Supplementary Materials

A Operator Splitting Methods for Convex Clustering

A.1 ADMM for Convex Clustering

In this section, we derive and give the full form of the ADMM presented in Algorithm 1 for the convex clustering problem (1). We begin by noting that, in typical applications, most of the weights w_{ij} are zero and hence do not enter into the optimization problem. We can omit the $\binom{n}{2}$-term sum and instead write the convex clustering problem (1) as

$$\underset{\boldsymbol{U}\in\mathbb{R}^{n\times p}}{\arg\min}\ \frac{1}{2}\|\boldsymbol{X}-\boldsymbol{U}\|_F^2+\lambda\left(\sum_{((i,j),w)\in\mathcal{E}} w_{ij}\|\boldsymbol{U}_{i\cdot}-\boldsymbol{U}_{j\cdot}\|_q\right)$$

where $\mathcal{E}$ is the set of directed edges with non-zero weights w connecting i to j.

In this form, it is clear that the convex clustering problem is amenable to operator splitting methods; in particular, Chi and Lange (2015) showed that the Alternating Direction Method of Multipliers (ADMM) (Glowinski and Marroco, 1975; Gabay and Mercier, 1976; Boyd et al., 2011) works particularly well for this problem. Algorithm A1 differs from the ADMM derived in Chi and Lange (2015) in two significant ways: firstly, we only consider edges with non-zero weights, thereby greatly reducing storage requirements of the algorithm; and secondly, we implement the algorithm in "matrix-form" rather than in a fully vectorized form. These differences make the resulting algorithm both easier to derive and to read, as well as more able to take advantage of highly optimized numerical linear algebra libraries.

We note that while we are solving a matrix-valued problem, it is not a semi-definite program, and the additional complexity typically associated with matrix-valued optimization does not apply here. Because we are optimizing over the space of all matrices of a certain size,

the underlying problem is essentially Euclidean in geometry and standard (vector-valued) optimization techniques can be applied, replacing the (squared) Euclidean norm with the (squared) Frobenius norm and the standard Euclidean inner product with the Frobenius inner product as necessary.

The derivation of the ADMM for convex clustering 1 is relatively straight-forward. We begin by introducing an auxiliary variable $\boldsymbol{V}$ containing the pairwise differences between connected rows of $\boldsymbol{U}$. The problem then becomes

$$\underset{\boldsymbol{U}\in\mathbb{R}^{n\times p}}{\arg\min}\ \frac{1}{2}\|\boldsymbol{U}-\boldsymbol{X}\|_F^2+\lambda\underbrace{\left(\sum_{(e_l,w_l)\in\mathcal{E}} w_l\|\boldsymbol{V}_{l\cdot}\|_q\right)}_{P(\boldsymbol{V};\boldsymbol{w},q)} \quad \text{subject to } \boldsymbol{D}\boldsymbol{U}-\boldsymbol{V}=0.$$

From here, we use the scaled form of the ADMM as given by a matrix version of Equations (3.5) to (3.7) of Boyd et al. (2011):

$$\begin{aligned}
\boldsymbol{U}^{(k+1)} &= \underset{\boldsymbol{U}\in\mathbb{R}^{n\times p}}{\arg\min}\ \frac{1}{2}\|\boldsymbol{U}-\boldsymbol{X}\|_F^2+\frac{\rho}{2}\left\|\boldsymbol{D}\boldsymbol{U}-\boldsymbol{V}^{(k)}+\boldsymbol{Z}^{(k)}\right\|_F^2\\
\boldsymbol{V}^{(k+1)} &= \underset{\boldsymbol{V}\in\mathbb{R}^{|\mathcal{E}|\times p}}{\arg\min}\ \lambda P(\boldsymbol{V};\boldsymbol{w},q)+\frac{\rho}{2}\left\|\boldsymbol{D}\boldsymbol{U}^{(k+1)}-\boldsymbol{V}+\boldsymbol{Z}^{(k)}\right\|_F^2\\
\boldsymbol{Z}^{(k+1)} &= \boldsymbol{Z}^{(k)}+\boldsymbol{D}\boldsymbol{U}^{(k+1)}-\boldsymbol{V}^{(k+1)}
\end{aligned}$$

where the dual variable is denoted by $\boldsymbol{Z}$. The analytical solution to the first subproblem is given by:

$$\boldsymbol{U}^{(k+1)}=(\boldsymbol{I}+\rho\boldsymbol{D}^T\boldsymbol{D})^{-1}\left(\boldsymbol{X}+\rho\boldsymbol{D}^T(\boldsymbol{V}^{(k)}-\boldsymbol{Z}^{(k)})\right)$$

This update is the most expensive step in the ADMM, though it can be significantly sped up by pre-calculating caching the Cholesky factorization of $\boldsymbol{I}+\rho\boldsymbol{D}^T\boldsymbol{D}$ and using it at each $\boldsymbol{U}$-update:

$$\boldsymbol{U}^{(k+1)}=\boldsymbol{L}^{-T}\boldsymbol{L}^{-1}\left(\boldsymbol{X}+\rho\boldsymbol{D}^T(\boldsymbol{V}^{(k)}-\boldsymbol{Z}^{(k)})\right) \quad \text{where} \quad \boldsymbol{L}\boldsymbol{L}^T=\boldsymbol{I}+\rho\boldsymbol{D}^T\boldsymbol{D}$$

To solve the second subproblem, we note that it can be written as a proximal operator:

$$\begin{aligned}\underset{\boldsymbol{V}\in\mathbb{R}^{|\mathcal{E}|\times n}}{\arg\min}\ \lambda P(\boldsymbol{V};\boldsymbol{w},q)+\frac{\rho}{2}\left\|\boldsymbol{D}\boldsymbol{U}^{(k+1)}-\boldsymbol{V}+\boldsymbol{Z}^{(k)}\right\|_F^2 &= \underset{\boldsymbol{V}\in\mathbb{R}^{|\mathcal{E}|\times n}}{\arg\min}\ \frac{\lambda}{\rho}P(\boldsymbol{V};\boldsymbol{w},q)+\frac{\rho}{2}\left\|\boldsymbol{V}-\left(\boldsymbol{D}\boldsymbol{U}^{(k+1)}+\boldsymbol{Z}^{(k)}\right)\right\|_F^2 \\ &= \mathsf{prox}_{\frac{\lambda}{\rho}P(\cdot;\boldsymbol{w},q)}\left(\boldsymbol{D}\boldsymbol{U}^{(k+1)}+\boldsymbol{Z}^{(k)}\right)\end{aligned}$$

We note that, due to the row-wise structure of P, this proximal operator can be computed separately across the rows of its argument. In the cases $q = 1$ or $q = 2$, the proximal operator reduces to element-wise ($q = 1$) or group ($q = 2$) soft-thresholding row l at the level $w_l\lambda/\rho$. If $q = \infty$, Moreau's decomposition (Moreau, 1962) can be combined with the the efficient projection onto the ℓ_1 ball developed by Duchi et al. (2008) to evaluate the prox in $\mathcal{O}(p \log p)$ steps. For other values of q, an iterative algorithm must be used.

Several stopping criteria for the ADMM have been proposed in the literature. We have found a simple stopping rule based on the change in $\boldsymbol{U}$ being small sufficient in all cases. While some authors report speed-ups due to varying the ADMM relaxation parameter ρ, we have found that fixing ρ and re-using the Cholesky factor $\boldsymbol{L}$ to be more efficient. Combining these steps, we obtain Algorithm A1.

A.2 Algorithmic Regularization for Convex Clustering

In this section, we give a the full version of the `CARP` algorithm presented in Algorithm 2. `CARP` can be obtained from the standard ADMM for convex clustering (Algorithm A1) by replacing the inner ADMM loop with a single iteration. This modification gives `CARP` (Algorithm A2). As with Algorithm A1, we prefer to use a matrix formulation, instead of a fully vectorized formulation, to simplify the implementation and to take advantage of high-performance numerical linear algebra libraries.

Algorithm A1 Warm-Started ADMM for the Convex Clustering Problem (1)

1. Input:
   - Data Matrix: $\boldsymbol{X} \in \mathbb{R}^{n \times p}$
   - Weighted Directed Edge Set: $\mathcal{E} = \{(e_l, w_l)\}$
   - Relaxation Parameter: $\rho \in \mathbb{R}_{>0}$
   - Initial Regularization Parameter ϵ and Multiplicative Step-Size t
2. Precompute:
   - Difference Matrix: $\boldsymbol{D} \in \mathbb{R}^{|\mathcal{E}| \times n}$ where D_{ij} is 1 if edge i begins at node j, -1 if edge i ends at node j, and 0 otherwise
   - Cholesky Factor: $\boldsymbol{L} = \mathsf{chol}(\boldsymbol{I} + \rho \boldsymbol{D}^T \boldsymbol{D}) \in \mathbb{R}^{n \times n}$
3. Initialize:
   - $\boldsymbol{U}^{(0)} = \boldsymbol{X}$
   - $\boldsymbol{V}^{(0)} = \boldsymbol{Z}^{(0)} = \boldsymbol{D}\boldsymbol{X}$
   - $l = 0$, $\lambda_0 = \epsilon$, $k = 0$,
4. Repeat until $\|\boldsymbol{V}^{(k)}\| = 0$:
   - Repeat until convergence:
     (i) $\boldsymbol{U}^{(k+1)} = \boldsymbol{L}^{-T}\boldsymbol{L}^{-1}\left(\boldsymbol{X} + \rho \boldsymbol{D}^T(\boldsymbol{V}^{(k)} - \boldsymbol{Z}^{(k)}\right)$
     (ii) $\boldsymbol{V}^{(k+1)} = \mathsf{prox}_{\lambda_l/\rho\, P(\cdot;\boldsymbol{w},q)}\left(\boldsymbol{D}\boldsymbol{U}^{(k+1)} + \boldsymbol{Z}^{(k)}\right)$
     (iii) $\boldsymbol{Z}^{(k+1)} = \boldsymbol{Z}^{(k)} + \boldsymbol{D}\boldsymbol{U}^{(k+1)} - \boldsymbol{V}^{(k+1)}$
     (iv) $k := k + 1$
   - Store $\hat{\boldsymbol{U}}_{\lambda_l} = \boldsymbol{U}^{(k)}$
   - Update Regularization Parameter $l := l + 1$; $\lambda_l := \lambda_{l-1} * t$
5. Return $\left\{\hat{\boldsymbol{U}}_{\lambda_i}\right\}_{i=0}^{l-1}$ as the regularization path

Algorithm A2 `CARP`: Convex Clustering via Algorithmic Regularization Paths

1. Input:
   - Data Matrix: $\boldsymbol{X} \in \mathbb{R}^{n \times p}$
   - Weighted Directed Edge Set: $\mathcal{E} = \{(e_l, w_l)\}$
   - Relaxation Parameter: $\rho \in \mathbb{R}_{>0}$
   - Initial Regularization Parameter ϵ and Multiplicative Step-Size t
2. Precompute:
   - Difference Matrix: $\boldsymbol{D} \in \mathbb{R}^{|\mathcal{E}| \times n}$ where D_{ij} is 1 if edge i begins at node j, -1 if edge i ends at node j, and 0 otherwise
   - Cholesky Factor: $\boldsymbol{L} = \mathsf{chol}(\boldsymbol{I} + \rho \boldsymbol{D}^T \boldsymbol{D}) \in \mathbb{R}^{n \times n}$
3. Initialize:
   - $\boldsymbol{U}^{(0)} = \boldsymbol{X}$
   - $\boldsymbol{V}^{(0)} = \boldsymbol{Z}^{(0)} = \boldsymbol{D}\boldsymbol{X}$
   - $k = 0$, $\gamma^{(0)} = \epsilon$
4. Repeat until $\|\boldsymbol{V}^{(k)}\| = 0$:
   (i) $\boldsymbol{U}^{(k+1)} = \boldsymbol{L}^{-T}\boldsymbol{L}^{-1}\left(\boldsymbol{X} + \rho \boldsymbol{D}^T(\boldsymbol{V}^{(k)} - \boldsymbol{Z}^{(k)}\right)$
   (ii) $\boldsymbol{V}^{(k+1)} = \mathsf{prox}_{\gamma^{(k)}/\rho\, P(\cdot;\boldsymbol{w},q)}\left(\boldsymbol{D}\boldsymbol{U}^{(k+1)} + \boldsymbol{Z}^{(k)}\right)$
   (iii) $\boldsymbol{Z}^{(k+1)} = \boldsymbol{Z}^{(k)} + \boldsymbol{D}\boldsymbol{U}^{(k+1)} - \boldsymbol{V}^{(k+1)}$
   (iv) $k := k + 1$, $\gamma^{(k)} = \gamma^{(k-1)} * t$
5. Return $\left\{\boldsymbol{U}^{(k)}\right\}_{i=0}^{k}$ as the `CARP` algorithmic regularization path

A.3 AMA for Convex Clustering

In addition to the AMA, Chi and Lange (2015) also show that the convex clustering problem (1) can be efficiently solved using the Alternating Minimization Algorithm (AMA) of Tseng (1991). In our notation, the AMA becomes

$$\begin{aligned}
\boldsymbol{U}^{(k+1)} &= \underset{\boldsymbol{U} \in \mathbb{R}^{n \times p}}{\arg\min} \frac{1}{2}\|\boldsymbol{U} - \boldsymbol{X}\|_F^2 + \langle \boldsymbol{Z}^{(k)}, \boldsymbol{D}\boldsymbol{U} - \boldsymbol{V}^{(k)} \rangle \\
\boldsymbol{V}^{(k+1)} &= \underset{\boldsymbol{V} \in \mathbb{R}^{|\mathcal{E}| \times n}}{\arg\min} \lambda P(\boldsymbol{V}; \boldsymbol{w}, q) + \langle \boldsymbol{Z}^{(k)}, \boldsymbol{D}\boldsymbol{U}^{(k+1)} - \boldsymbol{V}^{(k)} \rangle + \frac{\rho}{2} \left\| \boldsymbol{D}\boldsymbol{U}^{(k+1)} - \boldsymbol{V} \right\|_F^2 \\
\boldsymbol{Z}^{(k+1)} &= \boldsymbol{Z}^{(k)} + \rho(\boldsymbol{D}\boldsymbol{U}^{(k+1)} - \boldsymbol{V}^{(k+1)})
\end{aligned}$$

(Note that we use the unscaled updates for $\boldsymbol{V}, \boldsymbol{Z}$ here as the AMA uses different values of the relaxation parameter in the $\boldsymbol{U}$ and $\boldsymbol{V}$ updates. In particular, this means that the dual variables $\boldsymbol{Z}$ from the ADMM are not the same as those from the AMA.) Simplifying these updates as before, the AMA becomes:

$$\begin{aligned}
\boldsymbol{U}^{(k+1)} &= \boldsymbol{X} - \boldsymbol{D}^T \boldsymbol{Z}^{(k)} \\
\boldsymbol{V}^{(k+1)} &= \mathsf{prox}_{\frac{\lambda}{\rho} P(\cdot; \boldsymbol{w}, q)} \left(\boldsymbol{D}\boldsymbol{U}^{(k+1)} + \boldsymbol{Z}^{(k)}/\rho \right) \\
\boldsymbol{Z}^{(k+1)} &= \boldsymbol{Z}^{(k)} + \rho(\boldsymbol{D}\boldsymbol{U}^{(k+1)} - \boldsymbol{V}^{(k+1)})
\end{aligned}$$

Chi and Lange (2015) note that a clever application of Moreau's decomposition (Moreau, 1962) allows the $\boldsymbol{V}$-updates to be elided and for the AMA to be simplified into a two-step scheme. The $\boldsymbol{V}^{(k)}$ iterates are key to dendrogram reconstruction, however, so such a simplification could not be used in an AMA-based `CARP` variant.

