An $\ell_0$-Norm-Based Centers Selection for Failure Tolerant RBF Networks

HAO WANG, ZHANGLEI SHI, HIU TUNG WONG, CHI-SING LEUNG, (Senior Member, IEEE), HING CHEUNG SO, (Fellow, IEEE), AND RUIBIN FENG

Department of Electrical Engineering, City University of Hong Kong, Hong Kong

Corresponding author: Chi-Sing Leung (eleungc@cityu.edu.hk)

The work was supported by the City University of Hong Kong under Grant 7005063 and Grant 9610431.

ABSTRACT There are two important issues in the construction of a radial basis function (RBF) neural network. The first one is to select suitable RBF centers. The second one is that the resultant RBF network should be with good fault tolerance. This paper proposes an algorithm that is able to select RBF centers and to train fault tolerant RBF networks simultaneously. The proposed algorithm borrows the concept from sparse approximation. In our formulation, we first define a fault tolerant objective function based on all input vectors from the training samples. We then introduce the minimax concave penalty (MCP) function, which is an approximation of $\ell_0$-norm, into the objective function. The MCP term is able to force some unimportant RBF weights to zero. Hence the RBF node selection process can be achieved during training. As the MCP function is nondifferentiable and nonconvex, traditional gradient descent based algorithms are still unable to minimize the modified objective function. Based on the alternating direction method of multipliers (ADMM) framework, we develop an algorithm, called ADMM-MCP, to minimize the modified objective function. The convergent proof of the proposed ADMM-MCP algorithm is also presented. Simulation results show that the proposed ADMM-MCP algorithm is superior to many existing center selection algorithms under the concurrent fault situation.

INDEX TERMS Failure tolerant, RBF, center selection, ADMM, $\ell_0$-norm, global convergence.

I. INTRODUCTION Constructing a radial basis function (RBF) network [1]–[3] includes two key issues. The first issue is to select some suitable RBF centers. The second issue is to determine the RBF weights. There are many ways to perform RBF center selection. We can select all the input vectors from the training samples as RBF centers [4]. But this method may create a network with unnecessarily complicated structure. Another way is to randomly select a subset of the input vectors as the RBF centers, but this simple method cannot ensure that the constructed RBFs cover the input space well [5]. Other advanced methods include clustering algorithms [6], orthogonal least squares (OLS) approach [7], [8], and support vector regression (SVR) [9], [10]. However, many algorithms in this area did not consider the fault tolerant issue.

Biological neural networks have the capability to tolerate fault or noise conditions [11]. For example, human can recognize an object from a noisy image/video. In addition, when a few neural cells or synapses malfunction, human brain can still works properly. Since the concept of neural networks comes from biological neural networks, researchers expected that a trained neural network has a capability to tolerate weight or neuron failure. However, many literatures reported that when fault/noise tolerant procedures are not introduced during training, the faulty version of a well trained network may have a poor performance [12]–[15].

In realizing a neural network, we face some practical issues [16]. When we use analog technology to realize a dot-product operation, the accuracy is affected by the offset voltage of operational amplifiers [17]. In addition, for analog components, we usually define their accuracy in terms of percentage of error [18]. In this case, we can use multiplicative noise to model the error. For digital technology realization, when a weight is represented in the floating...
point format, the round-off errors happen and they can be described by the multiplicative noise model [19]. Apart from multiplicative noise, physical fault may happen [20]. It blocks the data/signal transmission between two connected neurons. Besides, nowadays, the very large scale integration (VLSI) implementation could be at nano-scale and transient noise/failure may happen [21].

In the last twenty years, several fault tolerant algorithms were proposed [13], [14], [22]–[24]. However, most of them assume that a trained network is affected by one kind of fault conditions only. For instance, in [13], [14], only the open node fault model was considered. Recently, [25] first described the concurrent fault situation in which a trained network is affected by multiplicative weight noise and open weight fault concurrently. However, the weight decay term in [25] is not optimal for fault tolerance situation. Later, another approach was proposed in [19]. It is based on regularization and OLS center selection. The performances of this algorithm are better than those of many existing methods. Due to the OLS approach, it cannot select centers and train an RBF network at the same time. To improve the performances of the network and complete the two steps simultaneously, an \( \ell_0 \)-norm based fault tolerant RBF center selection method [26] was proposed recently.

As the \( \ell_0 \)-norm is a much better approach in compressive sensing for retaining nonzero elements, this paper investigates to use the \( \ell_0 \)-norm to replace the \( \ell_1 \)-norm for center selection. Since the \( \ell_0 \)-norm is a noncontinuous function, it is difficult to design an algorithm to minimize an \( \ell_0 \)-norm based objective function. This paper introduces the minimax concave penalty (MCP) function [27], [28], which is an approximation of \( \ell_0 \)-norm, into the objective function. Since the MCP function is able to limit the number of RBF nodes used, minimizing the modified objective function can remove some unimportant RBF nodes during training. However, the MCP function is still nondifferentiable and nonconvex, traditional gradient descent like algorithms are unable to minimize the modified objective function. Based on the alternating direction method of multipliers (ADMM) framework [29], we develop an algorithm, called ADMM-MCP, to minimize the modified objective function. The ADMM framework breaks down the minimization problem into three parts. Each part can be solved in a much easier way, even though some of them contain nonconvex and nondifferentiable terms. A theoretical analysis of the convergence of the proposed ADMM-MCP algorithm is then provided. Simulation results show that the proposed ADMM-MCP algorithm is superior to many existing center selection algorithms under the concurrent fault situation.

The contributions of this paper are as follows.

- We show that the training weight vector converges to a limit point. Besides, the limit point is a stationary point of the Lagrangian function of the objective function.
- Simulation shows that the performances of the proposed algorithm are better than those of many existing training algorithms. Besides, from the paired t-test result, the improvement of using the proposed algorithm over other algorithms is statistically significant.

The rest of this paper is organized as follows. The backgrounds of the ADMM framework and RBF neural networks under the concurrent fault situation are presented in Section II. In Section III, the proposed algorithm is developed. Its convergent property is presented in Section IV. Simulation results are provided in the Section V. Finally, the concluding remark is drawn in Section VI.

II. BACKGROUND

A. NOTATION

We use a lower-case or upper-case letter to represent a scalar while vectors and matrices are denoted by bold lower-case and upper-case letters, respectively. The transpose operator is denoted as \((\cdot)^T\), and \( I \) represents the identity matrix with appropriate dimensions. Other mathematical symbols are defined in their first appearance.

