Non-Redundant Multi-View Clustering Via Orthogonalization

Ying Cui
ECE Department
Northeastern University
Boston, MA 02115
cui.yi@neu.edu

Xiaoli Z. Fern
School of EECS
Oregon State University
Corvallis, OR 97339
xfern@eecs.oregonstate.edu

Jennifer G. Dy
ECE Department
Northeastern University
Boston, MA 02115
jdy@ece.neu.edu

Abstract

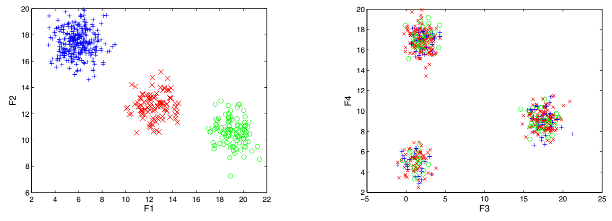
Typical clustering algorithms output a single clustering of the data. However, in real world applications, data can often be interpreted in many different ways; data can have different groupings that are reasonable and interesting from different perspectives. This is especially true for high-dimensional data, where different feature subspaces may reveal different structures of the data. Why commit to one clustering solution while all these alternative clustering views might be interesting to the user. In this paper, we propose a new clustering paradigm for explorative data analysis: find all non-redundant clustering views of the data, where data points of one cluster can belong to different clusters in other views. We present a framework to solve this problem and suggest two approaches within this framework: (1) orthogonal clustering, and (2) clustering in orthogonal subspaces. In essence, both approaches find alternative ways to partition the data by projecting it to a space that is orthogonal to our current solution. The first approach seeks orthogonality in the cluster space, while the second approach seeks orthogonality in the feature space. We test our framework on both synthetic and high-dimensional benchmark data sets, and the results show that indeed our approaches were able to discover varied solutions that are interesting and meaningful.

keywords: multi-view clustering, non-redundant clustering, orthogonalization

1 Introduction

Many applications are characterized by data in high dimensions. Examples include text, image, and gene data. Automatically extracting interesting structure in such data has been heavily studied in a number of different areas including data mining, machine learning and statistical data analysis. One approach for extracting information from unlabeled data is through clustering. Given a data set, typical clustering algorithms group similar objects together based on some fixed notion of similarity (distance) and output a single clustering solution. However, in real world applications, data can often be interpreted in many different ways and there may exist multiple groupings of the data that are all reasonable in some perspective.

This problem is often more prominent for high dimensional data, where each object is described by a large number of features. In such cases, different feature subspaces can often warrant different ways to partition the data, each presenting the user a different view of the data's structure. Figure 1 illustrates one such scenario. In particular, Figure 1a shows a scatter plot of the data in feature subspace $\{F_1, F_2\}$. Figure 1b shows how the data looks like in feature subspace $\{F_3, F_4\}$. Note that each subspace leads to a different clustering structure. When faced with such a situation, which features should we select (i.e., which clustering solution is better)? Why do we have to choose? Why not keep both solutions? In fact both clustering solutions could be important, and provide different interpretations of the same data. For example, for the same medical data, what is interesting to physicians might be different from what is interesting to insurance companies.



(a) (b)

Figure 1. This is a scatter plot of the data in (a) subspace $\{F_1, F_2\}$ and (b) subspace $\{F_3, F_4\}$. Note that the two subspaces lead to different clustering structures.

Hierarchical clustering [18] presents a hierarchical grouping of the objects. However, the different clustering solutions obtained at different hierarchical levels differ only in their granularity – objects belonging to the same cluster in fine resolutions remain in the same cluster at the coarser levels. On the other hand, cluster ensemble methods [22, 10] do create a set of cluster solutions but the final goal is still to find a single clustering solution that is most consistent throughout the set.

The goal of exploratory data analysis is to find structures in data, which maybe multi-faceted by nature. Traditional clustering methods seek to find a unified clustering solution and are thus inherently limited in achieving this goal. In this paper, we suggest a new exploratory clustering paradigm: *the goal is to find a set of non-redundant clustering views from data, where data points belonging to the same cluster in one view can belong to different clusters in another view.*

Toward this goal, we present a framework that extracts multiple clustering views of high-dimensional data that are orthogonal to each other. Note that there are k^N possible k disjoint partitioning of N data points. Not all of them are meaningful. We wish to find good clustering solutions based on a clustering objective function. Meanwhile, we would like to minimize the redundancy among the obtained solutions. Thus, we include an orthogonality constraint in our search for new clustering views to avoid providing the user with redundant clustering results. The proposed framework works iteratively, at each step adding one clustering view by searching for solutions in a space that is orthogonal to the space of the existing solutions. Within this framework, we develop two general approaches. The first approach seeks orthogonality in the cluster space, while the second one seeks orthogonality in the feature subspace. We present all the multiple view clustering solutions to the user.

2 Literature Review

In this section, we briefly review the literature related to our research in different aspects.

First, our research can be considered as performing non-redundant clustering [13, 5]. In non-redundant clustering, we are typically given a set of data objects together with an existing clustering solution and the goal is to learn an alternative clustering that captures new information about the structure of the data. Existing non-redundant clustering techniques include the conditional information bottleneck approach [5, 14, 15], the conditional ensemble based approach [16] and the constrained model based approach [17]. Compared to existing non-redundant clustering techniques, the critical differences of our proposed research are:

(1) Focusing on searching for orthogonal clustering of high-dimensional data, our research combines dimensionality reduction and orthogonal clustering into a unifying framework and seeks lower dimensional representation of the data that reveals non-redundant information about the data.

(2) Existing non-redundant clustering techniques are limited to finding one alternative structure given a known structure. Our framework works successively to reveal a sequence of different clustering of the data.

Our framework produces a set of different clustering solutions, which is similar to cluster ensembles[22, 10]. A clear distinction of our work from cluster ensembles is that we intentionally search for orthogonal clusterings and do not seek to find a consensus clustering as our end product.

