SPLITTING METHODS FOR CONVEX CLUSTERING

ERIC C. CHI[†] AND KENNETH LANGE[‡]

Abstract. Clustering is a fundamental problem in many scientific applications. Standard methods such as k-means, Gaussian mixture models, and hierarchical clustering, however, are beset by local minima, which are sometimes drastically suboptimal. Recently introduced convex relaxations of k-means and hierarchical clustering shrink cluster centroids toward one another and ensure a unique global minimizer. In this work we present two splitting methods for solving the convex clustering problem. The first is an instance of the alternating direction method of multipliers (ADMM); the second is an instance of the alternating minimization algorithm (AMA). In contrast to previously considered algorithms, our ADMM and AMA formulations provide simple and unified frameworks for solving the convex clustering problem under the previously studied ℓ_1, ℓ_2, and ℓ_∞ norms and open the door to potentially novel norms. We demonstrate the performance of our algorithm on both simulated and real data examples.

Key words. Convex optimization, Regularization paths, Alternating minimization algorithm, Alternating direction method of multipliers, Hierarchical clustering, k-means

AMS subject classifications. 62H30, 90C25, 90C90

1. Introduction.

In recent years convex relaxations of many fundamental, yet combinatorially hard, optimization problems in engineering, applied mathematics, and statistics have been introduced [62]. Good, and sometimes nearly optimal solutions, can be achieved at affordable computational prices for problems that appear at first blush to be computationally intractable. In this paper we introduce two new algorithmic frameworks based on variable splitting that generalize and extend recent efforts to convexify the classic unsupervised problem of clustering.

Lindsten et al. [40] and Hocking et al. [34] formulate the clustering task as a convex optimization problem. Given p points $\mathbf{x}_1, \ldots, \mathbf{x}_p$ in $\mathbb{R}^q$, they suggest minimizing the convex criterion

$$F_\gamma(\mathbf{U}) = \frac{1}{2}\sum_{i=1}^{p} \|\mathbf{x}_i - \mathbf{u}_i\|_2^2 + \gamma \sum_{i<j} w_{ij}\|\mathbf{u}_i - \mathbf{u}_j\|, \tag{1.1}$$

where γ is a positive tuning constant, w_{ij} is a nonnegative weight, and the ith column $\mathbf{u}_i$ of the matrix $\mathbf{U}$ is the cluster center attached to point $\mathbf{x}_i$. Lindsten et al. [40] consider an ℓ_p norm penalty on the differences $\mathbf{u}_i - \mathbf{u}_j$ while Hocking et al. [34] consider ℓ_1, ℓ_2, and ℓ_∞ penalties. In the current paper an arbitrary norm defines the penalty.

The objective function bears some similarity to the fused lasso signal approximator [59]. In fact, the graphical fused lasso as described in [60] is a special case of the problem studied here. In the graphical interpretation of clustering, each point corresponds to a node in a graph, and an edge connects nodes i and j whenever $w_{ij} > 0$. Figure 1.1 depicts an example. In this case the objective function $F_\gamma(\mathbf{U})$ separates over the connected components of the underlying graph. Thus, one can solve for the optimal $\mathbf{U}$ component by component. Without loss of generality, we assume the graph is connected.

[†] Dept. Human Genetics, University of California, Los Angeles, CA. Email: ecchi@ucla.edu

[‡] Depts of Biomathematics, Human Genetics, and Statistics, University of California, Los Angeles, CA. Email: klange@ucla.edu

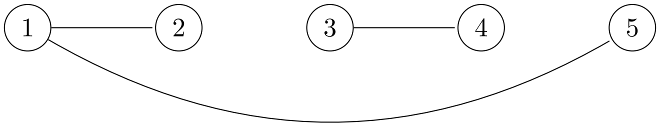


Fig. 1.1: A graph with positive weights w_{12}, w_{15}, w_{34} and all other weights $w_{ij} = 0$.

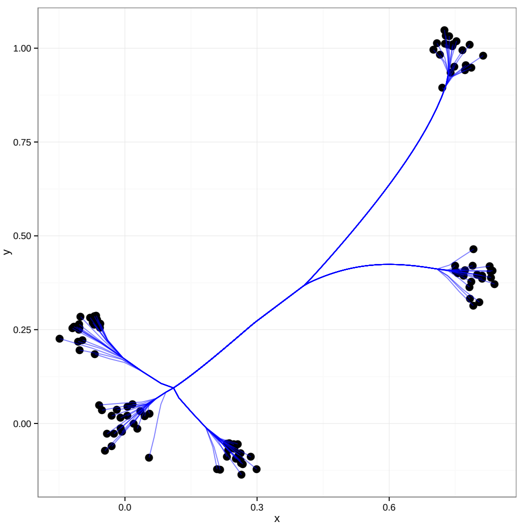


Fig. 1.2: Cluster path assignment: The simulated example shows five well separated clusters and the assigned clustering from applying the convex clustering algorithm using an ℓ_2-norm. The blue lines trace the path of the individual cluster centers as the regularization parameter γ increases.

When $\gamma = 0$, the minimum is attained when $\mathbf{u}_i = \mathbf{x}_i$, and each point occupies a unique cluster. As γ increases, the cluster centers begin to coalesce. Two points $\mathbf{x}_i$ and $\mathbf{x}_j$ with $\mathbf{u}_i = \mathbf{u}_j$ are said to belong to the same cluster. For sufficiently high γ all points coalesce into a single cluster. Because the objective function $F_\gamma(\mathbf{U})$ in equation (1.1) is strictly convex and coercive, it possesses a unique minimum point for each value of γ. If we plot the solution matrix $\mathbf{U}$ as a function of γ, then we can ordinarily identify those values of γ giving k clusters for any integer k between p and 1. In theory k can decrement by more than 1 as certain critical values of γ are passed. Indeed, when points are not well separated, we observe that many centroids will coalesce abruptly unless care is taken in choosing the weights w_{ij}.

The benefits of this formulation are manifold. As we will show, convex relaxation admits a simple and fast iterative algorithm that is guaranteed to converge to the unique global minimizer. In contrast, the classic k-means problem has been shown to be NP-hard [1, 12]. In addition, the classical greedy algorithm for solving k-means clustering often gets stuck in suboptimal local minima [20, 41, 43].

Another vexing issue in clustering is determining the number of clusters. Agglomerative hierarchical clustering [30, 36, 39, 49, 65] finesses the problem by computing an entire clustering path. Agglomerative approaches, however, can be computationally demanding and also tend to fall into suboptimal local minima since coalescence events are not reversed. The alternative convex relaxation considered here performs continuous clustering just as the lasso [9, 58] performs continuous variable selection. Figure 1.2 shows how the solutions to the alternative convex problem traces out an intuitively appealing, globally optimal, and computationally tractable solution path.

1.1. Contributions. Our main contributions are two new methods for solving the convex relaxation. Relatively little work has been published on algorithms for solving this optimization problem. Lindsten et al. [40] used an off-the-shelf convex solver, CVX [11, 31], to generate solution paths. Hocking et al. [34] introduced three distinct algorithms for the three most commonly encountered norms. Given the ℓ_1 norm and unit weights w_{ij}, the objective function separates, and they solve the convex clustering problem by the exact path following method designed for the fused lasso [35]. For the ℓ_1 and ℓ_2 norms with arbitrary weights w_{ij}, they employ subgradient descent in conjunction with active sets. Finally, they solved the convex clustering problem under the ℓ_∞ norm by viewing it as minimization of a Frobenius norm over a polytope. In this guise the problem succumbs to the Frank-Wolfe algorithm [22] of quadratic programming.

In contrast to this piecemeal approach, we introduce two similar generic frameworks for minimizing the convex clustering objective function with an arbitrary norm. One approach solves the problem by the alternating direction method of multipliers (ADMM), while the other solves it by the alternating minimization algorithm (AMA). The key step in both cases requires computing the proximal map of a given norm. Consequently, both of our algorithms apply provided the penalty norm admits efficient computation of its proximal map. Both algorithms are also amenable to acceleration.

In addition to introducing new algorithms for solving the convex clustering problem, the current paper contributes in other concrete ways: (a) We characterize both of the new algorithms theoretically. This entails proving the convergence of their iterates and explicitly showing how their computational complexity scales with the connectivity of the underlying graph. (b) We provide new proofs of intuitive properties of the solution path. These results are tied solely to the minimization of the objective function (1.1) and hold regardless of the algorithm used to find the minimum point. (c) We suggest stopping rules based on suboptimality conditions. (d) We provide guidance on how to choose the weights w_{ij}. Our suggested choices diminish computational complexity and enhance solution quality.

1.2. Related Work. The literature on clustering is immense; the reader can consult the books [29, 32, 38, 48, 67] for a comprehensive review. The clustering function (1.1) can be viewed as a convex relaxation of either k-means clustering [40] or hierarchical agglomerative clustering [34]. Both of these classical clustering methods [56, 57, 65] come in several varieties. The literature on k-means clustering reports notable improvements in the computation [17] and quality of solutions [4, 8, 38] delivered by the standard greedy algorithms. Faster methods for agglomerative hierarchical clustering have been developed as well [21]. Many statisticians view the hard cluster assignments of k-means as less desirable than the probabilistic assignments generated by mixture models [45, 61]. Mixture models have the advantage of gracefully assigning points to overlapping clusters. These models are amenable to the EM algorithm and can be extended infinite mixtures [18, 53, 50].

Alternative approaches to clustering involve identifying components in the associated graph via its Laplacian matrix. Spectral clustering [42] can be effective in cases when the clusters are non-convex and linearly inseparable. Although spectral clustering is valuable, it is not in conflict with convex relaxation. Indeed, Hocking et al [34] argue that convex clustering can be effectively merged with spectral clustering. Although we agree with this point, the solution path uncovered by convex clustering is meritorious in its own right because it partially obviates the persistent need for determining the number of clusters.

1.3. Notation. Throughout, scalars are denoted by lowercase letters (a), vectors by boldface lowercase letters ($\mathbf{u}$), and matrices by boldface capital letters ($\mathbf{U}$). The jth column of a matrix $\mathbf{U}$ is denoted by $\mathbf{u}_j$. At times in our derivations, it will be easier to work with the vectorization of matrices. We adopt the convention of denoting the vectorization of a matrix ($\mathbf{U}$) by its lower case letter in boldface ($\mathbf{u}$). Finally, we denote sets by upper case letters (B).

1.4. Organization. The rest of the paper is organized as follows. We first characterize the solution path theoretically. Previous papers take intuitive properties of the path for granted. We then review the ADMM and AMA algorithms and adapt them to solve the convex clustering problem. Once the algorithms are specified, we discuss their computational complexity, convergence, and acceleration. The paper concludes with a few numerical examples and a general discussion.

2. Properties of the solution path. We prove that the solution path $\mathbf{U}(\gamma, \mathbf{w})$, as a function of the regularization parameter γ and its weights $\mathbf{w} = \{w_{ij}\}$, has several nice properties that expedite its numerical computation.

Proposition 2.1. *The solution path $\mathbf{U}(\gamma)$ exists and depends continuously on γ. The path also depends continuously on the weight matrix $\mathbf{w}$.*

Proof. The existence and uniqueness of $\mathbf{U}(\gamma)$ are immediate consequences of the coerciveness and strict convexity of $F_\gamma(\mathbf{U})$. To prove continuity in γ, consider a subinterval $[a, b]$ and fix an arbitrary point $\mathbf{U}$. For any $\gamma \in [a, b]$ the inequalities

$$F_a[\mathbf{U}(\gamma)] \leq F_\gamma[\mathbf{U}(\gamma)] \quad \leq \quad F_\gamma(\mathbf{U}) \quad \leq \quad F_b(\mathbf{U})$$

hold because $F_\gamma(\mathbf{U})$ is non-decreasing in γ and $\mathbf{U}(\gamma)$ minimizes $F_\gamma(\mathbf{U})$. Since $F_a(\mathbf{U})$ is coercive, $\mathbf{U}(\gamma)$ is bounded for all $\gamma \in [a, b]$. Suppose that $\mathbf{U}(\gamma)$ fails to be continuous at γ. Then there is an $\epsilon > 0$ and a sequence γ_k tending to γ such that $\|\mathbf{U}(\gamma_k) - \mathbf{U}(\gamma)\| \geq \epsilon$ for all k. Because the $\mathbf{U}(\gamma_k)$ are bounded, we can pass to a subsequence if necessary and assume that $\mathbf{U}(\gamma_k)$ converges to a point $\tilde{\mathbf{U}}$. Because $F_\gamma(\mathbf{U})$ is continuous in $(\gamma, \mathbf{U})$, taking limits in the inequality $F_{\gamma_k}[\mathbf{U}(\gamma_k)] \leq F_{\gamma_k}(\mathbf{U})$ shows that $F_\gamma(\tilde{\mathbf{U}}) \leq F_\gamma(\mathbf{U})$ for arbitrary $\mathbf{U}$. Since $\mathbf{U}(\gamma)$ is uniquely determined, this implies that $\tilde{\mathbf{U}} = \mathbf{U}(\gamma)$, which obviously contradicts the assumption that $\|\mathbf{U}(\gamma_k) - \mathbf{U}(\gamma)\| \geq \epsilon$ for all k. The proof of the continuity of $\mathbf{U}$ in $\mathbf{w}$ proceeds along similar lines but is notationally more cumbersome. Details are left to the reader. □

Existence and uniqueness of $\mathbf{U}$ sets the stage for a well-posed optimization problem. Continuity of $\mathbf{U}$ suggests employing homotopy continuation. Indeed, empirically we find great time savings in solving a sequence of problems over a grid of γ values when we use the solution of a previous value of γ as a warm start or initial value for the next larger γ value.

We next prove that the centroids eventually coalesce to a common point. For the example shown in Figure 1.1, we intuitively expect for sufficiently large γ that the columns of $\mathbf{U}$ satisfy $\mathbf{u}_3 = \mathbf{u}_4 = \bar{\mathbf{x}}_{34}$ and $\mathbf{u}_1 = \mathbf{u}_2 = \mathbf{u}_5 = \bar{\mathbf{x}}_{125}$, where $\bar{\mathbf{x}}_{34}$ is the mean of $\mathbf{x}_3$ and $\mathbf{x}_4$ and $\bar{\mathbf{x}}_{125}$ is the mean of $\mathbf{x}_1$, $\mathbf{x}_2$, and $\mathbf{x}_5$. The next proposition confirms our intuition.

