Homotopy Based Algorithms for \( \ell_0 \)-Regularized Least-Squares

Charles Soussen, Member, IEEE, Jérôme Idier, Member, IEEE, Junbo Duan, and David Brie, Member, IEEE

Abstract—Sparse signal restoration is usually formulated as the minimization of a quadratic cost function \( \| y - Ax \|_2^2 \) where \( A \) is a dictionary and \( x \) is an unknown sparse vector. It is well-known that imposing an \( \ell_0 \) constraint leads to an NP-hard minimization problem. The convex relaxation approach has received considerable attention, where the \( \ell_0 \)-norm is replaced by the \( \ell_1 \)-norm. Among the many effective \( \ell_1 \) solvers, the homotopy algorithm minimizes \( \| y - Ax \|_2^2 + \lambda \| x \|_1 \) with respect to \( x \) for a continuum of \( \lambda \)'s. It is inspired by the piecewise regularity of the \( \ell_1 \)-regularization path, also referred to as the homotopy path. In this paper, we address the minimization problem \( \| y - Ax \|_2^2 + \lambda \| x \|_1 \) for a continuum of \( \lambda \)'s and propose two heuristic search algorithms for \( \ell_0 \)-homotopy. Continuation Single Best Replacement is a forward–backward greedy strategy extending the Single Best Replacement algorithm, previously proposed for \( \ell_0 \)-minimization at a given \( \lambda \). The adaptive search of the \( \lambda \)-values is inspired by \( \ell_1 \)-homotopy. \( \ell_0 \) Regularization Path Descent is a more complex algorithm exploiting the structural properties of the \( \ell_0 \)-regularization path, which is piecewise constant with respect to \( \lambda \). Both algorithms are empirically evaluated for difficult inverse problems involving ill-conditioned dictionaries. Finally, we show that they can be easily coupled with usual methods of model order selection.

Index Terms—\( \ell_0 \)-homotopy, \( \ell_0 \)-regularized least-squares, \( \ell_1 \)-homotopy, model order selection, orthogonal least squares, sparse signal estimation, stepwise algorithms.

I. INTRODUCTION

Sparse approximation from noisy data is traditionally addressed as the constrained least-square problems

\[
\min_x \| y - Ax \|_2^2 \quad \text{subject to} \quad \| x \|_0 \leq k \tag{1}
\]

Manuscript received January 31, 2014; revised October 24, 2014 and March 18, 2015; accepted March 25, 2015. Date of publication April 10, 2015; date of current version May 28, 2015. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Joakim Jaldén. This work was carried out in part while C. Soussen was visiting IRCyN during the academic year 2010-2011 with the financial support of CNRS. (Corresponding author: Charles Soussen.)

C. Soussen and D. Brie are with the University of Lorraine and CNRS at the Centre de Recherche en Automatique de Nancy, UMR 7099, F-54506 Vandœuvre-lès-Nancy, France (e-mail: charles.soussen@univ-lorraine.fr, david.brie@univ-lorraine.fr).

J. Idier is with L’UNAM Université, Ecole Centrale Nantes and CNRS at the Institut de Recherche en Communications et Cybernétique de Nantes, UMR 6597, F-44321 Nantes, France (e-mail: jerome.idier@ircyn.ec-nantes.fr).

J. Duan was with CRAN. He is now with the Department of Biomedical Engineering, Xi’an Jiaotong University, Xi’an 710049, China (e-mail: junbo.duan@mail.xjtu.edu.cn).

This paper has supplementary downloadable multimedia material available at http://ieeexplore.ieee.org provided by the authors. This includes the Matlab functions implementing the Continuous Single Best Replacement (CSBR) and \( \ell_0 \)-Regularization Path Descent (\( \ell_0 \)-PD) algorithms together with simple test programs showing how to call these functions. This material is 304 KB in size. Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSP.2015.2421476

or

\[
\min_x \| x \|_0 \quad \text{subject to} \quad \| y - Ax \|_2^2 < \varepsilon \tag{2}
\]

where \( \| x \|_0 \) is the \( \ell_0 \)-“norm” counting the number of nonzero entries in \( x \), and the fidelity-to-data term \( \| y - Ax \|_2^2 \) measures the quality of approximation. Formulation (1) is well adapted when one has a knowledge of the maximum number \( k \) of atoms to be selected in the dictionary \( A \). On the contrary, the choice of (2) is more appropriate when the variance of the observation noise is known. \( \varepsilon \) may then be chosen relative to the noise variance. Since both (1) and (2) are subset selection problems, they are discrete optimization problems. They are known to be NP-hard except for specific cases [1].

When no knowledge is available on either \( k \) or \( \varepsilon \), the unconstrained formulation

\[
\min_x \{ \mathcal{J}(x; \lambda) - \| y - Ax \|_2^2 + \lambda \| x \|_0 \} \tag{3}
\]

is worth being considered, where \( \lambda \) expresses the trade-off between the quality of approximation and the sparsity level [2]. In a Bayesian viewpoint, (3) can be seen as a (limit) maximum a posteriori formulation where \( \| y - Ax \|_2^2 \) and the penalty \( \| x \|_0 \) are respectively related to a Gaussian noise distribution and a prior distribution for sparse signals [3].

A. Classification of Methods

1) \( \ell_0 \)-Constrained Least-Squares: The discrete algorithms dedicated to problems (1)–(2) can be categorized into two classes. First, forward greedy algorithms explore subsets of increasing cardinalities starting from the empty set. At each iteration, a new atom is appended to the current subset, therefore gradually improving the quality of approximation [4]. By increasing order of complexity, greedy algorithms include Matching Pursuit (MP) [5], Orthogonal Matching Pursuit (OMP) [6] and Orthogonal Least Squares (OLS) [7], also referred to as forward selection in statistical regression [8] and known as Order Recursive Matching Pursuit (ORMP) [9] and Optimized Orthogonal Matching Pursuit (OOMP) [10]. The second category are thresholding algorithms, where each iteration delivers a subset of same cardinality \( k \). Popular thresholding algorithms include Iterative Hard Thresholding [11], Subspace Pursuit [12] and CoSaMP [13].

Among these two categories, greedy algorithms are well-adapted to the resolution of (1) and (2) for \textit{variable} sparsity levels. Indeed, they yield a series of subsets for consecutive \( k \) (i.e., for decreasing approximation errors \( \varepsilon \)) since at each iteration, the current subset is increased by one element.
2) $\ell_2$-Penalized Least-Squares: In [3], we evidenced that the minimization of $J(x; \lambda)$ using a descent algorithm leads to bidirectional extensions of forward (orthogonal) greedy algorithms. To be more specific, consider a candidate subset $S$ corresponding to the support of $x$. Including a new element into $S$ yields a decrease of the square error, defined as the minimum of $||y - Ax||_2^2$ for $x$ supported by $S$. On the other hand, the penalty term $\lambda ||x||_0$ is increased by $\lambda$. Overall, the cost function $J(x; \lambda)$ decreases as soon as the square error variation exceeds $\lambda$. Similarly, a decrease of $J(x; \lambda)$ occurs when an element is removed from $S$ as soon as the square error increment is lower than $\lambda$. Because both inclusion and removal operations can induce a decrease of $J$, the formulation (3) allows one to design descent schemes allowing a “forward-backward” search strategy, where each iteration either selects a new atom (forward selection) or de-selects an atom that was previously selected (backward elimination). The Bayesian OMP [14] and Single Best Replacement (SBR) [3] algorithms have been proposed in this spirit. They are extensions of OMP and OLS, respectively. Their advantage over forward greedy algorithms is that an early wrong atom selection may be later cancelled. Forward-backward algorithms include the so-called stepwise regression algorithms which are OLS extensions [8], [15], [16], and OMP based algorithms of lower complexity [14], [17].

3) Continuous Relaxation of the $\ell_0$-Norm: The algorithms described so far are discrete search strategies dedicated to $\ell_0$-regularized least-squares. A classical alternative consists in relaxing the $\ell_0$-norm by a continuous function that is non-differentiable at 0; see, e.g., [18], [19] and [20]–[27] for $\ell_1$ and nonconvex relaxation, respectively. The $\ell_1$ problem $\min_{x} ||y - Ax||_2^2 + \lambda ||x||_0$ is referred to as both the LASSO and Basis Pursuit Denoising (BPDN). It leads to stepwise implementations [18], [28] including the popular $\ell_1$-homotopy [28]–[30], a forward-backward greedy search whose complexity is close to that of OMP. Importantly, $\ell_1$-homotopy solves the BPDN for a continuum of values of $\lambda$. It is closely connected to the Least Angle Regression (LARS), a simpler forward strategy allowing only atom selections, and referred to as “LARS with the LASSO modification” in [30].

B. Main Idea

Our approach is dedicated to $\ell_0$-penalized least-squares. It is based on the following geometrical interpretation.

First, for any subset $S$, we can define a linear function $\lambda \rightarrow \mathcal{E}(S) + \lambda |S|$, where $\mathcal{E}(S) = ||y - Ax||_2^2$ is the corresponding least-square error and $|S|$ stands for the cardinality of $S$. For each subset $S$, this function yields a line in the 2D domain $(\lambda, \mathcal{E})$, as shown in Fig. 1.

Second, the set of solutions to (3) is piecewise constant with respect to $\lambda$ (see Appendix for a proof). Geometrically, this result can be easily understood by noticing that the minimum of $J(x; \lambda)$ with respect to $x$ is obtained for all $\lambda$-values by considering the concave envelope of the set of lines $\lambda \rightarrow \mathcal{E}(S) + \lambda |S|$ for all subsets $S$. The resulting piecewise affine curve is referred to as the $\ell_0$-curve (see Fig. 1). Its edges are related to the supports of the sparse solutions for all $\lambda$, and its vertices yield the breakpoints $\lambda^*_i$ around which the set of optimal solutions $\arg \min_{x} J(x; \lambda)$ is changing.

We take advantage of this interpretation to propose two suboptimal greedy algorithms that address (3) for a continuum of $\lambda$-values. Continuation Single Best Replacement (CSBR) repeatedly minimizes $J(x; \lambda)$ with respect to $x$ for decreasing $\lambda$’s. $\ell_0$ Regularization Path Descent ($\ell_0$-PD) is a more complex algorithm maintaining a list of subsets so as to improve (decrease) the current approximation of the $\ell_0$ curve.

C. Related Works

1) Bi-Objective Optimization: The formulations (1), (2) and (3) can be interpreted as the same bi-objective problem because they all intend to minimize both the approximation error $||y - Ax||_2^2$ and the sparsity measure $||x||_0$. Although $x$ is continuous, the bi-objective optimization problem should rather be considered as a discrete one where both objectives reread $\mathcal{E}(S)$ and $|S|$. Indeed, the continuous solutions deduce from the discrete solutions, $x$ reading as a least-square minimizer among all vectors supported by $S$.

