FLEXIBLE TREE-SEARCH BASED ORTHOGONAL MATCHING PURSUIT ALGORITHM

Güneş Z. Karabulut[1], *Lucia Moura*[1], *Daniel Panario*[2], *Abbas Yongaçoğlu*[1]

[1]School of Information Technology and Eng.
University of Ottawa
K1N 6N5, Ontario, Canada
Email: {gkarabul, lucia, yongacog}@site.uottawa.ca

[2]School of Mathematics and Statistics
Carleton University
K1S 5B6, Ontario, Canada
Email: daniel@math.carleton.ca

ABSTRACT

The orthogonal matching pursuit (OMP) algorithm is an adaptive nonlinear algorithm for signal decomposition using an overcomplete dictionary. In [1], a tree-search based orthogonal matching pursuit (TB-OMP) is proposed. Although the TB-OMP algorithm improves the approximation performance, its computation time requirement increases exponentially making the algorithm impractical for certain applications. In this paper, we propose the flexible tree-search based orthogonal matching pursuit (FTB-OMP). The algorithm provides design parameters that give flexibility to establish a tradeoff between approximation performance and experimental time complexity. Sparse signal representations are frequently required in problems related to signal processing and communication areas. The proposed FTB-OMP algorithm is a promising solution for such problems.

1. INTRODUCTION

The problem of basis selection for signal decomposition consists of determining a small, possibly the smallest, subset of vectors chosen from a large redundant set of vectors to match the given data. In the literature, basis selection algorithms are adopted to the specific application considered. Some of these applications are time/frequency representations [2] and speech coding [3].

Finding the smallest basis set is NP hard [4]. Hence it is not expected that one could find an efficient algorithm to solve this problem exactly. Greedy heuristic algorithms have been proposed as a fast method to obtain approximate solutions. These algorithms employ sequential selection of basis vectors from an overcomplete set of vectors called dictionary. In this paper, these algorithms are referred to as sequential basis selection (SBS) algorithms. SBS algorithms include matching pursuit (MP) [2], orthogonal matching pursuit (OMP) [5], and order recursive matching pursuit (ORMP) [4]. The OMP algorithm gives a good compromise between the performance and complexity among them [1]. Hence we focus on OMP.

A shortcoming of the OMP algorithm is that an erroneous decision on the initial iterations prevents the algorithm to find a good approximation to the input signal. In [1], a tree-search based OMP (TB-OMP) is proposed by substituting a single vector selection at each iteration by considering L possible vectors. In this way, the greedy heuristic is substituted by a more complete search on a tree with branching factor L. The TB-OMP algorithm dramatically improves the approximation performance of OMP, since it explores several choices at each iteration. However, algorithm's time complexity increases exponentially, making the algorithm impractical for applications where speed is a concern.

In this paper, we propose an efficient, flexible tree-search based orthogonal matching pursuit (FTB-OMP) algorithm for signal representation. The efficiency is achieved by reducing the number of children as the depth of nodes increases via the exponentially decaying characteristics of the search algorithm. Further time efficiency is achieved by using a correlation-based pruning in the search tree. Through the use of the design parameters, FTB-OMP offers numerous variations which range from OMP to TB-OMP. It should be noted that the exponentially decaying tree-search strategy is proposed in this paper for the first time, to the best of authors' knowledge.

Solving an overcomplete set of linear equations is encountered in several communications and signal processing applications, such as the angle of arrival detection [6] and the channel estimation [7] problems. The flexible and efficient nature of the proposed FTB-OMP algorithm, achieved via the design parameters, makes it a promising candidate for applications where a sparse signal representation is of interest.

This paper is organized as follows. In Section 2, a literature review of the existing OMP and TB-OMP algorithms is given. The proposed FTB-OMP algorithm is presented in Section 3. In Section 4, we give experimental results comparing approximation and detection performance and computer running times of the proposed algorithm. Conclusions are given in Section 5.

