Chapter 1
Discriminative Cluster Analysis

Fernando De la Torre and Takeo Kanade

Abstract *Clustering is one of the most widely used statistical tools for data analysis. Among all existing clustering techniques, k-means is a very popular method due to its ease of programming and its good trade-off between achieved performance and computational complexity. However, k-means is prone to local minima problems and does not scale well with high dimensional data sets. A common approach to clustering high dimensional data is to project in the space spanned by the principal components (PC). However, the space of PCs does not necessarily improve the separability of the clusters. In this paper, we propose Discriminative Cluster Analysis (DCA) that clusters data in a low dimensional discriminative that encourages cluster separability. DCA simultaneously performs dimensionality reduction and clustering, improving efficiency and cluster performance in comparison with generative approaches (e.g. PC). We exemplify the benefits of DCA versus traditional PCA+k-means clustering through several synthetic and real examples. Additionally, we provide connections with other dimensionality reduction and clustering techniques such as spectral graph methods and linear discriminant analysis.*

1.1 Introduction

Clustering is one of the most widely used statistical methods in data analysis (e.g. multimedia content-based retrieval, molecular biology, text mining, bioinformatics). Recently, with an increasing number of database applications that deal with very large high dimensional datasets, clustering has emerged as a very important research area in many disciplines. Unfortunately, many known algorithms tend to break down in high dimensional spaces because of the sparsity of the points. In

Robotics Institute, Carnegie Mellon University
5000 Forbes Avenue Pittsburgh, PA 15213
Contact author: Fernando De la Torre 211 Smith Hall
Tel.: 412-268-4708 Fax: 412-268-5571 e-mail: ftorre@cs.cmu.edu

such high dimensional spaces not all the dimensions might be relevant for clustering, outliers are difficult to detect, and the curse of dimensionality makes clustering a challenging problem. Also, when handling large amounts of data, time complexity becomes a limiting factor.

There are two types of clustering algorithms: partitional and hierarchical (**?**). Partitional methods (e.g. k-means, mixture of Gaussians, graph theoretic, mode seeking) only produce one partition of the data; whereas hierarchical ones (e.g single link, complete link) produce several of them. In particular, k-means (**?**) is one of the simplest unsupervised learning algorithms that has been extensively studied and extended (**?**). Although it is a widely used technique due to its ease of programming and good performance, k-means suffers from several drawbacks. It is sensitive to initial conditions, it does not remove undesirable features for clustering, and it is optimal only for hyper-spherical clusters. Furthermore, its complexity in time is $O(nkl)$ and in space is $O(k)$, where n is the number of samples, k is the number of clusters, and l the number of iterations. This degree of complexity can be impractical for large datasets.

To partially address some of these challenges, this papers proposes Discriminative Cluster Analysis (DCA). DCA jointly performs clustering and dimensionality reduction. In the first step, DCA finds a low dimensional projection of the data well suited for clustering by encouraging preservation of distances between neighboring data points belonging to the same class. Once the data is projected into a low dimensional space, DCA performs a "soft" clustering of the data. Later, this information is feedback into the dimensionality reduction step until convergence. Clustering in the DCA subspace is less prone to local minima, noisy dimensions that are irrelevant for clustering are removed, and clustering is faster to compute (especially for high dimensional data). Recently, other researchers (**?**), (**?**) have explored further advantages of discriminative clustering methods versus generative approaches.

1.2 Previous work

This section reviews previous work on k-means, spectral methods for clustering, and linear discriminant analysis in a unified framework.

1.2.1 k-means and spectral graph methods: a unified framework

k-means (**?**; **?**) is one of the simplest and most popular unsupervised learning algorithms used to solve the clustering problem. Clustering refers to the partition of n data points into c disjoint clusters. k-means clustering splits a set of n objects into c groups by maximizing the between-cluster variation relative to within-cluster variation. In other words, k-means clustering finds the partition of the data that is a local optimum of the following energy function:

$$J(\mathbf{m}_1,...,\mathbf{m}_c) = \sum_{i=1}^{c} \sum_{j \in C_i} ||\mathbf{d}_j - \mathbf{m}_i||_2^2 \tag{1.1}$$

where $\mathbf{d}_j$ (see notation [1]) is a vector representing the j^{th} data point and $\mathbf{m}_i$ is the geometric centroid of the data points for class i. The optimization criterion in eq. (1.1) can be rewritten in matrix form as:

$$E_1(\mathbf{M},\mathbf{G}) = ||\mathbf{D} - \mathbf{M}\mathbf{G}^T||_F \;\; subject\ to\ \mathbf{G}\mathbf{1}_c = \mathbf{1}_n \;\; and \;\; g_{ij} \in \{0,1\} \tag{1.2}$$

where $\mathbf{G}$ is an indicator matrix, such that $\sum_j g_{ij} = 1$, $g_{ij} \in \{0,1\}$ and g_{ij} is 1 if $\mathbf{d}_i$ belongs to class j, c denotes the number of classes and n is the number of samples. $\mathbf{M} \in \Re^{d \times c}$ is the matrix containing all the means for each cluster. The columns of $\mathbf{D} \in \Re^{d \times n}$ contain the original data points, and d is the number of features. The equivalence between the k-means error function of eq. (1.1) and eq. (1.2) is only valid if $\mathbf{G}$ strictly satisfies the constraints.