Proposition 2.2. *Suppose each point corresponds to a node in a graph with an edge between nodes i and j whenever $w_{ij} > 0$. If this graph is connected, then $F_\gamma(\mathbf{U})$ is minimized by $\bar{\mathbf{X}}$ for γ sufficiently large, where each column of $\bar{\mathbf{X}}$ equals the average $\bar{\mathbf{x}}$ of the n vectors $\mathbf{x}_i$.*

Proof. A point $\mathbf{X}$ furnishes a global minimum of the convex function $F_\gamma(\mathbf{X})$ if and only if all forward directional derivatives $d_{\boldsymbol{\Theta}} F_\gamma(\mathbf{X})$ at $\mathbf{X}$ are nonnegative. At the average matrix $\bar{\mathbf{X}}$ a brief calculation demonstrates that

$$d_{\boldsymbol{\Theta}} F_\gamma(\bar{\mathbf{X}}) = \sum_{i=1}^{p} \langle \bar{\mathbf{x}} - \mathbf{x}_i, \boldsymbol{\theta}_i \rangle + \gamma \sum_{i<j} w_{ij} \|\boldsymbol{\theta}_i - \boldsymbol{\theta}_j\|. \tag{2.1}$$

To construct a lower bound on this quantity, note that

$$\sum_{i=1}^{p} \langle \bar{\mathbf{x}} - \mathbf{x}_i, \boldsymbol{\theta}_j \rangle = 0$$

for all j. It follows that

$$\sum_{j=1}^{p} \sum_{i=1}^{p} \langle \bar{\mathbf{x}} - \mathbf{x}_i, \boldsymbol{\theta}_j \rangle = 0.$$

The generalized Cauchy-Schwartz inequality therefore implies

$$\begin{aligned}
\sum_{i=1}^{p} \langle \bar{\mathbf{x}} - \mathbf{x}_i, \boldsymbol{\theta}_i \rangle &= \frac{1}{p} \sum_{j=1}^{p} \sum_{i=1}^{p} \langle \bar{\mathbf{x}} - \mathbf{x}_i, \boldsymbol{\theta}_i \rangle \\
&= \frac{1}{p} \sum_{i=1}^{p} \sum_{j=1}^{p} \langle \bar{\mathbf{x}} - \mathbf{x}_i, \boldsymbol{\theta}_i - \boldsymbol{\theta}_j \rangle \\
&= \frac{1}{p} \sum_{i=1}^{p} \sum_{j \neq i} \langle \bar{\mathbf{x}} - \mathbf{x}_i, \boldsymbol{\theta}_i - \boldsymbol{\theta}_j \rangle \\
&\geq -\frac{2}{p} \sum_{i<j} \|\bar{\mathbf{x}} - \mathbf{x}_i\|_\dagger \|\boldsymbol{\theta}_i - \boldsymbol{\theta}_j\|
\end{aligned}$$

for the norm $\|\cdot\|_\dagger$ dual to $\|\cdot\|$. In view of expression (2.1), it suffices to prove that γ can be taken so large that

$$\gamma \sum_{i<j} w_{ij} \|\boldsymbol{\theta}_i - \boldsymbol{\theta}_j\| \geq \frac{2}{p} \sum_{i<j} \|\bar{\mathbf{x}} - \mathbf{x}_i\|_\dagger \|\boldsymbol{\theta}_i - \boldsymbol{\theta}_j\| \tag{2.2}$$

for all $\boldsymbol{\Theta}$.

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

$$\frac{2}{p} \max_{ij} \frac{\|\bar{\mathbf{x}} - \mathbf{x}_i\|_\dagger}{w_{ij}}.$$

In general set

$$\beta = \frac{2}{p \min_{w_{ij}>0} w_{ij}} \max_i \|\bar{\mathbf{x}} - \mathbf{x}_i\|_\dagger.$$

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

$$\|\boldsymbol{\theta}_i - \boldsymbol{\theta}_j\| \leq \|\boldsymbol{\theta}_i - \boldsymbol{\theta}_k\| + \cdots + \|\boldsymbol{\theta}_l - \boldsymbol{\theta}_j\|$$

and that

$$\frac{2}{p}\|\bar{\mathbf{x}} - \mathbf{x}_i\|_\dagger \|\boldsymbol{\theta}_i - \boldsymbol{\theta}_j\| \leq \beta \sum_{k<l} w_{kl}\|\boldsymbol{\theta}_k - \boldsymbol{\theta}_l\|.$$

Hence,

$$\frac{2}{p}\sum_{i<j}\|\bar{\mathbf{x}} - \mathbf{x}_i\|_\dagger \|\boldsymbol{\theta}_i - \boldsymbol{\theta}_j\| \leq \binom{p}{2}\beta \sum_{k<l} w_{kl}\|\boldsymbol{\theta}_k - \boldsymbol{\theta}_l\|,$$

and any $\gamma \geq \binom{p}{2}\beta$ satisfies the forward directional derivative test. □

We close this section by noting that in general the clustering paths are not guaranteed to be agglomerative. In the special case of the ℓ_1 norm with uniform weights $w_{ij} = 1$, Hocking et al [34] prove that the path is agglomerative. In the same paper they give an ℓ_2 norm example where the centroids fuse and then unfuse as the regularization parameter increases.

3. Algorithms to Compute the Clustering Path.

Having characterized the solution path $\mathbf{U}(\gamma)$, we now tackle the task of computing it. We present two closely related optimization approaches: the alternating direction method of multipliers (ADMM) [6, 24, 25] and the alternating minimization algorithm (AMA) [63]. Both approaches employ variable splitting to handle the shrinkage penalties in the convex clustering criterion (1.1).

3.1. Reformulation of Convex Clustering.

Let us first recast the convex clustering problem as the equivalent constrained problem

$$\begin{aligned} &\text{minimize } \frac{1}{2}\sum_{i=1}^{p}\|\mathbf{x}_i - \mathbf{u}_i\|_2^2 + \gamma\sum_l w_l\|\mathbf{v}_l\| \\ &\text{subject to } \mathbf{u}_{l_1} - \mathbf{u}_{l_2} - \mathbf{v}_l = \mathbf{0}. \end{aligned} \tag{3.1}$$

Here we index a centroid pair by $l = (l_1, l_2)$ with $l_1 < l_2$ and introduce a new variable $\mathbf{v}_l = \mathbf{u}_{l_1} - \mathbf{u}_{l_2}$ to account for the difference between the two centroids. The purpose of variable splitting is to simplify the penalties. The Lagrangian

$$\mathcal{L}_0(\mathbf{U}, \mathbf{V}, \boldsymbol{\Lambda}) = \frac{1}{2}\sum_{i=1}^{p}\|\mathbf{x}_i - \mathbf{u}_i\|_2^2 + \sum_l \langle \boldsymbol{\lambda}_l, \mathbf{v}_l - \mathbf{u}_{l_1} + \mathbf{u}_{l_2}\rangle + \gamma\sum_l w_l\|\mathbf{v}_l\|$$

for problem (3.1) leads to the dual function

$$\begin{aligned} D_\gamma(\boldsymbol{\Lambda}) &= \inf_{\mathbf{U},\mathbf{V}} \mathcal{L}_0(\mathbf{U}, \mathbf{V}, \boldsymbol{\Lambda}) \\ &= -\frac{1}{2}\sum_{i=1}^{p}\|\boldsymbol{\Delta}_i\|_2^2 - \sum_l \langle \boldsymbol{\lambda}_l, \mathbf{x}_{l_1} - \mathbf{x}_{l_2}\rangle - \sum_l \delta_{C_l}(\boldsymbol{\lambda}_l), \end{aligned} \tag{3.2}$$

where

$$\boldsymbol{\Delta}_i = \sum_{l:l_1=i}\boldsymbol{\lambda}_l - \sum_{l:l_2=i}\boldsymbol{\lambda}_l.$$

The set C_l indexing the convex indicator function δ_{C_l} in formula (3.2) equals the dual ball $\{\boldsymbol{\lambda}_l : \|\boldsymbol{\lambda}_l\|_\dagger \le \gamma w_l\}$. We will derive the dual function in a moment.

Splitting methods such as ADMM and AMA have been successfully used to attack similar problems in image restoration [28]. ADMM and AMA are now motivated as variants of the augmented Lagrangian method (ALM) [33, 51, 52, 54]. Let us review how ALM approaches the constrained optimization problem

$$\begin{aligned} &\text{minimize } f(\mathbf{u}) + g(\mathbf{v}) \\ &\text{subject to } \mathbf{A}\mathbf{u} + \mathbf{B}\mathbf{v} = \mathbf{c}, \end{aligned} \tag{3.3}$$

which includes the constrained minimization problem (3.1) as a special case. ALM solves the equivalent problem

$$\begin{aligned} &\text{minimize } f(\mathbf{u}) + g(\mathbf{v}) + \frac{\nu}{2}\|\mathbf{c} - \mathbf{A}\mathbf{u} - \mathbf{B}\mathbf{v}\|_2^2, \\ &\text{subject to } \mathbf{A}\mathbf{u} + \mathbf{B}\mathbf{v} = \mathbf{c} \end{aligned} \tag{3.4}$$

by imposing a quadratic penalty on deviations from the feasible set. The two problems (3.3) and (3.4) are equivalent because their objective functions coincide for any point $(\mathbf{u}, \mathbf{v})$ satisfying the equality constraint. The Lagrangian for the ALM problem

$$\mathcal{L}_\nu(\mathbf{u}, \mathbf{v}, \boldsymbol{\lambda}) = f(\mathbf{u}) + g(\mathbf{v}) + \langle \boldsymbol{\lambda}, \mathbf{c} - \mathbf{A}\mathbf{u} - \mathbf{B}\mathbf{v} \rangle + \frac{\nu}{2}\|\mathbf{c} - \mathbf{A}\mathbf{u} - \mathbf{B}\mathbf{v}\|_2^2$$

invokes the dual variable $\boldsymbol{\lambda}$ as a vector of Lagrange multipliers. If $f(\mathbf{u})$ and $g(\mathbf{v})$ are convex and $\mathbf{A}$ and $\mathbf{B}$ have full column rank, then the objective (3.4) is strongly convex, and the dual problem reduces to the unconstrained maximization of a concave function with Lipschitz continuous gradient. The dual problem is therefore a candidate for gradient ascent. In fact, this is the strategy that ALM takes in the updates

$$\begin{aligned} (\mathbf{u}^{m+1}, \mathbf{v}^{m+1}) &= \underset{\mathbf{u},\mathbf{v}}{\arg\min}\, \mathcal{L}_\nu(\mathbf{u}, \mathbf{v}, \boldsymbol{\lambda}^m) \\ \boldsymbol{\lambda}^{m+1} &= \boldsymbol{\lambda}^m + \nu(\mathbf{c} - \mathbf{A}\mathbf{u}^{m+1} - \mathbf{B}\mathbf{v}^{m+1}). \end{aligned} \tag{3.5}$$

Unfortunately, the minimization of the augmented Lagrangian over $\mathbf{u}$ and $\mathbf{v}$ jointly is often difficult. ADMM and AMA adopt different strategies in simplifying the minimization subproblem in the ALM updates (3.5). ADMM minimizes the augmented Lagrangian one block of variables at a time. This yields the algorithm

$$\begin{aligned} \mathbf{u}^{m+1} &= \underset{\mathbf{u}}{\arg\min}\, \mathcal{L}_\nu(\mathbf{u}, \mathbf{v}^m, \boldsymbol{\lambda}^m) \\ \mathbf{v}^{m+1} &= \underset{\mathbf{v}}{\arg\min}\, \mathcal{L}_\nu(\mathbf{u}^{m+1}, \mathbf{v}, \boldsymbol{\lambda}^m) \\ \boldsymbol{\lambda}^{m+1} &= \boldsymbol{\lambda}^m + \nu(\mathbf{c} - \mathbf{A}\mathbf{u}^{m+1} - \mathbf{B}\mathbf{v}^{m+1}). \end{aligned} \tag{3.6}$$

AMA takes a slightly different tack and updates the first block $\mathbf{u}$ without augmentation, assuming $f(\mathbf{u})$ is strongly convex. This change is accomplished by setting the positive tuning constant ν to be 0. The overall algorithm iterates according to

$$\begin{aligned} \mathbf{u}^{m+1} &= \underset{\mathbf{u}}{\arg\min}\, \mathcal{L}_0(\mathbf{u}, \mathbf{v}^m, \boldsymbol{\lambda}^m) \\ \mathbf{v}^{m+1} &= \underset{\mathbf{v}}{\arg\min}\, \mathcal{L}_\nu(\mathbf{u}^{m+1}, \mathbf{v}, \boldsymbol{\lambda}^m) \\ \boldsymbol{\lambda}^{m+1} &= \boldsymbol{\lambda}^m + \nu(\mathbf{c} - \mathbf{A}\mathbf{u}^{m+1} - \mathbf{B}\mathbf{v}^{m+1}). \end{aligned} \tag{3.7}$$

Although block descent appears to complicate matters, it often markedly simplifies optimization in the end. In the case of convex clustering, the updates are either simple linear transformations or proximal maps.

