Neurocomputing 102 (2013) 31–44

Contents lists available at SciVerse ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Robust extreme learning machine

Punyaphol Horata *, Sirapat Chiewchanwattana, Khamron Sunat

Computer Science Department, Faculty of Science, Khon Kaen University, Friendship Road, Khon Kaen 40002, Thailand

ARTICLE INFO

Available online 5 June 2012

Keywords:
Extreme learning machine
Moore–Penrose pseudo inverse
Singular Value Decomposition
Extended Complete Orthogonal Decomposition
Iteratively reweighted least squares
Multivariate least-trimmed squares
Minimax probability machine regression
Meta-metrics evaluation
Outlier
Robustness

ABSTRACT

The output weights computing of extreme learning machine (ELM) encounters two problems, the computational and outlier robustness problems. The computational problem occurs when the hidden layer output matrix is a not full column rank matrix or an ill-conditioned matrix because of randomly generated input weights and biases. An existing solution to this problem is Singular Value Decomposition (SVD) method. However, the training speed is still affected by the large complexity of SVD when computing the Moore–Penrose (MP) pseudo inverse. The outlier robustness problem may occur when the training data set contaminated with outliers then the accuracy rate of ELM is extremely affected. This paper proposes the Extended Complete Orthogonal Decomposition (ECOD) method to solve the computational problem in ELM weights computing via ECODLS algorithm. And the paper also proposes the other three algorithms, i.e. the iteratively reweighted least squares (IRWLS-ELM), ELM based on the multivariate least-trimmed squares (MLTS-ELM), and ELM based on the one-step reweighted MLTS (RMLTS-ELM) to solve the outlier robustness problem. However, they also encounter the computational problem. Therefore, the ECOD via ECODLS algorithm is also used successfully in the three proposed algorithms. The experiments of regression problems were conducted on both toy and real-world data sets. The outlier types are one-sided and two-sided outliers. Each experiment was randomly contaminated with outliers, of one type only, with 10%, 20%, 30%, 40%, and 50% of the total training data size. Meta-metrics evaluation was used to measure the outlier robustness of the proposed algorithms compared to the existing algorithms, i.e. the minimax probability machine regression (MPMR) and the ordinary ELM. The experimental results showed that ECOD can effectively replace SVD. The ECOD is robust to the not full column rank or the ill-conditional problem. The speed of the ELM training using ECOD is also faster than the ordinary training algorithm. Moreover, the meta-metrics measure showed that the proposed algorithms are less affected by the increasing number of outliers than the existing algorithms.

1. Introduction

Extreme learning machine (ELM) is an interesting learning algorithm proposed by Huang et al. [1]. It works on a simple structure named *s*ingle-hidden *l*ayer *f*eedforward *n*eural networks (SLFNs). It randomly applies computational hidden nodes. This mechanism is different from the conventional learning of SLFNs. It provides a good generalization and a highly accurate learning solution for both classification and regression problems [2,3]. Huang et al. mentioned that ELM yielded better performance than other conventional learning algorithms in application with higher noise [2]. ELM also has an extremely fast learning speed compared to traditional gradient-based algorithms. Furthermore, ELM technique successfully overcame the difficulty of the curse of dimensionality [4]. For regression problems, ELM has been widely applied in many real-world regression problems such as terrain models [5], prediction of melting points of organic compounds [6], sales forecasting of fashion retailing [7], modeling permeability prediction [8], etc. However, the computational robustness and the outlier robustness problems of ELM are not widely mentioned. The computational robustness is a computing ability of ELM to compute output weights from a formula like $\mathbf{H}\beta = \mathbf{T}$ even if the hidden layer output matrix $\mathbf{H}$ is not full column rank or ill-conditioned. The ordinary ELM solved this problem by using the Singular Value Decomposition (SVD) approach [9]. However, the shortcoming of SVD is that it is very slow whenever the computed data set is large. The outlier robustness problem may occur whenever the training data is contaminated with outliers, which may result in ELM producing a poor and unreliable solution. An outlier is a sample with an error or noise due to human or device error that is outstanding and far away from other regular samples [10].

Wang et al. [11] proposed the effective extreme learning machine (EELM) to improve the computational robustness of ELM. However, EELM is limited with an activation function with one peak such as Gaussian radial basis function. Yuan et al. [12] proposed an optimization approximation solution algorithm for ELM. The $\mathbf{H}\beta = \mathbf{T}$ is

* Corresponding author. Tel.: +66 816704734; fax: +66 043 342910.
E-mail addresses: punhor1@kku.ac.th (P. Horata), sunkra@kku.ac.th (S. Chiewchanwattana), khamron_sunat@yahoo.com (K. Sunat).

0925-2312/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.neucom.2011.12.045

Nomenclature

$\mathbf{I}_{\Sigma}$	the identity matrix having size of Σ
$\mathbf{\Delta}_k$	a row vector of errors of the kth sample
MAD	the median absolute deviation
Med	the median value
$M(1:h)$	the first element to the hth element of M
$\mathbf{H}(M,:)$	specifying the kth row of $\mathbf{H}$ for all k is in set M
$\mathbf{H}(:,M)$	specifying the kth column of $\mathbf{H}$ for all k is in set M
cond($\mathbf{A}$)	the condition number of matrix $\mathbf{A}$
rank($\mathbf{A}$)	the rank of matrix $\mathbf{A}$
$a{:}b$	the row vector generating $a,a+1,a+2,\ldots,b$
$\sharp M$	number of elements of a subset M
$\mathbf{H}_M$	a submatrix of $\mathbf{H}$, its rows are pointed by a row indices set M
Σ	the covariance matrix
$\det(\Sigma)$	the determinant of the covariance matrix Σ
c_{α}	the consistency factor
F_{χ^2}	the cumulative distribution function of the chi-square χ^2
χ^2_q	the Chi-square with q degrees of freedom

reformed to $\mathbf{H}^T\mathbf{H}\beta=\mathbf{H}^T\mathbf{T}$ to obtain a solution based on the rank of $\mathbf{H}$. Its advantage is that it can be used with $\mathbf{H}$ that is either full rank or not full rank. Huynh et al. [13] proposed ELM based on the weighted least squares (WLSs) scheme for reducing the effects of outliers in regression problems by using the penalty weights based on the norm of residuals. Their result showed a promising improvement in the outlier robustness of ELM. Deng et al. [14] proposed a regularized ELM to increase the generalization of ELM. They showed that ELM was seriously affected by outliers. However, the study into the effects of outliers in ELM is only in its infancy.

Thus, reducing the effects of not full column rank or ill-condition of the hidden layer output matrix $\mathbf{H}$ and the influence of outliers are extremely important and necessary. Therefore, this paper will address them simultaneously. The computational robustness problem is solved by replacing SVD with the Extended Complete Orthogonal Decomposition (ECOD). Three new iterative algorithms for ELM are proposed to improve the outlier robustness of ELM. They require robust inverse matrix computing. Accordingly, ECOD via ECODLS is also used for overcoming the not full column rank problem and the ill-conditioned matrix problem as well as the speeding up the training time of the common sub-problem, $\mathbf{Ax}=\mathbf{b}$, in the three proposed algorithms.

The remainder of this paper is organized into five sections. The second section describes the background knowledge of ELM, the iteratively reweighted least squares (IRWLSs), the multivariate least-trimmed squares (MLTSs), and the one-step reweighted MLTS (RMLTS). The third section discusses the new proposed algorithms. The fourth section covers the experiments and results, and the fifth section is the conclusion and discussion. Finally, the Appendix describes ECOD and the implementation of ECODLS.

2. ELM and existing algorithms

This section will briefly describe outlier definition, ELM, IRWLS, MLTS, and RMLTS. IRWLS, MLTS, and RMLTS are the three algorithms that will be improved and used as the basis for the proposed algorithms to improve the robust property of ELM.

2.1. The extreme learning machine and the computational robustness and outlier robustness problems

Let $\mathbf{Z}=\{(\mathbf{x}_i,\mathbf{T}_i)|\mathbf{x}_i\in\mathbb{R}^{1\times m},\mathbf{T}_i\in\mathbb{R}^{1\times q},\ i=1,\ldots,n\}$ be a sample set size n where $\mathbf{x}_i$ is the ith row input data matrix $\mathbf{X}\in\mathbb{R}^{n\times m}$, and $\mathbf{T}_i$ is the ith row of the target matrix $\mathbf{T}\in\mathbb{R}^{n\times q}$. Let p be a number of hidden nodes. ELM has only three steps. The first step, the input weights $\mathbf{W}\in\mathbb{R}^{m\times p}$ and hidden layer bias $b_i\in\mathbb{R}$, $i=1,\ldots,p$, are randomly generated. The second step is to compute the hidden layer output matrix of the network denoted by $\mathbf{H}\in\mathbb{R}^{n\times p}$, where $\mathbf{H}_{ij}$ is an output from the jth hidden node of the ith sample, $\mathbf{H}_{ij}=g(\mathbf{w}_j^T\mathbf{x}_i+b_j)$, where g is an activation function, $\mathbf{w}_j$ and b_j are the vector of the input weights and the hidden layer bias of the jth hidden node respectively. The third step is to compute the output weights β that is a solution of the following equation:

$$\mathbf{H}\beta=\mathbf{T}. \tag{1}$$

The output weights β can be computed by variants of ELM such as the batch ELM [1], I-ELM [15], EI-ELM [16]. For this paper, we use the batch ELM, therefore, the solution of Eq. (1) is computed by

$$\beta=(\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\mathbf{T}. \tag{2}$$

The term $(\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T$ is known as the pseudo inverse of $\mathbf{H}$ where it could be adopted when $\mathbf{H}^T\mathbf{H}$ is nonsingular. If the pseudo inverses properties satisfy four properties as described in [1,9] then the matrix is called the Moore–Penrose pseudo inverse matrix of $\mathbf{H}$ that it is denoted as $\mathbf{H}^{\dagger}=(\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T$. A computation robustness problem of ELM invokes about direct computing of $(\mathbf{H}^T\mathbf{H})^{-1}$ that may not obtain a solution whenever $\mathbf{H}$ is either not full column rank; $rank(\mathbf{H})<p$, or $\mathbf{H}$ is an ill-conditioned matrix. An example of not full column rank problem of $\mathbf{H}$ in ELM can be seen in [11]. However, Huang et al. [1] solved this problem using SVD approach to compute the Moore–Penrose pseudo inverse matrix instead of $(\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T$. That method can support either full or not full column rank or in case of ill-condition. That is to say, ELM has the property of computational robustness. However, SVD has the shortcoming as described in Section 1.

In this paper, the outlier robustness problem in ELM is simulated by randomly contaminating either one-sided or two-sided outliers into the targets of training data set. Let $K\subset\{1,\ldots,n\}$ be a subset of row indices of $\mathbf{T}$ to point to contaminated samples with some outliers and $\mathbf{\Delta}_k\in\mathbb{R}^{1\times q}$ be a row vector of errors, $\forall k\in K$, that it has normal distribution.

The outlier definitions are defined as follows.

Definition 2.1. Let $\mathbf{t}_k\in\mathbb{R}^{1\times q}$ be a row of $\mathbf{T}$, $\check{\mathbf{t}}_k$ be a contaminated row due to the one-sided outliers, and $\tilde{\mathbf{t}}_k$ be a contaminated row due to the two-sided outliers. The $\check{\mathbf{t}}_k$ and $\tilde{\mathbf{t}}_k$ are defined as

$$\check{\mathbf{t}}_k=\mathbf{t}_k+|\mathbf{\Delta}_k|, \tag{3}$$

$$\tilde{\mathbf{t}}_k=\mathbf{t}_k+\mathbf{\Delta}_k,\quad \forall k\in K, \tag{4}$$

where $|\mathbf{\Delta}_{kj}|>0$, for all $|\mathbf{\Delta}_{k,j}|>0$, $j=1,\ldots,q$.

Definition 2.2. Let $\mathbf{\Delta}\in\mathbb{R}^{n\times q}$ be a constructional matrix of $[\mathbf{\Delta}_1;\mathbf{\Delta}_2;\ldots;\mathbf{\Delta}_n]$ such that if $i\notin K$ then $\mathbf{\Delta}_i=\mathbf{0}\in\mathbb{R}^{1\times q}$.

Definition 2.3. A contaminated training target matrix $\check{\mathbf{T}}$ is defined as $\check{\mathbf{T}}=\mathbf{T}+\mathbf{\Delta}$.

