SIAM J. Sci. Comput.
Vol. 41, No. 1, pp. A246–A268

JOINT SPARSE RECOVERY BASED ON VARIANCES[*]

BEN ADCOCK[†], ANNE GELB[‡], GUOHUI SONG[§], AND YI SUI[†]

Abstract. Much research has recently been devoted to sparse signal recovery and image reconstruction from multiple measurement vectors. The assumption that the underlying signals or images have some common features with sparse representation suggests that using a joint sparsity approach to recover each individual signal or image can be more effective than recovering each signal or image separately using standard sparse recovery techniques. Joint sparsity reconstruction is typically performed using $\ell^{2,1}$-minimization, and although the process yields better results than separate recoveries, the inherent coupling makes the algorithm computationally intensive, since it is difficult to parallelize. In this investigation, we first observe that the elementwise variance of the signals convey information about their shared support. This observation motivates us to introduce a weighted ℓ^1-joint sparsity algorithm, where the weights depend on the calculated variance. Specifically, the ℓ^1-minimization should be more heavily penalized in regions where the corresponding variance is small, since it is likely there is no signal there. We demonstrate that our new method, which we refer to as variance-based joint sparse recovery, is more accurate and cost efficient. Applications in sparse signal recovery, parallel magnetic resonance imaging, and edge detection are considered.

Key words. sparse signal recovery, multiple measurement vectors, joint sparsity, variance

AMS subject classifications. 65F22, 65K10

DOI. 10.1137/17M1155983

1. Introduction.

Much research in recent years has been devoted to sparse signal and image recovery from multiple measurement vectors (MMV). While often the vectors are considered to be duplicates of the same data modulo noise or some other inconsistency, the MMV approach has been used to model various other data acquisitions, including parallel magnetic resonance imaging (MRI), hyperspectral imaging, and beyond. The assumption in all cases is that the underlying signals or images have some features that have sparse representations that can be readily obtained from each individual measurement vector. While they can be recovered separately using standard sparse recovery techniques, in this paper we consider the *joint sparsity approach* [2, 12, 10, 13]. In a nutshell, joint sparsity asserts that a set (or subset) of images or signals have similar support in a common sparsity domain (e.g., discrete gradient, wavelet transforms). Such information is exploited through additional constraints leading to a more robust recovery for *each* individual image or signal. Any subsequent joint processing of these individual recoveries is therefore greatly improved. Joint sparsity exploitation can be accomplished in different ways, typi-

[*]Submitted to the journal's Methods and Algorithms for Scientific Computing section November 8, 2017; accepted for publication (in revised form) November 19, 2018; published electronically January 10, 2019.
http://www.siam.org/journals/sisc/41-1/M115598.html

Funding: The first and fourth authors' work was supported in part by NSERC grant 611675 and an Alfred P. Sloan research fellowship. The second author's work was supported in part by grants NSF-DMS 1502640, NSF-DMS 1732434, AFOSR FA9550-18-1-0316, and AFOSR FA9550-15-1-0152. The third author's work was supported in part by grant NSF-DMS 1521661 and the Natural Science Foundation of China under grant 11701383. The fourth author received support from an NSERC PGSD scholarship.

[†]Department of Mathematics, Simon Fraser University, Burnaby, BC V5A 1S6, Canada (ben_adcock@sfu.ca, ysui@sfu.ca).

[‡]Department of Mathematics, Dartmouth College, Hanover, NH 03755 (annegelb@math.dartmouth.edu).

[§]Department of Mathematics, Clarkson University, Potsdam, NY 13699 (gsong@clarkson.edu).

cally via modifications of sparse recovery algorithms for single measurement vectors (see [3, 16, 26, 32, 33, 29] and references therein). Among the most popular is $\ell^{2,1}$-minimization [10, 12, 35, 13, 33]—a natural analogue of the popular ℓ^1-minimization procedure for sparse recovery of single vectors—which is widely used in practice. We consider this procedure as the benchmark technique for the remainder of this paper.

While often effective, a singular drawback of most joint sparse recovery techniques is that they are inherently coupled and therefore difficult to parallelize. For instance, $\ell^{2,1}$-minimization requires solving a convex optimization problem of size $N \times C$, where C is the number of signals and N is signal length (see section 2). When the number of signals is large, as is common in many applications, such approaches become increasingly slow.

With this in mind, in this paper we consider a different approach. This approach is based on the following observations, which will be further explained in section 2:

1. The supports of the signals to be recovered $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_C \in \mathbb{C}^N$ are similar.
2. The coefficients of signals are reasonably distinct.

We note that the first observation can be taken as the definition of joint sparsity (see section 2). Furthermore, in practice it may not hold in the signal domain itself but rather in a transform domain (e.g., wavelet or discrete gradient). This makes little difference to the current discussion. The second observation is essentially a necessary requirement to see benefits of joint sparsity. In the extreme case, if the signals are identical, then there is no additional information conveyed by the MMV (note that in this paper we consider the setting where the signals are measured with the same measurement operator).

These two observations suggest that the elementwise variance of the signals

$$v_i = \frac{1}{C}\sum_{c=1}^{C}(x_{ic})^2 - \left(\frac{1}{C}\sum_{c=1}^{C} x_{ic}\right)^2, \qquad i = 1, \ldots, N,$$

should convey information about their shared support. Specifically, the variance v_i will be large when the index i belongs to this support, and $v_i \approx 0$ otherwise. Hence, if the variance vector $\boldsymbol{v} = (v_i)_{i=1}^N$ is known or has been computed, rather than recovering each signal via ℓ^1-minimization (which gives equal weighting to each index) one should use this prior information to penalize indices unlikely to be in the support. This can be achieved, for instance, via a weighted ℓ^1 functional, that is, $\|\boldsymbol{x}\|_{1,\boldsymbol{w}} = \sum_{i=1}^N w_i |x_i|$ (see, for instance, [18]). The variance is used to compute the nonnegative weights in $[0, 1]$ and we choose $w_i \approx 0$ for large variance and conversely $w_i \approx 1$ when the variance is close to zero.

Our method, referred to as variance-based joint sparse (VBJS) recovery, proceeds in three steps:

(i) Compute approximations $\check{\boldsymbol{x}}_c$ to the signals $\boldsymbol{x}_c$ via ℓ^1-minimization.
(ii) Compute the elementwise variance of the $\check{\boldsymbol{x}}_c$.
(iii) Compute approximations $\hat{\boldsymbol{x}}_c$ to the signals $\boldsymbol{x}_c$ via weighted ℓ^1-minimization.

Unlike $\ell^{2,1}$-minimization, VBJS is easily parallelized, since the expensive computations in steps (i) and (iii) can be solved separately. Using standard black-box optimization solvers, our numerical results demonstrate a consistent reduction in computational time for VBJS over $\ell^{2,1}$-minimization. Furthermore, and perhaps surprisingly, the VBJS method often gives a recovery error superior to $\ell^{2,1}$-minimization. We document this observation on both synthetic experiments (i.e., phase transition curves) and numerical examples from imaging applications. Interestingly, on synthetic examples at least, we see the phase transition of VBJS approaches the optimal theoretical limit as the number of signals C increases.

The rest of our paper is organized as follows. In section 2 we describe our VBJS recovery algorithm. In section 3 we present a series of synthetic numerical experiments based on phase transitions and comparisons to existing methods. We illustrate in this section how our method yields both a greater probability of correct signal recovery than $\ell^{2,1}$-minimization and improved computational efficiency. In sections 4 and 5 we consider the use of VBJS in two applications—parallel MRI and edge detection from nonuniform Fourier data. In the former, we show how this new approach yields a consistent performance gain over existing techniques. In the latter application, only one set of data is acquired. However, it is processed in different ways, leading naturally to an MMV problem for which VBJS is shown to yield benefits. Theoretical discussion is provided in section 2.5, and we close with some concluding remarks in section 6.

2. Variance-based joint sparse recovery.

2.1. Problem setup.

Throughout this paper, we consider the recovery of $C \geq 1$ objects (e.g., signals or images) with certain common features. We consider a discrete setup, where these objects are represented as complex vectors $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_C \in \mathbb{C}^N$ (an extension of what follows to functions in Hilbert spaces can be developed along the lines of [1]). Note that these could be either the images or signals themselves or their coefficients in some orthogonal sparsifying transform. Whenever necessary we write

$$\boldsymbol{X} = [\boldsymbol{x}_1 | \cdots | \boldsymbol{x}_C] \in \mathbb{C}^{N \times C}$$

for the corresponding matrix to be recovered.

We assume the sensing protocol (e.g., an MR scanner) produces linear measurements of the vectors $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_C$. Specifically, we consider measurements of the form

$$\boldsymbol{y}_c = \boldsymbol{A}_c \boldsymbol{x}_c + \boldsymbol{n}_c, \qquad c = 1, \ldots, C, \tag{1}$$

where $\boldsymbol{n}_1, \ldots, \boldsymbol{n}_C$ are noise vectors and $\boldsymbol{A}_1, \ldots, \boldsymbol{A}_C \in \mathbb{C}^{m \times N}$ are *measurement matrices.* Often it will be the case that

$$\boldsymbol{A} = \boldsymbol{A}_1 = \cdots = \boldsymbol{A}_C,$$

although this condition is not necessary for the developments that follow. In this latter case, note that we may rewrite (1) as

$$\boldsymbol{Y} = \boldsymbol{A}\boldsymbol{X} + \boldsymbol{N},$$

where

$$\boldsymbol{Y} = [\boldsymbol{y}_1 | \cdots | \boldsymbol{y}_C] \in \mathbb{C}^{m \times C}, \qquad \boldsymbol{N} = [\boldsymbol{n}_1 | \cdots | \boldsymbol{n}_C] \in \mathbb{C}^{m \times C}.$$

With this in hand, the objective throughout the vector is to recover $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_C$ (or equivalently $\boldsymbol{X}$) from the measurements $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_C$ (or $\boldsymbol{Y}$). Our focus in this paper is on recovery of sparse objects, defined formally below. Hence, we usually consider the undetermined setting where m (the number of measurements) is much smaller than N (the signal dimension).

2.2. Further notation.

We now introduce several additional pieces of notation. We write $\|\cdot\|_p$ for the ℓ^p-norm on $\mathbb{C}^N$, where $1 \leq p \leq \infty$. As is conventional, we write $\|\cdot\|_0$ for the ℓ^0-"norm," i.e.,

$$\|\boldsymbol{x}\|_0 = |\text{supp}(\boldsymbol{x})| \, .$$

Here $\operatorname{supp}(\boldsymbol{x})$ is the support of $\boldsymbol{x} = (x_i)_{i=1}^N$, defined by

$$\operatorname{supp}(\boldsymbol{x}) = \{i : x_i \neq 0\}.$$

Given a vector $\boldsymbol{w} = (w_i)_{i=1}^N$ of positive weights, we define the weighted $\ell^1_{\boldsymbol{w}}$-norm as

$$\|\boldsymbol{x}\|_{1,\boldsymbol{w}} = \sum_{i=1}^N w_i |x_i|.$$

We note that it is also possible to define weighted $\ell^p_{\boldsymbol{w}}$-norms for $p \neq 1$, but this is not needed for the paper.

