FAST OPTIMAL AND SUBOPTIMAL ALGORITHMS FOR SPARSE SOLUTIONS TO LINEAR INVERSE PROBLEMS

G. Harikumar, Christophe Couvreur, and Yoram Bresler

Coordinated Science Laboratory
and Department of Electrical and Computer Engineering
University of Illinois at Urbana-Champaign

ABSTRACT

We present two "fast" approaches to the NP-hard problem of computing a maximally sparse approximate solution to linear inverse problems, also known as best subset selection. The first approach, a heuristic, is an iterative algorithm globally convergent to sparse elements of any given convex, compact $\mathcal{S} \subset \mathbb{R}^{m_x}$. We demonstrate its effectiveness in bandlimited extrapolation and in sparse filter design. The second approach is a polynomial-time greedy sequential backward elimination algorithm. We show that if $\mathbf{A}$ has full column rank and ϵ is small enough, then the algorithm will find the sparsest $\mathbf{x}$ satifying $\|\mathbf{Ax} - \mathbf{b}\| \leq \epsilon$, if such exists.

1. INTRODUCTION

A vector $\mathbf{x} \in \mathbb{R}^{m_x}$ is said to be sparse if a significant fraction of its components is zero. In this paper, we discuss two techniques to solve the so called *best subset selection problem:*

$$\text{compute the sparsest } \mathbf{x} \text{ satisfying: } \quad \|\mathbf{Ax} - \mathbf{b}\| \leq \epsilon \qquad (1)$$

for some $\|.\|$ and $\epsilon > 0$. This is an important problem arising naturally in a wide range of scientific and engineering scenarios, including the regularization if ill-posed problems (where the sparsity constraint can be either on the signal itself [1] or on its gradient [2]), matrix computations [3], statistical modeling [4], function interpolation [5], and many other signal processing applications [6, 7]. Whereas optimal algorithms are in general impractical because the problem is N-P hard [5], its importance calls for heuristic techniques that work well for specific instances.

The first of our techniques is a recently developed [8, 9], heuristic-based, iterative algorithm, converging globally to sparse elements of any given convex, compact $\mathcal{S} \subset \mathbb{R}^{m_x}$. We give further theoretical results on the convergence properties of this algorithm, and demonstrate its effectiveness in band-limited extrapolation and in sparse filter design. The second approach is a polynomial-time greedy sequential backward elimination algorithm for Problem (1). We show that if $\mathbf{A}$ has full column rank and ϵ is small enough, then the algorithm will find the sparsest $\mathbf{x}$ satisfying $\|\mathbf{Ax}-\mathbf{b}\|$, if such exists. Thus, under the stated conditions, the algorithm provides an *optimum* solution to the subset selection problem in polynomial time. The significance of this result, its relation to the NP-hardness result, and its applications to statistical decision problems are discussed.

Dr Ir Christophe Couvreur is a Research Assistant of the National Fund for Scientific Research of Belgium (F.N.R.S.). His work is also supported in part by a Fellowship of the Belgian American Educational Foundation (Francqui Foundation Fellowship).

The work of Y. Bresler and G. Harikumar was supported in part by NSF Grant MIP MIP 91-57377, and a Schlumberger-Doll research grant. G. Harikumar is now with Tellabs Research.

2. ITERATIVE ALGORITHM

The *sparsest element problem* of finding the sparsest element $\mathbf{x}_s$ of a convex set $\mathcal{S} \subset \mathbb{R}^{m_x}$, which is a generalization of the one we seek to solve, can be stated mathematically as

$$\mathbf{x}_s = \arg\min_{\mathbf{x}\in\mathcal{S}} f(\mathbf{x}); \quad f(\mathbf{x}) = \textstyle\sum_{i=1}^{m_x} \psi(|x_i|); \qquad (2)$$

where $\psi(t) = 0$ for $t = 0$, 1 else. The first of our techniques is a fast method for solving a relaxation of Problem (2). This is achieved by approximating f by $\xi(\mathbf{x}) = \sum_i \phi(x_i)$ over $\mathcal{S}$, where $\phi(r)$ is a strictly concave, monotone increasing function of $|r|$ and $\phi(0) = 0$. (As argued there, relaxations with a convex $\phi(r)$, including l_1 relaxations, can yield non-sparse solutions.) The resulting non-convex minimization is achieved by the following iterative algorithm. Let $D_i : \mathbb{R}^{m_x} \mapsto \mathbb{R}^+$, $1 \leq i \leq d$, be continuously differentiable, convex functions with $D_i(\mathbf{0}) = 0 \; \forall \; i$, such that $\sum_{i=1}^{d} e_i D_i(\mathbf{x})$ is a strictly convex function of $\mathbf{x}$ for all positive e_i. Define the continuously differentiable, integrable, and strictly decreasing function $\rho : \mathbb{R}^+ \mapsto [0, a]$. Then we have

