Streaming Measurements in Compressive Sensing: ℓ_1 Filtering

M. Salman Asif and Justin Romberg
School of Electrical and Computer Engineering
Georgia Institute of Technology, Atlanta, Georgia, 30332, USA
Email: {sasif, jrom}@ece.gatech.edu

***Abstract*—The central framework for signal recovery in compressive sensing is ℓ_1 norm minimization. In recent years, tremendous progress has been made on algorithms, typically based on some kind of gradient descent or Newton iterations, for performing ℓ_1 norm minimization. These algorithms, however, are for the most part "static": they focus on finding the solution for a fixed set of measurements. In this paper, we will present a method for quickly updating the solution to some ℓ_1 norm minimization problems as new measurements are added. The result is an "ℓ_1 filter" and can be implemented using standard techniques from numerical linear algebra. Our proposed scheme is homotopy based where we add new measurements in the system and instead of solving updated problem directly, we solve a series of simple (easy to solve) intermediate problems which lead to the desired solution.**

***Index Terms*—Homotopy, Lasso, BPDN, ℓ_1 decoding, compressed sensing, dynamic measurement update**

I. Introduction

There are several problems in signal processing where a sparse solution to the system of linear equations is desired. In recent years this trend has captured a lot of attention, as it plays a central role in the theory and practice of compressive sensing (CS), [1]–[4]. The essential problem in CS is to reconstruct a signal from a set of linear measurements. We have the following model: $y = Ax$, where $x \in \mathbb{R}^n$ is a signal we wish to acquire, $y \in \mathbb{R}^m$ is the observation/measurement vector and A is an $m \times n$ measurement matrix with $m \ll n$. The compressive sensing theory puts conditions of *sparsity* on the signal x and *incoherence* of matrix A under which we can recover x via ℓ_1 minimization.

Given a set of measurements y, we can recover (an approximation of) x by solving the following relaxed unconstrained optimization program:

$$\text{minimize } \tau\|\widetilde{x}\|_1 + \frac{1}{2}\|A\widetilde{x} - y\|_2^2, \tag{1}$$

for some $\tau > 0$. This program goes by the name of basis pursuit denoising (BPDN) [5] in signal processing, or the Lasso [6] in statistics. In recent years several methods have been proposed to solve it efficiently (e.g., see [7], [8] and references therein). Our focus in this paper will be on homotopy based methods [9], [10] and [11].

A related problem to compressive sensing, as proposed by Candès and Tao [3], is the decoding by linear programming. It can be seen as an error correction or channel coding scheme. Assume that we want to transmit a message $x \in \mathbb{R}^n$ over a noisy communication channel. In order to make the transmission reliable we can encode x into a higher dimensional codeword Ax, where A is an $m \times n$ coding matrix with $m \gg n$. At the receiver end we receive a corrupted codeword $y = Ax + e$, where $e \in \mathbb{R}^m$ is the error vector. The goal is to recover x from y. As shown in [3], under suitable conditions on coding matrix A, if e is sparse then we can recover x exactly by solving the following ℓ_1 norm minimization problem, also known as ℓ_1 decoding,

$$\text{minimize } \|A\widetilde{x} - y\|_1. \tag{2}$$

The main focus in compressive sensing so far has been in solving problems like (1) and (2) for a *fixed* set of measurements. In this paper we present an efficient mechanism for updating the solution to some of these ℓ_1 problems as new measurements are added to the system. These schemes can quickly update the solution whenever new measurements become available, resulting in what we call an "ℓ_1 filter", which can be implemented efficiently using standard techniques from numerical linear algebra. Our proposed method is homotopy based, which is an iterative technique to solve an optimization problem (indirectly) by solving a series of relaxed intermediate problems leading to the solution of original problem.

The organization of this paper is as follows: In section II we will formulate the homotopy problems for measurement update in (1) and (2), section III discusses dynamic update of measurements in Lasso, section IV discusses dynamic update in ℓ_1 decoding and in section V we will briefly discuss homotopy formulation for some related problems, without giving details of the algorithms.

II. Motivation and Problem Formulation

In order to motivate this idea of new or "dynamic" measurements, let us first look at the classical least squares problem. Consider the following system of linear equations: $Ax = y$, where A is an $m \times n$ matrix with full column rank ($m \geq n$). In order to find the least-squares solution we can solve the following optimization problem

$$\text{minimize } \|A\widetilde{x} - y\|_2. \tag{3}$$

Fortunately we have analytical form for the solution to this problem, which can be written as: $\widehat{x}_0 = (A^TA)^{-1}A^Ty$, and the main computational cost here comes from solving a system of linear equations. Now suppose that we add one new measurement of x: $w = bx$ to the system, where b is a row vector. The new system of equations becomes

$$\begin{bmatrix} A \\ b \end{bmatrix} x = \begin{bmatrix} y \\ w \end{bmatrix},$$

and the least square solution to this updated system can be written as: $\widehat{x}_1 = (A^TA + b^Tb)^{-1}(A^Ty + b^Tw)$. We can see that solving this modified system of equations from scratch is overkill. Instead we can quickly find $\widehat{x}_1$ by using previously computed $\widehat{x}_0$ and $(A^TA)^{-1}$ along with a low rank update. This update can written as

$$\widehat{x}_1 = \widehat{x}_0 + \frac{(A^TA)^{-1}b^T(w - b\widehat{x}_0)}{1 + b(A^TA)^{-1}b^T}.$$

This above-mentioned scheme is the well known *recursive least squares* (RLS) method. As measurements come in, we easily move from one solution of (3) to the next, with each step involving a simple low-rank update [12].