These definitions suppose that there are outliers of one type only in targets of training data of a system. The SVD-based solution $\hat{\beta}$ of $\mathbf{H}\beta=\check{\mathbf{T}}$ is written as

$$\hat{\beta}=\mathbf{V}\mathbf{S}^{-1}\mathbf{U}^T\mathbf{T}+\mathbf{V}\mathbf{S}^{-1}\mathbf{U}^T\mathbf{\Delta}. \tag{5}$$

Focussing on Eq. (5), the problem possibly caused by Eq. (5) occurs when there exists a large magnitude $\mathbf{\Delta}_k$ for some k. It can make a contaminated row of $\tilde{\mathbf{T}}$ be an outlying sample and differ from other regular samples of $\tilde{\mathbf{T}}$, resulting the second part, $\mathbf{V}\mathbf{S}^{-1}\mathbf{U}^T\mathbf{\Delta}$, to be large [17]. Typically, the output weights $\hat{\beta}$ in ELM will be used for approximating the testing data set without outliers. An estimated solution of $\hat{\mathbf{T}}_{testing}=\mathbf{H}_{testing}\hat{\beta}$ could be seriously affected by outliers [10] because $\|\hat{\mathbf{T}}_{testing}-\mathbf{T}_{testing}\|$ is large. Therefore, this paper also focuses on enhancement of the outlier robustness in ELM. The algorithms will be governed by three robust statistical techniques, i.e. IRWLS, MLTS, and RMLTS.

2.2. The iteratively reweighted least squares (IRWLSs)

The influence of outliers of the least squares can be reduced by the M-estimators family [10,18,19]. Let $Z_H=\{(\mathbf{H}_i,\mathbf{T}_i)|\mathbf{H}_i\in\mathbb{R}^{1\times p},\ \mathbf{T}_i\in\mathbb{R}^{1\times 1},\ i=1,\ldots,n\}$ be a data set size n. $\mathbf{H}_i$ is the ith row of the design matrix $\mathbf{H}$. $\mathbf{T}_i$ is the ith scalar of the target vector $\mathbf{T}$. The outlier effect of the squared residual of the ith sample, $\hat{r}_i^2=(\mathbf{T}_i-\mathbf{H}_i\beta)^2$, $i=1,\ldots,n$, can be reduced by the iterative procedure named the iteratively reweighted least squares algorithm (IRWLS) [10,19–22]. It is an algorithm in the M-estimators family and can only handle the single output problem. The IRWLS algorithm can be summarized by the following algorithm [22]:

*Step*1: Initialize $\hat{\beta}$ of β.

*Step*2: Set up the scale parameter or leverage.

Let $\hat{\delta}_i$ be the scale parameter recommended by DuMouchel and Brien [23] that is defined as $\hat{\delta}_i=(1-h)^{1/2}$, where $i=1,\ldots,n$, $h=\min(0.9999,\sum_{j=1}^{p}\hat{\mathbf{E}}_{ij}^2)$, $\hat{\mathbf{E}}=\mathbf{H}\Pi(\mathbf{R}^{-1})$, Π denotes the permutation matrix, $\mathbf{R}$ is the triangle matrix that was produced from the decomposition of $\mathbf{H}$ using QR.

*Step*3: Define the weight via an M-estimate function, such as Huber's function defined as

$$\psi_i(\omega_i)=\begin{cases}1 & \text{if } |\omega_i|\le k,\\ \dfrac{k}{|\omega_i|} & \text{if } |\omega_i|>k,\end{cases}\quad i=1,\ldots,n \tag{6}$$

where $k=1$, $\omega_i=(\hat{\varepsilon}_i/\max(\mathrm{MADN},xi))\eta$ is the normally adjusted residual of the ith sample, $\hat{\varepsilon}_i=(\mathbf{T}_i-\mathbf{H}_i\hat{\beta})/\hat{\delta}_i$ is the adjusted residual, ξ is a tiny positive value, for this paper $\xi=10^{-16}\sigma_T$ with σ_T being a standard deviation of $\mathbf{T}$, η is 1.3745 for the Huber function, and MADN=MAD/0.6745 is the normalized median absolute deviation, $\mathrm{MAD}=Med(\hat{\mathbf{r}}_k,\ldots,\hat{\mathbf{r}}_n)$ is the median of the sorted absolute of adjusted residuals $\hat{\mathbf{r}}=sort(|\hat{\varepsilon}|)$ and $k=\max(1,rank(\hat{\mathbf{r}}))$, see detail in [10,18,19].

*Step*4: Calculate a new solution $\hat{\beta}$ at iteration t and then go to Step 3 until convergence.

For each iteration t, the new solution $\hat{\beta}$ is the solution of

$$\mathbf{D}^{(t-1)}\mathbf{H}\beta=\mathbf{D}^{(t-1)}\mathbf{T}, \tag{7}$$

which is computed by

$$\hat{\beta}^{(t)}=(\mathbf{H}^T\mathbf{D}^{(t-1)}\mathbf{H})^{-1}\mathbf{H}^T\mathbf{D}^{(t-1)}\mathbf{T}, \tag{8}$$

where $\mathbf{D}^{(t-1)}\in\mathbb{R}^{n\times n}$ is the diagonal matrix at the iteration $t-1$ whose diagonal elements denote $D_{ii}^{(t-1)}=\sqrt{\psi_i(\omega_i)^{(t-1)}}$. The diagonal matrix $\mathbf{D}=diag(\sqrt{\psi(\omega)})$ is used to weight the training samples to reduce influence of the outliers. Therefore, the solution $\hat{\beta}^{(t)}$ is computed by the weighted least squares using the following condition:

$$\min_{\beta}\|\mathbf{D}(\mathbf{H}\beta-\mathbf{T})\|. \tag{9}$$

The third step to the fourth step will be repeated until either t is greater than or equal to the maximum of iterations *maxIter* or the estimated solution $\hat{\beta}$ converges according to the following condition:

$$\forall\{|\hat{\beta}_{0i}-\hat{\beta}_i|\}\le\theta\times\max(|\hat{\beta}_0|,|\hat{\beta}|)\ i=1,\ldots,p, \tag{10}$$

where $\theta=\sqrt{10^{-16}}$, in this paper, is a tiny threshold value.

IRWLS can reduce the influence of outliers by reducing the importance of the outlying samples. The D_{ii} in IRWLS is assigned different from the penalty weights in [13]. For IRWLS using the Huber function, the D_{ii} of an outlying sample is presumed to be a small value while D_{ii} of a regular sample is presumed to be close to or equal to 1. But the penalty weights of [13] is the norm of the residuals. However, the shortcoming of IRWLS in Eq. (8) may give a low accuracy solution because $\mathbf{D}$ is naturally ill-conditioned [24] and the condition number of $\mathbf{D}^{(t-1)}\mathbf{H}$ is also high. In this paper, an effective solution of Eq. (7) will be directly obtained using ECOD via Algorithm 7 ECODLS.

2.3. The multivariate least-trimmed squares estimator (MLTS and RMLTS)

The multivariate least-trimmed squares estimator was proposed by Rousseeuw and Van Driessen [25]. MLTS can reduce the influence of outliers using the minimum covariance determinant (MCD) estimator. The main concept of the MLTS algorithm is to select a subset of observations whose size is h and has the property that yielded the minimal determinant covariance matrix of its residuals from the least squares fit. Let $\overline{Z}_H=\{(\mathbf{H}_i,\mathbf{T}_i)|\mathbf{H}_i\in\mathbb{R}^{1\times p},\ \mathbf{T}_i\in\mathbb{R}^{1\times q},\ i=1,\ldots,n\}$ be a data set size n. $\mathbf{H}_i$ is the ith row of the design matrix $\mathbf{H}$. $\mathbf{T}_i$ is the ith row of the target matrix $\mathbf{T}$. Let $\mathcal{M}=\{M\subset\{1,\ldots,n\}|\ |M|=h\}$ be the collection of all subsets of size h and $1\le h\le n$. $\hat{\beta}_M^{OLS}$ is a solution of the ordinary least squares

$$\mathbf{H}_M\hat{\beta}_M^{OLS}=\mathbf{T}_M \tag{11}$$

such that fits the samples of a subset M, where $\mathbf{H}_M$ and $\mathbf{T}_M$ are row submatrices of $\mathbf{H}$ and $\mathbf{T}$ respectively. Each row of the two submatrices is pointed by a row index set M. The solution of Eq. (11) can be written as

$$\hat{\beta}_M^{OLS}=(\mathbf{H}_M^{\mathrm{T}}\mathbf{H}_M)^{-1}\mathbf{H}_M^{\mathrm{T}}\mathbf{T}_M. \tag{12}$$

The $(\mathbf{H}_M^{\mathrm{T}}\mathbf{H}_M)^{-1}\mathbf{H}_M^{\mathrm{T}}$ is called pseudo inverse matrix of $\mathbf{H}_M$. The corresponding covariance matrix of $\mathbf{H}_M$ and $\mathbf{T}_M$ is defined as

$$\hat{\Sigma}_M^{OLS}=\frac{1}{n-q}r_M^{\mathrm{T}}r_M, \tag{13}$$

where $r_M=(\mathbf{T}_M-\mathbf{H}_M\hat{\beta}_M^{OLS})$ is a residual matrix.

Agulló et al. [26] proposed an alternative criterion that has equivalent characteristics to the MLTS estimator. The solution of Eq. (11) is written in mathematical terms as

$$\hat{\beta}^{MLTS}=\arg\min_{\beta,\Sigma,|\Sigma|=1}\sum_{j=1}^{h}d_{j:n}^2(\beta,\Sigma), \tag{14}$$

where $d_{j:n}^2(\beta,\Sigma)$ is the squared Mahalanobis distance of its residuals, $|\Sigma|=1$ means that the determinant of Σ is 1 [26,27]. Thus, M consists of row indices of the samples that selected via the ordered sequence of the residual Mahalanobis distances $d_{1:n}(\beta,\Sigma)\le\cdots\le d_{n:n}(\beta,\Sigma)$. That their squared Mahalanobis distances are one of the h smallest of the sequence. The residual Mahalanobis distances [27] can be written as

$$d_j(\beta,\Sigma)=((\mathbf{T}_i-\mathbf{H}_i\beta)^{\mathrm{T}}\Sigma^{-1}(\mathbf{T}_i-\mathbf{H}_i\beta))^{1/2}, \tag{15}$$

where $\beta\in\mathbb{R}^{p\times q}$ is a solution of the MLTS that can be converged using the C-step algorithm (see Ref. [26]) and Σ^{-1} is the inverse covariance matrix. Therefore, an approximated solution $\hat{\beta}^{MLTS}$ can

be written as

$$\hat{\beta}^{MLTS} = \hat{\beta}^{OLS}_{\hat{M}}, \tag{16}$$

where $\hat{M} = \underset{M \in \mathcal{M}}{\operatorname{argmin}} \det(\hat{\Sigma}^{OLS}_{M})$ with $\hat{\Sigma}^{OLS}_{M} = cov(M, \hat{\beta}^{OLS}_{M})$ for any $M \in \mathcal{M}$. The corresponding covariance matrix in MLTS is used to measure the scattering of the residuals which is written as

$$\hat{\Sigma}^{MLTS} = c_{\alpha}\hat{\Sigma}^{OLS}_{\hat{M}}, \tag{17}$$

where $c_{\alpha} = (1-\alpha)/F_{\chi^2_{q+2}}(q_{\alpha})$ with $q_{\alpha} = \chi^2_{q+2}$ is a correction factor; α is a trimming proportion for MLTS, which can be defined by the user, and $F_{\chi^2_{q+2}}(q_{\alpha})$ denotes the cumulative distribution function of a χ^2 with $q+2$ degrees of freedom [18,26,27].

Sometimes the accuracy of MLTS may be quite low. So Agulló et al. [26] proposed a reweighted version of MLTS named the reweighted multivariate least-trimmed squares (RMLTS). The solution of RMLTS $\hat{\beta}^{RMLTS}$ and its covariance are defined as

$$\hat{\beta}^{RMLTS} = \hat{\beta}^{OLS}_{J} \quad \text{and} \quad \hat{\Sigma}^{RMLTS} = c_{\delta}\hat{\Sigma}^{OLS}_{J}, \tag{18}$$

where $J = \{j \in \{1,\ldots,n\} \mid d^2_j(\hat{\beta}^{MLTS}, \hat{\Sigma}^{MLTS}) \le q_{\delta}\}$, $d^2_j(\hat{\beta}^{MLTS}, \hat{\Sigma}^{MLTS})$ is a large residual Mahalanobis distance with respect to the initial MLTS estimator, δ is a trimming proportion for RMLTS, $c_{\delta} = (1-\delta)/F_{\chi^2_{q+2}}(q_{\delta})$ with $q_{\delta} = \chi^2_{q,1-\delta}$ is a correction factor for RMLTS, and $\hat{\beta}^{OLS}_{J}$ is computed as in Eq. (12).

The idea for detecting outliers in MLTS is considered from the residual Mahalanobis distances. An outlying sample has a large residual Mahalanobis distance, and its row index is not a member in M, see detail in [26]. After MLTS is performed, the final RMLTS is used to detect the additive outlying samples that were not detected by the MLTS estimator. If $d^2_j(\hat{\beta}^{MLTS}, \hat{\Sigma}^{MLTS})$ of a sample is greater than q_{δ}, then the sample is marked as an outlier [27]. However, the shortcomings of MLTS and RMLTS algorithm are the problems of computing Eq. (12) and the inverse of covariance matrix. The solutions may have low accuracy or do not exist if the matrices are ill-conditioned or rank deficient. These problems can be avoided by SVD-based pseudo inverse computing. However, the time scale is also large when the computed matrices are large. This paper will use ECOD via ECODLS to directly solve Eq. (11) and $\Sigma\Sigma^{-1} = \mathbf{I}_{\Sigma}$.

