Title

Extensions to the Proximal Distance Method of Constrained Optimization.

Permalink

https://escholarship.org/uc/item/5mn36731

Authors

Landeros, Alfonso

Padilla, Oscar Hernan Madrid

Zhou, Hua

et al.

Publication Date

2022

Copyright Information

Peer reviewed

Extensions to the Proximal Distance Method of Constrained Optimization

Alfonso Landeros,

Department of Computational Medicine, University of California, Los Angeles CA 90095-1596, USA

Oscar Hernan Madrid Padilla,

Department of Statistics, University of California, Los Angeles CA 90095-1554, USA

Hua Zhou,

Departments of Biostatistics and Computational Medicine, University of California, Los Angeles CA 90095-1596, USA

Kenneth Lange

Departments of Computational Medicine, Human Genetics, and Statistics,University of California, Los Angeles CA 90095-1596, USA

Abstract

The current paper studies the problem of minimizing a loss $f(x)$ subject to constraints of the form $Dx \in S$, where S is a closed set, convex or not, and D is a matrix that fuses parameters. Fusion constraints can capture smoothness, sparsity, or more general constraint patterns. To tackle this generic class of problems, we combine the Beltrami-Courant penalty method of optimization with the proximal distance principle. The latter is driven by minimization of penalized objectives $f(x) + \frac{\rho}{2}\mathrm{dist}(Dx, S)^2$ involving large tuning constants ρ and the squared Euclidean distance of Dx from S. The next iterate x_{n+1} of the corresponding proximal distance algorithm is constructed from the current iterate x_n by minimizing the majorizing surrogate function $f(x) + \frac{\rho}{2}\|Dx - \mathscr{P}_s(Dx_n)\|^2$.

For fixed ρ and a subanalytic loss $f(x)$ and a subanalytic constraint set S, we prove convergence to a stationary point. Under stronger assumptions, we provide convergence rates and demonstrate linear local convergence. We also construct a steepest descent (SD) variant to avoid costly linear system solves. To benchmark our algorithms, we compare their results to those delivered by the alternating direction method of multipliers (ADMM). Our extensive numerical tests include problems on metric projection, convex regression, convex clustering, total variation image denoising, and projection of a matrix to a good condition number. These experiments demonstrate the superior speed and acceptable accuracy of our steepest variant on high-dimensional problems. Julia code to replicate all of our experiments can be found at https://github.com/alanderos91/ProximalDistanceAlgorithms.jl

ALANDEROS@UCLA.EDU .

1. Introduction

The generic problem of minimizing a continuous function $f(x)$ over a closed set S of $\mathbb{R}^p$ can be attacked by a combination of the penalty method and distance majorization. The classical penalty method seeks the solution of a penalized version $h_\rho(x) = f(x) + \rho q(x)$ of $f(x)$, where the penalty $q(x)$ is nonnegative and 0 precisely when $x \in S$. If one follows the solution vector x_ρ as ρ tends to ∞, then in the limit one recovers the constrained solution (Beltrami, 1970; Courant, 1943). The function

$$q(x) = \frac{1}{2}\text{dist}(x, S)^2 = \frac{1}{2}\min_{y \in S}\left\|x - y\right\|^2,$$

is one of the most fruitful penalties in this setting. Our previous research for solving this penalized minimization problem has focused on an MM (majorization-minimization) algorithm based on distance majorization (Chi et al., 2014; Keys et al., 2019). In distance majorization one constructs the surrogate function

$$g_\rho(x \mid x_n) = f(x) + \frac{\rho}{2}\|x - \mathscr{P}(x_n)\|^2,$$

using the Euclidean projection $\mathscr{P}(x_n)$ of the current iterate x_n onto S. The minimum of the surrogate occurs at the proximal point

$$x_{n+1} = \text{prox}_{\rho^{-1}f}\left[\mathscr{P}(x_n)\right]. \tag{1}$$

According to the MM principle, this choice of x_{n+1} decreases $g_\rho(x \mid x_n)$ and hence the objective $h_\rho(x)$ as well. As we note in our previous *JMLR* paper (Keys et al., 2019), the update (1) reduces to the classical proximal gradient method when S is convex (Parikh, 2014).

We have named this iterative scheme the proximal distance algorithm (Keys et al., 2019; Lange, 2016). It enjoys several virtues. First, it allows one to exploit the extensive body of results on proximal maps and projections. Second, it does not demand that the constraint set S be convex. If S is merely closed, then the map $\mathscr{P}(x)$ may be multivalued, and one must choose a representative element from the projection $\mathscr{P}(x_n)$. Third, the algorithm does not require the objective function $f(x)$ to be differentiable. Fourth, the algorithm dispenses with the chore of choosing a step length. Fifth, if sparsity is desirable, then the sparsity level can be directly specified rather than implicitly determined by the tuning parameter of the lasso or other penalty.

Traditional penalty methods have been criticized for their numerical instability. This hazard is mitigated in the proximal distance algorithm by its reliance on proximal maps, which

are often highly accurate. The major defect of the proximal distance algorithm is slow convergence. This can be ameliorated by Nesterov acceleration (Nesterov, 2013). There is also the question of how fast one should send ρ to ∞. Although optimal schedules are rarely known, simple numerical experiments yield a good choice. Finally, soft constraints can be achieved by stopping the steady increase of ρ at a finite value.

1.1 Proposed Framework

Distance majorization can be generalized in various ways. For instance, it can be expanded to multiple constraint sets. In practice, at most two constraint sets usually suffice. Another generalization is to replace the constraint $x \in S$ by the constraint $Dx \in S$, where D is a compatible matrix. Again, the original case $D = I$ is allowed. By analogy with the fused lasso of Tibshirani et al. (2005), we will call the matrix D a fusion matrix. With these ideas in mind, we examine the general problem of minimizing a differentiable function $f(x)$ subject to r fused constraints $D_i x \in S_i$. We approach this problem by extending the proximal distance method. For a fixed penalty constant ρ, the objective function and its MM surrogate now become

$$h_\rho(x) = f(x) + \frac{\rho}{2} \sum_{i=1}^{r} \text{dist}(D_i x, S_i)^2,$$

$$g_\rho(x \mid x_n) = f(x) + \frac{\rho}{2} \sum_{i=1}^{r} \| D_i x - \mathscr{P}_i(D_i x_n) \|^2,$$

where $\mathscr{P}_i(y)$ denotes the projection of y onto S_i. Any or all of the fusion matrices D_i can be the identity I. Our motivating premise posits that projection onto each set S_i is more straightforward than projection onto its preimage $D_i^{-1}(S_i)$. When the contrary is true, one should clearly favor projection onto the preimage.

Fortunately, we can simplify the problem by defining S to be the Cartesian product $\prod_{i=1}^{r} S_i$ and D to be the stacked matrix

$$D = \begin{pmatrix} D_1 \\ \vdots \\ D_r \end{pmatrix}.$$

Our objective and surrogate then revert to the less complicated forms

$$h_\rho(x) = f(x) + \frac{\rho}{2} \text{dist}(Dx, S)^2, \tag{2}$$

$$g_\rho(x \mid x_n) = f(x) + \frac{\rho}{2} \| Dx - \mathscr{P}(Dx_n) \|^2, \tag{3}$$

respectively, where $\mathcal{P}(\boldsymbol{x})$ is the Cartesian product of the projections $\mathcal{P}_i(\boldsymbol{x})$. Note that all closed sets S_i with simple projections, including sparsity sets, are fair game.

1.2 Our Contributions

In the framework described above, we summarize the contributions of the current paper.

a.   Section 2 describes solution algorithms for minimizing the penalized loss $h_\rho(\boldsymbol{x})$. Our first algorithm is based on Newton's method applied to the surrogate $g_\rho(\boldsymbol{x} \mid \boldsymbol{x}_n)$. For some important problems, Newton's method reduces to least squares. Our second method is a steepest descent algorithm on $g_\rho(\boldsymbol{x} \mid \boldsymbol{x}_n)$ tailored to high dimensional problems.

b.   For a sufficiently large ρ, we show that when $f(\boldsymbol{x})$ and S are convex and $f(\boldsymbol{x})$ possesses a unique minimum point in the preimage of S under $\boldsymbol{D}$, $\boldsymbol{y} \in \boldsymbol{D}^{-1}(S)$, the penalized loss $h_\rho(\boldsymbol{x})$ attains its minimum value. This is the content of Proposition 3.1. Similarly, Proposition 3.2 shows that the surrogate $g_\rho(\boldsymbol{x} \mid \boldsymbol{x}_n)$ also attains its minimum.

c.   If in addition $f(\boldsymbol{x})$ is differentiable, then Proposition 3.3 demonstrates that the MM iterates $\boldsymbol{x}_n$ for minimizing $h_\rho(\boldsymbol{x})$ satisfy

$$h_\rho(\boldsymbol{x}_n) - h_\rho(\boldsymbol{z}_\rho) = O\left(\frac{\rho}{n}\right),$$

where $\boldsymbol{z}_\rho$ minimizes $h_\rho(\boldsymbol{x})$. If $f(\boldsymbol{x})$ is also L-smooth and μ-strongly convex, then Proposition 3.4 shows that $\boldsymbol{z}_\rho$ is unique and the iterates $\boldsymbol{x}_n$ converge to $\boldsymbol{z}_\rho$ at a linear rate.

d.   More generally, Proposition 4.1 proves that the iterates $\boldsymbol{x}_n$ of a generic MM algorithm converge to a stationary point of a coercive subanalytic function $h(\boldsymbol{x})$ with a good surrogate. Our objectives and their surrogates fall into this category. Proposition 4.2 specializes the results of Proposition 4.1 to proximal distance algorithms. Propositions 4.3 and 4.4 further specialize to proximal distance algorithms with sparsity constraints and demonstrate a linear rate of convergence. Sparsity sets appear in model selection.

e.   Finally, we discuss a competing alternating direction method of multipliers (ADMM) algorithm and note its constituent updates. Our extensive numerical experiments compare the two proximal distance algorithms to ADMM. We find that proximal distance algorithms are competitive with and often superior to ADMM in terms of accuracy and running time.

1.3 Notation

The symbols $\boldsymbol{x}$ and $\boldsymbol{y}$ and their decorated variants are typically reserved for optimization variables. Fusion operators $\boldsymbol{D}$ are problem specific but understood as real matrices. Generally, $\boldsymbol{D}$ is neither injective nor surjective; the inverse image of S is denoted $\boldsymbol{D}^{-1}(S)$. The function $f(\boldsymbol{x})$ represents the loss in the constrained minimization problem

$$\min_{\boldsymbol{x}} f(\boldsymbol{x}) \text{ such that } \boldsymbol{Dx} \in S,$$

whereas $h_\rho(\boldsymbol{x}) = f(\boldsymbol{x}) + \frac{\rho}{2}\text{dist}(\boldsymbol{Dx}, S)^2$ is the the penalized loss. Our point to set distance functions are based on the Euclidean norm via

$$\text{dist}(\boldsymbol{y}, S) = \inf_{\boldsymbol{z} \in S} \left\| \boldsymbol{y} - \boldsymbol{z} \right\|_2 \quad \text{and} \quad \text{dist}(\boldsymbol{Dx}, S) = \inf_{\boldsymbol{z} \in S} \left\| \boldsymbol{Dx} - \boldsymbol{z} \right\|_2.$$

Finally, the notation $\mathscr{P}_S(\boldsymbol{y})$ indicates the unique Euclidean projection of $\boldsymbol{y}$ onto the closed convex set S. If S is merely closed, then $\mathscr{P}_S(\boldsymbol{y})$ may be set-valued. Depending on the context, $\mathscr{P}_S(\boldsymbol{y})$ also denotes a particular representative of this set.

2. Different Solution Algorithms

Unless $f(\boldsymbol{x})$ is a convex quadratic, exact minimization of the surrogate $g_\rho(\boldsymbol{x} \mid \boldsymbol{x}_n)$ is likely infeasible. As we have already mentioned, to reduce the objective $h_\rho(\boldsymbol{x})$ in (2), it suffices to reduce the surrogate (3). For the latter task, we recommend Newton's method on small and intermediate-sized problems and steepest descent on large problems. The exact nature of these generic methods are problem dependent. The following section provides a high-level overview of each strategy and we defer details on our later numerical experiments to the appendices.

2.1 Newton's Method and Least Squares

Unfortunately, the proximal operator $\text{prox}_{\rho^{-1}f}(\boldsymbol{y})$ is no longer relevant in calculating the MM update $\boldsymbol{x}_{n+1}$. When $f(\boldsymbol{x})$ is smooth, Newton's method for the surrogate $g_\rho(\boldsymbol{x} \mid \boldsymbol{x}_n)$ defined in Equation (3) employs the update

$$\boldsymbol{x}_{n+1} = \boldsymbol{x}_n - \left[\boldsymbol{H}_n + \rho\boldsymbol{D}^t\boldsymbol{D}\right]^{-1}\left\{\nabla f(\boldsymbol{x}_n) + \rho\boldsymbol{D}^t[\boldsymbol{Dx}_n - \mathscr{P}(\boldsymbol{Dx}_n)]\right\},$$

where $\boldsymbol{H}_n = d^2 f(\boldsymbol{x}_n)$ is the Hessian. To enforce the descent property, it is often prudent to substitute a positive definite approximation $\boldsymbol{H}_n$ for $d^2 f(\boldsymbol{x}_n)$. In statistical applications, the expected information matrix is a natural substitute. It is also crucial to retain as much curvature information on $f(\boldsymbol{x})$ as possible. Newton's method has two drawbacks. First, it is necessary to compute and store $d^2 f(\boldsymbol{x}_n)$. This is mitigated in statistical applications by the substitution just mentioned. Second, there is the necessity of solving a large linear system. Fortunately, the matrix $\boldsymbol{H}_n + \rho\boldsymbol{D}^t\boldsymbol{D}$ is often well-conditioned relative to $\boldsymbol{H}_n$, for example, when $\boldsymbol{D}$ has full column rank and $\boldsymbol{D}^t\boldsymbol{D}$ is positive definite. The method of conjugate gradients can be called on to solve the linear system in this ideal circumstance.

To reduce the condition number of the matrix $\boldsymbol{H}_n + \rho\boldsymbol{D}^t\boldsymbol{D}$ even further, one can sometimes rephrase the Newton step as iteratively reweighted least squares. For instance, in a generalized linear model, the gradient $\nabla f(\boldsymbol{x})$ and the expected information $\boldsymbol{H}$ can be written as

$$\nabla f(\boldsymbol{x}) = -\boldsymbol{Z}^t \boldsymbol{W}^{1/2} \boldsymbol{r} \quad \text{and} \quad \boldsymbol{H} = \boldsymbol{Z}^t \boldsymbol{W} \boldsymbol{Z},$$

where $\boldsymbol{r}$ is a vector of standardized residuals, $\boldsymbol{Z}$ is a design matrix, and $\boldsymbol{W}$ is a diagonal matrix of case weights (Lange, 2010; Nelder and Wedderburn, 1972). The Newton step is now equivalent to minimizing the least squares criterion

$$\frac{1}{2}\boldsymbol{x}* \boldsymbol{H}_n \boldsymbol{x} - \nabla f(\boldsymbol{x}_n)* \boldsymbol{x} = \left\| \boldsymbol{W}_n^{1/2} \boldsymbol{Z} \boldsymbol{x} - \boldsymbol{W}_n^{-1/2} \nabla f(\boldsymbol{x}_n) \right\|^2$$

$$= \left\| \begin{pmatrix} \boldsymbol{W}_n^{1/2} \boldsymbol{Z} \\ \sqrt{\rho} \boldsymbol{D} \end{pmatrix} \boldsymbol{x} - \begin{pmatrix} \boldsymbol{W}_n^{1/2} \boldsymbol{Z} \boldsymbol{x}_n + \boldsymbol{r}_n \\ \sqrt{\rho} \mathscr{P}(\boldsymbol{D} \boldsymbol{x}_n) \end{pmatrix} \right\|^2.$$

In this context a version of the conjugate gradient algorithm adapted to least squares is attractive. The algorithms LSQR (Paige and Saunders, 1982) and LSMR (Fong and Saunders, 2011) perform well when the design is sparse or ill conditioning is an issue. Given the numerical stability of the iteratively reweighted least squares updates, we favor them over the explicit Newton steps in implementing proximal distance algorithms.

2.2 Proximal Distance by Steepest Descent

In high-dimensional optimization problems, gradient descent is typically employed to avoid matrix inversion. Determination of an appropriate step length is now a primary concern. In the presence of fusion constraints $\boldsymbol{D}\boldsymbol{x} \in S$ and a convex quadratic loss $f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^t \boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}^t \boldsymbol{x}$, the gradient of the proximal distance objective at $\boldsymbol{x}_n$ amounts to

$$\boldsymbol{v}_n = \boldsymbol{A}_n \boldsymbol{x} + \boldsymbol{b} + \rho \boldsymbol{D}^t [\boldsymbol{D}\boldsymbol{x}_n - \mathscr{P}(\boldsymbol{D}\boldsymbol{x}_n)].$$

For the steepest descent update $\boldsymbol{x}_{n+1} = \boldsymbol{x}_n - t_n \boldsymbol{v}_n$, one can show that the optimal step length is

$$t_n = \frac{\|\boldsymbol{v}_n\|^2}{\boldsymbol{v}_n^t \boldsymbol{A} \boldsymbol{v}_n + \rho \|\boldsymbol{D}\boldsymbol{v}_n\|^2}.$$

This update obeys the descent property and avoids matrix inversion. One can also substitute a local convex quadratic approximation around $\boldsymbol{x}_n$ for $f(\boldsymbol{x})$. If the approximation majorizes $f(\boldsymbol{x})$, then the descent property is preserved. In the failure of majorization, the safeguard of step halving is trivial to implement.

In addition to Nesterov acceleration, gradient descent can be accelerated by the subspace MM technique (Chouzenoux et al., 2010). Let $\boldsymbol{G}_n$ be the matrix with k columns determined by the k most current gradients of the objective $h_\rho(\boldsymbol{x})$, including $\nabla h_\rho(\boldsymbol{x}_n)$. Generalizing our previous assumption, suppose $f(x)$ has a quadratic surrogate with Hessian $\boldsymbol{H}_n$ at $\boldsymbol{x}_n$. Overall we get the quadratic surrogate

$$q_\rho(\boldsymbol{x} \mid \boldsymbol{x}_n) = g_\rho(\boldsymbol{x}_n \mid \boldsymbol{x}_n) + \nabla g_\rho(\boldsymbol{x}_n \mid \boldsymbol{x}_n)^t (\boldsymbol{x} - \boldsymbol{x}_n) + \frac{1}{2}(\boldsymbol{x} - \boldsymbol{x}_n)^t (\boldsymbol{H}_n + \rho \boldsymbol{D}^t \boldsymbol{D})(\boldsymbol{x} - \boldsymbol{x}_n).$$

of $g_\rho(x \mid x_n)$. We now seek the best linear perturbation $x_n + G_n \beta$ of x_n by minimizing $q_\rho(x_n + G_n \beta \mid x_n)$ with respect to the coefficient vector β. To achieve this end, we solve the stationary equation

$$0 = G_n^t \nabla g_\rho(x_n \mid x_n) + G_n^t \left(H_n + \rho D^t D\right) G_n \beta,$$

and find $\beta = - \left[G_n^t \left(H_n + \rho D^t D\right) G_n\right]^{-1} G_n^t \nabla g_\rho(x_n \mid x_n)$, where the gradient is

$$\nabla g_\rho(x_n \mid x_n) = \nabla h_\rho(x_n) = \nabla f(x_n) + \rho D^t [Dx_n - \mathscr{P}(Dx_n)].$$

The indicated matrix inverse is just $k \times k$.

