One-layer Neural Network for Solving Least Absolute Deviation Problems with Box and Equality Constraints

Cuiping Li
Shaanxi Normal University
China
cuipli@126.com

ABSTRACT: In this paper, I design a new neural network for solving least absolute deviation problems with equality and box constraints. Compared with some existing models, the proposed neural network needs the fewest state variables and has only one-layer structure. By constructing a proper Lyapunov function, the following two results can be proved. First, it is stable in the sense of Lyapunov. Second, the solution of the proposed model can converge to an exact optimal solution of the problem. Finally, its validity and transient behaviors are demonstrated by some simulation results.

Keywords: Least Absolute Deviation, Neural Network, One-layer, Lyapunov Stable

Received: 3 December 2017, Revised 27 January 2018, Accepted 7 February 2018

DOI: 10.6025/jes/2018/8/2/57-71

© 2018 DLINE. All Rights Reserved

1. Introduction

Consider the following least absolute deviation problem:

\[
\begin{aligned}
\min & \quad \|Ax - b\|_1 \\
\text{s.t} & \quad Cx = d, \ x \in X
\end{aligned}
\] (1)

where \( x = (x_1, x_2, \ldots, x_n)^T \in \mathbb{R}^n \) is the decision vector, \( A \in \mathbb{R}^{m \times n}, C \in \mathbb{R}^{p \times n}, b \in \mathbb{R}^m, d \in \mathbb{R}^p, \text{rank}(C) = p < n \) are given parameters. \( X = \{x \in \mathbb{R}^n | l < x < h\} \) is a nonempty box set, where \( l, h \in \mathbb{R}^n (1 < h) \) and some components of \( l \) or \( h \) can be \(-\infty \) or \( \infty \), suitably.

As we know, (1) is called a least absolute deviation problem (see [5-7, 9, 10, 12, 27]), which often encountered in signal processing as an \( l_1 \)-norm optimization problem.
In there, $x$ denotes the vector to be estimated, $A$ stands for the model matrix, $b$ is the vector of observation, the noise and errors contained in $b$ is dened as $\|Ax - b\|_1$, where $\|\cdot\|_1$ denotes $l_1$-norm in this paper. $Cx = d$ and $x \in X$ are constraints where we try to estimate $x$, conforming to min $\|Ax - b\|_1$.

Due to their excellent properties, especially the sparsity properties of the solution, least absolute deviation problems have been extensively studied by many scholars in the past decades. In the early stage, many numerical analysis methods were proposed for solving them have achieved good results (see [14-16]), while they maybe nor satisfy the requirement of real-time. Because the times they consumed in computing a solution is mainly dependent on the dimension and architecture of the problem, and the complexity of the algorithm they applied. Thus, in recent twenty years, a large number of neural network models which can be implemented by VLSI in parallel and competent for real time applications [32], have been designed and can be used to least absolute deviation problems (see [5-13, 19, 27, 30-32]) after [17] and [18]. Since least absolute deviation problems can be rewritten as minimax problems, models in [21-23, 25, 26, 28, 29, 33] can also be applied to these problems.

For the models presented in the above References, the neural network in [31] can be used to solve (1) and requires only $n + p$ state variables, but it is discontinuous. The models in [9, 30, 32] were designed to solve the min $\|x\|_1$ problem, the first two include equality constraints and the third has equality and box constraints. They are two special cases of (1), thus cannot be used to (1) if $A \neq I_n, b \neq 0$ or $X$ is not empty, where $I_n$ is a identity matrix of order $n$ in this paper. The models proposed by Xia at al. and Wang at al. in [5, 6, 8] can solve (1) without any constraints, thus they cannot be used to solve (1) when $C \neq 0$ or $X$ is not empty. After that, [22], [23] and [10] present three models can be solve (1) with only box constraints. Even though these models have good stability performances, they are hard to solve (1) unless $C$ is invertible. Furthermore, if $C$ is invertible, there will be only one solution for (1) which can be resulted from $Cx = d$ and adjusted by $X$. The model in [7] can be applied to (1), while it requires $m + 3n + 2p$ state variables and has two-layer structure (see (13) in Section 2). Furthermore, its solution may be feasible but not optimal in some case. (see [27, 32] and Example 1 in Section 3). The model proposed by Xue and Bian in [28] can be used to (1) which has only one-layer structure, but it needs $2(m + n) + p$ neurons. The neural networks in [12, 26, 27] designed by Xia and Hu at al. respectively, can be applied to (1). The resulting models are shown as (14), (15) and (16) in Section 2. We can see that they all require $m + n + p$ state variables, and have more than one layer structure. Scholars Liu and Wang [29] constructed one neural network to solve minimax problems which for (1) in detail is (17) in Section 2. Even though this model needs only $m + n$ state variables, it is a two-layer neural network and its stability needs to find a parameter larger than 1 to guarantee the establishment of the linear matrix inequality (see (17) in Section 2). Actually, that parameter maybe nor exist in same case(see [33] and Example 1 in Section 4). Recently, the model in [33] for (1) needs only $m + n$ state variables, but has three-layer structure. Thus, to find a new neural network for (1) which requires few state variables, and has good performances it is necessary.

