Signal Processing 93 (2013) 3143–3150

Contents lists available at SciVerse ScienceDirect

Signal Processing

journal homepage: www.elsevier.com/locate/sigpro

# Adaptive matching pursuit using coordinate descent and double residual minimization ☆

Alexandru Onose [a], Bogdan Dumitrescu [a,b,*]

[a] Department of Signal Processing, Tampere University of Technology, Finland
[b] Department of Automatic Control and Computers, University Politehnica of Bucharest, Romania

ARTICLE INFO

*Article history:*
Received 10 August 2012
Received in revised form
12 March 2013
Accepted 3 May 2013
Available online 10 May 2013

*Keywords:*
Sparse filters
Matching pursuit
Adaptive algorithm
Greedy method
Channel identification

ABSTRACT

We present a greedy recursive algorithm for computing sparse solutions to systems of linear equations. Derived from adaptive matching pursuit, the algorithm employs a greedy column selection strategy which, combined with coefficient update via coordinate descent, ensures a low complexity. The sparsity level is estimated online using the predictive least squares (PLS) criterion. The key to performance is the minimization of two residuals, corresponding to two solutions with different sparsity levels, one for finding the values of the nonzero coefficients, the other for maintaining a large enough pool of candidates for the PLS criterion. We test the algorithm for a sparse time-varying finite impulse response channel; the performance is comparable with or better than that of the competing methods, while the complexity is lower.



## 1. Introduction

In recent years numerous problems like compression, prediction, array processing, channel identification or echo cancellation have generated much interest in the development of algorithms for recursively finding sparse solutions to systems of linear [1,2]. The search for methods that produce sparse solutions began with the development of batch algorithms like the basis pursuit [3], the least absolute shrinkage and selection operator [4] or the orthogonal least squares [5]. In practice, however, the data are often available sequentially and thus adaptive algorithms that compute the solution recursively are much more efficient. The different adaptive methods proposed range from convex relaxation techniques [6–8] to algorithms based on projections onto convex sets [9] or greedy methods [10,11]. With roots in the traditional adaptive filtering, sparsity aware LMS algorithms have also been developed [12,13].

The aim of this paper is to present a new adaptive greedy algorithm that uses a selection strategy inspired from [11], but now coupled with coefficient update based on coordinate descent, thus having low complexity without negatively affecting the performance.

Let us consider a typical finite impulse response (FIR) channel identification problem where the input $u(t)$ and the output $d(t)$ are known at each time instance $t$. We desire to estimate the true coefficients $h_j$ that result from the minimization of the estimation error

$$e(t) = d(t) - \sum_{j=0}^{N-1} h_j u(t-j). \tag{1}$$

Using an exponential window with forgetting factor $0 < \lambda \leq 1$, the least-squares criterion to be minimized is

$$J(t) = \|\boldsymbol{b}^{(t)} - \boldsymbol{A}^{(t)}\boldsymbol{x}^{(t)}\|^2. \tag{2}$$

The solution $\boldsymbol{x}^{(t)}$ resulting from the minimization of $J(t)$ is the estimate of the vector $\boldsymbol{h}$ of true coefficients, which is

☆ This work has been supported by a Tekes FiDiPro grant and by GETA.
* Corresponding author at. Department of Signal Processing, Tampere University of Technology, Korkeakoulunkatu 1, 33720 Tampere, Finland.
*E-mail addresses:* Alexandru.Onose@tut.fi (A. Onose), bogdan.dumitrescu@tut.fi, bogdan.dumitrescu@acse.pub.ro (B. Dumitrescu).

0165-1684/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.sigpro.2013.05.001

assumed to be sparse, with $L_t \ll N$ nonzero elements. The matrix $\boldsymbol{A}^{(t)} \in \mathbb{R}^{t \times N}$, constructed using the input data, has the $i$-th row equal to $\lambda^{(t-i)/2}\boldsymbol{\alpha}^{(i)T}$ with

$$\boldsymbol{\alpha}^{(i)} = [u(i)\ u(i-1) \ldots u(i-N+1)]^T. \tag{3}$$

The vector $\boldsymbol{b}^{(t)} \in \mathbb{R}^t$ is composed of the exponentially weighted output data $\boldsymbol{b}_i = \lambda^{(t-i)/2} d(i)$. The index $^{(i)}$ is used to indicate the data from time $i$; we define $\beta^{(i)} = d(i)$ for later use. We presume the environment to be slowly varying such that the solution does not change greatly between two consecutive time instances.

We develop an adaptive algorithm for finding a sparse minimizer of the criterion (2). The most remote ancestors of our algorithm are the matching pursuit (MP) algorithm [14] and its adaptive counterpart, the adaptive matching pursuit (AMP) [10]. The solution is updated at each time $t$ via coordinate descent. This technique is used in other adaptive algorithms; it can be tailored for finding either full solutions like in [15] or sparse solutions, for example in [8]. A related algorithm is the (batch) cyclic MP [16,17], which uses several rounds of coordinate descent at each step of the greedy search. We have presented an adaptive version of cyclic MP [18]; however, the current algorithm is quite different in the organization of the computation and does not repeat the optimization of a coefficient at the same time instance, but spreads it over time; hence, its complexity is lower.

