From Bernoulli-Gaussian deconvolution to sparse signal restoration

Charles Soussen, Jérôme Idier, David Brie, Junbo Duan

From Bernoulli-Gaussian deconvolution to sparse signal restoration

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

C. Soussen, D. Brie and J. Duan are with the Centre de Recherche en Automatique de Nancy (CRAN, UMR 7039, Nancy-University, CNRS). Boulevard des Aiguillettes, B.P. 70239, F-54506 Vandœuvre-lès-Nancy, France. Tel: (+33)-3 83 68 44 71, Fax: (+33)-3 83 68 44 62. E-mail: {charles.soussen,david.brie,junbo.duan}@cran.uhp-nancy.fr.

J. Idier is with the Institut de Recherche en Communications et Cybernétique de Nantes (IRCCyN), BP 92101, 1 rue de la Noë, 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.

Abstract

Formulated as a least-square minimization problem under an ℓ_0 constraint, sparse signal approximation is a discrete optimization problem, known to be NP complete. Classical algorithms include, by increasing cost and efficiency, Matching Pursuit (MP), Orthogonal Matching Pursuit (OMP), Orthogonal Least Squares (OLS) and the exhaustive search algorithm. We focus on problems (namely for highly correlated dictionaries) in which OMP and OLS do not guarantee to find the optimal solution. Then, it is of interest to develop new sub-optimal search algorithms yielding better approximations within a computation time that may be slightly more expensive than that of OLS but remains much cheaper than the exhaustive search. We revisit the Single Most Likely Replacement (SMLR) algorithm, developed in the mid-1980's for Bernoulli-Gaussian signal restoration. We show that the formulation of sparse signal approximation as a limit case of Bernoulli-Gaussian signal restoration leads to an ℓ_0-penalized least-square minimization problem, for which the SMLR algorithm can be straightforwardly adapted. The adapted algorithm, called Single Best Replacement (SBR), is an OLS forward-backward extention based on successive updates of the sparse signal support by one element (insert a new element inside the support or remove an existing support element). We finally propose a fast and stable implementation based on an efficient update of the least-square error. The approach is illustrated on the deconvolution with a Gaussian impulse response and on the joint detection of discontinuities at different orders in a signal.

CONTENTS

I. Introduction

Sparse signal approximation consists in the decomposition of a given signal $\boldsymbol{y}$ by means of a limited number of elements from a dictionary $\boldsymbol{A}$. This problem has received considerable attention over the past few years, because it occurs in many fields of applications, among which Fourier synthesis, mono- and multidimensional deconvolution, image compression, statistical regression, compressive sensing. The popularity of sparse approximation algorithms relies on the fact that it is possible to provide efficient sparse approximations of a signal even for under-determined problems in which the size of dictionary is larger than the size of the data.

Sparse signal approximation can be formulated as the minimization of a least-square cost function of the form $\mathcal{E}(\boldsymbol{x}) = \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|^2$ under the constraint that $\boldsymbol{x}$ is sparse. This problem is often referred to as subset selection or feature selection, because imposing the sparsity constraint on the weights $\boldsymbol{x}$ consists

in allowing a limited number of non-zero coordinates x_i, or equivalently, selecting a subset of columns of $\boldsymbol{A}$. Let us denote by *active set* the set of selected columns. For a given active set, the minimization of $\mathcal{E}$ reduces to an unconstrained least-square problem in which the matrix $\boldsymbol{A}$ is replaced by a smaller dimension submatrix obtained by gathering the active columns of $\boldsymbol{A}$. Imposing the sparsity of the solution thus consists in minimizing $\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|^2$ subject to the constraint that the ℓ_0 pseudo-norm of $\boldsymbol{x}$, defined as the number of non-zero entries of $\boldsymbol{x}$, is lower than a given number k. This yields a discrete problem (since there are a finite number of possible active sets) which is known to be NP-complete [1], [2]. In this paper, we focus on "difficult" problems, in which some of the columns of $\boldsymbol{A}$ are highly correlated, the unknown weight vector $\boldsymbol{x}^\star$ is not necessarily sparse and/or the data are noisy. Hereafter, we distinguish two approaches to address the sparse signal approximation problem in a fast and sub-optimal manner and we discuss their relevance for difficult problems.

The first approach, which has been the most popular in the last decade, approximates the subset selection problem by a continuous optimization problem, convex or not, which is easier to solve. The function approximating the ℓ_0-norm must be non smooth at zero in order to yield sparse solutions [3], [4]. Among the nonconvex functions approximating the ℓ_0-norm, let us mention the Gaussian-shaped function [5] and the ℓ_p pseudo-norm ($p < 1$) [6]. Among the convex approximations, the approach utilizing the ℓ_1-norm instead of the ℓ_0-norm [7], [8] has been increasingly investigated, leading to the LASSO optimization problem. Its popularity relies on efficient algorithms, among which the LARS algorithm (also called homotopy) which finds the solution path, *i.e.,* the set of solutions *for all* degrees of sparsity [9], [10]. Several authors have provided sufficient conditions under which the ℓ_0- and ℓ_1-constrained least-square problems lead to solutions having the same support [8], [11], [12]. These conditions typically state that the unknown weight signal is highly sparse, that the correlation between any pair of columns of $\boldsymbol{A}$ is sufficiently small, and that the noise level must be low. They are often not satisfied when dealing with real data.

The second approach uses a fast and sub-optimal search algorithm to address the *exact* subset selection problem. A first possibility is to use a thresholding algorithm, *e.g.,* CoSaMP [13] and Iterative Hard Thresholding (IHT) [14]. These algorithms rely on gradient based iterations of the form $\boldsymbol{x}' = \boldsymbol{x} + \boldsymbol{A}^t(\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x})$, followed by the threshold of a number of non-zero components x_i. We observed that they do not yield accurate approximations in difficult cases in which the dictionary columns are highly correlated. Another possibility is to resort to greedy search algorithms which gradually increase or decrease by one the size of the active set. The simplest greedy algorithms are Matching Pursuit (MP) [15] and its improvement Orthogonal Matching Pursuit (OMP) [16]. Both are referred to as forward greedy algorithms, since they

start from an empty active set and then gradually increase it by one element. In contrast, the backward algorithm of Couvreur and Bresler [17] starts from a complete active set which is gradually decreased by one element. It is only valid if the dictionary is not overcomplete. A few authors have introduced forward-backward algorithms in which insertion and removals of one element into the active set are both allowed [18], [19]. Both contributions showed the interest of performing insertions and removals of atoms from the dictionary. This strategy can indeed handle an early false detection since its further removal from the support is allowed. In contrast, forward algorithms always add new entries into the active set, the insertion of a false entry being irreversible.

The choice of the algorithm depends on the amount of time available and on the structure of matrix $\boldsymbol{A}$. In specific favorable cases, the sub-optimal search algorithms described above (belonging to the first or the second approach) provide solutions having the same support than those of the ℓ_0-norm problem. For example, if the unknown signal $\boldsymbol{x}^\star$ is highly sparse and if the correlation between any pair of columns of $\boldsymbol{A}$ is limited, the ℓ_1-norm approximation provides optimal solutions [8], [11], [12]. In most cases, however, the only guarantee to recover the optimal support is to use the exhaustive search algorithm. When fast sub-optimal algorithms lead to unsatisfactory results, it is of great interest to develop "intermediate" sub-optimal algorithms providing more accurate solutions within a larger computation time, which nevertheless remains very small in comparison with the exhaustive search. The Orthogonal Least Squares algorithm (OLS) [20], which is sometimes confused with OMP [21], falls into this category of intermediate quality algorithms. The structure of OLS is the same as that of OMP, the difference being that at each iteration, OLS solves a large number of least-square problems ($n-k$, where k is the cardinal of the current active set) while OMP only performs the $n-k$ inner products between the current residual $\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}$ and the candidate columns $\boldsymbol{a}_i$ and chooses the column of $\boldsymbol{A}$ having the maximal inner product. OMP solves only one least-square problem per iteration, once the column to be inserted is selected (in order to update all the active set entries). In the following, we will propose a forward-backward extension of OLS allowing an insertion or a removal at each iteration, each iteration requiring to solve n least-square problems. It differs from the bidirectional search algorithm of Haugland [18] and the FoBa algorithm of Zhang [19] which are OMP forward-backward extensions.

The starting point of our forward-backward algorithm is the Single Most Likely Replacement (SMLR) algorithm, which proved to be a very efficient tool for the deconvolution of a sparse signal modeled as a Bernoulli-Gaussian process [22]–[25]. This approach relies on a Bayesian formulation of a deconvolution problem of the form $\boldsymbol{y}=\boldsymbol{A}\boldsymbol{x}+\boldsymbol{n}$ (where $\boldsymbol{A}$ denotes the convolution matrix) and on the maximum *a posteriori* (MAP) estimation of the sparse signal. The Bernoulli-Gaussian model is a probabilistic model

for sparse signals, in which (binary) Bernoulli random variables are associated to the position of the non zero values of $\boldsymbol{x}$, and the corresponding amplitudes are distributed according to an independent identically distributed (i.i.d.) centered Gaussian distribution of variance σ_x^2. SMLR is a deterministic ascent algorithm, which performs the optimization of the posterior likelihood of the support of $\boldsymbol{x}$ in a sub-optimal manner. It consists in updates (increase or decrease) of the support by one element, and the subsequent estimation of the amplitudes x_i.

Sparse signal approximation can be seen as a limit case of the Bernoulli-Gaussian restoration problem, in which the variance σ_x^2 of the amplitudes is set to infinity, because by definition, the ℓ_0-norm counts the number of non-zero values whatever their amplitudes. We will consider the limit case in which σ_x^2 tends to infinity and show that the MAP estimation of the weights $\boldsymbol{x}$ leads to an optimization problem which is close to the ℓ_0-constrained problem. This will result in an adaptation of the SMLR algorithm to the sparse approximation problem which relies on a single insertion or a single removal of an entry into/from the active set. The paper is organized as follows. In Section II, we introduce the Bernoulli-Gaussian model and the Bayesian framework in which we formulate the sparse signal approximation problem. In Section III, we adapt the SMLR algorithm resulting in the so-called Single Best Replacement (SBR) algorithm. In Section IV, a fast SBR implementation is proposed, based on the efficient update of the least-square error when the active set is modified by one element. Finally, Sections V and VI illustrate the method on the deconvolution with a Gaussian impulse response and on the joint detection of discontinuities at different orders in a signal, formulated as sparse signal approximation problems.

II. From Bernoulli-Gaussian signal restoration to sparse signal representation

The starting point of our study is the restoration of a sparse signal $\boldsymbol{x}$ from a linear observation $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{n}$, where $\boldsymbol{n}$ stands for the observation noise. An acknowledged probabilistic model dedicated to sparse signals is the Bernoulli-Gaussian (BG) model [22], [23], [25]. The BG model can actually lead to MAP and posterior mean estimators of the sparse signal, whose computation rely on optimization [25] and Monte Carlo Markov chain sampling, respectively [26]. We will first recall the known BG models and the formulation of sparse signal restoration in the Bayesian framework. Then, we will extend this formulation to a more general representation of sparse signals.

A. Preliminary definitions and working assumptions

Given an observation vector $\boldsymbol{y} \in \mathbb{R}^m$ and a dictionary $\boldsymbol{A} = [\boldsymbol{a}_1, \ldots, \boldsymbol{a}_n] \in \mathbb{R}^{m \times n}$, a subset selection algorithm aims at computing a weight vector $\boldsymbol{x} \in \mathbb{R}^n$ yielding an accurate approximation $\boldsymbol{y} \approx \boldsymbol{A}\boldsymbol{x}$ of

the observation. The columns $\boldsymbol{a}_i$ of $\boldsymbol{A}$ whose indices correspond to the non zero components x_i of $\boldsymbol{x}$ are referred to as the active (or selected) columns.

Throughout this paper, we do not make any assumption on the size of $\boldsymbol{A}$: m can be either lower or greater than n. Here, we will assume that $\boldsymbol{A}$ satisfies the unique representation property (URP). This assumption is classical in the sparse signal approximation literature, in the case where $m \leqslant n$ [27]. It is a stronger assumption than the full rank assumption. We now recall this definition and extend it to the case where $m \geqslant n$.

Definition 1 *When $m \leqslant n$, $\boldsymbol{A}$ satisfies the URP if and only if any selection of m columns of $\boldsymbol{A}$ forms a family of linearly independent vectors. When $m > n$, $\boldsymbol{A}$ satisfies the URP if and only if it is full rank.*

Before going further, let us mention that this assumption can be relaxed providing that the search strategy can guarantee that the selected columns of $\boldsymbol{A}$ result in a full rank matrix (see Section VI for details).

Under the URP assumption, when $m \leqslant n$, the system $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}$ has a number of solutions whose ℓ_0-norm are lower or equal to m: any active set of cardinality lower than m constitutes a possible support of such a solution. When $m > n$, there is generally no solution to $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}$ but the least-square estimator $\boldsymbol{x} = (\boldsymbol{A}^t\boldsymbol{A})^{-1}\boldsymbol{A}^t\boldsymbol{y}$ is unique, although not necessarily sparse.

Definition 2 *The support of a vector $\boldsymbol{x} \in \mathbb{R}^n$ is the set $\mathcal{S}(\boldsymbol{x}) \subseteq \{1, \ldots, n\}$ defined by $i \in \mathcal{S}(\boldsymbol{x})$ if and only if $x_i \neq 0$.*

Definition 3 *We denote by $\mathcal{Q} \subseteq \{1, \ldots, n\}$ the active set. Given $\mathcal{Q}$, we define the related vector $\boldsymbol{q} \in \{0,1\}^n$, by $q_i = 1$ if and only if $i \in \mathcal{Q}$. Let $\boldsymbol{A}_\mathcal{Q}$ be the matrix of size $m \times \mathrm{Card}\,[\mathcal{Q}]$ formed of the active columns of $\boldsymbol{A}$ ($\boldsymbol{a}_i$, $i \in \mathcal{Q}$), and let $\boldsymbol{t}$ be the reduced vector of size $\mathrm{Card}\,[\mathcal{Q}]$ gathering the values x_i for which $i \in \mathcal{Q}$. The observation model $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{n}$ also reads $\boldsymbol{y} = \boldsymbol{A}_\mathcal{Q}\boldsymbol{t} + \boldsymbol{n}$.*

Definition 4 *For all $\mathcal{Q} \subseteq \{1, \ldots, n\}$ such that $\mathrm{Card}\,[\mathcal{Q}] \leqslant \min(m, n)$, let $\boldsymbol{x}_\mathcal{Q}$ be the least-square solution and let $\mathcal{E}_\mathcal{Q}$ be the associated least-square error:*

$$\boldsymbol{x}_\mathcal{Q} \triangleq \underset{\mathcal{S}(\boldsymbol{x}) \subseteq \mathcal{Q}}{\arg\min} \{\mathcal{E}(\boldsymbol{x}) = \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|^2\} \tag{1}$$

$$\mathcal{E}_\mathcal{Q} \triangleq \mathcal{E}(\boldsymbol{x}_\mathcal{Q}) = \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}_\mathcal{Q}\|^2. \tag{2}$$

Notice that due to the URP assumption and because $\mathrm{Card}\,[\mathcal{Q}] \leqslant \min(m, n)$, $\boldsymbol{x}_\mathcal{Q}$ is uniquely defined.