The k-means algorithm performs coordinate descent in $E_1(\mathbf{M},\mathbf{G})$. Given the actual value of the means $\mathbf{M}$, the first step finds, for each data point $\mathbf{d}_j$, the value of $\mathbf{g}^j$ minimizing eq. (1.2) subject to the constraints. The second step optimizes $\mathbf{M} = \mathbf{D}\mathbf{G}(\mathbf{G}^T\mathbf{G})^{-1}$, which effectively computes the mean of each cluster. Although it can be proven that alternating these two steps will always converge, the k-means algorithm does not necessarily find the optimal configuration of all possible assignments. The algorithm is significantly sensitive to the initial randomly selected cluster centers. It is typically run multiple times, and the solution with less error is chosen. Despite these limitations, the algorithm is used frequently as a result of its easiness of implementation and effectiveness.

After optimizing over $\mathbf{M}$, eq. (1.2), can be rewritten as:

$$\begin{aligned} E_2(\mathbf{G}) &= ||\mathbf{D} - \mathbf{D}\mathbf{G}(\mathbf{G}^T\mathbf{G})^{-1}\mathbf{G}^T||_F = tr(\mathbf{D}^T\mathbf{D}) \\ &-tr((\mathbf{G}^T\mathbf{G})^{-1}\mathbf{G}^T\mathbf{D}^T\mathbf{D}\mathbf{G}) \geq \textstyle\sum_{i=c+1}^{min(d,n)} \lambda_i \end{aligned} \tag{1.3}$$

where λ_i are the eigenvalues of $\mathbf{D}^T\mathbf{D}$. Minimizing $E_2(\mathbf{G})$, eq. (1.3), is equivalent to maximizing $tr((\mathbf{G}^T\mathbf{G})^{-1}\mathbf{G}^T\mathbf{D}^T\mathbf{D}\mathbf{G})$. Ignoring the special structure of $\mathbf{G}$ and considering the continuous domain, the optimum $\mathbf{G}$ value is given by the eigenvectors of the Gram matrix $\mathbf{D}^T\mathbf{D}$. The error of E_2 with the optimal continuous $\mathbf{G}$ is $E_2 = \sum_{i=c+1}^{min(d,n)} \lambda_i$. A similar reasoning has been reported by (**?**; **?**), demonstrating that a lower bound of $E_2(\mathbf{G})$, eq. (1.3), is given by the sum of residual eigenvalues.

[1] Bold capital letters denote matrices $\mathbf{D}$, and bold lower-case letters signify a column vector $\mathbf{d}$. $\mathbf{d}_j$ represents the j^{th} column of the matrix $\mathbf{D}$. $\mathbf{d}^j$ is a column vector that designates the j-th row of the matrix $\mathbf{D}$. All non-bold letters refer to scalar variables. d_{ij} corresponds to the scalar in the row i and column j of the matrix $\mathbf{D}$, as well as the i-th element of a column vector $\mathbf{d}_j$. *diag* is an operator that transforms a vector into a diagonal matrix or transforms the diagonal of a matrix into a vector. *vec* vectorizes a matrix into a vector. $\mathbf{1}_k \in \Re^{k \times 1}$ is a vector of ones. $\mathbf{I}_k \in \Re^{k \times k}$ denotes the identity matrix. $||\mathbf{d}||_2^2$ denotes the norm of the vector $\mathbf{d}$. $tr(\mathbf{A}) = \sum_i a_{ii}$ is the trace of the matrix $\mathbf{A}$, and $|\mathbf{A}|$ denotes the determinant. $||\mathbf{A}||_F = tr(\mathbf{A}^T\mathbf{A}) = tr(\mathbf{A}\mathbf{A}^T)$ designates the Frobenious norm of matrix $\mathbf{A}$. $N_d(\mathbf{x};\mu,\Sigma)$ indicates a d-dimensional Gaussian on the variable $\mathbf{x}$ with mean μ and covariance Σ. $\circ$ denotes the Hadamard or point-wise product.

The continuous solution of $\mathbf{G}$ lies in the $c-1$ subspace, spanned by the first $c-1$ eigenvectors with highest eigenvalues (**?**) of $\mathbf{D}^T\mathbf{D}$.

Finally, it is worthwhile to point out the connections between k-means and standard spectral graph algorithms (**?**), such as Normalized Cuts (**?**), by means of kernel methods. The kernel trick is a standard method for lifting the points of a dataset to a higher dimensional space, where points are more likely to be linearly separable (assuming that the correct mapping is found). Consider a lifting of the original points to a higher dimensional space, $\Gamma = [\ \phi(\mathbf{d}_1)\ \phi(\mathbf{d}_2)\ \cdots\ \phi(\mathbf{d}_n)\ ]$ where ϕ represents a high dimensional mapping. The kernelized version of eq. (1.2) is:

$$E_3(\mathbf{M},\mathbf{G}) = ||(\Gamma - \mathbf{M}\mathbf{G}^T)\mathbf{W}||_F \tag{1.4}$$

in which we introduce a weighting matrix $\mathbf{W}$ for normalization purposes. Eliminating $\mathbf{M} = \Gamma\mathbf{W}\mathbf{W}^T\mathbf{G}(\mathbf{G}^T\mathbf{W}\mathbf{W}^T\mathbf{G})^{-1}$, it can be shown that:

$$E_3 \propto tr((\mathbf{G}^T\mathbf{W}\mathbf{W}^T\mathbf{G})^{-1}\mathbf{G}^T\mathbf{W}\mathbf{W}^T\Gamma^T\Gamma\mathbf{W}\mathbf{W}^T\mathbf{G}) \tag{1.5}$$

where $\Gamma^T\Gamma$ is the standard affinity matrix in Normalized Cuts (**?**). After a change of variable $\mathbf{Z} = \mathbf{G}^T\mathbf{W}$, the previous equation can be expressed as $E_3(\mathbf{Z}) \propto tr((\mathbf{Z}\mathbf{Z}^T)^{-1}\mathbf{Z}\mathbf{W}^T\Gamma^T\Gamma\mathbf{W}\mathbf{Z}^T)$. Choosing $\mathbf{W} = diag(\Gamma^T\Gamma\mathbf{1}_n)^{-\frac{1}{2}}$ the problem is equivalent to solving the Normalized Cuts problem. This formulation is more general since it allows for arbitrary kernels and weights. In addition, the weight matrix can be used to reject the influence of pairs of data points with unknown similarity (i.e. missing data).

