Proceedings of the 2007 IEEE Symposium on Computational Intelligence and Data Mining (CIDM 2007)

Mining Subspace Correlations

Rave Harpaz
Pattern Recognition Laboratory
The Graduate Center, City University of New York,
365 Fifth Avenue New York, NY 10016, USA
Email:rbharpaz@sci.brooklyn.cuny.edu

Robert Haralick
Pattern Recognition Laboratory
The Graduate Center, City University of New York,
365 Fifth Avenue New York, NY 10016, USA
Email:haralick@ptah.gc.cuny.edu

Abstract— **In recent applications of clustering such as gene expression microarray analysis, collaborative filtering, and web mining, object similarity is no longer measured by physical distance, but rather by the behavior patterns objects manifest or the magnitude of correlations they induce. Current state of the art algorithms aiming at this type of clustering typically postulate specific cluster models that are able to capture only specific behavior patterns or correlations, and omit the possibility that other information carrying patterns or correlations may coexist in the data. We cast the problem of searching for pattern clusters or clusters that induce large correlations in some subset of features into the problem of searching for groups of points embedded in lines. The advantage of this approach is that is allows the clustering of different patterns or correlations simultaneously. It also allows the clustering of patterns and correlations that are overlooked by existing methods. A formal stochastic line cluster model is presented and its connection to correlation is established. Based on this model an algorithm, which uses feature selection to search for line clusters embedded in subspaces of the data is presented.**

I. Introduction

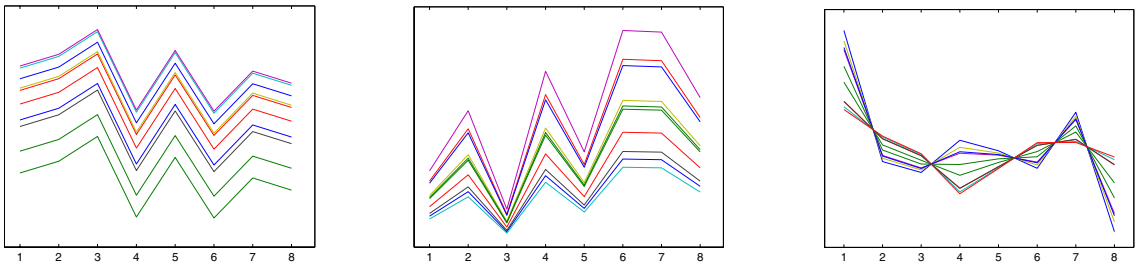
Interest in clustering as a data mining technique has increased substantially in recent years due to new areas of application such as DNA microarray analysis in bioinformatics, document clustering of web pages, image segmentation in computer vision, and recommendation or collaborative filtering systems in E-commerce. Many of these applications are now characterized by high dimensional data. An important advance in this area was the introduction of *subspace clustering* [1], [2], [3], [4] in an attempt to face the new challenges posed by high dimensional data. A subspace cluster consists of a subset of points and a corresponding subset of features (dimensions), such that these points form a dense region in a subspace defined by the set of corresponding features.

Traditional clustering methods including those used in subspace clustering 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. In recent studies these 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), independent of their magnitude, 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.

The most widely studied patterns are the *shift* and *scaling* patterns, 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 second pattern is often reduced to the first by various transformations (e.g. log transform), none of which is pattern preserving when more than one type of pattern coexist in the same data set [9]. 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 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.

In recent studies it has been suggested other types of information carrying patterns such as patterns inducing negative correlations are completely overlooked by most clustering methods, and that current state of the art algorithms are not flexible enough to mine different patterns simultaneously [10], [11], [12]. 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 exit in the data. This in turn typically leads to a large bias in the results.