Similar to cluster ensembles, multi-view clustering [3] seek a single final clustering solution by learning from multiple views, whereas our non-redundant multi-view clustering paradigm looks for multiple clustering solutions.

An integral part of our framework is to search a clustering structure in a high-dimensional space and find the corresponding subspace that best reveals the clustering structure. This topic has been widely studied, and one closely related work was conducted by [8]. In this work, after a clustering solution is obtained, a subspace is computed to best capture the clustering, and the clustering is then refined using the data projected onto the new subspace. Interestingly, our framework works in an opposite direction. We look at a subspace that is orthogonal to the space in which the original clustering is embedded to search for non-redundant clustering solutions.

Finally, while we search for different subspaces in our framework, it is different from the concept of subspace clustering in [1, 21]. In subspace clustering, the goal is still to learn a single clustering, where each cluster can be embedded in its own subspace. In contrast, our method searches for multiple clustering solutions, each is revealed in a different subspace.

3 Multi-View Orthogonal Clustering

Given data $X \in R^{d \times N}$, with N instances and d features. *Our goal is to learn a set of orthogonal clustering views from the data.*

There are several ways to find different clustering views [22, 10]. One can apply different objective functions, utilize different similarity measures, or use different density models. In general, one can apply different clustering algorithms, or utilize one algorithm on randomly sampled (either in instance space or feature space) data from X. Note that such methods produce each individual clustering independently from all the other clustering views. While the differences in the objective functions, similarity measures, density models, or different data samples may lead to clustering results that differ from one another, it is common to see redundancy in the obtained multiple clustering views. We present a framework for successively generating multiple clustering views that are orthogonal from one another and thus contain limited redundancy. Below, we describe our framework for generating different clustering views by orthogonalization.

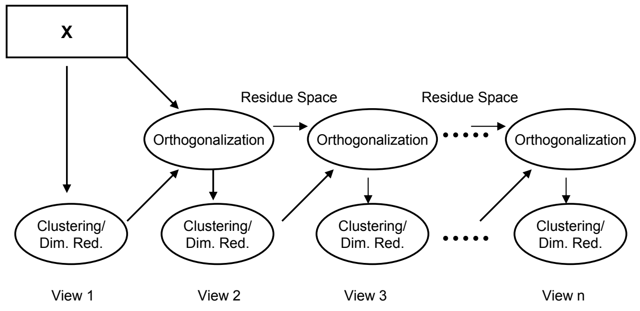


Figure 2. The general framework for generating multiple orthogonal clustering views.

Figure 2 shows the general framework of our approach. We first cluster the data (this can include dimensionality reduction followed by clustering), then we orthogonalize the data to a space that is not covered by the existing clustering solutions. Repeat the process until we cover most of the data space or no structure can be found in the remaining space.

We developed two general approaches within this framework: (1) orthogonal clustering, and (2) clustering in orthogonal subspaces.

These two approaches differ primarily in how they represent the existing clustering solutions and consequently how to orthogonalize the data based on existing solutions. Specifically, the first approach represents a clustering solution using its k cluster centroids. The second approach represents a clustering solution using the feature subspace that best captures the clustering result. In the next two subsections, we describe these two different representations in detail and explain how to obtain orthogonal clustering solutions based on these two representations.

3.1 Orthogonal Clustering

Clustering can be viewed as a way for compressing data X. For example in k-means [11, 20], the objective function is to minimize the sum-squared-error criterion (SSE):

$$SSE = \sum_{j=1}^{k} \sum_{x_i \in C_j} ||x_i - \mu_j||^2$$

where $x_i \in R^d$ is a data point assigned to cluster C_j, and μ_j is the mean of C_j. We represent x_i and μ_j as column vectors. The outputs for k-means clustering are the cluster means and the cluster membership of each data point x_i. One can consider k-means clustering as a compression of data X to the k cluster means μ_j.

Following the compression viewpoint, each data point x_i is represented by its cluster mean μ_j. Given k μ_j's for representing X, what is not captured by these μ_j's? Let us consider the space that is spanned by x_i, $i = 1 \ldots N$, we refer to this as the original data space. In contrast, the subspace that is spanned by the mean vectors μ_j, $j = 1 \ldots k$, is considered the compressed data space. Assigning data points to their corresponding cluster means can be essentially considered as projecting the data points from the original data space onto the compressed data space. What is not covered by the current clustering solution (i.e., its compressed data space) is simply its *residue space*. In this paper, we define residue space as the data projected onto the space orthogonal to our current representation. Thus, to find alternative clustering solutions not covered in the current solution, we can simply perform clustering in the space that is orthogonal to the compressed data space.

Given current data $X^{(t)}$, and the clustering solution we found on $X^{(t)}$ (i.e., $M^{(t)} = [\mu_1^{(t)} \mu_2^{(t)} \cdots \mu_k^{(t)}]$, and the cluster assignments), we describe two variations for representing data in the residue space, $X^{(t+1)}$ that are based on the "hard" and "soft" clustering views respectively.

