ONLINE LEAST-SQUARES ONE-CLASS SUPPORT VECTOR MACHINE FOR OUTLIER DETECTION IN POWER GRID DATA

Muhammad Sharif Uddin and Anthony Kuh

Department of Electrical Engineering, University of Hawaii, Honolulu, HI 96822
E-mail: uddin@hawaii.edu, kuh@hawaii.edu

ABSTRACT

This paper addresses the problem of outlier detection in the power grid. A sparse online least squares one-class support vector machine classification algorithm is presented to detect outliers in a data stream. An approximate linear dependence criteria is used to obtain a sparse solution by sequentially processing each data point only once, keeping with the requirement of data processing over a data stream. Experiments were conducted on a two dimensional synthetic data set and for bad data in a critical measurement in the IEEE 14 bus system to evaluate the performance of the proposed algorithm.

Index Terms— outlier detection, online learning, sparsification

1. INTRODUCTION

Development of small, low cost, efficient sensor devices and their widespread deployment has resulted in collection of large volume of data that arrives continuously and often needs to be processed as it arrives. Outlier detection is a fundamental step in data quality, management, and analysis tasks. For example, in the power grid system, a large amount of data is collected from sensors and then processed to provide a snapshot of the current system status of the grid to the operator. An outlier in the sensor data could be an indicator of faulty instruments, line faults, or false data injection attacks [1, 2]. Failure to promptly detect outliers in the incoming data may compromise the operator's ability to take remedial actions.

In the power system outlier detection is mostly studied in the context of the state estimator where the most commonly used method is the largest normalized residual (r^N_{max}) test [1]. Other methods of bad data detection have been proposed in recent literature [3, 4]. All of these solutions are performed offline and require multiple runs through the state estimator (SE), making them unsuitable for applications in data stream. In our prior work in [5] we explored an online kernel density based outlier detector with one-hop communication between a node and its neighbor. This method however becomes computationally expensive when the number of neighbors is large.

This work was supported in part by NSF grant ECCS-1310634 and the University of Hawaii REIS project.

In this paper we propose an online outlier detection algorithm based on least squares one-class support vector machine (SVM) classifiers for detecting outliers in a large power grid network. The one-class (OC) SVM is an unsupervised learning method, proposed by Schölkopf et al. in [6] and further advocated by Tax and Duin in [7], to extract regions in the input space where most of the training objects lie. A least squares (LS) version of the OC SVM was proposed by Choi in [8] such that the solution can be obtained by solving a linear system instead of a quadratic programming problem in the standard OC SVM. However this advantage comes at the cost of loss of sparsity of the support vectors (SVs). Several approaches to sparsification of kernel-based solutions have been proposed in the literature [9, 10, 11, 12]. In [13] the authors obtain a sparse set of SVs for LS-OC-SVM classifier for detection of abnormal events in video surveillance. However their approach still requires the storage of all the training objects to obtain the decision hyperplane. In this paper we utilize the approximate linear dependence (ALD) criterion [10] to obtain a sparse representation of the decision hyperplane in LS-OC-SVM. Our approach has lower computational complexity and memory requirement than the non-sparse LS-OC-SVM while still maintaining similar performance. Also, the parameters are updated recursively, thus making this method suitable for application on a data stream.

The remainder of the paper is organized as follows: Section 2 gives a brief introduction of the standard OC-SVM and its least squares version. ALD for sparsification is discussed next in Section 3. Section 4 presents the criteria for outlier classification. The performance of the proposed algorithm is evaluated on synthetic data and in the IEEE 14-bus test system [14] by comparing with the r^N_{max} test in Section 5.

2. LEAST-SQUARES ONE-CLASS SVM

For training data $\underline{x}_1, \ldots, \underline{x}_n \in \mathbb{R}^d$ the one-class (OC) SVM proposed by Schölkopf et al. [6] is the optimization problem

$$\min \frac{1}{2}\|\underline{w}\|^2 - \rho + C\sum_j \xi_j \quad \text{subject to } \underline{w}^T\phi(\underline{x}_j) \geq \rho - \xi_j \text{ and } \xi_j \geq 0, \tag{1}$$

where $\phi(\cdot) : \mathbb{R}^d \to \mathcal{H}$ is the mapping to a high dimensional feature space such that the dot product in the image of ϕ can be computed by evaluating a kernel $k(\underline{x}, \underline{y}) = \phi(\underline{x})^T\phi(\underline{y})$.

