Gradient Projection for Sparse Reconstruction: Application to Compressed Sensing and Other Inverse Problems

Mário A. T. Figueiredo, *Senior Member, IEEE*, Robert D. Nowak, *Senior Member, IEEE*, and Stephen J. Wright

***Abstract*—Many problems in signal processing and statistical inference involve finding sparse solutions to under-determined, or ill-conditioned, linear systems of equations. A standard approach consists in minimizing an objective function which includes a quadratic (squared ℓ_2) error term combined with a sparseness-inducing (ℓ_1) regularization term. *Basis pursuit*, the *least absolute shrinkage and selection operator* (LASSO), wavelet-based deconvolution, and *compressed sensing* are a few well-known examples of this approach. This paper proposes gradient projection (GP) algorithms for the bound-constrained quadratic programming (BCQP) formulation of these problems. We test variants of this approach that select the line search parameters in different ways, including techniques based on the Barzilai–Borwein method. Computational experiments show that these GP approaches perform well in a wide range of applications, often being significantly faster (in terms of computation time) than competing methods. Although the performance of GP methods tends to degrade as the regularization term is de-emphasized, we show how they can be embedded in a continuation scheme to recover their efficient practical performance.**

***Index Terms*—Compressed sensing, convex optimization, deconvolution, gradient projection, quadratic programming, sparseness, sparse reconstruction.**

I. Introduction

A. Background

THERE has been considerable interest in solving the convex unconstrained optimization problem

$$\min_{\mathbf{x}} \frac{1}{2}||\mathbf{y}-\mathbf{A}\mathbf{x}||_2^2 + \tau||\mathbf{x}||_1 \tag{1}$$

where $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{y} \in \mathbb{R}^k$, $\mathbf{A}$ is an $k \times n$ matrix, τ is a nonnegative parameter, $||\mathbf{v}||_2$ denotes the Euclidean norm of $\mathbf{v}$, and $||\mathbf{v}||_1 = \sum_i |v_i|$ is the ℓ_1 norm of $\mathbf{v}$. Problems of the form (1) have become familiar over the past three decades, particularly in statistical and signal processing contexts. From a Bayesian perspective, (1) can be seen as a maximum *a posteriori* criterion for estimating $\mathbf{x}$ from observations

$$\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{n} \tag{2}$$

where $\mathbf{n}$ is white Gaussian noise of variance σ^2, and the prior on $\mathbf{x}$ is Laplacian (that is, $\log p(\mathbf{x}) = -\lambda||\mathbf{x}||_1 + \mathbf{K}$) [1], [25], [54]. Problem (1) can also be viewed as a regularization technique to overcome the ill-conditioned, or even singular, nature of matrix $\mathbf{A}$, when trying to infer $\mathbf{x}$ from noiseless observations $\mathbf{y} = \mathbf{A}\mathbf{x}$ or from noisy observations as in (2).

The presence of the ℓ_1 term encourages small components of $\mathbf{x}$ to become exactly zero, thus promoting sparse solutions [11], [54]. Because of this feature, (1) has been used for more than three decades in several signal processing problems where sparseness is sought; some early references are [12], [37], [50], and [53]. In the 1990s, seminal work on the use of ℓ_1 sparseness-inducing penalties/log-priors appeared in the literature: the now famous *basis pursuit denoising* (BPDN, [11, Section 5]) criterion and the *least absolute shrinkage and selection operator* (LASSO, [54]). For brief historical accounts on the use of the ℓ_1 penalty in statistics and signal processing, see [41], [55].

Problem (1) is closely related to the following convex constrained optimization problems:

$$\min_{\mathbf{x}} ||\mathbf{x}||_1 \quad \text{subject to} \quad ||\mathbf{y}-\mathbf{A}\mathbf{x}||_2^2 \leq \varepsilon \tag{3}$$

and

$$\min_{\mathbf{x}} ||\mathbf{y}-\mathbf{A}\mathbf{x}||_2^2 \quad \text{subject to} \quad ||\mathbf{x}||_1 \leq t \tag{4}$$

where ε and t are nonnegative real parameters. Problem (3) is a *quadratically constrained linear program* (QCLP) whereas (4) is a *quadratic program* (QP). Convex analysis can be used to show that a solution of (3) (for any ε such that this problem is feasible) is either $\mathbf{x} = 0$, or else is a minimizer of (1), for some $\tau > 0$. Similarly, a solution of (4) for any $t \geq 0$ is also a minimizer of (1) for some $\tau \geq 0$. These claims can be proved using [49, Theorem 27.4].

The LASSO approach to regression has the form (4), while the basis pursuit criterion [11, (3.1)] has the form (3) with $\varepsilon = 0$, i.e., a linear program (LP)

$$\min_{\mathbf{x}} ||\mathbf{x}||_1 \quad \text{subject to} \quad \mathbf{y} = \mathbf{A}\mathbf{x}. \tag{5}$$

Manuscript received February 2, 2007; revised September 7, 2007. This work was supported in part by the NSF under Grants CCF-0430504 and CNS-0540147 and by the Fundação para a Ciência e Tecnologia, POSC/FEDER under Grant POSC/EEA-CPS/61271/2004. The associate editor coordinating the review of this manuscript and approving it for publication was Y. Eldar.

M. A. T. Figueiredo is with the Instituto de Telecomunicações and Department of Electrical and Computer Engineering, Instituto Superior Técnico, 1049-001 Lisboa, Portugal.

R. D. Nowak is with the Department of Electrical and Computer Engineering, University of Wisconsin, Madison, WI 53706 USA.

S. J. Wright is with Department of Computer Sciences, University of Wisconsin, Madison, WI 53706 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/JSTSP.2007.910281

Problem (1) also arises in wavelet-based image/signal reconstruction and restoration (namely deconvolution); in those problems, matrix $\mathbf{A}$ has the form $\mathbf{A} = \mathbf{RW}$, where $\mathbf{R}$ is (a matrix representation of) the observation operator (for example, convolution with a blur kernel or a tomographic projection), $\mathbf{W}$ contains a wavelet basis or a redundant dictionary (that is, multiplying by $\mathbf{W}$ corresponds to performing an inverse wavelet transform), and $\mathbf{x}$ is the vector of representation coefficients of the unknown image/signal [24]–[26].

We mention also image restoration problems under total variation (TV) regularization [10], [47]. In the one-dimensional (1-D) case, a change of variables leads to the formulation (1). In 2-D, however, the techniques of this paper cannot be applied directly.

Another intriguing new application for the optimization problems above is *compressed sensing*[1] (CS) [6]–[9], [18]. Recent results show that a relatively small number of random projections of a sparse signal can contain most of its salient information. It follows that if a signal is sparse or approximately sparse in some orthonormal basis, then an accurate reconstruction can be obtained from random projections, which suggests a potentially powerful alternative to conventional Shannon-Nyquist sampling. In the noiseless setting, accurate approximations can be obtained by finding a sparse signal that matches the random projections of the original signal. This problem can be cast as (5), where again matrix $\mathbf{A}$ has the form $\mathbf{A} = \mathbf{RW}$, but in this case $\mathbf{R}$ represents a low-rank randomized sensing matrix (e.g., a $k \times d$ matrix of independent realizations of a random variable), while the columns of $\mathbf{W}$ contain the basis over which the signal has a sparse representation (e.g., a wavelet basis). Problem (1) is a robust version of this reconstruction process, which is resilient to errors and noisy data, and similar criteria have been proposed and analyzed in [8], [32].

B. Previous Algorithms

Several optimization algorithms and codes have been proposed to solve the QCLP (3), the QP (4), the LP (5), and the unconstrained (but nonsmooth) formulation (1). We review that work here, identifying those contributions that are most suitable for signal processing applications, which are the target of this paper.

In the class of applications that motivates this paper, the matrix $\mathbf{A}$ cannot be stored explicitly, and it is costly and impractical to access significant portions of $\mathbf{A}$ and $\mathbf{A}^T\mathbf{A}$. In wavelet-based image reconstruction and some CS problems, for which $\mathbf{A} = \mathbf{RW}$, explicit storage of $\mathbf{A}$, $\mathbf{R}$, or $\mathbf{W}$ is not practical for problems of interesting scale. However, matrix-vector products involving $\mathbf{R}$ and $\mathbf{W}$ can be done quite efficiently. For example, if the columns of $\mathbf{W}$ contain a wavelet basis, then any multiplication of the form $\mathbf{Wv}$ or $\mathbf{W}^T\mathbf{v}$ can be performed by a fast wavelet transform (see Section III-G, for details). Similarly, if $\mathbf{R}$ represents a convolution, then multiplications of the form $\mathbf{Rv}$ or $\mathbf{R}^T\mathbf{v}$ can be performed with the help of the fast Fourier transform (FFT) algorithm. In some CS applications, if the dimension of $\mathbf{y}$ is not too large, $\mathbf{R}$ can be explicitly stored; however, $\mathbf{A}$ is still not available explicitly, because the large and dense nature of $\mathbf{W}$ makes it highly impractical to compute and store $\mathbf{RW}$.

Homotopy algorithms that find the full path of solutions, for all nonnegative values of the scalar parameters in the various formulations (τ in (1), ε in (3), and t in (4)), have been proposed in [22], [39], [46], and [57]. The formulation (4) is addressed in [46], while [57] addresses (1) and (4). The method in [39] provides the solution path for (1), for a range of values of τ. The *least angle regression* (LARS) procedure described in [22] can be adapted to solve the LASSO formulation (4). These are all essentially homotopy methods that perform pivoting operations involving submatrices of $\mathbf{A}$ or $\mathbf{A}^T\mathbf{A}$ at certain critical values of the corresponding parameter (τ, t, or ε). These methods can be implemented so that only the submatrix of $\mathbf{A}$ corresponding to nonzero components of the current vector $\mathbf{x}$ need be known explicitly, so that if $\mathbf{x}$ has few nonzeros, these methods may be competitive even for problems of very large scale. (See for example the `SolveLasso` function in the *SparseLab* toolbox, available from http://www.sparselab.stanford.edu.) In some signal processing applications, however, the number of nonzero $\mathbf{x}$ components may be significant, and since these methods require at least as many pivot operations as there are nonzeros in the solution, they may be less competitive on such problems. The interior-point (IP) approach in [58], which solves a generalization of (4), also requires explicit construction of $\mathbf{A}^T\mathbf{A}$, though the approach could in principle modified to allow iterative solution of the linear system at each primal-dual iteration.

