A Recurrent Neural Network Model for Solving Linear Semidefinite Programming

A. Malek, S. M. Mirhosseini Alizamini and G. Ahmadi

1 Department of Applied Mathematics, Faculty of Mathematical 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 range 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.


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 recent 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 semidefi-
nite 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 inter-
pretation. Frankly, we concentrate here on generalizing the primal-dual
method and neural network model of linear programming towards semi-
definite 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]

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

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

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

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

$$X \in \mathcal{S}^n, X \succeq 0, \text{ if } h^T X h \geq 0, \forall h \in \mathbb{R}^n.$$
The Primal Semidefinite Programming (PSDP) problem is [3]

\[
\begin{align*}
(PSDP) & \quad \min & \quad C \cdot X \\
& \text{s.t} & \quad A_i \cdot X = b_i, \quad i = 1, \ldots, m \\
& & \quad X \succeq 0, \; X \in \mathcal{S}^n,
\end{align*}
\]

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

\[
\begin{align*}
(DSDP) & \quad \max & \quad \sum_{i=1}^m y_i b_i \\
& \text{s.t} & \quad \sum_{i=1}^m y_i A_i + Z = C, \\
& & \quad Z \succeq 0, \; Z \in \mathcal{S}^n,
\end{align*}
\]

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

\[ C \cdot X - b^T y = X \cdot Z \geq 0. \]

**Proof:** We find

\[
C \cdot X - b^T y = \left( \sum_{i=1}^m y_i A_i + Z \right) \cdot X - b^T y = \sum_{i=1}^m (A_i \cdot X) y_i + Z \cdot X - b^T y = X \cdot Z
\]

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

\[ X \cdot Z = \text{trace}(XZ) = \text{trace}(X^{\frac{1}{2}}X^{\frac{1}{2}}Z) = \text{trace}(X^{\frac{1}{2}}ZX^{\frac{1}{2}}) \geq 0. \]
Using (2.1) and (2.2), first we transform the problem PSDP into a linear programming (P) as follows:

\[
\begin{align*}
(P) \quad & \min \sum_{k=1}^{2n} c_k x_k \\
& \text{s.t} \sum_{k=1}^{2n} a_{ik} x_k = b_i, \quad i = 1, \ldots, m
\end{align*}
\]

where

\[
\begin{align*}
c_k &= \{c_{ii}, k = 1, \ldots, n; \quad 2c_{ij}, k = n + 1, \ldots, 2n\}, \\
x_k &= \{x_{ii}, k = 1, \ldots, n; \quad x_{ij}, k = n + 1, \ldots, 2n\}, \\
a_{ik} &= \{a_{ii}, k = 1, \ldots, n; \quad 2a_{ij}, k = n + 1, \ldots, 2n\},
\end{align*}
\]

for \( i = 1, \ldots, n, j = 1, \ldots, n, \ i < j. \)

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

\[
\begin{align*}
(D) \quad & \max \sum_{i=1}^{m} y_i b_i \\
& \text{s.t} \sum_{k=1}^{2n} y_i a_{ik} \leq c_k
\end{align*}
\]

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

\[
\begin{align*}
(P) \quad & \min \ z(\mathcal{X}) = C^T \mathcal{X} \\
& \text{s.t} \ A\mathcal{X} = b,
\end{align*}
\]

the dual formulation is

\[
\begin{align*}
(D) \quad & \max \ h(\mathcal{Y}) = b^T \mathcal{Y} \\
& \text{s.t} \ A^T \mathcal{Y} \leq C.
\end{align*}
\]

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

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

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

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

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

\[
\begin{align*}
\frac{d\mathcal{X}}{dt} &= \mathbf{C} - \mathbf{A}^T \mathcal{Y}(t), \\
\frac{d\mathcal{Y}}{dt} &= \mathbf{A} \mathcal{X}(t) - b,
\end{align*}
\]

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.

\[
z(\mathcal{X}) = \mathbf{C}^T \mathcal{X} \geq z^*, \quad \forall \mathcal{X} \in \mathcal{F},
\]

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

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

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:

\[
\frac{dX}{dt} = C - A^T(Y + k\frac{dY}{dt}), \quad X \geq 0, \quad (3.2)
\]

\[
\frac{dY}{dt} = A(X + k\frac{dX}{dt}) - b, \quad Y \geq 0. \quad (3.3)
\]

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 \(X_i\) be the \(i\)th element of \(X\). Equation (3.2) can be rewritten as:

\[
\frac{dX_i}{dt} = [C - A^T(Y + k\frac{dY}{dt})]_i, \quad \text{if} \quad X_i > 0, \quad \forall i \quad (3.4)
\]

\[
\frac{dX_i}{dt} = \max\{[C - A^T(Y + k\frac{dY}{dt})]_i, 0\}, \quad \text{if} \quad X_i = 0, \quad \forall i \quad (3.5)
\]

Condition (3.5) is to ensure that \(X\) will be bounded from below by zero.

Let \(X^*\), \(Y^*\) be limits of \(X\) and \(Y\) respectively, that is

\[
\lim_{t \to \infty} X(t) = X^* \quad \lim_{t \to \infty} Y(t) = Y^*
\]

By stability of convergence, we have \(\frac{dX^*}{dt} = 0\) and \(\frac{dY^*}{dt} = 0\). Equations (3.4) and (3.5) then become:

\[
0 = [C - A^T Y^*]_i, \quad \text{if} \quad X_i^* > 0 \quad (3.6)
\]

\[
0 = \max\{[C - A^T Y^*]_i, 0\}, \quad \text{if} \quad X_i^* = 0 \quad (3.7)
\]

In other words:

\[
[C - A^T Y^*]_i = 0, \quad \text{if} \quad X_i^* > 0 \quad (3.8)
\]

\[
[C - A^T Y^*]_i \leq 0, \quad \text{if} \quad X_i^* = 0 \quad (3.9)
\]

Or:

\[
C - A^T Y^* \leq 0, \quad \forall i \quad (3.10)
\]
Similarly, taking the limit of (3.3) we will have:

\[ A\mathbf{x}^* - b \leq 0 , \]  

Equations (3.10) and (3.11) show that \( \mathbf{x}^* \) and \( \mathbf{y}^* \) are the feasible solutions for the problems (2.1) and (2.2).

Furthermore, from (3.8) and (3.9) we have:

\[ \mathbf{X}^*_i[C - A^T\mathbf{Y}^*]_i = 0 , \quad \forall i \]  

or in vector form:

\[ C^T\mathbf{x}^* - \mathbf{x}^* A^T\mathbf{y}^* = 0 , \]  

Similarly, form (2.17) we can write

\[ \mathbf{x}^* A^T\mathbf{y}^* - b^T\mathbf{y}^*_j = 0 , \quad \forall j \]  

Thus, (2.19) and (2.20):

\[ C^T\mathbf{x}^* = b^T\mathbf{y}^* , \]  

By the DSDP Duality theory, from (3.15) and the feasibility of \( \mathbf{x}^* \) and \( \mathbf{y}^* \), we can conclude that \( \mathbf{x}^* \) and \( \mathbf{y}^* \) 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 \( m = 3 \),

\[
A_1 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad A_2 = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix}, \quad A_3 = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix},
\]

\[ b^T = (1, 0, 0). \]

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 \( m = 5 \),

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

\[
b^T = (4, 3, 4, -5, -5), \]

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 \( y \).

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

\[
y = (4, 0, 6, 1, 7). \]

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


