Basis Pursuit Denoise with Nonsmooth Constraints

Robert Baraldi[1], Rajiv Kumar[2], and Aleksandr Aravkin[1].
[1] Department of Applied Mathematics, University of Washington
[2] Formerly School of Earth and Atmospheric Sciences, Georgia Institute of Technology, USA; Currently DownUnder GeoSolutions, Perth, Australia

***Abstract*—Level-set optimization formulations with data-driven constraints minimize a regularization functional subject to matching observations to a given error level. These formulations are widely used, particularly for matrix completion and sparsity promotion in data interpolation and denoising. The misfit level is typically measured in the ℓ_2 norm, or other smooth metrics.**

In this paper, we present a new flexible algorithmic framework that targets *nonsmooth* level-set constraints, including ℓ_1, ℓ_∞, and even ℓ_0 norms. These constraints give greater flexibility for modeling deviations in observation and denoising, and have significant impact on the solution. Measuring error in the ℓ_1 and ℓ_0 norms makes the result more robust to large outliers, while matching many observations exactly.

We demonstrate the approach for basis pursuit denoise (BPDN) problems as well as for extensions of BPDN to matrix factorization, with applications to interpolation and denoising of 4D seismic data. The new methods are particularly promising for seismic applications, where the amplitude in the data varies significantly, and measurement noise in low-amplitude regions can wreak havoc for standard Gaussian error models.

***Index Terms*—Nonconvex nonsmooth optimization, level-set formulations, basis pursuit denoise, interpolation, seismic data.**

I. Introduction

Basis Pursuit Denoise (BPDN) seeks a sparse solution to an under-deterimined system of equations that have been corrupted by noise. The classic level-set formulation [41], [3] is given by

$$\min_x \|x\|_1 \quad \text{s.t.} \quad \|\mathcal{A}(x) - b\|_2 \leq \sigma \tag{1}$$

where $\mathcal{A} : \mathbb{R}^{m\times n} \to \mathbb{R}^d$ is a linear functional taking unknown parameters $x \in \mathbb{R}^{m\times n}$ to observations $b \in \mathbb{R}^d$. Problem (1) is also known as a Morozov formulation (in contrast to Ivanov or Tikhonov [32]). The functional $\mathcal{A}$ can include a transformation to another domain, including Wavelets, Fourier, or Curvelet coefficients [13], as well as compositions of these transforms with other linear operators such as restriction in interpolation problems. The parameter σ controls the error budget, and is based on an estimate of noise level in the data.

Problems with cardinality or sparsity constraints, such as BPDN and the closely related LASSO formulation, have applications to compressed sensing [33], [11], [18], image processing [30], sparse controller design [28], and machine learning [22], [20], as well as to more applied domains including MRI [29] and seismic inversion [38]. Indeed, the total variation norm, as well as other sparsity-inducing functions, has been used to denoise images since the 1980s [34], [16]. Seismic data is a key use case [4], [27], [15], and the primary application for the techniques developed in this paper. Data acquisition is prohibitively expensive in seismology and interpolation techniques are used to fill in data volumes by promoting parsimonious representations in the Fourier [35] or Curvelet [24] domains. Matricization of the data leads to low-rank interpolation schemes [4], [27], [15], [44]. The types of errors encountered in seismic inversion motivate the technology developed here. In our numerics section, we show how using the ℓ_0 norm for the data misfit constraints can help deal with the situation where uniformly large errors applied across a range of scales creates a variable SNR that stops more common losses from effectively recovering the signal.

Related Work. Theoretical recovery guarantees for classes of operators $\mathcal{A}$ are developed in [11] and [39]. While BPDN uses nonsmooth regularizers (including the ℓ_1 norm, nuclear norm, and elastic net), the inequality constraint is ubiquitously smooth, and often taken to be the ℓ_2 norm as in (1). Indeed, the ℓ_1 norm is particularly useful in that it is a convex proxy for sparsity [37] and much work has been done to remain in this convex regime, despite some disadvantages of the ℓ_1 norm as a sparsity-inducing metric [36]. In most applications of BPDN, the ℓ_2 norm in the context of Gaussian noise takes on the interpretation that σ in Problem (1) is the variance of the noise [17]. In almost all contexts, this noise is assumed to be Gaussian [21], [37], [36]. This stems from the very nature of denoising/image reconstruction being a very ill-posed problem predicated on *a-priori* information about the data itself. Prior work, including [42], [4], [15], [3], exploits the smoothness of the inequality constraint in developing algorithms for the problem class.

These smooth constraints work well when errors are Gaussian, but this assumption fails for seismic data (explored in [38]) and is often violated in many applications, from sparse controller design [28] to compressed sensing [18].

The use of cardinality constraints (ie the ℓ_0 norm) to enforce sparsity has been studied in depth in the case of convex cost functionals [7] or continuously differentiable cost functionals [8]. Nonsmooth data fidelity terms have been explored by Nikolova in [31], whose work underscores the utility of using these terms in modeling. We develop a broad class that captures any nonsmooth nonconvex regularizer and/or misfit, and gives a way to set an error-budget (σ in Equation 1). We also explore the matrix variant of the problem, which can be viewed as an extension of robust PCA [2], [12], [23]. Robust PCA is equivalent to minimizing a rank functional subject to a Huber data misfit [19], and hence misfit constrained versions of the problem use smooth misfit constraints, see e.g. [4]. Our formulation extends all these approaches to nonsmooth data misfit constraints, including cardinality constraints, just as in the vector case.

Contributions. The main contribution of this paper is to provide a fast, easily adaptable algorithm to solve non-smooth and nonconvex data constraints in general level-set formulations including BPDN, and illustrate the efficacy of the approach using large-scale interpolation and denoising problems. To do this, we extend the universal regularization framework of [46] to level-set formulations with nonsmooth/nonconvex constraints. We develop a convergence theory for the optimization approach, and illustrate the practical performance of the new formulations for data interpolation and denoising in both sparse recovery and low-rank matrix factorization.

Roadmap. The paper proceeds as follows. Section II develops the general relaxation framework and approach. Section III specifies this framework to the BPDN setting with nonsmooth, nonconvex constraints. In Section IV we apply the approach to sparse signal recovery problem and sparse Curvelet reconstruction. In Section V, we extend the approach to a low-rank interpolation framework, which embeds matrix factorization within the BPDN constraint. In Section VI we test the low-rank extension using synthetic examples and data extracted from a full 5D dataset simulated on complex SEG/EAGE overthrust model.

II. Nonsmooth, nonconvex level-set formulations.

We consider the following problem class:

$$\min_x \phi(\mathcal{C}(x)) \quad \text{s.t.} \quad \psi(\mathcal{A}(x) - b) \leq \sigma, \tag{2}$$

where ϕ and ψ may be nonsmooth, nonconvex, but have well-defined proximity and projection operators:

$$\begin{aligned} \text{prox}_{\eta\phi}(y) &= \arg\min_x \frac{1}{2\eta}\|x-y\|^2 + \phi(x) \\ \text{proj}_{\psi(\cdot)\leq\sigma} &= \arg\min_{\psi(x)\leq\sigma} \frac{1}{2\eta}\|x-y\|^2. \end{aligned} \tag{3}$$

Here, $\mathcal{C} : \mathbb{C}^{m\times n} \to \mathbb{R}^c$ is typically a linear operator that converts x to some transform domain, while $\mathcal{A} : \mathbb{C}^{m\times n} \to \mathbb{R}^d$ is a linear observation operator also acting on x. In the context of interpolation, $\mathcal{A}$ is often a restriction operator.

This setting significantly extends that of [3], who assume ψ and ϕ are convex, $\mathcal{C} = I$, and use the *value function*

$$v(\tau) = \min_x \psi(\mathcal{A}(x) - b) \quad \text{s.t.} \quad \phi(x) \leq \tau$$

to solve (2) using root-finding to solve $v(\tau) = \sigma$. Variational properties of v are fully only understood in the convex setting, and efficient evaluation of $v(\tau)$ requires ψ to be smooth, so that efficient first-order methods are applicable.

Here, we develop an approach to solve any problem of type (2), including problems with nonsmooth and nonconvex ψ, ϕ, using only matrix vector products with $\mathcal{A}, \mathcal{A}^T, \mathcal{C}, \mathcal{C}^T$ and simple nonlinear operators. In special cases, the approach can also use equation solves to gain significant speedup.

Algorithm 1 Prox-gradient for (4).

Input: x^0, w_1^0, w_2^0
Initialize: $k = 0$
while not converged **do**
  $x^{k+1} \leftarrow x^k - \beta\left(\frac{1}{\eta_1}\mathcal{C}^T(\mathcal{C}(x) - w_1) + \frac{1}{\eta_2}\mathcal{A}^T(\mathcal{A}(x) - w_2^k - b)\right)$
  $w_1^{k+1} \leftarrow \text{prox}_{\beta\phi}\left(w_1^k - \frac{\beta}{\eta_1}(w_1^k - \mathcal{C}(x^{k+1}))\right)$
  $w_2^{k+1} \leftarrow \text{proj}_{\sigma\mathbb{B}_\psi}\left(w_2^k - \frac{\beta}{\eta_2}(w_2^k - (\mathcal{A}(x^{k+1}) - b))\right)$
  $k \leftarrow k + 1$
end while
Output: w_1^k, w_2^k, x^k

The general approach uses the relaxation formulation proposed in [46], [45]. We use relaxation to split ϕ, ψ from the linear map $\mathcal{A}$ and transformation map $\mathcal{C}$, extending (2) to

$$\begin{aligned} \min_{x,w_1,w_2} \phi(w_1) + \frac{1}{2\eta_1}\|\mathcal{C}(x) - w_1\|^2 + \frac{1}{2\eta_2}\|w_2 - \mathcal{A}(x) + b\|_2^2 \\ \text{s.t.} \quad \psi(w_2) \leq \sigma. \end{aligned} \tag{4}$$

with $w_1 \in \mathbb{R}^c$ and $w_2 \in \mathbb{R}^d$. In contrast to [46], we use a continuation scheme to force $\eta_i \to 0$, in order to solve the original formulation (2). Thus the only external algorithmic parameter the scheme requires is σ, which controls the error budget for ψ.

