Greedy Basis Pursuit

Patrick S. Huggins and Steven W. Zucker, *Fellow, IEEE*

***Abstract*—We introduce greedy basis pursuit (GBP), a new algorithm for computing sparse signal representations using overcomplete dictionaries. GBP is rooted in computational geometry and exploits an equivalence between minimizing the ℓ^1-norm of the representation coefficients and determining the intersection of the signal with the convex hull of the dictionary. GBP unifies the different advantages of previous algorithms: like standard approaches to basis pursuit, GBP computes representations that have minimum ℓ^1-norm; like greedy algorithms such as matching pursuit, GBP builds up representations, sequentially selecting atoms. We describe the algorithm, demonstrate its performance, and provide code. Experiments show that GBP can provide a fast alternative to standard linear programming approaches to basis pursuit.**

***Index Terms*—Computational geometry, linear programming, signal representation.**

I. Introduction

THE PROBLEM of computing sparse signal representations using an overcomplete dictionary arises in a wide range of signal processing applications [31], [52], [84], including image [7], [102], audio [40], [65], and video [3] compression and source localization [68]. The goal is to represent a given signal as a linear superposition of a small number of stored signals, called *atoms*, drawn from a larger set, called the *dictionary*. In traditional signal representation methods, such as the discrete cosine transform (DCT) or various wavelet transforms, the dictionary is simply a basis: the number of atoms in the dictionary is equal to the dimensionality of the signal space and representation is unique. By contrast, in an overcomplete dictionary the number of atoms is greater than the dimensionality of the signal space and representation is no longer unique; this enables flexibility in representation [69], "shiftability" [90], and the use of multiple bases [59], [94], but it requires a criterion to select from among the (many) possible representations. A natural one is sparsity, by which the representation selected is the one that uses as few atoms as possible.

Computing sparse representations is NP-hard [28], [75], and so several (heuristic) methods have been developed [16], [53], [69], [80]. These methods optimize various measures of sparsity, typically functions of the representation coefficients [62], [63], using, for example, greedy algorithms [69], gradient descent [66], linear programming [18], and global optimization [83]. Currently, the two most popular approaches are matching pursuit (MP) [69] and basis pursuit (BP) [17], [18].

MP is a greedy algorithm: a signal representation is iteratively built up by selecting the atom that maximally improves the representation at each iteration. While there is no guarantee that MP computes sparse representations, MP is easily implemented, converges quickly, and has good approximation properties [55], [69], [97]. Moreover, MP and one of its variants, orthogonal matching pursuit (OMP) [80], can be shown to compute sparse (or nearly sparse) representations under some conditions [55], [99].

BP, instead of seeking sparse representations directly, seeks representations that minimize the ℓ^1-norm of the coefficients. By equating signal representation with ℓ^1-norm minimization, BP reduces signal representation to linear programming [17], [18], which can be solved by standard methods [101]. Furthermore, BP methods can compute sparse solutions in situations where greedy algorithms fail [18]. Recent theoretical work shows that representations computed by BP are guaranteed to be sparse under certain conditions [33], [34], [48], [100].

While applying standard linear programming methods to compute minimum ℓ^1-norm signal representations is natural, such methods were developed with very different problems in mind and may not be ideally suited to the representation problem. For example, if the matrix corresponding to the dictionary is not sparse then the (normally fast) interior point methods advocated for BP [18] can be slow. Furthermore, the design required to produce examples on which greedy algorithms fail yet BP succeeds suggests that a greedy strategy could be successfully applied to minimum ℓ^1-norm representation.

In this paper, we develop a new algorithm for computing sparse signal representations, which we call greedy basis pursuit (GBP). GBP is an algorithm for BP: it minimizes the ℓ^1-norm of the representation coefficients. However, unlike standard linear programming methods for BP, GBP proceeds much like MP, building up the representation by iteratively selecting atoms.

While algorithmically similar to MP, GBP differs from MP in two key ways. 1) GBP uses a novel criterion for selecting the next atom in the representation. The criterion is based on computational geometry, and effects a search for the intersection between the signal vector and the convex hull of the dictionary. 2) GBP may discard atoms that it has already selected; this is crucial, as it allows GBP to overcome the "mistakes" that MP can make in atom selection when compared to BP [18].

While GBP returns the signal representation with the minimum ℓ^1-norm, and thus GBP enjoys the theoretical benefits of BP, the greedy strategy of GBP leads to computational gains when compared to standard linear programming methods. Experiments show our implementation of GBP to be faster than

Manuscript received May 30, 2006; revised October 26, 2006. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. David J. Miller. This work was supported by DARPA and AFOSR.

P. S. Huggins is with HSBC, New York, NY 10128 USA (e-mail: huggins@cs.yale.edu).

S. W. Zucker is with the Department of Computer Science, Yale University, New Haven, CT 06520 USA.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSP.2007.894287

off-the-shelf linear programming packages on some signal representation problems, particularly high-dimensional problems with very overcomplete dictionaries.

The remainder of this paper is organized as follows. In Section I-A, we formally state the sparse signal representation problem. In Section II, we review current approaches to the problem. Section III provides the geometric interpretation of BP that underlies GBP. In Section IV, we describe the GBP algorithm. Section V presents the results of experiments with GBP. We discuss GBP in Section VI and conclude in Section VII.

A. Problem Statement

Given a signal $\mathbf{x}$ and a dictionary $\mathcal{D}$ we seek a sparse representation of $\mathbf{x}$. We assume that $\mathbf{x}$ consists of d real valued measurements, that is, $\mathbf{x} \in \mathbb{R}^d$, for example, a sound wave sampled at d points. We assume that $\mathcal{D}$ consists of n atoms and is overcomplete, that is, $\mathcal{D} = \{\boldsymbol{\psi}_i\}_{i=1}^n$ and $n > d$, and that the atoms are also d-dimensional and have unit norm, that is, $\forall \boldsymbol{\psi}_i \in \mathcal{D}, \boldsymbol{\psi}_i \in \mathbb{R}^d$ and $||\boldsymbol{\psi}_i||_2 = 1$. A *representation* of $\mathbf{x}$ is a set of indices $\mathcal{I}$ into $\mathcal{D}$, where $\mathcal{I} \subseteq \{1, \ldots, n\}$, and a corresponding set of coefficients $\mathcal{A} = \{\alpha_i\}_{i \in \mathcal{I}}$, such that

$$\mathbf{x} = \sum_{i \in \mathcal{I}} \alpha_i \boldsymbol{\psi}_i. \tag{1}$$

A representation is *sparse* if the number of atoms used, $|\mathcal{I}|$ (here $|\cdot|$ denotes cardinality), is minimized over all possible representations.

Equivalently, in matrix notation, given a (column) vector $\mathbf{x} \in \mathbb{R}^d$ corresponding to the signal, and a $d \times n$ matrix $\mathbf{D}$ corresponding to the dictionary, where the ith column of $\mathbf{D}$ is the atom $\boldsymbol{\psi}_i$, the sparse signal representation problem is then to compute a (column) vector $\boldsymbol{\alpha} \in \mathbb{R}^n$ solving

$$\text{Minimize} ||\boldsymbol{\alpha}||_0 \quad \text{subject to } \mathbf{D}\boldsymbol{\alpha} = \mathbf{x} \tag{2}$$

where $||\boldsymbol{\alpha}||_0$ is the ℓ^0-norm of $\boldsymbol{\alpha}$, defined to be the number of nonzero entries of $\boldsymbol{\alpha}$. In general, the equality constraint can be relaxed to give a corresponding approximation problem (see [75], [97], [99], and [100]).

BP replaces the ℓ^0-norm with the ℓ^1-norm, seeking representations that minimize $\sum_{i \in I} |\alpha_i|$. In matrix form this corresponds to

$$\text{Minimize} ||\boldsymbol{\alpha}||_1 \quad \text{subject to } \mathbf{D}\boldsymbol{\alpha} = \mathbf{x} \tag{3}$$

GBP solves 3. The approximation problem corresponding to BP is called basis pursuit denoising (see [18], [39], and [66]).

II. Related Work

The design of GBP draws on previous work in sparse signal representation, particularly the contrast between MP and BP, and on ideas from subset selection, which we summarize here. We also highlight some unexplored connections between sparse signal representation and linear programming.

A. Matching Pursuit

MP [69] is the prototypical greedy algorithm [20] applied to sparse signal representation. MP is currently the most popular algorithm for computing sparse signal representations using an overcomplete dictionary, and is used in a variety of applications [3], [7], [81]. MP has also spawned several variants [43], [44], [60], including OMP [80], [29], which itself has several variants [21], [51], [85].

MP computes a signal representation by greedily constructing a sequence of approximations to the signal, $\tilde{\mathbf{x}}^{(0)}, \tilde{\mathbf{x}}^{(1)}, \tilde{\mathbf{x}}^{(2)}, \ldots$, where each consecutive approximation is closer to the signal. MP begins with an "empty" representation $\tilde{\mathbf{x}}^{(0)} = 0$ and at each iteration augments the current representation by selecting the atom from the dictionary which is closest to the residual $\tilde{\mathbf{x}}^{(t+1)} = \tilde{\mathbf{x}}^{(t)} + \alpha^{(t)} \boldsymbol{\psi}^{(t)}$, where $\boldsymbol{\psi}^{(t)} = \arg\max_{\boldsymbol{\psi} \in \mathcal{D}} \langle \boldsymbol{\psi}, \mathbf{x} - \tilde{\mathbf{x}} \rangle$.

MP is easy to implement, has a guaranteed exponential rate of convergence [55], [69], [97], and recovers relatively sparse solutions [99], particularly compared to earlier approaches such as the method-of-frames [18], [26].

A drawback of MP applied to sparse representation is its greediness. It is possible to construct signal representation problems where, because of its greediness, MP (or OMP) intially selects an atom that is not part of the optimal sparse representation; as a result, many of the subsequent atoms selected by MP simply compensate for the poor initial selection [18], [30]. This shortcoming motivated the development of BP, which succeeds on these problems [18]; recent theoretical work explains this phenomenon [33], [34], [48].

These problems are also motivation for the development of GBP. Here MP fails because of its poor initial selection of atoms; however, the atoms initially selected by MP are not necessarily bad in general, after all, these problems are specially designed for MP to fail on. For MP to succeed on these problems, it would need to either make "better" atom selections or be able to discard "bad" atoms to recover from poor selections (or both). GBP adapts the greedy strategy to incorporate both of these ideas and compute the same representations as BP.

B. Basis Pursuit

BP [16]–[18] approaches sparse signal representation by changing the problem to one of minimizing the ℓ^1-norm of the representation coefficients. This can be interpreted as assuming a "sparse prior" on the representation coefficients [77]. The ℓ^1-norm in particular implies that the resulting representations are sparse in the ℓ^0-norm sense under certain conditions [33], [34], [48] and algorithmically equates sparse signal representation with linear programming.

A linear program is defined as follows. Given a matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$, a (column) vector $\mathbf{b} \in \mathbb{R}^m$, and a (column) vector $\mathbf{c} \in \mathbb{R}^n$, compute a (column) vector $\mathbf{x} \in \mathbb{R}^n$ satisfying

$$\text{Minimize } \mathbf{c}^T\mathbf{x} \quad \text{subject to } \mathbf{A}\mathbf{x} = \mathbf{b}, \quad x_i \geq 0. \tag{4}$$

The signal representation problem is posed in BP as a linear program with the following assignments (the variables on the right hand side are as defined in Section I-A and the variables on the left hand side plug into the previous linear program):

$$\begin{aligned} \mathbf{A} &\leftarrow [\boldsymbol{\psi}_1\ \boldsymbol{\psi}_2\ \cdots\ \boldsymbol{\psi}_n\ -\boldsymbol{\psi}_1\ -\boldsymbol{\psi}_2\ \cdots\ -\boldsymbol{\psi}_n] \\ \mathbf{b} &\leftarrow \mathbf{x} \\ \mathbf{c} &\leftarrow [1\ 1\ \cdots\ 1] \\ \mathbf{x} &\leftarrow [\alpha_1\ \alpha_2\ \cdots\ \alpha_n]. \end{aligned}$$

Minimizing $\mathbf{c}^T\mathbf{x}$ is equivalent to minimizing the ℓ^1-norm of the coefficients. Note that $\mathbf{A}$, corresponding to the dictionary, is doubled to include the negative of each atom; this is due to the linear programming constraint that the coefficients be nonnegative.

