Segmentation of multivariate mixed data via lossy coding and compression

Harm Derksen, Yi Ma, Wei Hong, John Wright

Harm Derksen, Yi Ma, Wei Hong, John Wright, "Segmentation of multivariate mixed data via lossy coding and compression," Proc. SPIE 6508, Visual Communications and Image Processing 2007, 65080H (29 January 2007); doi: 10.1117/12.714912

SPIE.

Event: Electronic Imaging 2007, 2007, San Jose, CA, United States

Segmentation of multivariate mixed data via lossy coding and compression

Harm Derksen[a] and Yi Ma[b] and Wei Hong[c] and John Wright[b]

[a]Department of Mathematics, University of Michigan, USA;
[b]Coordinated Science Laboratory, University of Illinois at Urbana Champaign, USA;
[c]DSP Solutions Research and Development Center, Texas Instruments

ABSTRACT

In this paper, based on ideas from lossy data coding and compression, we present a simple but surprisingly effective technique for segmenting multivariate mixed data that are drawn from a mixture of Gaussian distributions or linear subspaces. The goal is to find the optimal segmentation that minimizes the overall coding length of the segmented data, subject to a given distortion. We show that deterministic segmentation minimizes an upper bound on the (asymptotically) optimal solution. The proposed algorithm does not require any prior knowledge of the number or dimension of the groups, nor does it involve any parameter estimation. Simulation results reveal intriguing phase-transition behaviors of the number of segments when changing the level of distortion or the amount of outliers. Finally, we demonstrate how this technique can be readily applied to segment real imagery and bioinformatic data.

Keywords: Multivariate mixed data, Data segmentation, Rate-distortion, Lossy data coding, Data compression, Image segmentation, Microarray data clustering.

1. INTRODUCTION

Data that arise from practical problems in such diverse fields as image/signal processing, pattern recognition, computer vision, and bioinformatics, are often characterized by complicated multi-modal distributions. Segmentation (or clustering) is widely recognized as an important step in analyzing, representing, interpreting or compressing such mixed data. Numerous objective functions and algorithms have been proposed for segmenting mixed data (see[1,2] and references therein).

The common statistical approach to data segmentation is to model the data, $X = \{x_i\} \subseteq \mathbb{R}^n$, as samples drawn from a mixture of simple probability distributions (e.g. Gaussians): $p(x|\theta,\pi) = \sum_{j=1}^{k} \pi_j p_j(x|\theta_j)$. The parameters of such a mixture model and the segmentation of the data can be obtained via model estimation criteria such as maximum likelihood (ML). The Expectation Maximization (EM) algorithm[3] offers an effective solution to this problem. Maximizing the likelihood is equivalent to minimizing the log-likelihood: $\sum_i -\log p(x_i|\hat{\theta},\hat{\pi})$, which is approximately the expected coding length, $\text{Length}(X|\hat{\theta},\hat{\pi})$, required to store the data using the optimal coding scheme for the distribution $p(x|\hat{\theta},\hat{\pi})$.[4]

When the number of component models, k, is not known *a priori*, one typically resorts to certain model selection criteria to find the optimal k (and θ,π). For instance, the minimum description length (MDL) criterion[5] attempts to minimize the total length needed to code the data X and the model parameters θ,π : $\text{Length}(X,\hat{\theta},\hat{\pi}) = \text{Length}(X|\hat{\theta},\hat{\pi}) + \text{Length}(\hat{\theta},\hat{\pi})$, where the parameters θ,π are assumed to have certain distribution $p(\theta,\pi)$. Since the model parameters are typically real-valued, in order for them to be represented with a finite number of bits, they must be quantized. Typically, the quantization error for the model parameters is chosen to be the one that minimizes the expected coding length.[6]

Further author information: (Send correspondence to J.W.)
H.D.: E-mail: hderksen@umich.edu, Telephone: 1 734 763 2309
Y.M.: E-mail: yima@uiuc.edu, Telephone: 1 217 244 0871
W.H.: E-mail: weihong@ti.com, Telephone: 1 214 480 1531
J.W.: E-mail: jnwright@uiuc.edu, Telephone: 1 217 244 6626

Visual Communications and Image Processing 2007, edited by Chang Wen Chen, Dan Schonfeld, Jiebo Luo,
Proc. of SPIE-IS&T Electronic Imaging, SPIE Vol. 6508, 65080H, © 2007 SPIE-IS&T · 0277-786X/07/$18

In contrast to the ML/MDL criteria, one may also seek a model which represents the (mixed) data subject to a given distortion. For instance, the important role of distortion in data clustering has been thoroughly investigated in the context of vector quantization (VQ).[7] There it is proposed that subject to a given distortion, the optimal model (or segmentation) is the one that maximizes the entropy between the data and the model (hence potentially maximizing the generalizability of the model learned).

This paper follows in the spirit of both MDL and VQ. Our goal is to *find the optimal segmentation of the mixed data which results in the shortest coding length subject to a given distortion of the data.* To this end, we develop a new technique for segmenting multivariate mixed data, which offers the following improvements over the aforementioned methods:

1. Our technique aims to segment the data into multiple Gaussian-like groups that may have significantly different and anisotropic covariances. The covariances of the groups may be nearly singular, in which case the technique essentially fits the data with *multiple subspaces, possibly of different dimensions.* In this context, VQ[7] can be viewed as the special case of fitting the data with zero-dimensional (affine) subspaces.

2. Given a distortion level, we choose the measure of coding length for each cluster to be the optimal rate-distortion function for a multivariate Gaussian source.[4] We will show that with this choice, the (asymptotically) optimal segmentation is *deterministic* – no probabilistic (or fuzzy) segmentation can further reduce the overall coding length.

3. The existence of an explicit formula for the rate-distortion of Gaussian sources allows us to directly seek the optimal segmentation, completely bypassing the estimation of any (mixture) model parameters. This leads to an efficient* "bottom-up" algorithm that minimizes the overall coding length by repeatedly merging groups, starting from individual data points. It resolves the difficult model selection issue[6] in a remarkably effective way, especially when *the number of groups is unknown or there is a significant amount of outliers.*

4. When the level of distortion varies continuously, the number of groups typically exhibits a *phase transition* behavior similar to that in statistical physics, with the "correct" segmentation corresponding to one of the stable phases.† Our simulations show that the number of segments need not be a monotonic function of the distortion.