If $\boldsymbol{X} = (x_{ic})_{i,c=1}^{N,C} \in \mathbb{C}^{N\times C}$ is a matrix, we define the $\ell^{p,q}$-norms by

$$\|\boldsymbol{X}\|_{p,q} = \left(\sum_{i=1}^N \left(\sum_{c=1}^C |x_{ic}|^p\right)^{q/p}\right)^{1/q}.$$

In particular, if $p = q = 2$, then

$$\|\boldsymbol{X}\|_{2,2} = \|\boldsymbol{X}\|_F = \left(\sum_{i=1}^N \sum_{c=1}^C |x_{ic}|^2\right)^{1/2}$$

is just the Frobenius norm of $\boldsymbol{X}$. We will mainly focus on the case $p = 2$ and $q = 1$, leading to the $\ell^{2,1}$-norm:

$$\|\boldsymbol{X}\|_{2,1} = \sum_{i=1}^N \left(\sum_{c=1}^C |x_{ic}|^2\right)^{1/2}.$$

Finally, we define the $\ell^{2,0}$-"norm" as follows:

$$\|\boldsymbol{X}\|_{2,0} = \left|\left\{ i : \sum_{c=1}^C |x_{ic}|^2 \neq 0 \right\}\right|.$$

2.3. Sparsity and joint sparsity. This paper concerns the recovery of sparse vectors and joint sparse collections of vectors. We now define these terms.

DEFINITION 1. *A vector $\boldsymbol{x} \in \mathbb{C}^N$ is s-sparse for some $1 \leq s \leq N$ if*

$$\|\boldsymbol{x}\|_0 = |\operatorname{supp}(\boldsymbol{x})| \leq s.$$

A collections of vectors $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_C \in \mathbb{C}^N$ is s-joint sparse if

$$\|\boldsymbol{X}\|_{2,0} = \left|\bigcup_{c=1}^C \operatorname{supp}(\boldsymbol{x}_c)\right| \leq s,$$

where $\boldsymbol{X} = [\boldsymbol{x}_1|\cdots|\boldsymbol{x}_C]$.

Much as we refer to the support of $\operatorname{supp}(\boldsymbol{x})$ of a single sparse vector $\boldsymbol{x}$, we refer to $\bigcup_{c=1}^C \operatorname{supp}(\boldsymbol{x}_c)$ as the *joint support* of the vectors $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_C$.

A standard procedure for recovering a single sparse vector $\boldsymbol{x}$ from noisy measurements $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{n}$ is to solve the ℓ^1-minimization problem

$$\min_{\boldsymbol{z}\in\mathbb{C}^N} \|\boldsymbol{z}\|_1 \text{ subject to } \|\boldsymbol{A}\boldsymbol{z} - \boldsymbol{y}\|_2 \leq \eta.$$

The field known as compressed sensing gives conditions of the measurement matrix $\boldsymbol{A}$ under which this recovery is accurate and robust. See, for example, [11, 17].

Analogously, for a collection of C joint sparse vectors $\boldsymbol{X} = [\boldsymbol{x}_1|\dots|\boldsymbol{x}_C]$ a standard procedure for recovery from the noisy measurements $\boldsymbol{Y} = \boldsymbol{A}\boldsymbol{X} + \boldsymbol{N}$ involves solving the $\ell^{2,1}$-minimization problem

$$\min_{\boldsymbol{Z}\in\mathbb{C}^{N\times C}} \|\boldsymbol{Z}\|_{2,1} \text{ subject to } \|\boldsymbol{A}\boldsymbol{Z}-\boldsymbol{Y}\|_F \le \eta. \tag{2}$$

For related compressed sensing-type recovery results for this procedure, we refer to [4, 5, 14, 23].

2.4. The variance-based joint sparse recovery method. Let $\boldsymbol{x}_1,\dots,\boldsymbol{x}_C \in \mathbb{C}^N$ be the sequence of vectors to recover and

$$\boldsymbol{y}_c = \boldsymbol{A}_c\boldsymbol{x}_c + \boldsymbol{n}_c, \qquad c = 1,\dots,C,$$

the vector of measurements. We shall assume the a priori noise bound

$$\|\boldsymbol{n}_c\|_2 \le \eta_c, \qquad c=1,\dots,C,$$

for known noise levels $\eta_1,\dots,\eta_C$.

We now make the following assumptions:

1. The supports of the vectors are similar, i.e., $\operatorname{supp}(\boldsymbol{x}_1) \approx \operatorname{supp}(\boldsymbol{x}_2) \approx \cdots \approx \operatorname{supp}(\boldsymbol{x}_C)$. Equivalently, the joint sparsity of $\boldsymbol{x}_1,\dots,\boldsymbol{x}_C$ does not greatly exceed the sparsity of each of the individual vectors.
2. The coefficients of the vectors are reasonably distinct. Specifically, the vector $\boldsymbol{v} = (v_i)_{i=1}^N$ of elementwise variances
$$v_i = \frac{1}{C}\sum_{c=1}^{C}(x_{ic})^2 - \left(\frac{1}{C}\sum_{c=1}^{C}x_{ic}\right)^2, \qquad i=1,\dots,N,$$
is nonzero with $\operatorname{supp}(\boldsymbol{v}) \approx \bigcup_{c=1}^{C}\operatorname{supp}(\boldsymbol{x}_c)$.

Both assumptions are reasonable in practice. We note in passing that VBJS will not fail per se when either of these assumptions does not hold. Rather, it will just not convey a substantial benefit over individual recovery of the $\boldsymbol{x}_c$. In fact, without both of these assumptions no joint sparse recovery technique can expect to achieve a significant performance gain over individual recovery of the vectors $\boldsymbol{x}_c$.

The VBJS method is now described in Algorithm 1. It first exploits the individual sparsity of the vectors $\boldsymbol{x}_c$ by solving C separate ℓ^1-minimization problems (step 1). This gives approximations $\check{\boldsymbol{x}}_c \approx \boldsymbol{x}_c$ from which the elementwise variance $\boldsymbol{v}$ can be estimated (step 2). Finally, this prior information is incorporated into a weighting vector $\boldsymbol{w}$ (step 3) which is used in solving C weighted ℓ^1-minimization problems to get the final approximations $\hat{\boldsymbol{x}}_c \approx \boldsymbol{x}_c$ (step 4).

Remark 2. Observe that Algorithm 1 is easily parallelizable, since the computationally intensive steps (steps 1 and 4) each require the solution of C separate (weighted) ℓ^1-minimization problems. Steps 2 and 3 require communications between cores but are extremely cheap in comparison.

Remark 3. In order to implement Algorithm 1, one needs to specify the weights $\boldsymbol{w}$. As noted, a weight w_i should be small at an index i which is expected to be in the support and large otherwise. Our primary choice of weights will be

$$w_i = \frac{1}{v_i+\epsilon}, \qquad i=1,\dots,N, \tag{3}$$

Algorithm 1. VBJS recovery.

```
1: Recover the vectors $\boldsymbol{x}_c$, $c = 1, \dots, C$, separately using standard $\ell^1$-minimization:
       $\check{\boldsymbol{x}}_c \in \underset{\boldsymbol{z} \in \mathbb{C}^N}{\operatorname{argmin}} \|\boldsymbol{z}\|_1$ subject to $\|\boldsymbol{A}_c \boldsymbol{z} - \boldsymbol{y}_c\|_2 \leq \eta_c, \qquad c = 1, \dots, C.$
2: Compute the elementwise variance of the vectors $\check{\boldsymbol{x}}_c = (\check{x}_{ic})_{i=1}^N$, $c = 1, \dots, C$, solved in the first step. That is, compute $\boldsymbol{v} = (\check{v}_i)_{i=1}^N$, where
       $v_i = \frac{1}{C} \sum_{c=1}^{C} (\check{x}_{ic})^2 - \left( \frac{1}{C} \sum_{c=1}^{C} \check{x}_{ic} \right)^2, \qquad i = 1, \dots, N.$
3: The two assumptions made above suggest that $\boldsymbol{v}$ should carry information about the shared support of the $\boldsymbol{x}_c$. Specifically, $\check{v}_i$ should be large when the index $i$ belongs to this support, and $\check{v}_i \approx 0$ otherwise. Hence we compute a vector of nonnegative weights $\boldsymbol{w} = (w_i)_{i=1}^N$ based on this information, where $0 \leq w_i \leq 1$. In particular, we choose $w_i \approx 0$ when $\check{v}_i$ is large and $w_i \approx 1$ when $\check{v}_i \approx 0$. While $\boldsymbol{w}$ may be tuned for a particular application (see, for example, section 5), we use (3) to demonstrate the general use of the VBJS recovery method.
4: Solve $C$ weighted $\ell^1$-minimization problems to get the final reconstruction of each vector $\boldsymbol{x}_c$:
       $\hat{\boldsymbol{x}}_c \in \underset{\boldsymbol{z} \in \mathbb{C}^N}{\operatorname{argmin}} \|\boldsymbol{z}\|_{1,\boldsymbol{w}}$ subject to $\|\boldsymbol{A}_c \boldsymbol{z} - \boldsymbol{y}_c\|_2 \leq \eta_c, \qquad c = 1, \dots, C.$
```

a strategy inspired by [6]. In general the weights should adapt easily to differing scales in the variance vector $\boldsymbol{v}$. We shall see in section 3 that using (3) makes the VBJS reasonably invariant to the specific choice of ϵ, so that careful parameter tuning is not required.

2.5. Theoretical discussion. A full theoretical analysis of VBJS and comparison is outside the scope of this paper. However, let us now make several remarks. First, existing theoretical results on so-called compressed sensing with partial support information (see [18] and references therein) have demonstrated the theoretical benefits of recovering a single vector $\boldsymbol{x}$ from measurements $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}$ using weighted ℓ^1-minimization. Specifically, when a sufficiently good prior on its support is known, the number of measurements required for weighted ℓ^1-minimization with a suitable choice of weights no longer exhibits a logarithmic growth in the ambient dimension N, as is the case for unweighted ℓ^1-minimization [27]. Note that in the setup of VBJS, the existence of a good support estimate depends on (i) how well the variance is estimated in step 1 of Algorithm 1 and (ii) how well the variance predicts the true support. The latter is intrinsic to the signals being recovered—at the extreme end, if the signals all happened to be identical, then $\boldsymbol{v} = 0$, thus giving no information.

