FAST ℓ_1-MINIMIZATION ALGORITHMS AND AN APPLICATION IN ROBUST FACE RECOGNITION: A REVIEW*

Allen Y. Yang, S. Shankar Sastry

Department of EECS, UC Berkeley
Berkeley, CA 94720, USA

Arvind Ganesh, Yi Ma

Coordinated Science Lab, UIUC
Urbana, IL 61801, USA

ABSTRACT

We provide a comprehensive review of five representative ℓ_1-minimization methods, i.e., *gradient projection*, *homotopy*, *iterative shrinkage-thresholding*, *proximal gradient*, and *augmented Lagrange multiplier*. The repository is intended to fill in a gap in the existing literature to systematically benchmark the performance of these algorithms using a consistent experimental setting. The experiment will be focused on the application of face recognition, where a sparse representation framework has recently been developed to recover human identities from facial images that may be affected by illumination change, occlusion, and facial disguise. The paper also provides useful guidelines to practitioners working in similar fields.

1. INTRODUCTION

ℓ_1-minimization (ℓ_1-min) has been one of the hot topics in the signal processing and optimization communities in the last five years. In the emerging compressive sensing (CS) theory [4], it has been shown to be an efficient approach to recover sparsest solutions to certain underdetermined systems of linear equations. More specifically, given an unknown signal $\boldsymbol{x}_0 \in \mathbb{R}^n$, an underdetermined full-rank matrix $A \in \mathbb{R}^{d \times n}$, and a $\boldsymbol{b} = A\boldsymbol{x}$, the ℓ_1-min problem solves the following convex program:

$$(P_1): \quad \min \|\boldsymbol{x}\|_1 \text{ subject to } \boldsymbol{b} = A\boldsymbol{x}. \tag{1}$$

Although the formulation of (P_1) constitutes a linear inverse problem, as the number of measurements in $\boldsymbol{b}$ is smaller than the number of unknowns in $\boldsymbol{x}$, CS theory shows that if $\boldsymbol{x}_0$ is sufficiently sparse and the sensing matrix A is *incoherent* with the basis under which $\boldsymbol{x}_0$ is sparse, $\boldsymbol{x}_0$ can be exactly recovered [5, 4]. This sparsity-seeking property of (P_1) has been shown to have tremendous applications in geophysics, data compression, image processing, and sensor networks.

In practice, the equality $\boldsymbol{b} = A\boldsymbol{x}$ is often relaxed to take into account the existence of measurement error in the sensing process: $\boldsymbol{b} = A\boldsymbol{x} + \boldsymbol{e}$. Particularly, if the error term $\boldsymbol{e}$ is assumed to be white noise such that $\|\boldsymbol{e}\|_2 \leq \epsilon$, the ground truth signal $\boldsymbol{x}_0$ can be well approximated by the so-called *basis pursuit denoising* (BPDN) [6, 5]:

$$(P_{1,2}): \quad \min \|\boldsymbol{x}\|_1 \text{ subject to } \|\boldsymbol{b} - A\boldsymbol{x}\|_2 \leq \epsilon. \tag{2}$$

In light of the high interest in finding more efficient algorithms to solve these problems, in this paper, we provide a comprehensive review of five representative methods, namely, *gradient projection* (GP) [10, 14], *homotopy* [19, 16, 8], *iterative shrinkage-thresholding* (IST)[7, 13, 22], *proximal gradient* (PG) [18, 1, 2], and *augmented Lagrange multiplier* (ALM) [24]. A full version of the study [23] together with the benchmark scripts is provided at `http://www.eecs.berkeley.edu/~yang/software/l1benchmark/`.

2. FAST ℓ_1-MIN ALGORITHMS

2.1. Gradient Projection (GP) Methods

We start with the ℓ_1-min problem $(P_{1,2})$. Using the Lagrangian method, the problem can be rewritten as:

$$\boldsymbol{x}^* = \arg\min_{\boldsymbol{x}} F(\boldsymbol{x}) = \arg\min_{\boldsymbol{x}} \frac{1}{2}\|\boldsymbol{b} - A\boldsymbol{x}\|_2^2 + \lambda\|\boldsymbol{x}\|_1, \tag{3}$$

where λ is the Lagrangian multiplier.