Considered the above cases, this paper presents a new neural network for (1). The proposed neural network is stable in the sense of Lyapunov. For any initial point, the solution of this model converges to an optimal solution of (1). Compared with neural networks in being for (1), the proposed one needs the fewest neurons and has only one layer structure. Some numerical examples are introduced to illustrate its performances.

In this paper, the existing of a point $x^*$ which satisfies the Slater condition [20] and is an optimal solution of (1) is assumed. And, $M^T$ is the transpose of $M$ whether it is a matrix or a vector. $P_C$, defined as $P_C(\omega) = \arg\min_{\zeta \in U} \|\omega - \zeta\|_1$, is a projection operator which project on a closed convex set $U \in R^n$, and its two basic natures [4] are

$$
\begin{align*}
\|P_C(\omega) - P_C(\omega')\| &\leq \|\omega - \omega'\|, \quad \forall \omega, \omega' \in R^n, \\
[\omega - P_C(\omega)]^T[P_C(\omega) - \zeta] &\leq 0, \quad \forall \omega \in R^n, \zeta \in U
\end{align*}
$$

A model is said to be stable in the sense of Lyapunov and globally asymptotically stable if the corresponding dynamical system is so (see [21, 24]).

The remainder of this paper is organized as follows. Section 2 includes three parts: A. constructed a projection neural network for (1); B. compared with existing models; C. analyzed the stability of the proposed neural network. Numerical simulations are shown in Sections 3. Sections 4 concludes this paper.

2. Model and Stability
This section includes three subsections. First, a new model for (1) will be built. Then, we will compare it with six existing neural networks. Finally, its stability and convergence will be studied.

2.1 Proposed Model
The following results is necessary for us to design a new neural network.

**Lemma 1.** $x^*$ is an optimal solution of (1) if and only if there exists a $(y^*, s^*) \in Y \times R_p$ such that

\[
\begin{align*}
x^* &= P_X(x^* - A^Ty^* + C^Ts^*) \\
y^* &= P_Y(y^* + Ax^* - b), \quad Cx^* = d,
\end{align*}
\]

where $Y = \{y \in R^m | -1 < y_i < 1; i = 1, 2, \ldots, m\}$.

**Proof.** Obviously, (1) is equivalent to the following optimization problem

\[
\begin{align*}
\min_{x \in X} \max_{y \in Y} y^T(Ax - b) \\
\text{s.t. } Cx = d, \ x \in X
\end{align*}
\]

where $Y = \{y \in R^m | -1 < y_i < 1; i = 1, 2, \ldots, m\}$. The Lagrange function for (4) is

\[
L(x, y, s) = y^T(Ax - b) - s^T(Cx - d)
\]

which is defined on $\Gamma = X \times Y \times R^p$. From the saddle point theorem in [3], $x^* \in R^p$ is an optimal solution of (4) if and only if there exists a $(y^*, s^*) \in R^{m+p}$ such that $(x^*, y^*, s^*)$ is a saddle point of $L(x, y, s)$ on $\Gamma$, i.e.,

\[
L(x^*, y, s) < L(x^*, y^*, s^*) < L(x, y^*, s^*), \ \forall (x, y, s) \in \Gamma
\]

