LINEAR MANIFOLD CORRELATION CLUSTERING

Rave HARPAZ [1], **Robert HARALICK** [1]

[1]*Pattern Recognition Laboratory, CS Dept.*
The Graduate Center, City University of New York
New York, N.Y.
E-mail: rbharpaz@sci.brooklyn.cuny.edu, haralick@netscape.com

Abstract

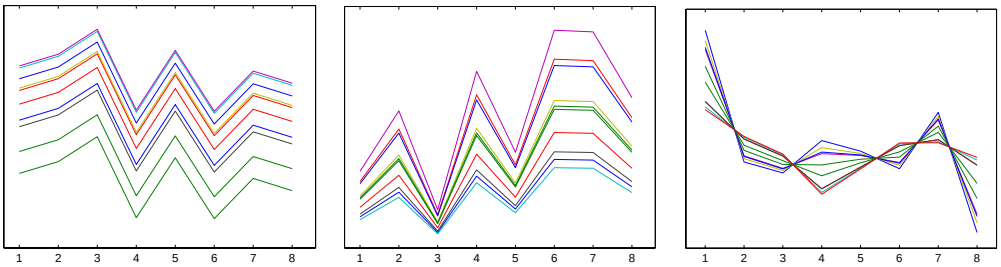
The detection of correlations is a data mining task of increasing importance due to new areas of application such as DNA microarray analysis, collaborative filtering, and text mining. In these cases object similarity is no longer measured by physical distance, but rather by the behavior patterns objects manifest or the magnitude of correlations they induce. Many approaches have been proposed to identify clusters complying with this requirement. However, most approaches assume specific cluster models, which in turn may lead to biased results. In this paper we present a novel methodology based on linear manifolds which provides a more general and flexible framework by which correlation clustering can be done. We discuss two stochastic linear manifold cluster models and demonstrate their applicability to a wide range of correlation clustering situations. The general model provides the ability to capture arbitrarily complex linear dependencies or correlations. The specialized model focuses on simpler forms of linear dependencies, yet generalizes the dependencies often sought by the so called "pattern" clustering methods. Based on these models we discuss two linear manifold clustering algorithms, the later a fine-tuned derivative of the first targeting simpler forms of correlation and "pattern" clusters. The efficacy of our methods is demonstrated by a series of experiments on real data from the microarray and collaborative filtering domains. One of the experiments demonstrates that our method is able to identify statistically significant correlation clusters that are overlooked by existing methods.

Keywords: linear manifold, clustering, correlation, patterns.

1 Introduction

Classical clustering methods as well as subspace methods [1, 2, 3, 4] focus on grouping objects with similar values. They define object similarity by the "physical" distance between the objects over all or a subset of dimensions, which in turn may not be adequate to capture correlations in the data. A set of points may be located far away from each other yet induce large correlations among some subset of dimensions. The detection of correlations is a an important data mining task because correlations may reveal a dependency or some cause and effect relationship between the features under consideration. Another important application of correlations is in data modeling where correlations may be used to carry out (local) dimensionality reduction by eliminating correlated (redundant) features. In recent studies correlations were often discussed and presented in terms of the behavior patterns objects manifest, hence the name *pattern clustering* often associated with methods amid at this type of problem. In gene expression microarray clustering the goal is to identify groups of genes that exhibit similar expression patterns under some subset of conditions (dimensions), from which gene function or regulatory mechanisms may be inferred. In recommendation or collaborative filtering systems, sets of customers with similar interest patterns need to be identified so that customers' future interests can be predicted and proper recommendations be made. From a correlation point of view it can be shown that objects exhibiting coherent behavior patterns induce large correlations among their defining features. Hence, the identification of large correlations is a means by which *pattern clusters* can also be discovered.

The most widely studied *pattern cluster models* are the *shift* and *scaling* models, which induce only positive correlations and are typically referred to as *biclusters* [5, 6, 7] in the microarray clustering literature, and *slope one* clusters in the collaborative filtering literature [8]. In the case of a shift pattern the behavior pattern of one object under a set of features is offset from another by some constant, whereas in the case of scaling the behavior pattern of one object is a scaler multiple of another. The corresponding linear dependencies between each pair of features x_i, x_j captured by these two types of patterns are of the form $x_j = x_i + c_{ij}$ and $x_j = b_{ij}x_i$ respectively, where b_{ij} and c_{ij} are constant coefficients. From the above it is now clear why the shift and scaling patterns induce large correlations. Fig. 1 shows *parallel coordinate* plots of three different types of patterns clusters each containing ten points embedded in an 8-dimensional space: a shift pattern inducing only positive correlations, a scaling pattern also inducing only positive correlations, and a pattern inducing both positive and negative correlations. These type of plots



(a) shift pattern **(b)** scaling pattern **(c)** negative correlations

Figure 1. Parallel coordinate plots of three different pattern clusters.

are used to emphasize symmetry or cohesion in behavior patterns. Note that the pattern inducing negative correlations does not manifest the same symmetry as the other two.

It has recently been suggested [9, 10, 11, 12] that other types of information carrying patterns such as patterns inducing negative correlations or patterns capturing more complex linear dependencies, such as $x_j = b_{ij}x_i + c_{ij}$ of which the shift and scaling are special cases, are completely overlooked by most clustering methods, and that current state of the art algorithms are not flexible enough to mine different patterns simultaneously. It has also been suggested [7] that traditional similarity measures, such as the *cosine* or the *Pearson correlation* measures, are not adequate to capture correlations or pattern clusters when those localize to subspaces of the data, as they are strongly biased by the data residing the in the "irrelevant" dimensions. While there is no consensus on what types of patterns should be considered meaningful, in practice pattern based clustering algorithms postulate a unique underlying "globally expressed" pattern or cluster model, while overlooking or rejecting the possibility that other types of information carrying patterns may co-exist in the data. This in turn is less truthful to the data and may lead to a large bias in the results. Therefore, a method which would target correlations in general and not specific models would be beneficial.

Avoiding (for the moment) mathematical abstraction, a *linear manifold* is simply a translated subspace, which can be visualized as a line, plane, hyperplane, etc., depending on its dimensionality. In many problem domains it assumed that linear models are sufficient enough to describe or capture the data's inherent structure. Typical examples include linear regression, PCA, and subspace clustering, which are all special cases of linear manifold learning. Yet very few remote attempts have been made to devise clustering methods able to

identify or learn **mixtures** of linear manifolds. Moreover, very often observed real data is a consequence of a process governed by a small number of factors. In the data space this is manifested by the data points lying or being located close to surfaces such as linear or non-linear manifolds whose intrinsic dimensionality is much smaller than the dimensionality of the data. *Linear manifold clustering* [13] seeks to identify groups of points that fit or are embedded in lower dimensional linear manifolds. One of the main advantages of the linear manifold clustering paradigm is that it is applicable to a wide range of clustering applications or problem domains. This is because a linear manifold is a generalization of more common and specific cluster models. It can easily be shown that "classical" clusters (hyper-spherical/ellipsoidal in shape) such as the ones sought by the K-means algorithm [14], and subspace clusters such as those discussed in [1, 2, 15] are special cases of linear manifolds. In the context of correlation, it can be shown that common to all forms of linear correlation and linear dependencies, is that in the data space they manifest themselves as lines, planes, and generally speaking as linear manifolds [16, 17, 18]. Hence, the detection of linear manifolds is a means by which correlations or linear dependencies may also be identified.

Correlations correspond to linear dependencies between two or more features (variables or attributes) of the data, and can therefore be discussed from this view point. These linear dependencies can be a simple as those sought by pattern clustering methods or may be more complex where one or more features are linearly dependent on a combination of others. Needless to say the more complex dependencies are harder to interpret which is why many methods focus on simpler models.

In the following we discuss the application of linear manifold clustering to the problem of correlation clustering. In section 2 we discuss a general *linear manifold cluster model* and demonstrate its ability to capture arbitrarily complex linear dependencies. In section 3 we discuss a *line cluster model*, which is a specialization of the linear manifold cluster model. The line cluster model targets simpler linear dependencies of the form $x_j = b_{ij}x_i + c_{ij}$, generalizing the type of correlations sought by pattern clustering methods, and thus rectifying the their shortcomings. In sections 4 and 5 we present two algorithms. The first called LMCLUS (Linear Manifold CLUStering) aimed at identifying general linear manifold clusters, and the second called SLCLUS (Subspace Line CLUStering) a fine-tuned specialization aimed at line clusters which are embedded in axis-parallel subspaces. In section 6 we present a series of experiments on real data sets demonstrating the efficacy of our methods, and in section 7 we conclude the paper.