Hard clustering. In hard clustering, each data point belongs to a single cluster, thus is represented using a single mean vector. For data point $x_i^{(t)}$ belonging to cluster j, we project it onto its center $\mu_j^{(t)}$ as its representation in the current clustering view. The residue, $x_i^{(t+1)}$, is then the projection of $x_i^{(t)}$ onto the subspace orthogonal to $\mu_j^{(t)}$. This can be formalized by the following formula:

$$x_i^{(t+1)} = (I - \mu_j^{(t)} \mu_j^{(t)T} / (\mu_j^{(t)T} \mu_j^{(t)})) x_i^{(t)}$$

Soft clustering. In soft clustering, each data point can partially belong to all clusters. Therefore, we can represent the data using all cluster centers. We achieve this by projecting the data onto the subspace spanned by all cluster means and compute the residue, $X^{(t+1)}$, as the projection of $X^{(t)}$ onto the subspace orthogonal to all the cluster centroids. This can be formalized by the following formula:

$$X^{(t+1)} = (I - M^{(t)} (M^{(t)T} M^{(t)})^{-1} M^{(t)T}) X^{(t)}$$

The algorithm for orthogonal clustering is summarized in Algorithm 1. The data is first centered to have zero mean. We then create the first view by clustering the original data X. Since most of the data in our experiments are high-dimensional, we apply principal components analysis [19] to reduce the dimensionality, followed by k-means.

```
Algorithm 1 Orthogonal Clustering.
  Inputs: The data matrix $X \in R^{d \times N}$, and the number of clusters $k^{(t)}$ for each iteration.
  Output: The multiple partitioning views of the data into $k^{(t)}$ clusters at each iteration.
  Pre-processing: Center the data to have zero mean.
  Initialization: Set the iteration number $t = 1$ and $X^{(1)} = X$.
  Step1: Cluster $X^{(t)}$. In our experiments, we performed PCA followed by k-means. The compressed solution are the $k$ means, $\mu_j^{(t)}$. Each $\mu_j^{(t)}$ is a column vector in $R^d$ (the original feature space).
  Step2: Project each $x_i^{(t)}$ in $X^{(t)}$ to the space orthogonal to its cluster mean to form the residue space representation, $X^{(t+1)}$.
  Step3: Set $t = t + 1$ and repeat steps 1 and 2 until the desired number of views or until the sum-squared-error, $\sum_{j=1}^{k} \sum_{x_i^{(t)} \in C_j^{(t)}} ||x_i^{(t)} - \mu_j^{(t)}||^2$, is very small.
```

Note that one can apply other clustering methods within our framework. We chose PCA followed by k-means because they are popular techniques. In step 2, we project the data to the space orthogonal to the current cluster representation (using cluster centers) to obtain our residue $X^{(t+1)}$. The next clustering view is then obtained by clustering in this residue space. We repeat steps 1 (clustering) and 2 (orthogonalization) until the desired number of views are obtained or when the SSE is very small. Small SSE signifies that the existing views have already covered most of the data.

3.2 Clustering in Orthogonal Subspaces

In this approach, given a clustering solution with means μ_j, $j = 1 \dots k$, we would like to find a feature subspace that best captures the clustering structure, or, in other words, discriminates these clusters well. One well-known method for finding a reduced dimensional space that discriminates classes (clusters here) is linear discriminant analysis (LDA) [12, 9]. Another approach is by applying principal component analysis (PCA) on the k mean vectors μ_j's [8].

Below we explain the mathematical differences between these two approaches. LDA finds a linear projection $Y = A^T X$ that maximizes the

$$trace(S_w^{-1} S_b)$$

where S_w is the within-class-scatter matrix and S_b is the between-class-scatter matrix defined as follows.

$$S_w = \sum_{j=1}^{k} \sum_{y_i \in C_j} (y_i - \mu_j)(y_i - \mu_j)^T$$

$$S_b = \sum_{j=1}^{k} n_j (\mu_j - \mu)(\mu_j - \mu)^T$$

where y_i's are projected data points; μ_j's are projected cluster centers; n_j is the total number of points in cluster j and μ is the center of the entire set of projected data. In essence, LDA finds the subspace that maximizes the scatter between the cluster means normalized by the scatter within each cluster.

Similarly, the PCA approach in [8] seeks a linear projection $Y = A^T X$, but maximizes a different objective function $trace(MM^T)$, where $M = [\mu_1 \mu_2 \cdots \mu_k]$ and the μ_j's are the projected cluster centers.

For both methods, the solution can be represented as $A = [\phi_1 \phi_2 \cdots \phi_q]$, which contains the q most important eigenvectors (corresponding to the q largest eigenvalues) of $S_w^{-1} S_b$ for LDA and MM^T for PCA respectively.

Note that the $trace(S_b) = trace(M' M'^T)$, where $M' = [\sqrt{n_1}\mu_1 \sqrt{n_2}\mu_2 \cdots \sqrt{n_k}\mu_k]$. The difference between the two approaches is the normalization by the within-class-scatter S_w^{-1} and a weighting of each μ_j by n_j, the number of data points in cluster C_j. But, both M and M' span the same space. Thus, in practice, both approaches result in similar results. For computational purposes, we choose the PCA approach on the μ_j's and set $q = k - 1$, the rank of MM^T.

Once we have obtained a feature subspace $A = [\phi_1 \phi_2 \cdots \phi_{k-1}]$ that captures the clustering structure well, we project $X^{(t)}$ to the subspace orthogonal to A to obtain the residue $X^{(t+1)} = P^{(t)} X^{(t)}$. The orthogonal projection operator, P, is: $P^{(t)} = I - A^{(t)} ({A^{(t)}}^T A^{(t)})^{-1} {A^{(t)}}^T$.

Algorithm 2 presents the pseudo-code for clustering in orthogonal subspaces. In step 1, we apply a clustering algorithm (PCA followed by k-means in our experiments). We then represent this clustering solution using the subspace that best separates these clusters. In step 2, we project the data to the space orthogonal to the computed subspace representation. We repeat steps 1 and 2 until the desired number of views are obtained or the SSE is very small.

4 Experiments

In this section, we investigate whether our multi-view orthogonal clustering framework can provide us with reasonable and orthogonal clustering views of the data. We start by performing experiments on synthetic data in Section 4.1 to get a better understanding of the methods, then we test the methods on benchmark data in Section 4.2. In these experiments, we chose as our base clustering – PCA followed by k-means clustering. This means, we first reduce the dimensionality with PCA, keeping a dimensionality that retains at least 90% of the original variance, then follow PCA

Algorithm 2 Clustering in Orthogonal Subspaces.

