Bayesian Pursuit Algorithms

Cedric Herzet, Angélique Drémeau

Bayesian Pursuit Algorithms

Cédric Herzet[(1)] and Angélique Drémeau[(2)]

[(1)] INRIA-Rennes, Centre Bretagne Atlantique, Rennes, France, cedric.herzet@inira.fr
[(2)] Institut Télécom, Télécom ParisTech, CNRS-LTCI, Paris, France, angelique.dremeau@telecom-paristech.fr

Abstract

This paper addresses the sparse representation (SR) problem within a general Bayesian framework. We show that the Lagrangian formulation of the standard SR problem, *i.e.*, $\mathbf{x}^\star = \arg\min_{\mathbf{x}}\{\|\mathbf{y} - \mathbf{D}\mathbf{x}\|_2^2 + \lambda\|\mathbf{x}\|_0\}$, can be regarded as a limit case of a general maximum a posteriori (MAP) problem involving Bernoulli-Gaussian variables. We then propose different tractable implementations of this MAP problem that we refer to as "Bayesian pursuit algorithms". The Bayesian algorithms are shown to have strong connections with several well-known pursuit algorithms of the literature (*e.g.*, MP, OMP, StOMP, CoSaMP, SP) and generalize them in several respects. In particular, *i)* they naturally allow for atom *deselection*; *ii)* they can include any prior information about the probability of occurrence of each atom within the selection process; *iii)* they can encompass the estimation of unkown model parameters into their recursions.

I. Introduction

Sparse representations (SR) aim at describing a signal as the combination of a small number of *atoms*, namely elementary signals, chosen from a given dictionary. More precisely, let $\mathbf{y} \in \mathbb{R}^N$ be an observed signal and $\mathbf{D} \in \mathbb{R}^{N\times M}$ a dictionary of atoms. Then, one standard formulation of the sparse representation problem writes

$$\mathbf{x}^\star = \arg\min_{\mathbf{x}} \|\mathbf{y} - \mathbf{D}\mathbf{x}\|_2^2 + \lambda\|\mathbf{x}\|_0, \tag{1}$$

where $\|\cdot\|_0$ denotes the l_0 pseudo-norm, which counts the number of non-zero elements in $\mathbf{x}$, and $\lambda > 0$ is a parameter specifying the trade-off between sparsity and distortion.

The resolution of sparse representation problems has been shown to be relevant in many practical situations. A few examples include statistical regression [1], digital communications [2], image processing [3], [4], interpolation/extrapolation [5], signal deconvolution [6], [7], Tomo PIV [8], etc. Another important application is the compressive-sampling paradigm, recently introduced in [9].

Finding the exact solution of (1) is an NP-hard problem [5], *i.e.*, it generally requires a combinatorial search over the entire solution space. For problems of moderate-to-high dimensionality, combinatorial

approaches are intractable and one has therefore to resort to heuristic procedures. In the current literature, three main families of algorithms can roughly be distinguished: the algorithms based on a problem relaxation, the pursuit algorithms, and the Bayesian algorithms. The work presented in this paper lies at the intersection of the Bayesian and pursuit families. Hence, in order to properly replace our work in the rich literature pertaining to SR algorithms, we briefly review hereafter some of the algorithms belonging to each family (with an emphasis on the Bayesian and pursuit families).

The *SR algorithms based on a problem relaxation* approximate the non-smooth and non-convex ℓ_0-norm by functions easier to handle. The resulting problem can then be solved by means of standard optimization techniques. Well-known instances of algorithms based on such an approach are Basis Pursuit (BP) [10] and FOCUSS [11] which approximate the ℓ_0-norm by the ℓ_1- and ℓ_p- ($p < 1$) norms, respectively.

The family of *pursuit algorithms* encompasses all the procedures looking for a solution of the sparse representation problem by making a succession of greedy decisions on the support *i.e.*, by iteratively selecting or deselecting one (or sometimes a few) atom(s) from a "local" perspective. Within the family of pursuit algorithms, one can distinguish between several approaches according to the way the algorithms update the support of the sparse representation: the *forward*, *backward* and *forward/backward* algorithms. The *forward* algorithms gradually increase the support by sequentially *adding* new atoms. Algorithms belonging to this familly include matching pursuit (MP) [12], orthogonal matching pursuit (OMP) [13], stagewise OMP (StOMP) [14], orthogonal least square (OLS) [15] or gradient pursuit (GP) [16]. *Backward* algorithms use the opposite strategy: they start from a support containing all the atoms of the dictionary and reduce it by sequentially removing "irrelevant" atoms, see [17]. Finally, *forward/backward* procedures make iteratively a new decision on the support of the sparse vector by *either* adding and/or removing atoms from the current support. Within this family, let us mention the "stepwise regression" algorithm by Efroymson [18] and its variations in [19]–[21], the iterative hard thresholding (IHT) [22], the hard thresholding pursuit (HTP) [23], the compressive sampling matching pursuit (CoSaMP) [24] and subspace pursuit (SP) [25] algorithms. A thorough description of pursuit algorithms is provided in section IV.

The *Bayesian algorithms* express the SR problem as the solution of a Bayesian inference problem and apply statistical tools to solve it. One key ingredient of the Bayesian algorithms is the choice of a proper prior on the sought sparse vector. A popular approach consists in modeling $\mathbf{x}$ as a continuous random variable whose distribution has a sharp peak to zero and heavy tails (*e.g.*, Laplace, t-Student or Jeyffrey's distributions). Such a strategy has been exploited in the following contributions [26]–[31]. Another approach, recently gaining in popularity, is based on a prior made up of the combination of Bernoulli and Gaussian distributions, see *e.g.*, [6], [7], [32]–[39]. Different (similar but distinct) variants of

Bernoulli-Gaussian (BG) models exist. A first approach, as considered in [32]–[36], consists in assuming that the elements of $\mathbf{x}$ are independently drawn from a Gaussian distribution whose variance is controlled by a Bernoulli variable. Another model on $\mathbf{x}$, based on BG variables, is as follows: the elements of the sparse vector are defined as the multiplication of Gaussian and Bernoulli variables. This leads to a sparse vector made up of zero and non-zero elements according to the realizations of their Bernoulli variables. This model has been exploited in the following contributions [6], [7], [37], [38] and will be considered in the present paper.

The contribution of this paper is twofold. First, we emphasize a possible connection between the standard problem (1) and a Bayesian inference problem. In particular we show that, under some mild conditions, the standard SR problem (1) can be regarded as a limit case of a maximum a posteriori (MAP) problem involving Bernoulli-Gaussian (BG) variables. Second, we exploit this connection to revisit and extend several standard pursuit algorithms within this Bayesian framework. The proposed algorithms, referred to as *"Bayesian pursuit algorithms"* hereafter, extend standard procedures in several aspects: *i)* they can exploit prior information about the atom occurrence and/or the amplitude of active coefficients; *ii)* the process of atom *deselection* (and therefore the design of forward/backward algorithms) finds a natural implementation within the proposed Bayesian framework ; *iii)* the estimation of model parameters (noise variance, etc) can be nicely included within the considered Bayesian framework.

The rest of the paper is organized as follows. In section III, we present a BG probabilistic framework modeling sparse processes and establish a connection between the standard problem and a maximum a posteriori (MAP) problem involving this model. In section IV, we briefly review the main concepts behind standard pursuit procedures. Section V is devoted to the derivation of the proposed algorithms. Finally, in section VI we provide extensive simulation results comparing, according to various criteria, the performance of the proposed SR procedures with the one of state-of-the-art algorithms.

II. NOTATIONS

The notational conventions adopted in this paper are as follows. The ith element of vector $\mathbf{a}$ is denoted a_i; $\langle \mathbf{a}, \mathbf{b} \rangle \triangleq \mathbf{a}^T\mathbf{b}$ defines the scalar product between vectors $\mathbf{a}$ and $\mathbf{b}$; $\|\mathbf{a}\|_2 \triangleq \langle \mathbf{a}, \mathbf{a} \rangle^{1/2}$ is the ℓ_2-norm (Euclidian norm) of $\mathbf{a}$; $\|\mathbf{a}\|_0$ denotes the ℓ_0 pseudo-norm of $\mathbf{a}$ and corresponds to the number of non-zero elements in $\mathbf{a}$. The Moore-Penrose pseudo-inverse of matrix $\mathbf{A}$ is denoted by $\mathbf{A}^\dagger$ and we use the notation $\mathbf{I}_N$ for the $N \times N$-identity matrix. The minimum of a function $f(\mathbf{a})$ is denoted by $\min_{\mathbf{a}} f(\mathbf{a})$ and the *set* of values at which this minimum is achieved by $\arg\min_{\mathbf{a}} f(\mathbf{a})$. With a slight abuse of notation, we will often use $\mathbf{a}^\star = \arg\min_{\mathbf{a}} f(\mathbf{a})$ to specify that $\mathbf{a}^\star$ belongs to the set of solutions, *i.e.*, $\mathbf{a}^\star \in \arg\min_{\mathbf{a}} f(\mathbf{a})$.

III. A BAYESIAN FORMULATION OF THE STANDARD SR PROBLEM

In this section, we present the probabilistic model that will be considered throughout this paper and state a result relating the standard formulation of the SR problem (1) to a MAP inference problem.

Let $\mathbf{D} \in \mathbb{R}^{N \times M}$ be a dictionary whose columns are normalized to 1. Let moreover $\mathbf{s} \in \{0,1\}^M$ be a vector defining the *support* of the sparse representation, *i.e.*, the subset of columns of $\mathbf{D}$ used to generate $\mathbf{y}$. We adopt the following convention: if $s_i = 1$ (resp. $s_i = 0$), the ith column of $\mathbf{D}$ is (resp. is not) used to form $\mathbf{y}$. Denoting by $\mathbf{d}_i$ the ith column of $\mathbf{D}$, we then consider the following observation model:

$$\mathbf{y} = \sum_{i=1}^{M} s_i \, x_i \, \mathbf{d}_i + \mathbf{w}, \tag{2}$$

where $\mathbf{w}$ is a zero-mean white Gaussian noise with variance σ_w^2. Therefore,

$$p(\mathbf{y}|\mathbf{x},\mathbf{s}) = \mathcal{N}(\mathbf{D_s}\mathbf{x_s}, \sigma_w^2 \mathbf{I}_N), \tag{3}$$

where $\mathbf{D_s}$ (resp. $\mathbf{x_s}$) is a matrix (resp. vector) made up of the $\mathbf{d}_i$'s (resp. x_i's) such that $s_i = 1$; $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ denotes a multivariate Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$. We suppose that $\mathbf{x}$ and $\mathbf{s}$ obey the following probabilistic model:

$$p(\mathbf{x}) = \prod_{i=1}^{M} p(x_i), \qquad\qquad p(\mathbf{s}) = \prod_{i=1}^{M} p(s_i), \tag{4}$$

where

$$p(x_i) = \mathcal{N}(0, \sigma_x^2), \qquad p(s_i) = \mathrm{Ber}(p_i), \tag{5}$$

and $\mathrm{Ber}(p_i)$ denotes a Bernoulli distribution of parameter p_i.

We emphasize hereafter a connection between the standard problem (1) and the Bernoulli-Gaussian model (3)-(5). In particular, we show in the following result that (1) can be regarded as a limit case of a joint MAP estimation problem involving the BG model (3)-(5):

Theorem 1: *Consider the following MAP estimation problem:*

$$(\hat{\mathbf{x}}, \hat{\mathbf{s}}) = \underset{(\mathbf{x},\mathbf{s})}{\arg\max} \ \log p(\mathbf{y}, \mathbf{x}, \mathbf{s}), \tag{6}$$

where $p(\mathbf{y}, \mathbf{x}, \mathbf{s}) = p(\mathbf{y}|\mathbf{x}, \mathbf{s})\, p(\mathbf{x})\, p(\mathbf{s})$ *is defined by the Bernoulli-Gaussian model* (3)-(5).
If $\|\mathbf{y}\|_2 < \infty$ *and*

$$\begin{aligned} &\sigma_x^2 \to \infty, \\ &p_i = p \quad \forall\ i,\ p \in [0,1], \\ &\lambda = 2\sigma_w^2 \log\left(\frac{1-p}{p}\right), \end{aligned} \tag{7}$$

the BG MAP problem (6) *and the standard SR problem* (1) *lead to the same set of solutions.* □

A proof of this result can be found in the Appendix. The result established in Theorem 1 recasts the standard sparse representation problem (1) into a more general Bayesian framework. In particular, the Bayesian formulation allows for more degrees of freedom than (1). For example, any prior information about the amplitude of the non-zero coefficients (σ_x^2) or the atom occurrence (p_i's) can explicitly be taken into account. We will see moreover in section V-D, that the proposed framework easily extends to the case of structured sparsity, in which some constraints are imposed between the elements of the support.

From condition (7), we see that the equivalence between the standard and Bayesian formulations occurs for particular values of the model parameters. First, one must have that $p_i = p\ \forall i$. This implies that the probability of occurrence of all the atoms are a priori equal. Moreover, $\sigma_x^2 = \infty$ defines a non-informative prior $p(\mathbf{x})$ on the amplitude of the non-zero coefficients. These requirements are in accordance with the standard SR formulation (1) which neither favors some atoms of the dictionary nor imposes constraints on the amplitude of the non-zero coefficients.

