SIAM J. SCI. COMPUT.
Vol. 31, No. 6, pp. 4533–4579

SIMULTANEOUSLY SPARSE SOLUTIONS TO LINEAR INVERSE PROBLEMS WITH MULTIPLE SYSTEM MATRICES AND A SINGLE OBSERVATION VECTOR*

ADAM C. ZELINSKI†, VIVEK K. GOYAL†, AND ELFAR ADALSTEINSSON†

Abstract. A problem that arises in slice-selective magnetic resonance imaging (MRI) radio-frequency (RF) excitation pulse design is abstracted as a novel linear inverse problem with a simultaneous sparsity constraint. Multiple unknown signal vectors are to be determined, where each passes through a different system matrix and the results are added to yield a single observation vector. Given the matrices and lone observation, the objective is to find a *simultaneously sparse* set of unknown vectors that approximately solves the system. We refer to this as the *multiple-system single-output* (MSSO) simultaneous sparse approximation problem. This manuscript contrasts the MSSO problem with other simultaneous sparsity problems and conducts an initial exploration of algorithms with which to solve it. Greedy algorithms and techniques based on convex relaxation are derived and compared empirically. Experiments involve sparsity pattern recovery in noiseless and noisy settings and MRI RF pulse design.

Key words. iterative shrinkage, iteratively reweighted least squares, magnetic resonance imaging excitation pulse design, matching pursuit, multiple measurement vectors, second-order cone programming, simultaneous sparse approximation, sparse approximation

AMS subject classifications. 15A29, 34A55, 94A12, 46N10

DOI. 10.1137/080730822

1. Introduction.

In this work we propose a linear inverse problem that requires a *simultaneously sparse* set of vectors as the solution, i.e., a set of vectors where only a small number of each vector's entries are nonzero, and where the vectors' *sparsity patterns* (the locations of the nonzero entries) are equal. Sparsity constraints and regularizations that promote sparsity have a long history that we will not attempt to recount; the reader is referred to [15] both for a theoretical result on the numerical robustness of using sparsity constraints and for references to early empirical work. While perhaps an old topic, the estimation of sparse or approximately sparse signals is also an extremely active area of research because of the emergence of compressed sensing [5, 17] and a multitude of applications for wavelet-domain sparsity of images; a review of recent developments with an emphasis on algorithms and performance guarantees is provided by [4].

We call the problem of interest *multiple-system single-output (MSSO) simultaneous sparse approximation*:

$$\mathbf{d} \approx \sum_{p=1}^{P} \mathbf{F}_p \mathbf{g}_p, \tag{1.1}$$

*Received by the editors July 21, 2008; accepted for publication (in revised form) October 26, 2009; published electronically January 20, 2010. This material is based upon work supported by the National Institutes of Health under grants 1P41RR14075, 1R01EB000790, 1R01EB006847, and 1R01EB007942; the National Science Foundation under CAREER Grant 0643836; United States Department of Defense National Defense Science and Engineering Graduate Fellowship F49620-02-C-0041; the MIND Institute; the A. A. Martinos Center for Biomedical Imaging; Siemens Medical Solutions; and R. J. Shillman's Career Development Award.
http://www.siam.org/journals/sisc/31-6/73082.html

†Research Laboratory of Electronics, Massachusetts Institute of Technology, 77 Massachusetts Avenue, Cambridge, MA 02139 (zelinski@mit.edu, vgoyal@mit.edu, elfar@mit.edu).

where $\mathbf{d} \in \mathbb{C}^M$, $\mathbf{F}_p \in \mathbb{C}^{M \times N}$ for each p, and each $\mathbf{g}_p \in \mathbb{C}^N$ has a common sparsity pattern or, more precisely, we measure the sparsity level by the number of positions at which *any* of the $\mathbf{g}_p$'s is nonzero. This name highlights the distinction from the more common setting of *single-system multiple-output (SSMO) simultaneous sparse approximation*:

$$\mathbf{d}_p \approx \mathbf{F}\mathbf{g}_p \quad \text{for } p = 1, \ldots, P, \tag{1.2}$$

where $\mathbf{d}_p \in \mathbb{C}^M$ for each p, $\mathbf{F} \in \mathbb{C}^{M \times N}$, and the $\mathbf{g}_p$'s again have a common sparsity pattern. In either case, associated inverse problems are to minimize the simultaneous sparsity level of the $\mathbf{g}_p$'s for a given goodness of fit, to optimize the goodness of fit for a given sparsity level, or to minimize an objective function that combines goodness of fit and sparsity level. There is a large and growing literature on SSMO problems, often using the phrase *multiple measurement vectors*, but MSSO problems have received much less attention.

MSSO sparse approximation came to our attention as an abstraction of the placement of spokes in short-duration, slice-selective magnetic resonance imaging (MRI) radio-frequency (RF) excitation pulse design [70, 73, 74], and we know of no earlier application. To keep the bulk of the developments applicable generally, we defer a description of the MRI application until the end of the paper. The paper is focused on formalizing the MSSO sparse approximation problem and introducing and contrasting several algorithms for finding approximate solutions. We have implemented and tested three greedy algorithms—generalizing matching pursuit (MP) [44], orthogonal matching pursuit (OMP) [6, 51, 14, 46, 9, 10], and least squares matching pursuit (LSMP) [10]—and also algorithms for solving a convex relaxation of the MSSO problem, applying second-order cone programming (SOCP) [3, 43], and generalizing iteratively reweighted least squares (IRLS) [36, 31] and iterative shrinkage [20, 12, 21, 22]. We evaluate the performance of the algorithms across three experiments: the first and second involve sparsity pattern recovery in noiseless and noisy scenarios, respectively, while the third deals with MRI RF excitation pulse design.

It is worth noting explicitly that this paper does not provide conditions for optimality (i.e., equivalence with a generally intractable sparsity-constrained problem) of greedy algorithms or convex relaxations, nor does it analyze random ensembles of problems. The sparse approximation literature is increasingly dominated by these types of analyses, and we cite these results in the appropriate sections.

The structure of this paper is as follows: in section 2, we provide background information about ordinary sparse approximation and SSMO sparse approximation. In section 3, we formulate the MSSO problem. Algorithms for solving the problem are then posed in section 4. The coverage of some algorithms is very brief; additional details on these can be found in [70, 72]. Details and results of the numerical experiments appear in section 5. Section 6 highlights the strengths and weaknesses of the algorithms and presents ideas for future work. Concluding remarks are given in section 7.

2. Background.

2.1. Single-system single-output (SSSO) sparse approximation.

Following our naming rubric, ordinary sparse approximation can be called SSSO sparse approximation for emphasis. The problem can be written as

$$\underset{\mathbf{g} \in \mathbb{C}^N}{\arg\min} \left\{ \tfrac{1}{2} \|\mathbf{d} - \mathbf{F}\mathbf{g}\|_2^2 + \lambda \|\mathbf{g}\|_0 \right\}, \quad \text{given } \mathbf{d} \in \mathbb{C}^M \text{ and } \mathbf{F} \in \mathbb{C}^{M \times N}, \tag{2.1}$$

where $\|\cdot\|_0$ denotes the number of nonzero elements of a vector and $\lambda \in (0,\infty)$ is a control parameter. Varying λ determines the relative importance of fitting the data $\mathbf{d}$ and keeping the solution $\mathbf{g}$ sparse. In many applications, λ is set to yield a specified sparsity or specified residual; it may also have other significance. For general $\mathbf{F}$, solving (2.1) is NP-hard [13, 46]. Thus, great effort has gone into the design and analysis of approximate algorithms.

A greedy approach is to iteratively choose one position for a nonzero entry of $\mathbf{g}$ at a time or, equivalently, to pick one column of $\mathbf{F}$ at a time. To select the column of $\mathbf{F}$ to maximize the magnitude of the inner product with the current residual is called matching pursuit (MP) [44], and several more sophisticated variants have been proposed [6, 51, 14, 46, 9, 10]. Most important among these is orthogonal MP (OMP), which avoids deleterious interactions from iteration to iteration by working in the orthogonal complement of all previously selected columns. Notable analyses of OMP are those in [19, 61], where sufficient conditions are given for OMP to recover the sparsity pattern of the solution of (2.1). Analyses of OMP for random problem ensembles are given in [26, 64].

A second approach is to replace $\|\cdot\|_0$ in (2.1) with its *relaxation* [7, 57]:

$$\operatorname*{arg\,min}_{\mathbf{g}\,\in\,\mathbb{C}^N} \left\{ \tfrac{1}{2}\|\mathbf{d}-\mathbf{F}\mathbf{g}\|_2^2 + \lambda\|\mathbf{g}\|_1 \right\}, \quad \text{given } \mathbf{d}\in\mathbb{C}^M \text{ and } \mathbf{F}\in\mathbb{C}^{M\times N}. \tag{2.2}$$

This is a convex optimization and thus may be solved efficiently [3]. Certain conditions on $\mathbf{F}$ guarantee proximity of the solutions to (2.1) and (2.2) [18, 19, 63]. Analyses of random problem ensembles are given in [53, 67].

Note that (2.2) applies an ℓ_1 norm to $\mathbf{g}$, but an ℓ_p norm (where $p<1$) may also be used to promote sparsity [31, 7]; this leads to a nonconvex problem and will not considered in this paper. A problem of the form (2.2) may arise as a proxy for (2.1) or as the inherent problem of interest. For example, in a Bayesian estimation setting, (2.2) yields the maximum a posteriori probability estimate of $\mathbf{g}$ from $\mathbf{d}$ when the observation model involves $\mathbf{F}$ and Gaussian noise and the prior on $\mathbf{g}$ is Laplacian. Similar statements can be made about the relaxations in the following sections.

2.2. Single-system multiple-output (SSMO) simultaneous sparse approximation. In SSMO, each of P observation vectors $\mathbf{d}_p$ is approximated by a product $\mathbf{F}\mathbf{g}_p$, where the $\mathbf{g}_p$'s are simultaneously sparse. This yields the problem

$$\operatorname*{arg\,min}_{\mathbf{G}\,\in\,\mathbb{C}^{N\times P}} \left\{ \tfrac{1}{2}\|\mathbf{D}-\mathbf{F}\mathbf{G}\|_{\mathrm{F}}^2 + \lambda\|\mathbf{G}\|_{0,2} \right\}, \quad \text{given } \mathbf{D}\in\mathbb{C}^{M\times P} \text{ and } \mathbf{F}\in\mathbb{C}^{M\times N}, \tag{2.3}$$

where $\mathbf{D}=[\mathbf{d}_1,\ldots,\mathbf{d}_P]$, $\mathbf{G}=[\mathbf{g}_1,\ldots,\mathbf{g}_P]$, $\|\cdot\|_{\mathrm{F}}$ is the Frobenius norm, and $\|\cdot\|_{0,2}$ is the number of rows with nonzero ℓ_2 norm (i.e., the simultaneous sparsity level).[1] This reduces to (2.1) when $P=1$ and is thus also generally computationally intractable.

Greedy algorithms for approximating (2.3) were first developed in [11, 65]. Deterministic conditions for sparsity pattern recovery and average case analyses for greedy algorithms are both presented in [32].

Analogous to the relaxation (2.2) is

$$\operatorname*{arg\,min}_{\mathbf{G}\,\in\,\mathbb{C}^{N\times P}} \left\{ \tfrac{1}{2}\|\mathbf{D}-\mathbf{F}\mathbf{G}\|_{\mathrm{F}}^2 + \lambda\|\mathbf{G}\|_{\mathrm{S}} \right\}, \quad \text{given } \mathbf{D}\in\mathbb{C}^{M\times P} \text{ and } \mathbf{F}\in\mathbb{C}^{M\times N}, \tag{2.4}$$

[1] The choice of ℓ_2 here is arbitrary; it can be replaced by any vector norm. We write $\|\cdot\|_{0,2}$ because the relaxation we use subsequently is then naturally $\|\cdot\|_{1,2}$.

where

$$\|\mathbf{G}\|_{\mathrm{S}} = \|\mathbf{G}\|_{0,1} = \sum_{n=1}^{N} \sqrt{\sum_{p=1}^{P} |\mathbf{G}(n,p)|^2} = \sum_{n=1}^{N} \sqrt{\sum_{p=1}^{P} |\mathbf{g}_p[n]|^2}; \tag{2.5}$$

i.e., $\|\mathbf{G}\|_{\mathrm{S}}$ is the ℓ_1 norm of the ℓ_2 norms of the rows of the $\mathbf{G}$ matrix.[2] This latter operator is a *simultaneous sparsity norm*: it penalizes the program (produces large values) when the columns of $\mathbf{G}$ have dissimilar sparsity patterns [43]. Fixing λ to a sufficiently large value and solving this optimization yields simultaneously sparse $\mathbf{g}_p$'s.

Given the convex objective function in (2.4), one may then attempt to find a solution that minimizes the objective using an IRLS-based [11] or SOCP-based [43] approach. A formal analysis of the minimization of the convex objective may be found in [62]. Convex relaxations for this problem are also studied in [25, 28]. A related approach that may incorporate additional prior information is given in [68], and a boosting strategy that may be combined with either a greedy algorithm or a convex relaxation is presented in [45] and analyzed further in [66]. Also applicable to both greedy algorithms and convex relaxation are results in [8] that are analogous to the principal results of [19, 61, 63].

3. Multiple-system single-output (MSSO) simultaneous sparse approximation. We outline the MSSO problem in a style analogous to that of SSMO systems in (2.3), (2.4) and then pose a second formulation that is useful for deriving several algorithms in section 4.

