A Compact Neural Network for Fused Lasso Signal Approximator

Mohammadi, Majid

DOI
10.1109/TCYB.2019.2925707

Publication date
2021

Document Version
Accepted author manuscript

Published in
IEEE Transactions on Cybernetics

Citation (APA)
Mohammadi, M. (2021). A Compact Neural Network for Fused Lasso Signal Approximator. *IEEE Transactions on Cybernetics*, *51*(8), 4327-4336. [8766144]. https://doi.org/10.1109/TCYB.2019.2925707

Accepted Author Manuscript. Link to published article (IEEE): https://doi.org/10.1109/TCYB.2019.2925707

A Compact Neural Network for Fused Lasso Signal Approximator

Majid Mohammadi

Abstract—The fused lasso signal approximator (FLSA) is a vital optimization problem with extensive applications in signal processing and biomedical engineering. However, the optimization problem is difficult to solve since it is both non- smooth and nonseparable. The existing numerical solutions implicate the use of several auxiliary variables in order to deal with the nondifferentiable penalty. Thus, the resulting algorithms are both time- and memory-inefficient. This paper proposes a compact neural network to solve the FLSA. The neural network has a one-layer structure with the number of neurons proportion- ate to the dimension of the given signal, thanks to the utilization of consecutive projections. The proposed neural network is stable in the Lyapunov sense and is guaranteed to converge globally to the optimal solution of the FLSA. Experiments on several applica- tions from signal processing and biomedical engineering confirm the reasonable performance of the proposed neural network.

Index Terms—Fused lasso, global convergence, Lyapunov, neural network.

I. Introduction

THE FUSED lasso signal approximator (FLSA) has diverse applications in several disciplines. The FLSA enforces sparsity both in the coefficients and the differences between consecutive elements in the coefficient vector. A class of the corresponding optimization problem is

$$\min_{x} \frac{1}{2}\|x-y\|^2 + M \sum_{i=1}^{n} N + A_{2} \sum_{i=1}^{n-1} |x - x + d \quad \text{(FLSA)}$$

where $k_1, k_2 > 0$ are the regularization parameters, $y \in R^n$ is the observed signal, and $x \in R^n$ is the noise-free approximation ofy. If $X_2 = 0$, FLSA has the closed-form solutionby applying the soft-thresholding operator [1], and for $M_1 = 0$, FLSA boils down to the total variation denoising problem [2]. As a result, the solution to this minimization yields the solution to the total variation denoising as well.

Neural networks have been long used for finding the optimal solution to optimization problems since the Hopfield's pioneering works on solving the combinatorial traveling sales- man problem [3] and linear programming [4]. Ever since, diverse neural solutions have been put forward to solve vari- ous optimization problems with respect to manifold types of constraints [5]-[14]. Solving optimization problems using neu- ral networks have several salient advantages over traditional numerical methods in real-time processing. First and fore- most, the structure of the neural network can be effectively implemented using very-large-scale integration and optical technologies [15]. Thus, when there is a demand for real- time optimization, neural networks offer the practical solution. In addition, ordinary differential equations (ODEs) repre- senting a recurrent neural network can be solved by using different methods so they can be implemented on digital computers as well. Another advantage of neural networks is that they can converge globally to the exact optimal solu- tion of the given minimization regardless of their initial values.

Zhang and Constantinides [6] presented a Lagrangian neural network for solving general nonlinear programming. Bouzerdoum and Pattison [5] developed a neural network for quadratic programming with box constraints only by writing the partial dual of the given minimization. Quadratic pro- gramming has been considered in numerous research studies and, thus, many networks exist to solve a particular case of quadratic minimization. One case is the strictly convex quadratic programming for which many neural solutions have been tailored [9]-[11], [16]-[18]. Another neural network is for the case when the second derivative of the objective func- tion equals the identity matrix [19], and it has been shown that the neural network has a simpler structure when other neural networks are applied to the same minimization. Several neural solutions are also presented to solve general quadratic programming [20]-[22].

Aside from neural networks for general optimization prob- lems, several well-known engineering problems have also been solved by using projection neural networks. For instance, there are several neural networks for support vector machine training [23], [24]. There are also neural solutions for regression analysis [25], [26], robot control [27], image restoration [28], image fusion [29], and non-negative matrix factorization [30]. One of the essential problems which has been considered recently is the l_1 minimization, for which several neural networks have been presented [31]-[36]. The optimization problem has a similar challenge to FLSA since it implicates the minimization of the l_1 -norm. However, the FLSA is more complicated as it entails two nonsmooth terms, one of which minimizes the difference among consecutive elements in the coefficient vector.

Manuscript received October 4, 2018; revised January 25, 2019, April 16, 2019, and June 11, 2019; accepted June 22, 2019. Date of publica- tion July 18, 2019; date of current version August 4, 2021. This paper was recommended by Associate Editor S. Squartini.

The author is with the Faculty of Technology, Policy, and Management, Delft University of Technology, 2628 BX Delft, The Netherlands (e-mail: m.mohammadi@tudelft.nl).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCYB.2019.2925707.

Digital Object Identifier 10.1109/TCYB.2019.2925707

As FLSA has two nondifferentiable terms, finding its optimal solution is not straightforward. Techniques in the lit- erature usually need to employ several auxiliary variables in order to deal with the nonsmoothness and nonseparability of the problem [37]-[40]. As a result, the resulting algorithms have higher time and memory complexity. In contrast to those solvers, the Condat's method is a noniterative method which solves FLSA for $A_1 = 0$ [41]. The complexity of algorithm on the most real problems is $O(n)$ and its worst case is $O(n^2)$. The method, though very fast, is sequential by nature since the optimal value of x_j is reliant on the optimal value x_{i-1}. As a result, the algorithm is not prone to parallel execution.

In this paper, a projection neural network is presented to solve FLSA. The proposed neural solution has a simple one- layer structure and is efficient in terms of both the component required for its circuit implementation and the number of oper- ations in each iteration. The neural model deals with the two nonsmooth terms in FLSA by using two consecutive projec- tions, which results in a network with $n — 1$ neurons, where n is the length of the given data. As a result, the proposed network is compact since the dimension of the network is lin- ear with respect to the dimension of the given data. The neural model is guaranteed to be stable in the sense of Lyapunov, and its trajectory is assured to converge to the optimal solution of FLSA given any arbitrary initial point. The neural solution is further applied to several problems in signal processing and biomedical engineering and the results are compared to the state-of-the-art solvers. In contrast to the Condat's method, the proposed network can also be applied to other problems, such as weighted total variation [42] and trend filtering [43]. The complexity of the solving is of $O(n)$ for a finite number of iterations.

In summary, the contributions of this paper are as follows.

1) A neural model is tailored for FLSA, which is, to the best of my knowledge, the first attempt to solve this problem using neurodynamic optimization. The neural model has a straightforward structure as it uses two successive projections.
2) The network is proved to be stable in the sense of Lyapunov, and it is assured that it converges globally to the optimal solution to FLSA.
3) The proposed model is simplified in the case where $A_1 = 0$ in FLSA. In this case, the neural network boils down to a well-known projection neural network, where its exponential convergence could also be guaranteed.

The remainder of this paper is structured as follows. The neural network model is developed in Section II, and its sta- bility is assured in Section III. In Section IV, the proposed model is simplified for circumstances when $A_1 = 0$ in FLSA, and it is shown that the proposed neural network boils down to a well-known projection model with guaranteed exponential convergence. The experiments regarding the proposed neural network are presented in Section VI. Finally, this paper is concluded in Section VII.