Let us mention that a result similar to Theorem 1 was already presented in our conference paper [37] and the parallel work by Soussen *et al.* [7]. The equivalence proposed in this paper is however more general since, unlike these previous results, it does not require any condition of the type

$$\|\mathbf{D}_{\mathbf{s}}^{\dagger}\mathbf{y}\|_0 = \|\mathbf{s}\|_0 \quad \forall\, \mathbf{s} \in \{0,1\}^M \tag{8}$$

to hold. In particular, Theorem 1 extends the equivalence between (1) and (6) to the important case of noise-free data. Indeed, assume that the observed vector is generated as follows:

$$\mathbf{y} = \mathbf{D}_{\tilde{\mathbf{s}}}\tilde{\mathbf{x}}_{\tilde{\mathbf{s}}}, \tag{9}$$

where $\tilde{\mathbf{s}} \in \{0,1\}^M$, $\|\tilde{\mathbf{s}}\|_0 < N$, and $\tilde{\mathbf{x}} \in \mathbb{R}^M$ are realizations of some arbitrary random variables. Then, any vector $\mathbf{s}$ such that

$$\begin{cases} s_i = 1 & \text{if } \tilde{s}_i = 1 \\ \|\mathbf{s}\|_0 \leq N \end{cases} \tag{10}$$

violates the equality (8) and the results in [7], [37] do therefore not apply.

To conclude this section, let us note that the BG MAP formulation (6) does not offer any advantage in terms of complexity with respect to (1), *i.e.*, it is NP-hard. The practical computation of solutions of (6) requires therefore to resort to heuristic (but practical) algorithms. In the rest of this paper, we will propose several greedy procedures looking for a solution of (6). Due to the equivalence between (1) and (6) emphasized in Theorem 1, the proposed greedy procedures will share some similarities with standard pursuit algorithms. More precisely, several well-known pursuit algorithms will be shown to be particular cases of the algorithms proposed in section V-C.

IV. Standard Pursuit Algorithms

Before going through the derivation of greedy procedures based on the Bayesian problem (6), we first review the main concepts underlying standard pursuit algorithms. *In a nutshell*, standard pursuit algorithms can be understood as iterative procedures looking for a solution of the following problem

$$(\hat{\mathbf{x}}, \hat{\mathbf{s}}) = \underset{\mathbf{x}, \|\mathbf{s}\|_0 \leq K}{\arg\min} \|\mathbf{r}(\mathbf{x}, \mathbf{s})\|_2^2, \tag{11}$$

where $K \geq 0$ is a prespecified number of non-zero coefficients and

$$\mathbf{r}(\mathbf{x}, \mathbf{s}) \triangleq \mathbf{y} - \mathbf{D}_{\mathbf{s}}\mathbf{x}_{\mathbf{s}} \tag{12}$$

is the residual error for given $\mathbf{x}$ and $\mathbf{s}$.

To reach this goal, pursuit algorithms generate a sequence of estimates $\{\hat{\mathbf{x}}^{(n)}, \hat{\mathbf{s}}^{(n)}\}_{n=0}^{\infty}$ by iteratively updating its current decision on the support and the coefficient amplitudes of the sparse representation. The algorithms available in the literature basically differ in the way they implement these two operations. In the rest of this section, we dwell upon the main strategies existing in the literature and recast some well-known pursuit algorithms in this respect. Note that the formalisation of the standard pursuit algorithms pursued hereafter might be slightly unconventional for the experienced reader but will be useful for the sake of comparison with the proposed algorithms in the next section.

A. Coefficient update

The coefficient update aims at finding the "best" estimate $\hat{\mathbf{x}}^{(n)}$ while taking the current estimate of the SR support $\hat{\mathbf{s}}^{(n)}$ into account. In most of the contributions we are aware of, the update of $\hat{\mathbf{x}}^{(n)}$ is steered by the goal of decreasing the norm of the residual error. The most common approach consists in

evaluating $\hat{\mathbf{x}}^{(n)}$ as follows:

$$\hat{\mathbf{x}}^{(n)} = \arg\min_{\mathbf{x}} \|\mathbf{r}(\mathbf{x}, \hat{\mathbf{s}}^{(n)})\|_2^2, \tag{13}$$

where $\mathbf{r}(\mathbf{x}, \hat{\mathbf{s}}^{(n)})$ is defined in (12). A solution of (13) writes

$$\begin{aligned} \hat{\mathbf{x}}^{(n)}_{\mathbf{s}^{(n)}} &= \mathbf{D}^{\dagger}_{\hat{\mathbf{s}}^{(n)}}\mathbf{y}, \\ \hat{x}_i^{(n)} &= 0 \qquad\qquad \text{if } \hat{s}_i^{(n)} = 0. \end{aligned} \tag{14}$$

In some scenarios, this coefficient update can be computationally too-demanding since it requires the evaluation of a pseudo-inverse in (14). Hence, some low-complexity alternatives have been proposed to search for a solution of (13). For example, matching pursuit [12] considers (to some extent) a block-coordinate descent implementation of (13), *i.e.*, at each iteration the norm of the residual error is optimized with respect to *one single* component of $\mathbf{x}$. Suppose coefficient x_j is optimized at iteration n, the latter strategy is therefore tantamount to adding the following constraints to (13): $\hat{x}_i^{(n)} = \hat{x}_i^{(n-1)}\ \forall\, i \neq j$. This leads to the simple update rules:

$$\hat{x}_i^{(n)} = \begin{cases} \hat{s}_i^{(n)}(\hat{x}_i^{(n-1)} + \langle \mathbf{r}(\hat{\mathbf{x}}^{(n-1)}, \hat{\mathbf{s}}^{(n-1)}), \mathbf{d}_i \rangle) & \text{if } i = j, \\ \hat{x}_i^{(n-1)} & \text{otherwise.} \end{cases} \tag{15}$$

Other procedures based on gradient optimization methods have been proposed in [16]. These methods can also be implemented in the coefficient update step but will not be further considered in the sequel.

B. Support Update

The support-update operation consists in making a guess about the columns of the dictionary (or *atoms*) involved in the generation of $\mathbf{y}$. *In a nutshell*, many support-update strategies pursued by standard pursuit algorithms can be formalized as follows:

$$\hat{\mathbf{s}}^{(n)} = \arg\min_{\mathbf{s} \in \mathcal{S}^{(n-1)}} \|\mathbf{r}(\mathbf{x}(\mathbf{s}), \mathbf{s})\|_2^2, \tag{16}$$

where $\mathcal{S}^{(n-1)}$ denotes a set of candidates for the new support update and $\mathbf{x}(\mathbf{s})$ is a coefficient vector obtained by applying (14) or (15), for a given support $\mathbf{s}$. Note that $\mathbf{x}(\mathbf{s})$ in (16) can be evaluated differently from $\hat{\mathbf{x}}^{(n)}$, and can therefore differ from the latter.

The algorithms available in the literature basically differ in two ways: *i)* the strategy used to compute $\mathbf{x}(\mathbf{s})$; *ii)* the definition of the set of support candidates $\mathcal{S}^{(n)}$. In the latter respect, we can distinguish three main families of algorithms: the *forward, backward* and *forward/backward* algorithms.

The *forward algorithms* update the support by adding (but never removing) new atoms. The most common approach consists in adding *one* new atom to the support at each iteration. The set of support candidates then reads

$$\mathcal{S}^{(n)} = \left\{ \mathbf{s} \,|\, \|\mathbf{s}\|_0 \leq \|\hat{\mathbf{s}}^{(n)}\|_0 + 1, s_i = 1 \text{ if } s_i^{(n)} = 1 \right\}, \tag{17}$$

The first condition in the right-hand side of (17) implies that the size of the support can *at most*[1] increase by one element at each iteration; the second condition ensures that once an atom has been selected, it can never be removed from the support. In the sequel, we will use the short-hand notation $\mathcal{S}_+^{(n)}$ to refer to the set of support candidates defined in (17).

MP [12], OMP [13] and OLS [15] implement forward strategies based on (16)-(17) but differ in the evaluation of $\mathbf{x}(\mathbf{s})$. While MP and OMP compute $\mathbf{x}(\mathbf{s})$ from (15), OLS considers the more complex operation (14). Other forward algorithms have been proposed, allowing for the selection of *several* atoms at each iteration. This is for example the case of stagewise OMP (StOMP) [14], a modified version of OMP.

The *backward algorithms* implement a support update which lies at the exact opposite of forward algorithms: they start from a support estimate containing *all* the atoms of the dictionary and sequentially *remove* the less appropriate. The set of support candidates considered by backward algorithms then typically writes:

$$\mathcal{S}^{(n)} = \left\{ \mathbf{s} \,|\, \|\mathbf{s}\|_0 = \|\hat{\mathbf{s}}^{(n)}\|_0 - 1, s_i = 0 \text{ if } s_i^{(n)} = 0 \right\}. \tag{18}$$

The first condition implies that the support has to be reduced by one element at each iteration; the second condition ensures that once an atom has been deselected, it can never return to the support. In the sequel, we will use the short-hand notation $\mathcal{S}_-^{(n)}$ to refer to the set (18).

Backward algorithms have been extensively studied for undercomplete dictionaries in the statistical regression community [40]. They have been revisited more recently by Couvreur *et al.* in [17]. In these algorithms, $\mathbf{x}(\mathbf{s})$ is computed from (14) and one atom is removed at each iteration. Note that backward algorithms are of poor practical interest for overcomplete dictionaries since $\min_{\mathbf{x}} \|\mathbf{r}(\mathbf{x}, \mathbf{s})\|_2^2 = 0 \ \forall \mathbf{s} \in \mathcal{S}^{(n)}$ as soon as $\|\hat{\mathbf{s}}^{(n)}\|_0 > N$. Hence, backward algorithms are therefore not able to make any relevant decision in such a case. Hence, we will not further consider this family of procedures hereafter.

Common to forward and backward algorithms is the fact that *error correction* is not possible: once a

[1]The decision to keep the current support unchanged can be made.

wrong (resp. good) atom has been added to (resp. removed from) the support, it can never (explicitly) be removed (resp. reincluded). *Forward/backward* algorithms provide a solution to this problem since they allow for both atom *selection* and *deselection*.

The main issue of forward/backward algorithms lies in the fact that the standard criterion based on the norm of the residual error (16) does not allow for atom deselection. Indeed, we have

$$\min_{\mathbf{s}\in\mathcal{S}_+^{(n)}} \|\mathbf{r}(\mathbf{x}(\mathbf{s}),\mathbf{s})\|_2^2 \leq \min_{\mathbf{s}\in\mathcal{S}_-^{(n)}} \|\mathbf{r}(\mathbf{x}(\mathbf{s}),\mathbf{s})\|_2^2, \tag{19}$$

where $\mathcal{S}_+^{(n)}$ and $\mathcal{S}_-^{(n)}$ are the sets of support candidates respectively defined in (17) and (18). Therefore, a support update of the form (16), with $\mathcal{S}^{(n)} = \mathcal{S}_-^{(n)} \cup \mathcal{S}_+^{(n)}$, always leads to a forward step, *i.e.*, an increase of the size of the support. Alternative strategies have thus to be considered to allow for atom deselection.

A popular approach in the statistical regression community considers the use of hypothesis testing techniques. The first procedure of this type we are aware of is the so-called *"stepwise regression"* proposed by Efroymson [18]. In substance, the algorithm operates as follows: after each atom selection, a backward step is performed if the following condition is satisfied:

$$\frac{\|\mathbf{r}(\mathbf{x}(\hat{\mathbf{s}}^-),\hat{\mathbf{s}}^-)\|_2^2}{\|\mathbf{r}(\mathbf{x}(\hat{\mathbf{s}}^{(n)}),\hat{\mathbf{s}}^{(n)})\|_2^2} \leq 1+\alpha^{(n)}, \tag{20}$$

where

$$\hat{\mathbf{s}}^- = \arg\min_{\mathbf{s}\in\mathcal{S}_-^{(n)}} \|\mathbf{r}(\mathbf{x}(\mathbf{s}),\mathbf{s})\|_2^2, \tag{21}$$

and $\alpha^{(n)} \geq 0$; otherwise a new forward step is realized. In a nutshell, (20) and (21) can be interpreted as follows: a backward step is performed if the removal of an atom does not lead to a "too large" increase of the norm of the current residual error. Efroymson suggests to approximate the (intractable) distribution of $\|\mathbf{r}(\mathbf{x}(\hat{\mathbf{s}}^-),\hat{\mathbf{s}}^-)\|_2^2/\|\mathbf{r}(\mathbf{x}(\hat{\mathbf{s}}^{(n)}),\hat{\mathbf{s}}^{(n)})\|_2^2$ by a Fisher distribution and tunes the value of $\alpha^{(n)}$ on the basis of hypothesis testing criteria. The Fisher distribution has later been shown [41], [42] to be a quite bad approximation of the actual distribution of $\|\mathbf{r}(\mathbf{x}(\hat{\mathbf{s}}^-),\hat{\mathbf{s}}^-)\|_2^2/\|\mathbf{r}(\mathbf{x}(\hat{\mathbf{s}}^{(n)}),\hat{\mathbf{s}}^{(n)})\|_2^2$ and other alternatives, such as the Mallow's distribution, have been proposed [19]. We refer the reader to [40] for a more detailed discussion on this topic. Note that these contributions deal exclusively with the undercomplete setting ($N > M$).