Second, we note there are a number of recovery guarantees for $\ell^{2,1}$-minimization [5, 4, 14]. In order to enforce that the signals are sufficiently distinct, thus avoiding the barrier discussed above when the signals are all equal, results of this type typically assume a random model for the nonzero coefficients. With this assumption, the recovery guarantee can be shown to improve, and in particular, the failure probability decreases exponentially with the number of signals C.

We expect a theoretical analysis of VBJS is possible (especially in the random setting discussed previously), combining elements of [27] and [14]. This is left as future work.

3. Numerical experiments.

We now present several synthetic experiments to illustrate the effectiveness of Algorithm 1. For comparison purposes, we also consider (i) separate recovery of the C signals via ℓ^1-minimization (equivalent to step 1 of Algorithm 1) and (ii) joint sparse recovery via $\ell^{2,1}$-minimization (as in (2)). Our comparison is done by analyzing the performance of each algorithm on randomly generated sets of sparse vectors of a given size N. Specifically, for each fixed m (number of measurements) and s (sparsity), we proceed as follows:

1. Fix a number of trials T. For each trial $t = 1, \dots, T$,
   - (i) generate a support set $S \subseteq \{1, \dots, N\}$ uniformly at random with size $|S| = s$;
   - (ii) define vectors $\boldsymbol{x}_1, \dots, \boldsymbol{x}_C$ such that $\operatorname{supp}(\boldsymbol{x}_1) = \cdots = \operatorname{supp}(\boldsymbol{x}_C) = S$; the nonzero entries x_{ic}, $c = 1, \dots, C$, $i \in S$, are drawn from the standard normal distribution;
   - (iii) generate a measurement matrix $\boldsymbol{A}$ and compute measurements $\boldsymbol{y}_c = \boldsymbol{A}\boldsymbol{x}_c$, $c = 1, \dots, C$;
   - (iv) compute the reconstructions $\hat{\boldsymbol{x}}_1, \dots, \hat{\boldsymbol{x}}_C$ using the desired algorithm (ℓ^1-minimization, VBJS or $\ell^{2,1}$-minimization);
   - (v) compute the normalized error $E_t = \sqrt{\sum_{c=1}^C \|\boldsymbol{x}_c - \hat{\boldsymbol{x}}_c\|_2^2 / \sum_{c=1}^C \|\boldsymbol{x}_c\|_2^2}$ for each technique.

   Finally, average the error E_t over the trials, $E = \frac{1}{T}(E_1 + \cdots + E_T)$.
2. Repeat step 1 for different values of s and m as required.

Note that it is possible to compute other quantities of interest, such as the computational time, in a similar manner. We may also compute the empirical success probability p, defined as the fraction of trials which successfully recover the vectors $\boldsymbol{x}_1, \dots, \boldsymbol{x}_C$ to within a given tolerance, i.e., $E_t < \mathrm{tol}$ for some fixed tolerance tol.

There are various options for generating the measurement matrix in step (iii). Since it frequently arises in applications, we choose $\boldsymbol{A}$ to be a subsampled discrete Fourier transform (DFT). That is, we construct a set $\Omega \subseteq \{1, \dots, N\}$ of size m uniformly at random and let

$$A = \frac{1}{\sqrt{m}} P_\Omega F,$$

where $F \in \mathbb{C}^{N \times N}$ is the DFT matrix and $P_\Omega \in \mathbb{C}^{m \times N}$ is the matrix that selects rows of F corresponding to the indices in Ω. The factor $\frac{1}{\sqrt{m}}$ is a normalization constant and ensures that $\mathbb{E}(A^* A) = I$. The above procedure also requires a number of parameters. Throughout, we shall choose these as $N = 256$, $T = 20$, and $\mathrm{tol} = 10^{-3}$, which is consistent with similar experiments performed in, for example, [30].

We also require a numerical solver for all three of the optimization problems considered: ℓ^1-minimization, weighted ℓ^1-minimization, and $\ell^{2,1}$-minimization. Unless otherwise stated, we use the SPGL1 package [34] with its default parameter values, except for the maximum number of iterations, which is set to 10,000. Since the data in this experiment is noiseless, we solve equality-constrained minimization problems (i.e., $\eta = 0$ or $\eta_c = 0$, respectively).

3.1. The weighting parameter $\boldsymbol{\epsilon}$.

Figure 1 plots the recovery error and success probability versus m for a fixed sparsity s using VBJS with various different values

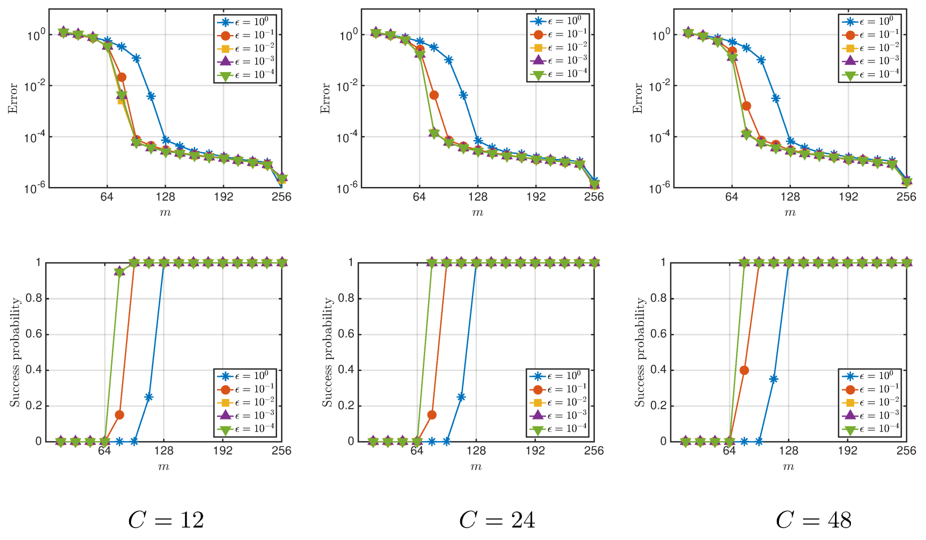

FIG. 1. *Recovery error (top row) and success probability (bottom row) against m for VBJS on randomly generated sparse vectors with sparsity $s = 64$. The weights* (3) *were used with various values of ϵ.*

of the weighting parameter ϵ in (3). The usual phase transition behavior is observed in the success probability as the number of measurements passes through a certain threshold. As expected, larger values of ϵ yield worse phase transitions, since the prior information obtained from the variance vector is less heavily exploited. However, we observe that decreasing ϵ beyond 10^{-2} does not improve the results. This is also to be expected, as the computed variance vector is only an approximation to the true variance, and thus too aggressive a weighting strategy (i.e., smaller ϵ) is more prone to this numerical error. With this in mind, for the remainder of this paper, when using (3) as a weighting strategy we choose $\epsilon = 10^{-2}$.

3.2. Comparison with other methods and solvers. We now compare VBJS with three other methods:

1. separate recovery of the individual signals via ℓ^1-minimization (i.e., step 1 of Algorithm 1),
2. two-step reweighted ℓ^1-minimization of the individual signals (see below),
3. joint sparse recovery via $\ell^{2,1}$-minimization (as in (2)).

Note that method 2 is similar to Algorithm 1, except that the weights do not use any joint information. Instead, following the reweighted ℓ^1-minimization procedure [6], for each signal, the weights $\boldsymbol{w} = \boldsymbol{w}_c = (w_{ic})_{i=1}^N$ are chosen according to

$$w_{ic} = \frac{1}{|\check{x}_{ic}| + \epsilon}.$$

For accurate approximations of $|\check{x}_{ic}|$, reweighted ℓ^1-minimization achieves better performance over ℓ^1-minimization for recovery of a single vector. Note that we limit ourselves to two steps for consistency with the VBJS algorithm.

Figure 2 reports the recovery error, success probability, and average computational time versus m for a fixed sparsity s. Unsurprisingly, ℓ^1-minimization has the lowest computational time, since it requires just C ℓ^1-minimization solves of size N which are done in parallel, followed by VBJS ($2C$ solves of size N done in parallel)

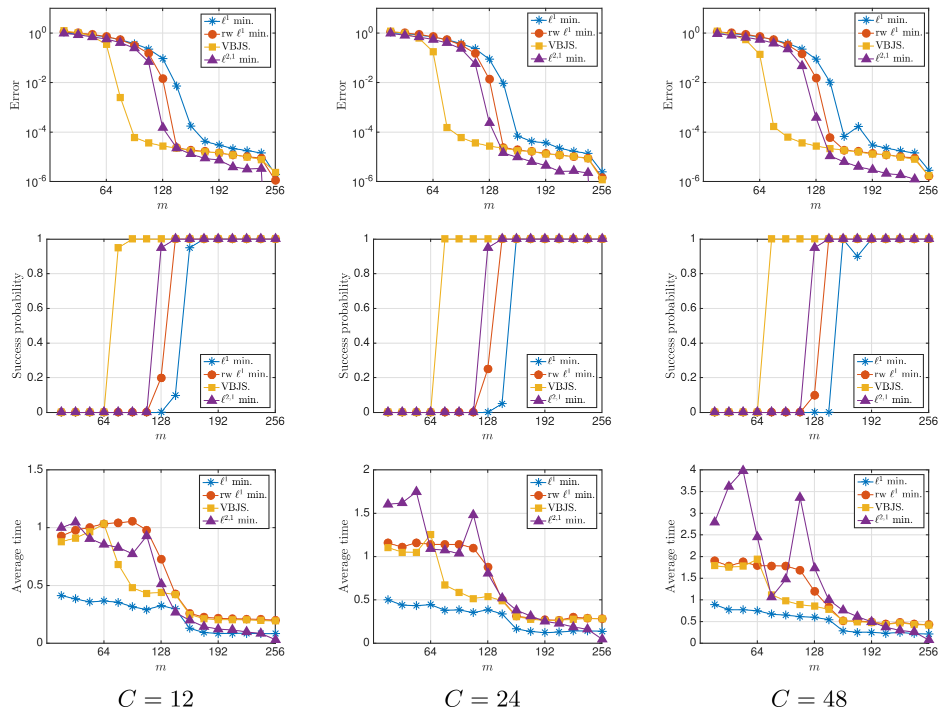

FIG. 2. *Comparison of ℓ^1-minimization, two-step reweighted (rw) ℓ^1-minimization, VBJS, and $\ell^{2,1}$-minimization for sparsity $s = 64$. The rows show the error (top), success probability (middle), and average time (bottom) versus m for each method. For this and the results shown in Tables 1 and 2 computations were performed on a cluster with 48 physical cores (96 logical cores), Intel Xeon $E5-4657L$ v2 processors, 2.90GHz, and 512GB of RAM memory.*