3.1. MSSO problem formulation. Consider the following system:

$$\mathbf{d} = \mathbf{F}_1\mathbf{g}_1 + \cdots + \mathbf{F}_P\mathbf{g}_P = [\mathbf{F}_1 \cdots \mathbf{F}_P] \begin{bmatrix} \mathbf{g}_1 \\ \vdots \\ \mathbf{g}_P \end{bmatrix} = \mathbf{F}_{\mathrm{tot}}\mathbf{g}_{\mathrm{tot}}, \tag{3.1}$$

where $\mathbf{d} \in \mathbb{C}^M$ and the $\mathbf{F}_p \in \mathbb{C}^{M\times N}$ are known. Unlike the SSMO problem, there are now only one observation and P different system matrices. Here again we desire an approximate solution where the $\mathbf{g}_p$'s are simultaneously sparse, formally,

$$\min_{\mathbf{g}_1,\ldots,\mathbf{g}_P} \|\mathbf{d} - \mathbf{F}_{\mathrm{tot}}\mathbf{g}_{\mathrm{tot}}\|_2 \quad \text{subject to (s.t.) the simultaneous } K\text{-sparsity of the } \mathbf{g}_p\text{'s}, \tag{3.2}$$

or an equivalent unconstrained formulation. There are no constraints on the values of M, N, or P; i.e., there is no explicit requirement that the system be overdetermined or underdetermined.

In the first half of section 4, we will pose three approaches that attempt to solve the MSSO problem (3.2) in a greedy fashion. Another approach is to apply a relaxation similar to (2.2), (2.4):

$$\min_{\mathbf{G}} \left\{ \tfrac{1}{2} \|\mathbf{d} - \mathbf{F}_{\mathrm{tot}}\mathbf{g}_{\mathrm{tot}}\|_2^2 + \lambda \|\mathbf{G}\|_{\mathrm{S}} \right\}, \tag{3.3}$$

[2] An ℓ_p norm with $p < 1$ could be used in place of the ℓ_1 norm if one is willing to deal with a nonconvex objective function. Further, an ℓ_q norm (with $q > 2$) rather than an ℓ_2 norm could be applied to each row of $\mathbf{G}$ because the purpose of the row operation is to collapse the elements of the row into a scalar value without introducing a sparsifying effect.

where $\mathbf{G}$ and $\|\mathbf{G}\|_{\mathrm{S}}$ are the same as in (2.4) and (2.5), respectively. In the second half of section 4, we will outline four algorithms for solving this relaxed problem.

3.2. Alternate formulation of the MSSO problem. The MSSO problem can be expressed in an equivalent form using new variables that are simply permutations of the $\mathbf{F}_p$'s and $\mathbf{g}_p$'s. First we define N new matrices:

$$\mathbf{C}_n = [\mathbf{f}_{1,n}, \ldots, \mathbf{f}_{P,n}] \in \mathbb{C}^{M \times P} \quad \text{for } n = 1, \ldots, N, \tag{3.4}$$

where $\mathbf{f}_{p,n}$ is the nth column of $\mathbf{F}_p$. Next we construct N new vectors:

$$\mathbf{h}_n = [\mathbf{g}_1[n], \ldots, \mathbf{g}_P[n]]^{\mathrm{T}} \in \mathbb{C}^P \quad \text{for } n = 1, \ldots, N, \tag{3.5}$$

where $\mathbf{g}_p[n]$ is the nth element of $\mathbf{g}_p$ and $^{\mathrm{T}}$ is the transpose operation. Using the $\mathbf{C}_n$'s and $\mathbf{h}_n$'s, we have another way to write $\mathbf{d}$:

$$\mathbf{d} = \mathbf{C}_1\mathbf{h}_1 + \cdots + \mathbf{C}_N\mathbf{h}_N = [\mathbf{C}_1 \cdots \mathbf{C}_N] \begin{bmatrix} \mathbf{h}_1 \\ \vdots \\ \mathbf{h}_N \end{bmatrix} = \mathbf{C}_{\mathrm{tot}}\mathbf{h}_{\mathrm{tot}}. \tag{3.6}$$

The relationship between the $\mathbf{g}_p$'s and $\mathbf{h}_n$'s implies that if we desire to find a set of simultaneously sparse $\mathbf{g}_p$'s to solve (3.1), we should seek out a set of $\mathbf{h}_n$'s where many of the $\mathbf{h}_n$'s equal an all-zeros vector, $\mathbf{0}$. This claim is apparent if we consider the fact that $\mathbf{H} = [\mathbf{h}_1, \ldots, \mathbf{h}_N]$ is equal to the transpose of $\mathbf{G}$. This formulation of MSSO has recently been termed block-sparsity [24, 55] because the nonzero entries of $\mathbf{h}_{\mathrm{tot}}$ come in contiguous blocks.

Continuing with this alternate formulation, and given our desire to find a solution where most of the $\mathbf{h}_n$'s are all-zeros vectors, we relax the problem as follows:

$$\min_{\mathbf{h}_{\mathrm{tot}}} \left\{ \tfrac{1}{2} \|\mathbf{d} - \mathbf{C}_{\mathrm{tot}}\mathbf{h}_{\mathrm{tot}}\|_2^2 + \lambda \sum_{n=1}^{N} \|\mathbf{h}_n\|_2 \right\}. \tag{3.7}$$

The equivalence of $\sum_{n=1}^{N} \|\mathbf{h}_n\|_2$ and $\|\mathbf{G}\|_{\mathrm{S}}$ means that (3.7) is equivalent to (3.3), and thus, just as in (3.3), the approach in (3.7) finds a set of simultaneously sparse $\mathbf{g}_p$'s.

3.3. Differences between the SSMO and MSSO problems. In the SSMO problem, the ratio of unknowns to knowns is N/M regardless of the number of observations, P. Increasing P has little effect on the residual error, but it is beneficial for estimation of a "best" or "correct" sparsity pattern because the simultaneous sparsity of the underlying $\mathbf{g}_p$'s becomes more exploitable. Empirical evidence of improved sparsity pattern recovery with increasing P may be found, for example, in [11, 43], and related theoretical results on random problem ensembles are given in [47, 48].

In contrast, the ratio of unknowns to knowns is not constant with respect to P in the MSSO problem; rather it is equal to PN/M. As P is increased, it becomes easier to achieve a small residual, but sparsity pattern recovery becomes harder. This is supported experimentally in section 5.

3.4. Previous results on MSSO problems. After the initial submission of this manuscript, algorithms similar to our OMP variation in section 4.2 were introduced as block OMP (BOMP) [24] and parallel OMP (POMP) [41]. (POMP is more general, and three variations are given in [41].) The speed of these algorithms over general-purpose convex optimization is demonstrated in [41], and theoretical results

analogous to those for ordinary sparse approximation from [19, 61] are given in [24]. Also, an empirical study of iterative shrinkage approaches to solving mixed-norm formulations like (3.3) is given in [38].

Another recent work is a random-ensemble analysis of high-dimensional block-sparse problems in [55]. This work uses the convex relaxation of (3.3) and studies problem size scalings at which a correct sparsity pattern can be recovered in the undersampled ($M < NP$), noiseless ($\mathbf{d} = \mathbf{C}_{\text{tot}}\mathbf{h}_{\text{tot}}$ for some block-sparse $\mathbf{h}_{\text{tot}}$) case.

4. Proposed algorithms. We now present algorithms for (approximately) solving the MSSO problem defined in section 3. Some algorithms are described in more detail, with pseudocode, in [70, 72].

4.1. Matching pursuit (MP). The classic MP technique first finds the column of the system matrix that best matches with the observed vector and then removes from the observation vector the projection of this chosen column. It proceeds to select a second column of the system matrix that best matches with the *residual observation*, and continues doing so until either K columns have been chosen (as specified by the user) or the residual observation ends up as a vector of all zeros.

Now let us consider the MSSO system as posed in (3.6). In the MSSO context, we need to seek out which of the N $\mathbf{C}_n$ matrices can be best used to represent $\mathbf{d}$ when the columns of $\mathbf{C}_n$ undergo an arbitrarily weighted linear combination. The key difference here is that on an iteration-by-iteration basis, we are no longer deciding which column vector best represents the observation, but which *matrix* does so. For the kth iteration of the algorithm, we need to select the proper index $n \in \{1, \ldots, N\}$ by solving

$$q_k = \arg\min_n \min_{\mathbf{h}_n} \|\mathbf{r}_{k-1} - \mathbf{C}_n\mathbf{h}_n\|_2^2, \tag{4.1}$$

where q_k is the index that will be selected and $\mathbf{r}_{k-1}$ is the current residual observation. For fixed n, the solution to the inner minimization is obtained via the pseudoinverse, $\mathbf{h}_n^{\text{opt}} = \mathbf{C}_n^{\dagger}\mathbf{r}_{k-1}$, yielding

$$q_k = \arg\min_n \min_{\mathbf{h}_n} \|\mathbf{r}_{k-1} - \mathbf{C}_n(\mathbf{C}_n^{\dagger}\mathbf{r}_{k-1})\|_2^2 = \arg\max_n \mathbf{r}_{k-1}^{\text{H}}\mathbf{C}_n\mathbf{C}_n^{\dagger}\mathbf{r}_{k-1}, \tag{4.2}$$

where $^{\text{H}}$ is the Hermitian transpose. From (4.2) we see that, analogously to standard MP, choosing the best index for iteration k involves computing and ranking a series of inner product–like quadratic terms.

4.2. Orthogonal matching pursuit (OMP). In single-vector MP, the residual $\mathbf{r}_k$ is orthogonal to the kth column of the system matrix, but as the algorithm continues iterating, there is no guarantee that the residual remains orthogonal to column k or is minimized in the least squares sense with respect to the entire set of k chosen column vectors (indexed by $q_1, \ldots, q_k$). Furthermore, K iterations of single-vector MP do not guarantee that K different columns will be selected. Single-vector OMP is an extension to MP that attempts to mitigate these problems by improving the calculation of the residual vector. During the kth iteration of single-vector OMP, column q_k is chosen exactly as in MP (by ranking the inner products of the residual vector $\mathbf{r}_{k-1}$ with the various column vectors), but the residual vector is updated by accounting for *all* columns chosen up through iteration k rather than simply the last one [46, 10].

To extend OMP to the MSSO problem, we choose matrix q_k during iteration k as in MSSO MP and then, in the spirit of single-vector OMP, compute the new residual as follows:

$$\mathbf{r}_k = \mathbf{d} - \mathbf{S}_k(\mathbf{S}_k^{\dagger}\mathbf{d}), \tag{4.3}$$

where $\mathbf{S}_k = [\mathbf{C}_{q_1}, \ldots, \mathbf{C}_{q_k}]$ and $\mathbf{S}_k^{\dagger}\mathbf{d}$ is the best choice of $\mathbf{x}$ that minimizes the residual error $\|\mathbf{d} - \mathbf{S}_k\mathbf{x}\|_2$. That is, to update the residual we now employ all chosen matrices, weighting and combining them to best represent $\mathbf{d}$ in the least squares sense, yielding an $\mathbf{r}_k$ that is orthogonal to the columns of $\mathbf{S}_k$ (and thus orthogonal to $\mathbf{C}_{q_1}, \ldots, \mathbf{C}_{q_k}$), which has the benefit of ensuring that OMP will select a new $\mathbf{C}_n$ matrix at each step.

4.3. Least squares matching pursuit (LSMP). Beyond OMP there exists a greedy algorithm with greater computational complexity known as LSMP. In single-vector LSMP, one solves $N - k + 1$ least squares minimizations during iteration k to determine which column of the system matrix may be used to best represent $\mathbf{d}$ [10].

To extend LSMP to MSSO systems, we must ensure that during iteration k we account for the $k-1$ previously chosen $\mathbf{C}_n$ matrices when choosing the kth one to best construct an approximation to $\mathbf{d}$. Specifically, index q_k is selected as follows:

$$q_k = \underset{n \in \{1,\ldots,N\},\, n \notin I_{k-1}}{\arg\min} \; \min_{\mathbf{x}} \; \|\mathbf{S}_k^{(n)}\mathbf{x} - \mathbf{d}\|_2^2, \tag{4.4}$$

where I_{k-1} is the set of indices chosen up through iteration $k-1$, $\mathbf{S}_k^{(n)} = [\mathbf{S}_{k-1}, \mathbf{C}_n]$, $\mathbf{S}_{k-1} = [\mathbf{C}_{q_1}, \ldots, \mathbf{C}_{q_{k-1}}]$, and $\mathbf{x} \in \mathbb{C}^{Pk}$. For fixed n, the solution of the inner iteration is $\mathbf{x}^{\text{opt}} = (\mathbf{S}_k^{(n)})^{\dagger}\mathbf{d}$; it is this step that ensures the residual observation error will be minimized by using *all* chosen matrices. Substituting $\mathbf{x}^{\text{opt}}$ into (4.4) and simplifying the expression yields

$$q_k = \underset{n \notin I_{k-1}}{\arg\max} \; \mathbf{d}^{\mathrm{H}}\mathbf{Q}_k^{(n)}\mathbf{d}, \tag{4.5}$$

where $\mathbf{Q}_k^{(n)} = (\mathbf{S}_k^{(n)})(\mathbf{S}_k^{(n)})^{\dagger}$.

Algorithm 1 describes the LSMP method. The complexity here is much greater than that of OMP because $N - k + 1$ pseudoinversions of an $M \times Pk$ matrix are required during each iteration k. Furthermore, the dependence of $\mathbf{Q}_k^{(n)}$ on both n and k makes precomputing all such matrices infeasible in most cases. One way to decrease computation and runtime might be to extend the projection-based recursive updating scheme of [10] to MSSO LSMP.

Algorithm 1. MSSO Least Squares Matching Pursuit.

Task: greedily choose K of the $\mathbf{C}_n$s to best represent $\mathbf{d}$ via $\mathbf{C}_1\mathbf{h}_1 + \cdots + \mathbf{C}_N\mathbf{h}_N$.

Data and Parameters: $\mathbf{d}$ and $\mathbf{C}_n$, $n \in \{1, \ldots, N\}$, are given. K iterations.

Initialize: Set $k = 0$, $I_0 = \emptyset$, $\mathbf{S}_0 = [\,]$.