In our experiments, this elision is necessary for the AMA to outperform the ADMM and so, without it, the AMA does not appear to be a promising basis for an algorithmic regularization scheme. In general, the ADMM appears to converge more rapidly *per iteration* than the AMA, while the simplified AMA has much faster updates, allowing better overall

computational performance in a standard optimization scheme. Since CARP performs only a single iteration per regularization level, however, the faster per iteration convergence of the ADMM is more important to us than the faster calculation of the AMA.

Finally, Chi and Lange (2015) also discuss the use of accelerated variants of the ADMM and AMA (Goldstein et al., 2014) to improve convergence. Because CARP uses only a single iteration for each regularization level, it is not amenable to acceleration.

B Proof of Theorem 1

In this section we prove Theorem 1 on the Hausdorff convergence of CARP to the convex clustering regularization path. We begin with 3 technical lemmas which may be of independent interest: Lemma 1 provides a convergence rate for the optimization step embedded within a CARP iteration; Lemma 2 establishes a form of Lipschitz continuity for convex clustering regularization paths; Lemma 3 provides a global bound for the approximation error induced by CARP at any iteration. In one step, our results are stated and proven for CARP with an ℓ_2-fusion penalty, but can be easily extended to other ℓ_q-fusion penalties.

Lemma 1 (Q-Linear Error Decrease)**.** *At each iteration k, the* CARP *approximation error decreases by a factor $c < 1$ not depending on t or ϵ. That is,*

$$\|\boldsymbol{U}^{(k)} - \hat{\boldsymbol{U}}_{\gamma^{(k)}}\| + \|\boldsymbol{Z}^{(k)} - \hat{\boldsymbol{Z}}_{\gamma^{(k)}}\| < c\left[\|\boldsymbol{U}^{(k-1)} - \hat{\boldsymbol{U}}_{\gamma^{(k)}}\| + \|\boldsymbol{Z}^{(k-1)} - \hat{\boldsymbol{Z}}_{\gamma^{(k)}}\|\right]$$

for some c strictly less than 1.

Proof. By construction, each CARP step is a single iteration of the ADMM for the convex clustering problem (Algorithm A1) initialized at $(\boldsymbol{U}^{(k-1)}, \boldsymbol{V}^{(k-1)}, \boldsymbol{Z}^{(k-1)})$. Hence it suffices to analyze the convergence of the ADMM for the convex clustering problem and to establish linear convergence.

The convex clustering problem (1) is strongly convex due to squared Frobenius norm term. Linear convergence of the standard ADMM for strongly convex problems was first shown by Lions and Mercier (1979) and has since been refined by several other authors including Shi et al. (2014), Nishihara et al. (2015), Deng and Yin (2016), and Yang and Han (2016).

In vectorized form, with $\boldsymbol{x} = \text{vec}(\boldsymbol{X})$, $\boldsymbol{u} = \text{vec}(\boldsymbol{U})$, and $\boldsymbol{v} = \text{vec}(\boldsymbol{V})$, the convex clustering problem (1) can be expressed as:

$$\underset{\boldsymbol{u}\in\mathbb{R}^{np},\boldsymbol{v}\in\mathbb{R}^{|\mathcal{E}|p}}{\arg\min} \frac{\|\boldsymbol{x}-\boldsymbol{u}\|_2^2}{2} + \lambda\|\boldsymbol{v}\|_{\text{vec}(\ell_q)} \quad \text{subject to} \quad (\boldsymbol{I}\otimes\boldsymbol{D})\boldsymbol{u} = \boldsymbol{v}$$

where $\|\cdot\|_{\text{vec}(\ell_q)}$ is an appropriately vectorized version of the row-wise ℓ_q norm (a standard ℓ_1 norm in the case $q = 1$ and a mixed ℓ_q/ℓ_1 norm otherwise) and we have omitted the fusion weights for brevity.

In the notation of Hong and Luo (2017), the constraint matrix for the convex clustering problem is given by $\boldsymbol{E} = \begin{pmatrix} \boldsymbol{I}\otimes\boldsymbol{D} & -\boldsymbol{I} \end{pmatrix}$, for appropriately sized identity matrices, which is clearly row-independent, yielding linear convergence of the primal and dual variables at a rate $c_\lambda < 1$ which may depend on λ. (We do not need to verify their additional technical assumptions as we are only using a two-block ADMM instead of the more general multi-block ADMM which is the focus of their paper.) Taking $c = \sup_{\lambda\leq\lambda_{\max}} c_\lambda$, we observe that the `CARP` iterates are uniformly Q-linearly convergent at a rate c. □

Remark. Recently, Deng and Yin (2016) gave a readable and precise analysis of the linear convergence of the ADMM, including estimates of the convergence rate c. The specific proof technique they employ does not strictly apply to the convex clustering problem 1, however, as the $\boldsymbol{D}$ matrix is rank-deficient (excluding their Scenario 1) and the norm used for the fusion penalty is non-differentiable at the origin (excluding their Scenario 3). If an estimate of the convergence rate is required, the analysis of Deng and Yin (2016) can be applied to the convex clustering problem 1 by re-parameterizing the problem to address the rank-deficiency of $\boldsymbol{D}$. In particular, if the redundant rows of $\boldsymbol{D}$ are combined (eliminating the nullspace

of $\boldsymbol{D}$), the resulting matrix will be full-row rank, allowing Scenario 1 and Case 2 of Deng and Yin (2016) to be applied. This re-parameterization results in different split and dual variables ($\boldsymbol{V}$ and $\boldsymbol{Z}$, corresponding to the $\boldsymbol{D}$ matrix), however, so we do not pursue that approach here. The primal variable, $\boldsymbol{U}$, remains unchanged under this re-parameterization.

Lemma 2 (Lipschitz Continuity of Solution Paths)**.** $(\hat{\boldsymbol{U}}_\lambda, \hat{\boldsymbol{Z}}_\lambda)$ *is L-Lipschitz with respect to* λ*. That is,*

$$\|\hat{\boldsymbol{U}}_{\lambda_1} - \hat{\boldsymbol{U}}_{\lambda_2}\| + \|\hat{\boldsymbol{Z}}_{\lambda_1} - \hat{\boldsymbol{Z}}_{\lambda_2}\| \leq L * |\lambda_1 - \lambda_2|$$

for some $L > 0$*.*

We note that this not the only form of Lipschitz continuity commonly considered for regularized estimation problems. In particular, Lipschitz continuity of the *solution with respect to the data* is a key element of various consistency results, while Lipschitz continuity of the *objective function with respect to the parameters* is a key assumption used to prove convergence of many optimization schemes.

Proof. It suffices to prove Lipschitz continuity of $\hat{\boldsymbol{U}}_\lambda$ and $\hat{\boldsymbol{Z}}_\lambda$ separately and then take the sum of their Lipschitz moduli as the joint Lipschitz modulus.

We first show that $\hat{\boldsymbol{U}}_\lambda$ is Lipschitz. In vectorized form, the convex clustering problem is

$$\hat{\boldsymbol{u}}_\lambda = \underset{\boldsymbol{u} \in \mathbb{R}^{np}}{\arg\min} \frac{1}{2}\|\boldsymbol{u} - \boldsymbol{x}\|_2^2 + \lambda f_q(\tilde{\boldsymbol{D}}\boldsymbol{u})$$

where $\boldsymbol{u} = \text{vec}(\boldsymbol{U})$, $\boldsymbol{x} = \text{vec}(\boldsymbol{X})$, f_q is a convex function, and $\tilde{\boldsymbol{D}} = \boldsymbol{I} \otimes \boldsymbol{D}$ is a fixed matrix (*cf.* Tan and Witten, 2015).

The KKT conditions give

$$0 \in \boldsymbol{u}_\lambda - \boldsymbol{x} + \lambda \tilde{\boldsymbol{D}}^T \partial f_q(\tilde{\boldsymbol{D}}\boldsymbol{u}_\lambda)$$

where $\partial f_q(\cdot)$ is the subdifferential of f_q. Since f_q is convex, it is differentiable almost every-

where (Rockafellar, 1970, Theorem 25.5), so the following holds for almost all $\boldsymbol{u}_\lambda$:

$$0 = \boldsymbol{u}_\lambda - \boldsymbol{x} + \lambda \tilde{\boldsymbol{D}}^T f_q'(\tilde{\boldsymbol{D}}\boldsymbol{u}_\lambda)$$

Differentiating with respect to λ, we obtain (*c.f.* Rosset and Zhu, 2007)

$$\begin{aligned}
0 &= \boldsymbol{u}_\lambda - \boldsymbol{x} + \lambda \tilde{\boldsymbol{D}}^T f_q'(\tilde{\boldsymbol{D}}\boldsymbol{u}_\lambda) \\
\frac{\partial}{\partial \lambda}[0] &= \frac{\partial}{\partial \lambda}\left[\boldsymbol{u}_\lambda - \boldsymbol{x} + \lambda \tilde{\boldsymbol{D}}^T f_q'(\tilde{\boldsymbol{D}}\boldsymbol{u}_\lambda)\right] \\
0 &= \frac{\partial \boldsymbol{u}_\lambda}{\partial \lambda} - 0 + \lambda \frac{\partial}{\partial \lambda}\left[\tilde{\boldsymbol{D}}^T f_q'(\tilde{\boldsymbol{D}}\boldsymbol{u}_\lambda)\right] + \tilde{\boldsymbol{D}}^T f_q'(\tilde{\boldsymbol{D}}\boldsymbol{u}_\lambda) \\
0 &= \frac{\partial \boldsymbol{u}_\lambda}{\partial \lambda} + \lambda \tilde{\boldsymbol{D}}^T f_q''(\tilde{\boldsymbol{D}}\boldsymbol{u}_\lambda)\tilde{\boldsymbol{D}}\frac{\partial \boldsymbol{u}_\lambda}{\partial \lambda} + \tilde{\boldsymbol{D}}^T f_q'(\tilde{\boldsymbol{D}}\boldsymbol{u}_\lambda) \\
\implies \frac{\partial \boldsymbol{u}}{\partial \lambda} &= -[\boldsymbol{I} + \lambda \tilde{\boldsymbol{D}}^T f_q''(\tilde{\boldsymbol{D}}\boldsymbol{u})\tilde{\boldsymbol{D}}]^{-1}\boldsymbol{D}^T f_q'(\tilde{\boldsymbol{D}}\boldsymbol{u}).
\end{aligned}$$

Note that $\boldsymbol{u}_\lambda$ depends on λ so the chain rule must be used here. From here, we note

$$\left\|\frac{\partial \boldsymbol{u}_\lambda}{\partial \lambda}\right\|_\infty = \left\|-[\boldsymbol{I} + \lambda \tilde{\boldsymbol{D}}^T f_q''(\tilde{\boldsymbol{D}}\boldsymbol{u}_\lambda)\tilde{\boldsymbol{D}}]^{-1}\tilde{\boldsymbol{D}}^T f_q'(\tilde{\boldsymbol{D}}\boldsymbol{u}_\lambda)\right\|_\infty \leq \|-[\boldsymbol{I}+0]^{-1}\tilde{\boldsymbol{D}}^T f_q'(\tilde{\boldsymbol{D}}\boldsymbol{u}_\lambda)\|_\infty = \|\tilde{\boldsymbol{D}}^T f_q'(\tilde{\boldsymbol{D}}\boldsymbol{u}_\lambda)\|_\infty.$$

For the convex clustering problem, we recall that $f_q(\cdot)$ is a norm and hence has bounded gradient; hence $f_q'(\tilde{\boldsymbol{D}}\boldsymbol{u}_\lambda)$ is bounded so the gradient of the regularization path is bounded and exists almost everywhere. This implies that the regularization path is *piecewise* Lipschitz. Since the solution path is constant for $\lambda \geq \lambda_{\max}$ and is continuous (Chi and Lange, 2015, Proposition 2.1), the solution path is globally Lipschitz with a Lipschitz modulus equal to the maximum of the piecewise Lipschitz moduli.

A similar argument shows Lipschitz continuity of $\hat{\boldsymbol{Z}}_\lambda$ or one can use the relationships between $\hat{\boldsymbol{U}}_\lambda$ and $\hat{\boldsymbol{Z}}_\lambda$ discussed in Section 2.1 of Tan and Witten (2015).