3. Robust extreme learning machine

The proposed algorithms to improve the outlier robustness characteristic of ELM are described in this section. These are the iteratively reweighted least squares extreme learning machine (IRWLS-ELM), the multivariate least-trimmed squares extreme learning machine (MLTS-ELM) and the reweighted multivariate least-trimmed squares extreme learning machine (RMLTS-ELM). However, there are several steps in the algorithms that $\mathbf{Ax} = \mathbf{b}$, where $\mathbf{A} \in \mathbb{R}^{n \times p}$, $\mathbf{x} \in \mathbb{R}^{p \times q}$, $\mathbf{b} \in \mathbb{R}^{n \times q}$, $q \ge 1$, must be solved. The ECODLS will be placing at those points. It can find an effective solution either $\mathbf{A}$ is full or not full column rank. Using of ECOD via ECODLS in IRWLS, MLTS, and RMLTS can be summarized as follows:

In Algorithm 1 *IRWLS*:

(1) ECODLS is to approximate the initial solution $\hat{\beta}$ of $\mathbf{H}\beta = \mathbf{T}$ before going into the *while* loop.
(2) ECODLS is repeatedly used within the *while* loop of re-weighting to compute an iterative solution $\hat{\beta}$ of Eq. (7).

In Algorithm 2 *MLTS*:

(1) ECODLS is used within the *for* loop i to approximate a solution with respect to a subset M, $\hat{\beta}^{OLS}_{M}$ of Eq. (11).
(2) ECODLS is repeatedly used in Algorithm 4 *CSTEPs* to compute the inverse covariance matrix Σ^{-1} by solving $\Sigma\Sigma^{-1} = \mathbf{I}_{\Sigma}$ and to solve a least squares solution $\hat{\beta}^{OLS}_{M}$ of Eq. (11).

Algorithm 3 *RMLTS*:

(1) Algorithm 2 *MLTS* is called.
(2) ECODLS is used to compute the inverse covariance matrix $(\hat{\Sigma}^{MLTS}_{\hat{M}})^{-1}$ by solving $\hat{\Sigma}^{MLTS}_{\hat{M}}(\hat{\Sigma}^{MLTS}_{\hat{M}})^{-1} = \mathbf{I}_{\hat{\Sigma}^{MLTS}_{\hat{M}}}$.
(3) ECODLS is used to compute a refined solution with respect to a subject J, $\hat{\beta}^{RMLTS}$.

Remark 1. The writing style of an algorithm in this paper is based on the MATLAB style.

Remark 2. A comment in an algorithm is denoted by //.

The details of the improved algorithms are described in Algorithm 1 *IRWLS*, Algorithm 2 *MLTS*, and Algorithm 3 *RMLTS*.

Algorithm 1. IRWLS.

Input: H, T, *maxIter*
Output: $\hat{\beta}$
$\theta = \sqrt{10^{-16}}$; $[n,q] = size(T)$; $[\hat{\beta}, r, Z, P] = ECODLS(H,T)$
$\hat{\beta}(P,:) = [\hat{\beta}(1:r,:), \mathbf{0}]$; $P = P(1:r)$; $\hat{\beta}_0 = \mathbf{0} \in \mathbb{R}^{n \times q}$
$\mathbf{E} = H(:,P) \times Z(1:r, 1:r)$
$\hat{\delta}(i) = (1 - \min(0.9999, \sum_{k=1}^{p} \mathbf{E}(i,k)^2))^{1/2}$, $\forall i = 1,\ldots,n$
$\xi = 10^{-16} \times \sigma_T$; $\eta = 1.345$;
for $k = 1$ **to** *maxIter* **do**
  $up_{i,j} = \theta \times \max(|\hat{\beta}_{0\,i,j}, \hat{\beta}_{i,j}|)$, $\forall i = 1,\ldots,n$, $j = 1,\ldots,q$
  if $(|\hat{\beta}_{0\,i,j} - \hat{\beta}_{i,j}| \le up_{i,j} == true, \forall i = 1,\ldots,n, j = 1,\ldots,q)$ **then**
    break
  end if
  $\hat{\epsilon} = (T - H \times \hat{\beta})/\hat{\delta}$; $ra = sort(|\hat{\epsilon}|)$
  $\mathrm{MAD} = \mathrm{Med}(ra_k, \ldots, ra_n)$, $k = \max(1,r), \ldots, n$; $\mathrm{MADN} = \mathrm{MAD}/0.6745$
  $\psi_i = |\hat{\epsilon}(i)|/\max(\mathrm{MADN}, \xi) \times \eta$; $\psi_i = 1/\max(1, |\psi_i|)$, $\forall i = 1,\ldots,n$
  $D = diag(\sqrt{\psi})$; $\overline{T} = D \times T$; $\overline{H} = D \times H$; $\hat{\beta}_0 = \hat{\beta}$
  $[\hat{\beta}, r] = ECODLS(\overline{H}, \overline{T})$
end for

Algorithm 2. MLTS.

Input: H, T, α, *maxIter*, n_c
Output: $\hat{\beta}^{MLTS}$
Optional outputs: $\hat{\sigma}^{MLTS}$
$[n,q] = size(T)$; $[n,p] = size(H)$; $\det_0 = 10^{-10}$
if $\alpha \ge 0$ **then**
  $h = \lfloor (n \times (1-\alpha)) \rfloor + 1$
else
  $h = \lfloor (n \times (1-\alpha)) \rfloor$
end if
for $i = 1$ **to** *maxIter* **do**
  M = the indices of the sorted random number size n
  $M = M(1 : \min(n, p+q))$; $H_M = H(M,:)$; $T_M = T(M,:)$
  $\hat{\beta}^{OLS}_{M} = ECODLS(H_M, T_M)$; $\hat{\beta} = \hat{\beta}^{OLS}_{M}$

$\Sigma_M^{OLS} = \frac{1}{n-q}(T_M - H_M\hat{\beta})^T \times (T_M - H_M\hat{\beta})$
$[\hat{\beta}_{\hat{M}}^{OLS};\ \hat{\Sigma}_{\hat{M}}^{OLS}] = CSTEPs(H,T,n_c,\hat{\beta}_M^{OLS},\Sigma_M^{OLS},h)$
$\det_1 = \det(\hat{\Sigma}_{\hat{M}}^{OLS})$
if $\det_1 < \det_0$ **then**
$\hat{\beta}^{MLTS} = \hat{\beta}_{\hat{M}}^{OLS};\ \hat{\Sigma}^{MLTS} = \hat{\Sigma}_{\hat{M}}^{OLS}$
if $|(\det_1 - \det_0)| < 10^{-16}$ **then**
break
end if
$\det_0 = \det_1$
end if
end for
$c_\alpha = (1-\alpha)/(F_{\chi^2_{q+2}}(q_\alpha))$ *with* $q_\alpha = \chi^2_{q+2};\ \hat{\Sigma}^{MLTS} = c_\alpha \times \hat{\Sigma}^{MLTS}$

Algorithm 3. RMLTS.

Input: H, T, α, δ, *maxIter*, n_c
Output: $\hat{\beta}^{RMLTS}$
Optional outputs: $\hat{\Sigma}^{RMLTS}$
$[n,q] = size(T)$
$[\hat{\beta}^{MLTS},\ \hat{\Sigma}_{\hat{M}}^{MLTS}] = MLTS(H,T,\alpha,maxIter,n_c)$
$r = (T - H \times \hat{\beta}^{MLTS});\ ci = ECODLS(\hat{\Sigma}_{\hat{M}}^{MLTS}, I_{\hat{\Sigma}_{\hat{M}}^{MLTS}})$
$d_j = (r_j \times ci \times r_j^T)^{1/2},\ \forall j = 1,\ldots,n;\ q_\delta = \chi^2_{q,1-\delta}$
J=Select row indices where $d_j \le q_\delta;\ H_J = H(J,:);\ T_J = T(J,:)$
$\hat{\beta}^{RMLTS} = ECODLS(H_J,T_J);\ r_J = T_J - H_J \times \hat{\beta}^{RMLTS}$
if $(\sharp J - p) = 0$ **then**
$\hat{\Sigma}_J^{OLS} = (r_J^T \times r_J)/p$
else
$\hat{\Sigma}_J^{OLS} = (r_J^T \times r_J)/(\sharp J - p)$
end if // $\sharp J$ *is a number of elements in* J.
$c_\delta = (1-\delta)/(F_{\chi^2_{q+2}}(q_\delta))$ *with* $q_\delta = \chi^2_{q+2};\ \hat{\Sigma}^{RMLTS} = c_\delta \times \hat{\Sigma}_J^{OLS}$

Algorithm 4. CSTEPs.

Input: H, T, n_c, $\hat{\beta}_M^{OLS}$, Σ_M^{OLS}, h
Output: $\hat{\beta}$, $\hat{\Sigma}$
$[n,p] = size(H);\ \hat{\beta} = \hat{\beta}_M^{OLS};\ \hat{\Sigma} = \Sigma_M^{OLS}$
for $j = 1$ **to** n_c **do**
$r = T - H \times \hat{\beta};\ \Sigma^{-1} = ECODLS(\hat{\Sigma}, I_{\hat{\Sigma}})$
// *The squared of Mahalanobis distance*
$d_k = r_k \times \Sigma^{-1} \times r_k^T,\ \forall k = 1,\ldots,n$
$rs = get\ row\ indices\ (sort(d_1,\ldots,d_n))$
$M = rs(1:h);\ T_M = T(M,:);\ H_M = H(M,:)$
$\hat{\beta} = ECODLS(H_M, T_M);\ \hat{r} = T_M - H_M \times \hat{\beta}$
if $h > p$ **then**
$\hat{\Sigma} = (\hat{r}^T \times \hat{r})/(h-p)$
else
$\hat{\Sigma} = (\hat{r}^T \times \hat{r})/h$
end if
end for

3.1. The iteratively reweighted least squares extreme learning machine (IRWLS-ELM)

The IRWLS-ELM is extended from the ELM, particularly in the output weights computing. The output weights of IRWLS-ELM are computed using the improved IRWLS instead of the ordinary least squares approach. The IRWLS-ELM algorithm has three steps:

*Step*1: Randomly assign input weight w_i and hidden layer bias b_i, $i = 1,\ldots,p$.
*Step*2: Calculate the hidden layer output matrix **H**.
*Step*3: Compute the output weights $\hat{\beta}$ using Algorithm 1 *IRWLS*, thus $\hat{\beta}$ is computed by

$$\hat{\beta} = IRWLS(\mathbf{H},\mathbf{T},maxIter), \tag{19}$$

where *maxIter* is a maximum number of iterations, for this paper, it was fixed as *maxIter*=50.

3.2. Robust extreme learning machine based on the multivariate least-trimmed squares (MLTS-ELM and RMLTS-ELM)

MLTS-ELM and RMLTS-ELM are extended versions of the ELM. MLTS is used to compute the output weights and it eliminates the effect of outliers at the same time. Details of these algorithms are described below. MLTS-ELM has three steps:

*Step*1: Randomly assign input weight w_i and hidden layer bias b_i, $i = 1,\ldots,p$.
*Step*2: Calculate the hidden layer output matrix **H**.
*Step*3: Compute the output weights $\hat{\beta}$ using Algorithm 2 *MLTS*

$$\hat{\beta} = MLTS(\mathbf{H},\mathbf{T},\alpha,maxIter,n_c), \tag{20}$$

where α is a trimming proportion constant, *maxIter* is a maximum of iterations, n_c is a maximum of iterations in Algorithm 4 *CSTEPs*.

RMLSTS-ELM is the extended version of MLTS-ELM that can be summarized as follows:

*Step*1: Randomly assign input weight w_i and hidden layer bias b_i, $i = 1,\ldots,p$.
*Step*2: Calculate the hidden layer output matrix **H**.
*Step*3: Compute the output weights $\hat{\beta}$ using Algorithm 3 *RMLTS*, $\hat{\beta}$ is calculated from

$$\hat{\beta} = RMLTS(\mathbf{H},\mathbf{T},\alpha,\delta,maxIter,n_c), \tag{21}$$

where α is a trimming proportion constant according to the contamination rate, δ is a constant value to a Chi-squared test to find a boundary q_δ to accept the typical samples or reject the outlying samples, *maxIter* is a maximum number of iterations, n_c is a maximum number of iterations in Algorithm 4 *CSTEPs*.