Iterate: Set $k = 1$ and apply:

- $q_k = \arg\max_{n \notin I_{k-1}} \; \mathbf{d}^{\mathrm{H}}(\mathbf{S}_k^{(n)})(\mathbf{S}_k^{(n)})^{\dagger}\mathbf{d}$, where $\mathbf{S}_k^{(n)} = [\mathbf{S}_{k-1}, \mathbf{C}_n]$
- $I_k = I_{k-1} \cup \{q_k\}$
- $\mathbf{S}_k = [\mathbf{S}_{k-1}, \mathbf{C}_{q_k}]$
- $k = k + 1$. Terminate loop if $k > K$ or $\mathbf{r}_k = \mathbf{0}$. I_K ends with $T \leq K$ elements.

Compute Weights: $\mathbf{x} = \mathbf{S}_K^{\dagger}\mathbf{d}$, unstack $\mathbf{x}$ into $\mathbf{h}_{q_1}, \ldots, \mathbf{h}_{q_T}$; set remaining $\mathbf{h}_n$s to $\mathbf{0}$.

4.4. Iteratively reweighted least squares (IRLS). Having posed three greedy approaches for solving the MSSO problem, we now turn our attention toward minimizing (3.7), the relaxed objective function. Here, the regularization term λ is used to trade off simultaneous sparsity with residual observation error.

One way to minimize (3.7) is to use an IRLS-based approach [36]. To begin, consider manipulating the right-hand term of (3.7) as follows:

(4.6)
$$\begin{aligned}
\lambda \sum_{n=1}^{N} \|\mathbf{h}_n\|_2 &= \lambda \sum_{n=1}^{N} \frac{\|\mathbf{h}_n\|_2^2}{\|\mathbf{h}_n\|_2} = \lambda \sum_{n=1}^{N} \frac{|\mathbf{h}_n[1]|^2 + \cdots + |\mathbf{h}_n[P]|^2}{\|\mathbf{h}_n\|_2} \\
&\approx \frac{\lambda}{2} \sum_{n=1}^{N} [\mathbf{h}_n^*[1], \ldots, \mathbf{h}_n^*[P]] \begin{bmatrix} \frac{2}{\|\mathbf{h}_n\|_2+\epsilon} & & \\ & \ddots & \\ & & \frac{2}{\|\mathbf{h}_n\|_2+\epsilon} \end{bmatrix} \begin{bmatrix} \mathbf{h}_n[1] \\ \vdots \\ \mathbf{h}_n[P] \end{bmatrix} \\
&= \frac{\lambda}{2} \sum_{n=1}^{N} \mathbf{h}_n^{\mathrm{H}} \mathbf{W}_n \mathbf{h}_n \\
&= \frac{\lambda}{2} \begin{bmatrix} \mathbf{h}_1^{\mathrm{H}} \cdots \mathbf{h}_N^{\mathrm{H}} \end{bmatrix} \begin{bmatrix} \mathbf{W}_1 & & \\ & \ddots & \\ & & \mathbf{W}_N \end{bmatrix} \begin{bmatrix} \mathbf{h}_1 \\ \vdots \\ \mathbf{h}_N \end{bmatrix} = \frac{\lambda}{2} \mathbf{h}_{\mathrm{tot}}^{\mathrm{H}} \mathbf{W}_{\mathrm{tot}} \mathbf{h}_{\mathrm{tot}},
\end{aligned}$$

where * is the complex conjugate of a scalar, $\mathbf{W}_n$ is a $P \times P$ real-valued diagonal matrix whose diagonal elements each equal $2/(\|\mathbf{h}_n\|_2 + \epsilon)$, and ϵ is some small nonnegative value introduced to mitigate poor conditioning of the $\mathbf{W}_n$s. If we fix $\mathbf{W}_{\mathrm{tot}} \in \mathbb{R}^{PN \times PN}$ by computing it using some prior estimate of $\mathbf{h}_{\mathrm{tot}}$, then the right-hand term of (3.7) becomes a quadratic function and (3.7) transforms into a Tikhonov optimization [58, 59]:

$$\min_{\mathbf{h}_{\mathrm{tot}}} \left\{ \tfrac{1}{2} \|\mathbf{d} - \mathbf{C}_{\mathrm{tot}} \mathbf{h}_{\mathrm{tot}}\|_2^2 + \tfrac{\lambda}{2} \mathbf{h}_{\mathrm{tot}}^{\mathrm{H}} \mathbf{W}_{\mathrm{tot}} \mathbf{h}_{\mathrm{tot}} \right\}. \tag{4.7}$$

Finally, by performing a change of variables and exploiting the properties of $\mathbf{W}_{\mathrm{tot}}$, we can convert (4.7) into an expression that may be minimized using the robust and reliable conjugate-gradient (CG) least squares solver LSQR [50, 49], so named because it applies a QR decomposition [30] when solving the system in the least squares sense. LSQR works better in practice than several other CG methods [2] because it restructures the input system via the Lanczos process [40] and applies a Golub–Kahan bidiagonalization [29] prior to solving it.

To apply LSQR to this problem, we first construct $\mathbf{W}_{\mathrm{tot}}^{1/2}$ as the element-by-element square root of the diagonal matrix $\mathbf{W}_{\mathrm{tot}}$ and then take its inverse to obtain $\mathbf{W}_{\mathrm{tot}}^{-1/2}$. Defining $\mathbf{q} = \mathbf{W}_{\mathrm{tot}}^{1/2} \mathbf{h}_{\mathrm{tot}}$ and $\mathbf{A} = \mathbf{C}_{\mathrm{tot}} \mathbf{W}_{\mathrm{tot}}^{-1/2}$, (4.7) becomes

$$\min_{\mathbf{q}} \left\{ \|\mathbf{d} - \mathbf{A}\mathbf{q}\|_2^2 + \lambda \|\mathbf{q}\|_2^2 \right\}. \tag{4.8}$$

LSQR is formulated to solve the exact problem in (4.8). Calling LSQR with $\mathbf{d}$, λ, and $\mathbf{A}$ yields $\mathbf{q}^{\mathrm{opt}}$, and the solution $\mathbf{h}_{\mathrm{tot}}^{\mathrm{opt}}$ is backed out via $\mathbf{W}_{\mathrm{tot}}^{-1/2} \mathbf{q}^{\mathrm{opt}}$.

Algorithm 2 outlines how one may iteratively apply (4.8) to attempt to find a solution that minimizes the original cost function, (3.7). The technique iterates until the objective function decreases by less than δ or the maximum number of iterations, K, is exceeded. The initial solution estimate is obtained via pseudoinversion of $\mathbf{C}_{\mathrm{tot}}$ (an all-zeros initialization would cause $\mathbf{W}_{\mathrm{tot}}$ to be poorly conditioned). A line search is used to step between the prior solution estimate and the upcoming one; this improves the rate of convergence and ensures that the objective decreases at each step. This method is global in that all PN unknowns are being estimated concurrently per iteration.

ALGORITHM 2. MSSO ITERATIVELY REWEIGHTED LEAST SQUARES.

Task: Minimize $\{\frac{1}{2}\|\mathbf{d}-\mathbf{C}_{\text{tot}}\mathbf{h}_{\text{tot}}\|_2^2+\lambda\sum_{n=1}^{N}\|\mathbf{h}_n\|_2\}$ using an iterative scheme.

Data and Parameters: λ, $\mathbf{d}$, $\mathbf{C}_{\text{tot}}$, δ, and K are given.

Initialize: Set $k=0$ and $\mathbf{h}_{\text{tot},k=0}=(\mathbf{C}_{\text{tot}})^{\dagger}\mathbf{d}$ (or, e.g., $\mathbf{h}_{\text{tot},k=0}=\mathbf{1}$).

Iterate: Set $k=1$ and apply:

- Create $\mathbf{W}_{\text{tot}}$ from $\mathbf{h}_{\text{tot},k-1}$; construct $\mathbf{W}_{\text{tot}}^{1/2}$, $\mathbf{W}_{\text{tot}}^{-1/2}$, and let $\mathbf{A}=\mathbf{C}_{\text{tot}}\mathbf{W}_{\text{tot}}^{-1/2}$.
- Obtain $\mathbf{q}_{\text{tmp}}$ by using LSQR to solve $\min_{\mathbf{q}}\{\|\mathbf{d}-\mathbf{A}\mathbf{q}\|_2^2+\lambda\|\mathbf{q}\|_2^2\}$.
- Set $\mathbf{h}_{\text{tot,tmp}}=\mathbf{W}_{\text{tot}}^{-1/2}\mathbf{q}_{\text{tmp}}$.
- Line search: find $\mu_0\in[0,1]$ such that $(1-\mu)\mathbf{h}_{\text{tot},k-1}+\mu\mathbf{h}_{\text{tot,tmp}}$ minimizes (3.7).
- Set $\mathbf{h}_{\text{tot},k}=(1-\mu_0)\mathbf{h}_{\text{tot},k-1}+\mu_0\mathbf{h}_{\text{tot,tmp}}$.
- $k=k+1$. Terminate loop when $k>K$ or (3.7) decreases by less than δ.

Finalize: Unstack the last $\mathbf{h}_{\text{tot}}$ solution into $\mathbf{h}_1,\ldots,\mathbf{h}_N$.

4.5. Row-by-row shrinkage (RBRS). The proposed IRLS technique solves for all PN unknowns during each iteration, but this is cumbersome when PN is large. An alternative approach is to apply an inner loop that fixes n and then iteratively tunes $\mathbf{h}_n$ while holding the other $\mathbf{h}_m$s ($m\neq n$) constant; thus only P (rather than PN) unknowns need to be solved for during each inner iteration.

This idea inspires the RBRS algorithm. The term "row-by-row" is used because each $\mathbf{h}_n$ corresponds to row n of the $\mathbf{G}$ matrix in (3.3), and "shrinkage" is used because the ℓ_2 energy of most of the $\mathbf{h}_n$'s will essentially be "shrunk" (to some extent) during each inner iteration: when λ is sufficiently large and many iterations are undertaken, many $\mathbf{h}_n$'s will be close to all-zeros vectors.

4.5.1. RBRS for real-valued data. Assume that $\mathbf{d}$ and the $\mathbf{C}_n$'s of (3.7) are real-valued. We seek to minimize the function by extending the single-vector sequential shrinkage technique of [21] and making modifications to (3.7). Assume that we have prior estimates of $\mathbf{h}_1$ through $\mathbf{h}_N$, and that we now desire to update only the jth vector while keeping the other $N-1$ fixed. The shrinkage update of $\mathbf{h}_j$ is achieved via

$$\min_{\mathbf{x}}\ \left\{\tfrac{1}{2}\left\|\left[\Sigma_{n=1}^{N}\mathbf{C}_n\mathbf{h}_n-\mathbf{C}_j\mathbf{h}_j\right]+\mathbf{C}_j\mathbf{x}-\mathbf{d}\right\|_2^2+\lambda\|\mathbf{x}\|_2\right\}, \tag{4.9}$$

where $\Sigma_{n=1}^{N}\mathbf{C}_n\mathbf{h}_n-\mathbf{C}_j\mathbf{h}_j$ forms an approximation of $\mathbf{d}$ using the prior solution coefficients, but discards the component contributed by the original jth vector, replacing the latter via an updated solution vector, $\mathbf{x}$. It is crucial to note that the right-hand term does not promote the element-by-element sparsity of $\mathbf{x}$; rather, it penalizes the overall ℓ_2 energy of $\mathbf{x}$, and thus both sparse and dense $\mathbf{x}$'s are penalized equally if their overall ℓ_2 energies are the same.

One way to solve (4.9) is to take its derivative with respect to $\mathbf{x}^{\mathrm{T}}$ and find $\mathbf{x}$ such that the derivative equals $\mathbf{0}$. By doing this and shuffling terms, and assuming we have an initial estimate of $\mathbf{x}$, we may solve for $\mathbf{x}$ iteratively:

$$\mathbf{x}_i=\left[\mathbf{C}_j^{\mathrm{T}}\mathbf{C}_j+\frac{\lambda}{\|\mathbf{x}_{i-1}\|_2+\epsilon}\mathbf{I}\right]^{-1}\mathbf{C}_j^{\mathrm{T}}\mathbf{r}_j, \tag{4.10}$$

where $\mathbf{r}_j = \mathbf{d} + \mathbf{C}_j\mathbf{h}_j - \Sigma_{n=1}^{N}\mathbf{C}_n\mathbf{h}_n$, $\mathbf{I}$ is a $P \times P$ identity matrix, and ϵ is a small value that avoids ill-conditioned results.[3] By iterating on (4.10) until (4.9) changes by less than δ_{inner}, we arrive at a solution to (4.9), $\mathbf{x}^{\text{opt}}$, and this then replaces the prior solution vector, $\mathbf{h}_j$. Having completed the update of the jth vector, we proceed to update the rest of the vectors, looping this outer process K times or until the main objective function, (3.7), changes by less than δ_{outer}. Algorithm 3 details the entire procedure; unlike IRLS, here we essentially repeatedly invert $P \times P$ matrices to pursue a row-by-row solution, rather than $PN \times PN$ matrices to pursue a solution that updates *all* rows per iteration.

ALGORITHM 3. MSSO ROW-BY-ROW SEQUENTIAL ITERATIVE SHRINKAGE.

Task: Minimize $\{\frac{1}{2}\|\mathbf{d} - \mathbf{C}_{\text{tot}}\mathbf{h}_{\text{tot}}\|_2^2 + \lambda\sum_{n=1}^{N}\|\mathbf{h}_n\|_2\}$ using an iterative scheme when all data is *real-valued*.

Data and Parameters: λ, $\mathbf{d}$, $\mathbf{C}_n$ ($n \in \{1, \ldots, N\}$), δ_{outer}, δ_{inner}, K, and I are given.

Initialize: Set $k = 0$ and $\mathbf{h}_{\text{tot}} = (\mathbf{C}_{\text{tot}})^{\dagger}\mathbf{d}$ (or, e.g., $\mathbf{h}_{\text{tot}} = \mathbf{1}$), unstack into $\mathbf{h}_1, \ldots, \mathbf{h}_N$.