There are two algorithms readily available to solve (4). The first is prox-gradient descent, detailed in Algorithm 1. We let $z = [x, w_1, w_2]^T$, and define

$$\Phi(z) = \phi(w_1) + \delta_{\psi(\cdot)\leq\sigma}(w_2),$$

where the *indicator function* $\delta_{\psi(\cdot)\leq\sigma}$ takes the value 0 if $\psi(w_2) \leq \sigma$, and infinity otherwise. Problem (4) can now be written as

$$\min_z \underbrace{\frac{1}{2}\left\|\begin{bmatrix} \frac{1}{\sqrt{\eta_1}}\mathcal{C} & -\frac{1}{\sqrt{\eta_1}}I & 0 \\ \frac{1}{\sqrt{\eta_2}}\mathcal{A} & 0 & -\frac{1}{\sqrt{\eta_2}}I \end{bmatrix} z - \begin{bmatrix} 0 \\ b \end{bmatrix}\right\|^2}_{f(z)} + \Phi(z). \tag{5}$$

Applying the prox-gradient descent iteration with step-size β

$$z^{k+1} = \text{prox}_{\beta\Phi}(z^k - \beta\nabla f(z^k)) \tag{6}$$

gives the coordinate updates in Algorithm 1.

Prox-gradient has been analyzed in the general nonconvex setting by [5], [6]. Since Problem 4 is semi-algebraic, we have from [5] that Algorithm 1 (and its subsequent reductions, Algorithms 2-4) converge to a critical point. However, we can exploit the nature of our relaxation to make simple improvements on the convergence rates for our particular problem via selection of proximal-gradient step β, detailed in upcoming sections.

Problem (5) is the sum of a convex quadratic and a nonconvex regularizer, and the rate of convergence for this problem class can be quantified using [46, Theorem 2], reproduced below.

Algorithm 2 Value-function optimization for (4).

Input: x^0, w_1^0, w_2^0
Initialize: $k = 0$
Define: $\mathcal{H} = \frac{1}{\eta_1}\mathcal{C}^T\mathcal{C} + \frac{1}{\eta_2}\mathcal{A}^T\mathcal{A}$
while not converged **do**
  $x^{k+1} \leftarrow \mathcal{H}^{-1}\left(\frac{1}{\eta_1}\mathcal{C}^T w_1^k + \frac{1}{\eta_2}\mathcal{A}^T(b + w_2^k)\right)$
  $w_1^{k+1} \leftarrow \mathrm{prox}_{\beta\phi}\left(w_1^k - \frac{\beta}{\eta_1}(w_1^k - \mathcal{C}(x^{k+1}))\right)$
  $w_2^{k+1} \leftarrow \mathrm{proj}_{\sigma\mathbb{B}_\psi}\left(w_2^k - \frac{\beta}{\eta_2}(w_2^k - (\mathcal{A}(x^{k+1}) - b)\right)$
  $k \leftarrow k + 1$
end while
Output: w_1^k, w_2^k, x^k

Theorem II.1 (Prox-gradient for Regularized Least Squares)**.** *Consider the least squares objective*

$$\min_z p(z) := \frac{1}{2}\|Gz - g\|^2 + \Phi(z).$$

with p bounded below, and Φ potentially nonsmooth, non-convex, and non-finite valued. With step $\beta = \|G\|_2^{-2} = \sigma_{\max}(G)^{-2}$, the iterates (6) *satisfy*

$$\min_{k=0,\ldots,N} \|\nu^{k+1}\|^2 \leq \frac{\|G\|^2}{N}(p(z^0) - \inf p)$$

where

$$\nu^k = (\|G\|_2^2 I - G^T G)(z^k - z^{k+1}) \in \partial p(z^{k+1})$$

is a subgradient (generalized gradient) of p at z^{k+1}.

We can specialize Theorem II.1 to our case by computing the norm of the least squares system in (5).

Corollary II.2 (Rate for Algorithm 1)**.** *Theorem II.1 applied to Problem 4 gives*

$$\min_{k=0,\ldots,N} \|\nu^{k+1}\|^2 \leq C(\eta_1, \eta_2, \mathcal{C}, \mathcal{A})\frac{1}{N}(p(z^0) - \inf p)$$

with

$$C(\eta_1, \eta_2, \mathcal{C}, \mathcal{A}) = \frac{1}{\eta_1}(c + \|\mathcal{C}\|_F^2) + \frac{1}{\eta_2}(d + \|\mathcal{A}\|_F^2).$$

Problem (4) also admits a different optimization strategy, summarized in Algorithm 2. We can formally minimize the objective in x directly via the gradient, with the minimizer given by

$$x(w) = \mathcal{H}^{-1}\left(\begin{bmatrix}\eta_1^{-1}\mathcal{C}^T & \eta_2^{-1}\mathcal{A}^T\end{bmatrix} w + \eta_2^{-1}\mathcal{A}^T b\right)$$
$$\mathcal{H} = \frac{1}{\eta_1}\mathcal{C}^T\mathcal{C} + \frac{1}{\eta_2}\mathcal{A}^T\mathcal{A}$$

with $w = [w_1, w_2]^T$. From $\mathcal{A}$ and $\mathcal{C}$, we have that $\mathcal{H} \in \mathbb{C}^{mn\times mn}$ (for vectorized x). A direct solution is obtained by taking a Cholesky decomposition $\mathcal{H}$ (as it is SPD) and using back-substitution on the result for $\mathcal{O}\left((mn)^3/3\right)$ FLOPs. However, this requires a Cholesky decomposition every time η_1 and η_2 are updated. This cost and potentially huge nature of $\mathcal{H}$ means that conjugate gradient descent can also be used to solve the least squares problem for $x(w)$. In subsection II-A, we explore *inexact* least square solves, and show that convergence is possible even for minimal CG iterations.

Once $x(w)$ is solved for directly, this expression is plugged back in to give a regularized least squares problem in w alone:

$$\min_{w_1,w_2} p(w) := \phi(w_1) + \left\|\mathcal{F}w - \tilde{b}\right\|^2 \quad \text{s.t.} \quad \psi(w_2) \leq \sigma$$
$$\mathcal{F} = \begin{bmatrix} \frac{1}{\sqrt{\eta_1}}\left(\frac{1}{\eta_1}\mathcal{C}\mathcal{H}^{-1}\mathcal{C}^T - I\right) & \frac{1}{\sqrt{\eta_1\eta_2}}\mathcal{C}\mathcal{H}^{-1}\mathcal{A}^T \\ \frac{-1}{\sqrt{\eta_2\eta_1}}\mathcal{A}\mathcal{H}^{-1}\mathcal{C}^T & \frac{1}{\sqrt{\eta_2}}\left(I - \frac{1}{\eta_1}\mathcal{A}\mathcal{H}^{-1}\mathcal{A}^T\right)\end{bmatrix}$$
$$\widetilde{b} = \begin{bmatrix} \frac{-1}{\sqrt{\eta_1\eta_2}}\mathcal{C}\mathcal{H}^{-1}\mathcal{A}^T b \\ \frac{1}{\sqrt{\eta_2}}\left(\frac{1}{\eta_1}\mathcal{A}\mathcal{H}^{-1}\mathcal{A}^T - I\right)b\end{bmatrix}. \tag{7}$$

Prox-gradient applied to the *value function* $p(w)$ in (7) with step β gives the iteration

$$w^+ = \mathrm{prox}_{\beta\Phi}(w^k - \beta\mathcal{F}^T(\mathcal{F}w - \widetilde{b})) \tag{8}$$

This iteration, as formally written, requires forming and applying the system $\mathcal{F}$ in (7) at each iteration. In practice we compute the $x(w)$ update on the fly, as detailed in Algorithm 2. The equivalence of Algorithm 2 to iteration (8) comes from the following derivative formula for value functions [9]:

$$\mathcal{F}^T(\mathcal{F}w - \widetilde{b})) = \frac{1}{\eta_1}\mathcal{C}^T(\mathcal{C}(x(w)) - w_1) + \frac{1}{\eta_2}\mathcal{A}^T(\mathcal{A}(x(w)) - (w_2 + b)).$$

In order to compute β, and apply Theorem II.1, we first prove the following lemma:

Lemma II.3 (Bound on $\|\mathcal{F}^T\mathcal{F}\|_2$)**.** *The operator norm $\|\mathcal{F}^T\mathcal{F}\|_2$ is bounded above by* $\max\left(\frac{1}{\eta_1}, \frac{1}{\eta_2}\right)$.

Proof. Considering the function

$$\|\mathcal{F}w - \widetilde{b}\|^2 = \min_x \underbrace{\frac{1}{2\eta_1}\|\mathcal{C}(x) - w_1\|^2 + \frac{1}{2\eta_2}\|w_2 - \mathcal{A}(x) + b\|_2^2}_{Q(x,w)},$$

we know that the gradient is given by $\mathcal{F}^T(\mathcal{F}w - \widetilde{b})$, and any Lipschitz bound L gives

$$\|\mathcal{F}^T\mathcal{F}w_1 - \mathcal{F}^T\mathcal{F}w_2\| \leq L\|w_1 - w_2\|,$$

which means $\|\mathcal{F}^T\mathcal{F}\|_2 \leq L$. On the other hand, we can write the right hand side as

$$Q(w, x) = q(Dw, x)$$

where

$$q(z, x) = \frac{1}{2}\left\|z - \begin{bmatrix}\frac{1}{\sqrt{\eta_1}}\mathcal{C}(x) \\ \frac{1}{\sqrt{\eta_2}}\mathcal{A}(x)\end{bmatrix} - \begin{bmatrix}0 \\ b\end{bmatrix}\right\|^2$$

and

$$D = \begin{bmatrix}\frac{1}{\sqrt{\eta_1}} & 0 \\ 0 & \frac{1}{\sqrt{\eta_2}}\end{bmatrix}.$$

Using Theorem 1 of [45] with $g(z) = 0$, we have that the value function

$$\widetilde{q}(z) = \min_x q(z, x)$$

is differentiable, with $\mathrm{lip}(\nabla\widetilde{q}) \leq 1$. Therefore

$$\widetilde{Q}(w) = \min_x Q(w, x)$$

Algorithm 3 Block-coordinate descent for (4).