2 The Linear Manifold Cluster Model

A linear manifold is a subspace that may have been translated away from the origin. A subspace is a special case of a linear manifold that contains the origin. Geometrically, a 1D manifold can be visualized as a line embedded in the space, a 2D manifold as a plane, and a 0D manifold as a point. Classical clustering algorithms such as K-Means assume that each cluster is associated with 0D manifold (a point typically the cluster center), and therefore omit the possibility that a cluster may have a non-zero dimensional linear manifold associated with it. For the sake of completeness we give a formal definition of a linear manifold.

Definition 1 (Linear Manifold) *L is a **linear manifold** of vector space V if and only if for some subspace S of V and translation $t \in V$, $L = \{x \in V | for\ some\ s \in S, x = t + s\}$. The dimension of L is the dimension of S, and if the dimension of L is one less than the dimension of V then L is called a **hyperplane**. A linear manifold L is rectangularly bounded if and only if for some translation t and bounding vectors a_L and a_H, $L = \{x \in V | for\ some\ s \in S, a_L \leq s \leq a_H, x = t + s\}$. A rectangularly bounded linear manifold has finite extent and is localized with center $t + \frac{a_L + a_H}{2}$. In the case that $a_L = -a_H$, its center is the translation t.*

The *linear manifold cluster model* has the following properties: The points in each cluster are embedded in a lower dimensional linear manifold of finite extent. The intrinsic dimensionality of the cluster is the dimensionality of the linear manifold. The manifold is arbitrarily oriented. The points in the cluster induce a correlation among two or more attributes (or a linear transformation of the original attributes) of the data set. The points in the orthogonal complement space to the manifold form a compact densely populated region. More formally let D be a set of d-dimensional points, $X \subseteq D$ be the subset of points that belong to a cluster, $\mathbf{x}$ be a $d \times 1$ vector representing some point in X, $\mathbf{b}_1, \ldots, \mathbf{b}_d$ be a set of orthonormal vectors that span a d-dimensional space, B be a $d \times k$ matrix whose k columns are a subset of the vectors $\mathbf{b}_1, \ldots, \mathbf{b}_d$, and $\overline{B}$ be a $d \times (d - k)$ matrix whose columns are the remaining vectors.

Definition 2 (The Linear Manifold Cluster Model) *Let $\boldsymbol{\mu}$ be some point in $\mathbb{R}^d$, $\boldsymbol{\phi}$ be a zero mean $k \times 1$ random vector whose entries are i.i.d. on support $(-R/2, +R/2)$ where R is the range of the data, and $\boldsymbol{\epsilon}$ be a zero mean $(d-k)\times 1$ random vector with small variance independent of $\boldsymbol{\phi}$. Then each $\mathbf{x} \in X$, a k-dimensional linear manifold cluster, is modeled by,*

$$\mathbf{x} = \boldsymbol{\mu} + B\boldsymbol{\phi} + \overline{B}\boldsymbol{\epsilon}. \tag{1}$$

The idea is that each point in a cluster lies close to a k-dimensional linear manifold of finite extent, which is defined by $\boldsymbol{\mu}$, a translation vector, the space spanned by the columns of B, and the range parameter R. Since

$$E[\mathbf{x}] = E[\boldsymbol{\mu} + B\boldsymbol{\phi} + \overline{B}\boldsymbol{\epsilon}] = \mu + BE[\boldsymbol{\phi}] + \overline{B}E[\boldsymbol{\epsilon}] = \boldsymbol{\mu} + \mathbf{0} + \mathbf{0} = \boldsymbol{\mu}$$

the cluster mean is $\boldsymbol{\mu}$. On the manifold the points are assumed to be distributed in each direction (the k column vectors of B) on bounded support $(-R/2, +R/2)$. It is in this manifold that the cluster is embedded, and therefore the intrinsic dimensionality of the cluster will be k. What characterizes this type of cluster is the third component that models a small random error associated with each point on the manifold. The idea is that each point may be perturbed in directions that are orthogonal to the subspace spanned by the columns of B, that is the subspace defined by the $d-k$ columns of $\overline{B}$. We model this behavior by requiring that $\boldsymbol{\epsilon}$ be a $(d-k)\times 1$ random vector, distributed with mean $\mathbf{0}$ and covariance Σ, where the square root of the largest eigenvalue of Σ is much smaller than R, the range of the data.

Traditional "full-space" clustering algorithms take $k = 0$, and therefore assume that each point in a cluster can be modeled by $\mathbf{x} = \boldsymbol{\mu} + \overline{B}\boldsymbol{\epsilon}$ where $\overline{B}$ is simply the identity matrix. Subspace clustering algorithms focus their clustering effort on the space spanned by the column vectors of $\overline{B}$, and when restricted to axis parallel subspaces, they assume both B and $\overline{B}$ contain columns of the identity matrix.

As mentioned linear manifolds also capture correlations or linear dependencies. More specifically, each linear manifold cluster gives rise to a system of linear equations that capture an arbitrarily complex set of linear dependencies. Just as a surface can be defined by the normal vector perpendicular to it. A linear manifold can be defined by the set of vectors which are orthogonal to it. That is, a linear manifold can be defined by the following vector equation,

$$\overline{B}'(\mathbf{x} - \boldsymbol{\mu}) = \mathbf{0}, \tag{2}$$

which essentially describes the linear dependencies induced by the linear manifold points. To see this more clearly we can rearrange eq. (2) as $\overline{B}'\mathbf{x} = \overline{B}'\boldsymbol{\mu}$ and rewrite the above system as a linear combination of scalars

$$\begin{array}{ccccccccc}
b_{k+1,1}x_1 & + & b_{k+1,2}x_2 & +, & \ldots & ,+ & b_{k+1,d}x_d & = & c_{k+1} \\
b_{k+2,1}x_1 & + & b_{k+2,2}x_2 & +, & \ldots & ,+ & b_{k+2,d}x_d & = & c_{k+2} \\
 & & & & \vdots & & & & \\
b_{d,1}x_1 & + & b_{d,2}x_2 & +, & \ldots & ,+ & b_{d,d}x_d & = & c_d
\end{array} \tag{3}$$

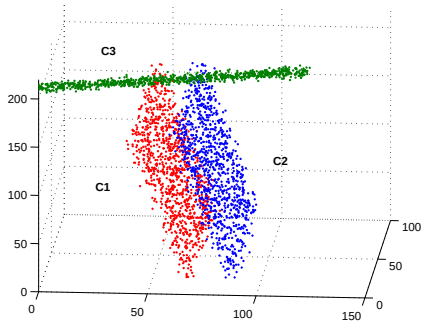


Figure 2. Sample data set of three clusters, each of which is embedded in a different linear manifold of one ($C3$) or two dimensions ($C1, C2$).

where $\mathbf{b}_{k+1}, \mathbf{b}_{k+2}, \ldots, \mathbf{b}_d$ are the column vectors of $\overline{B}$, $b_{i,j}$ the j-th component of $\mathbf{b}_i$, $c_{k+i} = \mathbf{b}'_{k+i}\boldsymbol{\mu}$, and x_i the i-th feature or attribute of the data. Note that the number of equations or linear dependencies is equal to $d-k$ the dimensionality of the space orthogonal to the linear manifold. Furthermore, using *Gauss-Jordan elimination* this set of equations can be simplified and put in *reduced row echelon* to produce a unique description of the linear dependencies. As a special case, note that a $d-1$-dimensional linear manifold can be defined by one equation of the form

$$b_{d,1}x_1 + b_{d,2}x_2 + , \ldots, + b_{d,d}x_d = c_d,$$

which if rearranged gives exactly a *regression model* that describes the relationship between a *response* variable (can be any of the $x'_i s$) and a set of *explanatory* or *predictor* variables (the remaining x_i's).

Fig. 2 is an example of data set modeled by eq. (1). The data set contains three non-overlapping clusters C_1, C_2, C_3 each consisting of 1000 points. C_1, C_2 which are almost planar and parallel to each other are embedded in $2D$ linear manifolds. Their points are uniformly distributed in the manifold and they include a small error term in the space complementary to the manifold. Similarly, C_3 an elongated line-like cluster, is embedded in a $1D$ linear manifold with an error element in the $2D$ space complementary to the manifold.