2.3 ADMM

ADMM (alternating direction method of multipliers) is a natural competitor to the proximal distance algorithms just described (Hong et al., 2016). ADMM is designed to minimize functions of the form $f(x) + g(Dx)$ subject to $x \in C$, where C is closed and convex. Splitting variables leads to the revised objective $f(x) + g(y)$ subject to $x \in C$ and $y = Dx$. ADMM invokes the augmented Lagrangian

$$\mathscr{A}_\mu(x, y, \lambda) = f(x) + g(y) + \lambda^t (Dx - y) + \frac{\mu}{2} \left\|Dx - y\right\|^2,$$

with Lagrange multiplier λ and step length $\mu > 0$. At iteration $n + 1$ of ADMM one calculates successively

$$x_{n+1} = \operatorname{argmin}_{x \in C} \left[f(x) + \frac{\mu}{2} \left\|Dx - y_n + \lambda_n\right\|^2\right], \tag{4}$$

$$y_{n+1} = \operatorname{argmin}_y \left[g(y) + \frac{\mu}{2} \left\|Dx_{n+1} - y + \lambda_n\right\|^2\right], \text{ and} \tag{5}$$

$$\lambda_{n+1} = \lambda_n + \mu\left(Dx_{n+1} - y_{n+1}\right). \tag{6}$$

Update (4) succumbs to Newton's method when $f(x)$ is smooth and $C = \mathbb{R}^p$, and update (5) reduces to a proximal map of $g(y)$. Update (6) of the Lagrange multiplier λ amounts to steepest ascent on the dual function. A standard extension to the scheme in (4) through (6) is to vary the step length μ by considering the magnitude of residuals (Boyd et al., 2011). For example, letting $r_n = Dx - y$ and $s_n = \mu D^t(y_{n-1} - y_n)$ denote primal and dual residuals at iteration n, we make use of the heuristic

$$\mu_{n+1} = \begin{cases} 2\mu_n, & \text{if } \|r_n\|/\|s_n\| > 10 \\ \mu_n/2, & \text{if } \|r_n\|/\|s_n\| < 10, \\ \mu_n, & \text{otherwise} \end{cases}$$

which (a) keeps the primal and dual residuals within an order of magnitude of each other, (b) makes ADMM less sensitive to the choice of step length, and (c) improves convergence.

Our problem conforms to the ADMM paradigm when S is equal to the Cartesian product $\prod_{i=1}^{r} S_i$ and $g(\boldsymbol{y}) = \frac{\rho}{2}\text{dist}(\boldsymbol{y}, S)^2$. Fortunately, the $\boldsymbol{y}$ update (5) reduces to a simple formula (Bauschke and Combettes, 2017). To derive this formula, note that the proximal map $\boldsymbol{y} = \text{prox}_{\alpha g}(\boldsymbol{z})$ satisfies the stationary condition

$$\boldsymbol{0} = \boldsymbol{y} - \boldsymbol{z} + \alpha[\boldsymbol{y} - \mathscr{P}(\boldsymbol{y})],$$

for any $\boldsymbol{z}$, including $\boldsymbol{z} = \boldsymbol{Dx}_{n+1} + \boldsymbol{\lambda}_n$, and any α, including $\alpha = \rho/\mu$. Since the projection map $\mathscr{P}(\boldsymbol{y})$ has the constant value $\mathscr{P}(\boldsymbol{z})$ on the line segment $[\boldsymbol{z}, \mathscr{P}(\boldsymbol{z})]$, the value

$$\text{prox}_{\alpha g}(\boldsymbol{z}) = \frac{\alpha}{1+\alpha}\mathscr{P}(\boldsymbol{z}) + \frac{1}{1+\alpha}\boldsymbol{z},$$

satisfies the stationary condition. Because the explicit update (5) for $\boldsymbol{y}$ decreases the Lagrangian even when S is nonconvex, we will employ it generally.

The $\boldsymbol{x}$ update (4) is given by the proximal map $\text{prox}_{\mu^{-1}f}(\boldsymbol{\lambda}_n - \boldsymbol{y}_n)$ when $S = \mathbb{R}^p$ and $\boldsymbol{D} = \boldsymbol{I}$. Otherwise, the update of $\boldsymbol{x}$ is more problematic. Assuming $f(\boldsymbol{x})$ is smooth and $S = \mathbb{R}^p$, Newton's method gives the approximate update

$$\boldsymbol{x}_{n+1} = \boldsymbol{x}_n - \left[d^2 f(\boldsymbol{x}_n) + \mu\boldsymbol{D}^t\boldsymbol{D}\right]^{-1}\left[\nabla f(\boldsymbol{x}_n) + \mu\boldsymbol{D}^t(\boldsymbol{Dx}_n - \boldsymbol{y}_n + \boldsymbol{\lambda}_n)\right].$$

Our earlier suggestion of replacing $d^2 f(\boldsymbol{x}_n)$ by a positive definite approximation also applies here. Let us emphasize that ADMM eliminates the need for distance majorization. Although distance majorization is convenient, it is not necessarily a tight majorization. Thus, one can hope to see gains in rates of convergence. Balanced against this positive is the fact that ADMM is often slow to converge to high accuracy.

2.4   Proximal Distance Iteration

We conclude this section by describing proximal distance algorithms in pseudocode. As our theoretical results will soon illustrate, the choice of penalty parameter ρ is tied to the convergence rate of any proximal distance algorithm. Unfortunately, a large value for ρ is necessary for the iterates to converge to the constraint set S. We ameliorate this issue by slowly sending $\rho \to \infty$ according to annealing schedules from the family of geometric progressions $\rho(t) = t^{t-1}$ with t 1. Here we parameterize the family by an initial value $\rho = 1$ and a multiplier $r > 1$. Thus, our methods approximate solutions to min $f(\boldsymbol{x})$ subject to $\boldsymbol{Dx} \in S$ by solving a sequence of increasingly penalized subproblems, min $_{\boldsymbol{x}} h_{\rho(t)}(\boldsymbol{x})$. Algorithm 1 gives a high-level sketch of proximal distance iteration.

Algorithm 1 Proximal Distance Iteration (Summary)
Require: Starting point x_0 and penalty coefficient sequence ρ_t for $t \geq 1$.
1: Initialize solution $x \leftarrow x_0$.
2: **for** $\rho = \rho_1, \rho_2, \rho_3, \ldots$ **do**
3:      Approximate $x_\rho = \operatorname{argmin} h_\rho(x')$ using x as a warm start.
4:      **if** $\operatorname{dist}(Dx_\rho, S) \approx 0$ **then**
5:          break.
6:      **end if**
7:      Set $x \leftarrow x_\rho$.
8: **end for**

In practice we can only solve a finite number of subproblems so we prescribe the following convergence criteria

$$\|\nabla h_{\rho(t)}(\boldsymbol{x}_n)\| \leq \delta_h, \tag{7}$$

$$\operatorname{dist}(\boldsymbol{Dx}_t, S) \leq \delta_d, \text{ or} \tag{8}$$

$$\left| \operatorname{dist}(\boldsymbol{Dx}_t, S) - \operatorname{dist}(\boldsymbol{Dx}_{t-1}, S) \right| \leq \delta_q \left[\operatorname{dist}(\boldsymbol{Dx}_{t-1}, S) + 1 \right]. \tag{9}$$

Condition (7) is a guarantee that a solution estimate $\boldsymbol{x}_n$ is close to a stationary point after n inner iterations for the fixed value of $\rho = \rho(t)$. In conditions (8) and (9), the vector $\boldsymbol{x}_t$ denotes the δ_h-optimal solution estimate once condition (7) is satisfied for a particular subproblem along the annealing path. Condition (8) is a guarantee that solutions along the annealing path adhere to the fusion constraints at level δ_d. In general, condition (8) can only be satisfied for large values of $\rho(t)$. Finally, condition (9) is used to terminate the annealing process if the relative progress made in decreasing the distance penalty becomes too small as measured by δ_q. Algorithm 2 describes the flow of proximal distance iteration in practice. Nesterov acceleration in inner iterations is a key ingredient. Warm starts in solving subsequent subproblems are implicit in our formulation.

3. Convergence Analysis: Convex Case

In this section we summarize convergence results for proximal distance algorithms on convex problems. Proofs of all propositions appear in Section 7. We begin our discussion by recalling the definition of the next iterate

$$\boldsymbol{x}_{n+1} \in \operatorname{argmin} g_\rho(\boldsymbol{x} \mid \boldsymbol{x}_n) \equiv \bigcup_{z_n \in \mathcal{P}(\boldsymbol{Dx}_n)} \operatorname*{argmin}_{\boldsymbol{x}} \left[f(\boldsymbol{x}) + \frac{\rho}{2} \|\boldsymbol{Dx} - z_n\|^2 \right], \tag{10}$$

and the descent property

$$h_\rho(\boldsymbol{x}_{n+1}) \overset{\text{domination}}{\leq} g_\rho(\boldsymbol{x}_{n+1} \mid \boldsymbol{x}_n) \overset{\text{definition}}{\leq} g_\rho(\boldsymbol{x}_n \mid \boldsymbol{x}_n) \overset{\text{tangency}}{=} h_\rho(\boldsymbol{x}_n),$$

enjoyed by all MM algorithms. As noted earlier, we can assume a single fusion matrix $\boldsymbol{D}$ and a single closed convex constraint set S. With these ideas firmly in mind, we state a sufficient condition for the existence of a minimum point of the penalized loss $h_\rho(\boldsymbol{x})$. Further constraints on $\boldsymbol{x}$ beyond those imposed in the distance penalties are rolled into the essential domain of the convex loss $f(\boldsymbol{x})$. This is particularly beneficial for a quadratic loss with an affine constraint (Keys et al., 2019; Lange, 2016).

Algorithm 2 Proximal Distance Iteration
1: Set tolerances $\delta_h, \delta_d, \delta_q$, and annealing schedule $\{\rho(t)\}_{t \geq 1}$.
2: Set maximum outer iterations i_{outer} and maximum inner iterations i_{inner}.
3: Set threshold for Nesterov acceleration, i_{Nesterov}.
4: Initialize $\boldsymbol{x}$ and $q_0 \leftarrow \text{dist}(\boldsymbol{Dx}, S)$.
5: Initialize counter for Nesterov acceleration $i \leftarrow 1$.
6: **for** outer iterations $t = 1, 2, \ldots, i_{\text{outer}}$ **do**
7:    Initialize $\boldsymbol{x}_1 \leftarrow \boldsymbol{x}$ and $\boldsymbol{z}_1 \leftarrow \boldsymbol{x}$.
8:    **for** inner iterations $n = 1, 2, \ldots, i_{\text{inner}}$ **do**
9:       **if** $\|\nabla h_{\rho(t)}(\boldsymbol{x}_n)\| \leq \delta_h$ **then**
10:          Break inner loop.
11:       **else**
12:          Solve the subproblem $\boldsymbol{x}_{n+1} \leftarrow \arg\min g_{\rho(t)}(\boldsymbol{x} \mid \boldsymbol{z}_n)$.   ▷ e.g. using Newton, SD, or ADMM
13:          **if** $h_{\rho(t)}(\boldsymbol{x}_{n+1}) < h_{\rho(t)}(\boldsymbol{x}_n)$ AND $n \geq i_{\text{Nesterov}}$ **then**   ▷ stability check before accelerating
14:            Accelerate $\boldsymbol{z}_{n+1} \leftarrow \boldsymbol{x}_{n+1} + \frac{i-1}{i+2}(\boldsymbol{x}_{n+1} - \boldsymbol{x}_n)$.
15:            Increment $i \leftarrow i + 1$.
16:          **else**
17:            Reset Nesterov acceleration; $i \leftarrow 1$ and $\boldsymbol{z}_{n+1} \leftarrow \boldsymbol{x}_{n+1}$.
18:          **end if**
19:       **end if**
20:    **end for**
21:    Update $\boldsymbol{x} \leftarrow \boldsymbol{x}_n$ and set $q_t \leftarrow \text{dist}(\boldsymbol{Dx}, S)$.
22:    **if** $q_t < \delta_d$ OR $|q_t - q_{t-1}| < \delta_q[1 + q_{t-1}]$ **then**
23:       Break outer loop.
24:    **end if**
25: **end for**

Proposition 3.1 *Suppose the convex function $f(\boldsymbol{x})$ on $\mathbb{R}^p$ possesses a unique minimum point y on the closed convex set $T = \boldsymbol{D}^{-1}(S)$. Then for all sufficiently large ρ, the objective $h_\rho(\boldsymbol{x}) = f(\boldsymbol{x}) + \frac{\rho}{2}\text{dist}(\boldsymbol{Dx}, S)^2$ is coercive and therefore attains its minimum value.*

Next we show that the surrogate function $g_\rho(\boldsymbol{x} \mid \boldsymbol{x}_{n+1})$ defined in equation (3) attains its minimum value for large enough ρ. This ensures that the algorithm map (10) is well-defined.

Proposition 3.2 *Under the conditions of Proposition 3.1, for sufficiently large ρ, every surrogate $g_\rho(\boldsymbol{x} \mid \boldsymbol{x}_n) = f(\boldsymbol{x}) + \frac{\rho}{2}\|\boldsymbol{Dx} - \mathcal{P}(\boldsymbol{Dx}_n)\|^2$ is coercive and therefore attains its minimum value. If*

$$f(\boldsymbol{x}) \geq f(\boldsymbol{x}_n) + \boldsymbol{v}_n^t(\boldsymbol{x} - \boldsymbol{x}_n) + \frac{1}{2}(\boldsymbol{x} - \boldsymbol{x}_n)^t \boldsymbol{A}(\boldsymbol{x} - \boldsymbol{x}_n),$$

for all $\boldsymbol{x}$ and some positive semidefinite matrix $\boldsymbol{A}$ and subgradient $\boldsymbol{v}_n$ at $\boldsymbol{x}_n$, and if the inequality $\boldsymbol{u}^t \boldsymbol{A} \boldsymbol{u} > 0$ holds whenever $\|\boldsymbol{Du}\| = 0$ and $\boldsymbol{u} \neq \boldsymbol{0}$, then for ρ sufficiently large, $g_\rho(\boldsymbol{x} \mid \boldsymbol{x}_n)$ is strongly convex and hence coercive.

As an illustration of Proposition 3.2, suppose that $\boldsymbol{D}$ is the forward difference operator with unit spacing. By virtue of our convexity assumption, it suffices to prove that the surrogate $g_\rho(\boldsymbol{x} \mid \boldsymbol{x}_{n+1})$ is coercive along all rays emanating from the origin. The only vectors with $\boldsymbol{Dv} = \boldsymbol{0}$ are multiples of $\boldsymbol{1}$. Thus, we only need the map $t \mapsto f(t\boldsymbol{1})$ to tend to ∞ as $|t|$ does. This is much weaker condition than the strong convexity of $f(\boldsymbol{x})$. In other words $f(\boldsymbol{x})$ must compensate for the penalty where the penalty is not coercive. Uniqueness of $\boldsymbol{x}_{n+1}$ holds whenever $g_\rho(\boldsymbol{x} \mid \boldsymbol{x}_n)$ is strictly convex regardless of whether the conditions imposed by the proposition are true.

In our ideal convex setting we have a first convergence result for fixed ρ.

Proposition 3.3 *Supposes (a) that S is closed and convex, (b) that the loss $f(\boldsymbol{x})$ is convex and differentiable, and (c) that the constrained problem possesses a unique minimum point. For ρ sufficiently large, let $\boldsymbol{z}_\rho$ denote a minimal point of the objective $h_\rho(\boldsymbol{x})$ defined by* equation (2). *Then the MM iterates* (10) *satisfy*

$$0 \le h_\rho(\boldsymbol{x}_n) - h_\rho(\boldsymbol{z}_\rho) \le \frac{\rho}{2(n+1)} \|\boldsymbol{D}(\boldsymbol{z}_\rho - \boldsymbol{x}_0)\|^2.$$

Furthermore, the iterate values $h_\rho(\boldsymbol{x}_n)$ systematically decrease.

In even more restricted circumstances, one can prove linear convergence of function values in the framework of Karimi et al. (2016). Specifically, our result hinges on deriving a Polyak-Łojasiewicz inequality for (sub)gradients of $h_\rho(\boldsymbol{x})$, from which linear convergence follows almost immediately.

Proposition 3.4 *Suppose that S is a closed and convex set and that the loss $f(\boldsymbol{x})$ is L-smooth and μ-strongly convex. Then the objective $h_\rho(\boldsymbol{x}) = f(\boldsymbol{x}) + \frac{\rho}{2}\mathrm{dist}(\boldsymbol{Dx}, S)^2$ possesses a unique minimum point $\boldsymbol{z}_\rho$, and the proximal distance iterates $\boldsymbol{x}_n$ satisfy*

$$0 \le h_\rho(\boldsymbol{x}_n) - h_\rho(\boldsymbol{z}_\rho) \le \left[1 - \frac{\mu^2}{2\left(L + \rho\|\boldsymbol{D}\|^2\right)^2}\right]^n [h_\rho(\boldsymbol{x}_0) - h_\rho(\boldsymbol{z}_\rho)].$$

Convergence of ADMM is well studied in the optimization literature (Beck, 2017). Appendix B summarizes the main findings as they apply to the ADMM algorithms of Section 2.3.

4. Convergence Analysis: General Case

We now depart the comfortable confines of convexity. Our analysis relies on the Fréchet subdifferential (Kruger, 2003) and the theory of semialgebraic sets and functions. Readers unfamiliar with these topics are encouraged to read Appendix A for a brief review of the relevant theory.

The presentation of our results relies on the prior chain of reasoning established by Keys et al. (2019). Specifically, our arguments invoke Zangwill's global convergence theorem for algorithm maps (Luenberger et al., 1984, Chapter 7, Section 7). The key ingredients of the theory are (i) a solution set (for instance a set of stationary points), (ii) an algorithm map that is closed outside the solution set, (iii) a compact set containing the iterates $\boldsymbol{x}_n$ generated by the map, and (iv) a Lyapunov function that decreases along the iterates. Keys et al. (2019) set the stage in their Propositions 5 through 8 by showing that proximal distance algorithms with $\boldsymbol{D} = \boldsymbol{I}$ satisfy the restrictions (i)-(iv) imposed by Zangwill's global convergence theorem. Note that our algorithm maps inherit their multivalent nature from multivalent Euclidean projections onto nonconvex sets. In any event, the main hurdles to overcome in proving convergence with fixed $\rho > 0$ are

a. establishing coercivity of $h_\rho(\boldsymbol{x})$,

b. demonstrating strong convexity of $g_\rho(\boldsymbol{x} \mid \boldsymbol{x}_n)$, and

c. showing $h_\rho(\boldsymbol{x})$ satisfies a Polyak-Łojasiewicz inequality.

In the present case with D I, the coercivity assumption (a) is tied to requirement (iii) in Zangwill's theorem as stated above. The strong convexity assumption (b) is fortunately imposed on the surrogate rather than the objective. It is also crucial that the Euclidean distance $\text{dist}(x, S)$ to a semialgebraic set S is a semialgebraic function. In view of the closure properties of such functions, the function $\frac{\rho}{2}\text{dist}(Dx, S)^2$ is also semialgebraic. Semialgebraic theory is quite general, and many common set constraints fall within its purview. For example, the nonnegative orthant $\mathbb{R}_+^p$ and the unit sphere $\mathscr{S}^{p-1}$ are semialgebraic.

The next proposition considers the convergence of the iterates of a generic MM algorithm to a stationary point. Readers seeking a proof may consult Section A.10 in the appendix of Keys et al. (2019).

Proposition 4.1 *In an MM algorithm suppose the objective $h(x)$ is coercive, continuous, and subanalytic and all surrogates $g(x \mid x_n)$ are continuous, μ-strongly convex, and satisfy the Lipschitz condition*

$$\|\nabla g(a \mid x_n) - \nabla g(b \mid x_n)\|, \leq L\|a - b\|$$

on the compact set $\{x : h(x) \quad h(x_0)\}$. Then the MM iterates $x_{n+1} = \text{argmin}_X g(x \mid x_n)$ converge to a stationary point.