In a recent work [13] we showed that different types of pattern clusters including those overlooked by most methods,



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

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

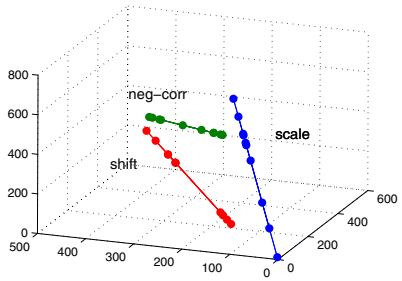


Fig. 2. The geometry of three different types of patterns clusters (shift, scaling, and negative correlations).

and the correlations they induce, can all be generalized to linear manifolds [1] of which a line (a 1-dimensional linear manifold) is a special case. The main insight provided by the study is that within the set of relevant features to a pattern cluster, the points form a line and in the full space a linear manifold of higher dimension. Hence by searching for lines or linear manifolds rather than speci□c patterns or the correlations they induce it is possible to cluster simultaneously all the different types of pattern clusters. We also found that the geometrical difference between the different pattern clusters is the orientation (e.g. a line cluster of slope one represents a shift pattern) and translation of the cluster in the space. Fig. 2 shows the geometry of three different types of pattern clusters each including 10 points embedded in a 3-dimensional space. Notice how they all form lines.

In this paper we focus our attention on line clusters - de□ned as groups of points that □t a line in some subspace of the data. We start by presenting a formal stochastic "line cluster model" and establish its connection to correlations. We then present an algorithm for detecting line clusters. The algorithm uses a *line detector* procedure which searches for line clusters in some subset of features, and a *forward-feature-selection* approach to re□ne the cluster. We conclude the paper with experiments demonstrating the potential of our clustering method.

[1] A linear manifold is a translated subspace. A subspace is a subset of points closed under linear combination.

II. The Line Cluster Model

Suppose a line cluster X exists in a k-dimensional subspace.[2] Let $\mathbf{x}$ be a $k \times 1$ vector representing some point in X, β be a unit norm $k \times 1$ vector that spans a $1D$ subspace, $\overline{\beta}$ be 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 β.

Definition 1 (The Line Cluster Model): Let μ be some point in $\mathbb{R}^k$, ϕ be a zero mean random scalar distributed according to $U(-R/2, +R/2)$ where R is the range of the data, and ϵ be a $k-1 \times 1$ random vector distributed according to $N(\mathbf{0}, \sigma^2 I)$, where $\sigma \ll R$. Then each $\mathbf{x} \in X$, a *line cluster* is modeled by,

$$\mathbf{x} = \mu + \beta\phi + \overline{\beta}\epsilon. \quad (1)$$

The idea is that each point in a cluster lies close to a line (1-dimensional linear manifold) of □nite extent, which is de□ned by μ, a translation vector, the space spanned by the vector β, and the range parameter R. Since

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

the cluster mean is μ. On the line the points are assumed to be uniformly distributed in direction β according to $U(-R/2, +R/2)$, where ϕ can be viewed as the displacement or distance of a point from the line's center. The assumption of uniformity is not binding, and can be replaced by any other distribution with symmetric support. What characterizes this type of cluster is the third component that models a small random error associated with each point on the line. The idea is that each point may be perturbed in directions that are orthogonal to the subspace spanned by β, that is the subspace spanned by the $k-1$ columns of $\overline{\beta}$. We model this behavior by requiring that ϵ be a $(k-1) \times 1$ random vector, normally distributed according to $N(\mathbf{0}, \sigma^2 I)$, where σ is much smaller than R. The error, ϵ, can be thought of as the displacement or distance of a point to its projection onto the line. The addition of the error term essentially transforms the line into a thin elongated cylinder.

[2] The term subspace in the context of clustering is often misused to indicate a subset of the original measurement features which is a special case of a subspace. We will follow the trend and leave it to the reader to distinguish between the two throughout the paper.

A. Lines and Correlations

Fundamental to our method is the connection between line clusters and correlation, established by the following proposition.

Proposition 1: A set of points induce perfect correlations among a set of features if and only if the set of points perfectly fit a line in these set of features.

Proof: Following the model given in (1) a set of points will perfectly fit a line if they do not include an error term which translates them away from the line. Formally, a perfect fit implies that each point can be modeled by $\mathbf{x} = \mu + \beta\phi$. By perfect correlations we mean that the correlation coefficient ρ_{ij} between any pair of features i and j or equivalently the random components x_i and x_j of the random vector $\mathbf{x}$, is equal to 1 or -1. The proposition can now be stated formally as

$$\mathbf{x} = \mu + \beta\phi \Leftrightarrow \forall i, j \ \rho_{ij} = \pm 1.$$

