A FAST ALGORITHM FOR SPARSE RECONSTRUCTION BASED ON SHRINKAGE, SUBSPACE OPTIMIZATION AND CONTINUATION

ZAIWEN WEN [†], WOTAO YIN [‡], DONALD GOLDFARB [†], AND YIN ZHANG [§]

January, 2009

Abstract. We propose a fast algorithm for solving the ℓ_1-regularized minimization problem $\min_{x\in\mathbb{R}^n} \mu\|x\|_1 + \|Ax-b\|_2^2$ for recovering sparse solutions to an undetermined system of linear equations $Ax = b$. The algorithm is divided into two stages that are performed repeatedly. In the first stage a first-order iterative method called "shrinkage" yields an estimate of the subset of components of x likely to be nonzero in an optimal solution. Restricting the decision variables x to this subset and fixing their signs at their current values reduces the ℓ_1-norm $\|x\|_1$ to a linear function of x. The resulting subspace problem, which involves the minimization of a smaller and smooth quadratic function, is solved in the second phase. Our code FPC_AS embeds this basic two-stage algorithm in a continuation (homotopy) approach by assigning a decreasing sequence of values to μ. This code exhibits state-of-the-art performance both in terms of its speed and its ability to recover sparse signals. It can even recover signals that are not as sparse as required by current compressive sensing theory.

Key words. ℓ_1-minimization, basis pursuit, compressive sensing, subspace optimization, active set, continuation, shrinkage

AMS subject classifications. 49M29, 65K05, 90C25, 90C06

1. Introduction.

Frequently, the dominant information in an image or signal is much "sparser" than the image or signal itself under a proper representation. The fundamental principal of the emerging technology of *compressive sensing* (CS) is that a K-sparse signal $\bar{x} \in \mathbb{R}^n$ can be recovered from relatively few incomplete measurements $b = A\bar{x}$ for a carefully chosen $A \in \mathbb{R}^{m\times n}$ by solving the ℓ_0-minimization problem

$$\min_{x\in\mathbb{R}^n} \|x\|_0 \text{ subject to } Ax = b, \tag{1.1}$$

where $\|x\|_0 := |\{i, x_i \neq 0\}|$, $K \leq m \leq n$ (often $K \ll m \ll n$). Moreover, Candes, Romberg and Tao [12, 13, 14], Donoho [21] and their colleagues have shown that, under some reasonable conditions on $\bar{x}$ and A, the sparsest solution $\bar{x}$ of problem (1.1) can be found by solving the basis pursuit (BP) problem

$$\min_{x\in\mathbb{R}^n} \|x\|_1 \text{ subject to } Ax = b. \tag{1.2}$$

For more information on compressive sensing, see [21, 55, 60, 61, 52, 41, 56, 38, 40, 39, 65].

Greedy algorithms have been proposed to recover the solution $\bar{x}$ of problem (1.1) when the data satisfy certain conditions, such as, the restricted isometry property [14]. These algorithms include Orthogonal Matching Pursuit (OMP) [42, 54], Stagewise OMP (StOMP) [22], CoSaMP [45], Subspace Pursuit (SP) [16], and many other variants. These algorithms, by and large, involve solving a sequence of subspace optimization problems of the form

$$\min_x \|A_T x_T - b\|_2^2, \quad \text{subject to } x_i = 0,\ \forall i \notin T, \tag{1.3}$$

[†]Department of Industrial Engineering and Operations Research, Columbia University, New York, 10027, U.S.A. (zw2109@columbia.edu, goldfarb@columbia.edu). Research supported in part by NSF Grant DMS 06-06712, ONR Grants N000140310514 and N00014-08-1-1118 and DOE Grants DE-FG01-92ER-25126 and DE-FG02-08ER58562.

[‡]Department of Computational and Applied Mathematics, Rice University, Texas, 77005, U.S.A. (wotao.yin@rice.edu). Research supported in part by NSF CAREER Award DMS-07-48839 and ONR Grant N00014-08-1-1101.

[§]Department of Computational and Applied Mathematics, Rice University, Texas, 77005, U.S.A. (yzhang@rice.edu). Research supported in part by NSF grants DMS-0405831 and DMS-0811188 and ONR grant N00014-08-1-1101.

where T is an index set of the components of x. Starting from $T = \emptyset$ and $x = 0$, OMP iteratively adds to T the index of the largest component of the current gradient $g(x)$ of $\|Ax - b\|_2^2$ and solves (1.3) to obtain a new point x. StOMP adds those indices (usually more than one) at each iteration that correspond to the components of $g(x)$ whose magnitudes exceed a threshold. Rather than being a monotonically growing index set as in OMP and StOMP, at each iteration of CoSaMP and SP the index set T is a union of the indices of the K most significant components of the current point x and the K most significant components of the gradient $g(x)$.

Closely related to the convex optimization problem (1.2) is the ℓ_1-regularized minimization problem

$$\min_{x \in \mathbb{R}^n} \psi_\mu(x) := \mu \|x\|_1 + \|Ax - b\|_2^2, \tag{1.4}$$

where $\mu > 0$. The theory for penalty functions implies that the solution of the quadratic penalty function (1.4) goes to the solution of (1.2) as μ goes to zero. It has been shown in [66] that (1.2) is equivalent to a ℓ_1-regularized problem of the form (1.4) for a suitable choice of b (which is different from the b in (1.4)). Furthermore, if the measurements are contaminated with noise ϵ, i.e., $b = A\bar{x} + \epsilon$, or $\bar{x}$ is only approximately sparse, namely, it contains a small number of components with magnitudes significantly larger than those of the remaining components which are not necessarily zero, then problem (1.4) is a more appropriate model than (1.2). Another related problem is the LASSO problem, i.e.,

$$\min_{x \in \mathbb{R}^n} \|Ax - b\|_2^2 \quad , \text{s.t.}, \|x\|_1 \leq t, \tag{1.5}$$

which is equivalent to (1.4) for an appropriate choice of the parameter t.

Problems (1.2), (1.4) and (1.5) are all convex optimization problems. Moreover, (1.2) can easily be transformed into a linear programming (LP) problem and (1.4) and (1.5) can be transformed into quadratic programming (QP) problems and these problems can be solved by standard LP and QP methods. However, computational challenges arise from the following facts. First, real-world applications are invariably large-scale. For example, the decision variable corresponding to a 1024×1024 image has over a million variables. Second, A is generally dense. Third, real-time or near real-time processing is required in some applications. Consequently, algorithms requiring matrix decompositions or factorizations are not practical. On the other hand, the measurement matrices A that arise in applications often correspond to partial transform matrices (e.g., discrete Fourier and cosine transforms), for which fast matrix-vector multiplications (e.g., FFT and DCT) are available. Moreover, the sparsity of the solutions presents a unique opportunity for achieving relatively fast convergence with a first-order method. These features make the development of efficient optimization algorithms for compressive sensing applications an interesting research area. Examples of such algorithms include, shrinkage-based algorithms [28, 48, 19, 3, 23, 24, 15, 34, 33, 63, 58, 49], the interior-point algorithm $\ell_1\_\ell_s$ [37], SPGL1 [59] for the LASSO problem, the spectral gradient projection method GPSR [29] and the fixed-point continuation method FPC [34] for the ℓ_1-regularized problem (1.4), and the gradient method in [46] for minimizing the more general function $J(x) + H(x)$, where J is non-smooth, H is smooth, and both are convex.

In this paper, we propose a two-stage algorithm for compressive sensing based on solving problem (1.4) by combining the good features of both greedy algorithms and convex optimization approaches. While the greedy algorithms (OMP and its variants) take advantage of the sparsity structure via minimizing a sequence of subspace problems (1.3), they require some prior information, like the cardinality K of the sparse solution. In contrast, the convex optimization approaches mentioned in the previous paragraph do not require any prior

information, but do not fully use the sparsity of the solution. Hence, our basic idea is to use some iterations of a convex optimization approach to generate the working set T for a subspace optimization problem similar to (1.3). First, a first-order method based on shrinkage is applied to obtain an approximate solution of (1.4) and identify a working set. Then a second-order method is applied to solve a smooth problem (subspace optimization) defined by the working set starting from the approximate solution. These two operations are then repeated until the solution satisfies approximate optimality conditions for problem (1.4). Embedding the basic two-stage algorithm in a continuation (homotopy) approach by assigning a decreasing sequence of values to μ, our algorithm exhibits state-of-the-art performance both in terms of its speed and its ability to recover sparse signals. In fact it can even recover signals that are not as sparse as required by current compressive sensing theory.

The purpose of shrinkage in our algorithm is two-fold, providing a good initial point and identifying a new working set for subspace optimization. Although subspace optimization is used in GPSR [29] and FPC [34], it is performed only once by these algorithms as a post-processing step (and referred to as debiasing). Our method is also significantly different from the greedy algorithms mentioned earlier in terms of the formulation of the subspace optimization problem and the way the working set T for that problem is chosen at each iteration using no prior information about the sparsity of the solution.

The rest of this paper is organized as follows. In section 2, we introduce an abstract framework for our two-stage algorithm. In section 3, we discuss the shrinkage phase, a nonmontone line search method and an exact line search method. In section 4, we state the formulation of the subspace optimization problem, present criteria for starting the subspace optimization phase and discuss methods for choosing the active set. Our continuation strategy is described in section 5 and the complete algorithm is presented in section 6. Finally, numerical results on an extensive collection of problems arising in compressive sensing, including several very challenging "pathological" problems, are presented in section 7 to demonstrate the robustness and efficiency of our algorithm.

2. Overview of the algorithm. Our primary goal is to obtain a solution of the ℓ_0-minimization problem (1.1) by solving the ℓ_1-regularized problem (1.4) using an active set method. Active set methods have been extensively studied in nonlinear programming. Compared with interior point methods, active set methods are more robust and better able to take advantage of warm starts [47, 51]. Our active set algorithm works as follows: a first-order method is used to define an active set, and then a second-order method is used to explore a subspace problem based on this active set. These two operations are iterated until convergence criteria are satisfied. Examples of other active set methods that follow this framework include those in which gradient projection and conjugate gradient methods are combined to solve problems with bound constraints or linear constraints [44, 7, 43, 8, 31] and those that combine linear and quadratic programming approaches to solve general nonlinear programming problems [9, 10]. In our approach, shrinkage is used to identify an active set in the first stage and a smooth subproblem, which is much easier to handle, is formulated and solved in the second stage. We briefly describe the motivation underlying our approach and introduce an abstract form of our algorithm in this section.

For convenience of notation let

$$f(x) := \|Ax - b\|_2^2 \quad \text{and} \quad g(x) := \nabla f(x).$$

For $x, y \in \mathbb{R}^n$, let $x \odot y$ denote the component-wise product of x and y, i.e., $(x \odot y)_i = x_i y_i$. Given $y \in \mathbb{R}^n$

and $\nu \in \mathbb{R}$, shrinkage is defined as

$$\mathcal{S}(y, \nu) := \operatorname{sgn}(y) \odot \max\{|y| - \nu, \mathbf{0}\}, \tag{2.1}$$

which is the unique minimizer of the function $\nu\|x\|_1 + \frac{1}{2}\|x - y\|_2^2$. Given a point x^k, our iterative shrinkage procedure generates a new point

$$x^{k+1} := \mathcal{S}\left(x^k - \lambda g^k, \mu\lambda\right), \tag{2.2}$$

where $g^k := g(x^k)$ and $\lambda > 0$. The iteration (2.2) has been independently proposed by different groups of researchers in various contexts [3, 19, 23, 24, 28, 34, 48]. One appealing feature, which has been proved in [34], is that it yields the support and the signs of the optimal solution x^* of (1.4) in a finite number of iterations under suitable conditions; that is, there exists a number $\bar{k}$ such that $\operatorname{sgn}(x^k) \equiv \operatorname{sgn}(x^*)$ (following the convention $\operatorname{sgn}(0) = 0$) for all $k \geq \bar{k}$. Convergence of (2.2) under suitable conditions on λ and the Hessian $\nabla^2 f$ has been studied in [15, 18, 34]. The local q-linear convergence proved in [34] depends on properties of the reduced Hessian of $f(x)$ restricted to the support of x^*. Various modifications and enhancements have been applied to (2.2), which has also been generalized to certain other nonsmooth functions; see [25, 6, 29, 63].

The first stage of our approach is based on the iterative shrinkage scheme (2.2). Once a "good" approximate solution x^k of the ℓ_1-regularized problem (1.4) is obtained from (2.2), the set of indices corresponding to the zero and nearly zero components of x^k (i.e., $|x_i^k| \leq \xi^k$, where $\xi^k \in (0, 1)$), is selected as a working set. In the second stage, a subspace optimization problem is formed by fixing the components in this working set to zero and approximating the ℓ_1-norm $\|x\|_1$ by a smooth function. For several possibilities, see [26]. In this paper, we simply use a linear approximation $c^\top x$, where the components of the fixed vector c are taken as the signs of the components of x^k since $|x_i| = \operatorname{sgn}(x_i)x_i$. Simple bound constraints can be imposed so that $c_i x_i \geq 0$. The choice of a practical termination criteria to judge whether the solution obtained from shrinkage is "good" enough is critical, since the estimate in [34] of the steps needed to identify the support and the signs of the nonzero components of x^* depends on the exact solution x^* which is unknown. We will present some effective rules and explain the motivation underlying them in Section 4.

Instead of solving problem (1.4) directly from scratch, we use a continuation (homotopy) procedure to solve a sequence of problems $\{x^*_{\mu_k} := \arg\min_{x \in \mathbb{R}^n} \psi_{\mu_k}(x)\}$, where $\mu_0 > \mu_1 > \cdots > \mu$, using the solution (or approximate solution) $x^*_{\mu_{k-1}}$ as the initial estimate of the solution to the next problem. It has been shown empirically in [32] that using the basic shrinkage scheme (2.2) to obtain each $x^*_{\mu_k}$ in a continuation strategy is far superior to applying the basic shrinkage scheme to (1.4) directly. Experiments in [59, 63] have further confirmed the effectiveness of continuation. Therefore, we embed our two-stage algorithm in a continuation procedure. Allowing the parameter to be chosen dynamically and incorporating a line search in each shrinkage step results in the following abstract algorithm, Algorithm 1.