□

Lemma 3 (Global Error Bound)**.** *The following error bound holds for all k:*

$$\|\boldsymbol{U}^{(k)} - \hat{\boldsymbol{U}}_{\gamma^{(k)}}\| + \|\boldsymbol{Z}^{(k)} - \hat{\boldsymbol{Z}}_{\gamma^{(k)}}\| \leq c^k L\epsilon + L(t-1)\epsilon t^k \sum_{i=1}^{k-1} \left(\frac{c}{t}\right)^i$$

Proof. Throughout, we let

$$\hat{\boldsymbol{\Upsilon}}_\lambda = \begin{pmatrix} \hat{\boldsymbol{U}}_\lambda \\ \hat{\boldsymbol{Z}}_\lambda \end{pmatrix} \text{ and } \hat{\boldsymbol{\Upsilon}}^{(k)} = \begin{pmatrix} \boldsymbol{U}^{(k)} \\ \boldsymbol{Z}^{(k)} \end{pmatrix}.$$

Our proof proceeds by induction on k. First note that, at initialization:

$$\|\boldsymbol{\Upsilon}^{(0)} - \hat{\boldsymbol{\Upsilon}}_\epsilon\| \leq L\epsilon$$

by Lemma 2.

Next, at $k = 1$, we note that

$$\|\boldsymbol{\Upsilon}^{(1)} - \hat{\boldsymbol{\Upsilon}}_{t\epsilon}\| \leq c\|\boldsymbol{\Upsilon}^{(0)} - \hat{\boldsymbol{\Upsilon}}_{t\epsilon}\|$$

by Lemma 1. We now the triangle inequality to split the right hand side:

$$\|\boldsymbol{\Upsilon}^{(0)} - \hat{\boldsymbol{\Upsilon}}_{t\epsilon}\| \leq \underbrace{\|\boldsymbol{\Upsilon}^{(0)} - \hat{\boldsymbol{\Upsilon}}_\epsilon\|}_{\text{RHS-1}} + \underbrace{\|\hat{\boldsymbol{\Upsilon}}_\epsilon - \hat{\boldsymbol{\Upsilon}}_{t\epsilon}\|}_{\text{RHS-2}}$$

From above, we have RHS-1 $\leq L\epsilon$. Using Lemma 2, RHS-2 can be bounded by

$$\|\hat{\boldsymbol{\Upsilon}}_\epsilon - \hat{\boldsymbol{\Upsilon}}_{t\epsilon}\| \leq L\,|t\epsilon - \epsilon| = L(t-1)\epsilon.$$

Putting these together, we get

$$\|\boldsymbol{\Upsilon}^{(1)} - \hat{\boldsymbol{\Upsilon}}_{t\epsilon}\| \leq c\,[\text{RHS-1} + \text{RHS-2}] \leq c\,[L\epsilon + L(t-1)\epsilon] = cLt\epsilon$$

Repeating this argument for $k = 2$, we see

$$\begin{aligned}
\|\boldsymbol{\Upsilon}^{(2)} - \hat{\boldsymbol{\Upsilon}}_{t^2\epsilon}\| &\leq c\|\boldsymbol{\Upsilon}^{(1)} - \hat{\boldsymbol{\Upsilon}}_{t^2\epsilon}\| \\
&\leq c\left[\|\boldsymbol{\Upsilon}^{(1)} - \hat{\boldsymbol{\Upsilon}}_{t\epsilon}\| + \|\hat{\boldsymbol{\Upsilon}}_{t\epsilon} - \hat{\boldsymbol{\Upsilon}}_{t^2\epsilon}\|\right] \\
&\leq c\left[cLt\epsilon + L\left|t^2\epsilon - t\epsilon\right|\right] \\
&= c^2Lt\epsilon + cL(t-1)\epsilon * t \\
&= c^2Lt\epsilon + L\epsilon(t-1)t^2 * \left(\frac{c}{t}\right) \\
&= c^2Lt\epsilon + L\epsilon(t-1)t^2 * \sum_{i=1}^{k-1}\left(\frac{c}{t}\right)^i
\end{aligned}$$

We use this as a base case for our inductive proof and prove the general case:

$$\begin{aligned}
\|\boldsymbol{\Upsilon}^{(k)} - \hat{\boldsymbol{\Upsilon}}_{t^k\epsilon}\| &\leq c\|\boldsymbol{\Upsilon}^{(k-1)} - \hat{\boldsymbol{\Upsilon}}_{t^k\epsilon}\| \\
&\leq c\left[\|\boldsymbol{\Upsilon}^{(k-1)} - \hat{\boldsymbol{\Upsilon}}_{t^{k-1}\epsilon}\| + \|\hat{\boldsymbol{\Upsilon}}_{t^{k-1}\epsilon} - \hat{\boldsymbol{\Upsilon}}_{t^k\epsilon}\|\right] \\
&\leq c\left[c^{k-1}Lt\epsilon + L\epsilon(t-1)t^{k-1}\sum_{i=1}^{k-2}\left(\frac{c}{t}\right)^i + L\left|t^k\epsilon - t^{k-1}\epsilon\right|\right] \\
&= c^kLt\epsilon + cL\epsilon(t-1)t^{k-1}\sum_{i=1}^{k-2}\left(\frac{c}{t}\right)^i + cL\epsilon(t^k - t^{k-1}) \\
&= c^kLt\epsilon + L\epsilon(t-1)t^k\left[\frac{c}{t}\sum_{i=1}^{k-2}\left(\frac{c}{t}\right)^i + \frac{c}{t}\right] \\
&= c^kLt\epsilon + L\epsilon(t-1)t^k\left[\sum_{i=2}^{k-1}\left(\frac{c}{t}\right)^i + \frac{c}{t}\right] \\
&= c^kLt\epsilon + L\epsilon(t-1)t^k\sum_{i=1}^{k-1}\left(\frac{c}{t}\right)^i
\end{aligned}$$

Expanding the definitions of $\boldsymbol{\Upsilon}^{(k)}, \hat{\boldsymbol{\Upsilon}}_\lambda$, we get the desired result.

□

With these results, we are now ready to prove Theorem 1:

Theorem 1. *As $(t, \epsilon) \to (1, 0)$, where t is the multiplicative step-size update and ϵ is the initial regularization level, the primal and dual* `CARP` *paths converge to the primal and dual convex clustering paths in the Hausdorff metric: that is,*

$$d_H(\{\boldsymbol{U}^{(k)}\}, \{\hat{\boldsymbol{U}}_\lambda\}) \equiv \max\left\{\sup_\lambda \inf_k \left\|\boldsymbol{U}^{(k)} - \hat{\boldsymbol{U}}_\lambda\right\|, \sup_k \inf_\lambda \left\|\boldsymbol{U}^{(k)} - \hat{\boldsymbol{U}}_\lambda\right\|\right\} \xrightarrow{(t,\epsilon)\to(1,0)} 0$$
$$d_H(\{\boldsymbol{Z}^{(k)}\}, \{\hat{\boldsymbol{Z}}_\lambda\}) \equiv \max\left\{\sup_\lambda \inf_k \left\|\boldsymbol{Z}^{(k)} - \hat{\boldsymbol{Z}}_\lambda\right\|, \sup_k \inf_\lambda \left\|\boldsymbol{Z}^{(k)} - \hat{\boldsymbol{Z}}_\lambda\right\|\right\} \xrightarrow{(t,\epsilon)\to(1,0)} 0$$

where $\boldsymbol{U}^{(k)}, \boldsymbol{Z}^{(k)}$ are the values of the k^{th} `CARP` *iterate and $\hat{\boldsymbol{U}}_\lambda, \hat{\boldsymbol{Z}}_\lambda$ are the exact solutions to the convex clustering problem (1) and its dual at λ.*

Proof of Theorem 1. It suffices to show that $\{\boldsymbol{\Upsilon}^{(k)}\}, \{\hat{\boldsymbol{\Upsilon}}_\lambda\}$ converge in the Hausdorff metric to show that the primal and dual paths converge separately. We break our proof into three steps:

i. $\sup_\lambda \inf_k \left\|\boldsymbol{\Upsilon}^{(k)} - \hat{\boldsymbol{\Upsilon}}_\lambda\right\| \to 0$;

ii. ϵt^{k^*} remains bounded as t, ϵ decrease and k^* increases, where k^* is the iteration at which `CARP` terminates; and

iii. $\sup_k \inf_\lambda \left\|\boldsymbol{\Upsilon}^{(k)} - \hat{\boldsymbol{\Upsilon}}_\lambda\right\| \to 0$.

Together, these give the desired result.

Step i. We first show that

$$\sup_\lambda \inf_k \left\|\boldsymbol{\Upsilon}^{(k)} - \hat{\boldsymbol{\Upsilon}}_\lambda\right\|$$

tends to zero. We begin by fixing temporarily λ and bounding

$$\inf_k \left\|\boldsymbol{\Upsilon}^{(k)} - \hat{\boldsymbol{\Upsilon}}_\lambda\right\|$$

The infimum over all k is less than the distance at any particular k, so it suffices to choose a value of k which gives convergence to 0. Let $\tilde{k}$ be the value of k which gives the closest

value of $\gamma^{(k)}$ to λ along the `CARP` path; and let $\tilde{\lambda} = \gamma^{(\tilde{k})} = \epsilon t^{\tilde{k}}$. That is,

$$\tilde{k} = \arg\min_{k} |\gamma^{(k)} - \lambda| \quad \text{and } \tilde{\lambda} = \gamma^{(\tilde{k})}$$

Then

$$\inf_{k} \left\| \boldsymbol{\Upsilon}^{(k)} - \hat{\boldsymbol{\Upsilon}}_{\lambda} \right\| \leq \|\boldsymbol{\Upsilon}^{(\tilde{k})} - \hat{\boldsymbol{\Upsilon}}_{\lambda}\| \leq \underbrace{\|\boldsymbol{\Upsilon}^{(\tilde{k})} - \hat{\boldsymbol{\Upsilon}}_{\tilde{\lambda}}\|}_{\text{RHS-1}} + \underbrace{\|\hat{\boldsymbol{\Upsilon}}_{\tilde{\lambda}} - \hat{\boldsymbol{\Upsilon}}_{\lambda}\|}_{\text{RHS-2}}$$

Using Lemma 2, we can bound RHS-2 as

$$\text{RHS-2} \leq L|\tilde{\lambda} - \lambda| \leq L|\gamma^{(\tilde{k}+1)} - \gamma^{(\tilde{k}-1)}| = L * \epsilon t^{\tilde{k}-1} * [t^2 - 1] = L * \epsilon t^{\tilde{k}-1} * [t^2 - 1] \leq L * \lambda_{\max} * [t^2 - 1]$$

Using Lemma 3, we can bound RHS-1 as

$$\begin{aligned} \text{RHS-1} &= \|\boldsymbol{\Upsilon}^{(\tilde{k})} - \hat{\boldsymbol{\Upsilon}}_{\tilde{\lambda}}\| \\ &= \|\boldsymbol{\Upsilon}^{(\tilde{k})} - \hat{\boldsymbol{\Upsilon}}_{\gamma^{(\tilde{k})}}\| \\ &= c^{\tilde{k}} L\epsilon + L(t-1) * \epsilon t^{\tilde{k}} \sum_{i=1}^{k-1} \left(\frac{c}{1+t} \right)^{i} \leq c^{\tilde{k}} L\epsilon + L(t-1) * \epsilon t^{\tilde{k}} * C \end{aligned} \tag{A1}$$

where $C = \sum_{i=1}^{\infty} \left(\frac{c}{1+t}\right)^{i}$ is large but finite. Since $c < 1$ and $\tilde{\lambda} = \epsilon t^{\tilde{k}} \leq \lambda_{\max}$, we can replace the k-dependent quantities to get

$$\text{RHS-1} = \|\boldsymbol{\Upsilon}^{(\tilde{k})} - \hat{\boldsymbol{\Upsilon}}^{\tilde{\lambda}}\| \leq L\epsilon + C * L(t-1) * \lambda_{\max}$$

Putting these together, we have

$$\inf_{k} \|\boldsymbol{\Upsilon}^{(k)} - \hat{\boldsymbol{\Upsilon}}_{\lambda}\| \leq \text{RHS-1} + \text{RHS-2} \leq L\epsilon + C * L(t-1) * \lambda_{\max} + L * \lambda_{\max} * [t^2 - 1]$$

Since the right-hand side doesnt' depend on λ, we have

$$\sup_{\lambda} \inf_{k} \|\boldsymbol{\Upsilon}^{(k)} - \hat{\boldsymbol{\Upsilon}}_{\lambda}\| \leq L\epsilon + C * L(t-1) * \lambda_{\max} + L * \lambda_{\max} * [t^2 - 1]$$

As $(t, \epsilon) \to (1, 0)$, we have that the right-hand side converges to zero and hence

$$\sup_{\lambda} \inf_{k} \|\boldsymbol{\Upsilon}^{(k)} - \hat{\boldsymbol{\Upsilon}}_{\lambda}\| \to 0$$

as desired.

Step ii. Before showing the other half of Hausdorff convergence, we pause to prove an intermediate result: As $(t, \epsilon) \to (1, 0)$, ϵt^{k^*} remains bounded, where $k^* = k^*(t, \epsilon)$ is the iteration at which `CARP` halts. For this step, we specialize to the ℓ_2-case for concreteness, though our results are easily generalized.

`CARP` terminates when $\|\boldsymbol{V}^{(k+1)}\|_{\infty,q} = \max_{i,j} \|\boldsymbol{U}_{i\cdot}^{(k+1)} - \boldsymbol{U}_{j\cdot}^{(k+1)}\|_q = 0$; that is, `CARP` terminates when all of the pairwise differences have gone to zero and the data has been fused into a single cluster.

Note that the update

$$\boldsymbol{V}_i^{(k+1)} = \left[1 - \frac{w_i \gamma^{(k)}}{\|(\boldsymbol{D}\boldsymbol{U}^{(k+1)} + \boldsymbol{Z}^{(k)})_i\|_2}\right] (\boldsymbol{D}\boldsymbol{U}^{(k+1)} + \boldsymbol{Z}^{(k)})_i$$

will set $\boldsymbol{V}_i^{(k+1)}$ to zero when

$$\|(\boldsymbol{D}\boldsymbol{U}^{(k+1)} + \boldsymbol{Z}^{(k)})_i\|_2 < w_i \gamma^{(k)}$$

Letting (j, k) be the endpoints of edge i, we find

$$\begin{aligned}\|(\boldsymbol{D}\boldsymbol{U}^{(k+1)} + \boldsymbol{Z}^{(k)})_i\|_2 &= \|\boldsymbol{U}_{j\cdot}^{(k+1)} - \boldsymbol{U}_{k\cdot}^{(k+1)} + \boldsymbol{Z}_{i\cdot}^{(k)}\|_2 \\ &= \left\|(\boldsymbol{U}_{j\cdot}^{(k+1)} - \overline{\boldsymbol{x}}) - (\boldsymbol{U}_{k\cdot}^{(k+1)} - \overline{\boldsymbol{x}}) + \boldsymbol{Z}_{i\cdot}^{(k)}\right\|_2 \\ &\leq \left\|\boldsymbol{U}_{j\cdot}^{(k+1)} - \overline{\boldsymbol{x}}\right\|_2 + \left\|\boldsymbol{U}_{k\cdot}^{(k+1)} - \overline{\boldsymbol{x}}\right\|_2 + \left\|\boldsymbol{Z}_{i\cdot}^{(k)}\right\|_2\end{aligned}$$

where $\overline{\boldsymbol{x}}$ is the column-wise mean of $\boldsymbol{X}$.

