A FIRST-ORDER SMOOTHED PENALTY METHOD FOR COMPRESSED SENSING

N. S. AYBAT* AND G. IYENGAR†

Abstract. We propose a first-order smoothed penalty algorithm (SPA) to solve the sparse recovery problem $\min\{\|x\|_1 : Ax = b\}$. SPA is efficient as long as the matrix-vector product Ax and A^Ty can be computed efficiently; in particular, A need not have orthogonal rows. SPA converges to the target signal by solving a sequence of penalized optimization sub-problems, and each sub-problem is solved using Nesterov's optimal algorithm for simple sets [17, 18]. We show that the SPA iterates x_k are ϵ-feasible, i.e. $\|Ax_k - b\|_2 \leq \epsilon$ and ϵ-optimal, i.e. $|\,\|x_k\|_1 - \|x^*\|_1| \leq \epsilon$ after $\tilde{\mathcal{O}}(\epsilon^{-\frac{3}{2}})$ iterations. SPA is able to work with ℓ_1, ℓ_2 or ℓ_∞ penalty on the infeasibility, and SPA can be easily extended to solve the relaxed recovery problem $\min\{\|x\|_1 : \|Ax - b\|_2 \leq \delta\}$.

1. Introduction. In this paper we design a new first-order penalty-based algorithm for solving the ℓ_1-minimization problem

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

where ℓ_1-norm $\|x\|_1 = \sum_{i=1}^n |x(i)|$, $x(i)$ denotes the i-th component of the vector x, $b \in \mathbb{R}^m$, $x \in \mathbb{R}^n$, $A \in \mathbb{R}^{m\times n}$ and the number of equations $m \ll n$. This problem can be reformulated into a linear program (LP) and therefore, can, in theory, be solved efficiently.

LPs of the form (1.1) have recently attracted a lot of attention since they serve as the basis for a new signal processing paradigm known as *compressive sensing* (CS) [5, 6, 7, 10]. The goal in CS is to recover a sparse signal x from a small set of linear measurements or transform values $b = Ax$. Ordinarily the sparse signal would have to be recovered by solving the NP-hard ℓ_0-minimization problem

$$\min_{x\in\mathbb{R}^n} \|x\|_0 \text{ subject to } Ax = b, \tag{1.2}$$

where the ℓ_0-norm $\|x\|_0 = \sum_{i=1}^n \mathbf{1}(x(i) \neq 0)$. Recently, Candes, Romberg and Tao [5, 6, 7] and Donoho [10] have shown that when the target signal x is s-sparse, i.e. only s of the n components are non-zeros, and the measurement matrix A satisfies some regularity conditions, the sparse signal can be recovered by solving the LP (1.1) with probability $1 - \mathcal{O}(e^{-\gamma n})$ for some $\gamma > 0$ when the number of measurements $m = \mathcal{O}(s\ln(n))$. Thus, in theory, the sparse signal can be recovered very efficiently.

However, in practice, solving the LP (1.1) is hard. This is because the constraint matrix A is large and dense, and the LPs are often ill-conditioned. Thus, general purpose simplex-based LP solvers are not able to efficiently solve (1.1). In typical CS applications, the problem dimension is large – $n \approx 10^6$; therefore, general purpose interior point methods that require factorization of an $m \times n$ matrix are not practical for solving LPs that arise in CS applications.

The measurement matrix A in CS applications has a lot of structure that can be exploited by special purpose algorithms. In many applications A is a partial discrete cosine transform matrix, i.e. measurement $b = Ax$ is the value of the discrete cosine transform (DCT) of the signal x for a small set of frequencies. Consequently, Ax and A^Ty can be computed very efficiently using the Fast Fourier Transform (FFT). In other applications A is a partial wavelet matrix; once again Ax and A^Ty can be computed efficiently by using forward and inverse wavelet transforms. A number of different recently proposed algorithms exploit this structural fact to efficiently solve (1.1). One class of these algorithms solve (1.1) in the Lagrangian form:

$$\min_{x\in\mathbb{R}^n} \lambda\|x\|_1 + \|Ax - b\|_2^2. \tag{1.3}$$

Figueiredo, Nowak and Wright [11] propose the GPSR algorithm that uses gradient projection method with Barzilai-Borwein steps to solve (1.3). Hale, Yin and Zhang [12, 13] propose solving (1.3) using fixed point continuation (FPC) algorithm that embeds soft-thresholding (IST) algorithm [8] in a continuation strategy. Wen, Yin, Goldfarb and Zhang [20] improve the performance of FPC by adding an active set (AS) step.

*IEOR Department, Columbia University. Email: nsa2106@columbia.edu

†IEOR Department, Columbia University. Email: gi10@columbia.edu

Yin, Osher, Goldfarb and Darbon [21] solve (1.3) using Bregman iterative regularization. In this algorithm one solves a sequence of problems of the form

$$\min_{x\in\mathbb{R}^n} \lambda\|x\|_1 + \frac{1}{2}\|Ax - f_k\|_2^2, \tag{1.4}$$

where f_k are obtained by suitably updating the measurement vector b. This method utilizes FPC for solving the unconstrained subproblems (1.4). GPSR, FPC and FPC-AS only converge to the optimal solution of (1.3). There are no known continuation schemes that ensure that these algorithm converge to the solution of (1.1). Bregman iteration based methods [21] provably converges to the optimal solution of the basis pursuit problem (1.1); however, the convergence rate is unknown.

Other algorithms for the ℓ_1-minimization problem include an iterative solver in an interior-point framework [16], and an accelerated projected gradient method [9]. In [19], Van den Berg and Friedlander adapt the nonmonotone spectral projected gradient algorithm to efficiently solve the LASSO subproblem $\Psi(t) = \min\{\|Ax - b\|_2 : \|x\|_1 \leq t\}$ and then update the LASSO parameter t using a Newton step to solve the relaxed ℓ_1-minimization problem

$$\begin{array}{ll} \min & \|x\|_1, \\ \text{s.t.} & \|Ax - b\|_2 \leq \epsilon. \end{array} \tag{1.5}$$

The algorithm in [19] provably converges to the optimal solution of the relaxed problem (1.5); however, the convergence rate is unknown.

In this paper we propose a new first-order smoothed penalty algorithm (SPA) to solve the sparse recovery problem $\min\{\|x\|_1 : Ax = b\}$. SPA employs Nesterov's optimal gradient method for non-smooth convex optimization [18] to solve the penalized subproblems. While this paper was being prepared for submission, we became aware of a technical report by Becker, Bobin and Candès [4] where they independently propose a new algorithm NESTA for solving the the relaxed ℓ_1-minimization problem (1.5) and, by setting $\epsilon = 0$, the ℓ_1-minimization problem (1.1), which is also an adaptation of Nesterov's optimal gradient method for non-smooth convex functions [18] to solve (1.5). NESTA computes an ϵ-optimal solution for (1.5) in $\mathcal{O}(\frac{1}{\epsilon})$ Nesterov updates, where each update involves computing the gradient of suitably smoothed version of the ℓ_1-norm $\|x\|_1$ and solving an optimization problem of the form

$$\min_{\{x:\|Ax-b\|_2\leq\epsilon\}} \Big\{c^T x + \frac{L}{2}\|x - z\|_2^2\Big\}. \tag{1.6}$$

See [18] and Section 2 for details on the smoothing and the optimization problem for Nesterov update. When $A^T A$ is an orthogonal projector, i.e. the rows of A are orthonormal (as is the case when A is a partial Fourier or DCT matrix), solving (1.6) requires one to compute one matrix-vector multiplication of the form Ax and one of the form $A^T y$, and is, therefore, very efficient in the CS context. However, when $A^T A$ is *not* orthogonal projector (as is the case when the measurement matrix A is a Gaussian matrix or when it corresponds to a partial non-orthogonal wavelet transform or the partial pseudo-polar Fourier transform that arises in the context of CT imaging [1]) the complexity of the update step is $\mathcal{O}(n^3)$ and is, therefore, prohibitive for practical applications. NESTA can be embedded in a continuation scheme that allows one to compute a solution with any desired accuracy.

In this paper we propose a new first-order sequential penalty algorithm (SPA) to solve the CS decoding problem (1.1). This algorithm was, in part, motivated by the fact that a direct application of the Nesterov non-smooth optimization to (1.1) (as in NESTA) results in a very expensive update step. SPA computes a solution for (1.1) by inexactly solving a sequence of optimization problems of the form

$$\min_{x\in\mathbb{R}^n} \big\{\lambda_k\|x\|_1 + \|Ax - b\|_2\big\},$$

where $\lambda_k \searrow 0$. In SPA the sub-problems are solved using the Nesterov optimal algorithm for simple sets [17, 18] . Each update step in the Nesterov algorithms reduces to computing the gradient of a suitably smoothed version of the function $\lambda_k\|x\|_1 + \|Ax - b\|_2$ and solving minimum norm problem of the form $\min\big\{c^T x + \frac{L}{2}\|x - z\|_2^2 : \|x\|_2 \leq \beta\big\}$. The optimal solution of the minimum norm problem $x^* = \frac{\beta}{\max\{\beta, \|z - \frac{1}{L}c\|_2\}}\big(z - \frac{1}{L}c\big)$

can be computed in $\mathcal{O}(n)$ complexity. We show in Section 2 that the complexity of computing gradient of a smoothed version of $\lambda\|x\|_1 + \|Ax-b\|_2$ is dominated by the time of computing $A^T(Ax-b)$, and can, therefore, be computed efficiently in the CS context. Since we penalize the infeasibility by the appropriately smoothed version of $\|Ax-b\|_2$, the iterates with small infeasibility are penalized harsher in SPA as compared to algorithms employing the smooth penalty $\|Ax-b\|_2^2$, and, therefore, we expect SPA to converge faster, especially when the tolerance on feasibility is small.

The main contributions of this paper are as follows:

(a) We show that SPA converges to an optimal solution x^* of (1.1), i.e. $x^* \in \operatorname{argmin}\{\|x\|_1 : Ax = b\}$. See Theorem 2.1 and Corollary 2.2 for details. In order for the algorithm to be efficient, we only require that the matrix-vector product Ax and A^Ty be computed efficiently; in particular, we do not require that A has orthonormal rows. This implies that our algorithm can be used to recover compressed CT scans [1] where A^TA is *not* an orthogonal projector.

(b) We show an explicit bound on the degree of sub-optimality $|\ \|x_k\|_1 - \|x^*\|_1|$ for *any* iterate x_k. Thus, the user can stop the algorithm at any iteration k with guarantee on the sub-optimality. See Theorem 2.4 for details. Using this result we also establish a convergence rate for the algorithm. We show that there exist a priori fixed parameter settings such that, for *all* small enough ϵ, the iterates x_k computed by our algorithm are ϵ-feasible, i.e. $\|Ax_k - b\|_2 \leq \epsilon$, and ϵ-optimal, $|\ \|x_k\|_1 - \|x^*\|_1| \leq \epsilon$, after $\tilde{\mathcal{O}}(\epsilon^{-\frac{3}{2}})$ iterations, where the complexity of each iteration is dominated by computing Ax and A^Ty for some x and y; and which is $\mathcal{O}(n\ln(n))$ when A is a partial DCT or DFT matrix. See Theorem 2.5 for details.

(c) The SPA algorithmic framework is very flexible. One can change the penalty $\|Ax-b\|_2$ to either $\|Ax-b\|_1$ or $\|Ax-b\|_\infty$ without affecting any aspect of the theoretical or practical performance. The framework easily extends to the relaxed recovery problem $\min\{\|x\|_1 : \|Ax-b\|_p \leq \delta\}$, where $p = 1, 2, \infty$.

As noted earlier, NESTA [4] can be embedded in a continuation scheme that computes a feasible and ϵ-optimal iterate in $\mathcal{O}(\epsilon^{-1})$ iterations, where the the complexity of each iteration is determined by one matrix-vector multiplication of the form Ax and one of the form A^Ty when A^TA is orthogonal projector (e.g. when A is a partial DCT or DFT matrix, the complexity is $\mathcal{O}(n\ln(n))$); otherwise, i.e. when A^TA is not an orthogonal projector, the complexity of each iteration is $\mathcal{O}(n^3)$. Thus, the worst case complexity of NESTA is superior to SPA when A^TA is orthogonal projector. However, since the Nesterov update in SPA is simply a projection of the gradient step on to an ℓ_2-ball, we expect that in practice SPA will be competitive with NESTA even in the special case. Our numerical results reported in Section 5 do support this hypothesis.

The rest of the paper is organized as follows. In Section 2 we motivate SPA and discuss its convergence properties. In Section 3 we discuss extensions of the algorithm to the related optimization problems. In Section 4 we discuss some implementation details and show SPA in full detail. In Section 5 we discuss results of our numerical experiments.

2. A smoothed penalty method for ℓ_1-minimization. We assume that A has full row rank. Consequently, A^T has full column rank. We propose solving (1.1) by inexactly solving a sequence of penalized problems of the form

$$\min\big\{\lambda\|x\|_1 + \|Ax-b\|_2\big\}, \tag{2.1}$$

with $\lambda \searrow 0$. Since $P(x) = \|Ax-b\|_2$ is an *exact* penalty function for the feasible region of (1.1), there exists $\lambda^* > 0$ such that the optimal solution x^* of (1.1) is optimal for (2.1) for all $\lambda \leq \lambda^* < \infty$. However, both $\|x\|_1$ and $P(x)$ are non-smooth convex functions of x; consequently, sub-gradient based optimization methods for (2.1) are likely to perform poorly. We propose an algorithm that computes an optimal solution for ℓ_1-minimization problem (1.1) by solving a sequence of appropriately "smoothed" version of (2.1). The smoothing and the algorithm builds on the work of Nesterov [18]. Since we solve a smoothed version of the penalized optimization problem (2.1), we are not guaranteed that the optimal solution x^* of (1.1) is a solution of the smoothed optimization problem for some $\lambda > 0$.

Since $\|x\|_1 = \max_{\{u:\|u\|_\infty \leq 1\}}\{u^Tx\}$, we smooth $\|x\|_1$ by setting

$$f_\mu(x) = \max_{\{u:\|u\|_\infty \leq 1\}} \Big\{x^Tu - \frac{\mu}{2}\|u\|_2^2\Big\}, \tag{2.2}$$

where $\mu > 0$. The optimal u for a particular x is given by

$$u_x(i) = \text{sign}(x(i)) \min\left\{\frac{|x(i)|}{\mu}, 1\right\}, \quad i = 1, \ldots, n, \tag{2.3}$$

where

$$\text{sign}(x) = \begin{cases} 1 & x > 0, \\ 0 & x = 0, \\ -1 & x < 0. \end{cases}$$

The smoothed function $f_\mu(x) = \sum_{i=1}^n H_\mu(x(i))$, where

$$H_\mu(y) = \begin{cases} \frac{y^2}{2\mu}, & 0 \le |y| \le \mu, \\ |y| - \frac{\mu}{2}, & \mu < |y|, \end{cases} \tag{2.4}$$

denotes the Hüber penalty function used in robust statistics [15]. The function $f_\mu(x)$ is convex with a Lipschitz continuous gradient $\nabla f_\mu(x) = u_x$ with the Lipschitz constant $L_\mu^f = \frac{1}{\mu}$.

