mathematics

MDPI

Article

An Efficient Algorithm for Convex Biclustering

Jie Chen * and Joe Suzuki

Graduate School of Engineering Science, Osaka University, Osaka 560-0043, Japan; j-suzuki@sigmath.es.osaka-u.ac.jp

* Correspondence: chen@sigmath.es.osaka-u.ac.jp

Abstract: We consider biclustering that clusters both samples and features and propose efficient convex biclustering procedures. The convex biclustering algorithm (COBRA) procedure solves twice the standard convex clustering problem that contains a non-differentiable function optimization. We instead convert the original optimization problem to a differentiable one and improve another approach based on the augmented Lagrangian method (ALM). Our proposed method combines the basic procedures in the ALM with the accelerated gradient descent method (Nesterov's accelerated gradient method), which can attain $O(1/k^2)$ convergence rate. It only uses first-order gradient information, and the efficiency is not influenced by the tuning parameter λ so much. This advantage allows users to quickly iterate among the various tuning parameters λ and explore the resulting changes in the biclustering solutions. The numerical experiments demonstrate that our proposed method has high accuracy and is much faster than the currently known algorithms, even for large-scale problems.

Keywords: clustering; convex biclustering; optimization; gradient descent method

check for updates

Citation: Chen, J.; Suzuki, J. An Efficient Algorithm for Convex Biclustering. *Mathematics* **2021**, *9*, 3021. https://doi.org/10.3390/math9233021

Academic Editors: Marina Alexandra Pedro Andrade and Maria Alves Teodoro

Received: 18 October 2021
Accepted: 22 November 2021
Published: 25 November 2021

1. Introduction

By clustering, such as k-means clustering [1] and hierarchical clustering [2,3], we usually mean dividing N samples, each consisting of p covariate values, into several categories, where $N, p \geq 1$.

In this paper, we consider biclustering [4] that is an extended notion of clustering. In biclustering, we divide both $\{1, \ldots, N\}$ and $\{1, \ldots, p\}$ based on the data simultaneously. If we are given a data matrix in $\mathbb{R}^{N \times p}$, then the rows and columns within the shared group exhibit similar characteristics. For example, given a gene expression data matrix, with genes as columns and samples as rows, the biclustering detects the submatrices, which represent the cooperative behavior of a group of genes corresponding to a group of samples [5]. Figure 1 illustrates an intuitive difference between standard clustering and biclustering. In recent years, biclustering has become a ubiquitous data-mining technique with varied applications, such as text mining, recommendation system, and bioinformatics. A comprehensive survey of biclustering was given by [6–8].

However, as noted in [6], biclustering is an NP-hard problem. Thus, the results may vary significantly with different initializations. Moreover, some conventional biclustering models suffer from poor performance due to non-convexity, which may return local optimal solutions. In order to avoid such an inconvenience, Chi et al. [9] proposed convex biclustering by reformulating the problem to convex formulations and using the fused lasso [10] concept.

In convex clustering [9,11–13], given a data matrix $X \in \mathbb{R}^{N \times p}$ and $\lambda > 0$, we compute a matrix U of the same size as X. Let $X_{i\cdot}$ and $U_{i\cdot}$ be the i-th rows of X and U, let $X_{\cdot j}$ and $U_{\cdot j}$ be the the j-th columns of X and U, and let $\|X - U\|_F^2$ denote the square sums of the Np elements, and $\|U_{i\cdot} - U_{j\cdot}\|$, $\|U_{\cdot m} - U_{\cdot n}\|$ as the ℓ_2 norm of p, N-dimensional vectors, respectively. Convex clustering finds $U \in \mathbb{R}^{N \times p}$ that minimizes a weighted sum of $\|X - U\|_F^2$ and $\{\lambda \|U_{i\cdot} - U_{j\cdot}\|\}_{i \neq j}$ (it may be formulated as minimizing a weighted sum of $\|X-U\|_F^2$ and $\{\lambda\|U_{\cdot m}-U_{\cdot n}\|\}_{m\neq n})$. If $X_{i\cdot}$ and $X_{j\cdot}$ share $U_{i\cdot}=U_{j\cdot}$, then they are in the same group w.r.t. $\{1,\cdots,N\}$. On the other hand, convex biclustering finds $U\in\mathbb{R}^{N\times p}$ that minimizes a weighted sum of $\|X-U\|_F^2$ and $\{\lambda\|U_{i\cdot}-U_{j\cdot}\|\}_{i\neq j}$ and $\{\lambda\|U_{\cdot m}-U_{\cdot n}\|\}_{m\neq n}$.

The convex biclustering achieves checker-board-like biclusters by penalizing both the rows and columns of U. When λ (the tuning parameter for $\{\|U_{i\cdot}-U_{j\cdot}\|\}_{i\neq j}$ and $\{\|U_{\cdot m}-U_{\cdot n}\|\}_{m\neq n})$ is zero, each (i,j) occupies a unique bicluster $\{(i,j)\}$ and $x_{ij}=u_{ij}$ for $i=1,\ldots,N$ and $j=1,\ldots,p$ when $X=(x_{ij})$ and $U=(u_{ij})$. As λ increases, the bicluster begins to fuse. For sufficiently large λ, all (i,j) merge into one single bicluster $\{(i,j)|i=1,\ldots,N,j=1,\ldots,p\}$. The convex formulation guarantees a globally optimal solution and demonstrates superior performance to competing approaches. Chi et al. [9] claimed that the convex biclustering performs better than the dynamic tree-cutting algorithm [14] and sparse biclustering algorithm [15] in their experiments.

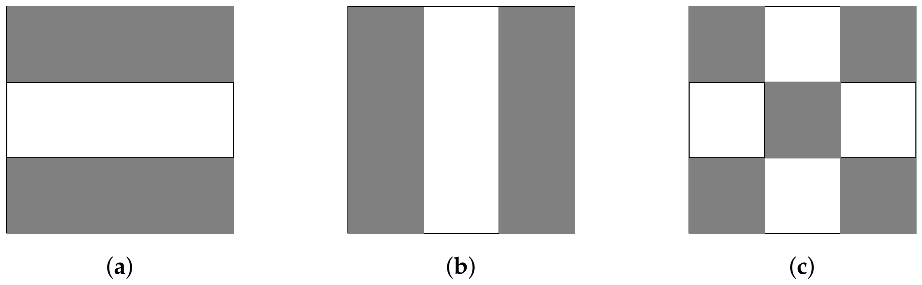


Figure 1. While standard clustering divides either rows or columns, the biclustering divides both. (**a**) Row clustering; (**b**) column clustering; (**c**) biclustering.

Nevertheless, despite these advantages, convex biclustering has not yet gained widespread popularity, due to its intensive computation. On the one hand, the main challenge of solving the optimization problem is the two fused penalty terms: indecomposable and non-differentiable. These properties increase the difficulty of solving. Many splitting methods for the indecomposable problem are complicated and create many subproblems to solve; techniques such as the subgradient method for the non-differentiable problem are slow to converge [16,17]. Moreover, it is difficult to find the optimal tuning parameter λ because we need to solve optimization problems with a sequence of parameters λ and select the one for the specific demand of researchers. Hence, we need to propose a fast way to solve the problems with the sequence of parameters λ. On the other hand, with the increased demands of biclustering techniques, convex biclustering is faced with large-scale data as the volume and complexity of data grows. Above all, it is necessary to propose an efficient algorithm to solve the convex biclustering problem.

There are limited algorithms for solving the problem in the literature. Chi et al. [9] proposed the convex biclustering algorithm (COBRA) using a Dykstra-like proximal algorithm [18] to solve the convex biclustering problem. Weylandt [19] proposed using alternating direction method of multipliers (ADMM) [20,21] and its variant, generalized ADMM [22], to solve the problem.

However, the COBRA yields subproblems, including the convex clustering problem, which requires expensive computations for large-scale problems due to the high per iteration cost [23,24]. Essentially, COBRA is a splitting method that separately solves a composite optimization problem containing three terms. Additionally, it is sensitive to tuning parameter λ. Therefore, obtaining the solutions under a wide range of parameters λ takes time, which is not feasible for broad applications and different demands for users. ADMM generally solves the problem by breaking it into smaller pieces and updating the variables alternately. Still, at the same time, it also introduces several subproblems which may cost much time. To be more specific, the ADMM proposed by Weylandt [19] requires solving the Sylvester equation in the step of updating the variable U. Hence, the Schur decomposition requires solving the Sylvester equation based on the numerical method from [25], which is complicated and time consuming. Additionally, it is known that ADMM exhibits $O(\frac{1}{k})$, where k is the number of iterations, convergence in general [26]. It often takes time to achieve relatively high precision [27], which is not feasible in some highly accurate applications. For example, the gene expression data contain huge information (the feature dimension usually exceeds 1000). However, COBRA and ADMM do not scale well for such large-scale problems. Overall, the above algorithm shows weak performance, which motivates us to combine some current algorithms to efficiently solve the convex biclustering problem like in reference [28].

