# Coclustering of Human Cancer Microarrays Using Minimum Sum-Squared Residue Coclustering

Hyuk Cho and Inderjit S. Dhillon

**Abstract**—It is a consensus in microarray analysis that identifying potential local patterns, characterized by coherent groups of genes and conditions, may shed light on the discovery of previously undetectable biological cellular processes of genes, as well as macroscopic phenotypes of related samples. In order to *simultaneously* cluster genes and conditions, we have previously developed a fast coclustering algorithm, *Minimum Sum-Squared Residue Coclustering (MSSRCC)*, which employs an alternating minimization scheme and generates what we call *coclusters* in a "checkerboard" structure. In this paper, we propose specific strategies that enable MSSRCC to escape poor local minima and resolve the degeneracy problem in partitional clustering algorithms. The strategies include binormalization, deterministic spectral initialization, and incremental local search. We assess the effects of various strategies on both synthetic gene expression data sets and real human cancer microarrays and provide empirical evidence that MSSRCC with the proposed strategies performs better than existing coclustering and clustering algorithms. In particular, the combination of all the three strategies leads to the best performance. Furthermore, we illustrate coherence of the resulting coclusters in a checkerboard structure, where genes in a cocluster manifest the phenotype structure of corresponding specific samples and evaluate the enrichment of functional annotations in Gene Ontology (GO).

**Index Terms**—Microarray analysis, coclustering, binormalization, deterministic spectral initialization, local search, gene ontology.

## 1 INTRODUCTION

Microarray technologies enable researchers to simultaneously measure the expression levels of thousands of genes in a massively parallel manner [1]. The measurements of a microarray experiment are usually summarized in the form of a data matrix, called *gene expression data matrix*, whose rows represent genes and columns represent various specific experimental conditions such as different samples, different time points, or different organisms. Each entry of the matrix corresponds to a numeric representation of the expression or activity of a particular gene under a given experimental condition. Microarray techniques have been extensively applied to diverse areas, where the main purpose is to detect differential expression levels, discover patterns, or predict classes of experimental conditions [2], [3]. In addition to the enormous scientific potential of microarrays in making biological inferences, microarrays have been employed to investigate global gene expression in human disease research, biomedical, pharmaceutical, and clinical studies. In particular, human cancer research has become one of the major applications of microarrays and accounts for more than 80 percent of the microarray publication on human disease to date [4]. Microarray technology in cancer research has many potential uses such as major insights into the genesis, progression, prognosis, and response to therapy on the basis of gene expression profiles [2], [5].

Various cluster analysis techniques have been applied to microarrays and have been proven to be useful for identifying biologically relevant groupings of genes and samples [6], [7]. For example, Eisen et al. [8] used clustering to identify groups of coregulated genes similarly expressed across all samples, expecting clustered genes to participate in similar cellular machinery or network, while Golub et al. [9] and Alizadeh et al. [10] applied clustering to cluster samples into homogeneous groups according to gene expression profiles, expecting them to correspond to particular macroscopic phenotypes. However, both approaches may fail to identify local patterns where subsets of genes are coregulated and coexpressed only under certain experimental conditions, and therefore, the discovery of such local expression patterns may be the key to uncovering many genetic pathways that are not apparent otherwise [11]. In contrast, coclustering, or the *simultaneous* clustering of genes and conditions, aims at finding such local coherent patterns that better reflect biological reality.

Recently, Cho et al. [12] developed two residue-based coclustering algorithms that simultaneously identify coclusters with coherent values in both rows and columns via an alternating $k$-means-like iterative algorithm, resulting in a checkerboard structure. The first algorithm is based on the partitioning model proposed by Hartigan [13] and captures coclusters with constant values. The other, which we refer to as Minimum Sum-Squared Residue Coclustering (MSSRCC) algorithm, is based on the squared residue formulated by Cheng and Church [14] and discovers coclusters with coherent "trends," as well as constant values. Therefore, in this paper, we intend to provide specific strategies that substantially enhance MSSRCC and

• *The authors are with the Department of Computer Science, The University of Texas at Austin, 1 University Station C0500, Taylor Hall 2.124, Austin, TX 78712. E-mail: {hyukcho, inderjit}@cs.utexas.edu.*

*Manuscript received 29 Jan. 2007; revised 5 Sept. 2007; accepted 21 Sept. 2007; published online 21 Nov. 2007.*
*For information on obtaining reprints of this article, please send e-mail to: tcbb@computer.org, and reference IEEECS Log Number TCBB-0010-0107.*
*Digital Object Identifier no. 10.1109/TCBB.2007.70268.*

also empirically demonstrate the advantage of the proposed strategies. First, we investigate different data transformation methods such as column/row standardization (CS/RS), double centering (DC), and binormalization (see Section 3). Our experimental results reveal that the "binormalization" scaling proposed by Livne and Golub [15] leads to the best performance in the context of sample accuracy. Second, we use a spectral relaxation of the objective function used in MSSRCC—the singular vectors of the gene expression matrix provide the relaxed solution. Like ordinary $k$-means-type clustering algorithms, MSSRCC suffers from the problems of being trapped in local minima and generating empty clusters. We resolve both these problems by adopting an incremental local search (LS) strategy [16], where incremental moves of rows and columns among clusters are taken in order to decrease the objective function value.

To evaluate the performance of the proposed algorithms, we first compare coclustering performance of MSSRCC with other coclustering algorithms on publicly available synthetic gene expression data sets, where we use two external measures called *average cocluster relevance* and *average cocluster recovery* (see Section 5). Furthermore, we investigate in detail its performance in various settings on publicly available human cancer microarray data sets in the literature, including Colon cancer, Leukemia, Lung cancer, and Mixed-Lineage Leukemia (MLL). We quantify the sample clustering performance using an *accuracy* measure (see Section 5). In most cases, we obtain better sample clustering accuracy by using MSSRCC than by using the ordinary one-way $k$-means algorithm. In particular, MSSRCC with the combination of binormalization, spectral initialization (SI), and LS strategy results in the best accuracy. In addition, we investigate the effectiveness of hierarchical initialization on MSSRCC. It turns out that average- or complete-linkage hierarchical clustering (HC) initialization helps generate similar performance to that with SI. We also assess gene clusters by illustrating coherence of the resulting coclusters in a checkerboard structure, where the genes in a cocluster manifest the phenotype structure of the corresponding specific samples. Then, we investigate how many informative discriminating genes reported in the literature are clustered together. Furthermore, we evaluate the enrichment of functional annotations of the gene clusters in the context of gene ontology (GO) [17] via DAVID [18], which is a functional profiling tool publicly available on the Web.

The remainder of this paper is organized as follows: Section 2 gives a brief survey of related work, focusing on existing coclustering algorithms. Section 3 summarizes MSSRCC and the proposed strategies, while Section 4 describes both the synthetic gene expression data sets and the real human cancer microarray data sets. Section 5 presents detailed empirical results that substantiate the usefulness of the proposed strategies for the MSSRCC algorithm. Finally, we conclude with a brief summary and directions for future work in Section 6.

## 2 Related Work

A vast amount of work has been done on coclustering algorithms in various contexts. For a taxonomic overview of "two-mode clustering methods," refer to the survey paper by Mechelen et al. [19]. In addition, for a survey on the application of coclustering to biological data analysis, see the recent paper by Madeira and Oliveira [20]. Since the detailed survey of existing coclustering algorithms is beyond the scope of this paper, here, we briefly overview some algorithms in chronological order.

One of the earliest coclustering formulations, block clustering was introduced by Hartigan who called it "direct clustering" [13], and it refers to the "simultaneous clustering" of both rows and columns of a data matrix [21]. Hartigan introduced various coclustering quality measures and models, including the partitional model and gave a greedy algorithm for a hierarchical coclustering model, where the partitions of row and columns can be described in a hierarchical manner by trees. Others have proposed nonhierarchical partitional coclustering algorithms that employ an alternating minimization scheme, which attempts to minimize a given objective function with respect to one variable while fixing the other, and vice versa, instead of solving over two variables (see [22] for details). For example, Gaul and Schader [23] introduced an alternating exchange algorithm for two-mode data, Baier et al. [24] developed two-mode (non-)overlapping additive clustering, and Maurizio [25] proposed an alternating least squares algorithm for the *double* $k$-means model. These algorithms are similar to our proposed algorithm, MSSRCC, in the sense that they employ an alternating optimization scheme. They tend to capture coclusters with constant values since their objectives optimize the partitioning model proposed by Hartigan [13]. In this paper, we are more interested in discovering coclusters with constant values, as well as ones with coherent trends using MSSRCC, and therefore, we omit a detailed discussion of these algorithms and focus on other specific algorithms that are directly related to our proposed approach.

Cheng and Church [14] are considered to be the first to apply coclustering, also called *biclustering*, to gene expression data. They proposed a greedy search heuristic to generate biclusters that satisfy a certain homogeneity constraint, called *mean squared residue*. The algorithm produces one cocluster at a time using a low mean squared residue (except the trivial case where all genes have constant expression values) as the criterion for identifying a cocluster. A sequence of node (i.e., row or column) deletions and additions is applied to the original matrix, while the mean squared residue of the cocluster is kept under a given threshold. After each cocluster is produced, the elements of the cocluster are replaced with random numbers, and then, the same procedure is applied to the modified matrix to generate another, possibly overlapping, cocluster until the required number of coclusters is found. Unlike this method, our algorithm finds all the coclusters simultaneously.

Since Cheng and Church [14], several similar approaches have been taken within gene expression analysis [26], [27], all of which basically utilize the definition of the mean squared residue and try to enhance the original work. For example, Yang et al. [26] point out that random number replacements can interfere with the future discovery of coclusters, especially ones that have overlap with the discovered ones. They present an algorithm called FLexible Overlapped biClustering (FLOC) that simultaneously produces $k$ coclusters whose mean residues are all less than a predefined constant. FLOC incrementally moves a row or column out of or into a cocluster depending on whether the row or column is already included in that cocluster or not, and this move is called an *action*. The idea of *action* is very similar to our incremental LS strategy in coclustering (see Section 3).

Kluger et al. [28] apply a spectral coclustering algorithm, motivated by Dhillon's bipartite formulation [29], on gene expression data to produce a "checkerboard" structure. The largest several left and right singular vectors of the normalized gene expression matrix are computed and then a final clustering step using $k$-means, and normalized cuts is applied to the data projected onto the topmost singular vectors. They incorporate different normalizations of genes and conditions in the hope of discarding irrelevant constant background noise. Similarly, we investigate the effect of various data transformations to MSSRCC.

Dhillon et al. [30] propose an information-theoretic coclustering algorithm that views a nonnegative matrix as the estimate of a (scaled) empirical joint probability distribution of two discrete random variables and poses the coclustering problem as an optimization problem in information theory, where the optimal coclustering maximizes the mutual information between the clustered random variables subject to constraints on the number of row and column clusters. This algorithm is restricted to nonnegative matrices, but the algorithm is similar to our batch coclustering algorithms with the main difference being the distance measure and the reconstruction scheme. See [31] for recent work that treats various distance measures in a unifying framework.

Recently, Bleuler et al. [32] apply an evolutionary algorithm (EA) to the biclustering formulation in [14] to explore the search space of biclusters. They also implement a LS method based on multiple node deletion and single node deletion/addition in [14]. They apply simulated annealing (SA) to overcome the local minima problem in greedy search algorithms. The main intuition behind the use of stochastic search techniques such as EA and SA is to escape from local minima in order to produce better coclusters than the greedy search algorithm of the original work at the expense of increased computation time. MSSRCC attempts to escape from local minima by interweaving global batch updates with LS steps in a ping-pong manner.

## 3 Proposed Algorithm

**Notation.** Upperase bold-faced letters like $\boldsymbol{A}$ denote matrices and lowercase bold-faced letters like $\boldsymbol{a}$ denote column vectors. $\boldsymbol{A}_{i\cdot}$ and $\boldsymbol{A}_{\cdot j}$ denote row $i$ and column $j$ of matrix $\boldsymbol{A}$, respectively. $A_{ij}$ (or $a_{ij}$) denotes the $(i,j)$th element of matrix $\boldsymbol{A}$, while $a_i$ denotes the $i$th element of vector $\boldsymbol{a}$. Uppercase letters such as $I$ and $J$ denote the set of row indices in a row cluster and the set of column indices in a column cluster, respectively. The submatrix of $\boldsymbol{A}$ determined by $I$ and $J$ is called a cocluster and denoted by $\boldsymbol{A}_{IJ}$. $\|\boldsymbol{A}\|$ denotes the *Frobenius* norm of $A$ (i.e., $\|\boldsymbol{A}\| = \sqrt{\sum_{i,j} A_{ij}^2}$). $\|\boldsymbol{a}\|_1$ denotes the $L1$-norm (i.e., $\|\boldsymbol{a}\|_1 = \sum_i |a_i|$), while $\|\boldsymbol{a}\|_2$ denotes the $L2$-norm (i.e., $\|\boldsymbol{a}\|_2 = \sqrt{\sum_i a_i^2}$).