We smooth the penalty function $P(x)$ by setting

$$P_\nu(x) = \max_{\{w:\|w\|_2 \le 1\}} \left\{(Ax-b)^T w - \frac{\nu}{2}\|w\|_2^2\right\} = H_\nu(\|Ax-b\|_2), \tag{2.5}$$

for $\nu > 0$. The optimal value of w for a particular x is given by

$$w_x = \begin{cases} \frac{Ax-b}{\nu}, & \|Ax-b\|_2 \le \nu, \\ \\ \frac{Ax-b}{\|Ax-b\|_2}, & \|Ax-b\|_2 > \nu. \end{cases} \tag{2.6}$$

Note that $\|w_x\|_2 = \min\{1, \nu^{-1}\|Ax-b\|_2\}$, and when $\|w_x\|_2 < 1$ we must have $w_x = \nu^{-1}(Ax-b)$.

The function $P_\nu(x)$ is convex and has Lipschitz continuous gradient $\nabla P_\nu(x) = A^T w_x$ with the Lipschitz constant $L_\nu^P = \frac{\|A\|_2^2}{\nu}$, where $\|A\|_2 = \max_{\{\|u\|_2 \le 1, \|v\|_2 \le 1\}}\{u^T A v\} = \sigma_{\max}(A)$, where $\sigma_{\max}(A)$ denotes the largest singular value of A. Moreover, the penalty $P_\nu(x) \ge 0$ for all x and $\nu > 0$.

One can smooth $f(x)$ (resp. $P(x)$) using any strongly convex function $h(u)$ (resp. $g(w)$) for which the maximization problem defining $f_\mu(x)$ (resp. $P_\nu(x)$) can be solved in closed form. We chose $h(u) = \frac{1}{2}\|u\|_2^2$ (resp. $g(w) = \frac{1}{2}\|w\|_2^2$) because u_x (resp. w_x) has a very simple structure which allows us to establish convergence results easily. See [18] for details of the smoothing.

We propose to solve (1.1) by solving a sequence of smoothed penalty problems of the form

$$\min_{x \in \mathbb{R}^n} \left\{\lambda f_\mu(x) + P_\nu(x)\right\}. \tag{2.7}$$

The outline of the Smoothed Penalty Algorithm (SPA) is displayed in Figure 2.1 (see Figure 4.1 for more implementation details). This is the version we use for the establishing the theoretical properties of the algorithm. We denote $x_k^* = \text{argmin}_{x \in \mathbb{R}^n}\left\{Q_k(x)\right\}$ and $x^* = \text{argmin}\{\|x\|_1 : \ Ax = b\}$. The algorithm takes the sequence of multipliers $\left\{(\mu_k, \nu_k, \lambda_k, \epsilon_k, \tau_k)\right\}_{k \in \mathbb{Z}_+}$ as an input. We let $\gamma_k \searrow \eta$ (resp. $\gamma_k \nearrow \eta$) denote that the sequence $\{\gamma_k\}$ is monotonically decreasing (resp. increasing). In Section 4 we describe how we set these multipliers.

THEOREM 2.1. *Let $\{x_k \in \mathbb{R}^n : k \in \mathbb{Z}_+\}$ denote the sequence of iterates generated by the Smoothed Penalty Algorithm (SPA) displayed in Figure 2.1 when*

(i) Smoothing parameter for $\|x\|_1$: $\mu_k \searrow 0$
(ii) Smoothing parameter for $P(x)$: $\nu_k \searrow 0$
(iii) Penalty multiplier: $\lambda_k \searrow 0$
(iv) Approximate optimality parameters: $\epsilon_k \searrow 0$ such that $\frac{\epsilon_k}{\lambda_k} \to 0$ and $\tau_k \searrow 0$ such that $\frac{\tau_k}{\lambda_k} \to 0$.

Outline of Smoothed Penalty Algorithm

input: multipliers $\left\{\left(\mu_k, \nu_k, \lambda_k, \epsilon_k, \tau_k\right)\right\}_{k\in\mathbb{Z}_+}$
$x_0 \leftarrow \arg\min\{\|x\|_2 : \ Ax = b\}$
$k \leftarrow 0$
while (Stopping Criterion not true)
do
$k \leftarrow k + 1$
$Q_k(x) = \lambda_k f_{\mu_k}(x) + P_{\nu_k}(x)$
$\beta_k \leftarrow f_{\mu_k}(x_0) + \frac{\mu_1 n}{2}$
$F_k = \{x \in \mathbb{R}^n : \ \|x\|_2 \le \beta_k\}$
1 Use Nesterov's algorithm [17, 18] starting from x_{k-1} to compute $x_k \in F_k$ **such that**
either
(a) $Q_k(x_k) \le \inf_{x\in\mathbb{R}^n} Q_k(x) + \epsilon_k$
or
(b) $\|\nabla Q_k(x_k)\|_2 \le \tau_k$

return x_k

FIG. 2.1. *Outline of Smoothed Penalty Algorithm (SPA)*

Then, $\{x_k \in \mathbb{R}^n : k \in \mathbb{Z}_+\}$ is a bounded sequence. Let $\bar{x}$ denote any limit point of $\{x_k : k \in \mathbb{Z}_+\}$. Then $\bar{x}$ is an optimal solution of the ℓ_1-minimization problem (1.1).

Proof. Fix $k \ge 1$. Then

$$\|x_k^*\|_2 \le \|x_k^*\|_1 \le f_{\mu_k}(x_k^*) + \frac{\mu_k n}{2} \le \lambda_k^{-1} Q_k(x_k^*) + \frac{\mu_k n}{2} \le \lambda_k^{-1} Q_k(x_0) + \frac{\mu_k n}{2} \le f_{\mu_k}(x_0) + \frac{\mu_1 n}{2} = \beta_k, \quad (2.8)$$

where the first inequality follows from $\|x\|_2 \le \|x\|_1$, the second inequality holds because the optimal u_x in (2.3) satisfies $\|u_x\|_2^2 \le n$, the third inequality follows from $P_{\nu_k}(x) \ge 0$ for all x, the fourth inequality uses the definition $x_k^* = \operatorname{argmin}_{x\in\mathbb{R}^n} Q_k(x)$, and the final inequality follows from the fact that $P_\nu(x_0) = 0$ since $Ax_0 = b$ and $\mu_1 \ge \mu_k$ for all $k \ge 1$. Thus, we can restrict the iterate $x_k \in F_k = \{x : \|x\|_2 \le \beta_k\}$ without any loss of generality. Moreover, since $f_{\mu_k}(x) \le \|x\|_1$ for all x and $\mu_k \searrow 0$, $\beta_k \le \|x_0\|_1 + \frac{\mu_1 n}{2}$ for all $k \ge 1$. Therefore, sequence $\{x_k\}_{k\in\mathbb{Z}_+}$ is bounded and has a limit point. Let $\bar{x}$ denote any limit point of this sequence and let $\mathcal{K} \subset \mathbb{Z}_+$ denote a sub-sequence such that $\lim_{k\in\mathcal{K}} x_k = \bar{x}$. Along the sub-sequence $\mathcal{K}$, we have one of the following three possibilities:

(i) eventually all iterates x_k satisfy the ϵ-optimality stopping condition (a),
(ii) eventually all iterates x_k satisfy the gradient stopping condition (b),
(iii) for all k, there exists $k_a > k$ and $k_b > k$ such that the iterate x_{k_a} satisfies the ϵ-optimality stopping condition (a), and x_{k_b} satisfy the gradient stopping condition (b).

Thus, we are guaranteed that we either have a sub-sequence $\mathcal{K}_a \subset \mathcal{K}$ such that for all $k \in \mathcal{K}_a$, x_k satisfies ϵ-optimality stopping condition (a); or we have a sub-squence $\mathcal{K}_b \subset \mathcal{K}$ such that for all $k \in \mathcal{K}_b$, x_k satisfies the gradient stopping condition (b). We consider each of these two cases below.

(a) There exists sub-sequence $\mathcal{K}_a \subset \mathcal{K}$ such that for all $k \in \mathcal{K}_a$, x_k satisfies ϵ-optimality stopping condition (a). Since $Q_k(x_k) \le Q_k(x_k^*) + \epsilon_k$, it follows that

$$\|x_k\|_1 \le f_{\mu_k}(x_k) + \frac{\mu_k n}{2} \le \lambda_k^{-1} Q(x_k) + \frac{\mu_k n}{2} \le \lambda_k^{-1} Q(x_k^*) + \frac{\epsilon_k}{\lambda_k} + \frac{\mu_k n}{2}.$$

Since $x^* = \operatorname{argmin}\{\|x\|_1 : Ax = b\}$ satisfies $Ax^* = b$, all the bounds in (2.8) hold when x_0 replaced by x^*. Thus, for all $k \in \mathcal{K}_a$

$$\|x_k\|_1 \le \|x^*\|_1 + \frac{\epsilon_k}{\lambda_k} + \frac{\mu_k n}{2}. \quad (2.9)$$

Since $\mu_k \searrow 0$, $\frac{\epsilon_k}{\lambda_k} \to 0$, taking the limit of both sides of (2.9) along the subsequence $\mathcal{K}_a$, we have

$$\|\bar{x}\|_1 = \lim_{k\in\mathcal{K}_a} \|x_k\|_1 \le \|x^*\|_1 + \lim_{k\in\mathcal{K}_a} \left(\frac{\mu_k n}{2} + \frac{\epsilon_k}{\lambda_k}\right) = \|x^*\|_1. \quad (2.10)$$

Since $f_\mu(.) \geq 0$ for all $\mu > 0$,

$$0 \leq P_{\nu_k}(x_k) \leq Q_k(x_k) \leq Q_k(x_k^*) + \epsilon_k \leq Q_k(x^*) + \epsilon_k \leq \lambda_k \|x^*\|_1 + \epsilon_k. \tag{2.11}$$

The penalty $P_\nu(x) = H_\nu(\|Ax - b\|_2)$ (see (2.5)), where $H_\nu(.)$ denotes the Hüber function given in (2.4). Since $\sup_x |H_{\nu_k}(x) - |x|| \leq \frac{\nu_k}{2}$, it follows that $\lim_{k \in \mathcal{K}_a} H_{\nu_k}(\|Ax_k - b\|_2) = \|A\bar{x} - b\|_2$. Since $\mu_k \searrow 0$ and $\lambda_k \searrow 0$, taking the limit of both sides of (2.11) along the sub-sequence $\mathcal{K}_a$, it follows that

$$\|A\bar{x} - b\|_2 = \lim_{k \in \mathcal{K}_a} P_{\nu_k}(x_k) \leq 0. \tag{2.12}$$

Thus, from (2.12), (2.10) and the fact that $x^* = \text{argmin}\{\|x\|_1 : \ Ax = b\}$, it follows that $\bar{x}$ is an optimal solution for the basis pursuit problem (1.1).

(b) There exists a sub-sequence $K_b \subseteq \mathcal{K}$ such that, for all $k \in \mathcal{K}_b$, x_k satisfies the gradient stopping condition (b). The gradient $\nabla Q_k(x_k) = \lambda_k u_k + A^T w_k$ where u_k satisfies (2.3) with $x = x_k$ and $\mu = \mu_k$, and w_k satisfies (2.6) with $x = x_k$ and $\nu = \nu_k$. Therefore, for all $k \in \mathcal{K}_b$,

$$\|\nabla Q_k(x_k)\|_2 = \|\lambda_k u_k + A^T w_k\|_2 \leq \tau_k.$$

Since $\|u_k\|_2 \leq \sqrt{n}$, it follows that

$$\|A^T w_k\|_2 \leq (\tau_k + \lambda_k \|u_k\|_2) \leq (\tau_k + \lambda_k \sqrt{n}).$$

Hence, $\lim_{k \in \mathcal{K}_b} \|A^T w_k\|_2 = 0$, or equivalently $\lim_{k \in \mathcal{K}_b} (A^T w_k) = 0$. Since A^T is assumed to have a full column rank, it follows that $\lim_{k \in \mathcal{K}_b} w_k = 0$.
Thus, $\exists K > 0$ such that $\|w_k\|_2 < 1$ for $k \in \mathcal{K}_b \cap \{l : l \geq K\}$. Recall that (2.6) implies that $\|w_k\|_2 = \min\{1, \nu_k^{-1} \|Ax_k - b\|_2\}$. Therefore, when $\|w_k\|_2 < 1$, we must have $w_k = \frac{1}{\nu_k}(Ax_k - b)$, i.e. $Ax_k - b = \nu_k w_k$. Consequently, $\lim_{k \in \mathcal{K}_b} w_k = 0$ and $\lim_{k \in \mathbb{Z}_+} \nu_k = 0$ together imply

$$A\bar{x} = b. \tag{2.13}$$

Next, we show that $\bar{x}$ is a Karush-Kuhn-Tucker (KKT) point, and is, therefore, optimal. For all $k \geq 1$, $\|\nabla f_{\mu_k}(x_k)\|_\infty = \|u_k\|_\infty \leq 1$. Therefore, there exists a vector $\bar{g} \in \mathbb{R}^n$ and a subsequence $\mathcal{K}_b' \subset \mathcal{K}_b$ such that

$$\lim_{k \in \mathcal{K}_b'} u_k = \bar{g}. \tag{2.14}$$

Since $\lim_{k \in \mathcal{K}} x_k = \bar{x}$ and (2.14) holds, it follows that $\bar{g} \in \partial \|x\|_1 \,|_{x=\bar{x}}$, i.e.

$$\bar{g}(i) = \begin{cases} \text{sign}(\bar{x}(i)) & |\bar{x}(i)| \neq 0, \\ \in [-1, 1] & \bar{x}(i) = 0. \end{cases}$$

Let $\theta_k = \frac{-w_k}{\lambda_k}$. Since $\nabla Q_k(x) = \lambda_k u_k + A^T w_k$ we have that $A^T \theta_k = u_k - \frac{1}{\lambda_k} \nabla Q_k(x_k)$. Since A^T has full column rank, $\theta_k = (AA^T)^{-1} A(u_k - \frac{1}{\lambda_k} \nabla Q_k(x_k))$. Since $\|\nabla Q_k(x_k)\|_2 \leq \tau_k$ all $k \in \mathcal{K}_b$, and $\frac{\tau_k}{\lambda_k} \to 0$, it follows that $\lim_{k \to \infty} \frac{\nabla Q_k(x_k)}{\lambda_k} = 0$. Since (2.14) implies that $\lim_{k \in \mathcal{K}_b'} u_k$ exists, it follows that $\bar{\theta} = \lim_{k \in \mathcal{K}_b'} \theta_k$ exists and we have

$$\bar{g} = A^T \bar{\theta}. \tag{2.15}$$

From (2.13) and (2.15), it follows that $\bar{x}$ is a KKT point for the ℓ_1-minimization problem (1.1). Since $\|x\|_1$ is convex, the optimization problem (1.1) is a convex programming problem with equality constraints, the KKT conditions are sufficient for optimality and, therefore, we can conclude that $\bar{x}$ is an optimal solution for (1.1).

