Convex Biclustering

Eric C. Chi\textsuperscript{1,*}, Genevera I. Allen\textsuperscript{2,3}, and Richard G. Baraniuk\textsuperscript{3}

\textsuperscript{1}Department of Statistics, North Carolina State University, 2311 Stinson Dr, Raleigh, North Carolina, U.S.A.

\textsuperscript{2}Department of Statistics, Rice University, 6100 Main St, Houston, Texas, U.S.A.

\textsuperscript{3}Department of Electrical and Computer Engineering, Rice University, 6100 Main St, Houston, Texas, U.S.A.

*email: eric.chi@ncsu.edu

Summary: In the biclustering problem, we seek to simultaneously group observations and features. While biclustering has applications in a wide array of domains, ranging from text mining to collaborative filtering, the problem of identifying structure in high dimensional genomic data motivates this work. In this context, biclustering enables us to identify subsets of genes that are co-expressed only within a subset of experimental conditions. We present a convex formulation of the biclustering problem that possesses a unique global minimizer and an iterative algorithm, COBRA, that is guaranteed to identify it. Our approach generates an entire solution path of possible biclusters as a single tuning parameter is varied. We also show how to reduce the problem of selecting this tuning parameter to solving a trivial modification of the convex biclustering problem. The key contributions of our work are its simplicity, interpretability, and algorithmic guarantees - features that arguably are lacking in the current alternative algorithms. We demonstrate the advantages of our approach, which includes stably and reproducibly identifying biclusterings, on simulated and real microarray data.

Key words: Clustering; Convex Optimization; Fused Lasso; Gene Expression; Reproducible Research; Structured Sparsity.

This paper has been submitted for consideration for publication in Biometrics
1. Introduction

In the biclustering problem, we seek to simultaneously group observations (columns) and features (rows) in a data matrix. Such data is sometimes described as two-way, or transposable, to put the rows and columns on equal footing and to emphasize the desire to uncover structure in both the row and column variables. Biclustering is used for visualization and exploratory analysis in a wide array of domains. For example, in text mining, biclustering can identify subgroups of documents with similar properties with respect to a subgroup of words (Dhillon, 2001). In collaborative filtering, it can be used to identify subgroups of customers with similar preferences for a subset of products (Hofmann and Puzicha, 1999). Comprehensive reviews of biclustering methods can be found in (Madeira and Oliveira, 2004; Tanay et al., 2005; Busygin et al., 2008).

In this work, we focus on biclustering to identify patterns in high dimensional cancer genomic data. While a cancer, such as breast cancer, may present clinically as a homogenous disease, it typically consists of several distinct subtypes at the molecular level. A fundamental goal of cancer research is the identification of subtypes of cancerous tumors that have similar molecular profiles and the genes that characterize each of the subtypes. Identifying these patterns is the first step towards developing personalized treatment strategies targeted to a patient’s particular cancer subtype.

Subtype discovery can be posed as a biclustering problem in which gene expression data is partitioned into a checkerboard-like pattern that highlights the associations between groups of patients and groups of genes that distinguish those patients. Biclustering has had some notable successes in subtype discovery. For instance, biclustering breast cancer data has identified sets of genes whose expression levels segregated patients into five subtypes with distinct survival outcomes.
These subtypes have been reproduced in numerous studies (Sørlie et al., 2001). Encouraged by these results, scientists have searched for molecular subtypes in other cancers, such as ovarian cancer (Tothill et al., 2008). Unfortunately, the findings in many of these additional studies have not been as reproducible as those identified by Sørlie et al. (2001). The failure to reproduce these other results may reflect a genuine absence of biologically meaningful groupings. But another possibility may be related to issues inherent in the computational methods currently used to identify biclusters.

While numerous methods for biclustering genomic data have been proposed (Madeira and Oliveira, 2004; Busygin et al., 2008), the most popular approach to biclustering cancer genomics data is the clustered dendrogram. This method performs hierarchical clustering (Hastie et al., 2009) on the patients (columns) as well as on the genes (rows). The matrix is then re-ordered according to these separate row and column dendrograms and visualized as a heatmap with dendrograms plotted alongside the row and column axes. Figure 1a illustrates an example of a clustered dendrogram of expression data from a lung cancer study. The data consists of the expression levels of 500 genes across 56 individuals, a subset of the data studied in Lee et al. (2010). Subjects belong to one of four subgroups: Normal, Carcinoid, Colon, or Small Cell.

This simple strategy seems to reasonably recover the clinical diagnoses and identify sets of genes whose dysregularization characterizes the subtypes.

As an algorithmic procedure, however, the clustered dendrogram has two characteristics that make it less than ideal for generating reproducible results. Dendrograms are constructed by greedily fusing observations (features) to decrease some criterion. Consequently the algorithm may return a biclustering that is only
locally optimal with respect to the criterion. Since solutions may vary depending
on how the algorithm is initialized, such procedures tend to be run from multiple
initializations, but even then there is no guarantee that a global optimum will be
reached. The algorithm is also not stable in the sense that small perturbations in
the data can lead to large changes in the clustering assignments.

More sophisticated methods have been proposed for biclustering, some based on
the singular value decomposition (SVD) of the data matrix (Lazzeroni and Owen,
2002; Bergmann et al., 2003; Turner et al., 2005; Witten et al., 2009; Lee et al.,
2010; Sill et al., 2011), and others based on graph cuts (Dhillon, 2001; Kluger
et al., 2003). Some approaches are similar in spirit to the clustered dendrogram
and directly cluster the rows and columns (Coifman and Gavish, 2011; Tan and
Witten, 2014). While these methods may provide worthwhile improvements in
empirical performance, none of them address the two fundamental issues that dog
the clustered dendrogram. Moreover, scientists may shy from using many of these
methods, since their outputs are typically not as easy to visualize as compared
to the simple clustered dendrogram. From a reproducible research perspective, a
biclustering method should (i) give the same, ideally unique, answer regardless of
how the algorithm is initialized, and (ii) be stable with respect perturbations in
the data.