In the literature, there exist two slightly different methods to engage (3) as a quadratic programming problem, namely, *gradient projection sparse representation* (GPSR) [10] and *truncated Newton interior-point method* (TNIPM) [14]. In the following, we mainly discuss the TNIPM algorithm.[1] The reader is referred to our technical report [23] for the discussion of GPSR.

TNIPM transforms the objective function (3) to a quadratic program with inequality constraints

$$\begin{array}{ll} \min & \frac{1}{2}\|A\boldsymbol{x} - \boldsymbol{b}\|_2^2 + \lambda \sum_{i=1}^n u_i \\ \text{s.t.} & -u_i \leq x_i \leq u_i, \quad i = 1, \cdots, n. \end{array} \tag{4}$$

Then a *logarithmic barrier* for the constraints $-u_i \leq x_i \leq u_i$ can be constructed [11]:

$$\Phi(\boldsymbol{x}, \boldsymbol{u}) = -\sum_i \log(u_i + x_i) - \sum_i \log(u_i - x_i). \tag{5}$$

Over the domain of $(\boldsymbol{x}, \boldsymbol{u})$, the central path consists of the unique minimizer $(\boldsymbol{x}^*(t), \boldsymbol{u}^*(t))$ of the convex function

$$F_t(\boldsymbol{x}, \boldsymbol{u}) = t(\|A\boldsymbol{x} - \boldsymbol{b}\|_2^2 + \lambda\sum_{i=1}^n u_i) + \Phi(\boldsymbol{x}, \boldsymbol{u}), \tag{6}$$

where the parameter $t \in [0, \infty)$. The function can then be minimized by standard interior-point algorithms [17, 15].

*THIS WORK WAS PARTIALLY SUPPORTED BY ARO MURI W911NF-06-1-0076, NSF IIS 08-49292, NSF ECCS 07-01676, AND ONR N00014-09-1-0230.

[1] A MATLAB Toolbox for TNIPM called L1LS is available at `http://www.stanford.edu/~boyd/l1_ls/`.

2.2. Homotopy Methods

Recall that in (3), w.r.t. a fixed λ, the optimal solution is achieved when $\mathbf{0} \in \partial F(\boldsymbol{x})$. If we define $\mathcal{X} \doteq \{\boldsymbol{x}_\lambda^* : \lambda \in [0, \infty)\}$, $\mathcal{X}$ identifies a solution path that follows the change in λ: when $\lambda \to \infty$, $\boldsymbol{x}_\lambda^* = 0$; when $\lambda \to 0$, $\boldsymbol{x}_\lambda^*$ converges to the solution of (P_1). The *homotopy methods* exploit the fact that the objective function $F(\boldsymbol{x})$ undergoes a homotopy from the ℓ_2 constraint to the ℓ_1 objective in (3) as λ decreases. Furthermore, one can show that the solution path $\mathcal{X}$ is piece-wise constant as a function of λ [19, 9, 8]. Therefore, in constructing a decreasing sequence of λ, it is only necessary to identify those "breakpoints" that lead to changes of the support set of $\boldsymbol{x}_\lambda^*$, namely, either a new nonzero coefficient added or a previous nonzero coefficient removed.

The major obstacle in computing $\partial F(\boldsymbol{x})$ is that the ℓ_1-norm term is not globally differentiable. Therefore, we can consider the *subdifferential* of $\|\boldsymbol{x}\|_1$ defined as the following:

$$\boldsymbol{u}(\boldsymbol{x}) \doteq \partial\|\boldsymbol{x}\|_1 = \left\{\boldsymbol{u} \in \mathbb{R}^n : \begin{array}{c} u_i = \operatorname{sign}(x_i), x_i \neq 0 \\ u_i \in [-1, 1], x_i = 0 \end{array}\right\}. \tag{7}$$