1.2.2 Linear Discriminant Analysis

The aim of LDA is to find a low dimensional projection, where the means of the classes are as far as possible from each other, and the intra-class variation is small. LDA can be computed in closed form using the following covariance matrices, conveniently expressed in matrix form (**?**):

$$\begin{aligned} f\mathbf{S}_t &= \sum_{j=1}^{n}(\mathbf{d}_j - \mathbf{m})(\mathbf{d}_j - \mathbf{m})^T = \mathbf{D}\mathbf{P}_1\mathbf{D}^T \\ f\mathbf{S}_w &= \sum_{i=1}^{c}\sum_{\mathbf{d}_j \in C_i}(\mathbf{d}_j - \mathbf{m}_i)(\mathbf{d}_j - \mathbf{m}_i)^T = \mathbf{D}\mathbf{P}_2\mathbf{D}^T \\ f\mathbf{S}_b &= \sum_{i=1}^{c} n_i(\mathbf{m}_i - \mathbf{m})(\mathbf{m}_i - \mathbf{m})^T = \mathbf{D}\mathbf{P}_3\mathbf{D}^T \end{aligned}$$

where $f = n-1$, and $\mathbf{P}_i$'s are projection matrices (i.e. $\mathbf{P}_i^T = \mathbf{P}_i$ and $\mathbf{P}_i^2 = \mathbf{P}_i$) with the following expressions:

$$\mathbf{P}_1 = \mathbf{I} - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^T \quad \mathbf{P}_2 = \mathbf{I} - \mathbf{G}(\mathbf{G}^T\mathbf{G})^{-1}\mathbf{G}^T \quad \mathbf{P}_3 = \mathbf{G}(\mathbf{G}^T\mathbf{G})^{-1}\mathbf{G}^T - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^T \tag{1.6}$$

$\mathbf{S}_b$ is the between-class covariance matrix and represents the average distance between the mean of the classes. $\mathbf{S}_w$ is the within-class covariance matrix and it is a measure of the average compactness of each class. Finally, $\mathbf{S}_t$ is the total covariance matrix. Through these matrix expressions, it can be easily verified that $\mathbf{S}_t = \mathbf{S}_w + \mathbf{S}_b$. The upper bounds on the ranks of the matrices are $min(c-1,d)$, $min(n-c,d)$, $min(n-1,d)$ for $\mathbf{S}_b, \mathbf{S}_w$, and $\mathbf{S}_t$ respectively.

LDA computes a linear transformation of the data $\mathbf{B} \in \Re^{d \times k}$ that maximizes the distance between class means and minimizes the variance within clusters. Rayleigh-like quotients are among the most popular LDA optimization criterion (**?**). For instance, LDA can be obtained by minimizing:

$$E_2(\mathbf{B}) = tr((\mathbf{B}^T\mathbf{S}_1\mathbf{B})^{-1}\mathbf{B}^T\mathbf{S}_2\mathbf{B}) \tag{1.7}$$

where several combinations of $\mathbf{S}_1$ and $\mathbf{S}_2$ matrices lead to the same LDA solution (e.g. $\mathbf{S}_1 = \{\mathbf{S}_b, \mathbf{S}_b, \mathbf{S}_t\}$ and $\mathbf{S}_2 = \{\mathbf{S}_w, \mathbf{S}_t, \mathbf{S}_w\}$). The Rayleigh quotient of eq.(1.7) has a closed-form solution in terms of a Generalized Eigenvalue Problem (GEP), $\mathbf{S}_2\mathbf{B} = \mathbf{S}_1\mathbf{B}\Lambda$ (**?**). In the case of high-dimensional data (e.g. images) the covariance matrices are not likely to be full rank due to the lack of training samples and alternative approaches to compute LDA are needed. This is the well-known small sample size (SSS) problem. There are many techniques to solve the GEP when $\mathbf{S}_1$ and $\mathbf{S}_2$ are rank deficient, see (**?**; **?**) for a recent review. However, solving LDA with standard eigensolvers is not efficient (neither space or nor time) for large amounts of high dimensional data. Formulating LDA as a least-squares problem suggests efficient methods to solve LDA techniques. Moreover, a least-squares formulation of LDA facilitates its analysis and generalization.