### 3.1 Data Preprocessing

We use publicly available gene expression matrices, which have already been preprocessed in various ways using image analysis, expression quantization, normalization, and screening out. The numerical values in each data set can be very different. Therefore, we employ the following simple preprocessing steps prior to applying our coclustering algorithm.

#### 3.1.1 Gene Selection

Typically, gene expression data sets contain thousands of genes but a relatively small number of samples, often less than a hundred. Moreover, many of the genes are noisy and redundant, raising an issue of feature selection. Instead of utilizing sophisticated feature selection algorithms, we apply the following simple preprocessing steps usually adopted in microarray experiments to detect differential expression [33], [34], [35]: First, we threshold genes with a predefined $floor$ value and/or a predefined $ceil$ value; then, we filter out genes whose relative deviation ($|max/min|$) or absolute deviation ($|max - min|$) is less than predefined values, where $max$ and $min$ refer, respectively, to the maximum and minimum expression levels for a particular gene across all samples.

#### 3.1.2 Data Transformation

Transformation of the raw data is considered one of the most important steps for various data mining processes since the variance of a variable will determine its importance in a given model [36]. Recently, Wouters et al. [37] emphasize the importance of appropriate weighting in the analysis of microarray data, where they compare different transformation methods as the building blocks of three multivariate projection methods. In addition, Kluger et al. [28] showed that simultaneous $L1$ normalization of the rows and columns of a positive matrix has consistent advantage over other forms of rescaling of the raw data. Effective data transformations for coclustering algorithms have not been investigated thoroughly in prior work. Therefore, we investigate the effectiveness of the following data transformation methods.

#### 3.1.3 No Transformation (NT)

No centering or scaling is taken and the raw matrix is directly input to MSSRCC. Therefore, $a'_{ij} = a_{ij}$, $\forall i$ and $\forall j$.

#### 3.1.4 Column/Row Standardization (CS/RS)

Column standardization is defined as

$$a'_{ij} = \frac{a_{ij} - \mu_j}{\sigma_j},$$

for $i = 1, \cdots, m$ and $j = 1, \cdots, n$, where $\mu_j = \frac{1}{m}\sum_{i=1}^m a_{ij}$, and $\sigma_j^2 = \frac{1}{m}\sum_{i=1}^m (a_{ij} - \mu_j)^2$. Row standardization is defined similarly with $\mu_i = \frac{1}{n}\sum_{j=1}^n a_{ij}$, and $\sigma_i^2 = \frac{1}{n}\sum_{i=1}^n (a_{ij} - \mu_i)^2$. Column standardization results in each column having zero mean and unit variance and is also called "autoscaling" [38]. Through this standardization, the relative variation in intensity is emphasized.

### 3.1.5 Double Centering (DC)

DC is defined as

$$a'_{ij} = a_{ij} - \mu_i - \mu_j + \mu_{ij},$$

for $i = 1, \cdots, m$ and $j = 1, \cdots, n$, where

$$\mu_{ij} = \frac{1}{mn} \sum_{i=1}^{m} \sum_{j=1}^{n} a_{ij}.$$

DC has become a classic technique for background correction. Through this centering, the data matrix is projected on a hyperplane that passes through the origin and is orthogonal to the all-ones vector. It leads to a reduction by one of the rank of the original matrix [37].

### 3.1.6 Binormalization (NBIN)

Livne and Golub [15] present iterative algorithms (called BIN for a square matrix and NBIN for a rectangle matrix) for scaling all the rows and columns of a square (rectangle) matrix to have unit (same) $L2$-norm. Binormalization of a rectangular matrix through NBIN results in $\sum_{j=1}^{n} {a'_{ij}}^2 = n$ for $i = 1, \cdots, m$ and $\sum_{i=1}^{m} {a'_{ij}}^2 = m$ for $j = 1, \cdots, n$. The effect is a bisphericalization, where the rows, as well as the columns are forced to lie on hyperspheres with radii $\sqrt{n}$ and $\sqrt{m}$, respectively.

An early study on preprocessing can be found in Harshman and Lundy [39]. Recently, Bro and Smilde [40] and Smilde et al. [41] discuss a number of important features of the common preprocessing steps of centering and scaling, focusing on a two-way bilinear data analysis and generalizing the results to a multiway data analysis.

## 3.2 Minimum Sum-Squared Residue Coclustering

Let the data matrix $A \in \mathbb{R}^{m \times n}$ be partitioned into $k$ row clusters and $\ell$ column clusters. The *residue* of an element $a_{ij}$ in cocluster, i.e., $i \in I$ and $j \in J$, is defined as

$$h_{ij} = a_{ij} - a_{iJ} - a_{Ij} + a_{IJ},$$

where $a_{iJ} = \frac{1}{|J|} \sum_{j \in J} a_{ij}$ is the mean of the entries in row $i$ whose column indices are in $J$, $a_{Ij} = \frac{1}{|I|} \sum_{i \in I} a_{ij}$ is the mean of the entries in column $j$ whose row indices are in $I$, and $a_{IJ} = \frac{1}{|I| \cdot |J|} \sum_{i \in I, j \in J} a_{ij}$ is the mean of all entries in the cocluster.

We now show how to write the residue in a compact manner. Assume row-cluster $r$ $(1 \leq r \leq k)$ has $m_r$ rows, so $\sum_{r=1}^{k} m_r = m$. Similarly, column-cluster $c$ $(1 \leq c \leq \ell)$ has $n_c$ columns, so $\sum_{c=1}^{\ell} n_c = n$. Then, we define a *row cluster indicator matrix*, $\boldsymbol{R} \in \mathbb{R}^{m \times k}$ and a *column cluster indicator matrix*, $\boldsymbol{C} \in \mathbb{R}^{n \times \ell}$ as follows: The column $r$ of $\boldsymbol{R}$ has $m_r$ nonzeros, each of which is equal to $m_r^{-1/2}$, and the nonzeros of $\boldsymbol{C}$ are defined similarly. Without loss of generality, we assume that the rows and the columns that belong to a particular cluster are contiguous. Then, the matrix $\boldsymbol{R}$ has the form

TABLE 1
Summary of Computational Complexity of Various Steps

| Algorithm step | Target Row | Target Column |
|---|---|---|
| Random init. | $O(m)$ | $O(n)$ |
| Spectral init. [44] | $O\left(m^{\frac{3}{2}}k + mk^2\right)$ | $O\left(n^{\frac{3}{2}}\ell + n\ell^2\right)$ |
| Single-linkage init. | $O(m^2)$ | $O(n^2)$ |
| Complete-linkage init. | $O(m^2 logm)$ | $O(n^2 logn)$ |
| Average-linkage init. | $O(m^2 logm)$ | $O(n^2 logn)$ |
| MSSRCC | $O(tkmn)$ | $O(t\ell mn)$ |
| Local search | $O(ikm)$ | $O(j\ell n)$ |

*m = no. of rows, n = no. of columns, k = no. of row clusters, $\ell$ = no. of column clusters, t = no. of batch iterations, i = no. of row LS steps, and j = no. of column LS steps.*

$$\boldsymbol{R} = \begin{bmatrix} m_1^{-1/2} & 0 & \cdots & 0 \\ m_1^{-1/2} & 0 & \cdots & 0 \\ \vdots & 0 & \cdots & 0 \\ m_1^{-1/2} & 0 & \cdots & 0 \\ 0 & m_2^{-1/2} & \cdots & 0 \\ 0 & m_2^{-1/2} & \cdots & 0 \\ \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & \cdots & m_k^{-1/2} \\ \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & \cdots & m_k^{-1/2} \end{bmatrix},$$

where the first column has $m_1$ nonzeros, the second column has $m_2$ nonzeros, and the last (i.e., $k$th) column has $m_k$ nonzeros. Matrix $\boldsymbol{C}$ has a similar structure. Therefore, $\|\boldsymbol{R}_{\cdot r}\|_1^2 = m_r$ and $\|\boldsymbol{C}_{\cdot c}\|_1^2 = n_c$. Note that $\boldsymbol{R}$ and $\boldsymbol{C}$ are column orthonormal matrices since the columns of $\boldsymbol{R}$ and $\boldsymbol{C}$ are clearly orthogonal and $\|\boldsymbol{R}_{\cdot r}\|_2 = 1$ and $\|\boldsymbol{C}_{\cdot c}\|_2 = 1$. Then, the residue matrix, $\boldsymbol{H} = [h_{ij}]_{m \times n}$, can be written as

$$\begin{aligned} \boldsymbol{H} &= (\boldsymbol{I} - \boldsymbol{R}\boldsymbol{R}^T)\boldsymbol{A}(\boldsymbol{I} - \boldsymbol{C}\boldsymbol{C}^T) \\ &= \boldsymbol{A} - \hat{\boldsymbol{A}}, \end{aligned}$$

where $\hat{\boldsymbol{A}} = \boldsymbol{R}\boldsymbol{R}^T\boldsymbol{A} + \boldsymbol{A}\boldsymbol{C}\boldsymbol{C}^T - \boldsymbol{R}\boldsymbol{R}^T\boldsymbol{A}\boldsymbol{C}\boldsymbol{C}^T$. The minimum squared residue coclustering objective is to find row clusters $I$ and column clusters $J$ such that $\|\boldsymbol{H}\|^2 = \sum_{I,J} \sum_{i \in I, j \in J} h_{ij}^2$ is minimized. However, finding globally optimal coclusters that minimize this residue is an NP-hard problem. Cheng and Church [14] show the NP-hardness of this problem by constructing a reduction from the balanced complete bipartite subgraph problem in [42].

Cho et al. [12] propose an alternating iterative algorithm that monotonically decreases the objective function and converges to a local minimum. Algorithm 1 describes the batch update algorithm, MSSRCC. The algorithm begins with the initialization of the indicator matrices $\boldsymbol{R}$ and $\boldsymbol{C}$ based on initial clustering indicator vectors $\rho$ and $\gamma$ (step (INIT)). Notice that initial row and column clusterings are taken as inputs of MSSRCC since there are various ways to obtain initial clusterings. In this paper, we will investigate the effectiveness of five different initializations: random initialization (RI), SI, and three HC initializations (See Table 1 and Section 5.1). Each iteration involves finding the closest row (column) cluster prototype, given by a row (column) of $\boldsymbol{A}^R$ $(\boldsymbol{A}^C)$, for each row (column) of $\boldsymbol{A}^P$ and setting its row

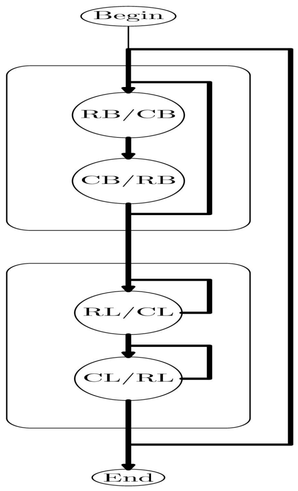

Fig. 1. The overall structure of ping-pong strategy. RB and CB stand for row batch update and column batch update in Algorithm 1, respectively. RL and CL stand for row local search and column local search, respectively.

(column) cluster accordingly (step (RB) for every row and step (CB) for every column, respectively). The matrices $A^R$, $A^C$, and $A^P$ are defined in Algorithm 1. Meanwhile, the row (column) clustering is stored in the form of the row (column) clustering indicator vector $\rho$ ($\gamma$), where $\rho(i) = r$ implies that row $i$ is in row cluster $r$; likewise, $\gamma(j) = c$ implies that column $j$ is in column cluster $c$. The algorithm iterates until the decrease in objective function becomes smaller than the specified tolerance factor $\tau$. Monotonic decrease of the objective function is proven in [12].