($\Rightarrow$): Given $\mathbf{x} = \mu + \beta\phi$, each component or feature of $\mathbf{x}$ denoted by x_i is equal to $\mu_i + \phi\beta_i$ where μ_i and β_i are the i-th components of vectors μ and β. Hence,

$$\text{Var}[x_i] = \text{Var}[\mu_i + \phi\beta_i] = \beta_i^2 \text{Var}[\phi],$$

$$\begin{aligned}
\text{Cov}(x_i, x_j) &= \text{E}[x_i x_j] - \text{E}[x_i]\text{E}[x_j] \\
&= \text{E}[(\mu_i + \phi\beta_i)(\mu_j + \phi\beta_j)] - \mu_i\mu_j \\
&= \text{E}[\mu_i\mu_j + \mu_i\phi\beta_j + \mu_j\phi\beta_i + \phi^2\beta_i\beta_j] - \mu_i\mu_j \\
&= \mu_i\mu_j + 0 + 0 + \beta_i\beta_j E[\phi^2] - \mu_i\mu_j \\
&= \beta_i\beta_j \text{E}[\phi^2] \\
&= \beta_i\beta_j \text{Var}[\phi],
\end{aligned}$$

and therefore

$$\begin{aligned}
\rho_{ij} &= \frac{\text{Cov}(x_i, x_j)}{\sqrt{\text{Var}[x_i]}\sqrt{\text{Var}[x_j]}} = \frac{\beta_i\beta_j \text{Var}[\phi]}{\sqrt{\beta_i^2 \text{Var}[\phi]}\sqrt{\beta_j^2 \text{Var}[\phi]}} \\
&= \frac{\beta_i\beta_j \text{Var}[\phi]}{\sqrt{\beta_i^2\beta_j^2}\text{Var}[\phi]} = \frac{\beta_i\beta_j}{|\beta_i\beta_j|} = \pm 1
\end{aligned}$$

($\Leftarrow$): Given $\forall i, j \ \rho_{ij} = \pm 1$. Assume $\text{Var}[x_i] = \alpha_i^2$ and $\text{Var}[x_j] = \alpha_j^2$, and let $\frac{x_i}{\alpha_i} - \frac{x_j}{\alpha_j}$ be a new random variable (r.v.). Then,

$$\begin{aligned}
\text{Var}\left[\frac{x_i}{\alpha_i} - \frac{x_j}{\alpha_j}\right] &= \text{Var}\left[\frac{x_i}{\alpha_i}\right] + \text{Var}\left[\frac{x_j}{\alpha_j}\right] - 2\text{Cov}\left(\frac{x_i}{\alpha_i}, \frac{x_j}{\alpha_j}\right) \\
&= \frac{\text{Var}[x_i]}{\alpha_i^2} + \frac{\text{Var}[x_j]}{\alpha_j^2} - 2\frac{\text{Cov}(x_i, x_j)}{\alpha_i\alpha_j} \\
&= 1 + 1 - 2\rho_{ij} \ = \ 2(1 - \rho_{ij}).
\end{aligned}$$

$\rho_{ij} = 1$ implies that $\text{Var}\left[\frac{x_i}{\alpha_i} - \frac{x_j}{\alpha_j}\right] = 0$ which in turn implies that the r.v. $\frac{x_i}{\alpha_i} - \frac{x_j}{\alpha_j} = c$, i.e. equals a constant, from which we can then establish a linear relationship between x_i and x_j of the form $x_j = bx_i + c$. Similarly, $\rho_{ij} = -1$ implies that $\text{Var}\left[\frac{x_i}{\alpha_i} + \frac{x_j}{\alpha_j}\right] = 0$, which again establishes a linear relationship between x_i and x_j of the form $x_j = bx_i + c$. By introducing $\beta_i, \beta_j, \mu_i, \mu_j$, the r.v. ϕ, and letting $b = \beta_j/\beta_i$ and $c = -\beta_j\mu_i/\beta_i + \mu_j$, and then substituting them into $x_j = bx_i + c$ we get

$$\frac{x_i - \mu_i}{\beta_i} = \frac{x_j - \mu_j}{\beta_j}.$$

Since both sides of the equation define a r.v. we may choose to call this r.v. ϕ, yielding that

$$x_i = \mu_i + \beta_i\phi \quad \text{and} \quad x_j = \mu_j + \beta_j\phi.$$

Now collecting all the equations of the form above, and putting then in vector format gives the final result that $\mathbf{x} = \mu + \beta\phi$.