II. Neurodynamic Model

In this section, a neurodynamic model is developed for FLSA. To this end, FLSA can be equivalently rewritten as

$$\min_x \frac{1}{2}\|x — y\|^2 + A_1 \|x\|_1 + k_2 \|Dx\|_1 \qquad \text{(FLSA2)}$$

where $\|.\|_1$ is the l_1-norm and $D \in R^{(n-1)xn}$ is defined as

$$D = \begin{bmatrix} 1 & —1 & 0 & \dots & 0 \\ 0 & 1 & —1 & & \\ & & \ddots & \ddots & 0 \\ . & 0 & & & 0 \\ 0 & 0 & \dots & 1 & —1 \end{bmatrix}$$

Since FLSA2 is convex, the Karush-Kuhn-Tucker (KKT) conditions [44] are necessary and sufficient for the optimality. On the other hand, it is required to use the property of the subgradients for FLSA2 due to its nonsmoothness. Thus, x* is the optimal solution of FLSA2 if there are $w*$ and $z*$ such that $x—y+w+D^T z = 0$, where $w \in A_1 9(|x|_1)$ and $z \in A_2 9(|Dx|_1)$. The derivative of $|x|_1$ is not defined at zero, but the derivative is simply one if $x_j > 0$, and —1 otherwise. Thus, one can write

$$w_j = \begin{cases} M & \text{if } x_j > 0 \\ \in [—A_1, M] & \text{if } x_j = 0 \\ —A_1 & \text{if } x_j < 0. \end{cases} \qquad (1)$$

The conditions in (1) can be restated using the well-known projection operator as [45]

$$w = P_{X1} (w + x) \qquad (2)$$

where $P^\wedge (a) = [P^\wedge (a_1), P^\wedge (a_2), \dots, P_{A1} (a_n)]$, and

$$P_M(a_j) = \begin{cases} A_1 & \text{if } a_j > A_1 \\ \in [—A_1; A_1] & \text{if } |a_j| < A_1 \\ —A_1 & \text{if } a_j < —A_1. \end{cases}$$

Similarly, one obtains the same projection equation for z as

$$z = P_{A_2} (z + Dx).$$

As a result, the KKT conditions can be summarized as

$$\begin{cases} x — y + w + D^T z = 0 \\ w = P_{X1} (w + x) \\ z = P_{A_2}(z + Dx). \end{cases} \qquad (3)$$

From the first equation in (3), it follows that $x = y—w —D^T z$. Replacing it in the second equation, one arrives at $x = y — D^T z — P_{A1} (y — D^T z)$. As a result, the conditions in (3) can be rewritten simply as

$$z = P_{A_2} (z — DP_{A1} (y — D^T z) — Dy + DD^T z). \qquad (4)$$

Based on (4), a neural network is proposed with the dynamic equation as being given by

State Equation

$$\frac{dz}{dt} = P_{A_2} (z — h(z)) — z \qquad (5)$$

output Equation

$$x = y — D^T z — P_{AI} (y — D^T z) \qquad (6)$$

where $h(z) = DP_{A1} (y — D^T z) — Dy + DD^T z$. The dynamic system given in (5) can be easily recognized by a simple one- layer neural network, as shown in Fig. 1 for a general matrix D. The number of neurons for FLSA is $k = n — 1$, hence

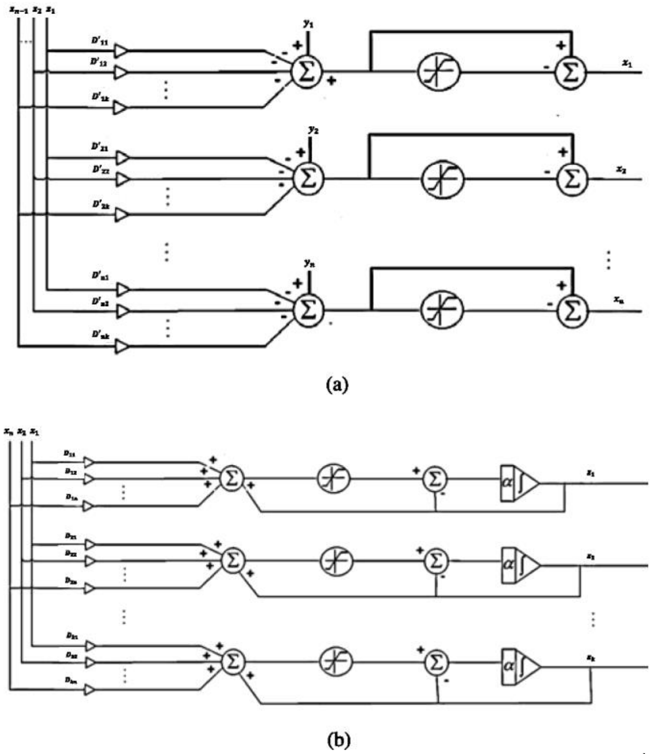

Fig. 1. Architecture of the proposed neural network for general matrix $D \in R^{k\times n}$, where $k = n - 1$ for FLSA. (a) Output circuit of the proposed neural network. (b) Dynamic circuit of the proposed neural network.

the structure of the network grows linearly with respect to the dimension of the given data. According to Fig. 1 with $D^{k\times n}$, the circuit realizing the proposed neural network consists of k integrators, $n + k$ piecewise activation functions, $2n + 2k$ summers, $2nk$ multipliers, and some weights.

III. Stability Analysis

This section contains the stability analysis of the proposed neural network. First, some required definitions and lemmas are presented.

Definition 1: x^* is the equilibrium point of the dynamic system

$$\dot{x} = f(x),\ x(t_0) = x_0 \in R^n \tag{7}$$

where $f : R^n \to R^n$ is a function, if $f(x^*) = 0$.

Definition 2 (Lyapunov Stability): Let $x(t)$ be a solution of the dynamic system in (7). An equilibrium point x^* is stable if, for $t > t_0$

$$\forall \epsilon > 0, \exists \delta > 0, \text{ s.t. } \|x(t) - x^*\| < \epsilon \text{ if } \|x(t_0) - x^*\| < \delta.$$

The dynamic system is asymptotically stable if $x(t) \to x^*$ as $t \to \infty$ holds as well.

Definition 3 (Monotonicity): A mapping H is called monotone at $y \in Q$ if

$$(x - y)^T (H(x) - H(y)) \geq 0 \quad \forall x \in Q.$$

If the above inequality holds for any $x, y \in Q$, then the mapping H is said to be monotone in Q.

Definition 4 (Variational Inequality): The variational inequality problem is about finding $x^* \in Q$ such that

$$(x - x^*)^T H(x) \geq 0 \ \forall x \in Q$$

where H is a function from R^n into itself. It was shown that x^* is the solution of the above inequality if and only if the following projection equation holds [46]:

$$x = P_Q(x - H(x))$$

where $P_Q()$ is the projection operator defined as

$$P_Q(z) = \underset{y \in Q}{\operatorname{argmin}} \|z - y\|.$$

Lemma 1 (Projection Properties [46]): Let $Q \subset R^n$ be a closed convex set, then:

1) $(a - P_Q(a))^T(P_Q(a) - b) \geq 0,\ a \in R^n,\ b \in Q$;
2) $\|P_Q(a) - P_Q(b)\|^2 \leq (P_Q(a) - P_Q(b))^T(a - b) \leq \|a - b\|^2,\ a, b \in R^n$.

The stability and the convergence of the proposed neural network in (5) is now examined. Without loss of generality, we assume that $D \in R^{(n-1)\times n}$. Before stating the main results, it is first shown that the function $h(z)$ in (6) is monotone.

Lemma 2: The mapping $h(z) = D(P_{X_l}(y - D^Tz) - y + D^Tz)$ is monotone in R^{n-1}.