Inputs: The data matrix $X \in R^{d \times N}$, and the number of clusters $k^{(t)}$ for each iteration.

Output: The multiple partitioning views of the data into $k^{(t)}$ clusters, and a reduced dimensional subspace for each iteration $A^{(t)}$.

Pre-processing: Center the data to have zero mean.

Initialization: Set the iteration number $t = 1$ and $X^{(1)} = X$.

Step1: Cluster $X^{(t)}$. In our experiments, we performed PCA followed by k-means. Then, find the PCA solution of $M^{(t)}$, $M^{(t)} = [\mu_1^{(t)} \mu_2^{(t)} \cdots \mu_k^{(t)}]$ and keep $k^{(t)} - 1$ dimensions to obtain the subspace, $A^{(t)}$, that captures the current clustering.

Step2: Project $X^{(t)}$ to the space orthogonal to $A^{(t)}$ to produce $X^{(t+1)} = P^{(t)} X^{(t)}$, where the projection operator $P^{(t)}$ is:

$$P^{(t)} = I - A^{(t)} ({A^{(t)}}^T A^{(t)})^{-1} {A^{(t)}}^T$$

Step3: Set $t = t + 1$ and repeat steps 1 and 2 until the desired number of views or until the sum-squared-error, $\sum_{i=1}^{N} ||x_i^{(t)} - A^{(t)} y_i^{(t)}||^2$, is very small.

with k-means clustering. Because we apply k-means clustering, we implement orthogonal clustering with the "hard" assumption. In this section, we refer to the orthogonal clustering approach as method 1, and the clustering in orthogonal subspaces as method 2.

4.1 Experiments on Synthetic Data

We would like to see whether our two methods can find diverse groupings of the data. We generate two synthetic data.

Data 1: We generate a four-cluster data in two dimensions with $N = 500$ instances as shown in Figure 3, where each cluster contains 125 data points. We test our methods by setting $k = 2$ for our k-means clustering. We would like to see that if the methods group the clusters into two in the first iteration, then they should group the clusters the other way in the next iteration. This data tests whether the methods can find orthogonal clusters.

Data 2: We generate a second synthetic data in four dimensions, with $N = 500$ instances as shown in Figure 4. We generate three Gaussian clusters in features F_1 and F_2 with 100, 100 and 300 data points and means $\mu_1 = (12.5, 12.5)$, $\mu_2 = (19, 10.5)$, and $\mu_3 = (6, 17.5)$, and identity covariances. We generate another mixture of three Gaussian clusters in features F_3 and F_4 with 200, 200 and 100 data points and means $\mu_1 = (2, 17)$, $\mu_2 = (17.5, 9)$, and $\mu_3 = (1.2, 5)$, and identity covariances. This data tests whether the methods can find different clustering solutions in different subspaces.

Table 1. Confusion Matrix for Synthetic Data1

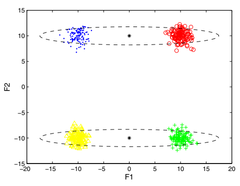
SYNTHETIC DATA1	METHOD1		METHOD2	
ITERATION1	C1	C2	C1	C2
L1	**125**	0	**125**	0
L2	0	**125**	0	**125**
L3	**125**	0	**125**	0
L4	0	**125**	0	**125**
ITERATION2	C1	C2	C1	C2
L1	**125**	0	**125**	0
L2	**125**	0	**125**	0
L3	0	**125**	0	**125**
L4	0	**125**	0	**125**

4.1.1 Results for Synthetic Data 1

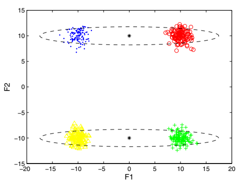
The confusion matrix in Table 1 shows the experimental results for synthetic data 1 for methods 1 and 2, in two iterations. We can see that for the first iteration, both methods grouped classes L_1 and L_3 into a single cluster C_1, and classes L_2 and L_4 into another cluster C_2. For the second iteration, the data was partitioned in a different way, which grouped classes L_1 and L_2 into one cluster, and classes L_3 and L_4 into another cluster. Figure 3 shows the scatter plot of the clustering results of both methods in the original $2D$ data space for the two iterations. Different colors are used to signify the true classes, and the ellipses show the clusters found by k-means. The figure confirms the result summarized in the confusion matrix. Both methods 1 and 2 have similar results as shown. In subfigure $a3$ and $b3$ of Figure 3, we plot the sum-squared-error (SSE) as a function of iteration. Note that, as expected, SSE for both methods decreases monotonically until convergence. Moreover, the SSE at iteration 2 and after is zero meaning that the first two clustering views have covered the data space completely.

4.1.2 Results for Synthetic Data 2

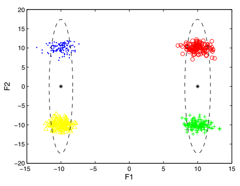
Table 2 shows the confusion matrix for our clustering with the two different labelings: labeling 1 is for features 1 and 2, and labeling 2 is for features 3 and 4. High number of common occurrences means that the cluster correspond to those labels. Observe that for both methods 1 and 2, they found the clusters in labeling 2 (features 3 and 4) perfectly with zero confusion in the off-diagonal elements in the first iteration/view. In the second iteration/view, methods 1 and



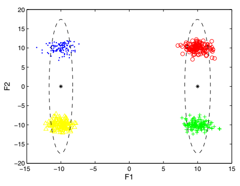
(a1.iteration1 for method1)



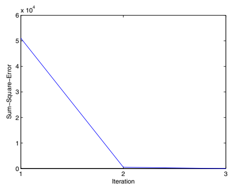
(b1.iteration1 for method2)



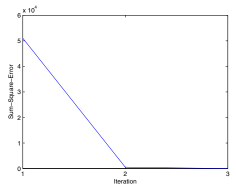
(a2.iteration2 for method1)



(b2.iteration2 for method2)



(a3.SSE for method1)



(b3.SSE for method2)

Figure 3. Scatter plots of synthetic data 1. The two columns show the results of methods 1 and 2 respectively. The colors represent different class labels and the ellipses represent the clusters found. Row 1 and 2 show the results for iteration 1 and 2 respectively; Row 3 shows SSE as a function of iteration.