Before moving on to discuss specific incarnations of the splitting algorithms and their implementation via proximal maps, it is helpful to recall how one calculates the dual to problem (3.3). The Lagrangian of the problem is

$$\mathcal{L}_0(\mathbf{u},\mathbf{v},\boldsymbol{\lambda}) = f(\mathbf{u}) + g(\mathbf{v}) + \langle \boldsymbol{\lambda}, \mathbf{c} - \mathbf{A}\mathbf{u} - \mathbf{B}\mathbf{v}\rangle.$$

This translates into the dual function

$$\begin{aligned}
\mathcal{D}(\boldsymbol{\lambda}) &= \inf_{\mathbf{u},\mathbf{v}} \mathcal{L}_0(\mathbf{u},\mathbf{v},\boldsymbol{\lambda}) \\
&= \inf_{\mathbf{u}} \left\{ f(\mathbf{u}) - \langle \mathbf{A}^t\boldsymbol{\lambda}, \mathbf{u}\rangle \right\} + \inf_{\mathbf{v}} \left\{ g(\mathbf{v}) - \langle \mathbf{B}^t\boldsymbol{\lambda}, \mathbf{v}\rangle \right\} + \langle \boldsymbol{\lambda}, \mathbf{c}\rangle \\
&= -f^\star(\mathbf{A}^t\boldsymbol{\lambda}) - g^\star(\mathbf{B}^t\boldsymbol{\lambda}) + \langle \boldsymbol{\lambda}, \mathbf{c}\rangle \\
&= -f^\star(\mathbf{A}^t\boldsymbol{\lambda}) - \tilde{g}(\boldsymbol{\lambda}),
\end{aligned}$$

involving Fenchel conjugates. If $f(\mathbf{u})$ is strongly convex, then $f^\star(\mathbf{A}^t\boldsymbol{\lambda})$ has Lipschitz continuous gradient [37]. If $g(\mathbf{v})$ is a convex and lower semicontinuous function, then the proximal map of the partial dual $\tilde{g}(\boldsymbol{\lambda}) = g^\star(\mathbf{B}^t\boldsymbol{\lambda}) - \langle \boldsymbol{\lambda}, \mathbf{c}\rangle$ is well defined. Consequently, the dual problem is a prime candidate for solution by the proximal gradient algorithm [10]

$$\boldsymbol{\lambda}^{m+1} = \operatorname{prox}_{\nu\tilde{g}}[\boldsymbol{\lambda}^m - \nu\mathbf{A}\nabla f^\star(\mathbf{A}^t\boldsymbol{\lambda}^m)] \tag{3.8}$$

with sufficiently small step size ν. Despite appearances, the updates (3.7) are actually equivalent to the proximal gradient update (3.8). Proofs of this fact can be found in the papers [27, 63]. We present an alternative proof in Appendix A.

Calculation of the dual problem for convex clustering fits within this framework. The key ingredient is the Fenchel conjugate pair

$$f(\mathbf{u}) = \frac{1}{2}\|\mathbf{x} - \mathbf{u}\|_2^2 \quad \text{and} \quad f^\star(\mathbf{z}) = \frac{1}{2}\|\mathbf{z}\|_2^2 + \langle \mathbf{z}, \mathbf{x}\rangle.$$

In the convex clustering problem $\mathbf{B} = -\mathbf{I}_{q\varepsilon}$, where q is the number of features and ε is the number of edges in the incidence matrix generated by the positive weights w_{ij}. A little thought shows that $\mathbf{A}$ can be expressed via Kronecker products and the standard basis of $\mathbb{R}^q$ as $\mathbf{A}^t = (\mathbf{A}_1^t \cdots \mathbf{A}_\varepsilon^t)$ and $\mathbf{A}_l^t = [\mathbf{e}_{l_1} - \mathbf{e}_{l_2}] \otimes \mathbf{I}_q$. In this notation

$$\begin{aligned}
f^\star(\mathbf{A}^t\boldsymbol{\lambda}) &= \frac{1}{2}\|\mathbf{A}^t\boldsymbol{\lambda}\|_2^2 + \langle \mathbf{A}^t\boldsymbol{\lambda}, \mathbf{x}\rangle \\
&= \frac{1}{2}\| \sum_l \mathbf{A}_l^t\boldsymbol{\lambda}_l\|_2^2 + \langle \sum_l \mathbf{A}_l^t\boldsymbol{\lambda}_l, \mathbf{x}\rangle \\
&= \frac{1}{2}\| \sum_l [\mathbf{e}_{l_1} \otimes \boldsymbol{\lambda}_l - \mathbf{e}_{l_2} \otimes \boldsymbol{\lambda}_l]\,\|_2^2 + \sum_l \langle \boldsymbol{\lambda}_l, \mathbf{A}_l\mathbf{x}\rangle \\
&= \frac{1}{2}\sum_{i=1}^p \| \sum_{l_1=i} \boldsymbol{\lambda}_l - \sum_{l_2=i} \boldsymbol{\lambda}_l\|_2^2 + \sum_l \langle \boldsymbol{\lambda}_l, \mathbf{x}_{l_1} - \mathbf{x}_{l_2}\rangle.
\end{aligned}$$

The passage from the third expression to the fourth expression on the right of this string of equalities is justified by the observation

$$\sum_l [\mathbf{e}_{l_1} \otimes \boldsymbol{\lambda}_l - \mathbf{e}_{l_2} \otimes \boldsymbol{\lambda}_l] = \begin{pmatrix} \sum_{l_1=1} \boldsymbol{\lambda}_l - \sum_{l_2=1} \boldsymbol{\lambda}_l \\ \vdots \\ \sum_{l_1=p} \boldsymbol{\lambda}_l - \sum_{l_2=p} \boldsymbol{\lambda}_l \end{pmatrix}$$

and the appropriate grouping of sums of squares. The partial dual $\tilde{g}(\boldsymbol{\lambda}) = g^\star(\boldsymbol{\lambda})$ in the clustering context is

$$\begin{aligned} g^\star(\boldsymbol{\lambda}) &= \sup_{\mathbf{v}} \left[\langle \boldsymbol{\lambda}, \mathbf{v} \rangle - \gamma \sum_l w_l \|\mathbf{v}_l\| \right] \\ &= \sup_{\mathbf{v}} \left[\left\langle \sum_l \boldsymbol{\lambda}_l, \mathbf{v}_l \right\rangle - \gamma \sum_l w_l \|\mathbf{v}_l\| \right] \\ &= \sum_l \sup_{\mathbf{v}_l} \left[\langle \boldsymbol{\lambda}_l, \mathbf{v}_l \rangle - \gamma w_l \|\mathbf{v}_l\| \right] \\ &= \sum_l \delta_{C_l}(\boldsymbol{\lambda}_l), \end{aligned}$$

where $C_l = \{\mathbf{z} : \|\mathbf{z}\|_\dagger \leq \gamma w_l\}$.

3.2. Proximal Map. For $\sigma > 0$ the function

$$\operatorname{prox}_{\sigma\Omega}(\mathbf{u}) = \arg\min_{\mathbf{v}} \left[\sigma\Omega(\mathbf{v}) + \frac{1}{2}\|\mathbf{u} - \mathbf{v}\|_2^2 \right]$$

is a well-studied operation called the proximal map of the function $\Omega(\mathbf{v})$. The proximal map exists and is unique whenever the function $\Omega(\mathbf{v})$ is convex and lower semicontinuous. Norms satisfy these conditions, and for many norms of interest the proximal map can be evaluated by either an explicit formula or an efficient algorithm. Table 3.1 lists some common examples. The proximal maps for the ℓ_1 and ℓ_2 norms have explicit solutions and can be computed in $\mathcal{O}(q)$ operations for a vector $\mathbf{v} \in \mathbb{R}^q$. The proximal map for the ℓ_∞ norm requires projection onto the unit simplex and lacks an explicit solution. However, there are good algorithms for projecting onto the unit simplex [15, 46]. In particular, Duchi et al.'s projection algorithm makes it possible to evaluate $\operatorname{prox}_{\sigma\|\cdot\|_\infty}(\mathbf{v})$ in $\mathcal{O}(q \log q)$ operations. The $\ell_{1,2}$ norm

$$\|\mathbf{v}\|_{1,2} = \sum_{g \in \mathcal{G}} \|\mathbf{v}_g\|_2$$

listed in Table 3.1 partitions the components of $\mathbf{v}$ into non-overlapping groups $\mathcal{G}$. In this case there is a simple shrinkage formula. If the groups overlap, then Mairal et al. [44] have shown that the associated proximal map can computed exactly in polynomial time by solving a quadratic min-cost flow problem.

3.3. ADMM updates. The augmented Lagrangian is given by

$$\begin{aligned} \mathcal{L}_\nu(\mathbf{U}, \mathbf{V}, \boldsymbol{\Lambda}) = \frac{1}{2}\sum_{i=1}^p \|\mathbf{x}_i - \mathbf{u}_i\|_2^2 + \gamma \sum_l w_l \|\mathbf{v}_l\| \\ + \sum_l \langle \boldsymbol{\lambda}_l, \mathbf{v}_l - \mathbf{u}_{l_1} + \mathbf{u}_{l_2} \rangle + \frac{\nu}{2} \sum_l \|\mathbf{v}_l - \mathbf{u}_{l_1} + \mathbf{u}_{l_2}\|_2^2. \end{aligned} \tag{3.9}$$

Table 3.1: Proximal maps for common norms.

Norm	$\Omega(\mathbf{v})$	$\mathrm{prox}_{\sigma\Omega}(\mathbf{v})$	Comment
ℓ_1	$\lVert\mathbf{v}\rVert_1$	$\left[1 - \frac{\sigma}{\lvert v_l \rvert}\right]_+ v_l$	Element-wise soft-thresholding
ℓ_2	$\lVert\mathbf{v}\rVert_2$	$\left[1 - \frac{\sigma}{\lVert\mathbf{v}\rVert_2}\right]_+ \mathbf{v}$	Block-wise soft-thresholding
ℓ_∞	$\lVert\mathbf{v}\rVert_\infty$	$\mathbf{v} - \mathcal{P}_{\sigma S}(\mathbf{v})$	S is the unit simplex
$\ell_{1,2}$	$\sum_{g\in\mathcal{G}} \lVert\mathbf{v}_g\rVert_2$	$\left[1 - \frac{\sigma}{\lVert\mathbf{v}_g\rVert_2}\right]_+ \mathbf{v}_g$	$\mathcal{G}$ is a partition of $\{1,\ldots,p\}$

The gradient of $\mathcal{L}_\nu(\mathbf{U},\mathbf{V},\boldsymbol{\Lambda})$ with respect to $\mathbf{u}_i$ is

$$\begin{aligned}\frac{\partial}{\partial \mathbf{u}_i}\mathcal{L}_\nu(\mathbf{U},\mathbf{V},\boldsymbol{\Lambda}) = \mathbf{u}_i - \mathbf{x}_i - \sum_{l_1=i}\boldsymbol{\lambda}_l + \sum_{l_2=i}\boldsymbol{\lambda}_l - \nu\sum_{l_1=i}(\mathbf{v}_l - \mathbf{u}_i + \mathbf{u}_{l_2}) \\ +\nu\sum_{l_2=i}(\mathbf{v}_l - \mathbf{u}_{l_1} + \mathbf{u}_i).\end{aligned}$$

To update $\mathbf{U}$, we set $\frac{\partial}{\partial \mathbf{u}_i}\mathcal{L}_\nu(\mathbf{U},\mathbf{V},\boldsymbol{\Lambda}) = \mathbf{0}$ and rearrange. This gives the equation

$$(1+p\nu)\mathbf{u}_i = \mathbf{y}_i + \nu\sum_{j=1}^{p}\mathbf{u}_j, \tag{3.10}$$

where

$$\mathbf{y}_i = \mathbf{x}_i + \sum_{l_1=i}[\boldsymbol{\lambda}_l + \nu\mathbf{v}_l] - \sum_{l_2=i}[\boldsymbol{\lambda}_l + \nu\mathbf{v}_l].$$

When we sum the equalities $\frac{\partial}{\partial \mathbf{u}_i}\mathcal{L}_\nu(\mathbf{U},\mathbf{V},\boldsymbol{\Lambda}) = \mathbf{0}$ on i and exploit symmetric cancellations, we deduce the identity $\sum_{i=1}^{p}\mathbf{u}_i = \sum_{i=1}^{p}\mathbf{x}_i$. Equation (3.10) therefore implies that the columns $\mathbf{u}_i$ of the optimal matrix $\mathbf{U}$ satisfy

$$\mathbf{u}_i = \frac{1}{1+p\nu}\mathbf{y}_i + \frac{p\nu}{1+p\nu}\bar{\mathbf{x}}.$$

To update the $\mathbf{V}$, we first observe that the Lagrangian $\mathcal{L}_\nu(\mathbf{U},\mathbf{V},\boldsymbol{\Lambda})$ is separable in the vectors $\mathbf{v}_l$. A particular difference vector $\mathbf{v}_l$ is determined by the proximal map

$$\begin{aligned}\mathbf{v}_l &= \arg\min_{\mathbf{v}} \frac{1}{2}\lVert\mathbf{v} - (\mathbf{u}_{l_1} - \mathbf{u}_{l_2} - \nu^{-1}\boldsymbol{\lambda}_l)\rVert_2^2 + \frac{\gamma w_l}{\nu}\lVert\mathbf{v}\rVert \\ &= \mathrm{prox}_{\sigma_l\lVert\cdot\rVert/\nu}(\mathbf{u}_{l_1} - \mathbf{u}_{l_2} - \nu^{-1}\boldsymbol{\lambda}_l),\end{aligned} \tag{3.11}$$

where $\sigma_l = \gamma w_l$. Finally, the Lagrange multipliers are updated by

$$\boldsymbol{\lambda}_l = \boldsymbol{\lambda}_l + \nu(\mathbf{v}_l - \mathbf{u}_{l_1} + \mathbf{u}_{l_2}).$$

When $w_l = 0$, the updates for $\boldsymbol{\lambda}_l$ and $\mathbf{v}_l$ simplify to

$$\mathbf{v}_l = \mathbf{u}_{l_1} - \mathbf{u}_{l_2} \quad \text{and} \quad \boldsymbol{\lambda}_l = \mathbf{0}.$$

Algorithm 1 summarizes the updates.