Proof: For any $u, v \in R^{n-1}$, one obtains

$$\begin{aligned}
&(h(u) - h(v))^T (u - v) \\
&\quad = (P_{X_l}(y - D^Tu) - y + D^Tu \\
&\qquad - P_{X_l}(y - D^Tv) + y - D^Tv)^T (D^Tu - D^Tv) \\
&\quad = (P_{X_l}(y - D^Tu) - P_{X_l}(y - D^Tv))^T(D^Tu - D^Tv) \\
&\qquad + \|D^Tu - D^Tv\|^2.
\end{aligned} \tag{8}$$

Let $a = y - D^Tu$ and $b = y - D^Tv$ in Lemma 1-2), then $(P_{X_l}(y - D^Tu) - P_{X_l}(y - D^Tv))^T(-D^Tu + D^Tv) \leq \|D^Tu - D^Tv\|^2$

$$\begin{aligned}
&\to \|D^Tu - D^Tv\|^2 + (P_{X_l}(y - D^Tu) - P_{X_l}(y - D^Tv))^T \\
&\quad \times (D^Tu - D^Tv) \geq 0.
\end{aligned} \tag{9}$$

Considering (8) and (9), it follows that:

$$(h(u) - h(v))^T(u - v) \geq 0$$

and h is thus monotone. ■

Now, consider the function H with the definition

$$H(z) = \frac{1}{2}\|y - D^Tz - P_{X_l}(y - D^Tz)\|^2. \tag{10}$$

The following two lemmas are about the features of this function.

Lemma 3 [47]: The derivative of the function $H(z)$ given in (10) is $h(z)$, for example, $\nabla H(z) = h(z)$.

Lemma 4: The function H given above is convex.

Proof: The convexity of $H(z)$ can be simply proved using its first derivative. In particular, one obtains

$$\begin{aligned}
&(z_1 - z_2)^T (\nabla H(z_1) - \nabla H(z_2)) \\
&\quad = (z_1 - z_2)^T(h(z_1) - h(z_2)) \geq 0
\end{aligned}$$

where the last inequality holds true since h is monotone. Thus, H is convex. ■

Lemma 5: There exists a unique continuous solution trajectory $z(t)$ for the dynamic system (5) with any given initial point for t e [t_0, T).

Proof: It is first required to show that the right-hand side of the dynamic system is Lipschitz. To do so, let $z_{1;}$ z_2 e R^{n-1} be two arbitrary variables, then

$$||z1 - Ph\ (Z1 - h(Z1)) - Z2 + Ph\ (Z2 - h(Z2))\ ||$$
$$< \ 2||z1\ -z2\backslash\backslash + ||h(z1) - h(z2)\backslash\backslash$$
$$< \ \backslash\backslash z1 - z2\backslash\backslash + ||D||\ \backslash y - D^Tz1 - P_{kl}\ \{y - D^Tz1)$$
$$-y + D^Tz1 + P_{Xl}\ \{y - D^Tz1)\backslash. \qquad (11)$$

Now, consider the following inequality for any arbitrary x, y, and PQ:

$$\backslash x - PQ(X)\ -y + PQO)\ \backslash\ |^2 = ||x -y||^2 + ||PQ(X) - Pa(y)\backslash\backslash^2$$
$$- 2(PQ(X) - Pn(y))^T(x\ -y)$$
$$<^{(1)} \backslash x -y||^2 -||PQ(X) -PQ(y)\backslash\backslash^2 < \backslash x - y||^2 \qquad (12)$$

where (1) is obtained by using Lemma 1-2). Let $x = b - D^Tz_1$ and $y = b - D^Tz_2$ in (12), then one can rewrite (11) as

$$\backslash z1 - P_{Xl}\ (z1 - h(z1)) - z2 + P_{Xl}\ (z2 - h(z2))\ \backslash$$
$$< \ 2\backslash z1 - z2\ \backslash| + ||D\ \backslash^2\ \backslash z1 - z2||$$
$$< \ (2 + ||D||^2)\ ||zx - z2|.$$

Therefore, the right-hand side of the dynamic equation is Lipschitz. According to the Peano's theorem for ODEs [48], a unique continuous solution $z(t)$ exists for the dynamic system in (5) in the interval [t_0, T). ■

The stability of the dynamic system in (5) is now explored.

Theorem 1: The neural network governed by the dynamic equation in (5) is stable in the sense of Lyapunov and con- verges globally to a unique equilibrium z^*. The unique solution to FLSA is then obtained by the output equation in (6).

Proof: With any arbitrary initialization z_0, the trajectory of the dynamic system in (5) has a unique continuous solution trajectory, thanks to Lemma 5. Consider the following Lyapunov function [19], [49]:

$$V(z) = H(z) - H(z^*) + h(z^*)^T(z^* - z) + 1\ ||z - z^*\ ||^2$$

where z^* is the equilibrium of the dynamic system and u e R^{n-1}. Since H is convex, it simply follows that [46]:

$$H(z) - H(z^*) + h(z^*)(z^* - z) > 0.$$

Thus, $V(z) > ||z - z^*||^2/2$, which means that $V(z)$ ^ +TO as $||z||$ ^ TO [49]. The time derivative of the Lyapunov function is then obtained as [19], [49] $\{h(z) - h(z^*) + z - z^*)^Tdt$

$$\frac{dV(z)}{dt} \quad \{h(z) - h(z^*) + z - z^*)^T\{PX_2\ (z - h(z)) - z). \qquad (13)$$

Now, let $a = z - h(z)$ and $b = z^*$ in Lemma 1-1), one obtains [19], [49]

$$\{PX_2\ (z - h(z)) - z^*)^T\{z - h(z) - PX2\ (z - h(z))) > 0. \qquad (14)$$

Further, using Definition 4 with $x = P_{X2}\ (z - h(z))$ results in

$$\{PX_2\ \ (z - h(z)) - z^*)^Th\{z^*) > 0, \ Vz\ e\ R^{n-1}. \qquad (15)$$

Adding the inequalities in (14) and (15), we obtain

$$\{PX_2\ (z - h(z)) - z^* + z - z)^T$$
$$X\ \{z - h(z) - PX_2\ (z - h(z)) + h(z^*)) > 0.$$

It follows [49]:

$$\{z - z^* + h(z) - h(z^*))^T\{PX2(z - h(z)) - z)$$
$$< \ \sim\{z - z^*)^T\{h(z) - h(z^*)) - ||z - PX2(z - h(z))W^2$$
$$< \ 0 \qquad (16)$$

where the last inequality is obtained as $(z - z^*)^T\ (h(z) - h(z^*)) >$ 0 due to the monotonicity of $h(z)$, and $||.||^2$ is evidently non-negative. Plugging (16) into (13), it follows:

$$\frac{d\ V(z)}{d\ t} < 0$$

and the dynamic system in (5) is thus stable in the sense of Lyapunov.

According to the invariant set theorem [50], all trajecto- ries of the proposed neural network converge to the largest invariant set n, where $d\ V(z)/dt = 0$. Now, it is shown that $dV(z)/dt = 0$ if and only if $dz/dt = 0$. If $dz/dt = 0$, then

$$\frac{d\ V(z)}{d\ t} \quad d\ V(z)\ \backslash^T\ /dz\ dz\ )\ l\ dt = 0.$$

Conversely, $d\ V(z)/dt = 0$ implies $(z - z^*)^T\{h(z) - h(z^*)) + ||z - PX2(z - h(z))W = 0.$

Since both terms in the above equation are non-negative, it follows that:

$$l||z - Ph2\ (z - h(z))W = 0\ ^\wedge\ z - Ph2\ (z - h(z)) = 0.$$ Hence, $dz/dt = 0$ and

$$\lim_{t^\wedge(X} dist(z(t), n) = 0.$$

Finally, the above inequalities show that $z(t)$ is bounded. The boundedness of trajectories of the dynamic system in (5) means that a subsequence $\{z(t_k)\}$ exists such that

$$\lim_{k^\wedge<X} z(tk) = z$$

where z is an equilibrium of (5). Consider the following Lyapunov function:

$$V(z) = H(z) - H(z) + h(z)^T(z - z) + —\ ||z - z||^2.$$

Then, lim^», $z(tk) = V(u) = 0$. Hence, there exists 5 > 0 for any $e > 0$ such that $< {}^t(z(t)) < V(z(t_q)) < e$.