In this paper, we pose biclustering as a convex optimization problem and intro-
duce a novel Convex BiclusterRing Algorithm (COBRA) for iteratively solving it.
COBRA outputs results that retain the simple interpretability and visualization of
the clustered dendrogram and also possesses several key advantages over existing
techniques: (i) **Stability and Uniqueness**: COBRA produces the unique global
minimizer to a convex program and this minimizer is continuous in the data. This
means that COBRA always maps the data to a single biclustering assignment, and
this solution is stable. (ii) **Simplicity:** COBRA employs a single tuning parameter that controls the number of biclusters. (iii) **Data Adapivity:** COBRA admits a simple and principled data adaptive procedure for choosing the tuning parameter that involves solving a convex matrix completion problem.

Returning to our motivating lung cancer example, Figure 1b illustrates the results of COBRA with the tuning parameter selected according to our data adaptive procedure. After reordering the columns and rows via the seriation package (Hahsler et al., 2008), with the ‘TSP’ option, a clearer biclustering patterns emerge.

2. **A Convex Formulation of Biclustering**

Our goal is to identify the groups of rows and groups of columns in a data matrix that are associated with each other. As seen in Figure 1b, when rows and columns are reordered according to their groupings, a checkerboard pattern emerges, namely the elements of the matrix partitions defined by row and column groups tend to display a uniform intensity.

We now describe a probabilistic model that can generate the observed checkerboard pattern. Our data, $X \in \mathbb{R}^{p \times n}$, consists of $n$ samples drawn from a $p$-dimensional feature space. Suppose that the latent checkerboard structure is defined by $R$ feature groups and $C$ observation groups. If the $ij$th entry in $X$ belongs to the cluster defined by the $r$th feature group and $c$th observation group, then we assume that the observed value $x_{ij}$ is given by $x_{ij} = \mu_0 + \mu_{rc} + \varepsilon_{ij}$, where $\mu_0$ is a baseline or grand mean shared by all entries of the data matrix, $\mu_{rc}$ is the mean of the cluster defined by the $r$th row partition and $c$th column partition, and $\varepsilon_{ij}$ are iid $N(0, \sigma^2)$ for some $\sigma^2 > 0$. To ensure identifiability of the mean parameters, we assume that $\mu_0 = 0$, which can be achieved by removing the grand sample mean from the data matrix $X$. 
This biclustering model corresponds to a checkerboard mean model (Madeira and Oliveira, 2004). This model is most similar to that assumed by Tan and Witten (2014) who propose methods to estimate a checkerboard-like structure with some of the bicluster mean entries being sparse. The checkerboard model is exhaustive in that each matrix element is assigned to one bicluster. This is in contrast to other biclustering models that identify potentially overlapping row and column subsets that are not exhaustive; these are typically estimated using SVD-like methods (Lazzeroni and Owen, 2002; Bergmann et al., 2003; Turner et al., 2005; Witten et al., 2009; Lee et al., 2010; Sill et al., 2011) or methods to find hot-spots (Shabalin et al., 2009).

Estimating the checkerboard model parameters consists of finding the partitions and the mean values of each partition. Estimating $\mu_{rc}$, given feature and observation clusterings, is trivial. Let $R$ and $C$ denote the indices of the $r$th row partition and $c$th column partition. The maximum likelihood estimate of $\mu_{rc}$ is simply the sample mean of the entries of $X$ over the indices defined by $R$ and $C$, namely

$$
\mu_{rc} = \frac{1}{|R||C|} \sum_{i \in R, j \in C} x_{ij}.
$$

In contrast, estimating the row and column partitions, is a combinatorially hard problem and characterizes the main objective of biclustering. This task is akin to best subset selection in regression problems (Hastie et al., 2009). However, just as the best subset selection problem has been successfully attacked by solving a convex surrogate problem, namely the Lasso (Tibshirani, 1996), we will develop a convex relaxation of the combinatorially hard problem of selecting the row and column partitions.

We propose to identify the partitions by minimizing the following convex crite-
\[ F_\gamma(U) = \frac{1}{2} \|X - U\|_F^2 + \gamma J(U), \]  
(1)

where \( \Omega_W(U) = \sum_{i<j} w_{ij} \| U_{i} - U_{j} \|_2 \), and \( U_{i} (U_{c}) \) denotes the \( i \)th column (row) of the matrix \( U \).

We have posed the biclustering problem as a penalized regression problem, where the matrix \( U \in \mathbb{R}^{p \times n} \) is our estimate of the means matrix \( \mu \). The quadratic term quantifies how well \( U \) approximates \( X \). The regularization term \( J(U) \) penalizes deviations away from a checkerboard pattern. The parameter \( \gamma \geq 0 \) tunes the tradeoff between the two terms. The parameters \( w_{ij} = w_{ji} \) and \( \tilde{w}_{ij} = \tilde{w}_{ji} \) are non-negative weights that will be explained shortly.

The penalty term \( J(U) \) is closely related to other sparsity inducing penalties. When only the rows or columns are being clustered, minimizing the objective function in (1) corresponds to solving a convex clustering problem (Pelckmans et al., 2005; Hocking et al., 2011; Lindsten et al., 2011) under an \( \ell_2 \)-norm fusion penalty. The convex clustering problem in turn can be seen as a generalization of the Fused Lasso (Tibshirani et al., 2005). When the \( \ell_1 \)-norm is used in place of the \( \ell_2 \)-norm in \( \Omega_W(U) \), we recover a special case of the general Fused Lasso (Tibshirani and Taylor, 2011). Other norms can also be employed in our framework; see for example Chi and Lange (2015). In this paper, we restrict ourselves to the \( \ell_2 \)-norm since it is rotationally invariant. In general, we do not want a procedure whose biclustering output may non-trivially change when the coordinate representation of the data is trivially changed.