Chen *et al.* [17], [18] describe two algorithms for BP, BP-Simplex and BP-Interior, which are the well-known simplex and interior point methods of linear programming [101] applied to signal representation. The choice of which BP algorithm to use depends on the structure of the dictionary: for dictionaries that have fast transforms, BP-interior exploits these transforms in the solution of the corresponding linear program. However, the running time of linear programming is still typically an order of magnitude slower than that of MP on realistic problems [18].

While standard linear programming methods have been highly tuned over time, they are not necessarily ideally suited to the specific problem of computing signal representations. For example, many linear programming methods assume that the matrix $\mathbf{A}$ is sparse, as is the case for constraints that arise in typical operations research problems, while this may not be the case in signal representation problems. This raises the possibility that alternative approaches could prove more efficient for the particular problem of signal representation. Some inspiration for an alternative approach is provided by Chen *et al.* [18], who contrast MP and BP-Simplex, characterizing MP as a "build-up" approach and BP-Simplex as a "swap-down" approach. If $\mathbf{A}$ is not sparse, then the swaps (or pivots) executed by BP-Simplex can be costly, in the computation of an individual swap, in the number of swaps, and in the computation of an initial basis. GBP instead takes the "build-up" approach to solving linear programming.

C. Subset Selection

Sparse signal representation is closely related to the problem of subset selection for regression, i.e., determining the optimal subset of variables on which to regress a data set [71]. In sparse signal representation, the signal corresponds to the data set, while the atoms correspond to the variables. In fact, MP was inspired by projection pursuit [47], [58], in particular its use as a regression algorithm [46]. Given this connection, it should not be surprising that some algorithmic ideas in sparse signal representation correspond to earlier work in regression. For example, in forward selection the optimal subset is constructed by starting with the empty subset and iteratively adding variables to it, selecting at each iteration the variable that accounts for most of the residual variance; this is essentially what OMP does. Backward elimination, which starts with the full set of variables and iteratively pares it down, has similarly been adapted for signal representation [22], [56].

One standard algorithm for subset selection in regression which appears to have no analogue in sparse signal representation is Efroymson's algorithm [38], also called step-wise regression, which proceeds like forward selection, but, like backward elimination, drops variables from the subset as they become irrelevant. GBP follows a similar strategy, iteratively selecting atoms and occasionally discarding them.

D. Linear Programming

While BP represents the first formal casting of signal representation as linear programming, linear programming has long been used in sparse signal representation, particularly for deconvolution in various applications [5], [37], [76]. It is, therefore, not surprising that developments in sparse signal representation closely parallel earlier developments in linear programming.

Examining the literature in linear programming reveals that MP and OMP have linear programming analogues: MP is technically equivalent to one of the earliest (1948) methods developed for linear programming, called von Neumann's algorithm [23]. Similarly, OMP is equivalent to a phase I algorithm [64] for the simplex method.

GBP builds up a solution to a linear programming problem; several linear programming methods adopt a similar strategy, solving increasingly complex problems as constraints or variables are iteratively introduced [78], [89], [95] (see also [57]). We remark that one method, an interior point method called the gravitational method [15], [74], can be shown to be equivalent to GBP when applied to the problem dual to (4). Empirically, the gravitational method is faster than standard methods on some problems [15], which is consistent with our results.

III. Geometry of Basis Pursuit

GBP is based on computatonal geometry, specifically on the following geometric interpretation of BP. Given a signal $\mathbf{x}$ and a dictionary $\mathcal{D}$, let $\mathbf{conv}(\mathcal{D})$ denote the convex hull of $\mathcal{D}$; *the vertices of the facet of* $\mathbf{conv}(\mathcal{D})$ *intersected by the vector* $\mathbf{x}$ *are the atoms in the minimum* ℓ^1*-norm representation of* $\mathbf{x}$.

To see this, treat the signal as a vector and the atoms as points in $\mathbb{R}^d$. First consider the set of signals that have representations $\boldsymbol{\alpha}$ such that $\|\boldsymbol{\alpha}\|_1 = 1$. By definition, this is the convex hull of the dictionary

$$\mathbf{conv}(\mathcal{D}) = \left\{ \mathbf{x} \,|\, \mathbf{x} = \sum_{i\in\mathcal{I}} \alpha_i \boldsymbol{\psi}_i \text{ and } \sum_{i\in\mathcal{I}} \alpha_i = 1, \alpha_i > 0 \right\}.$$

Note that because $\|\boldsymbol{\psi}_i\|_2 = 1$, $\mathbf{conv}(\mathcal{D})$ is a polytope inscribed in the unit sphere. Let $\mathbf{x}_\mathcal{D}$ be the point of intersection between the vector $\mathbf{x}$ and the boundary of $\mathbf{conv}(\mathcal{D})$. $\mathbf{x}_\mathcal{D}$ lies on the boundary of $\mathbf{conv}(\mathcal{D})$ and can be represented as a linear combination of the vertices of the facet of $\mathbf{conv}(\mathcal{D})$ containing $\mathbf{x}_\mathcal{D}$; call this facet $F_\mathbf{x}$. This representation is the minimum ℓ^1-norm representation: its ℓ^1-norm is 1, and it is impossible to construct a representation with ℓ^1-norm less than 1. The minimum ℓ^1-norm representation of $\mathbf{x}$ is simply a scaling of the minimum ℓ^1-norm representation of $\mathbf{x}_\mathcal{D}$, and the atoms in the representation are the same (see Fig. 1). (Note that if we know the atoms in a representation of $\mathbf{x}$ it is straightforward to calculate the corresponding coefficients.)

Thus, BP is equivalent to finding the facet of $\mathbf{conv}(\mathcal{D})$ which intersects $\mathbf{x}$. Computing this intersection is known to reduce to linear programming [87]; to our knowledge, the converse is known [12] but never utilized to solve linear programming. We use this equivalence to drive GBP.

A previous geometric interpretation of sparse representation [11] recognizes that in two dimensions BP computes representations with atoms that "enclose" $\mathbf{x}$. The intepretation provided here can be viewed as the generalization of this notion to higher dimensions.

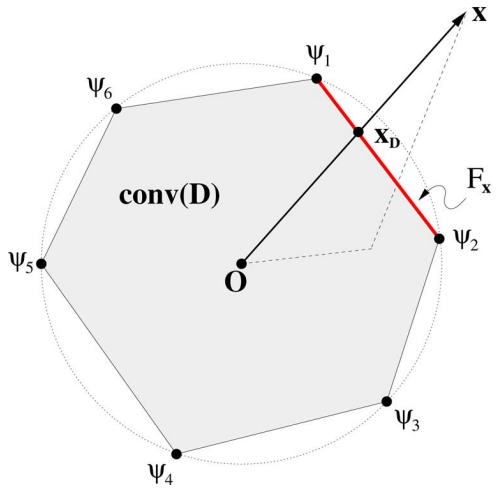


Fig. 1. *Geometric interpretation of BP.* The signal vector $\mathbf{x}$ intersects the facet $F_{\mathbf{x}}$ of the convex hull of the dictionary, shown in gray. The vertices of $F_{\mathbf{x}}$, $\boldsymbol{\psi}_1$, and $\boldsymbol{\psi}_2$, are the atoms in the BP representation of $\mathbf{x}$.

IV. Greedy Basis Pursuit Algorithm

Given the equivalence between BP and finding the facet of the convex hull of the dictionary that intersects the signal vector, we propose GBP. GBP computes the minimum ℓ^1-norm representation by searching for this facet directly.

The main idea behind GBP is to find the facet of interest by iteratively "pushing" a hyperplane onto the surface of the convex hull of the dictionary until it coincides with the supporting hyperplane containing the facet. This approach is inspired by gift-wrapping methods [14], [61], [96] for the convex hull problem in computational geometry [88]. To adapt gift-wrapping to the problem of finding a particular facet, we need to specify how the initial hyperplane is chosen and the direction in which the "wrapping" proceeds at each iteration. In the following, we describe the GBP algorithm, prove its convergence, and discuss implementation issues.

A. Main Algorithm

GBP takes as input a signal $\mathbf{x} \in \mathbb{R}^d$ and an overcomplete dictionary $\mathcal{D} = \{\boldsymbol{\psi}_i\}_{i=1}^n$, where $n > d$ and $\forall i$, $\boldsymbol{\psi}_i \in \mathbb{R}^d$ and $\|\boldsymbol{\psi}_i\|_2 = 1$, and outputs a representation of $\mathbf{x}$ as a set of indices $\mathcal{I} \subseteq \{1, \ldots, n\}$ and a corresponding set of coefficients $\mathcal{A} = \{\alpha_i\}_{i \in \mathcal{I}}$, such that $\mathbf{x} = \sum_{i \in \mathcal{I}} \alpha_i \boldsymbol{\psi}_i$. Note we assume that if $\boldsymbol{\psi}_i \in \mathcal{D}$ then $-\boldsymbol{\psi}_i \in \mathcal{D}$ (see Section II-B).

GBP greedily searches for the facet of $\mathbf{conv}(\mathcal{D})$ that intersects $\mathbf{x}$, call it $F_{\mathbf{x}}$. GBP proceeds by iteratively constructing a sequence of hyperplanes, $H^{(0)}, H^{(1)}, H^{(2)}, \ldots$, supporting $\mathbf{conv}(\mathcal{D})$. (We use the superscript (t) to denote iteration t.) At each iteration, GBP maintains a set of indices $\mathcal{I}^{(t)}$ and a set of coefficients $\mathcal{A}^{(t)}$, defining an approximation to $\mathbf{x}$: $\tilde{\mathbf{x}}^{(t)} = \sum_{i \in \mathcal{I}^{(t)}} \alpha_i \boldsymbol{\psi}_i$, and a normal vector $\mathbf{n}^{(t)}$. The current hyperplane $H^{(t)}$ is defined to have normal $\mathbf{n}^{(t)}$ and contain the set $\{\boldsymbol{\psi}_i\}_{i \in \mathcal{I}^{(t)}}$. Each consecutive hyperplane $H^{(t+1)}$ is a rotation of the current hyperplane $H^{(t)}$ determined by $\tilde{\mathbf{x}}^{(t)}$. GBP stops when $H^{(t)}$ contains $F_{\mathbf{x}}$ (and, therefore, $\tilde{\mathbf{x}}^{(t)} = \mathbf{x}$).

1) Initialization: As we do not *a priori* know the orientation of $F_{\mathbf{x}}$, we optimistically choose the initial supporting hyperplane $H^{(0)}$ to have normal $\mathbf{n}^{(0)} = \mathbf{x}/\|\mathbf{x}\|_2$. In general, $H^{(0)}$ will intersect only one vertex of $\mathbf{conv}(\mathcal{D})$, in particular, the atom $\boldsymbol{\psi}_{i_0}$, where $i_0 = \arg\max_i \langle \boldsymbol{\psi}_i, \mathbf{n}^{(0)} \rangle$. To see this, consider a hyperplane with normal $\mathbf{n}^{(0)}$ at some distance greater than one away from the origin; if we move this hyperplane in the negative normal direction (towards the origin), the first point of $\mathbf{conv}(\mathcal{D})$ it will intersect is $\boldsymbol{\psi}_{i_0}$. (Note that this is also the first atom selected by MP and OMP.) Thus $\mathcal{I}^{(0)} = \{i_0\}$; this gives us $\alpha_{i_0} = \langle \boldsymbol{\psi}_{i_0}, \mathbf{x} \rangle$, $\mathcal{A}^{(0)} = \{\alpha_{i_0}\}$, and $\tilde{\mathbf{x}}^{(t)} = \alpha_{i_0} \boldsymbol{\psi}_{i_0}$. For convenience, we denote the set of currently selected atoms by $\Psi^{(t)} = \{\boldsymbol{\psi}_i\}_{i \in \mathcal{I}^{(t)}}$.