From the left part of (5), the following results can be obtained

\[
\begin{align*}
(y - y^*)^T(b - Ax^*) > 0, \ \forall y \in Y \\
(s - s^*)^T(Cx^* - d) > 0, \ \forall s \in R^p
\end{align*}
\]

As everyone knows the first inequality of the above is called a variational inequality, and there exist a point $y^*$ satisfies it if and only if $y^* = P_Y(y^* + Ax^* - b)$ (see [2]). Then

\[
\begin{align*}
y^* &= P_Y(y^* + Ax^* - b) \\
Cx^* &= d
\end{align*}
\]

From the right part of (5), the next result holds true

\[
(x - x^*)^T(A^Ty^* - C^Ts^*) > 0 \ \forall x \in X.
\]

For the principle of variational inequality, the above inequality can be changed into

\[
x^* = (P_X(x^* - A^Ty^* + C^Ts*)).
\]

Hence (3) holds true.

Lemma 1 indicates that the optimal solution of (1) can be obtained by solving (3).
Thus, to construct a new model for (1), the following variables are introduced:

\[
\begin{align*}
    u^* &= x^* + A^T y^* - C^T x^*, \\
    v^* &= y^* - A x^* + b.
\end{align*}
\]  

(6)

By substituting \( y^* = v^* + A x^* - b \) into the first equality of (6), we obtain

\[ x^* = H(u^* + A^T b - A^T v^* + C^T x^*) \]  

(7)

where \( H = (I_n + A^T A)^{-1} \) is symmetric and positive definite as we know. This and \( C x^* = d \) imply that

\[ C H^* = (u^* + A^T b - A^T v^* + C^T x^*) = d. \]

Since \( \text{rank} (C) = p < n \), \( C C^T \) is nonsingular and \( (C C^T)^T = C C^T \). This and the Symmetric positive definiteness of \( H \) simply \( C H C^T \) is also symmetric and positive definite.

Then

\[ s^* = (C H C^T)^{-1} \left[ d - C H (u^* + A^T b - A^T v^*) \right] \]

By substituting this equality into (7), we have

\[ x^* = \tilde{H} u^* - \tilde{H} A^T v^* + \tilde{d}. \]

Then \( y^* = v^* + A(\tilde{H} u^* - \tilde{H} A^T v^* + \tilde{d}) - b \). By substituting them and \( x^* \) into (3), we can obtain the following systems

\[
\begin{align*}
    \tilde{H} u^* - \tilde{H} A^T v^* + \tilde{d} &= P_x[2(\tilde{H} u^* - \tilde{H} A^T v^* + \tilde{d}) - u^*], \\
    v^* + A(\tilde{H} u^* - \tilde{H} A^T v^* + \tilde{d}) - b &= P_y[2(v^* + A(\tilde{H} u^* - \tilde{H} A^T v^* + \tilde{d}) - b) - v^*].
\end{align*}
\]

(9)

The following lemma will reveal the relationship between solutions of (1) and (9).

**Lemma 2.** Let \( \mathcal{K}^* = \{ \eta = (u^T, v^T)^T \in R^{m+n} | \eta \text{ solves } (9) \} \), then

i) If \( x \) is an optimal solution of (1), then there exist a \( \eta \in R^{m+n} \) such that \( \eta \in \mathcal{K}^* \) with \( x = \tilde{H} u - \tilde{H} A^T v + \tilde{d} \).

ii) If \( \eta \in \mathcal{K}^* \), then \( \tilde{H} u - \tilde{H} A^T v + \tilde{d} \) is an optimal solution of (1), where \( \tilde{H} \) and \( \tilde{d} \) are defined in (8).

**Proof.** From the above analysis i) holds true, clearly.

ii) Let \( x = \tilde{H} u - \tilde{H} A^T v + \tilde{d}, \) then

\[
\begin{align*}
    x &= P_x[2x - u], \\
    v + Ax - b &= P_y[2(v + Ax - b) - v].
\end{align*}
\]

(10)

from (9), and \( C x = d \) since \( C \tilde{H} = 0 \) and \( C \tilde{d} = d \). On the other hand, from (8), we have
That is

\[ u^* = x^* + A^T y^* + C^T s^* \]

This, (9) and \( y^* = v^* + Ax^* - b \), imply that (3) holds. The proof is completed.