2. BASIC IDEAS AND ALGORITHM

Lossy Coding of Multivariate Data. A lossy coding scheme maps a set of vectors $V = (v_1, v_2, \ldots, v_m) \in \mathbb{R}^{n\times m}$ to a sequence of binary bits, such that the original vectors can be recovered up to an allowable distortion $\mathbb{E}[\|v_i - \hat{v}_i\|^2] \leq \varepsilon^2$. The length of the encoded sequence is denoted as the function $L(V) : \mathbb{R}^{n\times m} \to \mathbb{Z}_+$.

In general, the coding scheme and the associated L function can be chosen to be optimal for any family of distributions of interest. In the case where the data are i.i.d. samples from a zero-mean multivariate Gaussian distribution $\mathcal{N}(0, \Sigma)$, it is well known in information theory[4] that the optimal rate-distortion function is $R = \frac{1}{2}\log_2\det(I + \frac{n}{\varepsilon^2}\Sigma)$ when the distortion ε^2 is small. As $\hat{\Sigma} = \frac{1}{m}VV^T$ is an estimate of the covariance Σ, the average number of bits needed per vector is:

$$R(V) \doteq \frac{1}{2}\log_2\det\big(I + \frac{n}{\varepsilon^2 m}VV^T\big). \tag{1}$$

Representing the m vectors of V therefore requires $mR(V)$ bits‡. Since the optimal codebook is adaptive to the

*The complexity of the proposed algorithm is polynomial in both the size and dimension of the data.

†A different phase transition has been noticed in vector quantization using deterministic annealing, where the number of clusters increases monotonically when the annealing temperature decreases.[7]

‡The construction of efficient coding schemes which achieve the optimal rate-distortion bound is itself a difficult problem (see, for example,[8] and references therein). However, for the purpose of measuring the quality of segmentation, all that matters is that *in principle* a scheme attaining the optimal rate $R(V)$ exists.

data V, we must also represent it with an additional $nR(V)$ bits[§], yielding an overall coding length of

$$L(V) \doteq (m+n)R(V) = \frac{m+n}{2}\log_2\det\big(I + \frac{n}{\varepsilon^2 m}VV^T\big). \tag{2}$$

We will study the properties of this function in Section 3. For purposes of segmentation, it suffices to note that in addition to being asymptotically optimal for Gaussian data, $L(V)$ also provides a tight bound on the number of bits needed to code a finite number of vectors that span a (low-dimensional) subspace, regardless of the underlying probability distribution (see Appendix A for a proof).

Segmentation via Data Compression. Given a set of samples, $X = (x_1, x_2, \ldots, x_m) \in \mathbb{R}^{n\times m}$, one can always view them as drawn from a single Gaussian source and code X subject to distortion ε^2 using $L(X)$ bits. However, if the samples are drawn from a mixture of Gaussian distributions or linear subspaces, it may be more efficient to code X as the union of multiple (disjoint) subsets: $X = X_1 \cup X_2 \cup \cdots \cup X_k$. If each subset is coded separately, the total number of bits needed is

$$L^s(X) = L^s(X_1, X_2, \ldots, X_k) \doteq \sum_{i=1}^{k} L(X_i) + |X_i|\big(-\log_2(|X_i|/m)\big). \tag{3}$$

Here, the second term counts the number of bits needed to code (losslessly) the membership of the m samples in the k groups (e.g. using the Huffman coding).[¶]

Then, given a fixed coding scheme with its associated coding length function $L(\cdot)$, an optimal segmentation is one which minimizes the segmented coding length, $L^s(X)$, over all possible partitions of X. Moreover, we will see that due to the properties of the rate-distortion function (1) for Gaussian data, softening the objective function (3) by allowing probabilistic (or fuzzy) segmentation does *not* further reduce the (expected) overall coding length (see Theorem 3.1 of Section 3).

Minimizing the Coding Length. Finding the global minimum of the overall coding length $L^s(X)$ over all partitions of the dataset is a daunting combinatorial optimization problem, intractable for large data sets. Nevertheless, the coding length can be effectively minimized in a steepest descent fashion, as outlined in Algorithm 1. The minimization proceeds in a "bottom-up" fashion: initially, every sample is treated as its own group. At each iteration, two groups S_1 and S_2 are chosen so that merging them results in the greatest decrease in the coding length. The algorithm terminates when the coding length cannot be further reduced by merging any pair of groups.[‖] A simple implementation which maintains a table containing $L^s(S_i \cup S_j)$ for all i, j requires $O(m^3 + m^2n^3)$ time, where m is the number of samples and n the dimension of the space.

Algorithm 1 (Pairwise Steepest Descent of Coding Length).

input: the data $X = (x_1, x_2, \ldots, x_m) \in \mathbb{R}^{n\times m}$ and a distortion $\varepsilon^2 > 0$.
initialize $\mathcal{S} := \{S = \{x\} \mid x \in X\}$.
while $|\mathcal{S}| > 1$ **do**
    choose distinct sets $S_1, S_2 \in \mathcal{S}$ such that $L^s(S_1 \cup S_2) - L^s(S_1, S_2)$ is minimal.
    if $L^s(S_1 \cup S_2) - L^s(S_1, S_2) \geq 0$ **then** break;
    else $\mathcal{S} := \big(\mathcal{S} \setminus \{S_1, S_2\}\big) \cup \{S_1 \cup S_2\}$.
end
output: $\mathcal{S}$

[§]This can be viewed as the cost of coding the n principal axes of the data covariance $\frac{1}{m}VV^T$. For a detailed derivation of $L(V)$, please refer to Appendix A

[¶]Here we assume the ordering of the samples is random and entropy coding is the best we can do to code the membership. However, if the samples are ordered such that nearby samples more likely belong to the same group (e.g., in segmenting pixels of an image), the second term can and should be replaced by a tighter estimate.

