*mathematics*

MDPI

*Article*

# A Lagrange Programming Neural Network Approach with an $\ell_0$-Norm Sparsity Measurement for Sparse Recovery and Its Circuit Realization

**Hao Wang [1], Ruibin Feng [2], Chi-Sing Leung [2,*], Hau Ping Chan [2] and Anthony G. Constantinides [3]**

[1] College of Electronics and Information Engineering, Shenzhen University, Shenzhen 518060, China
[2] Department of Electrical Engineering, City University of Hong Kong, Hong Kong
[3] Department of Electrical and Electronic Engineering, Imperial College, London SW7 2BX, UK
* Correspondence: eeleungc@cityu.edu.hk

**Abstract:** Many analog neural network approaches for sparse recovery were based on using $\ell_1$-norm as the surrogate of $\ell_0$-norm. This paper proposes an analog neural network model, namely the Lagrange programming neural network with $\ell_p$ objective and quadratic constraint (LPNN-LPQC), with an $\ell_0$-norm sparsity measurement for solving the constrained basis pursuit denoise (CBPDN) problem. As the $\ell_0$-norm is non-differentiable, we first use a differentiable $\ell_p$-norm-like function to approximate the $\ell_0$-norm. However, this $\ell_p$-norm-like function does not have an explicit expression and, thus, we use the locally competitive algorithm (LCA) concept to handle the nonexistence of the explicit expression. With the LCA approach, the dynamics are defined by the internal state vector. In the proposed model, the thresholding elements are not conventional analog elements in analog optimization. This paper also proposes a circuit realization for the thresholding elements. In the theoretical side, we prove that the equilibrium points of our proposed method satisfy Karush Kuhn Tucker (KKT) conditions of the approximated CBPDN problem, and that the equilibrium points of our proposed method are asymptotically stable. We perform a large scale simulation on various algorithms and analog models. Simulation results show that the proposed algorithm is better than or comparable to several state-of-art numerical algorithms, and that it is better than state-of-art analog neural models.

**Keywords:** analog neural networks; LPNN; optimization; real-time solution

**MSC:** 94A12; 68T01; 68T07



**Citation:** Wang, H.; Feng, R.; Leung, C.-S.; Chan, H.P.; Constantinides, A.G. A Lagrange Programming Neural Network Approach with an $\ell_0$-Norm Sparsity Measurement for Sparse Recovery and Its Circuit Realization. *Mathematics* **2022**, *10*, 4801. https://doi.org/10.3390/math10244801

Academic Editors: Araceli Queiruga-Dios, Fatih Yilmaz, Ion Mierlus-Mazilu, Deolinda M. L. Dias Rasteiro and Jesús Martín Vaquero

Received: 25 November 2022
Accepted: 13 December 2022
Published: 16 December 2022



## 1. Introduction

### 1.1. Background

The last few decades have seen increasingly rapid advances in analog neural networks for solving optimization problems. In an analog neural network, the state transitions of neurons are governed by some differential equations. After the dynamics of the network converge to an equilibrium point, the solution of the problem is obtained from the state of the neurons. From many neural pioneers [1–6], this approach is very attractive when real-time solutions are required.

The research of the analog neural network approach could be dated back to the 1980s [2]. One of the earliest analog networks is the Hopfield model [2]. Early applications of the Hopfield model are analog-to-digital conversion and the traveling salesman problem. Later, many analog models [3–7] for various optimization problems were proposed. Over the last decade, many new applications of the analog neural modes were investigated, including image processing, sparse approximation [5,8], mobile target localization [9,10], and feature selection [6]. Recently, several analog techniques [5,8,11–13] were designed for solving sparse recovery problems.

In sparse recovery [8,14–18], the aim is to recover an unknown sparse vector $\boldsymbol{x} \in \mathcal{R}^n$ from an observation vector $\boldsymbol{b} \in \mathcal{R}^m$. For many real life signals, their internal representations are with the sparse property [19]. For example, audio signals are approximately sparse in the time-frequency domain [20]. Sparse recovery techniques can be in many signal processing applications. For instance, we use can sparse recovery techniques for image restoration [18,21,22]. Additionally, they can be used to remove the stripes in hyperspectral images [23]. In the inverse synthetic aperture radar (ISAR) application [24,25], a high-quality image of an object can be obtained from the Fourier transformed signal based on sparse recovery techniques. Another application of sparsity recovery is to process electrocardiogram (ECG) signal for classification of various heart diseases [26].

One of sparse recovery problems is the following $\ell_0$-norm optimization problem:

$$\min_{\boldsymbol{x}} \|\boldsymbol{x}\|_0, \text{ subject to } \boldsymbol{b} = \boldsymbol{\Phi x}, \tag{1}$$

where $\boldsymbol{\Phi} \in \mathcal{R}^{m \times n}$ is the measurement matrix. When there are some measurement noise in $\boldsymbol{b}$, the problem becomes the constrained basis pursuit denoise (CBPDN) problem:

$$\min_{\boldsymbol{x}} \|\boldsymbol{x}\|_0, \text{ subject to } ||\boldsymbol{b} - \boldsymbol{\Phi x}||_2^2 \leq m\theta^2, \tag{2}$$

where $\theta > 0$ is the standard deviation of observation noise. Since the $\ell_0$-norm is difficult to handle, we usually use the $\ell_1$-norm to replace the $\ell_0$-norm. The problems, stated in (1) and (2), become

$$\min_{\boldsymbol{x}} \|\boldsymbol{x}\|_1, \text{ subject to } \boldsymbol{b} = \boldsymbol{\Phi x}, \text{ and } \min_{\boldsymbol{x}} \|\boldsymbol{x}\|_1, \text{ subject to } ||\boldsymbol{b} - \boldsymbol{\Phi x}||_2^2 \leq m\theta^2, \tag{3}$$

respectively. In the last two decades, many $\ell_1$-norm based numerical algorithms were proposed, such as BPDN-interior [27] and Homotopy [28]. In addition, elegant implementation packages [29,30] are available, such as SPGL1 [30].

Although the aforementioned $\ell_1$-norm relaxation approaches were well studied, they have some drawbacks. For instance, in the BPDN-interior algorithm, the solution vector contain many small non-zero elements [31]. As mentioned in [32,33], $\ell_p$-norm ($0 < p < 1$) is a better choice for replacing $\ell_0$-norm. However, $\ell_p$-norm is a non-convex function, which introduces complex behaviours in the problem-solving process. Therefore, we can use some approximation functions to replace the $\ell_p$-norm term, such as minimax concave penalty (MCP) function [34,35]. In addition, there are other methods, which directly handle the $\ell_0$-norm. They are normalized iterative hard threshold (NIHT) method [36], approximate message passing (AMP) [37], $\ell_0$-norm zero attraction projection ($l_0$-ZAP) [38], $\ell_0$-norm alternating direction method of multipliers ($\ell_0$-ADMM) [39,40], and expectation-conditional maximization either (ECME) [41]. All the mentioned $\ell_p$-norm or $\ell_0$-norm techniques in this paragraph are digital numerical algorithms.

### *1.2. Motivation*

Apart from using the numerical methods to solve the sparse recovery problem, we can consider using the analog neural approach for solving sparse recovery problems [5,8,11–13,21]. However, those analog models in [5,8,11–13,21] were developed based on the $\ell_1$-norm relaxation techniques. Since the $\ell_1$-norm is a surrogate function of the $\ell_0$-norm only, directly using the $\ell_0$-norm or a $\ell_0$-like norm usually leads to a better performance. There is indeed a $\ell_0$-norm-based analog model, namely local competition algorithm (LCA) [31]. However, it was designed for unconstrained sparse recovery problems only.

As directly working with the $\ell_p$-norm or $\ell_0$-norm usually results in better performance, it is interesting to develop some $\ell_p$-norm or $\ell_0$-norm analog models for sparse recovery problems with constraints. Another shortcoming of existing $\ell_1$-norm relaxation neural models [8,11–13] is that the corresponding circuit realizations, especially the circuit for projection and thresholding operations, were not discussed.

### *1.3. Contribution and Organization*

This paper focuses on using the analog technique to solve the CBPDN problem with the $\ell_0$-norm objective, stated in (2). Strictly speaking, the $\ell_0$-norm is not a norm and is not differentiable. These properties create difficulties for constructing the analog model for the CBPDN problem.

The paper proposes a $\ell_p$-norm-like function for representing the sparsity measurement. The proposed $\ell_p$-norm-like function is differentiable. We then apply the Lagrange programming neural network (LPNN) framework [8,42] for solving the CBPDN problem. The proposed $\ell_p$-norm-like function does not have a simple expression, but its derivative has. Hence, we borrow the internal state concept from the LCA [31] to construct a LPNN model for the CBPDN problem. We call our model "LPNN with $\ell_p$ objective and quadratic constraint (LPNN-LPQC)".

In developing an analog neural model, one of difficulties is to analyze the behaviour of the analog neural network dynamics, especially, for non-convex objective function without an explicit expression. The paper discusses the stability of the proposed LPNN-LPQC model. We theoretically prove that the equilibrium points of our proposed LPNN-LPQC satisfy Karush Kuhn Tucker (KKT) conditions of the approximated CBPDN problem. We use the term "approximated" because we have applied an approximation for the $\ell_0$-norm. In addition, we prove that the equilibrium points of our proposed method are asymptotically stable.

Unlike some existing analog neural results which do not discuss the circuit realization [8,11–13], this paper also discusses the circuit realization of the proposed model. In particular, the detailed circuit design for the thresholding element is given. We then use the MATLAB Simulink to verify our design. In the verification, we find that the MATLAB Simulink results are nearly the same as the corresponding results obtained from the discretized dynamic equations.

This paper also presents a large scale simulation. The simulation result shows that the proposed LPNN-LPQC is better than or comparable to several state-of-art digital numerical algorithms, and that is better than state-of-art analog neural networks.

The rest of this paper is organized as follows. Backgrounds on the LPNN and LCA models are described in Section 2. In Section 3, the proposed LPNN-LPQC is developed. Section 4 discusses the circuit realization of the thresholding element. Section 5 discusses the LPNN-LPQC's stability. Simulation results and comparisons with state-of-art numerical algorithms are provided in Section 6. Finally, conclusions are drawn in Section 7.

## 2. LPNN Framework and LCA

### *2.1. LPNN*

The LPNN technique [42] can be used in many applications, such as locating a target in a radar system [9] and $\ell_1$-norm-based sparse recovery [8], and ellipse fitting [43]. It was developed for solving a general non-linear constrained optimization problem:

$$\min_{\boldsymbol{x}} \phi(\boldsymbol{x}), \text{ subject to } \boldsymbol{h}(\boldsymbol{x}) = \boldsymbol{0}\,, \tag{4}$$

where $\phi : \mathcal{R}^n \to \mathcal{R}$ is the objective, and $\boldsymbol{h} : \mathcal{R}^n \to \mathcal{R}^m$ ($m < n$) represents $m$ equality constraints. In the LPNN approach, we first set up a Lagrangian function, given by

$$\mathcal{L}_{eq} = \phi(\boldsymbol{x}) + \boldsymbol{\lambda}^\top \boldsymbol{h}(\boldsymbol{x})\,, \tag{5}$$

where $\boldsymbol{\lambda} = [\lambda_1, \cdots, \lambda_m]^\top$ is the Lagrange multiplier vector. An LPNN has two classes of neurons: variable neurons for holding $\boldsymbol{x}$ and Lagrange neurons for holding $\boldsymbol{\lambda}$. Its neural dynamics are

$$\mu \frac{d\boldsymbol{x}}{dt} = -\frac{\partial \mathcal{L}_{eq}}{\partial \boldsymbol{x}}, \text{ and } \mu \frac{d\boldsymbol{\lambda}}{dt} = \frac{\partial \mathcal{L}_{eq}}{\partial \boldsymbol{\lambda}}, \tag{6}$$

where $\mu$ is the characteristic time constant. Without loss of generality, we consider that $\mu$ is equal to 1. With (6), when some mild conditions [42] are held, the network settles down at a stable state. The main restriction of using the LPNN framework is that $\phi(\boldsymbol{x})$ and $\boldsymbol{h}(\boldsymbol{x})$ should be differentiable.

