TEACHING A NEW TRICK TO AN OLD DOG: REVISITING THE QUADRATIC PROGRAMMING FORMULATION OF SPARSE RECOVERY USING ADMM

Mário A. T. Figueiredo

Instituto de Telecomunicações, Instituto Superior Técnico, Universidade de Lisboa, Portugal

ABSTRACT

One of the early successful approaches to deal with the now classical $\ell_2 + \ell_1$ optimization formulation of sparse signal recovery (often known as the LASSO) was based on re-writing it as a bound-constrained quadratic program (BCQP), which was then tackled using a gradient projection (GP) algorithm with a spectral (Barzilai-Borwein) step choice. The resulting algorithm (called *gradient projection for sparse reconstruction* – GPSR) exhibited state-of-the-art speed when it was introduced, but now, 6 years later, much faster alternatives exist. In this paper, we revisit the BCQP formulation and show how it can be efficiently dealt with using the *alternating direction method of multipliers* (ADMM). We give preliminary experimental evidence that this approach is competitive with the current state-of-the-art, in a set of benchmark problems.

Index Terms— Sparse signal recovery, convex optimization, alternating direction optimization, deconvolution, inpainting.

1. INTRODUCTION

This paper revisits the now very classical $\ell_2 + \ell_1$ problem,

$$\min_{\mathbf{x}\in\mathbb{R}^n} \; \frac{1}{2}\|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2^2 + \tau\|\mathbf{x}\|_1, \tag{1}$$

where $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{y} \in \mathbb{R}^m$, $\mathbf{A}$ is an $m \times n$ matrix (typically with $m \leq n$), 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 $\mathbf{v}$. This formulation is often used to identify sparse approximate solutions to the underdetermined system $\mathbf{y} = \mathbf{A}\mathbf{x}$, and has become very familiar in the past few decades, particularly in statistics, machine learning, and signal/image processing. For brief historical accounts on the use of the ℓ_1 penalty in statistics and signal processing, see [21], [28].

Problems with the form (1) arise in wavelet-based image/signal reconstruction/restoration [16], [19], [20], [21]. In these problems, matrix $\mathbf{A}$ usually has the form $\mathbf{A} = \mathbf{B}\mathbf{W}$, where $\mathbf{B}$ is (the matrix representing) the observation operator (*e.g.*, a convolution with a blur kernel, a tomographic projection, loss of samples), $\mathbf{W}$ contains a wavelet basis or redundant dictionary (*i.e.*, 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. The interest in problems of the form (1) was greatly amplified in the past decade due to the central role that they play in the theory and practice of *compressed* (or *compressive*) *sensing* (CS) [11], [14].

Of particular interest in signal processing contexts are problems of the form (1), where matrix $\mathbf{A}$ is too large (and too dense) to be handled explicitly, but it is still possible to compute matrix-vector products involving $\mathbf{A}$ (or its transpose) efficiently. In other words, matrix $\mathbf{A}$ represents an operator (or a composition of several operators) for which there is an efficient algorithm (*e.g.*, a convolution with some kernel or some transform). This limitation precludes the use of many off-the-shelf convex optimization algorithms and has stimulated much research work devoted to finding efficient methods for solving (1) is this type of scenario. Arguably, the standard algorithm for solving (1) is the so-called *iterative shrinkage/thresholding* (IST) algorithm (a.k.a. *proximal-gradient*) [19], [13], [12], which has the well-known drawback of being very slow when $\mathbf{A}$ is poorly conditioned. To address this issue, several accelerated versions of IST have been proposed, namely by using two-step-type iterations [5], [7], or spectral (Barzilai-Borwein – BB [4]) step-size selection techniques [29].

Another class of algorithms that has seen a recent explosion of interest to solve problems of the form (1) is based on *augmented Lagrangian* formulations, in particular, the *alternating direction method of multiplers* (ADMM) (see [10], for an introduction and comprehensive review). In fact, for several problems of the form (1), arising in sparsity-based recovery (such as signal/image deconvolution, reconstruction from partial Fourier observations [26], image inpainting, and others), the state-of-the-art algorithms are based on ADMM [1], [2]. Intuitively, the excellent speed of these methods can be traced to the fact that in each iteration they require solving a linear system where the matrix is a regularized version of the Hessian of the smooth term of (1), thus having a "Newton flavour", *i.e.*, using second-order information. In fact, there has been great recent interest in deriving and studying proximal-type algorithms that make use, not only of first order information (gradient), but also of second order information (curvature/Hessian); these methods are referred to as *proximal-Newton* [6], [25], [27].