To understand how the regularization term \( J(U) \) steers solutions toward a checkerboard pattern, consider the effects of \( \Omega_W(U) \) and \( \Omega_{\tilde{W}}(U^T) \) separately. Suppose \( J(U) = \Omega_W(U) \). The \( i \)th column \( U_{i} \) of the matrix \( U \) can be viewed
as a cluster center or centroid attached to the $i$th column $X_i$ of the data matrix $X$. When $\gamma = 0$, the minimum is attained when $U_i = X_i$, and each column occupies a unique column cluster. As $\gamma$ increases, the cluster centroids are shrunk together and in fact begin to coalesce. Two columns $X_i$ and $X_j$ are assigned to the same column partition if $U_i = U_j$. We will prove later that for sufficiently large $\gamma$, all columns coalesce into a single cluster. Similarly, suppose $J(U) = \Omega W(U^T)$ and view the rows of $U$ as the cluster centroids attached to the rows of $X$. As $\gamma$ increases, the row centroids will begin to coalesce, and we likewise say that the $i$th and $j$th rows of $X$ belong to the same row partition if their centroid estimates $U_i$ and $U_j$ coincide.

When $J(U)$ includes both $\Omega W(U)$ and $\Omega W(U^T)$, rows and columns of $U$ are simultaneously shrunk towards each other as the parameter $\gamma$ increases. The penalized estimates exhibit the desired checkerboard structure as seen in Figure 1b. Note this shrinkage procedure is fundamentally different from methods like the clustered dendrogram, which independently cluster the rows and columns. By coupling row and column clustering, our formulation explicitly seeks out a solution with a checkerboard mean structure.

We now address choosing the weights $w_{ij}$ and $\tilde{w}_{ij}$. A judicious choice of the weights enables us to (i) employ a single regularization parameter $\gamma$, (ii) obtain more parsimonious clusterings, and (iii) speed up the convergence of key subroutines employed by COBRA. With these goals in mind, we recommend weights having following properties: (i) The column weights should sum to $1/\sqrt{n}$ and the row weights should sum to $1/\sqrt{p}$. (ii) The column weight $w_{ij}$ should be inversely proportional to the distance between the $i$th and $j$th columns $\|X_i - X_j\|_2$. The row weights should be assigned analogously. (iii) The weights should be sparse, namely consist mostly of zeros.
We now discuss the rationale behind our weight recommendations. The key to ensuring that a single tuning parameter suffices for identifying the checkerboard pattern, is keeping the two penalty terms $\Omega_W(U)$ and $\Omega_{\tilde{W}}(U^T)$ on the same scale. If this does not hold, then either column or row clusterings will dominate. Consequently, since the columns are in $\mathbb{R}^p$ and the rows are in $\mathbb{R}^n$, we choose the column weights to sum to $1/\sqrt{p}$ and the row weights to sum to $1/\sqrt{n}$. Using weights that are inversely proportional to the distances between points, more aggressively shrinks rows and columns that are more similar to each other, and less aggressively shrinks rows and columns that are less similar to each other. Finally, sparse weights expedite computation. COBRA solves a sequence of convex clustering problems.

The algorithm we employ for solving the convex clustering subproblem scales in storage and computational operations as $O(npq)$, where $q$ is the number of non-zero weights. Shrinking all pairs of columns (rows) and taking $q = n^2$ ($q = p^2$) not only increases computational costs but also typically produces inferior clustering to sparser ones as seen in Chi and Lange (2015). We employ the sparse Gaussian kernel weights described in Chi and Lange (2015), which satisfy the properties outlined above. Additional discussion on this choice of weights is given in Web Appendix A.

3. Properties of the Convex Formulation and COBRA’s Solution

The solution to minimizing (1) has several attractive properties as a function of the data $X$, the regularization parameter $\gamma$, and its weights $W = \{w_{ij}\}$ and $\tilde{W} = \{\tilde{w}_{ij}\}$, some of which can be exploited to expedite its numerical computation. We emphasize that these results are inherent to the minimization of the objective function (1). They hold regardless of the algorithm used to find the minimum point, since they are a consequence of casting the biclustering problem as a convex
program. Proofs of all propositions can be found in Web Appendix B. First, minimizing (1) is a well-posed optimization problem.

**Proposition 3.1:** The function $F_\gamma(U)$ defined in (1) has a unique global minimizer.

Furthermore, since $F_\gamma(U)$ is convex, the only local minimum is the global minimum, and any algorithm that converges to a stationary point of $F_\gamma(U)$ will converge to the global minimizer. The next result will have consequences for numerical computation and is also the foundation underpinning COBRA’s stability.

**Proposition 3.2:** The minimizer $U^*$ of (1) is jointly continuous in $(X, \gamma, W, \tilde{W})$.

Continuity of $U^*$ in the regularization parameter $\gamma$ suggests employing warm-starts, or using the solution at one $\gamma$ as the starting point for a problem with a slightly larger $\gamma$, because small changes in $\gamma$ will result in small changes in $U^*$. Continuity of $U^*$ in $X$ tells us that the solution varies smoothly with perturbations in the data, our main stability result. Recall that the $i$th and $j$th columns (rows) are assigned to the same cluster if $U^*_i = U^*_j$ ($U^*_i = U^*_j$). Since $U^*$ varies smoothly in the data, so do the differences $U^*_i - U^*_j$. While assigning entries to biclusters is an inherently discrete operation, we will see in Section 7 that this continuity result manifests itself in assignments that are robust to perturbations in the data.

Continuity of $U^*$ in the weights also give us stability guarantees and assurances that COBRA’s biclustering results should be locally insensitive to changes in the weights.

The parameter $\gamma$ tunes the complexity of COBRA’s solution. We next show that COBRA’s most complicated (small $\gamma$) and simplest (large $\gamma$) solutions coincide with the clustered dendrogram’s most complicated and simplest solutions. Clearly when $\gamma = 0$, the solution is just the data, namely $U^* = X$. To get some intuition
on how the solution behaves as $\gamma$ increases, observe that the penalty $J(U)$ is a semi-norm. Moreover, under suitable conditions on the weights, spelled out in Assumption 1 below, $J(U)$ is zero if and only if $U$ is a constant matrix.

**Assumption 1:** For any pair of columns (rows), indexed by $i$ and $j$ with $i < j$, there exists a sequence of indices $i \to k \to \cdots \to l \to j$ along which the weights, $w_{ik}, \ldots, w_{lj}$ ($\tilde{w}_{ik}, \ldots, \tilde{w}_{lj}$) are positive.