More recently, contributions in the same spirit have considered the overcomplete problem, see [20], [21]. Other forward/backward procedures, more conceptually-involved, have also appeared in the literature. In particular, CoSaMP [24] and the related subspace pursuit (SP) algorithm [25], the iterative hard

thresholding (IHT) [22] and the hard thresholding pursuit (HTP) [23] algorithms are among the most effective procedures of the state of the art.

Note that most of the forward-backward algorithms mentioned above (with the exception of [19] which only applies to undercomplete problems) are not expressed as optimization algorithms looking for the minimum of a goal function but rather as clever heuristic schemes. Hence, inspired by the work by Kormylo and Mendel [6], Soussen and Idier recently proposed a forward-backward algorithm (see [7]) looking iteratively for a solution of (1). Formally, their algorithm allows for atom deselection if

$$\|\mathbf{r}(\mathbf{x}(\hat{\mathbf{s}}^{-}),\hat{\mathbf{s}}^{-})\|_2^2 - \|\mathbf{r}(\mathbf{x}(\hat{\mathbf{s}}^{(n)}),\hat{\mathbf{s}}^{(n)})\|_2^2 < \lambda, \tag{22}$$

where $\mathbf{x}(\mathbf{s})$ is evaluated as in (14).

C. Some Well-known Pursuit Algorithms

For the sake of comparison with the procedures proposed in section V, we now briefly remind the reader of the main expressions of some well-known pursuit algorithms of the literature. In particular, we dwell upon matching pursuit (MP), orthogonal matching pursuit (OMP), stagewise OMP (StOMP), and subspace pursuit (SP). We refer the reader to the original papers for more information.

1) (Orthogonal) Matching Pursuit: MP and OMP are forward pursuit algorithms implementing the same support update but differing in the coefficient update step. Both algorithms select one index to update at each iteration as

$$j = \arg\max_i \ \langle \mathbf{r}(\hat{\mathbf{x}}^{(n-1)},\hat{\mathbf{s}}^{(n-1)}),\mathbf{d}_i\rangle^2 \tag{23}$$

and consider the following support update:

$$\hat{s}_i^{(n)} = \begin{cases} 1 & \text{if } i=j, \\ \hat{s}_i^{(n-1)} & \text{otherwise.} \end{cases} \tag{24}$$

This support update is obtained by solving (16) with $\mathcal{S}^{(n)}$ defined in (17) and $\mathbf{x}(\mathbf{s})$ computed from (15). It allows therefore for the inclusion of (at most) one single atom in the support at each iteration. Note that $\langle \mathbf{r}(\hat{\mathbf{x}}^{(n-1)},\hat{\mathbf{s}}^{(n-1)}),\mathbf{d}_i\rangle^2$ corresponds to the decrease of the residual error when the ith atom is added to the support and its amplitude is evaluated via (15).

Regarding coefficient update, MP implements (15) whereas OMP considers the update defined in (14). MP and OMP can be regarded as descent algorithms (the descent function being the norm of the residual error $\|\mathbf{r}(\mathbf{x},\mathbf{s})\|_2$) and are therefore ensured to converge.

2) Stagewise OMP: StOMP is a modified version of OMP which allows for the selection of several new atoms at each iteration. The coefficient update is therefore performed via (14). The choice of the atoms added to the support estimate $\hat{\mathbf{s}}^{(n)}$ deviates from the general rule (16) but is inspired from (O)MP's (23)-(24). In particular, the support decision is made by a threshold decision on the projections of the residual:

$$\hat{s}_i^{(n)} = \begin{cases} 1 & \text{if } \langle \mathbf{r}(\hat{\mathbf{x}}^{(n-1)}, \hat{\mathbf{s}}^{(n-1)}), \mathbf{d}_i \rangle^2 > T^{(n)}, \\ \hat{s}_i^{(n-1)} & \text{otherwise,} \end{cases} \tag{25}$$

where $T^{(n)}$ is a threshold depending on the iteration number. In [14], the authors suggested two different approaches to set the value of the threshold $T^{(n)}$, although no explicit expression was provided. StOMP is thus a forward algorithm since the size of the support can only increase throughout the iterations.

3) Subspace Pursuit/CoSaMP: CoSaMP and SP are two slightly different versions of the same pursuit procedure. These algorithms can be regarded as forward/backward in the sense that they successively select and deselect several atoms to maintain a sparse decomposition containing a prespecified number of atoms, say K. SP/CoSaMP proceeds in two steps. For the sake of clarity, we thus divide each iteration into two sub-iterations denoted by (n) and $(n + \frac{1}{2})$. First a forward step is performed where P atoms are added to the previous support:

$$\hat{\mathbf{s}}^{(n)} = \underset{\mathbf{s} \in \mathcal{S}_P}{\arg\max} \left\{ \sum_i s_i \langle \mathbf{r}(\hat{\mathbf{x}}^{(n-\frac{1}{2})}, \hat{\mathbf{s}}^{(n-\frac{1}{2})}), \mathbf{d}_i \rangle^2 \right\} \tag{26}$$

where

$$\mathcal{S}_P \triangleq \{\mathbf{s} | \, \|\mathbf{s} - \hat{\mathbf{s}}^{(n-\frac{1}{2})}\|_0 = P, s_i = 1 \text{ if } s_i^{(n-\frac{1}{2})} = 1\}. \tag{27}$$

We remind the reader that $\langle \mathbf{r}(\hat{\mathbf{x}}^{(n-\frac{1}{2})}, \hat{\mathbf{s}}^{(n-\frac{1}{2})}), \mathbf{d}_i \rangle^2$ corresponds to the decrease of the norm of the residual error if atom i is added to the support and coefficient update (15) is performed. The atoms selected are therefore those leading to the maximum decrease of the residual norm (when added individually). Moreover, the atoms selected at the previous iteration have to remain selected. The first sub-iteration ends with the estimation of $\hat{\mathbf{x}}^{(n)}$ according to (14).

In a second step, SP/CoSaMP removes some atoms from the support by applying the following rule:

$$\hat{\mathbf{s}}^{(n+\frac{1}{2})} = \underset{\mathbf{s} \in \mathcal{S}_K}{\arg\max} \left\{ \sum_i s_i \, (\hat{x}_i^{(n)})^2 \right\} \tag{28}$$

where

$$\mathcal{S}_K \triangleq \{\mathbf{s}|\, \|\mathbf{s}\|_0 = K, s_i = 0 \text{ if } s_0^{(n)} = 0\}. \tag{29}$$

Here, $(\hat{x}_i^{(n)})^2$ corresponds to the increase of the residual norm when atom i is removed from the support. The atoms deselected by SP/CoSaMP are therefore those leading to the smallest increase of the norm of the residual (when removed individually). Moreover, the atoms which were not selected at the previous iteration have to remain unselected. Then, SP evaluates $\hat{\mathbf{x}}^{(n+\frac{1}{2})}$ from (14) whereas CoSaMP considers the following update:

$$\hat{x}_i^{(n+1/2)} = \begin{cases} \hat{x}_i^{(n)} & \text{if } \hat{s}_i^{(n+1/2)} = 1, \\ 0 & \text{otherwise.} \end{cases} \tag{30}$$

V. Bayesian Pursuit Algorithms

In this section, we introduce novel pursuit algorithms arising from the Bayesian framework described in section III. The proposed algorithms search for a solution of the MAP problem (6) by sequentially updating the current support and coefficient estimates, $\hat{\mathbf{s}}^{(n)}$ and $\hat{\mathbf{x}}^{(n)}$. Due to the similarity between (6) and (11) (both problems involve the optimization of a function with respect to a continuous-valued vector $\mathbf{x}$ and a discrete-valued vector $\mathbf{s}$), update strategies based on the same spirit as those described for standard pursuit algorithms (see section IV) can be considered here.

Because of their connections with the standard pursuit algorithms described in section IV and the Bayesian framework from which they arise, we will refer to the procedures introduced hereafter as *Bayesian pursuit algorithms*. As we will see, the proposed algorithms can somehow be regarded as extensions of the standard pursuit algorithms since they encompass the later as particular cases and offer the additional desirable features:

- They allow for the inclusion of some a priori information in the atom selection/deselection process, *i.e.*, the prior information about the occurrence of each atom, p_i, can be explicitly taken into account into the sparse decomposition. Moreover, as we will emphasize in section V-D, the methodology described in this paper easily extends to the paradigm of "structured sparsity" where additional constraints on the support of the sparse representation have to be satisfied.
- The implementation of forward/backward algorithms finds a natural solution within the considered Bayesian framework. In particular, there is no need for the inclusion of additional heuristics allowing for atom deselection: the removal of some atoms from the support straightforwardly follows from the considered Bayesian framework (see section V-C).

- The estimation of the model parameters can be naturally included into the considered Bayesian framework (see section V-E). In particular, we will emphasize that the inclusion of the noise variance estimation throughout the pursuit iterations plays a crucial role in the algorithm performance.

The rest of this section is organized as follows. In the two next subsections, we revisit the coefficient and support update strategies described in section III within the considered Bayesian framework. More formally, we study the update equations

$$\hat{\mathbf{x}}^{(n)} = \arg\max_{\mathbf{x}} \log p(\mathbf{y}, \mathbf{x}, \hat{\mathbf{s}}^{(n)}), \tag{31}$$

$$\hat{\mathbf{s}}^{(n)} = \arg\max_{\mathbf{s}\in\mathcal{S}^{(n-1)}} \log p(\mathbf{y}, \mathbf{x}(\mathbf{s}), \mathbf{s}), \tag{32}$$

and highlight their differences with the standard approaches (13), (16). Then, in section V-C four instances of Bayesian pursuit algorithms based on these strategies (or variations thereof) are presented. In subsection V-D, we discuss how the proposed procedures can be extended to any type of prior $p(\mathbf{s})$, establishing a bridge between the Bayesian pursuit algorithms and the "structured sparsity" paradigm. Finally, in subsection V-E we show how the estimation of the noise variance can be encompassed within the proposed algorithms and show its impact on the Bayesian pursuit algorithms.

A. Coefficient Update within the Bayesian Framework

Let us consider the coefficient update defined in (31). This update can be regarded as the Bayesian counterpart of (13). Indeed, whereas (13) leads to the coefficient update maximizing the decrease of $\|\mathbf{r}(\mathbf{x},\mathbf{s})\|_2^2$, (31) maximizes the increase of $\log p(\mathbf{y}, \mathbf{x}, \hat{\mathbf{s}}^{(n)})$. Taking model (3)-(5) into account, the solution of (31) writes as

$$\begin{aligned} \hat{\mathbf{x}}^{(n)}_{\hat{\mathbf{s}}^{(n)}} &= \left(\mathbf{D}^T_{\hat{\mathbf{s}}^{(n)}}\mathbf{D}_{\hat{\mathbf{s}}^{(n)}} + \tfrac{\sigma_w^2}{\sigma_x^2}\mathbf{I}_{\|\hat{\mathbf{s}}^{(n)}\|_0}\right)^{-1}\mathbf{D}^T_{\hat{\mathbf{s}}^{(n)}}\mathbf{y}, \\ \hat{x}_i^{(n)} &= 0 \quad \text{if } \hat{s}_i^{(n)} = 0. \end{aligned} \tag{33}$$

On the other hand, the counterpart of (15) within the considered Bayesian framework corresponds to maximizing $\log p(\mathbf{y}, \mathbf{x}, \hat{\mathbf{s}}^{(n)})$ with respect to one single element x_j while keeping other elements fixed, *i.e.*, $x_i = \hat{x}_i^{(n-1)}\ \forall\, i \neq j$. Solving (31) subject to this constraint leads to:

$$\hat{x}_j^{(n)} = \frac{\hat{s}_j^{(n)}\,\sigma_x^2}{\sigma_x^2 + \sigma_w^2}\left(\hat{x}_j^{(n-1)} + \langle \mathbf{r}(\hat{\mathbf{x}}^{(n-1)}, \hat{\mathbf{s}}^{(n-1)}), \mathbf{d}_j\rangle\right). \tag{34}$$

We note that (33) and (34) are the exact respective counterparts of (14) and (15) in a Bayesian context. The main difference between the standard and the Bayesian updates lies in the exploitation of some

coefficient prior information by the latter: unlike standard strategies, (33) and (34) update the coefficient amplitude by taking into account the variance σ_x^2. Interestingly, (33) and (34) reduce to the expressions obtained in the standard setup (resp. (14) and (15)) when one considers a non-informative prior on $\mathbf{x}$, *i.e.*, $\sigma_x^2 \to \infty$.

B. Support Update within the Bayesian Framework

We now emphasize that the considered MAP problem (6) naturally allows for atom deselection. Let us focus on the following support update:

$$\hat{\mathbf{s}}^{(n)} = \underset{\mathbf{s}\in\mathcal{S}^{(n-1)}}{\arg\max}\ \log p(\mathbf{y}, \mathbf{x}(\mathbf{s}), \mathbf{s}), \tag{35}$$

where $\mathbf{x}(\mathbf{s})$ is a trial coefficient update[2] defined for example by (33) or (34), and $\mathcal{S}^{(n-1)}$ denotes a set of support candidates. Update (35) is the exact counterpart of (16) in the considered Bayesian context.