□

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

COROLLARY 2.2. *Suppose the ℓ_1-minimization problem* $\min\{\|x\|_1 : Ax = b\}$ *has a* unique *optimal solution. Let $\{x_k : k \in \mathbb{Z}_+\}$ denote the sequence of iterates generated by the Smoothed Penalty Algorithm (SPA) displayed in Figure 2.1 when*

(i) Smoothing parameter for $\|x\|_1$*:* $\mu_k \searrow 0$
(ii) Smoothing parameter for $P(x)$*:* $\nu_k \searrow 0$
(iii) Penalty multiplier: $\lambda_k \searrow 0$
(iv) Approximate optimality parameters: $\epsilon_k \searrow 0$ *such that* $\frac{\epsilon_k}{\lambda_k} \to 0$ *and* $\tau_k \searrow 0$ *such that* $\frac{\tau_k}{\lambda_k} \to 0$.
Then $\lim_{k\to\infty} x_k = x^*$ *where* $x^* = \arg\min\{\|x\|_1 : Ax = b\}$. Theorem 2.1 and Corollary 2.2 continue to hold when we penalize the infeasibility by the ℓ_1 or the ℓ_∞-norm. Therefore, the version of SPA that uses ℓ_1 or ℓ_∞ penalty also recovers the optimal solution.

In order to minimize a convex function g with a Lipschitz continuous gradient over the simple constraint set of the form $F := \{x \in \mathbb{R}^n : \ \|x\|_2 \le \beta\}$, in every iteration of the Nesterov's optimal gradient algorithm we need to solve two problems of the form

$$\min_{\|x\|_2 \le \beta} \Big\{ c^T x + \frac{L}{2}\|x - z\|_2^2 \Big\},$$

where c is a function of the gradients of the function g computed in all the previous iterates and L denotes the Lipschitz constant of the gradient of the function g. Therefore, the Nesterov update reduces to $x = \frac{\beta}{\max\{\beta,\ \|z - \frac{1}{L}c\|_2\}}\big(z - \frac{1}{L}c\big)$. Consequently, the most expensive step of the Nesterov algorithm when used in SPA is computing the gradient, $\nabla Q_k(x)$ which involves matrix multiplications of the form $A^T(Ax - b)$.

LEMMA 2.3. *Nesterov's optimal algorithm [17] with prox function* $h_k(x) = \frac{1}{2}\|x - x_{k-1}\|_2^2$ *applied to the constrained optimization problem* $\min\{Q_k(x) : \|x\|_2 \le \beta_k\}$ *computes an* x_k *satisfying Step 1 in SPA in*

$$N_k = \left\lfloor \beta_k \sqrt{\frac{8L_k}{\epsilon_k}} \right\rfloor \tag{2.16}$$

iterations, where $L_k = \frac{\lambda_k}{\mu_k} + \frac{\|A\|_2^2}{\nu_k}$ *denotes the Lipschitz constant for* ∇Q_k. *The computational complexity of each iteration is bounded by the complexity of computing* $A^T(Ax - b)$ *for an arbitrary* x.

Remark 2.1. *Nesterov optimal algorithm guarantees the bound (2.16) for* all *initial starting points for the k-th subproblem. We are not able to take advantage of the fact that the particular initial point* x_{k-1} *for the k-th subproblem is close to an optimal solution for the k-th subproblem since* $Q_{k-1}(x) \approx Q_k(x)$ *for all* x.

Proof. We apply Nesterov's optimal algorithm to the optimization problem $\min\{Q_k(x) : \|x\|_2 \le \beta_k\}$ with the prox function $h_k(x) = \frac{1}{2}\|x - x_{k-1}\|^2$. Let $\{z_l : l \ge 1\}$ denote the sequence of iterates computed by the Nesterov algorithm. We terminate and set $x_k = z_l$ whenever either $\|\nabla Q_k(z_l)\|_2 \le \tau_k$ or we can guarantee $Q_k(z_l) \le Q_k(x_k^*) + \epsilon_k$.

In this proof we will bound the number of Nesterov iterations required to compute an iterate z_l that satisfies the ϵ-optimality stopping condition $Q_k(z_l) \le Q_k(x_k^*) + \epsilon_k$. The bound we compute is clearly an upper bound on the number of Nesterov iterations required to compute the iterate x_k.

Since the convexity parameter σ_k of the prox-function $h_k(x)$ is 1 and $h(x_k^*) \le \frac{1}{2}(\|x_k^*\|_2 + \|x_{k-1}\|_2)^2 \le 2\beta_k^2$, Nesterov optimal algorithm [17, 18] guarantees that

$$Q_k(z_l) - Q_k(x_k^*) \le \frac{4L_k h(x_k^*)}{\sigma_k(l+1)(l+2)} \le \frac{8L_k\beta_k^2}{(l+1)^2}.$$

Thus, z_l is ϵ_k-optimal for all $l \ge \left\lfloor \beta_k\sqrt{\frac{8L_k}{\epsilon_k}} \right\rfloor \equiv N_k$.

In each iteration of the Nesterov algorithm one has to compute one gradient $\nabla Q_k(y)$ and solve two optimization problems of the form $\min\{c^T x + \frac{L_k}{2}\|x - z\|_2^2 : \|x\|_2 \le \beta_k\}$ for given c and z. From (2.6) and (2.3), it follows that the complexity of computing the gradient $\nabla Q_k(y) = \lambda_k u_y + A^T w_y$ is given by the complexity of computing the matrix-vector product of the form $A^T(Ay - b)$. Since $\operatorname{argmin}\{c^T x + \frac{L_k}{2}\|x - z\|^2 : \|x\|_2 \le \beta_k\} = \frac{\beta_k}{\max\{\beta_k,\ \|z - \frac{1}{L_k}c\|_2\}}\ (z - \frac{1}{L_k}c)$ can be computed in $\mathcal{O}(n)$ worst-case complexity, it follows that the computational complexity of each Nesterov update step is bounded by the complexity of computing $A^T(Ay - b)$ for an arbitrary y. □

Next, we characterize the finite iteration performance of SPA. This analysis will lead to a convergence rate result in Theorem 2.5.

THEOREM 2.4. *Let $\{x_k\}_{k\in\mathbb{Z}_+}$ denote the sequence of iterates generated by the Smoothed Penalty Algorithm (SPA) where the multipliers τ_k and ϵ_k are chosen to satisfy $\tau_k \leq \sqrt{2L_k\epsilon_k}$, for all $k \geq 1$. Then for all $k \geq 1$*

$$\|x_k\|_1 - \|x^*\|_1 \leq \max\left\{\frac{\epsilon_k}{\lambda_k},\ \frac{\tau_k}{\lambda_k}\cdot(\|x^*\|_1+\beta_k)\right\} + \frac{\mu_k n}{2}. \tag{2.17}$$

Let $\sigma_{\min}(A)$ denote the smallest non-zero singular value of A. Then for all k satisfying $\sqrt{2L_k\epsilon_k} + \lambda_k\sqrt{n} < \sigma_{min}(A)$,

$$\begin{aligned} \|x_k\|_1 - \|x^*\|_1 &\geq -\frac{\mu_k n}{2} - \frac{\nu_k}{\lambda_k}, \\ \|Ax_k - b\|_2 &\leq \nu_k. \end{aligned} \tag{2.18}$$

Proof. Fix $k \geq 1$. Suppose x_k satisfies the ϵ-optimality stopping condition (a), i.e. $Q_k(x_k) \leq \inf_{x\in\mathbb{R}^n} Q_k(x) + \epsilon_k$. Then the bound (2.9) in the proof of Theorem 2.1 implies that

$$\|x_k\|_1 \leq \|x^*\|_1 + \frac{\epsilon_k}{\lambda_k} + \frac{\mu_k n}{2}.$$

Now, suppose x_k satisfies the gradient-stopping condition (b). Since Q_k is convex, it follows that $Q_k(x^*) \geq Q_k(x_k) + \nabla Q_k(x_k)^T(x^* - x_k)$. Thus, Cauchy-Schwartz inequality implies that

$$Q_k(x_k) \leq Q_k(x^*) + \|\nabla Q_k(x_k)\|_2\|x^* - x_k\|_2 \leq \lambda_k\|x^*\|_1 + \tau_k(\|x^*\|_1 + \beta_k).$$

where the last inequality follows from the fact that $f_{\mu_k}(x) \leq \|x\|_1$ for all x, $P_{\nu_k}(x^*) = 0$ because $Ax^* = b$, $\|x_*\|_2 \leq \|x_*\|_1$ and $\|x_k\|_2 \leq \beta_k$. Since $f_{\mu_k}(x) \geq \|x\|_1 - \frac{\mu_k n}{2}$ and $P_\nu(x) \geq 0$, it follows that

$$\|x_k\|_1 \leq \lambda_k^{-1}Q_k(x_k) + \frac{\mu_k n}{2} \leq \|x^*\|_1 + \frac{\tau_k}{\lambda_k}\cdot(\|x^*\|_1 + \beta_k) + \frac{\mu_k n}{2}.$$

Thus, (2.17) follows.

Next, we establish the bound on the infeasibility of the iterate x_k for iteration count k satisfying $\sqrt{2L_k\epsilon_k} + \lambda_k\sqrt{n} < \sigma_{\min}(A)$. Since Q_k is convex and has a Lipschitz continuous gradient with the Lipschitz constant L_k, it follows that

$$\frac{1}{2L_k}\|\nabla Q_k(x)\|_2^2 \leq Q_k(x) - Q_k(x_k^*),$$

for all $x \in \mathbb{R}^n$. When x_k satisfies the ϵ_k-optimality condition we are guaranteed that $\|\nabla Q_k(x_k)\|_2 \leq \sqrt{2L_k\epsilon_k}$. Therefore, the iterate x_k always satisfies $\|\nabla Q_k(x_k)\|_2 \leq \max\{\sqrt{2L_k\epsilon_k},\ \tau_k\}$. Since $\tau_k \leq \sqrt{2L_k\epsilon_k}$ for all $k \geq 1$, we have $\|\nabla Q_k(x_k)\|_2 \leq \sqrt{2L_k\epsilon_k}$ for all $k \geq 1$ and this, in turn, implies that

$$\|\nabla P_{\nu_k}(x_k)\|_2 = \|A^T w_k\|_2 \leq (\sqrt{2L_k\epsilon_k} + \lambda_k\|\nabla f_{\mu_k}(x_k)\|_2) \leq \sqrt{2L_k\epsilon_k} + \lambda_k\sqrt{n},$$

where the last inequality follows from the fact that $\|\nabla f_{\mu_k}(x_k)\|_\infty = \|u_k\|_\infty \leq 1$. Since A has full row rank, it follows that

$$\|w_k\|_2 \leq \frac{\|A^T w_k\|_2}{\sigma_{\min}(A)} \leq \frac{\sqrt{2L_k\epsilon_k} + \lambda_k\sqrt{n}}{\sigma_{\min}(A)} < 1.$$

Since $\|w_k\| < 1$, (2.6) implies that $w_k = \frac{Ax_k - b}{\nu_k}$, and therefore, $\|Ax_k - b\|_2 = \nu_k\|w_k\|_2 \leq \nu_k$.

The final step in the proof is to establish the lower bound in (2.18). Fix an iterate k such that $\sqrt{2L_k\epsilon_k} + \lambda_k\sqrt{n} < \sigma_{\min}(A)$. We establish a lower bound for $\|x_k\|_1$ using the linear programming duality:

$$\begin{array}{lll} \text{minimize} & \|x\|_1, & \qquad \text{maximize} \quad b^T w, \\ \text{subject to} & Ax = b. & \qquad \text{subject to} \quad \|A^T w\|_\infty \leq 1. \end{array}$$

Let w^* denote the optimal dual solution. Then $\|x^*\|_1 = b^T w^*$ and $\|A^T w^*\|_2 \leq \sqrt{n}$ and $\|w^*\|_2 \leq \frac{\sqrt{n}}{\sigma_{\min}(A)}$. Linear programming duality also implies that

$$\begin{array}{lll} \underset{x \in \mathbb{R}^n}{\text{minimize}} & \lambda\|x\|_1 + \|Ax - b\|_2 & \qquad \underset{\text{subject to}}{\text{maximize}} \quad \lambda b^T w, \\ & & \qquad\quad \|A^T w\|_\infty \leq 1, \\ & & \qquad\quad \|w\|_2 \leq \lambda^{-1}. \end{array} \tag{2.19}$$

Since $\|w^*\|_2 \leq \frac{\sqrt{n}}{\sigma_{\min}(A)}$, it follows that w^* is feasible for the dual program in (2.19) whenever $\lambda\sqrt{n} \leq \sigma_{\min}(A)$. Since $\sqrt{2L_k\epsilon_k} + \lambda_k\sqrt{n} \leq \sigma_{\min}(A)$, weak duality implies that $\min_x\{\lambda_k\|x\|_1 + \|Ax - b\|_2\} \geq \lambda_k b^T w^*$. Next, we relate the exact penalty objective function to the smoothed penalty function $Q_k(x)$.

$$\begin{aligned} \min_x Q_k(x) &\geq \min_x \left\{\lambda_k\|x\|_1 + \|Ax - b\|_2\right\} - \frac{\lambda_k\mu_k n}{2} - \frac{\nu_k}{2}, \\ &\geq \lambda_k b^T w^* - \frac{\lambda_k\mu_k n}{2} - \frac{\nu_k}{2}, \\ &= \lambda_k\|x^*\|_1 - \frac{\lambda_k\mu_k n}{2} - \frac{\nu_k}{2}. \end{aligned} \tag{2.20}$$

