Data Clustering Method Using a Modified Gaussian Kernel Metric and Kernel PCA

Hansung Lee, Jang-Hee Yoo, and Daihee Park

Most hyper-ellipsoidal clustering (HEC) approaches use the Mahalanobis distance as a distance metric. It has been proven that HEC, under this condition, cannot be realized since the cost function of partitional clustering is a constant. We demonstrate that HEC with a modified Gaussian kernel metric can be interpreted as a problem of finding condensed ellipsoidal clusters (with respect to the volumes and densities of the clusters) and propose a practical HEC algorithm that is able to efficiently handle clusters that are ellipsoidal in shape and that are of different size and density. We then try to refine the HEC algorithm by utilizing ellipsoids defined on the kernel feature space to deal with more complex-shaped clusters. The proposed methods lead to a significant improvement in the clustering results over K-means algorithm, fuzzy C-means algorithm, GMM-EM algorithm, and HEC algorithm based on minimum-volume ellipsoids using Mahalanobis distance.

Keywords: Data clustering, hyper-ellipsoidal clustering, minimum-volume ellipsoids, kernel PCA, Gaussian kernel.

I. Introduction

With the growing popularity of clustering techniques in machine learning, pattern recognition, computer vision, and data mining a number of clustering algorithms have been extensively studied over several decades. An optimal clustering algorithm that can deal with all cases does not currently exist. Despite numerous research efforts, data clustering remains a difficult and challenging problem [1]–[2].

The ideal objective function for clustering, aims to maximize the similarity of data points within clusters and to minimize the similarity of data points between clusters. It is difficult to find a feasible object function of clustering that meets the ideal criteria [1]–[2]. The most commonly used criterion for clustering is to minimize the sum of the Euclidean distances of data points within a cluster, for example, as in the case of K-means clustering. The clustering algorithms based on Euclidean metric have a tendency to divide the data points into clusters of equal size, equal density, and spherical shapes. This tendency has the drawback of splitting large and elongated clusters under certain circumstances, and the cluster allocation may change significantly with linear transformations of the data space. Real-world data often exhibits a mixture of Gaussian distribution (that is, ellipsoidal or complex shapes).

To resolve the aforementioned problems, numerous hyper-ellipsoidal clustering (HEC) algorithms have been proposed [3]–[11]. These approaches generally employ the Mahalanobis distance as a distance metric to create ellipsoidal clusters, however, there are several problems in using it. Firstly, it is difficult to directly compute the covariance matrix in the Mahalanobis distance because of the expensive computational complexity, and secondly, it may be singular when clusters include a small number of data points. To deal with these

Manuscript received June 6, 2013; revised Sept. 19, 2013; accepted Nov. 25, 2013.

This work was supported by the IT R&D program of MOTIE/KEIT, Korea (10039149, Development of Basic Technology of Human Identification and Retrieval at a Distance for Active Video Surveillance Service with Real-time Awareness of Safety Threats).

Hansung Lee (phone: +82 42 860 6037, mohan@etri.re.kr), Jang-Hee Yoo (jhy@etri.re.kr) are with the SW · Content Research Laboratory, ETRI, Daejeon, Rep. of Korea.

Daihee Park (corresponding author, dhpark@korea.ac.kr) is with the Department of Computer and Information Science, Korea University, Sejong, Rep. of Korea.

difficulties, HEC algorithms based on the modified Mahalanobis distance and with a pseudo-covariance matrix have been proposed [3]–[5]. The time complexity of these methods is still expensive. On the other hand, HEC approaches that approximate the volume of clusters—that is, find minimum-volume ellipsoids (MVE) instead of the covariance matrix—have been presented to alleviate the problems in computing the covariance matrix [8]–[11]. Most of the related methods use the Mahalanobis distance as a distance metric. However, it has been proven that an ellipsoidal clustering cannot be obtained and that the cost function of partitional clustering cannot limit the size of the clusters and relations among the clusters if the Mahalanobis distance is directly applied to the clustering [7]. Furthermore, the HEC algorithms based on MVE work well on clusters of equal density, because these methods do not consider the density of each cluster.

The primary goal of this paper is to realize an ellipsoidal clustering and to provide practical implementations of an HEC algorithm. First, we recast a partitional clustering algorithm with the modified Gaussian kernel metric, as a problem of finding condensed clusters with respect to the volumes and densities of clusters. We then propose practical HEC algorithms based on the approximation of MVE with the modified Gaussian kernel metric, which are able to efficiently deal with clusters that are ellipsoidal in shape and that are of different size and density. We next attempt to enhance the capability of the HEC algorithms by mapping ellipsoids on the kernel feature space to handle nonlinear and elongated clusters. Experimental results show that the proposed methods lead to a significant improvement in the clustering results over K-means algorithm, fuzzy C-means algorithm, GMM-EM algorithm, and MVE-based clustering algorithms with Mahalanobis distance.

The rest of this paper is organized as follows. In section II, we provide the problem definition and theoretical explanation. In section III, we then propose the HEC algorithm based on the approximation of MVE with the modified Gaussian kernel metric and its extended algorithm on the kernel feature map, which is able to find more complex cluster shapes. Experimental results and discussions are then provided in section IV. Finally, some concluding remarks are given in section V.

II. Problem Definition

Given a set of n data points in a d-dimensional space, $\boldsymbol{x} = \{\boldsymbol{x}_i \in \mathbb{R}^d\}_{i=1}^n$, the objective of a partitional clustering is then to determine an assignment of the partition matrix, $\mathbf{P} = \{P_{ik} \mid P_{ik} \in \{0,1\};\ i = 1,2,...,n;\ k = 1,2,...,C\}$, such that the cost function $E_C(\mathbf{P})$ is minimized as

$$\mathbf{P} = \arg(\mathbf{P}) \min E_C(\mathbf{P}), \tag{1}$$

where C is the number of clusters generally determined by the user. If data point $\boldsymbol{x}_i$ is assigned to the kth cluster $P_{ik} = 1$; otherwise $P_{ik} = 0$. For a unique assignment, $\Sigma_{k=1}^{C} P_{ik} = 1$ for $i = 1, 2, \ldots, n$ should be satisfied. The cost function for the partitional clustering can be defined as

$$E_C(\mathbf{P}) = \sum_{i=1}^{n} \sum_{k=1}^{C} P_{ik} D(\boldsymbol{x}_i, \boldsymbol{m}_k), \tag{2}$$

where $D(\boldsymbol{x}_i, \boldsymbol{m}_k)$ is a distance metric between the input pattern $\boldsymbol{x}_i$ and the mean vector of kth cluster $\boldsymbol{m}_k$.

To create ellipsoidal clusters, HEC algorithms generally adopt the Mahalanobis distance. However, the cost function of partitional clustering is constant under this condition [7].