### 2.2. LCA

The LCA [31,44] aims at solving the following unconstrained optimization problem:

$$\min_{x} \mathcal{L}(\boldsymbol{x}) = \frac{1}{2}||\boldsymbol{b} - \boldsymbol{\Phi x}||_2^2 + \kappa \sum_{i=1}^{n} S_{\alpha,\gamma,\kappa}(x_i), \tag{7}$$

where $\kappa \sum_{i=1}^{n} S_{\alpha,\gamma,\kappa}(x_i)$ is a penalty term to improve the sparsity of the resultant $\boldsymbol{x}$. The function $\kappa S_{\alpha,\gamma,\kappa}(x)$ does not have an exact expression. Instead, it is defined by introducing an internal state vector $\boldsymbol{u}$. To define $\kappa S_{\alpha,\gamma,\kappa}(x)$, a thresholding function on $\boldsymbol{u}$ is first introduced

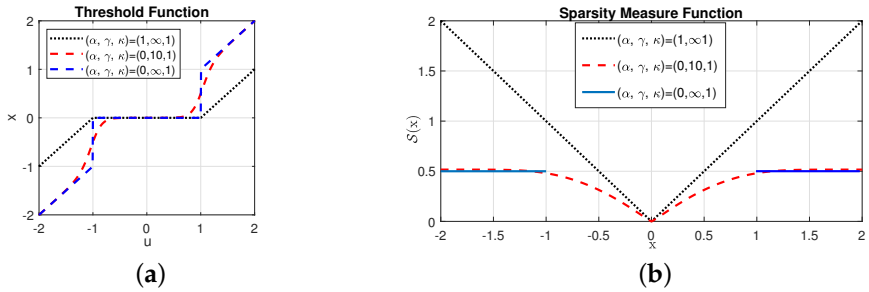
$$x_i = T_{\alpha,\gamma,\kappa}(u_i) = \text{sign}(u_i)\frac{|u_i| - \alpha\kappa}{1 + \exp\left(-\gamma(|u_i| - \kappa)\right)}, \tag{8}$$

where $\kappa > 0$ is the threshold and is related to the magnitude of the non-zero elements, $\gamma \in (0, +\infty)$ controls the threshold transition rate or slope around threshold, and $\alpha \in [0, 1]$ indicates adjustment fraction after the internal neuron across threshold. Figure 1a illustrates the shape of $T_{\alpha,\gamma,\kappa}(u_i)$ under various settings. With $(\alpha, \gamma, \kappa) = (1, \infty, \kappa)$, the threshold function $T_{\alpha,\gamma,\kappa}(\cdot)$ is the well-known soft threshold function. Given a threshold function, the penalty function $S_{\alpha,\gamma,\kappa}(x_i)$ is then defined by

$$\kappa \partial S_{\alpha,\gamma,\kappa}(x_i) = u_i - x_i = u_i - T_{\alpha,\gamma,\kappa}(u_i), \tag{9}$$

$$S_{\alpha,\gamma,\kappa}(0) = 0, \tag{10}$$

where $\partial S_{\alpha,\gamma,\kappa}(x_i)$ is the gradient or sub-gradient of $S_{\alpha,\gamma,\kappa}(x_i)$.




(**a**) (**b**)

**Figure 1.** Threshold function and sparsity measure function. (**a**) The shape of $T_{\alpha,\gamma,\kappa}(u_i)$ under various settings. (**b**) Sparsity measure function. For $\alpha = 0$, $\gamma \to \infty$ and $\kappa = 1$, the value of $x$ cannot be in the range of $(0, 1)$ based on the property of the ideal thresholding function $T_{0,\infty,1}(u)$.

In the vector form, (9) is written as

$$\kappa \partial\Big(\sum_{i=1}^{n} S\alpha, \gamma, \kappa(x_i)\Big) = \boldsymbol{u} - \boldsymbol{x} = \boldsymbol{u} - \boldsymbol{T}_{\alpha,\gamma,\kappa}(\boldsymbol{u}), \tag{11}$$

where $\boldsymbol{T}_{\alpha,\gamma,\kappa}(\boldsymbol{u}) = [T_{\alpha,\gamma,\kappa}(u_1), \cdots, T_{\alpha,\gamma,\kappa}(u_n)]^\top$.

From (9), $S_{\alpha,\gamma,\kappa}(x_i)$ is defined by the derivative. Hence, in general, there is no explicit expression for $S_{\alpha,\gamma,\kappa}(x_i)$. To visualize them, we should use numerical integration. Figure 1b shows the sparsity measure function $S_{\alpha,\gamma,\kappa}(x_i)$ under various parameter settings. As shown in the figure, in some cases, such as a large $\gamma$ value, the sparsity measure function $\boldsymbol{S}_{0,\gamma,\kappa}(\boldsymbol{x})$ is closed to $\ell_p$-norm ($0 < p < 1$). Hence, $\boldsymbol{S}_{0,\gamma,\kappa}(\boldsymbol{x})$ is called the $\ell_p$-norm-like sparsity function.

With the internal state concept and (9), LCA defines the dynamics on $\boldsymbol{u}$ (rather than on $\boldsymbol{x}$) as

$$\frac{d\boldsymbol{u}}{dt} = -\frac{\partial \mathcal{L}_{eq}}{\partial \boldsymbol{x}} = -\kappa\partial\Big(\sum_{i=1}^{n} S\alpha, \gamma, \kappa(x_i)\Big) + \boldsymbol{\Phi}^{\top}(\boldsymbol{b} - \boldsymbol{\Phi}\boldsymbol{x}) = -\boldsymbol{u} + \boldsymbol{x} + \boldsymbol{\Phi}^{\top}(\boldsymbol{b} - \boldsymbol{\Phi}\boldsymbol{x}). \quad (12)$$

## 3. LPNN-LPQC Model

For the proposed model, $\alpha = 0$, $\kappa = \frac{1}{2}$, and $\gamma$ is a large positive number. The meaning of $\kappa = \frac{1}{2}$ is that the magnitude of the non-zero elements in the resultant $\boldsymbol{x}$ should be greater than $\frac{1}{2}$.

For simplicity, we use notation $S(x)$ to replace $S_{0,\gamma,\frac{1}{2}}(x)$. We consider the following LPQC problem:

$$\min_{x} \frac{1}{2}\sum_{i=1}^{n} S(x_i), \text{ subject to } ||\boldsymbol{b} - \boldsymbol{\Phi}\boldsymbol{x}||_2^2 \leq m\theta^2, \quad (13)$$

We first discuss some properties of $S(x)$. Afterwards, we derive the LPNN-LPQC model and perform the stability analysis on the proposed model.

### *3.1. $\ell_p$-Norm-like Sparsity Measure Function*

Recall that from Figure 1b, for large $\gamma$, $S(x)$ is similar to the $\ell_p$-norm. When $\alpha = 0$ and $\kappa = \frac{1}{2}$, the threshold function becomes

$$x = T(u) = \text{sign}(u)\frac{|u|}{1 + \exp\left(-\gamma(|u| - \frac{1}{2})\right)}. \quad (14)$$

Sparsity measurement function $S(x)$ is defined by its derivative:

$$\frac{1}{2}\partial S(x) = u - x = u - T(u), \quad (15)$$

$$S(0) = 0. \quad (16)$$

Before discussing the properties of $\frac{1}{2}S(x)$ and $T(x)$, we would like to make some remarks. First, the explicit expression of $\frac{1}{2}S(x)$ and $T^{-1}(\cdot)$ are not available, but the value of $\frac{1}{2}\partial S(x)$ is easily obtained from $u$ based on (14). Second, as shown in the rest of this paper, in the implementation of the proposed model, we need to implement $T(u)$ rather than $\frac{1}{2}S(x)$.

We first list a number of properties of $T(u)$ and $\frac{1}{2}S(x)$. From (14) and basic mathematics, we have Property 1.

**Property 1.**

**P1.a:** *$T(u)$ is a continuous odd function.*

**P1.b:** *$T(u)$ is strictly monotonically increasing.*

**P1.c:** *Inverse of $T(u)$ exists.*

**P1.d:** *$T(u)$ is differentiable at everywhere for all real $u$.*

Since $T(u)$ is an odd function and monotonically increasing, from basic mathematics, $S(x)$ has the following properties.

**Property 2.**

**P2.a:** *$\frac{1}{2}S(x)$ is an even function.*

**P2.b:** *$\frac{1}{2}S(|x|)$ is monotonically increasing with respect to $|x|$.*

Additionally, $T(u)$ has the following properties.

**Property 3**. *Define* $f(u) = \frac{1}{1+\exp(-\gamma(|u|-\frac{1}{2}))}$. *Thus, we have* $T(u) = uf(u)$. *For small positive* $\epsilon$, *we have the following properties:*

**P3.a:** *If* $|u| \geq \frac{1}{2} + \frac{1}{\gamma}\log\frac{1}{\epsilon}$, *then* $\frac{1}{1+\epsilon} \leq f(u) < 1$.

**P3.b:** *If* $|u| \leq \frac{1}{2} - \frac{1}{\gamma}\log\frac{1}{\epsilon}$, *then* $0 < f(u) \leq \frac{\epsilon}{1+\epsilon}$.

**P3.c:** *If* $|u| \geq \frac{1}{2} + \frac{1}{\gamma}\log\frac{\gamma}{\epsilon}$, *then* $1 < \frac{dT(u)}{du} \leq 1 + \frac{3}{2}\epsilon$;

**P3.d:** *If* $0 \leq |u| \leq \frac{1}{2} - \frac{1}{\gamma}\log\frac{\gamma}{\epsilon}$, *then* $0 < \frac{dT(u)}{du} \leq \frac{3}{2}\epsilon$.

**Proof.** The proof is exhibited in Appendix A. □

**Remark 1.** **P3.a** *and* **P3.b** *mean that in some regions of* $u$, $T(u)$ *is approximately equal to either* $u$ *or 0. On the contrary, the regions of* $|\frac{1}{2} - \frac{1}{\gamma}\log\frac{1}{\epsilon}| < |u| < |\frac{1}{2} + \frac{1}{\gamma}\log\frac{1}{\epsilon}|$ *are uncertain regions. However, the uncertain regions can be arbitrarily small by choosing a sufficient large* $\gamma$.

With Properties 1–3, we can prove that $\frac{1}{2}S(x)$ is differentiable at everywhere.

**Property 4**. $\frac{1}{2}S(x)$ *is a continuously differentiable function.*

**Proof.** The proof is exhibited in Appendix B. □

Since $\frac{1}{2}S(x)$ is differentiable, in the rest of this paper, we replace "$\partial$" with "$\nabla$" to indicate the partial derivative of $\frac{1}{2}\sum_{i=1}^{n} S(x_i)$, i.e., $\frac{1}{2}\nabla(\sum_{i=1}^{n} S(x_i)) = \boldsymbol{u} - \boldsymbol{x}$. In addition, we discuss two features of $\frac{1}{2}S(\cdot)$, which make $\frac{1}{2}S(\cdot)$ to be a good sparsity measure function.

**Property 5**. *For a given* $\epsilon$ *and sufficient large* $\gamma$,

- *Boundness: for* $|u| \geq \frac{1}{2} + \frac{1}{\gamma}\log\frac{1}{\epsilon}$, $\frac{1}{2}S(x) \approx 1/8$.
- *Sparsity: for* $|u| \leq \frac{1}{2} - \frac{1}{\gamma}\log\frac{1}{\epsilon}$, $\frac{1}{2}S(x) \approx 0$.

**Proof.** The proof is exhibited in Appendix C. □

Note that when $\gamma$ tends to $+\infty$, $S(x)$ becomes the $\ell_0$-norm and $T(u)$ is the hard threshold function.

### *3.2. Properties of LPQC Problem*

In order to analyze the properties of the LPQC problem, stated in (13), we review the KKT necessary conditions for general constrained optimization problems.

**Lemma 1.** *Consider the following non-linear optimization problem:*

$$\min_{\boldsymbol{x}} \phi(\boldsymbol{x}), \text{ subject to } h(\boldsymbol{x}) \leq 0, \tag{17}$$

*where* $\phi : \mathcal{R}^n \to \mathcal{R}$ *is the objective,* $h : \mathcal{R}^n \to \mathcal{R}$ *defines the inequality constraint,* $\phi$ *and* $h$ *are continuously differentiable. If* $\boldsymbol{x}^*$ *is a local optimum, then there exists a constant* $\lambda^*$, *called Lagrange multiplier, such that*

**Stationarity:** $\nabla\phi(\boldsymbol{x}^*) + \lambda^*\nabla h(\boldsymbol{x}^*) = \boldsymbol{0}$,

**Primal feasibility:** $h(\boldsymbol{x}^*) \leq 0$,

**Dual feasibility:** $\lambda^* \geq 0$,

**Complementary slackness:** $\lambda^* h(\boldsymbol{x}^*) = 0$.

With the arm with Lemma 1 and some reasonable assumptions, we could simplify the KKT conditions of the LPQC, stated in (13). The result is summarized in Theorem 1.

**Theorem 1.** *Given that* $||\boldsymbol{b}||_2^2 > m\theta^2$, *and that* $\boldsymbol{x}^*$ *is a local optimum of the optimization problem (13), the KKT conditions become*

$$\boldsymbol{u}^* - \boldsymbol{x}^* - \lambda^* \boldsymbol{\Phi}^\top (\boldsymbol{b} - \boldsymbol{\Phi}\boldsymbol{x}^*) = \boldsymbol{0}, \tag{18a}$$
$$||\boldsymbol{b} - \boldsymbol{\Phi}\boldsymbol{x}^*||_2^2 - m\theta^2 = 0, \tag{18b}$$
$$\lambda^* > 0. \tag{18c}$$

*Note that* $\nabla(\sum_{i=1}^n S(x_i^*)) = \boldsymbol{u}^* - \boldsymbol{x}^*$.