One of the early approaches to deal with problem (1) started by re-writing it as a bound-constrained quadratic problem (BCQP) [21], [22], which was then addressed via a gradient projection algorithm with a BB step-size choice. That method was proposed under the name *gradient projection for sparse reconstruction* (GPSR) [21] and was a state-of-the-art algorithm when it appeared. The goal of this paper is to investigate the use of the ADMM algorithm in dealing with the BCQP formulation of (1). As in ADMM applied directly to the $\ell_2 + \ell_1$ [1], the resulting algorithm involves a matrix inversion; in the BCQP formulation, this matrix is in fact a regularized version of the Hessian of the whole objective, since the non-smooth ℓ_1 was transformed into a bound-constraint. For this reason, the resulting algorithm has some similarity with a projected regularized Newton algorithm [24]; this connection deserves further study, which is left for future work. We are specially interested in cases where the matrix inversion mentioned above can be efficiently carried out, even when the parameter of the ADMM algorithm changes along the iterations.

The rest of the paper is organized as follows. Section 2 reviews the BCQP re-formulation of (1) and Section 3 reviews the ADMM.

This work was partially supported by *Fundação para a Ciência e Tecnologia* (Portuguese Ministry of Education and Science), grants PEst-OE/EEI/LA0008/2013 and PTDC/EEI-PRO/1470/2012.

978-1-4799-2893-4/14/$31.00 ©2014 IEEE

The application of ADMM to the BCQP is presented in Section 4, where several implementation aspects are discussed. The instantiation to imaging inverse problems is the topic of Section 5. Section 6 reports experimental results and Section 7 concludes the paper.

Note: we use the notation $\mathbf{M}^*$ to represent the conjugate transpose of a matrix $\mathbf{A}$, which coincides with the transpose, if the matrix has only real-valued elements.

2. THE BCQP FORMULATION

To express (1) as a BCQP [20], the variable $\mathbf{x}$ is split into its positive and negative parts, denoted $\mathbf{u}$ and $\mathbf{v}$, that is,

$$\mathbf{x} = \mathbf{u} - \mathbf{v}, \;\; \mathbf{u} \geq 0, \; \mathbf{v} \geq 0, \tag{2}$$

where $u_i = (x_i)_+$ and $v_i = (-x_i)_+$, for all $i = 1, 2, \ldots, n$, with $(\cdot)_+$ denoting the *positive-part operator* defined as $(x)_+ = \max\{0, x\}$. We thus have $\|\mathbf{x}\|_1 = \mathbf{1}_n^* \mathbf{u} + \mathbf{1}_n^* \mathbf{v}$, where $\mathbf{1}_n = [1, 1, \ldots, 1]^*$, and (1) can be written as

$$\begin{aligned} \min_{\mathbf{u},\mathbf{v}} \quad & \frac{1}{2} \|\mathbf{y} - \mathbf{A}(\mathbf{u} - \mathbf{v})\|_2^2 + \tau \, \mathbf{1}_n^* \mathbf{u} + \tau \, \mathbf{1}_n^* \mathbf{v}, \\ \text{s.t.} \quad & \mathbf{u} \geq 0, \;\; \mathbf{v} \geq 0. \end{aligned} \tag{3}$$

Note that the ℓ_2-norm term is unaffected by adding the same $\mathbf{s} \geq 0$ to both $\mathbf{u}$ and $\mathbf{v}$. However such a shift increases the other terms by $2\,\tau\,\mathbf{1}_n^* \mathbf{s} \geq 0$; consequently, at the solution of (3), $u_i = 0$ or $v_i = 0$, so that in fact $u_i = (x_i)_+$ and $v_i = (-x_i)_+$, for all $i = 1, 2, \ldots, n$. Problem (3) can be written in standard BCQP form,

$$\begin{aligned} \min_{\mathbf{z} \in \mathbb{R}^{2n}} \quad & \mathbf{c}^* \mathbf{z} + \frac{1}{2} \, \mathbf{z}^* \mathbf{B} \mathbf{z} \equiv F(\mathbf{z}), \\ \text{s.t.} \quad & \mathbf{z} \geq 0, \end{aligned} \tag{4}$$

where: $\mathbf{z} = [\mathbf{u}^*, \, \mathbf{v}^*]^*$; $\mathbf{c} = \tau \, \mathbf{1}_{2n} + [-\mathbf{b}^*, \, \mathbf{b}^*]^*$; $\mathbf{b} = \mathbf{A}^* \mathbf{y}$;

$$\mathbf{B} = \begin{bmatrix} \mathbf{A}^*\mathbf{A} & -\mathbf{A}^*\mathbf{A} \\ -\mathbf{A}^*\mathbf{A} & \mathbf{A}^*\mathbf{A} \end{bmatrix} = \mathbf{C}^* \mathbf{C}; \tag{5}$$

and $\mathbf{C} = [\mathbf{A}, \, -\mathbf{A}]$.