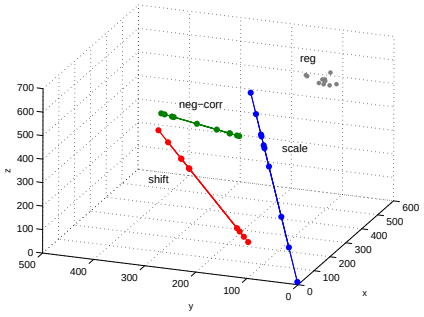


Figure 3. The geometry of the three pattern clusters in the data space. All are manifested by lines but are oriented and translated differently depending on the type of pattern they manifest. A regular cluster (reg) is also plotted to emphasize the difference between correlation and regular clusters.

3 The Line Cluster Model

Line clusters arise naturally from the kind of patterns that are often sought in data (e.g., shift, scaling, negative correlations, and generally correlations induced by the linear dependency $x_j = b_{ij}x_i + c_{ij}$). Each of these kinds of patterns in its ideal form is a line cluster. Figure 3 shows how these patterns look from the point of view of line clusters.

In terms of our formal definition, a line cluster occurs when B is just a $k \times 1$ vector spanning a 1D subspace, $\overline{B}$ is a $k \times k-1$ matrix whose $k-1$ column vectors form an orthonormal basis that spans the space orthogonal to the space spanned by B. In following we will use $\boldsymbol{\beta}$ and $\overline{\boldsymbol{\beta}}$ instead of B and $\overline{B}$ to distinguish between the two cases.

Definition 3 (The Line Cluster Model) *Let $\boldsymbol{\mu}$ be some point in $\mathbb{R}^k$, ϕ be a zero mean random scalar distributed on the support $(-R/2, +R/2)$ where R is the range of the data, and $\boldsymbol{\epsilon}$ is a $(k-1) \times 1$ random vector having mean $\mathbf{0}$ and covariance matrix $\sigma^2 I$, where $\sigma \ll R$. Then each $\mathbf{x} \in X$, a k-dimensional line cluster, is modeled by,*

$$\mathbf{x} = \boldsymbol{\mu} + \boldsymbol{\beta}\phi + \overline{\boldsymbol{\beta}}\boldsymbol{\epsilon}. \tag{4}$$

Note that in this case ϕ is a scalar, and homoscedasticity (constant variance) is assumed about the error term $\boldsymbol{\epsilon}$. This assumption is common to other quantitative models such as the shift, scaling, and regression models, and simplifies the line cluster model making statistical inference easier.

The interesting property of ideal line clusters ($\sigma^2 = 0$) is that the correlation between variables (features of the data) is +1 or -1. This is easy to see. Let the density function of the cluster be f. Without loss of generality, f can be taken to be the density of the scaler random variable ϕ, which has mean 0 and variance σ_ϕ^2. Let σ_{ij} be the covariance between variables x_i and x_j. Let $\mu' = (\mu_1, \mu_2, \ldots, \mu_k)$ and let $\boldsymbol{\beta}' = (\beta_1, \beta_2, \ldots, \beta_k)$. Since $\sigma^2 = 0$, $\boldsymbol{\epsilon}$ is effectively 0, and the model for $\mathbf{x}$ simplifies to

$$\mathbf{x} = \boldsymbol{\mu} + \boldsymbol{\beta}\phi. \tag{5}$$

Then,

$$\sigma_{ij} = E[(\mu_i + \phi\beta_i - \mu_i)(\mu_j + \phi\beta_j - \mu_j)] = \beta_i\beta_j\sigma_\phi^2$$

Hence, $\sigma_{ii} = \beta_i^2\sigma_\phi^2$ and $\sigma_{jj} = \beta_j^2\sigma_\phi^2$. By definition, the correlation ρ_{ij} between the variables x_i and x_j is defined by

$$\begin{aligned} \rho_{ij} &= \sigma_{ij} / \sqrt{\sigma_{ii}\sigma_{jj}} \\ &= \beta_i\beta_j\sigma_\phi^2 / \sqrt{\beta_i^2\sigma_\phi^2\beta_j^2\sigma_\phi^2} \\ &= \pm 1. \end{aligned}$$

It can also be shown that correlations give rise to line clusters [17], and that the more a set of points deviates from a predefined line (the parameters $\boldsymbol{\mu}$ and $\boldsymbol{\beta}$ are fixed), the less correlated the features underlying the points will be, where the amount of correlation depends on the size of σ^2.

4 The General Algorithm

LMCLUS can be viewed as an hierarchical-divisive clustering procedure. It executes three levels of iteration (Fig. 4), and expects three inputs: L, an upper limit on the dimension of the linear manifolds in which we believe clusters may be embedded; S, a sampling level parameter used to determine the number of trial linear manifolds of a given dimensionality that will be examined in order to reveal the best possible partitioning of a given set of points; Γ, a sensitivity or "goodness of separation" threshold, which is used to determine whether or not a partitioning should take place based on a trial linear manifold.

At the highest level of iteration the algorithm monitors the size of the data which is being partitioned. When no data is left to be partitioned the algorithm terminates. The second level of iteration causes the algorithm to iterate over a range of manifold dimensionalities, commencing with one-dimensional manifolds, and terminating with L-dimensional manifolds. For each linear manifold

dimension the algorithm enters the third level of iteration, in which *FindSeparation* (Fig. 5) is invoked in an attempt to reveal separations among subsets of the data and to determine whether some of the points are embedded in linear manifolds.

The idea behind *FindSeparation* is to successively sample points that can define a linear manifold of a given dimension, and select the linear manifold that is closest to a substantial number of points. This subset of closest points will typically correspond to a cluster. The proximity of the input data points to the manifold is captured by a histogram of the distances the points are from the manifold. If the manifold indeed has some subset of points near it, then the distance histogram will have a mixture of two distributions. One of the distributions has a mode near zero and arises from the distances of points belonging to a cluster. The other distribution arises from the points not belonging to a cluster.

The problem of separating the cluster points from the rest is then cast into a histogram thresholding problem. Upon termination *FindSeparation* returns four values γ- which is a measure of the "goodness" of the separation, τ- a proximity threshold that is computed from the histogram and is used to split the data into two groups, β- the basis of the manifold which exposes the separation, and o-a point on the manifold representing its origin. When γ exceeds the value of the input sensitivity threshold parameter Γ, indicating that a worthy separation has been found, then the data set is split according to τ. This split corresponds to the partitioning of all the points which are located close enough to the just determined manifold, i.e. all points that potentially belong to a given cluster, and those that belong to other clusters.

The third iteration continues reapplying *FindSeparation* in an attempt to further partition the cluster which may consist of sub-clusters, until the selected data points can not be further separated. At this point the algorithm will retract to the second level of iteration in an attempt to partition the cluster in higher dimensions, a process which will continue until the dimension limit L is reached. When L is reached we have a subset of the points that cannot be partitioned any more, and declare that a cluster is found.

The algorithm then retracts to the first level of iteration and is reapplied on the remaining set of points until no more points are left to be partitioned, detecting one cluster at a time. We note that if outliers exist then the last cluster/partition that is found will contain this set of points. By definition outliers do not belong to any cluster and therefore will remain the last group of points to be associated to any other group. Since they are unlikely to form any clusters the algorithm will not be able to partition them, and they will therefore be

all grouped together.

```
Algorithm LMCLUS (D, L, S, Γ)
  C = ∅  # set of labeled clusters initially empty.
  Dims = ∅  # set of intrinsic dimensionalities of each cluster.
  i = 1  # cluster label
  while D ≠ ∅ do
      X = D
      for k = 1 to L do
          while [γ, τ, o, β] = FindSeparation(X, k, S), γ > Γ do
              # a separation is revealed by a k-dimensional
              # manifold. Collect all points residing in the vicinity
              # of that manifold.
              X = {x| x ∈ Y, || (x − o) ||^2 − || β'(x − o) ||^2 < τ}
              LmDim = k
      # a cluster is found, add it to the set of labeled clusters.
      C_i = X, C = C ∪ {C_i}
      # recored the intrinsic dimensionality of the cluster
      dim_i = LmDim, Dims = Dim ∪ {dim_i}
      i = i + 1
      # remove cluster points from the dataset.
      D = D − X
  return [C, Dims]
```

Figure 4. The linear manifold clustering algorithm.