B. Bernoulli-Gaussian models

A BG process can be defined as a random vector $\boldsymbol{x}$ ([1]) described by means of a Bernoulli random vector $\boldsymbol{q} \in \{0,1\}^n$ corresponding to the active set, and a Gaussian random vector $\boldsymbol{r} \sim \mathcal{N}(\boldsymbol{0}, \sigma_x^2 \boldsymbol{I}_n)$ such as each sample x_i of $\boldsymbol{x}$ is modeled as $x_i = q_i r_i$ [22], [23]. Here, $\boldsymbol{I}_n$ stands for the identity matrix of size $n \times n$. The Bernoulli random variables $q_i \sim \mathcal{B}(\rho)$ are i.i.d. They code for the presence ($q_i = 1$) or absence ($q_i = 0$) of signal at location i, the Bernoulli parameter $\rho = \Pr(q_i = 1)$ being the probability of presence of signal. The nonzero signal amplitudes r_i are controlled by their variance σ_x^2. Because $\boldsymbol{q}$ and $\boldsymbol{r}$ are independent random variables, the prior likelihoods of $\boldsymbol{q}$ and $\boldsymbol{x} = (\boldsymbol{q}, \boldsymbol{r})$ read:

$$l(\boldsymbol{q}) = \rho^{\|\boldsymbol{q}\|_0} (1-\rho)^{n-\|\boldsymbol{q}\|_0} \tag{3}$$

$$l(\boldsymbol{q}, \boldsymbol{r}) = l(\boldsymbol{r})\, l(\boldsymbol{q}) = g(\boldsymbol{r}; \sigma_x^2 \boldsymbol{I}_n)\, \rho^{\|\boldsymbol{q}\|_0} (1-\rho)^{n-\|\boldsymbol{q}\|_0}, \tag{4}$$

where $g(.\,; \boldsymbol{\Gamma})$ denotes the probability density function of the centered Gaussian distribution with covariance matrix $\boldsymbol{\Gamma}$.

C. Bayesian formulation of sparse signal restoration

The Bayesian formulation of an inverse problem of the form $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{n}$, where $\boldsymbol{n}$ stands for the observation noise, consists in inferring the distribution of $\boldsymbol{x} = (\boldsymbol{q}, \boldsymbol{r})$ knowing $\boldsymbol{y}$ using Bayes' rule. One can either infer the marginal distribution $l(\boldsymbol{q}|\boldsymbol{y})$ [25] or the joint distribution $l(\boldsymbol{q}, \boldsymbol{r}|\boldsymbol{y})$ [23], [24]. Following [23], we focus on the joint likelihood $l(\boldsymbol{q}, \boldsymbol{r}|\boldsymbol{y})$, leading to a cost function involving the least-square error $\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|^2$ and the ℓ_0-norm of $\boldsymbol{x}$.

Assuming an i.i.d. Gaussian noise distribution ($\boldsymbol{n} \sim \mathcal{N}(\boldsymbol{0}, \sigma_n^2 \boldsymbol{I}_m)$) and that the noise is independent from the sparse signal $\boldsymbol{x}$, the posterior likelihood $l(\boldsymbol{q}, \boldsymbol{r}|\boldsymbol{y})$ can be expressed using Bayes' rule. Denoting $\mathcal{L}(\boldsymbol{q}, \boldsymbol{r}) \triangleq -2\sigma_n^2 \log[l(\boldsymbol{q}, \boldsymbol{r}|\boldsymbol{y})]$, we have:

$$\begin{aligned} l(\boldsymbol{q}, \boldsymbol{r}|\boldsymbol{y}) &\propto g(\boldsymbol{y} - \boldsymbol{A}\boldsymbol{r}; \sigma_n^2 \boldsymbol{I}_m)\, g(\boldsymbol{r}; \sigma_x^2 \boldsymbol{I}_n)\, \rho^{\|\boldsymbol{q}\|_0} (1-\rho)^{n-\|\boldsymbol{q}\|_0}. \\ \mathcal{L}(\boldsymbol{q}, \boldsymbol{r}) &= \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{r}\|^2 + \frac{\sigma_n^2}{\sigma_x^2} \|\boldsymbol{r}\|^2 + 2\sigma_n^2 \log\left(\frac{1-\rho}{\rho}\right) \|\boldsymbol{q}\|_0 + \text{constant}(m, \sigma_n, n, \sigma_x), \end{aligned} \tag{5}$$

where $\propto$ indicates proportionality. Introducing the reduced vector $\boldsymbol{t}$ (see definition 3), the amplitudes $\boldsymbol{r}$ reread $\boldsymbol{r} = \{\boldsymbol{t}, \boldsymbol{u}\}$ (with $\boldsymbol{u} = \{r_i \mid q_i = 0\}$), and $\mathcal{L}(\boldsymbol{q}, \boldsymbol{r})$ takes the separable form $\mathcal{L}(\boldsymbol{q}, \boldsymbol{r}) = \mathcal{C}(\boldsymbol{q}, \boldsymbol{t}) + \sigma_n^2/\sigma_x^2 \|\boldsymbol{u}\|^2 + \text{constant}(m, \sigma_n, n, \sigma_x)$, where

$$\mathcal{C}(\boldsymbol{q}, \boldsymbol{t}) = \|\boldsymbol{y} - \boldsymbol{A}_{\mathcal{Q}} \boldsymbol{t}\|^2 + \frac{\sigma_n^2}{\sigma_x^2} \|\boldsymbol{t}\|^2 + 2\sigma_n^2 \log\left(\frac{1-\rho}{\rho}\right) \|\boldsymbol{q}\|_0. \tag{6}$$

[1]For convenience, we will use the same notations for random variables and their realization.

The minimization of (5) over $\{0,1\}^n \times \mathbb{R}^n$ leads to $\boldsymbol{u} = \mathbf{0}$. Finally, the joint MAP estimation problem consists of the minimization of (6) w.r.t. $(\boldsymbol{q}, \boldsymbol{t}) \in \{0,1\}^n \times \mathbb{R}^{\|\boldsymbol{q}\|_0}$.

Remark 1 *In* (6)*, the weight of* $\|\boldsymbol{q}\|_0$ *is non-negative if and only if* $\rho \leqslant 1/2$*. This condition imposes that in average, at least half of the samples* x_i *are equal to 0. This is coherent with the sparse assumption.*

D. Mixed ℓ_2-ℓ_0 minimization as a limit case

A sparse signal $\boldsymbol{x}$ is a signal for which a number of entries are equal to 0, *i.e.,* $\|\boldsymbol{x}\|_0 \leqslant k$ for some value of k. Since this definition does not involve constraints on the range of values of the non zero amplitudes, we choose to describe a sparse signal by a limit Bernoulli-Gaussian model, in which the amplitude variance σ_x^2 is set to infinity. The minimization of (6) thus rereads:

$$\min_{\boldsymbol{q},\boldsymbol{t}} \{\mathcal{C}(\boldsymbol{q}, \boldsymbol{t}) = \|\boldsymbol{y} - \boldsymbol{A}_{\mathcal{Q}}\boldsymbol{t}\|^2 + \lambda \|\boldsymbol{q}\|_0\}, \tag{7}$$

with $\lambda = 2\sigma_n^2 \log(1/\rho - 1)$. This compound criterion is composed of a quadratic data-fitting term, and a penalization term favoring the sparsity of the signal $\boldsymbol{x}$. The hyperparameter λ is related to the level of sparsity of the desired solution.

Theorem 1 *The above formulation* (7) *is equivalent to the following problem:*

$$\min_{\boldsymbol{x} \in \mathbb{R}^n} \{\mathcal{J}(\boldsymbol{x}; \lambda) \; = \; \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|^2 + \lambda \|\boldsymbol{x}\|_0\}, \tag{8}$$

which is referred to as the ℓ_0-penalized least-square problem. *The term "equivalent" means that given a minimizer* $(\boldsymbol{q}, \boldsymbol{t})$ *of* (7)*, the related vector* $\boldsymbol{x} = \{\boldsymbol{t}, \mathbf{0}\}$ *is a minimizer of* (8)*, and conversely, given a minimizer* $\boldsymbol{x}$ *of* (8)*, the vectors* $\boldsymbol{q}$ *and* $\boldsymbol{t}$ *defined as the support of* $\boldsymbol{x}$ *and its non-zero amplitudes, respectively, are such that* $(\boldsymbol{q}, \boldsymbol{t})$ *is a minimizer of* (7)*.*

Proof: To prove the equivalence, we first prove that $\min_{\boldsymbol{x}} \mathcal{J} = \min_{\boldsymbol{q},\boldsymbol{t}} \mathcal{C}$:

— Let $\boldsymbol{x}$ be a minimizer of $\mathcal{J}(.\,; \lambda)$. We set $\boldsymbol{q}$ to the support of $\boldsymbol{x}$ ($q_i = 1$ *if and only if* $x_i \neq 0$) and $\boldsymbol{t}$ to the non zero amplitudes of $\boldsymbol{x}$. Obviously, it follows that $\mathcal{J}(\boldsymbol{x}; \lambda) = \mathcal{C}(\boldsymbol{q}, \boldsymbol{t})$. Finally, $\min_{\boldsymbol{x}} \mathcal{J}(\boldsymbol{x}\,; \lambda) \geqslant \min_{\boldsymbol{q},\boldsymbol{t}} \mathcal{C}(\boldsymbol{q}, \boldsymbol{t})$.

— Let $(\boldsymbol{q}, \boldsymbol{t})$ be a minimizer of $\mathcal{C}$. Then, the vector $\boldsymbol{x}$ defined by $\boldsymbol{x} = \{\boldsymbol{t}, \mathbf{0}\}$ is such that $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{A}_{\mathcal{Q}}\boldsymbol{t}$ and $\|\boldsymbol{x}\|_0 = \|\boldsymbol{t}\|_0 \leqslant \|\boldsymbol{q}\|_0$. Therefore, $\mathcal{J}(\boldsymbol{x}; \lambda) \leqslant \mathcal{C}(\boldsymbol{q}, \boldsymbol{t})$. It follows that $\min_{\boldsymbol{q},\boldsymbol{t}} \mathcal{C}(\boldsymbol{q}, \boldsymbol{t}) \geqslant \min_{\boldsymbol{x}} \mathcal{J}(\boldsymbol{x}\,; \lambda)$.

In other words, we have $\min_{\boldsymbol{x}} \mathcal{J} = \min_{\boldsymbol{q},\boldsymbol{t}} \mathcal{C}$. We have also shown that the minimizers of both problems coincide, *i.e.,* are vectors describing identical signals. ■

In the following sections, we will focus on the minimization problem (8), involving the penalization term $\|\boldsymbol{x}\|_0$. The algorithm that will be developed hereafter is based on an efficient search of the support of $\boldsymbol{x}$. In that respect, the ℓ_0-penalized least-square problem does not drastically differ from the ℓ_0-constrained problem $\min \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|^2$ subject to $\|\boldsymbol{x}\|_0 \leqslant k$.

III. Adaptation of SMLR to ℓ_0-penalized least-square optimization

In this section, we propose to adapt the SMLR algorithm to the minimization of the mixed ℓ_2-ℓ_0 cost function $\mathcal{J}(\boldsymbol{x}\,;\,\lambda)$ defined in (8). However, to clearly distinguish SMLR which specifically aims at minimizing (6), the adapted algorithm will be termed as Single Best Replacement (SBR).

A. Principle of the SMLR algorithm

The Single Most Likely Replacement (SMLR) algorithm [22] is a deterministic coordinatewise ascent algorithm to maximize log-likelihood functions of the form $l(\boldsymbol{q}|\boldsymbol{y})$ (marginal MAP estimation) or $l(\boldsymbol{q},\boldsymbol{t}|\boldsymbol{y})$ (joint MAP estimation). In the latter case, it is worth noticing from (6) that given $\boldsymbol{q}$, the minimizer of $\mathcal{C}(\boldsymbol{q},\boldsymbol{t})$ w.r.t. $\boldsymbol{t}$ has a closed form expression $\boldsymbol{t} = \boldsymbol{t}(\boldsymbol{q})$. Consequently, the joint MAP estimation reduces to the minimization of the cost function $\mathcal{C}(\boldsymbol{q}) \triangleq \mathcal{C}(\boldsymbol{q},\boldsymbol{t}(\boldsymbol{q}))$ w.r.t. $\boldsymbol{q}$. At each SMLR iteration, all the possible single replacements of the support $\boldsymbol{q}$ (set $q_i = 1 - q_i$ while keeping the other q_j, $j \neq i$ unchanged) are tested, then the replacement yielding the maximal increase of $\mathcal{C}(\boldsymbol{q})$ is chosen. This task is repeated iteratively until no single replacement can increase $\mathcal{C}(\boldsymbol{q})$ anymore. The number of possible supports $\boldsymbol{q}$ being finite (2^n) and SMLR being an ascent algorithm, it terminates after a finite number of iterations.

Let us introduce some useful notations.

Definition 5 *For convenience, we will use the notation $\mathcal{Q} \bullet i$ to refer to a single replacement, i.e., the insertion ($\cup$) or removal ($\setminus$) of an index i into/from the active set $\mathcal{Q}$:*

$$\mathcal{Q} \bullet i \triangleq \begin{cases} \mathcal{Q} \cup \{i\} & \text{if } i \notin \mathcal{Q}, \\ \mathcal{Q} \setminus \{i\} & \text{otherwise.} \end{cases} \tag{9}$$

Definition 6 *For a given subset $\mathcal{Q}$ of $\{1,\ldots,n\}$ such that $\mathrm{Card}\,[\mathcal{Q}] \leqslant \min(m,n)$, we define the cost functions:*

$$\mathcal{J}_{\mathcal{Q}}(\lambda) \triangleq \mathcal{J}(\boldsymbol{x}_{\mathcal{Q}};\lambda) = \mathcal{E}_{\mathcal{Q}} + \lambda\|\boldsymbol{x}_{\mathcal{Q}}\|_0, \tag{10}$$

$$\mathcal{K}_{\mathcal{Q}}(\lambda) \triangleq \mathcal{E}_{\mathcal{Q}} + \lambda \mathrm{Card}\,[\mathcal{Q}]\,, \tag{11}$$

where the least-square solution $\boldsymbol{x}_{\mathcal{Q}}$ and the corresponding error $\mathcal{E}_{\mathcal{Q}}$ have been defined in (1) *and* (2)*.*

Obviously, $\mathcal{J}_{\mathcal{Q}}(\lambda) = \mathcal{K}_{\mathcal{Q}}(\lambda)$ *if and only if* the minimizer $\boldsymbol{x}_{\mathcal{Q}}$ has a support equal to $\mathcal{Q}$. In the next two paragraphs, we will introduce a first version of SBR involving $\mathcal{J}_{\mathcal{Q}}(\lambda)$ only, and then we will present an alternative (simpler) version relying on the computation of $\mathcal{K}_{\mathcal{Q}}(\lambda)$ instead of $\mathcal{J}_{\mathcal{Q}}(\lambda)$. We will discuss the extent to which both versions differ.