2 found the clusters in labeling 1 (features 1 and 2) perfectly also with zero confusion. This result confirms that indeed our multi-view approach can discover multiple clustering solutions in different subspaces. Figure 4 shows scatter plots of the data. The left column ($(a1)$, $(a2)$, $(a3)$) is the plot for method 1. $(a1)$ shows the clustering in ellipses found by method 1 in iteration 1. The left sub-figure shows the groupings in the original features 1 and 2, and the data points are colored based on true labeling 1. The right sub-figure shows the clusterings in the original features 3 and 4, and the color of the data points are based on true labeling 2. $(a2)$ is the same scatter plot of the original data X with the clusters found by method 1 as shown by the ellipses in iteration 2. Similarly, $(b1)$ and $(b2)$ show the results of method 2. $(a3)$ and $(b3)$ are the SSE for the two methods in each iteration. Method 2 converges to zero much faster than method 1 here. Note that SSE monotonically decreases with iteration and that the algorithm captures most of the information in two clustering views. From these results, in iteration 1 we found the right partition based on features 3 and 4, but group the clusters in features 1 and 2 incorrectly. On the other hand, iteration 2 groups the clusters based on features 1 and 2 correctly, but the partition for the clusters in features 3 and 4 is wrong. The results confirm that indeed our multi-view approach can discover multiple clustering solutions in different subspaces.

Table 2. Confusion Matrix for Synthetic Data2

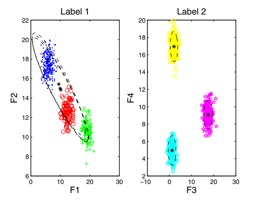
SYNTHETIC DATA2	METHOD1			METHOD2		
ITERATION 1						
LABELLING1	C1	C2	C3	C1	C2	C3
L1	**41**	40	19	**41**	40	19
L2	**44**	34	22	**44**	34	22
L3	115	**126**	59	115	**126**	59
LABELLING2	C1	C2	C3	C1	C2	C3
L1	**200**	0	0	**200**	0	0
L2	0	**200**	0	0	**200**	0
L3	0	0	**100**	0	0	**100**
ITERATION 2						
LABELLING1	C1	C2	C3	C1	C2	C3
L1	**100**	0	0	**100**	0	0
L2	0	**100**	0	0	**100**	0
L3	0	0	**300**	0	0	**300**
LABELLING2	C1	C2	C3	C1	C2	C3
L1	**126**	34	40	**126**	34	40
L2	**115**	44	41	**115**	44	41
L3	**59**	22	19	**59**	22	19

4.2 Experiments on Benchmark Data

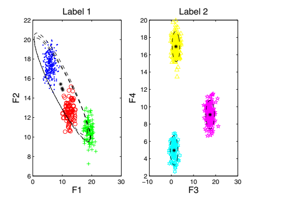
We have shown that our two methods work on synthetic data. Here, we investigate whether they reveal interesting and diverse clustering solutions on real benchmark data. We select data sets that have high-dimensionality and that have multiple possible partitioning.

In this section, we investigate the performance of our multi-view orthogonal clustering algorithms on four real world data sets, including the digits data set from the UCI machine learning repository [4], the face data set from the UCI KDD repository [2], and two text data sets: the mini-newsgroups data [2] and the WebKB data set [6].

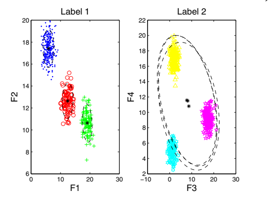
The digits data is a data set for an optical recognition problem of handwritten digits with ten classes, 5620 cases, and 64 attributes (all input attributes are integers from $0 \ldots 16$). The face data consists of 640 face images of 20 people taken with varying pose (straight, left, right, up), expression (neutral, happy, sad, angry), eyes (wearing sunglasses or not). Each person has 32 images capturing every combination of features. The image resolution is 32×30. We removed the missing data and formed a 960×624 data matrix. Each of the 960 features represents a pixel value. The mini-newsgroups data comes from the UCI KDD repository which contains 2000 articles from 20



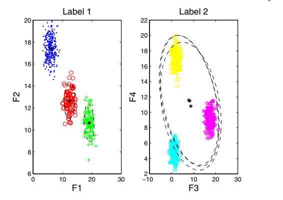
(a1.iteration1 for method1)



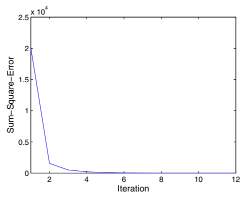
(b1.iteration1 for method2)



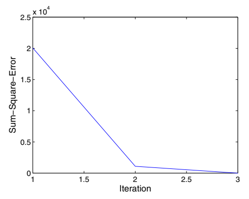
(a2.iteration2 for method1)



(b2.iteration2 for method2)



(a3.SSE for method1)



(b3.SSE for method2)

Figure 4. These are scatter plots of synthetic data 2 and the clusters found by methods 1 (a1, a2) and 2 (b1, b2). The color of the data points reflect different class labels and the ellipses represent the clusters found. a1, b1 are the results for iteration 1; a2, b2 are the results for iteration 2; a3 and b3 are SSE as a function of iteration for methods 1 and 2 respectively.

newsgroups. The second text data is the CMU four university WebKB data set as described in [6]. Both text data sets were processed following the standard procedure, including stemming and removing stopwords.