Iterate: Set $k = 1$ and apply:

- Sweep over row vectors: set $j = 1$ and apply:
  - Optimize a row vector: set $i = 1$ and $\mathbf{x}_0 = \mathbf{h}_j$ and then apply:
    - $\mathbf{x}_i = [\mathbf{C}_j^{\mathrm{T}}\mathbf{C}_j + \frac{\lambda}{\|\mathbf{x}_{i-1}\|_2^{+\epsilon}}\mathbf{I}]^{-1}\mathbf{C}_j^{\mathrm{T}}\mathbf{r}_j$, where $\mathbf{r}_j = \mathbf{d} + \mathbf{C}_j\mathbf{h}_j - \Sigma_{n=1}^{N}\mathbf{C}_n\mathbf{h}_n$.
    - $i = i + 1$. Terminate when $i > I$ or (4.9) decreases by less than δ_{inner}.
  - Finalize row vector update: set $\mathbf{h}_j$ to equal the final $\mathbf{x}$.
  - $j = j + 1$. Terminate loop when $j > N$.
- $k = k + 1$. Terminate loop when $k > K$ or (3.7) decreases by less than δ_{outer}.

Finalize: If λ was large enough, several $\mathbf{h}_n$s should be dense and others close to $\mathbf{0}$.

4.5.2. Extending RBRS to complex-valued data. If (3.7) contains complex-valued terms, we may structure the row-by-row updates as in (4.9), but because the derivative of the objective function in (4.9) is more complicated due to the presence of complex-valued terms, the simple update equation given in (4.10) is no longer applicable. One way to overcome this problem is to turn the complex-valued problem into a real-valued one.

Let us create several real-valued expanded vectors,

$$\widetilde{\mathbf{d}} = \begin{bmatrix} \mathrm{Re}(\mathbf{d}) \\ \mathrm{Im}(\mathbf{d}) \end{bmatrix} \in \mathbb{R}^{2M}, \quad \widetilde{\mathbf{h}}_n = \begin{bmatrix} \mathrm{Re}(\mathbf{h}_n) \\ \mathrm{Im}(\mathbf{h}_n) \end{bmatrix} \in \mathbb{R}^{2P}, \tag{4.11}$$

as well as real-valued expanded matrices,

$$\widetilde{\mathbf{C}}_n = \begin{bmatrix} \mathrm{Re}(\mathbf{C}_n) & -\mathrm{Im}(\mathbf{C}_n) \\ \mathrm{Im}(\mathbf{C}_n) & \mathrm{Re}(\mathbf{C}_n) \end{bmatrix} \in \mathbb{R}^{2M \times 2P}. \tag{4.12}$$

Due to the structure of (4.11), (4.12) and the fact that $\|\mathbf{h}_n\|_2$ equals $\|\widetilde{\mathbf{h}}_n\|_2$, the

[3] Equation (4.10) consists of a direct inversion of a $P \times P$ matrix, which is acceptable in this paper because all experiments involve $P \leq 10$. If P is large, (4.10) could be solved via a CG technique (e.g., LSQR).

following optimization is equivalent to (3.7):

$$\min_{\widetilde{\mathbf{h}}_1,\ldots,\widetilde{\mathbf{h}}_N} \left\{ \tfrac{1}{2} \left\| \widetilde{\mathbf{d}} - \sum_{n=1}^{N} \widetilde{\mathbf{C}}_n \widetilde{\mathbf{h}}_n \right\|_2^2 + \lambda \sum_{n=1}^{N} \|\widetilde{\mathbf{h}}_n\|_2 \right\}. \tag{4.13}$$

This means that we may apply RBRS to complex-valued scenarios by substituting the $\widetilde{\mathbf{h}}_n$'s for the $\mathbf{h}_n$'s and $\widetilde{\mathbf{C}}_n$'s for the $\mathbf{C}_n$'s in (4.9), (4.10), and Algorithm 3. (Equation (4.10) becomes an applicable update equation because (4.9) will consist of only real-valued terms and the derivative calculated earlier is again applicable.) Finally, after running the algorithm to obtain finalized $\widetilde{\mathbf{h}}_n$'s, we may simply restructure them into complex $\mathbf{h}_n$'s.

In embedding the complex-valued problem into a larger real-valued problem, we have demonstrated a fact about simultaneous sparse approximation that seems to not have been remarked upon previously: by setting $P = 1$, we see that seeking a single sparse complex-valued vector is equivalent to seeking two *simultaneously sparse* real-valued vectors.

4.6. Column-by-column shrinkage (CBCS). We have also developed a dual of RBRS—a technique that sequentially updates the *columns* of $\mathbf{G}$ (i.e., the $\mathbf{g}_p$'s) in (3.1), (3.3) rather than its rows (the $\mathbf{h}_n$'s). This approach yields a *separable* optimization and reduces the overall problem to simply repeated *element-by-element* shrinkages of each $\mathbf{g}_p$. For a detailed derivation and pseudocode, see [70, 72].

4.7. Second-order cone programming (SOCP). We now propose a seventh and final algorithm for solving the MSSO problem as given in (3.3). We branch away from the shrinkage approaches that operate on individual columns or rows of the $\mathbf{G}$ matrix and again seek to concurrently estimate all PN unknowns. Rather than using an IRLS technique, however, we pursue an SOCP approach, motivated by the fact that second-order cone programs may be solved via efficient interior point algorithms [56, 60] and are able to encapsulate conic, convex-quadratic [1], and linear constraints. (Quadratic programming is not an option because the $\mathbf{g}_p$'s, $\mathbf{F}_p$'s, and $\mathbf{d}$ may be complex.)

Second-order conic constraints are of the form $\mathbf{a} = [a_1, \ldots, a_N]^{\mathrm{T}}$ such that

$$\|[a_1, \ldots, a_{N-1}]^{\mathrm{T}}\|_2 \leq a_N. \tag{4.14}$$

The generic format of an SOC program is

$$\min_{\mathbf{x}} \ \mathbf{c}^{\mathrm{T}}\mathbf{x} \quad \text{s.t.} \quad \mathbf{A}\mathbf{x} = \mathbf{b} \quad \text{and} \quad \mathbf{x} \in \mathbf{K}, \tag{4.15}$$

where $\mathbf{K} = \mathbb{R}_+^N \times \mathbf{L}_1 \times \cdots \times \mathbf{L}_N$, $\mathbb{R}_+^N$ is the N-dimensional positive orthant cone and the $\mathbf{L}_n$s are second-order cones [1]. To convert (3.3) into the second-order cone format, we first write

$$\begin{aligned} &\min_{\mathbf{G}} \ \left\{ \tfrac{1}{2} s + \lambda \mathbf{1}^{\mathrm{T}} \mathbf{t} \right\} \quad \text{s.t.} \quad \mathbf{z} = \mathbf{d}_{\mathrm{tot}} - \mathbf{F}_{\mathrm{tot}} \mathbf{g}_{\mathrm{tot}}, \quad \|\mathbf{z}\|_2^2 \leq s, \\ &\text{and} \quad \|[\mathrm{Re}(\mathbf{g}_1[n]), \mathrm{Im}(\mathbf{g}_1[n]), \ldots, \mathrm{Re}(\mathbf{g}_P[n]), \mathrm{Im}(\mathbf{g}_P[n])]^{\mathrm{T}}\|_2 \leq t_n, \end{aligned} \tag{4.16}$$

where $n \in \{1, \ldots, N\}$ and $\mathbf{t} = [t_1, \ldots, t_N]^{\mathrm{T}}$. The splitting of the complex elements of the $\mathbf{g}_p$'s mimics the approach used when extending CBCS to complex data, and (4.16) makes the objective function linear, as required. Finally, in order to represent the $\|\mathbf{z}\|_2^2 \leq s$ inequality in terms of second-order cones, an additional step is needed. Given

that $s = \frac{1}{4}(s+1)^2 - \frac{1}{4}(s-1)^2$, the inequality may be rewritten as $\mathbf{z}^{\mathrm{H}}\mathbf{z} + \frac{1}{4}(s-1)^2 \leq \frac{1}{4}(s+1)^2$ and then expressed as a conic constraint: $\|[\mathbf{z}^{\mathrm{T}}, \frac{1}{2}(s-1)]^{\mathrm{T}}\|_2 \leq \frac{1}{2}(s+1)$ [1, 42]. Applying these changes yields

$$\begin{aligned} &\min \left\{\tfrac{1}{2}s + \lambda \mathbf{1}^{\mathrm{T}}\mathbf{t}\right\} \quad \text{s.t.} \quad \mathbf{z} = \mathbf{d}_{\mathrm{tot}} - \mathbf{F}_{\mathrm{tot}}\mathbf{g}_{\mathrm{tot}}, \\ &\|[\mathbf{z}^{\mathrm{T}}, u]^{\mathrm{T}}\|_2 \leq v, \quad u = (s-1)/2, \quad v = (s+1)/2, s \geq 0, \quad \text{and} \\ &\|[\mathrm{Re}(\mathbf{g}_1[n]), \mathrm{Im}(\mathbf{g}_1[n]), \ldots, \mathrm{Re}(\mathbf{g}_P[n]), \mathrm{Im}(\mathbf{g}_P[n])]^{\mathrm{T}}\|_2 \leq t_n, \end{aligned} \tag{4.17}$$

which is a fully defined second-order cone program that may be implemented and solved numerically. There is no algorithm pseudocode for this technique because having set up the variables in (4.17), one may simply plug them into an SOCP solver. In this paper we implement (4.17) in SeDuMi (Self-Dual-Minimization) [56], a free software package consisting of MATLAB and `C` routines.

5. Experiments and results. Our motivation for solving MSSO sparse approximation problems comes from MRI RF excitation pulse design. Before turning to this problem in section 5.3, we present several synthetic experiments. These experiments allow comparisons among algorithms and also empirically reveal some properties of the relaxation (3.3). Theoretical exploration of this relaxation is also merited but is beyond the scope of this manuscript. One work in this area is [55].

Experiments were performed on a Linux server with a 3.0-GHz Intel Pentium IV processor. The system has 16 GB of random access memory, ample to ensure that none of the algorithms requires the use of virtual memory and to avoid excessive hard drive paging. MP, LSMP, IRLS, RBRS, and CBCS are implemented in MATLAB, whereas SOCP is implemented in SeDuMi. The runtimes could be reduced significantly by implementing the methods in a completely compiled format such as `C`. Note: OMP is not evaluated because its performance always falls in between that of MP and LSMP.

5.1. Sparsity pattern estimation in a noiseless setting.

5.1.1. Overview. We now evaluate how well the algorithms of section 4 estimate sparsity patterns when the underlying $\mathbf{g}_p$'s are each strictly and simultaneously K-sparse and the observation $\mathbf{d}$ of (3.1) is not corrupted by noise. This caricatures a high signal-to-noise ratio (SNR) source localization scenario, where the sparsity pattern indicates locations of emitters and our goal is to find the locations of these emitters [34, 39, 42, 43].

We synthetically generate real-valued sets of $\mathbf{F}_p$'s and $\mathbf{g}_p$'s in (3.1), apply the algorithms, and record the fraction of correct sparsity pattern entries recovered by each. We vary M in (3.1) to see how performance at solving the MSSO problem varies when the $\mathbf{F}_p$'s are underdetermined vs. overdetermined and also vary P to see how rapidly performance degrades as more system matrices and vectors are employed.

5.1.2. Details. For all trials, we fix $N = 30$ in (3.1) and $K = 3$, which means each $\mathbf{g}_p$ vector consists of 30 elements, three of which are nonzero. We consider $P \in \{1, 2, \ldots, 8\}$ and $M \in \{10, 15, \ldots, 40\}$. For each of the 56 fixed (M, P) pairs, we create 50 random instances of (3.1) over which to report the average performance. Each of the 2 800 instances is constructed with the sparsity pattern chosen uniformly at random and the nonzero entries of the $\mathbf{g}_p$'s drawn from the standard normal $\mathcal{N}(0, 1)$ distribution. The columns of the $\mathbf{F}_p$'s are chosen independently from the uniform distribution on the $\mathbb{R}^M$ unit sphere.

MP and LSMP are applied by iterating until K elements are chosen or the residual approximation is $\mathbf{0}$. If fewer than K terms are chosen, this hurts the recovery score.

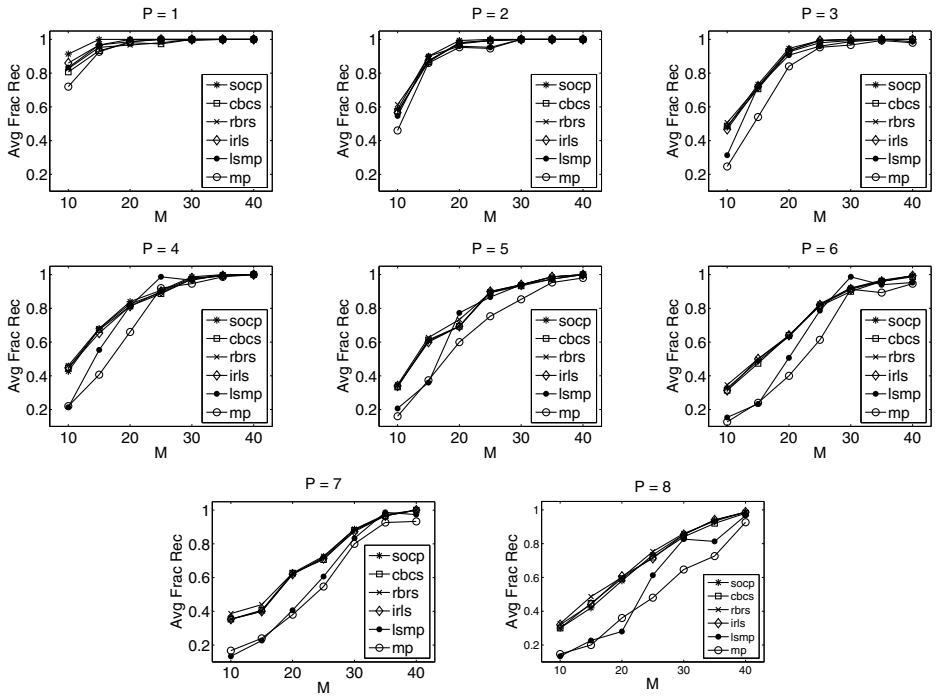

FIG. 5.1. Sparsity pattern estimation in a noiseless setting. *Subplots depict average fraction of sparsity pattern elements recovered over* 50 *trials of six algorithms as* M *is varied.* P *is fixed per subplot, and* $N = 30$ *and* $K = 3$ *for all trials. Data is generated as described in section* 5.1.2. *Recovery scores for IRLS, RBRS, CBCS, and SOCP assume that a good choice of* λ *is known. For large* M*, all algorithms exhibit high recovery rates; for large* P*, small* M*, or both, the algorithms that seek to minimize* (3.3), (3.7) *generally outperform those that greedily pursue a solution.*

For IRLS, RBRS, CBCS, and SOCP, we approximate a λ oracle as follows: loop over roughly 70 λ's in $[0, 2]$, running the given algorithm each time. For each of the estimated $\widehat{\mathbf{g}}_{\text{tot}}$'s (that vary with λ), estimate a sparsity pattern by noting the largest ℓ_2 energy rows of the associated $\widehat{\mathbf{G}}$ matrix.[4] Remember the highest fraction recovered across all λ's.