Sparseness Algorithm $\mathbb{A}$

1. Start from $\mathbf{x}_0 \in \mathbb{R}^{m_x}$.
2. $e_{n;i} = \rho\left(D_i(\mathbf{x}_n)\right)$.
3. $\mathbf{x}_{n+1} = \arg\min_{\mathbf{x}\in\mathcal{S}} \sum_{i=1}^{d} e_{n;i} D_i(\mathbf{x})$.
4. If convergence criterion not met, go to Step 2.

As we show later, the ρ and D_i determine $\phi()$ and have to be chosen appropriately to yield the desired shape for it. For a different interpretation, note that, because ρ is decreasing, the smaller $D_i(\mathbf{x}_n)$ are weighted more heavily, and each iterate tends to drive them further toward zero, encouraging solutions with sparse $D_i(\mathbf{x})$. From this point of view, our approach turns out to be a generalization of a similar technique proposed [10] for the case where $\mathcal{S}$ is a linear variety.

A natural choice to make $\mathbf{x}$ sparse is $d = m_x$ and $D_i(\mathbf{x}) = x_i^2$. As a different example, to solve for an image with a sparse-edge map, set $D_i(\mathbf{x}) = \mathbf{x}^T \mathbf{W}_i \mathbf{x}$, where $\mathbf{W}_i$ is an appropriate gradient operator.

2.1. Convergence

Let $\phi : [0, \infty) \mapsto [0, \infty)$ be defined by $\phi'(t) = \rho(t)$, $\phi(0) = 0$, and define the cost function

$$J(\mathbf{x}) = \sum_{i=1}^{d} \phi(D_i(\mathbf{x})) \tag{3}$$

We denote by $\mathfrak{G}$ the set of fixed points of $\mathbb{A}$.

Theorem 1 *Let $\mathcal{S}$ be a convex and compact subset of $\mathbb{R}^{m_x}$, defined by $g_i(\mathbf{x}) \leq 0$, $1 \leq i \leq m_g$, where the $g_i()$ are twice (partially) differentiable everywhere in $\mathcal{S}$. Let $\{\mathbf{x}_n\}$ be a sequence of iterates generated by Algorithm $\mathbb{A}$. Then,*

1. *$\mathfrak{G}$ consists of stationary points of J over $\mathcal{S}$; it contains all local minima and no saddle points.*
2. *The sequence $J(\mathbf{x}_n)$ is non-increasing, and strictly decreasing as long as $\mathbf{x}_n \notin \mathfrak{G}$.*
3. *Any convergent subsequence of $\{\mathbf{x}_n\}$ converges to a local extremum of J over $\mathcal{S}$.*
4. *If $\mathbf{0} \in \mathcal{S}$, the algorithm converges to $\mathbf{0}$ from all starting points. If $\mathbf{0} \notin \mathcal{S}$, all fixed points lie on the boundary of $\mathcal{S}$.*
5. *If the local extrema of J over $\mathcal{S}$ are isolated, the sequence $\{\mathbf{x}_n\}$ converges.*

Thus our algorithm is a descent technique: the sequence $J(\mathbf{x}_n)$ decreases until convergence. However, Algorithm $\mathbb{A}$ has two important advantages over conventional descent-based techniques like gradient descent. First, as is well known from classical examples, descent-based algorithms often get trapped in the large, almost flat regions of the cost function, owing to numerical difficulties. Non-convexity further aggravates the situation. Instead, at every iterate in Algorithm $\mathbb{A}$ we are minimizing a convex function over a convex set – a much easier problem, admitting a single global optimum. In fact, closed-form (or cheaply computable) solutions for the next iterate often exist, as in the important cases when $\mathcal{S}$ is a subset of a linear variety, or a hyper-ellipsoid [9]. Even with more involved set shapes, appropriate choice of the D_i (e.g., quadratic) can make each iteration very easy. For example, when $\mathcal{S}$ is a simplex, the global minimum over $\mathcal{S}$ is computed cheaply using quadratic programming. Second, numerical studies and theoretical results (below) for the case when $\mathcal{S}$ is a linear variety show that Algorithm $\mathbb{A}$ has much better overall convergence properties than gradient descent or other standard optimization algorithms.