Fig. 2 is a classical representation where each axis is related to a single objective [31], namely $|S|$ and $\mathcal{E}(S)$. In bi-objective optimization, a point $S$ is called Pareto optimal when no other point $S'$ can decrease both objectives [32]. Here, $|S|$ takes integer values, thus the Pareto solutions are the minimizers of $\mathcal{E}(S)$ subject to $|S| < k$ for consecutive values of $k$. Equivalently, they minimize $|S|$ subject to $\mathcal{E}(S) \leq \varepsilon$ for some $\varepsilon$. They are called “supported” when they lay on the convex envelope of the Pareto frontier. On Fig. 2, the bullet points are supported solutions whereas the square point is non-supported. It is well known that a supported solution can be reached when minimizing the weighted sum of both objectives, i.e., when minimizing $\mathcal{E}(S) + \lambda |S|$ with respect to $S$ for some weight $\lambda$. On the contrary, the non-supported solutions cannot [32]. Choosing between the weighted sum method and a more complex approach is a nontrivial question. The answer depends on the problem.
at-hand and specifically, on the size of the nonconvex parts of the Pareto frontier.

2) $\ell_1$ and $\ell_0$-Homotopy Seen as a Weighted Sum Method: It is important to notice that for convex objectives, the Pareto solutions are all supported. For instance, consider the BPDN. Because $\|y - Ax\|_2^2$ and $\|x\|_1$ are convex functions of $x$, the set of minimizers of $\|y - Ax\|_2^2 + \lambda \|x\|_1$ for all $\lambda$ coincides with the set of minimizers of $\|y - Ax\|_2^2$ s.t. $\|x\|_1 \leq t$ for all $\ell$. [33]. They are referred to as the (unique) “$\ell_1$-regularization path”. The situation is different with $\ell_0$-regularization. Now, the weighted sum formulation (3) may not yield the same solutions as the constrained formulations (1) and (2) because the $\ell_0$-norm is nonconvex [2]. This will lead us to define two $\ell_0$-regularization paths, namely the “$\ell_0$-penalized path” and the “$\ell_0$-constrained path” (Section II).

On the algorithmic side, the $\ell_0$ problems are acknowledged to be difficult. Many authors actually discourage the direct optimization of $F$ because there are a very large number of local minimizers [20], [23]. In [3], however, we showed that forward-backward extensions of OLS are able to escape from some local minimizers of $\|y - Ax\|_2^2 + \lambda \|x\|_1$ for all $\lambda$. This motivates us to propose effective OLS-based strategies for minimizing $F$ for variable $\lambda$-values.

3) Positioning With Respect to Other Stepwise Algorithms: In statistical regression: the word “stepwise” originally refers to Efroymson’s algorithm [15], proposed in 1960 as an empirical extension of forward selection, i.e., OLS. Other stepwise algorithms were proposed in the 1980’s [8, Chapter 3] among which Berk’s and Broersen’s algorithms [16], [34]. All these algorithms perform a single replacement per iteration, i.e., a forward selection or a backward elimination. They were originally applied to over-determined problems in which the number of columns of $A$ is lower than the number of rows. Recent stepwise algorithms were designed as either OMP [14], [17] or OLS extensions [35], [36]. They all aim to find subsets of cardinality $k$ yielding a low approximation error $E(S)$ for all $k$. Although our algorithms share the same objective, they are inspired by (i) the $\ell_1$-homotopy algorithm; and (ii) the structural properties of the $\ell_0$-regularization paths. To the best of our knowledge, the idea of reconstructing an $\ell_0$-regularization path using $\ell_0$-homotopy procedures is novel.

CSBR, first sketched in [37], repeatedly minimizes $F(x; \lambda)$ for decreasing $\lambda$’s. On the contrary, $\ell_0$-PD minimizes $F(x; \lambda)$ for any $\lambda$ simultaneously by maintaining a list of candidate subsets. The idea of maintaining a list of support candidates was recently developed within the framework of forward selection [38], [39]. Our approach is different because a family of optimization problems are being addressed together. In contrast, the supports in the list are all candidates to solve the same problem in [38], [39].

4) Positioning With Respect to Continuation Algorithms: The principle of continuation is to handle a difficult problem by solving a sequence of simpler problems with warm start initialization, and gradually tuning some continuous hyperparameter [40]. In sparse approximation, the word continuation is used in two opposite contexts.

First, the BDPN problem involving the $\ell_1$-norm. BPDN is solved for decreasing hyperparameter values using the solution for each value as a warm starting point for the next value [4]. $\ell_1$-homotopy [28], [30], [41] exploits that the $\ell_1$ regularization path is piecewise affine and tracks the breakpoints between consecutive affine pieces. CSBR is designed in a similar spirit and can be interpreted as an $\ell_0$-homotopy procedure although the $\ell_0$ minimization steps are solved in a sub-optimal way.

Second, the continuous approximation of the (discrete) $\ell_0$ pseudo-norm [42] using a Graduated Non Convexity (GNC) approach [43]: a series of continuous concave metrics is considered leading to the resolution of continuous optimization problems with warm start initialization. Although the full reconstruction of the $\ell_0$-regularization paths has been rarely addressed, it is noticeable that a GNC-like approach, called SparseNet, aims to gradually update some estimation of the regularization path induced by increasingly non-convex sparsity measures [44]. This strategy relies on the choice of a grid of $\lambda$-values. Because the influence of the grid is critical [33], it is useful to adapt the grid while the nonconvex measure is modified [44]. On the contrary, our approach does not rely on a grid definition. The $\lambda$-values are rather adaptively computed.

The paper is organized as follows. In Section II, we define the $\ell_0$-regularization paths and establish their main properties. The CSBR and $\ell_0$-PD algorithms are respectively proposed in Sections III and IV. In Section V, both algorithms are analyzed and compared with the state-of-art algorithms based on nonconvex penalties for difficult inverse problems. Additionally, we investigate the automatic choice of the cardinality $k$ using classical order selection rules.

II. $\ell_0$-Regularization Paths

A. Definitions, Terminology and Working Assumptions

Let $m \times n$ denote the size of the dictionary $A$ (usually, $m < n$ in sparse approximation). The observation signal $y$ and the sparse vector $x$ are of size $m \times 1$ and $n \times 1$, respectively. We assume that any $\min(m, n)$ columns of $A$ are linearly independent so that for any subset $S \subset \{1, \ldots, n\}$, the submatrix of $A$ gathering the columns indexed by $S$ is full column rank, and the least-square error $E(S)$ can be numerically computed. This assumption is however not necessary for the theoretical results provided hereafter.
We denote by $|S|$ the cardinality of a subset $S$. We use the alternative notations “$S + \{i\}$” and “$S - \{i\}$” for the forward selection $S \cup \{i\}$ and backward elimination $S \setminus \{i\}$. We can then introduce the generic notation $S \pm \{i\}$ for single replacements: $S \pm \{i\}$ stands for $S + \{i\}$ if $i \notin S$, and $S \pm \{i\}$ if $i \in S$. We will frequently resort to the geometrical interpretation of Fig. 1. With a slight abuse of terminology, the line $\lambda \mapsto \mathcal{E}(S) + \lambda S$ will be simply referred to as “the line $S$”.

Hereafter, we start by defining the $\ell_0$-regularized paths as the sets of supports of the solutions to problems (1), (2) and (3) for varying hyperparameters. As seen in Section I, the solutions may differ whether the $\ell_0$-regularization takes the form of a bound constraint or a penalty. This will lead us to distinguish the “$\ell_0$-constrained path” and the “$\ell_0$-penalized path”. We will keep the generic terminology “$\ell_0$-regularization paths” for statements that apply to both.

B. Definition and Properties of the $\ell_0$-Regularized Paths

The continuous problems (1), (2) and (3) can be converted as discrete problems:

\[
\min_S \mathcal{E}(S) \quad \text{subject to} \quad S \leq k, \tag{4}
\]

\[
\min_S |S| \quad \text{subject to} \quad \mathcal{E}(S) \leq \varepsilon, \tag{5}
\]

\[
\min_S \{ \mathcal{J}(S; \lambda) \triangleq \mathcal{E}(S) + \lambda |S| \}, \tag{6}
\]

where $S$ stands for the support of $x$. The optimal solutions $x$ to problems (1), (2) and (3) can indeed be deduced from those of (4), (5) and (6), respectively, $x$ reading as the least-square minimizers among all vectors supported by $S$. In the following, the formulation (5) will be omitted because it leads to the same $\ell_0$-regularization path as formulation (4) [2].

Let us first define the set of solutions to (4) and (6) and the $\ell_0$-curve, related to the minimum value in (6) for all $\lambda > 0$.

**Definition 1:** For $k \leq \min(m, n)$, let $S^*_0(k)$ be the set of minimizers of the constrained problem (4).

For $\lambda > 0$, let $S^*_0(\lambda)$ be the set of minimizers of the penalized problem (6). Additionally, let us define the $\ell_0$-curve as the function $\lambda \mapsto \min_S \{ \mathcal{J}(S; \lambda) \}$. It is the concave envelope of a finite number of linear functions. Thus, it is concave and piecewise affine. Let $\lambda^*_1 < \cdots < \lambda^*_n$ denote the affine intervals ($I + 1$ contiguous intervals; see Fig. 1 in the case where $I = 2$).

Each set $S^*_0(k)$ or $S^*_0(\lambda)$ can be thought of as a single support ($e.g.$, $S^*_0(k)$ is reduced to the support $S^*_0$ in the example of Fig. 2). They are defined as sets of supports because the minimizers of (4) and (6) might not be always unique. Let us now provide a key property of the set $S^*_0(\lambda)$.

**Theorem 1:** $S^*_0(\lambda)$ is a piecewise constant function of $\lambda$, being constant on each interval $\lambda \in (\lambda^*_i, \lambda^*_i + 1)$.

**Proof:** See Appendix. 

This property allows us to define the $\ell_0$-regularization paths in a simple way.

**Definition 2:** The $\ell_0$-constrained path is the set (of sets) $S^*_0 = \{S^*_0(k), k = 0, \ldots, \min(m, n)\}$.

The $\ell_0$-penalized path is defined as $S^*_p = \{S^*_p(\lambda), \lambda > 0\}$. According to Theorem 1, $S^*_p(\lambda)$ is composed of $(I + 1)$ distinct sets $S^*_p(\lambda_i)$, one for each interval $\lambda \in (\lambda^*_i, \lambda^*_i + 1)$.

C. Approximate $\ell_0$-Penalized Regularization Path

The solutions delivered by our greedy algorithms will be referred to as the “approximate $\ell_0$-penalized path” since they are suboptimal algorithms. Throughout the paper, the $*$ notation is reserved for optimal solutions (e.g., $S^*_0$). It is removed when dealing with approximate solutions. The outputs of our algorithms will be composed of a list $\lambda = \{\lambda_1, \ldots, \lambda_{J+1}\}$ of decreasing $\lambda$-values and a list $S = \{S_1, \ldots, S_J\}$ of candidate supports, with $S_0 = \emptyset$. $S_j$ is a suboptimal solution to (6) for $\lambda \in (\lambda_{j+1}, \lambda_j)$. In the first interval $\lambda > \lambda_1$, the solution is $S_0 = \emptyset$. The reader shall keep in mind that each output $S_j$ induces a suboptimal solution $x_j$ to (3) for $\lambda \in (\lambda_{j+1}, \lambda_j)$. This vector is the least-square solution supported by $S_j$. It can be computed using the pseudo-inverse of the subdictionary indexed by the set of atoms in $S_j$.

Geometrically, each support $S_j$ yields a line segment. Appending these segments yields an approximate $\ell_0$-curve covering the domain $(\lambda_{J+1}, +\infty)$, as illustrated in Fig. 3.