This result builds on theoretical contributions extending Kurdyka's, Łojasiewicz's, and Polyak's inequalities to nonsmooth analysis, the generic setting of semialgebraic sets and functions, and proximal algorithms (Bolte et al., 2007; Attouch et al., 2010; Kang et al., 2015; Cui et al., 2018; Le Thi et al., 2018). Note that the stationary point may represent a local minimum or even a saddle point rather than a global minimum.

Before stating a precise result for our proposed proximal distance algorithms, let us clarify the nature of the Fréchet subdifferential in the current setting. This entity is determined by the identity

$$\frac{1}{2}\text{dist}(Dx, S)^2 = \frac{1}{2}\min_{z \in S}\left\|Dx - z\right\|^2,$$

for which Danskin's theorem yields the directional derivative

$$d_v\frac{1}{2}\text{dist}(Dx, S)^2 = \min_{z \in S(x)}(Dx - z)^t Dv.$$

Here $S(x)$ is the solution set where the minimum is attained. The Fréchet differential

$$\begin{aligned}\partial^F h_\rho(x) &= \nabla f(x) + \rho\Big\{u : (Dx - z)^t Dv \geq u^t v, z \in S(x) \text{ and all } v\Big\} \\ &= \nabla f(x) + \rho\Big\{u : u = D^t(Dx - z), z \in S(x)\Big\},\end{aligned}$$

holds owing to Corollary 1.12.2 and Proposition 1.17 of Kruger (2003) since $\mathrm{dist}(\boldsymbol{Dx}, S)^2$ is locally Lipschitz. The latter fact follows from the identity $a^2 - b^2 = (a + b)(a - b)$ with $a = \mathrm{dist}(\boldsymbol{Dy}, S)^2$ and $b = \mathrm{dist}(\boldsymbol{Dx}, S)^2$, given that $\mathrm{dist}(\boldsymbol{w}, S)$ is Lipschitz and bounded on bounded sets.

In any event, a stationary point x satisfies $\boldsymbol{0} = \nabla f(\boldsymbol{x}) + \rho \boldsymbol{D}^t(\boldsymbol{Dx} - \boldsymbol{z})$ for all $\boldsymbol{z} \in S(\boldsymbol{x})$. As we expect, the stationary condition is necessary for $\boldsymbol{x}$ to furnish a global minimum. Indeed, if it fails, take $\boldsymbol{z} \in S(\boldsymbol{x})$ with surrogate satisfying $\nabla g_\rho(\boldsymbol{x} \mid \boldsymbol{x}) \neq \boldsymbol{0}$. Then the negative gradient $-\nabla g_\rho(\boldsymbol{x} \mid \boldsymbol{x})$ is a descent direction for $g_\rho(\boldsymbol{x} \mid \boldsymbol{x})$, which majorizes $h_\rho(\boldsymbol{x})$. Hence, $-\nabla g_\rho(\boldsymbol{x} \mid \boldsymbol{x})$ is also a descent direction for $h_\rho(\boldsymbol{x})$. This conclusion is inconsistent with x being a local minimum of the objective.

Having clarified Proposition 4.1 in our context, we state a convergence result on $h_\rho(x)$.

Proposition 4.2 *Suppose in our proximal distance setting that ρ is sufficiently large, the closed constraint sets S_i and the loss $f(\boldsymbol{x})$ are semialgebraic, and $f(\boldsymbol{x})$ is differentiable with a locally Lipschitz gradient. Under the coercive assumption made in Proposition 3.2, the proximal distance iterates $\boldsymbol{x}_n$ converge to a stationary point of the objective $h_\rho(\boldsymbol{x})$.*

The coercivity assumption requires that $h_\rho(\boldsymbol{x})$ be coercive for sufficiently large ρ. This is not as restrictive as it sounds. If S is compact and $f(\boldsymbol{x})$ is convex or bounded below, then the primary hindrances are the directions $\boldsymbol{v}$ where $\boldsymbol{Dv} = \boldsymbol{0}$. Unless $\boldsymbol{D}$ is the trivial matrix $\boldsymbol{0}$, this null space has Lebesgue measure 0.

We conclude this section by communicating two results involving the set $S_k \subset R^p$ whose members have at most k nonzero components. The sparsity constraint defining S_k is usually expressed as $\|\boldsymbol{x}\|_0 \leq k$ for all feasible vectors $\boldsymbol{x}$. Projection of $\boldsymbol{x}$ onto S_k preserves the top k components of $\boldsymbol{x}$ in absolute value but sends all other components to 0. Sparsity sets can be extended to matrix-valued variables $\boldsymbol{X}$ with the sparsity constraint applying to the entries, rows, or columns. Projection in the second case ranks rows by their norms and replaces the lowest ranked rows by the $\boldsymbol{0}$ vector. The next proposition establishes that S_k is semialgebraic in a general setting.

Proposition 4.3 *The order statistics of a finite set $\{f_i(\boldsymbol{x})\}_{i=1}^n$ of semialgebraic functions are semialgebraic. Hence, sparsity sets are semialgebraic.*

For sparsity constrained problems, one can establish a linear rate of convergence under the right hypotheses. The next proposition proves convergence for a wide class of fused models.

Proposition 4.4 *Suppose in our proximal distance setting that ρ is sufficiently large, the constraint set is a sparsity set S_k, and the loss $f(\boldsymbol{x})$ is semialgebraic, strongly convex, and has a Lipschitz gradient. Then the proximal distance iterates $\boldsymbol{x}_n$ converge to a stationary point $\boldsymbol{x}_\infty$. Convergence occurs at a linear rate provided $\boldsymbol{Dx}_\infty$ has k unambiguous largest components in magnitude. When the rows of $\boldsymbol{D}$ are unique, the complementary set of points $\boldsymbol{x}$ where $\boldsymbol{Dx}$ has k ambiguous largest components in magnitude has Lebesgue measure 0.*

5. Numerical Examples

This section considers five concrete examples of constrained optimization amenable to distance majorization with fusion constraints, with $\boldsymbol{D}$ denoting the fusion matrix in each problem. In each case, the loss function is both strongly convex and differentiable. The specific examples that we consider are the metric projection problem, convex regression, convex clustering, image denoising with a total variation penalty, and projection of a matrix to one with a better condition number. Each example is notable for the large number of fusion constraints and projections to convex constraint sets, except in convex clustering. In convex clustering we encounter a sparsity constraint set. Quadratic loss models feature prominently in our examples. Interested readers may consult our previous work for nonconvex examples with $\boldsymbol{D} = \boldsymbol{I}$ (Keys et al., 2019; Xu et al., 2017).

5.1 Mathematical Descriptions

Here we provide the mathematical details for each example.

5.1.1 Metric Projection

Solutions to the metric projection problem restore the triangle inequality to noisy distance data represented as m nodes of a graph (Brickell et al., 2008; Sra et al., 2005). Specifically, data are encoded in an $m \times m$ dissimilarity matrix $\boldsymbol{Y} = (y_{ij})$ with nonnegative weights in the matrix $\boldsymbol{W} = (w_{ij})$. Metric projection requires finding the symmetric semi-metric $\boldsymbol{X} = (x_{ij})$ minimizing

$$f(\boldsymbol{X}) = \sum_{i > j} w_{ij}(x_{ij} - y_{ij})^2,$$

subject to all $\binom{m}{2}$ nonnegativity constraints $x_{ij} \geq 0$ and all $3\binom{m}{3}$ triangle inequality constraints $x_{ij} - x_{ik} - x_{kj} \leq 0$. The diagonal entries of $\boldsymbol{Y}$, $\boldsymbol{W}$, and $\boldsymbol{X}$ are zero by definition. The fusion matrix $\boldsymbol{D}$ has $\binom{m}{2} + 3\binom{m}{3}$ rows, and the projected value of $\boldsymbol{DX}$ must fall in the set S of symmetric matrices satisfying all pertinent constraints.

One can simplify the required projection by stacking the nonredundant entries along each successive column of $\boldsymbol{X}$ to create a vector $\boldsymbol{x}$ with $\binom{m}{2}$ entries. This captures the lower triangle of $\boldsymbol{X}$. The sparse matrix $\boldsymbol{D}$ is correspondingly redefined to be $\left[\binom{m}{2} + 3\binom{m}{3}\right] \times \binom{m}{2}$. These maneuvers simplify constraints to $\boldsymbol{Dx} \geq \boldsymbol{0}$, and projection involves sending each entry u of $\boldsymbol{Dx}$ to $\max\{0, u\}$. Putting everything together, the objective to minimize is

$$h_\rho(\boldsymbol{x}) = \frac{1}{2}\left\|\boldsymbol{W}^{1/2}(\boldsymbol{x} - \boldsymbol{y})\right\|_2^2 + \frac{\rho}{2}\mathrm{dist}\left(\boldsymbol{Tx}, \mathbb{R}_+^{p_1}\right)^2 + \frac{\rho}{2}\mathrm{dist}\left(\boldsymbol{x}, \mathbb{R}_+^{p_2}\right)^2,$$

where $\boldsymbol{D}$ consists of blocks $\boldsymbol{T}$ and $\boldsymbol{I}_{p2}$ and p_1 and p_2 count the number of triangle inequality and nonnegativity constraints, respectively. The linear system $(\boldsymbol{I} + \rho\boldsymbol{D}^t\boldsymbol{D})\boldsymbol{x} = \boldsymbol{b}$ appears in both the MM and ADMM updates for $\boldsymbol{x}_n$. Application of the Woodbury and Sherman-

Morrison formulas yield an exact solution to the linear system and allow one to forgo iterative methods. The interested reader may consult Appendix C for further details.

5.1.2 Convex Regression—Convex regression is a nonparametric method for estimating a regression function under shape constraints. Given m responses y_i and corresponding predictors $x_i \in \mathbb{R}^d$, the goal is to find the convex function $\psi(x)$ minimizing the sum of squares $\frac{1}{2}\sum_{i=1}^m [y_i - \psi(x_i)]^2$. Asymptotic and finite sample properties of this convex estimator have been described in detail by Seijo and Sen (2011). In practice, a convex regression program can be restated as a finite dimensional problem of finding the value θ_i and subgradient $\xi_i \in \mathbb{R}^d$ of $\psi(x)$ at each sample point (y_i, x_i). Convexity imposes the supporting hyperplane constraint $\theta_j + \xi_j^t(x_i - x_j) \leq \theta_i$ for each pair $i \quad j$. Thus, the problem becomes one of minimizing $\frac{1}{2}\|y - \theta\|^2$ subject to these $m(m-1)$ inequality constraints. In the proximal distance framework, we must minimize

$$h_\rho(\theta, \Xi) = \frac{1}{2}\|y - \theta\|^2 + \frac{\rho}{2}\text{dist}\big(A\theta + B\Xi, \mathbb{R}_-^{m(m-1)}\big)^2,$$

where $D = [A \; B]$ encodes the required fusion matrix. The reader may consult Appendix D for a description of each algorithm map.

5.1.3 Convex Clustering—Convex clustering of m samples based on d features can be formulated in terms of the regularized objective

$$F_\gamma(U) = \frac{1}{2}\sum_{i=1}^m \|u_i - x_i\|^2 + \gamma \sum_{i > j} w_{ij}\|u_i - u_j\|,$$

based on columns x_i and u_i of $X \in \mathbb{R}^{d \times m}$ and $U \in \mathbb{R}^{d \times m}$, respectively. Here each $x_i \in \mathbb{R}^d$ is a sample feature vector and the corresponding u_i represents its centroid assignment. The predetermined weights w_{ij} have a graphical interpretation under which similar samples have positive edge weights w_{ij} and distant samples have 0 edge weights. The edge weights are chosen by the user to guide the clustering process. In general, minimization of $F_\gamma(U)$ separates over the connected components of the graph. To allow all sample points to coalesce into a single cluster, we assume that the underlying graph is connected. The regularization parameter $\gamma > 0$ tunes the number of clusters in a nonlinear fashion and potentially captures hierarchical information. Previous work establishes that the solution path $U(\gamma)$ varies continuously with respect to γ (Chi and Lange, 2015). Unfortunately, there is no explicit way to determine the number of clusters entailed by a particular value of γ prior to fitting $U(\gamma)$.

Alternatively, we can attack the problem using sparsity and distance majorization. Consider the penalized objective

$$h_{\rho,k}(\boldsymbol{U}) = \frac{1}{2}\left\|\boldsymbol{U} - \boldsymbol{X}\right\|_F^2 + \frac{\rho}{2}\mathrm{dist}(\boldsymbol{U}\boldsymbol{D}, S_k)^2 .$$

The fusion matrix $\boldsymbol{D}$ has $\binom{m}{2}$ columns $w_{ij}(\boldsymbol{e}_i - \boldsymbol{e}_j)$ and serves to map the centroid matrix $\boldsymbol{U}$ to a $d \times \binom{m}{2}$ matrix $\boldsymbol{V}$ encoding the weighted differences $w_{ij}(\boldsymbol{u}_i - \boldsymbol{u}_j)$. The members of the sparsity set S_k are $d \times \binom{m}{2}$ matrices with at most k non-zero columns. Projection of $\boldsymbol{UD}$ onto the closed set S_k forces some centroid assignments to coalesce, and is straightforward to implement by sorting the Euclidean lengths of the columns of $\boldsymbol{UD}$ and sending to $\boldsymbol{0}$ all but the k most dominant columns. Ties are broken arbitrarily.

Our sparsity-based method trades the continuous penalty parameter $\gamma > 0$ in the previous formulation for an integer sparsity index $k \in \left\{0, 1, 2, ..., \binom{m}{2}\right\}$. For example with $k = 0$, all differences $\boldsymbol{u}_i - \boldsymbol{u}_j$ are coerced to $\boldsymbol{0}$, and all sample points cluster together. The other extreme $k = \binom{m}{2}$ assigns each point to its own cluster. The size of the matrices $\boldsymbol{D}$ and $\boldsymbol{UD}$ can be reduced by discarding column pairs corresponding to 0 weights. Appendix E describes the projection onto sparsity sets and provides further details.

5.1.4   Total Variation Image Denoising—To approximate an image $\boldsymbol{U}$ from a noisy input $\boldsymbol{W}$ matrix, Rudin et al. (1992) regularize a loss function $f(\boldsymbol{U})$ by a total variation (TV) penalty. After discretizing the problem, the least squares loss leads to the objective

$$F_\gamma(\boldsymbol{U}) = \sum_{i,j}(U_{i,j} - W_{i,j})^2 + \gamma\sum_{i,j}\sqrt{(U_{i+1,j} - U_{i,j})^2 + (U_{i,j+1} - U_{i,j})^2},$$

where $\boldsymbol{U}, \boldsymbol{W} \in \mathbb{R}^{m \times p}$ are rectangular monochromatic images and γ controls the strength of regularization. The anisotropic norm

$$\mathrm{TV}_1(\boldsymbol{U}) = \sum_{i,j}|U_{i+1,j} - U_{i,j}| + |U_{i,j+1} - U_{i,j}| = \|\boldsymbol{D}_m\boldsymbol{U}\|_1 + \|\boldsymbol{U}\boldsymbol{D}_p^t\|_1,$$

is often preferred because it induces sparsity in the differences. Here $\boldsymbol{D}_p$ is a forward difference operator on p data points. Stacking the columns of $\boldsymbol{U}$ into a vector $\boldsymbol{u} = \mathrm{vec}(\boldsymbol{U})$ allows one to identify a fusion matrix $\boldsymbol{D}$ and write $\mathrm{TV}_1(\boldsymbol{U})$ compactly as $\mathrm{TV}_1(\boldsymbol{u}) = \|\boldsymbol{Du}\|_1$. We append a row with a single 1 in the last component to make $\boldsymbol{D}$ full rank. In this context we reformulate the denoising problem as minimizing $f(\boldsymbol{u})$ subject to the set constraint $\|\boldsymbol{Du}\|_1$ γ. This revised formulation directly quantifies the quality of a solution in terms of its total variation and brings into play fast pivot-based algorithms for projecting onto multiples of the ℓ_1 unit ball (Condat, 2016). Appendix F provides descriptions of each algorithm.

5.1.5 Projection of a Matrix to a Good Condition Number—Consider an $m \times p$ matrix M with $m \geq p$ and full singular value decomposition $M = U\Sigma V^t$. The condition number of M is the ratio $\sigma_{\max}/\sigma_{\min}$ of the largest to the smallest singular value of M. We denote the diagonal of Σ as σ. Owing to the von Neumann-Fan inequality, the closest matrix N to M in the Frobenius norm has the singular value decomposition $N = UXV^t$, where the diagonal x of X satisfies inequalities pertinent to a decent condition number (Borwein and Lewis, 2010). Suppose $c \geq 1$ is the maximum condition number. Then every pair (x_i, x_j) satisfies $x_i - cx_j \geq 0$. Note that $x_i - cx_i > 0$ if and only if $x_i < 0$. Thus, nonnegativity of the entries of x is enforced. The proximal distance approach to the condition number projection problem invokes the objective and majorization

$$
\begin{aligned}
h_\rho(x) &= \frac{1}{2}\left\|\sigma - x\right\|^2 + \frac{\rho}{2}\sum_{(i,j)}\mathrm{dist}(x_i - cx_j, \mathbb{R}_-)^2 \\
&= \frac{1}{2}\left\|\sigma - x\right\|^2 + \frac{\rho}{2}\sum_{(i,j)}(x_i - cx_j)_+^2 \\
&\leq \frac{1}{2}\left\|\sigma - x\right\|^2 + \frac{\rho}{2}\sum_{(i,j)}(x_i - cx_j - q_{nij})^2,
\end{aligned}
$$

at iteration n, where $q_{nij} = \min\{x_{ni} - cx_{nj}, 0\}$. We can write the majorization more concisely as

$$
h_\rho(x) \leq \frac{1}{2}\|A_\rho x - r_n\|^2, \quad A_\rho = \begin{pmatrix} I_p \\ \sqrt{\rho}D \end{pmatrix}, \quad r_n = \begin{pmatrix} \sigma \\ \sqrt{\rho}\,\mathrm{vec}\,Q_n \end{pmatrix},
$$

where $\mathrm{vec}\,Q_n$ stacks the columns of $Q_n = (q_{nij})$ and the $p^2 \times p$ fusion matrix D satisfies $(Dx)_k = x_i - cx_j$ for each component k. The minimum of the surrogate occurs at the point $x_{n+1} = (A_\rho^t A_\rho)^{-1}A_\rho^t r_n$. This linear system can be solved exactly. Appendix G provides additional details.

5.2 Numerical Results

Our numerical experiments compare various strategies for implementing Algorithm 2. We consider two variants of proximal distance algorithms. The first directly minimizes the majorizing surrogate (MM), while the second performs steepest descent (SD) on it. In addition to the aforementioned methods, we tried the subspace MM algorithm described in Section 2.2. Unfortunately, this method was outperformed in both time and accuracy comparisons by Nesterov accelerated MM; the MM subspace results are therefore omitted. We also compare our proximal distance approach to ADMM as described in Section 2.3. In many cases updates require solving a large linear system; we found that the method of conjugate gradients sacrificed little accuracy and largely outperformed LSQR and therefore omit comparisons. The clustering and denoising examples are exceptional in that the associated matrices $D^t D$ are sufficiently ill-conditioned to cause failures in conjugate gradients. Table 1 summarizes choices in control parameters across each example.

We now explain example by example the implementation details behind our efforts to benchmark the three strategies (MM, SD, and ADMM) in implementing Algorithm 2. In

each case we initialize the algorithm with the solution of the corresponding unconstrained problem. Performance is assessed in terms of speed in seconds or milliseconds, number of iterations until convergence, the converged value of the loss $f(x)$, and the converged distance to the constraint set dist(Dx, S), as described in Algorithm 2. Additional metrics are highlighted where applicable. The term *inner iterations* refers to the number of iterations to solve a penalized subproblem argmin $h_\rho(x)$ for a given ρ whereas *outer iterations* count the total number of subproblems solved. Lastly, we remind readers that the approximate solution to argmin $h_{\rho(t)}(x)$ is used as a warm start in solving argmin $h_{\rho(t+1)}(x)$.