5.1.3. Results. Each subplot of Figure 5.1 depicts the average fraction of recovered sparsity pattern elements vs. the number of knowns, M, for a fixed value of P, revealing how performance varies as the $\mathbf{F}_p \in \mathbb{R}^{M \times N}$ matrices go from being underdetermined to overdetermined.

Recovery trends. As the number of knowns M increases, recovery rates improve substantially, which is sensible. For large M and small P, the six algorithms behave similarly, consistently achieving nearly 100% recovery. For large P and moderate M, however, sparsity pattern recovery rates are dismal—as P increases, the underlying simultaneous sparsity of the $\mathbf{g}_p$'s is not enough to combat the increasing number of unknowns, PN. As M is decreased and especially when P is increased, the performance of the greedy techniques falls off relative to that of IRLS, RBRS, CBCS, and SOCP, showing that the convex relaxation approach itself is a sensible way to approximately solve the formal NP-hard combinatorial MSSO simultaneous sparsity

[4]This could be described as thresholding the output of the original computation.

TABLE 5.1

Average algorithm runtimes for noiseless sparsity pattern estimation. *For several fixed* (M, P) *pairs, each algorithm's average runtime over the corresponding* 50 *trials is given in units of milliseconds;* $N = 30$ *and* $K = 3$ *for all trials (runtimes of the latter four algorithms are also averaged over the multiple* λ *runs per trial). MP is substantially faster than the other techniques, as expected. For larger problems, e.g.,* $(M, P) = (10, 8)$*, the runtimes of both RBRS and CBCS are excessive relative to those of the other convex minimization techniques, IRLS and SOCP.*

	(M, P)			
Algorithm	(10,8)	(20,1)	(30,5)	(40,8)
MP	5.4	1.8	2.6	4.0
LSMP	11.4	5.6	15.6	27.6
IRLS	92.6	10.1	73.2	175.0
RBRS	635.7	36.0	236.8	401.6
CBCS	609.8	7.1	191.4	396.3
SOCP	44.3	37.0	64.3	106.5

problem. Furthermore, the behavior of the convex algorithms relative to the greedy ones coincides with the studies of greedy vs. convex programming sparse approximation methods in single-vector [7, 10] and SSMO contexts [11]. LSMP tends to perform slightly better than MP because it solves a least squares minimization and explicitly considers earlier chosen rows whenever it seeks to choose another row of $\mathbf{G}$.

Convergence. Across most trials, IRLS, RBRS, CBCS, and SOCP converge rapidly and do not exceed the maximum limit of 500 outer iterations. The exception is CBCS when M is small and $P = 8$: here, the objective function frequently fails to decrease by less than the specified $\delta = 10^{-5}$.

Runtimes. For several fixed (M, P) pairs, Table 5.1 lists the average runtimes of each algorithm across the 50 trials associated with each pair. For IRLS, RBRS, CBCS, and SOCP, runtimes are also averaged over the many λ runs. Among the convex minimization methods, SOCP seems superior given its fast runtimes in three out of four cases. Peak memory usage is not tracked here because it is difficult to do so when using MATLAB for such small problems; it will be tracked during the third experiment where the system matrices are vastly larger and differences in memory usage across the six algorithms are readily apparent.

Some details on how IRLS, RBRS, CBCS, and SOCP differ in the objective functions that they achieve are provided in [70, 72].

5.2. Sparsity pattern estimation in the presence of noise.

5.2.1. Overview. We now modify the scenario of section 5.1 to include additive white Gaussian noise in the observation $\mathbf{d}$. The simultaneous sparsity level K and SNR (in dB) are varied across sets of Monte Carlo trials. The variance of each component of the noise is related to the SNR as follows:

$$\sigma^2 = \frac{1}{M} \|\mathbf{d}_{\text{true}}\|_2^2 \cdot 10^{-\text{SNR}/10}. \tag{5.1}$$

This noise measure is analogous to that of [11].

5.2.2. Details. We fix $N = 30$, $M = 25$, and $P = 3$, and consider $\text{SNR} \in \{-10, -5, 0, \ldots, 25, 30\}$ and $K \in \{1, 3, 5, 7, 9\}$. For each fixed (SNR, K) pair, we generate 100 random instances over which to report the average performance. The $\mathbf{g}_p$'s and $\mathbf{F}_p$'s are generated as in section 5.1.2. The algorithms are applied as before, with the exception that IRLS, RBRS, CBCS, and SOCP use a fixed λ as described below.

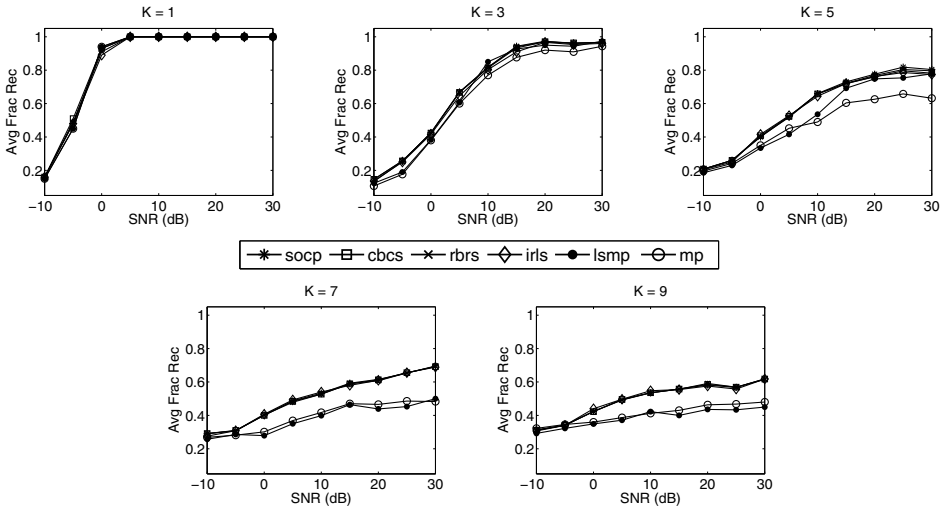

FIG. 5.2. Sparsity pattern estimation in the presence of noise. *Each subplot depicts the average fraction of recovered sparsity pattern elements vs. SNR for a fixed K, revealing how well the algorithms recover the K elements of the sparsity pattern amidst noise in the observation. Each data point is the average fraction recovered across* 100 *trials; data is randomly generated as described in section* 5.2.2. *N, M, and P are always fixed at* 30, 25, *and* 3, *respectively. For each (SNR, K) pair, a "good" λ is chosen by denoising a few cases by hand and then using this fixed λ for* 100 *fresh denoising trials. Performance degrades with increasing K and decreasing SNR. For large K, the greedy algorithms perform worse than IRLS, SOCP, RBRS, and CBCS, whereas the latter four methods perform essentially identically across all (SNR, K) combinations.*

Control parameter selection. Before running the overall experiment, we generate three noisy observations for each (SNR, K) pair. We then apply IRLS, RBRS, CBCS, and SOCP, tuning the control parameter λ by hand until finding a single value that produces reasonable solutions. All algorithms then use this fixed λ for all 100 trials with the (SNR, K) pair under consideration. Thus, in contrast to the noiseless experiment, we no longer assume an ideal λ is known for each denoising trial.

5.2.3. Results. Each subplot of Figure 5.2 depicts the average fraction of recovered sparsity pattern elements vs. SNR for a fixed K, revealing how well the six algorithms are able to recover the sparsity pattern amid noise.

Recovery trends. When $K = 1$, we see from the upper-left subplot of Figure 5.2 that all algorithms have essentially equal performance for each SNR. Recovery rates improve substantially with increasing SNR, which is sensible. For each algorithm, we see across the subplots that performance generally decreases with increasing K; this too is as expected. For low SNRs, e.g., -5 dB, all algorithms tend to perform similarly, but the greedy algorithms perform increasingly worse than the others as K goes from moderate to large and SNR surpasses zero dB. In general, MP performs worse than LSMP, and LSMP in turn performs worse than IRLS, SOCP, RBRS, and CBCS, while the latter four methods exhibit essentially the same performance across all SNRs and K's. Overall, Figure 5.2 shows that convex programming algorithms are superior to greedy methods when estimating sparsity patterns in noisy situations; this coincides with data collected in the noiseless experiment in section 5.1, as well as the empirical findings of [10, 11].

Convergence. CBCS typically requires more iterations than the other techniques

in order to converge. At times, it fails to converge to within the specified $\delta = 10^{-5}$, similarly to how it behaves during the noiseless experiment of section 5.1.

Runtimes. Across all denoising trials, MP, LSMP, IRLS, RBRS, CBCS, and SOCP have average runtimes of 3.1, 25.1, 57.2, 247.0, 148.5, and 49.2 milliseconds. It seems that SOCP is best for denoising given that it is the fastest algorithm among the four methods that outperform the greedy ones. IRLS is nearly as fast as SOCP and thus is a close second choice for sparsity pattern estimation.

This experiment is extended with a discussion of mean-squared error of the estimates in [70, 72].

5.3. MRI RF excitation pulse design.

5.3.1. Overview. For the final experiment we study how well the six algorithms design MRI RF excitation pulses. In the interest of space and because the conversion of the physical problem into an MSSO format involves MRI physics and requires significant background, we only briefly outline how the system matrices arise and why simultaneously sparse solutions are necessary. A complete formulation of the problem for engineers and mathematicians is given in [70, 71]; MRI pulse designers may refer to [74]. We limit our evaluation here to fidelity of the designed excitations. Related papers provide results from a real system for head-shaped water phantom imaging [74] and in vivo human brain imaging [73].

5.3.2. Formulation. Consider an MRI experiment in which thin slices are desired in a spatial dimension defined to be the z direction. Thin-slice imaging is a dominant use of RF excitation in clinical MRI. For the purposes of this paper, the design of an MRI RF excitation pulse reduces to the following problem: assume that we are given M points in the two-dimensional (2-D) (x, y) spatial domain, $\mathbf{r}_1, \ldots, \mathbf{r}_M$, along with N points in a 2-D "Fourier-like" domain, $\mathbf{k}_1, \ldots, \mathbf{k}_N$. Each $\mathbf{r}_m$ equals $[x_m, y_m]^{\mathrm{T}}$, a point in space, while each $\mathbf{k}_n$ equals $[k_{x,n}, k_{y,n}]^{\mathrm{T}}$, a point in the Fourier-like domain, referred to as "k-space." The $\mathbf{r}_m$'s and $\mathbf{k}_n$'s are in units of centimeters (cm) and inverse centimeters (cm^{-1}), respectively. The $\mathbf{k}_n$'s are Nyquist-spaced relative to the sampling of the $\mathbf{r}_m$'s and may be visualized as a 2-D grid located at low k_x and k_y frequencies (where "k_x" denotes the frequency domain axis that corresponds to the spatial x axis). Under a small-tip-angle approximation, energy placed at one or more points in k-space produces a pattern in the spatial domain; this pattern is related to the k-space energy via a "Fourier-like" transform [52]. Assume that we place an arbitrary complex weight $g_n \in \mathbb{C}$ (i.e., both a magnitude and phase) at each of the N locations in k-space. Let us represent these weights using a vector $\mathbf{g} = [g_1, \ldots, g_N]^{\mathrm{T}} \in \mathbb{C}^N$. In an ideal setting (i.e., using the small-tip-angle approximation [52] as mentioned above and neglecting coil transmission profiles), the following holds:

$$\mathbf{m} = \mathbf{A}\mathbf{g}, \tag{5.2}$$

where $\mathbf{A} \in \mathbb{C}^{M \times N}$ is a known dense Fourier matrix[5] and the mth element of $\mathbf{m} \in \mathbb{C}^M$ is the image that arises at $\mathbf{r}_m$, denoted $m(\mathbf{r}_m)$, due to the energy deposition along the N points on the k-space grid as described by the weights in the $\mathbf{g}$ vector. The energy deposition in the k_z dimension is along sinc-like pulse segments that do not enter our design process because the z dependence and (x, y) dependence of the desired excitation are separable. A formulation that considers three dimensions more generally is presented in [70, sect. 5.2].

[5] Formally, $\mathbf{A}(m, n) = j\gamma e^{i\mathbf{r}_m \cdot \mathbf{k}_n}$, where $j = \sqrt{-1}$ and γ is a known lumped gain constant.