**Proposition 3.3:** Under Assumption 1, $J(U) = 0$ if and only if $U = c11^T$ for some $c \in \mathbb{R}$.

This result suggests that as $\gamma$ increases the solution to the biclustering problem converges to the solution of the following constrained optimization problem:

$$
\min_U \frac{1}{2} \|X - U\|_F^2 \quad \text{subject to } U = c11^T \text{ for some } c \in \mathbb{R},
$$

the solution to which is just the global mean $\bar{X}$, whose entries are all identically the average value of $X$ over all its entries. The next result formalizes our intuition that the centroids eventually coalesce to $\bar{X}$ as $\gamma$ becomes sufficiently large.

**Proposition 3.4:** Under Assumption 1, $F_\gamma(U)$ is minimized by the grand mean $\bar{X}$ for $\gamma$ sufficiently large.

Thus, as $\gamma$ increases from 0, the centroids matrix $U^*$ traces a continuous solution path that starts from $np$ biclusters, consisting of $u_{ij} = x_{ij}$, to a single bicluster, where $u_{ij} = (1/np) \sum_{i',j'} x_{i'j'}$ for all $i, j$.

### 4. Estimation of Biclusters with COBRA

Having characterized our estimator of the checkerboard means as the minimizer to (1), we now turn to the task of computing it. From here on, we fix the data $X$ and the weights $W$ and $\tilde{W}$ and consider the biclustering solution as a function of the
Convex Biclustering

11

parameter $\gamma$, denoting the solution as $U_\gamma$. The penalty term $J(U)$ in (1) makes
minimization challenging since it is non-smooth and not separable over any block
partitioning of $U$. Coordinate descent (Friedman et al., 2007; Wu and Lange, 2008)
is an effective solution when the non-smooth penalty term is separable over some
block partitioning of the variables, which is unfortunately not the case for (1).

Another popular iterative method for minimizing non-smooth convex functions
is the alternating direction method of multipliers (ADMM) (Boyd et al., 2011).

While an ADMM algorithm is feasible, we take a more direct approach with the
Dykstra-like proximal algorithm (DLPA) proposed by Bauschke and Combettes
(2008) because it yields a simple meta-algorithm that can take advantage of fast
solvers for the convex clustering problem.

DLPA generalizes a classic algorithm for fitting restricted least squares regression
problems (Dykstra, 1983) and solves minimization problems of the form

$$
\min_{U} \frac{1}{2}\|X - U\|_F^2 + f(U) + g(U),
$$

(2)

where $f$ and $g$ are lower-semicontinuous, convex functions. The biclustering prob-
lem is clearly an instance of (2). Setting $f = \gamma \Omega_W$ and $g = \gamma \Omega_{\tilde{W}}$ in (2) gives us
the pseudocode for COBRA shown in Algorithm 1. The operation $\text{prox}_{\gamma \Omega_W}(Z)$ is
the proximal mapping of the function $\gamma \Omega_W$ and is defined to be

$$
\text{prox}_{\gamma \Omega_W}(Z) = \arg \min_{V} \left[ \frac{1}{2} \|Z - V\|_F^2 + \gamma \Omega_W(V) \right].
$$

Each proximal mapping in Algorithm 1 corresponds to solving a convex clustering
problem.

The COBRA is very intuitive. The matrices $Y_m^T$ and $U_m$ are estimates of the
means matrix at the $m$th iteration. The matrices $P_m$ and $Q_m$ encode discrepancies
between these two estimates. We alternate between clustering the rows of the
Algorithm 1 Convex Biclustering Algorithm (COBRA)

Set $U_0 = X, P_0 = 0, Q_0 = 0$ for $m = 0, 1, \ldots$

```
repeat
    $Y_m = \text{prox}_{\gamma \Omega_w}(U_m^T + P_m^T)$  \quad \triangleright \text{Convex Clustering of Rows}
    P_{m+1} = U_m + P_m - Y_m^T$
    $U_{m+1} = \text{prox}_{\gamma \Omega_w}(Y_m^T + Q_m^T)$  \quad \triangleright \text{Convex Clustering of Columns}
    $Q_{m+1} = Y_m + Q_m - U_{m+1}$
until convergence
```

matrix $U_m + P_m$ and the columns of the matrix $Y_m + Q_m$. The following result guarantees that $Y_m^T$ and $U_m$ converge to the desired solution.

**Proposition 4.1:** The COBRA iterates $U_m$ and $Y_m^T$ in Algorithm 1 converge to the unique global minimizer of the convex biclustering objective (1).

Proposition 4.1 not only ensures the algorithmic convergence of Algorithm 1, but it also provides a natural stopping rule. We stop iterating once $\|U_m - Y_m^T\|_F$ falls below some tolerance $\tau > 0$. A proof of Proposition 4.1, as well as additional technical details and discussion on DLPA and COBRA, can be found in Web Appendix C.

The advantage of using DLPA is that COBRA is agnostic to the actual algorithm used to solve the proximal mapping. This is advantageous since we cannot analytically compute $\text{prox}_{\gamma \Omega_w}(Z)$. In this paper we use the alternating minimization algorithm (AMA) introduced in Chi and Lange (2015) to solve the convex clustering problem. The algorithm performs projected gradient ascent on the Lagrangian dual problem. Its main advantage is that it requires computational work and storage that is linear in the size of the data matrix $X$, when we use the sparse Gaussian kernel weights described in Web Appendix A. A second advantage is that hard clustering assignments are trivially obtained from variables employed in the splitting method. Nonetheless, the DLPA framework makes it trivial to swap in more efficient solvers that may become available in the future.
5. Model Selection

Estimating the number of clusters or biclusters in a data set is a major challenge. With many existing biclustering methods, this is further exacerbated by the many tuning parameters that must be selected and the fact that biclustering assignments do not always change smoothly with the number of biclusters. For example, the sparse SVD method (Lee et al., 2010) requires three tuning parameters: two parameters controlling the sparsity of the left and right singular vectors of the sparse SVD and one controlling its rank. Furthermore, selecting the number of biclusters and other tuning parameters can be a major computational burden for large data sets. For example, the sparse biclustering method (Tan and Witten, 2014) uses cross-validation to select the number of row and column partitions. This can be time consuming if a large range of possible number of row and column partitions are explored. In contrast, COBRA has one parameter $\gamma$, that controls both the number of biclusters and bicluster assignments; moreover, the number of biclusters and assignments varies smoothly with $\gamma$.