Our strategy will be to show that this quantity is less that $w_i\gamma^{(k)}$ for some $k > k^*$ small enough that ϵt^k remains bounded and hence ϵt^{k^*} remains bounded. Let

$$\tilde{k} = \left\lceil \frac{\log(\lambda_{\max}/\epsilon)}{\log(t)} \right\rceil = \lceil \log_t(\lambda_{\max}/\epsilon) \rceil$$

be the first value of k such that $\gamma^{(k)} > \lambda_{\max}$, *i.e.*, the value of λ such that all of the pairwise differences have gone to zero and the data has been fused into a single cluster in the regularization path ($\tilde{k}$ is to the regularization path as k^* is to the `CARP` path).

Using the bound from Equation (A1), we have that

$$\|\boldsymbol{U}_{j\cdot}^{(k+1)} - \overline{\boldsymbol{x}}\| = \|\boldsymbol{u}_l^{(k+1)} - (\hat{\boldsymbol{U}}_{\lambda_{\max}})_{j\cdot}\| < L\epsilon + C * L(t-1) * \lambda_{\max}$$

so

$$\|\boldsymbol{U}_{j\cdot}^{(k+1)} - \overline{\boldsymbol{x}}\| + \|\boldsymbol{U}_{k\cdot}^{(k+1)} - \overline{\boldsymbol{x}}\| < 2\left(L\epsilon + C * L(t-1) * \lambda_{\max}\right)$$

Bounding $\|\boldsymbol{Z}_i^{(k)}\|_2$ is more subtle, but a rough bound can be obtained again using Equation (A1) to obtain:

$$\|\boldsymbol{Z}_{i\cdot}^{(k)} - (\hat{\boldsymbol{Z}}_{\lambda_{\max}})_{i\cdot}\|_2 < L\epsilon + C * L(t-1) * \lambda_{\max}$$

so

$$\|\boldsymbol{Z}_{i\cdot}^{(k)}\| \leq \|(\hat{\boldsymbol{Z}}_{\lambda_{\max}})_{i\cdot}\|_2 + L\epsilon + C * L(t-1) * \lambda_{\max}$$

Putting these together, we obtain

$$\|\boldsymbol{u}_l^{(k+1)} - \boldsymbol{u}_m^{(k+1)} - \boldsymbol{z}_{l,m}^{(k)}\|_2 \leq \|(\hat{\boldsymbol{z}}_{\lambda_{\max}})_{l,m}\|_2 + 3\left(L\epsilon + C * L(t-1) * \lambda_{\max}\right)$$

To stop, we require that

$$\|(\hat{\boldsymbol{Z}}_{\lambda_{\max}})_{i\cdot}\|_2 + 3\left(L\epsilon + C * L(t-1) * \lambda_{\max}\right) < w_{l,m} \underbrace{\epsilon t^k}_{\gamma^{(k)}}$$

which occurs when

$$k > \log_t \frac{\|(\hat{\boldsymbol{Z}}_{\lambda_{\max}})_{i\cdot}\|_2 + 3(L\epsilon + C * L(t-1) * \lambda_{\max})}{w_{l,m}\epsilon}$$

Taking the max over all (l, m)-pairs we find

$$k^* \leq \max_{l,m} \log_t \frac{\|(\hat{\boldsymbol{z}}_{\lambda_{\max}})_{l,m}\|_2 + 3(L\epsilon + C * L(t-1) * \lambda_{\max})}{w_{l,m}\epsilon}$$

Hence it suffices to note

$$\epsilon t^{k^*} \leq \epsilon t^{\max_i \log_t \frac{\|(\hat{\boldsymbol{Z}}_{\lambda_{\max}})_{i\cdot}\|_2 + 3(L\epsilon + C * L(t-1) * \lambda_{\max})}{w_i \epsilon}} \leq \max_i \frac{\|(\hat{\boldsymbol{Z}}_{\lambda_{\max}})_{i\cdot}\|_2 + 3(L\epsilon + C * L(t-1) * \lambda_{\max})}{w_i}$$

which clearly remains bounded as $t, \epsilon \to (1, 0)$.

Step iii. With this result in hand, the proof is similar to the first half. Again, we can invoke Lemma 3 to find that

$$\inf_{\lambda} \|\boldsymbol{\Upsilon}^{(k)} - \hat{\boldsymbol{\Upsilon}}_{\lambda}\| \leq \|\boldsymbol{\Upsilon}^{(k)} - \hat{\boldsymbol{\Upsilon}}_{\epsilon t^k}\| \leq c^k L\epsilon + CL * (t-1) * \epsilon t^k$$

With the result from above, ϵt^k remains bounded above by some $B < \infty$, so

$$\sup_k \inf_\lambda \|\boldsymbol{\Upsilon}^{(k)} - \hat{\boldsymbol{\Upsilon}}_\lambda\| = \sup_{1 \le k \le k^*} \inf_\lambda \|\boldsymbol{\Upsilon}^{(k)} - \hat{\boldsymbol{\Upsilon}}_\lambda\| \le \sup_{1 \le k \le k^*} c^k L\epsilon + CL*(t-1)*\epsilon t^k \le L\epsilon + CL*(t-1)*B$$

As $(t, \epsilon) \to (1, 0)$, the right hand-side goes to zero so

$$\sup_k \inf_\lambda \|\boldsymbol{\Upsilon}^{(k)} - \hat{\boldsymbol{\Upsilon}}_\lambda\| \to 0$$

Combining this with step i, we have

$$d_H(\{\boldsymbol{\Upsilon}^{(k)}\}, \{\hat{\boldsymbol{\Upsilon}}_\lambda\}) = \max \left\{ \sup_\lambda \inf_k \left\| \boldsymbol{\Upsilon}^{(k)} - \hat{\boldsymbol{\Upsilon}}_\lambda \right\|, \sup_k \inf_\lambda \left\| \boldsymbol{\Upsilon}^{(k)} - \hat{\boldsymbol{\Upsilon}}_\lambda \right\| \right\} \xrightarrow{(t,\epsilon) \to (1,0)} 0$$

as desired. □

C CBASS: Algorithmic Regularization Paths for Convex Bi-Clustering

Having explored the computational, theoretical, and practical advantages of CARP, we now turn to the closely related problem of bi-clustering. *Bi-clustering* refers to the simultaneous clustering of rows and columns. Building on the convex clustering formulation (1), Chi et al. (2017) propose the following convex formulation of bi-clustering:

$$\hat{\boldsymbol{U}}_\lambda = \underset{\boldsymbol{U} \in \mathbb{R}^{n \times p}}{\arg\min} \frac{1}{2} \|\boldsymbol{U} - \boldsymbol{X}\|_F^2 + \lambda \left(\sum_{\substack{i,j=1 \\ i \neq j}}^{n} w_{ij} \|\boldsymbol{U}_{i\cdot} - \boldsymbol{U}_{j\cdot}\|_q + \sum_{\substack{k,l=1 \\ k \neq l}}^{p} \tilde{w}_{kl} \|\boldsymbol{U}_{\cdot k} - \boldsymbol{U}_{\cdot l}\|_q \right). \tag{A2}$$

The second penalty term induces row fusions, similarly to how the first term induces column fusions. The resulting $\hat{\boldsymbol{U}}_\lambda$ has a 'checkerboard' pattern where groups of rows and columns

are clustered together. Note that for bi-clustering the centroids are *scalars* instead of vectors as in the clustering case.

Despite their relatively similar appearances, the convex bi-clustering problem (A2) is significantly more complicated than the convex clustering problem (1) and cannot be directly solved directly using an operator splitting method. Chi et al. (2017) propose the use of the *Dykstra-Like Proximal Algorithm* (DLPA) of Bauschke and Combettes (2008) to solve the convex bi-clustering problem (A2) and refer to the resulting algorithm as `COBRA` (**Co**nvex **B**i-Cluste**R**ing **A**lgorithm). `COBRA` works by alternating solving row- and column-wise convex clustering problems until convergence. As with convex clustering, calculating the bi-clustering solution path with sufficient accuracy to accurately reconstruct both row and column dendrograms poses significant computational burden, which is exacerbated by `COBRA`'s requirement to evaluate several convex clustering subproblems for each value of λ. While `CARP` could be used to solve each subproblem quickly, we would still have to run `CARP` many times, incurring a non-trivial total cost.

Instead, we apply the technique of algorithmic regularization to `COBRA` directly: we take only a single DLPA step and, within that step, we take only a single ADMM step for each of the row- and column-subproblems. We refer to the resulting algorithm as `CBASS`– **C**onvex **B**i-clustering via **A**lgorithmic Regularization with **S**mall **S**teps. Details of the `CBASS` algorithm are given in Algorithm A6 below. Our `clustRviz` software implements `CBASS` with and without a back-tracking step to ensure exact recovery or both the row- and column-dendrograms.

C.1 Algorithms for Convex Bi-Clustering

The DLPA can be used to solve problems of the form

$$\mathsf{prox}_{(f+g)(\cdot)}(\boldsymbol{r}) = \arg\min_{\boldsymbol{x}} \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{r}\|_2^2 + f(\boldsymbol{x}) + g(\boldsymbol{x})$$

where f and g are proximable but $f+g$ is not using the following iterative algorithm (see also Algorithm 10.18 in Combettes and Pesquet (2011)):

Algorithm A3 DLPA: Dykstra-Like Proximal Algorithm

1. Initialize: $\boldsymbol{x}^{(0)} = \boldsymbol{r}$, $\boldsymbol{p}^{(0)} = \boldsymbol{q}^{(0)} = 0$, $k = 0$
2. Repeat until convergence:
   - $\boldsymbol{y} = \mathsf{prox}_f(\boldsymbol{x}^{(k)} + \boldsymbol{p}^{(k)})$
   - $\boldsymbol{p}^{(k+1)} = \boldsymbol{p}^{(k)} + \boldsymbol{x}^{(k)} - \boldsymbol{y}$
   - $\boldsymbol{x}^{(k+1)} = \mathsf{prox}_g(\boldsymbol{y}^{(k+1)} + \boldsymbol{q}^{(k)})$
   - $\boldsymbol{q}^{(k+1)} = \boldsymbol{q}^{(k)} + \boldsymbol{y} - \boldsymbol{x}^{(k+1)}$
   - $k := k + 1$
3. Return $\boldsymbol{x}^{(k)}$

To apply Algorithm A3 to convex bi-clustering (A2), we note that the problem can be rewritten as:

$$\underset{\boldsymbol{U}\in\mathbb{R}^{n\times p}}{\arg\min}\ \frac{1}{2}\|\boldsymbol{U}-\boldsymbol{X}\|_F^2 + \underbrace{\lambda\left(\sum_{(e_l,w_l)\in\mathcal{E}_{\text{row}}} w_l\|(\boldsymbol{D}_{\text{row}}\boldsymbol{U})_{l\cdot}\|_q\right)}_{f(\boldsymbol{U})=P_{\text{row}}(\boldsymbol{U};\boldsymbol{w}_{\text{row}},q)} + \underbrace{\lambda\left(\sum_{(e_l,w_l)\in\mathcal{E}_{\text{col}}} w_l\|(\boldsymbol{U}\boldsymbol{D}_{\text{col}})_{\cdot l}\|_q\right)}_{g(\boldsymbol{U})=P_{\text{col}}(\boldsymbol{U};\boldsymbol{w}_{\text{col}},q)}$$

and apply the DLPA with $f(\boldsymbol{U}) = P_{\text{row}}(\boldsymbol{U};\boldsymbol{w}_{\text{row}},q)$ and $g(\boldsymbol{U}) = P_{\text{col}}(\boldsymbol{U};\boldsymbol{w}_{\text{col}},q)$. We note that prox_f is a standard convex clustering problem and can be evaluated using the ADMM or AMA approaches described above. To evaluate prox_g, we note that $\|(\boldsymbol{U}\boldsymbol{D}_{\text{col}})_{l\cdot}\|_q = \|(\boldsymbol{D}_{\text{col}}^T\boldsymbol{U}^T)_{\cdot l}\|_q$ so we simply need to perform standard convex clustering on transposed data. The DLPA then becomes:

Algorithm A4 DLPA for Convex Bi-Clustering

1. Initialize: $\boldsymbol{U}^{(0)} = \boldsymbol{X}$, $\boldsymbol{P}^{(0)} = \boldsymbol{Q}^{(0)} = 0$, $k = 0$
2. Repeat until convergence:
   - $\boldsymbol{T} = \mathsf{Convex\text{-}Clustering}(\boldsymbol{U}^{(k)} + \boldsymbol{P}^{(k)};\mathcal{E}_{\text{row}})$
   - $\boldsymbol{P}^{(k+1)} = \boldsymbol{P}^{(k)} + \boldsymbol{X}^{(k)} - \boldsymbol{T}$
   - $\boldsymbol{U}^{(k+1)} = \mathsf{Convex\text{-}Clustering}((\boldsymbol{Q}^{(k)} + \boldsymbol{T})^T;\mathcal{E}_{\text{col}})^T$
   - $\boldsymbol{Q}^{(k+1)} = \boldsymbol{Q}^{(k)} + \boldsymbol{T} - \boldsymbol{U}^{(k+1)}$
   - $k := k + 1$
3. Return $\boldsymbol{U}^{(k)}$

Expanding the Convex-Clustering steps with the ADMM from Algorithm A1 yields Algorithm

A5. To obtain `CBASS` from Algorithm A5, we replace the inner row- and column-subproblem loops with a single iteration of the convex clustering ADMM. Additionally, we do not reset the auxiliary $\boldsymbol{U}, \boldsymbol{P}, \boldsymbol{Q}$ variables, instead carrying forward their values from each `CBASS` iteration to the next. These two modifications yield `CBASS` (Algorithm A6).