B. The Single Best Replacement algorithm (preliminary version)

The SMLR algorithm described above can be seen as an exploration strategy for discrete optimization rather than an algorithm specific to a posterior likelihood function. Here, we will use the same strategy to minimize the cost function $\mathcal{J}(\boldsymbol{x}; \lambda)$. However, we rename the algorithm Single Best Replacement (SBR) to remove the statistical connotation, the search strategy being applicable to cost functions which are not likelihood functions. The SBR algorithm works as follows. At each iteration, the n possible single replacements $\mathcal{Q} \bullet i$, $i = 1, \ldots, n$ are tested, then the best is selected, *i.e.*, the replacement yielding the maximal decrease of $\mathcal{J}(\boldsymbol{x}; \lambda)$. This task is repeated until $\mathcal{J}(\boldsymbol{x}; \lambda)$ cannot decrease anymore. We now detail one SBR iteration.

Given an active set $\mathcal{Q}$, the vector $\boldsymbol{x}_{\mathcal{Q}}$ defined in (1) is the corresponding least-square solution. For each index $i \in \{1, \ldots, n\}$, we compute the minimizer $\boldsymbol{x}_{\mathcal{Q}\bullet i}$ of $\mathcal{E}(\boldsymbol{x})$ whose support is included in $\mathcal{Q} \bullet i$, and we keep in memory the value of $\mathcal{J}_{\mathcal{Q}\bullet i}(\lambda) = \mathcal{J}(\boldsymbol{x}_{\mathcal{Q}\bullet i}\,;\, \lambda)$. Finally, we compute the minimum of $\mathcal{J}_{\mathcal{Q}\bullet i}(\lambda)$, $i = 1, \ldots, n$. If the minimal value is strictly lower than $\mathcal{J}_{\mathcal{Q}}(\lambda)$, then we select the index i yielding this minimal value:

$$\ell \quad \in \quad \underset{i \in \{1,\ldots,n\}}{\arg\min}\ \mathcal{J}_{\mathcal{Q}\bullet i}(\lambda). \tag{12}$$

The next SBR iterate is thus defined as $\mathcal{Q}' = \mathcal{Q} \bullet \ell$, yielding the vector $\boldsymbol{x}_{\mathcal{Q}'}$.

SBR terminates when none of the indices i yield a decrease of $\mathcal{J}$. Except when an initial support estimate (of cardinality lower than $\min(m, n)$) is available, we suggest to set the initial active set to the empty set.

Remark 2 (Relationship between SBR and SMLR) *We introduced SBR as the application of the SMLR search strategy to the ℓ_0-penalized least square cost function, which is obtained by taking the limit of the cost function* (6) *when σ_x tends towards infinity. In other words, we first considered the limit form of the cost function* (6)*, and then applied the search strategy. Conversely, applying SMLR to the cost function* (6) *and then, taking the limit of the SMLR formula when σ_x tends to infinity also yields the SBR algorithm.*

Actually, the main difference between the SMLR and SBR algorithms is that SMLR (which can take several forms depending on the use of the joint distribution $l(\boldsymbol{q},\boldsymbol{r}|\boldsymbol{y})$ or the marginal distribution $l(\boldsymbol{q}|\boldsymbol{y})$) involves the inversion of a matrix of the form $\boldsymbol{A}_{\mathcal{Q}}^t\boldsymbol{A}_{\mathcal{Q}}+\alpha\boldsymbol{I}_{\mathrm{Card}[\mathcal{Q}]}$ whereas SBR involves the inverse of the Gram matrix $\boldsymbol{A}_{\mathcal{Q}}^t\boldsymbol{A}_{\mathcal{Q}}$. For this reason, instabilities are likely to occur while using SBR in the cases where $\boldsymbol{A}_{\mathcal{Q}}$ is ill conditioned, for low λ-values. The use of the term $\alpha\boldsymbol{I}_{\mathrm{Card}[\mathcal{Q}]}$, which acts as a regularization on the amplitude values, avoids such a degeneracy while using SMLR, at the price of handling the additional hyperparameter α.

C. Slight modification of SBR (final version)

We introduce a slight modification of SBR by replacing (12) with:

$$\ell \in \underset{i\in\{1,\ldots,n\}}{\arg\min}\ \mathcal{K}_{\mathcal{Q}\bullet i}(\lambda). \tag{13}$$

We propose this modification because $\mathcal{K}_{\mathcal{Q}}(\lambda)=\mathcal{E}_{\mathcal{Q}}+\lambda\mathrm{Card}\,[\mathcal{Q}]$ can be computed more efficiently than $\mathcal{J}_{\mathcal{Q}}(\lambda)$, the computation of $\boldsymbol{x}_{\mathcal{Q}}$ being no longer necessary. The use of $\mathcal{K}_{\mathcal{Q}}(\lambda)$ makes the penalization term very easy to update when $\mathcal{Q}$ is modified by one element (add or remove λ), and the only necessary update is that of $\mathcal{E}_{\mathcal{Q}}$. The following theorem shows that there is almost surely no difference between both versions of SBR provided that the data $\boldsymbol{y}$ are corrupted with "non degenerate" noise.

Theorem 2 *Let $\boldsymbol{y}=\boldsymbol{y}_0+\boldsymbol{n}$, where $\boldsymbol{y}_0$ is a given vector of $\mathbb{R}^m$ and $\boldsymbol{n}$ is a random vector. We assume that $\boldsymbol{n}$ is an absolute continuous random vector, i.e., one that admits a probability density function w.r.t. the Lebesgue measure. Then, when $\mathrm{Card}\,[\mathcal{Q}]\leqslant\min(m,n)$, the probability that $\|\boldsymbol{x}_{\mathcal{Q}}\|_0<\mathrm{Card}\,[\mathcal{Q}]$ is equal to 0, i.e., $\|\boldsymbol{x}_{\mathcal{Q}}\|_0=\mathrm{Card}\,[\mathcal{Q}]$ almost surely.*

Proof: Let $k=\mathrm{Card}\,[\mathcal{Q}]$ and let $\boldsymbol{t}_{\mathcal{Q}}$ be the minimizer of $\|\boldsymbol{y}-\boldsymbol{A}_{\mathcal{Q}}\boldsymbol{t}\|^2$ over $\mathbb{R}^k$. Obviously, $\|\boldsymbol{x}_{\mathcal{Q}}\|_0=\|\boldsymbol{t}_{\mathcal{Q}}\|_0\leqslant k$. Let $\boldsymbol{V}_{\mathcal{Q}}=(\boldsymbol{A}_{\mathcal{Q}}^t\boldsymbol{A}_{\mathcal{Q}})^{-1}\boldsymbol{A}_{\mathcal{Q}}^t$ be the matrix of size $k\times m$ such that $\boldsymbol{t}_{\mathcal{Q}}=\boldsymbol{V}_{\mathcal{Q}}\boldsymbol{y}$. Denoting by $\boldsymbol{v}^1,\ldots,\boldsymbol{v}^k\in\mathbb{R}^m$ the row vectors of $\boldsymbol{V}_{\mathcal{Q}}$, $\|\boldsymbol{t}_{\mathcal{Q}}\|_0<k$ *if and only if* there exists i such that $\boldsymbol{y}^t\boldsymbol{v}^i=0$. Because $\boldsymbol{A}_{\mathcal{Q}}$ is full rank, $\boldsymbol{V}_{\mathcal{Q}}$ is full rank and then $\forall i$, $\boldsymbol{v}^i\neq\boldsymbol{0}$. Denoting by $\mathcal{H}^{\perp}(\boldsymbol{v}^i)$ the hyperplane of $\mathbb{R}^m$ which is orthogonal to $\boldsymbol{v}^i$, we have

$$\|\boldsymbol{x}_{\mathcal{Q}}\|_0<k \iff \boldsymbol{y}\in\bigcup_{i=1}^{k}\mathcal{H}^{\perp}(\boldsymbol{v}^i). \tag{14}$$

Because the set $\bigcup_i\mathcal{H}^{\perp}(\boldsymbol{v}^i)$ has a Lebesgue measure equal to zero and the random vector $\boldsymbol{y}$ admits a probability density function, the probability of event (14) is zero, thus $\Pr(\|\boldsymbol{x}_{\mathcal{Q}}\|_0<k)=0$. ■

TABLE I

SBR ALGORITHM (FINAL VERSION). BY DEFAULT, THE INITIAL ACTIVE SET IS EMPTY: $\mathcal{Q}_1 = \emptyset$.

Input: $\boldsymbol{A}$, $\boldsymbol{y}$, λ and active set $\mathcal{Q}_1$ of cardinality lower than $\min(m,n)$
Step 1: Set j to 1.
Step 2: For $i = 1$ to n,
Compute $\mathcal{K}_{\mathcal{Q}_j \bullet i}(\lambda)$.
End for.
Compute ℓ using (13).
If $\mathcal{K}_{\mathcal{Q}_j \bullet \ell}(\lambda) < \mathcal{K}_{\mathcal{Q}_j}(\lambda)$,
Set $\mathcal{Q}_{j+1} = \mathcal{Q}_j \bullet \ell$,
else,
Terminate SBR.
End if.
Step 3: Do $j = j + 1$ and go to step 2.
Output: active set $\mathcal{Q}_j = \mathsf{SBR}(\mathcal{Q}_1; \lambda)$

The above theorem implies that when dealing with real noisy data, it is almost sure that $\|\boldsymbol{x}_{\mathcal{Q}}\|_0 = \mathrm{Card}\,[\mathcal{Q}]$, *i.e.,* that no active component is exactly equal to 0. Thus, the original and slightly modified versions of SBR almost surely lead to exactly the same iterates. Even in the noiseless case, an active component is rarely numerically evaluated to 0 due to the round-off errors occurring during the numerical computations. In all cases, the modified version of SBR can be applied without restriction and the properties stated below (*e.g.,* the termination after a finite number of iterations) remain valid even when an SBR iterate satisfies $\|\boldsymbol{x}_{\mathcal{Q}}\|_0 < \mathrm{Card}\,[\mathcal{Q}]$.

For all these reasons, we will adopt the modified version of SBR in the rest of the paper. It is summarized in Table I.

D. Behavior and adaptations of SBR

Termination of SBR: SBR is a descent algorithm in the sense that the value of $\mathcal{K}_{\mathcal{Q}}(\lambda)$ is always decreasing. Consequently, a set $\mathcal{Q}$ cannot be explored twice and similarly to SMLR, SBR terminates after a finite number of iterations. The SBR output $\mathcal{Q}$ is a "local minimizer" of the function $\mathcal{Q} \mapsto \mathcal{K}_{\mathcal{Q}}(\lambda)$ in the sense that no replacement of $\mathcal{Q}$ with $\mathcal{Q} \bullet i$ yields a decrease of the cost: $\forall i,\, \mathcal{K}_{\mathcal{Q}}(\lambda) \leqslant \mathcal{K}_{\mathcal{Q} \bullet i}(\lambda)$.

Notice that the size of $\mathcal{Q}$ remains lower or equal to $\min(m,n)$. Indeed, if a set $\mathcal{Q}$ of cardinality

$\min(m,n)$ is reached, then $\mathcal{E}_{\mathcal{Q}}$ is equal to 0 due to the URP assumption. Then, any set $\mathcal{Q}'$ of the form $\mathcal{Q} \cup i$ yields a larger value $\mathcal{K}_{\mathcal{Q}'}(\lambda) = \mathcal{K}_{\mathcal{Q}}(\lambda) + \lambda$ of the cost function. We emphasize that no stopping condition is needed unlike many algorithms which require to set a maximum number of iterations (MP and variations, OLS) and/or a threshold on the squared error variation (CoSaMP, IHT).

Proposition 1 *Under the assumptions of Theorem 2, each SBR iterate $\boldsymbol{x}_{\mathcal{Q}}$ is almost surely a local minimizer of the ℓ_0-constrained problem*

$$\min_{\|\boldsymbol{x}\|_0 \leqslant k} \mathcal{E}(\boldsymbol{x}) \tag{15}$$

with $k = \mathrm{Card}\,[\mathcal{Q}]$. This property holds in particular for the SBR output.

Proof: Let $\boldsymbol{x} = \boldsymbol{x}_{\mathcal{Q}}$ be an SBR iterate and let $k = \mathrm{Card}\,[\mathcal{Q}]$. According to Theorem 2, $\|\boldsymbol{x}\|_0 = k$ almost surely. Setting $\varepsilon = \min_{i \in \mathcal{Q}} |x_i|$ ($\varepsilon > 0$), it is obvious that if $\boldsymbol{x}' \in \mathbb{R}^n$ satisfies $\|\boldsymbol{x}' - \boldsymbol{x}\|_2 < \varepsilon$, then $\forall i \in \mathcal{Q}$, $x'_i \neq 0$. Thus, $\|\boldsymbol{x}' - \boldsymbol{x}\|_2 < \varepsilon$ implies that $\mathcal{S}(\boldsymbol{x}) \subseteq \mathcal{S}(\boldsymbol{x}')$ and $\|\boldsymbol{x}'\|_0 \geqslant k$.

If $\boldsymbol{x}'$ satisfies $\|\boldsymbol{x}' - \boldsymbol{x}\|_2 < \varepsilon$ and $\|\boldsymbol{x}'\|_0 \leqslant k$, then necessarily, $\|\boldsymbol{x}'\|_0 = k$ and $\mathcal{S}(\boldsymbol{x}) = \mathcal{S}(\boldsymbol{x}')$. Since $\boldsymbol{x} = \boldsymbol{x}_{\mathcal{Q}}$, it follows that $\mathcal{E}(\boldsymbol{x}') \geqslant \mathcal{E}(\boldsymbol{x})$ almost surely. ■

OLS as a special case: When $\lambda = 0$, SBR coincides with the well known Orthogonal Least Squares (OLS) algorithm [20], [28]. The removal operation never occurs, because it automatically leads to an increase of the least-square cost $\mathcal{K}_{\mathcal{Q}}(0) = \mathcal{E}_{\mathcal{Q}}$. Consequently, only insertions are worth being tested ($\mathcal{Q}' = \mathcal{Q} \cup i$, $i \notin \mathcal{Q}$).