5.2.1   Metric Projection.—In our comparisons, we use input matrices $Y \in \mathbb{R}^{m \times m}$ whose iid entries y_{ij} are drawn uniformly from the interval [0, 10] and set weights $w_{ij} = 1$. Each algorithm is allotted a maximum of 200 outer and 10^5 inner iterations, respectively, to achieve a gradient norm of $\delta_h = 10^{-3}$ and distance to feasibility of $\delta_d = 10^{-2}$. The relative change parameter is set to $\delta_q = 0$ and the annealing schedule is set to $\rho(t) = \min\{10^8, 1.2^{t-1}\}$ for the proximal distance methods. Table 2 summarizes the performance of the three algorithms. Best values appear in boldface. All three algorithms converge to a similar solution as indicated by the final loss $\|y - x\|^2$ and distance values. It is clear that SD matches or outperforms MM and ADMM approaches on this example. Notably, the linear system appearing in the MM update admits an exact solution (see Appendix C.5), yet SD has a faster time to solution with fewer iterations taken. The selected convergence metrics in Figure 1 vividly illustrate stability of solutions $x_\rho = $ argmin $h_\rho(x)$ along an annealing path from $\rho = 1$ to $\rho = 1.2^{40} \approx 1470$. Specifically, solving each penalized subproblem along the sequence results in marginal increase in the loss term with appreciable decrease in the distance penalty. Except for the first outer iteration, there is minimal decrease of the loss, distance penalty, or penalized objective within a given outer iteration even as the gradient norm vanishes. The observed tradeoff between minimizing a loss model and minimizing a nonnegative penalty is well-known in penalized optimization literature (Beltrami, 1970; Lange, 2016, see Proposition 7.6.1 on p. 183).

5.2.2   Convex Regression.—In our numerical examples the observed functional values y_i are independent Gaussian deviates with means $\psi(x_i)$ and common variance $\sigma^2 = 0.1$. The predictors are iid deviates sampled from the uniform distribution on $[-1, 1]^d$. We choose the simple convex function $\psi(x_i) = \|x_i\|^2$ for our benchmarks for ease in interpretation; the interested reader may consult the work of Mazumder et al. (2019) for a detailed account of the applicability of the technique in general. Each algorithm is allotted a maximum of 200 outer and 10^4 inner iterations, respectively, to converge with $\delta_h = 10^{-3}$, $\delta_d = 10^{-2}$, and $\delta_q = 10^{-6}$. The annealing schedule is set to $\rho(t) = \min\{10^8, 1.2^{t-1}\}$.

Table 3 demonstrates that although the SD approach is appreciably faster than both MM and ADMM, the latter appear to converge on solutions with marginal improvements in minimizing the loss $\|y - \theta\|^2$, distance, and mean squared error (MSE) measured using ground truth functional values $\psi(x_i)$ and estimates θ_i. Interestingly, increasing both the number of features and samples does not necessarily increase the amount of required computational time in using a proximal distance approach; for example, see results with $d = 2$ and $d = 20$ features. This may be explained by sensitivity to the annealing schedule.

5.2.3 Convex Clustering.—To evaluate the performance of the different methods on convex clustering, we consider a mixture of simulated data and discriminant analysis data from the UCI Machine Learning Repository (Dua and Graff, 2019). The simulated data in `gaussian300` consists of 3 Gaussian clusters generated from bivariate normal distributions with means $\mu = (0.0, 0.0)^t$, $(2.0, 2.0)^t$, and $(1.8, 0.5)^t$, standard deviation $\sigma = 0.1$, and class sizes $n_1 = 150$, $n_2 = 50$, $n_3 = 100$. This easy dataset is included to validate Algorithm 3 described later as a reasonable solution path heuristic. The data in `iris` and `zoo` are representative of clustering with purely continuous or purely discrete data, respectively. In these two datasets, samples with same class label form a cluster. Finally, the simulated data `spiral500` is a classic example that thwarts *k*-means clustering. Each algorithm is allotted a maximum of 10^4 inner iterations to solve a ρ-penalized subproblem at level $\delta_h = 10^{-2}$. The annealing schedule is set to $\rho(t) = \min\{10^8, 1.2^{t-1}\}$ over 100 outer iterations with $\delta_d = 10^{-5}$ and $\delta q = 10{-6}$.


```
Algorithm 3 Search Heuristic
1: procedure CVXCLUSTERPATH(X, s₀, s_step)
2:     U ← X                                                    ▷ Initialize centroid assignments.
3:     s ← s₀                                                   ▷ Initialize sparsity level in [0, 1].
4:     k_max ← (ᵐ₂)                                             ▷ Determine upper bound from samples m.
5:     while s < 1 do
6:         k ← round((1 − s) × k_max)                           ▷ Set parameter for sparsity projection.
7:         U_s ← argmin h_ρ,k(U)                                ▷ Minimize with choice of k.
8:         c ← count(U_s)                       ▷ Count the number of coalesced centroid differences.
9:         s_proposal ← c/k_max             ▷ Propose a new sparsity level based on satisfied constraints.
10:        if s_proposal > s + s_step then
11:            s ← s_proposal                                   ▷ Accept proposal if it increases sparsity.
12:        else
13:            s ← s + s_step                                   ▷ Otherwise move by a fixed amount.
14:        end if
15:    end while
16: end procedure
```

Because the number of clusters is usually unknown, we implement the search heuristic outlined in Algorithm 3. The idea behind the heuristic is to gradually coerce clustering without exploring the full range of the hyperparameter *k*. As one decreases the number of admissible nonzero centroid differences *k* from k_{max} to 0, sparsity $(1 - k/k_{max})$ in the columns of **UD** increases to reflect coalescing centroid assignments. Thus, Algorithm 1 generates a list of candidate clusters that can be evaluated by various measures of similarity (Vinh et al., 2010). For example, the adjusted Rand index (ARI) provides a reasonable measure of the distance to the ground truth in our examples as it accounts for both the number of identified clusters and cluster assignments. We also report the related normed Mutual Information (NMI). The ARI takes values on $[-1, 1]$ whereas NMI appears on a $[0, 1]$ scale.

ADMM, as implemented here, is not remotely competitive on these examples given its extremely long compute times and failure to converge in some instances. These times are only exacerbated by the search heuristic and therefore omit ADMM from this example. The findings reported in Table 4 indicate the same accuracy for MM (using LSQR) and SD as measured by loss and distance to feasibility. Here we see that the combination of the proximal distance algorithms and the overall search heuristic (Algorithm 3) yields perfect clusters in the `gaussian300` example on the basis of ARI and NMI. To its disadvantage, the search heuristic is greedy and generally requires tuning. Both MM and SD achieve similar clusters as indicated by ARI and NMI. Notably, SD generates candidate clusters faster than MM.

5.2.4   Total Variation Image Denoising.—To evaluate our denoising algorithm, we consider two standard test images, cameraman and peppers_gray. White noise with $\sigma =$ 0.2 is applied to an image and then reconstructed using our proximal distance algorithms. Only MM and SD are tested with a maximum of 100 outer and 10^4 inner iterations, respectively, and convergence thresholds $\delta_h = 10^{-1}$, $\delta_d = 10^{-1}$, and $\delta_q = 10^{-6}$. A moderate schedule $\rho(t) = \{10^8, 1.5^{t-1}\}$ performs well even with such lax convergence criteria. Table 5 reports convergence metrics and image quality indices, MSE and Peak Signal-to-Noise Ratio (PSNR). Timings reflect the total time spent generating solutions, starting from a 0% reduction in the total variation of the input image U up to 90% reduction in increments of 10%. Explicitly, we take $\gamma_0 = \mathrm{TV}_1(U)$ and vary the control parameter $\gamma = (1 - s)\gamma_0$ with $s \in$ [0, 1] to control the strength of denoising. Figure 5.2.4 depicts the original and reconstructed images along the solution path.

5.2.5   Projection of a Matrix to a Good Condition Number.—We generate base matrices $M \in \mathbb{R}^{p \times p}$ as random correlation matrices using MatrixDepot.jl (Zhang and Higham, 2016), which relies on Davies' and Higham's refinement (Davies and Higham, 2000) of the Bendel-Mickey algorithm (Bendel and Mickey, 1978). Simulations generate matrices with condition numbers c(M) in the set {119, 1920, 690}. Our subsequent analyses target condition number decreases by a factor a such $a \in$ {2, 4, 16, 32}. Each algorithm is allotted a maximum of 200 outer and 10^4 inner iterations, respectively with choices $\delta_h = 10^{-3}$, $\delta_d = 10^{-2}$, $\delta_q = 10^{-6}$, and $\rho(t) = 1.2^{t-1}$. Table 6 summarizes the performance of the three algorithms. The quality of approximate solutions is similar across MM, SD, and ADMM in terms of loss, distance, and final condition number metrics. Interestingly, the MM approach requires less time to deliver solutions of comparable quality to SD solutions as the size p of the input matrix $M \in \mathbb{R}^{p \times p}$ increases.

6.   Discussion

We now recapitulate the main findings of our numerical experiments. Tables 2 through 6 show a consistent pattern of superior speed by the steepest descent (SD) version of the proximal distance algorithm. This is hardly surprising since unlike ADMM and MM, SD avoids solving a linear system at each iteration. SD's speed advantage tends to persist even when the linear system can be solved exactly. The condition number example summarized in Table 6 is an exception to this rule. Here the MM updates leverage a very simple matrix inverse. MM is usually faster than ADMM. We attribute MM's performance edge to the extra matrix-vector multiplications involving the fusion matrix D required by ADMM. In fairness, ADMM closes the speed gap and matches MM on convex regression.

The choice of annealing schedule can strongly impact the quality of solutions. Intuitively, driving the gradient norm $\|\nabla h_\rho(x)\|$ to nearly 0 for a given ρ keeps the proximal distance methods on the correct annealing path and yields better approximations. Provided the penalized objective is sufficiently smooth, one expects the solution $x_\rho \approx \mathrm{argmin}\ h_\rho(x)$ to be close to the solution $x_{\rho'} \approx \mathrm{argmin}\ h_{\rho'}(x)$ when the ratio $\rho'/\rho > 1$ is not too large. Thus, choosing a conservative δ_h for the convergence criterion $\|\nabla h_\rho(x)\| \quad \delta_h$ may guard against a poorly specified annealing schedule. Quantifying sensitivity of intermediate solutions

x_ρ with respect to ρ is key in avoiding an increase in inner iterations per subproblem; for example, as observed in Figure 1. Given the success of our practical annealing recommendation to overcome the unfortunate coefficients in Propositions 3.3 and 3.4, this topic merits further consideration in future work.

In practice, it is sometimes unnecessary to impose strict convergence criteria on the proximal distance iterates. It is apparent that the convergence criteria on convex clustering and image denoising are quite lax compared to choices in other examples, specifically in terms of δ_h. Figure 1 suggests that most of the work in metric projection involves driving the distance penalty downhill rather than in fitting the loss. Surprisingly, Table 4 shows that our strict distance criterion $\delta_d = 10^{-5}$ in clustering is achieved. This implies dist$(\boldsymbol{Dx}, S)^2 \quad 10^{-10}$ on the selected solutions with $\rho \quad 10^8$, yet we only required $\|\nabla h_\rho(\boldsymbol{x})\| \quad 10^{-2}$ on each subproblem. Indeed, not every problem may benefit from precise solution estimates. The image processing example underscores this point as we are able to recover denoised images with the choices $\delta_h = \delta_d = 10^{-1}$. Problems where patterns or structure in solutions are of primary interest may stand to benefit from relaxed convergence criteria.

Our proximal distance method, as described in Algorithm 2, enjoys several advantages. First, fusion constraints fit naturally in the proximal distance framework. Second, proximal distances enjoy the descent property. Third, there is a nearly optimal step size for gradient descent when second-order information is available on the loss. Fourth, proximal distance algorithms are competitive if not superior to ADMM on many problems. Fifth, proximal distance algorithms like iterative hard thresholding rely on set projection and are therefore helpful in dealing with hard sparsity constraints. The main disadvantages of the proximal distance methods are (a) the overall slow convergence due to the loss of curvature information on the distance penalty and (b) the need for a reasonable annealing schedule. In practice, a little experimentation can yield a reasonable schedule for an entire class of problems. Many competing methods are only capable of dealing with soft constraints imposed by the lasso and other convex penalties. To their detriment, soft constraints often entail severe parameter shrinkage and lead to an excess of false positives in model selection.

Throughout this manuscript we have stressed the flexibility of the proximal distance framework in dealing with a wide range of constraints as a major strength. From our point of view, proximal distance iteration adequately approximates feasible, locally optimal solutions to constrained optimization problems for well behaved constraint sets, for instance convex sets or semialgebraic sets. Combinatorially complex constraints or erratic loss functions can cause difficulties for the proximal distance methods. The quadratic distance penalty dist$(\boldsymbol{Dx}, S)^2$ is usually not an issue, and projection onto the constraint should be fast. Poor loss functions may either lack second derivatives or may possess a prohibitively expensive and potentially ill-conditioned Hessian $d^2 f(\boldsymbol{x})$. In this setting techniques such as coordinate descent and regularized and quasi-Newton methods are viable alternatives for minimizing the surrogate $g_\rho(\boldsymbol{x} \mid \boldsymbol{x}_n)$ generated by distance majorization. In any event, it is crucial to design a surrogate that renders each subproblem along the annealing path easy to solve. This may entail applying additional majorizations in $f(\boldsymbol{x})$. Balanced against this possibility is the sacrifice of curvature information with each additional majorization.

We readily acknowledge that other algorithms may perform better than MM and proximal distance algorithms on specific problems. The triangle fixing algorithm for metric projection is a case in point (Brickell et al., 2008), as are the numerous denoising algorithms based on the ℓ_1 norm. This objection obscures the generic utility of the proximal distance principle. ADMM can certainly be beat on many specific problems, but nobody seriously suggests that it be rejected across the board. Optimization, particularly constrained optimization, is a fragmented subject, with no clear winner across problem domains. Generic methods serve as workhorses, benchmarks, and backstops.

As an aside, let us briefly note that ADMM can be motivated by the MM principle, which is the same idea driving proximal distance algorithms. The optimal pair (x, y) and λ furnishes a stationary point of the Lagrangian. Because the Lagrangian is linear in λ, its maximum for fixed (x, y) is ∞. To correct this defect, one can add a viscosity minorization to the Lagrangian. This produces the modified Lagrangian

$$\mathscr{L}_\mu(x, y, \lambda) = f(x) + g(y) + \mu \lambda^t (Dx - y) + \frac{\mu}{2}\left\|Dx - y\right\|^2 - \frac{\alpha}{2}\|\lambda - \lambda_n\|^2.$$

The penalty term has no impact on the x and y updates. However, the MM update for λ is determined by the stationary condition

$$0 = \mu(Dx_{n+1} - y_{n+1}) - \alpha(\lambda - \lambda_n),$$

so that

$$\lambda_{n+1} = \lambda_n + \frac{\mu}{\alpha}(Dx_{n+1} - y_{n+1}).$$

The choice $\alpha = 1$ gives the standard ADMM update. Thus, the ADMM algorithm alternates decreasing and increasing the Lagrangian in a search for the saddlepoint represented by the optimal trio (x, y, λ).

In closing we would like to draw the reader's attention to some generalizations of the MM principle and connections to other well-studied algorithm classes. For instance, a linear fusion constraint $Dx \in S$ can in principle by replaced by a nonlinear fusion constraint $M(x) \in S$. The objective and majorizer are then

$$h_\rho(x) = f(x) + \frac{\rho}{2}\text{dist}[M(x), S]^2$$

$$g(x \mid x_n) = f(x) + \frac{\rho}{2}\|M(x) - \mathscr{P}_S[M(x_n)]\|^2.$$

The objective has gradient $g = \nabla f(x) + \rho dM(x)^t\{M(x) - \mathscr{P}_S[M(x)]\}$. The second differential of the majorizer is approximately $d^2 f(x) + \rho dM(x)^t dM(x)$ for $M(x)$ close to $\mathscr{P}_S[M(x)]$. Thus, gradient descent can be implemented with step size

$$\gamma = \frac{\|g_n\|^2}{g_n^t d^2 f(x_n) g_n + \rho \|dM(x_n)g_n\|^2},$$

assuming the denominator is positive.

Algebraic penalties such as $\|g(x)\|^2$ reduce to distance penalties with constraint set $\{0\}$. The corresponding projection operator sends any vector y to 0 so that the algebraic penalty $\|g(x)\|^2 = \text{dist}[g(x), \{0\}]^2$. This observation is pertinent to constrained least squares with $g(x) = d - Cx$ (Golub and Van Loan, 1996). The proximal distance surrogate can be expressed as

$$\frac{1}{2}\left\|y - Ax\right\|^2 + \frac{\rho}{2}\left\|d - Cx\right\|^2 = \frac{1}{2}\left\|\begin{pmatrix} y \\ \sqrt{\rho}d \end{pmatrix} - \begin{bmatrix} A \\ \sqrt{\rho}C \end{bmatrix}x\right\|^2,$$

and minimized by standard least squares algorithms. No annealing is necessary. Inequality constraints $g(x) \le 0$ behave somewhat differently. The proximal distance majorization $\text{dist}[g(x), \mathbb{R}_-^m]^2 \le \|g(x) - \mathscr{P}_{\mathbb{R}_-^m}[g(x_n)]\|^2$ is not the same as the Beltrami quadratic penalty $g(x)_+^2$ (Beltrami, 1970). However, the standard majorization (Lange, 2016)

$$g(x)_+^2 \le \|g(x) - \mathscr{P}_{\mathbb{R}_-^m}[g(x_n)]\|^2,$$

brings them back into alignment.

7. Proofs

In this section we provide proofs for the convergence results discussed in Section 3 and Section 4 for the convex and nonconvex cases, respectively.

7.1 Proposition 3.1

Proof Without loss of generality we can translate the coordinates so that $y = 0$. Let B be the unit sphere $\{x : \|x\| = 1\}$. Our first aim is to show that $h_\rho(x) > f(0)$ throughout B. Consider the set $B \cap T$, which is possibly empty. On this set the infimum b of $f(x)$ is attained, so $b > f(0)$ by assumption. The set $B \backslash T$ will be divided into two regions, a narrow zone adjacent to T and the remainder. Now let us show that there exists a $\delta > 0$ such that $h_\rho(x) \ge f(x) \ge f(0) + \delta$ for all $x \in B$ with $\text{dist}(Dx, S) \ge \delta$. If this is not so, then there exists a sequence $x_n \in B$ with $f(x_n) < f(0) + \frac{1}{k}$ and $\text{dist}(Dx_n, S) \le \frac{1}{k}$. By compactness, some subsequence of x_n converges to $z \in B \cap T$ with $f(z) \le f(0)$, contradicting the uniqueness of y. Finally, let $a = \min_{x \in B} f(x)$. To deal with the remaining region take ρ large enough so that $a + \frac{\rho}{2}\delta^2 > f(0)$. For such ρ, $h_\rho(x) > f(0)$ everywhere on B. It follows that on the unit ball $\{x : \|x\| \le 1\}$, $h_\rho(x)$ is minimized at an interior point. Because $h_\rho(x)$ is convex, a local minimum is necessarily a global minimum.