B. RBF NETWORKS UNDER CONCURRENT FAULT SITUATION

In this paper, the training set is expressed as

\[
\mathcal{D} = \left\{ \left( \mathbf{x}_i, y_i \right) : \mathbf{x}_i \in \mathbb{R}^K, y_i \in \mathbb{R}, i = 1, \cdots, N \right\} ,
\]

where \( \mathbf{x}_i \) is the input vector of the \( i \)-th sample with dimension \( K \), and \( y_i \) is the corresponding output. Similarly, the test set is expressed as

\[
\mathcal{D'} = \left\{ \left( \mathbf{x}'_i, y'_i \right) : \mathbf{x}'_i \in \mathbb{R}^K, y'_i \in \mathbb{R}, i' = 1, \cdots, N' \right\} .
\]

In the RBF approach, the input-output relationship of the data set is approximated by a weighted sum of the outputs of \( M \) RBFs, given by

\[
f(\mathbf{x}) = \sum_{j=1}^{M} w_j a_j(\mathbf{x}) = \sum_{j=1}^{M} w_j \exp \left( -\frac{\| \mathbf{x} - \mathbf{c}_j \|_2^2}{s} \right) ,
\]

where \( a_j(\mathbf{x}) = \exp \left( -\frac{\| \mathbf{x} - \mathbf{c}_j \|_2^2}{s} \right) \) is the output of the \( j \)-th RBF node, \( w_j \) is the corresponding weight, \( \mathbf{c}_j \) is the center of the \( j \)-th RBF node, and \( s \) is a parameter which controls the RBF width. Usually, the RBF centers are selected from the training input vectors \( \{ \mathbf{x}_1, \cdots, \mathbf{x}_N \} \). If we use all training input vectors as centers, the resultant network will have serious overfitting problem for faultless situation. In addition, using all the training input vectors is waste of resources. Therefore, center selection is a key step in the training of an RBF network.
For a fault-free network, the training set mean square error (MSE) is given by
\[ E_{\text{train}} = \frac{1}{N} \| y - Aw \|_2^2, \]
(4)
where \( w = [w_1, \cdots, w_M]^T, y = [y_1, \cdots, y_N]^T \), \( A \) is an \( N \times M \) matrix, and the \((i,j)\) entry of \( A \) is given by
\[ [A]_{i,j} = a_j(x_i) = \exp \left( -\frac{\| x_i - c_j \|_2^2}{s} \right). \]
(5)

In the implementation of an RBF network, weight failure may happen. Multiplicative weight noise and open weight fault are two common fault in the RBF network [18, 22–24, 30–33]. When they occur concurrently [19, 25], the implemented weights can be described by
\[ \bar{w}_j = (w_j + b_jw_j) \beta_j, \quad \forall j = 1, \cdots, M. \]
(6)

In (6), \( \beta_j \)'s are variables that describe whether the weights are opened or not. When a weight \( \beta_j \) is opened, \( \beta_j \) is equal to 0. Otherwise, \( \beta_j \) is equal to 1. This paper assumes that \( \beta_j \)'s are independently identically distributed (i.i.d.) binary random variables. The probability mass function of \( \beta_j \) is given by
\[ \text{Prob}(\beta_j) = \begin{cases} \rho, & \text{for } \beta_j = 0, \\ 1 - \rho, & \text{for } \beta_j = 1. \end{cases} \]
(7)
The statistics of \( \beta_j \)'s are then given by
\[ \langle \beta_j \rangle = \langle \beta_j^2 \rangle = 1 - \rho, \quad \langle \beta_j \beta_{j'} \rangle = (1 - \rho)^2, \quad \forall j \neq j'. \]
(9a)

The term \( b_jw_j \) in (6) is the multiplicative noise. It can be seen that the magnitude of the noise is proportional to that of the weight. This paper assumes that \( b_j \) are i.i.d. zero-mean random variables with variance \( \sigma_b^2 \). With this assumption, the statistics of \( b_j \)'s are summarized as
\[ \langle b_j \rangle = 0, \quad \langle b_j^2 \rangle = \sigma_b^2, \quad \forall j \neq j'. \]
(10a)

where \( \langle \cdot \rangle \) is the expectation operator.

Given a particular fault pattern, the training set MSE is
\[ \bar{E}_{\text{train}} = \frac{1}{N} \| \bar{y} - A\bar{w} \|_2^2, \]
\[ = \frac{1}{N} \sum_{i=1}^{N} \left[ y_i^2 - 2y_i \sum_{j=1}^{M} \beta_jw_ja_j(x_i) \ight. \\
+ \sum_{j=1}^{M} \sum_{j=1}^{M} \beta_j \beta_{j'} w_jw_j(1 + b_jb_{j'}a_j(x_i)a_j(x_i)) \\
\left. + \sum_{j=1}^{M} \sum_{j=1}^{M} (b_j + b_{j'}) \beta_j \beta_{j'} w_jw_j a_j(x_i)a_j(x_i) \right.
\right. \\
\left. - 2y_i \sum_{j=1}^{M} b_jw_ja_j(x_i) \right]. \]
(11)

From (10) and (9), the average training set MSE [19] over all possible fault patterns is given by
\[ \bar{E}_{\text{train}} = \left\langle \bar{E}_{\text{train}} \right\rangle = \frac{P_{\beta}}{N} \sum_{i=1}^{N} y_i^2 + \frac{1 - P_{\beta}}{N} \| y - Aw \|_2^2 \\
+ \frac{1}{N} P_{\beta} w^T \left[(P_{\beta} + \sigma_b^2) \text{diag} \left( A^T A \right) - P_{\beta} AA^T \right]w. \]
(12)

In (12), the term \( P_{\beta} \sum_{i=1}^{N} y_i^2 \) is independent of the weight vector \( w \). Hence the training objective can be defined as
\[ \psi(w) = \frac{1}{N} \| y - Aw \|_2^2 + w^T R w, \]
(13)
where \( R = (P_{\beta} + \sigma_b^2) \text{diag} \left( \frac{1}{N} AA^T \right) - \frac{P_{\beta}}{N} AA^T. \)

C. ADMM

The ADMM framework solves optimization problems by breaking them into smaller pieces [29]. Suppose we have the following objective function:
\[ Q(z) = \psi(z) + g(z), \]
(14)
where \( z \in \mathbb{R}^n \). The objective function can be separated into two terms: \( \psi(z) \) and \( g(z) \). If the term \( g(z) \) is nonconvex and nondifferentiable, then it is difficult to minimize \( Q(z) \) directly. The ADMM framework introduces a dummy vector \( y \in \mathbb{R}^n \) and reformulates the minimization problem as a constrained optimization problem, given by
\[ \min_{z,y} : \psi(z) + g(y) \quad \text{s.t.} \quad z = y. \]
(15a)

We then construct an augmented Lagrangian function:
\[ L(z, y, \alpha) = \psi(z) + g(y) + \alpha^T (z - y) + \frac{\rho}{2} \| z - y \|_2^2, \]
(16)
where \( \alpha \in \mathbb{R}^p \) is the Lagrange multiplier vector, and \( \rho > 0 \) is a parameter that affects the convergent speed. The algorithm consists of three steps, given by
\[ y^{k+1} = \arg \min_y L(z^k, y, \alpha^k), \]
(17a)
\[ z^{k+1} = \arg \min_z L(z, y^{k+1}, \alpha^k), \]
(17b)
\[ \alpha^{k+1} = \alpha^k + \rho (z^{k+1} - y^{k+1}). \]
(17c)
It should be noticed that for many forms of \( g(y) \), we have closed form solutions for (17a), even though \( g(y) \) is nonconvex and nondifferentiable.