1: **Input:** x^0, w_1^0, w_2^0
2: Initialize: $k = 0$
3: Define: $\mathcal{H} = \frac{1}{\eta_1}\mathcal{C}^T\mathcal{C} + \frac{1}{\eta_2}\mathcal{A}^T\mathcal{A}$
4: **while** not converged **do**
5: $\quad x^{k+1} \leftarrow \mathcal{H}^{-1}\left(\frac{1}{\eta_1}\mathcal{C}^T w_1^k + \frac{1}{\eta_2}\mathcal{A}^T(b + w_2^k)\right)$
6: $\quad w_1^{k+1} \leftarrow \text{prox}_{\eta_1\phi}\left(\mathcal{C}(x^{k+1})\right)$
7: $\quad w_2^{k+1} \leftarrow \text{proj}_{\sigma\mathbb{B}_\psi}\left(\mathcal{A}(x^{k+1}) - b\right)$
8: $\quad k \leftarrow k + 1$
9: **end while**
10: **Output:** w_1^k, w_2^k, x^k

is also differentiable, with

$$\nabla\widetilde{Q}(w) = D^T\nabla\widetilde{q}(Dw),$$

and hence

$$\text{lip}(\nabla\widetilde{Q}) \leq \|D^T D\|_2 = \max\left(\frac{1}{\eta_1}, \frac{1}{\eta_2}\right).$$

This immediately gives the result. □

Now we can combine iteration (8) with Theorem II.1 to get a rate of convergence for Algorithm 2.

Corollary II.4 (Convergence of Algorithm 2)**.** *When β satisfies*

$$\beta \leq \min(\eta_1, \eta_2),$$

the iterates of Algorithm 2 satisfy

$$\min_{k=0,\ldots,N} \|\nu^{k+1}\|^2 \leq \frac{1}{N}\max\left(\frac{1}{\eta_1}, \frac{1}{\eta_2}\right)(p(w^0) - \inf p))$$

where ν^k is in the subdifferential (generalized gradient) of objective (7) *at w^k. Moreover, if $\eta_1 = \eta_2$, then Algorithm* (2) *is equivalent to block-coordinate descent, as detailed in Algorithm 3.*

Proof. The convergence statement comes directly from plugging the estimate of iteration 8 into Theorem II.1. The equivalence of Algorithm 3 with Algorithm 2 is obtained by plugging in step size $\beta = \eta_1 = \eta_2$ into each line of Algorithm 2. □

An important consequence of Corollary II.4 is that the convergence rate of Algorithm 2 does not depend on $\mathcal{C}$ or $\mathcal{A}$, in contrast to Algorithm 1, whose rate depends on both matrices (Corollary II.2). The rates of both algorithms are affected by (η_1, η_2). We use continuation in η_i, driving (η_1, η_2) to $(0, 0)$ at the same rate, and warm-starting each problem at the previous solution. A convergence theory that takes this continuation into account is left to future work.

Algorithm (3) is similar to the Proximal Alternating Minimization (PAM) algorithm [6]. Indeed PAM and other algorithms, such as the linearized version of PAM called PALM [10] can be used to solve the relaxed problem (2). However the PAM algorithm is different from Algorithm 3, since it requires additional proximal terms. The analysis using the value function reduces problem (2) to a simpler problem, the sum of a quadratic in $\begin{bmatrix} w_1 & w_2 \end{bmatrix}$ and a nonconvex regularizer in $\begin{bmatrix} w_1 & w_2 \end{bmatrix}$, and allows the simple proximal gradient method. The detailed implementation of this approach, with explicit x updates, gives Algorithm 3. Moreover we get a clear rate of convergence for the algorithm and can show that it does not depend on the quantities $\mathcal{C}$ and $\mathcal{A}$.

A. Inexact Least-Squares Solves.

Algorithm 3 has a provably faster rate of convergence than Algorithm 1. The practical performance of these algorithms is compared in Figure 1, which is solving a problem with both a ℓ_1 norm regularizer and ℓ_1 norm BPDN constraint, with $\alpha = \|\mathcal{A}\|_F^{-2}$, $\mathcal{C} = I$, and $\eta_1 = \eta_2 = 10^{-4}$. We see a huge performance difference in practice as well as in theory: the proximal gradient descent from Algorithm 1 yields a slower cost function decay than solving exactly for $x(w)$ as in Algorithm 3. Indeed, Algorithm 3 admits the fastest cost function decay as shown in Corollary II.4, albeit at the expense of more operations per iteration. This is due to the fact that fully solving the least squares problem in Line 5 is not tractable for large-scale problems. Hence, we implement Algorithm 3 inexactly, using the Conjugate Gradient (CG) algorithm. Figure 1 shows the results when we use 1, 5, and 20 CG iterations. Each CG iteration is implemented using matrix-vector products, and at 20 iterations the results are indistinguishable from those of Algorithm 3 with full solves. Even at 5 iterations, the performance is remarkably close to that of of Algorithm 3 with full solves. Algorithm 3 has a natural warm-start strategy, with the x from each previous iteration used in the subsequent least-squares solve using CG. Using a CG method with a bounded number of iterates gives fast convergence and saves computational time. This approach is used in the subsequent experiments.

III. Application to Basis Pursuit Denoise Models

The Basis Pursuit Denoise problem can be formulated as

$$\min_x \phi(x) \quad \text{s.t. } \psi\left(\mathcal{A}(x) - b\right) \leq \sigma \tag{9}$$

where $\psi(\cdot)$ is classically taken to be the ℓ_2 norm and the sparsity-inducing cost-functional $\phi(\cdot)$ is often chosen to be $\|\cdot\|_1$, though more general norms/gauges are also used [43]. In this problem, x represents unknown coefficients that are sparse in a transform domain, while $\mathcal{A}$ is a composition of the observation operator with a transform matrix; popular examples of transform domains include discrete cosine transforms, wavelets, and curvelets. The observed and noisy data b resides in the temporal/spatial domain, and σ is the misfit tolerance. This problem was famously solved with the SPGL1 [42] algorithm for $\psi(\cdot) = \|\cdot\|_2$ while minimizing $\|x\|_1$.

A nonsmooth variant of (9) is very difficult for approaches such as SPGL1, which solves subproblems of the form

$$\min_x \psi\left(\mathcal{A}(x) - b\right) \quad \text{s.t.} \quad \|x\|_1 \leq \tau.$$

and cannot handle nonsmooth functions, let alone the cardinality 'norm' $\psi(\cdot) = \ell_0(\cdot)$. When the observed data is affected by large sparse noise, this smooth constraint used by SPGL1 is ineffective. However, the proposed Algorithm 2 is easily adaptable to different norms/penalties in both cost-functional and constraint. We can solve Problem 9 by applying

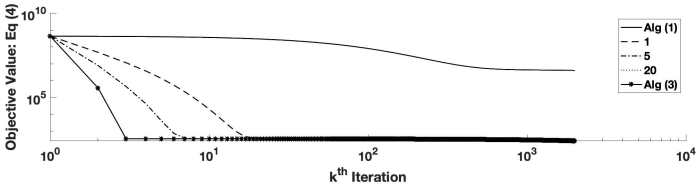


Fig. 1. Objective function decay for Equation 4 with proximal-gradient descent (Algorithm 1), Direct solving (Algorithm 3), and several steps in between where we only partially solve for $\mathcal{H}^{-1}(\ldots)$ with Algorithm 2.

TABLE I
PROJECTORS FOR ℓ_p BALLS.

Norm	$\ell(x)$	$\text{proj}_{\tau\mathbb{B}_\ell}(z)$	Solution
ℓ_2	$\sqrt{\sum_i x_i^2}$	$\begin{cases} z, & \|z\| < \tau \\ \tau z/\|z\|_2, & \|z\| > \tau \end{cases}$	Analytic
ℓ_∞	$\max_i \lvert x_i\rvert$	$\max(\min(x,1),-1)$	Analytic
ℓ_1	$\sum_i \lvert x_i\rvert$	See e.g. [41]	$O(n \ln n)$
ℓ_0	$\sum_i \mathbf{1}_{x_i \neq 0}$	$\begin{cases} z_i, & i \text{ in } \tau \text{ largest indices} \\ 0 & \text{otherwise.} \end{cases}$	Analytic

Algorithm 3 with $\phi(x) = \|x\|_1$, taking $(\eta_1, \eta_2) \to (0,0)$ so that $(w_1, w_2) \to (x, \mathcal{A}(x) - b)$. This means that the w_1 update in Algorithm 3 is the well-known proximal operator of ℓ_1, given by

$$\text{prox}_{\beta\|\cdot\|_1}(x) = \text{sign}(x)\max(0, |x| - \beta).$$

In addition, we know from [36] that the ℓ_1 norm can underestimate true signal values. To that end, our first example in Section IV also solves the sparsity-inducing cost-functional $\|x\|_0$, where our w_1 update in Algorithm 3 becomes

$$\text{prox}_{\beta\|\cdot\|_0}(x)_i = \begin{cases} 0 & |x_i| < \sqrt{2\beta}, \\ \{0, x_i\} & |x_i| = \sqrt{2\beta}, \\ x_i & |x_i| > \sqrt{2\beta} \end{cases} \tag{10}$$

or the *hard-thresholding* operator. We can take many different $\psi(\cdot)$, including $\ell_2, \ell_1, \ell_\infty$, and ℓ_0.

Algorithm 3 is simple to implement. The least squares update in step 4 can be computed efficiently using either factorization with Woodbury, or an iterative method in cases where $\mathcal{A}$ is too large to store.

For the Woodbury approach, we have

$$\left(\eta_2 + \eta_1 \mathcal{A}^T\mathcal{A}\right)^{-1} = \frac{1}{\eta_2} I - \frac{1}{\eta_2^2}\mathcal{A}^T\left(\frac{1}{\eta_1} I + \frac{1}{\eta_2}\mathcal{A}\mathcal{A}^T\right)^{-1}\mathcal{A}. \tag{11}$$

For moderate size systems, we can store Cholesky factor

$$LL^T = \frac{1}{\eta_1} I + \frac{1}{\eta_2}\mathcal{A}\mathcal{A}^T,$$

with $L \in \mathbb{R}^{m\times m}$, and use L with (11) to implement step 4. However, in the seismic/curvelet experiment described below, the left-hand side of Equation 11 is too large to store in memory, but is positive definite. Hence, we solve the resulting linear system in step 4 of Algorithm 3 with CG, using matrix-vector products. The w_1 update is implemented via the ℓ_1 proximal operator (soft thresholding), while the w_2 update requires a projection onto the ℓ_p ball. The projectors used in our experiments are collected in Table I.