For realizing ellipsoidal clustering, we employ the modified Gaussian kernel as the distance metric:

$$\begin{aligned} D(\boldsymbol{x}_i, \boldsymbol{m}_k) = \alpha \cdot (\boldsymbol{x}_i - \boldsymbol{m}_k)^T \mathbf{Q}_k^{-1} (\boldsymbol{x}_i - \boldsymbol{m}_k) \\ + (1-\alpha) \cdot \log\det \mathbf{Q}_k, \end{aligned} \tag{3}$$

where $\boldsymbol{m}_k$ and $\mathbf{Q}_k$ are the mean vector and covariance matrix of the kth cluster, respectively. The variable $\alpha \in [0,1]$ controls the weight of the first and second terms of the modified Gaussian kernel. Note that the first term of (3) represents the Mahalanobis distance and that the second term is proportional to the volume of the kth ellipsoidal cluster, which is represented by the covariance matrix $\mathbf{Q}_k$. The clustering cost function with the modified Gaussian kernel can be written as

$$\begin{aligned} E_C(\mathbf{P}) = \sum_i \sum_k P_{ik} [\alpha \cdot (\boldsymbol{x}_i - \boldsymbol{m}_k)^T \mathbf{Q}_k^{-1} (\boldsymbol{x}_i - \boldsymbol{m}_k) \\ + (1-\alpha) \cdot \log\det \mathbf{Q}_k]. \end{aligned} \tag{4}$$

The necessary condition for the optimality of $E_C(\mathbf{P})$, $\frac{\partial E_C(\mathbf{P})}{\partial \boldsymbol{m}_k^T} = 0$, the centers of the clusters allocated by minimizing the clustering cost function of (4) can be represented as

$$\boldsymbol{m}_k = \frac{\sum_i P_{ik} \boldsymbol{x}_i}{\sum_i P_{ik}}. \tag{5}$$

Lemma 1: Given a set of n data points in a d-dimensional space $\boldsymbol{x} = \{\boldsymbol{x}_i \in \mathbb{R}^d\}_{i=1}^n$, where $\boldsymbol{m}_k$ indicates C means (for k = 1, 2,…, C, with C being the number of clusters) and its covariance matrix $\mathbf{Q}_k = \frac{1}{n-1} \sum_{i=1}^{n} (\boldsymbol{x}_i - \boldsymbol{m}_k)(\boldsymbol{x}_i - \boldsymbol{m}_k)^T$ is invertible, then

$$\begin{aligned} \sum_{i=1}^{n} [\alpha \cdot (\boldsymbol{x}_i - \boldsymbol{m}_k)^T \mathbf{Q}_k^{-1} (\boldsymbol{x}_i - \boldsymbol{m}_k) \\ + (1-\alpha) \cdot \log\det \mathbf{Q}_k] \\ = \alpha \cdot d \cdot (n-1) + n \cdot (1-\alpha) \cdot \log\det \mathbf{Q}_k. \end{aligned} \tag{6}$$

***Proof*:** By Theorem 1 in [7] and Theorem 1 in [4], we have $\sum_{i=1}^{n} \alpha \cdot (\boldsymbol{x}_i - \boldsymbol{m}_k)^T \mathbf{Q}_k^{-1} (\boldsymbol{x}_i - \boldsymbol{m}_k) = \alpha \cdot d \cdot (n-1)$, and

$$\sum_{i=1}^{n}(1-\alpha)\cdot\log\det\mathbf{Q}_k = n\cdot(1-\alpha)\cdot\log\det\mathbf{Q}_k,$$
$$\therefore \sum_{i=1}^{n}[\alpha\cdot(\boldsymbol{x}_i-\boldsymbol{m}_k)^T\mathbf{Q}_k^{-1}(\boldsymbol{x}_i-\boldsymbol{m}_k)+(1-\alpha)\cdot\log\det\mathbf{Q}_k]$$
$$=\alpha\cdot d\cdot(n-1)+n\cdot(1-\alpha)\cdot\log\det\mathbf{Q}_k. \quad \blacksquare$$

Theorem 1: If the modified Gaussian kernel (3) is used as the distance metric in the clustering cost function (2), then

$$E_C(\mathbf{P}) \cong \sum_{k=1}^{C} n_k \log\det\mathbf{Q}_k, \tag{7}$$

where $n_k = \sum_i P_{ik}$ is the number of data points assigned to the *k*th cluster.

***Proof*:** From Lemma 1, we have $\sum_{i=1}^{n} P_{ik} D(\boldsymbol{x}_i, \boldsymbol{m}_k) = \alpha\cdot d\cdot(n_k-1) + n_k\cdot(1-\alpha)\cdot\log\det\mathbf{Q}_k$. The cost function can then be expressed as

$$\begin{aligned} E_C(\mathbf{P}) &= \sum_{i=1}^{n}\sum_{k=1}^{C} P_{ik} D(\boldsymbol{x}_i,\boldsymbol{m}_k) \\ &= \sum_{k=1}^{C}\alpha\cdot d\cdot(n_k-1)+\sum_{k=1}^{C} n_k\cdot(1-\alpha)\cdot\log\det\mathbf{Q}_k \\ &= \alpha\cdot d\cdot(n-C)+(1-\alpha)\sum_{k=1}^{C} n_k\log\det\mathbf{Q}_k, \end{aligned}$$

where α, d, n, and C are all constants with respect to variable k and therefore $E_C(\mathbf{P}) \cong \sum_{k=1}^{C} n_k \log\det\mathbf{Q}_k$. $\blacksquare$

Remarks: Note that the problem of minimizing the cost function (7) is equivalent to minimizing the weighted sum of the cluster volumes $E_C(\mathbf{P}) \propto \sum_{k=1}^{C} w_k Vol(k)$. The weight value of each cluster is defined as the number of data points that belong to it. This can be intuitively interpreted as allocating data points to one of the clusters, such that its compactness and density is maximized. In other words, the problem is one of finding condensed clusters with respect to the volumes and densities of clusters.

III. Ellipsoidal and Complex-Shaped Clustering

In this section, we propose two HEC algorithms that attempt to minimize the weighted sum of the cluster volumes: i) K-HEC—an iterative HEC algorithm using the modified Gaussian kernel and MVE approximation, which allocates the data points into a specified number of clusters of ellipsoidal shape and ii) EK-HEC—an extended iterative HEC algorithm that improves the capability of the K-HEC by utilizing ellipsoids defined on the kernel feature space to deal with more complex-shaped clusters.

