

# BAYESIAN PURSUIT ALGORITHM FOR SPARSE REPRESENTATION

H. Zayyani, M. Babaie-Zadeh\*

C. Jutten

Sharif University of Technology  
Department Of Electrical Engineering  
Tehran, Iran

Institut National Polytechnique de Grenoble (INPG)  
Laboratoire des Images et de Signaux (LIS)  
Grenoble, France

## ABSTRACT

In this paper, we propose a Bayesian Pursuit algorithm for sparse representation. It uses both the simplicity of the pursuit algorithms and optimal Bayesian framework to determine the active atoms in the sparse representation of the signal. We show that using the Bayesian Hypothesis testing to determine the active atoms from the correlations, leads to an efficient activity measure. Simulation results show that our suggested algorithm has the best performance among the algorithms which are implemented in our simulations.

**Index Terms**— Sparse representation, Sparse Component Analysis (SCA), Compressed Sensing (CS), Pursuit algorithms.

## 1. INTRODUCTION

Finding (sufficiently) sparse solutions of underdetermined systems of linear equations (possibly in the noisy case) has been used extensively in the signal processing community. This problem has been found applications in a wide range of diverse fields. Some applications are Blind Source Separation (BSS) and Sparse Component Analysis (SCA) [1], decoding [2] and Compressive Sensing (CS) [3], [4].

The problem can be defined in various context such as sparse representation, SCA or Compressed Sensing (CS). Here, we use the notation of the sparse representations of signals. Let the model be:

$$\mathbf{x} = \Phi \mathbf{y} + \mathbf{e} \quad (1)$$

where  $\mathbf{x}$  is an  $n \times 1$  signal vector and  $\mathbf{y}$  is an  $m \times 1$  sparse coefficient vector and  $\Phi$  is an  $n \times m$  matrix called dictionary and  $\mathbf{e}$  is a  $n \times 1$  error vector. It is assumed that  $n < m$  which means that the signal length is smaller than the number of columns of the dictionary (which are called atoms). The main assumption is that the signal has a sparse representation in the dictionary. The main goal is to find the sparse coefficient vector  $\mathbf{y}$  based on the signal  $\mathbf{x}$  and knowing the dictionary  $\Phi$ .

In different fields of application, the interpretations of vectors are different, but in all of them the model is as (1). For example, in the context of CS,  $\Phi$  is the measurement matrix and  $\mathbf{x}$  is the very few measurements of the signal and  $\mathbf{y}$  is the sparse representation of the true signal in a probable domain. In the context of SCA,  $\Phi$  can be interpreted as mixing matrix and  $\mathbf{x}$  is the mixture vector and  $\mathbf{y}$  is the source vector.

However, finding the sparsest solution, that is, the solution with the minimum number of nonzero elements, is a hard combinatorial problem. So, different methods should be used to solve the problem in a tractable way. There are many different methods for sparse representation. Most of them are divided in two main categories: 1) The optimization approaches and 2) The greedy approaches. The first category convert the problem to an optimization problem and then use different methods to solve that. But, the second methods try to find active coefficients (with nonzero elements) directly by an algorithm.

In the first category, the most successful approach which is the Basis Pursuit (BP) [5], suggests a convexification of the problem by replacing the  $\ell^0$ -norm<sup>1</sup> with the  $\ell^1$ -norm. It can then be implemented by Linear Programming (LP) methods. Another method is the FOCUSS algorithm which uses  $\ell^p$ -norm with  $p \leq 1$  as a replacement for the  $\ell^0$ -norm [6]. Recently, a novel Expectation-Maximization (EM) algorithm [7] and a Bayesian Compressive Sensing (BCS) algorithm [8] are proposed to solve the problem in a Bayesian framework. There is also a new method for minimization a smoothed version of the  $\ell^0$ -norm which is called SLO method [9].

The other category is the greedy algorithms which chooses the active coefficients by iterative algorithms. Generally, they use the correlation between the signal (or residual signal) and the atoms of the dictionary as an informative measure for deciding which coefficients are actually active (or nonzero). These algorithms are Matching Pursuit (MP) [10], Orthogonal Matching Pursuit (OMP) [11], Stage-wise OMP (StOMP) [12] and Gradient Pursuit (GP) [13].

Our proposed method uses the simplicity of the greedy

\*Thanks to ITRC and INSF and ISMO (by the GundiShapour program) for funding.

<sup>1</sup>It is not a mathematical norm, but we use this name because it is mainly used in the literature.

pursuit algorithms while simultaneously uses Bayesian tools for optimal selection of active atoms.