Algorithms that require only matrix-vector products involving $\mathbf{A}$ and $\mathbf{A}^T$ have been proposed in a number of recent works. In [11], the problems (5) and (1) are solved by first reformulating them as "perturbed linear programs" (which are linear programs with additional terms in the objective which are squared norms of the unknowns), then applying a standard primal-dual IP approach [60]. The linear equations or least-squares problems that arise at each IP iteration are then solved with iterative methods such as LSQR [48] or conjugate gradients (CG). Each iteration of these methods requires one multiplication each by $\mathbf{A}$ and $\mathbf{A}^T$. MATLAB implementations of related approaches are available in the *SparseLab* toolbox; see in particular the routines `SolveBP` and `pdco`. For additional details see [51].

Another IP method was recently proposed to solve a quadratic programming reformulation of (1), different from the one used here. Each search step is computed using preconditioned conjugate gradient (PCG) and requires only products by $\mathbf{A}$ and $\mathbf{A}^T$ [36]. The code, available at http://www.stanford.edu/~boyd/l1_ls/, is reported to be faster than competing codes on the problems tested in [36].

The ℓ_1-*magic* suite of codes (which is available at http://www.l1-magic.org) implements algorithms for several of the formulations described in Section I-A. In particular, the formulation (3) is solved by recasting it as a *second-order cone program* (SOCP), then applying a primal log-barrier approach. For each value of the log-barrier parameter, the smooth unconstrained subproblem is solved using Newton's method with line search, where the Newton equations may be

[1]A comprehensive repository of CS literature and software can be found in http://www.dsp.ece.rice.edu/cs/.

solved using CG. (Background on this approach can be found in [6] and [9].) As in [11] and [36], each CG iteration requires only multiplications by $\mathbf{A}$ and $\mathbf{A}^T$; these matrices need not be known or stored explicitly.

Iterative shrinkage/thresholding (IST) algorithms can also be used to handle (1) and only require matrix-vector multiplications involving $\mathbf{A}$ and $\mathbf{A}^T$. Initially, IST was presented as an EM algorithm, in the context of image deconvolution problems [25], [45]. IST can also be derived in a *majorization-minimization* (MM) framework[2] [16], [26] (see also [23], for a related algorithm derived from a different perspective). Convergence of IST algorithms was shown in [13], [16]. IST algorithms are based on bounding the matrix $\mathbf{A}^T\mathbf{A}$ (the Hessian of $\|\mathbf{y}-\mathbf{A}\mathbf{x}\|_2^2$) by a diagonal $\mathbf{D}$ (i.e., $\mathbf{D}-\mathbf{A}^T\mathbf{A}$ is positive semi-definite), thus attacking (1) by solving a sequence of simpler denoising problems. While this bound may be reasonably tight in the case of deconvolution (where $\mathbf{R}$ is usually a square matrix), it may be loose in the CS case, where matrix $\mathbf{R}$ usually has many fewer rows than columns. For this reason, IST may not be as effective for solving (1) in CS applications, as it is in deconvolution problems.

Finally, we mention matching pursuit (MP) and orthogonal MP (OMP) [5], [17], [20], [56], which are greedy schemes to find a sparse representation of a signal on a dictionary of functions. (Matrix $\mathbf{A}$ is seen as an n-element dictionary of k-dimensional signals). MP works by iteratively choosing the dictionary element that has the highest inner product with the current residual, thus most reduces the representation error. OMP includes an extra orthogonalization step, and is known to perform better than standard MP. Low computational cost is one of the main arguments in favor of greedy schemes like OMP, but such methods are not designed to solve any of the optimization problems above. However, if $\mathbf{y}=\mathbf{A}\mathbf{x}$, with $\mathbf{x}$ sparse and the columns of $\mathbf{A}$ sufficiently incoherent, then OMP finds the sparsest representation [56]. It has also been shown that, under similar incoherence and sparsity conditions, OMP is robust to small levels of noise [20].

C. Proposed Approach

The approach described in this paper also requires only matrix-vector products involving $\mathbf{A}$ and $\mathbf{A}^T$, rather than explicit access to $\mathbf{A}$. It is essentially a gradient projection (GP) algorithm applied to a quadratic programming formulation of (1), in which the search path from each iterate is obtained by projecting the negative-gradient direction onto the feasible set. (See [3], for example, for background on gradient projection algorithms.) We refer to our approach as GPSR (*gradient projection for sparse reconstruction*). Various enhancements to this basic approach, together with careful choice of stopping criteria and a final debiasing phase (which finds the least squares fit over the support set of the solution to (1)), are also important in making the method practical and efficient.

Unlike the MM approach, GPSR does not involve bounds on the matrix $\mathbf{A}^T\mathbf{A}$. In contrasts with the IP approaches discussed above, GPSR involves only one level of iteration. (The approaches in [11] and [36] have two iteration levels—an outer IP loop and an inner CG, PCG, or LSQR loop. The ℓ_1-*magic* algorithm for (3) has three nested loops—an outer log-barrier loop, an intermediate Newton iteration, and an inner CG loop.)

GPSR is able to solve a sequence of problems (1) efficiently for a sequence of values of τ. Once a solution has been obtained for a particular τ, it can be used as a "warm-start" for a nearby value. Solutions can therefore be computed for a range of τ values for a small multiple of the cost of solving for a single τ value from a "cold start." This feature of GPSR is somewhat related to that of LARS and other homotopy schemes, which compute solutions for a range of parameter values in succession. In particular, "warm-starting" allows using GPSR within a continuation scheme (as suggested in [31]). IP methods such as those in [11], [36], and ℓ_1-*magic* have been less successful in making effective use of warm-start information, though this issue has been investigated in various contexts (see, e.g., [30], [35], and [61]). To benefit from a warm start, IP methods require the initial point to be not only close to the solution but also sufficiently interior to the feasible set and close to a "central path," which is difficult to satisfy in practice.

II. Proposed Formulation

A. Formulation as a Quadratic Program

The first key step of our GPSR approach is to express (1) as a quadratic program; as in [28], this is done by splitting the variable $\mathbf{x}$ into its positive and negative parts. Formally, we introduce vectors $\mathbf{u}$ and $\mathbf{v}$ and make the substitution

$$\mathbf{x}=\mathbf{u}-\mathbf{v},\ \ \mathbf{u}\geq 0,\ \ \mathbf{v}\geq 0. \tag{6}$$

These relationships are satisfied by $u_i = (x_i)_+$ and $v_i = (-x_i)_+$ for all $i = 1,2,\ldots,n$, where $(\cdot)_+$ denotes the *positive-part operator* defined as $(x)_+ = \max\{0,x\}$. We thus have $\|\mathbf{x}\|_1 = \mathbf{1}_n^T\mathbf{u}+\mathbf{1}_n^T\mathbf{v}$, where $\mathbf{1}_n=[1,1,\ldots,1]^T$ is the vector consisting of n ones, so (1) can be rewritten as the following bound-constrained quadratic program (BCQP):

$$\begin{aligned}\min_{\mathbf{u},\mathbf{v}}\quad & \frac{1}{2}\|\mathbf{y}-\mathbf{A}(\mathbf{u}-\mathbf{v})\|_2^2+\tau\mathbf{1}_n^T\mathbf{u}+\tau\mathbf{1}_n^T\mathbf{v},\\ \text{s.t.}\quad & \mathbf{u}\geq 0\\ & \mathbf{v}\geq 0.\end{aligned} \tag{7}$$

Note that the ℓ_2-norm term is unaffected if we set $\mathbf{u}\leftarrow\mathbf{u}+\mathbf{s}$ and $\mathbf{v}\leftarrow\mathbf{v}+\mathbf{s}$, where $\mathbf{s}\geq 0$ is a shift vector. However such a shift increases the other terms by $2\tau\mathbf{1}_n^T\mathbf{s}\geq 0$. It follows that, at the solution of the problem (7), $u_i=0$ or $v_i=0$, for $i=1,2,\ldots,n$, so that in fact $u_i=(x_i)_+$ and $v_i=(-x_i)_+$ for all $i=1,2,\ldots,n$, as desired.

Problem (7) can be written in more standard BCQP form

$$\begin{aligned}\min_{\mathbf{z}}\quad & \mathbf{c}^T\mathbf{z}+\frac{1}{2}\mathbf{z}^T\mathbf{B}\mathbf{z}\equiv F(\mathbf{z}),\\ \text{s.t.}\quad & \mathbf{z}\geq 0\end{aligned} \tag{8}$$

where

$$\mathbf{z}=\begin{bmatrix}\mathbf{u}\\ \mathbf{v}\end{bmatrix},\quad \mathbf{b}=\mathbf{A}^T\mathbf{y},\quad \mathbf{c}=\tau\mathbf{1}_{2n}+\begin{bmatrix}-\mathbf{b}\\ \mathbf{b}\end{bmatrix}$$

[2]Also known as bound optimization algorithms (BOA). For a general introduction to MM/BOA, see [33].

and

$$\mathbf{B} = \begin{bmatrix} \mathbf{A}^T\mathbf{A} & -\mathbf{A}^T\mathbf{A} \\ -\mathbf{A}^T\mathbf{A} & \mathbf{A}^T\mathbf{A} \end{bmatrix}. \tag{9}$$

B. Dimensions of the BCQP

It may be observed that the dimension of problem (8) is twice that of the original problem (1): $\mathbf{x} \in \mathbb{R}^n$, while $\mathbf{z} \in \mathbb{R}^{2n}$. However, this increase in dimension has only a minor impact. Matrix operations involving $\mathbf{B}$ can be performed more economically than its size suggests, by exploiting its particular structure (9). For a given $\mathbf{z} = \left[\mathbf{u}^T\ \mathbf{v}^T\right]^T$, we have

$$\mathbf{B}\mathbf{z} = \mathbf{B}\begin{bmatrix}\mathbf{u}\\ \mathbf{v}\end{bmatrix} = \begin{bmatrix} \mathbf{A}^T\mathbf{A}(\mathbf{u}-\mathbf{v}) \\ -\mathbf{A}^T\mathbf{A}(\mathbf{u}-\mathbf{v}) \end{bmatrix}$$

indicating that $\mathbf{B}\mathbf{z}$ can be found by computing the vector difference $\mathbf{u}-\mathbf{v}$ and then multiplying once each by $\mathbf{A}$ and $\mathbf{A}^T$. Since $\nabla F(\mathbf{z}) = \mathbf{c} + \mathbf{B}\mathbf{z}$ (the gradient of the objective function in (8)), we conclude that computation of $\nabla F(\mathbf{z})$ requires one multiplication each by $\mathbf{A}$ and $\mathbf{A}^T$, assuming that $\mathbf{c}$, which depends on $\mathbf{b} = \mathbf{A}^T\mathbf{y}$, is pre-computed at the start of the algorithm.

