# A FIRST-ORDER AUGMENTED LAGRANGIAN METHOD FOR COMPRESSED SENSING

N. S. AYBAT[*] AND G. IYENGAR[†]

**Abstract.** We propose a First-order Augmented Lagrangian algorithm (FAL) for solving the basis pursuit problem. FAL computes a solution to this problem by inexactly solving a sequence of $\ell_1$-regularized least squares sub-problems. These sub-problems are solved using an infinite memory proximal gradient algorithm wherein each update reduces to "shrinkage" or constrained "shrinkage". We show that FAL converges to an optimal solution of the basis pursuit problem whenever the solution is unique, which is the case with very high probability for compressed sensing problems. We construct a parameter sequence such that the corresponding FAL iterates are $\epsilon$-feasible and $\epsilon$-optimal for all $\epsilon > 0$ within $\mathcal{O}\left(\log\left(\epsilon^{-1}\right)\right)$ FAL iterations. Moreover, FAL requires at most $\mathcal{O}(\epsilon^{-1})$ matrix-vector multiplications of the form $Ax$ or $A^T y$ to compute an $\epsilon$-feasible, $\epsilon$-optimal solution. We show that FAL can be easily extended to solve the basis pursuit denoising problem when there is a non-trivial level of noise on the measurements. We report the results of numerical experiments comparing FAL with the state-of-the-art solvers for both noisy and noiseless compressed sensing problems. A striking property of FAL that we observed in the numerical experiments with randomly generated instances when there is no measurement noise was that FAL *always* correctly identifies the support of the target signal without any thresholding or post-processing, for moderately small error tolerance values.

**Key words.** $\ell_1$-minimization, augmented Lagrangian method, first order method, compressed sensing, basis pursuit, sparse optimization, denoising

**AMS subject classifications.** 90C25, 90C06, 49M29, 90C90, 65K05

## 1. Introduction.

In this paper we propose a new first-order augmented Lagrangian algorithm to solve the *basis pursuit* problem

$$\min_{x\in\mathbb{R}^n} \|x\|_1 \text{ subject to } Ax = b, \tag{1.1}$$

where $\ell_1$-norm $\|x\|_1 := \sum_{i=1}^n |x_i|$, $x_i$ denotes the $i$-th component of $x \in \mathbb{R}^n$, $b \in \mathbb{R}^m$, $A \in \mathbb{R}^{m\times n}$, with $m \ll n$, and $\text{rank}(A) = m$, i.e. $A$ has full row rank. The basis pursuit problem appears in the context of *compressed sensing* (CS) [6, 8, 9, 12] where the goal is to recover a sparse signal $x_*$ from a small set of linear measurements or transform values $b = Ax_*$. Candes, Romberg and Tao [6, 8, 9] and Donoho [12] have shown that when the target signal $x_*$ is $s$-sparse, i.e. only $s$ of the $n$ components are non-zero, and the measurement matrix $A \in \Re^{m\times n}$ satisfies some regularity conditions, the sparse signal $x_*$ can be recovered by solving the basis pursuit problem (1.1) with high probability provided that the number of measurements $m = \mathcal{O}(s\log(n))$. The basis pursuit problem is a linear program (LP). Therefore, computing the sparsest solution to the set of linear equations $Ax = b$, which is an NP-hard problem for general $A$, can be done efficiently, in theory, by solving an LP.

However, in typical CS applications the signal dimension $n$ is large, e.g. $n \approx 10^6$, and the LP (1.1) is often ill-conditioned. Consequently, general purpose simplex-based LP solvers are unable to solve the LP. Moreover, the constraint matrix $A$ is typically dense. Therefore, general purpose interior point methods that require factorization of $A^TA$ are not practical for solving LPs arising in CS applications.

On the other hand, in CS applications the $A$, although dense, still has a lot of structure. In many applications, $A$ is a partial transform matrix, e.g. partial discrete cosine transform (DCT), a partial wavelet, or a partial pseudo-polar Fourier matrix. Therefore, the matrix-vector product $Ax$ and $A^Ty$ can be computed in $\mathcal{O}(n\log(n))$ time using either the Fast Fourier Transform (FFT) or forward and backward Wavelet transforms. This fact has been recently exploited by a number of *first-order* algorithms. In this paper, we propose a new first-order augmented Lagrangian algorithm for the basis pursuit problem. Since the basic steps in a first-order algorithm are the matrix-vector multiplications in the form of $Ax$ and $A^Ty$, we will report complexity in terms of the number of such matrix-vector multiplications required to solve the problem.

[*]Industrial Engineering Department, The Pennsylvania State University, University Park, PA 16802. Email: nsa10@psu.edu
[†]IEOR Department, Columbia University, New York, NY 10027. Email: gi10@columbia.edu

**1.1. Previous work on first-order algorithms for compressed sensing.** When the measurement data $b$ contains a non-trivial level of noise, one can solve

$$\min_{x\in\mathbb{R}^n} \bar{\lambda}\|x\|_1 + \|Ax-b\|_2^2, \tag{1.2}$$

for an appropriately chosen $\bar{\lambda} > 0$ depending on the noise level to recover the sparse target signal $x_*$ with some error proportional to the noise on $b$ [7]. On the other hand, when there is no noise on the measurements, $b$, or when the noise level is low, one can solve (1.2) for a fixed small $\bar{\lambda} > 0$, which can be viewed as a penalty approximation to (1.1).

In [15] Figueiredo, Nowak and Wright proposed the GPSR algorithm that uses gradient projection method with Barzilai-Borwein steps to solve (1.2). Hale, Yin and Zhang [16, 17] proposed to solve (1.2) via the fixed point continuation (FPC) algorithm that embeds the soft-thresholding (IST) algorithm [10] in a continuation scheme on $\lambda$, i.e. FPC begins with $\lambda > \bar{\lambda}$ and gradually decreases it to $\bar{\lambda}$, to recover the sparse solution of (1.2). Wen, Yin, Goldfarb and Zhang [23] improved the performance of FPC by adding an active set (AS) step. Please note that GPSR, FPC and FPC-AS only converge to the optimal solution of (1.2), *not* to the optimal solution of (1.1). Hence, when there is no noise or when it is low, the solutions produced by these algorithms are only good approximations to $x_*$.

Yin, Osher, Goldfarb and Darbon [25] solve (1.1) using a Bregman iterative regularization scheme that involves a sequence of problems of the form $\min_{x\in\mathbb{R}^n} \bar{\lambda}\|x\|_1 + \frac{1}{2}\|Ax - b^{(k)}\|_2^2$, where $b^{(k)}$ are obtained by suitably updating the measurement vector $b$, and each sub-problem is solved using FPC. For the basis pursuit problem, the so-called Bregman iterative regularization procedure is nothing but the classic augmented Lagrangian method. The algorithm YALL1 developed by Yang and Zhang [24], which is an alternating direction algorithm, is able to solve the basis pursuit problem (1.1), the penalty formulation (1.2), and the basis pursuit denoising problem

$$\begin{array}{ll} \min & \|x\|_1, \\ \text{s.t.} & \|Ax-b\|_2 \le \delta. \end{array} \tag{1.3}$$

Bregman iteration based methods [25] and YALL1 [24] provably converge to the optimal solution of the basis pursuit problem (1.1); however, their convergence rates are unknown.

Other algorithms for $\ell_1$-regularized least squares problem (1.2) include an iterative interior-point solver [18], and an accelerated projected gradient method [11]. Van den Berg and Friedlander [22] proposed SPGL1 to solve the penalty formulation (1.3) by solving a sequence of LASSO sub-problems $\Psi(t) = \{\|Ax-b\|_2^2 : \|x\|_1 \le t\}$ where parameter $t$ is updated by a Newton step. This algorithm provably converges to the optimal solution of (1.3); however, the convergence rate is again unknown.

Aybat and Iyengar [2] have proposed a first-order Smoothed Penalty Algorithm (SPA) to solve the basis pursuit problem. SPA iterates $\{x^{(k)}\}_{k\in\mathbb{Z}_+}$ are computed by inexactly solving a sequence of smoothed penalty problems of the form

$$\min_{\|x\|_2\le\eta^{(k)}} \left\{\lambda^{(k)} p_\mu^{(k)}(x) + f_\nu^{(k)}(x)\right\},$$

where $p_\mu^{(k)}(x)$ is a smooth approximation of $\|x\|_1$, $f_\nu^{(k)}(x)$ is a smooth approximation of $\|Ax-b\|_2$ and $\eta^{(k)}$ is a suitably chosen bound on the $\ell_2$-norm of an optimal solution of the $k$-th sub-problem. SPA calls Nesterov's optimal algorithm for simple sets [19, 20] to solve the sub-problems. SPA iterates provably converge to an optimal solution $x_*$ of the basis pursuit problem whenever it is unique. Moreover, for all small enough $\epsilon > 0$, SPA requires $\mathcal{O}(\sqrt{n}\epsilon^{-\frac{3}{2}})$ matrix-vector multiplies to compute an $\epsilon$-feasible, i.e. $\|Ax^{(k)} - b\|_2 \le \epsilon$, and $\epsilon$-optimal, $\big|\ \|x^{(k)}\|_1 - \|x_*\|_1\big| \le \epsilon$ iterate.

Becker, Bobin and Candès [4] have proposed NESTA for solving the formulation (1.3) (NESTA can also be used to solve the basis pursuit problem (1.1) by setting $\delta$ to 0). NESTA calls Nesterov's optimal gradient method for non-smooth convex functions [20] to solve the sub-problems. When the matrix $A$ is orthogonal, i.e. $AA^T = I$, NESTA requires $\mathcal{O}(\sqrt{n}\epsilon^{-1})$ matrix-vector multiplications to compute a feasible $\epsilon$-optimal iterate to (1.3). When the matrix $A$ is a partial transform matrix, i.e. $Ax$ and $A^T y$ is $\mathcal{O}(n\log(n))$, but $A$ is not orthogonal, NESTA, in general, needs to compute $(A^TA+\mu I)^{-1}$, and therefore, its $\mathcal{O}(n^3)$ per iteration complexity is quite prohibitive for practical applications. Moreover, the sequence of NESTA iterates does not converge an optimal solution of (1.3) but to a solution of a smooth approximation of (1.3).

**1.2. New results.** In this paper we propose a first-order augmented Lagrangian (FAL) algorithm that solves the basis pursuit problem by inexactly solving a sequence of optimization problems of the form

$$\min_{x\in\mathbb{R}^n:\ \|x\|_1\leq\eta^{(k)}} \Big\{\lambda^{(k)}\|x\|_1 - \lambda^{(k)}(\theta^{(k)})^T(Ax-b) + \frac{1}{2}\|Ax-b\|_2^2\Big\}, \tag{1.4}$$

for an appropriately chosen sequence $\{(\lambda^{(k)},\theta^{(k)},\eta^{(k)})\}_{k\in\mathbb{Z}_+}$. Each of these sub-problems are solved using a variant (see Figure 2.1) of the infinite-memory proximal gradient algorithm in [21] (see, also FISTA [3] and Nesterov infinite-memory algorithm [20]). Each update in this proximal gradient algorithm involves computing the gradient $A^T(Ax-b)$ of the quadratic term $\frac{1}{2}\|Ax-b\|_2^2$ and computing two constrained "shrinkage" (see Equation A.10), which require $\mathcal{O}(n\log(n))$ work. Hence, the complexity of each update is dominated by computing the gradient $A^T(Ax-b)$ or equivalently two matrix-vector multiplies.

In Theorem 3.1 in Section 3 we prove that every limit point of the FAL iterate sequence is an optimal solution of (1.1). Thus, the FAL iterates converge to the optimal solution when the solution is unique. In Theorem 3.4 we show that for all $\epsilon > 0$, the FAL iterates $x^{(k)}$ are $\epsilon$-feasible, i.e. $\|Ax^{(k)}-b\|_2\leq\epsilon$, and $\epsilon$-optimal, $\big|\,\|x^{(k)}\|_1-\|x_*\|_1\big|\leq\epsilon$, for $k\geq\mathcal{O}(\log(\epsilon^{-1}))$. Moreover, FAL requires at most $\mathcal{O}(n\epsilon^{-1})$ matrix-vector multiplications to compute an $\epsilon$-feasible, $\epsilon$-optimal solution to (1.1). Thus, the overall complexity of FAL computing an $\epsilon$-feasible and $\epsilon$-optimal iterate is $\mathcal{O}(n^2\log(n)\epsilon^{-1})$ in the CS context. And in Section 4, we briefly discuss how to extend FAL to solve the noisy recovery problem $\min_{x\in\mathbb{R}^n}\{\|x\|_1:\ \|Ax-b\|_2\leq\delta\}$.

In Section 6 we report the results of our experiments with FAL. We tested FAL on randomly generated problems both with and without measurement noise and also on known hard instances of the CS problems. We compared the performance of FAL with SPA [2], NESTA [4], FPC [17], FPC-AS [23], YALL1 [24] and SPGL1 [22]. On randomly generated problem instances FAL is at least two times faster than all the other solvers. On known hard CS instances the run times of FAL were of the same order of magnitude as the best solver; but FAL was able to identify significantly sparser solutions. We also observed that for all randomly generated instances with no measurement noise FAL *always* correctly identified the support of the target signal $x_*$, without any additional heuristic thresholding, when the error tolerance was set to moderate values. Once the support is known, the signal $x_*$ can often be very accurately computed by solving a set of linear equations. Moreover, although the bound in Theorem 3.4 implies that FAL requires $\mathcal{O}\left(\frac{1}{\epsilon}\right)$ matrix-vector multiplies to compute an $\epsilon$-feasible, $\epsilon$-optimal solution, in practice we observed that FAL required only $\mathcal{O}\left(\log\left(\frac{1}{\epsilon}\right)\right)$ matrix-vector multiplies to compute an $\epsilon$-feasible, $\epsilon$-optimal solution.

FAL is superior to SPA [2] both in terms of the theoretical guarantees as well as practical performance on the basis pursuit problem. However, FAL explicitly uses the structure of the $\ell_1$-norm and is, therefore, restricted to basis pursuit and related problems. On the other hand, SPA can be extended easily to solve the following much larger class non-smooth convex optimization problems:

$$\begin{array}{rl} \min & \max_{u\in U}\{\phi(x,u)\}, \\ \text{subject to} & \|Ax-b\|_\gamma\leq\delta, \end{array}$$

where $U$ is a compact convex set and $\phi:\mathbb{R}^n\times U\to\mathbb{R}$ is a bi-affine function [20], and $\gamma\in\{1,2,\infty\}$. This class includes as special cases, basis pursuit, matrix games with side constraints, group LASSO, and problems of the form $\min\{\sum_{k=1}^p\|B_kx\|_1 : Ax=b\}$ that appears in the context of reconstructing a piecewise flat sparse image.

## 2. Preliminaries.

In this section we state and briefly discuss the details of a particular variant of Tseng's Algorithm 3 in [21] that we use in FAL. Algorithm 3 [21] computes $\epsilon$-optimal solutions for the optimization problem

$$\min_{x\in F} p(x)+f(x), \tag{2.1}$$

where $f$, $p$ and $F$ satisfy the following conditions.

$$\begin{array}{rl} \text{(i)} & p:\mathbb{R}^n \to \mathbb{R} \text{ proper, lower-semicontinuous (lsc) and convex function, and } \mathbf{dom}\, p \text{ closed,} \\ \text{(ii)} & f:\mathbb{R}^n \to \mathbb{R} \text{ proper, lsc, convex function, differentiable on an open set containing } \mathbf{dom}\, p, \\ \text{(iii)} & \nabla f \text{ is Lipschitz continuous on } \mathbf{dom}\, p \text{ with constant } L, \\ \text{(iv)} & F \cap \operatorname{argmin}_{x\in\mathbb{R}^n}\{p(x)+f(x)\} \neq \emptyset. \end{array} \tag{2.2}$$

We refer to a function $h:\mathbb{R}^n \to \mathbb{R}$ as a *prox* function if $h$ is differentiable and strongly convex function with convexity parameter $c>0$, i.e. $h(y) \geq h(x) + \nabla h(x)^T(y-x) + \frac{c}{2}\|y-x\|_2^2$ for all $x, y \in \mathbf{dom}\, h$.

Algorithm APG($p, f, L, F, x^{(0)}, h$, APGstop)

```
1: $u^{(0)} \leftarrow x^{(0)}$, $w^{(0)} \leftarrow \operatorname{argmin}_{x\in\mathbf{dom}\,p} h(x)$, $\vartheta^{(0)} \leftarrow 1$, $\ell \leftarrow 0$
2: while (APGstop is false) do
3:     $v^{(\ell)} \leftarrow (1-\vartheta^{(\ell)})u^{(\ell)} + \vartheta^{(\ell)} w^{(\ell)}$
4:     $w^{(\ell+1)} \leftarrow \operatorname{argmin}\left\{\sum_{i=0}^{\ell} \frac{1}{\vartheta^{(i)}}\left(p(z) + \nabla f(v^{(i)})^T z\right) + \frac{L}{c}\ h(z) : z \in F\right\}$
5:     $\hat{u}^{(\ell+1)} \leftarrow (1-\vartheta^{(\ell)})u^{(\ell)} + \vartheta^{(\ell)} w^{(\ell+1)}$
6:     $H^{(\ell)}(x) := p(x) + \nabla f(v^{(\ell)})^T x + \frac{L}{2}\|x - v^{(\ell)}\|_2^2$
7:     $u^{(\ell+1)} \leftarrow \operatorname{argmin}\{H^{(\ell)}(x) :\ x \in F\}$
8:     $\vartheta^{(\ell+1)} \leftarrow \frac{\sqrt{(\vartheta^{(\ell)})^4 - 4(\vartheta^{(\ell)})^2} - (\vartheta^{(\ell)})^2}{2}$
9:     $\ell \leftarrow \ell + 1$
10: end while
11: return $u^{(\ell)}$ or $v^{(\ell)}$ depending on APGstop
```

Fig. 2.1. *Accelerated Proximal Gradient Algorithm*

Our variant of Algorithm 3 in [21] is displayed in Figure 2.1. Algorithm APG takes as input the functions $f$ and $p$, a prox function $h$, the set $F$, an initial iterate $x^{(0)}$ and a stopping criterion APGstop.

Lemma 2.1. *Suppose $p$, $f$ and $F$ satisfy (2.2). Let $h$ be a prox function on an open set containing $\mathbf{dom}\, p$ and $\min_{x\in\mathbf{dom}\,p} h(x) = 0$. Fix $\epsilon > 0$ and let $\{u^{(\ell)}, v^{(\ell)}, w^{(\ell)}\}_{\ell\in\mathbb{Z}}$ be the sequence generated by* Algorithm APG *displayed in Figure 2.1. Then $p(u^{(\ell+1)}) + f(u^{(\ell+1)}) \leq \min_{x\in\mathbb{R}^n}\{p(x)+f(x)\} + \epsilon$ for all $\ell \geq \sqrt{\frac{4L}{c\epsilon}\ h(x_*)} - 1$, where $x_* \in \operatorname{argmin}_{x\in\mathbb{R}^n}\{p(x)+f(x)\}$.*

*Proof.* Corollary 3 in [21] implies this result provided that $H(u^{(\ell+1)}) \leq H(\hat{u}^{(\ell+1)})$ holds for all $\ell \geq 0$. Using induction, it is easy to show that this is true when the update rule (7) is used for all $\ell \geq 1$. $\square$

## 3. Convergence Properties of FAL.

In this section, we describe FAL and prove the main convergence results for the algorithm. The outline of FAL is given in Figure 3.1. Algorithm FAL takes as inputs a sequence of $\{(\lambda^{(k)}, \epsilon^{(k)}, \tau^{(k)})\}_{k\in\mathbb{Z}_+}$, a starting point $x^{(0)}$ and a bound $\eta$ on the $\ell_1$-norm of an optimal solution $x_*$ of the basis pursuit problem. One such bound $\eta$ can be computed as follows. Let $\tilde{x} = \operatorname{argmin}\{\|x\|_2 : Ax = b\} = A^T(AA^T)^{-1}b$. Clearly, $\|x_*\|_1 \leq \eta := \|\tilde{x}\|_1$. We will next describe each of the steps in this outline.