## 2. SYSTEM MODEL

The noise vector in the model (1) is assumed to be zero-mean Gaussian with covariance matrix  $\sigma_e^2 \mathbf{I}$ . We model the sparse coefficients as follows. In our model the coefficients are inactive with probability  $p$ , and are active with probability  $1 - p$  (sparsity of  $\mathbf{s}$  implies that  $p$  should be near 1). In the inactive case, the value of the coefficients is zero and in the active case the value is obtained from a Gaussian distribution. We call this model the ‘spiky model’ which is a special case of the Bernoulli-Gaussian model with the variance of the inactive samples being zero. This model has been also used in [7]. It is suitable for sparse representation of a signal where we want to decompose a signal as a combination of only a few atoms of the dictionary and the coefficients of the other atoms are zero. So, the probability density of the coefficients in our problem is:

$$p(y_i) = p\delta(y_i) + (1 - p)N(0, \sigma_r^2) \quad (2)$$

In this model, each coefficient can be written as  $y_i = q_i r_i$  where  $q_i$  is a binary variable (with binomial distribution) and  $r_i$  is the amplitude of the  $i$ ’th coefficient with Gaussian distribution. Each element  $q_i$  shows the activity of the corresponding coefficient (or corresponding atom). That is:

$$q_i = \begin{cases} 1 & \text{if } y_i \text{ is active (with probability } 1 - p) \\ 0 & \text{if } y_i \text{ is inactive (with probability } p) \end{cases} \quad (3)$$

Consequently, the probability  $p(\mathbf{q})$  of activity vector  $\mathbf{q} \triangleq (q_1, q_2, \dots, q_m)^T$  is equal to:

$$p(\mathbf{q}) = (1 - p)^{n_a} (p)^{m - n_a} \quad (4)$$

where  $n_a$  is the number of active coefficients i.e., the number of 1’s in  $\mathbf{q}$ . So, the coefficient vector can be written as:

$$\mathbf{y} = \mathbf{Q}\mathbf{r}, \quad \mathbf{Q} = \text{diag}(\mathbf{q}) \quad (5)$$

where  $\mathbf{q}$  and  $\mathbf{r} \triangleq (r_1, r_2, \dots, r_m)^T$  are the ‘activity vector’ and ‘amplitude vector’, respectively.

## 3. BAYESIAN PURSUIT ALGORITHM

The main task in the sparse recovery algorithms is to determine which atoms are active in the sparse representation of the signal. In some pursuit algorithms (e.g., MP), it is determined by correlation maximization. In some other pursuit algorithms (e.g., StOMP), it is done by comparing the correlations with a threshold. In optimal Bayesian framework and in the MAP sense, it is done with posterior maximization over all possible activity vectors. But, here we want to determine

it by a Bayesian hypothesis testing from the correlations. To develop the hypothesis testing in our Bayesian Pursuit Algorithm (BPA), we write (1) as:

$$\mathbf{x} = \sum_{i=1}^m \varphi_i y_i + \mathbf{e} \quad (6)$$

where  $\varphi_i$  is the columns of the dictionary or atoms. So, the correlations between the original signal and the atoms are:

$$z_j \triangleq \langle \mathbf{x}, \varphi_j \rangle = y_j + \sum_{i \neq j} y_i b_{ij} + v_j \quad (7)$$

where  $b_{ij} \triangleq \langle \varphi_i, \varphi_j \rangle$  and  $v_j \triangleq \langle \mathbf{e}, \varphi_j \rangle$  and the atoms are assumed to have unit norm.

To do a Bayesian hypothesis testing based on correlations for determining the activity of the  $j$ ’th atom, we should compute the posteriors  $p(H_1|\mathbf{z})$  and  $p(H_2|\mathbf{z})$ , where the hypothesis  $H_1$  is the hypothesis that the  $j$ ’th atom is active and  $H_2$  is the hypothesis that the  $j$ ’th atom is inactive. To obtain simple algorithm like pursuit algorithms, we assume that we know the previous estimations of other coefficients (except the  $j$ ’th coefficient). And now, we want to know the activity of only  $j$ ’th atom and then update just only the  $j$ ’th coefficient.