In this paper we introduce the similar concept of "dynamic update of measurements" to the ℓ_1 problems given in (1) and (2). Let us first consider the BPDN or Lasso problem in (1). Assume that we add one new measurement to the system, given as $w := bx + d$, where b is a new row in the measurement matrix and d is some error in the new measurement. This gives us the following updated system

$$\begin{bmatrix} y \\ w \end{bmatrix} = \begin{bmatrix} A \\ b \end{bmatrix} x + \begin{bmatrix} e \\ d \end{bmatrix}. \tag{4}$$

Now in order to estimate x from this new system we want to solve the following updated version of (1)

$$\text{minimize } \tau\|\widetilde{x}\|_1 + \frac{1}{2}(\|A\widetilde{x} - y\|_2^2 + |b\widetilde{x} - w|^2), \tag{5}$$

for the same τ.

Similarly in case of ℓ_1 decoding, assume that we solve (2) to get the decoded message $\hat{x}$, and if original message x is not recovered we transmit one new measurement of x. This gives rise to the following updated form of ℓ_1 decoding problem in (2)

$$\text{minimize } \|A\widetilde{x} - y\|_1 + |b\widetilde{x} - w|. \tag{6}$$

The question we ask and try to answer here is whether we can quickly find the solution of the updated problems (5) and (6) by using information from the solutions of (1) and (2) respectively, without solving these new optimization problems from scratch.

The technique we present here for solving the updated ℓ_1 problems is in some sense similar to RLS but has some more structure to it, as we will discuss soon. The inherent difficulty in the dynamic update for ℓ_1 problems originate from the fact that we do not have any analytical form for the solutions. In addition to this ℓ_1 problems are not as smooth as least squares; the solution can change drastically with the addition of a single new measurement. We will move between the solutions using *homotopy* schemes, which will break the transition into a (hopefully small) series of low-rank updates.

Homotopy methods provide a general framework in which we can create some continuous transformation that changes a given *difficult* problem into a related but easy to solve problem. Then we attempt to solve the actual problem by starting from the easy one and solving a series of simple problems along the *homotopy path* towards the actual problem [13]. This progression along the homotopy path is controlled by some transformation parameter called *homotopy parameter*; usually varied between 0 and 1, where 0 and 1 correspond to the two end points of the homotopy path.

Our proposed homotopy formulation (see also [14]) for (1) is as follows

$$\text{minimize } \tau\|\widetilde{x}\|_1 + \frac{1}{2}(\|A\widetilde{x} - y\|_2^2 + \epsilon|b\widetilde{x} - w|^2), \tag{7}$$

where $\epsilon \in [0, 1]$ is the homotopy parameter. Similarly, the homotopy formulation for (6) can be written as

$$\text{minimize } \|A\widetilde{x} - y\|_1 + \epsilon|b\widetilde{x} - w|. \tag{8}$$

The solutions to (7) and (8) at $\epsilon = 0$ are the same as that for (1) and (2) respectively. And as ϵ increases from 0 to 1, respective solutions trace some homotopy path towards the solutions to (5) and (6).

III. Dynamic Lasso

Let us first look at the measurement update of Lasso in (7). And before discussing the algorithm we would like to mention that while preparing this manuscript we became aware of a similar algorithm independently developed by Garrigues and El Ghaoui [14]. Both the algorithms are based on homotopy principle and derive the desired homotopy path using the optimality conditions for problem in (7).

Using subgradient arguments it can be shown that any solution x_0 to (1) must obey the following necessary condition

$$\|A^T(Ax_0 - y)\|_\infty \le \tau. \tag{K}$$

In addition to this, a sufficient condition for the optimality of x_0 requires that (K) holds with equality for the indices where x_0 is non zero, and strict inequality elsewhere. So the necessary and sufficient optimality conditions [15] for any solution x_0 to (1) can be written as

K1. $A_\Gamma^T(Ax_0 - y) = -\tau z$

K2. $\|A_{\Gamma^c}^T(Ax_0 - y)\|_\infty < \tau$,

where index set Γ denotes the support of x_0, z is its sign sequence on Γ and A_Γ denotes columns of A indexed by elements in Γ. This tells us that for any given τ the solution to (1) is completely described by the support Γ and sign sequence z corresponding to τ, given as

$$x_0 = \begin{cases} (A_\Gamma^TA_\Gamma)^{-1}(A_\Gamma^Ty - \tau z) & \text{on } \Gamma \\ 0 & \text{otherwise} \end{cases}.$$

Now if we add one new measurement, as described in (4), the support and sign sequence of solution can change. So in our proposed homotopy scheme we keep track of the changes in support and sign sequence of solution as we increase ϵ from 0 to 1.

Let us now discuss the homotopy scheme for the solution update. First of all, we need the optimality conditions that must be satisfied by any solution $x^{(\epsilon)}$ to (7) for any given value of τ and ϵ. These conditions can be written as

L1. $A^T_{\Gamma_\epsilon}(Ax^{(\epsilon)} - y) + \epsilon b^T_{\Gamma_\epsilon}(bx^{(\epsilon)} - w) = -\tau z_\epsilon$

L2. $\|A^T_{\Gamma^c_\epsilon}(Ax^{(\epsilon)} - y) + \epsilon b^T_{\Gamma^c_\epsilon}(bx^{(\epsilon)} - w)\|_\infty < \tau$,

where Γ_ϵ is the support of $x^{(\epsilon)}$ and z_ϵ is the corresponding sign sequence. In our proposed algorithm for the solution update, we *fix* τ and iteratively change ϵ from 0 to 1 in such a way that at each step either a new element enters the support of solution or an existing element leaves the support. So we move in a particular direction (given by optimality conditions) such that one element change occurs in the support, at which point we update the support and find the new direction to move in. This gives us the desired homotopy path traced by the solution $x^{(\epsilon)}$ to (7), which is parameterized by $\epsilon \in [0, 1]$. We will refer to the values of ϵ for which some change occurs in the support as the *critical values*, respective points on the homotopy path as *vertices* and the intervals between any two vertices as *facets* of the homotopy path. It is important to note that support Γ_ϵ remains same throughout any given facet determined by the values of ϵ at the two end vertices. So what we do in this algorithm is that we move along these facets of the homotopy path from one vertex to another by increasing ϵ, while updating the support and sign sequence of solution, until ϵ becomes equal to 1. And as given in (L1), the solution $x^{(\epsilon)}$ at any value of ϵ is completely defined by the support Γ_ϵ and sign sequence z_ϵ. The update direction (i.e., facets) on the homotopy path and the step size (i.e., length of each facet) can be derived using the optimality conditions in (L1-L2), as described next.