This paper proposes an efficient algorithm with simple subproblems and a fast convergence rate to solve the convex biclustering problem. Rather than update each variable alternately, like ADMM, we use the augmented Lagrangian method (ALM) to update the primal variables simultaneously. In this way, we can transform the optimization problem to be differentiable, solve the problem via an efficient gradient descent method and further simplify the subproblems. Our proposed method is motivated by the work [29] in which the authors presented a way to convert the augmented Lagrangian function to a composite optimization that can be solved by the proximal gradient method [30]. Using the process twice to handle the two fused penalties $\{\lambda\|U_{i\cdot} - U_{j\cdot}\|\}_{i\neq j}$ and $\{\lambda\|U_{\cdot m} - U_{\cdot n}\|\}_{m\neq n}$, we obtain a differentiable problem from the augmented Lagrangian function. Then, we propose Nesterov's accelerated gradient method to solve the differentiable problem, which has $O(\frac{1}{k^2})$ global convergence rate.

Our main contributions are as follows:

- We propose an efficient algorithm to solve the convex biclustering model for large-scale N and p. The algorithm is a first-order method with simple subproblems. It only requires calculating the matrix multiplications and simple proximal operators, while the ADMM approaches require matrix inversion.
- Our proposed method does not require as much computation, even when the tuning parameter λ is large, as the existing approaches do, which means that it is easier to obtain biclustering results simultaneously for several λ values.

The remaining parts of this paper are as follows. In Section 2, we provide some preliminaries which are used in the paper and introduce the convex biclustering problem. In Section 3, we illustrate our proposed algorithm for solving the convex biclustering model. After that, we conduct numerical experiments to evaluate the performance of our algorithm in Section 4.

Notation: In this paper, we use $||x||_p$ to denote the ℓ_p norm of a vector $x \in \mathbb{R}^d$, $||x||_p := (\sum_{i=1}^{d} |x_i|^p)^{\frac{1}{p}}$ for $p \in [1, \infty)$, and $||x||_\infty := \max_i |x_i|$. For a matrix $X \in \mathbb{R}^{p\times q}$, we use $||X||_F$ to denote the Frobenius norm, $||X||_2$ denotes the spectral norm, and $||X||_1 := \sum_{i=1}^{p}\sum_{j=1}^{q} |x_{ij}|$ if not specified.

2. Preliminaries

In this section, we provide the background for understanding the proposed method in Section 3. In particular, we introduce the notions of ADMM, ALM, NAGM, and convex biclustering.

We say that differentiable $f : \mathbb{R}^n \to \mathbb{R}$ has a *Lipschitz-continuous gradient* if there exists $L > 0$ (Lipschitz constant) such that

$$||\nabla f(x) - \nabla f(y)||_2 \leq L||x - y||_2, \forall x, y \in \mathbb{R}^n. \tag{1}$$

We define the *conjugate* of a function $f : \mathbb{R}^n \to \mathbb{R}$ by

$$f^*(y) = \sup_{x\in \mathrm{dom} f} (y^T x - f(x)),$$

where dom $f \subseteq \mathbb{R}^n$ is the domain of f, and know that f^* is closed ($\{x \in \mathrm{dom}(f^*)|f^*(x) \leq \alpha\}$ is a closed set for any $\alpha \in \mathbb{R}$) and convex. It is known that $(f^*)^* = f$ when f is closed and convex, and that Moreau's decomposition [31] is available. Let $f : \mathbb{R}^n \to \mathbb{R}$ be closed and convex. For any $x \in \mathbb{R}^n$ and $\gamma > 0$, we have

$$prox_{\gamma f}(x) + \gamma prox_{\gamma^{-1} f^*}(\gamma^{-1} x) = x. \tag{2}$$

where $\text{prox}_f : \mathbb{R}^n \to \mathbb{R}^n$ is the proximal operator defined by

$$\text{prox}_f(x) := \arg\min_{y \in \mathbb{R}^n} \{ \frac{1}{2} ||x - y||_2^2 + f(y) \} . \tag{3}$$

The relation (2) is derived from $(f^*)^* = f$ and the definition (3) [31].

2.1. ADMM and ALM

In this subsection, we introduce the general optimization procedures of ADMM and ALM.

Let $f, g : \mathbb{R}^n \to \mathbb{R}$ and $h : \mathbb{R}^p \to \mathbb{R}$ be convex. Assume that f is differentiable, and g and h are not necessarily differentiable. We consider the following optimization problem:

$$\begin{aligned} &\min_{x,y} f(x) + h(y) + g(x) \\ &\text{subject to } Ax = y, \end{aligned} \tag{4}$$

with variables $x \in \mathbb{R}^n$ and $y \in \mathbb{R}^p$, and matrix $A \in \mathbb{R}^{p \times n}$. To this end, we define the augmented Lagrangian function as the following,

$$L_\nu(x, y, \lambda) := f(x) + g(x) + h(y) + \langle \lambda, Ax - y \rangle + \frac{\nu}{2} ||Ax - y||_2^2 , \tag{5}$$

where $\nu > 0$ is an augmented Lagrangian parameter, and $\lambda \in \mathbb{R}^p$ is the Lagrangian multipliers.

ADMM is a general procedure to find the solution to the problem (4) by iterating

$$\begin{aligned} x^{k+1} &:= \arg\min_x L_\nu(x, y^k, \lambda^k), \\ y^{k+1} &:= \arg\min_y L_\nu(x^{k+1}, y, \lambda^k), \\ \lambda^{k+1} &:= \lambda^k + \nu(Ax^{k+1} - y^{k+1}), \end{aligned}$$

given the initial values y^1 and λ^1.

What we mean by the ALM [32–34] is to minimize the augmented Lagrangian function (5) w.r.t. variables x and y simultaneously given a λ value, i.e., we iterate the following steps:

$$(x^{k+1}, y^{k+1}) := \arg\min_{x,y} L_\nu(x, y, \lambda^k), \tag{6}$$

$$\lambda^{k+1} := \lambda^k + \nu(Ax^{k+1} - y^{k+1}) . \tag{7}$$

Shimmura and Suzuki [29] considered the minimization of the function $\phi : \mathbb{R}^n \to \mathbb{R}$,

$$\phi(x) := f(x) + \min_y \{ h(y) + \langle \lambda, Ax - y \rangle + \frac{\nu}{2} ||Ax - y||_2^2 \},$$

and the non-differentiable function $g(x)$ over $x \in \mathbb{R}^n$, replacing the minimization over $x \in \mathbb{R}^n$ and $y \in \mathbb{R}^n$ in (6).

Lemma 1 ([29], Theorem 1)**.** *The function $\phi(x)$ is differentiable and its differential is*

$$\nabla \phi(x) = \nabla f(x) + A^T(prox_{\nu h^*}(\nu Ax + \lambda)).$$

By Lemma 1, the minimization in (6) can be regarded as the composite optimization problem with the differentiable function $\phi(x)$ and the non-differentiable function $g(x)$. Therefore, it is feasible to use the proximal gradient method to update the variable x, such as the fast iterative shrinkage-thresholding algorithm (FISTA) [30].

2.2. Nesterov's Accelerated Gradient Method

Nesterov [35] proposed a variant of the gradient descent method for Lipschitz differentiable functions. It has $O(\frac{1}{k^2})$ convergence rate while the (traditional) gradient descent method has $O(\frac{1}{k})$ [35]. Considering the minimization of a convex and differentiable function $F(x)$, NAGM is described in Algorithm 1 when $\nabla F(x)$ has a Lipschitz constant L in (1).

Algorithm 1 NAGM.

```
Input: Lipschitz constant $L$, initial value $x^0 = y^0$, $t^1 = 1$.
While $k < k_{\max}$ (until convergence) do
  1: $x^{k+1} = y^k - \frac{1}{L}\nabla F(y^k)$
  2: $t^{k+1} = \frac{1+\sqrt{4{t^k}^2+1}}{2}$
  3: $y^{k+1} = x^{k+1} + \frac{t^k-1}{t^{k+1}}(x^{k+1} - x^k)$
  4: $k = k+1$
End while
```

Algorithm 1 replaces the gradient descent $y^{k+1} = y^k - \frac{1}{L}\nabla F(y^k)$ by Steps 1 to 3: Step 1 executes the gradient descent to obtain x^{k+1} from y^k, Steps 2 and 3 calculate new y^{k+1} based on the previous x^k, x^{k+1}, and then return to the gradient descent in Step 1. NAGM assumes that F is differentiable, while FISTA [30], an accelerated version of ISTA [30], deals with non-differentiable F using the proximal gradient descent (see Table 1).

Table 1. Gradient descent and its modifications .

	Differentiable	**Non-Differentiable**
Ordinary	Gradient descent	Proximal gradient descent (e.g., ISTA [30])
Accelated	NAGM [35]	FISTA [30]

2.3. Convex Biclustering

We consider the convex biclustering problem in a general setting. Suppose we have a data matrix $X = (x_{ij})$ consisting of N observations, $i = 1, \ldots, N$, w.r.t. p features $j = 1, \ldots, p$. Our task is to assign each observation to one of the non-overlapped row clusters $C_1, \cdots, C_R \subseteq \{1, \ldots, N\}$ and assign each feature to one of the non-overlapped column clusters $D_1, \cdots, D_K \subseteq \{1, \ldots, p\}$. We assume that the clusters $C_1, \cdots, C_R$ and $D_1, \cdots, D_K$ and the values of R and K are not known a priori.

