based on Yashtini and Malek model for linear programming [7]. The penalty function method is a popular technique for optimization in which it is used to construct a single unconstrained problem or a sequence of unconstrained problems. By applying this approach to optimization problem (P), we have the following unconstrained optimization problem

$$\min_{\mathcal{X}\in\mathcal{F}} E(\mathcal{X},\mathcal{Y}) = \mathcal{C}^T\mathcal{X} + \mathcal{Y}^T(\mathcal{A}\mathcal{X} - b),$$

where $\mathcal{F}$ is the feasible set of (P).

Often, $E(\mathcal{X},\mathcal{Y})$ is called the energy function, where the decision variables in (P) and (D) become state variables in the energy function. They are actually time-dependent, i.e., $\mathcal{X} = \mathcal{X}(t)$ and $\mathcal{Y} = \mathcal{Y}(t)$, $t \geq 0$. Now minimization of the energy reads to

$$\min_{\mathcal{X}(t)\in\mathcal{F}} E(\mathcal{X}(t),\mathcal{Y}(t)) = \mathcal{C}^T\mathcal{X}(t) + \mathcal{Y}(t)^T(\mathcal{A}\mathcal{X}(t) - b).$$

To solve the problem (P), let us define the dynamics of the proposed neural network to be [7]

$$\begin{cases} \frac{d\mathcal{X}}{dt} = \mathcal{C} - \mathcal{A}^T\mathcal{Y}(t), \\ \\ \frac{d\mathcal{Y}}{dt} = \mathcal{A}\mathcal{X}(t) - b, \end{cases} \tag{3.1}$$

where $\mathcal{X}(t) \geq 0$, $\mathcal{Y}(t) \geq 0$, $t \geq 0$ and $(\mathcal{X}(t),\mathcal{Y}(t))^T$ is a state vector. The solution of (3.1) exists and is unique for some given initial conditions.

Theorem 3.1. If problem (P) has an optimal solution, the equilibrium point of dynamical system (3.1) equals with the optimal solution of (P).

Proof : Suppose problem (P) has an optimal solution with optimal value z^*. Then, the objective function $z(\mathcal{X})$ is bounded below over the feasible region $\mathcal{F}$ by z^*, i.e.

$$z(\mathcal{X}) = \mathcal{C}^T\mathcal{X} \geq z^*, \qquad \forall\ \mathcal{X} \in \mathcal{F},$$

and equality holds when $\mathcal{X}$ is the optimal solution of problem (P). By attention to the notes, mentioned at the beginning of this section, we have

$$E(\mathcal{X}(t),\mathcal{Y}(t)) = \mathcal{C}^T\mathcal{X}(t) + \mathcal{Y}(t)^T(\mathcal{A}\mathcal{X}(t) - b) \geq \mathcal{C}^T\mathcal{X}(t) = z(\mathcal{X}(t)) \geq z^*,$$

for all $t \geq 0$ and arbitrary $\mathcal{X}(0) \in \mathbb{R}^n$. Equality holds when $\mathcal{X}(t)$ is the optimal solution of problem (P). Therefore, if (P) has an optimal solution then, this optimal solution is a minimizer for $E(\mathcal{X}(t),\mathcal{Y}(t))$.

Since the problem is linear, $E(\mathcal{X}(t),\mathcal{Y}(t))$ is convex. Thus, a sufficient and necessary condition for optimal is: $\mathcal{X}^*$ is a minimizer of $E(\mathcal{X}(t),\mathcal{Y}(t))$, if and only if, $\nabla_{\mathcal{X}(t)}E(\mathcal{X}(t),\mathcal{Y}(t)) = 0$. This result shows that $\mathcal{X}^*$ is a solution for dynamical system (3.1). Therefore, $\mathcal{X}^*$ is an equilibrium point of (3.1). ■

Based on dynamical system (3.1), we propose the following recurrent neural network model to solve SDP:

$$\frac{d\mathcal{X}}{dt} = \mathcal{C} - A^T(\mathcal{Y} + k\frac{d\mathcal{Y}}{dt}), \qquad \mathcal{X} \geq 0, \tag{3.2}$$

$$\frac{d\mathcal{Y}}{dt} = A(\mathcal{X} + k\frac{d\mathcal{X}}{dt}) - b, \qquad \mathcal{Y} \geq 0. \tag{3.3}$$

Coefficient k is a positive constant [7]. The main property of this system is stated in the following theorem.