[‖]In the supplementary material, we have included a video showing the convergence of this algorithm on data drawn from mixtures of subspaces in $\mathbb{R}^3$.

Extensive simulations (see Section 4) demonstrate that this algorithm is consistently and remarkably effective in segmenting mixed data. It tolerates significant amounts of outliers, and requires no prior knowledge of the number of groups. As a greedy descent scheme, the algorithm does not guarantee to always find the globally optimal segmentation for any given (X, ε).** We found that empirically the main factor affecting the global convergence of the algorithm seems to be the density of the samples relative to the distortion ε^2.

3. FURTHER ANALYSIS

In this section, we examine several mathematical properties of the coding length function, and highlight their importance and implications in the context of data compression and segmentation. Readers who are interested only in the algorithm and experimental results may bypass this section without loss of continuity.

Invariant Property. The coding length function $L(X)$ in equation (2) is invariant under any orthogonal transformation of the data X. That is, for any $U \in O(n)$ or $V \in O(m)$, we have

$$L(XW) = L(X) = L(XV). \tag{4}$$

This equality suggests that one may choose any orthonormal basis (e.g., Fourier, wavelets) to represent and encode the data and the number of bits needed would always be the same. This agrees with the fact that the chosen coding length (or rate) is optimal for a stationary Gaussian process. However, if the data are non-Gaussian or nonlinear, a proper transformation can be useful for compressing the data.†† Here we are seeking a partition, rather than a transformation, of the non-Gaussian (or non-linear) data set, such that each subset is sufficiently Gaussian (or linear) and hence cannot be compressed any further, either by (orthogonal) transformation or segmentation.

Commutative Property. Since $XX^T \in \mathbb{R}^{n\times n}$ and $X^TX \in \mathbb{R}^{m\times m}$ have the same eigenvalues, the coding length function (2) can also be expressed as:

$$L(X) = \frac{m+n}{2}\log_2\det\big(I + \frac{n}{m\varepsilon^2}XX^T\big) = \frac{m+n}{2}\log_2\det\big(I + \frac{n}{m\varepsilon^2}X^TX\big). \tag{5}$$

Thus, if $n \ll m$, the second expression will be less costly for computing the coding length. The matrix X^TX, which depends only on the inner products between pairs of data vectors, is known in the statistical learning literature as the *kernel matrix*. This property suggests that the ideas and the algorithm presented in Section 2 can be readily extended to segment data sets that have *piecewise smooth nonlinear* structures, by choosing a proper kernel function.

Optimality of Deterministic Segmentation. In motivating the greedy algorithm of Section 2, we have considered only the deterministic partition of a dataset X into groups $X_1, X_2, \ldots, X_k$, with each sample vector x_i assigned to exactly one group. More generally though, one can consider a probabilistic segmentation, in which x_i is assigned to group j with probability $\pi_{ij} \in [0, 1]$. This raises an important theoretical question, with practical implications for gradient descent approaches to minimizing the coding length: can such a probabilistic segmentation reduce the number of bits needed to code the whole data set? Surprisingly, we shall see that it cannot: *the global minimum of the (asymptotic) coding rate always occurs at a deterministic segmentation.*

To count the number of bits required for a probabilistic segmentation, we introduce a set of diagonal matrices $\Pi = \{\Pi_j\}$. Each $\Pi_j \doteq \mathrm{diag}\{\pi_{1j}, \pi_{2j}, \ldots, \pi_{mj}\}$ encodes the membership of the m vectors in the j-th group. The expected number of vectors in group X_j is then $\mathbf{tr}(\Pi_j)$, and the group's covariance is $\frac{1}{\mathbf{tr}(\Pi_j)}X\Pi_jX^T$. From the normalization constraint $\sum_{j=1}^k \pi_{ij} = 1\,\forall\, i$, we have that $\sum_{j=1}^k \Pi_j = I_{m\times m}$.

**Due to Theorem 1 of Section 3, the globally (asymptotically) optimal segmentation can also be computed via concave optimization,[9] at the cost of potentially exponential computation time.

††For a more thorough discussion on why some transformations (such as wavelets) are useful for data compression, the reader may refer to.[10]

As a probabilistic counterpart of equation (3), the following formula gives an upper bound on the expected number of bits needed to code the m vectors W according to the probabilistic segmentation Π:

$$L^s(X,\Pi) = \sum_{i=1}^{k} \frac{\mathbf{tr}(\Pi_j)+n}{2} \log_2 \det\big(I + \frac{n}{\mathbf{tr}(\Pi_j)\varepsilon^2} X\Pi_j X^T\big) + \mathbf{tr}(\Pi_j)\Big(-\log_2 \frac{\mathbf{tr}(\Pi_j)}{m}\Big). \tag{6}$$

The corresponding rate (bits per vector) is

$$R^s(X,\Pi) \doteq \frac{1}{m} L^s(X,\Pi) = \sum_{j=1}^{k} \frac{\mathbf{tr}(\Pi_j)}{m}\Big(R(X,\Pi_j) - \log_2 \frac{\mathbf{tr}(\Pi_j)}{m}\Big) + \frac{n}{m} R(X,\Pi_j), \tag{7}$$

where $R(X,\Pi_j) \doteq \frac{1}{2}\log_2\det(I + \frac{n}{\mathbf{tr}(\Pi_j)\varepsilon^2} X\Pi_j X^T)$. When $m \gg n$, the $\frac{n}{m}R(X,\Pi_j)$ term becomes negligible. Hence the first term of equation (7) determines the asymptotic behavior of the rate function, as the number of vectors in each group goes to infinity. We denote the asymptotic part of the rate function as

$$R^{s,\infty}(X,\Pi) \doteq \sum_{j=1}^{k} \frac{\mathbf{tr}(\Pi_j)}{2m} \log_2 \det\big(I + \frac{n}{\varepsilon^2\,\mathbf{tr}(\Pi_j)} X\Pi_j X^T\big) - \frac{\mathbf{tr}(\Pi_j)}{m} \log_2\Big(\frac{\mathbf{tr}(\Pi_j)}{m}\Big). \tag{8}$$

THEOREM 3.1. *The asymptotic part $R^{s,\infty}(X,\Pi)$ of the rate distortion function $R^s(X,\Pi)$ is a concave function of Π in the convex domain $\Omega \doteq \{\Pi : \sum_{j=1}^k \Pi_j = I, \Pi_j \succeq 0\}$.*

Proof. Let $\mathcal{S}$ be the set of all $m \times m$ non-negative definite symmetric matrices. We will show that $R^{s,\infty}(X,\Pi)$ is concave as a function from $\mathcal{S}^k \to \mathbb{R}$, and so is it when restricted to $\Omega \subset \mathcal{S}^k$.

