# Homotopy based algorithms for L0-regularized least-squares

Charles Soussen, Jérome Idier, Junbo Duan, David Brie

# Homotopy based algorithms for $\ell_0$-regularized least-squares

Charles Soussen*, Jérôme Idier, Junbo Duan, and David Brie

**Abstract**

Sparse signal restoration is usually formulated as the minimization of a quadratic cost function $\|\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}\|_2^2$, where $\boldsymbol{A}$ is a dictionary and $\boldsymbol{x}$ 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 efficient $\ell_1$ solvers, the homotopy algorithm minimizes $\|\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}\|_2^2+\lambda\|\boldsymbol{x}\|_1$ with respect to $\boldsymbol{x}$ 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 $\|\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}\|_2^2+\lambda\|\boldsymbol{x}\|_0$ 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.

This work was carried out in part while C. Soussen was visiting IRCCyN during the academic year 2010-2011 with the financial support of CNRS.

C. Soussen and D. Brie are with the Université de Lorraine and CNRS at the Centre de Recherche en Automatique de Nancy (UMR 7039).Campus Sciences, B.P. 70239, F-54506 Vandœuvre-lès-Nancy, France. Tel: (+33)-3 83 59 56 43, Fax: (+33)-3 83 68 44 62. 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), 1 rue de la Noë, BP 92101, F-44321 Nantes Cedex 3, France. Tel: (+33)-2 40 37 69 09, Fax: (+33)-2 40 37 69 30. E-mail: jerome.idier@irccyn.ec-nantes.fr.

J. Duan was with CRAN. He is now with the Department of Biomedical Engineering, Xi'an Jiaotong University. No. 28, Xianning West Road, Xi'an 710049, Shaanxi Province, China. Tel: (+86)-29-82 66 86 68, Fax: (+86)-29 82 66 76 67. E-mail: junbo.duan@mail.xjtu.edu.cn.

**Index Terms**

Sparse signal estimation; $\ell_0$-regularized least-squares; $\ell_0$-homotopy; $\ell_1$-homotopy; stepwise algorithms; orthogonal least squares; model order selection.

## I. INTRODUCTION

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

$$\min_{\boldsymbol{x}} \|\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}\|_2^2 \;\text{ subject to }\; \|\boldsymbol{x}\|_0 \leq k \tag{1}$$

or

$$\min_{\boldsymbol{x}} \|\boldsymbol{x}\|_0 \;\text{ subject to }\; \|\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}\|_2^2 \leq \varepsilon \tag{2}$$

where $\|\boldsymbol{x}\|_0$ is the $\ell_0$-"norm" counting the number of nonzero entries in $\boldsymbol{x}$, and the quadratic fidelity-to-data term $\|\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}\|_2^2$ 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 $\boldsymbol{A}$. On the contrary, the choice of (2) is more appropriate when $k$ is unknown but one has a knowledge of the variance of the observation noise. The value of $\varepsilon$ 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

$$\min_{\boldsymbol{x}} \{\mathcal{J}(\boldsymbol{x};\lambda) = \|\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}\|_2^2 + \lambda\|\boldsymbol{x}\|_0\} \tag{3}$$

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 $\|\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}\|_2^2$ and the penalty $\|\boldsymbol{x}\|_0$ are respectively related to a Gaussian noise distribution and a prior distribution for sparse signals (a limit Bernoulli-Gaussian distribution with infinite Gaussian variance) [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, the 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]. Greedy algorithms include, by increasing order of complexity: 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 *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) $\ell_0$-penalized least-squares:* In [3], we evidenced that the minimization of $\mathcal{J}(\boldsymbol{x};\lambda)$ 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 $\boldsymbol{x}$. Including a new element into $S$ yields a decrease of the square error, defined as the minimum of $\|\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}\|_2^2$ for $\boldsymbol{x}$ supported by $S$. On the other hand, the penalty term $\lambda\|\boldsymbol{x}\|_0$ is increased by $\lambda$. Overall, the cost function $\mathcal{J}(\boldsymbol{x};\lambda)$ decreases as soon as the square error variation exceeds $\lambda$. Similarly, a decrease of $\mathcal{J}(\boldsymbol{x};\lambda)$ occurs when an element is removed from $S$ provided that the squared error increment is lower than $\lambda$. Because both inclusion and removal operations can induce a decrease of $\mathcal{J}$, 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) Connection with the 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 nondifferentiable at 0, and optimizing the resulting cost function. See, *e.g.,* [18], [19] and [20]–[27] for convex ($\ell_1$) and nonconvex relaxation, respectively. The convex problem $\min_{\boldsymbol{x}}\|\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}\|_2^2$ s.t. $\|\boldsymbol{x}\|_1 \leq t$ is referred to as both Basis Pursuit Denoising (BPDN) and the LASSO. It is noticeable that BPDN leads to stepwise algorithms [18], [28] including the popular $\ell_1$-homotopy [28]–[30], a forward-backward greedy search whose complexity is close to that of OMP. $\ell_1$-homotopy is closely connected to the Least Angle Regression (LARS), a simpler forward strategy allowing only atom selections. It is referred to as "LARS with the LASSO modification" in [30].

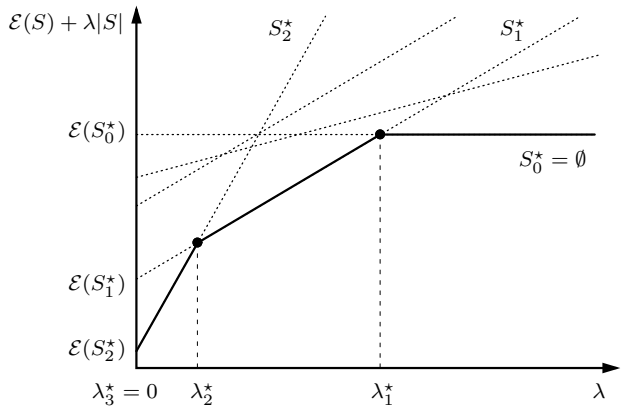



Fig. 1. Representation of lines $\lambda \mapsto \mathcal{E}(S)+\lambda|S|$ for various subsets $S$. The $\ell_0$-curve, in plain line, is the minimal curve $\lambda \mapsto \min_S\{\mathcal{E}(S)+\lambda|S|\}$. It is continuous, concave, and piecewise affine with a finite number of pieces. The $\ell_0$-penalized regularization path is composed of the supports (here, $S_0^\star$, $S_1^\star$, $S_2^\star$) that are optimal for some $\lambda$-values. For instance, $S_1^\star$ is optimal for $\lambda \in [\lambda_2^\star, \lambda_1^\star]$. These supports $S^\star$ induce global minimizers of $\mathcal{J}(\boldsymbol{x};\lambda)$, defined as the least-square solutions $\boldsymbol{x}_{S^\star}$. For instance, $\boldsymbol{x}_{S_1^\star}$ is a global minimizer of $\mathcal{J}(\boldsymbol{x};\lambda)$ with respect to $\boldsymbol{x}$ whenever $\lambda \in [\lambda_2^\star, \lambda_1^\star]$.

Importantly, $\ell_1$-homotopy solves the BPDN for a continuum of values of $t$.

### *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 $\lambda \mapsto \mathcal{E}(S)+\lambda|S|$, where $\mathcal{E}(S)=\|\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}\|_2^2$ 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 $(\lambda, \mathcal{J})$, as shown on Fig. 1.

Second, the set of solutions to (3) is piecewise constant with respect to $\lambda$ (see Appendix A for a proof). Geometrically, this result can be easily understood by noticing that the minimum of $\mathcal{J}(\boldsymbol{x};\lambda)$ with respect to $\boldsymbol{x}$ is obtained for all $\lambda$-values by considering the concave envelope of the set of lines $\lambda \mapsto \mathcal{E}(S)+\lambda|S|$ 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 $\lambda_i^\star$ around which the set of optimal solutions $\arg\min_{\boldsymbol{x}} \mathcal{J}(\boldsymbol{x};\lambda)$ 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 $\mathcal{J}(\boldsymbol{x};\lambda)$ with respect to $\boldsymbol{x}$ 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.

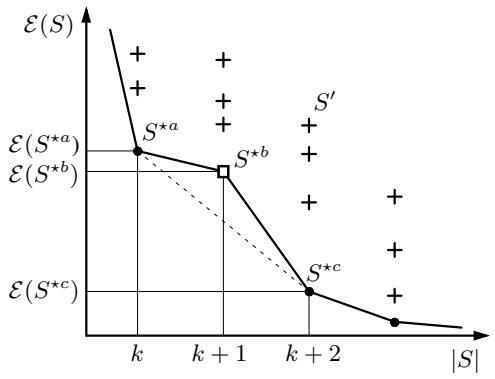



Fig. 2. Sparse approximation seen as a bi-objective optimization problem. The Pareto frontier gathers the non-dominated points: no other point can strictly decrease both $|S|$ and $\mathcal{E}(S)$. Bullets and squares are all Pareto solutions. A supported solution is a minimizer of $\mathcal{E}(S) + \lambda|S|$ with respect to $S$ for some $\lambda$. $S^{\star a}$ and $S^{\star c}$ are supported, contrary to $S^{\star b}$.

### 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 $\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2$ and the sparsity measure $\|\boldsymbol{x}\|_0$. Although $\boldsymbol{x}$ 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, $\boldsymbol{x}$ reading as a least-square minimizer among all vectors supported by $S$.

Fig. 2 is a classical bi-objective 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]. In the present context, $|S|$ takes integer values, thus the Pareto solutions are the minimizers of $\mathcal{E}(S)$ subject to $|S| \leq k$ for consecutive values of $k$. Equivalently, they minimize $|S|$ subject to $\mathcal{E}(S) \leq \varepsilon$ for some $\varepsilon$. They are usually classified as supported or non-supported. The former lay on the convex envelope of the Pareto frontier (the bullet points in Fig. 2) whereas the latter lay in the nonconvex areas (the square point). 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 weighting sum method and a more complex method is a nontrivial question. The answer depends on the problem at-hand and specifically, on the size of the nonconvex areas in 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. Consider the BPDN; because $\|\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}\|_2^2$ and $\|\boldsymbol{x}\|_1$ are convex functions of $\boldsymbol{x}$, the set of minimizers of $\|\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}\|_2^2+\lambda\|\boldsymbol{x}\|_1$ for all $\lambda$ coincides with the set of minimizers of $\|\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}\|_2^2$ s.t. $\|\boldsymbol{x}\|_1\leq t$ for all $t$ [33]. Both sets 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 $\mathcal{J}$ 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 $\mathcal{J}(\boldsymbol{x};\lambda)$ for a given $\lambda$. This motivates us to propose efficient OLS-based strategies for minimizing $\mathcal{J}$ 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 $\boldsymbol{A}$ 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 $\mathcal{E}(S)$ 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 and $\ell_0$-PD both read as descent algorithms in different senses: CSBR, first sketched in [37], repeatedly minimizes $\mathcal{J}(\boldsymbol{x};\lambda)$ for decreasing $\lambda$'s. On the contrary, $\ell_0$-PD minimizes $\mathcal{J}(\boldsymbol{x};\lambda)$ for any $\lambda$-value *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 candidate solutions 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) working for decreasing $\lambda$-values.

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 path 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 similar to the $\ell_1$-homotopy principle [28], [30].

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 $\boldsymbol{A}$ (usually, $m \leq n$ in sparse approximation). The observation signal $\boldsymbol{y}$ and the weight vector $\boldsymbol{x}$ are of size $m \times 1$ and $n \times 1$, respectively. We assume that any $\min(m, n)$ columns of $\boldsymbol{A}$ are linearly independent so that for any subset $S \subset \{1, \ldots, n\}$, the submatrix of $\boldsymbol{A}$ gathering the columns indexed by $S$ is full rank, and the least-square error $\mathcal{E}(S)$ 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 $S - \{i\}$ 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 set 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. The solutions delivered by our greedy algorithms will be referred to as the "approximate $\ell_0$-penalized path" since they are suboptimal algorithms.