Lemma 2 indicates that the optimal solution of (1) can be gotten by solving (9), and vice versa. Let \( \eta = (u^T, v^T)^T \), then a neural-network model to solve (1) can be defined as follows:

- **State equation**
  \[
  \frac{d}{dt} \begin{pmatrix} u \\ v \end{pmatrix} = -\rho \begin{pmatrix} x - P_X(2x - u) \\ y - P_Y(2y - v) \end{pmatrix}
  \]
  (11)

- **Output equation**
  \[
  x = \tilde{H} u - \tilde{H} A^T v + \tilde{d},
  \]
  (12)

where \( y = v + Ax - b \), \( \tilde{H} \) and \( \tilde{d} \) are defined in (8), \( \rho > 0 \) is a scaling constant.

Since \( P_{12} \) and \( P_{12} \) can be implemented by using piecewise-activation functions [21]. Thus (11)-(12) could be implemented in simple hardware units [33].

### 2.2 Model Comparisons

To show the advantages of the proposed model (11)-(12), we will compare it with six neural network in being.

First, consider the model in [7] for (1) is

\[
\frac{d}{dt} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = -\begin{pmatrix} D^T(Dx - f - z) + A^T P_Y(y + Ax - b) \\ (I_m + AA^T)y - AD^T z - P_Y(y + Ax - b) \\ D(x - A^T y + D^T z - f) \end{pmatrix},
\]
(13)

where \( A \) and \( b \) are defined in (1), \( I_n \) is an identity matrix of order \( n \),

\[ x \in R^n, y \in R^m, z \in R^{(n+p)}, D = [C^T, -C^T, I_n, -I_n] \in R^{[n+(n+2p)] \times n}, f = (d^T, -d^T, h, -l)^T \in R^{[n+(n+p)]}. \]

The model (13) has two-layer structure and requires \( m + 3n + 2p \) neurons, while (11)-(12) is only one-layer and needs \( m + n \) state variables. Moreover, the convergence of (13) to an optimal solution is questionable (see [27, 33] and Example 1 in Section 3).

Next, let us focus on the models in [12], [26] and [27], their state equations for (1) are listed as follow
For all of (14), (15) and (16), \( F_1 = P_Y(y + Ax - b), \ x \in \mathbb{R}^m, y \in \mathbb{R}^n, z \in \mathbb{R}^p. \) Their corresponding output equation are 
\( x = P_X(x), \ x = x \) and \( x = x \), respectively. Thus, they all require \( m + n + p \) state variables, while (11)-(12) needs \( m + n \) state variables. Furthermore, the first two of them both have two-layer and the third one has three-layer, however the model designed by this paper has only one-layer.

Third, let us look at the model in [29], for (1), which can be written in details as

\[
\frac{d}{dt} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = -\rho \begin{pmatrix} F_2x + (I_n - 2F_2)P_X(x) + F_2A^TP_Y(y) - d \\ y - AP_xP_Y(x) - P_Y(y) - Ad + b \end{pmatrix}
\]  

(17)

where \( F_i = I_n - C^T(CC^T)^{-1}C, \ d = C^T(CC^T)^{-1}d \in \mathbb{R}^n, y \in \mathbb{R}^n \) and \( x = P_X(x) \) is the corresponding output equation. Although, (17) has \( m + n \) state variables which is same as (11)-(12), this model is two-layer. More importantly, the stability of this model maybe questionable in some case (see [33]). To guarantee the solution of (17) convergent to the optimization solution of (1), there should be exist a parameter \( k > 1 \) such that

\[
W = k \begin{pmatrix} I_n - F_2 & 0 \\ 0 & 0 \end{pmatrix} - \begin{pmatrix} F_2A^TP_Y & 0 \\ 0 & AP_xP_Y \end{pmatrix}
\]

(18)

is positive semi-definite where \( F_2 \) is defined in (17) and 0 is the zero matrix with corresponding dimension. Actually, that parameter maybe not exist (see Example 1 in Section 3).