An augmented Lagrangian function for the basis pursuit problem (1.1) can be written as

$$P(x) := \lambda\|x\|_1 - \lambda\theta^T(Ax-b) + \frac{1}{2}\|Ax-b\|_2^2,$$

where $\lambda$ is the penalty parameter and $\theta$ is a dual variable for the constraints $Ax = b$. From Lines 4-7 in Figure 3.1 it follows that in the $k$-th iteration of Algorithm FAL we inexactly minimize the augmented Lagrangian function

$$P^{(k)}(x) := \lambda^{(k)}\|x\|_1 + \frac{1}{2}\|Ax - b - \lambda^{(k)}\theta^{(k)}\|_2^2 = \lambda^{(k)}\|x\|_1 - \lambda^{(k)}(\theta^{(k)})^T(Ax-b) + \frac{1}{2}\|Ax-b\|_2^2 - \frac{1}{2}\|\lambda^{(k)}\theta^{(k)}\|_2^2,$$

over the set $F^{(k)} := \{x : \|x\|_1 \leq \eta^{(k)}\}$ using Algorithm APG with the prox function $h^{(k)} = \frac{1}{2}\|x - x^{(k-1)}\|_2^2$.

```
ALGORITHM FAL$\big(\{(\lambda^{(k)},\epsilon^{(k)},\tau^{(k)})\}_{k\in\mathbb{Z}_+},x^{(0)},\eta\big)$
1: $\theta^{(1)}=0,\ k\leftarrow 0,\ L\leftarrow\sigma_{\max}(AA^T)$
2: while (FALSTOP is false) do
3:     $k\leftarrow k+1$
4:     $p^{(k)}(x):=\lambda^{(k)}\|x\|_1,\quad f^{(k)}(x):=\frac{1}{2}\|Ax-b-\lambda^{(k)}\theta^{(k)}\|_2^2$
5:     $h^{(k)}(x):=\frac{1}{2}\|x-x^{(k-1)}\|_2^2$
6:     $\eta^{(k)}\leftarrow\eta+\frac{\lambda^{(k)}}{2}\|\theta^{(k)}\|_2^2$
7:     $F^{(k)}:=\{x\in\mathbb{R}^n:\ \|x\|_1\le\eta^{(k)}\}$
8:     $\ell_{\max}^{(k)}\leftarrow\sigma_{\max}(A)(\eta^{(k)}+\|x^{(k-1)}\|_2)\sqrt{\frac{2}{\epsilon^{(k)}}}$
9:     APGSTOP $:=\{\ell\ge\ell_{\max}^{(k)}\}$ or $\big\{\exists g\in\partial P^{(k)}(x)|_{v^{(\ell)}}$ with $\|g\|_2\le\tau^{(k)}\big\}$
10:    $x^{(k)}\leftarrow$ APG$\Big(p^{(k)},f^{(k)},L,F^{(k)},x^{(k-1)},h^{(k)},$ APGSTOP$\Big)$
11:    $\theta^{(k+1)}\leftarrow\theta^{(k)}-\frac{Ax^{(k)}-b}{\lambda^{(k)}}$
12: end while
13: return $x_{\rm sol}\leftarrow x^{(k)}$
```

FIG. 3.1. *Outline of First-Order Augmented Lagrangian Algorithm (FAL)*

Recall that when using ALGORITHM APG we need to ensure that $F^{(k)}\cap\operatorname{argmin}_{x\in\mathbb{R}^n}\{P^{(k)}(x)\}\neq\emptyset$. Let $x_*^{(k)}\in\operatorname{argmin}_{x\in\mathbb{R}^n}P^{(k)}(x)$. Since $P^{(k)}(x_*^{(k)})\le P^{(k)}(x_*)$, $Ax_*=b$ and $\|x_*\|_1\le\eta$, we have $\|x_*^{(k)}\|_1\le\eta^{(k)}$. Thus, $x_*^{(k)}\in F^{(k)}$.

Next, we discuss the stopping criterion set in Line 9 in Figure 3.1. The convexity parameter of the prox-function $h^{(k)}$ is 1. Hence, Lemma 2.1 establishes that

$$P^{(k)}(u^{(\ell)})\le P^{(k)}(x_*^{(k)})+\epsilon^{(k)}\quad\text{for}\quad\ell\ge\sqrt{\frac{2L}{\epsilon^{(k)}}}\ \|x_*^{(k)}-x^{(k-1)}\|_2,$$

where $\{u^{(\ell)}\}_{\ell\in\mathbb{Z}_+}$ is the sequence of u-iterates when ALGORITHM APG is applied to the $k$-th subproblem and $L$ denotes the Lipschitz constant of the gradient $\nabla f^{(k)}(x)$. Since $\|x_*^{(k)}\|_1\le\eta^{(k)}$ and $\|.\|_1\ge\|.\|_2$, triangle inequality implies that

$$\|x_*^{(k)}-x^{(k-1)}\|_2\le\|x_*^{(k)}\|_1+\|x^{(k-1)}\|_2\le\eta^{(k)}+\|x^{(k-1)}\|_2. \tag{3.1}$$

Since $\nabla f^{(k)}(x)=A^T(Ax-b-\lambda^{(k)}\theta^{(k)})$ it follows that $L=\sigma_{\max}^2(A)$, where $\sigma_{\max}(A)$ denote the largest singular value of $A$. Thus, it follows that

$$\sqrt{\frac{2L}{\epsilon^{(k)}}}\ \|x_*^{(k)}-x^{(k-1)}\|_2\le\sigma_{\max}(A)(\eta^{(k)}+\|x^{k-1}\|_2)\sqrt{\frac{2}{\epsilon^{(k)}}}=:\ell_{\max}^{(k)}.$$

Consequently, it follows that the stopping criterion APGSTOP in Line 9 ensures that the iterate $x^{(k)}$ satisfies one of the following two conditions

$$\begin{array}{ll}\text{(a)} & P^{(k)}(x^{(k)})\le\min_{x\in\mathbb{R}^n}P^{(k)}(x)+\epsilon^{(k)},\\ \text{(b)} & \exists g\in\partial P^{(k)}(x)|_{x^{(k)}}\text{ with }\|g\|_2\le\tau^{(k)},\end{array} \tag{3.2}$$

where $\partial P^{(k)}(x)|_{x^{(k)}}$ denotes the set of subgradients of the function $P^{(k)}$ at $x^{(k)}$. When $\ell\ge\ell_{\max}^{(k)}$ in APGSTOP holds, ALGORITHM APG returns $u^{(\ell)}$; otherwise, when $\exists g\in\partial P^{(k)}(x)|_{v^{(\ell)}}$ with $\|g\|_2\le\tau^{(k)}$, ALGORITHM APG returns $v^{(\ell)}$. And we set $x^{(k)}$ to what ALGORITHM APG returns.

In Line 11, we update the dual variables $\theta^{(k)}$ in a manner that is standard for augmented Lagrangian algorithms. This completes the description of ALGORITHM FAL displayed in Figure 3.1.

In the result below we establish that every limit point of the FAL iterate sequence $\{x^{(k)}\}_{k\in\mathbb{Z}}$, is an optimal solution of the basis pursuit problem.

THEOREM 3.1. *Fix $x^{(0)} \in \mathbb{R}^n$, $\eta > 0$ such that $\eta \geq \|x_*\|_1$ and a sequence of parameters $\{(\lambda^{(k)}, \epsilon^{(k)}, \tau^{(k)})\}_{k\in\mathbb{Z}_+}$ such that*

*(i) penalty parameters, $\lambda^{(k)} \searrow 0$,*
*(ii) approximate optimality parameters, $\epsilon^{(k)} \searrow 0$ such that $\frac{\epsilon^{(k)}}{(\lambda^{(k)})^2} \leq B_1$ for all $k \geq 1$,*
*(iii) subgradient tolerance parameters, $\tau^{(k)} \searrow 0$ such that $\frac{\tau^{(k)}}{\lambda^{(k)}} \leq B_2$ for all $k \geq 1$, and $\frac{\tau^{(k)}}{\lambda^{(k)}} \to 0$ as $k \to 0$.*

*Let $\mathcal{X} = \{x^{(k)}\}_{k\in\mathbb{Z}_+}$ denote the iterates computed by* ALGORITHM FAL *for this set of parameters. Then, $\mathcal{X}$ is a bounded sequence and any limit point $\bar{x}$ of $\mathcal{X}$ is an optimal solution of the basis pursuit problem (1.1).*

*Proof.* Since $\ell^{(k)}_{\max}$ is finite for all $k \geq 1$, it follows that the sequence $\mathcal{X}$ exists.

As a first step towards establishing that $\mathcal{X}$ is bounded, we establish a uniform bound on the sequence of dual multipliers $\{\theta^{(k)}\}_{k\in\mathbb{Z}_+}$. Suppose in the $k$-th FAL iteration ALGORITHM APG terminates with the iterate $x^{(k)}$ satisfying (3.2)(a). Then Corollary A.2 applied to $P^{(k)}(x) = \lambda^{(k)}\|x\|_1 + \|Ax - b - \lambda^{(k)}\theta^{(k)}\|_2^2$ guarantees that

$$\|A^T(Ax^{(k)} - b - \lambda^{(k)}\theta^{(k)})\|_\infty \leq \sqrt{2\epsilon^{(k)}}\ \sigma_{\max}(A) + \lambda^{(k)}.$$

Instead, if the iterate $x^{(k)}$ satisfies (3.2)(b), i.e. there exists $q^{(k)} \in \partial\|x\|_1|_{x^{(k)}}$ such that $\|\lambda^{(k)}q^{(k)} + A^T(Ax^{(k)} - b - \lambda^{(k)}\theta^{(k)})\|_2 \leq \tau^{(k)}$, then

$$\|A^T(Ax^{(k)} - b - \lambda^{(k)}\theta^{(k)})\|_\infty \leq \tau^{(k)} + \lambda^{(k)}\|q^{(k)}\|_\infty \leq \tau^{(k)} + \lambda^{(k)}, \tag{3.3}$$

where the second inequality follows from the fact that $\|q^{(k)}\|_\infty \leq 1$ for all $q^{(k)} \in \partial\|x\|_1|_{x^{(k)}}$.

Since $\theta^{(1)} = 0$, $\theta^{(k+1)} = \theta^{(k)} - \frac{Ax^{(k)}-b}{\lambda^{(k)}}$ for all $k \geq 1$ and $A$ has full row-rank, it follows that

$$\begin{aligned}\|\theta^{(k+1)}\|_2 &\leq \frac{1}{\lambda^{(k)}\sigma_{\min}(A)}\|A^T(Ax^{(k)} - b - \lambda^{(k)}\theta^{(k)})\|_2, \\ &\leq \frac{\sqrt{n}}{\sigma_{\min}(A)}\left(\max\left\{\sigma_{\max}(A)\sqrt{\frac{2\epsilon^{(k)}}{(\lambda^{(k)})^2}},\ \frac{\tau^{(k)}}{\lambda^{(k)}}\right\} + 1\right), \quad \forall k \geq 1. \end{aligned}\tag{3.4}$$

The bounds $\frac{\epsilon^{(k)}}{(\lambda^{(k)})^2} \leq B_1$, and $\frac{\tau^{(k)}}{\lambda^{(k)}} \leq B_2$, together with (3.4) imply that

$$\|\theta^{(k)}\|_2 \leq B_\theta := \frac{\sqrt{n}}{\sigma_{\min}(A)}\left(\max\{\sqrt{2B_1}\ \sigma_{\max}(A),\ B_2\} + 1\right), \quad \forall k > 1. \tag{3.5}$$

From this bound, it follows that

$$\|x^{(k)}\|_1 \leq \eta^{(k)} = \eta + \frac{\lambda^{(k)}}{2}\|\theta^k\|_2^2 \leq B_x := \eta + \frac{1}{2}\lambda^{(1)}B_\theta^2. \tag{3.6}$$

Thus, $\mathcal{X}$ is a bounded sequence and it has a limit point. Let $\bar{x}$ denote any limit point and let $\mathcal{K} \subset \mathbb{Z}_+$ denote a subsequence such that $\lim_{k\in\mathcal{K}} x^{(k)} = \bar{x}$.

Suppose that there exists a further sub-sequence $\mathcal{K}_a \subset \mathcal{K}$ such that for all $k \in \mathcal{K}_a$ calls to ALGORITHM APG terminates with an iterate $x^{(k)}$ satisfying (3.2)(a). Then, for $k \in \mathcal{K}_a$, we have that

$$\begin{aligned}\|x^{(k)}\|_1 &\leq \frac{P^{(k)}(x^{(k)})}{\lambda^{(k)}} \leq \frac{P^{(k)}(x^{(k)}_*) + \epsilon^{(k)}}{\lambda^{(k)}} \leq \frac{P^{(k)}(x_*) + \epsilon^{(k)}}{\lambda^{(k)}} \\ &= \|x_*\|_1 + \frac{\lambda^{(k)}}{2}\|\theta^{(k)}\|_2^2 + \frac{\epsilon^{(k)}}{\lambda^{(k)}} \leq \|x_*\|_1 + \frac{1}{2}\lambda^{(k)}B_\theta^2 + \lambda^{(k)}B_1, \end{aligned}\tag{3.7}$$

where the first inequality follows from the fact $f^{(k)}(x) \geq 0$, second follows from the stopping condition, the third follows from the fact that $P^{(k)}(x_*^{(k)}) \leq P^{(k)}(x_*)$, the equality follows from the fact that $Ax_* = b$, and the last inequality follows from the bounds $\|\theta^{(k)}\|_2 \leq B_\theta$ and $\frac{\epsilon^{(k)}}{(\lambda^{(k)})^2} \leq B_1$. Since $\lambda^{(k)} \to 0$, taking the limit along $\mathcal{K}_a$ we get

$$\|\bar{x}\|_1 = \lim_{k\in\mathcal{K}_a} \|x^{(k)}\|_1 \leq \|x_*\|_1 + \lim_{k\in\mathcal{K}_a} \left\{\lambda^{(k)}\left(\frac{1}{2}B_\theta^2 + B_1\right)\right\} = \|x_*\|_1. \tag{3.8}$$

Next, consider feasibility of the limit point $\bar{x}$.

$$\begin{aligned}\|Ax^{(k)} - b - \lambda^{(k)}\theta^{(k)}\|_2^2 &\leq P^{(k)}(x^{(k)}) \leq P^{(k)}(x_*^{(k)}) + \epsilon^{(k)} \leq P^{(k)}(x_*) + \epsilon^{(k)},\\ &\leq \lambda^{(k)}\|x_*\|_1 + \frac{1}{2}\|\lambda^{(k)}\theta^{(k)}\|_2^2 + \epsilon^{(k)} \leq \lambda^{(k)}\|x_*\|_1 + \frac{1}{2}(\lambda^{(k)}B_\theta)^2 + \epsilon^{(k)},\end{aligned}$$

where the first inequality follows from the fact $\lambda^{(k)}\|x^{(k)}\| \geq 0$, the third follows from the fact that $P^{(k)}(x_*^{(k)}) \leq P^{(k)}(x_*)$, the fourth follows from the fact $Ax_* = b$, and the last follows from the bound $\|\theta^{(k)}\|_2 \leq B_\theta$. Taking the limit along $\mathcal{K}_a$, we have

$$\frac{1}{2}\|A\bar{x} - b\|_2^2 \leq 0,$$

i.e. $A\bar{x} = b$. Since $\bar{x}$ is feasible, and $\|\bar{x}\|_1 \leq \|x_*\|_1$, it follows that $\bar{x}$ is an optimal solution for the basis pursuit problem (1.1).

Now, consider the complement case, i.e. there exists $K \in \mathcal{K}$ such that for all $k \in \mathcal{K}_b := \mathcal{K} \cap \{k \geq K\}$, calls to Algorithm APG terminate with an iterate $x^{(k)}$ that satisfies (3.2)(b). For all $k \in \mathcal{K}_b$, there exists $q^{(k)} \in \partial\|x\|_1|_{x^{(k)}}$ such that

$$\|\lambda^{(k)}q^{(k)} + A^T(Ax^{(k)} - b - \lambda^{(k)}\theta^{(k)})\|_2 \leq \tau^{(k)}. \tag{3.9}$$

Then, for all $k \in \mathcal{K}_b$,

$$\begin{aligned}\|Ax^{(k)} - b\|_2 &\leq \frac{1}{\sigma_{\min}(A)}\|A^T(Ax^{(k)} - b)\|_2,\\ &\leq \frac{1}{\sigma_{\min}(A)}\Big(\|A^T(Ax^{(k)} - b - \lambda^{(k)}\theta^{(k)}) + \lambda^{(k)}q^{(k)}\|_2 + \|\lambda^{(k)}q^{(k)}\|_2 + \|A^T(\lambda^{(k)}\theta^{(k)})\|_2\Big),\\ &\leq \frac{1}{\sigma_{\min}(A)}\Big(\tau^{(k)} + \lambda^{(k)}\sqrt{n} + \sigma_{\max}(A)\lambda^{(k)}B_\theta\Big),\end{aligned}$$

where $\sigma_{\min}(A)$ denotes the smallest non-zero singular value of $A$. The first inequality follows from the definition of $\sigma_{\min}(A)$, the second inequality follows from triangle inequality, and last inequality follows from (3.9), the bound $\|\theta^{(k)}\|_2 \leq B_\theta$, and fact that $\|q\|_2 \leq \sqrt{n}$ for any $q \in \partial\|x\|_1|_{x^{(k)}}$. Taking the limit along $\mathcal{K}_b$, we have $\|A\bar{x} - b\|_2 \leq 0$, or equivalently $A\bar{x} = b$.

For all $k \in \mathcal{K}_b$, $q^{(k)} \in \partial\|x\|_1|_{x^{(k)}}$, therefore, $\|q^{(k)}\|_\infty \leq 1$. Hence, there exists a subsequence $\mathcal{K}_b' \subset \mathcal{K}_b$ such that $\lim_{k\in\mathcal{K}_b'} q^{(k)} = \bar{q}$ exists. One can easily show that $\bar{q} \in \partial\|x\|_1|_{\bar{x}}$. Dividing both sides of (3.9) by $\lambda^{(k)}$, we get

$$\|q^{(k)} - A^T\theta^{(k+1)}\|_2 \leq \frac{\tau^{(k)}}{\lambda^{(k)}}, \tag{3.10}$$

for all $k \in \mathcal{K}_b'$. Since $\lim_{k\in\mathcal{K}_b'} q^{(k)} = \bar{q}$, $\lim_{k\in\mathbb{Z}_+} \frac{\tau^{(k)}}{\lambda^{(k)}} = 0$ and $A$ has full row rank, it follows that $\{\theta^{(k)} : k \in \mathcal{K}_b'\}$ is a Cauchy sequence; therefore, $\lim_{k\in\mathcal{K}_b'} \theta^{(k+1)} = \bar{\theta}$ exists. Taking the limit of both sides of (3.10) along $\mathcal{K}_b'$, we have

$$\bar{q} = A^T\bar{\theta}. \tag{3.11}$$

(3.11) together with that the fact that $A\bar{x} = b$ and $q \in \partial\|\bar{x}\|_1$, it follows that the KKT conditions for optimality is satisfied at $\bar{x}$; thus, $\bar{x}$ is optimal for the basis pursuit problem. □

In compressed sensing exact recovery occurs only when $\min\{\|x\|_1 : Ax = b\}$ has a *unique* solution. The following Corollary establishes that FAL converges to this solution.

COROLLARY 3.2. *Suppose the basis pursuit problem* $\min\{\|x\|_1 : Ax = b\}$ *has a* unique *optimal solution* $x_*$*. Let* $\{x^{(k)}\}_{k\in\mathbb{Z}_+}$ *denote the sequence of iterates generated by* ALGORITHM FAL*, displayed in Figure 3.1, corresponding to a sequence* $\{(\lambda^{(k)}, \epsilon^{(k)}, \tau^{(k)})\}_{k\in\mathbb{Z}_+}$ *that satisfies all the conditions in Theorem 3.1. Then* $\lim_{k\to\infty} x^{(k)} = x_*$.