Although the dimension of problem (4) is twice that of (1), this increase in dimension has a minor impact. Matrix operations involving $\mathbf{B}$ can be performed more economically than its size suggests, by exploiting its structure (5). For a given $\mathbf{z} = [\mathbf{u}^* \; \mathbf{v}^*]^*$,

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

indicating that $\mathbf{B}\mathbf{z}$ can be found by computing the vector difference $\mathbf{u} - \mathbf{v}$ and then multiplying by $\mathbf{A}$ and $\mathbf{A}^*$. Since $\nabla F(\mathbf{z}) = \mathbf{c} + \mathbf{B}\mathbf{z}$ (the gradient of the objective function in (4)), computing $\nabla F(\mathbf{z})$ requires one multiplication by $\mathbf{A}$ and $\mathbf{A}^*$; notice that $\mathbf{c}$, which depends on $\mathbf{b} = \mathbf{A}^* \mathbf{y}$, is constant and can be pre-computed at the start of any algorithm. Another common operation is to compute the objective at a given $\mathbf{z} = [\mathbf{u}^*, \, \mathbf{v}^*]^*$, which requires computing $\mathbf{z}^* \mathbf{B} \mathbf{z}$. Since $\mathbf{z}^* \mathbf{B} \mathbf{z} = (\mathbf{u} - \mathbf{v})^* \mathbf{A}^* \mathbf{A} (\mathbf{u} - \mathbf{v}) = \|\mathbf{A}(\mathbf{u} - \mathbf{v})\|_2^2$, this can be obtained using a single multiplication by $\mathbf{A}$.

3. THE ADMM

Consider the unconstrained optimization problem

$$\min_{\mathbf{z} \in \mathbb{R}^d} \; f_1(\mathbf{z}) + f_2(\mathbf{G}\,\mathbf{z}), \tag{6}$$

where $f_1 : \mathbb{R}^d \to \bar{\mathbb{R}} \equiv \mathbb{R} \cup \{-\infty, \infty\}$, $f_2 : \mathbb{R}^p \to \bar{\mathbb{R}}$, and $\mathbf{G} \in \mathbb{R}^{p \times d}$. The ADMM for this problem is shown in Fig. 1; its convergence (in fact, of a generalized version thereof) was shown in a seminal paper by Eckstein and Bertsekas [15], under quite weak conditions: $\mathbf{G} \in \mathbb{R}^{p \times d}$ has full column rank; $f_1 : \mathbb{R}^d \to \bar{\mathbb{R}}$ and $f_2 : \mathbb{R}^p \to \bar{\mathbb{R}}$ are closed, proper, convex functions. A recent and comprehensive review of ADMM can be found in [10].

Algorithm *ADMM*

1. Set $k = 0$, choose $\mu > 0$, $\mathbf{u}_0$, and $\mathbf{d}_0$.
2. **repeat**
3. $\quad \mathbf{z}_{k+1} \in \arg\min_{\mathbf{z}} f_1(\mathbf{z}) + \frac{\mu}{2} \|\mathbf{G}\,\mathbf{z} - \mathbf{u}_k - \mathbf{d}_k\|_2^2$
4. $\quad \mathbf{u}_{k+1} \in \arg\min_{\mathbf{u}} f_2(\mathbf{u}) + \frac{\mu}{2} \|\mathbf{G}\,\mathbf{z}_{k+1} - \mathbf{u} - \mathbf{d}_k\|_2^2$
5. $\quad \mathbf{d}_{k+1} \leftarrow \mathbf{d}_k - (\mathbf{G}\,\mathbf{z}_{k+1} - \mathbf{u}_{k+1})$
6. $\quad k \leftarrow k + 1$
7. **until** stopping criterion is satisfied.

Fig. 1. The ADMM algorithm.

Given a convex function g, its so-called *Moreau proximity operator* (MPO) [12], denoted prox_g, is uniquely defined as

$$\text{prox}_g(\mathbf{s}) \equiv \arg\min_{\mathbf{x}} \frac{1}{2} \|\mathbf{x} - \mathbf{s}\|_2^2 + g(\mathbf{x}). \tag{7}$$

It is thus clear that line 4 of ADMM can be written as

$$\mathbf{u}_{k+1} = \text{prox}_{f_2/\mu}(\mathbf{G}\,\mathbf{z}_{k+1} - \mathbf{d}_k).$$

For several functions, the corresponding MPO can be computed exactly in closed form [12]; *e.g.*, the well-known *soft-thresholding* function is the MPO of the ℓ_1 norm. Of particular interest in this paper is the indicator of a convex set $\mathcal{C}$, $\iota_{\mathcal{C}}(\mathbf{x}) = 0$, if $\mathbf{x} \in \mathcal{C}$ and $\iota_{\mathcal{C}}(\mathbf{x}) = +\infty$, if $\mathbf{x} \notin \mathcal{C}$. In this case, $\text{prox}_{\iota_{\mathcal{C}}} = \mathcal{P}_{\mathcal{C}}$ is simply the Euclidean projector onto $\mathcal{C}$ [12]. Due to the presence of matrix $\mathbf{G}$ in the quadratic term of the optimization problem that defines line 3 of ADMM, this step is not (in general) an MPO.