4.1 Finding Separations

Let $D, X, B, \overline{B}$, and $\boldsymbol{\mu}$ be as defined in section 2. The distance of a point $\mathbf{x} \in D$ to a linear manifold defined by $\boldsymbol{\mu}$ and the column vectors of B is given by

$$\|(I - BB^T)(\mathbf{x} - \boldsymbol{\mu})\| = \|\overline{B}\overline{B}^T(\mathbf{x} - \boldsymbol{\mu})\|. \tag{6}$$

As mentioned earlier in section 2 the points of X are likely to form a compact and dense region in the space orthogonal to the manifold in which they are embedded. Therefore by projecting D into the space spanned the column vectors of $\overline{B}$ and executing some form of clustering in the reduced space it is possible to identify and separate X from the rest of the data. However, eq. (6) shows that the distance of a point to the cluster center in the reduced space is equivalent to the distance of a point to the linear manifold. Thus, rather than clustering in the reduced space it is also possible to measure distances from the manifold and collect all the points that lie in the vicinity of this manifold, essentially executing one-dimensional clustering. Since we are interested in estimating B, and because we are interested in detecting one cluster at a time, and since one-dimensional clustering is typically faster than clustering in higher dimensions, we choose to take this path.

Algorithm FindSeparation $(X,\ k,\ S)$
  $\gamma = -\infty,\ \tau = -\infty,\ o = \mathbf{0},\ \beta = \mathbf{0}$
  $N = \log \epsilon / \log(1 - (1/S)^k)$
  for $i = 1$ **to** N **do**
    $M =$ **Sample** $(k+1)$ *points from* X
    $O = x \in M$
    $B =$ **FormOrthonormalBasis**$(M,\ O)$
    $Distances = \emptyset$
    for *each* $x \in X - M$ **do**
      $y = x - O$
      $Distances = Distances \cup \{\| y \|^2 - \| B'y \|^2\}$
    $H =$ **MakeHistogram**$(Distances)$
    $T =$ **FindMinimumErrorThreshold**(H)
    $G =$ **EvaluateGoodnessOfSeparation**$(T,\ H)$
    if $G > \gamma$ **then**
      $\gamma = G,\ \tau = T,\ o = O,\ \beta = B$
  return $[\gamma,\ \tau,\ o,\ \beta]$

Figure 5. Detecting separations among clusters embedded in lower dimensionality linear manifolds.

Lemma 1 $\|(I - BB^T)(\mathbf{x} - \boldsymbol{\mu})\| = \sqrt{\|(\mathbf{x} - \boldsymbol{\mu})\|^2 - \|B^T(\mathbf{x} - \boldsymbol{\mu})\|^2}$

Proof: Let $\mathbf{y} = \mathbf{x} - \boldsymbol{\mu}$,

$$\begin{aligned}
\|(I - BB')\mathbf{y}\|^2 &= \|\mathbf{y} - BB'\mathbf{y}\|^2 \\
&= (\mathbf{y} - BB'\mathbf{y})'(\mathbf{y} - BB'\mathbf{y}) \\
&= \mathbf{y}'\mathbf{y} - 2\mathbf{y}'BB'\mathbf{y} - \mathbf{y}'(BB')^2\mathbf{y} \\
&= \mathbf{y}'\mathbf{y} - \mathbf{y}'BB'\mathbf{y} \qquad (\ BB' \text{ is idempotent so } (BB')^2 = BB'\ ) \\
&= \|\mathbf{y}\|^2 - \|B'\mathbf{y}\|^2.
\end{aligned}$$

Lemma 1 provides us a much more efficient way of computing the distance of a point to a manifold. If d is the dimension of the data, then computing the distance using lemma 1 gives us a speedup of $O(d)$, which for high dimensional data becomes a significant factor. To simplify the computation even further we choose to use the squared distance rather than the distance, henceforth we will use the term "distance" to mean the squared distance.

4.1.1 Minimum Error Thresholding

The problem of separating all the points that lie in the vicinity of a manifold can be cast into the problem of finding a *minimum error threshold* that is used to classify points as either embedded in the manifold (belonging to X) or not, based on their distances to the manifold. Kittler and Illingworth [19] (KI) describe an efficient method for finding the minimum error threshold. Their

method was designed for segmenting an object from its background in gray scale images using a grey level histogram of the image. Their method views the histogram segmentation problem as a two class classification problem, where the goal is to minimize the number of misclassified pixels. Analogous to our problem, the distance histogram can be viewed as an estimate of the probability density function of the mixture population comprising of distances of points belonging to a linear manifold cluster and those that do not.

The KI procedure is based on the assumption that each component of the mixture is normally distributed. To support this assumption in our case, we note that as a consequence to the central limit theorem, the distances to the manifold which are merely sums of random variables will approach distribution-wise the normal, as the dimension of the space increases.

Let δ be the distance of a point to the manifold, $p(\delta|i)$ be the probability density function of the distances of class i, where $i \in \{1, 2\}$, μ_i, σ_i be the mean and standard deviation of distances in class i, and P_i be the prior of class i. Then because of the normality assumption

$$p(\delta|i) = \frac{1}{\sqrt{2\pi}\sigma_i} \exp\left(\frac{-(\delta - \mu_i)^2}{2\sigma_i^2}\right).$$

Given μ_i, σ_i, P_i, and $p(\delta|i)$ there exists a threshold τ such that

$$P_1 p(\delta|1) > P_2 p(\delta|2) \quad \text{if} \quad \delta \leq \tau \qquad \text{and} \qquad P_1 p(\delta|1) < P_2 p(\delta|2) \quad \text{if} \quad \delta > \tau,$$

where τ is the *Bayes minimum error threshold* [20], which can be found by solving for δ the following equation

$$P_1 \frac{1}{\sqrt{2\pi}\sigma_1} \exp\left(\frac{(\delta - \mu_1)^2}{-2\sigma_1^2}\right) = P_2 \frac{1}{\sqrt{2\pi}\sigma_2} \exp\left(\frac{(\delta - \mu_2)^2}{-2\sigma_2^2}\right).$$

However, the true values of μ_i, σ_i, P_i are usually unknown. KI propose to obtain these estimates from the distance histogram h. Suppose that the histogram is thresholded at an arbitrary threshold t, then we can model the two resulting populations by a normal density $h(\delta|i, t)$ with parameters:

$$P_i(t) = \sum_{\delta=a}^{b} h(\delta), \qquad \mu_i(t) = \frac{\sum_{\delta=a}^{b} \delta * h(\delta)}{P_i(t)}, \qquad \sigma_i^2(t) = \frac{\sum_{\delta=a}^{b} (\delta - \mu_i(t))^2 * h(\delta)}{P_i(t)}$$

where $a = 0$ and $b = t$ if $i = 1$, and $a = t + 1$ and $b = max(\delta)$ if $i = 2$. Now using the model $h(\delta|i, t)$ for $i \in \{1, 2\}$, the conditional probability of δ being correctly classified is given by

$$p(i|\delta, t) = \frac{h(\delta|i, t)P_i(t)}{h(\delta)}.$$

We wish to find the threshold t that maximizes this probability. Since $h(\delta)$ is independent of i and t it can be safely ignored. Furthermore, since the logarithm is a strictly increasing function, taking the logarithm and multiplying by a constant will not change the maximizing value. Therefore

$$\epsilon(\delta, t) = \left(\frac{\delta - \mu_i(t)}{\sigma_i(t)}\right)^2 + 2\log\sigma_i(t) - 2\log P_i(t)$$

can be considered as an alternative index of the correct classification performance, and the overall performance is given by

$$J(t) = \sum_{\delta} h(\delta)\epsilon(\delta, t),$$

which reflects indirectly the amount of overlap between the two Gaussian populations. Substituting $\mu_i(t), \sigma_i(t), P_i(t)$, and $\epsilon(\delta, t)$ into $J(t)$ we get

$$J(t) = 1+2\left(P_1(t)\log\sigma_1(t) + P_2(t)\log\sigma_2(t)\right)-2\left(P_1(t)\log P_1(t) + P_2(t)\log P_2(t)\right), \tag{7}$$

and the minimum error threshold selection problem can be formulated as

$$\tau = \arg\min_t J(t).$$

$J(t)$ can be computed easily and finding its minima is a relatively simple task as the function is smooth.