III. GREEDY CONTINUATION ALGORITHM (CSBR)

Our starting point is the Single Best Replacement algorithm [3] dedicated to the minimization of $\mathcal{J}(x; \lambda)$ with respect to $x$, or equivalently to $\mathcal{J}(S; \lambda) = \mathcal{E}(S) + \lambda S$ with respect to $S$. We first describe SBR for a given $\lambda$. Then, the CSBR extension is presented for decreasing and adaptive $\lambda$’s.

A. Single Best Replacement

SBR is a deterministic descent algorithm dedicated to the minimization of $\mathcal{J}(S; \lambda)$ with the initial solution $S = \emptyset$. An SBR iteration consists of three steps:

1) Compute $\hat{\mathcal{J}}(S \pm \{i\}; \lambda)$ for all possible single replacements $S \pm \{i\}$ (selection and de-selection trials); 2) Select the best replacement $S_{\text{best}} = S \pm \{\ell\}$, with

$$\ell \in \arg \min_{i \in \{1, \ldots, n\}} \hat{\mathcal{J}}(S \pm \{i\}; \lambda);$$

Fig. 3. Notations relative to our heuristic search algorithms. Their outputs are: (i) a sequence of values $\lambda_j$ sorted in the decreasing order; (ii) as many supports $S_j, S_j$ being the solution associated to all $\lambda \in (\lambda_{j+1}, \lambda_j)$. By extension, $S_0 = \emptyset$ for $\lambda > \lambda_1$.

$S^*_0$ gathers the solutions to (4) for all $k$. As illustrated in Fig. 2, the elements of $S^*_0$ are the Pareto solutions whereas the elements of $S^*_p$ correspond to the convex envelope of the Pareto frontier. Therefore, both $\ell_0$-regularization paths may not coincide [2], [31]. As stated in Theorem 2, $S^*_p \subset S^*_0$, but the reverse inclusion is not guaranteed.

**Theorem 2:** $S^*_p \subset S^*_0$. Moreover, for any $\lambda \notin \{\lambda_1^*, \ldots, \lambda_0^*\}$, there exists $k$ such that $S^*_p(\lambda) = S^*_0(k)$.

**Proof:** See Appendix.
columns 38x749] SOUSSEN et al.: HOMOTOPY BASED ALGORITHMS FOR \textit{\ell}_1-REGULARIZED LEAST-SQUARES 3305

| TABLE I |
| SBR ALGORITHM FOR MINIMIZATION OF $\mathcal{J}(S; \lambda)$ FOR FIXED $\lambda$ [3]. BY DEFAULT, $S_{\text{init}} = \emptyset$. THE OUTPUTS $\delta \varepsilon_{\text{add}}$ AND $\delta \varepsilon_{\text{delt}}$ ARE OPTIONAL |

| inputs | $A, y, \lambda, S_{\text{init}}$ |
| outputs | $S, \delta \varepsilon_{\text{add}}, \delta \varepsilon_{\text{delt}}$ |
| $S_{\text{best}} \leftarrow S_{\text{init}}$; |
| repeat |
| for $i = 1$ to $n$ do |
| Compute $\mathcal{J}(S \pm \{i\}; \lambda)$; |
| end |
| until $\mathcal{J}(S_{\text{best}}; \lambda) \geq \mathcal{J}(S; \lambda)$; |
| Compute $\delta \varepsilon_{\text{delt}}$ according to (11); |
| Set $\delta \varepsilon_{\text{add}} \leftarrow \mathcal{E}(S) - \mathcal{E}(S + \{i\})$; |

3) Update $S \leftarrow S_{\text{best}}$.

SBR terminates when $\mathcal{J}(S_{\text{best}}; \lambda) \geq \mathcal{J}(S; \lambda)$, i.e., when no single replacement can decrease the cost function. This occurs after a finite number of iterations because SBR is a descent algorithm and there are a finite number of possible subsets $S \subset \{1, \ldots, n\}$. In the limit case $\lambda = 0$, only forward selections can yield a decrease of the square error $\mathcal{J}(S; 0) = \mathcal{E}(S)$. SBR coincides with the well-known OLS algorithm [7]. Generally, the $n$ replacement trials necessitate to compute $\mathcal{E}(S + \{i\})$ for all selection trials and $\mathcal{E}(S - \{i\})$ for all de-selections. In [3], we proposed a fast and stable recursive implementation based on the Cholesky factorization of the Gram matrix $A_S^T A_S$ when $S$ is modified by one element (with $A_S$ the submatrix of $A$ gathering the columns indexed by $S$). SBR is summarized in Table I.

The optional output parameters $\delta \varepsilon_{\text{add}}$ and $\delta \varepsilon_{\text{delt}}$ are unnecessary in the standard version. Their knowledge will be useful to implement the extended CSBR algorithm.

Let us illustrate the behavior of SBR on a simple example using the geometrical interpretation of Fig. 4, where a single replacement is represented by a vertical displacement (from top to bottom) between the two lines $S$ and $S \pm \{i\}$. $S_{\text{init}} = \emptyset$ yields an horizontal line since $\mathcal{J}(\emptyset; \lambda) = \|y\|_2^2$ does not depend on $\lambda$. At the first iteration, a new dictionary atom $\ell = a$ is selected. The line related to the updated support $S \leftarrow \{a\}$ is of slope $|S| = 1$. Similarly, some new dictionary atoms $b$ and $c$ are being selected in the next iterations, yielding $S \leftarrow \{a, b\}$ and $S \leftarrow \{a, b, c\}$. On Fig. 4, the dotted lines related to the latter supports have slopes equal to 2 and 3. At iteration 4, the single best replacement is the de-selection $\ell = a$. The resulting support $S \leftarrow \{b, c\}$ is of cardinality 2, and the related line is parallel to the line $\{a, b\}$ found at iteration 2. During the fifth iteration, none of the single replacements $S \pm \{i\}$ decreases $\mathcal{J}(\{b, c\}; \lambda)$. SBR stops with output $S = \{b, c\}$.

B. Principle of the Continuation Search

Our continuation strategy is inspired by $\ell_1$-homotopy which recursively computes the minimizers of $\|y - Ax\|_2^2 + \lambda \|x\|_1$ when $\lambda$ is continuously decreasing [28]–[30]. An iteration of $\ell_1$-homotopy consists of two steps:

- Find the next value $\lambda_{\text{new}} < \lambda_{\text{cur}}$ for which the $\ell_1$ optimality conditions are violated with the current active set $S$ ($\lambda_{\text{cur}}$ denotes the current value);
- Compute the single replacement $S \leftarrow S \pm \{i\}$ allowing to fulfill the $\ell_1$ optimality conditions at $\lambda = \lambda_{\text{new}}$.

CSBR follows the same principle. The first step is now related to some local $\ell_1$-optimality conditions, and the second step consists in calling SBR at $\lambda_{\text{new}}$ with the current active set as initial solution; see Fig. 5 for a sketch.

1) Local Optimality Conditions: Let us first reformulate the stopping conditions of SBR at a given $\lambda$. SBR terminates when a local minimum of $\mathcal{J}(S; \lambda)$ has been found:

$$\forall i \in \{1, \ldots, n\}, \mathcal{J}(S \pm \{i\}; \lambda) \geq \mathcal{J}(S; \lambda). \quad (8)$$

By separating the conditions related to selections $S + \{i\}$ and de-selections $S - \{i\}$, (8) rereads as the interval condition:

$$\lambda \in [\delta \varepsilon_{\text{add}}(S), \delta \varepsilon_{\text{remv}}(S)], \quad (9)$$

where

$$\delta \varepsilon_{\text{add}}(S) \triangleq \max_{i \in S} \{\mathcal{E}(S) - \mathcal{E}(S + \{i\})\} \quad (10a)$$

$$\delta \varepsilon_{\text{remv}}(S) \triangleq \min_{i \in S} \{\mathcal{E}(S - \{i\}) - \mathcal{E}(S)\} \quad (10b)$$

refer to the maximum decrease (respectively, minimum increase) of the square error when an atom is selected (de-selected). Condition (8) is illustrated in Fig. 6(a): all lines related to single replacements $S \pm \{i\}$ lay above the black point depicting the value of $\mathcal{J}(S; \lambda)$ for the current $\lambda$. 

Fig. 4. Step-by-step illustration of the call $S = \text{SBR}(\emptyset; \lambda)$. Each single replacement is represented by a vertical displacement (from top to bottom) from lines $S$ to $S \pm \{i\}$. The symbols ‘+’ and ‘−’ respectively refer to the selection and de-selection of atoms $a$, $b$, and $c$. Four SBR iterations are done from the initial support $S_{\text{init}} = \emptyset$, the selection of $a$, $b$, and $c$, and the de-selection of $a$. The final output $S = \{b, c\}$ is of cardinality 2.

Fig. 5. Step-by-step illustration of CSBR with the early stopping condition $\lambda_{\text{stop}}$. The initial support is $S_0 = \emptyset$. SBR is called for three decreasing values (solid vertical arrows), with output $S_2$ at $\lambda_2$. The search for the next value $\lambda_{\text{stop}}$ is represented by an oblique displacement along the line $S_j$.
2) Violation of the Local Optimality Conditions: Consider the current output $S = SBR(S_{\text{init}}; \lambda_{\text{cut}})$. The local optimality condition (9) is then met for $\lambda = \lambda_{\text{cut}}$, but also for any $\lambda \in [\delta E_{\text{add}}(S), \lambda_{\text{cut}}]$. The new value for which (9) is violated is $\lambda_{\text{new}} = \delta E_{\text{add}}(S) - \epsilon$ where $\epsilon > 0$ is arbitrarily small. The violation occurs for $S + \{\ell_{\text{add}}\}$, with

$$\ell_{\text{add}} \in \arg \max_{i \in S} \{E(S) - E(S + \{i\})\}. \quad (11)$$

In practice, $\lambda_{\text{new}}$ can be set to the limit value

$$\lambda_{\text{new}} = \delta E_{\text{add}}(S) \quad (12)$$

provided that $S$ is replaced with $S + \{\ell_{\text{add}}\}$.

As illustrated in Fig. 6(b), the line $S + \{\ell_{\text{add}}\}$ lays below all other parallel lines $S + \{i\}$ (dotted lines). Here, the $\lambda$-axis has been stretched by an arbitrary factor for improved readability. The horizontal length $\lambda_{\text{new}}$ would match the vertical length $\delta E_{\text{add}}(S)$ without any stretching. The same stretching process will be done in Fig. 7.