Empty solutions:

Proposition 2 (Empty solutions) *Denoting by $\lambda_{\max} \triangleq \max_i (\boldsymbol{a}_i^t \boldsymbol{y})^2 / \|\boldsymbol{a}_i\|^2$, the output of SBR($\emptyset; \lambda$) is equal to the empty set if and only if $\lambda \geqslant \lambda_{\max}$.*

Proof: SBR stops during its first iteration if all the insertion trials fail, *i.e.*,

$$\forall i,\ \mathcal{E}_{\{i\}} + \lambda \ \ \geqslant \ \ \mathcal{E}_{\emptyset} = \|\boldsymbol{y}\|^2. \tag{16}$$

For a given value of i, the minimum of $\|\boldsymbol{y} - x_i \boldsymbol{a}_i\|^2$ is reached when $x_i = \boldsymbol{a}_i^t \boldsymbol{y} / \|\boldsymbol{a}_i\|^2$, leading to $\mathcal{E}_{\{i\}} = \|\boldsymbol{y}\|^2 - (\boldsymbol{a}_i^t \boldsymbol{y})^2 / \|\boldsymbol{a}_i\|^2$. Thus, (16) is equivalent to the condition $\forall i$, $\lambda \geqslant (\boldsymbol{a}_i^t \boldsymbol{y})^2 / \|\boldsymbol{a}_i\|^2$, *i.e.*, to $\lambda \geqslant \lambda_{\max}$. ■

Reduced search: Instead of trying all the replacements $\mathcal{Q}' = \mathcal{Q} \bullet i$ at each SBR iteration, it is advantageous, if possible, to explore only a subset of these n replacements. We give two ideas to reduce the number of trials: the first idea is an acceleration of the SBR algorithm, yielding the same iterates with a slightly reduced search. The second idea is a modification of SBR.

Given an active set $\mathcal{Q}$, a removal $\mathcal{Q}' = \mathcal{Q}\backslash\{i\}$ yields an increase of the squared error and a decrease of the penalty equal to λ. Hence, the maximum decrease of the ℓ_0-penalized cost function which can be expected with a removal is λ: $\mathcal{K}_{\mathcal{Q}'}(\lambda) - \mathcal{K}_{\mathcal{Q}}(\lambda) \geqslant -\lambda$. Consequently, if a given insertion $\mathcal{Q}' = \mathcal{Q} \cup \{i\}$ is such that $\mathcal{K}_{\mathcal{Q}'}(\lambda) - \mathcal{K}_{\mathcal{Q}}(\lambda) < -\lambda$, then no removal can yield a larger decrease. The acceleration of the SBR algorithm thus consists in trying all the insertions first, and if the best insertion yields a decrease larger than λ, selecting the best insertion. Otherwise, all the removals need to be explored as stated in Table I. This acceleration does not alter the SBR iterates. However, the gain is limited when the level of sparsity is high, *i.e.*, when the number of removals to be tried is reduced.

Haugland and Zhang pointed out that in a forward-backward strategy, it can be helpful to favor removals [18], [19]. Adapted to SBR, this idea leads to a modified algorithm in which the removal operations are explored in a first pace, and the insertions are explored only if no removal yields a decrease of the cost function. If a removal decreases the cost, then the selected replacement is the removal yielding the maximal decrease.

In our experiments, the average performance of SBR and this modified version are quite comparable (there is no obvious gain or loss of quality nor a significant saving in computation time). Thus, in the following, we will keep the version of SBR presented on Table I for the sake of clarity.

IV. Implementation issues

Given the current active set $\mathcal{Q}$, an SBR iteration consists in computing the least-square error $\mathcal{E}_{\mathcal{Q}'}$ for all the configurations $\mathcal{Q}' = \mathcal{Q} \bullet i$, allowing the computation of $\mathcal{K}_{\mathcal{Q}'}(\lambda)$ using (11). We first present a basic implementation in which $\mathcal{E}_{\mathcal{Q}'}$ is computed independently of the knowledge of $\mathcal{E}_{\mathcal{Q}}$, and then an efficient implementation allowing a fast update when $\mathcal{Q}$ is modified. We will denote by $k \triangleq \mathrm{Card}\,[\mathcal{Q}]$ the cardinality of the active set.

A. Basic implementation

Given a support $\mathcal{Q} \subseteq \{1, \ldots, n\}$ of cardinality lower than $\min(m, n)$, (1) reduces to the unconstrained minimization of $\|\boldsymbol{y} - \boldsymbol{A}_{\mathcal{Q}}\boldsymbol{t}\|^2$ w.r.t. $\boldsymbol{t} \in \mathbb{R}^k$. Because $\boldsymbol{A}_{\mathcal{Q}}$ is full rank, the unconstrained problem has a unique minimizer that reads:

$$\boldsymbol{t}_{\mathcal{Q}} \;\triangleq\; \arg\min_{\boldsymbol{t}} \|\boldsymbol{y} - \boldsymbol{A}_{\mathcal{Q}}\boldsymbol{t}\|^2 \;=\; (\boldsymbol{A}_{\mathcal{Q}}^t \boldsymbol{A}_{\mathcal{Q}})^{-1} \boldsymbol{A}_{\mathcal{Q}}^t \boldsymbol{y} \tag{17}$$

and the minimal least-square error reads:

$$\mathcal{E}_{\mathcal{Q}} \;=\; \|\boldsymbol{y} - \boldsymbol{A}_{\mathcal{Q}}\boldsymbol{t}_{\mathcal{Q}}\|^2 \;=\; \|\boldsymbol{y}\|^2 - \boldsymbol{y}^t \boldsymbol{A}_{\mathcal{Q}} \boldsymbol{t}_{\mathcal{Q}}. \tag{18}$$

Finally, given an active set $\mathcal{Q}$, an SBR iteration involves the computation of $\boldsymbol{t}_{\mathcal{Q}'}$ and the corresponding error $\mathcal{E}_{\mathcal{Q}'}$ for all possible updates $\mathcal{Q}' = \mathcal{Q} \bullet i$ of $\mathcal{Q}$, using (17) and (18).

B. Recursive implementation of SBR

At each SBR iteration, n least-square problems of the form (17) must be solved, each requiring the inversion of the Gram matrix (of size $k \times k$)

$$\boldsymbol{G}_{\mathcal{Q}} \triangleq \boldsymbol{A}_{\mathcal{Q}}^t \boldsymbol{A}_{\mathcal{Q}}. \tag{19}$$

The computational cost can be high when the number of active entries k is large since in the general case, a matrix inversion costs $\mathcal{O}(k^3)$ scalar operations. Following an idea widely spread in the subset selection literature, we propose to use a recursive computation of the inverse of the Gram matrix.

A first possibility is to use the Gram-Schmidt procedure [20], [28] which yields an orthogonal decomposition of $\boldsymbol{A}_{\mathcal{Q}} = \boldsymbol{W}\boldsymbol{U}$, where $\boldsymbol{W}$ is an $m \times k$ matrix with orthogonal columns and $\boldsymbol{U}$ is a $k \times k$ upper triangular matrix. Although it yields an efficient updating strategy when including an index into the active set (leading to the update of $\boldsymbol{A}_{\mathcal{Q}'} = [\boldsymbol{A}_{\mathcal{Q}}, \boldsymbol{a}_i]$), the Gram-Schmidt procedure does not extend with the same level of efficiency when an index removal is considered [29].

An alternative possibility is to use the block matrix inversion lemma [30] allowing an efficient update of $\boldsymbol{G}_{\mathcal{Q}}^{-1}$ for both index insertion and removal. The reader is referred to [25] which proposed an efficient SMLR implementation based on the recursive update of matrices of the form $(\boldsymbol{G}_{\mathcal{Q}} + \alpha \boldsymbol{I}_k)^{-1}$. This approach can also be used with SBR. However, the matrix to update is $\boldsymbol{G}_{\mathcal{Q}}^{-1}$, thus numerical instabilities are likely to occur when the selected columns of $\boldsymbol{A}$ are highly correlated and for low λ-values.

A possible stable solution is based on the Cholesky factorization $\boldsymbol{G}_{\mathcal{Q}} = \boldsymbol{L}_{\mathcal{Q}} \boldsymbol{L}_{\mathcal{Q}}^t$, where $\boldsymbol{L}_{\mathcal{Q}}$ is a lower triangular matrix. Updating $\boldsymbol{L}_{\mathcal{Q}}$ rather than $\boldsymbol{G}_{\mathcal{Q}}^{-1}$ is advantageous, since $\boldsymbol{L}_{\mathcal{Q}}$ is better conditioned. Its update can be easily done in the insertion case [31] but the removal case necessitates more care, as a removal breaks the structure of the lower triangular matrix $\boldsymbol{L}_{\mathcal{Q}}$. Ge *et al.* recently proposed a stable implementation of SMLR [32] which relies on the recursive update of the Cholesky factor of $\boldsymbol{G}_{\mathcal{Q}}^{-1}$. Here, we propose a slightly simpler strategy that relies on the factorization of the Gram matrix $\boldsymbol{G}_{\mathcal{Q}}$ itself.

C. Efficient strategy based on the Cholesky factorization

First, we notice that any new column $\boldsymbol{a}_i$ can be inserted at the last location in $\boldsymbol{A}_{\mathcal{Q}\cup i}$, since the value of $\mathcal{E}_{\mathcal{Q}\cup i}$ does not depend on the position of $\boldsymbol{a}_i$ in matrix $\boldsymbol{A}_{\mathcal{Q}\cup i}$. On the contrary, when removing a column

from the active set, we do not know *a priori* the position of the column to be removed, thus it cannot be assumed to be the last column of $\boldsymbol{A}_{\mathcal{Q}}$.

Given an active set $\mathcal{Q}$ and the corresponding matrix $\boldsymbol{A}_{\mathcal{Q}}$ of size $m \times k$, we will describe the cases where:

- a non active index $i \notin \mathcal{Q}$ is included after the other columns: $\boldsymbol{A}_{\mathcal{Q}'} = [\boldsymbol{A}_{\mathcal{Q}}, \boldsymbol{a}_i]$;
- an active index $i \in \mathcal{Q}$ is to be removed, the column $\boldsymbol{a}_i$ being in an arbitrary position.

As a symmetric positive-definite matrix, $\boldsymbol{G}_{\mathcal{Q}}$ reads $\boldsymbol{G}_{\mathcal{Q}} = \boldsymbol{L}_{\mathcal{Q}}\boldsymbol{L}_{\mathcal{Q}}^t$ where the Cholesky factor $\boldsymbol{L}_{\mathcal{Q}}$ is a lower triangular matrix of size $k \times k$. Applying (17), the least-square minimizer reads $\boldsymbol{t}_{\mathcal{Q}} = \boldsymbol{L}_{\mathcal{Q}}^{-t}\boldsymbol{L}_{\mathcal{Q}}^{-1}\boldsymbol{A}_{\mathcal{Q}}^t\boldsymbol{y}$ where the superscript $-t$ refers to the inverse transposition operator, and using (18), the cost function rereads:

$$\mathcal{K}_{\mathcal{Q}}(\lambda) \;=\; \mathcal{E}_{\mathcal{Q}}(\lambda) + \lambda k \;=\; \|\boldsymbol{y}\|^2 - \|\boldsymbol{L}_{\mathcal{Q}}^{-1}\boldsymbol{A}_{\mathcal{Q}}^t\boldsymbol{y}\|^2 + \lambda k. \tag{20}$$

Given $\boldsymbol{L}_{\mathcal{Q}}$, its computation costs $\mathcal{O}(k^2)$ scalar operations to solve the triangular system $\boldsymbol{L}_{\mathcal{Q}}^{-1}(\boldsymbol{A}_{\mathcal{Q}}^t\boldsymbol{y})$.

Insertion of a new column after the existing columns: Given an active set $\mathcal{Q}$ of size k, including a new index into $\mathcal{Q}$ leads to $\boldsymbol{A}_{\mathcal{Q}'} = [\boldsymbol{A}_{\mathcal{Q}}, \boldsymbol{a}_i]$. Thus, the new Gram matrix can be expressed as a 2×2 block matrix:

$$\boldsymbol{G}_{\mathcal{Q}'} = \begin{bmatrix} \boldsymbol{G}_{\mathcal{Q}} & \boldsymbol{A}_{\mathcal{Q}}^t\boldsymbol{a}_i \\ (\boldsymbol{A}_{\mathcal{Q}}^t\boldsymbol{a}_i)^t & \|\boldsymbol{a}_i\|^2 \end{bmatrix}, \tag{21}$$

and the Cholesky factor of $\boldsymbol{G}_{\mathcal{Q}'}$ can be straightforwardly updated:

$$\boldsymbol{L}_{\mathcal{Q}'} = \begin{bmatrix} \boldsymbol{L}_{\mathcal{Q}} & \boldsymbol{0} \\ \boldsymbol{l}_{\mathcal{Q},i}^t & \alpha_{\mathcal{Q},i} \end{bmatrix}, \tag{22}$$

with $\boldsymbol{l}_{\mathcal{Q},i} = \boldsymbol{L}_{\mathcal{Q}}^{-1}\boldsymbol{A}_{\mathcal{Q}}^t\boldsymbol{a}_i$ and $\alpha_{\mathcal{Q},i} = \sqrt{\|\boldsymbol{a}_i\|^2 - \|\boldsymbol{l}_{\mathcal{Q},i}\|^2}$.

The computation of $\mathcal{K}_{\mathcal{Q}'}(\lambda)$ using (20) leads to two inversions of triangular systems (computation of $\boldsymbol{l}_{\mathcal{Q},i}$ and computation of $\mathcal{K}_{\mathcal{Q}'}(\lambda)$). Advantageously, by computing

$$\mathcal{K}_{\mathcal{Q}'}(\lambda) - \mathcal{K}_{\mathcal{Q}}(\lambda) \;=\; \lambda - \left(\boldsymbol{l}_{\mathcal{Q},i}^t\boldsymbol{L}_{\mathcal{Q}}^{-1}\boldsymbol{A}_{\mathcal{Q}}^t\boldsymbol{y}\right)^2/\alpha_{\mathcal{Q},i}^2, \tag{23}$$

the cost can be reduced up to the pre-computation and storage of $\boldsymbol{L}_{\mathcal{Q}}^{-1}(\boldsymbol{A}_{\mathcal{Q}}^t\boldsymbol{y})$ at the beginning of the SBR iteration. The computation of $\mathcal{K}_{\mathcal{Q}'}(\lambda)$ only requires one inversion of a triangular system (computation of $\boldsymbol{l}_{\mathcal{Q},i}$).