1. K-HEC Algorithm

The K-HEC algorithm starts with an initial clustering into C clusters and iteratively finds the improved assignment of the partition matrix, such that the weighted sum of volumes of clusters is decreasing and until there is no further possible improvement. It is a modification of the K-means algorithm, which combines the modified Gaussian kernel and MVE approximation. Finding the optimal minimum-volume covering ellipsoid is the core step of the MVE-based HEC algorithm. Due to the combinatorial nature of the clustering problem, the computational complexity of the MVE is critical. Given a set of n data points in a d-dimensional space $\boldsymbol{x} = \{\boldsymbol{x}_i \in \mathbb{R}^d\}_{i=1}^{n}$, computing the MVE enclosing a set of data points is defined as a maxdet problem [10]–[12], and it is a difficult problem to solve directly.

$$\begin{aligned} &\text{Minimize} && \log\det\mathbf{Q}, \\ &\text{subject to} && \mathbf{Q}=\mathbf{Q}^T>0 \\ &\text{and} && (\boldsymbol{x}_i-\boldsymbol{x}_C)^T\mathbf{Q}^{-1}(\boldsymbol{x}_i-\boldsymbol{x}_C)\le 1, \\ &\text{where} && i=1,\,2,\ldots,\,n. \end{aligned} \tag{8}$$

Herein, we use three practical approximation approaches for finding the MVE. First, we derive the convex optimization problem of (9) from the Löwner-John ellipsoid [12], which can be interpreted geometrically as minimizing the volume of the ellipsoid.

$$\begin{aligned} &\text{Minimize} && \det\mathbf{A}^{-1}, \\ &\text{subject to} && \mathbf{A}>0 \\ &\text{and} && \|\mathbf{A}\boldsymbol{x}_i-\boldsymbol{b}\|\le 1, \\ &\text{where} && i=1,\,2,\ldots,\,n. \end{aligned} \tag{9}$$

Second, we solve the convex optimization problem of (10) by approximating the object function with the sum of the eigenvalues of matrix $\mathbf{Q}$ [9], [12].

$$\begin{aligned} &\text{Minimize} && \text{Trace}\,(\mathbf{Q}), \\ &\text{subject to} && \mathbf{Q}=\mathbf{Q}^T>0 \\ &\text{and} && (\boldsymbol{x}_i-\boldsymbol{x}_C)^T\mathbf{Q}^{-1}(\boldsymbol{x}_i-\boldsymbol{x}_C)\le 1, \\ &\text{where} && i=1,\,2,\ldots,\,n. \end{aligned} \tag{10}$$

Third, we employ a fast-approximation algorithm for finding MVE using Khachiyan's algorithm [13]–[15].

$$\begin{aligned} &\text{MVE} = \{\boldsymbol{x}\in\mathbb{R}^d \mid (\boldsymbol{x}-\boldsymbol{x}_C^*)^T\mathbf{Q}^*(\boldsymbol{x}-\boldsymbol{x}_C^*)\le 1\}, \\ &\text{where}\ \ \mathbf{Q}^* = \frac{1}{d}(\mathbf{P}\mathbf{U}^*\mathbf{P}^T-\mathbf{P}\boldsymbol{u}^*(\mathbf{P}\boldsymbol{u}^*)^T)^{-1},\ \ \boldsymbol{x}_C^* = \mathbf{P}\boldsymbol{u}^*, \end{aligned} \tag{11}$$

$\mathbf{P}=[\boldsymbol{q}_1,\boldsymbol{q}_2,\ldots,\boldsymbol{q}_n]\in\mathbb{R}^{d\times n}$, $\boldsymbol{q}_i^T=[\boldsymbol{x}_i^T\ 1]$; $i=1,\,2,\ldots,\,n$, u is the dual variable, and $\mathbf{U}$ = diag($\boldsymbol{u}$). The K-HEC algorithm is described as follows.

Algorithm 1. K-HEC algorithm
Input: Given a set of n data points in a d-dimensional space, $\boldsymbol{x} = \{\boldsymbol{x}_i \in \mathbb{R}^d\}_{i=1}^{n}$.
Output: The partition matrix $\mathbf{P}$.
Step 1: Fix the number of clusters, C. Randomly select C data points, and set these as the centers of clusters, $\boldsymbol{m}_k$.

Step 2: Initially determine the assignment of the partition matrix **P**, using the Euclidean distance measure with $\boldsymbol{m}_k$.

$$P_{ik} = 1, \text{ if } D_{\text{Euc}}(\boldsymbol{x}_i, \boldsymbol{m}_k) < D_{\text{Euc}}(\boldsymbol{x}_i, \boldsymbol{m}_j),$$
$$j = 1, 2, \ldots, C \text{ and } j \neq k.$$
$$P_{ik} = 0, \text{ otherwise.}$$

Step 3: Compute new centers of clusters and the number of data points belonging to the clusters.

$$\boldsymbol{m}_k = \frac{\sum_i P_{ik}\boldsymbol{x}_i}{\sum_i P_{ik}}, \quad n_k = \sum_i P_{ik}.$$

Step 4: Compute the pseudo-covariance matrix $\mathbf{Q}_k$, using the MVE approximation algorithm.

Step 5: Determine the new assignment of the partition matrix **P**, using the modified Gaussian kernel measure (3) with $\boldsymbol{m}_k$ and $\mathbf{Q}_k$.

$$P_{ik} = 1, \text{ if } D_{\text{MGK}}(\boldsymbol{x}_i, \boldsymbol{m}_k; \mathbf{Q}_k) < D_{\text{MGK}}(\boldsymbol{x}_i, \boldsymbol{m}_j; \mathbf{Q}_j),$$
$$j = 1, 2, \ldots, C \text{ and } j \neq k.$$
$$P_{ik} = 0, \text{ otherwise.}$$

Step 6: If there are no changes to the partition matrix **P**, then STOP. Otherwise, repeat Step 3 through Step 5.

2. EK-HEC Algorithm

Most partitional clustering algorithms do not work well on a dataset with a nonlinear and elongated structure. To overcome this difficulty, spectral clustering algorithms have been proposed [16], and there is ongoing research into kernel methods [17]–[19]. To perform clustering on a dataset with more complex-shaped clusters—that is, a nonlinear and elongated structure—we employ one of the kernel tricks—a kernel principal component analysis (KPCA). Using integral operator kernel functions, we can efficiently compute the principal components in high-dimensional feature spaces, which are related to the input space based on a nonlinear map [17], [18]. The KPCA can be summarized as follows [17], [18]: Given a set of n data points mapped onto a feature space $\Phi(\boldsymbol{x}) = \{\Phi(\boldsymbol{x}_i) \in \mathcal{F}\}_{i=1}^n$, the covariance matrix in a kernel feature space is defined as

$$\mathbf{C}^{\Phi} = \frac{1}{n}\sum_{j=1}^{n} \Phi(\boldsymbol{x}_j)\Phi(\boldsymbol{x}_j)^T. \tag{12}$$

We then have

$$\lambda(\Phi(\boldsymbol{x}_k) \cdot \boldsymbol{V}) = (\Phi(\boldsymbol{x}_k) \cdot \mathbf{C}^{\Phi}\boldsymbol{V}); \; k = 1, 2, \ldots, n, \tag{13}$$

where $\lambda \geq 0$ is the eigenvalue, and $\boldsymbol{V}$ is the eigenvector $\boldsymbol{V} = \sum_{i=1}^n \alpha_i \Phi(\boldsymbol{x}_i)$. The eigenvector $\boldsymbol{V}$ lies in the span of $\Phi(\boldsymbol{x}) = \{\Phi(\boldsymbol{x}_i) \in \mathcal{F}\}_{i=1}^n$.