A. Homotopy update

Let us assume that we already have a solution x_k to (7) at some critical value $\epsilon = \epsilon_k$, with support Γ and sign sequence z on Γ. The corresponding optimality conditions in (L1-L2) can be written as

$$A^T_\Gamma(Ax_k - y) + \epsilon_k b^T_\Gamma(bx_k - w) = -\tau z \tag{9a}$$

$$\|A^T_{\Gamma^c}(Ax_k - y) + \epsilon_k b^T_{\Gamma^c}(bx_k - w)\|_\infty < \tau. \tag{9b}$$

Now we need to find an update direction (i.e., a new facet) on the homotopy path such that ϵ increases by most. Here we use the fact that support of the solution will not change on this new facet until we hit another vertex, which will give us the new critical value and update in the support. So with an update direction $\widetilde{\partial x}$, the condition in (L1) at some new ϵ_{k+1} on this facet can be written as

$$A^T_\Gamma[A(x_k + \widetilde{\partial x}) - y] + \epsilon_{k+1} b^T_\Gamma[b(x_k + \widetilde{\partial x}) - w] = -\tau z$$
$$(\epsilon_{k+1} - \epsilon_k) b^T_\Gamma(bx_k - w) + (A^T_\Gamma A + \epsilon_{k+1} b^T_\Gamma b)\widetilde{\partial x} = 0,$$

where the second equation follows from (9a). This gives us the update direction along with the step size to go from ϵ_k to ϵ_{k+1} on the facet determined by the support Γ as

$$\widetilde{\partial x} = \begin{cases} -(\epsilon_{k+1} - \epsilon_k)(A^T_\Gamma A_\Gamma + \epsilon_{k+1} b^T_\Gamma b_\Gamma)^{-1} b^T_\Gamma(bx_k - w) & \text{on } \Gamma \\ 0 & \text{otherwise.} \end{cases}$$

We can simplify this equation using Sherman-Woodbury formula [16], to separate the step size from the update direction, which gives us the following equations for homotopy update (let $U := A^T_\Gamma A_\Gamma + \epsilon_k b^T_\Gamma b_\Gamma$ and $u := b_\Gamma U^{-1} b^T_\Gamma$)

$$\partial x = \begin{cases} -U^{-1} b^T_\Gamma(bx_k - w) & \text{on } \Gamma \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

$$\theta = \frac{\epsilon_{k+1} - \epsilon_k}{1 + (\epsilon_{k+1} - \epsilon_k)u}, \tag{11}$$

where ∂x is the new update direction and θ is the step size in this direction such that ϵ changes from ϵ_k to ϵ_{k+1}. As we increase θ, ϵ increases and at some point we hit a vertex where support of the solution changes, i.e., either a new element enters the support Γ or an existing element in x_k shrinks to zero. So next thing is to find the smallest step size θ such that one of these two things happen.

Before discussing the algorithm for choosing the step size θ, let us first describe how to update the optimality conditions given in (L1-L2) as ϵ changes. With update direction ∂x and step size θ, as defined in (10) and (11), we can write the optimality conditions at $\epsilon = \epsilon_{k+1}$ as the set of following constraints

$$\|A^T[A(x_k + \theta\partial x) - y] + \epsilon_{k+1} b^T[b(x_k + \theta\partial x) - w]\|_\infty \le \tau,$$

which are active (i.e., hold with equality) only on the indices in the support set Γ. We can simplify this equation further to write it as

$$\|p_k + \theta d_k\|_\infty \le \tau, \tag{12}$$

where p_k and d_k are defined as

$$p_k = A^T(Ax_k - y) + \epsilon_k b^T(bx_k - w) \tag{13a}$$

$$d_k = (A^T A + \epsilon_k b^T b)\partial x + b^T(bx_k - w). \tag{13b}$$

In (12), p_k is the set of old optimality conditions from (9) at $\epsilon = \epsilon_k$, d_k is the update, θ is the step size from (11) and $p_k + \theta d_k$ is the set of new optimality conditions at $\epsilon = \epsilon_{k+1}$.

B. Algorithm

We are ready to discuss the main algorithm for the homotopy update. Assume that we have a solution x_k to (7) at $\epsilon = \epsilon_k$, with support Γ and sign sequence z on Γ. Compute $\partial x, p_k$ and d_k. Select the smallest $\theta > 0$ such that either one of the constraints in $p_{k+1} := p_k + \theta d_k$ (originally inactive in p_k) becomes active, or one of the elements in $x_{k+1} := x_k + \theta\partial x$ (originally non-zero in x_k) shrinks to zero. The appropriate step size θ can be chosen as described below:

$$|p_k(j) + \theta d_k(j))| = \tau \qquad \text{for all } j \in \Gamma$$
$$|p_k(j) + \theta d_k(j))| \le \tau \qquad \text{for all } j \in \Gamma^c$$
$$\theta^+ = \min_{j \in \Gamma^c}\left(\frac{\tau - p_k(j)}{d_k(j)}, \frac{\tau + p_k(j)}{-d_k(j)}\right)_+$$
$$\theta^- = \min_{j \in \Gamma}\left(\frac{-x_k(j)}{\partial x(j)}\right)_+$$
$$\theta = \min(\theta^+, \theta^-), \tag{14}$$