Also,

$$\begin{aligned} Q_k(x_k) &= \lambda_k f_{\mu_k}(x_k) + P_{\nu_k}(x_k), \\ &\leq \lambda_k\|x_k\|_1 + \frac{\|Ax_k - b\|_2^2}{2\nu_k}, \qquad (2.21) \\ &\leq \lambda_k\|x_k\|_1 + \frac{\nu_k}{2}, \qquad (2.22) \end{aligned}$$

where (2.21) and (2.22) follow from the fact that $\frac{\sqrt{2L_k\epsilon_k}+\sqrt{n}\lambda_k}{\sigma_{\min}(A)} < 1$ implies that $w_k = \frac{Ax_k - b}{\nu_k}$ and $\|Ax_k - b\|_2 \leq \nu_k$. The lower bound follows from (2.20) and (2.22). □

Theorem 2.1 and 2.4 reveal the relationship between four multipliers $(\mu_k, \nu_k, \epsilon_k, \tau_k, \lambda_k)$ used in SPA. For the convergence proof and finite iteration performance of SPA given in Theorem 2.1 and Theorem 2.4 to be true at the same time, we should require that $\epsilon_k/\lambda_k \to 0$ and $\tau_k/\lambda_k \to 0$ as $k \to \infty$ such that $\tau_k \leq \sqrt{2L_k\epsilon_k}$ for all $k \geq 1$. Both the upper and lower bounds (2.17) and (2.18) in Theorem 2.4 imply that all the terms in the multiplier sequence $\{\mu_k\}_{k\in\mathbb{Z}_+}$ should be scaled as $\frac{1}{n}$ as a function of the target signal dimension n to ensure that the error is $\mathcal{O}(1)$. The lower bound (2.18) in Theorem 2.4 implies that $\nu_k/\lambda_k \to 0$ as $k \to \infty$. The lower bound holds only if $\sqrt{2L_k\epsilon_k} + \lambda_k\sqrt{n} \leq \sigma_{\min}(A)$; therefore, the terms in the sequence $\{\lambda_k\}_{k\in\mathbb{Z}_+}$ should scale as $\frac{\sigma_{\min}(A)}{\sqrt{n}}$. In compressive sensing applications with Discrete Cosine Transform (DCT) $\sigma_{\min}(A) = 1$; thus, in this setting $\lambda_k = \mathcal{O}(\frac{1}{\sqrt{n}})$ for all $k \in \mathbb{Z}_+$. In the result below we fix $(\kappa_0, \epsilon_0, \tau_0, \lambda_0) > 0$ (independent of the problem dimension n) and then use these scaling rules to construct a multiplier sequence that guarantees a very good convergence rate for SPA.

THEOREM 2.5. *Fix $0 < \delta < 1$, $0 < \alpha < 1$ and strictly positive parameters $(\kappa_0, \epsilon_0, \tau_0, \lambda_0)$ such that $\epsilon_0 \leq 4n\left(\lambda_0 + \|A\|_2^2\right)^2$. Select the sequence multipliers as follows:*

$$\begin{array}{llll} \mu_k = \begin{cases} \frac{\kappa_0}{n}, & k = 1, \\ \alpha^{(1+\delta)}\mu_{k-1} & k > 1, \end{cases} & \nu_k = \begin{cases} \frac{\kappa_0}{\sqrt{n}}, & k = 1, \\ \alpha^{(1+\delta)}\nu_{k-1} & k > 1, \end{cases} & & \\ \epsilon_k = \begin{cases} \frac{\epsilon_0}{2L_1}, & k = 1, \\ \alpha^{(1+\delta)}\left(\frac{L_{k-1}}{L_k}\right)\epsilon_{k-1}, & k > 1, \end{cases} & \lambda_k = \begin{cases} \frac{\lambda_0}{\sqrt{n}} & k = 1, \\ \alpha^{\delta}\lambda_{k-1} & k > 1, \end{cases} & \tau_k = \frac{\epsilon_k}{2\|x_0\|_1 + \kappa_0} & k \geq 1, \end{array} \tag{2.23}$$

where $L_k = \frac{\lambda_k}{\mu_k} + \frac{\|A\|_2^2}{\nu_k}$ denotes the Lipschitz constant of $\nabla Q_k(.)$. Let $\{x_k \in \mathbb{R}^n : k \in \mathbb{Z}_+\}$ denote sequence of iterates computed by the SPA algorithm corresponding to this set of multipliers.

Let $\tilde{\epsilon} = \frac{\kappa_0}{\sqrt{n}C^{\frac{1+\delta}{\delta}}}$, where $C = \frac{(\lambda_0 + \sqrt{\epsilon_0})}{\sigma_{min}(A)}$. Then for $\epsilon < \tilde{\epsilon}$, there exists an k_ϵ, which is $\mathcal{O}\left(\frac{\ln(\epsilon^{-1})}{\ln(\alpha^{-1})}\right)$, such that for all $k \geq k_\epsilon$,

$$\|Ax_{k_\epsilon} - b\|_2 \leq \epsilon, \qquad |\,\|x_{k_\epsilon}\|_1 - \|x^*\|_1| \leq \epsilon,$$

and x_{k_ϵ} can be computed in at most $\mathcal{O}\Big(\big(2\|x_0\|_1+\kappa_0\big)\sqrt{n}\;\epsilon^{-\frac{3}{2}(1+\delta)}\Big)$ Nesterov updates, where x^* denotes the optimal solution of (1.1).

Remark 2.2.

1. *Note that the parameter sequence $\{(\mu_k,\nu_k,\epsilon_k,\tau_k,\lambda_k): k\geq 1\}$ is fixed* apriori*, in particular it is independent of ϵ. Theorem 2.5 establishes that the iterates x_k are ϵ-feasible and ϵ-optimal for all $k\geq k_\epsilon$ and $\epsilon\leq\tilde{\epsilon}$, and the running time to compute x_{k_ϵ} grows as $\epsilon^{-\frac{3}{2}(1+\delta)}$.*
2. *From Lemma 2.3 it follows that the complexity of computing one matrix-vector product of Ax and one matrix-vector product of the form A^Ty are the dominant terms in the computational complexity of computing a Nesterov update. When A is partial DCT or Fourier matrix, both these matrix-vector products can be computed in $\mathcal{O}(n\ln(n))$ operations.*
3. *The bounds established above for $\tilde{\epsilon}$, k_ϵ and the running time all contain constants that depend on the initial values $(\kappa_0,\epsilon_0,\tau_0,\lambda_0)$, α and δ. We now evaluate the constants as a function of α and δ for a particular choice of the initial values and demonstrate that these constants are not very large. Suppose we scale A so that $\|A\|_2=\sigma_{max}(A)=1$ and set $\kappa_0=\epsilon_0=1$, and $\lambda_0=2$. Then $\tilde{\epsilon}=\frac{1}{\sqrt{n}}\left(\frac{\sigma_{min}(A)}{3}\right)^{\frac{1+\delta}{\delta}}$, for all $\epsilon<\tilde{\epsilon}$, $k_\epsilon\leq\lceil\frac{\ln(\epsilon^{-1})}{\ln(\alpha^{-1})}\rceil$, and SPA requires at most $\left\lceil\frac{6(2\|x_0\|_1+1)\sqrt{n}}{\alpha^{-\frac{3}{2}(1+\delta)}-1}\;\epsilon^{-\frac{3}{2}(1+\delta)}\right\rceil$ Nesterov updates, i.e. inner iterations denoted by N_{in}, for computing x_{k_ϵ}. In particular, for $0<\alpha<\frac{1}{4}$, then we simply have $N_{in}\leq\frac{6}{7}\left(2\|x_0\|_1+1\right)\sqrt{n}\;\epsilon^{-\frac{3}{2}(1+\delta)}$.*

Proof. In this proof we rely on the results in Theorem 2.1 and Theorem 2.4. Therefore, we have to ensure that all the relevant conditions are met for the specific choice of multipliers selected above. Since $\alpha<1$, all the multipliers μ_k, ν_k, λ_k converge to zero monotonically. Since $L_k=\frac{\lambda_k}{\mu_k}+\frac{\|A\|_2^2}{\nu_k}\geq\alpha^{-1}L_{k-1}$, it follows that $\epsilon_k=\alpha^{(1+\delta)}\Big(\frac{L_{k-1}}{L_k}\Big)\epsilon_{k-1}\leq\alpha^{(2+\delta)}\epsilon_{k-1}$. Thus, ϵ_k and $\tau_k=\frac{\epsilon_k}{2\|x_0\|_1+\kappa_0}$ converge to zero. The ratio $\frac{\epsilon_k}{\lambda_k^2}\leq\alpha^{(2-\delta)}\frac{\epsilon_{k-1}}{\lambda_{k-1}^2}$. Therefore, $\frac{\epsilon_k}{\lambda_k^2}$ converges to zero and since $\tau_k=(2\|x_0\|_1+\kappa_0)^{-1}\epsilon_k$, it follows that

$$\frac{\tau_k}{\lambda_k}=(2\|x_0\|_1+\kappa_0)^{-1}\cdot\frac{\epsilon_k}{\lambda_k^2}\cdot\lambda_k\to 0.$$

Moreover, $L_1=\left(\frac{\lambda_0+\|A\|_2^2}{\kappa_0}\right)\sqrt{n}$, $L_k\geq L_1$ for all $k\geq 1$ and $\epsilon_0\leq 4n\left(\lambda_0+\|A\|_2^2\right)^2$ together imply that for all $k\geq 1$,

$$\tau_k=\frac{\epsilon_k}{(2\|x_0\|_1+\kappa_0)}\leq\frac{\epsilon_k}{\kappa_0}\leq\left(\frac{\sqrt{\epsilon_0}}{2L_k\kappa_0}\right)\sqrt{2L_k\epsilon_k}\leq\left(\frac{\sqrt{\epsilon_0}}{2L_1\kappa_0}\right)\sqrt{2L_k\epsilon_k}\leq\sqrt{2L_k\epsilon_k}.$$

Thus, the multiplier sequence $\{(\mu_k,\nu_k,\epsilon_k,\tau_k,\lambda_k)\}_{k\in\mathbb{Z}_+}$ satisfies all the conditions for Theorem 2.1 and Theorem 2.4.

Since $\sqrt{2L_k\epsilon_k}+\lambda_k\sqrt{n}\leq\sqrt{\epsilon_0}\alpha^{\frac{1}{2}(1+\delta)(k-1)}+\lambda_0\alpha^{\delta(k-1)}\leq(\lambda_0+\sqrt{\epsilon_0})\alpha^{\delta(k-1)}$, it follows that $\sqrt{2L_k\epsilon_k}+\lambda_k\sqrt{n}\leq\sigma_{\min}(A)$ for all

$$k\geq K_1+1\equiv\frac{\ln\left(\frac{(\lambda_0+\sqrt{\epsilon_0})}{\sigma_{\min}(A)}\right)}{\delta\ln(\frac{1}{\alpha})}+1=\frac{\ln(C)}{\delta\ln(\frac{1}{\alpha})}+1.$$

Then Theorem 2.4 implies that for all $k\geq K_1+1$,

$$\begin{aligned}\|x_k\|_1-\|x^*\|_1\geq-\frac{\mu_k n}{2}-\frac{\nu_k}{\lambda_k}\quad&=\quad-\frac{\kappa_0}{2}\alpha^{(1+\delta)(k-1)}-\frac{\kappa_0}{\lambda_0}\alpha^{(k-1)},\\ \|Ax-b\|_2\leq\nu_k=\;&\frac{\kappa_0}{\sqrt{n}}\alpha^{(1+\delta)(k-1)},\end{aligned}$$

and for all $k\geq 1$

$$\begin{aligned}\|x_k\|_1-\|x^*\|_1\leq\max\left\{\frac{\epsilon_k}{\lambda_k},\;\frac{\tau_k\;(\|x^*\|_1+\beta_k)}{\lambda_k}\right\}+\frac{\mu_k n}{2}\quad&=\quad\frac{\epsilon_k}{\lambda_k}+\frac{\mu_k n}{2},&(2.24)\\ \leq\left(\frac{\kappa_0\epsilon_0}{2\lambda_0\|A\|_2^2}\right)\alpha^{(2+\delta)(k-1)}+\frac{\kappa_0}{2}\alpha^{(1+\delta)(k-1)},&(2.25)\end{aligned}$$

where the equality in (2.24) follows from the fact that

$$\tau_k(\|x^*\|_1+\beta_k)=\Big(\frac{\|x^*\|_1+\beta_k}{2\|x_0\|_1+\kappa_0}\Big)\epsilon_k<\Big(\frac{2\|x_0\|_1+\frac{\kappa_0}{2}}{2\|x_0\|_1+\kappa_0}\Big)\epsilon_k<\epsilon_k,$$

and the inequality in (2.25) follows from $L_k\geq\frac{\|A\|_2^2}{\nu_k}$ for all $k\geq 1$. Define

$$\begin{aligned}
K_2(\epsilon)&=\max\left\{\frac{\ln(\frac{\kappa_0}{\epsilon})}{(1+\delta)\ln(\frac{1}{\alpha})},\frac{\ln(\frac{2\kappa_0}{\lambda_0\epsilon})}{\ln(\frac{1}{\alpha})}\right\},\\
K_3(\epsilon)&=\frac{\ln(\frac{\kappa_0}{\epsilon\sqrt{n}})}{(1+\delta)\ln(\frac{1}{\alpha})},\\
K_4(\epsilon)&=\max\left\{\frac{\ln(\frac{\kappa_0}{\epsilon})}{(1+\delta)\ln(\frac{1}{\alpha})},\frac{\ln(\frac{\kappa_0\epsilon_0}{\lambda_0\|A\|_2^2\epsilon})}{(2+\delta)\ln(\frac{1}{\alpha})}\right\}.
\end{aligned}$$

Then, for all $k\geq K\equiv\max\{K_1,K_2(\epsilon),K_3(\epsilon),K_4(\epsilon)\}+1$, we $|\|x_k\|_1-\|x^*\|_1|\leq\epsilon$, and $\|Ax_k-b\|_2\leq\epsilon$. Since K_1 is independent of ϵ, $K_2(\epsilon)$ and $K_3(\epsilon)$ are all monotonically decreasing in ϵ, it follows that for all small enough ϵ, $\max\{K_2(\epsilon),K_3(\epsilon)\}\geq K_1$. In particular, this bound holds for all

$$\epsilon<\tilde{\epsilon}\equiv\frac{\kappa_0}{\sqrt{n}C^{\frac{1+\delta}{\delta}}}\leq\min\left\{\frac{\kappa_0}{\sqrt{n}C^{\frac{1+\delta}{\delta}}},\ \max\left\{\frac{\kappa_0}{C^{\frac{1+\delta}{\delta}}},\ \frac{2\kappa_0}{\lambda_0C^{\frac{1}{\delta}}}\right\}\right\}.$$