Removal of an arbitrary column: When removing a column $\boldsymbol{a}_i$, updating $\boldsymbol{L}_{\mathcal{Q}}$ remains possible, although slightly more expensive. This idea was first developed by Ge *et al.* [32], who update the Cholesky factorization of matrix $\boldsymbol{G}_{\mathcal{Q}}^{-1}$. We adapt it to the direct factorization of $\boldsymbol{G}_{\mathcal{Q}}$. Let I be the index

such that $\boldsymbol{a}_i$ is the I-th column of $\boldsymbol{A}_{\mathcal{Q}}$ (with $1 \leqslant I \leqslant k$). Matrix $\boldsymbol{L}_{\mathcal{Q}}$ can be written in a block matrix form:

$$\boldsymbol{L}_{\mathcal{Q}} = \begin{bmatrix} \boldsymbol{A} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{b}^t & d & \boldsymbol{0} \\ \boldsymbol{C} & \boldsymbol{e} & \boldsymbol{F} \end{bmatrix}, \tag{24}$$

where the lowercase characters refer to the scalar (d) and vector quantities ($\boldsymbol{b}$, $\boldsymbol{e}$) which appear at the I-th row and at the I-th column. The computation of $\boldsymbol{G}_{\mathcal{Q}} = \boldsymbol{L}_{\mathcal{Q}}\boldsymbol{L}_{\mathcal{Q}}^t$ and the removal of the I-th row and the I-th column in $\boldsymbol{G}_{\mathcal{Q}}$ leads to

$$\boldsymbol{G}_{\mathcal{Q}'} = \begin{bmatrix} \boldsymbol{A} & \boldsymbol{0} \\ \boldsymbol{C} & \boldsymbol{F} \end{bmatrix} \begin{bmatrix} \boldsymbol{A}^t & \boldsymbol{C}^t \\ \boldsymbol{0} & \boldsymbol{F}^t \end{bmatrix} + \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{e} \end{bmatrix} \begin{bmatrix} \boldsymbol{0} & \boldsymbol{e}^t \end{bmatrix}. \tag{25}$$

By identification of this expression with the Cholesky factorization $\boldsymbol{G}_{\mathcal{Q}'} = \boldsymbol{L}_{\mathcal{Q}'}\boldsymbol{L}_{\mathcal{Q}'}^t$ and because the Cholesky factorization is unique, $\boldsymbol{L}_{\mathcal{Q}'}$ necessarily reads:

$$\boldsymbol{L}_{\mathcal{Q}'} = \begin{bmatrix} \boldsymbol{A} & \boldsymbol{0} \\ \boldsymbol{C} & \boldsymbol{X} \end{bmatrix}, \tag{26}$$

where $\boldsymbol{X}$ is a lower triangular matrix satisfying

$$\boldsymbol{X}\boldsymbol{X}^t = \boldsymbol{F}\boldsymbol{F}^t + \boldsymbol{e}\boldsymbol{e}^t. \tag{27}$$

The problem of computing $\boldsymbol{X}$ from $\boldsymbol{F}$ and $\boldsymbol{e}$ is classical; it is known as a positive rank 1 Cholesky update (update of the Cholesky factor $\boldsymbol{F}$ corresponding to a rank 1 update of the matrix $\boldsymbol{F}\boldsymbol{F}^t$ to be decomposed), and there exists a stable algorithm in $\mathcal{O}(f^2)$ operations, where $f = k - I$ is the size of $\boldsymbol{F}$ [33].

Finally, the computation of $\mathcal{K}_{\mathcal{Q}'}(\lambda)$ involves a positive Cholesky update and a triangular system inversion in (20). Thus, its overall cost is in $\mathcal{O}(k^2)$. Notice that matrix $\boldsymbol{F}$ is of size $k - I$. Therefore, the cost of a Cholesky update completely depends on the position I of the column $\boldsymbol{a}_i$ to be removed. The larger I, the more expensive is the Cholesky update.

D. Memory requirements and computation burden

The efficient (fast and stable) procedure is finally summarized in Table II. Given the current active set $\mathcal{Q}$, the index ℓ defining the next SBR iterate $\mathcal{Q} \bullet \ell$ is chosen according to (13) and $\boldsymbol{L}_{\mathcal{Q}\bullet\ell}$ is finally updated. No update of the amplitudes is necessary. If needed, their computation can be done using (17) and the knowledge of $\boldsymbol{L}_{\mathcal{Q}}$.

TABLE II

EFFICIENT IMPLEMENTATION OF AN ELEMENTARY SBR ITERATION.

Input: $\mathcal{Q}$, λ
Pre-computed quantities: $\boldsymbol{A}^t\boldsymbol{y}$ and $\|\boldsymbol{a}_i\|^2$ for all i
Stored quantities: $\mathcal{K}_{\mathcal{Q}}(\lambda)$, $\boldsymbol{L}_{\mathcal{Q}}$ and $\boldsymbol{L}_{\mathcal{Q}}^{-1}(\boldsymbol{A}_{\mathcal{Q}}^t\boldsymbol{y})$
Set $k = \mathrm{Card}\,[\mathcal{Q}]$.
Set $\ell = 0$.
Set `least_cost` $= \mathcal{K}_{\mathcal{Q}}(\lambda)$.
For $i = 1$ to n,
If $i \notin \mathcal{Q}$,
`/* Try the insertion of` i `*/`
Compute $\boldsymbol{l}_{\mathcal{Q},i} = \boldsymbol{L}_{\mathcal{Q}}^{-1}\boldsymbol{A}_{\mathcal{Q}}^t\boldsymbol{a}_i$ and $\mathcal{K}_{\mathcal{Q}'}(\lambda)$ using (23).
else,
`/* Try the removal of` i `*/`
Update the Cholesky decomposition (27): $\boldsymbol{X} =$`cholupdate`$(\boldsymbol{F}, \boldsymbol{e}, '+')$
Compute $\boldsymbol{L}_{\mathcal{Q}'}$ and $\mathcal{K}_{\mathcal{Q}'}(\lambda)$ using (26) and (20).
End if.
If $\mathcal{K}_{\mathcal{Q}'}(\lambda) <$ `least_cost`,
Set $\ell = i$.
Do `least_cost` $= \mathcal{K}_{\mathcal{Q}'}(\lambda)$.
End if.
End for.
If $\ell = 0$,
Terminate SBR.
else, `/* Perform the single replacement */`
Set $\mathcal{Q}' = \mathcal{Q} \bullet \ell$ and $\mathcal{K}_{\mathcal{Q}'}(\lambda) =$ `least_cost`.
Compute $\boldsymbol{L}_{\mathcal{Q}'} = \boldsymbol{L}_{\mathcal{Q}\bullet\ell}$ using (22) or (26), and then $\boldsymbol{L}_{\mathcal{Q}'}^{-1}(\boldsymbol{A}_{\mathcal{Q}'}^t\boldsymbol{y})$.
End if.
Output: next SBR iterate $\mathcal{Q}' = \mathcal{Q} \bullet \ell$, $\mathcal{K}_{\mathcal{Q}'}(\lambda)$, $\boldsymbol{L}_{\mathcal{Q}'}$ and $\boldsymbol{L}_{\mathcal{Q}'}^{-1}(\boldsymbol{A}_{\mathcal{Q}'}^t\boldsymbol{y})$

The actual implementation may vary depending on the size and the structure of matrix $\boldsymbol{A}$. We now detail the main possible implementations and their requirements in terms of storage and computation. Regarding the computation burden, we count the number of elementary operations, expressed in terms of scalar multiplications, since the cost of a scalar addition is negligible with respect to that of a multiplication.

When $\boldsymbol{A}$ is relatively small, one can take advantage of the situation by computing the full Gram

matrix $\boldsymbol{A}^t\boldsymbol{A}$ prior to any SBR iteration (storage of n^2 scalar elements). Its storage avoids recomputing vectors $\boldsymbol{A}_{\mathcal{Q}}^t\boldsymbol{a}_i$ which are needed whenever the insertion of $\boldsymbol{a}_i$ into the active set is tried. Similarly, we systematically store the values $\|\boldsymbol{a}_i\|^2$ ($i = 1, \ldots, n$) and $\boldsymbol{A}^t\boldsymbol{y}$ in two 1D arrays of size n, prior to any SBR loop. The storage of the other quantities (mainly $\boldsymbol{L}_{\mathcal{Q}}$) that are being updated in the SBR loops amounts to $\mathcal{O}(k^2)$ scalar elements, and each trial costs $\mathcal{O}(k^2)$ elementary operations, as it involves the inversion of a triangular system of size $k \times k$, plus a positive rank 1 Cholesky update in the removal case. This cost must be compared with the $\mathcal{O}(k^3)$ scalar operations which are necessary when inverting the Gram matrix in the basic implementation of SBR.

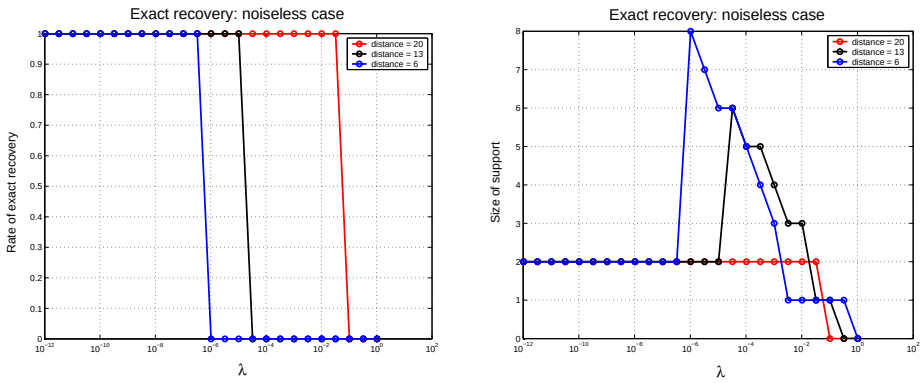
When $\boldsymbol{A}$ is larger, the computation of $\boldsymbol{A}^t\boldsymbol{A}$ is not possible anymore, and vectors $\boldsymbol{A}_{\mathcal{Q}}^t\boldsymbol{a}_i$ must be recomputed at any SBR iteration, for each insertion trial $\mathcal{Q}' = \mathcal{Q} \cup \{i\}$. The computation of $\boldsymbol{A}_{\mathcal{Q}}^t\boldsymbol{a}_i$ costs km elementary operations. It is a great burden and actually the main part of the cost corresponding to the trial of one single replacement, since the remaining part is in $\mathcal{O}(k^2)$ and for sparse representations, k is expected to be much lower than m. The cost of a single replacement finally amounts to $\mathcal{O}(k^2) + \mathcal{O}(km)$ elementary operations.

When the dictionary $\boldsymbol{A}$ has some specific structure, the above storage limitation can be alleviated, enabling a fast implementation even for large values of n. For instance, if $\boldsymbol{A}^t\boldsymbol{A}$ is a sparse matrix (*i.e.*, a large number of pairs of columns of $\boldsymbol{A}$ are orthogonal to each other), it can be stored as a sparse array in the sense that only the non-zero elements and their indices are stored. Also, deconvolution problems enable a fast implementation, since $\boldsymbol{A}^t\boldsymbol{A}$ is then a Toeplitz matrix (except for a north-west and/or a south-east submatrix in some cases of boundary conditions); the knowledge of the auto-correlation of the impulse response is sufficient to completely describe the matrix or a large part of it.

V. Deconvolution of a sparse signal with a Gaussian impulse response

We will analyze the behavior and performance of the proposed algorithm on two difficult problems, in which the dictionaries are highly correlated: the deconvolution of a sparse signal with a Gaussian impulse response, and the joint detection of discontinuities at different orders in a signal (section VI). The first problem is a typical problem for which the SMLR algorithm was introduced [25]. It affords to study the ability of SBR to perform an exact recovery in a simple noiseless case (separation of two Gaussian features from noiseless data) and to roughly understand the behavior of SBR in a noisy case (approximation of a larger number of features from noisy data).

In the following and for simulated problems, we will denote by $\boldsymbol{x}^\star$ the exact (known) sparse signal and we will generate noisy data according to $\boldsymbol{y} = \boldsymbol{y}^\star + \boldsymbol{n} = \boldsymbol{A}\boldsymbol{x}^\star + \boldsymbol{n}$, where $\boldsymbol{y}^\star = \boldsymbol{A}\boldsymbol{x}^\star$ denotes



(a) Exact recovery (binary) (b) Size of support

Fig. 1. Separation of two Gaussian features from noiseless data. Behavior of the reconstruction $\widehat{\mathcal{Q}}(\lambda; d)$ as a function of λ and of the distance d between both Gaussian features. For a given d-value, the curves $\lambda \mapsto true(d, \lambda)$ (a) and $\lambda \mapsto size(d, \lambda)$ (b) indicate the exact recovery (binary value) and the size of the support $\widehat{\mathcal{Q}}(\lambda; d)$, respectively.

the noiseless data and $\boldsymbol{n}$ stands for the observation noise. The dictionary columns $\boldsymbol{a}_i$ will always be normalized: $\|\boldsymbol{a}_i\|^2 = 1$. The signal to noise ratio (SNR) is defined by SNR $= 10\ \log_{10}(P_Y/P_N)$, where $P_Y = \|\boldsymbol{y}^\star\|^2/m$ is the average power of the noiseless data and P_N is the variance of the noise process $\boldsymbol{n}$.

A. Dictionary and simulated data

The impulse response $\boldsymbol{h}$ is a Gaussian signal of standard deviation σ, sampled on a regular grid at integer locations. For convenience reasons, it is approximated by a finite impulse response of length 6σ by thresholding the least values. The deconvolution problem leads to a Toeplitz matrix $\boldsymbol{A}$ whose columns $\boldsymbol{a}_i$ are obtained by shifting the signal $\boldsymbol{h}$. The dimension of $\boldsymbol{A}$ is chosen in such a way that any Gaussian feature resulting from the convolution $\boldsymbol{h} * \boldsymbol{x}^\star$ belongs to the observation window $\{1, \ldots, m\}$. This implies that $\boldsymbol{A}$ is slightly overcomplete ($m > n$). Denoting by $n_h = 1 + 2round(3\sigma)$ the size of the support of $\boldsymbol{h}$, the data size reads $m = n + n_h - 1$. Setting a large σ-value yields a high correlation between the neighboring columns of the dictionary.

B. Separation of two close Gaussian features

We first analyze the ability of SBR to separate two Gaussian features from noiseless data ($\|\boldsymbol{x}^\star\|_0 = 2$). The centers of both Gaussian features lay at a relative distance d and their amplitude is set to 1. We generate the corresponding noiseless data $\boldsymbol{y}^\star$ and we run SBR$(\emptyset; \lambda)$ for a number of predefined λ-values.