More precisely, the convex biclustering in this paper is formulated as follows:

$$\min_{U\in\mathbb{R}^{N\times p}} \frac{1}{2}||X-U||_F^2 + \lambda\left(\sum_{i=1}^{N-1}\sum_{j=i+1}^{N}\omega_{ij}||U_{i\cdot} - U_{j\cdot}||_2 + \sum_{m=1}^{p-1}\sum_{n=m+1}^{p}\tilde{\omega}_{mn}||U_{\cdot m} - U_{\cdot n}||_2\right), \quad (8)$$

where $U_{i\cdot}$ and $U_{\cdot j}$ are the i-th row and j-th column of $U \in \mathbb{R}^{N\times p}$. Chi et al. [9] suggested a requirement on the weights selection:

$$\omega_{ij} := 1_{i,j}^k \exp\left(-\phi\|x_{i\cdot} - x_{j\cdot}\|_2^2\right)$$

and

$$\tilde{\omega}_{mn} := 1_{m,n}^k \exp\left(-\tilde{\phi}\|x_{\cdot m} - x_{\cdot b}\|_2^2\right),$$

where $1^k_{i,j}$ is 1 if j belongs to the i's k-nearest neighbors and 0 otherwise. $1^k_{m,n}$ is defined similarly (the parameter k should be specified beforehand), and $x_{i\cdot}$ and $x_{\cdot j}$ are the i-th row and j-th column of the matrix X. They suggested that the constants ϕ and $\tilde{\phi}$ are determined so that the sums $\sum_{i=1}^{N-1}\sum_{j=i+1}^{N}\omega_{ij}$ and $\sum_{m=1}^{p-1}\sum_{n=m+1}^{p}\tilde{\omega}_{mn}$ are $N^{-1/2}$ and $p^{-1/2}$, respectively.

Chi et al. [9] proposed COBRA to solve the problem (8). Essentially, COBRA solves the standard convex clustering problems of rows and columns alternately,

$$\min_{U\in\mathbb{R}^{N\times p}} \frac{1}{2}||X-U||_F^2 + \lambda \sum_{i=1}^{N-1}\sum_{j=i+1}^{N} \omega_{ij}||U_{i\cdot}-U_{j\cdot}||_2 \tag{9}$$

and

$$\min_{U\in\mathbb{R}^{N\times p}} \frac{1}{2}||X-U||_F^2 + \lambda \sum_{m=1}^{p-1}\sum_{n=m+1}^{p} \tilde{\omega}_{mn}||U_{\cdot m}-U_{\cdot n}||_2, \tag{10}$$

until the solution converges. However, both optimization problems contain a non-differentiable ℓ_2 norm term, and solving the convex clustering problems (9) and (10) take much time [23]. Later, the ADMM-based approaches considered alternating variables procedures and outperformed the COBRA for large parameter λ [19].

3. The Proposed Method and Theoretical Analysis

In this section, we first show that the whole terms in the augmented Lagrangian function of (8) can be differentiable w.r.t. U after introducing two dual variables. Therefore, we use NAGM rather than FISTA in [29], where assumes the objective function contains a non-differentiable term.

In order to make the notation clear, we formulate the problem (8) in another way. Let ϵ_1 and ϵ_2 be the sets $\{(i,j)|\omega_{ij}>0, i<j\}$ and $\{(m,n)|\tilde{\omega}_{mn}>0, m<n\}$, respectively, and denote the cardinality of a set S by $|S|$. We define the matrices $C\in\mathbb{R}^{|\epsilon_1|\times n}$ and $D\in\mathbb{R}^{p\times|\epsilon_2|}$ by

$$C_{l,i}=1,\ C_{l,j}=-1,\ C_{l,k}=0,\ k\neq i,j \Longleftrightarrow l=(i,j)\in\epsilon_1,$$

$$C=\begin{bmatrix}1 & -1 & 0 & \cdots & 0 & 0\\ 1 & 0 & -1 & \cdots & 0 & 0\\ \vdots & & & \ddots & & \vdots\\ 0 & 0 & 0 & \cdots & 1 & -1\end{bmatrix}_{|\epsilon_1|\times n},$$

and

$$D_{m,l}=1,\ D_{n,l}=-1,\ D_{k,l}=0,\ k\neq m,n \Longleftrightarrow l=(m,n)\in\epsilon_2\ ,$$

$$D=\begin{bmatrix}1 & 1 & \cdots & 0\\ -1 & 0 & \cdots & 0\\ 0 & -1 & \cdots & 0\\ \vdots & & \ddots & \vdots\\ 0 & 0 & \cdots & 1\\ 0 & 0 & \cdots & -1\end{bmatrix}_{p\times|\epsilon_2|},$$

respectively.

Then, the optimization problem (8) can be reformulated as follows:

$$\min_{U\in\mathbb{R}^{N\times p}} \frac{1}{2}||X-U||_F^2 + \lambda\left(\sum_{l\in\epsilon_1}\omega_l||C_{l,\cdot}U||_2 + \sum_{l\in\epsilon_2}\tilde{\omega}_l||UD_{\cdot,l}||_2\right). \tag{11}$$

3.1. The ALM Formulation

To implement ALM, we further construct the problem (11) into the following constrained optimization problem by introducing the dual variables $V \in \mathbb{R}^{|\epsilon_1| \times p}$ and $Z \in \mathbb{R}^{N \times |\epsilon_2|}$,

$$\begin{aligned} \min_{U,V,Z} \quad & \frac{1}{2}||X - U||_F^2 + \lambda\left(\sum_{l \in \epsilon_1} \omega_l ||V_l||_2 + \sum_{l \in \epsilon_2} \tilde{\omega}_l ||Z_l||_2\right) \\ \text{subject to} \quad & C_{l,\cdot}U - V_l = 0, \quad \forall l \in \epsilon_1, \\ & UD_{\cdot,l} - Z_l = 0, \quad \forall l \in \epsilon_2\,, \end{aligned} \tag{12}$$

where V_l and Z_l are the l-th row and l-th column of V and Z, respectively. If we introduce the following functions,

$$\begin{aligned} f(U) :&= \frac{1}{2}||X - U||_F^2 \\ h(V) :&= \lambda \sum_{l \in \epsilon_1} \omega_l ||V_l||_2 \\ g(Z) :&= \lambda \sum_{l \in \epsilon_2} \tilde{\omega}_l ||Z_l||_2\,, \end{aligned}$$

then the problem in (12) becomes

$$\min_{U,V,Z} \{f(U) + h(V) + g(Z)\}. \tag{13}$$

The augmented Lagrangian function of the problem (12) is given by

$$\begin{aligned} L_\nu(U, V, Z, \Lambda_1, \Lambda_2) :=& f(U) + h(V) + \sum_{l \in \epsilon_1} \langle \Lambda_{1l}, C_{l,\cdot}U - V_l \rangle + \frac{\nu}{2} \sum_{l \in \epsilon_1} ||C_{l,\cdot}U - V_l||_2^2 \\ &+ g(Z) + \sum_{l \in \epsilon_2} \langle \Lambda_{2l}, UD_{\cdot,l} - Z_l \rangle + \frac{\nu}{2} \sum_{l \in \epsilon_2} ||UD_{\cdot,l} - Z_l||_2^2, \end{aligned} \tag{14}$$

where $\nu > 0$ is an augmented Lagrangian penalty, $\Lambda_1 \in \mathbb{R}^{|\epsilon_1| \times p}$ and $\Lambda_2 \in \mathbb{R}^{N \times |\epsilon_2|}$ are Lagrangian multipliers, and Λ_{1l} and Λ_{2l} are the l-th row and l-th column of Λ_1 and Λ_2, respectively.

Hence, the ALM procedure of the problem (12) consists of the following three steps:

$$(U^k, V^k, Z^k) = \arg\min_{U,V,Z} L_\nu(U, V, Z, \Lambda_1^{k-1}, \Lambda_2^{k-1}), \tag{15}$$

$$\Lambda_{1l}^k = \Lambda_{1l}^{k-1} + \nu(C_{l,\cdot}U^k - V_l^k), \quad \forall l \in \epsilon_1, \tag{16}$$

$$\Lambda_{2l}^k = \Lambda_{2l}^{k-1} + \nu(C_{l,\cdot}U^k - V_l^k), \quad \forall l \in \epsilon_2. \tag{17}$$

3.2. The Proposed Method

We construct our proposed method: repeatedly minimizing the augmented Lagrangian function in Equation (15) w.r.t U, V_l, Z_l and updating the Lagrange multipliers in Equations (16) and (17). The whole procedure is summarized in Algorithm 2.