**Proof.** From Lemma 1, the KKT conditions of (13) are

$$\boldsymbol{u}^* - \boldsymbol{x}^* - \lambda^* \boldsymbol{\Phi}^\top (\boldsymbol{b} - \boldsymbol{\Phi}\boldsymbol{x}^*) = \boldsymbol{0}, \tag{19a}$$
$$||\boldsymbol{b} - \boldsymbol{\Phi}\boldsymbol{x}^*||_2^2 - m\theta^2 \leq 0, \tag{19b}$$
$$\lambda^* \geq 0, \tag{19c}$$
$$\lambda^*(||\boldsymbol{b} - \boldsymbol{\Phi}\boldsymbol{x}^*||_2^2 - m\theta^2) = 0. \tag{19d}$$

According to (19c), $\lambda^*$ is either greater than or equal to 0. The proof of $\lambda^* > 0$ is by contradiction. Assume that $\lambda^* = 0$. From (19a), we have $\nabla(\sum_{i=1}^n S(x_i^*)) = \boldsymbol{0}$. Furthermore, from $\frac{1}{2}\nabla(\sum_{i=1}^n S(x_i^*)) = \boldsymbol{u}^* - \boldsymbol{x}^*$ and (14), it is easy to prove that $\boldsymbol{u}^* = \boldsymbol{x}^*$, and that $\boldsymbol{u}^* = \boldsymbol{x}^* = \boldsymbol{0}$. On the other hand, from (19b), when $\boldsymbol{u}^* = \boldsymbol{x}^* = \boldsymbol{0}$, we have $||\boldsymbol{b}||_2^2 - m\theta^2 \leq 0$, which contradicts our earlier assumption $||\boldsymbol{b}||_2^2 > m\theta^2$. Hence, $\lambda^*$ must be greater than 0. In other words, the KKT conditions of (13) become (18). The proof is completed. □

### *3.3. Dynamics of LPNN-LPQC*

In the analog neural approach, we need to design the neural dynamics such that the equilibrium points of the dynamics fulfill the KKT conditions of the problem.

From the LPNN framework, we set up a Lagrangian function:

$$\mathcal{L}_{LPQC} = \frac{1}{2}(\sum_{i=1}^n S(x_i^*)) + \alpha^2(||\boldsymbol{b} - \boldsymbol{\Phi}\boldsymbol{x}||_2^2 - m\theta^2), \tag{20}$$

where $\alpha^2 := \lambda$, which is used to simplify the proof of the Lagrange multiplier $\lambda$ being greater than zero. Based on the concepts of LPNN and LCA, we propose the dynamics of the LPNN-LPQC as

$$\begin{aligned} \frac{d\boldsymbol{u}}{dt} &= -\frac{\partial \mathcal{L}_{LPQC}}{\partial \boldsymbol{x}} = -\frac{1}{2}\nabla(\sum_{i=1}^n S(x_i)) + 2\alpha^2 \boldsymbol{\Phi}^\top(\boldsymbol{b} - \boldsymbol{\Phi}\boldsymbol{x}) \\ &= -\boldsymbol{u} + \boldsymbol{x} + 2\alpha^2 \boldsymbol{\Phi}^\top(\boldsymbol{b} - \boldsymbol{\Phi}\boldsymbol{x}), \end{aligned} \tag{21a}$$
$$\frac{d\alpha}{dt} = \frac{\partial \mathcal{L}_{LPQC}}{\partial \alpha} = 2\alpha(\|\boldsymbol{b} - \boldsymbol{\Phi}\boldsymbol{x}\|_2^2 - m\theta^2). \tag{21b}$$

## 4. Circuit Realization

In the concept of analog neural models for optimization process, one important issue is whether the operations in the dynamic equation can be implemented with analog circuits or not. This section addresses this issue.

### *4.1. Thresholding Element*

From the dynamic equations, stated in (21), there are many conventional analog operations, such as adders [45], integrators [45], multipliers [46,47] and square circuits [48,49]. The circuit realization of those common operations are well discussed in [45–49].

There is an unconventional element in the dynamic equations. It is the thresholding function:

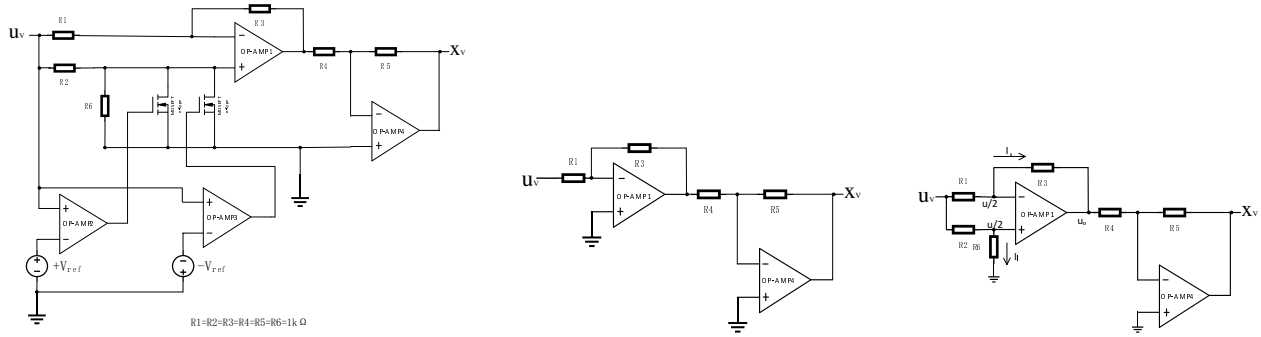
$$x = T(u) = \text{sign}(u)\frac{|u|}{1 + \exp\left(-\gamma(|u| - \frac{1}{2})\right)}.$$

Figure 2a shows a generalized realization of the thresholding element for large $\gamma$. In this generalized realization, the thresholding level is $V_{ref}$, where $V_{ref}$ is a positive number. The detailed function is given by

$$x = T(u) = \text{sign}(u)\frac{|u|}{1 + \exp\left(-\gamma(|u| - V_{ref})\right)}.$$

The two MOSFETs in Figure 2a control the thresholding mode.

If the magnitude $|u|$ of the input is greater than $V_{ref}$, then one of the two MOSFETs is on and the circuit in Figure 2a becomes an equivalent one shown in Figure 2b. Clearly, for this equivalent circuit, the output is equal to input.




(**a**) (**b**) (**c**)

**Figure 2.** (**a**) Circuit for the thresholding function $T(x)$. (**b**) Equivalent circuit of the thresholding function $T(x)$ when the magnitude of input is greater than $V_{ref}$. (**c**) Equivalent circuit of the thresholding function $T(x)$ when the magnitude of input is less than or equal to $V_{ref}$.

On the other hand, in Figure 2a, if the magnitude $|u|$ of the input is less than or equal to $V_{ref}$, then the two MOSFETs are off and we obtain another equivalent circuit shown in Figure 2c. In this case, the inputs of the OP-AMP1 are clamped at $u/2$. Since the two current values, $I_u$ and $I_l$, of the upper and lower paths are equal, the output $u_o$ of the OP-AMP1 is zero.

In Figure 3, we show the transfer function of our threshold function for $V_{ref} = 1$. The transfer function is obtained from a circuit simulator. In the simulator, the open loop gain of OP-AMP1 is set to $10^8$ and the open loop gain of OP-AMP2-to-OP-AMP4 is set to $10^5$. For the two MOSFETs, they are n-type with the following parameters: channel width = 100 μm, channel length = 200 nm, transconductance = 118 μ A/V$^2$, zero-bias threshold voltage = 430 mV and channel-length modulation = 60 mV$^{-1}$. From the figure, the transfer function quite matches our theoretical model with a large value of $\gamma$.

### *4.2. Circuit Structure*

Figure 4 shows the analog realization of (21) in the block diagram level. In the realization, there are $n$ blocks to compute $\frac{du_i}{dt}$'s. Inside each block, there are adders, multipliers, and square circuits. Additionally, there is a block to compute $\frac{d\alpha}{dt}$. The time derivatives $\frac{du_i}{dt}$'s and $\frac{d\alpha}{dt}$ are then fed to the integrators to obtain internal variables $u_i(t)$ and $\alpha$. In order to obtain the decision variables $x_i(t)$'s, the internal variables are fed to $n$ thresholding elements (Figure 2), where $V_{ref} = 1/2$.

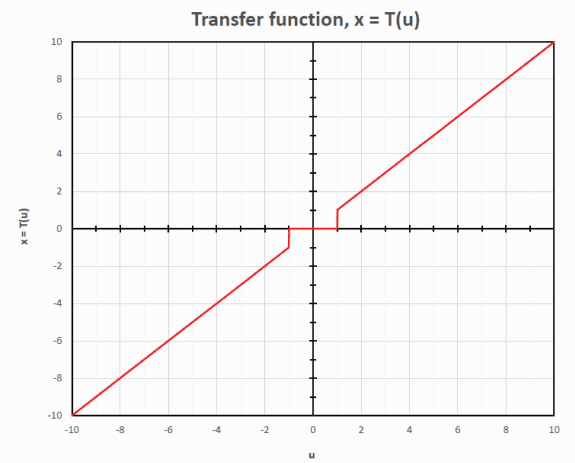



**Figure 3.** Thresholding function $T(x)$ obtained from the circuit simulation of Figure 2 with $V_{ref} = 1$.

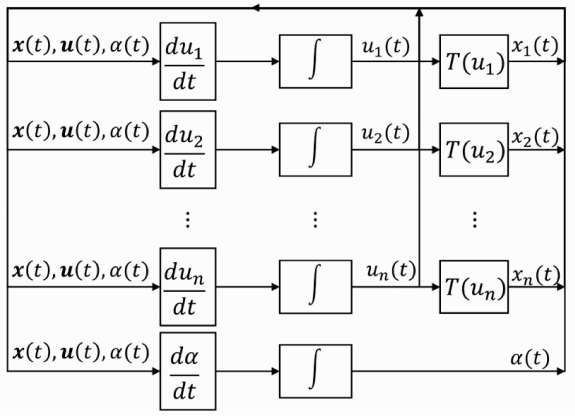



**Figure 4.** Analog realization of (21) in the block diagram level.

### *4.3. Circuit Simulation*

In this subsection, we use a small scale problem to verify our approach based on the Matlab Simulink platform. The problem details are:

- $n = 8$ and $m = 6$.
- Measurement matrix:

$$\mathbf{\Phi} = \frac{1}{\sqrt{8}} \begin{pmatrix} 1 & -1 & -1 & -1 & 1 & -1 & 1 & 1 \\ 1 & 1 & -1 & 1 & 1 & 1 & 1 & -1 \\ -1 & 1 & -1 & -1 & -1 & 1 & 1 & 1 \\ -1 & 1 & 1 & 1 & 1 & -1 & 1 & 1 \\ 1 & 1 & 1 & -1 & 1 & 1 & -1 & 1 \\ -1 & -1 & -1 & 1 & 1 & 1 & -1 & 1 \end{pmatrix}.$$

- Real sparse vector $\boldsymbol{x} = [-2, 0, 0, 0, 0, 0, 2, 0]^{\mathrm{T}}$ and noisy observation vector $\boldsymbol{b} = [-0.0001, 0.0038, 1.4149, 1.4145, -1.4140, -0.0035]^{\mathrm{T}}$.
- Noise tolerant parameter $\theta = 0.001$.

We build the Simulink file for our model, shown in Figure 4. We implement the thresholding function based on Figure 2. Other blocks are based on Simulink's functional blocks. To verify our Simulink result, we also consider the discrete time simulation on (21). For digitization of (21), the discrete time simulation equations are

$$\boldsymbol{u}(t+\Delta t) = \boldsymbol{u}(t) + \Delta t \frac{d\boldsymbol{u}}{dt} \tag{22a}$$

$$\alpha(t+\Delta t) = \alpha(t) + \Delta t \frac{d\alpha}{dt}. \tag{22b}$$

where $\Delta t = 0.001$. Figure 5 shows the dynamics from the Simulink results and discrete time simulation. From the figure, the dynamics from the two approaches are nearly the same. The final outputs $\boldsymbol{x}$ are

$$\text{Simulink: } \boldsymbol{x} = [-1.9970, 0, 0, 0, 0, 0, 2.0037, 0]^\top \tag{23a}$$

$$\text{Discrete time: } \boldsymbol{x} = [-1.9965, 0, 0, 0, 0, 0, 2.0032, 0]^\top. \tag{23b}$$

Clearly, from Figures 5 and (23), both approaches produce the similar results.

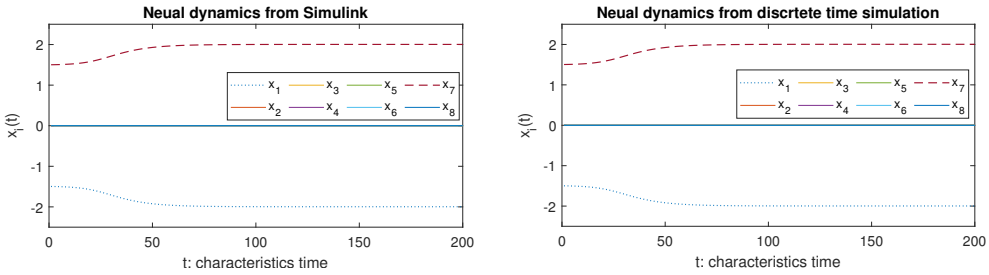



**Figure 5.** Dynamics obtained from Simulink and discrete time simulation.

## 5. Properties of the Dynamics

The first issue that we need to address is that equilibrium points of the LPNN-LPQC model should fulfill the KKT conditions of the LPQC problem, stated in (13). Otherwise, the LPNN-LPQC model cannot find out local/global minimums of the LPQC problem. Note that since the LPQC problem or the original $\ell_0$-norm CBPDN problem, stated in (2), are non-convex, few algorithms can ensure that their solutions are the global minimum.