Consider the first term of $R^{s,\infty}(X,\Pi)$. Let $h(\Pi_j) \doteq \mathbf{tr}(\Pi_j)\log_2\det\big(I + \frac{n}{\varepsilon^2\,\mathbf{tr}(\Pi_j)} X\Pi_j X^T\big)$. It is well-known in information theory that the function $q(P) \doteq \log_2\det(P)$ is concave for $P \in \mathcal{S}$ and $P \succ 0$ (Theorem 16.8.1 of[4]). Now define $r : \mathcal{S} \to \mathbb{R}$ to be $r(\Pi_j) \doteq \log_2\det(I + \alpha X\Pi_j X^T) = q(I + \alpha X\Pi_j X^T)$. Since r is just the concave function q composed with an affine transformation $\Pi_j \mapsto I + \alpha X\Pi_j X^T$, r is concave (see Section 3.2.3 of[11]). Let $\psi : \mathcal{S} \times \mathbb{R}_+ \to \mathbb{R}$ as $\psi(\Pi_j, t) \doteq t \cdot \log_2\det\big(I + \frac{n}{\varepsilon^2 t} X\Pi_j X^T\big) = t \cdot r\big(\frac{1}{t}\Pi_j\big)$. According to Theorem 3.2.6 of,[11] ψ is concave. Notice that $H \doteq \{(\Pi_j, t) : t = \mathbf{tr}(\Pi_j)\}$ is a linear subspace in the product space of $\mathbb{R}$ and the space of all symmetric matrices. So, $H \cap (\mathcal{S} \times \mathbb{R}_+)$ is a convex set, and the desired function, $h(\Pi_j) = \psi(\Pi_j, \mathbf{tr}(\Pi_j))$, is just the restriction of ψ to this convex set. Thus, h is concave.

Now consider the second term of $R^{s,\infty}(X,\Pi)$. We only need to show the concavity of the function $g(P) \doteq -\mathbf{tr}(P)\log_2\mathbf{tr}(P)$ for $P \in \mathcal{S}$, since the second term of $R^{s,\infty}$ is just $g(\frac{1}{m}\Pi_j)$. The function, $f(x) = -x\log_2 x$ is concave, and $g(P) = f(\mathbf{tr}(P))$. So for $\lambda \in [0,1]$, $g(\lambda P_1 + (1-\lambda)P_2) = f(\lambda\,\mathbf{tr}(P_1) + (1-\lambda)\,\mathbf{tr}(P_2)) \geq \lambda f(\mathbf{tr}(P_1)) + (1-\lambda) f(\mathbf{tr}(P_2)) = \lambda g(P_1) + (1-\lambda) g(P_2)$. Thus, $g(P)$ is concave in P.

Since $R^{s,\infty}(X,\Pi)$ is a sum of concave functions in Π_j, it is concave as a function from $\mathcal{S}^k$ to $\mathbb{R}$, and so is its restriction to the convex set Ω in $\mathcal{S}^k$. □

Since $R^{s,\infty}(X,\Pi)$ is concave in Π, its global minimum Π^* always lies at the boundary, or more precisely, at a vertex of the convex domain Ω. As Ω is a high-dimensional simplex, the entries π_{ij} of the optimal vertex Π^* are either 0s or 1s. This means that even if we allow probabilistic assignment of each point to the k groups according to any probabilistic distribution, the optimal solution can always be achieved by assigning each point to one of the groups with probability one! This suggests that deterministic segmentation is the optimal solution to an even large class of minimum coding length problems.[‡‡]

4. SIMULATION AND EXPERIMENTAL RESULTS

Segmentation of Linear Subspaces of Different Dimensions. We first demonstrate the ability of the algorithm to segment noisy samples drawn from a mixture of linear subspaces of different dimensions. Figure 1 summarizes the configurations tested. For every d-dimensional subspace, $d \times 100$ samples are drawn uniformly

Subspace dimensions	Identified dimensions	Classification (%) (Algorithm 1)	Classification (%) (E-M)
$(2,1,1)$ in $\mathbb{R}^3$	$2,1,1$	96.62	39.33
$(2,2,1)$ in $\mathbb{R}^3$	$2,2,1$	90.00	68.98
$(4,2,2,1)$ in $\mathbb{R}^5$	$4,2,2,1$	98.53	43.36
$(6,3,1)$ in $\mathbb{R}^7$	$6,3,1$	99.77	66.16
$(7,5,2,1,1)$ in $\mathbb{R}^8$	$7,5,2,1,1$	98.04	42.29

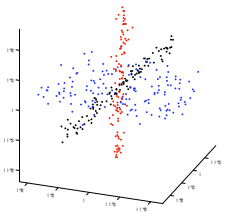


Figure 1. Left: Simulation results for data drawn from mixtures of noisy linear subspaces. Classification percentages are averaged over 25 trials. Our algorithm correctly identifies the number and dimension of the subspaces in all 25 trials, for all configurations. Far right column: results using Expectation-Maximization with random initialization. Right: the computed segmentation for $(2,1,1)$ in $\mathbb{R}^3$ is displayed.

from a ball of diameter 1 lying on the subspace. Each sample is corrupted with independent Gaussian noise of standard deviation $\varepsilon_0 = 0.04$. The segmentation is computed using Algorithm 1, with $\varepsilon = \varepsilon_0$.