Finally, we compare the proposed model with the model in [33], whose state equations for (1) is

\[
\frac{d}{dt} \begin{pmatrix} x \\ y \end{pmatrix} = -\rho \begin{pmatrix} 2(F_2x + d - F_3) \\ y - P_Y(y + AF_2x + d - b) \end{pmatrix}
\]  

(19)

where \( F_3 = P_X[2(F_2x + d) - x - A^TP_Y(y + AF_2x + d - b)], \ x \in \mathbb{R}^n, y \in \mathbb{R}^n, F_2, \ d \) are defined in (17). The corresponding output equation is \( x = F_3x + d \). This neural network needs \( m + n \) state variables is same as (11)-(12), while it has three-layer. Thus it is more complex than the proposed one.

To see more clearly, we introduce Table 1. Table 1 shows that the proposed model has the fewest state variables and layer in structure.

<table>
<thead>
<tr>
<th>Model</th>
<th>(11)-(12)</th>
<th>(13)</th>
<th>(14)</th>
<th>(15)</th>
<th>(16)</th>
<th>(17)</th>
<th>(19)</th>
</tr>
</thead>
<tbody>
<tr>
<td>layer(s)</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Neurons</td>
<td>( m + n )</td>
<td>( m + 3n + 2p )</td>
<td>( m + n + p )</td>
<td>( m + n + p )</td>
<td>( m + n + p )</td>
<td>( m + n )</td>
<td>( m + n )</td>
</tr>
</tbody>
</table>

Table 1. Comparison of related models in layer(s) and required neurons
## 2.3 Stability Result

In this subsection, the convergence and stable performance of the proposed model will be studied.

**Theorem 1.** For any starting point \( \eta(0) = ((u(0))^T, (v(0))^T)^T \in \mathbb{R}^{m+n} \), (11) has a unique continuous solution. (11)-(12) is Lyapunov stable. The output variables and state variables will approach an optimal solution of (1) and a point in \( K^* \), respectively. If \( K^* = \{ y^* \} \), the proposed model is globally asymptotically stable.

**Proof:** From the first inequality of (2), \( P_X(\cdot) \) and \( P_Y(\cdot) \) are non-expansive. So, the right-hand-side of (11) is Lipschitz continuous on \( \mathbb{R}^{m+n} \). Such, for any \( \eta(0) = ((u(0))^T, (v(0))^T)^T \in \mathbb{R}^{m+n} \), (11) has a unique continuous solution \( \eta(t) \) on \([0, +\infty)\).

From the assume in Section 1, we know (1) has an optimal solution \( x^* \). There exist a \( \eta^* = [(u^*)^T, (v^*)^T]^T \in K^* \) such that \( x^* = \hat{H}u^* - \hat{H}A^Tv^* + \hat{d} \) by Lemma 2 i). From the second inequality of (2), let \( \omega = 2x - u \) and \( \zeta = P_X(2x^* - u^*) \), we get

\[
[2x - u - P_X(2x - u)]^T [P_X(2x - u) - P_X(2x^* - u^*)] \geq 0;
\]

by setting \( \omega = 2x^* - u^* \) and \( \zeta = P_X(2x - u) \) we have

\[
[2x^* - u^* - P_X(2x^* - u^*)]^T [P_X(2x^* - u^*) - P_X(2x - u)] \geq 0;
\]

let \( \omega = 2y - v \) and \( \zeta = P_Y(2y - v) \) we obtain

\[
[2y - v - P_Y(2y - v)]^T [P_Y(2y - v) - P_Y(2y^* - v^*)] \geq 0;
\]

let \( \omega = 2y^* - v^* \) and \( \zeta = P_Y(2y - v) \), we obtain \( [2y^* - v^* - P_Y(2y^* - v^*)]^T [P_Y(2y^* - v^*) - P_Y(2y - v)] \geq 0. \)

Since \( \hat{H} - \hat{H}H^{-1}\hat{H} = 0 \), then

\[
(u - u^*)^T (x - x^*) - (x - x^*)^T (x - x^*) + (v - v^*)^T (y - y^*) - (y - y^*)^T (y - y^*) \\
= [(u - u^*)^T (v - v^*)]^T (\hat{H} - \hat{H}H^{-1}\hat{H}) (u - u^*) - A^T (v - v^*)] \\
= 0
\]

Now, look at the following Lyapunov function

\[
V(\eta, \eta^*) = \frac{1}{2} ||\eta - \eta^*||^2.
\]