When applying ADMM, one of the central issues is the choice of parameter μ, which may strongly affect its practical performance [10]. Also in the theoretical analysis front, the effect on convergence speed and the optimal choice of μ are subjects of active research, with results only available for a few particular problems [9], [23]. A practical rule that was proposed in [10] aims at updating μ to maintain a balance between the *primal* and *dual* residuals, which are given by $\mathbf{r}_k = \mathbf{G}\mathbf{z}_k - \mathbf{u}_k$ and $\mathbf{s}_k = \mu \mathbf{G}^*(\mathbf{u}_{k-1} - \mathbf{u}_k)$, respectively; the rule consists in increasing (resp. decreasing) μ by some factor, say γ, when $\|\mathbf{r}_k\|$ becomes larger (resp. smaller) than $\|\mathbf{s}_k\|$ by some factor, say ξ. Finally, replacing vector $\mathbf{G}\,\mathbf{z}_{k+1}$ in lines 4 and 5 of ADMM by $\alpha \mathbf{G}\mathbf{z}_{k+1} + (1 - \alpha)\mathbf{z}_k$, with $0 < \alpha < 2$, leads to over-relaxation ($\alpha > 1$) or under-relaxation ($\alpha < 1$) [9], [10], [15], [23]; using $\alpha \in [1.5, 1.8]$ has been shown to speed up convergence [10].

4. ADMM FOR THE BCQP

To tackle the BCQP (4) using ADMM we re-write it as

$$\min_{\mathbf{z} \in \mathbb{R}^{2n}} \mathbf{c}^* \mathbf{z} + \frac{1}{2} \, \mathbf{z}^* \mathbf{C}^* \mathbf{C} \mathbf{z} + \iota_{\mathbb{R}^{2n}_+}(\mathbf{z}). \tag{8}$$

Of course, there are many ways to map (8) into (6); a naïve observation could suggest using $f_2(\mathbf{a}) = \frac{1}{2}\|\mathbf{a}\|_2^2$ and $\mathbf{G} = \mathbf{C}$, and

consequently $f_1(\mathbf{z}) = \mathbf{c}^*\mathbf{z} + \iota_{\mathbb{R}^{2n}_+}(\mathbf{z})$. However, this choice would lead to line 1 of ADMM (see Fig. 1) which would be itself a BCQP, thus as hard as the original problem. A better choice (see also [10], [23]) is $f_2(\mathbf{a}) = \iota_{\mathbb{R}^{2n}_+}(\mathbf{z})$, $\mathbf{G} = \mathbf{I}$, and $f_1(\mathbf{z}) = \mathbf{c}^*\mathbf{z} + \frac{1}{2}\,\mathbf{z}^*\mathbf{C}^*\mathbf{C}\mathbf{z}$. With this formulation, the resulting ADMM algorithm is as shown in Fig.2, where we have also included the over-relaxation scheme and the adaptation of parameter μ every δ iterations. Of course, the Euclidean projection on the first orthant (in line 4) is simply the component-wise positive part: $\big(\mathcal{P}_{\mathbb{R}^{2n}_+}(\mathbf{z})\big)_i = \max\{0, z_i\}$.

Algorithm *ADMM*

1. Set $k = 0$, choose $\mu_0 > 0$, $\alpha \in (0, 2)$, $\gamma > 1$, $\xi > 1$, $\delta \in \mathbb{N}$, $\mathbf{u}_0$, $\mathbf{d}_0$.
2. **repeat**
3. $\quad \mathbf{z}_{k+1} = -(\mathbf{C}^*\mathbf{C} + \mu_k\mathbf{I})^{-1}\left(\mathbf{c} - \mu_k(\mathbf{u}_k + \mathbf{d}_k)\right)$
4. $\quad \mathbf{u}_{k+1} = \mathcal{P}_{\mathbb{R}^{2n}_+}\left(\alpha\,\mathbf{z}_{k+1} + (1-\alpha)\mathbf{z}_k - \mathbf{d}_k\right)$
5. $\quad \mathbf{d}_{k+1} \leftarrow \mathbf{d}_k - \alpha\,\mathbf{z}_{k+1} - (1-\alpha)\mathbf{z}_k + \mathbf{u}_{k+1}$
6. $\quad$ **if** $\mathrm{mod}(k, \delta) = 0$
7. $\quad\quad r = \|\mathbf{z}_k - \mathbf{u}_{k+1}\|$
8. $\quad\quad s = \mu_k\|\mathbf{u}_k - \mathbf{u}_{k+1}\|$
9. $\quad\quad$ **if** $r > \xi s$ **then** $\mu_k = \gamma\,\mu_k$, $\mathbf{u}_{k+1} = \gamma\,\mathbf{u}_{k+1}$
10. $\quad\quad$ **if** $s > \xi r$ **then** $\mu_k = \mu_k/\gamma$, $\mathbf{u}_{k+1} = \mathbf{u}_{k+1}/\gamma$
11. $\quad k \leftarrow k + 1$
12. **until** stopping criterion is satisfied.

