HHS Public Access

Author manuscript

J Mach Learn Res. Author manuscript; available in PMC 2021 November 05.

Published in final edited form as:

J Mach Learn Res. 2021 January ; 22: .

Integrative Generalized Convex Clustering Optimization and Feature Selection for Mixed Multi-View Data

Minjie Wang,

Department of Statistics, Rice University, Houston, TX 77005, USA

Genevera I. Allen

Departments of Electrical and Computer Engineering, Statistics, and Computer Science, Rice University, Houston, TX 77005, USA; Jan and Dan Duncan Neurological Research Institute, Baylor College of Medicine, Houston, TX 77030, USA

Abstract

In mixed multi-view data, multiple sets of diverse features are measured on the same set of samples. By integrating all available data sources, we seek to discover common group structure among the samples that may be hidden in individualistic cluster analyses of a single data view. While several techniques for such integrative clustering have been explored, we propose and develop a convex formalization that enjoys strong empirical performance and inherits the mathematical properties of increasingly popular convex clustering methods. Specifically, our Integrative Generalized Convex Clustering Optimization (iGecco) method employs different convex distances, losses, or divergences for each of the different data views with a joint convex fusion penalty that leads to common groups. Additionally, integrating mixed multi-view data is often challenging when each data source is high-dimensional. To perform feature selection in such scenarios, we develop an adaptive shifted group-lasso penalty that selects features by shrinking them towards their loss-specific centers. Our so-called iGecco+ approach selects features from each data view that are best for determining the groups, often leading to improved integrative clustering. To solve our problem, we develop a new type of generalized multi-block ADMM algorithm using sub-problem approximations that more efficiently fits our model for big data sets. Through a series of numerical experiments and real data examples on text mining and genomics, we show that iGecco+ achieves superior empirical performance for high-dimensional mixed multi-view data.

Keywords

Integrative clustering; convex clustering; feature selection; convex optimization; sparse clustering; GLM deviance; Bregman divergences

MW50@RICE.EDU .

1. Introduction

As the volume and complexity of data grows, statistical data integration has gained increasing attention as it can lead to discoveries which are not evident in analyses of a single data set. We study a specific data-integration problem where we seek to leverage common samples measured across multiple diverse sets of features that are of different types (e.g., continuous, count-valued, categorical, skewed continuous and etc.). This type of data is often called mixed, multi-view data (Hall and Llinas, 1997; Acar et al., 2011; Lock et al., 2013; Tang and Allen, 2018; Baker et al., 2019). While many techniques have been developed to analyze each individual data type separately, there are currently few methods that can directly analyze mixed multi-view data jointly. Yet, such data is common in many areas such as electronic health records, integrative genomics, multi-modal imaging, remote sensing, national security, online advertising, and environmental studies. For example in genomics, scientists often study gene regulation by exploring only gene expression data, but other data types, such as short RNA expression and DNA methylation, are all part of the same gene regulatory system. Joint analysis of such data can give scientists a more holistic view of the problem they study. But, this presents a major challenge as each individual data type is high-dimensional (i.e., a larger number of features than samples) with many uninformative features. Further, each data view can be of a different data type: expression of genes or short RNAs measured via sequencing is typically count-valued or zero-inflated plus skewed continuous data whereas DNA methylation data is typically proportion-valued. In this paper, we seek to leverage multiple sources of mixed data to better cluster the common samples as well as select relevant features that distinguish the inherent group structure.

We propose a convex formulation which integrates mixed types of data with different data-specific losses, clusters common samples with a joint fusion penalty and selects informative features that separate groups. Due to the convex formulation, our methods enjoy strong mathematical and empirical properties. We make several methodological contributions. First, we consider employing different types of losses for better handling non-Gaussian data with Generalized Convex Clustering Optimization (Gecco), which replaces Euclidean distances in convex clustering with more general convex losses. We show that for different losses, Gecco's fusion penalty forms different types of centroids which we call loss-specific centers. To integrate mixed multi-view data and perform clustering, we incorporate different convex distances, losses, or divergences for each of the different data views with a joint convex fusion penalty that leads to common groups; this gives rise to Integrative Generalized Convex Clustering (iGecco). Further, when dealing with high-dimensional data, practitioners seek interpretability by identifying important features which can separate the groups. To facilitate feature selection in Gecco and iGecco, we develop an adaptive shifted group-lasso penalty that selects features by shrinking them towards their loss-specific centers, leading to Gecco+ and iGecco+ which performs clustering and variable selection simultaneously. To solve our problems in a computationally efficient manner, we develop a new general multi-block ADMM algorithm using sub-problem approximations, and make an optimization contribution by proving that this new class of algorithms converge to the global solution.

1.1 Related Literature

Our goal is to develop a unified, convex formulation of integrative clustering with feature selection based on increasingly popular convex clustering methods. Pelckmans et al. (2005); Lindsten et al. (2011); Hocking et al. (2011) proposed convex clustering which uses a fusion penalty to achieve agglomerative clustering like hierarchical clustering. This convex formulation guarantees a global optimal solution, enjoys strong statistical and mathematical theoretical properties, and often demonstrates superior empirical performance to competing approaches. Specifically, in literature, Pelckmans et al. (2005); Chi et al. (2017) showed it yields stable solutions to small perturbations on the data or tuning parameters; Radchenko and Mukherjee (2017) established clustering consistency by proving the clustering tree produced by convex clustering consistently estimates the clustering tree produced by the population procedure for the ℓ_1 penalty case; Tan and Witten (2015) established its link to hierarchical clustering as well as prediction consistency (finite sample bound for the prediction error); the perfect recovery properties of convex clustering with uniform weights have been proved by Zhu et al. (2014) for the two-clusters case and Panahi et al. (2017) for the general k-clusters case while Sun et al. (2021) proved results for general weighted convex clustering model; and many others have studied other appealing theoretical properties (Wu et al., 2016; Chi and Steinerberger, 2019). Despite these advantages, convex clustering has not yet gained widespread popularity due to its intensive computation. Recently, some proposed fast and efficient algorithms to solve convex clustering and estimate its regularization paths (Chi and Lange, 2015; Weylandt et al., 2020). Meanwhile, convex clustering has been extended to biclustering (Chi et al., 2017) and many other applications (Chi et al., 2018; Choi et al., 2019).

One potential drawback to convex clustering however, is that thus far, it has only been well-studied employing Euclidean distances between data points and their corresponding cluster centers. As is well known, the Euclidean metric suffers from poor performance with data that is highly non-Gaussian such as binary, count-valued, skewed data, or with data that has outliers. To alleviate the impact of outliers, Wang et al. (2016) studied robust convex clustering, Sui et al. (2018) investigated convex clustering with metric learning and Wu et al. (2016) mentioned replacing ℓ_2-norm with ℓ_1-norm loss function as extensions. Despite these, however, no one has conducted a general investigation of convex clustering for non-Gaussian data, let alone studied data integration on mixed data, to the best of our knowledge. But, many others have proposed clustering methods for non-Gaussian data in other contexts. One approach is to perform standard clustering procedures on transformed data (Anders and Huber, 2010; Bullard et al., 2010; Marioni et al., 2008; Robinson and Oshlack, 2010). But, choosing an appropriate transformation that retains the original cluster signal is a challenging problem. Another popular approach is to use hierarchical clustering with specified distance metrics for non-Gaussian data (Choi et al., 2010; Fowlkes and Mallows, 1983). Closely related to this, Banerjee et al. (2005) studied different clustering algorithms utilizing a large class of loss functions via Bregman divergences. Yet, the proposed methods are all extensions of existing clustering approaches and hence inherit both good and bad properties of those approaches. There has also been work on model-based clustering, which assumes that data are generated by a finite mixture model; for example Banfield and Raftery (1993); Si et al. (2013) proposed such a model for the Poisson and

negative binomial distributions. Still these methods have a non-convex formulation and local solutions like all model-based clustering methods. We propose to adopt the method similar to Banerjee et al. (2005) and study convex clustering using different loss functions; hence our method inherits the desirable properties of convex clustering and handles non-Gaussian data as well. More importantly, there is currently no literature on data integration using convex clustering and we achieve this by integrating different types of general convex losses with a joint fusion penalty.

Integrative clustering, however, has been well-studied in the literature. The most popular approach is to use latent variables to capture the inherent structure of multiple types of data. This achieves a joint dimension reduction and then clustering is performed on the joint latent variables (Shen et al., 2009, 2012, 2013; Mo et al., 2013, 2017; Meng et al., 2015). Similar in nature to the latent variables approach, matrix factorization methods assume that the data has an intrinsic low-dimensional representation, with the dimension often corresponding to the number of clusters (Lock et al., 2013; Hellton and Thoresen, 2016; Zhang et al., 2012; Chalise and Fridley, 2017; Zhang et al., 2011; Yang and Michailidis, 2015). There are a few major drawbacks of latent variable or dimension reduction approaches, however. First it is often hard to directly interpret latent factors or low-dimensional projections. Second, many of these approaches are based on non-convex formulations yielding local solutions. And third, choosing the rank of factors or projections is known to be very challenging in practice and will often impact resulting clustering solutions. Another approach to integrative clustering is clustering of clusters (COC) which performs cluster analysis on every single data set and then integrates the primary clustering results into final group assignments using consensus clustering (Hoadley et al., 2014; Lock and Dunson, 2013; Kirk et al., 2012; Savage et al., 2013; Wang et al., 2014). This, however, has several potential limitations as each individual data set might not have enough signal to discern joint clusters or the individual cluster assignments are too disparate to reach a meaningful consensus. Finally, others have proposed to use distance-based clustering for mixed types of data by first defining an appropriate distance metric for mixed data (for example, the Gower distance by Gower, 1971) and then applying an existing distance-based clustering algorithm such as hierarchical clustering (Ahmad and Dey, 2007; Ji et al., 2012). Consequently, this method inherits both good and bad properties of distance-based clustering approaches. Notice that all of these approaches are either two-step approaches or are algorithmic or non-convex problems that yield local solutions. In practice, such approaches often lead to unreliable and unstable results.

Clustering is known to perform poorly for high-dimensional data as most techniques are highly sensitive to uninformative features. One common approach is to reduce the dimensionality of the data via PCA, NMF, or t-SNE before clustering (Ghosh and Chinnaiyan, 2002; Bernardo et al., 2003; Tamayo et al., 2007). A major limitation of such approaches is that the resulting clusters are not directly interpretable in terms of feature importance. To address this, several have proposed sparse clustering for high-dimensional data. This performs clustering and feature selection simultaneously by iteratively applying clustering techniques to subsets of features selected via regularization (Witten and Tibshirani, 2010; Sun et al., 2012; Chang et al., 2014). The approach, however, is non-convex and is highly susceptible to poor local solutions. Others have proposed

penalized model-based clustering that selects features (Raftery and Dean, 2006; Wang and Zhu, 2008; Pan and Shen, 2007). Still, these methods inherit several advantages and disadvantages of model-based clustering approaches. Moreover, sparse integrative clustering is relatively under-studied. Shen et al. (2013); Mo et al. (2013) extended iCluster using a penalized latent variable approach to jointly model multiple omics data types. They induced sparsity on the latent variable coefficients via regularization. As feature selection is performed on the latent variables, however, this is less interpretable in terms of selecting features directly responsible for distinguishing clusters. Recently, and most closely related to our own work, Wang et al. (2018) proposed sparse convex clustering which adds a group-lasso penalty term on the cluster centers to shrink them towards zero, thus selecting relevant features. This penalty, however, is only appropriate for Euclidean distances when the data is centered; otherwise, the penalty term shrinks towards the incorrect cluster centers. For feature selection using different distances and losses, we propose an adaptive shifted group-lasso penalty that will select features by shrinking them towards their appropriate centroid.

2. Integrative Generalized Convex Clustering with Feature Selection

In this section, we introduce our new methods, beginning with the Gecco and iGecco and then show how to achieve feature selection via regularization. We also discuss some practical considerations for applying our methods and develop an adaptive version of our approaches.

2.1 Generalized Convex Clustering Optimization (Gecco)

In many applications, we seek to cluster data that is non-Gaussian. In the literature, most do this using different distance metrics other than Euclidean distances (Choi et al., 2010; Fowlkes and Mallows, 1983; de Souza and De Carvalho, 2004). Some use losses based on exponential family or deviances closely related to Bregman divergences (Banerjee et al., 2005).

To account for different types of losses for non-Gaussian data, we propose to replace the Euclidean distances in convex clustering with more general convex losses; this gives rise to Generalized Convex Clustering Optimization (Gecco):

$$\underset{\mathbf{U}}{\text{minimize}} \sum_{i=1}^{n} \ell\left(\mathbf{X}_{i\cdot}, \mathbf{U}_{i\cdot}\right) + \gamma \sum_{i<i'} \omega_{ii'} \|\mathbf{U}_{i\cdot} - \mathbf{U}_{i'\cdot}\|_q.$$

Here, our data $\mathbf{X}$ is an $n \times p$ matrix consisting of n observations and p features; $\mathbf{U}$ is an $n \times p$ centroid matrix with the i^{th} row, $\mathbf{U}_{i\cdot}$, the cluster centroid attached to point $\mathbf{X}_{i\cdot}$. The general loss $\ell(\mathbf{X}_{i\cdot}, \mathbf{U}_{i\cdot})$ refers to a general loss metric that measures dissimilarity between the data point $\mathbf{X}_{i\cdot}$ and assigned centroids $\mathbf{U}_{i\cdot}$. We choose one loss type as ℓ that is appropriate based on the data type of $\mathbf{X}$. For example, we use ℓ_1 loss in the presence of outliers. $\|\cdot\|_q$ is the ℓ_q-norm of a vector and usually $q \in \{1, 2, \infty\}$ is considered (Hocking et al., 2011). Here we prefer using the ℓ_2-norm in the fusion penalty ($q = 2$) as it encourages the entire rows of

similar observations to be fused together simultaneously and is also rotation-invariant; but one could use ℓ_1 or ℓ_∞-norm as well. γ is a positive tuning constant and w_{ij} is a nonnegative weight. When γ equals zero, each data point occupies a unique cluster. As γ increases, the fusion penalty encourages some of the rows of the cluster center $\mathbf{U}$ to be exactly fused, forming clusters. When γ becomes sufficiently large, all centroids eventually coalesce to a single cluster centroid, which we define as the loss-specific center associated with $\ell(\cdot)$. Hence γ regulates both the cluster assignment and number of clusters, providing a family of clustering solutions. The weight w_{ij} should be specified by the user in advance and is not a tuning parameter; we discuss choices of weights for various convex losses in Section 2.5.

Going beyond Euclidean distances, we propose to employ convex distance metrics as well as deviances associated with exponential family distributions and Bregman divergences, which are always convex. Interestingly, we show that each of these possible loss functions shrinks the cluster centers, $\mathbf{U}$, to different loss-specific centers, instead of the mean-based centroid as in convex clustering with Euclidean distances. For example, one may want to use least absolute deviations (ℓ_1-norm or Manhattan distances) for skewed data or for data with outliers; with this loss, we show that all observations fuse to the median when γ is sufficiently large. We emphasize loss-specific centers here as they will be important in feature selection in the next section. For completeness, we list common distances and deviance-based losses, as well as their loss-specific centers $\tilde{x}_j$ respectively in Table 1. (See Appendix F for all calculations associated with loss-specific centers, and we provide a formal proof when studying the properties of our approaches in Section 2.4.)

2.2 Integrative Generalized Convex Clustering (iGecco)

In data integration problems, we observe data from multiple sources and would like to get a holistic understanding of the problem by analyzing all the data simultaneously. In our framework, we integrate mixed multi-view data and perform clustering by employing different convex losses for each of the different data views with a joint convex fusion penalty that leads to common groups. Hence we propose Integrative Generalized Convex Clustering (iGecco) which can be formulated as follows:

$$\underset{\mathbf{U}^{(k)}}{\text{minimize}} \sum_{k=1}^{K} \pi_k \ell_k\left(\mathbf{X}^{(k)}, \mathbf{U}^{(k)}\right) + \gamma \sum_{i<i'} w_{ii'} \sqrt{\sum_{k=1}^{K} \left\| \mathbf{U}_{i\cdot}^{(k)} - \mathbf{U}_{i'\cdot}^{(k)} \right\|_2^2}.$$

Here, we have K data sources. The k^{th} data view $\mathbf{X}^{(k)}$ is an $n \times p_k$ matrix consisting of n observations and p_k features; $\mathbf{U}^{(k)}$ is also an $n \times p_k$ matrix and the i^{th} row, $\mathbf{U}_{i\cdot}^{(k)}$, is the cluster center associated with the point $\mathbf{X}_{i\cdot}^{(k)}$. And, $\ell_k(\mathbf{X}_{i\cdot}^{(k)}, \mathbf{U}_{i\cdot}^{(k)})$ is the loss function associated with the k^{th} data view. Still, we choose one loss type as ℓ_k that is appropriate based on the data type of each view. Each loss function is weighted by π_k, which is fixed by the user in advance. We have found that setting π_k to be inversely proportional to the null deviance evaluated at the loss-specific center, i.e., $\pi_k = \frac{1}{\ell_k(\mathbf{X}^{(k)}, \tilde{\mathbf{X}}^{(k)})}$, performs

well in practice. The null deviance, $\ell_k(\mathbf{X}^{(k)}, \widetilde{\mathbf{X}}^{(k)})$ refers to the loss function evaluated at $\widetilde{\mathbf{X}}^{(k)}$ where each j^{th} column of $\widetilde{\mathbf{X}}^{(k)}$ denotes the loss-specific center $\tilde{x}_j^{(k)}$. We employ this loss function weighting scheme to ensure equal scaling across data sets of different types. Recall that in generalized linear model (GLM), the likelihood-ratio test statistic, or the difference between the log-likelihoods, $2\left(\ell_k(\mathbf{X}^{(k)}, \widehat{\mathbf{U}}^{(k)}) - \ell_k(\mathbf{X}^{(k)}, \mathbf{U}_0^{(k)})\right)$, follows a χ^2-distribution. Here $\widehat{\mathbf{U}}^{(k)}$ is our iGecco estimate while $\mathbf{U}_0^{(k)}$ is the loss-specific center $\widetilde{\mathbf{X}}^{(k)}$ (cluster centroid when there is only one cluster). Therefore, the ratio of the two quantities, i.e., $\frac{\ell_k(\mathbf{X}^{(k)}, \mathbf{U}^{(k)})}{\ell_k(\mathbf{X}^{(k)}, \widetilde{\mathbf{X}}^{(k)})} = \pi_k \ell_k(\mathbf{X}^{(k)}, \mathbf{U}^{(k)})$ should be the same scale for each data view k. Finally, notice that we employ a joint convex fusion penalty on all of the $\mathbf{U}^{(k)}$'s; this incorporates information from each of the data sources and enforces the same group structure amongst the shared observations. Similar to convex clustering, our joint convex fusion penalty encourages the differences in the rows of concatenated centroids $\left(\mathbf{U}^{(1)} \ldots \mathbf{U}^{(K)}\right)$ to be shrunk towards zero, inducing a clustering behavior. Specifically, it forces the group structure of the i^{th} row of $\mathbf{U}^{(k)}$ to be the same for all k data views. Note the joint convex fusion penalty can be also written as $\left\| \begin{bmatrix} \mathbf{U}_{i\cdot}^{(1)} \\ \vdots \\ \mathbf{U}_{i\cdot}^{(K)} \end{bmatrix} - \begin{bmatrix} \mathbf{U}_{i'\cdot}^{(1)} \\ \vdots \\ \mathbf{U}_{i'\cdot}^{(K)} \end{bmatrix} \right\|_2$. We say that subject i and i' belong to the same cluster if $\mathbf{U}_{i\cdot}^{(K)} = \mathbf{U}_{i'\cdot}^{(K)}$, for all k. Hence, due to this joint convex fusion penalty, the common group structure property always holds. We study our methods further and prove some properties in Section 2.4.

2.3 Feature Selection: Gecco+ and iGecco+

In high dimensions, it is important to perform feature selection both for clustering purity and interpretability. Recently, Wang et al. (2018) proposed sparse convex clustering by imposing a group-lasso-type penalty on the cluster centers which achieves feature selection by shrinking noise features towards zero. This penalty, however, is only appropriate for Euclidean distances when the data is centered; otherwise, the penalty term shrinks towards the incorrect cluster centers. For example, the median is the cluster center with the ℓ_1 or Manhattan distances. Thus, to select features in this scenario, we need to shrink them towards the median, and we should enforce "sparsity" with respect to the median and not the origin. To address this, we propose adding a shifted group-lasso-type penalty which forces cluster center $\mathbf{U}_{\cdot j}$ to shrink towards the appropriate loss-specific center $\tilde{x}_j$ for each feature. Both the cluster fusion penalty and this new shifted-group-lasso-type feature selection penalty will shrink towards the same loss-specific center.

To facilitate feature selection with the adaptive shifted group-lasso penalty for one data type, our Generalized Convex Clustering Optimization with Feature Selection (Gecco+) is formulated as follows:

$$\underset{\mathbf{U}}{\text{minimize}} \sum_{i=1}^{n} \ell\left(\mathbf{X}_{i|}, \mathbf{U}_{i|}\right) + \gamma \sum_{i<i'}^{n} \omega_{ii'} \|\mathbf{U}_{i|} - \mathbf{U}_{i'|}\|_2 + \alpha \sum_{j=1}^{p} \zeta_j \|\mathbf{U}_{|j} - \tilde{x}_j\|_{\mathbf{1}_n}.$$

Again, $\mathbf{U}$ is an $n \times p$ matrix and $\tilde{x}_j$ is the loss-specific center for the j^{th} feature introduced in Table 1. The tuning parameter α controls the number of informative features and the feature weight ζ_j is a user input which plays an important role to adaptively penalize the features. (We discuss choices of ζ_j in Section 2.5.2 when we introduce the adaptive version of our method.) When α is small, all features are selected and contribute to defining the cluster centers. When α grows sufficiently large, all features coalesce at the same value, the loss-specific center $\tilde{x}_j$, and hence no features are selected and contribute towards determining the clusters. Another way of interpreting this is that the fusion penalty exactly fuses some of the rows of the cluster center $\mathbf{U}$, hence determining groups of rows. On the other hand, the shifted group-lasso penalty shrinks whole columns of $\mathbf{U}$ towards their loss-specific centers, thereby essentially removing the effect of uninformative features. Selected features are then columns of $\mathbf{U}$ that are not shrunken to their loss-specific centers, $\mathbf{U}_{\cdot j} \neq \tilde{x}_j \cdot \mathbf{1}_n$. These selected features, then, exhibit differences across the clusters determined by the fusion penalty. Clearly, sparse convex clustering of Wang et al. (2018) is a special case of Gecco+ using Euclidean distances with centered data. Our approach using both a row and column penalty is also reminiscent of convex biclustering (Chi et al., 2017) which uses fusion penalties on both the rows and columns to achieve checker-board-like biclusters.

Building upon integrative generalized convex clustering in Section 2.2 and our proposed feature selection penalty above, our Integrative Generalized Convex Clustering Optimization with Feature Selection (iGecco+) is formulated as follows:

$$\begin{aligned} \underset{\mathbf{U}^{(k)}}{\text{minimize}} & \sum_{k=1}^{K} \pi_k \ell_k(\mathbf{X}^{(k)}, \mathbf{U}^{(k)}) + \gamma \sum_{i<i'} w_{ii'} \sqrt{\sum_{k=1}^{K} \|\mathbf{U}_{i|}^{(k)} - \mathbf{U}_{i'|}^{(k)}\|_2^2} \\ & + \alpha \sum_{k=1}^{K} \sum_{j=1}^{p_k} \zeta_j^{(k)} \|\mathbf{U}_{|j}^{(k)} - \tilde{x}_j^{(k)}\|_{\mathbf{1}_n}. \end{aligned} \tag{1}$$

Again, $\mathbf{U}^{(k)}$ is an $n \times p_k$ matrix and $\tilde{x}_j^{(k)}$ is the loss-specific center for the j^{th} feature for k^{th} data view. By construction, iGecco+ directly clusters mixed multi-view data and selects features from each data view simultaneously. Similarly, adaptive choice of $\zeta_j^{(k)}$ gives rise to adaptive iGecco+ which will be discussed in Section 2.5.2. Detailed discussions on practical choices of tuning parameters and weights can be also found in Section 2.5.

2.4 Properties

In this section, we develop some properties of our methods, highlighting several advantages of our convex formulation. Corresponding proofs can be found in Section A of the Appendix.

Define the objective function in (1) as $F_{\gamma,\alpha}(\mathbf{U})$ where $\mathbf{U} = \left(\mathbf{U}^{(1)} \ldots \mathbf{U}^{(K)}\right)$. Then due to convexity, we have the following properties. First, any minimizer achieves a global solution.

Proposition 1 *(Global solution) If ℓ_k is convex for all k, then any minimizer of $F_{\gamma,\alpha}(\mathbf{U})$, $\mathbf{U}^*$, is a global minimizer. If ℓ_k is strictly convex for all k, then $\mathbf{U}^*$ is unique.*

Our method is continuous with respect to data, tuning parameters and input parameters.

Proposition 2 *(Continuity with respect to data, tuning and input parameters) The global minimizer $\mathbf{U}^*_{w,\pi,\zeta,\mathbf{X}}(\gamma,\alpha)$ of iGecco+ exists and depends continuously on the data, $\mathbf{X}$, tuning parameters γ and α, the weight matrix w, the loss weight π_k, and the feature weight $\zeta_j^{(k)}$.*

When tuning parameters are sufficiently large, all $\mathbf{U}$'s coalesce to the loss-specific centers.

Proposition 3 *(Loss-specific center) Define $\widetilde{\mathbf{X}} = \left(\widetilde{\mathbf{X}}^{(1)} \cdots \widetilde{\mathbf{X}}^{(K)}\right)$ where each j^{th} column of $\widetilde{\mathbf{X}}^{(k)}$ equals the loss-specific center $\tilde{x}_j^{(k)}$. Suppose each observation corresponds to a node in a graph with an edge between nodes i and j whenever $w_{ij} > 0$. If this graph is fully connected, then $F_{\gamma,\alpha}(\mathbf{U})$ is minimized by the loss-specific center $\widetilde{\mathbf{X}}$ when γ is sufficiently large or α is sufficiently large.*

Remark. As Gecco, Gecco+ and iGecco are special cases of iGecco+, it is easy to show that all of our properties hold for these methods as well.

These properties illustrate some important advantages of our convex clustering approaches. Specifically, many other widely used clustering methods are known to suffer from poor local solutions, but any minimizer of our problem will achieve a global solution. Additionally, we show that iGecco+ is continuous with respect to the data, tuning parameters, and other input parameters. Together, these two properties are very important in practice and illustrate that the global solution of our method remains stable to small perturbations in the data and input parameters. Stability is a desirable property in practice as one would question the validity of a clustering result that can change dramatically with small changes to the data or parameters. Importantly, most popular clustering methods such as k-means, hierarchical clustering, model-based clustering, or low-rank based clustering, do not enjoy these same stability properties.

Finally in Proposition 3, we verify that when the tuning parameters are sufficiently large, full fusion of all observations to the loss-specific centers is achieved. Hence, our methods indeed behave as intended, achieving joint clustering of observations. We illustrate this property in Figure 1 where we apply Gecco+ to the authors data set (described fully in Section 4). Here, we illustrate how our solution, $\widehat{\mathbf{U}}(\gamma,\alpha)$, changes as a function of γ and α. This so-called "cluster solution path" begins with each observation as its own cluster center when γ is small and stops when all observations are fused to the loss-specific center when γ is sufficiently large. In between, we see that observations are fusing together as γ increases.

Similarly, when α is small, all features are selected and as α increases, some of the features get fused to their loss-specific center.

2.5 Practical Considerations and Adaptive iGecco+

In this section, we discuss some practical considerations for applying our method to real data. In the iGecco+ problem, π_k, w and ζ_j are user-specific fixed inputs while γ and α are tuning parameters; γ controls the number of clusters while α controls the number of features selected. We discuss choosing user-specific inputs such as weights as well as how to select tuning parameters. In doing so, we introduce an adaptive version of our method as well.

2.5.1 Choice of Weights and Tuning Parameters—In practice, a good choice of fusion weights w_{ij} has been shown to enhance both computational efficiency and clustering quality of convex clustering (Chi and Lange, 2015). It has been empirically demonstrated that using weights inversely proportional to the distances yields superior clustering performance; this approach is widely adopted in practice. Further, setting many of the weights to zero helps reduce computation cost. Considering these two, the most common weights choice for convex clustering is to use K-nearest-neighbors method with a Gaussian kernel. Specifically, the weight between the sample pair (i, j) is set as $w_{ij} = I_{ij}^{k}\exp\left(-\phi d\left(\mathbf{X}_{i\cdot}, \mathbf{X}_{j\cdot}\right)\right)$, where I_{ij}^{k} equals 1 if observation j is among observation i's κ nearest neighbors or vice versa, and 0 otherwise. However, this choice of weights based on Euclidean distances may not work well for non-Gaussian data in Gecco(+) or for mixed data in iGecco(+). To account for different data types and better measure the similarity between observations, we still adopt K-nearest-neighbors method with an exponential kernel, but further extend this by employing appropriate distance metrics for specific data types in the exponential kernel. In particular, for weights in Gecco and Gecco+, we suggest using the same distance functions or deviances in the loss function of Gecco and Gecco+. For weights in iGecco and iGecco+, the challenge is that we need to employ a distance metric which measures mixed types of data. In this case, the Gower distance, which is a distance metric used to measure the dissimilarity of two observations measured in different data types (Gower, 1971), can address our problem. To be specific, the Gower distance between observation i and i' overall is defined as $d(\mathbf{X}_{i\cdot}, \mathbf{X}_{i'\cdot}) = \sum_{k=1}^{K}\sum_{j=1}^{p_k} d_{ii'j}^{(k)} \Big/ \sum_{k=1}^{K} p_k$ where $d_{ii'j}^{(k)} = \frac{|\mathbf{X}_{ij}^{(k)} - \mathbf{X}_{i'j}^{(k)}|}{R_j^{(k)}}$ refers to the Gower distance between observation i and i' for feature j in data view k and $R_j^{(k)} = \max_{i,i'}|\mathbf{X}_{ij}^{(k)} - \mathbf{X}_{i'j}^{(k)}|$ is the range of feature j in data view k. In the literature, Gower distance has been commonly used as distance metrics for clustering mixed types of data (Wangchamhan et al., 2017; Hummel et al., 2017; Akay and Yüksel, 2018) and shown to yield superior performance than other distance metrics (Ali and Massmoudi, 2013; dos Santos and Zárate, 2015).

Alternatively, we also propose and explore using stochastic neighbor embedding weights based on symmetrized conditional probabilities (Maaten and Hinton, 2008). These have been shown to yield superior performance in high-dimensions and if there are potential outliers. Specifically, the symmetrized conditional probabilities are defined

as $p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n}$, where $p_{j|i} = \frac{\exp(-\phi d(\mathbf{X}_{i.}, \mathbf{X}_{j.}))}{\sum_{k \neq i} \exp(-\phi d(\mathbf{X}_{i|}, \mathbf{X}_{k|}))}$. We propose to use the weights $w_{ij} = I_{ij}^{k} \cdot p_{ij}$ where I_{ij}^{k} still equals 1 if observation j is among observation i's κ nearest neighbors or vice versa, and 0 otherwise. Again, we suggest using distance metrics appropriate for specific data types or the Gower distance for mixed data. In empirical studies, we experimented with both weight choices and found that stochastic neighbor embedding weights tend to work better in high-dimensional settings and if there are outliers. Hence, we recommend these and employed them in our empirical investigations in Section 4 and 5.

Estimating the number of clusters in a data set is always challenging. Going beyond, we have two tuning parameters in our iGecco+ problem; γ controls the number of clusters while α controls the number of features selected. Current literature for tuning parameter selection for convex clustering mainly focuses on stability selection (Wang, 2010; Fang and Wang, 2012), hold-out validation (Chi et al., 2017) and information criterion (Tan and Witten, 2015). We first adopt the stability selection based approach for tuning parameter selection and follow closely the approach described in the work of Wang (2010); Fang and Wang (2012). We choose stability selection based approach because i) its selection consistency has been established and ii) Wang et al. (2018) adopted similar approach for tuning parameter selection and demonstrated strong empirical performance. However, stability selection is often computationally intensive in practice. To address this, we further explore information criterion based approaches like the Bayesian information criterion (BIC). We explain full details of both approaches in Appendix J and demonstrate empirical results when the number of clusters and features are not fixed but estimated based on the data.

2.5.2 Adaptive Gecco+ and iGecco+ to Weight Features—Finally, we consider how to specify the feature weights, ζ_j used in the shifted group-lasso penalty. While employing these weights are not strictly necessary, we have found, as did Wang et al. (2018), that like the fusion weights, well-specified ζ_j's can both improve performance and speed up computation. But unlike the fusion weights where we can use the pairwise distances, we don't have prior information on which features may be relevant in clustering. Thus, we propose to use an adaptive scheme that first fits the iGecco+ with no feature weights and uses this initial estimate to define feature importance for use in weights. This is similar to many adaptive approaches in the literature (Zou, 2006; Wang et al., 2018).

Our adaptive iGecco+ approach is given in Algorithm 1; this applies to adaptive Gecco+ as a special case as well. We assume that the number of clusters (or a range of the number of clusters) is known a priori. We begin by fitting iGecco+ with $\alpha = 1$ and uniform feature weights $\zeta_j^{(k)} = 1$. We then find the γ which gives the desired number of clusters, yielding the initial estimate, $\widehat{\mathbf{U}}^{(k)}$. (We provide alternative adaptive iGecco+ Algorithm 17 when the number of clusters is not known in Appendix J.) Next, similar to the adaptive approaches by Zou (2006); Wang et al. (2018), we use this initial estimate to adaptively weight features by proposing the following weights: $\zeta_j^{(k)} = 1/\|\widehat{\mathbf{U}}_{.j}^{(k)} - \tilde{x}_j^{(k)} \cdot \mathbf{1}_n\|_2$. (To avoid numerical issues, we add $\epsilon = 0.01$ to the denominator.) These weights place a large penalty on noise features as

$\|\widehat{\mathbf{U}}_{\cdot j}^{(k)} - \tilde{x}_j^{(k)} \cdot \mathbf{1}_n\|_2$ is close to zero in this case. Note, compared with sparse convex clustering where the authors defined feature weights ζ_j by solving a convex clustering problem with feature penalty $\alpha = 0$, we propose to fit iGecco+ with feature penalty $\alpha = 1$ first and then update the feature weights adaptively. We find this weighting scheme works well in practice as it shrinks noise features more and hence penalizes more on those features. Such with-penalty initialization for adaptive weights has also been proposed in literature (Zhou et al., 2009; Fan et al., 2009; van de Geer et al., 2011). We also notice that noise features impact the distances used in the fusion weights as well. Hence, we suggest updating the distances adaptively by using the selected features to better measure the similarities between observations. To this end, we propose a new scheme to compute weighted Gower distances. First, we scale the features within each data view so that informative features in different data views contribute equally and on the same scale. Then, we employ the inverse of π_k, i.e., the null deviance, to weight the distances from different data types, resulting in an aggregated and weighted Gower distance, $\hat{d}(\mathbf{X}_{i\cdot}, \mathbf{X}_{i'\cdot})$ as further detailed in Algorithm 1. Note that if the clustering signal from one particular data type is weak and there are few informative features for this data type, then our weighting scheme will down-weight this entire data type in the weighted Gower distance. In practice, our adaptive iGecco+ scheme works well as evidenced in our empirical investigations in the next sections.