2) Iteration: Each consecutive hyperplane $H^{(t+1)}$ is constructed by rotating $H^{(t)}$ in a 2-D plane around a pivot point until another vertex of $\mathbf{conv}(\mathcal{D})$ is intersected. The plane of rotation and the pivot point are defined in terms of $\tilde{\mathbf{x}}^{(t)}$. We define $\tilde{\mathbf{x}}^{(t)}$ to be the best current approximation to $\mathbf{x}$ using $\Psi^{(t)}$ and positive coefficients, that is, $\tilde{\mathbf{x}}^{(t)} = \sum_{i \in \mathcal{I}^{(t)}} \alpha_i \boldsymbol{\psi}_i$, where $\alpha_i > 0$ and $\|\tilde{\mathbf{x}}^{(t)} - \mathbf{x}\|_2$ is minimized. Note that $\tilde{\mathbf{x}}^{(t)}$ is the projection of $\mathbf{x}$ onto the convex cone spanned by $\Psi^{(t)}$ with the origin at the apex; we provide details on computing $\tilde{\mathbf{x}}^{(t)}$ in Section IV-A3. Let $\tilde{\mathbf{x}}_H^{(t)}$ be the intersection of the vector $\tilde{\mathbf{x}}^{(t)}$ with $H^{(t)}$. If $d_H^{(t)}$ is the (orthogonal) distance from the hyperplane to the origin, i.e., $d_H^{(t)} = \langle \boldsymbol{\psi}_i, \mathbf{n} \rangle, \forall i \in \mathcal{I}^{(t)}$, then

$$\tilde{\mathbf{x}}_H^{(t)} = \left(d_H^{(t)} / \langle \tilde{\mathbf{x}}^{(t)}, \mathbf{n}^{(t)} \rangle \right) \tilde{\mathbf{x}}. \tag{5}$$

Let $\mathbf{r}^{(t)}$ denote the residual vector, $\mathbf{r}^{(t)} = \mathbf{x} - \tilde{\mathbf{x}}^{(t)}$. Define $\mathbf{v}^{(t)}$ to be the unit vector in the direction of $\mathbf{r}^{(t)}$ projected onto $H^{(t)}$

$$\mathbf{v}^{(t)} = \frac{\mathbf{r}^{(t)} - \left\langle \mathbf{r}^{(t)}, \mathbf{n}^{(t)} \right\rangle \mathbf{n}^{(t)}}{\left\| \mathbf{r}^{(t)} - \left\langle \mathbf{r}^{(t)}, \mathbf{n}^{(t)} \right\rangle \mathbf{n}^{(t)} \right\|}. \tag{6}$$

The plane of rotation is the 2-D plane defined by the point $\tilde{\mathbf{x}}_H^{(t)}$ and the vectors $\mathbf{n}^{(t)}$ and $\mathbf{v}^{(t)}$. The pivot point around which H is rotated is $\tilde{\mathbf{x}}_H^{(t)}$.

To compute the first vertex which the hyperplane intersects under this rotation, we order the atoms by the angle θ they form with $\mathbf{v}$, where θ is given by

$$\theta_i = \arctan\left(\left\langle \boldsymbol{\psi}_i - \tilde{\mathbf{x}}_H^{(t)}, \mathbf{n}^{(t)} \right\rangle \Big/ \left\langle \boldsymbol{\psi}_i - \tilde{\mathbf{x}}_H^{(t)}, \mathbf{v}^{(t)} \right\rangle \right).$$

The atom selected is then $\boldsymbol{\psi}_k$, where

$$k = \arg\min_i \theta_i. \tag{7}$$

Once selected, the atom $\boldsymbol{\psi}_k$ is added to the set $\Psi^{(t)}$ and a new approximation to $\mathbf{x}$ is computed, $\tilde{\mathbf{x}}^{(t+1)}$. In this new approximation, some atoms in $\Psi^{(t)} \cup \{\boldsymbol{\psi}_k\}$ may become extraneous: they are discarded to form $\Psi^{(t+1)}$ (see Section IV-A3).

The new hyperplane $H^{(t+1)}$ can now be computed; it has normal

$$\mathbf{n}^{(t+1)} = \frac{-\left\langle \boldsymbol{\psi}_k - \tilde{\mathbf{x}}_H^{(t)}, \mathbf{n}^{(t)} \right\rangle \mathbf{v}^{(t)} + \left\langle \boldsymbol{\psi}_k - \tilde{\mathbf{x}}_H^{(t)}, \mathbf{v}^{(t)} \right\rangle \mathbf{n}^{(t)}}{\left\| -\left\langle \boldsymbol{\psi}_k - \tilde{\mathbf{x}}_H^{(t)}, \mathbf{n}^{(t)} \right\rangle \mathbf{v}^{(t)} + \left\langle \boldsymbol{\psi}_k - \tilde{\mathbf{x}}_H^{(t)}, \mathbf{v}^{(t)} \right\rangle \mathbf{n}^{(t)} \right\|} \tag{8}$$

and contains $\tilde{\mathbf{x}}_H^{(t+1)}$.

The procedure is repeated until $\tilde{\mathbf{x}}^{(t)} = \mathbf{x}$, that is, $H^{(t)}$ contains $F_{\mathbf{x}}$ (see Fig. 2).

Algorithm 1 GBP

Input

- A signal $\mathbf{x} \in \mathbb{R}^d$
- A dictionary $\mathcal{D} = \{\boldsymbol{\psi}_i\}_{i=1}^n$
- A threshold $\epsilon \geq 0$

Output

A representation of $\mathbf{x}$, consisting of

- A set of indices $\mathcal{I} \subseteq \{1, \ldots, n\}$
- A set of coefficients $\mathcal{A} = \{\alpha_i\}_{i \in \mathcal{I}}$

such that $\mathbf{x} - \sum_{i \in \mathcal{I}} \alpha_i \boldsymbol{\psi}_i < \epsilon$

Procedure

1) Initialize
   a) Select the first atom
   $$k \leftarrow \arg \max_{i \in \{1,\ldots,n\}} \langle \mathbf{x}, \boldsymbol{\psi}_i \rangle$$
   b) Compute the initial approximation
   $$\alpha_k \leftarrow \langle \mathbf{x}, \boldsymbol{\psi}_k \rangle, \mathcal{I}^{(0)} \leftarrow \{k\}, \mathcal{A}^{(0)} \leftarrow \{\alpha_k\}.$$
   c) Initialize the biorthogonal system
   $$\tilde{\Psi}^{\perp} \leftarrow \{\boldsymbol{\psi}_k\}.$$
   d) Initialize the hyperplane
   $$\tilde{\mathbf{x}}^{(0)} \leftarrow \alpha_k \boldsymbol{\psi}_k, \mathbf{n} \leftarrow \mathbf{x}/\|\mathbf{x}\|, \mathbf{r} \leftarrow \mathbf{x} - \tilde{\mathbf{x}}.$$
2) Repeat until $\|\mathbf{r}\| < \epsilon$
   a) Compute the center and plane of rotation
   $$\tilde{\mathbf{x}}_H \leftarrow (\langle \boldsymbol{\psi}_i, \mathbf{n} \rangle / \langle \tilde{\mathbf{x}}, \mathbf{n} \rangle) \tilde{\mathbf{x}}, \text{ for any } i \in \mathcal{I}$$
   $$\mathbf{v} \leftarrow (\mathbf{r} - \langle \mathbf{r}, \mathbf{n} \rangle \mathbf{n}) / \|\mathbf{r} - \langle \mathbf{r}, \mathbf{n} \rangle \mathbf{n}\|.$$
   b) Project atoms into the $\mathbf{n}$-$\mathbf{v}$-plane and select the next atom
   $$k \leftarrow \arg \min_{i, \in \{1,\ldots,n\}} \tan^{-1} \frac{\langle \boldsymbol{\psi}_i - \tilde{\mathbf{x}}_H, \mathbf{n} \rangle}{\langle \boldsymbol{\psi}_i - \tilde{\mathbf{x}}_H, \mathbf{v} \rangle}.$$
   c) Compute the new representation and update the biorthogonal system
   $$\{\mathcal{I}, \mathcal{A}, \tilde{\Psi}^{\perp}\} \leftarrow \text{AddAtom}(\mathbf{x}, \mathcal{I}, \mathcal{A}, \boldsymbol{\psi}_k, \tilde{\Psi}^{\perp}).$$
   d) Discard any extraneous atoms

while $\exists \alpha_i \leq 0, i \in \mathcal{I}$ **do**

$$\{\mathcal{I}, \mathcal{A}, \tilde{\Psi}^{\perp}\} \leftarrow \text{SubtractAtom}(\mathbf{x}, \mathcal{I}, \mathcal{A}, \boldsymbol{\psi}_j, \tilde{\Psi}^{\perp}).$$

   e) Update the hyperplane parameters

$$\tilde{\mathbf{x}} \leftarrow \sum_{i \in \mathcal{I}} \alpha_i \boldsymbol{\psi}_i$$
$$\mathbf{n} \leftarrow \frac{-\langle \boldsymbol{\psi}_k - \tilde{\mathbf{x}}_H, \mathbf{n} \rangle \mathbf{v} + \langle \boldsymbol{\psi}_k - \tilde{\mathbf{x}}_H, \mathbf{v} \rangle \mathbf{n}}{\| - \langle \boldsymbol{\psi}_k - \tilde{\mathbf{x}}_H, \mathbf{n} \rangle \mathbf{v} + \langle \boldsymbol{\psi}_k - \tilde{\mathbf{x}}_H, \mathbf{v} \rangle \mathbf{n} \|}$$
$$\mathbf{r} \leftarrow \mathbf{x} - \tilde{\mathbf{x}}.$$

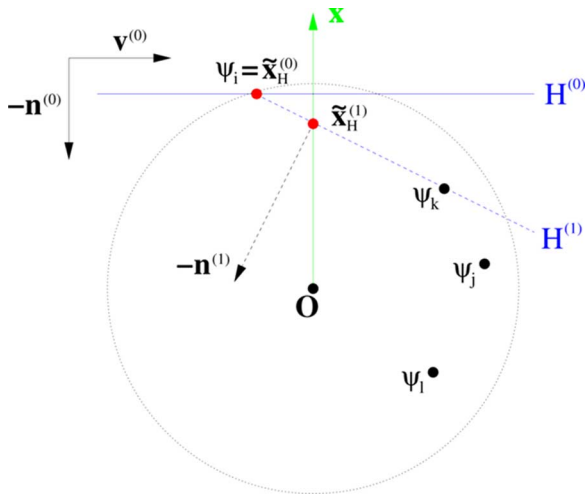


Fig. 2. Schematic of the first iteration of GBP. The intial hyperplane $H^{(0)}$ has normal $\mathbf{n}^{(0)}$ in the direction of the signal $\mathbf{x}$ and contains $\boldsymbol{\psi}_i$. The atoms are projected from $\mathbb{R}^d$ to the $\mathbf{n}^{(0)}$-$\mathbf{v}^{(0)}$ plane (shown) and sorted by θ. The second atom selected is $\boldsymbol{\psi}_k$, corresponding to a rotation of $H^{(0)}$ around $\tilde{\mathbf{x}}_H^{(0)}$ to $H^{(1)}$. Note that $\mathbf{v}^{(1)}$ is orthogonal to the $\mathbf{n}^{(0)}$-$\mathbf{v}^{(0)}$ plane (and, therefore, is not shown).