4.2.1 Results for the Digit Data

Table 3 shows the confusion matrix for the digit data for both methods 1 and 2. Because the results for methods 1 and 2 are similar, we will concentrate on the results from method 1. For both iterations, we partition the data into three clusters. In iteration 1, the resulting partition clustered digits $\{0, 4, 6\}$, $\{1,7,8\}$ and $\{2,3,5,9\}$ into different groups. In iteration 2, our method clustered $\{0,3,5,7,9\}$, $\{2,6,8,1\}$ and $\{4\}$ into another set of clusters. And, in iteration 3, the clusters we found are $\{0,1,8,9\}$, $\{2,3,6,7,4\}$ and $\{5\}$. These results show that in each iteration we can find a different way of partitioning the ten classes (digits).

In Figure 5, we present the mean image of each cluster obtained by method 1 in three iterations. Below each image we show the dominant digits contained in the cluster. For a digit to be considered as contained in a cluster, we require that at least 70% of its data points fall into the cluster. It is interesting to note that digits 4 and 5 were not well captured by any of the clusters in iteration 1. In contrast, in iteration 2, we see digit 4 well separated and captured by cluster 2. In iteration 3, we were able to capture digit 5 nicely in a single cluster. This further demonstrated that our method is capable of discovering multiple reasonable structures from data.

Table 3. Confusion Matrix for the Digits Data

DIGIT	METHOD1			METHOD2		
IT1	C1	C2	C3	C1	C2	C3
0	**550**	0	4	**550**	0	4
4	**371**	197	0	**369**	199	0
6	**555**	2	1	**555**	2	1
1	12	**477**	82	12	**477**	82
7	0	**566**	0	0	**566**	0
8	15	**321**	218	15	**321**	218
2	5	27	**525**	5	27	**525**
3	0	41	**531**	0	41	**531**
5	35	236	**287**	35	236	**287**
9	2	152	**408**	2	152	**408**
IT2	C1	C2	C3	C1	C2	C3
0	**547**	2	5	**548**	2	4
3	**380**	92	100	**348**	66	158
5	**326**	216	16	**322**	212	24
7	**492**	68	6	**492**	67	7
9	**470**	8	84	**458**	9	95
2	58	**490**	9	58	**488**	11
6	12	**539**	7	12	**532**	14
8	182	**354**	18	173	**350**	31
1	2	**308**	261	0	285	**286**
4	65	41	**462**	138	49	**381**
IT3	C1	C2	C3	C1	C2	C3
0	**529**	4	21	**545**	1	8
1	**478**	45	48	**394**	77	100
8	**484**	46	24	**317**	213	24
9	**381**	54	127	**265**	143	154
2	120	**397**	40	**522**	21	14
3	106	**454**	12	151	**393**	28
6	107	**407**	44	47	**486**	25
7	16	**536**	14	54	**500**	12
4	67	**275**	226	187	159	**222**
5	27	1	**530**	18	5	**535**

4.2.2 Results for the Face Data

Face data is a very interesting data set because it can be grouped in several different ways (e.g., by person, pose, etc.). We design the experiment to see if we can get different clustering information in different iterations.

First, we begin with our number of clusters $K = 20$ in the first iteration, hopefully to find the 20 persons in the

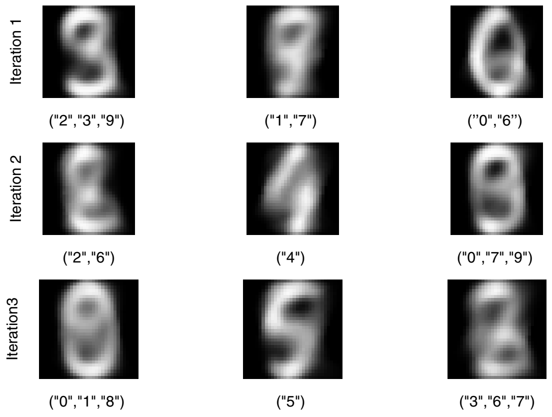


Figure 5. The average digit for images within each cluster found by method 1 in iterations/views 1, 2 and 3. These clustering views correspond to different digits.

database. Then, from the second iteration to the rest of the iterations, we set $K = 4$ to see if the partitions found in the remaining iterations can tell us any useful information. Figure 6 shows the average image for each cluster we find in iteration 1. We observed from this figure and from the confusion matrix (not shown due to space limitations and information clutter) that iteration 1 leads to a clustering corresponding to the different persons in the database.The number below the image is the percentage this person appears in the cluster. It is a confidence measure of the identification. And the images clearly show different persons. In the second iteration, the four clusters we found are shown in Figure 7. Each image is an average image of the images within each cluster. It is clear that the clustering in iteration 2 groups the data based on different poses. This again suggested that our method can find different clustering views from the data. The more independent the important partitions lie in the data, the better are the results of our method. Since method 2 gave us similar images, we only provided the results by method 1 here due to space limitation.

4.2.3 Results for the Mini-Newsgroups Data

The mini-newsgroups data set originally contains 20 classes. We removed the classes that are under the "misc" category because it does not correspond to a clear concept class. We also pre-processed the data to remove stop words, words that appeared in less than 40 documents, and the words that had low variance of occurrence in all documents. After the pre-processing, the data contains 1700 documents from 17 classes. Each document is represented by a 500-dimensional term frequency vector.