We note that shrinkage provides a strategy similar to those used by greedy algorithms to select the set T based on information about x and the gradient $g(x)$ in problem (1.4). This connection is most obvious if we look at a shrinkage step right after greedy algorithms perform the subspace optimization step (1.3). Assume that x^{k+} is generated by subspace optimization (1.3) with an index set T^k. The optimality conditions for (1.3) are $A_{T^k}^\top (A_{T^k} x_{T^k}^{k+} - b) = 0$, which implies that the gradient $g^{k+} = (\mathbf{0}, g_{\overline{T}^k}^{k+})^\top$. Substituting g^{k+} and

Algorithm 1: An Abstract Active Set Algorithm

Initialization: Choose x^0, μ_0.
for $k = 0, 1, \cdots$ *until convergence* **do**
S1 Shrinkage phase: Select a parameter λ^k and compute a direction $d^k \leftarrow \mathcal{S}(x^k - \lambda^k g^k, \mu_k \lambda^k) - x^k$.
Do a line search to obtain a step size α_k and set the new point $x^{k+1} \leftarrow x^k + \alpha_k d^k$.
S2 **if** *certain conditions are met* **then**
Sub-optimization: Determine an active set based upon x^{k+1}.
$x^{k+1} \leftarrow$ the solution of the subspace optimization problem over the active set.
Compute μ_{k+1}

$x^{k+} = (x^{k+}_{T^k}, \mathbf{0})^\top$ into the shrinkage operator, we obtain

$$x^{k+1} = \mathcal{S}\left(x^{k+} - \lambda g^{k+}, \mu_k \lambda\right) = \begin{cases} \operatorname{sgn}(x_i^{k+}) \max(|x_i^{k+}| - \mu_k \lambda, 0), & \text{if } i \in T^k, \\ \operatorname{sgn}(\lambda g_i^{k+}) \max(|\lambda g_i^{k+}| - \mu_k \lambda, 0), & \text{if } i \in \overline{T}^k. \end{cases} \tag{2.3}$$

Hence, shrinkage selects indices corresponding to components x_i in the previous working set T^k whose magnitudes are larger than the threshold $\mu_k \lambda$ and indices corresponding to components of the gradient in the complement of T^k whose magnitudes are larger than $\mu_k \lambda$.

3. The shrinkage phase. We now describe the first stage of our algorithm. In the fixed point method in [32], the parameter λ in (2.2) is fixed so that the fixed-point iteration is a contraction at every iteration. Since a bad value of λ usually slows down the rate of convergence, we choose λ dynamically to improve the performance of shrinkage. We also incorporate a line search scheme to guarantee global convergence. The theoretical properties of our algorithm, including global convergence, *R*-linear convergence and the identification of the active set after a finite number of steps are studied in a companion paper [62].

Our line search scheme is based on properties of the search direction determined by shrinkage (2.1) and (2.2). Let $d^k := d^{(\lambda^k)}(x^k)$ denote this direction, i.e.,

$$d^{(\lambda)}(x) := x^+ - x, \quad x^+ = \mathcal{S}(x - \lambda g, \mu\lambda), \tag{3.1}$$

for $x \in \mathbb{R}^n$, $\mu > 0$ and $\lambda > 0$. Since shrinkage (2.1) is the solution of the non-smooth unconstrained minimization problem $\min_{x \in \mathbb{R}^n} \nu\|x\|_1 + \frac{1}{2}\|x - y\|_2^2$, and the latter is equivalent to the smooth constrained problem

$$\min \frac{1}{2}\|x - y\|_2^2 + \nu\xi, \text{ subject to } (x, \xi) \in \Omega := \{(x, \xi) \mid \|x\|_1 \le \xi\},$$

we can obtain from the optimality conditions for latter problem that

$$(\mathcal{S}(x, \nu) - x)^\top (y - \mathcal{S}(x, \nu)) + \nu(\xi - \|\mathcal{S}(x, \nu)\|_1) \ge 0 \tag{3.2}$$

for all $x \in \mathbb{R}^n$ and $(y, \xi) \in \Omega$ and $\nu > 0$ [62]. Substituting $x - \lambda g$ for x, x for y and $\|x\|_1$ for ξ and setting $\nu = \mu\lambda$ in (3.2), we obtain

$$(\mathcal{S}(x - \lambda g, \mu\lambda) - (x - \lambda g))^\top (x - \mathcal{S}(x - \lambda g, \mu\lambda)) + \mu\lambda(\|x\|_1 - \|\mathcal{S}(x - \lambda g, \mu\lambda)\|_1) \ge 0,$$

which gives

$$g^\top d + \mu(\|x^+\|_1 - \|x\|_1) \leq -\frac{1}{\lambda}\|d\|_2^2, \tag{3.3}$$

after rearranging terms. An alternative derivation of (3.3) is given in Lemma 2.1 in [57].

We can also reformulate (1.4) as

$$\min f(x) + \mu\xi, \quad \text{subject to } (x, \xi) \in \Omega, \tag{3.4}$$

whose first-order optimality conditions for a stationary point x^* are

$$\nabla f(x^*)(x - x^*) + \mu(\xi - \|x^*\|_1) \geq 0, \text{ for all } (x, \xi) \in \Omega, \tag{3.5}$$

since $\xi^* = \|x^*\|_1$. Hence, d^k is similar to a gradient projection direction for solving (3.4), and should have many properties of the latter. In particular, it has been shown in [57, 62] that for any $x^* \in \mathbb{R}^n$ and $0 < \lambda < \infty$,

$$d^{(\lambda)}(x^*) = \mathbf{0} \tag{3.6}$$

if and only if x^* is a stationary point for (1.4).

Since in our method λ is not chosen to ensure contraction, a backtracking line search is necessary to guarantee global convergence. Consequently, at each iteration, we compute the next point as $x^{k+1} = x^k + \alpha_k d^k$, where $\alpha_k = \rho^h$ and $0 < \rho < 1$ are constants and h is the smallest integer that satisfies the Armijo-like condition

$$\psi_\mu(x^k + \rho^h d^k) \leq C_k + \rho^h \Delta^k. \tag{3.7}$$

Here C_k is a reference value with respect to the previous values $\{\psi_\mu^0, \cdots, \psi_\mu^k\}$, and

$$\Delta^k := (g^k)^\top d^k + \mu\|x^{k+}\|_1 - \mu\|x^k\|_1 \leq 0. \tag{3.8}$$

From (3.3) and the convexity of the ℓ_1-norm, it is easy to show that there exists a (backtracking) step size that satisfies (3.7) with $C^k = \psi_\mu(x^k)$. Such a line search method is monotone since $\psi_\mu(x^{k+1}) < \psi_\mu(x^k)$. Instead of using it, we use a nonmonotone line search method based on a strategy proposed in [67] (see algorithm "NMLS" (Algorithm 2)). In this method, the reference value C_k in the Armijo-like condition (3.7) is taken as a convex combination of the previous value of C_{k-1} and the function value $\psi_\mu(x^k)$, and as the iterations proceed the weight on C_k is increased. For further information on nonmonotone line search methods, see [17, 30, 53].

In [62], we show that there exists a step size satisfying the Armijo-like condition (3.7) for C_k generated in Algorithm "NMLS". Therefore, every iteration of algorithm "NMLS" is well defined. From that algorithm, we have

$$\psi_\mu(x^{k+1}) \leq C_k \leq C_{k-1} \leq \cdots \leq C_0 = \psi_\mu(x^0). \tag{3.9}$$

We also prove that Algorithm "NMLS" converges. Specifically, let $\mathcal{L}$ be the level set $\mathcal{L} := \{x \in \mathbb{R}^n : \psi_\mu(x) \leq$

Algorithm 2: Nonmonotone Line Search Algorithm (NMLS)

Initialization: Choose a starting guess x^0, and parameters $0 < \eta < 1$, $0 < \rho < 1$, and $0 < \lambda_m < \lambda_M < \infty$. Set $C_0 = \psi_\mu(x^0)$, $Q_0 = 1$, and $k = 0$.

while *"not converge"* **do**

- Computing a search direction: Choose a $\lambda_m \le \lambda^k \le \lambda_M$. Set $d^k = \mathcal{S}(x^k - \lambda^k g^k, \mu\lambda^k) - x^k$.
- Selecting a step size: Set $x^{k+1} = x^k + \alpha_k d^k$, where $\alpha_k = \rho^{h_k}$ and h_k is the smallest integer such that α_k satisfies the non-monotone Armijo-like condition (3.7).
- Update: Set $Q_{k+1} = \eta Q_k + 1$, $C_{k+1} = (\eta Q_k C_k + \psi_\mu(x^{k+1}))/Q_{k+1}$. Set $k \leftarrow k + 1$

$\psi_\mu(x^0)\}$ and $\widetilde{\mathcal{L}}$ be the set of points of $x \in \mathbb{R}^n$ whose distance to $\mathcal{L}$ is at most $\sup_k \|d^k\| < \infty$. Assuming that $f(x)$ is bounded from below on $\widetilde{\mathcal{L}}$ and ∇f is Lipschitz continuous on $\widetilde{\mathcal{L}}$, we prove that the sequence $\{x^k\}$ is globally convergent in the sense that $\lim_{k\to\infty}\|d^k\| = 0$. It is also proved that the sequence $\{x^k\}$ is at least R-linearly convergent under some mild assumptions.

We now specify a strategy, which is based on the Barzilai-Borwein method (BB) [2], for choosing the parameter λ^k. The shrinkage iteration (2.2) first takes a gradient descent step with step size λ^k along the negative gradient direction g^k of the smooth function $f(x)$, and then applies the shrink operator $\mathcal{S}(\cdot,\cdot)$ to accommodate the nonsmooth term $\|x\|_1$. Hence, it is natural to choose λ^k based on the function $f(x)$ alone. Let

$$s^{k-1} = x^k - x^{k-1}, \quad y^{k-1} = g^k - g^{k-1}.$$

The BB step is defined so that it corresponds to premultiplying the negative gradient by a multiple of identity that has a quasi-Newton property; specifically,

$$\lambda^{k,BB1} = \frac{(s^{k-1})^\top s^{k-1}}{(s^{k-1})^\top y^{k-1}} \quad \text{or} \quad \lambda^{k,BB2} = \frac{(s^{k-1})^\top y^{k-1}}{(y^{k-1})^\top y^{k-1}}. \tag{3.10}$$

To avoid the parameter λ being either too small or too large, we take

$$\lambda^k = \max\{\lambda_m, \min\{\lambda^{k,BB1}, \lambda_M\}\} \text{ or } \lambda^k = \max\{\lambda_m, \min\{\lambda^{k,BB2}, \lambda_M\}\}, \tag{3.11}$$

where $0 < \lambda_m \le \lambda_M < \infty$ are fixed parameters. We should point out that the idea of using the BB step in compressive sensing has also appeared in [64, 35]. However, Algorithm "NMLS" only requires that λ^k be bounded and other strategies could easily be adopted.

3.1. An exact line search. An exact line search is possible if $\psi_\mu(\cdot)$ is a piecewise quadratic function. We want to solve

$$\begin{aligned} \min_{\alpha\in[0,1]} \psi_\mu(x + \alpha d) &:= \mu\|x + \alpha d\|_1 + \frac{1}{2}\|A(x + \alpha d) - b\|_2^2, \\ &= \mu\|x + \alpha d\|_1 + \frac{1}{2}c_1\alpha^2 + c_2\alpha + c_3, \end{aligned} \tag{3.12}$$

where $c_1 = \|Ad\|_2^2$, $c_2 = (Ad)^\top(Ax - b)$ and $c_3 = \|Ax - b\|_2^2$. The break points of $\psi_\mu(x + \alpha d)$ are $\{\alpha_i = -x_i/d_i, d_i \neq 0, i = 1, \cdots, n\}$. Since $\alpha \in [0, 1]$, we select those $\alpha_i \in [0, 1]$ and sort these points together with

0 and 1 as

$$\alpha_{(0)} = 0 < \alpha_{(1)} < \cdots < \alpha_{(\kappa-1)} < 1 = \alpha_{(\kappa)}. \tag{3.13}$$

For each interval $[\alpha_{(l)}, \alpha_{(l+1)}]$ for $l = 0, \cdots, \kappa$ the function $\psi_\mu(x + \alpha d)$ is a smooth quadratic function of α. Let the minimizer of the function determined by the interval $[\alpha_{(l)}, \alpha_{(l+1)}]$ be denoted by $\bar{\alpha}_{(l)}$. Then $\bar{\alpha}_{(l)}$ is the optimal solution of (3.12) if $\bar{\alpha}_{(l)} \in [\alpha_{(l)}, \alpha_{(l+1)}]$. Hence, we only have to search each interval to obtain the optimal solution of (3.12). This algorithm is outlined in Algorithm 3.

Algorithm 3: An exact line search algorithm for solving (3.12)

Initialization: Compute $c_1 = \|Ad\|_2^2$, $c_2 = (Ad)^\top(Ax - b)$. Compute $\alpha_i = -x_i/d_i$ for $d_i \neq 0$ and sort the α_i such that (3.13) is satisfied.
for $i = \kappa, \cdots, 1$ **do**
  Compute $x^l = x + \alpha_{(i-1)}d$ and $x^u = x + \alpha_{(i)}d$
  Set $\mathcal{I}^l = \{i : x_i^l \leq 0 \text{ and } x_i^u \leq 0\}$ and $\mathcal{I}^u = \{i : x_i^l \geq 0 \text{ and } x_i^u \geq 0\}$
  Compute $\rho = \mu(\sum_{i\in\mathcal{I}^u} d_i - \sum_{i\in\mathcal{I}^l} d_i)$ and $\alpha = -(c_2 + \rho)/c_1$.
  if $\alpha_{(i-1)} \leq \alpha \leq \alpha_{(i)}$ **then** return α and exit the loop.
Return $\alpha = \alpha_i$ such that $i = \arg\min_{i=1,\cdots,\kappa} \psi_\mu(x + \alpha_i d)$.