Another common operation in the GP algorithms described below is to find the scalar $\mathbf{z}^T\mathbf{B}\mathbf{z}$ for a given $\mathbf{z} = [\mathbf{u}^T, \mathbf{v}^T]^T$. It is easy to see that

$$\mathbf{z}^T\mathbf{B}\mathbf{z} = (\mathbf{u}-\mathbf{v})^T\mathbf{A}^T\mathbf{A}(\mathbf{u}-\mathbf{v}) = \|\mathbf{A}(\mathbf{u}-\mathbf{v})\|_2^2$$

indicating that this quantity can be calculated using only a single multiplication by $\mathbf{A}$. Since $F(\mathbf{z}) = (1/2)\mathbf{z}^T\mathbf{B}\mathbf{z} + \mathbf{c}^T\mathbf{z}$, it follows that evaluation of $F(\mathbf{z})$ also requires only one multiplication by $\mathbf{A}$.

C. A Note Concerning Nonnegative Solutions

It is worth pointing out that when the solution of (1) is known in advance to be nonnegative, we can directly rewrite the problem as

$$\begin{aligned} \min_{\mathbf{x}} \quad & (\tau\mathbf{1}_n - \mathbf{A}^T\mathbf{y})^T\mathbf{x} + \frac{1}{2}\mathbf{x}^T\mathbf{A}^T\mathbf{A}\mathbf{x}, \\ \text{s.t.} \quad & \mathbf{x} \geq 0. \end{aligned} \tag{10}$$

This problem is, as (8), a BCQP, and it can be solved with the same algorithms. However the presence of the constraint $\mathbf{x} \geq 0$ allows us to avoid splitting the variables into positive and negative parts.

III. Gradient Projection Algorithms

In this section we discuss GP techniques for solving (8). In our approaches, we move from iterate $\mathbf{z}^{(k)}$ to iterate $\mathbf{z}^{(k+1)}$ as follows. First, we choose some scalar parameter $\alpha^{(k)} > 0$ and set

$$\mathbf{w}^{(k)} = \left(\mathbf{z}^{(k)} - \alpha^{(k)}\nabla F\left(\mathbf{z}^{(k)}\right)\right)_+. \tag{11}$$

We then choose a second scalar $\lambda^{(k)} \in [0,1]$ and set

$$\mathbf{z}^{(k+1)} = \mathbf{z}^{(k)} + \lambda^{(k)}\left(\mathbf{w}^{(k)} - \mathbf{z}^{(k)}\right). \tag{12}$$

Our approaches described next differ in their choices of $\alpha^{(k)}$ and $\lambda^{(k)}$.

A. Basic Gradient Projection: The GPSR-Basic Algorithm

In the basic approach, we search from each iterate $\mathbf{z}^{(k)}$ along the negative gradient $-\nabla F(\mathbf{z}^{(k)})$, projecting onto the nonnegative orthant, and performing a backtracking line search until a sufficient decrease is attained in F. (Bertsekas [3, p. 226] refers to this strategy as "Armijo rule along the projection arc.") We use an initial guess for $\alpha^{(k)}$ that would yield the exact minimizer of F along this direction if no new bounds were to be encountered. Specifically, we define the vector $\mathbf{g}^{(k)}$ by

$$\mathbf{g}_i^{(k)} = \begin{cases} \left(\nabla F\left(\mathbf{z}^{(k)}\right)\right)_i, & \text{if } \mathbf{z}_i^{(k)} > 0 \text{ or } \left(\nabla F\left(\mathbf{z}^{(k)}\right)\right)_i < 0, \\ 0, & \text{otherwise.} \end{cases}$$

We then choose the initial guess to be

$$\alpha_0 = \arg\min_{\alpha} F\left(\mathbf{z}^{(k)} - \alpha\mathbf{g}^{(k)}\right)$$

which we can compute explicitly as

$$\alpha_0 = \frac{\left(\mathbf{g}^{(k)}\right)^T\mathbf{g}^{(k)}}{\left(\mathbf{g}^{(k)}\right)^T B\mathbf{g}^{(k)}}. \tag{13}$$

To protect against values of α_0 that are too small or too large, we confine it to the interval $[\alpha_{\min}, \alpha_{\max}]$, where $0 < \alpha_{\min} < \alpha_{\max}$. (In this connection, we define the operator $\mathrm{mid}(a,b,c)$ to be the middle value of its three scalar arguments.) This technique for setting α_0 is apparently novel, and produces an acceptable step much more often than the earlier choice of α_0 as the minimizer of F along the direction $-\nabla F(\mathbf{z}^{(k)})$, ignoring the bounds.

The complete algorithm is defined as follows.

Step 0 (initialization): Given $\mathbf{z}^{(0)}$, choose parameters $\beta \in (0,1)$ and $\mu \in (0, 1/2)$; set $k = 0$.
Step 1: Compute α_0 from (13), and replace α_0 by $\mathrm{mid}(\alpha_{\min}, \alpha_0, \alpha_{\max})$.
Step 2 (backtracking line search): Choose $\alpha^{(k)}$ to be the first number in the sequence $\alpha_0, \beta\alpha_0, \beta^2\alpha_0, \ldots$ such that

$$F\left(\left(\mathbf{z}^{(k)} - \alpha^{(k)}\nabla F\left(\mathbf{z}^{(k)}\right)\right)_+\right) \leq F\left(\mathbf{z}^{(k)}\right) - \mu\nabla F\left(\mathbf{z}^{(k)}\right)^T\left(\mathbf{z}^{(k)} - \left(\mathbf{z}^{(k)} - \alpha^{(k)}\nabla F\left(\mathbf{z}^{(k)}\right)\right)_+\right),$$

and set $\mathbf{z}^{(k+1)} = \left(\mathbf{z}^{(k)} - \alpha^{(k)}\nabla F(\mathbf{z}^{(k)})\right)_+$.
Step 3: Perform convergence test and terminate with approximate solution $\mathbf{z}^{(k+1)}$ if it is satisfied; otherwise set $k \leftarrow k+1$ and return to **Step 1**.

Termination tests used in Step 3 are discussed below in Section III-D.

The computation at each iteration consists of matrix-vector multiplications involving $\mathbf{A}$ and $\mathbf{A}^T$, together with a few (less significant) inner products involving vectors of length n. Step 2 requires evaluation of F for each value of $\alpha^{(k)}$ tried, where each such evaluation requires a single multiplication by $\mathbf{A}$. Once the value of $\alpha^{(k)}$ is determined, we can find $\mathbf{z}^{(k+1)}$ and then

$\nabla F(\mathbf{z}^{(k+1)})$ with one more multiplication by $\mathbf{A}^T$. Another multiplication by $\mathbf{A}$ suffices to calculate the denominator in (13) at the start of each iteration. In total, the number of multiplications by $\mathbf{A}$ or $\mathbf{A}^T$ per iteration is two plus the number of values of $\alpha^{(k)}$ tried in Step 2.

B. Barzilai–Borwein Gradient Projection: The GPSR-BB Algorithm

Algorithm GPSR-Basic ensures that the objective function F decreases at every iteration. Recently, considerable attention has been paid to an approach due to Barzilai and Borwein (BB) [2] that does not have this property. This approach was originally developed in the context of unconstrained minimization of a smooth nonlinear function F. It calculates each step by the formula $\boldsymbol{\delta}^{(k)} = -H_k^{-1}\nabla F(\mathbf{z}^{(k)})$, where H_k is an approximation to the Hessian of F at $\mathbf{z}^{(k)}$. Barzilai and Borwein propose a particularly simple choice for the approximation H_k: they set it to be a multiple of the identity $H_k = \eta^{(k)} I$, where $\eta^{(k)}$ is chosen so that this approximation has similar behavior to the true Hessian over the most recent step, that is,

$$\nabla F\left(\mathbf{z}^{(k)}\right) - \nabla F\left(\mathbf{z}^{(k-1)}\right) \approx \eta^{(k)}\left[\mathbf{z}^{(k)} - \mathbf{z}^{(k-1)}\right]$$

with $\eta^{(k)}$ chosen to satisfy this relationship in the least-squares sense. In the unconstrained setting, the update formula is

$$\mathbf{z}^{(k+1)} = \mathbf{z}^{(k)} - \left(\eta^{(k)}\right)^{-1}\nabla F\left(\mathbf{z}^{(k)}\right)$$

and this step is taken even if it yields an increase in F. This strategy is proved analytically in [2] to be effective on simple problems. Numerous variants have been proposed recently, and subjected to a good deal of theoretical and computational evaluation.

The BB approach has been extended to BCQPs in [15], [52]. The approach described here is simply that of [52, Sec. 2.2]. We choose λ_k in (12) as the exact minimizer over the interval [0,1] and choose $\eta^{(k)}$ at each iteration in the manner described above, except that $\alpha^{(k)} = (\eta^{(k)})^{-1}$ is restricted to the interval $[\alpha_{\min}, \alpha_{\max}]$. In defining the value of $\alpha^{(k+1)}$ in Step 3 below, we make use of the fact that for F defined in (3), we have

$$\nabla F\left(\mathbf{z}^{(k)}\right) - \nabla F\left(\mathbf{z}^{(k-1)}\right) = \mathbf{B}\left(\mathbf{z}^{(k)} - \mathbf{z}^{(k-1)}\right).$$

Step 0 (initialization): Given $\mathbf{z}^{(0)}$, choose parameters $\alpha_{\min}$, $\alpha_{\max}$, $\alpha^{(0)} \in [\alpha_{\min}, \alpha_{\max}]$, and set $k = 0$.
Step 1: Compute step:

$$\boldsymbol{\delta}^{(k)} = \left(\mathbf{z}^{(k)} - \alpha^{(k)}\nabla F\left(\mathbf{z}^{(k)}\right)\right)_+ - \mathbf{z}^{(k)}. \tag{14}$$

Step 2 (line search): Find the scalar $\lambda^{(k)}$ that minimizes $F(\mathbf{z}^{(k)} + \lambda^{(k)}\boldsymbol{\delta}^{(k)})$ on the interval $\lambda^{(k)} \in [0, 1]$, and set $\mathbf{z}^{(k+1)} = \mathbf{z}^{(k)} + \lambda^{(k)}\boldsymbol{\delta}^{(k)}$.