Consider the following weighted between-class covariance matrix $\hat{\mathbf{S}}_b = \mathbf{D}\mathbf{G}\mathbf{G}^T\mathbf{D}^T = \sum_{i=1}^{c}(\frac{n_i}{n})^2\mathbf{m}_i\mathbf{m}_i^T$, that favors classes with more samples. $\mathbf{m}_i$ is the mean vector for class i, and we assume zero mean data (i.e. $\mathbf{m} = \frac{1}{n}\mathbf{D}\mathbf{1}_n$). Previous work on neural networks (**?**; **?**) have shown that maximizing $J_4(\mathbf{B}) = tr((\mathbf{B}^T\hat{\mathbf{S}}_b\mathbf{B})(\mathbf{B}^T\mathbf{S}_t\mathbf{B})^{-1})$ is equivalent to minimizing:

$$E_4(\mathbf{B},\mathbf{V}) = ||\mathbf{G}^T - \mathbf{V}\mathbf{B}^T\mathbf{D}||_F \propto -tr((\mathbf{B}^T\mathbf{D}\mathbf{D}^T\mathbf{B})^{-1}\mathbf{B}^T\mathbf{D}\mathbf{G}\mathbf{G}^T\mathbf{D}^T\mathbf{B}) \tag{1.8}$$

This approach is attractive since (**?**) have shown that the surface of eq. (1.8) has a unique local minima, and several saddle points.

1.3 Discriminative Cluster Analysis

In the previous section, we have provided a least-squares framework for LDA (supervised dimensionality reduction) and k-means (unsupervised clustering). The aim of DCA is to combine clustering and dimensionality reduction in an unsupervised manner. In this section, we propose a least-squares formulation for DCA.

1.3.1 Error function for LDA and DCA

The key aspect to simultaneously performing dimensionality reduction and clustering is the analysis of eq. (1.8). Ideally we would like to optimize eq. (1.8) w.r.t. $\mathbf{B}$ and $\mathbf{G}$. However, directly optimizing eq. (1.8) has several drawbacks. First, eq. (1.8) biases the solution towards classes that have more samples because it maximizes $\hat{\mathbf{S}}_b = \mathbf{D}\mathbf{G}\mathbf{G}^T\mathbf{D}^T = \sum_{i=1}^{c}(\frac{n_i}{n})^2(\mathbf{m}_i)(\mathbf{m}_i)^T$. Secondly, eq. (1.8) does not encourage sparseness in $\mathbf{G}$ if $g_{ij} > 0$. That is, assuming that $\mathbf{C} = \mathbf{B}^T\mathbf{D} \in \Re^{k\times n}$, then eq. (1.8) is equivalent to $E_4 = tr(\mathbf{G}^T\mathbf{G}) - tr(\mathbf{G}^T\mathbf{C}^T(\mathbf{C}\mathbf{C}^T)^{-1}\mathbf{C}\mathbf{G})$. If $g_{ij}\ \ \forall\, i,j$ is positive, minimizing the first term, $tr(\mathbf{G}^T\mathbf{G})$, does not encourage sparseness in $\mathbf{g}^i\ \forall i$ ($\mathbf{g}^i$ represents the i^{th} row of $\mathbf{G}$, see notation).

In this section, we correct eq. (1.8) to obtain the unbiased LDA criterion by normalizing E_4 as follows:

$$E_5(\mathbf{B},\mathbf{V},\mathbf{G}) = ||(\mathbf{G}^T\mathbf{G})^{-\frac{1}{2}}(\mathbf{G}^T - \mathbf{V}\mathbf{B}^T\mathbf{D})||_F \tag{1.9}$$

where $(\mathbf{G}^T\mathbf{G})^{-\frac{1}{2}}$ is the normalization factor. After eliminating $\mathbf{V}$, eq. (1.9) can be written as:

$$\begin{aligned} E_5(\mathbf{B},\mathbf{G}) &= ||(\mathbf{G}^T\mathbf{G})^{-\frac{1}{2}}\mathbf{G}^T(\mathbf{I}_n - \mathbf{C}^T(\mathbf{C}\mathbf{C}^T)^{-1}\mathbf{C})||_F \\ &\propto tr((\mathbf{B}^T\underbrace{\mathbf{D}\mathbf{D}^T}_{f\mathbf{S}_t}\mathbf{B})^{-1}\mathbf{B}^T\underbrace{\mathbf{D}\mathbf{G}(\mathbf{G}^T\mathbf{G})^{-1}\mathbf{G}^T\mathbf{D}^T}_{f\mathbf{S}_b}\mathbf{B}) \end{aligned} \tag{1.10}$$

If $\mathbf{G}$ is known, eq. (1.10) is the exact expression for LDA.

Eq. (1.10) is also the basis for DCA. unlike LDA, DCA is an unsupervised technique and $\mathbf{G}$ will consider a variable to optimize, subject to the constraints that $g_{ij} \in \{0,1\}$, and $\mathbf{G}\mathbf{1}_c = \mathbf{1}_n$. DCA jointly optimizes the data projection matrix $\mathbf{B}$ and the indicator matrix $\mathbf{G}$.

1.3.2 Updating $\mathbf{B}$

The optimal $\mathbf{B}$ given $\mathbf{G}$ can be computed in closed form by solving the following GEP:

$$\mathbf{D}\mathbf{R}\mathbf{D}^T\mathbf{B} = \mathbf{D}\mathbf{D}^T\mathbf{B}\Lambda_1 \quad where \quad \mathbf{R} = \mathbf{G}(\mathbf{G}^T\mathbf{G})^{-1}\mathbf{G}^T \tag{1.11}$$

There are many methods for efficiently solving the GEP in the case of high-dimensional data when ($d >> n$) (**?**; **?**; **?**). In this section, we propose a regularized stable closed form solution. Assuming $\mathbf{D}^T\mathbf{D}$ is full rank, computing $(\mathbf{D}^T\mathbf{D})^{-1}$ can be a numerically unstable process, especially if $\mathbf{D}^T\mathbf{D}$ has eigenvalues close to zero. A common method to solve ill-conditioning is to regularize the solution by factorizing $\Sigma = \mathbf{D}^T\mathbf{D}$ as the sum of the outer products plus a scaled identity matrix, i.e. $\Sigma \approx \mathbf{V}\Lambda\mathbf{V}^T + \sigma^2\mathbf{I}_d$. $\mathbf{V} \in \Re^{n\times k}$, $\Lambda \in \Re^{k\times k}$ is a diagonal matrix. The parameters σ^2, $\mathbf{V}$

and Λ are estimated by minimizing:

$$E_c(\mathbf{V},\Lambda,\sigma^2) = ||\Sigma - \mathbf{V}\Lambda\mathbf{V}^T - \sigma^2\mathbf{I}_n||_F \tag{1.12}$$