The parameter C is predefined and controls the fraction of possible outliers [6] and ξ_j are the non-zero slack variables.

The goal of the standard OC-SVM is to map the data to a high dimensional feature space and obtain a hyperplane $f(\underline{x}) = \underline{w}^T\phi(\underline{x}) - \rho = 0$ with maximal distance $\rho/\left\|\underline{w}\right\|$ from the origin such that most of the data points reside beyond it. $f(\underline{x})$ can be obtained by solving the dual problem [6]

$$\begin{aligned} &\min \frac{1}{2}\textstyle\sum_{i,j}\alpha_i\alpha_j k(\underline{x}_i,\underline{x}_j) \\ &\text{s.t. } 0 \le \alpha_j \le C \text{ and } \textstyle\sum \alpha_j = 1, \end{aligned} \tag{2}$$

where α_j are the Lagrangian multipliers. The solution in the primal space is then given by

$$\underline{w} = \textstyle\sum \alpha_j \phi(\underline{x}_j). \tag{3}$$

The least squares (LS) version of the standard OC-SVM was proposed by Choi in [8] by using a quadratic loss function and equality constraint. The hyperplane obtained by LS-OC-SVM is one such that the distances to most of the training points are minimized in a regularized least squares sense. Thus for any new data, the distance to the hyperplane may be used to measure how well it resembles the training set. The optimization problem for LS-OC-SVM is formulated as

$$\begin{aligned} &\min J = \min \frac{1}{2}\left\|\underline{w}\right\|^2 - \rho + \frac{C}{2}\left\|\underline{\xi}\right\|^2, \\ &\text{subject to } \underline{w} = \mathbf{\Phi}\underline{\alpha} \text{ and } \underline{\xi} = \underline{1}\rho - \mathbf{\Phi}^T\underline{w}, \end{aligned} \tag{4}$$

where the matrix $\mathbf{\Phi} = [\phi(\underline{x}_1), \ldots, \phi(\underline{x}_n)]$ and the column vector $\underline{1}$ consists of all ones. Substituting $\underline{w}$ and $\underline{\xi}$ into (4)

$$J = \frac{1}{2}\underline{\alpha}^T\mathbf{K}\underline{\alpha} - \rho + \frac{C}{2}\left\|\underline{1}\rho - \mathbf{K}\underline{\alpha}\right\|^2, \tag{5}$$

where $\mathbf{K}$ denotes the kernel matrix with entries $\mathbf{K}(i,j) = k(\underline{x}_i,\underline{x}_j) = \phi(\underline{x}_i)^T\phi(\underline{x}_j)$. Taking derivative of (5) with respect to $\underline{\alpha}$ and ρ and setting equal to zero yields:

$$\begin{bmatrix} \underline{1}^T\underline{1} & -\underline{1}^T\mathbf{K} \\ -\underline{1} & \mathbf{K} + \mathbf{I}/C \end{bmatrix} \begin{bmatrix} \rho \\ \underline{\alpha} \end{bmatrix} = \begin{bmatrix} 1/C \\ \underline{0}, \end{bmatrix} \tag{6}$$

where $\underline{0}$ is a column vector containing all zeros and $\mathbf{I}$ is the identity matrix. Applying block matrix inversion lemma [15],

$$\rho = \left(\underline{1}^T(\mathbf{K} + \mathbf{I}/C)^{-1}\underline{1}\right)^{-1}, \tag{7}$$

$$\underline{\alpha} = \left(\underline{1}^T(\mathbf{K} + \mathbf{I}/C)^{-1}\underline{1}\right)^{-1}\left((\mathbf{K} + \mathbf{I}/C)^{-1}\underline{1}\right). \tag{8}$$

The hyperplane is then given by $f(\underline{x}) = \underline{\alpha}^T\underline{k} - \rho = 0$, where $\underline{k}$ denotes a vector with entries $k(\underline{x}_j,\underline{x})$, $j = 1,\ldots,n$.