Although not done in this paper, using elements of the previous proof it can be shown that the more a set of points deviates from a predefined line of the form $\mathbf{x} = \mu + \beta\phi$, the less correlated the features corresponding to the points will be.

III. The Algorithm

Supported by proposition 1 the problem of searching for groups of points which induce large correlations in subspaces (subsets of features/dimensions) of the data is cast into the problem of searching for line clusters embedded in subspaces of the data. In addition we require that the algorithm detects 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.

The algorithm in its most generic form can be stated as follows: find the "best" line cluster in an initial set of dimensions, using "Feature Selection" add or remove dimensions to refine the line cluster, remove the refined line cluster from data set and reapply the first two steps on remaining set of points. The two main components of the algorithm are; a *line detector* procedure which searches for line clusters in some subset of features, and a *feature selection* procedure which refines the clusters (based on some criteria) by either adding or removing features.

A. Line Detectors

Four line detectors were evaluated amongst which the most accurate and efficient was chosen; a variation of RANSAC [14], two versions of LMCLUS [15], and an adaptation of K-means [16] to line clustering, all which are stochastic in nature (based on sampling). The *Hough transform* [17] which may also be used as a line detector was omitted from the evaluation due to its relative inefficiency in higher dimensional spaces. We found LMCLUS and RANSAC to be the most accurate, and among the two RANSAC to be more efficient. Based on this conclusion in addition to RANSAC's simplicity and relatively intuitive input parameters, we chose the modified version of RANSAC as our line detector. The final version of the line detector algorithm is outlined in Fig. 3. The last step (return statement) ensures that the largest line clusters are detected by the overall line clustering algorithm.

Line Detector (X, s, d, t)
  repeat s times
    sample two points x_1, x_2 from X
    determine line parameters $\mu = x_1,\ \beta = (x_2 - x_1)/\|(x_2 - x_1)\|$
    determine the number of points in the data that are within distance d ("inliers") from the line
  return the line with most inliers if it includes at least t inliers

Fig. 3. A RANSAC based line detector algorithm.

B. Feature Selection

Feature selection techniques attempt to discover the most revelent attributes of the data as part of a machine learning or model building process. In our case feature selection is used to select the largest possible set of dimensions in which a set of points fits a line. Finding the the optimal set of features by an exhaustive search trough all possible sets of features is typically infeasible. For this reason most feature selection techniques employ a greedy hill-climbing approach. The basic procedure involves identifying an initial model, and iteratively improving the model by adding or removing features in accordance with some criteria until there is no improvement or when a predetermined number of steps has been reached. *Forward selection* techniques are bottom-up methods which start with an empty or small set of features and at each step add the most relevant feature to the model. *Backward elimination* techniques are top-down approaches which start with the full set of features and remove the most irrelevant feature at each step. *Stepwise* methods employ a combination of two, and depending on the direction of search are called *forward stepwise* and *backward stepwise*.

Motivated by the "Downward Closure Property of Lines" stated in the following proposition, the forward selection approach was chosen to be employed in the algorithm.

Proposition 2: 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.

Proof: Based on the model given in (1) and similar to the proof of proposition 1 each of the k components x_i of each point constituting a line can be modeled by $x_i = \mu_i + \beta_i\phi$ where x_i and ϕ are random variables. We can therefore select any subset of $k-1$ components where each of the components is modeled as above, and join them together into a vector producing $\mathbf{x}' = \mu' + \beta'\phi$, the model of a $k-1$-dimensional line.

Proposition 2 tell 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, supporting the bottom-up feature selection approach we chose to employ. Moreover, the search for clusters in lower dimensions is typically easier (faster) than a search for clusters in higher dimension. Proposition 2 also provides the algorithm with pruning power. That is, 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 algorithm, i.e. if the line detector is not able to detect any more clusters in a small initial set of dimensions then the algorithm should terminate. The combination of the bottom-up approach and proposition 2 also ensures that the line clusters are found in the largest possible subspaces.