Fig. 2. The ADMM algorithm for the BCQP in (4).

The computational bottleneck of this algorithm is clearly the matrix inversion in line 3, which we will now address. It may seem that the move from an n-dimensional problem (1) to a $2n$-dimensional problem will seriously affect the cost of this inversion, which in general grows (almost) cubically with the matrix dimension. We now show that this is not so, by exploiting the *matrix inversion lemma* (MIL) and the structure of matrix $\mathbf{C} = [\mathbf{A}, -\mathbf{A}]$. Direct application of the MIL yields (dropping the k subscript from μ_k)

$$(\mathbf{C}^*\mathbf{C} + \mu\mathbf{I})^{-1} = \frac{1}{\mu}\Big(\mathbf{I} - \mathbf{C}^*\big(\mathbf{C}\mathbf{C}^* + \mu\mathbf{I}\big)^{-1}\mathbf{C}\Big).$$

Since $\mathbf{C}\mathbf{C}^* = 2\mathbf{A}\mathbf{A}^*$, we further have that, when multiplying this matrix by some vector $\mathbf{z} = [\mathbf{u}^*, \mathbf{v}^*]^*$,

$$\begin{aligned}(\mathbf{C}^*\mathbf{C}+\mu\mathbf{I})^{-1}\mathbf{z} &= \frac{1}{\mu}\left(\mathbf{I} - \begin{bmatrix}\mathbf{A}^*\\ -\mathbf{A}^*\end{bmatrix}\big(2\mathbf{A}\mathbf{A}^*+\mu\mathbf{I}\big)^{-1}[\mathbf{A}, -\mathbf{A}]\right)\begin{bmatrix}\mathbf{u}\\ \mathbf{v}\end{bmatrix}\\ &= \frac{1}{\mu}\begin{bmatrix}\mathbf{u} + \mathbf{A}^*\mathbf{Q}(\mu)\mathbf{A}(\mathbf{u}-\mathbf{v})\\ \mathbf{v} - \mathbf{A}^*\mathbf{Q}(\mu)\mathbf{A}(\mathbf{u}-\mathbf{v})\end{bmatrix},\end{aligned}$$

where $\mathbf{Q}(\mu) = \big(2\mathbf{A}\mathbf{A}^*+\mu\mathbf{I}\big)^{-1}$. This shows that: **(i)** the matrix that needs to be inverted is of dimension $m \times m$, where $m \leq n$, often $m < n$ (*e.g.*, in compressive sensing); **(ii)** the matrix-vector products by matrices $\mathbf{A}$, $\mathbf{A}^*$, and $\mathbf{Q}(\mu)$ are computed only once; **(iii)** all the matrix-vector products involve dimensions no larger than n.

When $\mu_k = \mu$ is fixed, matrix $\mathbf{Q}(\mu)$ can (should) be precomputed and stored; in some cases, rather than computing $\mathbf{Q}(\mu)$, its inverse is factored (*e.g.*, Cholesky) and the corresponding system back-solved at each iteration [10]. If μ_k changes along the algorithm, those strategies are not directly applicable. However, if it is possible to obtain the *singular value decomposition* (SVD) of $\mathbf{A}$, *i.e.*, $\mathbf{A} = \mathbf{U}\mathbf{S}\mathbf{V}^*$, where $\mathbf{U}$ and $\mathbf{V}$ are unitary matrices and $\mathbf{S}$ is diagonal, we can conveniently write

$$\mathbf{Q}(\mu_k) = \big(2\mathbf{A}\mathbf{A}^* + \mu_k\mathbf{I}\big)^{-1} = \mathbf{U}\big(2\mathbf{S} + \mu_k\mathbf{I}\big)^{-1}\mathbf{U}^*, \quad (9)$$

where the matrix being inverted is diagonal. This equality shows that we can pre-compute $\mathbf{U}$ and $\mathbf{S}$ and any subsequent update of μ_k only requires recomputing a diagonal inversion, *i.e.*, with cost $O(m)$. This is particularly useful in scenarios where many instances of problem (1) are to be solved, all sharing the same matrix $\mathbf{A}$ and only differing in $\mathbf{y}$; this is the case, for example, in the sparse regression approach to hyper-spectral unmixing [8].

5. APPLICATION TO IMAGING PROBLEMS

This section shows how the algorithm in Fig. 2 can be conveniently applied in several imaging problems (namely, periodic deconvolution, inpainting, and compressive Fourier imaging), following the derivations in [1]. The essential issue in each case is the form of matrix $\mathbf{Q}(\mu_k)$ and its efficient computation. In all the cases, we adopt a frame-based synthesis formulation [17], thus $\mathbf{A} = \mathbf{B}\mathbf{W}$, where $\mathbf{W}$ is assumed to be the synthesis operator of a normalized tight (Paseval) frame (thus $\mathbf{W}\mathbf{W}^* = \mathbf{I}$).