A crucial difference of considering $\log p(\mathbf{y}, \mathbf{x}, \mathbf{s})$ as a goal function (by opposition to $\|\mathbf{r}(\mathbf{x}, \mathbf{s})\|_2^2$ in the standard pursuit procedures) is that it naturally allows for the implementation of forward/backward algorithms. In particular, the following inequality does *not* necessarily occur:

$$\max_{\mathbf{s}\in\mathcal{S}_-^{(n)}}\ \log p(\mathbf{y}, \mathbf{x}(\mathbf{s}), \mathbf{s}) \leq \max_{\mathbf{s}\in\mathcal{S}_+^{(n)}}\ \log p(\mathbf{y}, \mathbf{x}(\mathbf{s}), \mathbf{s}) \tag{36}$$

where $\mathcal{S}_+^{(n)}$ and $\mathcal{S}_-^{(n)}$ are the sets of support candidates respectively defined in (17) and (18). In other words, decreasing the size of the support of the sparse representation may possibly lead to an increase of $\log p(\mathbf{y}, \mathbf{x}(\mathbf{s}), \mathbf{s})$ and therefore be beneficial. In particular, an atom deselection is possible as long as

$$\log p(\mathbf{y}, \mathbf{x}(\mathbf{s}^-), \mathbf{s}^-) \geq \log p(\mathbf{y}, \mathbf{x}(\hat{\mathbf{s}}^{(n)}), \hat{\mathbf{s}}^{(n)}), \tag{37}$$

where

$$\mathbf{s}^- = \underset{\mathbf{s}\in\mathcal{S}_-^{(n)}}{\arg\max}\ \log p(\mathbf{y}, \mathbf{x}(\mathbf{s}), \mathbf{s}). \tag{38}$$

Inequality (37) has to be compared to (19) in the standard setup, in which an atom deselection never leads to a decrease of the residual error norm.

For the sake of comparison with the standard procedures described in section IV, let us rewrite (37) in the particular case where $\sigma_x^2 \to \infty$. After some simple mathematical derivations, we obtain:

[2]Note that the same remark as in section IV-B applies here regarding $\mathbf{x}(\mathbf{s})$: the strategy considered to evaluate $\mathbf{x}(\mathbf{s})$ may differ from the one used to compute the new coefficient estimate $\hat{\mathbf{x}}^{(n)}$.

$$\|\mathbf{r}(\mathbf{x}(\mathbf{s}^-),\mathbf{s}^-)\|^2 - \|\mathbf{r}(\mathbf{x}(\hat{\mathbf{s}}^{(n)}),\hat{\mathbf{s}}^{(n)})\|^2 \leq \lambda_i^{(n)} \tag{39}$$

where

$$\lambda_i^{(n)} = \sigma_w^2 \sum_i (\hat{s}_i^{(n)} - s_i^-) \log\left(\frac{1-p_i}{p_i}\right). \tag{40}$$

Expression (39) is very similar to (22), which is a direct consequence of the equivalence theorem proved in section III. Two crucial differences however arise: *i)* the value of the threshold in the right-hand side of (39) is not fixed but depends on the sequence $\hat{\mathbf{s}}^{(n)}$ and $\mathbf{s}^-$; *ii)* the threshold is directly related to the parameters of the considered probabilistic model. This will for example have an impact when estimating the noise variance (see section V-E).

C. Pursuit Algorithms Revisited within a Bayesian Framework

In this section, we revisit the greedy procedures presented in section IV-C within the proposed Bayesian framework and emphasize their differences with respect to their standard counterparts. The resulting algorithms belong to the family of forward-backward procedures and can be expressed as optimization algorithms looking for a (local) maximum of (6). To the best of our knowledge, the four algorithms presented hereafter are new and have never been proposed before in this form. The forward-backward procedure the most related in spririt with the proposed methodologies is the SBR algorithm [7] which performs the optimization of $\log p(\mathbf{y},\mathbf{x},\mathbf{s})$ under the hypotheses of Theorem 1 and with an optimization strategy different form these presented hereafter.

1) Bayesian Matching Pursuit (BMP): As mentioned in section IV, MP iteratively updates the SR support by *adding* the atom leading to the maximum decrease of the residual norm when its amplitude is optimized via (15). A similar approach can be followed within the Bayesian framework considered here: we define the BMP algorithm so that the couple (x_j, s_j) updated at each iteration "locally" maximizes the increase of $\log p(\mathbf{y},\mathbf{x},\mathbf{s})$.

More formally, BMP is defined as a particular instance of a block-coordinate ascent algorithm applied to (6). Let us first define

$$\hat{\mathbf{x}}_i^{(n)} = [\hat{x}_1^{(n)} \ \ldots \ \hat{x}_{i-1}^{(n)} \ x_i \ \hat{x}_{i+1}^{(n)} \ \ldots \ \hat{x}_M^{(n)}]^T, \tag{41}$$

$$\hat{\mathbf{s}}_i^{(n)} = [\hat{s}_1^{(n)} \ \ldots \ \hat{s}_{i-1}^{(n)} \ s_i \ \hat{s}_{i+1}^{(n)} \ \ldots \ \hat{s}_M^{(n)}]^T. \tag{42}$$

Hence, $\hat{\mathbf{x}}_i^{(n)}$ (resp. $\hat{\mathbf{s}}_i^{(n)}$) denotes a vector in which the ith component, x_i (resp. s_i), is free to vary but all other components are fixed to the value of the current estimate $\hat{\mathbf{x}}^{(n)}$ (resp. $\hat{\mathbf{s}}^{(n)}$).

At each iteration of BMP, the couple (x_j, s_j) leading to the maximum increase of $\log p(\mathbf{y}, \mathbf{x}, \mathbf{s})$, under the constraint

$$(x_i, s_i) = (\hat{x}_i^{(n-1)}, \hat{s}_i^{(n-1)}) \quad \forall i \neq j, \tag{43}$$

is updated. This leads to the following recursion:

$$s_i^{(n)} = \begin{cases} \tilde{s}_i^{(n)} & \text{if } i = j \\ \hat{s}_i^{(n-1)} & \text{otherwise} \end{cases}, \; x_i^{(n)} = \begin{cases} \tilde{x}_i^{(n)} & \text{if } i = j \\ \hat{x}_i^{(n-1)} & \text{otherwise} \end{cases}$$

where

$$j = \arg\max_i \{ \max_{(x_i, s_i)} \log p(\mathbf{y}, \hat{\mathbf{x}}_i^{(n-1)}, \hat{\mathbf{s}}_i^{(n-1)}) \}, \tag{44}$$

and

$$(\tilde{x}_i^{(n)}, \tilde{s}_i^{(n)}) = \arg\max_{(x_i, s_i)} \{ \log p(\mathbf{y}, \hat{\mathbf{x}}_i^{(n-1)}, \hat{\mathbf{s}}_i^{(n-1)}) \}. \tag{45}$$

The operations performed by BMP are summarized in Table I. Expressions (89)-(90) give the explicit solutions of (45). Problem (92) is an alternative, equivalent, formulation of (44) where

$$\begin{aligned} \rho_i^{(n)}(s_i) \triangleq \max_{x_i} \{ & \log p(\mathbf{y}, \hat{\mathbf{x}}_i^{(n-1)}, \hat{\mathbf{s}}_i^{(n-1)}) \} \\ & - \log p(\mathbf{y}, \mathbf{x}^{(n-1)}, \mathbf{s}^{(n-1)}). \end{aligned} \tag{46}$$

Hence, $\rho_i^{(n)}(s_i)$ represents the maximum increase (or the minimum decrease) of $\log p(\mathbf{y}, \hat{\mathbf{x}}^{(n-1)}, \hat{\mathbf{s}}^{(n-1)})$ with respect to x_i and for a given value of s_i, while keeping all other elements in $\hat{\mathbf{x}}^{(n-1)}$ and $\hat{\mathbf{s}}^{(n-1)}$ unchanged. Note that support update (44) is tantamount to solving (16) with $\mathcal{S}^{(n)} = \mathcal{S}_+^{(n)} \cup \mathcal{S}_-^{(n)}$ and $\mathbf{x}(\mathbf{s})$ computed from (34).

We see from (92) that BMP updates the element leading to the largest increase of the cost function $\log p(\mathbf{y}, \mathbf{x}, \mathbf{s})$. This observation is to parallel with MP which updates the element leading to the largest decrease of the residual error. A crucial difference between MP and BMP is the fact that BMP naturally implements the process of deselecting some of the columns of the current support. Indeed, we see from (89) that the modification of the jth element of the support arises from a threshold decision. The value of $\tilde{s}_j^{(n)}$ is based on the comparison of the residual energy in the direction of $\mathbf{d}_j$ to a threshold T_j, defined in (91). In particular, if $\tilde{s}_j^{(n)} = 0$ whereas $\hat{s}_j^{(n-1)} = 1$, the decision consists in *removing* the jth column of $\mathbf{D}$ from the support. The threshold T_j depends on the probability of occurrence of each atom, p_j :

the larger p_j the smaller T_j and the more likely is the corresponding column to be selected in the sparse representation. Regarding BMP's coefficient update, we see that (90) is similar to (34). It is therefore tantamount to solving (31) subject to the constraints $x_i = \hat{x}_i^{(n-1)} \, \forall \, i \neq j$.

Let us now discuss the relation existing between MP and BMP. In section III, we emphasized the equivalence of the joint BG MAP problem (6) and the standard SR problem (1) when $\sigma_x^2 \to \infty$ and $p_i = p \; \forall \; i$. Subsequently, one can ask the question of the equivalence between MP and BMP under the same conditions. The answer is negative: these conditions are not sufficient to ensure the equivalence between the two algorithms. Indeed, letting $\sigma_x^2 \to \infty$ and $p_i = p \; \forall \; i$ in (89)-(91), one can observe that all equations but (90) and (91) remain unchanged: (90) reduces to (15) whereas (91) becomes

$$T_i \triangleq 2\sigma_w^2 \log \left(\frac{1 - p_i}{p_i} \right). \tag{47}$$

Hence, we note that the atom deselection allowed by BMP but impossible in the MP procedure is still present in the particular case $\sigma_x^2 \to \infty$ and $p_i = p \; \forall \; i$. Withdrawing this opportunity (by forcing $\tilde{s}_j^{(n)} = 1 \; \forall \; j$ in (89)), *i.e.*, only considering the addition (but never the removal) of new atoms in the support, one recovers standard MP implementation. The standard MP algorithm can therefore be regarded as a particular case of the Bayesian pursuit algorithm presented in this section.

2) Bayesian Orthogonal Matching Pursuit (BOMP) : We now consider the implementation of the Bayesian version of OMP. We define BOMP by modifying the coefficient-update step of the BMP algorithm. In particular, BOMP computes the estimate of $\mathbf{x}$, given the support $\hat{\mathbf{s}}^{(n)}$, as follows:

$$\hat{\mathbf{x}}^{(n)} = \arg \max_{\mathbf{x}} \log p(\mathbf{y}, \mathbf{x}, \hat{\mathbf{s}}^{(n)}). \tag{48}$$

The explicit solution to this problem is given in (33). The update of the support remains unchanged with respect to BMP. The operations performed by BOMP are therefore similar to those described in Table I but with the coefficient update step (94) replaced by (33).

Like BMP, BOMP also implements atom deselection. For this reason, similar to the one mentioned for the BMP/MP equivalence, BOMP does not reduce to OMP when $\sigma_x^2 \to \infty$ and $p_i = p \; \forall \; i$. BOMP can also be interpreted as a block-coordinate ascent algorithm applied to $\log p(\mathbf{y}, \mathbf{x}, \mathbf{s})$, the subsets of variables optimized sequentially being a couple (x_i, s_i), then the whole vector $\mathbf{x}$.

3) Bayesian Stagewise Orthogonal Matching Pursuit (BStOMP) : We define BStOMP as a modified version of BOMP where several entries of the support vector $\mathbf{s}$ can be changed at each iteration. We

propose the following approach:

$$\hat{s}_j^{(n)} = \begin{cases} 1 & \text{if } \rho_i^{(n)}(1) > \rho_i^{(n)}(0), \\ 0 & \text{otherwise,} \end{cases} \tag{49}$$

where $\rho_i^{(n)}(s_i)$ is defined in (46). The interpretation of (49) is thus as follows: the ith column of $\mathbf{D}$ is added to (resp. removed from) the support of the sparse representation if and only if it leads to a local increase of the goal function.