In each case, the algorithm stops at the correct number of groups, and the dimensions of the segments X_i match those of the generating subspaces. The correctness of the segmentation is further corroborated by the high percentage of points correctly classified (by comparing the segments with the *a priori* groups). For all five configurations, the average percentage of samples assigned to the correct group was at least 90.0%. The main cause of classification error is points which lie near the intersection of multiple subspaces. Due to noise, it may actually be more efficient to code such points according to the optimal coding scheme for one of the other subspaces. We compare our method to Expectation-Maximization, seeded with a random initialization. In all cases, Algorithm 1 dramatically outperforms EM in terms of classification error, despite requiring less prior knowledge.

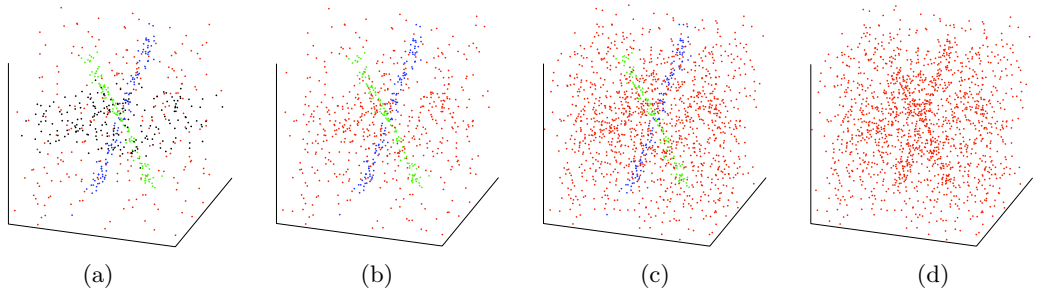
Global Convergence. Empirically, we find that Algorithm 1 does not suffer many of the difficulties with local minima that plague other clustering algorithms such as EM. The convergence appears to depend mostly on the density of the samples relative to the distortion, ε^2. For example, if the number of samples is fixed at $m = 1200$, and the data are drawn from three $\lceil \frac{n}{2} \rceil$-dimensional subspaces in $\mathbb{R}^n$, the algorithm converges to the correct solution for $n = 2$ upto $n = 56$. Beyond that, the algorithm fails to converge to the three *a priori* subspaces as the samples have become too sparse. For $n > 56$, the computed segmentation gives a higher coding length than the *a priori* segmentation.

Robustness to Outliers. We test the robustness of Algorithm 1 to outliers on the easily visualized example of two lines and a plane in $\mathbb{R}^3$. 158 samples are drawn uniformly from a 2-D disc of diameter 1. 100 samples are drawn uniformly from each of the two line segments of length 1. The additive noise level is $\varepsilon_0 = 0.03$. The data set is contaminated with m_o outliers, whose three coordinates are uniformly distributed on $[-0.5, 0.5]$.

As the number of outliers increases, the segmentation exhibits several distinct phases. For $m_o \leq 300$ (45.6% outliers), the algorithm always finds the correct segmentation. The outliers are merged into a single (three-dimensional) group. From $m_o = 400$ (52.8% outliers) upto $m_o = 1100$ (75.4% outliers), the two lines are correctly identified, but samples on the plane are merged with the outliers. For $m_o = 1200$ (77.4% outliers) and higher, all of the data samples are merged into one group, as the distribution of data has become essentially random in the ambient space. Figure 2 shows the results for $m_o = 300, 400, 1100, 1200$. Notice that the effect of adding the outliers resembles the effect of ice (the lines and the plane) being melted away by warm water. This suggests a similarity between the artificial process of data clustering and the physical process of phase transition.

Number of Segments versus Distortion Level. Figure 3 shows how the number of segments changes as ε varies. $m = 358$ points are drawn from two lines and a plane, as in the previous experiment, and then perturbed with noise of standard deviation $\varepsilon_0 = 0.05$. Notice that the number of groups experiences distinct phases, with abrupt transitions around several critical values of ε. For sufficiently small ε, each data point is grouped by itself. However, as ε increases, the cost of coding the group membership begins to dominate, and

[‡‡]Notice, however, that we have only demonstrated that a deterministic segmentation minimizes this *upper bound* on the expected coding length, rather than the coding length itself.



(a) (b) (c) (d)

Figure 2. Segmentation results for data drawn from three linear subspaces, corrupted with various numbers of outliers, m_o. (a) $m_o = 300$ (45.6% outliers). (b) $m_o = 400$ (52.8% outliers). (c) $m_o = 1100$ (75.4% outliers). (d) $m_o = 1200$ (77.0% outliers).

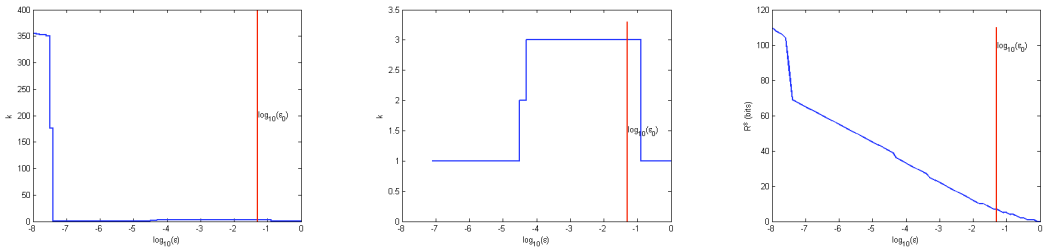


Figure 3. The effect of varying ε, with $\varepsilon_0 = 0.05$. Left: number of groups, k, versus $\log(\varepsilon)$. Center: detail of k versus $\log(\varepsilon)$ around $\log(\varepsilon_0)$. Right: the coding rate (bits per vector) versus $\log(\varepsilon)$.

all the points are grouped together in a single three-dimensional subspace (the ambient space). Around the true noise level, ε_0, there is another stable phase, corresponding to the three *a priori* subspaces. Finally, as ε becomes large, the number of segments reverts to 1, as it becomes most efficient to represent the points using a single zero-dimensional subpsace (the origin).