Moreover, for all $\epsilon<\tilde{\epsilon}$,

$$K=\max\{K_1,K_2(\epsilon),K_3(\epsilon),K_4(\epsilon)\}=\max\{K_2(\epsilon),K_3(\epsilon),K_4(\epsilon)\}\leq\frac{\ln(\frac{B}{\epsilon})}{\ln(\frac{1}{\alpha})},\tag{2.26}$$

where

$$B\equiv\max\left\{\kappa_0,\frac{2\kappa_0}{\lambda_0},\frac{\kappa_0\epsilon_0}{\lambda_0\|A\|_2^2},\frac{\kappa_0}{\sqrt{n}}\right\}.\tag{2.27}$$

Since $\beta_k\leq\|x_0\|_1+\frac{\mu_1 n}{2}=\|x_0\|_1+\frac{\kappa_0}{2}$, Lemma 2.3 imply that K iterations of Algorithm SPA require a total of

$$N_{in}\leq\sum_{k=1}^{K}\left\lfloor\sqrt{\frac{2L_k}{\epsilon_k}}(2\|x_0\|_1+\kappa_0)\right\rfloor$$

iterations of the Nesterov optimal algorithm for simple sets. Then

$$\sum_{k=1}^{K}\sqrt{\frac{2L_k}{\epsilon_k}}=2\sum_{k=0}^{K-1}\left(\left(\frac{\lambda_0\sqrt{n}}{\sqrt{\epsilon_0}\kappa_0}\right)\cdot\alpha^{-\frac{1}{2}(3+\delta)k}+\left(\frac{\|A\|_2^2\sqrt{n}}{\sqrt{\epsilon_0}\kappa_0}\right)\alpha^{-\frac{3}{2}(1+\delta)k}\right).$$

Thus,

$$N_{in}=\mathcal{O}\left(\sqrt{n}\left(\|x_0\|_1+\frac{\kappa_0}{2}\right)\left(\frac{\lambda_0+\|A\|_2^2}{\sqrt{\epsilon_0}\kappa_0}\right)\alpha^{-\frac{3}{2}(1+\delta)K}\right).$$

From (2.26) it follows that for all $\epsilon<\tilde{\epsilon}$,

$$N_{in}=\mathcal{O}\left(\sqrt{n}\left(\|x_0\|_1+\frac{\kappa_0}{2}\right)\left(\frac{\lambda_0+\|A\|_2^2}{\sqrt{\epsilon_0}\kappa_0}\right)\cdot B^{\frac{3}{2}(1+\delta)}\cdot\epsilon^{-\frac{3}{2}(1+\delta)}\right).$$

Suppose the complexity of computing matrix-vector products of the form Ax and A^Ty is $\mathcal{O}(n\ln(n))$. This is the case, for example, when A is a partial DCT or Fourier matrix. Then the complexity of each iteration in

the Nesterov algorithm [17] for simple set with the family of prox-functions $h_k(x) = \frac{1}{2}\|x - x_{k-1}\|_2^2$ requires $\mathcal{O}(n\ln(n))$ operations. Therefore, it follows that SPA computes an ϵ-infeasible and ϵ-optimal solution in $\mathcal{O}\Big(\Big(\|x_0\|_1 + \frac{\kappa_0}{2}\Big)n^{\frac{3}{2}}\ln(n)\epsilon^{-\frac{3}{2}(1+\delta)}\Big)$ operations. □

Suppose A does not have orthogonal rows but the complexity of computing the matrix-vector products of the form Ax, A^Ty is still $\mathcal{O}(n\ln(n))$. This is the case, for example, when A corresponds to a partial non-orthogonal wavelet transform or a partial pseudo-polar Fourier transform that arises in the context of CT imaging [1]. Theorem 2.5 establishes that SPA computes an ϵ-optimal solution in $\mathcal{O}(\epsilon^{-\frac{3}{2}})$ operations. An ϵ-optimal solution to the basis pursuit problem (1.1) can be computed in $\mathcal{O}(\epsilon^{-1})$ iterations by applying Nesterov algorithm [18] to the following smoothed problem $\min\{f_{\bar{\mu}}(x) : Ax = b, \|x\|_2 \leq \beta\}$, where $f_{\bar{\mu}}$ denotes the smooth approximation to the $\|x\|_1$ defined in (2.2), $\bar{\mu} = \frac{\epsilon}{n}$ and the bound β can be set at $\|x_0\|_1 + \epsilon$ such that $x_0 = \text{argmin}\{\|x\|_2 : \ Ax = b\}$ as in SPA (NESTA [4] calls Nesterov's algorithm to solve Π without bounding the feasible region). However, now the Nesterov iterates must satisfy $Ax = b$. This requires a projection; thus, the complexity of each Nesterov update is now $\mathcal{O}(m^2 + n\log(n))$. Thus, the complexity of solving the basis pursuit problem by computing Nesterov [18] updates over the set $\{x : Ax = b, \|x\|_2 \leq \beta\}$ is $\mathcal{O}\left(\sqrt{n}(m^2 + n\log(n))\left(\|x_0\|_1 + \epsilon\right)\epsilon^{-1}\right)$ (see Theorem 3 in [18] for the iteration complexity). Since, in practice, $m = \mathcal{O}(n)$, it follows that SPA is superior to directly applying the Nesterov algorithm [18] to the smoothed basis pursuit problem provided $\epsilon \geq \mathcal{O}(\frac{1}{n^2})$. Since the problem dimension n is typically large, the lower bound on ϵ is quite small. In the next section, we show that SPA is superior to a feasible Nesterov-type algorithms for noisy recovery for all $\epsilon \geq \mathcal{O}(\frac{1}{n^4})$, i.e. for almost all practical instances.

3. Extensions of the SPA to noisy recovery. A simple modification of SPA solves the noisy signal recovery problem

$$\begin{array}{ll} \text{minimize} & \|x\|_1, \\ \text{subject to} & \|Ax - b\|_2 \leq \delta. \end{array} \tag{3.1}$$

To solve this problem we use the exact penalty function

$$\phi(x) = \max\Big\{0, \|Ax - b\|_2 - \delta\Big\} = \max_{0\leq t\leq 1}\Big\{t\big(\|Ax - b\|_2 - \delta\big)\Big\} = \max_{\{(w,t):\|w\|_2\leq t, t\in[0,1]\}}\Big\{w^T(Ax - b)_2 - t\delta\Big\}.$$

Next, we smooth this function to get the smoothed penalty function

$$\phi_\nu(x) = \max_{\{(w,t):\|w\|_2\leq t, t\in[0,1]\}}\Big\{w^T(Ax - b)_2 - t\delta - \frac{\nu}{2}\Big(t^2 + t\left\|\frac{w}{t}\right\|_2^2\Big)\Big\}.$$

Using results in Hoda et al. [14] one can show that the function $h(t, w) = \frac{1}{2}\big(t^2 + t\|\frac{w}{t}\|_2^2\big)$ is strongly convex over the truncated cone $\{(t, w) : \|w\|_2 \leq t, t \in [0, 1]\}$, and consequently, $\phi_\nu(x)$ is a convex function with a Lipschitz continuous gradient. Given the structure of $h(t, w)$, one can rewrite $P_\nu(x)$ as follows:

$$\phi_\nu(x) = \max_{t\in[0,1]}\Big\{t\big(\max_{\|\hat{w}\|_2\leq 1}\big\{\hat{w}^T(Ax - b) - \frac{\nu}{2}\|\hat{w}\|_2^2\big\}\big) - t\delta - \frac{\nu}{2}t^2\Big\}.$$

Recall that the smoothed ℓ_2-penalty function $P_\nu(x) = \max_{\|\hat{w}\|_2\leq 1}\big\{\hat{w}^T(Ax - b) - \frac{1}{2}\|\hat{w}\|_2^2\big\}$ with the optimal $\hat{w}_x$ given by (2.6). Thus,

$$\phi_\nu(x) = \max_{t\in[0,1]}\Big\{t\big(P_\nu(x) - \delta\big) - \frac{\nu}{2}t^2\Big\},$$

with the optimal $t_x = \min\big\{\frac{(P_\nu(x)-\delta)^+}{\nu}, 1\big\}$. Thus, the gradient $\nabla\phi_\nu(x) = t_x\hat{w}_x$, where $\hat{w}_x$ is given by (2.6).

Theorem 2.1, Corollary 2.2, Theorem 2.4 and Theorem 2.5 all remain valid for SPA applied to the penalized objective function $\lambda f_\mu(x) + \phi_\nu(x)$ (see [2]). Thus, SPA efficiently computes a solution for the noisy recovery problem (3.1). Note that unlike NESTA [4], SPA does not require A^TA to be orthogonal projector, i.e. A does not need to have orthonormal rows. Suppose matrix-vector multiplications of the form Ax, A^Ty are fast in that they can be computed in $\mathcal{O}(n\ln(n))$ time; but, A does not have orthogonal rows. For such matrices the complexity of computing a feasible Nesterov update is $\mathcal{O}(n^3)$; thus, comparing $\mathcal{O}(n^{\frac{7}{2}}\epsilon^{-1})$

SMOOTHED PENALTY ALGORITHM $\left(c_\tau^{(0)}, c_\tau^{(1)}, c_\mu, c_\nu, c_\epsilon, c_\lambda\right)$

$q_0 \leftarrow \arg\min\{\|x\|_2 |\ Ax = b\}, \quad x_0 \leftarrow q_0, \quad k \leftarrow 0$

while $(k \geq 0)$

do

$k \leftarrow k + 1$

$Q_k(.) \leftarrow \lambda_k f_{\mu_k}(.) + P_{\nu_k}(.)$

/* Update parameter values */

if $(k == 1)$

then $\lambda_1 \leftarrow \frac{2}{\sqrt{n}}, \ \tau_1 \leftarrow c_\tau^{(0)} \|\nabla Q_1(x_0)\|_2, \ \epsilon_1 \leftarrow c_\epsilon \lambda_1 \|x_0\|_1$

$\mu_1 \leftarrow \frac{\epsilon_1}{(\sqrt{n}\lambda_1+1)\sqrt{n}}, \quad \nu_1 \leftarrow \frac{\epsilon_1}{(\sqrt{n}\lambda_1+1)},$

else $\lambda_k \leftarrow c_\lambda \lambda_{k-1}, \ \tau_k \leftarrow \min\left\{c_\tau^{(1)}\tau_{k-1}, c_\tau^{(0)}\|\nabla Q_k(x_{k-1})\|_2\right\}, \ \epsilon_k \leftarrow c_\epsilon \min\{\epsilon_{k-1}, \eta_{k-1}\}$

$\mu_k \leftarrow \min\left\{c_\mu \mu_{k-1}, \frac{\epsilon_k}{(\sqrt{n}\lambda_k+1)\sqrt{n}}\right\}, \quad \nu_k \leftarrow \min\left\{c_\nu \nu_{k-1}, \frac{\epsilon_k}{\sqrt{n}\lambda_k+1}\right\}$

$L_k \leftarrow \frac{\lambda_k}{\mu_k} + \frac{\|A\|_2^2}{\nu_k}$

/* Start Nesterov Update */

$l \leftarrow 0, \quad x_{k,l} \leftarrow x_{(k-1)}, \quad \beta_k \leftarrow f_{\mu_k}(x_0) + \frac{\mu_1 n}{2}, \quad N_k \leftarrow \left\lfloor \beta_k \sqrt{\frac{8L_k}{\epsilon_k}} \right\rfloor$

compute $\nabla Q_k(x_{k,l})$

while $(\|\nabla Q_k(x_{k,l})\|_2 > \tau_k) \ ||(l < N_k)$

do

1 $y_{k,l} \leftarrow \left(\frac{2}{l+2}\right) \Pi_{\ell_2}\left(\beta_k, \frac{(k+2)(x_{k,l} - \nabla Q_k(x_{k,l})/L_k) - k y_{k,l-1}}{2}\right) + \left(\frac{l}{l+2}\right) y_{k,l-1}$

2 $z_{k,l} \leftarrow \Pi_{\ell_2}\left(\beta_k, x_{k,0} - \frac{\sum_{i=0}^{l}(\frac{i+1}{2})\nabla Q_k(x_{k,i})}{L_k}\right)$

$x_{k,l+1} \leftarrow \left(\frac{2}{l+3}\right) z_{k,l} + \left(\frac{l+1}{l+3}\right) y_{k,l}$

if $\|y_{k,l} - y_{k,l-1}\|_\infty \leq \gamma$

then $x^{sol} \leftarrow y_{k,l}$ **return** x^{sol}

$l \leftarrow l + 1$

compute $\nabla Q_k(x_{k,l})$

/* End Nesterov Updates */

$x_k \leftarrow x_{k,l}$

$\eta_k \leftarrow \lambda_k \|x_k\|_1 + \|Ax_k - b\|_2$

return x_k

FIG. 4.1. *Implementable version of Smoothed Penalty Algorithm (SPA)*

operations for Nesterov-type algorithms with feasible updates (see the paragraph below Theorem 2.5) with $\mathcal{O}(n^{\frac{3}{2}}\epsilon^{-\frac{3}{2}})$ operations for SPA, it follows that the complexity bound for SPA is superior to Nesterov-type algorithms that compute feasible iterates as long as $\epsilon \geq \mathcal{O}(\frac{1}{n^4})$.

The analysis in this section can be extended to solve noisy recovery problems $\min\{\|x\|_1 : \|Ax - b\|_1 \leq \delta\}$ and $\min\{\|x\|_1 : \|Ax - b\|_\infty \leq \delta\}$ [2]. These formulations are interesting when the measurement noise has a Laplacian or Extreme Value distributions.

4. Implementation details of Algorithm SPA. In this section we describe SPA in full detail which can be implemented. This detailed version of SPA is shown in Figure 4.1.

4.1. Bounds on iterates and modified Nesterov updates. Recall that for the k-th subproblem iterate x_k does not need to be ϵ_k-optimal, Nesterov update steps can be terminated early when $\|\nabla Q_k(x_k)\| \leq \tau_k$ for some τ_k such that $\tau_k/\lambda_k \to 0$, provided we can ensure that the iterates are uniformly bounded. We found that in practice terminating the Nesterov updates using the gradient condition was significantly faster.