Algorithm 1 Dynamic Lasso Homotopy

Start with $\epsilon_0 = 0$ at solution x_0 to (1) with support Γ and sign sequence z on the Γ for $k = 0$.
repeat
  compute ∂x as in (10)
  compute p_k, d_k as in (13) and θ as in (14)
  $x_{k+1} = x_k + \theta\partial x$
  $\epsilon_{k+1} = \epsilon_k + \dfrac{\theta}{1-\theta u}$
  if $\epsilon_{k+1} \geq 1$ **or** $\epsilon_{k+1} < 0$ **then**
    $\theta = \dfrac{1-\epsilon_k}{1+(1-\epsilon_k)u}$
    $x_{k+1} = x_k + \theta\partial x$
    $\epsilon_{k+1} = 1$
    `break;` {Quit without any further update}
  end if
  if $\theta = \theta^-$ **then**
    $\Gamma \leftarrow \Gamma \setminus \{j^-\}$
    update z
  else
    $\Gamma \leftarrow \Gamma \cup \{j^+\}$
    $z(j^+) = \text{sign}\,[p_k(j^+) + \theta d_k(j^+)]$
  end if
  $k \leftarrow k+1$
until `stopping criterion is satisfied`

where $\min(\cdot)_+$ denotes that minimum is taken over positive arguments only. Let us call indices corresponding to θ^+ and θ^- as j^+ and j^- respectively. So either j^+ enters the support Γ (if $\theta^+ < \theta^-$) or j^- leaves the support Γ (if $\theta^- < \theta^+$), sign sequence z is updated accordingly. The new value of ϵ becomes

$$\epsilon_{k+1} = \epsilon_k + \frac{\theta}{1-\theta u} \tag{15}$$

and the solution at this new vertex becomes $x_{k+1} = x_k + \theta\partial x$. Repeat this procedure until ϵ becomes equal to 1.

A word of caution: since we are tracking ϵ indirectly using step size θ, so at the last step ϵ_{k+1} can be greater than 1 and we will need to adjust θ such that final value of ϵ becomes equal to 1. Also it can happen that $1-\theta u$ in (15) reduces towards zero and becomes negative to make ϵ_{k+1} very large and then negative (it cannot become positive again because negative values of $\frac{\theta}{1-\theta u}$ lie in the interval $(-\infty, -\epsilon_k)$). As soon as this happens, it gives us the indication that the value of ϵ_{k+1} changed from ϵ_k to a very large value and then negative. So we need to choose a smaller value of θ such that final value of ϵ is 1 and quit. A pseudocode for this procedure is given in Algorithm 1.

C. Numerical implementation

The main computational cost for the Algorithm 1 comes from computing update direction ∂x and step size θ at each homotopy step. Step size calculation involves 2 matrix vector multiplications of an $m \times n$ system and a few vector multiplications. In order to compute update direction at each step we need to find $(A_\Gamma^T A_\Gamma + \epsilon b_\Gamma^T b_\Gamma)^{-1}$. Since at each step our support set Γ changes by one element at most, we do not need to compute $(A_{\Gamma_k}^T A_{\Gamma_k} + \epsilon_k b_{\Gamma_k}^T b_{\Gamma_k})^{-1}$ from scratch, and instead we can easily update the previously computed $(A_{\Gamma_{k-1}}^T A_{\Gamma_{k-1}} + \epsilon_{k-1} b_{\Gamma_{k-1}}^T b_{\Gamma_{k-1}})^{-1}$ using matrix inversion lemma [16], where Γ_k and Γ_{k-1} denote support set at kth and $(k-1)$th homotopy step. So essentially the computational cost at each step is equivalent to a few (say 3 to 5) matrix vector multiplications [1].

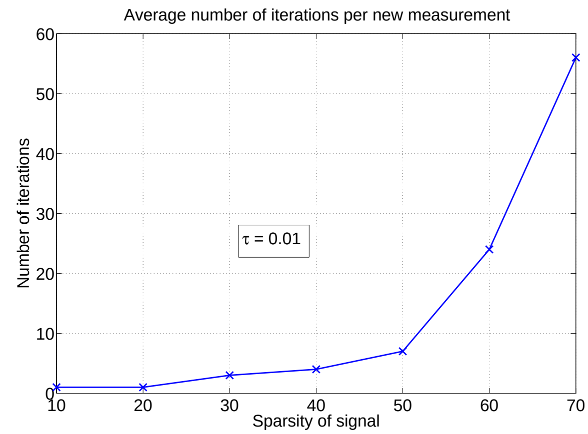

Fig. 1. Average number of homotopy steps with one new measurement at different sparsity levels. ($n = 256, m = 150, \tau = 0.01$)

D. Simulation results

In this section we discuss some simulation results for Algorithm 1. We took $m = 150$ noiseless measurements of an $n = 256$ dimensional sparse signal x, given as $y = Ax$, using a random matrix A whose entries were independently chosen from $\mathcal{N}(0,1)$. After solving (1) for a fixed parameter τ, we added one new random measurement $w = bx$ to the system and solved (5) as described in Algorithm 1. We performed this experiment 500 times for each different sparsity level of x (as specified along x-axis). To generate sparse signal, each time we chose a random support and selected the entries independently from $\mathcal{N}(0,1)$. Fig. 1 plots the average number of homotopy steps taken by Algorithm 1 to update the solution after adding one new measurement. As we can see here that for a *reasonable* sparsity level (e.g., $m/4$ or $m/5$) we need about 3 or 4 homotopy steps for solution update.

IV. ℓ_1 Decoding

In the same spirit we can form a homotopy algorithm to solve (8) for increasing value of $\epsilon : 0 \rightarrow 1$, while obeying some optimality conditions. For the sake of simplicity, here we will discuss the algorithm with one new measurement but this method can easily be extended to add multiple new measurements with homotopy. The homotopy scheme for ℓ_1 decoding can be seen as a type of *primal-dual* homotopy, where we update primal and dual vectors at every homotopy step, and use strong duality [17] between the objectives of