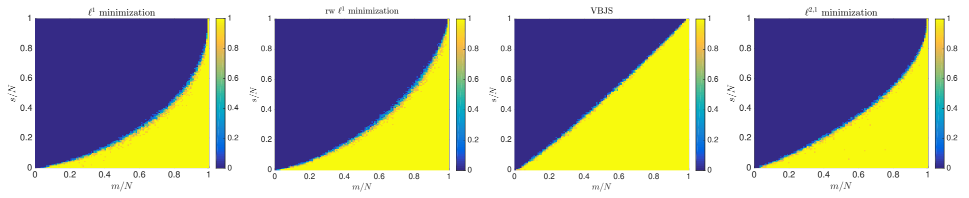

FIG. 3. *Phase transition diagrams for ℓ^1-minimization, two-step reweighted ℓ^1-minimization, VBJS, and $\ell^{2,1}$-minimization for $C = 12$ signals using $T = 10$ trials. The diagrams show the success probability for values $1 \leq s \leq N$ and $1 \leq m \leq N$.*

and then $\ell^{2,1}$-minimization (one solve of size NC). VBJS *also* achieves the best phase transition. For instance, with $C = 24$ signals, successful recovery requires only around 80 measurements per signal, in comparison to around 128 for $\ell^{2,1}$-minimization. In other words, the $\ell^{2,1}$-minimization requires over 50% additional measurements to recover the same signals. This is further illustrated in Figures 3 and 4, which show the full phase transition plots and phase transition curves, respectively, for each method. Interestingly, VBJS exhibits a phase transition curve which is close to the optimal $m = s$ line.

Up to this point we have used the SPGL1 package to solve the various optimization problems. For completeness, in Figure 5 we repeat the results of Figure 2 using the

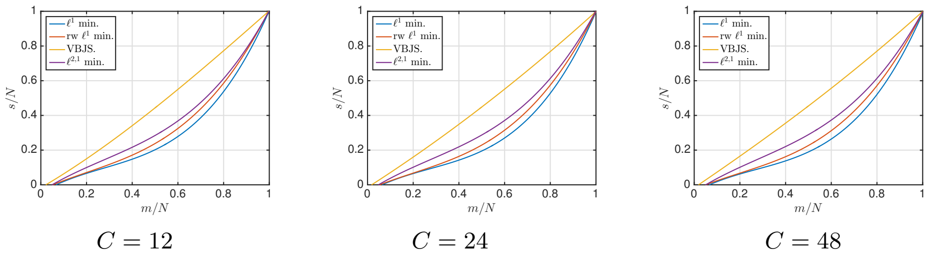

FIG. 4. *Phase transition curves for ℓ^1-minimization, two-step reweighted ℓ^1-minimization, VBJS, and $\ell^{2,1}$-minimization using $T = 10$ trials. The curves show the phase transition from successful recovery (below the line) to unsuccessful recovery (above the line). The criterion for successful recovery used was an empirical success probability $p > 0.75$.*

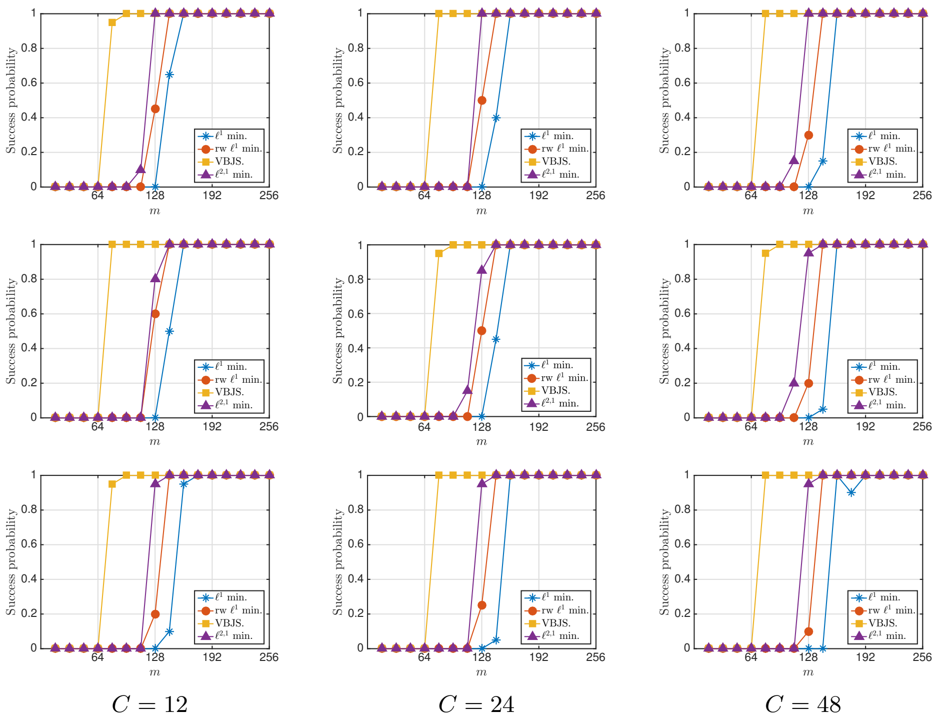

FIG. 5. *Comparison of ℓ^1-minimization, two-step reweighted ℓ^1-minimization, VBJS, and $\ell^{2,1}$-minimization for sparsity $s = 64$ using CVX (top row), YALL1 (middle row), and SPGL1 (bottom row). The plots show the success probability versus m for each method and package.*

YALL1 [36] and CVX [22] packages. Similar results are produced by these packages, with VBJS giving a consistently better phase transition than ℓ^1-minimization or $\ell^{2,1}$-minimization in all cases.

3.3. Signals with partially overlapping supports. Up to this point, the numerical experiments have considered signals $\boldsymbol{x}_c$ that are s-sparse and have a common support S. In practice, it may be more realistic to assume that only a fraction of the support is shared. We next present several experiments for this scenario.

To model this setup, we introduce a parameter $0 < \tau \leq 1$ corresponding to the fraction of shared supports. We then replace steps (i) and (ii) in the previous experiment with the following:

(i′) Generate a support set $S \subseteq \{1, \ldots, N\}$ uniformly at random with size $|S| = \lfloor \tau s \rfloor$. Generate supports sets $S_1, \ldots, S_c \subseteq \{1, \ldots, N\} \backslash S$ uniformly and independently at random with size $|S_c| = s - \lfloor \tau s \rfloor$.

(ii′) Define vectors $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_C$ such that $\text{supp}(\boldsymbol{x}_c) = S \cup S_c$, $c = 1, \ldots, C$. The nonzero entries x_{ic}, $c = 1, \ldots, C$, $i \in S$, are drawn from the standard normal distribution.

In other words, each signal has roughly τs elements in the common support S and $(1-\tau)s$ elements in a unique support S_c. Note that $\tau = 1$ corresponds to the previous experiment.

The top row of Figure 6 show the success probability for VBJS and various different values of τ with weights as in (3). As is evident, the performance of VBJS quickly declines as τ increases. However, this is due to the choice of weights, which, while well suited to the $\tau = 1$ case, produce incorrectly scaled weights to effectively handle the partially shared support case. Fortunately, a different weighting strategy, provided in Algorithm 2, can overcome this issue.

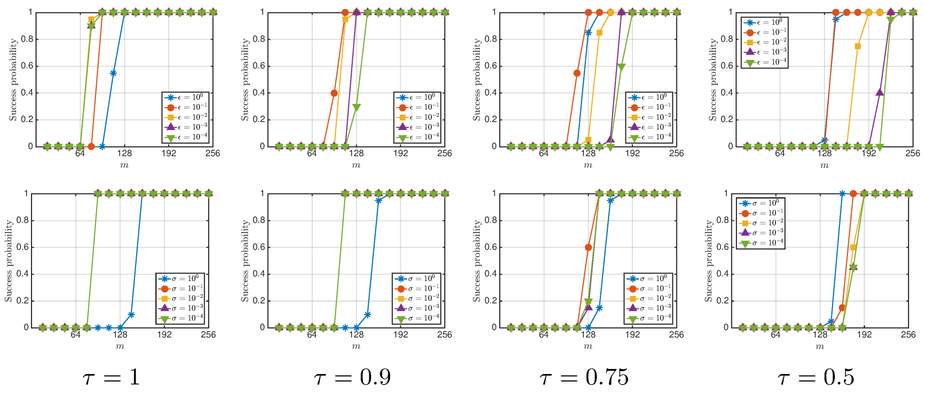

FIG. 6. *Comparison of the success probability for two weighting strategies with various* τ. *Results for weights as in* (3) *are shown at the top, and the energy weights with* $\delta = 0.05$ *are shown at the bottom.*

Algorithm 2. Weighting strategy for signals with partially overlapping supports.

```
1: Choose $\delta \in (0,1)$ and $\sigma \in (0,1)$.
2: Sort the variance vector $\boldsymbol{v}$ into decreasing order to form a new vector $\boldsymbol{v}^*$. Let $I$
   be the index set of sorted indices, i.e., $\boldsymbol{v}_i^* = \boldsymbol{v}_{I(i)}$.
3: Determine the smallest $K \leq N$ such that
       $\sum_{k=1}^{K} \boldsymbol{v}_k^* \geq (1-\delta) \sum_{n=1}^{N} v_n.$
4: Define the set $T := \{I(1), \ldots, I(K)\}$.
5: Choose weights $w_i = \sigma$ if $i \in T$ and $w_i = 1$ if $i \notin T$.
```

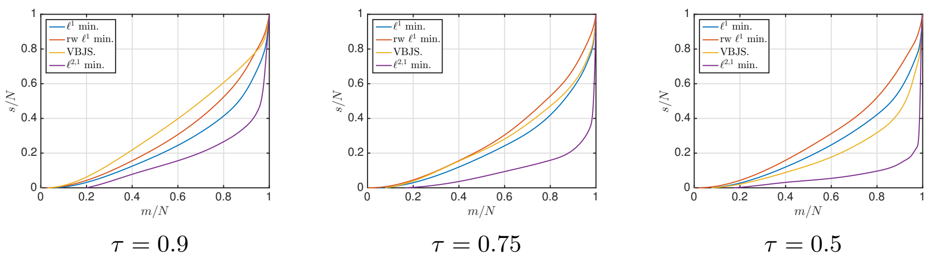

FIG. 7. *Phase transition curves for ℓ^1-minimization, two-step reweighted ℓ^1-minimization, VBJS, and $\ell^{2,1}$-minimization with $\sigma = 10^{-2}$ for various τ.*

Unlike the previous weights, Algorithm 2 constructs a candidate support set T that capture all but δ of the norm of the variance vector and then uses a binary weighting strategy based on whether an index i is inside or outside T. Note that a similar approach has been used in, for instance, [18]. This strategy requires choosing two parameters δ and σ. However, as shown in Figure 6, the performance is fairly insensitive to the choice of σ (further results, not shown, indicate similar behavior with respect to the other parameter δ). Henceforth we use the values $\delta = 0.05$ and $\sigma = 10^{-2}$. Fianlly we note that it is still possible for T to be nonunique, for example, if $\boldsymbol{v}_i = \boldsymbol{v}_j$ for some $i, j \in [1, N]$. This is generally unlikely to happen and most often should affect only cases where there is very small overlap of support (i.e., τ is small).