5.1 Hold-Out Validation

We present a simple but principled approach to selecting $\gamma$ in a data-driven manner by posing the model selection problem as another convex program. We randomly select a hold-out set of elements in the data matrix and assess the quality of a model $U_\gamma$ on how well it predicts the hold-out set. This idea was first proposed by Wold (1978) for model selection in principal component analysis and has been used more recently to select tuning parameters for matrix completion problems (Mazumder et al., 2010). Denote these index pairs $\Theta \subset \{1, \ldots, p\} \times \{1, \ldots, n\}$, and let $|\Theta|$ denote the cardinality of the set $\Theta$. We may select a relatively small fraction of the elements, say 10%, for validation, namely $|\Theta| \approx 0.1 \times np$. Denote
the projection operator onto the set of indices $\Theta$ by $P_\Theta(X)$. The $ij$th entry of 
$P_\Theta(X)$ is $x_{ij}$ if $(i, j) \in \Theta$ and is zero otherwise. We then solve the following convex 
optimization problem
\[
\min_U \tilde{F}_\gamma(U) := \frac{1}{2} \|P_{\Theta^c}(X) - P_{\Theta^c}(U)\|_F^2 + \gamma J(U)
\]
for a sequence of $\gamma \in \mathcal{G} = \{\gamma_1 = 0, \ldots, \gamma_{\text{max}}\}$. Recall that we denote the minimizer
of $\tilde{F}_\gamma(U)$ by $U_\gamma$. We choose the $\gamma$ that minimizes the prediction error over the
hold-out set $\Theta$, namely $\gamma^* = \arg \min_{\gamma \in \mathcal{G}} \|P_\Theta(X) - P_\Theta(U_\gamma)\|_F$.

5.2 Solving the Hold-Out Problem

The problem defined in (3) can be seen as a convex matrix completion problem. 
Algorithm 2 summarizes a simple procedure for reducing the problem of mini-
mizing $\tilde{F}_\gamma(U)$ to solving a sequence of the complete biclustering problems (1). 
The solution from the previous iteration is used to fill in the missing entries in 
the current iteration; COBRA is then applied to the complete data matrix. This
approach is identical to the soft-impute approach of Mazumder et al. (2010) for 
solving the matrix completion problem using a nuclear norm penalty instead of
our fusion penalty $J(U)$. The similarity is not a coincidence as both procedures
are instances of a majorization-minimization (MM) algorithm (Lange et al., 2000)
which apply the same majorization on the smooth quadratic term. We defer details
on this connection to Web Appendix D. Algorithm 2 has the following convergence
 guarantees for the imputation algorithm.

**Proposition 5.1:** The limit points of the sequence of iterates $U^{(m)}$ of Algorithm 2 are solutions to (3).
Algorithm 2 COBRA with missing data

1: Initialize $U^{(0)}$.
2: repeat
3:    $M \leftarrow \mathcal{P}_{\Theta^v}(X) + \mathcal{P}_{\Theta}(U^{(m)})$
4:    $U^{(m+1)} \leftarrow \text{COBRA}(M)$
5: until convergence

Thus, we have turned the model selection problem of selecting both the number of biclusters and bicluster assignments into a principled convex program with strong convergence guarantees.

6. Simulation Studies

We compare COBRA and two other biclustering methods that also assume an underlying checkerboard mean structure. The first is the clustered dendrogram; hard biclustering assignments are made for the clustered dendrogram using the widely used dynamic tree cutting algorithm (Langfelder et al., 2008) implemented in the package `dynamicTreeCut`. The second is the sparse biclustering method (Tan and Witten, 2014) implemented in the package `sparseBC` All parameters were selected according to methods in the R packages.

Assessing the quality of a clustering is almost as hard as the clustering problem itself, as evidenced by a plethora of quantitative measures for comparing how similar two clusterings are. In this paper, we use the following three measures: the Rand index (RI), the adjusted Rand index (ARI), and the variation of information (VI). We included the RI (Rand, 1971), since it is one of the most widely used criteria for comparing partitions; it maps a pair of partitions to a number between 0 and 1, where 1 indicates perfect agreement between two partitions. Despite its popularity, the RI has some limitations (See Web Appendix E). Consequently, we also use the ARI (Hubert and Arabie, 1985), which was engineered to address deficiencies in the RI. Like the RI, the ARI takes a maximum value of 1 if the
clustering solution is identical to the true structure and takes a value close to 0 if the clustering result is obtained from random partitioning. Finally, we compared clustering results using the VI (Meilă, 2007). Unlike the RI and ARI, the VI is a metric. Consequently, under the VI we can speak rigorously about a neighborhood around a given clustering. In fact, the nearest neighbors of a clustering $C$ under the VI metric are clusterings obtained by splitting or merging small clusters in $C$. While not as popular as the RI and ARI, the VI is perhaps the most appropriate for assessing two hierarchical clusterings. As a metric, the VI takes a minimum value of 0 when there is perfect agreement between two partitions. Definitions of these criteria are given in Web Appendix E.

To simulate data with a checkerboard partition mean pattern, we consider the partition of the data matrix as the set of biclusters induced by taking the cross-product of the row and column groups. For all experiments, we used a single fixed weight assignment rule (See Web Appendix A), therefore COBRA selects a single tuning parameter $\gamma$ by validation. We perform computations in serial on a multi-core computer with 24 3.3 GHz Intel Xeon processors and 189 GB of RAM. Note that run times may vary depending on input parameters chosen. For example, COBRA estimates can be computed to high accuracy at greater computation, and sparse biclustering can explore a wider range of candidate row and column clusters at greater computation. To be fair, we did not cherry pick these parameters but picked some reasonable values and used them throughout our numerical experiments. For example, in sparse biclustering when there are 8 true row clusters and 8 true column clusters, we set the range of row and column clusters to be 1 to 12.