Algorithm 2 Proposed method.

```
Input: Data $X$, matrices $C$ and $D$, Lipschitz constant $L$ calculated by (27), penalties $\lambda$ and $\nu$, initial value $\Lambda_1^0, \Lambda_2^0, Y^1, t^1 = 1$.
While $k < k_{\max}$ (until convergence) do
  1: Calculate the gradient $\nabla F(Y^k)$ by (20).
  2: Update iterate: $U^k \leftarrow Y^k - \frac{1}{L}\nabla F(Y^k)$.
  3: Update iterate: $\Lambda_{1l}^k \leftarrow P_{\mathcal{B}_l}(\Lambda_{1l}^{k-1} + \nu C_{l,\cdot}U^k)$, for $l \in \epsilon_1$, by (24) and (25), where $\mathcal{B}_l := \{y : ||y||_2 \le \lambda\omega_l\}$.
  4: Update iterate: $\Lambda_{2l}^k \leftarrow P_{\tilde{\mathcal{B}}_l}(\Lambda_{2l}^{k-1} + \nu U^k D_{\cdot,l})$, for $l \in \epsilon_2$, by (26) and (25), where $\tilde{\mathcal{B}}_l := \{\tilde{y} : ||\tilde{y}||_2 \le \lambda\tilde{\omega}_l\}$.
  5: $t^{k+1} = \frac{1+\sqrt{1+4t^{k^2}}}{2}$
  6: $Y^{k+1} = U^k + \frac{t^k-1}{t^{k+1}}(U^k - U^{k-1})$
  7: $k = k+1$
Output: optimal solution to problem (8), $U^* = U^k$.
```

Step 1: update $\boldsymbol{U}$. In the U-update, if we define the following function in Equation (14),

$$\begin{aligned} F(U) :&= f(U) + \min_{V,Z}\{L(U,V,Z,\Lambda_1,\Lambda_2)\} \\ &= f(U) + \min_{V,Z}\Bigg\{h(V) + \sum_{l\in\epsilon_1}\langle\Lambda_{1l}, C_{l,\cdot}U - V_l\rangle + \frac{\nu}{2}\sum_{l\in\epsilon_1}||C_{l,\cdot}U - V_l||_2^2 \\ &\quad + g(Z) + \sum_{l\in\epsilon_2}\langle\Lambda_{2l}, UD_{\cdot,l} - Z_l\rangle + \frac{\nu}{2}\sum_{l\in\epsilon_2}||UD_{\cdot,l} - Z_l||_2^2\Bigg\}, \end{aligned} \tag{18}$$

then the update of U in (15) can be written as

$$U^{k+1} := \arg\min_U F(U^k). \tag{19}$$

We find that (18) is differentiable due to Lemma 1, and obtain the following proposition.

Proposition 1. *The function $F(U)$ is differentiable with respect to U, and*

$$\nabla_U F(U) = -X + U + C^T(prox_{\nu h^*}(\nu CU + \Lambda_1)) + \Big(prox_{\nu g^*}(\nu UD + \Lambda_2)\Big)D^T. \tag{20}$$

For the proof, see the Appendix A.1.

With Proposition 1, we can use NAGM (Algorithm 1) to update U by solving the differentiable optimization problem (19).

Step 2: update $\boldsymbol{V_l}$ and $\boldsymbol{\Lambda_1}$. By step (15) in the ALM procedure, we must minimize the functions in Equation (18) corresponding to the vector V_l by updating the following,

$$\begin{aligned} V_l^k &= \arg\min_{V_l}\Bigg\{\lambda\omega_l||V_l||_2 + \frac{\nu}{2}||V_l||_2^2 - \sum_{l\in\epsilon_1}\langle\Lambda_{1l}^{k-1} + \nu C_{l,\cdot}U^k, V_l\rangle\Bigg\} \\ &= \arg\min_{V_l}\Big\{\frac{\nu}{2}||V_l - (C_{l,\cdot}U^k + \nu^{-1}\Lambda_{1l}^k)||_2^2 + h_l(V_l)\Big\} \\ &= \text{prox}_{h_l/\nu}(C_{l,\cdot}U^k + \nu^{-1}\Lambda_{1l}^{k-1}), \end{aligned}$$

where $h_l(V_l) := \lambda\omega_l||V_l||_2$ denotes the l-th term in $h(V)$.

We substitute the optimal V_l^k in the k-th iteration into step (16)

$$\Lambda_{1l}^k \leftarrow \Lambda_{1l}^{k-1} + \nu(C_{l,\cdot}U^k - V_l^k), \quad \forall l \in \epsilon_1,$$

to obtain

$$\Lambda_{1l}^{k} \leftarrow \Lambda_{1l}^{k-1} + \nu C_{l,\cdot} U^{k} - \nu \text{prox}_{h_l/\nu}(C_{l,\cdot} U^{k} + \nu^{-1}\Lambda_{1l}^{k-1}). \tag{21}$$

By Moreau's decomposition (2), we further simplify the update (21) as follows,

$$\Lambda_{1l}^{k} \leftarrow \text{prox}_{\nu h_l^*}(\Lambda_{1l}^{k-1} + \nu C_{l,\cdot} U^{k}), \tag{22}$$

which means the updates of V_l and Λ_{1l} become one update (22). Hence, there is no longer a need to store and compute the variable V_l in the ALM updates, which reduces computational costs.

In the update (22), the conjugate function $h_l^*(y)$ of the ℓ_2 norm is an indicator function ([36], Example 3.26):

$$h_l^*(y) = \begin{cases} 0, & \text{if } ||y||_2 \le \lambda\omega_l, \\ \infty, & \text{otherwise.} \end{cases} \tag{23}$$

Moreover, the proximal operator of the indicator function (22) is the projection problem ([37], Theorem 6.24):

$$\text{prox}_{\nu h_l^*}(\nu C_{l,\cdot} U^{k} + \Lambda_{1l}^{k-1}) = P_{\mathcal{B}_l}(\nu C_{l,\cdot} U^{k} + \Lambda_{1l}^{k-1})\,, \tag{24}$$

where $\mathcal{B}_l := \{y : ||y||_2 \le \lambda\omega_l\}$, and the operator $P_{\mathcal{B}_l}$ denotes the projection onto the ball $\mathcal{B}_l$. It solves the problem $P_{\mathcal{B}_l}(x) := \arg\min_{u\in\mathcal{B}_l} ||u - x||_2^2$, i.e.,

$$P_{\mathcal{B}_l}(x) = \begin{cases} x, & \text{if } ||x||_2 \le \lambda\omega_l, \\ \lambda\omega_l, & \text{otherwise.} \end{cases} \tag{25}$$

This projection problem completes in $O(p)$ operations for a p-dimensional vector $x \in \mathbb{R}^p$.

Step 3: update Z and Λ_2. Similarly, we can derive the following equations:

$$\begin{aligned} Z_l^{k+1} &= \arg\min_{Z_l}\left\{\lambda\tilde{\omega}_l||Z_l||_2 + \frac{\nu}{2}||Z_l||_2^2 - \sum_{l\in\epsilon_2}\langle\Lambda_{2l}^{k-1} + \nu U^k D_{\cdot,l}, Z_l\rangle\right\} \\ &= \arg\min_{Z_l}\left\{\frac{\nu}{2}||Z_l - (U^k D_{\cdot,l} + \nu^{-1}\Lambda_{2l}^{k-1})||_2^2 + g_l(Z_l)\right\} \\ &= \text{prox}_{g_l/\nu}(U^k D_{\cdot,l} + \nu^{-1}\Lambda_{2l}^{k-1}) \end{aligned}$$

where $g_l(Z_l) := \lambda\tilde{\omega}_l||Z_l||_2$. Then, the dual variable Λ_2 update becomes

$$\Lambda_{2l}^{k} \leftarrow \text{prox}_{\nu g_l^*}(\Lambda_{2l}^{k-1} + \nu U^k D_{\cdot,l}), \quad \text{for } l \in \epsilon_2.$$

If we write in the projection operator, then it becomes

$$\Lambda_{2l}^{k} \leftarrow P_{\tilde{\mathcal{B}}_l}(\Lambda_{2l}^{k-1} + \nu U^k D_{\cdot,l}) \tag{26}$$

where $\tilde{\mathcal{B}}_l := \{\tilde{y} : ||\tilde{y}||_2 \le \lambda\tilde{\omega}_l\}$.

Our proposed method only uses first-order information. Furthermore, we just need to calculate the gradient of the function F and proximal operators in each iteration, where the proximal operators are easy to obtain by solving the projection problem.

3.3. Lipschitz Constant and Convergence Rate

By the following lemma, we know that if we choose $\frac{1}{L}$ as the step size for each iteration in the NAGM, then the convergence rate is, at most, $O(\frac{1}{k^2})$.

Lemma 2 ([30,35,38])**.** *Let $\{U^k\}$ as the sequence generated by Algorithm 1, and U_0 as an initial value. If we take the step size as $\frac{1}{L}$, then for any $k \geq 1$ we have*

$$F(U^k) - F(U^*) \leq \frac{2L||U_0 - U^*||_F^2}{(k+1)^2} .$$

In order to examine the performance of the proposed method, we derive the Lipschitz constant L of $\nabla_U F(U)$ as in the following proposition.

Proposition 2. *The Lipschitz constant of $\nabla_U F(U)$ is upperbounded by*

$$1 + \nu\lambda_{\max}(C^T C) + \nu\lambda_{\max}(D^T D) , \tag{27}$$

where $\lambda_{\max}$ denotes the maximum eigenvalue of the corresponding matrix.