Algorithm 1

Adaptive iGecco+

1. Fit iGecco+ with $\alpha = 1, \zeta_j^{(k)} = 1$ and a sequence of γ.

2. Find γ which gives desired number of clusters; Get the estimate $\widehat{\mathbf{U}}^{(k)}$.

3. Update the feature weights $\hat{\zeta}_j^{(k)} = 1/\|\widehat{\mathbf{U}}_{\cdot j}^{(k)} - \tilde{x}_j^{(k)} \cdot \mathbf{1}_n\|_2$
and fusion weights $\widehat{w}_{ij} = I_{ij}^K \exp(-\phi \hat{d}(\mathbf{X}_{i\cdot}, \mathbf{X}_{i'\cdot}))$, where

$$\hat{d}(\mathbf{X}_{i\cdot}, \mathbf{X}_{i'\cdot}) = \sum_{k=1}^{K} \sum_{j=1}^{p_k} \frac{\|\widehat{\mathbf{U}}_{\cdot j}^{(k)} - \tilde{x}_j^{(k)}\mathbf{1}_n\|_2}{\max_j \|\widehat{\mathbf{U}}_{\cdot j}^{(k)} - \tilde{x}_j^{(k)}\mathbf{1}_n\|_2} \cdot \frac{1}{\pi_k} \cdot d_{ii'j}^{(k)}.$$

4. Fit adaptive iGecco+ with $\hat{\zeta}$, $\widehat{w}$ and a sequence of γ and α; Find optimal γ and α which give desired number of clusters and features.

Note that Algorithm 1 for adaptive iGecco+ assumes desired number of clusters and features. (Panahi et al. (2017); Sun et al. (2021) proved that perfect recovery can be guaranteed for the general case of q-clusters for convex clustering. To yield exact q desired number of clusters, Weylandt et al. (2020) suggested back-tracking in practice.) We provide the alternative adaptive iGecco+ (Algorithm 17) in Appendix J when the number of clusters or features are not known a priori but estimated from the data.

3. iGecco+ Algorithm

In this section, we introduce our algorithm to solve iGecco+, which can be easily extended to Gecco, Gecco+ and iGecco. We first propose a simple, but rather slow ADMM algorithm

as a baseline approach. To save computation cost, we further develop a new multi-block ADMM-type procedure using inexact one-step approximation of the sub-problems. Our algorithm is novel from optimization perspective as we extend the multi-block ADMM to a higher number of blocks and combine it with the literature related to inexact-solve ADMM with sub-problem approximations, which often results in major computational savings.

3.1 Full ADMM to Solve iGecco+ (Naive Algorithm)

Given the shifted group-lasso and fusion penalties along with general losses, developing an optimization routine for iGecco+ method is less straight-forward than convex clustering or sparse convex clustering. In this section, we propose a simple ADMM algorithm to solve iGecco+ as a baseline algorithm and point out its drawbacks.

The most common approach to solve problems with more than two non-smooth functions is via multi-block ADMM (Lin et al., 2015; Deng et al., 2017), which decomposes the original problem into several smaller sub-problems and solves them in parallel at each iteration. Chen et al. (2016) established a sufficient condition for the convergence of three-block ADMM. We develop a multi-block ADMM approach to fit our problem for certain types of losses and prove its convergence.

We first recast iGecco+ problem (1) as the equivalent constrained optimization problem:

$$\underset{\mathbf{U}^{(k)},\mathbf{V}}{\text{minimize}} \sum_{k=1}^{K} \pi_k \ell_k(\mathbf{X}^{(k)}, \mathbf{U}^{(k)}) + \gamma \underbrace{\left(\sum_{l \in \varepsilon} w_l \|\mathbf{V}_l\|_2 \right)}_{P_1(\mathbf{V};\mathbf{w})} + \alpha \sum_{k=1}^{K} \sum_{j=1}^{p_k} \zeta_j^{(k)} \|\mathbf{U}_{\cdot j}^{(k)} - \tilde{x}_j^{(k)}\mathbf{1}_n\|_2$$

$$\text{subject to} \quad \mathbf{D}\left[\mathbf{U}^{(1)} \cdots \mathbf{U}^{(K)}\right] - \mathbf{V} = \mathbf{0}.$$

Recently, Weylandt et al. (2020) derived the ADMM for convex clustering in matrix form and we adopt similar approach. We index a centroid pair by $l = (l_1, l_2)$ with $l_1 < l_2$, define the set of edges over the non-zero weights $\varepsilon = \{l = (l_1, l_2) : w_l > 0\}$, and introduce a new variable $\mathbf{V} = \left[\mathbf{V}^{(1)} \quad \cdots \quad \mathbf{V}^{(K)}\right] \in \mathbb{R}^{|\varepsilon| \times \Sigma p_k}$ where $\mathbf{V}_{l\cdot}^{(k)} = \mathbf{U}_{l_1\cdot}^{(k)} - \mathbf{U}_{l_2\cdot}^{(k)}$ to account for the difference between the two centroids. Hence $\mathbf{V}^{(k)}$ is a matrix containing the pairwise differences between connected rows of $\mathbf{U}^{(k)}$ and the constraint is equivalent to $\mathbf{D}\mathbf{U}^{(k)} - \mathbf{V}^{(k)} = 0$ for all k; $\mathbf{D} \in \mathbb{R}^{|\varepsilon| \times n}$ is the directed difference matrix corresponding to the non-zero fusion weights. We give general-form multi-block ADMM (Algorithm 2) to solve iGecco+. Here $prox_{h(\cdot)}(\mathbf{x}) = \text{argmin}_{\mathbf{z}} \frac{1}{2}\|\mathbf{x} - \mathbf{z}\|_2^2 + h(\mathbf{z})$ is the proximal mapping of function h. Also, the superscript in $\mathbf{U}^{(k)}$ in Algorithm 2 refers to the k^{th} data view; we omit iteration counter indices in all iGecco+ algorithm for notation purposes and use the most recent values of the updates. The dual variable is denoted by $\mathbf{\Lambda}^{(k)}$.

Algorithm 2

General multi-block algorithm for iGecco+

while not converged **do**

for all $k = 1, \cdots, K$ **do**

$$\mathbf{U}^{(k)} = \underset{\mathbf{U}}{\operatorname{argmin}}\ \pi_k \ell_k(\mathbf{X}^{(k)}, \mathbf{U}) + \frac{\rho}{2}\|\mathbf{DU} - \mathbf{V}^{(k)} + \mathbf{\Lambda}^{(k)}\|_F^2 + \alpha \sum_{j=1}^{p_k} \zeta_j^{(k)} \|\mathbf{U}_{\cdot j} - \tilde{x}_j^{(k)} \mathbf{1}_n\|_2$$

end for

$$\mathbf{V} = \text{prox}_{\gamma/\rho P_1(\cdot\,; w)}([\mathbf{DU}^{(1)} + \mathbf{\Lambda}^{(1)} \cdots \mathbf{DU}^{(K)} + \mathbf{\Lambda}^{(K)}])$$

$$\mathbf{\Lambda}^{(k)} = \mathbf{\Lambda}^{(k)} + (\mathbf{DU}^{(k)} - \mathbf{V}^{(k)}) \quad \text{for all} k$$

end while

Notice that, in Algorithm 2, there is no closed-form analytical solution for the $\mathbf{U}^{(k)}$ sub-problem for general losses. Typically, at each iteration of Algorithm 2, we need to apply an inner optimization routine, which requires a nested iterative solver, to solve the $\mathbf{U}^{(k)}$ sub-problem until full convergence. In the next section, we seek to speed up this algorithm by using $\mathbf{U}^{(k)}$ sub-problem approximations. But, first we propose two different approaches to fully solve the $\mathbf{U}^{(k)}$ sub-problem based on specific loss types and then use these to develop a one-step update to solve the sub-problem approximately with guaranteed convergence. For the $\mathbf{V}$ sub-problem, one can easily show that it has a closed-form analytical solution for each iteration, as given in Algorithm 2.

3.2 iGecco+ Algorithm

We have introduced Algorithm 2, a simple baseline ADMM approach to solve iGecco+. In this section, we consider different ways to solve the $\mathbf{U}^{(k)}$ sub-problem in Algorithm 2. First, based on specific loss types (differentiable or non-differentiable), we propose two different algorithms to solve the $\mathbf{U}^{(k)}$ sub-problem to full convergence. These approaches, however, are rather slow for general losses as there is no closed-form solution which requires another nested iterative solver. To address this and in connection with current literature on variants of ADMM with sub-problem approximations, we propose iGecco+ algorithm, a multi-block ADMM algorithm which solves the sub-problems approximately by taking a single one-step update. We prove convergence of this general class of algorithms, a novel result in the optimization literature.

3.2.1 Differentiable Case—When the loss ℓ_k is differentiable, we consider solving the $\mathbf{U}^{(k)}$ sub-problem with proximal gradient descent, which is often used when the objective function can be decomposed into a differentiable and a non-differentiable function. While there are many other possible optimization routines to solve the $\mathbf{U}^{(k)}$ sub-problem, we choose proximal gradient descent as there is existing literature proving convergence of ADMM algorithms with approximately solved sub-problems using proximal gradient descent (Liu et al., 2013; Lu et al., 2016). We will discuss in detail how to approximately solve the sub-problem by taking a one-step approximation in Section 3.2.3. Based upon

this, we propose Algorithm 3, which solves the $\mathbf{U}^{(k)}$ sub-problem by running full iterative proximal gradient descent to convergence. Here $P_2(\widetilde{\mathbf{U}}^{(k)}; \zeta^{(k)}) = \sum_{j=1}^{p_k} \zeta_j^{(k)} \|\widetilde{\mathbf{U}}_{\cdot j}^{(k)}\|_2$.

Algorithm 3

$\mathbf{U}^{(k)}$ sub-problem for differentiable loss ℓ_k (proximal gradient):

while not converged **do**

$$\mathbf{U}^{(k)} = \text{prox}_{s_k \cdot \alpha P_2(\cdot; \zeta^{(k)})}(\mathbf{U}^{(k)} - \widetilde{\mathbf{X}}^{(k)} - s_k \cdot [\pi_k \nabla \ell_k(\mathbf{X}^{(k)}, \mathbf{U}^{(k)}) + \rho \mathbf{D}^T(\mathbf{D}\mathbf{U}^{(k)} - \mathbf{V}^{(k)} + \mathbf{\Lambda}^{(k)})]) + \widetilde{\mathbf{X}}^{(k)}$$

end while

In Algorithm 3 and typically in general (proximal gradient) descent algorithms, we need to choose an appropriate step size s_k to ensure convergence. Usually we employ a fixed step size by computing the Lipschitz constant as in the squared error loss case; but in our method, it is hard to compute the Lipschitz constant for most of our general losses. Instead, we suggest using backtracking line search procedure proposed by Beck and Teboulle (2009); Parikh et al. (2014), which is a common way to determine step size with guaranteed convergence in optimization. Further, we find decomposing the $\mathbf{U}^{(k)}$ sub-problem to p_k separate $\mathbf{U}_{\cdot j}^{(k)}$ sub-problems brings several advantages such as i) better convergence property (than updating $\mathbf{U}^{(k)}$'s all together) due to adaptive step size for each $\mathbf{U}_{\cdot j}^{(k)}$ sub-problem and ii) less computation cost by solving each in parallel. Hence, in this case, we propose to use proximal gradient for each separate $\mathbf{U}_{\cdot j}^{(k)}$ sub-problem. To achieve this, we assume that the loss is elementwise, which is satisfied by every deviance-based loss. Last, as mentioned, there are many other possible ways to solve the $\mathbf{U}^{(k)}$ sub-problem than proximal gradient, such as ADMM. We find that when the loss function is squared Euclidean distances or the loss function has a Hessian matrix that can be upper bounded by a fixed matrix, using ADMM approach saves more computation. We provide all implementation details discussed above in Section C of the Appendix.

3.2.2 Non-differentiable Case—When the loss ℓ_k is non-differentiable, we can no longer adopt the proximal gradient method to solve the $\mathbf{U}^{(k)}$ sub-problem as the objective is now a sum of more than one separable non-smooth function. To address this, as mentioned, we can use multi-block ADMM; in this case, we introduce new blocks for the non-smooth functions and hence develop a full three-block ADMM approach to fit our problem.

To augment the non-differentiable term, we assume that our loss function can be written as $\ell_k(\mathbf{X}^{(k)}, \mathbf{U}^{(k)}) = f_k(g_k(\mathbf{X}^{(k)}, \mathbf{U}^{(k)}))$ where f_k is convex but non-differentiable and g_k is affine. This condition is satisfied by all distance-based losses with $g_k(\mathbf{X}^{(k)}, \mathbf{U}^{(k)}) = \mathbf{X}^{(k)} - \mathbf{U}^{(k)}$; for example, for Manhattan distances, we have $f_k(\mathbf{Z}) = \sum_{j=1}^{p} \|\mathbf{z}_j\|_1 = \|\text{vec}(\mathbf{Z})\|_1$, and $g_k(\mathbf{X}, \mathbf{U}) = \mathbf{X} - \mathbf{U}$. The benefit of doing this is that now the $\mathbf{U}^{(k)}$ sub-problem has a closed-form solution. Particularly, we can rewrite the $\mathbf{U}^{(k)}$ sub-problem as:

$$\underset{\mathbf{U}^{(k)},\mathbf{V}}{\text{minimize}} \sum_{k=1}^{K} \pi_k f_k(\mathbf{Z}^{(k)}) + \frac{\rho}{2}\|\mathbf{D}\mathbf{U} - \mathbf{V}^{(k)} + \mathbf{\Lambda}^{(k)}\|_F^2 + \alpha \sum_{k=1}^{K} \underbrace{\left(\sum_{j=1}^{p_k} \zeta_j^{(k)} \|\mathbf{r}_j^{(k)}\|_2\right)}_{P_2(\mathbf{R}^{(k)}, \zeta^{(k)})}$$

$$\text{subject to} \quad \mathbf{X}^{(k)} - \mathbf{U}^{(k)} = \mathbf{Z}^{(k)}, \quad \mathbf{U}^{(k)} - \widetilde{\mathbf{X}}^{(k)} = \mathbf{R}^{(k)},$$

where $\widetilde{\mathbf{X}}^{(k)}$ is an $n \times p_k$ matrix with the j^{th} column equal to scalar $\tilde{x}_j^{(k)}$.

It is clear that we can use multi-block ADMM to solve the problem above and each primal variable has a simple update with a closed-form solution. We propose Algorithm 4, a full, iterative multi-block ADMM, to solve the $\mathbf{U}^{(k)}$ sub-problem when the loss is a non-differentiable distance-based function. Algorithm 4 applies to iGecco+ with various distances such as Manhattan, Minkowski and Chebychev distances and details are given in Section D of the Appendix.

Algorithm 4

$\mathbf{U}^{(k)}$ sub-problem for non-differentiable distance-based loss ℓ_k (multi-block ADMM):

Precompute: Difference matrix $\mathbf{D}, \mathbf{M} = (\mathbf{D}^T\mathbf{D} + 2\mathbf{I})^{-1}$.

while not converged **do**

$$\mathbf{U}^{(k)} = \mathbf{M}(\mathbf{D}^T(\mathbf{V}^{(k)} - \mathbf{\Lambda}^{(k)}) + \widetilde{\mathbf{X}}^{(k)} + \mathbf{R}^{(k)} - \mathbf{N}^{(k)} + \mathbf{X}^{(k)} - \mathbf{Z}^{(k)} + \mathbf{\Psi}^{(k)})$$

$$\mathbf{Z}^{(k)} = \text{prox}_{\pi_k f_k/\rho}(\mathbf{X}^{(k)} - \mathbf{U}^{(k)} + \mathbf{\Psi}^{(k)})$$

$$\mathbf{R}^{(k)} = \text{prox}_{\alpha/\rho P_2(\cdot;\zeta^{(k)})}(\mathbf{U}^{(k)} - \widetilde{\mathbf{X}}^{(k)} + N^{(k)})$$

$$\mathbf{\Psi}^{(k)} = \mathbf{\Psi}^{(k)} + (\mathbf{X}^{(k)} - \mathbf{U}^{(k)} - \mathbf{Z}^{(k)})$$

$$\mathbf{N}^{(k)} = \mathbf{N}^{(k)} + (\mathbf{U}^{(k)} - \widetilde{\mathbf{X}}^{(k)} - R^{(k)})$$

end while

3.2.3 IGECCO+ ALGORITHM: FAST ADMM WITH INEXACT ONE-STEP APPROXIMATION TO THE SUB-PROBLEM—

Notice that for both Algorithm 3 and 4, we need to run them iteratively to full convergence in order to solve the $\mathbf{U}^{(k)}$ sub-problem in Algorithm 2 for each iteration, which is dramatically slow in practice. To address this, in literature, many have proposed variants of ADMM with guaranteed convergence that find an inexact, one-step, approximate solution to the sub-problem (without fully solving it); these include the generalized ADMM (Deng and Yin, 2016), proximal ADMM (Shefi and Teboulle, 2014; Banert et al., 2016) and proximal linearized ADMM (Liu et al., 2013; Lu et al., 2016). Thus, we propose to solve the $\mathbf{U}^{(k)}$ sub-problem approximately by taking a single one-step update of the algorithm (Algorithm 3 or 4) for both types of losses and prove convergence. For the differentiable loss case, we propose to apply the proximal linearized ADMM approach while for the non-differentiable case, we show that taking a one-step update of Algorithm 4, along with $\mathbf{V}$ and $\mathbf{\Lambda}$ update in Algorithm 2, is equivalent to applying a four-block ADMM to the original

iGecco+ problem and we provide a sufficient condition for the convergence of four-block ADMM. Our algorithm, to the best of our knowledge, is the first to incorporate higher-order multi-block ADMM and inexact ADMM with a one-step update to solve sub-problems for general loss functions.

When the loss is differentiable, as mentioned in Algorithm 3, one can use full iterative proximal gradient to solve the $\mathbf{U}_{\cdot j}^{(k)}$ sub-problem, which however, is computationally burdensome. To avoid this, many proposed variants of ADMM which find approximate solutions to the sub-problems. Specifically, closely related to our problem here, Liu et al. (2013); Lu et al. (2016) proposed proximal linearized ADMM which solves the sub-problems efficiently by linearizing the differentiable part and then applying proximal gradient due to the non-differentiable part. We find their approach fits into our problem and hence develop a proximal linearized 2-block ADMM to solve iGecco+ when the loss ℓ_k is differentiable and gradient is Lipschitz continuous. It can be shown that applying proximal linearized 2-block ADMM to Algorithm 2 is equivalent to taking a one-step update of Algorithm 3 along with $\mathbf{V}$ and $\mathbf{\Lambda}$ update in Algorithm 2. In this way, we avoid running full iterative proximal gradient updates to convergence for the $\mathbf{U}^{(k)}$ sub-problem as in Algorithm 3 and hence save computation cost.

When the loss is non-differentiable, we still seek to take an one-step update to solve the $\mathbf{U}^{(k)}$ sub-problem. We achieve this by noticing that taking a one-step update of Algorithm 4 along with $\mathbf{V}$ and $\mathbf{\Lambda}$ update in Algorithm 2 is equivalent to applying multi-block ADMM to the original iGecco+ problem recast as follows (for non-differentiable distance-based loss):

$$\begin{aligned}&\underset{\mathbf{U}^{(k)},\mathbf{V}}{\text{minimize}} \quad \sum_{k=1}^{K}\pi_k f_k(\mathbf{Z}^{(k)}) + \underbrace{\gamma\left(\sum_{l\in\varepsilon} w_l\|\mathbf{V}_l\|_2\right)}_{P_1(\mathbf{V};\mathbf{w})} + \alpha\sum_{k=1}^{K}\underbrace{\left(\sum_{j=1}^{p_k}\zeta_j^{(k)}\|\mathbf{r}_j^{(k)}\|_2\right)}_{P_2\left(\mathbf{R}^{(k)};\zeta^{(k)}\right)}\\ &\text{subject to} \quad \mathbf{X}^{(k)}-\mathbf{U}^{(k)}=\mathbf{Z}^{(k)}, \quad \mathbf{D}\,[\mathbf{U}^{(1)} \quad \cdots \quad \mathbf{U}^{(K)}]-\mathbf{V}=\mathbf{0}, \quad \mathbf{U}^{(k)}-\widetilde{\mathbf{X}}^{(k)}=\mathbf{R}^{(k)}.\end{aligned}$$

Typically, general higher-order multi-block ADMM algorithms do not always converge, even for convex functions (Chen et al., 2016). We prove convergence of our algorithm and establish a novel convergence result by casting the iGecco+ with non-differentiable losses as a four-block ADMM, proposing a sufficient condition for convergence of higher-order multi-block ADMMs, and finally showing that our problem satisfies this condition. (Details are given in the proof of Theorem 4 in Appendix B.) Therefore, taking a one-step update of Algorithm 4 converges for iGecco+ with non-differentiable losses.

So far, we have proposed inexact-solve one-step update approach for both differentiable loss and non-differentiable loss case. For mixed type of losses, we combine these two algorithms and this gives Algorithm 5, a multi-block ADMM algorithm with inexact one-step approximation to the $\mathbf{U}^{(k)}$ sub-problem to solve iGecco+. We also establish the following convergence result.

Algorithm 5

iGecco+ algorithm

while not converged **do**

  for all $k = 1, \cdots, K$ **do**

    Update $\mathbf{U}^{(k)}$:

    if ℓ_k is differentiable **then**

      Take a one-step update of Algorithm 3

    else if ℓ_k is non-differentiable **then**

      Take a one-step update of Algorithm 4

    end if

  end for

$$\mathbf{V} = \text{prox}_{\gamma/\rho P_1(\cdot\,; w)}\left(\left[\mathbf{D}\mathbf{U}^{(1)} + \mathbf{\Lambda}^{(1)} \quad \cdots \quad \mathbf{D}\mathbf{U}^{(K)} + \mathbf{\Lambda}^{(K)}\right]\right)$$

$$\mathbf{\Lambda}^{(k)} = \mathbf{\Lambda}^{(k)} + \left(\mathbf{D}\mathbf{U}^{(k)} - \mathbf{V}^{(k)}\right) \quad \text{for all } k$$

end while

Theorem 4 *(iGecco+ convergence) Consider the iGecco+ problem* (1) *with fixed inputs* π_k, w *and* ζ_j. *If* ℓ_k *is convex for all* k, Algorithm 5 *converges to a global solution. In addition, if each* ℓ_k *is strictly convex, it converges to the unique global solution.*

Remark. Our corresponding Theorem 4 establishes a novel convergence result as it is the first to show the convergence of four-block or higher ADMM using approximate sub-problems for both differentiable and non-differentiable losses.

It is easy to see that Algorithm 5 can be applied to solve other Gecco-related methods as special cases. When $K = 1$, Algorithm 5 gives the algorithm to solve Gecco+. When $\alpha = 0$, Algorithm 5 gives the algorithm to solve iGecco+. When $K = 1$ and $\alpha = 0$, Algorithm 5 gives the algorithm to solve Gecco.

To conclude this section, we compare the convergence results of using both full ADMM and inexact ADMM with one-step update in the sub-problem to solve Gecco+ ($n = 120$ and $p = 210$) in Figure 2. The left plots show the number of iterations needed to yield optimization convergence while the right plots show computation time. We see that Algorithm 5 (one-step update to solve the sub-problem) saves much more computational time than Algorithm 2 (full updates to solve the sub-problem). It should be pointed out that though Algorithm 5 takes more iterations to converge due to inexact approximation for each iteration, we still reduce computation time dramatically as the computation time per iteration is much less than the full-solve approach.

4. Simulation Studies

In this section, we first evaluate the performance of Gecco+ against existing methods on non-Gaussian data. Next we compare iGecco+ with other methods on mixed multi-view data.

4.1 Non-Gaussian Data

In this subsection, we evaluate the performance of Gecco and (adaptive) Gecco+ by comparing it with k-means, hierarchical clustering and sparse convex clustering. For simplicity, we have the following naming convention for all methods: loss type name + Gecco(+). For example, Poisson Deviance Gecco+ refers to Generalized Convex Clustering with Feature Selection using Poisson deviance. Sparse CC refers to sparse convex clustering using Euclidean distances where each feature is centered first. We measure the accuracy of clustering results using adjusted Rand index (Hubert and Arabie, 1985). The adjusted Rand index is the corrected-for-chance version of the Rand index, which is used to measure the agreement between the estimated clustering assignment and the true group label. A larger adjusted Rand index implies a better clustering result. For all methods we consider, we assume oracle number of clusters for fair comparisons.

Each simulated data set is comprised of $n = 120$ observations with 3 clusters. Each cluster has an equal number of observations. Only the first 10 features are informative while the rest are noise. We consider the following simulation scenarios.

- S1: Spherical data with outliers

  The first 10 informative features in each group are generated from a Gaussian distribution with different μ_k's for each class. Specifically, the first 10 features are generated from $N(\mu_k, \mathbf{I}_{10})$ where $\mu_1 = (-2.5 \cdot \mathbf{1}_5^T, \mathbf{0}_5^T)^T$, $\mu_2 = (\mathbf{0}_5^T, 2.5 \cdot \mathbf{1}_5^T)^T$, $\mu_3 = (2.5 \cdot \mathbf{1}_5^T, \mathbf{0}_5^T)^T$. The outliers in each class are generated from a Gaussian distribution with the same mean centroid μ_k but with higher variance, i.e., $N(\mu_k, 25 \cdot \mathbf{I}_{10})$. The remaining noise features are generated from $N(0,1)$.

  In the first setting (S1A), number of noise features ranges in 25, 50, 75, ⋯ up to 225 with the proportion of number of outliers fixed (= 5%). We also consider the setting when the variance of noise features increases with number of features fixed $p = 200$ and number of outliers fixed (S1B) and high-dimensional setting where p ranges from 250, 500, 750 to 1000 (S1C).

- S2: Non-spherical data with three half moons

  Here we consider the standard simulated data of three interlocking half moons as suggested by Chi and Lange (2015) and Wang et al. (2018). The first 10 features are informative in which each pair makes up two-dimensional three interlocking half moons. We randomly select 5% of the observations in each group and make them outliers. The remaining noise features are generated from $N(0,1)$. The number of noise features ranges from 25, 50, 75, ⋯ up to 225. In both S1 and S2, we compare Manhattan Gecco+ with other existing methods.

- S3: Count-valued data

  The first 10 informative features in each group are generated from a Poisson distribution with different μ_k's ($i = 1, 2, 3$) for each class. Specifically, $\mu_1 = 1 \cdot \mathbf{1}_{10}$, $\mu_2 = 4 \cdot \mathbf{1}_{10}$, $\mu_3 = 7 \cdot \mathbf{1}_{10}$. The remaining noise features are generated from

a Poisson distribution with the same μ's which are randomly generated integers from 1 to 10. The number of noise features ranges from 25, 50, 75, ⋯ up to 225.

We summarize simulation results in Figure 3. We find that for spherical data with outliers, adaptive Manhattan Gecco+ performs the best in high dimensions. Manhattan Gecco performs well in low dimensions but poorly as the number of noisy features increases. Manhattan Gecco+ performs well as the dimension increases, but adaptive Manhattan Gecco+ outperforms the former as it adaptively penalizes the features, meaning that noisy features quickly get zeroed out in the clustering path and that only the informative features perform important roles in clustering. We see that, without adaptive methods, we do not achieve the full benefit of performing feature selection. As we perform adaptive Gecco+, we show vast improvement in clustering purity as the number of noise features grows where regular Gecco performs poorly. Sparse convex clustering performs the worst as it tends to pick outliers as singleton clusters. In the presence of outliers, Manhattan Gecco+ performs much better than sparse convex clustering as we choose a loss function that is more robust to outliers. Interestingly, k-means performs better than sparse convex clustering. This is mainly because sparse convex clustering calculates pairwise distance in the weights, placing outliers in singleton clusters more likely than k-means which calculates within-cluster variances (where outliers could be closer to the cluster centroids than to other data points). Our simulation results also show that adaptive Manhattan Gecco+ works well for non-spherical data by selecting the correct features. For count data, all three adaptive Gecco+ methods perform better than k-means, hierarchical clustering and sparse convex clustering. We should point out that there are several linkage options for hierarchical clustering. For visualization purposes, we only show the linkage with the best and worst performance instead of all the linkages. Also we use the appropriate data-specific distance metrics in hierarchical clustering. For k-means, we use k-means++ (Arthur and Vassilvitskii, 2006) for initialization.

Table 2 shows the variable selection accuracy of sparse convex clustering and adaptive Gecco+ in terms of F_1 score. In all scenarios, we fix $p = 225$. We see that adaptive Gecco+ selects the correct features, whereas sparse convex clustering performs poorly.

4.2 Multi-View Data

In this subsection, we evaluate the performance of iGecco and (adaptive) iGecco+ on mixed multi-view data by comparing it with hierarchical clustering, iClusterPlus (Mo et al., 2013) and Bayesian Consensus Clustering (Lock and Dunson, 2013). Again, we measure the accuracy of clustering results using the adjusted Rand index (Hubert and Arabie, 1985).

As before, each simulated data set is comprised of $n = 120$ observations with 3 clusters. Each cluster has an equal number of observations. Only the first 10 features are informative while the rest are noise. We have three data views consisting of continuous data, count-valued data and binary/proportion-valued data. We investigate different scenarios with different dimensions for each data view and consider the following simulation scenarios:

- S1: Spherical data with $p_1 = p_2 = p_3 = 10$
- S2: Three half-moon data with $p_1 = p_2 = p_3 = 10$

- S3: Spherical data with $p_1 = 200$, $p_2 = 100$, $p_3 = 50$
- S4: Spherical data with $p_1 = 50$, $p_2 = 200$, $p_3 = 100$
- S5: Three half-moon data with $p_1 = 200$, $p_2 = 100$, $p_3 = 50$
- S6: Three half-moon data with $p_1 = 50$, $p_2 = 200$, $p_3 = 100$

We employ a similar simulation setup as in Section 4.1 to generate each data view. The difference is that here for informative features, we increase the within-cluster variance for Gaussian data and decrease difference of cluster mean centroids μ_k's for binary and count data so that there is overlap between different clusters. Specifically, for spherical cases, Gaussian data is generated from $N(\mu_k, 3 \cdot \mathbf{I}_{10})$; count data is generated from Poisson with different μ_k's ($\mu_1 = 2$, $\mu_2 = 4$, $\mu_3 = 6$, etc); binary data is generated from Bernoulli with different μ_k's ($\mu_1 = 0.5$, $\mu_2 = 0.2$, $\mu_3 = 0.8$, etc). For half-moon cases, continuous data is simulated with larger noise and the count and proportion-valued data is generated via a copula transform. In this manner, we have created a challenging simulation scenario where accurate clustering results cannot be achieved by considering only a single data view.

Again, for fair comparisons across methods, we assume oracle number of clusters. When applying iGecco(+) methods, we employ Euclidean distances for continuous data, Manhattan distances for count-valued data and Bernoulli log-likelihood for binary or proportion-valued data. We use the latter two losses as they perform well compared with counterpart losses in Gecco+ and demonstrate faster computation speed.

Simulation results in Table 3 and Table 4 show that our methods perform better than existing methods. In low dimensions, iGecco performs comparably with iCluster and Bayesian Consensus Clustering for spherical data. For non-spherical data, iGecco performs much better. For high dimensions, iGecco+ performs better than iGecco while adaptive iGecco+ performs the best as it achieves the full benefit of feature selection. We also applied k-means to the simulated data. The results of k-means are close to (or in some cases worse than) hierarchical clustering with the best-performing linkage; hence we only show the results of hierarchical clustering here for comparison.

Also we show the variable selection results in Table 5 and compare our method to that of iClusterPlus. For fair comparisons, we assume oracle number of features. For our method, we choose α that gives oracle number of features; for iClusterPlus, we select top features based on Lasso coefficient estimates. Our adaptive iGecco+ outperforms iClusterPlus for all scenarios.

Note in this section, we assume that the oracle number of clusters and features are known a priori for fair comparisons. Results when the number of clusters and features are not fixed but estimated based on the data using tuning parameter selection, are given in Appendix J.3.

5. Real Data Examples

In this section, we apply our methods to various real data sets and evaluate our methods against existing ones. We first evaluate the performance of Gecco+ for several real data sets and investigate the features selected by various Gecco+ methods.

5.1 Authors Data

The authors data set consists of word counts from $n = 841$ chapters written by four famous English-language authors (Austen, London, Shakespeare, and Milton). Each class contains an unbalanced number of observations with 69 features. The features are common "stop words" like "a", "be" and "the" which are typically removed before text mining analysis. We use Gecco+ not only to cluster book chapters and compare the clustering assignments with true labels of authors, but also to identify which key words help distinguish the authors. We choose tuning parameters using BIC based approach; results for stability selection based approach are given in Table 15, Appendix J.3.

In Table 6, we compare Gecco+ with existing methods in terms of clustering quality. For hierarchical clustering, we only show the linkage with the best performance (in this whole section). Our method outperforms k-means and the best hierarchical clustering method. Secondly, we look at the word texts selected by Gecco+. As shown in Table 7, Jane Austen tended to use the word "her" more frequently than the other authors; this is expected as the subjects of her novels are typically females. The word "was" seems to separate Shakespeare and Jack London well. Shakespeare preferred to use present tense more while Jack London preferred to use past tense more. To summarize, our Gecco+ not only has superior clustering performance but also selects interpretable features.

5.2 TCGA Breast Cancer Data

The TCGA data set consists of log-transformed Level III RPKM gene expression levels for 445 breast-cancer patients with 353 features from The Cancer Genome Atlas Network (The Cancer Genome Atlas Network, 2012). Five PAM50 breast cancer subtypes are included, i.e., Basal-like, Luminal A, Luminal B, HER2-enriched, and Normal-like. Only 353 genes out of 50,000 with somatic mutations from COSMIC (Forbes et al., 2010) are retained. The data is Level III TCGA BRCA RNA-Sequencing gene expression data that have already been pre-processed according to the following steps: i) reads normalized by RPKM, and ii) corrected for overdispersion by a log-transformation. We remove 7 patients, who belong to the normal-like group and the number of subjects n becomes 438. We also combine Luminal A with Luminal B as they are often considered one aggregate group (Choi et al., 2014).

From Table 8, our method outperforms k-means and the best hierarchical clustering method. Next, we look at the genes selected by Gecco+ in Table 9. FOXA1 is known to be a key gene that characterizes luminal subtypes in DNA microarray analyses (Badve et al., 2007). GATA binding protein 3 (GATA3) is a transcriptional activator highly expressed by the luminal epithelial cells in the breast (Mehra et al., 2005). ERBB2 is known to be associated with HER2 subtype and has been well studied in breast cancer (Harari and Yarden, 2000). Hence our Gecco+ not only outperforms existing methods but also selects genes which are relevant to biology and have been implicated in previous scientific studies.

Next we evaluate the performance of iGecco+ for mixed multi-view data sets and investigate the features selected by iGecco+.

5.3 Multi-omics Data

One promising application for integrative clustering for multi-view data lies in integrative cancer genomics. Biologists seek to integrate data from multiple platforms of high-throughput genomic data to gain a more thorough understanding of disease mechanisms and detect cancer subtypes. In this case study, we seek to integrate four different types of genomic data to study how epigenetics and short RNAs influence the gene regulatory system in breast cancer.

We use the data set from The Cancer Genome Atlas Network (2012). Lock and Dunson (2013) analyzed this data set using integrative methods and we follow the same data pre-processing procedure: i) filter out genes in expression data whose standard deviation is less than 1.5, ii) take square root of methylation data, and iii) take log of miRNA data. We end up with a data set of 348 tumor samples including:

- RNAseq gene expression (GE) data for 645 genes,
- DNA methylation (ME) data for 574 probes,
- miRNA expression (miRNA) data for 423 miRNAs,
- Reverse phase protein array (RPPA) data for 171 proteins.

The data set contains samples used on each platform with associated subtype calls from each technology platform as well as integrated cluster labels from biologists. We use the integrated labels from biologists as true label to compare different methods. Also we merged the subtypes 3 and 4 in the integrated labels as those two subtypes correspond to Luminal A and Luminal B respectively from the predicted label given by gene expression data (PAM50 mRNA).

Figure 6 in Appendix H gives the distribution of data from different platforms. For our iGecco+ methods, we use Euclidean distances for gene expression data and protein data as the distributions of these two data sets appear Gaussian; binomial deviances for methylation data as the value is between [0, 1]; Manhattan distances for miRNA data as the data is highly-skewed.

We compare our adaptive iGecco+ with other existing methods. From Table 10, we see that our method outperforms all the existing methods.