To show that the objective $h_\rho(x)$ is coercive, it suffices to show that it is coercive along every ray $\{tv : t \geq 0, \|v\| = 1\}$ (Lange, 2016). The convex function $r(t) = h_\rho(tv)$ satisfies $r(t) \geq r(1) + r'_+(1)(t-1)$. Because $r(0) < r(1)$, the point 1 is on the upward slope of $r(t)$, and the one-sided derivative $r'_+(1) > 0$. Coerciveness follows from this observation. ∎

7.2 Proposition 3.2

Proof The first assertion follows from the bound $g_\rho(x \mid x_n) \geq h_\rho(x)$. To prove the second assertion, we note that it suffices to prove the existence of some constant $\rho > 0$ such that the matrix $A + \rho D^t D$ is positive definite (Debreu, 1952). If no choice of ρ renders $A + \rho D^t D$ positive definite, then there is a sequence of unit vectors u_m and a sequence of scalars ρ_m tending to ∞ such that

$$u_m^t A u_m + \rho_m u_m^t D^t D u_m \leq 0. \tag{11}$$

By passing to a subsequence if needed, we may assume that the sequence u_m converges to a unit vector u. On the one hand, because $D^t D$ is positive semidefinite, inequality (11) compels the conclusions $u_m^t A u_m \leq 0$, which must carry over to the limit. On the other hand, dividing inequality (11) by ρ_m and taking limits imply $u^t D^t D u \leq 0$ and therefore $\|Du\| = 0$. Because the limit vector u violates the condition $u^t A u > 0$, the required $\rho > 0$ exists. ∎

7.3 Proposition 3.3

Proof Systematic decrease of the iterate values $h_\rho(x_n)$ is a consequence of the MM principle. The existence of z_ρ follows from Proposition 3.1. To prove the stated bound, first observe that the function $g_\rho(x \mid x_n) - \frac{\rho}{2}\|Dx\|^2$ is convex, being the sum of the convex function $f(x)$ and a linear function. Because $\nabla g_\rho(x_{n+1} \mid x_n)^t(x - x_{n+1}) \geq 0$ for any x in S, the supporting hyperplane inequality implies that

$$g_\rho(x \mid x_n) - \frac{\rho}{2}\|Dx\|^2 \geq g_\rho(x_{n+1} \mid x_n) - \frac{\rho}{2}\|Dx_{n+1}\|^2 - \rho x_{n+1}^t D^t D(x - x_{n+1}),$$

or equivalently

$$g_\rho(x \mid x_n) \geq g_\rho(x_{n+1} \mid x_n) + \frac{\rho}{2}\|D(x - x_{n+1})\|^2. \tag{12}$$

Now note that the difference

$$d(x \mid y) = \frac{1}{2}\|x - \mathscr{P}(y)\|^2 - \frac{1}{2}\|x - \mathscr{P}(x)\|^2,$$

has gradient

$$\nabla d(x \mid y) = \mathscr{P}(x) - \mathscr{P}(y).$$

Because $\mathscr{P}(\boldsymbol{x})$ is non-expansive, the gradient $\nabla d(\boldsymbol{x} \mid \boldsymbol{y})$ is Lipschitz with constant 1. The tangency conditions $d(\boldsymbol{y} \mid \boldsymbol{y}) = 0$ and $\nabla d(\boldsymbol{y} \mid \boldsymbol{y}) = \boldsymbol{0}$ therefore yield

$$
\begin{aligned}
d(\boldsymbol{x} \mid \boldsymbol{y}) &\leq d(\boldsymbol{y} \mid \boldsymbol{y}) + \nabla d(\boldsymbol{y} \mid \boldsymbol{y})^t(\boldsymbol{x} - \boldsymbol{y}) + \frac{1}{2}\left\|\boldsymbol{x} - \boldsymbol{y}\right\|^2 \\
&= \frac{1}{2}\left\|\boldsymbol{x} - \boldsymbol{y}\right\|^2,
\end{aligned}
\tag{13}
$$

for all $\boldsymbol{x}$. At a minimum $\boldsymbol{z}_\rho$ of $h_\rho(\boldsymbol{x})$, combining inequalities (12) and (13) gives

$$
\begin{aligned}
&h_\rho(\boldsymbol{x}_{n+1}) + \frac{\rho}{2}\|\boldsymbol{D}(\boldsymbol{z}_\rho - \boldsymbol{x}_{n+1})\|^2 \\
&\leq g_\rho(\boldsymbol{x}_{n+1} \mid \boldsymbol{x}_n) + \frac{\rho}{2}\|\boldsymbol{D}(\boldsymbol{z}_\rho - \boldsymbol{x}_{n+1})\|^2 \\
&\leq g_\rho(\boldsymbol{z}_\rho \mid \boldsymbol{x}_n) \\
&= h_\rho(\boldsymbol{z}_\rho) - \frac{\rho}{2}\|\boldsymbol{D}\boldsymbol{z}_\rho - \mathscr{P}(\boldsymbol{D}\boldsymbol{z}_\rho)\|^2 + \frac{\rho}{2}\|\boldsymbol{D}\boldsymbol{z}_\rho - \mathscr{P}(\boldsymbol{D}\boldsymbol{x}_n)\|^2 \\
&= h_\rho(\boldsymbol{z}_\rho) + \rho d(\boldsymbol{D}\boldsymbol{z}_\rho \mid \boldsymbol{D}\boldsymbol{x}_n) \\
&\leq h_\rho(\boldsymbol{z}_\rho) + \frac{\rho}{2}\|\boldsymbol{D}\boldsymbol{z}_\rho - \boldsymbol{D}\boldsymbol{x}_n\|^2.
\end{aligned}
$$

Adding the result

$$
h_\rho(\boldsymbol{x}_{n+1}) - h_\rho(\boldsymbol{z}_\rho) \leq \frac{\rho}{2}\Big[\|\boldsymbol{D}(\boldsymbol{z}_\rho - \boldsymbol{x}_n)\|^2 - \|\boldsymbol{D}(\boldsymbol{z}_\rho - \boldsymbol{x}_{n+1})\|^2\Big],
$$

over n and invoking the descent property $h_\rho(\boldsymbol{x}_{n+1}) \quad h_\rho(\boldsymbol{x}_n)$, telescoping produces the desired error bound

$$
\begin{aligned}
h_\rho(\boldsymbol{x}_{n+1}) - h_\rho(\boldsymbol{z}_\rho) &\leq \frac{\rho}{2(n+1)}\Big[\|\boldsymbol{D}(\boldsymbol{z}_\rho - \boldsymbol{x}_0)\|^2 - \|\boldsymbol{D}(\boldsymbol{z}_\rho - \boldsymbol{x}_{n+1})\|^2\Big] \\
&\leq \frac{\rho}{2(n+1)}\|\boldsymbol{D}(\boldsymbol{z}_\rho - \boldsymbol{x}_0)\|^2.
\end{aligned}
$$

This is precisely the asserted bound. ∎

7.4 Proposition 3.4

Proof The existence and uniqueness of $\boldsymbol{z}_\rho$ are obvious. The remainder of the proof hinges on the facts that $h_\rho(\boldsymbol{x})$ is μ-strongly convex and the surrogate $g_\rho(\boldsymbol{x} \mid \boldsymbol{w})$ is $L + \rho\|\boldsymbol{D}\|^2$-smooth for all $\boldsymbol{w}$. The latter assertion follows from

$$
\nabla g_\rho(\boldsymbol{x} \mid \boldsymbol{w}) - \nabla g_\rho(\boldsymbol{y} \mid \boldsymbol{w}) = \nabla f(\boldsymbol{x}) - \nabla f(\boldsymbol{y}) + \rho\boldsymbol{D}^t\boldsymbol{D}(\boldsymbol{x} - \boldsymbol{y}).
$$

These facts together with $\nabla g_\rho(\boldsymbol{z}_\rho \mid \boldsymbol{z}_\rho) = \nabla h_\rho(\boldsymbol{z}_\rho) = \boldsymbol{0}$ imply

$$h_\rho(\boldsymbol{x}) - h_\rho(\boldsymbol{z}_\rho) \le g_\rho(\boldsymbol{x} \mid \boldsymbol{z}_\rho) - g_\rho(\boldsymbol{z}_\rho \mid \boldsymbol{z}_\rho)$$

$$\le \nabla g_\rho(\boldsymbol{z}_\rho \mid \boldsymbol{z}_\rho)^t(\boldsymbol{x} - \boldsymbol{z}_\rho) + \frac{L + \rho\|\boldsymbol{D}\|^2}{2}\|\boldsymbol{x} - \boldsymbol{z}_\rho\|^2 \tag{14}$$

$$= \frac{L + \rho\|\boldsymbol{D}\|^2}{2}\|\boldsymbol{x} - \boldsymbol{z}_\rho\|^2.$$

The strong convexity condition

$$0 \ge h_\rho(\boldsymbol{z}_\rho) - h_\rho(\boldsymbol{x}) \ge \nabla h_\rho(\boldsymbol{x})^t(\boldsymbol{z}_\rho - \boldsymbol{x}) + \frac{\mu}{2}\|\boldsymbol{z}_\rho - \boldsymbol{x}\|^2$$

entails

$$\|\nabla h_\rho(\boldsymbol{x})\| \cdot \|\boldsymbol{z}_\rho - \boldsymbol{x}\| \ge -\nabla h_\rho(\boldsymbol{x})^t(\boldsymbol{z}_\rho - \boldsymbol{x}) \ge \frac{\mu}{2}\|\boldsymbol{z}_\rho - \boldsymbol{x}\|^2.$$

It follows that $\|\nabla h_\rho(\boldsymbol{x})\| \ge \frac{\mu}{2}\|\boldsymbol{x} - \boldsymbol{z}_\rho\|$. This last inequality and inequality (14) produce the Polyak-Łojasiewicz bound

$$\frac{1}{2}\|\nabla h_\rho(\boldsymbol{x})\|^2 \ge \frac{\mu^2}{2\left(L + \rho\|\boldsymbol{D}\|^2\right)}[h_\rho(\boldsymbol{x}) - h_\rho(\boldsymbol{z}_\rho)].$$

Taking $c = L + \rho\|\boldsymbol{D}\|^2$ and

$$\boldsymbol{x} = \boldsymbol{x}_n - c^{-1}\nabla g_\rho(\boldsymbol{x}_n \mid \boldsymbol{x}_n) = \boldsymbol{x}_n - c^{-1}\nabla h_\rho(\boldsymbol{x}_n),$$

the Polyak-Łojasiewicz bound gives

$$h_\rho(\boldsymbol{x}_{n+1}) - h_\rho(\boldsymbol{x}_n) \le g_\rho(\boldsymbol{x}_{n+1} \mid \boldsymbol{x}_n) - g_\rho(\boldsymbol{x}_n \mid \boldsymbol{x}_n)$$
$$\le g_\rho(\boldsymbol{x} \mid \boldsymbol{x}_n) - g_\rho(\boldsymbol{x}_n \mid \boldsymbol{x}_n)$$
$$\le -c^{-1}\nabla g_\rho(\boldsymbol{x}_n \mid \boldsymbol{x}_n)^t\nabla h_\rho(\boldsymbol{x}_n) + \frac{c}{2}\|c^{-1}\nabla h_\rho(\boldsymbol{x}_n)\|^2$$
$$= -\frac{1}{2c}\|\nabla h_\rho(\boldsymbol{x}_n)\|^2$$
$$\le -\frac{\mu^2}{2c^2}[h_\rho(\boldsymbol{x}_n) - h_\rho(\boldsymbol{z}_\rho)].$$

Rearranging this inequality yields

$$h_\rho(\boldsymbol{x}_{n+1}) - h_\rho(\boldsymbol{z}_\rho) \le \left[1 - \frac{\mu^2}{2c^2}\right][h_\rho(\boldsymbol{x}_n) - h_\rho(\boldsymbol{z}_\rho))],$$

which can be iterated to give the stated bound. ∎

7.5   Proposition 4.2

Proof To validate the subanalytic premise of Proposition 4.1, first note that semialgebraic functions and sets are automatically subanalytic. The penalized loss

$$h_\rho(\boldsymbol{x}) = f(\boldsymbol{x}) + \frac{\rho}{2}\mathrm{dist}(\boldsymbol{x}, S)^2,$$

is semialgebraic by the sum rule. Under the assumption stated in Proposition 3.2, $g_\rho(\boldsymbol{x} \mid \boldsymbol{x}_n)$ is strongly convex and coercive for ρ sufficiently large. Continuity of $g_\rho(\boldsymbol{x} \mid \boldsymbol{x}_n)$ is a consequence of the continuity of $f(\boldsymbol{x})$. The Lipschitz condition follows from the fact that the sum of two Lipschitz functions is Lipschitz. Under these conditions and regardless of which projected point $P_S(\boldsymbol{x})$ is chosen, the MM iterates are guaranteed to converge to a stationary point. ∎

7.6   Proposition 4.3

Proof The first claim is true owing to the inclusion-exclusion formula

$$f_{(k)}(\boldsymbol{x}) = \sum_{j=k}^{n} \sum_{|S|=j} (-1)^{j-k} \binom{j-1}{k-1} \max_{i \in S} f_i(\boldsymbol{x}),$$

and the previously stated closure properties. For $n = 3$ and $k = 2$ the inclusion-exclusion formula reads $f_{(2)} = \max\{f_1, f_2\} + \max\{f_1, f_3\} + \max\{f_2, f_3\} - 2\max\{f_1, f_2, f_3\}$. To prove the second claim, note that a sparsity set in $\mathbb{R}^p$ with at most k nontrivial coordinates can be expressed as the zero set $\{\boldsymbol{x}: y_{(p-k)} = 0\}$, where $y_i = |x_i|$. Thus, the sparsity set is semialgebraic. ∎

7.7   Proposition 4.4

Proof Proposition 4.2 proves that the proximal distance iterates $\boldsymbol{x}_n$ converge to $\boldsymbol{x}_\infty$. Suppose that $\boldsymbol{Dx}_\infty$ has k unambiguous largest components in magnitude. Then $\boldsymbol{Dx}_n$ shares this property for large n. It follows that all $\boldsymbol{p}_n = \mathscr{P}_{S_k}(\boldsymbol{Dx}_n)$ occur in the same k-dimensional subspace S for large n. Thus, we can replace the sparsity set S_k by the subspace S in minimization from some n onward. Convergence at a linear rate now follows from Proposition 3.4.

To prove that the set A of points $\boldsymbol{x}$ such that $\boldsymbol{Dx}$ has k ambiguous largest components in magnitude has measure 0, observe that it is contained in the set T where two or more coordinates tie. Suppose $\boldsymbol{x}$ satisfies the tie condition $\boldsymbol{d}_i^t \boldsymbol{x} = \boldsymbol{d}_j^t \boldsymbol{x}$ for two rows $\boldsymbol{d}_i^t$ and $\boldsymbol{d}_j^t$ of $\boldsymbol{D}$. If the rows of $\boldsymbol{D}$ are unique, then the equality $(\boldsymbol{d}_i - \boldsymbol{d}_j)^t \boldsymbol{x} = 0$ defines a hyperplane in $\boldsymbol{x}$ space and consequently has measure 0. Because there are a finite number of row pairs, T as a union has measure 0. ∎

Acknowledgements

Appendix A.: Relevant Theory for Nonconvex Analysis

Let us first review the notion of a Fréchet subdifferential (Kruger, 2003). If $h(x)$ is a function mapping $\mathbb{R}^p$ into $\mathbb{R} \cup \{+\infty\}$, then its Fréchet subdifferential at $x \in \operatorname{dom} f$ is defined as

$$\partial^F h(x) = \left\{ v : \liminf_{y \to x} \frac{h(y) - h(x) - v^t(y - x)}{\|y - x\|} \geq 0 \right\}.$$

The set $^F h(x)$ is closed, convex, and possibly empty. If $h(x)$ is convex, then $^F h(x)$ reduces to its convex subdifferential. If $h(x)$ is differentiable, then $^F h(x)$ reduces to its ordinary differential. At a local minimum x, Fermat's rule $\mathbf{0} \in {}^F h(x)$ holds. For a locally Lipschitz and directionally differentiable function, the Fréchet subdifferential becomes

$$\partial^F h(x) = \left\{ v : d_u h(x) \geq v^t u \text{ for all directions } u \right\}.$$

Here $d_u h(x)$ is the directional derivative of $h(x)$ at x in the direction u. This result makes it clear that at a critical point, all directional derivatives are flat or point uphill.

We will also need some notions from algebraic geometry (Bochnak et al., 2013). For simplicity we focus on the class of semialgebraic functions and the corresponding class of semialgebraic subsets of $\mathbb{R}^p$. The latter is the smallest class that:

a. contains all sets of the form $\{x : q(x) > 0\}$ for a polynomial $q(x)$ in p variables, and

b. is closed under the formation of finite unions, finite intersections, set complements, and Cartesian products.

A function $a : \mathbb{R}^p \mapsto \mathbb{R}^r$ is said to be semialgebraic if its graph is a semialgebraic set of $\mathbb{R}^{p+r}$. The class of real-valued semialgebraic functions contains all polynomials $p(x)$ and all 0/1 indicators of algebraic sets. It is closed under the formation of sums and products and therefore constitutes a commutative ring with identity. The class is also closed under the formation of absolute values, reciprocals when $a(x) \quad 0$, nth roots when $a(x) \quad 0$, and maxima $\max\{a(x), b(x)\}$ and minima $\min\{a(x), b(x)\}$. Finally, the composition of two semialgebraic functions is semialgebraic.

Appendix B.: Convergence Properties of ADMM

To avail ourselves of the known results, we define three functions

$$H_\rho(x, y) = f(x) + \frac{\rho}{2} \operatorname{dist}(y, S)^2,$$

$$\mathcal{L}_\rho(x, y, \lambda) = H_\rho(x, y) + \lambda^t (Dx - y), \text{ and}$$
$$q(\lambda) = \min_{x, y} \mathcal{L}_\rho(x, y, \lambda),$$

the second and third being the Lagrangian and dual function. This notation leads to following result; see Beck (2017) for an accessible proof.

Proposition B.1 *Suppose that S is closed and convex and that the loss $f(x)$ is proper, closed, and convex with domain whose relative interior is nonempty. Also assume the dual function $q(\lambda)$ achieves its maximum value. If the objective $f(x) + \frac{\rho}{2}\|Dx\|^2 + a^t x$ achieves its minimum value for all $a \succeq 0$, then the ADMM running averages*

$$\bar{x}_n = \frac{1}{n}\sum_{k=1}^{n} x_k \text{ and } \bar{y}_n = \frac{1}{n}\sum_{k=1}^{n} y_k,$$

satisfy

$$|H_\rho(\bar{x}_n, \bar{y}_n) - h_\rho(x_\rho)| = O\left(\frac{\rho}{n}\right), \text{ and}$$

$$\|D\bar{x}_n - y_n\| = O\left(\frac{1}{k}\right).$$

Note that Proposition 3.2 furnishes a sufficient condition under which the functions $f(x) + \frac{\rho}{2}\|Dx\|^2 + a^t x$ achieve their minima. Linear convergence holds under stronger assumptions.

Proposition B.2 *Suppose that S is closed and convex, that the loss $f(x)$ is L-smooth and μ-strongly convex, and that the map determined by D is onto. Then the ADMM iterates converge at a linear rate.*

Giselsson and Boyd (2016) proved Proposition B.2 by operator methods. A range of convergence rates is specified there.