Let $q_0 = \text{argmin}\{\|x\|_2 : Ax = b\} = A^T(AA^T)^{-1}b$. An analysis similar to that in the proof of Theorem 2.1 establishes that $\|x_k^*\|_2 \leq f_{\mu_k}(q_0) + (\mu_1 n)/2$. Thus, when A has orthonormal rows, $q_0 = A^T b$ and the bound

$f_{\mu_k}(q_0) + (\mu_1 n)/2$ can be computed efficiently. When A does not have orthonormal rows, the complexity of computing $q_0 = A^T(AA^T)^{-1}b$, and the bound $f_{\mu_k}(q_0) + (\mu_1 n)/2$ is $\mathcal{O}(m^2 n)$. Alternatively, we note that

$$f_{\mu_k}(q_0) \leq \|q_0\|_1 \leq \sqrt{n}\ \|q_0\|_2 \leq \sqrt{n}\ \|A^T(AA^T)^{-1}\|_2 \|b\|_2 \leq \frac{\sqrt{n}}{\sigma_{min(A)}}\|b\|_2. \tag{4.1}$$

Thus, one can restrict the iterates to the set $\left\{x \in \mathbb{R}^n : \|x\|_2 \leq \frac{\sqrt{n}}{\hat{\sigma}_{min(A)}}\|b\|_2 + (\mu_1 n)/2\right\}$, where $\hat{\sigma}_{\min}(A)$ is an estimate for $\sigma_{\min}(A)$. In our numerical experiments we found that it was computationally more efficient to compute the latter bound.

We compute the SPA iterates $\{x_k : k \geq 1\}$ using a slightly modified version of Nesterov's optimal algorithm for simple sets [17, 18]. We solve the k-th subproblem

$$\begin{array}{rl} \text{minimize} & Q_k(x), \\ \text{subject to} & \|x\|_2 \leq \beta_k, \end{array}$$

by iteratively computing three sets of iterates$\{(x_{k,l}, y_{k,l}, z_{k,l}) : l \geq 0\}$:

1. $y_{k,l}$ iterate in Step 1 of Figure 4.1 is computed using $x_{k,l}$, $y_{k,l-1}$ and the gradient $\nabla Q_k(x_{k,l})$:

$$\begin{aligned} y_{k,l} &= \text{argmin}\left\{\frac{L_k}{2}\|y - x_{k,l}\|_2^2 + \nabla Q_k(x_{k,l})^T y :\ y \in \left(\frac{2}{l+2}\right) S_k + \left(\frac{l}{l+2}\right) y_{k,l-1}\right\}, \\ &= \left(\frac{2}{l+2}\right)\ \Pi_{\ell_2}\left(\beta_k,\ \left(\frac{l+2}{2}\right)\left(x_{k,l} - \frac{1}{L_k}\nabla Q_k(x_{k,l})\right) - \left(\frac{l}{2}\right) y_{k,l-1}\right) + \left(\frac{l}{l+2}\right) y_{k,l-1}, \end{aligned}$$

   where $S_k = \{y \in \mathbb{R}^n :\ \|y\|_2 \leq \beta_k\}$ and the projection $\Pi_{\ell_2}(\beta, \hat{y}) = \text{argmin}\{\|y - \hat{y}\|_2^2 : \|y\|_2 \leq \beta\}$. The projection Π_{ℓ_2} can be computed with a $\mathcal{O}(n)$ worst case complexity. This update scheme is *not* the standard Nesterov y-update [18]; however, using the last paragraph of Lemma 1 in [18], one can show that this is a valid and possibly an improved update.
2. The $z_{k,l}$ iterate in Step 2 is computed using the initial point $x_{k,0}$ and the gradients $\nabla Q_k(x_{k,i})$ for all the iterates $i \leq l$:

$$\begin{aligned} z_{k,l} &= \text{argmin}\left\{\sum_{i=0}^{l}\Big(\frac{i+1}{2}\Big)\nabla Q_k(x_{k,i})^T z + \frac{L}{2}\|z - x_{k,0}\|_2^2 : \|z\|_2 \leq \beta_k\right\}, \\ &= \Pi_{\ell_2}\left(\beta_k, x_{k,0} - \frac{1}{L}\sum_{i=0}^{n}\left(\frac{i+1}{2}\right)\nabla Q_k(x_{k,i})\right). \end{aligned}$$

   Note that in the k-th sub-problem, we use the *prox* function $h(x) = \frac{1}{2}\|x - x_{k,0}\|_2^2$ to compute the iterates $\{z_{k,l}\}$.
3. The $\{x_{k,l} : l \geq 0\}$ iterates are computed by initializing $x_{k,0} = x_{k-1}$, the previous SPA iterate, and setting $x_{k,l} = \Big(\frac{2}{l+3}\Big) z_{k,l-1} + \Big(\frac{l+1}{l+3}\Big) y_{k,l-1}$, for all $l \geq 1$.

We terminate the iterations when either the gradient condition $\|\nabla Q(x_{k,l})\|_2 \leq \tau_k$ or the inner iteration counter $l > N_k$ which guarantees that the iterate $x_{k,l}$ is ϵ_k-optimal. In all of our numerical experiments, the Nesterov updates were always terminated when $\|\nabla Q(x_{k,l})\|_2 \leq \tau_k$, i.e. the gradient termination condition was always satisfied before the ϵ-optimality termination condition.

4.2. Stopping criterion for SPA. We terminate SPA when the ℓ_∞ difference between successive inner iterates are below a threshold γ, i.e. $\|y_{k,l} - y_{k,l-1}\|_\infty \leq \gamma$ for any $k \geq 1$. In our numerical experiments we set γ by experimenting with a small instance of the problem.

4.3. Multiplier selection. The approximate optimality parameter τ_k is set as follows:

$$\begin{array}{rcl} \tau_1 & \leftarrow & c_\tau^{(0)}\|\nabla Q_1(x_0)\|_2, \\ \tau_{k+1} & \leftarrow & \min\left\{c_\tau^{(1)}\tau_k, c_\tau^{(0)}\|\nabla Q_{k+1}(x_k)\|_2\right\}, \quad \text{for all } k \geq 1. \end{array}$$

Guided by the scaling result implicit in Theorem 2.5 we set $\lambda_1 = \frac{2}{\sqrt{n}}$, and then set $\lambda_k = c_\lambda \lambda_{k-1}$ for all $k > 1$.

We arbitrarily set initial $\epsilon_0 = \eta_0 = \lambda_1 \|x_0\|_1$. In the k-th iteration of SPA, we solve a smoothed version of the penalized optimization problem

$$\begin{array}{rl} \text{minimize} & \lambda_k \|x\|_1 + \|Ax - b\|_2, \\ \text{subject to} & \|x\|_2 \leq \beta_k \end{array} \tag{4.2}$$

The dual of (4.2) is given by

$$\begin{array}{rl} \text{maximize} & -b^T w - \beta_k \|A^T w + \lambda_k u\|_2, \\ \text{subject to} & \|u\|_\infty \leq 1, \\ & \|w\|_2 \leq 1. \end{array} \tag{4.3}$$

We use the duality gap of the primal iterates x_k to update the multiplier sequence $\{\epsilon_k\}_{k \in \mathbb{Z}_+}$. For $k \geq 1$, let η_k denote an upper bound on duality gap of x_k iterate computed by approximately solving the k-th sub-problem. Then η_k can be set in either of the following two update methods:

(i) Since $(u_k, v_k) = (0, 0)$ is feasible for the dual problem (4.3) for all $k \geq 1$, $\eta_k = \lambda_k \|x_k\|_1 + \|Ax_k - b\|_2$ is a valid upper bound on the duality gap of the iterate x_k.

(ii) Nesterov's non-smooth optimization algorithm [18] returns an approximately optimal dual $(\hat{u}_k, \hat{w}_k)$. Thus, $\eta_k = \lambda_k \|x_k\|_1 + \|Ax_k - b\|_2 + b^T \hat{w}_k + \beta_k \|A^T \hat{w}_k + \lambda_k \hat{u}_k\|_2$ is a valid upper bound on the duality gap. Note that η_k can be computed efficiently using *one* additional matrix-vector multiplication of the form $A^T y$.

For the numerical results reported in Section 5, we we take η_k to be $\lambda_k \|x_k\|_1 + \|Ax_k - b\|_2$.

$$\epsilon_k \leftarrow c_\epsilon \ \min\{\epsilon_{k-1}, \eta_{k-1}\}, \qquad \text{for all } k \geq 1,$$

is set as the target approximation error for the next subproblem. The Nesterov non-smooth optimization algorithm then dictates that the smoothing parameter (μ, ν) should be set to $\mu = \frac{\epsilon_k}{(\sqrt{n}\lambda_k + 1)\sqrt{n}}$ and $\nu = \frac{\epsilon_k}{\sqrt{n}\lambda_k + 1}$ in order to minimize the number of Nesterov updates required to compute a ϵ_k-optimal solution. Since we require that μ_k and ν_k be monotonically decreasing we modify this parameter update as follows:

$$\begin{array}{rcll} \mu_k & \leftarrow & \min\left\{c_\mu \mu_{k-1}, \frac{\epsilon_k}{(\sqrt{n}\lambda_k + 1)\sqrt{n}}\right\}, & k \geq 1, \quad \mu_0 = \infty, \\ \nu_k & \leftarrow & \min\left\{c_\nu \nu_{k-1}, \frac{\epsilon_k}{\sqrt{n}\lambda_k + 1}\right\}, & k \geq 1, \quad \nu_0 = \infty. \end{array}$$

We "tune" the constants $(c_\tau^{(0)}, c_\tau^{(1)}, c_\mu, c_\nu, c_\epsilon, c_\lambda)$ on the smallest $n = 64 \times 64$ problem and then used the values for all the other problems.

Note that we require that the conditions of Theorem 2.1 hold, i.e. $\frac{\epsilon_k}{\lambda_k} \to 0$ and $\frac{\tau_k}{\lambda_k} \to 0$, the condition for Theorem 2.4 to hold, i.e. $\tau_k \leq \sqrt{2L_k \epsilon_k}$ for all $k \geq 1$, is not imposed. This is because in our numerical experiments we found that for moderate values for the coefficient $c_\tau^{(1)}$ the gradient stopping condition was satisfied before the ϵ-optimality stopping condition, and therefore, we do not require $\tau_k \leq \sqrt{2L_k \epsilon_k}$ in practice.

Since the parameter sequence $\{(\mu_k, \nu_k, \lambda_k) : k \geq 1\}$ follow the scaling in Theorem 2.5, we should expect that the optimal choice of initial multipliers $(\mu_0, \nu_0, \lambda_0)$ should be independent of n for a given measurement ratio m/n and sparsity ratio s/n. In our numerical experiments we found this to be approximately true. We exploit this fact by tuning the parameters $(\mu_0, \nu_0, \lambda_0)$ on the smallest problem (with $n = 64 \times 64$) and then use these parameters for all larger problems.

5. Numerical experiments. We conducted two sets of numerical experiments with SPA. The goal in the first set of experiments was to investigate how the complexity of SPA grows with the problem dimension. The second set of experiments compares the performance of SPA with another Nesterov-type algorithm NESTA [4] and a fixed point continuation algorithm FPC [12, 13]. All the numerical experiments were conducted on an IBM Thinkpad laptop with a Intel Core 2 CPU T7200 @2.0 GHz processor, 3GB SDRAM running MATLAB 7.2 on Windows XP Professional operating system.

5.1. Experimental setup. We tested SPA on randomly generated target signals. The target signal $x^* \in \mathbb{R}^n$ was chosen to be s-sparse, i.e. exactly s out of n components were nonzero. Following the experimental setup in a recent paper of Becker, et al. [4] we set

$$x^*(i) = \mathbf{1}(i \in \Lambda)\ \eta_1(i) 10^{5\eta_2(i)} \tag{5.1}$$

where

(i) the set Λ was constructed by randomly selecting s indices from the set $\{1, \ldots, n\}$,
(ii) $\eta_1(i)$, $i \in \Lambda$, were independently, and identically distributed Bernoulli random variables taking values $+1$ or -1 with equal probability,
(iii) $\eta_2(i)$, $i \in \Lambda$, were independently, and identically distributed uniform$[0, 1]$ random variables.

The signals x^* were created in this manner have a dynamic range of 100dB.

The measurement matrix A and the measurement vector b were constructed as follows. We randomly selected $m = \frac{n}{4}$ frequencies from the set $\{0, \ldots, n\}$. Let $A \in \mathbb{R}^{m \times n}$ denote a $m \times n$ partial Discrete Cosine matrix constructed from these randomly selected frequencies and $b = Ax^*$ denote the Discrete Cosine transform of the signal x^* evaluated at the chosen frequencies.

We found that for a fixed measurement ratio m/n, sparsity ratio s/n, and the accuracy tolerance γ, the total number of Nesterov updates is effectively independent of the dimension n of the target signal. In our experiment we exploit this empirical result by first tuning the constants controlling the parameter updates for a smallest sized problem and subsequently using these fixed parameters for solving all larger problems. In our numerical experiments the constants controlling the parameter update were set as follows:

$$c_\tau^{(0)} = 0.2, \quad c_\tau^{(1)} = 0.855, \quad c_\mu = 0.1, \quad c_\nu = 0.1, \quad c_\epsilon = 0.8, \quad c_\lambda = 0.9. \tag{5.2}$$

For the numerical results reported in this section we only used *one* projection at the beginning of the algorithm to uniformly bound the iterates x_k using $q_0 = \operatorname{argmin}\{\|x\|_2 :\ Ax = b\} = A^T b$.

5.2. Algorithm scaling results. We tested the algorithm for s-sparse signals with

(i) three different sizes: small $n = 64 \times 64$, medium $n = 256 \times 256$, and large $n = 512 \times 512$,
(ii) two sparsity levels: high $s = \lceil n/400 \rceil$, and low $s = \lceil n/40 \rceil$.

In order to assess the convergence properties of the SPA we replaced the stopping criterion $\|y_{k,l} - y_{k,l-1}\|_\infty \leq \gamma$ with

$$\|y_{k,l} - x^*\|_\infty \leq \gamma. \tag{5.3}$$

We report results for $\gamma = 1$, 10^{-1} and 10^{-2}. The signal model in (5.1) and the stopping criterion implies that the algorithm produces x_k with $5 + \log_{10}(1/\gamma)$ digits of accuracy. Note that the stopping criterion (5.3) is only used to test the convergence properties the algorithm in this simulation study.

The Table 5.1 summarizes the sparsity conditions and the parameter settings that were investigated in the numerical 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 $s = n/40$ and $\gamma = 0.1$ are displayed in Table 5.4. In Tables 5.2– 5.7, the row labeled `Update Iter.` lists the total number of Nesterov update iterations during the course of SPA, the row labeled `SPA Iter.` lists the number of SPA iterations, the row labeled `nMat` lists the number of matrix-vector multiplications required to compute the Nesterov updates. All other rows are self-explanatory. We generated $N = 10$ random instances for each of the experimental conditions. The column labeled `average` lists the average taken over the $N = 10$ random instances, the columns labeled `min` (resp. `max`) list the minimum (resp. maximum) over the 10 instances.