Therefore, $\frac{1}{2}$ ■$x\backslash z(t)$ -z $\lim_{z^\wedge OT} z(t) = z$. As a result, the dynamic system in (5) is globally convergent to one of its equilibrium points. According to Lemma 5, the state trajectory of (5) is unique, thus it converges to a unique equilibrium point z^*, where the unique solution to FLSA is obtained by $x^* = y - D^Tz^* - PX2\ (y - D^Tz^*)$. ■

IV. Simplification for Üther Problems

This section presents the simplification of the proposed neu- ral network in order to be applied to two well-known problems:

1) total variation denoising and 2) trend filtering.

If $\lambda_1 = 0$, FLSA can be restated as

$$\min_x \frac{1}{2} \|x - y\| + \lambda_2 \|Dx\|_1 \quad (17)$$

which is the total variation denoising problem [2]. The KKT conditions for this minimization can be written as

$$\begin{cases} x - y + D^T z = 0 \\ z = P_{X_2}(z + Dx). \end{cases}$$

Thus, the neural model can be simply rewritten as

State equation

$$\frac{dz}{dt} = P_{X_2}(z - h(z)) - z \quad (18)$$

Output

$$x = y - D^T z$$

where $h_1(z) = DD^T z - Dy$. In comparison to the neural network for FLSA, the neural network in (18) does not have the projection to X_1, since it is zero. The neural model in (18) is a linear projection equation, which has been extensively studied in different research [11], [49], [51], [52]. Since DD^T is positive definite, it has been investigated that the trajectory of this neural network is exponentially convergent in finite time to the unique equilibrium [51].

The approximation of the trend in a given time series is a fundamental problem that arises in a variety of disciplines. Trend filtering has been also modeled similar to (17), in which matrix D is replaced with $D \in R^{(n-2)\times n}$ defined as [43]

$$\begin{bmatrix} -1 & 2 & -1 & \dots & 0 & 0 & 0 \\ 0 & -1 & 2 & \dots & 0 & 0 & 0 \\ & & & \ddots & & & \\ 0 & 0 & \dots & & -1 & 2 & -1 \end{bmatrix}$$

Since the neural model in (18) does not impose any assumption on D, it can be simply applied to this problem as well. On the other hand, matrix D is also full row rank, thus DD^T is positive definite, and the dynamic system in (18) is thus exponentially convergent in finite time.

Another important problem is the proposed model can solve the weighted total variation, that is, [42]

$$\min \frac{1}{2} \|x - y\|^2 + \lambda_1 \sum_{i=1}^{n} |x_i| + \lambda_2 \sum_{i=1}^{n-1} a_i |x_i - x_{i+1}|$$

where $a \in R^{n-1}, a_i > 0$ penalizes the difference between the associated elements in the vector. One can define $D = \mathrm{diag}(a)D$, where diag is a diagonal matrix whose diagonal elements are a, making the problem turn into FLSA2. Thus, the proposed neural network can also solve this problem.

V. Complexity of the Neural Solution

For a general matrix $D \in R^{k\times n}$, the proposed neural network in (5) requires $2nk$ multiplications and $2nk + n + k$ additions/subtractions. However, the computations can be more economical in some special cases, such as FLSA or the total variation denoising (17).

Algorithm 1 FLSA Computations

```
Input: y ∈ R^n, z ∈ R^(n-1), λ1, λ2
for iter=1:n do
  if i == 1 then
      temp = y1 − z1
  end if
  if i == n then
      temp = yn + zn
  else
      temp = yi + z(i−1) − zi
  end if
  if temp > λ1 then
      xi = temp − λ1
  end if
  if temp < −λ1 then
      xi = temp + λ1
  else
      xi = 0
  end if
end for
for i=1:n-1 do
  temp = zi + xi − x(i+1)
  if temp > λ2 then
      newZi = λ2 − zi
  end if
  if temp < −λ2 then
      newZi = −λ2 − zi
  else
      newZi = temp − zi
  end if
end for
Output newZ
```

For FLSA, the value of $y - D^T z$ is

$$y - D^T z = \begin{bmatrix} y_1 - z_1 \\ y_2 + z_1 - z_2 \\ \vdots \\ y_{n-1} + z_{n-2} - z_{n-1} \\ y_n + z_{n-1} \end{bmatrix}$$

Having the value of λ_1, the value of $x = y - D^T z - P_{X_1}(y - D^T z)$ can be obtained by 3n+3 additions/subtractions (and no multiplication). The time complexity is also linear in n since we only need to iterate over the length of $y - D^T z$. Similarly, $P_X(z + Dx) - z$ can be computed in linear time with respect to n. Overall, the time complexity of recurrent neural network is linear in n for a number of iterations. Algorithm 1 summarizes the procedure of the FLSA computations in each iteration. The algorithm can also be written by one loop, but we deliberately use two loops so that they can be executed in parallel. Each iteration in these loops can be executed independently with respect to other iterations of the loop. Therefore, the algorithm is ideal for parallel computing.

For the total variation denoising, the order is also linear in n, and the number of iterations is finite [11], [49], [51], [52]. Thus, the overall time complexity of the neural network is of $O(n)$ as well, similar to the Condat's method. However, the difference is that the Condat's method is of order n^2 in the worst case and not prone to parallelism. In the proposed neu- ral network, aside from its possibility to be implement using circuits, the computations of x and $P\backslash(z + Dx) — z$ can be conducted in parallel.

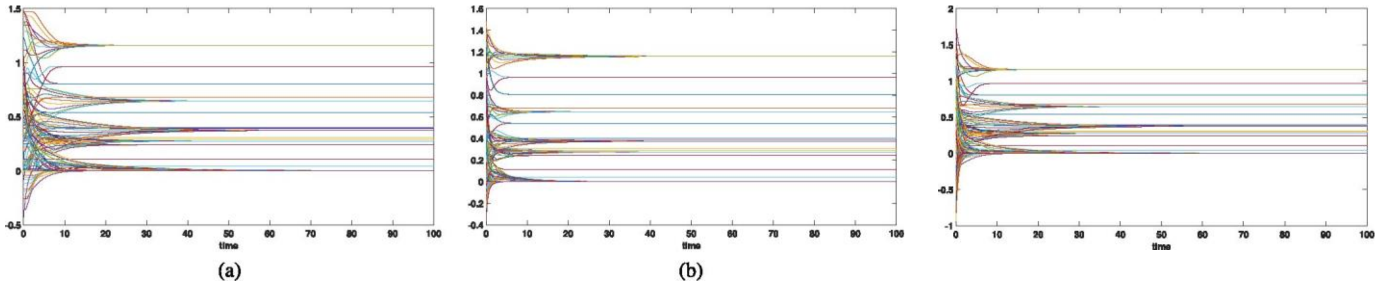

Fig. 2. Convergence of the proposed neural network over a random sample selected from the Pollack *et al.* dataset [58] for aCGH data. The selected sample was subjected to the neural network with three different initial values: (a) with initialization z = 1; (b) with initialization z = 0; and (c) with random initialization. The abscissa is the time, and the ordinate is the value of each element of z at a specific time. The trajectories of each element of z are shown by distinct colors.

VI. Experiments

This section presents the experiments regarding the proposed neural network. The neural network is implemented using the ODE solvers of MATLAB with the tolerance 10^{-5}. The implementation of the proposed network is publicly avail- able.[1] First, we study how parameters in FLSA can be tuned, and then the convergence of the dynamic system in (5) was empirically investigated. Then, the proposed neural network was applied to several real-world problems, including array comparative genomic hybridization (aCGH or CGH array) data recovery and trend filtering. The performance of the neural solution was compared to those of the state-of-the-art methods in each application.