In our algorithm, we perform an exact line search if the Armijo-like condition (3.7) is not satisfied with a unit step size, but we still update the parameters Q_{k+1} and C_{k+1} for the next iteration. Such a hybrid method works well in practice.

4. The subspace optimization phase. We begin by introducing some notation. The active set is denoted by $\mathcal{A}(x)$ and the inactive set (or support) is denoted by $\mathcal{I}(x)$, i.e.,

$$\mathcal{A}(x) := \{i \in \{1, \cdots, n\} \,|\, |x_i| = 0\} \text{ and } \mathcal{I}(x) := \{i \in \{1, \cdots, n\} \,|\, |x_i| > 0\}. \tag{4.1}$$

The active set is further subdivided into two sets

$$\mathcal{A}_\pm(x) := \{i \in \mathcal{A}(x) \,|\, |g_i(x)| < \mu\} \text{ and } \mathcal{A}_0(x) := \{i \in \mathcal{A}(x) \,|\, |g_i(x)| \geq \mu\}. \tag{4.2}$$

If x^* is an optimal solution of (1.4) and $i \in \mathcal{A}_0(x^*)$, then $|g_i(x^*)| = \mu$. The problem (1.4) is said to be *degenerate* at x^* if $\mathcal{A}_0(x^*) \neq \emptyset$. The components of x^* in $\mathcal{A}_0(x)$ are called degenerate while those in $\mathcal{A}_\pm(x^*)$ are called nondegenerate.

We now show that the iterative shrinkage scheme (2.2) essentially reduces to a gradient projection method for solving a subspace minimization problem after sufficiently many iterations. Suppose that $f(x)$ is a twice differentiable convex function and the eigenvalues of its Hessian are uniformly bounded. Let X^* be the set of optimal solutions of (1.4). It has been shown in [34] that there exists a vector

$$g_i^* \begin{cases} = -\mu, & \max\{x_i : x \in X^*\} > 0, \\ = +\mu, & \min\{x_i : x \in X^*\} < 0, \\ \in [-\mu, \mu], & otherwise, \end{cases} \tag{4.3}$$

such that $g(x^*) \equiv g^*$ for all $x^* \in X^*$ and X^* is included in the orthant

$$\Omega^* := \{x \in \mathbb{R}^n : -\text{sgn}^+(g_i^*)x_i \geq 0, i \in \{1, \cdots, n\}\},$$

where $\text{sgn}^+(t) = 1$ if $t \geq 0$ and $\text{sgn}^+(t) = -1$, otherwise. Assume that x^k is generated by the iterative shrinkage scheme (2.2) with

$$\lambda^k \in (0, 2/\lambda_{\max}), \tag{4.4}$$

where $\lambda_{\max} :=$ maximum eigenvalue of $\nabla^2 f(x) < \infty$. Then the support and the sign of the optimal solution can be identified after a finite number of steps, i.e., $x_i^k = 0$ for $i \in \mathcal{A}_\pm(x^*)$ and $\text{sgn}(x_i^k - \lambda^k g^k) = \text{sgn}(x_i^* - \lambda^k g^*)$ for $i \in T := \mathcal{I}(x^*) \cup \mathcal{A}_0(x^*)$ for k large enough [34]. Let Ω_T^* be the subset of Ω^* with respect to the index set T and $P_{\Omega_T^*}$ be the orthogonal projection onto Ω_T^*. Consequently, for k large enough, the iterative shrinkage scheme (2.2) effectively works only on T and can be shown to equal the gradient projection method $x_T^{k+1} = P_{\Omega_T^*}(x_T^k - \lambda^k \nabla \phi_\mu(x_T^k))$ for solving the subspace minimization problem

$$\min_x \phi_\mu(x) := -(g_T^*)^\top x_T + f(x), \text{ subject to } x_T \in \Omega_T^*, \text{ and } x_i = 0, \forall i \in \mathcal{A}_\pm(x^*). \tag{4.5}$$

Our subspace optimization approach is partially motivated by our belief that a second-order type method might be faster than the iterative shrinkage scheme for solving (4.5). Although λ^k generated by Algorithm "NMLS" might not satisfy (4.4), shrinkage is still able to identify the nondegenerate components $\mathcal{A}_\pm(x^*)$ after a finite number of steps under mild conditions [62]. When $\mathcal{A}(x^k)$ is a good estimate of the true active set $\mathcal{A}(x^*)$, we define subspace optimization as follows. Since $|x_i| = \text{sgn}(x_i)x_i$, $\mu\text{sgn}(x_i^*) = -g_i^*$ and $\text{sgn}(x_i^*) = -\text{sgn}(g_i^*)$ for $i \in \mathcal{I}(x^*)$, we approximate $\text{sgn}(x_i^*)$ by $\text{sgn}(x_i^k)$ and replace $\phi_\mu(x)$ in (4.5) by a smooth function

$$\varphi_\mu(x) := \mu \, \text{sgn}(x_{\mathcal{I}^k}^k)^\top x_{\mathcal{I}^k} + f(x). \tag{4.6}$$

We require that each x_i either has the same sign as x_i^k or is zero, i.e., x is required to be in the set

$$\Omega(x^k) := \left\{x \in \mathbb{R}^n : \text{sgn}(x_i^k)x_i \geq 0, i \in \mathcal{I}(x^k) \text{ and } x_i = 0, i \in \mathcal{A}(x^k)\right\}. \tag{4.7}$$

Therefore, our subspace optimization problem is

$$\min \varphi_\mu(x) \text{ s.t. } x \in \Omega(x^k), \tag{4.8}$$

which can be solved by a limited-memory quasi-Newton method for problems with simple bound constraints (L-BFGS-B) [68]. In our implementations, we also consider subspace optimization without the bound constraints, i.e.,

$$\min_{x \in \mathbb{R}^n} \varphi_\mu(x), \text{ s.t. } x_i = 0, \forall i \in \mathcal{A}(x^k). \tag{4.9}$$

This is essentially an unconstrained minimization problem which can be solved by a linear conjugate gradient method or a limited-memory quasi-Newton method.

We switch to the subspace optimization phase if for some fixed constants $\delta > 0$ and $\epsilon_f, \epsilon_g \in (0, 1)$ either

one of the two conditions

$$\frac{\lambda^{k-1}\|g^k_{\mathcal{I}(x^k)}\|}{\|d^{(\lambda^{k-1})}\|_2} > \delta \text{ and } \|(|g(x^k)| - \mu)_{\mathcal{I}(x^k)\cup\mathcal{A}_0(x^k)}\|_\infty \le \epsilon_g \max(\|x^k\|, 1) \tag{4.10}$$

$$|\psi^{k-1}_\mu - \psi^k_\mu| \le \epsilon_f \max(|\psi^k_\mu|, |\psi^{k-1}_\mu|, 1) \tag{4.11}$$

is satisfied during the shrinkage phase. The justification for tests (4.10) and (4.11) is based on the convergence properties of Algorithm "NMLS". On the one hand, we want to start subspace optimization as soon as possible; on the other hand, we want the active set that defines the subspace optimization problem to be as accurate as possible. If there is at least one nonzero component in x^*, then $\|g^*_{\mathcal{I}^*}\| \ge \mu$ since $|g^*_i| = \mu$ for $i \in \mathcal{I}^*$ from the optimality conditions. Suppose the sequence $\{x^k\}$ generated by the first stage converges to an optimal solution x^* of (1.4); then $g(x^k)$ converges $g(x^*)$ and $\|d^{(\lambda^k)}(x^k)\|_2$ converges to zero from (3.6). Hence, the quantity $\lambda^{k-1}\|g^k_{\mathcal{I}(x^k)}(x^k)\|/\|d^{(\lambda^{k-1})}(x^{k-1})\|_2$ converges to infinity and the first part of condition (4.10) will be satisfied after a finite number of iterations. However, the quantity $\lambda^{k-1}\|g^k_{\mathcal{I}(x^k)}(x^k)\|/\|d^{(\lambda^{k-1})}(x^{k-1})\|_2$ cannot tell us whether the current point x^k is nearly optimal or not. Hence, we also check the second condition in (4.10) in which $\|(|g(x^k)| - \mu)_{\mathcal{I}(x^k)\cup\mathcal{A}_0(x^k)}\|_\infty$ is a measure of optimality (see subsection 4.1). If it happens that the shrinkage phase converges slowly and cannot make sufficient progress after a large number of iterations, the relative change of the objective function value between two consecutive iterations usually will be small. Hence, satisfaction of condition (4.11) indicates that Algorithm "NMLS" is stagnating.

Suppose subspace optimization starts from the point x^k. Clearly, $\varphi_\mu(x^k) = \psi_\mu(x^k)$ from the definition of $\varphi_\mu(x)$. We denote the (approximate) solution of the subspace optimization problem (4.8) by x^{k+1}. Since subspace optimization will not cause a zero component in $\mathcal{A}(x^k)$ to become nonzero and $\mathcal{I}(x^{k+1}) \subset \mathcal{I}(x^k)$, it follows that

$$\varphi_\mu(x^{k+1}) := \mu\text{sgn}(x^k_{\mathcal{I}^k})^\top x^{k+1}_{\mathcal{I}^k} + f(x^{k+1}) \equiv \mu\text{sgn}(x^{k+1}_{\mathcal{I}^{k+1}})^\top x^{k+1}_{\mathcal{I}^k} + f(x^{k+1}) =: \psi_\mu(x^{k+1}).$$

Hence, if we use a decent method to solve (4.8), x^{k+1} will satisfy $\varphi_\mu(x^{k+1}) \le \varphi_\mu(x^k)$, and we can guarantee that there exists at least a sub-sequence generated by the abstract Algorithm 1 that converges. We terminate subspace optimization if the norm of the projected gradient $P_{\Omega^k}(\nabla\varphi(x^{k+1}))$ is small or the relative change of the objective function value between two consecutive iterations is small.

If the active sets provided to two subspace optimizations are identical, we refer to this as a **cycle**. It is hard to detect a cycle in practice unless we store all of the support sets that have been supplied to subspace optimization. However, it is easy to check if there is a cycle between two consecutive subspace optimizations. In such a case, we do not start a second subspace optimization and continue doing the iterative shrinkage.

4.1. Identification of the active set and measures of optimality. The efficiency of our active set algorithm depends how fast and how well the active set is identified. Assume that the sequence $\{x^k\}$ converges to x^*. Then there exists a finite number $\bar{k} > 0$ so that for all $k > \bar{k}$, $\text{sgn}(x^k_i) = \text{sgn}(x^*_i)$ for all $i \in \mathcal{I}(x^*)$ and $|x^k_i| < \epsilon$ for all $i \in \mathcal{A}(x^*)$, if $0 < \epsilon < \min\{|x^*_i|, \forall i \in \mathcal{I}(x^*)\}$. The true nonzero components that are not too small in magnitude can easily be identified. However, the true zero components may be nonzero after many iterations in practice. Hence, the size of the subspace optimization problem which equals the size of the support $\mathcal{I}(x^k)$ can be quite large. One approach is to replace the active set $\mathcal{A}(x^k)$ and the support $\mathcal{I}(x^k)$ by the sets

$$\mathcal{A}(x^k, \xi_k) := \{i \in \{1, \cdots, n\} \,|\, |x^k_i| \le \xi_k\}, \quad \mathcal{I}(x^k, \xi_k) := \{i \in \{1, \cdots, n\} \,|\, |x^k_i| > \xi_k\}, \tag{4.12}$$

where $\xi_k > 0$.

The threshold ξ_k in (4.12) can be simply set to a number $\bar{\xi}_m$ that is approximately equal to the machine accuracy. We now present some criteria for checking optimality which can also be used to choose the value of ξ_k. Let $\mathcal{A}(x^k, \xi_k)$ be divided into two sets

$$\mathcal{A}_{\pm}(x^k, \mu, \xi_k) := \{i \in \mathcal{A}(x, \xi_k) \,|\, |g_i^k| < \mu\} \text{ and } \mathcal{A}_0(x^k, \mu, \xi_k) := \{i \in \mathcal{A}(x^k, \xi_k) \,|\, |g_i^k| \geq \mu\}. \tag{4.13}$$

Then the value

$$\chi(x^k, \mu, \xi_k) := \|(|g^k| - \mu)_{\mathcal{I}(x^k, \xi_k) \cup \mathcal{A}_0(x^k, \mu, \xi_k)}\| \tag{4.14}$$

is a measure of the violation of the first-order optimality conditions (4.3), since $\chi(x^*, \mu, 0) = 0$ follows from the fact that $|g_i^*| = \mu$ for $i \in \mathcal{A}_0(x^*, \mu, 0)$. Suppose that x^* satisfies (4.3). Then the *complementary* conditions $x_i^* \, (|g_i^*| - \mu) = 0, \forall i \in \{1, \cdots, n\}$ also have to be satisfied. Hence,

$$\zeta^k = \|x_i^k \cdot (|g_i^k| - \mu)\| \tag{4.15}$$

provides a measure of the violation of the complementary conditions at the point x^k.

To calculate ξ_k, we use an identification functions

$$\rho(x^k, \xi_k) := \sqrt{\chi(x^k, \mu, \xi_k) + \zeta^k} \tag{4.16}$$

proposed in [27] for nonlinear programming that is based on the amount that the current iterate x^k violates the optimality conditions for (1.4). Specifically, we set the threshold ξ_k initially to $\xi_0 = \bar{\xi}_m$, and then update it as

$$\xi_{k+1} := \min\left(\max\left(\eta_2 \rho(x^k, \xi_k), \bar{\xi}_m\right), \|x^{k+1}\|_1/n\right), \tag{4.17}$$

where $0 < \eta_2 < 1$. Note that, inequality $\xi_{k+1} \geq \|x^{k+1}\|_1/n$ ensure that $\mathcal{I}(x^{k+1}, \xi_{k+1}) \neq \emptyset$.