[1] Matlab files implementing these algorithms will be available from this webpage: http://users.ece.gatech.edu/∼sasif/

primal and dual problems to derive the required optimality conditions. Let us first write the dual problem to (8) as

$$\begin{aligned} \text{maximize} \quad & -\lambda^T y - \epsilon \nu w \\ \text{subject to} \quad & A^T\lambda + \epsilon b^T \nu = 0 \\ & \|\lambda\|_\infty \leq 1, \quad |\nu| \leq 1, \end{aligned} \tag{16}$$

where $\lambda \in \mathbb{R}^m$ and $\nu \in \mathbb{R}$ are the dual variables. Assume that (x_k, λ_k, ν_k) is the solution set to primal and dual problems in (8) and (16) for $\epsilon = \epsilon_k$. Let $e_k := Ax_k - y$ be the error estimate corresponding to old measurements with support Γ_e and $d_k := bx_k - w$ be the error estimate for new measurement at index γ_d. Using strong duality between the primal and dual objectives in (8) and (16) respectively, we get the following set of conditions which must be obeyed by any primal and dual solution pair (x_k, λ_k, ν_k) at any given value of ϵ_k

$$\lambda_k = \text{sign}(Ax_k - y) \ \text{ on } \Gamma_e, \qquad \|\lambda_k\|_\infty < 1 \ \text{ on } \Gamma_e^c \tag{D1}$$

$$\nu_k = \text{sign}(bx_k - w) \ \text{ if } d_k \neq 0, \quad |\nu_k| < 1 \ \text{ otherwise} \tag{D2}$$

$$A^T\lambda_k + \epsilon_k b^T \nu_k = 0. \tag{D3}$$

These conditions tell us that the dual vectors lie in the left null space of the coding matrix (ignoring the presence of ϵ_k here), and whenever an entry in the error estimate is non-zero the corresponding dual element is equal to the sign of error at that location, and the absolute value of dual element at all other indices is strictly less than 1. So the homotopy scheme for ℓ_1 decoding will be all about the update of the active set for the error estimate (or dual vectors), let us denote that as $\Gamma := \{\Gamma_e \cup \gamma_d\}$, and keeping dual vector in the left null space of coding matrix as we increase ϵ from 0 to 1.

In order to build the homotopy scheme for ℓ_1 decoding we need following assumptions for the coding matrix and any solution of ℓ_1 decoding problem in (8).

Assumption 1: Any $n \times n$ sub-matrix of the coding matrix is non singular.

Assumption 2: The error estimate of ℓ_1 decoding problem will have exactly n zero entries whenever exact message x is not recovered.

These assumptions hold for Gaussian matrix with probability 1, and with very high probability for Bernoulli matrix. In addition to this, the condition number of any sub matrix from such ensembles is also *fairly* controlled [18]. In our proposed algorithm (as we discuss in detail later) we need Assumption 1 because for every homotopy step we will be computing inverse of an $n \times n$ matrix to find the update direction for primal and dual vectors. Assumption 2 ensures that such update direction exists and is unique. In addition to this, it gives us a stopping rule, because whenever number of zero entries in error estimate exceeds n, it means we have decoded the message correctly.

A. Homotopy update

The homotopy algorithm for ℓ_1 decoding can be divided into two main parts: primal and dual update. We first update the dual vector pair (λ, ν) to find one new element in the error estimates (e, d) and then update primal vector (x) to remove one of the existing elements from the error estimates (e, d). Our proposed scheme relies heavily on Assumption 2, i.e., at any homotopy step exactly n entries in the dual vector pair (λ, ν) have their absolute values strictly less than 1, which correspond to the zero entries in the respective error estimates (due to optimality conditions in (D1) and (D2)). This gives us n degrees of freedom for update of dual vector at the indices in Γ^c, and we update dual vectors with increasing value of ϵ (in a particular direction) until one element of dual vector in Γ^c becomes equal to +1 or -1, which indicates that a new element has entered the support of error estimates (e, d) and the corresponding value of dual vector gives its sign. Since by Assumption 2 we will have exactly n zero entries in the error estimate at any point, so during primal update phase we update x in such a way that atleast one old entry from the error estimate shrinks to zero. Repeat this procedure until ϵ becomes equal to 1. If at any point along the homotopy path, d shrinks to zero (lucky breakdown!), we set the corresponding dual variable ν to the value of ϵ at that point and quit.

Initialization: Assume that we solve ℓ_1 decoding problem in (2) to get the solution x_0, with error estimate $e_0 := Ax_0 - y$ supported on the set Γ_e. Without loss of generality assume that $d_0 := bx_0 - w$ is non-zero. So as described in (D2) the dual vector $\nu_0 = z_d$, where $z_d = \text{sign}(bx_0 - w)$. The new support of error estimates becomes $\Gamma := \{\Gamma_e \cup \gamma_d\}$.

Assume that we already have primal-dual solutions (x_k, λ_k, ν_k) to the problems in (8) and (16) for $\epsilon = \epsilon_k$, with support set Γ which corresponds to the non zero entries in the error estimates $e_k := Ax_k - y$ and $d_k := bx_k - w$, where $d_k \neq 0$.

Dual update: The dual feasibility condition in (D3) gives us the following equation

$$A^T\lambda_k + \epsilon_k b^T \nu_k = 0,$$

Since $\nu_k = z_d$, by Assumption 2 we have n degrees of freedom to change λ at indices corresponding to Γ^c. So with an update direction $\widetilde{\partial\lambda}$ (supported on the set Γ^c only), the feasibility condition in (D3) at some new ϵ_{k+1} can be written as

$$A^T(\lambda_k + \widetilde{\partial\lambda}) + \epsilon_{k+1} b^T z_d = 0,$$

or equivalently (only on index set Γ^c)

$$(A^T)_{\Gamma^c}\widetilde{\partial\lambda}_{\Gamma^c} + (\epsilon_{k+1} - \epsilon_k) b^T z_d = 0. \tag{17}$$

So we can write the update direction $\partial\lambda$ (whenever the inverse of $(A^T)_{\Gamma^c}$ exists under Assumption 1) and the step size θ^+, required to increase ϵ from ϵ_k to ϵ_{k+1} on the homotopy facet determined by Γ as

$$\partial\lambda = \begin{cases} -[(A^T)_{\Gamma^c}]^{-1} b^T z_d & \text{on } \Gamma^c \\ 0 & \text{otherwise,} \end{cases} \tag{18}$$

$$\theta^+ = \epsilon_{k+1} - \epsilon_k.$$

As we increase θ^+, ϵ increases and at some point we hit a new vertex where an element of λ, say at index $\gamma^+ \in \Gamma^c$, gets active (i.e., becomes equal to +1 or -1). This tells us that we have a new element in the error estimate e at index γ^+ with sign z_γ, same as that of the new element in λ.