3. SPARSE ONLINE LS ONE-CLASS SVM

In an online application a sparse solution for the LS-OC-SVM is desirable so that instead of storing the entire history of training set, the solution can be stored in a compact form. However the LS-OC-SVM in Section 2 does not introduce sparsity by itself due to the quadratic loss function in the objective function (4). Several approaches to sparsification of kernel-based solutions have been proposed in the literature [9, 10, 11, 12]. In this paper we utilize the approximate linear dependence (ALD) criterion proposed by Engel et al. in [10] to induce sparsity in the LS-OC-SVM solution.

In an online outlier detection scheme we sequentially process a stream of incoming data points. As more and more data become available the memory and processing requirement increases indefinitely. We adopt a dictionary for the sparsity requirement [10]. Let the dictionary at time step $n-1$ be $\mathcal{X}_{\mathcal{D}}^{(n-1)} = \{\underline{x}_{D,1},\ldots,\underline{x}_{D,m}\}$ where $m < n-1$; $\underline{x}_{D,j}$ are the support vectors (SVs) for the LS-OC-SVM. When a new data point $\underline{x}_n$ arrives we compute the following ALD cost [10]:

$$\delta = \min_{\underline{\beta}} \left\|\textstyle\sum_{j=1}^{m}\beta_j\phi(\underline{x}_{D,j}) - \phi(\underline{x}_n)\right\|^2, \tag{9}$$

which is the distance of $\underline{x}_n$ to the linear span of the dictionary in the feature space. By expanding the inner product and substituting $\phi(\underline{x})^T\phi(\underline{x}') = k(\underline{x},\underline{x}')$, we can rewrite (9) as

$$\delta = \min_{\underline{\beta}}\{\underline{\beta}^T\mathbf{K}_{\mathcal{D},(n-1)}\underline{\beta} - 2\beta^T\underline{k}_{n-1} + k_n\}, \tag{10}$$

where $[\mathbf{K}_{\mathcal{D},(n-1)}]_{i,j} = k(\underline{x}_{D,i},\underline{x}_{D,j})$; $[\underline{k}_{n-1}]_i = k(\underline{x}_{D,i},\underline{x}_n)$ and $k_n = k(\underline{x}_n,\underline{x}_n)$. Solving (10) yields

$$\underline{\beta}_n = \mathbf{K}_{\mathcal{D},(n-1)}^{-1}\underline{k}_{n-1}, \tag{11}$$

$$\delta_n = k_{nn} - \underline{k}_{n-1}^T\mathbf{K}_{\mathcal{D},(n-1)}^{-1}\underline{k}_{n-1}. \tag{12}$$

If δ_n is less than an accuracy parameter γ, then $\phi(\underline{x}_n)$ can be approximated by some linear combination of the present dictionary members within a squared error γ. Defining $\mathbf{\Phi}_n = [\phi(\underline{x}_1),\ldots,\phi(\underline{x}_n)]$ and $\mathbf{\Phi}_{\mathcal{D},(n)} = [\phi(\underline{x}_{D,1}),\ldots,\phi(\underline{x}_{D,m})]$, we can then write $\mathbf{\Phi}_n \approx \mathbf{\Phi}_{\mathcal{D},(n)}\mathbf{B}_n^T$, where $[\mathbf{B}_n]_{i,j} = \beta_{i,j}$, for $i,j = 1,\ldots,n$. Then from (3)

$$\underline{w}_n = \mathbf{\Phi}_n\underline{\alpha}_n \approx \mathbf{\Phi}_{\mathcal{D},(n)}\mathbf{B}_n^T\underline{\alpha}_n = \mathbf{\Phi}_{\mathcal{D},(n)}\hat{\underline{\alpha}}_n = \hat{\underline{w}}_n, \tag{13}$$

where $\hat{\underline{\alpha}}_n = \mathbf{B}_n^T\underline{\alpha}_n$ is the $m \times 1$ approximation of the $n \times 1$ vector $\underline{\alpha}_n$. We can now rewrite optimization problem (4) as an online optimization problem as