Since the cardinality of the estimate of the support $|\mathcal{I}(x^k, \xi_k)|$ can be greater than m, we check if $|\mathcal{I}(x^k, \xi_k)| \leq m$ before doing subspace optimization. If $|\mathcal{I}(x^k, \xi_k)| > m$, we set ξ_k to be $|x_{(\Pi)}|$ and recalculate the set $\mathcal{I}(x^k, \xi_k)$, where $x_{(\Pi)}$ is the component of x^k with the Π-th largest magnitude and $1 \leq \Pi \leq m$.

5. The continuation (homotopy) strategy. A continuation (homotopy) procedure sets up an easy problem and gradually transforms it into the original one, creating a path of solutions that converges to the solution of the original problem. Solving the intermediate problems usually is cheaper and this path often provides a good initial point for solving the original problem. Hence, the performance of the basic iterative shrinkage iteration (2.2) can be improved by this strategy. However, we only follow this path inexactly due to the computational cost.

We now describe in detail our method for updating μ_k. First, we check whether the zero vector $\mathbf{0}$ satisfies the first-order optimality conditions (4.3) or not. If the inequality $\|g(\mathbf{0})\|_\infty \leq \mu$ holds, the zero vector is a stationary point. Otherwise, the initial μ_0 is chosen to be $\gamma_1 \|g(\mathbf{0})\|_\infty$, where $0 < \gamma_1 < 1$. For each intermediate value of μ, our algorithm needs only to compute $x(\mu)$ approximately before decreasing μ. Specifically, at the end of iteration k, the next parameter μ_{k+1} is set to a value smaller than μ_k if for

$\epsilon_x \in (0,1)$ the point x^k satisfies the scaled condition

$$\chi(x^k, \mu_k, \xi_k) \leq \epsilon_x \max(\|x^k\|_2, 1) \tag{5.1}$$

which implies that x^k is a good estimate of the solution of the problem $\min_{x\in\mathbb{R}^n} \psi_{\mu_k}(x)$. If x^k is a solution to the subspace optimization problem, we update μ_{k+1} even if the condition (5.1) does not hold. A heuristic strategy for this update is to set

$$\mu_{k+1} = \gamma_1 \|g_{\mathcal{A}(x^k,\mu_k,\xi_k)}\|_\infty,$$

since by (4.3) the norm $\|g_{\mathcal{A}(x^k)}\|_\infty$ converges to a number less than or equal to μ as x^k converges to a stationary point x^*. A fixed fractional reduction of μ_k is also enforced to make sure that continuation will terminate after a finite number of steps. Since the parameter should not be less than μ, we use in our algorithm the updating formula

$$\mu_{k+1} = \max(\gamma_1 \min(\|g_{\mathcal{A}(x^k,\mu_k,\xi_k)}\|_\infty, \mu_k), \mu). \tag{5.2}$$

6. The complete algorithm and default parameter values. The complete pseudo-code for our algorithm FPC_AS (fixed-point continuation active set) is presented in Algorithm 4 below. FPC_AS uses around a dozen parameters, the values of only a few of which are critical to its convergence and performance. The threshold $\bar{\xi}_m = 10^{-10}$ is used to calculate ξ_k in (4.12). The index for the hard truncation is $\Pi = m/2$. It is generally fine to use these default values for $\bar{\xi}_m$ and Π for most CS problems. However, in order to improve the performance of FPC_AS on relatively easy CS problems, one can let $\bar{\xi}_m$ be as large as one thousandth of the smallest magnitude of the nonzero entries of the solution and Π be slightly more than the number of nonzeros in the solution if the quantities or their approximate values are known. The values of parameters related to the activation of subspace optimization are also critical to the performance of FPC_AS. We found that FPC_AS is generally efficient with the values $\delta = 10$ and $\gamma_2 = 10$. The factor to reduce the weight μ_k in continuation is $\gamma_1 = 0.1$. The following parameter values are important but less critical. The parameter values for the nonmonotone line search, $\eta = 0.85$ and $\sigma = 10^{-3}$, are taken from [67]. The maximal and minimal values of the step size for shrinkage are set to $\lambda_m = 10^{-4}$ and $\lambda_M = 10^3$, respectively. The termination rules are $\chi(x^{k+1}, \mu, \xi_k) \leq \epsilon$ or $\chi(x^{k+1}, \mu, \xi_k) \leq \epsilon_x \max(\|x^{k+1}\|, 1)$. The second inequality is especially useful when the magnitude of the solution is large. Default values for $\epsilon, \epsilon_x, \epsilon_g$ are 10^{-6}, 10^{-12} and 10^{-6}, respectively, for noiseless data. We also terminate FPC_AS if the relative change of the objective function value between two consecutive iterations corresponding to the final μ is smaller than $\epsilon_f = 10^{-20}$.

7. Numerical Results. In order to demonstrate the effectiveness of the active-set algorithm FPC_AS (version 1.1), we tested it on three different sets of problems and compared it with the state-of-the-art codes FPC (version 2.0) [34], spg_bp in the software package SPGL1 (version 1.5) [4], and CPLEX [36] with a MATLAB interface [1].

In subsection 7.1, we compare FPC_AS to FPC, spg_bp, and CPLEX for solving the basis pursuit problem (1.2) on a set of "pathological" problems with large dynamic ranges. We note that we are able to solve these problems using CPLEX because they are "small". Three CPLEX solvers are tested; they include the primal simplex method cplex_pp, the dual simplex method cplex_dp, and the barrier method cplex_ip, all of which were applied to a linear programming formulation of (1.2). In subsection 7.2, FPC_AS is compared to spg_bp on various types of synthetic compressive sensing problems. FPC and CPLEX are not included in this

Algorithm 4: FPC_AS Algorithm

Initialization: Set $\mu > 0$. Choose x^0 and set parameters $0 < \lambda_m < \lambda_M < \infty$, $0 < \sigma < 1$, $\bar{\xi}_m > 0$, $\epsilon, \epsilon_x, \epsilon_f, \epsilon_g > 0$, $\delta > 1$, $0 < \eta < 1$, $0 < \gamma_1 < 1$, $\gamma_2 > 1$, $1 \le \Pi \le m$. Set $\mathcal{I}_{sp} = \emptyset$. Set $\mu_0 = \gamma_1 \|g(\mathbf{0})\|_\infty$. Set $C_0 = \psi_\mu(x^0)$, $Q_0 = 1$, and $\xi_0 = \bar{\xi}_m$.

for $k = 0, 1, \cdots$ **do**

begin

S1 Compute a step size for shrinkage: Set $\lambda^k = \max\left\{\lambda_m, \min\left\{\frac{(s^{k-1})^\top s^{k-1}}{(s^{k-1})^\top y^{k-1}}, \lambda_M\right\}\right\}$

S2 Compute a search direction by shrinkage: Set $x^{k+1} = \mathcal{S}(x^k - \lambda^k g^k, \mu_k \lambda^k)$ and $d^k = x^{k+1} - x^k$.

S3 Do a line search: Compute $\Delta_k = (g^k)^\top d^k + \mu_k(\|x^{k+1}\|_1 - \|x^k\|_1)$ and set $\alpha_k = 1$.

if $\psi_{\mu_k}(x^{k+}) > C_k + \sigma\alpha_k\Delta^k$ **then**

Compute $\alpha_k = \arg\min_{\alpha\in[0,1]} \mu_k\|x^k + \alpha d^k\|_1 + f(x^k + \alpha d^k)$ or select α_k satisfying the non-monotone Armijo conditions (3.7).

Set $x^{k+1} = x^k + \alpha_k d^k$

Set $Q_{k+1} = \eta Q_k + 1$, $C_{k+1} = (\eta Q_k C_k + \psi_\mu(x^{k+1}))/Q_{k+1}$.

S4 Compute measures of optimality:

Calculate the sets $\mathcal{I}(x^{k+1}, \xi_k)$ and $\mathcal{A}(x^{k+1}, \xi_k)$ by (4.12). Update the threshold ξ_{k+1} by (4.17).

S5 **if** $\chi(x^{k+1}, \mu, \xi_k) \le \epsilon$ *or* $\chi(x^{k+1}, \mu, \xi_k) \le \epsilon_x \max(\|x^{k+1}\|, 1)$ **then** return the solution

S6 Check rules for doing subspace optimization:

if $\mathcal{I}_{sp} = \mathcal{I}(x^{k+1}, \xi_k)$ **then**

set $do\_sub =$ "$false$"

else if $\frac{\lambda^k \|g^{k+1}_{\mathcal{I}(x^{k+1},\xi_k)}\|}{\|d^{\lambda^k}\|_2} > \delta$ *and* $\chi(x^{k+1}, \mu_k, \xi_k) \le \epsilon_g \max(\|x^{k+1}\|, 1)$ **then**

set $do\_sub =$ "$true$". Set $\delta = \gamma_2\delta$.

else if $\frac{|\psi^{k+1}_{\mu_k} - \psi^k_{\mu_k}|}{\max(|\psi^k_{\mu_k}|, |\psi^{k+1}_{\mu_k}|, 1)} \le \epsilon_f$ **then** set $do\_sub =$ "$true$".

else set $do\_sub =$ "$false$".

end

S7 Do Sub-optimization:

if *"do-sub" = "true"* **then**

if $|\mathcal{I}(x^{k+1}, \xi_k)| > m$ **then** do hard truncation:

Set $\xi_k = |x^{k+1}_{(\Pi)}|$ and recalculate the set $\mathcal{I}(x^{k+1}, \xi_k)$, where $x^{k+1}_{(\Pi)}$ is the component with the Π-th largest magnitude of x^{k+1}.

Set $\mathcal{I}_{sp} = \mathcal{I}(x^{k+1}, \xi_k)$.

Solve the subspace optimization problem (4.9) or (4.8) to obtain a solution x^{k+1}.

if $\chi(x^{k+1}, \mu, \xi_k) \le \epsilon$ *or* $\chi(x^{k+1}, \mu, \xi_k) \le \epsilon_x \max(\|x^{k+1}\|, 1)$ **then** return the solution

S8 Do continuation:

if $(\chi(x^{k+1}, \mu_k, \xi_k) \le \epsilon_x \max(\|x^{k+1}\|, 1)$ *or "do-sub" = "true") and* $\mu_k > \mu$ **then**

Compute $\mu_{k+1} = \max(\gamma_1 \min(\|g_{\mathcal{A}(x^{k+1},\xi_k)}\|_\infty, \mu_k), \mu)$. Set $\delta = \delta_0$.

Set $C_{k+1} = \psi_\mu(x^{k+1})$, $Q_{k+1} = 1$.

else Set $\mu_{k+1} = \mu_k$.

comparison because they take too long to solve all problems. In subsection 7.3, FPC_AS is compared to spg_bp and FPC on the problem of recovering real medical images. It is important to point out that these comparisons are not meant to be a rigorous assessment of the performance of the solvers tested, as this would require very careful handling of subtle details such as parameter tuning and comparable termination criteria, and would be outside the scope of this paper. In addition, a comparison can quickly be out of date since the solvers are continuously improved. The purpose of these comparisons is to demonstrate that FPC_AS is a practical solver for compressive sensing problems.

Our comparison used three different versions of FPC_AS: FPC_AS_CG – the default version that uses the linear conjugate gradient method (CG) to solve the unconstrained subspace optimization problem (4.9), FPC_AS_LB – which uses the limited-memory quasi-Newton method (L-BFGS) to solve (4.9), FPC_AS_BD – which uses L-BFGS-B [68] through the MATLAB wrapper [50] to solve the bound constrained subspace optimization problem (4.8). The default storage number, an important parameter of L-BFGS and L-BFGS-B, is set to 5 and the maximum number of iterations for each call is 50. Other critical parameters of all solvers are reported in each subsection because they vary with the type of problem. We note however, that all of our numerical results for FPC_AS were obtained with all parameters set at their default values except for μ, ϵ and ϵ_x. The main parts of FPC_AS , FPC, and spg_bp were written in MATLAB, and all tests described in this section were performed on a Dell Precision 670 workstation with an Intel Xeon 3.4GHZ CPU and 6GB of RAM running Linux (2.6.9) and MATLAB 7.3.0.

We measured performance by CPU time, relative error, the ℓ_1-norm of the recovered solution, the ℓ_2-norm of the residual, the number of matrix-vector products, as well as the difference between the number of nonzeros in the recovered and exact solutions. In the tables in this section, "CPU" denotes CPU seconds, "rel.err" denotes the relative error between the recovered solution x and the exact sparsest solution $\bar{x}$, i.e., rel.err $= \frac{\|x-\bar{x}\|}{\|\bar{x}\|}$, $\|x\|_1$ denotes the ℓ_1-norm of the recovered solution x, $\|r\|_2 := \|Ax-b\|$ denotes the l_2-norm of the residual, and nMat denotes the total number matrix-vector products involving A or $A^\top$. We also use "nnzx" to denote the number of nonzeros in x which we estimate (as in [5]) by the minimum cardinality of a subset of the components of x that account for 99.5% of $\|x\|_1$, i.e.,

$$\text{nnzx} := \min\left\{ |\Omega| : \sum_{i\in\Omega} |x_i| > 0.995\|x\|_1 \right\} = \min\left\{ k : \sum_{i=0}^{k} |x_{(i)}| \geq 0.995\|x\|_1 \right\} \tag{7.1}$$

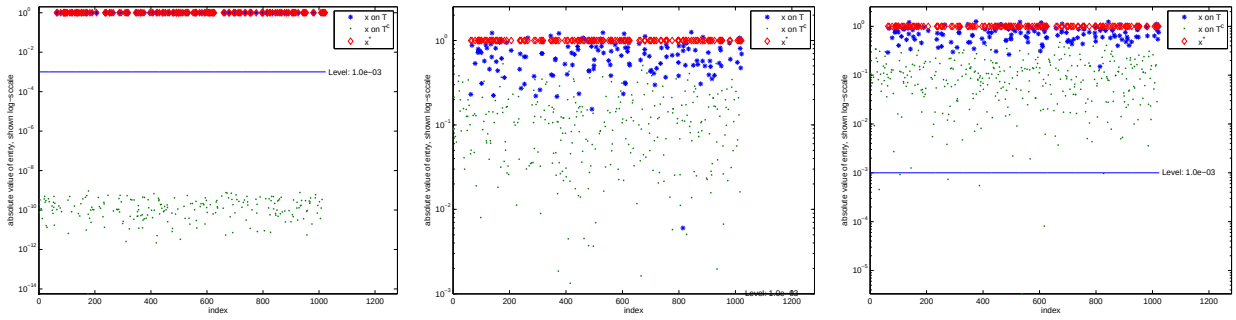
where $x_{(i)}$ is the i-th largest element of x in magnitude, i.e., $|x_{(1)}| \geq \cdots \geq |x_{(n)}|$. This measure is used because solvers often return solutions with the tiny but nonzero entries that can be regarded as zero. In order to compare the supports of x and $\bar{x}$, we first remove tiny entries of x by setting all of its entries with a magnitude smaller than $0.1|\bar{x}_m|$ to zero, where $\bar{x}_m$ is the smallest entry of $\bar{x}$ in magnitude, and then compute the quantities "sgn", "miss" and "over", where "sgn":$= |\{i, x_i\bar{x}_i < 0\}|$ denotes the number of corresponding entries of x and $\bar{x}$ that are both nonzero but have opposite signs, "miss" $:= |\{i, x_i = 0, \bar{x}_i \neq 0\}|$ denotes the number of zero entries in x with a corresponding nonzero entry in $\bar{x}$, and "over":$= |\{i, x_i \neq 0, \bar{x}_i = 0\}|$ denotes the number of nonzero entries in x with a corresponding zero entry in $\bar{x}$. If x matches $\bar{x}$ in term of support and sign, the values of "sgn", "miss" and "over" should all be zero.