Algorithm 1 ADMM

```
Initialize $\boldsymbol{\lambda}^0$ and $\mathbf{V}^0$.
for $m = 1, 2, 3, \ldots$ do
    for $i = 1, \ldots, p$ do
        $\mathbf{y}_i = \mathbf{x}_i + \sum_{l_1 = i}[\boldsymbol{\lambda}_l^{m-1} + \nu \mathbf{v}_l^{m-1}] - \sum_{l_2 = i}[\boldsymbol{\lambda}_l^{m-1} + \nu \mathbf{v}_l^{m-1}]$
    end for
    $\mathbf{U}^m = \frac{1}{1+p\nu}\mathbf{Y} + \frac{p\nu}{1+p\nu}\bar{\mathbf{X}}$
    for all $l$ do
        $\mathbf{v}_l^m = \text{prox}_{\sigma_l \|\cdot\|/\nu}(\mathbf{u}_{l_1}^m - \mathbf{u}_{l_2}^m - \nu^{-1}\boldsymbol{\lambda}_l^{m-1})$
        $\boldsymbol{\lambda}_l^m = \boldsymbol{\lambda}_l^{m-1} + \nu(\mathbf{v}_l^m - \mathbf{u}_{l_1}^m + \mathbf{u}_{l_2}^m)$
    end for
end for
```

3.3.1. Stopping Criterion for ADMM. To track the progress of ADMM we rely on computing the primal and dual residuals [6]. The necessary and sufficient conditions for $(\mathbf{u}^*, \mathbf{v}^*, \boldsymbol{\lambda}^*)$ to be an optimal solution to (3.3) are primal feasibility

$$\mathbf{A}\mathbf{u}^* + \mathbf{B}\mathbf{v}^* = \mathbf{c}, \tag{3.12}$$

and dual feasibility

$$\mathbf{0} \in \partial f(\mathbf{u}^*) + \mathbf{A}^t\boldsymbol{\lambda}^* \tag{3.13}$$
$$\mathbf{0} \in \partial g(\mathbf{v}^*) + \mathbf{B}^t\boldsymbol{\lambda}^*. \tag{3.14}$$

After the mth round of updates (3.6), $\mathbf{v}^m$ and $\boldsymbol{\lambda}^m$ satisfy condition (3.14). This suggests relying on conditions (3.12) and (3.13) to track the progress of ADMM. Inspection of the updates (3.6) reveals that $\mathbf{u}^{m+1}$ satisfies

$$\mathbf{0} \in \partial f(\mathbf{u}^{m+1}) + \mathbf{A}^t\boldsymbol{\lambda}^{m+1} + \nu\mathbf{A}^t\mathbf{B}(\mathbf{v}^m - \mathbf{v}^{m+1}),$$

or after rearrangement

$$\nu\mathbf{A}^t\mathbf{B}(\mathbf{v}^{m+1} - \mathbf{v}^m) \in \partial f(\mathbf{u}^{m+1}) + \mathbf{A}^t\boldsymbol{\lambda}^{m+1}.$$

The quantity on the left is referred to as the dual residual since it quantifies the deviation of the iterates from dual feasibility. In the convex clustering problem, there are p dual residual vectors

$$\mathbf{s}_i^{m+1} = -\nu\left(\sum_{l_1 = i}[\mathbf{v}_l^{m+1} - \mathbf{v}_l^m] - \sum_{l_2 = i}[\mathbf{v}_l^{m+1} - \mathbf{v}_l^m]\right).$$

The primal residuals are given by

$$\mathbf{r}_l^{m+1} = \mathbf{u}_{l_1}^{m+1} - \mathbf{u}_{l_2}^{m+1} - \mathbf{v}_l^{m+1}.$$

Let $\mathbf{S}^m$ and $\mathbf{R}^m$ denote the matrices with ith columns $\mathbf{s}_i^m$ and $\mathbf{r}_i^m$ respectively. If we wish to base our stopping on a measure of suboptimality, then following Frobenius norm inequality

$$\frac{1}{2}\|\mathbf{X} - \mathbf{U}^m\|_{\text{F}}^2 + \gamma\sum_l w_l\|\mathbf{v}_l^m\| - F_\gamma(\mathbf{U}^*) \le \|\boldsymbol{\Lambda}^m\|_{\text{F}}\|\mathbf{R}^m\|_{\text{F}} + \|\mathbf{U}^m - \mathbf{U}^*\|_{\text{F}}\|\mathbf{S}^m\|_{\text{F}}$$

comes in handy. The derivation of this bound can be found in appendix A of [6]. Of course, the term on the right cannot be computed since $\mathbf{U}^*$ is unknown. We will show later, however, that $(\mathbf{U}^m, \mathbf{V}^m)$ converges to $(\mathbf{U}^*, \mathbf{V}^*)$. Therefore, $\mathbf{S}^m$ converges to $\mathbf{0}$ and $\|\mathbf{U}^m - \mathbf{U}^*\|_{\text{F}} \leq \|\mathbf{X} - \overline{\mathbf{X}}\|_{\text{F}}$ for sufficiently large m. Thus, we can terminate when

$$\|\boldsymbol{\Lambda}^m\|_{\text{F}}\|\mathbf{R}^m\|_{\text{F}} + \|\mathbf{X} - \overline{\mathbf{X}}\|_{\text{F}}\|\mathbf{S}^m\|_{\text{F}} < \tau,$$

for some tolerance $\tau > 0$. This rule is engineered to track the suboptimality but does not gives rigorous guarantees, unlike the one we will see later for AMA.

3.4. AMA updates. Since AMA shares its update rules for $\mathbf{V}$ and $\boldsymbol{\Lambda}$ with ADMM, consider updating $\mathbf{U}$. Recall that AMA updates $\mathbf{U}$ by minimizing the ordinary Lagrangian ($\nu = 0$ case), namely

$$\mathbf{U}^{m+1} = \arg\min_{\mathbf{U}} \frac{1}{2}\sum_{i=1}^{p} \|\mathbf{x}_i - \mathbf{u}_i\|_2^2 + \sum_l \langle \boldsymbol{\lambda}_l^m, \mathbf{v}_l - \mathbf{u}_{l_1} + \mathbf{u}_{l_2}\rangle.$$

In contrast to ADMM, this minimization separates in each $\mathbf{u}_i$ and gives the much simpler update

$$\mathbf{u}_i^{m+1} = \mathbf{x}_i + \sum_{l_1 = i} \boldsymbol{\lambda}_l^m - \sum_{l_2 = i} \boldsymbol{\lambda}_l^m.$$

Further scrutiny of the updates for $\mathbf{V}$ and $\boldsymbol{\Lambda}$ reveals additional simplifications. Moreau's decomposition [10]

$$\mathbf{z} = \text{prox}_{th}(\mathbf{z}) + t\,\text{prox}_{t^{-1}h^\star}(t^{-1}\mathbf{z}).$$

allows one to express the proximal map of a function h in terms of the proximal map of its Fenchel conjugate $h^\star$. This decomposition generalizes the familiar orthogonal projection decomposition, namely $\mathbf{z} = \mathcal{P}_W(\mathbf{z}) + \mathcal{P}_{W^\perp}(\mathbf{z})$ where W is a closed Euclidean subspace and $W^\perp$ is its orthogonal complement. If $h(\mathbf{z}) = \|\mathbf{z}\|$ is a norm, then $h^\star(\mathbf{z}) = \delta_B(\mathbf{z})$ is the convex indicator function of the unit ball $B = \{\mathbf{y} : \|\mathbf{y}\|_\dagger \leq 1\}$ of the dual norm. Because the proximal map of the indicator function of a closed convex set collapses to projection onto the set, Moreau's decomposition leads to the identity

$$\text{prox}_{th}(\mathbf{z}) = \mathbf{z} - t\,\text{prox}_{t^{-1}\delta_B}(t^{-1}\mathbf{z}) = \mathbf{z} - t\mathcal{P}_B(t^{-1}\mathbf{z}) = \mathbf{z} - \mathcal{P}_{tB}(\mathbf{z}), \qquad (3.15)$$

where $\mathcal{P}_B(\mathbf{z})$ denotes projection onto B. In this derivation the identity $t^{-1}\delta_B = \delta_B$ holds because δ_B takes only the values 0 and ∞. Applying the projection formula (3.15) to the $\mathbf{v}_l$ update (3.11) yields the revised update

$$\mathbf{v}_l^{m+1} = \mathbf{u}_{l_1}^{m+1} - \mathbf{u}_{l_2}^{m+1} - \nu^{-1}\boldsymbol{\lambda}_l^m - \mathcal{P}_{tB}[\mathbf{u}_{l_1}^{m+1} - \mathbf{u}_{l_2}^{m+1} - \nu^{-1}\boldsymbol{\lambda}_l^m],$$

for the constant $t = \sigma_l/\nu = \gamma w_l/\nu$.

The update for $\boldsymbol{\lambda}_l$ is given by

$$\boldsymbol{\lambda}_l^{m+1} = \boldsymbol{\lambda}_l^m + \nu(\mathbf{v}_l^{m+1} - \mathbf{u}_{l_1}^{m+1} + \mathbf{u}_{l_2}^{m+1}).$$

Substituting for the above alternative expression for $\mathbf{v}_l^{m+1}$ leads to substantial cancellations and the revised formula

$$\boldsymbol{\lambda}_l^{m+1} = -\nu\mathcal{P}_{tB}[\mathbf{u}_{l_1}^{m+1} - \mathbf{u}_{l_2}^{m+1} - \nu^{-1}\boldsymbol{\lambda}_l^m].$$

Algorithm 2 AMA

```
Initialize $\boldsymbol{\lambda}^0$.
for $m = 1, 2, 3, \ldots$ do
    for $i = 1, \ldots, p$ do
        $\boldsymbol{\Delta}_i^m = \sum_{l_1 = i} \boldsymbol{\lambda}_l^{m-1} - \sum_{l_2 = i} \boldsymbol{\lambda}_l^{m-1}$
    end for
    for all $l$ do
        $\mathbf{g}_l^m = \mathbf{x}_{l_1} - \mathbf{x}_{l_2} + \boldsymbol{\Delta}_{l_1}^m - \boldsymbol{\Delta}_{l_2}^m$
        $\boldsymbol{\lambda}_l^m = \mathcal{P}_{C_l}(\boldsymbol{\lambda}_l^{m-1} - \nu \mathbf{g}_l^m)$
    end for
end for
```

The identities $-P_{tB}(\mathbf{z}) = \mathcal{P}_{tB}(-\mathbf{z})$ and $a\mathcal{P}_{tB}(\mathbf{z}) = \mathcal{P}_{atB}(a\mathbf{z})$ for $a > 0$ further simplify the update to

$$\boldsymbol{\lambda}_l^{m+1} = \mathcal{P}_{C_l}(\boldsymbol{\lambda}_l^m - \nu \mathbf{g}_l^{m+1}),$$

where $\mathbf{g}_l^m = \mathbf{u}_{l_1}^m - \mathbf{u}_{l_2}^m$ and $C_l = \{\boldsymbol{\lambda}_l : \|\boldsymbol{\lambda}_l\|_\dagger \leq \gamma w_l\}$. These updates are not surprising since, as noted earlier, the AMA algorithm is performing proximal gradient ascent on a dual problem. Algorithm 2 summarizes the AMA algorithm.

3.4.1. Stopping Criterion for AMA. The explicit functional forms (3.1) and (3.2) of the primal and dual functions make it trivial to evaluate the duality gap for feasible variables, since they depend on the quantities $\boldsymbol{\Delta}_i$ and $\mathbf{g}_l = \mathbf{u}_{l_1}^m - \mathbf{u}_{l_2}^m$, which are computed in the process of making the AMA updates. We can stop the AMA algorithm when the duality gap is small relative to the the optimal objective value. While we can define stopping rules that guarantee solutions within a tolerance on the suboptimality, in practice, we stop when

$$\frac{F_\gamma(\mathbf{U}^m) - D_\gamma(\boldsymbol{\Lambda}^m)}{1 + \frac{1}{2}[F_\gamma(\mathbf{U}^m) + D_\gamma(\boldsymbol{\Lambda}^m)]} < \tau$$

for $\tau > 0$ small. The average $\frac{1}{2}[F_\gamma(\mathbf{U}^m) + D_\gamma(\boldsymbol{\Lambda}^m)]$ serves as an estimate of $F_\gamma(\mathbf{U}^*)$.

4. Convergence.

Both ADMM and AMA converge under reasonable conditions.

4.1. AMA.

Tseng [63] provides sufficient conditions to ensure the convergence of AMA. In the following list of assumptions, the functions $f(\mathbf{u})$ and $g(\mathbf{v})$ and parameters $\mathbf{A}, \mathbf{B}$, and $\mathbf{c}$ refer to problem (3.3).

Assumption 4.1 (Assumptions B and C in [63]).

(a) $f(\mathbf{u})$ and $g(\mathbf{v})$ are convex lower-semicontinuous functions.

(b) $f(\mathbf{u})$ is strongly convex with modulus $\alpha > 0$.

(c) Problem (3.3) is feasible.

(d) The function $g(\mathbf{v}) + \|\mathbf{B}\mathbf{v}\|_2^2$ has a minimum.

(e) The dual of (3.3) has an optimal Lagrange multiplier corresponding to the constraint $\mathbf{A}\mathbf{u} + \mathbf{B}\mathbf{v} = \mathbf{c}$.

It is straightforward to verify that the functions and parameters in problem (3.3) satisfy Assumption 4.1. In particular, the strong convexity modulus $\alpha = 1$ for the convex clustering problem.

Proposition 4.2 (Proposition 2 in [63]). *Under Assumption 4.1 the iterates generated by the AMA updates (3.7) satisfy the following:*

(a) $\lim_{m\to\infty} \mathbf{u}^m = \mathbf{u}^*$,
(b) $\lim_{m\to\infty} \mathbf{B}\mathbf{v}^m = \mathbf{c} - \mathbf{A}\mathbf{u}^*$,
(c) $\lim_{m\to\infty} \boldsymbol{\lambda}^m = \boldsymbol{\lambda}^*$,
provided that $\nu < 2\alpha/\rho(\mathbf{A}^t\mathbf{A})$, *where* $\rho(\mathbf{A}^t\mathbf{A})$ *denotes the largest eigenvalue of* $\mathbf{A}^t\mathbf{A}$.

The parameter ν controlling the gradient step must be strictly less than twice the Lipschitz constant $1/\rho(\mathbf{A}^t\mathbf{A})$. To gain insight into how to choose ν, let $\varepsilon \leq \binom{p}{2}$ denote the number of edges. Then $\mathbf{A} = \mathbf{\Phi} \otimes \mathbf{I}$, where $\mathbf{\Phi}$ is the $\varepsilon \times p$ oriented edge-vertex incidence matrix

$$\mathbf{\Phi}_{lv} = \begin{cases} 1 & \text{If node } v \text{ is the head of edge } l \\ -1 & \text{If node } v \text{ is the tail of edge } l \\ 0 & \text{otherwise.} \end{cases}$$

As noted earlier, $\mathbf{A}^t\mathbf{A} = \mathbf{L} \otimes \mathbf{I}$, where $\mathbf{L} = \mathbf{\Phi}^t\mathbf{\Phi}$ is the Laplacian matrix of the associated graph. It is well known that the eigenvalues of $\mathbf{Z}\otimes\mathbf{I}$ coincide with the eigenvalues of $\mathbf{Z}$. See for example Theorem 6 in Chapter 9 of [47]. Therefore, $\rho(\mathbf{A}^t\mathbf{A}) = \rho(\mathbf{L})$. In general $\rho(\mathbf{L}) \leq p$ [3] with equality when the graph is fully connected, namely when $w_{ij} > 0$ for all $i < j$. Choosing a fixed step size of $\nu < 2/p$ certainly works in practice but may be too conservative. We adopt backtracking to achieve better overall performance. Details on our backtracking scheme are discussed in Section §7.3.