C. CSBR Algorithm

CSBR is summarized in Table II. The repeated calls to SBR deliver subsets $S_j$ for decreasing $\lambda_j$. As shown in Fig. 5, the solution $S_j$ covers the interval $[\lambda_{j+1}, \lambda_j]$. At the very first iteration, we have $S_0 = \emptyset$, and (11)–(12) reread:

$$\ell_{\text{add}} \in \arg \max_{i \in S} \left\{ \frac{|y_i|}{a_i^2} \right\}$$

and

$$\lambda_1 = \frac{|y_1 a_{\ell_{\text{add}}}^2}{a_{\ell_{\text{add}}}^2}. \quad (13)$$

By default, CSBR stops when $\lambda_j \rightarrow \infty$, i.e., the whole domain $\lambda \in \mathbb{R}_+$ has been scanned. However, this choice may not be appropriate when dealing with noisy data and overcomplete dictionaries. An early stopping rule can then be considered.

| INDEX | CSBR ALGORITHM: SBR IS CALLED REPEATEDLY FOR DECREASING $\lambda_j$’S THAT ARE RECURSIVELY COMPUTED, AND PROVIDED AS SBR OUTPUTS |
|-------|----------------------------------------------------------------------------------------------------------------------------|
| inputs | $\mathbf{A}, \mathbf{y}$                                                                                                           |
| outputs | $S$: list of supports $S_j$; $\lambda$: list of $\lambda_j$                                                                           |
| $S_0 \leftarrow \emptyset$; Compute $S_{\text{init}} \leftarrow \{\ell_{\text{add}}\}$ from (13); Compute $\lambda_1$ from (13); | |
| $j \leftarrow 1$; while $\lambda_j > 0$ do Call $[S_j, \delta E_{\text{add}}, \ell_{\text{add}}] = SBR(S_{\text{init}}; \lambda_j)$; | |
| $\lambda_{j+1} \leftarrow \delta E_{\text{add}}$; $S_{\text{init}} \leftarrow S_j + \{\ell_{\text{add}}\}$; | |
| $j \leftarrow j + 1$; end | |

IV. $\ell_0$-REGULARIZATION PATH DESCENT ($\ell_0$-PD)

On the theoretical side, the $\ell_0$-penalized regularization path is piecewise constant (Theorem 1). It yields the $\ell_0$ curve which is piecewise affine, continuous and concave (Fig. 1). The curve related to the CSBR outputs does not fulfill this property since there might be jumps in this curve, and the slope of line $S_j$ is not necessarily increasing with $j$ (see Fig. 5). This motivates us to propose another algorithm whose outputs are consistent with the structural properties of the $\ell_0$-curve.

We propose to gradually update a list $S$ of candidate subsets $S_j$ while imposing that the related curve is a concave polygon, obtained as the concave envelope of the set of lines $S_j$ (see Fig. 7(a)). The subsets in $S$ are updated so as to decrease at most the concave polygonal curve.

A. Descent of the Concave Polygon

The principle of $\ell_0$-PD is to perform a series of descent steps, where a new candidate subset $S_{\text{new}}$ is considered and included in the list $S$ only if the resulting concave polygon can be decreased. This descent test is illustrated in Fig. 7 for two examples (left and right columns). For each example, the initial polygon is represented in (a). It is updated when its intersection with the line $S_{\text{new}}$ is non-empty (b). The new concave polygon (c) is obtained as the concave envelope of the former polygon and the line $S_{\text{new}}$. All subsets in $S$ whose edges that belong to the line $S_{\text{new}}$ are removed from $S$. 

SOUSSEN et al.: HOMOTOPY BASED ALGORITHMS FOR $\ell_q$-REGULARIZED LEAST-SQUARES

Fig. 7. $\ell_0$-PD algorithm: descent of the concave polygon when a new support $S_{new} = S_j + \{\ell_\text{add}\}$ (left) or $S_{new} = S_j - \{\ell_\text{rmv}\}$ (right) is included. (a) Initial configuration. (b) The intersection with line $S_{new}$ is computed. This yields an interval $[\lambda_{\text{inf}}, \lambda_{\text{sup}}]$ for which $\lambda_j$ lays below the concave polygon. (c) When this interval is non-empty, the supports whose related edges lay above the line $S_{new}$ are removed while $\lambda_j$ is included in $\lambda^T$. The values $\lambda_j$ (corresponding to the vertices of the new concave polygon) are being updated. Their number may either increase or decrease.

TABLE III

TABLE III contains the concave polygon descent procedure. When a new subset is included, both lists $S$ and $\lambda$ are updated. The function intersect computes the intersection between a line and a concave polygon. This yields an interval $[\lambda_{\text{inf}}, \lambda_{\text{sup}}]$. By convention, $\lambda_{\text{inf}} > \lambda_{\text{sup}}$ when the intersection is empty.

**Procedure: CCV_Descent($S$, $S_{new}$, $\lambda$)**

- Call $[\lambda_{\text{inf}}, \lambda_{\text{sup}}] \leftarrow \text{intersect}(S, S_{new})$;
- if $\lambda_{\text{inf}} < \lambda_{\text{sup}}$ then
  - Include $S_{new}$ as an unexplored support in $S$;
  - Remove any subset $S_j$ from $S$ such that $[\lambda_{j+1}, \lambda_j) \subseteq [\lambda_{\text{inf}}, \lambda_{\text{sup}}]$;
  - Sort the subsets in $S$ by increasing cardinality;
  - Sort $\lambda$ in the decreasing order;
- end

This procedure is formally presented in Table III. Let us now specify how the new candidate subsets $S_{new}$ are built.

**B. Selection and Exploration of the New Candidate Subset**

We first need to assign a Boolean label $S_j.\text{expl}$ to each subset $S_j$. It equals 1 if $S_j$ has already been “explored” and 0 otherwise. The following exploration process is carried out given a subset $S = S_j$: all possible single replacements $S \pm \{i\}$ are tested. The best selection $\ell_{\text{add}}$ and de-selection $\ell_{\text{rmv}}$ are both kept in memory, with $\ell_{\text{add}}$ defined in (11) and similarly,

$$\ell_{\text{rmv}} \in \arg \min_{i \in S} \{E(S - \{i\}) - E(S)\}. \tag{14}$$

At any $\ell_0$-PD iteration, the unexplored subset $S_j$ of lowest cardinality (i.e., of lowest index $j$) is selected. $\ell_0$-PD attempts to include $S_{add} = S_j + \{\ell_{\text{add}}\}$ and $S_{rmv} = S_j - \{\ell_{\text{rmv}}\}$ into $S$ so that the concave polygon can be decreased at most. The $\text{CCV}_{\text{Descend}}$ procedure (Table III) is first called with $S_{new} \leftarrow S_{add}$ leading to possible updates of $S$ and $\lambda$. It is called again with $S_{new} \leftarrow S_{rmv}$. Fig. 7 illustrates each of these calls. When a support $S_j$ has been explored, the new supports that have been included in $S$ (if any) are tagged as unexplored.

**C. $\ell_0$-PD Algorithm**

$\ell_0$-PD is stated in Table IV. Initially, $S$ is formed of the empty support $S = \emptyset$. The resulting concave polygon is reduced to a single horizontal edge. The corresponding endpoints are $\lambda_1 = 0$ and (by extension) $\lambda_0 \triangleq +\infty$. In the first iteration, $S_0$ is explored: $S_{add} = \{\ell_{\text{add}}\}$ is computed in (13) and included in $S$ during the call to $\text{CCV}_{\text{Descend}}$. The updated set $S$ is now composed of $S_0 = \emptyset$ (explored) and $S_1 = S_{add}$. The new concave polygon has two edges delimited by $\lambda_2 = 0$, $\lambda_1$ and $\lambda_0 = +\infty$, with $\lambda_1$ given in (13). Generally, either 0, 1, or 2 new unexplored supports $S_{add}$ and $S_{rmv}$ may be included in $S$ at a given iteration while a variable number of supports may be removed from $S$.

$\ell_0$-PD terminates when all supports in $S$ have been explored. When this occurs, the concave polygon cannot decrease anymore with any single replacement $S_j \pm \{j\}$, with $S_j \in S$. Practically, the early stopping rule $\lambda_j \leq \lambda_{\text{stop}}$ can be adopted, where $\lambda_j$ denotes the unexplored subset having the least cardinality. This rule ensures that all candidate subsets $S_j$ corresponding to the interval $(\lambda_{\text{stop}}, +\infty)$ have been explored. Similar to CSBR, alternative stopping conditions of the form $S_j \geq k_{\text{stop}}$ or $E(S_j) < \varepsilon_{\text{stop}}$ can be adopted.

**D. Fast Implementation**

The $\text{CCV}_{\text{Descend}}$ procedure calls the function intersect to compute the intersection between a concave polygon $S$ and a line $S_{new}$. Lemma 1 states that this intersection is empty in two
simple situations. Hence, the call to intersect is not needed in these situations. This implementation detail is omitted in Table III for brevity reasons.

Lemma 1: Let \( S = \{S_0, \ldots, S_J\} \) be a list of supports associated to a continuous, concave polygon \( \lambda \rightarrow \min_{\lambda} \mathcal{J}(S_j; \lambda) \) with \( J + 1 \) edges, delimited by \( \lambda = \{\lambda_0, \ldots, \lambda_{J+1}\} \). The following properties hold for all \( j \):

- If \( \delta E_{\text{add}}(S_j) < \lambda_{j+1} \), then the line \( S_{\text{add}} = S_j + \{\ell_{\text{add}}\} \) lays above the current concave polygon.
- If \( \delta E_{\text{inv}}(S_j) > \lambda_j \), then the line \( S_{\text{inv}} = S_j - \{\ell_{\text{inv}}\} \) lays above the current concave polygon.

Proof: We give a sketch of proof using geometrical arguments. Firstly, \( \delta E_{\text{add}}(S_j) \) is the \( \lambda \)-value of the intersection point between lines \( S_j \) and \( S_{\text{add}} = S_j + \{\ell_{\text{add}}\} \); see Fig. 7(b). Secondly, we notice that \( S_j < S_{\text{add}} < S_{j+1} \) because the polygon is concave and \( S_{\text{add}} = S_j + 1 \). It follows from these two facts that if \( \delta E_{\text{add}}(S_j) < \lambda_{j+1} \), the line \( S_{\text{add}} \) lays above \( S_{j+1} \) for \( \lambda \leq \lambda_{j+1} \), and above \( S_j \) for \( \lambda > \lambda_{j+1} \).

This proves the first result. A similar sketch applies to the second result.

E. Main Differences Between CSBR and \( \ell_0 \)-PD

First, we stress that contrary to CSBR, \( \lambda_i \) does not identify with the iteration number anymore for \( \ell_0 \)-PD. Actually, the current iteration of \( \ell_0 \)-PD is related to an edge of the concave polygon, i.e., a whole interval \( [\lambda_{j+1}, \lambda_j] \) whereas the current iteration of CSBR is dedicated to a single value \( \lambda_j \).

Second, the computation of the next value \( \lambda_{j+1} \) in CSBR is only based on the violation of the lower bound of (9), corresponding to atom selections. In \( \ell_0 \)-PD, the upper bound is considered as well. This is the reason why the \( \lambda \)-values are not scanned in a decreasing order anymore. This may improve the very sparse solutions found in the early iterations within an increased computation time, as we will see hereafter.

V. NUMERICAL RESULTS

The algorithms are evaluated on two kinds of problems involving ill-conditioned dictionaries. Their behavior is first analyzed for simple examples. Then, we provide a detailed comparison with other nonconvex algorithms.

A. Two Generic Problems

The sparse deconvolution problem takes the form \( y = h \ast x^* + n \) where \( h \) is a Gaussian impulse response of standard deviation \( \sigma \) and the noise \( n \) is assumed i.i.d. and Gaussian. \( h \) is approximated by a finite impulse response of length \( 6\sigma \) by thresholding the smallest values. The problem rereads \( y = Ax^* + n \) where \( A \) is a convolution matrix. In the default setting, \( y \) and \( x \) are sampled at the same frequency. \( A \) is a Toeplitz matrix of dimensions chosen so that any Gaussian feature of the dictionary is fully contained within the observation window \([1, \ldots, m]\). This implies that \( A \) is slightly undercomplete (\( m > n \)). Two simulated data vectors \( y \) are represented in Fig. 8(a),(b) and (c),(d).