### *B. Definition and properties of the $\ell_0$-regularized paths*

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

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

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

$$\min_S \big\{ \hat{\mathcal{J}}(S;\lambda) \triangleq \mathcal{E}(S) + \lambda|S| \big\}, \tag{6}$$

where $S$ stands for the support of $\boldsymbol{x}$. The optimal solutions $\boldsymbol{x}$ to problems (1), (2) and (3) can indeed be simply deduced from those of (4), (5) and (6), respectively, $\boldsymbol{x}$ 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 $\mathcal{S}^\star_{\mathrm{C}}(k)$ be the set of minimizers of the constrained problem* (4)*.*

*For $\lambda > 0$, let $\mathcal{S}^\star_{\mathrm{P}}(\lambda)$ be the set of minimizers of the penalized problem* (6)*. Additionally, we define the $\ell_0$-curve as the function $\lambda \mapsto \min_S\{\hat{\mathcal{J}}(S;\lambda)\}$. It is the concave envelope of a finite number of linear functions. Thus, it is concave and piecewise affine. Let $\lambda^\star_{I+1} \triangleq 0 < \lambda^\star_I < \ldots < \lambda^\star_1 < \lambda^\star_0 \triangleq +\infty$ delimit the affine intervals ($I+1$ contiguous intervals; see Fig. 1 in the case where $I = 2$).*

Each set $\mathcal{S}^\star_{\mathrm{C}}(k)$ or $\mathcal{S}^\star_{\mathrm{P}}(\lambda)$ can be thought of as a single support (*e.g.,* $\mathcal{S}^\star_{\mathrm{C}}(k)$ is reduced to the support $S^{\star a}$ 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 $\mathcal{S}^\star_{\mathrm{P}}(\lambda)$.

**Theorem 1** *$\mathcal{S}^\star_{\mathrm{P}}(\lambda)$ is a piecewise constant function of $\lambda$, being constant on each interval $\lambda \in (\lambda^\star_{i+1}, \lambda^\star_i)$.*

*Proof:* See Appendix A. ■

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) $\mathcal{S}^\star_{\mathrm{C}} = \{\mathcal{S}^\star_{\mathrm{C}}(k),\, k = 0, \ldots, \min(m, n)\}$.*

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

$\mathcal{S}^\star_{\mathrm{C}}$ gathers the solutions to (4) for all $k$. As illustrated on Fig. 2, the elements of $\mathcal{S}^\star_{\mathrm{C}}$ are the Pareto solutions whereas the elements of $\mathcal{S}^\star_{\mathrm{P}}$ 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, $\mathcal{S}^\star_{\mathrm{P}} \subset \mathcal{S}^\star_{\mathrm{C}}$, but the reverse inclusion is not guaranteed.

**Theorem 2** *$\mathcal{S}^\star_{\mathrm{P}} \subset \mathcal{S}^\star_{\mathrm{C}}$. Moreover, for any $\lambda \notin \{\lambda^\star_I, \ldots, \lambda^\star_0\}$, there exists $k$ such that $\mathcal{S}^\star_{\mathrm{P}}(\lambda) = \mathcal{S}^\star_{\mathrm{C}}(k)$.*

*Proof:* See Appendix A. ■

### C. Approximate $\ell_0$-penalized regularization path

Let us introduce notations for the *approximate* $\ell_0$-penalized path delivered by our heuristic search algorithms. Throughout the paper, the $\star$ notation is reserved for optimal solutions (*e.g.,* $\mathcal{S}^\star_{\mathrm{P}}$). It is removed when dealing with numerical solutions. The outputs of our algorithms will be composed of a list $\boldsymbol{\lambda} = \{\lambda_1, \ldots, \lambda_{J+1}\}$ of decreasing $\lambda$-values, and a list $\mathcal{S} = \{S_0, \ldots, S_J\}$ 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 $\boldsymbol{x}_j$ 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 on Fig. 3.

## III. Greedy continuation algorithm (CSBR)

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

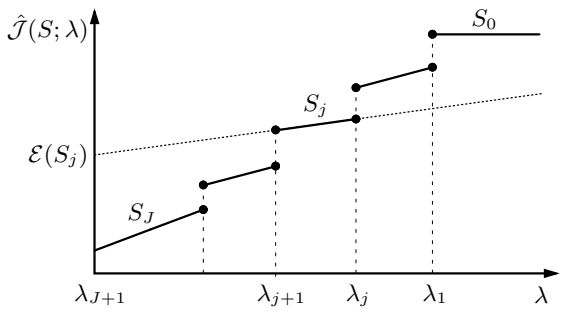



Fig. 3. Notations relative to our heuristic search algorithms. Their outputs are: *(i)* a sequence of values $\lambda_j$ sorted in the decreasing order; *(ii)* as many supports $S_j$, $S_j$ being the solution associated to all $\lambda \in (\lambda_{j+1}, \lambda_j)$. By extension, $S_0 = \emptyset$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

### A. Single Best Replacement

SBR is a deterministic descent algorithm dedicated to the minimization of $\hat{\mathcal{J}}(S;\lambda)$ 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 $S \pm \{i\}$ ($n$ insertion and removal trials);
2) Select the best replacement $S_{\text{best}} = S \pm \{\ell\}$, with

$$\ell \in \underset{i \in \{1,\ldots,n\}}{\arg\min}\ \hat{\mathcal{J}}(S \pm \{i\};\lambda); \tag{7}$$

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

SBR terminates when $\hat{\mathcal{J}}(S_{\text{best}};\lambda) \geq \hat{\mathcal{J}}(S;\lambda)$, *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 $\lambda = 0$, we have $\hat{\mathcal{J}}(S;0) = \mathcal{E}(S)$. Only insertions can be performed since any removal increases the squared error $\mathcal{E}(S)$. SBR coincides with the well-known OLS algorithm [7]. Generally, the $n$ replacement trials necessitate to compute $\mathcal{E}(S+\{i\})$ for all insertion trials and $\mathcal{E}(S-\{i\})$ for all removals. In [3], we proposed a fast and stable recursive implementation based on the Cholesky factorization of the Gram matrix $\boldsymbol{A}_S^T\boldsymbol{A}_S$ when $S$ is modified by one element (where $\boldsymbol{A}_S$ stands for the submatrix of $\boldsymbol{A}$ gathering the active columns). SBR is summarized in Tab. I. The optional output parameters $\ell_{\text{add}}$ and $\delta\mathcal{E}_{\text{add}}$ 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 $S \pm \{\ell\}$. $S_{\text{init}} = \emptyset$ yields an horizontal line since $\hat{\mathcal{J}}(\emptyset;\lambda) = \|\boldsymbol{y}\|_2^2$ does not depend on

TABLE I

SBR ALGORITHM FOR MINIMIZATION OF $\hat{\mathcal{J}}(S;\lambda)$ FOR FIXED $\lambda$ [3]. BY DEFAULT, $S_{\rm init}=\emptyset$. THE OUTPUTS $\delta\mathcal{E}_{\rm add}$ AND $\ell_{\rm add}$ ARE OPTIONAL. THE SINGLE REPLACEMENT TESTS APPEAR IN THE FOR LOOP.

**inputs** : $\boldsymbol{A}$, $\boldsymbol{y}$, $\lambda$, $S_{\rm init}$

**outputs**: $S$, $\delta\mathcal{E}_{\rm add}$, $\ell_{\rm add}$

$S_{\rm best} \leftarrow S_{\rm init}$;

**repeat**

  $S \leftarrow S_{\rm best}$;

  **for** $i = 1$ **to** $n$ **do**

    Compute $\hat{\mathcal{J}}(S \pm \{i\};\lambda)$;

  **end**

  $S_{\rm best} \leftarrow S \pm \{\ell\}$ with $\ell$ computed from (7);

**until** $\hat{\mathcal{J}}(S_{\rm best};\lambda) \geq \hat{\mathcal{J}}(S;\lambda)$;

Compute $\ell_{\rm add}$ according to (11);

Set $\delta\mathcal{E}_{\rm add} = \mathcal{E}(S) - \mathcal{E}(S+\{\ell_{\rm add}\})$;

$\lambda$. At the first SBR 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 the supports $S \leftarrow \{a,b\}$ 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 removal $\ell = a$. 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 $n$ single replacements decreases $\hat{\mathcal{J}}(\{b,c\};\lambda)$. 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 $\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 + \lambda\|\boldsymbol{x}\|_1$ when $\lambda$ is continuously decreasing [28]–[30]. An iteration of $\ell_1$-homotopy consists in two steps:

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

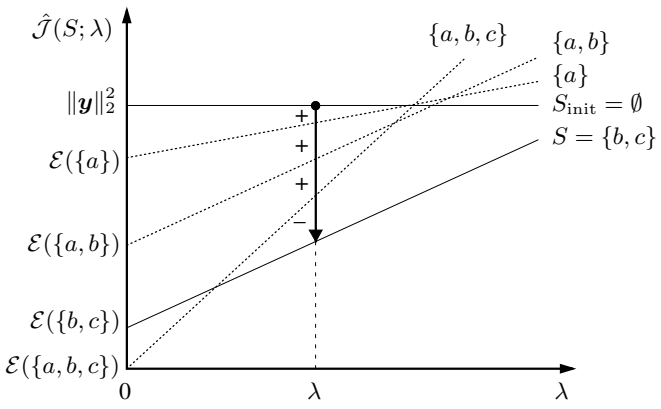



Fig. 4. Step-by-step illustration of the call $S = \text{SBR}(\emptyset;\lambda)$. Each single replacement is represented by a vertical displacement (from top to bottom) from lines $S$ to $S \pm \{\ell\}$. The symbols '+' and '-' respectively refer to the selection and de-selection of atoms $a$, $b$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

CSBR follows the same principle. The first step is now related to some *local* $\ell_0$-optimality conditions, and the second step consists in calling SBR at $\lambda_{\rm new}$ with the current active set as initial solution; see Fig. 5 for a sketch. A main difference with $\ell_1$-homotopy is that the $\ell_0$ solutions are suboptimal, *i.e.,* they are local minimizers of $\mathcal{J}(\boldsymbol{x};\lambda)$ with respect to $\boldsymbol{x}$.

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

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

This condition is illustrated on Fig. 6(a): all lines related to single replacements $S \pm \{i\}$ lay above the black point representing the value of $\hat{\mathcal{J}}(S;\lambda)$ for the current $\lambda$. By separating the conditions related to insertions $S + \{i\}$ and removals $S - \{i\}$, (8) rereads as the interval condition:

$$\lambda \in [\delta\mathcal{E}_{\rm add}(S), \delta\mathcal{E}_{\rm rmv}(S)], \tag{9}$$

where

$$\delta\mathcal{E}_{\rm add}(S) \triangleq \max_{i\notin S} \big\{\mathcal{E}(S) - \mathcal{E}(S+\{i\})\big\} \tag{10a}$$

$$\delta\mathcal{E}_{\rm rmv}(S) \triangleq \min_{i\in S} \big\{\mathcal{E}(S-\{i\}) - \mathcal{E}(S)\big\} \tag{10b}$$

refer to the maximum variation of the squared error when an atom is added in the support $S$ (respectively, removed from $S$).

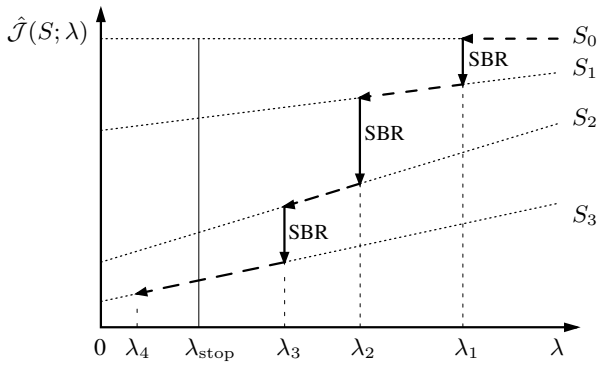



Fig. 5. Step-by-step illustration of CSBR with the early stopping condition $\lambda_j \leq \lambda_{\rm stop}$. The initial support is $S_0 = \emptyset$. SBR is called for three decreasing values (plain vertical arrows), with output $S_j$ at $\lambda_j$. The search for the next value $\lambda_{j+1}$ is represented by an oblique displacement along the line $S_j$.