Next, we characterize the finite iteration performance of FAL. This analysis will lead to a convergence rate result in Theorem 3.4.

THEOREM 3.3. *Fix* $x^{(0)} \in \mathbb{R}^n$, $\eta > 0$ *such that* $\eta \geq \|x_*\|_1$ *and a sequence of parameters* $\{(\lambda^{(k)}, \epsilon^{(k)}, \tau^{(k)})\}_{k\in\mathbb{Z}_+}$ *satisfying all the conditions in Theorem 3.1. In addition, suppose for all* $k \geq 1$, $\tau^{(k)} \leq c\ \epsilon^{(k)}$ *for some* $0 < c < 1$. *Let* $\{x^{(k)}\}_{k\in\mathbb{Z}_+}$ *denote the sequence of iterates generated by* ALGORITHM FAL*, displayed in Figure 3.1, for this set of parameters. Then, for all* $k \geq 1$,

*(i)* $\|Ax^{(k)} - b\|_2 \leq 2B_\theta\lambda^{(k)}$,

*(ii)* $\big|\|x^{(k)}\|_1 - \|x_*\|_1\big| \leq \max\left\{\left(\frac{B_\theta^2}{2} + B_1\ \max\{1,\ 2cB_x\}\right), \frac{\left(\frac{\sqrt{n}}{\sigma_{\min}(A)} + B_\theta\right)^2}{2}\right\}\lambda^{(k)}$,

*where* $B_\theta = \frac{\sqrt{n}}{\sigma_{\min}(A)}\left(\max\left\{\sqrt{2B_1}\ \sigma_{\max}(A),\ B_2\right\} + 1\right)$, *and* $B_x = \eta + \frac{1}{2}\lambda^{(1)}B_\theta^2$.

*Proof.* First note that we have established the uniform bounds $\|\theta^{(k)}\|_2 \leq B_\theta$ and $\|x^{(k)}\|_1 \leq B_x$ in (3.5) and (3.6), respectively.

The dual update in Line 11 of ALGORITHM FAL implies that

$$\|Ax^{(k)} - b\|_2 \leq \|Ax^{(k)} - b - \lambda^{(k)}\theta^{(k)}\|_2 + \lambda^{(k)}\|\theta^{(k)}\|_2 = \lambda^{(k)}\|\theta^{(k+1)}\|_2 + \lambda^{(k)}\|\theta^{(k)}\|_2 \leq 2B_\theta\lambda^{(k)},$$

where the last inequality follows the fact that $\|\theta^{(k)}\|_2 \leq B_\theta$. This establishes (i).

The dual update in Line 11 of ALGORITHM FAL also implies that

$$P^{(k)}(x^{(k)}) = \lambda^{(k)}\|x^{(k)}\|_1 + \frac{1}{2}\|Ax^{(k)} - b - \lambda^{(k)}\theta^{(k)}\|_2^2 = \lambda^{(k)}\left(\|x^{(k)}\|_1 + \frac{\lambda^{(k)}}{2}\|\theta^{(k+1)}\|_2^2\right).$$

Thus, for all $k \geq 1$,

$$\|x^{(k)}\|_1 \geq \frac{P^{(k)}(x_*^{(k)})}{\lambda^{(k)}} - \frac{\lambda^{(k)}}{2}\|\theta^{(k+1)}\|_2^2. \tag{3.12}$$

Next, we establish a lower bound for $P^{(k)}(x_*^{(k)})$. Consider the following primal-dual pair of problems:

$$\begin{array}{llll} \min_{x\in\mathbb{R}^n} & \|x\|_1 & \max_{w\in\mathbb{R}^m} & b^Tw \\ \text{subject to} & Ax = b, & \text{subject to} & \|A^Tw\|_\infty \leq 1. \end{array} \tag{3.13}$$

Let $w_* \in \mathbb{R}^m$ denote an optimal solution of the maximization problem in (3.13). Next, consider the primal-dual pair of problems corresponding to the penalty formulation for the basis pursuit problem:

$$\begin{array}{llll} \min_{x\in\mathbb{R}} & \lambda\|x\|_1 + \frac{1}{2}\|Ax - b - \lambda\theta\|_2^2 & \max_{w\in\mathbb{R}^m} & \lambda(b + \lambda\theta)^Tw - \frac{\lambda^2}{2}\|w\|_2^2 \\ & & \text{subject to} & \|A^Tw\|_\infty \leq 1. \end{array} \tag{3.14}$$

Since $w_*$ is feasible for the maximization problem in (3.14) is as well, it follows that

$$
\begin{aligned}
P^{(k)}(x_*^{(k)}) &= \min_{x\in\mathbb{R}^n} \{\lambda^{(k)}\|x\|_1 + \frac{1}{2}\|Ax - b - \lambda^{(k)}\theta^{(k)}\|_2^2\} \\
&\geq \lambda^{(k)}\left(b^T w_* + \lambda^{(k)}(\theta^{(k)})^T w_* - \frac{\lambda^{(k)}}{2}\|w_*\|_2^2\right), \qquad (3.15)\\
&\geq \lambda^{(k)}\left(\|x_*\|_1 - \lambda^{(k)}\|\theta^{(k)}\|_2\|w_*\|_2 - \frac{\lambda^{(k)}}{2}\|w_*\|_2^2\right), \qquad (3.16)
\end{aligned}
$$

where (3.15) follows from weak duality for primal-dual problems (3.14) and (3.16) follows from strong duality for primal-dual problems (3.13), i.e. $b^T w_* = \|x_*\|_1$, and Cauchy-Schwartz inequality. Thus, (3.12) implies that

$$
\|x^{(k)}\|_1 \geq \|x_*\|_1 - \lambda^{(k)}\left(\|\theta^{(k)}\|_2\|w_*\|_2 + \frac{1}{2}\|w_*\|_2^2 + \frac{1}{2}\|\theta^{(k+1)}\|_2^2\right) \geq \|x_*\|_1 - \frac{\left(\frac{\sqrt{n}}{\sigma_{\min}(A)} + B_\theta\right)^2}{2}\lambda^{(k)},
$$

where the second inequality follows from the fact that $\|A^T w_*\|_\infty \leq 1$.

The final step in the proof is to establish the upper bound in (ii). Suppose the iterate $x^{(k)}$ satisfies the stopping condition (3.2)(a). In (3.7) we show that

$$
\|x^{(k)}\|_1 \leq \|x_*\|_1 + \frac{\lambda^{(k)}}{2}\|\theta^{(k)}\|_2^2 + \frac{\epsilon^{(k)}}{\lambda^{(k)}} \leq \|x_*\|_1 + \lambda^{(k)}\left(\frac{1}{2}B_\theta^2 + B_1\right), \qquad (3.17)
$$

where we have used the fact that $\|\theta^{(k)}\|_2 \leq B_\theta$ and $\frac{\epsilon^{(k)}}{(\lambda^{(k)})^2} \leq B_1$.

Next, suppose $x^{(k)}$ satisfies (3.2)(b). Let $g^{(k)} \in \partial P^{(k)}(x^{(k)})$ such that $\|g^{(k)}\|_2 \leq \tau^{(k)}$. Then the convexity of $P^{(k)}$ implies that

$$
P^{(k)}(x^{(k)}) - P^{(k)}(x_*) \leq -(g^{(k)})^T(x_* - x^{(k)}) \leq \|g^{(k)}\|_2\|x_* - x^{(k)}\|_2 \leq \tau^{(k)}\|x_* - x^{(k)}\|_2. \qquad (3.18)
$$

Now, an argument similar to the one that establishes the bound (3.7), implies that

$$
\|x^{(k)}\|_1 \leq \|x_*\|_1 + \frac{\lambda^{(k)}}{2}\|\theta^{(k)}\|_2^2 + \frac{\tau^{(k)}}{\lambda^{(k)}}\|x_* - x^{(k)}\|_2 \leq \|x_*\|_1 + \lambda^{(k)}\left(\frac{1}{2}B_\theta^2 + 2cB_xB_1\right), \qquad (3.19)
$$

where we have used the fact that $\|\theta^{(k)}\|_2 \leq B_\theta$, $\tau^{(k)} \leq c\epsilon^{(k)}$, and $\frac{\epsilon^{(k)}}{(\lambda^{(k)})^2} \leq B_1$. The upper bound in (ii) follows from (3.17) and (3.19). □

Next, we use the bounds in Theorem 3.3 to compute a bound on the convergence rate of ALGORITHM FAL.

THEOREM 3.4. *Fix an $0 < \alpha < 1$. Then there exists and one can construct a sequence of parameters $\{(\lambda^{(k)}, \epsilon^{(k)}, \tau^{(k)})\}_{k\in\mathbb{Z}_+}$ such that the iterates generated by* ALGORITHM FAL*, displayed in Figure 3.1, are $\epsilon$-feasible, i.e. $\|Ax^{(k)} - b\|_2 \leq \epsilon$, and $\epsilon$-optimal, $\big|\, \|x^{(k)}\|_1 - \|x_*\|_1\big| \leq \epsilon$, for all $k \geq N_{\text{FAL}}(\epsilon) = \mathcal{O}\left(\log_{\frac{1}{\alpha}}\left(\frac{1}{\epsilon}\right)\right)$. Moreover, FAL requires*

$$
N_{\text{mat}} \leq 2n\kappa(A)^2\left(\frac{16\|x_*\|_1}{\alpha(1-\alpha)}\cdot\frac{1}{\epsilon} + \frac{9}{\alpha}\cdot\log_{\frac{1}{\alpha}}\left(\frac{8n\kappa^2(A)}{\epsilon}\right)\right) = \mathcal{O}\left(\frac{1}{\epsilon}\right), \qquad (3.20)
$$

*matrix-vector multiplies to compute an $\epsilon$-feasible, $\epsilon$-optimal iterate.*

*Proof.* Rescale the problem parameters $(\bar{A}, \bar{b}) = \frac{1}{\sigma_{\max}(A)}(A, b)$. Then for the rescaled problem $L = \sigma_{\max}(\bar{A}) = 1$, but the condition number $\kappa(\bar{A}) = \kappa(A)$. We will use ALGORITHM FAL to solve the rescaled problem $(\bar{A}, \bar{b})$.

Set $\lambda^{(1)} = 1$, $\epsilon^{(1)} = 2$, and update

$$
\begin{aligned}
\lambda^{(k+1)} &= \alpha\cdot\lambda^{(k)}, \\
\epsilon^{(k+1)} &= \alpha^2\cdot\epsilon^{(k)}, \qquad (3.21)\\
\tau^{(k)} &= \frac{1}{2\max\{1, \eta + \frac{9n}{2}\kappa(A)^2\}}\cdot\epsilon^{(k)}.
\end{aligned}
$$

For this choice of problem parameters, the constants

$$B_1 = \max_{k\geq 1}\left\{\frac{\epsilon^{(k)}}{(\lambda^{(k)})^2}\right\} = \frac{\epsilon^{(1)}}{(\lambda^{(1)})^2} = 2, \quad B_2 = \max_{k\geq 1}\left\{\frac{\tau^{(k)}}{\epsilon^{(k)}}\right\} = \frac{\alpha^k}{2\max\{1, \eta + \frac{9n}{2}\kappa(A)^2\}} \leq 1.$$

Therefore, the uniform bounds on $\|\theta^{(k)}\|_2$ and $\|x^{(k)}\|_1$ are given by

$$B_\theta = \frac{\sqrt{n}}{\sigma_{\min}(\bar{A})}\left(\max\{\sqrt{2B_1}\;\sigma_{\max}(\bar{A}),\; B_2\} + 1\right) \leq 3\kappa(A)\sqrt{n}, \quad B_x = \eta + \frac{1}{2}B_\theta^2 \leq \eta + \frac{9n}{2}\kappa(A)^2.$$

For the rescaled problem $(\bar{A}, \bar{b})$, the Theorem 3.3 guarantees that for all $k \geq 1$,

$$\left|\|x^{(k)}\|_1 - \|x_*\|_1\right| \leq \max\left\{\left(\frac{B_\theta^2}{2} + B_1\right), \frac{\left(\sqrt{n}\kappa(A) + B_\theta\right)^2}{2}\right\}\lambda^{(1)}\;\alpha^{k-1} \leq 8n\kappa(A)^2\alpha^{k-1},$$

where we use the fact that $\kappa(A) \geq 1$. Thus, $\left|\|x^{(k)}\|_1 - \|x_*\|_1\right| \leq \epsilon$, for all $k \in \mathbb{Z}_+$ such that

$$k \geq \ln_{\frac{1}{\alpha}}\left(\frac{8n\kappa(A)^2}{\epsilon}\right) + 1. \tag{3.22}$$

From Theorem 3.3 we also have that for all $k \geq 1$,

$$\|Ax^{(k)} - b\|_2 \leq 2B_\theta\lambda^{(1)}\;\alpha^{k-1} \leq 6\sqrt{n}\kappa(A).$$

Thus $\|Ax^{(k)} - b\|_2 \leq \epsilon$ for all

$$k \geq \ln_{\frac{1}{\alpha}}\left(\frac{6\sqrt{n}\kappa(A)}{\epsilon}\right) + 1. \tag{3.23}$$

From(3.22) and (3.23) it follows that for all $\epsilon > 0$, $N_{\text{FAL}}(\epsilon)$, the number of FAL iterations required to compute an $\epsilon$-feasible and $\epsilon$-optimal solution, is at most

$$N_{\text{FAL}}(\epsilon) \leq \left\lceil \ln_{\frac{1}{\alpha}}\left(\frac{8n\kappa(A)^2}{\epsilon}\right)\right\rceil + 1. \tag{3.24}$$

From the stopping condition APGSTOP defined in Line 9 of ALGORITHM FAL, it follows that the total number of the ALGORITHM APG iterations, $N_{\text{APG}}$, required during $N_{\text{FAL}}(\epsilon)$ many FAL iterations is bounded by

$$\begin{aligned}
\bar{N}_{\text{APG}} &= \sum_{k=1}^{N_{\text{FAL}}(\epsilon)} \left\lceil \ell_{\max}^{(k)} \right\rceil \\
&= \sum_{k=1}^{N_{\text{FAL}}(\epsilon)} \left\lceil \sigma_{\max}(\bar{A})\left(\eta^{(k)} + \|x^{(k-1)}\|_2\right)\sqrt{\frac{2}{\epsilon^{(k)}}}\right\rceil \\
&\leq \sum_{k=1}^{N_{\text{FAL}}(\epsilon)} \left(2\eta + \lambda^{(k-1)}B_\theta^2\right)\sqrt{\frac{2}{\epsilon^{(k)}}} \\
&= 2\eta\sum_{k=0}^{N_{\text{FAL}}(\epsilon)-1} \alpha^{-k} + \frac{B_\theta^2}{\alpha}\cdot N_{\text{FAL}}(\epsilon) &\quad (3.25)\\
&\leq \frac{2\alpha\eta}{1-\alpha}\cdot\alpha^{-N_{\text{FAL}}(\epsilon)} + \frac{B_\theta^2}{\alpha}\cdot N_{\text{FAL}}(\epsilon), &\quad (3.26)
\end{aligned}$$

where the first inequality follows from the fact that $\|x^{(k-1)}\|_2 \leq \|x^{(k-1)}\|_1 \leq \eta^{(k-1)}$ and $\|\theta^{(k)}\|_2 \leq B_\theta$, the third equality follows from substituting for the parameters $\lambda^{(k-1)}$ and $\epsilon^{(k)}$, and the last inequality follows from the summing the geometric series.

Algorithm FAL calls Algorithm APG with a quadratic prox function of the form $h(x) = \frac{1}{2}\|x - \bar{x}\|_2^2$ and the smooth function of the form $f(x) = \frac{1}{2}\|Ax - b - \lambda\theta\|_2^2$. In each iteration of Algorithm APG, we need to compute the gradient $\nabla f(x) = A^T(Ax - b - \lambda\theta)$ and solve two constrained shrinkage problems of the form

$$\min_{x \in \mathbb{R}^n} \left\{ \lambda\|x\|_1 + \frac{1}{2}\|x - \tilde{x}\|_2^2 : \ \|x\|_1 \leq \eta \right\}$$

We show in Lemma A.4 in Appendix A that the complexity of solving a constrained shrinkage problem is $\mathcal{O}(n\log(n))$. Thus, the computational complexity of each Algorithm APG iteration is dominated by the complexity of computing $A^T(Ax - b - \lambda\theta)$. The complexity result now follows from the bound in (3.26). $\square$

## 4. Extension of FAL to noisy recovery.

In this section, we briefly discuss how FAL can be extended to solve the noisy signal recovery problem of the form (1.3). See [1] for the further extensions of the methodology proposed here. Consider a noisy recovery problem

$$\begin{array}{rl} \min & \|x\|_1, \\ \text{s.t.} & \|Ax - b\|_\gamma \leq \delta, \end{array}$$

where $\gamma \in \{1, 2, \infty\}$. The formulation with $\gamma \in \{1, \infty\}$ are interesting when the measurement noise has a Laplacian or Extreme Value distribution. By introducing a slack variable $s \in \mathbb{R}^m$, the noisy recovery problem can be formulated as follows:

$$\begin{array}{rl} \min & \|x\|_1, \\ \text{s.t.} & Ax + s = b, \quad \|s\|_\gamma \leq \delta, \end{array} \tag{4.1}$$

We solve (4.1) by inexactly minimizing a sequence of sub-problems of the form

$$P^{(k)}(x, s) = \lambda^{(k)}\|x\|_1 + \frac{1}{2}\|Ax + s - b - \lambda^{(k)}\theta^{(k)}\|_2^2 \tag{4.2}$$

over sets $F^{(k)} = \{(x, s) : \|x\|_1 \leq \eta^{(k)}, \|s\|_\gamma \leq \delta\}$ using Algorithm APG with the prox function $h^{(k)}(x, s) := \frac{1}{2}\|x - x^{(k-1)}\|_2^2 + \frac{1}{2}\|s - s^{(k-1)}\|_2^2$ and initial iterate $(x^{(k-1)}, s^{(k-1)})$, where $\eta^{(k)} := \eta + \frac{\lambda^{(k)}}{2}\|\theta^{(k)}\|_2^2$.

In order to efficiently solve (4.1) we need a good stopping condition for terminating Algorithm APG. Since $\max\{h^{(k)}(x, s) : (x, s) \in F^{(k)}\} \leq (\mu^{(k)})^2 := \frac{1}{2}\left(\eta^{(k)} + \|x^{(k-1)}\|_2\right)^2 + \frac{1}{2}\left(\upsilon(\gamma)\delta + \|s^{(k-1)}\|_2\right)^2$, where $\upsilon(\gamma) = 1$, when $\gamma = 1, 2$ and $\sqrt{m}$ when $\gamma = \infty$, Lemma 2.1 implies that terminating Algorithm APG at iteration $\left\lceil \ell_{\max}^{(k)} \right\rceil$, where $\ell_{\max}^{(k)} := \sigma_{\max}(A)\mu^{(k)}\sqrt{\frac{2}{\epsilon^{(k)}}}$, guarantees that $(x^{(k)}, s^{(k)})$ is $\epsilon^{(k)}$-optimal for $P^{(k)}$.

Recall that in solving the basis pursuit problem using Algorithm FAL we terminate Algorithm APG when either we are guaranteed that the iterate $x^{(k)}$ is $\epsilon^{(k)}$-optimal for $P^{(k)}$ or there exists a sub-gradient $g^{(k)} \in \partial P^{(k)}(x^{(k)})$ with a sufficiently small norm. In our numerical experiments, we found that we always terminated the call to Algorithm APG using the sub-gradient stopping condition. In order to extend FAL to efficiently solve (4.1) we need an analog of the sub-gradient condition.