Fig. 3 provides a visualization of GBP in action in three dimensions.

3) Computational Details: At each iteration we compute $\tilde{\mathbf{x}}^{(t)}$ as the projection of $\mathbf{x}$ onto the convex cone of $\Psi^{(t)}$.

One approach to computing this projection is to maintain an orthogonal basis for the span of $\Psi^{(t)}$, updating it as atoms are added to $\Psi^{(t)}$, as in OMP [80]; this is impractical in our case as most iterative orthogonalization procedures are order-dependent and hyperplane rotation may cause us to discard arbitrary atoms from $\Psi^{(t)}$.

Instead, we maintain a biorthogonal system consisting of $\Psi^{(t)}$ and $\tilde{\Psi}^{\perp(t)}$, the set of vectors biorthogonal to $\Psi^{(t)}$. Each element $\tilde{\boldsymbol{\psi}}_i^{\perp(t)}$ of $\tilde{\Psi}^{\perp(t)}$ satisfies the following two equations:

$$\left\langle \boldsymbol{\psi}_i, \tilde{\boldsymbol{\psi}}_i^{\perp(t)} \right\rangle = 1 \tag{9}$$
$$\left\langle \boldsymbol{\psi}_i, \tilde{\boldsymbol{\psi}}_j^{\perp(t)} \right\rangle = 0, \text{if } i \neq j. \tag{10}$$

The biorthogonal vector $\tilde{\boldsymbol{\psi}}_i^{\perp(t)}$ can be understood as the component of $\boldsymbol{\psi}_i^{(t)}$ that is orthogonal to all of the other vectors in $\Psi^{(t)}$, appropriately scaled. That is, if we express an atom $\boldsymbol{\psi}_i \in \Psi^{(t)}$ as

$$\boldsymbol{\psi}_i = \boldsymbol{\psi}_i^{\|(t)} + \boldsymbol{\psi}_i^{\perp(t)} \tag{11}$$

where $\boldsymbol{\psi}_i^{\|(t)}$ is the component of $\boldsymbol{\psi}_i$ lying in the span of $\Psi^{(t)} - \{\boldsymbol{\psi}_i\}$

$$\boldsymbol{\psi}_i^{\|(t)} = \sum_{j \in \mathcal{I}^{(t)}, j \neq i} \beta_{ij}^{(t)} \boldsymbol{\psi}_j \tag{12}$$

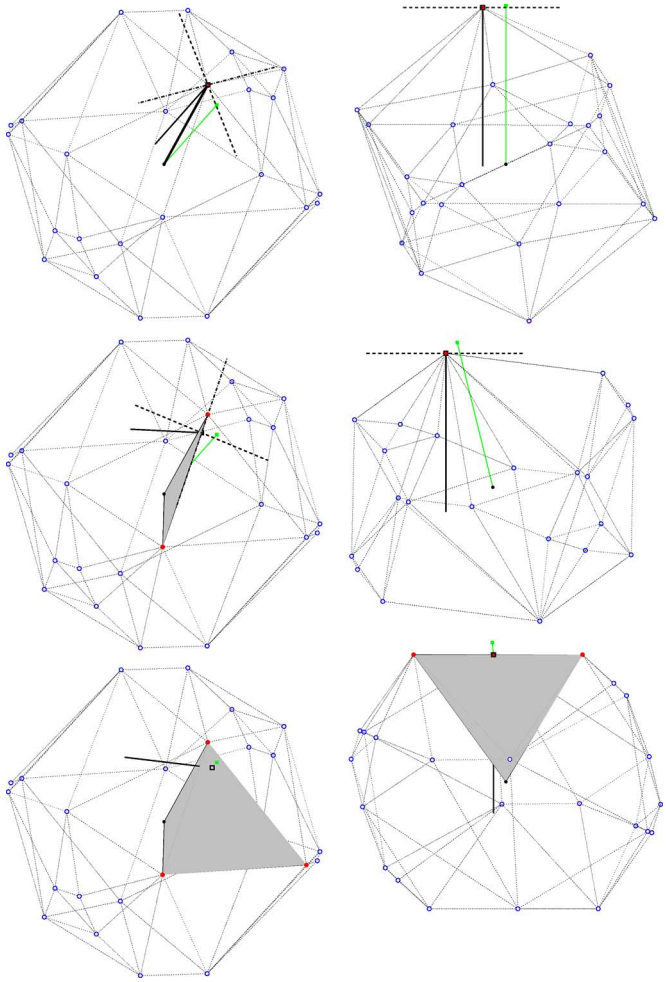


Fig. 3. GBP in action on a 3-D problem. Each row depicts one iteration, the left column from a fixed viewpoint, the right column projected to the $\mathbf{n}^{(t)}$-$\mathbf{v}^{(t)}$ plane. The signal vector is green, the unselected atoms blue circles, the selected atoms red discs, the convex cone of $\Psi^{(t)}$ is gray, the normal is the solid black line, and two vectors in $H^{(t)}$ are the dashed lines.

and $\boldsymbol{\psi}_i^{\perp(t)}$ is orthogonal to the span of $\Psi^{(t)} - \{\boldsymbol{\psi}_i\}$, then the biorthogonal vector to $\boldsymbol{\psi}_i^{(t)}$ is given by

$$\tilde{\boldsymbol{\psi}}_i^{\perp(t)} = \boldsymbol{\psi}_i^{\perp(t)} \Big/ \|\boldsymbol{\psi}_i^{\perp(t)}\|^2. \tag{13}$$

Given the biorthogonal system, we can compute the current approximation to $\mathbf{x}$ as

$$\tilde{\mathbf{x}}^{(t)} = \sum_{i \in \mathcal{I}^{(t)}} \alpha_i^{(t)} \boldsymbol{\psi}_i \text{ where } \alpha_i^{(t)} = \left\langle \mathbf{x}, \tilde{\boldsymbol{\psi}}_i^{\perp(t)} \right\rangle. \tag{14}$$

The sign of the coefficients indicates whether or not an approximation lies in the convex cone of the atoms: if $\alpha_i < 0$ for some i then the approximation does not lie in the convex cone; the corresponding atom $\boldsymbol{\psi}_i$ is deleted from the representation and the biorthogonal system is updated.

The biorthogonal system and $\tilde{\mathbf{x}}^{(t)}$ can be updated as atoms are added to and subtracted from $\Psi^{(t)}$. Such adaptive biorthogonalization methods have recently been applied to MP [85], [4] and are standard in linear programming (see [101, Ch. 8]). We present pseudocode for adding an atom in Algorithm 2. and pseudocode for substracting an atom in Algorithm 3.

Algorithm 2 AddAtom

Input
- The signal $\mathbf{x}$
- The dictionary $\mathcal{D}$
- The current representation $\mathcal{I}, \mathcal{A}$
- The atom to add k, $\boldsymbol{\psi}_k$
- The current biorthogonal vectors $\tilde{\Psi}^{\perp}$

Output
- The updated representation $\mathcal{I}, \mathcal{A}$
- The updated biorthogonal vectors $\tilde{\Psi}^{\perp}$

Procedure

1) Compute the new biorthogonal vector

$$\begin{aligned} &\forall i \in \mathcal{I}, \beta_i \leftarrow \left\langle \tilde{\boldsymbol{\psi}}_i^{\perp}, \boldsymbol{\psi}_k \right\rangle \\ &\boldsymbol{\psi}_k^{\perp} \leftarrow \boldsymbol{\psi}_k - \sum_{i \in \mathcal{I}} \beta_i \boldsymbol{\psi}_i \\ &\tilde{\boldsymbol{\psi}}_k^{\perp} \leftarrow \boldsymbol{\psi}_k^{\perp} / \|\boldsymbol{\psi}_k^{\perp}\|_2^2 \end{aligned}$$

2) Update the biorthogonal system

$$\begin{aligned} &\forall i \in \mathcal{I}, \tilde{\boldsymbol{\psi}}_i^{\perp} \leftarrow \tilde{\boldsymbol{\psi}}_i^{\perp} - \beta_i \tilde{\boldsymbol{\psi}}_k^{\perp} \\ &\tilde{\Psi}^{\perp} \leftarrow \tilde{\Psi}^{\perp} \cup \{\tilde{\boldsymbol{\psi}}_k^{\perp}\} \end{aligned}$$

3) Update the representation

$$\begin{aligned} &\alpha_k \leftarrow \langle \mathbf{x}, \tilde{\boldsymbol{\psi}}_k^{\perp} \rangle \\ &\forall i \in \mathcal{I}, \alpha_i \leftarrow \alpha_i - \beta_i \alpha_k \\ &\mathcal{I} \leftarrow \mathcal{I} \cup \{k\} \\ &\mathcal{A} \leftarrow \mathcal{A} \cup \{\alpha_k\}. \end{aligned}$$

Algorithm 3 SubtractAtom

Input
- The signal $\mathbf{x}$
- The current representation $\mathcal{I}, \mathcal{A}$
- The index of the atom to subtract k
- The current biorthogonal vectors $\tilde{\Psi}^{\perp}$

Output
- The updated representation $\mathcal{I}, \mathcal{A}$,
- The updated biorthogonal vectors $\tilde{\Psi}^{\perp}$

Procedure

1) Delete the atom from the representation

$$\begin{aligned} &\mathcal{I} \leftarrow \mathcal{I} - \{k\} \\ &\mathcal{A} \leftarrow \mathcal{A} - \{\alpha_k\} \end{aligned}$$

2) Update the biorthogonal system

$$\tilde{\Psi}^{\perp} \leftarrow \tilde{\Psi}^{\perp} - \left\{\tilde{\boldsymbol{\psi}}_k^{\perp}\right\}$$
$$\forall i \in \mathcal{I}, \gamma_i \leftarrow \left\langle \tilde{\boldsymbol{\psi}}_k^{\perp}, \tilde{\boldsymbol{\psi}}_i^{\perp} \right\rangle / \left\| \tilde{\boldsymbol{\psi}}_k^{\perp} \right\|_2^2$$
$$\forall i \in \mathcal{I}, \tilde{\boldsymbol{\psi}}_i^{\perp} \leftarrow \tilde{\boldsymbol{\psi}}_i^{\perp} - \gamma_i \tilde{\boldsymbol{\psi}}_k^{\perp}$$

3) Update the representation

$$\forall i \in \mathcal{I}, \alpha_i \leftarrow \alpha_i - \alpha_k \gamma_i.$$

B. Analysis

By construction, GBP computes the minimum ℓ^1-norm representation of a given signal. To prove this we show that GBP converges to an exact representation in a finite number of steps and that the representation corresponds to a facet of the convex hull of the dictionary.

First, we prove that GBP converges to an exact representation. At each iteration of GBP there is a decrease in approximation error, as stated in the following theorem.

Theorem 1: Given a signal $\mathbf{x} \in \mathbb{R}^d$ and a dictionary $\mathcal{D} = \{\boldsymbol{\psi}_i\}_{i=1}^n$, where $n \geq 2d$, $\forall \boldsymbol{\psi}_i \in \mathcal{D}$, $\boldsymbol{\psi}_i \in \mathbb{R}^d$ and $\|\boldsymbol{\psi}_i\|_2 = 1$, if $\boldsymbol{\psi}_i \in \mathcal{D}$, then $-\boldsymbol{\psi}_i \in \mathcal{D}$, and the atoms are in general position, if GBP is run with $\mathcal{D}$ and $\mathbf{x}$ as input and if $\tilde{\mathbf{x}}^{(t)} \neq 0$, then at iteration $t+1$ of GBP, $0 \leq \|\mathbf{x} - \hat{\mathbf{x}}^{(t+1)}\|_2 < \|\mathbf{x} - \tilde{\mathbf{x}}^{(t)}\|_2$.