Primal update: Since we need to have exactly n zero entries in the error estimate, so during primal update phase we use the information from dual update phase and change x in such a way that one of the existing element in the error estimate shrinks to zero. Consider the following set of equations (corresponding to the error estimates) at ϵ_k

$$\underbrace{\begin{bmatrix} A \\ b \end{bmatrix}}_{G} x_k - \underbrace{\begin{bmatrix} y \\ w \end{bmatrix}}_{q} = \underbrace{\begin{bmatrix} e_k \\ d_k \end{bmatrix}}_{c_k}, \quad (19)$$

where we know that c_k is supported only on the set Γ. In the dual update phase we added one new element in the support of c at index $\gamma^+ \in \Gamma^c$ with sign z_γ, so we need to update x in such a way that $\text{sign}[c_{k+1}(\gamma^+)] = z_\gamma$ and c_{k+1} is zero at all other indices in Γ^c. Using the update direction $\widetilde{\partial x}$ we can write (19), corresponding to the rows indexed by elements in Γ^c, as

$$(G(x_k + \widetilde{\partial x}) - q)_{[\Gamma^c,:]} = (c_k)_{[\Gamma^c,:]} + \theta^- \widetilde{\partial c}, \quad (20)$$

where the notation $U_{[\Gamma^c,:]}$ represents entries of U at rows indexed by elements in Γ^c, $\widetilde{\partial c}$ is defined as

$$\widetilde{\partial c} = \begin{cases} z_\gamma & \text{on } \gamma^+ \\ 0 & \text{on } \Gamma^c \backslash \{\gamma^+\} \end{cases}, \quad (21)$$

and θ^- is the unknown value for the new element in c. Using (20) and (21) we can write the following system of equations to compute the update direction ∂x

$$A_{[\Gamma^c,:]} \partial x = \begin{cases} z_\gamma & \text{on } \gamma^+ \\ 0 & \text{on } \Gamma^c \backslash \{\gamma^+\} \end{cases}. \quad (22)$$

The associated step size with ∂x is θ^-. As we increase θ^-, at some point one of the elements in c at some index $\gamma^- \in \Gamma$ will shrink to zero. We can update the primal-dual vectors and supports accordingly.

B. Algorithm

Assume that we have a solution (x_k, λ_k, ν_k) for ϵ_k with support Γ as described before.

Primal update: Compute $\partial\lambda$ as described in (18). Find the step size θ^+ as follows

$$\begin{aligned} &|\lambda_k(j) + \theta^+ \partial\lambda(j)| \leq 1 \quad \text{for all } j \in \Gamma^c \\ &\theta^+ = \min_{j \in \Gamma^c} \left\{ \frac{1 - \lambda_k(j)}{\partial\lambda(j)}, \frac{1 + \lambda_k(j)}{-\partial\lambda(j)} \right\}_+. \end{aligned} \quad (23)$$

Let us denote γ^+ as the index corresponding to θ^+ and z_γ as sign of $\lambda_k(\gamma^+) + \theta^+ \partial\lambda(\gamma^+)$. The new values for ϵ and dual vector λ are given as

$$\epsilon_{k+1} = \epsilon_k + \theta^+, \qquad \lambda_{k+1} = \lambda_k + \theta^+ \partial\lambda.$$

Dual update: Compute ∂x from (22), and define $\partial c := G \partial x$. Find the step size θ^- as

$$\theta^- = \min_{j \in \Gamma} \left(\frac{-c_k(j)}{\partial c(j)} \right)_+. \quad (24)$$

Algorithm 2 ℓ_1 Decoding Homotopy

Start at $\epsilon_0 = 0$ with primal-dual solution x_0, λ_0, error estimate $e_0 := Ax_0 - y$ with support Γ_e. set $d_0 := bx_0 - w$, and $\nu_0 := z_d$. Set $\Gamma = [\Gamma_e \cup \gamma_d]$, $c_0 := \begin{bmatrix} e_0 \\ d_0 \end{bmatrix}$ and $G := \begin{bmatrix} A \\ b \end{bmatrix}$

repeat

  Dual update:

  compute $\partial\lambda$ as in (18)

  find θ^+, γ^+ and z_γ as described in (23)

  $\lambda_{k+1} = \lambda_k + \theta^+ \partial\lambda$

  $\epsilon_{k+1} = \epsilon_k + \theta^+$

  if $\epsilon_{k+1} \geq 1$ **then**

    $\theta^+ = 1 - \epsilon_k$

    $\lambda_{k+1} = \lambda_k + \theta^+ \partial\lambda$

    $\nu_{k+1} = z_d$

    $\epsilon_{k+1} = 1$

    `break;` {Quit without any further update}

  end if

  Primal update:

  compute ∂x as in (22), set $\partial c := G \partial x$

  find θ^- and γ^- as described in (24)

  $x_{k+1} = x_k + \theta^- \partial x$

  $c_{k+1} = c_k + \theta^- \partial c$

  $\Gamma \leftarrow [\Gamma \cup \gamma^+] \backslash \{\gamma^-\}$

  if $\gamma^- = \gamma_d$ **then**

    $\nu_{k+1} = \epsilon_{k+1} z_d$

    `break;` {Lucky breakdown}

  end if

  $k \leftarrow k + 1$

until `stopping criterion is satisfied`

Let us denote γ^- as the index corresponding to θ^-. The new value of x is given as: $x_{k+1} = x_k + \theta^- \partial x$.

The support set can be updated as $\Gamma = [\Gamma \cup \gamma^+] \backslash \{\gamma^-\}$. Repeat this procedure until ϵ becomes equal to 1, or d_k shrinks to zero or the number of zero entries in c increases from n.