A. *Parameter Tuning*

There are two parameters in (FLsA) whose values can sig- nificantly impact the behavior and the outcome of the neural network. Consequently, it is vital to appropriately tune the parameters. There are several strategies for identifying the appropriate values of such parameters, including, but not lim- ited to genetic algorithms [53], sequential minimization [54], L-curve method [55], and variational Bayes [56]. Another sim- ple yet practical strategy is grid search where we build a model for every possible combination of the parameters X_1 and $A._2$, and then select the parameters with the best model fit gauged by some fitness function.

As there are two parameters in (FLSA), we need to apply a 2-D grid search, which is time consuming in general. However, following a related study [57], we restricted the choice of the parameters to the set 0.1, 0.3, 0.5, 0.7, and 0.9. We further gauged the fitness of an estimated model, shown by X, as [57]

$$/ \backslash\backslash y — XI|^{1}|$$

$$- n \log n \qquad —^{s} \log(n) \tag{19}$$

where s is the number of nonzero elements in X. Finally, we selected the model with the maximum fitness value.

B. *Convergence*

The theoretical analysis showed that the neural network in (5) converges globally to the optimal solution of FLSA, regardless of the initial point. In this experiment, the conver- gence of the network is empirically explored by performing a recovery on aCGH data. In this regard, a sample from the Pollack *et al*. dataset [58] was randomly selected, and it was subjected to the recurrent neural network for recovery.

The sensitivity of the proposed model was investigated by initializing the network with distinct values. The network was initialized with a vector of one, zero, and a random vector, respectively, so that the convergence of the neural solution could be explored in practice. Fig. 2 plots the trajectory of each element of z in the dynamic system in (5) against different time slots. The trajectory of each element is displayed by distinct colors so they can be distinguished in three plots. Based on this figure, the trajectory of the dynamic system is convergent to the same value, regardless of the initialization. This experiment corroborates the global convergence of the proposed neural network, which was theoretically assured in Section III.

C. *aCGH Data Analysis*

aCGH or CGH array is arguably the first important applica- tion of FLSA [59], [60]. The aCGH data helps the diagnosis and prognosis of various diseases such as cancer by finding the aberrational regions in the DNA genome of a given sample. The major impediment to find such regions is that aCGH data is highly corrupted by various sources of noise. Therefore, a recovery method is required to approximate the noise-free data from contaminated observations. It was shown that the result- ing aCGH data of a sample is both sparse and smooth [60]. Hence, FLSA is the minimization which is deemed to recover the noise-free data.

For this experiment, several synthesized and real aCGH data were subjected to the proposed neural solution, and its

[1] https://github. com/Majeed7

performance was compared with those of more sophisticated methods.

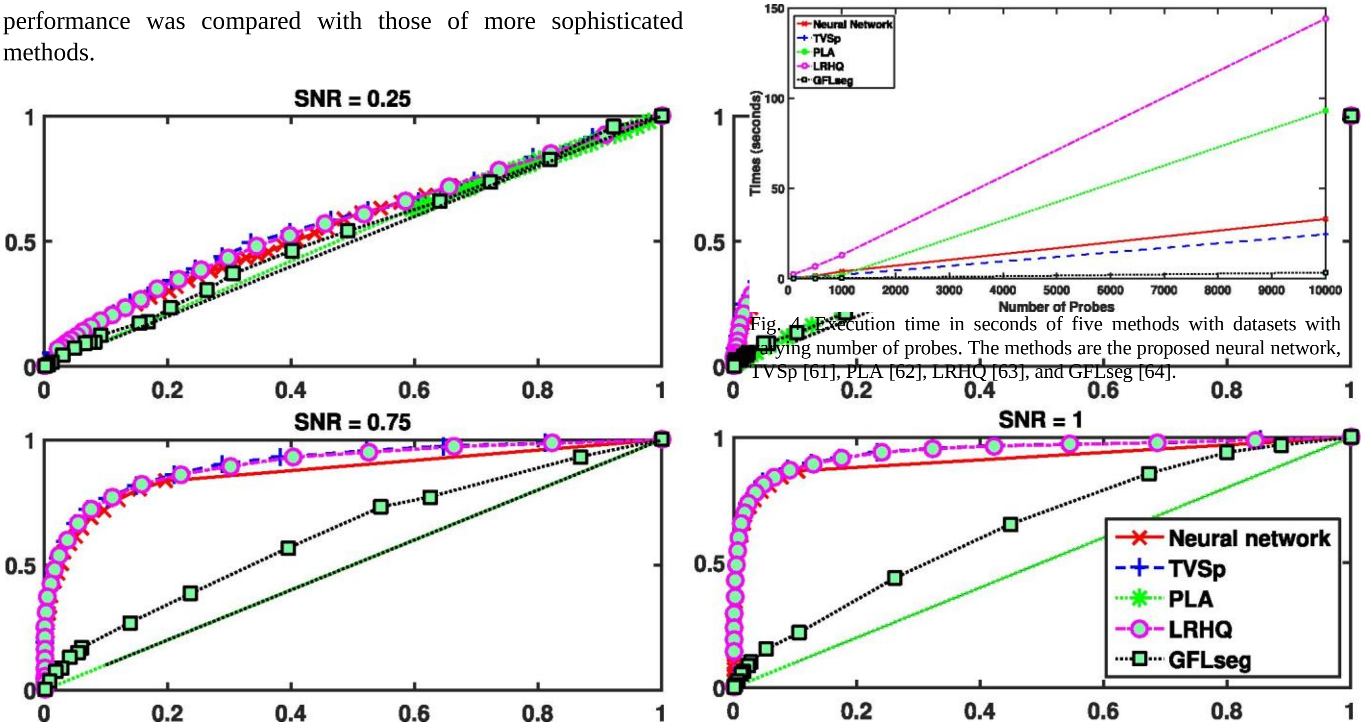

Fig. 4. Execution time in seconds of five methods with datasets with varying number of probes. The methods are the proposed neural network, TVSp [61], PLA [62], LRHQ [63], and GFLseg [64].

Fig. 3. ROC curves plotted from recovery of synthesized data with different SNRs. The abscissa and ordinate of each plot are the TPR and FPR, respectively. The methods are the proposed neural network, TVSp [61], PLA [62], LRHQ [63], and GFLseg [64].

For the synthesized data, 50 samples with the length of 120 were generated. The elements of each aCGH sample were assumed normal if it is zero, and the aberrant regions were created by distorting the elements of each sample with the value 1 or -1. The length of each aberrant region was ran- domly selected from the set {5, 10, 20, 30}. Then, a Gaussian noise was added to each sample to produce the observation. To compare the methods in different corruption levels, several ratios of the data to the noise were considered. These ratios are called signal-to-noise ratios (SNRs). To affect the SNRs, one can simply write

$$y = x + \mathrm{SNR}^{-1}e \qquad (20)$$

where e is the noise distributed according to the standard normal distribution. As a result, the higher values of SNR would result in less corrupted observations, and the smaller values would indicate that the data is highly contaminated with the noise e. The noisy samples were then subjected to the neural network, total variation and spectral regular- ization (TVSp) [61], piecewise and low-rank approximation (PLA) [62], low-rank approximation based on half-quadratic programming (LRHQ) [63], and group fused lasso segmen- tation (GFLseg) [64]. The parameters of methods, when not determined by the method itself, are identically identified similar to the proposed neural network.

Since the ground truth is available in the synthetic exper- iments, one can simply compare the performance of each method by juxtaposing real and recovered data. For this experiment, the receiver operator characteristic (ROC) curve is used to contrast different methods. The ROC curve plots the true positive rate (TPR) against the false positive rate (FPR), and deviation from diagonal is the indicator of the goodness of a method.

Fig. 3 plots the ROC curve of various methods over the syn- thesized data with different SNRs. As the level of corruption decreases, the performance of all methods increases signifi- cantly. In particular, the performance of the proposed neural network is competitive with TVSp and LRHQ in all levels of corruptions. On the other hand, it is significantly superior to GFLseg and PLA in all scenarios.