C. The Distance of a Point to a Line

The line detector algorithm requires the computation of a point's distance to a line. The squared distance (henceforth distance) of a point to a subspace is the norm squared of its projection to the orthogonal complement subspace. Formally, the distance δ of a point $\mathbf{x}$ modeled by (1) to a k-dimensional line is given by:

$$\begin{aligned}\delta &= \|(I - \beta\beta')(\mathbf{x} - \mu)\|^2 = \|(I - \beta\beta')(\beta\phi + \bar{\beta}\epsilon)\|^2 \\ &= \|\beta\phi - \beta\phi + \bar{\beta}\epsilon - 0\|^2 = \|\bar{\beta}\epsilon\|^2 = (\bar{\beta}\epsilon)'\bar{\beta}\epsilon \\ &= \epsilon'\bar{\beta}'\bar{\beta}\epsilon = \epsilon'\epsilon = \sum_{i=1}^{k-1}\epsilon_i^2.\end{aligned}$$

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

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

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

$$\mathrm{E}[\delta] = \mathrm{E}[\sigma^2\chi^2_{k-1}] = (k-1)\sigma^2,$$

and

$$\mathrm{Var}[\delta] = \mathrm{Var}[\sigma^2\chi^2_{k-1}] = 2(k-1)\sigma^4.$$

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 to. 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

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

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

Lemma 1: $\|(I-\beta\beta')(\mathbf{x}-\mu)\|^2 = ||\mathbf{x}-\mu||^2 - ||\beta'(\mathbf{x}-\mu)||^2$.

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

$$\begin{aligned}||(I - \beta\beta')\mathbf{y}||^2 &= ||\mathbf{z} - \beta\beta'\mathbf{y}||^2 \\ &= (\mathbf{y} - \beta\beta'\mathbf{y})'(\mathbf{y} - \beta\beta'\mathbf{y}) \\ &= \mathbf{y}'\mathbf{y} - 2\mathbf{y}'\beta\beta'\mathbf{y} - \mathbf{y}'(\beta\beta')^2\mathbf{y} \\ &= \mathbf{y}'\mathbf{y} - \mathbf{y}'\beta\beta'\mathbf{y} \\ &= ||\mathbf{y}||^2 - ||\beta'\mathbf{y}||^2.\end{aligned}$$

Lemma 1 provides us a much more efficient way of computing the distance. If k is the dimensionality of the subspace in which the distance is computed, then computing it using lemma 1 gives us a speedup of $O(k)$, which for high dimensional spaces becomes a significant factor.

D. The Score (Fit) function

At each forward selection step the quality of the line cluster returned by the line detector must be assessed according to some criteria in order to determine whether or not to proceed to the next step. The criteria used in this paper 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 (error) of the points to the line.

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

$$J(X) = \frac{1}{n(k-1)} \sum_{i=1}^{n} \left(||\mathbf{x}_i - \mu||^2 - ||\beta'(\mathbf{x}_i - \mu)||^2 \right). \quad (3)$$

Prior to the fit computation, μ and β must be estimated. μ is estimated by computing the sample mean of the cluster. It can be shown (using least-squares) that an estimate for β 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:

$$J'(X) = J(X)n^a(k-1)^b. \quad (4)$$

For example guiding the algorithm to prefer even higher dimensional subspaces we can set b to some value less than zero.

Based on (2)

$$\mathrm{E}[J(X)] = \sigma^2 \quad \text{and} \quad \mathrm{Var}[J(X)] = \frac{2\sigma^4}{n(k-1)}, \quad (5)$$

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

The overall and finalized version of the subspace line clustering algorithm is outlined in figures 4 and 5. We note that by calling the line detector procedure from within the forward selection procedure the algorithm ensures that the line clusters are refined by also pealing off points from them. This refinement is necessary when the projection of several line clusters appear as one line cluster in lower dimensional subspaces.

IV. Setting the Input Parameters

The algorithm requires the input of four parameters; s, a sample size parameter used to specify the maximum number of attempts that should be made by the line detector to identify a line cluster; d, the maximum distance of a point to a line allowed for it to be considered an "inlier", i.e., d can considered an error tolerance; t, a threshold used to specify that a large enough cluster has been detected; J, a "fit" threshold used to determine whether the algorithm should terminate, i.e., that no more clusters with a sufficient enough fit are left in the data.

Subspace Line Clustering (D, s, d, t, J)
  X=find the best 2D-line cluster(D)
  if $(J(X) > J)$
    terminate
  while (dim(X)<dim(D))
    X'=**Forward Selection** (X, s, d, t)
    if $(J(X') < J(X))$
      $X = X'$
    else
      break
  output X
  $D = D - X$, goto the first step

Fig. 4. The subspace line clustering algorithm. Finding the best 2D-line cluster can be done by examining all possible 2D subspaces and returning the line cluster with the best fit.