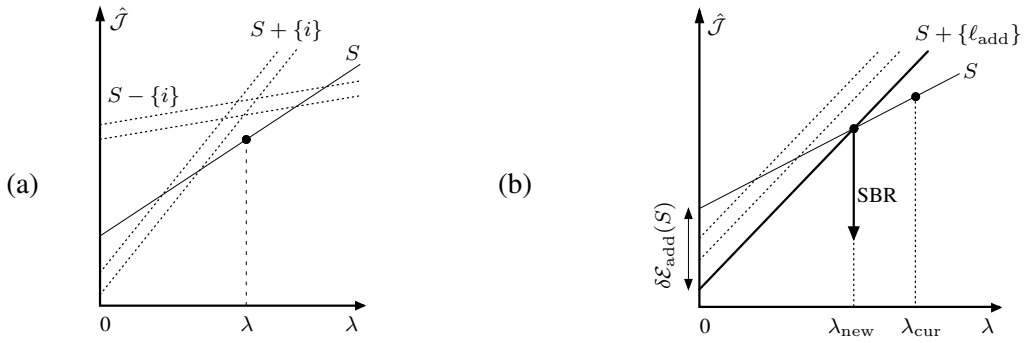



Fig. 6. Termination of SBR and next call to SBR. (a) When SBR terminates, no single replacement $S \pm \{i\}$ can decrease $\hat{\mathcal{J}}(S;\lambda)$. The dotted lines $S+\{i\}$ (of slope $|S|+1$) lay above the black point $(\lambda, \hat{\mathcal{J}}(S;\lambda))$. Similarly, all lines $S-\{i\}$, of slope $|S|-1$, lay above this point. (b) Here, $S$ is the SBR output at $\lambda_{\rm cur}$. The next call to SBR is done at $\lambda_{\rm new} = \delta\mathcal{E}_{\rm add}(S)$ with the initial subset $S+\{\ell_{\rm add}\}$. The line $S+\{\ell_{\rm add}\}$ lays below all other lines $S+\{i\}$ (dotted lines). Here, the $\lambda$-axis has been stretched by an arbitrary factor for improved readability. The horizontal length $\lambda_{\rm new}$ does not match the vertical length $\delta\mathcal{E}_{\rm add}(S)$, as it should without any stretching. The same stretching process will be done in Fig. 7.

*2) Violation of the local optimality conditions:* Consider the current output $S = \mathrm{SBR}(S_{\rm init}; \lambda_{\rm cur})$. The local optimality condition (9) is then met for $\lambda = \lambda_{\rm cur}$, but also for any $\lambda \in [\delta\mathcal{E}_{\rm add}(S), \lambda_{\rm cur}]$. The new value for which (9) is violated is $\lambda_{\rm new} = \delta\mathcal{E}_{\rm add}(S) - c$ where $c > 0$ is arbitrarily small. The violation occurs for $i = \ell_{\rm add}$, with

$$\ell_{\rm add} \in \operatorname*{arg\,max}_{i \notin S} \{\mathcal{E}(S) - \mathcal{E}(S+\{i\})\}. \tag{11}$$

TABLE II

CSBR ALGORITHM: SBR IS CALLED REPEATEDLY FOR DECREASING $\lambda_j$'S. AT ITERATION $j$, BOTH THE NEXT VALUE $\lambda_{j+1}$ AND THE NEXT INITIAL SUBSET $S_j + \{\ell_{\rm add}\}$ ARE PROVIDED AS SBR OUTPUTS.

**inputs** : $\boldsymbol{A}$, $\boldsymbol{y}$

**outputs**: $\mathcal{S}$: list of supports $S_j$; $\boldsymbol{\lambda}$: list of $\lambda_j$

$S_0 \leftarrow \emptyset$;

$S_{\rm init} \leftarrow \{\ell_{\rm add}\}$ with $\ell_{\rm add}$ computed from (13);

Compute $\lambda_1$ according to (13);

$j \leftarrow 1$;

**while** $\lambda_j > 0$ **do**

  Call $[S_j,\, \delta\mathcal{E}_{\rm add},\, \ell_{\rm add}] = \text{SBR}(S_{\rm init}; \lambda_j)$;

  $S_{\rm init} \leftarrow S_j + \{\ell_{\rm add}\}$;

  $\lambda_{j+1} \leftarrow \delta\mathcal{E}_{\rm add}$;

  $j \leftarrow j + 1$;

**end**

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

$$\lambda_{\rm new} = \delta\mathcal{E}_{\rm add}(S) \tag{12}$$

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

As illustrated on Fig. 6(b), the line $S + \{\ell_{\rm add}\}$ lays below all other parallel lines $S + \{i\}$. It intersects line $S$ at $\lambda_{\rm new}$. The vertical arrow represents the new call to SBR with inputs $S + \{\ell_{\rm add}\}$ and $\lambda_{\rm new}$. Because $S$ and $S + \{\ell_{\rm add}\}$ both lead to the same value of $\hat{\mathcal{J}}(\,.\,; \lambda_{\rm new})$, the de-selection of $\ell_{\rm add}$ is forbidden in the first iteration of SBR.

### C. CSBR algorithm

CSBR is summarized in Tab. II. The repeated calls to SBR deliver subsets $S_j$ for decreasing $\lambda_j$. As shown on 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:

$$\ell_{\rm add} \in \underset{i \in \{1,\ldots,n\}}{\arg\max} \frac{|\langle \boldsymbol{y}, \boldsymbol{a}_i \rangle|}{\|\boldsymbol{a}_i\|_2} \quad \text{and} \quad \lambda_1 = \frac{\langle \boldsymbol{y}, \boldsymbol{a}_{\ell_{\rm add}} \rangle^2}{\|\boldsymbol{a}_{\ell_{\rm add}}\|_2^2}. \tag{13}$$

According to Tab. II, CSBR stops when $\lambda_j = 0$, *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. In such cases, *ad hoc* early stopping rules can be considered [28], [45]. A natural rule takes the form

$\lambda_j \leq \lambda_{\text{stop}}$ with $\lambda_{\text{stop}} > 0$. Alternative rules involve a maximum cardinality ($|S_j| \geq k_{\text{stop}}$) and/or a minimum squared error ($\mathcal{E}(S_j) \leq \varepsilon_{\text{stop}}$).

Fig. 5 shows a step-by-step illustration with the early stop $\lambda_j \leq \lambda_{\text{stop}}$. The initial support $S_{\text{init}} = \{\ell_{\text{add}}\}$ and $\lambda_1$ are precomputed in (13). In the first call $S_1 = \text{SBR}(S_{\text{init}}; \lambda_1)$, a number of single replacements updates $S \leftarrow S \pm \{\ell\}$ are carried out leading to $S_1 = S$. This process is represented by the plain vertical arrow at $\lambda_1$ linking both lines $S_0$ and $S_1$ (the line $S_{\text{init}}$ is not shown for readability reasons). Once $S_1$ is obtained, the next value $\lambda_2$ is computed. This process is represented by an oblique, dashed arrow joining $\lambda_1$ and $\lambda_2$. These two processes are being repeated alternatively at the second and third iterations of CSBR. Finally, CSBR terminates after $\lambda_4$ has been computed because $\lambda_4 \leq \lambda_{\text{stop}}$.

## 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: *(i)* there might be jumps in this curve; and *(ii)* the slope of the 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 $\mathcal{S}$ 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 $\mathcal{S}$ are updated so as to decrease at most the concave polygonal curve. In particular, we impose that the least value is $\lambda_{J+1} = 0$, so that the concave envelope is computed over the whole domain $\lambda \in \mathbb{R}_+$.

### 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 $\mathcal{S}$ only if the resulting concave polygon can be decreased. This descent test is illustrated on Fig. 7 for two examples (top and bottom subfigures). 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 $\mathcal{S}$ whose edges lay above the line $S_{\text{new}}$ are removed from $\mathcal{S}$.

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

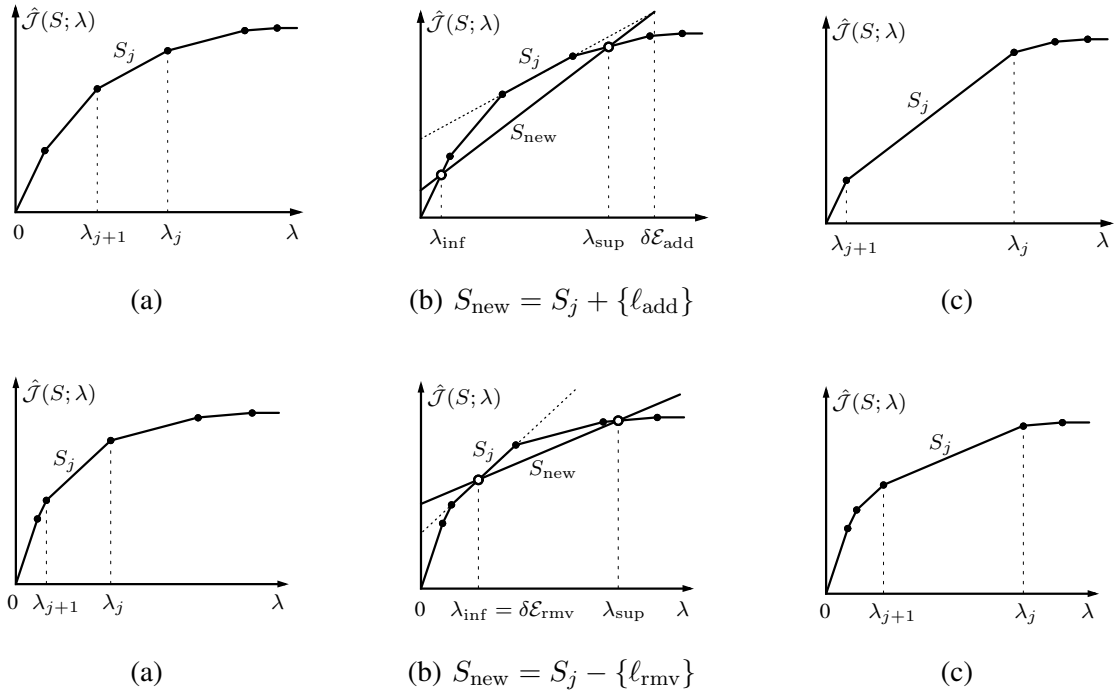



Fig. 7. $\ell_0$-PD algorithm: descent of the concave polygon when a new support $S_{\rm new} = S_j + \{\ell_{\rm add}\}$ (top) or $S_{\rm new} = S_j - \{\ell_{\rm rmv}\}$ (bottom) is included. (a) Initial configuration. (b) The intersection with line $S_{\rm new}$ is computed. This yields an interval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

### *B. Selection of the new candidate support*

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

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

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 $S_{\rm add} = S_j + \{\ell_{\rm add}\}$ and $S_{\rm rmv} = S_j - \{\ell_{\rm rmv}\}$ into $\mathcal{S}$, so that the concave polygon can be decreased at most. The `CCV_Descent` procedure (Tab. III) is first called with $S_{\rm new} \leftarrow S_{\rm add}$ leading to possible updates of $\mathcal{S}$ and $\boldsymbol{\lambda}$. It is called again with $S_{\rm new} \leftarrow S_{\rm rmv}$. Fig. 7 illustrates each of these calls: the slope of $S_{\rm new}$ is $|S_j| + 1$ and $|S_j| - 1$, respectively. When a support

TABLE III

CONCAVE POLYGON DESCENT PROCEDURE. WHEN A NEW SUBSET IS INCLUDED, BOTH LISTS $\mathcal{S}$ AND $\boldsymbol{\lambda}$ ARE UPDATED. THE FUNCTION `intersect` COMPUTES THE INTERSECTION BETWEEN A LINE AND A CONCAVE POLYGON. THIS YIELDS AN INTERVAL $[\lambda_{\rm inf}, \lambda_{\rm sup}]$. BY CONVENTION, $\lambda_{\rm inf} > \lambda_{\rm sup}$ WHEN THE INTERSECTION IS EMPTY.

**Procedure**: `CCV_Descent`($\mathcal{S}$, $S_{\rm new}$, $\boldsymbol{\lambda}$)

Call $[\lambda_{\rm inf}, \lambda_{\rm sup}] =$ `intersect`($\mathcal{S}$, $S_{\rm new}$);

**if** $\lambda_{\rm inf} < \lambda_{\rm sup}$ **then**

  Include $S_{\rm new}$ as an unexplored support in $\mathcal{S}$;

  Remove any subset $S_j$ from $\mathcal{S}$ such that $[\lambda_{j+1}, \lambda_j] \subset [\lambda_{\rm inf}, \lambda_{\rm sup}]$;

  Sort the subsets in $\mathcal{S}$ by increasing cardinality;

  Sort $\boldsymbol{\lambda}$ in the decreasing order;

**end**

$S_j$ has been explored, the new supports that have been included in $\mathcal{S}$ (if any) are tagged as unexplored.

### C. $\ell_0$-PD algorithm

$\ell_0$-PD is stated in Tab. IV. Initially, $\mathcal{S}$ is formed of the empty support $S_0 = \emptyset$. 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: the best insertion $S_{\rm add} = \{\ell_{\rm add}\}$ is computed in (13), and included in $\mathcal{S}$ during the call to `CCV_Descent`. The updated set $\mathcal{S}$ is now composed of $S_0 = \emptyset$ (explored) and $S_1 = S_{\rm add}$ (unexplored). 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 $S_{\rm add}$ and $S_{\rm rmv}$ may be included in $\mathcal{S}$ at a given iteration while a variable number of supports may be removed from $\mathcal{S}$.