III. DEVELOPMENT OF ADMM-MCP

In (13), we discuss to use \( M \) RBF centers, \( \{c_1, \cdots, c_M\} \), to construct a network. However, we do not discuss the way to create them yet. Suppose that we use all the training input
vectors, \( \{x_1, \ldots, x_N\} \), as the RBF centers. The expression of the training objective does not change and is still given by

\[
\psi(w) = \frac{1}{N} ||y - Aw||_2^2 + w^T R w. \tag{18}
\]

However, the definitions of \( \mathbf{w} \) and \( \mathbf{R} \) are changed to

\[
\mathbf{w} = [w_1, \ldots, w_N]^T, \tag{19a}
\]

\[
[A]_{i,j} = a_i(x_i) = \exp\left(- \frac{\|x_i - x_j\|_2^2}{s^2}\right), \tag{19b}
\]

where \( i = 1, \ldots, N \), \( j = 1, \ldots, N \), and \( \mathbf{A} \) is an \( N \times N \) matrix.

In the rest of this section, we will develop the ADMM-MCP algorithm, in which we pack an approximated \( \ell_0 \) term, namely MCP term, into the objective function stated in (18). The packed MCP term has an ability to force some RBF weights to zero. Hence during training, the center selection process is achieved automatically.

A. OBJECTIVE FUNCTION AND ADMM FORMULATION

We introduce an additional \( \ell_0 \) penalty term into (13), given by

\[
Q_0(\mathbf{w}, \lambda) = \frac{1}{N} ||y - Aw||_2^2 + w^T R w + \lambda ||\mathbf{w}||_0. \tag{20}
\]

where \( ||\mathbf{w}||_0 \) is the \( \ell_0 \)-norm of the weight vector and it represents the number of nonzero entries in the vector \( \mathbf{w} \). Parameter \( \lambda \) is a regularization parameter that controls the number of RBF nodes in the resultant network. Strictly speaking, the \( \ell_0 \)-norm is not a norm. Due to the nature of the \( \ell_0 \)-norm term, the problem stated in (20) is NP hard [34].

Inspired by [27], [28], the MCP function is a very attractive approximation of the \( \ell_0 \)-norm. Hence, we modify the objective function stated in (20) as

\[
Q_{mcp}(\mathbf{w}, \lambda) = \frac{1}{N} ||y - Aw||_2^2 + w^T R w + P_{\lambda, \gamma}(\mathbf{w}), \tag{21}
\]

where \( P_{\lambda, \gamma}(\mathbf{w}) = \sum_{i=1}^{M} P_{\lambda, \gamma}(w_i) \) \( (\lambda > 0, \gamma > 0) \) denotes the MCP function, given by

\[
P_{\lambda, \gamma}(w_i) = \begin{cases} 
\lambda |w_i| - \frac{w_i^2}{2\gamma}, & \text{if } |w_i| \leq \gamma \lambda, \\
\frac{1}{2\gamma} \gamma^2, & \text{if } |w_i| > \gamma \lambda.
\end{cases} \tag{22}
\]

The shape of the MCP penalty function with various settings is shown in Figure 1. Although the form of \( Q_{mcp}(\mathbf{w}, \lambda) \) has better property, we do not have a closed form solution for minimizing \( Q_{mcp}(\mathbf{w}, \lambda) \) directly.

We use the ADMM framework to minimize the objective function \( Q_{mcp}(\mathbf{w}, \lambda) \). Firstly, we introduce a dummy variable vector \( \mathbf{u} = [u_1, \ldots, u_N]^T \) and transform the unconstrained problem, stated in (21), into the standard ADMM form, given by

\[
\min_{\mathbf{w}, \mathbf{u}} \psi(\mathbf{w}) + P_{\lambda, \gamma}(\mathbf{u}), \tag{23a}
\]

\[
s.t. \mathbf{u} = \mathbf{w}, \tag{23b}
\]

where

\[
\psi(\mathbf{w}) = \frac{1}{N} ||y - Aw||_2^2 + w^T R w. \tag{24}
\]

We then construct the augmented Lagrangian as

\[
L(\mathbf{w}, \mathbf{u}, \nu) = Q_{mcp}(\mathbf{w}, \lambda) + \nu^T (\mathbf{u} - \mathbf{w}) + \frac{\rho}{2} ||\mathbf{w} - \mathbf{u}||_2^2,
\]

\[
= \psi(\mathbf{w}) + P_{\lambda, \gamma}(\mathbf{u}) + \nu^T (\mathbf{u} - \mathbf{w}) + \frac{\rho}{2} ||\mathbf{w} - \mathbf{u}||_2^2. \tag{25}
\]

B. UPDATING EQUATIONS

- According to (17a), the ADMM iteration of \( \mathbf{u}^{k+1} \) is

\[
\mathbf{u}^{k+1} = \arg \min_{\mathbf{u}} L(\mathbf{w}^k, \mathbf{u}, \nu^k),
\]

\[
= \arg \min_{\mathbf{u}} P_{\lambda, \gamma}(\mathbf{u}) + \nu^T (\mathbf{u} - \mathbf{w}^k) + \frac{\rho}{2} ||\mathbf{w}^k - \mathbf{u}||_2^2,
\]

\[
= \arg \min_{\mathbf{u}} P_{\lambda, \gamma}(\mathbf{u}) + \frac{\rho}{2} ||\mathbf{w}^k - \mathbf{u} - \frac{1}{\rho} \nu^k||_2^2. \tag{26}
\]

where \( \mathbf{u}^{k+1} = [u_1^{k+1}, \ldots, u_N^{k+1}]^T \).