5.1. Periodic Deconvolution

If $\mathbf{B}$ models a periodic convolution, it can be written as $\mathbf{B} = \mathbf{F}^*\mathbf{D}\mathbf{F}$, where $\mathbf{F}$ and $\mathbf{F}^*$ represent the 2D discrete Fourier transform (DFT) and its inverse, respectively, and $\mathbf{D}$ is a diagonal matrix containing the DFT coefficients of the convolution kernel. In this case,

$$\mathbf{Q}(\mu) = \big(2\mathbf{F}^*\mathbf{D}\mathbf{F}\mathbf{W}\mathbf{W}^*\mathbf{F}^*\mathbf{D}^*\mathbf{F}+\mu\mathbf{I}\big)^{-1} = \mathbf{F}^*\big(2|\mathbf{D}|^2+\mu\mathbf{I}\big)^{-1}\mathbf{F},$$

which has the same form as (9), with the DCT matrix $\mathbf{F}$ replacing $\mathbf{U}$. In this case, no SVD is required, but only knowledge of the DFT of the convolution kernel. Moreover, matrix-vector products by $\mathbf{F}$ and $\mathbf{F}^*$ are computed with cost $O(n \log n)$ using the *fast Fourier transform* (FFT) and updates to μ only imply recomputing a diagonal inverse (which has $O(n)$ cost).

5.2. Image Inpainting

To model the loss of pixels, which leads to inpainting problems, matrix $\mathbf{B}$ contains a subset of the rows of the identity matrix. Such a matrix satisfies $\mathbf{B}\mathbf{B}^* = \mathbf{I}$, thus we have simply

$$\mathbf{Q}(\mu) = \big(2\mathbf{B}\mathbf{W}\mathbf{W}^*\mathbf{B}^* + \mu\mathbf{I}\big)^{-1} = \frac{1}{2+\mu}\,\mathbf{I}.$$

Thus, multiplying a vector by matrix $\mathbf{Q}(\mu)$ simply corresponds to multiplying it by a scalar, which can be updated with negligible cost.

5.3. Compressive Fourier Imaging

In this case, $\mathbf{B} = \mathbf{M}\mathbf{F}$, where $\mathbf{M}$ is similar to the observation matrix in the inpainting problem (thus $\mathbf{M}\mathbf{M}^* = \mathbf{I}$), and $\mathbf{F}$ is the DFT matrix. Consequently, as in the inpainting case,

$$\mathbf{Q}(\mu) = \big(2\mathbf{M}\mathbf{F}\mathbf{W}\mathbf{W}^*\mathbf{F}^*\mathbf{M}^* + \mu\mathbf{I}\big)^{-1} = \frac{1}{2+\mu}\,\mathbf{I}.$$

6. EXPERIMENTS

In our experiments, we compare the proposed algorithm with the following baselines: ADMM directly applied to problem (1) (see [1], [10]), enhanced with over-relaxation with $\alpha = 1.8$; the *fast IST algorithm* (FISTA) [5]; the *sparse reconstruction by separable approximation* (SpaRSA) algorithm [29]; the monotonic version of the GPSR algorithm [21]. The proposed algorithm, herein referred to as BCQP-ADMM, is parametrized as follows: $\mu_0 = 0.1$, $\alpha = 1.8$, $\delta = 10$, $\xi = 10$, $\gamma = 2$. The initialization is as follows: $\mathbf{u}_0 = \left[\left((\mathbf{A}^*\mathbf{y})_+\right)^*, \left((-\mathbf{A}^*\mathbf{y})_+\right)^*\right]^*$, $\mathbf{d}_0 = \mathbf{0}$. The $\mathbf{Q}(\mu_k)$ matrix is computed and updated according to (9) (in both the ADMM and BCQP-ADMM algorithms).

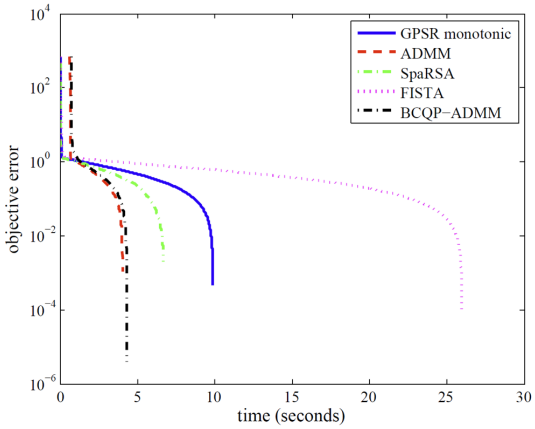

Fig. 3. Time evolution of the relative error in the objective function (see text for definition) in the compressive sensing experiment.