After optimizing over $\mathbf{V},\Lambda,\sigma^2$, it can be shown (**?**) that: $\sigma^2 = tr(\Sigma - \mathbf{V}\hat{\Lambda}\mathbf{V}^T)/d-k$, $\Lambda = \hat{\Lambda} - \sigma^2\mathbf{I}_d$, where $\hat{\Lambda}$ is a matrix containing the eigenvalues of the covariance matrix Σ and $\mathbf{V}$ the eigenvectors. This expression is equivalent to probabilistic PCA (**?**; **?**; **?**). After the factorization, the matrix inversion lemma (**?**) $(\mathbf{A}^{-1}+\mathbf{V}\mathbf{C}^{-1}\mathbf{V}^T)^{-1} = \mathbf{A}-\mathbf{A}\mathbf{V}(\mathbf{C}+\mathbf{V}^T\mathbf{A}\mathbf{V})^{-1}\mathbf{V}^T\mathbf{A}$ is applied to invert $(\mathbf{V}\Lambda\mathbf{V}^T+\sigma^2\mathbf{I}_n)^{-1}$, which results in:

$$(\mathbf{V}\Lambda\mathbf{V}^T+\sigma^2\mathbf{I}_n)^{-1} = \frac{1}{\sigma^2}(\mathbf{I}_n - \frac{1}{\sigma^2}\mathbf{V}(\Lambda^{-1}+\frac{\mathbf{I}_n}{\sigma^2})^{-1}\mathbf{V}^T)$$

Now, solving $(\mathbf{I}_n - \frac{1}{\sigma^2}\mathbf{V}(\Lambda^{-1}+\frac{\mathbf{I}_n}{\sigma^2})^{-1}\mathbf{V}^T)\mathbf{R}\mathbf{D}^T\mathbf{D}\alpha = \alpha\Lambda$ becomes a better conditioned problem.

The number of bases (k) are bounded by the number of classes (c), because the $rank(\mathbf{D}\mathbf{R}\mathbf{D}^T) = c$. We typically choose $c-1$ to be consistent with LDA. Moreover, the best clustering results are achieved by projecting the data into a space of $c-1$ dimensions. Also, observe that there is an ambiguity in the result, because for any invertible matrix $\mathbf{T}_1 \in \mathbf{R}^{k\times k}$, $E_5(\mathbf{B}) = E_5(\mathbf{B}\mathbf{T}_1)$.

1.3.3 Optimizing $\mathbf{G}$

Let $\mathbf{A} = \mathbf{C}^T(\mathbf{C}\mathbf{C}^T)^{-1}\mathbf{C} \in \Re^{n\times n}$, where $\mathbf{C} = \mathbf{B}^T\mathbf{D}$, then eq. (1.10) can be rewritten as:

$$E_5(\mathbf{G}) \propto tr((\mathbf{G}^T\mathbf{G})^{-1}\mathbf{G}^T\mathbf{A}\mathbf{G}) \tag{1.13}$$

Optimizing eq. (1.13) subject to $g_{ij} \in \{0,1\}$ and $\mathbf{G}\mathbf{1}_c = \mathbf{1}_n$ is an *NP* complete problem. To make it tractable, we relax the discrete constraint on g_{ij} allowing to take values in the range $(0,1)$. To use a gradient descent search mechanism, we parameterize $\mathbf{G}$ as the Hadamard (pointwise) product of two matrices $\mathbf{G} = \mathbf{V}\circ\mathbf{V}$ (**?**), and use the following updating scheme:

$$\begin{aligned} \mathbf{V}^{n+1} &= \mathbf{V}^n - \eta\frac{\partial E_5(\mathbf{G})}{\partial\mathbf{V}} \\ \frac{\partial E_5(\mathbf{G})}{\partial\mathbf{V}} &= (\mathbf{I}_c - \mathbf{G}(\mathbf{G}^T\mathbf{G})^{-1}\mathbf{G}^T)\mathbf{A}\mathbf{G}(\mathbf{G}^T\mathbf{G})^{-1}\circ\mathbf{V} \end{aligned} \tag{1.14}$$

The increment of the gradient, η, in eq. (1.14) is determined with a line search strategy (**?**). To impose $\mathbf{G}\mathbf{1}_c = \mathbf{1}_n$ in each iteration, $\mathbf{V}$ is normalized to satisfy the constraint. Because eq. (1.14) is prone to local minima, this method starts from several random initial points and selects the solution with smallest error.

This optimization problem is similar in spirit to recent work on clustering with non-negative matrix factorization (**?**; **?**; **?**). However, we optimize a discriminative

criterion rather than a generative one. Moreover, we simultaneously compute dimensionality reduction and clustering, using a different optimization technique.