The strongest result of Theorem 1 holds when the extrema of J are isolated. We now argue that for all practical purposes, this is always true. In view of Results 1 and 4 of Theorem 1 the set $\mathfrak{G}$ of fixed points is a subset of those points of $\mathcal{S}$ where the isocontours of J just touch the feasible set $\mathcal{S}$. Suppose the extrema are not all isolated. Then they must contain a continuum of points (a manifold in $\mathbb{R}^{m_x}$). The boundary of $\mathcal{S}$ and a single isocontour of J must coincide on this manifold. But this can only happen in pathological cases, in which the function ρ is chosen to be especially "compatible" with $\mathcal{S}$. Normally therefore, unless ρ is "maliciously" chosen, the fixed points are isolated.[1]

Although Theorem 1 does not rule out the possibility of $\mathfrak{G}$ including some local maxima of J, the algorithm will never converge to such a point, unless initialized there. This is so, owing to the strict descent property 2. Furthermore, even if initialized at a maximum , it is an unstable fixed point, and any perturbation e.g., due to finite word-length, would drive the iterates away.

We conclude *that except for pathological ρ, Algorithm $\mathbb{A}$ always converges to a local minimum of J, from any starting point.*

The rate of convergence of Algorithm $\mathbb{A}$ depends in general on the shape of the set $\mathcal{S}$ and the functions ρ and D_i. We present an analysis for the simple but important case when $\mathcal{S}$ is a linear variety $\mathcal{S} = \{\mathbf{x} : \mathbf{A}\mathbf{x} = \mathbf{b}\}$ and $D_i(\mathbf{x}) = x_i^2$. Let $\mathbf{x}^* \in \mathcal{S}$ and let $\mathbf{A}^\diamond$ be the sub-matrix of $\mathbf{A}$ whose columns correspond to non-zero elements of $\mathbf{x}^*$. Then $\mathbf{x}^*$ is defined to be *irreducible* if $\mathbf{A}^\diamond$ has full column rank. As its name suggests, an irreducible sparse point $\mathbf{x}^*$ can not be replaced by an even sparser solution obtained by setting certain components of $\mathbf{x}^*$ to zero.

Theorem 2 *Let $\mathcal{S} = \{\mathbf{x} : \mathbf{A}\mathbf{x} = \mathbf{b},\ |x_i| < r\}$ for some large r, $d = m_x$ and $D_i(\mathbf{x}) = x_i^2$. Let $\{\mathbf{x}_n\}$ be a sequence of iterates of Algorithm $\mathbb{A}$. Let $\mathbf{x}^* = [(\mathbf{x}^{1*})^T\ (\mathbf{x}^{2*})^T]^T$ be a an irreducible sparse point of convergence of the algorithm having m non-zero elements, such that, without loss of generality, $x_i^{1*} \neq 0\ \forall\ 1 \leq i \leq m$, and $\mathbf{x}^{2*} = \mathbf{0}$. Then, there exists an integer N and a constant C_1 such that for all $n > N$*

$$\|\mathbf{x}_{n+1} - \mathbf{x}^*\| \leq C_1 \left(\min_{m+1 \leq i \leq m_x} \rho(x_{n;i}^2) \right)^{-1}. \tag{4}$$

By choosing ρ appropriately, it is possible to control the rate of convergence. For example, suppose that for some $p \geq 1$, $\rho(t) \geq 1/t^p$ for $t \geq \delta$, where $\delta > 0$ is such that any component x_i of $\mathbf{x} \in \mathcal{S}$ satisfying $|x_i| < \delta$ can be considered 0. The behavior of $\rho(t)$ for $0 \leq t < \delta$ is chosen to satisfy our assumptions. Then it readily follows from (4) that provided $\max_{m+1 \leq i \leq m_x} x_{n;i}^2 \geq \delta^2$

$$\|\mathbf{x}_{n+1} - \mathbf{x}^*\| \leq C \|\mathbf{x}_n - \mathbf{x}^*\|^{2p}. \tag{5}$$

Thus this choice of ρ yields convergence of order $2p$ (at least to within distance δ from the sparse point).