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

TABLE IV

$\ell_0$-PD ALGORITHM. THE ALGORITHM MAINTAINS A LIST $\mathcal{S}$ OF SUPPORTS $S_j$ WHOSE CARDINALITY IS INCREASING WITH $j$. THE UNEXPLORED SUPPORT HAVING THE LOWEST CARDINALITY IS EXPLORED AT EACH ITERATION. THE LISTS $\mathcal{S}$ AND $\boldsymbol{\lambda}$ ARE UPDATED DURING THE CALLS TO `CCV_Descent`; $\boldsymbol{\lambda}$ IS SORTED IN THE DECREASING ORDER, WITH $\lambda_{J+1} = 0$. DURING THE FIRST ITERATION, $j = 0$ LEADS TO $S_{\rm rmv} \leftarrow \emptyset$.

**inputs** : $\boldsymbol{A}$, $\boldsymbol{y}$
**outputs**: $\mathcal{S}$, $\boldsymbol{\lambda}$

$\boldsymbol{\lambda} \leftarrow \{\lambda_1\}$ with $\lambda_1 \leftarrow 0$;
$S_0 \leftarrow \emptyset$, $S_0$.`expl` $\leftarrow 0$;
$\mathcal{S} \leftarrow \{S_0\}$;
**while** $\{\exists j : S_j$.`expl` $= 0\}$ **do**
  Set $j$ as the lowest index such that $S_j$.`expl` $= 0$;
  $S_j$.`expl` $\leftarrow 1$;
  Compute $S_{\rm add} \leftarrow S_j + \{\ell_{\rm add}\}$ from (11);
  **if** $j = 0$ **then**
    $S_{\rm rmv} \leftarrow \emptyset$;
  **else**
    Compute $S_{\rm rmv} \leftarrow S_j - \{\ell_{\rm rmv}\}$ from (14);
  **end**
  Call `CCV_Descent`($\mathcal{S}$, $S_{\rm add}$, $\boldsymbol{\lambda}$);
  Call `CCV_Descent`($\mathcal{S}$, $S_{\rm rmv}$, $\boldsymbol{\lambda}$);
**end**

### *D. Fast implementation*

The `CCV_Descent` procedure calls the function `intersect` to compute the intersection between a concave polygon $\mathcal{S}$ and a line $S_{\rm new}$. 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 Tab. III for brevity reasons.

**Lemma 1** *Let $\mathcal{S} = \{S_j,\, j = 0, \ldots, J\}$ be a list of supports associated to a continuous, concave polygon $\lambda \mapsto \min_j \hat{\mathcal{J}}(S_j; \lambda)$ with $J+1$ edges, delimited by $\boldsymbol{\lambda} = \{\lambda_0, \ldots, \lambda_{J+1}\}$. The following properties hold for all $j$:*

- *If $\delta\mathcal{E}_{\rm add}(S_j) < \lambda_{j+1}$, then the line $S_{\rm add} = S_j + \{\ell_{\rm add}\}$ lays above the current concave polygon.*
- *If $\delta\mathcal{E}_{\rm rmv}(S_j) > \lambda_j$, then the line $S_{\rm rmv} = S_j - \{\ell_{\rm rmv}\}$ lays above the current concave polygon.*

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

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, the index $j$ in $\lambda_j$ 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 $\lambda_j$ which is decreasing when the iteration number $j$ increases.

Second, the computation of the next value $\lambda_{j+1} \leq \lambda_j$ 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. The behavior of CSBR and $\ell_0$-PD is first analyzed for simple examples. Then, we provide a detailed comparison with other nonconvex algorithms for many scenarii.

### *A. Two generic problems*

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

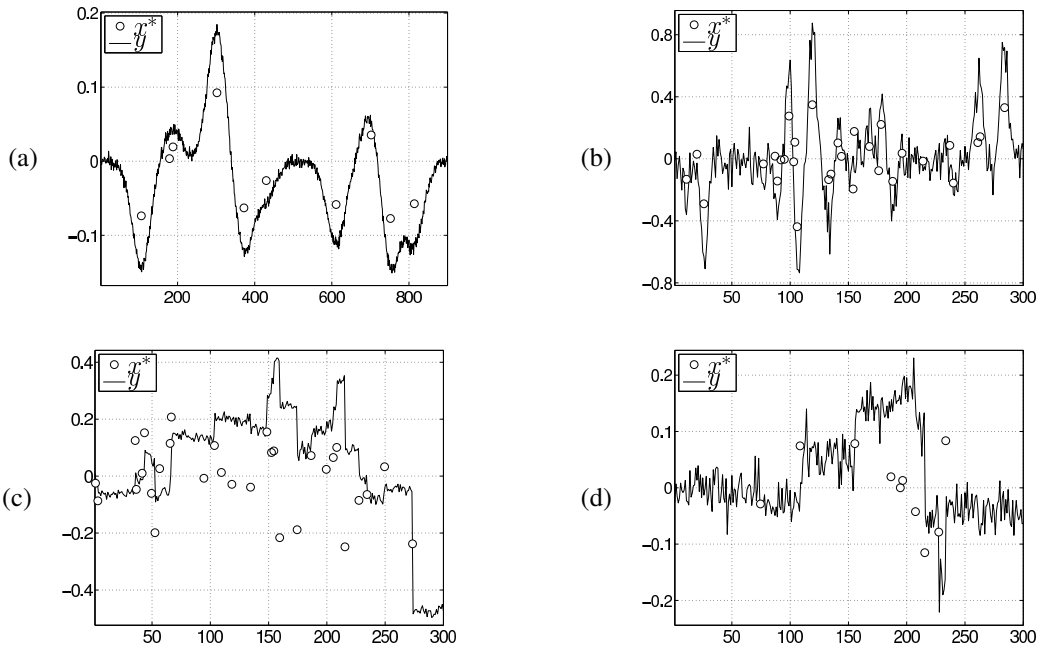
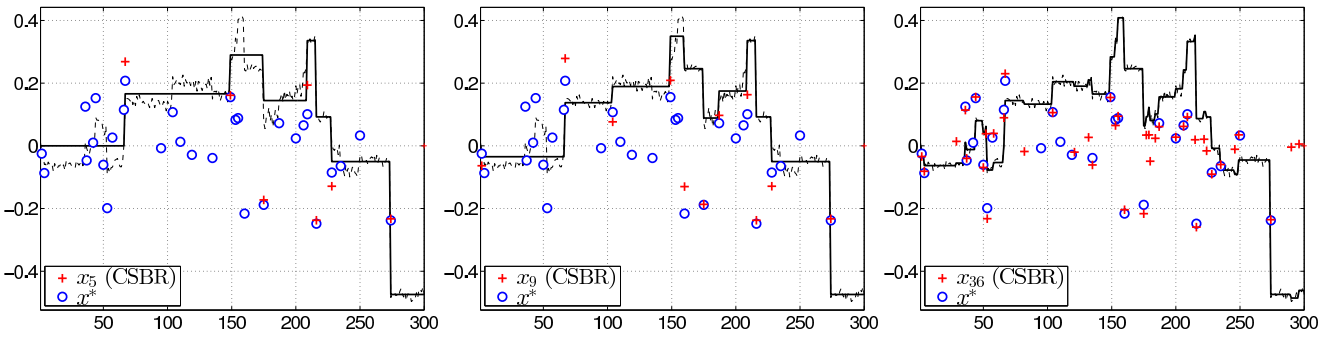



Fig. 8. Generic deconvolution (a,b) and jump detection (c,d) problems. The data vectors $\boldsymbol{y}$ and the $k$ nonzero amplitudes of $\boldsymbol{x}^\star$ are represented in plain lines and with small circles, respectively. (a) Sparse deconvolution problem with $k = 10$ spikes, $\mathrm{SNR} = 25$ dB, $\sigma = 24$ (Gaussian impulse response), and $m = 900$, $n = 756$ (size of dictionary $\boldsymbol{A}$). (b) Sparse deconvolution problem with $k = 30$, $\mathrm{SNR} = 10$ dB, $\sigma = 3$, $m = 300$, $n = 252$. (c) Jump detection problem with $k = 30$, $\mathrm{SNR} = 25$ dB, $m = n = 300$. (d) Jump detection problem with $k = 10$, $\mathrm{SNR} = 10$ dB, $m = n = 300$.

$k = 10$ and 30, and the signal-to-noise ratio (SNR) is equal to 25 and 10 dB, respectively. It is defined by $\mathrm{SNR} = 10\,\log(\|\boldsymbol{A}\boldsymbol{x}^\star\|_2^2/(m\sigma_n^2))$ where $\sigma_n^2$ is the variance of the noise process $\boldsymbol{n}$.

The jump detection problem is illustrated on Fig. 8(c,d). Here, $\boldsymbol{A}$ is the squared dictionary ($m = n$) defined by $A_{i,j} = 1$ if $i \geq j$, and 0 otherwise. The atom $\boldsymbol{a}_j$ codes for a jump at location $j$, and $x_j^\star$ matches the height of the jump. When $\boldsymbol{x}^\star$ is $k$-sparse, $\boldsymbol{A}\boldsymbol{x}^\star$ yields a piecewise constant signal with $k$ pieces, $\boldsymbol{x}^\star$ being the first-order derivative of the signal $\boldsymbol{A}\boldsymbol{x}^\star$.

Both generic problems involve either square or slightly undercomplete dictionaries. The case of overcomplete dictionaries will be discussed as well, *e.g.,* by considering the deconvolution problem with undersampled observations $\boldsymbol{y}$. The generic problems are already difficult because neighboring columns of $\boldsymbol{A}$ are highly correlated, and a number of fast algorithms that are efficient for well-conditioned dictionaries may fail to recover the support of $\boldsymbol{x}^\star$. The degree of difficulty of the deconvolution problem is controlled by the width $\sigma$ of the Gaussian impulse response and the sparsity $k$: for large values of $k$ and/or $\sigma$, the Gaussian features resulting from the convolution $\boldsymbol{h} * \boldsymbol{x}^\star$ strongly overlap. For the jump detection problem,




(a) $|S_5| = 7$, $\lambda_5 = 7.8e^{-2}$ (b) $|S_9| = 11$, $\lambda_9 = 4.1e^{-2}$ (c) $|S_{36}| = 41$, $\lambda_{36} = 3.4e^{-4}$

Fig. 9. Jump detection example: processing of the data of Fig. 8(c) using CSBR. Three sparse solutions $\boldsymbol{x}_j$ are shown, each being related to some CSBR output $S_j$, with $\|\boldsymbol{x}_j\|_0 = |S_j|$. The original vector $\boldsymbol{y}$ is represented in dashed lines and the approximation $\boldsymbol{A}\boldsymbol{x}_j$ is in solid line.

all the step signals related to the atoms $\boldsymbol{a}_j$ have overlapping supports.

### *B. Empirical behavior of CSBR and $\ell_0$-PD*

*1) Example:* Consider the problem shown on Fig. 8(c). Because CSBR and $\ell_0$-PD provide very similar results, we only show the CSBR results. CSBR delivers sparse solutions $\boldsymbol{x}_j$ for decreasing $\lambda_j$, $\boldsymbol{x}_j$ being the least-square solution supported by the $j$-th output of CSBR ($S_j$). Three sparse solutions $\boldsymbol{x}_j$ are represented on Fig. 9. For the first solution (lowest value of $|S_j|$, largest $\lambda_j$), only the seven main jumps are being detected (Fig. 9(a)). The cardinality of $S_j$ increases with $j$, and some other jumps are obtained together with possible false detections (Figs. 9(b,c)).