Step 3 (**update** α): compute

$$\gamma^{(k)} = \left(\boldsymbol{\delta}^{(k)}\right)^T \mathbf{B}\boldsymbol{\delta}^{(k)}; \tag{15}$$

if $\gamma^{(k)} = 0$, let $\alpha^{(k+1)} = \alpha_{\max}$, otherwise

$$\alpha^{(k+1)} = \operatorname{mid}\left\{\alpha_{\min}, \frac{\left\|\boldsymbol{\delta}^{(k)}\right\|_2^2}{\gamma^{(k)}}, \alpha_{\max}\right\}.$$

Step 4: Perform convergence test and terminate with approximate solution $\mathbf{z}^{(k+1)}$ if it is satisfied; otherwise set $k \leftarrow k + 1$ and return to **Step 1**.

Since F is quadratic, the line search parameter $\lambda^{(k)}$ in Step 2 can be calculated simply using the following closed-form expression:

$$\lambda^{(k)} = \operatorname{mid}\left\{0, \frac{\left(\boldsymbol{\delta}^{(k)}\right)^T \nabla F\left(\mathbf{z}^{(k)}\right)}{\left(\boldsymbol{\delta}^{(k)}\right)^T \mathbf{B}\boldsymbol{\delta}^{(k)}}, 1\right\}.$$

(When $(\boldsymbol{\delta}^{(k)})^T\mathbf{B}\boldsymbol{\delta}^{(k)} = 0$, we set $\lambda^{(k)} = 1$.) The use of this parameter $\lambda^{(k)}$ removes one of the salient properties of the Barzilai–Borwein approach, namely, the possibility that F may increase on some iterations. Nevertheless, in our problems, it appeared to improve performance over the more standard nonmonotone variant, which sets $\lambda^{(k)} \equiv 1$. We also tried other variants of the Barzilai–Borwein approach, including one proposed in [15], which alternates between two definitions of $\alpha^{(k)}$. The difference in performance were very small, so we focus our presentation on the method described above.

In earlier testing, we experimented with other variants of GP, including the GPCG approach of [43] and the proximal-point approach of [59]. The GPCG approach runs into difficulties because the projection of the Hessian $\mathbf{B}$ onto most faces of the positive orthant defined by $\mathbf{z} \geq 0$ is singular, so the inner CG loop in this algorithm tends to fail.

C. Termination

The decision about when an approximate solution is of sufficiently high quality to terminate the algorithms is a difficult one. We wish for the approximate solution $\mathbf{z}$ to be reasonably close to a solution $\mathbf{z}^*$ and/or that the function value $F(\mathbf{z})$ be reasonably close to $F(\mathbf{z}^*)$, but at the same time we wish to avoid the excessive computation involved in finding an overly accurate solution. For the problem (7), given that variable selection is the main motivation of the formulation (1) and that a debiasing step may be carried out in a postprocessing phase (see Section III-E), we wish the nonzero components of the approximate solution $\mathbf{z}$ to be close to the nonzeros of a true solution $\mathbf{z}^*$.

These considerations motivate a number of possible termination criteria. One simple criterion is

$$\left\|\mathbf{z} - (\mathbf{z} - \bar{\alpha}\nabla F(\mathbf{z}))_+\right\| \leq \texttt{tolP} \tag{16}$$

where `tolP` is a small parameter and $\bar{\alpha}$ is a positive constant. This criterion is motivated by the fact that the left-hand side is continuous in $\mathbf{z}$ and zero if and only if $\mathbf{z}$ is optimal. A second,

similar criterion is motivated by perturbation results for linear complementarity problems (LCP). There is a constant C_{LCP} such that

$$\mathrm{dist}(\mathbf{z}, \mathcal{S}) \leq C_{\mathrm{LCP}} \, ||\min(\mathbf{z}, \nabla F(\mathbf{z}))||$$

where $\mathcal{S}$ denotes the solution set of (8), $\mathrm{dist}(\cdot)$ is the distance operator, and the $\min$ on the right-hand side is taken componentwise [14]. With this bound in mind, we can define a convergence criterion as follows:

$$||\min(\mathbf{z}, \nabla F(\mathbf{z}))|| \leq \texttt{tolP}. \tag{17}$$

A third criterion proposed recently in [36] is based on duality theory for the original formulation (1). It can be shown that the dual of (1) is

$$\begin{aligned} \max_{\mathbf{s}} \quad & -\frac{1}{2}\mathbf{s}^T\mathbf{s} - \mathbf{y}^T\mathbf{s} \\ \text{s.t.} \quad & -\tau\mathbf{1}_n \leq \mathbf{A}^T\mathbf{s} \leq \tau\mathbf{1}_n. \end{aligned} \tag{18}$$

If $\mathbf{s}$ is feasible for (18), then

$$\frac{1}{2}||\mathbf{y} - \mathbf{A}\mathbf{x}||_2^2 + \tau||\mathbf{x}||_1 + \frac{1}{2}\mathbf{s}^T\mathbf{s} + \mathbf{y}^T\mathbf{s} \geq 0 \tag{19}$$

with equality attained if and only if $\mathbf{x}$ is a solution of (1) and $\mathbf{s}$ is a solution of (18). To define a termination criterion, we invert the transformation in (6) to obtain a candidate $\mathbf{x}$, and then construct a feasible $\mathbf{s}$ as follows:

$$\mathbf{s} \equiv \tau \frac{\mathbf{A}\mathbf{x} - \mathbf{y}}{||\mathbf{A}^T(\mathbf{A}\mathbf{x} - \mathbf{y})||_\infty}$$

(see [36]). Substituting these values into the left-hand side of (19), we can declare termination when this quantity falls below a threshold tolP. Note that this quantity is an upper bound on the gap between $F(\mathbf{z})$ and the optimal objective value $F(\mathbf{z}^*)$.

None of the criteria discussed so far take account of the nonzero indices of $\mathbf{z}$ or of how these have changed in recent iterations. In our fourth criterion, termination is declared when the set of nonzero indices of an iterate $\mathbf{z}^{(k)}$ changes by a relative amount of less than a specified threshold tolA. Specifically, we define

$$\begin{aligned} \mathcal{I}_k &= \left\{ i | z_i^{(k)} \neq 0 \right\}, \\ \mathcal{C}_k &= \{ i | (i \in \mathcal{I}_k \text{ and } i \notin \mathcal{I}_{k-1}) \text{or} (i \notin \mathcal{I}_k \text{ and } i \in \mathcal{I}_{k-1}) \} \end{aligned}$$

and terminate if

$$|\mathcal{C}_k| / |\mathcal{I}_k| \leq \texttt{tolA}. \tag{20}$$

This criterion is well suited to the class of problems addressed in this paper (where we expect the cardinality of $\mathcal{I}_k$, in later stages of the algorithm, to be much less than the dimension of $\mathbf{z}$), and to algorithms of the gradient projection type, which generate iterates on the boundary of the feasible set. However, it does not work for general BCQPs (e.g., in which *all* the components of $\mathbf{z}$ are nonzero at the solution) or for algorithms that generate iterates in the interior of the feasible set.

It is difficult to choose a termination criterion from among these options that performs well on all data sets and in all contexts. In the tests described in Section IV, unless otherwise noted, we use (17), with $\texttt{tolP} = 10^{-2}$, which appeared to yield fairly consistent results. We may also impose some large upper limit maxiter on the number of iterations.

D. Debiasing

Once an approximate solution has been obtained using one of the algorithms above, we optionally perform a *debiasing* step. The computed solution $\mathbf{z} = [\mathbf{u}^T, \mathbf{v}^T]^T$ is converted to an approximate solution $\mathbf{x}_{GP} = \mathbf{u} - \mathbf{v}$. The zero components of $\mathbf{x}_{GP}$ are fixed at zero, and the least-squares objective $||\mathbf{y} - \mathbf{A}\mathbf{x}||_2^2$ is then minimized subject to this restriction using a CG algorithm (see for example [44, ch. 5]). In our code, the CG iteration is terminated when

$$||\mathbf{y} - \mathbf{A}\mathbf{x}||_2^2 \leq \texttt{tolD} ||\mathbf{y} - \mathbf{A}\mathbf{x}_{GP}||_2^2 \tag{21}$$

where tolD is a small positive parameter. We also restrict the number of CG steps in the debiasing phase to maxiterD.

Essentially, the problem (1) is being used to select the "explanatory" variables (components of $\mathbf{x}$), while the debiasing step chooses the optimal values for these components according to a least-squares criterion (without the regularization term $\tau||\mathbf{x}||_1$). Similar techniques have been used in other ℓ_1-based algorithms, e.g., [42]. It is also worth pointing out that debiasing is not always desirable. Shrinking the selected coefficients can mitigate unusually large noise deviations [19], a desirable effect that may be undone by debiasing.

E. Warm Starting and Continuation

The gradient projection approach benefits from a good starting point. This suggests that we can use the solution of (1), for a given value of τ, to initialize GPSR in solving (1) for a nearby value of τ. The second solve will typically take fewer iterations than the first one; the number of iterations depends on the closeness of the values of τ and the closeness of the solutions. Using this warm-start technique, we can efficiently solve for a sequence of values of τ. We note that it is important to use the non-debiased solution as starting point; debiasing may move the iterates away from the true minimizer of (1).

One motivation for solving for a range of τ values is that we often wish to obtain solutions for a range of values of τ, possibly using some test based on the solution sparsity and the goodness of least-squares fit to choose the "best" solution from among these possibilities.

Another important application of warm-starting is *continuation*, as recently suggested in [31]. It has been noted recently that the speed of GPSR may degrade considerably for smaller values of the regularization parameter τ. However, if we use GPSR to minimize (1) for a larger value of τ, then decrease τ in steps toward its desired value, running GPSR with warm-start for each successive value of τ, we are often able to identify the solution much more efficiently than if we just ran GPSR once for the desired value of τ from a cold start. We illustrate this claim with a computational example in Section IV-D.

F. Analysis of Computational Cost

It is not possible to accurately predict the number of GPSR-Basic and GPSR-BB iterations required to find an approximate solution. We can however analyze the cost of each iteration of these algorithms. The main computational cost per iteration is

a small number of inner products, vector-scalar multiplications, and vector additions, each requiring n or $2n$ floating-point operations, plus a modest number of multiplications by $\mathbf{A}$ and $\mathbf{A}^T$. When $\mathbf{A} = \mathbf{RW}$, these operations entail a small number of multiplications by $\mathbf{R}$, $\mathbf{R}^T$, $\mathbf{W}$, and $\mathbf{W}^T$. The cost of each CG iteration in the debiasing phase is similar but lower; just one multiplication by each of $\mathbf{R}$, $\mathbf{R}^T$, $\mathbf{W}$, and $\mathbf{W}^T$ plus a number of vector operations. We next analyze the cost of multiplications by $\mathbf{R}$, $\mathbf{R}^T$, $\mathbf{W}$, and $\mathbf{W}^T$ for various typical problems; let us begin by recalling that $\mathbf{A} = \mathbf{RW}$ is a $k \times n$ matrix, and that $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{y} \in \mathbb{R}^k$. Thus, if $\mathbf{R}$ has dimensions $k \times d$, then $\mathbf{W}$ must be a $d \times n$ matrix.