Figure 7 compares this approach with the other three methods for various different τ. For larger τ, VBJS offers the best performance. On the other hand, as τ decreases its performance in comparison to ℓ^1-minimization and the two-step reweighted ℓ^1-minimization declines. This is to be expected: as the fraction of shared support decreases, there is less benefit to promoting joint structure. Interestingly, $\ell^{2,1}$-minimization offers very poor performance, even when τ is close to one.

4. Application to parallel magnetic resonance imaging.

In the final two sections of this paper, we consider the use of VBJS in several imaging applications. First, we consider parallel MRI. This is an example of a multisensor acquisition system [7] in which multiple coils simultaneously acquire measurements of the image to be recovered. A standard discrete model for parallel MRI is as follows (see, for example, [8, 24]). Let $\boldsymbol{x} \in \mathbb{C}^N$ be the (vectorized) image to recover, $F \in \mathbb{C}^{N\times N}$ be the DFT matrix, and C be the number of coils. In the cth coil the measurements acquired a given by

$$\boldsymbol{y}_c = P_\Omega F G_c \boldsymbol{x} + \boldsymbol{n}_c \in \mathbb{C}^m,$$

where $\boldsymbol{n}_c$ is noise, and P_Ω is a sampling map that selects rows of F corresponding to the frequencies Ω that are sampled. The matrix $G_c = \operatorname{diag}(\boldsymbol{g}_c) \in \mathbb{C}^{N\times N}$ is a diagonal matrix, intrinsic to the particular coil, and the vector $\boldsymbol{g}_c \in \mathbb{C}^N$ is the so-called sensitivity profile. We define the coil images as

$$\boldsymbol{x}_c = G_c \boldsymbol{x},$$

i.e., the overall image $\boldsymbol{x}$ multiplied by the sensitivity profile matrix G_c, so that the measurements in the cth coil can be interpreted as Fourier measurements $\boldsymbol{y}_c = P_\Omega F \boldsymbol{x}_c + \boldsymbol{n}_c$ of the coil image $\boldsymbol{x}_c$. Figure 8 shows typical sensitivity profiles and coil images.

Many techniques have been developed for sparse parallel MRI reconstruction. See, for example, [8] and references therein. Here we focus primarily on the class of methods

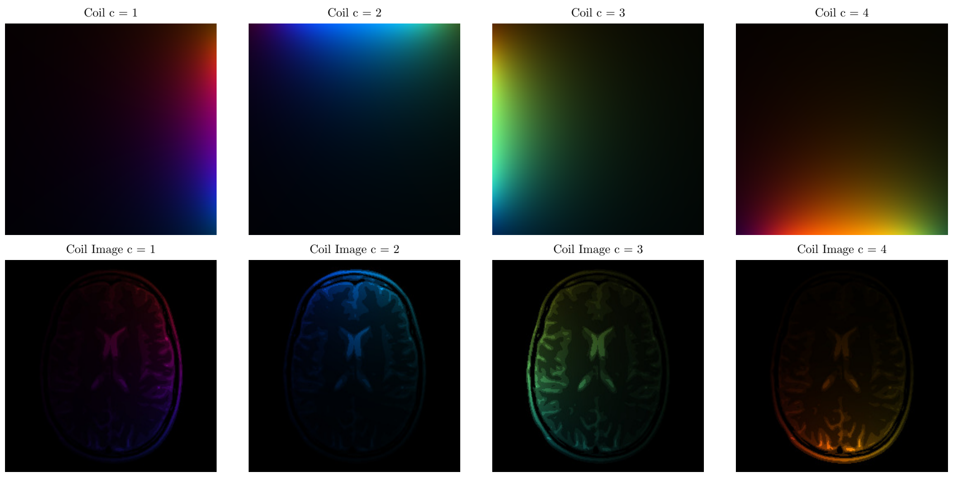

FIG. 8. *Complex sensitivity profiles (top) and coil images (bottom). The phase is color coded.*

known as *coil-by-coil techniques*. In these techniques, one first computes approximate coil images $\hat{\boldsymbol{x}}_1, \ldots, \hat{\boldsymbol{x}}_C$ and then combines them to compute an approximation $\hat{\boldsymbol{x}}$ to the overall image $\boldsymbol{x}$. The first stage can be performed using $\ell^{2,1}$-minimization, a technique first introduced in the parallel MRI context in [25] and known as calibration-less multi-coil (CaLM) MR reconstruction. We shall compare this with the VBJS procedure introduced in this paper.

An advantage of coil-by-coil recovery techniques is that they can avoid calibration of the sensitivity profiles G_c. Typically, this calibration step requires an additional prescan, which can be time-consuming. To avoid this, the second stage (recovery of $\boldsymbol{x}$ from the recovered coil images) is performed using a sum-of-squares procedure:

$$\hat{x}_i = \sqrt{\sum_{c=1}^{C} |\hat{x}_{ic}|^2}, \qquad i = 1, \ldots, N. \tag{4}$$

Unfortunately, this procedure can introduce additional inhomogeneity artifacts to the reconstruction for typical coil geometries. Since our main objective in this section is to compare methods for recovering the coil images, we shall avoid the sum-of-squares procedure and instead use the least-squares fit

$$\hat{\boldsymbol{x}} = \underset{\boldsymbol{z} \in \mathbb{C}^N}{\operatorname{argmin}} \sum_{c=1}^{C} \|G_c \boldsymbol{z} - \hat{\boldsymbol{x}}_c\|_2^2. \tag{5}$$

Note that since matrices G_c are diagonal, this can be conveniently expressed as

$$\hat{x}_i = \frac{\sum_{c=1}^{C} \hat{x}_{ic} \overline{g_{ic}}}{\sum_{c=1}^{C} |g_{ic}|^2}, \qquad i = 1, \ldots, N,$$

where $\boldsymbol{g}_c = (g_{ic})_{i=1}^N$. This avoids the inhomogeneity artifacts in the former procedure, at the expense of having to know (or precompute) the sensitivity profiles.

Remark 4. We refer readers to [8] for other types of coil-by-coil methods. We do not use them for comparison purposes here since they either (i) use a fundamentally

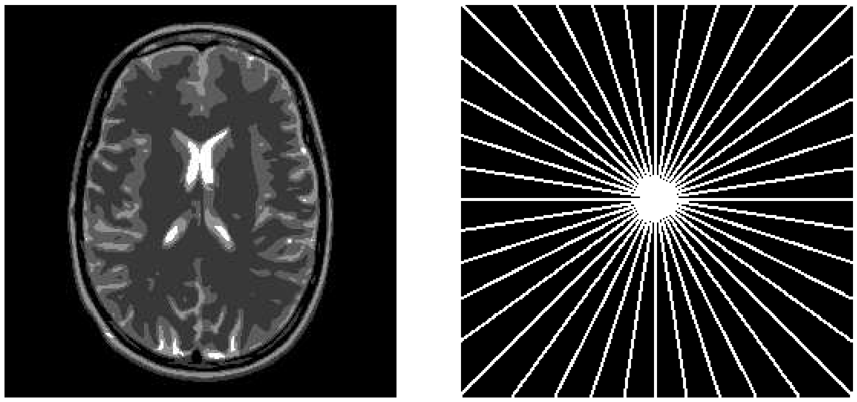

FIG. 9. 256×256 *phantom image (left) and radial sampling map (right).*

different approach to recover the coil images (e.g., low-rank matrix completion rather than $\ell^{2,1}$-minimization) or (ii) consider on-the-fly calibration procedures to estimate the sensitivity profiles at the same time as recovering the coil images. Hence, none of these procedures (including CaLM) is as easily parallelized as VBJS introduced in Algorithm 1, which is the main objective of the current comparison.

In the following experiment, we compare the recovery of the analytical phantom image shown in Figure 9 (due to [24]) using $\ell^{2,1}$-minimization and VBJS, respectively, followed in both cases by the least-squares fit (5). We use the same data for each method, taken as radial line sampling in Fourier space (see Figure 9). This is a typical sampling procedure for parallel MRI reconstruction. The number of lines is varied from 37 (corresponding to 29.7% sampling of k-space) to 93 (corresponding to 64.6% sampling). As is standard in sparse MRI reconstruction, DB4 wavelets are used in both cases as the sparsifying transform. The sensitivity profiles are generated using the Biot–Savart law, as described in [24]. Gaussian random noise with variance 10^{-3} was added to the data. The same value was used for the regularization parameter η in the various optimization problems. For the VBJS method we use the weights (3), which we have found to give slightly better reconstruction error than the weights introduced in section 3.3.

Tables 1 and 2 show the signal-to-error ratio (SER) and computational time for both procedures. In the same manner as the experiments in the previous section, VBJS both requires less time to compute the reconstruction and achieves a consistently higher SER. In particular, for large numbers of coils and radial lines, the time saving is by a factor of between 2 and 4.

5. Application to edge detection.

We now use the VBJS approach to detect edges from (nonuniform) Fourier measurements. Detecting edges from Fourier data is a well-studied problem (see, e.g., [15, 19, 20, 21, 28]) and is often used in conjunction with segmentation, feature identification, and extraction, or other postprocessing operations. It is inherently difficult since Fourier data are global measurements while edges are local features. For ease of notation we use one-dimensional functions to describe the process. Later we extend the results to two dimensions.