*2) Model order selection:* It may often be useful to select a single solution $\boldsymbol{x}_j$. The proposed algorithms are compatible with most classical 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 $\|\boldsymbol{x}_j\|_0 = |S_j|$. The first take the form $\min_j\{m \log \mathcal{E}(S_j) + \alpha|S_j|\}$ 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 number of spikes. On the contrary, the MDL

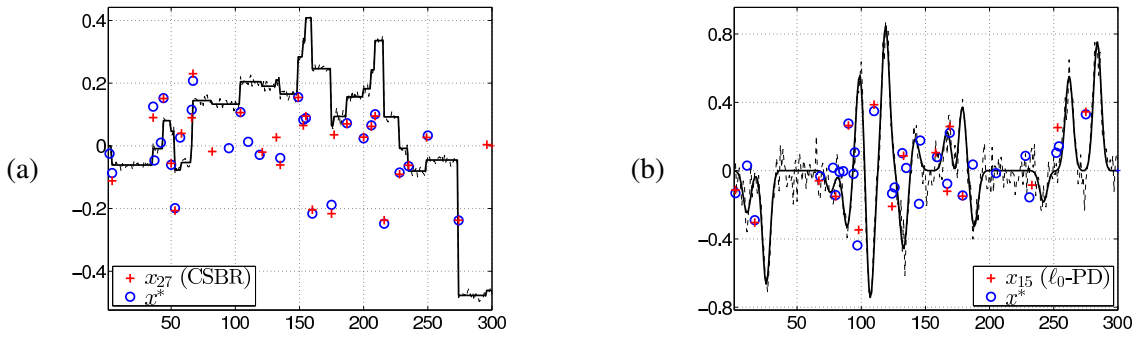



Fig. 10. Model order selection using MDLc: display of the selected sparse solution $\boldsymbol{x}_j$ and the related data approximation signal. The data of Fig. 8(c,b) ($k = 30$ true spikes) are processed using CSBR and $\ell_0$-PD, respectively. (a) corresponds to the simulation shown on Fig. 9. The MDLc solution is the CSBR output support $S_{27}$ of cardinality 27. (b) is related to the $\ell_0$-PD output $S_{15}$, with $|S_{15}| = 16$.

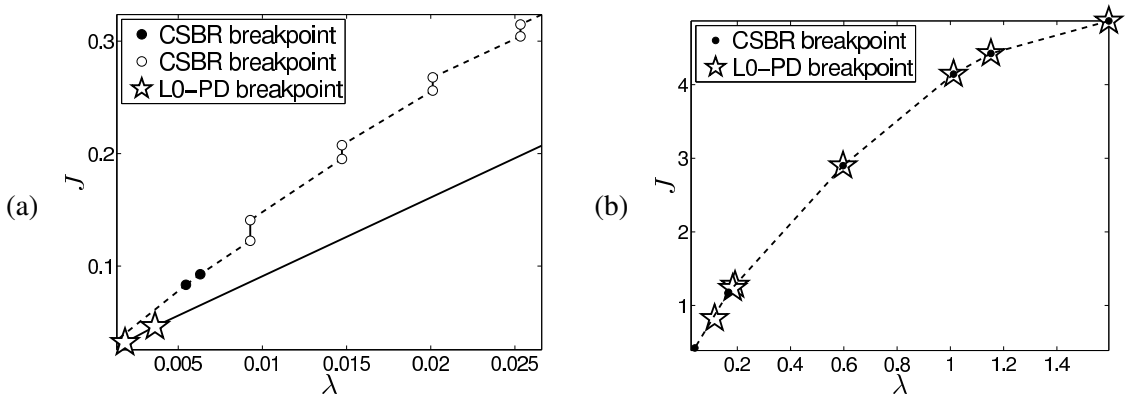



Fig. 11. Typical approximate $\ell_0$-curves found for the deconvolution problem of Fig. 8(a): zoom in for small and large $\lambda$'s. The $\ell_0$-PD curve is concave and continuous on $\lambda \in \mathbb{R}_+$. The CSBR curve is continuous only for large $\lambda$-values (black circles). For low $\lambda$'s, there can be discontinuities at breakpoint locations (white circles). Here, both curves almost coincide for large $\lambda$'s. The $\ell_0$-PD curve lays below the CSBR curve for low $\lambda$'s.

criterion yields quite accurate results. We found that 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 scenarii we have tested. It reads:

$$\min_j \Big\{ \log \mathcal{E}(S_j) + \frac{\log(m)(|S_j|+1)}{m - |S_j| - 2} \Big\}. \tag{15}$$

Fig. 10(a) illustrates that the number of spikes found using MDLc is very accurate for high SNRs (27 spikes are found, the unknown order being 30). It is underestimated for low SNRs: 16 spikes are found (instead of 30) for the simulation of Fig. 10(b) where SNR $= 10$ dB. This behavior is relevant because

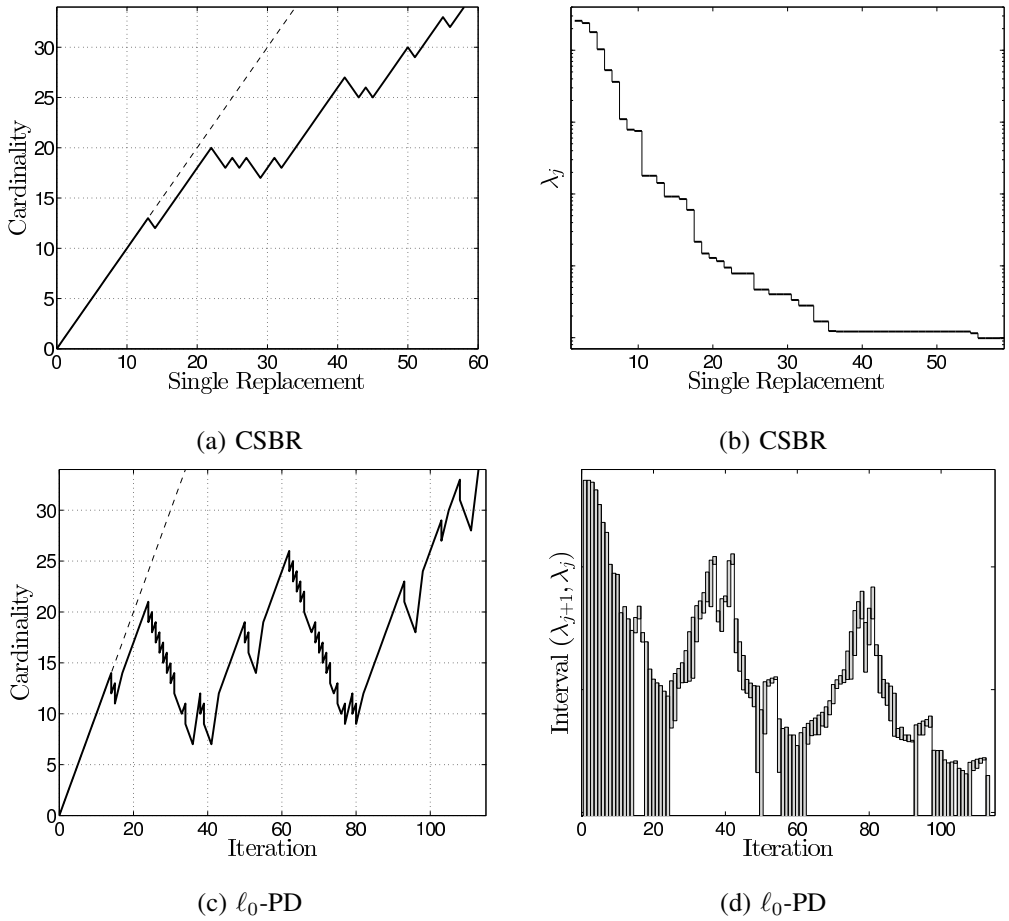
for noisy data, the spikes of smallest amplitudes are drowned in the noise. One cannot expect to detect them.

*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 for 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$(S_{\rm init};\lambda_j)$ returns $S_j = S_{\rm init}$; see Tab. II). Otherwise, a discontinuity breakpoint (white circle) appears. In Fig. 11, the CSBR and $\ell_0$-PD curves almost coincide for large $\lambda$'s, where only continuous breakpoints can be observed. For low $\lambda$'s, the $\ell_0$-PD curve lays below the CSBR curve, and discontinuity breakpoints appear in the latter curve.

Fig. 12 provides some insight on the CSBR and $\ell_0$-PD iterations for a sparse deconvolution problem with $\|\boldsymbol{x}^\star\|_0 = 17$ and SNR $= 20$ dB. In the CSBR subfigures, the horizontal axis represents the number of single replacements: 60 replacements are being performed from the initial empty support during the successive calls to SBR. For $\ell_0$-PD, the horizontal axis shows the iteration number. At most two new supports are being included in the list of candidate subsets at each iteration. The number of effective 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 cardinality and $\lambda$ is decreasing (Figs. 12(c,d)). From iterations 25 to 40, the very sparse solutions previously found ($k \leq 20$) are improved as a series of atom de-selections is performed. They are being improved again around iteration 80. On the contrary, the sparsest solutions are never improved with CSBR, which works for decreasing $\lambda$'s (Figs. 12(a,b)). For $\ell_0$-PD, the early stopping parameter $\lambda_{\rm stop}$ may have a strong influence on the improvement of the sparsest solutions and 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 subsection V-A with various parameter settings: problem dimension $(m, n)$, ratio $m/n$, signal-to-noise ratio, cardinality of $\boldsymbol{x}^\star$, and width $\sigma$ of the Gaussian impulse response for the deconvolution problem. The settings are listed on Table V for 10 scenarii. Because the proposed algorithms 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 $k = \|\boldsymbol{x}^\star\|_0 \leq 30$.




(a) CSBR (b) CSBR

(c) $\ell_0$-PD (d) $\ell_0$-PD

Fig. 12. Series of single replacements performed by CSBR and $\ell_0$-PD. (a) CSBR: cardinality of the current support found after each single replacement. (b) Breakpoints $\lambda_j$ found by CSBR, represented in log-scale. SBR is executed for each $\lambda_j$, and the number of single replacements for fixed $\lambda_j$ matches the length of the horizontal steps in the figure. (c) $\ell_0$-PD: cardinality of the supports appended to the regularization path during the iterations. At each iteration, 0, 1 or 2 supports are included. Vertical steps appear whenever two supports are simultaneously included. (d) $\ell_0$-PD: representation in log-scale of the current interval $(\lambda_{j+1}, \lambda_j)$ (grey color). When the grey bars reach the bottom of the image, the lower bound equals $\lambda_{j+1} = 0$.

*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 efficient 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 penalty $\log(|x_i| + \varepsilon)$ [20], [23], [53];

TABLE V

SETTINGS RELATED TO EACH SCENARIO: $k$ IS THE SPARSITY. $f$ CONTROLS THE DICTIONARY SIZE: $m = f\ m_{\mathrm{DEF}}$, $n = f\ n_{\mathrm{DEF}}$ WITH $n_{\mathrm{DEF}} \approx m_{\mathrm{DEF}} = 300$. BY DEFAULT, $f = 1$. THE UNDERSAMPLING PARAMETER $\Delta$ EQUALS 1 BY DEFAULT ($m_{\mathrm{DEF}} \geq n_{\mathrm{DEF}}$). IT IS INCREASED TO GENERATE PROBLEMS WITH OVERCOMPLETE DICTIONARIES ($m \approx n/\Delta$). THE GAUSSIAN IMPULSE RESPONSE WIDTH IS SET TO $\sigma = f\ \sigma_{\mathrm{DEF}}$ WITH $\sigma_{\mathrm{DEF}} = 3$ OR 8.

| Scenario | Type | SNR | $k$ | $f$ | $\Delta$ | $m$ | $n$ | $\sigma$ |
|---|---|---|---|---|---|---|---|---|
| A | Deconv. | 25 | 30 | 1 | 1 | 300 | 282 | 3 |
| B | Deconv. | 10 | 10 | 1 | 1 | 300 | 252 | 8 |
| C | Deconv. | 25 | 10 | 3 | 1 | 900 | 756 | 24 |
| D | Deconv. | 25 | 30 | 6 | 1 | 1800 | 1692 | 18 |
| E | Jumps | 25 | 10 | 1 | 1 | 300 | 300 | $\emptyset$ |
| F | Jumps | 25 | 30 | 1 | 1 | 300 | 300 | $\emptyset$ |
| G | Jumps | 10 | 10 | 1 | 1 | 300 | 300 | $\emptyset$ |
| H | Deconv. | $+\infty$ | 10 | 3 | 2 | 450 | 756 | 24 |
| I | Deconv. | $+\infty$ | 30 | 3 | 2 | 450 | 756 | 24 |
| J | Deconv. | $+\infty$ | 10 | 1 | 4 | 75 | 252 | 8 |

3) $\ell_0$ penalized least squares for cyclic descent (L0LS-CD) [54];
4) Smoothed $\ell_0$ (SL0) [43], [55].