By defining the $n \times n$ kernel matrix **K**, as $K_{ij} = (\Phi(\boldsymbol{x}_i) \cdot \Phi(\boldsymbol{x}_j))$, we can obtain

$$n\lambda\mathbf{K}\boldsymbol{\alpha} = \mathbf{K}^2\boldsymbol{\alpha}, \tag{14}$$

where $\boldsymbol{\alpha}$ denotes the column vector with entries α_1, $\alpha_2, \cdots, \alpha_n$. To find the solutions of (14), we solve the eigenvalue problem

$$n\lambda\boldsymbol{\alpha} = \mathbf{K}\boldsymbol{\alpha}, \tag{15}$$

for nonzero eigenvalues. The solutions $\boldsymbol{\alpha}^k$, belonging to nonzero eigenvalues, are normalized by requiring that the corresponding eigenvectors in a feature space be normalized—that is, $(\boldsymbol{V}^k \cdot \boldsymbol{V}^k) = 1$.

For the principal component extraction, we compute the projections of the image of the data points $\Phi(\boldsymbol{x})$ onto eigenvectors $\boldsymbol{V}^k$ in the feature space

$$\begin{aligned} \tilde{\boldsymbol{x}} = (\boldsymbol{V}^k \cdot \Phi(\boldsymbol{x})) &= \sum_{i=1}^n \alpha_i^k (\Phi(\boldsymbol{x}_i) \cdot \Phi(\boldsymbol{x})) \\ &= \sum_{i=1}^n \alpha_i^k k(\boldsymbol{x}_i, \boldsymbol{x}), \end{aligned} \tag{16}$$

where $k(\boldsymbol{x}, \boldsymbol{y})$ is the kernel function. We use the radial basis function $k(\boldsymbol{x}, \boldsymbol{y}) = \exp\left(-\frac{\| \boldsymbol{x} - \boldsymbol{y} \|^2}{2\sigma^2}\right)$ as the kernel function.

The EK-HEC algorithm performs the K-HEC algorithm in a kernel feature space. The EK-HEC algorithm is described as follows.

Algorithm 2. EK-HEC algorithm

Input: Given a set of n data points in a d-dimensional space, $\boldsymbol{x} = \{\boldsymbol{x}_i \in \mathbb{R}^d\}_{i=1}^n$.

Output: The partition matrix **P**.

Step 1: Compute kernel matrix **K**.

$$K_{ij} = \Phi(\boldsymbol{x}_i) \cdot \Phi(\boldsymbol{x}_j) = k(\boldsymbol{x}_i, \boldsymbol{x}_j); \; i, \; j = 1, 2, \ldots, n.$$

Step 2: Find a solution $\boldsymbol{\alpha}$ of the eigenvalue problem $n\lambda\boldsymbol{\alpha} = \mathbf{K}\boldsymbol{\alpha}$.

Step 3: Using the solutions $\boldsymbol{\alpha}^k$, belonging to nonzero eigenvalues, compute projections of the image of data points, $\Phi(\boldsymbol{x})$, onto eigenvectors, $\boldsymbol{V}^k$, in a feature space

$$\begin{aligned} \tilde{\boldsymbol{x}} = (\boldsymbol{V}^k \cdot \Phi(\boldsymbol{x}_i)) &= \sum_{i=1}^n \alpha_i^k (\Phi(\boldsymbol{x}_i) \cdot \Phi(\boldsymbol{x})) \\ &= \sum_{i=1}^n \alpha_i^k k(\boldsymbol{x}_i, \boldsymbol{x}). \end{aligned}$$

Step 4: Fix the number of clusters C. Randomly select C data points in a feature space and set these as the centers of clusters, $\tilde{\boldsymbol{m}}_k$.

Step 5: Initially determine the assignment of the partition matrix **P**, using the Euclidean distance measure with $\tilde{\boldsymbol{m}}_k$.

$$P_{ik} = 1, \text{ if } D_{\text{Euc}}(\tilde{\boldsymbol{x}}_i, \tilde{\boldsymbol{m}}_k) < D_{\text{Euc}}(\tilde{\boldsymbol{x}}_i, \tilde{\boldsymbol{m}}_j),$$
$$j = 1, 2, \ldots, C \text{ and } j \neq k.$$
$$P_{ik} = 0, \text{ otherwise.}$$

Step 6: Compute new centers of clusters and the number of data points belonging to the clusters

$$\tilde{\boldsymbol{m}}_k = \frac{\sum_i P_{ik}\tilde{\boldsymbol{x}}_i}{\sum_i P_{ik}}, \quad n_k = \sum_i P_{ik}.$$

Step 7: Compute the pseudo-covariance matrix $\tilde{\mathbf{Q}}_k$, using the MVE approximation algorithm.

Step 8: Determine a new assignment of the partition matrix **P**, using the modified Gaussian kernel measure (3) with $\tilde{\boldsymbol{m}}_k$ and

$\tilde{\mathbf{Q}}_k$.

$P_{ik} = 1$, if $D_{\mathrm{MGK}}(\tilde{\boldsymbol{x}}_i, \tilde{\boldsymbol{m}}_k; \tilde{\mathbf{Q}}_k) < D_{\mathrm{MGK}}(\tilde{\boldsymbol{x}}_i, \tilde{\boldsymbol{m}}_j; \tilde{\mathbf{Q}}_j)$,

$j = 1, 2, \ldots, C$ and $j \neq k$.

$P_{ik} = 0$, otherwise.

Step 9: If there are no changes in partition matrix **P**, then STOP. Otherwise, repeat Steps 6 through Step 8.

IV. Experimental Results

To validate the proposed HEC algorithms, we conducted experiments on synthetic and benchmark datasets. The experiments described in this paper were conducted on a PC with an Intel(R) Xeon(R) CPU W5590 @ 3.33 GHZ, and all algorithms were realized using Matlab with CVX [20], [21] and LMI Toolbox. We employed the misclassification rate (MCR) [6] and the normalized mutual information (NMI) [6], [16], [22] as performance evaluation measurements. The MCR can be computed by dividing the number of data points that are not in a correct cluster by the total number of data points in the dataset. Given two random variables, $\boldsymbol{X}$ and $\boldsymbol{Y}$, the NMI can be formulated as follows [6], [16]:

$$\mathrm{NMI}(\boldsymbol{X}, \boldsymbol{Y}) = \frac{I(\boldsymbol{X}, \boldsymbol{Y})}{\left[H(\boldsymbol{X}) + H(\boldsymbol{Y})\right]}, \qquad (17)$$

where $I(\boldsymbol{X}, \boldsymbol{Y})$ is the mutual information and $H(\boldsymbol{X})$ is the entropy of $\boldsymbol{X}$.