The experiment results support the following conclusions:

(a) SPA is very efficient in computing a solution to (1.1) – the algorithm requires anywhere from 5 to 8 iterations to converge.
(b) For a given sparsity type (high or low) and stopping criterion γ, the total number of SPA iterations and the number of Nesterov updates is a very slowly growing function of the dimension n of the target signal.
(c) The number of Nesterov updates (and also the overall running time of SPA) increases with the number of non-zero elements in the target signal x^*. Increasing the number of non-zero elements from $s = n/400$ to $s = n/40$ nearly doubled the total number of Nesterov update iterations.

As remarked in Section 4.3 we used the fixed set of constants in (5.2) to update the parameter sequence for all the experiments.

Sparsity	γ	Table
$s = n/400$	1	Table 5.3
$s = n/400$	0.1	Table 5.5
$s = n/400$	0.01	Table 5.7
$s = n/40$	1	Table 5.2
$s = n/40$	0.1	Table 5.4
$s = n/40$	0.01	Table 5.6

TABLE 5.1
Summary of numerical experiments

5.3. Comparison of SPA with other solvers. In this section we report the results of our numerical experiments comparing SPA with NESTAv1.1 [4] and FPCv2.0 [13]. We considered two sets of problems. In the first set the measurement matrix A was a partial DCT and in the second set A was a random Gaussian matrix. The experimental results for the case where A is a partial DCT matrix and for the case where A is a random Gaussian matrix, are reported in Table 5.8 and Table 5.9, respectively. In Tables 5.8–5.9, the row labeled `Update Iter.` lists the total number of Nesterov update iterations in the columns corresponding to SPA and NESTA, and the total number of shrinkage iterations in the column corresponding to FPC; the row labeled `nMat` lists the number of matrix-vector products computed during the course of the three algorithms. All other rows are self-explanatory.

5.3.1. DFT Case. In the first set of problems the matrix A was a partial DCT matrix and we created 10 random problems of size $n = 512 \times 512$ using the procedure described in Section 5.1.

We chose parameter values for each of the three algorithms so that they produced a solution x^{sol} with ℓ_∞-error approximately equal to 5×10^{-4}, i.e. $\|x^{sol} - x^*\|_\infty \approx 5 \times 10^{-4}$. We set the parameter values for each algorithm by solving a set of small size problems and these parameter values were fixed throughout the experiments.

1. For SPA, we set $\gamma = 5 \times 10^{-5}$.
2. NESTA solves $\min_{\|Ax-b\|_2 \le \sigma} f_\mu(x)$, where $f_\mu(x) = \max_{\{u:\|u\|_\infty \le 1\}} \left\{ x^T u - \frac{\mu}{2}\|u\|_2^2 \right\}$, using continuation on μ. When σ is set to 0, NESTA handles $\|Ax - b\|_2 \le \sigma$ constraint as $Ax = b$ and since $AA^T = I$ is assumed, projections on to $\{x \in \mathbb{R}^n : Ax = b\}$ affine space can be done efficiently. NESTA stops when $\frac{|f_\mu(x_k) - \bar{f}_\mu(x_k)|}{\bar{f}_\mu(x_k)} < \delta$, for some $\delta > 0$, where $\bar{f}_\mu(x_k) = \frac{1}{min\{10,k\}} \sum_{\ell=1}^{min\{10,k\}} f_\mu(x_{k-\ell})$. For NESTA, we set $\mu = 1 \times 10^{-4}$ and $\delta = 1 \times 10^{-10}$.
3. FPC solves $\min_{x \in \mathbb{R}^n} \|x\|_1 + \frac{1}{\lambda}\|Ax - b\|_2^2$. For FPC, we set $\frac{1}{\lambda} = 1.5 \times 10^4$.

As in Figure 4.1, we first compute a projection $q_0 = A^T b$ to bound the iterate sequence $\{x_k\}_{k \in \mathbb{Z}_+}$ and we start SPA with $x_0 = q_0$. For the k-th SPA iteration, the simple set $\{x \in \mathbb{R}^n : \|x\|_2 \le \|q_0\|_1 + \frac{\mu_1 n}{2}\}$ was chosen to bound the Nesterov iterates.

The experimental results for the case where A is a partial DCT matrix are reported in Table 5.8. The results show that all three algorithms produce similar results with comparable running times. While SPA and NESTA required approximately the same number of matrix-vector products, FPC required far fewer matrix-vector products to produce a result of similar ℓ_∞-error.

5.3.2. Gaussian Case. We created 10 random problems of size $n = 120 \times 120$ (for $n > 120^2$ the problem did not fit into the memory of the computer that we used for our experiments). The target signal are created using (5.1) described in Section 5.1 and the each element A_{ij} of the measurement matrix $A \in \mathbb{R}^{m \times n}$ was drawn independently and identically according to a univariate standard Gaussian distribution. As in the DFT case, we chose parameter values for each of the three algorithms so that they produced a solution x^{sol} with ℓ_∞-error approximately equal to 5×10^{-4}, i.e. $\|x^{sol} - x^*\|_\infty \approx 5 \times 10^{-4}$. We set the parameter values for each algorithm by solving a set of small size problems and these parameter values were fixed throughout the experiments. For SPA, we set $\gamma = 2 \times 10^{-5}$. For NESTA, we set $\mu = 1 \times 10^{-4}$ and $\delta = 1 \times 10^{-10}$. For FPC, we set $\frac{1}{\lambda} = 5 \times 10^4$.

For this set of experiments we did not compute the costly projection $q_0 = A^T(AA^T)^{-1}b$ to initialize the algorithm and to bound the iterate sequence $\{x_k\}_{k \in \mathbb{Z}_+}$. We instead used the alternative bound described in

ESTIMATION PROCEDURE (A)

input: constant multipliers $c1 > 1,\ c2 > 1$
$k \leftarrow 1$, $\hat{\sigma}_{min}(A) \leftarrow \infty$, $\hat{\sigma}_{max}(A) \leftarrow -\infty$
while $(k < 40)$
do
$k \leftarrow k + 1$
$y = randn(m, 1)$, $x \leftarrow x/\|x\|_2$
if $\|A^T y\|_2 < \hat{\sigma}_{min}(A)$
then $\hat{\sigma}_{min}(A) \leftarrow \|A^T y\|_2$
if $\|A^T y\|_2 > \hat{\sigma}_{max}(A)$
then $\hat{\sigma}_{max}(A) \leftarrow \|A^T y\|_2$
$\hat{\sigma}_{min}(A) \leftarrow \hat{\sigma}_{min}(A)/c1$, $\hat{\sigma}_{max}(A) \leftarrow \hat{\sigma}_{max}(A) \cdot c2$
return $\hat{\sigma}_{min}(A)$ and $\hat{\sigma}_{max}(A)$

FIG. 5.1. *Procedure for estimating $\sigma_{min}(A)$ and $\sigma_{max}(A)$*

Section 4.1. For the k-th SPA iteration, we restricted the iterates to the the simple set $\left\{x \in \mathbb{R}^n : \|x\|_2 \leq \frac{\sqrt{n}}{\hat{\sigma}_{min(A)}}\|b\|_2 + \frac{\mu_1 n}{2}\right\}$, where $\hat{\sigma}_{min}(A)$ was an estimate of $\sigma_{min}(A)$.

We pre-processed the problem data before using SPA to solve the problem. We rescaled the constraints by setting $(\hat{A}, \hat{b}) = \hat{\sigma}^{-1}_{\max}(A)(A, b)$, where $\hat{\sigma}_{max}(A)$ is an estimate of $\sigma_{max}(A)$. We set the initial iterate x_0 in SPA to $x_0 = (\hat{\sigma}_{max}(A))^{-2}A^T b$ – note that x_0 is *not* a projection on to $\{x : \hat{A}x = \hat{b}\}$ when A is Gaussian. The estimate $\hat{\sigma}_{max}(A)$ and $\hat{\sigma}_{min}(A)$ were computed using the procedure described in Figure 5.1. FPC also by default rescales the problem by setting $(\hat{A}, \hat{b}) = \sigma^{-1}_{\max}(A)(A, b)$.

The results for the case where A is a Gaussian matrix are reported in Table 5.9. The row labeled `Preprocessing Time` lists the elapsed time during the estimation procedure at the beginning of SPA and the time needed for FPC to rescale the problem; NESTA does not to any preprocessing, hence we report it as 0. The row labeled `Algorithm Time` lists the time needed for all three algorithms to terminate on the pre-processed input, and the row labeled `Total Time` lists the total time that is the sum of the pre-processing time and the algorithm time.

The experimental results show that while the running times for SPA and FPC produce similar results is comparable; the running time for NESTA is considerably larger. In each iteration of SPA and FPC, the most time consuming steps are one matrix-vector product of the form Ax and one of the form $A^T y$. On the other hand, in every iteration of NESTA one has to, in addition, project a vector z on to the feasible set $\{x \in \mathbb{R}^n :\ Ax = b\}$, i.e. compute $A^T(AA^T)^{-1}(b - Az)$. The projection operator $(AA^T)^{-1}$ is computed once at the beginning of the algorithm (note that we add the time required to compute $(AA^T)^{-1}$ to the algorithm time for NESTA). Therefore, in each iteration NESTA incurs an additional $\mathcal{O}(m^2)$ computational cost due to multiplication with $(AA^T)^{-1}$; consequently, although all three algorithms do compute approximately the same number of matrix-vector products, the running time of NESTA to produce a result of similar ℓ_∞-error is larger.

6. Conclusion. We propose a smoothed penalty algorithm (SPA) for the sparse recovery problem. The SPA recovers the target signal by solving a sequence of smoothed penalized sub-problems, and each sub-problem is solved using Nesterov's optimal method for simple sets [17, 18]. We show that the continuation scheme used in SPA provably converges to the target signal and we are also able to compute a convergence rate. Since we penalize infeasibility by the exact penalty function $\|Ax - b\|$, where $\|\cdot\|$ can be ℓ_1, ℓ_2 or ℓ_∞ norm, an accurate solution is obtained before penalty parameter takes on arbitrarily small value; consequently, our proposed algorithm is numerically stable. We found that for a fixed measurement ratio m/n, sparsity ratio s/n, and solution accuracy γ, the total number of Nesterov iterations is effectively independent of the dimension n of the target signal; thus, one can tune the parameters on the smallest problem and use these parameters for all larger problems. The numerical results reported in this paper show that SPA required very few iterations to accurately recover the target signal.

SPA is a very general algorithmic framework that can be used for ℓ_1-minimization, relaxed ℓ_1-

minimization, ℓ_1-minimization problems with linear side constraints, and also for convex optimization problems of the form

$$\min_{X=[x_1,\ldots,x_q]} \Big\{ \sum_{i\neq j} \|x_i - x_j\|_1 + \sum_{i=1}^{q} f(x_i) \Big\}, \tag{6.1}$$

that arise in the context of maximum likelihood estimation for sparse graphical networks. The cost of this flexibility is that SPA is not always as efficient as algorithms such as FPC that explicitly utilize the ℓ_1 objective term. In [3] we propose a new augmented Lagrangian based algorithm that explicitly uses the ℓ_1 structure and is competitive with other specialized algorithms for ℓ_1-minimization.

TABLE 5.2
Experiment Results for $m = n/4$, $s = m/10$ and $\|x^{sol} - x^*\|_\infty \leq 1$

	n=512×512			**n=256×256**			**n=64×64**		
	Average	**Min**	**Max**	**Average**	**Min**	**Max**	**Average**	**Min**	**Max**
Update Iter. #	181.5	178	186	176.4	170	183	170.7	163	178
$\|\|\mathbf{x^{sol}}\|_1 - \|\mathbf{x^*}\|_1\|/\|\mathbf{x^*}\|_1$	1.48E-05	2.39E-06	2.38E-05	6.87E-06	1.22E-06	2.07E-05	5.83E-06	1.23E-06	2.23E-05
$\max\{\|(\mathbf{x^{sol}})_i - (\mathbf{x^*})_i\| : \|(\mathbf{x^*})_i\| > \mathbf{0}\}$	9.76E-01	9.63E-01	9.97E-01	9.85E-01	9.45E-01	9.99E-01	9.77E-01	9.18E-01	9.98E-01
$\max\{\|(\mathbf{x^{sol}})_i\| : (\mathbf{x^*})_i = \mathbf{0}\}$	3.00E-01	2.10E-01	3.81E-01	1.93E-01	7.80E-02	2.96E-01	1.59E-01	4.95E-02	3.92E-01
$\|\mathbf{Ax^{sol} - b}\|_2$	4.387	2.118	7.464	3.835	1.518	5.215	1.318	0.604	1.684
$\|\mathbf{x^{sol}}\|_1$	56632639.0	55368345.9	59671260.7	14250715.2	13356214.3	15031055.1	1033579.3	836967.3	1289375.6
$\|\mathbf{x^*}\|_1$	56631797.7	55368213.9	59669841.3	14250648.3	13355937.4	15030813.1	1033574.2	836965.1	1289377.9
SPA iter. #	6.0	6.0	6.0	5.9	5.0	6.0	5.5	5.0	6.0
nMat	364.0	357.0	373.0	353.8	341.0	367.0	342.4	327.0	357.0
Total Time	85.2	83.5	87.7	19.2	18.3	20.2	1.1	1.0	1.3

TABLE 5.3
Experiment Results for $m = n/4$, $s = m/100$ and $\|x^{sol} - x^*\|_\infty \leq 1$

	n=512×512			**n=256×256**			**n=64×64**		
	Average	**Min**	**Max**	**Average**	**Min**	**Max**	**Average**	**Min**	**Max**
Update Iter. #	105.8	103	110	106.7	102	115	105.9	101	112
$\|\|\mathbf{x^{sol}}\|_1 - \|\mathbf{x^*}\|_1\|/\|\mathbf{x^*}\|_1$	1.08E-04	6.24E-05	1.55E-04	1.24E-04	7.76E-05	2.12E-04	9.26E-05	4.66E-05	1.59E-04
$\max\{\|(\mathbf{x^{sol}})_i - (\mathbf{x^*})_i\| : \|(\mathbf{x^*})_i\| > \mathbf{0}\}$	8.69E-01	6.87E-01	9.93E-01	8.95E-01	7.97E-01	9.76E-01	9.18E-01	8.00E-01	9.78E-01
$\max\{\|(\mathbf{x^{sol}})_i\| : (\mathbf{x^*})_i = \mathbf{0}\}$	2.14E-01	9.53E-02	3.19E-01	1.83E-01	1.14E-01	3.37E-01	1.25E-01	4.67E-02	2.07E-01
$\|\mathbf{Ax^{sol} - b}\|_2$	2.548	1.787	3.162	1.449	1.200	1.743	0.549	0.460	0.620
$\|\mathbf{x^{sol}}\|_1$	5588836.6	4423908.1	7001002.3	1508201.7	1171960.9	1838355.4	193844.3	108723.0	311461.7
$\|\mathbf{x^*}\|_1$	5588239.3	4423488.0	7000565.2	1508021.0	1171713.0	1838194.7	193828.2	108705.6	311447.1
SPA iter. #	5.0	5.0	5.0	5.0	5.0	5.0	5.1	5.0	6.0
nMat	212.6	207.0	221.0	214.4	205.0	231.0	212.8	203.0	225.0
Total Time	49.7	48.4	51.5	11.7	11.0	12.7	0.7	0.6	0.9