If $\mathbf{W}$ contains an orthogonal wavelet basis ($d = n$), matrix-vector products involving $\mathbf{W}$ or $\mathbf{W}^T$ can be implemented using fast wavelet transform algorithms with $O(n)$ cost [40], instead of the $O(n^2)$ cost of a direct matrix-vector product. Thus, the cost of a product by $\mathbf{A}$ or $\mathbf{A}^T$ is $O(n)$ plus that of multiplying by $\mathbf{R}$ or $\mathbf{R}^T$ which, with a direct implementation, is $O(kn)$. When using redundant translation-invariant wavelet systems, $\mathbf{W}$ is $d \times d(\log_2(d) + 1)$, but the corresponding matrix-vector products can be done with $O(d \log d)$ cost, using fast undecimated wavelet transform algorithms [40].

As mentioned above, direct implementations of products by $\mathbf{R}$ and $\mathbf{R}^T$ have $O(kd)$ cost. However, in some cases, these products can be carried out with significantly lower cost. For example, in image deconvolution problems [25], $\mathbf{R}$ is a $k \times k$ ($d = k$) block-Toeplitz matrix with Toeplitz blocks (representing 2-D convolutions) and these products can be performed in the discrete Fourier domain using the FFT, with $O(k \log k)$ cost, instead of the $O(k^2)$ cost of a direct implementation. If the blur kernel support is very small (say l pixels) these products can be done with even lower cost, $O(kl)$, by implementing the corresponding convolution. Also, in certain applications of CS, such as MR image reconstruction [38], $\mathbf{R}$ is formed from a subset of the discrete Fourier transform basis, so the cost is $O(k \log k)$ using the FFT.

IV. Experiments

This section describes some experiments testifying to the very good performance of the proposed algorithms in several types of problems of the form (1). These experiments include comparisons with state-of-the-art approaches, namely IST [16], [25], and the recent $l1\_ls$ package, which was shown in [36] to outperform all previous methods, including the ℓ_1-*magic* toolbox and the homotopy method from [21]. The algorithms discussed in Section III are written in MATLAB and are freely available for download from http://www.lx.it.pt/~mtf/GPSR/.

For the GPSR-BB algorithm, we set $\alpha_{\min} = 10^{-30}$, $\alpha_{\max} = 10^{30}$; the performance is insensitive to these choices, similar results are obtained for other small settings of $\alpha_{\min}$ and large values of $\alpha_{\max}$. We discuss results also for a nonmonotone version of the GPSR-BB algorithm, in which $\lambda_k \equiv 1$. In GPSR-Basic, we used $\beta = 0.5$ and $\mu = 0.1$.

A. Compressed Sensing (CS)

In our first experiment, we consider a typical CS scenario (similar to the one in [36]), where the goal is to reconstruct a length-n sparse signal (in the canonical basis) from k observations, where $k < n$. In this case, the $k \times n$ matrix $\mathbf{A}$ is obtained by first filling it with independent samples of a standard Gaussian distribution and then orthonormalizing the rows. In this example, $n = 4096$, $k = 1024$, the original signal $\mathbf{x}$ contains 160 randomly placed ± 1 spikes, and the observation $\mathbf{y}$ is generated according to (2), with $\sigma^2 = 10^{-4}$. Parameter τ is chosen as suggested in [36]

$$\tau = 0.1\|\mathbf{A}^T\mathbf{y}\|_\infty. \tag{22}$$

Notice that for $\tau \geq \|\mathbf{A}^T\mathbf{y}\|_\infty$ the unique minimum of (1) is the zero vector [29], [36].

The original signal and the estimate obtained by solving (1) using the monotone version of the GPSR-BB (which is essentially the same as that produced by the nonmonotone GPSR-BB and GPSR-Basic) are shown in Fig. 1. Also shown in Fig. 1 is the reconstruction obtained after the debiasing procedure described in Section III-E; although GPSR-BB does an excellent job at locating the spikes, the debiased reconstruction exhibits a much lower mean squared error[3] (MSE) with respect to the original signal. Finally, Fig. 1 also depicts the solution of minimal ℓ_2-norm to the undetermined system $\mathbf{y} = \mathbf{Ax}$, which is equal to $\mathbf{A}^T(\mathbf{AA}^T)^{-1}\mathbf{y}$.

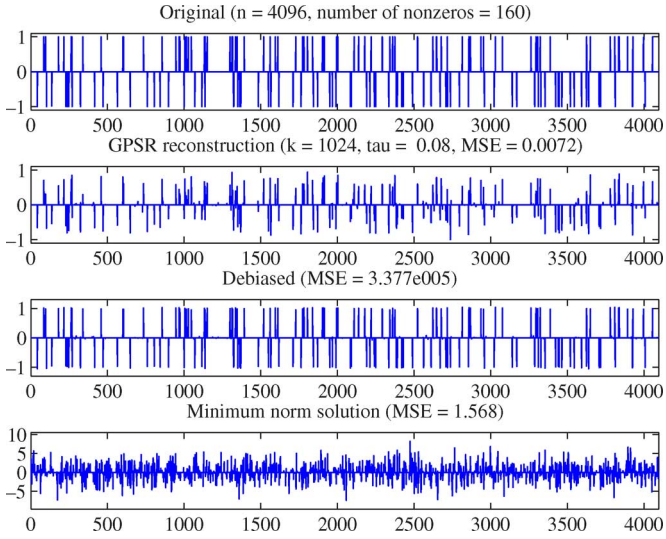

Fig. 1. From top to bottom: original signal, reconstruction via the minimization of (1) obtained by GPSR-BB, and reconstruction after debiasing, the minimum norm solution given by $\mathbf{A}^T(\mathbf{AA}^T)^{-1}\mathbf{y}$.

In Fig. 2, we plot the evolution of the objective function (without debiasing) versus iteration number and CPU time, for GPSR-Basic and both versions of GPSR-BB. The GPSR-BB variants are slightly faster, but the performance of all three codes is quite similar on this problem. Fig. 3 shows how the objective function (1) and the MSE evolve in the debiasing phase. Notice that the objective function (1) increases during the debiasing phase, since we are minimizing a different function in this phase.

Table I reports average CPU times (over ten experiments) required by the three GPSR algorithms as well as by $l1\_ls$ and

[3] $MSE = (1/n)\|\widehat{\mathbf{x}} - \mathbf{x}\|_2^2$, where $\widehat{\mathbf{x}}$ is an estimate of $\mathbf{x}$.

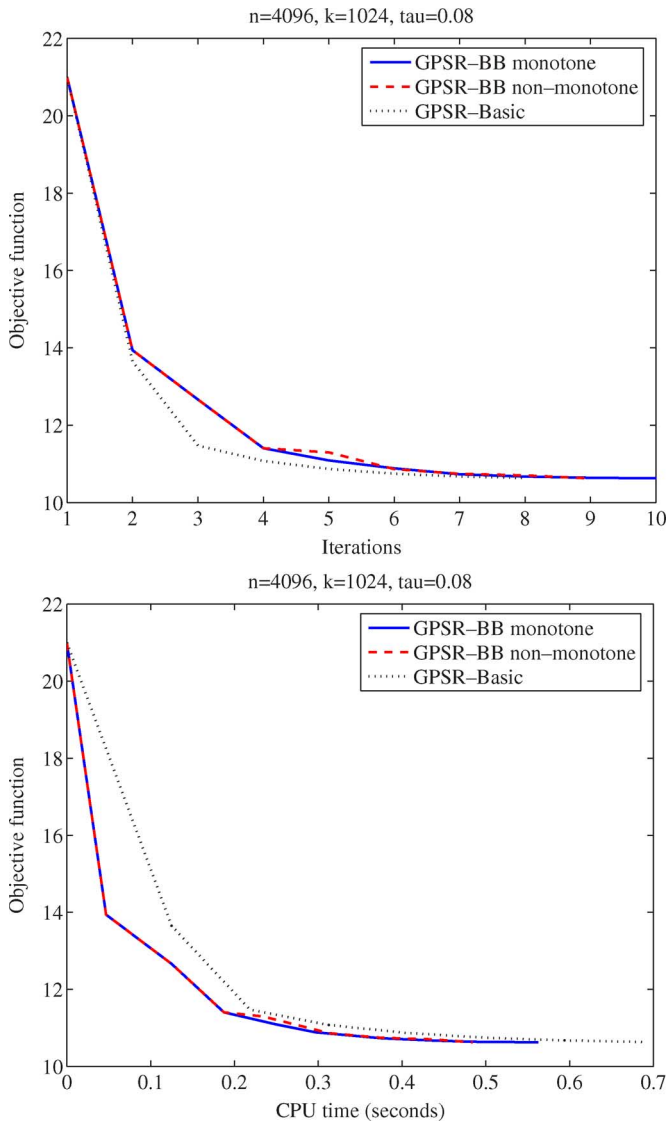

Fig. 2. Objective function plotted against iteration number and CPU time, for GPSR-Basic and the monotone and nonmonotone versions of the GPSR-BB, corresponding to the experiment illustrated in Fig. 1.

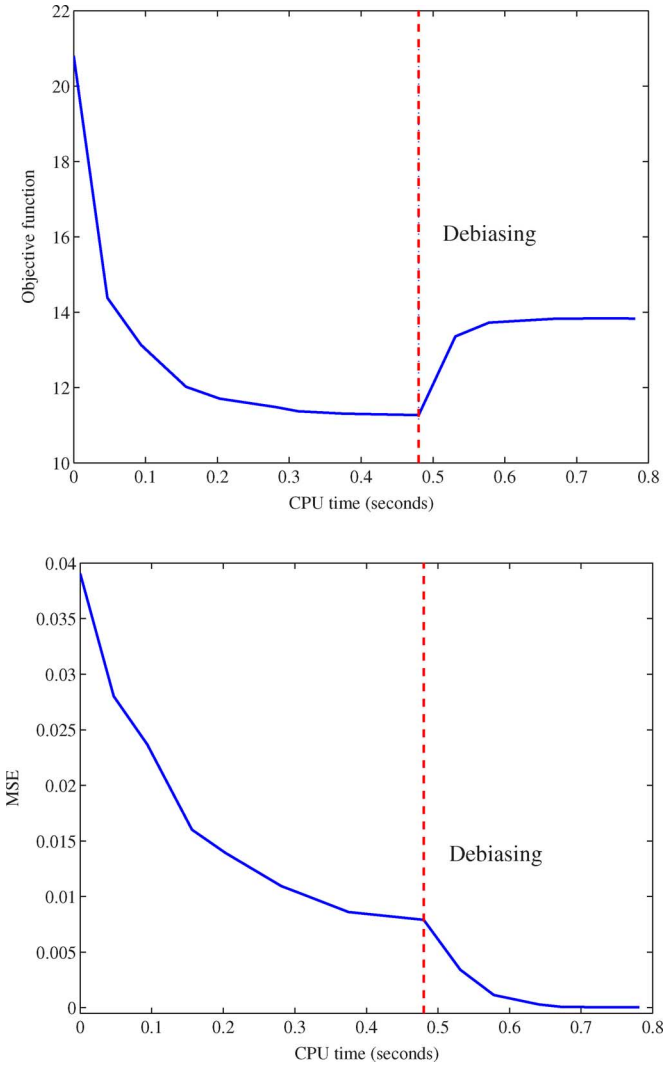

Fig. 3. Evolution of the objective function and reconstruction MSE, versus CPU time, including the debiasing phase, corresponding to the experiment illustrated in Fig. 1.