We also investigate the features selected by adaptive iGecco+, shown in Table 11, and find that our method is validated as most are known in the breast cancer literature. For example, FOXA1 is known to segregate the luminal subtypes from the others (Badve et al., 2007), and AGR3 is a known biomarker for breast cancer prognosis and early breast cancer detection from blood (Garczyk et al., 2015). Several well-known miRNAs were selected including MIR-135b, which is upregulated in breast cancer and promotes cell growth (Hua et al., 2016) as well as MIR-190 which suppresses breast cancer metastasis (Yu et al., 2018). Several known proteins were also selected including ERalpha, which is overexpressed in early stages of breast cancer (Hayashi et al., 2003) and GATA3 which plays an integral role in breast luminal cell differentiation and breast cancer progression (Cimino-Mathews et al., 2013).

We also visualize the resulting clusters from adaptive iGecco+. Figure 4 shows the cluster heatmap of multi-omics TCGA data with row orders determined by cluster assignments from iGecco+ and left bar corresponding to the integrated cluster labels from biologists. We see that there is a clear separation between groups and adaptive iGecco+ identifies meaningful subtypes. The black bars at the bottom of each data view correspond to the selected features in Table 11.

6. Discussion

In this paper, we develop a convex formulation of integrative clustering for high-dimensional mixed multi-view data. We propose a unified, elegant methodological solution to two critical issues for clustering and data integration: i) dealing with mixed types of data and ii) selecting sparse, interpretable features in high-dimensional settings. Specifically, we show that clustering for mixed, multi-view data can be achieved using different data-specific convex losses with a joint fusion penalty. We also introduce a shifted group-lasso penalty that shrinks noise features to their loss-specific centers, hence selecting features that play important roles in separating groups. In addition, we make an optimization contribution by proposing and proving the convergence of a new general multi-block ADMM algorithm with sub-problem approximations that efficiently solves our problem. Empirical studies show that iGecco+ outperforms existing clustering methods and selects sparse, interpretable features in separating clusters.

This paper focuses on the methodological development for integrative clustering and feature selection, but there are many possible avenues for future research related to this work. For example, we expect in future work to be able to show that our methods inherit the strong statistical and theoretical properties of other convex clustering approaches such as clustering consistency and prediction consistency. An important problem in practice is choosing which loss function is appropriate for a given data set. While this is beyond the scope of this paper, an interesting direction for future research would be to learn the appropriate convex loss function in a data-driven manner. Additionally, many have shown block missing structure is common in mixed data (Yu et al., 2019; Xiang et al., 2013). A potentially interesting direction for future work would be to develop an extension of iGecco+ that can appropriately handle block-missing multi-view data. Moreover, Weylandt et al. (2020) developed a fast algorithm to compute the entire convex clustering solution path and used this to visualize the results via a dendrogram and pathwise plot. In future work, we expect that algorithmic regularization path approaches can also be applied to our method to be able to represent our solution as a dendrogram and employ other dynamic visualizations. Finally, while we develop an efficient multi-block ADMM algorithm, there may be further room to speed up computation of iGecco+, potentially by using distributed optimization approaches.

In this paper, we demonstrate that our method can be applied to integrative genomics, yet it can be applied to other fields such as multi-modal imaging, national security, online advertising, and environmental studies where practitioners aim to find meaningful clusters and features at the same time. In conclusion, we introduce a principled, unified approach to a challenging problem that demonstrates strong empirical performance and opens many directions for future research.

Our method is implemented in MATLAB and available at https://github.com/DataSlingers/iGecco.

Acknowledgments

The authors would like to thank Michael Weylandt and Tianyi Yao for helpful discussions. Additionally, we would like to thank the anonymous reviewers and action editor for constructive comments and suggestions on this work. GA and MW also acknowledge support from NSF DMS-1554821, NSF NeuroNex-1707400, NSF DMS-1264058, and NIH 1R01GM140468.

Appendix A.: Proof of Propositions

Proposition 1 and 2 are direct extensions from Proposition 2.1 of Chi and Lange (2015). Notice they proved the solution path depends continuously on the tuning parameter γ and the weight matrix $\mathbf{w}$. It follows that the argument can be also applied to tuning parameter α, the loss weight π_k, and feature weight $\zeta_j^{(k)}$. Also it is obvious that the loss $\ell(\cdot)$ is continuous with respect to the data, $\mathbf{X}$.

We show in detail how to prove Proposition 3 in the following. First we rewrite the objective $F_{\gamma,\alpha}(\mathbf{U})$ as:

$$F_{\gamma,\alpha}(\mathbf{U}) = \sum_{k=1}^{K} \pi_k \ell_k(\mathbf{X}^{(k)} | \mathbf{U}^{(k)}) + \gamma \sum_{i<i'} w_{ii'} \sqrt{\sum_{k=1}^{K} \|\mathbf{U}_i^{(k)} - \mathbf{U}_{i'}^{(k)}\|_2^2} + \alpha \sum_{k=1}^{K} \sum_{j=1}^{p_k} \zeta_j^{(k)} \|\mathbf{U}_{\cdot j}^{(k)} - \tilde{x}_j^{(k)} \mathbf{1}_n\|_2$$
$$= \sum_{k=1}^{K} \sum_{i=1}^{n} \pi_k \ell_k(\mathbf{X}_{i\cdot}^{(k)} | \mathbf{U}_{i\cdot}^{(k)}) + \gamma \sum_{i<i'} w_{ii'} \sqrt{\sum_{k=1}^{K} \|\mathbf{U}_i^{(k)} - \mathbf{U}_{i'}^{(k)}\|_2^2} + \alpha \sum_{k=1}^{K} \sum_{j=1}^{p_k} \zeta_j^{(k)} \|\mathbf{U}_{\cdot j}^{(k)} - \tilde{x}_j^{(k)} \mathbf{1}_n\|_2.$$

By definition, loss-specific cluster center is $\tilde{\mathbf{x}}^{(k)} = \underset{\mathbf{u}}{\operatorname{argmin}} \sum_{i=1}^{n} \ell_k(\mathbf{X}_{i\cdot}^{(k)} | \mathbf{u})$. Since ℓ_k is convex, it is equivalent to $\mathbf{u}$ such that $\partial \sum_i \ell_k(\mathbf{X}_{i\cdot}^{(k)} | \mathbf{u}) = 0$. Hence, $\partial \sum_i \ell_k(\mathbf{X}_{i\cdot}^{(k)} | \tilde{\mathbf{x}}^{(k)}) = 0$.

We use the similar proof approach of Chi and Lange (2015). A point $\mathbf{X}$ furnishes a global minimum of the convex function $F(\mathbf{X})$ if and only if all forward directional derivatives $d_{\Theta} F(\mathbf{X})$ at $\mathbf{X}$ are nonnegative. Here $\mathbf{\Theta} = \left\{\Theta^{(1)}, \cdots, \Theta^{(K)}\right\}$ represents a direction in the space of possible concatenated centroids, where $\Theta^{(k)} \in \mathbb{R}^{n \times p_k}$. We calculate the directional derivatives:

$$d_{\Theta} F_{\gamma,\alpha}(\tilde{\mathbf{X}}) = \sum_{k=1}^{K} \sum_{i=1}^{n} \pi_k \langle \partial \ell_k(\mathbf{X}_{i\cdot}^{(k)} | \tilde{\mathbf{x}}^{(k)}) | \Theta_{i\cdot}^{(k)} \rangle + \gamma \sum_{i<i'} w_{ii'} \sqrt{\sum_{k=1}^{K} \|\Theta_i^{(k)} - \Theta_{i'}^{(k)}\|_2^2}$$
$$+ \alpha \sum_{k=1}^{K} \sum_{j=1}^{p_k} \zeta_j^{(k)} \|\Theta_{\cdot j}^{(k)} - \tilde{x}_j^{(k)} \mathbf{1}_n\|_2.$$

Note $\sum_{i=1}^{n} \langle \partial \ell_k(\mathbf{X}_{i\cdot}^{(k)} | \tilde{\mathbf{x}}^{(k)}) | \Theta_{i'\cdot}^{(k)} \rangle = 0$. The generalized Cauchy-Schwartz inequality therefore implies:

$$\sum_{i=1}^{n}\langle\partial\ell_k(\mathbf{X}_i^{(k)}|\tilde{\mathbf{x}}^{(k)})|\Theta_i^{(k)}\rangle = \frac{1}{n}\sum_{i=1}^{n}\sum_{i'=1}^{n}\langle\partial\ell_k(\mathbf{X}_i^{(k)}|\tilde{\mathbf{x}}^{(k)})|\Theta_i^{(k)}\rangle$$
$$= \frac{1}{n}\sum_{i=1}^{n}\sum_{i'=1}^{n}\langle\partial\ell_k(\mathbf{X}_i^{(k)}|\tilde{\mathbf{x}}^{(k)})|\Theta_i^{(k)}-\Theta_{i'}^{(k)}\rangle$$
$$\geq -\frac{2}{n}\sum_{i<i'}\|\partial\ell_k(\mathbf{X}_i^{(k)}|\tilde{\mathbf{x}}^{(k)})\|_2\,\big|\,\|\Theta_i^{(k)}-\Theta_{i'}^{(k)}\|_2$$
$$\geq -\frac{2}{n}\sum_{i<i'}\|\partial\ell_k(\mathbf{X}_i^{(k)}|\tilde{\mathbf{x}}^{(k)})\|_2\,\Big|\sqrt{\sum_{k=1}^{K}\|\Theta_i^{(k)}-\Theta_{i'}^{(k)}\|_2^2}.$$

Hence, we have:

$$\sum_{k=1}^{K}\sum_{i=1}^{n}\pi_k\langle\partial\ell_k(\mathbf{X}_i^{(k)}|\tilde{\mathbf{x}}^{(k)})|\Theta_i^{(k)}\rangle \geq -\frac{2}{n}\sum_{k=1}^{K}\sum_{i<i'}\pi_k\|\partial\ell_k(\mathbf{X}_i^{(k)}|\tilde{\mathbf{x}}^{(k)})\|_2\,\Big|\sqrt{\sum_{k=1}^{K}\|\Theta_i^{(k)}-\Theta_{i'}^{(k)}\|_2^2}.$$

Take γ sufficiently large such that:

$$\gamma\sum_{i<i'}w_{ii'}\sqrt{\sum_{k=1}^{K}\|\Theta_i^{(k)}-\Theta_{i'}^{(k)}\|_2^2} \geq -\frac{2}{n}\sum_{k=1}^{K}\sum_{i<i'}\pi_k\|\partial\ell_k(\mathbf{X}_i^{(k)}|\tilde{\mathbf{x}}^{(k)})\|_2\,\Big|\sqrt{\sum_{k=1}^{K}\|\Theta_i^{(k)}-\Theta_{i'}^{(k)}\|_2^2}.$$

When all $w_{ii'} > 0$, one can take any γ that exceeds

$$K\cdot\frac{2}{n}\max_{i,i',k}\frac{\pi_k\|\partial\ell_k(\mathbf{X}_{i\cdot}^{(k)},\tilde{\mathbf{x}}^{(k)})\|_2}{w_{ii'}}.$$

In general set

$$\beta = K\cdot\frac{2}{n\min_{w_{ii'}>0}w_{ii'}}\max_{i,i',k}\pi_k\|\partial\ell_k(\mathbf{X}_i^{(k)},\tilde{\mathbf{x}}^{(k)})\|_2.$$

For any pair i and i' there exists a path $i\to k\to\cdots\to l\to i'$ along which the weights are positive. It follows that

$$\frac{2}{n}\sum_{k=1}^{K}\pi_k\|\partial\ell_k(\mathbf{X}_i^{(k)}|\tilde{\mathbf{x}}^{(k)})\|_2\cdot\sqrt{\sum_{k=1}^{K}\|\Theta_i^{(k)}-\Theta_{i'}^{(k)}\|_2^2} \leq \beta\sum_{i<i'}w_{ii'}\sqrt{\sum_{k=1}^{K}\|\Theta_i^{(k)}-\Theta_{i'}^{(k)}\|_2^2}.$$

We have

$$\frac{2}{n}\sum_{k=1}^{K}\sum_{i<i'}\pi_k\|\partial\ell_k(\mathbf{X}_i^{(k)}|\tilde{\mathbf{x}}^{(k)})\|_2\cdot\sqrt{\sum_{k=1}^{K}\|\Theta_i^{(k)}-\Theta_{i'}^{(k)}\|_2^2} \leq \binom{n}{2}\beta\sum_{i<i'}w_{ii'}\sqrt{\sum_{k=1}^{K}\|\Theta_i^{(k)}-\Theta_{i'}^{(k)}\|_2^2}.$$

Hence the forward directional derivative test is satisfied for any $\gamma \geq \binom{n}{2}\beta$.

On the other hand, for fixed γ, the generalized Cauchy-Schwartz inequality implies:

$$\begin{aligned}\sum_{i=1}^{n}\langle \partial \ell_k(\mathbf{X}_{i|}^{(k)} | \tilde{\mathbf{x}}^{(k)}) | \Theta_{i|}^{(k)}\rangle &= \sum_{i=1}^{n}\langle \partial \ell_k(\mathbf{X}_{i|}^{(k)} | \tilde{\mathbf{x}}^{(k)}) | \Theta_{i|}^{(k)} - \tilde{\mathbf{x}}^{(k)}\rangle \\ &= \sum_{j=1}^{p_k}\langle \partial \ell_k(\mathbf{X}_{|j}^{(k)} | \tilde{x}_j^{(k)} \,|\, \mathbf{1}_n) | \Theta_{|j}^{(k)} - \tilde{x}_j^{(k)} \,|\, \mathbf{1}_n\rangle \\ &\geq -\sum_{j=1}^{p_k} \| \partial \ell_k(\mathbf{X}_{|j}^{(k)} | \tilde{x}_j^{(k)} \,|\, \mathbf{1}_n)\|_2 \,|\, \|\Theta_{|j}^{(k)} - \tilde{x}_j^{(k)} \,|\, \mathbf{1}_n\|_2 .\end{aligned}$$

Hence, we have:

$$\sum_{k=1}^{K}\sum_{i=1}^{n}\pi_k\langle \partial \ell_k(\mathbf{X}_{i|}^{(k)} | \tilde{\mathbf{x}}^{(k)}) | \Theta_{i|}^{(k)}\rangle \geq -\sum_{k=1}^{K}\sum_{j=1}^{p_k}\pi_k \| \partial \ell_k(\mathbf{X}_{|j}^{(k)} | \tilde{x}_j^{(k)} \,|\, \mathbf{1}_n)\|_2 \,|\, \|\Theta_{|j}^{(k)} - \tilde{x}_j^{(k)} \,|\, \mathbf{1}_n\|_2 .$$

Take α sufficiently large so that

$$\alpha\sum_{k=1}^{K}\sum_{j=1}^{p_k}\zeta_j^{(k)} \|\Theta_{|j}^{(k)} - \tilde{x}_j^{(k)} \,|\, \mathbf{1}_n\|_2 \geq -\sum_{k=1}^{K}\sum_{j=1}^{p_k}\pi_k \| \partial \ell_k(\mathbf{X}_{|j}^{(k)} | \tilde{x}_j^{(k)} \,|\, \mathbf{1}_n)\|_2 \,|\, \|\Theta_{|j}^{(k)} - \tilde{x}_j^{(k)} \,|\, \mathbf{1}_n\|_2 .$$

When all $\zeta_j^{(k)} > 0$, one can take any α that exceeds

$$\max_{j,k}\frac{\pi_k \| \partial \ell_k(\mathbf{X}_{\cdot j}^{(k)}, \tilde{x}_j^{(k)} \cdot \mathbf{1}_n)\|_2}{\zeta_j^{(k)}}.$$

In general, set $\alpha \geq \frac{1}{\min_{\zeta_j^{(k)} > 0}\zeta_j^{(k)}} \cdot \max_{j,k}\pi_k \| \partial \ell_k(\mathbf{X}_{\cdot j}^{(k)}, \tilde{x}_j^{(k)} \cdot \mathbf{1}_n)\|_2$. It is easy to check the forward directional derivative test is satisfied. ■

Appendix B.: Proof of Theorem 4

Recall the iGecco+ problem is:

$$\begin{aligned}\min_{\mathbf{U}^{(k)}} &\sum_{k=1}^{K}\pi_k \ell_k(\mathbf{X}^{(k)} | \mathbf{U}^{(k)}) + \gamma\sum_{i<i'} w_{ii'}\sqrt{\sum_{k=1}^{K}\|\mathbf{U}_{i|}^{(k)} - \mathbf{U}_{i'|}^{(k)}\|_2^2} \\ &+\alpha\sum_{k=1}^{K}\sum_{j=1}^{p_k}\zeta_j^{(k)}\|\mathbf{U}_{|j}^{(k)} - \tilde{x}_j^{(k)} \,|\, \mathbf{1}_n\|_2 .\end{aligned}$$

We can recast the original iGecco+ problem as the equivalent constrained problem:

$$\underset{\mathbf{U}^{(k)},\mathbf{V}}{\text{minimize}} \sum_{k=1}^{K} \pi_k \ell_k(\mathbf{X}^{(k)}, \mathbf{U}^{(k)}) + \underbrace{\gamma\left(\sum_{l \in \varepsilon} w_l \|\mathbf{V}_{l|}\|_2\right)}_{P_1(\mathbf{V};\mathbf{w})}$$
$$+\alpha \sum_{k=1}^{K} \sum_{j=1}^{p_k} \zeta_j^{(k)} \|\mathbf{U}_{|j}^{(k)} - \tilde{x}_j^{(k)} \,|\, \mathbf{1}_n\|_2 + \sum_{k'=1}^{K} \pi_{k'} f_{k'}(\mathbf{Z}^{(k')}) + \alpha \sum_{k'=1}^{K} \underbrace{\left(\sum_{j=1}^{p'_k} \zeta_j^{(k')} \|\mathbf{r}_j^{(k')}\|_2\right)}_{P_2(\mathbf{R}^{(k')};\zeta^{(k')})} \quad (2)$$

$$\text{subject to} \quad \mathbf{D}\mathbf{U}^{(k)} - \mathbf{V}^{(k)} = 0,\ \mathbf{D}\mathbf{U}^{(k')} - \mathbf{V}^{(k')} = 0,\ \mathbf{X}^{(k')} - \mathbf{U}^{(k')} = \mathbf{Z}^{(k')}, \mathbf{U}^{(k')} - \widetilde{\mathbf{X}}^{(k')} = \mathbf{R}^{(k')},$$

where ℓ_k refers to the differentiable losses and $\ell_{k'}$ refers to the non-differentiable losses. We use multi-block ADMM algorithm (Algorithm 6) to solve the problem above.

To prove convergence of Algorithm 5, we first show that multi-block ADMM Algorithm 6 converges to a global minimum. Then we show that we can proximal-linearize the sub-problems in the primal updates of Algorithm 6 with proved convergence and this is equivalent to Algorithm 5. Without loss of generality, we assume we have one differentiable loss $\ell_1(\cdot)$ and one non-differentiable distance-based loss $\ell_2(\cdot)$.

Algorithm 6

Multi-block ADMM to solve iGecco+

while not converged **do**

for all $k = 1, \cdots, K$ **do**

$$\mathbf{U}^{(k)} = \underset{\mathbf{U}}{\text{argmin}}\ \pi_k \ell_k(\mathbf{X}^{(k)}, \mathbf{U}) + \frac{\rho}{2}\|\mathbf{D}\mathbf{U} - \mathbf{V}^{(k)} + \mathbf{\Lambda}^{(k)}\|_F^2 + \alpha \sum_{j=1}^{p_k} \zeta_j^{(k)} \|\mathbf{U}_{|j} - \tilde{x}_j^{(k)} \,|\, \mathbf{1}_n\|_2$$

$$\mathbf{U}^{(k')} = \underset{\mathbf{U}}{\text{argmin}}\ \frac{\rho}{2}\|\mathbf{X}^{(k')} - \mathbf{U} + \mathbf{Z}^{(k')} + \mathbf{\Psi}^{(k')}\|_F^2 + \frac{\rho}{2}\|\mathbf{U} - \widetilde{\mathbf{X}}^{(k')} - \mathbf{R}^{(k')} + \mathbf{N}^{(k')}\|_F^2 + \frac{\rho}{2}\|\mathbf{D}\mathbf{U} - \mathbf{V}^{(k')} + \mathbf{\Lambda}^{(k')}\|_F^2$$

$$\mathbf{Z}^{(k')} = \underset{\mathbf{Z}}{\text{argmin}}\ \pi_{k'} f_{k'}(\mathbf{Z}) + \frac{\rho}{2}\|\mathbf{X}^{(k')} - \mathbf{U}^{(k')} - \mathbf{Z} + \mathbf{\Psi}^{(k')}\|_F^2$$

$$\mathbf{R}^{(k')} = \underset{\mathbf{R}}{\text{argmin}}\ \alpha \sum_{j=1}^{p_{k'}} \zeta_j^{(k')} \|\mathbf{U}_{|j}^{(k')} - \tilde{x}_j^{(k')} \,|\, \mathbf{1}_n\|_2 + \frac{\rho}{2}\|\mathbf{U}^{(k')} - \widetilde{\mathbf{X}}^{(k')} - \mathbf{R} + \mathbf{N}^{(k')}\|_F^2$$

$$\mathbf{\Psi}^{(k')} = \mathbf{\Psi}^{(k')} + (\mathbf{X}^{(k')} - \mathbf{U}^{(k')} - \mathbf{Z}^{(k')})$$

$$\mathbf{N}^{(k')} = \mathbf{N}^{(k')} + (\mathbf{U}^{(k')} - \widetilde{\mathbf{X}}^{(k')} - \mathbf{R}^{(k')})$$

end for

$$\mathbf{V} = \underset{\mathbf{V}}{\text{argmin}} \frac{\rho}{2}\|\mathbf{D}\mathbf{U}^{(k)} - \mathbf{V}^{(k)} + \mathbf{\Lambda}^{(k)}\|_F^2 + \frac{\rho}{2}\|\mathbf{D}\mathbf{U}^{(k')} - \mathbf{V}^{(k')} + \mathbf{\Lambda}^{(k')}\|_F^2 + \gamma\left(\sum_{l \in \varepsilon} w_l \|\mathbf{V}_{l|}\|_2\right)$$

$$\mathbf{\Lambda}^{(k)} = \mathbf{\Lambda}^{(k)} + (\mathbf{D}\mathbf{U}^{(k)} - \mathbf{V}^{(k)}) \quad \text{for all } k \text{ and } k'$$

end while

To prove convergence of Algorithm 6, we first propose a sufficient condition for the convergence of four-block ADMM and prove it holds true. This is an extension of the

convergence results in Section 2 of the work by Chen et al. (2016). Suppose the convex optimization problem with linear constraints we want to minimize is

$$\begin{aligned} &\min\ \theta_1(\mathbf{x}_1)+\theta_2(\mathbf{x}_2)+\theta_3(\mathbf{x}_3)+\theta_4(\mathbf{x}_4)\\ &\text{s.t. } \mathbf{A}_1\mathbf{x}_1+\mathbf{A}_2\mathbf{x}_2+\mathbf{A}_3\mathbf{x}_3+\mathbf{A}_4\mathbf{x}_4=\mathbf{b}\,. \end{aligned} \tag{3}$$

The multi-block ADMM has the following form. Note here, the superscript in $\mathbf{x}_i^{(k+1)}$ refers to the $(k+1)^{th}$ iteration in the ADMM updates. We have:

$$\begin{cases} \mathbf{x}_1^{(k+1)} = \operatorname{argmin}\{L_{\mathbf{A}}(\mathbf{x}_1, \mathbf{x}_2^{(k)}, \mathbf{x}_3^{(k)}, \mathbf{x}_4^{(k)}, \boldsymbol{\lambda}^{(k)})\}\\ \mathbf{x}_2^{(k+1)} = \operatorname{argmin}\{L_{\mathbf{A}}(\mathbf{x}_1^{(k+1)}, \mathbf{x}_2, \mathbf{x}_3^{(k)}, \mathbf{x}_4^{(k)}, \boldsymbol{\lambda}^{(k)})\}\\ \mathbf{x}_3^{(k+1)} = \operatorname{argmin}\{L_{\mathbf{A}}(\mathbf{x}_1^{(k+1)}, \mathbf{x}_2^{(k+1)}, \mathbf{x}_3, \mathbf{x}_4^{(k)}, \boldsymbol{\lambda}^{(k)})\}\\ \mathbf{x}_4^{(k+1)} = \operatorname{argmin}\{L_{\mathbf{A}}(\mathbf{x}_1^{(k+1)}, \mathbf{x}_2^{(k+1)}, \mathbf{x}_3^{(k+1)}, \mathbf{x}_4, \boldsymbol{\lambda}^{(k)})\}\\ \boldsymbol{\lambda}^{(k+1)} = \boldsymbol{\lambda}^{(k)} - (\mathbf{A}_1\mathbf{x}_1^{(k+1)} + \mathbf{A}_2\mathbf{x}_2^{(k+1)} + \mathbf{A}_3\mathbf{x}_3^{(k+1)} + \mathbf{A}_4\mathbf{x}_4^{(k+1)} - \mathbf{b})\,, \end{cases} \tag{4}$$

where

$$\begin{aligned} L_{\mathbf{A}} = \sum_{i=1}^{4}\theta_i(\mathbf{x}_i) - \boldsymbol{\lambda}^T(\mathbf{A}_1\mathbf{x}_1+\mathbf{A}_2\mathbf{x}_2+\mathbf{A}_3\mathbf{x}_3+\mathbf{A}_4\mathbf{x}_4-\mathbf{b})\\ +\frac{1}{2}\|\mathbf{A}_1\mathbf{x}_1+\mathbf{A}_2\mathbf{x}_2+\mathbf{A}_3\mathbf{x}_3+\mathbf{A}_4\mathbf{x}_4-\mathbf{b}\|_2^2\,. \end{aligned}$$

We establish Lemma 5, a sufficient condition for convergence of four-block ADMM:

Lemma 5 *(Sufficient Condition for Convergence of Four-block ADMM) A sufficient condition ensuring the convergence of* (4) *to a global solution of* (3) *is:* $\mathbf{A}_2^T\mathbf{A}_3 = 0$, $\mathbf{A}_2^T\mathbf{A}_4 = 0$, $\mathbf{A}_3^T\mathbf{A}_4 = 0$.

We prove Lemma 5 at the end of this section. Note that Lemma 5 is stated in vector form and therefore we need to transform the constraints in the original iGecco+ problem (2) from matrix form to vector form in order to apply Lemma 5. Note that $\mathbf{D}\mathbf{U}^{(k)} = \mathbf{V}^{(k)} \Leftrightarrow \mathbf{U}^{(k)^T}\mathbf{D}^T = \mathbf{V}^{(k)^T} \Leftrightarrow (\mathbf{D}\otimes\mathbf{I}_{p_k})\text{vec}(\mathbf{U}^{(k)^T}) = \text{vec}(\mathbf{V}^{(k)^T})$. Hence we can write the constraints in (2) as:

$$\begin{pmatrix}\mathbf{A_1} & \mathbf{0}\\ \mathbf{0} & \mathbf{I}\\ \mathbf{0} & \mathbf{A_2}\\ \mathbf{0} & \mathbf{I}\end{pmatrix}\mathbf{u} + \begin{pmatrix}\mathbf{0}\\ \mathbf{I}\\ \mathbf{0}\\ \mathbf{0}\end{pmatrix}\mathbf{z} + \begin{pmatrix}\mathbf{0}\\ \mathbf{0}\\ \mathbf{0}\\ \mathbf{-I}\end{pmatrix}\mathbf{r} + \begin{pmatrix}\mathbf{-I} & \mathbf{0}\\ \mathbf{0} & \mathbf{0}\\ \mathbf{0} & \mathbf{-I}\\ \mathbf{0} & \mathbf{0}\end{pmatrix}\mathbf{v} = \mathbf{b}\,,$$

where $\mathbf{u} = \begin{pmatrix}\mathbf{u}_1\\ \mathbf{u}_2\end{pmatrix} = \begin{pmatrix}\mathrm{vec}(\mathbf{U}^{(1)^T})\\ \mathrm{vec}(\mathbf{U}^{(2)^T})\end{pmatrix}$, $\mathbf{A}_1 = \mathbf{D} \otimes \mathbf{I}_{p1}$, $\mathbf{A}_2 = \mathbf{D} \otimes \mathbf{I}_{p2}$, $\mathbf{z} = \mathrm{vec}(\mathbf{Z}^T)$, $\mathrm{r} = \mathrm{vec}(\mathbf{R}^T)$, $\mathbf{v} = \mathrm{vec}(\mathbf{V}^T) = \begin{pmatrix}\mathrm{vec}\left(\mathbf{V}^{(1)^T}\right)\\ \mathrm{vec}(\mathbf{V}^{(2)^T})\end{pmatrix}$, $\mathbf{b} = \begin{pmatrix}\mathbf{0}_{p1\times|\epsilon|}\\ \mathrm{vec}(\mathbf{X}^{(2)^T})\\ \mathbf{0}_{p2\times|\epsilon|}\\ \tilde{\mathbf{x}}^{(2)}\\ \vdots\\ \tilde{\mathbf{x}}^{(2)}\end{pmatrix}$. $\tilde{\mathbf{x}} \in \mathbb{R}^{p2}$ is a column vector consisting of all $\tilde{x}_j^2$ and is repeated n times in $\mathbf{b}$.

Next we show that the constraints in (2) for our problem satisfy the condition in Lemma 5 and hence the multi-block ADMM Algorithm 6 converges.

By construction, $\mathbf{E}_2 = \begin{pmatrix}\mathbf{0}\\ \mathbf{I}\\ \mathbf{0}\\ \mathbf{0}\end{pmatrix}$, $\mathbf{E}_3 = \begin{pmatrix}\mathbf{0}\\ \mathbf{0}\\ \mathbf{0}\\ -\mathbf{I}\end{pmatrix}$ and $\mathbf{E}_4 = \begin{pmatrix}-\mathbf{I} & \mathbf{0}\\ \mathbf{0} & \mathbf{0}\\ \mathbf{0} & -\mathbf{I}\\ \mathbf{0} & \mathbf{0}\end{pmatrix}$. It is easy to verify that: $\mathbf{E}_2^T\mathbf{E}_3 = \mathbf{0}$, $\mathbf{E}_2^T\mathbf{E}_4 = \mathbf{0}$, $\mathbf{E}_3^T\mathbf{E}_4 = \mathbf{0}$. Hence our setup satisfies the sufficient condition in Lemma 5 and the multi-block ADMM Algorithm 6 converges.

Next, we see that each primal update in Algorithm 5 is equivalent to the primal update by applying proximal linearized ADMM to the sub-problems in Algorithm 6. (We will show this in detail in Theorem 6.) It is easy to show that these updates with closed-form solutions are special cases of proximal-linearizing the sub-problems. Meanwhile, Lu et al. (2016); Liu et al. (2013) showed the convergence of proximal linearized multi-block ADMM. Hence Algorithm 5 converges to a global minimum if ℓ_k is convex for all k and has Lipschitz gradient when it is differentiable. Further, if each ℓ_k is strictly convex, it converges to the unique global solution. ■

Proof of Lemma 5:

According to the first-order optimality conditions of the minimization problems in (4), we have:

$$\begin{cases}\theta_1(\mathbf{x}_1) - \theta_1(\mathbf{x}_1^{(k+1)}) + (\mathbf{x}_1 - \mathbf{x}_1^{(k+1)})^T\{-\mathbf{A}_1^T[\boldsymbol{\lambda}^{(k)} - (\mathbf{A}_1\mathbf{x}_1^{(k+1)} + \mathbf{A}_2\mathbf{x}_2^{(k)} + \mathbf{A}_3\mathbf{x}_3^{(k)} + \mathbf{A}_4\mathbf{x}_4^{(k)} - \mathbf{b})]\} \geq 0\\ \theta_2(\mathbf{x}_2) - \theta_2(\mathbf{x}_2^{(k+1)}) + (\mathbf{x}_2 - \mathbf{x}_2^{(k+1)})^T\{-\mathbf{A}_2^T[\boldsymbol{\lambda}^{(k)} - (\mathbf{A}_1\mathbf{x}_1^{(k+1)} + \mathbf{A}_2\mathbf{x}_2^{(k+1)} + \mathbf{A}_3\mathbf{x}_3^{(k)} + \mathbf{A}_4\mathbf{x}_4^{(k)} - \mathbf{b})]\} \geq 0\\ \theta_3(\mathbf{x}_3) - \theta_3(\mathbf{x}_3^{(k+1)}) + (\mathbf{x}_3 - \mathbf{x}_3^{(k+1)})^T\{-\mathbf{A}_3^T[\boldsymbol{\lambda}^{(k)} - (\mathbf{A}_1\mathbf{x}_1^{(k+1)} + \mathbf{A}_2\mathbf{x}_2^{(k+1)} + \mathbf{A}_3\mathbf{x}_3^{(k+1)} + \mathbf{A}_4\mathbf{x}_4^{(k)} - \mathbf{b})]\} \geq 0\\ \theta_4(\mathbf{x}_4) - \theta_4(\mathbf{x}_4^{(k+1)}) + (\mathbf{x}_4 - \mathbf{x}_4^{(k+1)})^T\{-\mathbf{A}_4^T[\boldsymbol{\lambda}^{(k)} - (\mathbf{A}_1\mathbf{x}_1^{(k+1)} + \mathbf{A}_2\mathbf{x}_2^{(k+1)} + \mathbf{A}_3\mathbf{x}_3^{(k+1)} + \mathbf{A}_4\mathbf{x}_4^{(k+1)} - \mathbf{b})]\} \geq 0.\end{cases}$$

Since $\mathbf{A}_2^T\mathbf{A}_3 = \mathbf{0}$, $\mathbf{A}_2^T\mathbf{A}_4 = \mathbf{0}$, $\mathbf{A}_3^T\mathbf{A}_4 = \mathbf{0}$, we have:

$$\begin{cases} \theta_1(\mathbf{x}_1) - \theta_1(\mathbf{x}_1^{(k+1)}) + (\mathbf{x}_1 - \mathbf{x}_1^{(k+1)})^T \{-\mathbf{A}_1^T[\boldsymbol{\lambda}^{(k)} - (\mathbf{A}_1\mathbf{x}_1^{(k+1)} + \mathbf{A}_2\mathbf{x}_2^{(k)} + \mathbf{A}_3\mathbf{x}_3^{(k)} + \mathbf{A}_4\mathbf{x}_4^{(k)} - \mathbf{b})]\} \geq 0 \\ \theta_2(\mathbf{x}_2) - \theta_2(\mathbf{x}_2^{(k+1)}) + (\mathbf{x}_2 - \mathbf{x}_2^{(k+1)})^T \{-\mathbf{A}_2^T[\boldsymbol{\lambda}^{(k)} - (\mathbf{A}_1\mathbf{x}_1^{(k+1)} + \mathbf{A}_2\mathbf{x}_2^{(k+1)} - \mathbf{b})]\} \geq 0 \\ \theta_3(\mathbf{x}_3) - \theta_3(\mathbf{x}_3^{(k+1)}) + (\mathbf{x}_3 - \mathbf{x}_3^{(k+1)})^T \{-\mathbf{A}_3^T[\boldsymbol{\lambda}^{(k)} - (\mathbf{A}_1\mathbf{x}_1^{(k+1)} + \mathbf{A}_3\mathbf{x}_3^{(k+1)} - \mathbf{b})]\} \geq 0 \\ \theta_4(\mathbf{x}_4) - \theta_4(\mathbf{x}_4^{(k+1)}) + (\mathbf{x}_4 - \mathbf{x}_4^{(k+1)})^T \{-\mathbf{A}_4^T[\boldsymbol{\lambda}^{(k)} - (\mathbf{A}_1\mathbf{x}_1^{(k+1)} + \mathbf{A}_4\mathbf{x}_4^{(k+1)} - \mathbf{b})]\} \geq 0, \end{cases}$$

which is also the first-order optimality condition of the scheme:

$$\begin{cases} \mathbf{x}_1^{(k+1)} = \operatorname{argmin}\{\theta_1(\mathbf{x}_1) - (\boldsymbol{\lambda}^{(k)})^T(\mathbf{A}_1\mathbf{x}_1) + \frac{1}{2}\|\mathbf{A}_1\mathbf{x}_1 + \mathbf{A}_2\mathbf{x}_2^{(k)} + \mathbf{A}_3\mathbf{x}_3^{(k)} + \mathbf{A}_4\mathbf{x}_4^{(k)} - \mathbf{b}\|_2^2\} \\ (\mathbf{x}_2^{(k+1)}, \mathbf{x}_3^{(k+1)}, \mathbf{x}_4^{(k+1)}) = \operatorname{argmin}\{\theta_2(\mathbf{x}_2) + \theta_3(\mathbf{x}_3) + \theta_4(\mathbf{x}_4) - (\boldsymbol{\lambda}^{(k)})^T(\mathbf{A}_2\mathbf{x}_2 + \mathbf{A}_3\mathbf{x}_3 + \mathbf{A}_4\mathbf{x}_4) \\ \qquad\qquad + \frac{1}{2}\|\mathbf{A}_1\mathbf{x}_1^{(k+1)} + \mathbf{A}_2\mathbf{x}_2 + \mathbf{A}_3\mathbf{x}_3 + \mathbf{A}_4\mathbf{x}_4 - \mathbf{b}\|_2^2\} \\ \boldsymbol{\lambda}^{(k+1)} = \boldsymbol{\lambda}^{(k)} - (\mathbf{A}_1\mathbf{x}_1^{(k+1)} + \mathbf{A}_2\mathbf{x}_2^{(k+1)} + \mathbf{A}_3\mathbf{x}_3^{(k+1)} + \mathbf{A}_4\mathbf{x}_4^{(k+1)} - \mathbf{b}) \ . \end{cases} \tag{5}$$

Clearly, (5) is a specific application of the original two-block ADMM to (3) by regarding $(\mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4)$ as one variable, $[\mathbf{A}_2, \mathbf{A}_3, \mathbf{A}_4]$ as one matrix and $\theta_2(\mathbf{x}_2) + \theta_3(\mathbf{x}_3) + \theta_4(\mathbf{x}_4)$ as one function. ■

Appendix C.: Gecco+ for Differentiable Losses

In this section, we propose algorithms to solve Gecco+ when the loss ℓ is differentiable and gradient is Lipschitz continuous. In this case, we develop a fast two-block ADMM algorithm without fully solving the $\mathbf{U}$ sub-problem. Our result is closely related to the proximal linearized ADMM literature (Liu et al., 2013; Lu et al., 2016). Also solving the sub-problem approximately is closely connected with the generalized ADMM literature (Deng and Yin, 2016).