Finally, we also considered two variants of COBRA that employ standard refinements on the Lasso: the adaptive Lasso (Zou, 2006) and the thresholded Lasso
Convex Biclustering (Meinshausen and Yu, 2009). These refinements address the well known issue that the Lasso tends to select too many variables. Thus, we anticipate that COBRA estimates may identify too many biclusters. Consequently, we also compared the performance of an adaptive COBRA and thresholded COBRA in our study. Details on these refinements are in Web Appendix G.

We compare COBRA, the clustered dendrogram, and the sparse biclustering (spBC) algorithm of Tan and Witten (2014) on their abilities to recover a checkerboard pattern. Again for the clustered dendrogram, row and column assignments are made with the dynamic tree cutting (DCT) method (Langfelder et al., 2008). We simulate a $200 \times 200$ data matrix with a checkerboard bicluster structure, where $x_{ij} \sim \text{iid } \mathcal{N}(\mu_{rc}, \sigma^2)$ and $\mu_{rc}$ took on one of 25 equally spaced values between -6 and 6, namely $\mu_{rc} \sim \text{Uniform}\{-6, -5.5, \ldots, 5.5, 6\}$. We consider a high signal-to-noise ratio (SNR) situation where the minimum difference among bicluster means (0.5) is comparable to the noise ($\sigma = 1.5$) and a low SNR one where the minimum difference is dominated by the noise ($\sigma = 3.0$). To assess the performance as the number of column and row clusters are varied, we generated data using 16, 32, and 64 biclusters, corresponding to 2, 4, and 8 row groups and 8 column groups, respectively. Since typical clusters will not be equal in size, rows and columns are assigned to each of the groups randomly according to a non-uniform distribution. The probability that a row is assigned to the $i$th group is inversely proportional to $i$. Columns are assigned analogously to groups.

We computed the RI, ARI, and VI between the true biclusters and those obtained by the COBRA variants, DCT, and spBC. Table 1 reports the average RI, ARI, and VI over 50 replicates as well as the average number of recovered biclusters $\hat{N}_b$ and run times; $N_b$ denotes the true number of biclusters. In the high SNR scenario, all methods do well across all measures. In the low SNR scenario, the COBRA
variants often perform nearly as well or better than the other two methods. While DCT is significantly faster than all other methods, it also performs the worst at recovering the true biclusters in the low SNR scenario. The run times for COBRA indicate that it is computationally competitive compared to alternative biclustering solutions.

In closing our discussion on simulations, we reiterate that COBRA is designed to recover checkerboard patterns. While checkerboard patterns feature prominently in a range of applications, we also acknowledge that they are not universal. Nonetheless, by examining both the estimated biclusters and bicluster means, COBRA can potentially identify the correct biclusters even when the checkerboard assumption is violated. We discuss how COBRA can accomplish this in more detail with a case study in Web Appendix F.

### 7. Application to Genomics

To illustrate COBRA in action on a real example, we revisit the lung cancer data studied by Lee et al. (2010). We have selected the 500 genes with the greatest variance from the original collection of 12,625 genes. Subjects belong to one of four subgroups; they are either normal subjects (Normal) or have been diagnosed with one of three types of cancers: pulmonary carcinoid tumors (Carcinoid), colon metastases (Colon), and small cell carcinoma (Small Cell).

We first illustrate how the solution $U_{\gamma}$ evolves as $\gamma$ varies. Figure 2 shows snap shots of the COBRA solution path of this data set, as the parameter $\gamma$ increases. The path captures the whole range of behavior between under-smoothed estimates of the mean structure (small $\gamma$), where each cell is assigned its own bicluster, to over-smoothed estimates (large $\gamma$), where all cells belong to a single bicluster. In
between these extremes, we see rows and columns “fusing” together as $\gamma$ increases. Thus we have visual confirmation that minimizing (1) over a range of $\gamma$, yields a convex formulation of the clustered dendrogram.

While generating the entire solution path enables us to visualize the hierarchical relationships between biclusterings for different $\gamma$, we may ultimately require a hard biclustering assignment. By applying the validation procedure described in Section 5, we arrive at the smoothed mean estimate shown previously in Figure 1b.

We next conduct a simulation experiment based on the lung cancer data to test the stability and reproducibility of biclustering methods, critical qualities for real scientific analysis. To expedite computation in these experiments, we restrict our attention to the 150 genes with the highest variance. We first apply the biclustering methods on the original data to obtain baseline biclusterings. We then add iid $N(0,\sigma^2)$, noise where $\sigma = 0.5, 1.0, 1.5$ to create a perturbed data set on which to apply the same set of methods. We compute the RI, ARI, and VI between the baseline clustering and the one obtained on the perturbed data. Table 2 shows the average RI, ARI, and VI of 50 replicates as well as run times. For all values of $\sigma$, we see that the two COBRA variants tend to produce the most stable and reproducible results. The fact that the ARI scores are poor for plain COBRA but the RI and VI scores are good indicate that COBRA tends to shrink the same sets of rows and columns together even if it fails to fuse them together consistently. Again the run time results indicate that COBRA is computationally competitive. For completeness, results from an identical stability study for methods that do not assume a checkerboard pattern can be found in Web Appendix H.

[Figure 2 about here.]

[Table 2 about here.]
8. Discussion

Our proposed method for biclustering, COBRA, can be considered a principled reformulation of the clustered dendrogram. Unlike the clustered dendrogram, COBRA returns a unique global minimizer of a goodness-of-fit criterion, but like the clustered dendrogram, COBRA is simple to interpret. COBRA also sports two key improvements over existing biclustering methods. First, it is more stable. COBRA biclustering assignments on perturbations of the data agree noticeably more frequently than those of existing biclustering algorithms. Second, it admits an effective and efficient model selection procedure for selecting the number of biclusters, that reduces the problem to solving a sequence of convex biclustering problems. The upshot of these two qualities is that COBRA produces results that are both simple to interpret and reproducible.

The simplicity of our means model is also its greatest weakness, since we consider only checkerboard patterns, namely we assign each observation to exactly one bicluster and do not consider overlapping biclusters (Cheng and Church, 2000; Lazzeroni and Owen, 2002; Shabalin et al., 2009). Nonetheless, while models that allow for overlapping biclusters might be more flexible, they are also harder to interpret.