**Algorithm 1:** Minimum squared-sum residue coclustering algorithm (batch update only)

```
MSSRCC($A, k, \ell, \rho, \gamma$)
**Input:** Data matrix $A \in \mathbb{R}^{m\times n}$, number of row clusters $k$,
    number of column clusters $\ell$, and clustering indicator
    vectors $\rho \in \{1, \cdots, k\}^{m\times 1}$ and $\gamma \in \{1, \cdots, \ell\}^{n\times 1}$
**Output:** Clustering indicator vectors $\rho$ and $\gamma$
**begin**
    Initialize $R \in \mathbb{R}^{m\times k}$ and $C \in \mathbb{R}^{n\times \ell}$ based on the initial
    cluster assignment of $\rho$ and $\gamma$                         $(INIT)$
    $\tau \leftarrow 10^{-3}\|A\|^2$; {Adjustable parameter}
    $newobj \leftarrow \|(I - RR^T)A(I - CC^T)\|^2$
    $oldobj \leftarrow newobj + \tau + 1$
    **while** $|oldobj - newobj| > \tau$ **do**
        $A^P \leftarrow A(I - CC^T)$; $A^R \leftarrow R^T A^P$
        **for** $1 \le i \le m$ **do**
            $\rho(i) \leftarrow \arg\min_{1\le r\le k} \|A^P_{i\cdot} - m_r^{-1/2} A^R_{r\cdot}\|^2$        $(RB)$
        **end**
        $R \leftarrow$ Update using $\rho$
        $A^P \leftarrow (I - RR^T)A$; $A^C \leftarrow A^P C$
        **for** $1 \le j \le n$ **do**
            $\gamma(j) \leftarrow \arg\min_{1\le c\le \ell} \|A^P_{\cdot j} - n_c^{-1/2} A^C_{\cdot c}\|^2$        $(CB)$
        **end**
        $C \leftarrow$ Update using $\gamma$
        $oldobj \leftarrow newobj$
        $newobj \leftarrow \|(I - RR^T)A(I - CC^T)\|^2$
    **end**
**end**
```

The batch update algorithm (Algorithm 1) can get stuck in qualitatively poor local minima. Hence, the batch algorithm and an incremental LS algorithm (Algorithm 2) are alternatively invoked, as illustrated in Fig. 1. As we will demonstrate in Section 5, this "ping-pong" strategy yields qualitatively superior results.

**Algorithm 2:** Column LS step.

```
CL($A, R, C, \ell, \gamma$)
**Input**: Data matrix $A \in \mathbb{R}^{m\times n}$, Clustering indicator
    matrices $R \in \mathbb{R}^{m\times k}$ and $C \in \mathbb{R}^{n\times l}$, number of
    column clusters $\ell$, and column clustering indicator
    vector $\gamma \in \{1, \cdots, \ell\}^{n\times 1}$
**Output**: Column clustering indicator vector $\gamma$ and
    corresponding matrix $C$
**begin**
    $\tau \leftarrow 10^{-6}\|A\|^2$; {Adjustable parameter}
    $\bar{A} \leftarrow R^T A$
    **for** $1 \le j \le n$ **do**
        **for** $1 \le c' \le \ell$, $c' \ne \gamma(j) = c$ **do**
            $\delta_j(c') \leftarrow$ Update using formula (4.14) in [12]
        **end**
    **end**
    $(j^*, c^*) \leftarrow \arg\max_{(j,c)} \delta_j(c)$
    **if** $\delta_{j^*} > \tau$ **then**
        $\gamma(j^*) \leftarrow c^*$
    **end**
    $C \leftarrow$ Update using $\gamma$
**end**
```

## 3.3 Spectral Initialization (SI)

In this section, we describe a principled initialization scheme for MSSRCC. In minimizing the original objective function, the strong structural constraints on the cluster indicator matrices $R$ and $C$ make it a difficult problem. If we relax these constraints to just seek column orthogonal matrices $R$ and $C$, i.e., $R^T R = I_k$ and $C^T C = I_\ell$, then the minimization is dramatically eased.

Let $A = U\Sigma V^T$ be the singular value decomposition (SVD) of $A$. Then, it is a classical result that $A_s = U_s\Sigma_s V_s^T$ is the best rank-$s$ approximation to a matrix $A$ with respect to the *Frobenius* norm [43]. Based on this fact, it can be shown that one solution of the relaxed problem is given by $R = U_k$ and $C = V_\ell$, leading to

$$\begin{aligned}\hat{A} &= RR^T A + ACC^T - RR^T ACC^T \\ &= U_k U_k^T A + AV_\ell V_\ell^T - U_k U_k^T A V_\ell V_\ell^T \\ &= A_k + A_\ell - A_k V_\ell V_\ell^T \\ &= A_s, \quad \text{where } s = \max(k, \ell).\end{aligned}$$

Note that the relaxation shown above allows us to obtain a lower bound of $\sigma^2_{s+1} + \cdots + \sigma^2_{\mathrm{rank}(A)}$ on the value of the objective function $\|H\|^2$. Due to their global nature, spectral techniques seem to offer an ability for superior initializations. After obtaining a relaxed solution, we have to obtain a good coclustering. There are many ways to obtain a coclustering from the singular vectors. For example, we can cluster the rows of $U_k$ and $V_\ell$ using $k$-means and then obtain row and column clusters as done in [29]. Particularly, Yu and Shi [44] suggest a principled postprocessing heuristic for minimizing normalized cuts, where eigenvectors (normalized to be $L2\text{-norm}$ 1) are rotated to obtain a discrete near-global optimal solution. Their goal is to find the 0-1 partition matrix that is closest to the eigenvectors of the given weight matrix. The method consists of the following two steps: 1) eigendecomposition and orthonormal transformation for finding global optima in the relaxed continuous domain and 2) bilinear iterations for finding a discrete solution closest to continuous optima. In step 1, the first $k$ leading eigenvectors are used to generate all optima for optimal $k\text{-way}$ partitioning. In step 2, alternating minimization iterates between nonmaximum suppression to find the optimal discrete solution (i.e., set 1 for an optimal discrete cluster and set 0 for remaining clusters) and SVD to find the optimal orthonormal transform. These steps recover the rotation that aligns the columns of the partition matrix with those of the rotated eigenvector matrix that is closest in squared *Frobenius* norm (see [44] for details). Since the method generates a deterministic solution and guarantees the near-global optimum, we adapt it to get a discrete clustering solution for each dimension from the left and right singular vectors, respectively. The performance comparison between this SI and RI is discussed in Section 5 (also, see Table 1 for computational complexity). It turns out that this approach—first run spectral clustering to get an initial coclustering and then refine the result using the coclustering algorithm—results in a better coclustering of the data.

### 3.4 Incremental Local Search (LS)

In the experiments, we observe that Algorithm 1 makes large changes to the objective function value in its initial iterations, thereafter achieving smaller changes. Once the batch algorithm converges, it might suffer from two problems: qualitatively poor local minimum and presence of empty clusters. An effective technique to address these issues is to incorporate an incremental LS strategy [16] into the standard batch algorithm. A single step of row/column incremental LS looks for the single row/column move that maximizes the change of objective function value. Algorithm 2 describes a single column LS step, where $\delta_j(c')$ keeps track of the change of objective function value by moving column $j$ from cluster $c = \gamma(j)$ to cluster $c' \neq c$ (refer to [12] for derivation of update formulation and efficient implementation details). Furthermore, we can take a finite sequence (say, 20) of such moves (i.e., the chain of LSs) until the moves improve the objective function value. Such a LS strategy has been shown to be effective in escaping poor local minima resulting from the batch algorithm and avoiding empty clusters in both ordinary one-way $k\text{-means}$ [16] and the minimum sum-squared coclustering algorithms [12]. Note that moving a row or column from its current cluster to an empty cluster always leads to a decrease in the objective function (assuming nondegeneracy) and guarantees that no cluster is empty (see supplementary material for the proof, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TCBB.2007.70268).

In Fig. 1, we summarize the overall structure of the ping-pong strategy. It consists of two main algorithm steps, one for the row and/or column batch update (upper box) and the other for the row and/or column LS (lower box). Basically, one ping-pong iteration (outer loop) performs batch updates followed by LS updates (see Table 1 for complexity of various steps). Notice that we can start with either row batch update or column batch update and other variations are possible (see supplementary material, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TCBB.2007.70268, for other possible variations).

## 4 Description of Data Sets

To validate the performance of the proposed algorithm, we consider both synthetic and real gene expression data sets as follows:

### 4.1 Synthetic Gene Expression Data Sets

Recently, Prelić et al. [45] evaluate the performance of five prominent coclustering algorithms and provide synthetic gene expression data sets. Based on Madeira and Oliveira [20], they consider: 1) two types of cocluster concepts, coclusters with constant values (i.e., constant coclusters) and coclusters with coherent values over the conditions (i.e., additive coclusters) and 2) two types of cocluster structures, nonoverlapping coclusters, and overlapping coclusters. Each data matrix contains 10 nonoverlapping implanted coclusters, and each cocluster extends over 10 genes and 5 conditions, therefore, its dimension is 100 $\times$ 50. Furthermore, to study the effect of noise, they consider various noise levels and generate 10 input matrices for each considered noise level (see supplementary material in [45]). Since MSSRCC generates nonoverlapping exhaustive coclusters, we use only the data matrices with nonoverlapping additive coclusters in our experiments.

### 4.2 Real Gene Expression Data Sets

We target four human cancer microarray data sets and their various gene subsets. The microarray data sets are summarized in Table 2, and the details are explained below. Note that all the arrays were originally generated using Affymetrix technology.

#### 4.2.1 Colon Cancer

Alon et al. [46] selected only the genes $(m = 2{,}000)$ with highest minimal intensity across the samples and placed them in order of descending minimal intensity from a total of 6,500 human genes. The samples $(n = 62)$ consist of tumorous (40 samples) and normal (22 samples) colon tissues. Note that the raw gene expression matrix contains no negative values and only 1,909 unique expressed sequence tags (ESTs) exist among 2,000 genes because some ESTs are repeated. We first reduced the genes to 1,096 genes by removing genes with $|max/min| < 15$ and $|max - min| < 500$, see Table 2 for details. Further, the gene expression values were transformed by taking the base-10 logarithm.

#### 4.2.2 Leukemia

Golub et al. [9] used gene expression levels $(m = 7,129)$ along samples $(n = 72)$ suffering from leukemia. Each sample belongs to either acute lymphoblastic leukemia (ALL, 47 samples) or acute myeloid leukemia (AML, 25 samples). Following the preprocessing steps in [35], we

TABLE 2
Description of Microarray Data Sets Used in Our Experiments

| Dataset | Colon | Leukemia | Lung | MLL |
|---|---|---|---|---|
| Source | [46] | [9] | [47] | [48] |
| No. of original genes | 2000 | 7129 | 12533 | 12582 |
| No. of samples | 62 | 72 | 181 | 72 |
| No. of sample classes | 2 | 2 | 2 | 3 |
| Sample class names | Normal(20) Tumor(42) | ALL(47) AML(25) | ADCA(150) MPM(31) | ALL(24) AML(25) MLL(23) |
| Relative deviation threshold, $\|max/min\|$ | 15 | 5 | 5 | 5 |
| Absolute deviation threshold, $\|max-min\|$ | 500 | 500 | 600 | 5500 |
| No. of remaining genes | 1096 | 3571 | 2401 | 2474 |

first thresholded genes by using a floor of 100 and ceiling of 16,000 and then further screened out genes with $|max/min| < 5$ and $|max - min| < 500$, leaving a total of 3,571 genes.

### 4.2.3 Lung Cancer

Gordon et al. [47] provided human tissue samples $(n = 181)$ consisting of malignant pleural mesothelioma (MPM, 31 samples) and adenocarcinoma (ADCA, 150 samples with 139 patient tumors and 11 duplicates) of the lung. Each sample is described by 12,533 genes $(m = 12{,}533)$. For our experiment, we screened out genes by fixing $|max/min| < 5$ and varying absolute deviation value so that the resulting dimension is reduced to 1/5th of the original dimension, leaving a total of 2,401 genes with $|max - min| \geq 600$.

### 4.2.4 MLL Data Set

Armstrong et al. [48] obtained gene expression values $(m = 12{,}582)$ over all samples $(n = 72)$. Each sample was diagnosed by pathologists as one of the three types of leukemia: ALL (24 samples), AML (28 samples), and MLL (20 samples). We applied the same preprocessing step used for the Lung data set (i.e., fixing $|max/min| < 5$ and adjusting absolute deviation filter in order to reduce the gene size to 1/5th of the original size), leaving a total of 2,474 genes with $|max - min| \geq 5{,}500$.