Proof. By definition in (1) and Proposition 1, we derive the Lipschitz constant as follows,

$$\begin{aligned}||\nabla_U F(U_1) - \nabla_U F(U_2)||_2 &= ||U_1 - U_2 + C^T(\mathrm{prox}_{\nu h^*}(\nu C U_1 + \Lambda_1)) - C^T(\mathrm{prox}_{\nu h^*}(\nu C U_2 + \Lambda_1)) \\ &\quad + \Big(\mathrm{prox}_{\nu g^*}(\nu U_1 D + \Lambda_2)\Big)D^T - \Big(\mathrm{prox}_{\nu g^*}(\nu U_2 D + \Lambda_2)\Big)D^T||_2 \\ &\leq ||U_1 - U_2||_2 + ||C^T(\mathrm{prox}_{\nu h^*}(\nu C U_1 + \Lambda_1)) - C^T(\mathrm{prox}_{\nu h^*}(\nu C U_2 + \Lambda_1))||_2 \\ &\quad + ||\Big(\mathrm{prox}_{\nu g^*}(\nu U_1 D + \Lambda_2)\Big)D^T - \Big(\mathrm{prox}_{\nu g^*}(\nu U_2 D + \Lambda_2)\Big)D^T||_2.\end{aligned}$$

By the definition of matrix 2-norm and the nonexpansiveness of the proximal operators ([39], Lemma 2.4), we obtain

$$\begin{aligned}||\nabla_U F(U_1) - \nabla_U F(U_2)||_2 &\leq ||U_1 - U_2||_2 + \sqrt{\lambda_{\max}(C^T C)}||\nu C U_1 - \nu C U_2||_2 + ||\nu U_1 D - \nu U_2 D||_2\sqrt{\lambda_{\max}(D^T D)} \\ &\leq ||U_1 - U_2||_2 + \nu\lambda_{\max}(C^T C)||U_1 - U_2||_2 + \nu\lambda_{\max}(D^T D)||U_1 - U_2||_2 \\ &\leq \Big(1 + \nu\lambda_{\max}(C^T C) + \nu\lambda_{\max}(D^T D)\Big)||U_1 - U_2||_2\end{aligned}$$

□

Finally, it should be noted that for the time complexity, the proposed method is less sensitive to the λ value than the conventional methods. In fact, the λ value affects the proposed method only through the functions h_l^* and g_l^* that take 0 or ∞ depending on $\|y\|_2 \leq \lambda\omega_l$ and $\|\tilde{y}\|_2 \leq \lambda\tilde{\omega}_l$ in (23).

On the other hand, the COBRA solves two optimization problems, and the ADMM-based methods need to solve the Sylvester equation, which means that all of them are influenced by the λ value so much.

4. Experiments

In this section, we show the performance of the proposed approach for estimating and assessing the biclusters by conducting experiments on both synthetic and real datasets. We executed the following algorithms:

- COBRA: Dykstra-like proximal algorithm proposed by Chi et al. [9].
- ADMM: the ADMM proposed by Weylandt [19].
- G-ADMM (generalized ADMM): the modified ADMM presented by Weylandt [19].
- Proposed method: the proposed algorithm showed in Algorithm 2.

They were all implemented by Rcpp on a Macbook Air with 1.6 GHz Intel Core i5 and 8 GB memory. We recorded the wall times for the four algorithms.

4.1. Artificial Data Analysis

We evaluate the performance of the proposed methods on synthetic data in terms of the number of iterations, the execution time, and the clustering quality.

We generate the artificial data $X \in \mathbb{R}^{N \times p}$ with a checkerboard bicluster structure similar to the method in [9]. We simulate $X_{ij} \sim N(\mu_{rc}, \sigma^2)$ (i.i.d.) as follows, where the indices r and c range in clusters $\{1, \cdots, R\}$ and $\{1, \cdots, C\}$, respectively, which means that the number of biclusters is $M := R \times C$. We assign each x_{ij} randomly belongs to one of those M biclusters. The mean μ_{rc} is chosen uniformly from an equally spaced sequence $\{-10, -9, \cdots, 9, 10\}$, and the σ is chosen as 1.5 and 3.0 for different noise levels.

In our experiments, we consider the following stopping criteria for the four algorithms.

1. Relative error:
$$\frac{||U^{k+1} - U^k||_F}{\max\{||U^k||_F, 1\}} \le \epsilon.$$
2. Objective function error:
$$||F(U^k) - F(U^*)||_F \le \epsilon.$$

where ϵ is a given accuracy tolerance. We terminate the algorithm if the above error is smaller than ϵ or the maximum number of iterations exceeds 10,000. We use the relative error for the time comparisons and quality assessment and the objective function error for convergence rate analysis.

4.1.1. Comparisons

We change the sizes of N, p of the data matrix X and the tuning parameter λ to test the performance of four algorithms and compare the performance among the algorithms. At first, we compare the execution time with different λ, ranging from 1 to 2000, and setting $R = 4$, $C = 4$, $\sigma = 1.5$. We obtain the results shown in Figure 2a.

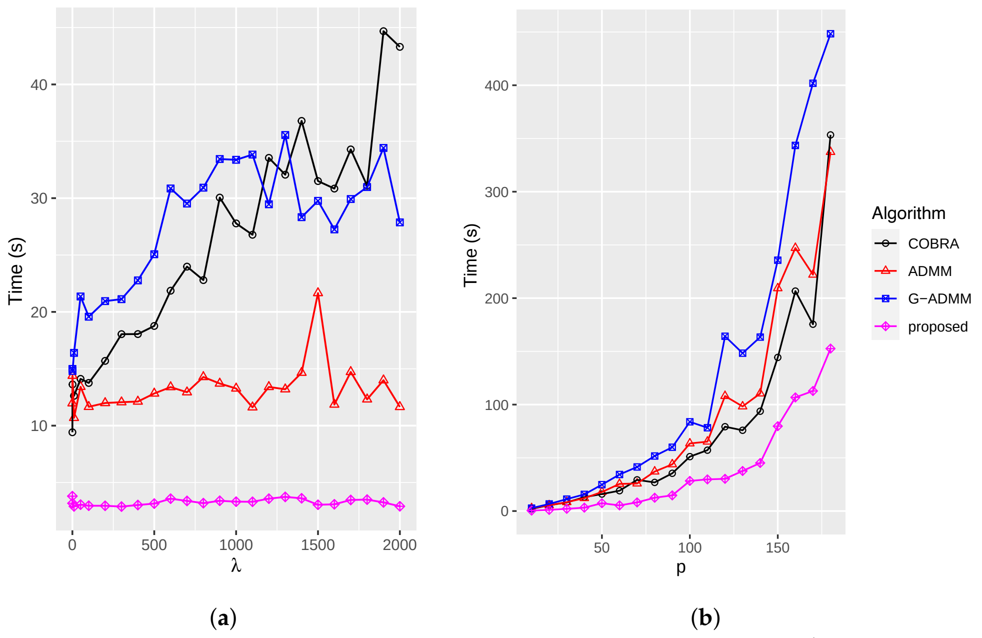


Figure 2. Execution time for various λ and p with $N = 100$ and $\epsilon = 1 \times 10^{-6}$. (**a**) Different λ, with $p = 40$; (**b**) different p, with $\lambda = 1$.

From Figure 2a, we observe that the execution time of the COBRA and G-ADMM increases rapidly as λ varies. The execution time of COBRA is the largest when $\lambda > 1400$. Therefore, it will take a long time for COBRA to visualize the whole fusion process, particularly the single bicluster case. Our proposed method significantly outperforms the other three algorithms and offers high stability in a wide range of λ. Due to its low computational time, our proposed method is a preferable choice to visualize the biclusters for various λ values when applying biclustering.

Next, we compare the execution time with different p (from 1 to 200). Here, we fix the number of column clusters and row clusters ($C = R = 4$). Figure 2b shows that the execution time of the algorithms increases as p grows. The proposed method shows better performance than the other three. In particular, the ADMM and G-ADMM are suffered from the feature dimension p and the computations grow dramatically.

Then, we vary the sample size N from 100 to 1000 and fix the size of the feature $p = 40$, with $\lambda = 1$, $R = 4$, $C = 4$, $\sigma = 1.5$. Figure 3 shows the execution times of the three algorithms (COBRA, G-ADMM, and Proposed) for each N.

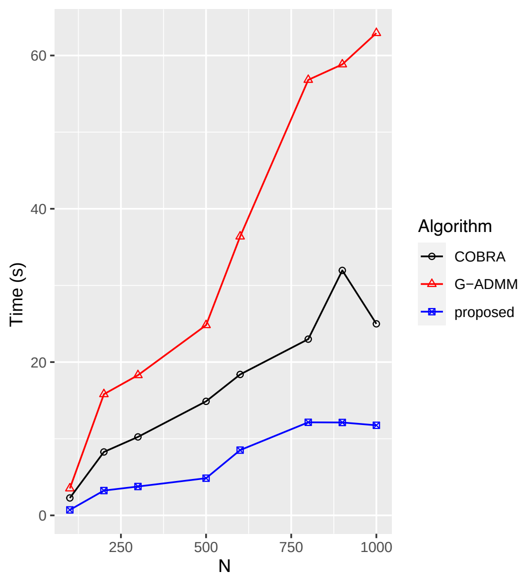


Figure 3. Execution times for each N with $\lambda = 1$, $p = 40$ and $\epsilon = 1 \times 10^{-6}$.

The curves reveal that when the larger the sample size N, the more time the algorithms require. Moreover, the ADMM takes more than 500 s when $N > 500$ and takes around 1800 s when $N = 1000$, which are much larger than the other three algorithms. Thus, we remove the result of ADMM from the figure. However, our proposed method only takes around 10 s even when $N = 1000$, which is six times smaller than G-ADMM.

4.1.2. Assessment

We evaluate the clustering quality by a widely used criterion called the Rand index (RI) [40]. The value of RI ranges from 0 to 1; a higher value shows better performance, and 1 indicates the perfect quality of the clustering. Note that we can obtain the true bicluster labels in the data generation procedure. We generate the matrix data with $N = 100$ and $p = 100$, and set two noise levels, low ($\sigma = 1.5$) and high ($\sigma = 3.0$). We compare the clustering quality of our proposed method with ADMM, G-ADMM, and COBRA under different settings. Setting 1: $R = 2$, $C = 4$, $\sigma = 1.5$; Setting 2: $R = 4$, $C = 4$, $\sigma = 1.5$; Setting 3: $R = 4$, $C = 8$, $\sigma = 1.5$; Setting 4: $R = 2$, $C = 4$, $\sigma = 3$; Setting 5: $R = 4$, $C = 4$, $\sigma = 3$; Setting 6: $R = 4$, $C = 8$, $\sigma = 3$.