Algorithm A5 Warm-Started DLPA + ADMM for the Convex Bi-Clustering Problem (A2)

1. Input:
   - Data Matrix: $\boldsymbol{X} \in \mathbb{R}^{n \times p}$
   - Weighted Directed Edge Sets: $\mathcal{E}_{\text{row}} = \{(e_l, w_l)\}$, $\mathcal{E}_{\text{col}} = \{(e_l, w_l)\}$
   - Relaxation Parameter: $\rho \in \mathbb{R}_{>0}$
   - Initial Regularization Parameter ϵ and Multiplicative Step-Size t
2. Precompute:
   - Difference Matrices: $\boldsymbol{D}_{\text{row}} \in \mathbb{R}^{|\mathcal{E}_{\text{row}}| \times n}$ and $\boldsymbol{D}_{\text{col}} \in \mathbb{R}^{p \times |\mathcal{E}_{\text{col}}|}$
   - Cholesky Factors: $\boldsymbol{L}_{\text{row}} = \mathsf{chol}(\boldsymbol{I} + \rho \boldsymbol{D}_{\text{row}}^T \boldsymbol{D}_{\text{row}}) \in \mathbb{R}^{n \times n}$ and $\boldsymbol{L}_{\text{col}} = \mathsf{chol}(\boldsymbol{I} + \rho \boldsymbol{D}_{\text{col}} \boldsymbol{D}_{\text{row}}^T) \in \mathbb{R}^{p \times p}$
3. Initialize:
   - $\boldsymbol{U}^{(0)} = \boldsymbol{X}$
   - $\boldsymbol{V}_{\text{row}}^{(0)} = \boldsymbol{Z}_{\text{row}}^{(0)} = \boldsymbol{D}_{\text{row}} \boldsymbol{X}$
   - $\boldsymbol{V}_{\text{col}}^{(0)} = \boldsymbol{Z}_{\text{col}}^{(0)} = (\boldsymbol{X} \boldsymbol{D}_{\text{col}})^T = \boldsymbol{D}_{\text{col}}^T \boldsymbol{X}^T$
   - $\boldsymbol{P}^{(0)} = \boldsymbol{Q}^{(0)} = 0$
   - $l = 0$, $\lambda_0 = \epsilon$, $k = 0$,
4. Repeat until $\|\boldsymbol{V}_{\text{row}}^{(k)}\| = \|\boldsymbol{V}_{\text{col}}^{(k)}\| = 0$:
   - Repeat Until Convergence:
     - Row Sub-Problem – Repeat Until Convergence:
       (i) $\boldsymbol{T} = \boldsymbol{L}_{\text{row}}^{-T} \boldsymbol{L}_{\text{row}}^{-1} \left(\boldsymbol{U}^{(k)} + \boldsymbol{P}^{(k)} + \rho \boldsymbol{D}_{\text{row}}^T (\boldsymbol{V}_{\text{row}}^{(k)} - \boldsymbol{Z}_{\text{row}}^{(k)} \right)$
       (ii) $\boldsymbol{V}_{\text{row}}^{(k+1)} = \mathsf{prox}_{\lambda_l / \rho \, P(\cdot; \boldsymbol{w}_{\text{row}}, q)} \left(\boldsymbol{D}_{\text{row}} \boldsymbol{T} + \boldsymbol{Z}_{\text{row}}^{(k)} \right)$
       (iii) $\boldsymbol{Z}_{\text{row}}^{(k+1)} = \boldsymbol{Z}_{\text{row}}^{(k)} + \boldsymbol{D}\boldsymbol{T} - \boldsymbol{V}_{\text{row}}^{(k+1)}$
     - $\boldsymbol{P}^{(k+1)} = \boldsymbol{P}^{(k)} + \boldsymbol{U}^{(k-1)} - \boldsymbol{T}$
     - Column Sub-Problem – Repeat Until Convergence:
       (i) $\boldsymbol{S} = \boldsymbol{L}_{\text{col}}^{-T} \boldsymbol{L}^{-1} \left((\boldsymbol{T} + \boldsymbol{Q}^{(k)})^T + \rho \boldsymbol{D}_{\text{col}} (\boldsymbol{V}_{\text{col}}^{(k)} - \boldsymbol{Z}_{\text{col}}^{(k)} \right)$
       (ii) $\boldsymbol{V}_{\text{col}}^{(k+1)} = \mathsf{prox}_{\lambda_l / \rho \, P(\cdot; \boldsymbol{w}_{\text{col}}, q)} \left(\boldsymbol{D}_{\text{col}}^T \boldsymbol{S} + \boldsymbol{Z}_{\text{col}}^{(k)} \right)$
       (iii) $\boldsymbol{Z}_{\text{col}}^{(k+1)} = \boldsymbol{Z}^{(k)} + \boldsymbol{D}_{\text{col}}^T \boldsymbol{S} - \boldsymbol{V}_{\text{col}}^{(k+1)}$
     - $\boldsymbol{U}^{(k+1)} = \boldsymbol{S}^T$
     - $\boldsymbol{Q}^{(k+1)} = \boldsymbol{Q}^{(k)} + \boldsymbol{T} - \boldsymbol{U}^{(k+1)}$
     - $k := k + 1$
   - Store $\hat{\boldsymbol{U}}_{\lambda_l} = \boldsymbol{U}^{(k)}$
   - Reset Auxiliary Variables: $\boldsymbol{U}^{(k+1)} = \boldsymbol{X}$, $\boldsymbol{P}^{(k+1)} = \boldsymbol{Q}^{(k+1)} = 0$
   - Update Regularization Parameter $\lambda_l := \lambda_{l-1} * t$, $l := l + 1$
5. Return $\left\{ \hat{\boldsymbol{U}}_{\lambda_i} \right\}_{i=0}^{l-1}$ as the regularization path

Algorithm A6 `CBASS`: Convex Bi-Clustering via Algorithmic regularization with Small Steps

1. Input:
   - Data Matrix: $\boldsymbol{X} \in \mathbb{R}^{n \times p}$
   - Weighted Directed Edge Sets: $\mathcal{E}_{\text{row}} = \{(e_l, w_l)\}$, $\mathcal{E}_{\text{col}} = \{(e_l, w_l)\}$
   - Relaxation Parameter: $\rho \in \mathbb{R}_{>0}$
   - Initial Regularization Parameter ϵ and Multiplicative Step-Size t
2. Precompute:
   - Difference Matrices: $\boldsymbol{D}_{\text{row}} \in \mathbb{R}^{|\mathcal{E}_{\text{row}}| \times n}$ and $\boldsymbol{D}_{\text{col}} \in \mathbb{R}^{p \times |\mathcal{E}_{\text{col}}|}$
   - Cholesky Factors: $\boldsymbol{L}_{\text{row}} = \mathsf{chol}(\boldsymbol{I} + \rho \boldsymbol{D}_{\text{row}}^T \boldsymbol{D}_{\text{row}}) \in \mathbb{R}^{n \times n}$ and $\boldsymbol{L}_{\text{col}} = \mathsf{chol}(\boldsymbol{I} + \rho \boldsymbol{D}_{\text{col}} \boldsymbol{D}_{\text{row}}^T) \in \mathbb{R}^{p \times p}$
3. Initialize:
   - $\boldsymbol{U}^{(0)} = \boldsymbol{X}$
   - $\boldsymbol{V}_{\text{row}}^{(0)} = \boldsymbol{Z}_{\text{row}}^{(0)} = \boldsymbol{D}_{\text{row}} \boldsymbol{X}$
   - $\boldsymbol{V}_{\text{col}}^{(0)} = \boldsymbol{Z}_{\text{col}}^{(0)} = (\boldsymbol{X} \boldsymbol{D}_{\text{col}})^T = \boldsymbol{D}_{\text{col}}^T \boldsymbol{X}^T$
   - $\boldsymbol{P}^{(0)} = \boldsymbol{Q}^{(0)} = 0$
   - $k = 0$, $\gamma^{(0)} = \epsilon$
4. Repeat until $\|\boldsymbol{V}_{\text{row}}^{(k)}\| = \|\boldsymbol{V}_{\text{col}}^{(k)}\| = 0$:
   - Row Updates:
     (i) $\boldsymbol{T} = \boldsymbol{L}_{\text{row}}^{-T} \boldsymbol{L}_{\text{row}}^{-1} \left(\boldsymbol{U}^{(k)} + \boldsymbol{P}^{(k)} + \rho \boldsymbol{D}_{\text{row}}^T (\boldsymbol{V}_{\text{row}}^{(k)} - \boldsymbol{Z}_{\text{row}}^{(k)} \right)$
     (ii) $\boldsymbol{V}^{(k+1)} = \mathsf{prox}_{\gamma^{(k)}/\rho \, P(\cdot; \boldsymbol{w}_{\text{row}}, q)} \left(\boldsymbol{D}_{\text{row}} \boldsymbol{T} + \boldsymbol{Z}^{(k)} \right)$
     (iii) $\boldsymbol{Z}_{\text{row}}^{(k+1)} = \boldsymbol{Z}_{\text{row}}^{(k)} + \boldsymbol{D}_{\text{row}} \boldsymbol{T} - \boldsymbol{V}_{\text{row}}^{(k+1)}$
   - $\boldsymbol{P}^{(k+1)} = \boldsymbol{P}^{(k)} + \boldsymbol{U}^{(k-1)} - \boldsymbol{T}$
   - Column Updates:
     (i) $\boldsymbol{S} = \boldsymbol{L}_{\text{col}}^{-T} \boldsymbol{L}^{-1} \left((\boldsymbol{T} + \boldsymbol{Q}^{(k)})^T + \rho \boldsymbol{D}_{\text{col}} (\boldsymbol{V}_{\text{col}}^{(k)} - \boldsymbol{Z}_{\text{col}}^{(k)} \right)$
     (ii) $\boldsymbol{V}_{\text{col}}^{(k+1)} = \mathsf{prox}_{\gamma^{(k)}/\rho \, P(\cdot; \boldsymbol{w}_{\text{col}}, q)} \left(\boldsymbol{D}_{\text{col}}^T \boldsymbol{S} + \boldsymbol{Z}_{\text{col}}^{(k)} \right)$
     (iii) $\boldsymbol{Z}_{\text{col}}^{(k+1)} = \boldsymbol{Z}^{(k)} + \boldsymbol{D}_{\text{col}}^T \boldsymbol{S} - \boldsymbol{V}_{\text{col}}^{(k+1)}$
   - $\boldsymbol{U}^{(k+1)} = \boldsymbol{S}^T$
   - $\boldsymbol{Q}^{(k+1)} = \boldsymbol{Q}^{(k)} + \boldsymbol{T} - \boldsymbol{U}^{(k+1)}$
   - $k := k + 1$, $\gamma^{(k)} = \gamma^{(k-1)} * t$
5. Return $\left\{ \boldsymbol{U}^{(k)} \right\}_{i=0}^{k}$ as the `CBASS` algorithmic regularization path

C.2 Visualizations for Convex Bi-Clustering

While it is possible to construct row- and column-wise CBASS analogues of the CARP dendrogram and path plots discussed above, the primary visualization associated with bi-clustering is the *cluster heatmap*, which combines a heatmap visualization of the raw data with independent row- and column-dendrograms (Wilkinson and Friendly, 2009). We modify the standard cluster heatmap by creating dendrograms using the fusions identified by CBASS. As Chi et al. (2017) argue, the joint estimation of dendrograms provided by convex bi-clustering often produces better results than independent dendrogram construction.

We applied CBASS to the Presidents data and show the resulting cluster heatmap in Figure A1. A close examination reveals several interesting patterns. This data clearly exhibits a bi-clustered structure, with certain words being strongly associated with certain groups of presidents. Examining the two clear bi-clusters on the left, we see that words such as "billion," "soviet," and "technology" are frequently used by modern presidents and rarely used by pre-modern presidents. Conversely, we see that words which may be considered somewhat antiquated, such as "vessel" or "shall," are associated with pre-modern presidents. For data with less clear structure, the interpretability of the cluster heatmap can sometimes be increased by plotting the smoothed estimates $\boldsymbol{U}^{(k)}$ rather than the raw data.

In simulation studies, CBASS appears to converge to the exact regularization path as $t \to 1$. While this is consistent with both our theory and observations for CARP, we leave the theoretical analysis of CBASS to future work. As far as we know, a rate of convergence has not been established for the DLPA in the optimization literature, without which the techniques used to prove Theorem 1 cannot be applied to CBASS.