Proof: At iteration t, let S be the hypersphere centered at $\mathbf{x}$ with radius $\|\mathbf{x} - \tilde{\mathbf{x}}^{(t)}\|_2$, let $\boldsymbol{\psi}_k$ be the next atom selected by GBP, and let T be the tangent plane to S at $\tilde{\mathbf{x}}^{(t)}$. T contains the origin (if it did not, then some scaling of $\tilde{\mathbf{x}}^{(t)}$ would be a better approximation to $\mathbf{x}$), and thus bisects the unit sphere. Because the atoms are in general position, $n \geq 2d$, and $\boldsymbol{\psi}_i \in \mathcal{D}$ if $-\boldsymbol{\psi}_i \in \mathcal{D}$, if $|\Psi^{(t)}| < d$, then there will be at least one atom in the same half-space of T as $\mathbf{x}$. (Note that if $|\Psi^{(t)}| = d$, we are done, as we would also have $\tilde{\mathbf{x}} = \mathbf{x}$.)

$\boldsymbol{\psi}_k$ lies in the same half-space of T as $\mathbf{x}$: by construction, there is no atom $\boldsymbol{\psi}_0$, such that $\langle \boldsymbol{\psi}_0 - \tilde{x}_H^{(t)}, \mathbf{n}^{(t)} \rangle > 0$, by general position, there is no atom $\boldsymbol{\psi}_0$, such that $\langle \boldsymbol{\psi}_0 - \tilde{x}_H^{(t)}, \mathbf{n}^{(t)} \rangle = 0$ and $\langle \boldsymbol{\psi}_0 - \tilde{x}_H^{(t)}, \mathbf{v}^{(t)} \rangle > 0$, and, by the ordering of atoms by step 2(b) of GBP, GBP selects an atom in the same half-space of T as $\mathbf{x}$, if one exists. $\boldsymbol{\psi}_k$ lies in the same half-space as $\mathbf{x}$, we can find a point $\hat{\mathbf{x}} + \epsilon(\boldsymbol{\psi}_k - \hat{\mathbf{x}}^{(t)})$ that is interior to S and, therefore, closer to $\mathbf{x}$ than $\tilde{\mathbf{x}}^{(t)}$. Therefore, $\|\mathbf{x} - \hat{\mathbf{x}}^{(t+1)}\| < \|\mathbf{x} - \tilde{\mathbf{x}}^{(t)}\|$. ■

Theorem 1 also implies that GBP does not cycle. GBP may select the same atom more than once, that is, GBP may select an atom, discard it, and select it again (this behavior depends on the shape of the facets of $\mathbf{conv}(\mathcal{D})$), but GBP will never revisit the same state. Because there are a finite number of states and GBP improves at each iteration, GBP converges. By the same arguments as Theorem 1, at convergence the final supporting hyperplane contains a facet of $\mathbf{conv}(\mathcal{D})$ and thus GBP computes the minimum ℓ^1-norm representation.

The duality of GBP to the gravitational method [74] of linear programming, implies that the computational complexity of GBP is exponential in the worst case [73]. Current results on the simplex algorithm suggest that GBP is likely to be polynomial in the average [12] and smoothed [93] cases.

C. Implementation Issues

We briefly describe two obstacles that any implementation of GBP may encounter, degeneracy and numerical instability, and our approach to handling them.

1) Degeneracy: Degeneracy occurs when the atoms of the dictionary are not in general position; the atoms are in general position if every k-face of $\mathbf{conv}(\mathcal{D})$ contains exactly $k+1$ atoms [104]. Degeneracy can occur if the dictionary is specially designed, for example, if the atoms are defined to be the vertices of a hypercube inscribed in the unit hypersphere. If GBP encounters degeneracy, the updates described in Section IV-A3 will fail, resulting in an error. Although GBP does not currently include a mechanism to detect and handle degeneracy, incorporating such a feature is possible. A simple solution is to perturb the atoms of the dictionary sufficiently to place them in general position (see Section V-A).

2) Numerical Instability: Numerical instability can occur in the biorthogonalization stage of GBP. Let $\boldsymbol{\Psi}$ be a matrix corresponding to $\Psi^{(t)}$ for some t, where each row of $\boldsymbol{\Psi}$ is an atom in $\Psi^{(t)}$, and let $\tilde{\boldsymbol{\Psi}}^{\perp}$ denote the corresponding matrix of biorthogonal vectors. If at any iteration the matrix $\boldsymbol{\Psi}$ is ill-conditioned, the computation of the biorthogonal vectors we have described may be unstable (similar difficulties arise in Gram–Schmidt orthogonalization [86], [10]). One work around is to compute a full biorthogonalization at each iteration, or at least whenever instability is detected. However, a full biorthogonalization can be costly, as it is typically computed via the pseudoinverse [54]: since $\boldsymbol{\Psi}(\tilde{\boldsymbol{\Psi}}^{\perp})^T = \mathbf{I}$, where $\mathbf{I}$ denotes the identity matrix, we can compute $\tilde{\boldsymbol{\Psi}}^{\perp}$ as $(\boldsymbol{\Psi}^{+})^T$, where "+" denotes the pseudoinverse.

We instead opt to compute the biorthogonalization using an iterative pseudoinverse technique [6]. This technique takes an initial estimate of the pseudoinverse and iteratively updates it, converging to the true pseudoinverse. If the initial estimate is sufficiently close to the true pseudoinverse, then the iterative pseudoinverse computation is substantially faster than the standard pseudoinverse. This approach is well suited to GBP, as the adaptive biorthogonalization already provides such an estimate.

The iterative pseudoinverse algorithm proceeds as follows. Given a matrix $\boldsymbol{\Psi}$ and an initial estimate of the pseudoinverse $\boldsymbol{\Psi}^{+(0)}$, the updates to $\boldsymbol{\Psi}^{+}$ are computed by

$$\boldsymbol{\Psi}^{+(t+1)} \leftarrow \boldsymbol{\Psi}^{+(t)} \left(2\mathbf{I} - \boldsymbol{\Psi}\boldsymbol{\Psi}^{+(t)}\right).$$

(Note that here t denotes the iteration of the pseudoinverse algorithm, not the iteration of GBP.) For a detailed analysis of this algorithm, see [92]. While a classic technique, this algorithm is the subject of ongoing research [79], [82].

Our implementation of GBP tests if $\boldsymbol{\Psi}(\tilde{\boldsymbol{\Psi}}^{\perp})^T = \mathbf{I}$ within a specified level of tolerance after each adaptive biorthogonalization. If the test fails, the iterative pseudoinverse algorithm is applied.

V. Results

We examine the performance of GBP. We compared the running time of GBP to that of standard linear programming algorithms on three data sets, random data, speech data, and seismic data, described as follows. We also provide an example of GBP's performance on a single signal and contrast it with that of MP.

In each experiment, we measured the running times of GBP and standard linear programming algorithms on the signal representation problems described below. The algorithms we compared were GBP, two variants of the simplex method, and an interior point method.

The implementation of GBP used was our own, written entirely in MATLAB. The linear programming solvers used were those included in the MATLAB optimization toolbox 3.0 [1], and a freely available MATLAB implementation [72] of the revised simplex method [24]. The optimization toolbox version of the simplex method is the classical simplex method [25], with the initial basis determined as in [8]. The optimization toolbox version of the interior point method is essentially LIPSOL [103], a freely available interior point solver that implements Mehrotra's predictor-corrector method [67], [70].

For each problem, all algorithms were run and timed. All algorithms were run under MATLAB 7 on a 1.5-GHz Pentium M processor running Windows XP, with 1.25-GB memory. On all problems all algorithms returned identical representations (up to the specified error tolerance).

A. Running Times: Random Data

The random data set consisted of 3000 randomly generated signal representation problems, varying both the dimension of the signal space and the overcompleteness of the dictionary. Each problem consisted of a randomly generated signal and a randomly generated dictionary. The dimension d of the problems varied over the set $\{8, 16, 32, 64, 128, 256\}$. The overcompleteness k of the dictionaries varied over the set $\{2, 4, 8, 16, 32\}$. In each problem, the signal $\mathbf{x}$ was randomly generated to be uniformly distributed on the unit hypersphere in $\mathbb{R}^d$. The dictionary for each problem had $2kd$ atoms; the first kd of these atoms were generated in the same fashion as the signal, the second kd atoms were the negatives of the first kd atoms. Additionally, the dictionary of each problem was perturbed. To each atom was added Gaussian noise with variance 0.000001, after which the atom was normalized to lie on the unit hypersphere; this perurbation ensures that the linear programming algorithms can compute the requisite matrix inverses; for structured dictionaries this perturbation also ensures that the atoms are in general position. For each d-k pair, 100 problems were generated.

Fig. 4 shows running times of the three algorithms as a function of overcompleteness for each dimension; the curve plotted shows the mean running time of each algorithm over the 100 problems of the specified dimension and overcompleteness, with error bars showing the corresponding minimum and maximum running times. (We do not show the results of the revised simplex method here, as it was outperformed by the MATLAB's simplex algorithm.)

B. Running Times: Speech Data

The speech data set consisted of 100 signal representation problems. Each problem consisted of a signal randomly drawn from the TIMIT database [50] and an overcomplete multiscale Gabor dictionary.

Each signal comprised 256 samples $(d = 256)$ and was randomly selected from the "train" subset of the TIMIT database. The signals were mean centered and normalized. Some samples are shown in Fig. 5.

The dictionary used was a $9\times$ overcomplete multiscale Gabor dictionary (4608 atoms). The dictionary consisted of several fixed scale critically sampled cosine Gabor bases. Each atom was defined by the parameters t_0 and f as $G(t; t_0, f) = (1/(2\pi\sigma)) \exp^{-(t-t_0)^2/\sigma^2} \cos(2\pi f(t - t_0))$, where $t_0 \in \{0 : \Delta t : 1\}$ and $f \in \{0 : \Delta f : d/2\}$, with $\Delta t = 2^j/d$, $\sigma = \sqrt{\pi/2}/\Delta t$, and $\Delta f = \sigma/\sqrt{2\pi}$; the scale parameter j varied over $\{0, 1, \ldots, 8\}$. See [42] and [49] for details and other sampling schemes. Once the atoms were defined, they were perturbed as in the random data case. Some samples from the final dictionary are shown in Fig. 6.

We show the running times of GBP, LIPSOL, and the revised simplex method on the sound data set in Table I. (The revised simplex method outperformed MATLAB's simplex method.) We show the mean, minimum, and maximum runninng times for each algorithm on the 100 signals.

C. Running Times: Seismic Data

The seismic data consists of 100 signal representation problem. Each problem consists of a 256 sample signal of seismic recordings from the North Sea, $4\times$ downsampled from the original data [91]; some samples are shown in Fig. 7. The dictionary used was the same as used in the previous speech experiment. We show the running times of GBP, LIPSOL, and the revised simplex method on the seismic data set in Table II.

D. Example: Speech Signal

Fig. 8 provides an example comparing GBP to MP and OMP on a 1024-D signal [see Fig. 8(a)], selected from the TIMIT speech database [50], using a multiscale Gabor dictionary $(n = 22528)$, similar to the one used for the sound data. (Note that the other BP methods were unable to compute representations on problems of this size in our environment.) Examining the approximation error of each algorithm as a function of iteration [see Fig. 8(b)], we observe that while the approximation error of GBP decreases somewhat more slowly than that of MP (note also that each iteration of GBP is more costly), the error of GBP does appear to decrease approximately exponentially. Also note that the representation computed by GBP is sparser than that of MP, though less sparse than that of OMP, as indicated by the sorted-amplitudes-curves and the ℓ^1-norm of the representations. The sorted-amplitudes-curves [59], [63] [see Fig. 8(c)] are plots of the logarithm of the final coefficients, sorted in descending order; the rates of decrease indicate the relative sparsity of the representations. The ℓ^1-norm of the representation coefficients are 0.3274, 0.4569, and 0.4156 for GBP, MP, and OMP, respectively. (Note that the results for GBP would be the same as those for standard linear programming methods for BP.) The feature of GBP to note here is its "greediness": the coefficients in the order of atom selection track the sorted-amplitudes-curve, that is, GBP tends to select significant atoms early on [see Fig. 8(d)]. This demonstrates that it is possible to compute BP signal representations and to be greedy at the same time.