This behavior contrasts with the phase transition discussed in.[7] There, the number of segments increases monotonically throughout the simulated annealing process. Because our formulation allows the dimension of the segments to vary, the number of segments does not decrease monotonically with ε. Notice, however, that the phase corresponding to the "correct" (*a priori*) segmentation is stable over three orders of magnitude of the parameter ε. This is important since in practice the true noise level ε_0 is usually unknown.

Applications to Real Data. Figure 4 shows the segmentation of several images from the Berkeley image segmentation database via Algorithm 1 (using $L(\cdot)$ for nonzero-mean Gaussian data[12]). We select an 8×8 window around each pixel to use as a feature vector for segmentation. A random subset of 1,000 vectors are selected and projected onto the first 8 principal components , which are then clustered using Algorithm 1 with $\varepsilon = 1$. The remaining vectors are grouped to the nearest segment. The results are displayed without any further pre- or post-processing.

Figure 5 shows the result of applying Algorithm 1 to gene expression data. The dataset consists of 13,872 vectors in $\mathbb{R}^{19}$, each of which describes the expression level of a single gene at different time points during an experiment on anthrax sporulation. A random subset of 600 vectors is visualized in figure 5(a). Here, rows correspond to genes and columns to time points. We cluster these vectors without any preprocessing, using Algorithm 1 with $\varepsilon = 1$. The algorithm finds three distinct clusters, which are displayed in figure 5(b) by reordering the rows.

While Figure 4 and 5 demonstrate that our method is capable of finding visually appealing structures in real data, we emphasize that it does not provide a complete solution to either of these practical problems. Such a solution usually entails a significant amount of domain-specific knowledge and engineering. Nevertheless, from our preliminary experiences with images and microarray data, as well as speech data and handwritten digits, we

Figure 4. Image segmentation (via the L formula for nonzero-mean Gaussian data[12]). Top row: original images. Bottom row: computed segmentations. Each segment is painted with its mean color.

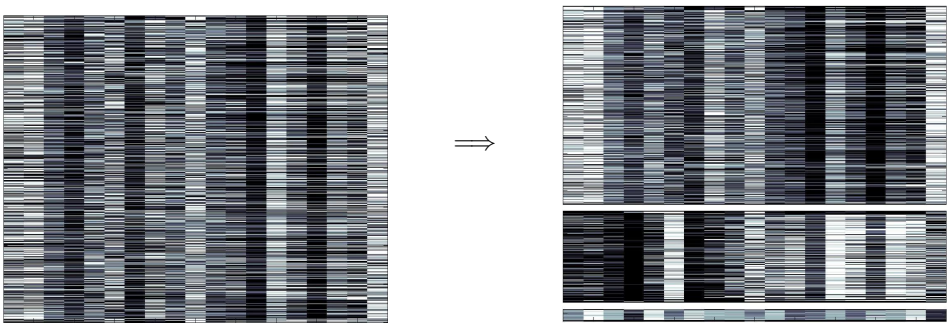


Figure 5. Segmentation of microarray data. Left: raw data. Each row represents the expression level of a single gene. Right: Three distinct clusters are found, visualized by reordering the rows.

believe that the method presented in this paper provides a generic solution for segmenting mixed data that is simple and effective enough to be easily customized for a broad range of practical problems.

5. CONCLUSION AND FUTURE WORK

In this paper, we have introduced a new technique for segmenting multivariate data drawn from a mixture of Gaussians or subspaces, based on ideas from lossy data coding and compression. The proposed algorithm provides a remarkably simple and yet effective solution, which automatically handles difficult issues such as model selection and outliers.

One direction for future work is to extend this framework to segment other types of structures, such as non-Gaussian probabilistic distributions and nonlinear manifolds. Another natural extension is to a supervised-learning scenario for purposes such as classification and recognition. From a theoretical standpoint, it would be highly desirable to obtain analytical conditions on the critical values of the distortion and the sampling density of outliers that can explain and predict the phase transition of the number of segments.

In this appendix, we derive two results which were ommited from our paper due to space limitations. The first shows that the function $L(V)$ gives a tight bound on the number of bits needed to code a finite number of vectors lying on a (low-dimensional) subspace. The second gives a tighter formula for $L(V)$ when the data vectors V are not zero-mean. This formula can be used to segment mixed data drawn from nonzero-mean Gaussian distributions or affine subspaces.

APPENDIX A. DERIVATION OF CODING LENGTH FOR FINITE SAMPLE SETS

In Section 2 of our paper, we derive the formula for the coding length of a set of vectors $V = (v_1 \dots v_m) \in \mathbb{R}^{n \times m}$, by starting from the optimal Gaussian rate-distortion:

$$R(V) = \frac{1}{2} \log_2 \det \big(I + \frac{n}{\varepsilon^2 m} V V^T\big). \tag{9}$$

However, in general, $R(V)$ is only achievable in an asymptotic setting, as $m \to \infty$. In order for $L(V)$ to be truly useful for segmentation, we need some assurance that it is meaningful even for finite sample sets. In this section, we will show that (even when m is finite and the underlying probability distribution unknown), the vectors in V can be encoded upto distortion ε^2, using at most

$$L(V) = (m+n)R(V) = \frac{m+n}{2} \log_2 \det \big(I + \frac{n}{m\varepsilon^2} V V^T\big) \tag{10}$$

bits. Interestingly, the construction exploits subspace structure in the data by separately coding the subspace spanned by the data vectors (i.e. their singular vectors) and the coordinates of the vectors with respect to that subspace.

Consider the singular value decomposition (SVD) of the data matrix $V = U \Sigma W^T$. Let $B = (b_{ij}) \doteq \Sigma W^T$. The column vectors of $U = (u_{ij})$ form a basis for the subspace spanned by vectors in V, and the columns of B are the coordinates of the data vectors with respect to this basis.