2. BACKGROUND ON EXISTING ALGORITHMS

Let $\mathcal{D} = \{a_k\}_{k=1}^n$ be a dictionary of vectors which is highly redundant, i.e. $a_k \in \mathbb{C}^m$ and $m << n$ with $\mathbb{C}^m = \text{Span}\,(\mathcal{D})$. The best basis selection problem can be viewed as finding the sparsest solution to a linear system of equations. More precisely, if we form a matrix A from the columns of the dictionary $\mathcal{D}$, $A = [a_1, a_2, \ldots, a_n]$, the problem can be stated as finding an $\bar{\mathbf{x}}$, with at most r non-zero entries such that

$$\|A\bar{\mathbf{x}} - \mathbf{x}\| \leq \epsilon, \tag{1}$$

for given $\epsilon \geq 0$, and $r > 1$. For $\epsilon = 0$, i.e. the perfect representation case, the problem reduces to solving the system $A\bar{\mathbf{x}} = \mathbf{x}$.

2.1. Orthogonal Matching Pursuit Algorithm

The orthogonal matching pursuit (OMP) algorithm is proposed in [5]. The OMP algorithm is also called modified matching pursuit algorithm in [8], from which we take the notation presented here. Let the residual vector after the p^{th} iteration be denoted by b_p, with $b_0 = \mathbf{x}$. Let P_{S_p} denote the orthogonal projection matrix onto the range space of S_p, and $P_{S_p}^{\perp} = I - P_{S_p}$ denote its orthogonal complement, with $P_{S_0} = 0$ and $P_{S_0}^{\perp} = I$. The projection matrix on the space spanned by a_k, with $\|a_k\| = 1$, is $P_{a_k} = a_k a_k^T$. The algorithm terminates after r iterations.

The OMP algorithm selects k_p in the p^{th} iteration by finding the vector best aligned with the residual obtained by projecting b onto the orthogonal complement of the range space S_{p-1}, that is

$$k_p \quad = \quad \arg\max_l |a_l^T b_{p-1}|, \quad l \notin I_{p-1}. \tag{2}$$

With the initial values, $\hat{a}_{k_p}^0 = a_{k_p}$, $q_0 = 0$, applying the orthogonalization process as

$$\hat{a}_{k_p}^l = \hat{a}_{k_p}^{l-1} - (q_{l-1}^T \hat{a}_{k_p}^{l-1}) q_{l-1}, \quad l = 1, 2, ..., p, \tag{3}$$

and normalization as

$$q_p = \frac{\hat{a}_{k_p}^p}{\|\hat{a}_{k_p}^p\|}, \tag{4}$$

we can write

$$P_{S_p} = P_{S_{p-1}} + q_p q_p^T. \tag{5}$$

The residual b_p is updated as

$$b_p = P_{S_p}^{\perp} b_{p-1} = b_{p-1} - (q_p^T b_{p-1}) q_p. \tag{6}$$

The algorithm terminates when either $p = r$, or $\|b_p\| \leq \epsilon$.

2.2. Tree-Search Based Orthogonal Matching Pursuit Algorithm

The tree-search based orthogonal matching pursuit (TB-OMP) algorithm is proposed in [1]. In this algorithm, the best matching vector indices $\{k_p^{(1)}, k_p^{(2)}, \ldots, k_p^{(L)}\}$ at the p^{th} iteration are selected according to

$$k_p^{(i)} = \arg\max_l |a_l^T P_{S_{p-1}}^{\perp} b|,$$
$$l \neq \{k_p^{(1)}, k_p^{(2)}, \ldots, k_p^{(i-1)}\},\ i = 1, \ldots, L. \tag{7}$$

Each of these vectors represents one alternative to be explored in each of the branches for the current partial solution. This algorithm follows the same basic iterations as OMP, but explores L choices for the next vector selected at each iteration. At the end of r iterations, the search grows exponentially to a tree with L^r leaves. The leaf corresponding to the smallest residual error vector yields the solution.