Appendix C.: Additional Details for Metric Projection Example

Given a $n \times n$ dissimilarity matrix $C = (c_{ij})$ with non-negative weights w_{ij}, our goal is to find a semi-metric $X = (x_{ij})$. We start by denoting trivec an operation that maps a symmetric matrix X to a vector x, $x = \text{trivec}(X)$ (Figure 3). Then we write the metric projection objective as

$$h_\rho(x) = \frac{1}{2}\left\|W^{1/2}(x - c)\right\|_2^2 + \frac{\rho}{2}\text{dist}(Tx, \mathbb{R}_+^{m_1})^2 + \frac{\rho}{2}\text{dist}(x, \mathbb{R}_+^{m_2})^2,$$

where $c = \text{trivec}(C)$. Here T encodes triangle inequalities and the m_i count the number of constraints of each type. The usual distance majorization furnishes a surrogate

$$g_\rho(x \mid x_n) = \frac{1}{2}\left\|W^{1/2}(x - c)\right\|_2^2 + \frac{\rho}{2}\left\|Tx - \mathscr{P}(Tx_n, \mathbb{R}_+^{m_1})\right\|_2^2 + \frac{\rho}{2}\left\|x - \mathscr{P}(x_n, \mathbb{R}_+^{m_2})\right\|_2^2$$
$$= \frac{1}{2}\left\|W^{1/2}(x - c)\right\|_2^2 + \frac{\rho}{2}\|Dx - \mathscr{P}(Dx_n)\|_2^2.$$

The notation $\mathscr{P}(\cdot, S)$ denotes projection onto a set S. The fusion matrix $D = [T\,;I]$ stacks the two operators; the joint projection operates in a block-wise fashion.
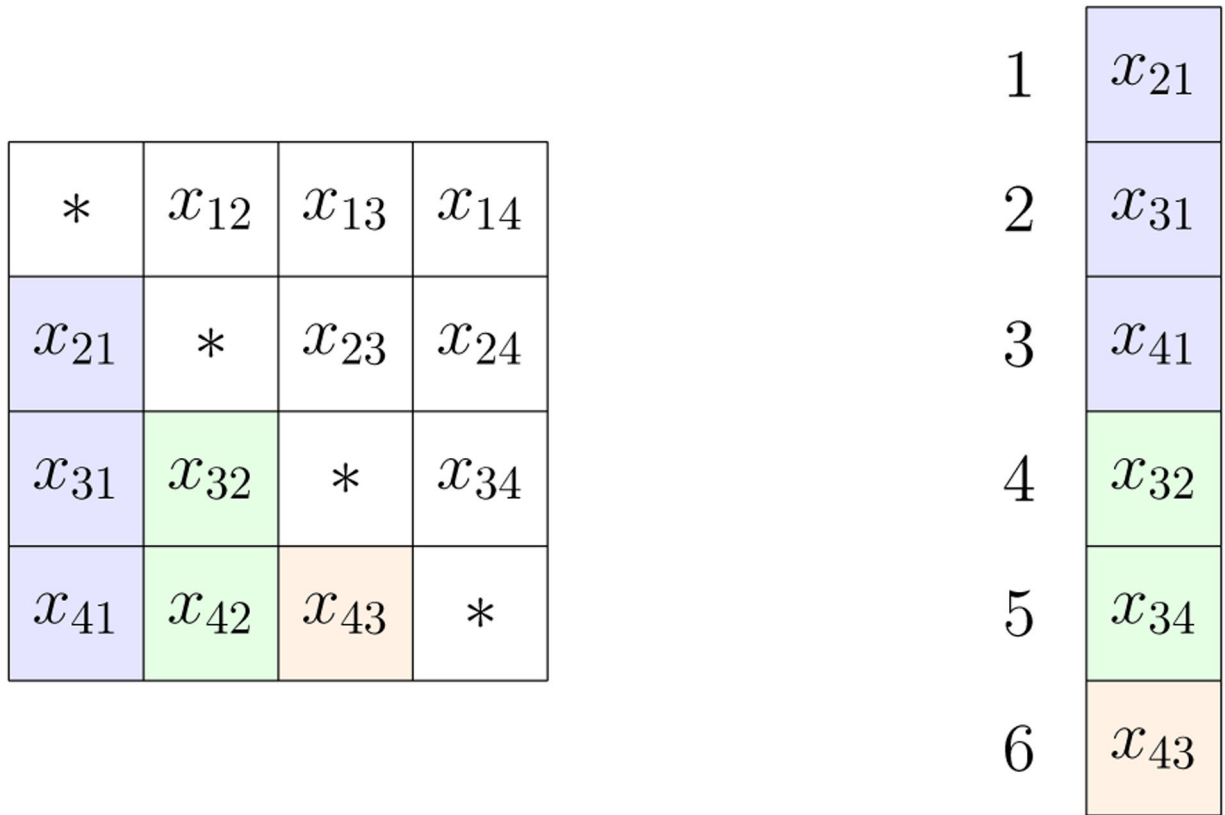

Figure 3:
Example of a symmetric matrix X and its minimal representation $x = \mathrm{trivec}(X)$.

C.1 MM

We rewrite the surrogate explicitly as a least squares problem minimizing $\|Ax - b_n\|_2^2$:

$$x_{n+1} = \underset{x}{\mathrm{argmin}}\,\frac{1}{2}\left\|\begin{bmatrix} W^{1/2} \\ \sqrt{\rho}D \end{bmatrix} x - \begin{bmatrix} c \\ \sqrt{\rho}\mathscr{P}(Dx_n) \end{bmatrix}\right\|_2^2,$$

where $c \equiv y$ from the main text. Updating the RHS b_n in the linear system reduces to evaluating the projection and copy operations. It is worth noting that triangle fixing algorithms that solve the metric nearness problem operate in the same fashion, except they work one triangle a time. That is, each iteration solves $\binom{n}{3}$ least squares problems compared to 1 in this formulation. A conjugate gradient type of algorithm solves the normal equations directly using $A^t A$, whereas LSQR type methods use only A and A^t.

C.2 Steepest Descent

The updates $x_{n+1} = x_n - t_n \nabla h_\rho(x_n)$ admit an exact solution for the line search parameter t_n. Recall the generic formula from the main text:

$$t_n = \frac{\|v_n\|^2}{v_n^t A v_n + \rho \|D v_n\|^2} .$$

Identifying v_n with $\nabla h_\rho(x_n)$ we have

$$\nabla h_\rho(x_n) = W(x_n - c) + \rho D^t [D x_n - \mathscr{P}(D x_n)]$$
$$= W(x_n - c) + \rho(I + T^t T)x_n - \rho[T^t \mathscr{P}(T x_n, \mathbb{R}_+^{m_1}) + \mathscr{P}(x_n, \mathbb{R}_+^{m_1})],$$
$$t_n = \frac{\|v_n\|^2}{\|W^{1/2} v_n\|^2 + \rho \|D v_n\|^2} .$$

C.3 ADMM

Taking y as the dual variable and λ as scaled multipliers, the updates for each ADMM block are

$$x_{n+1} = \underset{x}{\arg\min} \left\| \begin{bmatrix} W^{1/2} \\ \sqrt{\mu} D \end{bmatrix} x - \begin{bmatrix} c \\ \sqrt{\mu}(y_n - \lambda_n)) \end{bmatrix} \right\|_2^2 ,$$
$$y_{n+1} = \frac{\alpha}{1+\alpha} \mathscr{P}(z_n) + \frac{1}{1+\alpha} z_n; \quad z_n = D x_{n+1} + \lambda_n, \alpha = \rho/\mu .$$

Finally, the Multipliers follow the standard update.

C.4 Properties of the Triangle Inequality Matrix

These results have been documented before and are useful in designing fast subroutines for Dx and $D^t Dx$. Recall that m counts the number of nodes in the problem and $p = \binom{m}{2}$ is the number of parameters. In this notation $D = \binom{T}{I_p}$ and $D^t D = T^t T + I_p$.

Proposition C.1 *The matrix T has $3\binom{m}{3}$ rows and $\binom{m}{2}$ columns.*

Proof Interpret X as the adjacency matrix for a complete directed graph on m nodes without self-edges. When X is symmetric the number of free parameters is therefore $\binom{m}{2}$. An oriented 3-cycle is formed by fixing 3 nodes so there are $\binom{m}{3}$ such cycles. Now fix the orientation of the 3-cycles and note that each triangle encodes 3 metric constraints. The number of constraints is therefore $3\binom{m}{3}$. ∎

Proposition C.2 *Each column of* T *has* $3(m-2)$ *nonzero entries.*

Proof In view of the previous result, the entries T_{ij} encode whether edge j participates in constraint i. We proceed by induction on the number of nodes m. The base case $m = 3$ involves one triangle and is trivial. Note that a triangle encodes 3 inequalities.

Now consider a complete graph on m nodes and suppose the claim holds. Without loss of generality, consider the collection of 3-cycles oriented clockwise and fix an edge j. Adding a node to the graph yields $2m$ new edges, two for each of the existing m nodes. This action also creates one new triangle for each existing edge. Thus, edge j appears in $3(m-2)+3 = 3[(m+1)-2]$ triangle inequality constraints based on the induction hypothesis. ∎

Proposition C.3 *Each column of* T *has* $m - 2$ $+1s$ *and* $2(m-2)$ $-1s$.

Proof Interpret the inequality $x_{ij} \le x_{ik} + x_{kj}$ with $i > k > j$ as the ordered triple x_{ij}, x_{ik}, x_{kj}. The statement is equivalent to counting

$$a(N) = \text{number of times } x_{ij} \text{ appears in position 1, and,}$$

$$b(N) = \text{number of times } x_{ij} \text{ appears in position 2, or 3,}$$

where N denotes the number of constraints. In view of the previous proposition, it is enough to prove $a(N) = m - 2$. Note that $a(3) = 1$, meaning that x_{ij} appears in position 1 exactly once within a given triangle. Given that an edge (i, j) appears in $3(m-2)$ constraints, divide this quantity by the number of constraints per triangle to arrive at the stated result. ∎

Proposition C.4 *The matrix* T *has full column rank.*

Proof It is enough to show that $A = T^t T$ is full rank. The first two propositions imply

$$a_{ii} = \langle T_i, T_i \rangle = \sum (\pm 1)^2 = 3(m-2).$$

To compute the off-diagonal entries, fix a triangle and note that two edges i and j appear in all three of its constraints of the form $x_i \le x_j + x_k$. There are three possibilities for a given constraint c:

$$T_{c,i} T_{c,j} = \begin{cases} -1, & \text{if } i \text{ LHS, } j \text{ RHS or vice-versa} \\ 1, & \text{if } i \text{ and } j \text{ both appear on RHS} \\ 0, & \text{if one of } i \text{ or } j \text{ is missing}. \end{cases}$$

It follows that

$$a_{ij} = \langle T_i, T_j \rangle = \begin{cases} -1, & \text{if edges } i \text{ and } j \text{ overlap in constraints} \\ 0, & \text{otherwise.} \end{cases}$$

By Proposition C.2, an edge i appears in $3(m-2)$ constraints. Imposing the condition that edge j also appears reduces this number by $m-2$, the number of remaining nodes that can contribute edges in our accounting. The calculation

$$\sum_{j \neq i} |a_{ij}| = 2(m-2) < 3(m-2) = |a_{ii}|,$$

establishes that A is strictly diagonally dominant and hence full rank. ∎

Proposition C.5 *The matrix T^tT has at most 3 distinct eigenvalues of the form $m-2$, $2m-2$, and $3m-4$ with multiplicities 1, $m-1$, and $\frac{1}{2}m(m-3)$, respectively.*

Proof Let $M \in \{0,1\}^{\binom{m}{2} \times m}$ be the incidence matrix of a complete graph with m vertices. That is M has entry $m_{e,v} = 1$ if vertex v occurs in edge e and 0 otherwise. Each row of M has two entries equal to 1; each column of M has $m-1$ entries equal to 1. It is easy to see

$$T^tT = (3m-4)I_{\binom{m}{2}} - MM^t.$$

The Gram matrices M^tM and MM^t share the same positive eigenvalues. Since $M^tM = (m-2)I_m + m(\mathbf{1}_m/\sqrt{m})(\mathbf{1}_m/\sqrt{m})^t$ has eigenvalue $2m-2$ with multiplicity 1 and eigenvalue $m-2$ with multiplicity $m-1$, MM^t has eigenvalue $2m-2$ with multiplicity 1, eigenvalue $m-2$ with multiplicity $m-1$, and eigenvalue 0 with multiplicity $m(m-3)/2$. Therefore the eigenvalues of T^tT are $m-2$, $2m-2$, and $3m-4$ with multiplicities 1, $m-1$, and $m(m-3)/2$ respectively. ∎

In general, it is easy to check that the matrix $m \times m$ matrix $aI + b\mathbf{1}\mathbf{1}^t$ has the eigenvector $\mathbf{1}$ with eigenvalue $a + mb$ and $m-1$ orthogonal eigenvectors

$$u_i = \frac{1}{i-1}\sum_{j=1}^{i-1} e_j - e_i, \quad i = 2, \ldots, m$$

with eigenvalue a. Note that each u_i is perpendicular to $\mathbf{1}$. None of these eigenvectors is normalized to have length 1. Although the eigenvectors u_i are certainly convenient, they are not unique.

To recover the eigenvectors of T^tT, and hence those D^tD, we can leverage the eigenvectors of M^tM, which we know. The following generic observations are pertinent. If a matrix A has full SVD USV^t, then its transpose has full SVD $A^t = VSU^t$. As mentioned AA^t and A^tA share the same nontrivial eigenvalues. These can be recovered as the nontrivial diagonal entries of S^2. Suppose we know the eigenvectors U of $AA^t = US^2U^t$. Since $A^tU = VS$, then presumably we can recover some of the eigenvectors V as A^tUS^+, where S^+ is the diagonal pseudo-inverse of S.

C.5 Fast Subroutines for Solving Linear Systems

Using the Woodbury formula, the inverse of $T^t T$ can be expressed as

$$\left(T^t T\right)^{-1}$$

$$= \left[(3m-4)I_{\binom{m}{2}} - MM^t\right]^{-1}$$

$$= (3m-4)^{-1}I_{\binom{m}{2}} - (3m-4)^{-2}M\left[-I_m + (3m-4)^{-1}M^t M\right]^{-1}M^t$$

$$= (3m-4)^{-1}I_{\binom{m}{2}} - (3m-4)^{-1}M\left[-(2m-2)I_m + 1_m 1_m^t\right]^{-1}M^t$$

$$= (3m-4)^{-1}I_{\binom{m}{2}} - (3m-4)^{-1}M\left[-(2m-2)^{-1}I_m - (2m-2)^{-1}(m-2)^{-1}1_m 1_m^t\right]M^t$$

$$= \frac{1}{3m-4}I_{\binom{m}{2}} + \frac{2}{(3m-4)(m-1)(m-2)}1_{\binom{m}{2}}1_{\binom{m}{2}}^t + \frac{1}{2(3m-4)(m-1)}MM^t.$$

Solving linear system $T^t T$ invokes two matrix vector multiplications involving the incidence matrix M. Mv corresponds to taking pairwise sums of the components of a vector v of length m. $M^t w$ corresponds to taking a combination of column and row sums of a lower triangular matrix with the lower triangular part populated by the components of a vector w with length $\binom{m}{2}$. Both operations cost $O(m^2)$ flops. This result can be extended to the full fusion matrix $D^t D$ that incorporates non-negativity constraints and, more importantly, to the linear system $I + \rho D^t D$:

$$\left[I_{\binom{m}{2}} + \rho D^t D\right]^{-1} = \left[I_{\binom{m}{2}} + \rho\left(T^t T + I_{\binom{m}{2}}\right)\right]^{-1} \qquad a = [3(m-1)\rho + 1]^{-1}$$

$$= a\,I_{\binom{m}{2}} + ab\rho MM^t + 4abc\rho^2 1_{\binom{m}{2}}1_{\binom{m}{2}}^t; \quad b = [(2m-1)\rho + 1]^{-1}$$

$$c = [(m-1)\rho + 1]^{-1}.$$

Appendix D.: Additional Details for Convex Regression Example

We start by formulating the proximal distance version of the problem:

$$h_\rho(v) = \frac{1}{2}\left\|Mv - y\right\|_2^2 + \frac{\rho}{2}\text{dist}(Dv, \mathbb{R}_-^m)^2,$$

where $v = [\theta, \text{vec}(\Xi)]$ stacks each optimization variable into a vector of length $n(1 + d)$. This maneuver introduces matrices

$$M = [I_{n \times n}\ 0_{n \times nd}], \quad D = [A\ B],$$

where $[A\theta]_k = \theta_j - \theta_i$ and $[B\,\text{vec}(\Xi)]_k = \langle x_i - x_j, \xi_j\rangle$ according to the ordering $i > j$.

D.1 MM

We rewrite the surrogate explicitly a least squares problem minimizing $\left\| \widetilde{M} \boldsymbol{v} - \widetilde{\boldsymbol{b}}_n \right\|_2^2$:

$$\boldsymbol{v}_{n+1} = \underset{\boldsymbol{v}}{\operatorname{argmin}} \frac{1}{2} \left\| \begin{bmatrix} \boldsymbol{M} \\ \sqrt{\rho}\boldsymbol{D} \end{bmatrix} \boldsymbol{v} - \begin{bmatrix} \boldsymbol{b} \\ \sqrt{\rho}\mathscr{P}(\boldsymbol{D}\boldsymbol{v}_n) \end{bmatrix} \right\|_2^2,$$

where $\boldsymbol{b} \equiv \boldsymbol{y}$ to avoid clashing with notation in ADMM below. In this case it seems better to store $\boldsymbol{D}$ explicitly in order to avoid computing $\boldsymbol{x}_i - \boldsymbol{x}_j$ each time one applies $\boldsymbol{D}$, $\boldsymbol{D}^t$, or $\boldsymbol{D}^t\boldsymbol{D}$.

D.2 Steepest Descent

The updates $\boldsymbol{v}_{n+1} = \boldsymbol{v}_n - t_n \nabla h_\rho(\boldsymbol{v}_n)$ admit an exact solution for the line search parameter t_n. Taking $\boldsymbol{q}_n = \nabla h_\rho(\boldsymbol{v}_n)$ as the gradient we have

$$\boldsymbol{q}_n = \boldsymbol{A}^t \boldsymbol{A}(\boldsymbol{v}_n - \boldsymbol{b}) + \rho \boldsymbol{D}^t [\boldsymbol{D}\boldsymbol{v}_n - \mathscr{P}(\boldsymbol{D}\boldsymbol{v}_n)],$$

$$t_n = \frac{\|\boldsymbol{q}_n\|^2}{\|\boldsymbol{A}\boldsymbol{q}_n\|^2 + \rho\|\boldsymbol{D}\boldsymbol{q}_n\|^2}.$$

Note that $\boldsymbol{A}\boldsymbol{q}_n = \nabla_{\boldsymbol{\theta}} h_\rho(\boldsymbol{v}_n)$, the gradient with respect to function values $\boldsymbol{\theta}$.

D.3 ADMM

Take $\boldsymbol{y}$ as the dual variable and $\boldsymbol{\lambda}$ as scaled multipliers. Then the ADMM updates are

$$\boldsymbol{v}_{n+1} = \underset{\boldsymbol{v}}{\operatorname{argmin}} \frac{1}{2} \left\| \begin{bmatrix} \boldsymbol{A} \\ \sqrt{\mu}\boldsymbol{D} \end{bmatrix} \boldsymbol{v} - \begin{bmatrix} \boldsymbol{b} \\ \sqrt{\mu}(\boldsymbol{y}_n - \lambda_n) \end{bmatrix} \right\|_2^2,$$

$$\boldsymbol{y}_{n+1} = \frac{\alpha}{1+\alpha}\mathscr{P}(\boldsymbol{z}_n) + \frac{1}{1+\alpha}\boldsymbol{z}_n; \quad \boldsymbol{z}_n = \boldsymbol{D}\boldsymbol{v}_{n+1} + \lambda_n, \alpha = \rho/\mu,$$

and with the update for the multipliers being standard.