7.1. Recoverability for some "pathological" problems. In order to demonstrate the robustness of FPC_AS we tested it on a set of small-scale, "pathological" problems. Only the performance of FPC_AS_CG is reported because FPC_AS_LB and FPC_AS_BD performed similarly. The first test set includes four problems CaltechTest1, CaltechTest2, CaltechTest3 and CaltechTest4 [11], which are "pathological" because the magnitudes of the nonzero entries of the exact solutions $\bar{x}$ lie in a large range, i.e., the largest magnitudes are significantly larger than the smallest magnitudes. The characteristics of these problems are summarized in Table 7.1, in which the last column gives the distinct orders of magnitudes O_i of the nonzeros entries in $\bar{x}$, as well as the number of elements N_i of x that are of order of magnitude O_i in the form of (O_i, N_i) pairs. For example, for the problem CaltechTest3, "$(10^{-1}, 31), (10^{-6}, 1)$" means that there are thirty-one entries in $\bar{x}$ with a magnitude of 10^{-1} and one entry with a magnitude of 10^{-6}. Such "pathological" problems are exaggerations of a large number of realistic problems in which the signals have both large and small

TABLE 7.1
Problem information

Problem	n	m	K	(magnitude, num. of elements on this level)
CaltechTest1	512	128	38	$(10^5, 33)$, $(1, 5)$
CaltechTest2	512	128	37	$(10^5, 32)$, $(1, 5)$
CaltechTest3	512	128	32	$(10^{-1}, 31)$, $(10^{-6}, 1)$
CaltechTest4	512	102	26	$(10^4, 13)$, $(1, 12)$, $(10^{-2}, 1)$
Ameth6Xmeth2K150	1024	512	150	$(1, 150)$
Ameth6Xmeth2K151	1024	512	151	$(1, 151)$

FIG. 7.1. *Recovered solutions of "Ameth6Xmeth2K150"*



(a) FPC_AS_CG (b) FPC (c) spg_bp

entries. The second test set includes two problems Ameth6Xmeth2K150 and Ameth6Xmeth2K151. Problem "Ameth6Xmeth2K150" is obtained by fixing the first nonzero component of the signal $\bar{x}$ in problem "Ameth6Xmeth2K151" to zero. These two problems are difficult because the numbers of nonzero elements in their solutions are around the limit of where the l_0-minimization problem (1.1) is equivalent to the basis pursuit problem (1.2). The coefficient matrix A here is the partial discrete cosine transform (DCT) matrix whose m rows were chosen randomly from the $n \times n$ DCT matrix.

We compared the results from FPC_AS_CG with the results from the solver FPC and the solver spg_bp. We set the termination criteria sufficiently small for each solver. Specifically, we set the parameters $xtol = 10^{-10}$, $gtol = 10^{-8}$ and $mxitr = 2 * 10^4$ for FPC, the parameters $bpTol = 10^{-10}$, $optTol = 10^{-10}$, $decTol = 10^{-10}$ and $iteration = 10^4$ for spg_bp and the parameters $\mu = 10^{-10}$, $\epsilon = 10^{-12}$ and $\epsilon_x = 10^{-16}$ for FPC_AS_CG. All other parameters of each solver were set to their default values. The termination criteria are not directly comparable due to the different formulations of the problems used by the solvers, but we believe that on average the chosen criteria for FPC_AS_CG is tighter than those of the other two solvers.

A summary of the computational results for all of the six problems is presented in Table 7.2. From that table, the superiority of FPC_AS_CG is obvious on this set of problems. For the first five problems, the solutions of the basis pursuit problem (1.2) are the same as the sparsest signal $\bar{x}$ if we trust the solutions obtained by CPLEX. It is interesting that FPC_AS_CG is faster than the three variants of CPLEX on all of the first five problems with all four codes achieving comparable accuracy. This is most obvious on problem "Ameth6Xmeth2K150", on which FPC_AS_CG exhibited an approximately 30-fold improvement in terms of CPU time over the fastest variant cplex_dp of CPLEX. FPC_AS_CG also performs significantly better than FPC and spg_bp in terms of both CPU time and the total number of matrix-vector products on these problems. Both FPC and spg_bp failed to identify the sparsest solution of "Ameth6Xmeth2K150" and the corresponding recovered solutions are depicted in Figure 7.1. Adjusting other parameters of FPC and spg_bp might give better results, but that is outside the scope of this paper.

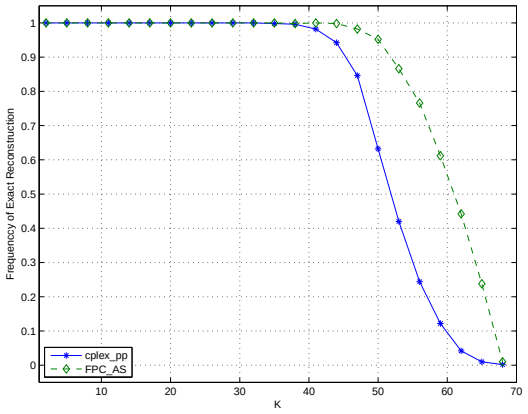
TABLE 7.2

Computational results for the difficult problems. The matrices A are constructed explicitly by the command "dctmtx" of MATLAB for all of the solvers.

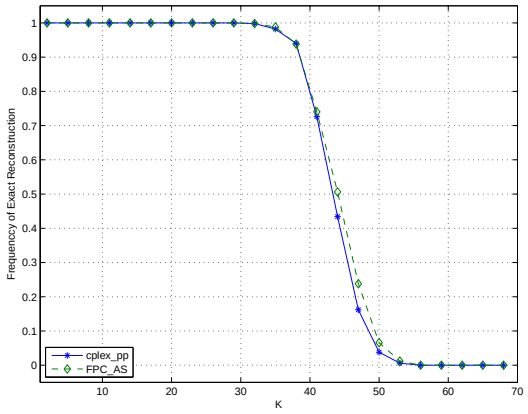
Problem	solver	CPU(sec.)	rel.err	$\|x\|_1$	$\|r\|_2$	nMat	nnzx	(sgn,miss,over)
CaltechTest1	FPC_AS_CG	0.105	5.04e-12	3.300e+06	1.67e-09	441	33	(0, 0, 0)
	FPC	12.048	3.09e-06	3.300e+06	1.78e-01	40001	33	(0, 0, 18)
	spg_bp	11.552	3.34e-06	3.300e+06	3.63e-03	29733	33	(0, 0, 29)
	cplex_pp	0.245	5.10e-12	3.300e+06	8.87e-10		33	(0, 0, 0)
	cplex_dp	0.302	5.52e-12	3.300e+06	1.04e-08		33	(0, 0, 0)
	cplex_ip	0.778	5.18e-12	3.300e+06	5.34e-09		33	(0, 0, 0)
CaltechTest2	FPC_AS_CG	0.102	7.44e-14	3.200e+06	1.75e-09	322	32	(0, 0, 0)
	FPC	12.204	8.57e-08	3.200e+06	9.56e-03	40001	32	(0, 0, 0)
	spg_bp	11.227	8.47e-10	3.200e+06	3.71e-06	29238	32	(0, 0, 0)
	cplex_pp	0.258	1.91e-13	3.200e+06	7.05e-10		32	(0, 0, 0)
	cplex_dp	0.245	2.27e-13	3.200e+06	4.68e-09		32	(0, 0, 0)
	cplex_ip	0.857	3.79e-13	3.200e+06	6.95e-10		32	(0, 0, 0)
CaltechTest3	FPC_AS_CG	0.067	1.51e-09	6.200e+00	1.26e-09	249	31	(0, 0, 0)
	FPC	11.471	3.54e-05	6.200e+00	9.18e-06	40001	31	(0, 1, 46)
	spg_bp	6.685	4.31e-09	6.200e+00	9.99e-11	17346	31	(0, 0, 0)
	cplex_pp	0.217	4.56e-09	6.200e+00	1.21e-13		31	(0, 0, 0)
	cplex_dp	0.175	4.56e-09	6.200e+00	1.45e-13		31	(0, 0, 0)
	cplex_ip	0.547	4.56e-09	6.200e+00	1.47e-13		31	(0, 0, 0)
CaltechTest4	FPC_AS_CG	0.131	5.75e-13	1.300e+05	3.51e-09	498	13	(0, 0, 0)
	FPC	11.255	1.60e-07	1.300e+05	1.11e-03	40001	13	(0, 0, 0)
	spg_bp	8.330	3.77e-12	1.300e+05	3.81e-10	22824	13	(0, 0, 0)
	cplex_pp	0.141	1.24e-13	1.300e+05	3.28e-11		13	(0, 0, 0)
	cplex_dp	0.182	2.82e-13	1.300e+05	3.13e-11		13	(0, 0, 0)
	cplex_ip	0.548	2.64e-13	1.300e+05	3.96e-11		13	(0, 0, 0)
Ameth6Xmeth2K150	FPC_AS_CG	0.362	7.25e-10	1.500e+02	2.26e-09	448	150	(0, 0, 0)
	FPC	60.765	4.84e-01	1.414e+02	3.42e-01	40001	424	(0, 1, 170)
	spg_bp	50.670	4.29e-01	1.500e+02	5.34e-03	29346	452	(0, 0, 167)
	cplex_pp	19.815	1.02e-12	1.500e+02	1.30e-12		150	(0, 0, 0)
	cplex_dp	11.053	9.70e-13	1.500e+02	1.41e-12		150	(0, 0, 0)
	cplex_ip	22.062	2.73e-12	1.500e+02	3.62e-12		150	(0, 0, 0)
Ameth6Xmeth2K151	FPC_AS_CG	0.377	7.45e-10	1.510e+02	2.27e-09	446	151	(0, 0, 0)
	FPC	61.872	4.77e-01	1.487e+02	8.74e-02	40001	451	(0, 1, 181)
	spg_bp	52.293	4.86e-01	1.508e+02	4.68e-03	29704	461	(0, 1, 190)
	cplex_pp	19.536	4.85e-01	1.509e+02	9.40e-14		460	(0, 1, 189)
	cplex_dp	11.828	4.85e-01	1.509e+02	9.40e-14		460	(0, 1, 189)
	cplex_ip	23.398	4.85e-01	1.509e+02	9.40e-14		460	(0, 1, 189)

The solutions obtained by CPLEX in Table 7.2 show that the solution of the basis pursuit version (1.2) of problem "Ameth6Xmeth2K151" is not sparse. The minimum objective function value obtained by all CPLEX variants was 150.9 implying that the minimal objective value of (1.2) is 150.9. This is pretty close to but not equal to the cardinality 151 of the sparse solution $\bar{x}$. Hence the equivalence between (1.1) and (1.2) does not hold for this example. This partly explains why it is difficult to recover the solution to "Ameth6Xmeth2K150" since there is only one element in the original signals $\bar{x}$ of these two examples that is different. However, FPC_AS_CG is able to find the sparse solution successfully because of the hard truncation that is applied before subspace optimization if the iterate is not sparse. Since FPC_AS_CG will eventually fail to recover a sparse solution if the exact minimum of the l_0-minimization problem is not sparse, we investigate the recoverability success of FPC_AS_CG and cplex_pp with respect to Gaussian sparse signals and zero-one sparse signals, by using a partial DCT matrix with $n = 256$ and $m = 128$. Figure 7.2 depicts the empirical frequency of exact reconstruction. The numerical values on the x-axis denote the sparsity level K, while the numerical values on the y-axis represent the reconstruction rate, i.e., the fraction of 500 instances that are recovered with a relative error less than 10^{-2}. From the figure, we observe that for Gaussian signals, FPC_AS_CG was able to continue recovering sparse signals even after the equivalence between the ℓ_0- and ℓ_1-minimization problems started to break down as cplex started to find non-sparse solutions. We will further

FIG. 7.2. *Reconstruction rate (500 replications): $m = 128$, $n = 256$*



(a) Gaussian signals: FPC_AS_CG is able to recover the sparse solution until $K \geq 41$, while the comparison with cplex_pp indicates that the equivalence between the ℓ_0 and ℓ_1 minimization problems begins to break down when $K \geq 35$.



(b) zero-one signals: both FPC_AS_CG and cplex_pp start to fail to reconstruct the signal when $K \geq 29$.

explain this phenomenon in Remark 7.1.