IST. To perform this comparison, we first run the *l1_ls* algorithm and then each of the other algorithms until each reaches the same value of the objective function reached by *l1_ls*. The results in this table show that, for this problem, all GPSR variants are about one order of magnitude faster than *l1_ls* and five times faster than IST.

An indirect performance comparison with other codes on this problem can be made by referring to [36, Table 1], which shows that *l1_ls* outperforms the homotopy method from [21] (6.9 s versus 11.3 s). It also outperforms ℓ_1*-magic* by about two orders of magnitude and the `pdco` algorithms from *SparseLab* by about one order of magnitude.

B. Comparison With OMP

Next, we compare the computational efficiency of GPSR algorithms against OMP, often regarded as a highly efficient method that is especially well-suited to very sparse cases. We use two efficient MATLAB implementations of OMP: the

TABLE I
CPU TIMES (AVERAGE OVER TEN RUNS) OF SEVERAL ALGORITHMS ON THE EXPERIMENT OF FIG. 1

Algorithm	CPU time (seconds)
GPSR-BB monotone	0.59
GPSR-BB nonmonotone	0.51
GPSR-Basic	0.69
GPSR-BB monotone + debias	0.89
GPSR-BB nonmonotone + debias	0.82
GPSR-Basic + debias	0.98
l1_ls	6.56
IST	2.76

`greed_omp_qr` function of the *Sparsify* toolbox (available at http://www.see.ed.ac.uk/~tblumens/sparsify), which is based on QR factorization [5], [17], and the function `SolveOMP` of the *SparseLab* toolbox, which is based on the Cholesky factorization. Because `greed_omp_qr` requires each column of the matrix $\mathbf{R}$ to have unit norm, we use matrices with this property in all our comparisons.

Since OMP is not an optimization algorithm for minimizing (1) (or any other objective function), it is not obvious how to

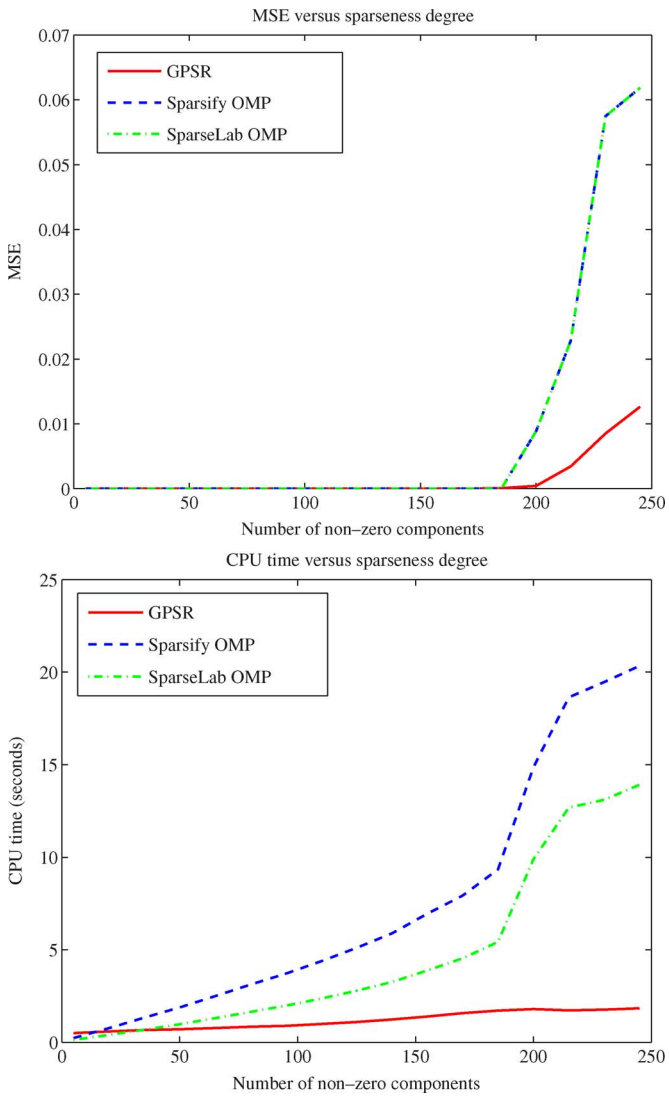

Fig. 4. Average MSE and CPU times for GPSR and two implementations of OMP, as a function of the number of nonzero components of $\mathbf{x}$.

compare it with GPSR. In our experiments, we fix the matrix size (1024×4096) and consider a range of degrees of sparseness: the number m of nonzeros spikes in $\mathbf{x}$ (randomly located values of ± 1) ranges from 5 to 250. For each value of m we generate ten random data sets, i.e., triplets $(\mathbf{x}, \mathbf{y}, \mathbf{R})$. For each data set, we first run GPSR-BB (the monotone version) and store the final value of the residual; we then run `greed_omp_qr` and `SolveOMP` until they reach the same residual norm. Finally, we compute average MSE (with respect to the true $\mathbf{x}$) and average CPU time, over the ten runs.

Fig. 4 plots the average reconstruction MSE and the average CPU times, as a function of the number of nonzero components in $\mathbf{x}$. We observe that all methods basically obtain exact reconstructions for m up to almost 200, with the OMP solutions (which are equal up to some numerical fluctuations) starting to degrade earlier and faster than those produced by solving (1). Concerning computational efficiency, our main focus in this experiment, we can observed that GPSR-BB is clearly faster

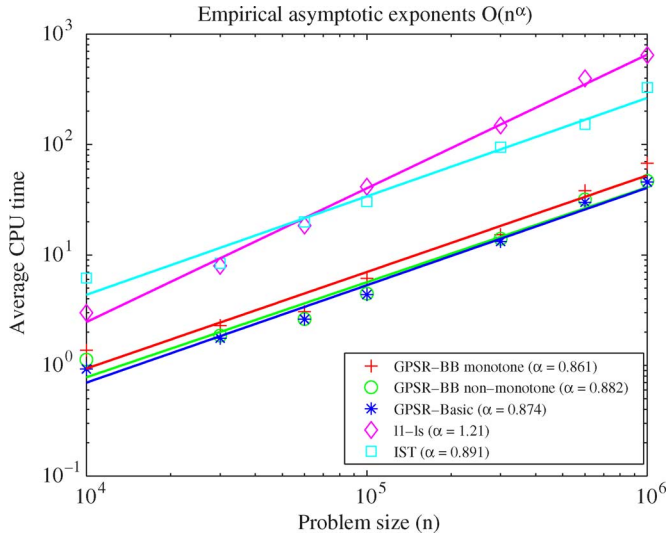

Fig. 5. Assessment of the empirical growth exponent of the computational complexity of several algorithms.

than both OMP implementations, except in the case of extreme sparseness ($m < 50$ nonzero elements in the 4096-vector $\mathbf{x}$).

C. Scalability Assessment

To assess how the computational cost of the GPSR algorithms grows with the size of matrix $\mathbf{A}$, we have performed an experiment similar to the one in [36, Sec. 5.3]. The idea is to assume that the computational cost is $O(n^\alpha)$ and obtain empirical estimates of the exponent α. We consider random sparse matrices (with the nonzero entries normally distributed) of dimensions $(0.1n) \times n$, with n ranging from 10^4 to 10^6. Each matrix is generated with about $3n$ nonzero elements and the original signal with $n/4$ randomly placed nonzero components. For each value of n, we generate ten random matrices and original signals and observed data according to (2), with noise variance $\sigma^2 = 10^{-4}$. For each data set (i.e., each pair $\mathbf{A}, \mathbf{y}$), τ is chosen as in (22). The results in Fig. 5 (which are average for ten data sets of each size) show that all GPSR algorithms have empirical exponents below 0.9, thus much better than *l1_ls* (for which we found $\alpha = 1.21$, in agreement with the value 1.2 reported in [36]); IST has an exponent very close to that of GPSR algorithms, but a worse constant, thus its computational complexity is approximately a constant factor above GPSR. Finally, notice that, according to [36], ℓ_1-*magic* has an exponent close to 1.3, while all the other methods considered in that paper have exponents no less than 2.

D. Warm Starting and Continuation

As mentioned in Section III-F, GPSR algorithms can benefit from being warm-started, that is, initialized at a point close to the solution. This property can be exploited to find minima (1) for a sequence of values of τ, at a modest multiple of the cost of solving only for one value of τ. We illustrate this possibility in a problem with $k = 1024$, $n = 8192$, which we wish to solve for nine different values of τ

$$\tau \in \{0.05, 0.075, 0.1, \ldots, 0.275\} \|\mathbf{A}^T \mathbf{y}\|_\infty.$$

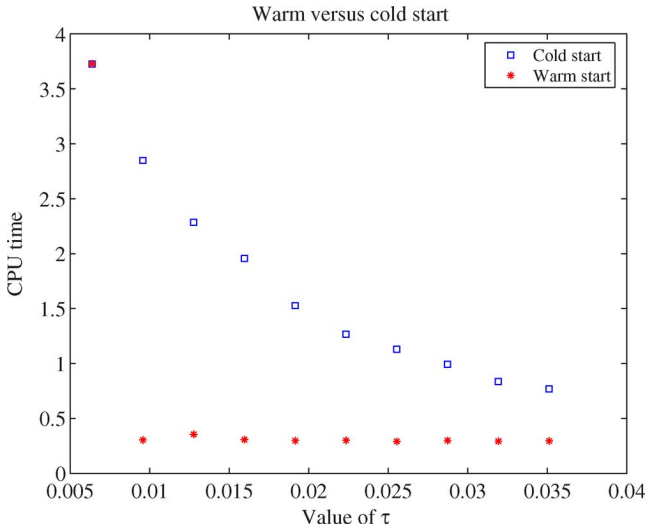

Fig. 6. CPU times for a sequence of values of τ with warm starting and without warm starting (cold starting).