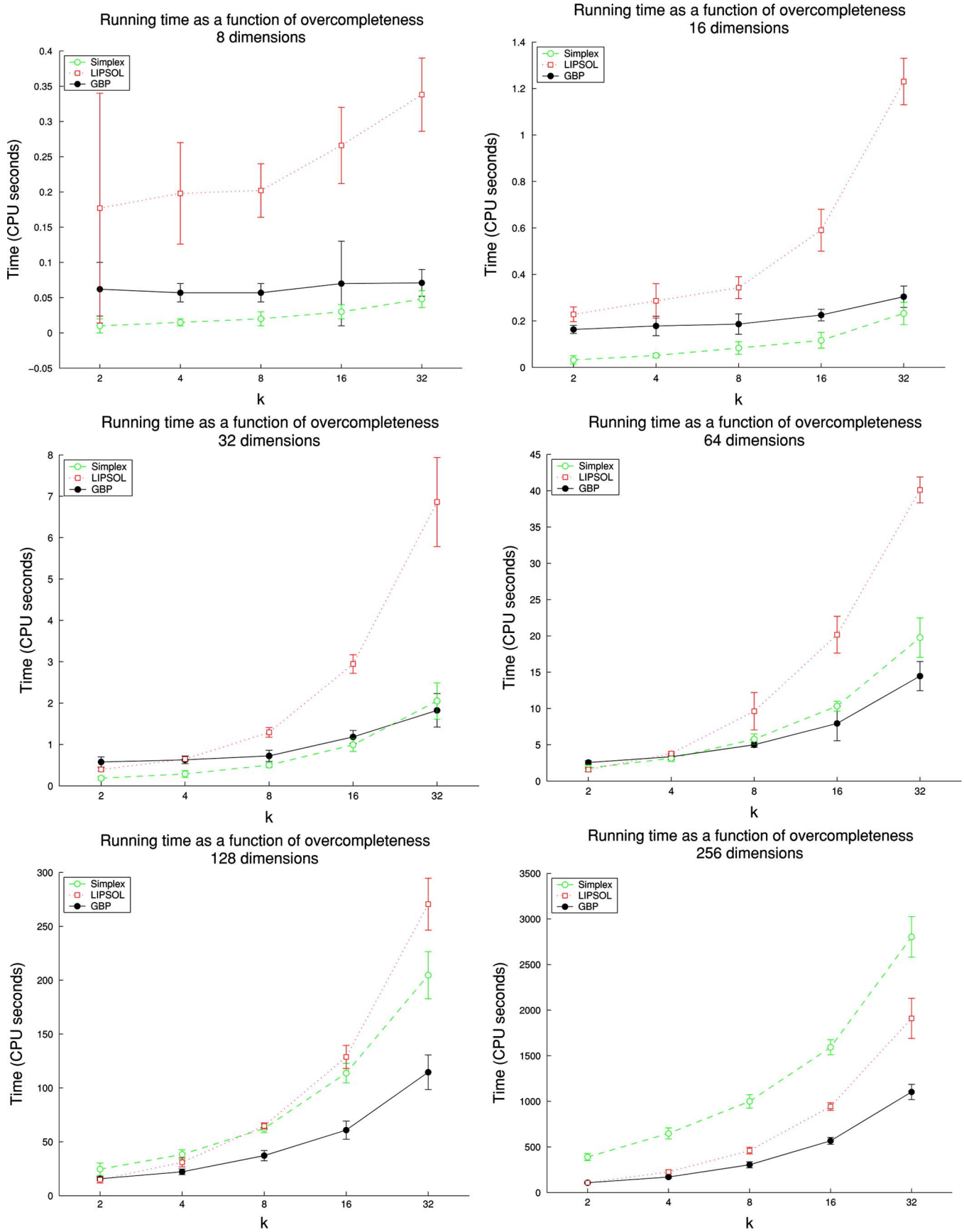


Fig. 4. Running times for GBP, LIPSOL, and the simplex method (MATLAB) on the random data set, plotted as a function of overcompleteness for each dimension. Note that GBP's performance improves relative to the other methods as the dimensionality of the problems increase.

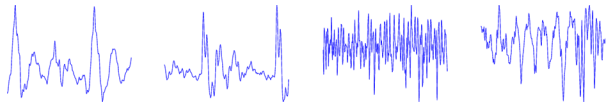


Fig. 5. Four signals drawn from the speech data set. Each signal consists of 256 samples.

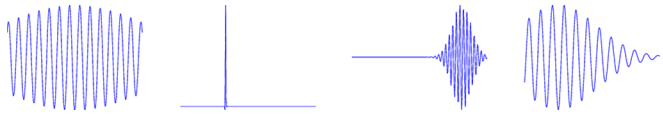


Fig. 6. Four atoms drawn from the multiscale Gabor cosine dictionary.

TABLE I
RUNNING TIMES OF GBP, LIPSOL, AND THE REVISED SIMPLEX METHOD ON THE SOUND DATA SET, IN CPU SECONDS

Algorithm	Min	Mean	Max
GBP	40.41	48.72	56.35
LIPSOL	58.62	75.97	155.56
Revised Simplex	441.66	1297.65	2700.51

Fig. 7. Four signals drawn from the seismic data set. Each signal consists of 256 samples.

TABLE II
RUNNING TIMES OF GBP, LIPSOL, AND THE REVISED SIMPLEX METHOD ON THE SEISMIC DATA SET, IN CPU SECONDS

Algorithm	Min	Mean	Max
GBP	42.29	48.83	55.05
LIPSOL	60.52	70.36	112.90
Revised Simplex	2233.45	2489.05	2831.59

VI. DISCUSSION

Our results show that GBP provides a fast alternative to standard linear programming methods for sparse signal representation problems, particularly when the dimension of the signal space is high and the dictionary is very overcomplete. While there are a variety of factors which may contribute to the results, there are several algorithmic reasons why we expect GBP to perform relatively well.

The efficient solution of linear programming problems depends in a complicated way on the problem, the method of solution and its implementation, and the available resources (see Bixby [9]). Thus, the relative success of GBP compared to the linear programming methods implemented in the MATLAB optimization toolbox is partially a function of the specific methods used and their implementation. There are many available linear programming packages [45], some specific to sparse representation include Atomizer,[1] ℓ_1-MAGIC,[2] and SparseLab.[3] An exhaustive comparison of GBP against all of these methods is out of the scope of this paper. However, while the linear programming methods against which GBP was compared may not represent the current state-of-the-art, it is worth noting that GBP itself has the potential for significant speed increases through more efficient implementation.

Algorithmically, GBP has several advantages over standard linear program solvers. First, most linear program solvers assume, for historical reasons, that the constraint matrix is sparse, and they, therefore, rely on techniques that exploit this sparsity, whether or not sparsity is actually present [19]. The signal representation problems considered here are not particularly sparse in this sense: the random dictionary matrices used in the random data set are certainly not sparse, while the Gabor dictionary matrices used with the sound and seismic data sets are somewhat sparse, however, the matrices are not sufficiently structured for certain fast algorithms to be applicable [17], [18]. GBP does not exploit this sparsity, and therefore, does not suffer when it is not present. Second, GBP is efficient in the search for the next atom to select, because this search is based on a geometric criterion that involves two projections per possible atom. Simplex methods can be inefficient at this task as the search can involve evaluating more than two, even d, projections per possible atom (see [98] and [101]). Third, the updates in GBP are seldom of a full basis, further reducing computation. Finally, the complexity of the simplex method depends on the closeness of the initial solution to the optimal solution, which in turn depends on the phase I algorithm by which the initial basis is selected. GBP does not depend on an intial solution; in fact, GBP can be interpreted as a combined phase I/ phase II linear programming algorithm.

One area which we have not explored that merits further investigation is the dependence of the performance of GBP (and other sparse representation algorithms) on the structure of the dictionary. For example, a dictionary optimized for use with MP [27] or OMP [41] may well have very different properties from one optimized for BP. The design of dictionaries has only recently received attention in the signal processing community [2], [27], [41] (for work in neural computation, see [66] and [77]); our work suggests that the geometric properties of dictionaries play a crucial role in both the efficiency of representation algorithms and the quality of the resulting representations. Indeed, geometric considerations have already led to a better theoretical understanding of sparse signal representation [35], [36].

As noted, part of the motivation for the development of BP is the observation that MP and OMP can fail to find sparse, in the ℓ^0-norm sense, signal representations [18], with much theoretical work showing under exactly what conditions BP finds sparse representations, i.e., when the minimal ℓ^1-norm solution is equivalent to the minimal ℓ^0-norm solution [33], [34], [48]. These findings have made BP useful for areas beyond signal representation, including compressed sensing [32] and error correcting codes [13], thus GBP may prove useful in these domains.

VII. CONCLUSION

We have described GBP, a new algorithm for BP, and demonstrated that it is faster than standard linear programming

[1][Online]: Available: http://www-stat.stanford.edu/atomizer

[2][Online]. Available: http://www.acm.caltech.edu/l1magic

[3][Online]. Available: http://sparselab.stanford.edu

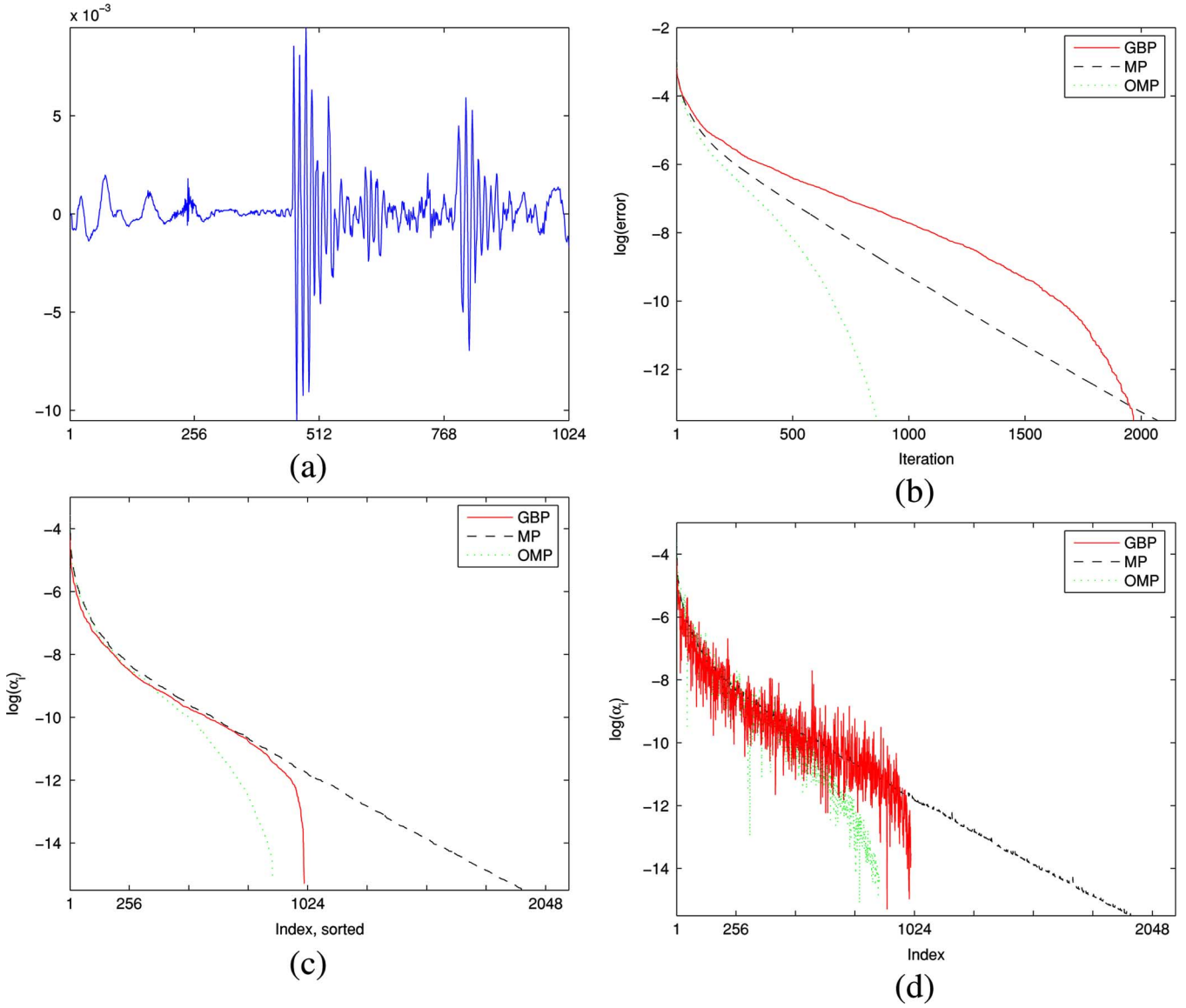


Fig. 8. (a) Example comparing GBP to MP on a speech signal. (b) The log of the error (ℓ^2-norm of the residual) as a function of iteration. (c) The sorted-amplitudes-curves; observe that GBP produces a sparser representation than MP. (d) The (final) coefficient values, in order of atom selection. (Note that the coefficient values change in GBP at each iteration.) See text for discussion.