3.3. Complexity comparison

Instead of using SVD approach during the calculation of the output weights, ECOD is used in the proposed algorithms. For matrix $\mathbf{H} \in \mathbb{R}^{n\times p}$, the complexity of SVD, is approximated to $O(4np^2+8p^3)$ flops [9]. The complexity of ECOD is $O(2np^2-2p^3/3+p^2)$ flops [24,28,29]. Therefore, by the method represented in [17], the comparison of the complexity of them is written as follows:

$$\frac{O(ECOD)}{O(SVD)} = \frac{O(2np^2-2p^3/3+p^2)}{O(4np^2+8p^3)}. \tag{22}$$

The right hand side of Eq. (22) can be derived further as

$$\frac{(2np^2-2p^3/3+p^2)}{(4np^2+8p^3)} \approx \frac{2-\frac{2p}{3n}+\frac{1}{n}}{4+\frac{8p}{n}} \leq 1. \quad (23)$$

From Eq. (23), it can be inferred that the complexity of ECOD is smaller than that of the SVD. By the same way, the complexity of ECODLS is also smaller than that of SVDLS.

4. Experimental results

The purpose of the experimental design is to compare (a) the computational robustness of ECOD and SVD in ELM and (b) the outlier robustness of the proposed algorithms, IRWLS-ELM, MLTS-ELM, and RMLTS-ELM to the ordinary ELM (OLS-ELM) and the minimax probability machine regression (MPMR). The function *pinv* of MATLAB contributed to the output weights computing of OLS-ELM. MPMR has been used to successfully detect outliers [30]. It can predict whether the output of a regression problem according to the minimum probability, ω, is within a lower bounded range, $\epsilon > 0$. The advantage of MPMR is that it can handle an input having any bounded distribution.

The experiments are divided into two categories. The first category is to compare the computational robustness of ECOD and SVD via a generated data set and an example of a real-world data set, Abalone. The second category looks at the outlier robustness of the proposed algorithms, i.e. IRWLS-ELM, MLTS-ELM, RMLTS-ELM, compared to the ordinary ELM and MPMR.

The experimental problems used in the second are divided into two types, the toy and the real-world problems. The attributes of their training and testing data sets were scaled to [0,1] and their target values were scaled to $[-1, 1]$. Each of the problems simulated sub-problems depending on the type of outlier and contamination rate. The targets of training data of each sub-problem were contaminated either by the one-sided outliers or by the two-sided outliers. The number of its contaminated samples is one of 10%, 20%, 30%, 40%, and 50% of the total number of training data. For example, the Abalone problem was simulated into 10 sub-problems. The first to the fifth sub-problems were contaminated with the one-sided outliers. Their contamination rates were 10%, 20%, 30%, 40%, and 50% respectively. The sixth to the tenth sub-problems were contaminated with the two-sided outliers and their contamination rates series were the same as the first to the fifth sub-problems. The training samples and testing samples of the specified problem were randomly generated into 50 runs. The sigmoid function $g(x) = 1/(1+\exp(-x))$ was used as an activation function for all models. The comparative algorithms were trained and tested with a data set to compare the accuracy rate and training time in each type of outlier and each of the contamination rates. As an example, the Sin problem at the contamination rate 50% was simulated into two sub-problems. The training targets of the first and the second of Sin sub-problem were contaminated with the one-sided and with the two-sided outliers respectively. The proposed algorithms, OLS-ELM and MPMR, were trained. Fig. 1(a) and (b) shows that the Sin curve of training target with the one-sided and two-sided outliers respectively. In those figures, they can be seen that the positions in the *y*-axis of outlying sample are far away from the regular Sin curve. Fig. 1(c) and (d) shows that the Sin curve of the sub-problems

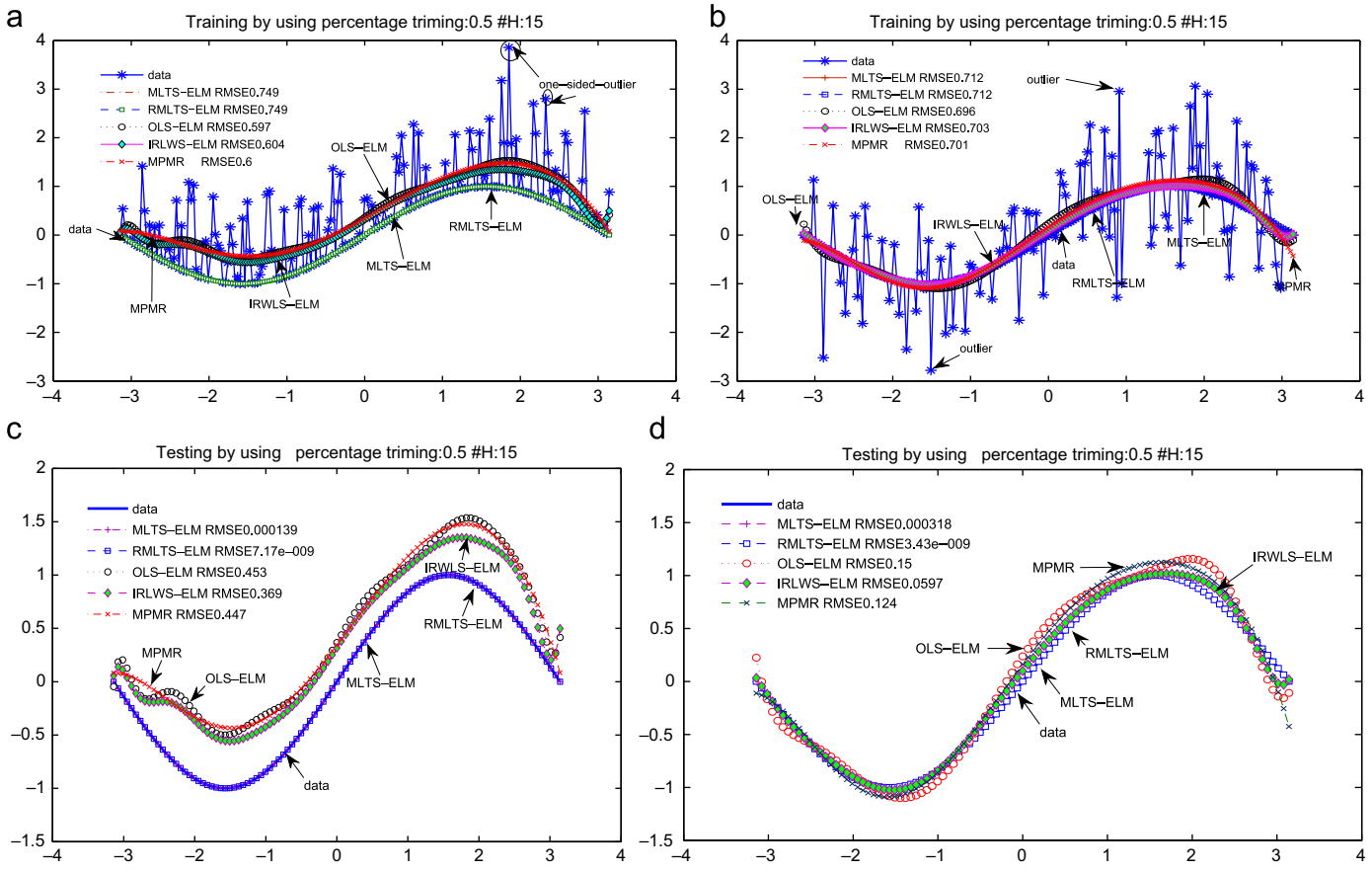

Fig. 1. An example of the estimated lines yielded from the training and testing of the Sin problem at 50% contamination rate: (a) one-sided outliers (training); (b) two-sided outliers (training); (c) one-sided outliers (testing); and (d) two-sided outliers (testing).

with the one-sided and two-sided outliers after testing respectively. All experiments were simulated within the MATLAB 7.9 environment on a Pentium Core2 Duo PC, 4 GB RAM.

The preprocessing of the MPMR has to find suitable parameters in every problem before testing and training. The RBF kernel was used with all our experiments. The best parameters $\pm\epsilon$ and the RBF scaling ϱ were selected before training. For each of the problems, the training data and the testing data were typical data without outliers and trained and tested with the all combinations of the $\pm\epsilon \in \{1e{-}1, 1e{-}5, 1e{-}10\}$ and $\varrho[0.1,0.2,\ldots,1]$. The parameters that yielded the best results were then selected. The parameters of MLTS-ELM and RMLTS-ELM are assigned as α, which is a value from the set {0.1,0.2,0.3,0.4,0.5}, $maxIter=500$, and $nc=10$. The δ for RMLTS is 0.01.

The number of attributes, samples for training and testing, hidden nodes of ELM, and the parameters of MPMR used in this paper are shown in Table 1.

Table 1
Specification of the toy and real-world problems.

Data sets	#Attr.	#Output column	# Training data	# Testing data	#Hidden nodes	RBF($\pm\epsilon$,ϱ) MPMR
Sin	1	1	201	100	15	(1e−5,1.0)
SinC	1	1	1000	1000	35	(1e−5,0.1)
Multi-output	1	2	201	100	15	(0.1,0.8)
Abalone	7	1	2000	2177	25	(1e−5,0.9)
Boston	13	1	379	127	35	(1e−5,0.9)
Protein	77	1	2072	2072	50	(1e−1,1.0)

4.1. General computational robustness comparison of ECOD and SVD

The comparison of the computational robustness of SVD and ECOD measured the time scale and accuracy of the solution to a given system equation $\mathbf{Ax}=\mathbf{b}$. Let s be a number of samples; $s \in \{100,500,1000,2000,3000,4000,5000,6000\}$. The input matrix $\mathbf{A} \in \mathbb{R}^{s\times p}$ and the target matrix $\mathbf{b} \in \mathbb{R}^{s\times q}$ were randomly generated with 50 runs, $p=50$ and $q=2$. Each matrix $\mathbf{A}$ was simulated to be a rank deficient matrix by assigning one or more of its columns to be a zeros vector. A kth zeros column is written as

$$\mathbf{A}_k = \mathbf{0}, \quad \forall k = p, p-1, \ldots, p-(j-1), \tag{24}$$

where $\mathbf{0}$ is a zeros vector, k is the column number to be zeros, $j \in \{2,4,6,8,10,12,14,16\}$ is a number of zero columns of $\mathbf{A}$. If j was assigned to 2, $s=100$, then the zeros columns are for all $k \in \{10,9\}$ columns. Its value is increased by 2 for the next sample size s such as $j=4$ for $s=500$ until the last element $j=16$ for $s=6000$. The solution $\mathbf{x}$ of each of sample s was calculated using SVD and ECOD via ECODLS. Fig. 2 shows that most average computing times using ECOD are faster than those of SVD, especially when the size of the samples is large. The accuracy is measured by the root mean square error (RMSE), which is defined as

$$\mathrm{RMSE} = \left(\frac{1}{(s-1)} \left[\sum_{i=1}^{s} (\mathbf{b} - \mathbf{A}\hat{\mathbf{x}})^2 \right] \right)^{1/2}, \tag{25}$$

where s is a number of samples, $\hat{\mathbf{x}}$ is an estimated solution of the sample size s. The averages of the RMSE of each of the samples computed using SVD are 0.2067, 0.2849, 0.2877, 0.2886, 0.2895, 0.2900, and 0.2907. These values are the same as the RMSE by ECOD. Thus, the accuracy by ECOD is comparable to SVD.

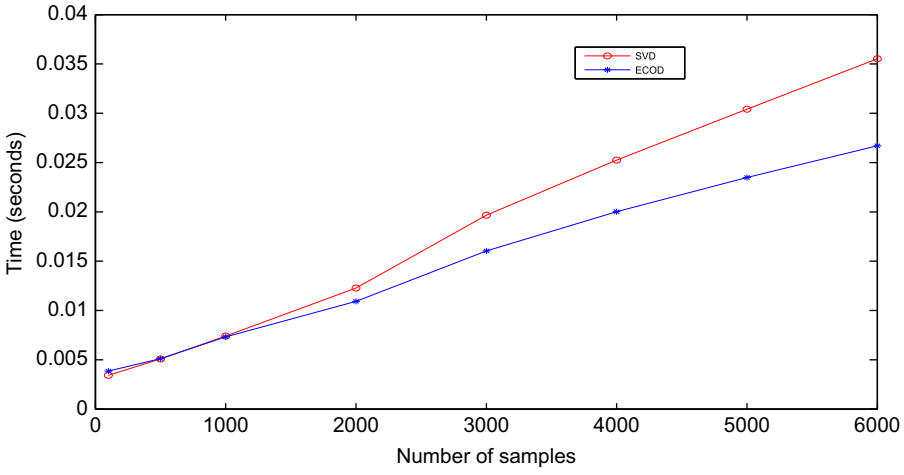

Fig. 2. Comparison of computing time scale of ECOD and SVD.

Table 2
Error results for `chow`, `lotkin`, and `prolate`, the size of all matrices is 300×300.