3. FLEXIBLE TREE-SEARCH BASED ORTHOGONAL MATCHING PURSUIT ALGORITHM

In this section, we introduce an efficient tree-search based OMP algorithm with branch pruning: the flexible tree-search based orthogonal matching pursuit (FTB-OMP) algorithm. Let us first introduce the decaying parameter $d \geq 1$, which affects the tree structure by reducing the branching factor at each iteration. A maximum of L branches are searched at each partial solution. In the initial iteration, the branching factor is set to L; at the i^{th} iteration the branching factor is set to $\lceil L/d^i \rceil$, where $\lceil \cdot \rceil$ represents the ceiling function. The idea in this algorithm is to start the search with a large number of branches at the initial iteration where an erroneous selection is more likely to appear, and to reduce the branching factor as the number of iterations increases.

Further reduction on the tree size is achieved by the correlation threshold $0 \leq \xi \leq 1$. Our objective is to prune the tree branches that are heuristically believed to be unnecessary. Our heuristic is to keep only the branches among $k_p^{(1)}, k_p^{(2)}, \ldots, k_p^{(L)}$ which are closely "aligned" with OMP's first choice branch $k_p^{(1)}$. We measure this alignment by the correlation between vectors which is defined as

$$\rho_{ij} = \frac{\langle a_i, a_j \rangle}{\|a_i\| \, \|a_j\|}. \tag{8}$$

A branch is assumed to be unnecessary when the candidate vector is not aligned with $k_p^{(1)}$, that is when $|\rho_{k_p^{(1)}, k_p^{(i)}}| < \xi$.

A pseudocode for FTB-OMP is given in Table 1. Note that FTB-OMP is a generalization of both OMP and TB-OMP algorithms. By choosing $\xi = 1$, we require full alignment so that only $k_p^{(1)}$ is kept, reproducing OMP. By choosing $\xi = 0$, and $d = 1$, we place no restriction on alignment,

reproducing TB-OMP. A value $0 < \xi < 1$ represents a compromise between the number of tree nodes for OMP ($r+1$ nodes), and for TB-OMP ($\frac{L^{r+1}-1}{L-1}$ nodes).

Table 1. Pseudo-code for FTB-OMP

$FTB\text{-}OMP(d, p, r, L, \xi, \epsilon)$

Global $K = [k_1, k_2, \ldots]$, Best_res, Best_k
  Calculate b_{p-1} as in (6)
  If $\|b_{p-1}\|$ <Best_res
    Best_k=$[k_1, \ldots, k_{p-1}]$
    Best_res$\leftarrow \|b_{p-1}\|$
  end
  If $p > r$ or $\|b_{p-1}\| < \epsilon$, then **return**
  Calculate $\{k_p^{(1)}, k_p^{(2)}, \ldots, k_p^{(L)}\}$ as in (7)
  For each $i = 1$ to L do
  If $|\rho_{k_p^{(1)}, k_p^{(i)}}| \geq \xi$
    $k_p = k_p^{(i)}$
    $FTB\text{-}OMP(d, p+1, r, \lceil L/d \rceil, \xi, \epsilon)$
  end
end

4. EXPERIMENTAL RESULTS

In order to compare the approximation performance and the search tree sizes of the proposed algorithm, various parameter combinations are considered. The experiments are run using MATLAB on Windows XP environment of 2.4 GHZ CPU, and 1024 MB of RAM.

In order to compare the algorithm complexities, we considered the number of search nodes and computer running times averaged over 1000 runs. Under a different environment a faster implementation is possible by employing faster machines and lower level programming languages.

Table 2. Component detection experiment with various L and d values for $\xi = 0$.