Then

\[
\frac{d}{dt} \frac{d}{dt} V(\eta, \eta^*) = -\rho [(u - u^*)^T (x - P_X(2x - u)) + (v - v^*)^T (y - P_Y(2y - v))] \\
= -\rho [||x - P_X(2x - u)||^2 + ||y - P_Y(2y - v)||^2 \\
+ [2x - u - P_X(2x - u)]^T [P_X(2x - u) - P_X(2x^* - u^*)] \\
+ [2x^* - u^* - P_X(2x^* - u^*)]^T [P_X(2x^* - u^*) - P_X(2x - u)] \\
+ [2y - v - P_Y(2y - v)]^T [P_Y(2y - v) - P_Y(2y^* - v^*)] \\
+ [2y^* - v^* - P_Y(2y^* - v^*)]^T [P_Y(2y^* - v^*) - P_Y(2y - v)] \\
+ (u - u^*)^T (x - x^*) - (x - x^*)^T (x - x^*)
\]
by (11)-(12), the above inequalities and equality in this subsection. Thus, \(V(\eta(t))\) is non-increasing for all \(t > 0\) and (11)-(12) is Lyapunov stable. Hence the set \(\{\eta(t) \mid t > 0\}\) is bounded. The set \(\{\eta(0)\}\) which called as \(\omega\)-limit of \(\eta(t)\), is nonempty, compact, connected and invariant, and \(\eta(t) \to S(\eta(0))\) as \(t \to \infty\) (see [1]). Clearly, \(dV(\eta, \eta^*) / dt = 0\) for all \(\eta \in S(\eta(0))\). So, \(x = P_x(2x-u)\) for all \(\eta \in S(\eta(0))\).

For all \(\eta \in S(\eta(0))\), let \(\eta(t)\) be a unique solution of (11) with \(\eta(0) = \eta\) over \([0, +\infty)\), then \(\eta(t) \in S(\eta(0))\) from the invariance of \(S(\eta(0))\). Thus \(\dot{x}(t) = P_{11}[2\dot{x}(t) - \ddot{u}(t)]\) for all \(t > 0\). Since \(S(z^0)\) is bounded, \(C\dot{x} = d\) and \(\eta \in \mathcal{K}^+\). The remain proof is so similar like Theorem 3 in [21], so they are leaved out in here.

3. Illustrative Examples

Four examples will be introduced in this section to illustrate the theoretical results achieved in Section 2 and the simulation performance of (11)-(12). In the following discussion, \(e_n = (1,1,\cdots,1)^T \in \mathbb{R}^n\) and \(0_n = (0,0,\cdots,0)^T \in \mathbb{R}^n\). The first example shows that the convergence of (13) and (17) is questionable.

**Example 1.** Consider problem (1), where \(X = \{x \in \mathbb{R}^n \mid -1 \leq x_i \leq 1, i = 1,2,\cdots, 5\}\), \(b = (1,-1)^T\), \(d = (-1,0,1)^T\), and

\[
A = \begin{pmatrix} 1 & 1 & 2 & 3 & 2 \\ 1 & 2 & 1 & -2 & 1 \end{pmatrix}, \quad C = \begin{pmatrix} 1 & 1 & 2 & -1 & 2 \\ 3 & -1 & 2 & 1 & 4 \\ 2 & 2 & 4 & 1 & 1 \end{pmatrix}
\]

Then it has a unique optimal point \(x^* = (0.5, 0, 0, 0.5, -0.5)^T\).

When (11)-(12) applied to this question, all simulation results with different starting points indicate that its state and output variables converge to \((u^*, v^*)\) and \(x^*\) respectively. For example, let \(\rho = 10\), Figure 1 shows the convergence of state variables, Figure 2 is the trajectories of output vector, and Figure 3 depicts the convergence performance of ||\(x - x^*\)||.

Figure 4 shows that the output trajectory of the model (13) with three starting points only converges to \((0.125236 - 0.0168240.1084110.476629 - 0.280376)^T\), which is only feasible, but not optimal. The error \(||x(t_f) - x^*|| = 0.448626\). Thus, the model (13) for this example is questionable.