C. Numerical implementation

The main computational cost in this algorithm also comes from solving a system of equations to find update direction. Similarly, at every homotopy step we have one element change in the support set Γ^c, and consequently in $A_{[\Gamma^c,:]}$ a row at index γ^+ is replaced with a_{γ^-} (row of A at index γ^-) . So instead of computing inverse of $A_{[\Gamma^c,:]}$ at each step, we can simply update the old one using Sherman-Woodbury formula [16]. This is because we can write $A_{[\Gamma^c,:]}$ at $(k+1)$th step as rank one update of $A_{[\Gamma^c,:]}$ at kth step, given as

$$A^{(k+1)}_{[\Gamma^c,:]} = A^{(k)}_{[\Gamma^c,:]} + \mathbf{1}_\gamma (a_{\gamma^-} - a_{\gamma^+}),$$

where $\mathbf{1}_\gamma \in \mathbb{R}^n$ represents a vector with all zeros except at index γ (where γ corresponds to the location of γ^+ in Γ^c).

D. Simulation results

In this section we discuss some simulation results for Algorithm 2, as given in Table I and Fig. 2. In this experiment,

TABLE I
AVERAGE LENGTH OF CODEWORD REQUIRED FOR PERFECT RECOVERY IN THE PRESENCE OF S SPARSE ERRORS USING ℓ_1 DECODING ($n = 64$).

No. of errors (S)	Average redundancy	Average codeword length
10	31	95
20	54	118
30	73	137
40	91	155
50	108	172

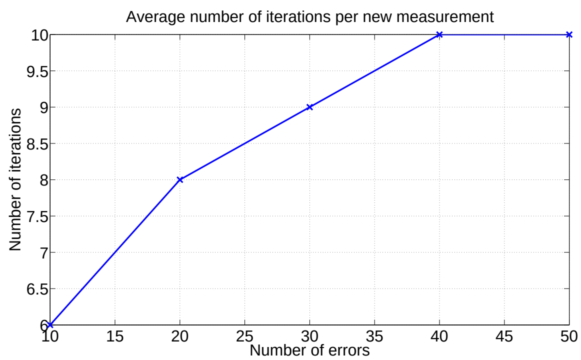

Fig. 2. Average number of homotopy iterations required per one measurement update ($n = 64$)

we start with an arbitrary $n = 64$ dimensional message signal x, encode it using an $n \times n$ random matrix A and introduce some random errors at S randomly chosen locations. The first estimated codeword can be found as $x_0 = A^{-1}y$. Afterwards we start adding new measurements (one at a time) and solve (6) using Algorithm 2 until the exact message is recovered. Table I gives the average number of measurements (length of codeword) required for perfect reconstruction when S sparse errors are present in the starting codeword. As it can be seen that the redundancy $(m - n)$ required for perfect recovery is about $2S$-$3S$. Fig. 2 gives a *rough* estimate for number of homotopy steps required, per new measurement, to update the solution. It is observed that the algorithm takes lesser number of iterations when sparsity level of error estimate is less (i.e., when $m \approx n$), and as we add new entries to the codeword (i.e., $m - n$ increases) the number of homotopy steps taken increases as well.

V. SOME MORE HOMOTOPY SCHEMES

In this section we will mention some related problems for which we can develop similar dynamic measurement update algorithms using the homotopy ideas. We will discuss these algorithms in detail in a longer version of the paper we are currently working on.

In section III we discussed the homotopy scheme for measurement update in Lasso. In a very similar way we can develop the homotopy algorithm for measurement update in the Dantzig selector (DS) [19]. Consider same system model as for Lasso, the Dantzig selector solves following optimization problem to estimate x

$$\text{minimize } \|\widetilde{x}\|_1 \quad \text{subject to} \quad \|A^T(A\widetilde{x} - y)\|_\infty \leq \tau,$$

for some $\tau > 0$. Similarly if we add a new measurement $w := bx + d$ to the system, we can write the homotopy formulation for updated problem as:

$$\text{minimize } \|\widetilde{x}\|_1 \ \text{ subject to } \ \|A^T(A\widetilde{x}-y)+\epsilon b^T(b\widetilde{x}-w)\|_\infty \leq \tau,$$

for the *same* τ. And using the optimality conditions for DS (derived from strong duality between primal and dual forms [20]) we can develop a homotopy algorithm for dynamic measurement update which is very similar to Algorithm 1.

The ℓ_1 decoding scheme we discussed in section IV assumes that e is sparse error vector. However, in practice we cannot expect that received codeword y is corrupted only on a small number of locations and rest of the codeword is perfectly received, i.e., without any noise or error. In [21] Candès and Randall proposed a robust error correction scheme, in which they assumed that received codeword is corrupted at some locations by gross errors (sparse vector) and in addition to that all entries of the codeword are contaminated with a small amount of noise (e.g., quantization). The system model can be written as

$$y = Ax + e + q_y, \tag{25}$$

where A is an $m \times n$ coding matrix (with $m \gg n$), e is assumed to be sparse error vector and q_y is a vector consisting of small errors (noise) spread over all entries of the codeword Ax. We propose a similar scheme for decoding, which closely resembles the model for second order cone program in [21] and can be converted into an unconstrained program similar to Lasso in (1). Our proposed optimization problem to recover x from corrupted received codeword y in (25) is as follows

$$\begin{aligned} \underset{\widetilde{x},\widetilde{e},\widetilde{q}_y}{\text{minimize}} \quad & \tau\|\widetilde{e}\|_1 + \frac{1}{2}\|\widetilde{q}_y\|_2^2 \\ \text{subject to} \quad & A\widetilde{x} + \widetilde{e} + \widetilde{q}_y = y, \end{aligned} \tag{26}$$

which is equivalent to the following unconstrained problem

$$\underset{\widetilde{e}}{\text{minimize}} \quad \tau\|\widetilde{e}\|_1 + \frac{1}{2}\|Q(\widetilde{e} - y)\|_2^2, \tag{27}$$

where $Q := I - A(A^TA)^{-1}A^T$ is the projection matrix. This problem can be efficiently solved to estimate e for any value of $\tau > 0$. The decoded message $\widehat{x}$ in turn can be found using the solution $\widehat{e}$ of (27) as: $\widehat{x} = (A^TA)^{-1}A^T(y - \widehat{e})$.