While our simulation studies demonstrated the effectiveness of COBRA, there is room for improvement. We highlight an intriguing suggestion made during the review of this article. In many real-world applications there is no “true” fixed number of biclusters. Instead, the underlying latent structure may be a continuum of biclusters at different scales of row and column aggregation. Indeed, COBRA has the potential to estimate a multiscale model of the data. When the weights are uniform, all columns (rows) are averaged together. When the weights are positive only among nearest neighbors, only nearest neighboring columns (rows)
are averaged together. Thus, by tuning the weights, we can obtain smoothed estimates of the data at different scales of resolution.

The ability to smooth estimates at different scales suggests a connection to computational harmonic analysis. Indeed, Coifman and Gavish (2011) explore the biclustering problem through a wavelet representation of the data matrix. They also seek a representation that is smooth with respect to partitions of the row and column graphs that specify the similarity among the observations and features. A checkerboard mean structure at different scales can be obtained via operations in the wavelet domain, namely by thresholding wavelet coefficients corresponding to different scales. We are currently exploring how to adapt Coifman and Gavish (2011)’s strategy to solve a sequence of COBRA problems at different scales in order to recover a continuum of biclusters.

An R package, called cvxbiclustr, implementing COBRA is available on CRAN.

Supplementary Materials

Web Appendices references in Sections 2-7 are available with this paper at the Biometrics website on Wiley Online Library.

Acknowledgements

EC acknowledges support from CIA Postdoctoral Fellowship 2012-12062800003. GA acknowledges support from NSF DMS 1209017, 1264058, and 1317602. RB acknowledges support from ARO MURI W911NF-09-1-0383 and AFOSR grant FA9550-14-1-0088.

References


Wold, S. (1978). Cross-validatory estimation of the number of components in


Figure 1: Heatmaps of the expression of 500 genes (rows) across 56 subjects (columns). Figure 1a depicts the clustered dendrogram applied to the raw data; Figure 1b depicts COBRA smoothed data after reordering the columns and rows via the seriation package (Hahsler et al., 2008). Subjects belong to one of four subgroups: Normal (o), Carcinoid (x), Colon (*), and Small Cell (+). This figure appears in color in the electronic version of this article.
Figure 2: Snap shots of the COBRA solution path of the lung cancer data set, as the parameter $\gamma$ increases. The path captures the whole range of behavior between under-smoothed estimates of the mean structure (small $\gamma$), where each cell is assigned its own bicluster, to over-smoothed estimates (large $\gamma$), where all cells belong to a single bicluster. This figure appears in color in the electronic version of this article.
<table>
<thead>
<tr>
<th>$N_b$</th>
<th>$\sigma$</th>
<th>COBRA</th>
<th>COBRA (A)</th>
<th>COBRA (T)</th>
<th>DCT</th>
<th>spBC</th>
</tr>
</thead>
<tbody>
<tr>
<td>RI</td>
<td>16</td>
<td>1.5</td>
<td>0.993</td>
<td><strong>0.994</strong></td>
<td>0.993</td>
<td>0.952</td>
</tr>
<tr>
<td></td>
<td>32</td>
<td>1.5</td>
<td><strong>0.999</strong></td>
<td><strong>0.999</strong></td>
<td>0.993</td>
<td>0.993</td>
</tr>
<tr>
<td></td>
<td>64</td>
<td>1.5</td>
<td><strong>0.999</strong></td>
<td><strong>0.999</strong></td>
<td>0.999</td>
<td>0.999</td>
</tr>
<tr>
<td></td>
<td>16</td>
<td>3.0</td>
<td>0.971</td>
<td><strong>0.982</strong></td>
<td>0.959</td>
<td>0.944</td>
</tr>
<tr>
<td></td>
<td>32</td>
<td>3.0</td>
<td><strong>0.997</strong></td>
<td><strong>0.997</strong></td>
<td>0.996</td>
<td>0.990</td>
</tr>
<tr>
<td></td>
<td>64</td>
<td>3.0</td>
<td><strong>0.999</strong></td>
<td><strong>0.999</strong></td>
<td>0.999</td>
<td>0.999</td>
</tr>
<tr>
<td>ARI</td>
<td>16</td>
<td>1.5</td>
<td>0.952</td>
<td>0.924</td>
<td>0.950</td>
<td>0.713</td>
</tr>
<tr>
<td></td>
<td>32</td>
<td>1.5</td>
<td>0.995</td>
<td>0.978</td>
<td><strong>0.996</strong></td>
<td>0.916</td>
</tr>
<tr>
<td></td>
<td>64</td>
<td>1.5</td>
<td><strong>0.999</strong></td>
<td>0.996</td>
<td><strong>0.999</strong></td>
<td>0.981</td>
</tr>
<tr>
<td></td>
<td>16</td>
<td>3.0</td>
<td>0.798</td>
<td><strong>0.909</strong></td>
<td>0.741</td>
<td>0.449</td>
</tr>
<tr>
<td></td>
<td>32</td>
<td>3.0</td>
<td>0.958</td>
<td><strong>0.982</strong></td>
<td>0.945</td>
<td>0.844</td>
</tr>
<tr>
<td></td>
<td>64</td>
<td>3.0</td>
<td>0.992</td>
<td>0.992</td>
<td>0.985</td>
<td>0.979</td>
</tr>
<tr>
<td>VI</td>
<td>16</td>
<td>1.5</td>
<td><strong>0.117</strong></td>
<td>0.132</td>
<td><strong>0.117</strong></td>
<td>0.713</td>
</tr>
<tr>
<td></td>
<td>32</td>
<td>1.5</td>
<td>0.013</td>
<td>0.032</td>
<td><strong>0.009</strong></td>
<td>0.195</td>
</tr>
<tr>
<td></td>
<td>64</td>
<td>1.5</td>
<td><strong>0.001</strong></td>
<td>0.022</td>
<td><strong>0.001</strong></td>
<td>0.043</td>
</tr>
<tr>
<td></td>
<td>16</td>
<td>3.0</td>
<td>0.627</td>
<td>0.446</td>
<td>0.730</td>
<td>2.245</td>
</tr>
<tr>
<td></td>
<td>32</td>
<td>3.0</td>
<td><strong>0.097</strong></td>
<td>0.125</td>
<td>0.127</td>
<td>0.516</td>
</tr>
<tr>
<td></td>
<td>64</td>
<td>3.0</td>
<td><strong>0.020</strong></td>
<td>0.046</td>
<td>0.034</td>
<td>0.061</td>
</tr>
<tr>
<td>$\hat{N}_b$</td>
<td>16</td>
<td>1.5</td>
<td>18.8</td>
<td>25.3</td>
<td><strong>16.8</strong></td>
<td>10.7</td>
</tr>
<tr>
<td></td>
<td>32</td>
<td>1.5</td>
<td>32.9</td>
<td>38.1</td>
<td><strong>32.1</strong></td>
<td>28.8</td>
</tr>
<tr>
<td></td>
<td>64</td>
<td>1.5</td>
<td><strong>64.3</strong></td>
<td>69.7</td>
<td><strong>64.3</strong></td>
<td>62.5</td>
</tr>
<tr>
<td></td>
<td>16</td>
<td>3.0</td>
<td>43.7</td>
<td>46.3</td>
<td>30.3</td>
<td>54.1</td>
</tr>
<tr>
<td></td>
<td>32</td>
<td>3.0</td>
<td><strong>30.4</strong></td>
<td>44.2</td>
<td>29.9</td>
<td>44.7</td>
</tr>
<tr>
<td></td>
<td>64</td>
<td>3.0</td>
<td>65.1</td>
<td>77.1</td>
<td><strong>63.2</strong></td>
<td>65.9</td>
</tr>
<tr>
<td>time (sec)</td>
<td>16</td>
<td>1.5</td>
<td>26.83</td>
<td>50.36</td>
<td>27.30</td>
<td><strong>0.21</strong></td>
</tr>
<tr>
<td></td>
<td>32</td>
<td>1.5</td>
<td>30.07</td>
<td>57.49</td>
<td>30.49</td>
<td><strong>0.21</strong></td>
</tr>
<tr>
<td></td>
<td>64</td>
<td>1.5</td>
<td>29.93</td>
<td>58.54</td>
<td>30.35</td>
<td><strong>0.21</strong></td>
</tr>
<tr>
<td></td>
<td>16</td>
<td>3.0</td>
<td>35.55</td>
<td>67.56</td>
<td>36.02</td>
<td><strong>0.22</strong></td>
</tr>
<tr>
<td></td>
<td>32</td>
<td>3.0</td>
<td>33.99</td>
<td>66.06</td>
<td>34.44</td>
<td><strong>0.21</strong></td>
</tr>
<tr>
<td></td>
<td>64</td>
<td>3.0</td>
<td>33.09</td>
<td>65.10</td>
<td>33.50</td>
<td><strong>0.20</strong></td>
</tr>
</tbody>
</table>