4.2. ADMM. Convergence of the ADMM algorithm has been proved under very modest assumptions, which we now restate.

Proposition 4.3. *If the functions* $f(\mathbf{x})$ *and* $g(\mathbf{x})$ *are closed, proper, and convex, and the unaugmented Lagrangian has a saddle point, then the ADMM iterates satisfy*

$$\lim_{m\to\infty} \mathbf{r}^m = \mathbf{0}$$
$$\lim_{m\to\infty} [f(\mathbf{U}^m) + g(\mathbf{V}^m)] = p^*$$
$$\lim_{m\to\infty} \boldsymbol{\lambda}^m = \boldsymbol{\lambda}^*,$$

where $\mathbf{r}^m = \mathbf{c} - \mathbf{A}\mathbf{u}^m - \mathbf{B}\mathbf{v}^m$ *denotes the primal residuals and* p^* *denotes the minimal objective value of the primal problem.*

Proofs of the above result can be found in the references [6, 16, 23]. Since the convex clustering criterion $F_\gamma(\mathbf{U})$ defined by equation (1.1) is strictly convex and coercive, we have the stronger result that the ADMM iterate sequence converges to the unique global minimizer $\mathbf{U}^*$ of $F_\gamma(\mathbf{U})$.

Proposition 4.4. *The iterates* $\mathbf{U}^m$ *in Algorithm 1 converge to the unique global minimizer* $\mathbf{U}^*$ *of the clustering criterion* $F_\gamma(\mathbf{U})$.

Proof. The conditions required by Proposition 4.3 are obviously met by $F_\gamma(\mathbf{U})$. In particular, the unaugmented Lagrangian possesses a saddle point since the primal problem has a global minimizer. To validate the conjectured limit, we first argue that the iterates $(\mathbf{U}^m, \mathbf{V}^m)$ are bounded. If on the contrary some subsequence is unbounded, then passing to the limit along this subsequence contradicts the limit

$$\lim_{m\to\infty} H_\gamma(\mathbf{U}^m, \mathbf{V}^m) = F_\gamma(\mathbf{U}^*) \tag{4.1}$$

guaranteed by Proposition 4.3 for the continuous function

$$H_\gamma(\mathbf{U}, \mathbf{V}) = \frac{1}{2}\sum_{i=1}^{p} \|\mathbf{x}_i - \mathbf{u}_i\|_2^2 + \gamma \sum_l w_l \|\mathbf{v}_l\|.$$

Algorithm 3 Fast AMA

```
Initialize $\boldsymbol{\lambda}^{-1} = \tilde{\boldsymbol{\lambda}}^{0}, \alpha_0 = 1$
for $m = 0, 1, 2, \ldots$ do
    for $i = 1, \ldots, p$ do
        $\boldsymbol{\Delta}_i^m = \sum_{l_1 = i} \boldsymbol{\lambda}_l^{m-1} - \sum_{l_2 = i} \boldsymbol{\lambda}_l^{m-1}$
    end for
    for all $l$ do
        $\mathbf{g}_l^m = \mathbf{x}_{l_1} - \mathbf{x}_{l_2} + \boldsymbol{\Delta}_{l_1}^m - \boldsymbol{\Delta}_{l_2}^m$
        $\tilde{\boldsymbol{\lambda}}_l^m = P_{C_l}(\boldsymbol{\lambda}_l^{m-1} - \nu \mathbf{g}_l^m)$
    end for
    $\alpha_m = (1 + \sqrt{1 + 4\alpha_{m-1}^2})/2$
    $\boldsymbol{\lambda}^{m+1} = \tilde{\boldsymbol{\lambda}}^m + \frac{\alpha_{m-1}}{\alpha_m}[\tilde{\boldsymbol{\lambda}}^m - \tilde{\boldsymbol{\lambda}}^{m-1}]$
end for
```

To prove convergence of the sequence $(\mathbf{U}^m, \mathbf{V}^m)$, it therefore suffices to check that every limit point coincides with the minimum point of $F_\gamma(\mathbf{U})$. Let $(\mathbf{U}^{m_n}, \mathbf{V}^{m_n})$ be a subsequence with limit $(\tilde{\mathbf{U}}, \tilde{\mathbf{V}})$. According to Proposition 4.3, the differences $\mathbf{u}_{l_1}^m - \mathbf{u}_{l_2}^m - \mathbf{v}_l^m$ tend to $\mathbf{0}$. Thus, the limit $(\tilde{\mathbf{U}}, \tilde{\mathbf{V}})$ is feasible. Furthermore,

$$\lim_{n \to \infty} H_\gamma(\mathbf{U}^{m_n}, \mathbf{V}^{m_n}) = H_\gamma(\tilde{\mathbf{U}}, \tilde{\mathbf{V}}) = F_\gamma(\tilde{\mathbf{U}}).$$

This limit contradicts the limit (4.1) unless $F_\gamma(\tilde{\mathbf{U}}) = F_\gamma(\mathbf{U}^*)$. Because $\mathbf{U}^*$ uniquely minimizes $F_\gamma(\mathbf{U})$, it follows that $\tilde{\mathbf{U}} = \mathbf{U}^*$. □

5. Acceleration.

Both AMA and ADMM admit acceleration at little additional computational cost. Given that AMA is a proximal gradient algorithm, Goldstein et al. [27] show that it can be effectively accelerated via Nesterov's method [5]. Algorithm 3 conveys the accelerated AMA method. Goldstein et al. [27] also present methods for accelerating ADMM. Algorithm 4 summarizes an accelerated version of ADMM found to work well in practice. As in Nesterov's method, updates are extrapolated. However, acceleration is restarted whenever the larger of the primal and dual residuals increases. Thus, this ADMM formulation forces the largest residual to decrease monotonically.

6. Computational Complexity.

6.1. AMA.

Suppose we wish to compute for a given γ a solution such that the duality gap is at most τ. We start by tallying the computational burden for a single round of AMA updates. Inspection of Algorithm 2 shows that computing all $\boldsymbol{\Delta}_i$ requires $q(2\varepsilon - p)$ total additions and subtractions. Computing all vectors $\mathbf{g}_l$ in Algorithm 2 takes $\mathcal{O}(\varepsilon q)$ operations, and taking the subsequent gradient step also costs $\mathcal{O}(\varepsilon q)$ operations. Computing the needed projections costs $\mathcal{O}(\varepsilon q)$ operations for the ℓ_1 and ℓ_2 norms and $\mathcal{O}(\varepsilon q \log q)$ operations for the ℓ_∞ norm. Finally computing the duality gap costs $\mathcal{O}(pq + \varepsilon q)$ operations. The assumption that p is $\mathcal{O}(\varepsilon)$ entails smaller costs. A single iteration with gap checking then costs just $\mathcal{O}(\varepsilon q)$ operations for the ℓ_1 and ℓ_2 norms and $\mathcal{O}(\varepsilon q \log q)$ operations for the ℓ_∞ norm.

Estimation of the number of iterations until convergence for proximal gradient descent and its Nesterov variant complete our analysis. The $pq \times \varepsilon q$ matrix $\mathbf{A}^t$ is typically short and fat. Consequently, the function $f^\star(\mathbf{A}^t \boldsymbol{\lambda})$ is not strongly convex.

Algorithm 4 Fast ADMM

```
Initialize $\tilde{\mathbf{V}}^{-1} = \mathbf{V}^0, \tilde{\boldsymbol{\lambda}}^{-1} = \boldsymbol{\lambda}^0, \alpha_0 = 1$
for $m = 1, 2, 3, \ldots$ do
    for $i = 1, \ldots, p$ do
        $\mathbf{y}_i = \mathbf{x}_i + \sum_{l_1 = i}[\boldsymbol{\lambda}_l^m + \nu \mathbf{v}_l^m] - \sum_{l_2 = i}[\boldsymbol{\lambda}_l^m + \nu \mathbf{v}_l^m]$
    end for
    $\mathbf{U}^m = \frac{1}{1+p\nu}\mathbf{Y} + \frac{p\nu}{1+p\nu}\bar{\mathbf{X}}$
    for all $l$ do
        $\tilde{\mathbf{v}}_l^m = \text{prox}_{\sigma_l \|\cdot\|/\nu}(\mathbf{u}_{l_1}^m - \mathbf{u}_{l_2}^m - \nu^{-1}\boldsymbol{\lambda}_l^m)$
        $\tilde{\boldsymbol{\lambda}}_l^m = \boldsymbol{\lambda}_l^m + \nu(\mathbf{v}_l^m - \mathbf{u}_{l_1}^m + \mathbf{u}_{l_2}^m)$
    end for
    if $\max(\|\mathbf{s}^{m-1}\|, \|\mathbf{r}^{m-1}\|) > \max(\|\mathbf{s}^m\|, \|\mathbf{r}^m\|)$ then
        $\alpha_{m+1} = (1 + \sqrt{1 + 4\alpha_m^2})/2$
        $\mathbf{V}^{m+1} = \tilde{\mathbf{V}}^m + \frac{\alpha_m - 1}{\alpha_{m+1}}[\tilde{\mathbf{V}}^m - \tilde{\mathbf{V}}^{m-1}]$
        $\boldsymbol{\lambda}^{m+1} = \tilde{\boldsymbol{\lambda}}^m + \frac{\alpha_m - 1}{\alpha_{m+1}}[\tilde{\boldsymbol{\lambda}}^m - \tilde{\boldsymbol{\lambda}}^{m-1}]$
    else
        $\alpha_{m+1} = 1, \mathbf{V}^{m+1} = \tilde{\mathbf{V}}^m, \boldsymbol{\lambda}^{m+1} = \tilde{\boldsymbol{\lambda}}^m.$
    end if
end for
```

Therefore, the best known convergence bounds for the proximal gradient method and its accelerated variant are sublinear [5]. Specifically we have the following non-asymptotic bounds on the convergence of the objective values:

$$D_\gamma(\boldsymbol{\lambda}^*) - D_\gamma(\boldsymbol{\lambda}^m) \leq \frac{\rho(\mathbf{A}^t\mathbf{A})\|\boldsymbol{\lambda}^* - \boldsymbol{\lambda}^0\|_2^2}{2m}$$

for the unaccelerated proximal gradient ascent and

$$D_\gamma(\boldsymbol{\lambda}^*) - D_\gamma(\boldsymbol{\lambda}^m) \leq \frac{2\rho(\mathbf{A}^t\mathbf{A})\|\boldsymbol{\lambda}^* - \boldsymbol{\lambda}^0\|_2^2}{(m+1)^2},$$

for its Nesterov accelerated alternative. Thus taking into account operations per iteration, we see that the unaccelerated version and acceleration algorithms respectively require a computational effort of $\mathcal{O}(\frac{\varepsilon q}{\tau})$ and $\mathcal{O}(\frac{\varepsilon q}{\sqrt{\tau}})$ respectively for the ℓ_1 and ℓ_2 norms to attain a duality gap less than τ. These bounds are respectively $\mathcal{O}(\frac{\varepsilon q \log q}{\tau})$ and $\mathcal{O}(\frac{\varepsilon q \log q)}{\sqrt{\tau}})$ for the ℓ_∞ norm. Total storage is $\mathcal{O}(q\varepsilon + qp)$. In the worst case ε is $\binom{p}{2}$. However, if we limit a node's connectivity to its k nearest neighbors, then ε is $\mathcal{O}(kp)$. Thus, the computational complexity of the problem in the worst case is quadratic in the number of points p and linear under the restriction to k-nearest neighbors connectivity. The storage is quadratic in p in the worst case and linear in p under the k-nearest neighbors restriction. Thus, limiting a point's connectivity to its k-nearest neighbors renders both the storage requirements and operation counts linear in the problem size, namely $\mathcal{O}(kqp)$.

6.2. ADMM. By nearly identical arguments, the complexity of a single round of ADMM updates with primal and dual residual calculation requires $\mathcal{O}(\varepsilon q)$ operations for the ℓ_1 and ℓ_2 norms and $\mathcal{O}(\varepsilon q \log q)$ operations for the ℓ_∞ norm. Likewise storage requirements are $\mathcal{O}(q\varepsilon + qp)$. Thus, ADMM and AMA share the same complexity per update round and the same global storage requirements. Unfortunately, the convergence rate of the current ADMM algorithm is unknown, rendering it impossible to give overall complexity bounds analogous to those for AMA. We note, however, that a closely related alternating linearization method and its accelerated variant [26] show sublinear iteration complexity similar to AMA and accelerated AMA.

7. Practical Implementation.

This section addresses practical issues of algorithm implementation.

7.1. Choosing weights.

The choice of the weight can dramatically affect the quality of the clustering path. We set the value of the weight between the ith and jth points to be $w_{ij} = \iota^k_{\{i,j\}} \exp(-\phi\|\mathbf{x}_i - \mathbf{x}_j\|_2^2)$, where $\iota^k_{\{i,j\}}$ is 1 if j is among i's k-nearest-neighbors or vice versa and 0 otherwise. The second factor is a Gaussian kernel that slows the coalescence of distant points. The constant ϕ is nonnegative; the value $\phi = 0$ corresponds to uniform weights. As noted earlier, limiting positive weights to nearest neighbors improves both computational efficiency and clustering quality. Although the two factors defining the weights act similarly, their combination increases the sensitivity of the clustering path to the local density of the data.

7.2. Making cluster assignments.

For both ADMM and AMA cluster assignments can be performed easily once we estimate $\mathbf{V}$. The columns $\mathbf{v}_l$ of $\mathbf{V}$ that are $\mathbf{0}$ or nearly so create an adjacency list of nodes belonging to the same cluster. We apply breadth first search on this adjacency list to identify the clusters.