4.2 Sampling Linear Manifolds

A line, which is a 1D linear manifold, can be defined by two points, a plane which is a 2D manifold can be defined using three points. To construct a random k-dimensional linear manifold by sampling points from the data we need to sample $k + 1$ linearly independent points. Let $\mathbf{x}_0, \ldots \mathbf{x}_k$ denote these points. Choosing one of the points, say $\mathbf{x}_0$ as the origin, the k vectors spanning the manifold are given by

$$\mathbf{y}_i = \mathbf{x}_i - \mathbf{x}_0$$

where $i = 1 \ldots k$. Assuming each of these sampled points came from the same cluster, then according to eq. (1)

$$\mathbf{y}_i = \mathbf{x}_i - \mathbf{x}_0 = (\boldsymbol{\mu}_0 + B\boldsymbol{\phi}_i + \overline{B}\boldsymbol{\epsilon}_i) - (\boldsymbol{\mu}_0 + B\boldsymbol{\phi}_0 + \overline{B}\boldsymbol{\epsilon}_0)$$

$$= B(\boldsymbol{\phi}_i - \boldsymbol{\phi}_0) + \overline{B}(\boldsymbol{\epsilon}_i - \boldsymbol{\epsilon}_0).$$

If the cluster points did not have an error component off the manifold, i.e., they all lie on the linear manifold, then sampling any $k+1$ points which are linearly independent and belong to the same cluster would enable us to reconstruct B. So in order to get a good approximation of B we would like each of the sampled points to come from the same cluster and to be as close as possible to the linear manifold. From the equation above we see that this occurs when

$$\boldsymbol{\epsilon}_i - \boldsymbol{\epsilon}_0 \approx \mathbf{0},$$

resulting in a set of k vectors which are approximately a linear combination of the original vectors in B. A good indication as to why this is likely to occur when the sampled points come from the same cluster, is given by the fact that

$$E[\boldsymbol{\epsilon}_i - \boldsymbol{\epsilon}_0] = \mathbf{0},$$

and that normally distributed data ($\boldsymbol{\epsilon}_i - \boldsymbol{\epsilon}_0$ follows a normal distribution) tends to cluster around its mean. In cases where the clusters are well separated, the requirement that $\boldsymbol{\epsilon}_i - \boldsymbol{\epsilon}_0 \approx \mathbf{0}$ can be relaxed. That is, when the clusters are well separated more sets of points coming from the same cluster, and not only those that are relatively close to the manifold will be good candidates to a construct a manifold that will induce a large valley in the distance histogram that separates the linear manifold cluster from the remaining points. As a consequence, the problem of sampling a linear manifold that will enable us to separate a linear manifold cluster from the rest of the data can be reduced to the problem of sampling $k+1$ points that all come from the same cluster.

Assuming there are S clusters in the data set whose size is distributed with low variance, then for large data sets the probability that a sample of $k+1$ points all come from the same cluster is approximately

$$\left(\frac{1}{S}\right)^k.$$

If we want to ensure with probability $(1-\epsilon)$ that at least one of our random samples of $k+1$ points all come from the same cluster, then we should expect to make at least n selections of $k+1$ points, where

$$\left[1 - \left(\frac{1}{S}\right)^k\right]^n \leq \epsilon,$$

yielding that

$$n \geq \frac{\log\ \epsilon}{\log(1-(1/S)^k)}. \tag{8}$$

Therefore, by computing n given ϵ, and S which is an input to the algorithm we can approximate a lower bound on the number of samples required, such that with high probability, at least one of the n samples contains $k+1$ points that all come from the same cluster. Unlike related methods, the user input S does not predetermine the number of clusters LMCLUS will output. It is a rough estimate of the number of clusters in the data set, which is only used to compute an initial estimate of the sample size required to ensure we sample points coming from the same cluster. It is rather a "gauge" with which we can tradeoff accuracy with efficiency.

Putting it all together, for each sample of points $\{\mathbf{y}_1, \ldots, \mathbf{y}_k\}$ we construct an orthonormal basis B of a linear manifold using the Gram-Schmidt process. (If the Gram-Schmidt process indicates that the sampled points are not linearly independent, a new sample of points is taken.) Then using KI's method we compute a threshold τ. Of all possible thresholds corresponding to different linear manifolds which induce different separations, we prefer the one which induces the best separation. The best separation is defined as the separation which induces the largest discriminability given by

$$discriminability = \frac{(\mu_1(\tau) - \mu_2(\tau))^2}{\sigma_1(\tau)^2 + \sigma_2(\tau)^2}, \tag{9}$$

and the one which causes the deepest broadest minimum in KI's criterion function-$J(t)$. The deepest minimum can be quantified by computing the difference/depth of the criterion function evaluated at the minimum τ and the value evaluated at the closest local maxima τ', i.e.

$$depth = J(\tau') - J(\tau).$$

The composite measure of the goodness of a separation is then given by

$$G = discriminability \times depth. \tag{10}$$

A typical run of the algorithm is illustrated in Figs. 6 and 7 by a tree which summarizes the clustering process of the sample data set depicted in Fig. 2, and the corresponding histograms that were used to separate the linear manifold clusters in this data set. At the beginning of the process the algorithm searches for 1D linear manifolds in which some clusters may be embedded. Since cluster C_3 is such a cluster it is separated from C_1, C_2 using the threshold returned by KI's procedure, and the algorithm proceeds by trying to further partition it in higher dimensions. Since it cannot be further partitioned using 2D manifolds, cluster C_3 is declared to be found. The algorithm then attempts

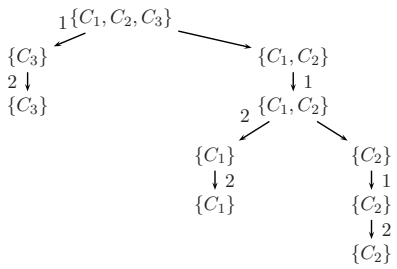


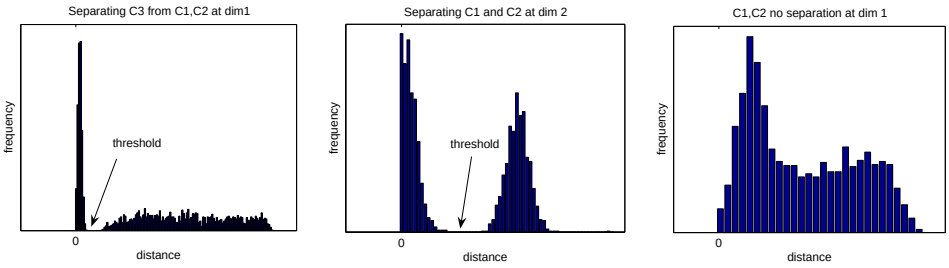
Figure 6. A tree summarizing the clustering process of the sample data set from Fig. 2. The labels on the arrows specify the dimension of the linear manifold which was used to separate the clusters.

to separate the remaining clusters C_1, C_2 using 1D manifolds. Since both these clusters are embedded in 2D linear manifolds the algorithm will fail. However by trying to separate them using 2D manifolds the algorithm will succeed. At this point the algorithm will attempt to further partition each of the clusters C_1 and C_2, however since they are inseparable C_1 and C_2 are declared to be found, and the algorithm terminates.

5 The Line Cluster Algorithm

The line cluster algorithm SLCLUS is a specialization of the general algorithm of the previous section that is aimed at specific linear dependencies of the form $x_j = b_{ij}x_i + c_{ij}$ between pairs of features. Because of this and unlike LMCLUS it is restricted to axis-parallel subspace. Consequently, is based *feature selection* techniques. It also exploits the *downward closure property for lines*: If there exists a line in a set of k dimensions then there exits a line in all $k-1$ subsets of these k dimensions [17].

Downward closure tells us that if a set of points form a line cluster in some set of dimensions it is possible to commence the search for the cluster in a smaller set of dimensions, and iteratively extend it in a bottom-up manner using forward selection. Moreover, the search for clusters in lower dimensions is typically easier (faster) than a search for clusters in higher dimensions. Downward closure also provides pruning power. If a line cluster is not visible in a smaller set of dimensions it is not necessary to search for it in higher dimensions. Using this property we can also devise a termination condition for the



(a) (b) (c)

Figure 7. Histograms used to separate the clusters from Fig. 2. (a) C_3 is separated from C_2 and C_3 by sampling 1D linear manifolds. (b) C_1 is separated from C_2 by sampling 2D linear manifolds. (c) a histogram for which no separation can be found.

algorithm, i.e., if the algorithm is unable to detect any more clusters in a small initial set of dimensions then the algorithm should terminate. Starting from lower dimensions and working up ensures that the line clusters are found in the largest possible subspaces.

The algorithm in its most generic form can be stated as follows: find a line cluster in an initial set of dimensions using a random walk on the feature's lattice. Using forward-feature-selection add dimensions (features) to extend and refine the line cluster until no more dimensions can be added, in which case a line cluster is assumed to be found. Remove the identified line cluster from data set and reapply the first two steps on remaining set of points. The algorithm is designed so that it detects the the largest possible clusters embedded in the largest possible subspaces. The rational is that correlations induced by larger clusters in a larger set of features provide stronger evidence pertaining to the relationship between the objects under consideration, and from a statistical point of view is less likely to occur by chance.