B. Empirical Behavior of CSBR and \( \ell_0 \)-PD

1) Example: Consider the problem shown in Fig. 8(c). Because CSBR and \( \ell_0 \)-PD provide similar results, we only display the CSBR results. Sparse solutions \( x_j \) are delivered for decreasing \( \lambda_j \), \( x_j \) being the least-square solution supported by the \( j \)-th output of CSBR \( (S_j) \). Three sparse solutions are represented on Fig. 9(b). On Fig. 9(a), only the seven main jumps are...
Fig. 9. Jump detection example: processing of the data of Fig. 8(c) using CSBR. Three sparse solutions $\bm{x}_j$ are shown, each being related to some CSBR output $S_j$, with $\| \bm{x}_j \|_0 = |S_j|$. The data vector $\bm{y}$ is represented in dashed lines and the approximation $\mathcal{A}_{\bm{x}_j}$ is in solid line. (a) $|S_7| = 7$, $\lambda_7 = 7.8e^{-2}$. (b) $|S_9| = 11$, $\lambda_9 = 4.1e^{-3}$. (c) $|S_{29}| = 41$, $\lambda_{29} = 3.4e^{-4}$. 

Fig. 10. Model order selection using MDLc: sparse solution and related data approximation. (a) Simulation shown in Fig. 9, with $k = 30$ and SNR = 25 dB. The MDLc solution related to CSBR is $S_{27}$, with $|S_{27}| = 29$. Simulation of Fig. 8(b), with $k = 30$ and SNR = 10 dB. The MDLc solution related to $\ell_0$-PD is $S_{15}$, with $|S_{15}| = 16$. 

being detected (lowest value of $|S_j|$, largest $\lambda_j$). The cardinality of $S_j$ increases with $j$, and some other jumps are found together with possible false detections (Fig. 9(b),(c)).

2) Model Order Selection: It may often be useful to select a single solution $\bm{x}_j$. The proposed algorithms are compatible with most methods of model order selection [46], [47] because they are greedy algorithms. Assuming that the variance of the observation noise is unknown, we distinguish two categories of cost functions for estimation of the order $|\bm{x}_j|_0 = |S_j|$. The first take the form $\min_j \left\{ m \log \mathcal{E}(S_j) + \alpha|S_j| \right\}$ where $\alpha$ equals 2, $\log m$, and $2 \log \log m$ for the Akaike, Minimum Description Length (MDL) and Hannan and Quinn criteria, respectively [46]. The second are cross-validation criteria [48], [49]. The sparse approximation framework allows one to derive simplified expressions of the latter up to the storage of intermediate solutions of greedy algorithms for consecutive cardinalities [8], [47], [50].

For the sparse deconvolution and jump detection problems, we found that the Akaike and cross validation criteria severely over-estimate the expected order. On the contrary, the MDL criterion yields quite accurate results. Specifically, the modified MDLc version dedicated to short data records (i.e., when the number of observations is moderately larger than the model order) [51] yields the best results for all the scenarios we have tested. It reads:

$$
\min_j \left\{ \log \mathcal{E}(S_j) + \frac{\log(m)(|S_j| + 1)}{m - |S_j| - 2} \right\}.
$$

(15)

Fig. 10 illustrates that the sparsity level found using MDLc is very accurate for high SNRs and underestimated for low SNRs. This behavior is relevant because for noisy data, the features related to the smallest coefficients $x_j^*$ are drowned in the noise. One cannot expect to detect them.

3) Further Empirical Observations: Fig. 11 is a typical display of the approximate $\ell_0$-curves yielded by CSBR and $\ell_0$-PD. The $\ell_0$-PD curve is structurally continuous and concave whereas in the CSBR curve, there are two kinds of breakpoints depicted with black and white circles. The former are
“continuous” breakpoints. This occurs when no single replacement is done during the call to SBR (SBR(\(S_{\text{init}}; \lambda \)) returns \(S_{\text{final}} = S_{\text{init}}\)). Otherwise, a discontinuity breakpoint appears (white circle). In Fig. 11, the \(\ell_0\)-PD and CSBR curves almost coincide for large \(\lambda\)’s. Discontinuity breakpoints appear in the latter curve for low \(\lambda\)’s.

Fig. 12 provides some insight on the CSBR and \(\ell_0\)-PD iterations for the sparse deconvolution problem with \(|x^*|_0 = 17\) and SNR = 20 dB. The CSBR subfigures show that 60 replacements have been performed from the initial empty support during the successive calls to SBR. For \(\ell_0\)-PD, at most two new supports are being included in the list of candidate subsets at each iteration. The number of actual single replacements is therefore increased by 0, 1 or 2. During the first 25 iterations, \(\ell_0\)-PD mainly operates atom selections similar to CSBR. The explored subsets are thus of increasing cardinalities and \(\lambda\) is decreasing (Fig. 12(c),(d)). From iterations 25 to 40, the very sparse solutions previously found are improved as a series of atom de-selections is performed. They are being improved again around iteration 75. On the contrary, the sparsest solutions are never improved with CSBR, which works for decreasing \(\lambda\)’s (Fig. 12(a),(b)). For \(\ell_0\)-PD, the early stopping parameter \(\lambda_{\text{step}}\) may have a strong influence on the improvement of the sparsest solutions and on the overall computation time. This point will be further discussed below.

C. Extensive Comparisons

The proposed algorithms are compared with popular nonconvex algorithms for both problems introduced in Section V-A with various parameter settings: problem dimension \((m, n)\), SNR, sparsity \(k = \|x^*\|_0\), and width \(\sigma\) of the Gaussian impulse response for the deconvolution problem. The settings are listed on Table V for 10 scenarios. Because CSBR and \(\ell_0\)-PD are orthogonal greedy algorithms, they are better suited to problems in which the level of sparsity is moderate to high. We therefore restrict ourselves to the case where \(k \leq 30\).

1) Competing Algorithms: We focus on the comparison with algorithms based on nonconvex penalties. It is indeed increasingly acknowledged that the BPDN estimates are less accurate than sparse approximation estimates based on nonconvex penalties. We do not consider forward greedy algorithms either; we already showed that SBR is (unsurprisingly) more effective than the simpler OMP and OLS algorithms [3]. Among the popular nonconvex algorithms, we consider:

1) Iterative Reweighted Least Squares (IRLS) for \(\ell_q\) minimization, \(q < 1\) [52];

2) Iterative Reweighted \(\ell_1\) (IR\(\ell_1\)) coupled with the sparsity measure \(\varphi(x) = \sum_i \log(1 + |x_i| + \gamma)\) where \(\gamma\) denotes the sparsity measure. The only algorithm directly working with \(\ell_0\) is L0LS-CD. We do not consider simpler thresholding algorithms (iterative hard thresholding, CoSaMP, subspace pursuit) proposed in the context of compressive sensing since we found that SBR behaves much better than these algorithms for ill-conditioned dictionaries [3]. We found that L0LS-CD is more effective than thresholding algorithms. Moreover, the cyclic descent approach is becoming very popular in the recent literature [44], [56] although its speed of convergence is sensitive to the initial solution accuracy. Here, we use the BPDN initial solution \(\arg \min_{\hat{x}} \{|y - A\hat{x}|_2^2 + \lambda \varphi(\hat{x})\}\) where \(\varphi\) denotes the sparsity measure. The only algorithm directly working with \(\ell_0\) is L0LS-CD. We do not consider simpler thresholding algorithms (iterative hard thresholding, CoSaMP, subspace pursuit) proposed in the context of compressive sensing since we found that SBR behaves much better than these algorithms for ill-conditioned dictionaries [3]. We found that L0LS-CD is more effective than thresholding algorithms. Moreover, the cyclic descent approach is becoming very popular in the recent literature [44], [56] although its speed of convergence is sensitive to the initial solution accuracy. Here, we use the BPDN initial solution
maximum tested $\lambda$-value (more details will be given hereafter). This simple ad hoc setting allows us to get a rough initial solution that is very sparse but nonzero within a fast computation time.

The three other considered algorithms work with sparsity measures depending on an arbitrary parameter. Regarding IRLS, we set $q=0.5$ or 0.1 as suggested in [52]. We chose to run IRLS twice, with $q=0.5$ and then $q=0.1$ (with the previous output at $q=0.5$ as initial solution) so that IRLS is less sensitive to local solutions at $q=0.1$. SL0 is a GNC-like algorithm working for increasingly non-convex penalties (Gaussian functions of decreasing widths). For simplicity reasons, the lowest width is set relative to the smallest nonzero coefficient of the ground truth solution $\mathbf{x}^*$. The basic SL0 implementation is dedicated to noise-free problems [43]. There exist several adaptations in the noisy setting [55], [57] including the precursory work [58]. We chose the efficient implementation of [57] in which the original pseudo-inverse calculations are replaced by a quasi-Newton strategy using limited memory BFGS updates. Finally, the $\text{IR} \ell_1$ implementation depends on both the choice of parameter $\gamma$ (which controls the degree of nonconvexity) and the $\ell_1$ solver. We have tested the in-crowd algorithm [59] together with an empirical setting of $\gamma > 0$, and $\ell_1$-homotopy in the limit case $\gamma \rightarrow 0$, following [53]. We found that the latter strategy is faster, mainly because the Matlab implementation of in-crowd makes calls to the $\text{quadprog}$ built-in function, which is computationally expensive.

2) Numerical Protocol: Because the competing algorithms work for a single $\lambda$-value, we need to define a grid, denoted by $\{\lambda_i^G, i = 1, \ldots, N_\lambda\}$, for comparison purposes. Such grid is defined in logscale for each of the 10 scenarios $(k, A, \text{SNR})$ defined in Table V. The number of grid points is $N_\lambda = 11$. For a given scenario, $T=30$ trials are being performed in which $k$-sparse vectors $\mathbf{x}^*(t)$ and noise vector $\mathbf{n}(t)$ are randomly drawn. This leads us to simulate $T$ observation vectors $\mathbf{y}(t) = A\mathbf{x}^*(t) + \mathbf{n}(t)$ with $t \in \{1, \ldots, T\}$. Specifically, the location of the nonzero coefficients in $\mathbf{x}^*(t)$ are uniformly distributed and the coefficient values are drawn according to an i.i.d. Gaussian distribution. For each trial $t$, all competing algorithms need to be run $N_\lambda$ times with $\mathbf{y}(t)$ and $\lambda_i^G$ as inputs whereas CSBR and $\ell_0$-PD are run only once since they deliver estimates for a continuum of values of $\lambda$. Their solution for each $\lambda_i^G$ directly deduces from their set of output supports and the knowledge of both breakpoints surrounding $\lambda_i^G$.