Since we assume that we know the previous estimations of other coefficients, so (7) can be written as:

$$z_j - \sum_{i \neq j} \hat{y}_i b_{ij} = y_j + \sum_{i \neq j} (y_i - \hat{y}_i) b_{ij} + v_j \quad (8)$$

where  $\hat{y}_i$  is the estimation of the  $i$ ’th coefficient up to the current iteration. With the following definitions:

$$\begin{aligned} m_j &\triangleq \sum_{i \neq j} \hat{y}_i b_{ij} \\ \gamma_j &\triangleq \sum_{i \neq j} (y_i - \hat{y}_i) b_{ij} + v_j \end{aligned} \quad (9)$$

Then two hypothesis  $H_1$  and  $H_2$  are equivalent to:

$$\text{Hypotheses : } \begin{cases} H_1 : z_j - m_j = r_j + \gamma_j \\ H_2 : z_j - m_j = \gamma_j \end{cases} \quad (10)$$

where  $m_j$  is known and  $\gamma_j$  has a flavour of noise and error. (10) resembles a classical detection problem. The optimal hypothesis testing involves the computation of the overall posteriors  $p(H_1|\mathbf{z})$  and  $p(H_2|\mathbf{z})$ . But, with the previous assumptions and formulations, we reach a relatively simple detection problem as in (10). So, for the simplicity of the algorithm like the pursuit algorithms, we rely only to the simpler posteriors as  $p(H_1|z_j)$  and  $p(H_2|z_j)$ . So, the hypothesis  $H_1$  is assumed to be true when  $p(H_1|z_j) > p(H_2|z_j)$ . Based on the Bayes rule, the above posteriors are proportional to  $p(H_1)p(z_j|H_1)$  and  $p(H_2)p(z_j|H_2)$  respectively. The prior probabilities for the hypotheses are  $p(H_1) = 1 - p$  and  $p(H_2) = p$  where

$p$  is defined in Section 2. We assume that the coefficient errors  $(y_i - \hat{y}_i)$  has a Gaussian distribution with variance  $\sigma_{i,e_y}^2$ . Hence, by assuming that the error  $(y_i - \hat{y}_i)$  is Gaussian, the term  $\gamma_j$  is Gaussian and we assume whose variance is  $\sigma_{\gamma_j}^2$ . Therefore, we have:

$$\frac{(1-p)}{\sqrt{2\pi(\sigma_{\gamma_j}^2 + \sigma_r^2)}} \exp\left(\frac{-(z_j - m_j)^2}{2(\sigma_{\gamma_j}^2 + \sigma_r^2)}\right) > \frac{p}{\sqrt{2\pi\sigma_{\gamma_j}^2}} \exp\left(\frac{-(z_j - m_j)^2}{2\sigma_{\gamma_j}^2}\right) \quad (11)$$

Simplyfying (11) with the assumption that the unknown parameters ( $p$ ,  $\sigma_r$  and  $\sigma_{\gamma_j}$ ) are known, leads to the following decision rule for the hypothesis testing:

$$\text{Activity}(y_j) \triangleq |z_j - m_j| > \text{Th}_j \quad (12)$$

where  $\text{Th}_j$  is defined as:

$$\text{Th}_j \triangleq \frac{\sigma_{\gamma_j}}{\sigma_r} \sqrt{2(\sigma_r^2 + \sigma_{\gamma_j}^2) \ln\left(\frac{p}{1-p} \frac{\sqrt{\sigma_r^2 + \sigma_{\gamma_j}^2}}{\sigma_{\gamma_j}}\right)} \quad (13)$$

However, (13) determines the optimum threshold, but it depends on unknown parameters ( $p$ ,  $\sigma_r$  and  $\sigma_{\gamma_j}$ ) which should be estimated from the original signal ( $\mathbf{x}$ ). To estimate the parameters  $p$ ,  $\sigma_r$  and  $\sigma_e$ , we can use similar formulas as in [7] which are:

$$\hat{p} = \frac{\|\mathbf{q}\|_0}{m} \quad (14)$$

$$\hat{\sigma}_e = \frac{\|\mathbf{x} - \Phi\hat{\mathbf{y}}\|_2}{\sqrt{n}} \quad (15)$$

$$\hat{\sigma}_r = \frac{\|\mathbf{r}\|_2}{\sqrt{m}} \quad (16)$$

The problem here is to estimate the parameter  $\sigma_{\gamma_j}$  which is the standard deviation of  $\gamma_j$  in (9). We assume the independence between  $v_j$  and coefficient error  $(y_i - \hat{y}_i)$  and also the independence between distinct coefficient errors  $y_i - \hat{y}_i$  and  $y_j - \hat{y}_j$  for  $i \neq j$ , and also knowing that  $v_j$  is a Gaussian random variable with the similar variance as  $e_j$  which is uniformly  $\sigma_e^2$ , results in the following formula for the parameter estimations:

$$\sigma_{\gamma_j}^2 = \sigma_e^2 + \sum_{i \neq j} b_{ij}^2 \sigma_{i,e_y}^2 \quad (17)$$

where  $\sigma_{i,e_y}^2$  is the variance of the coefficient error  $y_i - \hat{y}_i$ . If our algorithm converges, we expect that it decreases. So, we force that this error variance decreases linearly with a coefficient  $\alpha$  which is less but near one. So, we select this decreasing sequence as:

$$\sigma_{i,e_y}^{(n+1)} = \alpha \sigma_{i,e_y}^{(n)} \quad (18)$$

where parameter  $\alpha$  determines the rate of convergence.