Using (46), (49) writes explicitly as

$$\hat{s}_j^{(n)} = \begin{cases} 1 & \text{if } \langle \mathbf{r}(\hat{\mathbf{x}}^{(n-1)}, \hat{\mathbf{s}}^{(n-1)}) + \hat{x}_j^{(n-1)} \mathbf{d}_j, \mathbf{d}_j \rangle^2 > T_j, \\ 0 & \text{otherwise,} \end{cases} \tag{50}$$

where T_j is defined in (91). Note that if the jth atom was not selected at iteration $n-1$, *i.e.*, $(\hat{x}_j^{(n-1)}, \hat{s}_j^{(n-1)}) = (0, 0)$, (50) becomes

$$\hat{s}_j^{(n)} = \begin{cases} 1 & \text{if } \langle \mathbf{r}(\hat{\mathbf{x}}^{(n-1)}, \hat{\mathbf{s}}^{(n-1)}), \mathbf{d}_j \rangle^2 > T_j, \\ \hat{s}_j^{(n-1)} & \text{otherwise.} \end{cases} \tag{51}$$

In such a case, the support update rules of StOMP and BStOMP are therefore similar. However, in the general case (50), BStOMP allows for the deselection of atoms.

Another crucial difference between StOMP and BStOMP is the definition of the threshold T_j. Indeed, the Bayesian framework considered in this paper naturally leads to a definition of the threshold as a function of the model parameters. Unlike the approach followed in [14], it requires therefore no additional hypothesis and/or design criterion.

The operations performed by BStOMP are therefore similar to those described in Table I but with the support update (93) (resp. coefficient update (94)) replaced by (50) (resp. (33)).

4) Bayesian Subspace Pursuit (BSP) : We finally propose a Bayesian pursuit algorithm having some flavor of CoSaMP/SP. We will refer to this algorithm as Bayesian subspace pursuit (BSP) algorithm. BSP alternates between sparse representations whose support respectively contains K and *at most* $K+P$ atoms. In the same spirit as CoSaMP/SP, the choice of the atoms added or removed from the support is based on the local increase of the goal function $\log p(\mathbf{y}, \hat{\mathbf{x}}^{(n)}, \hat{\mathbf{s}}^{(n)})$ when the support is changed by one single element. More formally, BSP is implemented as follows: in a first step, the support update is

evaluated as

$$\hat{\mathbf{s}}^{(n)} = \underset{\mathbf{s}\in\mathcal{S}_P}{\arg\max} \left\{ \sum_i s_i \, \rho_i^{(n)}(1) + (1 - s_i) \, \rho_i^{(n)}(0) \right\}, \tag{52}$$

where

$$\mathcal{S}_P = \{\mathbf{s} \,|\, \|\mathbf{s} - \hat{\mathbf{s}}^{(n-1)}\|_0 = P\}, \tag{53}$$

and the coefficient estimate $\hat{\mathbf{x}}^{(n)}$ is computed from (48). Then, in a second step, the support update is performed by solving the following problem

$$\hat{\mathbf{s}}^{(n+1)} = \underset{\mathbf{s}\in\mathcal{S}_K}{\arg\max} \left\{ \sum_i s_i \, \rho_i^{(n+1)}(1) + (1 - s_i) \, \rho_i^{(n+1)}(0) \right\}. \tag{54}$$

where

$$\mathcal{S}_K = \{\mathbf{s} \,|\, \|\mathbf{s}\|_0 = K\}, \tag{55}$$

whereas the coefficient estimate $\hat{\mathbf{x}}^{(n+1)}$ is again computed from (48).

Note that the ith term of the cost function in (52) and (54) corresponds to the maximum increase (or minimum decrease) of $\log p(\mathbf{y}, \hat{\mathbf{x}}^{(n)}, \hat{\mathbf{s}}^{(n)})$ if $\hat{s}_i^{(n)} = s_i$ and x_i is free to vary. BSP update equations (52)-(55) follow therefore the same idea as CoSaMP/SP (namely select the atoms which lead to the largest increase of the cost function) but with a noticeable difference: at each step, contrarily to CoSaMP/SP, BSP allows for atom deselection. This can easily be seen from the definition of the search spaces, $\mathcal{S}_P$ and $\mathcal{S}_K$, in (52), (54): $\mathcal{S}_P$ (resp. $\mathcal{S}_K$) defines to the set of supports differing from $\hat{\mathbf{s}}^{(n-1)}$ at P positions (resp. having a ℓ_0-norm equal to K); unlike the definitions in (27) and (29), no constraint is made on the position of the zero or the non-zero elements. The operations performed by BSP are summarized in Table II.

D. A Bridge to Structured Sparsity

Recent contributions (see *e.g.*, [43]–[45]) have emphasized the interest of considering structures between atoms selected in sparse representations, for a wide range of dictionaries and classes of signals. By clearly differentiating the support from the coefficients of the decomposition, the Bayesian framework (3)-(5) adopted in this paper is well-suited to an easy extension of the proposed algorithms to structured decompositions. In fact, the algorithms derived in section V-C remain valid for arbitrary prior $p(\mathbf{s})$ as

long as the parameters p_i's are updated through the iterations as:

$$p_i^{(n)} = p(s_1 = \hat{s}_1^{(n)}, \ldots, s_{i-1} = \hat{s}_{i-1}^{(n)}, s_i = 1, s_{i+1} = \hat{s}_{i+1}^{(n)}, \ldots, s_M = \hat{s}_M^{(n)}), \tag{56}$$

i.e., the value of the p_i's considered by the Bayesian pursuit procedures depends on the current estimate of the support $\hat{\mathbf{s}}^{(n)}$. In particular, with this modification, BMP and BOMP remain descent algorithms searching a (local) minimum of $\log p(\mathbf{y}, \mathbf{x}, \mathbf{s})$ for any choice of $p(\mathbf{s})$. Bayesian pursuits algorithm can therefore straightforwardly be extended to any type of "structured" prior $p(\mathbf{s})$. The comprehensive study of this new paradigm is however out of the scope of this contribution and will not be further considered in the rest of the paper.

E. Parameter estimation and adaptive threshold

In this section, we discuss the embedding of the estimation of the model parameters into the iterative process defined by the pursuit algorithms. We exclusively focus hereafter on the estimation of the noise variance σ_w^2 which has revealed to be crucial for the algorithm performance in our empirical experiments. Other parameters such as *e.g.*, σ_x^2, can be estimated in the same way.

A (conditional) maximum-likelihood estimate of σ_w^2 can be computed at each iteration as

$$\left(\hat{\sigma_w^2}\right)^{(n)} = \arg\max_{\sigma_w^2} \log p(\mathbf{y}, \hat{\mathbf{x}}^{(n)}, \hat{\mathbf{s}}^{(n)}). \tag{57}$$

We see that $\left(\hat{\sigma_w^2}\right)^{(n)}$ depends on the current value of $\mathbf{x}^{(n)}$ and $\mathbf{s}^{(n)}$. Such an operation can therefore be included within the pursuit recursions to refine the estimation of the noise variance through the iterations.

The solution of (57) reads

$$\left(\hat{\sigma_w^2}\right)^{(n)} = N^{-1} \|\mathbf{r}(\hat{\mathbf{x}}^{(n-1)}, \hat{\mathbf{s}}^{(n-1)})\|_2^2, \tag{58}$$

i.e., the estimate of the noise variance is proportional to the modulus of the error residual.

Including the estimation of the noise variance within the pursuit recursion has a direct impact on the decision threshold T_j appearing in the Bayesian pursuit algorithms. Indeed, plugging (58) into (91), we obtain:

$$T_j^{(n)} \triangleq 2 \frac{\|\mathbf{r}(\hat{\mathbf{x}}^{(n)}, \hat{\mathbf{s}}^{(n)})\|_2^2}{N} \log\left(\frac{1-p_j}{p_j}\right) \frac{\sigma_x^2 + N^{-1}\|\mathbf{r}(\hat{\mathbf{x}}^{(n)}, \hat{\mathbf{s}}^{(n)})\|_2^2}{\sigma_x^2}. \tag{59}$$

The threshold therefore becomes a function of the iteration number. Note that, when $\sigma_x^2 \to \infty$, the last factor in (59) tends to one. $T_j^{(n)}$ is then proportional to the residual energy; the proportionality

factor depends on the occurrence probability of each atom. In practice, $T_j^{(n)}$ has therefore the following operational meaning: during the first iterations, the residual is large (and so is $T_j^{(n)}$), and only the atoms having a large correlation with $\mathbf{y}$ are likely to be included in the support; after a few iterations, the norm of the residual error decreases and atoms weighted by smaller coefficients can enter the support.

VI. Simulation Results

In this section we illustrate the performance of the proposed algorithms by simulation results. We consider the following metrics, where $\hat{\mathbf{x}}$ and $\hat{\mathbf{s}}$ are the final estimates of $\mathbf{x}$ and $\mathbf{s}$ delivered by the sparse representation algorithms:

- the mean relative square error (MRSE) between the true and the estimated vectors of coefficients:

$$MRSE = E_{\mathbf{y},\mathbf{x}}\left[\|\hat{\mathbf{x}} - \mathbf{x}\|_2^2 / \|\mathbf{x}\|_2^2\right],$$

  where $E_{\mathbf{y},\mathbf{x}}[\cdot]$ represents the expectation operator with respect to $\mathbf{y}$ and $\mathbf{x}$.
- the probability of missed detection on the elements of the support, *i.e.*, $p(\hat{s}_i = 0 | s_i = 1)$.
- the probability of false detection on the elements of the support, *i.e.*, $p(\hat{s}_i = 1 | s_i = 0)$.

We evaluate these metrics by Monte-Carlo simulations. For each point of simulation, 5000 trials are generated. We generate the data as follows. We assume $N = 128$, $M = 256$, $\sigma_n^2 = 10^{-4}$. The elements of the dictionary are generated for each observation as realizations of a zero-mean Gaussian distribution with variance N^{-1}. The data vector $\mathbf{y}$ is generated according to model (3). Regarding the generation of the support $\mathbf{s}$ and coefficient amplitude $\mathbf{x}$, we consider different scenarios that we detail below.

We compare the performance of BMP, BOMP, BStOMP and BSP to MP, OMP, StOMP, SP, IHT, HTP. For StOMP, SP, IHT and HTP, we use the implementations available on the author's webpages (see [46]–[49]). The software implementing the proposed algorithms will be released on the website [50] at the publication of the paper.

A. The Uniform Case

We first consider the case where all the atoms have the same probability to be active, *i.e.*, $p_i = p \ \forall i$. Each point of simulation corresponds to a fixed number of non-zero coefficients, say K, and, given this number, the positions of the non-zero coefficients are drawn uniformly at random for *each* observation. The elements of $\mathbf{x}$ are generated according to two different distributions (see below).

The stopping criterion of MP and OMP is based on the norm of the residual: the recursions are stopped as soon as the norm of the residual drops below $\sqrt{N\sigma_n^2}$. StOMP is run with the "CFAR" thresholding

criterion [14]. The noise variance is set to $\sigma_n^2 = 5 \times 10^{-5}$ in BMP and BOMP, *i.e.*, slightly below its actual value. BStOMP and BSP implement the noise variance estimation described in section V-E. We set $p_i = \frac{K}{M}$, $\forall i$ in all the Bayesian pursuit algorithms.

a) Gaussian model: the non-zero elements of $\mathbf{x}$ are drawn from a zero-mean Gaussian with variance $\sigma_x^2 = 1$. The MRSE, the missed and false detection probabilities are represented in Fig. 1, 2 and 3, respectively, for different algorithms. We see that the Bayesian pursuit algorithm outperform their standard counterparts with respect to all figures of merit. In particular, considering Fig. 2 and 3, we can observe that the proposed procedures usually offer enhanced support-recovery performance, both in terms of probabilities of missed and false detection. The most important gain is brought by BStOMP in terms of false detection: StOMP, which does not allow for atom deselection, suffers from a high false detection rate; on the contrary the atom deselection process implemented by BStOMP leads to a significant decrease of the number of false detections. OMP leads to a smaller probability of false detection than BOMP in the range $K \leq 30$ but this is at the expense of a higher probability of missed detection. Finally, HTP (resp. IHT) has similar (resp. slightly worse) performance to (resp. than) SP.

The improvement of the support reconstruction brought by the Bayesian pursuit algorithms is reflected in terms of MRSE in Fig. 1. BOMP, BStOMP and BSP all lead to similar performance. BMP, which enjoys the smallest complexity, also exhibits the poorest performance. A similar behavior can be observed for the standard pursuit algorithms.

b) 0-1 model: in this scenario, the amplitude of the non-zero elements in $\mathbf{x}$ is set to 1. The performance of the considered algorithms is represented in Fig. 4, 5 and 6. We can observe that all the Bayesian pursuit algorithms but BSP lead to similar or improved performance as compared to their standard counterparts. MP and BMP have similar probabilities of missed detection but BMP considerably decreases the probability of false detection. The same behavior is observed for OMP and BOMP. BStOMP significantly improves both the probabilities of missed and false detection as compared to StOMP. HTP and IHT have probabilities of missed detection similar to BOMP and very low probabilities of false detection in the range $K \leq 20$ (the absence of point means that no errors occured during the 5000 trials). As in the Gaussian case, these improvements of the support reconstruction are reflected in terms of MRSE in Fig. 4. Finally, we notice that BSP suffers from a bad recovery performance in this particular scenario. We remind the reader that the main difference between SP and BSP algorithms is in the atom selection process: whereas SP can exclusively either add or remove atoms from the support, BSP allows for atom selection/deselection at each step of the algorithm. Although the latter approach pays off in the Gaussian scenario, it seems to be a poor strategy for the "0-1 model" as illustrated in Fig. 4-6.