$$\begin{aligned} &\min\ J_n = \min_{\hat{\underline{w}}_n,\rho} \frac{1}{2}\left\|\hat{\underline{w}}_n\right\|^2 - \rho_n + \frac{C}{2}\left\|\underline{\xi}_n\right\|^2 \\ &\text{s. t. } \hat{\underline{w}}_n = \mathbf{\Phi}_{\mathcal{D},(n)}\hat{\underline{\alpha}}_n, \text{ and } \underline{\xi}_n = \underline{1}\rho_n - \mathbf{\Phi}_n^T\hat{\underline{w}}_n. \end{aligned} \tag{14}$$

Substituting $\hat{\underline{w}}_n$ and $\underline{\xi}_n$ into (14), we have

$$J_n = \frac{1}{2}\hat{\underline{\alpha}}_n^T\mathbf{K}_{\mathcal{D},(n)}\hat{\underline{\alpha}}_n - \rho_n + \frac{C}{2}\left\|\underline{1}\rho_n - \mathbf{B}_n\mathbf{K}_{\mathcal{D},(n)}\hat{\underline{\alpha}}_n\right\|^2. \tag{15}$$

Setting the derivatives of (15) with respect to $\hat{\underline{\alpha}}_n$ and ρ_n to zero, we obtain the following set of linear equations:

$$\begin{bmatrix} \underline{1}^T\underline{1} & -\underline{1}^T\mathbf{B}_n\mathbf{K}_{\mathcal{D},(n)} \\ -\mathbf{B}_n^T\underline{1} & \mathbf{I}/C + \mathbf{B}_n^T\mathbf{B}_n\mathbf{K}_{\mathcal{D},(n)} \end{bmatrix} \begin{bmatrix} \rho_n \\ \hat{\underline{\alpha}}_n \end{bmatrix} = \begin{bmatrix} 1/C \\ \underline{0} \end{bmatrix} \tag{16}$$

Defining $\mathbf{P}_n = \mathbf{I}/C + \mathbf{B}_n^T\mathbf{B}_n\mathbf{K}_{\mathcal{D},(n)}$ and applying block matrix inversion lemma [15], we can compute ρ_n and $\hat{\underline{\alpha}}_n$ as:

$$\rho_n = \left(C\underline{1}^T\left(\mathbf{I} - \mathbf{B}_n\mathbf{K}_{\mathcal{D},(n)}\mathbf{P}_n^{-1}\mathbf{B}_n^T\right)\underline{1}\right)^{-1}, \tag{17}$$

$$\hat{\underline{\alpha}}_n = \mathbf{P}_n^{-1}\mathbf{B}_n^T\underline{1}\left(C\underline{1}^T\left(\mathbf{I} - \mathbf{B}_n\mathbf{K}_{\mathcal{D},(n)}\mathbf{P}_n^{-1}\mathbf{B}_n^T\right)\underline{1}\right)^{-1}. \tag{18}$$

Note that when there is no sparsity, i.e., $\mathbf{B}_n = \mathbf{I}$, (17) and (18) reduces to (7) and (8), respectively.

In the online data stream application, ρ and $\hat{\underline{\alpha}}_n$ need to be updated every time a new data point becomes available. At time step n, one of the following two cases can happen:

Case 1: $\delta_n \le \gamma$, i.e., $\phi(\underline{x}_n)$ is approximately linearly dependent on $\mathcal{X}_{\mathcal{D}}^{(n-1)}$ and therefore, not included in the dictionary, i.e., $\mathcal{X}_{\mathcal{D}}^{(n)} = \mathcal{X}_{\mathcal{D}}^{(n-1)}$. Here the matrix $\mathbf{K}_{\mathcal{D}}$ remains un-

changed, only $\mathbf{B}$ is updated as $\mathbf{B}_n = [\mathbf{B}_{n-1}^T, \underline{\beta}_n]^T$. Then we have $\mathbf{P}_n = \mathbf{P}_{n-1} + \underline{\beta}_n \underline{k}_{n-1}^T$. By applying the Sherman-Morrison formula [15] we can compute $\mathbf{P}_n^{-1}$ recursively as

$$\mathbf{P}_n^{-1} = \mathbf{P}_{n-1}^{-1} - \frac{\mathbf{P}_{n-1}^{-1}\underline{\beta}_n \underline{k}_{n-1}^T \mathbf{P}_{n-1}^{-1}}{1 + \underline{k}_{n-1}^T \mathbf{P}_{n-1}^{-1}\underline{\beta}_n}. \quad (19)$$

Then ρ_n and $\hat{\underline{\alpha}}_n$ are updated using (17)-(18).