Appendix E.: Additional Details for Convex Clustering Example

We write $\boldsymbol{u} = \text{vec}(\boldsymbol{U})$ and $\boldsymbol{x} = \text{vec}(\boldsymbol{X})$, so the surrogate becomes

$$g_\rho(\boldsymbol{u} \mid \boldsymbol{u}_n) = \frac{1}{2}\left\| \boldsymbol{u} - \boldsymbol{x} \right\|_2^2 + \frac{\rho}{2}\left\| \boldsymbol{D}\boldsymbol{u} - \mathscr{P}_{S_k}(\boldsymbol{D}\boldsymbol{u}_n) \right\|^2.$$

E.1 MM

Rewrite the surrogate explicitly a least squares problem minimizing $\|\boldsymbol{A}\boldsymbol{u} - \boldsymbol{b}_n\|_2^2$:

$$u_{n+1} = \underset{u}{\operatorname{argmin}} \frac{1}{2} \left\| \begin{bmatrix} I \\ \sqrt{\rho}D \end{bmatrix} u - \begin{bmatrix} x \\ \sqrt{\rho}\mathscr{P}_{S_k}(Du_n) \end{bmatrix} \right\|_2^2.$$

E.2   Steepest Descent

The updates $u_{n+1} = u_n - t_n \nabla h_\rho(u_n)$ admit an exact solution for the line search parameter t_n. Taking $q_n = \nabla h_\rho(u_n)$ as the gradient we have

$$q_n = (u_n - x) + \rho D^t [Du_n - \mathscr{P}_{S_k}(Du_n)],$$

$$t_n = \frac{\|q_n\|^2}{\|q_n\|^2 + \rho\|Dq_n\|^2}.$$

Note that blocks in $[Du_n - \mathscr{P}_{S_k}(Du_n)]_\ell$ are equal to $\mathbf{0}$ whenever the projection of block $[Du_n]_\ell$ is non-zero.

E.3   ADMM

Take y as the dual variable and λ as scaled multipliers. Minimizing the u block involves solving a single linear system:

$$u_{n+1} = \underset{u}{\operatorname{argmin}} \frac{1}{2} \left\| \begin{bmatrix} I \\ \sqrt{\mu}D \end{bmatrix} u - \begin{bmatrix} x \\ \sqrt{\mu}(y_n - \lambda_n) \end{bmatrix} \right\|_2^2,$$

$$y_{n+1} = \frac{\alpha}{1+\alpha}\mathscr{P}_{S_k}(z_n) + \frac{1}{1+\alpha}z_n; \quad z_n = Du_{n+1} + \lambda_n, \alpha = \rho/\mu.$$

Multipliers follow the standard update.

E.4   Blockwise Sparse Projection

The projection $\mathscr{P}_{S_k}$ maps a matrix to a sparse representation with k non-zero columns (or blocks in the case of the vectorized version). In the context of clustering, imposing sparsity permits a maximum of k violations in consensus of centroid assignments, $u_i = u_j$. Letting $\ell \equiv ij = \|u_i - u_j\|$ denote pairwise distances and $M = \binom{m}{2}$ denote the number of unique pairwise distances, we define the projection along blocks $v_\ell = u_i - u_j$ for each pair as

$$\mathscr{P}_{S_k}(v_\ell) = \begin{cases} v_\ell, & \text{if } \Delta_\ell \in \{\Delta_{(M)}, \Delta_{(M-1)}, \dots \Delta_{(M-k+1)}\} \\ \mathbf{0}, & \text{otherwise.} \end{cases}$$

Here the notation $x_{(i)}$ represents the i-th element in an ascending list. Concretely, the magnitude of a difference v_ℓ must be within the top k distances. An alternative, helpful definition is based on the smallest distances

$$\mathscr{P}_{S_k}(\boldsymbol{v}_\ell) = \begin{cases} \mathbf{0}, & \text{if } \Delta_k \in \{\Delta_{(1)}, \Delta_{(2)}, \ldots \Delta_{(k)}\} \\ \boldsymbol{v}_\ell, & \text{otherwise} \end{cases}$$

Thus, it is enough to find a pivot $_{(M-k+1)}$ or $_{(k)}$ that splits the list into the top k elements. Because the hyperparameter k has a finite range in $\left\{0, 1, 2, \ldots, \binom{m}{2}\right\}$ one can exploit symmetry to reduce the best/average computational complexity in a search procedure. We implement this projection using a partial sorting algorithm based on quicksort, and note that it is set-valued in general.

Appendix F.: Additional Details for Image Denoising Example

Here we restate the total variation denoising problem to take advantage of proximal operators in the proximal distance framework. We minimize the penalized objective

$$h_\rho(\boldsymbol{U}) = \frac{1}{2}\left\|\boldsymbol{u} - \boldsymbol{w}\right\|_F^2 + \frac{\rho}{2}\text{dist}(\boldsymbol{Du}, S_\gamma)^2,$$

where $\boldsymbol{w} = \text{vec}(\boldsymbol{W})$ is a noisy input image and S_γ is the ℓ_1 ball with radius γ. Thus, γ may be interpreted as the target total variation of the reconstructed image. Distance majorization yields the surrogate

$$g_\rho(\boldsymbol{u} \mid \boldsymbol{u}_n) = \frac{1}{2}\left\|\boldsymbol{u} - \boldsymbol{w}\right\|_2^2 + \frac{\rho}{2}\left\|\boldsymbol{Du} - \mathscr{P}_\gamma(\boldsymbol{Du}_n)\right\|^2.$$

Here $\mathscr{P}_\gamma(\boldsymbol{Du})$ enforces sparsity in all derivatives through projection onto the ℓ_1 ball with radius γ. Because $\boldsymbol{D}$ is ill-conditioned, we append an additional row with zeros everywhere except the last entry; that is, $\boldsymbol{D} = [\boldsymbol{D}_n, \boldsymbol{D}_p, \boldsymbol{e}_p]$ with $\boldsymbol{u} \in \mathbb{R}^p$. In this case, the sparse projection applies to all but the last component of $\boldsymbol{Du}$.

F.1   MM

Rewrite the surrogate explicitly as a least squares problem:

$$\boldsymbol{u}_{n+1} = \underset{\boldsymbol{x}}{\text{argmin}} \frac{1}{2}\left\|\begin{bmatrix}\boldsymbol{I} \\ \sqrt{\rho}\boldsymbol{D}\end{bmatrix}\boldsymbol{u} - \begin{bmatrix}\boldsymbol{w} \\ \sqrt{\rho}\mathscr{P}_\gamma(\boldsymbol{Du}_n)\end{bmatrix}\right\|_2^2.$$

F.2   Steepest Descent

The updates $\boldsymbol{u}_{n+1} = \boldsymbol{u}_n - t_n\nabla h_\rho(\boldsymbol{u}_n)$ admit an exact solution for the line search parameter t_n. Taking $\boldsymbol{q}_n = \nabla h_\rho(\boldsymbol{u}_n)$ as the gradient we have

$$\boldsymbol{q}_n = (\boldsymbol{u}_n - \boldsymbol{w}) + \rho\boldsymbol{D}^t[\boldsymbol{Du}_n - \mathscr{P}_\gamma(\boldsymbol{Du}_n)],$$

$$t_n = \frac{\|\boldsymbol{q}_n\|^2}{\|\boldsymbol{q}_n\|^2 + \rho\|\boldsymbol{Dq}_n\|^2}.$$

F.3 ADMM

We denote by y the dual variable and λ the scaled multipliers. Minimizing the u block involves solving a single linear system:

$$u_{n+1} = \underset{x}{\operatorname{argmin}} \frac{1}{2} \left\| \begin{bmatrix} I \\ \sqrt{\mu} D \end{bmatrix} u - \begin{bmatrix} w \\ \sqrt{\mu}(y_n - \lambda_n) \end{bmatrix} \right\|_2^2,$$

$$y_{n+1} = \frac{\alpha}{1+\alpha} \mathscr{P}_{\gamma}(z_n) + \frac{1}{1+\alpha} z_n; \quad z_n = D u_{n+1} + \lambda_n, \alpha = \rho/\mu.$$

Multipliers follow the standard update.

Appendix G.: Additional Details for Condition Number Example

Given a matrix $M = U\Sigma V^{-1}$ with singular values $\sigma_1 \quad \sigma_2 \quad \dots \quad \sigma_p$, we seek a new matrix $N = UXV^{-1}$ such that $\operatorname{cond}(B) = x_1/x_p \quad c$. We minimize the penalized objective

$$h_{\rho}(x) = \frac{1}{2} \left\| x - \sigma \right\|^2 + \frac{\rho}{2} \operatorname{dist}\left(Dx, \mathbb{R}_-^{p^2} \right)^2,$$

as suggested by the Von Neumann-Fan inequality. The fusion matrix $D = C + S$ encodes the constraints $x_i - cx_j \quad 0$. Distance majorization yields the surrogate

$$g_{\rho}(x \mid x_n) = \frac{1}{2} \left\| x - w \right\|_2^2 + \frac{\rho}{2} \left\| Dx - \mathscr{P}_-(Dx_n) \right\|^2.$$

To be specific, the matrix $C = - c\mathbf{1}_p \otimes I_p$ scales the $p \times p$ identity matrix by $-c$ and stacks it p times. Similarly, the matrix $S = I_p \otimes \mathbf{1}_p$ stacks p matrices of dimension $p \times p$. Each of these stacked matrices has $(p-1)$ $\mathbf{0}_p$ columns and one shifted $\mathbf{1}_p$ column. For example, for $p = 2$

$$S = \begin{bmatrix} \begin{pmatrix} 1 & 0 \\ 1 & 0 \end{pmatrix} \\ \begin{pmatrix} 0 & 1 \\ 0 & 1 \end{pmatrix} \end{bmatrix}.$$

G.1 MM

Rewrite the surrogate explicitly a least squares problem minimizing $\|Ax - b_n\|_2^2$:

$$x_{n+1} = \underset{x}{\operatorname{argmin}} \frac{1}{2} \left\| \begin{bmatrix} I \\ \sqrt{\rho} D \end{bmatrix} x - \begin{bmatrix} \sigma \\ \sqrt{\rho} \mathscr{P}(Dx_n) \end{bmatrix} \right\|_2^2.$$

Applying the matrix inverse from before yields an explicit formula (with a and b defined as before):

$$x_{n+1} = \frac{1}{a}\left[z_n - \frac{1^t z_n}{p - (a/b)}1\right]; \quad z_n = \sigma + \rho D^t \mathcal{P}(Dx_n), a = 1 + \rho p(c^2 + 1), b = 2\rho c.$$

G.2 Steepest Descent

The updates $x_{n+1} = x_n - t_n \nabla h_\rho(x_n)$ admit an exact solution for the line search parameter t_n. Taking $q_n = \nabla h_\rho(x_n)$ as the gradient we have

$$q_n = (x_n - u) + \rho D^t[Dx_n - \mathcal{P}_v(Dx_n)],$$

$$t_n = \frac{\|q_n\|^2}{\|q_n\|^2 + \rho\|Dq_n\|^2}.$$

G.3 ADMM

Take y as the dual variable and λ as scaled multipliers. The formula for the MM algorithm applies in updating x_n, except we replace ρ with μ and (Dx_n) with $y_n - \lambda_n$:

$$x_{n+1} = \frac{1}{a}\left[z_n^1 - \frac{1^t z_n^1}{p - (a/b)}1\right]; \quad z_n^1 = \sigma + \mu D^t(y_n - \lambda_n), a = 1 + \mu p(c^2 + 1), b = 2\mu c$$

$$y_{n+1} = \frac{\alpha}{1 + \alpha}\mathcal{P}(z_n^2) + \frac{1}{1 + \alpha}z_n^2; \quad z_n^2 = Dx_{n+1} + \lambda_n, \alpha = \rho/\mu.$$

Multipliers follow the standard update.

G.4 Explicit Matrix Inverse

Both ADMM and MM reduce to solving a linear system. Fortunately, the Hessian for $h_\rho(x)$ reduces to a Householder-like matrix. First we note that it is trivial to multiply either C^t or S^t by a p^2-vector. The more interesting problem is calculating $(A_\rho^t A_\rho)^{-1}$, where $\nabla h_\rho^2 = A_\rho^t A_\rho = I_p + \rho D^t D$. The reader can check the identities

$$C^t C = c^2 p I_p \text{ and } C^t S = -c1_p 1_p^t,$$

$$S^t C = -c1_p 1_p^t \text{ and } S^t S = pI_p.$$

It follows that $A_\rho^t A_\rho = \left(1 + \rho p(c^2 + 1)\right)I_p - 2c\rho 1_p 1_p^t$. Applying the Sherman-Morrison formula to results in

$$\left[I_p + \rho D^t D\right]^{-1} = \left[aI_p - b1_p 1_p^t\right]^{-1}$$

$$= -b^{-1}\left[-(b/a)I_p - \frac{(b/a)^2 1_p 1_p^t}{1 - (a/b)1_p^t 1_p}\right]$$

$$= \frac{1}{a}\left[I_p - \frac{1_p 1_p^t}{p - a/b}\right],$$

where $a = 1 + \rho p(c^2 + 1)$ and $b = 2\rho c$. These simplifications make the exact proximal distance updates easy to compute.

Table 7:

Performance of MM on convex regression using CG and LSQR. Both inner and outer iterations are reported with the latter in parentheses.

features	samples	Time (s)		Loss ×10^6		Distance ×10^6		Iterations	
		CG	LSQR	CG	LSQR	CG	LSQR	CG	LSQR
20	50	**0.39** (0.00132)	0.68 (0.00607)	**0.991**	**0.991**	2.7	**2.65**	**104** (2)	**104** (2)
20	100	**1.46** (0.000164)	2.67 (0.00555)	0.995	**0.994**	**3.08**	3.16	**190** (2)	**190** (2)
20	200	**7.7** (0.166)	14.7 (0.00913)	**0.984**	**0.984**	**25.1**	**25.1**	**298** (2)	**298** (2)
20	400	**30.2** (0.0275)	63 (0.268)	**0.997**	**0.997**	9.21	**8.89**	**412** (2)	**412** (2)

Table 8:

Performance of ADMM on convex regression using CG and LSQR. Both inner and outer iterations are reported with the latter in parentheses.

features	samples	Time (s)		Loss ×10^6		Distance ×10^6		Iterations	
		CG	LSQR	CG	LSQR	CG	LSQR	CG	LSQR
20	50	**0.399** (0.00193)	0.677 (0.00507)	**0.91**	**0.91**	**6.96**	**6.96**	**98** (2)	**98** (2)
20	100	**1.58** (0.0329)	2.83 (0.00398)	**0.996**	**0.996**	**0**	**0**	**194** (2)	**194** (2)
20	200	**7.78** (0.0105)	15 (0.0386)	**0.961**	**0.961**	**43.8**	44	**296** (2)	**296** (2)
20	400	**30** (0.0157)	60.4 (0.0841)	**1.42**	1.45	**64.6**	67.8	377 (2)	**376** (2)

Appendix H.: Choice of Linear Solver

Updating parameters using MM or ADMM requires solving large-scale linear systems of the form $(I + cD^tD)x = b$. Here c_t is a scalar that depends on the outer iteration number t, in general, and the matrix on the LHS is square, symmetric, and often reasonably well-conditioned. Standard factorization methods like Cholesky and spectral decompositions cannot be applied without efficient update rules based on c_t. Instead, we turn to iterative methods, specifically conjugate gradients (CG) and LSQR, and use a linear map approach to adequately address sparsity, structure, and computational efficiency in matrix-vector multiplication. Tables 7 and 8 summarize performance metrics for MM and ADMM using both iterative linear solvers on instances of the convex regression problem. Times are averages taken over 3 replicates with standard deviations in parentheses, and iteration counts reflect the total number of inner iterations with outer counts in parentheses. We find no appreciable difference between CG and LSQR except on timing, and therefore favor CG in all our benchmarks.

Appendix I.: Software & Computing Environment

Code for our implementations and numerical experiments is available at https://github.com/alanderos91/ProximalDistanceAlgorithms.jl and is based on the Julia language (Bezanson et al., 2017). Additional packages used include Plots.jl (Breloff, 2021), GR.jl (Heinen et al., 2021), and (Udell et al., 2014). Numerical experiments were carried out on a Manjaro Linux 5.10.89–1 desktop environment using 8 cores on an Intel 10900KF at 4.9 GHz and 32 GB RAM.

References

Attouch Hédy, Bolte Jérôme, Redont Patrick, and Soubeyran Antoine. Proximal alternating minimization and projection methods for nonconvex problems: An approach based on the Kurdyka-Łojasiewicz inequality. Mathematics of Operations Research, 35(2):438–457, 2010.

Bauschke Heinz H and Combettes Patrick L. Convex Analysis and Monotone Operator Theory in Hilbert Spaces, 2nd edition, volume 408. Springer, 2017.

Beck Amir. First-Order Methods in Optimization, volume 25. SIAM, 2017.

Beltrami Edward J. An Algorithmic Approach to Nonlinear Analysis and Optimization. Academic Press, 1970.

Bendel Robert B. and Mickey M. Ray. Population correlation matrices for sampling experiments. Communications in Statistics - Simulation and Computation, 7(2):163–182, January 1978. ISSN 0361–0918. doi: 10.1080/03610917808812068. URL 10.1080/03610917808812068. Publisher: Taylor & Francis.

Bezanson Jeff, Edelman Alan, Karpinski Stefan, and Shah Viral B. Julia: A fresh approach to numerical computing. SIAM Review, 59(1):65–98, 2017. doi: 10.1137/141000671. URL https://epubs.siam.org/doi/10.1137/141000671.

Bochnak Jacek, Coste Michel, and Roy Marie-Françoise. Real Algebraic Geometry, volume 36. Springer Science & Business Media, 2013.

Bolte Jérôme, Daniilidis Aris, and Lewis Adrian. The łojasiewicz inequality for nonsmooth subanalytic functions with applications to subgradient dynamical systems. SIAM Journal on Optimization, 17(4):1205–1223, 2007.

Borwein Jonathan and Lewis Adrian S. Convex Analysis and Nonlinear Optimization: Theory and Examples. Springer Science & Business Media, 2010.

Boyd Stephen, Parikh Neal, Chu Eric, Peleato Borja, and Eckstein Jonathan. Distributed optimization and statistical learning via the alternating direction method of multipliers. Foundations and Trends in Machine Learning, 3(1):1–122, 2011.

Breloff Tom. Plots.jl, 2021. URL https://zenodo.org/record/4725317.

Brickell Justin, Dhillon Inderjit S., Sra Suvrit, and Tropp Joel A.. The Metric Nearness Problem. SIAM Journal on Matrix Analysis and Applications, 30(1):375–396, January 2008. ISSN 0895–4798, 1095–7162. doi: 10.1137/060653391. URL http://epubs.siam.org/doi/10.1137/060653391.

Chi Eric C. and Lange Kenneth. Splitting Methods for Convex Clustering. Journal of Computational and Graphical Statistics, 24(4):994–1013, October 2015. ISSN 1061–8600, 1537–2715. doi: 10.1080/10618600.2014.948181. URL http://www.tandfonline.com/doi/full/10.1080/10618600.2014.948181. [PubMed: 27087770]

Chi Eric C, Zhou Hua, and Lange Kenneth. Distance majorization and its applications. Mathematical Programming, 146(1–2):409–436, 2014. [PubMed: 25392563]

Chouzenoux Emilie, Idier Jérôme, and Moussaoui Saïd. A majorize–minimize strategy for subspace optimization applied to image restoration. IEEE Transactions on Image Processing, 20(6):1517–1528, 2010. [PubMed: 21193375]

Condat Laurent. Fast projection onto the simplex and the ℓ_1 ball. Mathematical Programming, 158(1–2):575–585, July 2016. ISSN 0025–5610, 1436–4646. doi: 10.1007/s10107-015-0946-6. URL http://link.springer.com/10.1007/s10107–015–0946–6.

Courant Richard. Variational Methods for the Solution of Problems of Equilibrium and Vibrations. Verlag Nicht Ermittelbar, 1943.

Cui Ying, Pang Jong-Shi, and Sen Bodhisattva. Composite difference-max programs for modern statistical estimation problems. SIAM Journal on Optimization, 28(4):3344–3374, 2018.

Davies Philip I. and Higham Nicholas J.. Numerically Stable Generation of Correlation Matrices and Their Factors. BIT Numerical Mathematics, 40(4):640–651, December 2000. ISSN 1572–9125. doi: 10.1023/A:1022384216930. URL 10.1023/A:1022384216930.

Debreu Gerard. Definite and semidefinite quadratic forms. Econometrica: Journal of the Econometric Society, pages 295–300, 1952.