Although the performance of the proposed network is com- petitive with TVSp and LRHQ, the neural solution is more time- and memory-efficient. TVSp and LRHQ (and also PLA) use the nuclear norm regularization, which needs to compute the singular value decomposition in each iteration. For m samples with the length of n, the singular value decomposition has the complexity of order $O(mn^2)$. Hence, the methods based on the nuclear norm are of higher time complexity.

Regarding memory complexity, the proposed neural network does not need to use any auxiliary variables. The size of the network is commensurate with the length of the given data.

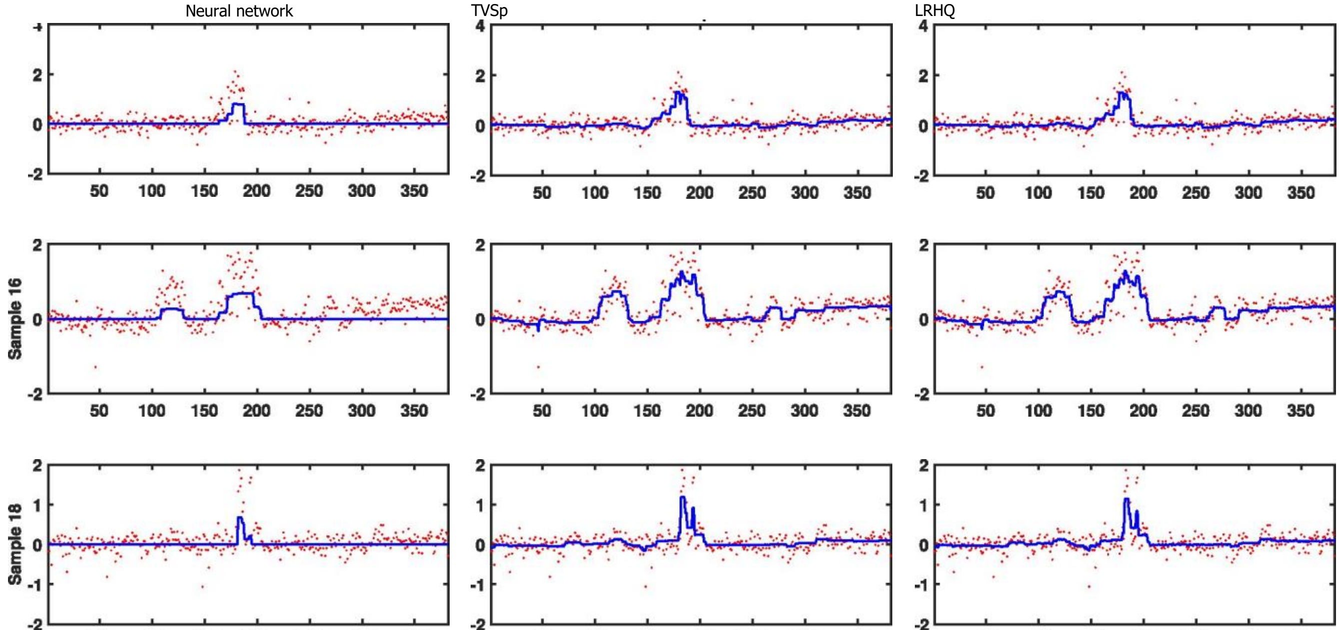

Fig. 5. Recovery of three samples from the Pollack *et al.* dataset by three methods. Each row is dedicated to a particular sample, while each column corresponds to each method. The methods are the proposed neural network, TVSp [61], and LRHQ [63]. The blue lines are the data recovered by the corresponding methods and the red dots denote the data before recovery.

However, TVSp and LRHQ require the use of several auxiliary variables in order to solve their corresponding optimization problem. As a result, they occupy more space in the main memory compared with the proposed neural network.

To show the efficiency of the proposed neural network in practice, we generate synthesized data with varying lengths. The proposed neural solution is then compared with other methods based on the execution time required for each gen- erated data. Fig. 4 plots the execution time in seconds of the foregoing methods on data with differing lengths.

The neural network is significantly superior to PLA and LRHQ regarding execution time. The neural solution is fur- ther competitive with TVSp, and GFLseg is significantly faster than other methods. The reason for this difference is that the proposed neural network is solely implemented in MATLAB while different parts of TVSp and GFLseg are implemented in C/C++. Generally, the proposed neural network is efficient concerning execution time.

The neural network is further applied to a real aCGH data to verify its efficiency. The Pollack *et al*. [58] is a well-known aCGH dataset on breast cancer which contains 44 samples of 6691 human mapped genes. The dataset was subjected to the neural network, TVSp, and LRHQ, and three recov- ered samples are displayed in Fig. 5. In this figure, each row is dedicated to each sample, and each column corresponds to a method. The red dots in this figure are the real data, and the blue line is the data recovered by the corresponding techniques. Based on this figure, the recovered data by the proposed neural network is way smoother than those of TVSp and LRHQ, even though the neural network has less time and memory complexity in comparison to competing meth- ods. To quantify the difference between methods in terms of smoothness, we gauge the total variation of each sample

Fig. 6. CD diagram to compare methods with respect to the smoothness of samples on the Pollack *et al.* dataset.

from the Pollack *et al*. dataset and compare the average of the total variation of methods using the Friedman test and the corresponding *post-hoc* analysis with Nemenyi's correc- tion method [65]. The null hypothesis of the Friedman test was rejected with p-value was close to zero. We further visualize the critical difference (CD) diagram to compare the methods based on the smoothness. Fig. 6 plots the rank of each method obtained by the Friemdan test; the lower the rank, the better the method. Also, the methods that are not significantly different from each other are connected with red dots. According to this figure, GFLseg and the proposed neural network are identical top methods in terms of smooth recovery, followed by TVSp, LRHQ, and PLA. The reason that GFLSeg performs well is that, rather than conducting the recovery, it first finds the mutated points and then connects the points by a line. Thus, it has a very smooth recovery. This experiment corroborates the efficiency of the neural network in real-world situations.

D. Trend Filtering

In this section, the proposed neural network in (18) is applied to a fictitious dataset for the so called trend filtering, and its results are compared with HP filtering [66].

The synthesized data is created based on the following equation:

$t = 1, 2, \ldots, n$

$$X_{t+1} = x_t + V_t, \qquad (21)$$

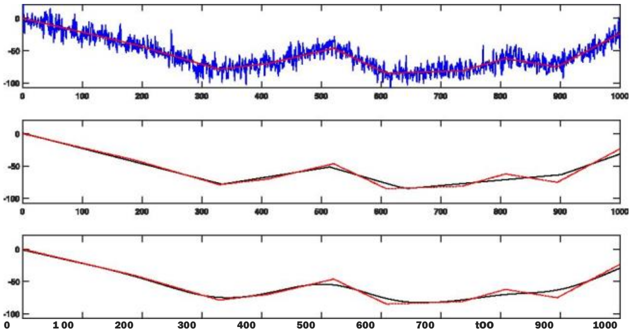

Fig. 7. Detection of trends in a synthesized time series. The top panel plots the noise-free data and noise with red and blue lines, respectively. The black lines in the middle and bottom panels show the recovered data by the proposed neural network and the HP filtering, respectively, and display the noise-free data with red dots.

where x_t is the data at the time t, v_t is the trend slope, and $x_0 = 0$. The slope v_t is generated based on the simple Markov process in a way that with the probability of p, $v_{t+\backslash} = v_t$, and v_t is selected from the interval [—b, *b]* otherwise.