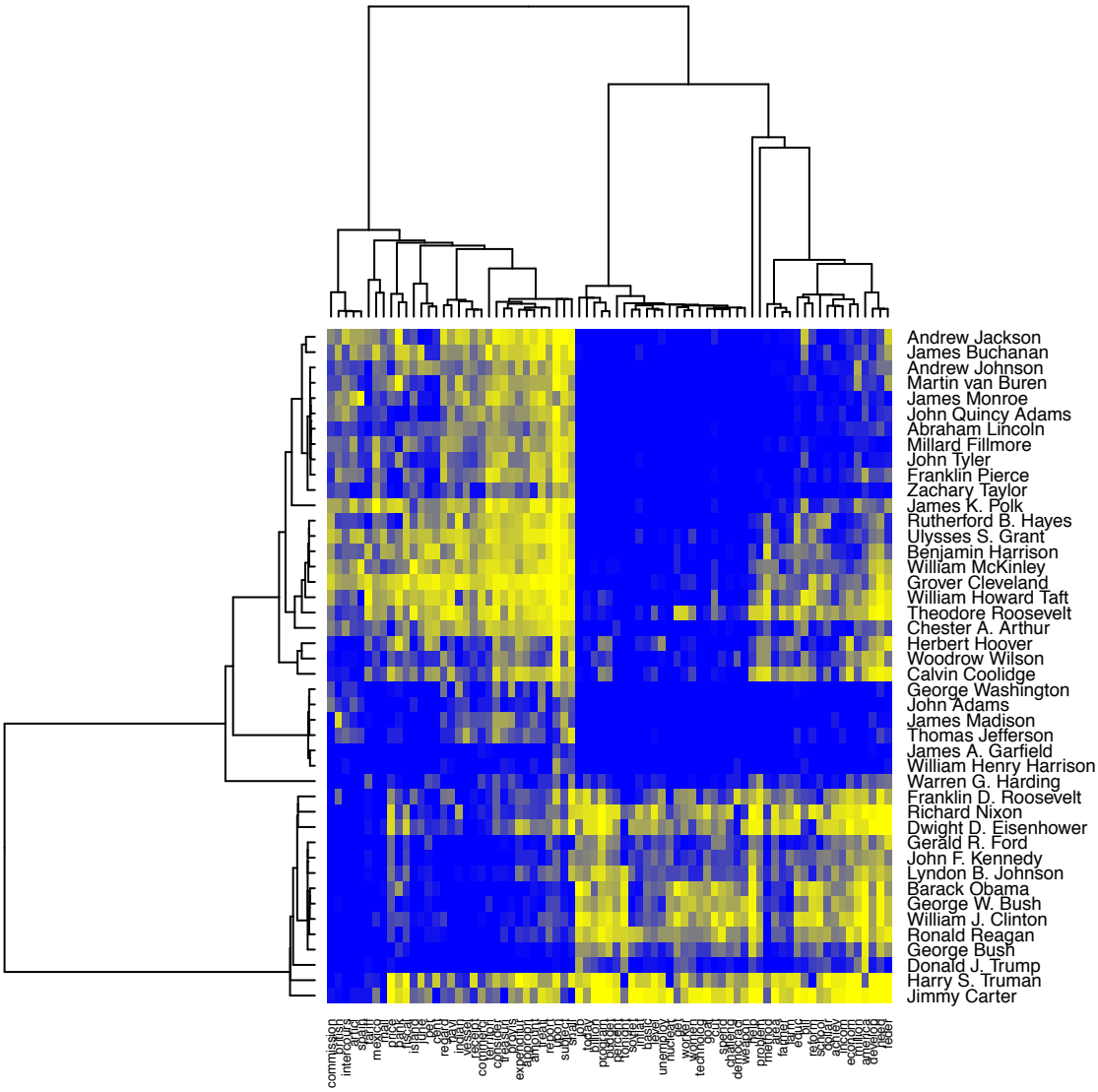

Figure A1: Cluster heatmap of the Presidents data, with row and column dendrograms jointly estimated by `CBASS`. The partitions estimated by `CBASS` clearly associated modern words with modern presidents and old-fashioned words with pre-modern presidents.

D Additional Comparisons

Figure A2 compares the accuracy of `CARP`, `CBASS`, hierarchical clustering, and K-means clustering on the TCGA and Authors data sets discussed in Section 4. While certain forms of hierarchical clustering perform well on this data, `CARP` achieves superior performance without requiring the user to select a distance or linkage.

Figure A3 compares the performance of `CARP`, hierarchical clustering with Euclidean distance and Ward's, complete, and single linkage, and K-means on data simulated from a Gaussian mixture model. The cluster centroids were equally spaced on a 2-dimensional subspace and $n = 54$ observations were generated from a Gaussian distribution with unit variance centered at the cluster centroid. Each of the clustering methods exhibit similar behaviors, with improved performance as the inter-cluster distance increases and decreased performance with higher ambient dimensionality or more clusters. Because these data were generated from isotropic Gaussians, all methods except single linkage hierarchical clustering perform well.

Figure A4 compares the performance of the same methods on non-convex clusters. In particular, we consider a version of the "half-moons" example proposed by Hocking et al. (2011). (See also Figure A5.) The data were generated on a two-dimensional subspace with $n = 50$ observations from each cluster and Gaussian noise orthogonal to the signal subspace were added. Not surprisingly, the performance of all methods degrades as the degree of noise and the ambient dimensionality are increased. Despite this, we see that `CARP` and single-linkage hierarchical clustering clearly outperform other methods, with `CARP` being more robust to the presence of noise.

Comparing these two simulations, we see that only convex clustering (`CARP`) is able to consistently perform well on both the convex and non-convex simulated data without requiring the user to select a distance metric or linkage. This is in large part due to the sparse weighting

scheme used in the `clustRviz` package, which is able to flexibly and robustly adapt to the observed data distribution. Our findings should be contrasted with those of Tan and Witten (2015) who focus only on the case of uniform weights and show that, without informative weights, convex clustering performs similarly to single linkage convex clustering.

E Back-Tracking, Post-Processing, and Dendrogram Construction

The `CARP-VIZ` variant of our `CARP` algorithm implements a back-tracking scheme in order to improve dendrogram recovery. Because a relatively large value of λ is typically required for any fusions to occur in convex clustering (1), `CARP-VIZ` begin with a large step-size (by default, $t = 1.1$) and performs standard `CARP` iterations until the first fusion is identified (*i.e.*, a row of $\boldsymbol{V}^{(k)}$ is set to zero). After the first fusion is identified, `CARP-VIZ` switches to a smaller step-size (by default, $t = 1.01$) for the remainder of the algorithm. At each iteration, `CARP-VIZ` counts the number of fusions that occur. If more than one fusion occurs, instead of proceeding, `CARP-VIZ` attempts to determine which fusion occurred first. It does so using a back-tracking scheme, similar to those used in optimization methods. `CARP-VIZ` discards the iteration with multiple fusions, halves the step-size, and performs another iteration. If this half-step iteration has only one fusion, `CARP-VIZ` accepts it and continues as before. Otherwise, `CARP-VIZ` again halves the step-size and repeats this process until the correct order of fusions is identified (or a limit on the number of back-tracking steps is hit). Once the first fusion is identified, `CARP-VIZ` resets t and continues. `CBASS-VIZ` uses essentially the same scheme, though it checks for both row and column fusions. We have found that, because it only uses a small step-size at "interesting" parts of the solution space, this back-tracking scheme typically produces more accurate dendrogram recovery at less expense than running standard `CARP` with a very small step-size.

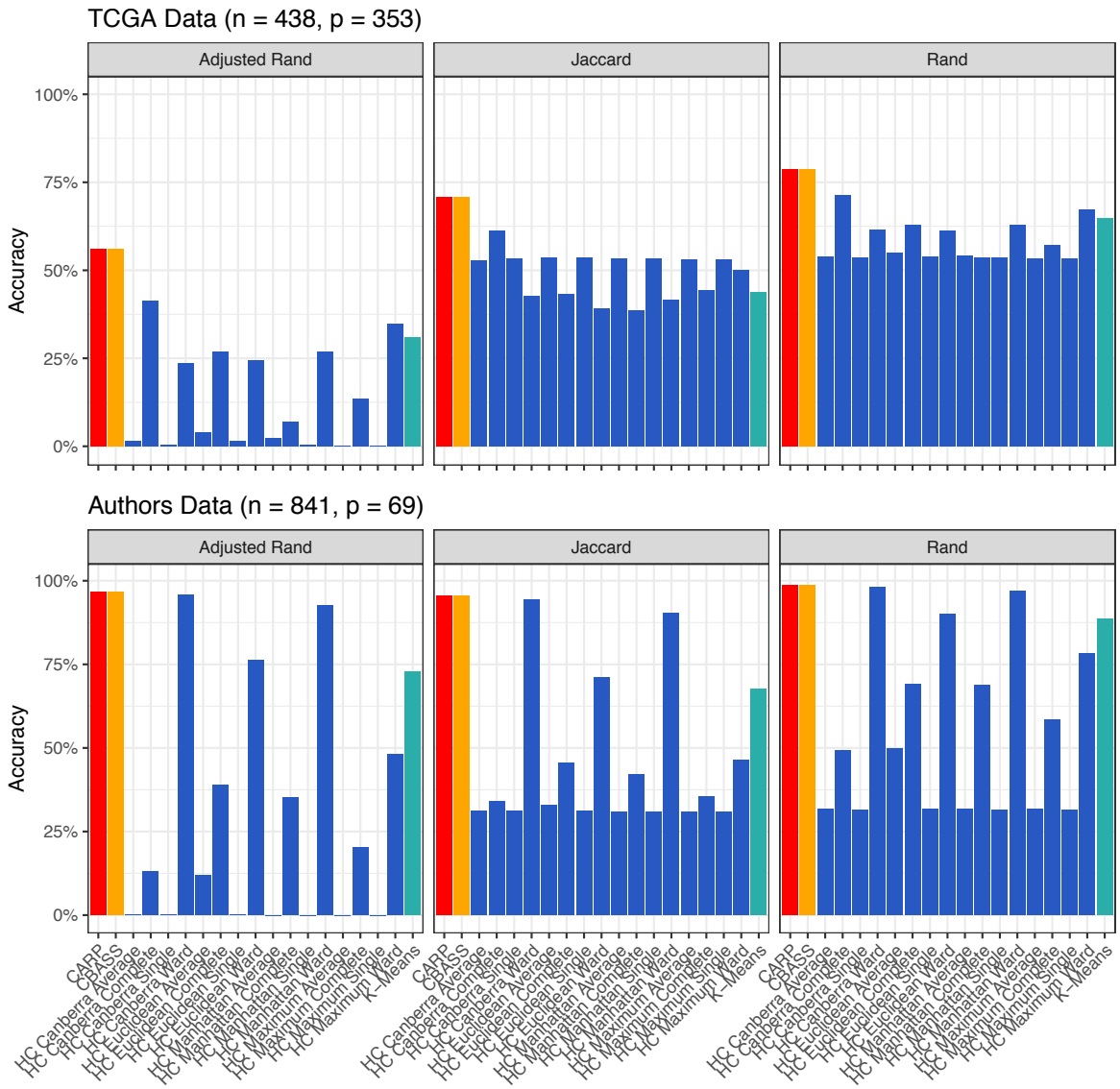

Figure A2: Accuracy of convex clustering (`CARP`, red), convex bi-clustering (`CBASS`, yellow, discussed in Section C), Hierarchical Clustering (`HC`, blue), and K-means Clustering (teal) on the TCGA data set. `CARP` and `CBASS` consistently outperform both hierarchical and K-means clustering, as measured by the Rand (Rand, 1971), Adjusted Rand (Hubert and Arabie, 1985), and Jaccard indices. `CARP` and `CBASS` were run using `clustRviz`'s default settings ($t = 1.05$ and $t = 1.01$ respectively).

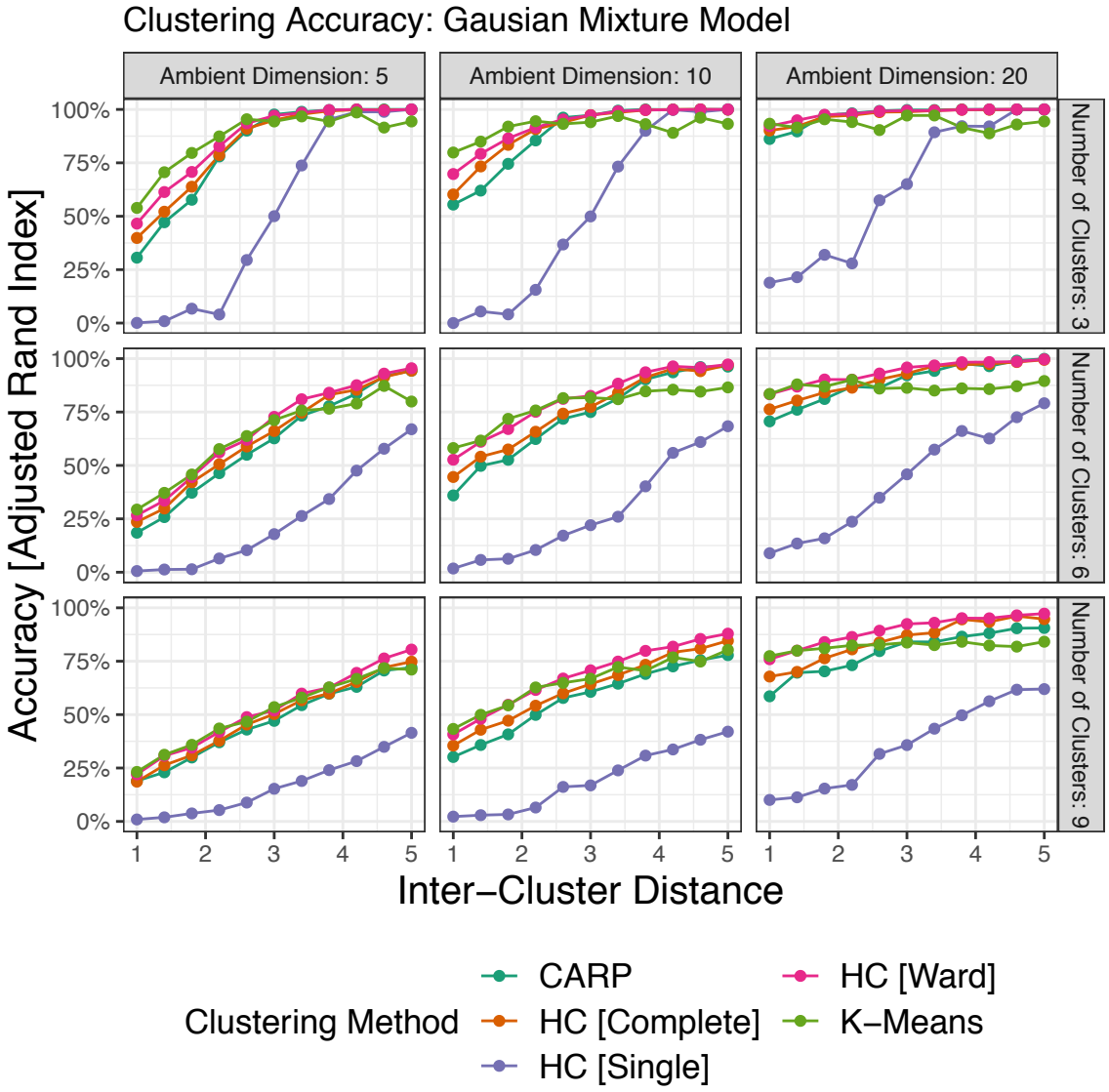

Figure A3: Accuracy of convex clustering, hierarchical clustering with Euclidean distance and several linkages, and K-means clustering (teal) on data simulated from a Gaussian mixture model, as measured by the Adjusted Rand (Hubert and Arabie, 1985) index. Because these clusters are spherical, with sufficient inter-cluster separation all methods except hierarchical clustering with single linkage perform well.