REMARK 7.1. *The solution of the ℓ_1-regularized problem* (1.4) *should be close to the minimizer of the BP problem* (1.2) *if μ is small enough from the theory of penalty functions. However, due to the hard truncation in Step S7 of Algorithm 4, FPC_AS can return a different solution than a minimizer of* (1.2) *when the latter is not sufficiently sparse, as is demonstrated by the last problem in Table 7.2. We introduced this hard truncation heuristic based on two observations from our experiments. First, this strategy can make FPC_AS run significantly faster on problems with sparse solutions. Second, it frequently enhances the algorithm's ability in recovering sparse signals even after the equivalence between ℓ_0- and ℓ_1-optimizations no longer holds, as is demonstrated by Figure 7.2(a).*

7.2. Quality of Compressed Sensing Reconstruction. In this subsection, we evaluate the suitability of FPC_AS for compressive sensing on some randomly generated problems. Given the dimension of the signal n, the number of observations m and the number of nonzeros K, we generated a random matrix A and a random vector $\bar{x}$ as follows. First, the type of matrix A was chosen from:

Type 1: Gaussian matrix whose elements are generated from i.i.d normal distributions $\mathcal{N}(0, 1)$;

Type 2: Orthogonalized Gaussian matrix whose rows were orthogonalized using a QR decomposition;

Type 3: Bernoulli matrix whose elements are generated from $+/-$ 1 distribution;

Type 4: Hadamard matrix H which is a matrix of 1's and -1's whose columns are orthogonal;

Type 5: Partial discrete cosine transform matrix.

We randomly selected m rows from these matrices to construct the matrix A. To avoid potential numerical issues, we scaled the matrix A constructed from matrices of types 1, 3 and 4 by the largest eigenvalue of $AA^\top$. To generate the signal $\bar{x}$, we first generated the support by randomly selecting K indices between 1 and n, and then assigned a value to x_i for each i in the support by one of the following eleven methods:

Type 1: a normally distributed random variable (Gaussian signal);

Type 2: a uniformly distributed random variable in (-1,1);

Type 3: one (zero-one signal);

Type 4: the sign of a normally distributed random variable;

TABLE 7.3
Robustness results for the exact data

Solver	nMat ($\leq 10^3$)		$\|r\|_2$ ($\leq 10^{-6}$)		rel.err ($\leq 10^{-8}$)	
	num.	per.	num.	per.	num.	per.
FPC_AS_CG	329	99.70	330	100.00	329	99.70
FPC_AS_LB	328	99.39	330	100.00	329	99.70
FPC_AS_BD	295	89.39	330	100.00	327	99.09
spg_bp	171	51.82	320	96.97	264	80.00

Type 5,6,7,8: Type **1, 2, 3, 4** scaled by 10^5, respectively

Type 9: Type **4** but half of the elements in the support are scaled by 10^5;

Type 10: a signal x with power-law decaying entries (also known as compressible sparse signals) whose components satisfy $|x_i| \leq c_x \cdot i^{-p}$, where we take $c_x = 10^5$ and $p = 1.5$.

Type 11: a signal x with exponentially decaying entries whose components satisfy $|x_i| \leq c_x \cdot e^{-pi}$, where we take $c_x = 1$ and $p = 0.005$.

Finally, the observation b was computed as $b = A\bar{x}$. The matrices of types 1, 2, 3 and 4 were stored explicitly and we tested signals with three different sizes $n = 2^{10}, 2^{11}, 2^{12}$. The matrices of type 5 were stored implicitly and we tested signals with three different sizes $n = 2^{10}, 2^{12}, 2^{15}$. Given n, we set the number of observations $m = n/2$ and the number of nonzeros $K = \text{round}(\rho m)$ for $\rho = 0.2$ and 0.3. The above procedure gave us a total of 330 problems.

Since spg_bp has been proved to be robust in many different applications [5], we continue to compare FPC_AS with spg_bp in this subsection. We set the parameters $bpTol = 10^{-8}$, $optTol = 10^{-8}$ and $decTol = 10^{-8}$ for spg_bp and the parameters $\mu = 10^{-10}$, $\epsilon = 10^{-12}$ and $\epsilon_x = 10^{-16}$ for the three variants of FPC_AS. All other parameters of each solver were set to their default values.

We present several statistics on robustness of these solvers in the Table 7.3. In the second column, we present the number (num.) and percentage (per.) of the problems that were solved within 1000 matrix-vector products by each solver. We present the number and percentage of problems for which the norms of the computed residual were less than 10^{-6} and the relative errors between the solution x and the exact sparsest solution $\bar{x}$ were less than 10^{-8}, in the third and fourth column, respectively. The active-set algorithms required less matrix-vector products to achieve a higher reconstruction rate than spg_bp on average.

We now illustrate the numerical results using the performance profiles as proposed in [20]. These profiles provide a way to graphically present the comparison of the quantities $t_{p,s}$, such as the number of iterations or required CPU time, obtained for each problem p and each solver s. Define $r_{p,s}$ to be the ratio between the quantity $t_{p,s}$ obtained on problem p by solver s over the lowest such quantity obtained by any of the solvers on problem p, i.e.,

$$r_{p,s} := \frac{t_{p,s}}{\min\{t_{p,s} : 1 \leq s \leq n_s\}}.$$

Whenever solver s fails to solve problem p, the ratio $r_{p,s}$ is set to infinity or some sufficiently large number. Then

$$\pi_s(\tau) := \frac{\text{number of problems where } \log_2(r_{p,s}) \leq \tau}{\text{total number of problems}}, \quad \tau \geq 0,$$

is the fraction of the test problems that were solved by solver s within a factor $2^\tau \geq 1$ of the performance obtained by the best solver. The performance plots present π_s for each solver s as a function of τ.

TABLE 7.4
Robustness results for the noisy data

Solver	nMat ($\leq 10^3$)		$\|r\|_2$ ($\leq 10^{-4}$)		rel.err ($\leq 10^{-2}$)	
	num.	per.	num.	per.	num.	per.
FPC_AS_CG	327	99.09	283	85.76	278	84.24
FPC_AS_LB	309	93.64	301	91.21	278	84.24
FPC_AS_BD	291	88.18	309	93.64	280	84.85
spg_bp	193	58.48	194	58.79	246	74.55

A performance plot for the CPU time is presented in Figure 7.3(a). All three variants of the active-set algorithm appear to be faster than spg_bp. A performance plot for the total number of matrix-vector products involving A or $A^\top$ is presented in 7.3(b). The variant FPC_AS_CG requires overall fewer matrix-vector products for the given test set. Figure 7.3(c) presents a performance plot for the ℓ_1-norm $\|x\|_1$ achieved by each solver. It shows that the objective function values obtained by all four solvers are essentially identical. Figure 7.3(d) compares the l_2-norms of the residual $\|r\|_2$ obtained by the solvers. Figure 7.3(e) presents a comparison of the ratio of the number of nonzero components recovered over the number of the nonzero components in the sparsest solution. Here, as in Figure 7.3(c), there appears to be no discernable difference between these solvers. The relative error between the recovered solution and the exact solution is depicted in Figure 7.3(f). On this measure, FPC_AS_CG and FPC_AS_BD seem to perform better than the other solvers.

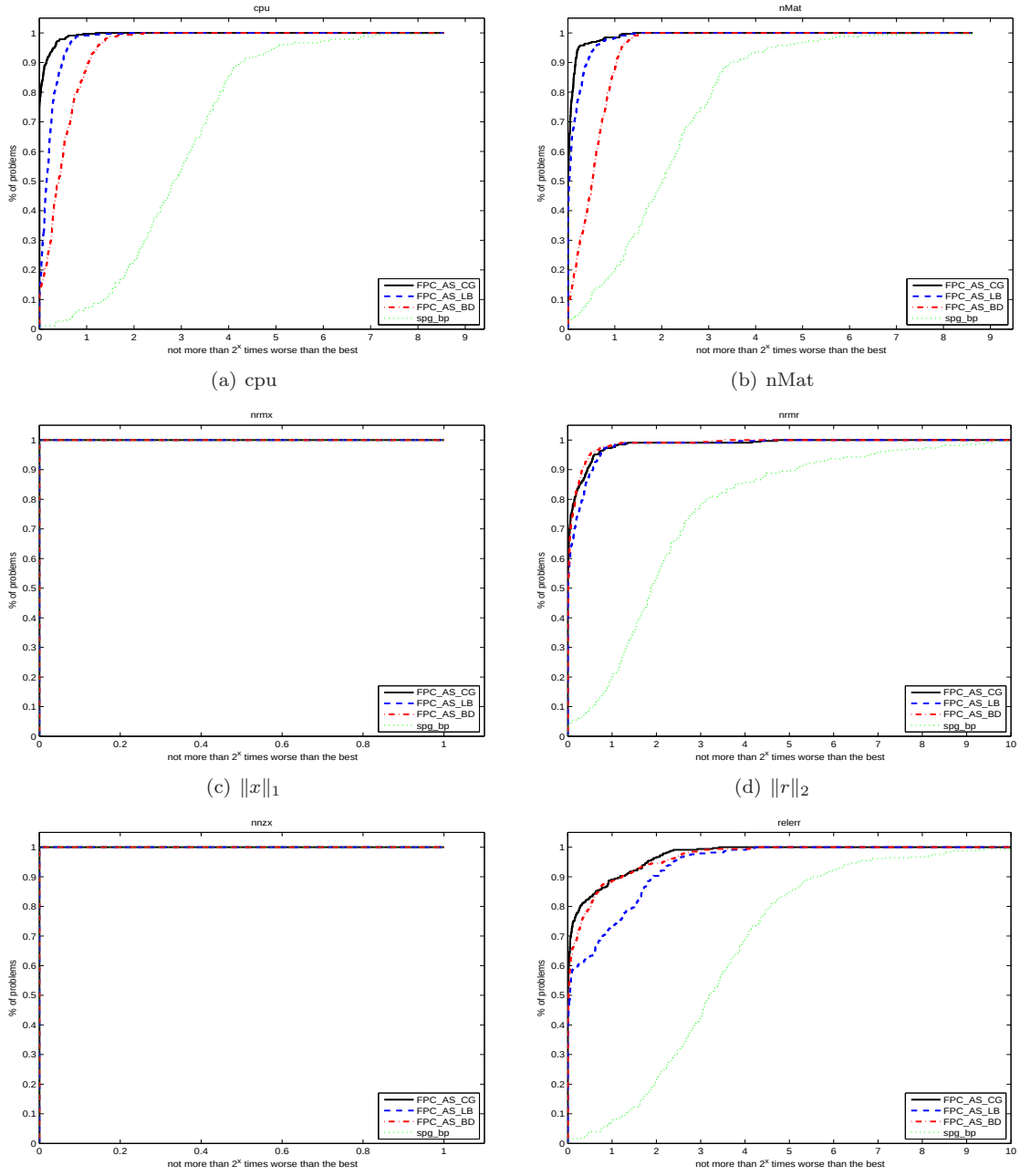
Since real problems usually are corrupted by noise, we tested the recoverability of each solver on the same data set after adding Gaussian noise. Specifically, we let

$$b = A(\bar{x} + \epsilon_1) + \epsilon_2,$$

where ϵ_1 and ϵ_2 are vectors whose elements are i.i.d distributed as $N(0, \sigma_1^2)$ and $N(0, \sigma_2^2)$, respectively. Here $\sigma_1 = 10^{-3}$ and $\sigma_2 = 10^{-3}$. In this test, we required less accurate solutions of problem (1.4) due to the noise. We set the parameters $bpTol = 10^{-4}$, $optTol = 10^{-4}$ and $decTol = 10^{-4}$ for spg_bp and the parameters $\mu = 10^{-10}$, $\epsilon = 10^{-6}$ and $\epsilon_x = 10^{-12}$ for the three variants of FPC_AS. Our computational results are summarized in Table 7.4 which displays similar patterns to Table 7.3. We present the performance profiles in Figure 7.4 which show that the active-set family took less CPU time and fewer matrix-vector products to obtain smaller residuals and relative errors. In particular, FPC_AS_BD seems to be the most robust in the sense that it obtained solutions with the smallest relative error.

Finally, we present some statistics on the performance of the three FPC_AS variants. For each problem, let "nSubOpt" denote the number of sub-optimizations and "nCont" denote the number of continuations called by the active-set algorithm. The mean and standard deviation of "nSubOpt" and "nCont" over all of the randomly generated problems are reported in the second and third columns of Table 7.5, respectively. The average number of continuations is approximately 4 and 6 in the exact case and in the noisy case, respectively, for all variants. The average number of subspace optimizations is greater than the average number of continuations. The percentages of the matrix-vector products spent on various tasks in the shrinkage and subspace optimization stages, are also presented. The mean and standard deviation of these percentages for the two stages are reported in the fourth and fifth columns of Table 7.5, respectively. It is obvious that most of the matrix-vector products are spent on solving subspace optimization in the exact case. Since the solutions of problems with noisy data may not be sparse, a highly accurate solution of the ℓ_1-regularized problem might not be better than an approximate solution with a low accuracy. Hence, how to choose a proper termination rule is an important issue.