For the purpose of lossy coding, we store the approximated matrices $U + \delta U$ and $B + \delta B$. The data V can be approximately recovered as

$$V + \delta V \doteq (U + \delta U)(B + \delta B) = UB + \delta U B + U \delta B + \delta U \delta B. \tag{11}$$

When ε is small (relative to the data V), the entries of $\delta U \delta B$ are negligible, and so $\delta V \approx \delta U B + U \delta B$. The squared error introduced to the entries of V is

$$\sum_{i,j} \delta v_{ij}^2 = \mathbf{tr}\big(\delta V \delta V^T\big) \approx \mathbf{tr}\big(U \delta B \delta B^T U^T + \delta U B B^T \delta U^T + \delta U B \delta B^T U^T + U \delta B B^T \delta U^T\big).$$

We may further assume that the coding errors δU and δB are zero-mean independent random variables. Using the fact that $\mathbf{tr}(AB) = \mathbf{tr}(BA)$, the expected squared error becomes

$$\mathbb{E}\big(\mathbf{tr}\big(\delta V \delta V^T\big)\big) \quad = \quad \mathbb{E}\big(\mathbf{tr}\big(\delta B \delta B^T\big)\big) + \mathbb{E}\big(\mathbf{tr}\big(\Sigma^2 \delta U^T \delta U\big)\big).$$

Now, let us encode each entry b_{ij} with precision $\varepsilon' = \frac{\varepsilon}{\sqrt{n}}$ and u_{ij} with precision $\varepsilon''_j = \frac{\varepsilon\sqrt{m}}{\sqrt{\lambda_j n}}$, where λ_j is the jth eigenvalue of VV^T. This is equivalent to assuming that the error δb_{ij} is uniformly distributed in the interval $\big[-\frac{\varepsilon}{\sqrt{n}}, \frac{\varepsilon}{\sqrt{n}}\big]$ and δu_{ij} is uniformly distributed in the interval $\big[-\frac{\varepsilon\sqrt{m}}{\sqrt{\lambda_j n}}, \frac{\varepsilon\sqrt{m}}{\sqrt{\lambda_j n}}\big]$. Under such a coding precision, it is easy to verify that

$$\mathbb{E}\big(\mathbf{tr}\big(\delta V \delta V^T\big)\big) \le \frac{2\varepsilon^2 m}{3} < \varepsilon^2 m. \tag{12}$$

Then the mean squared error per vector in V is

$$\frac{1}{m} \mathbb{E}\big(\mathbf{tr}\big(\delta V \delta V^T\big)\big) < \varepsilon^2. \tag{13}$$

The number of bits to store the coordinates b_{ij} with precision $\varepsilon' = \frac{\varepsilon}{\sqrt{n}}$ is

$$\begin{aligned} & \sum_{i=1}^{n} \sum_{j=1}^{m} \frac{1}{2} \log_2 \Big(1 + \big(\frac{b_{ij}}{\varepsilon'}\big)^2\Big) = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{m} \log_2 \Big(1 + \frac{b_{ij}^2 n}{\varepsilon^2}\Big) \\ \le \quad & \frac{m}{2} \sum_{i=1}^{n} \log_2 \Big(1 + \frac{n \sum_{j=1}^{m} b_{ij}^2}{m\varepsilon^2}\Big) = \frac{m}{2} \sum_{i=1}^{n} \log_2 \Big(1 + \frac{n\lambda_i}{m\varepsilon^2}\Big). \end{aligned}$$

In the above inequality, we have applied the following inequality:

$$\begin{aligned}&\frac{\log(1+a_1)+\log(1+a_2)+\cdots+\log(1+a_n)}{n}\\ \le\ &\log\Big(1+\frac{a_1+a_2+\cdots+a_n}{n}\Big)\end{aligned}\tag{14}$$

for nonnegative real numbers $a_1, a_2, \ldots, a_n \ge 0$.

Similarly, the number of bits to store the entries of the singular vectors u_{ij} with precision $\varepsilon'' = \frac{\varepsilon\sqrt{m}}{\sqrt{\lambda_i}n}$ is

$$\begin{aligned}&\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{1}{2}\log_2\Big(1+\big(\frac{u_{ij}}{\varepsilon''}\big)^2\Big)=\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\log_2\Big(1+\frac{u_{ij}^2n^2\lambda_j}{m\varepsilon^2}\Big)\\ \le\ &\frac{n}{2}\sum_{j=1}^{n}\log_2\Big(1+\frac{n^2\lambda_j\sum_{i=1}^{n}u_{ij}^2}{m\varepsilon^2}\Big)=\frac{n}{2}\sum_{j=1}^{n}\log_2\Big(1+\frac{n\lambda_j}{m\varepsilon^2}\Big).\end{aligned}$$

Thus, for U and B together, we need a total of

$$L(V)=\frac{m+n}{2}\sum_{i=1}^{n}\log_2\Big(1+\frac{n\lambda_i}{m\varepsilon^2}\Big)=\frac{m+n}{2}\log_2\det\big(I+\frac{n}{m\varepsilon^2}VV^T\big).\tag{15}$$

We thus have proved the statement given in the beginning of this section: $L(V) = (m+n)R(V)$ gives a good upper bound on the number of bits needed to encode V.

APPENDIX B. EXTENSION TO NON-ZERO MEAN CASE

In the above analysis, we have assumed that the given vectors $V = (v_1, v_2, \ldots, v_m)$ are zero-mean. In general, these vectors may have a non-zero mean. In other words, the points represented by these vectors may lie in an affine subspace, instead of a linear subspace.

In case V is not zero mean, let $\mu \doteq \frac{1}{m}\sum_{i=1}^{m} v_i \in \mathbb{R}^n$ and define the matrix

$$M \doteq \mu \cdot 1_{1\times m} = (\mu, \mu, \ldots, \mu) \in \mathbb{R}^{n\times m}.\tag{16}$$

Then $\bar{V} \doteq V - M$ is a matrix whose column vectors have zero mean. We may apply the same coding scheme in the previous section to $\bar{V}$.