1. Synthetic Dataset

To compare the performance of the MVE approximation algorithms, a dataset with a Gaussian distribution was used in the experiments. The dataset consists of 1,000 data points in a 2-dimensional space, which is generated using the multivariate Gaussian function in Matlab.

We first estimated the execution time for approximating the MVE, as shown in Table 1. Moshtagh's method of (11), based on Khachiyan's algorithm, was the fastest. The boundaries of the approximated MVE are illustrated in Fig. 1. The ellipses with the red lines are depicted with the centers obtained using the approximation algorithms. The ellipses with the blue lines are depicted with the centers $\boldsymbol{m} = [1.0085\ 1.0113]^T$, computed using (5). As shown in Fig. 1, the pseudo-covariance matrices, **Q** are well approximated using all methods. Although the computation time of (9) is much faster than that of (8), they have the same approximation results ($\mathbf{Q}=\begin{bmatrix} 0.0225 & -0.0077 \\ -0.0077 & 0.0189 \end{bmatrix}$ and $\boldsymbol{m} = [0.9485\ \ 0.7902]^T$). The decision boundaries of (9) and (11) are quite similar compared to the results of (10). The MVE approximated by (10) has the pseudo-covariance matrices, $\mathbf{Q} = \begin{bmatrix} 0.0228 & -0.0060 \\ -0.0060 & 0.0173 \end{bmatrix}$ and the center $\boldsymbol{m} = [0.9014\ 0.5369]^T$ and the MVE approximated by (11) has $\mathbf{Q} = \begin{bmatrix} 0.0230 & -0.0081 \\ -0.0081 & 0.0194 \end{bmatrix}$ and $\boldsymbol{m} = [0.9532\ \ 0.7472]^T$.

To illustrate the efficacy of the proposed method, three synthetic datasets were used in the experiments. The detailed characteristics of the synthetic datasets are given in Table 2.

Table 1. Execution time of MVE approximation algorithms.

Algorithm	(8)	(9)	(10)	(11)
Execution time	46.50 sec	6.77 sec	1.89 sec	0.71 sec

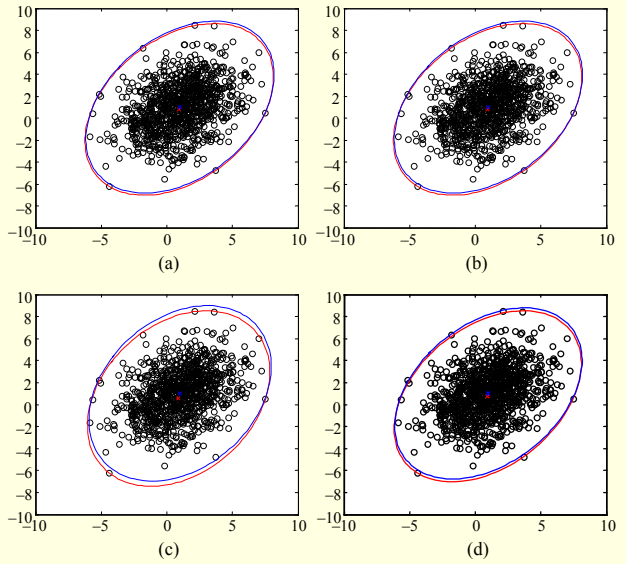

Fig. 1. Illustrated decision boundaries of approximated MVE. Red lines are drawn with centers obtained by approximation algorithms and blue lines are drawn with center computed by (5): (a) MVE approximated by (8); (b) MVE approximated by (9); (c) MVE approximated by (10); and (d) MVE approximated by (11).

Table 2. Three synthetic datasets for the experiments.

Dataset	Dataset 1		Dataset 2		Dataset 3	
Description	Ball-ellipse shape		Gaussian-banana shape		Gaussian-circle shape	
Dimension	2		2		2	
Number of clusters	2		2		2	
Number of instances	14		128		156	
	4	10	100	28	56	100

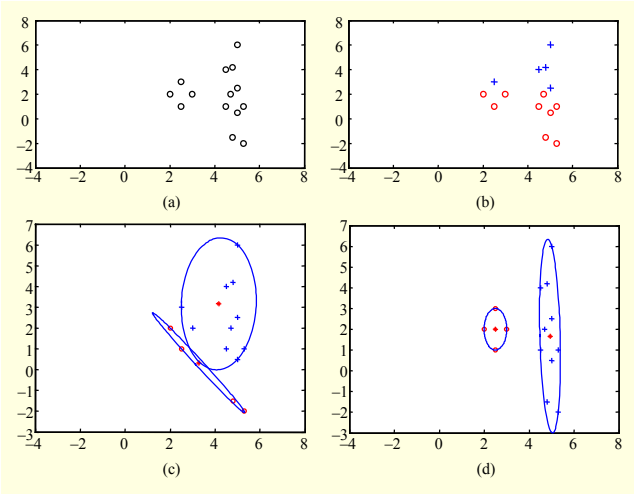

Fig. 2. Synthetic dataset 1 and some clustering results of this data: (a) dataset consisting of two clusters of different size, different density, and different shape, (b) clustering results of K-means algorithm, (c) clustering results of HEC algorithm with Mahalanobis distance, and (d) clustering results of proposed K-HEC algorithm, with $\alpha = 0.2$.

The first dataset shows the capability of the K-HEC algorithm, which is able to deal with clusters of different size, different density, and ellipsoidal shapes. This dataset contains a small ball-shaped cluster and an elongated cluster in the shape of an ellipse. The clustering results of the K-HEC with three different MVE approximation methods (9)–(11), are the same on dataset 1; the rest — that is, (9) and (10) — are omitted. The clustering results for dataset 1 are depicted in Fig. 2.

As shown in Fig. 2, the proposed K-HEC algorithm can separate the two clusters well, but the other methods fail to find correct clusters in dataset 1. As we mentioned in the introduction, the K-means algorithm performs well for clusters of equal size, equal density, and spherical shapes. Although the HEC algorithm with Mahalanobis distance divides data points into clusters of different size and ellipsoidal shapes, it fails to find the correct clusters when two clusters are close together. In the case of two clusters that are far enough apart, the algorithm is able to find the correct clusters. The proposed K-HEC algorithm can control the clustering results by adjusting the variable $\alpha \in [0, 1]$, which controls the weight of the first and second terms of the modified Gaussian kernel. This experiment shows that the K-HEC algorithm — which minimizes the weighted sum of volumes of clusters — allocates data points into clusters such that its compactness and density are maximized and consequently finds condensed clusters with respect to the volumes and densities of the clusters.