Table 1: Checkerboard mean structure with iid $N(0, \sigma^2)$ noise: low-noise ($\sigma = 1.5$) and high-noise ($\sigma = 3.0$). COBRA (A) is the adaptive COBRA and COBRA (T) is the thresholded COBRA. Details on these two variants are in Web Appendix G.
<table>
<thead>
<tr>
<th>σ</th>
<th>COBRA</th>
<th>COBRA (A)</th>
<th>COBRA (T)</th>
<th>DCT</th>
<th>spBC</th>
</tr>
</thead>
<tbody>
<tr>
<td>RI</td>
<td>0.5</td>
<td>0.984</td>
<td><strong>0.992</strong></td>
<td>0.959</td>
<td>0.979</td>
</tr>
<tr>
<td></td>
<td>1.0</td>
<td>0.981</td>
<td><strong>0.990</strong></td>
<td>0.944</td>
<td>0.974</td>
</tr>
<tr>
<td></td>
<td>1.5</td>
<td>0.973</td>
<td><strong>0.989</strong></td>
<td>0.896</td>
<td>0.973</td>
</tr>
<tr>
<td>ARI</td>
<td>0.5</td>
<td>0.350</td>
<td>0.788</td>
<td><strong>0.813</strong></td>
<td>0.530</td>
</tr>
<tr>
<td></td>
<td>1.0</td>
<td>0.233</td>
<td>0.686</td>
<td><strong>0.766</strong></td>
<td>0.439</td>
</tr>
<tr>
<td></td>
<td>1.5</td>
<td>0.201</td>
<td><strong>0.667</strong></td>
<td>0.644</td>
<td>0.340</td>
</tr>
<tr>
<td>VI</td>
<td>0.5</td>
<td>1.924</td>
<td>0.882</td>
<td><strong>0.776</strong></td>
<td>2.120</td>
</tr>
<tr>
<td></td>
<td>1.0</td>
<td>2.380</td>
<td>1.276</td>
<td><strong>0.962</strong></td>
<td>2.769</td>
</tr>
<tr>
<td></td>
<td>1.5</td>
<td>2.721</td>
<td><strong>1.312</strong></td>
<td>1.320</td>
<td>3.505</td>
</tr>
<tr>
<td>time (sec)</td>
<td>0.5</td>
<td>15.44</td>
<td>23.46</td>
<td>15.65</td>
<td><strong>0.07</strong></td>
</tr>
<tr>
<td></td>
<td>1.0</td>
<td>25.21</td>
<td>34.51</td>
<td>25.53</td>
<td><strong>0.11</strong></td>
</tr>
<tr>
<td></td>
<td>1.5</td>
<td>18.18</td>
<td>26.43</td>
<td>18.50</td>
<td><strong>0.12</strong></td>
</tr>
</tbody>
</table>

Table 2: Stability and reproducibility of biclusterings in lung cancer microarray data. COBRA variants, the clustered dendrogram with dynamic tree cutting, and sparse Biclustering are applied to the lung cancer data to obtain baseline biclusterings. We then perturb the data by adding iid $N(0, \sigma^2)$ noise where $\sigma = 0.5$ (Small Pert.), 1.0 (Medium Pert.), 1.5 (Large Pert.).