The relationship between the equilibrium points of the LPNN-LPQC model and the KKT conditions are summarized in the following theorem.

**Theorem 2.** *Given that* $\{\boldsymbol{u}^*, \alpha^*\}$ *with* $\alpha^* \neq 0$ *is an equilibrium point of the LPNN-LPQC model, this point corresponds to the KKT conditions of the LPQC problem. Note that* $\boldsymbol{x}^* = T(\boldsymbol{u}^*)$.

**Proof.** From (21), when $\{\boldsymbol{u}^*, \alpha^*\}$ is an equilibrium point, we

$$-\boldsymbol{u}^* + \boldsymbol{x}^* + 2\alpha^{*2}\boldsymbol{\Phi}^{\mathrm{T}}(\boldsymbol{b} - \boldsymbol{\Phi}\boldsymbol{x}^*) = \boldsymbol{0}, \tag{24a}$$

$$2\alpha^*(\|\boldsymbol{b} - \boldsymbol{\Phi}\boldsymbol{x}^*\|_2^2 - m\theta^2) = 0. \tag{24b}$$

Clearly, (24a) is the same as (19a) with $\alpha^{*2} = \lambda^*$. Additionally, when $\alpha^* \neq 0$, from (24b) becomes $\|\boldsymbol{b} - \boldsymbol{\Phi}\boldsymbol{x}^*\|_2^2 - m\theta^2 = 0$ which is the same (19b). The proof is completed. □

Theorem 2 tells us that equilibrium points of the LPNN-LPQC model corresponds to the KKT condition of the LPQC problem. Another concern is the stability of the equilibrium points of the LPNN-LPQC model. Theorem 3 presents the stability of the equilibrium points of the LPNN-LPQC model.

**Theorem 3.** *Given a positive* $\epsilon$ *and for a sufficient large positive* $\gamma$, *if* $(\boldsymbol{u}^*, \alpha^*)$ *with* $\alpha \neq 0$ *is an equilibrium point of the LPNN-LPQC model and for all* $i$ *either* $|u_i^*| > \frac{1}{2} + \frac{1}{\gamma}\log\frac{\gamma}{\epsilon}$ *or* $|u_i^*| < \frac{1}{2} - \frac{1}{\gamma}\log\frac{\gamma}{\epsilon}$, *then the equilibrium point is an asymptotically stable point.*

**Proof.** The condition of either $|u_i| > \frac{1}{2} + \frac{1}{\gamma}\log\frac{\gamma}{\epsilon}$ or $|u_i| < \frac{1}{2} - \frac{1}{\gamma}\log\frac{\gamma}{\epsilon}$ is equivalent to for all $i$, $|u_i| \notin [\frac{1}{2} - \zeta, \frac{1}{2} + \zeta]$, where $\zeta = \frac{1}{\gamma}\log\frac{\gamma}{\epsilon}$. With the condition, we define two index sets, $\Gamma_{\boldsymbol{u}}$ and $\Gamma_{\boldsymbol{u}}^c$, given by

**active set $\Gamma_{\boldsymbol{u}}$:** If $|u_i| > 1 + \zeta$, then $i \in \Gamma_{\boldsymbol{u}}$.

**inactive set $\Gamma_{\boldsymbol{u}}^c$:** If $|u_i| < 1 - \zeta$, then $i \in \Gamma_{\boldsymbol{u}}^c$.

Obviously, for a given $\epsilon$ and a sufficient large $\gamma$, $\zeta$ tends to zero and the value of $\zeta$ can be arbitrary small. Thus, the inactive neurons have nearly no effect on the dynamics, i.e., $T(u_i)$ tends to 0 for sufficient large $\gamma$ (see Property 3).

For a given $\boldsymbol{a}$, we define $\boldsymbol{a}_{\Gamma_{\boldsymbol{u}}}$ as the vector composed of the elements of $\boldsymbol{a}$ indexed by $\Gamma_{\boldsymbol{u}}$ and $\boldsymbol{a}_{\Gamma_{\boldsymbol{u}}^c}$ as the vector composed of the elements of $\boldsymbol{a}$ indexed by $\Gamma_{\boldsymbol{u}}^c$. Similarly, given a matrix $\boldsymbol{\Phi}$, we define $\boldsymbol{\Phi}_{\Gamma_{\boldsymbol{u}}}$ as the matrix composed of the columns of $\boldsymbol{\Phi}$ indexed by $\Gamma_{\boldsymbol{u}}$ and $\boldsymbol{\Phi}_{\Gamma_{\boldsymbol{u}}^c}$ as the matrix composed of the columns of $\boldsymbol{\Phi}$ indexed by $\Gamma_{\boldsymbol{u}}^c$.

Now, we consider the dynamics near $\boldsymbol{u}^*$ and $\alpha^*$. We have two index sets $\Gamma_{\boldsymbol{u}^*}$ and $\Gamma_{\boldsymbol{u}^*}^c$ As mentioned in the above, for inactive states $\boldsymbol{x}_{\Gamma_{\boldsymbol{u}^*}^c} \to \boldsymbol{0}$ for sufficient large $\gamma$. The dynamics given in (21) can be rewritten as:

$$\frac{d\boldsymbol{u}_{\Gamma_{\boldsymbol{u}^*}}}{dt} = -\boldsymbol{u}_{\Gamma_{\boldsymbol{u}^*}} + \boldsymbol{x}_{\Gamma_{\boldsymbol{u}^*}} + 2\alpha^2\boldsymbol{\Phi}_{\Gamma_{\boldsymbol{u}^*}}^\top(\boldsymbol{b} - \boldsymbol{\Phi}_{\Gamma_{\boldsymbol{u}^*}}\boldsymbol{x}_{\Gamma_{\boldsymbol{u}^*}}), \tag{25a}$$

$$\frac{d\alpha}{dt} = 2\alpha(\|\boldsymbol{b} - \boldsymbol{\Phi}_{\Gamma_{\boldsymbol{u}^*}}\boldsymbol{x}_{\Gamma_{\boldsymbol{u}^*}}\|_2^2 - m\theta^2), \tag{25b}$$

$$\frac{d\boldsymbol{u}_{\Gamma_{\boldsymbol{u}^*}^c}}{dt} = -\boldsymbol{u}_{\Gamma_{\boldsymbol{u}^*}^c} + 2\alpha^2\boldsymbol{\Phi}_{\Gamma_{\boldsymbol{u}^*}^c}^\top\left(\boldsymbol{b} - \boldsymbol{\Phi}_{\Gamma_{\boldsymbol{u}^*}}\boldsymbol{x}_{\Gamma_{\boldsymbol{u}^*}}\right). \tag{25c}$$

Furthermore, the linearization of (25) around the equilibrium point $(\boldsymbol{u}^*, \alpha^*)$ is

$$\begin{bmatrix} \frac{d\boldsymbol{u}_{\Gamma_{\boldsymbol{u}^*}}}{dt} \\ \frac{d\alpha}{dt} \\ \frac{d\boldsymbol{u}_{\Gamma_{\boldsymbol{u}^*}^c}}{dt} \end{bmatrix} = -\boldsymbol{H}\begin{bmatrix} \boldsymbol{u}_{\Gamma_{\boldsymbol{u}^*}} - \boldsymbol{u}^*_{\Gamma_{\boldsymbol{u}^*}^c} \\ \alpha - \alpha^* \\ \boldsymbol{u}_{\Gamma_{\boldsymbol{u}^*}^c} - \boldsymbol{u}^*_{\Gamma_{\boldsymbol{u}^*}^c} \end{bmatrix}, \tag{26}$$

where "$-\boldsymbol{H}$" is the Jacobian matrix at $(\boldsymbol{u}^*, \alpha^*)$ and it is given by

$$-\boldsymbol{H} = \left[\begin{array}{c|c|c} \frac{d\dot{\boldsymbol{u}}_{\Gamma_{\boldsymbol{u}^*}}}{d\boldsymbol{u}_{\Gamma_{\boldsymbol{u}^*}}} & \frac{d\dot{\boldsymbol{u}}_{\Gamma_{\boldsymbol{u}^*}}}{d\alpha} & \frac{d\dot{\boldsymbol{u}}_{\Gamma_{\boldsymbol{u}^*}}}{d\boldsymbol{u}_{\Gamma_{\boldsymbol{u}^*}^c}} \\ \hline \frac{d\dot{\alpha}}{d\boldsymbol{u}_{\Gamma_{\boldsymbol{u}^*}}} & \frac{d\dot{\alpha}}{d\alpha} & \frac{d\dot{\alpha}}{d\boldsymbol{u}_{\Gamma_{\boldsymbol{u}^*}^c}} \\ \hline \frac{d\dot{\boldsymbol{u}}_{\Gamma_{\boldsymbol{u}^*}^c}}{d\boldsymbol{u}_{\Gamma_{\boldsymbol{u}^*}}} & \frac{d\dot{\boldsymbol{u}}_{\Gamma_{\boldsymbol{u}^*}^c}}{d\alpha} & \frac{d\dot{\boldsymbol{u}}_{\Gamma_{\boldsymbol{u}^*}^c}}{d\boldsymbol{u}_{\Gamma_{\boldsymbol{u}^*}^c}} \end{array}\right]\Bigg|_{(\boldsymbol{u},\alpha)=(\boldsymbol{u}^*,\alpha^*)}. \tag{27}$$

From the given condition, for active nodes, we have $dx_i/du_i \approx 1$, and for inactive nodes $dx_i/du_i \approx 0$. After deriving the sub-matrices in (27), we obtain

$$\boldsymbol{H} = \left[\begin{array}{c|c|c} 2\bar{\alpha}^2\boldsymbol{\Phi}_{\bar{\Gamma}}^\top\boldsymbol{\Phi}_{\bar{\Gamma}} & -4\bar{\alpha}\boldsymbol{\Phi}_{\bar{\Gamma}}^\top(\boldsymbol{b} - \boldsymbol{\Phi}_{\bar{\Gamma}}\bar{\boldsymbol{x}}_{\bar{\Gamma}}) & \varnothing \\ \hline 4\bar{\alpha}(\boldsymbol{b} - \boldsymbol{\Phi}_{\bar{\Gamma}}\bar{\boldsymbol{x}}_{\bar{\Gamma}})^\top\boldsymbol{\Phi}_{\bar{\Gamma}} & 0 & \varnothing \\ \hline 2\bar{\alpha}^2\boldsymbol{\Phi}_{\bar{\Gamma}^c}^\top\boldsymbol{\Phi}_{\bar{\Gamma}} & -4\bar{\alpha}\boldsymbol{\Phi}_{\bar{\Gamma}^c}^\top(\boldsymbol{b} - \boldsymbol{\Phi}_{\bar{\Gamma}}\bar{\boldsymbol{x}}_{\bar{\Gamma}}) & \boldsymbol{I} \end{array}\right], \tag{28}$$

where $\varnothing$ denotes a matrix of zero with appropriate size. Following the proof logic of Theorem 5 in [8], one can find that all eigenvalues of $\boldsymbol{H}$ are with positive real part. Therefore, according to the classical control theory, the corresponding equilibrium point $(\boldsymbol{u}^*, \alpha^*)$ is asymptotically stable. The proof is completed. □

## 6. Experiment Results

### *6.1. Comparison Algorithms and Settings*

This section compares our proposed LPNN-LPQC model with a number of digital numerical methods and analog neural methods. The comparison numerical methods are SPGL1 [30,50], MCP [35], AMP [37],$\ell_0$-ADMM [39,40], $\ell_0$-ZAP [38], NIHT [36], and ECME [41]. The comparison analog methods are IPNNSR [11] and PNN-$\ell_1$ [13].

The seven numerical algorithms are described as follows. The SPGL1 [30,50] is a standard $\ell_1$-norm approach. The MCP [35] is based on the approximation of the $\ell_0$-norm function and its turning parameters are selected by cross validation. The NIHT [36], ECME [41], and AMP [37] are iterative algorithms, and they handle the $\ell_0$-norm term by the hard thresholding concept. The $\ell_0$-ADMM [39,40] uses the frameworks of ADMM and hard thresholding. The $\ell_0$ ZAP [38] utilizes the idea of adaptive filter and projection. The two analog comparison models are IPNNSR [11] and PNN-$\ell_1$ [13], and are developed based the projection concepts.

The setting of our experiment follows that of the experiment in [51]. The measurement matrix $\mathbf{\Phi}$ is a random $\pm 1/\sqrt{m}$ matrix. The dimension $n$ of the sparse signal $\boldsymbol{x}$ is 4096. A sparse signal contains $k$ non-zero elements which locations are randomly chosen with uniform distribution. Their corresponding value are random $\pm 1$. In our experiment we set $k = \{75, 100, 125\}$.