In FAL we were able to set the tolerance $\tau^{(k)}$ small since we are guaranteed that $\operatorname{argmin}_{x \in \mathbb{R}^n}\{P^{(k)}(x)\} \subseteq F^{(k)}$. Let $\partial_x P^{(k)}(x, s)$ denote the projection of the set of sub-gradients on the $x$-variables. The definition of $F^{(k)}$ guarantees that $x_*^{(k)} \in F^{(k)}$ for all $(x_*^{(k)}, s_*^{(k)}) \in \operatorname{argmin}\{P^{(k)}(x, s) : x \in \mathbb{R}^n, \|s\|_\gamma \leq \delta\}$. Therefore, we can continue to use the gradient condition $g_x \in \partial_x P^{(k)}(x^{(k)}, s^{(k)})$ with $\|g_x\|_2 \leq \tau_x$. However, since $s$ is constrained, we cannot force $\|g_s\|_2$ to be close to zero; therefore, we need an alternative gradient condition.

Fix $x^{(k)}$. Define $\zeta(s) = P^{(k)}(x^{(k)}, s)$ and $Q = \{s : \|s\|_\gamma \leq \delta\}$. The function $\zeta(s)$ is differentiable and $\nabla\zeta(s)$ is Lipschitz continuous. Let $\pi_Q(s) := \min_{y \in Q}\{\|y - (s - \frac{1}{L}\nabla\zeta(s))\|_2\}$ denote the projection of the gradient step $s - \frac{1}{L}\nabla\zeta(s)$ onto the constraint set $Q$. Then Theorem 2.2.7 in [19] establishes that $\hat{s} \in \mathrm{argmin}\{P^{(k)}(x^{(k)}, s) : \|s\|_\gamma \leq \delta\}$ if and only if $d(\hat{s}, Q) := L\|\hat{s} - \pi_Q(\hat{s})\| = 0$. Thus, we set the stopping condition for ALGORITHM APG as follows:

$$\text{APGSTOP} := \left\{\ell \geq \ell_{\max}^{(k)} := \sigma_{\max}(A)\mu^{(k)}\sqrt{\frac{2}{\epsilon^{(k)}}}\right\} \textbf{ or}$$
$$\left\{\exists\ g_x \in \partial_x P^{(k)}(x,s)|_{v_x^{(\ell)}, v_s^{(\ell)}} \text{ with } \|g_x\|_2 \leq \tau_x^{(k)},\ \text{and } d\left(v_s^{(\ell)}, F^{(k)}\right) \leq \tau_s^{(k)}\right\},$$

where $v_x^{(\ell)}$ and $v_s^{(\ell)}$ are the components of $v^{(\ell)}$ corresponding to $x$ and $s$ variables in (4.2).

Consequently, it follows that the stopping criterion APGSTOP ensures that the iterate $(x^{(k)}, s^{(k)})$ satisfies one of the following two conditions

$$\begin{array}{ll} (a) & P^{(k)}(x^{(k)}, s^{(k)}) \leq \min_{x\in\mathbb{R}^n, \|s\|_\gamma \leq \delta} P^{(k)}(x,s) + \epsilon^{(k)}, \\ (b) & \exists\ g_x^{(k)} \in \partial_x P^{(k)}(x,s)|_{x^{(k)}, s^{(k)}} \text{ with } \|g_x^{(k)}\|_2 \leq \tau_x^{(k)},\ \text{and } d\left(s^{(k)}, F^{(k)}\right) \leq \tau_s^{(k)}. \end{array} \tag{4.3}$$

With this modification, Theorem 3.1, Corollary 3.2, Theorem 3.3 and Theorem 3.4 all remain valid for the relaxed recovery problem (4.1). Thus, FAL efficiently computes a solution for the noisy recovery problem (1.3). The per iteration cost of FAL is the cost of computing $A^T(Ax - b)$. Thus, the per iteration complexity of FAL is always $\mathcal{O}(n^2)$ for any $A$; whereas the per iteration complexity of NESTA [4] is $\mathcal{O}(n^3)$ when $A$ is not orthogonal. Examples of non-orthogonal $A$ include Gaussian measurement matrices, partial psuedo-polar Fourier tranforms, and non-orthogonal partial wavelet transforms.

## 5. Implementation details of Algorithm FAL for numerical experiments.

In this section we describe the details of the implemented version of ALGORITHM FAL that we used in our numerical experiments described in the next section.

### 5.1. Initial iterate $x^{(0)}$ and bound $\eta$.

In our numerical experiments, when $A$ was a partial DCT matrix, we set $x^{(0)} = \mathrm{argmin}\{\|x\|_2 : Ax = b\} = A^T(AA^T)^{-1}b = A^Tb$, where the last equality follows from the fact that $A$ has orthogonal rows. The complexity of computing $x^{(0)}$ in this case is $\mathcal{O}(n\log(n))$. Since $x_* \in \mathrm{argmin}\{\|x\|_1 :\ Ax = b\}$ and $Ax^{(0)} = b$, we have $\|x_*\|_1 \leq \|x^{(0)}\|_1$, we use $\eta = \|x^{(0)}\|_1$ in FAL.

For general $A$, the computational cost of computing $A(AA^T)^{-1}b$ is $\mathcal{O}(n^2 + m^3)$. In order to avoid $\mathcal{O}(m^3)$ inversion cost, we set $x^{(0)} = A^Tb$ when $A$ was a standard Gaussian matrix, i.e. each element $A_{ij}$ is an independent sample from the standard $\mathcal{N}(0,1)$. Since $\|x^{(0)}\|_1 = \|A^Tb\|_1$ is no longer an upper bound on $\|x_*\|_1$, we set $\eta$ as follows. It well known that for an $m\times n$ standard Gaussian matrix $\sigma_{\min}(A) \approx \left(1 - \sqrt{\frac{m}{n}}\right)\sqrt{n}$ for large $n$ (in fact, the approximation is very accurate even at $n = 100$) [14]. Then,

$$\|x_*\|_1 \leq \|A^T(AA^T)^{-1}b\|_1 \leq \frac{\sqrt{n}}{\sigma_{\min}(A)}\|b\|_2 \leq \eta := \frac{1}{1 - \sqrt{\frac{m}{n}}}\|b\|_2. \tag{5.1}$$

### 5.2. APGstop and FALstop conditions.

When the ALGORITHM APG terminates with $\ell \geq \ell_{\max}^{(k)}$ we return $u^{(l)}$. Since gradient computation is computationally the most expensive step in ALGORITHM APG, and we are required to compute the gradient at the $v$ iterates, we checked the sub-gradient stopping condition at the these iterates. Hence, we stopped ALGORITHM APG and returned $v^{(\ell)}$ when $\min\{\|g\|_2^2 : g \in \partial P^{(k)}(x)|_{v^{(\ell)}}\} \leq \tau^{(k)}$.

$g \in \partial P^{(k)}(x)|_{v^{(\ell)}}$ if, and only if, there exists $q \in \partial\|x\|_1\ |_{v^{(\ell)}}$ such that $g = \lambda^{(k)}q + \nabla f^{(k,l)}$, where $\nabla f^{(k,l)} :=$

$\nabla f^{(k)}(v^{(\ell)})$. Thus, it follows that

$$\begin{aligned}
\min\{\|g\|_2^2 : g \in \partial P^{(k)}(x)|_{v^{(\ell)}}\} &= \min\{\|\lambda^{(k)} q + \nabla f^{(k,l)}\|_2^2 : q \in \partial\|x\|_1 \;|_{v^{(\ell)}}\} \\
&= \sum_{\{i: v_i^{(\ell)} > 0\}} |\lambda^{(k)} + \nabla_i \nabla f_i^{(k,l)}|^2 + \sum_{\{i: v_i^{(\ell)} < 0\}} |-\lambda^{(k)} + \nabla f_i^{(k,l)}|^2 \\
&\quad + \sum_{\{i: v_i^{(\ell)} = 0\}} \min_{q_i \in [-1,1]} |\lambda^{(k)} q_i + \nabla f_i^{(k,l)}|^2, \\
&= \sum_{\{i: v_i^{(\ell)} > 0\}} |\lambda^{(k)} + \nabla f_i^{(k,l)}|^2 + \sum_{\{i: v_i^{(\ell)} < 0\}} |-\lambda^{(k)} + \nabla f_i^{(k,l)}|^2 \\
&\quad + \sum_{\{i: v_i^{(\ell)} = 0\}} \min\{\big||\nabla f_i^{(k,l)}| - \lambda^{(k)}\big|, 0\},
\end{aligned}$$

where $\nabla f_i^{(k,l)}$ denote the $i$-th component of the gradient $\nabla f^{(k,l)}$. The first equality follows from the fact $q_i = +1$ (resp. $-1$) whenever $v_i^{(\ell)} > 0$ (resp. $v_i^{(\ell)} < 0$), and $q_i \in [-1,1]$ for $i$ such that $v_i^{(\ell)} = 0$, and the last equality follows from explicitly computing the minimum. Thus, given $\nabla f^{(k,l)}$, the complexity of computing $g$ is $\mathcal{O}(n)$.

We used different stopping conditions depending on the existence of measurement noise. First, we modified the stopping condition APGstop as follows

$$\text{APGSTOP} = \text{FALSTOP} \ \textbf{or} \ \left\{\ell \geq \ell_{\max}^{(k)}\right\} \ \textbf{or} \ \left\{\exists g \in \partial P^{(k)}(x)|_{v^{(\ell)}} \text{ with } \|g\|_2 \leq \tau^{(k)}\right\},$$

Then, we set the FAL stopping condition FALstop as follows.

(i) Noiseless measurements:

$$\text{FALSTOP} = \{\|u^{(l)} - u^{(l-1)}\|_\infty \leq \gamma\}, \tag{5.2}$$

where we set the threshold $\gamma$ by experimenting with a small instance of the problem. Algorithm FAL produces $x_{sol} = u^{(\ell)}$ when FALstop is **true**.

(ii) Noisy measurements:

$$\text{FALSTOP} = \left\{\frac{\|u^{(l)} - u^{(l-1)}\|_2}{\|u^{(l-1)}\|_2} \leq \gamma\right\}, \tag{5.3}$$

where $\gamma$ was set equal to the standard deviation of the noise, i.e. when $b = Ax_* + \zeta$ such that $\zeta$ is a vector of i.i.d. random variables with standard deviation $\varrho$, we set $\gamma = \varrho$. As in the noiseless case, we terminated FAL and set $x_{sol} = u^{(\ell)}$ when FALstop is **true**.

**5.3. Parameter sequence selection.** Recall that we require

$$\lambda^{(k)} \searrow 0, \qquad \epsilon^{(k)} \searrow 0, \quad \frac{\epsilon^{(k)}}{(\lambda^{(k)})^2} \leq B_1, \qquad \tau^{(k)} \searrow 0, \quad \frac{\tau^{(k)}}{\lambda^{(k)}} \to 0, \qquad \tau^{(k)} \leq c\epsilon^{(k)},$$

for Theorem 3.1, Corollary 3.2, Theorem 3.3 and Theorem 3.4 to hold. These conditions are satisfied if one updates the parameters as follows: for $k \geq 1$,

$$\lambda^{(k+1)} = c_\lambda \, \lambda^{(k)}, \qquad \tau^{(k+1)} = \min\{c_\tau \, \tau^{(k)}, \hat{c}_\tau \, \|g_0^{(k+1)}\|_2\}, \qquad \epsilon^{(k+1)} = c_\lambda^2 \, \epsilon^{(k)},$$

where $g_0^{(k)} = \text{argmin}\{\|\lambda^{(k)} q + \nabla f^{(k)}(x^{(k-1)})\|_2^2 : q \in \partial\|x\|_1 \;|_{x^{(k-1)}}\}$ is the minimum norm sub-gradient at the initial iterate $x^{(k-1)}$ for the $k$-th sub-problem for $k \geq 1$, and $c_\lambda$, $c_\tau$ and $\hat{c}_\lambda$ are appropriately chosen constants in $(0,1)$. Note that we still have to set the initial iterates $\lambda^{(1)}$, $\tau^{(1)}$, and $\epsilon^{(1)}$.

In our preliminary numerical experiments with FAL we found that it was sufficient to set $\hat{c}_\tau = 0.9$, and the optimal choice for the constants $c_\lambda$ and $c_\tau$ was only a function sparsity ratio $\xi = \|x_*\|_0/m$, and was effectively independent of the problem size $n$. Moreover, the optimal choice for $c_\tau = c_\lambda - 0.01$. We set

$$c_\lambda(\xi) = \begin{cases} 0.9, & \text{if } \xi \geq 0.9; \\ 0.85, & \text{if } 0.9 > \xi \geq 0.6; \\ 0.8, & \text{if } 0.6 > \xi \geq 0.25; \\ 0.6, & \text{if } 0.25 > \xi \geq 0.1; \\ 0.4, & \text{if } \xi < 0.1, \end{cases} \tag{5.4}$$

and approximated the sparsity ratio $\xi$ at the beginning of $k$-th FAL by $\xi^{(k)} = \|x^{(k-1)}\|_0/m$. Since $x^{(0)}$ is set arbitrarily, and is unlikely to be sparse, we use the following parameter update rule for $k \geq 2$,

$$\tau^{(k)} = \min\left\{c_\tau(\xi^{(k)})\,\tau^{(k-1)}, \hat{c}_\tau\|g_0^{(k)}\|_2\right\}, \quad \lambda^{(k)} = c_\lambda(\xi^{(k)})\,\lambda^{(k-1)}, \quad \epsilon^{(k+1)} = c_\lambda(\xi^{(k)})^2\,\epsilon^{(k)}, \tag{5.5}$$

where $c_\tau(\xi^{(k)}) = c_\lambda(\xi^{(k)}) - 0.01$, and $\hat{c}_\tau = 0.9$. We set $c_\lambda^{(1)}$ and $c_\tau^{(1)}$ for each problem class separately.

We set the initial parameter values $\lambda^{(1)}$, $\tau^{(1)}$ and $\epsilon^{(1)}$ as follows. Let $x^{(0)} = A^T b$ denote the initial FAL iterate. We set

$$\tau^{(1)} = \hat{c}_\tau\|g_0^{(1)}\|_2, \qquad \lambda^{(1)} = 0.99\|x^{(0)}\|_\infty.$$

We use the duality gap at $x^{(0)}$ to set $\epsilon^{(1)}$. Since the bound $\eta^{(1)} = \eta$ is set to ensure that $\operatorname{argmin}_{x\in\mathbb{R}} P^{(1)}(x) \in F^{(k)} = \{x : \|x\|_1 \leq \eta^{(1)}\}$, we are effectively computing an unconstrained minimum. Since $P^{(1)}(x) \geq 0$ for all $x \in \mathbb{R}^n$, it follows that the duality gap of the initial iterate $x^{(0)}$ is at most $P^{(1)}(x^{(0)})$. We set $\epsilon^{(1)} = 0.99P^{(1)}(x^{(0)})$. Then set $\epsilon^{(k+1)} = \left(c_\lambda^{(k)}\right)^2\epsilon^{(k)}$ for all $k \geq 1$.

The step length is Algorithm APG is proportional to $\frac{1}{L}$ where $L$ denotes the Lipschitz constant of $\nabla f$. In our numerical tests, we observed that taking long steps, i.e. steps of size $\frac{t}{L}$, for $t > 1$, improves the speed of convergence in practice. And we chose the step-size $t$ as a function of the sparsity ratio $\|x_*\|_0/m$. In iteration $k$, we approximate the sparsity ratio by $\xi^{(k)} = \|x^{(k-1)}\|_0/m$, and set the Lipschitz constant to be used in Algorithm APG to

$$L^{(k)} = \frac{\sigma_{\max}(AA^T)}{t(\xi^{(k)})},$$

where the function

$$t(\xi) = \begin{cases} 1.8, & \text{if } \xi \geq 0.9; \\ 1.85, & \text{if } 0.9 > \xi \geq 0.6; \\ 1.9, & \text{if } 0.6 > \xi \geq 0.25; \\ 2, & \text{if } 0.25 > \xi \geq 0.1; \\ 3, & \text{if } \xi < 0.1. \end{cases} \tag{5.6}$$

## 6. Numerical experiments.

We conducted three sets of numerical experiments with FAL.

1. In the first set of experiments we solve randomly generated basis pursuit problems when there is no measurement noise. Our goal in this set of experiments were to benchmark the practical performance of FAL and to compare FAL with the Nesterov-type algorithms, SPA [2], and NESTA [4], fixed point continuation algorithms, FPC, FPC-BB [16, 17], and FPC-AS [23], the alternating direction proximal gradient method YALL1 [24], and a root finding algorithm SPGL1 [22]. We describe the results for this set of experiments in Section 6.1.1.
2. In the second set of experiments, we compare the performance of FAL with the performances of the same set solvers, on randomly generated basis pursuit denoising problems when there is a non-trivial level of noise on the measurement vector $b$. The results for this set of experiments is described in Section 6.2.1.

3. In the third set of experiments, we compare the performance and robustness of FAL with the same solvers on a set of small sized, hard compressed sensing problems, CaltechTest[5]. The results for this set of experiments is reported in Section 6.3.

All the numerical experiments were conducted on a desktop with 4 dual-core AMD Opteron 2218 @2.6 GHz processors, 16GB RAM running MATLAB 7.12 on Fedora 14 operating system.

### 6.1. Experiments with no measurement noise.

**6.1.1. Signal generation.** We generated the target signal $x_*$ and the measurement matrix $A$ using the experimental setup in [4]. In particular, we set

$$(x_*)_i = \mathbf{1}(i \in \Lambda)\ \Theta_i^{(1)} 10^{5\Theta_i^{(2)}}, \tag{6.1}$$

where,

(i) $\Lambda$ is constructed by randomly selecting $s$ indices from the set $\{1, \ldots, n\}$,
(ii) $\Theta_i^{(1)}$, $i \in \Lambda$, are IID Bernoulli random variables taking values $\pm 1$ with equal probability,
(iii) $\Theta_i^{(2)}$, $i \in \Lambda$, are IID uniform $[0, 1]$ random variables.

We then scale $\Theta^{(2)}$ such that $\min_i \Theta_i^{(2)} = 0$ and $\max_i \Theta_i^{(2)} = 1$. Therefore, the signal $x_*$ has a dynamic range of $100dB$.

We randomly selected $m = \frac{n}{4}$ frequencies from the set $\{0, \ldots, n\}$ and set the measurement matrix $A \in \Re^{m\times n}$ to the partial DCT matrix corresponding to the chosen frequencies. The measurement vector, $b$, is then set to the DCT evaluated at the chosen frequencies, i.e. $b = Ax_*$.

**6.1.2. Algorithm scaling results.** For this set of numerical experiments,

$$\text{FAL}\textsc{stop} = \{\|u^{(\ell)} - x_*\|_\infty \leq \gamma\}, \tag{6.2}$$

and Algorithm FAL produces $x_{sol} = u^{(\ell)}$ when FALstop is **true**, where $x_*$ is the randomly generated target signal. Since the largest magnitude of the target signal, i.e. $\max_i |(x_*)_i|$ is $10^5$, the stopping condition FALstop implies that $x_{sol}$ has $5 + \log_{10}(1/\gamma)$ digits of accuracy. We report results for $\gamma = 1$, $10^{-1}$ and $10^{-2}$. For the first iteration of FAL, we set $c_\tau^{(1)} = c_\lambda^{(1)} = 0.4$. For $k \geq 2$, we used the functions $c_\lambda(\cdot)$ and $t(\cdot)$ described in (5.4) and (5.6), respectively. The parameters $(\lambda^{(k)}, \tau^{(k)}, \epsilon^{(k)})$ were set as described in Section 5.3.

| Sparsity | $\gamma$ | Table |
|---|---|---|
| $s = m/100$ | 1 | Table 6.2 |
| $s = m/100$ | 0.1 | Table 6.3 |
| $s = m/100$ | 0.01 | Table 6.4 |
| $s = m/10$ | 1 | Table 6.5 |
| $s = m/10$ | 0.1 | Table 6.6 |
| $s = m/10$ | 0.01 | Table 6.7 |

Table 6.1
*Summary of numerical experiments*

The Table 6.1 summarizes the sparsity conditions and the parameter settings for this set of experiments. The column marked `Table` lists the table where we display the results corresponding to the parameter setting of the particular row, e.g. the results for $s = m/10$ and $\gamma = 0.1$ are displayed in Table 6.6.