Matrix name	Rank	Cond	Method	$\|\mathbf{AA}^{\dagger}\mathbf{A}-\mathbf{A}\|_2$	$\|\mathbf{A}^{\dagger}\mathbf{AA}^{\dagger}-\mathbf{A}^{\dagger}\|_2$	$\|(\mathbf{AA}^{\dagger})^T-\mathbf{AA}^{\dagger}\|_2$	$\|(\mathbf{A}^{\dagger}\mathbf{A})^T-\mathbf{A}^{\dagger}\mathbf{A}\|_2$
chow	299	1.0518E+196	pinv	1.1489E−12	4.0919E−13	5.5562E−13	4.1059E−13
			ECOD	5.0439E−13	4.2346E−13	4.1587E−13	4.1422E−13
			geninv	5.7210E−09	1.9994E−11	2.1097E−11	3.6404E−09
lotkin	20	9.6892E+020	pinv	7.3284E−05	2.4557E+07	2.1707E−03	1.1277E−03
			ECOD	2.0056E−05	4.6672E−09	1.6133E−01	8.4033E−12
			geninv	1.1778E−01	4.4961E+03	1.0528E−03	1.7364E+03
prolate	168	1.8161e+017	pinv	1.1706E−04	1.1502E+09	6.3662E−03	8.6057E−03
			ECOD	3.6313E−06	6.5394E−08	7.1016E−02	2.9359E−11
			geninv	1.8962E+05	5.5478E+10	3.4275E−08	2.1477E+05

The experiment is further conducted on the three high conditional matrices namely *chow*, *lotkin*, and *protate*. They were selected from the Matrix Computation Toolbox [31]. They are both ill-conditioned and rank-deficient. The robustness of pseudo inverse computing of SVD, geninv, and ECOD algorithms were compared. The geninv is a very fast algorithm proposed by Courrieu [32]. The SVD-based pseudo inverse algorithm is the function *pinv* of MATLAB. The pseudo inverse computing of ECOD can be simply done by dropping the target matrix out, see ECODLS algorithm in the Appendix. The four errors, i.e.

$$\|\mathbf{A}\mathbf{A}^{\dagger}\mathbf{A}-\mathbf{A}\|_2, \|\mathbf{A}^{\dagger}\mathbf{A}\mathbf{A}^{\dagger}-\mathbf{A}^{\dagger}\|_2, \|(\mathbf{A}\mathbf{A}^{\dagger})^T-\mathbf{A}\mathbf{A}^{\dagger}\|_2, \|((\mathbf{A}^{\dagger}\mathbf{A})^T-\mathbf{A}^{\dagger}\mathbf{A}\|_2$$

are the metrics for the computational robustness comparison of SVD, geninv, and ECOD. Table 2 shows that ECOD approach is more reliable than SVD and geninv approaches. In case of the *lotkin* and *prolate*, the $\|\mathbf{A}^{\dagger}\mathbf{A}\mathbf{A}^{\dagger}-\mathbf{A}^{\dagger}\|_2$ of SVD and geninv are much higher than those of ECOD approach. Most of the errors of geninv for *protate* are very high except in case of $\|(\mathbf{A}\mathbf{A}^{\dagger})^T-\mathbf{A}\mathbf{A}^{\dagger}\|_2$. Although, geninv is very fast, but the results indicate that it is not reliable. However, the metrics show that ECOD is more robust to the ill-conditioned problem than SVD and geninv approaches.

For another real-world example, Abalone, the training time of the proposed algorithms, IRWLS-ELM, MLTS-ELM, and RMLTS-ELM using ECOD were compared to the training time of the proposed algorithms IRWLS-SVD-ELM, MLTS-SVD-ELM and RMLTS-SVD-ELM using SVD.

Fig. 3 shows the training time for Abalone with the one-sided and two-sided outliers at the contamination rate of 10%. It can be seen that the time scale of the proposed algorithm using ECOD is less than the time scale of the proposed algorithms using SVD in both types of outliers. The testing RMSE by IRWLS-ELM, IRWLS-SVD-ELM, MLTS-ELM, MLTS-SVD-ELM, RMLTS-ELM, and RMLTS-SVD-ELM at the one-sided outlier contamination rate of 10% are 0.0805, 0.0805, 0.0818, 0.0818, 0.0820, and 0.0819 respectively. The testing RMSE by those algorithms at the two-sided outlier contamination rate of 10% are 0.0828, 0.0828, 0.0842, 0.0843, 0.0842, and 0.0844 respectively. These results indicate that the accuracy of the proposed algorithm using ECOD is comparable to the proposed algorithms using SVD for Abalone. From the training time and accuracy rate of the two examples and considering their complexity, it can be concluded that ECOD should be used in the proposed algorithms instead of SVD.

4.2. Evaluation outlier robustness of the toy data set regression problems

The objective of this evaluation was to measure the outlier robustness of the proposed algorithms, IRWLS-ELM, MLTS-ELM, RMLTS-ELM, compared to OLS-ELM and MPMR.

The toy regression problems were generated from three functions, Sin, SinC and Multi-output function. The Sin function was generated from

$$y_i = \sin(x_i), \quad i = 1, \ldots, n. \tag{26}$$

The SinC function was generated from

$$y_i = \sin(\pi x_i)/(\pi x_i), \quad i = 1, \ldots, n. \tag{27}$$

The multi-output function is to study the multiple output approximation of MLTS-ELM and RMLTS-ELM. It is defined as

$$y_{i,j} = \begin{cases} \sin(x_i) & \text{if } j = 1, \\ \cos(x_i)\exp^{(x_i)} & \text{if } j = 2, \end{cases} \tag{28}$$

where $-\pi \le x_i \le \pi$, $i = 1, \ldots, n$.

For convenience of description in this paper, Sin, SinC, Multi-output will have alias names according to the contaminated outlier types. The aliases of Sin, SinC, Multi-output, for the target of training data that are contaminated with the one-sided outliers, are Sin(1), SinC(1), and Multi(1) respectively. The aliases of the mentioned problems for the target of training data contaminated with the two-sided outliers, are Sin(2), SinC(2), and Multi(2) respectively.

Table 3 shows accuracy and deviation of the RMSE testing for Sin(1), SinC(1), Multi(1), Sin(2), SinC(2), and Multi(2).

Fig. 4 shows the RMSE at each of the contamination rates and the changing of the RMSE of the algorithms when the outlier contamination rate was increased. For this paper, a highly robust algorithm is the algorithm whose RMSE is slightly increased whenever its contamination rate is increased. The outlier robustness can be measured by considering from a range of RMSE at contamination rates, see the meta-metrics evaluation in Section 4.4. The order of outlier affected algorithms, in ascending order, for each problem from Fig. 4(a) and (b) can be summarized as

- RMLTS-ELM, MLTS-ELM, IRWLS-ELM, MPMR, and OLS-ELM for Sin(1), Sin(2).
- RMLTS-ELM, MLTS-ELM, MPMR, IRWLS-ELM, and OLS-ELM for SinC(1), SinC(2).

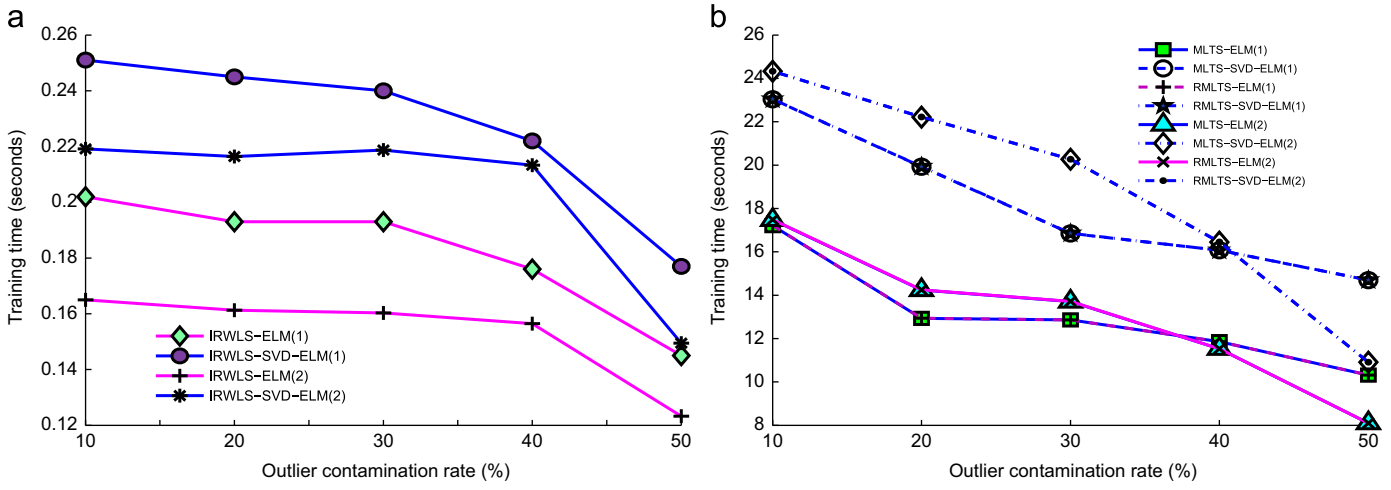

Fig. 3. Comparison of average training time of algorithms based on ECOD: IRWLS-ELM, MLTS-ELM, RMLTS-ELM, and algorithms based on SVD: IRWLS-SVD-ELM, MLTS-SVD-ELM, RMLTS-SVD-ELM and renamed these methods by appending (1) and (2) for Abalone with the one-sided and two-sided outliers respectively. (a) is for IRWLS-ELM, IRWLS-SVD-ELM and (b) is for MLTS-ELM, MLTS-SVD-ELM, RMLTS-ELM and RMLTS-SVD-ELM.

Table 3
Comparison of testing RMSE and standard deviation of IRWLS-ELM, MLTS-ELM, RMLTS-ELM, OLS-ELM and MPMR in the toy problems.

Problems	Algorithms	Testing RMSE and standard deviation (Dev.)				
		Contamination rate (%)				
		10 (RMSE ± dev.) × 10^{-4}	20 (RMSE ± dev.) × 10^{-4}	30 (RMSE ± dev.) × 10^{-4}	40 (RMSE ± dev.) × 10^{-2}	50 RMSE ± dev.
Sin(1)	IRWLS-ELM	0.0027 ± 0.0048	86.7220 ± 298.4000	688.7800 ± 298.4000	23.0430 ± 4.8696	0.3434 ± 0.0455
	MLTS-ELM	12.0810 ± 14.2020	6.9074 ± 7.6970	5.9037 ± 7.6970	0.0649 ± 0.1388	**0.1073** ± 0.3084
	RMLTS-ELM	**0.0004** ± 0.0010	**0.0002** ± 0.0002	**0.0005** ± 0.0002	**0.0245** ± 0.1438	**0.1073** ± 0.3109
	OLS-ELM	1099.4000 ± 220.3500	1980.3000 ± 274.1000	2741.2000 ± 274.1000	35.1970 ± 3.4671	0.4283 ± 0.0350
	MPMR	958.7800 ± 186.3400	1792.6000 ± 219.0300	2599.7000 ± 219.0300	33.8550 ± 3.4566	0.4146 ± 0.0326
Sin(2)	IRWLS-ELM	0.0013 ± 0.0006	0.4374 ± 3.0754	65.9330 ± 353.0500	2.0928 ± 3.4845	0.1025 ± 0.0449
	MLTS-ELM	16.4710 ± 22.4350	6.5300 ± 6.6202	5.9963 ± 6.0241	0.0711 ± 0.1639	0.0171 ± 0.0561
	RMLTS-ELM	**0.0002** ± 0.0006	**0.0005** ± 0.0019	**0.0003** ± 0.0004	**0.0256** ± 0.1702	**0.0170** ± 0.0564
	OLS-ELM	854.9400 ± 182.0900	1191.3000 ± 316.5600	1466.1000 ± 347.2200	16.8550 ± 4.2127	0.1793 ± 0.0423
	MPMR	574.5800 ± 172.7300	823.8000 ± 227.2800	975.8000 ± 279.3200	11.0400 ± 3.7810	0.1287 ± 0.0394
SinC(1)	IRWLS-ELM	863.4100 ± 3075.8000	367.6800 ± 1273.1000	3876.6000 ± 20 003.0000	182.2800 ± 553.2800	2.28270 ± 6.14200
	MLTS-ELM	**601.3200** ± 2393.9000	97.1200 ± 292.5600	112.6700 ± 289.3700	15.9200 ± 95.9340	0.41344 ± 1.12430
	RMLTS-ELM	626.7500 ± 2496.8000	**65.0840** ± 119.3700	**109.3300** ± 290.2400	**8.5732** ± 47.2450	**0.31420** ± 0.85528
	OLS-ELM	33 168.0000 ± 138 580.0000	19 385.0000 ± 68 766.0000	25 670.0000 ± 89 520.0000	147.3100 ± 436.8000	2.40630 ± 7.71870
	MPMR	1119.0000 ± 435.0800	1923.2000 ± 268.8500	2732.4000 ± 355.8300	34.4170 ± 3.3316	0.42493 ± 0.02571
SinC(2)	IRWLS-ELM	1602.7000 ± 5832.6000	367.6700 ± 958.3700	360.4100 ± 1020.6000	11.9990 ± 52.4430	0.5930 ± 1.2817
	MLTS-ELM	904.6300 ± 3474.7000	176.9700 ± 564.2000	553.0200 ± 2883.7000	4.1258 ± 12.3280	0.0818 ± 0.2535
	RMLTS-ELM	**895.9000** ± 3313.9000	**129.4300** ± 412.3700	**189.3300** ± 617.8200	**3.1041** ± 9.1087	**0.0546** ± 0.1640
	OLS-ELM	14 293.0000 ± 35 928.0000	30 161.0000 ± 141 980.0000	18 696.0000 ± 52 399.0000	428.6300 ± 1854.0000	1.9222 ± 5.1584
	MPMR	793.4200 ± 469.4800	1135.0000 ± 414.4300	1484.4000 ± 541.4500	15.6400 ± 4.8714	0.1705 ± 0.0535
Multi(1)	MLTS-ELM	68.5610 ± 51.9980	58.2750 ± 78.5470	91.9440 ± 268.6200	0.6170 ± 1.0687	0.0069 ± 0.0136
	RMLTS-ELM	0.0776 ± 0.0826	0.0795 ± 0.0957	4.5693 ± 31.7740	0.2243 ± 1.1298	**0.0032** ± 0.0144
	OLS-ELM	1186.8000 ± 212.1600	1921.7000 ± 144.6500	2714.4000 ± 205.1400	35.2610 ± 2.4963	0.4317 ± 0.0284
	MPMR	1063.7000 ± 138.3400	1788.5000 ± 138.2800	2587.2000 ± 190.4200	34.1260 ± 2.2271	0.4200 ± 0.0265
Multi(2)	MLTS-ELM	84.5980 ± 68.8560	57.9650 ± 56.2590	47.3030 ± 63.3280	0.3890 ± 0.2399	0.0083 ± 0.0161
	RMLTS-ELM	0.0783 ± 0.1039	0.0635 ± 0.0499	**0.0791** ± 0.0889	**0.0008** ± 0.0012	**0.0046** ± 0.0171
	OLS-ELM	814.0000 ± 229.3600	1198.6000 ± 203.3700	1445.9000 ± 232.0500	16.5110 ± 3.3193	0.1902 ± 0.0279
	MPMR	639.4700 ± 165.1700	938.0100 ± 167.0600	1101.8000 ± 194.1500	12.2760 ± 2.7605	0.1445 ± 0.0239