FIG. 7.3. *Performance profiles for exact data*



(a) cpu (b) nMat

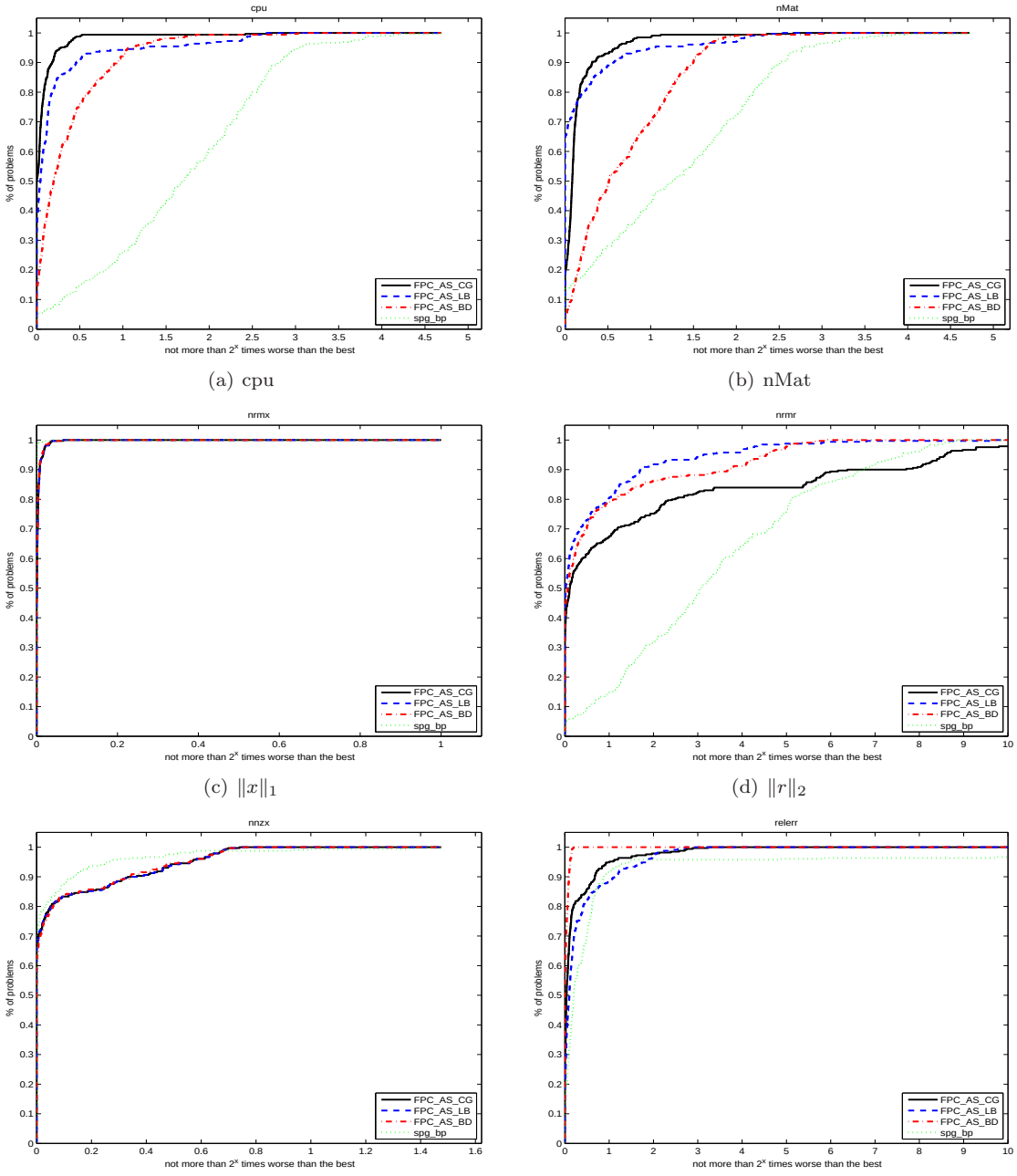
(c) $\|x\|_1$ (d) $\|r\|_2$

(e) nnzx (f) rel.err

We conclude this subsection with some comments and observations, based on Figures 7.3 and 7.4, Tables 7.2, 7.3, 7.4 and 7.5.

1. The three variants of FPC_AS performed very robustly and efficiently on the set of randomly generated compressive sensing problems described in this paper. FPC_AS_CG was the fastest variant in terms of CPU time and matrix-vector products while FPC_AS_BD was more robust than FPC_AS_CG in terms of relative errors. This shows that restricting the signs of the solution in the subspace optimization problem by adding the bound constraints in (4.8) might be helpful.

FIG. 7.4. *Performance profile for noisy data*



(a) cpu (b) nMat

(c) $\|x\|_1$ (d) $\|r\|_2$

(e) nnzx (f) rel.err

2. The conjugate gradient method (CG) can be less efficient if the columns selected from the matrix A in the subspace optimization problem are or almost are linearly dependent. This kind of potentially ill-conditioned system might cause trouble to the CG method. (We should point out that the function "PCG" in MATLAB (version 2008a or earlier) computes two matrix-vector products at each iteration while only one of such product is needed. This bug has been corrected in our implementation of "CG".)
3. The iterative shrinkage scheme is able to identify the signs of the components of x fairly quickly for

TABLE 7.5
Statistics of the variants of FPC_AS

Exact data								
Solver	nSubOpt		nCont		nMat in Shrinkage (per.)		nMat in Sub-Opt (per.)	
	mean	std.	mean	std.	mean	std.	mean	std.
FPC_AS_CG	3.66	2.12	3.37	2.08	0.39	0.06	0.61	0.07
FPC_AS_LB	3.98	2.53	3.57	2.17	0.38	0.08	0.61	0.08
FPC_AS_BD	4.95	3.71	3.83	2.15	0.36	0.12	0.64	0.13
Noisy data								
Solver	nSubOpt		nCont		nMat in Shrinkage (per.)		nMat in Sub-Opt (per.)	
	mean	std.	mean	std.	mean	std.	mean	std.
FPC_AS_CG	6.49	4.40	6.19	1.83	0.57	0.12	0.43	0.12
FPC_AS_LB	7.93	8.43	6.34	1.91	0.59	0.12	0.41	0.13
FPC_AS_BD	6.16	4.28	5.85	1.57	0.37	0.10	0.63	0.10

a fixed μ_k but it can take a lot of iterations to recover their magnitudes. On the contrary, subspace optimization depends on a good active set and is able to recover the magnitude of the x_i fairly well by applying second-order methods. Our active-set algorithms combines the good features of these two different schemes.

4. From our limited numerical experience, FPC_AS performs better than the fixed-point solver FPC [33]. It also performs better than some other solvers like the projected gradient solver GPSR [29] and the interior-point algorithm $\ell_1\_\ell_s$ [37] on the test problems in this paper. A comparison with greedy algorithms would also be very interesting. We did not include these comparisons to keep the length of the paper within bounds.

7.3. Realistic Examples. In this subsection, we demonstrate the efficiency of FPC_AS_CG for solving the l_1-regularized problem on six images: a Shepp-Logan phantom available through the Matlab Image Processing Toolbox, and five medical images (three MRI's and two CT scans) in the public domain. These signals have relatively sparse representations in Haar wavelets; that is, there exists an $x^* \in \mathbb{R}^n$ such that $z = Wx^*$ for a true signal $z \in \mathbb{R}^n$, where $W \in \mathbb{R}^{n\times n}$ is the Haar wavelet basis and x^* is approximately sparse. The measurements are constructed as $b = \bar{A}z$, where $\bar{A} \in \mathbb{R}^{m\times n}$ is the partial discrete cosine transformation and the number of observations $m = \tau n$ with $\tau = m/n = 0.25, 0.50, 0.75$. We then obtain approximate wavelet coefficients x of z by solving the ℓ_1-regularized problem (1.4) with $A = \bar{A}W$ and $\mu = 10^{-3}$. Finally, we complete the recovery by computing $\hat{z} = Wx$. We compared FPC_AS with FPC and spg_bp. Since x^* is not really sparse and is corrupted by noise, we use a relative large termination criteria. We use the default parameters for FPC and the parameters $bpTol = 10^{-3}$, $optTol = 10^{-3}$ and $decTol = 10^{-3}$ for spg_bp. For FPC_AS_CG, we set the tolerances $\epsilon = 10^{-3}$, $\epsilon_x = 10^{-6}$ and the maximal iteration number for subspace optimization to 10. The reconstruction results are summarized in Table 7.6. In this table, the relative error is that between the true image z and the recovered image $\hat{z}$. FPC_AS_CG is considerably faster than FPC and spg_bp in terms of CPU time and the number of matrix-vector products to achieve comparable relative errors except on "phantom" with $\tau = 0.75$ (although it is still faster). The reconstructed images obtained by FPC_AS_CG are depicted in Figure 7.5. In each row of Figure 7.5, the first image is the original image with a caption stating its resolution. The second, third and fourth are recovered images with respect to $\tau = 0.25, 0.5, 0.75$, respectively, together with a caption stating the relative errors between the true image z and the recovered image $\hat{z}$.

8. Conclusions. A two stage active-set algorithm with continuation for the ℓ_1-norm regularized optimization is presented and tested. It starts with an easier problem and strategically applies a decreasing

TABLE 7.6
Statistics of recovering medical images

Problem		FPC			spg_bp			FPC_AS_CG		
	τ	CPU	nMat	rel.err	CPU	nMat	rel.err	CPU	nMat	rel.err
ct_thighs	0.25	45.78	403	5.7e-02	70.49	433	6.4e-02	16.30	130	5.9e-02
ct_thighs	0.50	31.34	263	9.8e-03	71.21	394	1.0e-02	15.63	120	1.0e-02
ct_thighs	0.75	17.60	147	3.2e-03	39.14	210	3.0e-03	11.28	81	3.9e-03
ct_thorax	0.25	50.80	451	7.2e-02	79.86	494	7.5e-02	18.28	142	7.1e-02
ct_thorax	0.50	37.06	315	1.9e-02	70.75	391	1.9e-02	16.63	126	2.0e-02
ct_thorax	0.75	23.55	195	6.0e-03	60.98	325	6.2e-03	16.08	116	7.3e-03
mri_abdomen	0.25	10.36	531	1.9e-01	18.64	606	2.1e-01	2.77	146	1.9e-01
mri_abdomen	0.50	8.28	417	9.7e-02	17.93	519	9.7e-02	2.26	124	9.0e-02
mri_abdomen	0.75	4.80	233	3.9e-02	11.74	322	4.0e-02	2.07	110	4.5e-02
mri_brain	0.25	46.98	417	7.3e-02	98.50	606	7.8e-02	17.46	136	7.1e-02
mri_brain	0.50	38.02	325	2.8e-02	72.28	400	3.0e-02	14.87	116	3.0e-02
mri_brain	0.75	21.18	175	7.9e-03	65.19	336	8.3e-03	18.81	141	7.8e-03
mri_pelvis	0.25	11.88	613	1.5e-01	15.26	501	1.5e-01	2.52	138	1.4e-01
mri_pelvis	0.50	7.22	379	7.3e-02	10.94	327	7.4e-02	2.58	134	7.5e-02
mri_pelvis	0.75	4.01	205	2.7e-02	9.14	258	3.1e-02	2.33	112	4.8e-02
phantom	0.25	3.47	799	3.6e-01	4.41	586	3.8e-01	1.04	136	3.6e-01
phantom	0.50	2.23	569	1.5e-01	3.72	506	1.6e-01	0.62	126	1.6e-01
phantom	0.75	1.06	259	2.7e-03	1.46	236	2.4e-03	0.58	116	4.1e-03

sequence of weights μ_k to the ℓ_1-norm term in the objective to gradually transform this easier problem to the given, more difficult problem with the prescribed regularization weight μ. Shrinkage is performed iteratively until the support of the current point becomes a good estimate to the support of the solution corresponding to the current weight. This estimate is used to define a subset of the solution domain over which a smaller subspace optimization problem is solved to yield a relatively accurate point. Usually, after only a small number of subproblems, a solution of high accuracy can be obtained. At each iteration of shrinkage in the first stage, a search direction is generated along with an automatically adjusting step-size parameter λ, and either an exact or an inexact line search is carried out to guarantee global convergence. In the second stage, the subspace optimization problem has a simple objective and may include bound constraints to restrict the signs of decision variables. The numerical results presented in section 7 demonstrate the effectiveness of the algorithm for solving compressive sensing problems of varying difficulties.

In [62], we present a convergence analysis for the algorithm.

Acknowledgements. We want to thank Shiqian Ma, Lifeng Chen and Stephen Becker for their helpful discussions and comments. We want to thank Elaine T. Hale for providing us the code for accessing the medical images in subsection 7.3.

REFERENCES

[1] M. BAOTIC AND M. KVASNICA, *Cplexint - matlab interface for the cplex solver*, 2006.

[2] J. BARZILAI AND J. M. BORWEIN, *Two-point step size gradient methods*, IMA J. Numer. Anal., 8 (1988), pp. 141–148.

[3] J. BECT, L. BLANC-FERAUD, G. AUBERT, AND A. CHAMBOLLE, *A ℓ_1-unified variational framework for image restoration*, European Conference on Computer Vision, Prague, Lecture Notes in Computer Sciences 3024, (2004), pp. 1–13.

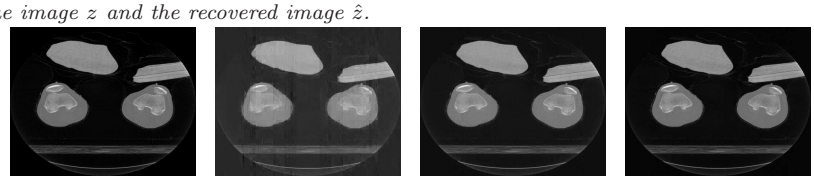
[4] E. V. BERG AND M. P. FRIEDLANDER, *SPGL1: A solver for large-scale sparse reconstruction*, June 2007. http://www.cs.ubc.ca/labs/scl/spgl1.

[5] E. V. BERG AND M. P. FRIEDLANDER, *Probing the Pareto frontier for basis pursuit solutions*, Tech. Rep. TR-2008-01, Department of Computer Science, University of British Columbia, January 2008. To appear in *SIAM J. Sci. Comp.*

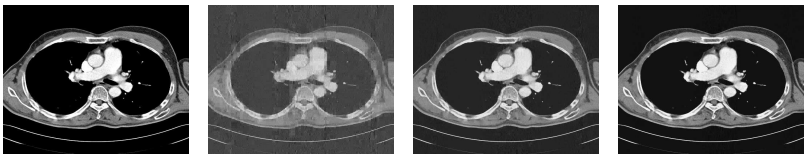
[6] J. BIOUCAS-DIAS AND M. FIGUEIREDO, *Two-step algorithms for linear inverse problems with non-quadratic regularization*, IEEE International Conference on Image Processing ICIP' 2007, San Antonio, TX, (2007).