Let f be a piecewise continuous function supported on $[-1, 1]$. We define the corresponding *jump function* as

Table 1
SER $-20\log_{10}\left(\|\boldsymbol{x}-\hat{\boldsymbol{x}}\|_2/\|\boldsymbol{x}\|_2\right)$ *in dB for each method, where* $\hat{\boldsymbol{x}}$ *is the recovered image and* C *is the number of coils.*

$C = 8$

No. of lines	37	45	53	61	69	77	85	93
VBJS SER	21.32	23.54	25.95	28.48	30.93	33.70	36.58	39.32
$\ell^{2,1}$ min SER	20.77	22.93	25.08	27.43	29.66	32.32	35.08	37.73

$C = 16$

No. of lines	37	45	53	61	69	77	85	93
VBJS SER	21.20	23.41	25.73	28.19	30.59	33.31	36.14	38.81
$\ell^{2,1}$ min SER	20.76	22.94	25.08	27.40	29.69	32.38	35.09	37.55

$C = 32$

No. of lines	37	45	53	61	69	77	85	93
VBJS SER	21.20	23.41	25.73	28.19	30.58	33.29	36.16	38.81
$\ell^{2,1}$ min SER	20.75	22.87	25.06	27.45	29.67	32.32	35.09	37.5963

Table 2
Computational time (in seconds) for each method, where C *is the number of coils.*

$C = 8$

No. of lines	37	45	53	61	69	77	85	93
VBJS time	113.10	55.96	48.20	42.44	39.87	28.95	32.92	25.64
$\ell^{2,1}$ min time	70.81	83.03	71.38	81.20	34.91	49.47	42.25	39.17

$C = 16$

No. of lines	37	45	53	61	69	77	85	93
VBJS time	56.59	40.79	34.98	35.19	30.53	27.57	27.86	21.38
$\ell^{2,1}$ min time	67.87	142.70	81.64	68.70	82.22	78.63	51.58	60.94

$C = 32$

No. of lines	37	45	53	61	69	77	85	93
VBJS time	67.35	65.32	49.67	46.80	36.21	27.58	29.95	25.92
$\ell^{2,1}$ min time	101.89	103.95	103.99	154.28	83.43	100.08	110.93	77.3205

$$[f](x) = f(x^+) - f(x^-), \quad x \in [-1,1].$$

If f has a finite number of edges (jump discontinuities) given at locations $\xi_1, \xi_2, \ldots, \xi_K$, we can write

$$[f](x) = \sum_{k=1}^{K} [f](\xi_k)\chi_{\xi_k}(x), \quad x \in [-1,1],$$

where the indicator function $\chi_{\xi_j}(x)$ is 1 if $x = \xi_j$ and 0 otherwise. Suppose we are given the Fourier measurements

$$\hat{f}(\lambda_j) = \langle f, \varphi_j \rangle = \int_{-1}^{1} f(x) \mathrm{e}^{-\pi i x \lambda_j} dx, \quad -m \le j \le m, \tag{6}$$

where $\varphi_j(x) = \mathrm{e}^{\pi i \lambda_j x}$ and $\{\lambda_j : -m \le j \le m\}$ are nonuniformly sampled frequencies in $\mathbb{R}$. For a given set of grid points $\boldsymbol{x} = (x_l : 1 \le l \le N)$ in $[-1,1]$, we wish to obtain

Algorithm 3. VBJS method for edge detection from Fourier data.

```
1: Compute a set of edge detection vectors $\boldsymbol{y}_c$, $1 \leq c \leq C$ from the Fourier measurements in (6).
2: Compute the elementwise variance $\check{\boldsymbol{v}} = (\check{v}_i)_{i=1}^N$ of the vectors $\boldsymbol{y}_c = (y_{ic})_{i=1}^N$ as
       $$\check{v}_i = \frac{1}{C}\sum_{c=1}^{C}(y_{ic})^2 - \left(\frac{1}{C}\sum_{c=1}^{C} y_{ic}\right)^2, \qquad i = 1, \dots, N.$$
3: Determine the weight vector $\boldsymbol{w}$ using (10).
4: Solve $C$ weighted $\ell^1$-minimization problems to get the refined approximation:
       $$\hat{\boldsymbol{x}}_c \in \underset{\boldsymbol{z}\in\mathbb{C}^N}{\operatorname{argmin}} \|\boldsymbol{z}\|_{1,\boldsymbol{w}} \text{ subject to } \|\boldsymbol{z} - \boldsymbol{y}_c\|_2 \leq \eta_c, \qquad c = 1, \dots, C.$$
5: Combine all refined approximations $\hat{\boldsymbol{x}}_c$ to obtain the final edge vector approximation. For example,
       $$\mathbf{y} = \frac{1}{C}\sum_{c=1}^{C}\hat{\boldsymbol{x}}_c.$$
```

a corresponding vector $\boldsymbol{y} \in \mathbb{R}^N$ of edges. More realistically, as will be explained in more detail below, $\mathbf{y}$ has "sharp peaks" around the edges, that is, $|y_l|$ is large when x_l is very near an edge and small if it is not.

Let $\boldsymbol{y}_c$, $c = 1, \dots, C$, be a set of edge approximation vectors for a piecewise smooth function $f(x)$. That is, each $\boldsymbol{y}_c$ approximates $[f](x)$ at a set of grid points x_i, $i = 1, \dots, N$. The VBJS method can be applied to this set of vectors to obtain a more accurate edge detection approximation. The measurement matrix in (1) is given by $\boldsymbol{A}_c = \boldsymbol{I}$, the identity matrix, for $1 \leq c \leq C$. The specific use of Algorithm 1 for detecting edges from the given Fourier data in (6) is provided in Algorithm 3.

Before implementing Algorithm 3, we briefly describe the concentration factor edge detection method [21, 20], which will be used to construct each measurement vector of edges, $\mathbf{y}_c, c = 1, \dots, C$. To simplify the description, we assume that f has only one jump discontinuity in $(-1, 1)$, that is,

$$[f](x) = [f](\xi)\chi_\xi(x), \quad x \in [-1, 1],$$

where the indicator function $\chi_\xi(x)$ is 1 if $x = \xi$ and 0 otherwise, and note that the technique works for multiple jumps presuming they are adequately separated [20]. Since $\chi_\xi(x)$ is not smooth, we first approximate it using $h(\frac{x-\xi}{\sigma})$, $\sigma > 0$, which is the scaled bell-shape function approximation around $h(0) = 1$. When the scaling parameter σ is small, $h(\frac{x}{\sigma})$ is narrow and closely approximates the indicator function. Thus we write

$$[f](x) \approx [f](\xi)h\left(\frac{x-\xi}{\sigma}\right), \quad x \in [-1, 1].$$

Observe that the Fourier measurements for the approximate jump function are given by

$$\widehat{[f]}(\lambda_j) = \left\langle [f](\xi)h\left(\frac{x-\xi}{\sigma}\right), \psi_j \right\rangle = [f](\xi)\mathrm{e}^{-\pi i\lambda_j\xi}\sigma\hat{h}(\lambda_j\sigma).$$

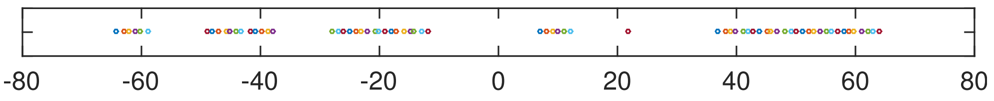

FIG. 10. *One-dimensional sparse nonuniform frequencies.*

Moreover, a first order approximation of $\hat{f}(\lambda_j)$ obtained through integration by parts is

$$\hat{f}(\lambda_j) \approx \frac{1}{\pi i \lambda_j}[f](\xi)\mathrm{e}^{-\pi i \lambda_j \xi},$$

implying

$$\widehat{[f]}(\lambda_j) \approx \pi i \lambda_j \hat{f}(\lambda_j)\sigma\hat{h}(\lambda_j\sigma). \tag{7}$$

Combining this approximation with the approximation of the canonical dual frame [31] yields the jump function approximation given by

$$[f](x) \approx \sum_{|j|\le m} \pi i \lambda_j \hat{f}(\lambda_j)\sigma\hat{h}(\lambda_j\sigma)\sum_{|l|\le m} b_{l,j}\phi_l(x), \quad x \in [-1,1], \tag{8}$$

where $\phi_l(x) = \mathrm{e}^{\pi i l x}$ and $\boldsymbol{B} = [b_{l,j}]_{l=-m,j=-m}^{m,m}$ is the Moore–Penrose pseudoinverse of $[\langle \psi_j, \phi_l\rangle]_{j=-m,l=-m}^{m,m}$.[1]

We now choose various bell-shape functions h_c for (8) and evaluate it on a fixed grid $\{x_1, x_2, \dots, x_N\} \subseteq [-1,1]$ to construct the (approximated) edge vectors, $\boldsymbol{y}_c$, $c = 1,\dots,C$, where each element $\boldsymbol{y}_{tc}$ is given by

$$\boldsymbol{y}_{tc} = \sum_{|j|\le m} \pi i \lambda_j \hat{f}(\lambda_j)\sigma\widehat{h_c}(\lambda_j\sigma)\sum_{|l|\le m} b_{l,j}\phi_l(x_t), \quad 1 \le t \le N. \tag{9}$$

Numerical experiments for one-dimensional edge detection. For our numerical experiments in one-dimensional edge detection, we assume we are given a set of Fourier data $\hat{f}(\lambda_j)$, $j = -m, \dots, m$, which we generate by randomly selecting (subsampling) 70 points from the jittered nonuniform frequencies

$$\lambda_j = j + \delta_j, \quad -64 \le j \le 64.$$

Here δ_j is a random perturbation uniformly sampled in $[-1/4, 1/4]$. The set of nonuniform frequencies used in our experiments is displayed in Figure 10.

We then construct $\boldsymbol{y}_c$, $c = 1,\dots,C$, using the following choices for h_c (and scaling parameter $\sigma = .05$ in $h_c(\frac{(x-\xi)}{\sigma})$):

1. $h_1(x) = \chi_{[-1,1]}(x)$;
2. $h_2(x) = 1 - |x|$;
3. $h_3(x) = 1 - x^2$;
4. $h_4(x) = (1-x^2)^2$;
5. $h_5(x) = (1-x^2)^3$;
6. $h_6(x) = \mathrm{e}^{-5x^2}$.

[1]The approximation (8) for uniform coefficients was coined the "concentration factor" edge detection method in [21].