The noise-free data is then required to contaminate with a random noise. In this experiment, we use an additive Gaussian noise to corrupt data based on the following equation [43]:

$$yt = xt + et \tag{22}$$

where e_t is a zero-mean Gaussian noise with standard devia- tion o. It is now required to specify the fixed parameters b, p, o, and n. For this experiment, these values are set as

$$n = 1000, p = 0.985, \qquad o = 10, b = 0.3. \tag{23}$$

The k_2 is also set to 100 for both the HP filtering and the proposed neural network in (18). The real data and the noise generated to corrupt it are plotted at the top panel of Fig. 7 with red and blue lines, respectively. The generated corrupted data were subjected to the proposed neural network and the HP filtering to conduct the recovery and discover the underlying trend in the fictitious time series. The middle and bottom panel in Fig. 7 display the recovered time series by the proposed neural network and the HP filtering, respectively. The black lines are the recovered data and the red dots are the real data before corruption. According to this figure, the neural network could successfully detect four kink points out of six. Similarly, the HP filtering could also detect the same number of kink points. The difference is, however, the recovered data by the HP filtering is curvy while the neural network estimation is angular.

VII. Conclusion

This paper presented a recurrent neural network for FLSA. The proposed neural network was assured to be stable in the sense of Lyapunov and converge globally to the optimal solu- tion of the problem. The network was also simplified to solve several real problems, including total variation denoising, trend filtering, and weighted total variation problem. The experi- ments illustrated the reasonable performance of the proposed neural solution.

The proposed neural network can also be applied to several other problems as well. For instance, the total variation-regularized problem has diverse applications, such as image restoration, image deblurring, and hot-spot detection. The neu- ral network is thus possible to solve these problems as well, which is left for the future research. Also, as Algorithm 1 suggests, the neural network can be executed highly in parallel. Therefore, another avenue for the future research is to take advantage of the graphics processing units (GPUs) and execute the program in parallel with those technologies.

Acknowledgment

The author would like to thank the anonymous reviewers for their constructive comments. He would also like to thank the efforts of A. A. Atashin, M.Sc., for implementing the proposed neural network.

References

[1] D. L. Donoho, “De-noising by soft-thresholding,” *IEEE Trans. Inf. Theory*, vol. 41, no. 3, pp. 613-627, May 1995.

[2] L. I. Rudin, S. Osher, and E. Fatemi, “Nonlinear total variation based noise removal algorithms,” *Physica D Nonlin. Phenom.*, vol. 60, nos. 1^, pp. 259-268, 1992.

[3] J. J. Hopfield and D. W. Tank, “‘Neural’ computation of decisions in optimization problems,”*Biol. Cybern.*, vol. 52, no. 3, pp. 141-152, 1985.

[4] D. Tank and J. J. Hopfield, “Simple ‘neural’ optimization networks: An A/D converter, signal decision circuit, and a linear programming circuit,” *IEEE Trans. Circuits Syst.*, vol. 33, no. 5, pp. 533-541, May 1986.

[5] A. Bouzerdoum and T. R. Pattison, “Neural network for quadratic optimization with bound constraints,” *IEEE Trans. Neural Netw.*, vol. 4, no. 2, pp. 293-304, Mar. 1993.

[6] S. Zhang and A. Constantinides, “Lagrange programming neural networks,” *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.*, vol. 39, no. 7, pp. 441-452, Jul. 1992.

[7] S. Qin, X. Yang, X. Xue, and J. Song, “A one-layer recurrent neu- ral network for pseudoconvex optimization problems with equality and inequality constraints,” *IEEE Trans. Cybern.*, vol. 47, no. 10, pp. 3063-3074, Oct. 2017.

[8] Q. Liu, C. Dang, and T. Huang, “A one-layer recurrent neural network for real-time portfolio optimization with probability criterion,” *IEEE Trans. Cybern.*, vol. 43, no. 1, pp. 14-23, Feb. 2013.

[9] M. J. Perez-Ilzarbe, “New discrete-time recurrent neural network proposal for quadratic optimization with general linear constraints,” *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 24, no. 2, pp. 322-328, Feb. 2013.

[10] Q. Liu and J. Cao, “A recurrent neural network based on projection operator for extended general variational inequalities,” *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 40, no. 3, pp. 928-938, Jun. 2010.

[11] Y. Xia, H. Leung, and J. Wang, “A projection neural network and its application to constrained optimization problems,” *IEEE Trans. Circuits Syst. I, Fundam. Theory Appl.*, vol. 49, no. 4, pp. 447-458, Apr. 2002.

[12] C.-F. Juang and Y.-T. Yeh, “Multiobjective evolution of biped robot gaits using advanced continuous ant-colony optimized recurrent neu- ral networks,” *IEEE Trans. Cybern.*, vol. 48, no. 6, pp. 1910-1922, Jun. 2018.

[13] X. Le, S. Chen, Z. Yan, and J. Xi, “A neurodynamic approach to dis- tributed optimization with globally coupled constraints,” *IEEE Trans. Cybern.*, vol. 48, no. 11, pp. 3149-3158, Nov. 2018.

[14] M. Eshaghnezhad, S. Effati, and A. Mansoori, “A neurodynamic model to solve nonlinear pseudo-monotone projection equation and its applica- tions,” *IEEE Trans. Cybern.*, vol. 47, no. 10, pp. 3050-3062, Oct. 2017.

[15] Y. Lu, D. Li, Z. Xu, and Y. Xi, “Convergence analysis and digital implementation of a discrete-time neural network for model predictive control,” *IEEE Trans. Ind. Electron.*, vol. 61, no. 12, pp. 7035-7045, Dec. 2014.

[16] S. Qin, X. Le, and J. Wang, “A neurodynamic optimization approach to bilevel quadratic programming,” *IEEE Trans. Neural Netw. Learn. Syst.* , vol. 28, no. 11, pp. 2580-2591, Nov. 2017.

[17] Y. Xia, G. Feng, and J. Wang, “A recurrent neural network with expo- nential convergence for solving convex quadratic program and related linear piecewise equations,”*NeuralNetw.*, vol. 17, no. 7, pp. 1003-1015, 2004.

[18] G. Costantini, R. Perfetti, and M. Todisco, “Quasi-Lagrangian neural network for convex quadratic optimization,” *IEEE Trans. Neural Netw.* , vol. 19, no. 10, pp. 1804-1809, Oct. 2008.

[19] X. Hu and J. Wang, “An improved dual neural network for solving a class of

quadratic programming problems and its k-winners-take-all application," *IEEE Trans. Neural Netw.*, vol. 19, no. 12, pp. 2022-2031, Dec. 2008.

[20] X. Gao and L.-Z. Liao, "A new one-layer neural network for linear and quadratic programming," *IEEE Trans. Neural Netw.*, vol. 21, no. 6, pp. 918-929, Jun. 2010.

[21] Q. Liu and J. Wang, "A one-layer recurrent neural network with a discontinuous hard-limiting activation function for quadratic programming," *IEEE Trans. Neural Netw.*, vol. 19, no. 4, pp. 558-570, Apr. 2008.

[22] Y. Xia, "An extended projection neural network for constrained optimization," *Neural Comput.*, vol. 16, no. 4, pp. 863-883, Apr. 2004.

[23] Y. Xia and J. Wang, "A one-layer recurrent neural network for support vector machine learning," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 34, no. 2, pp. 1261-1269, Apr. 2004.

[24] M. Mohammadi, S. H. Mousavi, and S. Effati, "Generalized variant support vector machine," *IEEE Trans. Syst., Man, Cybern., Syst.*, to be published.

[25] Y. Xia, H. Leung, N. Xie, and E. Bossé, "A new regression estimator with neural network realization," *IEEE Trans. Signal Process.* , vol. 53, no. 2, pp. 672-685, Feb. 2005.

[26] Y. Xia and J. Wang, "Robust regression estimation based on low- dimensional recurrent neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 12, pp. 5935-5946, Dec. 2018.

[27] Y. Xia and J. Wang, "A dual neural network for kinematic control of redundant robot manipulators," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 31, no. 1, pp. 147-154, Feb. 2001.

[28] Y. Xia, C. Sun, and W. X. Zheng, "Discrete-time neural network for fast solving large linear L_1 estimation problems and its application to image restoration," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 5, pp. 812-820, May 2012.