Theorem 3.2. If the neural network whose dynamics is described by the differential equations (3.2) and (3.3) converges to a stable state, then this solution will be the optimal solutions for the PSDP (2.1) and its dual (2.2).

Proof : Let$\mathcal{X}_i$ be the ith element of $\mathcal{X}$. Equation (3.2) can be rewritten as:

$$\frac{d\mathcal{X}_i}{dt} = [\mathcal{C} - A^T(\mathcal{Y} + k\frac{d\mathcal{Y}}{dt})]_i \ , \ if \ \mathcal{X}_i > 0 \ , \ \forall i \tag{3.4}$$

$$\frac{d\mathcal{X}_i}{dt} = max\{[\mathcal{C} - A^T(\mathcal{Y} + k\frac{d\mathcal{Y}}{dt})]_i, 0\} \ , \ if \ \mathcal{X}_i = 0 \ , \ \forall i \tag{3.5}$$

Condition (3.5) is to ensure that $\mathcal{X}$ will be bounded from below by zero.
Let $\mathcal{X}^*$, $\mathcal{Y}^*$ be limits of $\mathcal{X}$ and $\mathcal{Y}$ respectively, that is

$$\lim_{t\to\infty} \mathcal{X}(t) = \mathcal{X}^*$$

$$\lim_{t\to\infty} \mathcal{Y}(t) = \mathcal{Y}^*$$

By stability of convergence, we have $\frac{d\mathcal{X}^*}{dt} = 0$ and $\frac{d\mathcal{Y}^*}{dt} = 0$. Equations (3.4) and (3.5) then become:

$$0 = [\mathcal{C} - A^T\mathcal{Y}^*]_i \ , \ if \ \mathcal{X}_i^* > 0 \tag{3.6}$$

$$0 = max\{[\mathcal{C} - A^T\mathcal{Y}^*]_i, 0\} \ , \ if \ \mathcal{X}_i^* = 0 \tag{3.7}$$

In other words:

$$[\mathcal{C} - A^T\mathcal{Y}^*]_i = 0 \ , \ if \ \mathcal{X}_i^* > 0 \tag{3.8}$$

$$[\mathcal{C} - A^T\mathcal{Y}^*]_i \leq 0 \ , \ if \ \mathcal{X}_i^* = 0 \tag{3.9}$$

Or:

$$\mathcal{C} - A^T\mathcal{Y}^* \leq 0 \ , \ \forall i \tag{3.10}$$

Similarly, taking the limit of (3.3) we will have:

$$A\mathcal{X}^* - b \leq 0 \ , \tag{3.11}$$

Equations (3.10) and (3.11) show that $\mathcal{X}^*$ and $\mathcal{Y}^*$ are the feasible solutions for the problems (2.1) and (2.2).
Furthermore, from (3.8) and (3.9) we have:

$$\mathcal{X}_i^*[\mathcal{C} - A^T\mathcal{Y}^*]_i = 0 \ , \ \forall i \tag{3.12}$$

or in vector form:

$$\mathcal{C}^T\mathcal{X}^* - \mathcal{X}^*A^T\mathcal{Y}^* = 0 \ , \tag{3.13}$$

Similarly, form (2.17) we can write

$$\mathcal{X}^*A^T\mathcal{Y}^* - b^T\mathcal{Y}_j^* = 0 \ , \ \forall j \tag{3.14}$$

Thus, (2.19) and (2.20):

$$\mathcal{C}^T\mathcal{X}^* = b^T\mathcal{Y}^* \ , \tag{3.15}$$

By the DSDP Duality theory, from (3.15) and the feasibility of $\mathcal{X}^*$ and $\mathcal{Y}^*$, we can conclude that $\mathcal{X}^*$ and $\mathcal{Y}^*$ are the optimal solutions for the PSDP and DSDP problems (2.1) and (2.2). ■

4. Numerical examples

In following illustrative examples are solved to demonstrate the effectiveness of the proposed recurrent neural network model. The software MATLAB 7.10.0 is used to make this solutions.