The second and third datasets were generated to validate the capability of the EK-HEC algorithm, which is designed for and specializes in handling geometrically complex-shaped clusters,

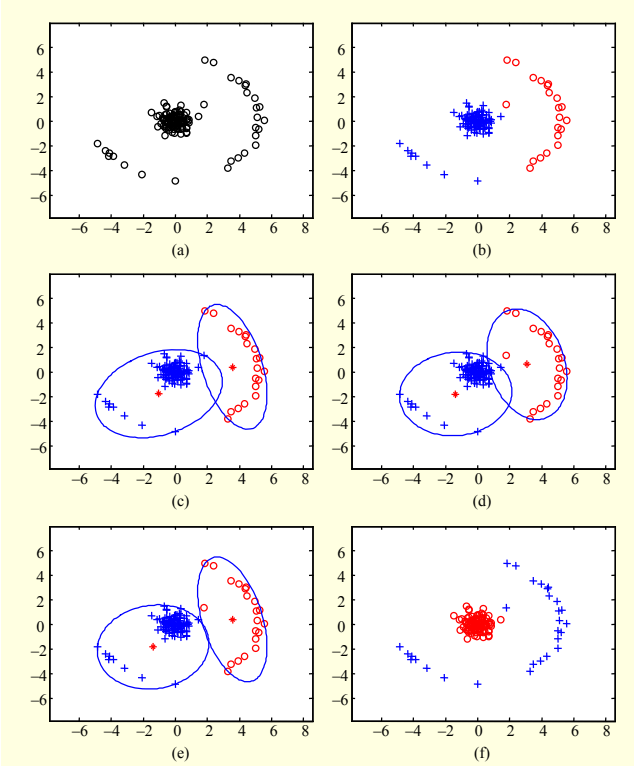

Fig. 3. Synthetic dataset 2 and some clustering results of this data: (a) Gaussian-banana shaped dataset, (b) clustering results of K-means algorithm, (c) clustering results of HEC algorithm with Mahalanobis distance, (d) clustering results of K-HEC algorithm with LMI, α=0.2, (e) clustering results of K-HEC algorithm using Moshtagh's method, α=0.2, and (f) clustering results of EK-HEC algorithm, with α=0.02.

as shown in Figs. 3 and 5. The clustering results of the K-HEC with MVE approximations (9) and (10) are omitted, because their clustering results are the same as the clustering results of the K-HEC based on Moshtagh's method, which are depicted in Figs. 3(e) and 5(e).

Figure 3 illustrates the clustering results on dataset 2, which consists of a cluster with Gaussian distribution and a cluster with a banana shape. The first four methods show similar results, except for one data point. Note that the decision boundaries of clusters determined by each algorithm are quite different, although the clustering results are similar. The K-HEC algorithm with Moshtagh's MVE approximation creates the most separable decision boundaries compared to the others. This implies that, in this case, the Moshtagh's MVE approximation finds a more compact ellipsoid than the linear matrix inequality (LMI)-based MVE approximation. The HEC algorithm with a Mahalanobis distance and the K-HEC algorithm with LMI have overlapped decision boundaries. The EK-HEC algorithm is able to find the correct clusters as expected and only one data point is clustered incorrectly. The

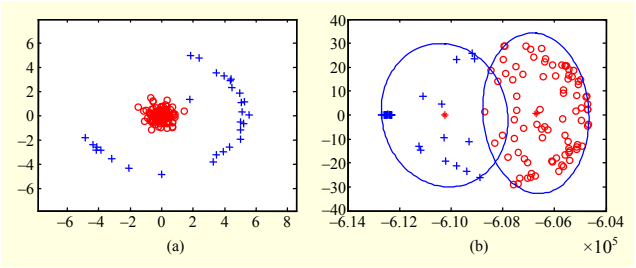

Fig. 4. Clustering results of EK-HEC algorithm on dataset 2: (a) clustering results and dataset depicted in input space and (b) clustering results and dataset depicted in feature space.

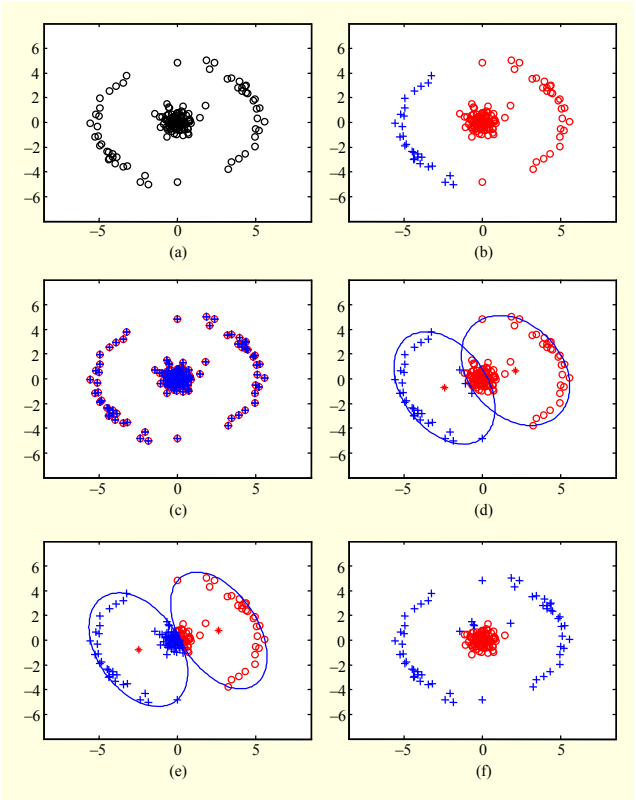

Fig. 5. Synthetic dataset 3 and some clustering results of this data: (a) Gaussian-circle shaped dataset, (b) clustering results of K-means algorithm, (c) clustering results of HEC algorithm with a Mahalanobis distance, (d) clustering results of K-HEC algorithm using LMI, α=0.2, (e) clustering results of K-HEC algorithm using Moshtagh's method, α=0.2, and (f) clustering results of EK-HEC algorithm, with α=0.09.

clustering results of the EK-HEC algorithm on dataset 2 are illustrated in Fig. 4. The clustering results in the input and feature spaces are depicted in Figs. 4(a) and 4(b), respectively. The data points mapped onto the feature space are well separable and can be clustered based on the decision boundaries of the ellipsoids.

The clustering results on dataset 3 are presented in Fig. 5. Unlike the experiments on dataset 2, the HEC algorithm with

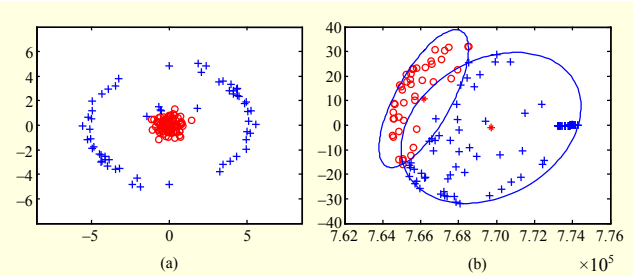

Fig. 6. Clustering results of EK-HEC algorithm on dataset 3: (a) clustering results and dataset depicted in input space and (b) clustering results and dataset depicted in feature space.

the Mahalanobis distance fails to allocate data points into two clusters. The drawback of the HEC with Mahalanobis distance is that it may produce empty clusters if all data points in a cluster are reassigned to others, thereby, reducing the number of clusters; such as in the K-means algorithm [10]. The K-HEC algorithm based on Moshtagh's method creates more compact decision boundaries than the K-HEC with LMI, as in the experiments on dataset 2. The EK-HEC algorithm finds the correct clusters and few data points are allocated incorrectly. The clustering results of EK-HEC on dataset 3 are given in Fig. 6. The clustering results in the input and feature spaces are shown in Figs. 6(a) and 6(b), respectively. In this case, the cluster boundaries in the feature space are close to each other and overlap.