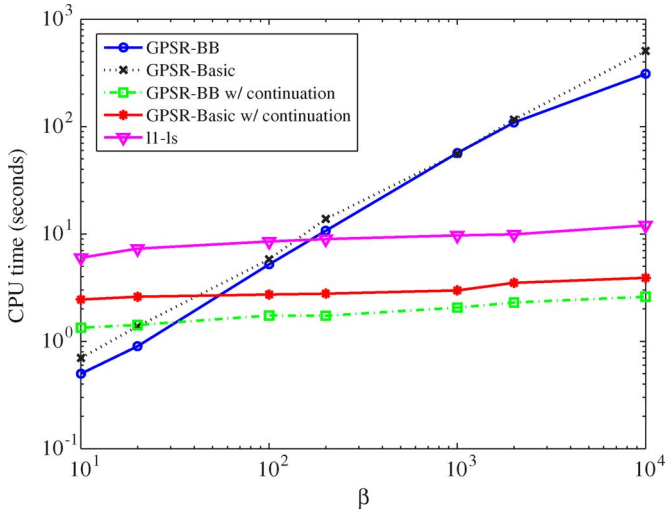

Fig. 7. CPU times of the GPSR-BB and GPSR-Basic algorithms, with and without continuation, as a function of $\beta = \|\mathbf{A}^T\mathbf{y}\|_\infty/\tau$.

As shown in Fig. 6, warm starting does indeed significantly reduce the CPU time required by GPSR. The total CPU time of the nine runs was about 6.5 s, less than twice that of the first run, which is about 3.7 s. The total time required using a cold-start for each value of τ was about 17.5 s.

It has been pointed out recently that the speed of GPSR can degrade as the value of τ becomes small [31]. (This observation is confirmed by the CPU times of the cold-started runs of GPSR shown in Fig. 6.) The GPSR approaches can be improved by adding a continuation heuristic, as suggested in [31] and explained in Section III-F. Our simple heuristic starts by setting $\tau = 0.8\|\mathbf{A}^T\mathbf{y}\|_\infty$, then decreases in τ by a constant factor in five steps until the desired value of τ is obtained. GPSR is run from a "cold start" at the first (largest) value of τ, then run from a warm start for each of the other five values in the sequence.

We illustrate the performance of this heuristic by using the same test problem as in Section IV-A, but with $\sigma^2 = 0$. In this noiseless case, CS theory states that it is possible to reconstruct $\mathbf{x}$ accurately by solving (5). Since the solution of (1) approaches that of (5), as τ goes to zero, it makes sense for this problem to work with small values of τ.

Fig. 7 shows the average (over ten runs) of the CPU times required by GPSR-BB and GPSR-Basic with and without continuation, as well as $l1\_ls$, for several values of β, where we define $\beta = \|\mathbf{A}^T\mathbf{y}\|_\infty/\tau$. Although the original versions of the GPSR algorithms are slower than $l1\_ls$, for β sufficiently large (τ sufficiently small), the continuation schemes are faster than $l1\_ls$ for the whole range of values.

E. Image Deconvolution Experiments

In this subsection, we illustrate the use of the GPSR-BB algorithm in image deconvolution. Recall that (see Section I-A) wavelet-based image deconvolution, under a Laplacian prior on the wavelet coefficients, can be formulated as (1). We stress that the goal of these experiments is not to assess the performance (e.g., in terms of SNR improvement) of the criterion form (1). Such an assessment has been comprehensively carried out in [25], [26], and several other recent works on this topic. Rather, our goal is to compare the speed of the proposed GPSR algorithms against the competing IST.

TABLE II
IMAGE DECONVOLUTION EXPERIMENTS

Experiment	blur kernel	σ^2
1	9×9 uniform	0.56^2
2	$h_{ij} = 1/(i^2 + j^2)$	2
3	$h_{ij} = 1/(i^2 + j^2)$	8

TABLE III
CPU TIMES (IN SECONDS) FOR THE IMAGE DECONVOLUTION EXPERIMENTS

Experiment	GPSR-BB monotone	GPSR-BB nonmonotone	IST
1	1.69	1.04	3.82
2	1.11	0.84	2.63
3	1.21	1.01	2.38

We consider three standard benchmark problems summarized in Table II, all based on the well-known Cameraman image; these problems have been studied in [25], [26] (and other papers). In this experiments, $\mathbf{W}$ represents the inverse orthogonal wavelet transform, with Haar wavelets, and $\mathbf{R}$ is a matrix representation of the blur operation; we have $k = n = 256^2$, and the difficulty comes not form the indeterminacy of the associated system, but from the very ill-conditioned nature of matrix $\mathbf{RW}$. Parameter τ is hand-tuned for the best SNR improvement. In each case, we first run IST and then run the GPSR algorithms until they reach the same final value of the objective function; the final values of MSE are essentially the same. Table III lists the CPU times required by GPSR-BB algorithms and IST, in each of these experiments, showing that GPSR-BB is two to three times faster than IST.

V. CONCLUSIONS

We have proposed gradient projection algorithms for solving a quadratic programming reformulation of a class of convex nonsmooth unconstrained optimization problems arising in

compressed sensing and other inverse problems in signal processing and statistics. In experimental comparisons to state-of-the-art algorithms, the proposed methods are significantly faster (in some cases by orders of magnitude), especially in large-scale settings. Instances of poor performance have been observed when the regularization parameter τ is small, but in such cases the gradient projection methods can be embedded in a simple continuation heuristic to recover their efficient practical performance. The new algorithms are easy to implement, work well across a large range of applications, and do not appear to require application-specific tuning. Our experiments also evidenced the importance of a *debiasing* phase, in which we use a linear CG method to minimize the least squares cost of the inverse problem, under the constraint that the zero components of the sparse estimate produced by the GP algorithm remain at zero.

MATLAB implementations of the algorithms discussed in this paper are available at http://www.lx.it.pt/~mtf/GPSR.

REFERENCES

[1] S. Alliney and S. Ruzinsky, "An algorithm for the minimization of mixed ℓ_1 and ℓ_2 norms with application to Bayesian estimation," *IEEE Trans. Signal Processing*, vol. 42, pp. 618–627, Mar. 1994.

[2] J. Barzilai and J. Borwein, "Two point step size gradient methods," *IMA J. Numer. Anal.*, vol. 8, pp. 141–148, 1988.

[3] D. P. Bertsekas, *Nonlinear Programming*, 2nd ed. Boston, MA: Athena, 1999.

[4] E. G. Birgin, J. M. Martinez, and M. Raydan, "Nonmonotone spectral projected gradient methods on convex sets," *SIAM J. Optim.*, vol. 10, pp. 1196–1211, 2000.

[5] T. Blumensath and M. Davies, Gradient Pursuits 2007 [Online]. Available: http://www.see.ed.ac.uk/~tblumens/papers

[6] E. Candès, J. Romberg, and T. Tao, "Stable signal recovery from incomplete and inaccurate information," *Commun. Pure Appl. Math.*, vol. 59, pp. 1207–1233, 2005.

[7] E. Candès and T. Tao, "Near optimal signal recovery from random projections: Universal encoding strategies," *IEEE Trans. Inform. Theory*, vol. 52, pp. 5406–5425, Dec. 2006.

[8] E. Candès and T. Tao, "The Dantzig selector: Statistical estimation when p is much larger than n," *Ann. Statist.* 2007 [Online]. Available: http://arxiv.org/abs/math.ST/0506081

[9] E. Candès, J. Romberg, and T. Tao, "Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information," *IEEE Trans. Inform. Theory*, vol. 52, pp. 489–509, Feb. 2006.

[10] A. Chambolle, "An algorithm for total variation minimization and applications," *J. Math. Imag. Vis.*, vol. 20, pp. 89–97, 2004.

[11] S. Chen, D. Donoho, and M. Saunders, "Atomic decomposition by basis pursuit," *SIAM J. Sci. Comput.*, vol. 20, pp. 33–61, 1998.

[12] J. Claerbout and F. Muir, "Robust modelling of erratic data," *Geophys.*, vol. 38, pp. 826–844, 1973.

[13] P. Combettes and V. Wajs, "Signal recovery by proximal forward-backward splitting," *SIAM J. Multiscale Model. Simul.*, vol. 4, pp. 1168–1200, 2005.

[14] R. W. Cottle, J.-S. Pang, and R. E. Stone, *The Linear Complementarity Problem*. New York: Academic, 1993.

[15] Y.-H. Dai and R. Fletcher, "Projected Barzilai–Borwein methods for large-scale box-constrained quadratic programming," *Numerische Mathematik*, vol. 100, pp. 21–47, 2005.

[16] I. Daubechies, M. De Friese, and C. De Mol, "An iterative thresholding algorithm for linear inverse problems with a sparsity constraint," *Commun. Pure Appl. Math.*, vol. 57, pp. 1413–1457, 2004.

[17] G. Davis, S. Mallat, and M. Avellaneda, "Greedy adaptive approximation," *J. Construct. Approx.*, vol. 12, pp. 57–98, 1997.

[18] D. Donoho, "Compressed sensing," *IEEE Trans. Inform. Theory*, vol. 52, pp. 1289–1306, Apr. 2006.

[19] D. Donoho, "De-noising by soft-thresholding," *IEEE Trans. Inform. Theory*, vol. 41, pp. 613–627, May 1995.

[20] D. Donoho, M. Elad, and V. Temlyakov, "Stable recovery of sparse overcomplete representations in the presence of noise," *IEEE Trans. Inform. Theory*, vol. 52, pp. 6–18, Jan. 2006.

[21] D. Donoho and Y. Tsaig, Fast Solution of L1-Norm Minimization Problems When the Solution May be Sparse Inst. Comput. Math. Eng., Stanford Univ., Stanford, CA, 2006 [Online]. Available: http://www.stanford.edu/~tsaig, Tech. Rep.

[22] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani, "Least angle regression," *Ann. Statist.*, vol. 32, pp. 407–499, 2004.

[23] M. Elad, "Why simple shrinkage is still relevant for redundant representations?," *IEEE Trans. Inform. Theory*, vol. 52, pp. 5559–5569, Dec. 2006.