Now we want to introduce the same concept of dynamic measurements to this decoding scheme. Assume that we receive one new element[2] of codeword for the system in (25); given as $w := bx + d + q_w$, where b is the new row in the coding matrix, d is the new element in the sparse error vector and q_w is the new element in small noise vector. The homotopy formulation for the updated decoding problem can be written as

$$\begin{aligned} \text{minimize} \quad & \tau(\|\widetilde{e}\|_1 + \epsilon|\widetilde{d}|) + \frac{1}{2}(\|\widetilde{q}_y\|_2^2 + |\widetilde{q}_w|^2) \\ \text{subject to} \quad & A\widetilde{x} + \widetilde{e} + \widetilde{q}_y = y \\ & b\widetilde{x} + \widetilde{d} + \widetilde{q}_w = w, \end{aligned} \tag{28}$$

[2] This scheme can also be generalized to multiple new measurements.

where again $\epsilon \in [0, 1]$ is the homotopy parameter. It can easily be seen that at $\epsilon = 0$ the problem in (28) is equivalent to (26), and as ϵ is increased towards 1, the solution of (28) traces a homotopy path towards the solution of updated system. Similar to (27) we can form a Lasso type equivalent problem to (28), which can be written as

$$\underset{\widetilde{e},\widetilde{d}}{\text{minimize}} \;\; \tau(\|\widetilde{e}\|_1 + \epsilon|\widetilde{d}|) + \frac{1}{2}\left\| P\left(\begin{bmatrix}\widetilde{e}\\ \widetilde{d}\end{bmatrix} - \begin{bmatrix}y\\ w\end{bmatrix}\right)\right\|_2^2, \quad (29)$$

where $P := I - G(G^TG)^{-1}G^T$ and $G := \begin{bmatrix}A\\ b\end{bmatrix}$. Using optimality conditions for (29) (similar to Lasso) we can develop a homotopy algorithm, which is similar to Algorithm 1.

VI. Conclusion

In this paper we have discussed the homotopy algorithms for dynamic update of measurements in ℓ_1 norm minimization problems. The main advantage of these methods is low computational cost, where each homotopy step costs a few matrix vector multiplications. In addition, the empirical results suggest that number of homotopy steps required for solution update is also small.

References

[1] E. Candes, J. Romberg, and T. Tao, "Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information," *Information Theory, IEEE Transactions on*, vol. 52, no. 2, pp. 489–509, Feb. 2006.

[2] D. Donoho, "Compressed sensing," *Information Theory, IEEE Transactions on*, vol. 52, no. 4, pp. 1289–1306, April 2006.

[3] E. Candes and T. Tao, "Decoding by linear programming," *Information Theory, IEEE Transactions on*, vol. 51, no. 12, pp. 4203–4215, Dec. 2005.

[4] E. Candes, "Compressive sampling," *Proceedings of the International Congress of Mathematicians, Madrid, Spain*, vol. 3, pp. 1433–1452, 2006.

[5] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM Journal on Scientific Computing*, vol. 20, no. 1, pp. 33–61, 1999.

[6] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society, Series B*, vol. 58, no. 1, pp. 267–288, 1996.

[7] S.-J. Kim, K. Koh, M. Lustig, S. Boyd, and D. Gorinevsky, "An interior-point method for large-scale ℓ_1-regularized least squares," *Selected Topics in Signal Processing, IEEE Journal of*, vol. 1, no. 4, pp. 606–617, 2007.

[8] M. Figueiredo, R. Nowak, and S. Wright, "Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems," *Selected Topics in Signal Processing, IEEE Journal of*, vol. 1, no. 4, pp. 586–597, 2007.

[9] M. Osborne, B. Presnell, and B. Turlach, "A new approach to variable selection in least squares problems," *IMA Journal of Numerical Analysis*, vol. 20, no. 3, pp. 389–403, 2000.

[10] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani, "Least angle regression," *Annals of Statistics*, vol. 32, no. 2, pp. 407–499, 2004.

[11] D. Malioutov, M. Cetin, and A. Willsky, "Homotopy continuation for sparse signal representation," *IEEE International Conference on Acoustics, Speech, and Signal Processing,*, vol. 5, pp. v/733–v/736, March 2005.

[12] M. Hayes, *Statistical Digital Signal Processing and Modeling*. John Wiley & Sons, Inc. New York, NY, USA, 1996.

[13] R. Vanderbei, *Linear Programming: Foundations and Extensions*. Kluwer Academic Publishers, 2001.

[14] P. J. Garrigues and L. E. Ghaoui, "An homotopy algorithm for the Lasso with online observations," *To Appear in Neural Information Processing Systems (NIPS) 21*, 2008.

[15] J. Fuchs, "On sparse representations in arbitrary redundant bases," *Information Theory, IEEE Transactions on*, vol. 50, no. 6, pp. 1341–1344, 2004.

[16] G. Golub and C. Van Loan, *Matrix Computations*. Johns Hopkins University Press, 1996.

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

[18] M. Rudelson and R. Vershynin, "The Littlewood–Offord problem and invertibility of random matrices," *Advances in Mathematics*, 2008.

[19] E. Candes and T. Tao, "The Dantzig selector: Statistical estimation when p is much larger than n," *Annals of Statistics*, vol. 35, no. 6, pp. 2313–2351, 2007.

[20] M. S. Asif, "Primal Dual Pursuit: A homotopy based algorithm for the Dantzig selector," Master's thesis, Georgia Institute of Technology, August 2008.

[21] E. J. Candes and P. A. Randall, "Highly robust error correction by convex programming," *Information Theory, IEEE Transactions on*, vol. 54, no. 7, pp. 2829–2840, 2008.