To detect lines in subsets of features (axis-parallel subspaces) SLCLUS uses LMCLUS restricted to 1D linear manifolds (henceforth *line detector*). However, rather than using a goodness of separation threshold to qualify clusters, SLCLUS uses an error tolerance threshold Δ, defined as the the maximum distance of points to a line, we are willing to allow. The tradeoff is that at the expense of detecting clusters which are to some extent less data-driven we get clusters which are easier to interpret, more consistent with the line cluster model, and more consistent with the notion of correlation. In addition, rather than returning the line which is best separated from the remaining set of points,

SLCLUS returns the line having most points, ensuring that the largest possible clusters are detected.

5.1 Selecting an Initial Set of Features by Random Walk

One possibility is to start the clustering process with line clusters of the smallest possible dimensionality, i.e., 2D line clusters. This can achieved by searching through all possible 2D subspaces for a line cluster using the line detector algorithm. However, confirmed by experiments presented in ref. [17], clusters tend to overlap when projected from higher dimensional spaces into lower dimensional spaces. Hence, the projection of several line clusters embedded in higher dimensional spaces into a lower dimensional space may either mask each other or appear as a single cluster. This in turn may "confuse" feature selection used to extend a cluster in the determination of which features are relevant to the cluster. Ideally the clustering process should start with a larger set of initial features, close in number to the dimensionality of the subspace in which some line cluster exists. This will not only improve cluster detection accuracy but also improve efficiency as the extension process will be shorter. To achieve this goal we propose a method that is based on a *random walk* on the *features lattice*.

The basic idea can be stated as follows: starting from the full set of features, randomly remove one feature at a time, after each removal invoke the line detector algorithm to detect line clusters in the subspace defined by the remaining features, and repeat the process until a line cluster is detected, or until no more features are left to be removed. As mentioned, ideally we would like the random walk to stop sooner. More formally, let F be the full set of features, and $\mathcal{L}_F$ be the lattice (poset) defined by $(\mathcal{P}(F), \subseteq)$. If F_1 and F_2 are two elements in $\mathcal{L}_F$ (subsets of F), we say that F_2 is more *general* than F_1 or F_1 more *specific* than F_2 denoted by $F_2 \prec F_1$ if $F_2 \subset F_1$. Let $F' \subseteq F$ be a set of features (subspace) in which some line cluster exists, and F'' be the set of features remaining after each feature removal during the walk. If at a certain point during the random walk $F'' \subseteq F'$, i.e., a subset of features that constitute a higher dimensional line cluster is detected, we can stop the random walk and use F'' as our starting point (initial set of features). Then using forward selection we can extend the line cluster currently residing in the subspace defined by the features of F'' to the higher dimensional subspace in which the cluster exists defined by the features of F'. Due to the downward closure property of lines, once this condition is met it is not necessary to continue the walk (remove features) as any subset of F'' will also contain a line cluster. Thus, the method can be restated as: perform a *random walk* on $\mathcal{L}_F$ starting from F and

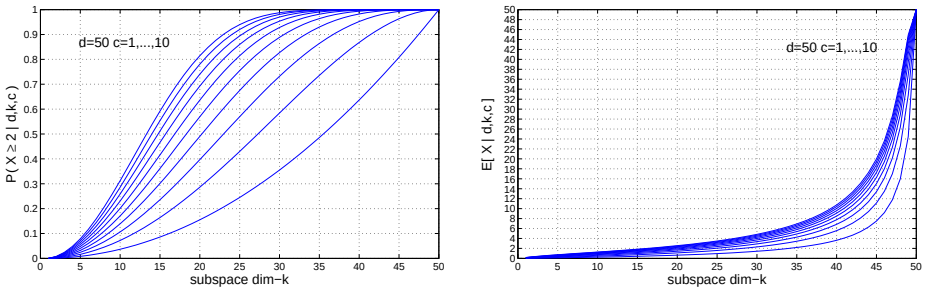
moving in the *generalization* direction until either $F'' \subseteq F'$ or $F'' = \emptyset$.

Fig. 8 shows the change in the probability (the derivation is lengthy and beyond the scope of this paper) of the random walk succeeding (detecting clusters at dimensionality larger than two) and the expected dimensionality of the clusters intercepted by the random walk, as the number of clusters c in the data and the dimensionality of the subspaces k in which the clusters exist are varied. Each curve represents a different number of clusters in the data set, where the lowest curve represents a data set with one cluster and the highest with ten clusters ($c = 1, \ldots, 10$). For illustrative purposes the dimensionality of the data is fixed at $d = 50$, but similar patterns can be observed for other data dimensionalities. It is clear from Fig. 8(a) that as the number of clusters increases and/or the dimensionality of the subspaces in which clusters are embedded is increased the probability of success increases. It is also evident that even when a small number of clusters exist in the data and the clusters are embedded in higher dimensional subspaces there is a high probability of success. Hence, one conclusion that can be drawn is that the random walk is likely to succeed when a large (not necessarily extremely large) number of clusters exist in the data and the clusters are embedded in a relatively higher dimensional subspace. Fig. 8(b) similarly shows that the random walk is more effective, that is, clusters are intercepted sooner and at higher dimensions when the clusters are embedded in higher dimensional subspaces. The more effective region of the random walk seems to be approximately the upper third range of subspace dimensionalities. The figure also shows that the expected value converges to an exponential and that the addition of more clusters beyond a certain point will not enhance the capability of the random walk detecting clusters sooner. One more important conclusion that can be drawn from the figures is that if the random walk fails, it is likely that the clusters are embedded in lower dimensional subspaces, in which case we can revert to the first possibility, which is to initialize the clustering process by searching for 2D line clusters in all possible 2D subspaces.

5.2 The Distance of a Point to a Line

SLCLUS requires the computation of a point's distance (squared) to a line. The distance δ of a point $\mathbf{x}$ modeled by eq. (4) to a k-dimensional line is given by:

$$\begin{aligned} \delta &= \|(I - \boldsymbol{\beta}\boldsymbol{\beta}')(\mathbf{x} - \boldsymbol{\mu})\|^2 = \|(I - \boldsymbol{\beta}\boldsymbol{\beta}')(\boldsymbol{\beta}\boldsymbol{\phi} + \bar{\boldsymbol{\beta}}\bar{\boldsymbol{\epsilon}})\|^2 \\ &= \|\boldsymbol{\beta}\boldsymbol{\phi} - \boldsymbol{\beta}\boldsymbol{\phi} + \bar{\boldsymbol{\beta}}\bar{\boldsymbol{\epsilon}} - 0\|^2 = \|\bar{\boldsymbol{\beta}}\bar{\boldsymbol{\epsilon}}\|^2 = (\bar{\boldsymbol{\beta}}\bar{\boldsymbol{\epsilon}})'\bar{\boldsymbol{\beta}}\bar{\boldsymbol{\epsilon}} \end{aligned}$$



(a) probability of success (b) expected dimensionality

Figure 8. Random Walk Statistics.

$$= \quad \boldsymbol{\epsilon}'\bar{\boldsymbol{\beta}}'\bar{\boldsymbol{\beta}}\boldsymbol{\epsilon} = \boldsymbol{\epsilon}'\boldsymbol{\epsilon} = \sum_{i=1}^{k-1} \epsilon_i^2.$$

According to the line cluster model $\epsilon_i \sim N(0, \sigma^2)$. Therefore the distance δ normalized by σ^2 will have

$$\frac{\delta}{\sigma^2} = \sum_{i=1}^{k-1} \frac{\epsilon_i^2}{\sigma^2} \sim \chi^2_{k-1}, \tag{11}$$

a chi-squared distribution with $k-1$ degrees of freedom. Hence,

$$\mathrm{E}[\delta] = \mathrm{E}[\sigma^2\chi^2_{k-1}] = (k-1)\sigma^2 \quad \text{and} \quad \mathrm{Var}[\delta] = \mathrm{Var}[\sigma^2\chi^2_{k-1}] = 2(k-1)\sigma^4. \tag{12}$$

Because the distance grows with the dimensionality of the subspace in which it measured, and since the search for line clusters will be computed across different dimensionalities, we normalize the distance by its degrees of freedom $(k-1)$ or equivalently the dimensionality of the space orthogonal to the line. This creates a uniform or normalized distance measure which is independent of the dimensionality of the subspace in which it is measured. Therefore the normalized distance $\delta/(k-1)$ has

$$\mathrm{E}\left[\frac{\delta}{k-1}\right] = \sigma^2 \quad \text{and} \quad \mathrm{Var}\left[\frac{\delta}{k-1}\right] = \frac{2\sigma^4}{k-1}. \tag{13}$$

The expected value and variance of the normalized distance will be used as heuristics to set the input parameters to the algorithm.