methods on some problems, particularly in high-dimensional signal spaces using very overcomplete dictionaries. A MATLAB implementation of GBP is currently available online at: http://www.cs.yale.edu/~huggins/gbp.html

Computational geometry has traditionally been the preserve of computer science, particularly computer graphics and theoretical computer science; its use here in the development of GBP highlights the relevance of computational geometry to signal processing. GBP also illustrates the interplay between signal processing and linear programming. That an efficient linear programming algorithm falls naturally out of sparse signal representation is surprising, and suggests that researchers in signal processing should not view linear programming, or optimization in general, as a black box: on one hand signal processing naturally defines a set of problems that can serve to drive research in linear programming, on the other hand, given the historical parallels, optimization research deserves deeper examination by the signal processing community.

ACKNOWLEDGMENT

The authors would like to thank M. Maggioni for many helpful discussions; O. Ben-Shahar, P. Dimitrov, and G. Li for their advice; K. Skretting for providing the seismic data; and D. Spielman for his insight into linear programming. They would also like to thank the three anonymous reviewers for their suggestions which have considerably improved this manuscript.

REFERENCES

[1] The MathWorks, Natick, MA, "Optimization toolbox user's guide," 2003.

[2] M. Aharon, M. Elad, A. M. Bruckstein, and Y. Katz, "The K-SVD: An algorithm for designing of overcomplete dictionaries for sparse representation," *IEEE Trans. Signal Process.*, to be published.

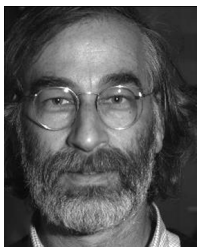
[3] O. K. Al-Shaykh, E. Miloslavsky, T. Nomura, R. Neff, and A. Zakhor, "Video compression using matching pursuits," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 9, no. 1, pp. 123–143, Feb. 1999.

[4] M. Andrle, L. Rebollo-Neira, and E. Sagianos, "Backward-optimized orthogonal matching pursuit approach," *IEEE Signal Process. Lett.*, vol. 11, no. 9, pp. 705–708, Sep. 2004.
[5] J. B. Bednar, R. Yarlagadda, and T. Watt, "L_1 deconvolution and its application to seismic signal processing," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 34, no. 6, pp. 1655–1658, Dec. 1986.
[6] A. Ben-Israel, "An iterative method for computing the generalized inverse of an arbitrary matrix," *Math. Comput.*, vol. 19, pp. 452–455, 1965.
[7] F. Bergeaud and S. Mallat, "Matching pursuit of images," in *Proc. Int. Conf. Image Process.*, 1995, p. 53.
[8] R. E. Bixby, "Implementing the simplex method: The initial basis," *ORSA J. Comput.*, vol. 4, no. 3, pp. 267–284, 1992.
[9] R. E. Bixby, "Solving real-world linear programs: A decade and more of progress," *Oper. Res.*, vol. 50, no. 1, pp. 3–15, 2002.
[10] A. Björck, "Numerics of Gram-Schmidt orthogonalization," *Linear Algebra Appl.*, vol. 197–198, pp. 297–316, 1994.
[11] P. Bofill and M. Zibulevsky, "Underdetermined blind source separation using sparse representations," *Signal Process.*, vol. 81, pp. 2353–2362, 2001.
[12] K. H. Borgwardt, *The Simplex Method—A Probabilistic Analysis*. New York: Springer-Verlag, 1987.
[13] E. Candes, M. Rudelson, R. Vershynin, and T. Tao, "Error correction via linear programming," in *Proc. FOCS*, 2005, pp. 295–308.
[14] D. R. Chand and S. S. Kapur, "An algorithm for convex polytopes," *J. Association Comput. Mach.*, vol. 17, no. 1, pp. 78–86, 1970.
[15] S. Y. Chang and K. G. Murty, "The steepest descent gravitational method for linear programming," *Discrete Appl. Math.*, vol. 25, pp. 211–239, 1989.
[16] S. Chen and D. Donoho, "Basis pursuit," in *Proc. 28th Asilomar Conf. Signals, Syst. Comput.*, 1994, pp. 41–44.
[17] S. S. Chen, "Basis Pursuit," Ph.D. dissertation, Dept. Statistics, Stanford Univ, Stanford, CA, 1995.
[18] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM Rev.*, vol. 43, no. 1, pp. 129–159, 2001.
[19] V. Chvátal, *Linear Programming*. San Francisco, CA: Freeman, 1983.
[20] T. H. Cormen, C. E. Leiserson, and R. L. Rivest, *Introduction to Algorithms*. New York: McGraw-Hill, 1990.
[21] S. F. Cotter, J. Adler, B. D. Rao, and K. Kreutz-Delgado, "Forward sequential algorithms for best basis selection," *Inst. Elect. Eng. Proc.-Vis. Signal Process.*, vol. 146, no. 5, pp. 235–244, Oct. 1999.
[22] S. F. Cotter, K. Kreutz-Delgado, and B. D. Rao, "Backward sequential elimination for sparse vector subset selection," *Signal Process.*, vol. 81, pp. 1849–1864, 2001.
[23] G. B. Dantzig, "Converting a converging algorithm into a polynomial bounded algorithm," Syst. Opt. Lab., Stanford Univ., Stanford, CA, Tech. Rep. SOL 91-5, 1991.
[24] G. B. Dantzig and W. Orchard-Hays, "The product form for the inverse in the simplex method," *Math. Tables Other Aids Comput.*, vol. 8, pp. 64–67, 1954.
[25] G. B. Dantzig, A. Orden, and P. Wolfe, "The generalized simplex method for minimizing a linear form under linear inequality restraints," *Pacific J. Math.*, vol. 5, pp. 183–195, 1955.
[26] I. Daubechies, "Time-frequency localization operators: A geometric phase space approach," *IEEE Trans. Inf. Theory*, vol. 34, no. 4, pp. 605–612, Jul. 1988.
[27] G. Davis, "Adaptive nonlinear approximations," Ph.D. dissertation, Dept. Math., New York Univ., New York, 1994.
[28] G. Davis, S. Mallat, and M. Avellaneda, "Adaptive greedy approximation," *Constructive Approximation*, vol. 13, pp. 57–98, 1997.
[29] G. Davis, S. Mallat, and Z. Zhang, "Adaptive time-frequency decompositions with matching pursuit," *Opt. Eng.*, vol. 33, no. 7, pp. 2183–2191, 1994.
[30] R. A. DeVore and V. N. Temlyakov, "Some remarks on greedy algorithms," *Adv. Comput. Math.*, vol. 5, pp. 173–187, 1996.
[31] D. L. Donoho, "Sparse components of images and optimal atomic decompositions," *Constructive Approximation*, vol. 17, no. 3, pp. 353–382, 2001.
[32] D. L. Donoho, "Compressed sensing," Dept. Statistics, Stanford Univ., 2004.
[33] D. L. Donoho and M. Elad, "Optimally sparse representation in general (nonorthogonal) dictionaries via ℓ^1 minimization," in *Proc. PNAS*, 2003, pp. 2197–2202.
[34] D. L. Donoho and X. Huo, "Uncertainty principles and ideal atomic decomposition," *IEEE Trans. Inf. Theory*, vol. 47, no. 7, pp. 2845–2862, Nov. 2001.
[35] D. L. Donoho and J. Tanner, "Neighborliness of randomly projected simplices in high dimensions," in *Proc. Nat. Academy Sci.*, 2005, pp. 9452–9457.
[36] D. L. Donoho and J. Tanner, "Sparse nonnegative solution of underdetermined linear equations by linear programming," in *Proc. Nat. Academy Sci.*, 2005, pp. 9446–9451.
[37] B. Drachman, "Two methods to deconvolve: L_1-method using simplex algorithm and L_2-method using least squares and a parameter," *IEEE Trans. Ant. Propag.*, vol. 32, no. 3, pp. 219–225, Mar. 1984.
[38] M. A. Efroymson, "Multiple regression analysis," in *Mathematical Methods for Digital Computers*. New York: Wiley, 1960, pp. 191–203.
[39] M. Elad, "Why simple shrinkage is still relevant for redundant representations?," *IEEE Trans. Inf. Theory*, vol. 52, no. 12, pp. 5559–5569, Dec. 2006.
[40] L. O. Endelt and A. la Cour-Harbo, "Comparison of methods for sparse representation of music signals," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2005, pp. 13–16.
[41] K. Engan, S. O. Aase, and J. H. Husøy, "Multi-frame compression: Theory and deisgn," *Signal Process.*, vol. 80, no. 10, pp. 2121–2140, 2001.
[42] H. G. Feichtinger and T. Strohmer, Eds., *Gabor Analysis and Algorithms, Theory and Applications*. Cambridge, MA: Birkhäuser, 1998.
[43] H. G. Feichtinger, A. Türk, and T. Strohmer, "Hierarchical parallel matching pursuit," in *Proc. SPIE: Image Reconstruction Restoration*, 1994, pp. 222–232.
[44] S. E. Ferrando, E. J. Doolittle, A. J. Bernal, and L. J. Bernal, "Probabilistic matching pursuit with Gabor dictionaries," *Signal Process.*, vol. 80, pp. 2099–2120, 2000.
[45] R. Fourer, "Software survey: Linear programming," *OR/MS Today*, Jun. 2005.
[46] J. H. Friedman and W. Stuetzle, "Projection pursuit regression," *J. Amer. Stat. Assoc.*, vol. 76, no. 376, pp. 817–823, 1981.
[47] J. H. Friedman and J. W. Tukey, "A projection pursuit algorithm for exploratory data analysis," *IEEE Trans. Comput.*, vol. C-23, no. 9, pp. 881–890, Sep. 1974.
[48] J.-J. Fuchs, "On sparse representations in arbitrary redundant bases," *IEEE Trans. Inf. Theory*, vol. 50, no. 6, pp. 1341–1344, Jun. 2004.
[49] D. Gabor, "Theory of communication," *J. IEE*, vol. 93, pp. 429–457, 1946.
[50] J. S. Garofolo, L. F. Lamel, W. M. Fisher, J. G. Fiscus, D. S. Pallett, and N. L. Dahlgren, "DARPA TIMIT acoustic-phonetic continuous speech corpus CD-ROM," U.S. Dept. Commerce, Nat. Inst. Standards Technol., Gaithersburg, MD, Tech. Rep. NISTIR-4930, 1993.
[51] M. Gharavi-Alkhansari and T. S. Huang, "A fast orthogonal matching pursuit algorithm," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 1998, pp. 1389–1392.
[52] A. C. Gilbert and J. A. Tropp, "Applications of sparse approximations in communications," in *Proc. IEEE Int. Symp. Inf. Theory*, 2005, pp. 1000–1004.
[53] I. F. Gorodnitsky and B. D. Rao, "Sparse signal reconstruction from limited data using FOCUSS: A re-weighted minimum norm algorithm," *IEEE Trans. Signal Process.*, vol. 45, no. 3, pp. 600–616, Mar. 1997.
[54] T. N. E. Greville, "The pseudoinverse of a rectangular or singular matrix and its application to the solution of systems of linear equations," *SIAM Rev.*, vol. 1, no. 1, pp. 38–43, 1959.
[55] R. Gribonval and P. Vandergheynst, "On the exponential convergence of matching pursuits in quasi-incoherent dictionaries," *IEEE Trans. Inf. Theory*, vol. 52, no. 1, pp. 255–261, Jan. 2006.
[56] G. Harikumar, C. Couvreur, and Y. Bresler, "Fast optimal and suboptimal algorithms for sparse solutions to linear inverse problems," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 1998, pp. 1877–1881.
[57] A. J. Hoffman, "On greedy algorithms that succeed," *Surveys in Combinatorics*, pp. 97–112, 1985.
[58] P. J. Huber, "Projection pursuit," *Annals Stat.*, vol. 13, no. 2, pp. 435–475, 1985.
[59] X. Huo, "Sparse image representation via combined transforms," Ph.D. dissertation, Dept. Stat., Stanford Univ., Stanford, CA, 1999.
[60] S. Jaggi, W. C. Karl, S. Mallat, and A. S. Willsky, "High resolution pursuit for feature extraction," *Appl. Comput. Harmonic Anal.*, vol. 5, pp. 428–449, 1998.
[61] R. A. Jarvis, "On the identification of the convex hull of a finite set of points in the plane," *Inf. Process. Lett.*, vol. 2, pp. 18–21, 1973.