We generated 10 random instances for each of the experimental conditions. In Tables 6.2–6.7, the column labeled `average` lists the average taken over the 10 random instances, the columns labeled `max` list the maximum over the 10 instances. The rows labeled $\mathbf{N}_{\text{FAL}}$ and $\mathbf{N}_{\text{APG}}$ list the total number of FAL and APG iterations required, respectively, to solve the instance for the given tolerance parameter $\gamma$. The row labeled **CPU** lists the running time in seconds and the row labeled **nMat** lists the total number of matrix-vector multiplies of the form $Ax$ or $A^Ty$ computed during the FAL run. In Section 6.1.3, we report two

**nMat** numbers for FPC-AS: the first one is the number of multiplications with $A$ during the fixed point iterations and the second one is the number of multiplications with a reduced form of $A$ during the subspace optimization iterations. All other rows are self-explanatory.

The experiment results support the following conclusions. FAL is very efficient - it requires only 11-20 iterations to converge to an high accuracy solution of the basis pursuit problem. For a given sparsity level $s$ and a stopping criterion $\gamma$, $\mathbf{N_{FAL}}$ is a very slowly growing function of the dimension $n$ of the target signal. The total number of matrix-vector multiplies increases with the number of non-zero elements in the target signal $x_*$ – increasing $s$ from $m/100$ to $m/10$ increases the number of matrix-vector multiplies by 30%. On problems with high sparsity, FAL always recovers the support of the target signal. We find that FAL is always able to discover the support of the target signal when the tolerance $\gamma$ is set sufficiently low.

| | **n=512×512** | | **n=256×256** | | **n=64×64** | |
|---|---|---|---|---|---|---|
| | **Average** | **Max** | **Average** | **Max** | **Average** | **Max** |
| $\mathbf{N}_{\text{APG}}$ | 27.0 | 27 | 26.5 | 27 | 29.3 | 34 |
| $\lvert\lVert\mathbf{x_{sol}}\rVert_1 - \lVert\mathbf{x_*}\rVert_1\rvert/\lVert\mathbf{x_*}\rVert_1$ | 1.70E-06 | 2.12E-06 | 3.96E-06 | 1.12E-05 | 1.29E-05 | 2.62E-05 |
| $\max\{\lvert(\mathbf{x_{sol}})_i - (\mathbf{x_*})_i\rvert : (\mathbf{x_*})_i \neq \mathbf{0}\}$ | 3.23E-01 | 3.78E-01 | 5.73E-01 | 1.00E+00 | 9.09E-01 | 1.00E+00 |
| $\max\{\lvert(\mathbf{x_{sol}})_i\rvert : (\mathbf{x_*})_i = \mathbf{0}\}$ | 0.00E+00 | 0.00E+00 | 0.00E+00 | 0.00E+00 | 0.00E+00 | 0.00E+00 |
| $\lVert\mathbf{Ax_{sol}} - \mathbf{b}\rVert_2$ | 0.703 | 0.781 | 0.799 | 1.809 | 0.604 | 0.850 |
| $\lVert\mathbf{x_{sol}}\rVert_1$ | 5588229.9 | 7000555.1 | 1508014.9 | 1838186.7 | 193826.1 | 311446.4 |
| $\lVert\mathbf{x_*}\rVert_1$ | 5588239.3 | 7000565.2 | 1508021.0 | 1838194.7 | 193828.2 | 311447.1 |
| **CPU** | 13.5 | 13.7 | 3.1 | 3.1 | 0.2 | 0.3 |
| $\mathbf{N}_{\text{FAL}}$ | 14.0 | 14.0 | 13.6 | 14.0 | 12.6 | 14.0 |
| **nMat** | 56 | 56 | 55 | 56 | 60.6 | 70 |

TABLE 6.2
*FAL scaling results: $m = n/4$, $s = m/100$ and $\|x_{sol} - x_*\|_\infty \leq 1$*

| | **n=512×512** | | **n=256×256** | | **n=64×64** | |
|---|---|---|---|---|---|---|
| | **Average** | **Max** | **Average** | **Max** | **Average** | **Max** |
| $\mathbf{N}_{\text{APG}}$ | 28.9 | 29 | 28.4 | 29 | 35.1 | 45 |
| $\lvert\lVert\mathbf{x_{sol}}\rVert_1 - \lVert\mathbf{x_*}\rVert_1\rvert/\lVert\mathbf{x_*}\rVert_1$ | 6.79E-08 | 2.69E-07 | 1.03E-07 | 2.49E-07 | 4.96E-07 | 1.06E-06 |
| $\max\{\lvert(\mathbf{x_{sol}})_i - (\mathbf{x_*})_i\rvert : (\mathbf{x_*})_i \neq \mathbf{0}\}$ | 2.58E-02 | 9.51E-02 | 5.57E-02 | 9.07E-02 | 6.22E-02 | 8.76E-02 |
| $\max\{\lvert(\mathbf{x_{sol}})_i\rvert : (\mathbf{x_*})_i = \mathbf{0}\}$ | 0.00E+00 | 0.00E+00 | 0.00E+00 | 0.00E+00 | 0.00E+00 | 0.00E+00 |
| $\lVert\mathbf{Ax_{sol}} - \mathbf{b}\rVert_2$ | 0.047 | 0.187 | 0.062 | 0.109 | 0.037 | 0.065 |
| $\lVert\mathbf{x_{sol}}\rVert_1$ | 5588239.4 | 7000565.5 | 1508020.8 | 1838194.5 | 193828.1 | 311446.9 |
| $\lVert\mathbf{x_*}\rVert_1$ | 5588239.3 | 7000565.2 | 1508021.0 | 1838194.7 | 193828.2 | 311447.1 |
| **CPU** | 14.5 | 14.7 | 3.3 | 3.4 | 0.3 | 0.4 |
| $\mathbf{N}_{\text{FAL}}$ | 14.9 | 15.0 | 14.4 | 15.0 | 14.0 | 14.0 |
| **nMat** | 59.8 | 60 | 58.8 | 60 | 72.2 | 92 |

TABLE 6.3
*FAL scaling results: $m = n/4$, $s = m/100$ and $\|x_{sol} - x_*\|_\infty \leq 10^{-1}$*

| | **n=512×512** | | **n=256×256** | | **n=64×64** | |
|---|---|---|---|---|---|---|
| | **Average** | **Max** | **Average** | **Max** | **Average** | **Max** |
| $\mathbf{N}_{\text{APG}}$ | 29.9 | 30 | 29.5 | 30 | 37.7 | 49 |
| $\lvert\lVert\mathbf{x_{sol}}\rVert_1 - \lVert\mathbf{x_*}\rVert_1\rvert/\lVert\mathbf{x_*}\rVert_1$ | 6.36E-08 | 9.54E-08 | 4.77E-08 | 6.85E-08 | 4.57E-08 | 1.66E-07 |
| $\max\{\lvert(\mathbf{x_{sol}})_i - (\mathbf{x_*})_i\rvert : (\mathbf{x_*})_i \neq \mathbf{0}\}$ | 7.66E-03 | 9.27E-03 | 6.92E-03 | 8.60E-03 | 5.63E-03 | 9.63E-03 |
| $\max\{\lvert(\mathbf{x_{sol}})_i\rvert : (\mathbf{x_*})_i = \mathbf{0}\}$ | 0.00E+00 | 0.00E+00 | 0.00E+00 | 0.00E+00 | 0.00E+00 | 0.00E+00 |
| $\lVert\mathbf{Ax_{sol}} - \mathbf{b}\rVert_2$ | 0.027 | 0.032 | 0.013 | 0.015 | 0.004 | 0.006 |
| $\lVert\mathbf{x_{sol}}\rVert_1$ | 5588239.7 | 7000565.6 | 1508021.0 | 1838194.8 | 193828.2 | 311447.1 |
| $\lVert\mathbf{x_*}\rVert_1$ | 5588239.3 | 7000565.2 | 1508021.0 | 1838194.7 | 193828.2 | 311447.1 |
| **CPU** | 15.0 | 15.2 | 3.4 | 3.5 | 0.3 | 0.4 |
| $\mathbf{N}_{\text{FAL}}$ | 15.0 | 15.0 | 15.0 | 15.0 | 14.7 | 16.0 |
| **nMat** | 61.8 | 62 | 61 | 62 | 77.4 | 100 |

TABLE 6.4
*FAL scaling results: $m = n/4$, $s = m/100$ and $\|x_{sol} - x_*\|_\infty \leq 10^{-2}$*

The worst case bound in Theorem 3.4 suggests that FAL requires $\mathcal{O}(\frac{1}{\epsilon})$ ALGORITHM APG iterations (or equivalently, matrix-vector multiplies) to compute an $\epsilon$-feasible and $\epsilon$-optimal solution to the basis pursuit problem. In our numerical experiments we found that we required only $4 \pm 1$ APG iterations per FAL iteration; therefore, we required only $\mathcal{O}(\log(\frac{1}{\epsilon}))$ APG iterations to compute an $\epsilon$-optimal solution. In order to clearly demonstrate this phenomenon, we created 5 random instances of $x_* \in \mathbb{R}^n$ and partial DCT matrix $A \in \mathbb{R}^{m\times n}$ such that $n = 64^2$ and $m = \frac{n}{4}$ as described in Section 6.1.1. Any $x_*$ created contains $\lceil \frac{m}{10} \rceil$ nonzero components such that the largest and smallest magnitude of those components are $10^5$ and 1, respectively.

| | **n=512×512** | | **n=256×256** | | **n=64×64** | |
|---|---|---|---|---|---|---|
| | **Average** | **Max** | **Average** | **Max** | **Average** | **Max** |
| $\mathbf{N}_{\text{APG}}$ | 28.9 | 29 | 28.2 | 29 | 27.2 | 28 |
| $\lvert\lVert\mathbf{x_{sol}}\rVert_1 - \lVert\mathbf{x_*}\rVert_1\rvert/\lVert\mathbf{x_*}\rVert_1$ | 6.82E-07 | 2.51E-06 | 2.01E-06 | 3.31E-06 | 4.93E-06 | 8.35E-06 |
| $\max\{\lvert(\mathbf{x_{sol}})_\mathbf{i} - (\mathbf{x_*})_\mathbf{i}\rvert : (\mathbf{x_*})_\mathbf{i} \neq \mathbf{0}\}$ | 6.46E-01 | 9.83E-01 | 8.26E-01 | 9.88E-01 | 7.92E-01 | 1.00E+00 |
| $\max\{\lvert(\mathbf{x_{sol}})_\mathbf{i}\rvert : (\mathbf{x_*})_\mathbf{i} = \mathbf{0}\}$ | 1.31E-01 | 2.24E-01 | 1.83E-01 | 4.07E-01 | 1.19E-01 | 2.12E-01 |
| $\lVert\mathbf{Ax_{sol}} - \mathbf{b}\rVert_2$ | 2.432 | 4.802 | 2.020 | 2.441 | 0.857 | 1.203 |
| $\lVert\mathbf{x_{sol}}\rVert_1$ | 56631758.9 | 59669790.2 | 14250619.6 | 15030777.3 | 1033569.1 | 1289376.0 |
| $\lVert\mathbf{x_*}\rVert_1$ | 56631797.7 | 59669841.3 | 14250648.3 | 15030813.1 | 1033574.2 | 1289377.9 |
| **CPU** | 14.5 | 14.6 | 3.3 | 3.4 | 0.2 | 0.6 |
| $\mathbf{N}_{\text{FAL}}$ | 14.9 | 15.0 | 14.2 | 15.0 | 13.8 | 14.0 |
| **nMat** | 59.8 | 60 | 58.4 | 60 | 56.4 | 58 |

TABLE 6.5
*FAL scaling results: $m = n/4$, $s = m/10$ and $\|x_{sol} - x_*\|_\infty \leq 1$*

| | **n=512×512** | | **n=256×256** | | **n=64×64** | |
|---|---|---|---|---|---|---|
| | **Average** | **Max** | **Average** | **Max** | **Average** | **Max** |
| $\mathbf{N}_{\text{APG}}$ | 33.7 | 35 | 32.7 | 34 | 31.2 | 33 |
| $\lvert\lVert\mathbf{x_{sol}}\rVert_1 - \lVert\mathbf{x_*}\rVert_1\rvert/\lVert\mathbf{x_*}\rVert_1$ | 5.30E-07 | 7.38E-07 | 6.70E-07 | 1.03E-06 | 4.56E-07 | 8.46E-07 |
| $\max\{\lvert(\mathbf{x_{sol}})_\mathbf{i} - (\mathbf{x_*})_\mathbf{i}\rvert : (\mathbf{x_*})_\mathbf{i} \neq \mathbf{0}\}$ | 9.21E-02 | 9.98E-02 | 8.96E-02 | 9.58E-02 | 7.06E-02 | 8.92E-02 |
| $\max\{\lvert(\mathbf{x_{sol}})_\mathbf{i}\rvert : (\mathbf{x_*})_\mathbf{i} = \mathbf{0}\}$ | 1.17E-03 | 6.17E-03 | 2.38E-03 | 1.26E-02 | 7.16E-03 | 2.40E-02 |
| $\lVert\mathbf{Ax_{sol}} - \mathbf{b}\rVert_2$ | 0.601 | 0.748 | 0.357 | 0.469 | 0.087 | 0.120 |
| $\lVert\mathbf{x_{sol}}\rVert_1$ | 56631827.7 | 59669880.2 | 14250657.8 | 15030821.1 | 1033574.7 | 1289378.1 |
| $\lVert\mathbf{x_*}\rVert_1$ | 56631797.7 | 59669841.3 | 14250648.3 | 15030813.1 | 1033574.2 | 1289377.9 |
| **CPU** | 16.8 | 17.6 | 3.8 | 4.0 | 0.2 | 0.7 |
| $\mathbf{N}_{\text{FAL}}$ | 17.1 | 18.0 | 16.7 | 17.0 | 15.7 | 16.0 |
| **nMat** | 69.4 | 72 | 67.4 | 70 | 64.4 | 68 |

TABLE 6.6
*FAL scaling results: $m = n/4$, $s = m/10$ and $\|x_{sol} - x_*\|_\infty \leq 10^{-1}$*

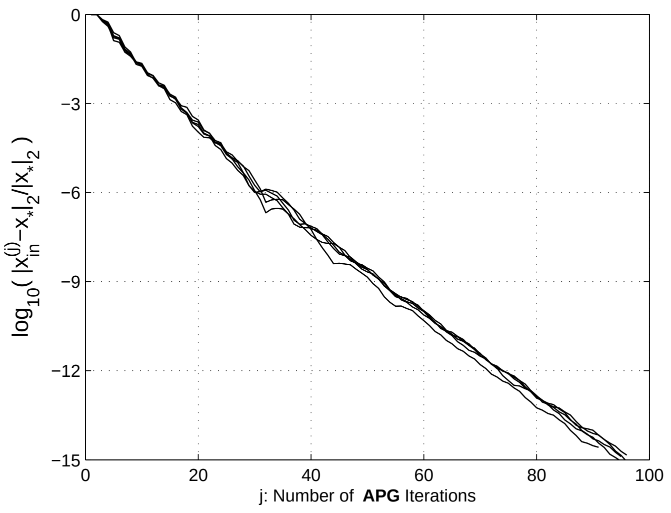
| | **n=512×512** | | **n=256×256** | | **n=64×64** | |
|---|---|---|---|---|---|---|
| | **Average** | **Max** | **Average** | **Max** | **Average** | **Max** |
| $\mathbf{N}_{\text{APG}}$ | 38.7 | 39 | 38.3 | 39 | 37.7 | 39 |
| $\lvert\lVert\mathbf{x_{sol}}\rVert_1 - \lVert\mathbf{x_*}\rVert_1\rvert/\lVert\mathbf{x_*}\rVert_1$ | 4.94E-08 | 5.80E-08 | 4.32E-08 | 5.54E-08 | 2.16E-08 | 5.06E-08 |
| $\max\{\lvert(\mathbf{x_{sol}})_\mathbf{i} - (\mathbf{x_*})_\mathbf{i}\rvert : (\mathbf{x_*})_\mathbf{i} \neq \mathbf{0}\}$ | 8.71E-03 | 9.96E-03 | 8.51E-03 | 9.88E-03 | 7.19E-03 | 9.41E-03 |
| $\max\{\lvert(\mathbf{x_{sol}})_\mathbf{i}\rvert : (\mathbf{x_*})_\mathbf{i} = \mathbf{0}\}$ | 1.58E-04 | 1.11E-03 | 0.00E+00 | 0.00E+00 | 0.00E+00 | 0.00E+00 |
| $\lVert\mathbf{Ax_{sol}} - \mathbf{b}\rVert_2$ | 0.069 | 0.084 | 0.038 | 0.042 | 0.010 | 0.011 |
| $\lVert\mathbf{x_{sol}}\rVert_1$ | 56631794.9 | 59669838.5 | 14250647.7 | 15030812.4 | 1033574.2 | 1289377.9 |
| $\lVert\mathbf{x_*}\rVert_1$ | 56631797.7 | 59669841.3 | 14250648.3 | 15030813.1 | 1033574.2 | 1289377.9 |
| **CPU** | 19.3 | 19.5 | 4.4 | 4.5 | 0.3 | 0.7 |
| $\mathbf{N}_{\text{FAL}}$ | 19.7 | 20.0 | 19.3 | 20.0 | 18.8 | 19.0 |
| **nMat** | 79.4 | 80 | 78.6 | 80 | 77.4 | 80 |

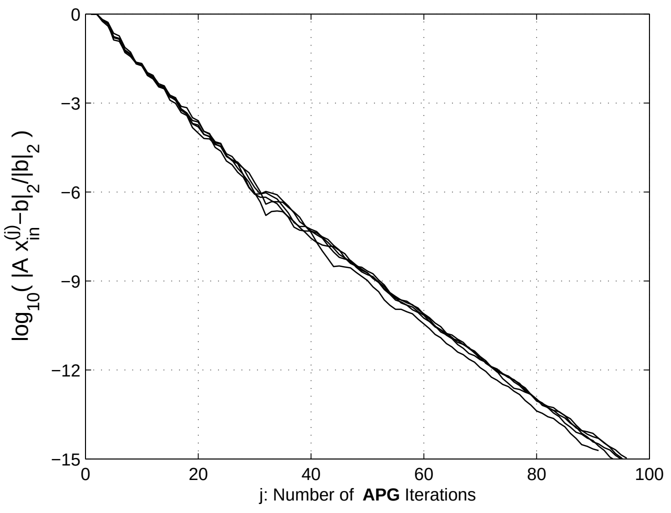
TABLE 6.7
*FAL scaling results: $m = n/4$, $s = m/10$ and $\|x_{sol} - x_*\|_\infty \leq 10^{-2}$*

We solved this set of random instances with ALGORITHM FAL using FALSTOP $= \{\|u^{(\ell)} - u^{(\ell-1)}\|_\infty \leq 5 \times 10^{-11}\}$. As before, let $\mathbf{N}_{\text{FAL}}$ denote the number of FAL iterations required to compute $x_{sol}$ satisfying the stopping condition FALSTOP. Let $\mathbf{N^{(k)}}$ be the number of ALGORITHM APG iterations done on the $k$-th call until the inner stopping condition (3.2) holds and $\mathbf{N}_{\text{APG}} = \sum_{k=1}^{\mathbf{N}_{\text{FAL}}} \mathbf{N^{(k)}}$ be the *total* number of inner iterations, i.e. *total* number of APG iterations, to compute $x_{sol}$.

For all five instances $\mathbf{N}_{\text{FAL}} \approx 45$, $\mathbf{N}_{\text{APG}} \approx 95$, $\max_{1\leq i\leq n}\{|(x_{sol})_i - (x_*)_i| : (x_*)_i \neq 0\} \approx 7 \times 10^{-11}$, $\max_{1\leq i\leq n}\{|(x_{sol})_i| : (x_*)_i = 0\} = 0$ and $\|Ax_{sol} - b\|_2 \approx 1 \times 10^{-10}$. These numbers show that each output $x_{sol}$ is 15 digits accurate and very close to feasibility.