The goal now is to form a desired (possibly complex-valued) spatial-domain image $d(\mathbf{r})$ at the given set of spatial-domain coordinates (the $\mathbf{r}_m$'s) by placing energy at some of the given $\mathbf{k}_n$ locations while obeying a special constraint on how the energy is deposited. To produce the spatial-domain image, we will use a "P-channel MRI parallel excitation system" [37, 54]—each of the system's P channels is able to deposit energy of varying magnitudes and phases at the k-space locations and is able to influence the resulting spatial-domain pattern $m(\mathbf{r})$ to some extent. Each channel p has a known "profile" across space, $S_p(\mathbf{r}) \in \mathbb{C}$, that describes how the channel influences the magnitude and phase of the resulting image at different spatial locations. For example, if $S_3(\mathbf{r}_5) = 0$, then the third channel is unable to influence the image that arises at location $\mathbf{r}_5$, regardless of how much energy it deposits along $\mathbf{k}_1, \ldots, \mathbf{k}_N$. The special constraint mentioned above is that *the system's channels may visit only a small number of points in k-space—they may deposit energy only at $K \ll N$ locations.*

We now finalize the formulation: first, we construct P diagonal matrices $\mathbf{S}_p \in \mathbb{C}^{M \times M}$ such that $\mathbf{S}_p(m,m) = S_p(\mathbf{r}_m)$, $m = 1, \ldots, M$. Now we assume that each channel deposits arbitrary energies at each of the N points in k-space and describe the weighting of the k-space grid by the pth channel with the vector $\mathbf{g}_p \in \mathbb{C}^N$. Based on the physics of the P-channel parallel excitation system, the overall image $m(\mathbf{r})$ that forms is the *superposition* of the profile-scaled subimages produced by each channel:

$$\mathbf{m} = \mathbf{S}_1\mathbf{A}\mathbf{g}_1 + \cdots + \mathbf{S}_P\mathbf{A}\mathbf{g}_P = \mathbf{F}_1\mathbf{g}_1 + \cdots + \mathbf{F}_P\mathbf{g}_P = \mathbf{F}_{\text{tot}}\mathbf{g}_{\text{tot}}, \tag{5.3}$$

where $\mathbf{m} = [m(\mathbf{r}_1), \ldots, m(\mathbf{r}_M)]^{\mathrm{T}}$. Essentially, (5.3) refines the model of (5.2) by incorporating multiple excitation channels and accounting for coil transmission profiles $\{S_p(\mathbf{r})\}_{p=1}^P$ that are not necessarily constant across $\mathbf{r}$.

Recalling that our overall goal is to deposit energy in k-space to produce the image $d(\mathbf{r})$, and given the special constraint that we may deposit energy only among a small subset of the N points in k-space, we arrive at the following problem:

$$\min_{\mathbf{g}_1, \ldots, \mathbf{g}_P} \|\mathbf{d} - \mathbf{m}\|_2 \quad \text{s.t. the simultaneous } K\text{-sparsity of the } \mathbf{g}_p\text{'s}, \tag{5.4}$$

where $\mathbf{d} \in \mathbb{C}^M = [d(\mathbf{r}_1), \ldots, d(\mathbf{r}_M)]^{\mathrm{T}} \in \mathbb{C}^M$ and $\mathbf{m}$ is given by (5.3). That is, we seek out $K < N$ locations in k-space at which to deposit energy to produce an image $m(\mathbf{r})$ that is close in the ℓ_2 sense to the desired image $d(\mathbf{r})$. Strictly and simultaneously K-sparse $\mathbf{g}_p$'s are the only valid solutions to the problem.

One sees that (5.4) is precisely the MSSO system given in (3.2), and thus the algorithms posed in section 4 are applicable to the pulse design problem. In order to apply the convex minimization techniques (IRLS, SOCP, RBRS, and CBCS) to this problem, the only additional step needed is to retune any given solution estimate $\widehat{\mathbf{g}}_{\text{tot}}(\text{alg}, \lambda)$ into a strictly and simultaneously K-sparse set of vectors.

Aside. An alternative approach to deciding where to place energy at K locations in k-space is to compute the Fourier transform of $d(\mathbf{r})$ and decide to place energy at (k_x, k_y) frequencies where the Fourier coefficients are largest in magnitude [69]. This method does yield valid K-sparse energy placement patterns, but empirically it is always outperformed by the convex minimization approaches [73, 71, 74]; thus we do not delve into the Fourier-based method in this paper.

5.3.3. Experimental setup. Using an eight-channel system (i.e., $P = 8$) whose profile magnitudes (the $S_p(\mathbf{r})$'s) are depicted in Figure 5.3, we will design pulses to produce the desired image shown in the left subplot of Figure 5.4. We sample the spatial (x, y) domain at $M = 356$ locations within the region where at least one of

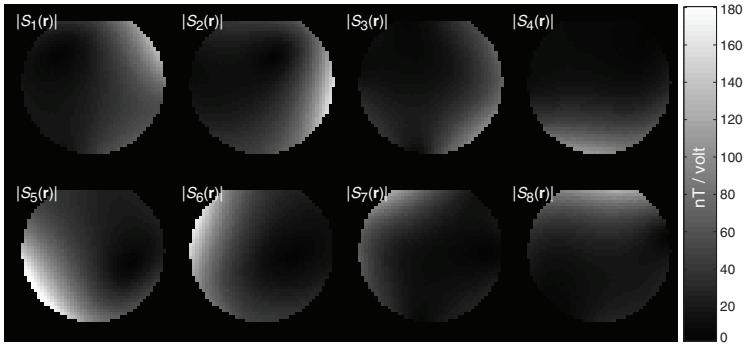

FIG. 5.3. Profile magnitudes of the eight-channel parallel excitation MRI system. *Here the magnitudes of the $S_p(\mathbf{r})$s are depicted for $p = 1, \ldots, 8$; 356 samples of each $S_p(\mathbf{r})$ are taken within the nonzero region of influence and stacked into the diagonal matrix $\mathbf{S}_p$ used in* (5.3). *Across space, the $S_p(\mathbf{r})$s are not orthogonal—their regions of influence overlap each other to some extent.*

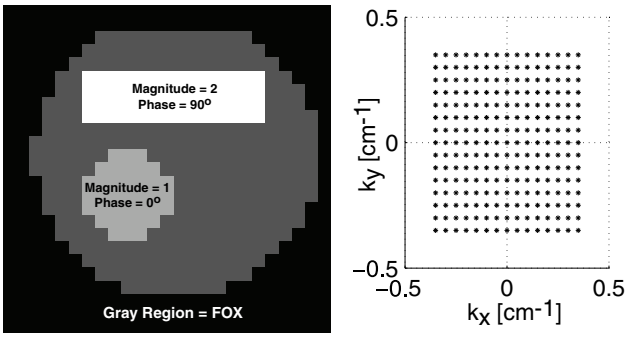

FIG. 5.4. Desired image and k-space grid. *Left image: desired complex-valued image, $d(\mathbf{r})$. Medium-gray region = FOX; other regions denote locations where we want image to be nonzero with the given magnitudes and phases. Sampling $d(\mathbf{r})$ at the 356 locations within FOX allows us to construct $\mathbf{d}$ in* (5.3). *Right subplot: 15×15 grid of $N = 225$ candidate k-space locations, $\mathbf{k}_1, \ldots, \mathbf{k}_{225}$, at which the P channels may deposit energy and thus influence the resulting image. The physical constraints of the MRI excitation process force us to place energy at only a small number of grid locations.*

the profiles in Figure 5.3 is active; this region of interest is referred to as the *field of excitation* (FOX) in MRI literature.[6] The spatial samples are spaced by 0.8 cm along each axis, and the FOX has a diameter of roughly 20 cm. Given our choice of $\mathbf{r}_1, \ldots, \mathbf{r}_{356}$, we sample the $S(\mathbf{r})$'s and $d(\mathbf{r})$ and construct the $\mathbf{S}_p$'s and $\mathbf{d}$. Next, we define a grid of $N = 225$ points in (k_x, k_y)-space that is 15×15 in size and extends outward from the k-space origin. The points are spaced by $\frac{1}{20}$ cm^{-1} along each k-space axis, and the overall grid is shown in the right subplot of Figure 5.4. Finally, because we know the 356 $\mathbf{r}_m$'s and 225 $\mathbf{k}_n$'s, we construct the 356×225 matrix $\mathbf{A}$ in (5.2), (5.3) along with the $\mathbf{F}_p$'s in (5.3). We now have all the data we need to apply the algorithms and determine simultaneously K-sparse $\mathbf{g}_p$'s that (approximately) solve (5.4).

We apply the algorithms and evaluate designs where the use of $K \in \{1, \ldots, 30\}$ candidate points in k-space is permitted (in practical MRI scenarios, K up to 30 is

[6]Sampling points outside of the FOX where no profile has influence is unnecessary because an image can never be formed at these points no matter how much energy any given channel places throughout k-space.

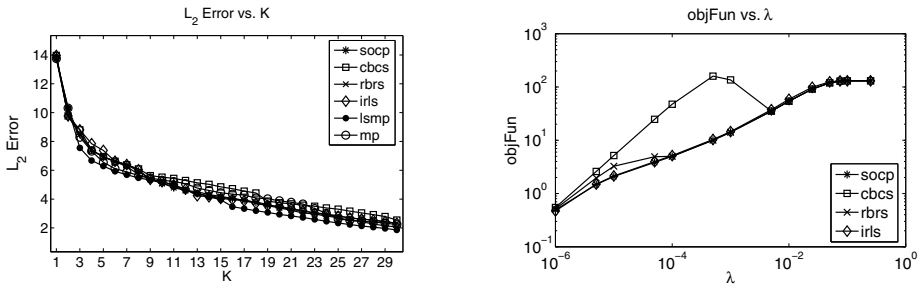

FIG. 5.5. MRI pulse design results. *Left subplot: ℓ_2 error vs. K is given for MP, LSMP, IRLS, RBRS, CBCS, and SOCP. For fixed K, LSMP consistently outperforms the other algorithms. Right subplot: objective function values vs. λ when SOCP, CBCS, RBRS, and IRLS attempt to minimize* (3.3), (3.7). *SOCP and IRLS converge and seem to minimize the objective; RBRS does so as well for most λ's. CBCS routinely fails to converge even after* 10 000 *iterations, and thus its solutions yield extremely large objective function values.*

permissible). Typically, the smallest K that produces a version of $d(\mathbf{r})$ to within some ℓ_2 fidelity is the design that the MRI pulse designer will use on a real system.

To obtain simultaneously K-sparse solutions with MP and LSMP, we set $K = 30$, run each algorithm once, remember the ordered list of chosen indices, and back out every solution for $K = 1$ through $K = 30$.

For each convex minimization method (IRLS, SOCP, RBRS, and CBCS), we apply the following procedure: first, we run the algorithm for 14 values of $\lambda \in \left[0, \frac{1}{4}\right]$, storing each resulting solution, $\widehat{\mathbf{g}}_{\text{tot}}(\text{alg}, \lambda)$. Then, for fixed K, to determine a simultaneously K-sparse deposition of energy on the k-space grid, we apply the retuning process to each of the 14 solutions, obtaining 14 strictly simultaneously K-sparse retuned sets of solution vectors, $\widehat{\mathbf{g}}_{\text{tot}}^{(K)}(\text{alg}, \lambda)$. The one solution among the 14 that best minimizes the ℓ_2 error between the desired and resulting images, $\|\mathbf{d} - \mathbf{F}_{\text{tot}}\mathbf{g}_{\text{tot}}^{(K)}(\text{alg}, \lambda)\|_2$, is chosen as the solution for the K under consideration. Essentially, we again assume we know a good value for λ when applying each of the convex minimization methods. To attempt to avoid convergence problems, RBRS and CBCS are permitted 5 000 and 10 000 maximum outer iterations, respectively (see below).

5.3.4. Results.

Image ℓ_2 error vs. number of energy depositions in k-space. The left subplot of Figure 5.5 shows the ℓ_2 error vs. K curves for each algorithm. As K is increased, each method produces images with lower ℓ_2 error, which is sensible: depositing energy at more locations in k-space gives each technique more degrees of freedom with which to form the image. In contrast to the sparsity pattern estimation experiments in sections 5.1 and 5.2, however, here LSMP is the best algorithm: for each fixed K considered in Figure 5.5, the LSMP technique yields a simultaneously K-sparse energy deposition that produces a higher-fidelity image than all other techniques. For example, when $K = 17$, LSMP yields an image with ℓ_2 error of 3.3. In order to produce an image with equal or better fidelity, IRLS, RBRS, and SOCP need to deposit energy at $K = 21$ points in k-space and thus produce less useful designs from an MRI perspective. CBCS fares the worst, needing to deposit energy at $K = 25$ grid points in order to compete with the fidelity of LSMP's $K = 17$ image.

Closer look: Objective function vs. λ. The right subplot of Figure 5.5 shows how well the four convex minimization algorithms minimize the objective function (3.3),

TABLE 5.2

Algorithm runtimes and peak memory usage for MRI pulse design. *Each algorithm's runtime and peak memory usage are listed. The runtimes of the latter four algorithms are averaged over the* 14 λ*'s per trial. MP is again faster than the other techniques, but consumes more memory because of its precomputation step* [70, 72]. *IRLS and SOCP are quite similar performance-wise and minimize the convex objective function equally well (see Figure* 5.5*), but we see here that IRLS is approximately* 1.9 *times faster and uses* 1.4 *times less peak memory than SOCP, making the former the superior technique among the convex methods.*

Algorithm	Runtime	Peak memory usage (MB)
MP	11 sec	704
LSMP	46 min	304
IRLS	50 sec	320
RBRS	87 min	320
CBCS	3.3 hr	320
SOCP	96 sec	432

(3.7) before retuning any solutions and enforcing strict simultaneous K-sparsity. For each fixed λ, SOCP and IRLS find solutions that yield the same objective function value. RBRS's solutions generally yield objective function values equal to those of SOCP and IRLS, but at times lead to higher values: in these cases RBRS almost converges but does not do so completely. Finally, for many λ's, CBCS fails to converge and yields extremely large objective function values. Some additional detail on convergence behavior is provided in [70, 72].