7.3. Implementing backtracking in AMA.

Backtracking is a useful adjunct to the AMA algorithm. Recall that AMA performs proximal gradient ascent to maximize the dual function $D_\gamma(\boldsymbol{\lambda})$ defined by equation (3.2). Let $h(\boldsymbol{\lambda})$ denote the smooth part of $D_\gamma(\boldsymbol{\lambda})$ ignoring the indicator functions. The gradient of $h(\boldsymbol{\lambda})$ can be computed blockwise as

$$\begin{aligned}\nabla_{\boldsymbol{\lambda}_l} h(\boldsymbol{\lambda}) &= \mathbf{x}_{l_1} - \mathbf{x}_{l_2} + \boldsymbol{\Delta}_{l_1} - \boldsymbol{\Delta}_{l_2}\\ \boldsymbol{\Delta}_i &= \sum_{l_1 = i} \boldsymbol{\lambda}_l - \sum_{l_2 = i} \boldsymbol{\lambda}_l.\end{aligned}$$

A potential update $\boldsymbol{\lambda}^+$ is calculated by taking a forward step of length ν in the direction $\nabla h(\boldsymbol{\lambda})$ and then projecting the result backward onto the corresponding constraint sets. These two steps are expressed blockwise by

$$\boldsymbol{\lambda}_l^+ = P_{C_l}[\boldsymbol{\lambda}_l + \nu \nabla_{\boldsymbol{\lambda}_l} h(\boldsymbol{\lambda})].$$

We then check to see if the local quadratic approximation to $h(\boldsymbol{\lambda})$ increases. This amounts to checking whether the inequality

$$h(\boldsymbol{\lambda}^+) \geq h(\boldsymbol{\lambda}^{m-1}) + \sum_l \langle \boldsymbol{\lambda}_l^+ - \boldsymbol{\lambda}_l^{m-1}, \nabla_{\boldsymbol{\lambda}_l} h(\boldsymbol{\lambda}^{m-1})\rangle - \frac{1}{2\nu}\|\boldsymbol{\lambda}^+ - \boldsymbol{\lambda}^{m-1}\|_2^2$$

holds. If it does, then we accept $\boldsymbol{\lambda}^+$ as our update. Otherwise, we decrease ν and try again. This rule ensures convergence in both the regular proximal gradient algorithm as well as its Nesterov accelerated version [5]. Algorithm 5 shows our AMA approach with both backtracking and Nesterov acceleration.

Algorithm 5 Accelerated AMA with backtracking

```
Initialize $\nu > 0$ and $\boldsymbol{\lambda}^0$. Take $\eta > 1$. Set $\alpha_1 = 1$ and $\mathbf{s}^0 = \boldsymbol{\lambda}^0$.
for $m = 1, 2, 3, \ldots$ do                                        ▷ Compute gradient
    for $i = 1, \ldots, p$ do
        $\boldsymbol{\Delta}_i^m = \sum_{l_1 = i} \boldsymbol{\lambda}_l^{m-1} - \sum_{l_2 = i} \boldsymbol{\lambda}_l^{m-1}$
    end for
    for all $l$ do
        $\mathbf{g}_l^m = \mathbf{x}_{l_1} - \mathbf{x}_{l_2} + \boldsymbol{\Delta}_{l_1}^m - \boldsymbol{\Delta}_{l_2}^m$
    end for                                                          ▷ Backtrack
    loop
        for all $l$ do
            $\boldsymbol{\lambda}_l^+ = P_{C_l}(\boldsymbol{\lambda}_l^{m-1} - \nu \mathbf{g}_l^m)$
        end for
        if $f(\boldsymbol{\lambda}^+) < f(\boldsymbol{\lambda}^{m-1}) + \sum_l \langle \boldsymbol{\lambda}_l^+ - \boldsymbol{\lambda}_l^{m-1}, \mathbf{g}_l^m \rangle - \frac{1}{2\nu} \lVert \boldsymbol{\lambda}^+ + \boldsymbol{\lambda}^{m-1} \rVert_2^2$ then
            $\nu \leftarrow \nu / \eta$
        else
            break
        end if
    end loop                                                         ▷ Take extrapolated step
    $\mathbf{y}^m \leftarrow \boldsymbol{\lambda}^+$
    $\alpha_{m+1} = \frac{1 + \sqrt{1 + 4\alpha_m^2}}{2}$
    $\boldsymbol{\lambda}^m \leftarrow \mathbf{y}^m + \left( \frac{\alpha_m - 1}{\alpha_{m+1}} \right) (\mathbf{y}^m - \mathbf{y}^{m-1})$
end for
```

8. Numerical Experiments.

We now report numerical experiments on convex clustering for a synthetic data set and three real data sets. In particular we focus on how the choice of the weights w_{ij} affects the quality of the clustering solution. Prior research on this question is limited. Both Lindsten et al. and Hocking et al. suggest weights derived from Gaussian kernels and k-nearest neighbors. Because Hocking et al. try only Gaussian kernels, in this section we follow up their untested suggestion of combining Gaussian kernels and k-nearest neighbors.

We also compare the run times of our splitting methods to the run times of the subgradient algorithm employed by Hocking et al. for ℓ_1 and ℓ_2 paths. They provide R and C++ code for their algorithms. Our algorithms are implemented in R and Fortran. A direct comparison of our code and theirs is impossible since we use a different lower-level language and perhaps more importantly different stopping criteria. Although we appear to achieve comparable and sometimes more accurate solutions in less time, it is worth pointing out that generating the most accurate minima to problem (1.1) is somewhat irrelevant. Two approximate solutions with the same clusters should be judged equivalent. Our primary purpose is to document that the splitting strategy is computationally competitive with existing approaches.

8.1. Qualitative Comparisons.

Our next few examples demonstrate how the character of the solution paths can vary drastically with the choice of weights w_{ij}.

8.1.1. Two Half Moons.

Consider the standard simulated data of two interlocking half moons in $\mathbb{R}^2$ composed of 100 points each. Figure 8.1 shows four convex clustering paths computed assuming two different numbers of nearest neighbors (10

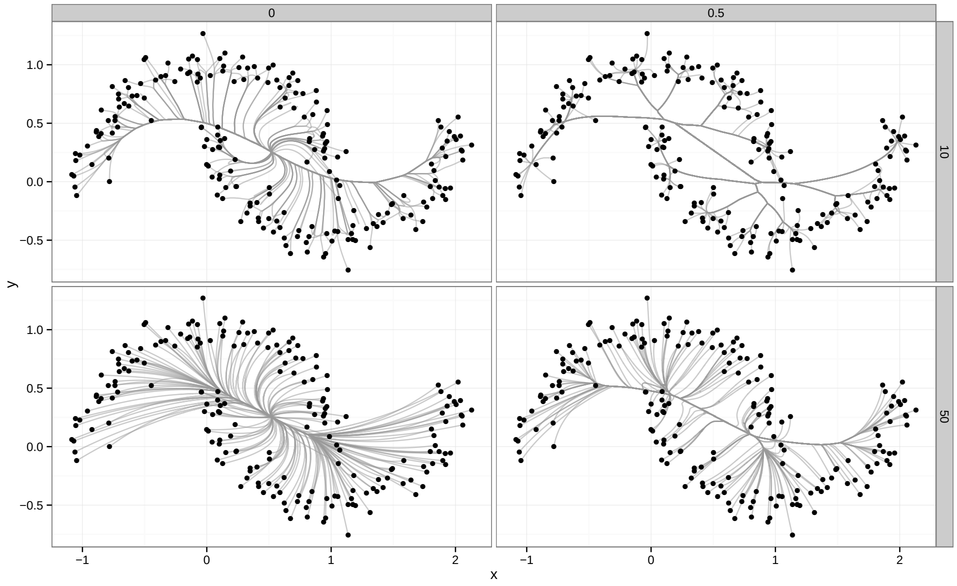


Fig. 8.1: Halfmoons Example: The first and second rows show results using $k = 10$ and 50 nearest neighbors respectively. The first and second columns show results using $\phi = 0$ and 0.5 respectively.

and 50) and two different kernel constants ϕ (0 and 0.5). The upper right panel makes it evident that limiting the number of nearest neighbors ($k = 10$) and imposing non-uniform Gaussian kernel weights ($\phi = 0.5$) produce the best clustering path. Using too many neighbors and assuming uniform weights results in little agglomerative clustering until late in the clustering path (lower left panel). The two intermediate cases diverge in interesting ways. The hardest set of points to cluster are the points in the upper half moon's right tip and the lower half moon's left tip. Limiting the number of nearest neighbors and omitting the Gaussian kernel (upper left panel) correctly agglomerates the easier points, but waffles on the harder points, agglomerating them only at the very end when all points coalesce at the grand mean. Conversely, using many neighbors and the Gaussian kernel (lower right panel) leads to a clustering path that does not hedge but incorrectly assigns the harder points.

8.1.2. Fisher's Iris Data. Fisher's Iris data [19] consists of four measurements on 150 samples of iris flowers. There are three species present: setosa, versicolor, and virginica. Figure 8.2a shows the resulting clustering paths under two different choices of weights. On the left $w_{ij} = 1$ for all $i < j$, and on the right we used 5-nearest neighbors and $\phi = 4$. Since there are four variables, to visualize results we project the data and the fitted clustering paths onto the first two principal components of the data. Again we see that more sensible clustering is observed when we choose weights to be sensitive to the local data density. We even get some separation between the overlapping species virginica and versicolor.

8.1.3. Senate Voting. We consider senate voting in 2001 on a subset of 15 issues selected by Americans for Democratic Action [13, 2]. The data is binary. We limited our study to the 29 senators with unique voting records. The issues ranged over a wide spectrum: domestic, foreign, economic, military, environmental and social concerns. The final group of senators included 15 Democrats, 13 Republicans, and 1 Independent. Figure 8.2b shows the resulting clustering paths under two different choices of weights. On the left $w_{ij} = 1$ for all $i < j$, and on the right we used 15-nearest neighbors and $\phi = 0.5$. As observed previously, better clustering is observed when we choose the weights to be sensitive to the local data density. In particular, we get clear party separation. Note that we have an outlying Democrat in Zel Miller and that the clustering seen agrees well with what PCA exposes.

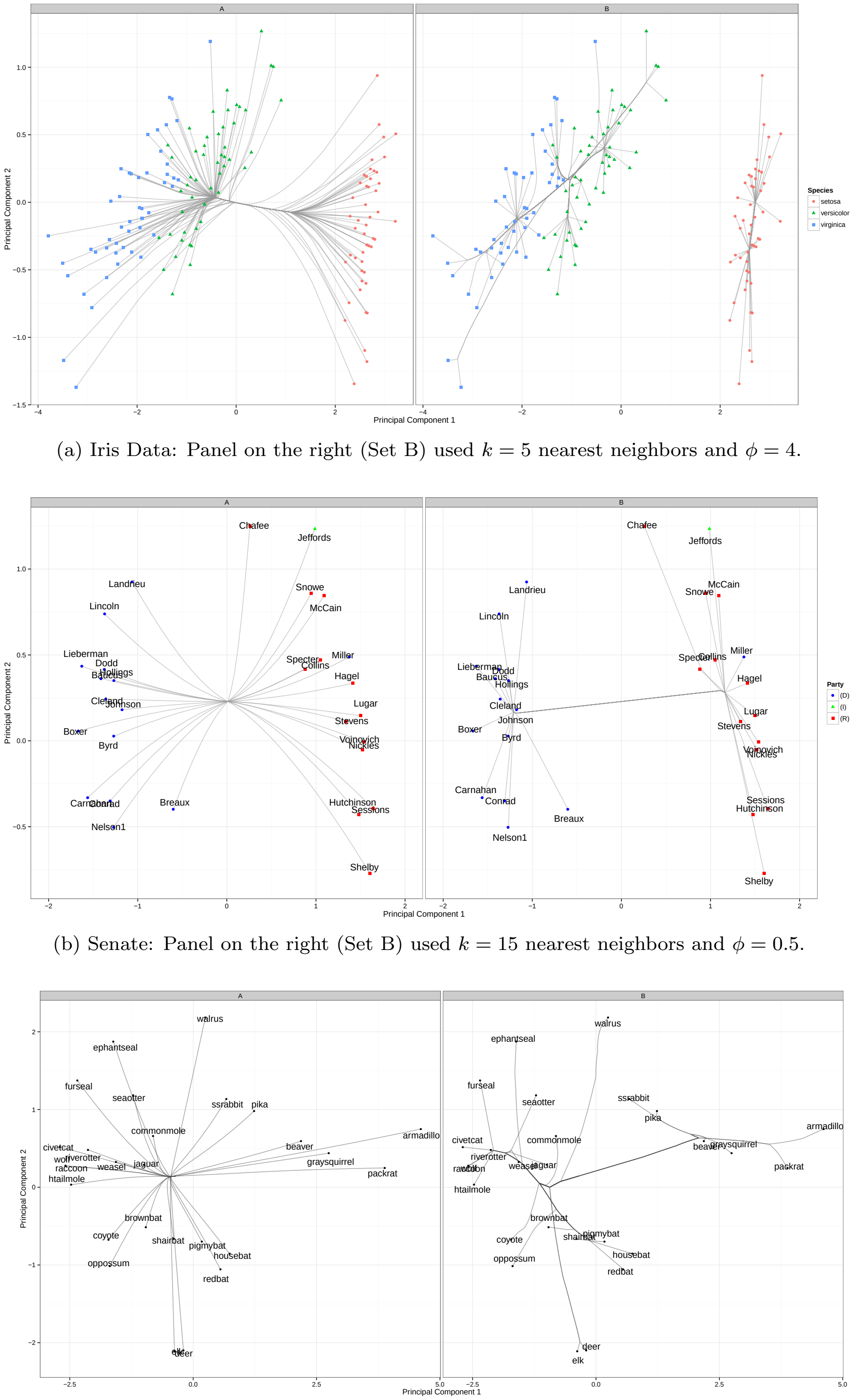
8.1.4. Dentition of mammals. Finally, we consider the problem of clustering mammals based on their dentition [13, 32]. Eight different kinds of teeth are tallied up for each mammal: the number of top incisors, bottom incisors, top canines, bottom canines, top premolars, bottom premolars, top molars, and bottom molars. Again we removed observations with teeth distributions that were not unique, leaving us with 27 mammals. Figure 8.2c shows the resulting clustering paths under two different choices of weights. On the left $w_{ij} = 1$ for all $i < j$, and on the right we used 5-nearest neighbors and $\phi = 0.5$. Once again, weights sensitive to the local density give superior results. In contrast to the iris and senate data, the cluster path gives a different and perhaps more sensible solution than projections onto the first two components PCA. For example, the brown bat is considered more similar to the house bat and red bat, even though it is closer in the first two PCA coordinates to the coyote and oppossum.