The algorithms are first evaluated in the optimization viewpoint: the related criteria are their capacity to reach a low value of $\mathcal{J}(\mathbf{x}; \lambda)$ and the corresponding CPU time. In this viewpoint, the proposed methods might be somehow favored since they are more directly designed with the criterion $\mathcal{J}(\mathbf{x}; \lambda)$ in mind. On the other hand, $\mathcal{J}(\mathbf{x}; \lambda)$ appears to be a natural indicator because solving either $\ell_0$-minimization problem (1), (2) or (3) is the ultimate goal of any sparse approximation method. As detailed below, some post-processing will be applied to the outputs of algorithms that do not rely on the $\ell_0$-norm so that they are not strongly disadvantaged. Practically, we store the value of $\mathcal{J}(\mathbf{x}; \lambda_i^G)$ found for each trial and each $\lambda_i^G$. Averaging this value over the trials $t$ yields a table $\text{Tab}\{\lambda_i^G\}$ where $\lambda_i^G$ denotes a candidate algorithm. Similarly, the CPU time is averaged over the trials, yielding another table $\text{TabCPU}\{\lambda_i^G\}$. Each table is represented separately as a 2D plot with a specific label for each algorithm: see, e.g., Fig. 13. CSBR and $\ell_0$-PD are represented with continuous curves because $\mathcal{J}(\mathbf{x}; \lambda)$ is computed for a continuum of $\lambda$’s, and the CPU time is computed only once.

The algorithms are also evaluated in terms of support recovery accuracy. For this purpose, let us first define the “support error” as the minimum over $i$ of the distance

$$S^*(t) \setminus S(t, a, \lambda_i^G) + |S(t, a, \lambda_i^G) \setminus S^*(t)|$$ (16)
between the ground truth support $S^*(t)$ (i.e., the support of $\mathbf{z}^*(t)$) and the support $S(t, a, \lambda^G_t)$ of the sparse reconstruction at $\lambda^G_t$ with algorithm $a$. In (16), both numbers of false negatives $|S^*(t) \setminus S(t, a, \lambda^G_t)|$ and of false positives $|S(t, a, \lambda^G_t) \setminus S^*(t)|$ are taken into account. Denoting by $S(t, a, \lambda^G_{opt}) \leftarrow S(t, a, \lambda^G_t)$ the solution support that is the closest to $S^*(t)$ according to (16), we further consider the number of true positives in $S(t, a, \lambda^G_{opt})$, defined as $|S^*(t) \cap S(t, a, \lambda^G_{opt})|$ and the model order $S(t, a, \lambda^G_{opt})$. We will finally report:

- the average support error score $SE(a)$;
- the corresponding number of true positives $TP(a)$;
- the corresponding model order $Order(a)$;

which are average measures over $T$ trials. The false positive (FP) and true/false negatives scores can be directly deduced, e.g., $FP(a) = Order(a) - TP(a)$.

The underlying idea in this analysis is that when $SE$ is small (respectively, $TP$ is high), the algorithms are likely to perform well provided that $\lambda$ is appropriately chosen. However, in practical applications, only one estimate is selected using a suitable model selection criterion. We therefore provide additional evaluation of the MDLc estimate accuracy. For CSBR and $\ell_0$-PD, all output supports are considered to compute the MDLc estimate whereas for other algorithms, it is equal to one of the sparse reconstructions obtained at $\lambda^G_i$ for $i \in \{1, \ldots, N_\lambda\}$. The same three measures as above are computed and averaged over $T$ trials. They are denoted by $MDLc$-$SE(a)$, $MDLc$-$TP(a)$ and $MDLc$-$Order(a)$.

3) Technical Adaptations for Comparison Purposes: Because IRLS and SL0 do not deliver sparse vectors in the strict sense, it is necessary to sparsify their outputs before computing their $SE(a)$ and $TP(a)$ scores. This is done by running one iteration of L0LS-CD. Regarding the values of $J(x; \lambda)$, a post-processing (local descent of $J(x; \lambda)$) is performed for algorithms that do not rely on the $\ell_0$-norm: run one iteration of L0LS-CD, and then compute the least-square error related to the output support.

4) Analysis in the Optimization Viewpoint: As illustrated in Figs. 13 – 15, CSBR and $\ell_0$-PD are always among the most accurate to minimize the cost function. We can clearly distinguish two groups of algorithms on these figures: IRLS, L0LS-CD and SL0 one the one hand, and the OLS-based algorithms (SBR, CSBR, $\ell_0$-PD) and IR$\ell_1$, on the other hand, which are the most accurate. We cannot discriminate the accuracy of SBR and CSBR: one may behave slightly better than the other depending on the scenarios. On the contrary, SBR and CSBR are often outperformed by $\ell_0$-PD. The obvious advantage of CSBR and $\ell_0$-PD over SBR and IR$\ell_1$ is that they deliver some solutions for many sparsity levels, and the corresponding $\lambda$-values are adaptively found. On the contrary, the SBR output is related to a single $\lambda$ whose tuning may be tricky. Another advantage over IR$\ell_1$ is that the structure of forward-backward algorithms is simpler, as no call to any $\ell_1$ solver is required. Moreover, the number of parameters to tune is lower: there is a single (early) stopping parameter $\lambda_{stop}$.

The price to pay for a better performance is an increase of the computation time. On Figs. 13 – 15, two lines are drawn for CSBR (respectively, for $\ell_0$-PD). They are horizontal because the algorithm is run only once per trial, so there is a single computation time measurement. The first line corresponds to the overall computation time, i.e., from the start to the termination of CSBR/$\ell_0$-PD. This time is often more expensive than for other algorithms because the latter times refer to a single execution for some $\lambda^G_t$ value. This is the reason why a second horizontal line is drawn, corresponding to the average computation time (normalization by $N_\lambda - 11$ of the overall computation time). Now, the average CPU times of CSBR and $\ell_0$-PD are very reasonable.

The computation time depends on many factors among which the implementation of algorithms (including the memory storage) and the chosen stopping rules. We have followed an homogeneous implementation of the competing algorithms, and defined two sets of stopping rules depending on the problem dimension. The default parameters apply to medium size problems ($m = 300$). They are relaxed for problems of larger dimension ($m \geq 500$) to avoid huge computational costs. The stopping rule of CSBR and $\ell_0$-PD is $\lambda \leq \lambda_{stop} = \alpha \min\{\lambda^G_i\}$ with $\alpha = 1$ for CSBR and $\alpha = 0.5$ (medium size) or 0.8 (large size) for $\ell_0$-PD. The maximum number of cyclic descents in L0LS-CD is set to 60 or 10 depending on the dimension. For SL0, we have followed the default setting of [43] for the rate of
TABLE VI
JUMP DETECTION PROBLEM IN THE NOISY SETTING: THE ALGORITHMS ARE COMPARED IN TERMS OF SUPPORT ERROR (SE) AND NUMBER OF TRUE POSITIVES (TP). THE NUMBER OF JUMPS THAT ARE FOUND IS REPORTED (ORDER) TOGETHER WITH THE "TRUE ORDER" CORRESPONDING TO THE GROUND TRUTH \( r \). THE SCORES RELATED TO THE MDLc ESTIMATE ARE INDICATED SIMILARLY.

| Scenario | \( \ell_p \)-PD | CSBR | SBR | LOLS-CD | SL0 | IR\( \ell_1 \) | IRLS |
|----------|----------------|------|-----|--------|-----|-----------|------|
| SE       | 1.6            | 1.6  | 1.6 | 5.3    | 4.0 | 1.5       | 1.8  |
| TP       | 8.6            | 8.7  | 8.6 | 5.2    | 7.8 | 8.7       | 8.7  |
| Order true (10) | 8.8 | 9.0  | 8.9 | 5.7    | 9.6 | 8.9       | 9.1  |
| MDLc-SE | 4.7            | 4.3  | 4.1 | 22.7   | 5.6 | 4.1       | 3.5  |
| MDLc-TP | 8.7            | 8.8  | 8.8 | 6.9    | 8.6 | 8.8       | 8.8  |
| MDLc-Order | 12.2 | 11.9 | 11.6 | 26.6  | 12.7 | 11.7     | 11.0 |
| Scenario F |                |      |     |        |     |           |      |
| SE       | 11.1           | 11.9 | 11.8 | 22.5   | 11.6 | 10.9     | 11.6 |
| TP       | 21.2           | 20.6 | 20.7 | 9.2    | 20.3 | 20.8     | 20.7 |
| Order true (30) | 23.6 | 23.2 | 23.2 | 10.9  | 22.2 | 22.5     | 23.1 |
| MDLc-SE | 13.7           | 13.4 | 13.4 | 39.2   | 14.0 | 13.1     | 13.3 |
| MDLc-TP | 21.8           | 21.8 | 21.4 | 12.9   | 21.6 | 22.1     | 21.5 |
| MDLc-Order | 27.3 | 27.0 | 26.3 | 35.0  | 27.2 | 27.2     | 26.4 |
| Scenario G |                |      |     |        |     |           |      |
| SE       | 7.3            | 7.5  | 7.5 | 8.9    | 10.3 | 7.2       | 7.5  |
| TP       | 4.0            | 3.6  | 3.6 | 3.1    | 2.9  | 3.9       | 4.0  |
| Order true (10) | 5.2 | 4.7  | 4.7 | 5.2    | 6.1  | 5.0       | 5.6  |
| MDLc-SE | 11.4           | 10.7 | 10.9 | 11.7   | 15.1 | 11.2     | 10.7 |
| MDLc-TP | 4.2            | 4.2  | 4.2 | 3.0    | 3.9  | 4.2       | 4.4  |
| MDLc-Order | 9.8  | 9.1  | 9.3 | 7.6    | 12.8 | 9.6       | 9.5  |

TABLE VII
SPARSE DECONVOLUTION PROBLEM IN THE NOISE-FREE SETTING: EXACT SUPPORT RECOVERY

| Scenario H | \( \ell_p \)-PD | CSBR | SBR | LOLS-CD | SL0 | IR\( \ell_1 \) | IRLS |
|------------|----------------|------|-----|--------|-----|-----------|------|
| SE         | 2.5            | 3.6  | 4.8 | 11.4   | 13.0 | 9.4       | 6.1  |
| TP         | 8.3            | 8.2  | 6.8 | 0.4    | 0.1  | 9.5       | 9.4  |
| Order true (10) | 9.1 | 10.0 | 8.3 | 2.2    | 3.2  | 9.8       | 14.9 |
| MDLc-SE   | 3.8            | 3.8  | 5.8 | 168.5  | 343.8 | 1.1       | 9.0  |
| MDLc-TP   | 8.6            | 8.6  | 7.9 | 3.3    | 6.6  | 9.5       | 9.6  |
| MDLc-Order | 10.8           | 11.0 | 11.6 | 153.5 | 347.0 | 10.1      | 18.2 |

| Scenario I | \( \ell_p \)-PD | CSBR | SBR | LOLS-CD | SL0 | IR\( \ell_1 \) | IRLS |
|------------|----------------|------|-----|--------|-----|-----------|------|
| SE         | 0.9            | 1.3  | 2.1 | 36.7   | 48.5 | 3.8       | 9.4  |
| TP         | 29.4           | 29.3 | 29.1 | 0.7    | 0.8  | 28.0      | 27.7 |
| Order true (30) | 29.7 | 29.8 | 30.2 | 8.2    | 20.1 | 29.8      | 34.8 |
| MDLc-SE   | 3.5            | 3.5  | 3.7 | 686.0  | 444.9 | 9.5       | 114.3|
| MDLc-TP   | 29.5           | 29.4 | 29.2 | 28.6   | 17.5 | 28.5      | 26.4 |
| MDLc-Order | 32.8           | 32.3 | 32.1 | 437.0  | 449.8 | 36.5      | 137.2|

| Scenario J | \( \ell_p \)-PD | CSBR | SBR | LOLS-CD | SL0 | IR\( \ell_1 \) | IRLS |
|------------|----------------|------|-----|--------|-----|-----------|------|
| SE         | 0.3            | 3.5  | 5.3 | 10.3   | 10.4 | 2.4       | 4.3  |
| TP         | 9.8            | 7.3  | 5.8 | 0.6    | 2.6  | 8.8       | 9.2  |
| Order true (10) | 9.8 | 8.1  | 6.9 | 1.4    | 5.6  | 10.0      | 12.7 |
| MDLc-SE   | 2.6            | 7.7  | 12.4 | 176.2  | 78.6 | 7.2       | 69.0 |
| MDLc-TP   | 9.7            | 8.9  | 8.0 | 8.6    | 3.0  | 8.9       | 4.1  |
| MDLc-Order | 12.0           | 15.5 | 18.5 | 73.0   | 74.6 | 14.9      | 67.2 |

Deformation of the nonconvex penalty. The number of BFGS iterations done in the local minimization steps for each penalty is set to \( I = 40 \) or 5. It is set to 5 for the last penalty which is the most nonconvex. Regarding IRLS and IR\( \ell_1 \), we keep the same settings whatever the dimension since the computation times remain reasonable. Finally, SBR does not require any arbitrary stopping rule. For the problems of large dimensions (scenarios C and D), we observe on Fig. 13 that the comparison (trade-off performance vs computation time) is now clearly in favor of CSBR and \( \ell_p \)-PD. IR\( \ell_1 \) remains very competitive although the average numerical cost becomes larger.