- For (26), when \( \rho \gamma > 1 \), the closed form solution [27], [28] is given by

\[
u_i^{k+1} = \begin{cases} 
S(w_i^k - v_i^k/\rho, \lambda), & \text{if } |w_i^k - v_i^k/\rho| \leq \gamma \lambda, \\
1 - 1/\gamma \frac{w_i^k - v_i^k/\rho}{|w_i^k - v_i^k/\rho|}, & \text{if } |w_i^k - v_i^k/\rho| > \gamma \lambda.
\end{cases} \tag{27}
\]

for \( i = 1, \ldots, N \), where \( S \) denotes the soft-threshold operator [35],

\[
S(z, \lambda) = \text{sign}(z) \max(|z| - \lambda, 0). \tag{28}
\]

It is worth noting that when \( \gamma \to \infty \), \( S(\cdot, \cdot) \) is a soft-threshold function. When \( \gamma \to 1 \), \( S(\cdot, \cdot) \) is a hard-threshold function.

- When \( \rho \gamma = 1 \), the closed form solution of (26) is given by

\[
u_i^{k+1} = \begin{cases} 
0, & \text{if } |w_i^k - v_i^k/\rho| \leq \gamma \lambda, \\
w_i^k - v_i^k/\rho, & \text{if } |w_i^k - v_i^k/\rho| > \gamma \lambda.
\end{cases} \tag{29}
\]
When $\rho \gamma < 1$, the closed form solution of (26) is given by
\begin{equation}
\begin{aligned}
u_{k+1}^t &= \begin{cases}
0, & \text{if } |\psi_{t}^k - \psi_{t}^1| / \rho \leq \sqrt{\gamma / \rho \lambda}, \\
\psi_{t}^k - \psi_{t}^1 / \rho, & \text{if } |\psi_{t}^k - \psi_{t}^1| / \rho > \sqrt{\gamma / \rho \lambda}.
\end{cases}
\end{aligned}
\end{equation}

• According to (17b), the ADMM iteration of $w_{k+1}$ is given by
\begin{equation}
\begin{aligned}
w_{k+1} &= \arg\min_w L(w, u_{k+1}, v_{k}) \\
&= \arg\min_w \psi(w) + P_{\lambda, r}(u_{k+1}) + \nu_{k+1}^T (u_{k+1} - w) + \frac{\rho}{2} \left\| w - u_{k+1} \right\|^2_2 \\
&= \arg\min_w \psi(w) + \frac{\rho}{2} \left\| w - u_{k+1} - \frac{1}{\rho} \nu_k \right\|^2_2 \\
&= \left[ \frac{1}{N} A^T A + 2R + \rho I \right]^{-1} \left[ \frac{1}{N} A^T y + \rho u_{k+1} + \nu_k \right].
\end{aligned}
\end{equation}

• From (17c), $v_{k+1}$ is updated as
\begin{equation}
v_{k+1} = v_k + \rho \left( u_{k+1} - w_{k+1} \right).
\end{equation}

IV. ANALYSIS OF CONVERGENCE

This section presents the convergent properties of the ADMM-MCP algorithm. We use the general convergence result of nonconvex ADMM given by [36]. When we use the general convergence proof of nonconvex ADMM, we need to show that our algorithm satisfies three conditions. The general convergence result is given by Theorem 1 [36].

Theorem 1: If an ADMM based algorithm satisfies the following three conditions stated below, then the sequence $\{w_k, u_k, v_k\}$ has at least a limit point $\{w^*, u^*, v^*\}$ and any limit point $\{w^*, u^*, v^*\}$ is a stationary point of the Lagrangian function.

C1 (Sufficient decrease condition) For each $k$, there exists $\tau_1 > 0$ such that
\begin{equation}
L(w_{k+1}^l, u_{k+1}^l, v_{k+1}^l) - L(w_k^l, u_k^l, v_k^l) \leq -\tau_1 \|w_{k+1}^l - w_k^l\|_2^2.
\end{equation}

C2 (Boundness condition) The sequence $\{w_k, u_k, v_k\}$ is bounded and its Lagrangian function is lower bounded.

C3 (Subgradient bound condition) For each $k \in \mathbb{N}$, there exists $d_{k+1}^l \in \partial L(w_{k+1}^l, u_{k+1}^l, v_{k+1}^l)$, and $\tau_2 > 0$ such that
\begin{equation}
\|d_{k+1}^l\|^2_2 \leq \tau_2 \|w_{k+1}^l - w_k^l\|_2^2.
\end{equation}

Proof: The proof of the theorem is in Proposition 2 in [36] and Theorem 2.9 in [37].

In the rest of this section, we will show that the proposed algorithm satisfies the three conditions of Theorem 1.

Proposition 1: If $\rho$ is greater than a certain value, the ADMM-MCP algorithm satisfies the sufficient decrease condition in C1.
To ensure $l^2_\phi / \rho - a/2 < 0$, we need $\rho > 2l^2_\phi / a$. Hence the
$\tau_1 = a/2 - l^2_\phi / \rho$ in C1. The proof for C1 is completed. 

Now, we show that the proposed algorithm satisfies C2 in
Theorem 1.

Proposition 2: If $\rho \geq l_\phi$, then $L(w^k, u^k, v^k)$ is bounded
for all $k$, and $L(w^k, u^k, v^k)$ converges, as $k \to \infty$. In addition, the sequence $\{w^k, u^k, v^k\}$ is bounded.

Proof: The proof consists of two parts. The first one is that $L(w^k, u^k, v^k)$ is bounded. The second one is that the sequence $\{w^k, u^k, v^k\}$ is bounded.

The proof for $L(w^k, u^k, v^k)$ being bounded:
First, we prove that $L(w^k, u^k, v^k)$ is lower bounded for all $k$.
From (38), $\nabla \psi(w^k) = w^k$. Thus

$$L(w^k, u^k, v^k) = \psi(w^k) + P_{\lambda, \gamma}(u^k) + v^k \nabla \psi(w^k) - \frac{\rho}{2} \|w^k - u^k\|^2.\tag{43}$$

From Lemma 3.1 in [37] and the Lipschitz continuous gradient of $\psi(w)$,

$$\psi(w^k) + \nabla \psi(w^k)^T (u^k - w^k) \geq \psi(u^k) - \frac{l_\phi}{2} \|u^k - w^k\|^2.\tag{44}$$

Hence, we have

$$L(w^k, u^k, v^k) \geq \psi(u^k) + P_{\lambda, \gamma}(u^k) + \left( \frac{\rho}{2} - \frac{l_\phi}{2} \right) \|u^k - w^k\|^2.\tag{45}$$

Obviously, if $\rho \geq l_\phi$, then the right-hand side of (45) is greater than $-\infty$. Hence $L(w^k, u^k, v^k)$ is lower bounded. According to the proof of Proposition 1, we know that $L(w^k, u^k, v^k)$ is sufficient descent. Hence $L(w^k, u^k, v^k)$ is upper bounded by $L(w^0, u^0, v^0)$.