5.3 The Score (Fit) function

At each forward selection step the quality of the line cluster must be assessed according to some criteria in order to determine whether or not to proceed to the next step. The criteria we use to assess the quality of a cluster is the "fit" of the set of points constituting the cluster to the line in which they are embedded. The fit is defined to be the average-normalized-squared-distance (average error) of the points to the line.

If k is the dimensionality of the subspace in which a cluster is detected, n the number of points constituting the cluster, X the cluster points, and $\mathbf{x}_i$ the i-th point. Then the fit or score function $J(X)$ is defined as

$$J(X) = \frac{1}{n(k-1)} \sum_{i=1}^{n} \left(\|(I - \boldsymbol{\beta}\boldsymbol{\beta}')(\mathbf{x} - \boldsymbol{\mu})\|^2 \right). \tag{14}$$

Prior to the fit computation, $\boldsymbol{\mu}$ and $\boldsymbol{\beta}$ must be estimated. $\boldsymbol{\mu}$ is estimated by computing the sample mean of the cluster. Using least-squares it can be shown that an estimate for $\boldsymbol{\beta}$ is the largest eigenvector of the cluster's covariance matrix, which can be computed using the *power method*.

To guide the algorithm to give certain preferences to the size and dimensionality of the cluster, $J(X)$ can be modified as follows:

$$J'(X) = J(X) n^a (k-1)^b. \tag{15}$$

For example guiding the algorithm to prefer even higher dimensional subspaces we can set b to some value less than zero. Based on eq. (13)

$$\mathrm{E}[J(X)] = \sigma^2 \quad \text{and} \quad \mathrm{Var}[J(X)] = \frac{2\sigma^4}{n(k-1)}, \tag{16}$$

which will also be used as heuristics to set the input parameters.

5.4 Putting it All Together

SLCLUS (Fig. 9) commences by a random walk to select an initial set of features to initiate the clustering process. If the walk failed then the initial set of features is determined by searching through all possible 2D subspaces for the best 2D line cluster. If no such cluster exists (determined by a fit threshold J) then SLCLUS terminates. If either the random walk succeeds or a 2D line cluster is detected, then SLCLUS proceeds by repeatedly calling the *forward selection* procedure (Fig. 10) to extend the cluster into higher dimensions and refine it, until no "improvement" can be made. A cluster is improved if its

```
Algorithm SLCLUS (D, S, Δ, J)
  J(X) = ∞
  X=perform random walk(D)
  if (J(X) > J)
      X=find the best 2D-line cluster(D)
      if (J(X) > J)
          terminate
  while ( |dims(X)| < |dims(D)| )
      X′=ForwardSelection (X, S, Δ)
      if (J(X′) < J(X))
          X = X′
      else
          break
  output X
  D = D − pts(X)
  if (D ≠ ∅)
      goto first step
  return
```

Figure 9. The subspace line clustering algorithm.

fit $J(X)$ decreased. At this point a line cluster is assumed to be found, it is outputted, removed from the data, and the algorithm is reapplied on the remaining set of points until no more points are left to be clustered. The forward selection procedure extends a cluster one dimension at a time, by choosing the dimension whose data when added to an existing cluster, attains the maximum reduction in the fit $J(X)$. We note that by calling the line detector procedure from within the forward selection procedure the algorithm ensures that line clusters are refined by also pealing off unrelated points from them. This refinement is necessary when the projection of several line clusters appear as one line cluster in lower dimensional subspaces. In addition, if the random walk fails it is likely that a cluster is embedded in a lower dimensional subspace, in which case searching through all possible 2D subspaces is likely to reveal it since it is less masked or overlapped by other cluster projections.

SLCLUS requires three input: S, Δ, and J. S is the sampling level parameter used by LMCLUS, which is set in accordance with the heuristic discussed in section 4.2. Δ and J pertain to the error tolerance or deviation from the line we are willing to allow, and indirectly effect the magnitude of correlations the generated clusters will induce. E.g., setting them to higher values will likely generate clusters inducing lower correlations. Assuming we are willing to accept clusters whose average deviation (distance) from a line is σ (i.e, we are assuming that $\boldsymbol{\epsilon} \sim N(\mathbf{0}, \sigma^2 I)$), then using the statistics derived in eq. (13) and eq. (16) as heuristics, we can set Δ and J to their expected value plus a number of their standard deviations.

Algorithm ForwardSelection (X, S, Δ)
  $Y = X$
  while (unexamined dimensions of X remain)
    select an unexamined dimension of X
    add dimension data to X
    X' =**LineDetector**(X, S, Δ)
    if $(J(X') < J(Y))$
      $Y = X'$
    restore X
  return Y

Figure 10. The forward selection subroutine used to gradually extend the subspace in which line clusters are detected and to peel off points which do not belong to the cluster.

6 Empirical validation

The aim of the experiments presented in this section is to demonstrate the applicability and efficacy of the linear manifold clustering paradigm to the problem of correlation clustering. Due to space limitations we omit experiments with synthetic data and refer the reader to ref. [13, 17] for this propose. For the same reason we narrowed the scope of experiments to the problem of pattern clustering and the detection of the simpler form of linear dependencies or correlation. For experiments discussing general linear manifolds and general linear dependencies we refer the reader to ref. [13, 18].

Extensive tests were conducted on three real data set. Two data sets that have become standard benchmarking data sets for clustering gene expression data–the *yeast Saccharomyces Cerevisiae* cell cycle expression data obtained from `http://arep.med.harvard.edu/biclustering/`, and the *Colon Cancer* data obtained from `http://microarray.princeton.edu/oncology/`. The third– *Jester* obtained from `http://ieor.berkeley.edu/~goldberg/jester-data/`, is a data set used in an online joke collaborative filtering/recommender system [21].

6.0.1 Yeast Data

The yeast data contains 2884 genes (objects) and 17 time points/conditions (dimensions), and is a very attractive data set for evaluating clustering algorithms because many of the genes contained in it are biologically characterized and have already been assigned to different phases of the cell cycle. Applied on this data set SLCLUS discovered 62 line clusters. We compared these clusters with the 100 biclusters reported by the biclustering algorithm [5], which were obtained from `http://arep.med.harvard.edu/`

biclustering/. The clusters' size detected by SLCLUS ranged in $[15, 255]$ and on average much smaller than the clusters reported by the biclustering algorithm. From a biological standpoint this makes sense, as the whole yeast genome contains roughly only 6000 genes, and typical functional categories of the yeast genome contain dozens rather than hundreds of genes that were included in some of the biclusters. The dimensionality of the clusters detected by our algorithm ranged in $[3, 16]$ and was on average smaller than the dimensionality of the biclusters. We also compared the *mean squared residue score* (MSRS) [5], which is part of the *shift pattern* cluster model and used by the biclustering algorithm as a criteria to identify shift pattern clusters. The authors of the biclustering algorithm used MSRS=300 as a threshold to qualify "worthy" biclusters. SLCLUS detected on average clusters with a slightly larger MSRS. However, this is reasonable since our algorithm was not restricted to searching only for shift pattern clusters for which this score was designed. Nonetheless, our algorithm was successful in finding clusters that induce large correlations. We used *average correlation*, defined to be the average of the absolute value of the correlation coefficient between each pair of features belonging to a cluster, to quantify the degree of correlation induced by a cluster. After the removal of 3 outlier clusters (clusters with average correlation less than 0.8) the mean average correlation was found to be 0.96. Fig. 11 is a plot of MSRS versus average correlation of the clusters detected by our algorithm. The figure shows that there are gene clusters in the data that induce large correlations and may be functionally related, yet do not follow the shift pattern cluster model (have MSRS>300), supporting one of the main motivations for this work (overlooked patterns). We note that some of the clusters found by our algorithm did follow the shift pattern cluster model.

We also evaluated the biological significance of the clusters SLCLUS produced by means of *function enrichment*–the degree to which the clusters grouped genes of common function. This was done by computing for each cluster P-values (using the hypergeometric distribution) of observing a certain number of genes within a cluster from a particular MIPS (Munich Information Center for Protein Sequences, http://mips.gsf.de/) functional category. Table 1 shows five of them that had smaller P-values, indicating that SLCLUS is able to detect statistically significant clusters.