As illustrated in Figs. 4 and 6, the EK-HEC algorithm is able to handle complex-shaped data by applying the kernel trick—that is, KPCA. The EK-HEC algorithm is more suitable for applications that deal with complex-shaped datasets, such as geospatial analysis and image segmentations. The experimental results on three synthetic datasets show that the K-HEC algorithm performs well for datasets with clusters of different size, different density, and ellipsoidal shapes and that EK-HEC has the capability to deal with geometrically complex-shaped clusters.

2. Benchmark Dataset

We conducted the experiments on the five benchmark datasets from UCI data repository [23] to show the general clustering performance of the proposed methods. The descriptions of the benchmark datasets and the values of weight parameter α are presented in Table 3.

We compared the proposed methods against K-means clustering, fuzzy C-means clustering, GMM-EM algorithm, and the MVE-based HEC algorithm with a Mahalanobis distance. The clustering results on the benchmark datasets are shown in Fig. 7. The K-HEC algorithm outperforms K-means,

Table 3. Benchmark datasets from UCI.

Dataset		Iris	Wine	Glass	Yeast	Breast tissue
Attributes		4	13	11	8	10
Clusters		3	3	7	10	6
Instances		150	178	214	1,484	106
Values of α	K-HEC	0.3	0.6	0.95	0.6	0.5
	EK-HEC	0.5	0.45	0.4	0.4	0.4

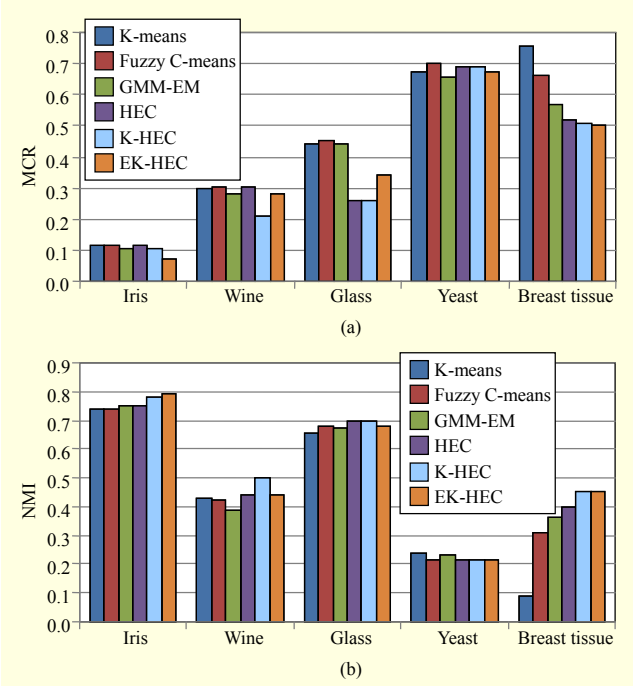

Fig. 7. Clustering results of benchmark dataset: (a) MCR and (b) NMI.

fuzzy C-means, GMM-EM, and the HEC algorithm on Iris, Wine, Glass, and Breast Tissue datasets in terms of MCR and NMI criterion. On Yeast dataset, it shows comparable performance. The EK-HEC algorithm shows better or comparable clustering performance compared to the others.

3. Sensitivity Analysis of Weight Parameter

In the proposed algorithms, the parameter α controls the weight of the first and second terms of the modified Gaussian kernel (3) and influences the result of clustering. We empirically analyze the sensitivity of the weight parameter α on the iris dataset of UCI repository.

Figure 8 shows the changes in the MCR and NMI value for the two proposed methods with the weight parameter α varying from 0.0 to 1.0. The experiments indicate that the K-HEC and EK-HEC algorithms achieve stable and desirable clustering results, for values of α ranging from 0.3 to 0.6, in terms of MCR and NMI criterion. It should be noted that the selection of the weight parameter α depends on a dataset. In the experiments of this paper, the value of the weight parameter α was set empirically.

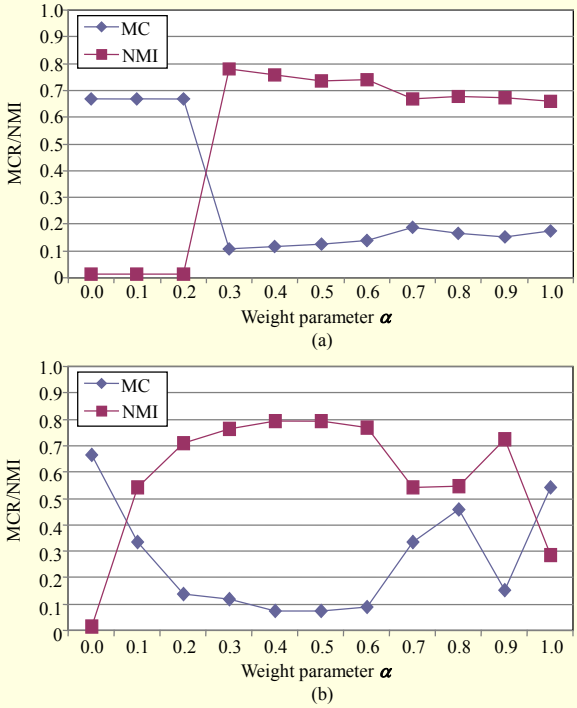

Fig. 8. Sensitivity analysis of weight parameter α: (a) K-HEC algorithm and (b) EK-HEC algorithm.

V. Conclusion

We described a theoretical explanation and proposed the practical implementations of the HEC algorithm based on MVE with a modified Gaussian kernel. First, we showed that the HEC with a modified Gaussian kernel metric can be formulated as the problem of finding condensed clusters with respect to their volume and density. Second, we proposed a practical HEC algorithm, namely, K-HEC, which is able to handle clusters of different size, different density, and ellipsoidal shapes. Third, we enhanced the capability of the K-HEC algorithms—namely, EK-HEC—by utilizing ellipsoids defined on the kernel feature space, which is able to deal with geometrically complex-shaped clusters—that is, a nonlinear and elongated structure. The experimental results showed that the K-HEC algorithm can effectively separate the clusters by creating compact boundaries of clusters. Unlike other methods, EK-HEC can work on nonlinear and elongated structures, as can be seen from the experiments. In our experiments, the

ETRI Journal, Volume 36, Number 3, June 2014
http://dx.doi.org/10.4218/etrij.14.0113.0553

proposed approaches outperformed the K-means algorithm, fuzzy C-means algorithm, GMM-EM algorithm, and MVE-based HEC algorithm with a Mahalanobis distance.