We resort to a penalized least-square implementation for all algorithms, the only algorithm directly working with the $\ell_0$ penalty being 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 efficient than thresholding algorithms. Moreover, the cyclic descent approach is becoming very popular in the recent sparse approximation literature [44], [56] although its speed of convergence is sensitive to the quality of the initial solution. Here, we use the BPDN initial solution $\arg\min_{\boldsymbol{x}} \{\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 + \mu\|\boldsymbol{x}\|_1\}$ where $\mu$ is set to half of the 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 nonzero and very sparse 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

(*i.e.,* Gaussian functions of decreasing widths). For simplicity reasons, we set the lowest width relative to the knowledge of the smallest nonzero amplitude of the ground truth solution $\boldsymbol{x}^\star$. 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 IR$\ell_1$ implementation depends on both the choice of parameter $\varepsilon$ (which controls the degree of nonconvexity) and the $\ell_1$ solver. We have tested two $\ell_1$ solvers: the in-crowd algorithm [59] together with an empirical setting of $\varepsilon > 0$, and $\ell_1$ homotopy in the limit case $\varepsilon \to 0$, following [53]. We found that $\ell_1$ homotopy is faster than in-crowd, mainly because the Matlab implementation of in-crowd (provided by the authors) makes calls to the `quadprog` built-in function, which is computationally expensive for large dimension problems.

*2) Numerical protocol:* Because the competing algorithms work for a single $\lambda$ value, we need to define a grid, denoted by $\{\lambda_i^{\mathrm{G}},\, i = 1, \ldots, N_\lambda\}$, for comparison purposes. Such grid is defined in logscale for each of the 10 scenarii $(k, \boldsymbol{A}, \mathrm{SNR})$ 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 $\boldsymbol{x}^\star(t)$ and noise vector $\boldsymbol{n}(t)$ are randomly drawn. This leads us to simulate $T$ observation vectors $\boldsymbol{y}(t) = \boldsymbol{A}\boldsymbol{x}^\star(t) + \boldsymbol{n}(t)$ with $t \in \{1, \ldots, T\}$. Specifically, the location of the nonzero amplitudes in $\boldsymbol{x}^\star(t)$ are uniformly distributed and the amplitude 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 $\boldsymbol{y}(t)$ and $\lambda_i^{\mathrm{G}}$ 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 $\lambda_i^{\mathrm{G}}$ directly deduces from their set of output supports and the knowledge of both breakpoints surrounding $\lambda_i^{\mathrm{G}}$.

The algorithms are first evaluated in the optimization viewpoint: the related criteria are their capacity to reach a low value of $\mathcal{J}(\boldsymbol{x}; \lambda)$ and the corresponding CPU time. In this viewpoint, the proposed methods might be somehow favored since they are more directly designed with the criterion $\mathcal{J}(\boldsymbol{x}; \lambda)$ in mind. On the other hand, $\mathcal{J}(\boldsymbol{x}; \lambda)$ 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 $\mathcal{J}(\boldsymbol{x}; \lambda_i^{\mathrm{G}})$ found for each trial and each $\lambda_i^{\mathrm{G}}$. Averaging this value over the trials $t$ yields a table TabJ$(a, \lambda_i^{\mathrm{G}})$ where $a$ denotes a candidate algorithm. Similarly, the CPU time is averaged over the trials $t$, leading to another table TabCPU$(a, \lambda_i^{\mathrm{G}})$. Each table is represented separately as a 2D plot with a specific color for each algorithm: see, *e.g.,* Fig. 13. CSBR and $\ell_0$-PD are represented with continuous curves because $\mathcal{J}(\boldsymbol{x}; \lambda)$ is computed for a

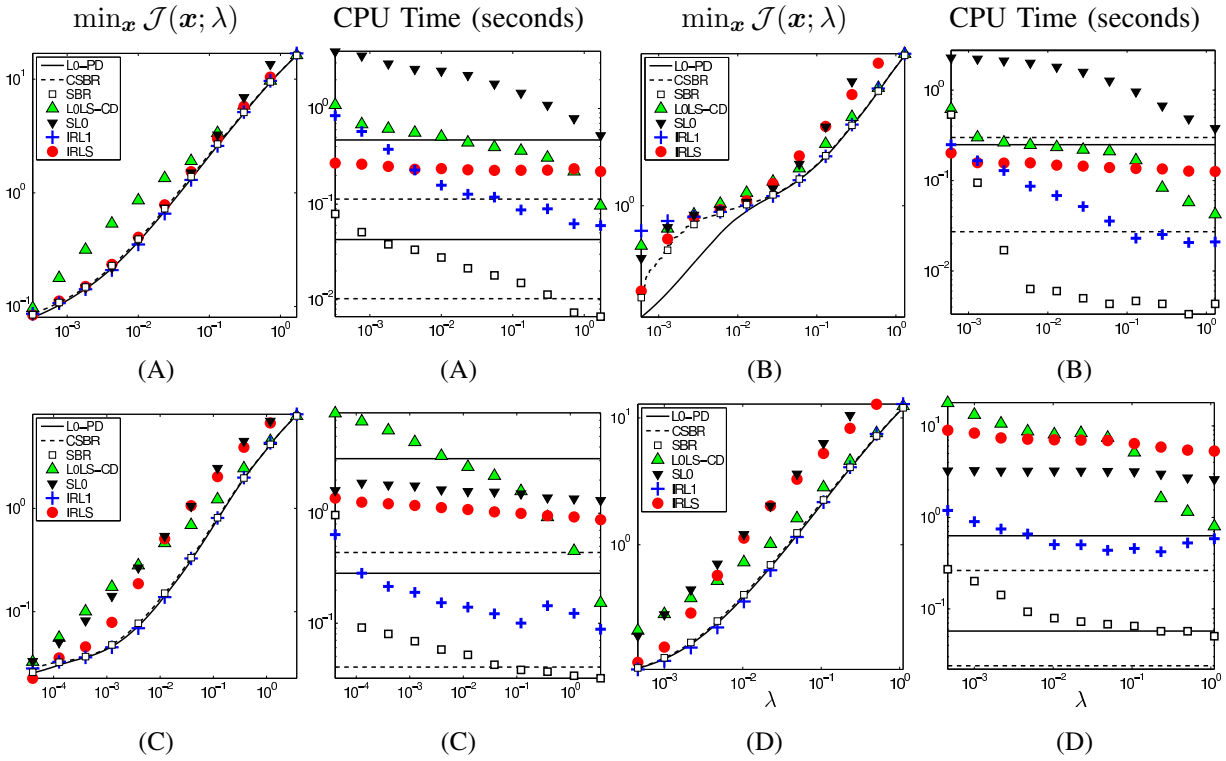



Fig. 13. Comparison of algorithms for the noisy deconvolution problem, *i.e.,* for the first scenarii reported on Table V. For each scenario, the algorithms are being evaluated in terms of $\mathcal{J}$-value and of CPU time for $N_\lambda = 11$ values $\lambda_i^{\mathrm{G}}$. Evaluations are averaged over 30 trials. The overall and mean (normalization by $N_\lambda = 11$) CPU times related to CSBR (respectively, $\ell_0$-PD) are shown as two parallel horizontal lines.

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

$$|S^\star(t)\backslash S(t,a,\lambda_i^{\mathrm{G}})| + |S(t,a,\lambda_i^{\mathrm{G}})\backslash S^\star(t)| \tag{16}$$

between the support $S^\star(t)$ of the unknown sparse vector $\boldsymbol{x}^\star(t)$ and the support $S(t,a,\lambda_i^{\mathrm{G}})$ of the sparse reconstruction at $\lambda_i^{\mathrm{G}}$ with algorithm $a$. (16) takes into account both numbers of false negatives $|S^\star(t)\backslash S(t,a,\lambda_i^{\mathrm{G}})|$ and of false positives $|S(t,a,\lambda_i^{\mathrm{G}})\backslash S^\star(t)|$. Denoting by $S(t,a,\lambda_{\mathrm{opt}}^{\mathrm{G}}) \leftarrow S(t,a,\lambda_i^{\mathrm{G}})$ the solution support that is the closest to $S^\star(t)$ according to (16), we further consider the number of true positives in $S(t,a,\lambda_{\mathrm{opt}}^{\mathrm{G}})$, defined as $|S^\star(t) \cap S(t,a,\lambda_{\mathrm{opt}}^{\mathrm{G}})|$. We will thus report:

- the support error;
- the corresponding number of true positives;

- the corresponding model order $|S(t, a, \lambda^{\mathrm{G}}_{\mathrm{opt}})|$.

Averaging these measures over $T$ trials yields the support error score SE($a$), the true positive score TP($a$) and the model order, denoted by Order($a$). The numbers of false positives (FP) and of true/false negatives 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 evaluations of the MDLc estimate accuracy. For CSBR and $\ell_0$-PD, all output supports are considered to compute the MDLc estimate as described in subsection V-B. For other algorithms, it is equal to one of the sparse reconstructions obtained at $\lambda^{\mathrm{G}}_i$ for $i \in \{1, \ldots, N_\lambda\}$. The same three measures as above are computed for the MDLc estimate 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($a$) score. This is done by running one iteration of cyclic descent (L0LS-CD): most small nonzero amplitudes are then thresholded to 0. Regarding the values of $\mathcal{J}(\boldsymbol{x}; \lambda)$, a post-processing is performed for algorithms that do not rely on the $\ell_0$-norm. This post-processing can be interpreted as a local descent of $\mathcal{J}(\boldsymbol{x}; \lambda)$. It consists in: *(i)* running one iteration of cyclic descent (L0LS-CD); *(ii)* computing the squared error related to the output support. L0LS-CD is indeed a local descent algorithm dedicated to $\mathcal{J}(\boldsymbol{x}; \lambda)$ but the convergence towards a least-square minimizer is not reached in one iteration.

*4) Analysis in the optimization viewpoint:* CSBR and $\ell_0$-PD are always among the most accurate to minimize the cost function, as illustrated on Figs. 13, 14 and 15. 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 clearly discriminate the accuracy of SBR and CSBR: one may behave slightly better than the other depending on the scenarii. 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 are $\ell_0$-homotopy algorithms, *i.e.,* a set of solutions are delivered 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 $\lambda_{\mathrm{stop}}$.

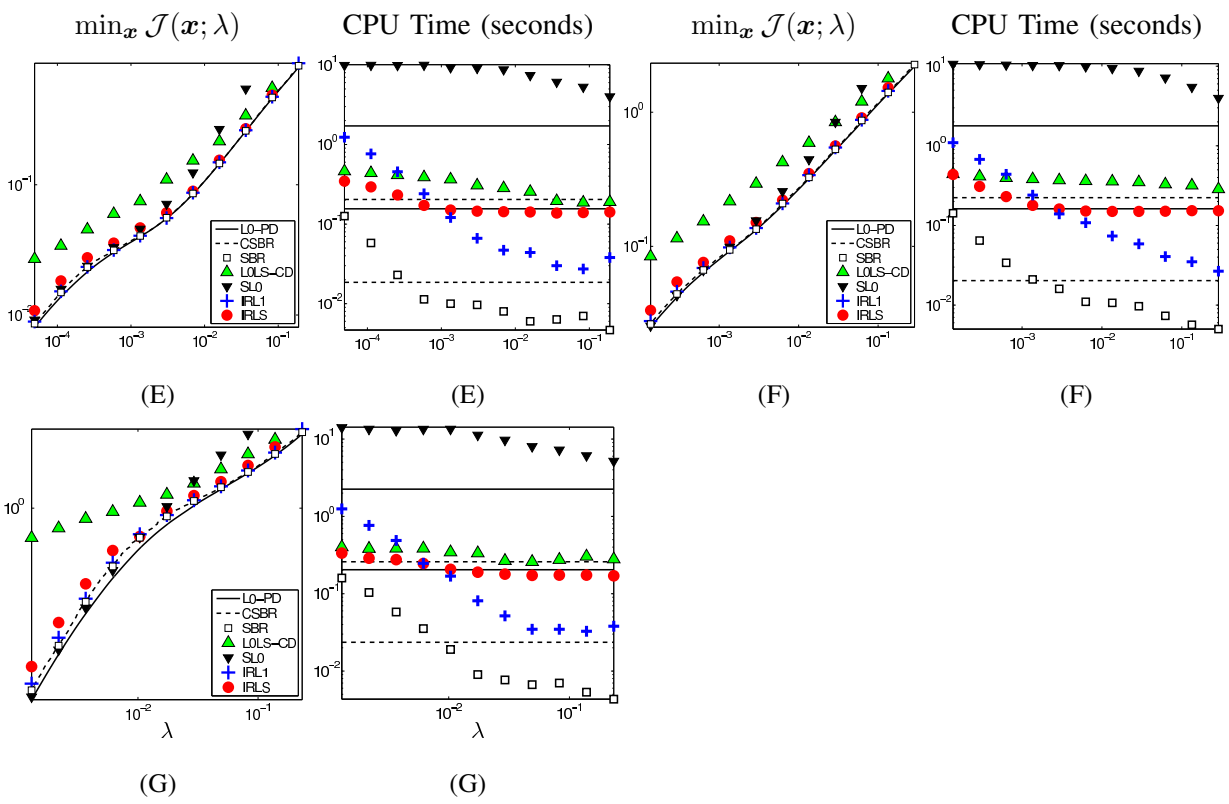



Fig. 14. Comparison of algorithms for the jump detection problem for the scenarii E, F and G of Table V.