[29] Y. Xia and M. S. Kamel, "Novel cooperative neural fusion algorithms for image restoration and image fusion," *IEEE Trans. Image Process.* , vol. 16, no. 2, pp. 367-381, Feb. 2007.

[30] J. Fan and J. Wang, "A collective neurodynamic optimization approach to nonnegative matrix factorization," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 10, pp. 2344-2356, Oct. 2017.

[31] Z. Guo and J. Wang, "A neurodynamic optimization approach to con- strained sparsity maximization based on alternative objective functions," in *Proc. IEEE Int. Joint Conf Neural Netw. (IJCNN)*, 2010, pp. 1-8.

[32] C. Guo and Q. Yang, "A neurodynamic optimization method for recovery of compressive sensed signals with globally converged solution approx- imating to l_0 minimization," *IEEE Trans. Neural Netw. Learn. Syst.* , vol. 26, no. 7, pp. 1363-1374, Jul. 2015.

[33] B. Xu, Q. Liu, and T. Huang, "A discrete-time projection neural network for sparse signal reconstruction with application to face recognition," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 1, pp. 151-162, Jan. 2019.

[34] M. Mohammadi, Y.-H. Tan, W. Hofman, and S. H. Mousavi, "A novel one-layer recurrent neural network for the l_1-regularized least square problem," *Neurocomputing*, vol. 315, pp. 135-144, Nov. 2018.

[35] M. Mohammadi and A. Mansoori, "A projection neural network for identifying copy number variants," *IEEE J. Biomed. Health Informat.* , to be published.

[36] Q. Liu and J. Wang, "L_1-minimization algorithms for sparse signal reconstruction based on a projection neural network," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 3, pp. 698-707, Mar. 2016.

[37] L. Wang, Y. You, and H. Lian, "A simple and efficient algorithm for fused lasso signal approximator with convex loss function," *Comput. Stat.*, vol. 28, no. 4, pp. 1699-1714, 2013.

[38] Y.-X. Wang, J. Sharpnack, A. J. Smola, and R. J. Tibshirani, "Trend filtering on graphs," *J. Mach. Learn. Res.*, vol. 17, no. 1, pp. 3651-3691, 2016.

[39] J. Liu, L. Yuan, and J. Ye, "An efficient algorithm for a class of fused lasso problems," in *Proc. 16th ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, 2010, pp. 323-332.

[40] D. Yu, J.-H. Won, T. Lee, J. Lim, and S. Yoon, "High-dimensional fused lasso regression using majorization-minimization and parallel processing," *J. Comput. Graph. Stat.*, vol. 24, no. 1, pp. 121-153, 2015.

[41] L. Condat, "A direct algorithm for 1-D total variation denoising," *IEEE Signal Process. Lett.*, vol. 20, no. 11, pp. 1054-1057, Nov. 2013.

[42] M. Wytock, S. Sra, and J. Z. Kolter, "Fast Newton methods for the group fused lasso," in *Proc. UAI*, 2014, pp. 888-897.

[43] S.-J. Kim, K. Koh, S. Boyd, and D. Gorinevsky, "L_1 trend filtering," *SIAMRev.*, vol. 51, no. 2, pp. 339-360, 2009.

[44] M. S. Bazaraa, H. D. Sherali, and C. M. Shetty, *Nonlinear Programming: Theory and Algorithms.* New York, NY, USA: Wiley, 2013.

[45] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods*, vol. 23. Englewood Cliffs, NJ, USA: Prentice-Hall, 1989.

[46] D. Kinderlehrer and G. Stampacchia, *An Introduction to Variational Inequalities and Their Applications*, vol. 31. Philadelphia, PA, USA: SIAM, 1980.

[47] X. Hu and J. Wang, "A recurrent neural network for solving a class of general variational inequalities," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 37, no. 3, pp. 528-539, Jun. 2007.

[48] J. K. Hale, "Functional differential equations," in *Analytic Theory of Differential Equations*. New York, NY, USA: Springer, 1971, pp. 9-22.

[49] Y. S. Xia, "Further results on global convergence and stability of globally projected dynamical systems," *J. Optim. Theory Appl.*, vol. 122, no. 3, pp. 627-649, 2004.

[50] J. P LaSalle, *The Stability of Dynamical Systems*, vol. 25. Philadelphia, PA, USA: SIAM, 1976.

[51] X. Hu and J. Wang, "Solving pseudomonotone variational inequalities and pseudoconvex optimization problems using the projection neural network," *IEEE Trans. Neural Netw.*, vol. 17, no. 6, pp. 1487-1499, Nov. 2006.

[52] Y. Xia and J. Wang, "A recurrent neural network for solving lin- ear projection equations," *Neural Netw.*, vol. 13, no. 3, pp. 337-350, 2000.

[53] F. P Such, V. Madhavan, E. Conti, J. Lehman, K. O. Stanley, and J. Clune, "Deep neuroevolution: Genetic algorithms are a competitive alternative for training deep neural networks for reinforcement learning," *arXiv preprint arXiv:1712.06567*, 2017.

[54] T. Bartz-Beielstein, C. W. G. Lasarczyk, and M. Preuss, "Sequential parameter optimization," in *Proc. IEEE Congr. Evol. Comput.*, vol. 1, 2005, pp. 773-780.

[55] P. C. Hansen, "Analysis of discrete ill-posed problems by means of the L-curve," *SIAM Rev.*, vol. 34, no. 4, pp. 561-580, 1992.

[56] S. D. Babacan, R. Molina, and A. K. Katsaggelos, "Parameter esti- mation in TV image restoration using variational distribution approx- imation," *IEEE Trans. Image Process.*, vol. 17, no. 3, pp. 326-339, Mar. 2008.

[57] G. Nowak, T. Hastie, J. R. Pollack, and R. Tibshirani, "A fused lasso latent feature model for analyzing multi-sample aCGH data," *Biostatistics*, vol. 12, no. 4, pp. 776-791, 2011.

[58] J. R. Pollack *et al.*, "Microarray analysis reveals a major direct role of DNA copy number alteration in the transcriptional program of human breast tumors," *Proc. Nat. Acad. Sci. USA*, vol. 99, no. 20, pp. 12963-12968, 2002.

[59] R. Tibshirani and P. Wang, "Spatial smoothing and hot spot detection for CGH data using the fused lasso," *Biostatistics*, vol. 9, no. 1, pp. 18-29, 2007.

[60] R. Tibshirani, M. Saunders, S. Rosset, J. Zhu, and K. Knight, "Sparsity and smoothness via the fused lasso," *J. Roy. Stat. Soc. B (Stat. Methodol.)*, vol. 67, no. 1, pp. 91-108, 2005.

[61] X. Zhou, C. Yang, X. Wan, H. Zhao, and W. Yu, "Multisample aCGH data analysis via total variation and spectral regularization," *IEEE/ACM Trans. Comput. Biol. Bioinf.*, vol. 10, no. 1, pp. 230-235, Jan./Feb. 2013.

[62] X. Zhou, J. Liu, X. Wan, and W. Yu, "Piecewise-constant and low-rank approximation for identification of recurrent copy num- ber variations," *Bioinformatics*, vol. 30, no. 14, pp. 1943-1949, 2014.

[63] M. Mohammadi, G. A. Hodtani, and M. Yassi, "A robust correntropy- based method for analyzing multisample aCGH data," *Genomics*, vol. 106, no. 5, pp. 257-264, 2015.

[64] K. Bleakley and J.-P. Vert, "The group fused lasso for multiple change-point detection," *arXiv preprint arXiv:1106.4199*, 2011.

[65] M. Mohammadi, W. Hofman, and Y.-H. Tan, "A comparative study of ontology matching systems via inferential statistics," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 4, pp. 615-628, Apr. 2019.

[66] R. J. Hodrick and E. C. Prescott, "Postwar U.S. business cycles: An empirical investigation," *J. Money Credit Banking*, vol. 29, no. 1, pp. 1-16, 1997.