After the simple relative and absolute thresholding, we obtain a reduced number of genes from each data set, given in the last row in Table 2. We will refer to the data sets with these reduced gene sets as *the reduced gene subsets*. In addition, various gene subsets of Colon cancer, Leukemia, and MLL have been identified in the literature to have discriminating capability among the specific cancer types considered in this paper. These gene subsets were selected from either an individual platform or cross-platform combinations using various feature selection algorithms (see supplementary material, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TCBB.2007.70268, for details). We collect these gene subsets and also generate new gene subsets by either intersection or union of the related subsets. We will refer to these gene sets as *the selected gene subsets* to emphasize that these genes were selected using more sophisticated feature selection methods. We prepare these discriminative gene subsets because 1) less computation time is required with smaller dimensional data, 2) more stable and reliable clustering can be obtained with less noisy and more coherent data, and 3) clustering performance can be evaluated with their discriminating and separating capabilities. By utilizing the selected discriminating gene subsets, we intend to find the best performing strategy for MSSRCC and then evaluate the gene clustering quality of the reduced gene subsets in Table 2 with the best strategy.

## 5 Experimental Results

We implement the proposed algorithms in C++, perform all experiments on a workstation (Linux, Intel Pentium 2.53 GHz, 1 Gbyte of main memory), and generate figures using MATLAB. As specified in Algorithms 1 and 2, $\tau$ is adjustable, and one might use other values to adjust convergence speed. In the experiments, we fix $\tau = 10^{-3}\|A\|^2$ for the batch updates and $\tau = 10^{-6}\|A\|^2$ for the LS steps. Note that $\tau$ for the batch updates should be larger than that for the LS steps to proceed with the ping-pong strategy. Algorithm 1 generates $10 \times 10$ (i.e., $k = 10$ and $l = 10$) coclusters for the synthetic data sets within 5 seconds and $100 \times 2$ (i.e., $k = 100$ and $l = 2$) or $100 \times 3$ (i.e., $k = 100$ and $l = 3$) coclusters within $5 \sim 20$ seconds for all the data sets. However, SVD for the SI and the LS take additional time.

We evaluate gene cluster quality using the following external evaluation measure, *average cocluster relevance* (see [45] for details) defined as

$$S^*_G(M, M_{opt}) = \frac{1}{|M|} \sum_{(G,C)\in M} \max_{(G_{opt},C_{opt})\in M_{opt}} \frac{|G \cap G_{opt}|}{|G \cup G_{opt}|},$$

where $M$ denotes the coclusters from a coclustering algorithm, and $M_{opt}$ is the set of implanted coclusters, each of which contains a gene set $G$ (or $G_{opt}$) and a condition set $C$ (or $C_{opt}$). This score reflects to what extent the generated coclusters represent true coclusters in the gene dimension. Similarly, *average cocluster recovery* is defined as $S^*_G(M_{opt}, M)$ and quantifies how well each of the true coclusters is recovered by the corresponding coclustering algorithm. Both scores take a maximum value of 1, if $M = M_{opt}$.

In addition, we assess sample clustering quality according to a priori assigned class labels using the external evaluation measure, *accuracy*, defined as

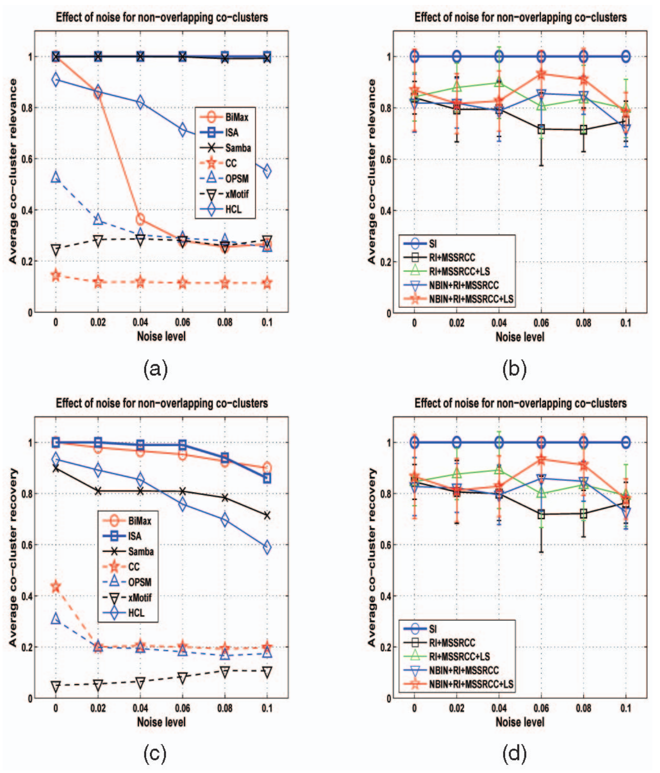

Fig. 2. Effect of noise on coclustering algorithms: (a) and (b) average cocluster relevance and (c) and (d) average cocluster recovery. Plots (a) and (c) are reproduced based on that in [45] for comparison with the performance of MSSRCC as shown in (b) and (d). Note that standard deviations of performance with MSSRCC are shown in plots (b) and (d). Abbrevations: RI is for random initialization, SI is for spectral initializations, NBIN is for binormalization, and LS is for local search.

$$accuracy(\%) = \frac{1}{T}\left(\sum_{i=1}^{\ell} t_i\right) \times 100,$$

where $T$ denotes the total number of samples, $\ell$ denotes the number of sample clusters, and $t_i$ denotes the number of samples correctly clustered into sample class $i$. We first form a confusion matrix whose $(i,j)$th entry gives the number of samples in cluster $i$ that belong to the true class $j$. Each $t_i$ is a diagonal element of the corresponding confusion matrix whose cluster labels are permuted so that the sum of diagonal elements is maximized.

Since we usually do not have prior knowledge about which genes are relevant, we adopt other criteria to evaluate gene clustering. We visualize the coherence of genes in some gene clusters using checkerboard plots and evaluate the gene clusters by checking how many well-known discriminating genes in the literature are clustered together. In addition, we assess the enrichment of GO biological functional categories in some gene clusters via DAVID [18].

In the upcoming sections, we evaluate the performance of the various preprocessing schemes (i.e., NT, RS/CS, DC, and NBIN), the initialization schemes (i.e., RI and SI), and LS. See Sections 3.1, 3.3, and 3.4 for details.

## 5.1 Comparison of Coclustering Algorithms

For the comparison study, we utilize the synthetic gene expression data sets (having nonoverlapping additive coclusters) and the "Biclustering Analysis Toolbox (BicAT)," which implements the following coclustering algorithms: Cheng and Church's (CC) algorithm, Samba, Order Preserving Submatrix Algorithm (OPSM); Iterative Signature Algorithm (ISA); xMotif; and BiMax (see [45] for a detailed discussion of each algorithm).

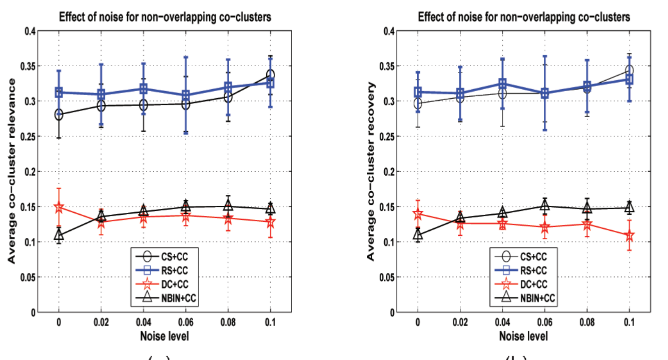

Fig. 3. Effect of data transformation on the CC algorithm: (a) average cocluster relevance and (b) average cocluster recovery. Abbreviations: CS/RS is for column/row standardization, DC is for double centering, and NBIN is for binormalization.

We partition the synthetic data sets into 10 × 10 coclusters using MSSRCC with the proposed strategies and as done in [45], the scores are averaged over the 10 input matrices at each corresponding noise level. Fig. 2 compares the performance of our proposed strategies with several existing coclustering algorithms. With SI, we are able to recover all the 10 implanted coclusters over all the considered noise levels (thus, further algorithmic steps are not necessary after SI). However, without SI, MSSRCC fails to capture all the coclusters like other coclustering algorithms (except BiMax). For example, MSSRCC with RI shows the worst performance (see Fig. 2b and Fig. 2d), and its performance decreases as noise increases. However, it still generates better performance than other coclustering algorithms such as CC, OPSM, and xMotif (see Fig. 2a and Fig. 2c). In summary, we have the following:

1. SI itself obtains perfect recovery.
2. In general, LS itself improves the overall performance.
3. MSSRCC with NBIN leads to better performance at increasing noise levels.
4. MSSRCC with RI results in better performance with either NBIN or LS and best with both NBIN and LS.

We observe in our experiments that MSSRCC with RS and MSSRCC with NBIN give comparable performance (see supplementary material, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TCBB.2007.70268). However, we emphasize in Fig. 2 the performance of MSSRCC with NBIN, because it results in a good performance in general on both synthetic and real gene expression data sets. Furthermore, we summarize the effect of data transformation on the CC algorithm in Fig. 3. CS and RS are beneficial for CC and result in better performance than CC without data transformation in Fig. 2. Interestingly, NBIN and DC hardly improve the performance of CC, and they do not seem to be appropriate for CC.

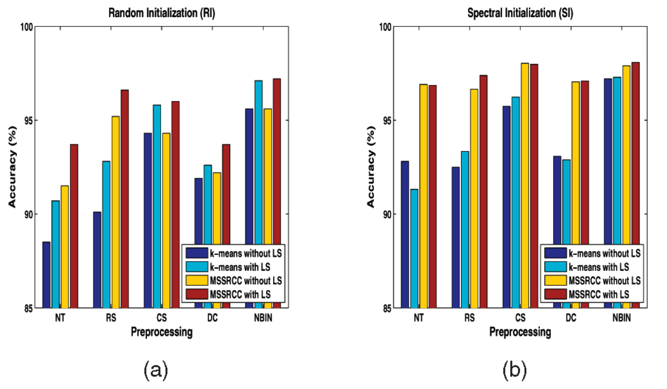

Fig. 4. Average sample clustering accuracy for all the selected discriminating gene subsets: (a) RI and (b) SI. For the cases using $k$-means algorithm, the accuracy is averaged over 1,000 random runs. Abbreviations: NT is for number of transformations, CS/RS is for column/row standardization, DC is for double centering, NBIN is for binormalization, and LS is for local search. See supplementary material, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TCBB.2007.70268, for the detailed performance of each strategy on the individual data sets.

## 5.2 Evaluation of Sample Clustering

We utilize all the selected gene subsets in order to find out the best combination of the strategies proposed in Section 3: for NBIN, two initializations (RI and SI), two algorithms ($k$-means and MSSRCC), and two update strategies (without LS and with LS) and for the five data transformations, NT, RS/CS, DC and NBIN, two initializations (RI and SI), two algorithms ($k$-means and MSSRCC), and two update strategies (with/without LS). Consequently, we have a total of 40 different combinations of the strategies.

Fig. 4 summarizes the accuracy values of all the 40 combinations, each of which is averaged over all the selected gene subsets. The average accuracy varies with the individual strategy as follows: RI and SI achieve 90.98 percent and 94.10 percent of accuracy, respectively; NT, RS, CS, DC, and NBIN obtain 90.25 percent, 91.74 percent, 93.64 percent, 91.56 percent, and 95.51 percent, respectively; $k$-means and MSSRCC lead to 90.95 percent and 94.13 percent, respectively; and without-LS and with-LS result in 91.82 percent and 93.26 percent, respectively. Interestingly, DC only gives a slightly better accuracy than NT, even though DC is widely used in various algorithms [28], [49], [36]. More interestingly, the combination of NT, RI, and $k$-means without LS gives the worst accuracy of 84.29 percent and the combination of NBIN, SI, MSSRCC, with LS obtains the best accuracy of 97.43 percent. In summary, we obtain a good accuracy with an individual strategy, NBIN, SI, or MSSRCC, and better accuracy with combination of these strategies than an individual strategy. The best performing combination is to employ all the four strategies.