Example 4.1. Consider the following SDP problem:

let $n = 3$ and $m = 3$,

$$A_1 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, A_2 = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix}, A_3 = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix},$$

$b^T = (1, 0, 0).$

Consider an objective function matrices C, as follows

$$C = \begin{pmatrix} 2 & 2 & 2 \\ 2 & 3 & -2 \\ 2 & -2 & 3 \end{pmatrix}$$

Then the primal and dual optimal solutions, using neural network model are:

$$X = \begin{pmatrix} 1 & -2 & -2 \\ -2 & 4 & 4 \\ -2 & 4 & 4 \end{pmatrix}, y = (6, 0, 0), Z = \begin{pmatrix} 8 & 2 & 2 \\ 2 & 3 & -2 \\ 2 & -2 & 3 \end{pmatrix},$$

It can be checked that the primal-dual solutions are equal to the exact solutions.

Example 4.2. Consider the SDP problem described by:

let $n = 5$ and $m = 5$,

$$A_1 = \begin{pmatrix} 2 & 1 & -1 & 3 & 6 \\ 0 & -1 & 2 & -4 & 0 \\ 5 & 2 & -5 & 1 & -2 \\ 3 & 0 & 0 & 1 & -2 \\ 1 & -6 & 0 & 1 & -4 \end{pmatrix}, A_2 = \begin{pmatrix} -3 & 0 & 1 & 4 & -2 \\ 0 & 7 & -3 & 8 & 2 \\ 1 & 0 & -3 & -2 & -5 \\ -1 & 0 & 2 & -3 & 0 \\ -4 & 0 & 8 & -1 & 7 \end{pmatrix},$$

$$b^T = (4, 3, 4, -5, -5),$$

and

$$C = \begin{pmatrix} 2 & 2 & 2 & -1 & 1 \\ 2 & 3 & -2 & 0 & 2 \\ 2 & -2 & 3 & 2 & 0 \\ 0 & 0 & -2 & -3 & 0 \\ -1 & 3 & 0 & 1 & 2 \end{pmatrix}.$$

We can obtain directly the components of the solution X, Z and y.

$$X = \begin{pmatrix} 1 & -2 & -2 & 4 & 6 \\ -2 & 4 & 4 & 1 & 12 \\ -2 & 4 & 4 & -4 & 0 \\ 1 & 0 & 4 & 0 & 0 \\ 3 & 4 & 5 & 0 & 2 \end{pmatrix}, Z = \begin{pmatrix} 8 & 2 & 2 & 13 & 4 \\ 2 & 3 & -2 & 8 & 3 \\ 2 & -2 & 3 & 6 & 10 \\ 11 & 0 & -4 & 1 & -8 \\ 0 & 3 & -6 & 1 & -4 \end{pmatrix},$$

$$y = (4, 0, 6, 1, 7).$$

In this example, it can be checked that the primal-dual solutions are equal to the numerical results obtained by Matlab toolbox for solving linear programming which used the Runge-Kutta triple BS (2,3) method.

5. Conclusions

In this paper, we have proposed a recurrent neural network approach to linear semidefinite programming. Based on a primal-dual reformulation of the problem. It is shown that there exists a primal-dual transformation between the (PSDP)-(DSDP) problems and the proposed neural network model.

ACKNOWLEDGEMENTS

Authors would like to thank from the anonymous referees for their valuable comments to improve the earlier version of this paper.

REFERENCES

[1] V. Blanco, J. Puerto, S. Ben, A semidefinite programming approach for solving multiobjective linear programming, J Glob Optim. 58 (2014) 465-480.

[2] M. E. Ghamia, C. Roosb, T. Steihauga, A generic primal-dual interior point method for semidefinite optimization based on a new class of kernel functions, Opt. Methods Soft. 25(3) (2010) 387-403.

[3] L. Guoyin, M. Alfred Ka Chun, KP. Ting, Robust least square semidefinite programming with applications, Comput Optim Appl. 58 (2014) 347-379.

[4] A. Malek, G. Ahmadi and S. M. Mirhosseini Alizamini, Solving linear semi-infinite programming problems using recurrent neural networks, Opt. Methods and Soft, Submitted.

[5] H. Yamashita, H. Yabe, K. Harada, A primal dual interior point method for nonlinear semidefinite programming, Comput Optim Appl. 135 (2012) 89-121.

[6] L. Yang, B. Yu, A homotopy method for nonlinear semidefinite programming, Comput Optim Appl. 56 (2013) 81-96.

[7] M. Yashtini and A. Malek, A discrete-time neural network for solving nonlinear convex problems with hybrid constraints, Appl. Math. Comput. 195 (2008) 576-584.

[8] V. G. Zhadan, A. A. Orlov, Dual interior point methods for linear semidefinite programming problems, Comput. Math. Phy. 51(12) (2011) 2031-2051.