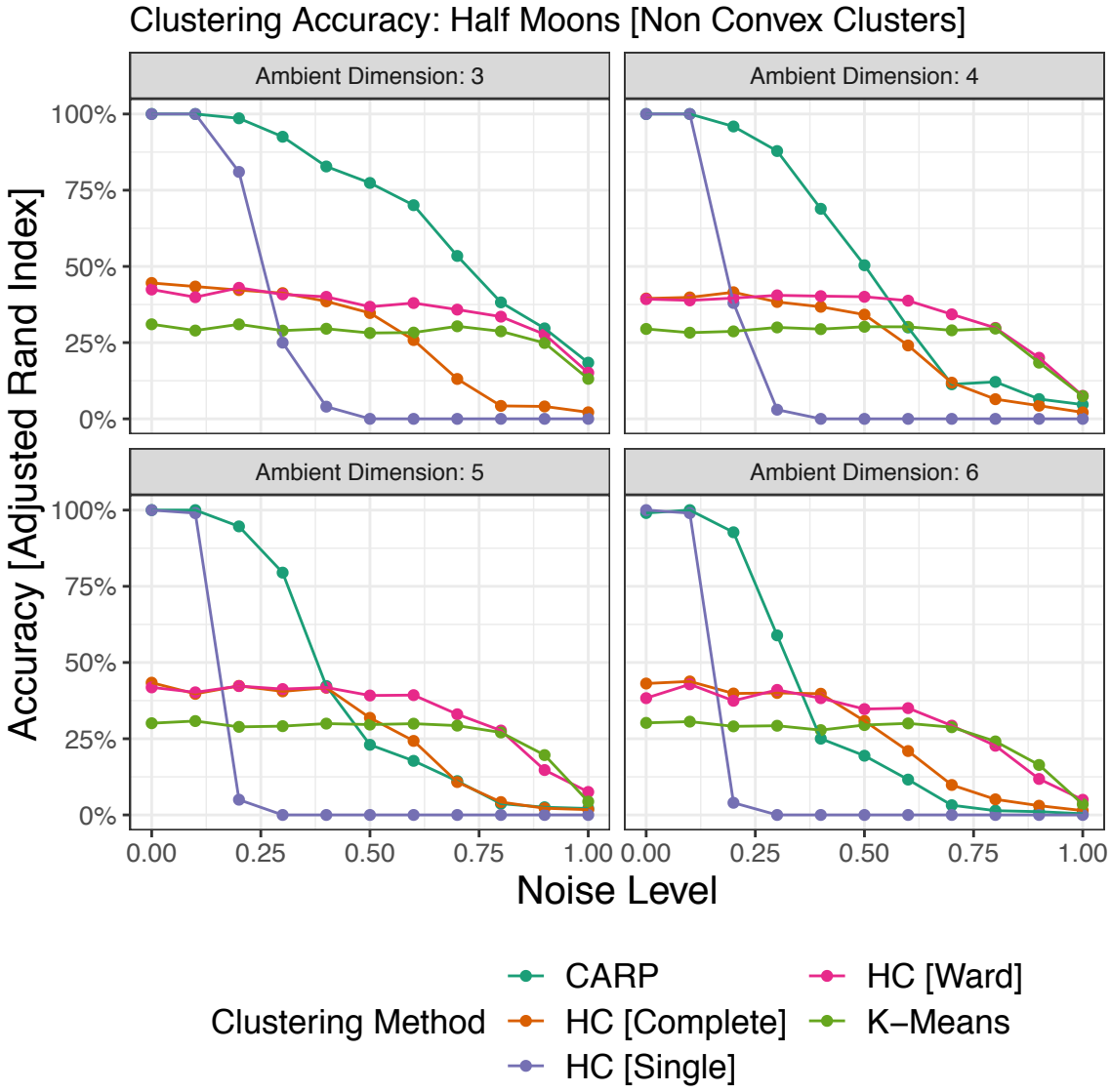

Figure A4: Accuracy of convex clustering, hierarchical clustering with Euclidean distance and several linkages, and K-means clustering (teal) on data simulated from a the two-circles and two-half-moons model, as measured by the Adjusted Rand (Hubert and Arabie, 1985) index. Note that only `CARP` and single linkage hierarchical clustering are able to adapt to the non-convex cluster shapes.

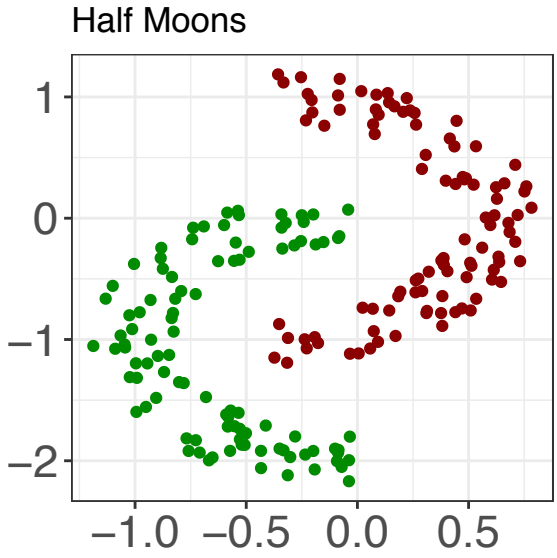

Figure A5: A sample realization of the interlocking half moons test data of Hocking et al. (2011) used for Figure A4. Data were generated from these clusters along a random two-dimensional subspace and Gaussian noise orthogonal to the signal subspace was added to increase the difficulty of the clustering problem. `CARP` and single-linkage hierarchical clustering are able to exactly recover the true clustering in the noiseless case; the performance of hierarchical clustering quickly degrades as more noise is added, however, while `CARP` is more robust.

Once `CARP` or `CARP-VIZ` terminate, `clustRviz` performs an additional post-processing step to isolate individual fusions. `clustRviz` reviews the fusions at each iteration and, if an iteration has multiple fusions, linearly interpolates between $\boldsymbol{U}^{(k)}$ and $\boldsymbol{U}^{(k+1)}$ to determine the approximate regularization level at which each fusion occurred. The interpolated iterate is only approximate, but is necessary for dendrogram construction. We note that no interpolation is typically needed for `CARP-VIZ` results, due to the back-tracking step used to isolate individual fusions, but, by default, `clustRviz` post-processes both `CARP` and `CARP-VIZ` output. The same post-processing scheme is applied separately to the row and column fusions from `CBASS`.

Once post-processing is performed, a dendrogram is constructed from the interpolated iterates. The dendrogram construction proceeds in the opposite order as hierarchical clustering: we begin with the fully fused data and decrease $\gamma^{(k)}$, noting the order in which centroids were fused. (We use the reverse ordering so that, in the rare case where the path contains fissions, the *final* fusion is reflected in the resulting dendrogram.) The dendrogram height associated with each fusion is the $\gamma^{(k)}$ at which that fusion is first observed. Finally, we check whether fusions are more uniformly distributed on the $\gamma^{(k)}$ scale or the $\log(\gamma^{(k)})$ and adjust the dendrogram height accordingly to provide less cluttered visualizations.

Since the weight selection, post-processing, and dendrogram reconstruction steps could potentially be applied to any convex clustering algorithm, they are omitted from all timing results shown in this paper.

F Additional Related Work

Following its original introduction by Pelckmans et al. (2005) and popularization by Hocking et al. (2011) and Lindsten et al. (2011), convex clustering has been the subject of much methodological and theoretical research. In this section, we review some of this related

work which, while not directly relevant to the computational or visualization strategies we propose, may be of interest to readers interested in convex clustering.

The convex clustering problem can be generalized as

$$\hat{\boldsymbol{U}}_\lambda = \underset{\boldsymbol{U} \in \mathbb{R}^{n \times p}}{\arg\min} \frac{1}{2} \|\boldsymbol{X} - \boldsymbol{U}\|_F^2 + \lambda \sum_{\substack{i,j=1 \\ i<j}}^{n} w_{ij} p\left(\boldsymbol{U}_{i\cdot} - \boldsymbol{U}_{j\cdot}\right)$$

where $p(\cdot)$ is any sparsity-inducing function. The choice of an ℓ_q-norm ($p(\cdot) = \|\cdot\|_q$) gives standard convex clustering as considered in this paper. Pan et al. (2013), Marchetti and Zhou (2014), Wu et al. (2016), and Shah and Koltun (2017) have all considered the use of non-convex choices of $p(\cdot)$, typically using the popular SCAD or MCP penalty functions to reduce bias and improve estimation performance (Fan and Li, 2001; Zhang, 2010). We do not consider non-convex $p(\cdot)$ in this paper, though the computational techniques and visualizations we propose could be adapted to non-convex penalties in a relatively straightforward manner.

Restricting our attention to standard convex clustering ($p(\cdot) = \|\cdot\|_q$), several useful methodological extensions have been proposed in the literature. For example, Wang et al. (2016) augment the convex clustering problem (1) with an additional sparse component to add robustness to outliers, similar to the robust PCA formulation of Candès et al. (2011), while Wang et al. (2018) propose a variant which incorporates feature selection into the clustering objective using an ℓ_1 penalty (Tibshirani, 1996). As discussed in Section C, Chi et al. (2017) extend convex clustering to the bi-clustering setting, where rows and columns are simultaneously clustered. Building on this work, Chi et al. (2018) extend bi-clustering to general co-clustering of k-order tensors, where they note several surprising theoretical advantages. The recent paper by Park et al. (2018+) extends convex clustering to *histogram-valued* data by replacing the Euclidean distance with an appropriate metric on the space of histograms.

The squared Frobenius loss function of the convex clustering problem may be interpreted as an isotropic Gaussian likelihood, suggesting another avenue for generalization. Sui et al. (2018) replace the Frobenius loss with a squared Mahalanobis distance to improve performance on non-spherical clusters. If the metric (inverse covariance matrix) is known, simple variants on the techniques used in this paper may be used; if the metric must be estimated from the data, the resulting problem is bi-convex and an alternating minimization scheme must be used, only guaranteeing convergence to a stationary point.

The use of a convex formulation allows the sophisticated tools of modern high-dimensional statistics to be brought to bear (Bühlmann and Geer, 2011; Hastie et al., 2015). In addition to the work of Tan and Witten (2015) proving a form prediction consistency and of Radchenko and Mukherjee (2017) proving asymptotic dendrogram recovery, Zhu et al. (2014) give sufficient conditions for exact cluster recovery in the two-cluster case. The results of Zhu et al. (2014) were later extended by Panahi et al. (2017) and by Sun et al. (2018) to the more general multi-cluster case.

In addition to the general purpose operator-splitting algorithms proposed by Chi and Lange (2015), specialized algorithms have been proposed for convex clustering in the "large n" (many observations) setting. Panahi et al. (2017) propose a stochastic incremental algorithm based on the framework of Bertsekas (2011), while Sun et al. (2018) propose a semi-smooth Newton algorithm based on the framework of Li et al. (2016). Chen et al. (2015) propose a proximal distance-based algorithm (Lange and Keys, 2014) and provide a GPU-based implementation. Recently, Ho et al. (2019) proposed a generalized dual gradient ascent algorithm with linear convergence, though their approach only works for the $q = 1$ case; their approach is likely amenable to algorithmic regularization schemes similar to those we have propose for the ADMM.

The special case of convex clustering in $\mathbb{R}$ has been studied under various names, including *total variation denoising* (Rudin et al., 1992), the *edge lasso* (Sharpnack et al., 2012) and the

graph-fused lasso (Hoefling, 2010), or as a special case of the generalized lasso (Tibshirani and Taylor, 2011). When the underlying graph is a chain graph, convex clustering simplifies to the well-studied *fused lasso* problem (Tibshirani et al., 2005; Rinaldo, 2009; Johnson, 2013).

Convex clustering has not yet seen significant adoption outside of the statistics and machine learning communities, though Chen et al. (2015) discuss applications to human genomics. Nagorski and Allen (2018) propose an alternative weighting scheme based on genetic distances which they use to perform genomic region segmentation.

Additional References

Bauschke, Heinz H. and Patrick L. Combettes (2008). "A Dykstra-Like Algorithm for Two Monotone Operators". *Pacific Journal of Optimization* 4.3, pp. 383–391. URL: http://www.ybook.co.jp/online2/oppjo/vol4/p383.html.

Bertsekas, Dimitri P. (2011). "Incremental proximal methods for large scale convex optimization". *Mathematical Programming, Series B* 129, pp. 163–195. DOI: 10.1007/s10107-011-0472-0.

Boyd, Stephen, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein (2011). "Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers". *Foundations and Trends® in Machine Learning* 3.1, pp. 1–122. DOI: 10.1561/2200000016.

Bühlmann, Peter and Sara van de Geer (2011). *Statistics for High-Dimensional Data: Methods, Theory, and Applications*. Springer Series in Statistics. Springer Verlag. ISBN: 978-3-642-20191-2. DOI: 10.1007/978-3-642-20192-9.

Candès, Emmanuel J., Xiaodong Li, Yi Ma, and John Wright (2011). "Robust Principal Component Analysis?" *Journal of the ACM* 58.3, 11:1–11:37. DOI: 10.1145/1970392.1970395.

Chen, Gary K., Eric C. Chi, John Michael O. Ranola, and Kenneth Lange (2015). "Convex Clustering: An Attractive Alternative to Hierarchical Clustering". *PLOS Computational Biology* 11.5, e1004228. DOI: 10.1371/journal.pcbi.1004228.

Chi, Eric C., Genevera I. Allen, and Richard G. Baraniuk (2017). "Convex Biclustering". *Biometrics* 73.1, pp. 10–19. DOI: 10.1111/biom.12540.

Chi, Eric C., Brian R. Gaines, Will Wei Sun, Hua Zhou, and Jian Yang (2018). "Provable Convex Co-Clustering of Tensors". *ArXiv Pre-Print 1803.06518*. URL: https://arxiv.org/abs/1803.06518.