As we can see from (13), the optimum threshold is changed from an initial large value to a small final value. The initial value and the final value of the threshold are:

$$\begin{aligned} \text{Th}_j^{(0)} &= \text{Th}|_{\sigma_{\gamma_j}^{(0)}} \\ \text{Th}_j^{(\infty)} &= \text{Th}|_{\sigma_{\gamma_j} = \sigma_e} \approx K \sigma_e \end{aligned} \quad (19)$$

where  $K = \sqrt{2 \ln\left(\frac{p}{1-p} \frac{\sigma_r}{\sigma_e}\right)}$ . As we can see from (19), the initial thresholds are different from one coefficient to another. But, all the thresholds have converged to the same threshold which does not depend on the coefficient.

As the value of threshold changes from a large value to a small value, the algorithm can detect more and more atoms. At first iterations, the optimal thresholding strategy in (13) changes the thresholds very fast and then after some iterations, the thresholds converge to the final small value.

After updating the activity vector based on BPA decision rule in (12), the estimation of amplitude vector  $\mathbf{r}$  which was defined in Section 2, based on this estimated activity vector can be done with a Linear Least Square (LLS) estimation [7] as:

$$\hat{\mathbf{r}} = \sigma_r^2 \hat{\mathbf{Q}} \Phi^T (\sigma_r^2 \Phi \hat{\mathbf{Q}} \Phi^T + \sigma_e^2 \mathbf{I})^{-1} \mathbf{x} \quad (20)$$

where  $\hat{\mathbf{Q}} = \text{diag}(\hat{\mathbf{q}})$  and  $\hat{\mathbf{q}}$  is the updated activity vector.

#### 4. EXPERIMENTS

The proposed BPA algorithm is investigated in this section. The comparison of our BPA algorithm is done with some other algorithms in the literature both from the performance and the complexity viewpoint. The performance of the algorithms are compared with the Signal to Noise Ratio between the true coefficient and the recovered coefficients, which is defined as:

$$\text{SNR}_o \triangleq 10 \log\left(\frac{\|\mathbf{y}\|_2}{\|\mathbf{y} - \hat{\mathbf{y}}\|_2}\right) \quad (21)$$

where the index indicates that it is an output SNR. We define another measure which determines the noise level. We refer to it as input SNR and is defined as:

$$\text{SNR}_i \triangleq 20 \log\left(\frac{\sigma_r}{\sigma_e}\right) \quad (22)$$

We use the CPU time as a measure of complexity. Our simulations were performed in MATLAB7.0 environment using an Intel 2.80 GHz processor with 1024 MB of RAM and under Linux operating system.

In our experiment, we used a random dictionary matrix with uniform distributed elements from  $[-1, 1]$ , and then normalized its columns. The dimension of our problem was selected as  $m = 512$  for the number of atoms and  $n = 256$  for the signal length. For the sparse coefficients, we used the model in (2) with the probability  $p = 0.9$  and unit variance for the active coefficients ( $\sigma_r = 1$ ). So, approximately 51 atoms

are active in the sparse representation of the signal. The noise level or error is considered to have a Gaussian distribution with different variances. The measure of performance which is the output SNR in (21) is averaged over 100 different random realizations of the dictionary, sparse coefficients and noise vector.

For the initialization of the unknown statistical parameters ( $p$ ,  $\sigma_r$  and  $\sigma_e$ ), we use  $\hat{p}^{(0)} = 0.8$ ,  $\hat{\sigma}_r^{(0)} = \frac{\|\mathbf{x}\|_2}{\sqrt{m(1-\hat{p}^{(0)})}}$  and  $\hat{\sigma}_e^{(0)} = \frac{\hat{\sigma}_r^{(0)}}{5}$  which are similar to those used in [7]. The important note is to use an overestimate of noise variance ( $\sigma_e$ ) in the initial iteration. We also used 20 iterations for BPA algorithm and the simulation parameter  $\alpha$  was selected as 0.9.