The least squares solve for x is when $\mathcal{C}^T$ is an orthogonal matrix or tight frame, so that $\mathcal{C}^T\mathcal{C} = I$; this is the case for Fourier transforms, wavelets, and curvelets. When $\mathcal{A}$ is a restriction operator, as for many data interpolation problems, $\mathcal{A}^T\mathcal{A}$ is a diagonal matrix with zeros and ones, and hence

$$\mathcal{H} = \frac{1}{\eta_1}\mathcal{C}^T\mathcal{C} + \frac{1}{\eta_2}\mathcal{A}^T\mathcal{A}$$

is a diagonal matrix with entries either $\frac{1}{\eta_1}$ or $\frac{1}{\eta_1} + \frac{1}{\eta_2}$; the least squares problem for the x update is then trivial.

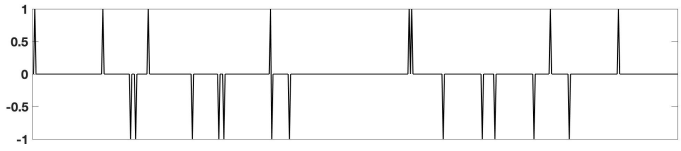
IV. Basis Pursuit De-Noise Experiments

In this application, we consider two examples: the first is a small-scale BPDN to illustrate the proof of concept of our technique, while the second is an application to denoising a common source gather extracted from a seismic line simulated using a 2D BG Compass model.

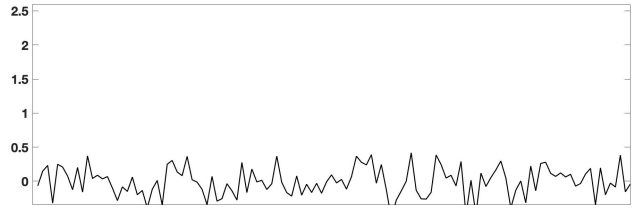
A. Spike-Train BPDN

The first example considers the same model as in (9) where we want to enforce sparsity on x while constraining the data misfit. The variable x is a vector of length n that has values $\{-1, 1\}$ on a random 4% of its entries and zeros everywhere else; represents a spike train that is acted upon by a linear operator, $A \in \mathbb{R}^{n,m}$. A was generated with independent standard Gaussian entries, and $b \in \mathbb{R}^m$ is observed data with large, sparse noise. We take $m = 120$ and $n = 512$. The noise is generated by placing large values on 10% of the observations and assuming everything else was observed cleanly (ie no noise). For this example, we first test the efficacy of using different ℓ_p norms on the residual constraint only, keeping the $\phi(\cdot) = \|\cdot\|_1$. With the addition of large, sparse noise to the data, smooth norms on the residual constraint should not be able to effectively deal with such outlier residuals. With our adaptable formulation, it should be easy to enforce both sparsity in the x domain as well as the residuals. Other formulations, such as SPGL1, do not have this capability. We offer a comparable result to CVX with the linear program $\phi(\cdot),\ \psi(\cdot) = \ell_1$.

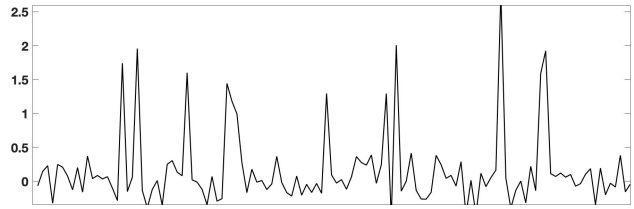
True signal values, the transformed signal, and the observed data is given in Figure 2. The results of solving Problem 9 with cost-function $\phi(\cdot) = \|\cdot\|_1$ are shown in Figure 3 and in Table II. From these, we can clearly see that the ℓ_2 norm is not effective for sparse noise, even at the correct error budget σ. Our approach is resilient to different types of noise since we can easily change the residual ball projection. This is seen by the almost exact accuracy of the ℓ_1 and ℓ_0 norms as choices for $\psi(\cdot)$, with SNR's of 33 and 45 respectively. This is comparable to solving $\phi(\cdot), \psi(\cdot) = \ell_1(\cdot)$ with CVX, as implementation



(a) True Signal: x



(b) True transformed signal: Ax



(c) Observed/noisy values: b

Fig. 2. True signal, transformed signal, and noisy signals used in for the first experiment in Section IV.

with that program yields an SNR of 35, as seen in Table II. CVX, however, fails for larger examples given in subsequent sections.

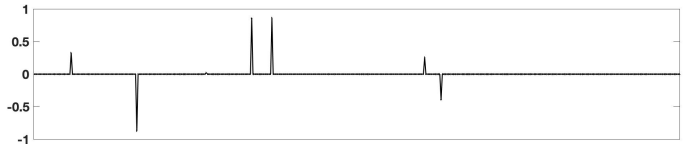
TABLE II
SNR VALUES AGAINST THE TRUE x FOR $\phi(\cdot) = \ell_1$ AND DIFFERENT $\psi(\cdot) = \ell_p$ NORMS WITH SPGL1, CVX, AND ALGORITHM 3.

Spike-Train BPDN (9)	
$\psi(\cdot)$/Method	SNR
ℓ_2 with SPGL1	0.2007
ℓ_2 with Alg.3	0.2032
ℓ_1 with CVX	35.3611
ℓ_1 with Alg.3	33.7281
ℓ_∞ with Alg.3	-0.6708
ℓ_0 with Alg.3	45.0601

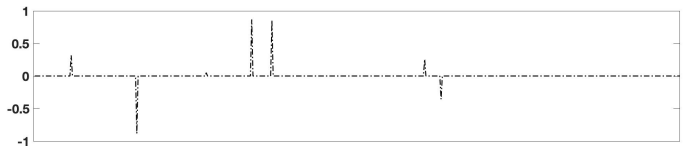
TABLE III
SNR VALUES AGAINST THE TRUE x FOR DIFFERENT COMBINATIONS OF SPARSITY-INDUCING $\phi(\cdot) = \ell_1$, ℓ_0 AND $\psi(\cdot) = \ell_2$, ℓ_0 NORMS WITH SPGL1 AND ALGORITHM 3.

Spike-Train BPDN (9)	
$\phi(\cdot)/\psi(\cdot)$/Method	SNR
ℓ_1 / ℓ_2 with SPGL1	0.2007
ℓ_1 / ℓ_2 with Alg.3	0.2031
ℓ_1 / ℓ_0 with Alg.3	45.0440
ℓ_0 / ℓ_2 with Alg.3	-1.2828
ℓ_0 / ℓ_0 with Alg.3	44.4239

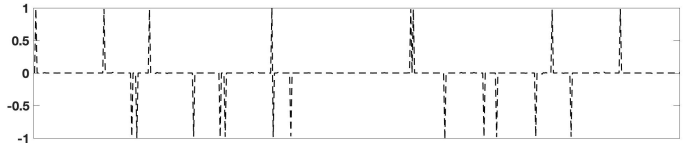
Secondly, we conduct a similar experiment, but test the resilience of our formulation to different sparsity-inducing metrics; this amounts to changing the $\phi(\cdot)$ in the cost-function of Problem 9. Specifically, we change it to the $\ell_0(\cdot)$ norm (since we know we have to promote sparsity), and conduct a similar experiment as above - that is, we solve Problem 9,



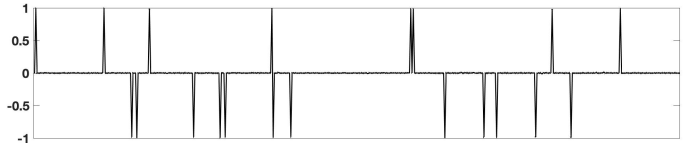
(a) SPGL1, $\psi = \ell_2$



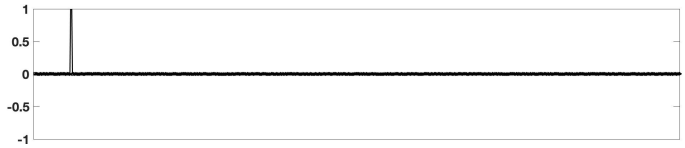
(b) $\psi = \ell_2$



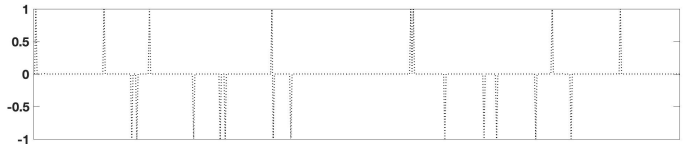
(c) $\psi = \ell_1$



(d) CVX, $\psi = \ell_1$



(e) $\psi = \ell_\infty$



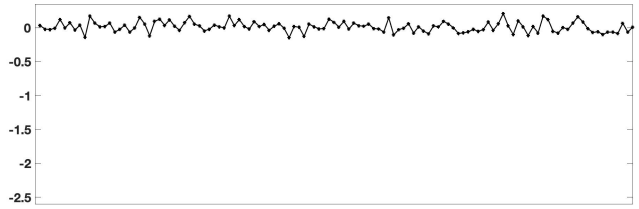
(f) $\psi = \ell_0$

Fig. 3. Basis Pursuit Denoising results on Problem 9 with cost-functional $\phi(\cdot) = \|\cdot\|_1$ for a randomly generated linear model with large, sparse noise. Here, $\psi(\cdot)$ can be any of the ℓ_p norms in Table I.

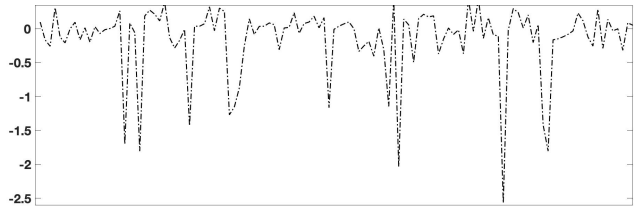
desiring a sparse signal from observed data with large sparse outliers. Hence, we want sparsity in *both* our signals and our observations. Note that this equations to setting w_1 as the proximal operator of the ℓ_0 norm, which is the hard-thresholding operator detailed in Equation 10. Our results are shown in Table III. We see that the in Table III, typical Problem 9 with ℓ_1 norm solved with both Algorithm 3 and SPGL1 does not perform well. Indeed, even inducing sparsity with the $\ell_0(\cdot)$ cost-functional and $\psi(\cdot) = \ell_2(\cdot)$ in the constraints, we do not get the desired performance. However, inducing sparsity in the cost-functional (with both ℓ_1, and ℓ_0) gives us the most favorable results, with SNRs of 45 and 44 respectively. Recovered signals are shown in Figure 5.