Let $u^{(k,\ell)}$ denote $u^{(\ell)}$ iterate on the $k$-th APG call. For any $1 \leq k \leq \mathbf{N}_{\text{FAL}}$ and $1 \leq \ell \leq \mathbf{N^{(k)}}$, define $x_{\mathbf{in}}^{(\sum_{i=1}^{k-1}\mathbf{N^{(i)}}+\ell)} := u^{(k,\ell)}$. In Figure 6.1, we plot the relative error, relative feasibility and relative optimality of the inner iterates $x_{\mathbf{in}}^{(j)}$ as functions of ALGORITHM APG cumulative iteration counter $j \in \{1, ..., \mathbf{N}_{\text{APG}}\}$. From the plots in Figure 6.1, it is clear that, in practice, the complexity of computing an $\epsilon$-feasible, $\epsilon$-optimal iterate is $\mathcal{O}(\log(1/\epsilon))$, as opposed to the $\mathcal{O}(1/\epsilon)$ worst case complexity bound established Theorem 3.4.





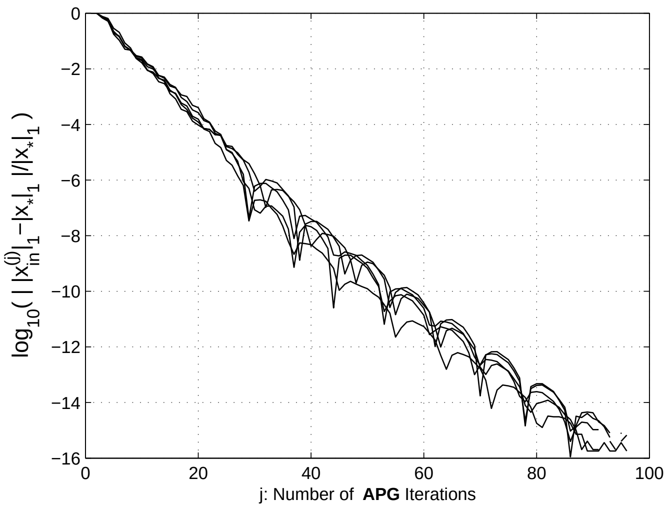


FIG. 6.1. *Relative solution error, feasibility and optimality vs APG iterations*

**6.1.3. Comparison with other solvers.** In this section, we report the results of our numerical experiments comparing FAL with SPA [2], NESTA v1.1 [4] [`http://www.acm.caltech.edu/~nesta/`], FPC and FPC-BB from FPC v2.0 [16, 17] [`http://www.caam.rice.edu/~optimization/L1/fpc/`], FPC-AS v1.21 [23] [`http://www.caam.rice.edu/~optimization/L1/FPC_AS/`], YALL1 v1.4 [24] [`http://www.yall1.blogs.rice.edu`] and SPGL1 v1.7 [22] [`http://www.cs.ubc.ca/labs/scl/spgl1/`]. We set the parameter values for each of the six solvers so that they all produce a solution with $\ell_\infty$-error approximately equal to $5 \times 10^{-4}$, i.e. $\|x_{sol} - x_*\|_\infty \approx 5 \times 10^{-4}$. This criterion results in the following set of parameters (all other parameters not mentioned below are set to their default values).

(a) **FAL**: We set $\gamma = 2.5 \times 10^{-4}$, and the initial update coefficients $c_\lambda^{(1)} = 0.4$, $c_\tau^{(1)} = 0.4$ and $t^{(1)} = 2$. For $k \geq 2$, we used the functions $c_\lambda(\cdot)$ and $t(\cdot)$ described in (5.4) and (5.6), respectively. The parameters $(\lambda^{(k)}, \tau^{(k)}, \epsilon^{(k)})$ were set as described in Section 5.3.
(b) **SPA**: $\gamma = 5 \times 10^{-5}$, $c_\tau^{(0)} = 0.2$, $c_\tau^{(1)} = 0.855$, $c_\epsilon = 0.8$, $c_\lambda = 0.9$ and $c_\mu = c_\nu = 0.1$. For details on these parameters refer to [2].
(c) **NESTA**: $\mu = 1 \times 10^{-4}$ and $\gamma = 1 \times 10^{-10}$. NESTA solves $\min_{\|Ax-b\|_2 \leq \delta} p_\mu(x)$, where $p_\mu(x) = \max\{x^T u - \frac{\mu}{2}\|u\|_2^2 : \|u\|_\infty \leq 1\}$. NESTA terminates when $\frac{|p_\mu(x^{(k)}) - \bar{p}_\mu(x^{(k)})|}{\bar{p}_\mu(x^{(k)})} < \gamma$, for some $\gamma > 0$, where $\bar{p}_\mu(x^{(k)}) = \frac{1}{min\{10,k\}} \sum_{\ell=1}^{min\{10,k\}} p_\mu(x^{(k-\ell)})$.
(d) **FPC** and **FPC-BB**: $\frac{1}{\lambda} = 1.5 \times 10^4$. FPC and FPC-BB solve $\min_{x \in \Re^n} \|x\|_1 + \frac{1}{\lambda}\|Ax - b\|_2^2$.
(e) **FPC-AS**: $\lambda = 7.5 \times 10^{-5}$. FPC-AS solves $\min_{x \in \Re^n} \lambda\|x\|_1 + \frac{1}{2}\|Ax - b\|_2^2$.
(f) **YALL1 (BP)**: $\gamma = 2 \times 10^{-9}$ and *nonorth* $= 0$. YALL1 (BP) algorithm solves the basis pursuit problem $\min_{x \in \Re^n}\{\|x\|_1 : \ Ax = b\}$ and terminates when $\frac{\|x_{k+1} - x_k\|_2}{\|x_{k+1}\|_2} \leq \gamma$. *nonorth* $= 0$ indicates that $AA^T = I$.
(g) **SPGL1 (BP)**: *optTol* $= 5 \times 10^{-3}$ and *bpTol* $= 1 \times 10^{-6}$. SPGL1 (BP) algorithm solves the basis pursuit problem $\min_{x \in \Re^n}\{\|x\|_1 : \ Ax = b\}$. For the optimality and basis pursuit tolerance parameters, *optTol* and *bpTol*, refer to [22].

The termination criteria for the different solvers were not directly comparable since the different solvers solve slightly different formulations of the basis pursuit problem. However, we attempted to set the stopping parameter $\gamma$ for FAL so that on average the stopping criterion for FAL was more stringent than any of the other solvers.

We tested each solver on the same set of 10 random instances of size $n = 512 \times 512$ that were generated using the procedure described in Section 6.1.1. The results of the experiments are displayed in Table 6.8. The experimental results in Table 6.8, show that FAL was *six* times faster than SPA and NESTA, approximately *four* times faster than FPC, and *two* times faster than FPC-BB and FPC-AS algorithms. Moreover, unlike the other solvers, for all 10 instances, FAL accurately identified the support of the target signal, without any heuristic thresholding step. This feature of FAL is very appealing in practice. For signals with a large dynamic range, almost all of the state-of-the-art efficient algorithms produce a solution with many small non zeros terms, and it is often hard to determine this threshold.

### 6.2. Experiments with measurement noise.

**6.2.1. Signal generation.** For this set of experiments the target signal $x_* \in \Re^n$ was generated as follows: $(x_*)_i = \mathbf{1}(i \in \Lambda)\ \Theta_i$, where

(i) the set $\Lambda$ was constructed by randomly selecting $s$ indices from the set $\{1, \ldots, n\}$,
(ii) $\Theta_i$, $i \in \Lambda$, were independently, and identically distributed standard Gaussian random variables.

The measurement matrix $A$ and the measurement vector $b$ were constructed as follows. We set the number of observations $m = \lceil \frac{n}{4} \rceil$. Each element $A_{ij}$ were sampled IID from a standard Normal distribution. The measurement $b = Ax_* + \zeta$, where each component $\zeta_i \in \mathbb{R}^m$ was sampled IID from a mean 0 and variance $\varrho^2$ Normal distribution. Therefore, the signal to noise ratio (SNR) of the measurement $b$ was

$$\mathrm{SNR}(b) = 10 \log_{10}\left(\frac{\mathbb{E}[\|Ax_*\|_2^2]}{\mathbb{E}[\|\zeta\|_2^2]}\right) = 10 \log_{10}\left(\frac{s}{\varrho^2}\right), \tag{6.3}$$

or equivalently, $\varrho^2 = s10^{-\mathrm{SNR}(b)/10}$. For each random $x_*$ and $A$, we considered SNR equal to 20dB, 30dB and 40dB.

| | **FAL** | | **FPC-AS** | |
|---|---|---|---|---|
| | **Average** | **Max** | **Average** | **Max** |
| $\lvert\lVert\mathbf{x_{sol}}\rVert_\mathbf{1} - \lVert\mathbf{x_*}\rVert_\mathbf{1}\rvert/\lVert\mathbf{x_*}\rVert_\mathbf{1}$ | 2.6E-09 | 3.2E-09 | 3.5E-08 | 3.6E-08 |
| $\max\{\lvert(\mathbf{x_{sol}})_\mathbf{i} - (\mathbf{x_*})_\mathbf{i}\rvert : (\mathbf{x_*})_\mathbf{i} \neq \mathbf{0}\}$ | 5.1E-04 | 6.2E-04 | 6.5E-04 | 7.1E-04 |
| $\max\{\lvert(\mathbf{x_{sol}})_\mathbf{i}\rvert : (\mathbf{x_*})_\mathbf{i} = \mathbf{0}\}$ | 0 | 0 | 1.2E-04 | 1.5E-04 |
| $\lVert\mathbf{Ax_{sol}} - \mathbf{b}\rVert_\mathbf{2}$ | 3.7E-03 | 4.4E-03 | 1.2E-02 | 1.2E-02 |
| $\lVert\mathbf{x_{sol}}\rVert_\mathbf{1}$ | 56631797.8 | 59669841.4 | 56631795.7 | 59669839.3 |
| $\lVert\mathbf{x_*}\rVert_\mathbf{1}$ | 56631797.7 | 59669841.3 | 56631797.7 | 59669841.3 |
| **CPU** | 11.0 | 12.3 | 22.2 | 23.9 |
| **nMat** | 98 | 99 | 109 / 205.6 | 109 / 208 |

| | **SPA** | | **NESTA** | |
|---|---|---|---|---|
| | **Average** | **Max** | **Average** | **Max** |
| $\lvert\lVert\mathbf{x_{sol}}\rVert_\mathbf{1} - \lVert\mathbf{x_*}\rVert_\mathbf{1}\rvert/\lVert\mathbf{x_*}\rVert_\mathbf{1}$ | 1.0E-08 | 1.1E-08 | 6.5E-08 | 6.7E-08 |
| $\max\{\lvert(\mathbf{x_{sol}})_\mathbf{i} - (\mathbf{x_*})_\mathbf{i}\rvert : (\mathbf{x_*})_\mathbf{i} \neq \mathbf{0}\}$ | 6.0E-04 | 6.8E-04 | 7.4E-04 | 8.4E-04 |
| $\max\{\lvert(\mathbf{x_{sol}})_\mathbf{i}\rvert : (\mathbf{x_*})_\mathbf{i} = \mathbf{0}\}$ | 6.6E-05 | 7.1E-05 | 2.3E-04 | 3.1E-04 |
| $\lVert\mathbf{Ax_{sol}} - \mathbf{b}\rVert_\mathbf{2}$ | 6.0E-03 | 6.3E-03 | 4.0E-10 | 4.1E-10 |
| $\lVert\mathbf{x_{sol}}\rVert_\mathbf{1}$ | 56631798.3 | 59669841.9 | 56631801.4 | 59669845.0 |
| $\lVert\mathbf{x_*}\rVert_\mathbf{1}$ | 56631797.7 | 59669841.3 | 56631797.7 | 59669841.3 |
| **CPU** | 67.3 | 73.0 | 72.1 | 80.1 |
| **nMat** | 583.2 | 587 | 632.4 | 636 |

| | **FPC** | | **FPC-BB** | |
|---|---|---|---|---|
| | **Average** | **Max** | **Average** | **Max** |
| $\lvert\lVert\mathbf{x_{sol}}\rVert_\mathbf{1} - \lVert\mathbf{x_*}\rVert_\mathbf{1}\rvert/\lVert\mathbf{x_*}\rVert_\mathbf{1}$ | 3.5E-08 | 3.5E-08 | 3.2E-08 | 3.3E-08 |
| $\max\{\lvert(\mathbf{x_{sol}})_\mathbf{i} - (\mathbf{x_*})_\mathbf{i}\rvert : (\mathbf{x_*})_\mathbf{i} \neq \mathbf{0}\}$ | 6.8E-04 | 7.3E-04 | 6.1E-04 | 6.7E-04 |
| $\max\{\lvert(\mathbf{x_{sol}})_\mathbf{i}\rvert : (\mathbf{x_*})_\mathbf{i} = \mathbf{0}\}$ | 1.6E-04 | 1.9E-04 | 1.3E-04 | 1.6E-04 |
| $\lVert\mathbf{Ax_{sol}} - \mathbf{b}\rVert_\mathbf{2}$ | 1.2E-02 | 1.2E-02 | 1.1E-02 | 1.1E-02 |
| $\lVert\mathbf{x_{sol}}\rVert_\mathbf{1}$ | 56631795.7 | 59669839.3 | 56631795.9 | 59669839.5 |
| $\lVert\mathbf{x_*}\rVert_\mathbf{1}$ | 56631797.7 | 59669841.3 | 56631797.7 | 59669841.3 |
| **CPU** | 40.4 | 50.0 | 22.7 | 26.4 |
| **nMat** | 383.0 | 387 | 195.0 | 195 |

| | **YALL1 (BP)** | | **SPGL1** | |
|---|---|---|---|---|
| | **Average** | **Max** | **Average** | **Max** |
| $\lvert\lVert\mathbf{x_{sol}}\rVert_\mathbf{1} - \lVert\mathbf{x_*}\rVert_\mathbf{1}\rvert/\lVert\mathbf{x_*}\rVert_\mathbf{1}$ | 9.4E-10 | 1.4E-09 | 3.2E-09 | 6.7E-09 |
| $\max\{\lvert(\mathbf{x_{sol}})_\mathbf{i} - (\mathbf{x_*})_\mathbf{i}\rvert : (\mathbf{x_*})_\mathbf{i} \neq \mathbf{0}\}$ | 5.7E-04 | 8.0E-04 | 5.3E-04 | 7.6E-04 |
| $\max\{\lvert(\mathbf{x_{sol}})_\mathbf{i}\rvert : (\mathbf{x_*})_\mathbf{i} = \mathbf{0}\}$ | 1.5E-19 | 1.5E-19 | 2.4E-04 | 3.3E-04 |
| $\lVert\mathbf{Ax_{sol}} - \mathbf{b}\rVert_\mathbf{2}$ | 4.4E-03 | 5.5E-03 | 4.2E-03 | 4.9E-03 |
| $\lVert\mathbf{x_{sol}}\rVert_\mathbf{1}$ | 56631797.7 | 59669841.3 | 56631797.5 | 59669841.1 |
| $\lVert\mathbf{x_*}\rVert_\mathbf{1}$ | 56631797.7 | 59669841.3 | 56631797.7 | 59669841.3 |
| **CPU** | 44.9 | 53.3 | 24.7 | 28.4 |
| **nMat** | 453.0 | 477 | 200.7 | 209 |

TABLE 6.8
*Noiseless comparison tests: $m = n/4$, $s = m/10$ and $\|x_{sol} - x_*\|_\infty \approx 5 \times 10^{-4}$*

**6.2.2. Comparison with other solvers.** For each noise level, we created 10 random instances of size $n = 128 \times 128$ using the procedure described in Section 6.2.1. We stopped each algorithm when the relative $\ell_2$-distance of consecutive iterates are less than $\varrho$, i.e. we impose the noisy stopping condition in Section 5.2 for all the solvers.

Some of the solvers we tested solve the penalty formulation $\min_{x\in\Re^n} \|x\|_1 + \frac{1}{\lambda}\|Ax - b\|_2^2$. Hale et. al. [16] proposed that when the measurement noise vector $\zeta$ is a $N(0, \sigma)$ Gaussian vector, the penalty parameter $\lambda$ should be set to $\lambda = \frac{\varrho\ \sigma_{\min}(A)}{\sigma^2_{\max}(A)}\sqrt{\frac{\chi^2_{1-\alpha,m}}{n}}$, where $\chi^2_{1-\alpha,m}$ denotes the $1-\alpha$ critical value of the $\chi^2$ distribution with $m$ degrees of freedom. We used the function `getM_mu.m` from FPC v.2.0 package to compute $\lambda$ according to this formula. The other parameters were set as follows.

1. **FAL**: We set the initial update coefficients $c_\lambda^{(1)} = 0.4$, $c_\tau^{(1)} = 0.4$ and $t^{(1)} = 2$. For $k \geq 2$, we used the functions $c_\lambda(\cdot)$ and $t(\cdot)$ described in (5.4) and (5.6), respectively. The parameters $(\lambda^{(k)}, \tau^{(k)}, \epsilon^{(k)})$ were set as described in Section 5.3.
2. **SPA**: $c_\tau^{(0)} = 0.2$, $c_\tau^{(1)} = 0.855$, $c_\epsilon = 0.8$, $c_\lambda = 0.9$ and $c_\mu = c_\nu = 0.1$. See [2] the parameter definitions.
3. **NESTA**: NESTA solves $\min_{\|Ax-b\|_2\leq\delta} p_\mu(x)$, where $p_\mu(x) = \max\{x^T u - \frac{\mu}{2}\|u\|_2^2 : \|u\|_\infty \leq 1\}$. $\mu = 2\times10^{-3}$ and the model parameter $\delta$ was set to $\sqrt{m + 2\sqrt{2m}}\ \varrho$ as described in [4].
4. **FPC** and **FPC-BB**: FPC and FPC-BB solve $\min_{x\in\Re^n} \|x\|_1 + \frac{1}{2\lambda}\|Ax - b\|_2^2$; $\lambda$ was set as described above.

5. **FPC-AS**: FPC-AS solves $\min_{x\in\Re^n} \lambda\|x\|_1 + \frac{1}{2}\|Ax-b\|_2^2$ and $\lambda$ was set as described.
6. **YALL1 (L1/L2)**: (L1/L2) option solves $\min_{x\in\Re^n} \|x\|_1 + \frac{1}{2\lambda}\|Ax-b\|_2^2$ and $\lambda$ was set as described above.
7. **YALL1 (L1/L2con)**: (L1/L2con) option solves $\min_{\|Ax-b\|_2\leq\delta} \|x\|_1$, where the model parameter $\delta$ was set to $\sqrt{m+2\sqrt{2m}}\ \varrho$.