8.2. Computational Comparisons. We compared the run times between the subgradient descent algorithm of Hocking et al., ADMM, and AMA. We ran all algorithms, including the regular and fast versions of ADMM and AMA, on the four data sets described above. For all tests we assigned weights using full connectivity, $\iota^k_{\{i,j\}} = 1$ and $\phi = -2$. The parameter ϕ was chosen to ensure that the smallest weight was bounded safely away from zero. We used the default stopping criterion for the Hocking algorithm, namely that the ℓ_2 norm of the gradient is no more than 0.001. For our AMA algorithm we used $\tau = 10^{-6}$. Both the regular and fast versions of AMA employed backtracking. For our ADMM algorithm we used $\tau = 5 \times 10^{-2}$ and $\nu = 1$. We used the same sequence of γ for all algorithms. The experiments were performed on an iMac computer with a 3.4 GHz Intel Core i7 processor and 8 GB of RAM. To ensure that all algorithms ran to comparable degree of accuracy, we plotted the relative difference in the primal objective function between the subgradient descent solution and our algorithm's solutions, namely

$$\frac{F_\gamma(\mathbf{U}^{sg}) - F_\gamma(\mathbf{U}^{alt})}{F_\gamma(\mathbf{U}^{sg})},$$

where $\mathbf{U}^{sg}$ is the estimated solution by subgradient descent and $\mathbf{U}^{alt}$ is the estimated solution by one of our alternating methods. Negative values indicate better solutions by the subgradient descent, while positive values indicate better solutions by method x.

Figure 8.3a and Figure 8.3b show that the relative difference over the sequence of γ used. We see that all algorithms produced solutions of comparable quality. Table 8.1 and Table 8.2, show the run times averaged over 10 trials. AMA appears to be superior in our study. We see that acceleration for AMA and ADMM does not pay dividends on the smaller data sets, Senate and Mammals. While ADMM is the slower

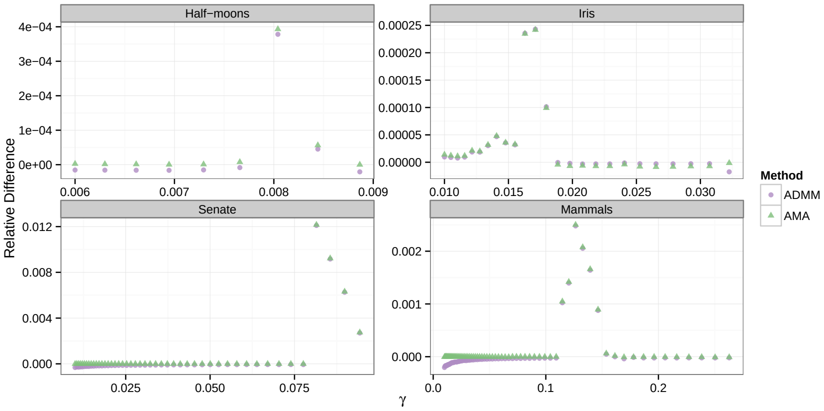


(a) Iris Data: Panel on the right (Set B) used $k = 5$ nearest neighbors and $\phi = 4$.

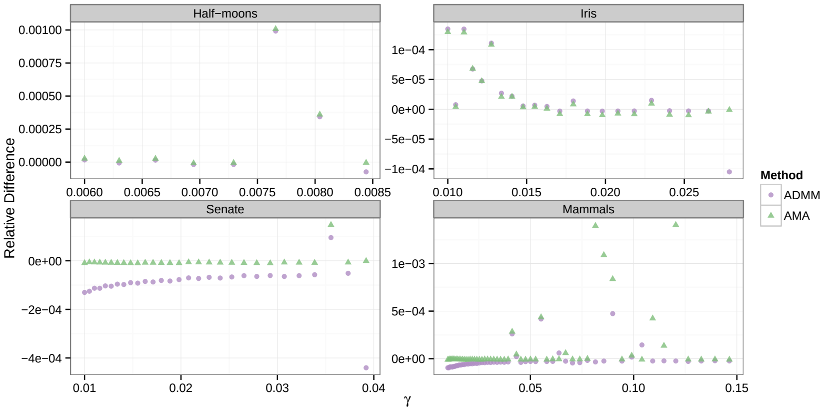
(b) Senate: Panel on the right (Set B) used $k = 15$ nearest neighbors and $\phi = 0.5$.

(c) Mammal Data: Panel on the right (Set B) used $k = 5$ nearest neighbors and $\phi = 0.5$.

Fig. 8.2: Clustering path under the ℓ_2 norm. All panels on the left (Set A) used $w_{ij} = 1$ for all $i < j$.



(a) Relative difference in primal objective between Subgradient Descent and ADMM/AMA (ℓ_2 norm penalty)



(b) Relative difference in primal objective between Subgradient Descent and ADMM/AMA (ℓ_1 norm penalty)

	Subgradient	AMA		ADMM	
Data	Descent	Regular	Fast	Regular	Fast
Half moons	42.01	1.56	**1.03**	7.81	4.88
Iris	5.97	1.84	**1.04**	13.59	9.06
Senate	0.20	**0.06**	0.07	4.36	4.22
Mammals	0.21	0.10	**0.08**	6.05	6.21

Table 8.1: Timing comparison under the ℓ_2 norm. Mean run times (sec) over 10 trials are reported.

of the two, ADMM proves its worth in the half moons example where the number of data points is larger.

9. Conclusion & Future Work. In this paper we introduce two splitting algorithms for solving the convex clustering problem. The splitting perspective encourages path following, one of the chief benefits of convex clustering. The splitting perspective

	Subgradient	AMA		ADMM	
Data	Descent	Regular	Fast	Regular	Fast
Half moons	58.31	1.25	**0.90**	4.44	3.12
Iris	12.11	1.18	**0.78**	8.00	5.28
Senate	0.30	0.06	**0.05**	2.52	2.58
Mammals	0.42	0.08	**0.06**	5.01	5.07

Table 8.2: Timing comparison under the ℓ_1 norm. Mean run times (sec) over 10 trials are reported.

also permits centroid penalties to invoke an arbitrary norm. The only requirement is that the proximal map for the norm be readily computable. Equivalently, projection onto the unit ball of the dual norm should be straightforward. Because proximal maps and projection operators are generally well understood, it is possible for us to quantify the computational complexity and convergence properties of our algorithms.

The accelerated AMA method appears to be the faster algorithm. On the other hand, ADMM is simpler to implement since the choice of the parameter ν is fixed. Given that alternative variants of ADMM may close the performance gap [14, 26], we are reluctant to dismiss ADMM too quickly. In any case, both algorithms provide viable alternatives to existing approaches and deserve further investigation. For instance, in both ADMM and AMA, updates of $\boldsymbol{\lambda}$ and $\mathbf{V}$ could be parallelized. Hocking et al. also employed an active set approach to reduce computations as the centroids coalesce. A similar strategy could be adopted in our framework, but it incurs additional overhead as checks for unfusing events have to be introduced.

The storage demands and computational complexity of convex clustering depend quadratically on the number of edges of the associated weight graph in the worst case. Limiting a point's connections to its k-nearest neighbors, for example, ensures that the number of edges in the graph is linear in the number of nodes in the graph. Eliminating long-range dependencies is often desirable anyway. Choosing sparse weights can improve both cluster quality and computational efficiency. Moreover, finding the exact k-nearest neighbors is likely not essential, and we conjecture the quality of solutions would not suffer greatly if approximate nearest neighbors are used and algorithms for fast computation of approximately nearest neighbors are leveraged [55]. On very large problems, the best strategy might be to exploit the continuity of solution paths in the weights. This suggests starting with even sparser graphs than the desired one and generating a sequence of solutions to increasingly dense problems. A solution with fewer edges can serve as a warm start for the next problem with more edges.

The splitting perspective also invite extensions that impose structured sparsity on the centroids. Witten and Tibshirani [66] discuss how sparse centroids can improve the quality of a solution, especially when only a relatively few features of data drive clustering. Structured sparsity can be accomplished by adding a sparsity inducing norm penalty to the $\mathbf{U}$ updates. The update for the centroids for both AMA and ADMM then rely on another proximal map of a gradient step. Introducing a sparsifying norm, however, raises the additional complication of choosing the amount of penalization.

More general problems than clustering can be solved via the proximal gradient method in conjunction with splitting. For instance, minimizing the criterion

$$\text{minimize } f(\mathbf{U}) + \gamma \sum_{i<j} w_{ij} \|\mathbf{u}_i - \mathbf{u}_j\|$$

yields to the proximal gradient method whenever the function $f(\mathbf{U})$ is convex with a Lipschitz continuous gradient. Finally, except for a few hints about weights, our analysis leaves the topic of optimal clustering untouched. Recently Luxburg [64] has suggested some principled approaches to assessing the quality of a clustering assignment via data perturbation and resampling. These clues are worthy of further investigation.

Appendix A. AMA performs proximal gradient ascent on the dual.

In demonstrating the equivalence of the proximal gradient updates and the AMA updates (3.7), we build on the findings and notation of Section §3.1. Our point of departure is the observation $\mathbf{u} = \nabla f^\star(\mathbf{A}^t\boldsymbol{\lambda})$ if and only if $\mathbf{u}$ minimizes $f(\mathbf{u}) - \langle \mathbf{u}, \mathbf{A}^t\boldsymbol{\lambda}\rangle$. Thus, we can write the proximal step in stages

$$\begin{aligned}\mathbf{u} &= \arg\min_{\mathbf{u}} \left[f(\mathbf{u}) - \langle \mathbf{A}\mathbf{u}, \boldsymbol{\lambda}\rangle \right] \\ \boldsymbol{\lambda} &= \operatorname{prox}_{\nu\tilde{g}}(\boldsymbol{\lambda} - \nu\mathbf{A}\mathbf{u}).\end{aligned}$$

We next show how the computation of the proximal map $\boldsymbol{\lambda} = \operatorname{prox}_{\nu\tilde{g}}(\boldsymbol{\mu})$ can be decomposed into two stages. This tactic involves solving the dual of the optimization problem associated with the proximal map and then converting the dual solution to the solution of the original proximal map. In the current setting the proximal map returns the global solution to the problem

$$\underset{\boldsymbol{\lambda}}{\text{maximize}} \left[\langle \boldsymbol{\lambda}, \mathbf{c}\rangle - g^\star(\mathbf{B}^t\boldsymbol{\lambda}) - \frac{1}{2\nu}\|\boldsymbol{\lambda} - \boldsymbol{\mu}\|_2^2 \right].$$

The above maximization is dual to the equality constrained minimization over $(\mathbf{v}, \mathbf{z})$

$$\begin{aligned}&\underset{\mathbf{v},\mathbf{z}}{\text{minimize}} \;\; g(\mathbf{v}) + \langle \boldsymbol{\mu}, \mathbf{z}\rangle + \frac{\nu}{2}\|\mathbf{z}\|_2^2 \\ &\text{subject to } \mathbf{B}\mathbf{v} + \mathbf{z} = \mathbf{c}.\end{aligned}$$

Strong duality holds by Slater's constraint qualification since $(\mathbf{v}, \mathbf{z}) = (\mathbf{0}, \mathbf{c})$ is a feasible point [7]. Thus, at a solution $(\boldsymbol{\lambda}, \mathbf{v})$, the primal and dual objectives satisfy

$$\langle \mathbf{c}, \boldsymbol{\lambda}\rangle - g^\star(\mathbf{B}^t\boldsymbol{\lambda}) - \frac{1}{2\nu}\|\boldsymbol{\lambda} - \boldsymbol{\mu}\|_2^2 = g(\mathbf{v}) + \langle \boldsymbol{\mu}, \mathbf{c} - \mathbf{B}\mathbf{v}\rangle + \frac{\nu}{2}\|\mathbf{c} - \mathbf{B}\mathbf{v}\|_2^2.$$

Rearranging terms we get

$$0 = -\langle \mathbf{c}, \boldsymbol{\lambda}\rangle + g^\star(\mathbf{B}^t\boldsymbol{\lambda}) + \frac{1}{2\nu}\|\boldsymbol{\lambda} - \boldsymbol{\mu}\|_2^2 + g(\mathbf{v}) + \langle \boldsymbol{\mu}, \mathbf{c} - \mathbf{B}\mathbf{v}\rangle + \frac{\nu}{2}\|\mathbf{c} - \mathbf{B}\mathbf{v}\|_2^2.$$

But the Fenchel-Young inequality says that $g(\mathbf{v}) + g^\star(\mathbf{B}^t\boldsymbol{\lambda}) \geq \langle \mathbf{B}\mathbf{v}, \boldsymbol{\lambda}\rangle$. Combining this fact with the above equality gives the inequality,

$$0 \geq -\langle \mathbf{c}, \boldsymbol{\lambda}\rangle + \langle \mathbf{B}\mathbf{v}, \boldsymbol{\lambda}\rangle + \frac{1}{2\nu}\|\boldsymbol{\lambda} - \boldsymbol{\mu}\|_2^2 + \langle \boldsymbol{\mu}, \mathbf{c} - \mathbf{B}\mathbf{v}\rangle + \frac{\nu}{2}\|\mathbf{c} - \mathbf{B}\mathbf{v}\|_2^2.$$

The right hand side simplifies to

$$\begin{aligned}0 &\geq \|\boldsymbol{\lambda} - \boldsymbol{\mu}\|_2^2 - 2\nu\langle \mathbf{c} - \mathbf{B}\mathbf{v}, \boldsymbol{\lambda} - \boldsymbol{\mu}\rangle + \nu^2\|\mathbf{c} - \mathbf{B}\mathbf{v}\|_2^2 \\ &= \|\boldsymbol{\lambda} - \boldsymbol{\mu} - \nu(\mathbf{c} - \mathbf{B}\mathbf{v})\|_2^2.\end{aligned}$$