### *6.2. Parameter Settings*

To conduct the experiment, we need to select some parameters for the proposed method and comparison methods. In our proposed analog method, there are two tuning parameters which are $\gamma$ and $\kappa$. To make the approximation $\ell_p$-norm close to the $\ell_0$ norm, we should use a large $\gamma$. In our experiment, we set $\gamma = 10,000$. Parameter $\kappa$ is used to control the minimum value of the magnitudes of the decision variables $x_i$. In our experiment, we set $\kappa = \frac{1}{2}$ and $\Delta t = 0.0001$.

The SPGL1 package is used to solve

$$\min_{\boldsymbol{x}} \|\boldsymbol{b} - \mathbf{\Phi}\boldsymbol{x}\|_2^2 \text{ subject to } \|\boldsymbol{x}\|_1 \leq k,$$

where $k$ is the number of non-zero elements in $\boldsymbol{x}$. In the experiment, we set the maximum number of iterations to 100,000.

The MCP algorithm is used to solve

$$\min_{\boldsymbol{x}} \|\boldsymbol{b} - \mathbf{\Phi}\boldsymbol{x}\|_2^2 + P_{(\lambda,\gamma)}(\boldsymbol{x}),$$

where $P_{(\lambda,\gamma)}(\boldsymbol{x})$ is a penalty function to control the sparsity of the solution, and $\lambda$ and $\gamma$ are parameters in $P_{(\lambda,\gamma)}(\boldsymbol{x})$. We set $\gamma = 1.5$ and use a linear search to obtain the best value of $\lambda$. We set the maximum number of iterations to 100,000.

The AMP, NIHT, and ECME algorithms are used to solve

$$\min_{\boldsymbol{x}} \|\boldsymbol{b} - \mathbf{\Phi}\boldsymbol{x}\|_2^2 \text{ subject to } \|\boldsymbol{x}\|_0 \leq k,$$

where $k$ is the number of non-zero elements in $\boldsymbol{x}$. We set the maximum number of iterations to 100,000.

The $\ell_0$-ADMM algorithm is used to solve

$$\min_{\boldsymbol{x}} \|\boldsymbol{b} - \mathbf{\Phi}\boldsymbol{x}\|_2^2 \text{ subject to } \|\boldsymbol{x}\|_0 \leq k,$$

where $k$ is the number of non-zero elements in $\boldsymbol{x}$. In $\ell_0$-ADMM, there is an augmented Lagrangian parameter $\rho$. It is set to 0.5. The maximum number of iterations to 100,000.

The $\ell_0$-ZAP algorithm is used to solve

$$\min_{\boldsymbol{x}} \|\boldsymbol{b} - \boldsymbol{\Phi}\boldsymbol{x}\|_2^2 + \gamma\|\boldsymbol{x}\|_0,$$

where $\gamma$ is used to control the sparsity of the solution vector. We use a linear search to obtain the best value of $\gamma$. In addition, there are two parameter $\kappa$ and $\alpha$. In the experiment, $\kappa = 0.001$ and $\alpha = 2$. Additionally, the maximum number of iterations is 100,000.

The IPNNSR and PNN-$\ell_1$ analog models are used to solve

$$\min_{\boldsymbol{x}} \|\boldsymbol{x}\|_1 \text{ subject to } \boldsymbol{b} = \boldsymbol{\Phi}\boldsymbol{x}.$$

For IPNNSR, we set $\Delta t = 1$. For PNN-$\ell_1$, we set $\Delta t = 0.01$.

### *6.3. Convergence*

The proposed algorithm is an analog neural network. Hence, one important issue is the time to reach the equilibrium. Here, we conduct an experiment to empirically study the convergent time. Some typical dynamics are given in Figure 6. In the figure, the first row is the case of $k = 75$ and $m = 500$, the second row is the case of $k = 100$ and $m = 600$, and the third row is the case of $k = 125$ and $m = 700$. Since there are 4096 elements in the decision variable vector $\boldsymbol{x}$, the legibility of the figure will be very poor, if we plot all $u_i(t)$'s and $x_i(t)$'s in the figure. Therefore, we only plot the dynamics of the $u_i(t)$'s and $x_i(t)$'s whose original $x_i$ values are non-zero. From the figure, it can be seen that within 20–40 characteristic time units, the dynamics of our proposed analog neural network settle down.

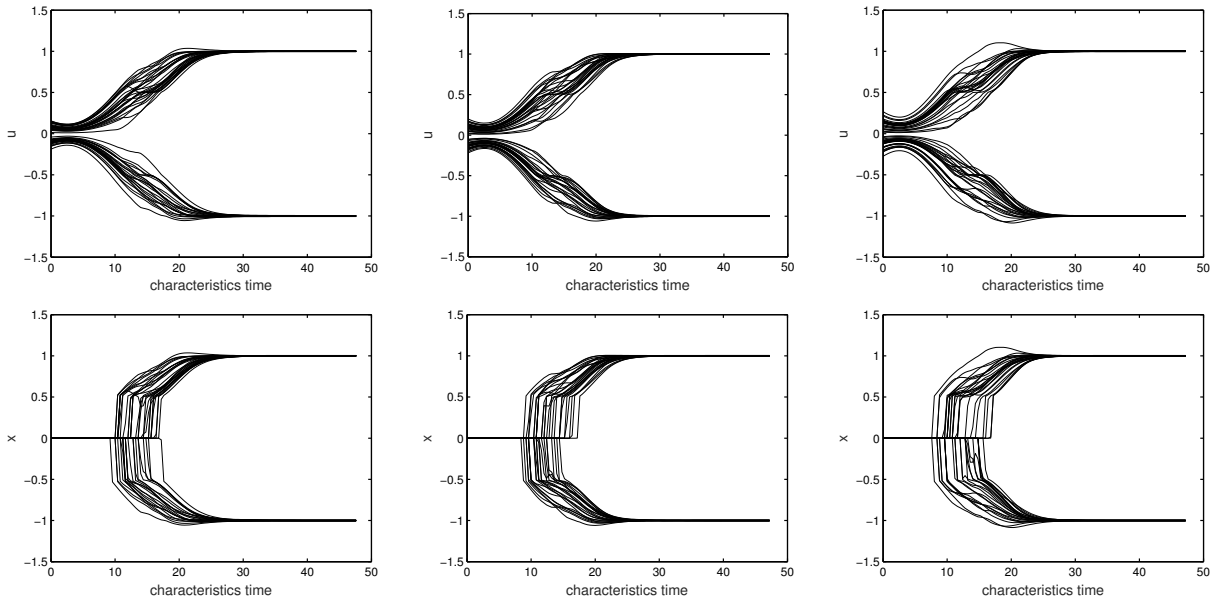



**Figure 6.** Typical dynamics of $\boldsymbol{u}$ and $\boldsymbol{x}$ for the LPNN-LPQC model, where $n = 4096$. The first column: $k = 75$, $m = 500$. The second column: $k = 100$, $m = 600$. The third column, $k = 125$, $m = 700$. In all sub-figures, we only show the dynamics of the $u_i$'s and $x_i$'s whose original $x_i$ values are non-zero, because there are 4096 curves in each sub-figure when we show all the dynamics for $u_i$'s and $x_i$'s.

### *6.4. Comparison with Other Algorithms*

To further analyze the performance of our proposed method, we compare it with seven numerical algorithms and two analog models.

The observation vectors are generated by the following noisy model:

$$\boldsymbol{b} = \boldsymbol{\Phi}\boldsymbol{x} + \boldsymbol{e} \tag{29}$$

where $\boldsymbol{e}$ is a zero mean Gaussian noise with standard deviation $\sigma = \{0.001, 0.005, 0.01\}$. The experiments repeat 100 times with different measurement matrix, initial states, and sparse signals. For each instance we declare that it is successful if

$$\frac{\|\boldsymbol{x}_0 - \hat{\boldsymbol{x}}\|_2}{\|\boldsymbol{x}_0\|_2} \leq tol, \tag{30}$$

where $\boldsymbol{x}_0$ denotes the true signal, $\hat{\boldsymbol{x}}$ is the recovered signal, and $tol$ is the tolerant value. In our experiment, we set $tol = 0.01$.

The successful rate results are shown in Figure 7. For all the algorithms, their performances are improved with the increasing numbers of measurements. From the figure, all $\ell_0$-norm and $\ell_p$-norm models are superior to the $\ell_1$ models, including SPGL1, IPNNSR, and PNN-$\ell_1$. In addition, comparing with the seven numerical methods, our proposed method usually needs less measurements to obtain the same probability of exact reconstruction.

Comparing with the two analog models, IPNNSR and PNN-$\ell_1$, our LPNN-LPQC model is better. For example, with 125 non-zero elements and noise level equal to 0.01, the IPNNSR and PNN-$\ell_1$ need around 650 measurement vectors, while our LPNN-LPQC needs around 575 to 600 measurement vectors only. The rationale of the improvement is that our model uses a $\ell_0$-norm approach, while IPNNSR and PNN-$\ell_1$ are based on the $\ell_0$-norm.

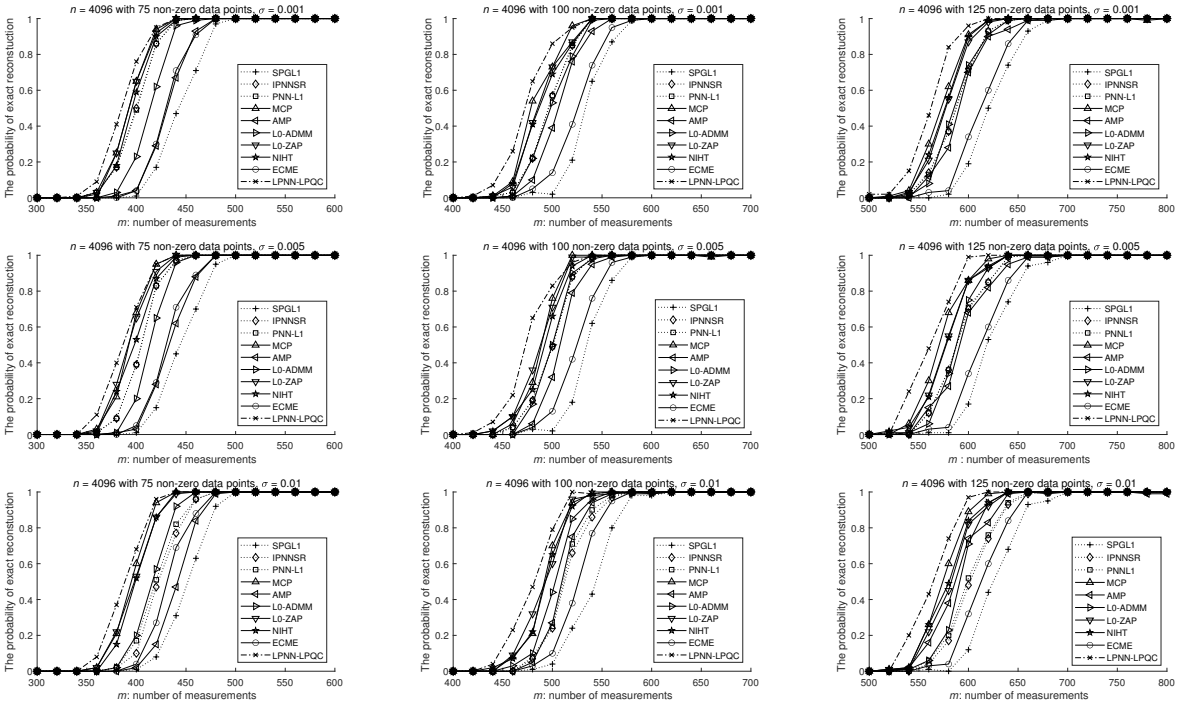



**Figure 7.** Simulation results of different algorithms, where $n = 4096$. For the first column, $k = 75$. For the second column, $k = 100$. For the third column, $k = 125$. The three rows are based on three different noise levels. The experiments are repeated 100 times using different settings.

### *6.5. Comparison with Other Analog Models*

As the paper proposed an analog model, namely LPNN-LPQC, for sparse recovery, this subsection performs a deep discussion on the comparison among our proposed LPNN-LPQC, IPNNSR [11], and PNN-$\ell_1$ [13]. We discuss three different aspects: probability of exact reconstruction and reconstruction error. In IPNNSR [11] and PNN-$\ell_1$ [13], the circuit realization of thresholding operator and the projection operator were not addressed.

#### 6.5.1. Successful Rate of Recall

Since Figure 7 shows the successful rates of all algorithms, it may not be easy to see the difference among the three analog models. In Figure 8, we only present the results of three analog models. From the figure, it can be seen that the performance of our LPNN-LPQC is better than that of IPNNSR and PNN-$\ell_1$. In particular, the performance improvement is significant when the number of non-zero elements is large and the noise level is high. For instance, with 125 non-zero elements and noise level = 0.01, the IPNNSR and PNN-$\ell_1$ need around 650 measurements for high successful rates of recall, while our LPNN-LPQC needs around 575 to 600 measurements only.