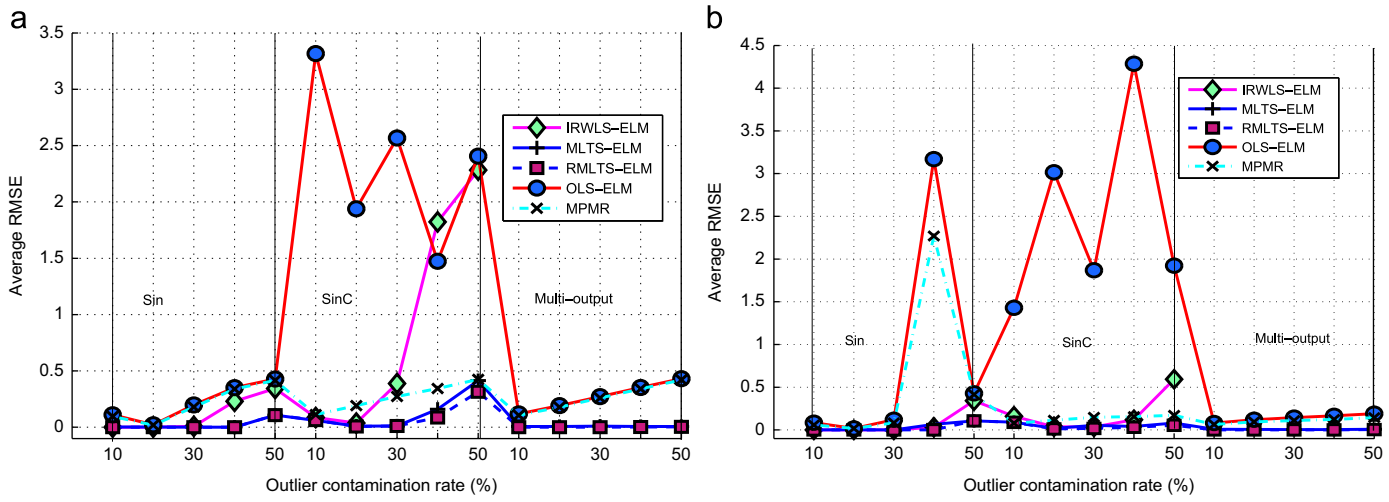

Fig. 4. Comparison of the average RMSE for testing data sets in the toy problems: (a) one-sided outliers and (b) two-sided outliers.

- RMLTS-ELM, MLTS-ELM, MPMR, and OLS-ELM for Multi(1), Multi(2).

4.3. *Evaluation of the outlier robustness of the real-world regression problems*

The real-world regression problems are Abalone, Boston, and Protein data sets. The first three data sets came from UCI [33]. The Protein data came from [34]. The last attribute of the Protein problem is selected to be a target. Table 1 shows the basic attributes for the real-world regression problems. The Abalone, Boston and Protein are single-output problems. For convenience of description, the shortened names of Abalone, Boston, and Protein that were contaminated with the one-sided outliers are Abalone(1), Boston(1), and Protein(1) respectively. By the same way the alias names for those problems were contaminated with the two-sided outliers are Abalone(2), Boston(2), and Protein(2) respectively.

Table 4 shows the comparison of training times for the real-world problems that their target matrix contaminated with both the two types of outliers. It shows that OLS-ELM is still the fastest training algorithm while the proposed algorithms are very faster than MPMR, especially when the size of samples is large. For example in Protein(1), the training times of IRWLS-ELM, MLTS-ELM, RMLTS-ELM, and MPMR are 1.87, 50.40, 50.48, and 1575.80 s respectively. The training time of IRWLS-ELM is less than MLTS-ELM, RMLTS-ELM. RMLTS-ELM is slightly slower than MLTS-ELM.

Tables 5 and 6 show the average testing results of RMSE for the real-world problems contaminated with the one-sided outliers and two-sided outliers respectively. Fig. 5 shows the changes of testing the RMSE of the comparative algorithms for the real-world problems when the number of outliers is increased. It can be seen that the proposed algorithms are less affected by the influence of outliers increasing than OLS-ELM and MPMR. Fig. 5(a), the order of the level of changing the RMSE of these algorithms, in ascending order, for the real-world problems in this figure can be summarized as RMLTS-ELM, MLTS-ELM, IRWLS-ELM, OLS-ELM, and MPMR. IRWLS-ELM is more affected by the one-sided outliers at the contamination rates of 30–50% for all of the real-world problems than MLTS-ELM and RMLTS-ELM. Fig. 5(b) shows the effects of the number of the two-sided outliers increasing via the changing of RMSE for the real-world problems. It can be seen that the proposed algorithms are still less affected the effects of outliers increasing than OLS-ELM and MPMR. The detail of the outlier robustness comparison is in Section 4.4.

4.4. Meta-metrics evaluation of the outlier robustness

The overall picture of all the experimental results, i.e. Tables 3, 5, and 6 will be concluded by the meta-metrics evaluation. This measurement has been used to measure the quality of E-commerce systems [35]. This paper applies the meta-metrics to evaluate the outlier robustness of the comparative algorithms. There are five selected metrics involved in the meta-metric

Table 4
Comparison of training time of IRWLS-ELM, MLTS-ELM, RMLTS-ELM, OLS-ELM and MPMR in the real-world regression problems.

Problems	Algorithms	Training time (s)										
		One-sided outlier contamination rate (%)					Two-sided outlier contamination rate (%)					Time (s) average
		10	20	30	40	50	10	20	30	40	50	
Abalone	IRWLS-ELM	27.38	23.70	65.12	55.64	56.82	9.09	13.24	13.12	16.56	13.01	29.37
	MLTS-ELM	81.70	110.36	249.24	253.38	240.34	675.85	564.76	268.74	639.30	640.41	372.41
	RMLTS-ELM	81.71	110.36	250.55	254.71	241.01	675.95	564.83	268.95	639.56	640.60	372.82
	OLS-ELM	1.67	1.86	3.79	3.39	2.93	0.74	1.22	0.83	1.18	1.12	1.87
	MPMR	389.47	389.70	629.32	679.38	633.93	365.69	381.54	358.49	374.04	390.04	459.16
Boston	IRWLS-ELM	1.25	3.93	4.43	8.79	5.85	4.23	0.80	3.64	0.53	3.38	3.68
	MLTS-ELM	21.69	22.11	18.65	19.70	20.96	27.09	24.93	108.33	38.88	47.77	35.01
	RMLTS-ELM	21.70	22.11	18.65	19.70	20.96	27.10	24.93	108.36	38.88	47.77	35.02
	OLS-ELM	0.30	0.16	0.19	0.53	0.66	0.44	0.10	0.12	0.06	0.12	0.27
	MPMR	11.07	16.76	13.49	10.58	12.07	13.21	11.82	13.40	10.57	9.66	12.26
Protein	IRWLS-ELM	1.87	2.31	1.64	0.22	0.29	91.33	42.38	62.13	7.01	1.69	21.09
	MLTS-ELM	50.40	40.59	40.82	22.50	21.06	404.08	552.73	280.18	32.60	32.80	147.78
	RMLTS-ELM	50.48	40.66	40.96	22.50	21.07	407.99	553.23	282.95	32.86	32.84	148.55
	OLS-ELM	0.86	1.01	0.89	0.02	0.12	6.19	4.71	5.02	2.39	0.53	2.17
	MPMR	1575.90	1558.20	1563.20	1232.00	1298.90	2230.47	2322.21	1962.84	1650.88	1665.12	1705.97

Table 5
Comparison of testing RMSE of IRWLS-ELM, MLTS-ELM, RMLTS-ELM, OLS-ELM and MPMR in the real-world regression problems that training data sets contaminated with the one-sided outliers.

Problems	Algorithms	Testing RMSE and standard deviation (Dev.)				
		One-sided outlier contamination rate (%)				
		10 RMSE ± dev.	20 RMSE ± dev.	30 RMSE ± dev.	40 RMSE ± dev.	50 RMSE ± dev.
Abalone(1)	IRWLS-ELM	**0.0811** ± 0.0088	0.0890 ± 0.0081	0.1271 ± 0.0129	0.2206 ± 0.0168	0.3317 ± 0.0213
	MLTS-ELM	0.0818 ± 0.0078	**0.0808** ± 0.0102	0.0826 ± 0.0116	0.0918 ± 0.0188	**0.0942** ± 0.0208
	RMLTS-ELM	0.0816 ± 0.0077	0.0812 ± 0.0099	**0.0813** ± 0.0072	**0.0902** ± 0.0147	0.0986 ± 0.0181
	OLS-ELM	0.1338 ± 0.0239	0.2009 ± 0.0284	0.2714 ± 0.0188	0.3495 ± 0.0241	0.4241 ± 0.0232
	MPMR	0.1537 ± 0.0142	0.2332 ± 0.0189	0.3075 ± 0.0167	0.3814 ± 0.0183	0.4607 ± 0.0180
Boston(1)	IRWLS-ELM	**0.0795** ± 0.0053	0.1172 ± 0.0136	0.1627 ± 0.0218	0.2602 ± 0.0234	0.3782 ± 0.0260
	MLTS-ELM	0.0804 ± 0.0067	0.0997 ± 0.0107	0.0975 ± 0.0118	0.1057 ± 0.0108	0.3303 ± 0.5714
	RMLTS-ELM	0.0805 ± 0.0065	**0.0996** ± 0.0103	**0.0960** ± 0.0112	**0.1027** ± 0.0100	**0.2929** ± 0.4598
	OLS-ELM	0.1445 ± 0.0183	0.2422 ± 0.0281	0.3067 ± 0.0317	0.3637 ± 0.0265	0.4548 ± 0.0270
	MPMR	0.9025 ± 0.3068	1.2549 ± 0.3076	1.3710 ± 0.3589	1.3850 ± 0.3634	1.5967 ± 0.3051
Protein(1)	IRWLS-ELM	0.1620 ± 0.0181	**0.1535** ± 0.0125	**0.1571** ± 0.0142	0.2091 ± 0.0386	0.2797 ± 0.0497
	MLTS-ELM	0.1731 ± 0.0226	0.1748 ± 0.0220	0.1825 ± 0.0326	0.1759 ± 0.0283	0.1692 ± 0.0207
	RMLTS-ELM	0.1700 ± 0.0220	0.1666 ± 0.0190	0.1649 ± 0.0236	**0.1566** ± 0.0171	**0.1537** ± 0.0098
	OLS-ELM	**0.1589** ± 0.0132	0.1731 ± 0.0188	0.2206 ± 0.0450	0.2931 ± 0.0531	0.3614 ± 0.0611
	MPMR	0.3339 ± 0.0763	0.5499 ± 0.2081	0.6781 ± 0.2824	0.7868 ± 0.3760	1.0919 ± 0.5257

Table 6
Comparison of testing RMSE of IRWLS-ELM, MLTS-ELM, RMLTS-ELM, OLS-ELM and MPMR in the real-world problems with training data contaminated with the two-sided outliers.