B. Curvelet Interpolation

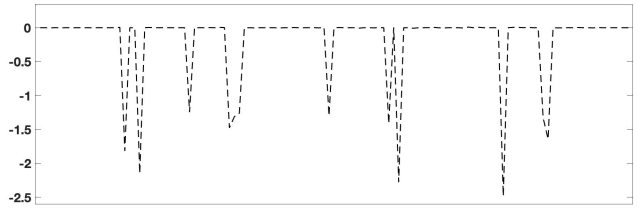
The next test of the BPDN formulation is for a common source gather where entries are both omitted and corrupted



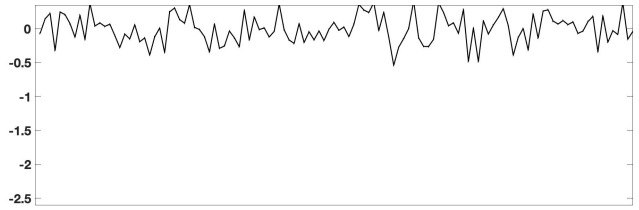
(a) SPGL1, $\psi = \ell_2$



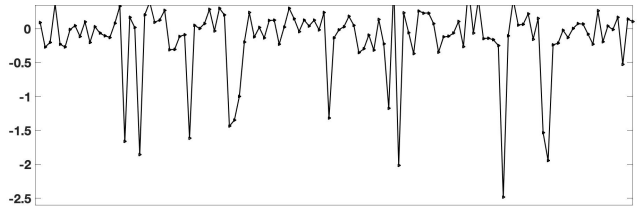
(b) $\psi = \ell_2$



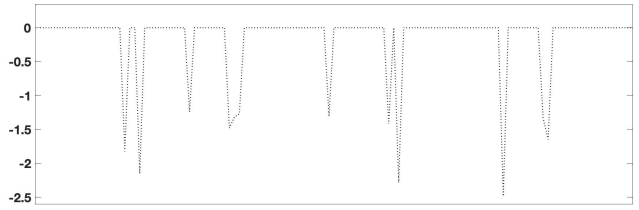
(c) $\psi = \ell_1$



(d) CVX, $\psi = \ell_1$

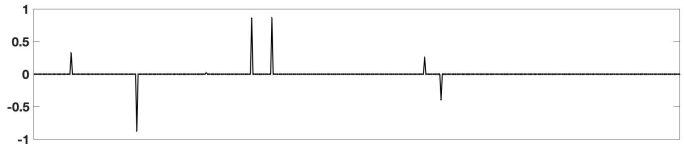


(e) $\psi = \ell_\infty$

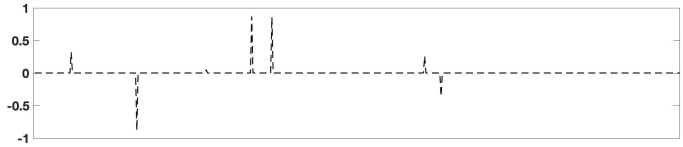


(f) $\psi = \ell_0$

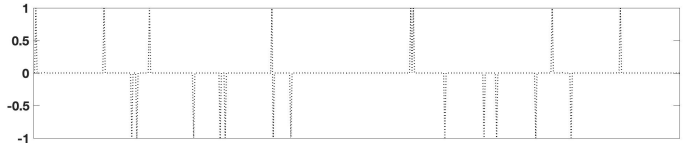
Fig. 4. Residuals after algorithm termination for solving Problem 9 where $\phi(\cdot) = \|\cdot\|_1$, and using SPGL1 and Alg. 3 with different $\psi(\cdot) = \ell_p$ norms. Note how the ℓ_1 and ℓ_0 norms can capture the outliers only.



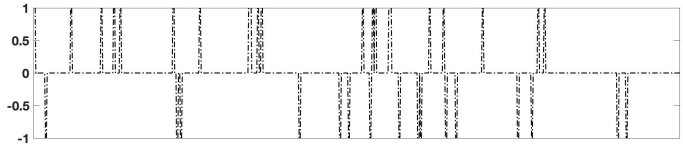
(a) SPGL1: $\phi(\cdot) = \ell_1,\ \psi(\cdot) = \ell_2$



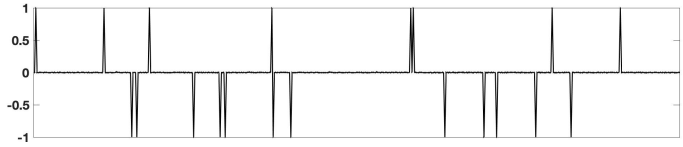
(b) Alg 3: $\phi(\cdot) = \ell_1,\ \psi(\cdot) = \ell_2$



(c) Alg 3: $\phi(\cdot) = \ell_1,\ \psi(\cdot) = \ell_0$



(d) Alg 3: $\phi(\cdot) = \ell_0,\ \psi(\cdot) = \ell_2$



(e) Alg 3: $\phi(\cdot) = \ell_0,\ \psi(\cdot) = \ell_0$

Fig. 5. Residuals after algorithm termination for solving Problem 9 where $\phi(\cdot) = \ell_1$, and using SPGL1 and Alg. 3 with different $\psi(\cdot) = \ell_p$ norms. Note how the ℓ_1- and ℓ_0 norms can capture the outliers only.

with synthetic noise. The data set contains time samples with a temporal-interval of 4ms, and the spatial sampling is 10m. Here, the objective function looks for sparsity in the curvelet domain, while the residual constraint seeks to match observed data within a certain tolerance σ. First, we note that doing interpolation only without added noise yields an SNR of approximately 13 for all formulations and algorithms; that is, all ℓ_p norms for Algorithm 3 and SPGL1. Here, we again want to enforce sparsity both in the curvelet domain ($\mathcal{C}(x)$) and the data residual ($\|\mathcal{A}(x)-b\|$), for which our algorithm is uniquely adapted to solve.

Following the first experiment of the spike-train example in subsection IV-A, we add large sparse noise to a handful of data points; in this case, we added large values to a random 1% of observations (this does not include omitted entries). The noise added is approximately 120, while the observed data can range from 0 to 30. The interpolated and denoising results are shown in Figure 6 and Table IV. Large, sparse noise cannot be filtered effectively by a smooth norm constraint, using either Algorithm 3 or SPGL1. However, ℓ_1 and ℓ_0 norms effectively handle such noise, and can be optimized using our approach. The SNR's for these implementations are approximately 15 respectively, approaching that of the noiseless data mentioned above.

We then repeated the experiment done in subsection IV-A where $\phi(\cdot)$ is changed to be the ℓ_0 norm as well as the ℓ_1 norm. Similarly to Table III, we have that sparsity in *both* $\phi(\cdot)$ and $\psi(\cdot)$ can efficiently recapture the original signal. Table V shows that setting $\phi(\cdot) = \ell_0$ together with sparse constraints

TABLE IV
CURVELET INTERPOLATION AND DENOISING RESULTS FOR SPGL1 AND ALGORITHM 3 $\phi(\cdot) = \ell_1$ AND $\psi(\cdot) = \ell_p$ NORMS FOR BPDN (9).

Curvelet Interpolation & Denoising			
$\psi(\cdot)$/Method	SNR	SNR w_1	Time (s)
ℓ_2 with SPGL1	1.4857	-	68.4 (early stoppage)
ℓ_2 with Alg.3	0.9769	0.9693	6199
ℓ_1 with Alg.3	14.9574	14.9436	5037
ℓ_∞ with Alg.3	0.0000	0	1527
ℓ_0 with Alg.3	14.9212	14.9142	6262

TABLE V
CURVELET INTERPOLATION AND DENOISING RESULTS FOR SPGL1 AND ALGORITHM 3 WITH DIFFERENT COMBINATIONS OF SPARSITY-INDUCING $\phi(\cdot) = \ell_1, \ell_0$, AND $\psi(\cdot) = \ell_2$, ℓ_0 NORMS FOR BPDN (9).

Curvelet Interpolation & Denoising			
$\phi(\cdot)/\psi(\cdot)$/Method	SNR	SNR w_1	Time (s)
ℓ_1 / ℓ_2 with SPGL1	1.4857	-	51.4 (early stoppage)
ℓ_1 / ℓ_2 with Alg.3	0.9769	0.9693	4043
ℓ_1 / ℓ_0 with Alg.3	14.9212	14.9142	4256
ℓ_0 / ℓ_2 with Alg.3	0.1542	0.1199	4084
ℓ_0 / ℓ_0 with Alg.3	14.042	13.7999	4086

(either ℓ_1 or ℓ_0) recapture the signal quite well; both have an SNR of approximately 14, over SNR's of close to 1 for every other combination.

V. EXTENSION TO LOW-RANK MODELS

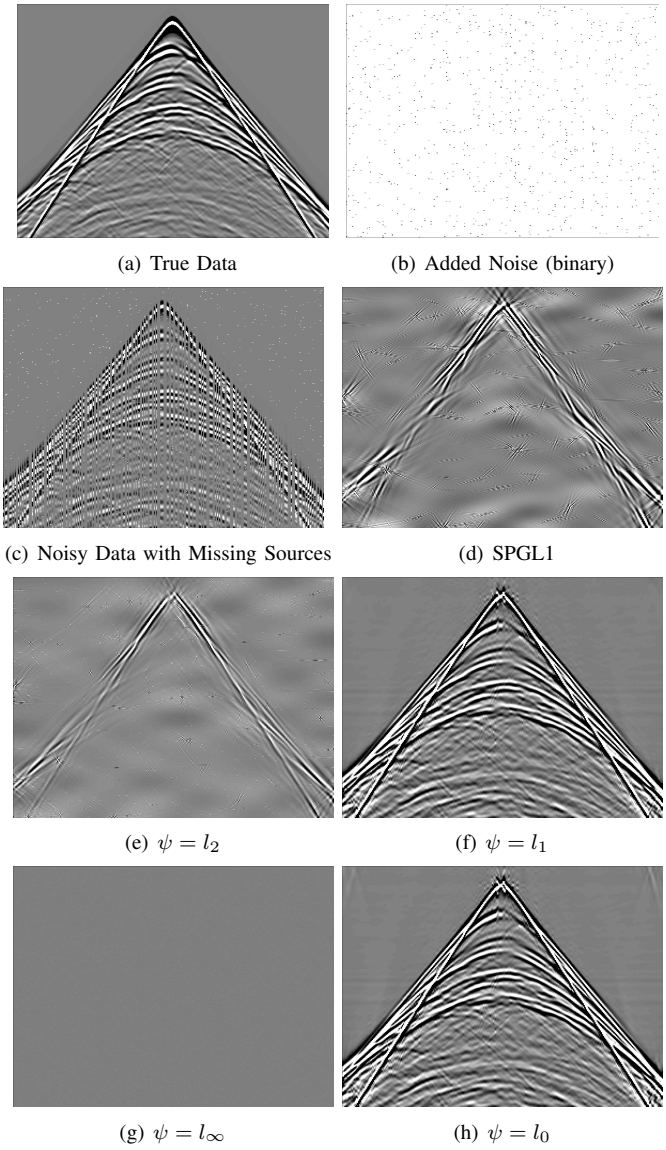
Treating the data as having a matrix structure gives additional regularization tools — in particular low-rank structure in particular domains. The BPDN formulation for residual-constrained low-rank interpolation is given by