We test Algorithm 3 on the piecewise continuous function given by

$$f(x) = \begin{cases} -1/2(1-x^2)^2, & x \le -1/2, \\ \cos(4\pi x), & -1/2 < x < 1/2, \\ (1-x^2)^4, & x \ge 1/2, \end{cases}$$

which has edges at $\xi = \pm\frac{1}{2}$. The weights in the fourth step are given as

$$w_i = \frac{\frac{1}{C}\sum_{c=1}^{C} |y_{i,c}| + \epsilon}{v_i + \epsilon}, \qquad i = 1, \dots, N. \tag{10}$$

Here $\epsilon = 10^{-6}$ so that $w_i \neq 0$ and $w_i \neq \infty$. Step 4 in Algorithm 3 was accomplished using the regularized unconstrained optimization

$$\hat{\boldsymbol{x}}_c = \underset{\boldsymbol{z}\in\mathbb{C}^N}{\operatorname{argmin}} \frac{1}{2}\|\boldsymbol{z} - \boldsymbol{y}_c\|_2^2 + \mu\|\boldsymbol{z}\|_{1,\boldsymbol{w}}, \qquad c = 1, \dots, C, \tag{11}$$

where the regularization parameter is set to be $\mu = .01$ in our numerical experiments. The minimization problem (11) has a closed form solution given by [9]

$$\hat{\boldsymbol{x}}_c = \max\{|\boldsymbol{y}_c| - \mu\boldsymbol{w}, 0\}\operatorname{sgn}(\boldsymbol{y}_c),$$

where the absolute value, max function, and sgn function are each evaluated in a componentwise manner.

We note that the weighting parameter in (10) is modified from the previous choice in (3), as it is designed to detect multiple edges with different jump heights. In particular, numerical experiments suggest that the variance surrounding edges with large magnitudes tend to dominate the variance around other smaller-in-magnitude edges. Consequently, when using the weight in (3), the smaller magnitude edges may be seen as part of smooth regions. By observing that the variance of a scaled random variable corresponds with its square, the weights in (10) are constructed with the idea that the variance is roughly proportional to the square of the jump heights. By defining our weights in this way, we can somewhat alleviate the problem caused by the dominance of the large magnitude edges. A more thorough analysis of scaling for weights and parameter selection will be discussed in future work.

Figure 11 compares the individual measurement vectors, $\boldsymbol{y}_c$, $c = 1, \dots, 6$, to the results when Algorithm 3 is used. The variance of these vectors is shown in the bottom left. We also display the result of the regular ℓ^1 penalty (that is, with constant weights) in the bottom middle. The VBJS method yields the closest approximation to the true edge locations and is significantly better than any of the individual edge detection results. It is important to note that in this example, only *one* measurement vector is collected. However, by *processing* this data in multiple ways, we are better able to tease out important information of the underlying signal. Moreover, the method is easily parallelizable for constructing each $\boldsymbol{y}_c$.

Numerical experiments for two-dimensional edge detection. We now consider detecting edges for a two-dimensional image from nonuniform Fourier data. Suppose we are given the Fourier coefficients $\hat{f}(\boldsymbol{\lambda})_j$ for $\boldsymbol{\lambda}_j = (\lambda_{j1}, \lambda_{j2})$ for piecewise smooth function $f : [-1,1]^2 \to \mathbb{R}$ with a smooth edge curve. As in the one-dimensional problem, we seek to recover the edge function $[f](x,y)\chi_\Gamma(x,y)$, where $[f](x,y)$ is the jump height and $\chi_\Gamma(x,y)$ is the indicator function. We parameterize the edge Γ as

$$x = u(s), \qquad y = v(s), \qquad s \in [a,b],$$

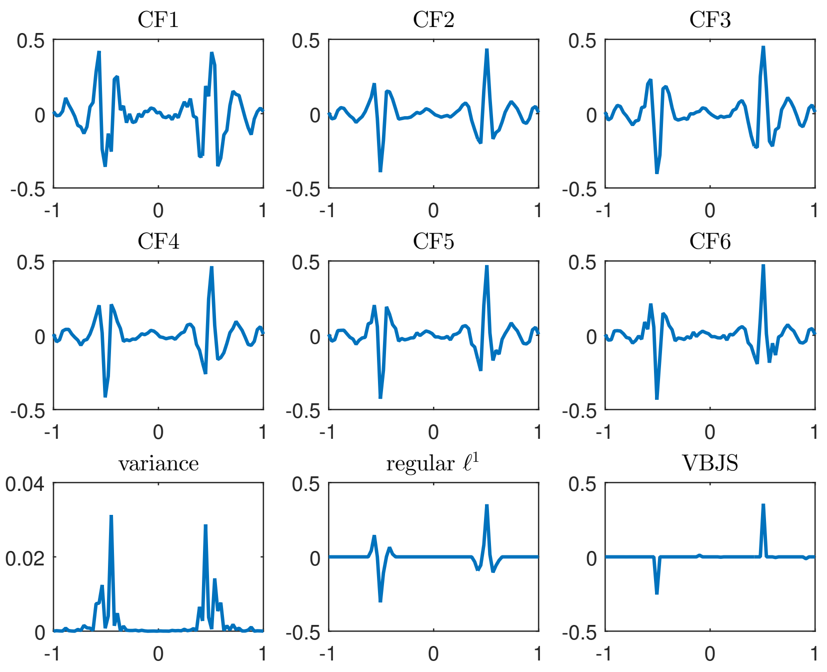

Fig. 11. *CF1–CF6 are the edge detectors $\boldsymbol{y}_c, 1 \le c \le 6$ (step 1 in Algorithm 3), computed using the concentration factor method in (9). The variance (step 2 in Algorithm 3) is shown in the bottom left. The final two pictures compare the results for the standard ℓ^1 regularization and the VBJS algorithm.*

where u and v are smooth functions. Now let $\theta(s)$ be its normal direction. As before, we devise a strategy to recover a regularized edge function. Therefore we describe the parameterized edge curve as

$$x = u(s) + r\cos\theta(s), \qquad y = v(s) + r\sin\theta(s), \qquad r \in [-\sigma, \sigma],$$

for some small σ. We now define the regularized edge function as

$$J(x,y) = h\left(\frac{r}{\sigma}\right)[f](s),$$

where $[f](s) = [f](u(s), v(s))$. In this case we only consider the bell-shaped regularization function, $h(z) = \mathrm{e}^{-5z^2}$, although other regularization functions are also suitable.

To generate multiple data sets, we will use different rotation angles

$$\theta_c := \pi c/C \tag{12}$$

for the construction of each edge detection approximation vector $\boldsymbol{y}_c, c = 1, \dots, C$. We will then combine the results to produce a final edge map. We note that a similar approach was used in [28]. To simplify notation, below we describe the rotation algorithm for uniform Fourier data, i.e., $(\lambda_{j1}, \lambda_{j2}) = (j, l)$, $-N \le j, l \le N$. However, for our numerical results we will apply the rotation technique to nonuniform Fourier data, as shown in (16).

Observe that

$$J(x,y) = \sum_{j,l} \hat{J}(j,l) e^{\pi i(jx+ly)}.$$

After eliminating the higher order terms (as in the one-dimensional case), in the new coordinate system $\hat{f}(j,l)$ can be approximated as

$$\hat{f}(j,l) \approx \int_a^b \frac{[f](s)}{\pi i(j\cos\theta(s) + l\sin\theta(s))} e^{-\pi i(ju+lv)}(v'\cos\theta(s) - u'\sin\theta(s))ds, \tag{13}$$

while

$$\hat{J}(j,l) \approx \int_a^b [f](s)\sigma\hat{h}\left(\sigma(j\cos\theta(s) + l\sin\theta(s))\right) e^{-\pi i(ju+lv)}(v'\cos\theta(s) - u'\sin\theta(s))ds. \tag{14}$$

Unlike in the one-dimensional case, here there is no linear relationship between $\hat{f}(j,l)$ and $\hat{J}(j,l)$ since $\theta = \theta(s)$. However, if we *fix* θ as in (12), then for $\theta = \theta_c$ we have

$$\hat{J}_\theta(j,l) = \pi i(j\cos\theta + l\sin\theta)\sigma\hat{h}(\sigma(j\cos\theta + l\sin\theta)\hat{f}(j,l), \tag{15}$$

which is comparable to what we see in (7). For $c = 1,\dots,C$ we now have

$$J_{\theta_c}(x,y) \approx \sum_{j=-N}^{N}\sum_{l=-N}^{N} \hat{J}_\theta(j,l)e^{\pi i(jx+ly)}.$$

We now follow the one-dimensional approach for *nonuniform* Fourier data. In particular, we extend (9) to two dimensions and calculate the components of each vector in step 1 of Algorithm 3 as

$$\boldsymbol{y}_{tc} = \sum_{|j|\le m} \pi i(\lambda_{j_1}\cos\theta_c + \lambda_{j_2}\sin\theta_c)\sigma\hat{h}((\lambda_{j_1}\cos\theta_c + \lambda_{j_2}\sin\theta_c)\sigma)\hat{f}(\lambda_{\boldsymbol{j}})\sum_{|l|\le m} b_{l,j}\phi_l(x_t), \tag{16}$$

where $\{x_t = (x_{t1}, x_{t2}) : 1 \le t \le N\}$ in $[-1,1]^2$. Here $\psi_j(x) = \mathrm{e}^{\pi i(\lambda_{j_1}x_1 + \lambda_{j_2}x_2)}$, $\phi_l(x) = \mathrm{e}^{\pi i(l_1x_1 + l_2x_2)}$, $\boldsymbol{B} = [b_{l,j}]_{|l|\le m,|j|\le m}$ is the Moore–Penrose pseudoinverse of $\left[\langle\psi_j,\phi_l\rangle\right]_{|l|\le m,|j|\le m}$, and once again σ is the scaling parameter of h.

Algorithm 3 is tested on the simulated phantom displayed in Figure 12 (left). The Fourier data are generated by randomly selecting 25% points from the jittered grid points

$$\lambda_j = (j_1, j_2) + (\delta_{j1}, \delta_{j2}), \quad -32 \le j_1, j_2 \le 32,$$

where both δ_{j1} and δ_{j2} are random perturbations uniformly sampled in $[-1/4, 1/4]$. The nonuniform frequencies used in our experiment are shown in Figure 12 (right). The scaling parameter is $\sigma = .05$ and the weights are given in (10). The number of different rotation angles is set at $C = 20$.

As before we compute $\boldsymbol{y}_c$ for $c = 1,\dots,C$ separately and then use Algorithm 3 to construct the VBJS approximation. The results are displayed in Figure 13, where it is evident that combining each individual result using VBJS yields better results than averaging across each edge detection approximation.