Forward Selection (X, s, d, t)
  while (unexamined dimensions remain)
    select an unexamined dimension
    add dimension data to X
    X' =**Line Detector**(X, s, d, t)
    if $(J(X') < J(X))$
      $X = X'$
    else
      restore X
  return X

Fig. 5. 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.

t is relatively intuitive but may require some domain knowledge about the number of points we should expect to see included within a cluster. For example, in gene expression clustering typical functional categories contain a small amount of genes, and therefore t should be set to a low value.

d and J are also relatively intuitive and easy to set. They 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. I.e., setting them to higher values will likely generate clusters inducing lower correlations. Assuming we are willing to accept clusters whose average deviation (normalized distance) from a line is σ then using to (2) and (5) as heuristics we can set d and J to the expected value plus a number of standard deviations of the normalized distance δ and the fit function $J(X)$, that is,

$$d = \sigma^2 + c\sigma^2 \sqrt{\frac{2}{(k-1)}}, \quad (6)$$

and

$$J = \sigma^2 + c\sigma^2 \sqrt{\frac{2}{n(k-1)}}, \quad (7)$$

TABLE I
VALUES FOR INPUT PARAMETER s

K	p	s given $\epsilon = 0.05$	s given $\epsilon = 0.01$
2	0.2500	10	16
4	0.0625	46	71
6	0.0278	106	163
8	0.0156	190	292
10	0.0100	298	458
12	0.0069	430	661
14	0.0051	586	900
16	0.0039	765	1177
18	0.0031	969	1490
20	0.0025	1197	1840

where c is the number of standard deviations. Since these values depend on k-the dimensionality of the subspace in which a cluster is either searched for or reported, the two parameters must be adjusted dynamically by the algorithm depending on k. Hence, the two parameters should simply be set to $d = J = \sigma$.

s should be selected large enough to ensure with high probability that at least one of the samples of two points are within error tolerance to a line, i.e., come from the same line cluster. This in turn will ensure that the sampled points can be used to approximate the line in which a possible cluster is embedded and collet its remaining points. Let p be the probability that two sampled points come from the same cluster, X be a geometric random variable denoting the number of trails (samples) needed to get one success (two points coming from the same cluster), and $F(X)$ its cumulative distribution function. If we want to ensure with probability $1 - \epsilon$ $(0 < \epsilon < 1)$ a success then the number of trails s needed should satisfy

$$F(X) = P(X \leq s) = 1 - (1 - p)^s \geq 1 - \epsilon,$$

yielding that s should be set to

$$s \geq \frac{\log \epsilon}{\log(1 - p)}. \tag{8}$$

It now remains to determine p, the probability that two sampled points come from the same cluster. One way is to assume that there are no more than K clusters of approximately the same size in the data set, and set $p = 1/K^2$. Another way is use t and set $p = (|D|/t)^2$, where $|D|$ is the number of points being clustered. Table I uses $p = 1/K^2$ to show some values of s, for corresponding values of K and ϵ.

V. A NOTE ON ALGORITHMIC COMPLEXITY

Because of the algorithm's stochastic nature a formal complexity analysis of its running time is not straightforward, and as a consequence omitted from the paper. We would like to note however that its flexibility comes at a price. Its main bottleneck is associated with the computation of the leading eigenvector of a cluster's covariance matrix, which is used to estimate the line parameter β and essentially fit the data to a line. One possible solution is not to fit the line and use the two original sampled points that are used by the line detector as a less accurate estimate of the line parameter β. That is, if $\mathbf{y}_1$ and $\mathbf{y}_2$ are the two sampled points that correspond to the line cluster returned by the line detector procedure, then β can be estimated by $\beta = (\mathbf{y}_1 - \mathbf{y}_2)/\|\mathbf{y}_1 - \mathbf{y}_2\|$. Nonetheless several more flexible algorithms aiming at similar problems, such as [11], [18] execute some form of an eigendecomposition to estimate eigenvectors. These algorithms are even more expensive than ours as they need to compute several eigenvectors and not only one. We plan however to investigate the above approach and other optimizations in future work.

VI. EMPIRICAL VALIDATION

We experimented with the subspace line clustering algorithm by applying it on both real and synthetic data sets. The algorithm was implemented in a Linux based Matlab environment.