We caution that larger values of p are not necessarily always better, because they affect not only the convergence rate, but also the shape of the cost function J and the basin of attraction of the algorithm. The combined effect is not always easy to predict. In examples with random initializations we found [9] that faster convergence rate may sometimes offset a decrease in success rate, in terms of total number of algorithm iterations to first success.

2.2. Bandlimited Extrapolation

In this well-known problem, the goal is to recover a vector $\mathbf{x}$ having spectral support in Ω from $\mathbf{P}_\Lambda \mathbf{x}$, where $\mathbf{P}_\Lambda$ is the projection operator on to the set of vectors band-limited to $\Lambda \subset \Omega$. In general, such reconstructions are unstable in the presence of noise, with the instability increasing as the measure of Λ decreases. Stable solution of such problems requires additional prior knowledge about $\mathbf{x}$. In particular, stable recovery is theoretically possible [1] under the assumption that $\mathbf{x}$ is sparse.

Consider the recovery of a vector $\mathbf{x}$ from its convolution $\mathbf{y} = \mathbf{h} * \mathbf{x}_b + \eta = \mathbf{A}(\mathbf{h})\mathbf{x}_b + \eta$ with a known vector $\mathbf{h} \in \mathbb{R}^{m_h}$, corrupted by the noise $\eta \sim \mathcal{N}(\mathbf{0}, \sigma\mathbf{I})$. $\mathbf{A}(\mathbf{h})$ is the corresponding convolution matrix. It is assumed that $\mathbf{x}$ is sparse, with each component bounded (for example by t). Hence, we search for the maximally sparse solutions in the set

$$\mathcal{S} = \{\mathbf{z} \in \mathbb{R}^{m_x} : \|\mathbf{A}(\mathbf{h})\mathbf{z} - \mathbf{y}\|^2 \leq \epsilon, |z_i| \leq t\ \forall\ i\}, \tag{6}$$

[1] For example, if $\mathcal{S}$ is a simplex, the extrema of J lie on its vertices for any ρ satisfying our assumptions. Likewise, if $\mathcal{S}$ is an ellipsoid and the D_i chosen to be quadratic, this will always be true unless ρ is chosen to have an appropriate linear segment.

where ϵ is chosen to be a large enough multiple of σ^2 that $\mathbf{x} \in \mathcal{S}$ with high probability.

We demonstrate for the case when $\mathbf{h} \in \mathbb{R}^{30}$ is a low-pass filter with cutoff frequency 0.5π (Fig. 1(b), heavy line). The resulting $\mathbf{A}(\mathbf{h})$ has full column rank, but is very ill-conditioned. In this example, $\mathbf{x}_b \in \mathbb{R}^{m_x}$, $m_x = 30$, $\sigma = 0.01$, $t = 5$ and $\epsilon = 10(m_x + m_h - 1)\sigma^2$, so that the true $\mathbf{x}_b \in \mathcal{S}$ with $> 99\%$ probability. The test data $\mathbf{y}$ are synthesized with $\mathbf{x}_b$ with just 6 non-zero elements (Fig 1(a)). Its DTFT (Fig. 1(b), light line) shows significant content above the cut-off frequency of $\mathbf{h}$.

In 20 trials with different random initializations Algorithm $\mathbb{A}$ always converged to an $\mathbf{x}$ with at most 8 non-zero elements. It converged to $\mathbf{x}$ with the correct support in 8 trials, to $\mathbf{x}$ with one spurious element in 7 trials, and to $\mathbf{x}$ with two spurious elements in the remaining 5 trials. Representative examples are shown in Figs. 1(c)-(e). All the points of convergence of Algorithm $\mathbb{A}$ are reasonable approximations to $\mathbf{x}_b$. In contrast, the least-squares solution (Fig. 1(f)), is neither sparse nor similar to $\mathbf{x}_b$.