In the following sections, we discuss algorithms to solve Gecco+ instead of iGecco+ for notation purposes as we would like to include iteration counter indices in the algorithm for illustrating backtracking; but we can easily extend the algorithm to solve iGecco+. To begin with, we clarify different notations in Gecco+ and iGecco+: the superscript in $\mathbf{U}^{(k)}$ in iGecco+ refers to the k^{th} data view while $\mathbf{U}^{(k)}$ in Gecco+ refers to the k^{th} iteration counter in the ADMM updates. We omit iteration counter indices in all iGecco+ algorithm for notation purposes and use the most recent values of the updates.

C.1 Two-block ADMM in Matrix Form

Suppose the loss $\ell(\mathbf{X},\mathbf{U})$ is differentiable. Similar to the formulation in convex clustering, we can recast the Gecco+ problem as the equivalent constrained problem:

$$\begin{aligned} \underset{\mathbf{U},\mathbf{V}}{\text{minimize}} \quad & \ell(\mathbf{X},\mathbf{U}) + \gamma \underbrace{\left(\sum_{l \in \varepsilon} w_l \|\mathbf{V}_{l\cdot}\|_2 \right)}_{P_1(\mathbf{V};\mathbf{w})} + \alpha \sum_{j=1}^{p} \zeta_j \|\mathbf{U}_{\cdot j} - \tilde{x}_j \mathbf{1}_n\|_2 \\ \text{subject to} \quad & \mathbf{D}\mathbf{U} - \mathbf{V} = 0 \ . \end{aligned}$$

Like in convex clustering (Chi and Lange, 2015; Weylandt et al., 2020), we index a centroid pair by $l = (l_1, l_2)$ with $l_1 < l_2$, define the set of edges over the non-zero weights $\varepsilon = \{l = (l_1, l_2) : w_l > 0\}$, and introduce a new variable $\mathbf{V}_{l\cdot} = \mathbf{U}_{l_1\cdot} - \mathbf{U}_{l_2\cdot}$ to account for the difference between the two centroids. Hence $\mathbf{V}$ is a matrix containing the pairwise differences between connected rows of $\mathbf{U}$. Also $\mathbf{D}$ is the directed difference matrix corresponding to the non-zero fusion weights defined in the work of Weylandt et al. (2020).

We can show that the augmented Lagrangian in scaled form is equal to:

$$L(\mathbf{U},\mathbf{V},\mathbf{\Lambda}) = \ell(\mathbf{X},\mathbf{U}) + \frac{\rho}{2}\|\mathbf{D}\mathbf{U} - \mathbf{V} + \mathbf{\Lambda}\|_F^2 + \alpha \sum_{j=1}^{p} \zeta_j \|\mathbf{U}_{\cdot j} - \tilde{x}_j \mathbf{1}_n\|_2 + \gamma \sum_{l \in \varepsilon} w_l \|\mathbf{V}_{l\cdot}\|_2 \ ,$$

where the dual variable is denoted by $\mathbf{\Lambda}$.

To update $\mathbf{U}$, we need to solve the following sub-problem:

$$\underset{\mathbf{U}}{\text{minimize}} \quad \ell(\mathbf{X},\mathbf{U}) + \frac{\rho}{2}\|\mathbf{D}\mathbf{U} - \mathbf{V} + \mathbf{\Lambda}\|_2^2 + \alpha \sum_{j=1}^{p} \zeta_j \|\mathbf{U}_{\cdot j} - \tilde{x}_j \mathbf{1}_n\|_2 \ .$$

Let $\widetilde{\mathbf{U}} = \mathbf{U} - \widetilde{\mathbf{X}}$. The sub-problem becomes:

$$\underset{\mathbf{U}}{\text{minimize}} \quad \ell(\mathbf{X}, \widetilde{\mathbf{U}} + \widetilde{\mathbf{X}}) + \frac{\rho}{2}\|\mathbf{D}(\widetilde{\mathbf{U}} + \widetilde{\mathbf{X}}) - \mathbf{V} + \mathbf{\Lambda}\|_2^2 + \alpha \sum_{j=1}^{p} \zeta_j \|\widetilde{\mathbf{u}}_j\|_2 \ ,$$

where $\widetilde{\mathbf{u}}_j$ is the j^{th} column of $\widetilde{\mathbf{U}}$. For each ADMM iterate, we have:

$$\widetilde{\mathbf{U}}^{(k)} = \underset{\widetilde{\mathbf{U}}}{\text{argmin}} \quad \ell(\mathbf{X}, \widetilde{\mathbf{U}} + \widetilde{\mathbf{X}}) + \frac{\rho}{2}\|\mathbf{D}(\widetilde{\mathbf{U}} + \widetilde{\mathbf{X}}) - \mathbf{V}^{(k-1)} + \mathbf{\Lambda}^{(k-1)}\|_2^2 + \underbrace{\alpha \sum_{j=1}^{p} \zeta_j \|\widetilde{\mathbf{u}}_j\|_2}_{P_2(\widetilde{\mathbf{U}};\zeta)} \ .$$

This can be solved by running iterative proximal gradient to full convergence:

$$\begin{aligned} \widetilde{\mathbf{U}}^{(k,m)} = & \underset{s_k \cdot \alpha P_2(\cdot\,;\zeta)}{\text{prox}} \left(\widetilde{\mathbf{U}}^{(k,m-1)} - s_k \cdot [\nabla \ell(\mathbf{X}, \widetilde{\mathbf{U}}^{(k,m-1)} + \widetilde{\mathbf{X}}) + \rho\, \mathbf{D}^T \right. \\ & \left. (\mathbf{D}(\widetilde{\mathbf{U}}^{(k,m-1)} + \widetilde{\mathbf{X}}) - \mathbf{V}^{(k-1)} + \mathbf{\Lambda}^{(k-1)})] \right) , \end{aligned}$$

which is equivalent to:

$$\mathbf{U}^{(k,m)} = \underset{s_k \cdot \alpha P_2(\cdot;\zeta)}{\text{prox}}(\mathbf{U}^{(k,m-1)} - \widetilde{\mathbf{X}} - s_k \cdot [\nabla \ell(\mathbf{X}, \mathbf{U}^{(k,m-1)}) + \rho \mathbf{D}^T(\mathrm{DU}^{(k,m-1)} - \mathbf{V}^{(k-1)} + \mathbf{\Lambda}^{(k-1)})]) + \widetilde{\mathbf{X}}\ .$$

Here $\mathbf{U}^{(k,m)}$ refers to the m^{th} inner iteration counter in the $\mathbf{U}$ sub-problem out of the k^{th} outer iteration counter of the ADMM update. It is straightforward that this is computationally expensive. To address this, we propose to solve the $\mathbf{U}$ sub-problem approximately using just a one-step proximal gradient update and prove convergence in the next section. This approach is based on proximal linearized ADMM (Liu et al., 2013; Lu et al., 2016), which solves the sub-problems efficiently by linearizing the differentiable part and then applying proximal gradient due to the non-differentiable part. To ensure convergence, the algorithm requires that gradient should be Lipschitz continuous. The $\mathbf{V}$ and $\mathbf{\Lambda}$ updates are just the same as in regular convex clustering.

We adopt such an approach and develop the proximal linearized 2-block ADMM (Algorithm 7) to solve Gecco+ when the loss is differentiable and gradient is Lipschitz continuous.

Algorithm 7

Proximal linearized 2-block ADMM when the loss is differentiable and gradient is Lipschitz continuous — matrix form

while not converged **do**

$$\mathbf{U}^{(k)} = \text{prox}_{s_k \cdot \alpha P_2(\cdot;\zeta)}(\mathbf{U}^{(k-1)} - \widetilde{\mathbf{X}} - s_k \cdot [\nabla \ell(\mathbf{X}, \mathbf{U}^{(k-1)}) + \rho \mathbf{D}^T(\mathbf{D}\mathbf{U}^{(k-1)} - \mathbf{V}^{(k-1)} + \mathbf{\Lambda}^{(k-1)})]) + \widetilde{\mathbf{X}}$$

$$\mathbf{V}^{(k)} = \text{prox}_{\gamma/\rho P_1(\cdot;\mathbf{w})}(\mathbf{D}\mathbf{U}^{(k)} + \mathbf{\Lambda}^{(k-1)})$$

$$\mathbf{\Lambda}^{(k)} = \mathbf{\Lambda}^{(k-1)} + (\mathbf{D}\mathbf{U}^{(k)} - \mathbf{V}^{(k)})$$

end while

Further, if the $\mathbf{U}$ sub-problem can be decomposed to p separate $\mathbf{U}_{\cdot j}$ sub-problems where the augmented Lagrangian for each now is a sum of a differentiable loss, a quadratic term and a sparse group-lasso penalty, we propose to use proximal gradient descent for each separate $\mathbf{U}_{\cdot j}$ sub-problem. In this way, we yield adaptive step size for each $\mathbf{U}_{\cdot j}$ sub-problem and hence our algorithm enjoys better convergence property than updating $\mathbf{U}$'s all together. (In the latter case, the step size becomes fairly small as we are moving all $\mathbf{U}$ to some magnitude in the direction of negative gradient.) To achieve this, we assume that the loss is elementwise, which means we can write the loss function as a sum of p terms. (The loss can be written as $\sum_i \ell(\mathbf{X}_{i|}|\mathbf{U}_{i|}) = \sum_j \ell(\mathbf{X}_{\cdot j}, \mathbf{U}_{\cdot j}) = \sum_i \sum_j q(x_{ij}|u_{ij})$ where $q(\cdot)$ is the element-wise version of the loss while $\ell(\cdot)$ is the vector-wise version of the loss.) We see that every deviance-based loss satisfies this assumption. Moreover, by decomposing to p sub-problems, we can solve each in parallel which saves computation cost. We describe in detail how to solve each $\mathbf{U}_{\cdot j}$ sub-problem in the next subsection.

C.2 Two-block ADMM in Vector Form in Parallel

Suppose the $\mathbf{U}$ sub-problem can be decomposed to p separate $\mathbf{U}_{\cdot j}$ sub-problems mentioned above. The augmented Lagrangian now becomes:

$$L(\mathbf{U},\mathbf{V},\mathbf{\Lambda}) = \sum_{j=1}^{p} \ell(\mathbf{X}_{\cdot j}, \mathbf{U}_{\cdot j}) + \frac{\rho}{2}\sum_{j=1}^{p} \|\mathbf{D}\mathbf{U}_{\cdot j} - \mathbf{V}_{\cdot j} + \mathbf{\Lambda}_{\cdot j}\|_2^2 + \alpha\sum_{j=1}^{p} \zeta_j \|\mathbf{U}_{\cdot j} - \tilde{x}_j \mathbf{1}_n\|_2 + \gamma \sum_{l \in \varepsilon} w_l \|\mathbf{V}_l\|_2 .$$

In this way we can perform block-wise minimization. Now minimizing the augmented Lagrangian over $\mathbf{U}$ is equivalent to minimizing over each $\mathbf{U}_{\cdot j}, j = 1, \cdots, p$:

$$\underset{\mathbf{U}_{\cdot j}}{\text{minimize}} \quad \ell(\mathbf{X}_{\cdot j}, \mathbf{U}_{\cdot j}) + \frac{\rho}{2}\|\mathbf{D}\mathbf{U}_{\cdot j} - \mathbf{V}_{\cdot j} + \mathbf{\Lambda}_{\cdot j}\|_2^2 + \alpha\zeta_j \|\mathbf{U}_{\cdot j} - \tilde{x}_j \cdot \mathbf{1}_n\|_2 .$$

Let $\tilde{\mathbf{u}}_j = \mathbf{U}_{\cdot j} - \tilde{x}_j \cdot \mathbf{1}_n$. The problem above becomes:

$$\underset{\tilde{\mathbf{u}}_j}{\text{minimize}} \quad \ell(\mathbf{X}_{\cdot j}, \tilde{\mathbf{u}}_j + \tilde{x}_j \cdot \mathbf{1}_n) + \frac{\rho}{2}\|\mathbf{D}(\tilde{\mathbf{u}}_j + \tilde{x}_j \cdot \mathbf{1}_n) - \mathbf{V}_{\cdot j} + \mathbf{\Lambda}_{\cdot j}\|_2^2 + \alpha\zeta_j \|\tilde{\mathbf{u}}_j\|_2 .$$

Similarly, this can be solved by running iterative proximal gradient to full convergence. However, as mentioned above, we propose to solve the $\mathbf{U}$ sub-problem approximately by taking just a one-step proximal gradient update and prove convergence. Still this approach is based on proximal linearized ADMM (Liu et al., 2013; Lu et al., 2016), which solves the sub-problems efficiently by linearizing the differentiable part and then applying proximal gradient due to the non-differentiable part. To ensure convergence, the algorithm requires that gradient should be Lipschitz continuous.

We propose Algorithm 8 to solve Gecco+ when ℓ is differentiable and gradient is Lipschitz continuous in vector form. Note the $\mathbf{U}$-update in Algorithm 8 is a just a vectorized version of that in Algorithm 7 if we use fixed step size s_k for each feature j. We use the vector form update here since it enjoys better convergence property mentioned above and we use this form to prove convergence. Next we prove the convergence of Algorithm 8.

Algorithm 8

Proximal linearized 2-block ADMM when the loss is differentiable and gradient is Lipschitz continuous — vector form in parallel

---

Input: $\mathbf{X}, \gamma, \mathbf{w}, \alpha, \zeta$

Initialize: $\mathbf{U}^{(0)}, \mathbf{V}^{(0)}, \mathbf{\Lambda}^{(0)}$

Precompute: Difference matrix $\mathbf{D}, \tilde{x}_j$

while not converged **do**

for $j = 1$ **to** p **do**

$$\mathbf{U}_{\cdot j}^{(k)} = \text{prox}_{s_k \cdot \alpha\zeta_j \|\cdot\|_2}(\mathbf{U}_{\cdot j}^{(k-1)} - \tilde{x}_j \cdot \mathbf{1}_n - s_k \cdot [\nabla \ell(\mathbf{X}_{\cdot j}, \mathbf{U}_{\cdot j}^{(k-1)}) + \rho \mathbf{D}^T(\mathbf{D}\mathbf{U}_{\cdot j}^{(k-1)} - \mathbf{V}_{\cdot j}^{(k-1)} + \mathbf{\Lambda}_{\cdot j}^{(k-1)})]) + \tilde{x}_j \cdot \mathbf{1}_n$$

end for

$$\mathbf{V}^{(k)} = \text{prox}_{\gamma/\rho P_1(\cdot;\mathbf{w})}(\mathbf{D}\mathbf{U}^{(k)} + \mathbf{\Lambda}^{(k-1)})$$

$$\mathbf{\Lambda}^{(k)} = \mathbf{\Lambda}^{(k-1)} + (\mathbf{D}\mathbf{U}^{(k)} - \mathbf{V}^{(k)})$$

end while

Output: $\mathbf{U}^{(k)}$.

C.3 Proof of Convergence

Theorem 6 *If ℓ is convex and differrentiable and $\nabla\ell$ is Lipschitz continuous, then* Algorithm 8 *converges to a global solution. Further, if ℓ is strictly convex, it converges to the unique global solution.*

Proof: We will show that the **U**-update in Algorithm 8 is equivalent to linearizing the **U** sub-problem and then applying a proximal operator, which is proximal linearized ADMM.

Note that each $\mathbf{U}_{\cdot j}$ sub-problem is:

$$\mathbf{U}_{\cdot j}^{(k+1)} = \underset{\mathbf{U}_{\cdot j}}{\text{argmin}} \quad \ell(\mathbf{X}_{\cdot j}, \mathbf{U}_{\cdot j}) + \frac{\rho}{2}\|\mathbf{D}\mathbf{U}_{\cdot j} - \mathbf{V}_{\cdot j}^{(k)} + \mathbf{\Lambda}_{\cdot j}^{(k)}\|_2^2 + \alpha\zeta_j \|\mathbf{U}_{\cdot j} - \tilde{x}_j \cdot \mathbf{1}_n\|_2 \ .$$

For simplicity of notation, we replace $\mathbf{U}_{\cdot j}$ with $\mathbf{u}_j$ in the following:

$$\mathbf{u}_j^{(k+1)} = \underset{\mathbf{u}_j}{\text{argmin}} \quad \ell(\mathbf{X}_{\cdot j}, \mathbf{u}_j) + \frac{\rho}{2}\|\mathbf{D}\mathbf{u}_j - \mathbf{V}_{\cdot j}^{(k)} + \mathbf{\Lambda}_{\cdot j}^{(k)}\|_2^2 + \alpha\zeta_j \|\mathbf{u}_j - \tilde{x}_j \cdot \mathbf{1}_n\|_2 \ .$$

Rearranging terms, we have:

$$\mathbf{u}_j^{(k+1)} = \underset{\mathbf{u}_j}{\text{argmin}} \quad \ell(\mathbf{X}_{\cdot j}, \mathbf{u}_j) + \frac{\rho}{2}\|\mathbf{D}\mathbf{u}_j - \mathbf{V}_{\cdot j}^{(k)}\|_2^2 + \rho\mathbf{\Lambda}_{\cdot j}^{(k)T}(\mathbf{D}\mathbf{u}_j - \mathbf{V}_{\cdot j}^{(k)}) + \alpha\zeta_j \|\mathbf{u}_j - \tilde{x}_j \cdot \mathbf{1}_n\|_2 \ .$$

According to the proximal linearized ADMM with parallel splitting algorithm (see Algorithm 3 of Liu et al. (2013) and Equation (14) of Lu et al. (2016)), we can linearize the first two terms and add a quadratic term in the objective:

$$\mathbf{u}_j^{(k+1)} = \underset{\mathbf{u}_j}{\operatorname{argmin}} \quad \ell(\mathbf{X}_{\cdot j}, \mathbf{u}_j^{(k)}) + \langle \nabla \ell(\mathbf{X}_{\cdot j}, \mathbf{u}_j^{(k)}), \mathbf{u}_j - \mathbf{u}_j^{(k)} \rangle + \langle \rho \mathbf{D}^T (\mathbf{D}\mathbf{u}_j^{(k)} - \mathbf{V}_{\cdot j}^{(k)}), \mathbf{u}_j - \mathbf{u}_j^{(k)} \rangle + \rho \mathbf{\Lambda}_{\cdot j}^{(k)T} (\mathbf{D}\mathbf{u}_j - \mathbf{V}_{\cdot j}^{(k)}) + \alpha \zeta_j \|\mathbf{u}_j - \tilde{x}_j \cdot \mathbf{1}_n\|_2 + \frac{1}{2s_k} \|\mathbf{u}_j - \mathbf{u}_j^{(k)}\|_2^2 .$$

Rearranging terms and removing irrelevant terms, we have:

$$\mathbf{u}_j^{(k+1)} = \underset{\mathbf{u}_j}{\operatorname{argmin}} \quad (\nabla g(\mathbf{u}_j^{(k)}))^T (\mathbf{u}_j - \mathbf{u}_j^{(k)}) + \frac{1}{2s_k} \|\mathbf{u}_j - \mathbf{u}_j^{(k)}\|_2^2 + \alpha \zeta_j \|\mathbf{u}_j - \tilde{x}_j \cdot \mathbf{1}_n\|_2 ,$$

where $\nabla g(\mathbf{u}_j^{(k)}) = \nabla \ell(\mathbf{X}_{\cdot j}, \mathbf{u}_j^{(k)}) + \rho \mathbf{D}^T (\mathbf{D}\mathbf{u}_j^{(k)} - \mathbf{V}_{\cdot j}^{(k)} + \mathbf{\Lambda}_{\cdot j}^{(k)})$.

Let $\hat{\mathbf{u}}_j = \mathbf{u}_j - \tilde{x}_j \cdot \mathbf{1}_n$. We have:

$$\hat{\mathbf{u}}_j^{(k+1)} = \underset{\hat{\mathbf{u}}_j}{\operatorname{argmin}} \quad (\nabla g(\mathbf{u}_j^{(k)}))^T (\hat{\mathbf{u}}_j + \tilde{x}_j \cdot \mathbf{1}_n - \mathbf{u}_j^{(k)}) + \frac{1}{2s_k} \|\hat{\mathbf{u}}_j + \tilde{x}_j \cdot \mathbf{1}_n - \mathbf{u}_j^{(k)}\|_2^2 + \alpha \zeta_j \|\hat{\mathbf{u}}_j\|_2 .$$

Recall the definition of proximal operator:

$$\mathbf{x}^{(k+1)} = \underset{th}{\operatorname{prox}}(\mathbf{x}^{(k)} - t \nabla g(\mathbf{x}^{(k)})) = \underset{\mathbf{u}}{\operatorname{argmin}}(h(\mathbf{u}) + g(\mathbf{x}^{(k)}) + \nabla g(\mathbf{x}^{(k)})^T (\mathbf{u} - \mathbf{x}^{(k)}) + \frac{1}{2t} \|\mathbf{u} - \mathbf{x}^{(k)}\|_2^2) .$$

Therefore, the $\hat{\mathbf{u}}_j$ update is just a proximal gradient descent update:

$$\hat{\mathbf{u}}_j^{(k+1)} = \underset{s_k \cdot \alpha \zeta_j \| \cdot \|_2}{\operatorname{prox}} (\mathbf{u}_j^{(k)} - \tilde{x}_j \cdot \mathbf{1}_n - s_k \cdot [\nabla \ell(\mathbf{X}_{\cdot j}, \mathbf{u}_j^{(k)}) + \rho \mathbf{D}^T (\mathbf{D}\mathbf{u}_j^{(k)} - \mathbf{V}_{\cdot j}^{(k)} + \mathbf{\Lambda}_{\cdot j}^{(k)})]) .$$

Now we plug back and get the $\mathbf{u}_j$, i.e., $\mathbf{U}_{\cdot j}$ update:

$$\mathbf{U}_{\cdot j}^{(k)} = \underset{s_k \cdot \alpha \zeta_j \| \cdot \|_2}{\operatorname{prox}} (\mathbf{U}_{\cdot j}^{(k-1)} - \tilde{x}_j \cdot \mathbf{1}_n - s_k \cdot [\nabla \ell(\mathbf{X}_{\cdot j}, \mathbf{U}_{\cdot j}^{(k-1)}) + \rho \mathbf{D}^T (\mathbf{D}\mathbf{U}_{\cdot j}^{(k-1)} - \mathbf{V}_{\cdot j}^{(k-1)} + \mathbf{\Lambda}_{\cdot j}^{(k-1)})]) + \tilde{x}_j \cdot \mathbf{1}_n ,$$

which is equivalent to the $\mathbf{U}_{\cdot j}$ update in Algorithm 8.

The $\mathbf{V}$ and $\mathbf{\Lambda}$ update is just the same as the one in convex clustering. Hence Algorithm 8 is equivalent to the form of proximal linearized ADMM by Liu et al. (2013); Lu et al. (2016) and hence converges to a global solution as long as $\nabla \ell$ is Lipschitz continuous. Note that Algorithm 7 is equivalent to Algorithm 8 with a fixed step size s_k. (We can choose s_k to be the minimum step size s_k over all features.) Therefore, Algorithm 7 also converges to

a global solution as long as $\nabla \ell$ is Lipschitz continuous. Further, if ℓ is strictly convex, the optimization problem has unique minimum and hence Algorithm 7 and 8 converges to the global solution. ■

Note:

- In Liu et al. (2013); Lu et al. (2016), the algorithm requires that $\nabla \ell$ is Lipschitz continuous to guarantee convergence. We know that $\nabla \ell$ being Lipschitz continuous is equivalent to ℓ being strongly smooth. It is easy to show that the Hessian of log-likelihood of exponential family and GLM deviance is upper bounded since in (generalized) convex clustering, the value of $\mathbf{U}$ is bounded as it moves along the regularization path from $\mathbf{X}$ to the loss-specific center; also to avoid numerical issues, we add trivial constraint that $u_{ij} > 0$ when zero is not defined in the log-likelihood/deviance. Hence the condition for convergence of proximal linearized ADMM is satisfied.
- To obtain a reasonable step size s_k, we need to compute the Lipschitz constant. However, it is non-trivial to calculate the Lipschitz constant for most of our general losses. Instead, we suggest using backtracking line search procedure proposed by Beck and Teboulle (2009); Parikh et al. (2014), which is a common way to determine step size with guaranteed convergence in optimization. Empirical studies show that choosing step size with backtracking in our framework also ensures convergence. The details for backtracking procedure are discussed below.
- For proximal linearized ADMM, Liu et al. (2013); Lu et al. (2016) established convergence rate of $O(1/K)$. An interesting future direction might be establishing the linear convergence rate of proximal linearized ADMM when the objective is strongly convex.

C.4 Backtracking Criterion

In this section we discuss how to choose the step size s_k in Algorithm 8. As mentioned, usually we employ a fixed step size by computing the Lipschitz constant as in the squared error loss case; but in our method, it is hard to compute the Lipschitz constant for most of our general losses. Instead, we propose using backtracking line search procedure proposed by Beck and Teboulle (2009); Parikh et al. (2014), which is a common way to determine step size with guaranteed convergence in optimization.

Recall the objective function we want to minimize in the $\mathbf{U}$ sub-problem is:

$$f(\tilde{\mathbf{u}}_j) = \ell(\mathbf{X}_{\cdot j}, \tilde{\mathbf{u}}_j + \tilde{x}_j \cdot \mathbf{1}_n) + \frac{\rho}{2} \|\mathbf{D}(\tilde{\mathbf{u}}_j + \tilde{x}_j \cdot \mathbf{1}_n) - \mathbf{V}_{\cdot j} + \mathbf{\Lambda}_{\cdot j}\|_2^2 + \alpha \zeta_j \|\tilde{\mathbf{u}}_j\|_2 ,$$

where $\tilde{\mathbf{u}}_j = \mathbf{U}_{\cdot j} - \tilde{x}_j \cdot \mathbf{1}_n$.

Define:

$$g(\tilde{\mathbf{u}}_j) = \ell(\mathbf{X}_{\cdot j}, \tilde{\mathbf{u}}_j + \tilde{x}_j \cdot \mathbf{1}_n) + \frac{\rho}{2}\|\mathbf{D}(\tilde{\mathbf{u}}_j + \tilde{x}_j \cdot \mathbf{1}_n) - \mathbf{V}_{\cdot j} + \mathbf{\Lambda}_{\cdot j}\|_2^2$$
$$h(\tilde{\mathbf{u}}_j) = \alpha\zeta_j\|\tilde{\mathbf{u}}_j\|_2$$
$$G_t(\tilde{\mathbf{u}}_j) = \frac{\tilde{\mathbf{u}}_j - \text{prox}_{t \cdot \alpha\zeta_j\| \cdot \|_2}(\tilde{\mathbf{u}}_j - t\nabla g(\tilde{\mathbf{u}}_j))}{t}.$$

We adopt the backtracking line search procedure proposed by Beck and Teboulle (2009); Parikh et al. (2014). At each iteration, while

$$g(\tilde{\mathbf{u}}_j - tG_t(\tilde{\mathbf{u}}_j)) > g(\tilde{\mathbf{u}}_j) - t\nabla g(\tilde{\mathbf{u}}_j)^T G_t(\tilde{\mathbf{u}}_j) + \frac{t}{2}\|G_t(\tilde{\mathbf{u}}_j)\|_2^2 \quad \text{i.e.,}$$
$$g(\underset{t}{\text{prox}}(\tilde{\mathbf{u}}_j - t\nabla g(\tilde{\mathbf{u}}_j))) > g(\tilde{\mathbf{u}}_j) - \nabla g(\tilde{\mathbf{u}}_j)^T(\tilde{\mathbf{u}}_j - \underset{t}{\text{prox}}(\tilde{\mathbf{u}}_j - t\nabla g(\tilde{\mathbf{u}}_j)))$$
$$+\frac{1}{2t}\|\tilde{\mathbf{u}}_j - \underset{t}{\text{prox}}(\tilde{\mathbf{u}}_j - t\nabla g(\tilde{\mathbf{u}}_j))\|_2^2$$

shrink $t = \beta t$.

We still adopt the one-step approximation and hence suggest taking a one-step proximal update with backtracking to solve the **U** sub-problem. To summarize, we propose Algorithm 9, which uses proximal linearized 2-block ADMM with backtracking when the loss is differentiable and gradient is Lipschitz continuous.

C.5 Alternative Algorithm for Differentiable Losses

It should be pointed out that there are many other methods to solve the **U** sub-problem when the loss ℓ is differentiable. We choose to use proximal gradient descent algorithm as there is existing literature on approximately solving the sub-problem using proximal gradient under ADMM with proved convergence (Liu et al., 2013; Lu et al., 2016). But there are many other optimization techniques to solve the **U** sub-problem such as ADMM.

In this subsection, we show how to apply ADMM to solve the **U** sub-problem and specify under which conditions this method is more favorable. Recall to update **U**, we need to solve the following sub-problem:

$$\underset{\mathbf{U}}{\text{minimize}} \quad \ell(\mathbf{X}, \mathbf{U}) + \frac{\rho}{2}\|\mathbf{D}\mathbf{U} - \mathbf{V} + \mathbf{\Lambda}\|_F^2 + \alpha\sum_{j=1}^{p}\zeta_j\|\mathbf{U}_{\lfloor j} - \tilde{x}_j\rfloor \mathbf{1}_n\|_2 .$$

We use ADMM to solve this minimization problem and can now recast the problem above as the equivalent constrained problem:

$$\underset{\mathbf{U},\mathbf{V},\mathbf{\Lambda},\mathbf{R}}{\text{minimize}} \quad \sum_{j=1}^{p}\ell(\mathbf{X}_{\lfloor j}, \mathbf{U}_{\lfloor j}) + \frac{\rho}{2}\|\mathbf{D}\mathbf{U} - \mathbf{V} + \mathbf{\Lambda}\|_F^2 + \underbrace{\alpha\left(\sum_{j=1}^{p}\zeta_j\|\mathbf{r}_j\|_2\right)}_{P_2(\mathbf{R};\zeta)}$$
$$\text{subject to} \quad \mathbf{U} - \widetilde{\mathbf{X}} = \mathbf{R} .$$

Algorithm 9

Proximal linearized 2-block ADMM with backtracking when the loss is differentiable and gradient is Lipschitz continuous

Input: $\mathbf{X}, \gamma, \mathbf{w}, \alpha, \zeta$

Initialize: $\mathbf{U}^{(0)}, \mathbf{V}^{(0)}, \mathbf{\Lambda}^{(0)}, t$

Precompute: Difference matrix $\mathbf{D}$, $\tilde{x}_j$

while not converged **do**

for $j = 1$ **to** p **do**

$t = 1$

$\tilde{\mathbf{u}}_j^{(k-1)} = \mathbf{U}_{\cdot j}^{(k-1)} - \tilde{x}_j \cdot \mathbf{1}_n$

$\nabla g(\tilde{\mathbf{u}}_j^{(k-1)}) = \nabla \ell(\mathbf{X}_{\cdot j}, \tilde{\mathbf{u}}_j^{(k-1)} + \tilde{x}_j \cdot \mathbf{1}_n) + \rho \mathbf{D}^T(\mathbf{D}(\tilde{\mathbf{u}}_j^{(k-1)} + \tilde{x}_j \cdot \mathbf{1}_n) - \mathbf{V}_{\cdot j}^{(k-1)} + \mathbf{\Lambda}_{\cdot j}^{(k-1)})$

$\mathbf{z} = \text{prox}_{t\alpha\zeta_j \| \cdot \|_2}(\tilde{\mathbf{u}}_j^{(k-1)} - t \nabla g(\tilde{\mathbf{u}}_j^{(k-1)}))$

while $g(\mathbf{z}) > g(\tilde{\mathbf{u}}_j^{(k-1)} - \nabla g(\tilde{\mathbf{u}}_j^{(k-1)})^T(\tilde{\mathbf{u}}_j^{(k-1)} - \mathbf{z}) + \frac{1}{2t}\|\mathbf{z} - \tilde{\mathbf{u}}_j^{(k-1)}\|_2^2$ **do**

$t = \beta t$

$\mathbf{z} = \text{prox}_{t\alpha\zeta_j \| \cdot \|_2}(\tilde{\mathbf{u}}_j^{(k-1)} - t \nabla g(\tilde{\mathbf{u}}_j^{(k-1)}))$

end while

$\mathbf{U}_{\cdot j}^{(k)} = \mathbf{z} + \tilde{x}_j \cdot \mathbf{1}_n$

end for

$\mathbf{V}^{(k)} = \text{prox}_{\gamma/\rho P_1(\cdot; \mathbf{w})}(\mathbf{D}\mathbf{U}^{(k)} + \mathbf{\Lambda}^{(k-1)})$

$\mathbf{\Lambda}^{(k)} = \mathbf{\Lambda}^{(k-1)} + (\mathbf{D}\mathbf{U}^{(k)} - \mathbf{V}^{(k)})$

end while

Output: $\mathbf{U}^{(k)}$.

The augmented Lagrangian in scaled form is:

$$L(\mathbf{U}, \mathbf{V}, \mathbf{R}, \mathbf{\Lambda}, \mathbf{N}) = \sum_{j=1}^{p} \ell(\mathbf{X}_{\mid j}, \mathbf{U}_{\mid j}) + \frac{\rho}{2}\|\mathbf{D}\mathbf{U} - \mathbf{V} + \mathbf{\Lambda}\|_F^2 + \frac{\rho}{2}\|(\mathbf{U} - \widetilde{\mathbf{X}}) - \mathbf{R} + \mathbf{N}\|_F^2 + \alpha \sum_{j=1}^{p} \zeta_j \|\mathbf{r}_j\|_2 ,$$

where the dual variable for $\mathbf{V}$ is denoted by $\mathbf{\Lambda}$; the dual variable for $\mathbf{R}$ is denoted by $\mathbf{N}$.