In order to verify the effectiveness of the best performing strategy on larger dimensional data sets, we perform further experiments with the data sets in Table 2, which performs better than $k$-means algorithm in the previous experiments. Here, fixing NBIN and MSSRCC, we evaluate performance of initialization and LS strategies varying the number of gene clusters from 1 to 100. Fig. 5 illustrates the initial and final average accuracies with different strategies. Each strategy performs differently on different data sets. However, in general, SI contributes more to the improvement of accuracy than LS, and SI itself does give better initial accuracy than RI in addition to better final accuracy. In addition, for most cases, we obtain smaller standard deviations with LS than without LS. All the empirical results support the necessity of the combination of NBIN, SI, and MSSRCC with LS to get the best accuracy.

Finally, in Table 3, we summarize the average sample clustering performance (with the best performing combination, i.e., MSSRCC with NBIN, SI, and LS) for the reduced gene subsets in Table 2, and the selected gene subsets whose details are given in the supplementary material, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TCBB.2007.70268. We observe that accuracy values for the reduced gene subsets (second column in Table 3) are significantly worse than these for the corresponding selected gene subsets (third column in Table 3). We ascribe this result to the difference in data preprocessing: the reduced gene subsets are obtained using unsupervised heuristics (i.e., relative and absolute deviation thresholds), while the selected gene subsets were carefully selected using specific feature selection techniques. The result also supports the observation that most genes individually do not offer good discriminative ability, but small subsets of genes together lead to better discriminative information for clustering as well for classification [33], [48], [50], [51]. It is

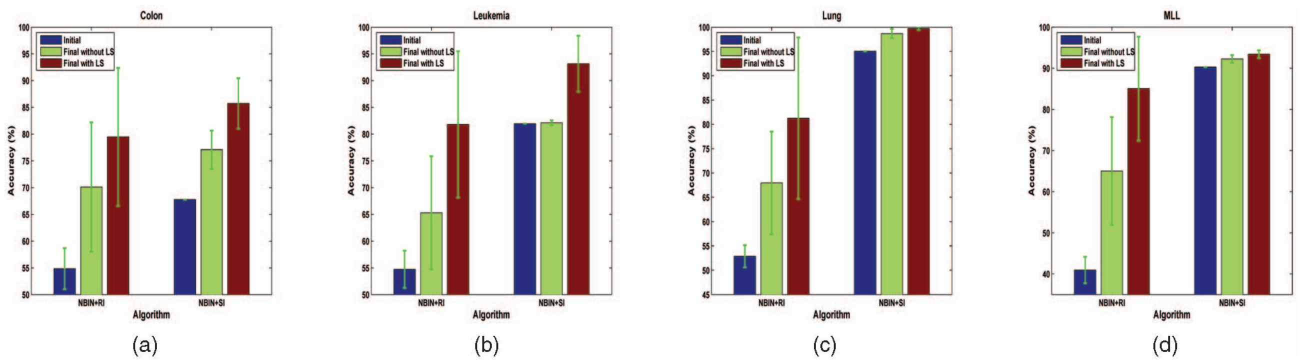

Fig. 5. Average sample clustering accuracy using MSSRCC with NBIN for the data sets in Table 2. The average and standard deviation values are obtained over 100 gene clusters ranging from 1 to 100. Abbreviations: NBIN is for binormalization, RI/SI is for random/spectral initialization, and LS is for local search.

TABLE 3
Comparison of the Average Sample Clustering Accuracy with the Best Performing Strategy

| | Average accuracy (%) | |
|---|---|---|
| Dataset | Reduced gene subsets | Selected gene subsets |
| Colon | 85.73 | 92.74 |
| Leukemia | 93.14 | 97.02 |
| Lung | 99.69 | $n/a$ |
| MLL | 93.40 | 99.58 |

*Note that selected gene subsets of Lung cancer are not available in the literature.*

therefore desirable to employ sophisticated dimensionality reduction techniques to keep such small subsets of genes.

## 5.3 MSSRCC Improves Hierarchical Clustering

HC itself has been reported to generate worse clustering results for gene expression data than other common clustering algorithms such as $k$-means [52], [53], [54]. However, HC is popular in analyzing genomic data, because of the early availability of free clustering and visualization software (see Eisen et al. [8]). Moreover, HC has been successfully applied as an initialization for other clustering algorithms [52], [55], [56]. In this paper, we extensively apply this idea (i.e., HC as a specific initialization for other clustering algorithms) to MSSRCC and evaluate its effectiveness on performance of MSSRCC. For this experiment, we consider HC algorithms with correlation-based distance metric and three variants using simple, complete, and average linkage algorithms. We utilize the CLUSTERDATA function in MATLAB to generate 100 seeds (i.e., the number of gene clusters is varied from 1 to 100) for each data set in Table 2, fixing the number of sample clusters to the number of true classes.

It has been observed that single-linkage HC suffers from outliers and chaining effect, performs poorly among the three linkages, and even shows worse performance than $k$-means with RI [52], [54], [56], [57]. Our experimental results in Fig. 6 are consistent with this observation. Interestingly, all the initial clusterings with single-linkage are not improved by MSSRCC without LS (see the first two bars in each plot in Fig. 6). In our experiments, the single-linkage initialization happens to generate initial singleton sample clusters as follows: Colon results in sample 13, Leukemia results in sample 18, and MLL results in sample 21 and sample 26. In addition, the single-linkage initialization results in two samples of Lung data set, sample 64 and sample 107, being put into one cluster. Since a singleton cluster is a trivial case whose residue value is zero, as discussed in [12], no points move to initial singleton clusters during the batch update. Therefore, initial singleton clusters still remain intact after the batch update. However, the chain of the LS strategy helps other points in moving to singleton clusters and eventually improves the clustering (see the third bar in each plot in Fig. 6).

In general, both the average- and the complete-linkage initializations do not generate initial singleton clusters. Accordingly, they start with a better initial clustering than the single-linkage initialization. Their initial clustering is improved using MSSRCC without LS and further improved with LS. Therefore, either average- or complete-linkage hierarchical initialization followed by MSSRCC is a desirable strategy (note that Gibbons and Roth [54] concluded that the complete-linkage is the only appropriate HC method to analyze microarray experiments). It is also worth mentioning that MSSRCC with hierarchical initialization obtains deterministic clustering results like our deterministic SI. We summarize accuracies with different strategies as follows: for Colon, $85.02 \pm 4.23$ percent (with average-linkage HC + LS) versus $85.73 \pm 4.72$ percent (with SI + LS); for Leukemia, $91.47 \pm 6.76$ percent (with complete-linkage HC + LS) versus $93.14 \pm 5.23$ percent (with SI + LS); for Lung Cancer, $99.45 \pm 0.06$ percent (with average-linkage HC + LS) versus $99.69 \pm 0.27$ percent (with SI + LS); and for MLL, $72.11 \pm 5.56$ percent (with complete-linkage HC + LS) versus $93.40 \pm 0.93$ percent (with SI + LS). Notice that even though MSSRCC can correct initial cluster assignments brought up by HC, the final assignments are not better than (but comparable to) those from SI followed by MSSRCC.

## 5.4 Evaluation of Gene Clusters

In this section, we evaluate the quality of some gene clusters obtained using MSSRCC with the best performing strategy. Here, we focus on some gene clusters that are frequently clustered together with high sample cluster accuracy and whose expression levels are clearly distinguished from other clusters. All the other gene clusters not shown in Figs. 7, 8, 9, and 10 can be found at the supplementary

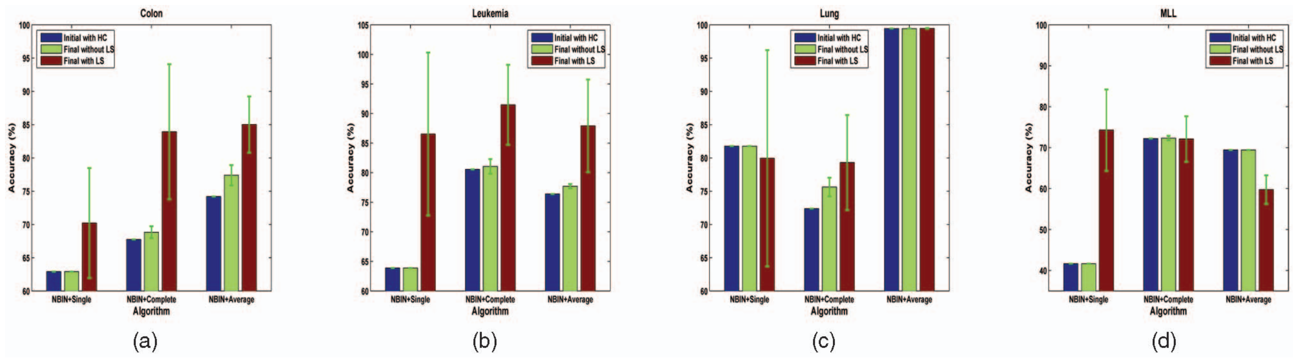

Fig. 6. Average sample clustering accuracy using MSSRCC with NBIN and HC initialization. The average and standard deviation values are obtained over 100 gene clusters ranging from 1 to 100. Abbreviations: HC is for hierarchical clustering, Single/Complete/Average is for single/complete/average linkage, and LS is for local search.

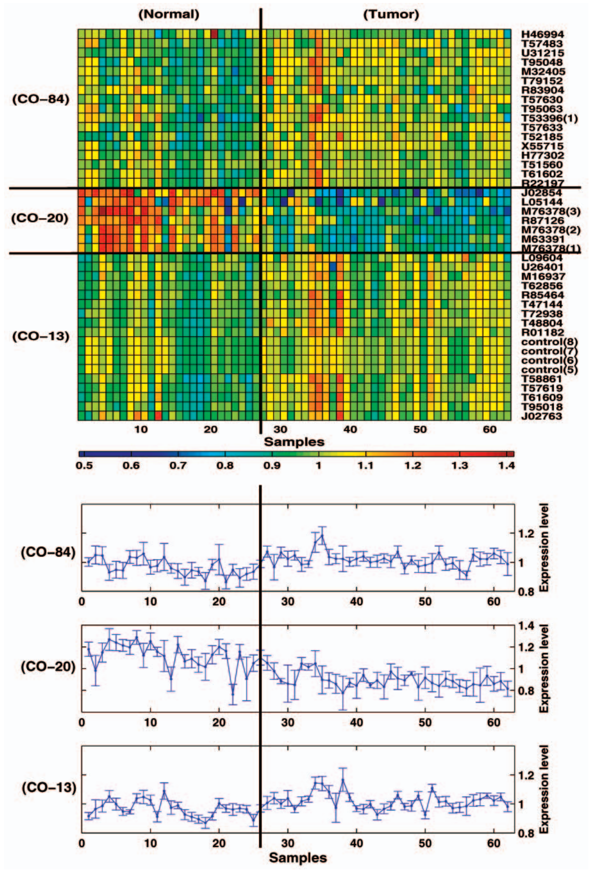

Fig. 7. Six exemplary coclusters of Colon cancer. The GenBank reference ID for each gene is listed on the right side of the plot. To distinguish the repeated ESTs, "(a number)" is assigned after the GenBank reference. The mean expression level of all genes in each gene cluster is shown in the corresponding errorbar plot. The gene clusters (CO-13), (CO-20), and (CO-84) contain 18, 7, and 17 genes, respectively. The sample clusters (Normal) and (Tumor) contain 26 and 36 samples, respectively, where eight samples are incorrectly clustered (i.e., $Accuracy = (54/62) \times 100 = 87.1$ percent).