In this experiment, we compared the suggested BPA algorithm with BP, MP, OMP, StOMP, SL0, EM and GP. For MP and OMP, we used 100 and 50 iterations, respectively. For StOMP, we used 20 iterations and the parameter  $t$  which determines the threshold is selected as  $t = 2.5$  [12]. For SL0, we used 7 iterations with the sequence of variances equal to [1 .5 .2 .1 .05 .02 0.01] [9]. For the EM algorithm, we used both 5 iterations for the overall EM algorithm and 4 iterations for the M-step [7]. Figure. 1 shows the performance of various algorithms versus the noise level. It shows that the best performance is due to our BPA algorithm.

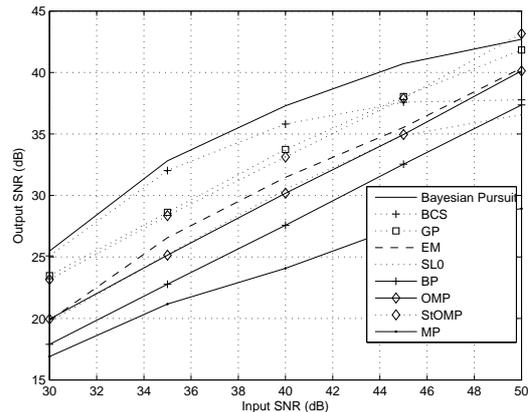
We computed the average simulation time for various algorithms. These are .048, .05, 0.06, .2, 1.8, 2.5, 3.2, 3.8 and 38 seconds for MP, SL0, GP, BCS, BP, StOMP, BPA, EM and OMP respectively. So, the complexity of the BPA algorithm is twice the BP and is roughly the same as StOMP and EM.

## 5. CONCLUSION

In this paper, we suggested the BPA algorithm which determines the active atoms based on a Bayesian hypothesis testing from the correlations of the signal with the atoms of the dictionary. Simulations show the advantage of the proposed method over some of the state-of-the-art algorithms in terms of performance.

## 6. REFERENCES

- [1] R. Gribonval and S. Lesage, "A survey of sparse component analysis for blind source separation: principles, perspectives, and new challenges," *ESSAN'06*, pp. 323–330, 2006.
- [2] E.J. Candes and T. Tao, "Decoding by linear programming," *IEEE Trans. Inf. Theory*, vol. 51, pp. 4203–4215, December 2005.
- [3] D.L. Donoho, "Compressed sensing," *IEEE Trans. Inf. Theory*, vol. 52, pp. 1289–1306, April 2006.
- [4] R. Baraniuk, "Compressive sensing," *IEEE Signal. Process. Magazine*, vol. 24, pp. 118–121, July 2007.



**Fig. 1.** The output SNR versus input SNR for various algorithms. The parameters are  $m = 512$ ,  $n = 256$ ,  $p = 0.9$ ,  $\sigma_r = 1$ ,  $\alpha = 0.9$  and 20 iterations are used for BPA.

- [5] S.S. Chen, D.L. Donoho, and M.A. Saunders, "Atomic decomposition by basis pursuit," *SIAM Journal on Scientific Computing*, vol. 20, pp. 33–61, 1998.
- [6] I.F. Gorodnitski and B.D. Rao, "Sparse signal reconstruction from limited data using focus: a re-weighted norm minimization algorithm," *IEEE Trans. Signal. Proc.*, vol. 45, pp. 600–616, 1997.
- [7] H. Zayyani, M. Babaie-zadeh, and C. Jutten, "Decoding real-field codes by an iterative expectation-maximization algorithm," *ICASSP 2008*, pp. 3169–3172, 2008.
- [8] S. Ji, Y. Xue, and L. Carin, "Bayesian compressive sensing," *IEEE Trans. Signal. Proc.*, vol. 56, pp. 2346–2356, June 2008.
- [9] H. Mohimani, M. Babaie-zadeh, and C. Jutten, "Fast sparse representation based on smoothed  $\ell^0$  norm," *To Appear in IEEE Trans. Signal. Proc.*, 2008.
- [10] S. Mallat and Z. Zhang, "Matching pursuits with time-frequency dictionaries," *IEEE Trans. Signal. Proc.*, vol. 41, pp. 3397–3415, 1993.
- [11] Y.C. Pati, R. Rezaifar, and P.S. Krishnaparsad, "Orthogonal matching pursuit: recursive function approximation with application to wavelet decomposition," in *proceeding of the 27th Annual Asilomar conference on Signals, Systems, and Computers*, pp. 40–44, 1993.
- [12] D.L. Donoho, Y. Tsaig, I. Drori, and J.L. Starck, "Sparse solution of underdetermined linear equations by stage-wise orthogonal matching pursuit," *Preprint*, 2006.
- [13] T. Blumensath and M. Davis, "Gradient pursuits," *IEEE Trans. Signal. Proc.*, vol. 56, pp. 2370–2382, June 2008.