1.3.4 Initialization

At the beginning, neither $\mathbf{G}$ nor $\mathbf{B}$ are known, but the matrix $\mathbf{G}(\mathbf{G}^T\mathbf{G})^{-1}\mathbf{G}^T$ can be estimated from the available data. Similar to previous work (**?**), we compute a local similarity matrix, $\mathbf{G}(\mathbf{G}^T\mathbf{G})^{-1}\mathbf{G}^T \in \Re^{n\times n}$, from data. We assume that $(\mathbf{G}^T\mathbf{G}) \approx s\mathbf{I}_c$, so that all classes are equally distributed and s is the number of samples per class. $\mathbf{R} = \frac{1}{s}\mathbf{G}\mathbf{G}^T$ is a hard-affinity matrix, where r_{ij} will be 1 if $\mathbf{d}_i$ and $\mathbf{d}_j$ are considered neighbors (i.e. belong to the same class). $\mathbf{R}$ can be estimated by simply computing the k nearest neighbors for each data point using the Euclidian distance. To make $\mathbf{R}$ symmetric, if $\mathbf{d}_i$ is within the k-neighborhood of $\mathbf{d}_j$, but not the contrary, then its similarity is set to zero. Figure 1.5.b shows an estimate of $\mathbf{R}$ for 15 subjects in the ORL database. Each subject (class) has ten samples and for each sample the nearest nine neighbors are selected. The samples are ordered by class. After factorizing $\mathbf{R} = \mathbf{U}\Sigma\mathbf{U}^T$, we normalize $\mathbf{R}$ as $\hat{\mathbf{R}} \approx \mathbf{U}_c\mathbf{U}_c^T$, where $\mathbf{U}_c \in \mathbf{R}^{n\times c}$ are the first c eigenvectors of $\mathbf{R}$. $\hat{\mathbf{R}}$ is the initial neighbor matrix.

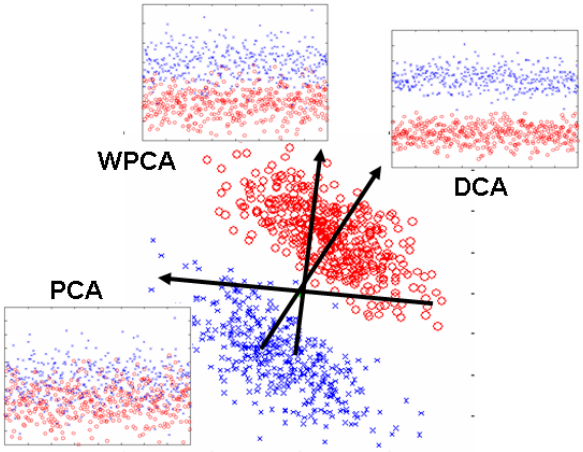


Fig. 1.1 Two class toy problem. PCA, WPCA, and DCA projections in one dimensional space.

1.3.5 Interpreting the weighted covariance matrix

A key aspect to understand DCA is the interpretation of the weighted covariance matrix $\mathbf{DRD}^T = \sum_{i=1}^{n}\sum_{j=1}^{n} r_{ij}\mathbf{d}_i\mathbf{d}_j^T$. Principal Component Analysis (PCA) (**?**) computes

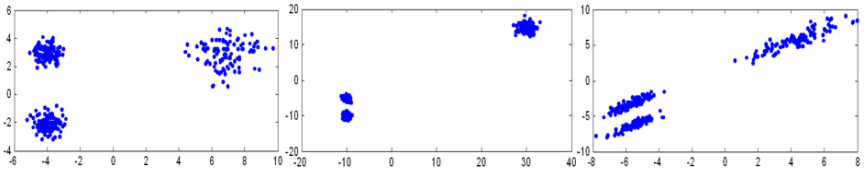
a basis $\mathbf{B}$ that maximizes the variance of the projected samples, i.e. PCA finds an orthonormal basis that maximizes $tr(\mathbf{B}^T\mathbf{D}\mathbf{D}^T\mathbf{B}) = \sum_{i=1}^{n} ||\mathbf{B}^T\mathbf{d}_i||_2^2$. The PCA solution $\mathbf{B}$ is given by the eigenvectors of $\mathbf{D}\mathbf{D}^T$. Finding the leading eigenvectors of $\mathbf{D}\mathbf{R}\mathbf{D}^T$ is equivalent to maximizing $tr(\mathbf{B}^T\mathbf{D}\mathbf{R}\mathbf{D}^T\mathbf{B}) = \sum_{i=1}^{n}\sum_{i=1}^{n} r_{ij}\mathbf{d}_i^T\mathbf{B}\mathbf{B}^T\mathbf{d}_j$. If $\mathbf{R} = \mathbf{I}$, it is equivalent to standard PCA. However, if $\mathbf{R}$ is $\mathbf{G}(\mathbf{G}^T\mathbf{G})^{-1}\mathbf{G}^T$, where $\mathbf{G}$ is the indicator matrix (or an approximation), the weighted covariance only maximizes the covariance within each cluster. This effectively maximizes the correlation between each pair of points in the same class. Figure 1.1 shows a toy problem with two oriented Gaussian classes. The first eigenvector in PCA finds a direction of maximum variance that does not necessarily correspond to maximum discrimination. In fact, by projecting the data into the first principal component, the clusters overlap. If $\mathbf{R}$ is the initial matrix of neighbors, the first step of DCA finds a more suitable projection that maximizes class separability (see fig. 1.1) .

1.4 Experiments

This section describes three experiments using synthetic and real data that demonstrate the effectiveness of DCA for clustering.

1.4.1 Clustering with DCA

In the first experiment, we show how the DCA error function is able to correctly cluster oriented clusters.



a) b) c)

Fig. 1.2 Three examples of three two-dimensional Gaussian clusters.