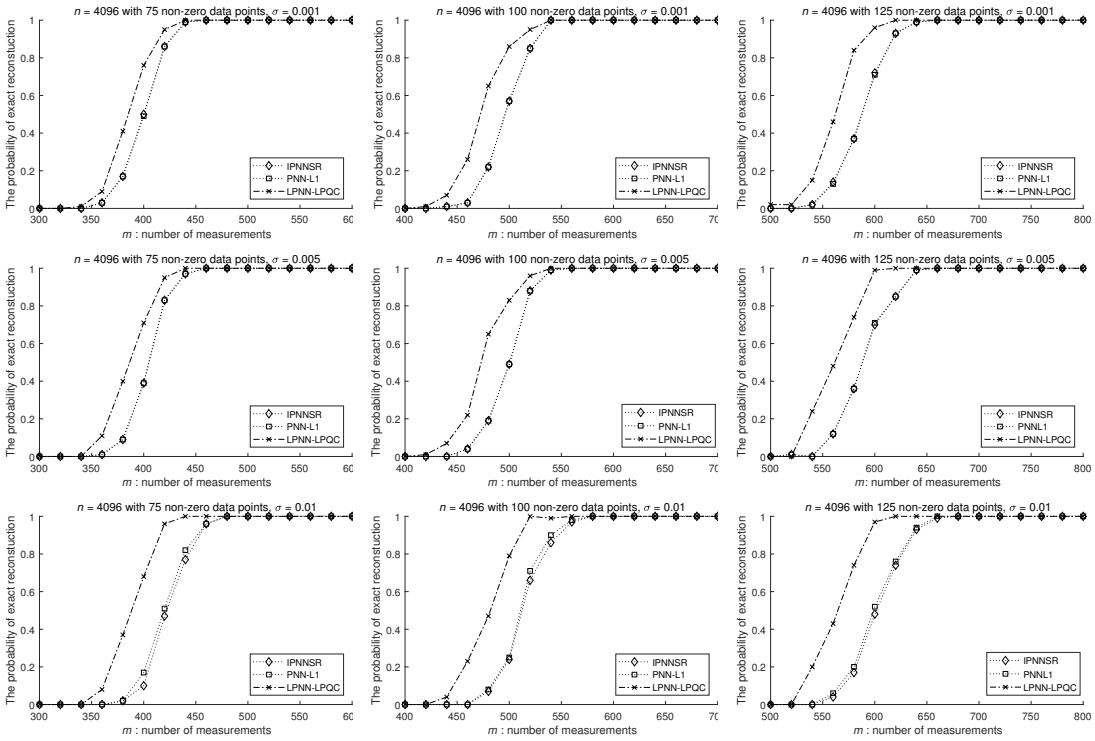



**Figure 8.** Comparison with other analog models on probability of reconstruction, where $n = 4096$. For the first column, $k = 75$. For the second column, $k = 100$. For the third column, $k = 125$. The three rows are based on three different noise levels. The experiments are repeated 100 times using different settings.

#### 6.5.2. MSE of Recall

The successful rate results concern about whether the estimated $\boldsymbol{x}$ has the correct non-zero positions. Here, in Figure 9, we present the MSE versus the number of measurements used. From Figure 9, it can be seen that in terms of MSE, the performance of our LPNN-LPQC is much better than that of IPNNSR and PNN-$\ell_1$. In most cases, the MSE values of our proposed model are less than those of IPNNSR and PNN-$\ell_1$ in one or two orders of magnitude. For example, for $k = 75$, $\sigma = 0.005$, and $M = 450$, the MSE values of the IPNNSR and PNN-$\ell_1$ are around $10^{-3}$, while the MSE value of the proposed model is around $10^{-5}$ only.

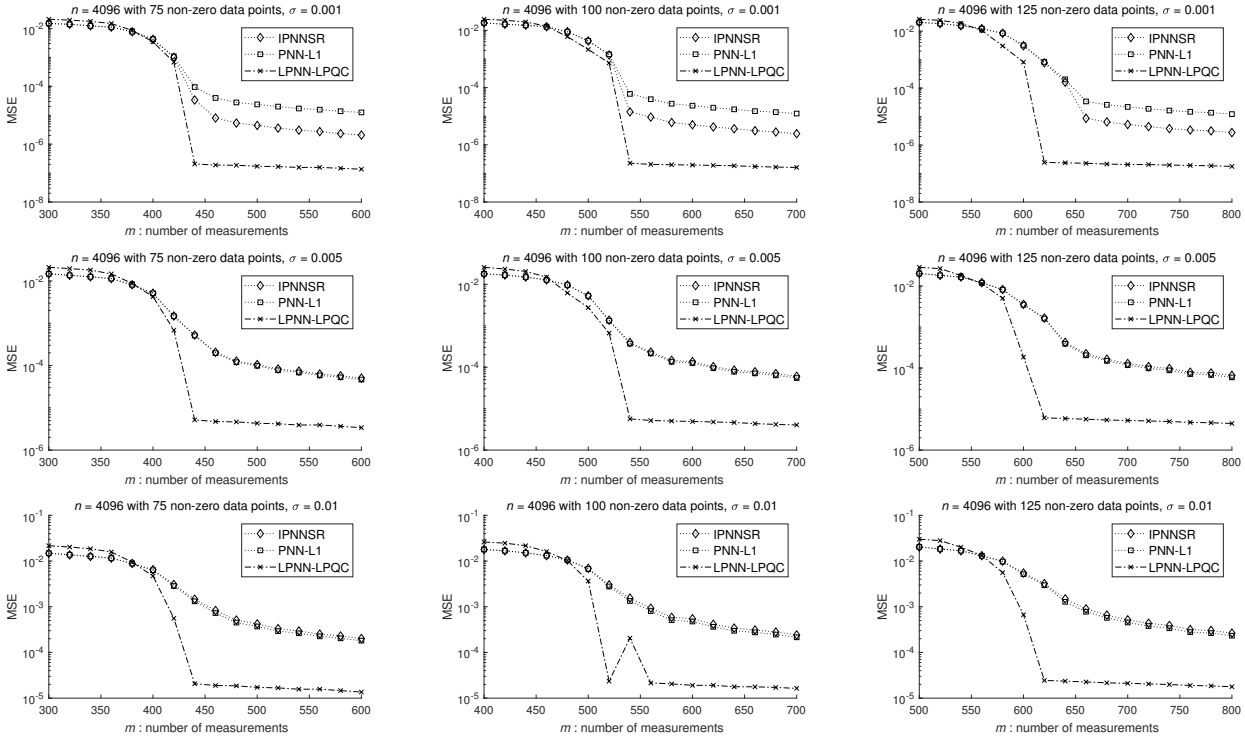



**Figure 9.** Comparison with other analog models on MSE, where $n = 4096$. For the first column, $k = 75$. For the second column, $k = 100$. For the third column, $k = 125$. The three rows are based on three different noise levels. The experiments are repeated 100 times using different settings.

## 7. Conclusions

This paper proposed a LPNN-LPQC method for sparse recovery under noise environment. The proposed algorithm is an analog neural network. We showed that the equilibrium points of the model satisfy the KKT conditions of the LPQC problem, and that the equilibrium points of the model are asymptotically stable. From the simulation results, we can see that the performance of the proposed algorithm is comparable to, even superior to many state-of-art $\ell_0$-norm or $\ell_p$-norm numerical methods. In addition, our proposed algorithm is superior to two analog models. We also presented the circuit realization for the thresholding element and performed circuit simulation for verifying our realization based on the MATLAB Simulink.

In our analysis, we assume that the realization of analog circuit does not have any time delay or synchronization problem. In fact, time delay and mis-synchronization may affect the stability of analog circuits [52,53]. Hence, one of future works is to investigate the behaviour of the proposed analog model with time delay or mis-synchronization.

**Author Contributions:** Conceptualization, H.W., R.F., C.-S.L. and A.G.C.; methodology, C.-S.L., H.W., R.F. and H.P.C.; software, H.W., R.F., H.P.C.; writing—original draft preparation, H.W.; writing—review and editing, C.-S.L.; supervision, C.-S.L. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work was supported by the National Natural Science Foundation of China (Grant No. 62206178) and a research grant from City University of Hong Kong. Grant number 9678295.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** Not applicable.

**Conflicts of Interest:** The authors declare no conflict of interest.

## Abbreviations

The following abbreviations are used in this manuscript:

| | |
|---|---|
| LPNN | Lagrange Programming Neural Network |
| LPNN-LPQC | LPNN with $\ell_p$ objective and Quadratic Constraint |
| CBPDN | Constrained Basis Pursuit Denoise (CBPDN) |
| LCA | Locally Competitive Algorithm |
| KKT | Karush Kuhn Tucker |
| MCP | Minimax Concave Penalty |
| NIHT | Normalized Iterative Hard Threshold |
| AMP | Approximate Message Passing |
| $\ell_0$-$ZAP$ | $\ell_0$-norm Zero Attraction Projection |
| $\ell_0$-$ADMM$ | $\ell_0$-norm Alternating Direction Method of Multipliers |
| ECME | Expectation Conditional Maximization Either |

## Appendix A. Proof of Property 3

**P3.a**

Recall that $f(u) = \frac{1}{1+\exp\left(-\gamma(|u|-\frac{1}{2})\right)}$, and that $f(u)$ is an even function. Clearly, $T(u) = uf(u)$ is an odd function. First, since $f(u)$ is an even function, the proof only presents the case of $u \geq \frac{1}{2} + \frac{1}{\gamma}\log\frac{1}{\epsilon}$.

In the case of $u \geq \frac{1}{2} + \frac{1}{\gamma}\log\frac{1}{\epsilon}$, $f(u)$ is monotonic increasing. Hence

$$f(u) \geq f(\frac{1}{2} + \frac{1}{\gamma}\log\frac{1}{\epsilon}) = \frac{1}{1+\exp\left(-\log\frac{1}{\epsilon}\right)} = \frac{1}{1+\epsilon}. \tag{A1}$$

In addition, it is easy to show that $f(u)$ is monotonic increasing, and that $\lim_{|u|\to\infty} f(u) = 1$. In conclusion, we have $\frac{1}{1+\epsilon} \leq f(u) < 1$. **P3.a** is proved.

**P3.b**

Since $f(u)$ is an even function, the proof only presents that the case of $u \leq \frac{1}{2} - \frac{1}{\gamma}\log\frac{1}{\epsilon}$.

In the case of $u \leq \frac{1}{2} - \frac{1}{\gamma}\log\frac{1}{\epsilon}$, $f(u)$ is monotonic increasing. Hence, we have

$$f(u) \leq f(\frac{1}{2} - \frac{1}{\gamma}\log\frac{1}{\epsilon}) = \frac{1}{1+\exp\left(\log\frac{1}{\epsilon}\right)} = \frac{\epsilon}{1+\epsilon}. \tag{A2}$$

Additionally, it is obvious that $f(u) > 0$. Therefore, $0 < f(u) \leq \frac{\epsilon}{1+\epsilon}$. **P3.b** is proved.

**P3.c**

For simplicity, let $g(u) = \exp\left(-\gamma(|u| - \frac{1}{2})\right)$. Thus, the derivative of $T(u)$ can be rewritten as

$$\frac{dT(u)}{du} = \frac{1}{1+g(u)} + \frac{\gamma|u|g(u)}{(1+g(u))^2} \tag{A3}$$

and the second derivative of $T(u)$ is

$$\frac{d^2T(u)}{du^2} = \frac{2\gamma\,\mathrm{sign}(u)g(u)}{(1+g(u))^2} - \frac{\gamma^2 u g(u)}{(1+g(u))^2} + \frac{2\gamma^2 u g(u)^2}{(1+g(u))^3}. \tag{A4}$$

Since $T(u)$ is an odd function and $\frac{dT(u)}{du}$ is an even function, we only present the proof for the case of $u > \frac{1}{2} + \frac{1}{\gamma}\log\frac{\gamma}{\epsilon}$. The proof consists of two parts. First, we show the monotonic property of $\frac{dT(u)}{du}$ for $u > \frac{1}{2} + \frac{1}{\gamma}\log\frac{\gamma}{\epsilon}$. Second, the upper and lower bounds of $\frac{dT(u)}{du}$ are established.