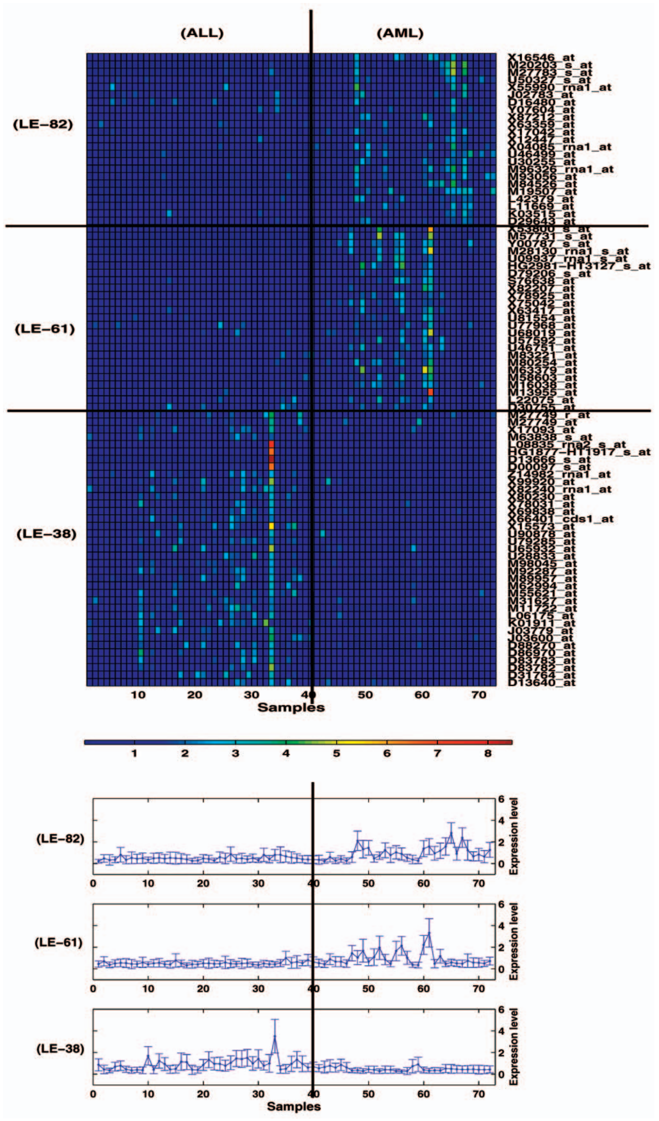

Fig. 8. Six exemplary coclusters of Leukemia. The Affymetrix probe set ID for each gene is listed on the right side of the plot. The mean expression level of all genes in each gene cluster is shown in the corresponding errorbar plot. The gene clusters (LE-38), (LE-61), and (LE-82) contain 37, 25, and 23 genes, respectively. The sample clusters (ALL) and (AML) contain 40 and 32 samples, respectively, where nine samples are incorrectly clustered (i.e., $Accuracy = (63/72) \times 100 = 87.5$ percent).

material, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TCBB.2007.70268.

### 5.4.1 Gene Clusters in the Colon Cancer Data Set

Fig. 7 illustrates the pseudocolor image of the $3 \times 2$ coclusters of Colon cancer and the corresponding mean expression level errorbars. As mentioned in [46], the ribosomal genes are partly upregulated in tumor samples (i.e., red vertical stripes) and comparatively downregulated in normal samples (i.e., blue vertical stripes) of Colon cancer. The (mean) expression levels clearly illustrate that MSSRCC captures homogeneous gene expression patterns. Among 29 distinct ribosomal genes in the reduced gene set of 1,096 genes, seven ribosomal genes reside in (CO-13), and 13 others reside in (CO-84). As in [46], we observe that the ribosomal protein genes are partially upregulated in the colon tumor tissues. In particular, (CO-20) contains six genes selected in either [33] or [50], and Fig. 7 illustrates that those six discriminative genes are differently expressed: upregulated in normal samples but downregulated in tumor samples. A similar pattern is observed in [50].

### 5.4.2 Gene Clusters in Leukemia Data Set

Fig. 8 shows $3 \times 2$ coclusters of Leukemia, each of which contains a subset of genes that are highly expressed (i.e., yellow or red vertical stripes) in either ALL or AML samples. For example, the genes in (LE-38) are partly but highly expressed in ALL samples, while the genes in (LE-61) and (LE-82) are highly expressed in AML samples. Each of the three gene clusters contains many discriminating genes in [33], [58], and [9]: (LE-38) contains six discriminative genes, (LE-61) contains six discriminative genes, and (LE-82) contains seven discriminative genes. In addition, see supplementary material, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TCBB.2007.70268, for other discriminating gene clusters.

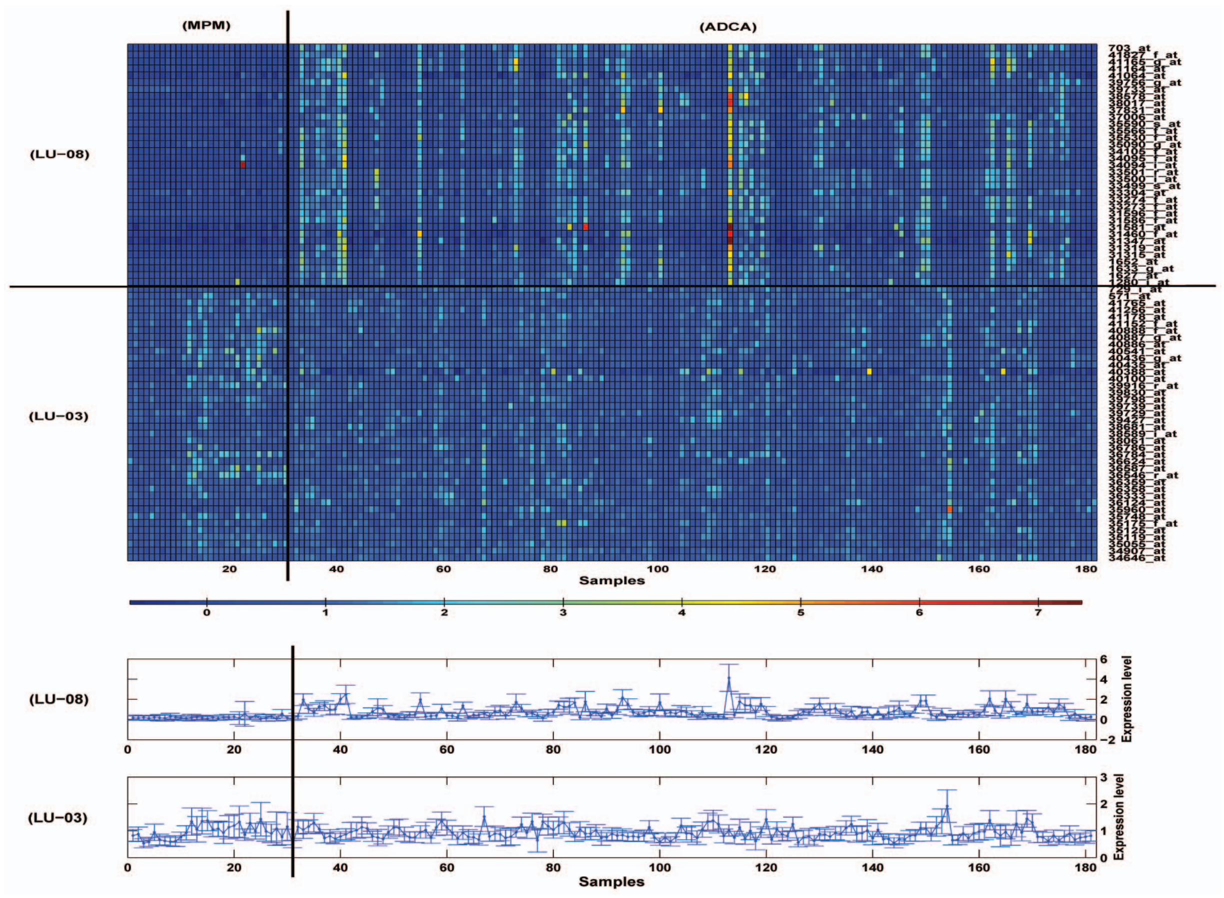

Fig. 9. Four exemplary coclusters of Lung cancer. The Affymetrix probe set ID for each gene is listed on the right side of the plot. The mean expression level of all genes in each gene cluster is shown in the corresponding errorbar plot. The gene clusters (LU-03) and (LU-08) contain 111 and 35 genes, respectively. However, only 40 among 111 genes in (LU-03) are shown in the plot. The sample clusters (MPM) and (ADCA) contain 30 and 151, where only one sample is incorrectly clustered (i.e., $Accuracy = (150/151) \times 100 = 99.4$ percent).

### 5.4.3 Gene Clusters in Lung Cancer Data Set

Fig. 9 shows 2 × 2 coclusters of Lung cancer. (LU-03) and (LU-08) depict a ribosomal gene cluster and an immunoglobulin-related gene cluster, respectively. Among 84 ribosomal genes in the reduced gene set, 68 ribosomal genes reside in (LU-03). (LU-03) also contains some genes related to cellular metabolism or cell growth such as 1288_*s_at* (*EEF1A1*), 36587_*at* (*EEF2*), 35175_*f_at* (*EEF1A2*), and 35748_*at* (*EEF1B2*). These clusters are consistent with the ribosomal clusters in [46] and [59]. Further, there are 95 immunoglobulin-related genes in the original data set, and 37 genes are left after preprocessing. Immunoglobulin-related genes have been a main target in immunology society because of their influence on the development of various tumors and interaction with oncogenes or tumor suppressor genes. Some recent studies on their relationship to lung cancer are reported in [60] and [61]. We observe that (LU-08) contains 21 immunoglobulin-related genes, which are relatively downregulated in MPM samples and upregulated in ADCA samples. This observation is consistent with the four immunoglobulin-related genes in the list of the top 20 differently expressed genes for lung ADCA in [62] and the four immunoglobulin-related genes in [63]. Still, several other clusters consist of major dominant genes in each cluster (see supplementary material, which can be found on the Computer Society Digital Library at http://doi.ieee computersociety.org/10.1109/TCBB.2007.70268).

### 5.4.4 Gene Clusters in MLL Data Set

Fig. 10 depicts 3 × 3 coclusters of MLL. It clearly illustrates the coherent gene expression level of each cluster, where a group of genes is correlated with a group of samples. In other words, the genes in a same gene cluster are over- or underexpressed over only a part of samples, which result in bright or dark vertical stripes in a cocluster. For example, the genes in (ML-47) and (ML-91) are overexpressed in ALL samples, while the genes in (ML-89) are overexpressed in AML samples and highly overexpressed in MLL samples. In particular, (ML-47) contains four discriminating genes in [48], (ML-91) contains five discriminating genes, and (ML-89) includes 10 MLL-discriminating genes. Several other gene clusters having a dominant gene in each cluster are discovered and listed in supplementary material, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TCBB.2007.70268. These demonstrate that MSSRCC has a capability to capture coherent genes.

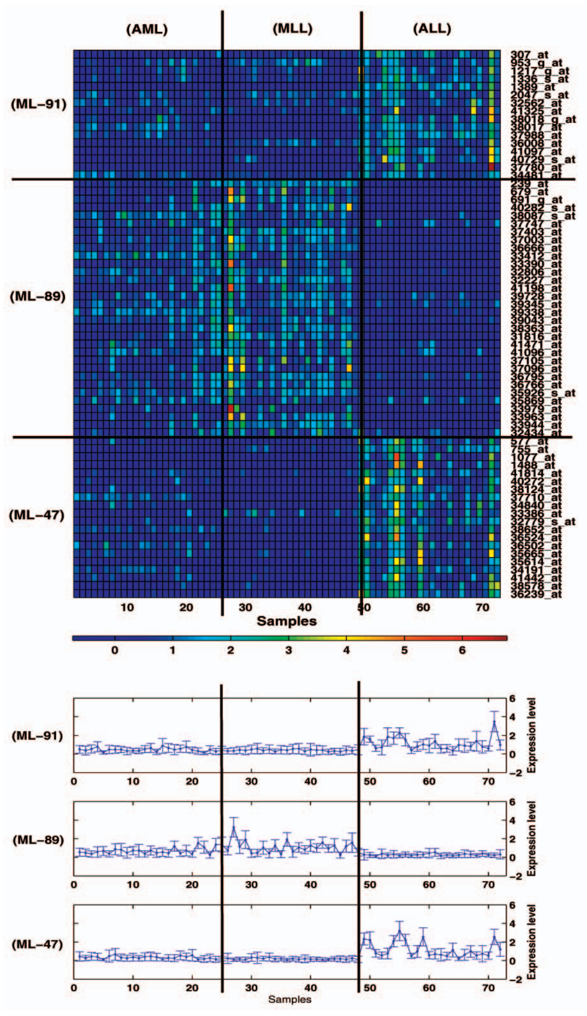

Fig. 10. Six exemplary coclusters of MLL. The Affymetrix probe set ID for each gene is listed on the right side of the plot. The mean expression level of all genes in each gene cluster is shown in the corresponding errorbar plot. The gene clusters (ML-47), (ML-89), and (ML-91) contain 20, 32, and 16 genes, respectively. The sample clusters (AML), (MLL), and (ALL) contain 25, 23, and 24 samples, where five samples are incorrectly clustered (i.e., $Accuracy = (67/72) \times 100 = 93.1$ percent).