The price to pay for a better performance is an increase of the computation burden. On Figs. 13, 14 and 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. However, the latter times refer to a single execution for some $\lambda_i^{\mathrm{G}}$ value. If one wants to recover sparse solutions for many $\lambda_i^{\mathrm{G}}$'s, they must be cumulated. This is the reason why we have drawn a second line for CSBR and $\ell_0$-PD corresponding to a normalization (by $N_\lambda = 11$) of the overall computation time. In this viewpoint, the CPU time 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 algorithms to make the CPU time comparisons meaningful. We have 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 ($m > 500$) to avoid huge

TABLE VI

JUMP DETECTION PROBLEM IN THE NOISY SETTING. THE ALGORITHMS ARE EVALUATED 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 $k$. THE SCORES RELATED TO THE MDLC ESTIMATE ARE INDICATED SIMILARLY.

| **Scenario E** | $\ell_0$-PD | CSBR | SBR | $\ell_0$LS-CD | S$\ell_0$ | 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** | $\ell_0$-PD | CSBR | SBR | $\ell_0$LS-CD | S$\ell_0$ | IR$\ell_1$ | IRLS |
| 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** | $\ell_0$-PD | CSBR | SBR | $\ell_0$LS-CD | S$\ell_0$ | IR$\ell_1$ | IRLS |
| 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) | 5.23 | 4.73 | 4.73 | 5.17 | 6.07 | 4.97 | 5.57 |
| 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 |

computational costs. The stopping rule of CSBR and $\ell_0$-PD is always $\lambda \leq \lambda_{\rm stop} = \alpha\lambda_1^{\rm G}$ with $\alpha = 1$ for CSBR and 0.5 (medium size) or 0.8 (large size) for $\ell_0$-PD. For L0LS-CD, the maximum number of cyclic descents (update of every amplitude $x_i$) is set to 60 or 10 depending on the dimension. For SL0, we have followed the default setting of [43] for the rate of deformation of the nonconvex penalty. The number of BFGS iterations done in the local minimization steps for each penalty is set to $L = 40$ or 5. It is set to $5L$ 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 for large dimensions. Finally, SBR does not require any arbitrary stopping rule. The problems of large dimensions correspond to scenarii C and D. We observe on Fig. 13 that the comparison (trade-off performance *vs*

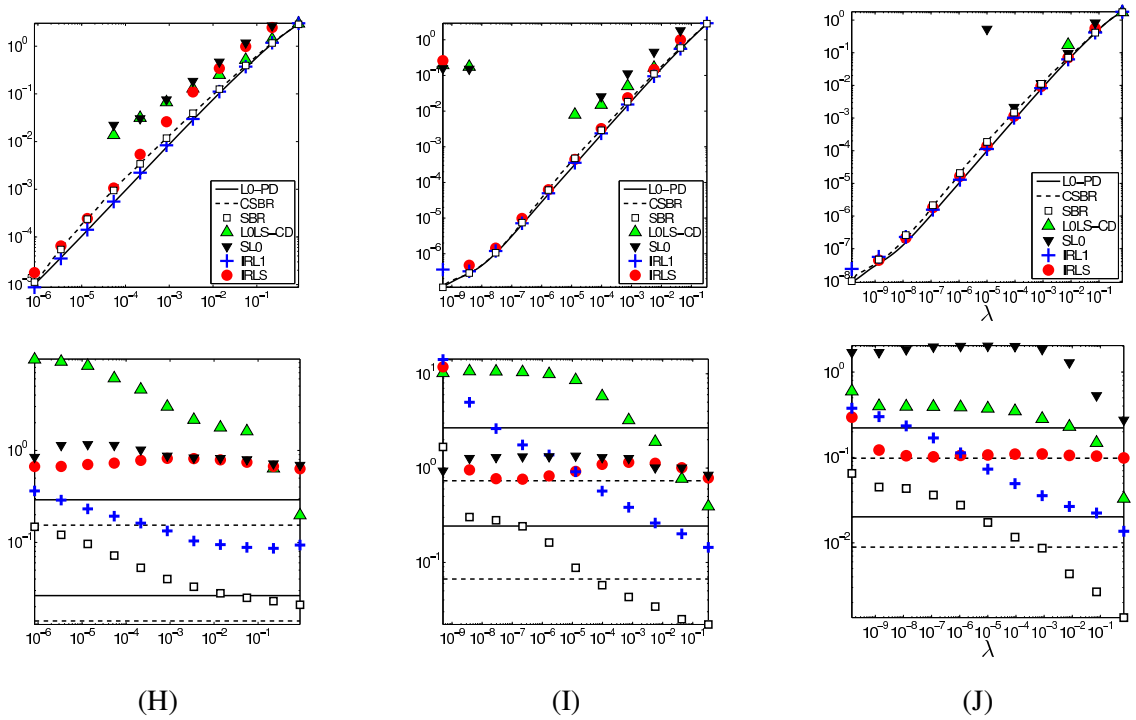



Fig. 15. Comparison of algorithms for the noise-free deconvolution problem, *i.e.,* for the scenarii H, I and J of Table V. Some markers do not appear for low $\lambda$'s (L0LS-CD, SL0) in the left figures because they do not lay in the zoom-in window (their performance is poor).

computation time) is now clearly in favor of CSBR and $\ell_0$-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 scenarii E to J (Tabs. VI and VII). For noisy deconvolution problems, these results are omitted because the support error is often quite large and the true positive scores are low whatever the algorithm, especially for scenarii B to D. Specifically, the least support error always exceeds 20, 10, 10 and 32 for the scenarii 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 true positive rate. More sophisticated localization tests are non binary 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 scenarii A to D (*e.g.,* the subsets found with CSBR are of cardinalities 33, 9, 15 and 38, the true cardinalities being 30, 10, 10 and 30). The model orders are also quite accurate for the noisy jump detection problem (Tab. VI) whereas the true support is often partially detected by

TABLE VII

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

| **Scenario H** | $\ell_0$-PD | CSBR | SBR | $\ell_0$LS-CD | S$\ell_0$ | IR$\ell_1$ | IRLS |
|---|---|---|---|---|---|---|---|
| SE | 2.5 | 3.6 | 4.8 | 11.4 | 13.0 | 0.8 | 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 | 3.6 | 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** | $\ell_0$-PD | CSBR | SBR | $\ell_0$LS-CD | S$\ell_0$ | 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 | 3.8 | 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** | $\ell_0$-PD | CSBR | SBR | $\ell_0$LS-CD | S$\ell_0$ | 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 |

several of the considered algorithms. Here, CSBR and $\ell_0$-PD are among the best algorithms in terms of support error.

The results of Tab. VII and Fig. 15 correspond to the deconvolution problem in noise-free case. The data $\boldsymbol{y}$ are undersampled so that the dictionary $\boldsymbol{A}$ is overcomplete. The undersampling rate $\Delta \approx m/n$ is set to 2 in scenarii H and I and 4 in scenario J. Again, CSBR and $\ell_0$-PD are among the best (SE, TP, MDLc-order) especially for the most difficult problem J.

*6) Overcomplete dictionaries with noise:* We now provide arguments indicating that the proposed algorithms are competitive as well for noisy problems with overcomplete dictionaries. The detailed experiments commented below are not reported for space reasons.

We have first considered the noisy deconvolution problem with $\Delta = 2$ or 4 leading to overcomplete dictionaries, the other parameters being set as in scenarii 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 discriminate the performance of algorithms because these measures are very weak for all considered algorithms. Moreover, the values of $\mathcal{J}(\lambda)$ found for most algorithms are often similar.

We have also considered an adaptive spline approximation problem generalizing the jump detection problem to the approximation of a signal using piecewise polynomials of degree $P = 1$ or 2 [3]. The jump detection problem can indeed be thought of as the approximation with a piecewise constant signal ($P = 0$). The generalized version [3] 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 $\boldsymbol{y}$ (subdictionaries $\boldsymbol{A}^0$, $\boldsymbol{A}^1$ and $\boldsymbol{A}^2$). The dictionary then takes the form $\boldsymbol{A} \leftarrow [\boldsymbol{A}^0, \boldsymbol{A}^1]$ or $\boldsymbol{A} \leftarrow [\boldsymbol{A}^0, \boldsymbol{A}^1, \boldsymbol{A}^2]$ where each sub-dictionary $\boldsymbol{A}^p$ ($p \leq P$) is formed of shifted versions of the one-sided power function $i \mapsto [\max(i, 0)]^p$. The size of the full dictionary $\boldsymbol{A}$ is approximately $m \times (P + 1)m$. Hence, it becomes overcomplete as soon as $P \geq 1$. We have shown [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 efficient than their competitors in terms of values of $\mathcal{J}(\lambda)$. However, the rate of true positives is 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. Software

The Matlab implementation of the proposed CSBR and $\ell_0$-PD algorithms is available at www.cran.univ-lorraine.fr/perso/charles.soussen/software.html including programs showing how to call these functions.

## VII. Conclusion

The choice of a relevant sparse approximation algorithm relies on a trade-off between the desired performance and the computation time one is ready to spend. 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 local minimizers of $\mathcal{J}(\boldsymbol{x}; \lambda) = \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 + \lambda\|\boldsymbol{x}\|_0$ [3]. This behavior is in contrast with other classical sparse algorithms.

We have shown the usefulness and efficiency of the two 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 efficient 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 of the cardinality $\|\boldsymbol{x}\|_0$.

Our perspectives include the proposal of forward-backward search algorithms that will be faster than SBR and potentially more efficient. In the standard version of SBR, CSBR and $\ell_0$-PD, a single replacement refers to the insertion or removal 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 A
## PROPERTIES OF THE $\ell_0$ REGULARIZATION PATHS

In this appendix, we prove that the $\ell_0$-penalized path $\mathcal{S}^\star_{\mathrm{P}}$ (see Definition 2) is piecewise constant (Theorem 1) and is a subset of the $\ell_0$-constrained regularization path $\mathcal{S}^\star_{\mathrm{C}}$ (Theorem 2). We will denote the $\ell_0$-curve by $\lambda \mapsto \mathcal{J}^\star(\lambda) = \min_S\{\hat{\mathcal{J}}(S;\lambda)\}$. Let us recall that this function is concave and affine on each interval $(\lambda^\star_{i+1}, \lambda^\star_i)$, with $i \in \{0, \ldots, I\}$ (Definition 1). Moreover, $\lambda^\star_{I+1} = 0$ and $\lambda^\star_0 = +\infty$.

### A. Proof of Theorem 1

We prove Theorem 1 together with the following lemma, which is informative about the content of $\mathcal{S}^\star_{\mathrm{P}}(\lambda)$ for the breakpoints $\lambda = \lambda^\star_i$.

**Lemma 2** *Let $i \in \{1, \ldots, I-1\}$. Then, for all $\lambda \in (\lambda^\star_{i+1}, \lambda^\star_i)$, $\mathcal{S}^\star_{\mathrm{P}}(\lambda) \subset \mathcal{S}^\star_{\mathrm{P}}(\lambda^\star_{i+1}) \cap \mathcal{S}^\star_{\mathrm{P}}(\lambda^\star_i)$.*

*For the first and last intervals, we have:*

- *For all $\lambda \in (0, \lambda^\star_I)$, $\mathcal{S}^\star_{\mathrm{P}}(\lambda) \subset \mathcal{S}^\star_{\mathrm{P}}(\lambda^\star_I)$.*
- *For all $\lambda \in (\lambda^\star_1, +\infty)$, $\mathcal{S}^\star_{\mathrm{P}}(\lambda) = \{\emptyset\} \subset \mathcal{S}^\star_{\mathrm{P}}(\lambda^\star_1)$.*

*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 $(\lambda^\star_{i+1}, \lambda^\star_i)$, $\mathcal{J}^\star(\lambda)$ coincides with $\hat{\mathcal{J}}(S_i;\lambda) = \mathcal{E}(S_i) + \lambda|S_i|$, where $S_i$ is some optimal subset for all $\lambda \in (\lambda^\star_{i+1}, \lambda^\star_i)$.