Runtimes and peak memory usage. Setting $K = 30$, we run MP and LSMP and record the runtime of each. Across the 14 λ's, the IRLS, RBRS, CBCS, and SOCP runtimes are recorded and averaged. The peak memory usage of each algorithm is also noted; these statistics are presented in Table 5.2. In distinct contrast to the smaller-variable-size experiments in sections 5.1 and 5.2 where all four convex minimization methods have relatively short runtimes (under one second), here RBRS and CBCS are much slower, leaving IRLS and SOCP as the only feasible techniques among the four. Furthermore, while LSMP does indeed outperform IRLS and SOCP on an ℓ_2 error basis (as shown in Figure 5.5), the runtime statistics here show that LSMP requires order-of-magnitude greater runtime to solve the problem—therefore, in some real-life scenarios where designing pulses in less than 5 minutes is a necessity, IRLS and SOCP are superior. Finally, in contrast to section 5.1's runtimes given in Table 5.1, here IRLS is 1.9 times faster than SOCP and uses 1.4 times less peak memory, making it the superior technique for MRI pulse design since IRLS's ℓ_2 error performance and ability to minimize the objective function (3.3), (3.7) essentially equal those of SOCP.

Closer look: Images and chosen k-space locations for $K = 17$. To conclude the experiment, we fix $K = 17$ and observe the images produced by the algorithms along with the points at which each algorithm chooses to deposit energy along the grid of candidate points in (k_x, k_y)-space. Figure 5.6 illustrates the images (in both magnitude and phase) that arise due to each algorithm's simultaneously 17-sparse set of solution vectors,[7] while Figure 5.7 depicts the placement pattern chosen by each method. From Figure 5.6, we see that each algorithm forms a high-fidelity version of the desired image $d(\mathbf{r})$ given in the left subplot of Figure 5.4, but among the images, LSMP's most accurately represents $d(\mathbf{r})$ (e.g., consider the sharp edges of the LSMP image's rectangular box). MP's and CBCS's images are noticeably fuzzy relative

[7]Each image is generated by taking the corresponding solution $\mathbf{g}_{\text{tot}}$, computing $\mathbf{m}$ in (5.3), unstacking the elements of $\mathbf{m}$ into $m(\mathbf{r})$, and then displaying the magnitude and phase of $m(\mathbf{r})$.

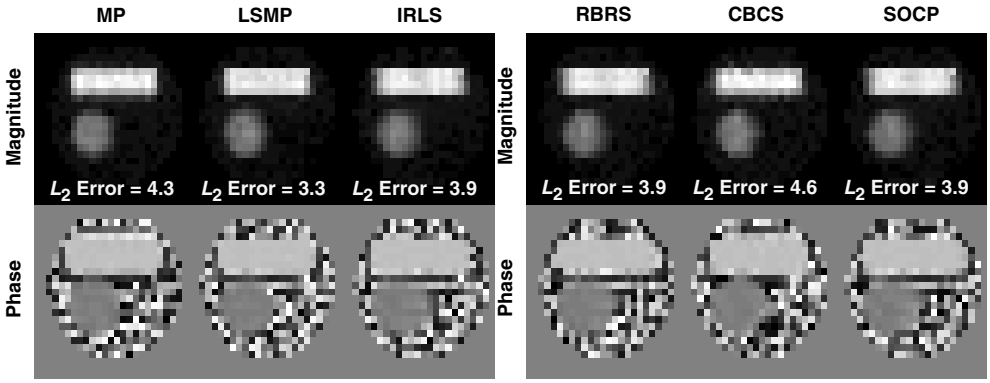

FIG. 5.6. MRI pulse design: Images per algorithm for $K = 17$. *Each algorithm is used to solve the MRI pulse design problem using* 17 *energy depositions along the k-space grid, attempting to produce an image close to the desired one,* $d(\mathbf{r})$*, given in the left subplot of Figure* 5.4. *From each set of simultaneously* 17*-sparse solution vectors, we calculate the resulting image via* (5.3) *and display both its magnitude and phase. LSMP's image best resembles the desired one; IRLS's, RBRS's, and SOCP's images are nearly as accurate; MP's and CBCS's images lack crisp edges, coinciding with their larger* ℓ_2 *errors.*

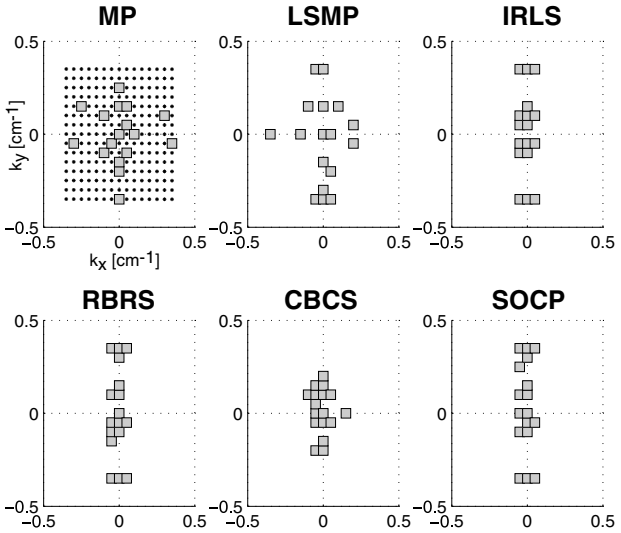

FIG. 5.7. MRI pulse design: Energy deposition patterns per algorithm for $K = 17$. *Each algorithm's placement of energy in k-space is displayed. LSMP branches out to moderate* k_x *frequencies and high* k_y *frequencies, partly explaining the superiority of its image in Figure* 5.6. *IRLS, RBRS, and SOCP succeed in branching out to higher* k_y *frequencies, but do not place energy at* $|k_x| \gg 0$. *MP and CBCS fail to spread their energy to high spatial frequencies, and thus their images in Figure* 5.6 *lack distinct edges and appear as "low-pass filtered" versions of* $d(\mathbf{r})$.

to the others. The placements in Figure 5.7 give insight into these performance differences. Here, LSMP places energy at several higher frequencies along the k_y and k_x axes, which ensures the resulting rectangle is narrow with sharp edges along the spatial y and x axes. In contrast, CBCS fails to place energy at moderate-to-high (k_x, k_y)-space frequencies and thus cannot produce a rectangle with desirable sharp edges, while MP branches out to some extent but fails to utilize high k_y frequencies.

IRLS, RBRS, and SOCP branch out to higher k_y frequencies but not to high k_x frequencies, and thus their associated rectangles in Figure 5.6 are sharp along the y axis but exhibit less distinct transitions (more fuzziness) along the spatial x axis. In general, each algorithm has determined 17 locations at which to place energy that yield a fairly good image, and each has avoided the computationally impossible scenario of searching over all N-choose-K (225-choose-17) possible placements.

6. Discussion.

6.1. MRI pulse design vs. denoising and source localization. The MRI pulse design problem in section 5.3 differs substantially from the source localization problem in section 5.1, the denoising experiment in section 5.2, and other routine applications of sparse approximation (e.g., [16, 7, 27, 22, 10, 11, 43]). It differs not only in purpose but in numerical properties, the latter of which are summarized in Table 6.1. While this list will not always hold true on an application-by-application basis, it does highlight general differences between the two problem classes.

TABLE 6.1
Unique trends of the MRI pulse design problem. *This table highlights differences between the MRI problem and standard denoising and source localization applications. Items here will not always be true, instead providing general highlights about each problem class.*

MRI pulse design	Denoising and source localization
• $\mathbf{F}_p$'s overdetermined • No concept of noise: given $\mathbf{d}$ is $\mathbf{d}_{\text{true}}$ • Sweep over λ useful • Metric: $\|\mathbf{d}-\mathbf{m}\|_2$	• $\mathbf{F}_p$'s underdetermined • Noisy: given $\mathbf{d}$ is not $\mathbf{d}_{\text{true}}$ • Ideal λ unknown • Metrics: $\|\mathbf{g}_{\text{tot}}-\widehat{\mathbf{g}}_{\text{tot}}\|_2$ and/or fraction of recovered sparsity pattern terms

6.2. Merits of row-by-row and column-by-column shrinkage. Even though LSMP, IRLS, and SOCP tend to exhibit superior performance across different experiments in this manuscript, RBRS and CBCS are worthwhile because, unlike the former methods that update all PN unknowns concurrently, the shrinkage techniques update only a subset of the total variables during each iteration.

For example, RBRS as given in Algorithm 3 updates only P unknowns at once, while CBCS updates but a single scalar at a time [70, 72]. RBRS and CBCS are thus applicable in scenarios where P and N are exceedingly large and tuning all PN parameters during each iteration is not possible. If storing and handling $M \times PN$ or $PN \times PN$ matrices exceeds a system's available memory and causes disk thrashing, RBRS and CBCS, though they require far more iterations, might still be better options than LSMP, IRLS, and SOCP in terms of runtime.

6.3. Future work.

6.3.1. Efficient automated control parameter selection. A fast technique for finding ideal values of λ is an open problem. It might be useful to investigate several approaches to automated selection such as the "L-curve" method [33], universal parameter selection [20], and min-max parameter selection [35].

6.3.2. Runtime, memory, and complexity reduction. As noted in section 4, LSMP's computation and runtime could be improved upon by extending the projection-based recursive updating schemes of [10, 11] to MSSO LSMP. In regard to the MRI design problem, runtime for any method might be reduced via a multiresolution approach (as in [43]) by first running the algorithm with a coarse k-space

frequency grid, noting which energy deposition locations are revealed, and then running the algorithm with a grid that is finely sampled around the locations suggested by the coarse result. This is faster than providing the algorithm a large, finely sampled grid and attempting to solve the sparse energy deposition task in one step. An initial investigation has shown that reducing the maximum frequency extent of the k-space grid (and thus the number of grid points, N) may also reduce runtime without significantly impacting image fidelity [74].

6.3.3. Shrinkage algorithm convergence improvements. When solving the MRI pulse design problem, both RBRS and CBCS required excessive iterations and hence exhibited lengthy runtimes. To mitigate these problems, one may consider extending parallel coordinate descent (PCD) shrinkage techniques used for SSSO sparse approximation (as in [21, 22]). Sequential subspace optimization (SESOP) [23] might also be employed to reduce runtime. Combining PCD with SESOP and adding a line search after each iteration would yield sophisticated versions of RBRS and CBCS.

6.3.4. Multiple-system multiple-output (MSMO) simultaneous sparse approximation. It may be useful to consider a combined problem where there are multiple observations as well as multiple system matrices. That is, assume we have a series of J observations, $\mathbf{d}_1, \ldots, \mathbf{d}_J$, each of which arises due to a set of P simultaneously K-sparse unknown vectors $\mathbf{g}_{1,j}, \ldots, \mathbf{g}_{P,j}$[8] passing through a set of P system matrices $\mathbf{F}_{1,j}, \ldots, \mathbf{F}_{P,j}$ and then undergoing linear combination, as follows:

$$\mathbf{d}_j = \mathbf{F}_{1,j}\mathbf{g}_{1,j} + \cdots + \mathbf{F}_{P,j}\mathbf{g}_{P,j} = \sum_{p=1}^{P} \mathbf{F}_{p,j}\mathbf{g}_{p,j} \quad \text{for } j = 1, \ldots, J. \tag{6.1}$$

If $\mathbf{F}_{p,j}$ is constant for all J observations, then the problem reduces to

$$\mathbf{d}_j = \mathbf{F}_1\mathbf{g}_{1,j} + \cdots + \mathbf{F}_P\mathbf{g}_{P,j} = \mathbf{F}_{\text{tot}}\mathbf{g}_{\text{tot},j} \quad \text{for } j = 1, \ldots, J, \tag{6.2}$$

and we may stack the matrices and terms as follows:

$$[\mathbf{d}_1, \ldots, \mathbf{d}_J] = \mathbf{F}_{\text{tot}} \left[\mathbf{g}_{\text{tot},1}, \ldots, \mathbf{g}_{\text{tot},J}\right]. \tag{6.3}$$

Having posed (6.1), (6.2), (6.3), one may formulate optimization problems similar to (2.4), (3.3) to determine simultaneously sparse $\mathbf{g}_{p,j}$'s that solve (6.3). Algorithms for solving such problems may arise by combining the concepts of SSMO algorithms [11, 43, 65, 62] with those of the MSSO algorithms posed earlier.

7. Conclusion. We defined the linear inverse MSSO simultaneous sparsity problem where simultaneously sparse sets of unknown vectors are required as the solution. This problem differed from prior problems involving multiple unknown vectors because, in this case, each unknown vector was passed through a different system matrix and the outputs of the various matrices underwent linear combination, yielding only one observation vector.

To solve the proposed MSSO problem, we formulated three greedy techniques (MP, OMP, and LSMP) along with algorithms based on IRLS, iterative shrinkage, and SOCP methodologies. The MSSO algorithms were evaluated across noiseless and noisy sparsity pattern estimation experiments as well as an MRI pulse design experiment; for sparsity pattern recovery, algorithms that minimized the relaxed convex

[8] The K-term simultaneous sparsity pattern of each set of $\mathbf{g}_{p,j}$'s may or may not change with j.

objective function outperformed the greedy methods, whereas in the MRI pulse design experiment, greedy LSMP exhibited superior performance.

When deriving RBRS for complex-valued data, we showed that seeking a single sparse complex-valued vector is equivalent to seeking two simultaneously sparse real-valued vectors: we mapped single-vector sparse approximation of a complex vector to the MSSO problem, increasing the applicability of algorithms that solve the latter.

Overall, while improvements upon these seven algorithms (and new algorithms altogether) surely do exist, this manuscript has laid the groundwork of the MSSO problem and conducted an initial exploration of algorithms with which to solve it.

Acknowledgments. The authors thank D. M. Malioutov for assistance with the derivation step that permitted the transition from (4.16) to (4.17) in section 4.7, as well as K. Setsompop, B. A. Gagoski, V. Alagappan, and L. L. Wald for collecting the experimental coil profile data in Figure 5.3. The authors also gratefully acknowledge discussions on related work with A. C. Gilbert, R. Maleh, and D. Yoon.