Let $\bar{V} = U\Sigma W^T \doteq UB$ be the singular value decomposition of $\bar{V}$. Let $\delta U, \delta B, \delta\mu$ be the error in coding U, B, μ, respectively. Then the error induced on the matrix V is

$$\delta V = \delta\mu \cdot 1_{1\times m} + U\delta B + \delta U B.\tag{17}$$

Assuming that $\delta U, \delta B, \delta\mu$ are zero-mean independent random variables, the expected total squared error is

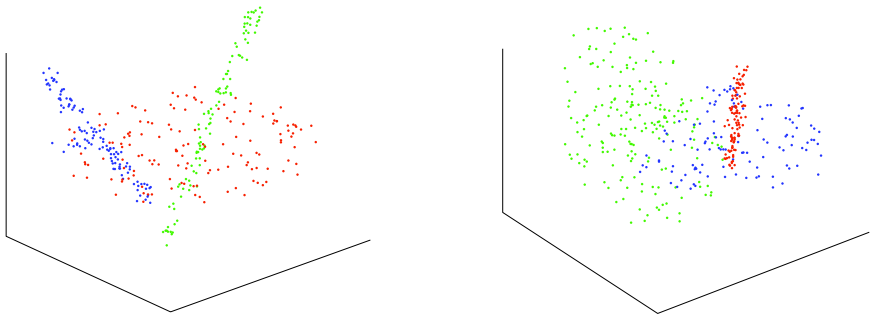
$$\mathbb{E}\big(\mathbf{tr}(\delta V\delta V^T)\big) \quad = \quad m\mathbb{E}(\delta\mu^T\delta\mu) + \mathbb{E}\big(\mathbf{tr}(\delta B\delta B^T)\big) + \mathbb{E}\big(\mathbf{tr}(\Sigma\delta U^T\delta U)\big).\tag{18}$$

We encode entries of B and U with the same precision as before. We encode each entry μ_i of the mean vector μ with the precision $\varepsilon' = \frac{\varepsilon}{\sqrt{n}}$ and assume that the error $\delta\mu_i$ is a uniform distribution in the interval $\big[-\frac{\varepsilon}{\sqrt{n}}, \frac{\varepsilon}{\sqrt{n}}\big]$. Then we have $m\mathbb{E}(\delta\mu^T\delta\mu) = \frac{m\varepsilon^2}{3}$. Using equation (12) for the zero-mean case, the total squared error satisfies

$$\mathbb{E}\big(\mathbf{tr}(\delta V\delta V^T)\big) \le \frac{m\varepsilon^2}{3} + \frac{2m\varepsilon^2}{3} = m\varepsilon^2.\tag{19}$$

Then the mean squared error per vector in V is still bounded by ε^2:

$$\frac{1}{m}\mathbb{E}\big(\mathbf{tr}(\delta V\delta V^T)\big) \le \varepsilon^2.\tag{20}$$



(a) $(2,1,1)$ in $\mathbb{R}^3$. (b) $(2,2,1)$ in $\mathbb{R}^3$.

Figure 6. Segmentation of affine subspaces via Pairwise Steepest Descent on Equation 22.

Now in addition to the $L(\bar{V})$ bits needed to encode U and B, the number of bits needed to encode the mean vector μ with precision $\varepsilon' = \frac{\varepsilon}{\sqrt{n}}$ is

$$\sum_{i=1}^{n} \frac{1}{2} \log_2 \left(1 + \left(\frac{\mu_i}{\varepsilon'}\right)^2\right) = \frac{1}{2} \sum_{i=1}^{n} \log_2 \left(1 + \frac{n\mu_i^2}{\varepsilon^2}\right) \leq \frac{n}{2} \log_2 \left(1 + \frac{\mu^T \mu}{\varepsilon^2}\right), \tag{21}$$

where the last inequality is from the inequality (14).

Thus, the total number bits needed to store V is

$$L(V) \quad = \quad \frac{m+n}{2} \log_2 \det \left(I + \frac{n}{m\varepsilon^2} \bar{V} \bar{V}^T\right) + \frac{n}{2} \log_2 \left(1 + \frac{\mu^T \mu}{\varepsilon^2}\right). \tag{22}$$

Notice that if V is actually zero-mean, we have $\mu = 0$, $\bar{V} = V$, and the above expression for $L(V)$ is exactly the same as before.

This coding length formula can be plugged directly into the Pairwise Steepest Descent algorithm (Algorithm 1 of the paper), and used to segment affine subspaces or non-zero mean Gaussians. Several segmentation results using this formula are shown in Figure 6.

ACKNOWLEDGMENTS

This unnumbered section is used to identify those who have aided the authors in understanding or accomplishing the work presented and to acknowledge sources of funding.

REFERENCES

1. A. Jain, M. Murty, and P. Flynn, "Data clustering: a review," *ACM Computing Surveys* **31**(3), pp. 264–323, 1999.
2. T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, Springer, 2001.
3. A. Dempster, N. Laird, and D. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society B* **39**, pp. 1–38, 1977.
4. T. Cover and J. Thomas, *Elements of Information Theory*, Wiley Series in Telecommunications, 1991.
5. J. Rissanen, "Modeling by shortest data description," *Automatica* **14**, pp. 465–471, 1978.
6. M. A. T. Figueiredo and A. K. Jain, "Unsupervised learning of finite mixture models," *IEEE Transactions on Pattern Analysis and Machine Intelligence* **24**(3), pp. 381–396, 2002.
7. K. Rose, "Deterministic annealing for clustering, compression, classification, regression, and related optimization problems," *Proceedings of the IEEE* **86**(11), pp. 2210–2239, 1998.
8. J. Hamkins and K. Zeger, "Gaussian source coding with spherical codes," *IEEE Transactions on Information Theory* **48**(11), pp. 2980–2989, 2002.

9. H. Benson, "Concave minimization: Theory, applications and algorithms," in R. Horst and P.M. Pardalos, eds., *Handbook of Global Optimization* , 1994.
10. D. Donoho, M. Vetterli, R. DeVore, and I. Daubechies, "Data compression and harmonic analysis," *IEEE Transactions on Information Theory* **44**(6), pp. 2435–2476, 1998.
11. S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge University Press, 2004.
12. Y. Ma, H. Derksen, W. Hong, and J. Wright, "Segmentation of multivariate mixed data via lossy coding and compression," *Technial Report UILU-ENG-06-2203, University of Illinois at Urbana-Champaign.* , 2006.