$$\min_X \|X\|_* \quad \text{s.t. } \psi\left(\mathcal{A}(X) - b\right) \leq \sigma \tag{12}$$

for $X \in \mathbb{C}^{m\times n}$, $\mathcal{A} : \mathbb{C}^{n\times m} \to \mathbb{C}^p$ is a linear masking operator from full to observed (noisy) data b, and σ is the misfit tolerance. The nuclear norm $\|X\|_*$ is the ℓ_1 norm of the singular values of X. Solving the problem (12) requires using a decision variable that is the size of the data, as well as updates to this variable that require SVDs at each iteration. It is much more efficient to model X is a product of two matrices L and R, given by

$$\min_{L,R} \frac{1}{2}(\|L\|_F^2 + \|R\|_F^2) \quad \text{s.t. } \psi\left(\mathcal{A}(LR^T) - b\right) \leq \sigma \tag{13}$$

where $L \in \mathbb{C}^{n\times k}$, $R \in \mathbb{C}^{m\times k}$, and LR^T is the low-rank representation of the data. The solution is guaranteed to be at most rank k, and in addition, the regularizer $\frac{1}{2}(\|L\|_F^2 + \|R\|_F^2)$ is an upper bound for $\|LR^T\|_*$, the sum of singular values of LR^T, further penalizing rank by proxy. The decision variables then have combined dimension $k(m \times n)$, which is much smaller than the nm variables required by convex formulations. When ψ is smooth, the problems are solved using a continuation that interchanges the roles of the objective and constraints, solving a sequence of problems where $\psi\left(\mathcal{A}(LR^T) - b\right)$ is minimized over the ℓ_2 ball [4] using projected gradient; an approach we call SPGLR below, which is a modification of SPGL1 specifically adapted for matrix completion.



(a) True Data (b) Added Noise (binary)

(c) Noisy Data with Missing Sources (d) SPGL1

(e) $\psi = l_2$ (f) $\psi = l_1$

(g) $\psi = l_\infty$ (h) $\psi = l_0$

Fig. 6. Interpolation and denoising results for BPDN in the curvelet domain. Observe the complete inaccuracy of smooth norms with large, sparse noise.

When ψ is not smooth, SPGLR does not work and there are no available implementations for (13). Nonsmooth ψ arise when we want the residual to be in the ℓ_1 norm ball, so we are robust to outliers in the data, and can exactly fit inliers. We now extend Algorithm 3 to this case. For any ψ (smooth or nonsmooth), we introduce a latent variable W for the data matrix, and solve

$$\min_{L,R,W} \left\| \begin{matrix} L \\ R \end{matrix} \right\|_F^2 + \frac{1}{2\eta}\|W - LR^T\|_2^2, \quad \text{s.t. } \|\mathcal{A}(W) - b\|_p \leq \sigma \tag{14}$$

with η a parameter that controls the degree of relaxation; as $\eta \downarrow 0$ we have $W \to LR^T$. The relaxation allows a simple block-coordinate descent, detailed in the simple to implement Algorithm 4. It requires two least squares solves (for L and R), which are inherently parallelizable. It also requires a projection of the updated data matrix estimates LR^T onto the σ-level set of the misfit penalty ψ.

Algorithm 4 Block-Coordinate Descent for (14).

Input: w_0, L_0, R_0
Initialize: $k = 0$
while not converged **do**
  $L_{k+1} \leftarrow \left(I + \eta R_k^T R_k\right)^{-1} (\eta W_k R_k)$
  $R_{k+1} \leftarrow (\eta W_k^T L_{k+1}) \left(I + \eta L_{k+1}^T L_{k+1}\right)^{-1}$
  $W_{k+1} \leftarrow \begin{cases} (L_{k+1} R_{k+1}^T)_{ij}, & (i,j) \in X_{obs} \\ \text{proj}_{\mathbb{B}_{\psi,\sigma}} \left(\mathcal{A}(L_{k+1} R_{k+1}^T) - b\right), & \text{o.w.} \end{cases}$
  $k \leftarrow k + 1$
end while
Output: w_k, L_k, R_k

For unobserved data $(i,j) \notin X_{obs}$, we have $W_{ij} = (LR^T)_{ij}$. For observed data, let v denote $\mathcal{A}(LR^T)$. Then the W update step is given by solving

$$\min_w \|w - v\|_2^2, \quad \text{s.t.} \|w - b\|_p \leq \sigma.$$

Using the simple substitution $z = w - b$, the we get

$$\min_z \|z - (v - b)\|_2^2, \quad \text{s.t.} \quad \|z\|_p \leq \sigma$$

which is precisely the projection of $\mathcal{A}(LR^T) - b$ onto $\mathbb{B}_{\psi,\sigma}$, the σ-level set of ψ. We use the same projectors for $\psi \in \{l_0, l_1, l_2, l_\infty\}$ as in Section IV, see Table I. The convergence criteria for Algorithm 4 is based on the optimality of the quadratic subproblems in L, R and feasibility measure of $W - LR^T$, though in practice we compare performance of algorithms based on a computational budget. This block-coordinate descent scheme converges to a stationary point of Equation 14 by [40, Theorem 4.1].

Implementing block-coordinate descent on these forms until convergence produces the completed low-rank matrix. Setting $\nu = \|LR^T - w\|_2^2$, we iterate until $\nu < 1e-5$ or a maximum number of iterations is reached. In the next section, we develop an application of this method to seismic interpolation and denoising.

VI. 4D Matrix completion with Denoising

There are two main requirements when using the rank-minimization based framework for seismic data interpolation and denoising: *(i)* underlying seismic data should exhibit low-rank structure (singular values should decay fast) in some transform domain, and, *(ii)* subsampling and noise destroy the low-rank structure (singular values decay slow) in that domain. For exploiting the low-rank structure during interpolation and denoising, we follow the matricization strategy proposed by [14]. The matricization (source-x, source-y), i.e., placing both the source coordinates along the columns, gives slow-decay of singular values, while the matricization (source-x, receiver-x) gives fast decay of the singular values. Subsampling destroys the fast singular value decay in the (source-x, receiver-x) matricization, but not in the (source-x, receiver-y) matricization. Thus the latter is more effective for low-rank interpolation. These concepts are discussed in great detail by [25], [26].

Similar to the BPDN experiments, we want to show that nonsmooth constraints on the data residual can be effective for dealing with large, sparse noise. The smooth ℓ_2 norm that is most common in BPDN problem will fail in such examples, thereby leading to better data estimation with the implementation of non-smooth norms on the residuals. Thus, the goal of the below experiments is to show that enforcing sparsity in the singular values (ie low-rank) and sparsity in the residual constraint can be more effective with large, sparse noise than smooth residual constraints solved by most contemporary algorithms.

A. Experiment Description

This example demonstrates the efficacy of the proposed approach using data created by a 5D dataset based on a complex SEG/EAGE overthrust model simulation [1]. The dimension of the model is $5\,\text{km} \times 10\,\text{km} \times 10\,\text{km}$ and is discretized on a $25\,\text{m} \times 25\,\text{m} \times 25\,\text{m}$ grid. The simulated data contains 201×201 receivers sampled at 50 m and 101×101 sources sampled at 100 m. We apply the Fourier transform along the time domain and extract a frequency slice at 10 Hz as shown in Figure 7(a), which is a 4D object (source-x, source-y, receiver-x and receiver-y). We eliminate 80% of the sources and add large sparse outliers from the random gaussian distribution $\mathcal{N}(0, a_i \max(X_{s_i}))$ (mean zero and variance on the order of the largest value in that particular source). The 10 generated values with the highest magnitudes are kept, and these are randomly added to observations in the remaining sources (Figure 7(f)). The largest value of our dataset is approximately 40, while the smallest is close to zero. Thus, we are essentially increasing/decreasing 1% of the entries by several orders of magnitude, which contaminates the data significantly, especially if the original entry was nearly 0. For all low-rank completion and denoising, we let $a_i = 10^{-1}$ except where we test the efficacy of Algorithm 4 against different noise levels σ. The objective is to recover missing sources and eliminate noise from observed data. We use a rank of $k = 75$ for the formulation (that is, $L \in \mathbb{C}^{n \times 75}$ and similarly for R), and run all algorithms for 150 iterations, using a fixed computational budget. We perform three experiments on the same dataset: 1) Denoising only (Figure 7(c)); 2) Interpolation only (Figure 7(d)); and 3) Combined Interpolation and Denoising (Figure 7(f)). Since we have ground truth, we pick σ to be the exact difference between generated noisy data and the true data; σ for the l_0 norm is a cardinality measure, so it is set to number of noisy points added.

B. Results

Tables VI-VIII display SNR values for different algorithms and formulations for the three types of experiments, and Figures 8-10 display the results for a randomly selected number of sources for the three experiments. Even a small number of outliers can greatly impact the quality of the low-rank denoising and interpolation for the standard, smoothly residual-constrained algorithms. The denoising only results (Figure 8, Table VI) show that all methods perform well when all sources are available. The interpolation only results (Figure 9, Table VII) show that all constraints perform well in interpolating the missing data. This makes sense, as all algorithms will simply

TABLE VI
4D DENOISING RESULTS FOR SPGLR AND ALGORITHM 4 FOR SELECTED ℓ_p NORMS.

4D Monochromatic Denoising			
Method/$\psi(\cdot)$	SNR	SNR-W	Time (s)
ℓ_2 with SPGLR	11.7489	-	16530
l_2 with Alg.4	11.7463	-2.3338	9430
l_1 with Alg.4	11.7638	-2.3063	11546
l_∞ with Alg.4	11.7456	-2.3338	12108
l_0 with Alg.4	17.9595	48.8607	11569

TABLE VII
4D INTERPOLATION RESULTS FOR SPGLR AND ALGORITHM 4 FOR SELECTED ℓ_p NORMS.