Case 2: $\delta_n > \gamma$. In this case $\underline{x}_n$ is added to the dictionary, i.e., $\mathcal{X}_{\mathcal{D}}^{(n)} = \mathcal{X}_{\mathcal{D}}^{(n-1)} \cup \{\underline{x}_n\}$ and $m = m + 1$. Accordingly both $\mathbf{K}_{\mathcal{D}}$ and $\mathbf{B}$ are updated. Then we have

$$\mathbf{B}_n = \begin{bmatrix} \mathbf{B}_{n-1} & \underline{0} \\ \underline{0}^T & 1 \end{bmatrix}. \quad (20)$$

The recursive formula for $\mathbf{K}_{\mathcal{D},(n)}^{-1}$ and $\mathbf{P}_n^{-1}$ are given by:

$$\begin{aligned} \mathbf{K}_{\mathcal{D},(n)}^{-1} &= \begin{bmatrix} \mathbf{K}_{\mathcal{D},(n-1)} & \underline{k}_{n-1} \\ \underline{k}_{n-1}^T & k_n \end{bmatrix}^{-1} \\ &= \begin{bmatrix} \mathbf{K}_{\mathcal{D},(n-1)}^{-1} & \underline{0} \\ \underline{0}^T & 0 \end{bmatrix} + \frac{1}{\delta_n} \begin{bmatrix} \underline{\beta}_n \\ -1 \end{bmatrix} \begin{bmatrix} \underline{\beta}_n \\ -1 \end{bmatrix}^T \end{aligned} \quad (21)$$

$$\begin{aligned} \mathbf{P}_n^{-1} = \begin{bmatrix} \mathbf{P}_{n-1} & \mathbf{B}_{n-1}^T \mathbf{B}_{n-1} \underline{k}_{n-1} \\ \underline{k}_{n-1}^T & k_n + 1/C \end{bmatrix}^{-1} = \begin{bmatrix} \mathbf{P}_{n-1}^{-1} & \underline{0} \\ \underline{0}^T & 0 \end{bmatrix} + \\ + \frac{\begin{bmatrix} -\mathbf{P}_{n-1}^{-1}\mathbf{B}_{n-1}^T \mathbf{B}_{n-1}\underline{k}_{n-1} \\ 1 \end{bmatrix} \begin{bmatrix} -\mathbf{P}_{n-1}^{-T}\underline{k}_{n-1} \\ 1 \end{bmatrix}^T}{k_n + 1/C - \underline{k}_{n-1}^T \mathbf{P}_{n-1}^{-1} \mathbf{B}_{n-1}^T \mathbf{B}_{n-1} \underline{k}_{n-1}}. \end{aligned} \quad (22)$$

Finally we compute ρ_n and $\hat{\underline{\alpha}}_n$ using (17)-(18).

4. OUTLIER DETECTION

An outlier is a data point that appears different from the other data points within the sample space. In an online application scenario when a new data point becomes available, three possible cases can happen:

- it closely resembles the learned input distribution,
- it represents a change in the underlying distribution of the input data,
- it differs significantly from the learned input distribution and indicates an outlier.

The sparse online LS-OC-SVM finds the hyperplane that minimizes the squared distances to the images of the training points in the feature space. Therefore, the distance from the hyperplane can be used as a measure of resemblance between a new data point and the training set. For new data $\underline{x}_n$, the distance of $\phi(\underline{x}_n)$ from the hyperplane can be computed as

$$d(\underline{x}_n) = \left| \hat{\underline{\alpha}}_{n-1}^T \underline{k}_{n-1} - \rho_n \right| \Big/ \sqrt{\hat{\underline{\alpha}}_{n-1}^T \mathbf{K}_{\mathcal{D},(n-1)} \hat{\underline{\alpha}}_{n-1}}. \quad (23)$$

If $d(\underline{x}_n)$ is greater than a threshold d_{th} then we say that $\underline{x}_n$ differs significantly from the training objects seen so far, and is a possible outlier. However whether this data point indicates a change in the underlying distribution or its a true outlier still needs to be determined.