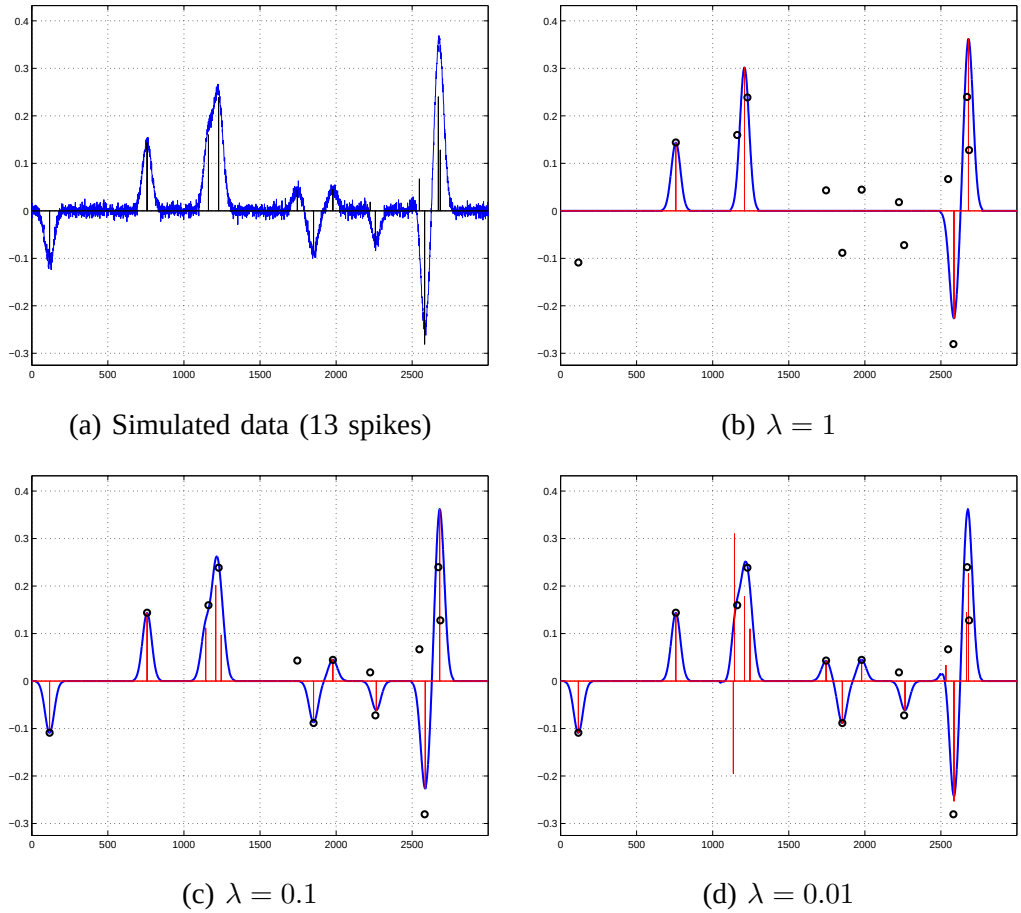
We analyze the SBR outputs $\widehat{\mathcal{Q}}(\lambda;d)$ by testing if $\widehat{\mathcal{Q}}(\lambda;d)$ is the true support $\mathcal{S}(\boldsymbol{x}^\star)$ and by computing its size. For each d-value, the procedure yields two curves $\lambda \mapsto true(d,\lambda)$ and $\lambda \mapsto size(d,\lambda)$.

Fig. 1 shows the curves obtained for a problem of size 300×270 ($m = 300, \sigma = 5$, and $n_h = 31$). These results correspond to distances equal to $d = 20$, 13 and 6 samples (red, black, and blue curves). It is noticeable that the exact recovery is always reached provided that λ is sufficiently small. This result remains true even for smaller distances (for all $d \geqslant 2$). The curve $\lambda \mapsto size(d,\lambda)$ illustrates that when the Gaussian features strongly overlap (*i.e.*, for $d = 13$ and 6), the size of the support obtained as output increases while λ decreases, and then for lower λ-values, removals start to occur, making the exact recovery possible. On the contrary, forward methods such as OMP and OLS start by positioning a (false) Gaussian feature in between the two Gaussians in their first iteration; this early false detection disables a true recovery in the further iterations.

C. Behavior of SBR for noisy data

In order to understand the behavior of SBR, we run SBR on more realistic noisy data and on a larger dimension problem ($m = 3000$ samples). The unknown sparse signal $\boldsymbol{x}^\star$ is generated by using the Bernoulli-Gaussian model introduced in Section II and is composed of $\|\boldsymbol{x}^\star\|_0 = 13$ Gaussian features. The impulse response $\boldsymbol{h}$ is of size $n_h = 181$ ($\sigma = 30$) yielding an observation matrix $\boldsymbol{A}$ of size 3000×2820, and the SNR is set to 20 dB.

Fig. 2 displays the simulated data and the SBR results obtained with a few λ-values. When λ decreases, the SBR approximations are of better quality but less sparse. The main Gaussian features are first found for large λ-values, and when λ decreases, the smaller features are being recovered. Removals rarely occur for coarse approximations. They occur more frequently when two spikes are overlapping and for low λ-values. For the reconstruction of Fig. 2, the exact support of $\boldsymbol{x}^\star$ is not found. However, it must be stressed that the columns of $\boldsymbol{A}$ are highly correlated and the approximations provided by SBR are of very good quality. When $\lambda = 0.01$, two very close neighboring columns of $\boldsymbol{A}$ are selected and both belong to the active set. Thus, the submatrix $\boldsymbol{A}_\mathcal{Q}$ formed of the active columns of $\boldsymbol{A}$ is ill conditioned. Despite the use of the Cholesky decomposition of the Gram matrix $\boldsymbol{G}_\mathcal{Q} = \boldsymbol{A}_\mathcal{Q}^t \boldsymbol{A}_\mathcal{Q}$, these highly correlated columns provoke numerical instability leading to degenerate amplitude values. We believe that this problem is not due to SBR itself but to the low level of regularization. The same problem occurs while running OLS for more than 14 iterations.



(a) Simulated data (13 spikes) (b) $\lambda = 1$

(c) $\lambda = 0.1$ (d) $\lambda = 0.01$

Fig. 2. Gaussian deconvolution results. Problem of size 3000×2820 ($\sigma = 30$). (a) Generated data signal, with 13 Gaussian features ($\|\boldsymbol{x}^\star\|_0$=13) and with SNR = 20 dB. (b,c,d) Sparse approximations of the data with empirical settings of λ: SBR outputs and data approximations. The amplitudes $\widehat{\boldsymbol{x}}$ are shown in red. The SBR outputs (supports) are of size 4, 10 and 14, respectively. The time of reconstruction always remains below 2 seconds (Matlab implementation).

VI. Joint detection of discontinuities at different orders in a signal

We now consider a more general problem, the joint detection of discontinuities at different orders $p = 0, \ldots, P$ in a signal. We will handle simulated and real data, and compare the performance of SBR with respect to other sparse approximation algorithms (OMP and OLS) in terms of approximation accuracy and computation time.

In a preliminary step, we formulate the detection of discontinuities at a single order p as a spline approximation problem. Then, we will take advantage of this formulation to introduce more easily the joint detection problem.

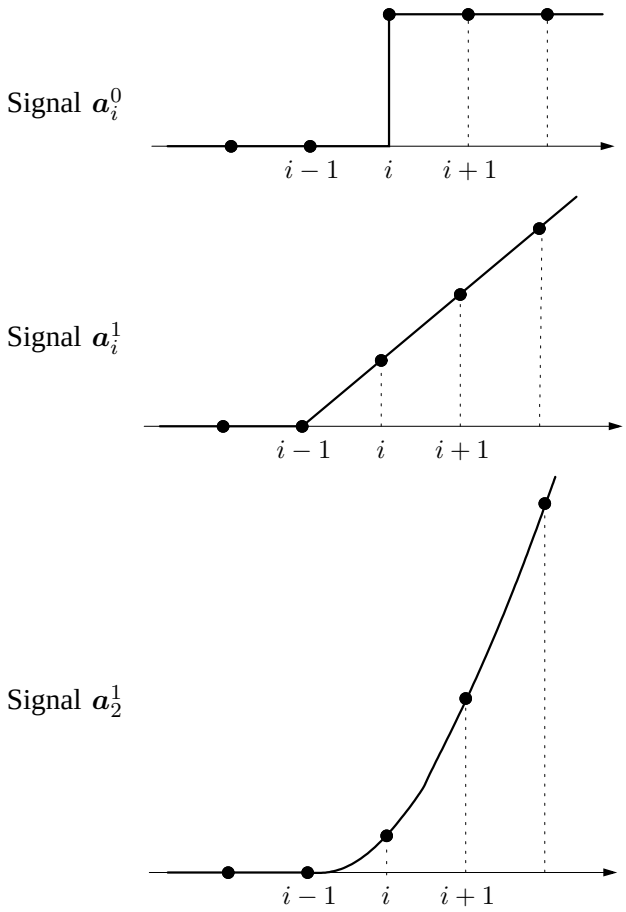


Fig. 3. The elementary signal $\boldsymbol{a}_i^p$ associated to a p-th order discontinuity at location i. $\boldsymbol{a}_i^0$ is the Heaviside step function, $\boldsymbol{a}_i^1$ is the ramp function and $\boldsymbol{a}_i^2$ is a truncated quadratic function. Each function is equal to 1 at location i, and its support is the interval $\{i, \ldots, m\}$.

A. Approximation of a spline of degree p

In the continuous case, a signal is a spline of degree p with k knots *if and only if* its $(p+1)$-th derivative is a stream of k weighted Diracs [34]. In the discrete case, we introduce the dictionary $\boldsymbol{A}^p$ formed of signals which are shifted versions of the one-sided power function $k \mapsto k_+^p \triangleq [\max(k, 0)]^p$ for all possible shifts (see Fig. 3). $\boldsymbol{A}^p$ represents the integration operator of degree $p+1$. Denoting by $\{1, \ldots, m\}$ the support of the data signal $\boldsymbol{y}$, the shifted signals $\boldsymbol{a}_i^p$ (for $i \in \{1, \ldots, m\}$) read

$$\forall k \in \{1, \ldots, m\},\ \boldsymbol{a}_i^p(k) = (k - i + 1)_+^p \tag{28}$$

and their support is equal to $\{i, \ldots, m\}$. Finally, we form the dictionary $\boldsymbol{A}^p = [\boldsymbol{a}_1^p, \ldots, \boldsymbol{a}_{m-p}^p]$ of size $m \times (m-p)$. It does not make sense to allow the occurrence of a p-th order discontinuity for the last

samples (*i.e.,* to include $\boldsymbol{a}_i^p$ for $i > m-p$) since the spline approximation would require to reconstruct a polynomial of degree p in the range $\{i,\dots,m\}$ from less than $p+1$ data samples.

We address the spline approximation problem as the sparse approximation of $\boldsymbol{y}$ by the piecewise polynomial $\boldsymbol{g}^p = \boldsymbol{A}^p\boldsymbol{x}^p$. The vector $\boldsymbol{x}^p$ refers to the $(p+1)$-th discrete derivatives of the approximate signal $\boldsymbol{g}^p$; its non zero values x_i^p code for the amplitude of a jump at location i ($p=0$), the change of slope at location i ($p=1$), *etc.* In brief, the sparse approximation of $\boldsymbol{y}$ provides the detection of the location (i) of the p-th order discontinuities and the estimation of their change of amplitudes (x_i^p).

B. Approximation of a piecewise polynomial of maximum degree P

Following [34], we formulate this problem as the joint detection of discontinuities at orders $p = 0,\dots,P$. Let us append the elementary dictionaries $\boldsymbol{A}^p$ in a global dictionary $\boldsymbol{A} = [\boldsymbol{A}^0,\dots,\boldsymbol{A}^P]$. The approximation $\boldsymbol{g} = \boldsymbol{A}\boldsymbol{x}$ of a given signal rereads $\boldsymbol{g} = \sum_p \boldsymbol{A}^p\boldsymbol{x}^p$ where vector $\boldsymbol{x} = \{\boldsymbol{x}^0,\dots,\boldsymbol{x}^P\}$ gathers the p-th order amplitudes $\boldsymbol{x}^p$ for all p. When $\boldsymbol{x}$ is sparse, all vectors $\boldsymbol{x}^p$ are sparse, and the approximate signal $\boldsymbol{g}$ is the sum of piecewise polynomials of degree lower than P with a limited number of pieces.

The dictionary $\boldsymbol{A}$ is undercomplete since it is roughly of size $m \times m(P+1)$ (there are actually $(P+1)(m-P/2)$ columns since matrices $\boldsymbol{A}_p$ are not exactly square). Moreover, it is highly correlated: any column $\boldsymbol{a}_i^p$ is strongly correlated with *all* other columns $\boldsymbol{a}_j^q$ because their respective supports are the intervals $\{i,\dots,m\}$ and $\{j,\dots,m\}$, and hence overlap. The discontinuity detection problem is difficult, as most algorithms are very likely to position false discontinuities in their early iterations. For example, when approximating a signal with two discontinuities at distinct locations i and j, they start to position a first (false) discontinuity in between i and j, and forward algorithms cannot remove it.

C. Adaptation of SBR

It is important to notice that the dictionary defined above does not satisfy the URP. For instance, the difference between two discrete ramps at locations i and $i+1$ yields the discrete Heaviside function at location i: $\boldsymbol{a}_i^1 - \boldsymbol{a}_{i+1}^1 = \boldsymbol{a}_i^0$. More generally, for $p > 1$, we have

$$\boldsymbol{a}_i^p - \boldsymbol{a}_{i+1}^p \;=\; \boldsymbol{a}_i^0 + \sum_{q=1}^{p-1} \left[\begin{array}{c} p \\ q \end{array} \right] \boldsymbol{a}_{i+1}^q$$

where $\left[\begin{array}{c} p \\ q \end{array} \right]$ refers to the binomial coefficient.

As mentioned in Section II, the SBR algorithm basically requires that the dictionary satisfies the URP in order to guarantee that the Gram matrix $\boldsymbol{G}_{\mathcal{Q}} = \boldsymbol{A}_{\mathcal{Q}}^t\boldsymbol{A}_{\mathcal{Q}}$ is invertible, but this assumption can be relaxed

provided that only full rank matrices $\boldsymbol{A}_{\mathcal{Q}}$ are explored. Here, it is not obvious to formulate a necessary and sufficient condition for the full rankness of $\boldsymbol{A}_{\mathcal{Q}}$. We rather favor a simple sufficient condition which is little enough restrictive (a more restrictive condition than the condition below would forbid the detection of two discontinuities at the same location).

Proposition 3 *Let $d(i)$ denote the number of discontinuities $\boldsymbol{a}_i^p$, $p = 0, \ldots, P$ which are being activated at sample i, i.e., for which $x_i^p \neq 0$. Let us define the binary condition $Cond(i)$:*

- *if $d(i) = 0$, $Cond(i) \triangleq 1$;*
- *if $d(i) \geqslant 1$, $Cond(i) \triangleq \big(\forall j \in \{1, \ldots, d(i)-1\},\ d(i+j) = 0\big)$.*

If the active set $\mathcal{Q}$ is such that for all i, $Cond(i) = 1$, then $\boldsymbol{G}_{\mathcal{Q}}$ is invertible.

To prove Proposition 3, we first prove the following lemma.

Lemma 1 *Consider an active set $\mathcal{Q}$ satisfying the condition of Proposition 3, and let $i^- = \min\{i \,|\, d(i) > 0\}$ denote the least location of an active entry. Up to a reordering of the columns, $\boldsymbol{A}_{\mathcal{Q}}$ rereads $\boldsymbol{A}_{\mathcal{Q}} = [\boldsymbol{A}_{i^-},\ \boldsymbol{A}_{\mathcal{Q}\setminus\{i^-\}}]$. If $\boldsymbol{A}_{\mathcal{Q}\setminus\{i^-\}}$ is full rank, then $\boldsymbol{A}_{\mathcal{Q}}$ is also full rank.*

Proof: [Proof of lemma 1] Let $I = d(i^-)$ denote the number of discontinuities at location i^- and let $0 \leqslant p_1 < p_2 < \ldots < p_I$ denote their order, sorted in the ascending order.