The proof for $\{w^k, u^k, v^k\}$ being bounded:
Next, we prove that the sequence $\{w^k, u^k, v^k\}$ is bounded. From (42), we have

$$L(w^{k+1}, u^{k+1}, v^{k+1}) - L(w^k, u^k, v^k) \leq - \tau_1 \|w^{k+1} - w^k\|^2,\tag{46}$$

where $\tau_1 > 0$. Hence, we have

$$\|w^{k+1} - w^k\|^2 \leq \frac{1}{\tau_1} \left( L(w^0, u^0, v^0) - L(w^{k+1}, u^{k+1}, v^{k+1}) \right).\tag{47}$$

Then we can deduce that

$$\sum_{k=1}^l \|w^{k+1} - w^k\|^2 \leq \frac{1}{\tau_1} \left( L(w^0, u^0, v^0) - L(w^{k+1}, u^{k+1}, v^{k+1}) \right) < \infty.\tag{48}$$

Even $l \to \infty$, we still have $\sum_{k=1}^\infty \|w^{k+1} - w^k\|^2 < \infty$. Thus $\{w^k\}$ is bounded.

From (40), we know

$$\|u^{k+1} - u^k\|^2 \leq \frac{\rho}{2} \|u^{k+1} - u^k\|^2.\tag{49}$$

That means, $\{u^k\}$ is also bounded. To sum up, the sequence $\{w^k, u^k, v^k\}$ is bounded. The proof is completed.

Proposition 3: The proposed ADMM-MCP algorithm satisfies the subgradient bound condition given by C3.

Proof: The proof involves the derivations of three gradients. They are the gradient $\partial L / \partial w$ of $L$ with respect to $w$, the limiting subgradient $\partial_u L$ of $L$ with respect to $u$, and the subgradient $\partial_v L$ of $L$ with respect to $v$.

For $\partial L / \partial w$, we have

$$\frac{\partial L}{\partial w}(w^{k+1}, u^{k+1}, v^{k+1}) = \nabla w \psi(w^{k+1}) + \rho \left( w^{k+1} - u^{k+1} \right) - v^{k+1}.\tag{52}$$

Since $w^{k+1}$ is the optimal solution of $L(w, u^{k+1}, v^k)$, we have

$$\nabla w \psi(w^{k+1}) + \rho \left( w^{k+1} - u^{k+1} \right) - v^k = 0.\tag{53}$$

From (52) and (53), we have

$$\frac{\partial L}{\partial w}(w^{k+1}, u^{k+1}, v^{k+1}) = v^k - v^{k+1}.\tag{54}$$

For the limiting subgradient $\partial_u L$, we have

$$\frac{\partial_u L}{\partial u}(w^{k+1}, u^{k+1}, v^{k+1}) = \nabla u \psi(u^{k+1}) + v^{k+1} - \rho \left( w^{k+1} - u^{k+1} \right).\tag{55}$$

Since $u^{k+1}$ is the optimal solution of $L(w^k, u, v^k)$, we have

$$0 \in \partial_u L (w^k, u^{k+1}) + v^{k+1} - \rho \left( w^k - u^{k+1} \right).\tag{56}$$

From (55) and (56), we have

$$\rho \left( w^k - w^{k+1} \right) + v^{k+1} - v^k \in \partial_u L(w^{k+1}, u^{k+1}, v^{k+1}).\tag{57}$$
TABLE 1. Properties of the eight data sets.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Number of features</th>
<th>Size of training set</th>
<th>Size of test set</th>
<th>RBF width</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abalone</td>
<td>7</td>
<td>2000</td>
<td>2177</td>
<td>0.1</td>
</tr>
<tr>
<td>ASN</td>
<td>5</td>
<td>751</td>
<td>752</td>
<td>0.5</td>
</tr>
<tr>
<td>Housing</td>
<td>13</td>
<td>400</td>
<td>106</td>
<td>2</td>
</tr>
<tr>
<td>Concrete</td>
<td>9</td>
<td>500</td>
<td>530</td>
<td>0.5</td>
</tr>
<tr>
<td>Energy</td>
<td>7</td>
<td>600</td>
<td>168</td>
<td>0.5</td>
</tr>
<tr>
<td>WQW</td>
<td>12</td>
<td>2000</td>
<td>2898</td>
<td>1</td>
</tr>
<tr>
<td>MG</td>
<td>6</td>
<td>700</td>
<td>685</td>
<td>1</td>
</tr>
<tr>
<td>Space-ga</td>
<td>6</td>
<td>1600</td>
<td>1507</td>
<td>0.1</td>
</tr>
</tbody>
</table>

For \( \partial_vL \), from (25) and (39), we have
\[
\partial_vL(w^{k+1}, u^{k+1}, v^{k+1}) = u^{k+1} - w^{k+1} = \frac{1}{\rho} (v^{k+1} - v^k). \quad (58)
\]

Define
\[
d^{k+1} := \left[ \rho \left( w^k - w^{k+1} \right), \frac{1}{\rho} \left( v^{k+1} - v^k \right), \right]. \quad (59)
\]

Hence, from (54), (57), and (58),
\[
d^{k+1} \in \partial L \left( w^{k+1}, u^{k+1}, v^{k+1} \right). \quad (60)
\]

From the inequality stated in (40), we can deduce that
\[
\|d^{k+1}\|_2^2 \leq \tau_2 \|w^{k+1} - w^k\|_2^2, \quad (61)
\]

where \( \tau_2 > 0 \). The proof is completed.

Since the ADMM-MCP algorithm satisfies the three conditions of Theorem 1, the sequence \( \{w^k, u^k, v^k\} \) has at least one limit point \( \{w^*, u^*, v^*\} \) and any limit point \( \{w^*, u^*, v^*\} \) is a stationary point. In other words, at least, the ADMM-MCP algorithm has the local convergent property.

V. SIMULATION RESULT
A. SETTINGS
This paper considers eight datasets. Six of them are from the University of California Irvine (UCI) machine learning repository [38]. They are respectively Abalone [39], [40], Airfoil Self-Noise (ASN) [41], Boston Housing (Housing) [39], [41], Concrete [42], Energy Efficiency (Energy) [41], and Wine Quality White (WQW) [43], [44]. The other two datasets are the Space-ga [45] and Mackey-Glass (MG) system [46] datasets. For each dataset, its RBF width is selected between 0.1 to 10. Table 1 summarizes the properties of these datasets.