Note that PCA followed by k-means does not work well

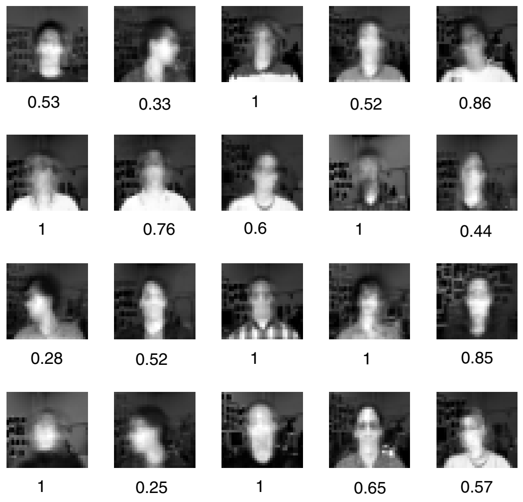


Figure 6. The average face image for each cluster found by method 1 in iteration 1. This clustering view corresponds to different persons.

for text data. Here, we apply the spherical k-means method [7] instead, which considers the correlation between documents rather than the Euclidean distance. Our experiments showed that this method provided a reasonable clustering of the text data sets.

Table 4. Confusion Matrix for WebKB Data

ITERATION 1	C1	C2	C3	C4
COURSE	**134**	12	81	17
FACULTY	2	**78**	61	12
PROJECT	1	**47**	28	10
STUDENT	2	68	**402**	86
ITERATION 2	C1	C2	C3	C4
CORNELL	**103**	86	27	10
TEXAS	50	**87**	83	32
WASHINGTON	35	77	**138**	5
WISCONSIN	60	86	30	**132**

Table 5 shows the confusion matrices by method 1 for three iterations. For the first iteration, we set $K = 3$. The results show that cluster $C1$ groups together recreation and computer categories. The ten most frequent words tell us that the documents here share information related to entertainment. Cluster $C2$ groups science and talks together, and the frequent words confirm that it groups science and the religion part of the talk. Cluster $C3$ is a mixture of different topics.

In iteration 2, we set $K = 4$ to see if it we can partition the data to capture the four categories "computer", "recre-

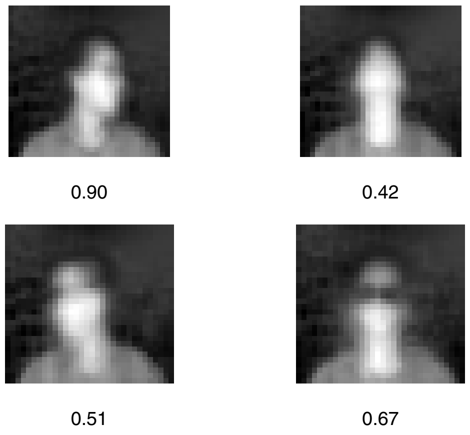


Figure 7. The average face image for each cluster found by method 1 in iteration 2. This clustering view corresponds to different poses.

ation", "talk" and "science". From the confusion matrix, we see that we were able to find these high level categories. $C1$ is about computers; $C2$ contains news about recreation; and $C3$ groups those files related to science. The last one $C4$ contains documents from the "talk" category that are related to politics.

In iteration 3, two of the "computer" classes (graphics, os.ms) were grouped together with the "talk" category, the remaining three "computer" classes were grouped together with the "recreation" category (auto, motorcycles and sports). This suggests that our method continued finding clustering structure that is different from the existing results.

4.2.4 Results for the WebKB Text Data

This data contains 1041 html documents, from four webpage topics: course, faculty, project and student. Alternatively, the webpage can also be grouped based on their regions/universities, which include four universities: Cornell University, University of Texas Austin, University of Washington and Wisconsin Madison. Following the same preprocessing procedure used for the mini-newsgroups data, we removed the rare words, stop words, and words with low variances. Finally, we obtained 350 words in the vocabulary. The final data matrix is of size 350×1041.

The experimental results are quite interesting. For the first iteration, we see our method found the partition that mostly corresponds to the different topics, which can be seen in Table 4. Cluster 1 contains course webpages, cluster 2 is a mix of faculty and project pages, cluster 3 consists of a majority of student webpages. In the second iteration, our method found a different clustering that corresponds to the universities, as shown in Table 4.

5 Conclusion

The goal of explorative data analysis is to find the underlying structure from a given set of data, which may be multi-faceted by nature. Existing work on non-redundant clustering attempts to address this problem by searching for a single alternative clustering solution that is different from an existing one. Our main contribution in this paper is that *we introduced a new paradigm for explorative data clustering that seeks to extract all non-redundant clustering views from a given set of data (until there is only noise left in the data).*

We presented a general framework for extracting multiple clustering views from high dimensional data. In essence, this framework works by incorporating orthogonality constraints into a clustering algorithm. In other words, the clustering algorithm will search for a new clustering in a space that is orthogonal to what has been covered by existing clustering solutions. We described two different methods for introducing orthogonality and conducted a variety of experiments on both synthetic data and real world benchmark data sets to evaluate these methods. The results can be summarized as follows.

1. Using two different synthetic data sets, our proposed framework was able to find different substructures of the data or different structures embedded in different subspaces in different iterations.

2. On benchmark data sets, our methods not only found different clustering structures in different iterations, but also discovered clustering structures that are sensible, judging from the various evaluation criteria reported (such as, confusion matrices and scatter plots). The face data set for example, PCA+K-means identified individuals in the first iteration. In the second iteration, our methods were able to identify a set of different clusters that correspond nicely to different poses. For other data sets, we observed similar results in the sense that different concept classes were identified in different iterations.

Note that this paper uses k-means as the basic clustering algorithm. However, the framework is designed with no specific algorithm in mind and can work with any clustering algorithm. Future directions will be to explore the framework with other clustering methods.

Acknowledgments

This research is supported by NSF CAREER IIS-0347532.

References

[1] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. In *Proc. of the ACM SIGMOD Int'l Conf. on Management of Data*, pages 94–105, 1998.

[2] S. D. Bay. The UCI KDD archive, 1999.

[3] S. Bickel and T. Scheffer. Multi-view clustering. In *Proc. of the IEEE Int'l Conf. on Data Mining*, pages 19–26, 2004.

[4] C. Blake and C. Merz. UCI repository of machine learning databases. In *http://www.ics.uci.edu/∼mlearn/MLRepository.html*, 1998.