Suppose that there exist two families of scalars $\{\mu_{i^-}^1, \ldots, \mu_{i^-}^I\}$ and $\{\mu_i^p \,|\, i \neq i^- \text{ and } i \text{ is active at order } p\}$ such that

$$\sum_{j=1}^{I} \mu_{i^-}^{p_j}\, \boldsymbol{a}_{i^-}^{p_j} + \sum_{i \neq i^-} \sum_{p} \mu_i^p\, \boldsymbol{a}_i^p \;=\; \boldsymbol{0}. \tag{29}$$

We will show that all μ-values are necessarily equal to 0.

Rewriting the first I nonzero equations in this system and because $\mathcal{Q}$ satisfies the condition of Proposition 3, we have

$$\forall k \in \{i^-, \ldots, i^- + I - 1\},\ \sum_{j=1}^{I} \mu_{i^-}^{p_j}\, (k + i^- - 1)^{p_j} = 0.$$

In other words, the polynomial $F(X) = \sum_{j=1}^{I} \mu_{i^-}^{p_j}\, X^{p_j}$ has I positive roots. It can be shown [35] (page 76) that a non-zero polynomial formed of I monomials of different degree has at most $I - 1$ positive roots. Therefore, F is the zero polynomial and all scalars $\mu_{i^-}^{p_j}$ are 0. We deduce from (29) and from the full rankness of $\boldsymbol{A}_{\mathcal{Q}\setminus\{i^-\}}$ that $\mu_i^p = 0$ for all (i, p).

We have shown that the column vectors of $\boldsymbol{A}_{\mathcal{Q}}$ are linearly independent, *i.e.,* that $\boldsymbol{A}_{\mathcal{Q}}$ is full rank. ■

Proof: [Proof of Proposition 3] The proof of Proposition 3 directly results from a recursive application of lemma 1 when including all active locations i, sorted in the decreasing order, into the empty set. ■

Roughly speaking, Proposition 3 states that we allow to activate several discontinuities at the same location i, but then, the next samples $i+1,\ldots,i+d(i)-1=0$ must not host any discontinuity. This condition ensures that there are at most $d(i)$ discontinuities in the interval $\{i,\ldots,i+d(i)-1\}$ of length $d(i)$. The adaptation of SBR consists in trying the insertion or removal of an index into the current active set only if the above condition is true, and ignoring the other trials.

D. Numerical simulations

We first consider the case where $P=1$, leading to the joint detection of zero and first order discontinuities, *i.e.,* the piecewise affine approximation problem. We simulate noiseless data $\boldsymbol{y}^\star=\boldsymbol{A}\boldsymbol{x}^\star$ of size $m=1000$ and with $\|\boldsymbol{x}^\star\|_0=18$ discontinuities (see Fig. 4 (a)). The dictionary is of size 1000×1999.

We use the result of Proposition 2 to compute the value $\lambda=\lambda_{\max}$ below which the SBR output is not the empty set, and we run:

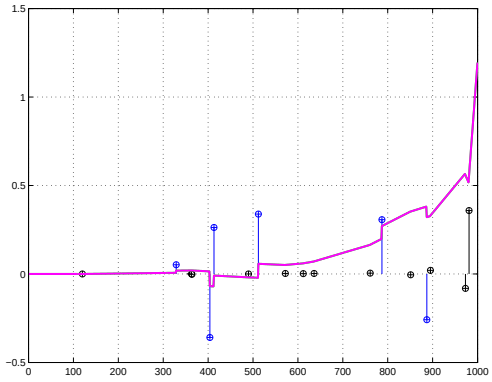
- SBR with $\lambda_k=\lambda_{\max}\,10^{(1-k)/2}$ for $k=1,\ldots,K_{\max}$, with $K_{\max}=20$. These executions provide a sequence of solutions at different sparsity levels;
- for comparison purpose, we run OMP and OLS until the iteration $k=27$ and we store all the OMP and OLS iterates.

The SBR reconstruction shown in Fig. 4 (a) corresponds to the least λ-value. The reconstructed signal totally coincides with the noiseless data although the recovery is not exact (19 discontinuities have been found among which two false discontinuities). The "ℓ_2-ℓ_0" curves represented on Fig. 4 (b) express the least-square residual $\|\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}\|^2$ versus the cardinality of the result $\|\boldsymbol{x}\|_0$, for each algorithm. This figure shows that for a given level of sparsity, SBR yields the best recovery.

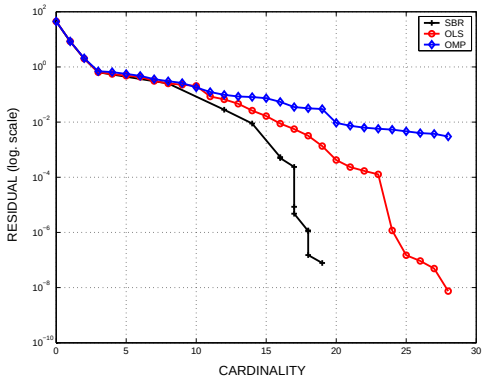
We did the same experiment with noisy data $\boldsymbol{y}=\boldsymbol{A}\boldsymbol{x}^\star+\boldsymbol{n}$, setting the SNR to 35 dB (see Figs. 4 (c,d)). Here again, the "ℓ_2-ℓ_0" curve corresponding to SBR lays below the OMP and OLS curves. For most sparsity levels, SBR outperforms the other algorithms. Note that for more noisy data (*e.g.,* SNR = 15 dB), the SBR and OLS curves coincide, and still lay below the OMP curve.

E. Real data processing

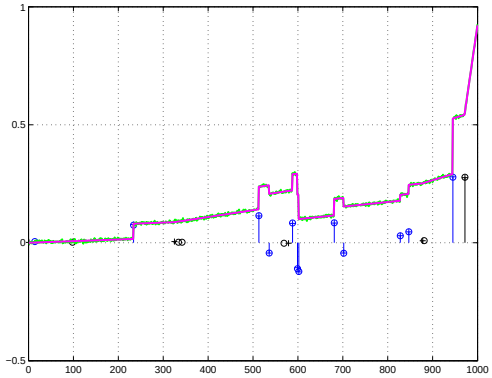
We process a set of experimental data, which are force curves measured in Atomic Force Microscopy (AFM). A force curve measures the interatomic forces exerting between a probe associated to a cantilever and a nano-object. This signal $z\mapsto y(z)$ shows the force evolution as a function of the probe-sample



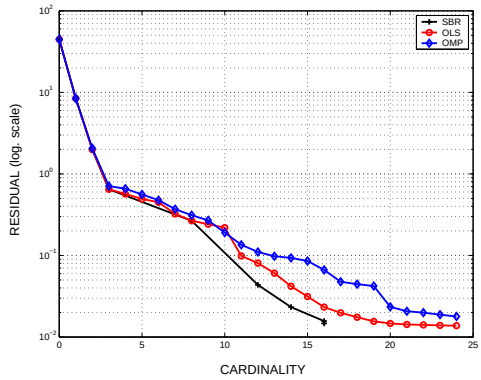
(a) Noiseless data and SBR reconstruction



(b) "ℓ_2-ℓ_0" curve



(c) Noisy data (SNR= 35 dB) and SBR reconstruction

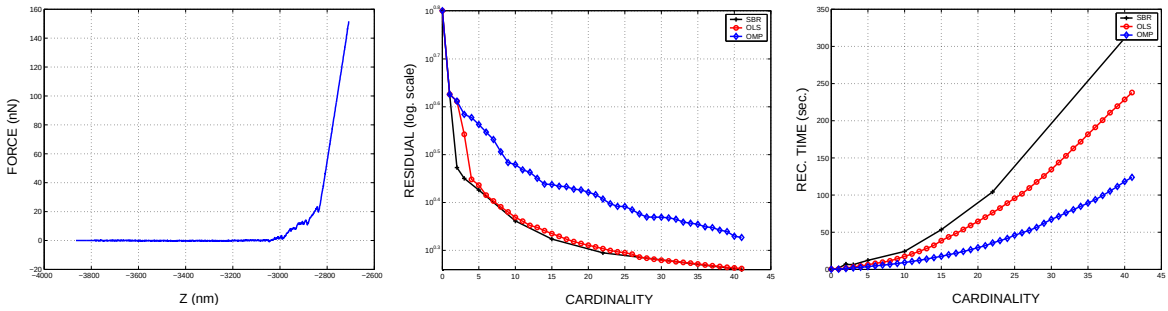


(d) "ℓ_2-ℓ_0" curve

Fig. 4. Joint detection of discontinuities at orders 0 and 1. The dictionary is of size 1000×1999 and the data signal $\boldsymbol{y}$ includes 18 jumps at orders 0 and 1 ($\|\boldsymbol{x}^\star\|_0 = 18$). The true and estimated positions of the jumps are labeled o and +. The blue and black colors indicate zero and first order discontinuities, respectively. The green and pink curves represent the data signal $\boldsymbol{y}$ and its approximation $\boldsymbol{A}\boldsymbol{x}$ for the least λ-value. (a) Signal approximation from noiseless data. The green and pink curves are superimposed. (b) Curves showing the least-square residual as a function of the cardinality for SBR, OLS, and OMP. (c,d) Similar results on noisy data (SNR = 35 dB).

distance z, expressed in nanometers. The research of discontinuities in a force curve is a critical task because the location of the discontinuities and their amplitude provide a precise characterization of the nano-object and its physico-chemical properties (topography, energy of adhesion, *etc.*) [36].

The data displayed on Fig. 5 (a) are related to a bacterial cell *Shewanella putrefaciens* laying in aqueous solution, in interaction with the tip of the AFM probe [37]. The recording of a force curve consists of two steps. Firstly, the tip lays far away from the sample. It is moved towards the sample



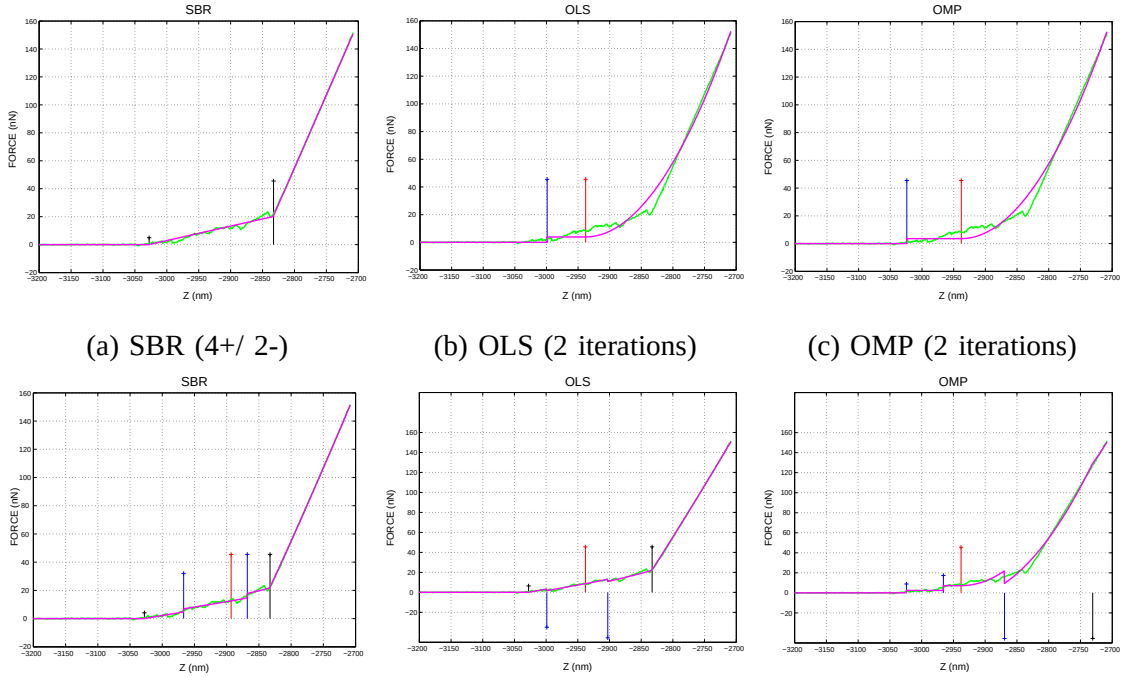
(a) Real data (b) "ℓ_2-ℓ_0" curve (c) Time *vs*. cardinality

Fig. 5. Experimental AFM data processing: joint detection of discontinuities at orders 0, 1, and 2 (problem of size 2167×6498). (a) Experimental data showing the force evolution as a function of the probe-sample distance z. (b) Curves showing the least-square residual as a function of the cardinality for the outputs of SBR, OLS and OMP. (c) Curves showing the time of reconstruction as a function of the cardinality for the three algorithms.

until the contact is reached and the surface of the bacterial cell is deformed (approach curve). Secondly, the tip is retracted from the sample. During the retraction, the occurrence of an hysteresis between the approach and retraction curves is due to the viscoelastic properties of the sample. When the tip continues to retract, several jumps are likely to occur in the force curve as the tip loses contact with the cell.

The experimental curve shown on Fig. 5 (a) is a retraction curve composed of $m = 2167$ force measurements. We can distinguish three regions of interest on this curve, from the right to the left. The linear region on the right part characterizes the contact between the probe and the sample. It describes the mechanical interactions of the cantilever and/or the sample. The contact is maintained until $z \approx -2840$ nm. The interactions occurring in the interval $z \in [-3050, -2840]$ nm are adhesion forces during the retraction of the tip. In the flat part on the left, no interaction occurs as the cantilever loses contact with the sample.

We search for the discontinuities of orders 0, 1 and 2. Similarly to what was done with the simulated data, we run SBR for $K_{\max} = 15$ λ-values and we run OLS and OMP until the iteration $k = 41$. We plot for each algorithm, the "ℓ_2-ℓ_0" curve representing the least-square residual $\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|^2$ versus the cardinality $\|\boldsymbol{x}\|_0$, and a curve showing the time of reconstruction versus the cardinality (see Fig. 5 (b,c)). These figures show that the performance of SBR is at least equal and sometimes better than that of OLS. Both algorithms yield results that are far more accurate than OMP, except for very sparse reconstructions. The price to pay for these accurate approximations is an increase of the computation time. However, notice that the recorded computation time always remains below 350 seconds in the case of SBR (in a Matlab implementation that takes advantage of the block Toeplitz structure of the dictionary: see Section IV-D).