	TB-OMP	FTB-OMP			OMP
L	3	8	16	32	1
d	1	2	4	4	1
ξ	0	0	0	0	0
ncd-0	0.6	1.3	1.3	0.6	15.2
ncd-1	0.7	1.7	1.4	1.1	6.9
ncd-2	0.6	0.9	1	0.6	4
ncd-3	0.5	0.9	0.8	0.6	2.9
ncd-4	1.1	1.1	1.2	1	2.4
ncd-5	0.7	0.9	0.8	0.7	2.1
ncd-6	1.6	1.3	1.4	1.2	1.4
ncd-7	0.4	0.6	0.6	0.4	0.6
ncd-8	93.8	91.3	91.5	93.8	64.5
t_{avg}	3.5653	0.16788	0.20186	1.3095	0.011263
lv_{avg}	6561	64	64	512	1
N_{avg}	9841	425	465	3391	9
MSE	0.01093	0.01502	0.015076	0.010181	0.10609

Table 3. Component detection experiment with various ξ values for $L = 32$, $d = 4$.

	FTB-OMP				
L	32	32	32	32	32
d	4	4	4	4	4
ξ	0.05	0.1	0.15	0.2	0.8
ncd-0	0.7	0.8	1.1	1.5	15.2
ncd-1	1.1	1.2	1.4	1.6	6.9
ncd-2	0.6	0.7	0.7	0.7	4
ncd-3	0.8	0.8	0.9	1.1	2.9
ncd-4	1	1	1.2	1.4	2.4
ncd-5	0.7	0.7	0.8	1.1	2.1
ncd-6	1.2	1.2	1.5	1.5	1.4
ncd-7	0.3	0.5	0.6	0.4	0.6
ncd-8	93.6	93.1	91.8	90.7	64.5
t_{avg}	1.0945	0.81992	0.54259	0.32489	0.01283
lv_{avg}	396.36	286.27	182.21	109.4	1
N_{avg}	2615.2	1902.1	1224.1	743.98	9
MSE	0.010562	0.011431	0.013137	0.015788	0.10609

Table 4. Component detection experiment with various ξ values for $L = 3$, $d = 1$.

	FTB-OMP				
L	3	3	3	3	3
d	1	1	1	1	1
ξ	0.05	0.1	0.15	0.2	0.8
ncd-0	0.8	0.8	1	1.7	15.2
ncd-1	1	1	1.1	1.7	6.9
ncd-2	0.6	0.8	0.7	0.8	4
ncd-3	0.5	0.5	0.4	0.4	2.9
ncd-4	1.1	1.2	1.3	1.7	2.4
ncd-5	0.7	0.8	0.9	1	2.1
ncd-6	1.5	1.7	1.8	1.7	1.4
ncd-7	0.5	0.6	0.8	0.5	0.6
ncd-8	93.3	92.6	92	90.5	64.5
t_{avg}	2.2583	1.3115	0.65108	0.32892	0.011015
lv_{avg}	3893.8	2022.5	868.99	367.6	1
N_{avg}	6037.2	3283.6	1508.5	692.17	9
MSE	0.01151	0.012895	0.013893	0.017443	0.10609

The dictionary is created as a 32×128 matrix whose components are independent Gaussian random variables with mean 0 and variance 1. A sparse solution for $\mathbf{x}$ is created using 8 dictionary components with random amplitudes. Gaussian noise is then added to $\mathbf{x}$ so that the signal to noise ratio (SNR) is 40 dB. The parameter ϵ is selected as 10^{-10}. The algorithms are run for 1000 distinct input vectors, and the correctly detected number of components are counted. The average running times of the algorithms and average number of nodes searched are evaluated. The experiment results are given in Tables 2 to 4. In these tables, ncd-i represents the average percentage that exactly i components are identified correctly, t_{avg} represents the average computer running time in seconds, and MSE represents the average residual approximation error after 8^{th} iteration. The average number of searched nodes and the average number of leaves in the search tree are represented by lv_{avg} and N_{avg}, respectively.