Consider (17), the \(AF_2A^T\) in (18) for this problem is

\[
AF_2A^T = \begin{pmatrix} 3.84 & -0.8 \\ -0.8 & 1.5 \end{pmatrix}.
\]

The eigenvalues of \(AF_2A^T\) are 4.0874 and 1.2526. Thus, for any \(k \in \mathbb{R}\), the \(W\) defined in (18) could not be semi-positive or positive. For example, Figure 5 shows that the second component of the output variables of (17) for \(e^+\) converges to 1, while for \(-e^+\) to -1; the fifth component of the output variables of (17) for \(e^+\) converges to 1, while for \(0^+\) and \(-e^+\) to -1. Therefore, the convergence of the output variables of (17) is questionable.

Example 2 shows (11) - (12) is better than (14), (15), (16) and (19).

**Example 2.** Consider problem (1), where \(X = \{x \in \mathbb{R}^n \mid -0.5 \leq x_i \leq 0.5, i = 1,2,\cdots, 5\}\), \(b = (0,1,1)^T\), \(d = (2,0)^T\), and
Figure 1. Transient states of (11)-(12) with 20 random initial points in Example 1

Figure 2. Transient behavior of (11)-(12) with 20 random starting points in Example 1

Figure 3. Convergence of (11)-(12) with 20 random starting points in Example 1
Clearly, it has a fundamental solution system as $x_1 = 11 + 21.6x_5$, $x_2 = 1 + 2.4x_5$, $x_3 = -5 - 10x_5$, $x_4 = -2 - 5x_5$ where $x_5 \in [-0.5, -35/72]$ is a free variable. Thus this problem has countless solutions. When (11)-(12) used to solve this problem, all numerical results with different starting points show that its state variables converge to one of equilibrium point and its output variables converge to one optimal solution of (1). Figure 6 shows $\|x(t) - x^*\|$ of (11)-(12) with $\rho = 10$ and 20 random initial points converge to zero.

To compare with the proposed one, (14), (15), (16) and (19) were used to this problem. Their simulation results with three types initial points are listed in Table 2.
In there, $t_f$ represents the time that the stopping criterion, $I_N$ denotes the iterative number, $\|X(t_f) - X^*\|$ is the error of output variables, the norm of the right-side hand of these neural network being less than $10^{-5}$, is met. Table 2 shows the proposed model convergence faster than other models.

Table 2. Numerical results of Example 2 with $\rho = 10$

<table>
<thead>
<tr>
<th>Model</th>
<th>initial point</th>
<th>$t_f$</th>
<th>ItN</th>
<th>CPU (sec.)</th>
<th>$X(t_f) - X^*$</th>
</tr>
</thead>
<tbody>
<tr>
<td>(11)-(12)</td>
<td>$e_7$</td>
<td>8.563179</td>
<td>88</td>
<td>0.078125</td>
<td>$4.45 \times 10^{-7}$</td>
</tr>
<tr>
<td></td>
<td>$0_7$</td>
<td>5.124266</td>
<td>99</td>
<td>0.062500</td>
<td>$5.62 \times 10^{-7}$</td>
</tr>
<tr>
<td></td>
<td>$-e_7$</td>
<td>7.480824</td>
<td>84</td>
<td>0.078125</td>
<td>$2.76 \times 10^{-7}$</td>
</tr>
<tr>
<td>(14)</td>
<td>$e_9$</td>
<td>6.603586</td>
<td>156</td>
<td>0.156250</td>
<td>$5.81 \times 10^{-7}$</td>
</tr>
<tr>
<td></td>
<td>$0_9$</td>
<td>3.648162</td>
<td>130</td>
<td>0.125000</td>
<td>$5.65 \times 10^{-7}$</td>
</tr>
<tr>
<td></td>
<td>$-e_9$</td>
<td>5.869221</td>
<td>171</td>
<td>0.187500</td>
<td>$3.51 \times 10^{-7}$</td>
</tr>
<tr>
<td>(15)</td>
<td>$e_9$</td>
<td>7.594577</td>
<td>168</td>
<td>0.156250</td>
<td>$7.98 \times 10^{-7}$</td>
</tr>
<tr>
<td></td>
<td>$0_9$</td>
<td>3.548143</td>
<td>126</td>
<td>0.109375</td>
<td>$5.64 \times 10^{-7}$</td>
</tr>
<tr>
<td></td>
<td>$-e_9$</td>
<td>5.078142</td>
<td>190</td>
<td>0.328125</td>
<td>$8.71 \times 10^{-7}$</td>
</tr>
<tr>
<td>(16)</td>
<td>$e_9$</td>
<td>11.258303</td>
<td>154</td>
<td>0.171875</td>
<td>$5.54 \times 10^{-7}$</td>
</tr>
<tr>
<td></td>
<td>$0_9$</td>
<td>6.432638</td>
<td>146</td>
<td>0.140625</td>
<td>$7.34 \times 10^{-8}$</td>
</tr>
<tr>
<td></td>
<td>$-e_9$</td>
<td>7.949948</td>
<td>166</td>
<td>0.234375</td>
<td>$6.17 \times 10^{-8}$</td>
</tr>
<tr>
<td>(19)</td>
<td>$e_7$</td>
<td>8.589204</td>
<td>126</td>
<td>0.093750</td>
<td>$4.11 \times 10^{-7}$</td>
</tr>
<tr>
<td></td>
<td>$0_7$</td>
<td>4.957489</td>
<td>120</td>
<td>0.109375</td>
<td>$3.71 \times 10^{-7}$</td>
</tr>
<tr>
<td></td>
<td>$-e_7$</td>
<td>5.711741</td>
<td>127</td>
<td>0.109375</td>
<td>$2.66 \times 10^{-7}$</td>
</tr>
</tbody>
</table>
Example 3 shows the good performances of (11)-(12) in solving general least absolute deviation.