B. The Non-uniform Case

We now consider the case where the atom of the dictionary have different probability to be active. As discussed in section V-D, this situation occurs for example when structured sparsity is considered or if any prior information is available about the probability of occurrence of each atom.

We assume that the p_i's are independent realizations of a beta distribution $Beta(\alpha, \beta)$ with $\alpha = 0.4$, $\beta = 0.4$. For *each* trial, the positions of the non-zero coefficients are drawn uniformly at random. This leads to a realization of $\mathbf{s}$. Knowing $\mathbf{s}$, we draw the value of p_i from their posterior distribution[3] $p(p_i|s_i)$ $\forall\, i$. The values of the non-zero coefficients in $\mathbf{x}$ are drawn from a zero-mean Gaussian with $\sigma_x^2 = 1$.

The Bayesian algorithms are assumed to have a perfect knowledge of the occurrence probabilities p_i's. They are therefore expected to exploit this additional information to improve the recovery performance. This is illustrated in Fig. 7, 8 and 9 which represent the performance of different algorithms for this particular setup. The performance has to be compared to the one of Fig. 1, 2 and 3 where all the atoms have the same probability of occurrence. On the one hand, we see that the standard pursuit algorithms exhibit the same performance since they do not exploit any information about the atom occurrence. On the other hand, we can observe that the Bayesian procedures can make a valuable use of this additional information. In particular, both BStOMP and BSP lead to probabilities of missed/false detection (roughly) equal to 10^{-2} up to $K = 70$. On the contrary, these probabilities deteriorate for BStOMP and BSP in Fig. 2-3 as soon as $K \simeq 45$ is reached. The improvement of the support recovery performance is reflected in Fig. 7 in terms of MRSE. We observe that the MRSE improves with respect to the one in Fig. 1 for all the Bayesian algorithms. Once again the gain is the most significant for BSP and BStOMP.

APPENDIX

In this appendix, we prove the equivalence between (1) and (6) under the hypotheses of Theorem 1, *i.e.*, *i)* $\|\mathbf{y}\|_2 < \infty$, *ii)* $\sigma_x^2 \to \infty$, *iii)* $p_i = p$ $\forall i$, *iv)* $\lambda \triangleq 2\sigma_w^2 \log(\frac{1-p}{p})$. We always assume hereafter, without explicitly mentioning it, that the third and fourth hypotheses are satisfied. For the sake of clarity and conciseness, we restrict the demonstration to the case where subsets of $L \leq N$ columns of $\mathbf{D}$ are linearly independent. The general case can however be derived in a similar way.

We first pose some notations and definitions. Let

$$\mathcal{X}^\star \triangleq \arg\min_{\mathbf{x}} \{\|\mathbf{y} - \mathbf{D}\mathbf{x}\|_2^2 + \lambda \|\mathbf{x}\|_0\}, \tag{60}$$

[3]It is easy to show that $p(p_i|s_i)$ is a beta distribution $Beta(\alpha', \beta')$ with $\alpha' = \alpha + s_i$ and $\beta' = \beta + 1 - s_i$

be the set of solutions of the standard sparse representation problem. Let moreover

$$g(\mathbf{s}) \triangleq \min_{\mathbf{x}}\{-2\sigma_w^2 \log p(\mathbf{y},\mathbf{x},\mathbf{s}) - 2\sigma_w^2 \log(1-p)\}, \tag{61}$$

$$\hat{\mathbf{x}}(\mathbf{s}) \triangleq \arg\min_{\mathbf{x}}\{-\log p(\mathbf{y},\mathbf{x},\mathbf{s})\}. \tag{62}$$

The goal function in (61) is equal, up to an additive and multiplicative constant, to $-\log p(\mathbf{y},\mathbf{x},\mathbf{s})$. This definition will lead to desirable simplifications in the proof. Notice that the equality in (62) is well-defined since the minimum exists and is unique as shown below. With these notations, we define the set of solutions of the BG problem (6) as

$$\hat{\mathcal{X}} \triangleq \{\mathbf{x} \in \mathbb{R}^N | \mathbf{x} = \hat{\mathbf{x}}(\mathbf{s}) \text{ with } \mathbf{s} \in \arg\min_{\mathbf{s}} g(\mathbf{s})\} \tag{63}$$

We want therefore to show that

$$\mathcal{X}^\star = \hat{\mathcal{X}}, \tag{64}$$

under the hypotheses of Theorem 1.

We first give an alternative, equivalent, definition to $\mathcal{X}^\star$. In order to do so, let us define $f(\mathbf{s})$ as

$$f(\mathbf{s}) \triangleq \min_{\mathbf{x}}\{\|\mathbf{y}-\mathbf{D}\mathbf{x}\|_2^2 + \lambda\|\mathbf{x}\|_0\} \text{ s.t. } x_i = 0 \text{ if } s_i = 0. \tag{65}$$

$f(\mathbf{s})$ is therefore the minimum of the standard SR problem (1) if the SR support is fixed. Let us moreover define $\mathbf{x}^\star(\mathbf{s})$ as

$$\mathbf{x}_{\mathbf{s}}^\star(\mathbf{s}) \triangleq \mathbf{D}_{\mathbf{s}}^\dagger \mathbf{y}, \text{ and } x_i^\star(\mathbf{s}) \triangleq 0 \text{ if } s_i = 0. \tag{66}$$

Clearly, $\mathbf{x}^\star(\mathbf{s})$ is a solution of (65), *i.e.*,

$$\mathbf{x}^\star(\mathbf{s}) \in \arg\min_{\mathbf{x}}\{\|\mathbf{y}-\mathbf{D}\mathbf{x}\|_2^2 + \lambda\|\mathbf{x}\|_0\} \text{ s.t. } x_i = 0 \text{ if } s_i = 0. \tag{67}$$

It is the unique (resp. the minimum ℓ_2-norm) solution if $\|\mathbf{s}\|_0 \leq N$ (resp. $\|\mathbf{s}\|_0 > N$). Using these notations, we can redefine $\mathcal{X}^\star$ as follows

$$\mathcal{X}^\star = \{\mathbf{x} \in \mathbb{R}^N | \mathbf{x} = \mathbf{x}^\star(\mathbf{s}) \text{ with } \mathbf{s} \in \arg\min_{\mathbf{s}} f(\mathbf{s})\}. \tag{68}$$

This definition is valid because the minimum of $f(\mathbf{s})$ is necessarily achieved for $\mathbf{s}$ such that $\|\mathbf{s}\|_0 \leq N$, in which case (66) is the unique solution of (67).

Finally, before going through the proof of the result, note that $f(\mathbf{s})$ and $g(\mathbf{s})$ can only take on a finite number of values since $\mathbf{s} \in \{0,1\}^M$. We therefore define

$$\mathcal{F} \triangleq \{f(\mathbf{s}) | \, \mathbf{s} \in \{0,1\}^M\}, \tag{69}$$

$$\mathcal{G} \triangleq \{\lim_{\sigma_x^2 \to \infty} g(\mathbf{s}) | \, \mathbf{s} \in \{0,1\}^M\}, \tag{70}$$

as the sets collecting the values of $f(\mathbf{s})$ and $\lim_{\sigma_x^2 \to \infty} g(\mathbf{s})$.

We prove Theorem 1 by first showing the following intermediate results:

$$\lim_{\sigma_x^2 \to \infty} \hat{\mathbf{x}}(\mathbf{s}) = \mathbf{x}^\star(\mathbf{s}) \quad \forall \mathbf{s}, \tag{71}$$

$$\lim_{\sigma_x^2 \to \infty} g(\mathbf{s}) \geq f(\mathbf{s}) \quad \forall \mathbf{s}, \tag{72}$$

$$\mathcal{F} \subseteq \mathcal{G} \tag{73}$$

$$\arg\min_{\mathbf{s}} \{\lim_{\sigma_x^2 \to \infty} g(\mathbf{s})\} \subseteq \arg\min_{\mathbf{s}} f(\mathbf{s}). \tag{74}$$

1) Proof of (71)*:* From standard Bayesian theory (see for example [51, Chap. 14]), it can be seen that the unique minimum of $\log p(\mathbf{y}, \mathbf{x}, \mathbf{s})$ is given by

$$\begin{aligned} &\hat{\mathbf{x}}_{\mathbf{s}}(\mathbf{s}) = \left(\mathbf{D}_{\mathbf{s}}^T \mathbf{D}_{\mathbf{s}} + \tfrac{\sigma_w^2}{\sigma_x^2} \mathbf{I}_k\right)^{-1} \mathbf{D}_{\mathbf{s}}^T \mathbf{y}, \\ &\hat{x}_i(\mathbf{s}) = 0 \qquad\qquad\qquad\qquad \text{if } s_i = 0. \end{aligned} \tag{75}$$

Taking the limit for $\sigma_x^2 \to \infty$, we obtain the equivalence between (66) and (75).

2) Proof of (72)*:* Using (71) and taking (3)-(5) into account, we have

$$\lim_{\sigma_x^2 \to \infty} g(\mathbf{s}) = \|\mathbf{y} - \mathbf{D}\mathbf{x}^\star(\mathbf{s})\|_2^2 - 2\sigma_w^2 \log p(\mathbf{s}) - 2\sigma_w^2 \log(1-p) + \sigma_w^2 \lim_{\sigma_x^2 \to \infty} \frac{\|\hat{\mathbf{x}}(\mathbf{s})\|_2^2}{\sigma_x^2}. \tag{76}$$

Since $\|\mathbf{y}\|_2 < \infty$ by hypothesis, it follows that

$$\lim_{\sigma_x^2 \to \infty} \|\hat{\mathbf{x}}(\mathbf{s})\|_2 = \|\mathbf{x}^\star(\mathbf{s})\|_2 = \|\mathbf{D}_{\mathbf{s}}^\dagger \mathbf{y}\|_2 < \infty, \tag{77}$$

since $\mathbf{D}_{\mathbf{s}}^\dagger$ is a bounded operator. Hence, the last term in (76) tends to zero. Moreover, since $p_i = p \; \forall i$ one can rewrite $p(\mathbf{s})$ as

$$\log p(\mathbf{s}) = -\|\mathbf{s}\|_0 \log\left(\frac{1-p}{p}\right) + \log(1-p). \tag{78}$$

Therefore, letting $\lambda \triangleq 2\sigma_w^2 \log(\frac{1-p}{p})$, we obtain

$$\lim_{\sigma_x^2 \to \infty} g(\mathbf{s}) = \|\mathbf{y} - \mathbf{D}\mathbf{x}^\star(\mathbf{s})\|_2^2 + \lambda \|\mathbf{s}\|_0. \tag{79}$$

Finally, realizing that

$$\|\mathbf{x}^\star(\mathbf{s})\|_0 \leq \|\mathbf{s}\|_0 \quad \forall \mathbf{s}, \tag{80}$$

we come up with (72). It is interesting to note that the equality holds in (72) if and only if $\|\mathbf{x}^\star(\mathbf{s})\|_0 = \|\mathbf{s}\|_0$, *i.e.*, when $x_i \neq 0 \ \forall s_i = 1$.

3) Proof of (73)*:* To prove this result, we show that $\forall \mathbf{s} \in \{0,1\}^M$, $\exists \tilde{\mathbf{s}} \in \{0,1\}^M$ such that $f(\mathbf{s}) = \lim_{\sigma_x^2 \to \infty} g(\tilde{\mathbf{s}})$. First notice that the value of $f(\mathbf{s})$ is only a function of $\mathbf{x}^\star(\mathbf{s})$. Now, $\forall \mathbf{s}$ one can find $\tilde{\mathbf{s}}$ such that

$$\mathbf{x}^\star(\mathbf{s}) = \mathbf{x}^\star(\tilde{\mathbf{s}}), \tag{81}$$

$$\|\mathbf{x}^\star(\tilde{\mathbf{s}})\|_0 = \|\tilde{\mathbf{s}}\|_0. \tag{82}$$

In order to see the last assertion, define $\tilde{\mathbf{s}}$ as

$$\tilde{s}_i = \begin{cases} 1 & \text{if } x_i^\star(\mathbf{s}) \neq 0 \\ 0 & \text{otherwise.} \end{cases} \tag{83}$$

Clearly, $\mathbf{s}$ and $\tilde{\mathbf{s}}$ differ as soon as $\|\mathbf{x}^\star(\mathbf{s})\|_0 \neq \|\mathbf{s}\|_0$. Considering problem (67), $\mathbf{s}$ and $\tilde{\mathbf{s}}$ introduce therefore different sets of constraints on the solution. By definition, the contraints defined by $\tilde{\mathbf{s}}$ include those defined by $\mathbf{s}$. However, the new constraints introduced by $\tilde{\mathbf{s}}$ has no effect on the solution since they correspond to $x_i^\star(\mathbf{s}) = 0$. Then clearly, (81) and (82) follow. Finally, (81) ensures that $f(\mathbf{s}) = f(\tilde{\mathbf{s}})$ and (82) implies $f(\tilde{\mathbf{s}}) = \lim_{\sigma_x^2 \to \infty} g(\tilde{\mathbf{s}})$ as emphasized in the remark below (80). This shows (73).