All the parameters other than ones explained above were set to their default values. The results of the experiments are displayed in Tables 6.9 – 6.11.

| | **FAL** | | **FPC-AS** | | **FPC** | | **YALL1 (L1/L2)** | |
|---|---|---|---|---|---|---|---|---|
| | **Average** | **Max** | **Average** | **Max** | **Average** | **Max** | **Average** | **Max** |
| $\|\mathbf{x_{sol}}-\mathbf{x_*}\|_2/\|\mathbf{x_*}\|_2$ | 0.007 | 0.008 | 0.007 | 0.008 | 0.012 | 0.013 | 0.008 | 0.009 |
| $\max\{\lvert(\mathbf{x_{sol}})_i-(\mathbf{x_*})_i\rvert : (\mathbf{x_*})_i\neq\mathbf{0}\}$ | 1.4E-02 | 1.7E-02 | 2.2E-02 | 2.9E-02 | 2.4E-02 | 2.6E-02 | 1.7E-02 | 2.0E-02 |
| $\max\{\lvert(\mathbf{x_{sol}})_i\rvert : (\mathbf{x_*})_i=\mathbf{0}\}$ | 1.1E-02 | 1.3E-02 | 9.0E-03 | 1.1E-02 | 1.4E-02 | 1.6E-02 | 1.4E-02 | 1.6E-02 |
| $\|\mathbf{Ax_{sol}}-\mathbf{b}\|_2$ | 4.1E-02 | 4.7E-02 | 4.4E-02 | 5.0E-02 | 2.5E-02 | 2.5E-02 | 5.1E-02 | 5.3E-02 |
| **CPU** | 13.1 | 13.8 | 18.9 | 20.3 | 156.9 | 166.9 | 31.8 | 34.1 |
| **nMat** | 62.8 | 65 | 67.8/113.8 | 71/117 | 735.4 | 769 | 149.5 | 157 |

| | **SPA** | | **NESTA** | | **FPC-BB** | | **YALL1 (L1/L2con)** | |
|---|---|---|---|---|---|---|---|---|
| | **Average** | **Max** | **Average** | **Max** | **Average** | **Max** | **Average** | **Max** |
| $\|\mathbf{x_{sol}}-\mathbf{x_*}\|_2/\|\mathbf{x_*}\|_2$ | 0.011 | 0.012 | 0.019 | 0.020 | 0.012 | 0.012 | 0.013 | 0.014 |
| $\max\{\lvert(\mathbf{x_{sol}})_i-(\mathbf{x_*})_i\rvert : (\mathbf{x_*})_i\neq\mathbf{0}\}$ | 2.3E-02 | 3.5E-02 | 4.1E-02 | 4.5E-02 | 2.3E-02 | 2.6E-02 | 2.2E-02 | 2.6E-02 |
| $\max\{\lvert(\mathbf{x_{sol}})_i\rvert : (\mathbf{x_*})_i=\mathbf{0}\}$ | 1.0E-02 | 1.4E-02 | 1.3E-02 | 1.5E-02 | 1.3E-02 | 1.6E-02 | 1.6E-02 | 1.8E-02 |
| $\|\mathbf{Ax_{sol}}-\mathbf{b}\|_2$ | 2.9E-02 | 6.8E-02 | 6.9E-02 | 6.9E-02 | 2.5E-02 | 2.6E-02 | 1.5E-02 | 1.6E-02 |
| **CPU** | 66.8 | 76.1 | 264.1 | 293.0 | 39.4 | 43.7 | 28.5 | 30.8 |
| **nMat** | 326.6 | 375 | 536.0 | 553 | 180.8 | 189 | 137.0 | 137 |
| **PreprocessTime** | N/A | N/A | 581.7 | 667.8 | N/A | N/A | N/A | N/A |

Table 6.9
*Noisy comparative tests: SNR(b)=40dB*

| | **FAL** | | **FPC-AS** | | **FPC** | | **YALL1 (L1/L2)** | |
|---|---|---|---|---|---|---|---|---|
| | **Average** | **Max** | **Average** | **Max** | **Average** | **Max** | **Average** | **Max** |
| $\|\mathbf{x_{sol}}-\mathbf{x_*}\|_2/\|\mathbf{x_*}\|_2$ | 0.024 | 0.027 | 0.023 | 0.027 | 0.036 | 0.038 | 0.031 | 0.033 |
| $\max\{\lvert(\mathbf{x_{sol}})_i-(\mathbf{x_*})_i\rvert : (\mathbf{x_*})_i\neq\mathbf{0}\}$ | 5.4E-02 | 6.1E-02 | 5.8E-02 | 7.4E-02 | 7.3E-02 | 8.0E-02 | 6.0E-02 | 6.9E-02 |
| $\max\{\lvert(\mathbf{x_{sol}})_i\rvert : (\mathbf{x_*})_i=\mathbf{0}\}$ | 3.7E-02 | 4.0E-02 | 3.8E-02 | 4.2E-02 | 4.1E-02 | 4.9E-02 | 4.1E-02 | 4.7E-02 |
| $\|\mathbf{Ax_{sol}}-\mathbf{b}\|_2$ | 1.2E-01 | 1.3E-01 | 1.0E-01 | 1.1E-01 | 7.8E-02 | 7.9E-02 | 1.1E-01 | 1.2E-01 |
| **CPU** | 10.8 | 11.0 | 19.8 | 22.2 | 90.1 | 101 | 21.7 | 22.3 |
| **nMat** | 51.8 | 53 | 72.4/118.4 | 79/125 | 436.8 | 493 | 106.0 | 107 |

| | **SPA** | | **NESTA** | | **FPC-BB** | | **YALL1 (L1/L2con)** | |
|---|---|---|---|---|---|---|---|---|
| | **Average** | **Max** | **Average** | **Max** | **Average** | **Max** | **Average** | **Max** |
| $\|\mathbf{x_{sol}}-\mathbf{x_*}\|_2/\|\mathbf{x_*}\|_2$ | 0.023 | 0.025 | 0.078 | 0.082 | 0.035 | 0.036 | 0.036 | 0.038 |
| $\max\{\lvert(\mathbf{x_{sol}})_i-(\mathbf{x_*})_i\rvert : (\mathbf{x_*})_i\neq\mathbf{0}\}$ | 5.1E-02 | 5.7E-02 | 1.7E-01 | 2.0E-01 | 6.9E-02 | 7.6E-02 | 6.0E-02 | 6.8E-02 |
| $\max\{\lvert(\mathbf{x_{sol}})_i\rvert : (\mathbf{x_*})_i=\mathbf{0}\}$ | 3.2E-02 | 3.9E-02 | 6.2E-02 | 7.3E-02 | 3.9E-02 | 4.6E-02 | 4.5E-02 | 5.3E-02 |
| $\|\mathbf{Ax_{sol}}-\mathbf{b}\|_2$ | 1.1E-01 | 1.1E-01 | 2.2E-01 | 2.2E-01 | 7.8E-02 | 7.9E-02 | 3.6E-02 | 3.8E-02 |
| **CPU** | 55.4 | 58.6 | 207.1 | 221.1 | 25.8 | 27.9 | 20.6 | 21.5 |
| **nMat** | 267.8 | 287 | 354.0 | 363 | 122.4 | 135 | 103.5 | 107 |
| **PreprocessTime** | N/A | N/A | 581.7 | 667.8 | N/A | N/A | N/A | N/A |

Table 6.10
*Noisy comparative tests: SNR(b)=30dB*

| | **FAL** | | **FPC-AS** | | **FPC** | | **YALL1 (L1/L2)** | |
|---|---|---|---|---|---|---|---|---|
| | **Average** | **Max** | **Average** | **Max** | **Average** | **Max** | **Average** | **Max** |
| $\|\mathbf{x_{sol}}-\mathbf{x_*}\|_2/\|\mathbf{x_*}\|_2$ | 0.090 | 0.103 | 0.099 | 0.105 | 0.104 | 0.111 | 0.100 | 0.107 |
| $\max\{\lvert(\mathbf{x_{sol}})_i-(\mathbf{x_*})_i\rvert : (\mathbf{x_*})_i\neq\mathbf{0}\}$ | 2.0E-01 | 2.4E-01 | 2.0E-01 | 2.3E-01 | 2.1E-01 | 2.4E-01 | 1.9E-01 | 2.2E-01 |
| $\max\{\lvert(\mathbf{x_{sol}})_i\rvert : (\mathbf{x_*})_i=\mathbf{0}\}$ | 1.3E-01 | 1.9E-01 | 1.2E-01 | 1.4E-01 | 1.2E-01 | 1.4E-01 | 1.3E-01 | 1.6E-01 |
| $\|\mathbf{Ax_{sol}}-\mathbf{b}\|_2$ | 3.4E-01 | 5.1E-01 | 2.8E-01 | 2.9E-01 | 2.5E-01 | 2.6E-01 | 2.9E-01 | 2.9E-01 |
| **CPU** | 8.3 | 8.6 | 22.6 | 23.9 | 71.2 | 74.5 | 13.8 | 14.1 |
| **nMat** | 39.6 | 41 | 78.8/124.8 | 83/129 | 349.6 | 365 | 67.0 | 67 |

| | **SPA** | | **NESTA** | | **FPC-BB** | | **YALL1 (L1/L2con)** | |
|---|---|---|---|---|---|---|---|---|
| | **Average** | **Max** | **Average** | **Max** | **Average** | **Max** | **Average** | **Max** |
| $\|\mathbf{x_{sol}}-\mathbf{x_*}\|_2/\|\mathbf{x_*}\|_2$ | 0.108 | 0.116 | 0.251 | 0.267 | 0.100 | 0.109 | 0.124 | 0.132 |
| $\max\{\lvert(\mathbf{x_{sol}})_i-(\mathbf{x_*})_i\rvert : (\mathbf{x_*})_i\neq\mathbf{0}\}$ | 2.0E-01 | 2.4E-01 | 5.4E-01 | 6.1E-01 | 2.0E-01 | 2.3E-01 | 2.1E-01 | 2.3E-01 |
| $\max\{\lvert(\mathbf{x_{sol}})_i\rvert : (\mathbf{x_*})_i=\mathbf{0}\}$ | 1.3E-01 | 1.5E-01 | 1.8E-01 | 2.1E-01 | 1.2E-01 | 1.4E-01 | 1.6E-01 | 1.8E-01 |
| $\|\mathbf{Ax_{sol}}-\mathbf{b}\|_2$ | 1.4E-01 | 1.4E-01 | 6.9E-01 | 6.9E-01 | 2.5E-01 | 2.5E-01 | 1.4E-01 | 1.6E-01 |
| **CPU** | 54.8 | 58.9 | 170.1 | 176.4 | 23.8 | 27.0 | 15.4 | 16.1 |
| **nMat** | 268.0 | 289 | 225.0 | 233 | 104.2 | 109 | 76.5 | 77 |
| **PreprocessTime** | N/A | N/A | 581.7 | 667.8 | N/A | N/A | N/A | N/A |

Table 6.11
*Noisy comparative tests: SNR(b)=20dB*

The results clearly show that FAL is faster then the other state-of-the-art algorithms over the SNR range 20dB–40dB. Since **nMat** only keeps tracks of matrix-vector multiplies with the full $m \times n$ measurement matrix, the CPU time **CPU** for some of the solvers is not completely determined by **nMat**. For instance, at the 40dB SNR level, FAL computed 62.8 and FPC-AS computed 67.8 matrix-vector multiplications on average; but the average CPU time for FAL was 13.1 secs, whereas it was 18.9 secs for FPC-AS. This difference in the CPU time is due to smaller size matrix-vector multiplies that FPC-AS computes during subspace optimization iterations. NESTA has the highest overhead cost: it needs SVD of $A = U\Sigma V^T$ at the beginning since Gaussian $A$ does not satisfy $AA^T = I$. The preprocess time reported for NESTA shows the time to compute the SVD of $A$. Moreover, on top of the reported number of matrix vector multiplications with $m \times n$ matrices, NESTA also computes 2 matrix vector multiplications with $m \times m$ matrices at each iteration, which is not reported.

**6.3. Comparison with other solvers on hard instances.** In order to demonstrate the robustness of FAL, we tested it on the Caltech test problems: **CaltechTest1**, **CaltechTest2**, **CaltechTest3** and **CaltechTest4** [5]. This is a set of small-sized hard instances of CS problems. The hardness of these instances is due to the very large dynamic range of the nonzero components (see Table 6.12). For example, the target signal $x_* \in \mathbb{R}^{512}$ in **CaltechTest1** has 33 nonzero components with magnitude of $10^5$ and 5 components with magnitude of 1, i.e. $x_*$ has a dynamic range of 100dB.

| **problem** | **n** | **m** | **s** | **(magnitude, # elements of this magnitude)** |
|---|---|---|---|---|
| **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 | $(10^4, 13),\ (1, 12),\ (10^{-2}, 1)$ |

Table 6.12
*Characteristics of The Problems in CaltechTest Problem Set*

The Caltech problems have measurement noise. However, the $\text{SNR}(b) = 20\log_{10}\left(\frac{\|Ax_*\|_2}{\|\zeta\|_2}\right)$ for **CaltechTest1**–**CaltechTest4** problems is 228dB, 265dB, 168dB and 261dB, respectively. Since the SNR values are very high, we solved this set of problems solve them via basis pursuit formulation (1.1) using FAL, SPA, NESTA, YALL1 and SPGL1; and via unconstrained basis pursuit denoising formulation (1.2) with small $\bar{\lambda} > 0$ values using FPC, FPC-BB and FPC-AS.

For FAL, SPA, NESTA v1.1, FPC and FPC-BB that come with v2.0 solver package, FPC-AS v1.21, YALL1 v1.4 and SPGL1 v1.7, we chose parameter values that produced a solution $x_{\text{sol}}$ with high accuracy in reasonable time.

1. **FAL**: We set $\gamma = 5 \times 10^{-9}$ and the initial update coefficients $c_\lambda^{(1)} = 0.8$, $c_\tau^{(1)} = 0.8$ and $t^{(1)} = 1.9$. For $k \geq 2$, we used the functions $c_\lambda(\cdot)$ and $t(\cdot)$ described in (5.4) and (5.6), respectively. The parameters $(\lambda^{(k)}, \tau^{(k)}, \epsilon^{(k)})$ were set as described in Section 5.3.
2. **SPA**: $\gamma = 1 \times 10^{-8}$, $c_\tau^{(0)} = 0.1$, $c_\tau^{(1)} = 0.76$, $c_\epsilon = 0.8$, $c_\lambda = 0.95$ and $c_\mu = c_\nu = 0.4$. For details on these parameters refer to [2].
3. **NESTA**: $\mu = 1 \times 10^{-6}$ and $\gamma = 1 \times 10^{-16}$. See Section 6.1.3 for the definition of $\mu$ and $\gamma$.
4. **FPC** and **FPC-BB**: $\frac{1}{\lambda} = 1 \times 10^{10}$, xtol $= 10^{-10}$, gtol $= 10^{-8}$ and mxitr $= 20000$, where xtol, gtol set the termination conditions on the relative change in iterates and gradient, respectively, and mxitr is the iteration limit allowed. See Section 6.1.3 for the definition of $\lambda$.
5. **FPC-AS**: $\lambda = 1 \times 10^{-10}$ and gtol $= 10^{-14}$, where gtol is the termination criterion on the maximum norm of sub-gradient. See Section 6.1.3 for the definition of $\lambda$.
6. **YALL1 (BP)**: $\gamma = 1 \times 10^{-11}$ and *nonorth* $= 0$. See the item describing YALL1 (BP) for the definition of of $\gamma$ and *nonorth* in Section 6.1.3.
7. **SPGL1 (BP)**: $optTol = 1 \times 10^{-7}$, $bpTol = 1 \times 10^{-9}$ and $decTol = 1 \times 10^{-7}$. For the details on optimality and basis pursuit tolerance parameters, $optTol$, $bpTol$ and $decTol$, refer to [22].

These parameter values were fixed for all 4 test problems and all other parameters are set to their default values. The termination criteria were not directly comparable since the different solvers use a slightly different formulation of the basis pursuit problem. However, we attempted to set the stopping parameter $\gamma$ such that

| **Problem** | **Solver** | $\|\mathbf{x}_*\|_1$ | $\|\mathbf{x}_{\mathbf{sol}}\|_1$ | rel.err | $inf.err_+$ | $inf.err_0$ | $\|\mathbf{r}\|_2$ | **CPU** | **nMat** | **nnz** |
|---|---|---|---|---|---|---|---|---|---|---|
| **Caltech1** | **FAL** | 3300005 | 3300005.00 | 5.15E-12 | 9.94E-07 | 0 | 1.16E-08 | 0.598 | 1715 | 38 |
| | **SPA** | | 3300005.00 | 1.85E-10 | 3.05E-05 | 1.68E-05 | 4.85E-06 | 7.783 | 20305 | 512 |
| | **NESTA** | | 3300005.00 | 2.43E-10 | 4.01E-05 | 2.18E-05 | 1.06E-10 | 9.902 | 18432 | 512 |
| | **FPC** | | 3300002.44 | 3.05E-06 | 5.17E-01 | 2.64E-01 | 1.78E-01 | 22.509 | 40001 | 109 |
| | **FPC-BB** | | 3300002.44 | 8.46E-06 | 5.16E-01 | 2.63E-01 | 1.78E-01 | 44.600 | 40001 | 109 |
| | **FPC-AS** | | 3300005.00 | 5.15E-12 | 9.97E-07 | 8.97E-10 | 1.62E-09 | 0.375 | 109 / 393 | 63 |
| | **YALL1** | | 3300005.00 | 5.61E-11 | 8.51E-05 | 1.19E-18 | 6.09E-05 | 5.486 | 14492 | 276 |
| | **SPGL1** | | 3300005.11 | 1.19E-06 | 1.20E+00 | 6.53E-01 | 9.94E-08 | 9.989 | 17705 | 171 |
| **Caltech2** | **FAL** | 3300005 | 3200005.00 | 7.17E-14 | 1.41E-08 | 0 | 7.03E-09 | 0.358 | 971 | 37 |
| | **SPA** | | 3200005.00 | 1.19E-10 | 2.04E-05 | 1.38E-05 | 4.73E-06 | 5.651 | 14001 | 512 |
| | **NESTA** | | 3200005.00 | 1.24E-10 | 2.10E-05 | 1.47E-05 | 9.34E-11 | 3.826 | 7204 | 512 |
| | **FPC** | | 3200004.97 | 2.15E-08 | 3.72E-03 | 2.32E-03 | 2.39E-03 | 23.540 | 40001 | 96 |
| | **FPC-BB** | | 3200004.47 | 3.43E-07 | 5.92E-02 | 3.70E-02 | 3.82E-02 | 43.019 | 40001 | 96 |
| | **FPC-AS** | | 3200005.00 | 7.58E-14 | 1.78E-08 | 2.03E-09 | 1.88E-09 | 0.222 | 127 / 407 | 63 |
| | **YALL1** | | 3200005.00 | 6.29E-11 | 1.01E-04 | 1.15E-18 | 5.76E-05 | 1.337 | 3137 | 275 |
| | **SPGL1** | | 3200005.00 | 1.35E-11 | 1.35E-05 | 8.15E-06 | 6.96E-08 | 16.413 | 28008 | 212 |
| **Caltech3** | **FAL** | 6.200000974 | 6.20000101 | 4.03E-08 | 1.49E-08 | 0 | 1.35E-08 | 0.166 | 359 | 32 |
| | **SPA** | | 6.19999388 | 5.82E-06 | 1.85E-06 | 8.99E-07 | 8.78E-07 | 4.663 | 9767 | 512 |
| | **NESTA** | | 6.20007451 | 5.02E-05 | 1.51E-05 | 8.72E-06 | 1.96E-16 | 5.131 | 8326 | 512 |
| | **FPC** | | 6.20000076 | 6.50E-08 | 2.01E-08 | 1.04E-08 | 1.80E-08 | 27.730 | 40001 | 78 |
| | **FPC-BB** | | 6.19975503 | 7.09E-05 | 2.22E-05 | 1.06E-05 | 1.84E-05 | 37.365 | 40001 | 80 |
| | **FPC-AS** | | 6.20000098 | 1.46E-09 | 3.78E-10 | 4.73E-10 | 1.23E-09 | 0.137 | 93 / 271 | 67 |
| | **YALL1** | | 6.30373200 | 8.53E-02 | 1.47E-01 | 1.19E-01 | 3.95E-16 | 23.048 | 50002 | 321 |
| | **SPGL1** | | 6.20000438 | 4.14E-06 | 6.62E-06 | 5.92E-06 | 9.99E-08 | 8.275 | 11885 | 131 |
| **Caltech4** | **FAL** | 130012.01 | 130012.010 | 1.28E-12 | 2.16E-08 | 0 | 1.24E-08 | 0.207 | 487 | 26 |
| | **SPA** | | 130012.010 | 3.80E-09 | 4.92E-05 | 1.86E-05 | 1.15E-05 | 3.788 | 8221 | 512 |
| | **NESTA** | | 130012.010 | 1.87E-09 | 2.37E-05 | 9.61E-06 | 5.71E-12 | 3.583 | 5904 | 512 |
| | **FPC** | | 130012.008 | 2.01E-08 | 2.62E-04 | 9.07E-05 | 1.39E-04 | 26.283 | 40001 | 71 |
| | **FPC-BB** | | 130010.234 | 1.92E-05 | 2.39E-01 | 7.46E-02 | 1.39E-01 | 44.804 | 40001 | 62 |
| | **FPC-AS** | | 130012.010 | 8.31E-13 | 9.01E-09 | 8.62E-09 | 3.86E-09 | 0.270 | 145 / 523 | 50 |
| | **YALL1** | | 130012.010 | 8.99E-11 | 5.34E-06 | 7.03E-20 | 3.64E-06 | 4.747 | 10682 | 305 |
| | **SPGL1** | | 130012.010 | 9.57E-11 | 4.42E-06 | 2.40E-06 | 9.97E-08 | 9.641 | 14647 | 97 |

Table 6.13
*CaltechTest Problem Set*

on the average the stopping criterion for FAL was more stringent than those for the other algorithms we tested. The results of the experiments are displayed in Table 6.13. In Table 6.13, the row labeled **CPU** lists the row labeled **rel.err** lists the relative error the solution, i.e $\mathbf{rel.err} = \frac{\|x_{sol}-x_*\|_2}{\|x_*\|_2}$, the row labeled $\mathbf{inf.err}_+$ lists the absolute error on the nonzero components of $x_*$, i.e $\mathbf{inf.err}_+ = \max\{|(x_{sol})_i - (x_*)_i| : (x_*)_i \neq 0\}$, the row labeled $\mathbf{inf.err_0}$ lists the absolute error on the zero components of $x_*$, without any thresholding or post-processing. None of the solvers, other than FAL, were able to identify the true support of the target signal for any of the **CaltechTest** instances.