[24] M. Elad, B. Matalon, and M. Zibulevsky, "Image denoising with shrinkage and redundant representations," in *Proc. IEEE Computer Soc. Conf. Computer Vision and Pattern Recognition—CVPR'2006*, New York, 2006.

[25] M. Figueiredo and R. Nowak, "An EM algorithm for wavelet-based image restoration," *IEEE Trans. Image Processing*, vol. 12, pp. 906–916, Aug. 2003.

[26] M. Figueiredo and R. Nowak, "A bound optimization approach to wavelet-based image deconvolution," in *Proc. IEEE Int. Conf. Image Processing—ICIP'2005*, 2005.

[27] M. Frank and P. Wolfe, "An algorithm for quadratic programming," *Naval Res. Logist. Quar.*, vol. 3, pp. 95–110, 1956.

[28] J. J. Fuchs, "Multipath time-delay detection and estimation," *IEEE Trans. Signal Processing*, vol. 47, pp. 237–243, Jan. 1999.

[29] J. J. Fuchs, "More on sparse representations in arbitrary bases," *IEEE Trans. Inform. Theory*, vol. 50, pp. 1341–1344, 2004.

[30] J. Gondzio and A. Grothey, A New Unblocking Technique to Warm-start Interior Point Methods Based on Sensitivity Analysis School Math., Univ. Edinburgh, Edinburgh, U.K., Tech. Rep. MS-06-005, 2006.

[31] E. Hale, W. Yin, and Y. Zhang, A Fixed-Point Continuation Method for ℓ_1-Regularized Minimization With Applications to Compresses Sensing Dept. Computational and Applied Mathematics, Rice Univ., Houston, TX, Tech. Rep. TR07-07, 2007.

[32] J. Haupt and R. Nowak, "Signal reconstruction from noisy random projections," *IEEE Trans. Inform. Theory*, vol. 52, pp. 4036–4048, Sep. 2006.

[33] D. Hunter and K. Lange, "A tutorial on MM algorithms," *Amer. Statistician*, vol. 58, pp. 30–37, 2004.

[34] A. N. Iusem, "On the convergence properties of the projected gradient method for convex optimization," *Comput. Appl. Math.*, vol. 22, pp. 37–52, 2003.

[35] E. John and E. A. Yıldırım, Implementation of Warm-Start Strategies in Interior-Point Methods for Linear Programming in Fixed Dimension Dept. Ind. Eng., Bilkent Univ., Ankara, Turkey, 2006.

[36] S. Kim, K. Koh, M. Lustig, S. Boyd, and D. Gorinvesky, A Method for Large-Scale ℓ_1-Regularized Least Squares Problems With Applications in Signal Processing and Statistics Dept. Elect. Eng., Stanford Univ., 2007 [Online]. Available: http://www.stanford.edu/~boyd/l1_ls.html

[37] S. Levy and P. Fullagar, "Reconstruction of a sparse spike train from a portion of its spectrum and application to high-resolution deconvolution," *Geophys.*, vol. 46, pp. 1235–1243, 1981.

[38] M. Lustig, D. Donoho, and J. Pauly, Sparse MRI: The Application of Compressed Sensing for Rapid MR Imaging 2007 [Online]. Available: http://www.stanford.edu/~mlustig/SparseMRI.pdf

[39] D. Malioutov, M. Çetin, and A. Willsky, "Homotopy continuation for sparse signal representation," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, Philadelphia, PA, 2005, vol. 5, pp. 733–736.

[40] S. Mallat, *A Wavelet Tour of Signal Processing*. San Diego, CA: Academic, 1998.

[41] A. Miller, *Subset Selection in Regression*. London, U.K.: Chapman & Hall, 2002.

[42] B. Moghaddam, Y. Weiss, and S. Avidan, "Spectral bounds for sparse PCA: Exact and greedy algorithms," in *Advances in Neural Information Processing Systems 18*. Cambridge, MA: MIT Press, 2006, pp. 915–922.

[43] J. Moré and G. Toraldo, "On the solution of large quadratic programming problems with bound constraints," *SIAM J. Optim.*, vol. 1, pp. 93–113, 1991.

[44] J. Nocedal and S. J. Wright, *Numerical Optimization*, 2nd ed. New York: Springer-Verlag, 2006.

[45] R. Nowak and M. Figueiredo, "Fast wavelet-based image deconvolution using the EM algorithm," in *Proc. 35th Asilomar Conf. Signals, Systems, Computers*, Monterey, CA, 2001.

[46] M. Osborne, B. Presnell, and B. Turlach, "A new approach to variable selection in least squares problems," *IMA J. Numer. Anal.*, vol. 20, pp. 389–403, 2000.

[47] S. Osher, L. Rudin, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Phys. D.*, vol. 60, pp. 259–268, 1992.

[48] C. Paige and M. A. Saunders, "LSQR: An algorithm for sparse linear equations and sparse least squares," *ACM Trans. Math. Softw.*, vol. 8, pp. 43–71, 1982.

[49] R. T. Rockafellar, *Convex Analysis*. Princeton, NJ: Princeton Univ. Press, 1970.

[50] F. Santosa and W. Symes, "Linear inversion of band-limited reflection histograms," *SIAM J. Sci. Statist. Comput.*, vol. 7, pp. 1307–1330, 1986.
[51] M. A. Saunders, PDCO: Primal-Dual Interior-Point Method for Convex Objectives Systems Optimization Laboratory, Stanford Univ.. Stanford, CA, 2002 [Online]. Available: http://www.stanford.edu/group/SOL/
[52] T. Serafini, G. Zanghirati, and L. Zanni, "Gradient projection methods for large quadratic programs and applications in training support vector machines," *Optim. Meth. Softw.*, vol. 20, pp. 353–378, 2004.
[53] H. Taylor, S. Bank, and J. McCoy, "Deconvolution with the ℓ_1 norm," *Geophys.*, vol. 44, pp. 39–52, 1979.
[54] R. Tibshirani, "Regression shrinkage and selection via the lasso," *J. R. Statist. Soc. B*, vol. 58, pp. 267–288, 1996.
[55] J. Tropp, "Just relax: Convex programming methods for identifying sparse signals," *IEEE Trans. Inform. Theory*, vol. 51, pp. 1030–1051, 2006.
[56] J. Tropp, "Greed is good: Algorithmic results for sparse approximation," *IEEE Trans. Inform. Theory*, vol. 50, pp. 2231–2242, Oct. 2004.
[57] B. Turlach, "On algorithms for solving least squares problems under an L_1 penalty or an L_1 constraint," in *Proc. American Statistical Association; Statistical Computing Section*, Alexandria, VA, 2005, pp. 2572–2577.
[58] B. Turlach, W. N. Venables, and S. J. Wright, "Simultaneous variable selection," *Technometrics*, vol. 27, pp. 349–363, 2005.
[59] S. J. Wright, "Implementing proximal point methods for linear programming," *J. Optim. Theory Applicat.*, vol. 65, pp. 531–554, 1990.
[60] S. J. Wright, *Primal-Dual Interior-Point Methods*. Philadelphia, PA: SIAM, 1997.
[61] E. A. Yildirim and S. J. Wright, "Warm-start strategies in interior-point methods for linear programming," *SIAM J. Optim.*, vol. 12, pp. 782–810, 2002.

Mário A. T. Figueiredo (S'87–M'95–SM'00) received the E.E., M.Sc., Ph.D., and "Agregado" degrees in electrical and computer engineering, all from Instituto Superior Técnico (IST), the engineering school of the Technical University of Lisbon, Lisbon, Portugal, in 1985, 1990, 1994, and 2004, respectively.

Since 1994, he has been with the faculty of the Department of Electrical and Computer Engineering, IST. He is also a Researcher and Area Coordinator, Institute of Telecommunications, Lisbon. His scientific interests include image processing and analysis, computer vision, statistical pattern recognition, and statistical learning.

Dr. Figueiredo received the Portuguese IBM Scientific Prize in 1995 for work on unsupervised image restoration. He is a member of the IEEE Image and Multidimensional Signal Processing Technical Committee. He is/was Associate Editor of the IEEE TRANSACTIONS ON IMAGE PROCESSING, IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, and IEEE TRANSACTIONS ON MOBILE COMPUTING, PATTERN RECOGNITION LETTERS, AND SIGNAL PROCESSING. He was guest co-editor of special issues of the IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE and IEEE TRANSACTIONS ON SIGNAL PROCESSING. He was a co-chair of the 2001 and 2003 Workshops on Energy Minimization Methods in Computer Vision and Pattern Recognition. He has been a member of program committees of several international conferences, including CVPR, ECCV, ICIAR, ICASSP, ICIP, ICML, ICPR, MLSP, and NIPS.

Robert D. Nowak (S'90–M'95–SM'04) received the B.S. (with highest distinction), M.S., and Ph.D. degrees in electrical engineering from the University of Wisconsin-Madison in 1990, 1992, and 1995, respectively.

He was a Postdoctoral Fellow at Rice University, Houston, TX, in 1995–1996, an Assistant Professor at Michigan State University, East Lansing, from 1996 to 1999, held Assistant and Associate Professor positions at Rice University from 1999–2003, and was a Visiting Professor at INRIA in 2001. He is now the McFarland-Bascom Professor of Engineering at the University of Wisconsin-Madison. His research interests include statistical signal processing, machine learning, imaging and network science, and applications in communications, bio/medical imaging, and genomics.

Dr. Nowak has served as an Associate Editor for the IEEE TRANSACTIONS ON IMAGE PROCESSING, and is currently an Associate Editor for the ACM Transactions on Sensor Networks and the Secretary of the SIAM Activity Group on Imaging Science. He has also served as a Technical Program Chair for the IEEE Statistical Signal Processing Workshop and the IEEE/ACM International Symposium on Information Processing in Sensor Networks. He received the General Electric Genius of Invention Award in 1993, the National Science Foundation CAREER Award in 1997, the Army Research Office Young Investigator Program Award in 1999, the Office of Naval Research Young Investigator Program Award in 2000, and IEEE Signal Processing Society Young Author Best Paper Award in 2000.

Stephen J. Wright received the B. Sc. (Hons.) and Ph.D. degrees from the University of Queensland, Australia, in 1981 and 1984, respectively.

After holding positions at North Carolina State University, Argonne National Laboratory, and the University of Chicago, he joined the Computer Sciences Department, University of Wisconsin-Madison, as a Professor in 2001. His research interests include theory, algorithms, and applications of computational optimization.

Dr. Wright is Chair of the Mathematical Programming Society and has served on the editorial boards of *Mathematical Programming, Series A and B*, the *SIAM Journal on Optimization*, the *SIAM Journal on Scientific Computing*, and other journals. He also serves on the Board of Trustees of SIAM.