4D Monochromatic Interpolation			
Method/$\psi(\cdot)$	SNR	SNR-W	Time (s)
ℓ_2 with SPGLR	16.3976	-	5817
l_2 with Alg.4	16.0629	16.5424	7526
l_1 with Alg.4	16.0692	16.5491	7996
l_∞ with Alg.4	16.0627	16.5423	8119
l_0 with Alg.4	16.0096	16.4728	6848

TABLE VIII
4D COMBINED DENOISING AND INTERPOLATION RESULTS FOR SPGLR AND ALGORITHM 4 FOR SELECTED ℓ_p NORMS.

4D Monochromatic Denoising & Interpolation			
Method/$\psi(\cdot)$	SNR	SNR-W	Time (s)
ℓ_2 with SPGLR	-3.2906	-	8712
l_2 with Alg.4	0.9185	-0.3321	6802
l_1 with Alg.4	0.9193	-0.3235	8068
l_∞ with Alg.4	0.9185	-0.3321	8117
l_0 with Alg.4	16.0655	16.5445	6893

TABLE IX
4D INTERPOLATION (LEFT), DENOISING (CENTER), AND COMBINED (RIGHT) SNR RESULTS FOR ALGORITHM 4 WITH $\phi(\cdot) = \ell_0$. THE NUMBER OF OUTLIERS IS CONSTANT PER SOURCE (10).

4D Monochromatic Denoising & Interpolation						
% Obs.	Int	σ	DN	%	σ	Both
50	17.7120	4.0e6	17.9597	50	2.1e6	17.7170
40	17.5445	5.2e7	17.9597	40	2.0e7	17.5459
30	17.2183	6.0e8	17.9597	30	1.8e8	17.2136
20	16.0522	6.9e9	17.9596	20	1.3e9	16.0263
10	9.2123	7.7e10	17.9596	10	7.8e9	9.2602

favor the low-rank nature of the data. However, the combined denoising and interpolation dataset shows that the ℓ_0 norm approach does far better than any smooth norm in comparable time. Table VIII shows that when data for similar sources is absent/not observed, the smoothly-constrained formulations fail completely. When noise is added to the low-amplitude section of the observed data, the smoothly-constrained norms fail drastically, while the ℓ_0 norm can effectively remove the errors. This is starkly evident in Figures 10(a)-10(e), where all except Figure 10(e) are essentially noise; the result is supported by the SNR values in Table VIII. While Figures 10(a)-10(e) can mostly capture the structure of the data where there were nonzero values (ie where the seismic wave is observed in the upper left corner of each source), only the ℓ_0 norm can capture the areas of lower energy data.

Tables IX-X give performance results across noise levels and degree of 'missingness' when using Algorithm 4. The combined problem, where a lot of data is omitted and outliers are present, is harder than either of the problems separately. We vary the percentage of sources omitted from 50% - 90%, and also vary the noise observed in two ways: (1) adding more noise while keeping number of outliers per source constant (nominally at 10), and (2) adding more noise by adding outliers at similar noise levels. The results are shown in Table IX and Table X. From Table IX, adding more noise to the outliers does not affect our results; the projection onto the ℓ_0 norm handles any obvious outlier. On the other hand, adding more outliers at varying magnitudes affects the results markedly. Increasing the number of outliers to 125 per source observed (which is roughly 1% of total data) affects performance of the ℓ_0 projection. From Table X, performance with added noise alone degrades slowly, but once we start omitting sources, the outlier/normal observation ratio increases drastically for each 10% of sources withheld, we see decreased performance.

VII. CONCLUSIONS

We proposed a new approach for level-set formulations, including basis pursuit denoise and residual-constrained low-rank formulations. The approach is easily adapted to a variety of nonsmooth and nonconvex data constraints. The resulting problems are solved using Algorithm 2 and 4; which require only that the penalty ψ has an efficient projector. The algorithms are simple, scalable, and efficient. Sparse curvelet denoising and low-rank interpolation of a monochromatic slice from the 4D seismic data volumes demonstrate the potential of the approach.

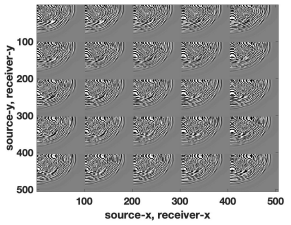
A particular quality of the seismic denoising and interpolation problem is that the amplitudes of the signal have significant spatial variation. The error in the data is a much larger problem for low-amplitude data. This quality makes it very difficult to obtain reasonable results using Gaussian misfits and constraints. Nonsmooth exact formulations (including ℓ_1 and particularly ℓ_0) appear to be extremely well-suited for this magnified heteroscedastic issue.

VIII. ACKNOWLEDGEMENTS

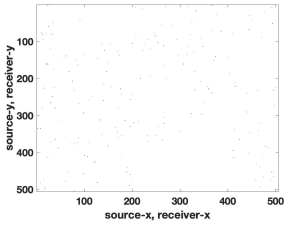
The authors acknowledge support from the Department of Energy Computational Science Graduate Fellowship, which is provided under grant number DE-FG02-97ER25308, and the Washington Research Foundation Data Science Professorship.

TABLE X
4D INTERPOLATION (LEFT), DENOISING (CENTER), AND COMBINED (RIGHT) SNR RESULTS FOR ALGORITHM 4 WITH $\phi(\cdot) = \ell_0$. THE '#OUT' COLUMN GIVES THE NUMBER OF OUTLIERS IS PER SOURCE.

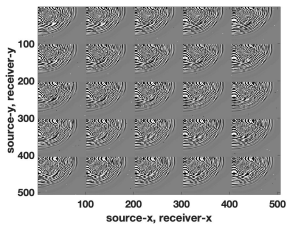
4D Monochromatic Denoising & Interpolation							
% Obs.	Int	σ	DN	%	σ	# Out.	Both
50	17.712	2.2e7	17.955	50	2.2e7	5	17.714
40	17.544	1.6e8	5.055	40	1.6e8	35	1.157
30	17.218	3.4e8	7.629	30	3.4e8	65	-1.019
20	16.052	5.5e8	5.519	20	5.5e8	95	-4.588
10	9.212	7.9e8	3.927	10	7.9e8	125	-9.627



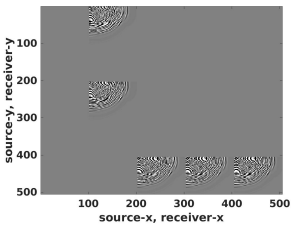
(a) Fully sampled monochromatic slize at 10 Hz.



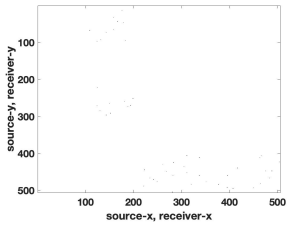
(b) Noisy data alone (binary). Sparse noise was added by keeping the top 10 entries generated from a normal distribution with mean zero and variance $0.1 \max(X_{s_i})$



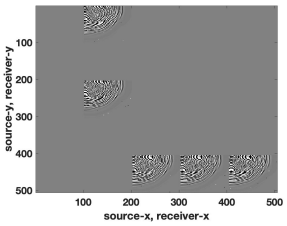
(c) Observed noisy data.



(d) Subsampled noiseless data. We omitted 80% of sources.



(e) Subsampled and noise, with noise only present (binary).



(f) Subsampled and noisy data. We again omitted 80% of sources and added the noise described above to the rest of the sources.

Fig. 7. True data and three different experiments for testing our completeness algorithm.

Code for this paper is listed at: https://github.com/rjbaraldi/bpdn-with-nonsmooth-constraints.

REFERENCES

[1] F. Aminzadeh, N. Burkhard, L. Nicoletis, F. Rocca, and K. Wyatt. Seg/eaeg 3-d modeling project: 2nd update. *The Leading Edge*, 13(9):949–952, 1994.

[2] A. Aravkin, S. Becker, V. Cevher, and P. Olsen. A variational approach to stable principal component pursuit. *UAI Proceedings*, 2014.

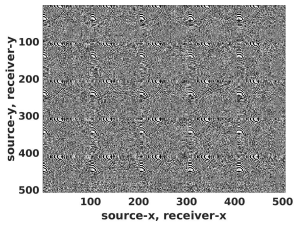
[3] A. Y. Aravkin, J. V. Burke, D. Drusvyatskiy, M. P. Friedlander, and S. Roy. Level-set methods for convex optimization. *To appear in Mathematical Programming, Series B.*, 2018.

[4] A. Y. Aravkin, R. Kumar, H. E. Mansour, B. Recht, and F. J. Herrmann. Fast methods for denoising matrix completion formulations, with applications to robust seismic data interpolation. *SIAM J. Scientific Computing*, 36, 2014.

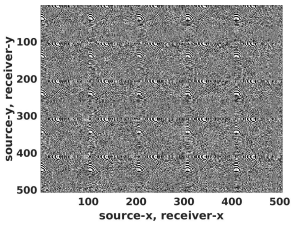
[5] H. Attouch, J. Bolte, and B. Fux Svaiter. Convergence of descent methods for semi-algebraic and tame problems: proximal algorithms, forward–backward splitting, and regularized gauss–seidel methods. *Mathematical Programming*, 137(1-2):91–129, 2013.

[6] H. Attouch, J. Bolte, P. Redont, and A. Soubeyran. 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.

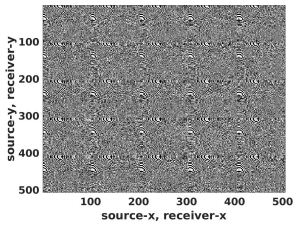
[7] G. Banjac and P. J. Goulart. A novel approach for solving convex problems with cardinality constraints. *IFAC-PapersOnLine*, 50(1):13182–13187, 2017.