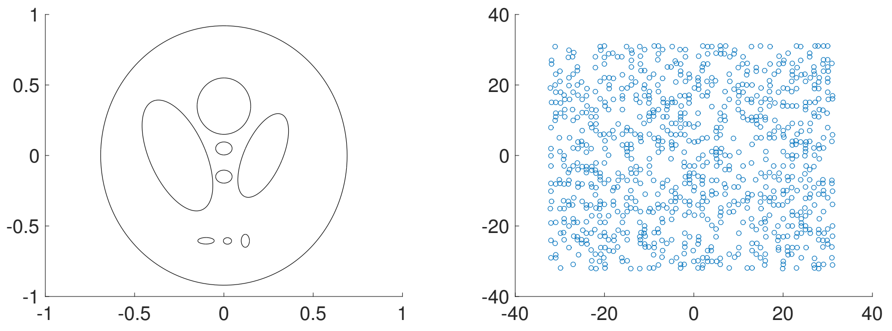

FIG. 12. *A two-dimensional phantom (left) and two-dimensional nonuniform sampling frequencies (right).*

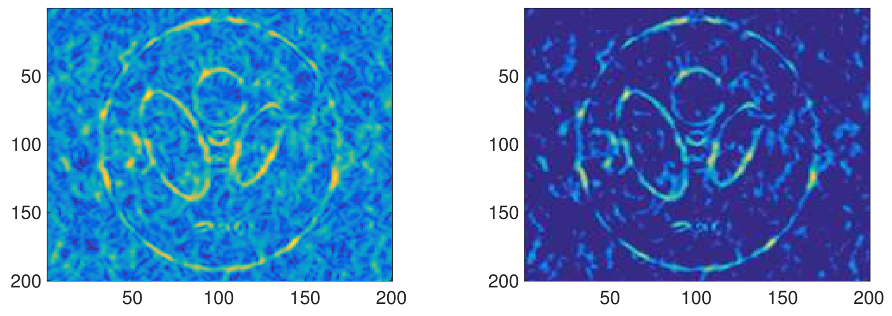

FIG. 13. *Two-dimensional edge detection: the average edge vector (left) and the VBJS edge vector (right).*

6. Conclusions. In this paper we presented a simple procedure for exploiting joint sparsity using variances. Unlike more standard approaches, it is easily parallelizable requiring just two steps of C parallel ℓ^1-minimization solves. Furthermore, it often achieves a better overall reconstruction, as was demonstrated both in synthetic phase transition experiments on randomly generated sparse vectors and in the two imaging applications provided.

There are several areas for future work. First, we have only considered variance-based approaches in the context of ℓ^1-minimization/$\ell^{2,1}$-minimization for sparse recovery. Greedy or thresholding algorithms such as orthogonal matching pursuit and iterative hard thresholding, and their various generalizations, also have variations akin to $\ell^{2,1}$-minimization for the MMV problem [3, 16, 32]. Developing variance-based versions of these algorithms is an objective for future work. Theoretical analysis of such approaches, as well as VBJS introduced in this paper, is another topic for future investigation.

Other weighting choices should also be explored, with optimal choices likely being signal dependent. In particular, there is no need to use the same weight vector for every signal in the second stage of the VBJS procedure. Moreover, an effective choice of weights should further increase the robustness of regularization parameter selection (e.g., μ in (11)).

The VBJS algorithm may also help to reduce the effects of bad information, since variance-based weighting means that locations where data measurements do not agree are deemed less significant. This is important in applications such as synthetic aperture radar, where sometimes shadowing and adjacency cause some measurements to be highly prone to errors. Measurements may also intentionally be false or misleading. Other areas of potential impact include multiple source information, that is, when different measurement matrices are used. In this case the VBJS algorithm should adapt in a straightforward manner. Such situations will be considered in future investigations.

Acknowledgment. The authors would like to thank Rodrigo Platte for helpful comments and feedback and for his assistance with several of the numerical experiments.

REFERENCES

[1] B. Adcock and A. C. Hansen, *Generalized sampling and infinite-dimensional compressed sensing*, Found. Comput. Math., 16 (2016), pp. 1263–1323.

[2] D. Baron, M. F. Duarte, M. B. Wakin, S. Sarvotham, and R. G. Baraniuk, *Distributed Compressive Sensing*, arXiv:0901.3403, 2009.

[3] J. D. Blanchard, M. Cermak, D. Hanle, and Y. Jing, *Greedy algorithms for joint sparse recovery*, IEEE Trans. Signal Process., 62 (2014), pp. 1694–1704.

[4] P. Boufounos, G. Kutyniok, and H. Rauhut, *Average case analysis of sparse recovery from combined fusion frame measurements*, in Proceedings of the 44th Annual Conference on Information Sciences and Systems (CISS), 2010, pp. 1–6.

[5] P. Boufounos, G. Kutyniok, and H. Rauhut, *Sparse recovery from combined fusion frame measurements*, IEEE Trans. Inform. Theory, 57 (2011), pp. 3864–3876.

[6] E. J. Candès, M. B. Wakin, and S. P. Boyd, *Enhancing sparsity by reweighted ℓ_1 minimization*, J. Fourier Anal. Appl., 14 (2008), pp. 877–905.

[7] I.-Y. Chun and B. Adcock, *Compressed sensing and parallel acquisition*, IEEE Trans. Inform. Theory, 63 (2017), pp. 4860–4882.

[8] I.-Y. Chun, B. Adcock, and T. Talavage, *Efficient compressed sensing sense PMRI reconstruction with joint sparsity promotion*, IEEE Trans. Med. Imag., 31 (2016), pp. 354–368.

[9] P. L. Combettes and J.-C. Pesquet, *Proximal splitting methods in signal processing*, in Fixed-Point Algorithms for Inverse Problems in Science and Engineering, Springer Optim. Appl. 49, Springer, New York, 2011, pp. 185–212, https://doi.org/10.1007/978-1-4419-9569-8_10.

[10] S. F. Cotter, B. D. Rao, K. Engan, and K. Kreutz-Delgado, *Sparse solutions to linear inverse problems with multiple measurement vectors*, IEEE Trans. Signal Process., 53 (2005), pp. 2477–2488.

[11] M. A. Davenport, M. F. Duarte, Y. C. Eldar, and G. Kutyniok, *Introduction to compressed sensing*, in Compressed Sensing: Theory and Applications, Cambridge University Press, Cambridge, UK, 2011.

[12] M. F. Duarte and Y. C. Eldar, *Structured compressed sensing: From theory to applications*, IEEE Trans. Signal Process., 59 (2011), pp. 4053–4085.

[13] Y. C. Eldar, *Robust recovery of signals from a structured union of subspaces*, IEEE Trans. Inform. Theory, 55 (2009), pp. 5302–5316.

[14] Y. C. Eldar and H. Rauhut, *Average case analysis of multichannel sparse recovery using convex relaxation*, IEEE Trans. Inform. Theory, 56 (2010), pp. 505–519.

[15] S. Engelberg and E. Tadmor, *Recovery of edges from spectral data with noise: A new perspective*, SIAM J. Numer. Anal., 46 (2008), pp. 2620–2635.

[16] S. Foucart, *Recovery jointly sparse vectors via hard thresholding pursuit*, in Proceedings of the 9th International Conference on Sampling Theory and Applications, 2011.

[17] S. Foucart and H. Rauhut, *A Mathematical Introduction to Compressive Sensing*, Birkhäuser, Basel, 2013.

[18] M. Friedlander, H. Mansour, R. Saab, and I. Yilmaz, *Recovering compressively sampled signals using partial support information*, IEEE Trans. Inform. Theory, 58 (2012), pp. 1122–1134.

[19] A. Gelb and T. Hines, *Detection of edges from nonuniform Fourier data*, J. Fourier Anal. Appl., 17 (2011), pp. 1152 – 1179.

[20] A. Gelb and G. Song, *Detecting edges from non-uniform Fourier data using Fourier frames*, J. Sci. Comput., 71 (2017), pp. 737–758.

[21] A. Gelb and E. Tadmor, *Detection of edges in spectral data*, Appl. Comput. Harmon. Anal., 7 (1999), pp. 101–135.

[22] M. Grant and S. Boyd, *CVX: MATLAB Software for Disciplined Convex Programming, Version* 2.0 *Beta*, http://cvxr.com/cvx (2013).

[23] R. Gribonval, H. Rauhut, K. Schnass, and P. Vandergheynst, *Atoms of all channels, unite! Average case analysis of multi-channel sparse recovery using greedy algorithms*, J. Fourier Anal. Appl., 13 (2008), pp. 655–687.

[24] M. Guerquin-Kern, L. Lejeune, K. P. Pruessmann, and M. Unser, *Realistic analytical phantoms for parallel Magnetic Resonance Imaging*, IEEE Trans. Med. Imaging, 31 (2012), pp. 626–636.

[25] A. Majumdar and R. K. Ward, *Calibration-less multi-coil MR image reconstruction*, Magn. Reson. Imaging, 7 (2012), pp. 1032–1045.

[26] A. Makhzani and S. Valaee, *Reconstruction of jointly sparse signals using iterative hard thresholding*, in Proceedings of the IEEE International Conference on Communications (ICC), 2012, pp. 3564–3568.

[27] H. Mansour and R. Saab, *Recovery analysis for weighted ℓ_1-minimization using the null space property*, Appl. Comput. Harmon. Anal., 43 (2017), pp. 23–38.

[28] A. Martinez, A. Gelb, and A. Gutierrez, *Edge detection from non-uniform Fourier data using the convolutional gridding algorithm*, J. Sci. Comput., 61 (2014), pp. 490–512.

[29] M. Mishali and Y. C. Eldar, *Reduce and boost: Recovering arbitrary sets of jointly sparse vectors*, IEEE Trans. Signal Process., 56 (2008), pp. 4692–4702.

[30] H. Monajemi, S. Jafarpour, M. Gavish, D. L. Donoho, S. Ambikasaran, S. Bacallado, D. Bharadia, Y. Chen, Y. Choi, M. Chowdhury, et al., *Deterministic matrices matching the compressed sensing phase transitions of Gaussian random matrices*, Proc. Natl. Acad. Sci. USA, 110 (2013), pp. 1181–1186.

[31] G. Song and A. Gelb, *Approximating the inverse frame operator from localized frames*, Appl. Comput. Harmon. Anal., 35 (2013), pp. 94–110.

[32] J. A. Tropp, A. C. Gilbert, and M. J. Strauss, *Algorithms for simultaneous sparse approximation. Part* I: *Greedy pursuit*, Signal Process., 86 (2006), pp. 572–588.

[33] J. A. Tropp, A. C. Gilbert, and M. J. Strauss, *Algorithms for simultaneous sparse approximation. Part* II: *Convex relaxation*, Signal Process., 86 (2006), pp. 589–602.

[34] E. van den Berg and M. P. Friedlander, *SPGL*1*: A Solver for Large-Scale Sparse Reconstruction*, http://www.cs.ubc.ca/labs/scl/spgl1 (2007).

[35] E. van den Berg and M. P. Friedlander, *Theoretical and empirical results for recovery from multiple measurements*, IEEE Trans. Inform. Theory, 56 (2010), pp. 2516–2527.

[36] J. Yang and J. Zhang, *Alternating direction algorithms for L*1*-problems in compressive sensing*, SIAM J. Sci. Comput., 33 (2011), pp. 250–278.