Consider the DCA optimization expression, eq. (1.10), when $\mathbf{B} = \mathbf{I}_d$ (i.e. no projection); in this case, eq. (1.10) becomes $tr((\mathbf{G}^T\mathbf{G})^{-1}\mathbf{G}^T\mathbf{D}^T(\mathbf{D}\mathbf{D}^T)^{-1}\mathbf{D}\mathbf{G})$. This error function, due to the term $(\mathbf{D}\mathbf{D}^T)^{-1}$, provides affine invariance to clustering. To illustrate this property, we have generated three examples of three two-dimensional random Gaussian clusters. Figure 1.2.a shows three clusters of 300 samples each, generated from three two-dimensional Gaussians: $N_2(\mathbf{x};[-4;3],0.25\mathbf{I}_2)$, $N_2(\mathbf{x};-[4;2],0.25\mathbf{I}_2)$ and $N_2(\mathbf{x};[7;3],0.25\mathbf{I}_2)$. Similarly, fig. 1.2.b illustrates 300 samples generated from

three two-dimensional Gaussians $N_2(\mathbf{x};[-10;-10],0.25\mathbf{I}_2)$, $N_2(\mathbf{x};[-10;-5],0.25\mathbf{I}_2)$ and $N_2(\mathbf{x};[30;15],0.25\mathbf{I}_2)$. Analogously, fig. 1.2.c shows $N_2(\mathbf{x};-[4;3],2[1\ 0.8;0.1\ 1])$, $N_2(\mathbf{x};-[4;2],0.25[1\ 0.8;0.1\ 1])$ and $N_2(\mathbf{x};[3;3],0.25[1\ 0.8;0.1\ 1])$.

We run DCA and k-means with the same random initialization and let both algorithms converge. To compute the accuracy of the results for a c cluster case, we compute a c-by-c confusion matrix $\mathbf{C}$, where each entry c_{ij} is the number of data points in cluster i that belong to class j. It is difficult to compute the accuracy by strictly using the confusion matrix $\mathbf{C}$, because it is unknown which cluster matches with which class. An optimal way to solve it is to compute the following maximization problem (**?**; **?**):

$$\max\ tr(\mathbf{CP}) \mid \mathbf{P}\ \textit{is a permutation matrix} \tag{1.15}$$

To solve eq. (1.15), we use the classical Hungarian algorithm (**?**). Table (1.2) shows the clustering accuracy for the three examples described above. We run the algorithms 1000 times from different random initializations (same for k-means and DCA).

	k-means	DCA
Fig. 1.2.a	0.713±0.23%	0.990±0.05%
Fig. 1.2.b	0.526±0.07%	0.959±0.09%
Fig. 1.2.c	0.594±0.13%	0.974±0.06%

Table 1.1 Comparison of clustering accuracy for DCA and k-means.

As we can see from the results in table 1.1, DCA is able to achieve better clustering results starting from the same initial condition as k-means. Moreover, DCA results in a more stable (less variance) clustering. k-means clustering accuracy largely degrades when two clusters are closer together or the clusters are not spherical. DCA is able to keep the accuracy even with oriented clusters (fig. 1.2.c).

1.4.2 Removing undesirable dimensions

The second experiment demonstrates the ability of DCA to deal with undesired dimensions not relevant for clustering. A synthetic problem is created as follows: 200 samples from a two-dimensional Gaussian distribution with mean $[-5,-5]$ and another 200 samples from another Gaussian distribution with mean $[5,5]$ are generated (x and y dimensions). We add a third dimension generated with uniform noise between $[0,35]$ (z dimension). Figure 1.3 shows 200 samples of each class in the original space (fig. 1.3.a), as well as the projection (fig. 1.3.b) onto x and y. The k-means algorithm is biased by the noise (fig. 1.4.a). Similarly, projecting the data into the first two principal components also produces the wrong clustering because PCA preserves the energy of the uniform noise, which is not relevant for clustering.

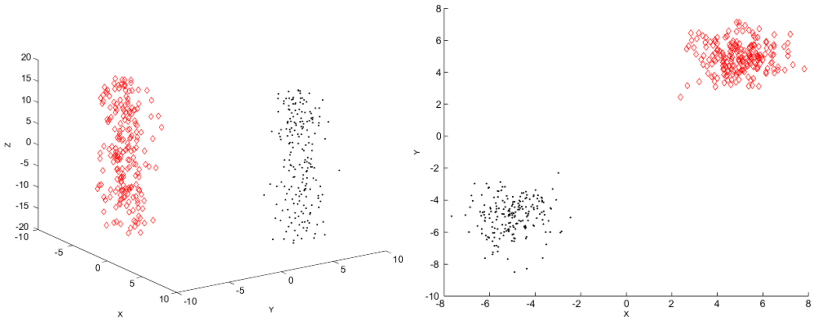


Fig. 1.3 a) 2 classes of 3 dimensional data. b) Projection onto XY space.

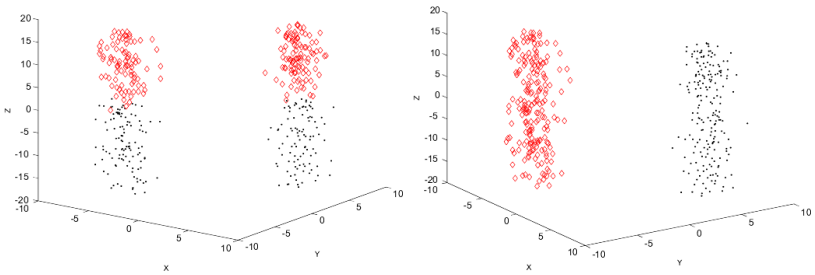


Fig. 1.4 a) *k*-means clustering. b) DCA clustering.