The algorithm operates in an iterative fashion with an initial value $\boldsymbol{x}^{(0)} = 0$. In each iteration w.r.t. a nonzero λ, we assign $\partial F(\boldsymbol{x}) = 0$: $\boldsymbol{c}(\boldsymbol{x}) = A^T\boldsymbol{b} - A^TA\boldsymbol{x} = \lambda\boldsymbol{u}(\boldsymbol{x})$. Hence, from the definition (7), we maintain a sparse support set: $\mathcal{I} \doteq \{i : |\boldsymbol{c}_i^{(l)}| = \lambda\}$. Then the algorithm computes the update for $\boldsymbol{x}^{(k)}$ in terms of the positive/negative directions for its coefficients and the magnitude.[2]

2.3. Iterative Shrinkage-Thresholding (IST) Methods

In a nutshell, IST considers $(P_{1,2})$ as a special case of the following *composite objective function*:

$$\min_x F(\boldsymbol{x}) \doteq f(\boldsymbol{x}) + \lambda g(\boldsymbol{x}), \tag{8}$$

where $f(\boldsymbol{x}) = \frac{1}{2}\|A\boldsymbol{x} - \boldsymbol{b}\|^2$ and $g(\boldsymbol{x}) = \|\boldsymbol{x}\|_1$.

The update rule of the algorithm is defined based on an approximation of f [22, 1]:

$$\begin{aligned} \boldsymbol{x}^{(k+1)} \quad &\approx \quad \arg\min_{\boldsymbol{x}}\{(\boldsymbol{x} - \boldsymbol{x}^{(k)})^T\nabla f(\boldsymbol{x}^{(k)}) \\ &\qquad + \tfrac{\alpha^{(k)}}{2}\|\boldsymbol{x} - \boldsymbol{x}^{(k)}\|_2^2 + \lambda g(\boldsymbol{x})\} \\ &= \quad \arg\min_{\boldsymbol{x}}\{\tfrac{1}{2}\|\boldsymbol{x} - \boldsymbol{u}^{(k)}\|_2^2 + \tfrac{\lambda}{\alpha^{(k)}} g(\boldsymbol{x})\}. \end{aligned} \tag{9}$$

where $\boldsymbol{u}^{(k)} = \boldsymbol{x}^{(k)} - \frac{1}{\alpha^{(k)}}\nabla f(\boldsymbol{x}^{(k)})$, and the hessian $\nabla^2 f(\boldsymbol{x})$ is approximated by a diagonal matrix αI.

Now since the ℓ_1-norm $\|\boldsymbol{x}\|_1$ is a separable function, (9) has a closed-form solution w.r.t. each scalar coefficient:

$$x_i^{(k+1)} = \arg\min_{\boldsymbol{x}_i}\{\frac{(x_i - u_i^{(k)})^2}{2} + \frac{\lambda|x_i|}{\alpha^{(k)}}\} = \operatorname{soft}(u_i^{(k)}, \frac{\lambda}{\alpha^{(k)}}), \tag{10}$$

where $\operatorname{soft}(u, a) \doteq \begin{cases} \operatorname{sign}(u)(|u| - a) & \text{if |u|>a} \\ 0 & \text{otherwise} \end{cases}$ is the so-called *soft-thresholding* function.[3]

2.4. Proximal Gradient (PG) Methods

PG algorithms represent another class of algorithms that solve convex optimization problems defined in (8). Assume f is a smooth convex function with Lipschitz continuous gradient, and g is a continuous convex function. The principle behind proximal gradient algorithms is to iteratively form quadratic approximations $Q(\boldsymbol{x}, \boldsymbol{y})$ to F around a carefully chosen point $\boldsymbol{y}$, and to minimize $Q(\boldsymbol{x}, \boldsymbol{y})$ rather than the original cost function F.