Table 2 presents the result of the experiment. As the tuning parameter λ increases, the biclusters tend to fuse and reduce noise interference in the raw data. While in some cases, for extremely high λ such as 10,000, the biclusters may be over-smoothed, and the value of the Rand index is decreased. For example, in the first case (Setting 1: the number of biclusters is 2×4 and $\sigma = 1.5$). The Rand index in COBRA, ADMM, and our proposed method shows a similar value in most cases because all the algorithms solve the same model. However, the G-ADMM exhibits the worst performance due to its slow convergence rate, and it cannot converge well when the tuning parameter λ is large ($\lambda = 5000$ and $\lambda = 10{,}000$). Overall, from the results in Table 2, our proposed method shows high accuracy and stability from low to high noise.

Table 2. Assessment result.

Setting	**Algorithm**	**Rand Index**			
		$\lambda = 100$	$\lambda = 1000$	$\lambda = 5000$	$\lambda = 10,000$
Setting 1	COBRA	0.874	0.875	0.999	0.931
	ADMM	0.872	0.875	0.999	0.931
	G-ADMM	0.874	0.875	0.874	0.872
	Proposed	0.875	0.875	0.999	0.931
Setting 2	COBRA	0.928	0.932	0.994	0.999
	ADMM	0.928	0.935	0.994	0.999
	G-ADMM	0.928	0.934	0.981	0.936
	Proposed	0.928	0.934	0.994	0.999
Setting 3	COBRA	0.959	0.962	0.962	0.999
	ADMM	0.961	0.962	0.962	0.998
	G-ADMM	0.959	0.962	0.967	0.967
	Proposed	0.961	0.962	0.962	0.999
Setting 4	COBRA	0.870	0.870	0.870	0.935
	ADMM	0.870	0.868	0.871	0.933
	G-ADMM	0.870	0.870	0.871	0.871
	Proposed	0.870	0.870	0.871	0.935
Setting 5	COBRA	0.934	0.934	0.934	0.964
	ADMM	0.934	0.932	0.934	0.964
	G-ADMM	0.934	0.934	0.932	0.932
	Proposed	0.934	0.934	0.934	0.964
Setting 6	COBRA	0.960	0.960	0.962	0.962
	ADMM	0.961	0.962	0.962	0.962
	G-ADMM	0.960	0.962	0.962	0.960
	Proposed	0.961	0.962	0.962	0.962

4.2. Real Data Analysis

In this section, we use three different real datasets to demonstrate the performance of our proposed method.

Firstly, we use the presidential speeches dataset preprocessed by Weylandt et al. [41] that contains 75 high-frequency words taken from the significant speeches of the 44 U.S. presidents around the year 2018. We show the heatmaps in Figure 4 under a wide range of tuning parameters λ to exhibit the fusion process of biclusters. We set the tolerance ϵ to be 1×10^{-6}, and use the relative error stopping criterion as described in Section 4.1. The columns represent the different presidents, and the rows represent the different words. When $\lambda = 0$, the heatmap is disordered, and there are no distinct subgroups. While we increase the λ, the biclusters begin to merge. We can further find out the common vocabulary used in some groups of the prime minister's speeches. Moreover, as shown in Figure 4f, the heatmap clearly shows four biclusters with two subgroups of presidents and two subgroups of words when $\lambda = 30{,}000$.

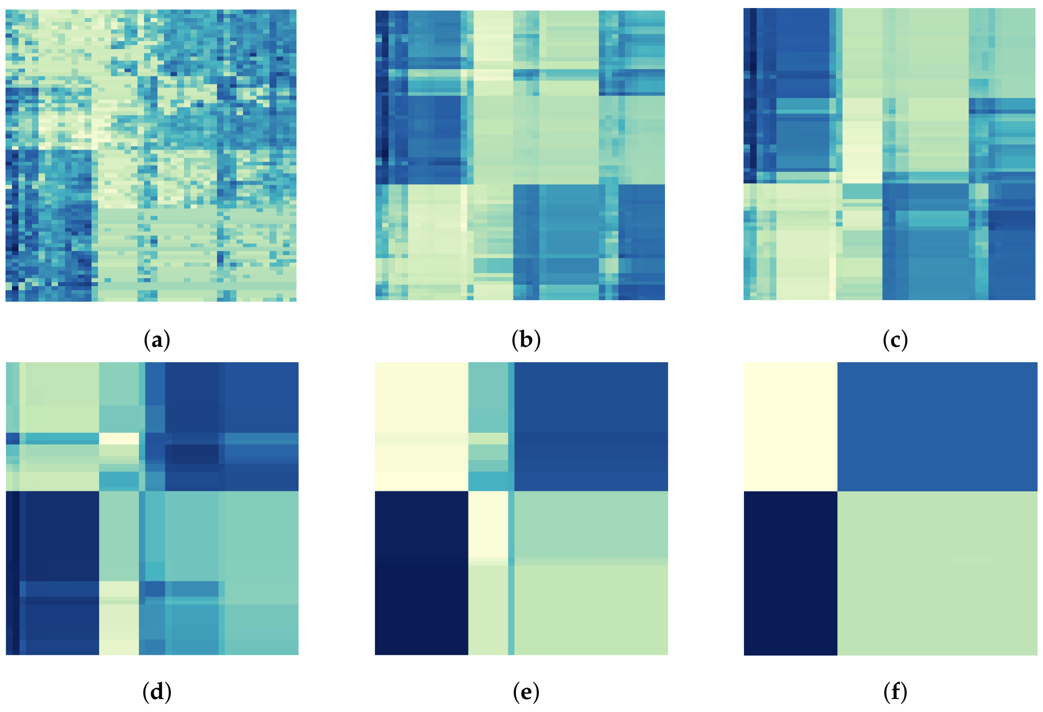


Figure 4. The heatmap results of proposed method implementation on the presidential speeches dataset under a wide range of λ. (**a**) $\lambda = 0$; (**b**) $\lambda = 1500$; (**c**) $\lambda = 2000$; (**d**) $\lambda = 5000$; (**e**) $\lambda = 15{,}000$; (**f**) $\lambda = 30{,}000$.

Secondly, we compare the computational time of four algorithms for two actual datasets. One is The Cancer Genome Atlas (TCGA) dataset [42], which contains 438 breast cancer patients (samples) and 353 genes (features), and the other one is the diffuse large-B-cell lymphoma (DLBCL) dataset [43] with 3795 genes and 58 patient samples. In DLBCL, there are 32 samples from cured patients and 26 samples from sick individuals among the 58 samples. Furthermore, we extract 500 genes with the highest variances among the original genes.

Figure 5a,b depicts the outcomes of the elapsed time comparison. From the curves, we observe that our proposed approach surpasses the other three methods. In contrast, ADMM shows the worst performance in the DLBCL dataset, and the case of tolerance $\epsilon < 10^{-3}$ requirement in the TCGA dataset.

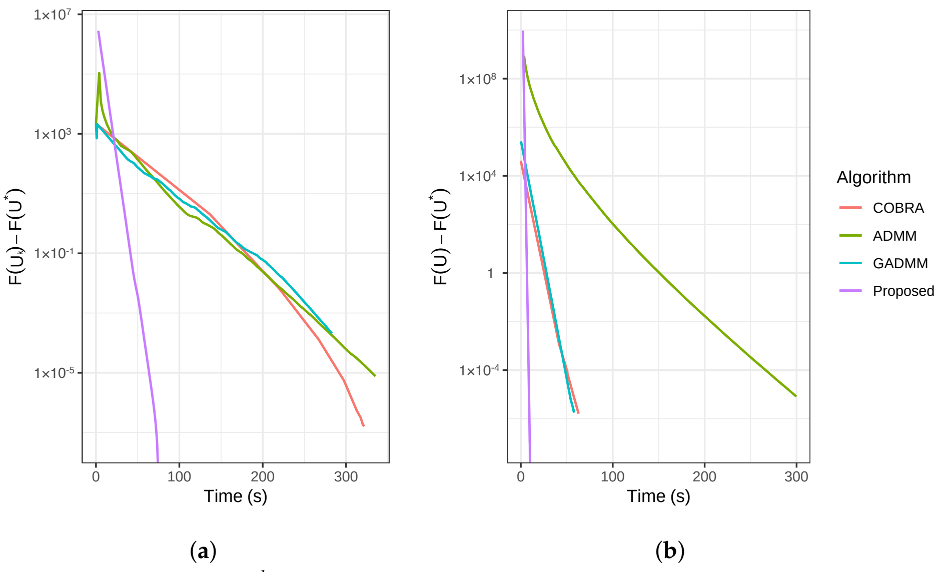
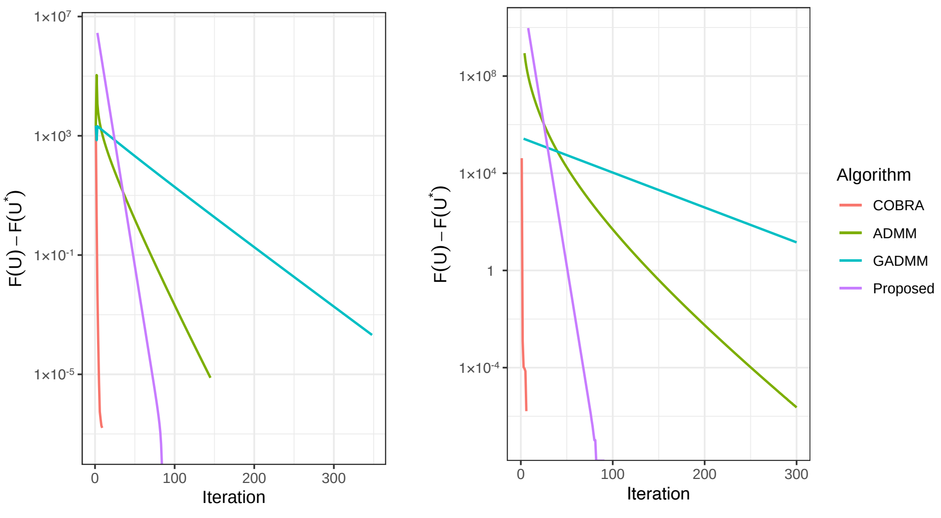


Figure 5. Plot of $\log(F(U^k) - F(U^*))$ vs. the elapsed time. (**a**) TCGA; (**b**) DLBCL.