## 7. Conclusion.

We propose a first-order augmented lagrangian algorithm (FAL) for basis pursuit. FAL computes a solution to the basis pursuit problem by solving a sequence of augmented lagrangian subproblems, and each subproblem is solved using a variant of the infinite memory proximal gradient algorithm (Algorithm 3) [21]. We prove that FAL iterates converge to the optimal solution of the basis pursuit problem whenever it is unique, which is true with overwhelming probability for compressed sensing problems (In [7] Candés and Tao have shown that for random measurement $A$ matrices the resulting basis pursuit problem has a unique solution with very high probability). We are able to prove FAL needs at most $\mathcal{O}(\epsilon^{-1})$ matrix-vector multiplies to compute an $\epsilon$-feasible and $\epsilon$-optimal solution. However, in our numerical experiments we observe that we only need $\mathcal{O}(\log(\epsilon^{-1}))$ matrix-vector multiplies! We found that for a fixed measurement ratio $m/n$, sparsity ratios $s/n$, and solution accuracy $\gamma$, the number of matrix-vector multiplies computed by FAL were effectively independent of the dimension $n$. This allows us to tune the algorithm parameters on small problems and then use these parameters for all larger problems with the same measurement and sparsity ratios. The numerical results reported in this paper clearly show that FAL solves both the noise-less and noisy versions of the compressive sensing problems very efficiently.

**8. Acknowledgments.** We thank the anonymous referees for their insightful comments that significantly improved both the algorithm and the paper. We also thank Professor Y. Zhang for helping us better understand the capabilities of YALL1.

## REFERENCES

[1] N. S. Aybat and G. Iyengar, *Unified approach for minimizing composite norms*, submitted to Mathematical Programming Journal, Series A, (2010).

[2] ———, *A first-order smoothed penalty method for compressed sensing*, SIAM Journal on Optimization, 21 (2011), pp. 287–313.

[3] A. Beck and M. Teboulle, *A fast iterative shrinkage-thresholding algorithm for linear inverse problems*, SIAM Journal on Imaging Sciences, 2 (2009), pp. 183–202.

[4] S. Becker, J. Bobin, and E. Candès, *Nesta: a fast and accurate first-order method for sparse recovery*, SIAM J. Imaging Sci., 4 (2011), pp. 1–39.

[5] E. Candès and S. Becker, *Some test problems for compressed sensing.* private communication, 2008.

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

[7] E. Candès, J. Romberg, and T. Tao, *Signal recovery from incomplete and inaccurate measurements*, Comm. Pure Appl. Math., 59 (2005), pp. 1207–1223.

[8] ———, *Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information*, IEEE Trans. Info. Th., 52 (2006).

[9] E. Candès and T. Tao, *Near optimal signal recovery from random projections: universal encoding strategies?*, IEEE Trans. Info. Th., 52 (2006), pp. 5406–5425.

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

[11] I. Daubechies, M. Fornasier, and I. Loris, *Accelerated projected gradient method for linear inverse problems with sparsity constraints*, Journal of Fourier Analysis and Applications, 14 (2008), pp. 764–792.

[12] D. Donoho, *Compressed sensing*, IEEE Trans. Info. Th., 52 (2006), pp. 1289–1306.

[13] J. Duchi, S. Shalew-Shwartz, Y. Singer, and T. Chandra, *Efficient projections onto the $\ell_1$-ball for learning in high dimensions*, in Proceedings, Twenty-Fifth International Conference on Machine Learning, Andrew McCallum and Sam Roweis, eds., Helsinki, Finland, 2008, pp. 272–279.

[14] A. Edelman, *Eigenvalues and condition numbers of random matrices*, SIAM Journal on Matrix Analysis and Applications, 9 (1988), pp. 543–560.

[15] M. A. Figueiredo, R. Nowak, and S. J. Wright, *Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems*, IEEE Journal of Selected Topics in Signal Processing, 1 (2007), pp. 586–597.

[16] E. T. Hale, W. Yin, and Y. Zhang, *A fixed-point continuation for $\ell 1$-regularized minimization with applications to compressed sensing*, tech. report, Rice University, 2007.

[17] ———, *Fized-point continuation for $\ell 1$-minimization: Methodology and convergence*, SIAM Journal on Optimization, 19 (2008), pp. 1107–1130.

[18] Seung-Jean Kim, K. Koh, M. Lustig, S. Boyd, and D. Gorinevsky, *An interior-point method for large-scale l1-regularized least squares*, Selected Topics in Signal Processing, IEEE Journal of, 1 (2007), pp. 606 –617.

[19] Y. Nesterov, *Introductory Lectures on Convex Optimization: A Basic Course*, Kluwer Academic Publishers, 2004.

[20] ———, *Smooth minimization of nonsmooth functions*, Mathematical Programming, 103 (2005), pp. 127–152.

[21] P. Tseng, *On accelerated proximal gradient methods for convex-concave optimization*, submitted to SIAM Journal on Optimization, (2008).

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

[23] Z. Wen, W. Yin, D. Goldfarb, and Y. Zhang, *A fast algorithm for sparse reconstruction based on shrinkage, subspace optimization and continuation*, To appear in SIAM Journal on Scientific Computing, (2009).

[24] J. Yang and Y. Zhang, *Alternating direction algorithms for l1-problems in compressive sensing*, Tech. Report TR09-37, CAAM, Rice University, 2009.

[25] W. Yin, S. Osher, D. Goldfarb, and J. Darbon, *Bregman iterative algorithms for $\ell_1$ minimization with applications to compressed sensing*, SIAM Journal on Imaging Sciences, 1 (2008), pp. 143–168.

## Appendix A. Auxiliary results.

THEOREM A.1. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a convex function. Suppose the $\nabla f$ is Lipschitz continuous with the Lipschitz constant $L$. Fix $\epsilon > 0$. Suppose $\bar{x} \in \mathbb{R}^n$ satisfies $\lambda\|\bar{x}\|_1 + f(\bar{x}) - (\lambda\|x^*\|_1 + f(x^*)) \leq \epsilon$, where $x^* \in \operatorname{argmin}\{\lambda\|x\|_1 + f(x) : x \in \mathbb{R}^n\}$. Then*

$$\frac{1}{2L}\sum_{i:|\nabla f_i(\bar{x})|>\lambda} (|\nabla f_i(\bar{x})| - \lambda)^2 \leq \epsilon. \tag{A.1}$$

*The bound* (A.1) *implies* $\|\nabla f(\bar{x})\|_\infty \leq \sqrt{2L\epsilon} + \lambda$.

*Proof.* Triangular inequality for $\|.\|_1$ and Lipschitz continuity of $\nabla f$ implies that for all $y \in \mathbb{R}^n$

$$\lambda\|y\|_1 + f(y) \leq \lambda\|\bar{x}\|_1 + f(\bar{x}) + \nabla f(\bar{x})^T(y - \bar{x}) + \frac{L}{2}\|y - \bar{x}\|_2^2 + \lambda\|y - \bar{x}\|_1.$$

Taking the minimum with respect to $y$, we get

$$\lambda\|x^*\|_1 + f(x^*) \leq \lambda\|\bar{x}\|_1 + f(\bar{x}) + \min_{y\in\mathbb{R}^n}\left\{\nabla f(\bar{x})^T(y - \bar{x}) + \frac{L}{2}\|y - \bar{x}\|_2^2 + \lambda\|y - \bar{x}\|_1\right\}. \tag{A.2}$$

Let $w \equiv \nabla f(\bar{x})$. Then

$$\begin{aligned}
y^* &= \operatorname*{argmin}_{y\in\mathbb{R}^n}\left\{w^T(y - \bar{x}) + \frac{L}{2}\|y - \bar{x}\|_2^2 + \lambda\|y - \bar{x}\|_1\right\}, && \text{(A.3)}\\
&= \operatorname*{argmin}_{y\in\mathbb{R}^n}\left\{\frac{1}{2}\|y - \bar{x} + \frac{w}{L}\|_2^2 + \frac{\lambda}{L}\|y - \bar{x}\|_1\right\}, && \text{(A.4)}\\
&= \bar{x} + \operatorname{sign}\left(\frac{-w}{L}\right) \odot \max\left\{\left|\frac{-w}{L}\right| - \frac{\lambda}{L}, 0\right\}, && \text{(A.5)}\\
&= \bar{x} + \frac{-\operatorname{sign}(w)}{L} \odot \max\{|w| - \lambda, 0\}, && \text{(A.6)}
\end{aligned}$$

where (A.5) follows from the fact that $\operatorname{argmin}_{z\in\mathbb{R}^n}\{\nu\|z\|_1 + \frac{1}{2}\|z - \zeta\|_2^2\} = \operatorname{sign}(\zeta) \odot \max\{|\zeta| - \nu, 0\}$, where $\odot$ is component-wise multiplication operator [17], and all other vector operators such as $|\cdot|$, $\operatorname{sign}(\cdot)$ and $\max\{\cdot,\cdot\}$ are defined to operate component-wise. Substituting $y^*$ in (A.2), we get

$$\begin{aligned}
&\min_{y\in\mathbb{R}^n}\left\{w^T(y - \bar{x}) + \frac{L}{2}\|y - \bar{x}\|_2^2 + \lambda\|y - \bar{x}\|_1\right\},\\
=&-\sum_i \frac{|w_i|}{L}\max\{|w_i| - \lambda, 0\} + \frac{1}{2L}\sum_i \max\{|w_i| - \lambda, 0\}^2 + \frac{\lambda}{L}\sum_i \max\{|w_i| - \lambda, 0\},\\
=&\frac{1}{L}\sum_{i:|w_i|>\lambda}\left(-|w_i| + \frac{1}{2}(|w_i| - \lambda) + \lambda\right)(|w_i| - \lambda),\\
=&-\frac{1}{2L}\sum_{i:|w_i|>\lambda}(|w_i| - \lambda)^2. && \text{(A.7)}
\end{aligned}$$

The bound (A.1) follows from the fact $\lambda\|\bar{x}\|_1 + f(\bar{x}) - (\lambda\|x^*\|_1 + f(x^*)) \leq \epsilon$. The bound (A.1) clearly implies that $|w_i| \leq \sqrt{2L\epsilon} + \lambda$ for all $i$, i.e. $\|\nabla f(\bar{x})\|_\infty \leq \sqrt{2L\epsilon} + \lambda$. □

COROLLARY A.2. *Suppose $A \in \mathbb{R}^{m\times n}$ with $m \leq n$ and full rank. Let $P(x) = \lambda\|x\|_1 + \frac{1}{2}\|Ax - b - \lambda\theta\|_2^2$. Suppose $\bar{x}$ is $\epsilon$-optimal for $\min_{x\in\mathbb{R}^n} P(x)$, i.e. $0 \leq P(\bar{x}) - \min_{x\in\mathbb{R}^n} P(x) \leq \epsilon$. Then*

$$\begin{aligned}
\|A^T(A\bar{x} - b - \lambda\theta)\|_\infty &\leq \sqrt{2\epsilon}\,\sigma_{max}(A) + \lambda,\\
\|A\bar{x} - b - \lambda\theta\|_2 &\leq \tfrac{\sqrt{n}}{\sigma_{min}(A)}\left(\sqrt{2\epsilon}\,\sigma_{max}(A) + \lambda\right),
\end{aligned} \tag{A.8}$$

*where $\sigma_{max}(A)$ denotes the maximum singular value of $A$.*

*Proof.* Let $f(x) = \frac{1}{2}\|Ax - b - \lambda\theta\|_2^2$, then $\nabla f(x) = A^T(Ax - b - \lambda\theta)$. For any $x, y \in \mathbb{R}^n$, we have

$$\|\nabla f(x) - \nabla f(y)\|_2 = \|A^T A(x-y)\|_2 \leq \sigma_{max}^2(A)\|x - y\|_2,$$

where $\sigma_{max}(A)$ is the maximum singular-value of $A$. Thus, $f : \Re^n \to \Re$ is a convex function and $\nabla f$ is Lipschitz continuous with the constant $L = \sigma_{max}^2(A)$.

Since $\bar{x}$ is an $\epsilon$-optimal solution to $\min_{x\in\mathbb{R}^n} P(x) = \min_{x\in\mathbb{R}^n}\{\lambda\|x\|_1 + f(x)\}$, Theorem A.1 immediately implies the first bound in (A.8). The second bound follows from the fact that

$$\|A\bar{x} - b - \lambda\theta\|_2 \leq \frac{\|A^T(A\bar{x} - b - \lambda\theta)\|_2}{\sigma_{min}(A)} \leq \frac{\sqrt{n}}{\sigma_{min}(A)}\|A^T(A\bar{x} - b - \lambda\theta)\|_\infty,$$

where the first inequality follows the definition of $\sigma_{\min}(A)$ and the second from the bound $\|y\|_2 \leq \sqrt{n}\|y\|_\infty$ for all $y \in \mathbb{R}^n$. ☐

LEMMA A.3. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a strictly convex function and $S \subset \mathbb{R}^n$ be a closed, convex set. Let $x_S^* = \operatorname{argmin}_{x\in S} f(x)$ and $x^* = \operatorname{argmin}_{x\in\mathbb{R}^n} f(x)$. Suppose the unconstrained optimum $x^* \notin S$, then $x_S^* \in \partial S$, where $\partial S$ denotes the boundary of the set $S$.*

*Proof.* We will establish the result by contradiction. Suppose $x_S^* \in \textbf{int}(S)$. Then, there exists an $\epsilon > 0$ such that $B_\epsilon = \{x \in \mathbb{R}^n \ : \ \|x - x_S^*\|_2 < \epsilon\} \subset S$. Since $f$ is strictly convex and $x^* \neq x_S^*$, $f(x^*) < f(x_S^*)$.

Fix $0 < \lambda < \frac{\epsilon}{\|\bar{x} - x^*\|_2} < 1$. Then $x_\lambda = \lambda x^* + (1-\lambda)x_S^* \in B_\epsilon \subset S$. Since $f$ is strictly convex,

$$f(x_\lambda) < \lambda f(x^*) + (1-\lambda)f(\bar{x}) < f(x_S^*). \tag{A.9}$$

This contradicts the fact that $x_S^* = \operatorname{argmin}_{x\in S}\{f(x)\}$. Thus, $x_S^* \in S \setminus \textbf{int}(S) = \partial S$. ☐

LEMMA A.4. *Fix $y \in \mathbb{R}^n$, $\lambda > 0$ and $\eta > 0$. Let $P(x) = \lambda\|x\|_1 + \frac{1}{2}\|x - y\|_2^2$ and*

$$x^* = \operatorname{argmin}\{P(x) : \|x\|_1 \leq \eta\}. \tag{A.10}$$

*Then the deterministic complexity of computing $x^*$ is $\mathcal{O}(n\log(n))$, and the randomized complexity is $\mathcal{O}(n)$.*

*Proof.* Since $P(x)$ is strongly convex, (A.10) has a unique primal optimal solution. Also, since the optimization problem (A.10) satisfies Slater's constraint qualification, strong duality holds, and since the primal optimal value bounded, the dual optimal value is attained.

Let

$$\mathcal{L}(x, \alpha) = \lambda\|x\|_1 + \frac{1}{2}\|x - y\|_2^2 + \alpha(\|x\|_1 - \eta), \tag{A.11}$$

$$= (\lambda + \alpha)\|x\|_1 + \frac{1}{2}\|x - y\|_2^2 - \alpha\eta, \tag{A.12}$$

denote the Lagrangian function. Since strong duality holds, $x^*$ is a minimizer of $\mathcal{L}(x, \alpha^*)$, where $\alpha^*$ denote the optimal dual solution. Since $\mathcal{L}(x, \alpha^*)$ is a strictly convex function of $x$, $x^*$ is the unique minimizer of $\mathcal{L}(x, \alpha^*)$. Let

$$x^*(\alpha) = \operatorname*{argmin}_{x\in\mathbb{R}^n} \mathcal{L}(x, \alpha) = \operatorname{sign}(y) \odot \max\{|y| - (\lambda + \alpha), 0\}. \tag{A.13}$$

It is clear that $x^* = x^*(\alpha^*)$. In the rest of this proof, we show how to efficiently compute $\alpha^*$.

Note that $x^*(0) = \operatorname{sign}(y) \odot \max\{|y| - \lambda, 0\}$ is the unique unconstrained minimizer of $P(x)$. When $\|x^*(0)\|_1 \leq \eta$, then trivially $x^* = x^*(0)$. However, when $\|x^*(0)\|_1 > \eta$, Lemma A.3 implies that $x^* \in \partial\{x \in \mathbb{R}^n \ : \ \|x\|_1 \leq \eta\}$, i.e. $\|x^*\|_1 = \eta$. Therefore,

$$\alpha^* \in \{\alpha > 0 : \ \|x^*(\alpha)\|_1 = \eta\}. \tag{A.14}$$

From (A.13), it follows that

$$\|x^*(\alpha)\|_1 = \sum_{i:|y_i|-\lambda \geq \alpha} ((|y_i| - \lambda) - \alpha) = \sum_{i=1}^{n} (|x^*(0)| - \alpha)^+. \tag{A.15}$$

Note that $\|x^*(\alpha)\|_1$ is a strictly decreasing continuous function of $\alpha$. Since $\|x^*(0)\|_1 > \eta$, there exists a unique $\hat{\alpha} > 0$ such that $\|x^*(\hat{\alpha})\|_1 = \eta$. From (A.14), we can conclude that $\alpha^* = \hat{\alpha}$.

To compute $\hat{\alpha}$ such that $\|x^*(\hat{\alpha})\|_1 = \eta$, sort $z = |x^*(0)|$ in decreasing order. Let $z_{[i]}$ denote the $i$-th largest component of $z$. It is clear that $\|x^*(w_{[n]})\|_1 > \eta > 0$ and $\|x^*(\alpha)\|_1 = 0$ for all $\alpha > w_{[1]}$. Hence, there exists an index $1 \leq k < n$ such that $\|x^*(w_{[k]})\|_1 \leq \eta$ and $\|x^*(w_{[k+1]})\|_1 > \eta$, and it follows that

$$\alpha^* = \frac{1}{k} \left( \sum_{j=1}^{k} w_{[j]} - \eta \right). \tag{A.16}$$

Thus, $x^* = x^*(\alpha^*)$ can be computed in $O(n \log(n))$ operations. Singer et al [13] show that $\alpha^*$ with a $\mathcal{O}(n)$ randomized complexity using a slightly modified version of the randomized median finding algorithm. $\square$