Again, we define $g(\boldsymbol{x}) = \|\boldsymbol{x}\|_1$, and $f(\boldsymbol{x}) = \frac{1}{2}\|A\boldsymbol{x} - \boldsymbol{b}\|_2^2$. We note that $\nabla f(\boldsymbol{x}) = A^T(A\boldsymbol{x} - \boldsymbol{b})$ is Lipschitz continuous with Lipschitz constant $L_f \doteq \|A\|^2$[4]. Define $Q(\boldsymbol{x}, \boldsymbol{y}) \doteq f(\boldsymbol{y}) + \langle\nabla f(\boldsymbol{y}), \boldsymbol{x} - \boldsymbol{y}\rangle + \frac{L_f}{2}\|\boldsymbol{x} - \boldsymbol{y}\|_2^2 + \lambda\, g(\boldsymbol{x})$. Thus, we have a slightly different problem whose solution gets closer to the solution set of (1) as $\lambda \to 0$.

Given a proximal point $\boldsymbol{y}$, we already know the closed-form solution for $\boldsymbol{x}$ is given by the soft-thresholding operator: $\arg\min_{\boldsymbol{x}} Q(\boldsymbol{x}, \boldsymbol{y}) = \operatorname{soft}(\boldsymbol{u}, \frac{\lambda}{L_f})$, where $\boldsymbol{u} = \boldsymbol{y} - \frac{1}{L_f}\nabla f(\boldsymbol{y})$. Unlike the iterative thresholding algorithm described earlier, we use a smoothed computation of the sequence $\boldsymbol{y}^{(k)}$. It has been shown that choosing

$$\boldsymbol{y}^{(k)} = \boldsymbol{x}^{(k)} + \frac{t_{k-1} - 1}{t_k}(\boldsymbol{x}^{(k)} - \boldsymbol{x}^{(k-1)}), \tag{11}$$

where $\{t_k\}$ is a positive real sequence satisfying $t_k^2 - t_k \leq t_{k-1}^2$, achieves an accelerated non-asymptotic convergence rate of $O(k^{-2})$ [18, 1]. [5]

2.5. Augmented Lagrange Multiplier (ALM) Methods

Lagrange multiplier methods are a popular class of algorithms in convex programming. ALM methods differ from other penalty-based approaches by simultaneously estimating the optimal solution and Lagrange multipliers iteratively.

We consider the general ℓ_1-min problem (1) with the optimal solution $\boldsymbol{x}^*$. The corresponding augmented Lagrangian function is

$$L_\mu(\boldsymbol{x}, \boldsymbol{\lambda}) = \|\boldsymbol{x}\|_1 + \langle\boldsymbol{\lambda}, \boldsymbol{b} - A\boldsymbol{x}\rangle + \frac{\mu}{2}\|\boldsymbol{b} - A\boldsymbol{x}\|_2^2, \tag{12}$$

where $\mu > 0$ is a constant that determines the penalty for infeasibility, and $\boldsymbol{\lambda}$ is a vector of Lagrange multipliers. Let $\boldsymbol{\lambda}^*$ be a Lagrange multiplier vector satisfying the second-order sufficiency conditions for optimality (see section 3.2 in [3] for more details). Then, for sufficiently large μ, it has been shown that

$$\boldsymbol{x}^* = \arg\min_{\boldsymbol{x}} L_\mu(\boldsymbol{x}, \boldsymbol{\lambda}^*). \tag{13}$$

Thus, we can solve the above unconstrained optimization problem to compute $\boldsymbol{x}^*$.

It is clear that to compute $\boldsymbol{x}^*$ by minimizing $L_\mu(\boldsymbol{x}, \boldsymbol{\lambda})$, we must choose $\boldsymbol{\lambda}$ close to $\boldsymbol{\lambda}^*$ and set μ to be a very large

[2] A MATLAB implementation can be found at `http://users.ece.gatech.edu/~sasif/homotopy/`.

[3] A MATLAB implementation called SpaRSA is available at `http://www.lx.it.pt/~mtf/SpaRSA/`.

[4] $\|A\|$ represents the spectral norm of the matrix A.