In the first experiment, we consider a typical CS toy problem: reconstructing a sparse vector from fewer observations than its dimension. In this example, $\mathbf{A}$ is a $2^{11} \times 2^{12}$ matrix filled with i.i.d. standard Gaussian samples, $\mathbf{x}$ is a 2^{12}-dimensional sparse vector with 150 randomly located non-zero (randomly chosen in $\{-1, +1\}$) components. Vector $\mathbf{y}$ is obtained by adding white Gaussian noise to $\mathbf{Ax}$ and the regularization parameter τ is set to $10^{-4}\|\mathbf{A}^*\mathbf{y}\|_\infty$, which corresponds to very weak regularization (see [21] for details). Fig. 3 plots (in log scale) the time evolution of the relative error in the objective function $(F_k - F^\star)/F^\star$, where F_k is the objective function value at iteration k and $F^\star$ is an estimate of its optimal value, obtained by previously running FISTA for a very large number of iterations. The plot shows that ADMM and BCQP-ADMM have remarkably similar behaviour, and clearly outperform the other methods. It is also visible that the ADMM and BCQP-ADMM algorithms have an initial delay associated to the computation of the SVD, but then quickly overtake the others. All the algorithms recover the original signal almost perfectly.

The second and third experiments use two benchmark image processing problems: deblurring and inpainting. In both cases, matrix $\mathbf{W}$ (see Section 5) is the synthesis operator of a 5-levels redundant (Daubechies-4) wavelet frame. In the deblurring example, $\mathbf{B}$ corresponds to the convolution with a 9×9 uniform kernel and noise of standard deviation 0.56 is added to the blurred image; in the inpainting problem, matrix $\mathbf{B}$ models the loss of 40% of the image pixels and noise of standard deviation 5 is added to the observed pixels (see [1] for further details about these benchmark problems). The plots in Figs. 4 and 5 reveal the same general conclusions as those of the first experiment: both ADMM and BCQP-ADMM outperform the other methods and have a remarkably similar behaviour.

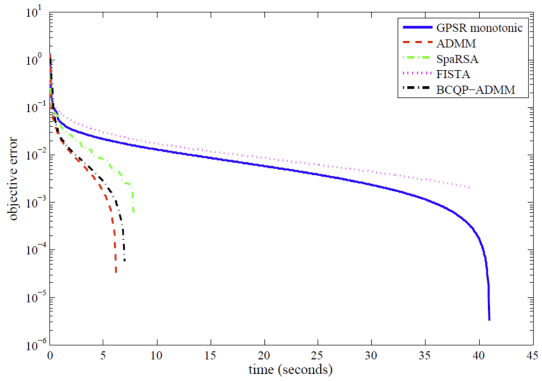

Fig. 4. Time evolution of the relative error in the objective function (see text for definition) in the image deblurring experiment.

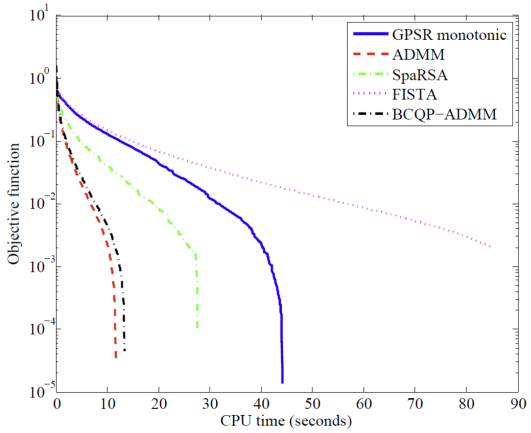

Fig. 5. Time evolution of the relative error in the objective function (see text for definition) in the image inpainting experiment.

7. CONCLUSIONS

In this paper we have revisited the BCQP formulation of the $\ell_2 + \ell_1$ optimization problem, introduced earlier in the context of sparse signal/image recovery [21]. Rather than the original gradient projection algorithm originally considered in [21], we have shown how the ADMM can be efficiently instantiated and implemented to address this problem. A preliminary set of experiments have shown that the ADMM applied to the BCQP formulation is competitive with the ADMM directly applied to the $\ell_2 + \ell_1$ problem.

Ongoing work includes trying to extend recent work on the optimal selection of the μ parameter of the ADMM algorithm applied to quadratic programming problems [23], for the case (not covered by the analysis in [23]) where the Hessian is not invertible. Another direction of ongoing research is the adaptation to the BCQP formulation of the method proposed in [3] to deal with non-periodic deconvolution problems.

8. REFERENCES

[1] M. Afonso, J. Bioucas-Dias, M. Figueiredo, "Fast image recovery using variable splitting and constrained optimization", *IEEE Trans. Image Processing,* vol. 19, pp. 2345-–2356, 2010.

[2] M. Afonso, J. Bioucas-Dias, M. Figueiredo, "An augmented Lagrangian approach to the constrained optimization formulation of imaging inverse problems", *IEEE Trans. Image Processing,*, vol. 20, pp. 681–695, 2011.

[3] M. Almeida and M. Figueiredo, "Deconvolving images with unknown boundaries using the alternating direction method of multipliers", *IEEE Transactions on Image Processing*, vol. 22, pp. 3084–3096, 2013.