6.0.2 Cancer Data

The cancer data contains 2000 genes (objects) and 62 tissue samples (dimensions), of which 40 are colon tumor and 22 normal colon samples. The goal in this experiment was to identify gene clusters that can differentiate the

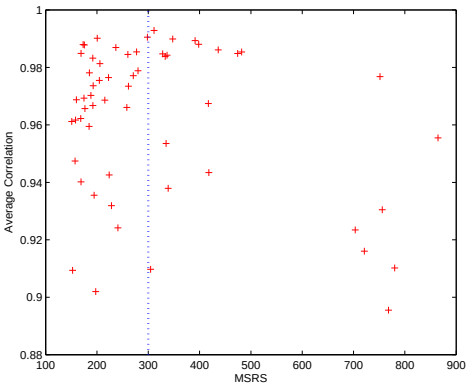


Figure 11. mean squared residue score (MSRS) versus average correlation of yeast clusters detected by SLCLUS.

Table 1. MIPS gene function enrichment.

Genes in Cluster	MIPS Functional Category	Genes in Category	Clustered Genes	P-value
211	ribosome biogenesis	215	27	1.698e-09
	protein synthesis	359	35	5.649e-09
	cytoplasm	554	37	2.696e-05
17	cytoplasm	554	15	1.565e-14
	protein synthesis	359	13	1.178e-13
	ribosome biogenesis	215	11	6.229e-13
	subcellular localisation	2256	16	8.658e-07
193	amino acid biosynthesis	118	13	5.462e-05
49	amino acid metabolism	204	6	6.740e-06
16	cell cycle and DNA processing	628	9	5.772e-06

cancerous tissues from the normal ones. These clusters may later afford researchers the ability design classifiers for diagnostic purposes. Applied on this data set SLCLUS detected 81 line clusters. The size of the clusters was generally small, the largest cluster contained 21 genes. The dimensionality of the subspaces in which the clusters were embedded ranged in $[4, 16]$. The average MSRS of the clusters was 15243, indicating that most of the clusters did not follow the shift pattern cluster model. However, their mean average correlation (after removal of two outlier clusters) on the other hand was around 0.83. Again indicating that related groups of genes may exist in the data, yet are overlooked by most clustering methods.

We also found seven gene clusters that were present in either only the normal tissues or only the cancerous tissues. One cluster contained only normal tissues and the remaining six only cancerous tissues. They contained a small number of genes 16-18, and were embedded in subspaces of dimensionality ranging in $[4, 9]$, i.e., contained between 4-9 tissues. The average correlation of these clusters was around 0.88, higher than the rest of the clusters, providing evidence that the genes within these clusters may be functionally related and can used for discriminatory purposes. Most of the remaining clusters contained a mixture of tissues none with an overwhelmingly majority of normal or cancerous tissues.

6.0.3 Jester Data

The Jester data set contains a million continuous ratings in the range $[-10.00, 10.00]$ of 100 jokes (dimensions) from 73421 users (objects). Not all users rated all jokes, thus the data contains missing values. We extracted a subset of 6916 users that rated all jokes. Applied on this data set SLCLUS identified 252 line clusters with the following statistics. The average size (number of users) of the clusters was 28 and the average dimensionality (number of jokes) of the clusters was 10. Again we examined the MSRS of the clusters to get an idea of what types of patterns were identified. Surprisingly we discovered that most of the clusters followed the shift pattern (slope one clusters in the collaborative filtering literature) with a very low average MSRS of approximately 5. This finding demonstrates that while being able to identify a wider and more general spectrum of patterns SLCLUS does not over look the more common patterns, which as stated are a special case of the line cluster model, when those are present or more evident in the data. After the removal of 29 outlier clusters the average correlation was found to be approximately 0.7, and not as high as in the preceding two experiments.

7 Conclusions

The aim of this paper was to demonstrate the applicability of the linear manifold clustering paradigm to the problem of correlation clustering. We showed that the linear manifold cluster model is a generalization of more common and specific cluster models. This generalization provides a more flexible framework in which cluster analysis may be performed. In the context of correlation we showed that general linear manifolds are able to capture arbitrarily complex linear dependencies that may give rise to correlations. We also showed that 1D linear manifolds as a special case of the general model can capture correlations of a simpler form, where each pair of features are linearly dependent upon each other. Based on these models two stochastic algorithms were presented, the later generalizing the so called "pattern" clustering paradigm. A series of experiments on real data sets was used to demonstrate the efficacy of our methods. One experiment demonstrated that our method is able to identify statistically significant correlation clusters that are overlooked by other existing methods. Until cluster validation techniques appropriate to the correlation or linear manifold clustering paradigms are devised, substantiating clustering results in the absence of ground truth or domain knowledge is hard task. In future work we plan to investigate several statistical approaches to cluster validation which are based on linear manifold likelihood models and on permutation tests.

References

[1] Rakesh Agrawal, Johannes Gehrke, Dimitrios Gunopulos, and Prabhakar Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. In *Proceedings of the international conference on Management of data*, pages 94–105, 1998.

[2] Charu C. Aggarwal, Joel L. Wolf, Philip S. Yu, Cecilia Procopiuc, and Jong Soo Park. Fast algorithms for projected clustering. In *Proceedings of the international conference on Management of data*, pages 61–72, 1999.

[3] Harsha S. Nagesh and Alok Choudhary Sanjay Goil. Mafia: Efficient and scalable subspace clustering for very large data sets. Technical Report 9906-010, Northwestern University, 1999.

[4] Lance Parsons, Ehtesham Haque, and Huan Liu. Subspace clustering for high dimensional data: A review. *Newsletter of the ACM Special Interest Group on Knowledge Discovery and Data Mining*, 6(1):90–105, 2004.

[5] Y. Cheng and G. Church. Biclustering of expression data. In *International Conference on Intelligent Systems for Molecular Biology*, pages 93–103, 2000.

[6] Jiong Yang, Wei Wang, and Haixun Wang Philip Yu. δ-clusters: Capturing subspace correlation in a large data set. In *Proceedings of the 18th International Conference on Data Engineering*, pages 517–528, 2002.

[7] Haixun Wang, Wei Wang, Jiong Yang, and Philip S. Yu. Clustering by pattern similarity in large data sets. In *Proceedings of the International Conference on Management of Data*, pages 394–405, 2002.

[8] Daniel Lemire and Anna Maclachlan. Slope one predictors for online rating-based collaborative filtering. In *Proceedings of SIAM Data Mining*, 2005.

[9] S. Erdal, O. Ozturk, D. Armbruster, H. Ferhatosmanoglu, and W.C. Ray. A time series analysis of microarray data. In *Proceedings of the 4th IEEE Symposium on Bioinformatics and Bioengineering*, pages 366–375, 2004.

[10] Christian Böhm, Karin Kailing, Peer Kröger, and Arthur Zimek. Computing clusters of correlation connected objects. In *Proceedings of the International Conference on Management of Data*, pages 455–466, 2004.

[11] Lizhuang Zhao and Mohammed J. Zaki. Microcluster: Efficient deterministic biclustering of microarray data. *IEEE Intelligent Systems*, 20(6):40–49, 2005.

[12] Jesus S. Aguilar-Ruiz. Shifting and scaling patterns from gene expression data. *Bioinformatics*, 21(20):3840–3845, 2005.

[13] Robert Haralick and Rave Harpaz. Linear manifold clustering in high dimensional spaces by stochastic search. *Pattern Recognition*. doi:10.1016/j.patcog.2007.01.020.

[14] J. B. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297. Berkeley, University of California Press, 1967.

[15] Charu C. Aggarwal and Philip S. Yu. Finding generalized projected clusters in high dimensional spaces. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pages 70–81, 2000.

[16] Rave Harpaz and Robert M. Haralick. Exploiting the geometry of gene expression patterns for unsupervised learning. In *Proceedings of the 18th International Conference on Pattern Recognition*, volume 2, pages 670–674, 2006.

[17] Rave Harpaz and Robert M. Haralick. Mining subspace correlations. In *Proceedings of the 1st IEEE Symposium on Computational Intelligence and Data Mining*, 2007. To appear.

[18] Elke Achtert, Christian Böhm, Hans-Peter Kriegel, Peer Körger, and Arthur Zimek. Deriving quantitative models for correlation clusters. In *Proceedings of the 12th international conference on Knowledge discovery and data mining*, pages 4–13, 2006.

[19] J. Kittler and J. Illingworth. Minimum error thresholding. *Pattern Recognition*, 19(1):41–47, 1986.

[20] Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification, Second Edition*. Wiley, 2000.

[21] Ken Goldberg, Theresa Roeder, Dhruv Gupta, and Chris Perkins. Eigentaste: A constant time collaborative filtering algorithm. *Information Retrieval*, 4(2):133–151, 2001.