A. Synthetic data

To better understand the algorithm, several dozen data sets were generated according to the line cluster model specified in (1). The aim of the experiment with the synthetic data was to evaluate the algorithm's accuracy in the detection of both the clusters that were generated and the subspaces in which they were embedded. To determine the algorithm's accuracy in cluster detection we measured the degree of correspondence between the point class labels of each output cluster with its mapped input cluster (input cluster with the largest number of points in common), weighted by the size of the output cluster. The determination of the algorithm's accuracy in subspace detection was measured by the degree of correspondence between the subspace feature labels of the output and input clusters. Data sets were generated with between 5-50 dimensions, and for each selected dimensionality three types of data sets were created. One with a small number of clusters (4-6 clusters), one with a small number of clusters to which noise points were added representing approximately 30% of the data. And a third type of data set containing a larger number of clusters (8-11). We also ensured that the dimensionality of the subspaces in which clusters were embedded ranged across the dimensionality of the data.

In addition to the algorithm presented in this paper, another version that used a forward-stepwise feature selection approach was implemented. We believed that the forward-stepwise approach would result in better accuracy, but discovered that on average not many backward elimination (feature removal) steps were executed, and better accuracy was not achieved. Due to the overhead associated with the backward steps we preferred the algorithm presented in this paper.

From a pool of dozens of synthetic data sets on which the algorithm was applied, a representative sample of seven of each of the three different types (21 in total) of data sets was selected for illustrative purposes. The performance (accuracy) of the algorithm applied on these data sets is summarized in table II. The accuracy measures range in $[0, 1]$, where larger values indicate better accuracy. Due to the stochastic nature of the algorithm it was run between 50-100 times on each

TABLE II

A SUMMARY OF THE ALGORITHM'S PERFORMANCE IN TERMS ACCURACY APPLIED ON SYNTHETIC DATA SETS

	small		small+noise		large	
data dim	pts	dim	pts	dim	pts	dim
5	0.96	0.99	0.89	0.97	0.92	0.94
8	0.99	1.0	0.94	0.99	0.98	0.99
10	0.99	0.98	0.97	0.98	0.93	0.94
15	0.97	0.93	0.96	0.93	0.97	0.91
20	0.99	0.79	0.90	0.90	0.96	0.95
30	0.94	0.96	0.82	0.95	0.96	0.91
50	0.97	0.65	0.93	0.92	0.90	0.67

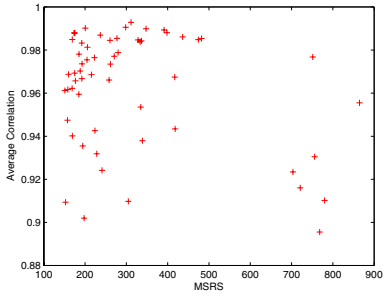


Fig. 6. mean squared residue score (MSRS) versus average correlation of yeast clusters detected by the algorithm.

data set and the average accuracy was recorded. The table shows that the algorithm is able to maintain high levels of accuracy in cluster point detection (denoted by 'pts' in the table) and good over overall performance in the detection of the subspaces which the clusters were embedded (denoted by 'dim' in the table). However, the table shows that as the dimension of the data sets increases, the accuracy in subspace detection deteriorates. We attribute this behavior to the possibility that the projection of several clusters embedded in high dimensional spaces into lower dimensional spaces appear as single clusters when the algorithm commences the search, and "confuses" it in the determination of which features are relevant to each cluster. We speculate that a top-down backward elimination approach might be able to rectify the problem, and look forward to experimenting with this approach. We note however, that a top-down approach will not be able to exploit the advantages of the downward closure property of lines.

B. Real Data

We conducted extensive tests on two typical data sets that have become standard benchmarking data sets for clustering gene expression data–the yeast *Saccharomyces Cerevisiae* cell cycle expression data [3], and the *Colon Cancer* data [4]. The yeast data contained 2884 genes and 17 conditions, and the cancer data contained 2000 genes and 62 tissue samples, of which 40 were colon tumor and 22 normal colon samples.