The performances of the resultant networks are evaluated by the average test set MSE, given by
\[
\mathcal{E}_{test} = \frac{1}{N'} \sum_{i'=1}^{N'} \|y' - A'w\|_2^2 + \frac{1 - P_{\beta}}{N'} \|y' - A'w\|_2^2 + \frac{1}{N'} \left( P_{\beta} + \sigma_0^2 \right) \|A'TA'\|_2^2, \quad (62)
\]

where \( \{x_{i'}, y_{i'}\}, i' = 1, \ldots, N' \) is the test set, \( N' \) is the number of samples in the test set, \( y' = [y'_{1}, \ldots, y'_{N'}] \), \( A' \) is an \( N' \times M \) matrix, and its element in \( i' \)th row and \( j \)th column is given by
\[
[A']_{i', j} = \exp \left( -\frac{\|x_{j} - c_{j}\|_2^2}{s} \right). \quad (63)
\]

The two parameters, \( P_{\beta} \) and \( \sigma_0^2 \), describe the failure levels for open weight fault and multiplicative weight noise, respectively. We consider three fault scenarios: \( P_{\beta} = \sigma_0^2 = 0.005 \), \( P_{\beta} = \sigma_0^2 = 0.01 \), and \( P_{\beta} = \sigma_0^2 = 0.05 \). For the ADMM-MCP algorithm, we set \( \gamma = 1.001 \) and \( \rho = 0.1 \). The parameter \( \lambda \) is used to control the number of nodes.

B. MSE VERSUS THE NUMBER OF HIDDEN NODES
We use the ASN data set to illustrate the way to control the number of nodes. We can vary the value of \( \lambda \) to control the number of RBF nodes. Figure 2(a)-(b) show test set MSE versus \( \lambda \) and number of nodes versus \( \lambda \). Combining Figure 2(a) and (b), we obtain MSE versus the number of RBF nodes.

![Figure 2](image-url)
faulty networks. This is because the additional term, $\mathbf{w}^T \mathbf{Rw}$ where $\mathbf{R} = (P_\beta + \sigma_b^2) \text{diag} \left( \frac{1}{N} \mathbf{A}^T \mathbf{A} \right) - \frac{P_\beta}{N} \mathbf{A}^T \mathbf{A}$, not only makes the trained network to tolerate the multiplicative noise and node failure, but also has the ability to avoid overfitting. For other datasets and settings, their MSE curves have the similar behaviour.

C. CONVERGENCE

Here we use the ASN dataset with $\{P_\beta = \sigma_b^2 = 0.01\}$ as an example to intuitively demonstrate the convergence. The result is shown in Figure 3 which shows the objective value $\psi(\mathbf{w}) = \frac{1}{N} \| \mathbf{y} - \mathbf{Aw} \|^2 + \mathbf{w}^T \mathbf{Rw}$ versus the number of iterations. We can see that within 100 to 200 iterations, the training objective value nearly settles down. If we increase the value of $\lambda$, then the algorithm converges to a larger objective value. It is because increasing $\lambda$ leads to a restriction on the approximation ability of the resultant networks and the resultant networks have larger objective value $\psi(\mathbf{w})$. For all other datasets and other settings, they have similar properties of convergence.

D. COMPARISON ALGORITHMS

We compare our proposed algorithm with six other algorithms. They are, respectively, the fault tolerant OLS algorithm (OLS) [19], the fault tolerant $l_1$-norm approach (ADMM-$l_1$) [26], the $l_1$-norm regularization approach ($l_1$-reg.) [39], the support vector regression algorithm SVR [39], the orthogonal forward regression algorithm (OFR) [47] and the Homotopy method (HOM) [48]. Our aim is to show that our proposed algorithm has better RBF center selection capability. We will show that when we do not use all the training input vectors as the RBF centers, the performances of our proposed algorithm are better than those of the six comparison algorithms.

The fault tolerant OLS algorithm and the fault tolerant ADMM-$l_1$-norm algorithm have fault tolerant ability. The fault tolerant OLS algorithm includes two stages. In the first one, it uses the OLS method to generate a sorted list of RBF nodes. In the second stage, it constructs a fault tolerant RBF network with desired number of nodes. The fault tolerant ADMM-$l_1$ approach is our previous work based on an $l_1$-norm regularizer.

The $l_1$-norm regularization approach [39] considers the original MSE training objective and uses an $l_1$-norm regularizer to control the number of RBF nodes. Its fault tolerant ability is inadequate. Especially, when the fault level is high. The SVR algorithm [39] is able to train the RBF network and to select the centers simultaneously. It uses two parameters $C$ and $\epsilon$ to control the training process. Table 2 shows the parameter settings for different datasets. The SVR algorithm has certain fault tolerant ability. It is because the parameter $C$
TABLE 3. Average test MSE over 20 trials under the fault-free situation.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>ADMM ω 201 and ω 203</th>
<th>ADMM ω 1</th>
<th>OLS</th>
<th>l1-reg.</th>
<th>SVR</th>
<th>HOM</th>
<th>OFR</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>AVG MSE of nodes</td>
<td>AVG MSE of nodes</td>
<td>AVG MSE of nodes</td>
<td>AVG MSE of nodes</td>
<td>AVG MSE of nodes</td>
<td>AVG MSE of nodes</td>
<td>AVG MSE of nodes</td>
</tr>
<tr>
<td>Abalone</td>
<td>4.5789</td>
<td>730.2</td>
<td>4.5697</td>
<td>726.4</td>
<td>4.6405</td>
<td>40.8</td>
<td>4.5697</td>
</tr>
<tr>
<td>ASN</td>
<td>0.01100</td>
<td>409.4</td>
<td>0.01096</td>
<td>401.0</td>
<td>0.00756</td>
<td>401.0</td>
<td>0.01096</td>
</tr>
<tr>
<td>Housing</td>
<td>0.00748</td>
<td>135.3</td>
<td>0.00746</td>
<td>134.2</td>
<td>0.06688</td>
<td>129.7</td>
<td>0.00746</td>
</tr>
<tr>
<td>Concrete</td>
<td>0.00842</td>
<td>327.9</td>
<td>0.00840</td>
<td>351.5</td>
<td>0.00606</td>
<td>203.4</td>
<td>0.00840</td>
</tr>
<tr>
<td>Energy</td>
<td>0.00453</td>
<td>328.5</td>
<td>0.00459</td>
<td>324.5</td>
<td>0.00356</td>
<td>324.5</td>
<td>0.00459</td>
</tr>
<tr>
<td>WQW</td>
<td>0.01473</td>
<td>1490.0</td>
<td>0.01476</td>
<td>1460.0</td>
<td>0.01424</td>
<td>338.0</td>
<td>0.01480</td>
</tr>
<tr>
<td>MG</td>
<td>0.01762</td>
<td>54.9</td>
<td>0.01933</td>
<td>57.2</td>
<td>0.01576</td>
<td>78.2</td>
<td>0.01933</td>
</tr>
<tr>
<td>Space-gz</td>
<td>0.02103</td>
<td>103.3</td>
<td>0.01947</td>
<td>249.6</td>
<td>0.01933</td>
<td>368.1</td>
<td>0.01947</td>
</tr>
</tbody>
</table>

FIGURE 5. Performances of various algorithms under the faulty situation.

is capable to limit the magnitudes of the trained weights. The parameter $\epsilon$ is used to control its approximation ability. However, the main drawback of SVR algorithm is that there is no simple way to find an appropriate pair of $C$ and $\epsilon$. In our experiment, we use trial-and-error method to determine them.