[7] J. V. BURKE AND J. J. MORÉ, *On the identification of active constraints*, SIAM J. Numer. Anal., 25 (1988), pp. 1197–1211.

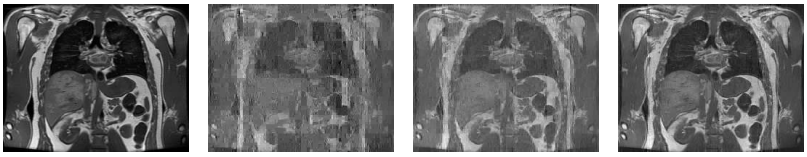
FIG. 7.5. *Medical image recovery by FPC_AS_CG. In each row, (a) is the original image with a caption stating its resolution, and (b), (c) and (d) are the recovered images for $\tau = 0.25, 0.5, 0.75$, respectively, together with a caption stating the relative errors between the true image z and the recovered image $\hat{z}$.*



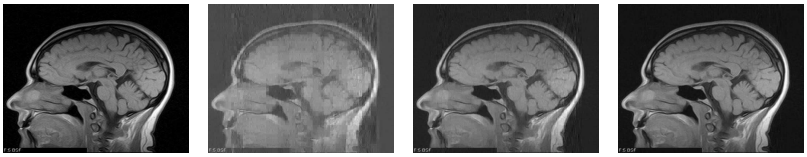
(a) 512 x 512 (b) 5.9e-02 (c) 1.0e-02 (d) 3.9e-03



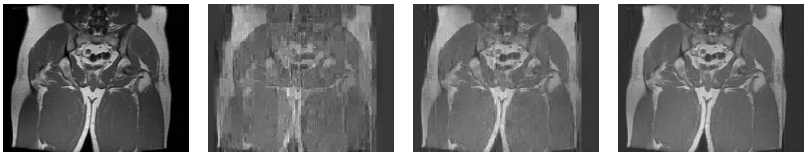
(e) 512 x 512 (f) 7.1e-02 (g) 2.0e-02 (h) 7.3e-03



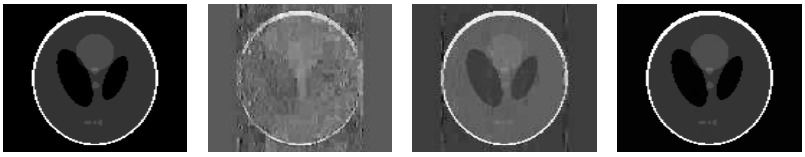
(i) 256 x 256 (j) 1.9e-01 (k) 9.0e-02 (l) 4.5e-02



(m) 512 x 512 (n) 7.1e-02 (o) 3.0e-02 (p) 7.8e-03



(q) 256 x 256 (r) 1.4e-01 (s) 7.5e-02 (t) 4.8e-02



(u) 128 x 128 (v) 3.6e-01 (w) 1.6e-01 (x) 4.1e-03

[8] ———, *Exposing constraints*, SIAM J. Optim., 4 (1994), pp. 573–595.

[9] R. H. BYRD, N. I. M. GOULD, J. NOCEDAL, AND R. A. WALTZ, *An algorithm for nonlinear optimization using linear programming and equality constrained subproblems*, Math. Program., 100 (2004), pp. 27–48.

[10] ———, *On the convergence of successive linear-quadratic programming algorithms*, SIAM J. Optim., 16 (2005), pp. 471–489 (electronic).

[11] E. CANDES AND S. BECKER, *Some test problems for compressed sensing.* private communication, 2008.

[12] E. CANDÈS AND J. ROMBERG, *Quantitative robust uncertainty principles and optimally sparse decompositions*, Foundations of Computational Mathematics, 6 (2006), pp. 227–254.

[13] E. CANDÈS, J. ROMBERG, AND T. TAO, *Robust uncertainty principles: Exact signal reconstruction from highly incomplete*

frequency information, IEEE Transactions on Information Theory, 52 (2006), pp. 489–509.

[14] E. Candès and T. Tao, *Near optimal signal recovery from random projections: universal encoding strategies*, IEEE Transactions on Information Theory, 52 (2004), pp. 5406–5425.

[15] P. L. Combettes and J.-C. Pesquet, *Proximal thresholding algorithm for minimization over orthonormal bases*, To appear in SIAM Journal on Optimization, (2007).

[16] W. Dai and M. Olgica, *Subspace pursuit for compressive sensing: Closing the gap between performance and complexity*, tech. rep., Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign, 2008.

[17] Y. H. Dai, *On the nonmonotone line search*, J. Optim. Theory Appl., 112 (2002), pp. 315–330.

[18] I. Daubechies, M. Defrise, and C. De Mol, *An iterative thresholding algorithm for linear inverse problems with a sparsity constraint*, Communications in Pure and Applied Mathematics, 57 (2004), pp. 1413–1457.

[19] C. De Mol and M. Defrise, *A note on wavelet-based inversion algorithms*, Contemporary Mathematics, 313 (2002), pp. 85–96.

[20] E. D. Dolan and J. J. Moré, *Benchmarking optimization software with performance profiles*, Math. Program., 91 (2002), pp. 201–213.

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

[22] D. Donoho, Y. Tsaig, I. Drori, and J.-C. Starck, *Sparse solution of underdetermined linear equations by stagewise orthogonal matching pursuit*, Submitted to IEEE Transactions on Information Theory, (2006).

[23] M. Elad, *Why simple shrinkage is still relevant for redundant representations?*, IEEE Transactions on Information Theory, 52 (2006), pp. 5559–5569.

[24] M. Elad, B. Matalon, J. Shtok, and M. Zibulevsky, *A wide-angle view at iterated shrinkage algorithms*, SPIE (Wavelet XII), San-Diego CA, August 26-29, 2007., (2007).

[25] M. Elad, B. Matalon, and M. Zibulevsky, *Coordinate and subspace optimization methods for linear least squares with non-quadratic regularization*, Journal on Applied and Computational Harmonic Analysis, (2006).

[26] 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.

[27] F. Facchinei, A. Fischer, and C. Kanzow, *On the accurate identification of active constraints*, SIAM J. Optim., 9 (1999), pp. 14–32 (electronic).

[28] M. Figueiredo and R. Nowak, *An EM algorithm for wavelet-based image restoration*, IEEE Transactions on Image Processing, 12 (2003), pp. 906–916.

[29] M. Figueiredo, R. Nowak, and S. J. Wright, *Gradient projection for sparse reconstruction: application to compressed sensing and other inverse problems*, To appear in the IEEE Journal of Selected Topics in Signal Processing: Special Issue on Convex Optimization Methods for Signal Processing, (2007).

[30] L. Grippo, F. Lampariello, and S. Lucidi, *A nonmonotone line search technique for Newton's method*, SIAM J. Numer. Anal., 23 (1986), pp. 707–716.

[31] W. W. Hager and H. Zhang, *A new active set algorithm for box constrained optimization*, SIAM J. Optim., 17 (2006), pp. 526–557 (electronic).

[32] E. Hale, W. Yin, and Y. Zhang, *A fixed-point continuation method for ℓ_1-regularization with application to compressed sensing*, Rice University CAAM Technical Report TR07-07, (2007).

[33] ———, *FPC: A fixed-point continuation method for ℓ_1-regularization,* `http://www.caam.rice.edu/~optimization/l1`, (2007).

[34] E. T. Hale, W. Yin, and Y. Zhang, *Fixed-point continuation for ℓ_1-minimization: methodology and convergence*, Submitted to SIAM Journal on Optimization, (2007).

[35] E. T. Hale, W. Yin, and Y. Zhang, *the barzilai-borwein method for l1-minimization.* private communication, 2007.

[36] Ilog, Inc., *Ilog cplex 10.0.* http://www.ilog.com, 2006.

[37] S.-J. Kim, K. Koh, M. Lustig, S. Boyd, and D. Gorinevsky, *A method for large-scale ℓ_1-regularized least squares problems with applications in signal processing and statistics,* `http://www.stanford.edu/~boyd/l1_ls.html`, (2007).

[38] S. Kirolos, J. Laska, M. Wakin, M. Duarte, D. Baron, T. Ragheb, Y. Massoud, and R. Baraniuk, *Analog-to-information conversion via random demodulation*, in Proceedings of the IEEE Dallas Circuits and Systems Workshop (DCAS), Dallas, Texas, 2006.

[39] J. Laska, S. Kirolos, M. Duarte, T. Ragheb, R. Baraniuk, and Y. Massoud, *Theory and implementation of an analog-to-information converter using random demodulation*, in Proceedings of the IEEE International Symposium on Circuites and Systems (ISCAS), New Orleans, Louisiana, 2007.

[40] J. Laska, S. Kirolos, Y. Massoud, R. Baraniuk, A. Gilbert, M. Iwen, and M. Strauss, *Random sampling for analog-to-information conversion of wideband signals*, in Proceedings of the IEEE Dallas Circuits and Systems Workshop,

Dallas, Texas, 2006.

[41] M. Lustig, D. Donoho, and J. Pauly, *Sparse MRI: The application of compressed sensing for rapid MR imaging*, Magnetic Resonance in Medicine, in press (2007).

[42] S. G. Mallat and Z. Zhang, *Matching pursuits with time-frequency dictionaries*, IEEE Transactions on Signal Processing, 41 (1993), pp. 3397–3415.

[43] J. J. Moré and G. Toraldo, *Algorithms for bound constrained quadratic programming problems*, Numer. Math., 55 (1989), pp. 377–400.

[44] ———, *On the solution of large quadratic programming problems with bound constraints*, SIAM J. Optim., 1 (1991), pp. 93–113.

[45] D. Needell and J. A. Tropp, *Cosamp: Iterative signal recovery from incomplete and inaccurate samples*, Applied and Computational Harmonic Analysis, (2008).

[46] Y. Nesterov, *Gradient methods for minimizing composite objective function*, `www.optimization-online.org`, CORE Discussion Paper 2007/76, (2007).

[47] J. Nocedal and S. J. Wright, *Numerical optimization*, Springer Series in Operations Research and Financial Engineering, Springer, New York, second ed., 2006.

[48] R. Nowak and M. Figueiredo, *Fast wavelet-based image deconvolution using the EM algorithm*, Proceedings of the 35th Asilomar Conference on Signals, Systems, and Computers, Monterey, CA, (2001).

[49] S. Osher, Y. Mao, B. Dong, and w. Yin, *Fast linearized bregman iteration for compressive sensing and sparse denoising.* To appear in Communications in Mathematical Sciences.

[50] L. Stewart, *Matlab lbfgs wrapper*, 2005. http://www.cs.toronto.edu/ liam/software.shtml.

[51] W. Sun and Y.-X. Yuan, *Optimization theory and methods*, vol. 1 of Springer Optimization and Its Applications, Springer, New York, 2006. Nonlinear programming.

[52] D. Takhar, J. Laska, M. Wakin, M. Duarte, D. Baron, S. Sarvotham, K. Kelly, and R. Baraniuk, *A new compressive imaging camera architecture using optical-domain compression*, in Proceedings of Computational Imaging IV at SPIE Electronic Image, San Jose, California, 2006.

[53] P. L. Toint, *An assessment of nonmonotone linesearch techniques for unconstrained optimization*, SIAM J. Sci. Comput., 17 (1996), pp. 725–739.

[54] J. Tropp, *Greed is good: Algorithmic results for sparse approximation*, IEEE Transactions on Information Theory, 50 (2006), pp. 2231–2342.

[55] ———, *Just relax: Convex programming methods for identifying sparse signals*, IEEE Transactions on Information Theory, 51 (2006), pp. 1030–1051.

[56] J. Tropp, M. Wakin, M. Duarte, D. Baron, and R. Baraniuk, *Random filters for compressive sampling and reconstruction*, in Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Toulouse, France, 2006.

[57] P. Tseng and S. Yun, *A coordinate gradient descent method for nonsmooth separable minimization*, tech. rep., Department of Mathematics, University of Washington, 2006.

[58] P. Tseng and S. Yun, *A block-coordinate gradient descent method for linearly constrained nonsmooth separable optimization*, Preprint, (2008).

[59] E. Van den Berg and M. P. Friedlander, *SPGL1: A MATLAB solver for large-scale sparse reconstruction,* `http://www.cs.ubc.ca/labs/scl/index.php/main/spgl1`, (2007).

[60] M. Wakin, J. Laska, M. Duarte, D. Baron, S. Sarvotham, D. Takhar, K. Kelly, and R. Baraniuk, *An architecture for compressive imaging*, in Proceedings of the International Conference on Image Processing (ICIP), Atlanta, Georgia, 2006.

[61] ———, *Compressive imaging for video representation and coding*, in Proceedings of Picture Coding Symposium (PCS), Beijing, China, 2006.

[62] Z. Wen, W. Yin, D. Goldfarb, and Y. Zhang, *On the convergence of an active set method for l_1 minimization*, tech. rep., Dept of IEOR, Columbia University, 2008.

[63] S. Wright, R. Nowak, and M. Figueiredo, *Sparse reconstruction by separable approximation*, Submitted, (2008).

[64] S. Wright, R. Nowak, and M. Figueiredo, *Sparse reconstruction by separable approximation*, Acoustics, Speech and Signal Processing, 2008. ICASSP 2008. IEEE International Conference on, (2008), pp. 3373–3376.

[65] J. Yang, Y. Zhang, and W. Yin, *A fast tvl1-l2 minimization algorithm for signal reconstruction from partial fourier data*, tech. rep. Rice CAAM Tech Report TR08-27.

[66] W. Yin, S. Osher, D. Goldfarb, and J. Darbon, *Bregman iterative algorithms for ℓ_1-minimization with applications to compressed sensing*, SIAM Journal on Imaging Sciences, 1 (2008), pp. 143–168.

[67] H. Zhang and W. W. Hager, *A nonmonotone line search technique and its application to unconstrained optimization*, SIAM J. Optim., 14 (2004), pp. 1043–1056 (electronic).

[68] C. Zhu, R. H. Byrd, P. Lu, and J. Nocedal, *Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization*, ACM Trans. Math. Software, 23 (1997), pp. 550–560.