In the proposed sparse online LS-OC-SVM our goal is to build a diverse support vector dictionary to approximate the input space while inducing sparsity. To achieve that we utilized ALD measure δ_n in Section 3. When an outlier data arrives the machine can detect a possible outlier using the distance measure. However it is difficult to distinguish between a change in underlying distribution and a true outlier. In both cases the ALD measure will be higher that the threshold. But if a true outlier is added into the SV dictionary, it will have detrimental effect on the estimation of the decision hyperplane. On the other hand if a non-outlier but differing data point were to be excluded from the SV dictionary we would fail to construct a diverse enough dictionary.

To overcome this we introduce another threshold γ_2 for the ALD measure assuming the change in the underlying distribution is gradual. After comparing the distance threshold, before a data point is added as an SV we have three cases:

- $\delta_n \leq \gamma_1$: very similar to the dictionary; only hyperplane updated.
- $\gamma_1 < \delta_n \leq \gamma_2$: somewhat similar; both dictionary and hyperplane updated.
- $\delta > \gamma_2$: very dissimilar; discard without updating either dictionary or hyperplane.

Similar approach has been used in [12]. In an online application it is however necessary to use a time varying γ_2, i.e., γ_2 should have a large value while the size of the data set observed is still small, and then gradually decrease. This helps maintain a relatively larger dictionary at the beginning of the learning algorithm.

5. SIMULATION RESULTS

To evaluate the performance of our proposed method we first applied it to a synthetic data set and then implemented the outlier detection in the power grid data, using a Gaussian kernel $k(\underline{x}_1, \underline{x}_2) = \exp(-\|\underline{x}_1 - \underline{x}_2\|/\sigma^2)$.

5.1. Decision boundaries on synthetic data

First we evaluate the performance of sparse online LS one-class SVM on the square-noise data set [16]. Fig. 1 shows the decision boundaries obtained using OC-SVM, LS-OC-SVM and the proposed sparse online LS-OC-SVM methods. The OC-SVM decision boundary was obtained using LIBSVM-3.20 library [17] with parameters $\sigma^2 = 0.2$ and $C = 1/(0.12n)$. For both batch and online LS-OC-SVM we set $\sigma^2 = 0.045$ and $C = 1$. For the online version we set $\gamma_1 = 0.01$ and after observing 50 data points $\gamma_2 = \max(\exp(-m/100), 2\gamma_1)$ is set.

From fig. 1 we observe that the SVs chosen by the standard OC-SVM algorithm are in fact outliers. This occurs because the algorithm tries to find the hyperplane such that most data points lie beyond it. Thus the data points located at the edge of the data set are chosen as support vectors. On the other hand, the offline LS-OC-SVM obtains a decision boundary that is more representative of the data. The drawback is that now every data point acts as an SV. Finally the decision region obtained by the proposed method is smoother and does not pick any outlier as an SV.