[62] J. Karvanen and A. Cichocki, "Measuring sparseness of noisy signals," in *Proc. 4th Int. Symp. Independent Component Anal. Blind Signal Separation*, 2003, pp. 125–130.
[63] K. Kreutz-Delgado and B. D. Rao, "Measures and algorithms for best basis selection," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 1998, pp. 1881–1884.
[64] S. A. Leichner, G. B. Dantzig, and J. W. Davis, "A strictly improving phase I algorithm using least-squares subproblems," Syst. Opt. Lab., Stanford Univ., Stanford, CA, Tech. Rep. SOL 92-1, 1992.
[65] M. S. Lewicki, "Efficient coding of natural sounds," *Nature Neurosci.*, vol. 5, no. 4, pp. 356–363, 2002.
[66] M. S. Lewicki and T. J. Sejnowski, "Learning overcomplete representations," *Neural Comput.*, vol. 12, pp. 337–365, 2000.
[67] I. J. Lustig, R. E. Marsten, and D. F. Shanno, "On implementing Mehrotra's predictor-corrector interior point method for linear programming," *SIAM J. Opt.*, vol. 2, no. 3, pp. 435–449, 1992.
[68] D. Malioutov, M. Cetin, and A. S. Willsky, "A sparse signal reconstruction perspective on source localization with sensor arrays," *IEEE Trans. Signal Process.*, vol. 53, no. 8, pp. 3010–3022, Aug. 2005.
[69] S. G. Mallat and Z. Zhang, "Matching pursuits with time-frequency dictionaries," *IEEE Trans. Signal Process.*, vol. 41, no. 12, pp. 3397–3415, Dec. 1993.
[70] S. Mehrotra, "On the implementation of a primal-dual interior point method," *SIAM J. Opt.*, vol. 2, no. 4, pp. 575–601, 1992.
[71] A. J. Miller, *Subset Selection in Regression*, 2nd ed. Boca Raton, FL: CRC Press, 2002.
[72] S. S. Morgan, "A comparison of simplex method algorithms," Master's thesis, Comput. Inf. Sci. Eng. Dept., Univ. Florida, Gainesville, 1997.
[73] T. L. Morin, N. Prabhu, and Z. Zhang, "Complexity of the gravitational method for linear programming," *J. Opt. Theory Appl.*, vol. 108, no. 3, pp. 633–658, 2001.
[74] K. G. Murty, "The gravitational method for linear programming," *Opsearch*, vol. 23, pp. 206–214, 1986.
[75] B. K. Natarajan, "Sparse approximate solutions to linear systems," *SIAM J. Comput.*, vol. 24, no. 2, pp. 227–234, 1995.
[76] M. S. O'Brien, A. N. Sinclair, and S. M. Kramer, "Recovery of a sparse spike time series by L_1 norm deconvolution," *IEEE Trans. Signal Process.*, vol. 42, no. 12, pp. 3353–3365, Dec. 1994.
[77] B. A. Olshausen and D. J. Field, "Sparse coding with an overcomplete basis set: A strategy employed by V1?," *Vision Res.*, vol. 37, no. 23, pp. 3311–3325, 1997.
[78] P.-Q. Pan, "A basis-deficiency-allowing variation of the simplex method for linear programming," *Comput. Math. With Appl.*, vol. 36, no. 3, pp. 33–53, 1998.
[79] V. Pan and R. Schreiber, "An improved {N}ewton iteration for the generalized inverse of a matrix, with applications," *SIAM J. Sci. Stat. Comput.*, vol. 12, no. 5, pp. 1109–1130, 1991.
[80] Y. C. Pati, R. Rezaiifar, and P. S. Krishnaprasad, "Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition," in *Proc. 27th Asilomar Conf. Signals, Syst. Comput.*, 1993, pp. 40–44.
[81] P. J. Phillips, "Matching pursuit filters applied to face identification," *IEEE Trans. Image Process.*, vol. 7, no. 8, pp. 1150–1164, Aug. 1998.
[82] W. H. Pierce, "A self-correcting matrix iteration for the Moore-Penrose generalized inverse," *Linear Algebra Its Appl.*, vol. 244, pp. 357–363, 1996.
[83] S. Qian and D. Chen, "Signal representation using adaptive normalized Gaussian functions," *Signal Process.*, vol. 36, pp. 1–11, 1994.
[84] B. D. Rao, "Signal processing with the sparseness constraint," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 1998, pp. 1861–1864.
[85] L. Rebollo-Neira and D. Lowe, "Optimized orthogonal matching pursuit approach," *IEEE Signal Process. Lett.*, vol. 9, no. 4, pp. 137–140, Apr. 2002.
[86] J. R. Rice, "Experiments on Gram-Schmidt orthogonalization," *Math. Comput.*, vol. 20, pp. 325–328, 1966.
[87] R. Seidel, "Constructing higher-dimensional convex hulls at logarithmic cost per face," in *Proc. 18th ACM Symp. Theory Comput.*, 1986, pp. 404–413.
[88] R. Seidel, "Convex hull computations," in *Handbook of Discrete and Computational Geometry*, 2nd ed. Boca Raton, FL: CRC Press, 2004.
[89] A. P. Sethi and G. L. Thompson, "The pivot and probe algorithm for solving a linear program," *Math. Program.*, vol. 29, pp. 219–233, 1984.
[90] E. P. Simoncelli, W. T. Freeman, E. H. Adelson, and D. J. Heeger, "Shiftable multi-scale transforms," *IEEE Trans. Inf. Theory*, vol. 38, no. 2, pp. 587–607, Mar. 1992.
[91] K. Skretting, "Pre-stack/post-stack seismic data from the north sea," (2000). [Online]. Available: http://www.ux.his.no/karlsk/sdata/
[92] T. Söderström and G. W. Stewart, "On the numerical properties of an iterative method for computing the Moore-Penrose generalized inverse," *SIAM J. Numer. Anal.*, vol. 11, no. 1, pp. 61–74, 1974.
[93] D. A. Spielman and S. Teng, "Smoothed analysis of algorithms: Why the simplex algorithm usually takes polynomial time," in *Proc. STOC*, 2001, pp. 296–305.
[94] J. L. Starck, M. Elad, and D. Donoho, "Redundant multiscale transforms and their application for morphological component separation," *Adv. Imag. Electron Phys.*, vol. 132, pp. 287–348, 2004.
[95] J. J. Stone, "The cross-section method," The RAND Corp., Santa Monica, CA, Tech. Rep. P-1490, 1958.
[96] G. Swart, "Finding the convex hull facet by facet," *J. Algorithms*, vol. 6, pp. 17–48, 1985.
[97] V. N. Temlyakov, "Greedy algorithms and m-term approximation with regard to redundant dictionaries," *J. Approximation Theory*, vol. 98, no. 1, pp. 117–145, 1999.
[98] T. Terlaky and S. Zhang, "A survey on pivot rules for linear programming," Delft Univ. Technol., Faculty Tech. Math. Inf., Delft, The Netherlands, Tech. Rep. 91-99, 1991.
[99] J. A. Tropp, "Greed is good: Algorithmic results for sparse approximation," *IEEE Trans. Inf. Theory*, vol. 50, no. 10, pp. 2231–2242, Oct. 2004.
[100] J. A. Tropp, "Just relax: Convex programming methods for identifying sparse signals," *IEEE Trans. Inf. Theory*, vol. 52, no. 3, pp. 1030–1051, Mar. 2006.
[101] R. J. Vanderbei, *Linear Programming: Foundations and Extensions*, 2nd ed. Norwell, MA: Kluwer, 2001.
[102] P. Vandergheynst and P. Frossard, "Image coding using redundant dictionaries," in *Document and Image Compression*. Boca Raton, FL: CRC Press, 2006.
[103] Y. Zhang, "User's guide to LIPSOL Linear-programming Interior Point SOLvers v0.4," *Opt. Methods Softw.*, pp. 11–12, 1999.
[104] G. M. Ziegler, *Lectures on Polytopes*. New York: Springer, 1994.

Patrick S. Huggins received the B.A. degree in mathematics and computer science from New York University, New York, in 1997, and the Ph.D. degree in computer science from Yale University, New Haven, CT, in 2005.

He is currently a quant at HSBC, New York. His research interests include signal processing, neural computation, computational geometry, optimization, and pattern recognition.



Steven W. Zucker (S'71–M'75–SM'86–F'88) obtained his education at Carnegie-Mellon University, Pittsburgh, PA, and at Drexel University, Philadelphia, PA.

He is the David and Lucile Packard Professor at Yale, New Haven, CT. He directs the Program in Applied Mathematics and is a Professor of Computer Science and of Biomedical Engineering. Before moving to Yale in 1996, he was a Professor at McGill University, Montreal, QC, Canada, and Director of the Program in Artificial Intelligence and Robotics of the Canadian Institute for Advanced Research. He was elected a Fellow of the Canadian Institute for Advanced Research and a Fellow of Churchill College, Cambridge, MA. He and was a post-doctoral Research Fellow in Computer Science at the University of Maryland, College Park. He was a Professor Inviteé at Institut National de Recherche en Informatique et en Automatique, Sophia-Antipolis, France, a Visiting Professor of Computer Science at Tel Aviv University, and a SERC Fellow of the Isaac Newton Institute for Mathematical Sciences, University of Cambridge, Cambridge, U.K. He is an external faculty member of the Sloan-Swartz Center for Theoretical Neurobiology at the University of California, San Francisco. He has authored or co-authored more than 150 papers on computational vision, biological perception, artificial intelligence, and robotics, and serves on the editorial boards of eight journals.