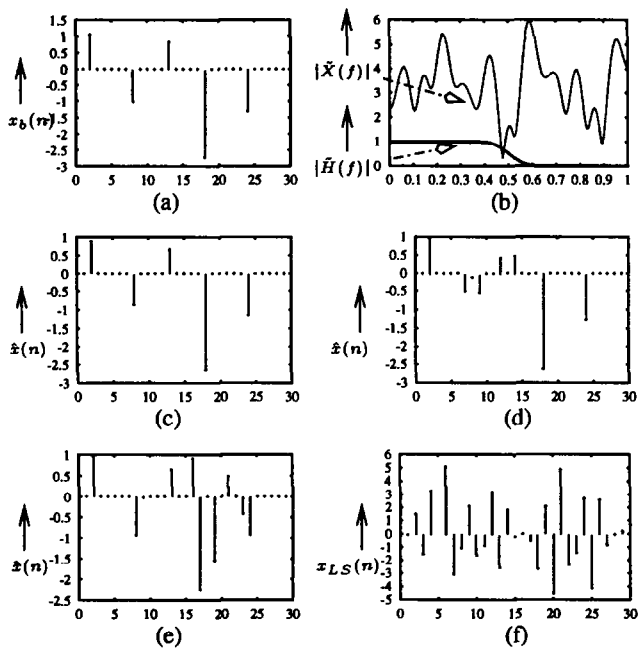

Figure 1: Bandlimited extrapolation using the sparseness algorithm. (a) $\mathbf{x}_b$. (b) DTFTs of $\mathbf{h}$ and $\mathbf{x}_b$. (c)–(e) Representative examples of points of convergence of the sparseness algorithm. (f) Least-squares estimate of $\mathbf{x}$.

2.3. Filter design example

Consider the design of a sparse length-N FIR filter $\mathbf{h}_s$ for a desired frequency response $H_d(\omega)$ within given tolerances. This is an important problem, and a variety of techniques have been proposed for it. In [11] the optimum filter is defined to be the one minimizing a cost function of the number of non-zero coefficients and their positions. The actual minimization is by mixed-integer linear programming. Because this method is globally optimal, it serves as a useful benchmark for comparison.

A sparse filter approximating $H_d(\omega)$ in a min-max sense to prescribed tolerances can be designed [9] by the using Algorithm $\mathbb{A}$ to find the sparsest element of a polytope in $\mathbb{R}^N$ defined by the frequency response specifications. Simulations show that the algorithm's performance matches that of the optimal one, needing just a fraction of the computation.

3. THE BACKWARD GREEDY ALGORITHM FOR SUBSET SELECTION

For this algorithm we begin with the following formulation of the subset selection problem. Given a full rank data matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$, $m \geq n$, and an observation vector $\mathbf{y} \in \mathbb{R}^m$, find the best least-squares solution to $\mathbf{Ax} = \mathbf{y}$ with at most r non-zero components. Note that the problem can be alternately viewed as estimating $\mathbf{x}_0 \in \mathbb{R}^n$ from $\mathbf{y}$ and the prior knowledge that $\mathbf{x}_0$ is sparse when $\boldsymbol{\eta} \in \mathbb{R}^m$ is an unknown noise vector and

$$\mathbf{y} = \mathbf{A}\mathbf{x}_0 + \boldsymbol{\eta} = \mathbf{y}_0 + \boldsymbol{\eta}. \tag{7}$$

3.1. The Backward Greedy Algorithm (BGA)

The well-known "greedy" heuristic of Golub and Van Loan [3] is a sequential forward selection scheme that is often used to compute sparse solutions to least-squares problems. The idea is to start by finding the column of $\mathbf{A}$ closest to $\mathbf{y}$, and then add columns one by one until r columns have been selected, each time adding the column that gives the largest decrement of the least-squares residual. Rather than adding columns to the solution one by one, it is also possible to start with all columns present (i.e., with the complete matrix $\mathbf{A}$) and remove one column at a time until r columns are left. That is, the subset selection can also be performed in a sequential backward fashion. The column that is removed at each step is chosen to minimize the increment in the least-squares residual. We call this alternative approach the "backward greedy algorithm" (BGA). To avoid confusion, the standard greedy algorithm will be called the "forward greedy algorithm."