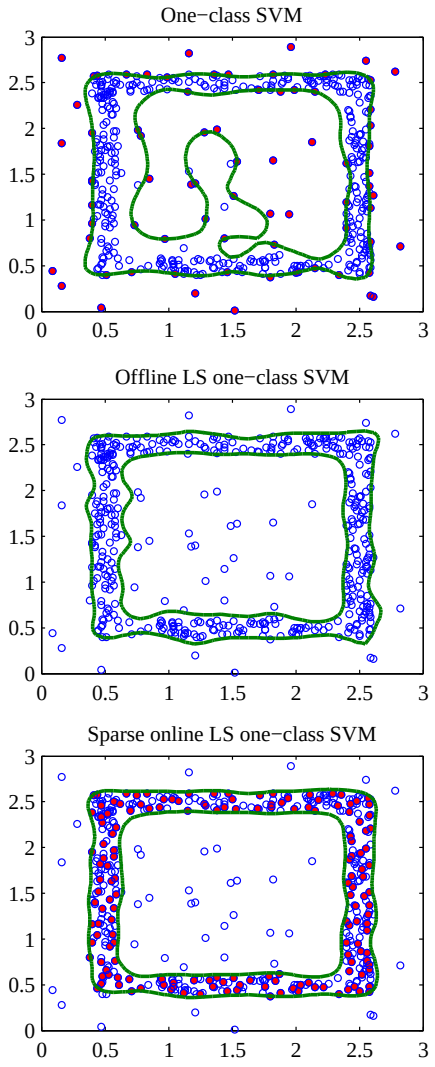

Fig. 1. Comparison of decision boundaries obtained by OC-SVM ($\sigma = 0.2$, $C = 1/(0.12n)$), offline LS-OC-SVM ($\sigma^2 = 0.045$, $C = 1$) and sparse online LS-OC-SVM ($\sigma^2 = 0.045$, $C = 1, \gamma_1 = 0.01$). The solid red circles represent the support vectors in OC-SVM and sparse online LS-OC-SVM.

5.2. Outlier detection in IEEE 14-bus system

To evaluate the performance of the proposed outlier detection algorithm we implemented the algorithm in the IEEE 14-bus test system [14]. The network diagram of the test system is shown in Fig. 2. We compare the performance of our algorithm for detection of bad data in *critical* measurement with one of the most commonly used bad data identification method in the power system, the largest normalized residual test (r^N_{max} test). A *critical* measurement is the one whose elimination from the measurement set renders the system *unobservable*. The r^N_{max} test fails to detect bad data in this type of measurements [1]. Using the proposed method we are able to detect bad data in critical measurements.

We used the state estimator in the MATPOWER software [18] as a tool for the r^N_{max} test. The measurement configuration used for the simulations is given in Table 1.

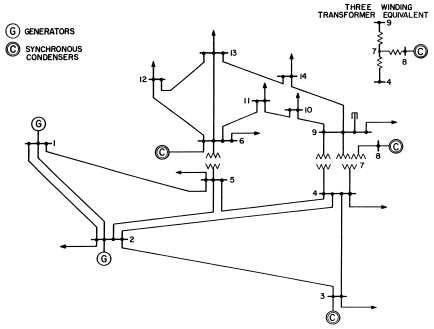

Fig. 2. IEEE 14 Bus test system [14]

Table 1. Measurement configuration for IEEE 14-bus system

Power injection	At buses 1, 2, 3, 8
Power flow	At branches 1-2, 1-5, 2-4, 2-5, 4-5, 4-7, 4-9, 5-6, 6-11, 6-13, 9-10, 9-14, 12-13
Voltage magnitude	At buses 3, 6, 8, 10, 14

Under the measurement configuration described in Table 1, P_{4-7} is a critical measurement. The true value for this flow is $P_{4-7} = 0.2285 p.u.$ The residual for this measurement will always be zero and therefore, any bad data in this measurement cannot be identified by the r^N_{max} test. To apply our online outlier detection technique we create a stream of 1000 samples by adding Gaussian noise $\mathcal{N}(0, 0.0004)$ to the true value. We then randomly inject five bad data points in this data stream. For outlier detection we considered the vector $\underline{x} = [P_{4-7}, P_{4-2}, P_{4-5}, P_{4-9}]^T$ which contains the concerned measurement and the available measurements from the neighboring buses. The parameters used for online LS-OC-SVM are $\sigma^2 = 0.002$, $\gamma_1 = 0.0001$, $\gamma_2 = \max(\exp(-m/50), 3\gamma_1)$. The values of the bad measurements at P_{4-7} and the performance of the different bad data detection algorithm is shown in Table 2. From the table we see that the proposed online outlier detection technique was able to identify every bad measurement *before* the data was fed to the state estimator.