[5] An implementation of the PG algorithm called FISTA is available for download from the website of the paper. Another Matlab toolbox called NESTA is available at: http: //www.acm.caltech.edu/ nesta/.

positive constant. The following iterative procedure has been proposed in [3] to simultaneously compute $\boldsymbol{\lambda}^*$ and $\boldsymbol{x}^*$:

$$\left\{ \begin{array}{rcl} \boldsymbol{x}_{k+1} & = & \arg\min_{\boldsymbol{x}} L_{\mu_k}(\boldsymbol{x}, \boldsymbol{\lambda}_k) \\ \boldsymbol{\lambda}_{k+1} & = & \boldsymbol{\lambda}_k + \mu_k(\boldsymbol{b} - A\boldsymbol{x}_{k+1}) \end{array} \right. , \tag{14}$$

where $\{\mu_k\}$ is a monotonically increasing positive sequence. We note that the first step in the above procedure is itself a convex optimization problem. Thus, the above iterative procedure is computationally efficient only if it is easier to minimize the augmented Lagrange function compared to solving the the original constrained optimization problem (1) directly.

Although it is not possible to obtain a closed-form solution for minimizing the first step of (14), since the quadratic penalty term is smooth and has a Lipschitz continuous gradient, we can still solve it efficiently using PG methods. [6]

3. SIMULATION: RANDOM SPARSE SIGNALS

We present two simulations to benchmark the performance of the five fast ℓ_1-min algorithms on random sparse signals, namely, L1LS, Homotopy, SpaRSA, FISTA, and ALM.

Firstly, we are interested in measuring the performance in the low-sparsity regime, where the ambient dimension $n = 2000$ and the sparsity $k = 200$ are fixed, and the dimension d of the random projection A varies between 800 and 1900. Secondly, we are interested in measuring the performance when $\boldsymbol{x}$ becomes dense w.r.t. a fixed sampling rate, where $n = 2000$ and $d = 1500$, and the sparsity ratio $\rho = k/n$ varies from 0.1 to 0.26. In both experiments, we corrupt the measurement vector with $\boldsymbol{e}$, an additive white noise term whose entries are i.i.d. distributed as $N(0, 0.01)$. The illustrations of the results are shown in our technical report [23].

We draw the following observations. First, when a low sparsity ratio of $\rho = 0.1$ is fixed, ℓ_1-min becomes better conditioned as the projection dimension d increases.

1. In terms of speed, L1LS and Homotopy take much longer time to converge than the first order methods.
2. The average run time of DALM is one of the lowest over all projection dimensions, which makes it the best algorithm in this comparison.

Second, when the projection dimension $d = 1500$ is fixed, we compare both the average run time and the average estimation error when the sparsity varies.

1. Among the five algorithms, only the run time of Homotopy grows mostly linearly with the increase of the sparsity ratio. It shows Homotopy becomes inefficient when the signal becomes dense.
2. DALM is the fastest algorithm in the low-sparsity regime, but its run time approaches that of the other first-order methods in the high-sparsity regime.

4. EXPERIMENT: ROBUST FACE RECOGNITION

This experiment is set up to estimate sparse representation of real face images based on a so-called *cross-and-bouquet* (CAB) model [20]. It has been shown in [21] that well-aligned frontal face images of a single human subject lie close to a low-dimensional linear subspace determined by:

$$A_i = [\boldsymbol{v}_{i,1}, \boldsymbol{v}_{i,2}, \cdots, \boldsymbol{v}_{i,n_i}] \in \mathbb{R}^{d \times n_i}, \tag{15}$$

where $\boldsymbol{v}_{i,j}$ represents the j-th training image from the i-th subject stacked in the vector form. Given C subjects and a test image $\boldsymbol{b}$ (in the vector form), we seek the sparsest linear representation of the image with respect to all training examples:

$$\boldsymbol{b} = [A_1, A_2, \cdots, A_C]\boldsymbol{x} = A\boldsymbol{x}. \tag{16}$$

where $A \in \mathbb{R}^{d \times n}$ collects all the training images. Clearly, if $\boldsymbol{b}$ is a valid test image, $\boldsymbol{b}$ must be associated with one of the C subjects. Therefore, the corresponding representation in (16) has a sparse representation $\boldsymbol{x} = [\cdots; \boldsymbol{0}; \boldsymbol{x}_i; \boldsymbol{0}; \cdots]$: on average only a fraction of $\frac{1}{C}$ coefficients are nonzero, and the dominant nonzero coefficients in sparse representation $\boldsymbol{x}$ reveal the true subject class.