TABLE 5.4
Experiment Results for $m = n/4$, $s = m/10$ and $\|x^{sol} - x^*\|_\infty \leq 1 \times 10^{-1}$

	n=512×512			**n=256×256**			**n=64×64**		
	Average	**Min**	**Max**	**Average**	**Min**	**Max**	**Average**	**Min**	**Max**
Update Iter. #	209.2	207	210	208.2	206	210	206.0	200	212
$\|\|\mathbf{x^{sol}}\|_1 - \|\mathbf{x^*}\|_1 / \|\mathbf{x^*}\|_1$	4.47E-07	3.69E-07	5.23E-07	4.99E-07	4.11E-07	5.75E-07	5.35E-07	4.25E-07	6.92E-07
$\max\{\|(\mathbf{x^{sol}})_i - (\mathbf{x^*})_i\| : \|(\mathbf{x^*})_i\| > \mathbf{0}\}$	9.28E-02	8.31E-02	9.84E-02	9.45E-02	8.82E-02	9.91E-02	9.10E-02	8.27E-02	9.70E-02
$\max\{\|(\mathbf{x^{sol}})_i\| : (\mathbf{x^*})_i = \mathbf{0}\}$	2.09E-02	9.23E-03	3.65E-02	2.22E-02	1.28E-02	3.13E-02	1.52E-02	5.48E-03	2.90E-02
$\|\mathbf{A x^{sol}} - \mathbf{b}\|_2$	0.966	0.914	1.024	0.508	0.487	0.539	0.145	0.129	0.157
$\|\mathbf{x^{sol}}\|_1$	56631823.0	55368236.0	59669868.3	14250655.4	13355945.0	15030820.7	1033574.8	836965.6	1289378.5
$\|\mathbf{x^*}\|_1$	56631797.7	55368213.9	59669841.3	14250648.3	13355937.4	15030813.1	1033574.2	836965.1	1289377.9
SPA iter. #	6.0	6.0	6.0	6.0	6.0	6.0	6.0	6.0	6.0
nMat	419.4	415.0	421.0	417.4	413.0	421.0	413.0	401.0	425.0
Total Time	98.3	97.3	99.1	22.7	22.2	23.2	1.3	1.2	1.5

TABLE 5.5
Experiment Results for $m = n/4$, $s = m/100$ and $\|x^{sol} - x^*\|_\infty \leq 1 \times 10^{-1}$

	n=512×512			**n=256×256**			**n=64×64**		
	Average	**Min**	**Max**	**Average**	**Min**	**Max**	**Average**	**Min**	**Max**
Update Iter. #	123.5	118	128	124.4	117	131	124.0	115	132
$\|\|\mathbf{x^{sol}}\|_1 - \|\mathbf{x^*}\|_1 / \|\mathbf{x^*}\|_1$	1.36E-05	1.01E-05	2.00E-05	1.49E-05	1.02E-05	2.08E-05	1.02E-05	5.73E-06	1.77E-05
$\max\{\|(\mathbf{x^{sol}})_i - (\mathbf{x^*})_i\| : \|(\mathbf{x^*})_i\| > \mathbf{0}\}$	8.64E-02	7.68E-02	9.43E-02	9.09E-02	8.05E-02	9.95E-02	8.78E-02	7.49E-02	9.86E-02
$\max\{\|(\mathbf{x^{sol}})_i\| : (\mathbf{x^*})_i = \mathbf{0}\}$	2.09E-02	4.87E-03	3.03E-02	2.66E-02	4.53E-03	6.03E-02	1.62E-02	6.15E-03	2.53E-02
$\|\mathbf{A x^{sol}} - \mathbf{b}\|_2$	0.265	0.238	0.312	0.146	0.114	0.195	0.053	0.050	0.059
$\|\mathbf{x^{sol}}\|_1$	5588313.8	4423576.4	7000636.1	1508043.1	1171729.1	1838213.4	193830.0	108707.6	311448.9
$\|\mathbf{x^*}\|_1$	5588239.3	4423488.0	7000565.2	1508021.0	1171713.0	1838194.7	193828.2	108705.6	311447.1
SPA iter. #	6.0	6.0	6.0	6.0	6.0	6.0	6.0	6.0	6.0
nMat	248.0	237.0	257.0	249.8	235.0	263.0	249.0	231.0	265.0
Total Time	58.1	55.2	60.0	13.6	12.6	14.5	0.8	0.7	1.1

TABLE 5.6
Experiment Results for $m = n/4$, $s = m/10$ and $\|x^{sol} - x^*\|_\infty \leq 1 \times 10^{-2}$

	n=512×512			**n=256×256**			**n=64×64**		
	Average	**Min**	**Max**	**Average**	**Min**	**Max**	**Average**	**Min**	**Max**
Update Iter. #	247.7	245	250	246.3	243	249	243.8	239	251
$\|\|\mathbf{x^{sol}}\|_1 - \|\mathbf{x^*}\|_1\|/\|\mathbf{x^*}\|_1$	6.84E-08	6.40E-08	7.66E-08	7.68E-08	6.19E-08	9.47E-08	8.59E-08	6.37E-08	1.00E-07
$\max\{\|(\mathbf{x^{sol}})_i - (\mathbf{x^*})_i\| : \|(\mathbf{x^*})_i\| > \mathbf{0}\}$	9.31E-03	8.79E-03	9.96E-03	9.75E-03	8.84E-03	9.99E-03	9.40E-03	8.57E-03	9.93E-03
$\max\{\|(\mathbf{x^{sol}})_i\| : (\mathbf{x^*})_i = \mathbf{0}\}$	1.95E-03	1.43E-03	2.84E-03	2.37E-03	1.18E-03	4.45E-03	1.95E-03	5.61E-04	3.07E-03
$\|\mathbf{Ax^{sol} - b}\|_2$	0.086	0.081	0.090	0.046	0.041	0.049	0.013	0.011	0.014
$\|\mathbf{x^{sol}}\|_1$	56631801.6	55368217.6	59669845.9	14250649.4	13355938.3	15030814.2	1033574.3	836965.2	1289378.0
$\|\mathbf{x^*}\|_1$	56631797.7	55368213.9	59669841.3	14250648.3	13355937.4	15030813.1	1033574.2	836965.1	1289377.9
SPA iter. #	7.1	7.0	8.0	7.0	7.0	7.0	7.1	7.0	8.0
nMat	496.4	491.0	501.0	493.6	487.0	499.0	488.6	479.0	503.0
Total Time	116.4	115.5	117.4	26.8	26.2	27.3	1.5	1.4	1.7

TABLE 5.7
Experiment Results for $m = n/4$, $s = m/100$ and $\|x^{sol} - x^*\|_\infty \leq 1 \times 10^{-2}$

	n=512×512			**n=256×256**			**n=64×64**		
	Average	**Min**	**Max**	**Average**	**Min**	**Max**	**Average**	**Min**	**Max**
Update Iter. #	141.0	136	145	138.8	128	150	140.4	125	154
$\|\|\mathbf{x^{sol}}\|_1 - \|\mathbf{x^*}\|_1\|/\|\mathbf{x^*}\|_1$	1.40E-06	9.15E-07	2.25E-06	1.59E-06	8.40E-07	2.28E-06	1.04E-06	4.84E-07	1.62E-06
$\max\{\|(\mathbf{x^{sol}})_i - (\mathbf{x^*})_i\| : \|(\mathbf{x^*})_i\| > \mathbf{0}\}$	9.09E-03	7.13E-03	9.73E-03	8.95E-03	7.24E-03	9.98E-03	9.13E-03	8.22E-03	9.87E-03
$\max\{\|(\mathbf{x^{sol}})_i\| : (\mathbf{x^*})_i = \mathbf{0}\}$	1.62E-03	6.15E-04	2.60E-03	2.39E-03	8.00E-04	5.54E-03	1.71E-03	2.05E-04	7.03E-03
$\|\mathbf{Ax^{sol} - b}\|_2$	0.023	0.014	0.033	0.015	0.010	0.024	0.005	0.004	0.007
$\|\mathbf{x^{sol}}\|_1$	5588246.9	4423498.0	7000572.2	1508023.3	1171714.9	1838196.3	193828.4	108705.8	311447.3
$\|\mathbf{x^*}\|_1$	5588239.3	4423488.0	7000565.2	1508021.0	1171713.0	1838194.7	193828.2	108705.6	311447.1
SPA iter. #	7.0	7.0	7.0	6.7	6.0	7.0	6.9	6.0	7.0
nMat	283.0	273.0	291.0	278.6	257.0	301.0	281.8	251.0	309.0
Total Time	66.3	63.8	68.2	15.2	13.9	16.5	0.9	0.7	1.1

TABLE 5.8
DFT Experiment Results for $n = 512 \times 512$, $m = n/4$, $s = m/10$

	SPA			**NESTA**			**FPC**		
	Average	**Min**	**Max**	**Average**	**Min**	**Max**	**Average**	**Min**	**Max**
Update Iter. #	291.1	290	293	314.2	313	316	190	189	192
$\|\|\mathbf{x^{sol}}\|_1 - \|\mathbf{x^*}\|_1\|/\|\mathbf{x^*}\|_1$	1.03E-08	1.01E-08	1.07E-08	6.55E-08	6.20E-08	6.68E-08	3.46E-08	3.27E-08	3.54E-08
$\max\{\|(\mathbf{x^{sol}})_i - (\mathbf{x^*})_i\| : \|(\mathbf{x^*})_i\| > \mathbf{0}\}$	5.96E-04	5.49E-04	6.82E-04	7.45E-04	7.03E-04	8.36E-04	6.79E-04	6.31E-04	7.34E-04
$\max\{\|(\mathbf{x^{sol}})_i\| : (\mathbf{x^*})_i = \mathbf{0}\}$	6.57E-05	6.06E-05	7.05E-05	2.32E-04	2.00E-04	3.11E-04	1.57E-04	1.16E-04	1.94E-04
$\|\mathbf{Ax^{sol}} - \mathbf{b}\|_2$	5.99E-03	5.75E-03	6.34E-03	4.11E-10	4.04E-10	4.24E-10	1.16E-02	1.15E-02	1.17E-02
$\|\mathbf{x^{sol}}\|_1$	56631798.3	55368214.4	59669841.9	56631801.4	55368217.6	59669845.0	56631795.7	55368211.9	59669839.3
$\|\mathbf{x^*}\|_1$	56631797.7	55368213.9	59669841.3	56631797.7	55368213.9	59669841.3	56631797.7	55368213.9	59669841.3
Total Time	137.3	136.9	138.3	143.5	142.9	145.1	85.7	84.9	86.4
nMat	583.2	581	587	631.4	629	635	381.0	379	385

TABLE 5.9
Gaussian Experiment Results for $n = 120 \times 120$, $m = n/4$, $s = m/10$

	SPA			**NESTA**			**FPC**		
	Average	**Min**	**Max**	**Average**	**Min**	**Max**	**Average**	**Min**	**Max**
Update Iter. #	383.3	364	413	312.9	303	327	479.5	459	523
$\|\|\mathbf{x^{sol}}\|_1 - \|\mathbf{x^*}\|_1\|/\|\mathbf{x^*}\|_1$	5.01E-09	4.09E-09	7.08E-09	6.65E-08	5.42E-08	7.76E-08	2.48E-08	1.99E-08	2.96E-08
$\max\{\|(\mathbf{x^{sol}})_i - (\mathbf{x^*})_i\| : \|(\mathbf{x^*})_i\| > \mathbf{0}\}$	2.66E-04	1.79E-04	3.28E-04	6.72E-04	6.06E-04	7.23E-04	4.65E-04	4.27E-04	5.25E-04
$\max\{\|(\mathbf{x^{sol}})_i\| : (\mathbf{x^*})_i = \mathbf{0}\}$	2.47E-05	1.63E-05	3.01E-05	1.61E-04	1.24E-04	2.20E-04	1.13E-04	8.22E-05	1.39E-04
$\|\mathbf{Ax^{sol}} - \mathbf{b}\|_2$	3.94E-04	3.15E-04	4.60E-04	1.40E-07	1.19E-07	1.62E-07	1.28E-03	1.24E-03	1.31E-03
$\|\mathbf{x^{sol}}\|_1$	3122669.0	2633124.5	3766314.4	3122669.2	2633124.6	3766314.6	3122668.9	2633124.4	3766314.3
$\|\mathbf{x^*}\|_1$	3122669.0	2633124.4	3766314.4	3122669.0	2633124.4	3766314.4	3122669.0	2633124.4	3766314.4
Algorithm Time	90.2	86.2	97.1	234.4	228.6	243.5	112.8	107.8	123.5
Preprocessing Time	6.0	5.8	6.5	0.0	0.0	0.0	49.3	48.5	51.0
Total Time	96.1	92.1	103.0	234.4	228.6	243.5	162.1	156.3	172.4
nMat	767.6	729	827	625.8	606	654	960.0	919	1047

REFERENCES

[1] N. S. Aybat and A. Chakraborty, *Fast reconstruction of CT images from parsimonious angular measurements via compressed sensing*, submitted to SIAM Journal on Imaging Sciences, (2009).

[2] N. S. Aybat and G. Iyengar, *Extended analysis of SPA algorithm*, tech. report, IEOR Department, Columbia University, 2009.

[3] ———, *A first-order augmented Lagrangian method for compressed sensing*, submitted to SIAM Journal on Optimization, (2009).

[4] S. Becker, J. Bobin, and E. Candès, *Nesta: a fast and accurate first-order method for sparse recovery.* Submitted for publication, April 2009.

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

[6] E. Candès, J. Romberg, and T. Tao, *Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information*, IEEE Trans. Info. Th., 52 (2006).

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

[8] 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.

[9] 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.

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

[11] 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.

[12] 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.

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

[14] S. Hoda, A. Gilpin, and J. Pena, *Smoothing techniques for computing nash equilibria of sequential games*, tech. report, Technical report, Carnegie Mellon University, 2008.

[15] P. J. Hüber, *Robust Statistics*, New York: Wiley, 1981.

[16] K. Koh, S. J. Kim, and S. Boyd, *Solver for $\ell 1$-regularized least squares problems*, tech. report, Stanford University, 2007.

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

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

[19] 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.

[20] 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).

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