The $\mathbf{U}_{\cdot j}$ sub-problem in the inner nested ADMM is:

$$\mathbf{U}_{\cdot j}^{(k)} = \underset{\mathbf{U}_{\cdot j}}{\operatorname{argmin}} \quad \ell(\mathbf{X}_{\cdot j}, \mathbf{U}_{\cdot j}) + \frac{\rho}{2}\|\mathbf{D}\mathbf{U}_{\cdot j} - \mathbf{V}_{\cdot j}^{(k-1)} + \mathbf{\Lambda}_{\cdot j}^{(k-1)}\|_F^2 + \frac{\rho}{2}\|(\mathbf{U}_{\cdot j} - \tilde{x}_j \cdot \mathbf{1}_n) - \mathbf{r}_j^{(k-1)} + \mathbf{N}_{\cdot j}^{(k-1)}\|_F^2.$$

Now, we are minimizing a sum of a differentiable loss ℓ and two quadratic terms which are all smooth. Still the $\mathbf{U}_{\cdot j}$ sub-problem does not have a closed-form solution for general convex losses and we need to run an iterative descent algorithm (such as gradient descent, Newton method) to full convergence to solve the problem. Similarly, to reduce computation cost, we take a one-step update by applying linearized ADMM (Lin et al., 2011) to the $\mathbf{U}_{\cdot j}$

Algorithm 10

Full-step multi-block algorithm for Gecco+ with Bernoulli log-likelihood ℓ_k:

Precompute: Difference matrix $\mathbf{D}, \mathbf{M}_1 = \left(\frac{1}{4}\mathbf{I} + \rho\mathbf{D}^T\mathbf{D} + \rho\mathbf{I}\right)^{-1}$.

while not converged **do**

  while not converged **do**

$$\mathbf{U}^{(k)} = \mathbf{U}^{(k-1)} - \mathbf{M}_1[\nabla\ell(\mathbf{X},\mathbf{U}^{(k-1)}) + \rho\mathbf{D}^T(\mathbf{D}\mathbf{U}^{(k-1)} - \mathbf{V}^{(k-1)} + \mathbf{\Lambda}^{(k-1)}) + \rho(\mathbf{U}^{(k-1)} - \widetilde{\mathbf{X}} - \mathbf{R}^{(k-1)} + \mathbf{N}^{(k-1)})]$$

$$\mathbf{R}^{(k)} = \operatorname{prox}_{\alpha/\rho P_2(\cdot;\zeta)}(\mathbf{U}^{(k)} - \widetilde{\mathbf{X}} + \mathbf{N}^{(k-1)})$$

$$\mathbf{N}^{(k)} = \mathbf{N}^{(k-1)} + (\mathbf{U}^{(k)} - \widetilde{\mathbf{X}} - \mathbf{R}^{(k)})$$

  end while

$$\mathbf{V}^{(k)} = \operatorname{prox}_{\gamma/\rho P_1(\cdot;\mathbf{w})}(\mathbf{D}\mathbf{U}^{(k)} + \mathbf{\Lambda}^{(k-1)})$$

$$\mathbf{\Lambda}^{(k)} = \mathbf{\Lambda}^{(k-1)} + (\mathbf{D}\mathbf{U}^{(k)} - \mathbf{V}^{(k)})$$

end while

sub-problem. The $\mathbf{U}_{\cdot j}$ update in the inner ADMM now becomes:

$$\mathbf{U}_{\cdot j}^{(k)} = \mathbf{U}_{\cdot j}^{(k-1)} - s_k[\nabla\ell(\mathbf{X}_{\cdot j}, \mathbf{U}_{\cdot j}^{(k-1)}) + \rho\mathbf{D}^T(\mathbf{D}\mathbf{U}_{\cdot j}^{(k-1)} - \mathbf{V}_{\cdot j}^{(k-1)} + \mathbf{\Lambda}_{\cdot j}^{(k-1)}) + \rho(\mathbf{U}_{\cdot j}^{(k-1)} - \tilde{x}_j \cdot \mathbf{1}_n - \mathbf{r}_j^{(k-1)} + \mathbf{N}_{\cdot j}^{(k-1)})] .$$

In this case, empirical studies show that taking a one-step Newton update is favored than a one-step gradient descent update as the former enjoys better convergence properties and generally avoids backtracking. However, inverting a Hessian matrix is computationally burdensome at each iteration when n is large. Exceptions are for Euclidean distances case where there is a closed-form solution for the $\mathbf{U}_{\cdot j}$ update and for Bernoulli log-likelihood case where the Hessian of the loss can be upper bounded by a fixed matrix. In the latter case, we propose to pre-compute the inverse of that fixed matrix instead of inverting a Hessian matrix at each iteration. To illustrate this, we write out the $\mathbf{U}_{\cdot j}$ sub-problem of Gecco+ with Bernoulli log-likelihood:

$$\mathbf{U}_{\cdot j} = \underset{\mathbf{U}_{\cdot j}}{\operatorname{argmin}} \left(\sum_{i=1}^{n} -x_{ij}u_{ij} + \log\left(1 + e^{u_{ij}}\right) \right) + \frac{\rho}{2} \|\mathbf{D}\mathbf{U}_{\cdot j} - \mathbf{V}_{\cdot j} + \mathbf{\Lambda}_{\cdot j}\|_F^2 + \frac{\rho}{2} \|(\mathbf{U}_{\cdot j} - \tilde{x}_j \cdot \mathbf{1}_n) - \mathbf{r}_j + \mathbf{N}_{\cdot j}\|_F^2 .$$

The Hessian is $\operatorname{diag}\left\{ \frac{e^{u_{ij}}}{\left(1 + e^{u_{ij}}\right)^2} \right\} + \rho \mathbf{D}^T\mathbf{D} + \rho\mathbf{I}$ which can be upper bounded by $\frac{1}{4}\mathbf{I} + \rho\mathbf{D}^T\mathbf{D} + \rho\mathbf{I}$. We propose to replace the Hessian with this fixed matrix in Newton method and use its inverse. This is closely related to the approximate Hessian literature (Krishnapuram et al., 2005; Simon et al., 2013). In this way, we just pre-compute this inverse matrix instead of inverting the Hessian matrix at each iteration, which dramatically saves computation. We give Algorithm 10 to solve Gecco+ for Bernoulli log-likelihood with a one-step update to solve the $\mathbf{U}$ sub-problem. Empirical studies show that this is faster than taking the inner nested proximal gradient approach as we generally don't need to perform the backtracking step.

Yet, Algorithm 10 is slow as we need to run iterative inner nested ADMM updates to full convergence. To address this, as mentioned, we can take a one-step update of the inner nested iterative ADMM algorithm. To see this, we can recast the original Gecco+ problem as:

$$\begin{aligned} \underset{\mathbf{U}^{(k)}, \mathbf{V}}{\text{minimize}} \quad & \ell(\mathbf{X}, \mathbf{U}) + \underbrace{\gamma \left(\sum_{l \in \varepsilon} w_l \|\mathbf{V}_l\|_2 \right)}_{P_1(\widetilde{\mathbf{V}}; \mathbf{w})} + \underbrace{\alpha \left(\sum_{j=1}^{p} \zeta_j \|\mathbf{r}_j\|_2 \right)}_{P_2(\widetilde{\mathbf{R}}; \zeta)} \\ \text{subject to} \quad & \mathbf{D}\mathbf{U} - \mathbf{V} = 0, \quad \mathbf{U} - \widetilde{\mathbf{X}} = \mathbf{R} . \end{aligned}$$

We apply multi-block ADMM to solve this optimization problem and hence get Algorithm 11. As discussed above, we take a one-step update to solve the $\mathbf{U}$ sub-problem with linearized ADMM and use the inverse of fixed approximate Hessian matrix, $\mathbf{M}_1$.

Algorithm 11

One-step inexact multi-block algorithm for Gecco+ with Bernoulli log-likelihood ℓ_k:

Precompute: Difference matrix $\mathbf{D}, \mathbf{M}_1 = \left(\frac{1}{4}\mathbf{I} + \rho\mathbf{D}^T\mathbf{D} + \rho\mathbf{I}\right)^{-1}$.

while not converged **do**

$$\mathbf{U}^{(k)} = \mathbf{U}^{(k-1)} - \mathbf{M}_1 \left[\nabla \ell(\mathbf{X}, \mathbf{U}^{(k-1)}) + \rho\mathbf{D}^T(\mathbf{D}\mathbf{U}^{(k-1)} - \mathbf{V}^{(k-1)} + \mathbf{\Lambda}^{(k-1)}) + \rho(\mathbf{U}^{(k-1)}) - \widetilde{\mathbf{X}} - \mathbf{R}^{(k-1)} + \mathbf{N}^{(k-1)}) \right]$$

$$\mathbf{R}^{(k)} = \operatorname{prox}_{\alpha/\rho P_2(\cdot\,; \zeta)}(\mathbf{U}^{(k)} - \widetilde{\mathbf{X}} + \mathbf{N}^{(k-1)})$$

$$\mathbf{N}^{(k)} = N^{(k-1)} + (\mathbf{U}^{(k)} - \widetilde{\mathbf{X}} - \mathbf{R}^{(k)})$$

$$\mathbf{V}^{(k)} = \operatorname{prox}_{\gamma/\rho P_1(\cdot\,; \mathbf{w})}(\mathbf{D}\mathbf{U}^{(k)} + \mathbf{\Lambda}^{(k-1)})$$

$$\mathbf{\Lambda}^{(k)} = \mathbf{\Lambda}^{(k-1)} + (\mathbf{D}\mathbf{U}^{(k)} - \mathbf{V}^{(k)})$$

end while

Similarly, we adopt this approach to solve Gecco+ with Euclidean distances (sparse convex clustering). We first recast the original problem as:

$$\begin{aligned} \underset{\mathbf{U}^{(k)},\mathbf{V}}{\text{minimize}} \quad & \tfrac{1}{2}\|\mathbf{X}-\mathbf{U}\|_2^2 + \gamma \underbrace{\left(\sum_{l\in\varepsilon} w_l \|\mathbf{V}_l\|_2\right)}_{P_1(\mathbf{V};\mathbf{w})} + \alpha \underbrace{\left(\sum_{j=1}^{p} \zeta_j \|\mathbf{r}_j\|_2\right)}_{P_2(\mathbf{R};\zeta)} \\ \text{subject to} \quad & \mathbf{D}\mathbf{U}-\mathbf{V}=0, \quad \mathbf{U}-\widetilde{\mathbf{X}}=\mathbf{R}\ . \end{aligned}$$

Still, we use multi-block ADMM to solve this optimization problem and hence get Algorithm 12. Note that $\mathbf{U}$ sub-problem now has a closed-form solution. Typically, we do not have an approximate Hessian matrix or a closed-form solution to the sub-problem for general losses and we have to use one-step gradient descent with backtracking to solve the $\mathbf{U}$ sub-problem. Empirical study shows that this approach converges slower than Algorithm 9 which uses one-step proximal gradient descent with backtracking.

Appendix D.: Gecco+ for Non-Differentiable Losses

In this section, we propose an algorithm to solve Gecco+ when the loss ℓ is non-differentiable. In this case, we develop a multi-block ADMM algorithm to solve Gecco+ and prove its algorithmic convergence.

Algorithm 12

One-step inexact multi-block algorithm for Gecco+ with Euclidean distances ℓ_k:

Precompute: Difference matrix $\mathbf{D}, \mathbf{M}_2 = (\mathbf{I}+\rho\mathbf{D}^T\mathbf{D}+\rho\mathbf{I})^{-1}$.

while not converged **do**

$$\mathbf{U}^{(k)} = \mathbf{M}_2[\mathbf{X}+\rho\mathbf{D}^T(\mathbf{V}^{(k-1)} - \mathbf{\Lambda}^{(k-1)}) + \rho(\widetilde{\mathbf{X}} + \mathbf{R}^{(k-1)} - \mathbf{N}^{(k-1)})]$$

$$\mathbf{R}^{(k)} = \mathrm{prox}_{\alpha/\rho P_2(\cdot\,;\zeta)}(\mathbf{U}^{(k)} - \widetilde{\mathbf{X}} + \mathbf{N}^{(k-1)})$$

$$\mathbf{N}^{(k)} = \mathbf{N}^{(k-1)} + (\mathbf{U}^{(k)} - \widetilde{\mathbf{X}} - \mathbf{R}^{(k)})$$

$$\mathbf{V}^{(k)} = \mathrm{prox}_{\gamma/\rho P_1(\cdot\,;\mathbf{w})}(\mathbf{D}\mathbf{U}^{(k)} + \mathbf{\Lambda}^{(k-1)})$$

$$\mathbf{\Lambda}^{(k)} = \mathbf{\Lambda}^{(k-1)} + (\mathbf{D}\mathbf{U}^{(k)} - \mathbf{V}^{(k)})$$

end while

D.1 Gecco+ Algorithm for Non-Differentiable Losses

Suppose the non-differentiable loss ℓ can be expressed as $\ell(\mathbf{X}, \mathbf{U}) = f(g(\mathbf{X}, \mathbf{U}))$ where f is convex but non-differentiable and g is affine. This expression is reasonable as it satisfies the affine composition of a convex function. For example, for the least absolute loss,

$f(\mathbf{Z}) = \sum_{j=1}^{p} \|z_j\|_1 = \|\mathrm{vec}(\mathbf{Z})\|_1$ and $g(\mathbf{X}, \mathbf{U}) = \mathbf{X} - \mathbf{U}$. We specify the affine function g as we want to augment the non-differentiable term in the loss function ℓ

We can rewrite the problem as:

$$\underset{\mathbf{U}}{\text{minimize}} \quad f(g(\mathbf{X}, \mathbf{U})) + \gamma \sum_{1 \le i \le i' \le n} w_{ii'} \|\mathbf{U}_{i\cdot} - \mathbf{U}_{i'\cdot}\|_2 + \alpha \sum_{j=1}^{p} \zeta_j \|\mathbf{U}_{\cdot j} - \tilde{x}_j \mathbf{1}_n\|_2 \ .$$

We can now recast the problem above as the equivalent constrained problem:

$$\begin{aligned} \underset{\mathbf{U}, \mathbf{V}, \mathbf{Z}, \mathbf{R}}{\text{minimize}} \quad & f(\mathbf{Z}) + \underbrace{\gamma \left(\sum_{l \in \varepsilon} w_l \|\mathbf{V}_{l\cdot}\|_2 \right)}_{P_1(\mathbf{V}; \mathbf{w})} + \underbrace{\alpha \left(\sum_{j=1}^{p} \zeta_j \|\mathbf{r}_j\|_2 \right)}_{P_2(\mathbf{R}; \zeta)} \\ \text{subject to} \quad & g(\mathbf{X}, \mathbf{U}) = \mathbf{Z} \\ & \mathbf{D}\mathbf{U} - \mathbf{V} = 0 \\ & \mathbf{U} - \widetilde{\mathbf{X}} = \mathbf{R} \ , \end{aligned}$$

where $\widetilde{\mathbf{X}}$ is an $n \times p$ matrix with the j^{th} column equal to scalar $\tilde{x}_j$.

The augmented Lagrangian in scaled form is:

$$\begin{aligned} L(\mathbf{U}, \mathbf{V}, \mathbf{Z}, \mathbf{R}, \mathbf{\Lambda}, \mathbf{N}, \mathbf{\Psi}) = & \frac{\rho}{2} \|\mathbf{D}\mathbf{U} - \mathbf{V} + \mathbf{\Lambda}\|_F^2 + \frac{\rho}{2} \|\left(\mathbf{U} - \widetilde{\mathbf{X}}\right) - \mathbf{R} + \mathbf{N}\|_F^2 \\ & + \frac{\rho}{2} \|g(\mathbf{X}, \mathbf{U}) - \mathbf{Z} + \mathbf{\Psi}\|_F^2 + f(\mathbf{Z}) + \gamma \sum_{l \in \varepsilon} w_l \|\mathbf{V}_{l\cdot}\|_2 + \alpha \sum_{j=1}^{p} \zeta_j \|\mathbf{r}_j\|_2 \ , \end{aligned}$$

where the dual variable for $\mathbf{V}$ is denoted by $\mathbf{\Lambda}$; the dual variable for $\mathbf{Z}$ is denoted by $\mathbf{\Psi}$; the dual variable for $\mathbf{R}$ is denoted by $\mathbf{N}$.

Since we assume g to be affine, i.e., $g(\mathbf{X}, \mathbf{U}) = \mathbf{A}\mathbf{X} + \mathbf{B}\mathbf{U} + \mathbf{C}$, the augmented Lagrangian in scaled form can be written as:

$$\begin{aligned} L(\mathbf{U}, \mathbf{V}, \mathbf{Z}, \mathbf{R}, \mathbf{\Lambda}, \mathbf{N}, \mathbf{\Psi}) = & \frac{\rho}{2} \|\mathbf{D}\mathbf{U} - \mathbf{V} + \mathbf{\Lambda}\|_F^2 + \frac{\rho}{2} \|\left(\mathbf{U} - \widetilde{\mathbf{X}}\right) - \mathbf{R} + \mathbf{N}\|_F^2 \\ & + \frac{\rho}{2} \|\mathbf{A}\mathbf{X} + \mathbf{B}\mathbf{U} + \mathbf{C} - \mathbf{Z} + \mathbf{\Psi}\|_F^2 + f(\mathbf{Z}) + \gamma \sum_{l \in \varepsilon} \omega_l \|\mathbf{V}_{l\cdot}\|_2 + \alpha \sum_{j=1}^{p} \zeta_j \|\mathbf{r}_j\|_2 \ . \end{aligned}$$

It can be shown that the $\mathbf{U}$ sub-problem has a closed-form solution.

Note that, hinge loss is also non-differentiable and we can write $g(\mathbf{X}, \mathbf{U}) = 1 - \mathbf{U} \circ \mathbf{X}$ where $\mathbf{1}$ is a matrix of all one and "$\circ$" is the Hadamard product. In this case, the $\mathbf{U}$ sub-problem does not have a closed-form solution and we will discuss how to solve this problem in the next section.

For distance-based losses, the loss function can always be written as: $\ell(\mathbf{X}, \mathbf{U}) = f(\mathbf{X} - \mathbf{U})$, which means $g(\mathbf{X}, \mathbf{U}) = \mathbf{X} - \mathbf{U}$. Then the augmented Lagrangian in scaled form can be simplified as:

$$L(\mathbf{U}, \mathbf{V}, \mathbf{Z}, \mathbf{R}, \mathbf{\Lambda}, \mathbf{N}, \mathbf{\Psi}) = \frac{\rho}{2}\|\mathbf{D}\mathbf{U} - \mathbf{V} + \mathbf{\Lambda}\|_F^2 + \frac{\rho}{2}\|(\mathbf{U} - \widetilde{\mathbf{X}}) - \mathbf{R} + \mathbf{N}\|_F^2 + \frac{\rho}{2}\|\mathbf{X} - \mathbf{U} - \mathbf{Z} + \mathbf{\Psi}\|_F^2 + f(\mathbf{Z}) + \gamma \sum_{l \in \varepsilon} \omega_l \|\mathbf{V}_l\|_2 + \alpha \sum_{j=1}^{p} \zeta_j \|\mathbf{r}_j\|_2 .$$

Now the $\mathbf{U}$ sub-problem has a closed-form solution:

$$\mathbf{U}^{(k)} = (\mathbf{D}^T\mathbf{D} + 2\mathbf{I})^{-1}(\mathbf{D}^T(\mathbf{V}^{(k-1)} - \mathbf{\Lambda}^{(k-1)}) + \widetilde{\mathbf{X}} + \mathbf{R}^{(k-1)} - \mathbf{N}^{(k-1)} + \mathbf{X} - \mathbf{Z}^{(k-1)} + \mathbf{\Psi}^{(k-1)}) .$$

This gives us Algorithm 13 to solve Gecco+ for non-differentiable distance-based loss.

Algorithm 13

Multi-block ADMM for non-differentiable distance-based loss

Precompute: Difference matrix $\mathbf{D}$, $\mathbf{M} = \left(\mathbf{D}^T\mathbf{D} + 2\mathbf{I}\right)^{-1}$.

while not converged **do**

$\mathbf{U}^{(k)} = \mathbf{M}(\mathbf{D}^T(\mathbf{V}^{(k-1)} - \mathbf{\Lambda}^{(k-1)}) + \widetilde{\mathbf{X}} + \mathbf{R}^{(k-1)} - \mathbf{N}^{(k-1)} + \mathbf{X} - \mathbf{Z}^{(k-1)} + \mathbf{\Psi}^{(k-1)})$

$\mathbf{Z}^{(k)} = \text{prox}_{f/\rho}(\mathbf{X} - \mathbf{U}^{(k)} + \mathbf{\Psi}^{(k-1)})$

$\mathbf{R}^{(k)} = \text{prox}_{\alpha/\rho P_2(\cdot;\zeta)}(\mathbf{U}^{(k)} - \widetilde{\mathbf{X}} + \mathbf{N}^{(k-1)})$

$\mathbf{\Psi}^{(k)} = \mathbf{\Psi}^{(k-1)} + (\mathbf{X} - \mathbf{U}^{(k)} - \mathbf{Z}^{(k)})$

$\mathbf{N}^{(k)} = \mathbf{N}^{(k-1)} + (\mathbf{U}^{(k)} - \widetilde{\mathbf{X}} - \mathbf{R}^{(k)})$

$\mathbf{V}^{(k)} = \text{prox}_{\gamma/\rho P_1(\cdot;\mathbf{w})}(\mathbf{D}\mathbf{U}^{(k)} + \mathbf{\Lambda}^{(k-1)})$

$\mathbf{\Lambda}^{(k)} = \mathbf{\Lambda}^{(k-1)} + (\mathbf{D}\mathbf{U}^{(k)} - \mathbf{V}^{(k)})$

end while

Algorithm 13 can be used to solve Gecco+ problem with various distances such as Manhattan, Minkowski and Chebychev distances by applying the corresponding proximal operator in the $\mathbf{Z}$-update. For example, for Gecco+ with Manhattan distances, the $\mathbf{Z}$-update is just applying element-wise soft-thresholding operator. For Gecco+ with Chebychev distances, the proximal operator in the $\mathbf{Z}$-update can be computed separately across the rows of its argument and reduces to applying row-wise proximal operator of the infinity-norm. For Gecco+ with Minkowski distances, we similarly apply row-wise proximal operator of the ℓ_q-norm.

Next we prove the convergence of Algorithm 13.

D.2 Proof of Convergence for Algorithm 13

Theorem 7 *If ℓ is convex*, Algorithm 13 *converges to a global minimum*.

Proof: Note we have provided a sufficient condition for the convergence of four-block ADMM in Lemma 5. Next we show that the constraint set in our problem satisfies the condition in Lemma 5 and hence the multi-block ADMM Algorithm 13 converges. Recall our problem is:

$$\begin{aligned} \underset{\mathbf{U},\mathbf{V},\mathbf{Z},\mathbf{R}}{\text{minimize}} \quad & f(\mathbf{Z}) + \gamma \sum_{l \in \varepsilon} w_l \|\mathbf{V}_{l\cdot}\|_2 + \alpha \sum_{j=1}^{p} \zeta_j \|\mathbf{r}_j\|_2 \\ \text{subject to} \quad & \mathbf{X} - \mathbf{U} = \mathbf{Z} \\ & \mathbf{D}\mathbf{U} - \mathbf{V} = 0 \\ & \mathbf{U} - \widetilde{\mathbf{X}} = \mathbf{R} \ . \end{aligned}$$

Note that Lemma 5 is stated in vector form. Hence we transform the constraints above from matrix form to vector form. Note that $\mathbf{DU} = \mathbf{V} \Leftrightarrow \mathbf{U}^T\mathbf{D}^T = \mathbf{V}^T \Leftrightarrow (\mathbf{D} \otimes \mathbf{I}_p)\text{vec}(\mathbf{U}^T) = \text{vec}(\mathbf{V}^T)$. Hence we can write the constraints as:

$$\begin{pmatrix} \mathbf{I} \\ \mathbf{A} \\ \mathbf{I} \end{pmatrix} \mathbf{u} + \begin{pmatrix} \mathbf{I} \\ \mathbf{0} \\ \mathbf{0} \end{pmatrix} \mathbf{z} + \begin{pmatrix} \mathbf{0} \\ \mathbf{0} \\ -\mathbf{I} \end{pmatrix} \mathbf{r} + \begin{pmatrix} \mathbf{0} \\ -\mathbf{I} \\ \mathbf{0} \end{pmatrix} \mathbf{v} = \mathbf{b} \ ,$$

where $\mathbf{u} = \text{vec}(\mathbf{U}^T)$, $\mathbf{A} = \mathbf{D} \otimes \mathbf{I}_p$, $\mathbf{z} = \text{vec}(\mathbf{Z}^T)$, $\mathbf{r} = \text{vec}(\mathbf{R}^T)$, $\mathbf{v} = \text{vec}(\mathbf{V}^T)$, $\mathbf{b} = \begin{pmatrix} \text{vec}(\mathbf{X}^T) \\ \mathbf{0}_{p \times |\epsilon|} \\ \widetilde{\mathbf{x}} \\ \vdots \\ \widetilde{\mathbf{x}} \end{pmatrix}$, $\widetilde{\mathbf{x}} \in \mathbb{R}^p$ is a column vector consisting of all $\tilde{x}_j$ and is repeated n times in $\mathbf{b}$.

By construction, $\mathbf{A}_2 = \begin{pmatrix} \mathbf{I} \\ \mathbf{0} \\ \mathbf{0} \end{pmatrix}$, $\mathbf{A}_3 = \begin{pmatrix} \mathbf{0} \\ \mathbf{0} \\ -\mathbf{I} \end{pmatrix}$, and $\mathbf{A}_4 = \begin{pmatrix} \mathbf{0} \\ -\mathbf{I} \\ \mathbf{0} \end{pmatrix}$. It is easy to verify that: $\mathbf{A}_2^T\mathbf{A}_3 = \mathbf{0}$, $\mathbf{A}_2^T\mathbf{A}_4 = \mathbf{0}$, $\mathbf{A}_3^T\mathbf{A}_4 = \mathbf{0}$. Hence our setup satisfies the sufficient condition in Lemma 5 and the multi-block ADMM Algorithm 13 converges. ■

D.3 Special Case: Gecco+ with Hinge Losses

As mentioned, we cannot directly apply Algorithm 13 to solve Gecco+ with hinge losses as the function $g(\mathbf{X}, \mathbf{U})$ in this case is not the same as the one in distance-based losses. Recall Gecco+ with hinge losses is:

$$\underset{\mathbf{U}}{\text{minimize}} \quad \sum_{i=1}^{n} \sum_{j=1}^{p} \max(0, 1 - u_{ij}x_{ij}) + \gamma \sum_{1 \le i < i' \le n} w_{ii'} \|\mathbf{U}_{i\cdot} - \mathbf{U}_{i'\cdot}\|_2 + \alpha \sum_{j=2}^{p} \zeta_j \|\mathbf{U}_{\cdot j} - \tilde{x}_j \cdot \mathbf{1}_n\|_2 \ .$$

Like before, we can rewrite the problem as:

$$\underset{\mathbf{U}}{\text{minimize}} \quad f(g(\mathbf{X},\mathbf{U})) + \gamma \sum_{1 \le i < i' \le n} w_{ii'} \|\mathbf{U}_{i\cdot} - \mathbf{U}_{i'\cdot}\|_2 + \alpha \sum_{j=1}^{p} \zeta_j \|\mathbf{U}_{\cdot j} - \tilde{x}_j \cdot \mathbf{1}_n\|_2 .$$

We can now recast the problem above as the equivalent constrained problem:

$$\begin{aligned} \underset{\mathbf{U},\mathbf{V},\mathbf{Z},\mathbf{R}}{\text{minimize}} \quad & f(\mathbf{Z}) + \gamma \underbrace{\left(\sum_{l \in \varepsilon} w_l \|\mathbf{V}_{l\cdot}\|_2 \right)}_{P_1(\mathbf{V};\mathbf{w})} + \alpha \underbrace{\left(\sum_{j=1}^{p} \zeta_j \|\mathbf{r}_j\|_2 \right)}_{P_2(\mathbf{R};\zeta)} \\ \text{subject to} \quad & \mathbf{1} - \mathbf{U} \circ \mathbf{X} = \mathbf{Z} \\ & \mathbf{D}\mathbf{U} - \mathbf{V} = 0 \\ & \mathbf{U} - \widetilde{\mathbf{X}} = \mathbf{R} . \end{aligned}$$

Here, $f(\mathbf{Z}) = \max(0, \mathbf{Z})$. With a slight abuse of notation, we refer f to applying element-wise maximum to all entries in the matrix. We set $g(\mathbf{X}, \mathbf{U}) = 1 - \mathbf{U} \circ \mathbf{X}$ where $\mathbf{1}$ is a matrix of all one and "$\circ$" is the Hadamard product. $\widetilde{\mathbf{X}}$ is an $n \times p$ matrix with the j^{th} column equal to scalar $\tilde{x}_j$.

The augmented Lagrangian in scaled form is:

$$\begin{aligned} L(\mathbf{U},\mathbf{V},\mathbf{Z},\mathbf{R},\mathbf{\Lambda},\mathbf{N},\mathbf{\Psi}) = {} & \frac{\rho}{2}\|\mathbf{D}\mathbf{U} - \mathbf{V} + \mathbf{\Lambda}\|_F^2 + \frac{\rho}{2}\|(\mathbf{U} - \widetilde{\mathbf{X}}) - \mathbf{R} + \mathbf{N}\|_F^2 \\ & + \frac{\rho}{2}\|\mathbf{1} - \mathbf{U} \circ \mathbf{X} - \mathbf{Z} + \mathbf{\Psi}\|_F^2 + f(\mathbf{Z}) + \gamma \sum_{l \in \varepsilon} w_l \|\mathbf{V}_{l\cdot}\|_2 + \alpha \sum_{j=1}^{p} \zeta_j \|\mathbf{r}_j\|_2 . \end{aligned}$$

The $\mathbf{U}$ sub-problem now becomes:

$$\begin{aligned} \mathbf{U}^{(k+1)} = \underset{\mathbf{U}}{\text{argmin}} \quad & \|\mathbf{D}\mathbf{U} - \mathbf{V}^{(k)} + \mathbf{\Lambda}^{(k)}\|_F^2 + \|\mathbf{U} - \widetilde{\mathbf{X}} - \mathbf{R}^{(k)} + \mathbf{N}^{(k)}\|_F^2 \\ & + \|\mathbf{1} - \mathbf{U} \circ \mathbf{X} - \mathbf{Z}^{(k)} + \mathbf{\Psi}^{(k)}\|_F^2 . \end{aligned}$$

The first-order optimality condition is:

$$\mathbf{D}^T(\mathbf{D}\mathbf{U} - \mathbf{V}^{(k)} + \mathbf{\Lambda}^{(k)}) + \mathbf{U} - \widetilde{\mathbf{X}} - \mathbf{R}^{(k)} + \mathbf{N}^{(k)} + \mathbf{X} \circ (\mathbf{U} \circ \mathbf{X} + \mathbf{Z}^{(k)} - 1 - \mathbf{\Psi}^{(k)}) = \mathbf{0} ,$$

which can be written as:

$$\begin{aligned} & (\mathbf{X} \circ \mathbf{X}) \circ \mathbf{U} + \mathbf{X} \circ (\mathbf{Z}^{(k)} - 1 - \mathbf{\Psi}^{(k)}) + \mathbf{D}^T\mathbf{D}\mathbf{U} - \mathbf{D}^T(\mathbf{V}^{(k)} - \mathbf{\Lambda}^{(k)}) + \mathbf{U} - \widetilde{\mathbf{X}} - \mathbf{R}^{(k)} + \mathbf{N}^{(k)} = \mathbf{0} \\ & (\mathbf{X} \circ \mathbf{X}) \circ \mathbf{U} + (\mathbf{D}^T\mathbf{D}\mathbf{U} + \mathbf{I})\mathbf{U} = \mathbf{D}^T(\mathbf{V}^{(k)} - \mathbf{\Lambda}^{(k)}) + \widetilde{\mathbf{X}} + \mathbf{R}^{(k)} - \mathbf{N}^{(k)} + \mathbf{X} \circ (1 + \mathbf{\Psi}^{(k)} - \mathbf{Z}^{(k)}) . \end{aligned}$$

To solve $\mathbf{U}$ from the above equation, one way is to first find the SVD of the leading coefficient:

$$\mathbf{X} \circ \mathbf{X} = \sum_k \sigma_k \tilde{\mathbf{u}}_k \tilde{\mathbf{v}}_k = \sum_k \widetilde{\mathbf{w}}_k \tilde{\mathbf{v}}_k \ .$$

From this decomposition we create two sets of diagonal matrices:

$$\widetilde{\mathbf{W}}_k = \mathrm{Diag}(\widetilde{\mathbf{w}}_k)$$
$$\widetilde{\mathbf{V}}_k = \mathrm{Diag}(\tilde{\mathbf{v}}_k) \ .$$

The Hadamard product can now be replaced by a sum

$$\sum_k \widetilde{\mathbf{W}}_k \mathbf{U} \widetilde{\mathbf{V}}_k + (\mathbf{D}^T\mathbf{D} + \mathbf{I})\mathbf{U} = \mathbf{C} \ ,$$

where $\mathbf{C} = \mathbf{D}^T(\mathbf{V}^{(k)} - \mathbf{\Lambda}^{(k)}) + \widetilde{\mathbf{X}} + \mathbf{R}^{(k)} - \mathbf{N}^{(k)} + \mathbf{X} \circ (1 + \mathbf{\Psi}^{(k)} - \mathbf{Z}^{(k)})$.

Now we can solve this equation via vectorization:

$$\mathrm{vec}(\mathbf{C}) = (\mathbf{I} \otimes (\mathbf{D}^T\mathbf{D} + \mathbf{I}) + \sum_k \widetilde{\mathbf{V}}_k \otimes \widetilde{\mathbf{W}}_k)\mathrm{vec}(\mathbf{U})$$
$$\mathrm{vec}(\mathbf{U}) = (\mathbf{I} \otimes (\mathbf{D}^T\mathbf{D} + \mathbf{I}) + \sum_k \widetilde{\mathbf{V}}_k \otimes \widetilde{\mathbf{W}}_k)^+ \mathrm{vec}(\mathbf{C})$$
$$\mathbf{U} = \mathrm{Mat}\left((\mathbf{I} \otimes (\mathbf{D}^T\mathbf{D} + \mathbf{I}) + \sum_k \widetilde{\mathbf{V}}_k \otimes \widetilde{\mathbf{W}}_k)^+ \mathrm{vec}(\mathbf{C})\right),$$

where $\mathbf{B}^+$ denotes the pseudo-inverse of $\mathbf{B}$, and Mat() is the inverse of the vec() operation.