In addition, we consider the situation where the query image $\boldsymbol{b}$ is severely occluded or corrupted. The problem is modeled by a corrupted set of linear equations $\boldsymbol{b} = A\boldsymbol{x} + \boldsymbol{e}$, where $\boldsymbol{e} \in \mathbb{R}^d$ is an unknown vector whose nonzero entries correspond to the corrupted pixels. In [21], the authors proposed to estimate $\boldsymbol{w} \doteq [\boldsymbol{x}; \boldsymbol{e}]$ together as the sparsest solution to an extended equation:

$$\min \|\boldsymbol{w}\|_1 \text{ subject to } \boldsymbol{b} = [A, \ I]\boldsymbol{w}. \tag{17}$$

The distribution of the new dictionary $[A, \ I]$ was dubbed as a cross-and-bouquet (CAB) model.

The performance of the five ℓ_1-min algorithms is benchmarked using a subset of the CMU Multi-PIE face database [12], which contains 249 subjects. The detailed experiment protocol is explained in the technical report. We measure the performance of the algorithms in terms of the recognition accuracy and the speed. The results are shown in Tables 1 and 2 w.r.t. different percentages of randomly corrupted pixels.

Table 1. Average recognition accuracy in percentage (bold numbers indicate the best).

Corruption	0%	20%	40%	60%	80%
L1LS	98.64	99.60	97.84	96.57	21.93
Homotopy	**99.88**	**99.88**	**99.91**	**98.67**	27.90
SpaRSA	99.69	99.47	98.8	90.51	21.1
FISTA	99.85	99.72	99.04	86.74	19.96
ALM	99.81	**99.88**	99.85	96.17	**29.01**

In terms of the accuracy, Homotopy achieves the best overall performance. For instance, with 60% of the pixels randomly corrupted, its average recognition rate based on the CAB model is about 99%. On the other hand, FISTA has the lowest recognition rates, followed by SpaRSA. In terms of the speed, Homotopy is also the fastest algorithm when the

[6] An implementation of ALM in both the primal and dual spaces can be found on the website of the paper. Another implementation in the dual space called YALL1 can be obtained from the authors of [24].

Table 2. Average run time in second (bold numbers indicate the best).

Corruption	0%	20%	40%	60%	80%
L1LS	19.48	18.44	17.47	16.99	14.37
Homotopy	**0.33**	**2.01**	**4.99**	12.26	20.68
SpaRSA	6.64	10.86	16.45	22.66	23.23
FISTA	8.78	8.77	8.77	**8.80**	**8.66**
ALM	18.91	18.85	18.91	12.21	11.21

corruption percentage is low. As Homotopy iteratively add or remove nonzero coefficients one at a time, the algorithm clearly is more efficient when the signal $\boldsymbol{w}$ is very sparse. It is also important to know that the speed of the two accelerated first-order algorithms, namely, FISTA and ALM, does not increase significantly as the sparsity of $\boldsymbol{w}$ increases.

It is more interesting to compare the difference in accuracy between L1LS and Homotopy, which provably solve the (P_1) problem, and SpaRSA, FISTA, and ALM, which essentially rely on the soft thresholding function and approximation of the gradients of the objective function. We observe that the exact solutions as a whole outperform the approximate solutions. Another important fact is that among the three first-order methods, ALM has been shown to boost the accuracy considerably

5. CONCLUSION AND DISCUSSION

The paper has provided a comprehensive review of the five fast ℓ_1-min methods. The extensive experiment has shown that, under a wide range of data conditions, there is no clear winner that always achieves the best performance. For noise-free data, on average PDIPA is more accurate than the rest of the algorithms, albeit at a much lower speed. Under random Gaussian dictionaries, approximate ℓ_1-min solutions (i.e., SpaRSA, FISTA, and ALM) are more efficient to estimate sparse signals. In the application of robust face recognition, a special CAB model was constructed based on real training images representing a large set of human subjects. Homotopy and ALM in turn achieve the highest recognition rate.