Lastly, we compare the number of iterations to achieve the specified tolerance of $F(U^k) - F(U^*)$ and run it on the TCGA and DLBCL datasets. Figure 6a,b reveals that the COBRA algorithm has the fastest convergence rate, whereas the generalized ADMM is the slowest to converge. Our proposed method shows competitive performance in the convergence rate.



(**a**) (**b**)

Figure 6. Plot of $\log(F(U^k) - F(U^*))$ vs. the number of iterations. (**a**) TCGA; (**b**) DLBCL.

Overall, from the above experiment results of the artificial and real datasets, our proposed method has superior computational performance with high accuracy.

5. Discussion

We proposed a method to find a solution to the convex biclustering problem. We found that it outperformed the conventional algorithms, such as COBRA and ADMM-based procedures, in the sense of efficiency. Our proposed method is more efficient than COBRA because the latter should solve two optimization problems containing non-differentiable fused terms in each cycle. Additionally, the proposed method performed better than the ADMM-based procedures because the former is based on [29] and uses the NAGM to update the variable U. However, ADMM spends much more time computing the matrix inverse. Moreover, our proposed method is stable while varying the tuning parameters λ, which is convenient for us to find the optimal λ and visualize the variation of the heatmaps under a wide range of λ.

As for further improvements, we can use ADMM as a warm start strategy to select an initial value for our proposed method. What is more, according to the fusion process of the heatmap results in Figure 4, it will be meaningful if we can derive the range of tuning parameters λ that yield the non-trivial solutions of the convex biclustering with more than one bicluster. Additionally, the proposed method can motivate future work. We can extend the proposed method to solve other clustering problems, such as the sparse singular value decomposition model [44] and the integrative generalized convex clustering model [45].

Author Contributions: Conceptualization, J.C. and J.S.; methodology, J.C. and J.S.; software, J.C.; formal analysis, J.C. and J.S.; investigation, J.C.; resources, J.C.; data curation, J.C.; writing—original draft preparation, J.C. and J.S.; writing—review and editing, J.C. and J.S.; visualization, J.C.; supervision, J.S.; project administration, J.S.; funding acquisition, J.S. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by Grant-in-Aid for Scientific Research (KAKENHI) C, Grant number: 18K11192.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Publicly available datasets were analyzed in this paper. Presidential speeches dataset: https://www.presidency.ucsb.edu (accessed date 18 October 2021); DLBCL dataset: http://portals.broadinstitute.org/cgi-bin/cancer/datasets.cgi (accessed date 18 October 2021).

Acknowledgments: The authors would like to thank Ryosuke Shimmura for helpful discussions.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

ADMM	alternating direction method of multipliers
ALM	augmented Lagrangian method
COBRA	convex biclustering algorithm
DLBCL	diffuse large-B-cell lymphoma
FISTA	fast iterative shrinkage-thresholding algorithm
ISTA	iterative shrinkage-thresholding algorithm
NAGM	Nesterov's accelerated gradient method
RI	Rand index
TCGA	The Cancer Genome Atlas

Appendix A

Appendix A.1. Proof of Proposition 1

First, we define the following two functions,

$$r_1(V) := h(V) + \frac{\nu}{2}||V||_F^2,$$

and

$$r_2(Z) := g(Z) + \frac{\nu}{2}||Z||_F^2.$$

By the definition of $F(U)$ in (18),

$$\begin{aligned}
F(U) :&= f(U) + \min_{V,Z}\{L(U,V,Z,\Lambda_1,\Lambda_2)\} \\
&= f(U) + \min_{V,Z}\Bigg\{h(V) + g(Z) + \sum_l \langle \Lambda_{1l}, C_{l,\cdot}U - V_l\rangle + \frac{\nu}{2}\sum_l ||C_{l,\cdot}U - V_l||_2^2 \\
&\qquad + \sum_l \langle \Lambda_{2l}, UD_{\cdot,l} - Z_l\rangle + \frac{\nu}{2}\sum_l ||UD_{\cdot,l} - Z_l||_2^2\Bigg\} \\
&= f(U) + \min_V\Bigg\{h(V) + \frac{\nu}{2}||V||_F^2 - \sum_l \langle \Lambda_{1l} + \nu C_{l,\cdot}U, V_l\rangle\Bigg\} + \frac{\nu}{2}||CU||_F^2 + \sum_l \langle \Lambda_{1l}, C_{l,\cdot}U\rangle \\
&\quad + \min_Z\Bigg\{g(Z) + \frac{\nu}{2}||Z||_F^2 - \sum_l \langle \Lambda_{2l} + \nu UD_{\cdot,l}, Z_l\rangle\Bigg\} + \frac{\nu}{2}||UD||_F^2 + \sum_l \langle \Lambda_{2l}, UD_{\cdot,l}\rangle \\
&= -\max_V\Big\{\langle \nu CU + \Lambda_1, V\rangle - h(V) - \frac{\nu}{2}||V||_F^2\Big\} + f(U) + \frac{\nu}{2}||CU||_F^2 + \sum_l \langle \Lambda_{1l}, C_{l,\cdot}U\rangle \\
&\quad - \max_Z\Big\{\langle \nu UD + \Lambda_2, Z\rangle - g(Z) - \frac{\nu}{2}||Z||_F^2\Big\} + \frac{\nu}{2}||UD||_F^2 + \sum_l \langle \Lambda_{2l}, UD_{\cdot,l}\rangle \\
&= f(U) - r_1^*(\nu CU + \Lambda_1) - r_2^*(\nu UD + \Lambda_2) + \frac{\nu}{2}||CU||_F^2 + \langle \Lambda_1, CU\rangle \\
&\quad + \frac{\nu}{2}||UD||_F^2 + \langle \Lambda_2, UD\rangle.
\end{aligned}$$

By Theorem 26.3 in [33], if the function $r : \mathbb{R}^p \to \mathbb{R}$ is closed and strongly convex, then we have the differentiable conjugate function $r^*(v)$, and

$$\nabla r^*(v) = \arg\max_{u \in \mathbb{R}^p} \{\langle u, v\rangle - r(u)\}.$$

Hence, we can derive the following equations,

$$\begin{aligned}
\nabla r_1^*(v) &= \arg\max_u \Big\{ \langle u, v\rangle - h(u) - \frac{\nu}{2}||u||_F^2 \Big\} \\
&= \arg\max_u \Big\{ -\frac{1}{2}||u||_F^2 + \frac{1}{\nu}\langle u, v\rangle - \frac{1}{\nu}h(u) \Big\} \\
&= \arg\min_u \Big\{ \frac{1}{2}||u - \frac{v}{\nu}||_F^2 + \frac{1}{\nu}h(u) \Big\} \\
&= \text{prox}_{h/\nu}(\frac{v}{\nu}).
\end{aligned}$$

Then, we obtain

$$\nabla r_1^*(\nu CU + \Lambda_1) = \nu C^T \Big(\text{prox}_{h/\nu}(CU + \frac{\Lambda_1}{\nu}) \Big)$$

and, similarly,

$$\nabla r_2^*(\nu UD + \Lambda_2) = \nu \Big(\text{prox}_{g/\nu}(UD + \frac{\Lambda_2}{\nu}) \Big) D^T.$$

Next, take the derivative of $F(U)$ w.r.t U,

$$\begin{aligned}
\nabla_U F(U) &= \nabla f(U) - \nabla r_1^*(\nu CU + \Lambda_1) - \nabla r_2^*(\nu UD + \Lambda_2) \\
&\quad + \nu C^T CU + \nu UDD^T + C^T\Lambda_1 + \Lambda_2 D^T \\
&= \nabla f(U) - \nu C^T \Big(\text{prox}_{h/\nu}(CU + \nu^{-1}\Lambda_1) \Big) - \nu \Big(\text{prox}_{g/\nu}(UD + \frac{\Lambda_2}{\nu}) \Big) D^T \\
&\quad + \nu C^T CU + C^T\Lambda_1 + \nu UDD^T + \Lambda_2 D^T \\
&= \nabla f(U) + C^T(\text{prox}_{\nu h^*}(\nu CU + \Lambda_1)) + \Big(\text{prox}_{\nu g^*}(\nu UD + \Lambda_2) \Big) D^T,
\end{aligned}$$

and we obtain the last equation by Moreau's decomposition.