Problems	Algorithms	Testing RMSE and standard deviation (Dev.)				
		Two-sided outlier contamination rate (%)				
		10 RMSE ± dev.	20 RMSE ± dev.	30 RMSE ± dev.	40 RMSE ± dev.	50 RMSE ± dev.
Abalone (2)	IRWLS-ELM	**0.0787** ± 0.0063	**0.0817** ± 0.0109	0.0865 ± 0.0166	0.0906 ± 0.0188	0.0953 ± 0.0192
	MLTS-ELM	0.0814 ± 0.0095	0.0834 ± 0.0119	0.0845 ± 0.0136	**0.0851** ± 0.0143	0.0900 ± 0.0163
	RMLTS-ELM	0.0812 ± 0.0093	0.0831 ± 0.0132	**0.0836** ± 0.0119	0.0857 ± 0.0144	**0.0867** ± 0.0165
	OLS-ELM	0.0984 ± 0.0197	0.1200 ± 0.0381	0.1397 ± 0.0499	0.1499 ± 0.0555	0.1404 ± 0.0502
	MPMR	0.1307 ± 0.0237	0.1651 ± 0.0310	0.1832 ± 0.0345	0.2173 ± 0.0438	0.2449 ± 0.0517
Boston (2)	IRWLS-ELM	**0.1067** ± 0.0076	**0.1025** ± 0.0104	0.1378 ± 0.0160	0.1251 ± 0.0169	**0.1895** ± 0.0349
	MLTS-ELM	0.1156 ± 0.0089	0.1146 ± 0.0214	0.1399 ± 0.0296	0.1070 ± 0.0244	0.2364 ± 0.5984
	RMLTS-ELM	0.1159 ± 0.0087	0.1146 ± 0.0221	**0.1371** ± 0.0281	**0.1045** ± 0.0230	0.2015 ± 0.4089
	OLS-ELM	0.1537 ± 0.0209	0.1724 ± 0.0209	0.2348 ± 0.0324	0.2126 ± 0.0332	0.2655 ± 0.0480
	MPMR	1.1031 ± 0.2795	1.5221 ± 0.3688	1.5655 ± 0.3048	1.6944 ± 0.3845	2.1353 ± 0.4154
Protein (2)	IRWLS-ELM	0.1762 ± 0.0248	**0.1726** ± 0.0263	**0.1705** ± 0.0186	0.1787 ± 0.0246	0.1831 ± 0.0242
	MLTS-ELM	0.1787 ± 0.0257	0.1772 ± 0.0300	0.1780 ± 0.0220	0.1844 ± 0.0321	0.1916 ± 0.0334
	RMLTS-ELM	**0.1760** ± 0.0247	0.1729 ± 0.0281	0.1706 ± 0.0176	**0.1769** ± 0.0277	**0.1821** ± 0.0279
	OLS-ELM	0.1833 ± 0.0289	0.1851 ± 0.0271	0.1915 ± 0.0265	0.2063 ± 0.0292	0.2079 ± 0.0303
	MPMR	0.2807 ± 0.0734	0.4740 ± 0.2046	0.6410 ± 0.3186	1.0193 ± 0.5201	1.0824 ± 0.6807

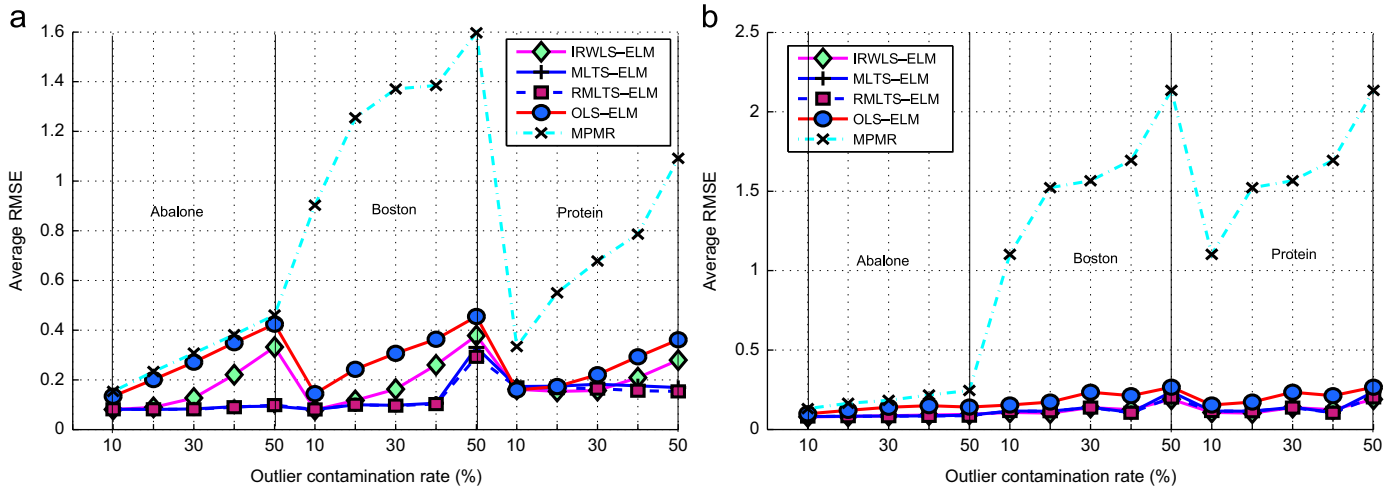

Fig. 5. Comparison of testing RMSE of IRWLS-ELM, MLTS-ELM, RMLTS-ELM, OLS-ELM and MPMR in the real-world problems: (a) one-sided outliers and (b) two-sided outliers.

evaluation. The first is a minimal average RMSE at each level of the contamination rate. The second is a minimal deviation of RMSE at each level of the contamination rate. The third, fourth, and fifth are the ranges of RMSE (maximum–minimum) at each level for the contamination rates of 10–30%, 40–50%, and 10–50% respectively. Their objective is to measure the influence of an increase in the number of outliers over the capability of the algorithms at the low level contamination rates (10–30%), the high level contamination rates (40–50%), and the overall picture (10–50%). For each metric, lower is better. If an algorithm has a minimal of each of their values then its score is assigned a winning score of 1, all other algorithms are assigned as 0. The counting scores of each algorithm have three subtotals. The subtotal (a) is a subtotal of scores at the low level contamination rate. The subtotal (b) is a subtotal of scores at the high level contamination rate. The subtotal (a) and (b) of each algorithm are calculated by the summation of the minimum average RMSE and deviation scores. The subtotal (c) is a subtotal of the overall picture scores (10–50%). (c) is the score of a minimum average RMSE. The total value of (a), (b), and (c) is the summation of the subtotal (a), the subtotal (b), and the subtotal (c). The total score expresses the outlier robustness of each algorithm. The higher the score it gets, the greater the robustness it has. For example, the computing of IRWLS's score in Table 7; the subtotal (a) is 16, which is computed from the summation between the score of RMSE and deviation at 10–30% (8+8). The subtotal (b) is 3, which is a yielding of summation of the score of RMSE and deviation at 40–50% (0+3). The subtotal (c) is 0. The total score of IRWLS-ELM is 19 (16+3+0). Using the same method, the total score of MLTS-ELM is 18. This means that IRWLS-ELM is less affected by the outliers than MLTS-ELM. A league table of these scores is shown in Table 7. Table 7 shows the comparison of the scores of the metrics for Sin, SinC, Abalone, Boston, Protein problems.

Table 7 shows that RMLTS-ELM gets the highest total score among the comparative algorithms at all levels of contamination rates. IRWLS-ELM came second and MLTS-ELM is the third. This means that RMLTS-ELM is less affected by the increasing number of outliers than the others but it still takes a longer time to train (see Table 4). IRWLS-ELM gets a higher subtotal score than MLTS-ELM at the low level of contamination rate. On the other hand MLTS-ELM gets a higher subtotal score than IRWLS-ELM at the high level of contamination rate. This means that IRWLS-ELM

Table 7
Comparison of score league of testing RMSE in the Sin, SinC, Abalone, Boston and Protein problems.

Algorithms	Scores at contamination rates									Total (a)+(b)+(c)
	10–30%				40–50%				10–50%	
	RMSE		Dev.	Subtot.	RMSE		Dev.	Subtot.	RMSE	
	Min.	Range	Min.	(a)	Min.	Range	Min.	(b)	Min. (c)	
IRWLS-ELM	8	0	8	16	0	0	3	3	0	19
MLTS-ELM	2	2	1	5	3	2	4	9	4	18
RMLTS-ELM	15	6	15	36	15	5	4	24	5	65
OLS-ELM	1	0	1	2	0	2	0	2	0	4
MPMR	1	1	3	5	0	0	7	7	0	12

should be used in the lower level contamination rates and MLTS-ELM should be used in the higher level contamination rates.

5. Conclusion

This paper solves the two robustness issues of ELM. The first issue is solved by improving the computational robustness of common sub-problems like $\mathbf{Ax}=\mathbf{b}$ using ECOD via ECODLS. The second is by proposing outlier robustness enhancement using three new algorithms: IRWLS-ELM, MLTS-ELM, and RMLTS-ELM. Our experimental results can be summarized as follows:

(a) ECOD can increase the computational robustness of ELM and produces more reliable and faster computing times, in case the large data set, than SVD.
(b) The proposed algorithm yields are less affected by increasing the number of outliers than the existing OLS-ELM and MPMR.

From the above summaries, we can conclude that the proposed algorithms can be used to improve both the computational and outlier robustness of ELM. They all outperform MPMR.

Future work on this project: IRWLS-ELM should be enhanced when using higher contamination rates and its behavior studied with other M-estimate functions, i.e. not only limited to the Huber function.

Acknowledgments

This work was partially supported by the Higher Education Research Promotion and National Research University Project of Thailand, Office of the Higher Education Commission, through the Cluster of Research to Enhance the Quality of Basic Education and Department of Computer Science, Faculty of Science, Khon Kaen University. This research is a part of the Computational Science Research Group (COSRG), Faculty of Science, Khon Kaen University (COSRG-SCKKU). We would also thank Mr. James McCloskey for proofreading the manuscript of this paper. KSN would like to thank Prof. C. Lursinsap of AVIC for the discussion.

Appendix A

A.1. Extended complete orthogonal decomposition (ECOD)

Let $\mathbf{A}$ be a matrix size $m\times n$ and r be the rank of $\mathbf{A}$. The decomposition of $\mathbf{A}$ by ECOD, see Algorithm 5, has two steps. The matrix $\mathbf{A}$ is firstly decomposed by rank-revealing QR [9,29] as

$$\mathbf{A}\Pi=\mathbf{QR}=\mathbf{Q}\begin{bmatrix}\mathbf{R}_{11} & \mathbf{R}_{12}\\ \mathbf{R}_{21} & \mathbf{R}_{22}\end{bmatrix},\quad r=rank(\mathbf{A}), \tag{A.1}$$

where $\mathbf{Q}\in\mathbb{R}^{m\times m}$ is orthogonal, Π is the permutation matrix, and $\mathbf{R}_{11}\in\mathbb{R}^{r\times r}$ is the upper triangular and nonsingular matrix, $\mathbf{R}_{12}\in\mathbb{R}^{r\times n-r}$ is the $r\times(n-r)$ matrix. Both $\mathbf{R}_{21}$ and $\mathbf{R}_{22}$ are zeros matrices whenever $\mathbf{A}$ is rank deficient, $r<\min(m,n)$. The second step is to select a method to further decompose the transposed matrix of the first r rows in $\mathbf{R}$; $\mathbf{L}=[\mathbf{R}_{11}\ \mathbf{R}_{12}]^T\in\mathbb{R}^{n\times r}$. $\mathbf{L}$ is a lower trapezoidal matrix. Based on considering the $rank(\mathbf{A})$ and size of $\mathbf{A}$, there are three cases. The first case is the case of $rank(\mathbf{A})\neq n$ and $m\geq n$. The second case is the case of $rank(\mathbf{A})\neq n$ and $n<m$. The last case, $rank(\mathbf{A})$ is equal to the number of columns. In the first case, $\mathbf{L}$ is further decomposed by Householder transformations [9, p. 256], this decomposition algorithm is implemented by Algorithm 6 *trap2tri* [31], while the second case $\mathbf{L}$ is further decomposed by QR. In the third case, $\mathbf{A}$ is decomposed by QR where $\mathbf{R}$ is the $n\times n$ matrix and $rank(\mathbf{A})=n$. The simple form of the matrix $\mathbf{A}$ of the first and second steps can be written as

$$\mathbf{A}=\mathbf{Q}\begin{bmatrix}\mathbf{U}_{11}^T & 0\\ 0 & 0\end{bmatrix}\mathbf{V}, \tag{A.2}$$

where $\mathbf{U}_{11}^T\in\mathbb{R}^{r\times r}$ is either the upper triangular matrix for the first case or the lower triangular matrix for the second case, $\mathbf{V}=\Pi\overline{\mathbf{V}}_1\cdots\overline{\mathbf{V}}_r$ is the column interchanged matrix of $\overline{\mathbf{V}}$. $\overline{\mathbf{V}}=[\overline{\mathbf{V}}_r\cdots\overline{\mathbf{V}}_1]$ is the Householder transformations matrix of $\mathbf{L}$ [9,36] for the first case. While $\mathbf{V}=\overline{\mathbf{V}}\Pi$ in the second case is the row interchanged matrix of $\overline{\mathbf{V}}$ such that $\mathbf{L}=\overline{\mathbf{V}}\mathbf{U}_{11}$. The overall complexity of ECOD is $O(2mn^2-2n^3/3+n^2)$.