Let $\lambda \in (\lambda^\star_{i+1}, \lambda^\star_i)$ and $S \in \mathcal{S}^\star_{\mathrm{P}}(\lambda)$. Then, $\hat{\mathcal{J}}(S;\lambda) = \hat{\mathcal{J}}(S_i;\lambda)$. 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 $(\lambda^\star_{i+1}, \lambda)$ or $(\lambda, \lambda^\star_i)$, which contradicts the definition of $S_i$. We conclude that $S \in \mathcal{S}^\star_{\mathrm{P}}(\lambda')$ for all $\lambda' \in (\lambda^\star_{i+1}, \lambda^\star_i)$.

We have shown that the content of $\mathcal{S}^\star_{\mathrm{P}}(\lambda)$ does not depend on $\lambda$ when $\lambda \in (\lambda^\star_{i+1}, \lambda^\star_i)$. ■

*Proof of Lemma 2:* The first result $\mathcal{S}^\star_{\mathrm{P}}(\lambda) \subset \mathcal{S}^\star_{\mathrm{P}}(\lambda^\star_{i+1}) \cap \mathcal{S}^\star_{\mathrm{P}}(\lambda^\star_i)$ is obtained by slightly adapting the proof of Theorem 1: replace $(\lambda^\star_{i+1}, \lambda^\star_i)$ by the closed interval $[\lambda^\star_{i+1}, \lambda^\star_i]$, and set $\lambda'$ to both endpoints of this interval.

The second and third results are obtained similarly, by considering the intervals $(0, \lambda^\star_I]$ and $[\lambda^\star_1, +\infty)$, and setting $\lambda' \leftarrow \lambda^\star_I$ and $\lambda' \leftarrow \lambda^\star_1$, respectively. It is obvious that $\mathcal{S}^\star_{\mathrm{P}}(\lambda)$ reduces to the empty support for $\lambda > \lambda^\star_1$ since the $\ell_0$-curve is constant for $\lambda > \lambda^\star_1$. ■

### *B. Proof of Theorem 2*

The first result is straightforward: for any $\lambda$ and for $S \in \mathcal{S}^\star_{\mathrm{P}}(\lambda)$, we have $S \in \mathcal{S}^\star_{\mathrm{C}}(|S|)$. Otherwise, there would exist $S'$ with $|S'| \leq |S|$ and $\mathcal{E}(S') < \mathcal{E}(S)$. Then, $\hat{\mathcal{J}}(S';\lambda) < \hat{\mathcal{J}}(S;\lambda)$ would contradict $S \in \mathcal{S}^\star_{\mathrm{P}}(\lambda)$.

To prove the second result, let us first show that for any $i$, $\exists k_i : \forall \lambda \in (\lambda^\star_{i+1}, \lambda^\star_i)$, $\mathcal{S}^\star_{\mathrm{P}}(\lambda) \subset \mathcal{S}^\star_{\mathrm{C}}(k_i)$.

Let $S \in \mathcal{S}^\star_{\mathrm{P}}(\lambda)$ for some $\lambda \in (\lambda^\star_{i+1}, \lambda^\star_i)$. Theorem 1 implies that $S \in \mathcal{S}^\star_{\mathrm{P}}(\lambda)$ for *any* $\lambda \in (\lambda^\star_{i+1}, \lambda^\star_i)$. Therefore, $\mathcal{J}^\star(\lambda) = \hat{\mathcal{J}}(S;\lambda)$ for $\lambda \in (\lambda^\star_{i+1}, \lambda^\star_i)$, and the slope of line $S$, *i.e.,* $|S|$, is constant whatever $S \in \mathcal{S}^\star_{\mathrm{P}}(\lambda)$ and $\lambda \in (\lambda^\star_{i+1}, \lambda^\star_i)$. Let us denote this constant by $k_i = |S|$. According to the first paragraph of the proof, $S \in \mathcal{S}^\star_{\mathrm{P}}(\lambda)$ implies that $S \in \mathcal{S}^\star_{\mathrm{C}}(k_i)$.

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

## 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 $\ell_0$ norm. Uniqueness of the global minimizer", *SIAM J. Imaging 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. 4572–4584, Oct. 2011.

[4] J. A. Tropp and S. J. Wright, "Computational methods for sparse solution of linear inverse problems", *Proc. IEEE, invited paper (Special Issue "Applications of sparse representation and compressive sensing")*, vol. 98, no. 5, pp. 948–958, June 2010.

[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. Rezaiifar, and P. S. Krishnaprasad, "Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition", in *Proc. 27th Asilomar Conf. on Signals, Systems and Computers*, 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*, Chapman and Hall, London, UK, 2nd edition, Apr. 2002.

[9] S. F. Cotter, J. Adler, B. D. Rao, and K. Kreutz-Delgado, "Forward sequential algorithms for best basis selection", *IEE Proc. Vision, Image and Signal Processing*, vol. 146, no. 5, pp. 235–244, Oct. 1999.

[10] L. Rebollo-Neira and D. Lowe, "Optimized orthogonal matching pursuit approach", *IEEE Signal Process. Lett.*, vol. 9, no. 4, pp. 137–140, Apr. 2002.

[11] T. Blumensath and M. E. Davies, "Iterative thresholding for sparse approximations", *J. Fourier Anal. Appl.*, vol. 14, no. 5, pp. 629–654, Dec. 2008.

[12] W. Dai and O. Milenkovic, "Subspace pursuit for compressive sensing signal reconstruction", *IEEE Trans. Inf. Theory*, vol. 55, no. 5, pp. 2230–2249, May 2009.

[13] D. Needell and J. A. Tropp, "CoSaMP: Iterative signal recovery from incomplete and inaccurate samples", *Appl. Comp. Harmonic Anal.*, vol. 26, no. 3, pp. 301–321, May 2009.

[14] C. Herzet and A. Drémeau, "Bayesian pursuit algorithms", Research Report, INRIA Rennes Bretagne Atlantique - Télécom ParisTech, Rennes, France, Jan. 2014.

[15] M. A. Efroymson, "Multiple regression analysis", in *Mathematical Methods for Digital Computers*, A. Ralston and H. S. Wilf, Eds., vol. 1, pp. 191–203. Wiley, New York, 1960.

[16] K. N. Berk, "Forward and backward stepping in variable selection", *J. Statist. Comput. Simul.*, vol. 10, no. 3-4, pp. 177–185, Apr. 1980.

[17] T. Zhang, "Adaptive forward-backward greedy algorithm for learning sparse representations", *IEEE Trans. Inf. Theory*, vol. 57, no. 7, pp. 4689–4708, July 2011.

[18] M. A. T. Figueiredo, R. D. Nowak, and S. J. Wright, "Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems", *IEEE J. Sel. Top. Signal Process.*, vol. 1, no. 4, pp. 586–597, Dec. 2007.

[19] M. Zibulevsky and M. Elad, "$\ell_1 - \ell_2$ optimization in signal and image processing", *IEEE Sig. Proc. Mag.*, vol. 27, no. 3, pp. 76–88, May 2010.

[20] E. J. Candès, M. B. Wakin, and S. P. Boyd, "Enhancing sparsity by reweighted $\ell_1$ minimization", *J. Fourier Anal. Appl.*, vol. 14, no. 5-6, pp. 877–905, Dec. 2008.

[21] G. Gasso, A. Rakotomamonjy, and S. Canu, "Recovering sparse signals with a certain family of nonconvex penalties and DC programming", *IEEE Trans. Signal Process.*, vol. 57, no. 12, pp. 4686–4698, Dec. 2009.

[22] N. Mourad and J. P. Reilly, "Minimizing nonconvex functions for sparse vector reconstruction", *IEEE Trans. Signal Process.*, vol. 58, no. 7, pp. 3485–3496, July 2010.

[23] D. P. Wipf and S. Nagarajan, "Iterative reweighted $\ell_1$ and $\ell_2$ methods for finding sparse solutions", *IEEE J. Sel. Top. Signal Process. (Special issue on Compressive Sensing)*, vol. 4, no. 2, pp. 317–329, Apr. 2010.

[24] A. Gholami and S. M. Hosseini, "A general framework for sparsity-based denoising and inversion", *IEEE Trans. Signal Process.*, vol. 59, no. 11, pp. 5202–5211, Nov. 2011.

[25] I. Ramírez and G. Sapiro, "Universal regularizers for robust sparse coding and modeling", *IEEE Trans. Image Process.*, vol. 21, no. 9, pp. 3850–3864, Sept. 2012.

[26] H. A. Le Thi, B. T. Nguyen Thi, and H. M. Le, "Sparse signal recovery by difference of convex functions algorithms", in *Intelligent Information and Database Systems*, A. Selamat, N. T. Nguyen, and H. Haron, Eds., Berlin, 2013, vol. 7803 of *Lecture Notes in Computer Science*, pp. 387–397, Springer Verlag.

[27] I. Selesnick and I. Bayram, "Sparse signal estimation by maximally sparse convex optimization", *IEEE Trans. Signal Process.*, vol. 62, no. 5, pp. 1078–1092, Mar. 2014.

[28] D. L. Donoho and Y. Tsaig, "Fast solution of $\ell_1$-norm minimization problems when the solution may be sparse", *IEEE Trans. Inf. Theory*, vol. 54, no. 11, pp. 4789–4812, Nov. 2008.

[29] M. R. Osborne, B. Presnell, and B. A. Turlach, "A new approach to variable selection in least squares problems", *IMA Journal of Numerical Analysis*, vol. 20, no. 3, pp. 389–403, 2000.

[30] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani, "Least angle regression", *Ann. Statist.*, vol. 32, no. 2, pp. 407–499, Apr. 2004.

[31] 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", *Structural optimization*, vol. 14, no. 1, pp. 63–69, Aug. 2007.

[32] R. T. Marler and J. S. Arora, "Survey of multi-objective optimization methods for engineering", *Structural and Multidisciplinary Optimization*, vol. 26, no. 6, pp. 369–395, Apr. 2004.

[33] 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. 890–912, Nov. 2008.

[34] P. M. T. Broersen, "Subset regression with stepwise directed search", *J. R. Statist. Soc. C*, vol. 35, no. 2, pp. 168–177, 1986.

[35] D. Haugland, "A bidirectional greedy heuristic for the subspace selection problem", in *Engineering stochastic local search algorithms. Designing, implementing and analyzing effective heuristics*, T. Stützle, M. Birattari, and H. H. Hoos, Eds., Berlin, Germany, Sept. 2007, vol. 4638 of *Lect. Notes Comput. Sci.*, pp. 162–176, Springer Verlag.

[36] S. Chatterjee, D. Sundman, M. Vehkaperä, and M. Skoglund, "Projection-based and look-ahead strategies for atom selection", *IEEE Trans. Signal Process.*, vol. 60, no. 2, pp. 634–647, Feb. 2012.

[37] J. Duan, C. Soussen, D. Brie, and J. Idier, "A continuation approach to estimate a solution path of mixed L2-L0 minimization problems", in *Signal Processing with Adaptive Sparse Structured Representations (SPARS workshop)*, Saint-Malo, France, Apr. 2009, pp. 1–6.

[38] S. Kwon, J. Wang, and B. Shim, "Multipath matching pursuit", *IEEE Trans. Inf. Theory*, vol. 60, no. 5, pp. 2986–3001, May 2014.

[39] S. Maymon and Y. Eldar, "The Viterbi algorithm for subset selection", *IEEE Signal Process. Lett.*, vol. 22, no. 5, pp. 524–528, May 2015.

[40] E. Wasserstrom, "Numerical solutions by the continuation method", *SIAM Rev.*, vol. 15, no. 1, pp. 89–119, Jan. 1973.

[41] D. M. Malioutov, M. Çetin, and A. S. Willsky, "Homotopy continuation for sparse signal representation", in *Proc. IEEE ICASSP*, Philadephia, PA, Mar. 2005, vol. V, pp. 733–736.

[42] J. Trzasko and A. Manduca, "Highly undersampled magnetic resonance image reconstruction via homotopic $\ell_0$-minimization", *IEEE Trans. Medical Imaging*, 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 $\ell^0$ 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, Sept. 2011.

[45] D. L. Donoho, V. Stodden, and Y. Tsaig, "About SparseLab", Tech. Rep., Stanford University, Mar. 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, July 2004.

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

[48] G. Wahba, "Practical approximate solutions to linear operator equations when the data are noisy", *SIAM J. Num. 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. Top. Signal Process.*, vol. 4, no. 3, pp. 298–309, June 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. and 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. Num. 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 *IEEE Conference on Decision and Control*, Honolulu, HI, 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, "$l_q$ 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 $l_0$-norm-regularized least squares compressed sensing", *EURASIP J. Wireless Comm. and Networking*, 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, 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 Computation*, vol. 13, no. 4, pp. 751–763, Apr. 2001.

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