### 5.4.5 Analysis of Biological Annotation Enrichment

The aim of this analysis is to assess the statistical significance of the interesting gene clusters, i.e., the enrichment of functional annotations. We identify GO "biological process" categories with significantly enriched gene numbers in each gene set with respect to the corresponding reference gene set of the reduced gene subsets in Table 2. Here, we focus on the same gene clusters that we investigate previously, since they recover the main block-structures associated with different cancer types. However, one might find other significant gene clusters by investigating all the gene clusters and built-in genome or Affy chip background.

Among many publicly available functional profiling tools, we utilize the DAVID software [18]. DAVID adopts Fisher Exact Test to measure the gene enrichment in annotation terms (refer to [64] for comparison of statistical tests for detecting significant changes of gene expression). Fisher Exact $p$-value for gene-enrichment analysis ranges from 0 to 1, where a Fisher Exact $p$-value of 0 represents perfect enrichment. Usually, categories with Fisher Exact $p\text{-value} < 0.05$ are considered statistically significant. Due to space limitation, we list only the first five enriched categories (sorted by $p$-value) for each gene cluster considered in Table 4. All the categories listed in Table 4 have $p\text{-value} < 0.05$, and the following GO biological process categories are dominantly enriched in each gene cluster: biosynthesis and metabolism in both (CO-13) and (CO-84); response and catabolism in (LE-38); regulation, transcription, and response in (LE-61); catabolism and metabolism in (LE-82); biosynthesis and metabolism in (LU-03); response and antigen in (LU-08); development in (ML-47); and response in (ML-89). Remaining categories and further biological interpretation are available in the supplementary material, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TCBB.2007.70268.

## 6 Conclusions and Future Work

Motivated by the observation that sets of genes that are similarly regulated across samples have related functions, we employ a fast coclustering algorithm to simultaneously discover those subsets of both genes and samples that are correlated. Our coclustering algorithm, MSSRCC, partitions both genes and samples based on expression profiles, thus all coclusters from our algorithm uncover a so-called nonoverlapping "checkerboard" structure.

We propose specific strategies to enhance the performance of MSSRCC. We compare its performance with other coclustering algorithms on two synthetic gene expression data sets. In addition, the strength of the proposed algorithm is demonstrated on four publicly available human cancer microarray data sets: Colon cancer, Leukemia, Lung cancer, and MLL. We evaluate the sample clustering of each data set using the external measure of *accuracy*. Our empirical results demonstrate that an individual strategy (i.e., NBIN, SI, MSSRCC, or LS) contributes toward better accuracy for most of the gene sets that we consider. A combination of these strategies gives better accuracy than an individual strategy. In particular, the best combination is to employ NBIN, SI, and MSSRCC with LS together.

We find that complete-linkage or average-linkage HC initialization start with better initial and end with better final clustering than both single-linkage HC and RI. MSSRCC can correct initial cluster assignments brought up by HC, and the final clustering is comparable to that from SI.

We also assess the gene clustering quality by visualizing the coherence of genes in some coclusters in checkerboard plots and observe that the selected coherent coclusters contain many discriminative genes identified in the literature. Furthermore, we evaluate functional enrichment of gene clusters according to major biological processes in GO.

In addition, MSSRCC prevents empty clusters by means of the LS strategy; discovers coclusters with coherent expression profiles, resulting in "checkerboard" patterns, many of which contain the discriminative genes previously identified in the literature; and produces very stable sample

TABLE 4
Enrichment of the GO Biological Process Categories in Gene Clusters

| Cluster | Biological Process | Count | Percentage | $p$-value |
|---|---|---|---|---|
| (CO-13) | protein biosynthesis | 7 | 50.0 | $1.5E-4$ |
| | macromolecule biosynthesis | 7 | 50.0 | $2.8E-4$ |
| | cellular biosynthesis | 8 | 57.1 | $4.4E-4$ |
| | biosynthesis | 8 | 57.1 | $8.0E-4$ |
| | cellular protein metabolism | 7 | 50.0 | $6.4E-2$ |
| (CO-20) | muscle contraction | 2 | 50.0 | $3.4E-2$ |
| | regulation of organismal physiological process | 2 | 50.0 | $4.4E-2$ |
| (CO-84) | protein biosynthesis | 12 | 70.6 | $3.2E-11$ |
| | macromolecule biosynthesis | 12 | 70.6 | $1.2E-10$ |
| | cellular biosynthesis | 12 | 70.6 | $2.1E-8$ |
| | biosynthesis | 12 | 70.6 | $5.7E-8$ |
| | cellular protein metabolism | 12 | 70.6 | $1.4E-5$ |
| (LE-38) | defense response | 14 | 36.8 | $3.5E-4$ |
| | response to biotic stimulus | 14 | 36.8 | $5.0E-4$ |
| | immune response | 13 | 34.2 | $5.8E-4$ |
| | response to stimulus | 18 | 47.4 | $9.7E-4$ |
| | organismal physiological process | 16 | 42.1 | $2.7E-3$ |
| (LE-61) | regulation of viral life cycle | 3 | 12.0 | $3.3E-4$ |
| | regulation of viral genome replication | 3 | 12.0 | $3.3E-4$ |
| | taxis | 5 | 20.0 | $5.2E-4$ |
| | locomotory behavior | 5 | 20.0 | $5.2E-4$ |
| | chemotaxis | 5 | 20.0 | $5.2E-4$ |
| (LE-82) | cellular catabolism | 7 | 31.8 | $2.6E-4$ |
| | catabolism | 7 | 31.8 | $7.1E-4$ |
| | cellular macromolecule catabolism | 5 | 22.7 | $3.8E-3$ |
| | generation of precursor metabolites and energy | 6 | 27.3 | $4.3E-3$ |
| | macromolecule catabolism | 5 | 22.7 | $6.6E-3$ |
| (LU-03) | protein biosynthesis | 75 | 69.4 | $1.3E-70$ |
| | macromolecule biosynthesis | 75 | 69.4 | $8.5E-65$ |
| | cellular biosynthesis | 78 | 72.2 | $8.6E-54$ |
| | biosynthesis | 78 | 72.2 | $4.2E-50$ |
| | cellular protein metabolism | 81 | 75.0 | $3.5E-33$ |
| (LU-08) | response to biotic stimulus | 13 | 48.1 | $7.8E-8$ |
| | response to stimulus | 15 | 55.6 | $4.6E-7$ |
| | defense response | 12 | 44.4 | $5.9E-7$ |
| | immune response | 11 | 40.7 | $3.4E-6$ |
| | antigen processing, endogenous antigen via MHC class I | 4 | 14.8 | $5.4E-4$ |
| (ML-47) | cell differentiation | 5 | 27.8 | $1.2E-3$ |
| | organ development | 5 | 27.8 | $2.0E-3$ |
| | development | 7 | 38.9 | $6.9E-3$ |
| | hemopoiesis | 3 | 16.7 | $1.5E-2$ |
| | hemopoietic or lymphoid organ development | 3 | 16.7 | $1.6E-2$ |
| (ML-89) | response to other organism | 10 | 32.3 | $3.7E-5$ |
| | defense response | 12 | 38.7 | $4.7E-5$ |
| | response to biotic stimulus | 12 | 38.7 | $7.7E-5$ |
| | immune response | 11 | 35.5 | $1.3E-4$ |
| | organismal physiological process | 13 | 41.9 | $1.3E-4$ |
| (ML-91) | cell-cell signaling | 4 | 28.6 | $1.1E-2$ |
| | cell communication | 8 | 57.1 | $1.4E-2$ |
| | organismal physiological process | 6 | 42.9 | $3.3E-2$ |

*For each cluster, only the first five enriched categories sorted by Fisher Exact p-value using DAVID are listed. Percentage refers to the percentage of unique DAVID genes in the list associated with a particular annotation term.*

clustering even with substantial variation in the number of gene clusters.

Finally, we list some avenues to be explored in the future. We observe that the performance of MSSRCC is best with the selected discriminative gene subsets. Further, little overlap of the identified genes in the literature suggests that there may exist multiple other gene sets that have similar or better discriminative capability. These observations reinforce the importance of feature quality and the need for sophisticated feature selection algorithms. We may vary the number of gene clusters to generate finer clusters for researchers to look at; however, it would be better if we can estimate the optimal number of gene/sample clusters. Currently, MSSRCC produces nonoverlapping coclusters, and we plan to enhance our coclustering algorithm to generate overlapping or soft coclusters.

## 7 Supplementary Information

The coclustering source code in C++ is available under the terms of the GNU Public License (GPL) from our Website.[1] All the gene lists of the selected genes and the list of 100 gene clusters for each data set are also available.

## Acknowledgments

The authors thank the anonymous reviewers for the constructive comments and suggestions. Special thanks to S. Bleuler at ETH Zürich for providing the data for Fig. 2a and Fig. 2c. This research was supported by US National Science Foundation (NSF) Grant CCF-0431257, US NSF Career Awards ACI-0093404 and IIS-0713142, and US NSF-ITR award IIS-0325116.

## References

[1] J.L. DeRisi, V.R. Iyer, and P.O. Brown, "Exploring the Metabolic and Genetic Control of Gene Expression on a Genomic Scale," *Science,* vol. 278, no. 5338, pp. 680-686, 1997.

[2] P.F. Macgregor and J.A. Squire, "Application of Microarrays to the Analysis of Gene Expression in Cancer," *Clinical Chemistry,* vol. 48, no. 8, pp. 1170-1177, 2002.

[3] D.K. Slonim, "From Patterns to Pathways: Gene Expression Data Analysis Comes of Age," *Nature Genetics Supplement,* vol. 32, pp. 502-508, 2002.

[4] M. Schena, *Microarray Analysis.* John Wiley & Sons, 2003.

[5] M.F. Ochs and A.K. Godwin, "Microarrays in Cancer: Research and Applications," *BioTechniques,* vol. 34, pp. S4-S15, 2003.

[6] D. Jiang, C. Tang, and A. Zhang, "Cluster Analysis for Gene Expression Data: A Survey," *IEEE Trans. Knowledge and Data Eng.,* vol. 16, no. 11, pp. 1370-1386, Nov. 2004.

[7] R. Shamir and R. Sharan, "Algorithmic Approaches to Clustering Gene Expression Data," *Current Topics in Computational Molecular Biology,* pp. 269-299, MIT Press, 2002.

[8] M.B. Eisen, P.T. Spellman, P.O. Brown, and D. Botstein, "Cluster Analysis and Display of Genome-Wide Expression Patterns," *Proc. Nat'l Academy of Science,* vol. 95, no. 25, pp. 14 863-14 868, 1998.

[9] T.R. Golub et al., "Molecular Classification of Cancer: Class Discovery and Class Prediction by Gene Expression Monitoring," *Science,* vol. 286, pp. 531-537, 1999.

[10] A.A. Alizadeh et al., "Distinct Types of Diffuse Large B-Cell Lymphoma Identified by Gene Expression Profiling," *Nature,* vol. 403, no. 6769, pp. 503-511, 2000.

[11] A. Ben-Dor, B. Chor, R.M. Karp, and Z. Yakhini, "Discovering Local Structure in Gene Expression Data: The Order-Preserving Submatrix Problem," *J. Computational Biology,* vol. 10, nos. 3-4, pp. 373-384, 2003.

[12] H. Cho, I.S. Dhillon, Y. Guan, and S. Sra, "Minimum Sum-Squared Residue Based Co-clustering of Gene Expression Data," *Proc. Fourth SIAM Int'l Conf. Data Mining (SDM '04),* pp. 114-125, 2004.

[13] J.A. Hartigan, "Direct Clustering of a Data Matrix," *J. Am. Statistical Assoc.,* vol. 67, no. 337, pp. 123-129, 1972.

[14] Y. Cheng and G.M. Church, "Biclustering of Expression Data," *Proc. Eighth Int'l Conf. Intelligent Systems for Molecular Biology (ISMB '00),* vol. 8, pp. 93-103, 2000.

[15] O.E. Livne and G.H. Golub, "Scaling by Binormalization," *Numerical Algorithms,* vol. 35, no. 1, pp. 97-120, 2004.