Formally, the BGA can be defined as follows.
Let $\Gamma = \{1, \ldots, n\}$ denote the ordered set of column indices of $\mathbf{A}$, and $\Xi = \{\xi_1, \ldots, \xi_{c(\Xi)}\}$, $\Xi \subseteq \Gamma$, an ordered subset of Γ, of cardinality $c(\Xi)$. The "colon" notation $\mathbf{A}(\Xi, :)$ is used to designate the matrix formed from the columns of $\mathbf{A}$ whose indices are in Ξ. Denote by

$$\rho(\Xi) = \min_{\mathbf{z} \in \mathbb{R}^{c(\Xi)}} \|\mathbf{A}(\Xi, :)\mathbf{z} - \mathbf{y}\|_2$$

the least-squares residual associated with the sparse LS solution of $\mathbf{Ax} = \mathbf{y}$ based on the Ξ-indexed subset of columns of $\mathbf{A}$. The BGA for subset selection is initialized by taking $\Xi = \Gamma$. Elements are then removed from Ξ one by one by repeating the iteration

$$\Xi \leftarrow \Xi \setminus \{k^*\} : k^* = \arg\min_{k \in \Xi} \rho(\Xi \setminus \{k\}) \tag{8}$$

until $c(\Xi) = r$. The column k^* that is removed at each iteration is chosen to minimize the increment in the least-squares residual ρ. Once the last iteration has been performed, the sparse least-squares solution associated with the column indices left in Ξ is computed. The subset of indices obtained at the last iteration of the BGA and the associated sparse least-squares solution will be denoted by Ξ_s and $\mathbf{x}_s$ respectively.

3.2. Implementation and Computational Cost

The forward greedy algorithm is usually implemented by means of a QR algorithm for least-squares solution of linear systems [3, 5]. Similarly, the BGA (8) can be efficiently implemented by combining the QR algorithm with a column-deletion QR downdating step based on Givens rotations [12] performed greedily. A description of such implementation can be found in [13].

Because the BGA starts with n columns and removes them one by one until only r columns are left, if r is small with respect to n, the computational cost of the BGA will be higher than that of the forward greedy algorithm. Conversely, if r is close to n, the cost of the backward algorithm is smaller than that of the forward algorithm. Indeed, in the case of the QR-based implementation, it can shown [13] that the computational cost is roughly $O((m + (n-r)n)n^2)$. For comparison, the cost of the QR-based forward greedy algorithm is $O(mnr)$ [3].

3.3. Optimality of the Backward Greedy Algorithm

The main result of this paper, presented in Theorem 3, is that, for any full rank matrix $\mathbf{A}$ and any sparse vector $\mathbf{x}_0$, there exists a bound on the perturbation η that guarantees that the BGA will select the correct subset of components, i.e., $\Xi_s = \Xi_0$. This means that the BGA is optimal for linear inverse problems solved with a sparsity constraint, at least for small "noise levels."

Theorem 3 ([13]) *For any full column rank matrix $\mathbf{A}$ and for any sparse vector x_0 with r non-zero components (or, alternately, for any $\mathbf{y}_0 = \mathbf{A}\mathbf{x}_0$), there exists $\delta > 0$ such that $\|\mathbf{y} - \mathbf{y}_0\| < \delta$ guarantees that the backward greedy algorithm for solving $\mathbf{A}\mathbf{x} = \mathbf{y}$ will select the correct subset of components Ξ_0. Furthermore, the corresponding sparse least-squares solution $\mathbf{x}_s$ satisfies $\|\mathbf{x}_s - \mathbf{x}_0\| \leq \delta/\sigma_{\min}[\mathbf{A}(\Xi_0, :)]$ where $\sigma_{\min}[\mathbf{A}]$ denotes the smallest singular value of $\mathbf{A}$.*

Our specific motivation for studying the BGA for subset selection arises from an estimation and classification problem in statistical signal processing [6]. In this application, the vector $\mathbf{y}$ is a random vector converging with probability one to $\mathbf{y}_0$. We are interested in the properties of the estimate of Ξ_0 and $\mathbf{x}_0$ obtained by the BGA. Note that for the application described in [6], finding Ξ_0 is at least as important as finding $\mathbf{x}_0$. As Corollary 1 establishes, the estimates Ξ_s and $\mathbf{x}_s$ obtained by the BGA are strongly consistent. That is, they converge to the true values Ξ_0 and x_0 with probability one.