Applied on the yeast data the algorithm discovered 62 line clusters. We compared these clusters with the 100 biclusters reported by the biclustering algorithm [3] [5]. The clusters' size detected by our algorithm 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) 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. Our algorithm detected on average clusters with a slightly larger MSRS. However, this 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. 6 is a plot of MSRS versus average correlation of the clusters detected by our algorithm. The figure essentially shows that there are gene clusters in the data that induce large correlations and may be functionally related, yet may not follow the shift pattern cluster model, supporting the main motivation for this work. 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 our algorithm produced by means of *function enrichment* [19]– 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 [5] functional category. Some of the clusters demonstrated significant grouping (very small P-values) of genes within the same functional class. Table III shows five of them that had larger P-values.

Applied on the cancer data our method detected 82 line clusters. The goal in this experiment was to identify gene clusters that can differentiate the cancerous tissues from the normal ones. These clusters may later afford researchers the ability design classifiers for diagnostic purposes. The size of the clusters detected by our algorithm was generally small, the largest cluster contained 24 genes. The dimensionality of the subspaces in which the clusters were embedded ranged in

[3] obtained from http://arep.med.harvard.edu/biclustering/

[4] obtained from http://microarray.princeton.edu/oncology/

[5] Munich Information Center for Protein Sequences, http://mips.gsf.de/

TABLE III

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

[4, 20]. The average MSRS of the clusters was 2343, indicating that most of the clusters did not follow the shift pattern cluster model. However, their mean average correlation (after removal of outlier clusters) on the other hand was around 0.8. 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. They contained a small number of genes (around 16) and were embedded in lower dimensional subspaces, i.e., were present in a small number of tissues. The average correlation of these clusters was around 0.93, higher than the rest of the clusters, providing strong evidence that the genes within these clusters may be functionally related and used for discriminatory purposes. Most of the remaining clusters contained a mixture of tissues none with an overwhelmingly majority of normal or cancerous tissues.

VII. CONCLUSION

The problem of searching for so called "pattern clusters" or clusters that induce large correlations in some subset of features was cast into the problem of searching for groups of points embedded in lines. The advantage of this paradigm of clustering is that is allows the clustering of different patterns or correlations simultaneously. It also allows the clustering of patterns and correlations that are overlooked by existing methods. A formal stochastic line cluster model was presented, and based on it an algorithm that searches for line clusters embedded in subspace of the data was presented. The algorithm was evaluated on real and synthetic data to demonstrate its potential. The algorithm uses a forward feature selection approach, and we postulate that a backward feature selection may be able to perform better on data sets whose clusters are embedded in higher dimensional subspace. We look forward to investigate this approach in future work. A formal complexity analysis of the algorithm's running time was not presented due to its stochastic nature, but its flexibility comes at a price. The main bottle neck in terms of efficiency is due to the computation of the leading eigenvector of a cluster's covariance matrix, and in future work we also plan to investigate possible optimizations.

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 ACM SIGMOD 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 1999 ACM SIGMOD international conference on Management of data*, pages 61–72. ACM Press, 1999.

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

[4] Lance Parsons, Ehtesham Haque, and Huan Liu. Subspace clustering for high dimensional data: A review. *SIGKDD Explorations, 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] Yang et al. δ-clusters: Capturing subspace correlation in a large data set. In *ICDE*, pages 517–528, 2002.

[7] Wang et al. A fast algorithm for subspace clustering by pattern similarity. In *SSDBM*, 2004.

[8] Daniel Lemire and Anna Maclachlan. Slope one predictors for online rating-based collaborative filtering. In *Proceedings of SIAM Data Mining (SDM'05)*, 2005.

[9] Jess S. Aguilar-Ruiz. Shifting and scaling patterns from gene expression data. *Bioinformatics*, 21(10):3840–3845, 2005.

[10] Erdal et al. A time series analysis of microarray data. In *BIBE*, page 366, 2004.

[11] Böhm et al. Computing clusters of correlation connected objects. In *ACM SIGMOD*, pages 455–466, 2004.

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

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

[14] Martin A. Fischler and Robert C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[15] R. Haralick and R. Harpaz. Linear manifold clustering. In *4th International Conference on Machine Learning and Data Mining in Pattern Recognition (MLDM 2005), Springer Verlag, LNAI 3587*, pages 132–141, 2005.

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

[17] J. Illingworth and J. Kittler. A survey of the hough transform. *Comput. Vision Graph. Image Process.*, 44(1):87–116, 1988.

[18] 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.

[19] Tavazoie et al. Systematic determination of genetic network architecture. *Nature Genetics*, 22(3):281 – 285, 1999.