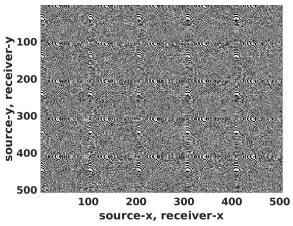
(a) SPGLR



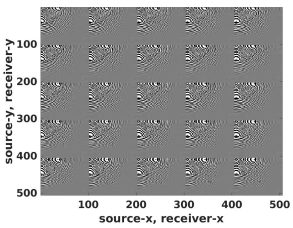
(b) $\psi = l_2$



(c) $\psi = l_1$

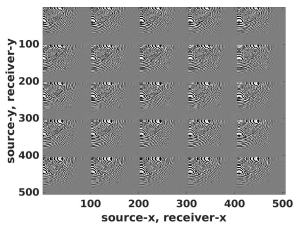


(d) $\psi = l_\infty$

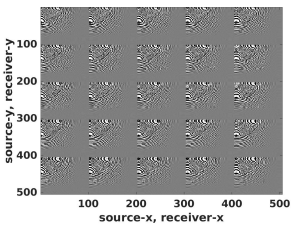


(e) $\psi = l_0$

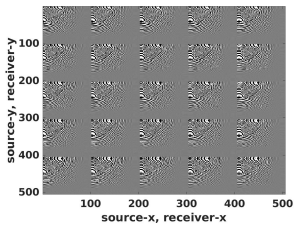
Fig. 8. Denoising-only results.



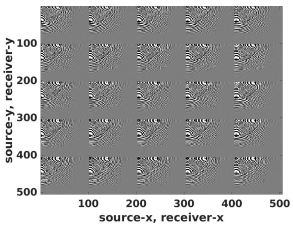
(a) SPGLR



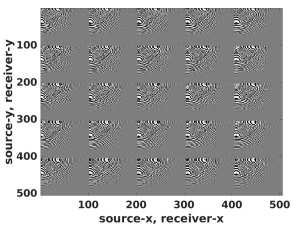
(b) $\psi = l_2$



(c) $\psi = l_1$

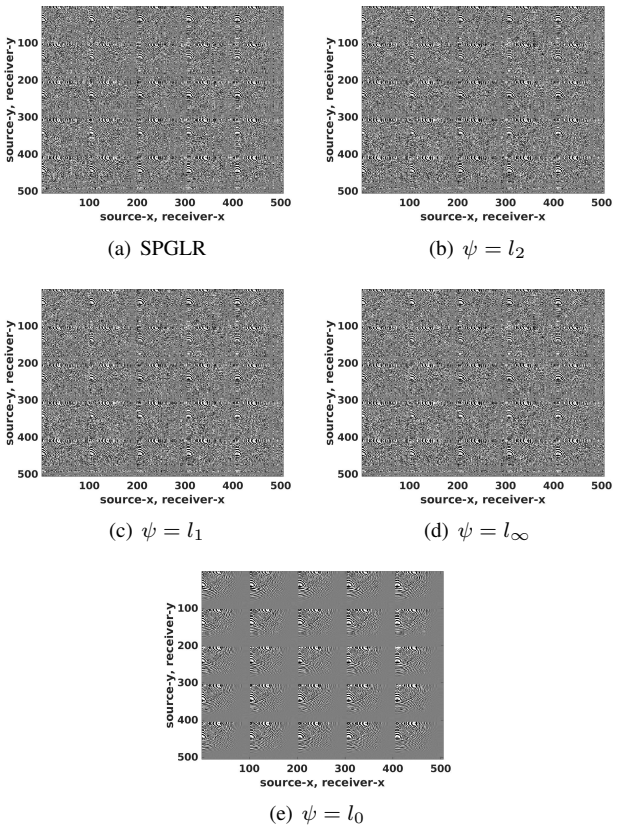


(d) $\psi = l_\infty$



(e) $\psi = l_0$

Fig. 9. Interpolation-only results.



(a) SPGLR (b) $\psi = l_2$

(c) $\psi = l_1$ (d) $\psi = l_\infty$

(e) $\psi = l_0$

Fig. 10. Interpolation and Denoising results.

[8] A. Beck and Y. C. Eldar. Sparsity constrained nonlinear optimization: Optimality conditions and algorithms. *SIAM Journal of Optimization*, 23(3):1480–1509, 2013.

[9] B. M. Bell and J. V. Burke. Algorithmic differentiation of implicit functions and optimal values. In *Advances in Automatic Differentiation*, pages 67–77. Springer, 2008.

[10] J. Bolte, S. Sabach, and M. Teboulle. Proximal alternating linearized minimization for nonconvex and nonsmooth problems. *Mathematical Programming*, 146(1-2):459–494, 2014.

[11] E. J. Candès and T. Tao. Near-optimal signal recovery from random projections: universal encoding strategies. *IEEE Transactions on Information Theory*, 52(12):5406–5425, 2006.

[12] E. J. Candès, L. Xiaodong, Y. Ma, and J. Wright. Robust principal component analysis? *Journal of the ACM*, 58(3):11, 2011.

[13] S. S. Chen, D. L. Donoho, and M. A. Saunders. Atomic decomposition by basis pursuit. *SIAM Journal on Scientific Computing*, 20(1):33–61, 1998.

[14] C. Da Silva and F. J. Herrmann. Optimization on the hierarchical tucker manifold–applications to tensor completion. *Linear Algebra and its Applications*, 481:131–173, 2015.

[15] D. Davis and W. Yin. Convergence rate analysis of several splitting schemes. In *Splitting Methods in Communication, Imaging, Science, and Engineering*, pages 115–163. Springer, 2016.

[16] G. Demoment. Image reconstruction and restoration: Overview of common estimation structures and problems. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 37(12):259–268, December 1989.

[17] D. C. Dobson and F. Santosa. Recovery of blocky images from noisy and blurred data. *SIAM Journal of Applied Mathematics*, 56(4):1181–1198, 1994.

[18] D. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4):1289–1306, 2006.

[19] D. Driggs, S. Becker, and A. Aravkin. Adapting regularized low-rank models for parallel architectures. *SIAM Journal on Scientific Computing*, 41(1):A163–A189, 2019.

[20] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least angle regression. *The Annals of Statistics*, 32(2):407–99, 2004.

[21] R. Foygel and L. Mackey. Corrupted sensing: Novel guarantees for separating structured signals. *IEEE Transactions on Information Theory*, 60(2):1223–1247, 2014.

[22] F. Girosi. An equivalence between sparse approximation and support vector machines. *Neural Comp.*, 10(6):1455–1480, 1998.

[23] W. Ha and R. Foygel Barber. Robust pca with compressed data. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 1936–1944. Curran Associates, Inc., 2015.

[24] F. J. Herrmann and G. Hennenfent. Non-parametric seismic data recovery with curvelet frames. *Geophysical Journal International*, 173(1):233–248, 2008.

[25] A. Kadu and R. Kumar. Decentralized full-waveform inversion. Submitted to EAGE on January 15, 2018, 2018.

[26] R. Kumar, C. Da Silva, O. Akalin, A. Y. Aravkin, H. Mansour, B. Recht, and F. J. Herrmann. Efficient matrix completion for seismic data reconstruction. *Geophysics*, 80(5):V97–V114, 2015.

[27] R. Kumar, O. López, D. Davis, A. Y. Aravkin, and F. J. Herrmann. Beating level-set methods for 5-d seismic data interpolation: A primal-dual alternating approach. *IEEE Transactions on Computational Imaging*, 3(2):264–274, June 2017.

[28] F. Lin, M. Fardad, and M. Jovanović. Design of optimal sparse feedback gains via the alternating direction method of multipliers. *IEEE Transactions on Automatic Control*, 58(9):2426–2431, 2013.

[29] M. Lustig, D. Donoho, and J. Pauly. Sparse mri: The application of compressed sensing for rapid mr imaging. *Magnetic Resonance in Medicine*, 58:1182–95, 2007.

[30] J. Mairal, F. Bach, and J. Ponce. Sparse modeling for image and vision processing. *Foundations and Trends in Computer Graphics and Vision*, 8((2-3)):85–283, 2014.

[31] M. Nikolova. Minimizers of cost-functions involving the nonsmooth data-fidelity terms. application to the processing of outliers. *SIAM Journal of Numerical Analysis*, 40(3):965–994, 2002.

[32] L. Oneto, S. Ridella, and D. Anguita. Tikhonov, Ivanov and Morozov regularization for support vector machine learning. *Machine Learning*, 103(1):103–136, 2016.

[33] B. Recht, M. Fazel, and P. A. Parrilo. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM Rev.*, 52(3):471–501, Aug. 2010.

[34] L. I. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D*, 60:259–268, 1992.

[35] M. D. Sacchi, T. J. Ulrych, and C. J. Walker. Interpolation and extrapolation using a high-resolution discrete fourier transform. *IEEE Transactions on Signal Processing*, 46(1):31–38, Jan 1998.

[36] I. W. Selesnick and İ. Bayram. Enhanced sparsity by non-separable regularization. *IEEE Transactions on Signal Processing*, 64(9):2298 – 2313, 2016.

[37] I. W. Selesnick and P.-Y. Chen. Group-sparse signal denoising: Non-convex regularization, convex optimization. *IEEE Transactions on Signal Processing*, 62(13):3464–3478, 2014.

[38] A. Tarantola. *Inverse problem theory: Methods for data fitting and model parameter estimation*.

[39] J. A. Tropp. Just relax: Convex programming methods for identifying sparse signals in noise. *IEEE Transactions on Information Theory*, 52(3):1030–1050, March 2006.

[40] P. Tseng. Convergence of a block coordinate descent method for nondifferentiable minimization. *Journal of optimization theory and applications*, 109(3):475–494, 2001.

[41] E. Van Den Berg and M. P. Friedlander. Probing the pareto frontier for basis pursuit solutions. *SIAM Journal on Scientific Computing*, 31(2):890–912, 2008.

[42] E. van den Berg and M. P. Friedlander. Probing the pareto frontier for basis pursuit solutions. *SIAM J. Sci. Comput.*, 31(2):890–912, Nov. 2008.

[43] E. Van den Berg and M. P. Friedlander. Sparse optimization with least-squares constraints. *SIAM Journal on Optimization*, 21(4):1201–1229, 2011.

[44] A. Yurtsever, M. Udell, J. A. Tropp, and V. Cevher. Sketchy Decisions: Convex Low-Rank Matrix Optimization with Optimal Storage. *ArXiv e-prints*, Feb. 2017.

[45] P. Zheng and A. Aravkin. Fast methods for nonsmooth nonconvex minimization. *ArXiv e-prints*, Feb. 2018.

[46] P. Zheng, T. Askham, S. L. Brunton, J. N. Kutz, and A. Y. Aravkin. A Unified Framework for Sparse Relaxed Regularized Regression: SR3. *ArXiv e-prints*, July 2018.