References

1. Hartigan, J.A.; Wong, M.A. Algorithm AS 136: A k-means clustering algorithm. *J. R. Stat. Soc. Ser. (Appl. Stat.)* **1979**, *28*, 100–108. [CrossRef]
2. Johnson, S.C. Hierarchical clustering schemes. *Psychometrika* **1967**, *32*, 241–254. [CrossRef] [PubMed]
3. Hastie, T.; Tibshirani, R.; Friedman, J. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*; Springer: Berlin/Heidelberg, Germany, 2017.
4. Hartigan, J.A. Direct clustering of a data matrix. *J. Am. Stat. Assoc.* **1972**, *67*, 123–129. [CrossRef]
5. Cheng, Y.; Church, G.M. Biclustering of expression data. *ISMB Int. Conf. Intell. Syst. Mol. Biol.* **2000**, *8*, 93–103.
6. Tanay, A.; Sharan, R.; Shamir, R. Discovering statistically significant biclusters in gene expression data. *Bioinformatics* **2002**, *18*, S136–S144. [CrossRef]
7. Prelić, A.; Bleuler, S.; Zimmermann, P.; Wille, A.; Bühlmann, P.; Gruissem, W.; Hennig, L.; Thiele, L.; Zitzler, E. A systematic comparison and evaluation of biclustering methods for gene expression data. *Bioinformatics* **2006**, *22*, 1122–1129. [CrossRef] [PubMed]
8. Madeira, S.C.; Oliveira, A.L. Biclustering algorithms for biological data analysis: A survey. *IEEE/ACM Trans. Comput. Biol. Bioinform.* **2004**, *1*, 24–45. [CrossRef]
9. Chi, E.C.; Allen, G.I.; Baraniuk, R.G. Convex biclustering. *Biometrics* **2017**, *73*, 10–19. [CrossRef]
10. Tibshirani, R.; Saunders, M.; Rosset, S.; Zhu, J.; Knight, K. Sparsity and smoothness via the fused lasso. *J. R. Stat. Soc. Ser. (Stat. Methodol.)* **2005**, *67*, 91–108. [CrossRef]
11. Hocking, T.D.; Joulin, A.; Bach, F.; Vert, J.P. Clusterpath an algorithm for clustering using convex fusion penalties. In Proceedings of the 28th International Conference on Machine Learning, ICML 2011, Bellevue, DC, USA, 28 June–2 July 2011; p. 1.

12. Lindsten, F.; Ohlsson, H.; Ljung, L. Clustering using sum-of-norms regularization: With application to particle filter output computation. In Proceedings of the 2011 IEEE Statistical Signal Processing Workshop (SSP), Nice, France, 28–30 June 2011; pp. 201–204.
13. Pelckmans, K.; De Brabanter, J.; Suykens, J.A.; De Moor, B. Convex clustering shrinkage. In Proceedings of the PASCALWorkshop on Statistics and Optimization of Clustering Workshop, London, UK, 4–5 July 2005.
14. Langfelder, P.; Horvath, S. WGCNA: An R package for weighted correlation network analysis. *BMC Bioinform.* **2008**, *9*, 1–13. [CrossRef]
15. Tan, K.M.; Witten, D.M. Sparse biclustering of transposable data. *J. Comput. Graph. Stat.* **2014**, *23*, 985–1008. [CrossRef] [PubMed]
16. Shor, N.Z. *Minimization Methods for Non-Differentiable Functions*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2012; Volume 3.
17. Boyd, S.; Xiao, L.; Mutapcic, A. Subgradient methods. *Lect. Notes EE392o Stanf. Univ. Autumn Quart.* **2003**, *2004*, 2004–2005.
18. Bauschke, H.H.; Combettes, P.L. A Dykstra-like algorithm for two monotone operators. *Pac. J. Optim.* **2008**, *4*, 383–391.
19. Weylandt, M. Splitting methods for convex bi-clustering and co-clustering. In Proceedings of the 2019 IEEE Data Science Workshop (DSW) , Minneapolis, MN, USA, 2–5 June 2019; pp. 237–242.
20. Glowinski, R.; Marroco, A. Sur l'approximation, par éléments finis d'ordre un, et la résolution, par pénalisation-dualité d'une classe de problèmes de Dirichlet non linéaires. *ESAIM Math. Model. Numer. Anal.-ModéLisation MathéMatique Anal. NuméRique* **1975**, *9*, 41–76. [CrossRef]
21. Gabay, D.; Mercier, B. A dual algorithm for the solution of nonlinear variational problems via finite element approximation. *Comput. Math. Appl.* **1976**, *2*, 17–40. [CrossRef]
22. Deng, W.; Yin, W. On the global and linear convergence of the generalized alternating direction method of multipliers. *J. Sci. Comput.* **2016**, *66*, 889–916. [CrossRef]
23. Chi, E.C.; Lange, K. Splitting methods for convex clustering. *J. Comput. Graph. Stat.* **2015**, *24*, 994–1013. [CrossRef] [PubMed]
24. Suzuki, J. *Sparse Estimation with Math and R: 100 Exercises for Building Logic*; Springer: Berlin/Heidelberg, Germany, 2021.
25. Bartels, R.H.; Stewart, G.W. Solution of the matrix equation AX+ XB= C [F4]. *Commun. ACM* **1972**, *15*, 820–826. [CrossRef]
26. Goldstein, T.; O'Donoghue, B.; Setzer, S.; Baraniuk, R. Fast alternating direction optimization methods. *Siam J. Imaging Sci.* **2014**, *7*, 1588–1623. [CrossRef]
27. Boyd, S.; Parikh, N.; Chu, E. *Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers*; Now Publishers Inc.: Boston, MA, USA, 2011.
28. Abualigah, L.; Diabat, A.; Mirjalili, S.; Abd Elaziz, M.; Gandomi, A.H. The arithmetic optimization algorithm. *Comput. Methods Appl. Mech. Eng.* **2021**, *376*, 113609. [CrossRef]
29. Shimmura, R.; Suzuki, J. Converting ADMM to a Proximal Gradient for Convex Optimization Problems. *arXiv* **2021**, arXiv:2104.10911.
30. Beck, A.; Teboulle, M. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *Siam J. Imaging Sci.* **2009**, *2*, 183–202. [CrossRef]
31. Moreau, J.J. Proximité et dualité dans un espace hilbertien. *Bull. Soc. Math. France* **1965**, *93*, 273–299. [CrossRef]
32. Hestenes, M.R. Multiplier and gradient methods. *J. Optim. Theory Appl.* **1969**, *4*, 303–320. [CrossRef]
33. Rockafellar, R.T. The multiplier method of Hestenes and Powell applied to convex programming. *J. Optim. Theory Appl.* **1973**, *12*, 555–562. [CrossRef]
34. Nocedal, J.; Wright, S. *Numerical Optimization*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2006.
35. Nesterov, Y.E. A method for solving the convex programming problem with convergence rate O $(1/k^2)$. *Dokl. Akad. Nauk Sssr* **1983**, *269*, 543–547.
36. Boyd, S.; Boyd, S.P.; Vandenberghe, L. *Convex Optimization*; Cambridge University Press: Cambridge, UK, 2004.
37. Beck, A. *First-Order Methods in Optimization*; SIAM: Philadelphia, PA, USA, 2017.
38. Nemirovski, A.; Yudin, D. *Problem Complexity and Method Efficiency in Optimization*; John Wiley: Hoboken, NJ, USA, 1983.
39. Combettes, P.L.; Wajs, V.R. Signal recovery by proximal forward-backward splitting. *Multiscale Model. Simul.* **2005**, *4*, 1168–1200. [CrossRef]
40. Rand, W.M. Objective criteria for the evaluation of clustering methods. *J. Am. Stat. Assoc.* **1971**, *66*, 846–850. [CrossRef]
41. Weylandt, M.; Nagorski, J.; Allen, G.I. Dynamic visualization and fast computation for convex clustering via algorithmic regularization. *J. Comput. Graph. Stat.* **2020**, *29*, 87–96. [CrossRef]
42. Koboldt, D.; Fulton, R.; McLellan, M.; Schmidt, H.; Kalicki-Veizer, J.; McMichael, J.; Fulton, L.; Dooling, D.; Ding, L.; Mardis, E.; et al. Comprehensive molecular portraits of human breast tumours. *Nature* **2012**, *490*, 61–70.
43. Rosenwald, A.; Wright, G.; Chan, W.C.; Connors, J.M.; Campo, E.; Fisher, R.I.; Gascoyne, R.D.; Muller-Hermelink, H.K.; Smeland, E.B.; Giltnane, J.M.; et al. The use of molecular profiling to predict survival after chemotherapy for diffuse large-B-cell lymphoma. *N. Engl. J. Med.* **2002**, *346*, 1937–1947. [CrossRef] [PubMed]
44. Lee, M.; Shen, H.; Huang, J.Z.; Marron, J. Biclustering via sparse singular value decomposition. *Biometrics* **2010**, *66*, 1087–1095. [CrossRef] [PubMed]
45. Wang, M.; Allen, G.I. Integrative generalized convex clustering optimization and feature selection for mixed multi-view data. *J. Mach. Learn. Res.* **2021**, *22*, 1–73.