For the case of $u > \frac{1}{2} + \frac{1}{\gamma}\log\frac{\gamma}{\epsilon}$, we have $0 < g(u) < \frac{\epsilon}{\gamma}$. The second derivative of $T(u)$ in (A4) can be rewritten as:

$$\frac{d^2T(u)}{du^2} = g(u) \cdot \frac{(2\gamma - \gamma^2 u)(1+g(u)) + 2\gamma^2 u g(u)}{(1+g(u))^3}. \tag{A5}$$

Since $0 < g(u) < \epsilon/\gamma$, we can deduce that

$$\begin{aligned} & (2\gamma - \gamma^2 u)(1+g(u)) + 2\gamma^2 u g(u) \\ = \;& 2\gamma - \gamma^2 u + (2\gamma + \gamma^2 u) g(u) < 2\gamma - \gamma^2 u + (2\gamma + \gamma^2 u)\frac{\epsilon}{\gamma} \\ = \;& 2\gamma - \gamma^2 u + 2\epsilon + 2\gamma u\epsilon < (2 - \gamma + 4\epsilon)\gamma u < 0. \end{aligned} \tag{A6}$$

From (A5), (A6), $g(u) > 0$, and $(1+g(u))^3 > 0$, we deduce that, if $u > \frac{1}{2} + \frac{1}{\gamma}\log\frac{\gamma}{\epsilon}$, then $d^2T(u)/du^2 < 0$ and $\frac{dT(u)}{du}$ is monotonic decreasing. Thus, for $u > \frac{1}{2} + \frac{1}{\gamma}\log\frac{\gamma}{\epsilon}$, a lower bound on $\frac{dT(u)}{du}$ is $\frac{dT(u)}{du}|_{u\to\infty}$. Since $\lim_{u\to\infty} g(u) = 0$ and $\frac{dT(u)}{du}|_{u\to\infty} = 1$, we have

$$\frac{dT(u)}{du} > 1. \tag{A7}$$

On the contrary, an upper bound in $u > \frac{1}{2} + \frac{1}{\gamma}\log\frac{\gamma}{\epsilon}$, is $\frac{dT(u)}{du}|_{u=\frac{1}{2}+\frac{1}{\gamma}\log\frac{\gamma}{\epsilon}}$. The value of $\frac{dT(u)}{du}|_{u=\frac{1}{2}+\frac{1}{\gamma}\log\frac{\gamma}{\epsilon}}$ is

$$\begin{aligned} & \tfrac{dT(u)}{du}|_{u=\frac{1}{2}+\frac{1}{\gamma}\log\frac{\gamma}{\epsilon}} = \tfrac{1}{1+\frac{\epsilon}{\gamma}} + \tfrac{\gamma u\frac{\epsilon}{\gamma}}{(1+\frac{\epsilon}{\gamma})^2} \\ = \;& \tfrac{\gamma^2+\gamma\epsilon+u\epsilon\gamma^2}{(\gamma+\epsilon)^2} = \tfrac{(\gamma+\epsilon)^2+u\epsilon\gamma^2-\epsilon\gamma-\epsilon^2}{(\gamma+\epsilon)^2} \\ = \;& 1 + \tfrac{\frac{1}{2}\gamma^2+\gamma\log\frac{\gamma}{\epsilon}-\epsilon-\gamma}{(\gamma+\epsilon)^2}\epsilon \\ = \;& 1 + \tfrac{\frac{1}{2}(\gamma+\epsilon)^2+(\log\frac{\gamma}{\epsilon}-1-\epsilon)\gamma-\epsilon-\frac{1}{2}\epsilon^2}{(\gamma+\epsilon)^2}\epsilon \\ < \;& 1 + \tfrac{\epsilon}{2} + \tfrac{(\log\frac{\gamma}{\epsilon}-1-\epsilon)\gamma}{(\gamma+\epsilon)^2}\epsilon < 1 + \tfrac{3\epsilon}{2}. \end{aligned} \tag{A8}$$

From (A5) and (A6), and the fact that $\frac{dT(u)}{du}$ is an even function, we can conclude that for a sufficiently large $\gamma$, if $|u| > \frac{1}{2} + \frac{1}{\gamma}\log\frac{\gamma}{\epsilon}$, then $1 < \frac{dT(u)}{du} < 1 + \frac{3}{2}\epsilon$. The proof of **P3.c** is completed.

**P3.d**

The proof consists of two parts. First, we show the monotonic properties of $\frac{dT(u)}{du}$ for $|u| < \frac{1}{2} - \frac{1}{\gamma}\log\frac{\gamma}{\epsilon}$. Second, the upper and lower bounds of $\frac{dT(u)}{du}$ are established. Since $T(u)$ is an odd function and $\frac{dT(u)}{du}$ is an even function, we only need to consider the region of $u < \frac{1}{2} - \frac{1}{\gamma}\log\frac{\gamma}{\epsilon}$.

For $0 < u < \frac{1}{2} - \frac{1}{\gamma}\log\frac{\gamma}{\epsilon}$, $\mathrm{sign}(u) > 0$, $g(u)$ is monotonic decreasing, and $\exp(\gamma) > g(u) > \frac{\gamma}{\epsilon}$. In this region, the second derivative of $T(u)$ is

$$\begin{aligned} & \tfrac{d^2T(u)}{du^2} = \tfrac{2\gamma g(u)}{(1+g(u))^2} - \tfrac{\gamma^2 u g(u)}{(1+g(u))^2} + \tfrac{2\gamma^2 u g(u)^2}{(1+g(u))^3} \\ > \;& -\tfrac{\gamma^2 g(u)u}{(1+g(u))^2} + \tfrac{2\gamma^2 u g(u)^2}{(1+g(u))^3} \\ = \;& (g(u)-1)\cdot\tfrac{\gamma^2 g(u)u}{(1+g(u))^3}. \end{aligned} \tag{A9}$$

Obviously, in this region, $\frac{\gamma^2 g(u)u}{(1+g(u))^3} > 0$, and

$$g(u) - 1 > \frac{\gamma}{\epsilon} - 1 > 0. \tag{A10}$$

Hence, we can conclude that, for $0 < u < \frac{1}{2} - \frac{1}{\gamma}\log\frac{\gamma}{\epsilon}$, $d^2T(u)/du^2 > 0$ and then $\frac{dT(u)}{du}$ is monotonically increasing. For $u < \frac{1}{2} - \frac{1}{\gamma}\log\frac{\gamma}{\epsilon}$, the upper bound of $dT(u)/du$ exists when $u = \frac{1}{2} - \frac{1}{\gamma}\log\frac{\gamma}{\epsilon}$. The upper bound is

$$\begin{aligned}&\tfrac{dT(u)}{du}|_{u=\frac{1}{2}-\frac{1}{\gamma}\log\frac{\gamma}{\epsilon}} = \tfrac{1}{1+\frac{\gamma}{\epsilon}} + \tfrac{\gamma^2|u|}{\epsilon(1+\frac{\gamma}{\epsilon})^2} = \tfrac{\gamma^2|u|+\epsilon+\gamma}{(\gamma+\epsilon)^2}\epsilon\\ &< \tfrac{\frac{1}{2}\gamma^2+\epsilon+\gamma}{(\gamma+\epsilon)^2}\epsilon = \tfrac{1}{2}\epsilon + \tfrac{\gamma+\epsilon+-2\gamma\epsilon-\frac{1}{2}\epsilon^2}{(\gamma+\epsilon)^2}\epsilon\\ &< \tfrac{3}{2}\epsilon.\end{aligned} \tag{A11}$$

In addition, a lower bound on $\frac{dT(u)}{du}$ exists at $u = 0$,

$$\frac{dT(u)}{du}|_{u=0} = \frac{1}{1+\exp(\gamma)} > 0 \tag{A12}$$

To sum up, for a sufficiently large $\gamma$, if $u < \frac{1}{2} - \frac{1}{\gamma}\log\frac{\gamma}{\epsilon}$, then $0 < \frac{dT(u)}{du} < \frac{3}{2}\epsilon$. The proof of **P3.d** is completed.

## Appendix B. Proof of Property 4

The continuity and differentiability of $S(x)$ can be proved according to the following lemma [54].

**Lemma A1.** *(Inverse function) Let $\phi$ be a strictly monotone continuous function on $[a, b]$, with $\phi$ differentiable at $x_0 \in (a, b)$ and $\frac{d\phi}{dx}|_{x=x_0} \neq 0$. Then $\phi^{-1}$ exists and is continuous and strictly monotone. Moreover, $\phi^{-1}$ is differentiable at $y_0 = \phi(x_0)$ and*

$$\frac{d\phi^{-1}(y)}{dy}|_{y=y_0} = \frac{1}{\frac{d\phi}{dx}|_{x=x_0}}. \tag{A13}$$

From (A3), one can easily verify that $\frac{dT(u)}{du} > 0$, and that $T(u)$ is a strictly monotone increasing continuous function. Based on Lemma A1, $T^{-1}(x)$ exists. Additionally, it is continuous and differentiable at every point. Hence $\frac{1}{2}\partial S(x) = T^{-1}(x) - x$ is a continuous and differentiable function. The proof is completed.

## Appendix C. Proof of Property 5

**Boundness:** Now, we use the Taylor series of $S(x)$ to estimate the values of $S(\pm\frac{1}{2})$. Since $S(x)$ is an even function, the proof only presents the case of $u > \frac{1}{2} + \frac{1}{\gamma}\log\frac{1}{\epsilon}$.

First, we can use $x = \frac{1}{4}$ to obtain the Taylor series expansion of $\frac{1}{2}S(x)$:

$$\frac{1}{2}S(x) \approx \frac{1}{2}S(\frac{1}{4}) + \frac{1}{2}S'(\frac{1}{4})(x-\frac{1}{4}) + \frac{1}{2}\frac{1}{2}S''(\frac{1}{4})(x-\frac{1}{4})^2, \tag{A14}$$

where $S'(\frac{1}{4})$ and $S''(\frac{1}{4})$ are the first and second order derivatives of $S(x)$ at $x = \frac{1}{4}$, respectively. Since $\frac{1}{2}S'(x) = u - x = T^{-1}(x) - x$ and $T(\frac{1}{2}) = \frac{1}{4}$, we obtain

$$\frac{1}{2}S'(\frac{1}{4}) = \frac{1}{2} - \frac{1}{4} = \frac{1}{4}. \tag{A15}$$

Additionally, $\frac{1}{2}S''(\frac{1}{4}) = \frac{dT^{-1}(x)}{dx}|_{x=1/4} - 1$. As $T(\frac{1}{2}) = \frac{1}{4}$, from Lemma 2,

$$\frac{dT^{-1}(x)}{dx}|_{x=1/4} = \frac{1}{\frac{dT(u)}{du}|_{u=\frac{1}{2}}}. \tag{A16}$$

Additionally, from (A3),

$$\frac{dT(u)}{du}\Big|_{u=\frac{1}{2}} = \frac{2+\gamma}{4}. \tag{A17}$$

From (A14)–(A17), for a small $\epsilon$ and a sufficient large $\gamma$,

$$\begin{aligned} \frac{1}{2}S(x) &\approx \frac{1}{2}S(\frac{1}{4}) + \frac{1}{4}(x-\frac{1}{4}) + \frac{1}{2}(\frac{4}{2+\gamma}-1)(x-\frac{1}{4})^2 \\ &\approx \frac{1}{2}S(\frac{1}{2}) + \frac{1}{4}(x-\frac{1}{4}) - \frac{1}{2}(x-\frac{1}{4})^2. \end{aligned} \tag{A18}$$

Thus, we have

$$\frac{1}{2}S(\frac{1}{2}) \approx \frac{1}{2}S(\frac{1}{4}) + \frac{1}{32}, \text{ and } \frac{1}{2}S(0) \approx \frac{1}{2}S(\frac{1}{4}) - \frac{3}{32}. \tag{A19}$$

As $\frac{1}{2}S(0) = 0$, we obtain $\frac{1}{2}S(\frac{1}{2}) \approx \frac{1}{8}$. Similarly, it is also easy to obtain that $\frac{1}{2}S(-\frac{1}{2}) \approx \frac{1}{8}$.

Now we would like to know for $u > \frac{1}{2} + \frac{1}{\gamma}\log\frac{1}{\epsilon}$, i.e., $x > T(\frac{1}{2} + \frac{1}{\gamma}\log\frac{1}{\epsilon})$, what the value of $\frac{1}{2}S(x)$ is.

Let $u_0 = \frac{1}{2} + \frac{1}{\gamma}\log\frac{1}{\epsilon}$ and $x_0 = T(u_0)$. Since $S(x)$ is an even function and is monotonic increasing (for $x > 0$), for $x \geq x_0$, we have

$$\frac{1}{2}S(x) - \frac{1}{2}S(x_0) = \int_{x_0}^{x}(u-\chi)d\chi. \tag{A20}$$

From **P3**, we have $u \leq (1+\epsilon)x$. Thus,

$$\frac{1}{2}S(x) - \frac{1}{2}S(x_0) \leq \int_{x_0}^{x}\epsilon\chi d\chi. \tag{A21}$$

Thus, for a sufficient small $\epsilon$, $\frac{1}{2}S(x) - \frac{1}{2}S(x_0) \approx 0$. In addition, for sufficient small $\epsilon$ and sufficient large $\gamma$, $u_0 \approx \frac{1}{2}$. As $\frac{1}{2}S(\frac{1}{2}) \approx \frac{1}{8}$, we have $\frac{1}{2}S(x) \approx \frac{1}{8}$ for $u \geq \frac{1}{2} + \frac{1}{\gamma}\log\frac{1}{\epsilon}$.