6. REFERENCES

[1] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM J. on Im. Sci.*, 2(1):183–202, 2009.

[2] S. Becker, J. Bobin, and E. Candes. NESTA: a fast and accurate first-order method for sparse recovery. *preprint*, 2009.

[3] D. Bertsekas. *Nonlinear Programming*. Athena Scientific, 2003.

[4] A. Bruckstein, D. Donoho, and M. Elad. From sparse solutions of systems of equations to sparse modeling of signals and images. *(in press) SIAM Review*, 2007.

[5] E. Candès, J. Romberg, and T. Tao. Stable signal recovery from incomplete and inaccurate measurements. *Comm. Pure and Appl. Math*, 59(8):1207–1223, 2006.

[6] S. Chen, D. Donoho, and M. Saunders. Atomic decomposition by basis pursuit. *SIAM Review*, 43(1):129–159, 2001.

[7] P. Combettes and V. Wajs. Signal recovery by proximal forward-backward splitting. *SIAM Multiscale Modeling and Simulation*, 4:1168–1200, 2005.

[8] D. Donoho and Y. Tsaig. Fast solution of ℓ^1-norm minimization problems when the solution may be sparse. *preprint*, 2006.

[9] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least angle regression. *The Annals of Statistics*, 32(2):407–499, 2004.

[10] M. Figueiredo, R. Nowak, and S. Wright. Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems. *IEEE J. Sel. Top. in Sig. Proc.*, 1(4):586–597, 2007.

[11] K. Frisch. The logarithmic potential method of convex programming. Technical report, University Institute of Economics (Oslo, Norway), 1955.

[12] R. Gross, I. Mathews, J. Cohn, T. Kanade, and S. Baker. Multi-pie. In *FGR*, 2006.

[13] E. Hale, W. Yin, and Y. Zhang. A fixed-point continuation method for ℓ^1-regularized minimization with applications to compressed sensing. Technical Report CAAM Tech. Rep. TR07-07, Rice University, 2007.

[14] S. Kim, K. Koh, M. Lustig, S. Boyd, and D. Gorinevsky. An interior-point method for large-scale ℓ_1-regularized least squares. *IEEE J. Sel. Top. Sig. Proc.*, 1(4):606–617, 2007.

[15] M. Kojima, N. Megiddo, and S. Mizuno. Theoretical convergence of large-step primal-dual interior point algorithms for linear programming. *Mathematical Programming*, 59:1–21, 1993.

[16] D. Malioutov, M. Cetin, and A. Willsky. Homotopy continuation for sparse signal representation. In *ICASSP*, 2005.

[17] R. Monteiro and I. Adler. Interior path following primal-dual algorithms. Part I: Linear programming. *Mathematical Programming*, 44:27–41, 1989.

[18] Y. Nesterov. A method of solving a convex programming problem with convergence rate $O(1/k^2)$. *Soviet Mathematics Doklady*, 27(2):372–376, 1983.

[19] M. Osborne, B. Presnell, and B. Turlach. A new approach to variable selection in least squares problems. *IMA Journal of Numerical Analysis*, 20:389–404, 2000.

[20] J. Wright and Y. Ma. Dense error correction via ℓ^1-minimization. *(accepted) IEEE Trans. Info. Theory*, 2010.

[21] J. Wright, A. Yang, A. Ganesh, S. Sastry, and Y. Ma. Robust face recognition via sparse representation. *IEEE Trans. PAMI*, 31(2):210 – 227, 2009.

[22] S. Wright, R. Nowak, and M. Figueiredo. Sparse reconstruction by separable approximation. In *ICASSP*, 2008.

[23] A. Yang, A. Ganesh, Z. Zhou, S. Sastry, and Y. Ma. Fast ℓ_1-minimization algorithms and an application in robust face recognition: a review. Technical Report UCB/EECS-2010-13, UC Berkeley, 2010.

[24] J. Yang and Y. Zhang. Alternating direction algorithms for ℓ_1-problems in compressive sensing. *preprint*, 2009.