Here we have to compute the pseudo-inverse of a matrix which is computationally expensive in practice. To avoid this, we adopt generalized ADMM approach proposed by Deng and Yin (2016) where the $\mathbf{U}$ sub-problem is augmented by a positive semi-definite quadratic operator. In our case, our modified $\mathbf{U}$ sub-problem becomes:

$$\underset{\mathbf{U}}{\mathrm{argmin}} \ \|\mathbf{D}\mathbf{U} - \mathbf{V}^{(k)} + \mathbf{\Lambda}^{(k)}\|_F^2 + \|(\mathbf{U} - \widetilde{\mathbf{X}}) - \mathbf{R}^{(k)} + \mathbf{N}^{(k)}\|_F^2 + \|\mathbf{1} - \mathbf{U} \circ \mathbf{X} - \mathbf{Z}^{(k)} + \mathbf{\Psi}^{(k)}\|_F^2$$
$$+ \|(\mathbf{1} - \mathbf{X} \circ \mathbf{X}) \circ \left(\mathbf{U} - \mathbf{U}^{(k)}\right)\|_F^2 \ .$$

The first-order optimality condition now becomes:

$$\mathbf{D}^T(\mathbf{D}\mathbf{U} - \mathbf{V}^{(k)} + \mathbf{\Lambda}^{(k)})\mathbf{U} - \widetilde{\mathbf{X}} - \mathbf{R}^{(k)} + \mathbf{N}^{(k)} + \mathbf{X} \circ (\mathbf{U} \circ \mathbf{X} + \mathbf{Z}^{(k)} - 1 - \mathbf{\Psi}^{(k)})$$
$$+ (1 - \mathbf{X} \circ \mathbf{X}) \circ (\mathbf{U} - \mathbf{U}^{(k)}) = 0 \ .$$

We have:

$$\mathbf{H}\mathbf{U} = \mathbf{D}^T(\mathbf{V}^{(k)} - \mathbf{\Lambda}^{(k)}) + \widetilde{\mathbf{X}} + \mathbf{R}^{(k)} - \mathbf{N}^{(k)} - \mathbf{X} \circ (\mathbf{Z}^{(k)} - 1 - \mathbf{\Psi}^{(k)}) + (1 - \mathbf{X} \circ \mathbf{X}) \circ \mathbf{U}^{(k)} \ ,$$

where $\mathbf{H} = (\mathbf{D}^T\mathbf{D} + \mathbf{I} + \mathbf{1})$. Hence we have analytical update:

$$\mathbf{U}^{(k+1)} = \mathbf{H}^{-1}(\mathbf{D}^T(\mathbf{V}^{(k)} - \mathbf{\Lambda}^{(k)}) + \widetilde{\mathbf{X}} + \mathbf{R}^{(k)} - \mathbf{N}^{(k)} - \mathbf{X} \circ (\mathbf{Z}^{(k)} - 1 - \mathbf{\Psi}^{(k)}) + (1 - \mathbf{X} \circ \mathbf{X}) \circ \mathbf{U}^{(k)}) \ .$$

It is easy to see that the $\mathbf{V}$, $\mathbf{Z}$ and $\mathbf{R}$ updates all have closed-form solutions.

Appendix E.: Multinomial Gecco+

In this section, we briefly demonstrate how Gecco+ with multinomial losses is formulated, which is slightly different from the original Gecco+ problems. Suppose we observe categorical data as follows ($K = 3$):

$$\mathbf{X}_{n \times p} = \begin{pmatrix} 1 & 1 & 1 \\ 2 & 2 & 2 \\ 3 & 3 & 3 \end{pmatrix} .$$

We can get the indicator matrix $\mathbf{X}^{(k)}$ for each class k as:

$$\mathbf{X}^{(1)} = \begin{pmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \quad \mathbf{X}^{(2)} = \begin{pmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \\ 0 & 0 & 0 \end{pmatrix} \quad \mathbf{X}^{(3)} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{pmatrix} .$$

Then we concatenate $\mathbf{X}^{(1)}$, $\mathbf{X}^{(2)}$, $\mathbf{X}^{(3)}$ and get $\widehat{\mathbf{X}}_{n \times (p * K)} = \left(\mathbf{X}^{(1)}\ \mathbf{X}^{(2)}\ \mathbf{X}^{(3)}\right)$. This is equivalent to the dummy coding of the categorical matrix $\widetilde{\mathbf{X}}$ after some row/column shuffle:

$$\widetilde{\mathbf{X}} = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \end{pmatrix}_{n \times (p * K)} .$$

It is obvious that measuring the difference of two observations by comparing rows of $\widehat{\mathbf{X}}$ is better than simply comparing the Euclidean distances of rows of original data matrix $\mathbf{X}$. Also parameterizing in $\widehat{\mathbf{X}}$ is beneficial for computing the multinomial log-likelihood or deviance. Hence we concatenate all columns of $\mathbf{X}^{(k)}$ as input data. Similarly, we concatenate all columns of the corresponding $\mathbf{U}^{(k)}$ and then fuse $\widehat{\mathbf{U}}$ in a row-wise way.

E.1 Gecco with Multinomial Log-likelihood

Gecco with multinomial log-likelihood can be formulated as:

$$\underset{\mathbf{U}}{\text{minimize}} \quad \sum_i \sum_j \left\{ \sum_{k=1}^{K} -x_{ijk} u_{ijk} + \log\left(\sum_{k=1}^{K} e^{u_{ijk}}\right) \right\} + \gamma \sum_{i < i'} w_{ii'} \left\| \begin{bmatrix} \mathbf{U}_{i\cdot}^{(1)} \\ \vdots \\ \mathbf{U}_{i\cdot}^{(1)} \end{bmatrix} - \begin{bmatrix} \mathbf{U}_{i'\cdot}^{(1)} \\ \vdots \\ \mathbf{U}_{i'\cdot}^{(K)} \end{bmatrix} \right\|_2 ,$$

where x_{ijk} refers to the elements of indicator matrix $\mathbf{X}_{ij}^{(k)}$ discussed previously and $\mathbf{U}_{i\cdot}^{(k)} = \begin{bmatrix} u_{i1k} \\ u_{i2k} \\ \vdots \\ u_{ipk} \end{bmatrix}$.

E.2 Gecco+ with Multinomial Log-likelihood

Gecco+ with multinomial log-likelihood can be formulated as:

$$\underset{\mathbf{U}}{\text{minimize}} \quad \sum_i \sum_j \left\{ \sum_{k=1}^{K} -x_{ijk} u_{ijk} + \log\left(\sum_{k=1}^{K} e^{u_{ijk}}\right) \right\} + \gamma \sum_{i<i'} w_{ii'} \left\| \begin{bmatrix} \mathbf{U}_{i\cdot}^{(1)} \\ \vdots \\ \mathbf{U}_{i\cdot}^{(K)} \end{bmatrix} - \begin{bmatrix} \mathbf{U}_{i'\cdot}^{(1)} \\ \vdots \\ \mathbf{U}_{i'\cdot}^{(K)} \end{bmatrix} \right\|_2 + \alpha \sum_{j=1}^{p} \sum_{k=1}^{K} \| \mathbf{U}_{\cdot j}^{(k)} - \tilde{x}_j^{(k)} \mathbf{1}_n \|_2 ,$$

where $\mathbf{U}_{i\cdot}^{(k)} = \begin{bmatrix} u_{i1k} \\ u_{i2k} \\ \vdots \\ u_{ipk} \end{bmatrix}$, $\mathbf{U}_{\cdot j}^{(k)} = \begin{bmatrix} u_{1jk} \\ u_{2jk} \\ \vdots \\ u_{njk} \end{bmatrix}$ and $\tilde{x}_j^{(k)}$ is the loss-specific center for j variable in k^{th} class.

Appendix F.: Loss-specific Center Calculation

In this section, we show how to calculate the loss-specific center in Table 1.

F.1 Continuous Data

For continuous data, we consider Gecco with Euclidean distances.

F.1.1 Euclidean Distance

$$\underset{\mathbf{U}}{\text{minimize}} \quad \frac{1}{2} \sum_{i=1}^{n} \| \mathbf{X}_{i\cdot} - \mathbf{U}_{i\cdot} \|_2^2 + \gamma \sum_{i<i'} w_{ii'} \| \mathbf{U}_{i\cdot} - \mathbf{U}_{i'\cdot} \|_2$$

When total fusion occurs, $\mathbf{U}_{i\cdot} = \mathbf{U}_{i'\cdot}, \forall i \neq i'$. Let $\mathbf{U}_{i\cdot} = \mathbf{U}_{i'\cdot} = \mathbf{u}$. The problem above becomes:

$$\underset{\mathbf{u}}{\text{minimize}} \quad \frac{1}{2} \sum_{i=1}^{n} \| \mathbf{X}_{i\cdot} - \mathbf{u} \|_2^2 .$$

Taking derivative, we get:

$$\sum_{i=1}^{n}(\mathbf{X}_{i\cdot}-\mathbf{u})=0\Rightarrow\mathbf{u}=\bar{\mathbf{x}}\ .$$

F.2 Count Data

For count-valued data, we consider Gecco with Poisson log-likelihood/deviance, negative binomial log-likelihood/deviance and Manhattan distances.

F.2.1 Poisson log-likelihood

$$\underset{\mathbf{U}}{\text{minimize}}\quad\sum_{i}^{n}\sum_{j}^{p}-x_{ij}u_{ij}+\exp(u_{ij})+\gamma\sum_{i<i'}w_{ii'}\|\mathbf{U}_{i\cdot}-\mathbf{U}_{i'\cdot}\|_2$$

When total fusion occurs, $\mathbf{u}_{ij}=\mathbf{u}_{i'j}$, $\forall i\neq i'$. Let $\mathbf{u}$ be the fusion vector and $\mathbf{u}=(u_1,\cdots,u_p)$. The problem above becomes:

$$\underset{\mathbf{U}}{\text{minimize}}\quad\sum_{i}^{n}\sum_{j}^{p}-x_{ij}u_j+\exp(u_j)\ .$$

Taking derivative, we get:

$$\sum_{i=1}^{n}-x_{ij}+n\exp(u_j)=0\Rightarrow\exp(u_j)=\bar{x}_j\Rightarrow u_j=\log(\bar{x}_j)\Rightarrow\mathbf{u}=\log(\bar{\mathbf{x}})\ .$$

F.2.2 Poisson Deviance

$$\underset{\mathbf{U}}{\text{minimize}}\quad\sum_{i}^{n}\sum_{j}^{p}-x_{ij}\log u_{ij}+u_{ij}+\gamma\sum_{i<i'}w_{ii'}\|\mathbf{U}_{i\cdot}-\mathbf{U}_{i'\cdot}\|_2$$

Let $\mathbf{u}$ be the fusion vector and $\mathbf{u}=(u_1,\cdots,u_p)$. The problem above becomes:

$$\underset{\mathbf{U}}{\text{minimize}}\quad\sum_{i}^{n}\sum_{j}^{p}-x_{ij}\log u_j+u_j\ .$$

Taking derivative, we get:

$$\sum_{i=1}^{n}\frac{-x_{ij}}{u_j}+n=0\Rightarrow u_j=\bar{x}_j\Rightarrow\mathbf{u}=\tilde{\mathbf{x}}$$

F.2.3 Negative Binomial log-likelihood

$$\underset{\mathbf{U}}{\text{minimize}} \quad \sum_{i}^{n}\sum_{j}^{p} -x_{ij}u_{ij} + (x_{ij} + \frac{1}{\alpha})\log(\frac{1}{\alpha} + e^{u_{ij}}) + \gamma \sum_{i<i'} \|\mathbf{U}_i - \mathbf{U}_{i'}\|_2$$

When total fusion occurs, $\mathbf{u}_{ij} = \mathbf{u}_{i'j}, \forall i \neq i'$. Let $\mathbf{u}$ be the fusion vector and $\mathbf{u} = (u_1, \cdots, u_p)$. The problem above becomes:

$$\underset{\mathbf{U}}{\text{minimize}} \quad \sum_{i}^{n}\sum_{j}^{p} -x_{ij}u_j + (x_{ij} + \frac{1}{\alpha})\log(\frac{1}{\alpha} + e^{u_j}) \ .$$

Taking derivative, we get:

$$\sum_{i}^{n} -x_{ij} + (x_{ij} + \frac{1}{\alpha})\frac{e^{u_j}}{\frac{1}{\alpha} + e^{u_j}} = 0$$

$$\sum_{i}^{n} x_{ij} = \sum_{i}^{n} (x_{ij} + \frac{1}{\alpha})\frac{e^{u_j}}{\frac{1}{\alpha} + e^{u_j}}$$

$$\sum_{i}^{n} x_{ij} \cdot \frac{1}{\alpha} + \sum_{i}^{n} x_{ij} \cdot e^{u_j} = \sum_{i}^{n} x_{ij} \cdot e^{u_j} + \frac{n}{\alpha} e^{u_j}$$

$$e^{u_j} = \sum_{i}^{n} x_{ij}/n$$

$$\exp(u_j) = \bar{x}_j \Rightarrow u_j = \log(\bar{x}_j) \Rightarrow \mathbf{u} = \log(\bar{\mathbf{x}}) \ .$$

F.2.4 Negative Binomial Deviance

$$\underset{\mathbf{U}}{\text{minimize}} \quad \sum_{i}^{n}\sum_{j}^{p} x_{ij}\log(\frac{x_{ij}}{u_{ij}}) - (x_{ij} + \frac{1}{\alpha})\log(\frac{1 + \alpha x_{ij}}{1 + \alpha u_{ij}}) + \gamma \sum_{i<i'} w_{ii'} \|\mathbf{U}_i - \mathbf{U}_{i'}\|_2$$

The formulation above is equivalent to:

$$\underset{\mathbf{U}}{\text{minimize}} \quad \sum_{i}^{n}\sum_{j}^{p} -x_{ij}\log u_{ij} + (x_{ij} + \frac{1}{\alpha})\log(1 + \alpha x_{ij}) + \gamma \sum_{i<i'} w_{ii'} \|\mathbf{U}_i - \mathbf{U}_{i'}\|_2 \ .$$

Let $\mathbf{u}$ be the fusion vector and $\mathbf{u} = (u_1, \cdots, u_p)$. The problem above becomes:

$$\underset{\mathbf{U}}{\text{minimize}} \quad \sum_{i}^{n}\sum_{j}^{p} -x_{ij}\log u_j + (x_{ij} + \frac{1}{\alpha})\log(1 + \alpha u_j) \ .$$

Taking derivative, we get:

$$\sum_{i=1}^{n} \frac{-x_{ij}}{u_j} + \frac{x_{ij} + \frac{1}{\alpha}}{1 + \alpha u_j} \cdot \alpha = 0$$

$$\sum_{i}^{n} x_{ij} + \alpha u_j \sum_{i=1}^{n} x_{ij} \Big| nu_j \Big| = \alpha u_j \sum_{i=1}^{n} x_{ij}$$

$u_j = \bar{x}_j \Rightarrow \mathbf{u} = \bar{\mathbf{x}}$.

F.2.5 Manhattan Distance

$$\underset{\mathbf{U}}{\text{minimize}} \quad \sum_{i=1}^{n} \|\mathbf{x}_i - \mathbf{u}_i\|_1 \Big| = \gamma \sum_{i<i'} w_{ii'} \|\mathbf{U}_{i\,|} - \mathbf{U}_{i'\,|}\|_2$$

When total fusion occurs, $\mathbf{U}_{i\,.} = \mathbf{U}_{i'\,.}, \forall i \neq i'$. Let $\mathbf{U}_{i\,.} = \mathbf{U}_{i'\,.} = \mathbf{u}$. We have:

$$\underset{\mathbf{U}}{\text{minimize}} \quad \sum_{i=1}^{n} \|\mathbf{x}_i - \mathbf{u}\|_1 \ .$$

For each j, we have:

$$\underset{u_j}{\text{minimize}} \quad \sum_{i=1}^{n} \|x_{ij} - u_j\|_1 \ .$$

We know that the optimal u_j is just the median of x_{ij} for each j.

F.3 Binary Data

For binary data, we consider Gecco with Bernoulli log-likelihood, binomial deviance and hinge loss.

F.3.1 Bernoulli log-likelihood

$$\underset{\mathbf{U}}{\text{minimize}} \quad \sum_{i}^{n} \sum_{j}^{p} -x_{ij} u_{ij} + \log(1 + e^{u_{ij}}) + \gamma \sum_{i<i'} w_{ii'} \|\mathbf{U}_{i\,|} - \mathbf{U}_{i'\,|}\|_2$$

When total fusion occurs, $\mathbf{u}_{ij} = \mathbf{u}_{i'j}, \forall i \neq i'$. Let $\mathbf{u}$ be the fusion vector and $\mathbf{u} = (u_1, \cdots, u_p)$. The problem above becomes:

$$\underset{\mathbf{U}}{\text{minimize}} \quad \sum_{i}^{n} \sum_{j}^{p} -x_{ij} u_j + \log(1 + e^{u_i}) \ .$$

Taking derivative, we get:

$$\sum_{i=1}^{n} -x_{ij} + n\frac{\exp(u_j)}{1+\exp(u_j)} = 0 \Rightarrow \frac{\exp(u_j)}{1+\exp(u_j)} = \bar{x}_j$$
$$\Rightarrow \text{logit}^{-1}(u_j) = \bar{x}_j \Rightarrow u_j = \text{logit}(\bar{x}_j) \Rightarrow \mathbf{u} = \text{logit}(\mathbf{x}) \ .$$

F.3.2 Binomial Deviance

$$\underset{\mathbf{U}}{\text{minimize}} \quad \sum_{i}^{n}\sum_{j}^{p} -x_{ij}\log u_{ij} - (1-x_{ij})\log(1-u_{ij}) + \gamma \sum_{i<i'} w_{ii'} \|\mathbf{U}_{i\cdot} - \mathbf{U}_{i'\cdot}\|_2$$

Let $\mathbf{u}$ be the fusion vector and $\mathbf{u} = (u_1, \cdots, u_p)$. The problem above becomes:

$$\underset{\mathbf{U}}{\text{minimize}} \quad \sum_{i}^{n}\sum_{j}^{p} -x_{ij}\log u_j - (1-x_{ij})\log(1-u_j) \ .$$

Taking derivative, we get:

$$\sum_{i=1}^{n} \frac{-x_{ij}}{u_j} + \frac{1-x_{ij}}{1-u_j} = 0 \Rightarrow \sum_{i}^{n} -x_{ij}(1-u_j) + (1-x_{ij})u_j = 0$$
$$\Rightarrow \sum_{i}^{n} -x_{ij} + u_j = 0 \Rightarrow u_j = \bar{x}_j \Rightarrow \mathbf{u} = \bar{\mathbf{x}} \ .$$

F.3.3 Hinge Loss

$$\underset{\mathbf{U}}{\text{minimize}} \sum_{i=1}^{n}\sum_{j=1}^{p} \max(0, 1 - u_{ij}x_{ij}) + \gamma \sum_{i<i'} \omega_{ii'} \|\mathbf{U}_{i\cdot} - \mathbf{U}_{i'\cdot}\|_2$$

Let $\mathbf{u}$ be the fusion vector and $\mathbf{u} = (u_1, \cdots, u_p)$. The problem above becomes:

$$\underset{\mathbf{U}}{\text{minimize}} \sum_{i=1}^{n}\sum_{j=1}^{p} \max(0, 1 - u_j x_{ij}) \ .$$

For each feature j, the problem becomes:

$$\underset{u_j}{\text{minimize}} \sum_{i=1}^{n} \max(0, 1 - u_j x_{ij}) \ .$$

Note in hinge loss, $x_{ij} \in \{-1, 1\}$. Suppose we have n_1 observations for class "1" and n_2 observations for class "−1". The problem now becomes:

$$\underset{u_j}{\text{minimize}}\ n_1\max(0, 1-u_j) + n_2\max(0, 1+u_j) .$$

Define $h(t) = n_1\max(0, 1-t) + n_2\max(0, 1+t)$. We have:

$$h(t) = \begin{cases} n_1(1-t) & \text{if } t \le -1 \\ n_1(1-t) + n_2(1+t) & \text{if } -1 < t < 1 \\ n_2(1+t) & \text{if } t \ge 1 . \end{cases}$$

Clearly, if $n_2 > n_1$ (more "−1"), $h(t)$ is minimized by $t = -1$; if $n_1 > n_2$ (more "1"), $h(t)$ is minimized by $t = 1$; if $n_1 = n_2$, $h(t)$ is minimized by any t between $[-1, 1]$. Therefore, u_j should be the mode of all observations for feature j:

$$u_j = \text{mode}_i(x_{ij}) .$$

F.4 Categorical Data

For categorical data, we consider Gecco with multinomial log-likelihood and deviance.

F.4.1 Multinomial log-likelihood

$$\underset{\mathbf{U}}{\text{minimize}} \sum_i^n \sum_j^p \{ \sum_{k=1}^K -x_{ijk}u_{ijk} + \log(\sum_{k=1}^K e^{u_{ijk}}) \} + \gamma \sum_{i<i'} \omega_{ii'} \left\| \begin{bmatrix} \mathbf{U}_{i\cdot}^{(1)} \\ \vdots \\ \mathbf{U}_{i\cdot}^{(K)} \end{bmatrix} - \begin{bmatrix} \mathbf{U}_{i'\cdot}^{(1)} \\ \vdots \\ \mathbf{U}_{i'\cdot}^{(K)} \end{bmatrix} \right\|_2$$

When total fusion occurs, $u_{ijk} = u_{i'jk}, \forall i \neq i'$. Let $\mathbf{u}$ be the fusion vector and $\mathbf{u}_k = (u_{1k}, \cdots, u_{pk})$. The problem above becomes:

$$\underset{\mathbf{U}}{\text{minimize}} \sum_i^n \sum_j^p \{ \sum_k^K -x_{ijk}u_{jk} + \log(\sum_k^K e^{u_{jk}}) \} .$$

Taking derivative with respect to u_{jk}, we get:

$$\sum_{i=1}^n -x_{ijk} + n\frac{\exp(u_{jk})}{\sum_k^K \exp(u_{jk})} = 0$$

$$u_{jk} = \log\frac{\bar{x}_{\cdot jk}}{\sum_k \bar{x}_{|jk}} = \text{mlogit}(\bar{x}_{\cdot jk}) .$$

F.4.2 Multinomial Deviance

$$\underset{\mathbf{U}}{\text{minimize}} \sum_{i}^{n}\sum_{j}^{p}\{\sum_{k}^{K}-x_{ijk}\log(u_{ijk})\}+\gamma\left\|\begin{bmatrix}\mathbf{U}_{i.}^{(1)}\\ \vdots \\ \mathbf{U}_{i.}^{(K)}\end{bmatrix}-\begin{bmatrix}\mathbf{U}_{i'.}^{(1)}\\ \vdots \\ \mathbf{U}_{i'.}^{(K)}\end{bmatrix}\right\|_2$$

$$\text{subject to } \sum_{k=1}^{K} u_{ijk}=1$$

When total fusion occurs, $\mathbf{u}_{ijk}=\mathbf{u}_{i'jk}, \forall i\neq i'$. Let $\mathbf{u}$ be the fusion vector and $\mathbf{u}_k=(u_{1k},\cdots,u_{pk})$. The problem above becomes:

$$\underset{\mathbf{U}}{\text{minimize}} \sum_{i}^{n}\sum_{j}^{p}\sum_{k}^{K}-x_{ijk}\log u_{jk}$$

$$\text{subject to } \sum_{k=1}^{K} u_{jk}=1 \ .$$

We can write the constraint in Lagrangian form:

$$\underset{\mathbf{U}}{\text{minimize}} \sum_{i}^{n}\sum_{j}^{p}\sum_{k}^{K}-x_{ijk}\log u_{jk}+\lambda(\sum_{k=1}^{K} u_{jk}-1) \ .$$

Taking derivative with respect to u_{jk}, we get:

$$\sum_{i=1}^{n}\frac{-x_{ijk}}{u_{jk}}+\lambda=0$$

$$\sum_{i=1}^{n} x_{ijk}=\lambda u_{jk} \ .$$

We have:

$$n=\sum_{i=1}^{n}\sum_{k=1}^{K}x_{ijk}=\sum_{k=1}^{K}\sum_{i=1}^{n}x_{ijk}=\sum_{k=1}^{K}\lambda u_{jk}=\lambda \ .$$

Therefore,

$$u_{jk}=\sum_{i=1}^{n}x_{ijk}/n \ .$$

Appendix G.: Visualization of Gecco+ for Authors Data

Figure 5 illustrates selected features and cluster assignment for authors data set with one combination of α and γ. We select meaningful features and achieve satisfactory clustering results. We have already discussed the results and interpretation in detail in Section 5.1.

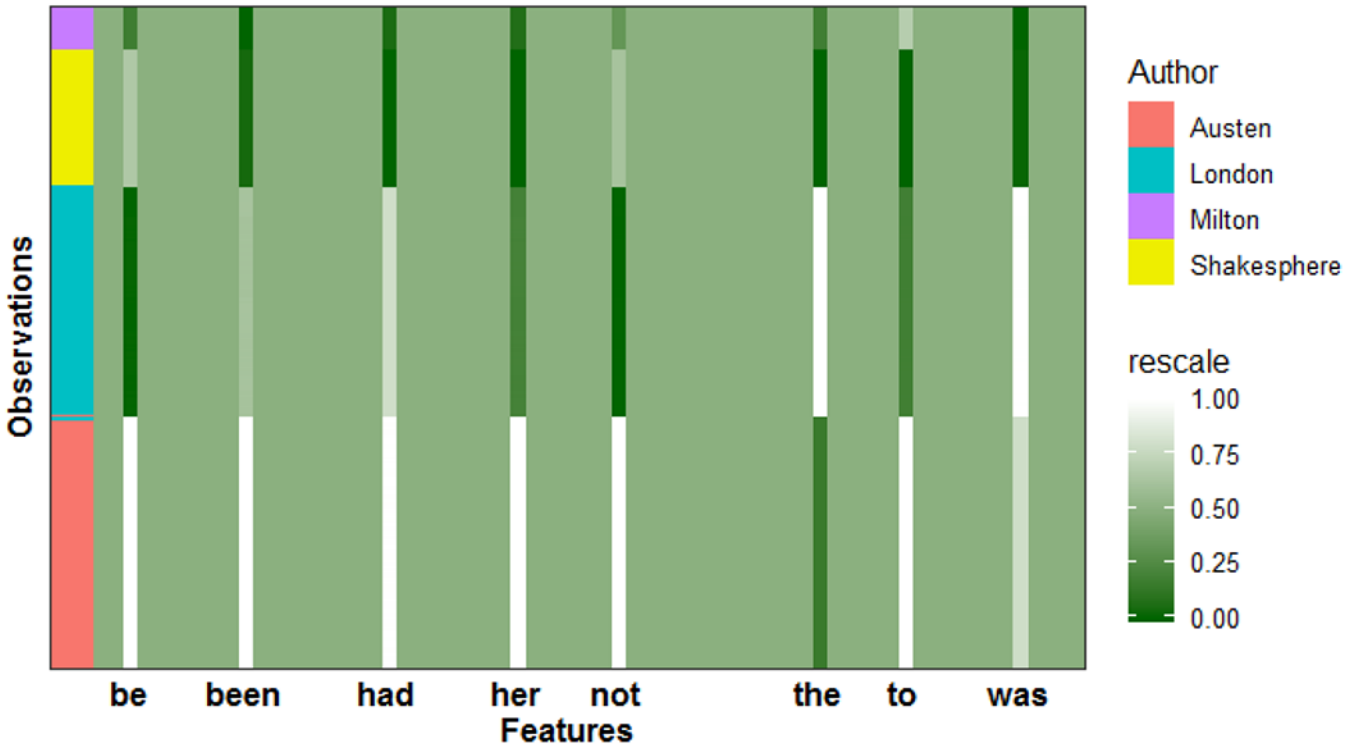

Figure 5:
Cluster heatmap of Gecco+ solution $\widehat{\mathbf{U}}(\gamma, \alpha)$ for authors data set with $\alpha = 15$ and $\gamma = 10^5$. The left bar refers to the true author label. We highlight selected features at the bottom. Gecco+ selects informative features that separate groups.

Appendix H.: Multi-omics Data

In this section, we show the distribution of data from different platforms in Section 5.3. We see that both gene expression data and protein data appear Gaussian; methylation data is between [0, 1]; miRNA data is highly-skewed.

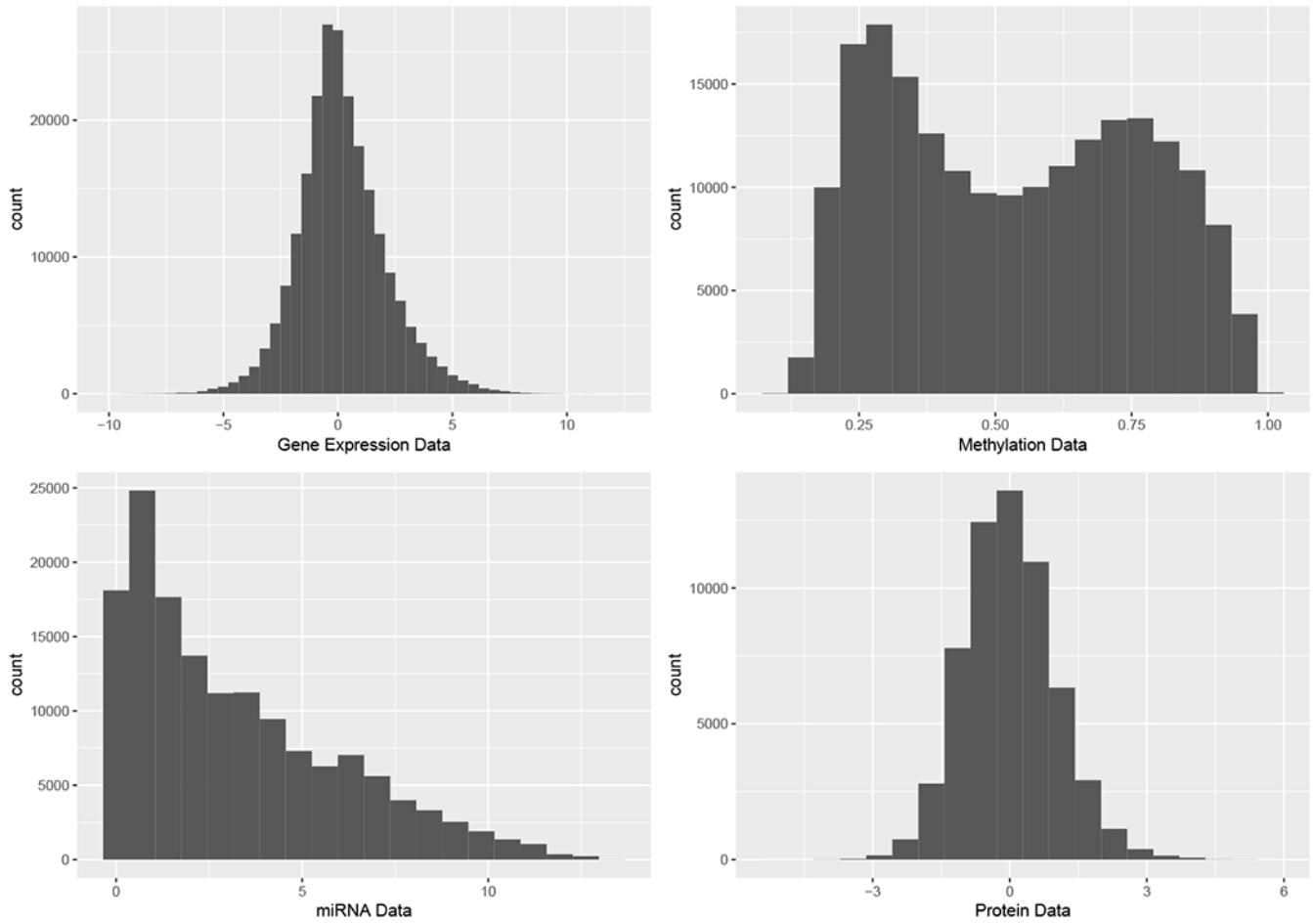

Figure 6:
Histograms of data from different platforms for multi-omics TCGA data set. Both gene expression data and protein data appear Gaussian; methylation data is proportion-valued; miRNA data is highly-skewed.

Appendix I.: Null Deviance

In this appendix, we list the null deviance D for some common losses used in iGecco. Recall in the iGecco formulation, π_k, which are set inversely proportional to the null deviance evaluated at the loss-specific center, are scaling factors to ensure that the losses are measured at the same scale in the objective function.

By definition, the null deviance D evaluated at the loss-specific center, refers to $\ell(\mathbf{X}, \widetilde{\mathbf{X}})$ where each j^{th} column of $\widetilde{\mathbf{X}}$ is the loss-specific center $\tilde{x}_j^{(k)}$ for that column/feature. Table 12 shows the null deviance D for some common losses.

Table 12:

Null deviance for common losses used in iGecco

Loss Function	Null Deviance D
Euclidean distance	$\|\mathbf{X}-\overline{\mathbf{X}}\|_2^2=\sum_i\sum_j(x_{ij}-\bar{x}_j)^2$
Manhattan distance	$\sum_i\sum_j\lvert x_{ij}-\text{median}(x_j)\rvert$
Bernoulli log-likelihood	$2\left[\sum_i\sum_j -x_{ij}\text{logit}(\bar{x}_j)+\log(1+e^{\text{logit}(\bar{x}_j)})\right]$
Binomial deviance	$2\left[\sum_i\sum_j -x_{ij}\log\bar{x}_j-(1-x_{ij})\log(1-\bar{x}_j)\right]$

Here $\bar{x}_j$ refers to the mean for the j^{th} column/feature while median(x_j) refers to the median for the j^{th} column. Simple calculation shows that the Bernoulli log-likelihood is equivalent to binomial deviance by noting $\text{logit}(\bar{x}_j)=\log\frac{\bar{x}_j}{1-\bar{x}_j}$.

Appendix J.: Choice of Tuning Parameters

In this appendix, we propose two different approaches to select tuning parameters γ and α in the iGecco+ problem; γ controls the number of clusters while α controls the number of features selected. We first consider stability selection based approach, which has been shown to enjoy nice statistical properties such as selection consistency (Wang, 2010; Fang and Wang, 2012) and been adopted in practice with strong empirical performance (Wang et al., 2018). Next, to reduce computation, we consider information criterion based approaches. Finally, we demonstrate empirical results when the number of clusters and features are not fixed but estimated based on the data.

J.1 Stability Selection Based Approach

We first adopt the stability selection based approach for tuning parameter selection and follow closely the approach described in the work of Wang (2010); Fang and Wang (2012). We choose the stability selection based approach because i) its selection consistency has been established and ii) Wang et al. (2018) adopted similar approach for tuning parameter selection and demonstrated strong empirical performance. The rationale behind stability selection is that a good clustering algorithm with optimal tuning parameter should produce clusterings that do not vary much with respect to a small perturbation to the training samples.

By Wang (2010); Fang and Wang (2012), a clustering $\psi(x)$ is defined as a mapping $\psi:\mathbb{R}^p\to\{1,\ldots,q\}$ where q is the given number of clusters. Here, we use q as the number of clusters since we have already used k to represent the k^{th} data view $\mathbf{X}^{(k)}$. A clustering algorithm $\boldsymbol{\Psi}(\cdot;q)$ with a given number of clusters $q\geq 2$ yields a clustering mapping $\psi(x)$ when applied to a sample. Here, we choose the number of clusters q, which is equivalent to choosing optimal γ since we can yield the cluster assignment and corresponding q

for each γ. To account for the tuning parameter α for feature selection, we further denote the clustering algorithm as $\boldsymbol{\Psi}(\cdot; q, \alpha)$. Then, in our case, for any given pair of q and α, two clustering results can be obtained from two sets of bootstrapped samples. Then the clustering distance d, defined by Fang and Wang (2012), can be computed to measure the dissimilarity between two clustering results. We repeat the procedure multiple times and the optimal tuning parameter pair $(\hat{q}, \hat{\alpha})$ is the one that minimizes the average clustering instability. This gives Algorithm 14, which uses stability selection to choose tuning parameters.

Algorithm 14

Choice of number of clusters q and feature penalty γ in Gecco+/iGecco+ using stability selection

1. Generate B independent bootstrap sample-pairs $\left(X_b, \widetilde{X}_b\right)$, $b = 1, \cdots, B$.

2. Construct $\boldsymbol{\Psi}_{X_b, q, \alpha}$ and $\boldsymbol{\Psi}_{\widetilde{X}_b, q, \alpha}$ based on $\left(X_b, \widetilde{X}_b\right)$, $b = 1, \cdots, B$.