In Table 2, various L and d values are investigated with

no correlation based pruning ($\xi = 0$). From the table, we observe that the OMP algorithm has poor detection capability, since only 64.5% of the time all the components are correctly detected. However, OMP requires a low computer running time since for 8 iterations only 9 nodes are investigated. The FTB-OMP algorithm performance with $L = 3$, $d = 1$, and $\xi = 0$ is also tabulated. We should note that with the selected set of parameters, this algorithm is equivalent to TB-OMP with $L = 3$. Due to the high computational complexity, $L > 3$ and $d = 1$ cases are not reported. The TB-OMP algorithm with $L = 3$ detects all components correctly at a rate of 93.8%, by searching 9841 nodes. From the results we can conclude that although a good detection performance is achieved with TB-OMP, the tree size is very large. For the case with $L = 32$, $d = 4$ only 3391 nodes are searched on the average and the same detection performance as TB-OMP is observed. From the results we can deduce that a better approximation performance can be obtained by keeping L large at the initial iterations.

In order to observe the effect of correlation threshold-based pruning we considered $L = 32$, $d = 4$ and $L = 3$, $d = 1$ as two distinct parameter sets. The results are given in Tables 3 and 4, respectively. In Table 3, component detection experiment results are given with various ξ values for $L = 32$, $d = 4$. From the results, we observe that computer running time and average number of nodes decrease as ξ increases, as expected. For the case where $\xi \geq 0.8$, the algorithm becomes OMP with single branching factor. The threshold value 0.1 gives a good component detection performance with ncd-8$= 93.1\%$. The algorithm searches through only 1902 nodes by taking approximately 0.8 seconds on the average. This gives a reduction of 38% in running time with respect to the case where $\xi = 0$. Therefore, we can conclude that correlation-based pruning reduces the tree size and computer running times of the algorithm, and gives comparable detection and approximation performance.

The effect of correlation based pruning in TB-OMP algorithm for $L = 3$ and $d = 1$ case is investigated in Table 4. Similarly to Table 3, the TB-OMP algorithm converges to OMP for $\xi \geq 0.8$. By changing the correlation threshold value between 0 and 0.8, a smooth transition is observed between the TB-OMP and OMP algorithms. As ξ decreases, the average approximation error decreases, while the detection performance, the average number of searched nodes and the average computer running times increase.

5. CONCLUSIONS

In this paper, we proposed a novel efficient tree-search based orthogonal matching pursuit algorithm for sparse signal representations, called FTB-OMP. The algorithm provides some design parameters, giving flexibility to choose between higher approximation performance and lower time complexity. The efficiency is achieved by using a correlation based pruning in the search tree, and by changing the number of children at each tree node.

From the experimental results, we can conclude that FTB-OMP algorithm with high L values and $d > 1$ is more effective than TB-OMP. The novel exponential decaying structure of the search tree makes the algorithm more computationally efficient and increases the approximation and detection performance. The threshold parameter ξ can be used for further computational efficiency.

It is known that solving an overcomplete set of linear equations is a frequently encountered problem in communications and signal processing areas. The proposed FTB-OMP algorithm is a suitable technique to solve problems where sparse signal representations are required, such as the angle of arrival detection [6] and channel estimation [7] problems.

6. REFERENCES

[1] S.F. Cotter and B.D. Rao, "Application of tree-based searches to matching pursuit," in *IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2001, vol. 6, pp. 3933–3936.

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

[3] A.M. Kondoz, *Digital speech*, Wiley, 1996.

[4] B.K. Natarajan, "Sparse approximate solutions to linear systems," *SIAM Journal on Computing*, vol. 24, no. 2, pp. 227–234, Apr. 1995.

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

[6] G. Karabulut, T. Kurt, and A. Yongaçoglu, "Angle of arrival detection by matching pursuit algorithm," in *Vehicular Technology Conference, Fall*, 2004.

[7] S.F. Cotter and B.D. Rao, "The adaptive matching pursuit algorithm for estimation and equalization of sparse time-varying channels," in *Asilomar Conference on Signals, Systems and Computers*, 2000, vol. 2, pp. 1772–1776.

[8] J. Adler, B.D. Rao, and K. Kreutz-Delgado, "Comparison of basis selection methods," in *Asilomar Conference on Signals, Systems and Computers*, 1996, vol. 1, pp. 252–257.