Chi, Eric C. and Kenneth Lange (2015). "Splitting Methods for Convex Clustering". *Journal of Computational and Graphical Statistics* 24.4, pp. 994–1013. DOI: 10.1080/10618600.2014.948181.

Combettes, Patrick L. and Jean-Cristophe Pesquet (2011). "Proximal Splitting Methods in Signal Processing". *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*. Ed. by Heinz H. Bauschke, Regina S. Burachik, Patrick L. Combettes, Veit Elser, D. Russell Luke, and Henry Wolkowicz. Springer. Chap. 10, pp. 185–212. DOI: 10.1007/978-1-4419-9569-8_10.

Deng, Wei and Wotao Yin (2016). "On the Global and Linear Convergence of the Generalized Alternating Direction Method of Multipliers". *Journal of Scientific Computing* 66.3, pp. 889–916. DOI: 10.1007/s10915-015-0048-x.

Duchi, John, Shai Shalev-Shwartz, Yoram Singer, and Tushar Chandra (2008). "Efficient Projections onto the ℓ_1-Ball for Learning in High Dimensions". *ICML 2008: Proceedings of the 25th International Conference on Machine Learning*. Ed. by Andrew McCallum and Sam Roweis. Helsinki, Finland: Omnipress, pp. 272–279. DOI: 10.1145/1390156.1390191.

Fan, Jianqing and Runze Li (2001). "Variable Selection via Nonconcave Penalized Likelihood and its Oracle Properties". *Journal of the American Statistical Association* 96.456, pp. 1348–1360. DOI: 10.1198/016214501753382273.

Gabay, Daniel and Bertrand Mercier (1976). "A Dual Algorithm for the Solution of Nonlinear Variational Problems via Finite Element Approximation". *Computers & Mathematics with Applications* 2.1, pp. 17–40. DOI: 10.1016/0898-1221(76)90003-1.

Glowinski, R. and A. Marroco (1975). "Sur l'Approximation, par Éléments finis d'ordre un, et la résolution, par pénalisation-dualité d'une classe de problèmes de Dirichlet non linéaires". *Revue française d'automatique, informatique, recherche opérationnelle. Analyse numérique* 9.R2, pp. 41–76.

Goldstein, Tom, Brendan O'Donoghue, Simon Setzer, and Richard Baraniuk (2014). "Fast Alternating Direction Optimization Methods". *SIAM Journal on Imaging Sciences* 7.3, pp. 1588–1623. DOI: 10.1137/120896219.

Hastie, Trevor, Robert Tibshirani, and Martin Wainwright (2015). *Statistical Learning with Sparsity: The Lasso and Generalizations*. Monographs on Statistics and Applied Probability 143. Chapman and Hall/CRC. ISBN: 978-1-498-71216-3.

Ho, Nhat, Tianyi Lin, and Michael I. Jordan (2019). "Global Error Bounds and Linear Convergence for Gradient-Based Algorithms for Trend Filtering and ℓ_1-Convex Clustering". *ArXiv Pre-Print 1904.07462*. URL: https://arxiv.org/abs/1904.07462.

Hocking, Toby Dylan, Armand Joulin, Francis Bach, and Jean-Philippe Vert (2011). "Clusterpath: An Algorithm for Clustering using Convex Fusion Penalties". *ICML 2011: Proceedings of the 28th International Conference on Machine Learning*. Ed. by Lise Getoor and Tobias Scheffer. Bellevue, Washington, USA: ACM, pp. 745–752. ISBN: 978-1-4503-0619-5. URL: http://www.icml-2011.org/papers/419_icmlpaper.pdf.

Hoefling, Holger (2010). "A Path Algorithm for the Fused Lasso Signal Approximator". *Journal of Computational and Graphical Statistics* 19.4, pp. 984–1006. DOI: 10.1198/jcgs.2010.09208.

Hong, Mingyi and Zhi-Quan Luo (2017). "On the linear convergence of the alternating direction method of multipliers". *Mathematical Programming, Series A* 162.1-2, pp. 165–1699. DOI: 10.1007/s10107-016-1034-2.

Hubert, Lawrence and Phipps Arabie (1985). "Comparing Partitions". *Journal of Classification* 2.1, pp. 193–218. DOI: 10.1007/BF01908075.

Johnson, Nicholas A. (2013). "A Dynamic Programming Algorithm for the Fused Lasso and L_0-Segmentation". *Journal of Computational and Graphical Statistics* 22.2, pp. 246–260. DOI: 10.1080/10618600.2012.681238.

Lange, Kenneth and Kevin L. Keys (2014). "The Proximal Distance Algorithm". *ICM 2014: Proceedings of the International Congress of Mathematicians*. Ed. by Sun Young Jang, Young Rock Kim, Dae-Woong Lee, and Ikkwon Yie. Vol. 4. Seoul, Korea, pp. 95–116. URL: http://www.icm2014.org/en/vod/proceedings.html.

Li, Xudong, Defeng Sun, and Kim-Chuan Toh (2016). "A Highly Efficient Semismooth Newton Augmented Lagrangian Method for Solving Lasso Problems". *SIAM Journal on Optimization* 28.1, pp. 433–458. DOI: 10.1137/16M1097572.

Lindsten, Fredrik, Henrik Ohlsson, and Lennart Ljung (2011). "Clustering using sum-of-norms regularization: With application to particle filter output computation". *SSP 2011: Proceedings of the 2011 IEEE Statistical Signal Processing Workshop*. Ed. by Petar M. Djuric. Nice, France: Curran Associates, Inc., pp. 201–204. DOI: 10.1109/SSP.2011.5967659.

Lions, P.L. and B. Mercier (1979). "Splitting Algorithms for the Sum of Two Nonlinear Operators". *SIAM Journal on Numerical Analysis* 16.6, pp. 964–979. DOI: 10.1137/0716071.

Marchetti, Yuliya and Qing Zhou (2014). "Solution Path Clustering with Adaptive Concave Penalty". *Electronic Journal of Statistics* 8.1, pp. 1569–1603. DOI: 10.1214/14-EJS934.

Moreau, Jean Jacques (1962). "Décomposition Orthogonale d'un Espace Hilbertien Selon Deux Cônes Mutuellement Polaires". *Comptes Rendus Hebdomadaires des Séances de l'Académie des Sciences* 255.1, pp. 238–240.

Nagorski, John and Genevera I. Allen (2018). "Genomic Region Detection via Spatial Convex Clustering". *PLoS One* 13.9, e0203007. DOI: 10.1371/journal.pone.0203007.

Nishihara, Robert, Laurent Lessard, Ben Recht, Andrew Packard, and Michael Jordan (2015). "A General Analysis of the Convergence of ADMM". *ICML:2015: Proceedings of the 32nd International Conference on Machine Learning*. Ed. by Francis Bach and David Blei. Lille, France: PMLR, pp. 343–352. URL: http://proceedings.mlr.press/v37/nishihara15.html.

Pan, Wei, Xiaotong Shen, and Binghui Liu (2013). "Cluster Analysis: Unsupervised Learning via Supervised Learning with a Non-convex Penalty". *Journal of Machine Learning Research* 14, pp. 1865–1889. URL: http://www.jmlr.org/papers/v14/pan13a.html.

Panahi, Ashkan, Devdatt Dubhashi, Fredrik D. Johansson, and Chiranjib Bhattacharyya (2017). "Clustering by Sum of Norms: Stochastic Incremental Algorithm, Convergence, and Cluster Recovery". *ICML:2017: Proceedings of the 34th International Conference on Machine Learning*. Ed. by Doina Precup and Yee Whye Teh. Vol. 70. Sydney, Australia: PMLR, pp. 2769–2777. URL: http://proceedings.mlr.press/v70/panahi17a.html.

Park, Cheolwoo, Hosik Choi, Chris Delcher, Yanning Wang, and Young Joo Yoon (2018+). "Convex Clustering Analysis for Histogram-Valued Data". *Biometrics* To appear. DOI: 10.1111/biom.13004.

Pelckmans, Kristiaan, Joseph de Brabanter, Bart de Moor, and Johan Suykens (2005). "Convex Clustering Shrinkage". *PASCAL Workshop on Statistics and Optimization of Clustering*.

Radchenko, Peter and Gourab Mukherjee (2017). "Convex Clustering via ℓ_1 Fusion Penalization". *Journal of the Royal Statistical Society, Series B: Statistical Methodology* 79.5, pp. 1527–1546. DOI: 10.1111/rssb.12226.

Rand, William M. (1971). "Objective Criteria for the Evaluation of Clustering Methods". *Journal of the American Statistical Association* 66.366, pp. 846–850. DOI: 10.1080/01621459.1971.10482356.

Rinaldo, Alessandro (2009). "Properties and Refinements of the Fused Lasso". *Annals of Statistics* 37.5B, pp. 2922–2952. DOI: 10.1214/08-AOS665.

Rockafellar, R. Tyrrell (1970). *Convex Analysis*. Princeton University Press. ISBN: 978-0-691-01586-6.

Rosset, Saharon and Ji Zhu (2007). "Piecewise Linear Regularized Solution Paths". *Annals of Statistics* 35.3, pp. 1012–1030. DOI: 10.1214/009053606000001370.

Rudin, Leonid I., Stanley Osher, and Emad Fatemi (1992). "Nonlinear Total Variation Based Noise Removal Algorithms". *Physica D: Nonlinear Phenomena* 60.1-4, pp. 259–268. DOI: 10.1016/0167-2789(92)90242-F.

Shah, Sohil Atul and Vladlen Koltun (2017). "Robust continuous clustering". *Proceedings of the National Academy of Sciences of the United States* 114.37, pp. 9814–9819. DOI: 10.1073/pnas.1700770114.

Sharpnack, James, Aarti Singh, and Alessandro Rinaldo (2012). "Sparsistency of the Edge Lasso over Graphs". *AISTATS 2012: Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics*. Ed. by Neil D. Lawrence and Mark Girolami. Vol. 22. La Palma, Canary Islands: PMLR, pp. 1028–1036. URL: http://proceedings.mlr.press/v22/sharpnack12.html.

Shi, Wei, Qing Ling, Kun Yuan, Gang Wu, and Wotao Yin (2014). "On the Linear Convergence of the ADMM in Decentralized Consensus Optimization". *IEEE Transactions on Signal Processing* 62.7, pp. 1750–1761. DOI: 10.1109/TSP.2014.2304432.

Sui, Xiaopeng Lucia, Li Xu, Xiaoning Qian, and Tie Liu (2018). "Convex Clustering with Metric Learning". *Pattern Recognition* 81, pp. 575–584. DOI: 10.1016/j.patcog.2018.04.019.

Sun, Defeng, Kim-Chuan Toh, and Yancheng Yuan (2018). "Convex Clustering: Model, Theoretical Guarantee and Efficient Algorithm". *ArXiv Pre-Print 1810.02677*. URL: https://arxiv.org/abs/1810.02677.

Tan, Kean Ming and Daniela Witten (2015). "Statistical Properties of Convex Clustering". *Electronic Journal of Statistics* 9.2, pp. 2324–2347. DOI: 10.1214/15-EJS1074.

Tibshirani, Robert (1996). "Regression Shrinkage and Selection via the Lasso". *Journal of the Royal Statistical Society, Series B (Methodological)* 58.1, pp. 267–288. DOI: 10.1111/j.2517-6161.1996.tb02080.x.

Tibshirani, Robert, Michael Saunders, Saharon Rosset, Ji Zhu, and Keith Knight (2005). "Sparsity and Smoothness via the Fused Lasso". *Journal of the Royal Statistical Society, Series B: Statistical Methodology* 67.1, pp. 91–108. DOI: 10.1111/j.1467-9868.2005.00490.x.

Tibshirani, Ryan J. and Jonathan Taylor (2011). "The Solution Path of the Generalized Lasso". *Annals of Statistics* 39.3, pp. 1335–1371. DOI: 10.1214/11-AOS878.

Tseng, Paul (1991). "Applications of a Splitting Algorithm to Decomposition in Convex Programming and Variational Inequalities". *SIAM Journal on Control and Optimization* 29.1, pp. 119–138. DOI: 10.1137/0329006.

Wang, Binhuan, Yilong Zhang, Will Wei Sun, and Yixin Fang (2018). "Sparse Convex Clustering". *Journal of Computational and Graphical Statistics* 27.2, pp. 393–403. DOI: 10.1080/10618600.2017.1377081.

Wang, Qi, Pinghua Gong, Shiyu Chang, Thomas S. Huang, and Jiayu Zhou (2016). "Robust Convex Clustering Analysis". *ICDM 2016: Proceedings of the 16th IEEE International Conference on Data Mining*. Ed. by Francesco Bonchi, Josep Domingo-Ferrer, Ricardo Baeza-Yates, Zhi-Hua Zhou, and Xindong Wu. Barcelona, Spain, pp. 1263–1268. DOI: 10.1109/ICDM.2016.0170.

Wilkinson, Leland and Michael Friendly (2009). "The History of the Cluster Heat Map". *The American Statistician* 63.2, pp. 179–184. DOI: 10.1198/tas.2009.0033.

Wu, Chong, Sunghoon Kown, Xiaotong Shen, and Wei Pan (2016). "A New Algorithm and Theory for Penalized Regression-based Clustering". *Journal of Machine Learning Research* 17.188, pp. 1–25. URL: http://jmlr.org/papers/v17/15-553.html.

Yang, Wei Hong and Deren Han (2016). "Linear Convergence of the Alternating Direction Method of Multipliers for a Class of Convex Optimization Problems". *SIAM Journal on Numerical Analysis* 54.2, pp. 625–640. DOI: 10.1137/140974237.

Zhang, Cun-Hui (2010). "Nearly Unbiased Variable Selection under Minimax Concave Penalty". *Annals of Statistics* 38.2, pp. 894–942. DOI: 10.1214/09-AOS729.

Zhu, Changbo, Huan Xu, Chenlei Leng, and Shuicheng Yan (2014). "Convex Optimization Procedure for Clustering: Theoretical Revisit". *NIPS 2014: Advances in Neural Information Processing Systems 27*. Ed. by Zoubin Ghahramani, Max Welling, Corinna Cortes, Neil D. Lawrence, and Killian Q. Weinberger. Montréal, Canada: Curran Associates, Inc., pp. 1619–1627. URL: https://papers.nips.cc/paper/5307-convex-optimization-procedure-for-clustering-theoretical-revisit.