4) Proof of (74)*:* We show that $\hat{\mathbf{s}} \in \arg\min_{\mathbf{s}} \{\lim_{\sigma_x^2 \to \infty} g(\mathbf{s})\}$ implies that $\hat{\mathbf{s}} \in \arg\min_{\mathbf{s}} f(\mathbf{s})$. First, (73) together with (72) leads to

$$\min_{\mathbf{s}} (\lim_{\sigma_x^2 \to \infty} g(\mathbf{s})) = \min_{\mathbf{s}} f(\mathbf{s}), \tag{84}$$

i.e., $\lim_{\sigma_x^2 \to \infty} g(\mathbf{s})$ and $f(\mathbf{s})$ have necessarily the same minimum value.

Let $\hat{\mathbf{s}} \in \arg\min_{\mathbf{s}} \{\lim_{\sigma_x^2 \to \infty} g(\mathbf{s})\}$ and let $\mathbf{s}^\star \in \arg\min_{\mathbf{s}} f(\mathbf{s})$. Then, we have

$$f(\mathbf{s}^\star) \overset{(a)}{\leq} f(\hat{\mathbf{s}}) \overset{(b)}{\leq} \lim_{\sigma_x^2 \to \infty} g(\hat{\mathbf{s}}) \overset{(c)}{=} \min_{\mathbf{s}} f(\mathbf{s}) = f(\mathbf{s}^\star), \tag{85}$$

where inequality *(a)* comes from the definition of $\mathbf{s}^\star$; *(b)* is a direct consequence of (72) and *(c)* follows from the definition of $\hat{\mathbf{s}}$ and (84). Looking at the left and right-hand sides of (85), we see that equality holds throughout the expression. Therefore $\hat{\mathbf{s}} \in \arg\min_{\mathbf{s}} f(\mathbf{s})$.

5) Proof of (64)*:* We are now ready to prove the main result (64). To do so, we show that, under the hypotheses of Theorem 1, the two following inclusions hold: $\hat{\mathcal{X}} \subseteq \mathcal{X}^\star$ and $\mathcal{X}^\star \subseteq \hat{\mathcal{X}}$. Considering the definition of $\mathcal{X}^\star$ and $\hat{\mathcal{X}}$ in (63) and (68), the first inclusion immediately follows from (71) and (74).

Let us focus on $\mathcal{X}^\star \subseteq \hat{\mathcal{X}}$. Using definition (68), proving this assertion is equivalent to showing that

$$\mathbf{x}^\star(\mathbf{s}^\star) \in \hat{\mathcal{X}} \quad \forall\, \mathbf{s}^\star \in \arg\min_{\mathbf{s}} f(\mathbf{s}).$$

First, note that $\forall \mathbf{s}^\star$ one can find $\tilde{\mathbf{s}}$ (defined as in (83)) satisfying (81)-(82). Now, $\tilde{\mathbf{s}}$ is such that

$$\tilde{\mathbf{s}} \in \arg\min_{\mathbf{s}} \{ \lim_{\sigma_x^2 \to \infty} g(\mathbf{s}) \}. \tag{86}$$

Indeed, we can write the following equalities from (81), (82) and (84) respectively:

$$f(\mathbf{s}^\star) = f(\tilde{\mathbf{s}}) = \lim_{\sigma_x^2 \to \infty} g(\tilde{\mathbf{s}}) = \min_{\mathbf{s}} (\lim_{\sigma_x^2 \to \infty} g(\mathbf{s})), \tag{87}$$

and (86) follows from the last equality. Finally exploiting (74), (81) and (87), we obtain the desired result:

$$\mathbf{x}^\star(\mathbf{s}^\star) = \mathbf{x}^\star(\tilde{\mathbf{s}}) = \lim_{\sigma_x^2 \to \infty} \hat{\mathbf{x}}(\tilde{\mathbf{s}}) \in \hat{\mathcal{X}}. \tag{88}$$

REFERENCES

[1] Alan Miller, *Subset Selection in Regression, Second Editon*, Chapman and Hall/CRC, 2 edition, Apr. 2002.

[2] E. J. Candes and T. Tao, "Decoding by linear programming," *Information Theory, IEEE Transactions on*, vol. 51, no. 12, pp. 4203–4215, Dec. 2005.

[3] B. D. Jeffs and M. Gunsay, "Restoration of blurred star field images by maximally sparse optimization," *IEEE Transactions on Image Processing*, vol. 2, no. 2, pp. 202–211, Apr. 1993.

[4] J. Bobin, J. L. Starck, J. M. Fadili, Y. Moudden, and D. L. Donoho, "Morphological component analysis: an adaptative thresholding strategy," *IEEE Trans. on Image Process.*, vol. 16, no. 11, pp. 2675–2681, Nov. 2007.

[5] B. K. Natarajan, "Sparse approximate solutions to linear systems," *SIAM J. Comput.*, vol. 24, pp. 227–234, Apr. 1995.

[6] John J. Kormylo and Jerry M. Mendel, "Maximum likelihood detection and estimation of Bernoulli-Gaussian processes," *IEEE Transactions on Information Theory*, vol. 28, pp. 482–488, 1982.

[7] C. Soussen, J. Idier, D. Brie, and J. Duan, "From bernoulli-gaussian deconvolution to sparse signal restoration," *Signal Processing, IEEE Transactions on*, vol. 59, no. 10, pp. 4572–4584, Oct. 2011.

[8] I. Barbu, C. Herzet, and E. Mémin, "Sparse models and pursuit algorithms for piv tomography," in *Forum on recent developments in Volume Reconstruction techniques applied to 3D fluid and solid mechanics (FVR)*, 2011.

[9] D. L. Donoho, "Compressed sensing," *IEEE Trans. Inform. Theory*, vol. 52, no. 4, pp. 1289–1306, Apr. 2006.

[10] S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by Basis Pursuit," *SIAM J. Sci. Comp.*, vol. 20, no. 1, pp. 33–61, 1999.

[11] I. Gorodnitsky and D. R. Bhaskar, "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.

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

[13] Y. C. Pati, R. Rezaiifar, and P. S. Krishnaprasad, "Orthogonal matching pursuit: recursive function approximation with applications to wavelet decomposition," in *Proc. 27th Ann. Asilomar Conf. Signals, Systems, and Computers*, 1993.

[14] D. L. Donoho, Y. Tsaig, I. Drori, and J. L. Starck, "Sparse solution of underdetermined linear equations by stagewise orthogonal matching pursuit," 2006.

[15] S. Chen, S. A. Billings, and W. Luo, "Orthogonal least squares methods and their application to non-linear system identification," *International Journal of Control*, vol. 50, no. 5, pp. 1873–1896, Nov. 1989.

[16] T. Blumensath and M. E. Davies, "Gradient pursuits," *IEEE Trans. Signal Processing*, vol. 56, no. 6, pp. 2370–2382, June 2008.

[17] Christophe Couvreur and Yoram Bresler, "On the optimality of the backward greedy algorithm for the subset selection problem," *SIAM J. Matrix Anal. Appl.*, vol. 21, pp. 797–808, Feb. 2000.

[18] M. A. Efroymson, *Multiple Analysis Regression*, vol. 1 of *Mathematical Methods for Digital Computers*, pp. 191–203, Wiley, New York, USA, 1960.

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

[20] D. Haugland, "A bidirectional greedy heuristic for the subspace selection problem," vol. 4638 of *Lecture Notes in Computer Science*, pp. 162–176. Springer Berlin / Heidelberg, 2007.

[21] T. Zhang, "Adaptive forward-backward greedy algorithm for learning sparse representations," *IEEE Trans. Inform. Theory*, 2011.

[22] T. Blumensath and M. E. Davies, "Iterative thresholding for sparse approximations," *Journal of Fourier Analysis and Applications*, vol. 14, no. 5-6, pp. 629–654, Dec. 2008.

[23] S. Foucart, "Hard thresholding pursuit: an algorithm for compressive sensing," 2011.

[24] D. Needell and J. A. Tropp, "CoSaMP: Iterative signal recovery from incomplete and inaccurate samples," *Appl. Comput. Harmon. Anal.*, vol. 26, pp. 301–321, 2009.

[25] W. Dai and O. Milenkovic, "Subspace pursuit for compressive sensing signal reconstruction," Jan. 2009.

[26] B. A. Olshausen and D. J. Field, "Sparse coding with an overcomplete basis set: a strategy employed by V1?," *Vision Res.*, vol. 37, no. 23, pp. 3311–3325, 1997.

[27] M. Girolami, "A variational method for learning sparse and overcomplete representations," *Neural Computation*, vol. 13, no. 11, pp. 2517–2532, 2001.

[28] C. Févotte and S. J. Godsill, "Blind separation of sparse sources using Jeffrey's inverse prior and the EM algorithm," in *International Conference on Independent Component Analysis and Blind Source Separation*, 2006, pp. 593–600.

[29] A. T. Cemgil, C. Févotte, and S. J. Godsill, "Variational and stochastic inference for Bayesian source separation," *Digital Signal Processing*, vol. 17, pp. 891–913, 2007.

[30] D. P. Wipf and B. D. Rao, "Sparse Bayesian learning for basis selection," *IEEE Transactions on Signal Processing*, vol. 52, no. 8, pp. 2153–2164, Aug. 2004.

[31] D. P. Wipf, J. A. Palmer, and B. D. Rao, "Perspectives on sparse Bayesian learning," in *Neural Inform. Process. Syst.*, 2004, vol. 16.

[32] E. G. Larrson and Y. Selen, "Linear regression with sparse parameter vector," *IEEE Trans. Signal Processing*, vol. 55, no. 2, pp. 451–460, Feb. 2007.

[33] P. Schniter, L. C. Potter, and J. Ziniel, "Fast Bayesian matching pursuit," in *IEEE Information Theory and Applications Workshop*, 2008, pp. 326–333.

[34] H. Zayyani, M. Babaie-Zadeh, and C. Jutten, "Decoding real field codes by an iterative Expectation-Maximization (EM) algorithm," in *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'*, 2008.

[35] D. Baron, S. Sarvotham, and R. G. Baraniuk, "Bayesian compressive sensing via belief propagation," June 2009.

[36] C. Herzet and A. Drémeau, "Sparse representation algorithms based on mean-field approximations," in *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'*, 2010.

[37] C. Herzet and A. Drémeau, "Bayesian pursuit algorithms," in *submitted to EURASIP European Signal Processing Conference, EUSIPCO'*, 2010.

[38] A. Drémeau and C. Herzet, "Soft bayesian pursuit algorithm for sparse representations," in *IEEE International Workshop on Statistical Signal Processing 2011 (SSP'11)*, 2011.

[39] K. Qiu and A. Dogandzic, "ECME thresholding methods for sparse signal reconstruction," 2011.

[40] A. J. Miller, *Subset Selection in Regression*, Chapman & Hall/CRC, 2002.

[41] N. R. Draper, I. Guttman, and H. Kanemasu, "The distribution of certain regression statistics," *Biometrika*, vol. 58, no. 2, 1971.

[42] P. Pope and J. Webster, "The use of an f-statistic in stepwise regression procedures," *Technometrics*, vol. 14, no. 2, 1972.

[43] V. Cevher, M. F. Duarte, C. Hegde, and R. G. Baraniuk, "Sparse signal recovery using markov random fields," in *Advances in Neural Information Processing Systems (NIPS)*, Vancouver, Canada, Dec. 2008.

[44] T. Faktor, Y. C. Eldar, and M. Elad, "Exploiting statistical dependencies in sparse representations for signal recovery," .

[45] A. Dremeau, C. Herzet, and L. Daudet, "Structured bayesian orthogonal matching pursuit," in *Proc. IEEE Int'l Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2012.

[46] "http://sparselab.stanford.edu/," .

[47] "http://www.ee.ic.ac.uk/wei.dai/research.html," .

[48] "http://users.fmrib.ox.ac.uk/∼tblumens/sparsify/sparsify.html," .

[49] "http://www.math.drexel.edu/∼foucart/software.htm," .

[50] "http://www.irisa.fr/fluminance/team/herzet/index.html," .

[51] J. M. Mendel, *Lessons in Estimation Theory for Signal Processing Communications and Control*, Prentice Hall Signal Processing Series, Englewood Cliffs, NJ, 1995.

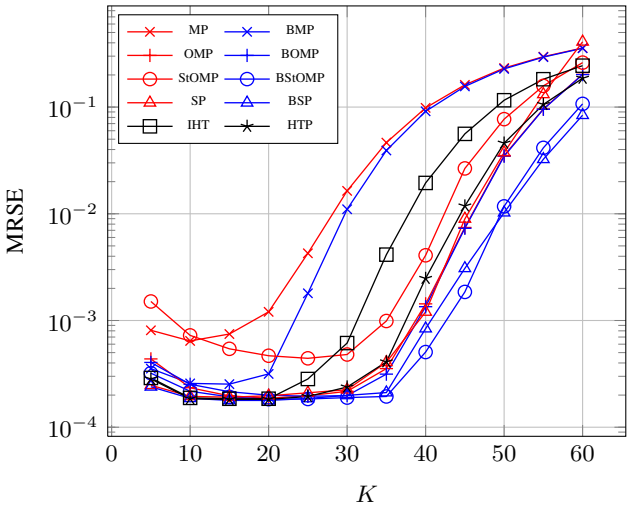

Fig. 1. MRSE versus the number of non-zero coefficients K. The non-zero coefficients in $\mathbf{x}$ follow the "Gaussian model".