The Homotopy method [48] is an incremental learning method. It has an $l_1$-norm regularization term, and it can tune its regularization parameter automatically. The OFR algorithm [47] is an incremental learning method too. It chooses one RBF center at a time with the orthogonal forward regression procedure. For OFR, an $l_2$-norm regularization term is used. It can also tune the regularization parameter automatically during training process.

In the following two experiments, the simulation was ran 20 times. In each trial, the samples of dataset were randomly split for training and testing set.

E. COMPARISON: FAULT FREE CASE

This subsection investigates the performances of various algorithms under the fault-free situation. It gives us a baseline of the performances under the fault-free situation. Some typical examples are given by Figure 4. In the fault-free case, the performances of the fault tolerant $l_1$-norm approach and the $l_1$-norm regularization approach are substantially same with each other. For OLS, HOM, OFR and SVR, we select their minimum MSE and the corresponding number of nodes to represent their performances. For other algorithms, we use the points where the number of nodes is close to the best result of SVR to represent their performances.

Table 3 shows the average test set error over the 20 trials. From the table and Figure 4, it can be observed that, under the fault-free environment, the performances of OLS and HOM are better than other algorithms.

F. COMPARISON: FAULTY CASE

This subsection compares the proposed algorithm with the aforementioned algorithms under the concurrent fault situation. We show that when we do not use all the training input vectors as the RBF centers, the performances of our proposed algorithm are better than those of the six comparison algorithms. Three fault levels are considered. They are $\{P_\beta = \sigma_b^2 = 0.005\}$, $\{P_\beta = \sigma_b^2 = 0.01\}$ and $\{P_\beta = \sigma_b^2 = 0.05\}$.

The typical result of one of the 20 trials of the ASN dataset under different fault levels is given by Figure 5. In the figure, the first, second and third columns show the test MSE results of faulty networks for $\{P_\beta = \sigma_b^2 = 0.005\}$, $\{P_\beta = \sigma_b^2 = 0.01\}$ and $\{P_\beta = \sigma_b^2 = 0.05\}$, respectively. Here we use the first column in Figure 5 to discuss the result.

- The performances of the HOM and OFR algorithms are very poor. Their minimum test set MSE values are equal to 0.13265 and 521.8148, respectively. They are much...
TABLE 4. Average test MSE over 20 trials under the concurrent fault situation.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Fault level</th>
<th>ADMM-MCP</th>
<th>OLS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>AVG MSE</td>
<td>AVG no. of nodes</td>
<td>AVG no. of nodes</td>
</tr>
<tr>
<td>Abalone</td>
<td>$P_0 = \sigma^2_0 = 0.005$</td>
<td>5.1417</td>
<td>5.3611</td>
</tr>
<tr>
<td></td>
<td>$P_0 = \sigma^2_0 = 0.01$</td>
<td>5.2567</td>
<td>5.5526</td>
</tr>
<tr>
<td></td>
<td>$P_0 = \sigma^2_0 = 0.05$</td>
<td>5.4169</td>
<td>5.6517</td>
</tr>
<tr>
<td>ASN</td>
<td>$P_0 = \sigma^2_0 = 0.005$</td>
<td>0.01512</td>
<td>0.01617</td>
</tr>
<tr>
<td></td>
<td>$P_0 = \sigma^2_0 = 0.01$</td>
<td>0.01598</td>
<td>0.01711</td>
</tr>
<tr>
<td></td>
<td>$P_0 = \sigma^2_0 = 0.05$</td>
<td>0.02026</td>
<td>0.02086</td>
</tr>
<tr>
<td>Housing</td>
<td>$P_0 = \sigma^2_0 = 0.005$</td>
<td>0.01527</td>
<td>0.01583</td>
</tr>
<tr>
<td></td>
<td>$P_0 = \sigma^2_0 = 0.01$</td>
<td>0.01456</td>
<td>0.01763</td>
</tr>
<tr>
<td></td>
<td>$P_0 = \sigma^2_0 = 0.05$</td>
<td>0.02060</td>
<td>0.02086</td>
</tr>
<tr>
<td>Concrete</td>
<td>$P_0 = \sigma^2_0 = 0.005$</td>
<td>0.01218</td>
<td>0.01334</td>
</tr>
<tr>
<td></td>
<td>$P_0 = \sigma^2_0 = 0.01$</td>
<td>0.01387</td>
<td>0.01489</td>
</tr>
<tr>
<td></td>
<td>$P_0 = \sigma^2_0 = 0.05$</td>
<td>0.02046</td>
<td>0.02086</td>
</tr>
<tr>
<td>Energy</td>
<td>$P_0 = \sigma^2_0 = 0.005$</td>
<td>0.00518</td>
<td>0.00500</td>
</tr>
<tr>
<td></td>
<td>$P_0 = \sigma^2_0 = 0.01$</td>
<td>0.00620</td>
<td>0.00610</td>
</tr>
<tr>
<td></td>
<td>$P_0 = \sigma^2_0 = 0.05$</td>
<td>0.01451</td>
<td>0.01486</td>
</tr>
<tr>
<td>WQW</td>
<td>$P_0 = \sigma^2_0 = 0.005$</td>
<td>0.01460</td>
<td>0.01479</td>
</tr>
<tr>
<td></td>
<td>$P_0 = \sigma^2_0 = 0.01$</td>
<td>0.01580</td>
<td>0.01583</td>
</tr>
<tr>
<td></td>
<td>$P_0 = \sigma^2_0 = 0.05$</td>
<td>0.02579</td>
<td>0.02583</td>
</tr>
<tr>
<td>MG</td>
<td>$P_0 = \sigma^2_0 = 0.005$</td>
<td>0.01803</td>
<td>0.01928</td>
</tr>
<tr>
<td></td>
<td>$P_0 = \sigma^2_0 = 0.01$</td>
<td>0.01974</td>
<td>0.02166</td>
</tr>
<tr>
<td></td>
<td>$P_0 = \sigma^2_0 = 0.05$</td>
<td>0.02558</td>
<td>0.02338</td>
</tr>
<tr>
<td>Space-ga</td>
<td>$P_0 = \sigma^2_0 = 0.005$</td>
<td>0.01796</td>
<td>0.01818</td>
</tr>
<tr>
<td></td>
<td>$P_0 = \sigma^2_0 = 0.01$</td>
<td>0.01994</td>
<td>0.02038</td>
</tr>
<tr>
<td></td>
<td>$P_0 = \sigma^2_0 = 0.05$</td>
<td>0.02781</td>
<td>0.02721</td>
</tr>
</tbody>
</table>

Higher than the test set MSE values obtained from the other algorithms. In the figure, the red colored marker shows the minimum test set MSE of the SVR algorithm. The minimum value is equal to 0.0015838 and the number of used nodes is 694. When the ADMM-MCP, ADMM-\ell_1, and OLS algorithms use around 694 nodes, their MSE values are lower than that of the SVR algorithms. From Figure 5, the MSE curve of the proposed ADMM-MCP is lower than those of the other algorithms. That means, in terms of the number of nodes and the test set MSE, the performances of the ADMM-MCP are better than those of the other algorithms.