REFERENCES

[1] A. Ben-Tal and A. Nemirovski, *Lectures on Modern Convex Optimization: Analysis, Algorithms, and Engineering Applications*, MPS/SIAM Ser. Optim. 2, SIAM, Philadelphia, 2001.

[2] A. Bjorck and T. Elfving, *Accelerated Projection Methods for Computing Pseudoinverse Solutions of Systems of Linear Equations*, Technical report LiTH-MAT-R-1978-5, Department of Mathematics, Linköping University, Linköping, Sweden, 1978.

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

[4] A. M. Bruckstein, D. L. Donoho, and M. Elad, *From sparse solutions of systems of equations to sparse modeling of signals and images*, SIAM Rev., 51 (2009), pp. 34–81.

[5] E. J. Candès and T. Tao, *Near-optimal signal recovery from random projections: Universal encoding strategies?*, IEEE Trans. Inform. Theory, 52 (2006), pp. 5406–5425.

[6] S. Chen, S. A. Billings, and W. Luo, *Orthogonal least squares methods and their application to non-linear system identification*, Int. J. Control, 50 (1989), pp. 1873–1896.

[7] S. S. Chen, D. L. Donoho, and M. A. Saunders, *Atomic decomposition by basis pursuit*, SIAM J. Sci. Comput., 20 (1998), pp. 33–61.

[8] J. Chen and X. Huo, *Theoretical results on sparse representations of multiple-measurement vectors*, IEEE Trans. Signal Process., 54 (2006), pp. 4634–4643.

[9] S. Chen and J. Wigger, *Fast orthogonal least squares algorithm for efficient subset model selection*, IEEE Trans. Signal Process., 43 (1995), pp. 1713–1715.

[10] S. F. Cotter, J. Adler, B. D. Rao, and K. Kreutz-Delgado, *Forward sequential algorithms for best basis selection*, IEE Proc. Vision Image Signal Process., 146 (1999), pp. 235–244.

[11] 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.

[12] I. Daubechies, M. Defrise, and C. De Mol, *An iterative thresholding algorithm for linear inverse problems with a sparsity constraint*, Comm. Pure Appl. Math., 57 (2004), pp. 1413–1457.

[13] G. Davis, *Adaptive Nonlinear Approximations*, Ph.D. thesis, New York University, New York, 1994.

[14] G. Davis, S. Mallat, and Z. Zhang, *Adaptive time-frequency decomposition*, Opt. Eng., 33 (1994), pp. 2183–2191.

[15] D. L. Donoho, *Superresolution via sparsity constraints*, SIAM J. Math. Anal., 23 (1992), pp. 1309–1331.

[16] D. L. Donoho, *De-noising by soft-thresholding*, IEEE Trans. Inform. Theory, 41 (1995), pp. 613–627.

[17] D. L. Donoho, *Compressed sensing*, IEEE Trans. Inform. Theory, 52 (2006), pp. 1289–1306.

[18] D. L. Donoho, *For most large underdetermined systems of equations, the minimal ℓ_1-norm near-solution approximates the sparsest near-solution*, Comm. Pure Appl. Math., 59 (2006), pp. 907–934.

[19] D. L. Donoho, M. Elad, and V. N. Temlyakov, *Stable recovery of sparse overcomplete representations in the presence of noise*, IEEE Trans. Inform. Theory, 52 (2006), pp. 6–18.
[20] D. L. Donoho and I. M. Johnstone, *Ideal spatial adaptation by wavelet shrinkage*, Biometrika, 81 (1994), pp. 425–455.
[21] M. Elad, *Why simple shrinkage is still relevant for redundant representations?*, IEEE Trans. Inform. Theory, 52 (2006), pp. 5559–5569.
[22] M. Elad, B. Matalon, and M. Zibulevsky, *Image denoising with shrinkage and redundant representations*, in Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), Vol. 2, 2006, pp. 1924–1931.
[23] M. Elad, B. Matalon, and M. Zibulevsky, *Coordinate and subspace optimization methods for linear least squares with non-quadratic regularization*, Appl. Comput. Harmon. Anal., 23 (2007), pp. 346–367.
[24] Y. C. Eldar and H. Bölcskei, *Block-sparsity: Coherence and efficient recovery*, in Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, Taipei, Taiwan, 2009, pp. 2885–2888.
[25] Y. C. Eldar and H. Rauhut, *Average Case Analysis of Multichannel Sparse Recovery Using Convex Relaxation*, 2009; available online from http://www.arxiv.org/abs/0904.0494v1.
[26] A. K. Fletcher and S. Rangan, *Orthogonal matching pursuit from noisy random measurements: A new analysis*, in Proceedings of the Neural Information Processing Systems Conference, Vancouver, Canada, 2009.
[27] A. K. Fletcher, S. Rangan, V. K. Goyal, and K. Ramchandran, *Denoising by sparse approximation: Error bounds based on rate-distortion theory*, EURASIP J. Appl. Signal Process., 2006 (2006), 26318.
[28] M. Fornasier and H. Rauhut, *Recovery algorithms for vector-valued data with joint sparsity constraints*, SIAM J. Numer. Anal., 46 (2008), pp. 577–613.
[29] G. Golub and W. Kahan, *Calculating the singular values and pseudo-inverse of a matrix*, SIAM J. Numer. Anal., 2 (1965), pp. 205–224.
[30] G. H. Golub and C. F. Van Loan, *Matrix Computations*, Johns Hopkins University Press, Baltimore, 1983.
[31] I. F. Gorodnitsky and B. D. Rao, *Sparse signal reconstruction from limited data using FOCUSS: A recursive weighted norm minimization algorithm*, IEEE Trans. Signal Process., 45 (1997), pp. 600–616.
[32] 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., 14 (2008), pp. 655–687.
[33] P. C. Hansen, *Regularization tools: A Matlab package for analysis and solution of discrete ill-posed problems*, Numer. Algorithms, 6 (1994), pp. 1–35.
[34] D. H. Johnson and D. E. Dudgeon, *Array Signal Processing: Concepts and Techniques*, Prentice–Hall, Englewood Cliffs, NJ, 1993.
[35] I. M. Johnstone, *On minimax estimation of a sparse normal mean vector*, Ann. Statist., 22 (1994), pp. 271–289.
[36] L. A. Karlovitz, *Construction of nearest points in the L^p, p even, and L^∞ norms*. I, J. Approx. Theory, 3 (1970), pp. 123–127.
[37] U. Katscher, P. Bornert, C. Leussler, and J. S. van den Brink, *Transmit SENSE*, Magn. Reson. Med., 49 (2003), pp. 144–150.
[38] M. Kowalski and B. Torrésani, *Structured sparsity: From mixed norms to structured shrinkage*, in SPARS'09—Signal Processing with Adaptive Sparse Structured Representations, Saint-Malo, France, 2009.
[39] H. Krim and M. Viberg, *Two decades of array signal processing research. The parametric approach*, IEEE Signal Process. Mag., 13 (1996), pp. 67–94.
[40] C. Lanczos, *An iteration method for the solution of the eigenvalue problem of linear differential and integral operators*, J. Research Nat. Bur. Standards, 45 (1950), pp. 255–282.
[41] R. Maleh, D. Yoon, and A. C. Gilbert, *Fast algorithm for sparse signal approximation using multiple additive dictionaries*, in SPARS'09—Signal Processing with Adaptive Sparse Structured Representations, Saint-Malo, France, 2009.
[42] D. M. Malioutov, *A Sparse Signal Reconstruction Perspective for Source Localization with Sensor Arrays*, Master's thesis, Massachusetts Institute of Technology, Cambridge, MA, 2003.
[43] D. M. Malioutov, M. Çetin, and A. S. Willsky, *A sparse signal reconstruction perspective for source localization with sensor arrays*, IEEE Trans. Signal Process., 53 (2005), pp. 3010–3022.

[44] S. G. Mallat and Z. Zhang, *Matching pursuits with time-frequency dictionaries*, IEEE Trans. Signal Process., 41 (1993), pp. 3397–3415.
[45] 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.
[46] B. K. Natarajan, *Sparse approximate solutions to linear systems*, SIAM J. Comput., 24 (1995), pp. 227–234.
[47] S. Negahban and M. J. Wainwright, *Joint support recovery under high-dimensional scaling: Benefits and perils of $\ell_{1,\infty}$-regularization*, in Proceedings of the Neural Information Processing Systems Conference, Vancouver, Canada, 2008.
[48] G. Obozinksi, M. J. Wainwright, and M. I. Jordan, *High-dimensional support union recovery in multivariate regression*, in Proceedings of the Neural Information Processing Systems Conference, Vancouver, Canada, 2008.
[49] C. C. Paige and M. A. Saunders, *Algorithm* 583. *LSQR: Sparse linear equations and least squares problems*, ACM Trans. Math. Software, 8 (1982), pp. 195–209.
[50] C. C. Paige and M. A. Saunders, *LSQR: An algorithm for sparse linear equations and sparse least squares*, ACM Trans. Math. Software, 8 (1982), pp. 43–71.
[51] Y. C. Pati, R. Rezaiifar, and P. S. Krishnaprasad, *Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition*, in Conference Record of the Twenty-Seventh Asilomar Conference on Signals, Systems, and Computers, Vol. 1, Pacific Grove, CA, 1993, pp. 40–44.
[52] J. M. Pauly, D. Nishimura, and A. Macovski, *A k-space analysis of small-tip-angle excitation*, J. Magn. Reson., 81 (1989), pp. 43–56.
[53] S. Rangan, A. K. Fletcher, and V. K. Goyal, *Asymptotic Analysis of MAP Estimation via the Replica Method and Applications to Compressed Sensing*, 2009; available online from http://www.arxiv.org/abs/0906.3234v1.
[54] K. Setsompop, L. L. Wald, V. Alagappan, B. A. Gagoski, F. Hebrank, U. Fontius, F. Schmitt, and E. Adalsteinsson, *Parallel RF transmission with* 8 *channels at* 3 *Tesla*, Magn. Reson. Med., 56 (2006), pp. 1163–1171.
[55] M. Stojnic, F. Parvaresh, and B. Hassibi, *On the reconstruction of block-sparse signals with an optimal number of measurements*, IEEE Trans. Signal Process., 57 (2009), pp. 3075–3085.
[56] J. Sturm, *Using SeDuMi* 1.02*, a MATLAB toolbox for optimization over symmetric cones*, Optim. Methods Softw., 11/12 (1999), pp. 625–653.
[57] R. Tibshirani, *Regression shrinkage and selection via the lasso*, J. Royal Statist. Soc. Ser. B, 58 (1996), pp. 267–288.
[58] A. N. Tikhonov, *Solution of incorrectly formulated problems and the regularization method*, Dokl. Akad. Nauk SSSR, 151 (1963), pp. 501–504 (in Russian).
[59] A. N. Tikhonov and V. A. Arsenin, *Solution of Ill-Posed Problems*, Winston & Sons, Washington, DC, 1977.
[60] K. C. Toh, M. J. Todd, and R. H. Tutuncu, *SDPT3—a MATLAB software package for semidefinite programming, version* 1.3, Optim. Methods Softw., 11/12 (1999), pp. 545–581.
[61] J. A. Tropp, *Greed is good: Algorithmic results for sparse approximation*, IEEE Trans. Inform. Theory, 50 (2004), pp. 2231–2242.
[62] J. A. Tropp, *Algorithms for simultaneous sparse approximation: Part* II: *Convex relaxation*, Signal Process., 86 (2006), pp. 589–602.
[63] J. A. Tropp, *Just relax: Convex programming methods for identifying sparse signals in noise*, IEEE Trans. Inform. Theory, 52 (2006), pp. 1030–1051.
[64] J. A. Tropp and A. C. Gilbert, *Signal recovery from random measurements via orthogonal matching pursuit*, IEEE Trans. Inform. Theory, 53 (2007), pp. 4655–4666.
[65] 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.
[66] E. van den Berg and M. P. Friedlander, *Joint-Sparse Recovery from Multiple Measurements*, 2009; available online from http://www.arxiv.org/abs/0904.2051v1.
[67] M. J. Wainwright, *Sharp thresholds for high-dimensional and noisy sparsity recovery using ℓ_1-constrained quadratic programming (Lasso)*, IEEE Trans. Inform. Theory, 55 (2009), pp. 2183–2202.
[68] D. P. Wipf and B. Rao, *An empirical Bayesian strategy for solving the simultaneous sparse approximation problem*, IEEE Trans. Signal Process., 55 (2007), pp. 3704–3716.

[69] C. Y. Yip, J. A. Fessler, and D. C. Noll, *Advanced 3D tailored RF pulse for signal recovery in T_2^*-weighted functional magnetic resonance imaging*, Magn. Reson. Med., 56 (2006), pp. 1050–1059.

[70] A. C. Zelinski, *Improvements in Magnetic Resonance Imaging Excitation Pulse Design*, Ph.D. thesis, Massachusetts Institute of Technology, Cambridge, MA, 2008.

[71] A. C. Zelinski, V. K. Goyal, E. Adalsteinsson, and L. L. Wald, *Sparsity in MRI RF excitation pulse design*, in Proceedings of the 42nd Annual Conference on Information Sciences and Systems, Princeton, NJ, 2008, pp. 252–257.

[72] A. C. Zelinski, V. K. Goyal, and E. Adalsteinsson, *Simultaneously Sparse Solutions to Linear Inverse Problems with Multiple System Matrices and a Single Observation Vector*, 2009; available online from http://www.arxiv.org/abs/0907.2083v1.

[73] A. C. Zelinski, L. L. Wald, K. Setsompop, V. Alagappan, B. A. Gagoski, V. K. Goyal, and E. Adalsteinsson, *Fast slice-selective radio-frequency excitation pulses for mitigating B_1^+ inhomogeneity in the human brain at 7 Tesla*, Magn. Reson. Med., 59 (2008), pp. 1355–1364.

[74] A. C. Zelinski, L. L. Wald, K. Setsompop, V. K. Goyal, and E. Adalsteinsson, *Sparsity-enforced slice-selective MRI RF excitation pulse design*, IEEE Trans. Med. Imaging, 27 (2008), pp. 1213–1229.