References

[1] A.K. Jain, "Data Clustering: 50 Years Beyond K-means," *Pattern Recognition Lett.*, vol. 31, no. 8, June 2010, pp. 651–666.

[2] J. Han and M. Kamber, *Data Mining: Concepts and Techniques*, Morgan Kaufmann Publishers, 2nd ed., 2007.

[3] J. Mao and A.K. Jain, "A Self-Organizing Network for Hyperellipsoidal Clustering (HEC)," *IEEE Trans. Neural Netw.*, vol. 7, no. 1, Jan. 1996, pp. 16–29.

[4] W. Song et al., "The Hyperellipsoidal Clustering Using Genetic Algorithm," *IEEE Int. Conf. Intell. Process. Syst.*, Beijing, China, Oct. 28–31, 1997, pp. 592–596.

[5] H. Ichihashi, M. Ohue, and T. Miyoshi, "Fuzzy C-Means Clustering Algorithm with Pseudo Mahalanobis Distances," *Proc. Third Asian Fuzzy Syst. Symp.*, Changwon, Rep. of Korea, June 18–21, 1998, pp. 148–152.

[6] M. Moshtaghi et al., "An Efficient Hyperellipsoidal Clustering Algorithm for Resource-Constrained Environments," *Pattern Recogn.*, vol. 44, no. 9, Sept. 2011, pp. 2197–2209.

[7] W. Song et al., "Comments on a Self-Organizing Network for Hyper-ellipsoidal Clustering (HEC)," *IEEE Trans. Neural Netw.*, vol. 8, no. 6, Nov. 1997, pp. 1561–1563.

[8] R. Krishnapuram and J. Kim, "A Clustering Algorithm Based on Minimum Volume," *IEEE Int. Conf. Fuzzy Syst.*, vol. 2, New Orleans, LA, USA, Sept. 8–11, 1996, pp. 1387–1392.

[9] H. Lee, J. Park, and D. Park, "Hyper-ellipsoidal Clustering Algorithm Using Linear Matrix Inequality," *J. Korea Institute Intell. Syst.*, vol. 12, no. 4, Aug. 2002, pp. 300–305.

[10] M. Kumar and J.B. Orlin, "Scale-Invariant Clustering with Minimum Volume Ellipsoids," *Comput. Operations. Res.*, vol. 35, no. 4, Apr. 2008, pp. 1017–1029.

[11] R. Shioda and L. Tunçel, "Clustering via Minimum Volume Ellipsoids," *Comput. Optim. Appl.*, vol. 37, no. 3, July 2007, pp. 247–295.

[12] S. Boyd and L. Vandenberghe, *Convex Optimization*, 1st ed., Cambridge, UK: Cambridge University Press, 2004.

[13] M.J. Todd and E.A. Yildirim, "On Khachiyan's Algorithm for the Computation of Minimum-Volume Enclosing Ellipsoids," *Discr. Appl. Math.*, vol. 155, no. 13, Aug. 15, 2007, pp. 1731–1744.

[14] N. Moshtagh, *Minimum Volume Enclosing Ellipsoids*, Tech. report, the School of Engineering and Applied Science, Univ. Pennsylvania, PA, 2005.

[15] P. Kumar and E.A. Yildirim, "Minimum Volume Enclosing Ellipsoids and Core Set," *J. Optim. Theory Appl.*, vol. 126, no. 1, July 2005, pp. 1–21.

[16] J. Cao et al., "A Max-Flow-Based Similarity Measure for Spectral Clustering," *ETRI J.*, vol. 35, no. 2, Apr. 2013, pp. 311–320.

[17] J. Shawe-Taylor and N. Cristianini, *Kernel Methods for Pattern Analysis*, Cambridge, UK: Cambridge University Press, 2004.

[18] B. Schölkopf, A. Smola, and K.-R. Müller, "Kernel Principal Component Analysis," *Proc. ICANN, LNCS*, Lausanne, Switzerland, vol. 1327, Oct. 8–10, 1997, pp. 583–588.

[19] D. Tax and P. Juszczak, "Kernel Whitening for One-Class Classification," *Proc. Pattern Recogn. Support Vector Mach.*, Niagara Fall, Canada, vol. 2388, Aug. 10, 2002, pp. 40–52.

[20] CVX Research, Inc. CVX: Matlab Software for Disciplined Convex Programming, version 2.0. Accessed Apr. 2013. http://cvxr.com/cvx

[21] M. Grant and S. Boyd, "Graph Implementations for Nonsmooth Convex Programs," *Recent Advances Learning Contr. LNCIS*, vol. 371, 2008, pp. 95–110.

[22] D. Cai, X. He, and J. Han, "Document Clustering Using Locality Preserving Indexing," *IEEE Trans. KDE*, vol. 17, no. 12, Dec. 2005, pp. 1624–1637.

[23] The machine learning dataset of UCI are available. Accessed Jan. 2013. http://archive.ics.uci.edu/ml/

Hansung Lee received his BS, MS, and PhD degrees in computer science from Korea University, Sejong City, Rep. of Korea, in 1996, 2002, and 2008, respectively. From July 1996 to July 1999, he worked for Daewoo engineering company, Seoul, Rep. of Korea. He was with Korea University, as a lecturer from March 2002 to February 2010. Since November 2009, he has been with Electronics and Telecommunications Research Institute (ETRI), Daejeon, Rep. of Korea as a senior member of the research staff. His current research interests include pattern recognition, machine learning, optimization, data mining, and big-data analytics.

Jang-Hee Yoo received his BSc in physics from Hankuk University of Foreign Studies (HUFS) Seoul, Rep. of Korea, in 1988 and his MSc in computer science from HUFS, in 1990. He received his PhD in electronics and computer science from the University of Southampton, Hampshire, UK, in 2004. Since November 1989, he has been with Electronics and Telecommunications Research Institute (ETRI), Daejeon, Rep. of Korea as a principal researcher, and is the director of the Human Identification Research Section. He has also been an adjunct professor with the Department of Information Security Engineering at the University of Science and Technology, Daejeon, Rep. of Korea. His current research interests include embedded computer vision, biometric systems, human-motion analysis, intelligent video surveillance, HCI, and intelligent robots. He is a member of the IEEE and the IEEK.

Daihee Park received his BS degree in mathematics from Korea University, Seoul, Rep. of Korea, in 1982 and his PhD degree in computer science from Florida State University, USA, in 1992. He joined Korea University, Sejong City, Rep. of Korea in 1993, where he is currently a professor in the Department of Computer and Information Science. His research interests include data mining and intelligent databases.

ETRI Journal, Volume 36, Number 3, June 2014
http://dx.doi.org/10.4218/etrij.14.0113.0553