3. For each pair, $\boldsymbol{\Psi}_{X_b, q, \alpha}$ and $\boldsymbol{\Psi}_{\widetilde{X}_b, q, \alpha}$, calculate their clustering distance $d(\boldsymbol{\Psi}_{X_b, q, \alpha}, \boldsymbol{\Psi}_{\widetilde{X}_b, q, \alpha})$ defined by Fang and Wang (2012). Then the clustering instability $\mathcal{S}(\boldsymbol{\Psi}, q, \alpha)$ can be estimated by

$$\widehat{\mathcal{S}}_B(\boldsymbol{\Psi}, q, \alpha) = \frac{1}{B}\sum_{b=1}^{B} d(\boldsymbol{\Psi}_{X_b, q, \alpha}, \boldsymbol{\Psi}_{\widetilde{X}_b, q, \alpha}) .$$

4. Finally, the optimal number of clusters q and feature penalty α can be estimated by

$$(\hat{q}, \hat{\alpha}) = \underset{2 \le q \le Q, \alpha}{\operatorname{argmin}} \widehat{\mathcal{S}}_B(\boldsymbol{\Psi}, q, \alpha) .$$

J.2 Information Criterion Based Approach

While choosing number of clusters based on stability selection has been shown to enjoy nice properties such as selection consistency (Wang, 2010), such methods are always computationally burdensome. To address this, on the other hand, information criterion based approach has been proposed for tuning parameter selection in convex clustering (Tan and Witten, 2015). We adopt the similar approach and propose the Bayesian information criterion (BIC) based approach to choose optimal number of clusters and features; this gives Algorithm 15.

Algorithm 15

Choice of number of clusters q and feature penalty α in Gecco+/iGecco+ using BIC

Initialize: $\hat{s} = \sum_{k=1}^{K} p_k$, $\widehat{\mathcal{S}}_k = \{1, \cdots, p_k\}$, $\hat{\alpha} = 1$.

1. Fit iGecco+ with a sequence of γ and fixed $\hat{\alpha}$. Choose the number of clusters $\hat{q}$ using BIC.

$$\hat{q} = \underset{q}{\text{argmin}} \quad n \cdot \sum_{k=1}^{K} \pi_{k|\hat{S}_k} \ell_k(\mathbf{X}^{(k)}_{\hat{S}_k}, \widehat{\mathbf{U}}^{(k)}_{\hat{S}_k}) + q\log(\hat{s}n) \ .$$

2. Fit iGecco+ with a sequence of α and fixed number of clusters $\hat{q}$. Choose number of features s using BIC. Get corresponding $\hat{\alpha}$, active set $\widehat{\mathcal{S}}_k$ and total number of selected features $\hat{s} = \sum_{k=1}^{K} \left|\widehat{\mathcal{S}}_k\right|$.

$$\hat{s} = \underset{s}{\text{argmin}} \quad n \cdot \sum_{k=1}^{K} \sum_{j=1}^{p_k} \frac{\ell_k(\mathbf{X}^{(k)}_{|j}, \widehat{\mathbf{U}}^{(k)}_{|j})}{\ell_k(\mathbf{X}^{(k)}_{|j}, \widehat{\mathbf{X}}^{(k)}_{|j})} + s\hat{q}\log(n) \ .$$

3. Repeat Step 1 and 2 until $\hat{q}$ and $\hat{\alpha}$ stabilize.

Here, the submatrix $\mathbf{X}^{(k)}_{\hat{S}_k}$ corresponds to the subset of features (columns) of $\mathbf{X}^{(k)}$ that are in the active set $\hat{S}_k$ selected by iGecco+. The quantity $\pi_{k,\hat{S}_k}$ refers to $\boldsymbol{\pi}_k$ calculated on the set of selected features $\hat{S}_k$, i.e., $\frac{1}{\ell_k(\mathbf{X}^{(k)}_{\hat{S}_k}, \widetilde{\mathbf{X}}^{(k)}_{\hat{S}_k})}$.

In summary, in step 1, we choose number of clusters only based on the selected features; in step 2, we choose number of features with the optimal number of clusters $\hat{q}$ estimated from the previous step. Note in step 1, $\widehat{\mathbf{U}}^{(k)}_{\hat{S}_k}$ is a function of number of clusters q while in step 2, $\widehat{\mathbf{U}}^{(k)}_{\cdot j}$ is a function of whether the feature is selected. Specifically, in step 1, since all features in $\hat{S}_k$ are selected, $\widehat{\mathbf{U}}^{(k)}_{\hat{S}_k}$ correspond to the cluster centroids for each cluster and change with respect to the number of clusters. In step 2, by iGecco+, if the j^{th} feature in the k^{th} data view is selected, $\widehat{\mathbf{U}}^{(k)}_{\cdot j}$ still correspond to the cluster centroids which are different for observations across different clusters; if the j^{th} feature is not selected, $\widehat{\mathbf{U}}^{(k)}_{\cdot j}$ corresponds to the same constant, i.e., loss-specific center $\widehat{\mathbf{X}}^{(k)}_{\cdot j}$ for that feature.

The criterion to minimize is inspired by the BIC approach for convex clustering proposed by Tan and Witten (2015). Notice that we use different criterion in step 1 and 2 to choose number of clusters and features respectively. When choosing the number of clusters, we use the same loss function as in iGecco since all the features are selected. When choosing the number of features, for the presence of noise features, we find that the weighted loss function with respect to each feature in step 2 works better, as the criterion in step 1 downweights the contribution of signal features by adding informative terms along with the noise terms in the denominator. (When a feature is not selected, $\ell_k(\mathbf{X}_{\cdot j}^{(k)}, \widehat{\mathbf{U}}_{\cdot j}^{(k)})$ equals $\ell_k(\mathbf{X}_{\cdot j}^{(k)}, \widetilde{\mathbf{X}}_{\cdot j}^{(k)})$; when a feature is selected, $\ell_k(\mathbf{X}_{\cdot j}^{(k)}, \widehat{\mathbf{U}}_{\cdot j}^{(k)})$ is less than $\ell_k(\mathbf{X}_{\cdot j}^{(k)}, \widetilde{\mathbf{X}}_{\cdot j}^{(k)})$; if we use the unweighted criterion in step 1, the noise terms will dominate this criterion given a large number of noise features.) The degrees of freedom for choosing number of clusters in step 1 is q while in step 2, the degrees of freedom is $s\hat{q}$ since we need to estimate $\hat{q}$ number of cluster centroids for each selected feature.

Stability selection is known to be more stable in terms of choosing number of clusters, with selection consistency theoretically established in literature. Meanwhile, BIC works better in choosing number of features in practice and saves much more computation compared with stability selection. Hence, to take full advantage of both approaches, we propose a sequential tuning parameter selection procedure, outlined in Algorithm 16. We demonstrate the clustering and variable selection accuracy results in Appendix J.3 using the proposed two tuning parameter selection approaches.

Algorithm 16

Choice of number of clusters q and feature penalty α in Gecco+/iGecco+ using stability selection + BIC

1. Choose the number of clusters $\hat{q}$ using Algorithm 14 with $\alpha = 1$ (stability selection).
2. Fit iGecco+ with a sequence of α and fixed number of clusters $\hat{q}$. Choose α using BIC based on Step 2 of Algorithm 15.

Based on the algorithms above for tuning parameter selection and the adaptive iGecco+ Algorithm 1 (with oracle number of clusters and features) to choose the feature weights, we propose Algorithm 17, the alternative adaptive iGecco+ when the number of clusters or features are not known a priori. Notice that in step 2, we do not perform tuning parameter selection for number of clusters q, as we are only interested in some type of adaptive feature selection to weigh the features.

Algorithm 17

Adaptive iGecco+ when the number of clusters or features are not known a priori

1. Fit iGecco+ with $\alpha = 1$, $\zeta_j^{(k)} = 1$ and a sequence of γ.
2. Find γ which gives non-trivial number of clusters, say $q = 2$; Get the estimate $\widehat{\mathbf{U}}^{(k)}$.

3. Update the feature weights $\hat{\zeta}_j^{(k)} = 1/\|\widehat{\mathbf{U}}_{\cdot j}^{(k)} - \tilde{x}_j^{(k)} \cdot \mathbf{1}_n\|_2$ and fusion weights $\widehat{w}_{ij} = \boldsymbol{I}_{ij}^k \exp(-\phi \hat{d}(\mathbf{X}_{i\cdot}, \mathbf{X}_{i'\cdot}))$, where

$$\hat{d}(\mathbf{X}_{i\cdot}, \mathbf{X}_{i'\cdot}) = \sum_{k=1}^{K} \sum_{j=1}^{p_k} \frac{\|\widehat{\mathbf{U}}_{\cdot j}^{(k)} - \tilde{x}_j^{(k)}\big|\mathbf{1}_n\|_2}{\max_j \|\widehat{\mathbf{U}}_{\cdot j}^{(k)} - \tilde{x}_j^{(k)}\big|\mathbf{1}_n\|_2} \cdot \frac{1}{\pi_k} \cdot d_{ii'j}^{(k)}$$

Fit adaptive iGecco+ with $\hat{\boldsymbol{\zeta}}$, $\widehat{\mathbf{w}}$ and a sequence of γ and α; Find optimal γ and α using the tuning parameter selection procedure in Algorithm 15 or 16.

J.3 Empirical Results in Simulation and Real Data Example

In this section, we demonstrate the empirical results when the number of clusters and features are not fixed but estimated based on the data. Specifically, we apply the two tuning parameter selection schemes proposed above: i) BIC based approach (Algorithm 17 + 15) and ii) stability selection + BIC based approach (Algorithm 17 + 16). We show the results of the tables in Section 4 and 5 when the number of clusters and features are estimated and not fixed to be the oracle. For simulation studies, we show the results of iGecco and iGecco+; Gecco and Gecco+ are special cases of these two. Moreover, we apply the proposed two tuning parameter selection approaches to the three real data examples in Section 5.

For iGecco, the stability selection based approach simplifies to choosing the number of clusters q in Algorithm 14; the BIC based approach refers to using the criterion in step 1 of Algorithm 15 to choose the number of clusters.

For iGecco+, we show the overall F1-score and number of selected features across all three data views. Recall that each data view has 10 true features and hence there are 30 true features in total. Also, the optimal number of clusters is $q = 3$. We compare the results with iClusterPlus when the number of clusters and features are not fixed. Note we only include estimated number of clusters for iClusterPlus as there is no tuning parameter for the number of selected features. (The authors mentioned feature selection could be achieved by selecting the top features based on Lasso coefficient estimates.)

Table 13 and 14 show the results of Table 3, 4 and 5 in Section 4.2 when the number of clusters (and features in iGecco+) is not fixed but estimated based on the data. Table 13 and 14 suggest that our proposed BIC based approach selects the correct number of clusters and features most of the time. On the other hand, information criterion based approaches save much more computation than stability selection. Hence, we recommend the BIC based approach for choosing tuning parameters which demonstrates strong empirical performance and saves computation. Yet, one might have their own justified choice of approach or information criterion to select tuning parameters.

Table 13:

Adjusted Rand index and estimated number of clusters of iGecco and iCluster for mixed multi-view data of Table 3 in Section 4.2 when the number of clusters is not fixed but estimated based on the data.

	S1		S2	
	ARI	# of Clusters	ARI	# of Clusters
iGecco with BIC	0.93 (5.2e-3)	3.10 (1.0e-1)	0.98 (2.2e-2)	3.00 (0.0e-0)
iGecco with Stability Selection	0.92 (8.1e-3)	3.20 (2.0e-1)	0.89 (4.1e-2)	4.00 (3.9e-1)
iCluster+ with $\lambda = 0$	0.92 (4.1e-2)	3.20 (2.0e-1)	0.67 (1.9e-2)	3.40 (1.6e-1)

Table 14:

Adjusted Rand index, F_1 score, along with estimated number of clusters and features of adaptive iGecco+ and iClusterPlus for high-dimensional mixed multi-view data of Table 4 and 5 in Section 4.2 when the number of clusters and features are not fixed but estimated based on the data. We only include estimated number of clusters for iClusterPlus as there is no tuning parameter for the number of selected features.

	S3		S4		S5		S6	
	ARI	# of Clusters	ARI	# of Clusters	ARI	# of Clusters	ARI	# of Clusters
Algorithm 17 + 15 (BIC)	0.97 (7.8e-3)	3.00 (0.0e-0)	0.97 (1.3e-2)	3.10 (1.0e-1)	1.00 (0.0e-0)	3.00 (0.0e-0)	1.00 (0.0e-0)	3.00 (0.0e-0)
Algorithm 17 + 16 (SS+BIC)	0.93 (4.1e-2)	2.90 (1.0e-1)	0.89 (5.5e-2)	2.80 (1.3e-1)	0.99 (1.2e-2)	3.10 (1.0e-1)	0.82 (8.1e-2)	4.90 (9.1e-1)
iCluster+	0.53 (8.1e-2)	2.70 (3.0e-1)	0.72 (5.3e-2)	3.50 (3.1e-1)	0.63 (2.4e-2)	3.50 (2.2e-1)	0.60 (1.4e-2)	3.30 (1.5e-1)

	S3		S4		S5		S6	
	F1-score	# of Features	F1-score	# of Features	F1-score	# of Features	F1-score	# of Features
Algorithm 17 + 15 (BIC)	0.93 (1.1e-2)	30.00 (1.1e-0)	0.95 (1.9e-2)	31.60 (7.2e-1)	0.99 (6.3e-3)	30.50 (4.0e-1)	0.99 (6.5e-3)	30.90 (4.1e-1)
Algorithm 17 + 16 (SS+BIC)	0.93 (1.2e-2)	29.60 (1.2e-0)	0.96 (1.5e-2)	29.40 (6.5e-1)	0.99 (6.3e-3)	30.50 (4.0e-1)	0.99 (6.0e-3)	30.10 (3.8e-1)

Also, we apply the proposed two tuning parameter selection approaches to the real data examples in Section 5. Table 15 shows the estimated number of clusters. Again, the BIC based approach selects the correct number of clusters for all three cases.

Table 15:

Adjusted Rand index, along with estimated number of clusters using adaptive (i)Gecco+ for real data of Table 6, 8 and 10 in Section 5 when the number of clusters and features are not fixed but estimated based on the data. For the first two data set, we show the results of Manhattan Gecco+.

	Authors Data		TCGA Data		Multi-omics Data	
	ARI	# of Clusters	ARI	# of Clusters	ARI	# of Clusters
Algorithm 17 + 15 (BIC)	0.96	4.00	0.76	3.00	0.71	3.00
Algorithm 17 + 16 (SS+BIC)	0.96	4.00	0.59	2.00	0.52	2.00

Appendix K.: Stable to Perturbations of Data

In this appendix, we demonstrate that clustering assignments of iGecco+ are stable to perturbations in the data as shown in Proposition 2 of Section 2.4.

To show this, we include a simulation study similar to the one by Chi et al. (2017). We first apply adaptive iGecco+ on the original data to obtain baseline clustering. Then we add i.i.d. noise to each data view to create a perturbed data set on which we apply the same iGecco+ method. Specifically, for Gaussian data view, we add i.i.d. $N(0, \sigma^2)$ noise where $\sigma = 0.5$, 1.0, 1.5; for count data view, we add i.i.d. Poisson noise; for binary data view, we randomly shuffle a small proportion of the entries. We compute the adjusted Rand index between the baseline clustering and the one obtained on the perturbed data. We adopt the same approach for other existing methods. Table 16 shows the average adjusted Rand index of 10 replicates. For all values of σ, we see that iGecco+ tends to produce the most stable results.

Table 16:

Stability and reproducibility of adaptive iGecco+ on simulated data. Adaptive iGecco+ and other existing methods are applied to the simulated data to obtain baseline clusterings. We then perturb the data by adding i.i.d. noise. Specifically, for Gaussian data view, we add i.i.d. $N(0, \sigma^2)$ noise where $\sigma = 0.5$, 1.0, 1.5; for count data view, we add i.i.d. Poisson noise; for binary data view, we randomly shuffle a small proportion of the entries. We compute the adjusted Rand index (ARI) between the baseline clustering and the one obtained on the perturbed data.

σ	A iGecco+	Hclust: Euclidean	Hclust: Gower	iCluster+	BCC
0.5	1.00 (0.0e-0)	0.85 (5.1e-2)	0.90 (1.4e-2)	0.69 (2.6e-2)	0.92 (1.6e-2)
1.0	1.00 (0.0e-0)	0.69 (8.3e-2)	0.89 (1.8e-2)	0.71 (3.0e-2)	0.90 (1.7e-2)
1.5	0.95 (3.4e-2)	0.66 (2.8e-2)	0.86 (2.0e-2)	0.70 (2.9e-2)	0.90 (2.0e-2)

Appendix L.: Noisy Data Sources

In this appendix, we show the performance of iGecco+ when a purely noisy data view is observed.

Often in real data, not all the data views observed contain clustering information, i.e., one or more data views might be pure noise. Our iGecco+ is able to filter out noisy data views by adaptively shrinking all the (noise) features in these data sets towards the loss-specific centers. We include a simulation study to demonstrate the performance of iGecco+ in the presence of some purely noisy data sets. Similar to the base simulation, each simulated data set consists of $n = 120$ observations with 3 clusters. Each cluster has an equal number of observations. Only the first data view contains clustering signal with the first 30 features being informative while the rest features being noisy; the rest two data views are pure noise.

Table 17 shows that iGecco+ still performs well in the presence of noise data sources by adaptively shrinking noise features.

Table 17:

Adjusted Rand index of different methods in the presence of noisy data sources

Method	Adjusted Rand Index
Hclust: $[\mathbf{X}_1\mathbf{X}_2\mathbf{X}_3]$ - Euclidean	0.40 (4.9e-2)
Hclust: $[\mathbf{X}_1\mathbf{X}_2\mathbf{X}_3]$ - Gower	0.33 (4.8e-2)
iCluster+	0.88 (5.8e-2)
Bayesian Consensus Clustering	0.00 (2.8e-4)
Adaptive iGecco+	**0.99** (5.6e-3)

Appendix M.: Computation Time

In this section, we provide some computation run time results of iGecco(+) with different sample sizes and dimensions. We include results for both run times per iteration of the ADMM algorithm in Table 18 as well as the full training time for tuning parameter selection in Table 19. All timing results are run on a Dell XPS 15 with a 2.4 GHz Intel i5 processor and 8 GB of 2666 MHz DDR4 memory.

For training, we use BIC based approach to select tuning parameters for both iGecco and iGecco+ as it works well in practice and saves computation. It takes more time to train iGecco+ than iGecco as iGecco+ needs to select two tuning parameters (the number of clusters and features). Yet, our BIC based approach selects optimal tuning parameters in a reasonable amount of time. Note, for sample size $n = 120$ and feature size $p_1 = 200$, $p_2 = 100$, $p_3 = 50$, it takes iClusterPlus hours for tuning parameter selection (using tune.iClusterPlus function in R).

Table 18:

Run time results per iteration of iGecco(+) with different sample sizes and dimensions; the first three experiments use iGecco while the rest use iGecco+

	Sample Size n	p_1	p_2	p_3	Computation Time (in seconds)
iGecco	120	10	10	10	0.0018

	Sample Size n	p_1	p_2	p_3	Computation Time (in seconds)
	300	10	10	10	0.0044
	120	30	30	30	0.0038
iGecco+	120	200	100	50	0.0041
	300	200	100	50	0.0105
	120	300	200	100	0.0055
	120	400	200	100	0.0061

Table 19:

Training run time results of iGecco(+) with different sample sizes and dimensions; the first three experiments use iGecco with BIC to choose number of clusters while the rest use adaptive iGecco+ with BIC to choose number of clusters and features.

	Sample Size n	p_1	p_2	p_3	Computation Time (in seconds)
iGecco	120	10	10	10	0.89
	300	10	10	10	2.58
	120	30	30	30	2.55
iGecco+	120	200	100	50	63.72
	300	200	100	50	117.94
	120	300	200	100	90.77
	120	400	200	100	104.04

References

Acar Evrim, Kolda Tamara G, and Dunlavy Daniel M. All-at-once optimization for coupled matrix and tensor factorizations. ArXiv Pre-Print 1105.3422, 2011. https://arxiv.org/abs/1105.3422.

Ahmad Amir and Dey Lipika. A *k*-mean clustering algorithm for mixed numeric and categorical data. Data & Knowledge Engineering, 63(2):503–527, 2007. doi:10.1016/j.datak.2007.03.016.

Akay Özlem and Yüksel Güzin. Clustering the mixed panel dataset using Gower's distance and *k*-prototypes algorithms. Communications in Statistics-Simulation and Computation, 47(10):3031–3041, 2018. doi:10.1080/03610918.2017.1367806.

Ali Bilel Ben and Massmoudi Youssef. K-means clustering based on gower similarity coefficient: A comparative study. In 2013 5th International Conference on Modeling, Simulation and Applied Optimization (ICMSAO), pages 1–5. IEEE, 2013. doi:10.1109/ICMSAO.2013.6552669.

Anders Simon and Huber Wolfgang. Differential expression analysis for sequence count data. Genome Biology, 11(10):R106, 2010. doi:10.1186/gb-2010-11-10-r106. [PubMed: 20979621]

Arthur David and Vassilvitskii Sergei. k-means++: The advantages of careful seeding. Technical report, Stanford, 2006.

Badve Sunil, Turbin Dmitry, Thorat Mangesh A, Morimiya Akira, Nielsen Torsten O, Perou Charles M, Dunn Sandi, Huntsman David G, and Nakshatri Harikrishna. FOXA1 expression in breast cancer—correlation with luminal subtype A and survival. Clinical Cancer Research, 13(15):4415–4421, 2007. doi:10.1158/1078-0432.CCR-07-0122. [PubMed: 17671124]

Baker Yulia, Tang Tiffany M, and Allen Genevera I. Feature selection for data integration with mixed multi-view data. ArXiv Pre-Print 1903.11232, 2019. https://arxiv.org/abs/1903.11232.

Banerjee Arindam, Merugu Srujana, Dhillon Inderjit S, and Ghosh Joydeep. Clustering with Bregman divergences. Journal of Machine Learning Research, 6(Oct):1705–1749, 2005. http://www.jmlr.org/papers/v6/banerjee05b.html.

Banert Sebastian, Bot Radu Ioan, and Csetnek Ernö Robert. Fixing and extending some recent results on the ADMM algorithm. ArXiv Pre-Print 1612.05057, 2016. https://arxiv.org/abs/1612.05057.

Banfield Jeffrey D and Raftery Adrian E. Model-based Gaussian and non-Gaussian clustering. Biometrics, pages 803–821, 1993. doi:10.2307/2532201.

Beck Amir and Teboulle Marc. Gradient-based algorithms with applications to signal recovery. Convex Optimization in Signal Processing and Communications, pages 42–88, 2009. doi:10.1017/CB09780511804458.003.

Bernardo JM, Bayarri MJ, Berger JO, Dawid AP, Heckerman D, Smith AFM, and West M. Bayesian clustering with variable and transformation selections. In Bayesian Statistics 7: Proceedings of the Seventh Valencia International Meeting, volume 249. Oxford University Press, USA, 2003.

Bullard James H, Purdom Elizabeth, Hansen Kasper D, and Dudoit Sandrine. Evaluation of statistical methods for normalization and differential expression in mRNA-Seq experiments. BMC Bioinformatics, 11(1):94, 2010. doi:10.1186/1471-2105-11-94. [PubMed: 20167110]

Chalise Prabhakar and Fridley Brooke L. Integrative clustering of multi-level 'omic data based on non-negative matrix factorization algorithm. PLoS ONE, 12(5):e0176278, 2017. doi:10.1371/journal.pone.0176278. [PubMed: 28459819]

Chang Xiangyu, Wang Yu, Li Rongjian, and Xu Zongben. Sparse K-means with ℓ_∞/ℓ_0 penalty for high-dimensional data clustering. ArXiv Pre-Print 1403.7890, 2014. https://arxiv.org/abs/1403.7890.

Chen Caihua, He Bingsheng, Ye Yinyu, and Yuan Xiaoming. The direct extension of ADMM for multi-block convex minimization problems is not necessarily convergent. Mathematical Programming, Series A, 155(1-2):57–79, 2016. doi:10.1007/s10107-014-0826-5.

Chi Eric C. and Lange Kenneth. Splitting methods for convex clustering. Journal of Computational and Graphical Statistics, 24(4):994–1013, 2015. doi:10.1080/10618600.2014.948181. [PubMed: 27087770]

Chi Eric C and Steinerberger Stefan. Recovering trees with convex clustering. SIAM Journal on Mathematics of Data Science, 1(3):383–407, 2019. doi:10.1137/18M121099X.

Chi Eric C., Allen Genevera I., and Baraniuk Richard G.. Convex biclustering. Biometrics, 73(1):10–19, 2017. doi:10.1111/biom.12540. [PubMed: 27163413]

Chi Eric C, Gaines Brian R, Sun Will Wei, Zhou Hua, and Yang Jian. Provable convex co-clustering of tensors. ArXiv Pre-Print 1803.06518, 2018. https://arxiv.org/abs/1803.06518.

Choi Hosik, Poythress JC, Park Cheolwoo, Jeon Jong-June, and Park Changyi. Regularized boxplot via convex clustering. Journal of Statistical Computation and Simulation, 89(7): 1227–1247, 2019. doi:10.1080/00949655.2019.1576045.

Choi Seung-Seok, Cha Sung-Hyuk, and Tappert Charles C. A survey of binary similarity and distance measures. Journal of Systemics, Cybernetics and Informatics, 8(1):43–48, 2010.

Choi Woonyoung, Porten Sima, Kim Seungchan, Willis Daniel, Plimack Elizabeth R, Hoffman-Censits Jean, Roth Beat, Cheng Tiewei, Tran Mai, Lee I-Ling, et al. Identification of distinct basal and luminal subtypes of muscle-invasive bladder cancer with different sensitivities to frontline chemotherapy. Cancer Cell, 25(2):152–165, 2014. doi:10.1016/j.ccr.2014.01.009. [PubMed: 24525232]

Cimino-Mathews Ashley, Subhawong Andrea P, Illei Peter B, Sharma Rajni, Halushka Marc K, Vang Russell, Fetting John H, Park Ben Ho, and Argani Pedram. GATA3 expression in breast carcinoma: utility in triple-negative, sarcomatoid, and metastatic carcinomas. Human Pathology, 44(7):1341–1349, 2013. doi:10.1016/j.humpath.2012.11.003. [PubMed: 23375642]

de Souza Renata M.C.R. and De Carvalho Francisco de A.T.. Clustering of interval data based on city-block distances. Pattern Recognition Letters, 25(3):353–365, 2004. doi:10.1016/j.patrec.2003.10.016.

Deng Wei and Yin Wotao. On the global and linear convergence of the generalized alternating direction method of multipliers. Journal of Scientific Computing, 66(3):889–916, 2016. doi:10.1007/s10915-015-0048-x.

Deng Wei, Lai Ming-Jun, Peng Zhimin, and Yin Wotao. Parallel multi-block ADMM with $o(1/k)$ convergence. Journal of Scientific Computing, 71(2):712–736, 2017. doi:10.1007/s10915-016-0318-2.

dos Santos Tiago R.L. and Zárate Luis E.. Categorical data clustering: What similarity measure to recommend? Expert Systems with Applications, 42(3):1247–1260, 2015. doi:10.1016/j.eswa.2014.09.012.

Fan Jianqing, Feng Yang, and Wu Yichao. Network exploration via the adaptive lasso and scad penalties. The annals of applied statistics, 3(2):521, 2009. [PubMed: 21643444]

Fang Yixin and Wang Junhui. Selection of the number of clusters via the bootstrap method. Computational Statistics & Data Analysis, 56(3):468–477, 2012. doi:10.1016/j.csda.2011.09.003.

Forbes Simon A, Bindal Nidhi, Bamford Sally, Cole Charlotte, Kok Chai Yin, Beare David, Jia Mingming, Shepherd Rebecca, Leung Kenric, Menzies Andrew, et al. COSMIC: mining complete cancer genomes in the Catalogue of Somatic Mutations in Cancer. Nucleic Acids Research, 39(suppl_1):D945–D950, 2010. doi:10.1093/nar/gkq929. [PubMed: 20952405]

Fowlkes Edward B and Mallows Colin L. A method for comparing two hierarchical clusterings. Journal of the American Statistical Association, 78(383):553–569, 1983. doi:10.2307/2288117.

Garczyk Stefan, von Stillfried Saskia, Antonopoulos Wiebke, Hartmann Arndt, Schrauder Michael G, Fasching Peter A, Anzeneder Tobias, Tannapfel Andrea, Ergönenc Yavuz, Knüchel Ruth, et al. AGR3 in breast cancer: Prognostic impact and suitable serum-based biomarker for early cancer detection. PLoS ONE, 10(4):e0122106, 2015. doi:10.1371/journal.pone.0122106. [PubMed: 25875093]

Ghosh Debashis and Chinnaiyan Arul M. Mixture modelling of gene expression data from microarray experiments. Bioinformatics, 18(2):275–286, 2002. doi:10.1093/bioinformatics/18.2.275. [PubMed: 11847075]

Gower John C. A general coefficient of similarity and some of its properties. Biometrics, pages 857–871, 1971. doi:10.2307/2528823.

Hall David L and Llinas James. An introduction to multisensor data fusion. Proceedings of the IEEE, 85(1):6–23, 1997. doi:10.1109/5.554205.

Harari Daniel and Yarden Yosef. Molecular mechanisms underlying ErbB2/HER2 action in breast cancer. Oncogene, 19(53):6102, 2000. doi:10.1038/sj.onc.1203973. [PubMed: 11156523]

Hayashi SI, Eguchi H, Tanimoto K, Yoshida T, Omoto Y, Inoue A, Yoshida N, and Yamaguchi Y. The expression and function of estrogen receptor alpha and beta in human breast cancer and its clinical application. Endocrine-Related Cancer, 10(2):193–202, 2003. doi:10.1677/erc.0.0100193. [PubMed: 12790782]

Hellton Kristoffer H and Thoresen Magne. Integrative clustering of high-dimensional data with joint and individual clusters. Biostatistics, 17(3):537–548, 2016. doi:10.1093/biostatistics/kxw005. [PubMed: 26917056]

Hoadley Katherine A, Yau Christina, Wolf Denise M, Cherniack Andrew D, Tamborero David, Ng Sam, Leiserson Max DM, Niu Beifang, McLellan Michael D, Uzunangelov Vladislav, et al. Multiplatform analysis of 12 cancer types reveals molecular classification within and across tissues of origin. Cell, 158(4):929–944, 2014. doi:10.1016/j.cell.2014.06.049. [PubMed: 25109877]

Hocking Toby Dylan, Joulin Armand, Bach Francis, and Vert Jean-Philippe. Clusterpath: An algorithm for clustering using convex fusion penalties. In Getoor Lise and Scheffer Tobias, editors, ICML 2011: Proceedings of the 28th International Conference on Machine Learning, pages 745–752. ACM, 2011. ISBN 978-1-4503-0619-5. http://www.icml-2011.org/papers/419_icmlpaper.pdf.

Hua Kaiyao, Jin Jiali, Zhao Junyong, Song Jialu, Song Hongming, Li Dengfeng, Maskey Niraj, Zhao Bingkun, Wu Chenyang, Xu Hui, et al. miR-135b, upregulated in breast cancer, promotes cell growth and disrupts the cell cycle by regulating LATS2. International Journal of Oncology, 48(5):1997–2006, 2016. doi:10.3892/ijo.2016.3405. [PubMed: 26934863]

Hubert Lawrence and Arabie Phipps. Comparing partitions. Journal of Classification, 2(1): 193–218, 1985. doi:10.1007/BF01908075.

Hummel Manuela, Edelmann Dominic, and Kopp-Schneider Annette. Clustering of samples and variables with mixed-type data. PloS ONE, 12(11):e0188274, 2017. doi:10.1371/journal.pone.0188274. [PubMed: 29182671]

Ji Jinchao, Pang Wei, Zhou Chunguang, Han Xiao, and Wang Zhe. A fuzzy *k*-prototype clustering algorithm for mixed numeric and categorical data. Knowledge-Based Systems, 30:129–135, 2012. doi:10.1016/j.knosys.2012.01.006.

Kirk Paul, Griffin Jim E, Savage Richard S, Ghahramani Zoubin, and Wild David L. Bayesian correlated clustering to integrate multiple datasets. Bioinformatics, 28(24):3290–3297, 2012. doi:10.1093/bioinformatics/bts595. [PubMed: 23047558]

Krishnapuram Balaji, Carin Lawrence, Figueiredo Mario AT, and Hartemink Alexander J. Sparse multinomial logistic regression: Fast algorithms and generalization bounds. IEEE Transactions on Pattern Analysis and Machine Intelligence, 27(6):957–968, 2005. doi:10.1109/TPAMI.2005.127. [PubMed: 15943426]

Lin Tianyi, Ma Shiqian, and Zhang Shuzhong. On the global linear convergence of the ADMM with multiblock variables. SIAM Journal on Optimization, 25(3):1478–1497, 2015. doi:10.1137/140971178.

Lin Zhouchen, Liu Risheng, and Su Zhixun. Linearized alternating direction method with adaptive penalty for low-rank representation. In Advances in neural information processing systems, pages 612–620, 2011.

Lindsten Fredrik, Ohlsson Henrik, and Ljung Lennart. Just relax and come clustering!: A convexification of k-means clustering. Linköping University Electronic Press, 2011.

Liu Risheng, Lin Zhouchen, and Su Zhixun. Linearized alternating direction method with parallel splitting and adaptive penalty for separable convex programs in machine learning. In Asian Conference on Machine Learning, pages 116–132, 2013. doi:10.1007/s10994-014-5469-5.

Lock Eric F and Dunson David B. Bayesian consensus clustering. Bioinformatics, 29(20): 2610–2616, 2013. doi:10.1093/bioinformatics/btt425. [PubMed: 23990412]

Lock Eric F, Hoadley Katherine A, Marron James Stephen, and Nobel Andrew B. Joint and individual variation explained (JIVE) for integrated analysis of multiple data types. The Annals of Applied Statistics, 7(1):523, 2013. doi:10.1214/12-AOAS597. [PubMed: 23745156]

Lu Canyi, Li Huan, Lin Zhouchen, and Yan Shuicheng. Fast proximal linearized alternating direction method of multiplier with parallel splitting. In AAAI, pages 739–745, 2016. https://arxiv.org/abs/1511.05133.

van der Maaten Laurens and Hinton Geoffrey. Visualizing data using t-SNE. Journal of Machine Learning Research, 9(Nov):2579–2605, 2008. http://www.jmlr.org/papers/v9/vandermaaten08a.html.

Marioni John C, Mason Christopher E, Mane Shrikant M, Stephens Matthew, and Gilad Yoav. RNA-seq: an assessment of technical reproducibility and comparison with gene expression arrays. Genome Research, 2008. doi:10.1101/gr.079558.108.

Mehra Rohit, Varambally Sooryanarayana, Ding Lei, Shen Ronglai, Sabel Michael S, Ghosh Debashis, Chinnaiyan Arul M, and Kleer Celina G. Identification of GATA3 as a breast cancer prognostic marker by global gene expression meta-analysis. Cancer Research, 65 (24):11259–11264, 2005. doi:10.1158/0008-5472.CAN-05-2495. [PubMed: 16357129]

Meng Chen, Helm Dominic, Frejno Martin, and Kuster Bernhard. moCluster: Identifying joint patterns across multiple omics data sets. Journal of Proteome Research, 15(3): 755–765, 2015. doi:10.1021/acs.jproteome.5b00824. [PubMed: 26653205]

Mo Qianxing, Wang Sijian, Seshan Venkatraman E, Olshen Adam B, Schultz Nikolaus, Sander Chris, Scott Powers R, Ladanyi Marc, and Shen Ronglai. Pattern discovery and cancer gene identification in integrated cancer genomic data. Proceedings of the National Academy of Sciences, 110(11):4245–4250, 2013. doi:10.1073/pnas.1208949110.

Mo Qianxing, Shen Ronglai, Guo Cui, Vannucci Marina, Chan Keith S, and Hilsenbeck Susan G. A fully Bayesian latent variable model for integrative clustering analysis of multi-type omics data. Biostatistics, 19(1):71–86, 2017. doi:10.1093/biostatistics/kxx017.

Pan Wei and Shen Xiaotong. Penalized model-based clustering with application to variable selection. Journal of Machine Learning Research, 8(May):1145–1164, 2007. http://www.jmlr.org/papers/v8/pan07a.html.

Panahi Ashkan, Dubhashi Devdatt, Johansson Fredrik D, and Bhattacharyya Chiranjib. Clustering by sum of norms: Stochastic incremental algorithm, convergence and cluster recovery. In International conference on machine learning, pages 2769–2777. PMLR, 2017.