Dua Dheeru and Graff Casey. UCI Machine Learning Repository, 2019. URL http://archive.ics.uci.edu/ml.

Chin-Lung Fong David and Saunders Michael. LSMR: An iterative algorithm for sparse least-squares problems. SIAM Journal on Scientific Computing, 33(5):2950–2971, 2011.

Giselsson Pontus and Boyd Stephen. Linear convergence and metric selection for Douglas-Rachford splitting and ADMM. IEEE Transactions on Automatic Control, 62(2):532–544, 2016.

Golub Gene H and Van Loan Charles F. Matrix Computations. Johns Hopkins University Press, 1996.

Heinen Josef, Deckers Malte, Kittisopikul Mark, Rhiem Florian, Rosario, Kojix2, Janssens Bart, Kaiser Daniel, Alobaid Faisal, Kelman Tony, Machakann, Tomaklutfu, Pfitzner Sebastian, Singhvi, Schnetter Erik, Zainchkovskyy Eugene, Van Der Plas Fons, Ekre Fredrik, Ling Jerry, Day Josh, TagBot Julia, Carlsson Kristoffer, Pastell Matti, Mehr, Sakse, Gezer Utkan, Goropikari Cnliao, Mtsch, and Raj. jheinen/gr.jl: release v0.63.0, 2021 URL https://zenodo.org/record/5798004.

Hong Mingyi, Luo Zhi-Quan, and Razaviyayn Meisam. Convergence analysis of alternating direction method of multipliers for a family of nonconvex problems. SIAM Journal on Optimization, 26(1):337–364, 2016.

Kang Yangyang, Zhang Zhihua, and Li Wu-Jun. On the global convergence of majorization minimization algorithms for nonconvex optimization problems. arXiv preprint arXiv:1504.07791, 2015.

Karimi Hamed, Nutini Julie, and Schmidt Mark. Linear convergence of gradient and proximal-gradient methods under the Polyak-Lojasiewicz condition. In Joint European Conference on Machine Learning and Knowledge Discovery in Databases, pages 795–811. Springer, 2016.

Keys Kevin L, Zhou Hua, and Lange Kenneth. Proximal distance algorithms: theory and practice. Journal of Machine Learning Research, 20(66):1–38, 2019.

Ya Kruger A. On Fréchet subdifferentials. Journal of Mathematical Sciences, 116(3):3325–3358, 2003.

Lange Kenneth. Numerical Analysis for Statisticians. Springer Science & Business Media, 2010.

Lange Kenneth. MM Optimization Algorithms, volume 147. SIAM, 2016.

Thi Hoai An Le, Dinh Tao Pham, et al. Convergence analysis of difference-of-convex algorithm with subanalytic data. Journal of Optimization Theory and Applications, 179(1): 103–126, 2018.

Luenberger David G, Ye Yinyu, et al. Linear and nonlinear programming, volume 2. Springer, 1984.

Mazumder Rahul, Choudhury Arkopal, Iyengar Garud, and Sen Bodhisattva. A Computational Framework for Multivariate Convex Regression and Its Variants. Journal of the American Statistical Association, 114(525):318–331, January 2019. ISSN 0162–1459, 1537–274X. doi: 10.1080/01621459.2017.1407771. URL https://www.tandfonline.com/doi/full/10.1080/01621459.2017.1407771.

Ashworth Nelder John and Wedderburn Robert WM. Generalized linear models. Journal of the Royal Statistical Society: Series A (General), 135(3):370–384, 1972.

Nesterov Yurii. Introductory Lectures on Convex Optimization: A Basic Course, volume 87. Springer Science & Business Media, 2013.

Paige Christopher C and Saunders Michael A. LSQR: An algorithm for sparse linear equations and sparse least squares. ACM Transactions on Mathematical Software (TOMS), 8 (1):43–71, 1982.

Parikh Neal. Proximal Algorithms. Foundations and Trends in Optimization, 1(3):127–239, 2014. ISSN 2167–3888, 2167–3918. doi: 10.1561/2400000003.

Rudin Leonid I., Osher Stanley, and Fatemi Emad. Nonlinear total variation based noise removal algorithms. Physica D: Nonlinear Phenomena, 60(1–4):259–268, November 1992. ISSN

01672789. doi: 10.1016/0167-2789(92)90242-F. URL https://linkinghub.elsevier.com/retrieve/pii/016727899290242F.

Seijo Emilio and Sen Bodhisattva. Nonparametric least squares estimation of a multivariate convex regression function. The Annals of Statistics, 39(3):1633–1657, June 2011. ISSN 0090–5364, 2168–8966. doi: 10.1214/10-AOS852. URL https://projecteuclid.org/euclid.aos/1311600278.

Sra Suvrit, Tropp Joel, and Dhillon Inderjit S. Triangle fixing algorithms for the metric nearness problem. In Advances in Neural Information Processing Systems, pages 361–368, 2005.

Tibshirani Robert, Saunders Michael, Rosset Saharon, Zhu Ji, and Knight Keith. Sparsity and smoothness via the fused lasso. Journal of the Royal Statistical Society: Series B, 67 (1):91–108, 2005.

Udell Madeleine, Mohan Karanveer, Zeng David, Hong Jenny, Diamond Steven, and Boyd Stephen. Convex optimization in Julia. In Proceedings of the 1st First Workshop for High Performance Technical Computing in Dynamic Languages, pages 18–28. IEEE Press, 2014.

Xuan Vinh Nguyen, Epps Julien, and Bailey James. Information Theoretic Measures for Clusterings Comparison: Variants, Properties, Normalization and Correction for Chance. Journal of Machine Learning Research, 11(95):2837–2854, 2010. ISSN 1533–7928. URL http://jmlr.org/papers/v11/vinh10a.html.

Xu Jason, Chi Eric, and Lange Kenneth. Generalized linear model regression under distance-to-set penalties. In Advances in Neural Information Processing Systems, pages 1385–1395, 2017.

Zhang Weijian and Higham Nicholas J.. Matrix Depot: an extensible test matrix collection for Julia. PeerJ Computer Science, 2:e58, April 2016. ISSN 2376–5992. doi: 10.7717/peerj-cs.58. URL https://peerj.com/articles/cs-58. Publisher: PeerJ Inc.
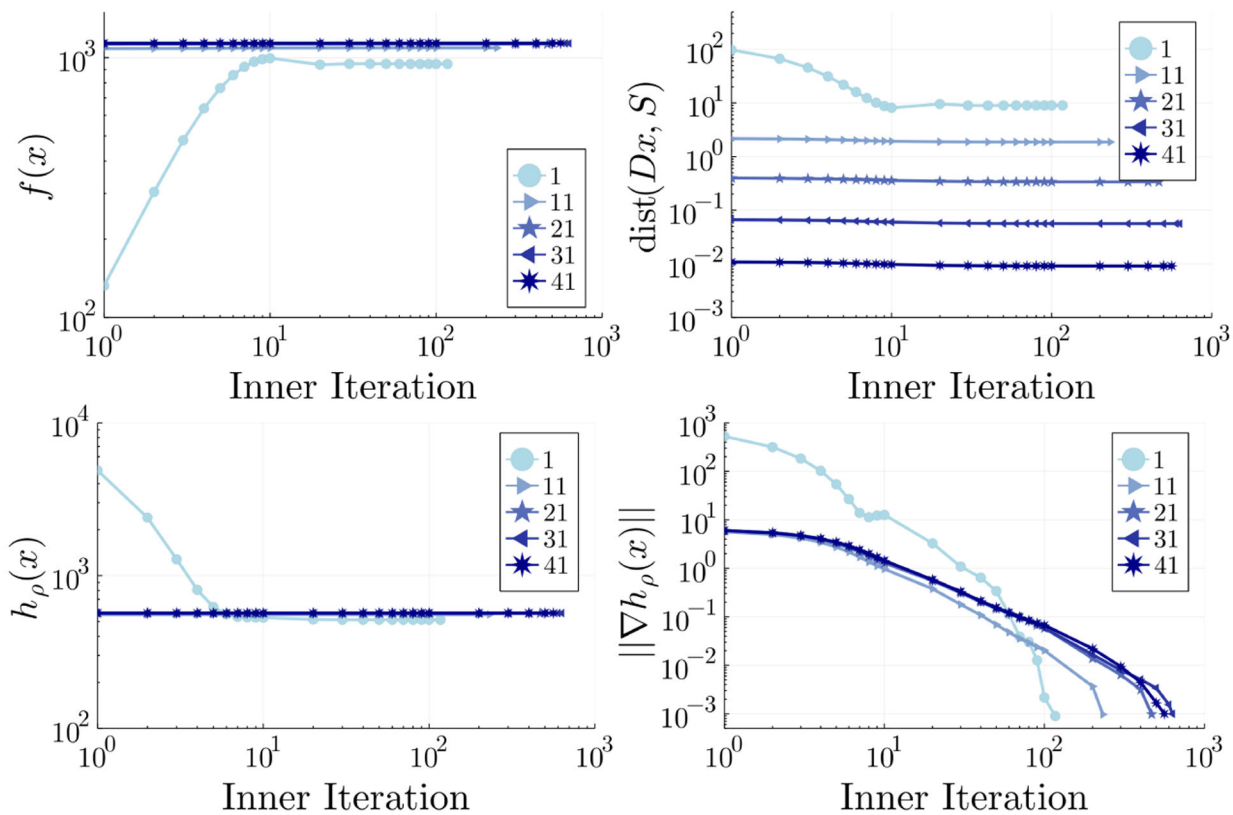
Figure 1:
Loss, distance, penalized objective, and gradient norm for SD on metric projection problem with 32 nodes, labeled by outer iteration.

Figure 2:
Sample images along the solution path of the search heuristic. Images are arranged from left to right as follows: reference, noisy input, and 90% reduction.

Table 1:

Summary of control parameters used in each example.

	δ_h	δ_d	δ_q	$\rho(t) = r^t$
Metric Projection	10^{-3}	10^{-2}	10^{-6}	$\min\{10^8, 1.2^{t-1}\}$
Convex Regression	10^{-3}	10^{-2}	10^{-6}	$\min\{10^8, 1.2^{t-1}\}$
Convex Clustering	10^{-2}	10^{-5}	10^{-6}	$\min\{10^8, 1.2^{t-1}\}$
Image Denoising	10^{-1}	10^{-1}	10^{-6}	$\min\{10^8, 1.5^{t-1}\}$
Condition Numbers	10^{-3}	10^{-2}	10^{-6}	$\min\{10^8, 1.2^{t-1}\}$

Table 2:

Metric projection. Times are averaged over 3 replicates with standard deviations in parentheses. Reported iteration counts reflect the total inner iterations taken with outer iterations in parentheses.

	Time (s)			Loss $\times 10^{-3}$			Distance $\times 10^3$			Iterations		
m	MM	SD	ADMM	MM	SD	ADMM	MM	SD	ADMM	MM	SD	ADMM
16	0.0429 (0:000709)	**0.0338** (0:000137)	0.107 (0.000858)	**0.237**	**0.237**	**0.237**	**9.24**	**9.24**	**9.24**	4980 (37)	**3920** (37)	7030 (37)
32	**1.24** (0.00309)	1:28 (0.00617)	2.6 (0.0149)	**1.14**	**1.14**	**1.14**	**9.13**	**9.13**	**9.13**	16000 (41)	**15400** (41)	17300 (41)
64	19.5 (0.00835)	**16.7** (0.0148)	43.9 (0.0616)	**4.69**	**4.69**	**4.69**	**8.7**	**8.7**	**8.7**	30100 (44)	24200 (44)	33700 (44)
128	171 (0.613)	**150** (0.28)	725 (0.772)	**18.4**	**18.4**	**18.4**	**9.44**	**9.44**	**9.44**	29900 (44)	**23700** (44)	51900 (44)
256	1670 (0.702)	**1570** (5.63)	9110 (75.4)	**75.3**	**75.3**	**75.3**	**8.68**	**8.68**	**8.68**	32500 (46)	**26700** (46)	76100 (46)

Table 3:

Convex regression. Times are reported as averages over 3 replicates.

d	m	Time (s) MM	Time (s) SD	Time (s) ADMM	Loss ×10³ MM	Loss ×10³ SD	Loss ×10³ ADMM	Distance ×10⁴ MM	Distance ×10⁴ SD	Distance ×10⁴ ADMM	MSE ×10² MM	MSE ×10² SD	MSE ×10² ADMM
1	50	0.015 (0.00477)	**0.0109** (0.00162)	0.0171 (0.0033)	**454**	**454**	**454**	**87.3**	87.4	87.5	**70.1**	**70.1**	**70.1**
	100	0.0382 (0.002)	**0.0311** (0.000301)	0.0578 (0.00179)	**510**	**510**	**510**	**94.3**	94.4	94.5	**75.3**	**75.3**	**75.3**
	200	0.138 (0.00662)	**0.0991** (0.00138)	0.205 (0.00213)	**471**	**471**	**471**	**92.2**	**92.2**	**92.2**	**71**	**71**	**71**
	400	0.565 (0.012)	**0.464** (0.000493)	0.862 (0.0184)	**501**	**501**	**501**	**96.9**	97	97	**79.1**	**79.1**	**79.1**
2	50	1.71 (0.00334)	**0.693** (0.00341)	16 (0.00511)	122	126	**118**	85.9	**85.3**	85.5	72.2	73.1	**70.8**
	100	11.8 (0.0189)	**3.33** (0.00704)	66.6 (0.0844)	**162**	163	**162**	99.4	**98**	98.4	95.4	95.8	**95.1**
	200	51.4 (0.0701)	**14.1** (0.00805)	230 (0.273)	**233**	234	**233**	98.1	**94.2**	96.8	123	123	**122**
	400	200 (1.06)	**50.3** (0.299)	917 (1.48)	239	239	**238**	94.3	**90.8**	91.9	**140**	**140**	**140**
10	50	0.19 (0.00281)	**0.00722** (9.99 × 10⁻⁵)	0.196 (0.00743)	0.000891	0.00109	**0.000838**	0.821	2.23	**0.488**	**8.59**	8.63	8.61
	100	0.854 (0.0002)	**0.0644** (0.000503)	0.873 (0.00191)	**0.000937**	0.00097	0.000943	0.154	**0**	0.11	10.3	**10.2**	10.3
	200	3.77 (0.00486)	**0.398** (0.00888)	3.89 (0.0132)	**0.000883**	0.00099	0.00101	0.281	**0**	0.292	9.64	**9.63**	9.65
	400	26.8 (0.405)	**3.17** (0.0183)	27.6 (0.0488)	**0.000992**	0.000997	0.000999	0.185	0.288	**0.176**	9.41	9.42	**9.39**
20	50	0.39 (0.00132)	**0.00791** (2.6 × 10⁻⁵)	0.399 (0.00193)	0.000991	0.00542	**0.00091**	**0.027**	7.06	0.0696	9.77	**9.34**	9.57
	100	1.46 (0.000164)	**0.0684** (0.000174)	1.58 (0.0329)	0.000995	**0.000965**	0.000996	0.0308	**0**	**0**	9.13	9.22	**9**
	200	7.7 (0.166)	**0.414** (0.00109)	7.78 (0.0105)	0.000984	0.00113	**0.000961**	**0.251**	1.74	0.438	**9.6**	9.61	**9.6**
	400	30.2 (0.0275)	**3.03** (0.00605)	30 (0.0157)	**0.000997**	0.00105	0.00142	**0.0921**	1.68	0.646	**10**	10.2	10.2

Table 4:

Convex clustering. Times reflect the total time spent generating candidate clusterings using Algorithm 3. Additional metrics correspond to the optimal clustering on the basis of maximal ARI. Time and clustering indices are averaged over 3 replicates with standard deviations reported in parentheses.

dataset	features	samples	classes	Time (s)		Loss		Distance ×10⁵		ARI		NMI	
				MM	SD	MM	SD	MM	SD	MM	SD	MM	SD
zoo	16	101	7	95.6 (0.268)	**77.1** (2.02)	**1600**	**1600**	**8.62**	**8.62**	0.841 (0)	**0.848** (0.0118)	0.853 (0)	**0.856** (0.00256)
iris	4	150	3	76.4 (0.129)	**62.8** (2)	**596**	**596**	**8.8**	**8.8**	**0.575** (0)	**0.575** (0)	**0.734** (0)	**0.734** (0)
gaussian300	2	300	3	190 (0.173)	**155** (0.177)	**598**	**598**	**8.98**	**8.98**	**1** (0)	**1** (0)	**1** (0)	**1** (0)
spiral500	2	500	2	715 (18.5)	**561** (0.501)	**998**	**998**	**8.98**	**8.98**	**0.133** (0)	**0.133** (0)	**0.366** (0)	**0.366** (0)

Table 5:

Image denoising. Times reflect the total time spent generating candidate images, averaged over 3 replicates, ultimately achieving 90% reduction in total variation

image	width	height	Time (s)		Loss		Distance $\times 10^3$		MSE $\times 10^5$		PSNR	
			MM	SD	MM	SD	MM	SD	MM	SD	MM	SD
cameraman	512	512	557 (5.01)	**181** (0.906)	**8090**	**8090**	**93.1**	**93.1**	**284**	**284**	**25.5**	**25.5**
peppers_gray	512	512	553 (6.28)	**183** (2.11)	**8020**	**8020**	92.6	**92.5**	**290**	**290**	**25.4**	**25.4**

Table 6:

Condition number experiments. Here $c(M)$ is the condition number of the input matrix, a is the decrease factor, and $c(X)$ is the condition number of the solution.

p	$c(M)$	a	Time (ms)			Loss $\times 10^3$			Distance $\times 10^5$			$c(X)$		
			MM	SD	ADMM	MM	SD	ADMM	MM	SD	ADMM	MM	SD	ADMM
10	119	2	0.576 (0.104)	**0.391** (0.00629)	0.582 (0.172)	0.549	**0.548**	0.549	41.1	41.1	**40.6**	**59.4**	**59.4**	**59.4**
		4	0.413 (0.00321)	**0.359** (0.00129)	0.543 (0.101)	**4.92**	4.92	4.92	236	**234**	236	**29.7**	**29.7**	**29.7**
		16	0. 439 (0.00319)	**0.424** (0.00427)	0.535 (0.00395)	**130**	130	130	**926**	929	927	**7.44**	**7.44**	**7.44**
		32	0.662 (0.00336)	**0.496** (0.00357)	0.824 (0.00459)	**821**	821	821	984	**983**	983	**3.72**	**3.72**	**3.72**
100	1920	2	**37.7** (2.89)	51.1 (3.77)	149 (0.972)	**0.00119**	0.00119	0.00119	0.208	0.208	**0.204**	**960**	960	960
		4	**17.1** (0.164)	26.5 (2.8)	66.4 (0.485)	**0.0107**	0.0107	0.0107	**0.845**	0.845	0.877	**480**	480	480
		16	**10.1** (0.301)	11. (0.0472)	35.3 (0.437)	**0.436**	0.436	0.436	**17.8**	17.9	17.9	**120**	120	120
		32	**26** (2.23)	33.4 (0.49)	74 (0.893)	**3.26**	3.26	3.26	**95.2**	95.2	95.2	**60**	60	60
1000	59400	2	**60.3** (0.83)	75.3 (1.35)	157 (3.98)	**1.15×10^{-6}**	**1.15×10^{-6}**	**1.15×10^{-6}**	**0**	**0**	**0**	29700	29700	**29600**
		4	**57.4** (3.37)	71.8 (0.882)	131 (8.44)	**1.04×10^{-5}**	**1.04×10^{-5}**	**1.04×10^{-5}**	**0**	**0**	**0**	14800	14800	14800
		16	**55.6** (3.3)	71.9 (2.05)	152 (38.8)	**0.000258**	0.000258	0.000261	**0**	**0**	**0**	3710	3710	**3690**
		32	**23200** (855)	30000 (337)	87400 (1340)	**0.0011**	**0.0011**	0.00111	0.11	0.11	**0**	1860	1860	**1850**