• To further discuss the result, we can use the SVM and ADMM-\ell_1 algorithms as anchors. As mentioned in the above, for SVR, the minimum value is equal to 0.0015838 and the number of used nodes is 694. The ADMM-\ell_1 algorithm is able to use 157 nodes to achieve a similar test set MSE value, i.e., 0.015672. For the OLS algorithm, when around 150 nodes are used, the MSE value is 0.016552 which is higher than that of the ADMM-\ell_1. For the \ell_1 algorithm, the minimum test

VOLUME 7, 2019

151911
set MSE is 0.016419 and the number of the used nodes is 169. For the proposed ADMM-MCP algorithm, it is able to use 139 nodes only to lower the test set MSE to 0.015281. Clearly, the performances of the proposed ADMM-MCP are better than those of the comparison algorithms.

For each fault level and each dataset, we repeated the experiment 20 trials. In each trail, the samples of dataset were randomly split for training and testing set. The results are summarized in Table 4. From the table, it can be seen that under the concurrent fault situation, even we select the best MSE values of SVR, $l_1$-reg., HOM, and OFR, their performance are still unacceptable. Especially, when the fault level is high, their test MSE values are much higher than those of the remaining algorithms. The ADMM-MCP, ADMM-$l_1$ and OLS algorithms can effectively reduce the influence of the concurrent fault. Among them, the ADMM-MCP is the best which has smaller average MSE values and uses fewer number of nodes.

### G. PAIRED T-TEST

This subsection uses the paired t-test to illustrate that the improvement of our proposed algorithm is statistically significant. From Table 4, The ADMM-$l_1$ and OLS algorithm are the second best and the third best. Hence we perform the paired tests between ADMM-MCP and ADMM-$l_1$, and between ADMM-MCP and OLS. The paired test results are summarized in Tables 5 and 6. For the one-tailed test with 95% level of confidence and 20 trials, the critical t-value is 1.729.

From the tables, we can see that all the test t-values are greater than 1.729 and all p-values are smaller than 0.05. In other words, we have enough confidence to say that on average the proposed ADMM-MCP is better than the ADMM-$l_1$ and OLS algorithms. Besides, all confidence intervals in the two tables do not include zero. Therefore, we can further confirm that the improvement of the proposed ADMM-MCP is statistically significant.

### VI. CONCLUSION

In the paper, the fault tolerant RBF neural network and its center selection problem are studied. Based on ADMM framework and $\ell_0$-norm, this paper proposes the ADMM-MCP algorithm. First we introduce an $\ell_0$-norm term, which has an ability to remove some unimportant RBF nodes during training, into the fault tolerant objective function. Since $\ell_0$-norm is noncontinuous, we cannot use traditional gradient descent like algorithms to minimize the modified objective function. We then approximate the $\ell_0$-norm term with the MCP function. However, the MCP-based objective function is still nonconvex and nonsmooth, traditional gradient descent like algorithms cannot handle it. This paper then applies the ADMM framework to construct an algorithm, namely ADMM-MCP, to train an RBF network and to select RBF nodes simultaneously. The ADMM framework breaks down the update into three parts. Each part can be effectively solved, even though some parts contain nonconvex and non-differentiable terms. In addition, we prove that the algorithm converges. From the experimental results, our ADMM-MCP algorithm is superior to many other existing algorithms.
REFERENCES


ZHANGLEI SHI is currently pursuing the Ph.D. degree with the Department of Electrical Engineering, City University of Hong Kong. His current research interests include neural networks and machine learning.

HIU TUNG WONG received the B.S. degree (Hons.) in applied physics from the City University of Hong Kong, Hong Kong, in 2014, where he is currently pursuing the Ph.D. degree with the Department of Electrical Engineering. His research interests include neural networks and machine learning.

CHI-SING LEUNG received the Ph.D. degree in computer science from The Chinese University of Hong Kong, Hong Kong, in 1995. He is currently a Professor with the Department of Electrical Engineering, City University of Hong Kong. He has authored over 120 journal articles in the areas of digital signal processing, neural networks, and computer graphics. His current research interests include neural computing and computer graphics.

Dr. Leung was a member of the Organizing Committee of ICONIP 2006. He received the 2005 IEEE TRANSACTIONS ON MULTIMEDIA Prize Paper Award for his article titled The Plenoptic Illumination Function, in 2005. He was the Program Chair of ICONIP 2009 and ICONIP 2012. He is a Governing Board Member and the Vice President of the Asian Pacific Neural Network Assembly (APNNA). He is/was the Guest Editor of several journals, including Neural Computing and Applications, Neurocomputing, and Neural Processing Letters.

HING CHEUNG SO was born in Hong Kong. He received the B.Eng. degree in electronic engineering from the City University of Hong Kong, in 1990, and the Ph.D. degree in electronic engineering from The Chinese University of Hong Kong, in 1995, where he was a Postdoctoral Fellow, from 1995 to 1996. From 1990 to 1991, he was an Electronic Engineer with the Research and Development Division, Everex Systems Engineering Ltd., Hong Kong. From 1996 to 1999, he was a Research Assistant Professor with the Department of Electronic Engineering, City University of Hong Kong, where he is currently an Associate Professor. His research interests include detection and estimation, fast and adaptive algorithms, multidimensional harmonic retrieval, robust signal processing, sparse approximation, and source localization.

Dr. So has been elected as a Fellow of the IEEE in recognition of his contributions to spectral analysis and source localization. In addition, he was an Elected Member of the Signal Processing Theory and Methods Technical Committee, IEEE Signal Processing Society, in 2011, where he is Chair of the awards subcommittee, in 2015. He was on the Editorial Board of the IEEE TRANSACTIONS ON SIGNAL PROCESSING, from 2010 to 2014, Signal Processing, in 2010, Digital Signal Processing, in 2011, and the IEEE Signal Processing Magazine, in 2014.

RUIBIN FENG received the Ph.D. degree in electronic engineering from the City University of Hong Kong, in 2017. His current research interests include neural networks and medical imaging.