Since the right hand side must always be non-negative, we arrive at the equality

$$\boldsymbol{\lambda} = \boldsymbol{\mu} + \nu(\mathbf{c} - \mathbf{B}\mathbf{v}).$$

This relationship tells us how to translate between the primal and dual solutions. Thus, we compute $\text{prox}_{\nu\tilde{g}}(\boldsymbol{\mu})$ in two steps.

$$\begin{aligned}
\mathbf{v} &= \arg\min_{\mathbf{v}} \left[g(\mathbf{v}) + \langle \boldsymbol{\mu}, \mathbf{c} - \mathbf{B}\mathbf{v} \rangle + \frac{\nu}{2}\|\mathbf{c} - \mathbf{B}\mathbf{v}\|_2^2 \right] \\
\text{prox}_{\nu\tilde{g}}(\boldsymbol{\mu}) &= \boldsymbol{\mu} + \nu(\mathbf{c} - \mathbf{B}\mathbf{v}).
\end{aligned}$$

Taking $\boldsymbol{\mu} = \boldsymbol{\lambda} - \nu\mathbf{A}\mathbf{u}$, we can compute the entire proximal gradient update in three steps as

$$\begin{aligned}
\mathbf{u}^{+} &= \arg\min_{\mathbf{u}} \left[f(\mathbf{u}) - \langle \mathbf{A}\mathbf{u}, \boldsymbol{\lambda} \rangle \right] \\
\mathbf{v}^{+} &= \arg\min_{\mathbf{v}} \left[g(\mathbf{v}) + \langle \boldsymbol{\lambda} - \nu\mathbf{A}\mathbf{u}^{+}, \mathbf{c} - \mathbf{B}\mathbf{v} \rangle + \frac{\nu}{2}\|\mathbf{c} - \mathbf{B}\mathbf{v}\|_2^2 \right] \\
\boldsymbol{\lambda}^{+} &= \boldsymbol{\lambda} - \nu\mathbf{A}\mathbf{u}^{+} + \nu(\mathbf{c} - \mathbf{B}\mathbf{v}^{+}) \\
&= \boldsymbol{\lambda} + \nu[\mathbf{c} - \mathbf{A}\mathbf{u}^{+} - \mathbf{B}\mathbf{v}^{+}].
\end{aligned}$$

Note that the first and second of these updates are equivalent to

$$\begin{aligned}
\mathbf{u}^{m+1} &= \arg\min_{\mathbf{u}} \mathcal{L}_0(\mathbf{u}, \mathbf{v}^m, \boldsymbol{\lambda}) \\
\mathbf{v}^{m+1} &= \arg\min_{\mathbf{v}} \mathcal{L}_\nu(\mathbf{u}^{m+1}, \mathbf{v}, \boldsymbol{\lambda}).
\end{aligned}$$

REFERENCES

[1] D. Aloise, A. Deshpande, P. Hansen, and P. Popat, *NP-hardness of Euclidean sum-of-squares clustering*, Machine Learning, 75 (2009), pp. 245–248.

[2] Americans for Democratic Action, *2001 voting record: Shattered promise of liberal progress*, ADA Today, 57 (2002), pp. 1–17.

[3] W. N. Anderson and T. D. Morley, *Eigenvalues of the Laplacian of a graph*, Linear and Multilinear Algebra, 18 (1985), pp. 141–145.

[4] D. Arthur and S. Vassilvitskii, *k-means++: The advantages of careful seeding*, in Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms, SODA '07, Philadelphia, PA, USA, 2007, Society for Industrial and Applied Mathematics, pp. 1027–1035.

[5] A. Beck and M. Teboulle, *A fast iterative shrinkage-thresholding algorithm for linear inverse problems*, SIAM Journal on Imaging Sciences, 2 (2009), pp. 183–202.

[6] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, *Distributed optimization and statistical learning via the alternating direction method of multipliers*, Found. Trends Mach. Learn., 3 (2011), pp. 1–122.

[7] S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge University Press, 2004.

[8] P. S. Bradley, O. L. Mangasarian, and W. N. Street, *Clustering via concave minimization*, in Advances in Neural Information Processing Systems, MIT Press, 1997, pp. 368–374.

[9] S. S. Chen, D. L. Donoho, and M. A. Saunders, *Atomic decomposition by basis pursuit*, SIAM Journal on Scientific Computing, 20 (1998), pp. 33–61.

[10] P. Combettes and V. Wajs, *Signal recovery by proximal forward-backward splitting*, Multiscale Modeling & Simulation, 4 (2005), pp. 1168–1200.

[11] CVX Research, Inc., *CVX: Matlab software for disciplined convex programming, version 2.0 beta*. http://cvxr.com/cvx, Sept. 2012.

[12] S. Dasgupta and Y. Freund, *Random projection trees for vector quantization*, IEEE Trans. Inf. Theor., 55 (2009), pp. 3229–3242.

[13] J. de Leeuw and P. Mair, *Gifi methods for optimal scaling in R: The Package homals*, Journal of Statistical Software, 31 (2009), pp. 1–21.

[14] W. Deng and W. Yin, *On the global and linear convergence of the generalized alternating direction method of multipliers*, Tech. Report CAAM Technical Report TR12-14, Rice University, 2012.

[15] J. Duchi, S. Shalev-Shwartz, Y. Singer, and T. Chandra, *Efficient projections onto the ℓ_1-ball for learning in high dimensions*, in Proceedings of the International Conference on Machine Learning, 2008.

[16] J. Eckstein and D. P. Bertsekas, *On the Douglas-Rachford splitting method and the proximal point algorithm for maximal monotone operators*, Mathematical Programming, 55 (1992), pp. 293–318.

[17] C. Elkan, *Using the triangle inequality to accelerate k-means*, in Proceedings of ICML 2003, 2003.

[18] T. S. Ferguson, *A Bayesian analysis of some nonparametric problems*, Annals of Statistics, 1 (1973), pp. 209–230.

[19] R. A. Fisher, *The use of multiple measurements in taxonomic problems*, Annals of Eugenics, 7 (1936), pp. 179–188.

[20] E. Forgy, *Cluster analysis of multivariate data: efficiency versus interpretability of classifications*, Biometrics, 21 (1965), pp. 768–780.

[21] C. Fraley, *Algorithms for model-based gaussian hierarchical clustering*, SIAM Journal on Scientific Computing, 20 (1998), pp. 270–281.

[22] M. Frank and P. Wolfe, *An algorithm for quadratic programming*, Naval Research Logistics Quarterly, 3 (1956), pp. 95–110.

[23] D. Gabay, *Augmented Lagrangian Methods: Applications to the Solution of Boundary-Value Problems*, North-Holland, Amsterdam, 1983, ch. Applications of the method of multipliers to variational inequalities.

[24] D. Gabay and B. Mercier, *A dual algorithm for the solution of nonlinear variational problems via finite-element approximations*, Comp. Math. Appl., 2 (1976), pp. 17–40.

[25] R. Glowinski and A. Marrocco, *Sur lapproximation par elements finis dordre un, et la resolution par penalisation-dualite dune classe de problemes de dirichlet nonlineaires*, Rev. Francaise dAut. Inf. Rech. Oper., 2 (1975), pp. 41–76.

[26] D. Goldfarb, S. Ma, and K. Scheinberg, *Fast alternating linearization methods for minimizing the sum of two convex functions*, Mathematical Programming, (2012), pp. 1–34.

[27] T. Goldstein, B. O'Donoghue, and S. Setzer, *Fast alternating direction optimization methods*, Tech. Report cam12-35, University of California, Los Angeles, 2012.

[28] T. Goldstein and S. Osher, *The split Bregman method for l1-regularized problems*, SIAM Journal on Imaging Sciences, 2 (2009), pp. 323–343.

[29] A. Gordon, *Classification*, Chapman and Hall/CRC Press, London, 2nd ed., 1999.

[30] J. C. Gower and G. J. S. Ross, *Minimum spanning trees and single linkage cluster analysis*, Journal of the Royal Statistical Society. Series C (Applied Statistics), 18 (1969), pp. 54–64.

[31] M. Grant and S. Boyd, *Graph implementations for nonsmooth convex programs*, in Recent Advances in Learning and Control, V. Blondel, S. Boyd, and H. Kimura, eds., Lecture Notes in Control and Information Sciences, Springer-Verlag Limited, 2008, pp. 95–110. http://stanford.edu/~boyd/graph_dcp.html.

[32] J. Hartigan, *Clustering Algorithms*, Wiley, New York, 1975.

[33] M. Hestenes, *Multiplier and gradient methods*, Journal of Optimization Theory and Applications, 4 (1969), pp. 303–320.

[34] T. Hocking, J.-P. Vert, F. Bach, and A. Joulin, *Clusterpath: an algorithm for clustering using convex fusion penalties*, in Proceedings of the 28th International Conference on Machine Learning (ICML-11), L. Getoor and T. Scheffer, eds., ICML '11, New York, NY, USA, June 2011, ACM, pp. 745–752.

[35] H. Hoefling, *A path algorithm for the fused lasso signal approximator*, Journal of Computational and Graphical Statistics, 19 (2010), pp. 984–1006.

[36] S. Johnson, *Hierarchical clustering schemes*, Psychometrika, 32 (1967), pp. 241–254.

[37] S. M. Kakade, S. Shalev-Shwartz, and A. Tewari, *On the duality of strong convexity and strong smoothness: Learning applications and matrix regularization*, tech. report, Toyota Technological Institute, 2009.

[38] L. Kaufman and P. Rousseeuw, *Finding Groups in Data: An Introduction to Cluster Analysis*, Wiley, New York, 1990.

[39] G. N. Lance and W. T. Williams, *A general theory of classificatory sorting strategies: 1. hierarchical systems*, The Computer Journal, 9 (1967), pp. 373–380.

[40] F. Lindsten, H. Ohlsson, and L. Ljung, *Just relax and come clustering! A convexication of k-means clustering*, tech. report, Linköpings universitet, 2011.

[41] S. Lloyd, *Least squares quantization in PCM*, Information Theory, IEEE Transactions on, 28

(1982), pp. 129 – 137.

[42] U. Luxburg, *A tutorial on spectral clustering*, Statistics and Computing, 17 (2007), pp. 395–416.

[43] J. MacQueen, *Some methods for classification and analysis of multivariate observations*, in Proc. Fifth Berkeley Symp. on Math. Statist. and Prob., vol. 1, Univ. of Calif. Press, 1967, pp. 281–297.

[44] J. Mairal, R. Jenatton, G. Obozinski, and F. Bach, *Convex and network flow optimization for structured sparsity*, J. Mach. Learn. Res., 12 (2011), pp. 2681–2720.

[45] G. McLachlan, *Finite Mixture Models*, Wiley, Hoboken, New Jersey, 2000.

[46] C. Michelot, *A finite algorithm for finding the projection of a point onto the canonical simplex of $\mathbb{R}^n$*, Journal of Optimization Theory and Applications, 50 (1986), pp. 195–200.

[47] K. S. Miller, *Some eclectic matrix theory*, Robert E. Krieger Publishing Company, Inc., 1987.

[48] B. Mirkin, *Mathematical Classification and Clustering*, Kluwer Academic Publishers, Dordrecht, The Netherlands, 1996.

[49] F. Murtagh, *A survey of recent advances in hierarchical clustering algorithms*, The Computer Journal, 26 (1983), pp. 354–359.

[50] R. M. Neal, *Markov chain sampling methods for Dirichlet process mixture models*, Journal of Computational and Graphical Statistics, 9 (2000), pp. pp. 249–265.

[51] J. Nocedal and S. Wright, *Numerical Optimization*, Springer, 2nd ed., 2006.

[52] M. Powell, *A method for nonlinear constraints in minimization problems*, in Optimization, R. Fletcher, ed., Academic Press, 1969, pp. 283–298.

[53] C. E. Rasmussen, *The infinite Gaussian mixture model*, in In Advances in Neural Information Processing Systems 12, MIT Press, 2000, pp. 554–560.

[54] R. Rockafellar, *The multiplier method of Hestenes and Powell applied to convex programming*, Journal of Optimization Theory and Applications, 12 (1973), pp. 555–562.

[55] M. Slaney and M. Casey, *Locality-sensitive hashing for finding nearest neighbors [lecture notes]*, Signal Processing Magazine, IEEE, 25 (2008), pp. 128–131.

[56] P. H. A. Sneath, *The application of computers to taxonomy*, Journal of General Microbiology, 17 (1957), pp. 201–226.

[57] T. Sørensen, *A method of establishing groups of equal amplitude in plant sociology based on similarity of species and its application to analyses of the vegetation on danish commons*, Biologiske Skrifter, 5 (1948), pp. 1–34.

[58] R. Tibshirani, *Regression shrinkage and selection via the lasso*, Journal of the Royal Statistical Society, Ser. B, 58 (1996), pp. 267–288.

[59] R. Tibshirani, M. Saunders, S. Rosset, J. Zhu, and K. Knight, *Sparsity and smoothness via the fused lasso*, Journal of the Royal Statistical Society: Series B (Statistical Methodology), 67 (2005), pp. 91–108.

[60] R. J. Tibshirani and J. Taylor, *The solution path of the generalized lasso*, Annals of Statistics, 39 (2011), pp. 1335–1371.

[61] D. M. Titterington, A. F. M. Smith, and U. E. Makov, *Statistical Analysis of Finite Mixture Distributions*, John Wiley & Sons, Hoboken, New Jersey, 1985.

[62] J. Tropp, *Just relax: Convex programming methods for identifying sparse signals in noise*, Information Theory, IEEE Transactions on, 52 (2006), pp. 1030 –1051.

[63] P. Tseng, *Applications of a splitting algorithm to decomposition in convex programming and variational inequalities*, SIAM Journal on Control and Optimization, 29 (1991), pp. 119–138.

[64] U. von Luxburg, *Clustering stability: An overview*, Found. Trends Mach. Learn., 2 (2010), pp. 235–274.

[65] J. H. Ward, *Hierarchical grouping to optimize an objective function*, Journal of the American Statistical Association, 58 (1963), pp. 236–244.

[66] D. M. Witten and R. Tibshirani, *A framework for feature selection in clustering*, J Am Stat Assoc., 105 (2010), pp. 713–726.

[67] R. Wu and D. Wunsch, *Clustering*, John Wiley & Sons, Hoboken, 2009.