Table 2. Detection of bad data in a critical measurement P_{4-7}

Bad data in P_{4-7}	r^N_{max} test	Sparse LS-OC-SVM
$0.3152 p.u.$	Not detected	Detected
$0.0660 p.u.$	Not detected	Detected
$0.3828 p.u.$	Not detected	Detected
$0.1477 p.u.$	Not detected	Detected
$0.3315 p.u.$	Not detected	Detected

6. SUMMARY

This paper proposes a sparse online LS-OC-SVM for outlier detection in the power grid. The algorithm identifies outliers in an online fashion *before* the state estimation process, without relying on the *observability* of the grid. From simulations performed on the IEEE 14-bus system we observed that the proposed method outperforms the widely used r^N_{max} test.

7. REFERENCES

[1] Ali Abur and Antonio Gómez Expósito, *Power System State Estimation: Theory and Implementation*, Marcel Dekker, Inc, New York, NY, 204.

[2] Yao Liu, Peng Ning, and Michael K. Reiter, "False data injection attacks against state estimation in electric power grids," in *Proc. 16th ACM Conf. Computer and Communications Security*, 2009, pp. 21–32.

[3] Hee-Myung Jeong, June Ho Park, and Hwa-Seok Lee, "Multiple bad data identification using binary particle swarm optimization," *Journal of International Council on Electrical Engineering*, vol. 1, no. 3, pp. 269–273, 2011.

[4] E.N. Asada, A.V. Garcia, and R. Romero, "Identifying multiple interacting bad data in power system state estimation," in *IEEE PES General Meeting, 2005*, June 2005, pp. 571–577.

[5] M. S. Uddin, A. Kuh, Y. Weng, and M. D. Ilić, "Online bad data detection using kernel density estimation," in *PES General Meeting, 2015 IEEE*, Denver, CO, July 2015.

[6] B. Schölkopf, J. C. Platt, J. C. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the support of a high-dimensional distribution," *Neural Comput.*, vol. 13, no. 7, pp. 1443–1471, 2001.

[7] D. M.J. Tax and R. P.W. Duin, "Support vector data description," *Machine Learning*, vol. 54, no. 1, pp. 45–66, 2004.

[8] Young-Sik Choi, "Least squares one-class support vector machine," *Pattern Recognition Letters*, vol. 30, no. 13, pp. 1236 – 1240, 2009.

[9] J. A. K. Suykens, T. Van Gestel, J. De Brabanter, B. De Moor, and J. Vandewalle, *Least Squares Support Vector Machines*, World Scientific, River Edge, NJ, 2002.

[10] Y. Engel, S. Mannor, and R. Meir, "The kernel recursive least-squares algorithm," *IEEE Transactions on Signal Processing*, vol. 52, no. 8, pp. 2275–2285, Aug 2004.

[11] C. Richard, J.C.M. Bermudez, and P. Honeine, "Online prediction of time series data with kernels," *IEEE Trans. Signal Processing*, vol. 57, no. 3, pp. 1058–1067, 2009.

[12] W. Liu, I. Park, and J.C. Principe, "An information theoretic approach of designing sparse kernel adaptive filters," *IEEE Trans. Neural Networks*, vol. 20, no. 12, pp. 1950–1961, Dec 2009.

[13] T. Wang, J. Chen, Y. Zhou, and H. Snoussi, "Online least squares one-class support vector machines-based abnormal visual event detection," *Sensors*, vol. 13, no. 12, pp. 17130–17155, 2013.

[14] "Power system test case archive," Available at http://www.ee.washington.edu/research/pstca/.

[15] F. Zhang, Ed., *The Schur Complement and Its Application*, vol. 4 of *Numerical Methods and Algorithms*, Springer US, 2005.

[16] H. Hoffmann, "Kernel PCA for novelty detection," *Pattern Recognition*, vol. 40, no. 3, pp. 863 – 874, 2007.

[17] Chih-Chung Chang and Chih-Jen Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, pp. 27:1–27:27, 2011, Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.

[18] Ray Daniel Zimmerman, Carlos Edmundo Murillo-Sánchez, and Robert John Thomas, "MATPOWER: Steady-state operations, planning, and analysis tools for power systems research and education," *IEEE Trans. Power Syst.*, vol. 26, no. 1, pp. 12–19, Feb. 2011.