However, DCA is able to remove the noise and achieve the correct clustering as evidenced in fig. 1.4.b. In this particular example 15 neighbors were initially selected and $\mathbf{B} \in \Re^{3\times 2}$.

1.4.3 Clustering faces

The final experiment shows results on clustering faces from the ORL face database (**?**). The ORL face database is composed of 40 subjects and 10 images per subject. We randomly select c subjects from the database and add the 10 images of the subject to $\mathbf{D} \in \Re^{d\times 10c}$ (e.g. fig. 1.5.a). Afterwards, we compute PCA, weighted PCA (WPCA), PCA+LDA (preserving 95% of the energy in PCA), and DCA. After computing PCA, WPCA (with the initial matrix $\mathbf{R}$), and PCA+LDA, we run the *k*-means algorithm 10 times and the solution with smallest error is chosen. This procedure is repeated 40 times for different number of classes (between 4 and 40 subjects). To perform a fair comparison, we project the data into the number of classes minus ones $(c-1)$ dimensions for all methods.

Fig. 1.6 shows the accuracy in clustering for PCA+*k*-means versus DCA. For a given number of clusters, we show the mean and variance over 40 realizations. DCA always outperforms PCA+*k*-means. Table 1.2 shows some numerical values for the

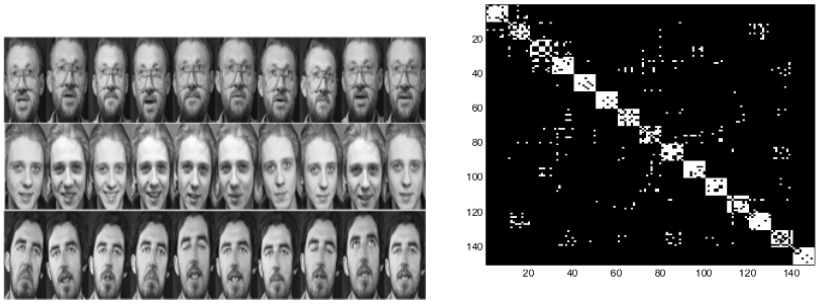


Fig. 1.5 a) Some faces of the ORL data base. b) Estimate of **R** for 15 clusters (people), each cluster has 10 samples. The samples are ordered by clusters.

c	PCA	WPCA	DCA	PCA+LDA
4	73±0%	1±0%	87±2%	**1**±0%
10	88±6%	95±6%	**97±4%**	88±8%
15	86±5%	88±4%	**96±1%**	82±6%
20	80±4%	84±4%	**87±2%**	83±4%
25	77±3%	80±4%	**87±2%**	80±4%
30	75±3%	79±3%	**81±3%**	81±4%
35	73±4%	77±3%	78±4%	**81±3%**
40	71±2%	74±3%	73±3%	**80±4%**

Table 1.2 Comparison of the clustering accuracy for several projection methods (same number of bases).

clustering accuracy. DCA outperforms most of the methods when there are between 5 and 30 classes. For more classes, $PCA + LDA$ performs marginally better. In addition, the accuracy of the PCA+k-means method drops as the number of classes increases (as expected).

1.5 Discussion and future work

In this paper, we have proposed DCA, a technique that jointly performs dimensionality reduction and clustering. In synthetic and real examples, DCA outperforms standard k-means and PCA+k-means, for clustering high dimensional data. DCA provides a discriminative embedding that maximize cluster separation and is less prone to local minima. Additionally, we have proposed an unbiased least-squares formulation for LDA.

Although DCA has shown promising preliminary results, several issues still need to be addressed. It remains unclear how to select the optimal number of clusters. Several model order selection (e.g. Minimum Description Length or Akaike information criterion) could be applied towards this end. On the other hand, DCA assumes that all the clusters have the same orientation (not necessarily spherical).

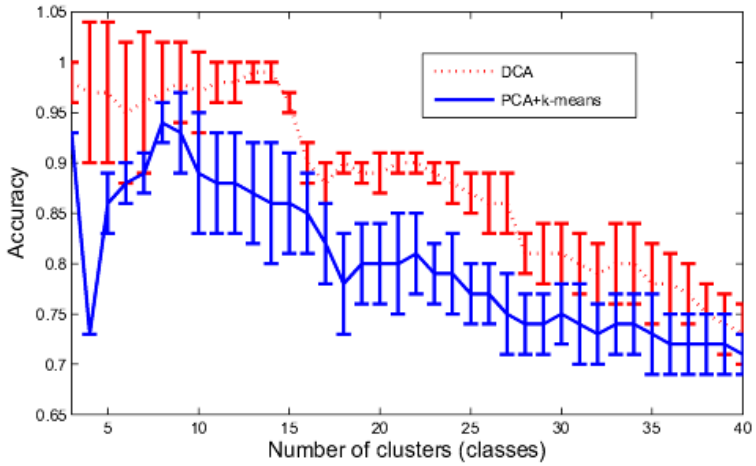


Fig. 1.6 Accuracy of clustering versus the number of classes. Blue PCA and red DCA (dotted line).

This limitation could be easily address by using kernel extensions of eq. (1.10) to deal with non-Gaussian clusters.

1.6 Acknowledgements

This work has been partially supported by MH R01 51435 from the National Institute of Mental Health, N000140010915 from the Naval Research Laboratory, the Department of the Interior National Business Center contract no. NBCHD030010, and SRI International subcontract no. 03-000211.

Bibliography