A.2. Solving the least squares problem by SVDLS

The algorithm for solving least squares problem using SVD in this paper is named as SVDLS. It has two steps. The first step, $\mathbf{A}\in\mathbb{R}^{m\times n}$ is decomposed by SVD and then the minimum norm solution is computed as

$$x=\mathbf{V}\mathbf{S}^{-1}\mathbf{U}^T\mathbf{b}, \tag{A.3}$$

where $\mathbf{V}$ and $\mathbf{U}$ are the orthogonal matrices, $\mathbf{S}^{-1}=diag(1/\sigma_1\cdots1/\sigma_r\cdots0)$, and r is the rank of $\mathbf{A}$, see details in [9,36,37].

A.3. Solving the least squares problem by ECODLS

ECODLS can solve the least squares problem, which has two steps [9,37]. The first is to decompose a matrix $\mathbf{A}$ using ECOD. The second step is to solve a solution $\mathbf{x}$. There are two cases in the second step: $m\geq n$ or $m<n$. The minimum norm solution of

the least squares problem is written in the general form as

$$x = \hat{\mathbf{V}}(\mathbf{U}_{11}^T)^{-1}\mathbf{Q}_{:,1:r}^T\mathbf{b}, \tag{A.4}$$

where $\hat{\mathbf{V}}$ has two cases; the first case $m \geq n$ is true then $\hat{\mathbf{V}} = \mathbf{V}_{1:r,:}$, which are the first r row vectors of $\mathbf{V}$, or $\hat{\mathbf{V}} = \mathbf{V}_{:,1:r} \in \mathbb{R}^{r\times r}$ for the case of $m < n$, which are the first r column vectors of $\mathbf{V}$, and $\mathbf{U}_{11}$ is obtained from ECOD. ECOD, *trap2tri*, and ECODLS can be summarized in Algorithm 5 ECOD, Algorithm 6 *trap2tri*, and Algorithm 7 ECODLS respectively.

Algorithm 5. ECOD.

Input: A, tol
Output: Q, R, V
Optional outputs: r, Z, P
$[m,n] = size(A)$; $[Q,Z,P] = QR(A,0)$
if tol was not defined **then**
$tol = \max(m,n) \times 10^{-16} \times |Z(1,1)|$
end if
$r = \sum(|diag(Z)| > tol)$; $r = r(1)$
if $r \neq n$ **then**
$PP(P) = 1:n$; $L = Z(1:r,:)^T$
if $m \geq n$ **then**
$[Q_1, U_{11}] = trap2tri(L)$; $V = Q_1(:,PP)$; $R = U_{11}^T$
else
$[Q_1, U_{11}] = QR(L,0)$; $V = Q_1(PP,:)$; $R = U_{11}^T$
end if
else
$I = I_n$; $V = I(P,:)$; $R = Z(1:r,:)$
end if

Algorithm 6. *trap2tri*.

Input: L
Output: Q_1, U
$[n,r] = \text{size}(L)$; $Q1 = I_n$
if $r \neq n$ **then**
$j = r$
for $j = r$ **to** 1 **do**
$w = j:n$; $x = L(w,j)$; $x(2:r-j+1) = zeros(r-j,1)$
$s = \|x\| \times (\text{sign}(x(1)) + (x(1) == 0))$
if $s \neq 0$ **then**
$x(1) = x(1) + s$; $b = s \times x(1)$
$xt = (x^T)/b$; $L(j,j) = -s$; $i = (1:j-1)$
if $j \geq 1$ **then**
$y = xt \times L(w,i)$; $L(w,i) = L(w,i) - x \times y$
end if
$y = xt \times Q1_1(w,:)$
for $k = j$ **to** n **do**
$Q_1(k,:) = Q_1(k,:) - x(k-j+1) \times y$
end for
end if
end for
end if
$U = L(1:r,:)$

Algorithm 7. ECODLS. // Calculate a solution of $Ax = b$ or $A^\dagger$ if b is eliminated.

Input: A, b
Output: x
Optional outputs: r, Z, P
$[Q,R,V,r,Z,P] = ECOD(A)$; $[m,n] = \text{size}(A)$
if $m \geq n$ **then**
$x = V(1:r,:)^T \times R^{-1} \times Q(:,1:r)^T \times b$
else
$x = V(:,1:r) \times R^{-1} \times Q(:,1:r)^T \times b$
end if

References

[1] G.-B. Huang, Q.-Y. Zhu, C.-K. Siew, Extreme learning machine: theory and applications, Neurocomputing 70 (2006) 489–501.
[2] G.-B. Huang, D.H. Wang, Y. Lan, Extreme learning machines: a survey, Int. J. Mach. Learn. Cybern. 2 (2) (2011) 107–122.
[3] G.-B. Huang, H. Zhou, X. Ding, R. Zhang, Extreme learning machine for regression and multiclass classification, IEEE Trans. Syst. Man Cybern.: Part B 42 (2) (2012) 513–529.
[4] X. Wang, A. Chen, H. Feng, Upper integral network with extreme learning mechanism, Neurocomputing 74 (16) (2011) 2520–2525.
[5] C.-W.T. Yeu, M.-H. Lim, G.-B. Huang, A new machine learning paradigm for terrain reconstruction, IEEE Geosci. Remote Sensing Lett. 3 (3) (2006) 382–386.
[6] A.U. Bhat, S.S. Merchant, S.S. Bhagwat, Prediction of melting points of organic compounds using extreme learning machines, Ind. Eng. Chem. Res. 47 (2008) 920–925.
[7] Z.-L. Sun, T.-M. Choi, K.-F. Au, Y. Yu, Sales forecasting using extreme learning machine with applications in fashion retailing, Decision Support Syst. 46 (1) (2008) 411–419.
[8] S.O. Olatunji, A. Selamat, A.A.A.R. Abdul, Modeling permeability prediction using extreme learning machines, in: Proceedings of the 2010 Fourth Asia International Conference on Mathematical/Analytical Modelling and Computer Simulation, AMS '10, IEEE Computer Society, Washington, DC, USA, 2010, pp. 29–33.
[9] G.H. Golub, C.F.V. Loan, Matrix Computations, third ed., Johns Hopkins University Press, MD, 1996.
[10] R. Maronna, R. Martin, V. Yohai, Robust Statistics Theory and Methods, Wiley, Chichester, West Sussex, England, 2006.
[11] Y. Wang, F. Cao, Y. Yuan, A study on effectiveness of extreme learning machine, Neurocomputing 74 (16) (2011) 2483–2490.
[12] Y. Yuan, Y. Wang, F. Cao, Optimization approximation solution for regression problem based on extreme learning machine, Neurocomputing 74 (16) (2011) 2475–2482.
[13] H.T. Huynh, Y. Won, Weighted least squares scheme for reducing effects of outliers in regression based on extreme learning machine, Int. J. Digital Content Technol. Appl. (JDCTA) 2 (3) (2008) 40–46.
[14] W. Deng, Q. Zheng, L. Chen, Regularized extreme learning machine, in: IEEE Symposium on Computational Intelligence and Data Mining, CIDM '09, 2009, pp. 389–395.
[15] G.-B. Huang, L. Chen, C.-K. Siew, Universal approximation using incremental constructive feedforward networks with random hidden nodes, IEEE Trans. Neural Networks 17 (4) (2006) 879–892.
[16] G.-B. Huang, L. Chen, Enhanced random search based incremental extreme learning machine, Neurocomputing 71 (2008) 3460–3468.
[17] X. Tang, M. Han, Partial Lanczos extreme learning machine for single-output regression problems, Neurocomputing 72 (13–15) (2009) 3066–3076.
[18] P.J. Rousseeuw, A.M. Leroy, Robust Regression and Outlier Detection, John Wiley & Sons, Inc., New York, 1987.
[19] P.J. Huber, Robust Statistics, John Wiley & Sons, 1981.
[20] G.A.N. Mbamalu, M.E. El-Hawary, Loading forecasting via suboptimal seasonal autoregressive models and iteratively reweighted least squares estimation, IEEE Trans. Power Syst. 8 (1) (1993) 343–348.
[21] G.A.N. Mbamalu, M.E. El-Hawary, F. El-Hawary, No_x emission modelling using the iteratively reweighted least-square procedures, Int. J. Electr. Power Energy Syst. 17 (2) (1995) 129–136.
[22] J.O. Street, R.J. Carroll, D. Ruppert, A note on computing robust regression estimates via iteratively reweighted least squares, Am. Stat. 42 (2) (1998) 152–154.
[23] W. Dumouchel, F. O'Brien, Integrating a Robust Option into a Multiple Regression Computing Environment, Springer-Verlag New York, Inc., New York, 1991, pp. 41–48.
[24] P.D. Hough, S.A. Vavasis, Complete orthogonal decomposition for weighted least squares, SIAM J. Matrix Anal. Appl. 18 (1997) 369–392.
[25] P.J. Rousseeuw, K.V. Driessen, A fast algorithm for the minimum covariance determinant estimator, Technometrics 41 (1999) 212–223.
[26] J. Agulló, C. Croux, S.V. Aelst, The multivariate least-trimmed squares estimator, J. Multivariate Anal. 99 (3) (2008) 311–338.
[27] C. Croux, K. Joossens, Robust estimation of the vector autoregressive model by a least trimmed squares procedure, in: COMPSTAT 2008: Proceedings in Computational Statistics, 2008, pp. 489–501.
[28] N.J. Higham, R.S. Schreiber, Fast polar decomposition of an arbitrary matrix, SIAM J. Sci. Stat. Comput. 11 (1990) 648–655.
[29] C.R. Goodall, Computation using the QR decomposition, Handbook of Statistics, vol. 9, Elsevier Science Publishers B.V., Amsterdam, The Netherlands, 1993, pp. 467–508 (Chapter 13).
[30] T. Strohmann, G.Z. Grudic, A formulation for minimax probability machine regression, in: Advances in Neural Information Processing Systems, vol. 15, MIT Press, 2003, pp. 769–776.
[31] N.J. Higham, The Matrix Computation Toolbox, URL ⟨http://www.ma.man.ac.uk/~higham/mctoolbox⟩.

[32] P. Courrieu, Fast computation of Moore–Penrose inverse matrices, Neural Inf. Process.—Lett. Rev. 8 (2) (2005) 25–29.
[33] A. Frank, A. Asuncion, UCI Machine Learning Repository, 2010. URL 〈http://archive.ics.uci.edu/ml〉.
[34] J.A. Khan, S.V. Aelst, R.H. Zamar, Robust linear model selection based on least angle regression, J. Am. Stat. Assoc. 102 (2007) 1289–1299.
[35] A. Stefani, M. Xenos, Meta-metric evaluation of e-commerce-related metrics, Electron. Notes Theor. Comput. Sci. 233 (2009) 59–72.
[36] Å. Björck, Numerical Methods for Least Squares Problems, SIAM, Philadelphia, 1996.
[37] D.S. Watkins, Fundamentals of Matrix computation, second ed., John Wiley & Sons, New York, 2002.

Punyaphol Horata received the B.Sc. degree in Mathematics from Khon Kaen University, and M.S. degree in Computer Science from Chulalongkorn University, Thailand. He is currently a Ph.D. student at Khon Kaen University, Thailand. His research interests include machine learning, soft computing, and pattern recognition.

Sirapat Chiewchanwattana graduated in Statistics from Khon Kaen University, Thailand. She received her M.Sc. in Computer Science from the National Institute of Development Administration, Thailand and Ph.D. in Computer Science from Chulalongkorn University in 2007. She works as an Assistant Professor in the department of Computer Science at Khon Kaen University, Thailand and joined in research group in Intelligence System and Machine Learning. Her research interests are in neural networks, soft computing, and pattern recognition.

Khamron Sunat graduated in chemical engineering in Chulalongkorn University, Thailand, in 1989. He received his M.Sc. in Computational Science in 1998 and Ph.D. in Computer Science from Chulalongkorn University in 2004. He now works as a lecturer in the Department of Computer Science at Khon Kaen University, Thailand and joined in research group in Intelligence System and Machine Learning. His research interests are in Neural Networks, Pattern Recognition, Computer Vision, Soft Computing, Fuzzy Systems, Evolutionary Computing and Learning, Optical Computing.