5) Analysis in the Support Recovery Viewpoint: The support recovery performance is only shown for the scenarios E to J (Tables VI and VII). These results are omitted for noisy deconvolution problems because the support error is often quite large and the true positive scores are low whatever the algorithm, especially for scenarios B to D. Specifically, SE always exceeds 20, 10, 10 and 32 for the scenarios A to D (\( k = 30, 10, 10 \) and 30, respectively). For such difficult problems, one can hardly discriminate algorithms based on simple binary tests such as the TP rate. More sophisticated localization tests are nonbinary and would take into account the distance between the location of the true spikes and their wrong estimates [60]. It is noticeable, though, that the MDLc estimator delivers subsets of realistic cardinality for scenarios A to D (e.g., the subsets found with CSBR are of cardinalities 33, 9, 15 and 38). The model orders are also quite accurate for the noisy jump detection problem although the true support is often partially detected by several of the considered algorithms (Table VI). Here, CSBR and \( \ell_1 \)-PD are among the best algorithms in terms of support error.

The results of Table VII and Fig. 15 correspond to the noise-free deconvolution problem. The data are undersampled so that the dictionary \( \Delta \approx m/n \) is set to 2 in scenarios H and I and 4 in scenario J. Again, CSBR and \( \ell_p \)-PD are among the best (SE, TP, MDLc-order) especially for the most difficult scenario J.
6) Overcomplete Dictionaries With Noise: We now argue that the proposed algorithms are competitive as well for noisy problems with overcomplete dictionaries. The experiments commented below are not reported for space reasons.

We have first considered the noisy deconvolution problem with \( \Delta = 2 \) or 4, the other parameters being set as in scenarios A to D. Although the data approximation is qualitatively good for CSBR and \( \ell_0 \)-PD, the SE and TP scores are very weak. It is hard to compare algorithms because these measures are very weak for all considered algorithms. Moreover, the values of \( J(\lambda) \) are often similar.

We also considered some adaptive spline approximation problem generalizing the jump detection problem to the approximation of a piecewise polynomial of degree \( P = 1 \) or 2 [3]. The jump detection problem can indeed be thought of as the approximation of a piecewise constant signal \((P = 0)\). The generalized version is inspired from the regression spline modeling in [61]. Now, the dictionary atoms are related to the detection of the locations of jumps, changes of slopes and changes of curvatures in the signal \( y \) (subdictionaries \( A^0, A^1 \) and \( A^2 \)). The dictionary then takes the form \( \mathcal{A} = \mathcal{A}^0, \mathcal{A}^1, \mathcal{A}^2 \) or \( \mathcal{A} = \mathcal{A}^0, \mathcal{A}^1, \mathcal{A}^2 \) where each sub-dictionary \( \mathcal{A}^p (p \leq P) \) is formed of shifted versions of the one-sided power function \( \max(t, 0)^p \). The size of the full dictionary \( A \) is approximately \( m \times (P + 1)m \). Hence, it becomes overcomplete as soon as \( P \geq 1 \). We showed [3] that SBR is competitive when \( P = 1 \) or 2. We have carried out new tests confirming that CSBR and \( \ell_0 \)-PD are more effective than their competitors in terms of values of \( J(\lambda) \). However, the true positives rates are low for \( P \geq 1 \) since the location of the change of slopes and of curvatures can hardly be exactly recovered from noisy data.

VI. CONCLUSION

The choice of a relevant algorithm relies on a trade-off between the desired accuracy and the computation time. The proposed algorithms are relatively expensive but very well suited to inverse problems inducing highly correlated dictionaries. A reason is that they have the capacity to escape from some local minimizers of \( J(\mathbf{x}; \lambda) = \| \mathbf{y} - \mathcal{A} \mathbf{x} \|_2^2 + \lambda \| \mathbf{x} \|_0 \) [3]. This behavior is in contrast with other classical sparse algorithms.

We have shown the effectiveness of both SBR extensions when the level of sparsity is moderate to high, i.e., \( k / \min(m, n) \) is lower than 0.1. They remain competitive when \( k / \min(m, n) \) ranges between 0.1 and 0.2, and their performance gradually degrade for weaker levels of sparsity, which is an expected behavior for such greedy type algorithms. For a single \( \lambda \), CSBR is as effective as SBR, and \( \ell_0 \)-PD improves the SBR and CSBR performance within a larger computation cost. The main benefit over SBR is that sparse solutions are provided for a continuum of \( \lambda \)-values, enabling the utilization of any classical order selection method. We found that the MDL criterion yields very accurate estimates.

Our perspectives include the proposal of forward-backward search algorithms that will be faster than SBR and possibly more effective. In the standard version of SBR, CSBR and \( \ell_0 \)-PD, a single replacement refers to the selection or de-selection of a dictionary element. The cost of an iteration is essentially related to the \( n \) linear system resolutions done to test single replacements for all dictionary atoms. The proposed algorithms obviously remain valid when working with a larger neighborhood, e.g., when testing the replacement of two atoms simultaneously, but their complexity becomes huge. To avoid such numerical explosion, one may rather choose not to carry out all the replacement tests, but only some tests that are likely to be effective. Extensions of OMP and OLS were recently proposed in this spirit [36] and deserve consideration for proposing efficient forward-backward algorithms.

APPENDIX

PROPERTIES OF THE \( \ell_0 \) REGULARIZATION PATHS

We prove that the \( \ell_0 \)-penalized path \( \mathcal{S}_0^p \) is piecewise constant (Theorem 1) and included in the \( \ell_1 \)-constrained path \( \mathcal{S}^p_1 \) (Theorem 2). The \( \ell_0 \)-curve will be denoted by \( \lambda \rightarrow J(\lambda) = \min_{\mathcal{S}} J(S; \lambda) \). Let us recall that this function is concave and affine on each interval \((\lambda^*_i, +\infty), i \in \{0, \ldots, I\} \) (Definition 1). Moreover, \( \lambda^*_I = 0 \) and \( \lambda^*_0 = +\infty \).

A. Proof of Theorem 1

We prove Theorem 1 together with the following lemma, which is informative about the content of \( \mathcal{S}_0^p(\lambda) \) for the breakpoints \( \lambda = \lambda^*_i \).

Lemma 2: Let \( i \in \{1, \ldots, I - 1\} \). Then, for all \( \lambda \in (\lambda^*_i, \lambda^*_{i+1}) \), \( \mathcal{S}_0^p(\lambda) \subset \mathcal{S}^p_1(\lambda^*_i, \lambda^*_{i+1}) \cap \mathcal{S}^0_p(\lambda) \).

For the first and last intervals, we have:

- For all \( \lambda \in (0, \lambda^*_1) \), \( \mathcal{S}_0^p(\lambda) \subset \mathcal{S}^0_p(\lambda^*_1) \).
- For all \( \lambda \in (\lambda^*_I, +\infty) \), \( \mathcal{S}_0^p(\lambda) \subset \mathcal{S}^p_1(\lambda^*_I) \).

Proof of Theorem 1: By definition, the \( \ell_0 \)-curve is the concave envelope of the (finite) set of lines \( S \) for all possible subsets \( S \). Because it is affine on the \( i \)-th interval \((\lambda^*_i, \lambda^*_i + 1) \), \( J(\lambda) \) coincides with \( J(S_i; \lambda) = \varepsilon(S_i) + \lambda S_i \), where \( S_i \) is some optimal subset for all \( \lambda \in (\lambda^*_i, \lambda^*_i + 1) \).

Let \( \lambda \in (\lambda^*_i + 1, \lambda^*_{i+1}) \) and \( S \in \mathcal{S}_0^p(\lambda) \). Then, \( J(S; \lambda) = J(S_i; \lambda) \). It follows that both lines \( S \) and \( S_i \) necessarily coincide; otherwise, they would intersect at \( \lambda \), and line \( S \) would lay below \( S_i \) on either interval \((\lambda^*_i, \lambda^*_i + 1) \) or \((\lambda^*_i + 1, \lambda^*_i) \), which contradicts the definition of \( S_i \). We conclude that \( S \in \mathcal{S}^p_1(\lambda^*_i) \) for all \( \lambda \in (\lambda^*_i + 1, \lambda^*_i) \).

We have shown that the content of \( \mathcal{S}_0^p(\lambda) \) does not depend on \( \lambda \) when \( \lambda \in (\lambda^*_i + 1, \lambda^*_i) \).

B. Proof of Theorem 2

The first result is straightforward: for any \( \lambda \) and \( S \in \mathcal{S}_0^p(\lambda) \), we have \( S \in \mathcal{S}^p_1(\lambda) \). Otherwise, there would exist \( S' \) with \( \lambda \geq \lambda^*_i \) and \( \mathcal{E}(S') < \mathcal{E}(S) \). Then, \( J(S'; \lambda) < J(S; \lambda) \) would contradict \( S \in \mathcal{S}_0^p(\lambda) \).
To prove the second result, let us first show that for any \( i, \quad k_i : \forall \lambda \in (\lambda_{i+1}^{*}, \lambda_{i}^{*}), \quad S_{\lambda}^{*}(\lambda) \subset S_{k_i}^{*}(k_i).

Let \( S \in S_{\lambda}^{*}(\lambda) \) for some \( \lambda \in (\lambda_{i+1}^{*}, \lambda_{i}^{*}) \). Theorem 1 implies that \( S \in S_{\lambda}^{*}(\lambda) \) for any \( \lambda \in (\lambda_{i+1}^{*}, \lambda_{i}^{*}) \). Therefore, \( J'(\lambda) = J'(S; \lambda) \) for \( \lambda \in (\lambda_{i+1}^{*}, \lambda_{i}^{*}) \), and the slope of line \( S \), i.e., \( J' \), is constant whatever \( S \in S_{\lambda}^{*}(\lambda) \) and \( \lambda \in (\lambda_{i+1}^{*}, \lambda_{i}^{*}) \). Let us denote this constant by \( k_i = |S| \). According to the first paragraph of the proof, \( S \in S_{\lambda}^{*}(\lambda) \) implies that \( S \in S_{k_i}^{*}(k_i) \).