Corollary 1 *If $\mathbf{y}$ is a strongly consistent estimator of $\mathbf{y}_0$, i.e., if $\mathbf{y} \to \mathbf{y}_0$ w.p.1, then $\Xi_s \to \Xi_0$ and $\mathbf{x}_s \to \mathbf{x}_0$ w.p.1.*

The BGA can be easily modified to solve Problem (1). The modified version of the algorithm simply needs a new stopping criterion. Instead of stopping at a predefined number r of components, it must now be stopped for the smallest r satisfying $\rho(\Xi_s) < \epsilon$. For this version of the algorithm, we have the following

Corollary 2 ([13]) *Given $\mathbf{A}$ and $\mathbf{y}$, there exists a bound $\delta > 0$ such that, if $\epsilon < \delta$, then the backward greedy algorithm (8) provides the optimal solution to Problem (1).*

The implication of Corollary 2 is clear: under the stated conditions, it is possible to find the *optimum* solution to Problem 1 in polynomial time. The contradiction with Natarajan's result [5] on NP-hardness is only apparent. Note that if $\mathbf{y} = \mathbf{y}_0$, the system of equation $\mathbf{A}\mathbf{x} = \mathbf{y}$ has an *exact* solution with some of its components equal to zero. Thus, if $\mathbf{y} = \mathbf{y}_0$, it is not necessary to perform any subset selection; solving $\mathbf{A}\mathbf{x} = \mathbf{y}_0$ and checking for the non-zero components yields directly the correct sparse solution $\mathbf{x}_s$. Thus, when $\mathbf{y} = \mathbf{y}_0$, the problem admits obviously a polynomial-time solution. Similarly, Corollary 2 states that here exists δ such that the optimal sparse solution to Problem 1 is found by the BGA in polynomial time if $\epsilon < \delta$. However, the bound δ is not given directly; it is a complex function of A and $\mathbf{y}$. Furthermore, computing δ is not a polynomial time operation [13].

4. REFERENCES

[1] D. L. Donoho, "Superresolution via sparsity constraints," *SIAM J. Math. Anal.*, vol. 23, pp. 1309–1331, Sep. 1992.

[2] A. H. Delaney and Y. Bresler, "Globally convergent edge preserving regularization: An application to limited angle tomography," *IEEE Trans. Image Process.,*, vol.7 Feb. 1998.

[3] G. H. Golub and C. F. Van Loan, *Matrix Computations*, Johns Hopkins Univ. Press, Baltimore, ML, 2nd ed., 1989.

[4] A. J. Miller. *Subset Selection in Regression.* Chapman and Hall, London, UK, 1990.

[5] B. K. Natarajan, "Sparse approximate solutions to linear systems", *SIAM J. Comp.*, vol. 24, pp. 227–234, 1995.

[6] C. Couvreur and Y. Bresler. Dictionary-based decomposition of linear mixtures of Gaussian processes. *Proc. ICASSP*, pp. 2519–2522, May 1996.

[7] M. Nafie, M. Ali, and A. H. Tewfik. Optimal susbet selection for adaptive signal representation. *Proc. ICASSP*, May 1996.

[8] G. Harikumar and Y. Bresler, "A new algorithm for computing sparse solutions to linear inverse problems," *Proc. ICASSP*, May 1996, vol. III, pp. 1331–1334.

[9] G. Harikumar and Y. Bresler, "Optimum sparse approximations: a new class of algorithms," submitted, 1997.

[10] I. Gorodnitsky and B. D. Rao, "Sparse signal reconstructions from limited data using FOCUSS: A re-weighted minimum norm algorithm," *IEEE Trans. Signal Process.*, pp. 600–616, March 1997.

[11] J. T. Lim, W. J. Oh, and Y. H. Lee, "Design of nonuniformly spaced linear phase FIR filters using mixed integer linear programming," *IEEE Trans. Signal Process.*, pp. 123–126, Jan. 1996.

[12] A. Björck, H. Park, and L. Eldén, "Accurate downdating of least squares solutions", *SIAM J. Mat. An. Appl.*, vol. 15, pp. 549–568, 1994.

[13] C. Couvreur and Y. Bresler, "Optimalilty of the Backward Greedy Algorithm for the Subset Selection Problem", submitted *SIAM J. Mat. An. Appl.*, 1998.