[16] I.S. Dhillon, Y. Guan, and J. Kogan, "Iterative Clustering of High Dimensional Text Data Augmented by Local Search," *Proc. Second IEEE Int'l Conf. Data Mining (ICDM),* 2002.

[17] T.G.O. Consortium, "Gene Ontology: Tool for the Unification of Biology," *Nature Genetics,* vol. 25, pp. 25-29, 2000.

[18] G. Dennis Jr. et al., "DAVID: Database for Annotation, Visualization, and Integrated Discovery," *Genome Biology,* vol. 4, no. R60, 2003.

[19] I.V. Mechelen, H. Bock, and P.D. Boeck, "Two-Mode Clustering Methods: A Structured Overview," *Statistical Methods in Medical Research,* vol. 13, pp. 363-394, 2004.

[20] S.C. Madeira and A.L. Oliveira, "Biclustering Algorithms for Biological Data Analysis: A Survey," *IEEE Trans. Computational Biology and Bioinformatics,* vol. 1, no. 1, pp. 24-45, Jan.-Mar. 2004.

[21] B. Mirkin, *Mathematical Classification and Clustering.* Kluwer Academic Publishers, 1996.

[22] I. Csiszár and G. Tusnády, "Information Geometry and Alternating Minimization Procedure," *Statistics and Decisions,* supplemental issue, vol. 1, pp. 205-237, 1984.

[23] W. Gaul and M. Schader, "A New Algorithm for Two-Mode Clustering," *Data Analysis and Information Systems,* H. Hermann and W. Polasek, eds., pp. 15-23, Springer, 1996.

[24] D. Baier, W. Gaul, and M. Schader, "Two-Mode Overlapping Clustering with Applications to Simultaneous Benefit Segmentation and Market Structuring," *Classification and Knowledge Organization: Recent Advances and Applications,* R. Klar and O. Opitz, eds., pp. 557-566, Springer, 1997.

[25] V. Maurizio, "Double $k$-means Clustering for Simultaneous Classification of Objects and Variables," *Advances in Classification and Data Analysis,* S. Borra, R. Rocci, M. Vichi, and M. Schader, eds., pp. 43-52, Springer, 2001.

[26] J. Yang, H. Wang, W. Wang, and P. Yu, "Enhanced Biclustering on Expression Data," *Proc. Third IEEE Symp. Bioinformatics and BioEngineering (BIBE '03),* pp. 321-327, 2003.

[27] J. Yang, W. Wang, H. Wang, and P. Yu, "$\delta$-Clusters: Capturing Subspace Correlation in a Large Data Set," *Proc. 18th IEEE Int'l Conf. Data Eng. (ICDE '02),* pp. 517-528, 2002.

[28] Y. Kluger, R. Basri, J.T. Chang, and M. Gerstein, "Spectral Biclustering of Microarray Data: Coclustering of Genes and Conditions," *Genome Research,* vol. 13, no. 4, pp. 703-716, 2003.

[29] I.S. Dhillon, "Co-Clustering Documents and Words Using Bipartite Spectral Graph Partitioning," *Proc. Seventh ACM Int'l Conf. Knowledge Discovery and Data Mining (SIGKDD '01),* pp. 269-274, 2001.

[30] I.S. Dhillon, S. Mallela, and D.S. Modha, "Information-Theoretic Co-Clustering," *Proc. Ninth ACM Int'l Conf. Knowledge Discovery and Data Mining (SIGKDD '03),* pp. 89-98, 2003.

[31] A. Banerjee, I.S. Dhillon, J. Ghosh, S. Merugu, and D.S. Modha, "A Generalized Maximum Entropy Approach to Bregman Co-Clustering and Matrix Approximation," *J. Machine Learning Research,* vol. 8, pp. 1919-1986, 2007.

[32] S. Bleuler, A. Prelić, and E. Zitzler, "An EA Framework for Biclustering of Gene Expression Data," *Proc. Sixth Congress on Evolutionary Computation (CEC '04),* pp. 166-173, 2004.

[33] T.H. Bø and I. Jonassen, "New Feature Subset Selection Procedures for Classification of Expression Profiles," *Genome Biology,* vol. 3, no. 4, 2002.

[34] M. Dettling and P. Bühlmann, "Supervised Clustering of Genes," *Genome Biology,* vol. 3, no. 12, 2002.

[35] S. Dudoit and J. Fridlyand, "A Prediction-Based Resampling Method for Estimating the Number of Clusters in a Dataset," *Genome Biology,* vol. 3, no. 7, pp. 0036.1-0036.21, 2002.

[36] F.C. Sánchez, P.J. Lewi, and D.L. Massart, "Effect of Different Preprocessing Methods for Principal Component Analysis Applied to the Composition of Mixtures: Detection of Impurities in HPLC-DAD," *Chemometrics and Intelligent Laboratory Systems,* vol. 25, no. 2, pp. 157-177, 1994.

[37] L. Wouters et al., "Graphical Exploration of Gene Expression Data: A Comparative Study of Three Multivariate Methods," *Biometrics,* vol. 59, pp. 1131-1139, 2003.

[38] B.R. Kowalski and C.F. Bender, "Pattern Recognition: A Powerful Approach to Interpreting Chemical Data," *J. Am. Chemical Soc.,* vol. 94, no. 16, pp. 5632-5639, 1972.

[39] R.A. Harshman and M.E. Lundy, "Data Preprocessing and the Extended PARAFAC Model," *Research Methods for Multimode Data Analysis,* pp. 216-284, Praeger, 1984.

[40] R. Bro and A.K. Smilde, "Centering and Scaling in Component Analysis," *J. Chemometrics,* vol. 17, pp. 16-33, 2003.

[41] A. Smilde, R. Bro, and P. Geladi, "Preprocessing" *Multi-Way Analysis with Applications in the Chemical Sciences,* pp. 221-255, John Wiley & Sons, 2004.

[42] D.S. Johnson, "The NP-Completeness Column: An Ongoing Guide," *J. Algorithms,* vol. 8, no. 3, pp. 438-448, 1987.

[43] C. Eckart and G. Young, "The Approximation of One Matrix by Another of Lower Rank," *Psychometrika,* vol. 1, pp. 211-218, 1936.

[44] S.X. Yu and J. Shi, "Multiclass Spectral Clustering," *Proc. Ninth IEEE Int'l Conf. Computer Vision,* 2003.

1. http://www.cs.utexas.edu/users/dml/Software/cocluster.html.

[45] A. Prelić et al., "A Systematic Comparison and Evaluation of Biclustering Methods for Gene Expression Data," *Bioinformatics,* vol. 22, no. 9, pp. 1122-1129, 2006.
[46] U. Alon et al., "Broad Patterns of Gene Expression Revealed by Clustering Analysis of Tumor and Normal Colon Tissues Probed by Oligonucleotide Arrays," *Proc. Nat'l Academy of Science,* vol. 96, no. 12, pp. 6745-6750, 1999.
[47] G.J. Gordon et al., "Translation of Microarray Data into Clinically Relevant Cancer Diagnostic Tests Using Gene Expression Ratios in Lung Cancer and Mesothelioma," *Cancer Research,* vol. 62, pp. 4963-4967, 2002.
[48] S.A. Armstrong et al., "MLL Translocations Specify a Distinct Gene Expression Profile that Distinguishes a Unique Leukemia," *Nature Genetics,* vol. 30, pp. 41-47, 2002.
[49] P.J. Lewi, "Spectral Map Analysis: Factorial Analysis of Contrast, Especially from Log Ratios," *Chemometrics and Intelligent Laboratory Systems,* vol. 5, no. 2, pp. 105-116, 1989.
[50] M. Xiong, W. Li, J. Zhao, L. Jin, and E. Boerwinkle, "Feature (Gene) Selection in Gene Expression-Based Tumor Classification," *Molecular Genetics and Metabolism,* vol. 73, pp. 239-247, 2001.
[51] E. Yeoh et al., "Classification, Subtype Discovery, and Prediction of Outcome in Pediatric Acute Lymphoblastic Leukemia by Gene Expression Profiling," *Cancer Cell,* vol. 1, no. 2, pp. 133-143, 2002.
[52] I.G. Costa, F.A. de Carvalho, and M.C. de Souto, "Comparative Analysis of Clustering Methods for Gene Expression Time Course Data," *Genetics and Molecular Biology,* vol. 27, no. 4, pp. 623-631, 2004.
[53] S. Datta and S. Datta, "Comparisons and Validation of Statistical Clustering Techniques for Microarray Gene Expression Data," *Bioinformatics,* vol. 19, pp. 459-466, 2003.
[54] F.D. Gibbons and F.P. Roth, "Judging the Quality of Gene Expression-Based Clustering Methods Using Gene Annotation," *Genome Research,* vol. 12, pp. 1574-1581, 2002.
[55] J. Chen, X. He, and L. Li, "Identifying the Patterns of Hematopoietic Stem Cells Gene Expressions Using Clustering Methods: Comparison and Summary," *J. Data Science,* vol. 2, pp. 297-379, 2004.
[56] K.Y. Yeung, D.R. Haynor, and W.L. Ruzzo, "Validating Clustering for Gene Expression Data," *Bioinformatics,* vol. 17, no. 4, pp. 309-318, 2001.
[57] J. Handl, J. Knowles, and D.B. Kell, "Computational Cluster Validation in Post-Genomic Data Analysis," *Bioinformatics,* vol. 21, no. 15, pp. 3201-3212, 2005.
[58] M.L. Chow, E.J. Moler, and I.S. Mian, "Identifying Marker Genes in Transcription Profiling Data Using a Mixture of Feature Relevance Experts," *Physiological Genomics,* vol. 5, pp. 99-111, 2001.
[59] G. Getz, E. Levine, and E. Domany, "Coupled Two-Way Clustering Analysis of Gene Microarray Data," *Proc. Nat'l Academy of Science,* vol. 97, no. 22, pp. 12 079-12 084, 2000.
[60] X. Qiu et al., "Human Epithelial Cancers Secrete Immunoglobulin G with Unidentified Specificity to Promote Growth and Survival of Tumor Cells," *Cancer Research,* vol. 63, pp. 6488-6495, 2003.
[61] N. Tsai, B. Chen, S. Wei, C. Wu, and S.R. Roffler, "Anti-Tumor Immunoglobulin M Increases Lung Metastasis in an Experimental Model of Malignant Melanoma," *Clinical and Experimental Metastasis,* vol. 20, pp. 103-109, 2003.
[62] T.J. Giordano et al., "Organ-Specific Molecular Classification of Primary Lung, Colon, and Ovarian Adenocarcinomas Using Gene Expression Profiles," *Am. J. Pathology,* vol. 159, no. 4, pp. 1231-1238, 2001.
[63] M. Nacht et al., "Molecular Characteristics of Non-Small Cell Lung Cancer," *Proc. Nat'l Academy of Science,* vol. 98, no. 26, pp. 15 203-15 208, 2001.
[64] M.Z. Man, X. Wang, and Y. Wang, "POWER_SAGE: Comparing Statistical Tests for Sage Experiments," *Bioinformatics,* vol. 16, no. 11, pp. 953-959, 2000.

**Hyuk Cho** received the BE degree from Chonbuk National University, the MA degree from Korea University, and the MS degree from the University of Texas, Austin. He is a PhD candidate of computer sciences at the University of Texas, Austin. His research interests include data mining, statistical pattern recognition, and machine learning. He is currently working on coclustering algorithms and their applications to various tasks in data/text/web mining, bioinformatics, and pervasive computing. He is a student member of the IEEE and SIAM.

**Inderjit S. Dhillon** received the BTech degree from the Indian Institute of Technology, Bombay, and the PhD degree from the University of California, Berkeley. He is an associate professor of computer sciences at the University of Texas, Austin. His main research interests are in numerical analysis, data mining, and machine learning. He is well known for his work on computational algorithms in these areas, in particular on eigenvalue computations, clustering, coclustering, and matrix approximations. Software based on his research on eigenvalue computations is now part of all state-of-the-art numerical software libraries. He received an US National Science Foundation Career Award in 2001, a best paper award at the SIAM data mining conference in 2003, a University Research Excellence Award in 2005, and the SIAG/LA Prize in 2006. He is a member of the ACM, the IEEE, and SIAM.

▷ **For more information on this or any other computing topic, please visit our Digital Library at** www.computer.org/publications/dlib.