Parikh Neal, Boyd Stephen, et al. Proximal algorithms. Foundations and Trends® in Optimization, 1(3):127–239, 2014. doi:10.1561/2400000003.

Pelckmans Kristiaan, de Brabanter Joseph, Suykens Johan, and de Moor Bart. Convex clustering shrinkage. In PASCAL Workshop on Statistics and Optimization of Clustering, 2005.

Radchenko Peter and Mukherjee Gourab. Convex clustering via ℓ_1 fusion penalization. Journal of the Royal Statistical Society, Series B: Statistical Methodology, 79(5):1527–1546, 2017. doi:10.1111/rssb.12226.

Raftery Adrian E and Dean Nema. Variable selection for model-based clustering. Journal of the American Statistical Association, 101(473):168–178, 2006. doi:10.1198/016214506000000113.

Robinson Mark D and Oshlack Alicia. A scaling normalization method for differential expression analysis of RNA-seq data. Genome Biology, 11(3):R25, 2010. doi:10.1186/gb-2010-11-3-r25. [PubMed: 20196867]

Savage Richard S, Ghahramani Zoubin, Griffin Jim E, Kirk Paul, and Wild David L. Identifying cancer subtypes in glioblastoma by combining genomic, transcriptomic and epigenomic data. ArXiv Pre-Print 1304.3577, 2013. https://arxiv.org/abs/1304.3577.

Shefi Ron and Teboulle Marc. Rate of convergence analysis of decomposition methods based on the proximal method of multipliers for convex minimization. SIAM Journal on Optimization, 24(1):269–297, 2014. doi:10.1137/130910774.

Shen Ronglai, Olshen Adam B, and Ladanyi Marc. Integrative clustering of multiple genomic data types using a joint latent variable model with application to breast and lung cancer subtype analysis. Bioinformatics, 25(22):2906–2912, 2009. doi:10.1093/bioinformatics/btp543. [PubMed: 19759197]

Shen Ronglai, Mo Qianxing, Schultz Nikolaus, Seshan Venkatraman E, Olshen Adam B, Huse Jason, Ladanyi Marc, and Sander Chris. Integrative subtype discovery in glioblastoma using iCluster. PloS ONE, 7(4):e35236, 2012. doi:10.1371/journal.pone.0035236. [PubMed: 22539962]

Shen Ronglai, Wang Sijian, and Mo Qianxing. Sparse integrative clustering of multiple omics data sets. The Annals of Applied Statistics, 7(1):269, 2013. doi:10.1214/12-AOAS578. [PubMed: 24587839]

Si Yaqing, Liu Peng, Li Pinghua, and Brutnell Thomas P. Model-based clustering for RNA-seq data. Bioinformatics, 30(2):197–205, 2013. doi:10.1093/bioinformatics/btt632. [PubMed: 24191069]

Simon Noah, Friedman Jerome, and Hastie Trevor. A blockwise descent algorithm for group-penalized multiresponse and multinomial regression. ArXiv Pre-Print 1311.6529, 2013. https://arxiv.org/abs/1311.6529.

Sui Xiaopeng, Xu Li, Qian Xiaoning, and Liu Tie. Convex clustering with metric learning. Pattern Recognition, 81:575–584, 2018. doi:10.1016/j.patcog.2018.04.019.

Sun Defeng, Toh Kim-Chuan, and Yuan Yancheng. Convex clustering: model, theoretical guarantee and efficient algorithm. Journal of Machine Learning Research, 22(9):1–32, 2021.

Sun Wei, Wang Junhui, and Fang Yixin. Regularized k-means clustering of high-dimensional data and its asymptotic consistency. Electronic Journal of Statistics, 6:148–167, 2012. doi:10.1214/12-EJS668.

Tamayo Pablo, Scanfeld Daniel, Ebert Benjamin L, Gillette Michael A, Roberts Charles WM, and Mesirov Jill P. Metagene projection for cross-platform, cross-species characterization of global transcriptional states. Proceedings of the National Academy of Sciences, 104(14): 5959–5964, 2007. doi:10.1073/pnas.0701068104.

Tan Kean Ming and Witten Daniela. Statistical properties of convex clustering. Electronic Journal of Statistics, 9(2):2324–2347, 2015. doi:10.1214/15-EJS1074. [PubMed: 27617051]

Tang Tiffany M and Allen Genevera I. Integrated principal components analysis. ArXiv Pre-Print 1810.00832, 2018. https://arxiv.org/abs/1810.00832.

The Cancer Genome Atlas Network. Comprehensive molecular portraits of human breast tumours. Nature, 490(7418):61–70, 2012. doi:10.1038/nature11412. [PubMed: 23000897]

van de Geer Sara, Bühlmann Peter, Zhou Shuheng, et al. The adaptive and the thresholded lasso for potentially misspecified models (and a lower bound for the lasso). Electronic Journal of Statistics, 5:688–749, 2011.

Wang Binhuan, Zhang Yilong, Sun Will Wei, and Fang Yixin. Sparse convex clustering. Journal of Computational and Graphical Statistics, 27(2):393–403, 2018. doi:10.1080/10618600.2017.1377081.

Wang Bo, Mezlini Aziz M, Demir Feyyaz, Fiume Marc, Tu Zhuowen, Brudno Michael, Haibe-Kains Benjamin, and Goldenberg Anna. Similarity network fusion for aggregating data types on a genomic scale. Nature Methods, 11(3):333, 2014. doi:10.1038/nmeth.2810. [PubMed: 24464287]

Wang Junhui. Consistent selection of the number of clusters via crossvalidation. Biometrika, 97(4):893–904, 2010. doi:10.1093/biomet/asq061.

Wang Qi, Gong Pinghua, Chang Shiyu, Huang Thomas S, and Zhou Jiayu. Robust convex clustering analysis. In Data Mining (ICDM), 2016 IEEE 16th International Conference on, pages 1263–1268. IEEE, 2016. doi:10.1109/ICDM.2016.0170.

Wang Sijian and Zhu Ji. Variable selection for model-based high-dimensional clustering and its application to microarray data. Biometrics, 64(2):440–448, 2008. doi:10.1111/j.1541-0420.2007.00922.x. [PubMed: 17970821]

Wangchamhan Tanachapong, Chiewchanwattana Sirapat, and Sunat Khamron. Efficient algorithms based on the *k*-means and Chaotic League Championship Algorithm for numeric, categorical, and mixed-type data clustering. Expert Systems with Applications, 90:146–167, 2017. doi:10.1016/j.eswa.2017.08.004.

Weylandt Michael, Nagorski John, and Allen Genevera I. Dynamic visualization and fast computation for convex clustering via algorithmic regularization. Journal of Computational and Graphical Statistics, 29(1):87–96, 2020. doi:10.1080/10618600.2019.1629943. [PubMed: 32982130]

Witten Daniela M and Tibshirani Robert. A framework for feature selection in clustering. Journal of the American Statistical Association, 105(490):713–726, 2010. doi:10.1198/jasa.2010.tm09415. [PubMed: 20811510]

Wu Chong, Kwon Sunghoon, Shen Xiaotong, and Pan Wei. A new algorithm and theory for penalized regression-based clustering. The Journal of Machine Learning Research, 17(1): 6479–6503, 2016.

Xiang Shuo, Yuan Lei, Fan Wei, Wang Yalin, Thompson Paul M, and Ye Jieping. Multi-source learning with block-wise missing data for Alzheimer's disease prediction. In Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pages 185–193. ACM, 2013. doi:10.1145/2487575.2487594.

Yang Zi and Michailidis George. A non-negative matrix factorization method for detecting modules in heterogeneous omics multi-modal data. Bioinformatics, 32(1):1–8, 2015. doi:10.1093/bioinformatics/btv544. [PubMed: 26377073]

Yu Guan, Li Quefeng, Shen Dinggang, and Liu Yufeng. Optimal sparse linear prediction for block-missing multi-modality data without imputation. Journal of the American Statistical Association, (just-accepted):1–35, 2019. doi:10.1080/01621459.2019.1632079. [PubMed: 34012183]

Yu Yue, Luo Wei, Yang Zheng-Jun, Chi Jiang-Rui, Li Yun-Rui, Ding Yu, Ge Jie, Wang Xin, and Cao Xu-Chen. miR-190 suppresses breast cancer metastasis by regulation of TGF-β-induced epithelial–mesenchymal transition. Molecular Cancer, 17(1):70, 2018. doi:10.1186/s12943-018-0818-9. [PubMed: 29510731]

Zhang Shihua, Li Qingjiao, Liu Juan, and Zhou Xianghong Jasmine. A novel computational framework for simultaneous integration of multiple types of genomic data to identify microRNA-gene regulatory modules. Bioinformatics, 27(13):i401–i409, 2011. doi:10.1093/bioinformatics/btr206. [PubMed: 21685098]

Zhang Shihua, Liu Chun-Chi, Li Wenyuan, Shen Hui, Laird Peter W, and Zhou Xianghong Jasmine. Discovery of multi-dimensional modules by integrative analysis of cancer genomic data. Nucleic Acids Research, 40(19):9379–9391, 2012. doi:10.1093/nar/gks725. [PubMed: 22879375]

Zhou Shuheng, van de Geer Sara, and Bühlmann Peter. Adaptive lasso for high dimensional regression and gaussian graphical modeling. arXiv preprint arXiv:0903.2515, 2009.

Zhu Changbo, Xu Huan, Leng Chenlei, and Yan Shuicheng. Convex optimization procedure for clustering: Theoretical revisit. In Advances in Neural Information Processing Systems,

pages 1619–1627, 2014. https://papers.nips.cc/paper/5307-convex-optimization-procedure-for-clustering-theoretical-revisit.

Zou Hui. The adaptive lasso and its oracle properties. Journal of the American Statistical Association, 101(476):1418–1429, 2006. doi:10.1198/016214506000000735.

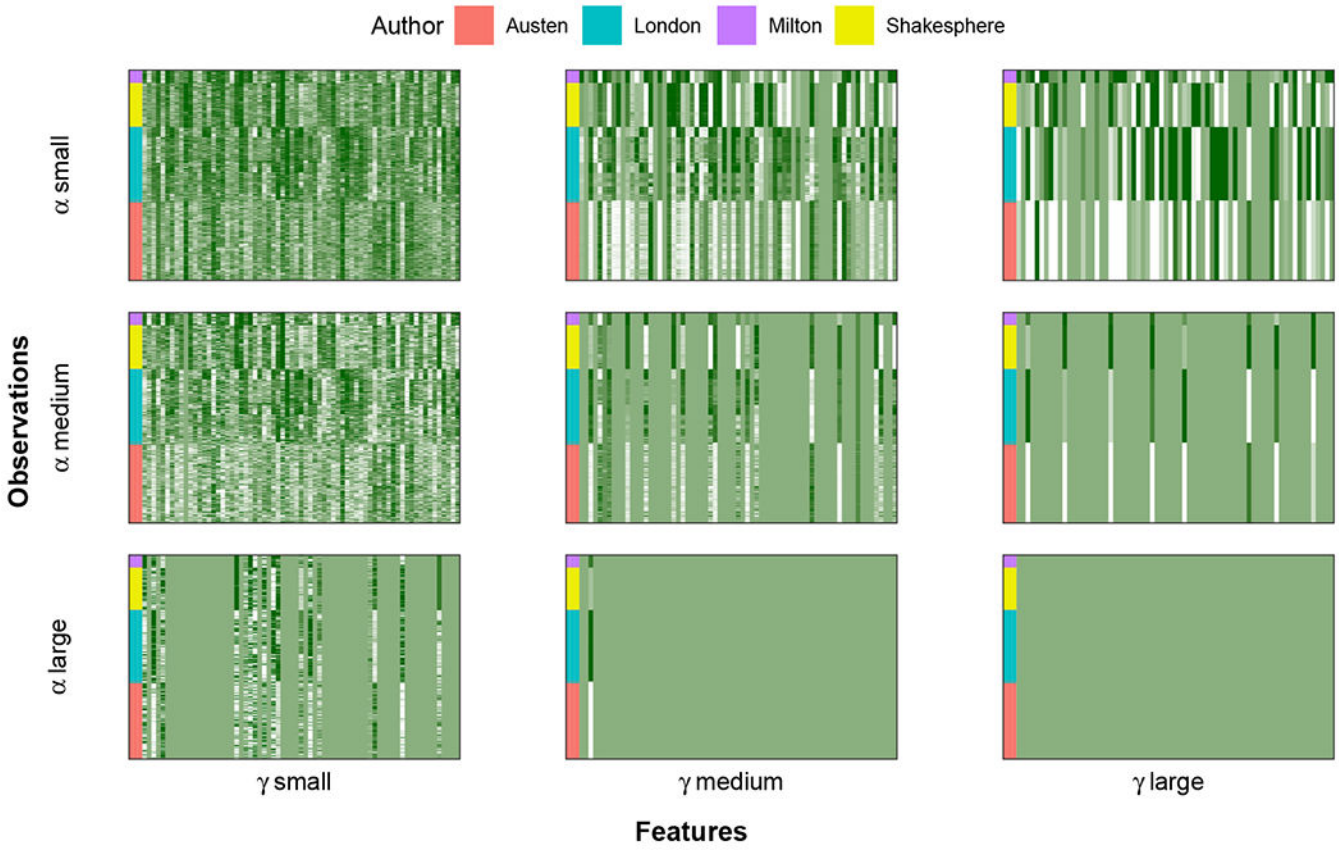

Figure 1:

Regularization path of Gecco+ solutions $\widehat{\mathbf{U}}(\gamma, \alpha)$ for authors data. From left to right, we increase the parameter for fusion penalty γ. From top to bottom, we increase the parameter for feature penalty α. The interpretation of regularization path is discussed in more detail in Section 2.4.

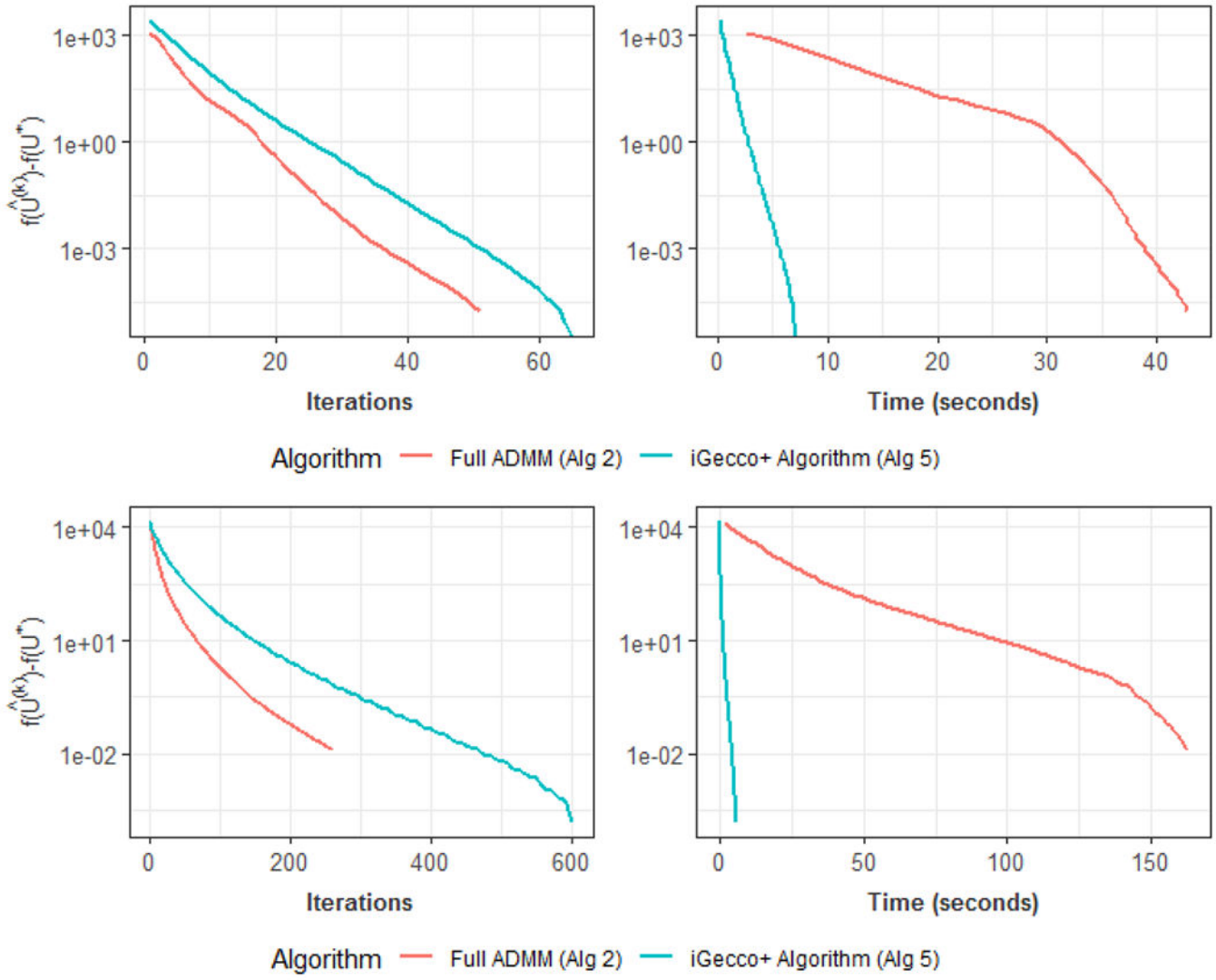

Figure 2:

Comparisons of full ADMM and one-step ADMM to solve Gecco+ with Poisson log-likelihood (top panel, differentiable loss) and Gecco+ with Manhattan distances (bottom panel, non-differentiable loss). Left plots show the number of iterations needed to converge while right plots show computation time. Algorithm with one-step update to solve the sub-problem saves much more computational time.

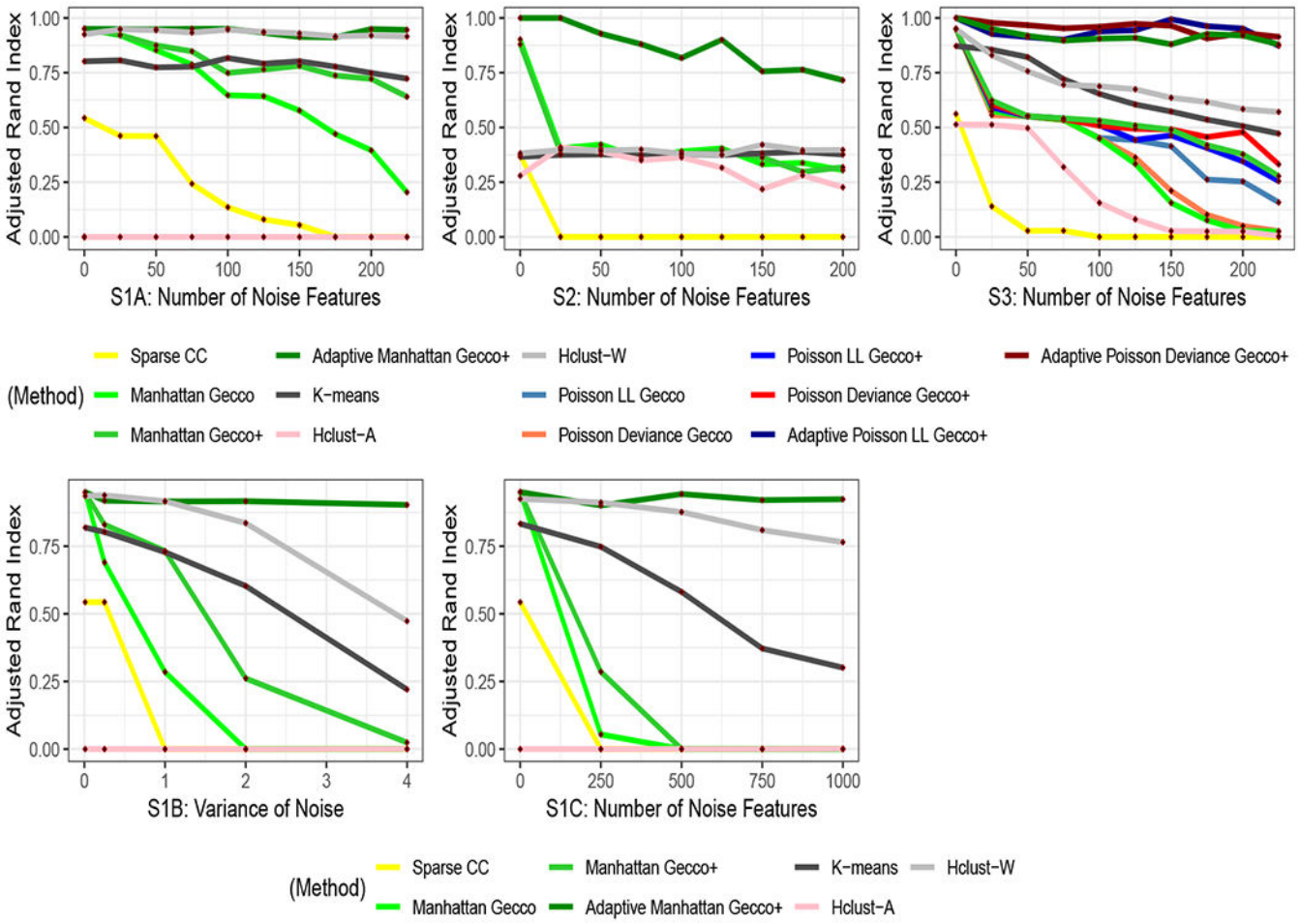

Figure 3:

Simulation results of non-Gaussian data: (S1A) We increase number of noise features for spherical data with outliers; (S2) We increase number of noise features for non-spherical data with outliers; (S3) We increase number of noise features for count-valued data; (S1B) We increase noise level for spherical data with outliers; (S1C) We further increase number of noise features for spherical data with outliers in high dimensions. The adaptive Gecco+ outperforms existing methods in high dimensions.

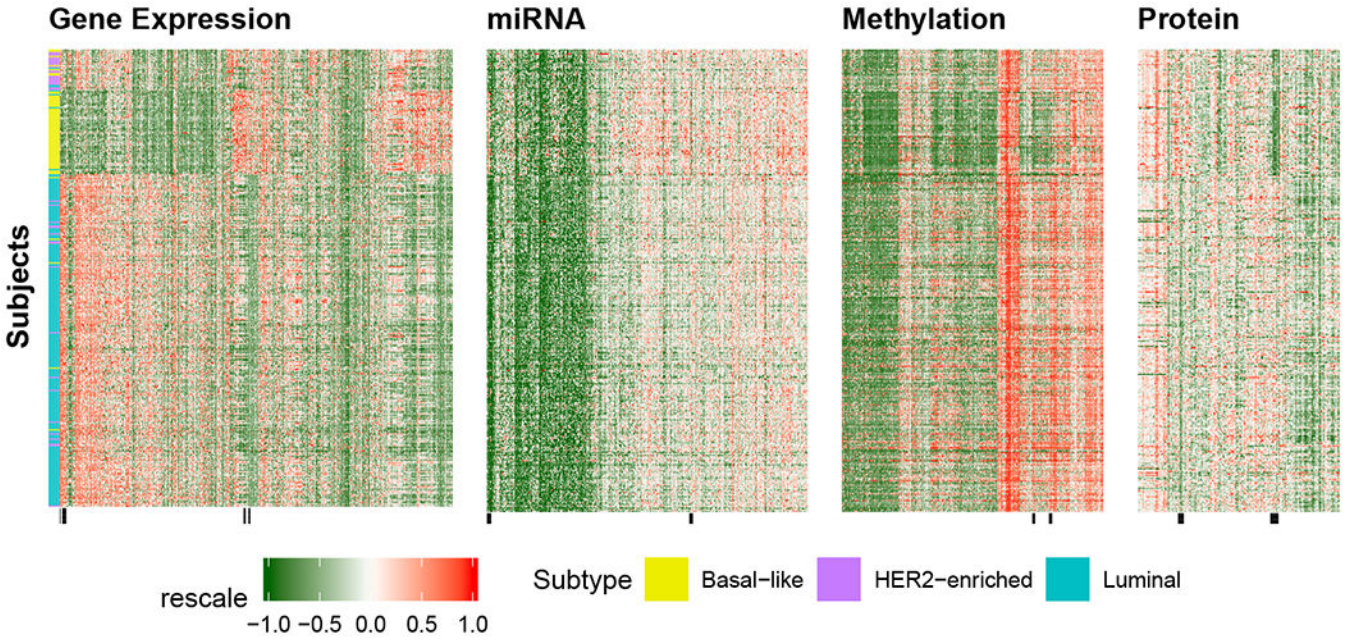

Figure 4:

Cluster heatmap of multi-omics TCGA data with row orders determined by cluster assignments from iGecco+. The left bar refers to the integrated cluster labels from biologists. The black bars at the bottom of each data view correspond to the selected features. Our adaptive iGecco+ identifies meaningful subtypes.

Table 1:

Different losses and their loss-specific centers. We provide all calculations associated with loss-specific centers in Appendix F. Note the Gecco problem with Hamming or Canberra distances is not convex. Though we discuss general convex losses in this paper, we list these non-convex losses for reference. For multinomial log-likelihood and multinomial deviance, we change Gecco formulation slightly to accommodate three indices; we provide a detailed formulation in Appendix E.

Data Type	Loss Type	Loss Function	Loss-specific Center $\tilde{\mathbf{x}}$
Continuous	Euclidean (ℓ_2)	$\frac{1}{2}\|\mathbf{x}_i - \mathbf{u}_i\|_2^2$	$\overline{\mathbf{x}}$
Skewed continuous	Manhattan (ℓ_1) Minkowski (ℓ_q) Mahalanobis (weighted ℓ_2) Chebychev (ℓ_∞) Canberra (weighted ℓ_1)	$\sum_{j=1}^{} \lvert x_{ij} - u_{ij}\rvert$ $\sqrt[q]{\sum_{j=1}^{} \lvert x_{ij} - u_{ij}\rvert^q}$ $(\mathbf{x}_i - \mathbf{u}_i)^T \mathbf{C}^{-1} (\mathbf{x}_i - \mathbf{u}_i)$ $\max_j \{\lvert x_{ij} - u_{ij}\rvert\}$ $\sum_{j=1}^{} \frac{\lvert x_{ij} - u_{ij}\rvert}{\lvert x_{ij}\rvert + \lvert u_{ij}\rvert}$	median($\mathbf{x}$) no closed form no closed form no closed form no closed form
Binary	Bernoulli log-likelihood Binomial deviance Hinge loss KL divergence Hamming (ℓ_0)	$-x_{ij} u_{ij} + \log(1 + e^{u_{ij}})$ $-x_{ij} \log u_{ij} - (1 - x_{ij}) \log(1 - u_{ij})$ $\max(0, 1 - u_{ij} x_{ij})$ $-x_{ij} \log_2 u_{ij}$ $\sum_j \#(x_{ij} \neq u_{ij}) / n$	logit($\overline{\mathbf{x}}$) $\overline{\mathbf{x}}$ mode($\mathbf{x}$) no closed form mode ($\mathbf{x}$)
Count	Poisson log-likelihood Poisson deviance Negative binomial log-likelihood Negative binomial deviance Manhattan (ℓ_1) Canberra (weighted ℓ_1)	$-x_{ij} u_{ij} + \exp(u_{ij})$ $-x_{ij} \log u_{ij} + u_{ij}$ $-x_{ij} u_{ij} + (x_{ij} + \frac{1}{\alpha}) \log(\frac{1}{\alpha} + e^{u_{ij}})$ $x_{ij} \log(\frac{x_{ij}}{u_{ij}}) - (x_{ij} + \frac{1}{\alpha}) \log(\frac{1 + \alpha x_{ij}}{1 + \alpha u_{ij}})$ $\sum_{j=1}^{} \lvert x_{ij} - u_{ij}\rvert$ $\sum_{j=1}^{} \frac{\lvert x_{ij} - u_{ij}\rvert}{\lvert x_{ij}\rvert + \lvert u_i\rvert}$	log($\overline{\mathbf{x}}$) $\overline{\mathbf{x}}$ log($\overline{\mathbf{x}}$) $\overline{\mathbf{x}}$ median($\mathbf{x}$) no closed form

Data Type	Loss Type	Loss Function	Loss-specific Center $\tilde{\mathbf{x}}$
Categorical	Multinomial log-likelihood Multinomial deviance	$\left\{\sum_{k=1}^{K} -x_{ijk}u_{ijk} + \log(\sum_{k=1}^{K} e^{u_{ijk}})\right\}$ $\left\{\sum_{k=1}^{K} -x_{ijk} + \log(u_{ijk})\right\}, \sum_{k=1}^{K} u_{ijk} = 1$	mlogit($\bar{\mathbf{x}}$) $\bar{\mathbf{x}}$

Table 2:

Comparisons of F_1 score for adaptive Gecco+ and sparse convex clustering

Method	Scenario 1 (A)	Scenario 2	Scenario 3
Sparse Convex Clustering	0.37 (3.1e-2)	0.25 (2.4e-2)	0.14 (7.2e-3)
Adaptive Gecco+	0.97 (1.9e-2)	0.99 (1.0e-2)	0.81 (8.0e-2)

Table 3:

Comparisons of adjusted Rand index for mixed multi-view data

Method	Scenario 1	Scenario 2
Hclust: $\mathbf{X}_1$	0.35 (2.9e-2)	0.53 (2.3e-2)
Hclust: $\mathbf{X}_2$	0.53 (4.6e-2)	0.65 (1.8e-2)
Hclust: $\mathbf{X}_3$	0.52 (2.2e-2)	0.70 (2.4e-2)
Hclust: $[\mathbf{X}_1\mathbf{X}_2\mathbf{X}_3]$ - Euclidean	0.68 (4.7e-2)	0.63 (3.3e-2)
Hclust: $[\mathbf{X}_1\mathbf{X}_2\mathbf{X}_3]$ - Gower	0.86 (1.5e-2)	0.83 (7.3e-2)
iCluster+ with $\lambda = 0$	0.90 (1.6e-2)	0.71 (1.6e-2)
Bayesian Consensus Clustering	**0.95 (1.2e-2)**	0.63 (1.1e-2)
iGecco	0.93 (5.0e-3)	**0.98 (2.2e-2)**

Table 4:

Comparisons of adjusted Rand index for high-dimensional mixed multi-view data

Method	Scenario 3	Scenario 4	Scenario 5	Scenario 6
Hclust: $\mathbf{X}_1$	0.42 (2.3e-2)	0.56 (2.5e-2)	0.43 (2.5e-2)	0.51 (2.7e-2)
Hclust: $\mathbf{X}_2$	0.23 (2.8e-2)	0.29 (3.4e-2)	0.51 (2.6e-2)	0.49 (2.1e-2)
Hclust: $\mathbf{X}_3$	0.25 (3.1e-2)	0.27 (3.1e-2)	0.55 (2.6e-2)	0.48 (1.9e-2)
Hclust: $[\mathbf{X}_1\mathbf{X}_2\mathbf{X}_3]$ - Euclidean	0.40 (3.7e-2)	0.57 (3.3e-2)	0.55 (2.5e-2)	0.52 (2.1e-2)
Hclust: $[\mathbf{X}_1\mathbf{X}_2\mathbf{X}_3]$ - Gower	0.68 (3.4e-2)	0.58 (6.3e-2)	0.58 (3.2e-2)	0.58 (3.0e-2)
iCluster+	0.57 (6.5e-2)	0.77 (2.7e-2)	0.61 (2.4e-2)	0.62 (1.6e-2)
Bayesian Consensus Clustering	0.35 (1.1e-1)	0.64 (1.0e-1)	0.59 (1.2e-2)	0.63 (6.6e-3)
iGecco	0.00 (6.7e-4)	0.06 (5.0e-2)	0.39 (4.5e-2)	0.23 (6.9e-2)
iGecco+	0.12 (3.3e-2)	0.16 (7.1e-2)	0.44 (3.6e-2)	0.39 (3.8e-2)
Adaptive iGecco+	**0.97 (7.8e-3)**	**0.99 (7.5e-3)**	**1.00 (0.0e-0)**	**1.00 (0.0e-0)**

Table 5:

Comparisons of F_1 score for adaptive iGecco+ and iClusterPlus

	Overall		Gaussian		Count		Binary	
	iCluster+	**A iGecco+**	**iCluster+**	**A iGecco+**	**iCluster+**	**A iGecco+**	**iCluster+**	**A iGecco+**
S3	0.81 (3.1e-2)	**0.94 (1.7e-2)**	0.84 (5.7e-2)	0.99 (6.3e-3)	0.73 (3.3e-2)	0.88 (3.5e-2)	0.85 (1.5e-2)	0.93 (2.1e-2)
S4	0.95 (9.9e-3)	**0.98 (1.3e-2)**	0.99 (6.7e-3)	0.99 (7.3e-3)	0.92 (1.3e-2)	0.97 (1.8e-2)	0.94 (1.9e-2)	0.97 (1.8e-2)
S5	0.94 (3.5e-2)	**1.00 (0.0e-0)**	0.95 (3.3e-2)	1.00 (0.0e-0)	0.91 (4.2e-2)	1.00 (0.0e-0)	0.95 (3.3e-2)	1.00 (0.0e-0)
S6	0.92 (3.3e-2)	**1.00 (3.3e-3)**	0.97 (2.1e-2)	1.00 (0.0e-0)	0.84 (4.5e-2)	0.99 (1.0e-2)	0.95 (3.3e-2)	1.00 (0.0e-0)

Table 6:

Adjusted Rand index of different methods for authors data set

Method	Adjusted Rand Index
K-means	0.74
Hierarchical Clustering	0.73
Sparse Convex Clustering	0.50
Manhattan Gecco+	0.96
Poisson LL Gecco+	0.96
Poisson Deviance Gecco+	0.96

Table 7:

Features selected by different Gecco+ methods for authors data set

Method	Features
Manhattan Gecco+	“be” ,“had” ,“her”, “the” ,“to”, “was”
Poisson LL Gecco+	“an” , “her” , “our”, “your”
Poisson Deviance Gecco+	“an”, “be” , “had”, “her”, “is”, “my” , “the”, “was”

Table 8:

Adjusted Rand index of different methods for TCGA data set

Method	Adjusted Rand Index
K-means	0.40
Hierarchical Clustering	0.37
Sparse Convex Clustering	0.01
Manhattan Gecco+	0.76
Poisson LL Gecco+	0.72
Poisson Deviance Gecco+	0.72

Table 9:

Features selected by different Gecco+ methods for TCGA data set

Method	Features
Manhattan Gecco+	“BCL2” , “ERBB2” ,“GATA3” “HMGA1”, “IL6ST”
Poisson LL Gecco+	“ERBB2” “FOXA1” “GATA3”
Poisson Deviance Gecco+	“ERBB2” , “FOXA1”, “GATA3” “RET”, “SLC34A2”

Table 10:

Adjusted Rand index of different methods for multi-omics TCGA data set

Method	Adjusted Rand Index
Hclust: $\mathbf{X}_1$ GE	0.51
Hclust: $\mathbf{X}_2$ Meth	0.39
Hclust: $\mathbf{X}_3$ miRNA	0.21
Hclust: $\mathbf{X}_4$ Protein	0.24
Hclust: $[\mathbf{X}_1\mathbf{X}_2\mathbf{X}_3\mathbf{X}_4]$ - Euclidean	0.51
Hclust: $[\mathbf{X}_1\mathbf{X}_2\mathbf{X}_3\mathbf{X}_4]$ - Gower	0.40
iCluster+	0.36
Bayesian Consensus Clustering	0.35
Adaptive iGecco+	**0.71**

Table 11:

Features selected by adaptive iGecco+ methods for multi-omics TCGA data set

Data view	Features
Gene Expression	"AGR3", "FOXA1", "AGR2", "ROPN1", "ROPN1B", "ESR1", "C1orf64", "ART3","FSIP1"
miRNA	"hsa-mir-135b", "hsa-mir-190b", "hsa-mir-577", "hsa-mir-934"
Methylation	"cg08047457", "cg08097882", "cg00117172", "cg12265829"
Protein	"ER.alpha", "GATA3", "AR", "CyclimE1"