**Example 3.** Consider problem (1), where \( X = \{ x \in \mathbb{R}^n \mid -1 < x_i < 1, i = 1, 2, \ldots, 5 \} \) and \( d = (1, -1)^T \).

\[
A = \begin{pmatrix}
1 & 0 & -1 & 0 \\
0 & 1 & 0 & -1 \\
2 & 0 & 1 & 0 \\
0 & 2 & 0 & 1 \\
1 & 0 & 2 & 1 \\
1 & 1 & -1 & 0 \\
0 & 1 & -1 & 1 \\
2 & 1 & 0 & 0 \\
1 & 2 & 1 & 0 \\
2 & -1 & 2 & 1
\end{pmatrix}, \quad b = \begin{pmatrix}
1 \\
-1 \\
-1 \\
-1 \\
1 \\
1 \\
1 \\
1 \\
1 \\
-1
\end{pmatrix}, \quad C = \begin{pmatrix}
1 & 2 & 1 & 1 \\
1 & 0 & 1 & 1
\end{pmatrix},
\]

We use (11)-(12) to solve this problem. All numerical results show that the state variables of (11)-(12) converge to one of its equilibrium point and its output variables are asymptotically stable at the optimal point, respectively.

![Figure 7. Out trajectories of (11)-(12) with 20 random starting points in Example 3](image)

Example 4 shows that (11)-(12) can be applied to solve large-scale problem.

**Example 4 [32].** Let \( A = I_n, b = 0, (1) \) is the same as the problem considered by Li i.e. in [32]. Then, (11)-(12) can be applied to solve the Example 3 in [32].

Here, we apply the neural network (11)-(12) to the two-circle, rose and three-roses images shown in Figure 9 (a), (a1) and (a2). All the three images can be changed into eight-bit gray level images. The sizes of them are: (a) has 64 by 64 pixels, both (a1) and (a2) have 128 by 128 pixels. Figure 9 (b), (b1) and (b2) show the blurred images. Figure 9 (c), (c1) and (c2) show the images restored by (11)-(12) with \( \rho = 1 \), from (b), (b1) and (b2), respectively. Clearly, the restored images are more clearer than blurred images.
4. Conclusion

This paper presents a one-layer model for solving least absolute deviation problems including both box and equality constraints. Compared with existing models, the proposed neural network requires the fewest neurons and layers and has good performance. The stability and convergence of proposed one is proved by introducing a Lyapunov function.

Some simulation results verify the good results in theory.
Figure 9. Images for Example 3. (a), (a1) and (a2) are original images; (b), (b1) and (b2) are blurred images; (c), (c1) and (c2) are restored by (11)-(12)

References