[4] J. Barzilai, J. Borwein, "Two point step size gradient methods," *IMA Journal of Numerical Analysis,* vol. 8, pp. 141–148, 1988.

[5] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems", *SIAM Journal on Imaging Sciences*, vol. 2, pp. 183–202, 2009.

[6] S. Becker and M. Jalal Fadili, "A quasi-Newton proximal splitting method", *Neural Information Processing Systems – NIPS*, 2012 (available at `http://arxiv.org/abs/1206.1156`).

[7] J. Bioucas-Dias and M. Figueiredo, "A new TwIST: two-step iterative shrinkage/thresholding algorithms for image restoration", *IEEE Transactions on Image Processing*, vol. 16, no. 12, pp. 2992-3004, 2007.

[8] J. Bioucas-Dias and M. Figueiredo, "Alternating direction algorithms for constrained sparse regression: Application to hyperspectral unmixing", *2nd Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing – WHISPERS*, Reykjavik, 2010.

[9] D. Boley, "Local linear convergence of the alternating direction method of multipliers on quadratic or linear programs", *SIAM Journal on Optimization*, 2013 (to appear).

[10] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers", *Foundations and Trends in Machine Learning*, vol. 3, pp. 1-–122, 2011.

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

[12] P. Combettes and V. Wajs, "Signal recovery by proximal forward-backward splitting," *SIAM Journal on Multiscale Modeling & Simulation*, vol. 4, pp. 1168–1200, 2005.

[13] I. Daubechies, M. De Friese, and C. De Mol. "An iterative thresholding algorithm for linear inverse problems with a sparsity constraint." *Communications in Pure and Applied Mathematics*, vol. 57, pp. 1413–1457, 2004.

[14] D. Donoho. "Compressed sensing," *IEEE Transactions on Information Theory*, vol. 52, pp. 1289–1306, 2006.

[15] J. Eckstein, D. Bertsekas, "On the Douglas-–Rachford splitting method and the proximal point algorithm for maximal monotone operators", *Math. Program.,* vol. 5, pp. 293—318, 1992.

[16] M. Elad, B. Matalon, and M. Zibulevsky, "Image denoising with shrinkage and redundant representations", *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition – CVPR'2006*, New York, 2006.

[17] M. Elad, P. Milanfar, and R. Rubinstein, "Analysis versus synthesis in signal priors", *Inverse Problems*, vol. 23, pp. 947–968, 2007.

[18] E. Esser, "Applications of Lagrangian-based alternating direction methods and connections to split Bregman", Tech. Rep. 09-31, Comp. and Applied Math., UCLA, 2009.

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

[20] M. Figueiredo, J. Bioucas-Dias, and R. Nowak, "Majorization-minimization algorithms for wavelet-based image restoration" *IEEE Transactions on Image Processing*, vol. 16, no. 12, pp. 2980–2991, 2007.

[21] M. Figueiredo, R. Nowak, S. Wright, "Gradient projection for sparse reconstruction: application to compressed sensing and other inverse problems," *IEEE Journal on Selected Topics in Signal Processing*, vol. 1, pp. 586–598, 2007.

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

[23] E. Ghadimi, A. Teixeira, I. Shames, and M. Johansson, "Optimal parameter selection for the alternating direction method of multipliers (ADMM): quadratic problems", submitted, 2013 (available at `http://arxiv.org/abs/1306.2454`).

[24] C. Kanzow, N. Yamashitab, M. Fukushima, "Levenberg–Marquardt methods with strong local convergence properties for solving nonlinear equations with convex constraints", *Journal of Computational and Applied Mathematics*, vol. 172, pp. 375-–397, 2004.

[25] J. Lee, Y. Sun, M. Saunders, "Proximal Newton-type methods for minimizing composite function", Technical Report no. SOL-2013-1, Depart. of Management Science and Engineering, Stanford University, 2013 (available at `http://arxiv.org/abs/1206.1623`).

[26] M. Lustig, D. Donoho, J. Pauly, "Sparse MRI: the application of compressed sensing for rapid MR imaging," *Magnetic Resonance in Medicine,* vol. 58, pp. 1182–1195, 2007.

[27] Q. Tran Dinh, A. Kyrillidis, and V. Cevher. "Composite self-concordant minimization", submitted, 2013 (available at `http://arxiv.org/abs/1308.2867`).

[28] J. Tropp, "Just relax: Convex programming methods for identifying sparse signals," *IEEE Transactions on Information Theory,* vol. 51, pp. 1030–1051, 2006.

[29] S. Wright, R. Nowak, M. Figueiredo, "Sparse reconstruction by separable approximation", *IEEE Transactions on Signal Processing*, vol. 57, pp. 2479–2493, 2009.

[30] W. Yin, S. Osher, D. Goldfarb, J. Darbon, "Bregman iterative algorithms for ℓ_1-minimization with applications to compressed sensing", *SIAM Jour. Imaging Sciences*, vol. 1, pp. 143–168, 2008.