Since we use some techniques from the GRLS algorithm [11], we discuss here the differences and similarities between this work and [11]. Our new algorithm requires a pseudo ordering of the columns from $\boldsymbol{A}$ that contribute to the solution; any method providing such an order may be used but we only present two approaches. The first is inspired from [11] and uses neighbor permutations, while the second, simpler, only tries to find the worst column. The neighbor selection strategy is used in [11] for a completely different underlying algorithm and it is one of the few viable methods that allow low complexity, while for the algorithm herein such restrictions do not apply.

We estimate online the sparsity level $L_t$ using the predictive least squares criterion (PLS) [19], like in [11], although other model selection criteria exist [20,21] and the Bayesian information criterion [20] is also used in [11]. However, the mechanism that is created to allow the use of PLS is new and is specifically tailored for the algorithm presented here. A distinctive feature is the minimization of two residuals related to criterion (2), corresponding to solutions with different sparsity levels; this also implies some modifications in the column selection strategy, compared to [11]. Finally, the coefficient estimation strategy is completely different from the orthogonalization from [11]; here, the coordinate descent ensures a good approximation of the least-squares solution, with significantly lower complexity than in [11]; the only similarity is that we use a fixed number of scalar products instead of the indefinitely long matrix $\boldsymbol{A}$ and vector $\boldsymbol{b}$, but this is a standard storage technique.

The contents of this paper is as follows. In Section 2 we present our proposed algorithm for computing a sparse solution with fixed sparsity level. In Section 3 we describe our procedure based on the minimization of two residuals for the online estimation of the sparsity level $L_t$ and the computation of the solution. In Section 4 we discuss the complexity of our algorithm, compared to that of other algorithms. Section 5 contains the simulation results used to validate the performance.

## 2. Fixed-sparsity level adaptive matching pursuit with coordinate descent

We describe the basic operation of our algorithm assuming that, at time $t$, we have computed an $m$-sparse solution $\boldsymbol{x}^{(t)}$, permuted such that its nonzero elements are in the first $m$ positions, which are named active. The columns of the matrix $\boldsymbol{A}^{(t)}$ are permuted accordingly; we will not show the permutation explicitly, but only explain how it changes. The value $m \geq L_t$ is assumed to be fixed in this section. The residual corresponding to the solution $\boldsymbol{x}^{(t)}$ is

$$\boldsymbol{r}_m^{(t)} = \boldsymbol{b}^{(t)} - \sum_{i=1}^{m} x_i^{(t)} \boldsymbol{a}_i^{(t)}, \tag{4}$$

where $\boldsymbol{a}_i^{(t)}$ is the $i$-th column of (the permuted) $\boldsymbol{A}^{(t)}$.

At time $t+1$, the new (permuted) data are appended

$$\boldsymbol{b}^{(t+1)} = \begin{bmatrix} \sqrt{\lambda}\boldsymbol{b}^{(t)} \\ \beta^{(t+1)} \end{bmatrix}, \quad \boldsymbol{A}^{(t+1)} = \begin{bmatrix} \sqrt{\lambda}\boldsymbol{A}^{(t)} \\ \boldsymbol{\alpha}^{(t+1)^T} \end{bmatrix}. \tag{5}$$

To ease the notation, from now on we remove the upper index $t+1$ that marks the variables affected by current computations. Using the solution from time $t$, the new residual is

$$\boldsymbol{r}_{m,0} = \boldsymbol{b} - \sum_{i=1}^{m} x_i^{(t)} \boldsymbol{a}_i = \begin{bmatrix} \boldsymbol{r}_m^{(t)} \\ \beta - \sum_{i=1}^{m} x_i^{(t)} \alpha_i \end{bmatrix}. \tag{6}$$

At time $t+1$ we perform two tasks: (i) we manage the order of the active positions and (ii) we update the values of the solution $\boldsymbol{x}$. Task (i) is typical to greedy algorithms and MP in particular; it aims to identify the nonzero coefficients and order them based on their importance. To reduce complexity and relying on the slow variability of the channel, we use for now the search scheme from [11] that allows order changes almost only between neighbors; the only exception is the last position in the active set, for which we permit all the inactive columns to compete with the current active column occupying that position. Task (ii) is performed by a simple coordinate descent on each of the active positions, optimizing the residual corresponding to the $m$-sparse solution. The two tasks are intertwined and act successively on each active coefficient.

To update the coefficient $i$, with $i < m$, we compute a partial residual with the two neighbor positions $i$ and $i+1$ temporarily removed from the active set

$$\tilde{\boldsymbol{r}}_{m,i} = \boldsymbol{r}_{m,i-1} + x_i^{(t)} \boldsymbol{a}_i + x_{i+1}^{(t)} \boldsymbol{a}_{i+1}. \tag{7}$$

The column that takes position $i$ is decided by looking at the best alignment with this residual, i.e. with the standard MP criterion

$$k = \arg\max_{l \in \mathcal{S}} \frac{|\boldsymbol{a}_l^T \tilde{\boldsymbol{r}}_{m,i}|^2}{\|\boldsymbol{a}_l\|^2}, \tag{8}$$

with $\mathcal{S} = \{i, i+1\}$. If position $i+1$ is better, then positions $i$ and $i+1$ are permuted. The optimization of coefficient $x_i$ is a