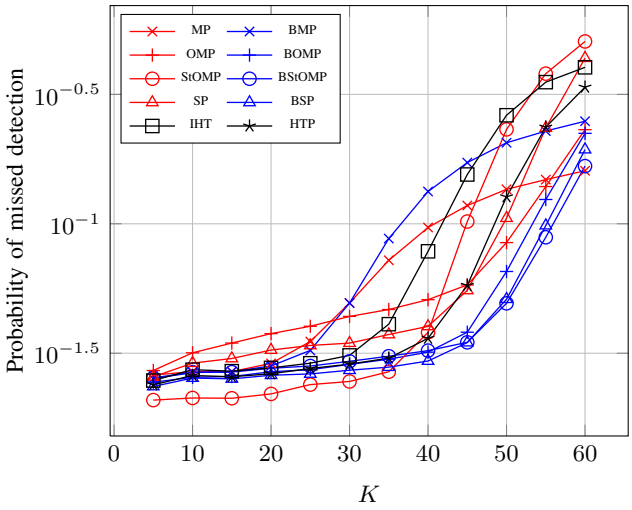

Fig. 2. Probability of missed detection versus the number of non-zero coefficients K. The non-zero coefficients in $\mathbf{x}$ follow the "Gaussian model".

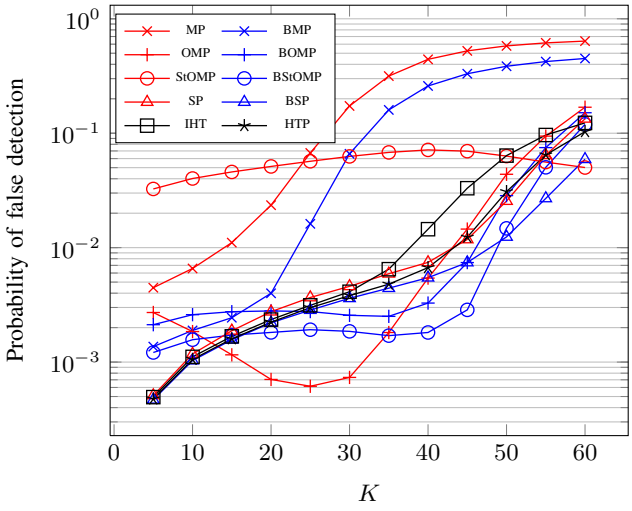

Fig. 3. Probability of false detection versus the number of non-zero coefficients K. The non-zero coefficients in $\mathbf{x}$ follow the "Gaussian model".

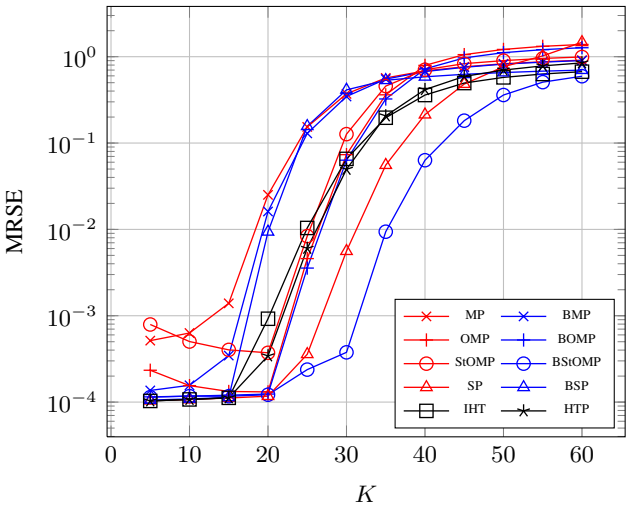

Fig. 4. MRSE versus the number of non-zero coefficients K. The non-zero coefficients in $\mathbf{x}$ follow the "$0-1$ model".

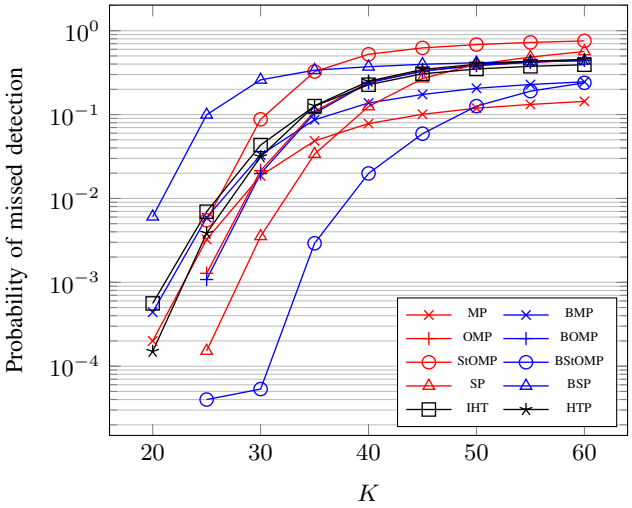

Fig. 5. Probability of missed detection versus the number of non-zero coefficients K. The non-zero coefficients in $\mathbf{x}$ follow the "0-1 model".

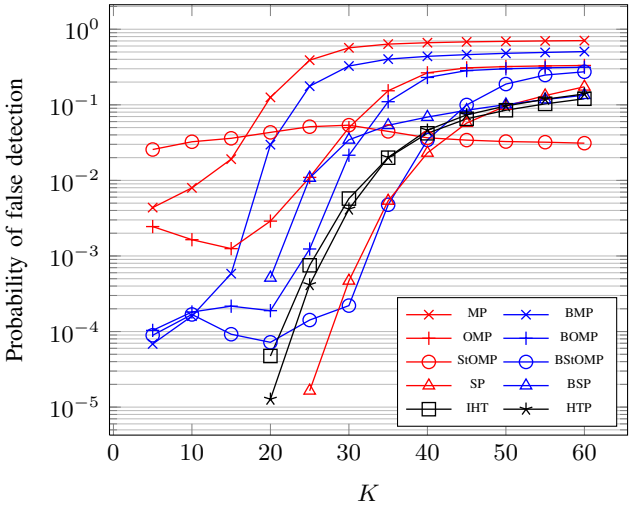

Fig. 6. Probability of false detection versus the number of non-zero coefficients K. The non-zero coefficients in $\mathbf{x}$ follow the "0-1 model".

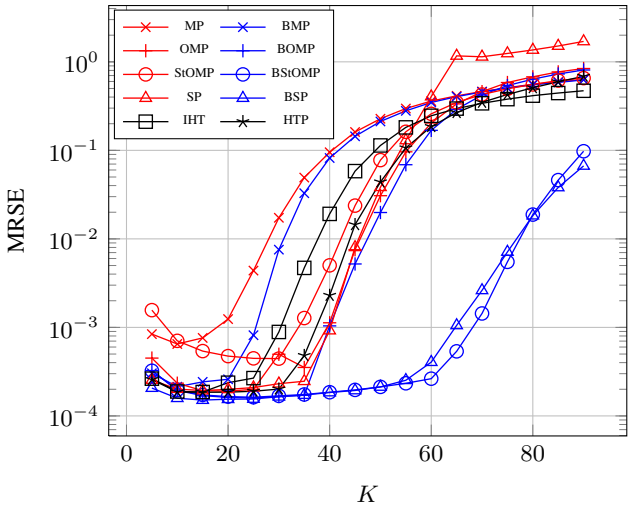

Fig. 7. MRSE versus the number of non-zero coefficients K. The Bayesian algorithms have a perfect knowledge of p_i's.

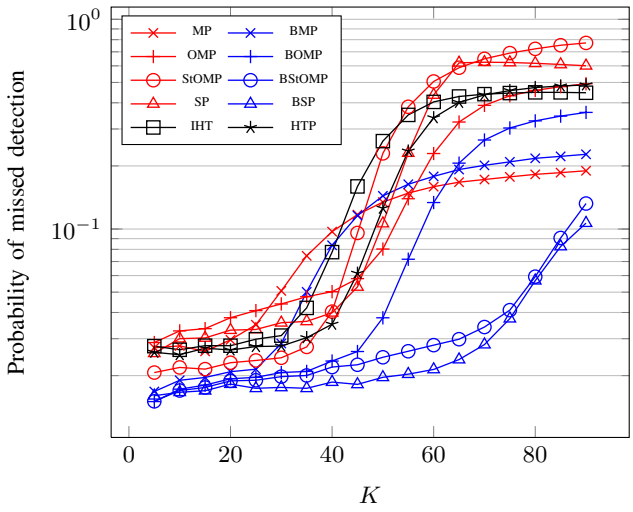

Fig. 8. Probability of missed detection versus the number of non-zero coefficients K. The Bayesian algorithms have a perfect knowledge of p_i's.

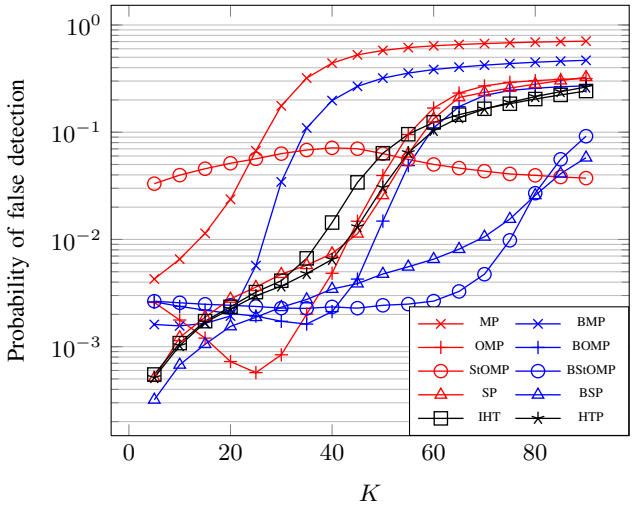

Fig. 9. Probability of false detection versus the number of non-zero coefficients K. The Bayesian algorithms have a perfect knowledge of p_i's.

Initialization : $\mathbf{x}^{(0)} = 0$, $\hat{\mathbf{s}}^{(0)} = 0$, $n = 1$.
Repeat :

1. Update of the residual:

$$\mathbf{r}(\hat{\mathbf{x}}^{(n-1)}, \hat{\mathbf{s}}^{(n-1)}) = \mathbf{y} - \mathbf{D}_{\hat{\mathbf{s}}^{(n-1)}} \hat{\mathbf{x}}^{(n-1)}_{\hat{\mathbf{s}}^{(n-1)}}.$$

2. Evaluate the solution of (45) $\forall i$:

$$\tilde{s}_i^{(n)} = \begin{cases} 1 & \text{if } \langle \mathbf{r}(\hat{\mathbf{x}}^{(n-1)}, \hat{\mathbf{s}}^{(n-1)}) + \hat{x}_i^{(n-1)} \mathbf{d}_i, \mathbf{d}_i \rangle^2 > T_i \\ 0 & \text{otherwise} \end{cases} \quad (89)$$

$$\tilde{x}_i^{(n)} = \frac{\tilde{s}_i^{(n)} \sigma_x^2}{\sigma_x^2 + \sigma_w^2} \left(\hat{x}_i^{(n-1)} + \langle \mathbf{r}(\hat{\mathbf{x}}^{(n-1)}, \hat{\mathbf{s}}^{(n-1)}), \mathbf{d}_i \rangle \right) \quad (90)$$

with

$$T_i \triangleq 2\sigma_w^2 \frac{\sigma_x^2 + \sigma_w^2}{\sigma_x^2} \log \left(\frac{1 - p_i}{p_i} \right). \quad (91)$$

3. Choice of the index to be modified (44):

$$j = \arg\max_i \rho_i^{(n)}(\tilde{s}_i^{(n)}). \quad (92)$$

4. Support update:

$$s_i^{(n)} = \begin{cases} \tilde{s}_i^{(n)} & \text{if } i = j \\ \hat{s}_i^{(n-1)} & \text{otherwise,} \end{cases} \quad (93)$$

5. Coefficient update:

$$x_i^{(n)} = \begin{cases} \tilde{x}_i^{(n)} & \text{if } i = j \\ \hat{x}_i^{(n-1)} & \text{otherwise.} \end{cases} \quad (94)$$

TABLE I
BMP ALGORITHM

Initialization : $\mathbf{x}^{(0)} = 0$, $\hat{\mathbf{s}}^{(0)} = 0$, $n = 1$.
Repeat :

1. Support update:

$$\hat{\mathbf{s}}^{(n)} = \arg\max_{\mathbf{s} \in \mathcal{S}_P} \left\{ \sum_i s_i \, \rho_i^{(n)}(1) + (1 - s_i) \, \rho_i^{(n)}(0) \right\}.$$

2. Update coeffcient from (33).

3. Support update:

$$\hat{\mathbf{s}}^{(n+1)} = \arg\max_{\mathbf{s} \in \mathcal{S}_K} \left\{ \sum_i s_i \, \rho_i^{(n+1)}(1) + (1 - s_i) \, \rho_i^{(n+1)}(0) \right\}.$$

4. Update coeffcient from (33).

TABLE II
BSP ALGORITHM