Let us prove the reverse inclusion \( S_{k_i}^{*}(k_i) \subset S_{\lambda}^{*}(\lambda) \). Let \( \lambda \in (\lambda_{i+1}^{*}, \lambda_{i}^{*}) \) and \( S \in S_{k_i}^{*}(k_i) \). First, we have \( S \leq k_i \). Second, for any \( S' \in S_{\lambda}^{*}(\lambda) \), we have \( S' = k_i \) by definition of \( k_i \). We also have that \( \mathcal{E}(S') - \mathcal{E}(S) \) because \( S_{\lambda}^{*}(\lambda) \subset S_{k_i}^{*}(k_i) \). Finally, \( J'(S'; \lambda) \geq J'(S; \lambda) \). \( S' \in S_{\lambda}^{*}(\lambda) \) implies that \( S \in S_{\lambda}^{*}(\lambda) \). This concludes the proof of the second result.

Acknowledgment

The authors would like to thank the anonymous reviewers and the Associate Editor Dr. J. Jaldén for their constructive remarks that significantly improved the manuscript.

References

[1] B. K. Natarajan, “Sparse approximate solutions to linear systems,” SIAM J. Comput., vol. 24, no. 2, pp. 227–234, Apr. 1995.
[2] M. Nikolova, “Description of the minimizers of least squares regularized with \( \ell_q \) norm. Uniqueness of the global minimizer,” SIAM J. Imag. Sci., vol. 6, no. 2, pp. 904–937, May 2013.
[3] C. Soussen, J. Idier, D. Brie, and J. Duan, “From Bernoulli-Gaussian deconvolution to sparse signal restoration,” IEEE Trans. Signal Process., vol. 59, no. 10, pp. 5178–5192, Oct. 2011.
[4] J. A. Tropp and S. J. Wright, “Computational methods for sparse solutions of linear inverse problems,” Proc. IEEE (Special Issue on Applications of Sparse Representation and Compressive Sensing), vol. 98, no. 5, pp. 948–958, Jun. 2010, Invited Paper.
[5] S. Mallat and Z. Zhang, “Matching pursuits with time-frequency dictionaries,” IEEE Trans. Signal Process., vol. 41, no. 12, pp. 3397–3415, Dec. 1993.
[6] Y. C. Pati, R. Rezaifar, and P. S. Krishnaprasad, “Orthogonal Matching Pursuit: Recursive function approximation with applications to wavelet decomposition,” in Proc. 27th Asilomar Conf. Signals, Syst., Comput., Nov. 1993, vol. 1, pp. 40–44.
[7] S. Chen, S. A. Billings, and W. Luo, “Orthogonal least squares methods and their application to non-linear system identification,” Int. J. Control, vol. 50, no. 5, pp. 1873–1896, Nov. 1989.
[8] A. J. Miller, Subset Selection in Regression, 2nd ed. London, U.K.: Chapman & Hall, 2002.
[9] S. F. Cotter, J. Adler, B. D. Rao, and K. Kreutz-Delgado, “Forward sequential algorithms for best Basis selection,” Proc. Inst. Electr. Eng.—Vision, Image, Signal Process., vol. 146, no. 5, pp. 235–244, Oct. 1999.
[10] J. R. Statist. Soc. C
[11] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani, “Least angle regression,” Ann. Statist., vol. 32, no. 2, pp. 407–499, Apr. 2004.
[12] I. Das and J. E. Dennis, “A closer look at drawbacks of minimizing weighted sums of objectives for Pareto set generation in multicriteria optimization problems,” Struct. Optim., vol. 14, no. 1, pp. 63–69, Aug. 2007.
[13] B. R. Marler and J. S. Arora, “Survey of multi-objective optimization methods for engineering,” Struct. Multidiscip. Optim., vol. 26, no. 6, pp. 369–395, Apr. 2004.
[14] E. van den Berg and M. P. Friedlander, “Probing the Pareto frontier for basis pursuit solutions,” SIAM J. Sci. Comput., vol. 31, no. 2, pp. 896–912, Nov. 2008.
[15] P. M. T. Broersen, “Subset regression with stepwise directed search,” J. R. Statist. Soc. C
[16] T. Zhang, “Adaptive forward-backward greedy algorithm for learning sparse representations,” IEEE Trans. Inf. Theory, vol. 57, no. 7, pp. 4689–4708, Jul. 2011.
[42] J. Trzasko and A. Manduca, “Highly undersampled magnetic resonance image reconstruction via homotopic $\ell_2$-minimization,” *IEEE Trans. Med. Imag.*, vol. 8, no. 1, pp. 106–121, Jan. 2009.

[43] G. H. Mohimani, M. Babaie-Zadeh, and C. Jutten, “A fast approach for overcomplete sparse decomposition based on smoothed $\ell^n$ norm,” *IEEE Trans. Signal Process.*, vol. 57, no. 1, pp. 289–301, Jan. 2009.

[44] R. Mazumder, J. H. Friedman, and T. Hastie, “SparseNet: Coordinate descent with nonconvex penalties,” *J. Acoust. Soc. Amer.*, vol. 106, no. 495, pp. 1125–1138, Sep. 2011.

[45] D. L. Donoho, V. Stodden, and Y. Tsaig, “About SparseLab,” Stanford Univ., Stanford, CA, USA, Tech. Rep., 2007.

[46] P. Stoica and Y. Selén, “Model-order selection: A review of information criterion rules,” *IEEE Sig. Proc. Mag.*, vol. 21, no. 4, pp. 36–47, Jul. 2004.

[47] Y. Wang, “Model selection,” in *Handbook of Computational Statistics*, J. E. Gentle, W. Härdle, and Y. Mori, Eds. Berlin, Germany: Springer, 2004, vol. 1, pp. 437–466.

[48] G. Wahba, “Practical approximate solutions to linear operator equations when the data are noisy,” *SIAM J. Numer. Anal.*, vol. 14, no. 4, pp. 651–667, 1977.

[49] G. H. Golub, M. Heath, and G. Wahba, “Generalized cross-validation as a method for choosing a good ridge parameter,” *Technometrics*, vol. 21, no. 2, pp. 215–223, May 1979.

[50] C. D. Austin, R. L. Moses, J. N. Ash, and E. Ertin, “On the relation between sparse reconstruction and parameter estimation with model order selection,” *IEEE J. Sel. Topics Signal Process.*, vol. 4, no. 3, pp. 298–309, Jun. 2010.

[51] F. de Ridder, R. Pintelon, J. Schoukens, and D. P. Gillikin, “Modified AIC and MDL model selection criteria for short data records,” *IEEE Trans. Instrum. Meas.*, vol. 54, no. 1, pp. 144–150, Feb. 2005.

[52] M.-J. Lai, Y. Xu, and W. Yin, “Improved iteratively reweighted least squares for unconstrained smoothed $\ell_q$ minimization,” *SIAM J. Numer. Anal.*, vol. 51, no. 2, pp. 927–957, Mar. 2013.

[53] H. Zou, “The adaptive Lasso and its oracle properties,” *J. Acoust. Soc. Amer.*, vol. 101, no. 476, pp. 1418–1429, Dec. 2006.

[54] A. J. Seneviratne and V. Solo, “Sparse coloured system identification with guaranteed stability,” in Proc. *IEEE Conf. Decision Control*, Honolulu, HI, USA, Dec. 2012, pp. 2826–2831.

[55] A. Eftekhari, M. Babaie-Zadeh, C. Jutten, and H. A. Moghaddam, “Robust-SL0 for stable sparse representation in noisy settings,” in *Proc. IEEE ICASSP*, Taipei, Taiwan, Apr. 2009, pp. 3433–3436.

[56] G. Marjanovic and V. Solo, “$\ell_q$ sparsity penalized linear regression with cyclic descent,” *IEEE Trans. Signal Process.*, vol. 62, no. 6, pp. 1464–1475, Mar. 2014.

[57] X. Ye, W.-P. Zhu, A. Zhang, and J. Yan, “Sparse channel estimation of MIMO-OFDM systems with unconstrained smoothed $\ell_q$-norm-regularized least squares compressed sensing,” *EURASIP J. Wireless Commun. Netw.*, vol. 2013, no. 282, pp. 1–13, Dec. 2013.

[58] N. Saito, “Superresolution of noisy band-limited data by data adaptive regularization and its application to seismic trace inversion,” in Proc. *IEEE ICASSP*, Albuquerque, NM, USA, Apr. 1990, pp. 1237–1240.

[59] P. R. Gill, A. Wang, and A. Molnar, “The in-crowd algorithm for fast basis pursuit denoising,” *IEEE Trans. Signal Process.*, vol. 59, no. 10, pp. 4595–4605, Oct. 2011.

[60] M. C. van Rossum, “A novel spike distance,” *Neural Comput.*, vol. 13, no. 4, pp. 751–763, Apr. 2001.

[61] J. H. Friedman, “Multivariate adaptive regression splines,” *Ann. Statist.*, vol. 19, no. 1, pp. 67, Mar. 1991.

Charles Soussen (M’12) was born in France in 1972. He graduated from the École Nationale Supérieure en Informatique et Mathématiques Appliquées, Grenoble, France in 1996. He received the Ph.D. degree in Physics from Université de Paris-Sud, Orsay, France, in 2000 and his Habilitation à Diriger des Recherches in signal processing from Université de Lorraine, France, in 2013. He is currently an Associate Professor at Université de Lorraine. He has been with the Centre de Recherche en Automatique de Nancy since 2005. His research interests are in inverse problems and sparse approximation.

Jérôme Idier (M’09) was born in France in 1966. He received the Diploma degree in electrical engineering from École Supérieure d’Electricité, Gif-sur-Yvette, France, in 1988, and the Ph.D. degree in physics from University of Paris-Sud, Orsay, France, in 1991. In 1991, he joined the Centre National de la Recherche Scientifique. He is currently a Senior Researcher with the Institut de Recherche en Communications et Cybernétique, Nantes, France. His major scientific interests are in probabilistic approaches to inverse problems for signal and image processing. Dr. Idier is currently an elected member of the French National Committee for Scientific Research.

Junbo Duan received the B.S. degree in information engineering and the M.S. degree in communication and information system from Xi’an Jiaotong University, Xi’an, China, in 2004 and 2007, respectively, and the Ph.D. degree in signal processing from Université Henry Poincaré, Nancy, France, in 2010. From 2011 to 2013, he was a postdoctoral fellow in the Department of Biomedical Engineering and the Department of Biostatistics and Bioinformatics, Tu- lane University, New Orleans, USA. He is currently an Assistant Professor in the Department of Biomedical Engineering, Xi’an Jiaotong University. His research interests are in inverse problems in biomedical engineering and bioinformatics.

David Brie (M’13) received the Ph.D. and Habilitation à Diriger des Recherches degrees from Université de Lorraine, France, in 1992 and 2000, respectively. Since 1990, he has been with the Centre de Recherche en Automatique de Nancy, Université de Lorraine. He has been a Full Professor, Université de Lorraine, since 2001. His research interests include statistical signal processing, inverse problems, and multidimensional signal processing. In 2013, he was a Visiting Researcher with the Department of Electronic and Information Engineering, Polytechnic University of Hong Kong. Since 2014, he has served as the Editor-in-Chief of the French journal *Traitement du Signal.*