**Sparsity:** Since $\frac{1}{2}S(x)$ is an even function, we only show the proof for $0 < x < T(\frac{1}{2} - \frac{1}{\gamma}\log\frac{1}{\epsilon})$, i.e., the case of $0 < u < \frac{1}{2} - \frac{1}{\gamma}\log\frac{1}{\epsilon}$. Let $u_0 = \frac{1}{2} - \frac{1}{\gamma}\log\frac{1}{\epsilon}$ and $x_0 = T(u_0)$. For $x > 0$, $\frac{1}{2}S(x)$ is a monotonical increasing. Hence, we have $0 = \frac{1}{2}S(0) < \frac{1}{2}S(x) < \frac{1}{2}S(x_0)$. Therefore, we have

$$\frac{1}{2}S(x_0) - \frac{1}{2}S(0) = \int_{0}^{x_0} u - \chi d\chi.$$

From **P3.b**, we can deduce that $0 < x_0 \leq \frac{\epsilon u_0}{1+\epsilon}$. For a sufficient small $\epsilon$, $x_0 \leq \frac{\epsilon u_0}{1+\epsilon} \approx 0$. Obviously, in this region $u - \chi$ is bounded. Hence, we have $\frac{1}{2}S(x_0) \approx 0$. As $0 = \frac{1}{2}S(0) < \frac{1}{2}S(x) < \frac{1}{2}S(x_0)$, we can say that $\frac{1}{2}S(x) \approx 0$ for $0 \leq x \leq x_0$. In conclusion, $\frac{1}{2}S(x) \approx 0$ if $|x| < T(\frac{1}{2} - \frac{1}{\eta}\log\frac{1}{\epsilon})$. The proof is completed.

## References


1. Chua, L.; Lin, G.N. Nonlinear programming without computation. *IEEE Trans. Circuits Syst.* **1984**, *31*, 182–188. [CrossRef]
2. Tank, D.; Hopfield, J. Simple 'neural' optimization networks: An A/D converter, signal decision circuit, and a linear programming circuit. *IEEE Trans. Circuits Syst.* **1986**, *33*, 533–541. [CrossRef]
3. Xia, Y.; Leung, H.; Wang, J. A projection neural network and its application to constrained optimization problems. *IEEE Trans. Circuits Syst. Fundam. Theory Appl.* **2002**, *49*, 447–458.
4. Xia, Y.; Wang, J. A general projection neural network for solving monotone variational inequalities and related optimization problems. *IEEE Trans. Neural Netw.* **2004**, *15*, 318–328. [CrossRef] [PubMed]
5. Wang, H.; Lee, C.M.; Feng, R.; Leung, C.S. An analog neural network approach for the least absolute shrinkage and selection operator problem. *Neural Comput. Appl.* **2018**, *29*, 389–400. [CrossRef]
6. Wang, Y.; Li, X.; Wang, J. A neurodynamic optimization approach to supervised feature selection via fractional programming. *Neural Netw.* **2021**, *136*, 194–206. [CrossRef]

7. Bouzerdoum, A.; Pattison, T.R. Neural network for quadratic optimization with bound constraints. *IEEE Trans. Neural Netw.* **1993**, *4*, 293–304. [CrossRef]
8. Feng, R.; Leung, C.S.; Constantinides, A.G.; Zeng, W.J. Lagrange Programming Neural Network for Nondifferentiable Optimization Problems in Sparse Approximation. *IEEE Trans. Neural Netw. Learn. Syst.* **2017**, *28*, 2395–2407. [CrossRef]
9. Wang, H.; Feng, R.; Leung, A.C.S.; Tsang, K.F. Lagrange programming neural network approaches for robust time-of-arrival localization. *Cogn. Comput.* **2018**, *10*, 23–34. [CrossRef]
10. Shi, Z.; Wang, H.; Leung, C.S.; So, H.C.; Member EURASIP. Robust MIMO radar target localization based on Lagrange programming neural network. *Signal Process.* **2020**, *174*, 107574. [CrossRef]
11. Liu, Q.; Wang, J. *L*_{1}-minimization algorithms for sparse signal reconstruction based on a projection neural network. *IEEE Trans. Neural Netw. Learn. Syst.* **2015**, *27*, 698–707. [CrossRef] [PubMed]
12. Yan, Z.; Le, X.; Wen, S.; Lu, J. A Continuous-Time Recurrent Neural Network for Sparse Signal Reconstruction Via $\ell_1$ Minimization. In Proceedings of the 2018 Eighth International Conference on Information Science and Technology (ICIST), Cordoba, Granada, and Seville, Spain, 30 June–6 July 2018; pp. 43–49. [CrossRef]
13. Wen, H.; Wang, H.; He, X. A Neurodynamic Algorithm for Sparse Signal Reconstruction with Finite-Time Convergence. *Circuits Syst. Signal Process.* **2020**, *39*, 6058–6072. [CrossRef]
14. Donoho, D.; Huo, X. Uncertainty principles and ideal atomic decomposition. *IEEE Trans. Inf. Theory* **1999**, *47*, 2845–2862. [CrossRef]
15. Donoho, D.L.; Elad, M. Optimally sparse representation in general (nonorthogonal) dictionaries via $l^1$ minimization. *Proc. Natl. Acad. Sci. USA* **2003**, *100*, 2197–2202. [CrossRef] [PubMed]
16. Chartrand, R. Exact reconstruction of sparse signals via nonconvex minimization. *IEEE Signal Process. Lett.* **2007**, *14*, 707–710. [CrossRef]
17. Blumensath, T.; Davies, M.E. Iterative thresholding for sparse approximations. *J. Fourier Anal. Appl.* **2008**, *14*, 629–654. [CrossRef]
18. Jin, D.; Yang, G.; Li, Z.; Liu, H. Sparse recovery algorithm for compressed sensing using smoothed $\ell_0$-norm and randomized coordinate descent. *Mathematics* **2019**, *7*, 834. [CrossRef]
19. Stanković, L.; Sejdić, E.; Stanković, S.; Daković, M.; Orović, I. A tutorial on sparse signal reconstruction and its applications in signal processing. *Circuits Syst. Signal Process.* **2019**, *38*, 1206–1263. [CrossRef]
20. Stanković, I.; Ioana, C.; Daković, M. On the reconstruction of nonsparse time-frequency signals with sparsity constraint from a reduced set of samples. *Signal Process.* **2018**, *142*, 480–484. [CrossRef]
21. Dai, C.; Che, H.; Leung, M.F. A neurodynamic optimization approach for *L*1 minimization with application to compressed image reconstruction. *Int. J. Artif. Intell. Tools* **2021**, *30*, 2140007. [CrossRef]
22. Bioucas-Dias, J.M.; Figueiredo, M.A. A new TwIST: Two-step iterative shrinkage/thresholding algorithms for image restoration. *IEEE Trans. Image Process.* **2007**, *16*, 2992–3004. [CrossRef] [PubMed]
23. Kong, X.; Zhao, Y.; Xue, J.; Chan, J.C.W.; Kong, S.G. Global and local tensor sparse approximation models for hyperspectral image destriping. *Remote Sens.* **2020**, *12*, 704. [CrossRef]
24. Costanzo, S.; Rocha, Á.; Migliore, M.D. Compressed sensing: Applications in radar and communications. *Sci. World J.* **2016**, *2016*, 5407415. [CrossRef] [PubMed]
25. Li, S.; Zhao, G.; Zhang, W.; Qiu, Q.; Sun, H. ISAR imaging by two-dimensional convex optimization-based compressive sensing. *IEEE Sens. J.* **2016**, *16*, 7088–7093. [CrossRef]
26. Craven, D.; McGinley, B.; Kilmartin, L.; Glavin, M.; Jones, E. Compressed sensing for bioelectric signals: A review. *IEEE J. Biomed. Health Inform.* **2014**, *19*, 529–540. [CrossRef]
27. Chen, S.S.; Donoho, D.L.; Saunders, M.A. Atomic decomposition by basis pursuit. *SIAM Rev.* **2001**, *43*, 129–159. [CrossRef]
28. Osborne, M.R.; Presnell, B.; Turlach, B.A. A new approach to variable selection in least squares problems. *IMA J. Numer. Anal.* **2000**, *20*, 389–403. [CrossRef]
29. Candes, E.; Romberg, J. l1-Magic. 2007. Available online: https://candes.su.domains/software/l1magic/downloads/l1magic.pdf (accessed on 1 December 2022)
30. van den Berg, E.; Friedlander, M.P. SPGL1: A Solver for Sparse Least Squares. 2007. Available online: https://friedlander.io/spgl1/ (accessed on 1 December 2022).
31. Rozell, C.J.; Johnson, D.H.; Baraniuk, R.G.; Olshausen, B.A. Sparse coding via thresholding and local competition in neural circuits. *Neural Comput.* **2008**, *20*, 2526–2563. [CrossRef]
32. Engan, K.; Rao, B.D.; Kreutz-Delgado, K. Regularized FOCUSS for subset selection in noise. In Proceedings of the NORSIG 2000, Kolmarden, Sweden, 13–15 June 2000; pp. 247–250.
33. Rao, B.D.; Kreutz-Delgado, K. An affine scaling methodology for best basis selection. *IEEE Trans. Signal Process.* **1999**, *47*, 187–200. [CrossRef]
34. Zhang, C.H. Nearly unbiased variable selection under minimax concave penalty. *Ann. Stat.* **2010**, *38*, 894–942. [CrossRef]
35. Breheny, P.; Huang, J. Coordinate descent algorithms for nonconvex penalized regression, with applications to biological feature selection. *Ann. Appl. Stat.* **2011**, *5*, 232. [CrossRef] [PubMed]
36. Blumensath, T.; Davies, M.E. Normalized iterative hard thresholding: Guaranteed stability and performance. *IEEE J. Sel. Top. Signal Process.* **2010**, *4*, 298–309. [CrossRef]

37. Donoho, D.L.; Maleki, A.; Montanari, A. Message-passing algorithms for compressed sensing. *Proc. Natl. Acad. Sci. USA* **2009**, *106*, 18914–18919. [CrossRef] [PubMed]
38. Jin, J.; Gu, Y.; Mei, S. A stochastic gradient approach on compressive sensing signal reconstruction based on adaptive filtering framework. *IEEE J. Sel. Top. Signal Process.* **2010**, *4*, 409–420. [CrossRef]
39. Boyd, S.; Parikh, N.; Chu, E.; Peleato, B.; Eckstein, J. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Found. Trends Mach. Learn.* **2011**, *3*, 1–122. [CrossRef]
40. Song, C.; Xia, S.T. Alternating direction algorithms for \ell_0 regularization in compressed sensing. *arXiv* **2016**, arXiv:1604.04424.
41. Qiu, K.; Dogandzic, A. ECME thresholding methods for sparse signal reconstruction. *arXiv* **2010**, arXiv:1004.4880.
42. Zhang, S.; Constantinidies, A.G. Lagrange Programming Neural Networks. *IEEE Trans. Circuits Syst. II* **1992**, *39*, 441–452. [CrossRef]
43. Shi, Z.; Wang, H.; Leung, C.S.; So, H.C.; Liang, J.; Tsang, K.F.; Constantinides, A.G. Robust ellipse fitting based on Lagrange programming neural network and locally competitive algorithm. *Neurocomputing* **2020**, *399*, 399–413. [CrossRef]
44. Balavoine, A.; Rozell, C.; Romberg, J. Global convergence of the locally competitive algorithm. In Proceedings of the IEEE Signal Processing Education Workshop (DSP/SPE) 2011, Sedona, AZ, USA, 4–7 January 2011; pp. 431–436.
45. Pandey, O. Operational Amplifier (Op-Amp). In *Electronics Engineering*; Springer: Berlin/Heidelberg, Germany, 2022; pp. 233–270.
46. Bult, K.; Wallinga, H. A CMOS four-quadrant analog multiplier. *IEEE J. Solid-State Circuits* **1986**, *21*, 430–435. [CrossRef]
47. Chen, C.; Li, Z. A low-power CMOS analog multiplier. *IEEE Trans. Circuits Syst. II Express Briefs* **2006**, *53*, 100–104. [CrossRef]
48. Filanovsky, I.; Baltes, H. Simple CMOS analog square-rooting and squaring circuits. *IEEE Trans. Circuits Syst. I Fundam. Theory Appl.* **1992**, *39*, 312–315. [CrossRef]
49. Sakul, C. A new CMOS squaring circuit using voltage/current input. In Proceedings of the 23rd International Technical Conference on Circuits/Systems, Computers and Communications ITC-CSCC, Phuket, Thailand, 5–8 July 2008; pp. 525–528.
50. van den Berg, E.; Friedlander, M.P. Probing the Pareto frontier for basis pursuit solutions. *SIAM J. Sci. Comput.* **2008**, *31*, 890–912. [CrossRef]
51. Ji, S.; Xue, Y.; Carin, L. Bayesian compressive sensing. *IEEE Trans. Signal Process.* **2008**, *56*, 2346–2356. [CrossRef]
52. Vadivel, R.; Hammachukiattikul, P.; Zhu, Q.; Gunasekaran, N. Event-triggered synchronization for stochastic delayed neural networks: Passivity and passification case. *Asian J. Control* **2022**. [CrossRef]
53. Chanthorn, P.; Rajchakit, G.; Humphries, U.; Kaewmesri, P.; Sriraman, R.; Lim, C.P. A delay-dividing approach to robust stability of uncertain stochastic complex-valued hopfield delayed neural networks. *Symmetry* **2020**, *12*, 683. [CrossRef]
54. Spruck, J. Strictly Monotone Functions and the Inverse Function Theorem. Lecture Notes in MATH–405. Available online: https://math.jhu.edu/~js/math405/405.monotone.pdf (accessed on 24 November 2021).