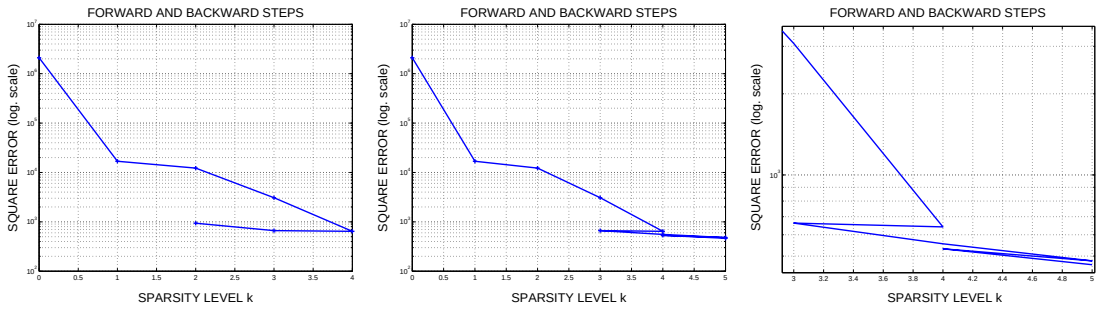


(a) SBR (4+/ 2-) (b) OLS (2 iterations) (c) OMP (2 iterations)

(d) SBR (7+/ 2-) (e) OLS (5 iterations) (f) OMP (5 iterations)

Fig. 6. Experimental AFM data processing: joint detection of discontinuities at orders 0, 1 and 2. The blue, black and red colors indicate zero, first and second order discontinuities, respectively. The estimated jumps $\boldsymbol{x}$ are labeled with a +. The green and pink curves represent the data signal $\boldsymbol{y}$ and its approximation $\boldsymbol{A}\boldsymbol{x}$. (a) SBR output of cardinality 2: 4 insertions and 2 removals have been done. (b,c) OLS and OMP output after 2 iterations. (d,e,f) Same simulation with a lower λ-value. The SBR output is of cardinality 5 (7 insertions and 2 removals) and we stop OLS and OMP after 5 iterations.

Fig. 6 shows the approximations yielded by the three algorithms for supports of cardinality 2 and 5, respectively. For the supports of cardinality 2, SBR actually runs during 6 iterations (4 insertions and 2 removals are performed) and the approximation is very accurate compared to the OMP and OLS results obtained after 2 iterations (which are identical). For the supports of cardinality 5, OLS now performs better than OMP and the solution obtained with SBR still yields a residual which is lower than the OLS and OMP residuals. In order to better understand the forward (insertions) and backward moves (removals) occurring during the SBR iterations, we plot on Fig. 7 a curve showing for each SBR iterate, the corresponding least-square residual $\|\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}\|^2$ versus its cardinality. Because SBR is a descent algorithm, the penalized cost $\mathcal{J}(\boldsymbol{x};\lambda)$ keeps decreasing but when a removal occurs, $\|\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}\|^2$ increases. On theses curves, insertion and removals correspond to south-east and north-west moves, respectively. Notice that for small λ-values, removals occur more often in the last iterations.



(a) SBR (4+/ 2-) (b) SBR (7+/2-) (c) Zoom in of (b)

Fig. 7. Display of the SBR iterates corresponding to both reconstructions of Fig. 6. The curves represent $\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|^2$ as a function of $\|\boldsymbol{x}\|_0$ for each iterate $\boldsymbol{x}$. (a) 4 insertions and 2 removals are done. (b,c) 7 insertions and 2 removals are done.

F. Discussion

In the comparisons above, we chose to compare SBR with OMP and OLS. We did not consider simpler algorithms like MP which are well suited to solve simpler problems, in which the columns of the dictionary are almost orthogonal, with speed (real-time) constraints. Because SBR involves more complex operations (matrix inversions), we chose to compare it with OMP and OLS because they also require to solve at least one least-square minimization problem per iteration, and their target is to provide results which are more accurate than the MP approximations in the case of difficult problems.

Up to our knowledge, the only minimization algorithm dedicated to the ℓ_0-penalized cost function $\mathcal{J}(\boldsymbol{x};\lambda) = \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|^2 + \lambda\|\boldsymbol{x}\|_0$ is the IHT algorithm proposed by Blumensath and Davies [14]. It relies on gradient based iterations of the form $\boldsymbol{x}' = \boldsymbol{x} + \boldsymbol{A}^t(\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x})$, followed by the threshold of all the non-zero components x_i such that $|x_i| \leqslant \lambda^{0.5}$ and their replacement with 0. On both deconvolution and discontinuity detection problems, we observed that this version of IHT is less accurate than the standard version of IHT, related to the ℓ_0-constrained problem. In the constrained version, the k components $|x_i|$ having the largest amplitudes are kept, and the others are being thresholded. Generally speaking, we observed that the IHT algorithm is competitive when the correlation between any pair of dictionary columns is limited, but for highly correlated dictionaries, a very large number of iterations ($O(m^2)$) are needed in order that IHT reaches convergence. SBR seems to be better suited to such difficult problems. It is less sensitive to the initial solution and "skips" some local minimizers whose cost is very high. We here recall that according to Proposition 1, each SBR iterate is almost surely a local minimizer of the cost function $\mathcal{J}(\boldsymbol{x};\lambda)$.

In order to compare our approach with the forward-backward algorithm of [19], we also programmed

an OMP-like adaptation of SBR in which only one least-square problem is solved at each iteration, instead of n. This adaptation consists in replacing the selection rule (13) in the following way. When an insertion $\mathcal{Q} \cup \{i\}$ is tried, all the active components x_j are kept constant and x_i is set to the minimizer of $\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}_{\mathcal{Q}} - x_i\boldsymbol{a}_i\|^2$. This leads to an approximation of $\mathcal{K}_{\mathcal{Q}\bullet i}(\lambda)$ without solving any least-square problem. Similarly, the removal of the active index i consists in setting x_i to 0 and leaving the other components x_j unchanged. In brief, this adapted version is an algorithm aimed at the minimization of $\mathcal{J}(\boldsymbol{x};\lambda)$ at a cost which is comparable to that of OMP. In all our trials, SBR yields a more accurate result than the adapted version except in very simple cases (limited correlation between the columns $\boldsymbol{a}_i$) in which SBR and the adapted versions yield the same result. The performance of the adapted version fluctuates below or above those of OMP, but are almost always far less accurate than the OLS and SBR approximations.

VII. Conclusion

We have evaluated the SBR algorithm on two problems in which the dictionary columns are highly correlated. SBR provides solutions which are at least as accurate as the OLS solutions, and sometimes more accurate, with a cost of the same order of magnitude. For such difficult problems, the MP and OMP algorithms provide poor approximations in comparison with OLS and SBR within a lower computation time.

For small λ-values, we believe that performing removals is the price to pay if one expects a better quality approximation in comparison with OLS. Zhang argued that the low number of removals occurring in the early iterations is a strong limitation of any descent algorithm dedicated to the minimization of the ℓ_0-penalized least-square cost function (see the discussion section in [19]). We rather believe that in the early iterations of SBR, the main features need to be found, thus justifying to process mainly insertions. More removals occur when a fine quality approximation is wanted, *i.e.*, for low λ-values. Nevertheless, it would be interesting to compare our approach with an algorithm like FoBa [19] which imposes removals even in its early iterations. This rule also provides a framework for proving exact recovery results for problems satisfying the Restricted Isometry Property (RIP). We will investigate whether these proofs are extendable to SBR.

In the proposed approach, the main difficulty relies in the choice of the λ-value. If a specific sparsity level k or approximation residual is desired, one needs to resort to a trial and error procedure in which a number of λ-values are tried until the desired approximation level is found. In [38], we proposed a continuation version in which a series of SBR solutions are successively estimated with a decreasing

level of sparsity λ, and the λ-values are recursively computed. The first λ-value is set to $\lambda_0 = +\infty$, and at a given value λ_i, the initial solution (input of SBR) is set to the SBR output at $\lambda = \lambda_{i-1}$. This continuation version provides promising results and will be the subject of a future extended contribution.

REFERENCES

[1] B. K. Natarajan, "Sparse approximate solutions to linear systems", *SIAM J. Comput.*, vol. 24, no. 2, pp. 227–234, Apr. 1995.

[2] G. Davis, S. Mallat, and M. Avellaneda, "Adaptive greedy approximations", *Constructive Approximation*, vol. 13, no. 1, pp. 57–98, Mar. 1997.

[3] M. Nikolova, "Local strong homogeneity of a regularized estimator", *SIAM J. Appl. Mathematics*, vol. 61, no. 2, pp. 633–658, 2000.

[4] J. Fan and R. Li, "Variable selection via nonconcave penalized likelihood and its oracle properties", *J. Acoust. Society America*, vol. 96, no. 456, pp. 1348–1360, Dec. 2001.

[5] G. H. Mohimani, M. Babaie-Zadeh, and C. Jutten, "A fast approach for overcomplete sparse decomposition based on smoothed ℓ^0 norm", *IEEE Trans. Signal Processing*, vol. 57, no. 1, pp. 289–301, Jan. 2009.

[6] B. D. Rao, K. Engan, S. F. Cotter, J. Palmer, and K. Kreutz-Delgado, "Subset selection in noise based on diversity measure minimization", *IEEE Trans. Signal Processing*, vol. 51, no. 3, pp. 760–770, Mar. 2003.

[7] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit", *SIAM J. Sci. Comput.*, vol. 20, no. 1, pp. 33–61, 1998.

[8] D. L. Donoho and Y. Tsaig, "Fast solution of l_1-norm minimization problems when the solution may be sparse", *IEEE Trans. Inf. Theory*, vol. 54, no. 11, pp. 4789–4812, Nov. 2008.

[9] 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.

[10] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani, "Least angle regression", *Annals Statist.*, vol. 32, no. 2, pp. 407–451, 2004.

[11] J.-J. Fuchs, "Recovery of exact sparse representations in the presence of bounded noise", *IEEE Trans. Inf. Theory*, vol. 51, no. 10, pp. 3601–3608, Oct. 2005.

[12] N. Meinshausen, "Relaxed Lasso", *Computational Statistics and Data Analysis*, vol. 52, no. 1, pp. 374–393, Sept. 2007.

[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] T. Blumensath and M. E. Davies, "Iterative thresholding for sparse approximations", *The Journal of Fourier Analysis and Applications*, vol. 14, no. 5, pp. 629–654, Dec. 2008.

[15] S. Mallat and Z. Zhang, "Matching pursuits with time-frequency dictionaries", *IEEE Trans. Signal Processing*, vol. 41, no. 12, pp. 3397–3415, Dec. 1993.

[16] Y. C. Pati, R. Rezaiifar, and P. S. Krishnaprasad, "Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition", in *Proceedings of 27th Asilomar Conference on Signals, Systems and Computers*, Nov. 1993, vol. 1, pp. 40–44.

[17] C. Couvreur and Y. Bresler, "On the optimallity of the backward greedy algorithm for the subset selection problem", *SIAM J. Matrix Anal. Appl.*, vol. 21, no. 3, pp. 797–808, Feb. 2000.

[18] D. Haugland, *A Bidirectional Greedy Heuristic for the Subspace Selection Problem*, vol. 4638 of *Lecture Notes in Computer Science*, pp. 162–176, Springer Verlag, Berlin, Germany, Engineering stochastic local search algorithms. Designing, implementing and analyzing effective heuristics edition, 2007.

[19] T. Zhang, "Adaptive forward-backward greedy algorithm for learning sparse representations", Tech. Rep., Rutgers Statistics Department, Apr. 2008.

[20] 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.

[21] T. Blumensath and M. E. Davies, "On the difference between orthogonal matching pursuit and orthogonal least squares", Tech. Rep., University of Edinburgh, Mar. 2007.

[22] J. J. Kormylo and J. M. Mendel, "Maximum-likelihood detection and estimation of Bernoulli-Gaussian processes", *IEEE Trans. Inf. Theory*, vol. 28, pp. 482–488, 1982.

[23] J. M. Mendel, *Optimal Seismic Deconvolution*, Academic Press, New York, NY, 1983.

[24] Y. Goussard, G. Demoment, and J. Idier, "A new algorithm for iterative deconvolution of sparse spike trains", in *Proc. IEEE ICASSP*, Albuquerque, NM, Apr. 1990, pp. 1547–1550.

[25] F. Champagnat, Y. Goussard, and J. Idier, "Unsupervised deconvolution of sparse spike trains using stochastic approximation", *IEEE Trans. Signal Processing*, vol. 44, no. 12, pp. 2988–2998, Dec. 1996.

[26] Q. Cheng, R. Chen, and T.-H. Li, "Simultaneous wavelet estimation and deconvolution of reflection seismic signals", *IEEE Trans. Geosci. Remote Sensing*, vol. 34, pp. 377–384, Mar. 1996.

[27] I. F. Gorodnitsky and B. D. Rao, "Sparse signal reconstruction from limited data using FOCUSS: A re-weighted minimum norm algorithm", *IEEE Trans. Signal Processing*, vol. 45, no. 3, pp. 600–616, Mar. 1997.

[28] S. Chen and J. Wigger, "Fast orthogonal least squares algorithm for efficient subset model selection", *IEEE Trans. Signal Processing*, vol. 43, no. 7, pp. 1713–1715, July 1995.

[29] S. J. Reeves, "An efficient implementation of the backward greedy algorithm for sparse signal reconstruction", *IEEE Signal Processing Letters*, vol. 6, no. 10, pp. 266–268, Oct. 1999.

[30] R. A. Horn and C. R. Johnson, *Matrix analysis*, Cambridge University Press, 1985.

[31] 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.

[32] D. Ge, J. Idier, and E. Le Carpentier, "Enhanced sampling schemes for MCMC based blind Bernoulli-Gaussian deconvolution", Tech. Rep., Institut de Recherche en Communication et Cybernétique de Nantes, Sept. 2009.

[33] P. E. Gill, G. H. Golub, W. Murray, and M. A. Saunders, "Methods for modifying matrix factorizations", *Mathematics of Computation*, vol. 28, no. 126, pp. 505–535, Apr. 1974.

[34] M. Vetterli, P. Marziliano, and T. Blu, "Sampling signals with finite rate of innovation", *IEEE Trans. Signal Processing*, vol. 50, no. 6, pp. 1417–1428, June 2002.

[35] F. R. Gantmakher and M. G. Krein, *Oscillation matrices and kernels and small vibrations of mechanical systems*, AMS Chelsea Publishing, Providence, RI, revised edition, 2002.

[36] H.-J. Butt, B. Cappella, and M. Kappl, "Force measurements with the atomic force microscope: Technique, interpretation and applications", *Surface Science Reports*, vol. 59, no. 1–6, pp. 1–152, Oct. 2005.

[37] F. Gaboriaud, B. S. Parcha, M. L. Gee, J. A. Holden, and R. A. Strugnell, "Spatially resolved force spectroscopy of bacterial surfaces using force-volume imaging", *Colloids and Surfaces B: Biointerfaces*, vol. 62, no. 2, pp. 206–213, Apr. 2008.

[38] 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.