[5] G. Chechik and N. Tishby. Extracting relevant structures with side information. In *Advances in Neural Information Processing Systems 15 (NIPS-2002)*, 2003.

[6] CMU. CMU 4 universities WebKB data, 1997.

[7] I. S. Dhillon and D. M. Modha. Concept decompositions for large sparse text data using clustering. *Machine Learning*, 42(1):143–175, 2001.

[8] C. Ding, X. He, H. Zha, and H. Simon. Adaptive dimension reduction for clustering high dimensional data. In *Proc. of the IEEE Int'l Conf. on Data Mining*, pages 147–154, 2002.

[9] R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis*. Wiley & Sons, NY, 1973.

[10] X. Z. Fern and C. E. Brodley. Random projection for high dimensional data clustering: A cluster ensemble approach. In *Proc. of the Int'l Conf. on Machine Learning*, 2003.

[11] E. Forgy. Cluster analysis of multivariate data: Efficiency vs. interpretability of classifications. *Biometrics*, 21:768, 1965.

[12] K. Fukunaga. *Statistical Pattern Recognition (second edition)*. Academic Press, San Diego, CA, 1990.

[13] D. Gondek. *Non-redundant clustering*. PhD thesis, Brown University, 2005.

[14] D. Gondek and T. Hofmann. Conditional information bottleneck clustering. In *The 3rd IEEE Intl. Conf. on Data Mining, Workshop on Clustering Large Data Sets*, 2003.

[15] D. Gondek and T. Hofmann. Non-redundant data clustering. In *Proc. of the 4th Intl. Conf. on Data Mining*, 2004.

[16] D. Gondek and T. Hofmann. Non-redundant clustering with conditional ensembles. In *Proc. of the 11th ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining (KDD'05)*, pages 70–77, 2005.

[17] D. Gondek, S. Vaithyanathan, and A. Garg. Clustering with model-level constraints. In *Proc. of SIAM International Conference on Data Mining*, 2005.

[18] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: A review. *ACM Computing Surveys*, 31(3):264–323, 1999.

[19] I. T. Jolliffe. *Principal Component Analysis*. Springer-Verlag, New-York, 1986.

[20] J. Macqueen. Some methods for classifications and analysis of multivariate observations. *Proc. Symp. Mathematical Statistics and Probability, 5th, Berkeley*, 1:281–297, 1967.

[21] L. Parsons, E. Haque, and H. Liu. Subspace clustering for high dimensional data: a review. *SIGKDD Explor. Newsl.*, 6(1):90–105, 2004.

[22] A. Strehl and J. Ghosh. Cluster ensembles – a knowledge reuse framework for combining multiple partitions. *Journal of Machine Learning Research*, pages 583–617, 2002.

Table 5. Confusion Matrix for the Mini-Newsgroups Data

ITERATION1 (K=3)	C1	C2	C3
COMP.GRAPHICS	**88**	0	12
COMP.OS.MS	**95**	0	5
COMP.SYS.IBM.PC.HARDWARE	**94**	0	6
COMP.SYS.MAC.HARDWARE	**88**	0	12
COMP.WINDOWS.X	**87**	0	13
REC.AUTOS	**81**	0	19
REC.MOTORCYCLES	**82**	0	18
REC.SPORT.BASEBALL	**81**	0	19
REC.SPORT.HOCKEY	**71**	2	27
SCI.CRYPT	0	**68**	32
SCI.ELECTRONICS	0	**76**	24
SCI.MED	0	**78**	22
SCI.SPACE	0	**74**	26
TALK.POLITICS.GUNS	0	**70**	30
TALK.POLITICS.MIDEAST	0	**61**	39
TALK.POLITICS.MISC	0	**72**	28
TALK.RELIGION.MISC	0	**77**	23

ITERATION2 (K=4)	C1	C2	C3	C4
COMP.GRAPHICS	**98**	0	2	0
COMP.OS.MS	**94**	0	0	6
COMP.SYS.IBM.PC.HARDWARE	**78**	15	3	4
COMP.SYS.MAC.HARDWARE	**66**	20	3	11
COMP.WINDOWS.X	**43**	39	5	13
REC.AUTOS	28	**51**	6	15
REC.MOTORCYCLES	17	**59**	6	18
REC.SPORT.BASEBALL	10	**67**	4	19
REC.SPORT.HOCKEY	5	**62**	4	29
SCI.CRYPT	5	8	**57**	30
SCI.ELECTRONICS	1	9	**65**	25
SCI.MED	1	22	**61**	16
SCI.SPACE	0	16	**58**	26
TALK.POLITICS.GUNS	0	37	20	**43**
TALK.POLITICS.MIDEAST	5	39	11	**45**
TALK.POLITICS.MISC	3	45	6	**46**
TALK.RELIGION.MISC	1	**58**	3	38

ITERATION3 (K=4)	C1	C2	C3	C4
COMP.GRAPHICS	**33**	32	6	29
COMP.OS.MS	**42**	23	10	25
COMP.SYS.IBM.PC.HARDWARE	17	**45**	11	27
COMP.SYS.MAC.HARDWARE	15	**41**	20	24
COMP.WINDOWS.X	19	**40**	18	23
REC.AUTOS	15	**47**	27	11
REC.MOTORCYCLES	10	**54**	22	14
REC.SPORT.BASEBALL	7	**51**	33	9
REC.SPORT.HOCKEY	5	**66**	21	8
SCI.CRYPT	5	15	**68**	12
SCI.ELECTRONICS	10	9	**65**	16
SCI.MED	31	8	**46**	15
SCI.SPACE	15	24	**48**	13
TALK.POLITICS.GUNS	**49**	19	18	14
TALK.POLITICS.MIDEAST	**45**	24	16	15
TALK.POLITICS.MISC	**55**	12	12	21
TALK.RELIGION.MISC	**56**	8	20	16